

**RANCANG BANGUN *BACKEND* WEB
UNTUK PEMESANAN MAKANAN *ONLINE*
DI KANTIN MENGGUNAKAN NODE.JS
(STUDI KASUS: KANTIN UNIVERSITAS PASUNDAN
SETIABUDI)**

TUGAS AKHIR

Disusun sebagai salah satu syarat untuk kelulusan
Program Strata 1, Program Studi Teknik Informatika,
Universitas Pasundan Bandung

oleh :

Putri Legiani Fahreza
nrp. 21.304.0039



**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS PASUNDAN BANDUNG
SEPTEMBER 2025**

LEMBAR PENGESAHAN
LAPORAN TUGAS AKHIR

Telah diujikan dan disahkan Tugas Akhir, dari :

Nama : Putri Legiani Fahreza

Nrp. : 21.304.0039

Dengan judul :

“Rancang Bangun *Backend* Web Untuk Pemesanan Makanan *Online* Di Kantin Menggunakan
Node.Js (Studi Kasus: Kantin Universitas Pasundan Setiabudi)”

Bandung, 25 September 2025

Menyetujui,
Pembimbing utama



(Dr. Ayi Purbasari, S.T., M.T.)

LEMBAR PERNYATAAN KEASLIAN
LAPORAN TUGAS AKHIR

Saya menyatakan dengan sesungguhnya bahwa:

1. Tugas akhir ini merupakan benar-benar asli dan belum pernah diajukan untuk mendapatkan gelar akademik, baik di Universitas Pasundan maupun di Perguruan Tinggi lainnya.
2. Tugas akhir ini merupakan gagasan, rumusan dan penelitian dari saya sendiri, tanpa bantuan pihak lain, kecuali arahan dari tim Dosen Pembimbing.
3. Dalam tugas akhir ini tidak terdapat karya atau pendapat orang lain, kecuali pada bagian-bagian tertentu dalam penulisan laporan tugas akhir yang saya kutip dari hasil karya orang lain yang telah dituliskan dalam sumbernya secara jelas sesuai dengan norma, kaidah, dan etika penulisan karya ilmiah, serta disebutkan dalam daftar pustaka pada tugas akhir ini
4. Kakas, perangkat lunak, dan alat bantu kerja lainnya yang digunakan dalam penelitian ini adalah sepenuhnya tanggung jawab saya sendiri dan bukan tanggung jawab Universitas Pasundan. Apabila di kemudian hari ditemukan seluruh atau sebagian laporan tugas akhir ini bukan hasil karya saya sendiri atau adanya plagiasi dalam bagian tertentu, saya bersedia menerima sanksi akademi, termasuk pencabutan gelar akademik yang saya sandang sesuai dengan norma yang berlaku di Universitas Pasundan, serta perundang-undangan lainnya.

Bandung, 25 September 2025

Yang membuat pernyataan,

(Putri Legiani Fahreza)

NRP. 21.304.0039

ABSTRAK

Perkembangan teknologi mendorong berbagai sektor untuk terus berinovasi, termasuk dalam sistem pemesanan makanan di lingkungan kampus. Kantin Universitas Pasundan Setiabudi saat ini masih menggunakan sistem pemesanan manual yang kurang efisien, terutama pada saat jam sibuk. Oleh karena itu, diperlukan suatu sistem yang dapat mempermudah dan mempercepat proses pemesanan.

Penelitian ini bertujuan untuk merancang dan membangun sistem dari website pemesanan makanan *online* di kantin dengan menggunakan Node.js dan Express.js. Sistem ini menyediakan REST API yang menjadi penghubung antara antarmuka pengguna dengan sisi *server*. Metode perancangan yang digunakan adalah metode Scrum, melalui tahapan perencanaan, pengumpulan kebutuhan (melalui wawancara dengan penjual dan pembeli di kantin), serta studi literatur terkait teknologi yang digunakan.

Hasil dari penelitian ini adalah sebuah sistem untuk aplikasi pemesanan makanan di kantin kampus yang telah berhasil dibangun dan diuji menggunakan metode *black box testing*. Sebagian besar fungsionalitas berjalan sesuai dengan skenario yang dirancang, mencakup proses pemesanan, verifikasi pembayaran, pencatatan transaksi, serta penyusunan laporan, meskipun masih terdapat beberapa hal yang perlu disempurnakan.

Kata kunci: Kantin Kampus, , Pemesanan Makanan *Online*, Node.Js, REST API, Scrum

ABSTRACT

The rapid development of technology has encouraged various sectors to continuously innovate, including food ordering systems in campus environments. The canteen at Universitas Pasundan Setiabudi currently still relies on a manual ordering method, which is less efficient, especially during peak hours. Therefore, a system is needed to simplify and accelerate the ordering process.

This study aims to design and develop the *backend* of an *online* food ordering website for the campus canteen using Node.js and Express.js. The system provides a REST API that serves as a bridge between the user interface and the *server* side. The system design was carried out using the Scrum methodology through several stages, namely planning, requirements gathering (through interviews with canteen sellers and buyers), and literature studies related to the applied technologies.

The result of this research is a *backend* system for a campus canteen food ordering application that has been successfully developed and tested using the black box testing method. Most functionalities worked according to the designed scenarios, covering ordering, payment verification, transaction recording, and report generation, although several aspects still require further refinement.

Kata kunci: Kantin Kampus, , Pemesanan Makanan *Online*, Node.Js, REST API, Scrum

KATA PENGANTAR

Ucapan dan rasa syukur penulis panjatkan ke hadirat Allah SWT, karena atas rahmat dan karunia-Nya, penulis dapat menyelesaikan Tugas Akhir dengan judul “Rancang Bangun *backend* Web Untuk Pemesanan Makanan *Online* di Kantin Menggunakan Node.js (Studi Kasus: Kantin Universitas Pasundan Setiabudi). Adapun penulisan laporan ini bertujuan untuk memenuhi salah satu syarat kelulusan Tugas Akhir, di Program Studi Teknik Informatika Universitas Pasundan.

Penulis menyadari laporan ini dapat terwujud berkat bantuan dan dorongan dari berbagai pihak. Maka pada kesempatan ini penulis sampaikan terima kasih yang sebesar-besarnya atas segala bantuan yang penulis terima baik secara moril maupun materil, sehingga penulis dapat menyelesaikan laporan ini kepada:

1. Allah SWT, atas segala Rahmat, nikmat, dan kemudahan yang diberikan selama proses pengerjaan tugas akhir ini.
2. Kedua orang tua tercinta, yang selalu memberikan doa, kasih sayang, serta dukungan moril dan materil.
3. Dr. Ayi Purbasari, S.T, M.T. selaku Dosen Pembimbing, yang telah meluangkan waktu, memberikan bimbingan dan arahan selama proses penyusunan tugas akhir ini.
4. Bapak/Ibu dosen penguji dan seluruh dosen Program Studi Teknik Informatika Universitas Pasundan, yang telah memberikan ilmu dan wawasan selama masa perkuliahan.
5. Semua pihak yang tidak dapat penulis sebutkan satu persatu, yang telah membantu secara langsung maupun tidak langsung dalam penyusunan tugas akhir ini.

Dalam penulisan laporan akhir ini, penulis menyadari bahwa masih terdapat banyak kesalahan dan kekeliruan. Oleh karena itu, penulis mengharapkan saran dan kritik yang bersifat membangun demi perbaikan di masa yang akan datang. Semoga laporan akhir ini dapat bermanfaat bagi semua pihak yang memerlukan

Bandung, 25 September 2025

Penulis

DAFTAR ISI

ABSTRAK	i
ABSTRACT	ii
KATA PENGANTAR.....	iii
DAFTAR ISI	iv
DAFTAR ISTILAH.....	x
DAFTAR TABEL.....	vi
DAFTAR GAMBAR.....	x
DAFTAR LAMPIRAN	xiii
DAFTAR SIMBOL	xiv
BAB 1 PENDAHULUAN	1-1
1.1 Latar Belakang.....	1-1
1.2 Identifikasi Masalah.....	1-2
1.3 Tujuan Tugas Akhir.....	1-3
1.4 Lingkup Tugas Akhir	1-3
1.5 Metodologi Tugas Akhir	1-3
1.6 Sitematika Penulisan.....	1-5
BAB 2 LANDASAN TEORI DAN PENELITIAN TERDAHULU.....	2-1
2.1 Teori Pendukung.....	2-1
2.1.1 Javascript.....	2-1
2.1.2 Node.js	2-1
2.1.3 Express JS	2-2
2.1.4 REST API.....	2-2
2.1.5 PostgreSQL	2-2
2.1.6 <i>Backend</i>	2-3
2.1.7 Website.....	2-3
2.1.8 Scrum	2-3
2.1.9 Kantin.....	2-5
2.1.10 Pemesanan.....	2-5
2.2 Penelitian Terdahulu	2-6
BAB 3 SKEMA PENELITIAN	3-1
3.1 Alur Penyelesaian Tugas Akhir.....	3-1
3.2 Perumusan Masalah	3-2
3.2.1 Analisis Sebab Akibat	3-2
3.2.2 Analisis Solusi	3-3

3.3	Profil Penelitian	3-4
3.3.1	Deskripsi Singkat	3-5
3.3.2	Struktur Organisasi	3-5
BAB 4	ANALISIS DAN PERANCANGAN	4-1
4.1	Analisis Kebutuhan Perangkat Lunak.....	4-1
4.1.1	Analisis Sistem Saat Ini	4-1
4.1.2	Analisis Kebutuhan Pengguna	4-2
4.1.3	Analisis Kebutuhan Fungsional	4-3
4.1.4	Analisis Kebutuhan Non Fungsional	4-3
4.1.5	Pemodelan Berbasis Skenario.....	4-3
4.2	Perancangan Kebutuhan Perangkat Lunak	4-39
4.2.1	Pemodelan Data	4-40
4.2.2	Pemodelan Berbasis Kelas	4-42
4.2.3	Pemodelan Berbasis Perilaku.....	4-53
4.2.4	Perancangan Arsitektur Aplikasi	4-76
BAB 5	SCRUM PROSES DAN DELIVERY	5-1
5.1	Scrum Proses	5-1
5.1.1	Product backlog	5-5
5.1.2	Sprint ke-1	5-5
5.1.3	Sprint ke-2.....	5-10
5.1.4	Sprint ke-3.....	5-14
5.1.5	Sprint ke-4.....	5-21
5.2	Delivery	5-27
5.2.1	Pengujian perangkat lunak	5-27
5.2.2	Deployment.....	5-59
BAB 6	KESIMPULAN DAN SARAN	6-1
6.1	Kesimpulan	6-1
6.2	Saran	6-1
	DAFTAR PUSTAKA	xv

DAFTAR TABEL

Tabel 2.1 Penelitian Terdahulu	2-6
Tabel 3.1 Alur Pengerjaan Tugas Akhir.....	3-1
Tabel 3.2 Detail Fishbone Sebab Akibat	3-3
Tabel 3.3 Detail Fishbone Analisis Solusi.....	3-4
Tabel 4.1 Diagram Swimlane Analisis Sistem Saat Ini	4-2
Tabel 4.2 User Requirement	4-2
Tabel 4.3 Software Requirement	4-3
Tabel 4.4 Analisis Kebutuhan Non Fungsional	4-3
Tabel 4.5 Deskripsi Aktor.....	4-4
Tabel 4.6 Diagram Use Case Pembeli	4-4
Tabel 4.7 Diagram Use Case Pembeli	4-5
Tabel 4.8 Diagram Use Case Penjual	4-5
Tabel 4.9 Deskripsi Diagram Use Case Pembeli.....	4-6
Tabel 4.10 Deskripsi Diagram Use Case Penjual.....	4-6
Tabel 4.11 Menampilkan Daftar Kios	4-6
Tabel 4.12 Menampilkan Daftar Menu Per Kios.....	4-7
Tabel 4.13 Menampilkan Detail Menu.....	4-8
Tabel 4.14 Mencari Menu	4-8
Tabel 4.15 Mengelola Keranjang	4-9
Tabel 4.16 Membuat Pesanan Dari Keranjang	4-10
Tabel 4.17 Menyediakan Payment Gateway	4-11
Tabel 4.18 Menampilkan Status Dan Estimasi Waktu Pesanan	4-12
Tabel 4.19 Mengirim Notifikasi Pesanan Selesai.....	4-13
Tabel 4.20 Mengelola Menu.....	4-13
Tabel 4.21 Mengirim Notifikasi Pesanan Masuk	4-14
Tabel 4.22 Menampilkan Daftar Pesanan Dan Detail Pesanan	4-15
Tabel 4.23 Mengubah Status Pesanan	4-16
Tabel 4.24 Menampilkan Tipe Pengantaran	4-16
Tabel 4.25 Diagram Aktifitas UC-01	4-17
Tabel 4.26 Diagram Aktivitas UC-02.....	4-18
Tabel 4.27 Diagram Aktivitas Uc-03.....	4-19
Tabel 4.28 Diagram Aktifitas Uc-04a.....	4-20
Tabel 4.29 Diagram Aktifitas Uc-04b	4-21

Tabel 4.30 Diagram aktivitas uc-05a.....	4-22
Tabel 4.31 Diagram aktivitas uc-05b	4-23
Tabel 4.32 Diagram aktivitas uc-05c.....	4-24
Tabel 4.33 Diagram aktivitas uc-05d	4-25
Tabel 4.34 Diagram aktivitas uc-06	4-26
Tabel 4.35 Diagram aktivitas uc-07	4-27
Tabel 4.36 Diagram aktivitas uc-08	4-28
Tabel 4.37 Diagram aktivitas uc-09	4-29
Tabel 4.38 Diagram Aktivitas Uc-10a.....	4-30
Tabel 4.39 Diagram Aktivitas Uc-10b.....	4-31
Tabel 4.40 Diagram Aktivitas Uc-10c.....	4-32
Tabel 4.41 Diagram Aktivitas Uc-10d.....	4-33
Tabel 4.42 Diagram Aktivitas Uc-10e.....	4-34
Tabel 4.43 Diagram Aktivitas Uc-11	4-35
Tabel 4.44 Diagram Aktivitas Uc-12a.....	4-36
Tabel 4.45 Diagram Aktivitas Uc-12b.....	4-37
Tabel 4.46 Diagram Aktivitas Uc-13.....	4-38
Tabel 4.47 Diagram Aktivitas Uc-14.....	4-39
Tabel 4.48 Object Data.....	4-40
Tabel 4.49 Atribut Data	4-40
Tabel 4.50 Kelas-Kelas Analisis.....	4-43
Tabel 4.51 Atribut-Atribut.....	4-43
Tabel 4.52 Operasi-Operasi.....	4-48
Tabel 4.53 Diagram Kelas.....	4-53
Tabel 4.54 Diagram Sekuens Uc-01	4-54
Tabel 4.55 Diagram Sekuens Uc-02.....	4-55
Tabel 4.56 Diagram Sekuens Uc-03.....	4-56
Tabel 4.57 Diagram Sekuens Uc-04.....	4-57
Tabel 4.58 Diagram Sekuens Uc-05a	4-58
Tabel 4.59 Diagram Sekuens Uc-05b.....	4-59
Tabel 4.60 Diagram Sekuens Uc-05c	4-60
Tabel 4.61 Diagram Sekuens Uc-05d.....	4-61
Tabel 4.62 Diagram Sekuens Uc-06.....	4-62
Tabel 4.63 Diagram Sekuens Uc-07.....	4-63
Tabel 4.64 Diagram Sekuens Uc-08.....	4-64

Tabel 4.65 Diagram Sekuens Uc-09.....	4-65
Tabel 4.66 Diagram Sekuens Uc-10a.....	4-66
Tabel 4.67 Diagram Sekuens Uc-10b.....	4-67
Tabel 4.68 Diagram Sekuens Uc-10c.....	4-68
Tabel 4.69 Diagram Sekuens Uc-10d.....	4-69
Tabel 4.70 Diagram Sekuens Uc-10e.....	4-70
Tabel 4.71 Diagram Sekuens Uc-11.....	4-71
Tabel 4.72 Diagram Sekuens Uc-12a.....	4-72
Tabel 4.73 Diagram Sekuens Uc-12b.....	4-73
Tabel 4.74 Diagram Sekuens Uc-13.....	4-74
Tabel 4.75 Diagram Sekuens Uc-14.....	4-75
Tabel 5.1 Kakas Perangkat Lunak.....	5-1
Tabel 5.2 Package yang Dibutuhkan.....	5-2
Tabel 5.3 Struktur Folder.....	5-3
Tabel 5.4 Konfigurasi Environtment.....	5-4
Tabel 5.5 Product Backlog.....	5-5
Tabel 5.6 Sprint Planning Ke-1.....	5-6
Tabel 5.7 Sprint Backlog Ke-1.....	5-6
Tabel 5.8 Potongan Kode Autentikasi.....	5-7
Tabel 5.9 Potongan Kode Mengelola Menu.....	5-8
Tabel 5.10 Potongan Kode Beranda.....	5-8
Tabel 5.11 Potongan Kode Menampilkan Daftar Kios.....	5-9
Tabel 5.12 Potongan Kode Menampilkan Daftar Menu Per Kios.....	5-9
Tabel 5.13 Sprint Review Ke-1.....	5-9
Tabel 5.14 Sprint planning ke-2.....	5-10
Tabel 5.15 Sprint Backlog Ke-2.....	5-10
Tabel 5.16 Potongan Kode Autentikasi (Logout).....	5-11
Tabel 5.17 Potongan Kode menampilkan detail menu.....	5-12
Tabel 5.18 Potongan Kode mengelola keranjang.....	5-12
Tabel 5.19 Potongan Kode membuat pesanan dari keranjang.....	5-13
Tabel 5.20 Potongan Kode payment gateway.....	5-13
Tabel 5.21 Potongan Kode notifikasi pesanan masuk.....	5-13
Tabel 5.22 Sprint Review ke-2.....	5-14
Tabel 5.23 Sprint Planning ke-3.....	5-14
Tabel 5.24 Sprint Backlog ke-3.....	5-15

Tabel 5.25 Potongan Kode Beranda	5-16
Tabel 5.26 Potongan Kode Mengubah status pesanan	5-16
Tabel 5.27 Potongan Kode Mengelola Keranjang.....	5-16
Tabel 5.28 Membuat Pesanan dari Keranjang.....	5-18
Tabel 5.29 Menampilkan Tipe Pengantaran	5-19
Tabel 5.30 Menampilkan Detail Pesanan	5-20
Tabel 5.31 Menampilkan Status dan Estimasi Waktu Pesanan	5-20
Tabel 5.32 Sprint Review ke-3	5-21
Tabel 5.33 Sprint Planning ke-4.....	5-21
Tabel 5.34 Sprint Backlog ke-4.....	5-22
Tabel 5.35 Potongan Kode Notifikasi Payment Gateway	5-23
Tabel 5.36 Potongan Kode Cloudinary	5-23
Tabel 5.37 Potongan Kode Notifikasi Pesanan Masuk	5-24
Tabel 5.38 Potongan Kode Menampilkan Status Dan Estimasi Waktu Pesanan.....	5-24
Tabel 5.39 Potongan Kode Mengirim Notifikasi Pesanan Selesai	5-26
Tabel 5.40 Sprint Review ke-4.....	5-26
Tabel 5.41 Rekapitulasi Pengujian Perangkat Lunak	5-28
Tabel 5.42 Pengujian Autentikasi.....	5-30
Tabel 5.43 Pengujian Menu.....	5-34
Tabel 5.44 Pengujian kios	5-41
Tabel 5.45 Pengujian Keranjang	5-45
Tabel 5.46 Pengujian Pesanan	5-48
Tabel 5.47 Pengujian Midtrans (Pembayaran)	5-56
Tabel 5.48 Pengujian Dashboard data	5-57
Tabel 5.49 Pengujian Search All	5-57
Tabel 5.50 Hasil Deployment.....	5-61

DAFTAR ISTILAH

Istilah	Deskripsi
<i>Scrum</i>	Metode pengembangan perangkat lunak agile yang berfokus pada iterasi dan kolaborasi tim.
	Bagian dari sistem yang menangani logika, <i>database</i> , dan proses di sisi <i>server</i> .
<i>Frontend</i>	Antarmuka pengguna yang digunakan oleh user untuk berinteraksi dengan sistem.
<i>Node.js</i>	Lingkungan <i>runtime</i> JavaScript berbasis <i>server</i> yang digunakan untuk membuat aplikasi <i>backend</i> .
<i>Express.js</i>	<i>Framework</i> Node.js yang memudahkan dalam membangun <i>server</i> dan REST API
<i>Database</i>	Tempat penyimpanan data yang digunakan oleh sistem seperti MySQL
<i>Endpoint</i>	Titik akses dari REST API yang memungkinkan <i>frontend</i> mengakses fungsi tertentu di
<i>JSON</i>	Format data ringan yang digunakan untuk pertukaran data antar sistem
<i>HTTP</i>	Protokol komunikasi standar pada web untuk mentransfer data antara <i>client</i> dan <i>server</i> .
<i>XML</i>	Extensible Markup Language, format markup yang digunakan untuk menyimpan dan mentransfer data.
<i>Web Service</i>	Layanan berbasis web yang memungkinkan sistem saling berkomunikasi dan bertukar data.
<i>Event-Driven</i>	Pola pemrograman di mana alur eksekusi dikendalikan oleh peristiwa (<i>event</i>) seperti klik, request, dll.
<i>Non-blocking I/O</i>	Model <i>input/output</i> yang memungkinkan proses lain berjalan tanpa harus menunggu proses I/O selesai.
<i>OOP (Object Oriented Programming)</i>	Paradigma pemrograman yang berfokus pada objek dan class untuk mengorganisir kode.
<i>CRM (Customer Relationship Management)</i>	Konsep pengelolaan hubungan dengan pelanggan untuk meningkatkan kepuasan dan loyalitas.

DAFTAR GAMBAR

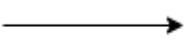
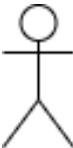
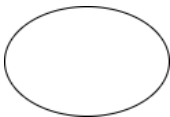
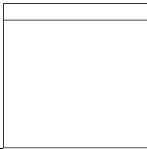
Gambar 1.1 Langkah-langkah penyelesaian tugas akhir	1-4
Gambar 3.1 Analisis Sebab Akibat.....	3-3
Gambar 3.2 Analisis Solusi	3-4
Gambar 3.3 Logo Unpas	3-4
Gambar 3.4 Struktur Organisasi Kantin	3-5
Gambar 4.1 Swimlane Analisis Sistem Saat Ini	4-2
Gambar 4.2 Diagram Use Case Pembeli	4-4
Gambar 4.3 Diagram Use Case Pembeli	4-5
Gambar 4.4 Diagram Use Case Penjual	4-5
Gambar 4.5 Diagram Aktivitas UC-01	4-17
Gambar 4.6 Diagram Aktivitas Uc-02.....	4-18
Gambar 4.7 Diagram Aktivitas Uc-03.....	4-19
Gambar 4.8 Diagram Aktifitas Uc-04a.....	4-20
Gambar 4.9 Diagram Aktifitas Uc-04b	4-21
Gambar 4.10 Diagram aktivitas uc-05a.....	4-22
Gambar 4.11 Diagram aktivitas uc-05b.....	4-23
Gambar 4.12 Diagram aktivitas uc-05c.....	4-24
Gambar 4.13 Diagram aktivitas uc-05d	4-25
Gambar 4.14 Diagram aktivitas uc-06	4-26
Gambar 4.15 Diagram aktivitas uc-07	4-27
Gambar 4.16 Diagram aktivitas uc-08	4-28
Gambar 4.17 Diagram aktivitas uc-09	4-29
Gambar 4.18 Diagram Aktivitas Uc-10a.....	4-30
Gambar 4.19 Diagram Aktivitas Uc-10b.....	4-31
Gambar 4.20 Diagram Aktivitas Uc-10c.....	4-32
Gambar 4.21 Diagram Aktivitas Uc-10d.....	4-33
Gambar 4.22 Diagram Aktivitas Uc-10e.....	4-34
Gambar 4.23 Diagram Aktivitas Uc-11	4-35
Gambar 4.24 Diagram Aktivitas Uc-12a.....	4-36
Gambar 4.25 Diagram Aktivitas Uc-12b.....	4-37
Gambar 4.26 Diagram Aktivitas Uc-13.....	4-38
Gambar 4.27 Diagram Aktivitas Uc-14.....	4-39
Gambar 4.28 Diagram Kelas	4-53

Gambar 4.29 Diagram Sekuens Uc-01	4-54
Gambar 4.30 Diagram Sekuens Uc-02	4-55
Gambar 4.31 Diagram Sekuens Uc-03	4-56
Gambar 4.32 Diagram Sekuens Uc-04	4-57
Gambar 4.33 Diagram Sekuens Uc-05a	4-58
Gambar 4.34 Diagram Sekuens Uc-05b	4-59
Gambar 4.35 Diagram Sekuens Uc-05c	4-60
Gambar 4.36 Diagram Sekuens Uc-05d	4-61
Gambar 4.37 Diagram Sekuens Uc-06	4-62
Gambar 4.38 Diagram Sekuens Uc-07	4-63
Gambar 4.39 Diagram Sekuens Uc-08	4-64
Gambar 4.40 Diagram Sekuens Uc-09	4-65
Gambar 4.41 Diagram Sekuens Uc-10a	4-66
Gambar 4.42 Diagram Sekuens Uc-10b	4-67
Gambar 4.43 Diagram Sekuens Uc-10c	4-68
Gambar 4.44 Diagram Sekuens Uc-10d	4-69
Gambar 4.45 Diagram Sekuens Uc-10e	4-70
Gambar 4.46 Diagram Sekuens Uc-11	4-71
Gambar 4.47 Diagram Sekuens Uc-12a	4-72
Gambar 4.48 Diagram Sekuens Uc-12b	4-73
Gambar 4.49 Diagram Sekuens Uc-13	4-74
Gambar 4.50 Diagram Sekuens Uc-14	4-75
Gambar 4.51 Arsitektur Aplikasi	4-76
Gambar 5.1 Struktur Folder	5-3
Gambar 5.2 Konfigurasi .env	5-4
Gambar 5.3 Render	5-60
Gambar 5.4 Supabase	5-60

DAFTAR LAMPIRAN

LAMPIRAN A BERITA ACARA WAWANCARA	A-1
LAMPIRAN B DOKUMENTASI	B-1
LAMPIRAN C DOKUMENTASI SPRINT.....	C-1

DAFTAR SIMBOL

No	Simbol	Nama	Deskripsi
1.		Start State	Merepresentasikan titik awal dimulainya suatu aktivitas dalam diagram aktivitas.
2.		End State	Merepresentasikan titik akhir dari suatu aktivitas dalam diagram aktivitas.
3.		Activity	Menggambarkan aktifitas yang sedang dilakukan.
4.		Control Flow	Menghubungkan aktivitas dan menunjukkan urutan alur proses.
5.		Actor	Menjelaskan peran yang dijalankan oleh pengguna saat berinteraksi use case.
6.		Use Case	Menjelaskan aktifitas yang dilakukan oleh aktor
7.		Association	Menghubungkan antara aktor dan use case, untuk menggambarkan hubungan/interaksi.
8.		Swimlane	Pembagian dalam diagram aktivitas untuk mengelompokkan aktivitas berdasarkan peran

BAB 1

PENDAHULUAN

Bab ini berisi uraian mengenai latar belakang permasalahan, identifikasi masalah, tujuan penelitian, ruang lingkup penelitian, metodologi tugas akhir, serta sistematika alur pengerjaan tugas akhir. Penjelasan dalam bab ini bertujuan untuk memberikan gambaran umum mengenai alasan dan dasar dilakukannya penelitian.

1.1 Latar Belakang

Kantin merupakan fasilitas penting di setiap universitas di Indonesia yang menyediakan tempat bagi mahasiswa dan pengunjung untuk membeli serta menikmati makanan dan minuman. Proses yang umum dilakukan di kantin adalah memesan, membayar, dan kemudian duduk untuk menikmati hidangan. Keberadaan kantin menjadi kebutuhan dasar bagi lingkungan akademik guna mendukung aktivitas mahasiswa sehari-hari. Selain itu, kantin juga menjadi tempat interaksi sosial dan sarana bagi mahasiswa untuk beristirahat di sela-sela perkuliahan[NUR17].

Sejauh ini proses pemesanan makanan di kantin umumnya masih dilakukan secara manual, yang menyebabkan beberapa kendala, terutama pada jam sibuk. Ketidakseimbangan antara jumlah pembeli dan penjual mengakibatkan antrean panjang, waktu tunggu yang lama, serta ketidaknyamanan bagi mahasiswa dalam mendapatkan layanan makanan dan minuman[NUR17]. Selain itu, keterbatasan ruang di kantin juga memperburuk kondisi, mahasiswa dan penjual kerap mengalami kendala dalam menemukan tempat duduk. Mahasiswa kesulitan mendapatkan tempat duduk ketika kantin dalam kondisi penuh, sedangkan penjual mengalami kesulitan mencari tempat duduk saat mengantar pesanan. Sistem pembayaran yang dilakukan di tempat yang sama dengan pemesanan juga semakin memperpanjang waktu transaksi.

Perkembangan teknologi saat ini sangatlah pesat dan memberikan pengaruh yang sangat besar bagi dunia teknologi. Munculnya berbagai macam solusi aplikasi yang bersifat web, desktop, dan mobile membuat pekerjaan seseorang lebih mudah[LUB20]. Oleh karena itu, teknologi ini dapat dimanfaatkan untuk mempermudah proses pemesanan makanan di kantin. Untuk mendukung hal tersebut, akan dibangun sebuah aplikasi berbasis web untuk pemesanan makanan *online* di kantin yang bermanfaat bagi pembeli dan penjual.

Dalam perancangan sistem ini, penulis berfokus pada pembangunan sisi *backend* dari aplikasi berbasis web. merupakan bagian belakang layar dari sebuah website yang memiliki tiga bagian komponen yaitu *server*, *database*, dan aplikasi. Pengembang *backend* bertugas menulis kode yang memungkinkan ketiga elemen tersebut untuk saling berinteraksi dan bekerja sama, sehingga dapat melakukan fungsi dan menyampaikan informasi kepada pengguna akhir[SAL21]. berfungsi untuk menyediakan layanan yang tidak terlihat oleh pengguna, seperti menyimpan, mengolah, serta mengamankan data. juga mendukung *frontend* dalam menjalankan fungsinya dengan optimal. Dalam sistem pemesanan kantin ini, bertanggung

jawab dalam menyediakan REST API untuk komunikasi antara *frontend* dan *database*, serta memastikan transaksi berjalan dengan aman dan efisien [BAC24]. REST API adalah jenis web service yang menerapkan konsep perpindahan antar state melalui link *HTTP* untuk melakukan aktivitas tertentu. Respons dari REST API ini bisa berupa format *XML* atau *JSON*[ARF19].

Teknologi yang digunakan dalam perancangan sistem ini adalah Node.js yang merupakan platform berbasis JavaScript yang berjalan pada *runtime* Chrome dan dirancang untuk membangun perangkat lunak web secara cepat serta mendukung aplikasi jaringan yang scalable. Dengan model *event-driven* dan *non-blocking I/O*, Node.js menjadi platform yang ringan serta efisien dalam menangani banyak permintaan secara bersamaan. Teknologi ini sangat sesuai untuk aplikasi real-time yang dapat digunakan pada berbagai perangkat [RIZ14]. Express.js juga digunakan sebagai kerangka kerja aplikasi web untuk Node.js yang menyediakan alat yang dibutuhkan untuk membangun website ini.

Penelitian mengenai sistem pemesanan makanan di kantin telah dilakukan sebelumnya, salah satu penelitiannya yaitu Nurjanah (2017) yang membuat rancangan model sistem informasi menggunakan konsep CRM untuk meningkatkan kualitas layanan di kantin. Sistem ini dirancang untuk membantu mahasiswa dalam proses pemesanan dan pembayaran. Penelitian ini dilakukan sampai dengan pembuatan *prototype*[NUR17]. Berdasarkan penelitian tersebut, penulis mengusulkan perancangan sistem pemesanan makanan di kantin, khususnya pada sisi . Sistem ini tidak hanya sebatas konsep atau *prototype*, melainkan dikembangkan secara fungsional dengan yang mampu menangani proses transaksi dan menyediakan REST API sebagai penghubung antara *frontend* dan .

Melalui tugas akhir ini, diharapkan sistem yang dibangun dapat menjadi solusi nyata untuk membantu mengatasi permasalahan dalam proses pemesanan manual di kantin. Sistem ini juga diharapkan dapat menjadi contoh pemanfaatan teknologi yang adaptif terhadap kebutuhan pengguna.

1.2 Identifikasi Masalah

Berdasarkan latar belakang diatas, maka dapat diperoleh permasalahan sebagai berikut :

1. Proses pemesanan makanan di kantin masih dilakukan secara manual, yang mengakibatkan inefisiensi waktu dan berpotensi menimbulkan antrean panjang, terutama pada jam sibuk.
2. Pencatatan pesanan manual rentan terhadap kesalahan manusia (*human error*), seperti kesalahan dalam mencatat menu yang dapat menurunkan akurasi data dan kepuasan pelanggan.
3. Tidak adanya jembatan penghubung data yang terstandarisasi (seperti REST API) antara sistem pengelola data (*backend*) dan antarmuka pengguna (*frontend*).

1.3 Tujuan Tugas Akhir

Berdasarkan identifikasi masalah diatas, maka dapat diperoleh tujuan tugas akhir sebagai berikut:

1. Merancang dan membangun sistem *backend* yang mampu mengelola proses pemesanan makanan secara terotomatisasi untuk meningkatkan efektivitas dan mengurangi potensi kesalahan.
2. Menghasilkan sebuah REST API yang berfungsi sebagai penghubung yang andal dan terstruktur antara sisi *frontend* dan untuk memfasilitasi pertukaran data pemesanan.

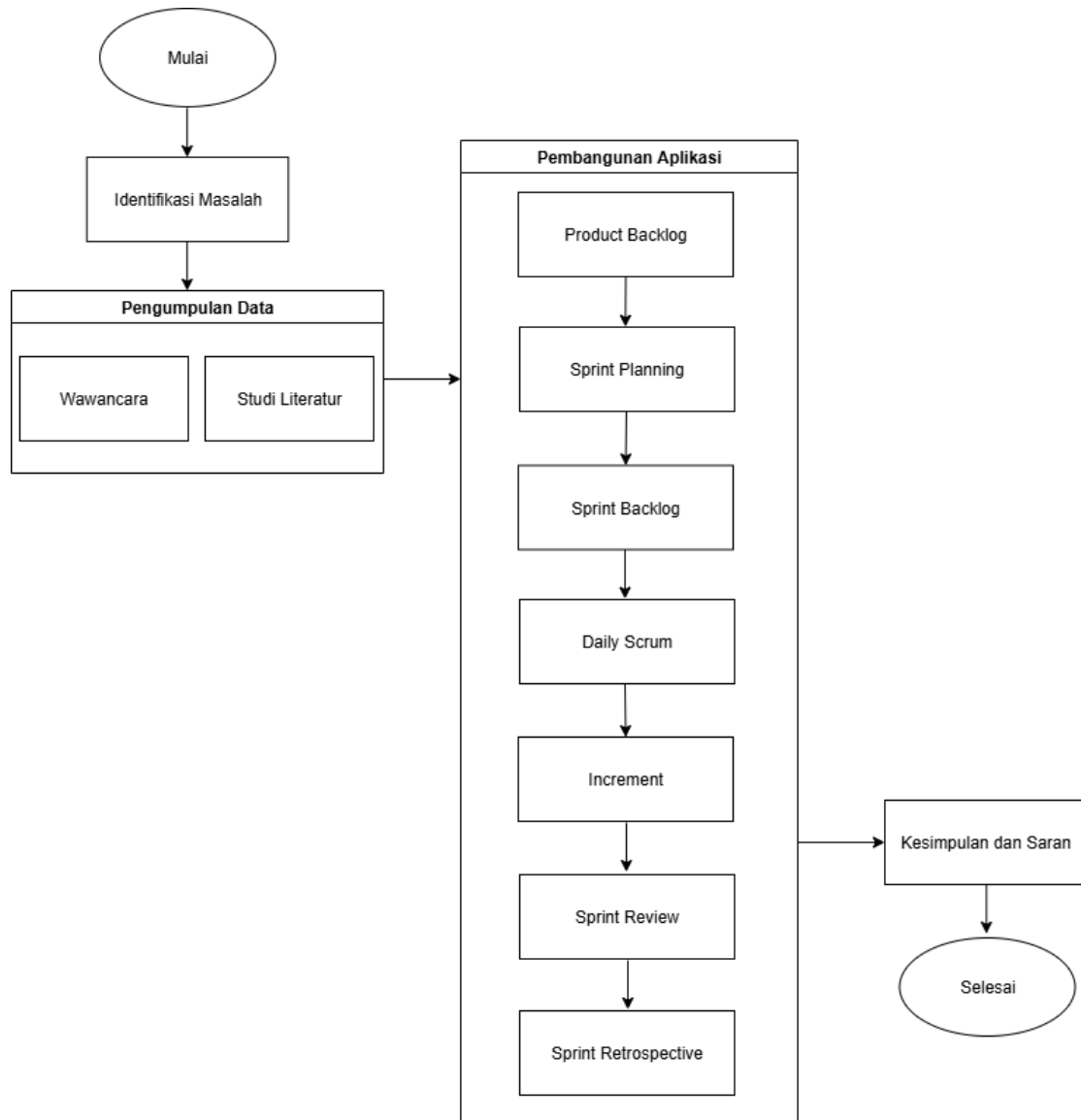
1.4 Lingkup Tugas Akhir

Berikut di bawah ini merupakan lingkup tugas akhir pada penelitian :

1. Penelitian ini berfokus pada perancangan dan implementasi sisi *server (backend)*.
2. Sistem dibangun menggunakan bahasa pemrograman Javascript dengan lingkungan eksekusi Node.js dan kerangka kerja Express.js.
3. Pengelolaan dan penyimpanan data dalam penelitian ini menggunakan sistem manajemen basis data PostgreSQL.

1.5 Metodologi Tugas Akhir

Metodologi penelitian ini menjelaskan kerangka kerja dan langkah-langkah sistematis yang dilakukan untuk menyelesaikan penelitian, mulai dari tahap persiapan hingga penarikan kesimpulan. Secara garis besar, metodologi ini terbagi menjadi tiga tahap utama: tahap pengumpulan data, tahap pengembangan aplikasi dengan kerangka kerja Scrum, dan tahap evaluasi. Alur lengkap dari metodologi penelitian ini divisualisasikan pada Gambar 1.1.



Gambar 1.1 Langkah-langkah penyelesaian tugas akhir

Berikut adalah penjelasan dari setiap tahapan:

1. Identifikasi Masalah:

Melakukan observasi awal untuk menemukan permasalahan utama dalam proses pemesanan makanan di kantin.

2. Tahap Pengumpulan Data

Tahap ini berfokus pada pengumpulan informasi yang dibutuhkan untuk memahami masalah dan menentukan kebutuhan sistem (requirements).

a. Wawancara : Melakukan sesi tanya jawab dengan pihak terkait

b. Studi Literatur : Mempelajari penelitian terdahulu, jurnal, dan dokumentasi terkait sistem pemesanan *online* untuk mendapatkan landasan teori yang kuat.

3. Tahap Pembangunan Aplikasi (Metode Scrum)

- a. Product Backlog : Membuat daftar awal berisi semua fitur (requirements) yang diinginkan untuk sistem, yang telah diperoleh dari identifikasi masalah dan tahap pengumpulan data.
- b. Sprint Planning : Merencanakan pekerjaan yang akan dilakukan dalam satu iterasi (sprint). Tim akan memilih item dari Product Backlog untuk dikerjakan dan memasukkannya ke dalam Sprint Backlog.
- c. Sprint Backlog : Daftar pekerjaan yang telah disetujui tim untuk diselesaikan dalam satu periode sprint.
- d. Daily Scrum : Pertemuan harian singkat untuk sinkronisasi pekerjaan tim dan mengidentifikasi hambatan.
- e. Increment : Hasil kerja yang bisa digunakan atau diuji, yang menambahkan fungsi baru ke produk yang sudah ada.
- f. Sprint Review : Pertemuan diakhir sprint untuk menunjukkan hasil kerja kepada *stakeholder*.
- g. Sprint Retrospective : Pertemuan internal tim untuk melakukan evaluasi terhadap proses kerja yang telah dilalui pada sprint sebelumnya dan mencari cara untuk perbaikan di sprint berikutnya.

4. Tahap Akhir

Kesimpulan dan Saran : Setelah proses pengembangan selesai dan sistem telah diuji, dilakukan penarikan kesimpulan berdasarkan hasil yang dicapai. Saran juga diberikan untuk potensi pengembangan lebih lanjut di masa depan.

1.6 Sitematika Penulisan

Sistematika penulisan tugas akhir dibuat berurutan untuk menggambarkan bab yang terdapat pada tugas akhir, berikut sistematika penulisannya:

BAB 1. PENDAHULUAN

Bab ini membahas gambaran umum penelitian yang berisi latar belakang masalah, identifikasi masalah yang diangkat, tujuan yang ingin dicapai, serta lingkup agar penelitian tetap fokus. Bab ini juga menjelaskan metodologi penelitian yang digunakan secara ringkas dan diakhiri dengan sistematika penulisan laporan.

BAB 2. LANDASAN TEORI DAN PENELITIAN TERDAHULU

Bab ini menguraikan teori-teori, konsep, serta definisi yang mendukung penelitian dalam tugas akhir. Selain itu, bab ini juga mencakup pembahasan mengenai penelitian terdahulu yang relevan dengan topik yang dibahas dalam tugas akhir ini.

BAB 3. SKEMA PENELITIAN

Bab ini menjelaskan rancangan penelitian yang dilakukan, mencakup tahapan analisis, penjelasan mengenai alur pengerjaan tugas akhir, identifikasi masalah, solusi yang diterapkan, serta gambaran mengenai lingkungan atau tempat penelitian.

BAB 4. ANALISIS DAN PERANCANGAN

Bab ini membahas tahapan awal dalam proses pembangunan yang terdiri dari analisis dan perancangan. Proses analisis, pendefinisian perangkat lunak dan analisis kebutuhan dengan menggunakan permodelan spesifikasi kebutuhan.

BAB 5. SCRUM PROSES DAN DELIVERY

Bab ini menjelaskan tahapan pelaksanaan penelitian berdasarkan metodologi Scrum, mulai dari proses perencanaan, pelaksanaan sprint, hingga penyampaian hasil (delivery).

BAB 6. KESIMPULAN DAN SARAN

Bab terakhir ini berisi kesimpulan dari penelitian dan perancangan sistem yang telah dilakukan, serta memberikan saran untuk pengembangan lebih lanjut agar sistem dapat terus disempurnakan di masa mendatang.

BAB 2

LANDASAN TEORI DAN PENELITIAN TERDAHULU

Bab ini membahas teori-teori yang mendasari penelitian serta hasil-hasil penelitian terdahulu yang relevan. Teori berfungsi sebagai dasar untuk menganalisis dan membangun sistem yang dikembangkan. Penelitian terdahulu digunakan sebagai referensi dan pembandingan dalam penulisan tugas akhir

2.1 Teori Pendukung

Dibawah ini merupakan teori-teori yang digunakan sebagai referensi dan pendukung dalam penelitian tugas akhir ini

2.1.1 Javascript

Javascript pertama kali diperkenalkan oleh Netscape pada tahun 1995. Pada awalnya javascript bernama “LiveScript” yang pada masa itu digunakan sebagai bahasa sederhana untuk browser Netscape Navigator 2. Kemudian saat kerjasama antara Netscape dan Sun (pengembang bahasa pemrograman “Java”) pada masa itu, Netscape memberikan nama “JavaScript” pada tanggal 4 Desember 1995 pada bahasa tersebut[WIJ17].

JavaScript adalah bahasa pemrograman yang berjalan di sisi klien dan berbasis prototipe. Dalam konteks web, JavaScript dapat dipahami sebagai bahasa pemrograman yang dirancang khusus untuk dijalankan di browser atau halaman web guna membuat tampilan lebih dinamis dan interaktif. JavaScript berasal dari dua kata: "Java", yang berarti bahasa pemrograman berorientasi objek, dan "Script", yang berarti kumpulan instruksi program[WIJ17].

Bahasa pemrograman ini mengadopsi konsep *Object Oriented Programming(OOP)*, yang dikembangkan dari bahasa pemrograman C++ yang dimana bahasa pemrograman java jauh lebih diandalkan sehingga dapat membuat seluruh bentuk pemrograman dari aplikasi desktop hingga mobile. Oleh karena itu Bahasa ini cocok digunakan sebagai Bahasa pemrograman untuk kalangan programmer yang ingin menggunakannya[KUR20].

2.1.2 Node.js

Ryan Dahl termotivasi untuk membangun node.js setelah melihat progress bar yang menunjukkan proses upload file di Flickr (Perusahaan Yahoo yang mengelola galeri foto), yang Dimana browser itu terus menerus menanyakan ke *server* seberapa besar file yang sedang di unggah. Pada bulan Maret 2009, Ryan menamai ciptaannya node.js, lalu pada Oktober 2009 *package* manager untuk node.js diperkenalkan bersamaan dengan preview awal npm (Node.js *Package* Manager). Pada bulan November di tahun yang sama, Ryan melakukan presentasi pertama tentang Node.js di JSConf, menjelaskan secara detail mengenai Node.js[SHA17].

Node.js adalah platform sisi *server* yang dikembangkan oleh Ryan Dahl pada tahun 2009 dan dibangun di atas mesin JavaScript V8 dari Google Chrome. Ribuan pengembang di seluruh dunia telah menggunakan Node.js, aplikasi open source yang tersedia secara gratis, untuk membuat aplikasi berbasis *server* dan jaringan. Node.js berfungsi sebagai lingkungan runtime JavaScript dan menyediakan berbagai komponen yang dibutuhkan pengembang web untuk membuat program berbasis JavaScript[WID21].

2.1.3 Express JS

Pada tahun 2010, *framework* web untuk node.js yaitu Express.js dikenalkan[SHA17]. Express.js adalah *framework* web yang dikembangkan oleh TJ Holowaychuk yang menggunakan bahasa pemrograman JavaScript dan dibuat khusus untuk Node.js. Ini memungkinkan pengembang membuat aplikasi web dengan lebih cepat dan efektif dengan menggunakan JavaScript sebagai bahasa utama. Beberapa keunggulan dari Express.js meliputi dukungan penuh dari JavaScript, integrasi dengan Google V8 Engine, biaya pengembangan yang lebih rendah, fleksibilitas tinggi, serta performa yang optimal[RIF24].

2.1.4 REST API

Application Programming Interface (API) merupakan antarmuka yang memungkinkan suatu aplikasi atau layanan diakses oleh program lain sehingga memudahkan pengembang dalam memanfaatkan fungsi yang sudah ada tanpa harus membangun ulang dari awal. Dalam konteks web, API biasanya beroperasi melalui protokol HTTP dan menghasilkan respons dalam format data seperti XML atau JSON. Salah satu bentuk API yang paling banyak digunakan adalah REST API (*Representational State Transfer*), yaitu arsitektur layanan web yang dikembangkan oleh Roy Fielding. REST API bekerja dengan prinsip *client-server*, di mana *client* mengirimkan permintaan melalui HTTP dan *server* memberikan respons berupa resource yang diidentifikasi dengan URI. *Resource* tersebut umumnya direpresentasikan dalam format JSON sehingga memudahkan pertukaran data secara sederhana, cepat, dan efisien antar aplikasi[KUR20].

2.1.5 PostgreSQL

PostgreSQL merupakan salah satu sistem manajemen basis data berbasis objek-relasional (*Object-Relational Database Management System/ORDBMS*) yang telah mengalami perkembangan sejak tahun 1977. Awalnya, proyek ini bernama Ingres yang dikembangkan di University of California, Berkeley, dan kemudian dipasarkan secara komersial oleh Relational Technologies/Ingres Corporation.

Pada tahun 1986, Michael Stonebraker bersama timnya di Berkeley melanjutkan pengembangan Ingres dan menghasilkan sistem basis data objek-relasional baru yang diberi nama Postgres. Seiring berjalannya waktu, pada tahun 1996, sistem ini mengalami peningkatan fungsi serta dukungan dari komunitas open source, sehingga namanya berubah menjadi PostgreSQL setelah sebelumnya sempat

dikenal dengan sebutan Postgres95. Hingga kini, PostgreSQL masih aktif dikembangkan oleh komunitas pengembang dan kontributor *open source* di seluruh dunia.

PostgreSQL banyak diakui sebagai salah satu sistem basis data *open source* yang paling canggih. Hal ini dikarenakan PostgreSQL menyediakan berbagai fitur yang umumnya hanya tersedia pada produk basis data komersial berskala besar [WOR02].

2.1.6 Backend

atau *side server* adalah tempat proses aplikasi atau sistem berjalan, di mana data ditambah, diubah, atau dihapus. juga menangani segala sesuatu yang biasanya tidak dilihat atau dihubungi langsung oleh pengguna, seperti *database* dan *server*[ARH16]. juga dapat didefinisikan sebagai komponen utama yang menangani logika fungsional serta operasi dalam sebuah aplikasi perangkat lunak atau sistem informasi. Bagian bertanggung jawab untuk memastikan bahwa data atau layanan yang diminta oleh *frontend* atau aplikasi dapat dikirim dan diterima melalui metode yang telah diprogram. mencakup logika inti aplikasi, basis data, integrasi data dengan aplikasi lain, API, serta berbagai proses lainnya[HER22].

2.1.7 Website

Menurut Surajino, S.H.R (2004), Website atau situs dapat diartikan sebagai kumpulan halaman-halaman yang digunakan untuk menampilkan informasi, teks, gambar diam atau bergerak, animasi, suara, dan atau gabungan dari semuanya itu, baik yang bersifat statis maupun dinamis yang membentuk satu rangkaian bangunan yang saling berkait dimana masing-masing dihubungkan dengan jaringan-jaringan halaman[KUR23]. Web juga dapat diartikan sebagai layanan yang memungkinkan pengguna komputer terhubung ke internet dan menyediakan konten interaktif seperti teks, gambar, suara, dan video. Selain itu, web memiliki fitur yang menghubungkan dokumen (*link*) satu sama lain, yang dapat diakses melalui browser[NOV21].

2.1.8 Scrum

Scrum adalah kerangka kerja ringan yang membantu orang, tim, dan organisasi untuk menghasilkan nilai melalui solusi adaptif untuk masalah yang kompleks. Scrum didasari pada prinsip empirisme dan lean thinking, yang menekankan pembelajaran dari pengalaman dan meminimalkan pemborosan. Scrum menggunakan pendekatan secara iterative dan incremental, dapat membantu mengurangi risiko dan meningkatkan prediktabilitas[SCH20]. Siklus Scrum dimulai dari pertemuan dengan pemangku kepentingan untuk merumuskan visi proyek. Selanjutnya, *Product Owner* menyusun *Product Backlog* yang berisi daftar kebutuhan proyek dalam bentuk User Stories yang diprioritaskan. Setiap Sprint berlangsung selama 1–4 minggu dan diawali dengan Sprint Planning Meeting untuk menentukan User Stories yang akan dikerjakan. Selama Sprint, tim melakukan Daily Standup singkat untuk memantau perkembangan pekerjaan.

Hasil Sprint berupa inkremen produk yang kemudian dipresentasikan dalam Sprint Review Meeting. Setelah itu, tim melaksanakan Sprint Retrospective untuk mengevaluasi proses kerja dan meningkatkan kinerja pada Sprint berikutnya [SCR25].

1. Proses Scrum adalah sebagai berikut [SCH20]:
 - a. Sprint Planning untuk mengatur pekerjaan yang akan dilakukan selama sprint.
 - b. Daily Scrum untuk memeriksa kemajuan menuju sprint goal.
 - c. Sprint Review untuk memeriksa hasil dari sprint dan menentukan adaptasi selanjutnya
 - d. Sprint Retrospective untuk merencanakan cara untuk meningkatkan kualitas dan efektifitas
2. Artefak Scrum adalah sebagai berikut [SCH20]:
 - a. Product Backlog (*Product Goal*) adalah daftar yang muncul dan tersusun tentang apa yang dibutuhkan untuk meningkatkan produk.
 - b. Sprint Backlog (*Sprint Goal*) adalah rencana kerja yang dibuat oleh dan ditujukan untuk para developer. Dokumen ini adalah Gambaran nyata mengenai pekerjaan yang akan dilakukan selama sprint untuk mencapai sprint goal.
 - c. Increment (*Definition of Done*) adalah hasil kerja yang mendekatkan pada product goal, terintegrasi dengan increment sebelumnya, telah diverifikasi, dan harus dapat digunakan agar bernilai.
3. Keunggulan penggunaan scrum[SCR25] :
 - a. Adaptabilitas (*Adaptability*): Kontrol proses empiris dan pengiriman iteratif membuat proyek menjadi adaptif dan terbuka untuk mengakomodasi perubahan.
 - b. Transparansi (*Transparency*): Semua penyebar informasi seperti Papan Scrum (Scrumboard) dibagikan kepada semua pihak, sehingga menciptakan lingkungan kerja yang terbuka.
 - c. Umpan Balik Berkelanjutan (*Continuous Feedback*): Umpan balik berkelanjutan diberikan melalui proses seperti Daily Standup dan Sprint Review.
 - d. Peningkatan Berkelanjutan (*Continuous Improvement*): Hasil kerja ditingkatkan secara progresif dari satu Sprint ke Sprint berikutnya.
 - e. Pengiriman Nilai secara Berkelanjutan (*Continuous Delivery of Value*): Proses iteratif memungkinkan pengiriman nilai secara terus-menerus kepada pelanggan.
 - f. Ritme Kerja yang Berkelanjutan (*Sustainable Pace*): Scrum dirancang agar tim dapat bekerja pada ritme yang sehat dan dapat dipertahankan dalam jangka panjang.
 - g. Pengiriman Awal Nilai Tertinggi (*Early Delivery of High Value*): Proses prioritas pada Product Backlog memastikan kebutuhan pelanggan yang paling berharga dikerjakan terlebih dahulu.
 - h. Proses Pengembangan yang Efisien (*Efficient Development Process*): Konsep *time-boxing* dan minimalisasi pekerjaan yang tidak esensial meningkatkan efisiensi.
 - i. Motivasi (*Motivation*): Keterlibatan aktif dan rasa kepemilikan dalam proses seperti Daily Standup dan Sprint Retrospective meningkatkan motivasi tim.

- j. fungsional (*cross-functional*) mempercepat proses penyelesaian masalah.
- k. Hasil Kerja yang Efektif (*Effective Deliverables*): Prioritas dan tinjauan rutin memastikan hasil kerja yang dikirimkan benar-benar efektif dan sesuai kebutuhan.
- l. Berpusat pada Pelanggan (*Customer Centric*): Penekanan pada nilai bisnis dan keterlibatan pemangku kepentingan memastikan kerangka kerja yang berorientasi pada pelanggan.
- m. Lingkungan dengan Kepercayaan Tinggi (*High Trust Environment*): Transparansi dan kolaborasi menciptakan lingkungan kerja dengan kepercayaan tinggi dan minim friksi.
- n. Kepemilikan Kolektif (*Collective Ownership*): Tim memiliki rasa kepemilikan bersama terhadap pekerjaan, yang akan menghasilkan kualitas lebih baik.
- o. Kecepatan (*Velocity*) yang Tinggi: Kerangka kerja kolaboratif memungkinkan tim mencapai potensi penuh dan kecepatan kerja yang tinggi.
- p. Lingkungan yang Inovatif (*Innovative Environment*): Proses retrospeksi menciptakan budaya introspeksi, pembelajaran, dan adaptasi yang mendorong inovasi.

2.1.9 Kantin

Kantin (dari bahasa Belanda: *kantine*) adalah sebuah ruangan dalam gedung umum yang dapat digunakan pengunjung untuk makan, baik makanan yang dibawa sendiri maupun yang dibeli di kantin. Kantin harus mengikuti prosedur tertentu terkait cara pengolahan makanan dan menjaga kebersihan. Makanan yang disediakan di kantin harus bersih dan halal. Selain itu, jenis-jenis makanan yang disediakan minimal harus memenuhi prinsip empat sehat lima sempurna, sehingga dapat memenuhi kebutuhan gizi pengunjung secara seimbang[OCT19]. Hampir setiap universitas di Indonesia memiliki kantin sebagai fasilitas utama. Kantin sekarang menjadi fasilitas penting di lingkungan universitas. Banyak mahasiswa memanfaatkannya sebagai tempat untuk membeli makanan dan minuman serta bersosialisasi dan berinteraksi dengan teman dari berbagai Angkatan[NUR17].

2.1.10 Pemesanan

Pemesanan adalah suatu kegiatan yang dilakukan pembeli sebelum membeli sesuatu [SYA14]. Sistem pemesanan yang baik diperlukan perusahaan untuk mewujudkan kepuasan konsumen. Tujuan utama dari pemesanan antara lain: (1) memaksimalkan pelayanan kepada konsumen, (2) meminimalkan investasi pada persediaan, (3) mendukung perencanaan kapasitas, serta (4) mengatur persediaan dan kapasitas perusahaan (Utara, 2011)[RAH15].

2.2 Penelitian Terdahulu

Penelitian terdahulu berfungsi sebagai landasan dalam pengembangan penelitian ini. Dengan meninjau berbagai penelitian yang relevan, peneliti dapat mengetahui kontribusi, kelebihan, maupun keterbatasan dari penelitian sebelumnya sehingga dapat menjadi acuan dalam merancang sistem yang lebih baik.

Tabel 2.1 Penelitian Terdahulu

No	Judul Penelitian	Peneliti	Hasil Penelitian
1.	Perancangan Sistem Pemesanan Makanan Berbasis Website Pada Kantin (Institut Shanti Bhuana)	[OSK23]	Kantin Institut Shanti Bhuana (ISB) memiliki sistem pemesanan makanan berbasis web yang berhasil dirancang dengan antarmuka responsif yang memudahkan pengguna memesan makanan, melakukan pembayaran, dan menerima notifikasi pesanan. Sistem ini memberikan kenyamanan dengan memungkinkan pemesanan tanpa antrai, akses ke informasi harga dan menu, dan proses transaksi yang aman dan efisien. Meski begitu, sistem ini tetap memerlukan koneksi internet yang stabil untuk mengaksesnya dan mungkin kurang diminati oleh pengguna yang lebih memilih datang langsung ke kantin.
2.	KantinKita : Sistem Pemesanan Makanan Berbasis Website	[SET25]	Hasil penelitian menunjukkan bahwa KantinKita berhasil menyediakan layanan pemesanan berbasis web yang efisien, mengatasi masalah umum seperti antrean Panjang dan keterlambatan penyajian, membuat fitur pemilihan menu, pembayaran digital, pelacakan status pesanan secara real time, membantu staf kantin dalam mengelola stok dan menghasilkan laporan penjualan harian, serta membuat super admin untuk mengontrol penuh aktivitas penjualan, mengelola data pelanggan, dan merespon kritik pengguna.
3.	Canteen Food Ordering and Managing System	[ANT22]	Hasil penelitian menunjukkan bahwa aplikasi ini telah berhasil diuji dan menunjukkan kinerja yang efektif dalam mengurangi antrean serta meningkatkan efisiensi pemesanan makanan. Hasil yang diperoleh dari implementasi sistem ini antara lain: fitur register & login, pemesanan menu secara <i>online</i> , notifikasi otomatis ke koki, pemberitahuan pesanan siap ke pelanggan, memberi kemudahan untuk penjual, dan mengurangi antrean panjang.
4.	Perancangan dan Pengimplementasian Sistem Backend Aplikasi Cafeasy (Studi Kasus: Kafe daerah Bandung)	[PRI24]	Hasil penelitian menunjukkan bahwa implementasinya berjalan dengan baik pada sisi <i>backend</i> yang dibuat. Hasil dari endpoint API admin berjumlah 24, dan endpoint pelanggan berjumlah 14. Semua endpoint API yang dibuat dapat berjalan dengan baik. endpoint API Cafeasy yang dibuat dalam perancangan dan pengimplementasian MERN dapat bekerja sesuai dengan product backlog, serta tes performa pemuatan data memiliki hasil yang baik, tidak menghabiskan waktu sampai 1 detik.
5.	Perancangan Sistem Informasi Kantin Menggunakan Customer Relationship Management (Studi Kasus: Kantin Fakultas Teknik Universitas Pasundan)	[NUR17]	Hasil penelitian menunjukkan bahwa penelitian ini berhasil membuat rancangan model sistem informasi pemesanan dan pembayaran di kantin dengan menggunakan konsep Customer Relationship Management (CRM). Rancangan tersebut bertujuan untuk meningkatkan kualitas layanan kantin dengan mempermudah proses pemesanan dan pembayaran. Namun, penelitian ini hanya sampai pada tahap perancangan, belum diimplementasikan secara fungsional.

BAB 3

SKEMA PENELITIAN

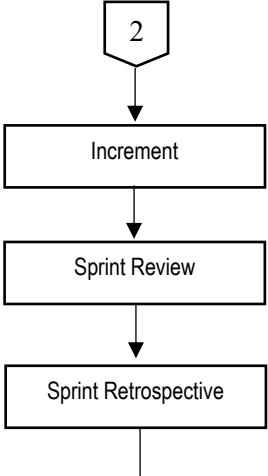
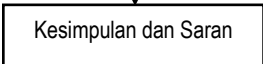
Bab ini berisi penjelasan mengenai alur penyelesaian tugas akhir, diikuti dengan analisis permasalahan dan solusi yang diusulkan. Selain itu, juga disajikan kerangka berpikir teoritis terkait produk yang akan dibuat serta profil objek penelitian.

3.1 Alur Penyelesaian Tugas Akhir

Alur penelitian merupakan kerangka kerja yang menjelaskan tahapan serta Langkah Langkah detail dalam pengerjaan tugas akhir. Adapun alur penelitian tugas akhir ini adalah sebagai berikut:

Tabel 3.1 Alur Pengerjaan Tugas Akhir

Tahap dan Hasil	Langkah Penelitian	Literatur & Referensi
Tahap 1: Menentukan topik tugas akhir, mengidentifikasi masalah tugas akhir Hasil: Mendapatkan topik Tugas Akhir dan masalah yang telah teridentifikasi Kontribusi: Berguna untuk memberikan batasan dalam analisis penelitian serta tahap pengumpulan data	<pre> graph TD A[Menentukan Topik Tugas Akhir] --> B[Mengidentifikasi Masalah] </pre>	
Tahap 2: Studi Literatur dan Pengumpulan Data Hasil: Teori yang relevan dipahami dan mendapatkan data yang dikumpulkan Kontribusi : Berguna untuk tahap selanjutnya yaitu pada tahapan perencanaan & perancangan	<pre> graph TD C[Melaksanakan wawancara] --> D[Mengumpulkan dan memahami studi literatur] </pre>	Kantin & [NUR17], [LUB20], pemesanan [RAH15]
Tahap 3: Perencanaan dan pelaksanaan sprint menggunakan metode Scrum Hasil : Menyusun backlog prioritas dan pelaksanaan perancangan harian Kontribusi : memberikan arah dan kontrol dalam pembangunan fitur	<pre> graph TD E[Product Backlog] --> F[Sprint Planning] F --> G[Sprint Backlog] G --> H[Daily Scrum] H --> I{{2}} </pre>	Pengertian, Teknik, artefak & proses scrum [SCH20], [SCR25]

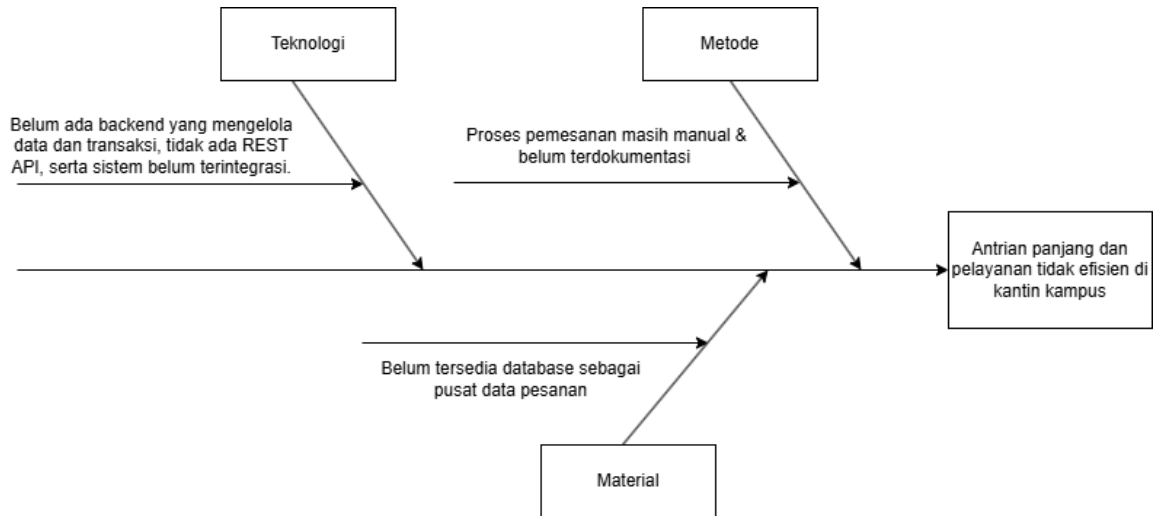
Tahap dan Hasil	Langkah Penelitian	Literatur & Referensi
Tahap 4: Mengevaluasi hasil sprint, menghasilkan <i>increment</i> yang sudah selesai, dan melakukan refleksi perbaikan. Hasil: Mengevaluasi hasil sprint dan melakukan refleksi perbaikan Kontribusi : Meningkatkan kualitas dan efisiensi sprint berikutnya	 <pre> graph TD A{{2}} --> B[Increment] B --> C[Sprint Review] C --> D[Sprint Retrospective] </pre>	Proses Scrum [SCH20]&[SCR25]
Tahap 5: Kesimpulan dan saran Hasil: Kesimpulan dan saran tugas akhir	 <pre> graph TD E[Kesimpulan dan Saran] </pre>	

3.2 Perumusan Masalah

Perumusan masalah bertujuan untuk mengidentifikasi suatu permasalahan yang terdapat pada penelitian tugas akhir. Proses ini dirumuskan dalam menggunakan diagram yang dikenal juga sebagai diagram fishbone. Adapun perumusan masalah dibagi dua hal, yaitu:

3.2.1 Analisis Sebab Akibat

Analisis sebab akibat dilakukan dengan menggunakan diagram fishbone sebagai representasi persoalan saat membangun desain aplikasi Kantin Unpas. Berikut ini merupakan gambar fishbone diagram analisis sebab akibat yang dapat dilihat pada gambar dibawah:



Gambar 3.1 Analisis Sebab Akibat

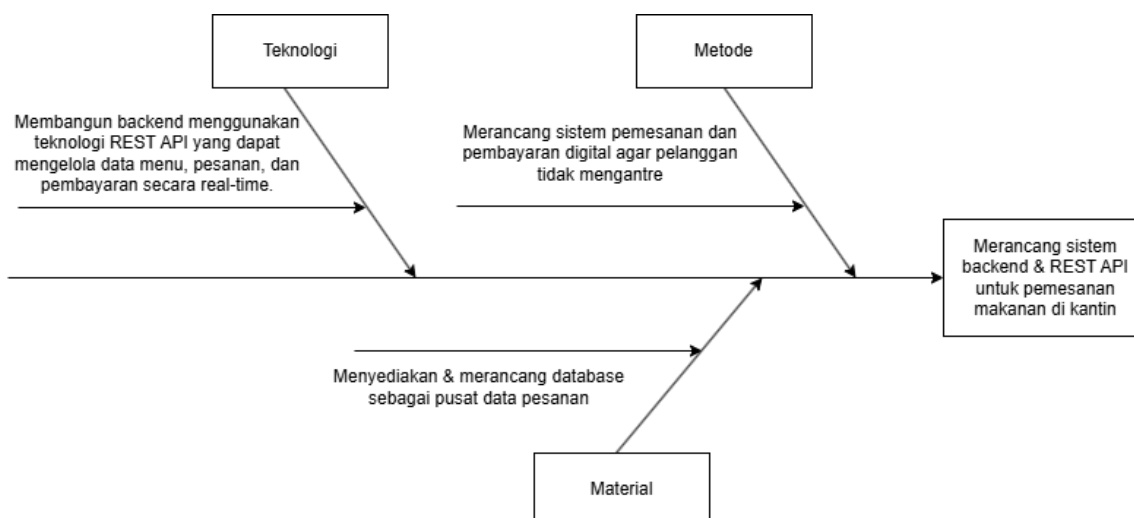
Dibawah ini adalah penjelasan mengenai detail hasil dari analisis sebab akibat yang digambarkan pada diagram diatas:

Tabel 3.2 Detail Fishbone Sebab Akibat

No.	Komponen Utama	Penjelasan
1.	Metode	Pemesanan masih manual, menyebabkan antrean panjang dan pelayanan lambat.
2.	Teknologi	Belum ada <i>backend</i> dan REST API, jadi data dan transaksi sulit dikelola secara efisien.
3.	Material	Tidak tersedia <i>database</i> sebagai pusat penyimpanan data pesanan dan menu, sehingga data tidak terorganisir dan sulit diakses secara sistematis. Hal ini menyebabkan ketidakefisienan dalam pengelolaan pesanan.

3.2.2 Analisis Solusi

Analisis solusi masalah dilakukan dengan menggunakan diagram fishbone sebagai solusi yang muncul dari masalah atau persoalan sebab akibat yang sedang berlangsung. Berikut merupakan analisis solusi masalah yang terdapat pada penelitian tugas akhir.



Gambar 3.2 Analisis Solusi

Dibawah ini adalah penjelasan mengenai detail hasil dari analisis solusi masalah yang digambarkan pada diagram diatas:

Tabel 3.3 Detail Fishbone Analisis Solusi

No.	Komponen Utama	Penjelasan
1.	Metode	Merancang sistem pemesanan dan pembayaran digital yang memungkinkan pelanggan untuk memesan dan membayar secara <i>online</i> tanpa harus antri secara langsung. Sistem ini akan mempercepat proses transaksi dan mengurangi kerumunan di kantin.
2.	Teknologi	Membangun <i>backend</i> dengan teknologi REST API yang mampu mengelola data menu, pesanan, dan pembayaran secara real-time. Sistem ini akan mengintegrasikan seluruh proses pemesanan sehingga lebih efisien dan terstruktur.
3.	Material	menyediakan <i>database</i> terstruktur untuk menyimpan data pesanan dan menu.

3.3 Profil Penelitian



Gambar 3.3 Logo Unpas

Nama Tempat: Kantin Universitas Pasundan Setiabudi

Alamat : Jl. Dr. Setiabudi No.193, Gegerkalong, Kec.
Sukasari, Kota Bandung, Jawa Barat 40153

3.3.1 Deskripsi Singkat

Sebelum mengalami perubahan yang signifikan, Kantin UNPAS dulu masih dalam kondisi yang kurang terawat dan dikelola oleh Koperasi Warga Unpas. Namun, sekarang pihak universitas sudah menaruh perhatian lebih terhadap kondisi tersebut. Wakil dekan II mengambil Langkah untuk mengalihkan pengelolaan kantin ke universitas secara terpusat. Bangunan kantin direnovasi untuk menjadi lebih nyaman dan representatif sebagai bagian dari perbaikan.

Saat ini, kantin UNPAS Setiabudi memiliki sekitar 50 kios. Universitas memberikan prioritas penyewaan pada pedagang lama yang telah berjualan sebelum renovasi dan juga pada pensiunan staf dan dosen UNPAS sebagai bentuk tanggung jawab social dari universitas. Tetapi, meskipun universitas memberikan prioritas pada pedagang lama serta pensiunan staf dan dosen, masyarakat umum juga boleh mendaftar sebagai penyewa selama masih ada kios yang kosong. Dengan sistem pengelolaan yang lebih terstruktur, kantin UNPAS kini sudah menjadi fasilitas kampus yang lebih nyaman, tertata dan juga mendukung kebutuhan warga UNPAS.

3.3.2 Struktur Organisasi



Gambar 3.4 Struktur Organisasi Kantin

BAB 4

ANALISIS DAN PERANCANGAN

Bab ini menjelaskan analisis kebutuhan dari aplikasi dan perancangan website pemesanan makanan *online* di kantin. Proses analisis sendiri terdiri dari tahap mendefinisikan kebutuhan yang harus terpenuhi oleh perangkat lunak. Sedangkan pada tahap perancangan arsitektur dari perangkat lunak yang akan dibangun pada tugas akhir ini.

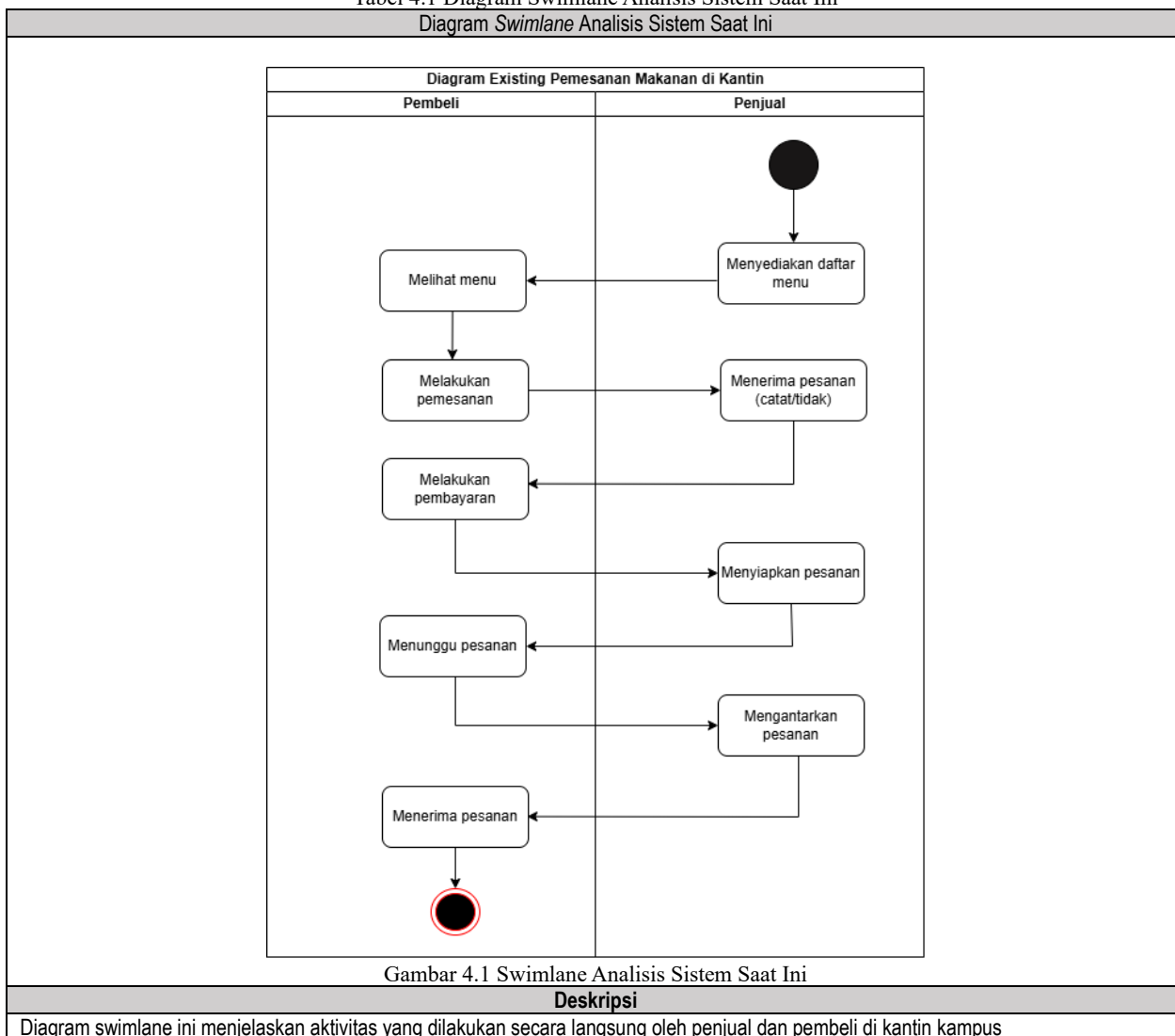
4.1 Analisis Kebutuhan Perangkat Lunak

Analisis kebutuhan perangkat lunak bertujuan untuk memahami kebutuhan dari sistem yang akan dibangun dan memahami spesifikasi, karakteristik, dan fitur yang akan dikembangkan pada perangkat lunak

4.1.1 Analisis Sistem Saat Ini

Analisis sistem saat ini adalah proses untuk mengidentifikasi dan mengevaluasi sistem yang sudah ada sebelum pengembangan perangkat lunak baru. Proses ini bertujuan untuk memahami kekuatan, kelemahan, peluang, dan ancaman dari sistem yang sedang berjalan, serta mengidentifikasi area yang memerlukan perbaikan atau peningkatan. Informasi yang diperoleh dari analisis ini akan mempengaruhi desain dan fitur dari sistem yang akan dikembangkan.

Tabel 4.1 Diagram Swimlane Analisis Sistem Saat Ini
Diagram *Swimlane* Analisis Sistem Saat Ini



4.1.2 Analisis Kebutuhan Pengguna

Analisis kebutuhan pengguna merupakan proses untuk mengidentifikasi pihak pihak yang akan menggunakan perangkat lunak yang akan mempengaruhi pembangunan fitur dari aplikasi yang akan dibangun.

Tabel 4.2 User Requirement

Kode	User Requirement
UR-01	Penjual ingin dapat menyediakan data menu untuk pembeli
UR-02	Pembeli ingin dapat melihat daftar menu dengan jelas
UR-03	Pembeli ingin dapat melakukan pemesanan dengan mudah
UR-04	Penjual ingin dapat menerima notifikasi saat ada pesanan masuk
UR-05	Pembeli ingin dapat melakukan pembayaran dengan mudah
UR-06	Penjual ingin dapat menyiapkan pesanan sesuai pesanan
UR-07	Pembeli ingin dapat menunggu pesanan dengan informasi yang jelas
UR-08	Penjual ingin dapat menyerahkan pesanan kepada pembeli
UR-09	Pembeli ingin dapat menerima pesanan sesuai pesanannya

4.1.3 Analisis Kebutuhan Fungsional

Analisis fungsional adalah tahap yang menghasilkan kebutuhan untuk perangkat lunak yang akan dikembangkan.

Tabel 4.3 Software Requirement

Kode	User Requirement	kode	Software Requirement
UR-01	Penjual ingin dapat menyediakan data menu untuk pembeli	KF-01	Perangkat lunak harus memungkinkan penjual mengelola menu (menambah, melihat, memperbaharui, dan menghapus)
UR-02	Pembeli ingin dapat melihat daftar menu dengan jelas	KF-02	Perangkat lunak harus dapat menampilkan daftar kios
		KF-03	Perangkat lunak dapat menampilkan daftar menu (per kios)
		KF-04	Perangkat lunak harus menampilkan detail menu
		KF-05	Perangkat lunak harus dapat melakukan pencarian menu berdasarkan keyword
UR-03	Pembeli ingin dapat melakukan pemesanan dengan mudah	KF-06	Perangkat lunak harus dapat mengelola keranjang
		KF-07	Perangkat lunak harus dapat membuat pesanan dari keranjang dengan data pembeli yang lengkap
UR-04	Penjual ingin dapat menerima notifikasi saat ada pesanan masuk	KF-08	Perangkat lunak harus dapat mengirim notifikasi saat ada pesanan masuk
UR-05	Pembeli ingin dapat melakukan pembayaran dengan mudah	KF-09	Perangkat lunak harus dapat menyediakan pembayaran menggunakan payment gateway
UR-06	Penjual ingin dapat menyiapkan pesanan sesuai pesanan	KF-10	Perangkat lunak harus dapat menampilkan daftar pesanan masuk beserta detail pesanan
		KF-11	Perangkat lunak harus dapat mengubah status pesanan
UR-07	Pembeli ingin dapat menunggu pesanan dengan informasi yang jelas	KF-12	Perangkat lunak harus dapat menampilkan status pesanan dan estimasi waktu selesai pesanan
UR-08	Penjual ingin dapat menyerahkan pesanan kepada pembeli	KF-13	Perangkat lunak harus dapat menangani perbedaan tipe penyerahan (ambil sendiri atau diantar) dengan tepat
UR-09	Pembeli ingin dapat menerima pesanan sesuai pesannya	KF-14	Perangkat lunak harus memberi notifikasi kepada pembeli ketika pesanan sudah siap atau sudah diserahkan.

4.1.4 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional merupakan proses identifikasi kebutuhan sistem yang tidak mempengaruhi bagaimana sistem bekerja secara langsung.

Tabel 4.4 Analisis Kebutuhan Non Fungsional

Kode	Kebutuhan Non Fungsional
KNF-01	Perangkat lunak harus dapat mengelola autentikasi pengguna
KNF-02	Perangkat lunak harus menyediakan REST API untuk setiap resource yang di kelola

4.1.5 Pemodelan Berbasis Skenario

Pemodelan skenario digunakan untuk merepresentasikan interaksi antara *user* (aktor) dengan sistem yang akan dikembangkan. Umumnya, pemodelan ini dimulai dengan pembuatan Use Case dan diagram aktivitas (sering disebut *Swimlane*). Berikut merupakan beberapa Use Case dari perangkat lunak yang akan dibangun.

4.1.5.1 Deskripsi Aktor

Aktor adalah pengguna atau entitas yang berinteraksi dengan sistem. Bagian ini menjelaskan aktor dalam *Use Case Diagram*. Berikut adalah tabel definisi aktor.

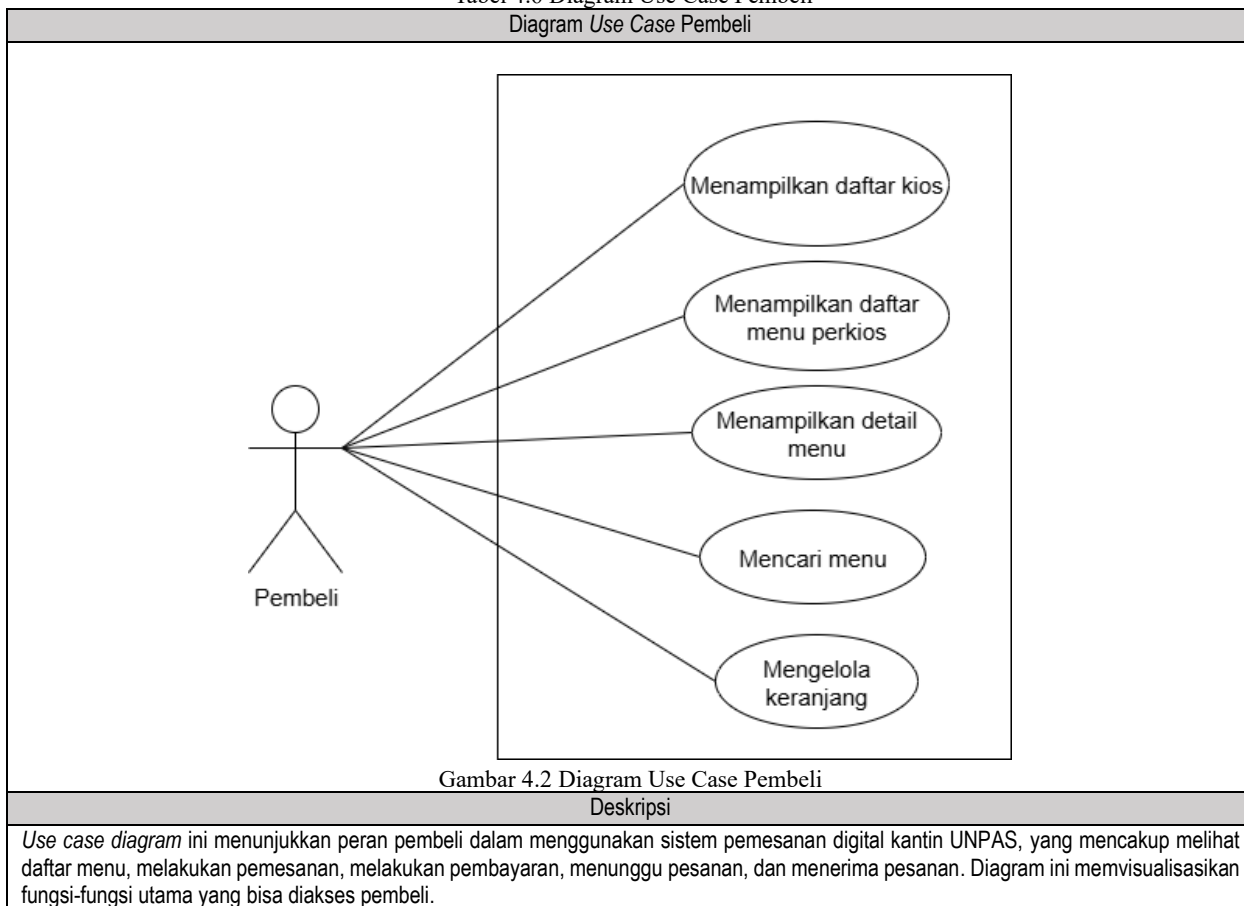
Tabel 4.5 Deskripsi Aktor

No	Aktor	Deskripsi Aktor
1	Pembeli	Orang yang ingin memesan makanan dan minuman dari kios kantin melalui website
2	Penjual	Orang yang memiliki kios dan bertanggung jawab mengelola menu serta menerima dan memproses pesanan dari pembeli melalui website

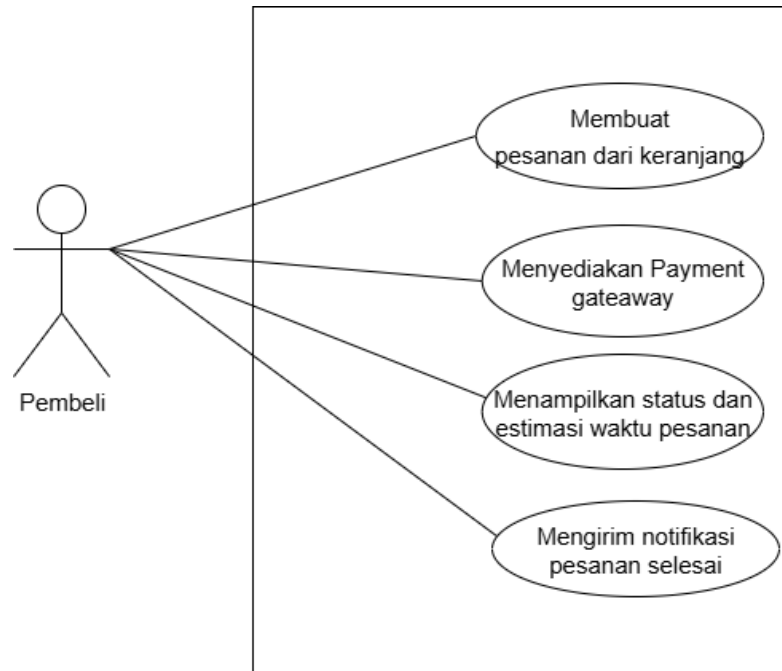
4.1.5.2 Diagram Use Case

Use Case merupakan rangkaian interaksi antara aktor dengan sistem yang dirancang untuk mencapai tujuan tertentu. Pada bagian ini dijelaskan gambaran Use Case Diagram yang terdapat dalam perangkat lunak. Diagram ini menunjukkan hubungan antara aktor dan sistem, serta aktivitas yang dapat dilakukan oleh masing-masing aktor.

Tabel 4.6 Diagram Use Case Pembeli



Tabel 4.7 Diagram Use Case Pembeli
Diagram Use Case Pembeli

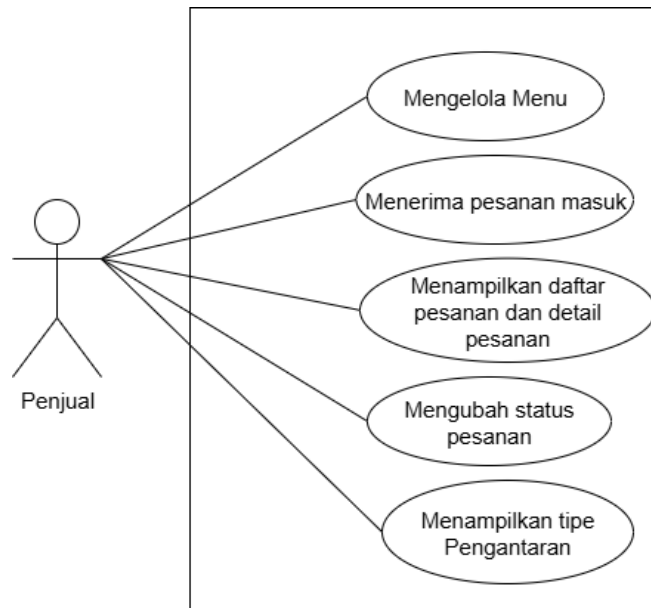


Gambar 4.3 Diagram Use Case Pembeli

Deskripsi

Use case diagram ini menunjukkan peran pembeli dalam menggunakan sistem pemesanan digital kantin UNPAS, yang mencakup melihat daftar menu, melakukan pemesanan, melakukan pembayaran, menunggu pesanan, dan menerima pesanan. Diagram ini memvisualisasikan fungsi-fungsi utama yang bisa diakses pembeli.

Tabel 4.8 Diagram Use Case Penjual
Diagram Use Case Penjual



Gambar 4.4 Diagram Use Case Penjual

Deskripsi

Use case diagram ini menunjukkan peran penjual dalam sistem pemesanan digital kantin UNPAS, yang meliputi menyediakan data menu, menerima pesanan masuk, menyiapkan pesanan, dan mengantarkan pesanan kepada pembeli. Diagram ini memvisualisasikan fungsi-fungsi utama yang dapat diakses oleh penjual untuk mendukung kelancaran proses pemesanan.

4.1.5.3 Deskripsi Diagram Use Case

Berikut merupakan deskripsi detail dari use case yang ditampilkan pada gambar di atas, yang akan dijelaskan lebih lanjut dalam tabel di bawah ini :

Tabel 4.9 Deskripsi Diagram Use Case Pembeli

Kode	Nama Use Case	Deskripsi
UC-01	Menampilkan daftar kios	Use case ini digunakan untuk menampilkan daftar kios yang tersedia agar pembeli dapat memilih kios tujuan.
UC-02	Menampilkan daftar menu per kios	Use case ini digunakan untuk menampilkan daftar menu dari kios yang dipilih
UC-03	Menampilkan detail menu	Use case ini digunakan untuk menampilkan informasi detail dari menu yang dipilih, seperti foto, nama menu, harga, dan deskripsi, sehingga pembeli dapat menentukan pilihan dengan lebih jelas.
UC-04	Mencari menu	Use case ini digunakan untuk memudahkan pembeli menemukan menu tertentu dengan menggunakan kata kunci pencarian.
UC-05	Mengelola keranjang	Use case ini digunakan untuk menambahkan, memperbarui, atau menghapus menu yang ada di dalam keranjang sebelum pesanan dikonfirmasi.
UC-06	Membuat pesanan dari keranjang	Use case ini digunakan untuk membuat pesanan berdasarkan isi keranjang dengan melengkapi data pembeli.
UC-07	Menyediakan payment gateway	Use case ini digunakan untuk memfasilitasi proses pembayaran
UC-08	Menampilkan status dan estimasi waktu pesanan	Use case ini digunakan untuk menampilkan status pesanan yang sedang diproses serta estimasi waktu selesai agar pembeli mendapatkan informasi yang jelas
UC-09	Mengirim notifikasi pesanan selesai	Use case ini digunakan untuk memberikan notifikasi kepada pembeli ketika pesanan telah selesai.

Tabel 4.10 Deskripsi Diagram Use Case Penjual

Kode	Nama Use Case	Deskripsi
UC-10	Mengelola menu	Use case ini digunakan untuk menambahkan, mengedit, dan menghapus menu agar informasi yang ditampilkan ke pembeli selalu sesuai.
UC-11	Mengirim notifikasi pesanan masuk	Use case ini digunakan untuk memberikan pemberitahuan kepada penjual saat ada pesanan baru dari pembeli.
UC-12	Menampilkan daftar pesanan dan detail pesanan	Use case ini digunakan untuk menampilkan daftar pesanan yang masuk beserta detailnya, seperti menu, jumlah, catatan pembeli, dan total harga.
UC-13	Mengubah status pesanan	Use case ini digunakan untuk memperbarui status pesanan (menunggu diproses, sedang diproses, dan selesai) agar pembeli mengetahui update status pesannya.
UC-14	Menampilkan tipe pengantaran	Use case ini digunakan untuk menampilkan pilihan tipe pengantaran pesanan (ambil sendiri atau antar) sesuai dengan yang dipilih pembeli.

4.1.5.4 Skenario Use Case

Skenario use case adalah representasi alur kerja sistem yang menggambarkan bagaimana perangkat lunak berjalan sesuai dengan use case yang sudah ditentukan. Di dalamnya, terdapat aktor yang berinteraksi dengan use case tersebut serta rangkaian aksi dan respon yang berlangsung antara aktor dengan sistem. Berikut disajikan skenario use case untuk layanan web service yang akan dikembangkan.

Tabel 4.11 Menampilkan Daftar Kios

Identifikasi	
Kode	UC-01
Tujuan	Pembeli melihat daftar kios aktif.
Deskripsi	Menampilkan list kios
Aktor	Pembeli (guest)

Kondisi Awal	Pembeli mengakses PasMeal via browser	
Skenario Utama		
Aksi Aktor		Reaksi Sistem
1. Pembeli mengakses PasMeal via browser		
2. Request data kios		3. Menerima request
		4. Sistem mengambil data kios terbaru dari basis data.
		5. Mengembalikan response berisi daftar kios
6. Pembeli melihat daftar kios (maksimal sejumlah default, misalnya 8).		
Kondisi Akhir	Daftar kios berhasil ditampilkan di beranda.	
Skenario Alternatif		
Aksi Aktor		Reaksi Sistem
1. Pembeli membuka halaman beranda untuk melihat daftar kios		2. Sistem mengalami kegagalan saat mengambil data (misalnya gangguan basis data).
		3. Sistem menampilkan pesan error atau notifikasi bahwa data tidak bisa dimuat.
4. Menerima response gagal dan pesan error tersebut		
Kondisi Akhir	Daftar kios tidak tampil; pembeli diberi informasi untuk mencoba kembali.	

Tabel 4.12 Menampilkan Daftar Menu Per Kios

Identifikasi	
Kode	UC-02
Tujuan	Pembeli melihat daftar menu dari kios tertentu.
Deskripsi	Mengembalikan list menu berdasarkan kios_id yang diminta.
Aktor	Pembeli (guest).
Kondisi Awal	Pembeli membuka halaman kios tertentu
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Pembeli membuka daftar menu pada sebuah kios.	
2. Request data menu	3. Menerima request
4.	5. Sistem membaca identitas kios dari permintaan pengguna.
	6. Sistem mengambil data menu yang tersedia untuk kios tersebut.
	7. Mengembalikan response berisi daftar menu
8. Daftar menu ditampilkan di UI.	
Kondisi Akhir	Daftar menu kios berhasil ditampilkan kepada pembeli.
Skenario Alternatif	
Aksi Aktor	Reaksi Sistem
1. Pembeli membuka daftar menu pada kios yang belum memiliki menu.	2. Sistem tidak menemukan data menu pada kios tersebut.
	3. Sistem mengembalikan data kosong.
4. UI menampilkan pesan "Belum ada menu".	
Kondisi Akhir	Tidak ada menu ditampilkan, pembeli mendapatkan informasi bahwa menu belum tersedia.
Skenario Alternatif	
Aksi Aktor	Reaksi Sistem
1. Pembeli membuka daftar menu kios.	2. Terjadi kegagalan pada sistem (misalnya error basis data).
	3. Sistem mengirimkan pesan kegagalan.
4. UI menampilkan notifikasi error/coba lagi.	
Kondisi Akhir	Daftar menu tidak tampil karena kegagalan server.

Tabel 4.13 Menampilkan Detail Menu

Identifikasi	
Kode	UC-03
Tujuan	Pembeli melihat detail satu menu.
Deskripsi	Mengembalikan informasi lengkap menu (foto, nama, deskripsi, harga, estimasi, status).
Aktor	Pembeli (guest).
Kondisi Awal	Pembeli membuka halaman detail menu
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Pembeli membuka halaman detail menu	
2. Request detail menu	3. Menerima request
	4. Sistem memvalidasi ID menu yang diberikan.
	5. Sistem mencari detail menu sesuai ID tersebut di basis data.
	6. Sistem memeriksa apakah data menu ditemukan.
	7. Mengembalikan response berisi detail menu
8. UI menerima response dan menampilkan detail menu.	
Kondisi Akhir	Detail menu berhasil ditampilkan di UI.
Skenario Alternatif	
Aksi Aktor	Reaksi Sistem
1. Pembeli meminta detail menu dengan ID yang tidak valid (misalnya bukan angka).	2. Sistem mendeteksi bahwa ID menu tidak valid.
	3. Sistem menolak permintaan dan mengirim pesan error.
4. UI menampilkan pesan error.	
Kondisi Akhir	Permintaan ditolak; tidak ada data yang ditampilkan.
Skenario Alternatif	
Aksi Aktor	Reaksi Sistem
1. Pembeli meminta detail sebuah menu.	2. Sistem mengalami gangguan saat mengambil data (misalnya kesalahan basis data).
	3. Sistem mengirim pesan kegagalan.
4. UI menampilkan notifikasi error/coba lagi.	
Kondisi Akhir	Detail menu gagal ditampilkan karena gangguan server.

Tabel 4.14 Mencari Menu

Identifikasi	
Kode	UC-04
Tujuan	Pembeli mencari menu (global / per-kios) dan/atau kios.
Deskripsi	Mengembalikan hasil pencarian berbasis keyword pada tabel menu (opsional join kios) dan tabel kios.
Aktor	Pembeli (guest).
Kondisi Awal	Pembeli membuka halaman pencarian
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Pembeli mengirimkan request pencarian (menu/kios) dengan keyword tertentu.	2. Menerima request
3.	4. Sistem memeriksa apakah kata kunci ada.
	5. Sistem mencari kecocokan pada data kios dan menu.
	6. Mengirim respon berhasil (hasil gabungan pencarian kios serta menu)
7. UI menampilkan hasil untuk tab Kios & Menu.	
Kondisi Akhir	Hasil gabungan (kios + menu) tampil sesuai keyword.
Skenario Utama	
Aksi Aktor	Reaksi Sistem

1. Pembeli melakukan request pencarian menu dalam kios tertentu dengan kata kunci.	2. Menerima request
3.	4. Sistem memvalidasi identitas kios
	5. memeriksa kata kunci.
	6. Sistem mencari menu pada kios tersebut sesuai kata kunci.
	7. Sistem menyiapkan daftar menu yang cocok.
8. UI menampilkan daftar menu khusus pada kios itu.	
Kondisi Akhir	Daftar menu pada kios yang dipilih berhasil ditampilkan sesuai kata kunci.
Skenario Alternatif	
Aksi Aktor	Reaksi Sistem
1. Pembeli melakukan pencarian tanpa mengisi kata kunci.	2. Sistem mendeteksi kata kunci kosong.
	3. Sistem menolak permintaan dan memberikan pesan error.
4. UI menampilkan pesan error.	
Kondisi Akhir	Permintaan ditolak; tidak ada hasil.
Skenario Alternatif	
Aksi Aktor	Reaksi Sistem
1. Pembeli melakukan pencarian dengan kata kunci.	2. Sistem mengalami gangguan (misalnya kesalahan basis data atau timeout).
	3. Sistem mengirimkan pesan kegagalan.
4. UI menampilkan notifikasi error / coba lagi.	
Kondisi Akhir	Pencarian gagal karena gangguan server; pengguna dapat mencoba ulang.

Tabel 4.15 Mengelola Keranjang

Identifikasi	
Kode	UC-05
Tujuan	Pembeli menambah, melihat, memperbarui, dan menghapus item di keranjang.
Deskripsi	Operasi add/update/delete item keranjang berbasis guest_id, membatasi item dari satu kios.
Aktor	Pembeli (guest)
Kondisi Awal	Halaman menu
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Pembeli request menambahkan menu ke keranjang dengan menentukan menu dan jumlah	2. Menerima request
	3. Sistem mengenali identitas pembeli (guest_id).
	4. memvalidasi menu yang dipilih.
	5. Sistem memeriksa apakah keranjang sudah berisi menu dari kios berbeda.
	6. Sistem mengambil informasi harga dan kios asal menu.
	7. Jika menu sudah ada di keranjang → sistem menambah jumlah dan memperbarui subtotal. Jika belum ada → sistem membuat item baru di keranjang.
	8. Sistem mengembalikan respons sukses (baru atau update).
9. UI menampilkan pesan "Ditambahkan ke keranjang" / "Jumlah diperbarui".	
Kondisi Akhir	Item keranjang berhasil tersimpan/terbarui. Keranjang tetap hanya berisi item dari satu kios.
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Pembeli request melihat isi keranjang.	2. Menerima request
	3. Sistem memvalidasi identitas pembeli(guest_id).

	4. Sistem menghitung total harga dan mengambil informasi kios asal menu.
	5. Sistem mengirimkan data keranjang lengkap.
6. UI menampilkan isi keranjang & total.	
Kondisi Akhir	Keranjang pembeli tampil lengkap dengan total harga.
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Pembeli request memperbarui jumlah pada item keranjang.	2. Menerima request
	3. Sistem memvalidasi identitas pembeli
	4. Memvalidasi keberadaan item di keranjang.
	5. Jika item tidak ditemukan → ditolak (404).
	6. Sistem memperbarui jumlah dan menghitung ulang subtotal.
	7. Sistem mengembalikan respons sukses.
8. UI menampilkan perubahan jumlah/catatan.	
Kondisi Akhir	Item keranjang diperbarui sesuai input; subtotal dan total harga konsisten.
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Pembeli request menghapus item dari keranjang.	2. Menerima request
	3. Sistem memvalidasi identitas pembeli
	4. Memvalidasi keberadaan item di keranjang..
	5. Jika item tidak ada → ditolak (404).
	6. Sistem menghapus item dari keranjang.
	7. Sistem mengirimkan respons sukses.
8. UI menghapus item dari daftar dan menghitung ulang total.	
Kondisi Akhir	Item hilang dari keranjang; total harga terhitung ulang.
Skenario Alternatif	
Aksi Aktor	Reaksi Sistem
1. Pembeli mencoba menambahkan menu dari kios lain.	2. Sistem mendeteksi bahwa keranjang sudah berisi item dari kios berbeda.
	3. Sistem menolak permintaan dan memberi peringatan.
4. UI menampilkan pesan agar pembeli mengosongkan keranjang terlebih dahulu.	
Kondisi Akhir	Keranjang tetap berisi item dari kios lama; pembeli diminta untuk mengosongkan keranjang sebelum pindah kios.
Skenario Alternatif	
Aksi Aktor	Reaksi Sistem
1. Pembeli menambah/melihat/mengubah keranjang	2. Sistem mengalami gangguan (misalnya error <i>database</i> atau <i>timeout</i>).
	3. Sistem mengirimkan pesan kegagalan.
4. UI menampilkan notifikasi error/coba lagi.	
Kondisi Akhir	Permintaan gagal diproses; tidak ada perubahan pada keranjang.

Tabel 4.16 Membuat Pesanan Dari Keranjang

Identifikasi	
Kode	UC-06
Tujuan	Pembeli membuat pesanan dari isi keranjang.
Deskripsi	Sistem memvalidasi data wajib dan isi keranjang, menghitung total harga dan estimasi waktu, membuat entri pesanan serta detail pesanan, lalu mengosongkan keranjang pembeli. Pesanan belum dianggap selesai/sukses sampai dilakukan pembayaran (ditangani di UC-08).
Aktor	Pembeli (guest).

Kondisi Awal	guest_id tersedia/terbentuk otomatis, keranjang berisi item dari satu kios.	
Skenario Utama		
Aksi Aktor		Reaksi Sistem
1. Memilih tipe pengantaran (diambil/diantar)		
2. Mengisi data wajib (guest_id, nama pemesan, nomor HP, dan opsional catatan/alamat).		
3. Pembeli request untuk membuat pesanan.		4. Menerima Request
		5. Sistem memvalidasi kelengkapan data wajib.
		6. Sistem memeriksa isi keranjang berdasarkan identitas pembeli.
		7. Jika keranjang berisi item → sistem menghitung total harga dan estimasi waktu.
		8. Sistem membuat data pesanan baru.
		9. Sistem menyalin item dari keranjang ke detail pesanan.
		10. Sistem mengosongkan keranjang pembeli.
		11. Sistem mengembalikan ringkasan pesanan dengan status awal <i>pending</i> serta instruksi pembayaran.
12. UI menampilkan ringkasan pesanan & pilihan metode pembayaran.		
Kondisi Akhir	Pesanan tercatat dengan status <i>pending</i> ; keranjang kosong; pembayaran belum dilakukan.	
Skenario Alternatif		
Aksi Aktor		Reaksi Sistem
1. Pembeli mencoba membuat pesanan tanpa mengisi salah satu data wajib (guest_id, tipe pengantaran, nama pemesan, nomor HP).		2. Sistem mendeteksi data tidak lengkap.
		3. Sistem menolak permintaan dan mengirim pesan error.
4. UI menampilkan pesan error.		:
Kondisi Akhir	Pesanan tidak dibuat; keranjang tetap seperti sebelumnya.	
Skenario Alternatif		
Aksi Aktor		Reaksi Sistem
1. Pembeli mencoba membuat pesanan saat keranjang kosong.		2. Sistem memeriksa isi keranjang dan menemukan tidak ada item.
		3. Sistem menolak permintaan dan mengirim pesan error "Keranjang kosong".
4. UI menampilkan informasi bahwa keranjang kosong.		
Kondisi Akhir	Pesanan tidak dibuat; keranjang tetap kosong.	
Skenario Alternatif		
Aksi Aktor		Reaksi Sistem
1. Pembeli mencoba membuat pesanan dengan data lengkap dan keranjang berisi item.		2. Terjadi kegagalan pada sistem saat memproses pesanan (misalnya saat menyimpan data).
		3. Sistem membatalkan proses pembuatan pesanan.
		4. Sistem mengirim pesan error <i>server</i> .
5. UI menampilkan notifikasi <i>error/coba lagi</i> .		
Kondisi Akhir	Pesanan tidak dibuat; keranjang tetap utuh; tidak ada perubahan data.	

Tabel 4.17 Menyediakan Payment Gateway

Identifikasi	
Kode	UC-07
Tujuan	Memfasilitasi pembeli melakukan pembayaran pesanan melalui integrasi sistem dengan payment gateway (Midtrans).
Deskripsi	Pembeli memilih metode pembayaran, lalu sistem berkomunikasi dengan Midtrans untuk membuat sesi pembayaran dan memberikan instruksi pembayaran (misalnya nomor Virtual Account atau QRIS) kepada

	pembeli. Setelah pembayaran dilakukan, Midtrans memberi notifikasi balik ke sistem untuk memperbarui status pesanan.
Aktor	Pembeli (guest)
Kondisi Awal	Pesanan tercatat dengan status <i>pending</i> ; keranjang kosong; pembayaran belum dilakukan.
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Pembeli mengirim request untuk memulai pembayaran dengan memilih metode pembayaran.	2. Memvalidasi request
	3. Mengambil detail pesanan dari <i>database</i>
	4. Membuat sesi pembayaran baru, lalu berkomunikasi dengan Midtrans untuk mendapatkan detail pembayaran (misalnya nomor Virtual Account atau QR code).
	5. Mengembalikan response ke pembeli berisi detail/instruksi pembayaran dari Midtrans.
6. Melakukan pembayaran sesuai instruksi	
	7. Sistem menerima notifikasi (<i>webhook</i>) dari Midtrans tentang status pembayaran.
	8. Memvalidasi notifikasi dan memperbarui status pesanan menjadi <i>paid</i> jika pembayaran berhasil
	9. Mengirim response sukses
10. Melihat status pembayaran berhasil.	
Kondisi Akhir	Pesanan berstatus <i>paid</i> , detail pembayaran tersimpan, pesanan siap diproses penjual.
Skenario Alternatif	
Aksi Aktor	Reaksi Sistem
1. Mengirim request untuk memulai pembayaran dengan memilih metode pembayaran.	2. Memvalidasi request dan mengambil detail pesanan.
	3. Membuat sesi pembayaran baru, lalu berkomunikasi dengan Midtrans untuk mendapatkan detail pembayaran (misalnya nomor Virtual Account atau QR code).
	4. Mengembalikan response ke pembeli berisi detail/instruksi pembayaran dari Midtrans.
5. Tidak menyelesaikan pembayaran atau pembayaran ditolak.	
	6. Midtrans mengirim notifikasi (<i>webhook</i>) ke sistem dengan status <i>expire</i> , <i>cancel</i> , atau <i>deny</i> .
	7. Memvalidasi notifikasi dan memperbarui status pesanan menjadi <i>failed/cancelled</i> .
	8. Mengirim response bahwa pembayaran gagal
9. Melihat status gagal di halaman konfirmasi atau riwayat pesanan.	
Kondisi Akhir	Pesanan berstatus <i>failed</i> ; tidak bisa diproses lebih lanjut.

Tabel 4.18 Menampilkan Status Dan Estimasi Waktu Pesanan

Identifikasi	
Kode	UC-08
Tujuan	Menampilkan status dan estimasi waktu penyelesaian pesanan.
Deskripsi	Pembeli memantau status & estimasi pesanan miliknya
Aktor	Pembeli (guest)
Kondisi Awal	Pesanan dibuat, Status awal pesanan = Pending. Estimasi awal dihitung berdasarkan item pesanan dan antrian kios.
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Pembeli membuat pesanan baru.	2. Sistem memverifikasi data pesanan dan menyimpannya.
	3. Sistem menetapkan status awal pesanan sebagai <i>pending</i> / <i>menunggu diproses</i> .

	4. Sistem menghitung estimasi waktu selesai berdasarkan total estimasi dari item pesanan dan antrean kios.
	5. Sistem mengembalikan respons sukses berisi detail pesanan, status awal, dan estimasi waktu selesai.
6. UI langsung menampilkan ringkasan pesanan dengan status awal + estimasi.	
	7. Sistem mengubah status pesanan menjadi Processing dan memulai timer estimasi waktu.
8. Pembeli melihat status berubah dari <i>Pending</i> → <i>Processing</i> serta melihat estimasi waktu berjalan.	
	9. Sistem mengubah status pesanan menjadi Completed .
10. Pembeli melihat status akhir pesanan = <i>Selesai</i> .	
Kondisi Akhir	Pembeli selalu bisa melihat status terbaru pesanan (<i>Pending</i> → <i>Processing</i> → <i>Completed</i>) dan estimasi waktu berjalan.

Tabel 4.19 Mengirim Notifikasi Pesanan Selesai

Identifikasi	
Kode	UC-09
Tujuan	Mengirim WA ke pembeli saat pesanan berstatus done .
Deskripsi	Setelah status jadi done (UC-13), sistem mengambil data pesanan & detail, menyusun pesan, lalu mengirim lewat WhatsApp.
Aktor	Sistem (trigger otomatis); Pembeli (penerima).
Kondisi Awal	Pesanan done ; data pembeli (nama, no_hp) dan detail item tersedia; layanan WA aktif.
Skenario Utama	
Aksi Aktor	Reaksi Sistem
	1. Sistem mendeteksi pesanan telah selesai.
	2. Sistem mengambil data pesanan (nama pemesan, kontak, total harga, tipe pengantaran, alamat).
	3. Sistem mengambil detail menu pesanan (nama menu, jumlah, harga).
	4. Sistem menyusun ringkasan pesanan dan pesan notifikasi.
	5. Sistem mengirimkan pesan WhatsApp ke nomor pembeli dan mencatat log pengiriman.
6. Pembeli menerima notifikasi WA berisi informasi bahwa pesanan sudah selesai.	
Kondisi Akhir	Pembeli menerima pesan WhatsApp berisi ringkasan pesanan dan total pembayaran; sistem menyimpan log notifikasi sebagai catatan.

Tabel 4.20 Mengelola Menu

Identifikasi	
Kode	UC-10
Tujuan	Penjual menambah, melihat, memperbarui, dan menghapus menu.
Deskripsi	Operasi CRUD menu dengan JWT; dukung unggah gambar ke Cloudinary.
Aktor	Penjual (JWT) .
Kondisi Awal	Penjual login ; req.user.id (dan kios_id untuk tambah) tersedia; DB & Cloudinary online .
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Penjual request meminta daftar menu miliknya.	2. Menerima request
	3. Memvalidasi identitas penjual.
	4. Sistem mengambil semua menu yang dimiliki penjual.
	5. Sistem mengirimkan daftar menu ke UI.
6. UI menampilkan daftar menu.	
Kondisi Akhir	Daftar menu tampil sesuai milik penjual.
Skenario Utama	
Aksi Aktor	Reaksi Sistem

1. Penjual request menambah menu baru (nama, deskripsi, harga, estimasi, status tersedia, foto opsional).	2. Menerima request
	3. Sistem mengenali identitas penjual dan kios terkait
	4. Sistem menyimpan foto menu ke media storage (jika ada)
	5. Sistem menetapkan nilai default untuk data yang kosong (misal: harga = 0, estimasi waktu = default, status = tersedia)
	6. Sistem menyimpan data menu baru ke <i>database</i>
	7. Sistem mengirimkan konfirmasi sukses.
8. UI menampilkan menu baru.	
Kondisi Akhir	Menu baru berhasil tersimpan, termasuk foto jika diunggah.
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Penjual request update data pada menu tertentu (field tertentu + foto opsional).	2. Menerima request
	3. Sistem mengenali identitas penjual dan kios terkait
	4. Sistem memverifikasi kepemilikan menu.
	5. Jika ada foto baru → sistem menyimpan foto baru & menghapus foto lama.
	6. Sistem memperbarui data menu sesuai input.
	7. Sistem mengirim konfirmasi sukses dengan data terbaru.
8. UI menampilkan data terbaru.	
Kondisi Akhir	Perubahan menu berhasil tersimpan; foto lama dihapus jika diganti.
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Penjual request melihat detail menu tertentu.	2. Menerima request
	3. Sistem mengenali identitas penjual dan kios terkait
	4. Sistem memverifikasi kepemilikan menu
	5. Jika ditemukan → sistem mengirim detail menu.
6. UI menampilkan detail menu.	
Kondisi Akhir	Detail menu tampil bila menu benar milik penjual.
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Penjual request menghapus menu tertentu.	2. Menerima request
	3. Sistem mengenali identitas penjual dan kios terkait
	4. Sistem memverifikasi kepemilikan menu.
	5. Jika menu punya foto → sistem menghapus asset foto.
	6. Sistem menghapus data menu.
	7. Sistem mengirim konfirmasi sukses.
8. UI menghapus menu dari daftar.	
Kondisi Akhir	Menu terhapus; foto juga ikut dihapus jika ada.
Skenario Alternatif	
Aksi Aktor	Reaksi Sistem
1. Penjual melakukan request dengan <i>token</i> tidak valid.	2. Sistem menolak permintaan dengan error 401 <i>Unauthorized</i> atau 403 <i>Forbidden</i> .
Kondisi Akhir	Akses ditolak; penjual tidak bisa mengakses data menu.

Tabel 4.21 Mengirim Notifikasi Pesanan Masuk

Identifikasi	
Kode	UC-11

Tujuan	Memberitahu penjual bahwa ada pesanan baru/terbayar dan mengarahkan ke dashboard untuk diproses
Deskripsi	Saat pesanan dibuat/masuk status paid notif masuk ke penjual
Aktor	Sistem (pengirim otomatis), Penjual (penerima notifikasi).
Kondisi Awal	Pesanan baru/paid terdeteksi; kiosId & pesananId valid; no_hp penjual tersedia; DB & layanan WA aktif; belum terlog (belum pernah terkirim).
Skenario Utama	
Aksi Aktor	Reaksi Sistem
	1. Sistem mendeteksi pesanan baru atau status pesanan menjadi paid
	2. Sistem memeriksa apakah notifikasi untuk pesanan ini sudah pernah dikirim
	3. Sistem mengambil informasi penjual yang terkait dengan kios dari pesanan.
	4. Sistem mendapatkan nomor WhatsApp penjual.
	5. Sistem menyusun pesan notifikasi dan tautan dashboard.
	6. Sistem mengirimkan pesan WhatsApp ke penjual.
	7. Sistem mencatat log bahwa notifikasi sudah dikirim.
8. Penjual menerima pesan WA berisi notifikasi "Pesanan Baru!".	
Kondisi Akhir	Penjual menerima notifikasi WhatsApp berisi informasi pesanan baru; log notifikasi tersimpan untuk mencegah duplikasi pengiriman.

Tabel 4.22 Menampilkan Daftar Pesanan Dan Detail Pesanan

Identifikasi	
Kode	UC-12
Tujuan	Penjual melihat daftar pesanan aktif dan detail tiap pesanan.
Deskripsi	Penjual dapat melihat daftar pesanan yang masuk dan detail dari tiap pesanan
Aktor	Penjual (JWT).
Kondisi Awal	Penjual login; DB <i>online</i> ; terdapat data pesanan (bisa kosong untuk daftar).
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Penjual request daftar pesanan	2. Menerima request
	3. Sistem memverifikasi identitas penjual.
	4. Sistem mengambil daftar pesanan aktif milik kios penjual (status: <i>paid</i> , <i>processing</i> , <i>ready</i> , <i>delivering</i>) secara berurutan.
	5. Sistem menambahkan metadata seperti nomor antrian dan format tanggal/status.
	6. Sistem mengirimkan data daftar pesanan ke UI.
7. UI menampilkan daftar pesanan beserta nomor antrian.	
Kondisi Akhir	Daftar pesanan aktif tampil dengan metadata, dan nomor antrian.
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. penjual request detail dari satu pesanan tertentu.	2. Menerima request
	3. Sistem memverifikasi identitas penjual.
	4. Sistem mengambil detail pesanan termasuk item, jumlah, harga, metode bayar, dan tipe pengantaran.
	5. Sistem mengirimkan data detail pesanan.
6. UI menampilkan detail pesanan lengkap dengan estimasi.	
Kondisi Akhir	Detail pesanan aktif tampil lengkap dengan metadata, estimasi, dan item pesanan.
Skenario Alternatif	
Aksi Aktor	Reaksi Sistem
1. Penjual meminta daftar pesanan masuk	2. Sistem membaca identitas penjual dan mendeteksi bahwa data tidak valid.

	3. Sistem menolak permintaan dan mengirim error.
4. UI menampilkan pesan error.	
Kondisi Akhir	Permintaan ditolak; tidak ada data ditampilkan.
Skenario Alternatif	
Aksi Aktor	Reaksi Sistem
1. Penjual meminta detail pesanan tertentu	2. Terjadi kegagalan pada sistem saat memproses pesanan (misalnya saat menyimpan data).
	3. Sistem mengirim error <i>server</i>
4. UI menampilkan notifikasi error/coba lagi.	
Kondisi Akhir	Detail pesanan tidak ditampilkan karena terjadi kesalahan pada <i>server</i> .

Tabel 4.23 Mengubah Status Pesanan

Identifikasi	
Kode	UC-13
Tujuan	Penjual memperbarui status pesanan.
Deskripsi	Update status dengan validasi akses; khusus status processing sistem set waktu mulai & estimasi selesai. Saat status done, sistem memicu notifikasi ke pembeli.
Aktor	Penjual (JWT), Sistem (pemicu notifikasi saat done).
Kondisi Awal	Penjual login; :id pesanan valid & milik kios penjual; DB <i>online</i> .
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Penjual request mengubah status pesanan .	2. Menerima request
	3. Sistem memverifikasi identitas penjual.
	4. Sistem memvalidasi bahwa status yang diminta valid (harus termasuk dalam daftar status yang diperbolehkan: <i>pending, paid, processing, ready, delivering, done</i>).
	5. Sistem mengambil data pesanan dan memverifikasi bahwa pesanan tersebut memang milik kios penjual.
	6. Sistem memperbarui status pesanan
	7. Sistem mengembalikan respons sukses berisi status baru
8. UI menampilkan status terbaru	
Kondisi Akhir	Pesanan berhasil diperbarui statusnya

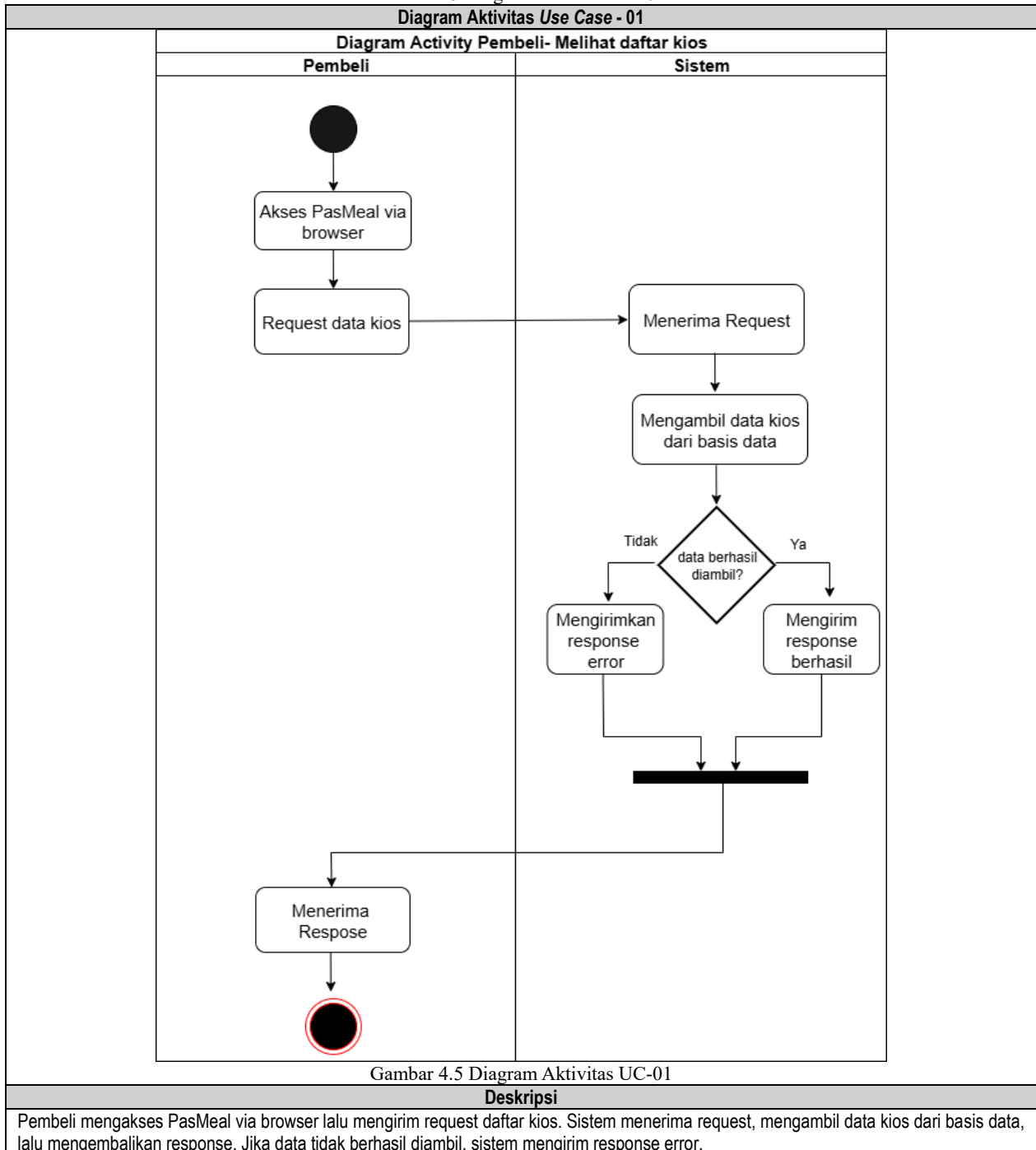
Tabel 4.24 Menampilkan Tipe Pengantaran

Identifikasi	
Kode	UC-14
Tujuan	Penjual melihat tipe pengantaran tiap pesanan (Ambil Sendiri / Diantar) untuk operasional.
Deskripsi	Tipe pengantaran dibuat untuk memudahkan penjual dalam mengantarkan makanan
Aktor	Penjual (JWT)
Kondisi Awal	Penjual login; DB <i>online</i> ; ada data pesanan (bisa kosong).
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Penjual request melihat tipe pengantaran	2. Sistem menerima request
	3. Sistem validasi data penjual
	4. Sistem mengambil detail pesanan termasuk item, jumlah, harga, metode bayar, dan tipe pengantaran.
	5. Sistem mengembalikan data daftar pesanan lengkap dengan metadata
6. UI menampilkan detail pesanan beserta kolom Tipe Pengantaran.	
Kondisi Akhir	kolom Tipe Pengantaran muncul sesuai jenis pengantaran (alamat tujuan atau "Ambil Sendiri").

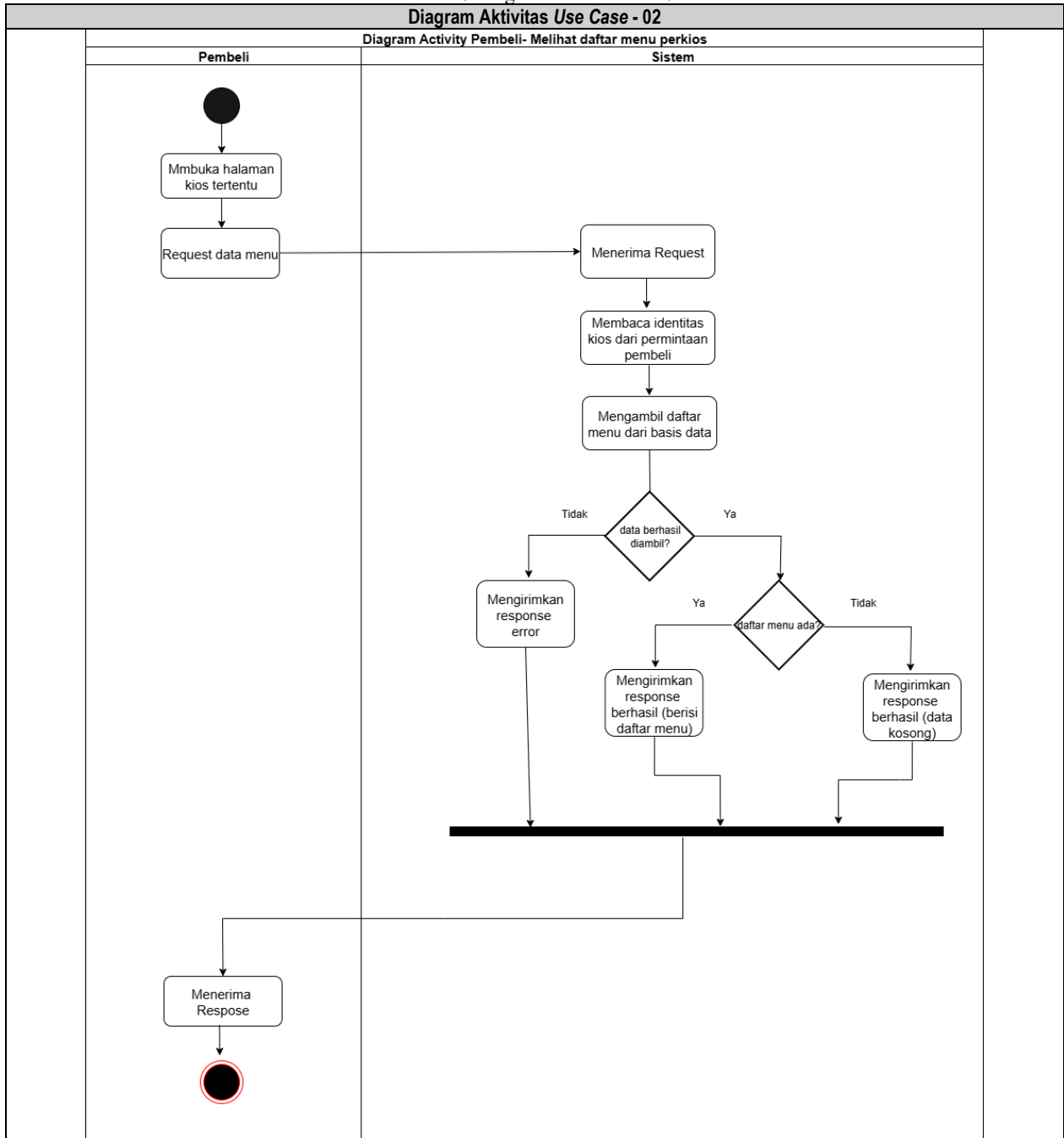
4.1.5.5 Diagram Aktivitas

Diagram Aktivitas digunakan untuk menggambarkan berbagai aliran aktivitas dalam sistem yang dirancang, bagaimana suatu proses dimulai, *decision* yang mungkin muncul dan bagaimana suatu proses diakhiri.

Tabel 4.25 Diagram Aktivitas UC-01



Tabel 4.26 Diagram Aktivitas UC-02

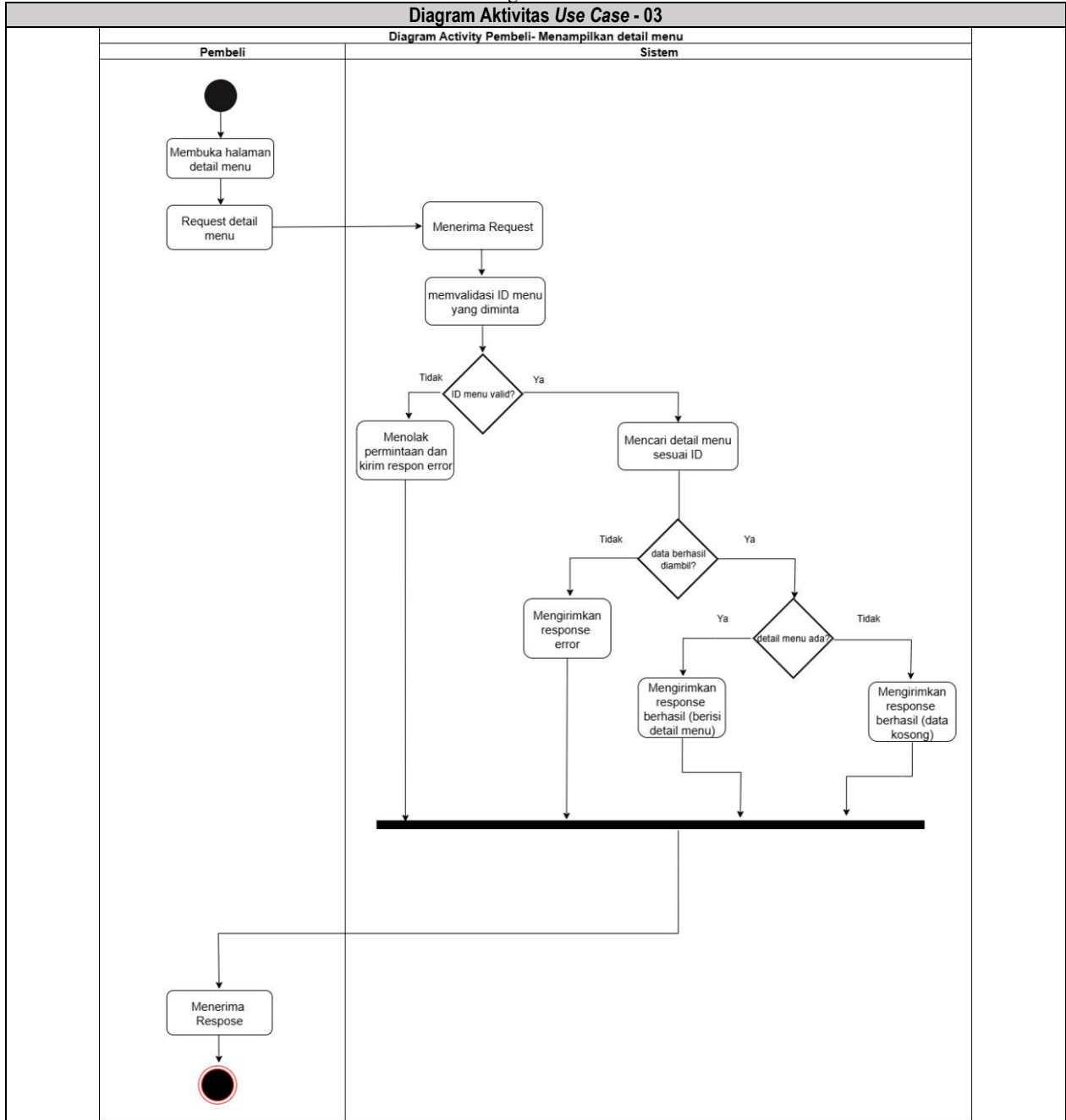


Gambar 4.6 Diagram Aktivitas Uc-02

Deskripsi

Pembeli membuka halaman kios tertentu dan mengirim request data menu. Sistem membaca identitas kios, mengambil daftar menu dari basis data, lalu mengembalikan response. Jika data ada maka ditampilkan daftar menu, jika tidak ada maka response berisi data kosong, dan bila gagal diambil sistem mengirim error.

Tabel 4.27 Diagram Aktivitas Uc-03

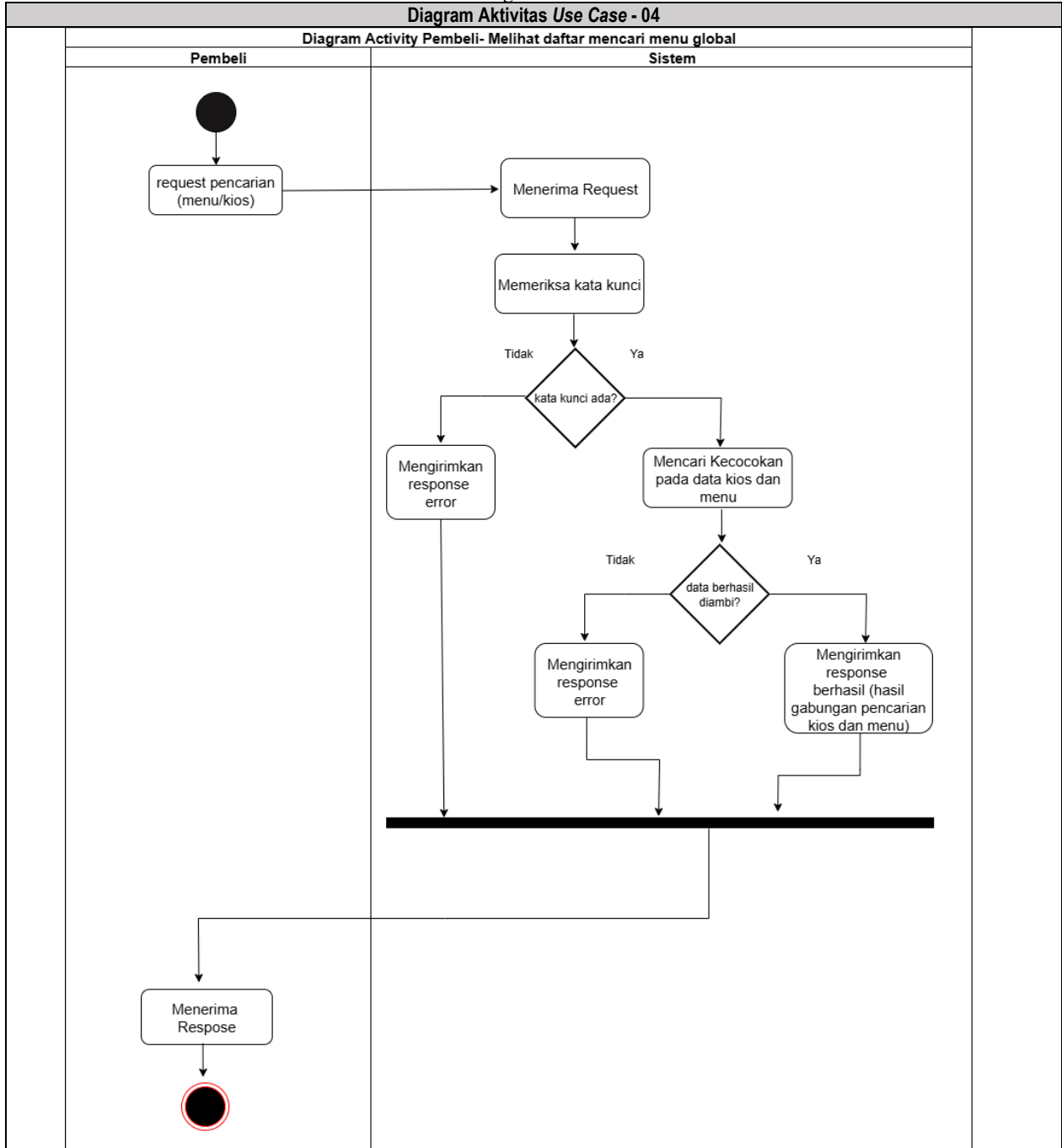


Gambar 4.7 Diagram Aktivitas Uc-03

Deskripsi

Pembeli membuka halaman detail menu dan meminta data sesuai ID menu. Sistem memvalidasi ID menu, kemudian mencari detail menu di basis data. Jika valid dan data ditemukan, sistem menampilkan detail menu; jika tidak valid atau gagal diambil, sistem mengirimkan error; bila menu tidak tersedia, sistem mengembalikan response kosong.

Tabel 4.28 Diagram Aktifitas Uc-04a



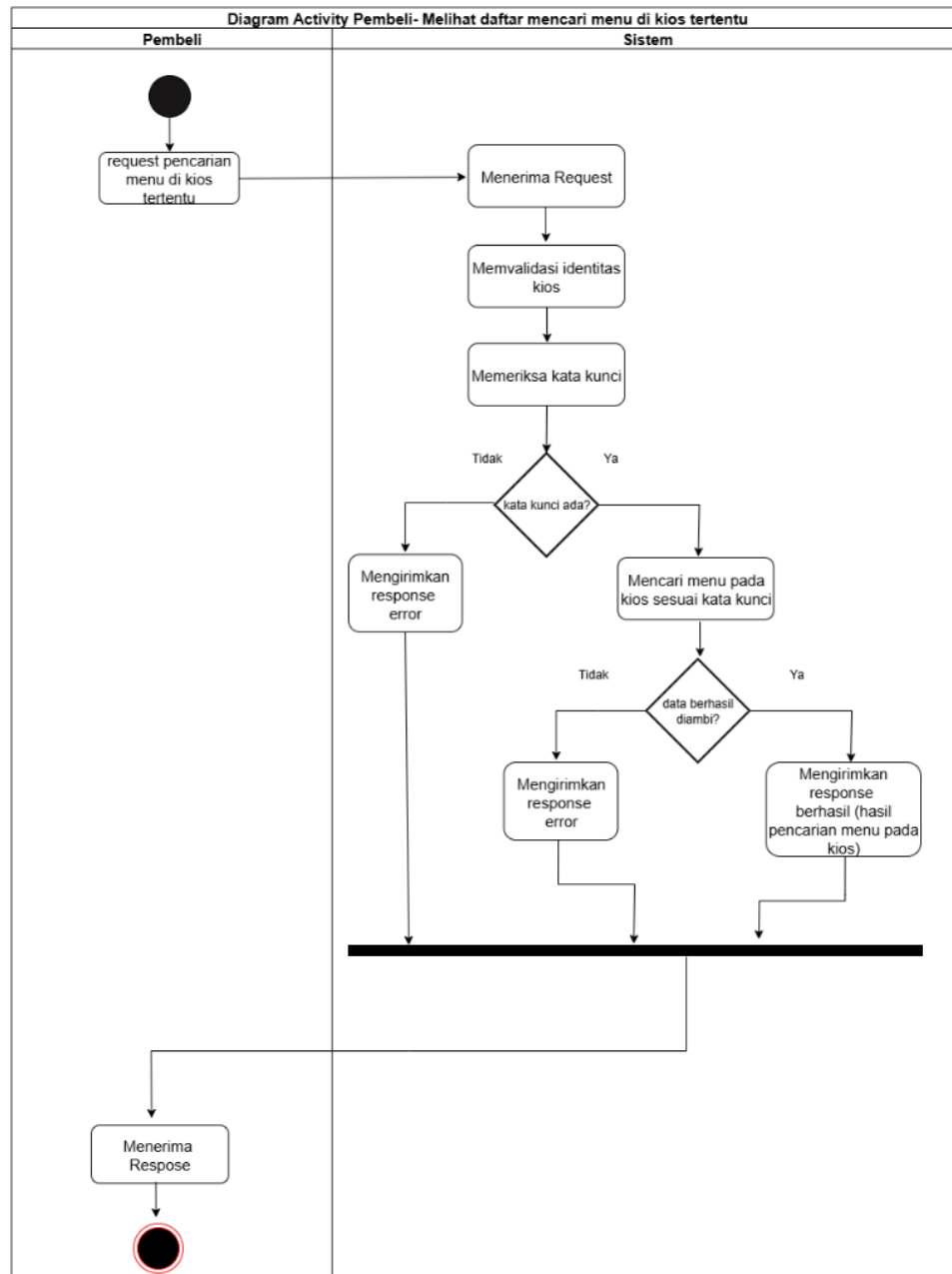
Gambar 4.8 Diagram Aktifitas Uc-04a

Deskripsi

Pembeli melakukan request pencarian menu/kios menggunakan kata kunci. Sistem memeriksa kata kunci, lalu mencari kecocokan pada data kios dan menu. Jika data ditemukan, sistem mengembalikan hasil pencarian gabungan kios dan menu; jika gagal atau kata kunci kosong, sistem mengirim error.

Tabel 4.29 Diagram Aktifitas Uc-04b

Diagram Aktivitas Use Case - 04

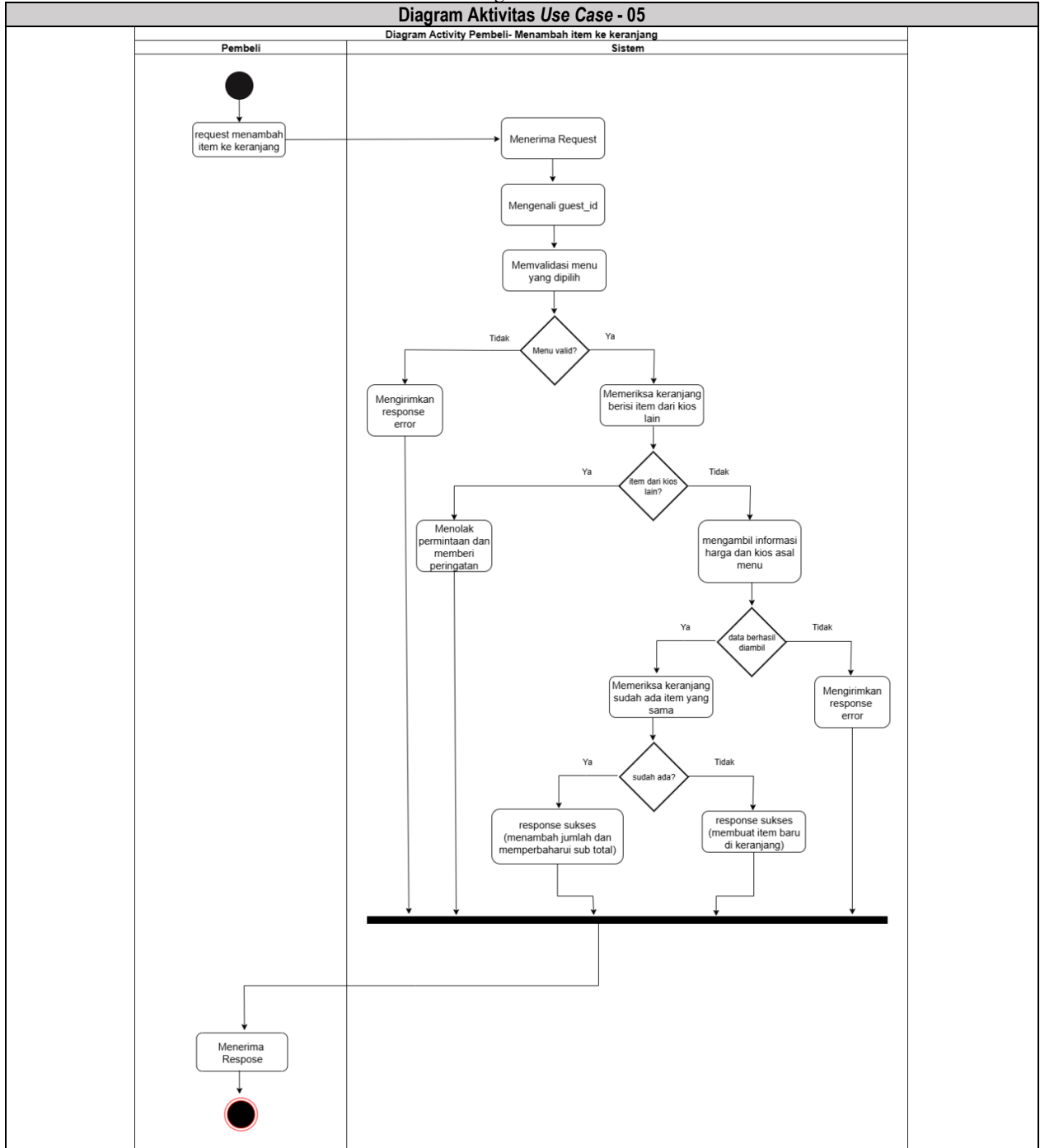


Gambar 4.9 Diagram Aktifitas Uc-04b

Deskripsi

Pembeli melakukan request pencarian menu dalam kios tertentu. Sistem memvalidasi identitas kios, memeriksa kata kunci, lalu mencari menu sesuai kata kunci di kios tersebut. Jika data ditemukan, sistem mengembalikan hasil pencarian; jika kata kunci tidak ada atau data gagal diambil, sistem mengirim error.

Tabel 4.30 Diagram aktivitas uc-05a

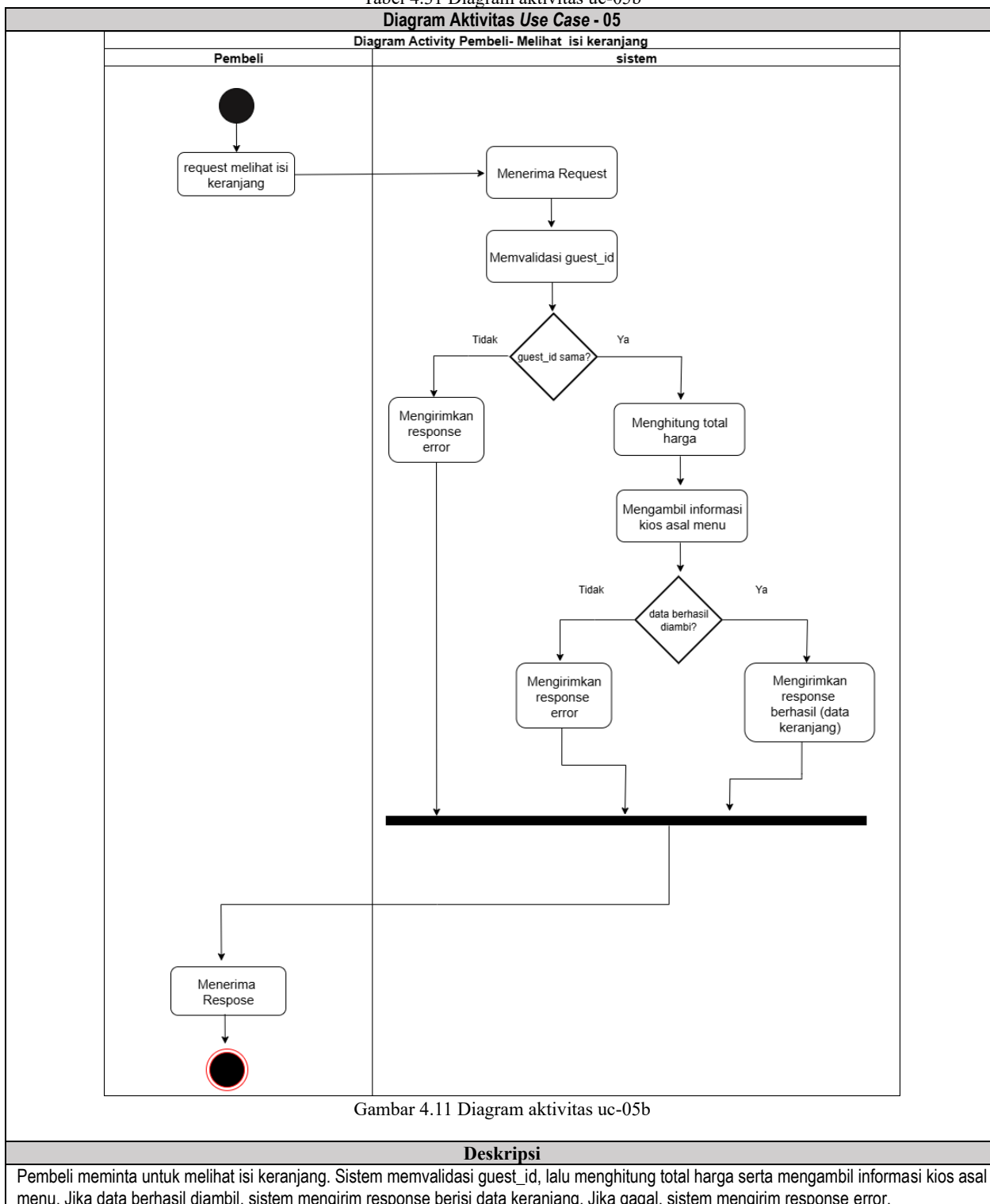


Gambar 4.10 Diagram aktivitas uc-05a

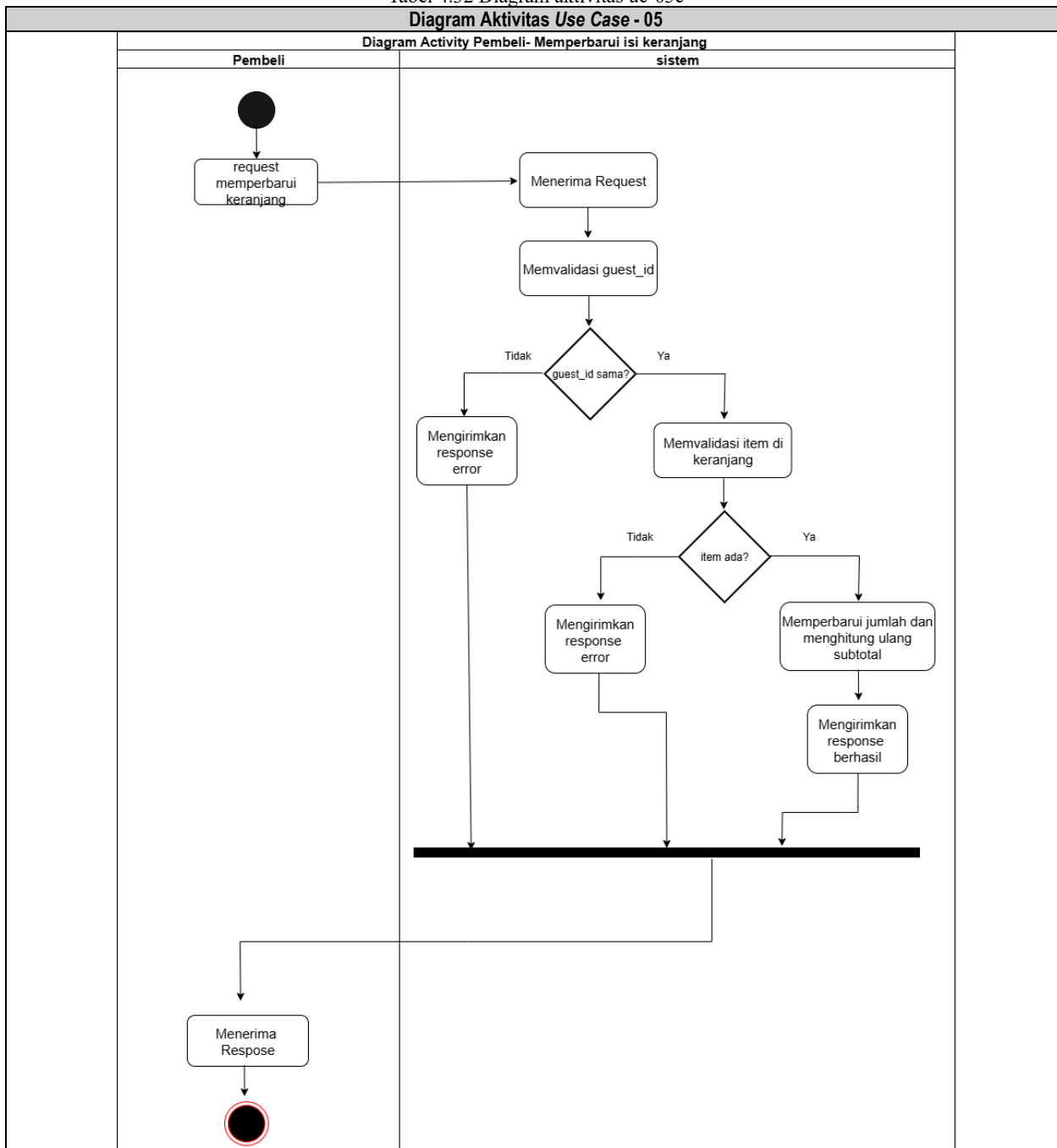
Deskripsi

Pembeli mengirim request untuk menambahkan menu ke keranjang. Sistem mengenali guest_id, memvalidasi menu, serta memastikan keranjang tidak berisi item dari kios lain. Jika valid, sistem mengecek apakah item sudah ada. Bila sudah ada maka jumlah diperbarui, jika belum maka item baru ditambahkan. Jika terjadi kesalahan validasi, sistem mengirim response error.

Tabel 4.31 Diagram aktivitas uc-05b



Tabel 4.32 Diagram aktivitas uc-05c

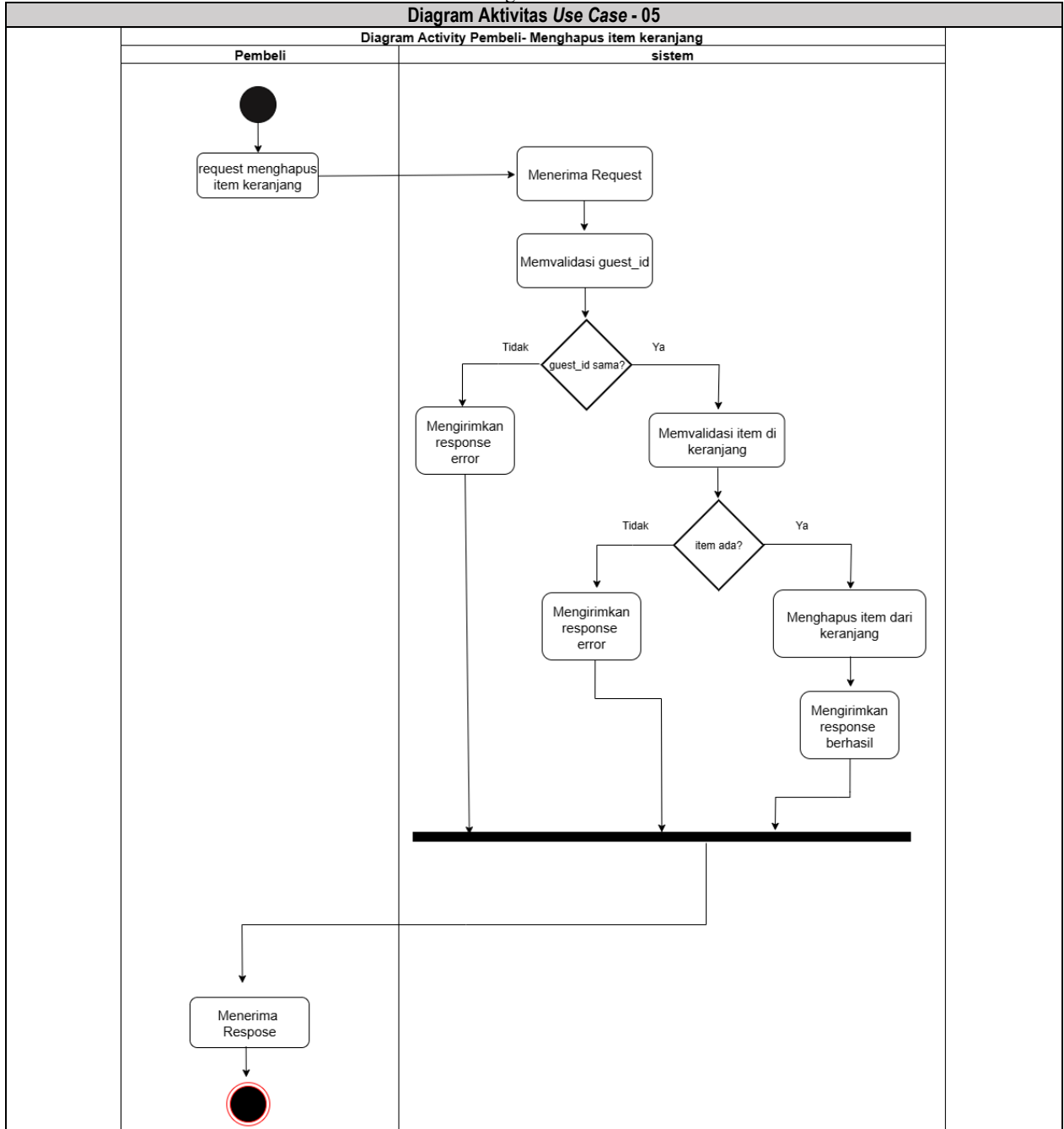


Gambar 4.12 Diagram aktivitas uc-05c

Deskripsi

Pembeli mengirim request untuk memperbarui keranjang. Sistem memvalidasi guest_id dan mengecek keberadaan item. Jika item ada, sistem memperbarui jumlah serta menghitung ulang subtotal, kemudian mengirim response berhasil. Jika validasi gagal, sistem memberikan response error.

Tabel 4.33 Diagram aktivitas uc-05d

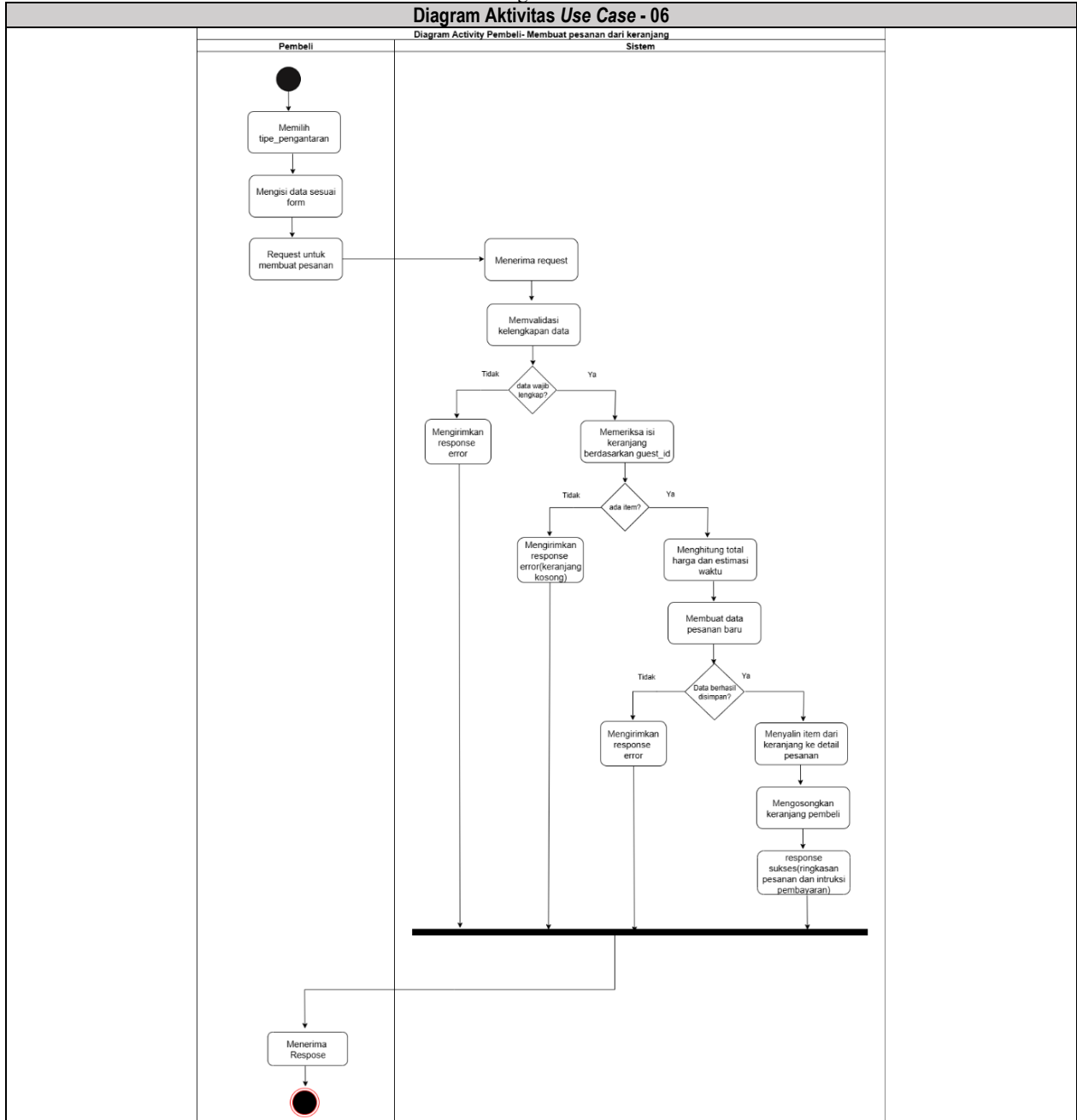


Gambar 4.13 Diagram aktivitas uc-05d

Deskripsi

Pembeli mengirim request untuk menghapus item. Sistem memvalidasi guest_id dan keberadaan item dalam keranjang. Jika valid dan item ada, sistem menghapus item lalu mengirim response berhasil. Jika tidak, sistem mengirim response error.

Tabel 4.34 Diagram aktivitas uc-06

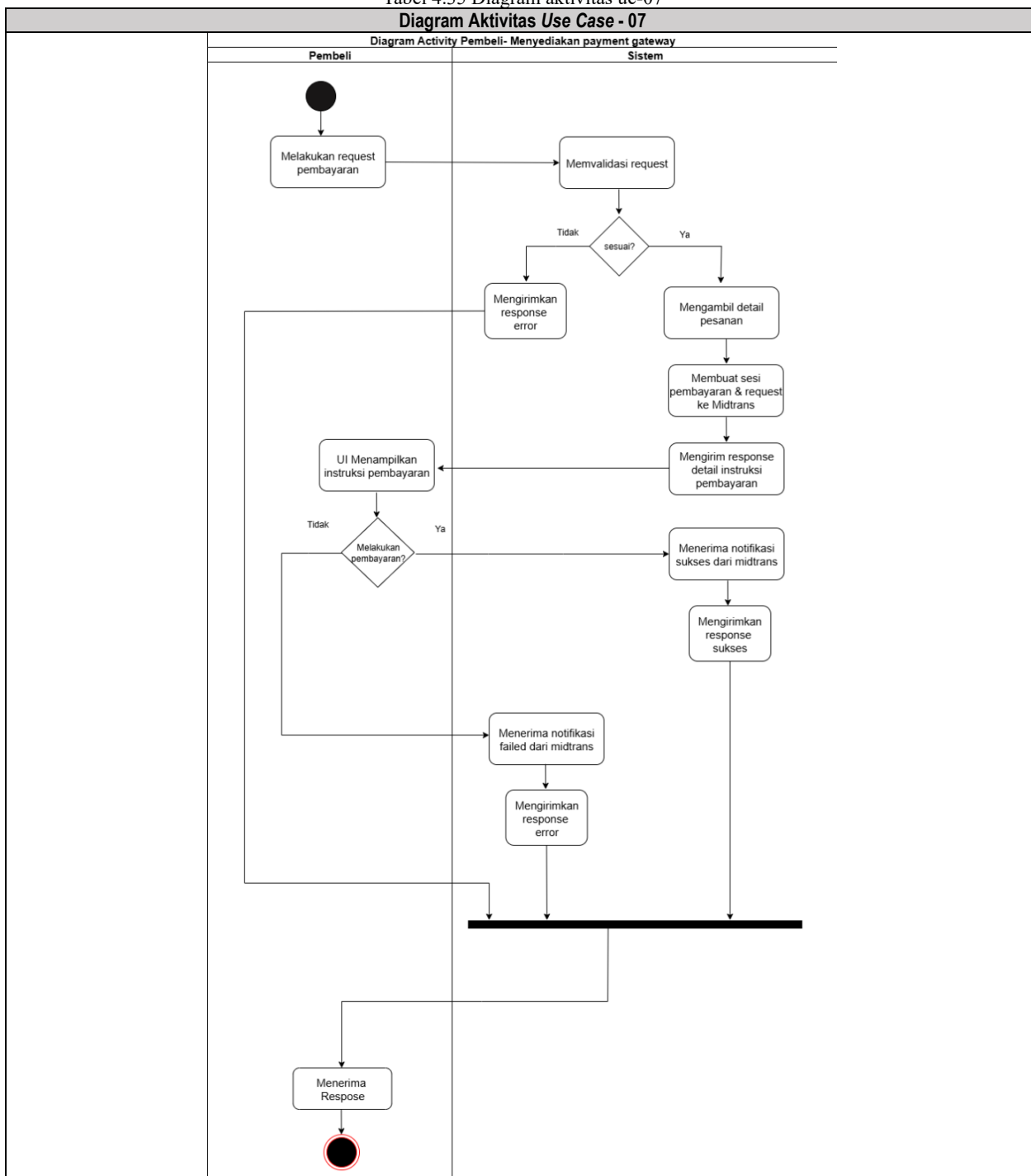


Gambar 4.14 Diagram aktivitas uc-06

Deskripsi

Pembeli memilih tipe pengantaran, mengisi form, lalu mengirim request untuk membuat pesanan. Sistem memvalidasi kelengkapan data dan memeriksa isi keranjang berdasarkan guest_id. Jika data valid dan keranjang berisi item, sistem menghitung total harga serta estimasi waktu, membuat data pesanan baru, menyalin item ke detail pesanan, mengosongkan keranjang, lalu mengirim response sukses berisi ringkasan pesanan dan instruksi pembayaran.

Tabel 4.35 Diagram aktivitas uc-07

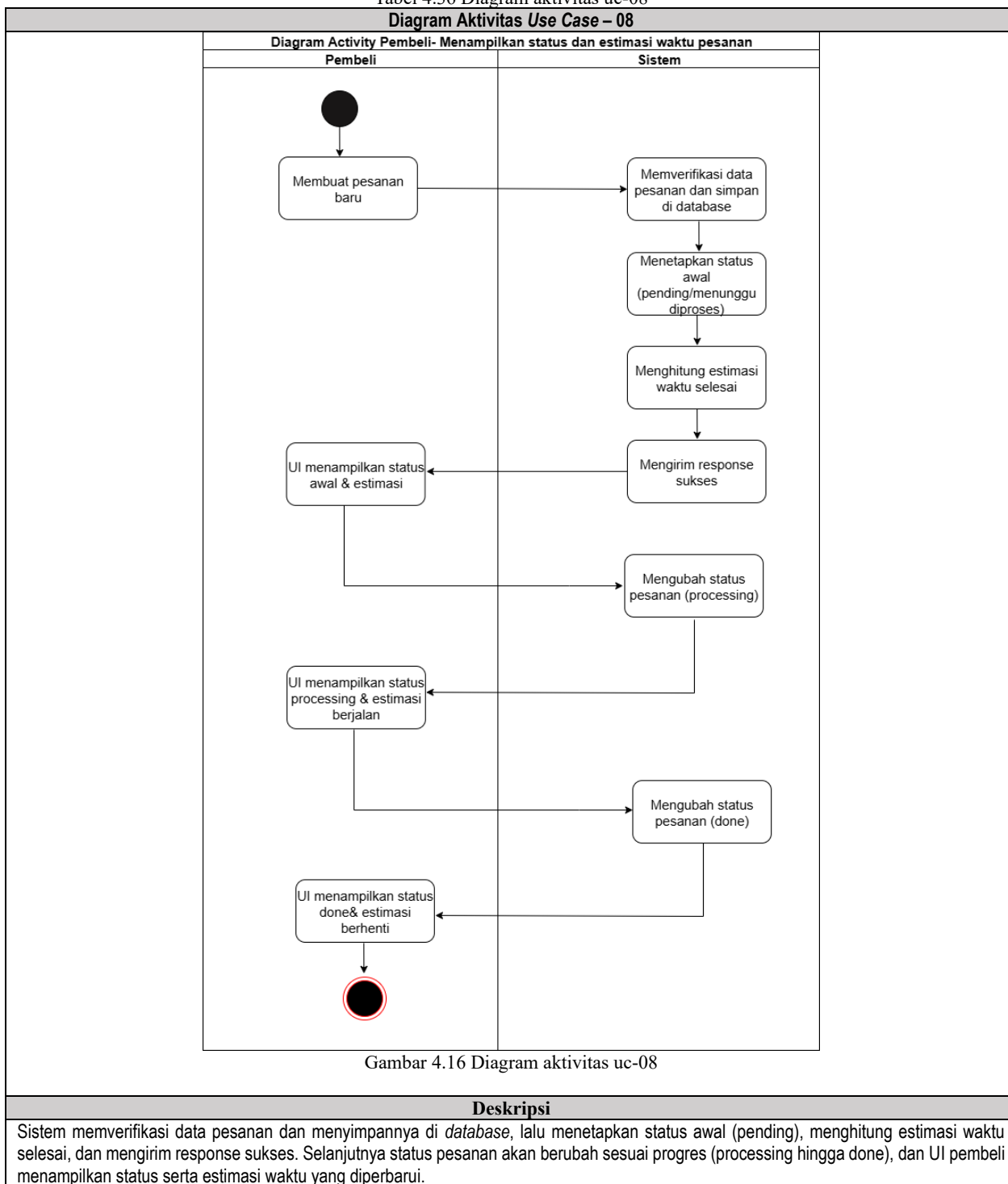


Gambar 4.15 Diagram aktivitas uc-07

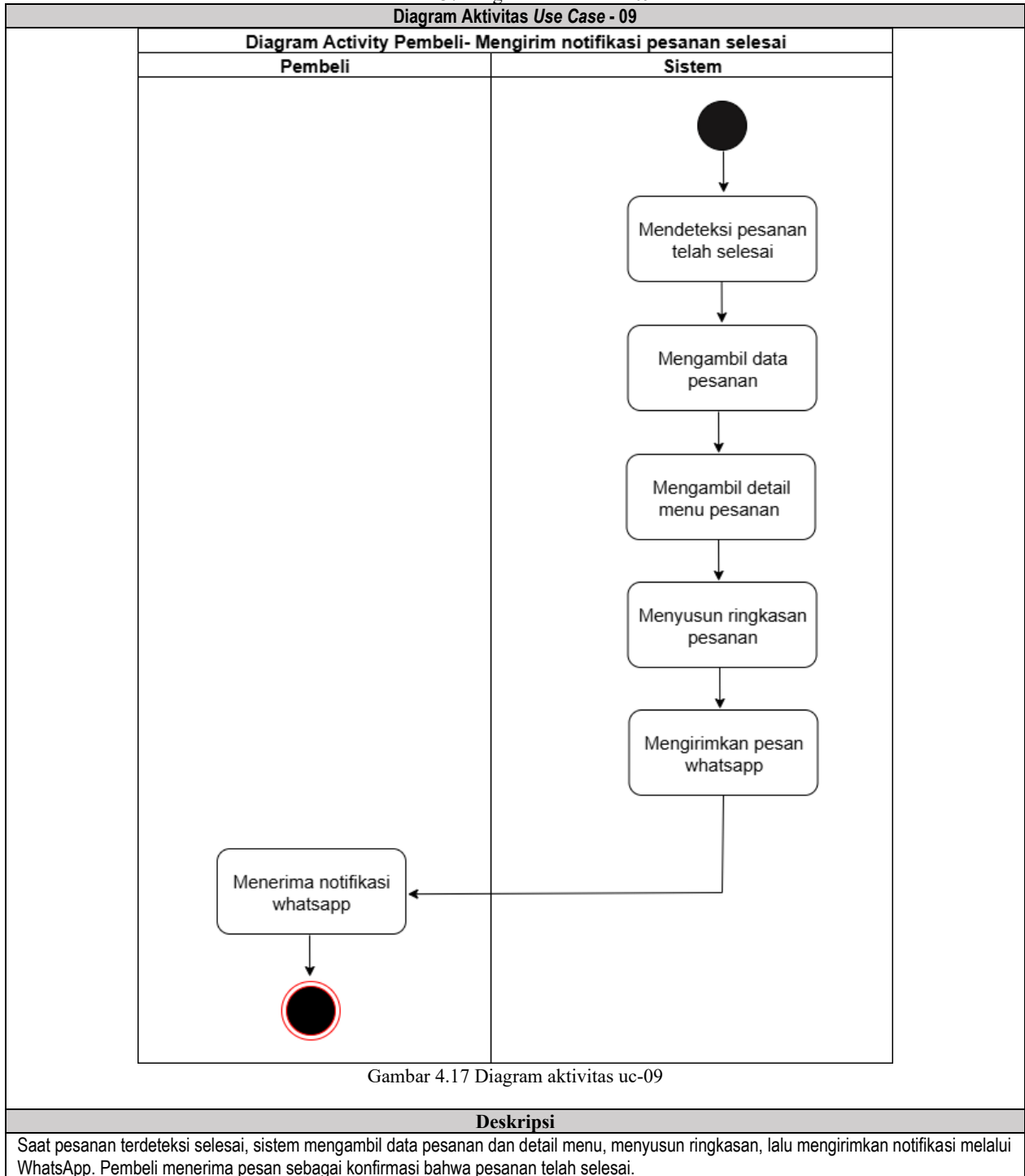
Deskripsi

Setelah itu, pembeli melakukan request pembayaran. Sistem memvalidasi request, mengambil detail pesanan, lalu membuat sesi pembayaran melalui Midtrans dan mengembalikan instruksi pembayaran. Jika pembeli melanjutkan pembayaran dan Midtrans mengirim notifikasi sukses, sistem mengembalikan response berhasil; jika gagal, sistem mengembalikan error.

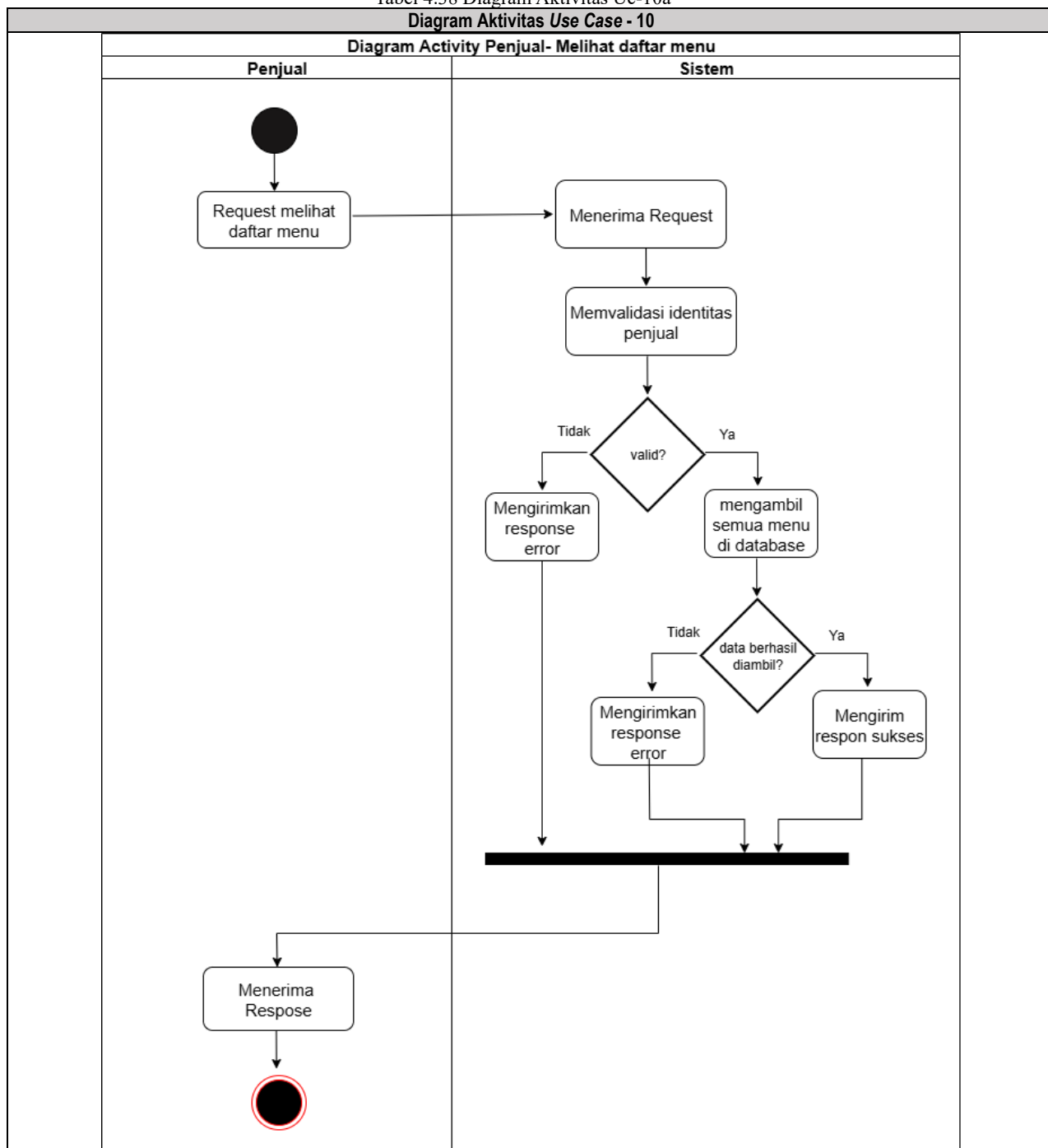
Tabel 4.36 Diagram aktivitas uc-08



Tabel 4.37 Diagram aktivitas uc-09



Tabel 4.38 Diagram Aktivitas Uc-10a

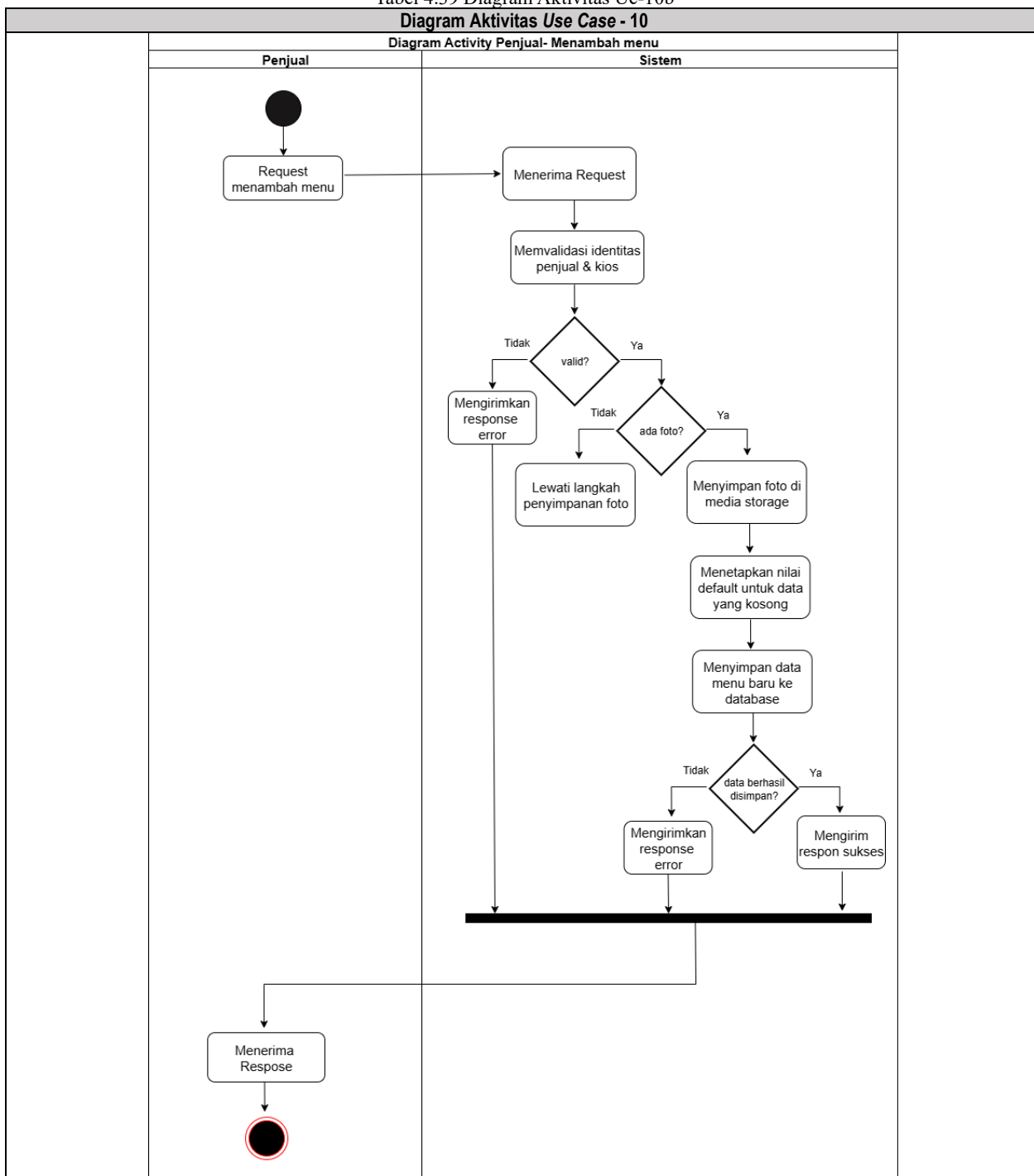


Gambar 4.18 Diagram Aktivitas Uc-10a

Deskripsi

Penjual mengirim request untuk melihat seluruh menu yang dimilikinya. Sistem memvalidasi identitas penjual lalu mengambil semua menu dari *database*. Jika data berhasil diambil, sistem mengirim respon sukses berisi daftar menu; jika gagal, sistem mengirim respon error.

Tabel 4.39 Diagram Aktivitas Uc-10b

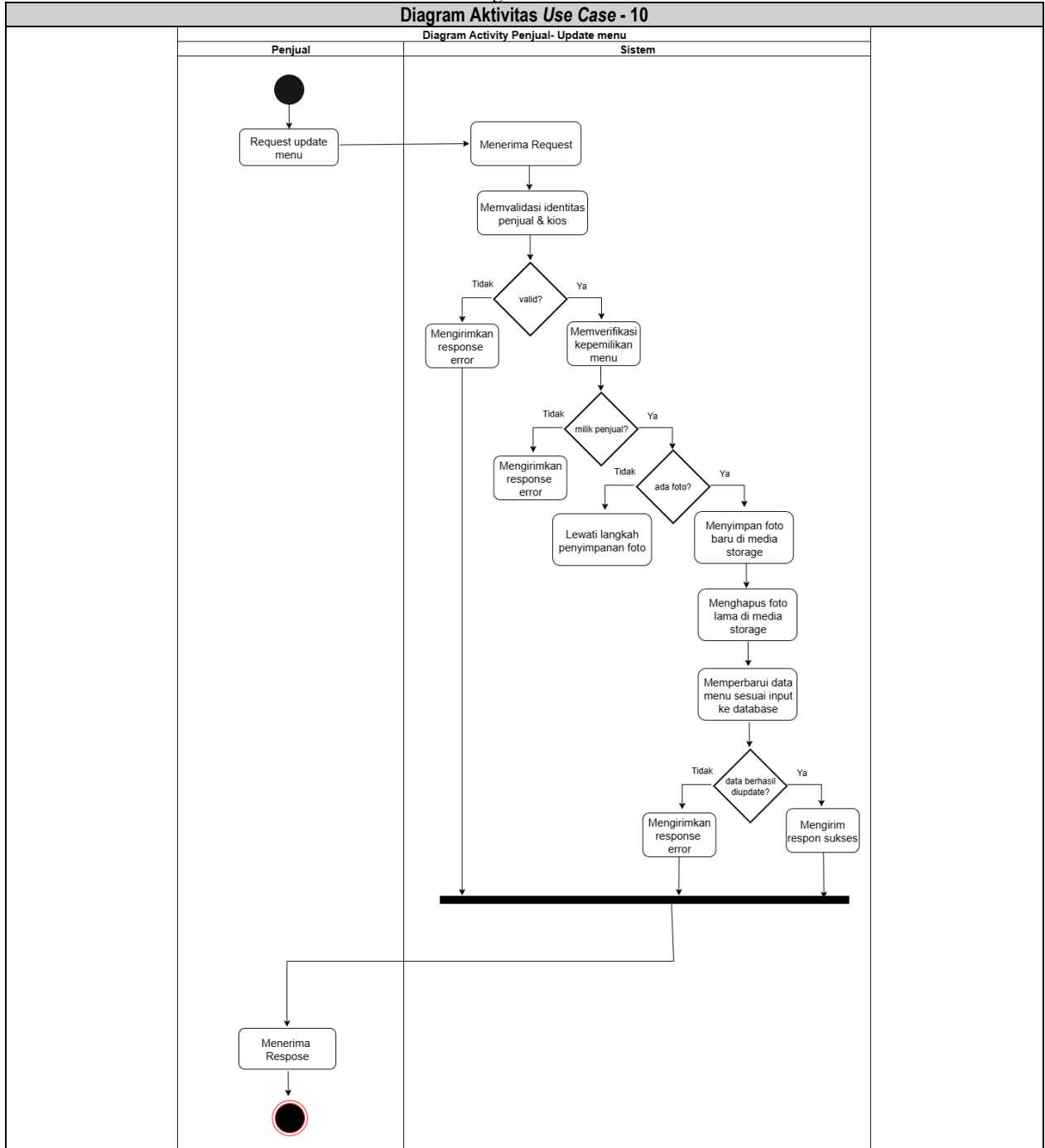


Gambar 4.19 Diagram Aktivitas Uc-10b

Deskripsi

Penjual mengajukan request menambah menu baru. Sistem memvalidasi identitas penjual dan kios, lalu menyimpan foto jika ada, menetapkan nilai default untuk data kosong, dan menyimpan data baru ke *database*. Jika penyimpanan berhasil, sistem mengirim respon sukses; jika gagal, sistem memberikan respon error.

Tabel 4.40 Diagram Aktivitas Uc-10c

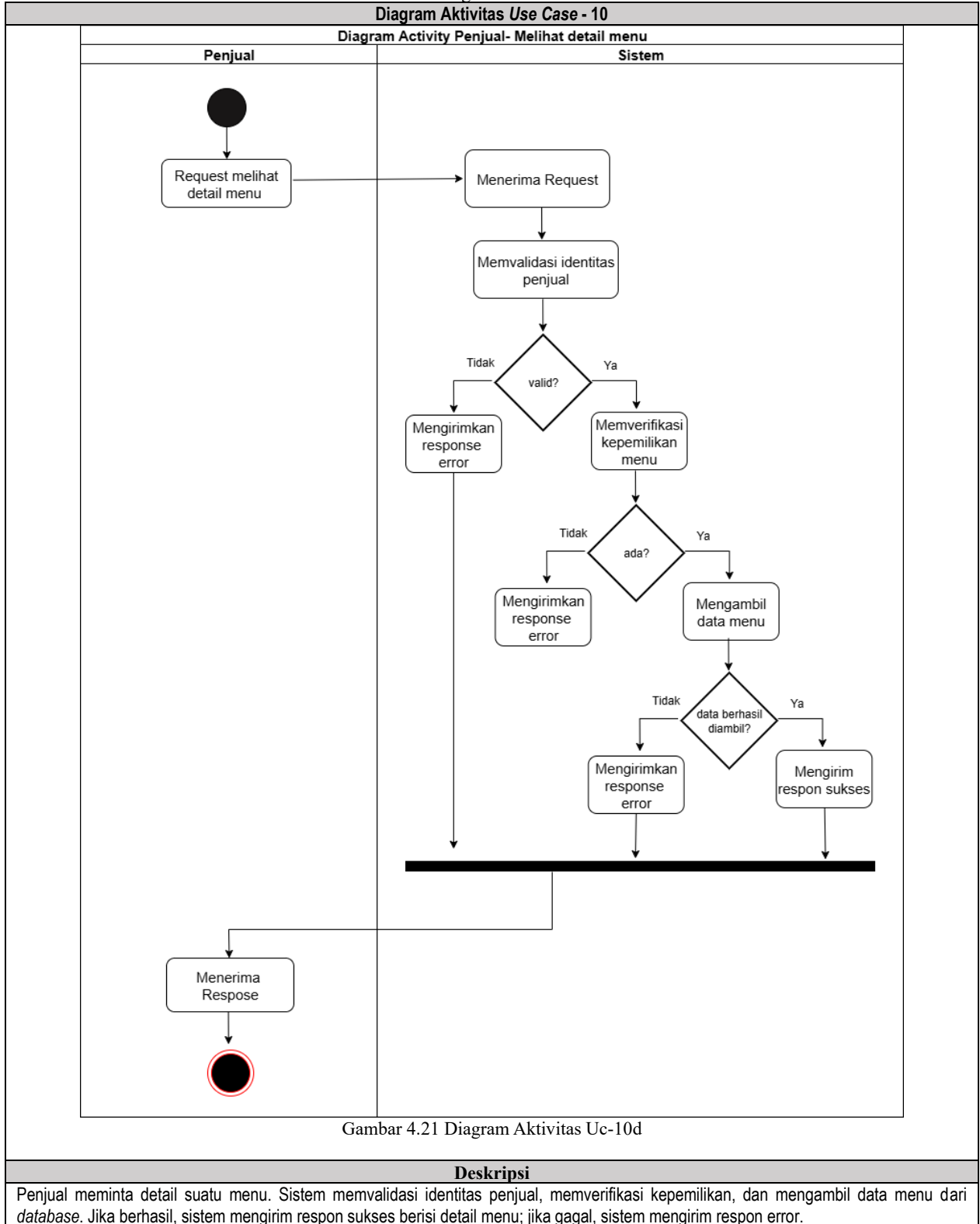


Gambar 4.20 Diagram Aktivitas Uc-10c

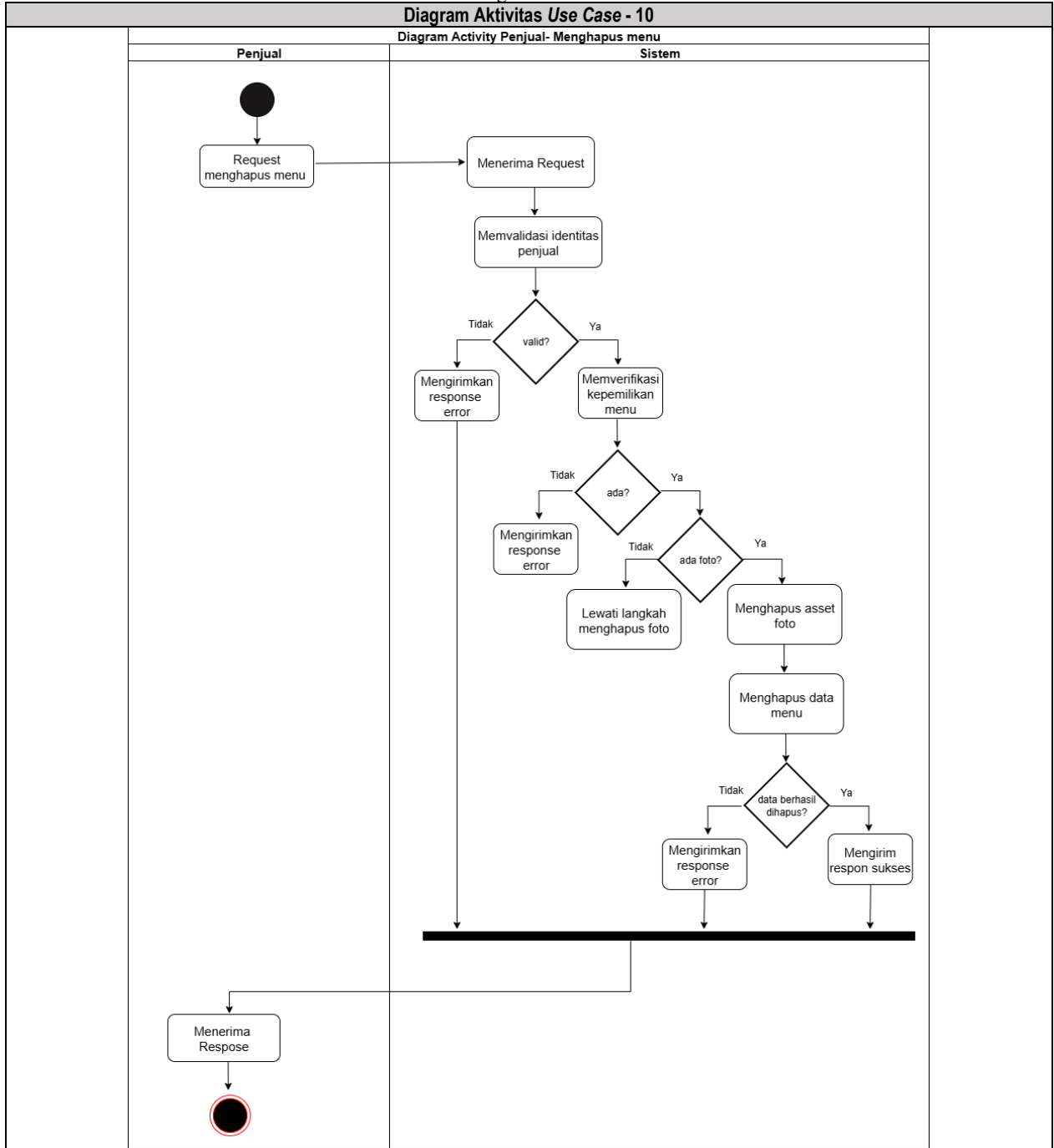
Deskripsi

Penjual melakukan request untuk memperbarui menu. Sistem memvalidasi identitas penjual dan kios, serta memverifikasi kepemilikan menu. Jika ada pembaruan foto, foto lama akan dihapus lalu diganti dengan yang baru. Setelah data menu diperbarui di *database*, sistem mengirim respon sukses; jika gagal, respon error dikirim.

Tabel 4.41 Diagram Aktivitas Uc-10d



Tabel 4.42 Diagram Aktivitas Uc-10e

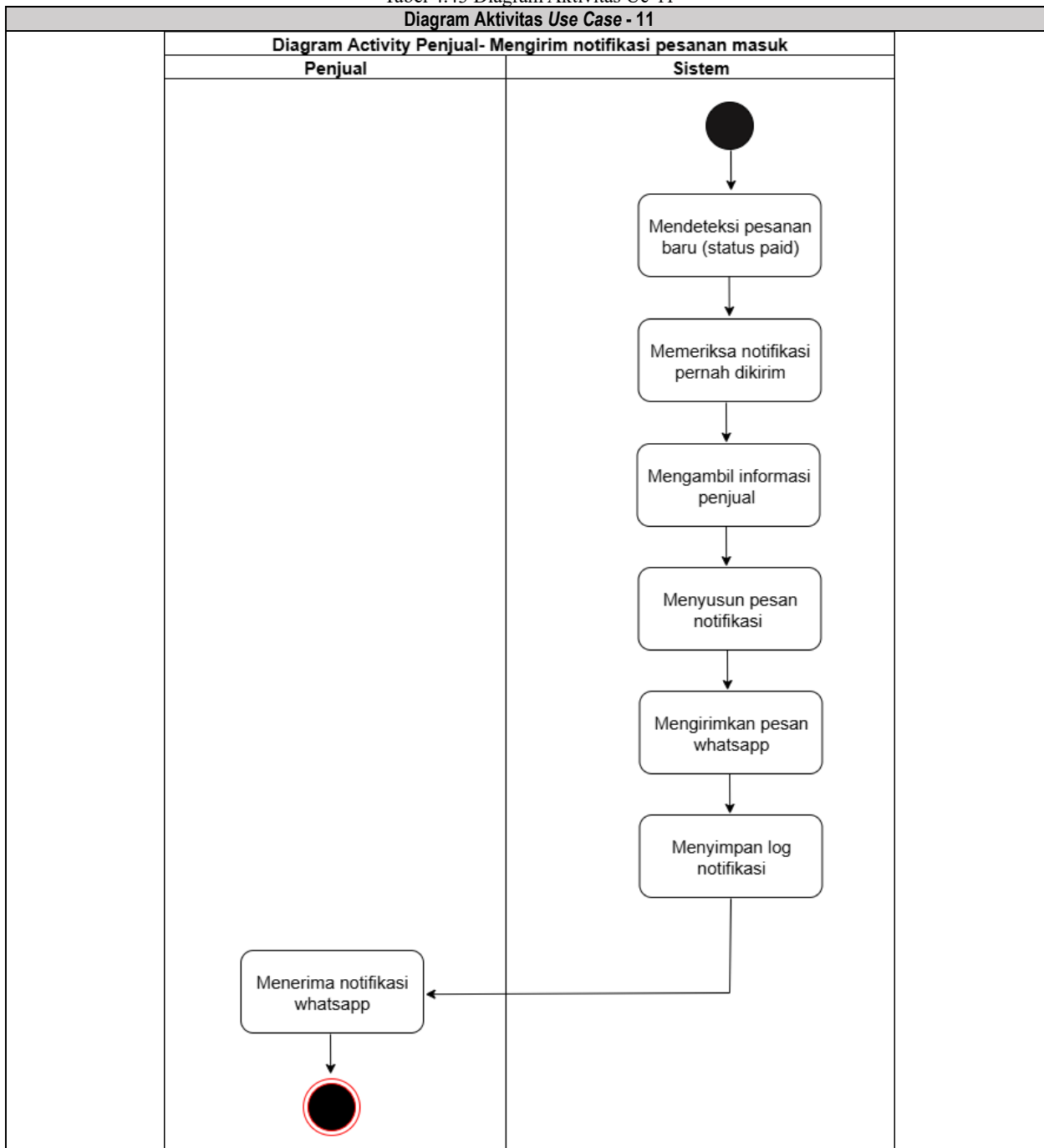


Gambar 4.22 Diagram Aktivitas Uc-10e

Deskripsi

Penjual mengirim request untuk menghapus menu. Sistem memvalidasi identitas penjual dan memverifikasi kepemilikan menu. Jika menu ada dan memiliki foto, sistem menghapus aset foto terlebih dahulu lalu menghapus data menu. Jika data berhasil dihapus, sistem mengirim respon sukses; sebaliknya, jika gagal, sistem mengirim respon error.

Tabel 4.43 Diagram Aktivitas Uc-11

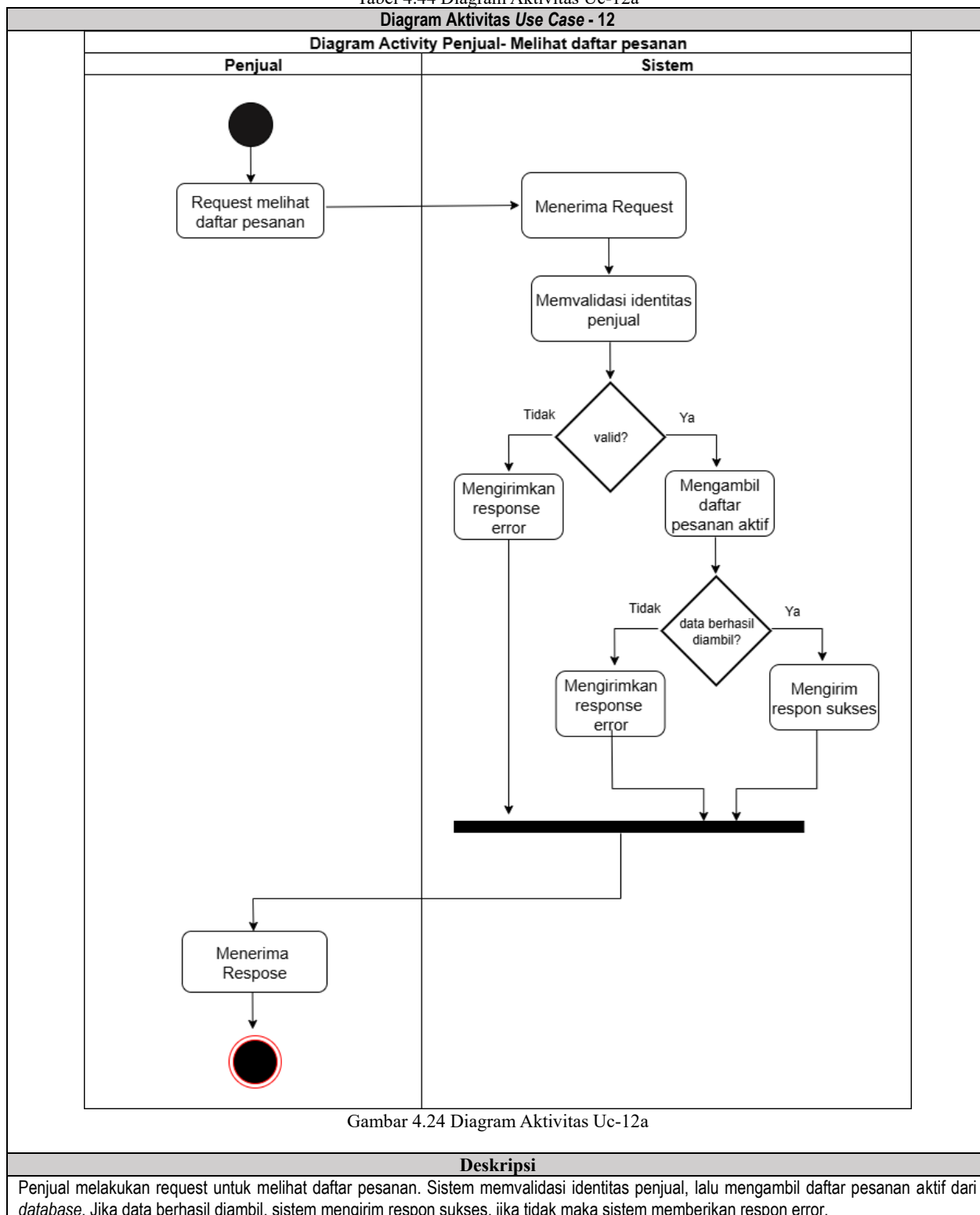


Gambar 4.23 Diagram Aktivitas Uc-11

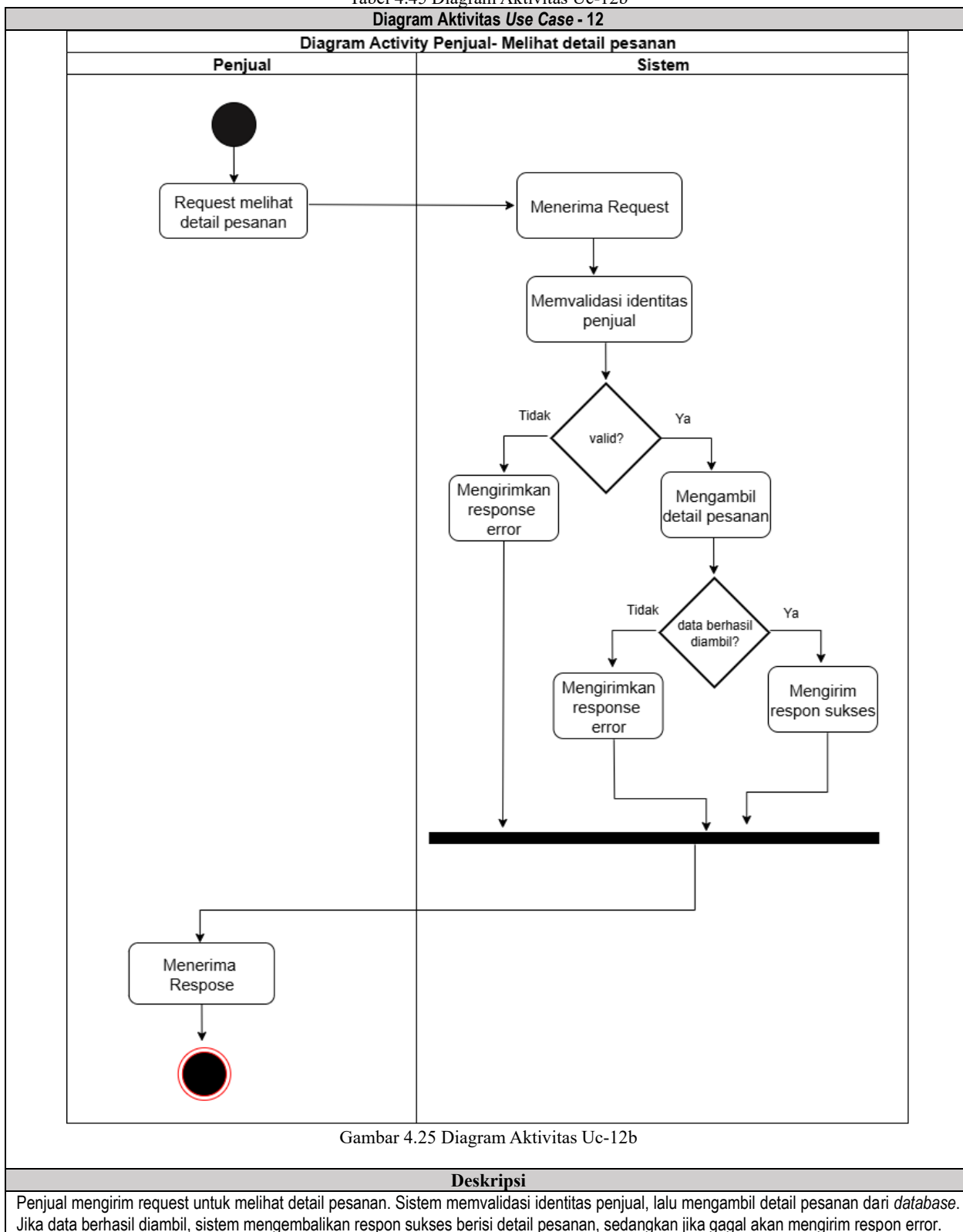
Deskripsi

Sistem mendeteksi pesanan baru dengan status *paid*, memeriksa apakah notifikasi sebelumnya sudah pernah dikirim, lalu mengambil informasi penjual. Setelah itu sistem menyusun pesan notifikasi, mengirimkannya via WhatsApp, dan menyimpan log pengiriman notifikasi. Penjual kemudian menerima notifikasi pesanan baru.

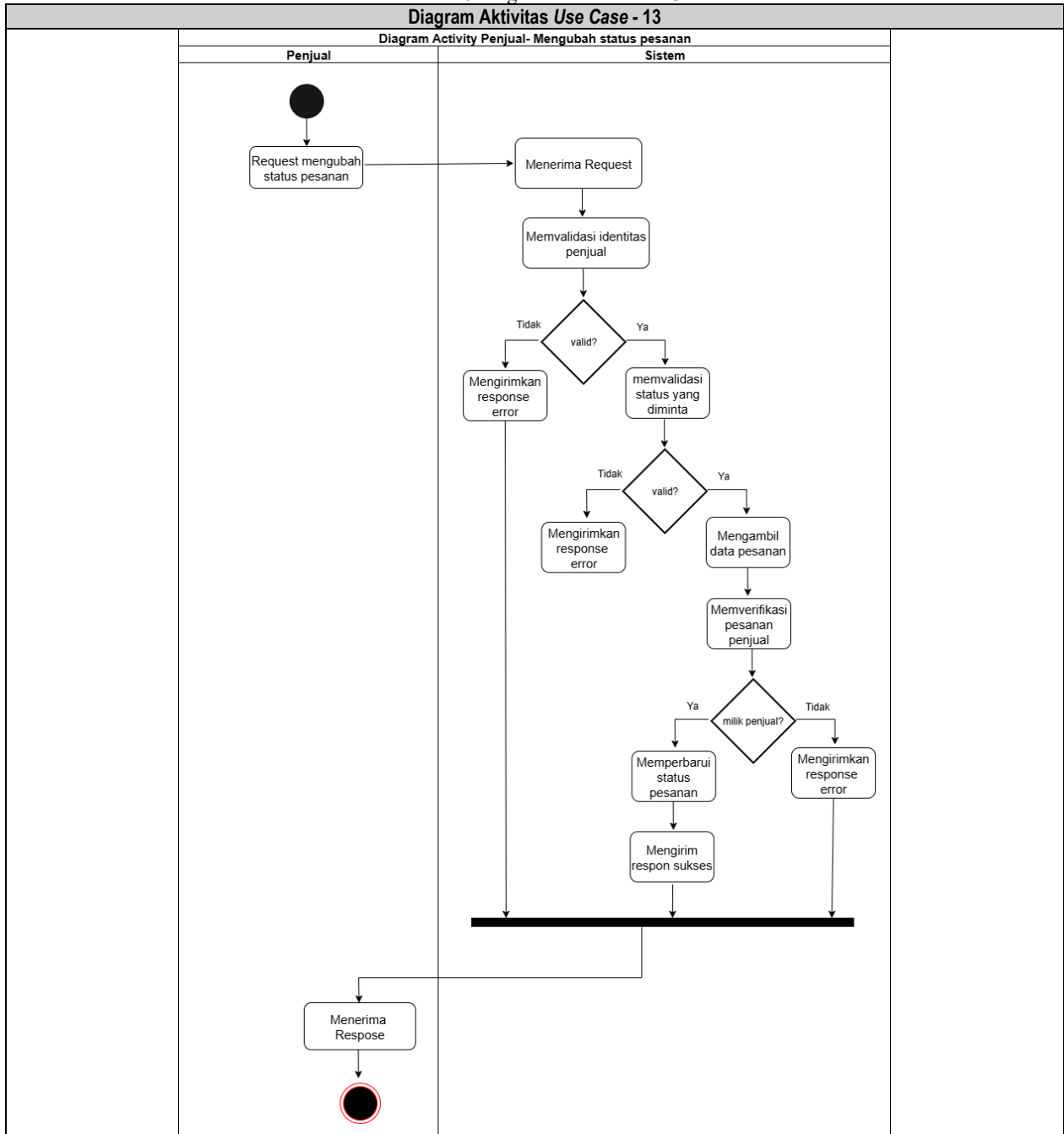
Tabel 4.44 Diagram Aktivitas Uc-12a



Tabel 4.45 Diagram Aktivitas Uc-12b



Tabel 4.46 Diagram Aktivitas Uc-13

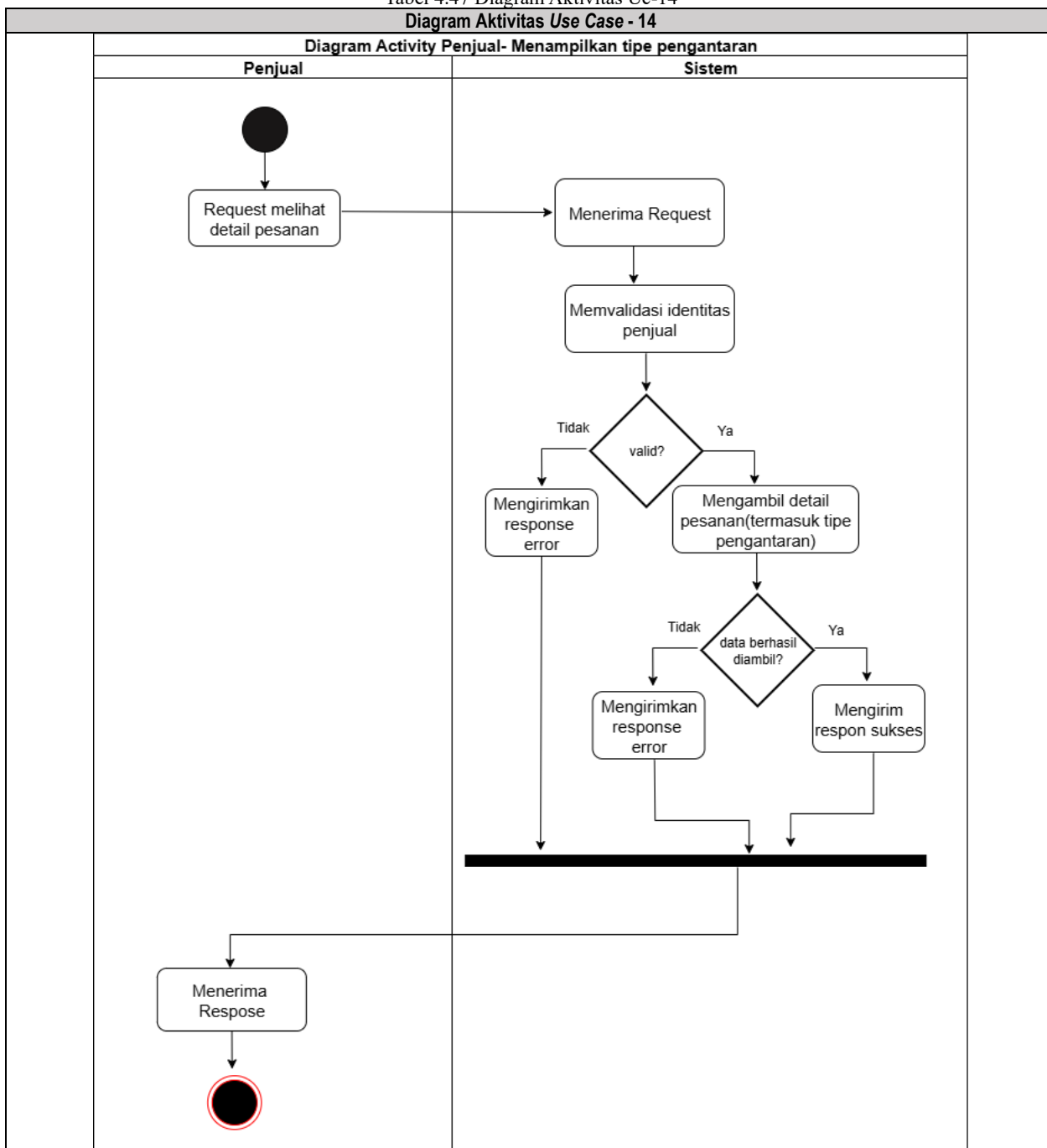


Gambar 4.26 Diagram Aktivitas Uc-13

Deskripsi

Penjual mengajukan request perubahan status pesanan. Sistem memvalidasi identitas penjual, memvalidasi status yang diminta, serta memverifikasi kepemilikan pesanan. Jika valid, sistem memperbarui status pesanan dan mengirim respon sukses; jika tidak, sistem mengirim respon error.

Tabel 4.47 Diagram Aktivitas Uc-14



Gambar 4.27 Diagram Aktivitas Uc-14

Deskripsi

Penjual melakukan request untuk melihat detail pesanan beserta tipe pengantarannya. Sistem memvalidasi identitas penjual, mengambil detail pesanan dari *database*, dan menampilkan informasi termasuk tipe pengantaran. Jika data berhasil diambil, sistem mengembalikan respon sukses, jika gagal maka respon error dikirim.

4.2 Perancangan Kebutuhan Perangkat Lunak

Perancangan kebutuhan perangkat lunak merupakan tahapan yang menjabarkan kebutuhan fungsional maupun non-fungsional sistem yang akan dikembangkan. Pada tahap ini, kebutuhan perangkat

lunak diidentifikasi, dianalisis, dan dirancang agar sesuai dengan tujuan pembangunan aplikasi *E-canteen* PasMeal.

4.2.1 Pemodelan Data

Kebutuhan-kebutuhan perangkat lunak mencakup di dalamnya kebutuhan-kebutuhan untuk membuat, memperluas, atau bersinggungan dengan basis data. Untuk itu, harus didefinisikan semua objek data yang akan diproses di dalam sistem atau perangkat lunak, mendefinisikan relasi antar objek, dan mendefinisikan informasi lain terkait relasi.

4.2.1.1 Objek Data

Objek data merupakan sebuah struktur yang mewakili sekumpulan informasi kompleks agar dapat diinterpretasikan oleh perangkat lunak. Setiap objek data tersusun dari berbagai properti atau atribut yang mendefinisikan karakteristiknya. Selain itu, objek data juga bisa memodelkan sebuah entitas eksternal, yaitu pihak yang menjadi sumber maupun pengguna informasi. Rincian mengenai objek data yang akan dirancang dapat dilihat pada tabel di bawah ini.

Tabel 4.48 Object Data

No	Entity	Fungsi
1	Penjual	Mewakili pemilik akun/kios; menyimpan identitas, kontak, kredensial, dan status verifikasi untuk autentikasi & otorisasi.
2	Kios	Menyimpan profil kios milik penjual (nama, deskripsi, kontak, rekening) sebagai sumber menu & pesanan.
3	Menu	Menyimpan daftar produk/masakan yang dijual pada kios, termasuk harga, ketersediaan, estimasi waktu, dan foto.
4	Keranjang	Menampung item yang dipilih pembeli (per <i>guest_id</i>) dari satu kios, lengkap dengan jumlah, catatan, dan total sementara.
5	Pesanan	Mewakili transaksi pemesanan: identitas pemesan/guest, tipe pengantaran, alamat/tujuan, status & progres, metode pembayaran, <i>token/snap</i> , total & estimasi waktu.
6	Pesanan_Detail	Rincian item per pesanan (menu, jumlah, harga, subtotal) sebagai snapshot saat checkout.
7	OTP	Menyimpan kode verifikasi satu kali untuk proses aktivasi/validasi penjual beserta masa berlaku & status penggunaan.
8	Password_Reset_Tokens	Menyimpan <i>token</i> sementara untuk reset kata sandi penjual beserta masa kedaluwarsa.
9	Blacklisted_Tokens	Menyimpan daftar <i>token</i> JWT yang diblokir (<i>logout/revoke</i>) agar tidak dapat digunakan kembali.
10	WA_Notif_Logs	Mencatat log pengiriman notifikasi WhatsApp terkait pesanan (<i>audit trail</i> /waktu kirim).

4.2.1.2 Atribut Data

Setiap objek data tersusun atas sekumpulan atribut. Atribut-atribut ini merupakan elemen data fundamental yang berfungsi untuk mendeskripsikan setiap properti dan karakteristik spesifik dari objek tersebut. Rincian dari setiap atribut data yang dirancang akan disajikan dalam tabel di bawah ini.

Tabel 4.49 Atribut Data

No	Nama	Atribut	Type	Value	Primary Key	Foreign Key
1	penjual	id	int8	–	✓	
		nama	varchar	255		
		no_hp	varchar	20		

No	Nama	Atribut	Type	Value	Primary Key	Foreign Key
		email	varchar	255		
		password	text			
		is_verified	bool			
		created_at	timestamp			
2	kios	id	int8		✓	
		penjual_id	int8			✓
		nama_kios	varchar	255		
		nama_bank	varchar	100		
		nomor_rekening	varchar	50		
		deskripsi	text			
		gambar_kios	varchar	225		
		created_at	timestamp			
3	menu	id	int8		✓	
		penjual_id	int8			✓
		kios_id	int8			✓
		nama_menu	varchar	255		
		deskripsi	text			
		harga	numeric			
		foto_menu	varchar	255		
		status_tersedia	bool			
		estimasi_minit	int4			
		foto_public_id	varchar	255		
		created_at	timestamp			
4	keranjang	id	int8		✓	
		guest_id	varchar	255		
		menu_id	int8			✓
		jumlah	int4			
		total_harga	int4			
		kios_id	int8			✓
5	pesanan	id	int8		✓	
		nama_pemesan	varchar	255		
		total_harga	int4			
		catatan	varchar	255		
		guest_id	varchar	255		
		tipe_pengantaran	varchar	50		
		no_hp	varchar	20		
		diantar_ke	text			
		created_at	timestamp			
		payment_type	varchar	50		
		payment_details	jsonb			
		status	varchar	50		
		snap_token	text			
		order_id	varchar	50		
		total_estimasi	int4			
		paid_at	timestamp			
		kios_id	int8			✓
		estimasi_mulai_at	timestamp			
		estimasi_selesai_at	timestamp			
		delayed	bool			
		waktu_proses_mulai	Timestamp			
6	pesanan_detail	id	int8		✓	
		pesanan_id	int8			✓
		menu_id	int8			✓
		nama_menu	varchar	255		
		harga	numeric			
		foto_menu	varchar	255		
		jumlah	int4			

No	Nama	Atribut	Type	Value	Primary Key	Foreign Key
		subtotal	numeric			
7	otp	id	int8		✓	
		otp_code	varchar	6		
		expired_at	timestamp			
		is_used	bool			
		penjual_id	int8			✓
		created_at	timestamp			
8	password_reset_tokens	id	int8		✓	
		penjual_id	int8			✓
		token	text			
		expired_at	timestamp			
9	blacklisted_tokens	id	int8		✓	
		token	text			
		expired_at	timestamp			
10	wa_notif_logs	id	int8		✓	
		pesanan_id	int8			✓
		created_at	timestamp			

4.2.2 Pemodelan Berbasis Kelas

Pemodelan berbasis kelas berfokus pada representasi struktur statis sebuah sistem. Pendekatan ini bertujuan untuk menjabarkan beberapa aspek fundamental, yaitu: objek-objek yang akan dimanipulasi oleh perangkat lunak, operasi (metode) yang dapat diterapkan pada objek tersebut beserta dampaknya, relasi yang terjalin antar objek, serta pola kolaborasi antar kelas yang didefinisikan. Adapun proses pemodelan ini dilakukan melalui beberapa tahapan utama. Proses diawali dengan identifikasi kelas-kelas analisis, dilanjutkan dengan pendefinisian atribut dan metode (perilaku) untuk setiap kelas, dan diakhiri dengan visualisasi keseluruhan struktur tersebut dalam sebuah diagram kelas.

4.2.2.1 Mengidentifikasi Kelas-Kelas Analisis

Tahap identifikasi kelas analisis didasarkan pada pemeriksaan mendalam terhadap skenario penggunaan (*use case*) perangkat lunak. Skenario-skenario ini telah dirancang sebelumnya sebagai bagian dari model kebutuhan sistem. Berikut adalah kelas-kelas analisis yang diidentifikasi dari proses tersebut:

Tabel 4.50 Kelas-Kelas Analisis

No	Nama Kelas	Jenis	Deskripsi
1.	Penjual	Entity	Merepresentasikan data penjual (akun, kontak, status verifikasi).
2.	Kios	Entity	Profil kios: nama_kios, deskripsi, gambar, rekening & bank.
3.	Menu	Entity	Data produk/masakan (nama, deskripsi, harga, foto, ketersediaan, estimasi).
4.	Keranjang	Entity	Item sementara yang dipilih pembeli per guest_id (1 kios per keranjang).
5.	Pesanan	Entity	Transaksi pemesanan (identitas pemesan, alamat/tujuan, status, pembayaran, estimasi).
6.	PesananDetail	Entity	Rincian item pesanan (snapshot menu, qty, harga, subtotal).
7.	OTP	Entity	Kode verifikasi satu kali beserta masa berlaku & status pemakaian.
8.	PasswordResetTokens	Entity	Token sementara untuk reset password penjual.
9.	BlacklistedTokens	Entity	Token JWT yang diblokir (logout/revoke).
10.	WANotifLogs	Entity	Log pengiriman notifikasi WhatsApp terkait pesanan.
11.	authRoutes	Boundary	Mendefinisikan endpoint autentikasi (login, register, verify/OTP, reset password).
12.	dashboardRoutes	Boundary	Endpoint ringkasan/monitor (statistik untuk dashboard).
13.	keranjangRoutes	Boundary	Endpoint HTTP operasi keranjang(add/update/remove/list)
14.	kiosRoutes	Boundary	Endpoint HTTP pengelolaan kios (CRUD profil kios).
15.	menuRoutes	Boundary	Endpoint HTTP pengelolaan menu & pencarian menu per kios.
16.	midtransRoutes	Boundary	Endpoint pembayaran (buat transaksi, callback/notifikasi).
17.	penjualRoutes	Boundary	Endpoint pengelolaan data penjual (profil, verifikasi).
18.	pesananRoutes	Boundary	Endpoint pemesanan (checkout, status, estimasi, konfirmasi).
19.	searchRoutes	Boundary	Endpoint pencarian global (menu/kios/kata kunci).
20.	authController	Controller	Mengelola alur permintaan autentikasi
21.	dashboardController	Controller	Logika agregasi data untuk kebutuhan dashboard.
22.	keranjangController	Controller	Logika bisnis item keranjang & validasi batas 1 kios/keranjang.
23.	kiosController	Controller	Logika bisnis pengelolaan kios (validasi penjual, rekening, gambar).
24.	menuController	Controller	Logika bisnis CRUD menu, status ketersediaan, estimasi, unggah foto.
25.	midtransController	Controller	Orkestrasi pembayaran (buat transaksi, snap token, verifikasi callback).
26.	penjualController	Controller	Pengelolaan profil penjual.
27.	pesananController	Controller	Pembuatan pesanan dari keranjang, perhitungan total/estimasi, pembaruan status.
28.	searchController	Controller	Implementasi pencarian & filter berdasarkan kata kunci.
29.	whatsappService	Controller	Integrasi WA (Baileys) untuk kirim notifikasi pesanan.
30.	getGuestId	Utility	Generator/normalizer guest_id untuk pengguna tamu.
31.	cloudinaryUtil	Utility	Helper unggah & hapus gambar di Cloudinary.
32.	authMiddleware	Middleware	Proteksi rute dengan JWT, ekstraksi req.user, pengecekan role.
33.	uploadMiddleware	Middleware	Upload/validasi file (foto menu/kios) menggunakan Multer.

4.2.2.2 Mengidentifikasi Atribut-Atribut

Setiap kelas yang telah diidentifikasi dalam pemodelan sistem memiliki serangkaian atribut. Atribut-atribut ini berfungsi untuk mendeskripsikan secara detail setiap properti atau karakteristik dari masing-masing kelas tersebut. Tabel berikut akan menyajikan daftar lengkap atribut untuk setiap kelas yang telah didefinisikan.

Tabel 4.51 Atribut-Atribut

No	Nama Kelas	Nama Atribut	Deskripsi
1.	Penjual	id	Atribut ini digunakan sebagai <i>primary key</i> unik untuk setiap penjual.
		nama	Atribut ini digunakan untuk menyimpan nama lengkap penjual.
		no_hp	Atribut ini digunakan untuk menyimpan nomor WhatsApp penjual yang digunakan untuk OTP dan notifikasi.
		email	Atribut ini digunakan untuk menyimpan alamat email penjual yang unik.
		password	Atribut ini digunakan untuk menyimpan <i>hash</i> dari password penjual.
		is_verified	Atribut ini digunakan untuk menandai apakah akun penjual sudah diverifikasi melalui OTP (<i>true/false</i>).

No	Nama Kelas	Nama Atribut	Deskripsi
2..	Kios	created_at	Atribut ini digunakan untuk menyimpan waktu ketika akun penjual dibuat.
		id	Atribut ini digunakan sebagai <i>primary key</i> unik untuk setiap kios.
		penjual_id	Atribut ini digunakan sebagai <i>foreign key</i> yang mereferensikan penjual pemilik kios.
		nama_kios	Atribut ini digunakan untuk menyimpan nama kios yang unik.
		nama_bank	Atribut ini digunakan untuk menyimpan nama bank yang digunakan pemilik kios.
		nomor_rekening	Atribut ini digunakan untuk menyimpan nomor rekening bank pemilik kios.
		deskripsi	Atribut ini digunakan untuk menyimpan deskripsi singkat mengenai kios.
		gambar_kios	Atribut ini digunakan untuk menyimpan URL atau path gambar profil kios.
		created_at	Atribut ini digunakan untuk menyimpan waktu saat kios dibuat.
3.	menu	id	Atribut ini digunakan sebagai <i>primary key</i> unik untuk setiap menu.
		penjual_id	Atribut ini digunakan sebagai <i>foreign key</i> yang mereferensikan penjual pemilik menu.
		kios_id	Atribut ini digunakan sebagai <i>foreign key</i> yang mereferensikan kios tempat menu dijual.
		nama_menu	Atribut ini digunakan untuk menyimpan nama menu.
		deskripsi	Atribut ini digunakan untuk menyimpan deskripsi detail mengenai menu.
		harga	Atribut ini digunakan untuk menyimpan harga menu.
		foto_menu	Atribut ini digunakan untuk menyimpan URL atau path gambar menu.
		status_tersedia	Atribut ini digunakan untuk menyimpan status ketersediaan menu (<i>tersedia</i> atau <i>tidak tersedia</i>).
		estimasi_minit	Atribut ini digunakan untuk menyimpan estimasi waktu penyajian menu dalam menit.
		foto_public_id	Atribut ini digunakan untuk menyimpan <i>public_id</i> dari Cloudinary sebagai referensi pengelolaan gambar.
		created_at	Atribut ini digunakan untuk menyimpan waktu saat menu ditambahkan.
4.	keranjang	id	Atribut ini digunakan sebagai <i>primary key</i> unik untuk setiap item keranjang.
		guest_id	Atribut ini digunakan untuk menyimpan identitas sementara pembeli (guest) yang memiliki keranjang.
		menu_id	Atribut ini digunakan sebagai <i>foreign key</i> untuk mereferensikan menu yang ditambahkan ke keranjang.
		jumlah	Atribut ini digunakan untuk menyimpan kuantitas (jumlah) menu yang dipesan.
		total_harga	Atribut ini digunakan untuk menyimpan total harga ($\text{harga} \times \text{jumlah}$) dari item dalam keranjang.
		kios_id	Atribut ini digunakan sebagai <i>foreign key</i> untuk menyimpan identitas kios tempat menu berasal, agar keranjang hanya berisi dari satu kios.
		created_at	Atribut ini digunakan sebagai <i>primary key</i> unik untuk setiap pesanan.
5.	pesanan	nama_pemesan	Atribut ini digunakan untuk menyimpan nama pembeli yang melakukan pesanan.
		total_harga	Atribut ini digunakan untuk menyimpan total harga keseluruhan pesanan.
		catatan	Atribut ini digunakan untuk menyimpan catatan tambahan dari pembeli terkait pesanan.
		guest_id	Atribut ini digunakan untuk menyimpan identitas sementara pembeli (guest).
		tipe_pengantaran	Atribut ini digunakan untuk menyimpan metode pengantaran (ambil sendiri / diantar).
		no_hp	Atribut ini digunakan untuk menyimpan nomor WhatsApp/telepon pembeli.
		diantar_ke	Atribut ini digunakan untuk menyimpan alamat tujuan apabila pesanan diantar.
		created_at	Atribut ini digunakan untuk mencatat waktu pesanan dibuat.
		payment_type	Atribut ini digunakan untuk menyimpan jenis metode pembayaran (misal: QRIS, transfer bank).
		payment_details	Atribut ini digunakan untuk menyimpan detail teknis pembayaran (misal: nomor VA, referensi Midtrans).
		status	Atribut ini digunakan untuk menyimpan status pesanan (pending, paid, processing, ready, delivering, done).
		snap_token	Atribut ini digunakan untuk menyimpan <i>token</i> pembayaran Midtrans Snap.
		order_id	Atribut ini digunakan untuk menyimpan ID unik transaksi/order dari Midtrans.
		total_estimasi	Atribut ini digunakan untuk menyimpan estimasi waktu penyelesaian pesanan (dalam menit).
		paid_at	Atribut ini digunakan untuk menyimpan waktu pembayaran berhasil dilakukan.
		kios_id	Atribut ini digunakan sebagai <i>foreign key</i> untuk mereferensikan kios yang menerima pesanan.
		estimasi_mulai_at	Atribut ini digunakan untuk menyimpan waktu estimasi mulai diproses.
		estimasi_selesai_at	Atribut ini digunakan untuk menyimpan estimasi waktu pesanan selesai.
		delayed	Atribut ini digunakan untuk menandai apakah pesanan mengalami keterlambatan.
		waktu_proses_mula	Atribut ini digunakan untuk mencatat waktu aktual ketika pesanan mulai diproses.
6.	pesanan_detail	id	Atribut ini digunakan sebagai <i>primary key</i> unik untuk setiap detail item pesanan.

No	Nama Kelas	Nama Atribut	Deskripsi
		pesanan_id	Atribut ini digunakan sebagai <i>foreign key</i> yang menghubungkan detail dengan tabel pesanan.
		menu_id	Atribut ini digunakan sebagai <i>foreign key</i> untuk mereferensikan menu yang dipesan.
		nama_menu	Atribut ini digunakan untuk menyimpan nama menu saat pesanan dibuat (snapshot).
		harga	Atribut ini digunakan untuk menyimpan harga menu saat transaksi.
		foto_menu	Atribut ini digunakan untuk menyimpan foto/menu image pada saat pesanan dibuat.
		jumlah	Atribut ini digunakan untuk menyimpan jumlah item menu yang dipesan.
		subtotal	Atribut ini digunakan untuk menyimpan hasil perhitungan harga × jumlah item.
7.	otp	id	Atribut ini digunakan sebagai <i>primary key</i> unik untuk setiap kode OTP.
		otp_code	Atribut ini digunakan untuk menyimpan kode OTP yang dikirim ke penjual.
		expired_at	Atribut ini digunakan untuk menyimpan batas waktu kedaluwarsa OTP.
		is_used	Atribut ini digunakan untuk menandai apakah OTP sudah digunakan atau belum.
		penjual_id	Atribut ini digunakan sebagai <i>foreign key</i> untuk menghubungkan OTP dengan penjual terkait.
		created_at	Atribut ini digunakan untuk mencatat waktu OTP dibuat.
8.	password_reset_tokens	id	Atribut ini digunakan sebagai <i>primary key</i> unik untuk setiap <i>token</i> reset password.
		penjual_id	Atribut ini digunakan sebagai <i>foreign key</i> untuk menghubungkan <i>token</i> dengan penjual terkait.
		token	Atribut ini digunakan untuk menyimpan <i>token</i> unik yang dipakai untuk proses reset password.
		expired_at	Atribut ini digunakan untuk menyimpan waktu kedaluwarsa <i>token</i> (valid hanya untuk periode tertentu).
9.	blacklisted_tokens	id	Atribut ini digunakan sebagai <i>primary key</i> unik untuk setiap entri <i>token</i> yang diblokir.
		token	Atribut ini digunakan untuk menyimpan JWT yang sudah di- <i>blacklist</i> saat logout.
		expired_at	Atribut ini digunakan untuk menyimpan waktu kedaluwarsa <i>token</i> yang diblokir, agar bisa dihapus otomatis setelah tidak valid.
10.	wa_notif_logs	id	Atribut ini digunakan sebagai <i>primary key</i> unik untuk setiap log notifikasi WhatsApp
		pesanan_id	Atribut ini digunakan sebagai <i>foreign key</i> untuk menghubungkan log dengan pesanan terkait.
		created_at	Atribut ini digunakan untuk menyimpan waktu kapan notifikasi WA berhasil tercatat/dikirim.
11.	authRoutes	POST /register	Endpoint untuk mendaftarkan akun penjual baru (publik).
		POST /login	Endpoint untuk login akun penjual (publik).
		POST /verify-otp	Endpoint untuk verifikasi OTP penjual (publik).
		POST /resend-otp	Endpoint untuk mengirim ulang OTP ke WhatsApp (publik).
		POST /forgot-password	Endpoint untuk mengajukan reset password (publik).
		POST /reset-password	Endpoint untuk mereset password menggunakan <i>token</i> (publik).
		POST /logout	Endpoint untuk logout & blacklist <i>token</i> (butuh <i>authMiddleware</i>).
12.	dashboardRoutes	GET /	Endpoint untuk mengambil data ringkasan dashboard. Butuh <i>verifiedMiddleware</i> .
13.	keranjangRoutes	POST /keranjang	Endpoint untuk menambah item ke keranjang.
		GET /keranjang	Endpoint untuk mengambil seluruh isi keranjang.
		PUT /keranjang/:id	Endpoint untuk memperbarui item dalam keranjang berdasarkan ID item.
		DELETE /keranjang/:id	Endpoint untuk menghapus item dari keranjang berdasarkan ID item.
14.	kiosRoutes	GET /profil	Endpoint untuk mengambil profil kios penjual. Butuh <i>authMiddleware</i> .
		PUT /:id	Endpoint untuk memperbarui data kios (dengan upload gambar_kios). Butuh <i>authMiddleware</i> .
		POST /	Endpoint untuk membuat kios baru penjual. Butuh <i>authMiddleware</i> .
		GET /homepage	Endpoint untuk menampilkan daftar kios di beranda (publik).
		GET /search	Endpoint untuk pencarian kios berdasarkan kata kunci (publik).
		GET /:id/menus/search	Endpoint untuk mencari menu berdasarkan kata kunci dalam kios tertentu (publik).
		GET /:id/menus	Endpoint untuk menampilkan daftar menu dari kios tertentu (publik).
		GET /:id	Endpoint untuk menampilkan detail kios berdasarkan ID (publik).
		GET /	Endpoint untuk menampilkan semua kios (publik).
15.	menuRoutes	GET /new	Endpoint untuk mengambil 5 menu terbaru (publik/pembeli).
		GET /search	Endpoint untuk mencari menu (publik/pembeli).
		GET /kios/:id/search	Endpoint untuk mencari menu pada kios tertentu (publik/pembeli).

No	Nama Kelas	Nama Atribut	Deskripsi
		GET /	Endpoint untuk mengambil semua menu milik penjual. Butuh <i>authMiddleware</i> .
		GET /paginated	Endpoint untuk mengambil menu penjual dengan pagination. Butuh <i>authMiddleware</i> .
		POST /	Endpoint untuk menambah menu baru (dengan upload foto_menu). Butuh <i>verifiedMiddleware</i> .
		PUT /:id	Endpoint untuk memperbarui menu berdasarkan ID (dengan upload foto_menu). Butuh <i>verifiedMiddleware</i> .
		DELETE /:id	Endpoint untuk menghapus menu berdasarkan ID. Butuh <i>verifiedMiddleware</i> .
		GET /buyer/:id	Endpoint untuk menampilkan detail menu (khusus pembeli, publik).
		GET /seller/:id	Endpoint untuk menampilkan detail menu milik penjual. Butuh <i>authMiddleware</i> .
16.	midtransRoutes	POST /create-transaction	Endpoint untuk membuat transaksi pembayaran baru melalui Midtrans.
		POST /notification	Endpoint callback/notification dari Midtrans untuk update status transaksi.
17.	penjualRoutes	GET /profil	Endpoint untuk mengambil data profil penjual yang sedang login. Butuh <i>authMiddleware</i> .
		PUT /profil	Endpoint untuk memperbarui data profil penjual yang sedang login. Butuh <i>authMiddleware</i> .
18.	pesananRoutes	POST /pesanan	Endpoint untuk membuat pesanan dari keranjang (pembeli).
		GET /pesanan	Endpoint untuk mengambil daftar pesanan berdasarkan <i>guest_id</i> (pembeli).
		GET /pesanan/riwayat	Endpoint untuk mengambil riwayat pesanan penjual. Butuh <i>verifiedMiddleware</i> .
		GET /pesanan/riwayat/:id	Endpoint untuk mengambil detail riwayat pesanan penjual. Butuh <i>verifiedMiddleware</i> .
		GET /pesanan-masuk	Endpoint untuk mengambil daftar pesanan masuk (penjual). Butuh <i>verifiedMiddleware</i> .
		GET /pesanan-masuk/count	Endpoint untuk menghitung jumlah pesanan masuk (penjual) untuk badge. Butuh <i>verifiedMiddleware</i> .
		GET /pesanan-masuk/:id	Endpoint untuk mengambil detail pesanan masuk (penjual). Butuh <i>verifiedMiddleware</i> .
		PATCH /pesanan/:id/status	Endpoint untuk memperbarui status pesanan (penjual). Butuh <i>verifiedMiddleware</i> .
		GET /pesanan/:id/status	Endpoint untuk melihat status pesanan (pembeli).
		GET /pesanan/:id	Endpoint untuk mengambil detail pesanan berdasarkan ID (pembeli).
19.	searchRoutes	GET /	Endpoint untuk melakukan pencarian gabungan pada kios dan menu (publik).
20.	authController	register	Method untuk menangani permintaan pendaftaran penjual baru (validasi input, hashing password, insert DB, keluarkan JWT status belum terverifikasi).
		verifyOtp	Method untuk verifikasi OTP: cek OTP aktif & belum digunakan, update <i>penjual.is_verified</i> = true, tandai <i>OTP is_used</i> = true, keluarkan JWT terverifikasi.
		resendOtp	Method untuk mengirim ulang OTP bila akun belum terverifikasi & OTP lama sudah kedaluwarsa; generate OTP baru (± 3 menit) dan kirim via WhatsApp.
		login	Method untuk login penjual: cek status verifikasi, verifikasi password (bcrypt), keluarkan JWT (durasi 1h / 7d sesuai <i>rememberMe</i>).
		forgotPassword	Method untuk permintaan reset password: generate <i>token</i> (15 menit), simpan ke <i>password_reset_tokens</i> , kirim link reset via WhatsApp.
		resetPassword	Method untuk reset password menggunakan <i>token</i> yang masih berlaku: hash password baru, update <i>penjual.password</i> , hapus <i>token reset</i> .
		logout	Method untuk logout: ambil Bearer <i>token</i> dari header, decode exp, simpan ke tabel <i>blacklisted_tokens</i> sebagai <i>token</i> yang diblokir.
21.	dashboardController	getDashboardData	Method untuk mengambil data ringkasan dashboard penjual: total pesanan aktif, total menu, dan total pendapatan (pesanan selesai). Query menyesuaikan berdasarkan <i>kios_id</i> atau <i>penjual_id</i> dari <i>req.user</i> .
		getStatusLabel	Utility method untuk memetakan status pesanan dari DB ke label yang ditampilkan (berbeda antara tipe pengantaran <i>ambil_sendiri</i> atau <i>diantar</i>)
22.	keranjangController	addToKeranjang	Method untuk menambah item ke keranjang. Validasi <i>guest_id</i> & <i>menu_id</i> , cek menu, pastikan hanya 1 kios per keranjang, update jumlah jika item sudah ada, atau insert baru jika belum ada.
		getKeranjang	Method untuk mengambil isi keranjang berdasarkan <i>guest_id</i> . Mengembalikan daftar item, total harga, dan <i>kios_id</i> jika ada.
		updateKeranjangItem	Method untuk memperbarui jumlah item di keranjang berdasarkan <i>id</i> & <i>guest_id</i> .

No	Nama Kelas	Nama Atribut	Deskripsi
		removeFromKeranjang	Method untuk menghapus item dari keranjang berdasarkan id & guest_id.
23.	kiosController	createKios	Method untuk registrasi kios baru oleh penjual. Validasi penjual, cek duplikasi, insert kios, generate OTP, kirim ke WhatsApp.
		getKiosHomepage	Method untuk menampilkan 8 kios terbaru di homepage pembeli.
		searchKios	Method untuk pencarian kios berdasarkan nama (LIKE query).
		getAllKios	Method untuk mengambil semua kios (pembeli).
		getMenusByKios	Method untuk mengambil semua menu dari kios tertentu berdasarkan kios_id.
		getKiosByPenjual	Method untuk mengambil profil kios milik penjual yang sedang login (via req.user.id).
		updateKios	Method untuk memperbarui profil kios: update nama, deskripsi, rekening, bank, serta upload gambar baru ke Cloudinary.
		getKiosDetail	Method untuk mengambil detail kios berdasarkan kios_id (join dengan data penjual).
24.	menuController	getAllMenu	Method untuk mengambil semua menu milik penjual yang login berdasarkan penjual_id,urut terbaru.
		addMenu	Method untuk menambahkan menu baru (nama, deskripsi, harga, estimasi, status). Mendukung upload foto ke Cloudinary; simpan foto_menu & foto_public_id.
		updateMenu	Method untuk memperbarui menu berdasarkan id & penjual_id. Jika ada file baru: upload ke Cloudinary, hapus file lama (destroy foto_public_id). Partial update untuk field lain.
		getMenuById	Method untuk mengambil detail menu milik penjual berdasarkan id (proteksi penjual_id).
		deleteMenu	Method untuk menghapus menu (cek kepemilikan; jika ada foto_public_id dihapus dari Cloudinary).
		getMenusPaginated	Method untuk mengambil menu dengan pagination (page, limit, offset) khusus penjual. Balikkan total & data.
		getNewMenus	Method untuk mengambil 5 menu terbaru (pembeli).
		searchMenus	Method untuk mencari menu secara global berdasarkan query (pembeli).
		searchMenusByKios	Method untuk mencari menu pada kios tertentu berdasarkan query dan kios_id (pembeli).
		getMenuByIdForBuyer	Method untuk mengambil detail menu untuk pembeli berdasarkan id.
25.	midtransController	createTransaction	Method untuk membuat transaksi Midtrans Snap: validasi input (guest_id, total_harga), buat pesanan baru jika belum ada, generate order_id, hitung gross_amount, panggil API Snap, simpan order_id & snap_token ke tabel pesanan, kembalikan token & redirect URL ke client.
		handleNotification	Method untuk menangani notifikasi dari Midtrans: baca transaction_status, update status pesanan (paid, pending, failed), simpan detail pembayaran (payment_type, payment_details, paid_at). Jika status paid, panggil notifyPenjual untuk mengirim notifikasi WhatsApp ke kios terkait.
26.	penjualController	getProfile	Method untuk mengambil profil penjual berdasarkan req.user.id. Mengembalikan id, nama, no_hp, email. Jika tidak ada ID → 401, jika profil tidak ditemukan → 404.
		updateProfil	Method untuk memperbarui profil penjual (nama, no_hp, email) berdasarkan req.user.id. Gunakan data lama jika field baru tidak dikirim. Jika profil tidak ditemukan → 404.
27.	pesananController	buatPesanan	Method untuk membuat pesanan dari keranjang berdasarkan guest_id: validasi input, hitung total, insert ke pesanan & pesanan_detail, kosongkan keranjang, kirim notifikasi WA ke penjual.
		getPesananByGuest	Method untuk mengambil daftar pesanan pembeli berdasarkan guest_id (dengan pagination).
		getDetailPesanan	Method untuk mengambil detail pesanan pembeli berdasarkan id & guest_id, sertakan item pesanan & antrean estimasi.
		getStatusPesananGuest	Method untuk mengecek status & estimasi selesai pesanan pembeli berdasarkan antrean aktif kios.
		getPesananMasuk	Method untuk mengambil daftar pesanan aktif (masuk) bagi penjual: filter berdasarkan penjual_id, tampilkan nomor antrean, status, estimasi, label status.
		getDetailPesananMasuk	Method untuk mengambil detail satu pesanan aktif penjual, sertakan jadwal estimasi & item menu.
		countPesananMasuk	Method untuk menghitung jumlah pesanan masuk aktif bagi penjual (untuk badge notifikasi).

No	Nama Kelas	Nama Atribut	Deskripsi
		updateStatusPesanan	Method untuk memperbarui status pesanan oleh penjual (pending → paid → processing → ready/delivering → done). Saat status done, otomatis kirim notifikasi WA ke pembeli.
		getRiwayatPesanan	Method untuk mengambil daftar riwayat pesanan selesai (done) bagi penjual (dengan pagination).
		getDetailRiwayatPesanan	Method untuk mengambil detail pesanan riwayat penjual: status, metode bayar, detail menu, estimasi & foto dari Cloudinary.
		notifyPenjual	Utility untuk mengirim WA notifikasi pesanan baru ke penjual (sekali saja, simpan log di wa_notif_logs).
		notifyPembeliPesananSelesai	Utility untuk mengirim WA notifikasi ke pembeli ketika pesanan sudah selesai.
		getStatusLabel	Utility mapping status DB ke label status UI, berbeda untuk ambil sendiri dan diantar.
28.	searchController	searchAll	Method untuk mencari kios & menu berdasarkan query string. Query wajib diisi. Mengembalikan hasil pencarian dari tabel kios (berdasarkan nama_kios) dan tabel menu (berdasarkan nama_menu, join dengan kios). Response berupa { kios, menus }.
29.	whatsappService	connectToWhatsApp	Method untuk membangun koneksi WhatsApp menggunakan Baileys. Menggunakan useMultiFileAuthState untuk autentikasi multi-file, auto-reconnect jika terputus, menampilkan QR di terminal untuk login, dan update status waReady
		sendWaMessage	Method untuk mengirim pesan teks ke nomor WhatsApp. Format nomor otomatis diganti 0 → 62. Mengembalikan true/false sesuai status pengiriman.
		sendOtpMessage	Method untuk mengirim pesan OTP ke nomor WhatsApp: format pesan berisi kode OTP & instruksi. Memanggil sendWaMessage.
		getWASocket	Method untuk mengambil instance socket WA global (globalSock) yang sedang aktif.
30.	getGuestId	getGuestId	Fungsi helper untuk mengambil guest_id dari beberapa sumber request: req.body.guest_id, req.query.guest_id, atau header x-buyer-id. Jika tidak ditemukan, mengembalikan null.
31.	cloudinaryUtil	cloudinary	Utility untuk integrasi dengan Cloudinary. Konfigurasi diambil dari environment variable: CLOUD_NAME, CLOUD_KEY, CLOUD_SECRET. Digunakan untuk upload, update, dan hapus file gambar.
32.	authMiddleware	authMiddleware	Middleware untuk validasi JWT. Mengecek token Bearer, memastikan tidak masuk blacklist, decode payload, dan attach req.user dengan data penjual (id, kios_id). Return 401/403 bila tidak valid.
		verifiedMiddleware	Middleware khusus untuk user yang sudah diverifikasi (is_verified=true). Mengecek JWT, blacklist, dan status verifikasi sebelum melanjutkan ke endpoint. Return 403 bila akun belum diverifikasi.
33.	uploadMiddleware	upload	Middleware untuk unggah file menggunakan Multer. Konfigurasi: simpan di folder /uploads, nama file unik (timestamp-random.ext), filter hanya gambar (jpeg, jpg, png, webp), dan limit ukuran maksimal 2MB.

4.2.2.3 Mengidentifikasi Operasi-Operasi

Operasi-operasi pada dasarnya mendefinisikan perilaku (behavior) suatu objek. Tabel berikut akan menguraikan secara rinci operasi-operasi yang dimiliki oleh setiap kelas yang telah diidentifikasi sebelumnya.

Tabel 4.52 Operasi-Operasi

No	Nama Kelas	Kandidat Perilaku	Nama Perilaku
1.	Penjual	Membuat data penjual baru	Penjual.create(data)
		Mencari penjual berdasarkan ID	Penjual.findByPk(id)
		Mencari satu penjual berdasarkan kondisi tertentu	Penjual.findOne(options)
		Menyimpan perubahan data penjual	penjual.save()
2.	Kios	Membuat kios baru	Kios.create(data)
		Mencari kios berdasarkan penjual_id	Kios.findOne({where: {penjual_id}})
		Mengambil semua kios	Kios.findAll()
		Update data kios	kios.save()
3.	Menu	Membuat menu baru	Menu.create(data)
		Mencari menu berdasarkan kios_id	Menu.findAll({where: {kios_id}})

No	Nama Kelas	Kandidat Perilaku	Nama Perilaku
4.	Keranjang	Update data menu	menu.save()
		Menghapus menu	Menu.destroy({where: {id}})
		Menambahkan item ke keranjang	Keranjang.create(data)
		Menampilkan isi keranjang berdasarkan guest_id	Keranjang.findAll({where: {guest_id}})
		Update jumlah item	Keranjang.update(values, { where: { id } })
		Menghapus item dari keranjang	Keranjang.destroy({where: {id}})
5.	Pesanan	Membuat pesanan baru	Pesanan.create(data)
		Mencari pesanan berdasarkan guest_id	Pesanan.findAll({where: {guest_id}})
		Update status pesanan	pesanan.save()
		Hapus pesanan	Pesanan.destroy({where: {id}})
6.	PesananDetail	Menambahkan detail pesanan	PesananDetail.create(data)
		Menampilkan semua detail pesanan berdasarkan pesanan_id	PesananDetail.findAll({where: {pesanan_id}})
7.	OTP	Membuat kode OTP baru	OTP.create(data)
		Validasi OTP aktif (belum terpakai & belum kedaluwarsa	OTP.findOne({ where: { otp_code, penjual_id, is_used: false, expired_at: { [Op.gt]: new Date() } } })
		Update status OTP menjadi terpakai	otp.save()
8.	PasswordResetTokens	Membuat token reset password baru	PasswordResetTokens.create(data)
		Mencari token reset berdasarkan penjual_id	PasswordResetTokens.findOne({where: {penjual_id}})
		Update status token	PasswordResetTokens.destroy({ where: { id } })
9.	BlacklistedTokens	Membuat blacklist token	BlacklistedTokens.create(data)
		Cek token di blacklist (masih berlaku)	BlacklistedTokens.findOne({ where: { token, expired_at: { [Op.gt]: new Date() } } })
10.	WANotifLogs	Menyimpan log notifikasi WA	WANotifLogs.create(data)
		Menampilkan log notifikasi berdasarkan pesanan_id	WANotifLogs.findAll({where: {pesanan_id}})
11.	authRoutes	Menerima permintaan POST untuk pendaftaran penjual baru.	router.post('/register')
		Menerima permintaan POST untuk autentikasi (login) penjual.	router.post('/login')
		Menerima permintaan POST untuk verifikasi OTP.	router.post('/verify-otp')
		Menerima permintaan POST untuk mengirim ulang OTP.	router.post('/resend-otp')
		Menerima permintaan POST untuk proses lupa kata sandi.	router.post('/forgot-password')
		Menerima permintaan POST untuk mengatur ulang kata sandi.	router.post('/reset-password')
		Menerima permintaan POST untuk keluar (logout).	router.post('/logout')
12.	dashboardRoutes	Menerima permintaan GET untuk mengambil data dashboard.	router.get('/')
13.	keranjangRoutes	Menerima permintaan POST untuk menambah item ke keranjang.	router.post('/keranjang')
		Menerima permintaan GET untuk mengambil isi keranjang.	router.get('/keranjang')
		Menerima permintaan PUT untuk memperbarui item di keranjang.	router.put('/keranjang/:id')
		Menerima permintaan DELETE untuk menghapus item dari keranjang.	router.delete('/keranjang/:id')
14.	kiosRoutes	Menerima permintaan POST untuk membuat kios baru.	router.post('/')
		Menerima permintaan GET untuk mengambil profil kios milik penjual.	router.get('/profil')
		Menerima permintaan PUT untuk memperbarui data kios.	router.put('/:id')
		Menerima permintaan GET untuk mengambil semua data kios.	router.get('/')

No	Nama Kelas	Kandidat Perilaku	Nama Perilaku
		Menerima permintaan GET untuk data kios di homepage.	<i>router.get('/homepage')</i>
		Menerima permintaan GET untuk mencari kios.	<i>router.get('/search')</i>
		Menerima permintaan GET untuk mengambil detail satu kios.	<i>router.get('/:id')</i>
		Menerima permintaan GET untuk mengambil semua menu dari kios tertentu.	<i>router.get('/:id/menus')</i>
		Menerima permintaan GET untuk mencari menu di dalam kios tertentu.	<i>router.get('/:id/menus/search')</i>
15.	menuRoutes	Menerima permintaan GET untuk mengambil menu terbaru.	<i>router.get('/new')</i>
		Menerima permintaan GET untuk mencari menu secara global.	<i>router.get('/search')</i>
		Menerima permintaan GET untuk mencari menu di kios tertentu.	<i>router.get('/kios/:id/search')</i>
		Menerima permintaan GET untuk mengambil detail menu.	<i>router.get('/buyer/:id')</i>
		Menerima permintaan POST untuk menambah menu baru.	<i>router.post('/')</i>
		Menerima permintaan GET untuk mengambil semua menu milik penjual.	<i>router.get('/')</i>
		Menerima permintaan GET untuk mengambil menu dengan paginasi.	<i>router.get('/paginated')</i>
		Menerima permintaan GET untuk mengambil detail menu miliknya.	<i>router.get('/seller/:id')</i>
		Menerima permintaan PUT untuk memperbarui data menu.	<i>router.put('/:id')</i>
		Menerima permintaan DELETE untuk menghapus menu.	<i>router.delete('/:id')</i>
16.	midtransRoutes	Menerima permintaan POST untuk membuat transaksi pembayaran.	<i>router.post('/create-transaction')</i>
		Menerima notifikasi status transaksi dari Midtrans (webhook).	<i>router.post('/notification')</i>
17.	penjualRoutes	Menerima permintaan GET untuk mengambil data profil penjual.	<i>router.get('/profil')</i>
		Menerima permintaan PUT untuk memperbarui data profil penjual.	<i>router.put('/profil')</i>
18.	pesananRoutes	Menerima permintaan POST untuk membuat pesanan baru.	<i>router.post('/pesanan')</i>
		Menerima permintaan GET untuk mengambil daftar pesanan miliknya.	<i>router.get('/pesanan')</i>
		Menerima permintaan GET untuk mengambil detail pesanan.	<i>router.get('/pesanan/:id')</i>
		Menerima permintaan GET untuk mengambil status pesanan saat ini.	<i>router.get('/pesanan/:id/status')</i>
		Menerima permintaan GET untuk mengambil daftar pesanan yang masuk.	<i>router.get('/pesanan-masuk')</i>
		Menerima permintaan GET untuk menghitung jumlah pesanan yang masuk.	<i>router.get('/pesanan-masuk/count')</i>
		Menerima permintaan GET untuk mengambil detail pesanan yang masuk.	<i>router.get('/pesanan-masuk/:id')</i>
		Menerima permintaan GET untuk mengambil riwayat semua pesanan.	<i>router.get('/pesanan/riwayat')</i>
		Menerima permintaan GET untuk mengambil detail dari riwayat pesanan.	<i>router.get('/pesanan/riwayat/:id')</i>
		Menerima permintaan PATCH untuk memperbarui status pesanan.	<i>router.patch('/pesanan/:id/status')</i>
19.	searchRoutes	Menerima permintaan GET untuk melakukan pencarian gabungan (kios dan menu).	<i>router.get('/')</i>
20.	authController	Mengelola proses pendaftaran penjual baru.	<i>register(req, res)</i>

No	Nama Kelas	Kandidat Perilaku	Nama Perilaku
		Mengelola proses verifikasi akun menggunakan OTP.	verifyOtp(req, res)
		Mengelola proses pengiriman ulang kode OTP.	resendOtp(req, res)
		Mengelola proses autentikasi (login) penjual.	login(req, res)
		Mengelola proses permintaan lupa kata sandi.	forgotPassword(req, res)
		Mengelola proses pengaturan ulang kata sandi menggunakan <i>token</i> .	resetPassword(req, res)
		Mengelola proses keluar (logout) dari sesi pengguna.	logout(req, res)
21.	dashboardController	Mengelola proses pengambilan data ringkasan untuk dashboard penjual.	getDashboardData(req, res)
		Mengelola proses konversi status pesanan internal menjadi label teks.	getStatusLabel(tipe_pengantaran, statusDb)
22.	keranjangController	Mengelola proses penambahan atau pembaruan item di keranjang.	addToKeranjang(req, res)
		Mengelola proses pengambilan seluruh isi keranjang belanja.	getKeranjang(req, res)
		Mengelola proses pembaruan jumlah atau catatan item di keranjang.	updateKeranjangItem(req, res)
		Mengelola proses penghapusan item spesifik dari keranjang.	removeFromKeranjang(req, res)
23.	kiosController	Mengelola proses pendaftaran kios baru oleh penjual.	createKios(req, res)
		Mengelola pengambilan data kios untuk halaman utama (homepage).	getKiosHomepage(req, res)
		Mengelola proses pencarian kios berdasarkan nama.	searchKios(req, res)
		Mengelola pengambilan semua data kios yang terdaftar.	getAllKios(req, res)
		Mengelola pengambilan semua menu yang dimiliki oleh kios tertentu.	getMenusByKios(req, res)
		Mengelola proses pembaruan profil kios oleh penjual.	updateKios(req, res)
		Mengelola pengambilan profil kios milik penjual yang sedang login.	getKiosByPenjual(req, res)
		Mengelola pengambilan detail satu kios spesifik (untuk pembeli).	getKiosDetail(req, res)
24.	menuController	Mengelola pengambilan semua menu milik penjual.	getAllMenu(req, res)
		Mengelola proses penambahan menu baru oleh penjual.	addMenu(req, res)
		Mengelola proses pembaruan data menu milik penjual.	updateMenu(req, res)
		Mengelola pengambilan detail menu spesifik (untuk penjual).	getMenuById(req, res)
		Mengelola proses penghapusan menu milik penjual.	deleteMenu(req, res)
		Mengelola pengambilan menu milik penjual dengan paginasi.	getMenusPaginated(req, res)
		Mengelola pengambilan menu-menu terbaru (untuk pembeli).	getNewMenus(req, res)
		Mengelola proses pencarian menu secara global (untuk pembeli).	searchMenus(req, res)
		Mengelola proses pencarian menu di dalam kios tertentu.	searchMenusByKios(req, res)
		Mengelola pengambilan detail menu spesifik (untuk pembeli).	getMenuByIdForBuyer(req, res)
25.	midtransController	Mengelola proses pembuatan transaksi pembayaran melalui Midtrans.	createTransaction(req, res)

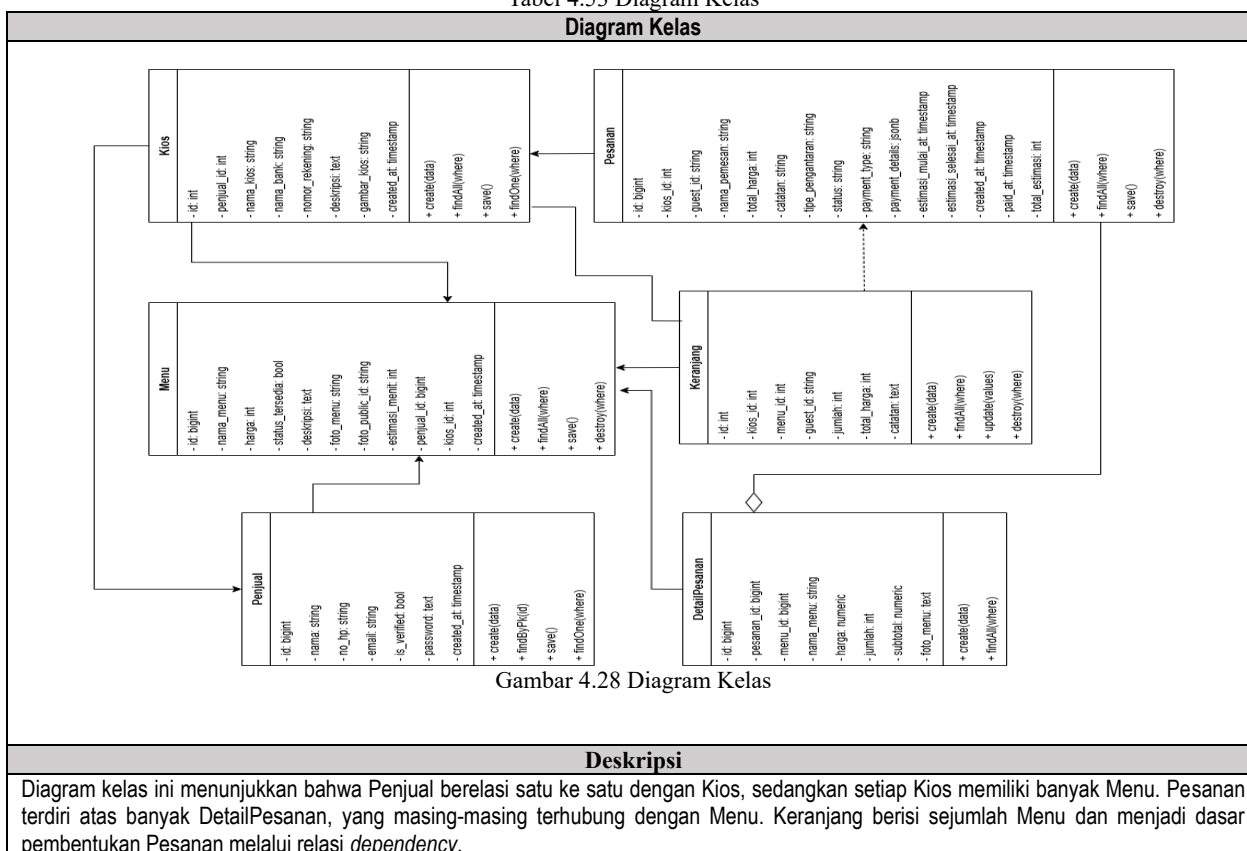
No	Nama Kelas	Kandidat Perilaku	Nama Perilaku
		Mengelola notifikasi status pembayaran dari Midtrans.	handleNotification(req, res)
26.	penjualController	Mengelola pengambilan data profil penjual yang sedang login.	getProfile(req, res)
		Mengelola proses pembaruan data profil penjual.	updateProfil(req, res)
27.	pesananController	Mengonversi status internal pesanan menjadi label teks untuk ditampilkan.	getStatusLabel(tipe_pengantaran, statusDb)
		Mengelola pengiriman notifikasi pesanan baru ke WhatsApp penjual.	notifyPenjual(kiosId, pesananId)
		Mengelola pengiriman notifikasi pesanan selesai ke WhatsApp pembeli.	notifyPembeliPesananSelesai(pesananId)
		Mengelola proses pembuatan pesanan baru dari keranjang pembeli.	buatPesanan(req, res)
		Mengelola pengambilan daftar riwayat pesanan milik pembeli.	getPesananByGuest(req, res)
		Mengelola pengambilan detail pesanan spesifik milik pembeli.	getDetailPesanan(req, res)
		Mengelola pengambilan status dan estimasi pesanan terkini.	getStatusPesananGuest(req, res)
		Mengelola pengambilan daftar pesanan yang sedang aktif.	getPesananMasuk(req, res)
		Mengelola pengambilan detail pesanan aktif spesifik.	getDetailPesananMasuk(req, res)
		Mengelola penghitungan jumlah pesanan aktif (untuk notifikasi).	countPesananMasuk(req, res)
		Mengelola proses pembaruan status pesanan oleh penjual.	updateStatusPesanan(req, res)
		Mengelola pengambilan riwayat pesanan yang telah selesai.	getRiwayatPesanan(req, res)
		Mengelola pengambilan detail riwayat pesanan yang telah selesai.	getDetailRiwayatPesanan(req, res)
28.	searchController	Mengelola proses pencarian gabungan untuk kios dan menu	searchAll(req, res)
29.	whatsappService	Menjalankan proses koneksi ke WhatsApp dan handle QR, reconnect, serta event WA	connectToWhatsApp()
		Mengirim pesan teks ke nomor WhatsApp	sendWaMessage(phoneNumber, message)
		Mengirim pesan OTP ke nomor WhatsApp	sendOtpMessage(phoneNumber, otpCode)
		Mengambil socket WhatsApp global yang sedang aktif	getWASocket()
30.	getGuestId	Menjalankan logika bisnis pengambilan Guest ID dari request	getGuestId(req)
31.	cloudinaryUtil	Menjalankan logika bisnis konfigurasi Cloudinary menggunakan environment variables agar dapat digunakan untuk upload, manage, dan akses media.	cloudinary (uploader: cloudinary.uploader.upload(...))
32.	authMiddleware	Menjalankan logika bisnis <i>middleware</i> autentikasi umum: validasi <i>token</i> JWT, cek blacklist, decode <i>token</i> , lampirkan data user ke req.user.	authMiddleware(req, res, next)
		Menjalankan logika bisnis <i>middleware</i> autentikasi user terverifikasi	verifiedMiddleware(req, res, next)
33.	uploadMiddleware	Menjalankan logika bisnis pengecekan & pembuatan folder uploads untuk menyimpan file yang diunggah.	fs.existsSync + fs.mkdirSync
		Menjalankan logika bisnis konfigurasi penyimpanan file: menentukan folder tujuan (/uploads) dan membuat nama file unik (timestamp + random + ekstensi).	multer.diskStorage({...})

No	Nama Kelas	Kandidat Perilaku	Nama Perilaku
		Menjalankan logika bisnis filter file upload: hanya memperbolehkan file gambar dengan format jpeg, jpg, png, webp, menolak selain itu.	fileFilter(req, file, cb)
		Menjalankan logika bisnis <i>middleware</i> upload: menggabungkan storage + filter + pembatasan ukuran file maksimal 2MB.	upload (instance multer)

4.2.2.4 Perancangan Kelas

Perancangan kelas disini adalah merancang diagram kelas dari kelas-kelas yang telah didefinisikan sebelumnya beserta atribut dan perilakunya. Berikut adalah diagram kelas yang diusulkan untuk sistem yang dikembangkan:

Tabel 4.53 Diagram Kelas



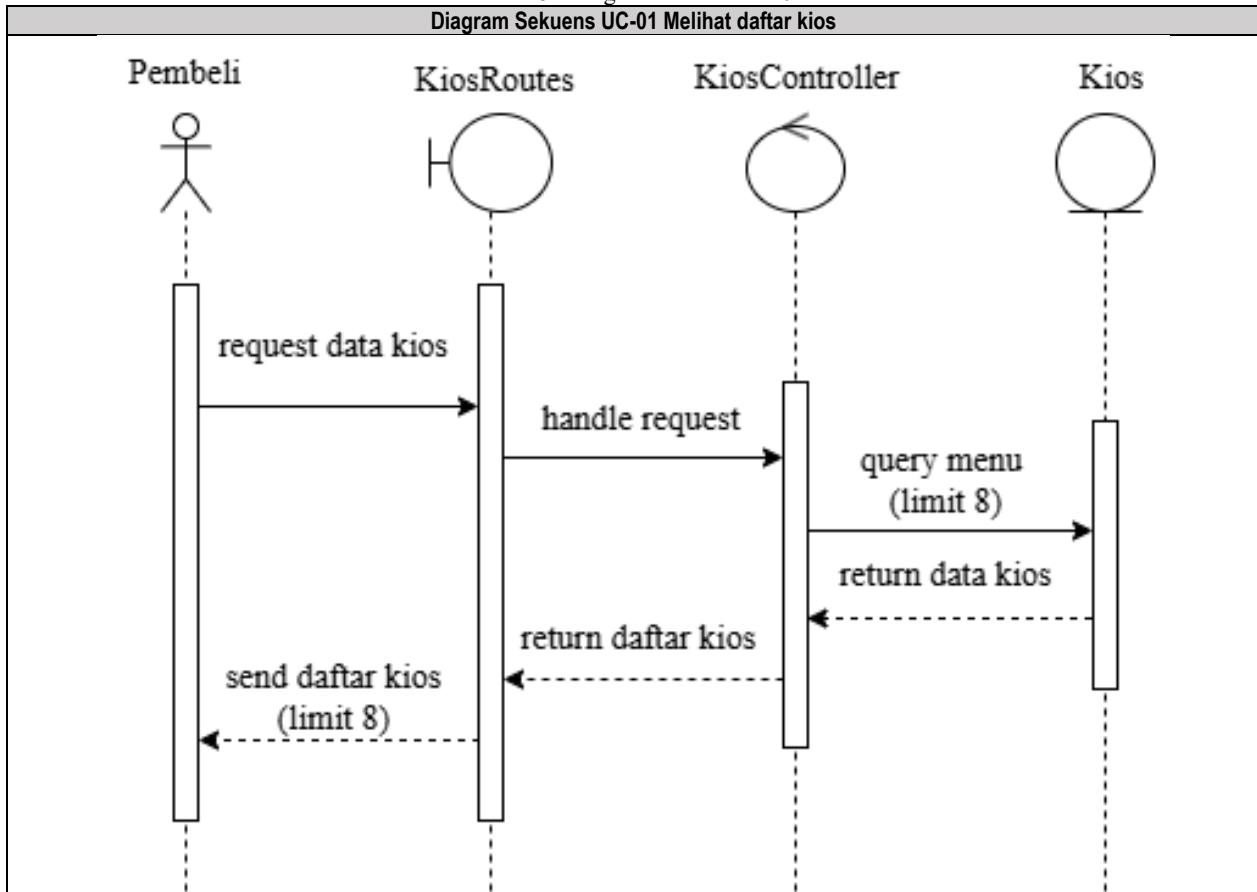
4.2.3 Pemodelan Berbasis Perilaku

Pemodelan berbasis perilaku bertujuan untuk mendeskripsikan secara rinci bagaimana sebuah perangkat lunak akan memberikan respons terhadap berbagai *event* yang berasal dari luar sistem. Proses pemodelan ini mencakup beberapa aspek utama, yang meliputi identifikasi *event* pemicu serta penggambaran interaksi melalui diagram sekuens. Kedua aspek tersebut akan diuraikan lebih lanjut pada sub-bab berikut.

4.2.3.1 Diagram Sekuens

Diagram sekuens pada dasarnya menggambarkan bagaimana perangkat lunak akan berperilaku dalam menanggapi *event-event* yang datang dari arah luar atau bagaimana perangkat lunak akan berperilaku terhadap tindakan yang muncul dari luar.

Tabel 4.54 Diagram Sekuens Uc-01
Diagram Sekuens UC-01 Melihat daftar kios



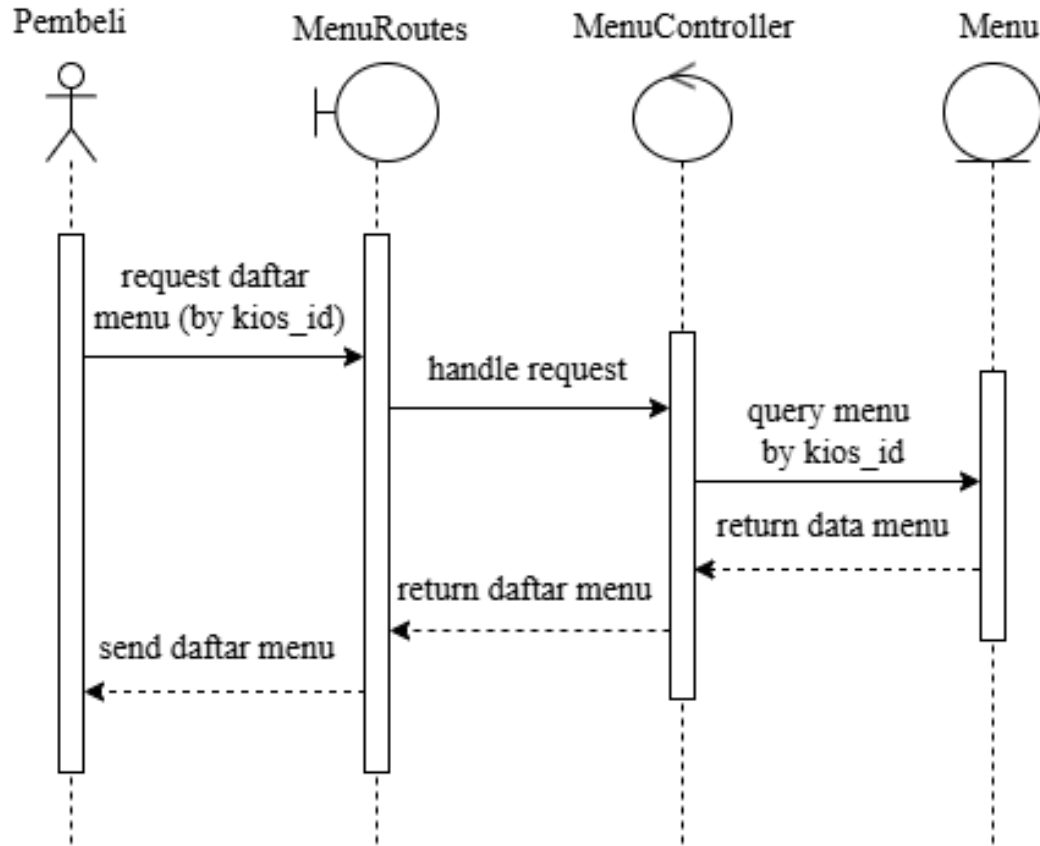
Gambar 4.29 Diagram Sekuens Uc-01

Deskripsi

Pembeli mengirim request data kios melalui KiosRoutes. Request diteruskan ke KiosController yang melakukan query daftar kios ke basis data dengan batas tertentu (limit 8). Setelah data berhasil diambil, daftar kios dikembalikan ke pembeli.

Tabel 4.55 Diagram Sekuens Uc-02

Diagram Sekuens UC-02 Melihat daftar menu per kios

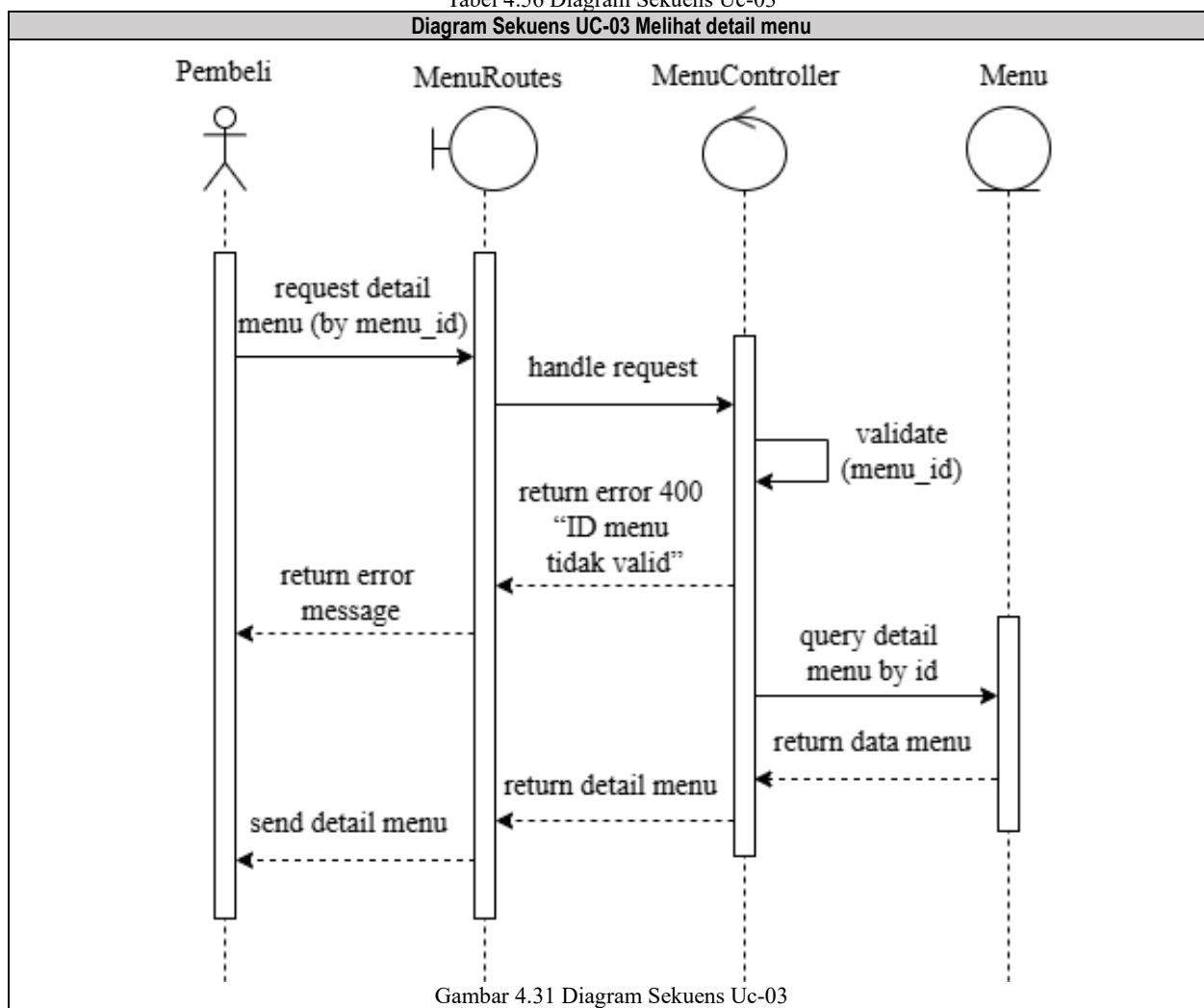


Gambar 4.30 Diagram Sekuens Uc-02

Deskripsi

Pembeli mengajukan request daftar menu berdasarkan kios_id. MenuRoutes meneruskan permintaan ke MenuController, kemudian controller melakukan query daftar menu sesuai kios. Data menu yang berhasil diambil dikembalikan ke pembeli.

Tabel 4.56 Diagram Sekuens Uc-03
Diagram Sekuens UC-03 Melihat detail menu

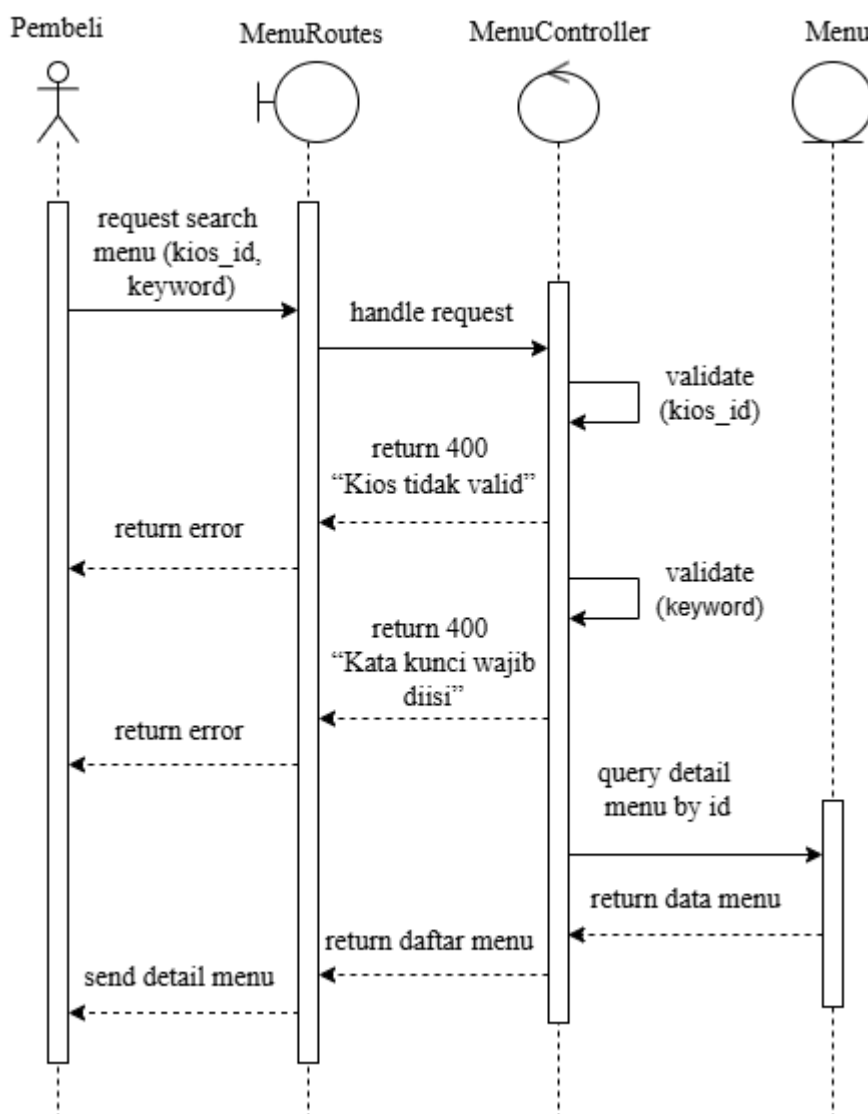


Deskripsi

Pembeli meminta detail menu berdasarkan menu_id. Request ditangani oleh MenuRoutes lalu diverifikasi oleh MenuController. Jika menu_id tidak valid, sistem mengembalikan error 400. Jika valid, controller melakukan query detail menu dan mengembalikan data menu ke pembeli.

Tabel 4.57 Diagram Sekuens Uc-04

Diagram Sekuens UC-04 Mencari menu

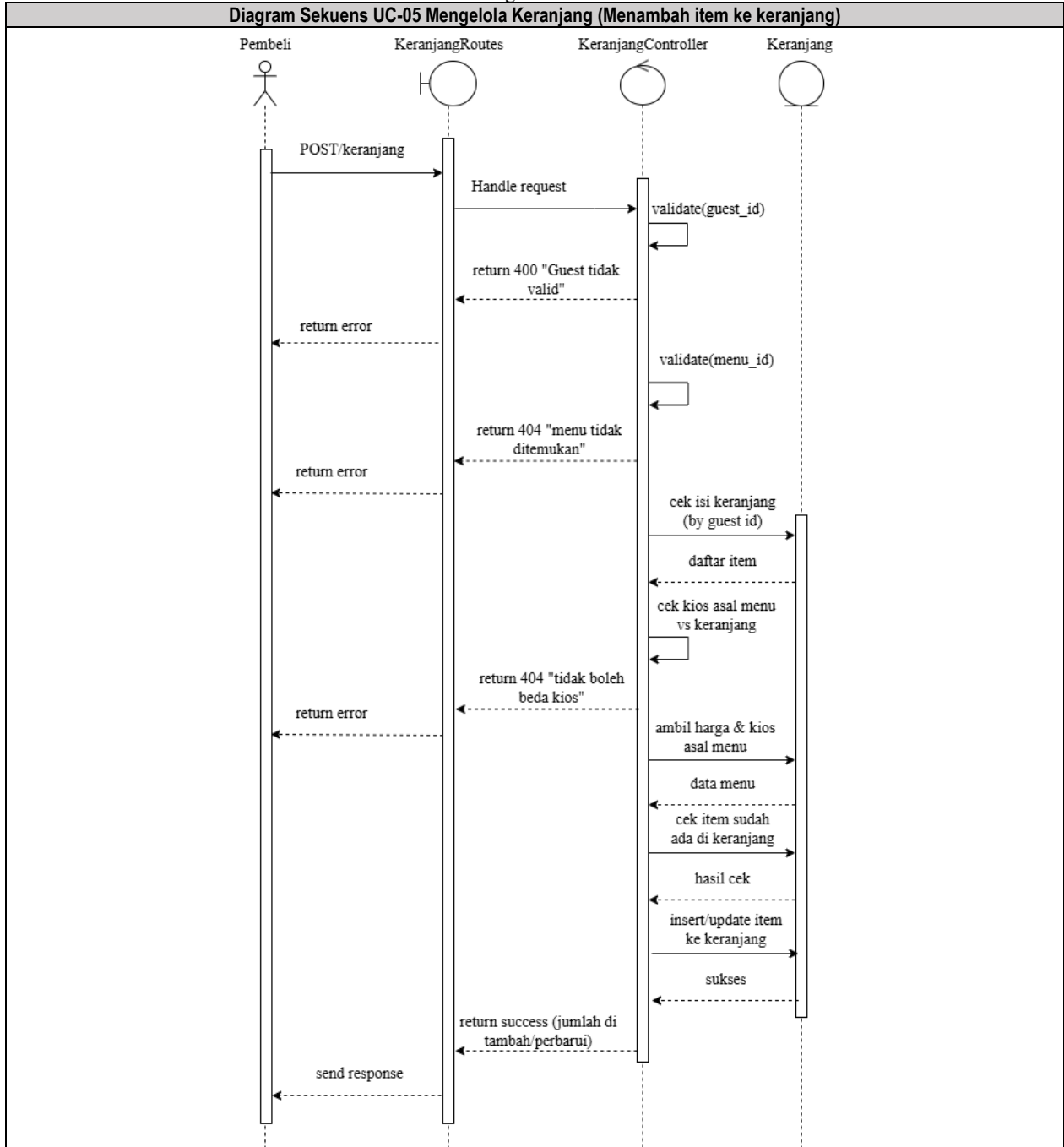


Gambar 4.32 Diagram Sekuens Uc-04

Deskripsi

Pembeli mengajukan request pencarian menu berdasarkan kios_id dan keyword. MenuController melakukan validasi input. Jika kios_id tidak valid atau kata kunci kosong, sistem mengembalikan error 400. Jika valid, sistem melakukan query data menu sesuai keyword lalu mengembalikan hasil pencarian ke pembeli.

Tabel 4.58 Diagram Sekuens Uc-05a

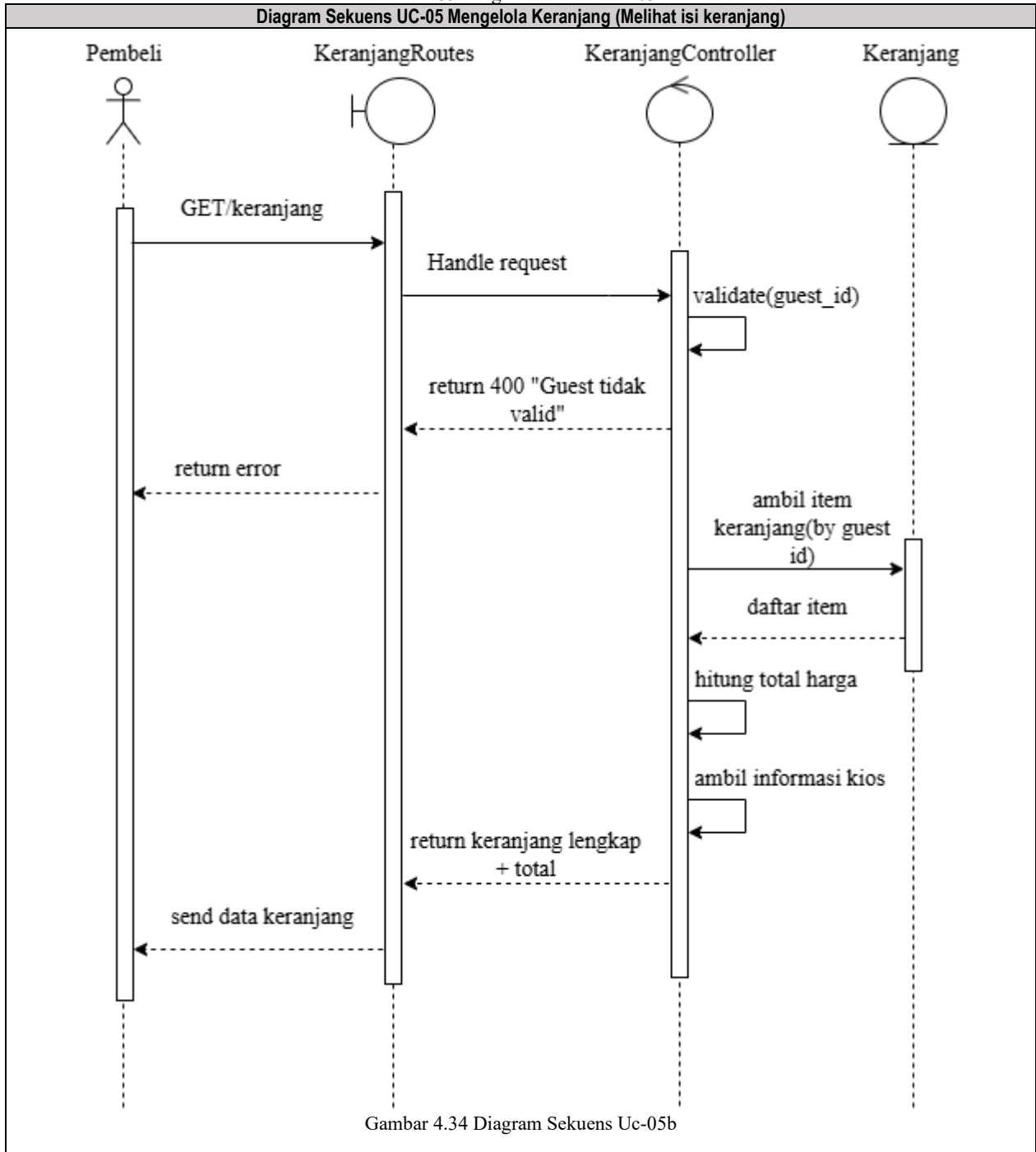


Gambar 4.33 Diagram Sekuens Uc-05a

Deskripsi

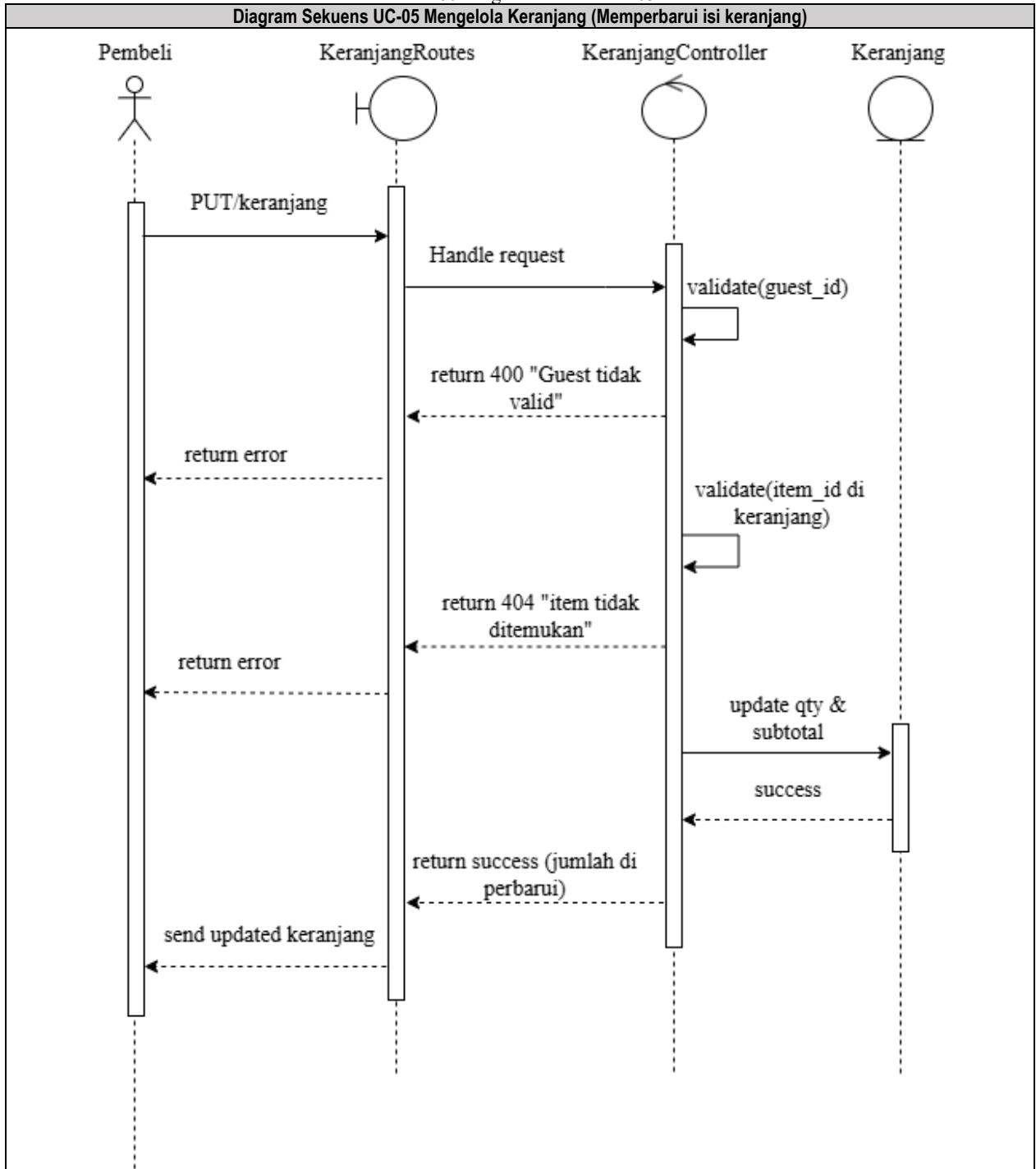
Pembeli mengirim request POST/keranjang. Sistem memvalidasi guest_id dan menu_id. Jika valid, sistem memeriksa isi keranjang. Jika item dari kiosk berbeda, sistem mengembalikan error. Jika item sama sudah ada, jumlah diperbarui; jika belum ada, item baru ditambahkan. Hasil akhir mengembalikan response sukses berisi jumlah yang diperbarui atau item baru.

Tabel 4.59 Diagram Sekuens Uc-05b

**Deskripsi**

Pembeli mengirim request GET/keranjang. Sistem memvalidasi guest_id, lalu mengambil daftar item keranjang berdasarkan guest_id. Setelah itu, sistem menghitung total harga dan mengambil informasi kios, kemudian mengembalikan response berisi data lengkap keranjang.

Tabel 4.60 Diagram Sekuens Uc-05c



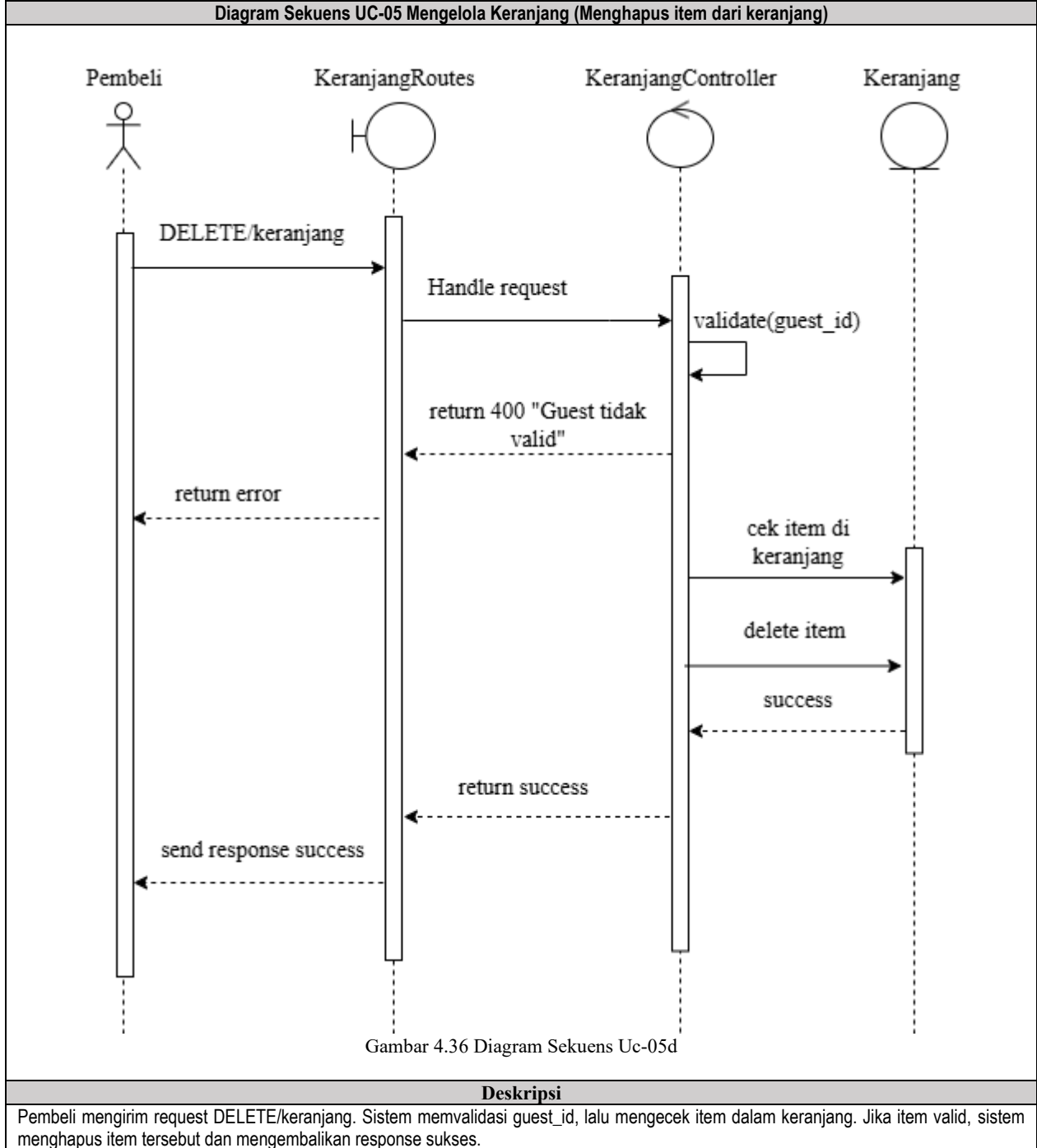
Gambar 4.35 Diagram Sekuens Uc-05c

Deskripsi

Pembeli mengirim request PUT/keranjang untuk mengubah jumlah item. Sistem memvalidasi guest_id dan item_id. Jika valid, sistem memperbarui jumlah dan subtotal item, lalu mengembalikan response sukses dengan data keranjang yang sudah diperbarui.

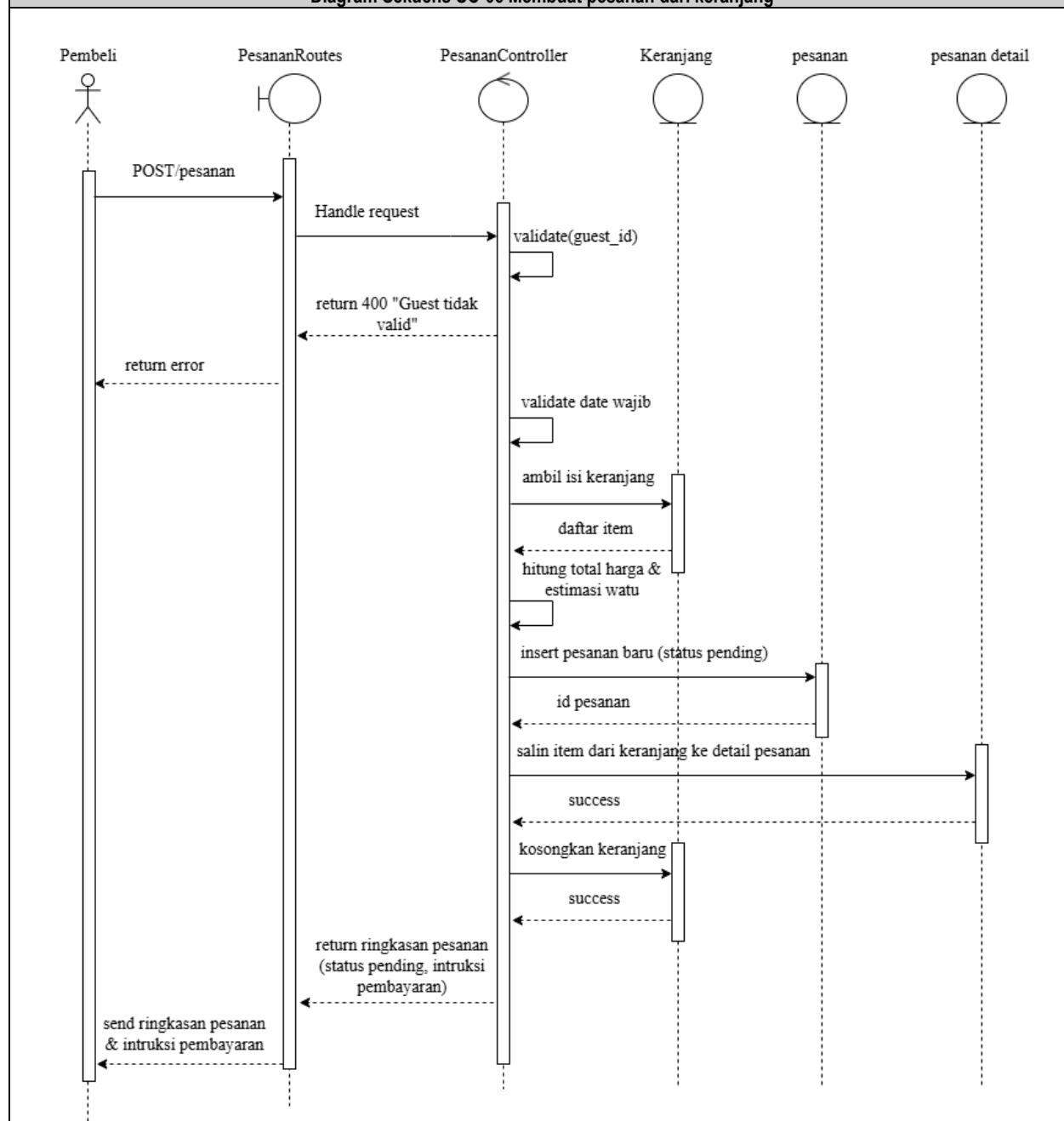
Tabel 4.61 Diagram Sekuens Uc-05d

Diagram Sekuens UC-05 Mengelola Keranjang (Menghapus item dari keranjang)



Tabel 4.62 Diagram Sekuens Uc-06

Diagram Sekuens UC-06 Membuat pesanan dari keranjang

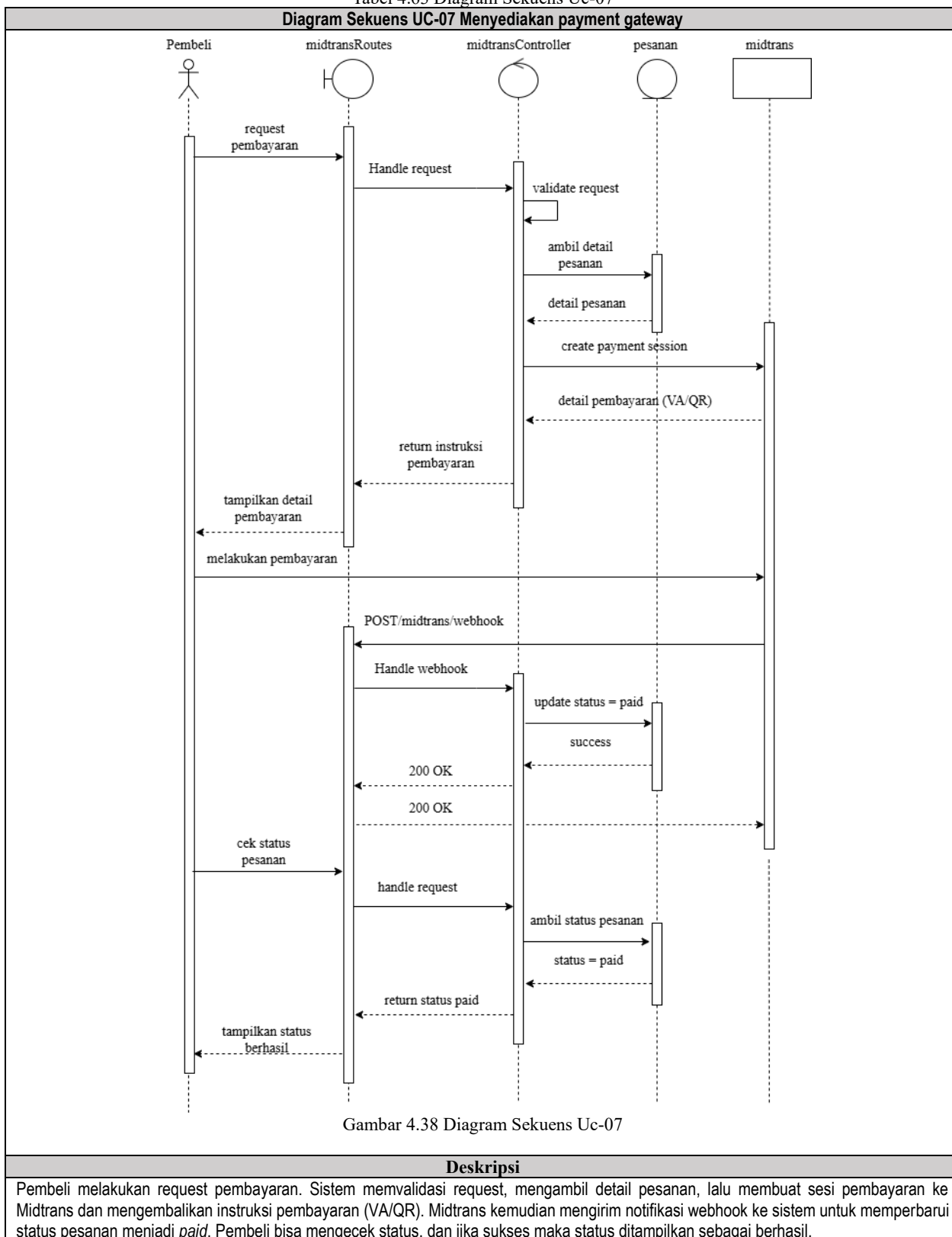


Gambar 4.37 Diagram Sekuens Uc-06

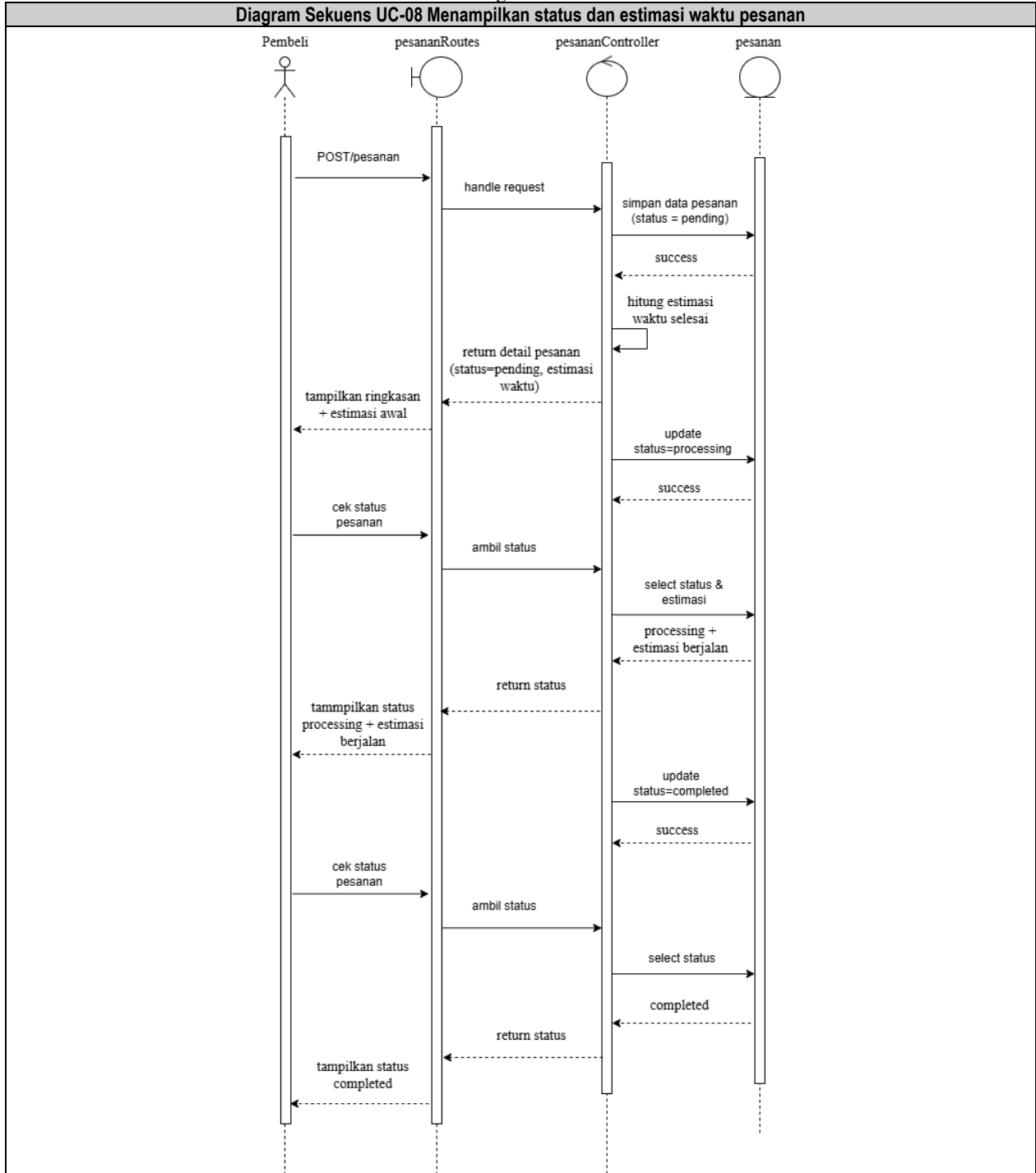
Deskripsi

Pembeli mengirim request POST/pesanan. Sistem memvalidasi guest_id serta data wajib, lalu mengambil isi keranjang. Setelah menghitung total harga dan estimasi waktu, sistem membuat data pesanan baru dengan status *pending*, menyalin item dari keranjang ke detail pesanan, lalu mengosongkan keranjang. Response dikembalikan berisi ringkasan pesanan dan instruksi pembayaran.

Tabel 4.63 Diagram Sekuens Uc-07



Tabel 4.64 Diagram Sekuens Uc-08

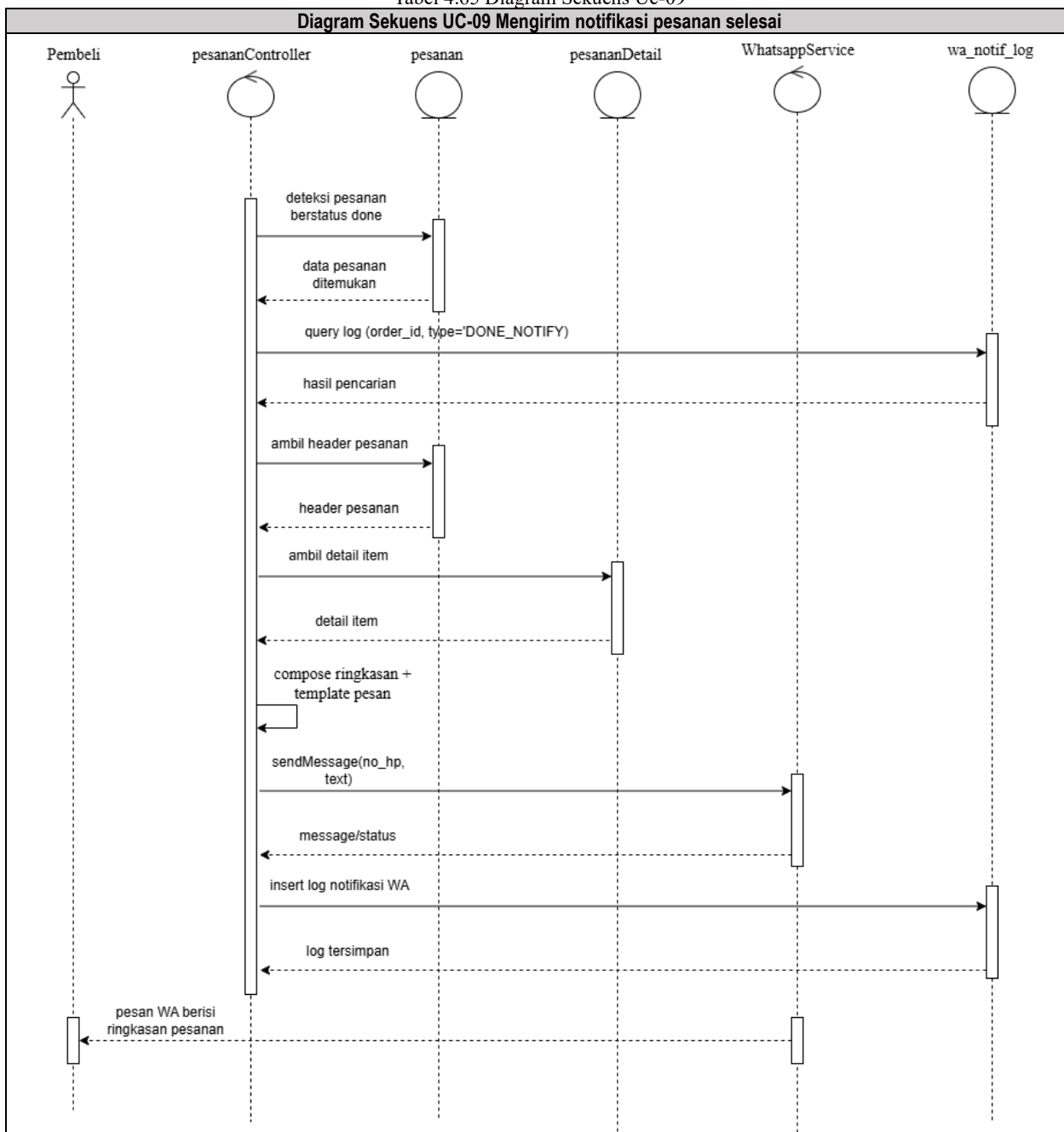


Gambar 4.39 Diagram Sekuens Uc-08

Deskripsi

Saat pesanan dibuat, sistem menyimpan data dengan status awal *pending* dan menghitung estimasi waktu selesai. Status ini dikembalikan ke pembeli. Seiring proses berjalan, status berubah menjadi *processing* dengan estimasi berjalan, hingga akhirnya selesai (*completed*). Pembeli dapat mengecek status kapan saja untuk melihat update terbaru.

Tabel 4.65 Diagram Sekuens Uc-09

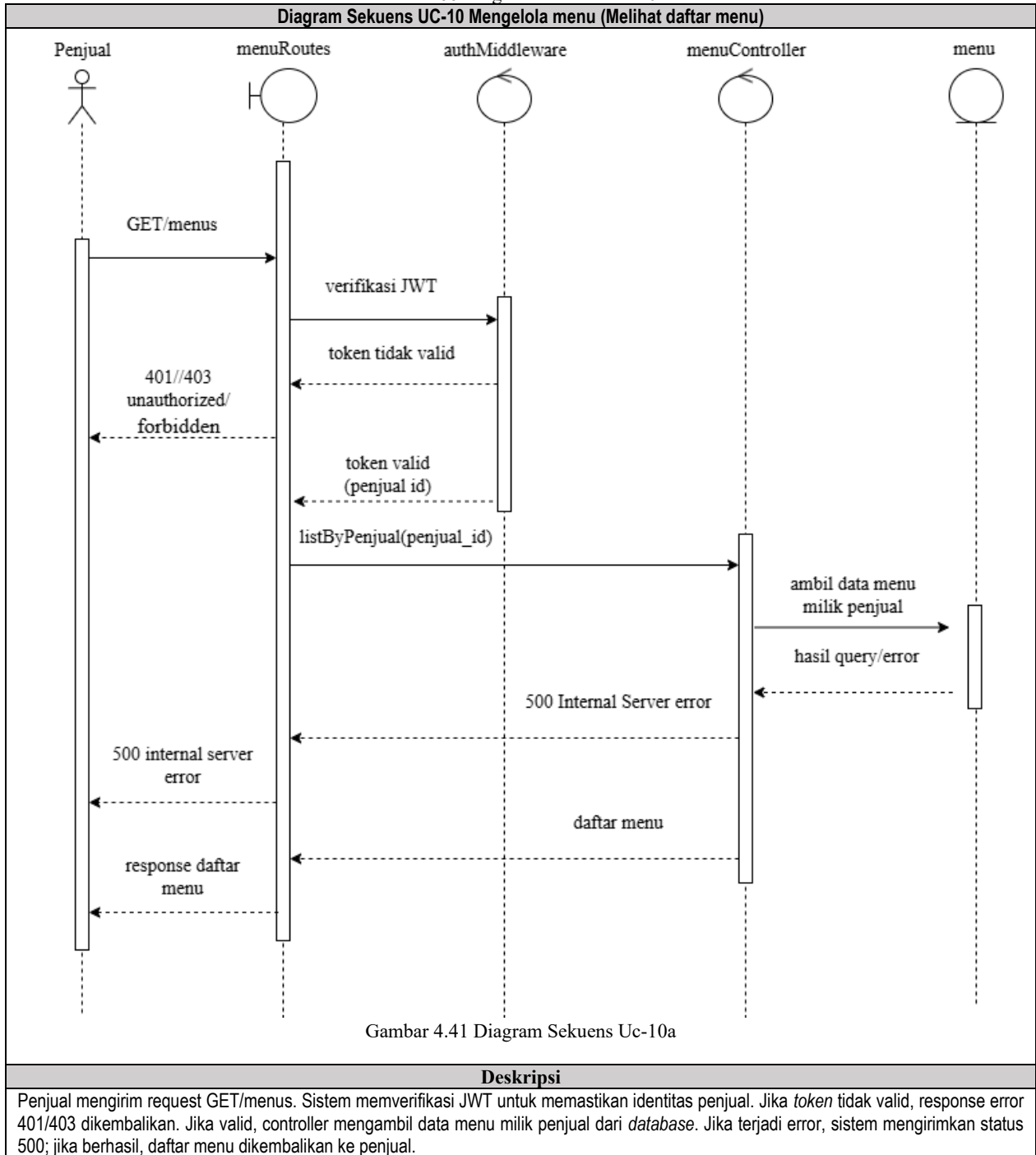


Gambar 4.40 Diagram Sekuens Uc-09

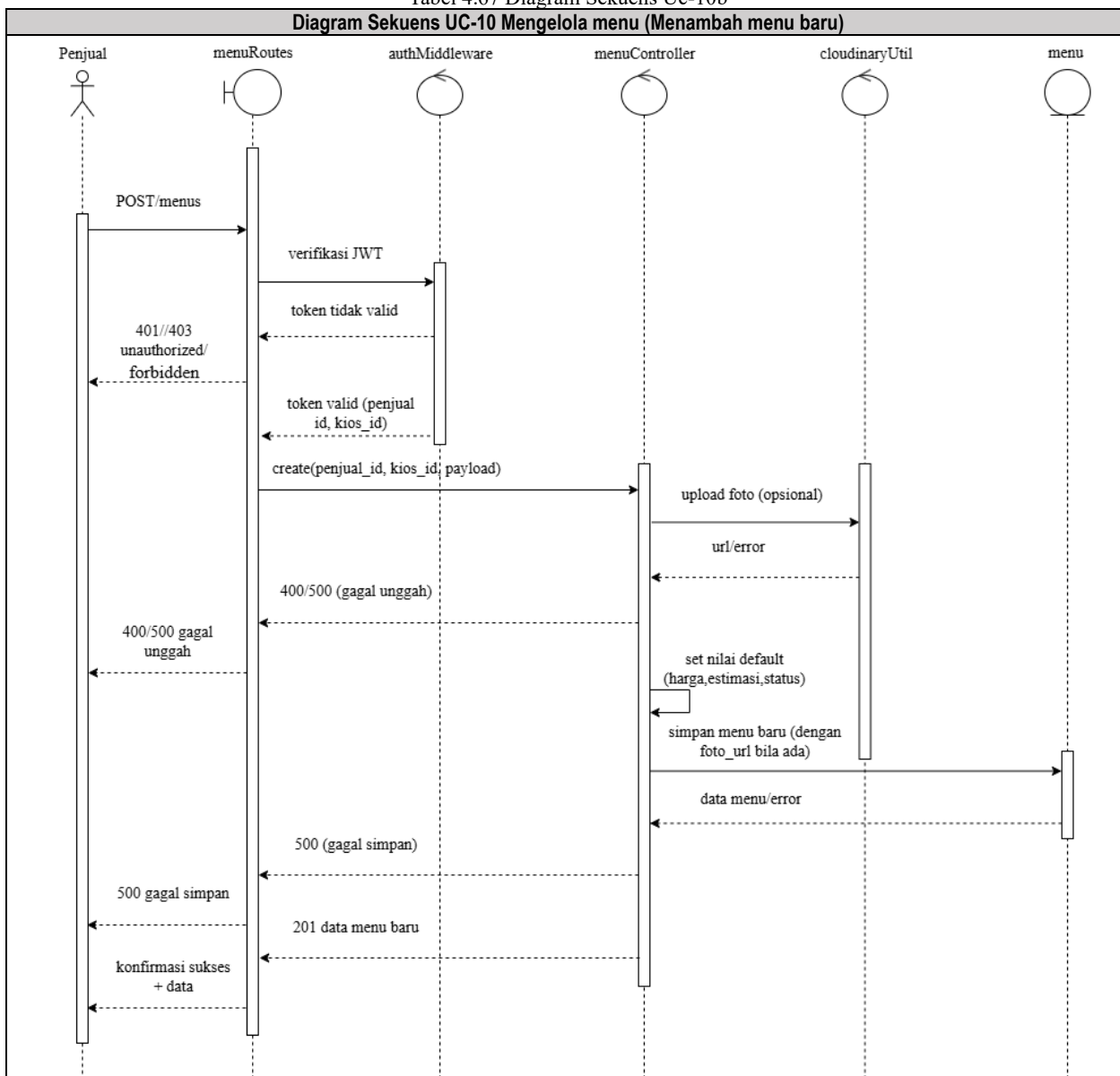
Deskripsi

Setelah pesanan berstatus *done*, sistem mengambil data pesanan dan detail item, lalu menyusun ringkasan dalam template pesan WhatsApp. Pesan ini dikirim ke nomor pembeli, dan log pengiriman notifikasi disimpan dalam sistem. Pembeli menerima pesan WA berisi ringkasan pesanan yang telah selesai.

Tabel 4.66 Diagram Sekuens Uc-10a



Tabel 4.67 Diagram Sekuens Uc-10b

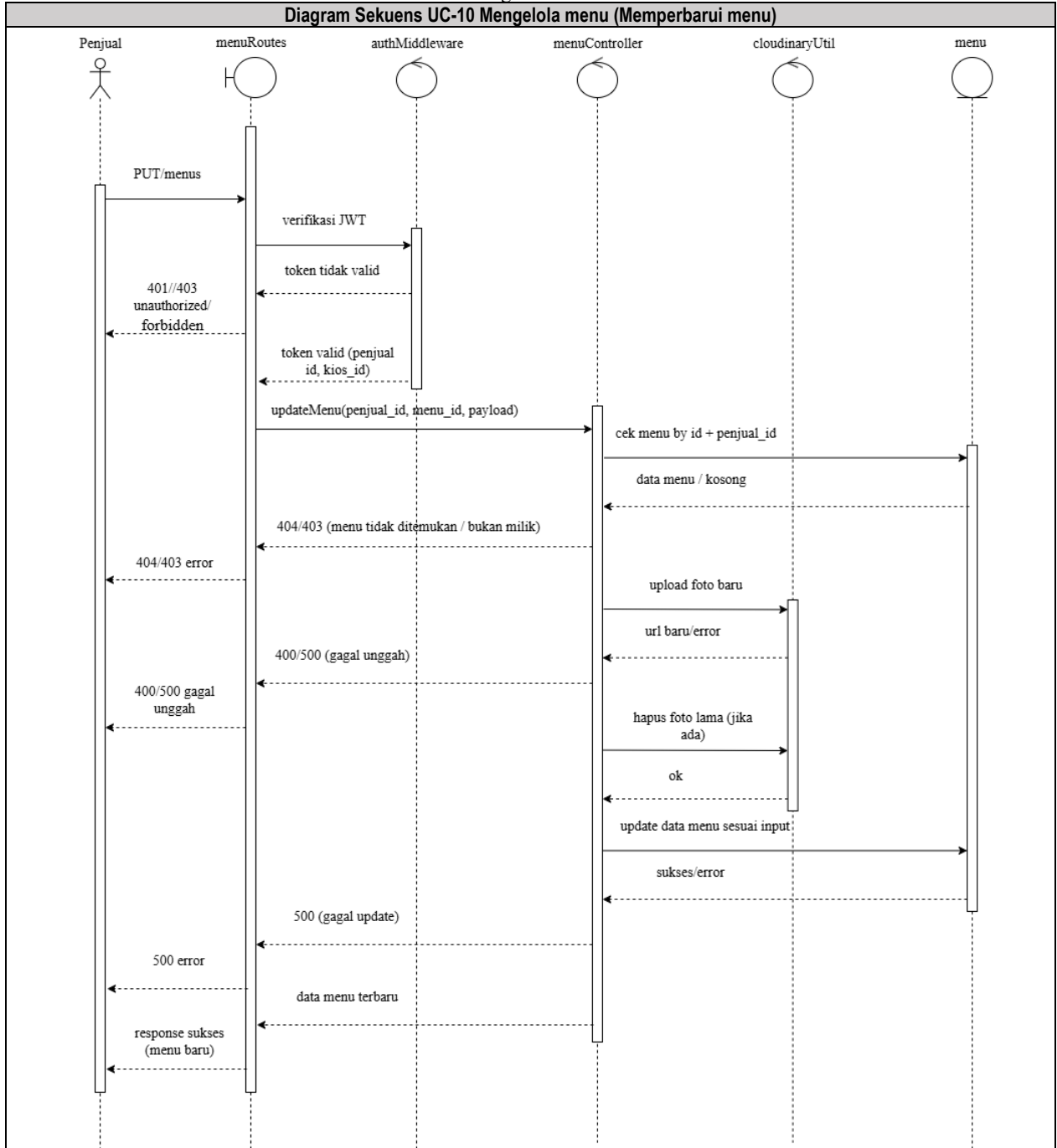


Gambar 4.42 Diagram Sekuens Uc-10b

Deskripsi

Penjual melakukan request POST/menus. Sistem memverifikasi JWT, lalu memproses data input termasuk opsi unggah foto. Jika foto diunggah, URL akan disimpan; jika gagal, response error dikembalikan. Controller kemudian menetapkan nilai default (harga, estimasi, status) dan menyimpan menu baru ke *database*. Jika berhasil, sistem merespons dengan data menu baru; jika gagal, dikembalikan error 400/500.

Tabel 4.68 Diagram Sekuens Uc-10c

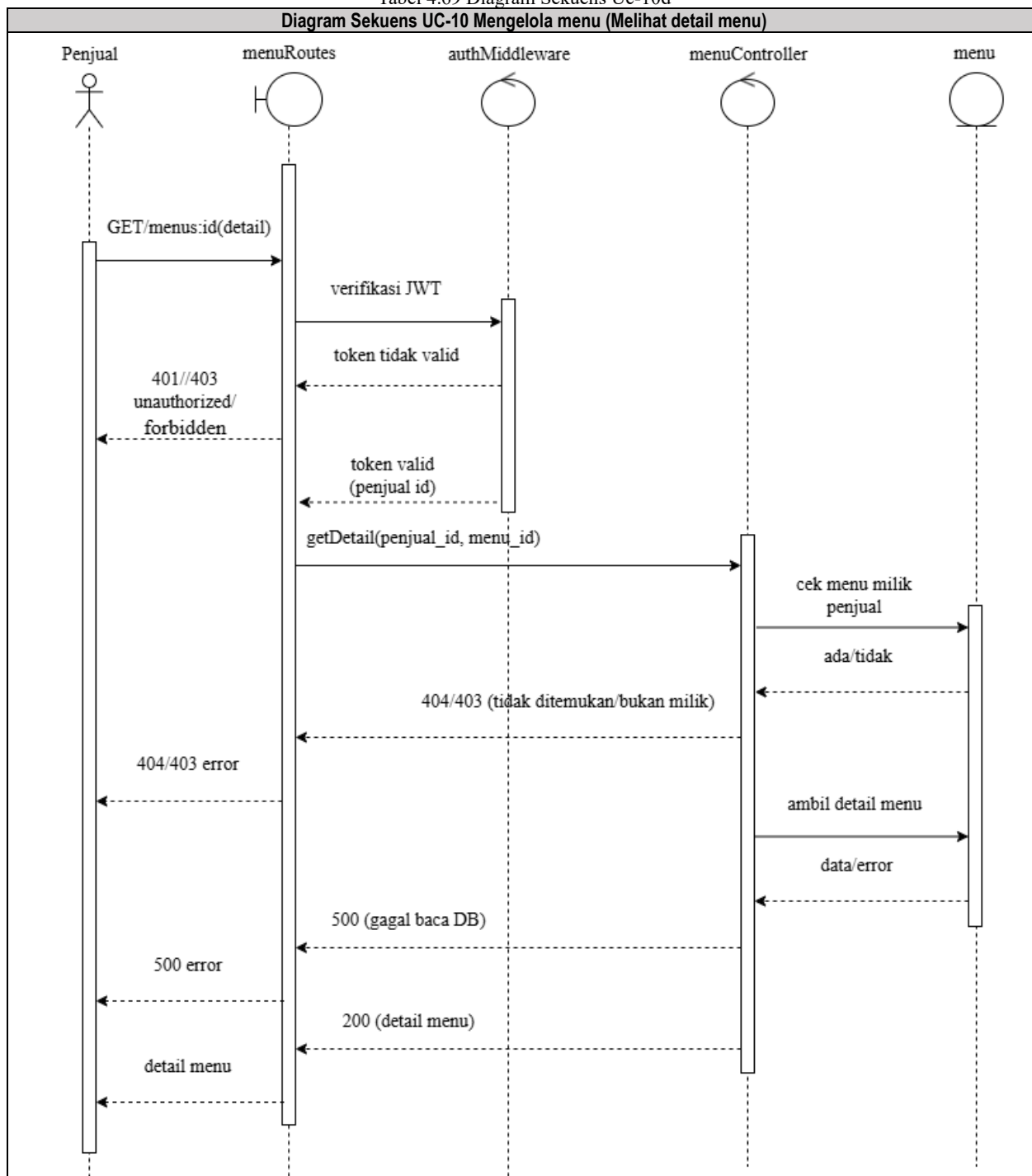


Gambar 4.43 Diagram Sekuens Uc-10c

Deskripsi

Penjual mengirim request PUT/menus. Sistem memverifikasi JWT, lalu mengecek apakah menu sesuai dengan penjual. Jika tidak ditemukan atau bukan milik penjual, sistem mengembalikan error 404/403. Jika ada foto baru, foto lama akan dihapus setelah unggahan berhasil. Data menu diperbarui sesuai input, kemudian response sukses atau error dikembalikan.

Tabel 4.69 Diagram Sekuens Uc-10d

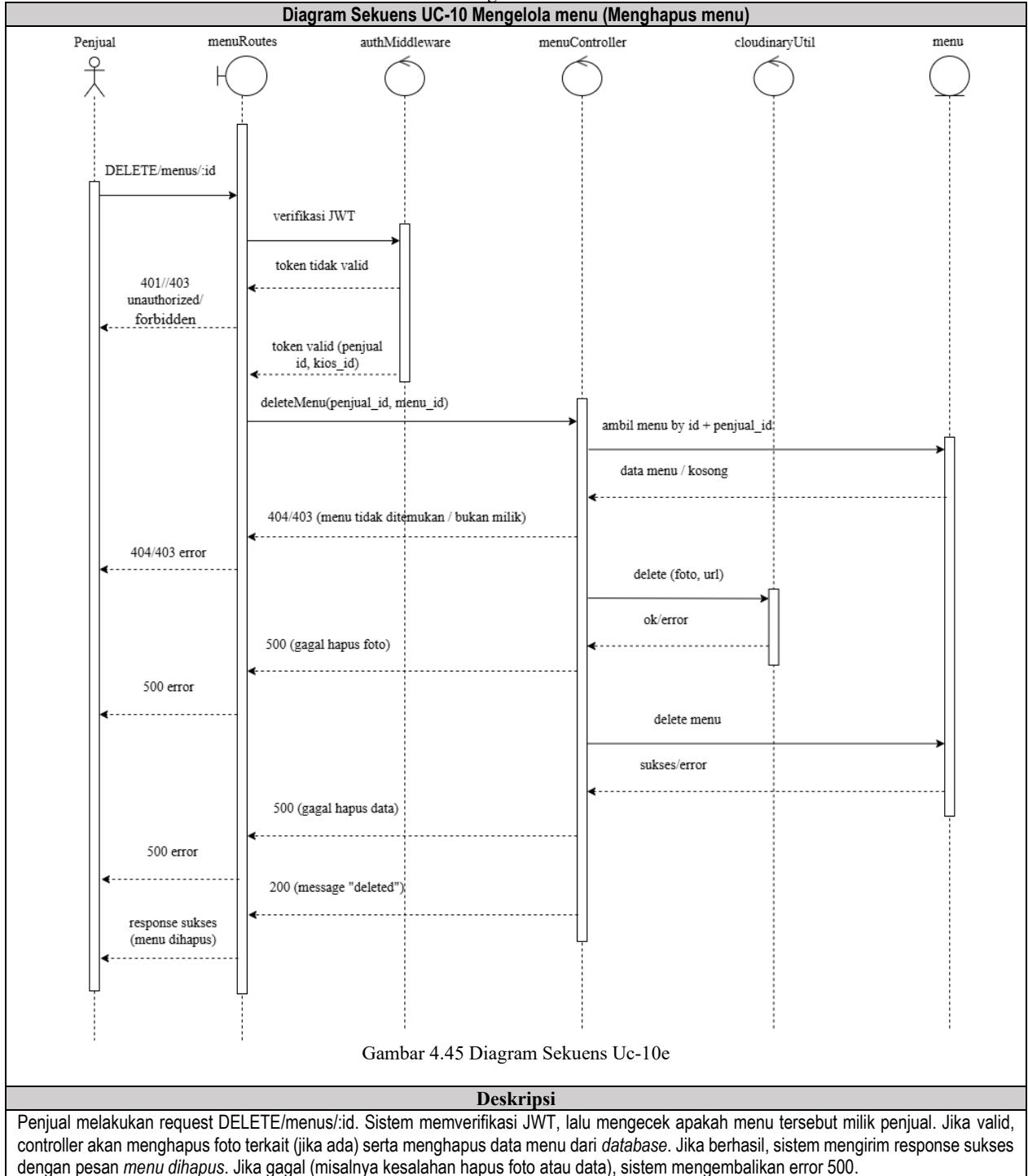


Gambar 4.44 Diagram Sekuens Uc-10d

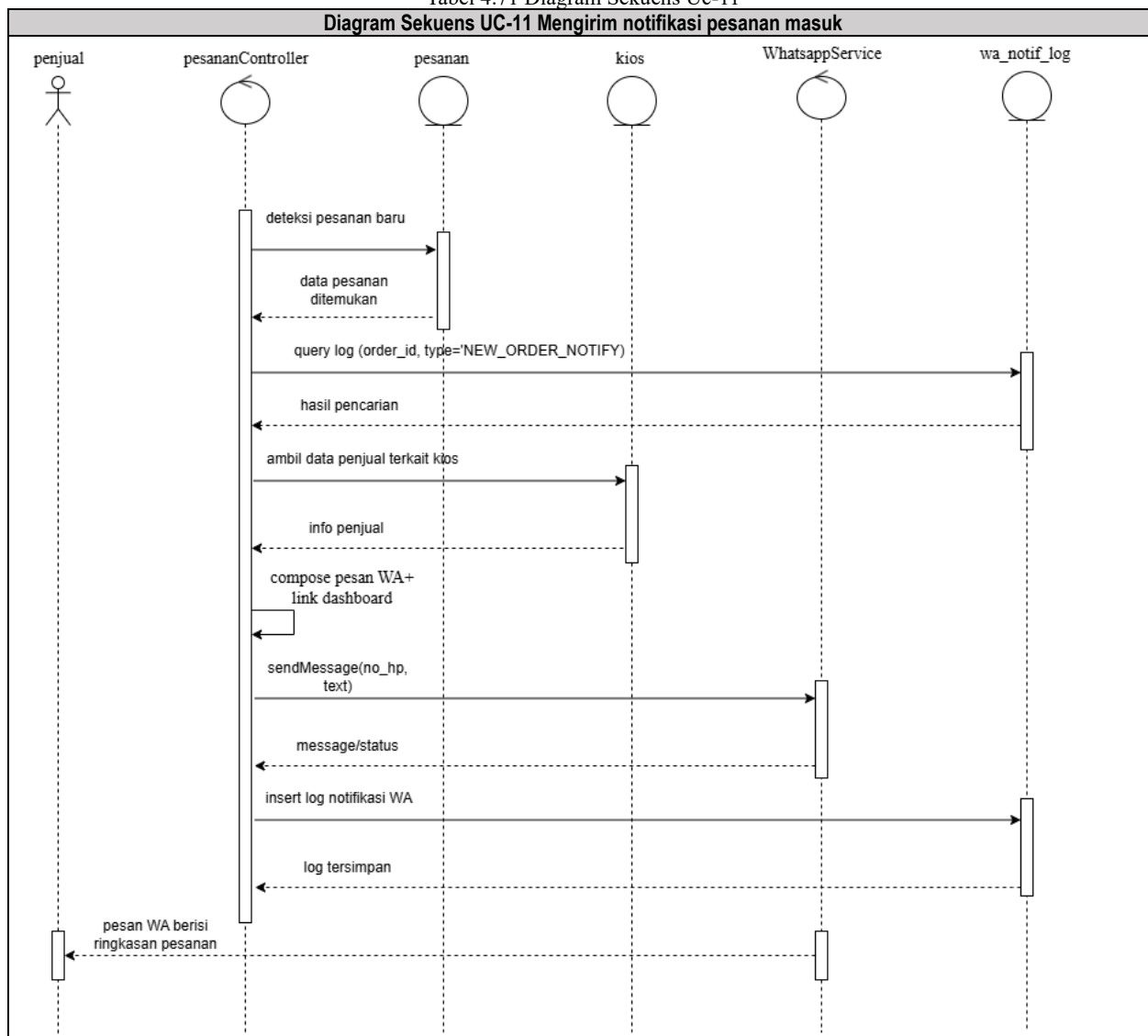
Deskripsi

Penjual mengirim request GET/menus:id. Sistem memverifikasi JWT dan memastikan menu tersebut dimiliki oleh penjual. Jika valid, detail menu diambil dari *database*. Jika menu tidak ditemukan atau bukan milik penjual, dikembalikan error 404/403. Jika terjadi kesalahan saat membaca *database*, response 500 diberikan; jika berhasil, detail menu dikembalikan.

Tabel 4.70 Diagram Sekuens Uc-10e



Tabel 4.71 Diagram Sekuens Uc-11

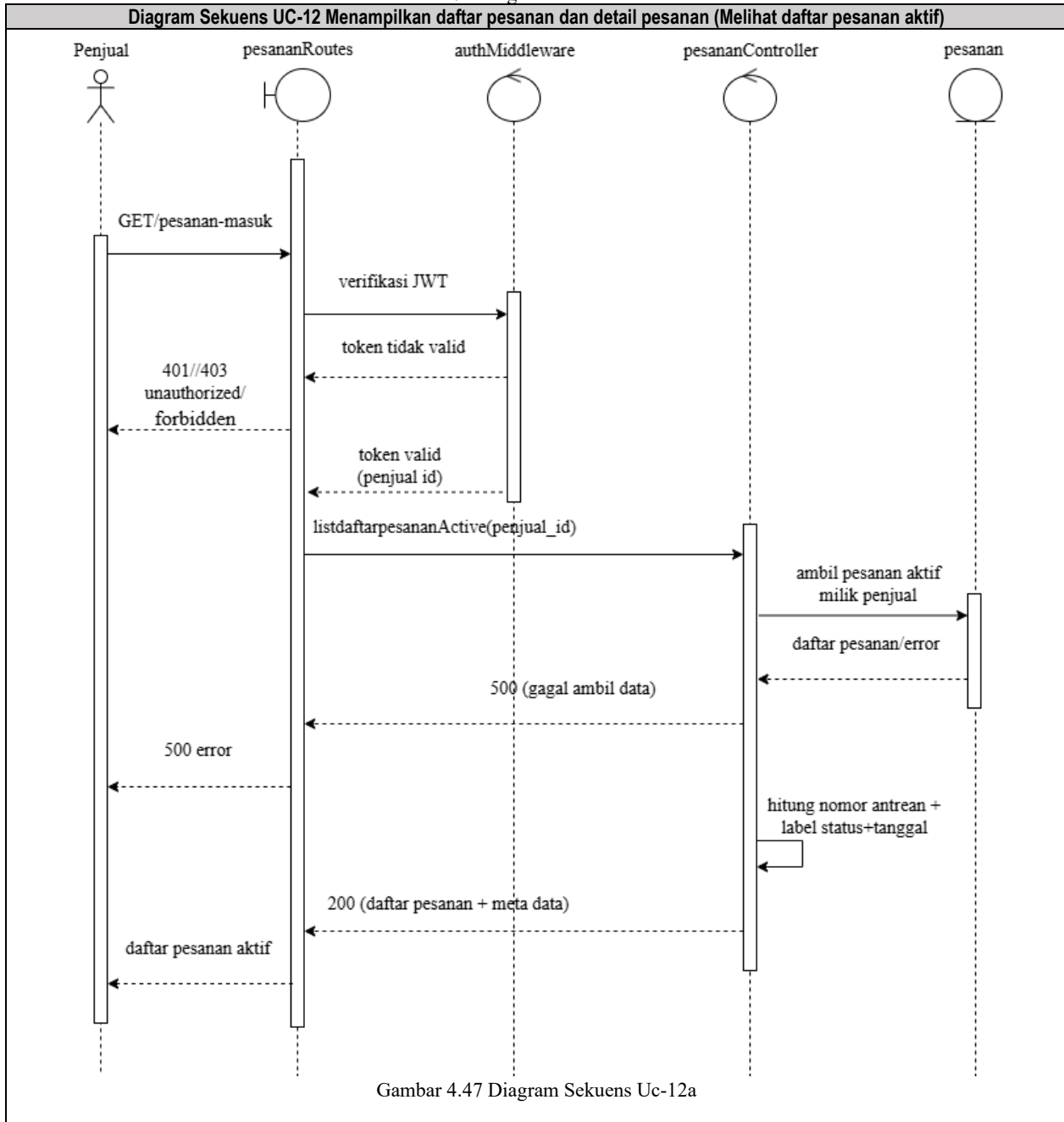


Gambar 4.46 Diagram Sekuens Uc-11

Deskripsi

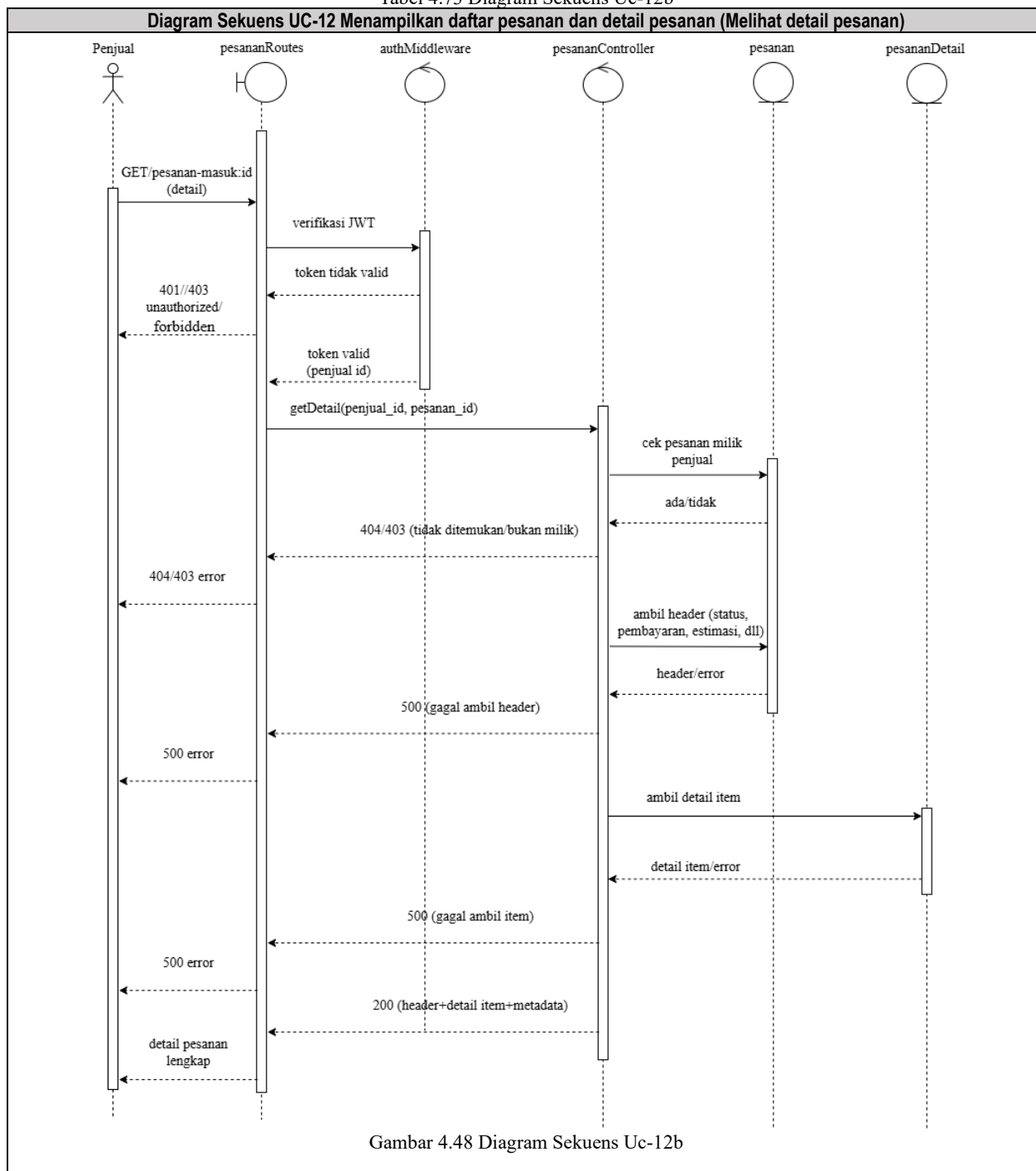
Sistem mendeteksi pesanan baru dengan status *paid*, lalu memeriksa log agar tidak mengirim notifikasi ganda. Setelah mengambil data penjual terkait kios, sistem menyusun pesan WhatsApp berisi ringkasan pesanan dan link dashboard. Pesan dikirim ke penjual melalui WhatsApp Service, kemudian log notifikasi disimpan.

Tabel 4.72 Diagram Sekuens Uc-12a

**Deskripsi**

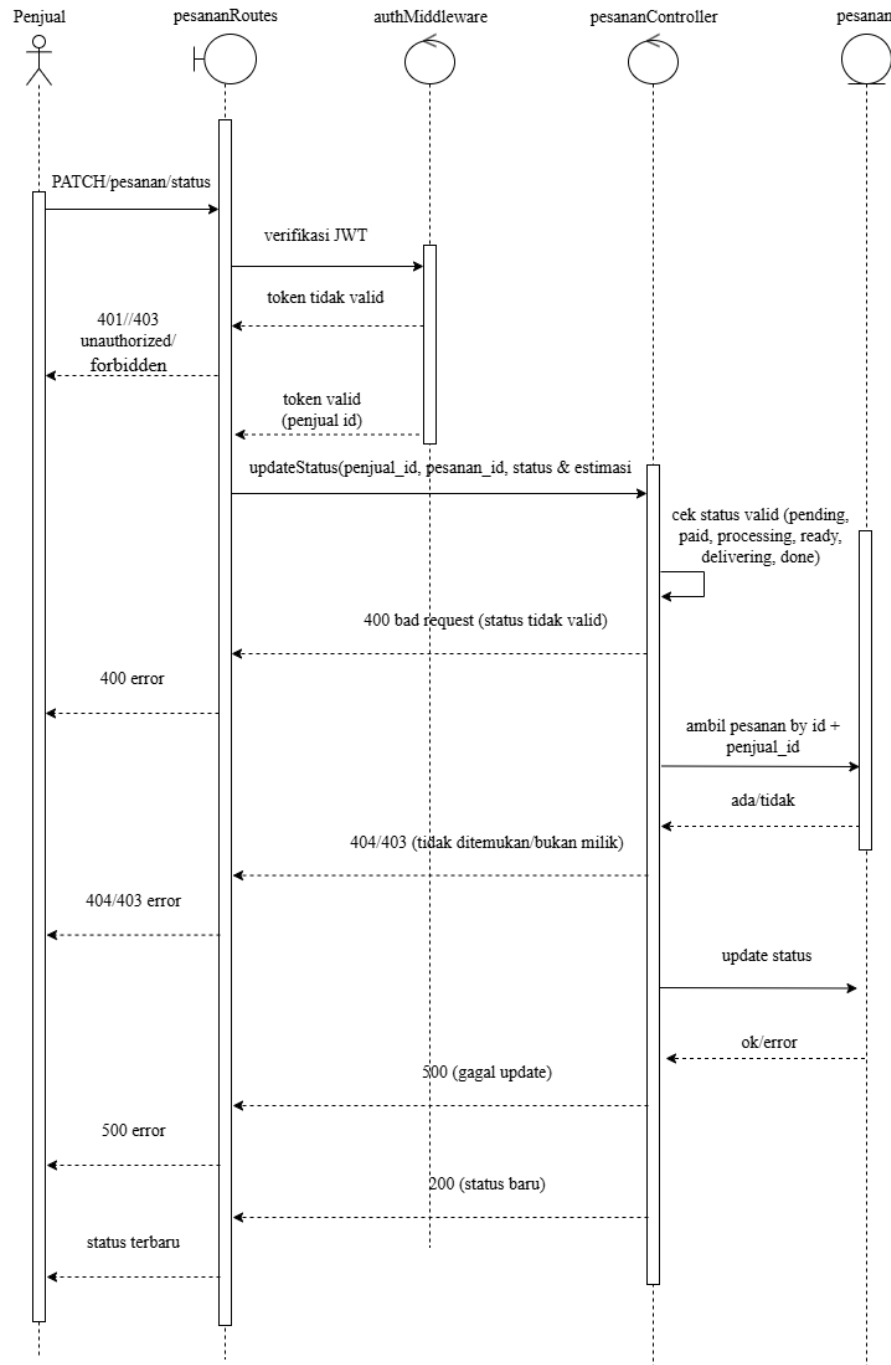
Penjual mengirim request GET/pesanan-masuk. Sistem memverifikasi JWT, lalu mengambil daftar pesanan aktif milik penjual dari *database*. Jika data berhasil diambil, sistem menambahkan informasi nomor antrean, status, dan tanggal sebelum mengirim response berisi daftar pesanan.

Tabel 4.73 Diagram Sekuens Uc-12b

**Deskripsi**

Penjual meminta detail pesanan tertentu dengan GET/pesanan-masuk/:id. Sistem memverifikasi JWT dan memastikan pesanan memang milik penjual. Jika valid, sistem mengambil header pesanan (status, pembayaran, estimasi) dan detail item. Hasil lengkap dikembalikan ke penjual, sedangkan jika tidak ditemukan maka sistem mengembalikan error 404/403.

Tabel 4.74 Diagram Sekuens Uc-13

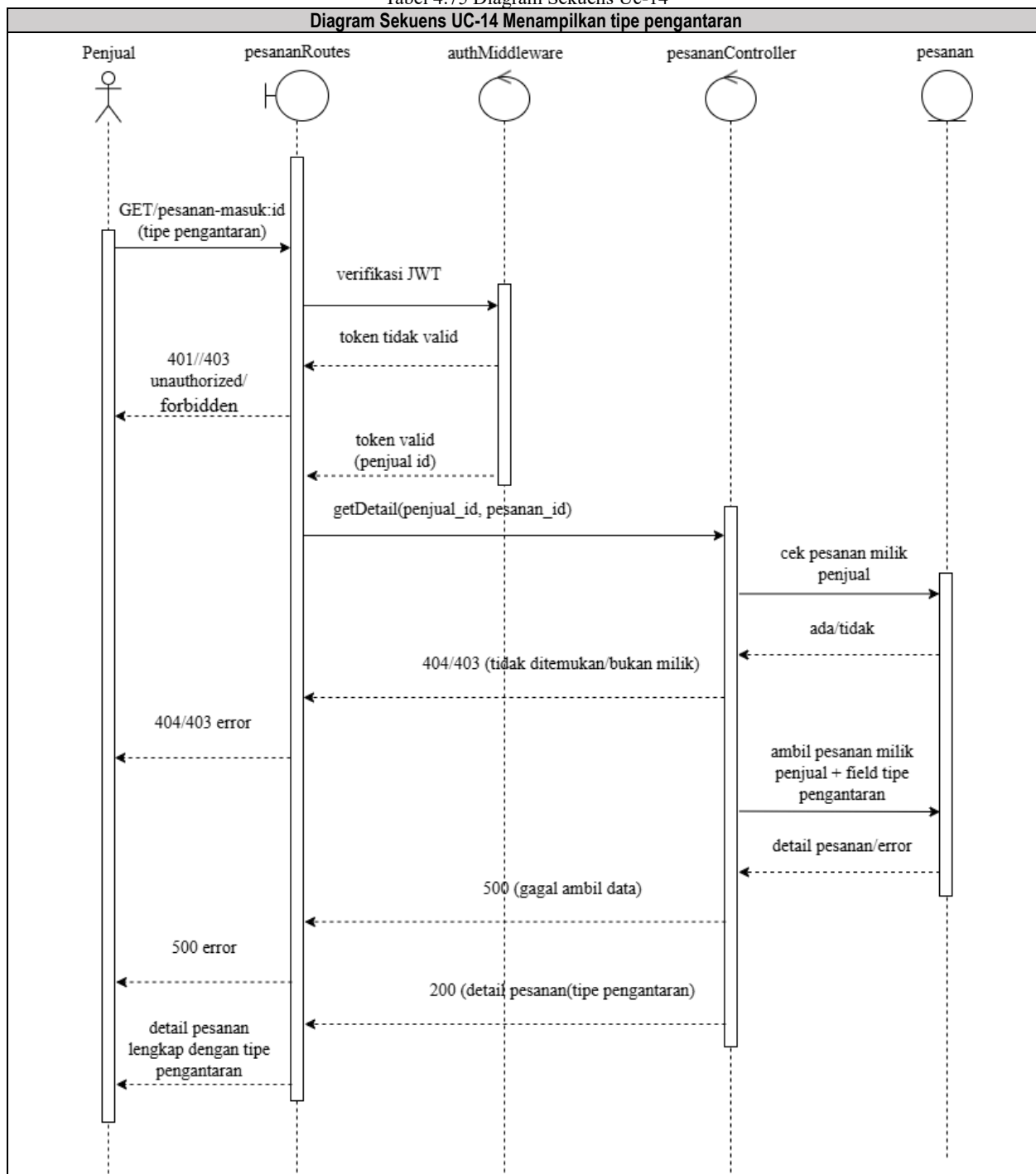
Diagram Sekuens UC-13 Mengubah status pesanan

Gambar 4.49 Diagram Sekuens Uc-13

Deskripsi

Penjual mengirim request PATCH/pesanan/status. Sistem memverifikasi JWT, lalu mengecek status yang diminta dan memastikan pesanan adalah milik penjual. Jika valid, status pesanan diperbarui sesuai input (pending, paid, processing, ready, delivering, done). Sistem kemudian mengembalikan response berisi status terbaru.

Tabel 4.75 Diagram Sekuens Uc-14



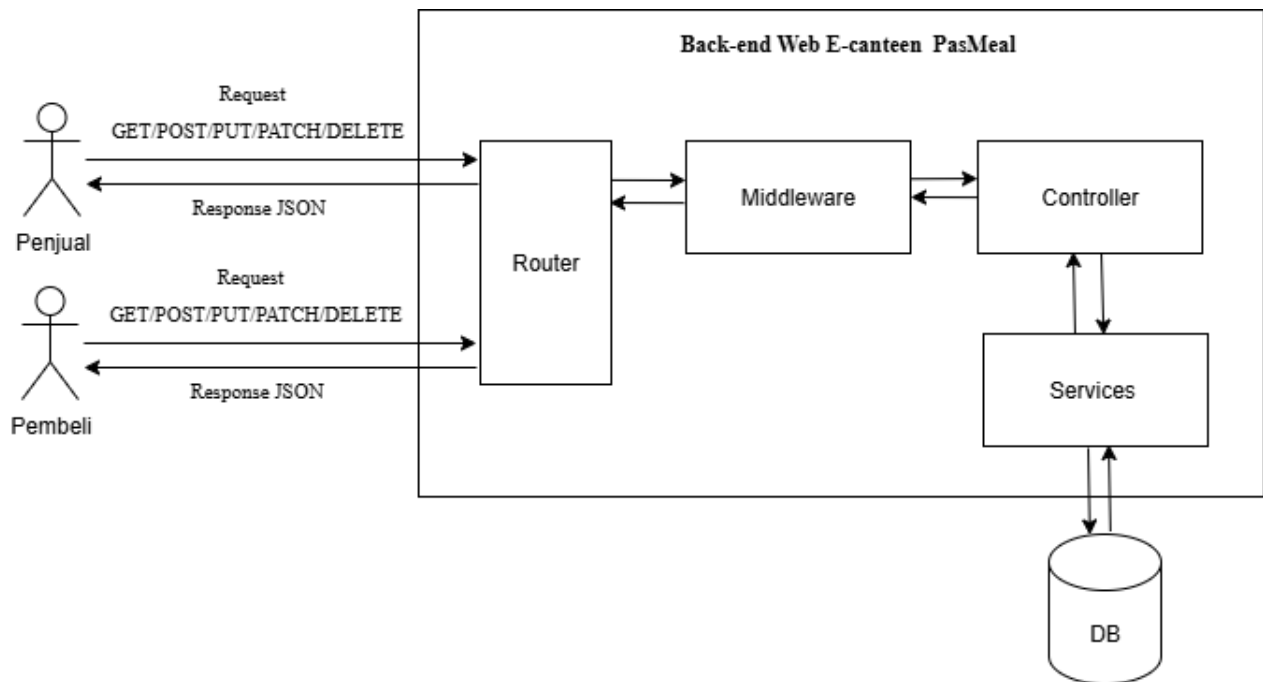
Gambar 4.50 Diagram Sekuens Uc-14

Deskripsi

Penjual melakukan request untuk melihat detail pesanan beserta tipe pengantarannya. Sistem memverifikasi JWT, lalu mengambil data pesanan milik penjual termasuk field *tipe pengantaran*. Jika valid, response berisi detail pesanan dan informasi pengantaran dikirim; jika tidak, sistem mengembalikan error.

4.2.4 Perancangan Arsitektur Aplikasi

Perancangan arsitektur digunakan untuk menggambarkan komunikasi antar komponen-komponen program yang diperlukan untuk membangun aplikasi atau perangkat lunak. Berikut adalah rancangan arsitektur yang digunakan untuk sistem ini.



Gambar 4.51 Arsitektur Aplikasi

Berikut merupakan penjelasan dari rancangan arsitektur sistem yang akan digunakan pada Web *E-canteen* PasMeal.

1. Aktor (Penjual & Pembeli)

Penjual dan Pembeli berinteraksi dengan sistem melalui endpoint API dengan menggunakan berbagai HTTP method seperti GET, POST, PUT, PATCH, dan DELETE. Setiap request akan menghasilkan response dalam format JSON.

2. Router

Request dari aktor pertama kali diterima oleh *Router*. *Router* berfungsi sebagai pintu masuk utama yang menentukan ke mana request tersebut harus diarahkan sesuai dengan path URL dan method yang digunakan. *Router* juga menjadi penghubung antara aktor dengan lapisan *middleware* dan controller.

3. Middleware

Setelah melewati *Router*, request akan diproses oleh *Middleware*. Lapisan ini berperan untuk menangani fungsi-fungsi seperti: Autentikasi (misalnya validasi *token* JWT), Otorisasi (memeriksa role pengguna, apakah penjual atau pembeli), Validasi data input. Jika validasi pada *middleware* gagal, sistem akan langsung mengembalikan response error kepada pengguna. Jika berhasil, request diteruskan

ke Controller.

4. Controller

Controller bertugas sebagai penghubung utama antara *middleware* dan service. Setelah request dinyatakan valid, controller akan memanggil fungsi atau method pada Service yang sesuai untuk mengeksekusi logika bisnis.

5. Services

Service merupakan tempat di mana logika bisnis dijalankan. Contohnya seperti pengolahan transaksi, perhitungan data, atau proses manajemen menu. Service juga bertanggung jawab untuk berinteraksi dengan *database* melalui query.

6. Database (DB)

Service akan mengirimkan permintaan ke *Database* untuk membaca, menambahkan, memperbarui, atau menghapus data. *Database* menyimpan seluruh informasi penting, seperti data pengguna (penjual dan pembeli), data menu, pesanan, serta transaksi.

7. Alur Kembali (Response)

Hasil operasi dari *database* akan dikembalikan ke Service, lalu diteruskan ke Controller. Controller kemudian membungkus hasil tersebut dalam format JSON yang konsisten, untuk akhirnya dikirimkan kembali ke aktor (Penjual atau Pembeli).

BAB 5

SCRUM PROSES DAN DELIVERY

Bab ini akan menjelaskan secara detail proses implementasi dan pengembangan perangkat lunak *backend* PasMeal *e-canteen* dengan menerapkan kerangka kerja Scrum. Pembahasan meliputi seluruh tahapan dalam Scrum, mulai dari penyusunan Product Backlog, perencanaan serta pelaksanaan Sprint, hingga tahap Delivery produk fungsional yang dihasilkan.

5.1 Scrum Proses

Tahapan ini menjelaskan proses pengembangan perangkat lunak dengan menggunakan kerangka kerja Scrum. Sebelum membahas detail aktivitas dalam siklus Scrum, terlebih dahulu dipaparkan lingkungan teknis yang digunakan, mencakup perangkat lunak pendukung, struktur direktori proyek, serta konfigurasi aplikasi. Selanjutnya, pembahasan diarahkan pada rangkaian kegiatan Scrum yang berlangsung secara iteratif dalam pengembangan produk. Setiap aktivitas tersebut akan diuraikan lebih lanjut pada subbab berikutnya.

1. Kakas Perangkat Lunak

Kakas perangkat lunak adalah seperangkat alat yang digunakan dalam proses pengembangan aplikasi. Berikut merupakan kakas yang dipakai untuk membangun aplikasi PasMeal *e-canteen*.

Tabel 5.1 Kakas Perangkat Lunak

No	Nama	Dekripsi	Kegunaan
1	Node.js	Runtime JavaScript yang berjalan di atas mesin V8 milik Chrome.	Digunakan sebagai platform utama untuk membangun dan menjalankan sisi <i>server (backend)</i> aplikasi.
2	Visual Studio Code	Editor kode lintas platform buatan Microsoft.	Berfungsi sebagai lingkungan utama untuk menulis, mengelola, serta debugging kode sumber aplikasi.
3	Supabase	Platform <i>backend-as-a-service</i> berbasis PostgreSQL open-source.	Digunakan sebagai <i>database</i> utama untuk menyimpan seluruh data aplikasi.
4	Postman	Platform untuk pengembangan dan pengujian API.	Digunakan untuk melakukan pengujian manual terhadap setiap endpoint API yang dibuat.
5	Git	Sistem kontrol versi terdistribusi untuk melacak perubahan pada kode sumber.	Berfungsi melacak perubahan kode sumber dan memfasilitasi manajemen versi secara lokal.
6	GitHub	Platform hosting repositori Git berbasis web.	Digunakan sebagai repositori utama, memfasilitasi kolaborasi tim, serta pelacakan isu.
7	Midtrans	Layanan payment gateway untuk transaksi <i>online</i> di Indonesia.	Memproses pembayaran pelanggan secara aman dan mendukung berbagai metode pembayaran.
8	Render	Platform cloud hosting modern untuk aplikasi, <i>database</i> , dan layanan web.	Menjadi platform utama untuk deployment aplikasi dan penyediaan infrastruktur cloud.
9	Cloudinary	Layanan manajemen media berbasis cloud.	Digunakan untuk menyimpan, mengelola, dan mengoptimalkan file gambar menu aplikasi.

Dalam rangka mendukung serta mempermudah pengembangan aplikasi, dilakukan instalasi sejumlah *package* Node.js melalui NPM (*Node Package Manager*). NPM berfungsi sebagai manajer paket bawaan dari Node.js yang digunakan untuk mengelola serta menginstal berbagai dependensi proyek. Berikut adalah *package-package* Node.js yang digunakan pada pengembangan aplikasi ini.

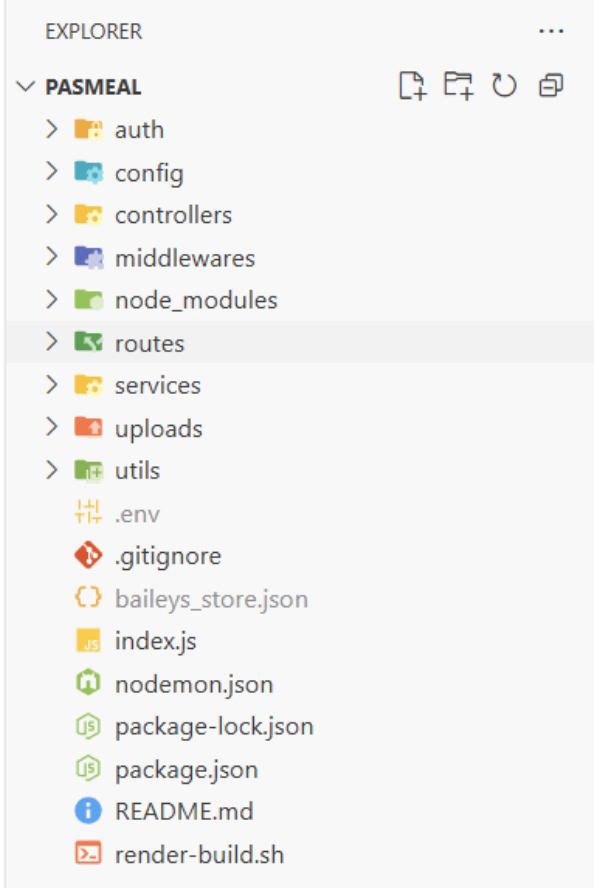
Tabel 5.2 Package yang Dibutuhkan

No	Nama	Link	Kegunaan
1	express	https://www.npmjs.com/package/express	Framework web untuk membangun server dan routing API.
2	@whiskeysockets/baileys	https://www.npmjs.com/package/@whiskeysockets/baileys	Library WhatsApp Web API untuk mengirim/terima pesan WA (notifikasi pesanan).
3	bcrypt	https://www.npmjs.com/package/bcrypt	Digunakan untuk hashing password (keamanan autentikasi).
4	cloudinary	https://www.npmjs.com/package/cloudinary	Menyimpan, mengelola, dan optimasi file gambar di cloud.
5	cors	https://www.npmjs.com/package/cors	Middleware untuk mengatur izin akses lintas domain pada API.
6	dotenv	https://www.npmjs.com/package/dotenv	Memuat konfigurasi environment dari file .env.
7	fs-extra	https://www.npmjs.com/package/fs-extra	Ekstensi fs Node.js dengan fitur tambahan untuk manipulasi file (copy, remove, move).
8	iconv-lite	https://www.npmjs.com/package/iconv-lite	Untuk konversi encoding teks (UTF-8, ASCII, dll).
9	jsonwebtoken	https://www.npmjs.com/package/jsonwebtoken	Membuat dan memverifikasi token JWT untuk autentikasi user.
10	midtrans-client	https://www.npmjs.com/package/midtrans-client	SDK Midtrans untuk integrasi pembayaran online.
11	multer	https://www.npmjs.com/package/multer	Middleware upload file (gambar menu, foto kios, dll).
12	path-to-regexp	https://www.npmjs.com/package/path-to-regexp	Parser untuk route path di Express menjadi regular expression.
131	pg	https://www.npmjs.com/package/pg	Driver PostgreSQL untuk Node.js (digunakan karena Supabase berbasis PostgreSQL).
14	pg-protocol	https://www.npmjs.com/package/pg-protocol	Protokol komunikasi PostgreSQL (dipakai bersama pg).
15	pino	https://www.npmjs.com/package/pino	Logger cepat untuk mencatat aktivitas server.
16	postgres	https://www.npmjs.com/package/postgres	Client PostgreSQL modern (alternatif dari pg).
17	qrcode	https://www.npmjs.com/package/qrcode	Client PostgreSQL modern (alternatif dari pg).
18	qrcode-terminal	https://www.npmjs.com/package/qrcode-terminal	Menampilkan QR Code langsung di terminal (berguna untuk WhatsApp session).
19	serverless-http	https://www.npmjs.com/package/serverless-http	Membungkus Express agar bisa jalan di platform serverless (Render, Vercel, AWS Lambda).
20	nodemon	https://www.npmjs.com/package/nodemon	Restart otomatis server saat ada perubahan file (untuk development).

2. Struktur Folder

Struktur folder digunakan untuk mengatur berkas dan direktori agar proses pengembangan lebih terorganisir, mudah dipelihara, serta memisahkan fungsi masing-masing komponen sistem. Dengan adanya struktur folder yang jelas, tim pengembang dapat bekerja lebih efisien dan meminimalisasi kesalahan dalam implementasi.

Tabel 5.3 Struktur Folder

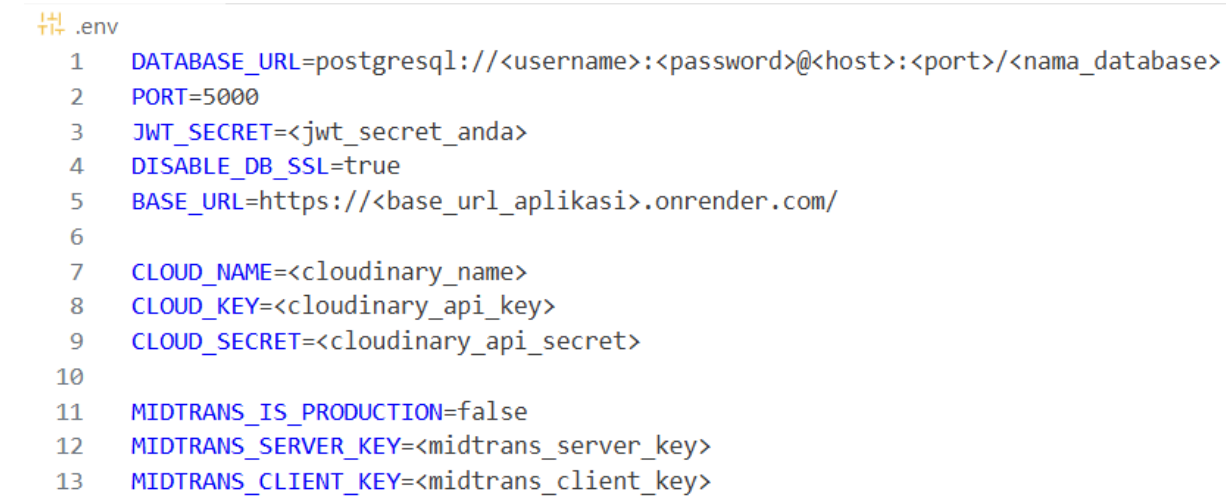
Struktur Folder	
	
	Gambar 5.1 Struktur Folder

Deskripsi
Gambar di atas merupakan struktur direktori dari aplikasi <i>backend</i> PasMeal <i>e-canteen</i>. Pada Node.js, struktur direktori ini dibagi menjadi beberapa bagian penting: 1. Folder Auth Berisi file atau modul yang mengatur autentikasi pengguna, misalnya proses login, register, validasi <i>token</i>, serta pengelolaan hak akses. 2. Folder Config Menyimpan file konfigurasi, seperti pengaturan koneksi <i>database</i>, API key layanan pihak ketiga, maupun setting environment yang dibutuhkan aplikasi. 3. Folder Controllers Berisi controller yang bertanggung jawab menangani request dari <i>client</i>, memanggil service yang sesuai, serta mengembalikan response ke <i>client</i>. 4. Folder Middlewares Berisi fungsi-fungsi <i>middleware</i> yang berjalan di antara request dan response. Misalnya <i>middleware</i> autentikasi JWT, validasi input, atau logging. 5. Folder node_modules Direktori yang dibuat otomatis oleh NPM, berisi seluruh <i>package</i> atau dependency pihak ketiga yang digunakan pada proyek. 6. Folder Routes Menyimpan definisi endpoint atau jalur API. Setiap route akan memetakan URL tertentu ke fungsi pada controller. 7. Folder services Berisi logika bisnis inti dari aplikasi. Service biasanya dipanggil oleh controller untuk memproses data atau berinteraksi dengan <i>database</i>. 8. Folder uploads Digunakan untuk menyimpan file sementara yang diunggah oleh user.

9. Folder <code>utils</code>	Berisi fungsi-fungsi utilitas (helper) yang dapat digunakan kembali di seluruh aplikasi. Contohnya format tanggal, pembuatan <i>token</i> , atau pengiriman notifikasi.
10. File <code>.env</code>	File yang berisi variabel environment seperti konfigurasi <i>database</i> , <i>secret key</i> , <i>API key</i> , dan data sensitif lainnya.
11. File <code>.gitignore</code>	Berisi daftar file atau folder yang tidak dilacak oleh Git, misalnya <code>node_modules</code> dan <code>.env</code> .
12. File <code>baileys_store.json</code>	File penyimpanan sesi untuk <i>library</i> WhatsApp Baileys, agar koneksi WhatsApp API bisa dipertahankan tanpa login ulang.
13. File <code>index.js</code>	Titik masuk (entry point) utama aplikasi. Di sini <i>server</i> Express diinisialisasi dan semua komponen seperti <i>route</i> , <i>middleware</i> , serta konfigurasi dihubungkan.
14. File <code>nodemon.json</code>	File konfigurasi untuk <i>nodemon</i> , agar saat development <i>server</i> otomatis restart ketika ada perubahan file.
15. File <code>package-lock.json</code>	Dicatat otomatis oleh NPM untuk mengunci versi dependensi proyek, memastikan instalasi konsisten di semua environment.
16. File <code>package.json</code>	Metadata proyek: mencatat dependensi, script, nama proyek, serta konfigurasi dasar Node.js.
17. File <code>README.md</code>	Dokumentasi utama proyek yang berisi pengantar, cara instalasi, cara menjalankan aplikasi, dan catatan penting lainnya.
18. File <code>render-build.sh</code>	Script build khusus untuk deployment di platform Render, berisi instruksi otomatis sebelum aplikasi dijalankan di <i>server</i> .

3. Konfigurasi

Konfigurasi pada *backend* PasMeal *e-canteen* berfungsi untuk menyimpan data *environment* serta memuat pengaturan bagi beberapa modul Node.js yang digunakan dalam aplikasi. Berikut adalah konfigurasi yang telah diterapkan.

Tabel 5.4 Konfigurasi Environment	
Konfigurasi Environment (.env)	
	
Gambar 5.2 Konfigurasi .env	
Deskripsi	

Gambar di atas merupakan konfigurasi environment untuk *backend* PasMeal *e-canteen*. Terdapat beberapa pengaturan konfigurasi antara lain:

1. `DATABASE_URL` : URL koneksi ke *database* Supabase berbasis PostgreSQL yang digunakan oleh aplikasi.
2. `PORT` : Port tempat *server* aplikasi berjalan (default 5000).
3. `JWT_SECRET` : Kunci rahasia yang digunakan untuk membuat dan memverifikasi *token* autentikasi (JWT).
4. `DISABLE_DB_SSL` : Pengaturan untuk mengaktifkan atau menonaktifkan penggunaan SSL pada koneksi *database*.
5. `BASE_URL` : Alamat dasar (base URL) aplikasi yang telah dideploy di Render.
6. `CLOUD_NAME` : Nama akun Cloudinary yang digunakan untuk menyimpan file gambar.
7. `CLOUD_KEY` : API Key dari Cloudinary untuk autentikasi ke layanan cloud storage.
8. `CLOUD_SECRET` : API Secret dari Cloudinary yang digunakan bersama API Key untuk mengelola file di cloud.
9. `MIDTRANS_IS_PRODUCTION` : Menentukan mode penggunaan Midtrans (false = sandbox/development, true = production).
10. `MIDTRANS_SERVER_KEY` : *Server* key dari Midtrans untuk memproses transaksi di sisi *backend* secara aman.
11. `MIDTRANS_CLIENT_KEY` : *Client* key dari Midtrans yang digunakan di sisi *frontend* untuk melakukan transaksi pembayaran.

Nilai variabel pada file `.env` bersifat sensitif sehingga tidak ditampilkan secara penuh di dalam laporan ini. Pada implementasi asli, nilai variabel diisi sesuai kredensial resmi yang digunakan pada sistem.

5.1.1 Product backlog

Dalam tahapan pembuatan Product Backlog, pemilihan fitur didasarkan pada prioritas aplikasi PasMeal sebagai solusi *e-canteen* yang telah teridentifikasi dari Use Case hasil analisis kebutuhan perangkat lunak. Setiap backlog mendeskripsikan fungsionalitas utama yang harus diimplementasikan untuk mendukung proses pemesanan, pembayaran, serta pengelolaan kios secara digital. Detail mengenai setiap fitur dapat dilihat pada tabel di bawah ini.

Tabel 5.5 Product Backlog

Kode	Use Case	Nama Backlog
PAS-BE.01	UC-01	Menampilkan daftar kios
PAS-BE.02	UC-02	Menampilkan daftar menu per kios
PAS-BE.03	UC-03	Menampilkan detail menu
PAS-BE.04	UC-04	Mencari menu
PAS-BE.05	UC-05	Mengelola keranjang
PAS-BE.06	UC-06	Membuat pesanan dari keranjang
PAS-BE.07	UC-07	Menyediakan payment gateway
PAS-BE.08	UC-08	Menampilkan status dan estimasi waktu pesanan
PAS-BE.09	UC-09	Mengirim notifikasi pesanan selesai
PAS-BE.10	UC-10	Mengelola menu
PAS-BE.11	UC-11	Mengirim notifikasi pesanan masuk
PAS-BE.12	UC-12	Menampilkan daftar pesanan dan detail pesanan
PAS-BE.13	UC-13	Mengubah status pesanan
PAS-BE.14	UC-14	Menampilkan tipe pengantaran
PAS-BE.15		Autentikasi
PAS-BE.16		Beranda

5.1.2 Sprint ke-1

Sprint pertama merupakan iterasi awal dalam siklus pengembangan aplikasi PasMeal sebagai solusi *e-canteen*. Pada tahap ini, seluruh aktivitas Scrum dilakukan untuk pertama kalinya. Fokus utama dari sprint ini adalah membangun komponen dasar yang menjadi fondasi utama sistem, seperti autentikasi penjual, pengelolaan menu, tampilan beranda, serta fitur dasar untuk menampilkan daftar kios dan daftar menu. Sprint Ke-1 dilaksanakan pada tanggal 1 Agustus 2025 – 7 Agustus 2025.

5.1.2.1 Sprint Planning Ke-1

Pada sesi sprint planning pertama, tim menyusun rencana pekerjaan yang berfokus pada pengembangan fitur-fitur inti agar sistem sudah dapat digunakan. Item backlog yang dipilih mencakup autentikasi, pengelolaan menu, beranda, menampilkan daftar kios, menampilkan daftar menu per kios, dan fitur pencarian menu.

Tabel 5.6 Sprint Planning Ke-1

Kode	Nama backlog	Estimasi Waktu	Prioritas
PAS-BE.16	Autentikasi	7 Hari	Tinggi
PAS-BE.10	Mengelola menu		Tinggi
PAS-BE.17	Beranda		Tinggi
PAS-BE.01	Menampilkan daftar kios		Tinggi
PAS-BE.02	Menampilkan daftar menu per kios		Tinggi
PAS-BE.04	Mencari menu		Tinggi

5.1.2.2 Sprint Backlog Ke-1

Pada sprint pertama, implementasi diarahkan pada fungsi-fungsi dasar yang menjadi komponen utama aplikasi. Rincian kebutuhan teknis setiap backlog ditampilkan pada tabel berikut:

Tabel 5.7 Sprint Backlog Ke-1

Kode	Nama backlog	Kebutuhan	Waktu Terpakai
PAS-BE.16	Autentikasi	Membuat fungsi Register Penjual	7 Hari
		Membuat fungsi Registrasi Kios	
		Membuat fungsi Verifikasi OTP	
		Membuat fungsi Resend OTP	
		Membuat fungsi Login Penjual	
		Membuat fungsi Forgot / Reset Password	
		Membuat fungsi Logout	
		Membuat fungsi Update Profil Penjual	
PAS-BE.10	Mengelola menu	Membuat fungsi Tambah Menu	
		Membuat fungsi Update Menu	
		Membuat fungsi Hapus Menu	
		Membuat fungsi menampilkan semua menu milik penjual	
PAS-BE.17	Beranda	Membuat fungsi menampilkan 5 menu terbaru	
		Membuat fungsi search berdasarkan keyword(menu/kios)	
		Membuat fungsi menampilkan 8 kios terbaru untuk ditampilkan di halaman beranda	
PAS-BE.01	Menampilkan daftar kios	Membuat fungsi untuk menampilkan semua kios (urut berdasarkan terbaru).	
PAS-BE.02	Menampilkan daftar menu per kios	Membuat fungsi menampilkan semua menu berdasarkan kios_id.	
PAS-BE.04	Mencari menu	Membuat fungsi cari menu berdasarkan keyword (nama menu) di seluruh kios.	
		Membuat fungsi cari menu berdasarkan keyword di kios tertentu.	

5.1.2.3 Increment Ke-1

Pada akhir Sprint Ke-1, sebuah increment fungsional berhasil dihasilkan yang menjadi fondasi utama dari sistem PasMeal. Increment ini mencakup fungsionalitas dasar pada sisi *backend*, yaitu autentikasi penjual, pengelolaan menu, tampilan beranda, serta fitur untuk menampilkan daftar kios, daftar menu per kios, dan pencarian menu. Hasil dari increment ini dapat diamati melalui struktur direktori proyek yang terbentuk, potongan kode yang mengimplementasikan logika bisnis, serta integrasi awal dengan basis data. Dengan increment ini, sistem sudah mulai dapat digunakan untuk proses dasar seperti login penjual, menambahkan menu, dan menampilkan kios/menu kepada pengguna.

1. Potongan Kode Autentikasi

Potongan kode berikut merepresentasikan fungsionalitas untuk mendaftarkan penjual baru. Fungsi *register* menerima data dari form registrasi (nama, email, nomor WhatsApp, *password*, konfirmasi *password*), melakukan validasi, menyimpan data ke *database*, lalu mengembalikan respon berupa ID penjual dan *token* JWT.

Tabel 5.8 Potongan Kode Autentikasi

Potongan Kode Autentikasi
<pre>const register = async (req, res) => { try { const { penjual_id, token } = await AuthService.register(req.body); return res.status(201).json({ message: 'Penjual berhasil didaftarkan', penjual_id, token }); } catch (err) { console.error(err); return res.status(err.status 500).json({ message: err.message 'Terjadi kesalahan server' }); } };</pre>
Deskripsi
Kode ini menunjukkan bagaimana fungsi register memproses pendaftaran penjual baru dengan melakukan validasi <i>input</i> , enkripsi <i>password</i> , penyimpanan ke <i>database</i> , dan menghasilkan <i>token</i> JWT sebagai hasil autentikasi awal.

2. Potongan Kode Mengelola Menu

Potongan kode dari menuController ini dipilih untuk mewakili proses menambahkan menu baru oleh penjual. Implementasi berikut menggunakan *multer* untuk *upload* gambar langsung ke *server*

Tabel 5.9 Potongan Kode Mengelola Menu

Potongan Kode Mengelola Menu
<pre>// Tambah menu (penjual) const addMenu = async (req, res) => { try { const { status, body } = await addMenuService(req); return res.status(status).json(body); } catch (error) { console.error(error); return res.status(500).json({ message: 'Terjadi kesalahan server', error: error.message }); } };</pre>
Deskripsi
Kode ini menunjukkan bagaimana fungsi addMenu digunakan untuk menambahkan menu baru. Data menu divalidasi, foto diunggah menggunakan multer dan disimpan di <i>server</i> lokal, lalu seluruh informasi menu disimpan ke <i>database</i> .

3. Potongan Kode Beranda

Potongan kode dari menuController ini dipilih untuk mewakili proses menampilkan menu terbaru di halaman beranda. Method `getNewMenus` menunjukkan bagaimana sistem mengambil data lima menu terbaru dari *database* berdasarkan waktu pembuatan untuk kemudian ditampilkan kepada pengguna di beranda.

Tabel 5.10 Potongan Kode Beranda

Potongan Kode Beranda
<pre>// Ambil 5 menu terbaru (pembeli) const getNewMenus = async (req, res) => { try { const { status, body } = await getNewMenusService(); return res.status(status).json(body); } catch (err) { return res.status(500).json({ error: err.message }); } };</pre>
Deskripsi
Kode ini menampilkan lima menu terbaru dari <i>database</i> untuk ditampilkan di halaman beranda. Data yang dikembalikan mencakup informasi penting seperti nama, deskripsi, harga, estimasi waktu, status ketersediaan, dan foto menu.

4. Potongan Kode Menampilkan Daftar Kios

Potongan kode dari kiosController ini dipilih untuk mewakili proses mengambil daftar semua kios yang ada di *database*.

Tabel 5.11 Potongan Kode Menampilkan Daftar Kios

Potongan Kode Menampilkan Daftar Kios
<pre>// Ambil semua kios (pembeli) const getAllKios = async (req, res) => { try { const { status, body } = await getAllKiosService(); return res.status(status).json(body); } catch (err) { return res.status(500).json({ error: err.message }); } };</pre>
Deskripsi
Kode ini menunjukkan bagaimana fungsi <code>getAllKios</code> digunakan untuk mengambil seluruh data kios dari tabel kios. Data diurutkan berdasarkan waktu pembuatan (<code>created_at</code> DESC) sehingga kios terbaru tampil lebih dahulu, lalu hasil query dikembalikan dalam bentuk JSON sebagai <code>respons</code> .

5. Potongan Kode Menampilkan Daftar Menu Per Kios

Potongan kode dari `menuController` ini dipilih untuk mewakili proses mengambil daftar menu berdasarkan kios tertentu.

Tabel 5.12 Potongan Kode Menampilkan Daftar Menu Per Kios

Potongan Kode Menampilkan Daftar Menu Per Kios
<pre>const getMenuByKios = async (req, res) => { try { const { status, body } = await getMenuByKiosService(req); return res.status(status).json(body); } catch (err) { return res.status(500).json({ error: err.message }); } };</pre>
Deskripsi
Kode ini menunjukkan bagaimana fungsi <code>getMenuByKios</code> digunakan untuk mengambil semua menu dari suatu kios. Parameter <code>id</code> kios diambil dari URL, lalu query dijalankan untuk mencari menu dengan <code>kios_id</code> tersebut. Data menu diurutkan berdasarkan waktu pembuatan (<code>created_at</code> DESC) sehingga menu terbaru muncul lebih dahulu, dan hasilnya dikembalikan dalam format JSON.

5.1.2.4 Sprint Review Ke-1

Sesi Sprint Review Ke-1 dilaksanakan pada akhir sprint secara internal oleh tim `PasMeal`. Pada sesi ini, setiap backlog yang telah dikerjakan ditunjukkan dan diuji untuk memastikan fungsionalitas berjalan sesuai harapan.

Tabel 5.13 Sprint Review Ke-1

Kode	Nama backlog	Status	Hasil dan Umpan Balik
PAS-BE.16	Autentikasi	Tidak Selesai	Fungsionalitas autentikasi berhasil berjalan, namun fitur update profil penjual belum berfungsi serta fungsi logout belum ada.
PAS-BE.10	Mengelola menu	Tidak Selesai	Fungsionalitas CRUD menu sudah diterapkan, namun masih ada kendala pada tampilan foto.
PAS-BE.17	Beranda	Tidak Selesai	Fitur beranda berhasil ditampilkan, tetapi pencarian global belum tersedia.
PAS-BE.01	Menampilkan daftar kios	Tidak Selesai	Fungsionalitas menampilkan daftar kios berhasil berjalan, namun terdapat bug pada foto kios yang terkadang hilang.
PAS-BE.02	Menampilkan daftar menu per kios	Tidak Selesai	Daftar menu per kios berhasil ditampilkan, namun terdapat bug pada foto menu yang terkadang hilang.
PAS-BE.04	Mencari menu	Selesai	Fitur pencarian menu berdasarkan nama berhasil diimplementasikan dan berjalan sesuai harapan.

Sebagian besar fungsionalitas inti seperti autentikasi, pengelolaan menu, dan pencarian sudah dapat dijalankan, namun masih terdapat kekurangan pada fitur pendukung seperti update profil, logout, serta kendala pada tampilan foto. Selain itu, beberapa backlog seperti beranda, daftar kios, dan daftar menu per kios telah ditampilkan tetapi masih terdapat bug dan pencarian global belum tersedia. Secara keseluruhan, progres sprint sudah menunjukkan hasil, namun masih diperlukan penyempurnaan dan perbaikan bug agar seluruh backlog dapat berjalan sesuai harapan.

5.1.3 Sprint ke-2

Sprint kedua merupakan kelanjutan dari iterasi pengembangan aplikasi PasMeal. Pada tahap ini, fokus utama diarahkan pada penyempurnaan fitur autentikasi, pengelolaan keranjang, pembuatan pesanan dari keranjang, integrasi payment gateway, serta pengiriman notifikasi pesanan masuk. Fitur-fitur ini menjadi pondasi penting agar sistem mulai dapat digunakan secara menyeluruh oleh pengguna untuk melakukan pemesanan. Sprint Ke-2 dilaksanakan pada tanggal 8 Agustus 2025 – 14 Agustus 2025.

5.1.3.1 Sprint Planning Ke-2

Pada sesi sprint planning kedua, tim menyusun rencana pekerjaan dengan tujuan memperluas fungsionalitas dasar aplikasi. Item backlog yang dipilih mencakup:

Tabel 5.14 Sprint planning ke-2

Kode	Nama backlog	Estimasi Waktu	Prioritas
PAS-BE.16	Autentikasi	7 Hari	Tinggi
PAS-BE.03	Menampilkan detail menu		Tinggi
PAS-BE.05	Mengelola keranjang		Tinggi
PAS-BE.06	Membuat pesanan dari keranjang		Tinggi
PAS-BE.07	Menyediakan payment gateway		Tinggi
PAS-BE.11	Mengirim notifikasi pesanan masuk		Tinggi

5.1.3.2 Sprint Backlog Ke-2

Pada Sprint Ke-2, implementasi diarahkan pada pengembangan fitur transaksi, mulai dari pemilihan menu hingga notifikasi pesanan. Rincian kebutuhan teknis setiap backlog ditampilkan pada tabel berikut:

Tabel 5.15 Sprint Backlog Ke-2

Kode	Nama backlog	Kebutuhan	Waktu Terpakai
PAS-BE.16	Autentikasi	Membuat fungsi Update Profil Penjual	7 hari
		Membuat fungsi Logout	
		Memperbaiki register kios	
PAS-BE.03	Menampilkan detail menu	Membuat fungsi untuk menampilkan detail menu berdasarkan id menu	
PAS-BE.05	Mengelola keranjang	Membuat fungsi tambah item ke keranjang	
		Membuat fungsi untuk menampilkan isi keranjang milik pembeli	
		Membuat fungsi untuk ubah jumlah item keranjang	
		Membuat fungsi untuk hapus item dari keranjang	
PAS-BE.06	Membuat pesanan dari keranjang	Membuat fungsi untuk membuat pesanan dari keranjang	
PAS-BE.07	Menyediakan payment gateway	Membuat fungsi untuk membuat transaksi pembayaran	

Kode	Nama backlog	Kebutuhan	Waktu Terpakai
		Membuat fungsi untuk memproses notifikasi pembayaran	
PAS-BE.11	Mengirim notifikasi pesanan masuk	Membuat fungsi untuk mengirim pesan WhatsApp ke nomor tertentu	
		Membuat fungsi untuk mengirim notifikasi pesanan baru ke penjual via WhatsApp.	

5.1.3.3 Increment Ke-2

Pada akhir Sprint Ke-2, sebuah *increment* fungsional berhasil dihasilkan untuk memperluas fitur dasar sistem PasMeal. Increment ini mencakup pengelolaan keranjang, pembuatan pesanan dari keranjang, integrasi awal *payment gateway*, serta pengiriman notifikasi pesanan masuk. Selain itu, penyempurnaan autentikasi juga dilakukan dengan penambahan fitur *update* profil penjual dan *logout*. Increment Ke-2 membuat sistem semakin siap digunakan karena alur utama mulai dari memilih menu, menambahkannya ke keranjang, membuat pesanan, hingga memproses pembayaran dan mengirimkan notifikasi pesanan telah dapat dijalankan meskipun masih terdapat kendala teknis.

1. Potongan Kode Autentikasi

Potongan kode dari `authController` ini dipilih untuk mewakili proses *logout* penjual dengan cara memasukkan *token* JWT ke dalam tabel *blacklist*.

Tabel 5.16 Potongan Kode Autentikasi (Logout)

Potongan Kode Autentikasi (Logout)
<pre>const logout = async (req, res) => { try { await AuthService.logout({ authorization: req.headers.authorization }); return res.status(200).json({ message: 'Logout berhasil' }); } catch (err) { console.error('Gagal logout:', err); return res.status(err.status 500).json({ message: err.message 'Terjadi kesalahan server' }); } };</pre>
Deskripsi
Kode ini menunjukkan bagaimana fungsi <code>logout</code> digunakan untuk mengakhiri sesi pengguna. <i>Token</i> JWT yang dikirim melalui header <code>Authorization</code> akan diambil dan didekode untuk mendapatkan waktu kadaluarsa (<code>exp</code>). <i>Token</i> tersebut kemudian dimasukkan ke tabel <code>blacklisted_tokens</code> agar tidak bisa digunakan lagi untuk autentikasi di request berikutnya. Respons JSON diberikan untuk menandakan <code>logout</code> berhasil.

2. Potongan Kode Menampilkan Detail Menu

Potongan kode dari `pesananController` ini dipilih untuk mewakili proses mengambil detail detail menu pembeli

Tabel 5.17 Potongan Kode menampilkan detail menu

Potongan Kode Menampilkan Detail Menu
<pre>const getMenuByIdForBuyer = async (req, res) => { try { const { status, body } = await getMenuByIdForBuyerService(req); return res.status(status).json(body); } catch (error) { if (error.status) return res.status(error.status).json({ message: error.message }); return res.status(500).json({ message: 'Gagal mengambil detail menu', error: error.message }); } };</pre>
Deskripsi
Kode ini menunjukkan bagaimana fungsi getMenuByIdForBuyer digunakan pembeli untuk melihat detail menu tertentu. Setelah memverifikasi guest_id, sistem mengambil data menu dari tabel menu.

3. Potongan Kode Mengelola Keranjang

Potongan kode dari keranjangController ini dipilih untuk mewakili proses menambahkan menu ke keranjang pembeli, termasuk update jumlah jika item sudah ada

Tabel 5.18 Potongan Kode mengelola keranjang

Potongan Kode Mengelola keranjang
<pre>const addToKeranjang = async (req, res) => { try { const { status, body } = await addToKeranjangService(req); return res.status(status).json(body); } catch (err) { // khusus 409 perlu kirim existing_kios_id bila ada if (err.status === 409 && err.existing_kios_id) { return res.status(409).json({ message: err.message, existing_kios_id: err.existing_kios_id }); } console.error('addToKeranjang error:', err); return res.status(err.status 500).json({ message: err.message 'Terjadi kesalahan server' }); } };</pre>
Deskripsi
Kode ini menunjukkan bagaimana fungsi addToKeranjang digunakan pembeli untuk menambahkan menu ke keranjang. Pertama, sistem memvalidasi guest_id dan menu_id, lalu mengambil informasi menu dari tabel menu. Jika item yang sama sudah ada di keranjang, jumlahnya diperbarui dan total harga disesuaikan. Jika belum ada, item baru ditambahkan ke tabel keranjang.

4. Potongan Kode Membuat Pesanan dari Keranjang

Potongan kode dari pesananController ini membuat pesanan dari isi keranjang dengan memvalidasi data wajib, membaca item keranjang, menghitung total harga saja, lalu menyimpan pesanan dan detailnya dalam satu transaksi; nilai total_estimasi belum dihitung sehingga diset NULL (atau 0 sesuai kebutuhan) dan setelah berhasil keranjang dikosongkan, penjual diberi notifikasi, serta respons 201 dikembalikan dengan status awal pending.

Tabel 5.19 Potongan Kode membuat pesanan dari keranjang

Potongan Kode Membuat pesanan dari keranjang
<pre>const buatPesanan = async (req, res) => { try { const { status, body } = await buatPesananService(req); return res.status(status).json(body); } catch (err) { if (err.status) return res.status(err.status).json({ message: err.message }); console.error(err); return res.status(500).json({ message: 'Terjadi kesalahan server' }); } };</pre>
Deskripsi
Kode ini membuat pesanan baru dengan menghitung total harga dari isi keranjang tetapi belum menghitung total_estimasi; seluruh proses dibungkus dalam transaksi untuk menjaga konsistensi, keranjang dibersihkan setelah pesanan dan detailnya tersimpan, penjual menerima notifikasi, dan respons berisi data pesanan berstatus pending.

5. Potongan Kode Menyediakan payment gateway

Potongan kode dari transaksiController ini dipilih untuk mewakili proses membuat transaksi pembayaran Midtrans berdasarkan pesanan pembeli.

Tabel 5.20 Potongan Kode payment gateway

Potongan Kode Menyediakan payment gateway
<pre>const createTransaction = async (req, res) => { try { const { status, body } = await createTransactionService(req); return res.status(status).json(body); } catch (err) { if (err.status) return res.status(err.status).json({ message: err.message }); // eslint-disable-next-line no-console console.error(err); return res.status(500).json({ message: "Gagal membuat transaksi" }); } };</pre>
Deskripsi
Kode ini menjelaskan bagaimana fungsi createTransaction dipakai untuk menghubungkan pesanan dengan sistem pembayaran Midtrans.

6. Potongan Kode Mengirim Notifikasi Pesanan Masuk

Potongan kode berikut digunakan untuk mengirimkan notifikasi WhatsApp ke penjual setiap kali ada pesanan baru yang masuk.

Tabel 5.21 Potongan Kode notifikasi pesanan masuk

Potongan Kode Mengirim notifikasi pesanan masuk
<pre>const notifyPenjual = async (kiosId, pesananId) => { return notifyPenjualService(kiosId, pesananId); };</pre>
Deskripsi
Fungsi notifyPenjual bertanggung jawab mengirimkan notifikasi ke nomor WhatsApp penjual ketika ada pesanan baru. Prosesnya dimulai dengan mengambil data penjual_id dari tabel kios, lalu mencari nomor handphone penjual dari tabel penjual. Setelah itu, sistem membentuk pesan berisi tautan menuju dashboard aplikasi (https://pas-meal.vercel.app/). Pesan ini dikirim menggunakan fungsi sendWaMessage. Jika berhasil, sistem akan menampilkan log di console untuk memastikan notifikasi telah terkirim.

5.1.3.4 Sprint Review Ke-2

Sesi Sprint Review Ke-2 dilaksanakan pada akhir sprint secara internal oleh tim PasMeal. Pada sesi ini, setiap backlog yang telah dikerjakan ditunjukkan dan diuji untuk memastikan fungsionalitas berjalan sesuai harapan.

Tabel 5.22 Sprint Review ke-2

Kode	Nama backlog	Status	Hasil dan Umpan Balik
PAS-BE.16	Autentikasi	Selesai	Fungsionalitas autentikasi berhasil diimplementasikan dan berjalan sesuai harapan.
PAS-BE.03	Menampilkan detail menu	Tidak Selesai	Fungsionalitas menampilkan detail menu sudah berjalan, namun terdapat bug pada foto menu yang terkadang hilang.
PAS-BE.05	Mengelola keranjang	Tidak Selesai	Fungsionalitas keranjang sudah berjalan, namun masih memungkinkan penambahan item dari kios berbeda; diperlukan pembatasan satu kios per keranjang.
PAS-BE.06	Membuat pesanan dari keranjang	Tidak Selesai	Fungsionalitas membuat pesanan dari keranjang sudah berjalan, namun perhitungan total estimasi belum sesuai.
PAS-BE.07	Menyediakan payment gateway	Tidak Selesai	Integrasi payment gateway di sisi <i>backend</i> sudah berjalan, namun sistem belum menerima notifikasi dari Midtrans ke <i>backend</i> .
PAS-BE.11	Mengirim notifikasi pesanan masuk	Tidak Selesai	Fungsionalitas pengiriman notifikasi pesanan masuk sudah berjalan, namun pesan WhatsApp masih terkirim lebih dari satu kali.

Hanya backlog autentikasi (PAS-BE.16) yang sepenuhnya selesai. Lima backlog lainnya masih terkendala, terutama pada bug keranjang, perhitungan estimasi, integrasi Midtrans, serta notifikasi ganda. Semua kendala ini akan dibawa ke Sprint berikutnya untuk diperbaiki.

5.1.4 Sprint ke-3

Sprint ketiga merupakan iterasi lanjutan dari pengembangan aplikasi PasMeal. Fokus utama diarahkan pada penyempurnaan fitur beranda dengan pencarian global, detail menu, pengelolaan keranjang, pembuatan pesanan, penambahan tipe pengantaran, serta menampilkan daftar pesanan berikut estimasi waktu penyelesaiannya. Sprint Ke-3 dilaksanakan pada tanggal 15 Agustus 2025 – 21 Agustus 2025.

5.1.4.1 Sprint Planning Ke-3

Pada sesi sprint planning ketiga, tim menetapkan item backlog yang diprioritaskan agar aplikasi semakin siap digunakan oleh pengguna akhir. Item backlog yang dipilih mencakup:

Tabel 5.23 Sprint Planning ke-3

Kode	Nama backlog	Estimasi Waktu	Prioritas
PAS-BE.17	Beranda	7 Hari	Tinggi
PAS-BE.05	Mengelola keranjang		Tinggi
PAS-BE.06	Membuat pesanan dari keranjang		Tinggi
PAS-BE.14	Menampilkan tipe pengantaran		Tinggi
PAS-BE.13	Mengubah Status Pesanan		
PAS-BE.12	Menampilkan daftar pesanan dan detail pesanan		Tinggi
PAS-BE.08	Menampilkan status dan estimasi waktu pesanan		Tinggi

5.1.4.2 Sprint Backlog Ke-3

Pada Sprint Ke-3, implementasi diarahkan pada penyempurnaan alur transaksi dan monitoring pesanan. Fokus pengerjaan meliputi pengembangan fitur pencarian global pada beranda, penyempurnaan detail menu, validasi keranjang agar tidak menampung item dari kios berbeda, perbaikan logika estimasi waktu pesanan, serta penambahan opsi tipe pengantaran. Selain itu, pada sprint ini juga ditambahkan fungsionalitas untuk menampilkan daftar pesanan berikut detailnya, sehingga pengguna dapat memantau status dan progres pesanan dengan lebih jelas. Rincian kebutuhan teknis setiap backlog ditampilkan pada tabel berikut:

Tabel 5.24 Sprint Backlog ke-3

Kode	Nama backlog	Kebutuhan	Waktu Terpakai
PAS-BE.17	Beranda	Membuat fungsi pencarian global (nama menu atau kios)	7 hari
PAS-BE.13	Mengubah Status Pesanan	Menyediakan fungsionalitas bagi penjual untuk mengubah status pesanan	
PAS-BE.05	Mengelola keranjang	Memperbaiki fungsi agar keranjang tidak dapat menambah item dari kios yang berbeda	
PAS-BE.06	Membuat pesanan dari keranjang	Memperbaiki fungsi agar menghitung total estimasi dengan benar	
PAS-BE.14	Menampilkan tipe pengantaran	Membuat fungsi untuk menampilkan detail pesanan masuk, termasuk informasi tipe pengantaran (Ambil Sendiri / alamat tujuan)	
PAS-BE.12	Menampilkan daftar pesanan dan detail pesanan	Membuat fungsi untuk menampilkan daftar pesanan masuk	
		Membuat fungsi untuk menampilkan detail pesanan masuk	
PAS-BE.08	Menampilkan status dan estimasi waktu pesanan	Membuat fungsi untuk menampilkan status dan estimasi waktu pesanan	

5.1.4.3 Increment Ke-3

Pada akhir Sprint Ke-3, sistem berhasil menghasilkan increment berupa fitur pencarian global di beranda, validasi keranjang agar tidak bisa menampung item dari kios berbeda, serta penyempurnaan alur pesanan dengan estimasi waktu dan tipe pengantaran. Selain itu, daftar pesanan dan detailnya sudah dapat ditampilkan meskipun masih ada kendala teknis pada bagian estimasi.

1. Potongan Kode Beranda

Potongan kode dari `searchController` ini dipilih untuk mewakili proses pencarian global pada aplikasi. Method `searchAll` melakukan pencarian berdasarkan kata kunci yang dimasukkan pengguna, baik pada nama kios maupun nama menu, lalu mengembalikan hasilnya dalam satu respons.

Tabel 5.25 Potongan Kode Beranda

Potongan Kode Beranda
<pre>const searchAll = async (req, res) => { try { const { status, body } = await searchAllService(req); return res.status(status).json(body); } catch (err) { if (err.status) return res.status(err.status).json({ message: err.message }); // eslint-disable-next-line no-console console.error('Gagal melakukan pencarian:', err); return res.status(500).json({ message: 'Terjadi kesalahan server' }); } };</pre>
Deskripsi
Kode ini memungkinkan pengguna mencari sekaligus pada tabel kios dan menu. Hasil pencarian mengembalikan daftar kios dan menu yang cocok dengan kata kunci dalam satu objek JSON.

2. Potongan Kode Mengubah Status Pesanan

Potongan kode dari pesananController ini dipilih untuk mewakili proses pembaruan status pesanan pada aplikasi. Method `updateStatusPesanan` memungkinkan penjual memperbarui status pesanan sesuai workflow yang berlaku, termasuk validasi hak akses penjual dan pengaturan estimasi waktu proses.

Tabel 5.26 Potongan Kode Mengubah status pesanan

Potongan Kode Mengubah status pesanan
<pre>const updateStatusPesanan = async (req, res) => { try { const { status, body } = await updateStatusPesananService(req); return res.status(status).json(body); } catch (err) { if (err.status) return res.status(err.status).json({ message: err.message }); console.error('updateStatusPesanan error:', err); return res.status(500).json({ message: 'Terjadi kesalahan server' }); } };</pre>
Deskripsi
Kode ini menangani pembaruan status pesanan oleh penjual. Pertama, sistem melakukan validasi agar status hanya sesuai daftar yang diizinkan. Jika status adalah <i>processing</i> , sistem mengambil data pesanan untuk menghitung estimasi waktu penyelesaian, mencatat waktu mulai, estimasi selesai, dan mereset flag keterlambatan. Jika status lain dipilih, sistem hanya memperbarui status sesuai input. Selain itu, ketika status berubah menjadi <i>done</i> , sistem akan mengirim notifikasi kepada pembeli bahwa pesanan telah selesai. Terakhir, sistem mengembalikan pesan sukses beserta data pesanan terbaru dalam format JSON.

3. Potongan Kode Mengelola Keranjang

Potongan kode dari keranjangController ini mewakili proses penambahan item ke keranjang berdasarkan identitas sementara pengguna (`guest_id`), termasuk validasi data wajib, pengecekan konsistensi kios, serta logika `update/insert` item.

Tabel 5.27 Potongan Kode Mengelola Keranjang

Potongan Kode Mengelola Keranjang
<pre>// Cek kalau ada item dari kios lain const existingCart = await pool.query('SELECT DISTINCT kios_id FROM keranjang WHERE guest_id = \$1',</pre>

Potongan Kode Mengelola Keranjang

```

[guest_id]
);
if (existingCart.rows.length > 0 && existingCart.rows[0].kios_id !== kios_id) {
  throw Object.assign(
    httpErr(409, 'Keranjang Anda masih ada dari kios lain. Hapus dulu untuk menambah
menu dari kios ini'),
    { existing_kios_id: existingCart.rows[0].kios_id }
  );
}
// Cek apakah item sudah ada
const existing = await pool.query(
  'SELECT * FROM keranjang WHERE guest_id = $1 AND menu_id = $2',
  [guest_id, menu_id]
);

let itemRow;
let isUpdate = false;

if (existing.rows.length > 0) {
  const newJumlah = existing.rows[0].jumlah + jumlah;
  const total_harga = harga * newJumlah;

  const updated = await pool.query(
    `UPDATE keranjang
      SET jumlah = $1, catatan = $2, total_harga = $3
      WHERE guest_id = $4 AND menu_id = $5
      RETURNING *`,
    [newJumlah, (catatan || existing.rows[0].catatan), total_harga, guest_id,
menu_id]
  );
  itemRow = updated.rows[0];
  isUpdate = true;
} else {
  const total_harga = harga * jumlah;

  const inserted = await pool.query(
    `INSERT INTO keranjang (guest_id, kios_id, menu_id, jumlah, catatan, total_harga)
      VALUES ($1, $2, $3, $4, $5, $6) RETURNING *`,
    [guest_id, kios_id, menu_id, jumlah, catatan, total_harga]
  );
  itemRow = inserted.rows[0];
}

const body = {
  message: isUpdate ? 'Jumlah diperbarui' : 'Ditambahkan ke keranjang',
  item: {
    ...itemRow,
    nama_menu,
    harga,
    foto_menu,
    subtotal: harga * itemRow.jumlah,
  }
};

// status harus 200 saat update, 201 saat insert (sesuai perilaku lama)
return { status: isUpdate ? 200 : 201, body };
}

```

Deskripsi

Kode ini memastikan keranjang tidak berisi item dari kios berbeda; jika berbeda, permintaan ditolak dengan status 409. Jika item yang sama sudah ada, jumlahnya diakumulasi dan total harga diperbarui; jika belum ada, item baru disisipkan. Respons mengembalikan status 201 saat insert atau 200 saat update, beserta detail item (termasuk nama_menu, harga, foto_menu, dan subtotal) dalam format JSON.

4. Potongan Kode Membuat Pesanan dari Keranjang

Potongan kode dari `pesananController` ini mewakili proses pembuatan pesanan dari keranjang: sistem memvalidasi data wajib, mengambil isi keranjang, menghitung total harga dan estimasi, membuat *record* pesanan beserta detailnya dalam transaksi *database*, mengosongkan keranjang, lalu mengirim notifikasi ke penjual; pesanan dibuat dengan status awal *pending*

Tabel 5.28 Membuat Pesanan dari Keranjang
Potongan Kode Membuat pesanan dari keranjang

```
async function buatPesananService(req) {
  const guest_id = req.body.guest_id || getGuestId(req);
  const { tipe_pengantaran, nama_pemesan, no_hp, catatan = '', diantar_ke } = req.body;

  if (!guest_id || !tipe_pengantaran || !nama_pemesan || !no_hp)
    throw httpErr(400, 'Data wajib tidak lengkap.');
```

```
  const client = await pool.connect();
  try {
    const keranjangRes = await client.query(
      `SELECT k.id AS keranjang_id, k.menu_id, k.jumlah,
           m.nama_menu, m.harga, m.foto_menu, m.kios_id, m.estimasi menit
      FROM keranjang k JOIN menu m ON k.menu_id = m.id
      WHERE k.guest_id = $1`,
      [guest_id]
    );
    if (!keranjangRes.rows.length) throw httpErr(400, 'Keranjang kosong');
```

```
    const items = keranjangRes.rows;
    const kios_id = items[0].kios_id;

    await client.query('BEGIN');
```

```
    const total_harga = items.reduce((sum, i) => sum + i.harga * i.jumlah, 0);
    const total_estimasi = items.reduce((sum, i) => sum + (i.estimasi menit || 0) *
i.jumlah, 0);

    // antrean aktif (info-tidak dipakai langsung dalam response)
    await client.query(
      `SELECT id, status, total_estimasi, estimasi_selesai_at
      FROM pesanan WHERE kios_id=$1 AND status IN
('paid','processing','ready','delivering')
      ORDER BY paid_at ASC`,
      [kios_id]
    );

    const pesananRes = await client.query(
      `INSERT INTO pesanan (
        guest_id, kios_id, tipe_pengantaran, nama_pemesan, no_hp, catatan, diantar_ke,
        total_harga, total_estimasi, status, estimasi_mulai_at, estimasi_selesai_at,
        waktu_proses_mulai
      )
      VALUES ($1,$2,$3,$4,$5,$6,$7,$8,$9,'pending', NULL, NULL, NULL)
      RETURNING *`,
      [guest_id, kios_id, tipe_pengantaran, nama_pemesan, no_hp, catatan, diantar_ke
|| null,
        total_harga, total_estimasi]
    );

    const pesanan = pesananRes.rows[0];
```

Potongan Kode Membuat pesanan dari keranjang
<pre> for (const item of items) { await client.query(`INSERT INTO pesanan_detail     (pesanan_id,menu_id,nama_menu,harga,foto_menu,jumlah,subtotal)     VALUES (\$1,\$2,\$3,\$4,\$5,\$6,\$7)` , [pesanan.id, item.menu_id, item.nama_menu, item.harga, item.foto_menu, item.jumlah, item.harga * item.jumlah]); } await client.query('DELETE FROM keranjang WHERE guest_id=\$1', [guest_id]); await client.query('COMMIT'); notifyPenjualService(kios_id, pesanan.id).catch(err => console.error(err)); return { status: 201, body: { message: 'Pesanan berhasil dibuat', pesanan } }; } catch (err) { await (async () => { try { await client.query('ROLLBACK'); } catch (_) {} })(); if (err.status) throw err; console.error(err); throw httpErr(500, 'Terjadi kesalahan server'); } finally { client.release(); } } </pre>
Deskripsi
Kode ini membuat pesanan baru dari isi keranjang milik guest_id dengan memverifikasi kelengkapan data, membaca daftar item keranjang, menghitung total harga serta total estimasi, dan menyimpan pesanan beserta detail item dalam satu transaksi agar konsisten; setelah berhasil, keranjang dibersihkan, penjual diberi notifikasi, dan respons 201 dikembalikan dengan data pesanan berstatus pending.

5. Potongan Kode Menampilkan Tipe Pengantaran

Potongan kode dari pesananController ini menampilkan daftar pesanan masuk untuk penjual sekaligus menormalkan tipe pengantaran: jika tipe_pengantaran = 'diantar' maka nilai yang ditampilkan adalah alamat diantar_ke, jika bukan maka ditampilkan teks Ambil Sendiri.

Tabel 5.29 Menampilkan Tipe Pengantaran
Potongan Kode Menampilkan tipe pengantaran
<pre> const getPesananMasuk = async (req, res) => { try { const { status, body } = await getPesananMasukService(req); return res.status(status).json(body); } catch (err) { if (err.status) return res.status(err.status).json({ message: err.message }); console.error("getPesananMasuk error:", err); return res.status(500).json({ message: "Terjadi kesalahan server" }); } }; </pre>
Deskripsi
Endpoint ini mengembalikan daftar pesanan aktif milik penjual dan menampilkan tipe pengantaran dalam bentuk yang langsung terbaca: alamat tujuan jika diantar, atau teks "Ambil Sendiri" jika pickup, sekaligus melampirkan informasi lain seperti nomor antrean, metode bayar, total harga, total estimasi, dan label status yang sesuai.

6. Potongan Kode Menampilkan Daftar Pesanan dan Detail Pesanan

Potongan kode dari `pesananController` ini menampilkan detail lengkap satu pesanan aktif milik penjual. *Endpoint* mengambil antrian pesanan yang masih aktif, menghitung estimasi waktu mulai dan selesai berdasarkan urutan proses, kemudian mengembalikan data pesanan berikut nomor antrian, label status yang kontekstual, metode pembayaran, tipe pengantaran, serta daftar menu dengan `foto_menu` menggunakan *path* asli.

Tabel 5.30 Menampilkan Detail Pesanan

Potongan Kode Menampilkan detail pesanan
<pre>const getDetailPesananMasuk = async (req, res) => { try { const { status, body } = await getDetailPesananMasukService(req); return res.status(status).json(body); } catch (err) { if (err.status) return res.status(err.status).json({ message: err.message }); console.error('getDetailPesananMasuk error:', err); return res.status(500).json({ message: "Gagal mengambil detail pesanan" }); } };</pre>
Deskripsi
Kode ini mengambil antrian pesanan aktif pada kios penjual, menghitung estimasi mulai dan selesai tiap pesanan, lalu mengembalikan detail pesanan yang diminta termasuk nomor antrian, label status yang disesuaikan dengan tipe pengantaran, informasi pembayaran, alamat atau keterangan ambil sendiri, total harga dan total estimasi, catatan, serta daftar menu dengan <code>foto_menu</code>

7. Potongan Kode Menampilkan Status dan Estimasi Waktu Pesanan

Potongan kode dari `pesananController` ini digunakan untuk menampilkan status pesanan pembeli sekaligus menghitung estimasi waktu selesai berdasarkan antrian pesanan di kios. *Endpoint* ini mengecek apakah pesanan milik *guest*, lalu mengambil antrian aktif dan mencoba menghitung estimasi mulai maupun selesai untuk tiap pesanan.

Tabel 5.31 Menampilkan Status dan Estimasi Waktu Pesanan

Potongan Kode Menampilkan status dan estimasi waktu pesanan
<pre>const getStatusPesananGuest = async (req, res) => { try { const { status, body } = await getStatusPesananGuestService(req); return res.status(status).json(body); } catch (err) { if (err.status) return res.status(err.status).json({ message: err.message }); console.error("getStatusPesananGuest error:", err); return res.status(500).json({ message: "Gagal mengambil status pesanan" }); } };</pre>
Deskripsi
Fungsionalitas ini sudah berhasil menampilkan status pesanan pembeli dengan benar, namun perhitungan estimasi waktu selesai pesanan masih belum sesuai harapan. Estimasi hanya dihitung sederhana berdasarkan waktu mulai proses dan total estimasi menit, sehingga hasilnya belum konsisten dengan antrian sebenarnya.

5.1.4.4 Sprint Review Ke-3

Sesi Sprint Review Ke-3 dilaksanakan pada akhir sprint secara internal oleh tim PasMeal. Pada sesi ini, setiap backlog yang telah dikerjakan ditunjukkan dan diuji untuk memastikan fungsionalitas berjalan sesuai harapan.

Tabel 5.32 Sprint Review ke-3

Kode	Nama backlog	Status	Hasil dan Umpan Balik
PAS-BE.17	Beranda	Selesai	Fungsionalitas beranda berhasil diimplementasikan dan berjalan sesuai harapan.
PAS-BE.13	Mengubah Status Pesanan	Selesai	Fungsionalitas status pesanan berhasil diimplementasikan dan berjalan sesuai harapan.
PAS-BE.05	Mengelola keranjang	Selesai	Fungsionalitas keranjang berhasil berjalan, termasuk menambah, mengubah, dan menghapus item sesuai harapan.
PAS-BE.06	Membuat pesanan dari keranjang	Selesai	Fungsionalitas membuat pesanan dari keranjang berhasil berjalan, meskipun perhitungan estimasi waktu masih perlu penyesuaian.
PAS-BE.14	Menampilkan tipe pengantaran	Selesai	Fungsionalitas menampilkan tipe pengantaran berhasil berjalan dan sesuai dengan kebutuhan (Ambil Sendiri/Diantar).
PAS-BE.12	Menampilkan daftar pesanan dan detail pesanan	Tidak Selesai	Fungsionalitas menampilkan daftar pesanan dan detail pesanan sudah berjalan, namun terdapat bug pada foto menu yang terkadang hilang.
PAS-BE.08	Menampilkan status dan estimasi waktu pesanan	Tidak Selesai	Fungsionalitas menampilkan status pesanan sudah berjalan, namun estimasi waktu pesanan masih belum sesuai harapan.

Sebagian besar backlog selesai dengan baik, terutama pencarian global, validasi keranjang, tipe pengantaran, dan pesanan dari keranjang. Namun, dua backlog masih belum selesai sepenuhnya yaitu daftar pesanan (PAS-BE.12) dan estimasi waktu pesanan (PAS-BE.08).

5.1.5 Sprint ke-4

Sprint keempat merupakan iterasi terakhir dalam pengembangan aplikasi PasMeal. Fokus pengerjaan diarahkan pada penyelesaian fitur-fitur yang masih tertunda serta penyempurnaan integrasi sistem agar aplikasi dapat digunakan secara *end-to-end*. Pada tahap ini, fitur utama yang ditangani mencakup integrasi *payment gateway*, pengelolaan menu, manajemen keranjang, pengiriman notifikasi, serta penampilan daftar pesanan lengkap dengan status dan estimasi waktunya. Sprint Ke-4 dilaksanakan pada tanggal 22 Agustus 2025 – 30 Agustus 2025.

5.1.5.1 Sprint Planning Ke-4

Pada sesi sprint planning keempat, tim menyusun rencana pekerjaan dengan fokus pada penyelesaian backlog yang masih tertunda dari sprint sebelumnya, sekaligus memastikan seluruh fungsionalitas utama dapat berjalan dengan stabil. Item backlog yang dipilih mencakup:

Tabel 5.33 Sprint Planning ke-4

Kode	Nama backlog	Estimasi Waktu	Prioritas
PAS-BE.07	Menyediakan <i>payment gateway</i>	9 Hari	Tinggi
PAS-BE.01	Menampilkan daftar kios		Tinggi
PAS-BE.02	Menampilkan daftar menu per kios		Tinggi

Kode	Nama backlog	Estimasi Waktu	Prioritas
PAS-BE.11	Mengirim notifikasi pesanan masuk		Tinggi
PAS-BE.10	Mengelola menu		Tinggi
PAS-BE.03	Menampilkan detail menu		Tinggi
PAS-BE.08	Menampilkan status dan estimasi waktu pesanan		Tinggi
PAS-BE.09	Mengirim notifikasi pesanan selesai		Tinggi
PAS-BE.12	Menampilkan daftar pesanan dan detail pesanan		Tinggi

5.1.5.2 Sprint Backlog Ke-4

Pada Sprint Ke-4, implementasi diarahkan pada penguatan modul transaksi dan *monitoring* pesanan, termasuk perbaikan bug serta optimasi tampilan. Fokus pengerjaan juga mencakup penyempurnaan integrasi cloudinary untuk menampilkan foto secara konsisten. Rincian kebutuhan teknis setiap backlog ditampilkan pada table

Tabel 5.34 Sprint Backlog ke-4

Kode	Nama backlog	Kebutuhan	Waktu Terpakai
PAS-BE.07	Menyediakan <i>payment gateway</i>	Konfigurasi api url di website midtrans	9 hari
PAS-BE.01	Menampilkan daftar kios	Install cloudinary untuk penyimpanan foto dan menerapkannya di kios controller agar foto dapat ditampilkan	
PAS-BE.02	Menampilkan daftar menu per kios	Install cloudinary untuk penyimpanan foto dan menerapkannya di menu controller agar foto dapat ditampilkan	
PAS-BE.11	Mengirim notifikasi pesanan masuk	Membuat wa logs agar memastikan notifikasi terkirim hanya sekali ke penjual	
PAS-BE.10	Mengelola menu	Install cloudinary untuk penyimpanan foto dan menerapkannya di menu controller agar foto dapat ditampilkan	
PAS-BE.03	Menampilkan detail menu	Install cloudinary untuk penyimpanan foto dan menerapkannya di menu controller agar foto dapat ditampilkan	
PAS-BE.08	Menampilkan status dan estimasi waktu pesanan	Memperbaiki fungsi estimasi agar ketika penjual klik proses estimasi baru berjalan	
PAS-BE.09	Mengirim notifikasi pesanan selesai	Membuat fungsi untuk mengirim notifikasi WhatsApp ke pembeli setelah pesanan selesai, berisi detail pesanan.	
PAS-BE.12	Menampilkan daftar pesanan dan detail pesanan	Install cloudinary untuk penyimpanan foto dan menerapkannya di menu controller agar foto dapat ditampilkan	

5.1.5.3 Increment Ke-4

Pada akhir Sprint Ke-4, sistem PasMeal telah memiliki alur transaksi yang lebih lengkap: mulai dari pemilihan menu, penambahan ke keranjang, pembuatan pesanan, pembayaran, hingga notifikasi ke penjual dan pembeli. Integrasi cloudinary berhasil membuat foto kios dan menu tampil konsisten. Selain itu, fitur status pesanan dan estimasi waktu semakin akurat meskipun masih ada beberapa penyempurnaan minor yang perlu dilakukan.

1. Potongan Kode Menyediakan Payment Gateway

Untuk mendukung transaksi pembayaran secara online, sistem diintegrasikan dengan *payment gateway* Midtrans. Integrasi ini memungkinkan proses pembayaran dilakukan secara otomatis, mulai dari verifikasi hingga pembaruan status pesanan di dalam basis data.

Tabel 5.35 Potongan Kode Notifikasi Payment Gateway

Potongan Kode notifikasi payment gateway
<pre>const handleNotification = async (req, res) => { try { const { status, body } = await handleNotificationService(req); return res.status(status).json(body); } catch (err) { if (err.status) return res.status(err.status).json({ message: err.message }); // eslint-disable-next-line no-console console.error("[NOTIFICATION ERROR]", err); return res.status(500).json({ message: "Gagal memproses notifikasi" }); } };</pre>
Deskripsi
Kode ini memastikan bahwa setiap notifikasi pembayaran dari Midtrans berhasil diterima oleh <i>backend</i>, lalu diproses untuk memperbarui status pesanan di <i>database</i>. Jika transaksi sudah settlement (dibayar), status pesanan diubah menjadi paid dan sistem secara otomatis mengirimkan notifikasi WhatsApp ke penjual agar segera memproses pesanan. Dengan alur ini, <i>backend</i> menegaskan bahwa notifikasi telah berhasil diterima dan ditindaklanjuti.

2. Potongan Kode Cloudinary untuk Menampilkan Foto

Potongan kode ini digunakan untuk mengatur Cloudinary sebagai media *storage*. Dengan konfigurasi ini, aplikasi dapat mengunggah gambar ke Cloudinary dan menyimpannya sebagai URL agar bisa ditampilkan kembali di *frontend*. Semua *endpoint* yang ada bug di bagian foto di perbaiki lewat cloudinary.

Tabel 5.36 Potongan Kode Cloudinary

Potongan Kode Cloudinary
<pre>const cloudinary = require('cloudinary').v2; // konfigurasi Cloudinary dari environment variables cloudinary.config({ cloud_name: process.env.CLOUD_NAME, api_key: process.env.CLOUD_KEY, api_secret: process.env.CLOUD_SECRET }); module.exports = cloudinary;</pre>
Deskripsi
Kode ini mengimpor <i>library</i> cloudinary, lalu membaca variabel environment (CLOUD_NAME, CLOUD_KEY, CLOUD_SECRET) untuk autentikasi. Hasil konfigurasi diekspor agar dapat dipakai di controller lain seperti menuController atau kiosController untuk proses upload foto. Dengan cara ini, file gambar tidak disimpan di <i>server</i> lokal, melainkan di-host di Cloudinary sehingga lebih mudah diakses oleh <i>client</i> melalui URL.

3. Potongan Kode Mengirim Notifikasi Pesanan Masuk

Potongan kode dari pesananController ini digunakan untuk mengirim notifikasi WhatsApp ke penjual setiap kali ada pesanan baru yang masuk. Sistem memastikan notifikasi hanya dikirim satu kali per pesanan dengan memanfaatkan tabel log (wa_notif_logs).

Tabel 5.37 Potongan Kode Notifikasi Pesanan Masuk

Potongan Kode notifikasi pesanan masuk
<pre> async function notifyPenjualService(kiosId, pesananId) { try { const cekLog = await pool.query('SELECT 1 FROM wa_notif_logs WHERE pesanan_id = \$1 LIMIT 1', [pesananId]); if (cekLog.rows.length) { console.log(`Notif pesanan \${pesananId} sudah pernah dikirim, skip.`); return; } const kiosData = await pool.query('SELECT penjual_id FROM kios WHERE id = \$1', [kiosId]); if (!kiosData.rows.length) return; const penjualId = kiosData.rows[0].penjual_id; const penjualData = await pool.query('SELECT no_hp FROM penjual WHERE id = \$1', [penjualId]); if (!penjualData.rows.length) return; const noHpPenjual = penjualData.rows[0].no_hp; const linkDashboard = `https://pas-meal.vercel.app/`; const message = `🔴 Pesan Baru! Silakan klik tautan ini untuk melihat pesanan:     \${linkDashboard}`; await sendWaMessage(noHpPenjual, message); console.log(`WA notifikasi pesanan \${pesananId} ke penjual (\${noHpPenjual})     terkirim.`); await pool.query('INSERT INTO wa_notif_logs (pesanan_id) VALUES (\$1)', [pesananId]); } catch (err) { console.error('Gagal kirim WA ke penjual:', err); } } </pre>
Deskripsi
Kode ini mengirimkan pesan WhatsApp otomatis ke penjual ketika ada pesanan baru. Sebelum mengirim, sistem memeriksa tabel <code>wa_notif_logs</code> untuk memastikan notifikasi belum pernah dikirim sebelumnya. Jika belum, sistem mengambil data penjual berdasarkan <code>kiosId</code>, lalu mengirim pesan berisi tautan menuju dashboard. Setelah berhasil, sistem mencatat log agar notifikasi tidak terkirim ganda. Dengan mekanisme ini, penjual selalu mendapatkan notifikasi pesanan baru secara cepat dan terkontrol.

4. Potongan Kode Menampilkan Status dan Estimasi Waktu Pesanan

Potongan kode dari `pesananController` ini menampilkan status pesanan pembeli sekaligus estimasi selesai yang berjalan secara akurat: jika pesanan sudah mulai diproses, estimasi dihitung dari `waktu_proses_mulai` ditambah durasi (`total_estimasi`) dan dikembalikan dalam bentuk timestamp; jika pesanan sudah `done`, sistem segera mengembalikan status beserta waktu selesai saat ini.

Tabel 5.38 Potongan Kode Menampilkan Status Dan Estimasi Waktu Pesanan

Potongan Kode menampilkan status dan estimasi waktu pesanan
<pre> async function getStatusPesananGuestService(req) { const { id } = req.params; const guest_id = getGuestId(req); const pesananRes = await pool.query(`SELECT kios_id, status FROM pesanan </pre>

Potongan Kode menampilkan status dan estimasi waktu pesanan

```

WHERE id = $1 AND guest_id = $2`,
[id, guest_id]
);
if (pesananRes.rows.length === 0) throw httpErr(404, 'Pesanan tidak ditemukan');

const { kios_id, status } = pesananRes.rows[0];

if (status === 'done') {
  return { status: 200, body: { status, estimasi_selesai_at: new Date().toISOString() } };
}

const antreanRes = await pool.query(
  `SELECT id, paid_at, total_estimasi, status, estimasi_mulai_at, waktu_proses_mulai
  FROM pesanan
  WHERE kios_id = $1
  AND status IN ('paid', 'processing', 'ready', 'delivering')
  ORDER BY paid_at ASC`,
  [kios_id]
);
if (antreanRes.rows.length === 0) throw httpErr(404, 'Tidak ada antrean aktif');

let waktuSelesaiSebelumnya = null;

const antreanDenganJadwal = antreanRes.rows.map(p => {
  let waktuMulaiMillis = null;
  if (p.waktu_proses_mulai) {
    waktuMulaiMillis = new Date(p.waktu_proses_mulai).getTime();
  }
  const durasiMillis = Number(p.total_estimasi || 0) * 60000;
  const waktuSelesaiMillis = waktuMulaiMillis !== null ? waktuMulaiMillis +
durasiMillis : null;

  if (waktuSelesaiMillis !== null) {
    waktuSelesaiSebelumnya = waktuSelesaiMillis;
  }

  return {
    ...p,
    estimasi_selesai_at_calc: waktuSelesaiMillis ? new
Date(waktuSelesaiMillis).toISOString() : null,
  };
});

const targetPesanan = antreanDenganJadwal.find(p => p.id == id);

if (!targetPesanan) {
  return { status: 200, body: { status, estimasi_selesai_at: null } };
}

return {
  status: 200,
  body: {
    status: targetPesanan.status,
    estimasi_selesai_at: targetPesanan.estimasi_selesai_at_calc,
  }
};
}

```

Deskripsi

Fungsi ini mengembalikan status pesanan milik pembeli dan estimasi selesai yang berjalan berdasarkan waktu mulai proses (waktu_proses_mulai) ditambah total_estimasi dalam menit; jika pesanan sudah selesai, sistem langsung merespons dengan status done dan cap waktu selesai saat ini.

5. Potongan Kode Mengirim Notifikasi Pesanan Selesai

Potongan kode dari pesananController ini digunakan untuk mengirim notifikasi WhatsApp ke pembeli ketika pesanan sudah selesai. Sistem mengambil data pesanan dan detail item, lalu membentuk pesan personal berisi ringkasan menu, total harga, serta tipe pengantaran, kemudian mengirimkannya ke nomor HP pembeli.

Tabel 5.39 Potongan Kode Mengirim Notifikasi Pesanan Selesai

Potongan Kode notifikasi pesanan selesai
<pre>const notifyPembeliPesananSelesai = async (pesananId) => { return notifyPembeliPesananSelesaiService(pesananId); };</pre>
Deskripsi
Kode ini dipicu saat pesanan berubah status menjadi selesai. Sistem mengambil informasi pembeli dan detail menu, lalu menyusun pesan WhatsApp berisi daftar item yang dipesan, total harga, dan informasi pengantaran. Pesan dikirim ke nomor pembeli agar mereka langsung mengetahui bahwa pesannya sudah selesai dan siap diterima.

5.1.5.4 Sprint Review Ke-4

Sesi Sprint Review Ke-4 dilaksanakan pada akhir sprint secara internal oleh tim PasMeal. Pada sesi ini, setiap backlog yang telah dikerjakan ditunjukkan dan diuji untuk memastikan fungsionalitas berjalan sesuai harapan.

Tabel 5.40 Sprint Review ke-4

Kode	Nama backlog	Status	Hasil dan Umpan Balik
PAS-BE.07	Menyediakan payment gateway	Selesai	Fungsionalitas payment gateway berhasil diimplementasikan dan berjalan sesuai harapan.
PAS-BE.01	Menampilkan daftar kios	Selesai	Fungsionalitas menampilkan daftar kios berhasil berjalan dan menampilkan data sesuai kebutuhan.
PAS-BE.02	Menampilkan daftar menu per kios	Selesai	Fungsionalitas menampilkan daftar menu per kios berhasil berjalan dan menampilkan data sesuai kebutuhan.
PAS-BE.11	Mengirim notifikasi pesanan masuk	Tidak Selesai	Fungsionalitas notifikasi pesanan masuk berhasil berjalan dan pesan terkirim sesuai kebutuhan. Namun, notifikasi sudah terkirim ke penjual bahkan sebelum pembeli menyelesaikan pembayaran.
PAS-BE.10	Mengelola menu	Selesai	Fungsionalitas pengelolaan menu berhasil berjalan, termasuk menambah, memperbarui, dan menghapus menu.
PAS-BE.03	Menampilkan detail menu	Selesai	Fungsionalitas menampilkan detail menu berhasil berjalan dan menampilkan data sesuai kebutuhan.
PAS-BE.08	Menampilkan status dan estimasi waktu pesanan	Tidak Selesai	Fungsionalitas untuk menampilkan status dan estimasi waktu pesanan sudah berjalan sesuai harapan, namun estimasi waktu tidak berkurang meskipun pesanan awal telah diselesaikan.
PAS-BE.09	Mengirim notifikasi pesanan selesai	Selesai	Fungsionalitas notifikasi pesanan selesai berhasil berjalan, dan pesan WhatsApp terkirim ke pembeli sesuai kebutuhan.
PAS-BE.12	Menampilkan daftar pesanan dan detail pesanan	Selesai	Fungsionalitas menampilkan daftar pesanan dan detail pesanan berhasil berjalan dan menampilkan data sesuai kebutuhan.

Seluruh backlog utama pada Sprint Ke-4 berhasil diselesaikan. Beberapa kendala seperti notifikasi pesanan masuk yang terkirim sebelum pembayaran selesai serta fitur estimasi waktu yang belum sepenuhnya berjalan masih ditemukan, namun secara keseluruhan sistem sudah dapat digunakan dengan baik.

5.2 Delivery

Setelah tahap scrum process selesai, tahap selanjutnya adalah Delivery. Pada tahap ini, increment produk yang telah fungsional dan teruji diserahkan untuk proses selanjutnya. Dalam konteks pengembangan PasMeal, proses delivery dilakukan dengan menyediakan *endpoint* API yang stabil dan telah di-*deploy* ke lingkungan *server*. Hasil akhir dari tahap ini adalah sebuah base URL API yang diberikan kepada *frontend* untuk proses integrasi, yang menandai penyelesaian dari lingkup pekerjaan pada penelitian ini.

5.2.1 Pengujian perangkat lunak

Pengujian perangkat lunak dilakukan untuk mengecek setiap fungsionalitas perangkat lunak apakah sudah sesuai dengan kebutuhan dan terbebas dari *bug*. Metode pengujian yang dilakukan adalah menggunakan *black box testing*. *Black box testing* berfokus pada fungsionalitas sistem. Berikut merupakan pengujian yang dilakukan pada *e-canteen* PasMeal.

5.2.1.1 Cek Fungsional Autentikasi

Cek fungsional autentikasi bertujuan untuk menguji apakah *backend e-canteen* PasMeal dapat melakukan autentikasi sesuai dengan request *client*. Pengujian yang dilakukan meliputi register, login, pengiriman OTP. Rincian hasil pengujian untuk setiap skenario dapat dilihat pada lampiran.

5.2.1.2 Cek Fungsional Beranda

Cek fungsional beranda bertujuan untuk menguji apakah *backend e-canteen* PasMeal dapat menampilkan data awal (landing) sesuai kebutuhan. Pengujian dilakukan pada endpoint beranda untuk memastikan API mengembalikan respons yang valid. Rincian hasil pengujian untuk setiap skenario dapat dilihat pada lampiran.

5.2.1.3 Cek Fungsional Menampilkan Daftar Kios

Pengujian ini bertujuan untuk memastikan API dapat menampilkan daftar kios yang terdaftar

5.2.1.4 Cek Fungsional Menampilkan Daftar Menu

Pengujian dilakukan untuk memastikan sistem dapat menampilkan daftar menu sesuai kios yang dipilih.

5.2.1.5 Cek Fungsional Menampilkan Detail Menu

Pengujian ini dilakukan untuk memastikan detail menu (nama, harga, deskripsi, foto) ditampilkan dengan benar.

5.2.1.6 Cek Fungsional Mencari Menu

Pengujian ini bertujuan untuk memastikan API dapat mencari menu berdasarkan kata kunci.

5.2.1.7 Cek Fungsional Mengelola Keranjang

Pengujian dilakukan untuk memastikan sistem dapat menambah, menampilkan, mengubah, dan menghapus item di keranjang.

5.2.1.8 Cek Fungsional Membuat Pesanan Dari Keranjang

Pengujian ini bertujuan untuk memastikan pesanan dapat dibuat berdasarkan isi keranjang.

5.2.1.9 Cek Fungsional Menyediakan Payment Gateway

Pengujian dilakukan untuk memastikan integrasi payment gateway berjalan baik, termasuk verifikasi pembayaran.

5.2.1.10 Cek Fungsional Menampilkan Status Estimasi Waktu Pesanan

Pengujian ini memastikan sistem dapat menampilkan status pesanan dan estimasi waktu secara akurat.

5.2.1.11 Cek Fungsional Mengirim Notifikasi Pesanan Selesai

Pengujian ini bertujuan untuk memastikan notifikasi dapat terkirim kepada pembeli saat pesanan selesai.

5.2.1.12 Cek Fungsional Mengelola Menu

Pengujian dilakukan untuk memastikan penjual dapat menambah, menampilkan, mengubah, dan menghapus menu

5.2.1.13 Cek Fungsional Mengirim Notifikasi Pesanan Masuk

Pengujian bertujuan untuk memastikan penjual menerima notifikasi saat ada pesanan masuk.

5.2.1.14 Cek Fungsional Menampilkan Daftar Pesanan Dan Detail Pesanan

Pengujian dilakukan untuk memastikan penjual dapat melihat daftar pesanan dan detail tiap pesanan.

5.2.1.15 Cek Fungsional Mengubah Status Pesanan

Pengujian ini bertujuan untuk memastikan status pesanan dapat diubah sesuai alur proses.

5.2.1.16 Cek Fungsional Menampilkan Tipe Pengantaran

Pengujian ini memastikan sistem dapat menampilkan tipe pengantaran pesanan (Ambil Sendiri / Diantar).

Tabel 5.41 Rekapitulasi Pengujian Perangkat Lunak

No	Test Scenario	Jumlah Test Case	Presentase Lulus Uji
1	TS-01 Cek fungsi registrasi penjual	4	100%
2	TS-02 Cek fungsi registrasi kios	3	100%
3	TS-03 Cek fungsi OTP	6	100%

No	Test Scenario	Jumlah Test Case	Presentase Lulus Uji
4	TC-04 Cek fungsi Login	4	100%
5	TS-05 Cek Fungsi Lupa & Reset Password	5	100%
6	TS-06 Cek fungsi Logout	2	100%
7	TS-07 Cek update profil penjual	3	100%
8	TS-08 Cek fungsi getAllMenu	1	100%
9	TS-09 Cek fungsi addMenu	2	100%
10	TS-10 Cek fungsi updateMenu	2	100%
11	TS-11 Cek fungsi getMenuById	3	100%
12	TS-12 Cek fungsi deleteMenu	3	100%
13	TS-13 Cek fungsi getMenuPaginated	2	100%
14	TS-14 Cek fungsi getNewMenus	1	100%
15	TS-15 Cek fungsi searchMenus	2	100%
16	TS-16 Cek fungsi searchMenusByKios	3	100%
17	TS-17 Cek fungsi getMenuByIdForBuyer	3	100%
18	TS-18 Cek fungsi getKiosHomepage	1	100%
19	TS-19 Cek fungsi searchKios	2	100%
20	TS-20 Cek fungsi getAllKios	1	100%
21	TS-21 Cek fungsi getMenuByKios	2	100%
22	TS-22 Cek fungsi getKiosByPenjual	2	100%
23	TS-23 Cek fungsi updateKios	3	100%
24	TS-24 Cek fungsi getKiosDetail	2	100%
25	TS-25 Cek Fungsi tambah keranjang	4	100%
26	TS-26 Cek Fungsi get keranjang	2	100%
27	TS-27 Cek Fungsi update keranjang	3	100%
28	TS-28 Cek Fungsi Hapus item keranjang	2	100%
29	TS-29 Cek Fungsi notifyPenjual	2	50%
30	TS-30 Cek Fungsi notifyPembeliPesananSelesai	2	100%
31	TS-31 Cek Fungsi buatPesanan	3	100%
32	TS-32 Cek Fungsi getStatusPesananGuest	5	80%
33	TS-33 Cek Fungsi getPesananByGuest	1	100%
34	TS-34 Cek Fungsi getDetailPesanan	2	100%
35	TS-35 Cek Fungsi getPesananMasuk	2	100%
36	TS-36 Cek Fungsi getDetailPesananMasuk	3	100%
37	TS-37 Cek Fungsi countPesananMasuk	1	100%
38	TS-38 Cek Fungsi updateStatusPesanan	4	100%
39	TS-39 Cek Fungsi getRiwayatPesanan	1	100%
40	TS-40 Cek Fungsi getDetailRiwayatPesanan	2	100%
41	TS-41 Cek Fungsi CreateTransaction	4	100%
42	TS-42 Cek Fungsi handleNotification	2	50%
43	TS-43 Cek Fungsi getDashboardData	1	100%
44	TS-44 Cek Fungsi searchAll	5	100%

Keterangan :

1. TS-29 : Fungsi dasar notifikasi penjual (kirim WA & log) bekerja, tapi gagal memenuhi rule bisnis karena notifikasi masih terkirim sebelum pembayaran sukses.
2. TS-32 : Fungsi status pesanan bekerja untuk done, processing, unpaid, dan not-owner. Namun ETA tidak ter-update ketika pesanan pertama selesai lebih cepat.
3. TS-42 : Callback settlement berhasil update status & paid_at, tapi belum ada trigger WA penjual saat *payment success*.

Detail pengujian perangkat lunak dapat dilihat pada tabel berikut di bawah ini.

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
					, pesan "Token tidak ditemukan"	"message": "Token tidak ditemukan"	
		TC-2-03	Registrasi kios dengan nama sama	{ "nama_kios": "kios sukses", ... }	Status 409 Conflict, pesan "Nama kios telah digunakan, silakan pilih nama lain".	409 Conflict { "message": "Nama kios telah digunakan, silakan pilih nama lain" }	PASS
TS-03	Cek fungsi OTP	TC-3-01	Verifikasi OTP valid	{ "kode_otp": "123456" }	Status 200 OK, pesan "OTP berhasil diverifikasi", return penjual_id & token.	200 OK { "message": "OTP berhasil diverifikasi", "penjual_id": "136", "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJwZW5qdWFsX2lkjoiMTM2liwiaXNfdmVyaWZpZWQiOnRydWU-bBA" }	PASS
		TC-3-02	OTP salah/expired	{ "kode_otp": "999999" }	Status 400 Bad Request, pesan "OTP salah, kadaluarsa, atau sudah digunakan".	400 Bad Request { "message": "OTP salah, kadaluarsa, atau sudah digunakan" }	PASS
		TC-3-03	Resend OTP penjual valid	{ "no_hp": "08123456789" }	Status 200 OK, pesan "OTP baru telah dikirim".	200 OK { "message": "OTP baru telah dikirim" }	PASS
		TC-3-04	Resend OTP dengan no_hp tidak ada	{ "no_hp": "08111111111" }	Status 404 Not Found, pesan "Akun tidak ditemukan".	404 Not Found { "message": "Akun tidak ditemukan" }	PASS
		TC-3-04	Resend OTP setelah akun verified	{ "no_hp": "08123456789" }	Status 400 Bad Request, pesan "Akun sudah diverifikasi"	400 Bad Request { "message": "Akun sudah diverifikasi" }	PASS
		TC-3-06	Resend OTP sebelum OTP lama expired	{ "no_hp": "08123456789" }	Status 400 Bad Request, pesan "Silakan tunggu hingga OTP sebelumnya kedaluwarsa".	400 Bad Request { "message": "Silakan tunggu hingga OTP sebelumnya kedaluwarsa" }	PASS
TC-04	Cek fungsi Login	TC-4-01	Login dengan	{ "nama": "Legi",	Status 200 OK, pesan "Berhasil	200 OK {	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
		TC-5-04	Reset dengan <i>token</i> salah	{ "token": "salahtoken", "password": "baru12345", "confirmPassword": "baru12345" }	Status 400 Bad Request, pesan " <i>Token</i> tidak valid atau expired".	400 Bad Request { "message": " <i>Token</i> tidak valid atau expired" }	PASS
		TC-5-05	Reset dengan password beda	"token": "502817f40d09567a3d531357f26a228c9c", "password": "newPassword1", "confirmPassword": "newPassword2"	Status 400 Bad Request, pesan "Password tidak cocok".	400 Bad Request { "message": "Password tidak cocok" }	PASS
TS-06	Cek fungsi Logout	TC-6-01	Logout dengan <i>token</i> valid	Authorization: Bearer validtoken	Status 200 OK, pesan "Logout berhasil".	200 OK { "message": "Logout berhasil" }	PASS
		TC-6-02	Logout tanpa <i>token</i>	Tidak ada <i>token</i>	Status 401 Unauthorized, pesan " <i>Token</i> tidak ditemukan".	401Unauthorized { "message": " <i>Token</i> tidak ditemukan" }	PASS
TS-07	Cek update profil penjual	TC-7-01	Update profil berhasil dengan semua field	{ "nama": "Luna Baru", "no_hp": "08123456789", "email": "luna@mail.com" } dengan req.user.id valid	Status 200 OK, pesan "Profil berhasil diperbarui", response body data profil baru.	200 OK { "message": "Profil berhasil diperbarui", "data": { "id": "136", "nama": " Luna ", "no_hp": "08123456789", "email": " luna@mail.com ", "password": "\$2b\$10\$b0.tOdLEGE8YUylBRKEdNeDqFRm0j8TflgO9D5v3ZriTa3Vw6EPXm", "is_verified": true, "created_at": "2025-08-26T09:03:46.724Z" } }	PASS
		TC-7-02	Update profil hanya sebagian field	{ "nama": "Liuna Update" }	Status 200 OK, hanya field nama berubah, field lain tetap sama.	200 OK { "message": "Profil berhasil diperbarui", "data": { "id": "136", "nama": " Liuna ", "no_hp": "08123456789", "email": " luna@mail.com ", } }	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
						"password": "\$2b\$10\$b0.tOdLEGE8YUylBRKEdNeDqFRm0j8TflgO9D5v3ZriTa3Vw6EPXm", "is_verified": true, "created_at": "2025-08-26T09:03:46.724Z"	
		TC-7-03	Update profile tanpa token	Tidak ada token	Status 401 Unauthorized, pesan "Token tidak ditemukan".	401Unauthorized { "message": "Token tidak ditemukan" }	PASS

Tabel 5.43 Pengujian Menu

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
TS-08	Cek fungsi getAllMenu	TC-8-01	Ambil semua menu penjual berhasil	Authorization: Bearer validtoken	Status 200 OK, array menu milik penjual muncul.	200 OK { "id": "137", "nama_menu": "bakso cincang", "harga": 12000, "deskripsi": "bakso halalan toyiban", "foto_menu": "https://res.cloudinary.com/dn etoznik /image/upload/v1756204764 /menus/diznfl8rva6oofjp18ah.jpg", "status_tersedia": true, "estimasi_menit": 5, "created_at": "2025-08-26T10:39:25.366Z" }	PASS
TS-09	Cek fungsi addMenu	TC-9-01	Tambah menu berhasil	{ nama_menu: "bakso urat ", deskripsi: "bakso daging sapi ", harga: 12000, estimasi_menit: 5, foto_menu: bakso.jpg }	Status 201 Created, pesan "Menu berhasil ditambahkan", data menu tersimpan.	201 Created { "message": "Menu berhasil ditambahkan", "data": { "id": "142", "penjual_id": "136", "nama_menu": "bakso urat", "deskripsi": "bakso daging sapi", "harga": 12000, "foto_menu": "https://res.cloudinary.com/dn etoznik /image/upload/v1756204764 /menus/diznfl8rva6oofjp18ah.jpg", "status_tersedia": true, }	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
						<pre> "created_at": "2025-08-29T23:00:27.873Z", "kios_id": 62, "estimasi_menit": 5, "foto_public_id": "menus/wsnyhsx9bqchq8tnwk1n" } </pre>	
		TC-9-02	Tidak isi deskripsi & estimasi_menit	<pre> { nama_menu: "Mie Rebus", harga: 10000 } </pre>	Status 201 Created, deskripsi default "", estimasi default 10.	<pre> 201 Created { "message": "Menu berhasil ditambahkan", "data": { "id": "143", "penjual_id": "136", "nama_menu": "mie rebus", "deskripsi": "", "harga": 10000, "foto_menu": null, "status_tersedia": true, "created_at": "2025-08-29T23:05:23.603Z", "kios_id": 62, "estimasi_menit": 10, "foto_public_id": null } } </pre>	PASS
TS-10	Cek fungsi updateMenu	TC-10-01	Update menu berhasil	<pre> { foto_menu: "mie.png", harga: 10000 } </pre>	Status 200 OK, pesan "Menu berhasil diperbarui", data terupdate.	<pre> 200 OK { "message": "Menu berhasil diperbarui", "menu": { "id": "143", "penjual_id": "136", "nama_menu": "mie rebus", "deskripsi": "mie rebus berbagai rasa", "harga": 10000, "foto_menu": "https://res.cloudinary.com/dnetoznik/image/upload/v1756508924/o1pwvujstunrtdvrpss.jpg", "status_tersedia": true, "created_at": "2025-08-29T23:05:23.603Z", "kios_id": 62, "estimasi_menit": 10, "foto_public_id": "o1pwvujstunrtdvrpss" } } </pre>	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
		TC-10-02	Update dengan ID menu salah	/menu/9999	Status 404 Not Found, pesan "Menu tidak ditemukan".	404 Not Found { "message": "Menu tidak ditemukan" }	PASS
TS-11	Cek fungsi getMenuById	TC-11-01	Ambil detail menu berhasil(dengan ID valid)	/menu/143	Status 200 OK, data detail menu muncul.	200 OK { "id": "143", "penjual_id": "136", "nama_menu": "mie rebus", "deskripsi": "mie rebus berbagai rasa", "harga": 10000, "foto_menu": "https://res.cloudinary.com/dn-etoznik/image/upload/v1756508924/o1pwvujstunrtldvrpss.jpg", "status_tersedia": true, "created_at": "2025-08-29T23:05:23.603Z", "kios_id": 62, "estimasi_menit": 10, "foto_public_id": "o1pwvujstunrtldvrpss" }	
		TC-11-02	ID menu tidak valid(string)	/menu/abc	Status 400 Bad Request, pesan "ID menu tidak valid".	400 Bad Request { "message": "ID menu tidak valid" }	PASS
		TC-11-03	ID menu salah	/menu/9999	Status 404 Not Found, pesan "Menu tidak ditemukan".	404 Not Found { "message": "Menu tidak ditemukan" }	PASS
TS-12	Cek fungsi deleteMenu	TC-12-01	Hapus menu berhasil	/menu/143	Status 200 OK, pesan "Menu berhasil dihapus".	200 OK { "message": "Menu berhasil dihapus" }	
		TC-12-02	ID menu tidak valid (String)	/menu/abc	Status 400 Bad Request, pesan "ID menu tidak valid".	400 Bad Request { "message": "ID menu tidak valid" }	PASS
		TC-12-03	DELETE menu dengan ID salah	/menu/9999	Status 404 Not Found, pesan "Menu tidak ditemukan..."	404 Not Found { "message": "Menu tidak ditemukan atau bukan milik kamu" }	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
TS-13	Cek fungsi getMenusPaginated	TC-13-01	Ambil menu dengan pagination berhasil	page=1, limit=5	limit=5 Status 200 OK, response { page, limit, total, data }.	200 OK { "page": 1, "limit": 5, "total": 2, "data": [{ "id": "142", "nama_menu": "bakso urat", "harga": 12000, "deskripsi": "bakso daging sapi", "foto_menu": "https://res.cloudinary.com/dnetoznik/image/upload/v1756508427/menus/wsnyhsx9bqchq8tnwk1n.jpg", "status_tersedia": true, "estimasi_menit": 5, "created_at": "2025-08-29T23:00:27.873Z" }, { "id": "137", "nama_menu": "bakso cincang", "harga": 12000, "deskripsi": "bakso halalan toyiban", "foto_menu": "https://res.cloudinary.com/dnetoznik/image/upload/v1756508427/menus/wsnyhsx9bqchq8tnwk1n.jpg", "status_tersedia": true, "estimasi_menit": 5, "created_at": "2025-08-26T10:39:25.366Z" }] }	PASS
		TC-13-02	Pagination default (tanpa query)	tidak ada page, limit	Default page=1, limit=10.	"page": 1, "limit": 10, "total": 2,	PASS
		TC-14-01	Ambil 5 menu terbaru	-	Status 200 OK, max 5 data terbaru.	200 OK { "id": "142", "foto_menu": "https://res.cloudinary.com/dnetoznik/image"	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
						/upload/v1756382 122/menus /ev1kngjsfr4af7vw m2vn.jpg", "nama_menu": "bakso urat", "deskripsi": "bakso daging sapi", "harga": 12000, "estimasi_menit": 5, "status_tersedia": true }, { "id": "141", "foto_menu": "https://res.cloudinary.com /dnetoznik/image /upload/v1756382 122/menus /ev1kngjsfr4af7vw m2vn.jpg", "nama_menu": "Milk tea", "deskripsi": "teh susu", "harga": 6000, "estimasi_menit": 2, "status_tersedia": true }, { "id": "140", "foto_menu": "https://res.cloudinary.com /dnetoznik/image /upload/v1756382 122/menus /ev1kngjsfr4af7vw m2vn.jpg", "nama_menu": "katsu", "deskripsi": "katsu biasa", "harga": 10000, "estimasi_menit": 7, "status_tersedia": false }, { "id": "139", "foto_menu": "https://res.cloudinary.com /dnetoznik/image /upload/v1756382 122/menus /ev1kngjsfr4af7vw m2vn.jpg", "nama_menu": "Martabak telur", "deskripsi": "isi sayuran aja", "harga": 5000, "estimasi_menit": 5,	

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
						<pre> "status_tersedia": true }, { "id": "137", "foto_menu": "https://res.cloudinary.com /dnetoznik/image /upload/v1756382 122/menus /ev1kngjsfr4af7vw m2vn.jpg", "nama_menu": "bakso cincang", "deskripsi": "bakso halalan toyiban", "harga": 12000, "estimasi_menit": 5, "status_tersedia": true }] </pre>	
TS-15	Cek fungsi searchMenu	TC-15-01	Cari menu berhasil	query=bakso	Status 200 OK, list menu dengan nama mengandung "bakso".	<pre> 200 OK [{ "id": "142", "foto_menu": "https://res.cloudinary.com /dnetoznik/image /upload/v1756382 122/menus /ev1kngjsfr4af7vw m2vn.jpg", "nama_menu": "bakso urat", "deskripsi": "bakso daging sapi", "harga": 12000, "estimasi_menit": 5, "status_tersedia": true }] </pre>	PASS
		TC-15-02	GET /menu/search tanpa query	-	Status 400 Bad Request, pesan "Query pencarian wajib diisi".	<pre> 400 Bad Request { "message": "Query pencarian wajib diisi" } </pre>	PASS
TS-16	Cek fungsi searchMenu ByKios	TC-16-01	Cari menu di kios tertentu berhasil	kiosId=1, query=bakso	Status 200 OK, list menu dengan nama mengandung "bakso" di kios 1.	<pre> 200 OK { "id": "142", "nama_menu": "bakso urat", "harga": 12000, "deskripsi": "bakso daging sapi", "foto_menu": "https://res.cloudinary.com/ dnetoznik/ image/upload/v1756508427/ menus/wsnyhsx9bqchq8tnwk 1n.jpg", </pre>	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
						<pre> "status_tersedia": true, "estimasi_menit": 5, "created_at": "2025-08-29T23:00:27.873Z" }, { "id": "137", "nama_menu": "bakso cincang", "harga": 12000, "deskripsi": "bakso halalan toyiban", "foto_menu": "https://res.cloudinary.com/ dnetoznik/ image/upload/v1756508427/ menus/wsnysx9bqchq8tnwk 1n.jpg", "status_tersedia": true, "estimasi_menit": 5, "created_at": "2025-08-26T10:39:25.366Z" } </pre>	
		TC-16-02	Query kosong	kiosId=1	Status 400 Bad Request, pesan "Query pencarian wajib diisi".	<pre> 400 Bad Request { "message": "Query pencarian wajib diisi" } </pre>	PASS
		TC-16-03	ID kios tidak valid	kiosId=abc	Status 400 Bad Request, pesan "ID kios tidak valid".	<pre> 400 Bad Request { "message": "ID kios tidak valid" } </pre>	PASS
TS-17	Cek fungsi getMenuById ForBuyer	TC-17-01	Ambil detail menu berhasil	id=1	Status 200 OK, detail menu muncul.	<pre> 200 OK { "id": "132", "foto_menu": "https://res.cloudinary.com /dnetoznik/image /upload/v1756382122/menus /ev1kngjsfr4af7vwm2vn.jpg", "nama_menu": "Ayam cabe ijo", "deskripsi": "Ayam, nasi, sambal ijo, kol goreng", "harga": 15000, "estimasi_menit": 10, "status_tersedia": true } </pre>	PASS
		TC-17-02	ID menu tidak valid	id=abc	Status 400 Bad Request, pesan "ID menu tidak valid".	<pre> 400 Bad Request { "message": "ID menu tidak valid" } </pre>	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
		TC-17-03	Menu tidak ditemukan	id=9999	Status 404 Not Found, pesan "Menu tidak ditemukan".	404 Not Found { "message": "Menu tidak ditemukan" }	PASS

Tabel 5.44 Pengujian kios

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
TS-18	Cek fungsi getKios Homepage	TC-18-01	Ambil 8 kios terbaru berhasil	-	Status 200 OK, mengembalikan max 8 data kios terbaru	200 OK [{ "id": 62, "penjual_id": 136, "nama_kios": "Kantin Legi", "nama_bank": "BCA", "nomor_rekening": "1234567890", "created_at": "2025-08-26T09:04:07.354Z", "deskripsi": "warung halal", "gambar_kios": "https://res.cloudinary.com/dnetoznik/image/upload/v1756203117/kios/fkxsgyztsozs3cqzlavj.png", }, { "id": 60, "penjual_id": 132, "nama_kios": "Kios Syifa", "nama_bank": "bni", "nomor_rekening": "345789123456", "created_at": "2025-08-26T08:45:34.238Z", "deskripsi": "Nasi ayam", "gambar_kios": "https://res.cloudinary.com/dnetoznik/image/upload/v1756203117/kios/fkxsgyztsozs3cqzlavj.png", }, { "id": 59, "penjual_id": 130, "nama_kios": "KATSUNYAA", "nama_bank": "BNI", }]	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
						<pre> "nomor_rekening": "123456789", "created_at": "2025-08-24T15:33:22.544Z", "deskripsi": "katsu-katsu kekinian", "gambar_kios": null }] </pre>	
TS-19	Cek fungsi searchKios	TC-19-01	Cari kios dengan nama valid	/search?query=Bakso	Status 200 OK, hasil list kios mengandung kata "Bakso"	<pre> 200 OK { "id": 62, "penjual_id": 136, "nama_kios": "Kantin Legi", "nama_bank": "BCA", "nomor_rekening": "1234567890", "created_at": "2025-08-26T09:04:07.354Z", "deskripsi": "warung halal", "gambar_kios": "https://res.cloudinary.com/dnetoznik/image/upload/v1756203117/kios/fkxsgyztsozs3cqzlavj.png" } </pre>	PASS
		TC-19-02	Cari kios dengan hasil kosong	/search?query=TidakAda	Status 200 OK, mengembalikan array kosong	<pre> 200 OK [] </pre>	PASS
TS-20	Cek fungsi getAllKios	TC-20-01	Ambil semua kios berhasil	-	Status 200 OK, mengembalikan semua kiosurut dari terbaru	<pre> 200 OK [{ "id": 62, "penjual_id": 136, "nama_kios": "Kantin Legi", "nama_bank": "BCA", "nomor_rekening": "1234567890", "created_at": "2025-08-26T09:04:07.354Z", "deskripsi": "warung halal", "gambar_kios": "https://res.cloudinary.com/dnetoznik/image/upload/v1756203117/kios/fkxsgyztsozs3cqzlavj.png" }, { "id": 60, "penjual_id": 132, "nama_kios": "Kios Syifa", </pre>	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
						<pre> "nama_bank": "bni", "nomor_rekening": "345789123456", "created_at": "2025-08-26T08:45:34.238Z", "deskripsi": "Nasi ayam", "gambar_kios": "https://res.cloudinary.com/dnetoznik/image/upload/v1756203117/kios/fkxsgyztsozs3cqzlavj.png" }, { "id": 59, "penjual_id": 130, "nama_kios": "KATSUNYAA", "nama_bank": "BNI", "nomor_rekening": "123456789", "created_at": "2025-08-24T15:33:22.544Z", "deskripsi": "katsu-katsu kekinian", "gambar_kios": null }] </pre>	
TS-21	Cek fungsi getMenus ByKios	TC-21-01	Ambil menu kios berhasil	id = 1	Status 200 OK, daftar menu dari kios dengan ID = 1	<pre> 200 OK [{ "id": "142", "penjual_id": "136", "nama_menu": "bakso urat", "deskripsi": "bakso daging sapi", "harga": 12000, "foto_menu": "https://res.cloudinary.com/dnetoznik/image/upload/v1756508427/menus/wsnyhsx9bqchq8tnwk1n.jpg", "status_tersedia": true, "created_at": "2025-08-29T23:00:27.873Z", "kios_id": 62, "estimasi_menit": 5, "foto_public_id": "menus/wsnyhsx9bqchq8tnwk1n" }, { "id": "137", "penjual_id": "136", "nama_menu": "bakso cincang", </pre>	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
						"deskripsi": "bakso halalan toyiban", "harga": 12000, "foto_menu": "https://res.cloudinary.com/dnetoznik/image/upload/v1756508427/menus/wsnyhsx9bqchq8tnwk1n.jpg", "status_tersedia": true, "created_at": "2025-08-26T10:39:25.366Z", "kios_id": 62, "estimasi_menit": 5, "foto_public_id": "menus/diznfl8rva6oofjp18ah"	
		TC-21-02	Kios tidak punya menu	id = 2	Status 200 OK, array kosong	200 OK []	PASS
TS-22	Cek fungsi getKios ByPenjual	TC-22-01	Ambil kios penjual berhasil	Token valid	Status 200 OK, JSON { message: "Data kios berhasil diambil", data: {...} }	200 OK { "message": "Data kios berhasil diambil", "data": { "id": 62, "penjual_id": 136, "nama_kios": "Kantin Legi", "nama_bank": "BCA", "nomor_rekening": "1234567890", "created_at": "2025-08-26T09:04:07.354Z", "deskripsi": "warung halal", "gambar_kios": "https://res.cloudinary.com/dnetoznik/image/upload/v1756203117/kios/fkxsgyztsozs3cqzlavj.png" } }	PASS
		TC-22-02	Ambil kios penjual tanpa token	Request tanpa token	401 Unauthorized, message "Tidak ada ID penjual"	401 Unauthorized { "message": "Token tidak ditemukan" }	PASS
TS-23	Cek fungsi updateKios	TC-23-01	Update kios berhasil	{nama_kios: "Update Baru"}	200 OK, data kios terupdate	200 OK { "id": 62, "penjual_id": 136, "nama_kios": " Update Baru", "nama_bank": "BCA",	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
						"nomor_rekening": "1234567890", "created_at": "2025-08-26T09:04:07.354Z", "deskripsi": "warung halal", "gambar_kios": "https://res.cloudinary.com /dnetoznik/image/upload/v1756203117/kios/ fkxsgyztsozs3cqzlvj.png"	
		TC-23-02	ID kios bukan angka	/update/abc	400 Bad Request, message "ID kios harus berupa angka"	400 Bad Request { "message": "ID kios harus berupa angka" }	PASS
		TC-23-03	ID kios valid tapi tidak ada di DB	id=999	404 Not Found, message "Kios tidak ditemukan"	404 Not Found { "message": "Kios tidak ditemukan" }	PASS
TS-24	Cek fungsi getKios Detail	TC-24-01	Ambil detail kios berhasil	Request dengan kiosId valid	200 OK, JSON { message: "Detail kios berhasil diambil", data: {...} }	200 OK { "message": "Detail kios berhasil diambil", "data": { "id": 62, "nama_kios": "Kantin Legi", "deskripsi": "warung halal", "gambar_kios": "https://res.cloudinary.com/ dnetoznik/image/u pload/v1756203117/kios/ fkxsgyztsozs3cqzl avj.png" } }	PASS
		TC-24-02	kiosId tidak ada di DB	id = 999	404 Not Found, message "Kios tidak ditemukan"	404 Not Found { "message": "Kios tidak ditemukan" }	PASS

Tabel 5.45 Pengujian Keranjang

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
TS-25	Cek Fungsi tambah keranjang	TC-25-01	Tambah item baru berhasil	Header Cookie/Guest: ada, Body: {	201 Created , body memuat message: "Ditambahkan ke	201 Created {	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
				"menu_id": 101, "jumlah": 2, "catatan": "pedas" }	keranjang", dan berisi detail menu	"message": "Ditambahkan ke keranjang", "item": { "id": 773, "guest_id": "1", "menu_id": 154, "jumlah": 1, "total_harga": 20000, "catatan": "", "kios_id": 85, "nama_menu": "nasi udang enak", "harga": 20000, "foto_menu": "https://res.cloudinary.com/dn-etoznik/image/upload/v1757255620/menus/hsyqve0mkdacu5qhz4xe.jpg", "subtotal": 20000 }	
		TC-25-02	Tambah item dari kios lain	Header Guest ada, Body: { "menu_id": 202 } (menu kios_id≠kios di keranjang)	409 Conflict , message berisi larangan campur kios & existing_kios_id	409 Conflict { "message": "Keranjang Anda masih ada dari kios lain. Hapus dulu untuk menambah menu dari kios ini", "existing_kios_id": 85 }	PASS
		TC-25-03	Tanpa menu_id/guest_id	Body: {} atau header guest kosong	400 Bad Request , message: 'guest_id dan menu_id wajib dikirim'	400Bad Request { "message": "guest_id dan menu_id wajib dikirim" }	PASS
		TC-25-04	menu_id tidak ada di DB	Body: { "menu_id": 99999 }	404 Not Found , message: 'Menu tidak ditemukan'	404Not Found { "message": "Menu tidak ditemukan" }	PASS
TS-26	Cek Fungsi get keranjang	TC-26-01	Get keranjang sukses	Header Guest ada	200 OK , body memuat kios_id, array items (id, menu_id, kios_id, nama_menu, harga, foto_menu, jumlah, catatan, subtotal), dan total_harga = sum(subtotal). Header respon mengandung X-Buyer-Id	200 OK { "kios_id": 85, "items": [{ "id": 773, "menu_id": 154, "kios_id": 85, "nama_menu": "nasi udang enak", "harga": 20000, "foto_menu": "https://res.cloudinary.com/dn-etoznik/image/upload/v1757255620/menus/hsyqve0mkdacu5qhz4xe.jpg", "jumlah": 1, }	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
						<pre> "catatan": "", "subtotal": 20000 }], "total_harga": 20000 } </pre>	
		TC-26-02	Tanpa guest_id	Header Guest kosong	400 Bad Request , message: 'guest_id wajib dikirim'	<pre> 400 Bad request { "message": "guest_id wajib dikirim" } </pre>	PASS
TS-27	Cek Fungsi update keranjang	TC-27-01	Update jumlah naik(nambah)	Header Guest ada, Param: /keranjang/:id (id valid), Body: { "jumlah": 5 }	200 OK , message: 'Item diperbarui', jumlah=5, subtotal = harga*5	<pre> 200 OK { "message": "Item diperbarui", "item": { "id": 774, "guest_id": "1", "menu_id": 154, "jumlah": 2, "total_harga": 40000, "catatan": "tanpa pedas", "kios_id": 85, "nama_menu": "nasi udang enak", "harga": 20000, "foto_menu": "https://res.cloudinary.com/dnetoznik/image/upload/v1757255620/menus/hsyqve0mkdacu5qhz4xe.jpg", "subtotal": 40000 } } </pre>	PASS
		TC-27-02	Update jumlah ke 0	Body: { "jumlah": 0 }	200 OK , message: 'Item dihapus karena jumlah = 0'	<pre> 200 OK { "message": "Item dihapus karena jumlah = 0" } </pre>	PASS
		TC-27-03	ID bukan milik guest / tidak ada	Param: /keranjang/9999	404 Not Found , message: 'Item tidak ditemukan'	<pre> 404 Not Found { "message": "Item tidak ditemukan" } </pre>	PASS
TS-28	Cek Fungsi Hapus item keranjang	TC-28-01	Hapus sukses	Header Guest ada, Param: /keranjang/:id (valid)	200 OK , message: 'Item dihapus', body mengembalikan ringkasan item yang dihapus + subtotal	<pre> 200 OK { "message": "Item dihapus", "item": { "id": 774, "guest_id": "1", "menu_id": 154, "jumlah": 2, "total_harga": 40000, "catatan": "tanpa pedas", "kios_id": 85, "nama_menu": "nasi udang enak", </pre>	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
						<pre> "harga": 20000, "foto_menu": "https://res.cloudinary.com/dn etoznik/image/upload/v17572 55620/menus/hsyqve0mkdac u5qhz4xe.jpg", "subtotal": 40000 } </pre>	
		TC-28-02	ID tidak ada / bukan milik guest	Param: /keranjang/9999	404 Not Found , message: 'Item tidak ditemukan'	404 Not Found { "message": "Item tidak ditemukan" }	PASS

Tabel 5.46 Pengujian Pesanan

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
TS-29	Cek Fungsi notifyPenjual	TC-29-01	Kirim WA & log pertama kali	kiosId valid, pesananId baru	sendWaMessage dipanggil; insert ke wa_notif_logs; no error	📩 Pesanan Baru! Silakan klik tautan ini untuk melihat pesanan: https://pas-meal.vercel.app/	PASS
		TC-29-01	Notif hanya setelah pembayaran sukses	pesanan paid=belum (status pending/paid? sebelum settlement)	TIDAK ada WA ke penjual	WA terkirim	FAIL
TS-30	Cek Fungsi notifyPembeli Pesanan Selesai	TC-30-01	Kirim ringkasan selesai	pesanan done	WA terkirim dengan daftar item + total + tipe_pengantaran	Hai lina! 🥳 Pesanan kamu sudah selesai dan berhasil diterima. Berikut detail pesananmu: bakso cincang x1 = Rp12.000 Total: Rp12.000 Ambil sendiri di kantin Terima kasih sudah memesan di kantin Universitas Setiabudi! 😊 Selamat menikmati 🍲!	PASS
TS-31	Cek Fungsi buatPesanan	TC-31-01	Sukses buat pesanan dari keranjang	Header guest, keranjang 1 kios	201 body {message:'Pesanan berhasil dibuat', pesanan{...status:'pending'}}; pesanan_detail terisi; keranjang dikosongkan	201 Created { "message": "Pesanan berhasil dibuat", "pesanan": { "id": "389", "nama_pemesan": "cinta", "total_harga": 20000, "catatan": "gaada", "guest_id": "1", "tipe_pengantaran": "ambil_sendiri", } }	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
						<pre> "no_hp": "02887827363", "diantar_ke": null, "created_at": "2025-09-07T08:16:23.984Z", "payment_type": null, "payment_details": null, "status": "pending", "snap_token": null, "order_id": null, "total_estimasi": 10, "paid_at": null, "kios_id": 85, "estimasi_mulai_at": null, "estimasi_selesai_at": null, "delayed": false, "waktu_proses_mulai": null } </pre>	
		TC-31-02	Keranjang kosong	Keranjang tidak ada item	400 {message:'Keranjang kosong'}	400 Bad Request { "message": "Keranjang kosong" }	PASS
		TC-31-03	Data wajib tidak lengkap	Tanpa tipe_pengantaran/ nama_pemesan/ no_hp/ guest_id	400 {message:'Data wajib tidak lengkap.'}	400 Bad Request { "message": "Data wajib tidak lengkap." }	PASS
TS-32	Cek Fungsi getStatus Pesanan Guest	TC-32-01	Pesanan done	Param :id milik guest, status done	200 {status:'done', estimasi_selesai_at: nowISO}	200 OK { "status": "done", "estimasi_selesai_at": "2025-09-07T16:56:38.114Z" }	PASS
		TC-32-02	Sedang diproses → ada estimasi	waktu_proses_mulai terisi	estimasi_selesai_at = start + total_estimasi	200 OK { "status": "processing", "estimasi_selesai_at": "2025-09-07T10:04:05.914Z" }	PASS
		TC-32-03	Belum diproses	waktu_proses_mulai = null	estimasi_selesai_at = null	200 OK { "status": "paid", "estimasi_selesai_at": "null" }	PASS
		TC-32-04	Bukan milik guest	id valid tapi guest beda	404 {message:'Pesanan tidak ditemukan'}	404 Not Found { "message": "Pesanan tidak ditemukan" }	PASS
		TC-32-05	Recompute ETA setelah pesanan	dua pesanan @5 menit; pesanan #1 selesai lebih cepat	ETA pesanan #2 berkurang (start baru = selesai #1;	ETA tidak berubah	FAIL

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
			sebelumnya selesai		estimasi_selesai_at disesuaikan)		
TS-33	Cek Fungsi getPesanan ByGuest	TC-33-01	Paging page 1	?page=1, guest valid	200 {page,totalPages,limit:5,total,data[..]}urut created_at DESC	200 OK { "page": 1, "totalPages": 1, "limit": 5, "total": 1, "data": [{ "id": "389", "kios_id": 85, "tipe_pengantaran": "ambil_sendiri", "nama_pemesan": "cinta", "total_harga": 20000, "total_estimasi": 10, "status": "done", "created_at": "2025-09-07T08:16:23.984Z" }] }	PASS
TS-34	Cek Fungsi getDetail Pesanan	TC-34-01	Sukses + items + antrean	:id milik guest, status≠done	200 data pesanan + items (foto jadi URL penuh bila path) + antrean[{id,sisaWaktu>0}]	{ "id": "389", "nama_pemesan": "cinta", "total_harga": 20000, "catatan": "gaada", "guest_id": "4", "tipe_pengantaran": "ambil_sendiri", "no_hp": "09282722763", "diantar_ke": null, "created_at": "2025-09-07T08:16:23.984Z", "payment_type": "bank_transfer", "payment_details": [{ "bank": "bca", "va_number": "123456789" }], "status": "done", "snap_token": "9c53828f-a89c-4f1c-b1df-102444cd65ed", "order_id": "ORDER-389-1757263025827", "total_estimasi": 10, "paid_at": "2025-09-07T09:42:46.352Z", "kios_id": 85, "estimasi_mulai_at": "2025-09-07T09:54:05.914Z", }	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
						<pre> "estimasi_selesai_at": "2025-09-07T10:04:05.914Z", "delayed": false, "waktu_proses_mulai": "2025-09-07T09:54:05.914Z", "items": [{ "id": "525", "pesanan_id": "389", "menu_id": "154", "nama_menu": "nasi udang enak", "harga": "20000.00", "foto_menu": "https://res.cloudinary.com/dn etoznik/image/upload/v17572 55620/menus/hsyqve0mkdac u5qhz4xe.jpg", "jumlah": 1, "subtotal": "20000.00" }], "antrean": [] </pre>	
		TC-34-02	Unauthorized	guest tidak ada	404 {message: Pesanan tidak ditemukan atau Anda tidak memiliki akses }	404 Not Found { "message": "Pesanan tidak ditemukan atau Anda tidak memiliki akses" }	PASS
TS-35	Cek Fungsi getPesanan Masuk	TC-35-01	Sukses paging	Penjual valid ?page=1	200 {page,totalPages,limit:8,total,data[...]}; status dilabel via getStatusLabel	200 OK { "page": 1, "totalPages": 0, "limit": 8, "total": 0, "data": [] }	PASS
		TC-35-02	User ID tidak valid	Token tidak valid	403 {message: Token tidak valid atau kadaluarsa}	403 Forbidden { "message": "Token tidak valid atau kadaluarsa" }	PASS
TS-36	Cek Fungsi getDetail Pesanan Masuk	TC-36-01	Estimasi mulai & selesai dihitung	Antrean aktif ada, id valid	200 data + estimasi_mulai_at/estimasi_selesai_at (chain dari waktu_proses_mulai atau paid_at)	200 OK { "id": "389", "nomor_antrian": 1, "status_label": "Menunggu Diproses", "nama": "cinta", "no_hp": "092827226", "metode_bayar": "BANK_TRANSFER", "tipe_pengantaran": "Ambil Sendiri", "tanggal_bayar": "07 September 2025 16:42", "paid_at": "2025-09-07T09:42:46.352Z", }	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
						"catatan": "gaada", "total_harga": 20000, "total_estimasi": 10, "status": "paid", "menu": [{ "nama_menu": "nasi udang enak", "jumlah": 1, "harga": "20000.00", "foto_menu": "https://res.cloudinary.com/dn etoznik/image/upload/https://r es.cloudinary.com/dnetoznik/i mage/upload/v1757255620/ menus/hsyqve0mkdacu5qhz 4xe.jpg" }], "estimasi_mulai_at": "2025-09-07T09:54:05.914Z", "estimasi_selesai_at": "2025-09-07T10:04:05.914Z" 	
		TC-36-02	Tidak ada antrean	Tidak ada row aktif	404 {message:'Tidak ada pesanan aktif'}	404 Not Found { "message": "Tidak ada pesanan aktif" }	PASS
		TC-36-03	Id tidak ada di antrean	id tidak ditemukan	404 {message:'Pesanan tidak ditemukan dalam antrean aktif'}	404 Not Found { "message": "Pesanan tidak ditemukan dalam antrean aktif" }	PASS
TS-37	Cek Fungsi count Pesanan Masuk	TC-37-01	Hitung jumlah badge	Penjual valid	200 {jumlah: <int>} hanya status paid/processing/ready/delivering	200 OK { "jumlah": 1 }	PASS
TS-38	Cek Fungsi update Status Pesanan	TC-38-01	Update ke processing	Body {status:'processing'}	200 ; set status='processing', estimasi_mulai_at=NOW(), estimasi_selesai_at=NOW()+interval(total_estimasi), waktu_proses_mulai terisi, delayed=false	200 OK { "message": "Status berhasil diperbarui", "pesanan": { "id": "389", "nama_pemesan": "cinta", "total_harga": 20000, "catatan": "gaada", "guest_id": "1", "tipe_pengantaran": "ambil_sendiri", "no_hp": "082626252", "diantar_ke": null, "created_at": "2025-09-07T08:16:23.984Z", "payment_type": "bank_transfer", "payment_details": [	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
						<pre> { "bank": "bca", "va_number": "123456789" }], "status": "processing", "snap_token": "9c53828f-a89c-4f1c-b1df-102444cd65ed", "order_id": "ORDER-389-1757263025827", "total_estimasi": 10, "paid_at": "2025-09-07T09:42:46.352Z", "kios_id": 85, "estimasi_mulai_at": "2025-09-07T10:19:04.741Z", "estimasi_selesai_at": "2025-09-07T10:29:04.741Z", "delayed": false, "waktu_proses_mulai": "2025-09-07T10:19:04.741Z" } } </pre>	
		TC-38-02	Update ke ready/delivering	Body {status:'ready'}	200 ; hanya status berubah	<pre> 200 OK { "message": "Status berhasil diperbarui", "pesanan": { "id": "389", "nama_pemesan": "cinta", "total_harga": 20000, "catatan": "gaada", "guest_id": "1", "tipe_pengantaran": "ambil_sendiri", "no_hp": "08282722726", "diantar_ke": null, "created_at": "2025-09-07T08:16:23.984Z", "payment_type": "bank_transfer", "payment_details": [{ "bank": "bca", "va_number": "123456789" }], "status": "ready", "snap_token": "9c53828f-a89c-4f1c-b1df-102444cd65ed", "order_id": "ORDER-389-1757263025827", "total_estimasi": 10, </pre>	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
						<pre>"paid_at": "2025-09-07T09:42:46.352Z", "kios_id": 85, "estimasi_mulai_at": "2025-09-07T10:19:04.741Z", "estimasi_selesai_at": "2025-09-07T10:29:04.741Z", "delayed": false, "waktu_proses_mulai": "2025-09-07T10:19:04.741Z" }</pre>	
		TC-38-03	Selesaikan (done) + WA pembeli	Body {status:'done'}	200; status='done'; notifyPembeliPesananSelesai terpanggil	<pre>{ "message": "Status berhasil diperbarui", "pesanan": { "id": "389", "nama_pemesan": "cinta", "total_harga": 20000, "catatan": "gaada", "guest_id": "1", "tipe_pengantaran": "ambil_sendiri", "no_hp": "0827266262", "diantar_ke": null, "created_at": "2025-09-07T08:16:23.984Z", "payment_type": "bank_transfer", "payment_details": [{ "bank": "bca", "va_number": "123456789" }], "status": "done", "snap_token": "9c53828f-a89c-4f1c-b1df-102444cd65ed", "order_id": "ORDER-389-1757263025827", "total_estimasi": 10, "paid_at": "2025-09-07T09:42:46.352Z", "kios_id": 85, "estimasi_mulai_at": "2025-09-07T10:19:04.741Z", "estimasi_selesai_at": "2025-09-07T10:29:04.741Z", "delayed": false, "waktu_proses_mulai": "2025-09-07T10:19:04.741Z" } }</pre>	PASS
		TC-38-04	Status tidak valid	Body {status:'cancelled'}	400 {message:'Status tidak valid'}	400 Bad Request {	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
						"message": "Status tidak valid" }	
TS-39	Cek Fungsi getRiwayat Pesanan	TC-39-01	List riwayat done	Penjual valid ?page=1	200 {page,totalPages,limit:8, total,data[.status:'done'..]}; metode_bayar uppercase; tipe_pengantaran diringkas	200 OK { "page": 1, "totalPages": 1, "limit": 8, "total": 1, "data": [{ "id": "389", "kios_id": 85, "nama_pemesan": "cinta", "no_hp": "082726225", "tipe_pengantaran": "Ambil Sendiri", "diantar_ke": null, "payment_type": "bank_transfer", "total_harga": 20000, "total_estimasi": 10, "status": "done", "tanggal": "07 Sep 2025, 15:16", "metode_bayar": "BANK_TRANSFER" }] }	PASS
TS-40	Cek Fungsi getDetail Riwayat Pesanan	TC-40-01	Detail riwayat sukses	id milik penjual, status='done'	200 data lengkap + status_label, tanggal_bayar/selesai, menu URL Cloudinary penuh, nomor_antrian berdasar urutan paid_at	{ "id": "389", "nomor_antrian": 1, "status_label": "Selesai", "nama": "cinta", "no_hp": "0882662651", "metode_bayar": "BANK_TRANSFER", "tipe_pengantaran": "Ambil Sendiri", "tanggal_bayar": "07 September 2025 16:42", "tanggal_selesai": "07 September 2025 15:16", "catatan": "gaada", "total_harga": 20000, "total_estimasi": 10, "status": "done", "menu": [{ "nama_menu": "nasi udang enak", "jumlah": 1, "harga": "20000.00", "subtotal": "20000.00", }] }	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
						<pre> "foto_menu": "https://res.cloudinary.com/dnetoznik/image/upload/https://res.cloudinary.com/dnetoznik/image/upload/v1757255620/menus/hsyqve0mkdacu5qhz4xe.jpg" }] } </pre>	
		TC-40-02	Tidak ditemukan	Bukan milik penjual/ID salah	404 {message:'Riwayat pesanan tidak ditemukan atau bukan milik Anda'}	404 Not Found { "message": "Riwayat pesanan tidak ditemukan atau bukan milik Anda" }	PASS

Tabel 5.47 Pengujian Midtrans (Pembayaran)

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
TS-41	Cek Fungsi Create Transaction	TC-41-01	Sukses buat transaksi (pakai total_harga langsung)	Body: {guest_id:'g-123', total_harga:50000 }	200 OK {order_id, token, redirect_url}	{ "order_id": "ORDER-389-1757258627117", "token": "dbfb4aa2-1505-421b-aab6-0937f39ffa51", "redirect_url": "https://app.sandbox.midtrans.com/snap/v4/redirection/dbfb4aa2-1505-421b-aab6-0937f39ffa51" }	PASS
		TC-41-02	Validasi: tanpa guest_id	Body: { total_harga: 10000 }	400 {message:'guest_id wajib diisi'}	400 Bad request { "message": "guest_id wajib diisi" }	PASS
		TC-41-03	Validasi: total_harga tidak ada & items kosong	Body: {guest_id:'g-123'}	400 {message:'total_harga harus positif'}	400 Bad request { "message": "total_harga harus positif" }	PASS
		TC-41-04	pesanan_id tidak ditemukan	Body: {guest_id:'g-123', pesanan_id:99999 }	404 {message:'Pesanan tidak ditemukan'}	404 Not Found { "message": "Pesanan tidak ditemukan" }	PASS
TS-42	Cek Fungsi handle Notification	TC-42-01	Settlement → status paid & set paid_at	Notif: {order_id:'ORDER-77-1725252525', transaction_status:'settlement', payment_type:'qris'}; DB pesanan(77) ada	200 OK {message:'Notifikasi diproses'}; DB: status='paid', payment_type='qris', payment_details terisi	200 OK { "message": "Notifikasi diproses" }	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
					(VA/per mata/payment_details jika ada), paid_at ter-set NOW()		
		TC-42-02	Payment success → kirim notifikasi ke penjual	callback Midtrans transaction_status = settlement/capture	WA ke penjual terkirim + log dicatat	WA tidak terkirim ke penjual	FAIL

Tabel 5.48 Pengujian Dashboard data

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
TS-43	Cek Fungsi get Dashboard Data	TC-43-01	Hitung data untuk kios tertentu	User: {kios_id:10}; DB: 3 pesanan aktif, 12 menu, pendapatan selesai Rp150.000	{ totalPesanan:3, totalMenu:12, pendapatan:150000 }	200 OK { "totalPesanan": 0, "totalMenu": 1, "pendapatan": 20000 }	PASS

Tabel 5.49 Pengujian Search All

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
TS-44	Cek Fungsi searchAll	TC-44-01	Query kosong	req.query = {}	400 {message:"Query pencarian wajib diisi"}	400 Bad request { "message": "Query pencarian wajib diisi" }	PASS
		TC-44-02	Cari kios yang cocok	req.query = {query:"Kantin"}; di DB ada nama_kios="Kantin Legi"	Response kios berisi row Kantin Legi; menus kosong jika tidak ada menu cocok	200 OK { "kios": [{ "id": 85, "penjual_id": 158, "nama_kios": "Kantin Legi", "nama_bank": "BCA", "nomor_rekening": "1234567890", "created_at": "2025-09-07T07:12:07.616Z", "deskripsi": "kios barokah", "gambar_kios": "https://res.cloudinary.com/dnetoznik/image/upload/v1757256347/kios/jl2qu5sbnucye2ebfl8r.jpg" }], "menus": [] }	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
		TC-44-03	Cari menu yang cocok	req.query = {query:"Nasi Goreng"}; di DB ada menu.nama_menu="Nasi Goreng Spesial" di kios 5	Response menus berisi row Nasi Goreng Spesial dengan field tambahan kios_id, nama_kios	<pre> { "kios": [], "menus": [{ "id": "154", "penjual_id": "158", "nama_menu": "nasi udang enak", "deskripsi": "nasi+udang", "harga": 20000, "foto_menu": "https://res.cloudinary.com/dn-etoznik/image/upload/v1757255620/menus/hsyqve0mkdacu5qhz4xe.jpg", "status_tersedia": true, "created_at": "2025-09-07T07:33:40.955Z", "kios_id": 85, "estimasi_menit": 10, "foto_public_id": "menus/hsyqve0mkdacu5qhz4xe", "nama_kios": "Kantin Legi" }] } </pre>	PASS
		TC-44-04	Cari kios dan menu sekaligus	req.query = {query:"Ayam"}; DB ada kios "Ayam Bakar Mbok D" dan menu "Ayam Geprek"	Response kios ada row kios; menus ada row menu; keduanya bisa berisi hasil	<pre> { "id": "148", "penjual_id": "150", "nama_menu": "Ayam cabe ijo", "deskripsi": "Nasi, ayam, sambal ijo, dan kol goreng", "harga": 30000, "foto_menu": "https://res.cloudinary.com/dn-etoznik/image/upload/v1756720859/menus/ov5orsshewixif2oiwir.jpg", "status_tersedia": true, "created_at": "2025-09-01T03:01:00.616Z", "kios_id": 78, "estimasi_menit": 8, "foto_public_id": "menus/ov5orsshewixif2oiwir", "nama_kios": "Kios Nasi ayam" } </pre>	PASS

Kode Test Scenario	Test Scenario	Kode Test Case	Test Case	Test Data	Expected Result	Actual Result	Pass/Fail
						"id": "135", "penjual_id": "132", "nama_menu": "Ayam geprek", "deskripsi": "Nasi, ayam, sambal", "harga": 12000, "foto_menu": "https://res.cloudinary.com/dn-etoznik/image/upload/v1756200139/menus/aus1b3aqtentmajtgm8.jpg", "status_tersedia": true, "created_at": "2025-08-26T02:22:20.447Z", "kios_id": 60, "estimasi_menit": 2, "foto_public_id": "menus/aus1b3aqtentmajtgm8", "nama_kios": "Kios Syifa" },	
		TC-44-05	Tidak ada hasil sama sekali	req.query = {query:"xyz"}; DB tidak ada kios/menu cocok	Response { kios:[], menus:[] }	<pre>{ "kios": [], "menus": [] }</pre>	PASS

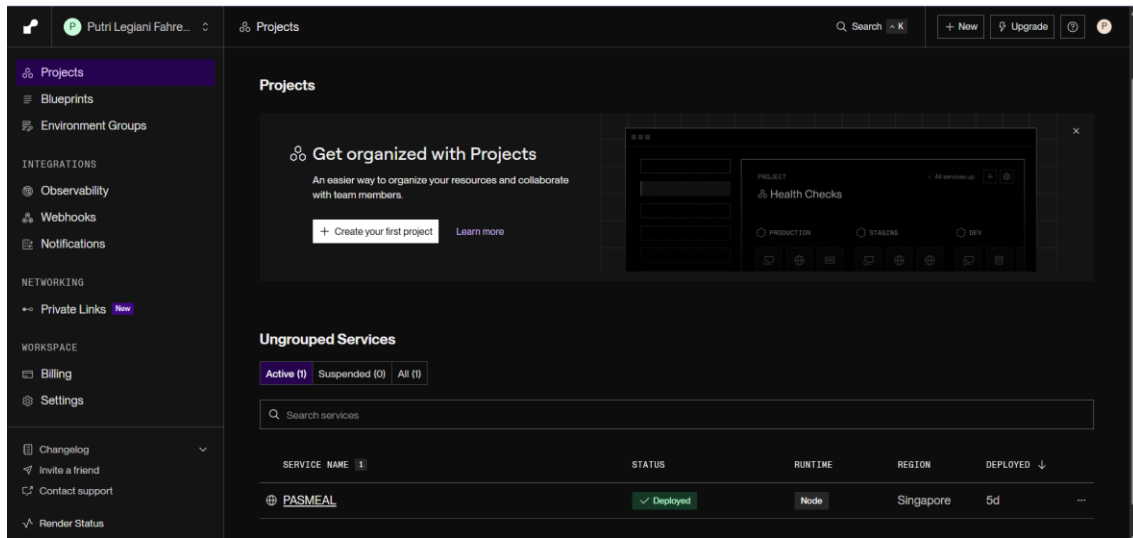
5.2.2 Deployment

Tahap deployment merupakan tahapan untuk menjalankan *server* agar dapat diakses secara *online* oleh pihak terkait (*frontend*) melalui base URL API. Pada penelitian ini, *deployment* PasMeal dilakukan menggunakan layanan Render sebagai *platform* hosting *server*, sedangkan Supabase digunakan sebagai layanan *database* berbasis PostgreSQL.

5.2.2.1 Persiapan Infrastruktur Server

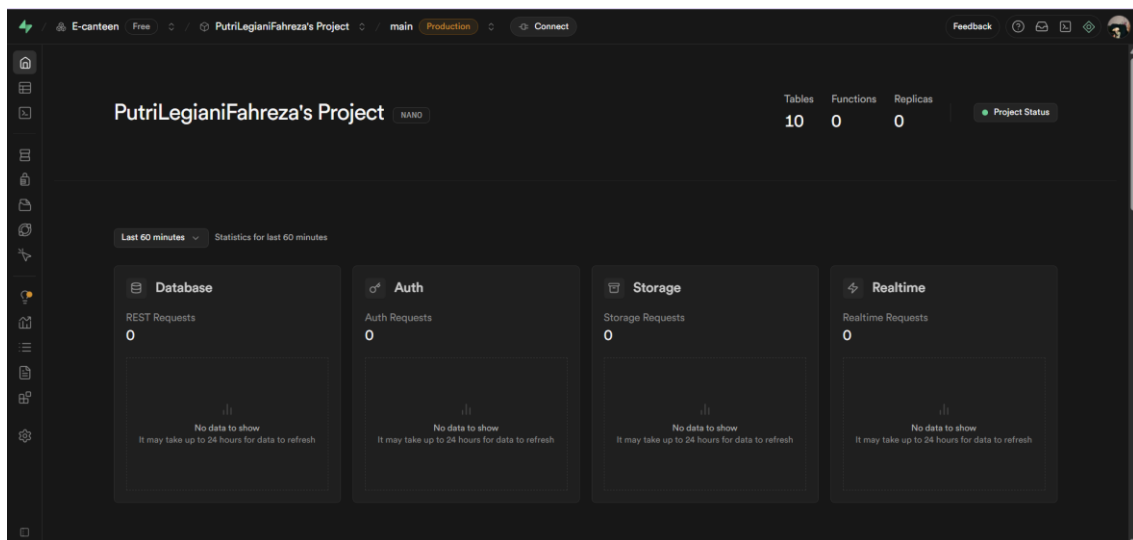
Infrastruktur untuk aplikasi PasMeal disiapkan dengan memanfaatkan layanan berikut :

1. Render (Web Service) : digunakan untuk menjalankan *backend* Node.js berbasis Express agar dapat diakses secara publik melalui domain yang disediakan Render.



Gambar 5.3 Render

2. Supabase (*Database*) : digunakan sebagai layanan *database* PostgreSQL yang terhubung langsung ke PasMeal. Supabase dipilih karena menyediakan layanan *database* yang stabil, *scalable*, dan mudah diintegrasikan dengan Node.js.



Gambar 5.4 Supabase

5.2.2.2 Alur Deployment

Proses *deployment* PasMeal dilakukan dengan memanfaatkan dua layanan utama, yaitu Supabase sebagai penyedia basis data PostgreSQL dan Render sebagai penyedia layanan web untuk *backend*. Adapun alur deployment dapat dijelaskan sebagai berikut:

1. *Deployment* Basis Data di Supabase
 - a. Membuat sebuah proyek baru pada layanan Supabase yang secara otomatis menyediakan sebuah basis data PostgreSQL.

- b. Supabase menyediakan *connection* string berupa *DATABASE_URL* yang berfungsi untuk menghubungkan aplikasi dengan basis data.
2. *Deployment* di Render
 - a. Kode diunggah terlebih dahulu ke GitHub agar dapat diintegrasikan dengan layanan Render.
 - b. Pada Render, dibuat sebuah Web Service dengan memilih repositori GitHub yang akan di *deploy*.
 - c. Konfigurasi yang dilakukan meliputi build command (npm install) dan start command (npm start), serta penambahan variabel lingkungan (environment variables) yang berisi data sensitif seperti *DATABASE_URL* (dari Supabase) dan kunci integrasi Midtrans.
 - d. Setelah konfigurasi selesai, Render akan membangun (build) dan menjalankan aplikasi *backend* sehingga dapat diakses melalui URL publik.
3. Mekanisme Pencegahan Sleep Pada Layanan Render
 - a. Karena penggunaan paket gratis (*free plan*) pada Render memiliki keterbatasan berupa auto sleep ketika tidak ada permintaan (*request*) dalam jangka waktu tertentu, maka dibuat sebuah mekanisme otomatis untuk menjaga agar layanan tetap aktif.
 - b. Mekanisme ini dibuat dengan menambahkan workflow pada GitHub Actions yang secara berkala melakukan ping (*request*) ke URL setiap 10 menit. Dengan demikian, layanan *backend* tidak mengalami sleep dan tetap responsif ketika diakses oleh pengguna.

Dengan alur deployment ini, aplikasi PasMeal dapat berjalan secara *online* dengan URL yang stabil, terkoneksi ke *database*, dan siap diintegrasikan oleh tim *frontend*.

5.2.2.3 Hasil Deployment

Setelah *deployment* berhasil, API untuk *e-canteen* PasMeal dapat diakses secara publik melalui *base* URL berikut.

Tabel 5.50 Hasil Deployment

Environment	URL
Production	https://pasmeal.onrender.com/

Dengan dapat diaksesnya *base* URL tersebut, proses *deployment* untuk aplikasi *e-canteen* PasMeal dinyatakan telah berhasil. *Server* telah berjalan di lingkungan *production* dan seluruh *endpoint* API siap untuk menerima koneksi serta melayani permintaan dari aplikasi *frontend*.

BAB 6

KESIMPULAN DAN SARAN

Bab ini menjelaskan kesimpulan yang dapat diambil dari hasil penelitian dan pengembangan yang telah dilakukan serta saran yang diberikan untuk pengembangan selanjutnya.

6.1 Kesimpulan

Berdasarkan perancangan dan implementasi yang telah dilakukan, diperoleh kesimpulan sebagai berikut:

1. Tugas akhir ini berhasil merancang dan mengimplementasikan API PasMeal menggunakan Node.js dengan *framework* Express.js serta PostgreSQL sebagai sistem manajemen basis data.
2. Arsitektur sistem disusun dengan pola berlapis (Routes → *Middleware* → Controller → Services → Database) sehingga tanggung jawab setiap komponen lebih terstruktur, mudah diuji, dan mendukung pengembangan berkelanjutan.
3. Penerapan metodologi Scrum pada proses pengembangan memungkinkan pengerjaan berjalan secara iteratif melalui sprint. Setiap sprint menghasilkan inkrement fungsional yang dapat ditinjau melalui sesi sprint review dan disesuaikan dengan kebutuhan baru yang muncul. Dengan pendekatan ini, backlog dapat dikelola dengan baik dan tujuan pengembangan tercapai secara bertahap.

6.2 Saran

Berdasarkan hasil implementasi dan evaluasi, beberapa saran pengembangan untuk penelitian selanjutnya adalah:

1. Mengoptimalkan fungsi notifikasi pesanan agar hanya terkirim setelah pembayaran berhasil dilakukan. Hal ini penting untuk mencegah penjual menerima pesanan yang belum valid dan menjaga keakuratan alur transaksi.
2. Menambahkan mekanisme pembaruan estimasi waktu secara otomatis ketika pesanan awal telah selesai diproses. Dengan demikian, *countdown* ETA dapat disesuaikan secara *real-time* sehingga lebih akurat mencerminkan kondisi antrean pesanan.
3. Menambahkan estimasi waktu pengantaran pesanan (*Delivery* ETA). Sistem dapat menghitung estimasi waktu pengantaran secara dinamis dengan mempertimbangkan estimasi masak per menu, jumlah antrean di kios, tipe pengantaran, dan jam operasional. Hal ini akan meningkatkan transparansi layanan bagi pembeli.
4. Membangun sistem admin *multi-tenant* untuk universitas. Dibutuhkan admin panel khusus untuk pihak kampus guna mengelola tenant/kios. Fitur yang disarankan meliputi registrasi dan verifikasi tenant, pengaturan profil kios (rekening, jam operasional, deskripsi), manajemen biaya layanan, pemantauan

transaksi, serta kemampuan suspend atau aktivasi kios. Implementasi *role-based access control* (RBAC) perlu dipertimbangkan agar hak akses lebih terstruktur.

5. Memisahkan histori pendapatan dari data menu agar catatan transaksi tidak hilang meskipun menu sudah dihapus.

DAFTAR PUSTAKA

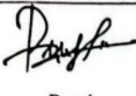
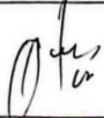
- [ANT22] Antony, C., Biju, A., Kuriakose, A., Mathew, A., & Jismon, M.U., “Canteen Food Ordering and Managing System”, *International Journal of Current Science Research and Review*, Volume 05, Issue 06, June 2022
- [ARH16] Arhandi, P.P., “Pengembangan Sistem Informasi Perijinan Tenaga Kesehatan Dengan Menggunakan Metode Back End Dan Front End.”, *Jurnal Teknologi Informasi: Teori, Konsep, dan Implementasi*, vol. 7, no. 1, 2016.
- [ARF19] Arfiyanti, I., Rija, T., & Hakim, D.S., “Membangun Aplikasi Mancoring Berbasis Android Dengan Menerapkan Rest Api Studi Kasus Pada Pt Telkom Akses Area Samarinda”, *Jurnal Informatika Wicida*, Vol. 8, No. 1, Januari 2019
- [BAC24] Bachtiar, Damara., “Perancangan *Backend* Pada Website Smart Campus Menggunakan Golang (Studi Kasus : Pt.360 Solusi Teknologi)”, tersedia : Februari 2025, <http://repository.unpas.ac.id/72183/> , Oktober 2024
- [HER22] Herdiyatomoko, H.F., “Desain Sistem *Backend* Berbasis REST API Menggunakan Framework Laravel 7.”, *SKANIKA: Sistem Komputer dan Teknik Informatika*, Volume 5, Nomor 2, Juli 2022
- [KUR20] Kurniawan, I., Humaira, & Rozi, F., “REST API Menggunakan NodeJS pada Aplikasi Transaksi Jasa Elektronik Berbasis Android”, *Jurnal Ilmiah Teknologi Sistem Informasi*, Volume 1 No 4, Desember 2020
- [KUR23] Kurnia, B., & Tumini, “Perancangan Sistem Informasi Pembelajaran *Online* (E-Learning) Berbasis Web.”, *Jurnal Ilmiah Teknik Informatika dan Komunikasi*, Vol. 3 No. 3, November 2023
- [LUB20] Lubis, M.D.S., Waruwu, T.S., & Lase, D., “PERANCANGAN DAN PEMBUATAN APLIKASI PEMESANAN MAKANAN *ONLINE* BERBASIS ANDROID”, *Jurnal Mahajana Informasi*, Vol. 5, No. 1, Juni 2020
- [NUR17] Nurjanah, Ria., “Perancangan Sistem Informasi Kantin Menggunakan Customer Relationship Management (Studi Kasus: Kantin Fakultas Teknik Universitas Pasundan)”, tersedia : Februari 2025, <http://repository.unpas.ac.id/15654/> , Jan 2017.
- [NOV21] Novitasari, L., “Sistem Informasi Pemesanan Berbasis Web Pada Sop Iga Sapi Rp. 15.000.”, tersedia : Februari 2025, https://elibrary.unikom.ac.id/id/eprint/6385/8/UNIKOM_Lula%20Novitasari_BAB%20II.pdf , Aug 2022
- [OSK23] Oskar & Noviyanti, “Perancangan Sistem Pemesanan Makanan Berbasis Website Pada Kantin (Institut Shanti Bhuna) ”, *INSTINK: (Jurnal Inovasi Pendidikan, Teknologi Informasi & Komputer)*, Vol.2, No.2, Oktober 2023
- [OCT19] Octavianus, C.R., dan Alexandra, L., “PENGARUH DESAIN KANTIN TERHADAP FLEKSIBILITAS KEGIATAN MAHASISWA UNIVERSITAS MATANA”, *Jurnal Ilmiah Penelitian Marka*, Jilid 2, Nomor 2, 2019.
- [PRI24] Priyowittesa, K.D., Sulisty, D., & Selviandro, N., “Perancangan dan Pengimplementasian Sistem *Backend* Aplikasi Cafeasy (Studi Kasus: Kafe daerah Bandung), *e-Proceeding of Engineering*, Vol.11, No.4 Agustus 2024
- [RIZ14] Rizkyana, Muhammad, Agung., & Amalga, R., Sandhika, Galih., “Rancangan Arsitektur Aplikasi Pengumpulan Tugas Dengan Push Notification Real-time Menggunakan Node.js”, *Seminar Nasional Informatika 2014 (semnasIF 2014)*, 12 Agustus 2014
- [RIF24] Rizqi & Rifda, N., “Sistem Informasi Pemesanan Kedai Kopi Berbasis Website.”, tersedia : Februari 2025, <http://eprints.poltektegal.ac.id/4377/3/BAB%202.pdf> , Sep 2024
- [SAL21] Salim, A., & Ishaq, C., “Pembuatan *Backend* aplikasi Pemandu Pariwisata Menggunakan Framework laravel Untuk Obyek Wisata Di Provinsi Jawa Barat”, *TEMATIK -Jurnal Teknologi Informasi Dan Komunikasi*, Vol. 7, No. 2, Desember 2021

- [SET25] Setiawan, M.F., Novitasari, R.F., & Baehaqie, L., “KantinKita : Sistem Pemesanan Makanan Berbasis Website”, *Seminar Nasional Teknologi & Sains (STAINS)*, Vol. 4 No. 1, 25 Januari 2025
- [SCH20] Schwaber, K., dan Sutherland, J., “The Scrum Guide”, Scrum.org, Burlington, MA, 2020.
- [SHA17] Shah, H., & Soomro, T.R., “Node.js Challenges in Implementation”, *Global Journal of Computer Science and Technology: E Network, Web & Security*, vol. 17, Issue 2, Version 1.0, Juni 2017
- [SCR25] SCRUMstudy™, "A Guide to the SCRUM BODY OF KNOWLEDGE (SBOK® Guide)" Fifth Edition, SCRUMstudy™, a brand of VMEdU, Inc.12725 W. Indian School Road, Suite F-112 Avondale, Arizona 85392 USA, 2025.
- [SYA14] Prakoso & Syaifullah, A., “Aplikasi Penanganan Transaksi Order Pada PT. Allib Indonesia.”, tersedia : Februari 2025,
https://repository.dinamika.ac.id/id/eprint/1285/5/BAB_III.pdf , Sep 2015.
- [WIJ17] Wijaya, E., “Pengenalan JavaScript.”, tersedia : Februari 2025,
<https://repository.unikom.ac.id/45420/> , Mar 2017
- [WID21] Widyoutomo, F., Ajie, H., & Widodo, “Pengembangan Web Servicemodul Mahasiswa Pada Sistem Informasi Akademik Universitas Negeri Jakarta.”, *Jurnal PINTER*, Vol. 5 No. 1, 2021
- [WOR02] Worsley, J.C. dan Drake, J.D., “Practical PostgreSQL”, First Edition, O’Reilly & Associates, Sebastopol, 2002.

LAMPIRAN A

BERITA ACARA WAWANCARA

LAMPIRAN A BERITA ACARA WAWANCARA

 Berita Acara Wawancara Tugas Akhir Program Studi Teknik Informatika, Fakultas Teknik, Universitas Pasundan			
Studi Kasus Pemesanan di Kantin Unpas Setiabudi Wawancara Penjual			
Hari, tanggal : Rabu, 26.02, 2025 Nama : Alclo Setiawan (Dapur Uje - 3g)			
Pertanyaan			
1. Bagaimana cara pelanggan memesan makanan di tempat Anda saat ini? (secara langsung/chat)			
Jawab : banyak chat			
2. Apakah pernah terjadi kesalahan dalam mencatat pesanan, seperti pesanan tertukar atau tidak tercatat? Jika iya, apakah pelanggan ada yang protes?			
Jawab : Salah bikin sama tertukar			
3. Apakah ada kesulitan saat mengantarkan pesanan, seperti menemukan nomor meja tempat pelanggan duduk?			
Jawab : Pelanggan numgu			
4. Ketika pelanggan sedang ramai, bagaimana Anda mengatur antrean pemesanan? Misalnya, jika pelanggan pertama memesan dalam jumlah banyak dan pelanggan berikutnya hanya memesan sedikit, pesanan mana yang akan dibuat lebih dulu?			
Jawab : Yang pertama / digabungin			
5. Biasanya, butuh waktu berapa lama untuk menyiapkan satu pesanan?			
Jawab : 15 - 20 menit			
6. Apakah pelanggan sering bertanya kapan pesanan mereka akan siap?			
Jawab : gada			
7. Saat ini, bagaimana Anda mencatat dan menyimpan daftar pesanan pelanggan ketika sedang ramai? Apakah dicatat di kertas, atau hanya diingat saja?			
Jawab : di catet			
8. Apakah Anda tertarik jika dibuatkan website atau aplikasi untuk mempermudah mengelola pemesanan?			
Jawab :			
Peneliti		Narasumber	
 Syifa	 Risma	 Putri	



Berita Acara Wawancara Tugas Akhir

Program Studi Teknik Informatika, Fakultas Teknik, Universitas Pasundan

Studi Kasus Pemesanan di Kantin Unpas Setiabudi

Wawancara Penjual

Hari, tanggal : Rabu, 26, 02, 2025

Nama : Ria (wawancara mamah obby - 49)

Pertanyaan

1. Bagaimana cara pelanggan memesan makanan di tempat Anda saat ini?
(secara langsung/chat)

Jawab : langsung

2. Apakah pernah terjadi kesalahan dalam mencatat pesanan, seperti pesanan tertukar atau tidak tercatat? Jika iya, apakah pelanggan ada yang protes?

Jawab : Pernah

3. Apakah ada kesulitan saat mengantarkan pesanan, seperti menemukan nomor meja tempat pelanggan duduk?

Jawab : jarang

4. Ketika pelanggan sedang ramai, bagaimana Anda mengatur antrean pemesanan? Misalnya, jika pelanggan pertama memesan dalam jumlah banyak dan pelanggan berikutnya hanya memesan sedikit, pesanan mana yang akan dibuat lebih dulu?

Jawab : berurutan

5. Biasanya, butuh waktu berapa lama untuk menyiapkan satu pesanan?

Jawab : 1 menit

6. Apakah pelanggan sering bertanya kapan pesanan mereka akan siap?

Jawab : jarang

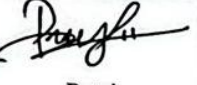
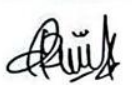
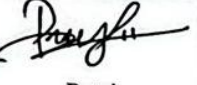
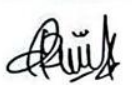
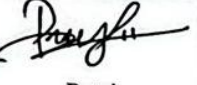
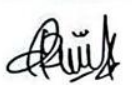
7. Saat ini, bagaimana Anda mencatat dan menyimpan daftar pesanan pelanggan ketika sedang ramai? Apakah dicatat di kertas, atau hanya diingat saja?

Jawab : di hafal

8. Apakah Anda tertarik jika dibuatkan website atau aplikasi untuk mempermudah mengelola pemesanan?

Jawab : Iya boleh

Peneliti			Narasumber
 Syifa	 Risma	 Putri	

	Berita Acara Wawancara Tugas Akhir Program Studi Teknik Informatika, Fakultas Teknik, Universitas Pasundan Studi Kasus Pemesanan di Kantin Unpas Setiabudi Wawancara Penjual								
Hari, tanggal : <u>Rabu, 26, 02, 2025</u> Nama : <u>Diva (50 - Salel Drinks)</u>									
Pertanyaan									
1. Bagaimana cara pelanggan memesan makanan di tempat Anda saat ini? (secara langsung/chat)	Jawab : <u>gaada chat</u>								
2. Apakah pernah terjadi kesalahan dalam mencatat pesanan, seperti pesanan tertukar atau tidak tercatat? Jika iya, apakah pelanggan ada yang protes?	Jawab : <u>Pernah salah bikin</u>								
3. Apakah ada kesulitan saat mengantarkan pesanan, seperti menemukan nomor meja tempat pelanggan duduk?	Jawab : <u>Sering bingung</u>								
4. Ketika pelanggan sedang ramai, bagaimana Anda mengatur antrean pemesanan? Misalnya, jika pelanggan pertama memesan dalam jumlah banyak dan pelanggan berikutnya hanya memesan sedikit, pesanan mana yang akan dibuat lebih dulu?	Jawab : <u>Yang Pertama</u>								
5. Biasanya, butuh waktu berapa lama untuk menyiapkan satu pesanan?	Jawab : <u>2 menit</u>								
6. Apakah pelanggan sering bertanya kapan pesanan mereka akan siap?	Jawab : <u>mgga karena cepet</u>								
7. Saat ini, bagaimana Anda mencatat dan menyimpan daftar pesanan pelanggan ketika sedang ramai? Apakah dicatat di kertas, atau hanya diingat saja?	Jawab : <u>di inget</u>								
8. Apakah Anda tertarik jika dibuatkan website atau aplikasi untuk mempermudah mengelola pemesanan?	Jawab : <u>Iya</u>								
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th colspan="3" style="text-align: center;">Peneliti</th> <th style="text-align: center;">Narasumber</th> </tr> </thead> <tbody> <tr> <td style="text-align: center; vertical-align: bottom;">  Syifa </td> <td style="text-align: center; vertical-align: bottom;">  Risma </td> <td style="text-align: center; vertical-align: bottom;">  Putri </td> <td style="text-align: center; vertical-align: bottom;">  </td> </tr> </tbody> </table>		Peneliti			Narasumber	 Syifa	 Risma	 Putri	
Peneliti			Narasumber						
 Syifa	 Risma	 Putri							



Berita Acara Wawancara Tugas Akhir

Program Studi Teknik Informatika, Fakultas Teknik, Universitas Pasundan

Studi Kasus Pemesanan di Kantin Unpas Setiabudi

Wawancara Pelanggan

Hari, tanggal : Selasa, 25/02/2025.

Nama : M. Luning Abid K.

Pertanyaan

1. Apakah Anda pernah mengalami kesalahan dalam pemesanan (misalnya pesanan salah atau terlambat)?

Jawab : Pernah

2. Seberapa sering Anda merasa harus menunggu lama untuk mendapatkan pesanan?

Jawab : Cukup Sering.

3. Apakah Anda ingin mendapatkan estimasi waktu yang lebih akurat saat menunggu makanan siap?

Jawab : tentu, Ingin.

4. Menurut Anda, apakah website pemesanan bisa membantu membuat proses pemesanan lebih cepat dan mudah?

Jawab: Ya, apps bisa membantu

5. Apakah Anda lebih suka melakukan pemesanan online atau langsung datang ke kantin?

Jawab : Secara pribadi, jika ada platform nya saya lebih suka online.

6. Fitur apa yang paling Anda inginkan dalam sistem pemesanan online?

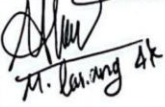
Jawab : Mungkin fitur yang ada di gofood.

7. Jika pemesanan dilakukan melalui website, apakah Anda merasa nyaman dengan pembayaran digital?

Jawab : tentu, nyaman & saya.

8. Bagaimana menurut Anda jika ada sistem yang memungkinkan Anda memesan makanan dulu, lalu mengambilnya saat sudah siap tanpa perlu menunggu lama?

Jawab : Jika hal itu diterapkan di kantin kampus, itu sangat bagus.

Peneliti			Narasumber
 Syifa	 Risma	 Putri	 M. Luning Abid K.

	Berita Acara Wawancara Tugas Akhir Program Studi Teknik Informatika, Fakultas Teknik, Universitas Pasundan Studi Kasus Pemesanan di Kantin Unpas Setiabudi Wawancara Pelanggan								
Hari, tanggal : 25 Februari 2025, Selasa Nama : M. Rivan Sahronie									
Pertanyaan									
1. Apakah Anda pernah mengalami kesalahan dalam pemesanan (misalnya pesanan salah atau terlambat)?	Jawab : Pernah								
2. Seberapa sering Anda merasa harus menunggu lama untuk mendapatkan pesanan?	Jawab : tidak terlalu sering								
3. Apakah Anda ingin mendapatkan estimasi waktu yang lebih akurat saat menunggu makanan siap?	Jawab : Belum								
4. Menurut Anda, apakah website pemesanan bisa membantu membuat proses pemesanan lebih cepat dan mudah?	Jawab: Bisa, tergantung dari sistem yang dibuat								
5. Apakah Anda lebih suka melakukan pemesanan online atau langsung datang ke kantin?	Jawab : online								
6. Fitur apa yang paling Anda inginkan dalam sistem pemesanan online?	Jawab : Pembayaran secara digital								
7. Jika pemesanan dilakukan melalui website, apakah Anda merasa nyaman dengan pembayaran digital?	Jawab : Iya.								
8. Bagaimana menurut Anda jika ada sistem yang memungkinkan Anda memesan makanan dulu, lalu mengambilnya saat sudah siap tanpa perlu menunggu lama?	Jawab : Fitur yg bagus								
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <th colspan="3" style="text-align: center;">Peneliti</th> <th style="text-align: center;">Narasumber</th> </tr> <tr> <td style="text-align: center; vertical-align: bottom;">  Syifa </td> <td style="text-align: center; vertical-align: bottom;">  Risma </td> <td style="text-align: center; vertical-align: bottom;">  Putri </td> <td style="text-align: center; vertical-align: bottom;">  </td> </tr> </table>		Peneliti			Narasumber	 Syifa	 Risma	 Putri	
Peneliti			Narasumber						
 Syifa	 Risma	 Putri							



Berita Acara Wawancara Tugas Akhir

Program Studi Teknik Informatika, Fakultas Teknik, Universitas Pasundan

Studi Kasus Pemesanan di Kantin Unpas Setiabudi

Wawancara Pelanggan

Hari, tanggal : Selasa, 25-02-2025

Nama : Widya Dwi Indrianti

Pertanyaan

1. Apakah Anda pernah mengalami kesalahan dalam pemesanan (misalnya pesanan salah atau terlambat)?

Jawab : Pernah

2. Seberapa sering Anda merasa harus menunggu lama untuk mendapatkan pesanan?

Jawab : Cukup sering

3. Apakah Anda ingin mendapatkan estimasi waktu yang lebih akurat saat menunggu makanan siap?

Jawab : Iya

4. Menurut Anda, apakah website pemesanan bisa membantu membuat proses pemesanan lebih cepat dan mudah?

Jawab : Iya bisa

5. Apakah Anda lebih suka melakukan pemesanan online atau langsung datang ke kantin?

Jawab : Pemesanan online, karena tidak memakan waktu

6. Fitur apa yang paling Anda inginkan dalam sistem pemesanan online?

Jawab : pemulihan menu makanan / minuman

7. Jika pemesanan dilakukan melalui website, apakah Anda merasa nyaman dengan pembayaran digital?

Jawab : Iya, karena pembayaran digital lebih simple

8. Bagaimana menurut Anda jika ada sistem yang memungkinkan Anda memesan makanan dulu, lalu mengambilnya saat sudah siap tanpa perlu menunggu lama?

Jawab : menurut saya akan sangat berguna

Peneliti			Narasumber
 Syifa	 Risma	 Putri	

	Berita Acara Wawancara Tugas Akhir Program Studi Teknik Informatika, Fakultas Teknik, Universitas Pasundan								
Studi Kasus Pemesanan di Kantin Unpas Setiabudi Wawancara Penjual									
Hari, tanggal : Kamis, 20 Februari 2025 Nama : Yayan Dwi Yana									
Pertanyaan									
1. Bagaimana kondisi kantin UNPAS sebelum dilakukan perubahan dan siapa yang mengelolanya?	Jawab : Sebelumnya, kantin masih kurang terawat dan dikelola oleh Keperasi Warga UNPAS.								
2. Setelah dikelola universitas, apa langkah utama yang dilakukan untuk memperbaiki kantin?	Jawab : Universitas melakukan renovasi bangunan agar lebih nyaman dan layak digunakan.								
3. Saat ini berapa jumlah kios yang tersedia dan siapa saja yang diprioritaskan dalam penyewaan?	Jawab : Ada sekitar 50 kios. Prioritas pertama untuk pedagang lama, kemudian pensiunan staf/dosen, serta masyarakat sekitar kampus.								
4. Bagaimana kondisi kantin sekarang setelah dikelola universitas?	Jawab : Kantin lebih nyaman, terawat, dan mendukung kebutuhan mahasiswa, dosen, staf, serta warga sekitar.								
5. Apakah saat ini sudah ada sistem digital yang digunakan dalam pengelolaan kantin?	Jawab : Belum, semua masih dilakukan secara manual.								
6. Bagaimana pendapat Bapak jika diterapkan sistem pemesanan makanan online di kantin UNPAS??	Jawab : Sangat mendukung, bisa memudahkan mahasiswa dalam memesan makanan dan penjual dalam mengelola pesanan.								
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <th colspan="3" style="text-align: center;">Peneliti</th> <th style="text-align: center;">Narasumber</th> </tr> <tr> <td style="text-align: center; vertical-align: middle;">  Syifa </td> <td style="text-align: center; vertical-align: middle;">  Risma </td> <td style="text-align: center; vertical-align: middle;">  Putri </td> <td style="text-align: center; vertical-align: middle;">  Yayan Dwi Yana </td> </tr> </table>		Peneliti			Narasumber	 Syifa	 Risma	 Putri	 Yayan Dwi Yana
Peneliti			Narasumber						
 Syifa	 Risma	 Putri	 Yayan Dwi Yana						

LAMPIRAN B DOKUMENTASI

LAMPIRAN B DOKUMENTASI



LAMPIRAN C

DOKUMENTASI SPRINT

LAMPIRAN C DOKUMENTASI SPRINT

