PENGEMBANGAN WEBSITE E-COMMERCE FURNITUR BERBASIS LARAVEL



Disusun Oleh:

N a m a : Muhammad Ali Zidan Laveda
NIM : 20523066

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
2025

HALAMAN PENGESAHAN DOSEN PEMBIMBING

PENGEMBANGAN WEBSITE E-COMMERCE FURNITUR BERBASIS LARAVEL

TUGAS AKHIR

Disusun Oleh:

N a m a : Muhmmad Ali Zidan Laveda
NIM : 20523066

Yogyakarta, 26 Juni 2025
Pembimbing,

(Dhomas Hatta Fudholi, S.T., M.Eng., PH.D.)

HALAMAN PENGESAHAN DOSEN PENGUJI

PENGEMBANGAN WEBSITE E-COMMERCE FURNITUR BERBASIS LARAVEL

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika – Program Sarjana di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 26 Juni 2025

Tim Penguji

Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D.

Anggota 1

Arrie Kurniawardhani, S.Si., M.Kom.

Anggota 2

Sri Mulyati, S.Kom., M.Kom.

Mengetahui,

Ketua Program Studi Informatika – Program Sarjana

Fakultas Teknologi Industri

Universitas Islam Indonesia

(Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Muhammad Ali Zidan Laveda

NIM : 20523066

Tugas akhir dengan judul:

PENGEMBANGAN WEBSITE E-COMMERCE FURNITUR BERBASIS LARAVEL

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila di kemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung risiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 26 Juni 2025



(Muhamad Ali Zidan Laveda)

HALAMAN PERSEMBAHAN

Alhamdulillahi Robbil'alamin, puji dan syukur ke hadirat Allah SWT atas segala nikmat yang telah diberikan sehingga penulis dapat menyelesaikan tugas akhir ini dengan baik. Dengan penuh rasa hormat, penulis mempersembahkan tugas akhir ini kepada :

1. Kedua orang tuaku tercinta, yang selalu menjadi sumber semangat, doa yang tiada henti, dan kasih sayang tanpa batas. Terima kasih atas segala dukungan, pengorbanan, dan cinta yang tak ternilai harganya.
2. Kakak dan Adik tercinta, yang selalu mendukung dan mendoakan dalam setiap langkah perjalanan hidup ini.
3. Dosen pembimbing, Bapak Dhomas Hatta Fudholi, S.T., M.Eng., PH.D., yang telah membimbing, mendidik, dan memberikan ilmu yang sangat berharga selama pengerjaan tugas akhir ini.
4. Teman-teman seperjuangan dan seluruh pihak yang telah membantu, baik secara langsung maupun tidak langsung, dalam menyelesaikan skripsi ini. Terima kasih atas kebersamaan, motivasi, dan dukungan yang tak terlupakan.
5. Diriku sendiri, yang telah berjuang melewati tantangan demi tantangan hingga akhirnya mampu menyelesaikan karya ilmiah ini.

Semoga skripsi ini menjadi langkah awal untuk kontribusi nyata di bidang keilmuan dan bermanfaat bagi banyak pihak.

HALAMAN MOTO

"Keraslah pada dirimu maka dunia akan lembut kepadamu, tetapi jika kamu lembut pada dirimu maka dunia akan keras kepadamu"

KATA PENGANTAR

Puji syukur penulis panjatkan ke hadirat Allah SWT atas segala limpahan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan penyusunan laporan Tugas Akhir yang berjudul "**PENGEMBANGAN WEBSITE E-COMMERCE FURNITUR BERBASIS LARAVEL"** dengan baik dan lancar. Laporan ini disusun sebagai salah satu syarat untuk memperoleh gelar Sarjana pada Program Studi Informatika, Universitas Islam Indonesia.

Dalam proses penyusunan laporan ini, penulis menerima banyak bantuan dan dukungan dari berbagai pihak. Oleh karena itu, dengan penuh rasa hormat dan terima kasih, penulis menyampaikan apresiasi kepada:

1. Allah SWT, atas rahmat dan petunjuk-Nya selama proses penyusunan Tugas Akhir ini.
2. Kedua orang tua tercinta, Isbat dan Dyah Utari Sintowati, kakak saya Dhia Salsabila Zahra, dan adik saya Queen Reva labibah yang selalu mendoakan, memberikan semangat, dan menjadi sumber kekuatan utama dalam perjalanan akademik penulis.
3. Prof. Fathul Wahid, S.T., M.Sc., Ph.D., selaku Rektor Universitas Islam Indonesia
4. Bapak Dr. Raden Teduh Dirgahayu, S.T., M.Sc., selaku Ketua Jurusan Informatika Universitas Islam Indonesia.
5. Bapak Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D., selaku Ketua Program Studi Informatika Program Sarjana Universitas Islam Indonesia dan dosen pembimbing tugas akhir penulis yang telah memberikan bimbingan, arahan, dan masukan berharga selama proses pengerjaan tugas akhir.
6. Seluruh dosen dan staf di Program Studi Informatika, Universitas Islam Indonesia, atas ilmu, bantuan, dan dukungan yang telah diberikan selama masa studi.
7. Rekan-rekan seperjuangan : Bagas, Fachri, Fachrul, dan seluruh anggota Ronda Maszeh atas kebersamaan dan pertemanan yang memberikan warna selama masa perkuliahan.
8. Pradipa Sekar Maheswari yang selalu mendukung, menyemangati, dan mendampingi penulis sepenuh hati dengan menjadi motivasi di setiap langkah penulisan tugas akhir. *Thank you and I love you.*
9. Semua pihak yang turut membantu yang tidak bisa disebutkan satu-persatu.

Akhir kata, penulis menyadari bahwa laporan ini masih jauh dari sempurna. Oleh karena itu, penulis membuka diri terhadap kritik dan saran yang bersifat membangun. Semoga laporan ini dapat bermanfaat bagi pembaca dan menjadi awal dari kontribusi ilmiah yang lebih baik di masa depan.

Yogyakarta, 26 Juni 2025



(Muhammad Ali Zidan Laveda)

SARI

Revolusi Industri 4.0 telah membawa perubahan signifikan dalam pola belanja masyarakat, termasuk dalam sektor furnitur. Banyak Usaha Mikro, Kecil, dan Menengah (UMKM) belum sepenuhnya memanfaatkan teknologi digital untuk mendukung proses bisnis mereka. Penelitian ini bertujuan untuk mengembangkan sebuah *website e-commerce* berbasis Laravel yang dirancang khusus untuk mendukung UMKM furnitur "Rahdatu Wooden Furniture" dalam memasarkan produknya secara digital.

Metode pengembangan sistem yang digunakan adalah model *waterfall* yang terdiri dari tahapan analisis kebutuhan, desain, implementasi, dan pengujian. Data dikumpulkan melalui observasi dan wawancara langsung dengan pemilik UMKM. Sistem yang dibangun mencakup fitur-fitur utama seperti katalog produk, sistem keranjang belanja, manajemen stok, pemesanan online, verifikasi pesanan, serta dashboard admin.

Hasil pengujian menggunakan metode *Black Box Testing* dan *User Acceptance Test* menunjukkan bahwa sistem berfungsi sesuai dengan kebutuhan pengguna. *Website* ini diharapkan mampu meningkatkan efisiensi operasional UMKM serta memperluas jangkauan pasar melalui digitalisasi.

Kata kunci: *E-commerce*, Laravel, UMKM, Furnitur, *Website*, *Waterfall*, Digitalisasi.

GLOSARIUM

Black Box	Metode pengujian perangkat lunak yang menitikberatkan pada evaluasi fungsionalitas sistem berdasarkan input dan output, tanpa mempertimbangkan struktur internal atau kode program yang digunakan.
E-Commerce	Aktivitas jual beli produk atau layanan yang dilakukan secara online melalui platform berbasis web atau aplikasi.
Laravel	*Framework* PHP *open*-source berbasis arsitektur Model-View-Controller (MVC) yang dirancang untuk memudahkan pengembangan aplikasi web modern.
Marketplace	Platform digital yang mempertemukan penjual dan pembeli dalam satu ekosistem daring untuk melakukan transaksi.
Model-View-Controller	Arsitektur perangkat lunak yang memisahkan logika data, Antar muka pengguna, dan kontrol alur kerja.
User Acceptance Test	Pengujian akhir oleh pengguna untuk memastikan sistem sesuai dengan kebutuhan bisnis.
Waterfall	Metode pengembangan perangkat lunak berurutan dan sistematis dari awal hingga akhir.

DAFTAR ISI

HALAMAN JUDUL i
HALAMAN PENGESAHAN DOSEN PEMBIMBING ii
HALAMAN PENGESAHAN DOSEN PENGUJI iii
HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR iv
HALAMAN PERSEMBAHAN v
HALAMAN MOTO vi
KATA PENGANTAR vii
SARI ix
GLOSARIUM x
DAFTAR ISI xi
DAFTAR TABEL xiii
DAFTAR GAMBAR xiv
BAB I PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Batasan Masalah 3
1.4 Tujuan Penelitian 3
1.5 Manfaat Penelitian 3
1.6 Sistematika Penulisan 4
BAB II KAJIAN PUSTAKA 5
2.1 Landasan Teori 5
2.1.1 Website 5
2.1.2 Laravel 5
2.1.3 MySQL 5
2.1.4 *Use Case Diagram* 5
2.1.5 *Activity Diagram* 6
2.1.6 *User Acceptance Test* (UAT) 7
2.1.7 *Black Box Testing* 8
2.1.8 Metode Waterfall 8
2.2 Penelitian Terdahulu 9
2.2.1 Pengembangan Platform E-Commerce UMKM Berbasis Laravel dengan Metode Waterfall 9
2.2.2 Transformasi Digital Bisnis UMKM di Indonesia Setelah Masa Pandemi 10
2.2.3 Pengaruh Marketplace dan Digital Marketing terhadap Peningkatan Penjualan UMKM 11
BAB III METODOLOGI PENELITIAN 13
3.1 Analisis Kebutuhan 13
3.2 Desain 14
3.2.1 Use Case Diagram 15
3.2.2 *Activity Diagram* 16
3.2.3 Rancangan Basis Data 17
3.3 Implementasi 17
3.4 Pengujian Sistem/ Testing 17
3.5 Teknologi dan Perangkat Lunak 18
BAB IV HASIL DAN PEMBAHASAN 20
4.1 Analisis Kebutuhan 20

4.2 Desain....21
4.2.1 *Use Case Diagram*....21
4.2.2 *Activity Diagram*....24
4.2.3 Rancangan Basis Data....34
4.3 Implementasi....36
4.3.1 Login dan Register....36
4.3.2 Tampilan Halaman Katalog Produk....37
4.3.3 Tampilan Halaman Detail Produk....39
4.3.4 Tampilan Halaman Keranjang Belanja....39
4.3.5 Tampilan Halaman Riwayat Pesanan....40
4.3.6 Tampilan Halaman *Dashboard* Admin....41
4.3.7 Tampilan Halaman Daftar Produk Admin....41
4.3.8 Tampilan Halaman Kelola Kategori....42
4.3.9 Tampilan Halaman Tambah Produk Admin....43
4.3.10 Tampilan Halaman Kelola Pesanan Admin....43
4.4 Implementasi Kode Program....44
4.4.1 Fitur Login dan Registrasi....44
4.4.2 Fitur Manajemen Kategori Produk....48
4.4.3 Fitur Manajemen Produk....51
4.4.4 Fitur Manajemen Pesanan....57
4.4.5 Fitur Katalog Produk....61
4.4.6 Fitur Detail Produk....63
4.4.7 Fitur Keranjang Belanja....64
4.4.8 Fitur Checkout dan Membuat Pesanan....67
4.4.9 Fitur Melihat Status Pesanan....70
4.4.10 Struktur Routing Laravel....71
4.5 Pengujian Sistem/Testing....73
4.5.1 *Black Box Testing*....73
4.5.2 *User Acceptance Test* (UAT)....76
BAB V KESIMPULAN DAN SARAN....81
5.1 Kesimpulan....81
5.2 Saran....81
DAFTAR PUSTAKA....83

DAFTAR TABEL

Tabel 2.1 Komponen *Use Case Diagram* 6
Tabel 2.2 Komponen *Activity Diagram* 7
Tabel 4.1 Deskripsi *Use Case Diagram* 22
Tabel 4.2 Hasil Pengujian *Black Box* 74
Tabel 4.3 Hasil Pengujian UAT Terhadap Pemilik UMKM 77
Tabel 4.4 Hasil Pengujian UAT Terhadap Calon Pelanggan 78

DAFTAR GAMBAR

Gambar 3.1 Tahapan Metodologi Penelitian 13
Gambar 3.2 Contoh *Use Case Diagram* 15
Gambar 3.3 Contoh *Activity Diagram* 16
Gambar 4.1 *Use Case Diagram* 22
Gambar 4.2 *Activity Diagram* Login 24
Gambar 4.3 *Activity Diagram* Mengelola Produk 26
Gambar 4.4 *Activity Diagram* Mengelola Kategori 28
Gambar 4.5 *Activity Diagram* Mengelola Pesanan 30
Gambar 4.6 *Activity Diagram* Berbelanja 32
Gambar 4.7 *Activity Diagram* Mengecek Status Pesanan 34
Gambar 4.8 Rancangan Basis Data 35
Gambar 4.9 Tampilan Halaman Login 37
Gambar 4.10 Tampilan Halaman *Register* 37
Gambar 4.11 Halaman Katalog Produk 38
Gambar 4.12 Halaman Detail Produk 39
Gambar 4.13 Halaman Keranjang Belanja 40
Gambar 4.14 Halaman Riwayat Pesanan 41
Gambar 4.15 Halaman *Dasboard* Admin 41
Gambar 4.16 Halaman Daftar Produk 42
Gambar 4.17 Tampilan Halaman Kelola Kategori 43
Gambar 4.18 Halaman Tambah Produk 43
Gambar 4.19 Halaman Kelola Pesanan 44
Gambar 4.20 Implementasi Formulir Registerasi Pengguna 46
Gambar 4.21 Implementasi Penyimpanan Data Pengguna ke *Database* 46
Gambar 4.22 Fungsi *Redirect* Berdasarkan Peran 47
Gambar 4.23 *Route* Proses Autentikasi 47
Gambar 4.24 *Function* Fitur Menampilkan Daftar Kategori 48
Gambar 4.25 Tampilan Tabel Kategori 49
Gambar 4.26 Implementasi Form Tambah Kategori 49
Gambar 4.27 *Function* Menyimpan Data Kategori 49
Gambar 4.28 *Function* Fitur Hapus Kategori 50
Gambar 4.29 *Function* Edit Kategori 50

Gambar 4.30 Tampilan *Input* Edit Data Kategori Baru 50
Gambar 4.31 *Function* Menyimpan Data Perubahan Kategori 50
Gambar 4.32 Implementasi Fitur Tambah Produk 52
Gambar 4.33 *Function* Fitur Tambah Produk di *AdminController* 52
Gambar 4.34 Tombol Edit Produk 53
Gambar 4.35 *Function* Edit Produk 53
Gambar 4.36 Tombol Hapus Produk 54
Gambar 4.37 *Function* Hapus Produk 54
Gambar 4.38 Fitur Filter Produk 55
Gambar 4.39 Fitur *Sorting List* Produk 55
Gambar 4.40 *Function* view_product 57
Gambar 4.41 Implementasi Tampilan Tabel Pesanan 58
Gambar 4.42 *Function* Menampilkan Data Pada Tabel 59
Gambar 4.43 Implementasi Tombol Verifikasi dan Verifikasi Gagal 59
Gambar 4.44 *Function* Mengubah Status Pembayaran 60
Gambar 4.45 Implementasi Fitur Mengubah Status Pesanan 60
Gambar 4.46 *Function* Untuk Menyimpan dan Mengubah Status Pesanan 61
Gambar 4.47 Implementasi Fitur Katalog Produk 62
Gambar 4.48 *Function* Menampilkan Katalog Produk 62
Gambar 4.49 Implementasi Fitur Tampilan Detail Produk 63
Gambar 4.50 *Function* Menampilkan Halaman Detail Produk 64
Gambar 4.51 Implementasi Fitur Tombol Tambah Keranjang 65
Gambar 4.52 Implementasi Fitur Halaman Keranjang 66
Gambar 4.53 *Function* Halaman Keranjang 67
Gambar 4.54 Implementasi Fitur Checkout dan Membuat Pesanan 68
Gambar 4.55 *Function* Proses *Checkout* Pesanan 69
Gambar 4.56 Implementasi Fitur Melihat Status Pesanan 70
Gambar 4.57 *Function* Menampilkan Data Pesanan 71
Gambar 4.58 *Routes* Untuk Admin 72
Gambar 4.59 *Routes* Untuk Pelanggan 73

BAB I
PENDAHULUAN

1.1 Latar Belakang

Revolusi industri 4.0 telah membawa transformasi digital yang signifikan dalam berbagai sektor, termasuk industri furnitur. Kini, semakin banyak konsumen yang menggunakan platform online untuk berbelanja barang furnitur, baik untuk kebutuhan rumah tangga maupun bisnis mereka (Kementerian Komunikasi dan Informatika, 2022). Perubahan perilaku ini menuntut para pelaku usaha, terutama Usaha Mikro, Kecil, dan Menengah (UMKM) di bidang furnitur, untuk menyesuaikan diri dengan perkembangan pasar digital. Namun, menurut data dari Badan Pusat Statistik (BPS), hanya 17,5% UMKM di sektor industri pengolahan, termasuk furnitur, yang telah memanfaatkan teknologi digital secara penuh dalam proses produksi dan pemasaran (BPS, 2023). Hal ini menunjukkan adanya kesenjangan digital yang signifikan dan kebutuhan akan solusi berbasis teknologi yang tepat sasaran.

Dalam konteks digitalisasi UMKM, penggunaan *website e-commerce* menjadi salah satu solusi strategis. *Website* memberikan keleluasaan penuh kepada pemilik usaha untuk mengontrol fitur bisnis, seperti tampilan produk, harga, stok, serta sistem transaksi, tanpa bergantung pada aturan dan algoritma platform pihak ketiga seperti *marketplace* atau media sosial. Menurut Chaffey & Ellis-Chadwick (2019), kepemilikan *website* sendiri dapat meningkatkan kredibilitas bisnis dan membangun identitas merek secara lebih konsisten. Selain itu, *website* memungkinkan integrasi langsung dengan alat analitik digital dan sistem manajemen data pelanggan, yang mendukung pengambilan keputusan bisnis berbasis data. Dibandingkan dengan platform seperti Instagram atau Facebook yang hanya berfungsi sebagai etalase visual, *website* memberikan pengalaman transaksi yang lebih komprehensif dan profesional.

Dengan memanfaatkan transformasi digital, UMKM memiliki peluang besar untuk menjangkau pasar yang lebih luas melalui platform *e-commerce* dan meningkatkan daya saingnya di era ekonomi digital (Kementerian Koperasi dan UKM, 2021). Keunggulan memiliki *website e-commerce* antara lain adalah fleksibilitas dalam mempromosikan produk, kebebasan mengatur strategi pemasaran, serta peningkatan efisiensi operasional.

Mengembangkan website e-commerce berbasis Laravel merupakan salah satu solusi yang efektif untuk membantu UMKM furnitur mengatasi masalah tersebut. Laravel adalah *framework* PHP modern yang populer digunakan untuk pengembangan aplikasi web karena mendukung arsitektur Model-View-Controller, memudahkan pengelolaan basis data, dan

mendukung pembuatan aplikasi yang aman dan dapat dikembangkan secara skalabel. (Stauffer, 2019). Platform ini dapat membantu UMKM dalam menampilkan produk secara professional, mengelola stok secara lebih terstruktur, dan memberikan kemudahan bagi pelanggan dalam melakukan transaksi. Platform ini tidak hanya membantu memasarkan produk secara luas, tapi juga meningkatkan penjualan UMKM di pasar digital. Platform ini dapat membantu UMKM furnitur dalam memamerkan produk mereka kepada audiens yang lebih luas, meningkatkan visibilitas dan daya saing di pasar digital.

Keunggulan lain dari pengembangan *website e-commerce* adalah fleksibilitas dalam memasarkan produk. UMKM dapat dengan mudah mengatur dan mengoptimalkan penggunaan media sosial untuk mengarahkan pelanggan mereka ke *website* mereka. Dengan adopsi e-commerce, UMKM terbukti mampu meningkatkan penjualan dan memperluas jangkauan bisnis mereka secara signifikan tanpa batasan geografis (Purwanegara *et al*., 2021). Selain itu, *website e-commerce* juga memungkinkan UMKM dalam menganalisis tren pasar dan personalisasi layanan dengan mengumpulkan data pelanggan.

Dari permasalahan tersebut, penulis mendapatkan solusi untuk mengembangkan *website e-commerce* khusus untuk salah satu pelaku UMKM bernama Rahdatu Wooden Furniture dengan tujuan membantu memasarkan produk secara efektif melalui saluran digital dan menjangkau pasar yang lebih luas dan membantu konsumen untuk lebih mudah mencari barang furnitur. UMKM ini telah berdiri sejak tahun 2022 dan dikenal menyediakan furnitur berkualitas tinggi menggunakan kayu Jepara dengan harga yang ekonomis, serta melayani pesanan khusus sesuai keinginan pelanggan di seluruh Indonesia.

Dengan adanya *website e-commerce* ini, diharapkan dapat meningkatkan daya saing UMKM di pasar digital, mempercepat proses transaksi, serta memperkuat identitas *brand* di industri furnitur nasional. Pengembangan website ini diharapkan dapat menjadi teladan bagi UMKM lain dalam mengadopsi teknologi digital demi mendukung pertumbuhan bisnis mereka di era Revolusi Industri 4.0.

1.2 Rumusan Masalah

Berdasarkan latar belakang diperoleh rumusan masalah dalama penelitian ini yakni bagaimana perancangan dan pengembangan *website marketplace* berbasis Laravel dapat dilakukan untuk mendukung pemasaran digital pada UMKM Rahdatu Wooden Furniture melalui implementasi fitur-fitur penting seperti katalog produk, manajemen stok, sistem transaksi, dan personalisasi layanan.

1.3 Batasan Masalah

Penelitian ini akan dibatasi pada beberapa aspek spesifik untuk mencapai tujuan yang telah ditetapkan serta dalam upaya untuk menghasilkan penelitian yang mendalam dan terkelola. Batasan masalah antara lain adalah sebagai berikut

1. Penelitian hanya berfokus pada pengembangan website marketplace untuk satu UMKM, yaitu Rahdatu Wooden Furniture.
2. Fitur yang dikembangkan meliputi: katalog produk, manajemen stok, sistem keranjang belanja, sistem transaksi sederhana (simulasi pembayaran manual), manajemen pesanan, dan halaman profil bisnis.
3. Penelitian tidak mencakup pengembangan aplikasi mobile, sistem pembayaran otomatis (payment gateway), ataupun integrasi ke marketplace pihak ketiga.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini antara lain adalah sebagai berikut:

1. Merancang dan mengembangkan sebuah *website marketplace* berbasis Laravel yang dapat membantu UMKM Rahdatu Wooden Furniture dalam memasarkan produknya secara digital.
2. Membangun fitur-fitur penting seperti katalog produk, manajemen stok, dan sistem transaksi sederhana.
3. Meningkatkan efisiensi operasional, memperluas jangkauan pasar, serta memperkuat identitas merek Rahdatu Wooden Furniture di era ekonomi digital.

1.5 Manfaat Penelitian

Penelitian ini diharapkan dapat memberi manfaat sebagai berikut:

1. Memberikan solusi nyata bagi Rahdatu Wooden Furniture dalam meningkatkan pemasaran dan operasional bisnis melalui pemanfaatan website e-commerce.
2. Menjadi model penerapan transformasi digital yang dapat dijadikan contoh bagi UMKM lain dalam meningkatkan daya saing di era Revolusi Industri 4.0.
3. Memberikan dasar bagi pengembangan lebih lanjut, seperti integrasi sistem pembayaran otomatis, penggunaan kecerdasan buatan dalam pemasaran, atau pengembangan aplikasi mobile di masa depan.

1.6 Sistematika Penulisan

Sistematika penulisan pada tugas akhir ini bertujuan untuk memudahkan dalam memahami isi dari laporan ini. Pada penulisan laporan ini terdapat beberapa pokok pembahasan, yaitu :

a. **BAB I PENDAHULUAN**

   Bab ini membahas latar belakang dari Rahdatu Wooden Furniture, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metode penelitian, dan sistematika penulisan.

b. **BAB II KAJIAN PUSTAKA**

   Bab ini menjelaskan landasan teori yang akan menjadi dasar pengetahuan dan tinjauan terhadap penelitian terdahulu yang digunakan dalam menyusun laporan untuk mengembangkan *website* Rahadatu Wooden Furniture.

c. **BAB III METODOLGI PENELITIAN**

   Bab ini menjelaskan proses teknis dalam pengembangan sistem, dimulai dari analisis kebutuhan hingga tahap pengujian akhir.

d. **BAB IV HASIL DAN PEMBAHASAN**

   Bab ini memuat pembahasan mengenai hasil dari pengembangan sistem yang telah dilakukan dan hasil pengujian terhadap sistem yang telah dikembangkan.

e. **BAB V KESIMPULAN DAN SARAN**

   Bab ini memuat kesimpulan terkait sistem yang telah dikembangkan dan memberikan saran-saran yang bertujuan untuk pengembangan website Rahdatu Wooden Furniture agar semakin baik.

BAB II
KAJIAN PUSTAKA

2.1 Landasan Teori

2.1.1 Website

Website merupakan sekumpulan halaman web yang saling terhubung dan dapat diakses melalui internet dengan menggunakan domain tertentu. Menurut Morville & Rosenfeld (2015), desain dan fungsionalitas website memainkan peran penting dalam menciptakan pengalaman pengguna yang baik. Dalam konteks e-commerce, website harus responsif, cepat, dan aman untuk mendukung aktivitas transaksi secara efektif.

2.1.2 Laravel

Laravel adalah *framework* PHP open-source berbasis arsitektur Model-View-Controller (MVC) yang dirancang untuk memudahkan pengembangan aplikasi web modern. Laravel menawarkan fitur unggulan seperti sistem *routing* yang fleksibel, manajemen autentikasi, ORM Eloquent untuk *database*, serta proteksi keamanan standar industri. Dengan struktur kode yang terorganisir dan dokumentasi resmi yang lengkap, Laravel memungkinkan pengembang membangun aplikasi web skala kecil hingga besar secara efisien. *Framework* ini sangat cocok digunakan untuk membangun website marketplace UMKM karena mendukung fleksibilitas, skalabilitas, dan integrasi layanan pihak ketiga (Laravel Team, 2025).

2.1.3 MySQL

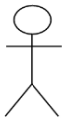
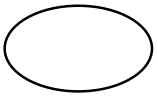
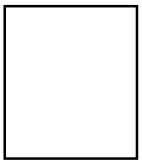
MySQL adalah sistem manajemen basis data relasional (RDBMS) yang berbasis SQL (Structured Query Language). Menurut Paul DuBois (2013), MySQL banyak digunakan dalam pengembangan aplikasi web karena kinerjanya yang cepat, dukungan skalabilitas, dan kompatibilitas yang luas. Dalam *e-commerce*, MySQL digunakan untuk menyimpan data pengguna, produk, transaksi, dan informasi lainnya dengan aman dan terstruktur.

2.1.4 *Use Case Diagram*

Use Case Diagram adalah diagram yang menggambarkan interaksi antara aktor (pengguna) dan sistem. Menurut Booch, Rumbaugh, dan Jacobson (2005), *Use Case Diagram* digunakan untuk mendeskripsikan perilaku sistem dari perspektif pengguna. Diagram ini membantu tim pengembang memahami kebutuhan fungsional sistem secara visual, seperti

proses registrasi, pencarian produk, hingga transaksi, sehingga perencanaan fitur menjadi lebih terstruktur dan sistematis. Komponen pada *Use Case Diagram* dapat dilihat pada Tabel 2.1.

Tabel 2.1 Komponen *Use Case Diagram*

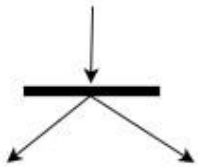
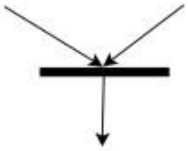
Simbol	Nama Simbol	Keterangan
	Aktor	Representasi pengguna atau sistem luar yang berinteraksi dengan sistem.
	*Use Case*	Fungsi atau layanan yang disediakan oleh sistem untuk aktor.
	Sistem	Batasan sistem yang menggambarkan ruang lingkup dari interaksi dengan aktor.
	Relasi (*Association*)	Menunjukkan hubungan antara aktor dan *use case*, menggambarkan interaksi mereka.
< ------ <<include>> ------ >	Relasi (*Include*)	Menunjukkan bahwa *use case* tertentu selalu memerlukan *use case* lainnya.
< ------ <<extend>> ------ >	Relasi (*Extend*)	Menunjukkan bahwa *use case* tertentu adalah ekstensi dari *use case* lainnya.

2.1.5 *Activity Diagram*

Activity Diagram adalah representasi grafis dari alur kerja atau proses dalam sebuah sistem. Menurut Fowler (2003), *Activity Diagram* digunakan untuk menunjukkan aliran aktivitas dalam sistem, baik yang berjalan secara berurutan maupun paralel. Diagram ini sangat bermanfaat dalam menggambarkan alur proses seperti checkout, mulai dari pemilihan produk

hingga konfirmasi pembayaran, sehingga memudahkan pemahaman dan dokumentasi proses bisnis. Komponen pada *Activity Diagram* dapat dilihat pada Tabel 2.2.

Tabel 2.2 Komponen *Activity Diagram*

Simbol	Nama Simbol	Keterangan
	Awal (*Initial State*)	Titik awal dari alur proses yang menggambarkan dimulainya aktivitas.
	Akhir (*Final State*)	Titik akhir dari alur proses yang menggambarkan penyelesaian aktivitas.
	Alur (*Flow*)	Menunjukkan arah alur aktivitas yang menunjukkan urutan langkah dalam proses.
	Aktivitas	Menunjukkan suatu aksi atau kegiatan yang dilakukan dalam alur proses.
	Keputusan	Titik percabangan dalam alur yang menghasilkan dua atau lebih pilihan.
	Fork	Menunjukkan pembagian alur menjadi beberapa jalur aktivitas paralel.
	Join	Menunjukkan penggabungan alur yang sebelumnya terpisah menjadi satu alur lagi.

2.1.6 *User Acceptance Test* (UAT)

User Acceptance Testing (UAT) adalah tahapan pengujian perangkat lunak yang dilakukan oleh pengguna akhir untuk memastikan bahwa sistem telah sesuai dengan kebutuhan mereka secara fungsional dan operasional. UAT biasanya dilakukan pada fase akhir sebelum sistem

benar-benar diimplementasikan, dan berfokus pada apakah fitur-fitur dapat digunakan dengan baik dalam konteks bisnis nyata.

Menurut Sommerville (2011), UAT merupakan bentuk validasi dari sudut pandang pengguna, berbeda dengan pengujian teknis seperti unit testing. Tujuan UAT adalah untuk menjamin bahwa sistem tidak hanya berjalan secara teknis, tetapi juga efektif digunakan oleh pengguna sasaran. UAT sangat penting khususnya untuk proyek-proyek seperti pengembangan sistem e-commerce UMKM, di mana keberhasilan sistem ditentukan oleh kenyamanan dan kemudahan pengguna.

Langkah-langkah UAT secara umum mencakup:

1. Menentukan fitur yang diuji berdasarkan kebutuhan pengguna.
2. Menyusun skenario uji dari aktivitas pengguna sehari-hari.
3. Melibatkan pengguna untuk menguji fitur secara langsung.
4. Mengumpulkan umpan balik dan mengevaluasi sistem berdasarkan pengalaman mereka.

Melalui proses ini, pengembang dapat memperoleh wawasan yang berguna untuk melakukan penyempurnaan sistem sebelum peluncuran penuh.

2.1.7 *Black Box Testing*

Black Box Testing adalah metode pengujian perangkat lunak yang menitikberatkan pada evaluasi fungsionalitas sistem berdasarkan input dan output, tanpa mempertimbangkan struktur internal atau kode program yang digunakan. Pengujian ini bertujuan untuk memastikan bahwa perangkat lunak memberikan respons yang sesuai dengan kebutuhan pengguna sebagaimana tercantum dalam spesifikasi. Teknik ini berguna untuk mengidentifikasi kesalahan atau cacat fungsional dalam sistem melalui pendekatan eksternal, sehingga sangat ideal digunakan dalam tahap akhir pengembangan sebelum produk dirilis (Patel & Amin, 2020). Beberapa teknik yang sering digunakan dalam Black Box Testing antara lain equivalence partitioning, boundary value analysis, decision table testing, dan state transition testing (Alam & Ahmad, 2021).

2.1.8 Metode Waterfall

Metode Waterfall merupakan salah satu pendekatan pengembangan perangkat lunak tradisional yang bersifat sistematis dan berurutan. Model ini pertama kali diperkenalkan oleh Winston W. Royce pada tahun 1970. Menurut Royce (1970), setiap tahapan dalam model ini harus diselesaikan terlebih dahulu sebelum melanjutkan ke tahap berikutnya, sehingga alurnya

menyerupai aliran air terjun (*waterfall*). Waterfall terdiri dari beberapa tahap, yaitu analisis kebutuhan, desain, implementasi, dan testing. Metode ini cocok untuk proyek yang memiliki kebutuhan yang jelas sejak awal dan memerlukan dokumentasi yang lengkap di setiap tahap.

a) **Analisis Kebutuhan**

   Pada tahap ini, pengembang melakukan proses pengumpulan informasi dari pengguna atau klien untuk mengidentifikasi kebutuhan sistem. Informasi ini didokumentasikan dalam bentuk *Software Requirements Specification* (SRS) yang akan menjadi dasar pengembangan selanjutnya (Pressman, 2010).

b) **Desain**

   Tahap ini bertujuan untuk mengubah kebutuhan yang telah dianalisis menjadi bentuk desain sistem yang rinci. Perancangan ini meliputi arsitektur sistem, struktur data, Antar muka pengguna, serta desain basis data (Sommerville, 2011).

c) **Implementasi**

   Pada tahap ini, desain sistem yang telah disusun diubah menjadi kode program dengan menggunakan bahasa pemrograman tertentu. Setiap modul yang telah dirancang dikembangkan secara mandiri hingga membentuk sistem yang utuh (Pressman, 2010).

d) **Pengujian/Testing**

   Setelah implementasi selesai, tahap selanjutnya adalah pengujian untuk memastikan bahwa perangkat lunak berfungsi sebagaimana mestinya. Pengujian dilakukan mulai dari level unit, integrasi antar modul, hingga sistem secara keseluruhan (Myers, Sandler, & Badgett, 2011).

2.2 Penelitian Terdahulu

2.2.1 Pengembangan Platform E-Commerce UMKM Berbasis Laravel dengan Metode Waterfall

Penelitian yang dilakukan oleh Rahayu dan Putra (2022) berjudul *"Pengembangan Platform E-Commerce UMKM Berbasis Laravel dengan Metode Waterfall"* mengkaji pengembangan aplikasi e-commerce berbasis Laravel khusus untuk mendukung UMKM lokal dalam memperluas jangkauan pemasaran produknya. Studi ini menggunakan model pengembangan Waterfall, yang mengandalkan tahapan linier mulai dari perencanaan kebutuhan, perancangan sistem, implementasi, hingga tahap pengujian akhir.

Laravel dipilih karena *framework* ini menawarkan struktur arsitektur Model-View-Controller (MVC) yang kuat, kemudahan dalam pengelolaan *database* menggunakan Eloquent

ORM, serta fitur keamanan bawaan seperti CSRF protection dan autentikasi. Peneliti juga menekankan kemudahan penggunaan fitur-fitur Laravel dalam mempercepat pengembangan modul-modul penting seperti katalog produk, manajemen transaksi, dan integrasi dengan media sosial.

Hasil implementasi menunjukkan bahwa marketplace berbasis Laravel yang dikembangkan berhasil meningkatkan akses pasar produk UMKM lokal hingga 40%, serta memberikan platform yang lebih independen dibandingkan bergantung sepenuhnya pada marketplace besar seperti Shopee atau Tokopedia. Marketplace ini juga dinilai lebih fleksibel dalam mengatur harga, promosi, dan branding.

Korelasi dengan penelitian ini sangat kuat karena proyek Rahdatu Wooden Furniture juga menggunakan Laravel dan menyasar kebutuhan UMKM, serta mengembangkan fitur inti seperti katalog produk dan sistem transaksi. Namun, perbedaan utama terletak pada fokus pendekatan sistem: penelitian Rahayu dan Putra menekankan pengembangan sistem umum untuk UMKM, sedangkan penelitian ini mengembangkan marketplace untuk satu UMKM saja, sehingga kebutuhan dan strategi pengembangannya lebih terarah dan spesifik.

Dari penelitian ini, penulis mengambil kontribusi penting dalam hal justifikasi pemilihan Laravel sebagai *framework* utama, serta validasi efektivitas pendekatan waterfall untuk pengembangan sistem skala kecil hingga menengah yang bersifat kustom dan terstruktur.

2.2.2 Transformasi Digital Bisnis UMKM di Indonesia Setelah Masa Pandemi

Penelitian oleh Wahyudi dan Mahendra (2023) dalam artikel *"Transformasi Digital Bisnis UMKM di Indonesia Setelah Masa Pandemi"* membahas fenomena percepatan adopsi teknologi digital di kalangan UMKM Indonesia sebagai respons terhadap dampak pandemi COVID-19. Studi ini menggunakan pendekatan kuantitatif melalui survei terhadap 150 UMKM dari berbagai sektor bisnis.

Hasil penelitian menunjukkan bahwa lebih dari 60% UMKM yang mengadopsi teknologi digital seperti website e-commerce, aplikasi marketplace, dan sistem pembayaran digital mengalami peningkatan pendapatan. Faktor-faktor utama yang mendorong keberhasilan transformasi digital UMKM mencakup peningkatan keterampilan digital pelaku usaha, dukungan pemerintah melalui program digitalisasi, serta perubahan pola konsumsi masyarakat ke arah transaksi daring.

Selain itu, penelitian ini menyoroti bahwa UMKM yang mengandalkan website e-commerce sendiri lebih mampu membangun hubungan yang kuat dengan pelanggan, meningkatkan loyalitas konsumen, serta mendapatkan data perilaku pelanggan untuk kebutuhan analitik bisnis. Transformasi digital terbukti tidak hanya menjadi opsi, melainkan kebutuhan strategis untuk kelangsungan bisnis jangka panjang.

Korelasi dengan penelitian ini tampak pada motivasi pengembangan website sebagai bentuk adaptasi terhadap perubahan perilaku konsumen dan kebutuhan digitalisasi pasca pandemi. Bedanya, studi Wahyudi dan Mahendra bersifat makro dan berbasis data kuantitatif agregat dari berbagai UMKM lintas sektor, sedangkan penelitian ini lebih bersifat mikro, fokus pada studi kasus spesifik terhadap satu UMKM dengan pendekatan pengembangan teknologi.

Penelitian ini memberi landasan kuat bahwa transformasi digital bukan hanya kebutuhan jangka pendek, tetapi strategi berkelanjutan. Oleh karena itu, pengembangan website Rahdatu menjadi langkah konkrit dalam menjawab kebutuhan jangka panjang bisnis furnitur lokal terhadap digitalisasi.

2.2.3 Pengaruh Marketplace dan Digital Marketing terhadap Peningkatan Penjualan UMKM

Penelitian yang dilakukan oleh Sari dan Hidayatullah (2021) dalam artikel *"Pengaruh Marketplace dan Digital Marketing terhadap Peningkatan Penjualan UMKM"* mengevaluasi seberapa besar peran penggunaan marketplace dan strategi pemasaran digital dalam mendukung pertumbuhan UMKM. Studi ini melibatkan 50 UMKM di Kabupaten Sumenep, Jawa Timur, menggunakan pendekatan kuantitatif dan analisis regresi sederhana.

Hasil penelitian menunjukkan bahwa kehadiran di platform marketplace secara langsung berdampak positif terhadap peningkatan volume penjualan UMKM. Selain itu, integrasi strategi pemasaran digital, seperti penggunaan media sosial, email marketing, dan iklan digital, memperkuat efektivitas marketplace dalam menarik konsumen baru.

Studi ini juga mengungkapkan bahwa UMKM yang mengembangkan platform sendiri, seperti website marketplace mandiri, memiliki fleksibilitas lebih dalam mengontrol brand positioning, menetapkan harga produk, serta menjaga loyalitas pelanggan melalui program-program eksklusif.

Dalam kaitannya dengan penelitian ini, hasil tersebut memperkuat argumentasi bahwa Rahdatu Wooden Furniture perlu memiliki website marketplace sendiri sebagai sarana untuk memperkuat identitas merek dan memperluas pasar nasional.

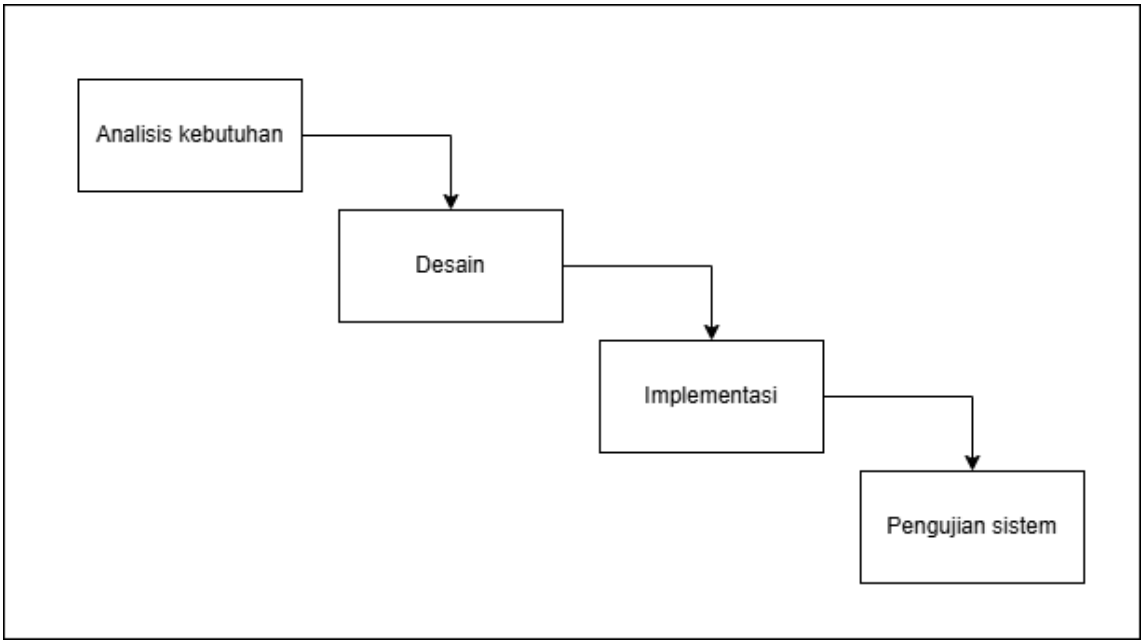
Penelitian ini menjadi salah satu dasar penting dalam menjelaskan kenapa marketplace digital yang dikembangkan secara mandiri lebih menguntungkan dibandingkan hanya bergantung pada marketplace besar.

Korelasi dengan penelitian ini terlihat dari penggunaan kanal digital sebagai pendorong pertumbuhan penjualan. Namun, penelitian ini masih terbatas pada analisis kehadiran di marketplace umum seperti Tokopedia atau Shopee, sementara penelitian yang dilakukan penulis terfokus pada pengembangan marketplace mandiri yang dapat mengatur sendiri strategi branding, promosi, dan interaksi pelanggan.

Kontribusi dari penelitian ini adalah pemahaman bahwa kehadiran digital memang krusial, namun untuk membangun brand identity yang kuat dan hubungan pelanggan jangka panjang, UMKM perlu mempertimbangkan membangun platform e-commerce sendiri. Hal ini mendukung keputusan pengembangan *website e-commerce* UMKM furniture yang lebih fleksibel dan berorientasi pada kontrol penuh atas brand dan data pelanggan.

BAB III
METODOLOGI PENELITIAN

Metodologi penelitian yang digunakan dalam pengembangan website ini menggunakan metode *waterfall.* Tahapan yang digunakan dalam pengembangan ini mencakup analisis kebutuhan, desain, implementasi, dan pengujian. Ilustrasi tahapan yang digunakan dalam metodologi penelitian ini dapat dilihat pada Gambar 3.1.



Gambar 3.1 Tahapan Metodologi Penelitian

3.1 Analisis Kebutuhan

Analisis kebutuhan merupakan tahap penting dalam pengembangan sistem untuk memahami apa yang dibutuhkan pengguna agar solusi yang dikembangkan tepat sasaran (Sugiyono, 2018). Tahap ini bertujuan untuk mengidentifikasi kebutuhan pengguna secara menyeluruh, baik dari aspek fungsional maupun non-fungsional, sehingga sistem yang dibangun benar-benar sesuai dengan permasalahan nyata yang dihadapi.

Untuk menggali informasi kebutuhan sistem, penulis melakukan pengumpulan data melalui metode observasi langsung dan wawancara. Observasi dilakukan untuk memahami alur kerja dan proses bisnis yang dijalankan oleh UMKM, sementara wawancara digunakan untuk menggali secara mendalam ekspektasi serta kebutuhan pengguna terhadap sistem yang akan dibangun. Metode wawancara dipilih karena memungkinkan penulis mendapatkan jawaban yang lebih terbuka dan kontekstual, serta mampu menangkap informasi yang tidak terlihat secara langsung dalam observasi. Sementara itu, observasi mendukung pemahaman penulis terhadap kondisi nyata di lapangan agar sistem yang dirancang benar-benar sesuai dengan

kebutuhan pengguna.

Wawancara dilakukan bersama pemilik UMKM Rahdatu Wooden Furniture, di sebuah *coffeeshop* yang beralamat di Jalan Pandanaran – UII Jalan Kaliurang km 14, Sleman, D.I.Yogyakarta. Tujuan dari wawancara ini adalah untuk mengidentifikasi kebutuhan sistem yang sesuai dengan aktivitas bisnis yang dijalankan oleh UMKM tersebut.

Pertanyaan-pertanyaan yang diajukan dalam sesi wawancara mencakup aspek operasional hingga harapan terhadap sistem, di antaranya sebagai berikut:

1. Bagaimana proses penjualan dilakukan pada saat ini?
2. Apa saja kendala yang dihadapi dalam proses penjualan?
3. Apa harapan terhadap sistem yang akan dikembangkan?
4. Fitur-fitur apa saja yang perlu ada dalam sistem untuk membantu proses penjualan dan manajemen produk?
5. Bagaimana harapan Anda mengenai keamanan dan pengelolaan data sistem?
6. Apakah ada fitur tambahan yang Anda harapkan untuk menunjang proses bisnis?

Pertanyaan-pertanyaan tersebut dirancang untuk menggali kebutuhan sistem baik dari dua sisi utama, yaitu:

- Kebutuhan Fungsional, yaitu fitur-fitur utama yang harus tersedia pada sistem seperti pemesanan produk, pengelolaan data produk, dan proses verifikasi pesanan.
- Kebutuhan Non-Fungsional, yang mencakup aspek-aspek seperti kemudahan penggunaan (*user friendly*), keamanan akses data, dan performa sistem dalam menangani transaksi secara efisien.

Penjabaran hasil lengkap dari sesi wawancara dan observasi akan dipaparkan lebih rinci pada Bab 4: Hasil dan Pembahasan.

Dengan pendekatan analisis kebutuhan ini, diharapkan sistem yang dikembangkan mampu menjawab permasalahan bisnis yang ada secara tepat dan relevan, serta dapat meningkatkan efisiensi dan kualitas layanan dari UMKM yang bersangkutan.

3.2 Desain

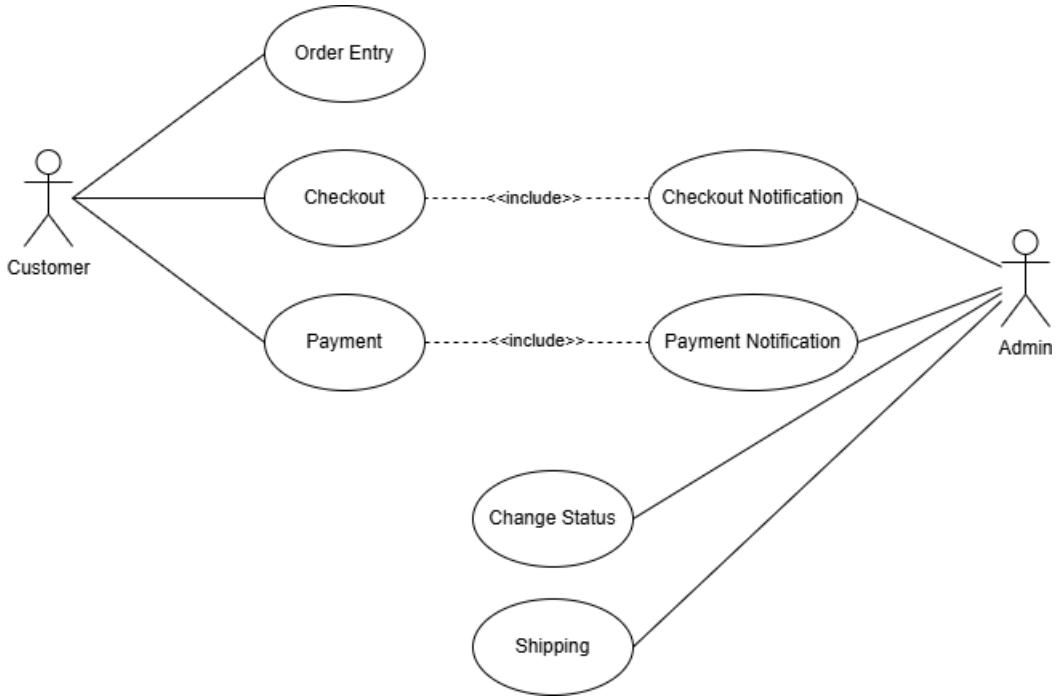
Desain sistem adalah proses perencanaan yang mendalam untuk menentukan struktur dan interaksi dari komponen sistem yang akan dibangun, dengan tujuan untuk memenuhi kebutuhan pengguna dan memastikan sistem dapat berfungsi secara efektif setelah implementasi (Milisavljevic-Syed *et al*., 2020). Tahap desain sistem merupakan kelanjutan dari hasil analisis kebutuhan yang telah dilakukan sebelumnya. Desain sistem bertujuan untuk

menggambarkan bagaimana sistem akan dibangun berdasarkan kebutuhan pengguna, baik dari sisi fungsionalitas maupun struktur data. Pada tahap ini, dilakukan pemodelan sistem untuk menggambarkan alur proses dan hubungan antar komponen yang akan dikembangkan.

Desain sistem ini disusun untuk menjawab permasalahan yang telah diidentifikasi pada tahap analisis kebutuhan. Oleh karena itu, pada tahap ini digunakan beberapa alat bantu pemodelan, yaitu:

3.2.1 Use Case Diagram

Use case diagram digunakan untuk menggambarkan interaksi antara aktor (pengguna) dengan sistem yang akan dibangun. Diagram ini memberikan gambaran menyeluruh terhadap fungsi-fungsi utama sistem dari perspektif pengguna. Penggunaan use case diagram bertujuan untuk memetakan kebutuhan fungsional yang harus disediakan sistem, seperti melakukan pemesanan, mengelola produk, mengelola pesanan, dan mengakses riwayat transaksi. Contoh *use case diagram* dapat dilihat pada Gambar 3.2.



Gambar 3.2 Contoh *Use Case Diagram*

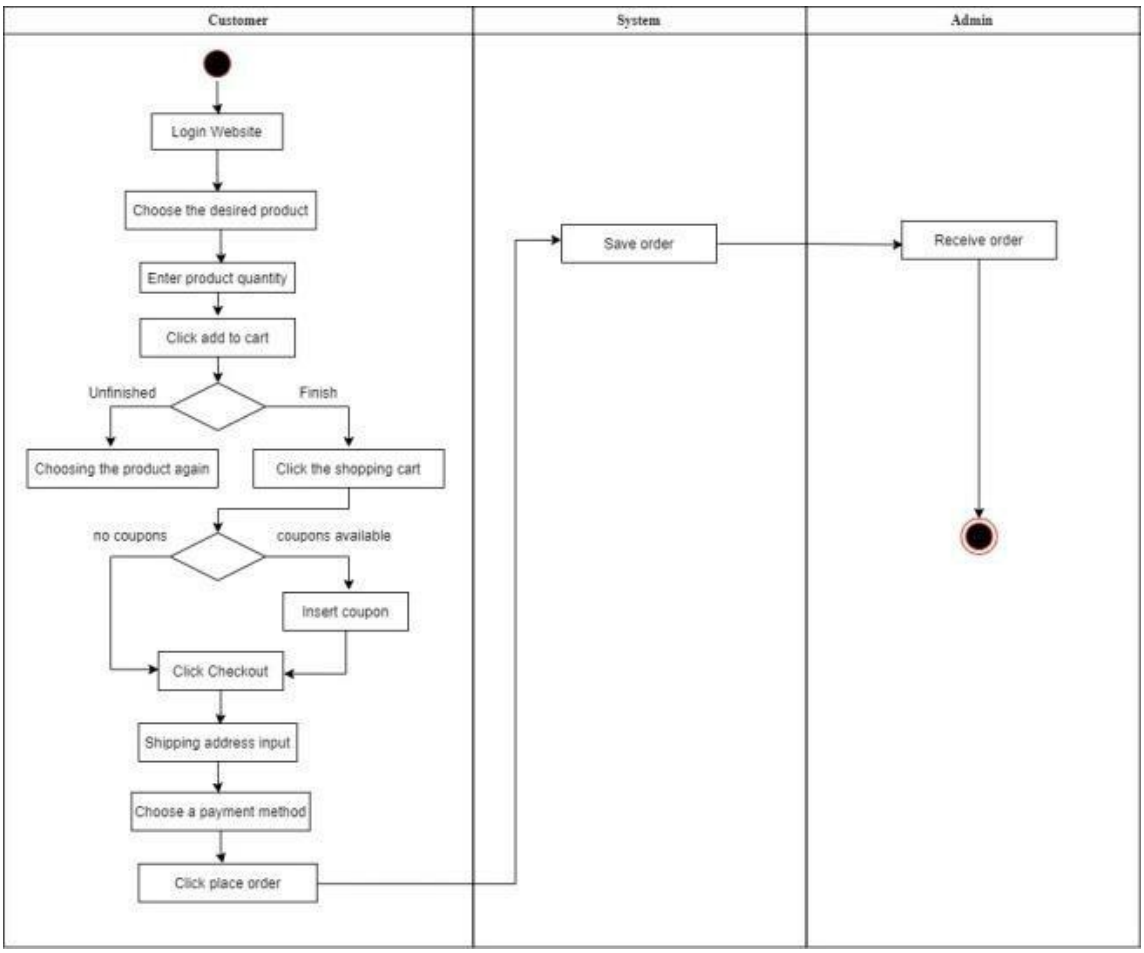
Sumber : Jabar, J. A., & Anubhakti, D. (2021)

Diagram ini menggambarkan intraksi antar aktor (pengguna dan admin) dalam sistem e-commerce. Customer dapat melakukan *Order Entry, Checkout,* dan *Payment*, dimana proses *Checkout* dan pembayaran akan memicu notifikasi otomatis kepada admin. Admin kemudian

akan menerima notifikasi tersebut dan bertanggung jawab untuk mengubah status pesanan serta melanjutkan ke proses pengiriman.

3.2.2 *Activity Diagram*

Activity diagram menggambarkan alur aktivitas atau proses dalam sistem secara rinci. Diagram ini digunakan untuk memvisualisasikan langkah-langkah dalam sebuah proses bisnis, seperti proses pemesanan produk atau manajemen pesanan oleh admin. Penggunaan *activity diagram* membantu dalam memahami urutan logis setiap aktivitas serta kemungkinan percabangan proses, sehingga memudahkan pengembang dalam merancang alur sistem yang efektif dan efisien. Contoh *activity diagram* dapat dilihat pada Gambar 3.3.



Gambar 3.3 Contoh *Activity Diagram*

Sumber : Jabar, J. A., & Anubhakti, D. (2021)

Diagram ini menggambarkan alur proses pemesanan produk oleh customer. Dimulai dari login ke website, memilih produk, memasukkan jumlah, dan menambahkannya ke keranjang.

Setelah selesai berbelanja, customer dapat menggunakan kupon (jika tersedia), lalu melanjutkan ke proses checkout, memilih alamat pengirimandan metode pembayaran, lalu menekan tombol “place order”. Sistem kemudian menyimpan pesanan dan mengirimkannya ke admin untuk diproses lebih lanjut.

3.2.3 Rancangan Basis Data

Perancangan *database* dilakukan untuk mendukung penyimpanan data yang sistematis dan terstruktur. *Database* menjadi komponen inti dalam sistem e-commerce ini karena mencatat data penting seperti produk, pelanggan, pesanan, dan riwayat transaksi. Desain *database* dilakukan dengan merancang *entity relationship diagram (ERD)* dan menentukan struktur tabel yang akan digunakan. Tujuan dari perancangan ini adalah memastikan integritas data, kemudahan akses, dan efisiensi dalam pengolahan informasi.

3.3 Implementasi

Implementasi sistem informasi merupakan tahap penting dalam menghubungkan solusi teknologi dengan kebutuhan bisnis secara nyata. Proses ini melibatkan penerapan rancangan sistem, integrasi teknologi ke dalam alur kerja organisasi, dan validasi kinerja sistem agar sesuai dengan tujuan strategis yang telah ditetapkan (Aljawarneh & Al-Omari, 2018). Tahap implementasi dilakukan dengan mengembangkan sistem berbasis web menggunakan bahasa pemrograman PHP dan *framework* Laravel. Laravel dipilih karena mendukung arsitektur Model-View-Controller (MVC) yang memudahkan pengelolaan struktur kode dan pengembangan fitur-fitur sistem. Selanjutnya, proses backend dikembangkan menggunakan Laravel untuk mendukung fungsi-fungsi seperti pendaftaran pengguna, manajemen produk, dan transaksi pemesanan. Sisi *front end* dikembangkan menggunakan Blade Template Laravel untuk membangun tampilan *website* yang responsif dan *user-friendly*.

3.4 Pengujian Sistem/ Testing

Setelah sistem berhasil diimplementasikan, tahap selanjutnya adalah melakukan pengujian untuk memastikan bahwa seluruh fungsionalitas berjalan sesuai dengan kebutuhan yang telah ditentukan. Pengujian ini penting untuk memvalidasi bahwa sistem tidak hanya berfungsi secara teknis, tetapi juga dapat digunakan dengan baik oleh pengguna sasaran, dalam hal ini pemilik UMKM dan pelanggan.

Dalam penelitian ini, digunakan dua metode pengujian, yaitu *Black Box Testing* dan pengujian berbasis umpan balik pengguna (user acceptance test) terhadap pemilik UMKM Rahdatu Wooden Furniture sebagai pengguna utama sistem.

Pada tahap *Black Box Testing*, penulis melakukan pengujian terhadap sistem menggunakan metode *Black Box Testing*. Pengujian ini difokuskan pada pemeriksaan fungsi-fungsi utama dalam sistem tanpa melihat kode sumber secara langsung. Penulis menguji setiap fitur berdasarkan masukan (input) yang diberikan dan keluaran (output) yang dihasilkan, untuk memastikan bahwa sistem telah berjalan sesuai dengan yang diharapkan oleh pengguna.

Beberapa skenario pengujian yang dilakukan mencakup validasi input pada form, proses login dan pendaftaran, penambahan produk, pemrosesan pesanan, serta fitur pencarian dan kategori produk. Hasil dari setiap pengujian dicatat untuk mengevaluasi apakah fungsi tersebut berjalan dengan benar atau masih ditemukan kesalahan (error). Dengan metode ini, penulis dapat memastikan bahwa sistem telah memenuhi spesifikasi fungsional dan dapat digunakan sesuai kebutuhan pengguna.

Selain pengujian teknis, dilakukan pula pengujian langsung terhadap pemilik UMKM sebagai bentuk pengujian pengguna akhir (user acceptance test). Tujuan dari pengujian ini adalah untuk mengetahui apakah sistem yang dikembangkan telah sesuai dengan kebutuhan dan ekspektasi pengguna. Pengujian dilakukan dengan menyusun perangkat pengujian dan scenario pengujian atau *test scenario.* Skenario pengujian disusun untuk memverifikasi fitur-fitur yang dikembangkan dapat berjalan dengan kebutuhan pengguna. Skenario tersebut berfungsi sebagai panduan bagi pemilik UMKM dalam megoperasikan sistem.

Dari hasil pengujian ini, diperoleh gambaran menyeluruh mengenai seberapa baik sistem dapat dioperasikan oleh pemilik UMKM secara langsung. Apabila terdapat bagian yang membingungkan atau tidak sesuai, maka akan dilakukan evaluasi dan penyempurnaan.

Dengan kombinasi pengujian teknis melalui Black Box Testing dan pengujian langsung oleh pengguna utama, sistem yang dikembangkan diharapkan tidak hanya berjalan dengan baik secara teknis, tetapi juga benar-benar bermanfaat dan mudah digunakan oleh pelaku UMKM dalam operasional sehari-hari.

3.5 Teknologi dan Perangkat Lunak

Dalam pengembangan sistem ini, digunakan berbagai perangkat lunak dan teknologi pendukung untuk menunjang proses implementasi secara efisien dan terstruktur. Adapun teknologi yang digunakan adalah sebagai berikut :

- Laravel v11 : *Framework* PHP untuk pengembangan *backend* dengan struktur MVC.
- MySQL : Sistem manajemen basis data relasional untuk menyimpan data pengguna, produk, dan transaksi.
- Blade Template : *Engine* tampilan (*view*) pada Laravel untuk membangun Antar muka pengguna secara dinamis.
- Visual Studio Code : Editor kode utama yang digunakan selama proses pengembangan.
- GitHub : *Platform version control* untuk mengelola versi dan *backup* kode program.
- XAMPP : Server lokal untuk menjalankan dan menguji aplikasi secara offline.
- Google Chrome : Digunakan untuk menguji kompatibilitas dan tampilan Antar muka pada berbagai browser.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Analisis Kebutuhan

Penulis melakukan observasi dan wawancara dengan pemilik UMKM untuk mengidentifikasi kebutuhan sistem yang akan dikembangkan. Pada hasil wawancara tersebut didapatkan jawaban dari pertanyaan-pertanyaan yang sudah disebutkan pada BAB 3 sebagai berikut:

1. Bagaimana proses penjualan dilakukan pada saat ini?
    - Diketahui Proses penjualan masih dilakukan secara manual. Komunikasi antara pembeli dan penjual dilakukan melalui WhatsApp dan Instagram. Pembeli harus menghubungi admin satu per satu untuk menanyakan ketersediaan produk.
2. Apa saja kendala yang dihadapi dalam proses penjualan?
    - Salah satu kendala utama adalah pencatatan pesanan dan pengelolaan stok produk yang cukup merepotkan, terutama saat jumlah pesanan meningkat.
3. Apa harapan terhadap sistem yang akan dikembangkan?
    - Saya menginginkan website yang bisa menampilkan katalog produk lengkap, termasuk gambar, deskripsi, dan harga. Selain itu, pelanggan harus bisa memesan langsung melalui website, tanpa harus menghubungi admin. Saya juga ingin ada fitur riwayat pemesanan pelanggan.
4. Bagaimana Anda ingin mengelola produk di website tersebut nantinya?
    - Saya berharap bisa menambahkan, mengedit, dan menghapus produk secara mandiri tanpa harus memahami bahasa pemrograman.
5. Bagaimana harapan Anda mengenai keamanan dan pengelolaan data sistem?
    - Saya ingin ada sistem login khusus untuk admin agar data tetap aman dan hanya bisa diakses oleh pihak internal.
6. Apakah ada fitur tambahan yang Anda harapkan untuk menunjang proses bisnis?
    - Saya berharap ada fitur pengelolaan pesanan, termasuk verifikasi pembayaran sebelum proses pengiriman dilakukan.

Berdasarkan hasil wawancara dapat disimpulkan oleh penulis bahwa pengembangan *website* ini memiliki dua kebutuhan sistem, yaitu siste fungsional dan non-fungsional. Dari sisi kebutuhan sistem fungsional,sistem diharapkan mencakup :

1. Katalog Produk : Menampilkan produk secara lengkap dengan gambar, deskripsi, dan harga, sehingga pelanggan dapat melihat semua pilihan tanpa harus menghubungi admin.
2. Fitur Pemesanan Online : Pelanggan dapat melakukan pemesanan langsung melalui website, yang mencakup formulir pemesanan, pemilihan produk, dan metode pembayaran.
3. Manajemen Produk oleh Admin : Admin dapat menambahkan, mengedit, atau menghapus produk secara mandiri melalui Antar muka backend yang disediakan.
4. Manajemen Pesanan : Admin dapat melihat daftar pesanan yang masuk, memverifikasi pembayaran, dan memperbarui status pengiriman.
5. Sistem Login Admin : Untuk menjaga keamanan dan membatasi akses ke fitur manajemen sistem.

Sedangkan kebutuhan non-fungsional, sistem diharapkan memiliki :

1. Kemudahan Pengguna : Antar muka pengguna yang intuitif untuk memudahkan pelanggan dan admin dalam menggunakan sistem tanpa pelatihan teknis yang kompleks.
2. Keamanan Data : Pembatasan akses melalui login admin serta validasi transaksi untuk mencegah kesalahan input data.

Keseluruhan kebutuhan ini menjadi acuan utama dalam proses perancangan sistem yang akan dijelaskan pada tahap selanjutnya. Sistem akan dirancang dengan mempertimbangkan kebutuhan pengguna sebagai aktor utama, baik dari sisi pelanggan sebagai pengguna akhir maupun admin sebagai pengelola sistem.

4.2 Desain

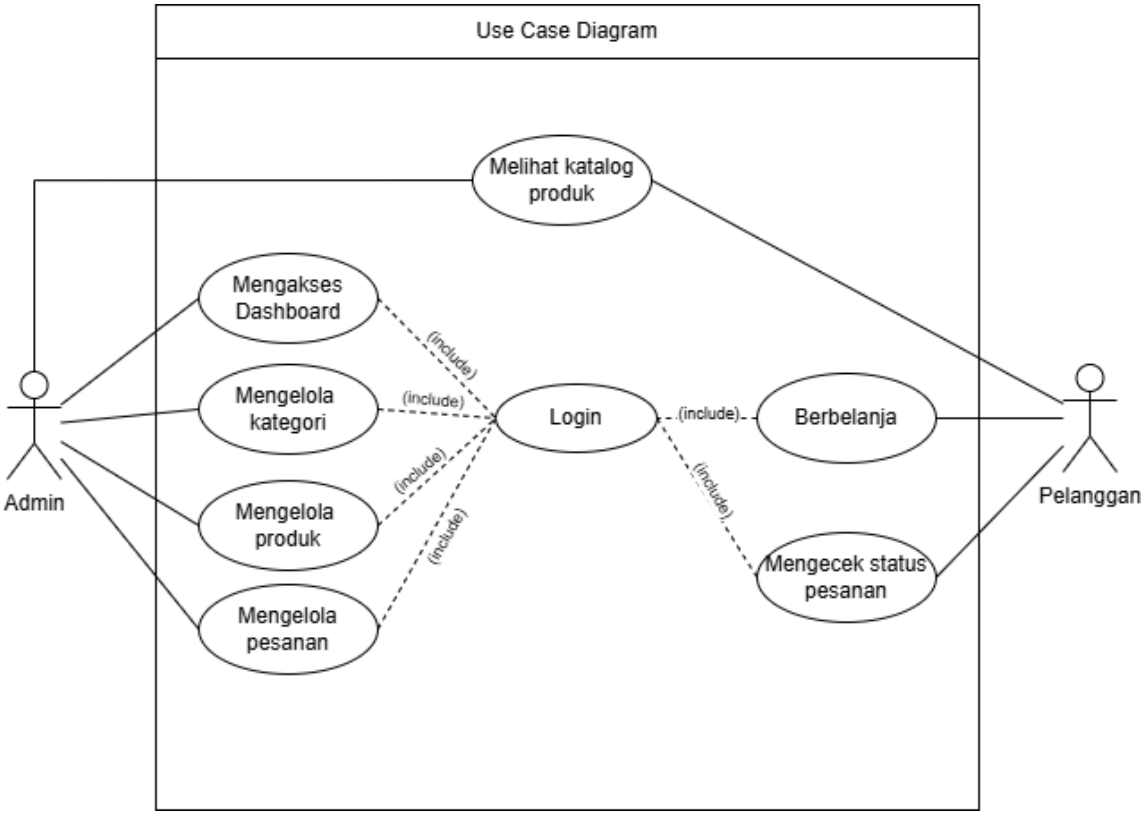
4.2.1 *Use Case Diagram*

Setelah dilakukan observasi dan wawancara dengan pemilik UMKM diperoleh kesimpulan bahwa akan ada 2 aktor pengguna, yaitu *Admin* sebagai pengelola dan *Pelanggan* sebagai pengguna layanan. Masing-masing aktor memiliki peran dan kebutuhan yang berbeda terhadap sistem, yang kemudian dituangkan dalam sebuah *use case* diagram untuk memetakan interaksi mereka terhadap fitur-fitur sistem.

Aktor Admin memiliki peran sebagai pengelola sistem. Setelah berhasil melakukan login, admin diarahkan ke halaman dashboard yang menampilkan informasi penting seperti jumlah pelanggan yang terdaftar, jumlah produk di katalog, pesanan yang masuk, dan pesanan

yang telah dikirim. Selain itu, admin memiliki akses penuh untuk mengelola data produk dan pesanan pelanggan. Hal ini mencakup proses penambahan, pengeditan, penghapusan produk, verifikasi pembayaran, hingga pembaruan status pengiriman.

Sementara itu, Pelanggan menggunakan sistem untuk melihat produk, melakukan pemesanan, serta mengecek status pesanan mereka. Pelanggan tidak diharuskan login untuk melihat katalog produk, namun login menjadi syarat wajib untuk melakukan pembelian dan memantau status pemesanan. Ilustrasi interaksi yang akan dilakukan dapat dilihat pada *use case diagram* pada Gambar 4.1.



Gambar 4.1 *Use Case Diagram*

Penjelasan lebih rinci terkait masing-masing *use case* dapat dilihat pada Tabel 4.1 berikut ini, yang menyajikan aktor, nama *use case*, dan deskripsi singkat fungsionalitasnya.

Tabel 4.1 Deskripsi *Use Case Diagram*

Kode	Aktor	*Use Case*	Deskripsi
UC1	Admin	Login	Admin melakukan proses autentikasi untuk dapat mengakses fitur manajemen sistem.

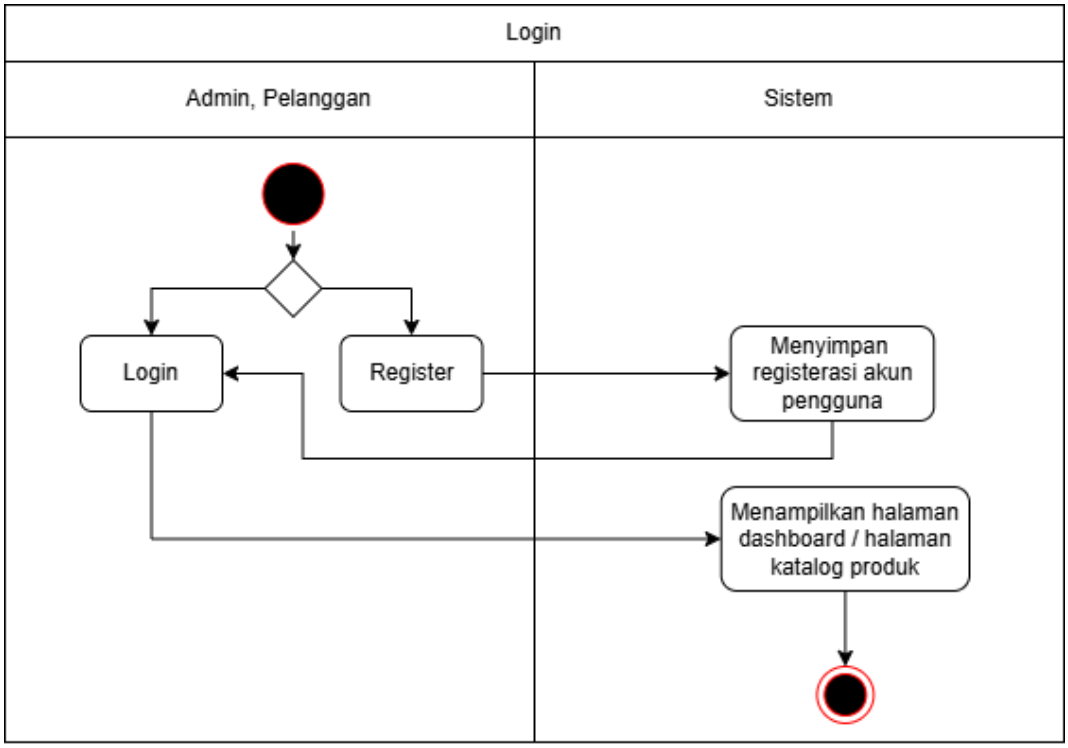
UC2	Admin	Mengakses dashboard	Setelah login, admin dapat melihat data ringkasan seperti jumlah pelanggan, produk, dan pesanan.
UC3	Admin	Mengelola kategori	Admin dapat menambahkan, mengedit, dan menghapus data kategori sebelum menambahkan produk.
UC4	Admin	Mengelola produk	Admin dapat menambahkan, mengedit, dan menghapus data produk dalam katalog.
UC5	Admin	Mengelola pesanan	Admin melihat daftar pesanan, memverifikasi pembayaran, dan memperbarui status pengiriman.
UC6	Admin, pelanggan	Melihat katalog produk	Admin dan pelanggan dapat melihat daftar produk yang tersedia pada halaman utama tanpa harus melakukan login terlebih dahulu.
UC7	Pelanggan	Login	Pelanggan masuk ke sistem menggunakan akun yang telah dibuat untuk melakukan transaksi.
UC8	Pelanggan	Berbelanja	Pelanggan memilih produk, mengisi data pemesanan, dan menyelesaikan transaksi pembelian.
UC9	Pelanggan	Mengecek status pesanan	Pelanggan dapat memantau status pesanan mereka

4.2.2 *Activity Diagram*

a) Login

Activity diagram login menggambarkan alur autentikasi awal yang dilakukan oleh pengguna sistem, baik itu admin maupun pelanggan. Proses ini dimulai ketika pengguna mengakses halaman login. Di halaman ini, pengguna diminta untuk memasukkan email dan kata sandi. Setelah pengguna menekan tombol “Login”, sistem akan menjalankan proses verifikasi terhadap data yang dimasukkan.

Jika data yang dimasukkan tidak valid, misalnya email belum terdaftar atau kata sandi salah, sistem akan memberikan notifikasi kesalahan dan meminta pengguna untuk mendaftarkan akun yg baru. Namun, jika data yang dimasukkan valid, maka sistem akan mengarahkan pengguna ke halaman sesuai dengan peran masing-masing. Pengguna dengan peran "admin" akan diarahkan ke dashboard admin untuk mengelola produk dan pesanan, sedangkan pelanggan akan diarahkan ke halaman katalog untuk mulai berbelanja. Proses ini penting karena menjadi gerbang utama untuk menjaga keamanan sistem serta memisahkan akses berdasarkan hak pengguna. Ilustrasi *Activity Diagram* Login dapat dilihat pada Gambar 4.2.



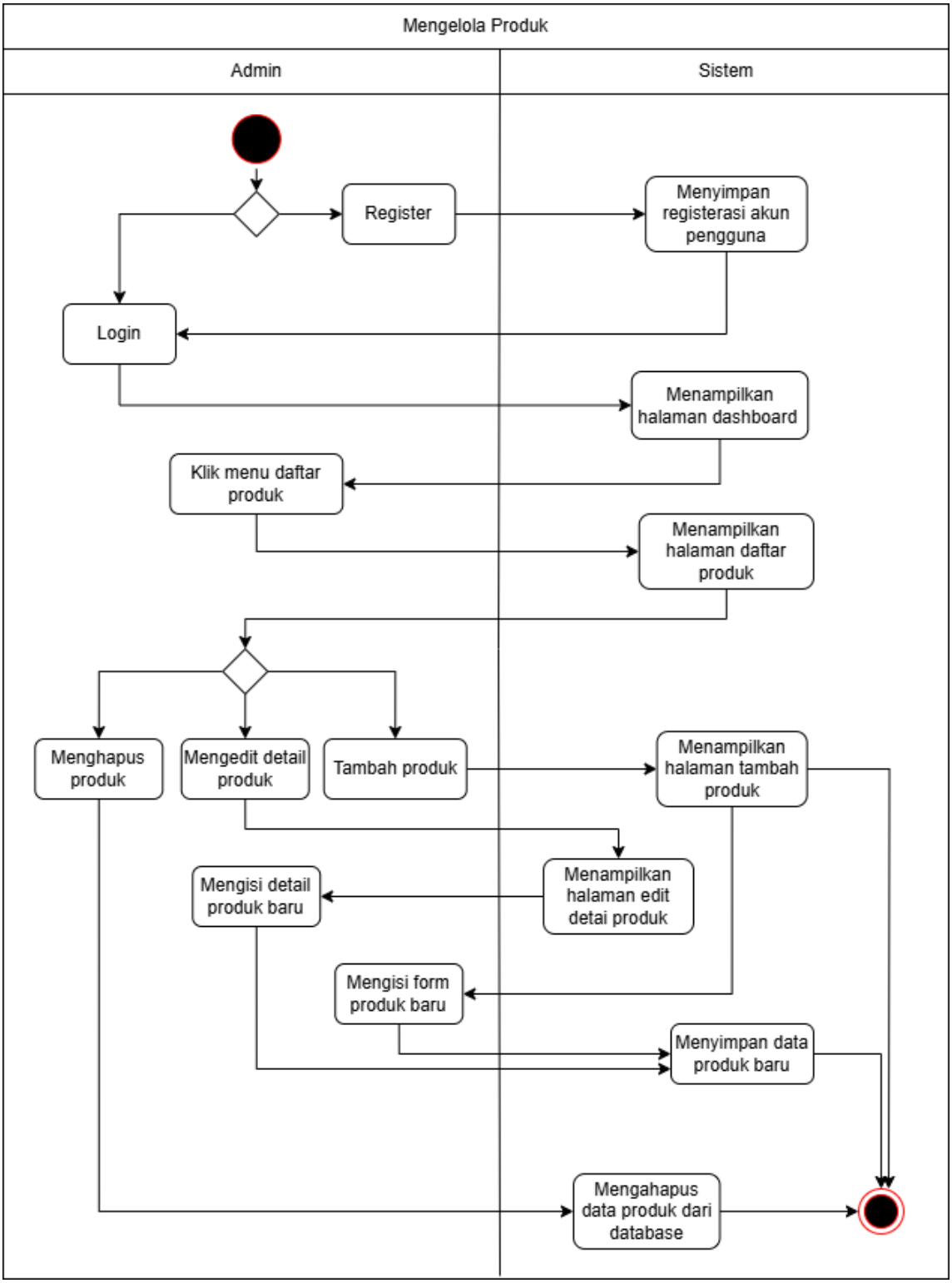
Gambar 4.2 *Activity Diagram* Login

b) Mengelola Produk

Diagram ini berfokus pada alur aktivitas admin dalam mengelola data produk yang tersedia di toko online. Proses ini hanya dapat diakses setelah admin berhasil login ke dalam sistem. Admin akan mengakses menu “Daftar Produk” yang menampilkan daftar produk yang sudah ada.

Di dalam menu ini, admin dapat melakukan tiga aktivitas utama: menambahkan produk baru, mengedit produk yang sudah ada, dan menghapus produk. Untuk menambahkan produk, admin harus mengisi formulir yang mencakup nama produk, deskripsi, harga, stok, dan gambar produk. Data ini akan divalidasi oleh sistem.

Begitu data valid, sistem akan menyimpan data produk ke dalam basis data dan menampilkan produk tersebut di katalog. Proses pengeditan dan penghapusan produk juga mengikuti alur serupa. Edit produk akan memperbarui data yang sudah ada dengan mengisi formular edit produk, sementara hapus produk akan menghapus data dari basis data. Diagram ini menggambarkan pentingnya kontrol dan fleksibilitas yang dimiliki admin untuk menjaga kelengkapan dan keakuratan informasi produk yang ditampilkan ke pelanggan. Ilustrasi *Activity Diagram* mengelola produk dapat dilihat pada Gambar 4.3.



Gambar 4.3 *Activity Diagram* Mengelola Produk

c) Mengelola Kategori

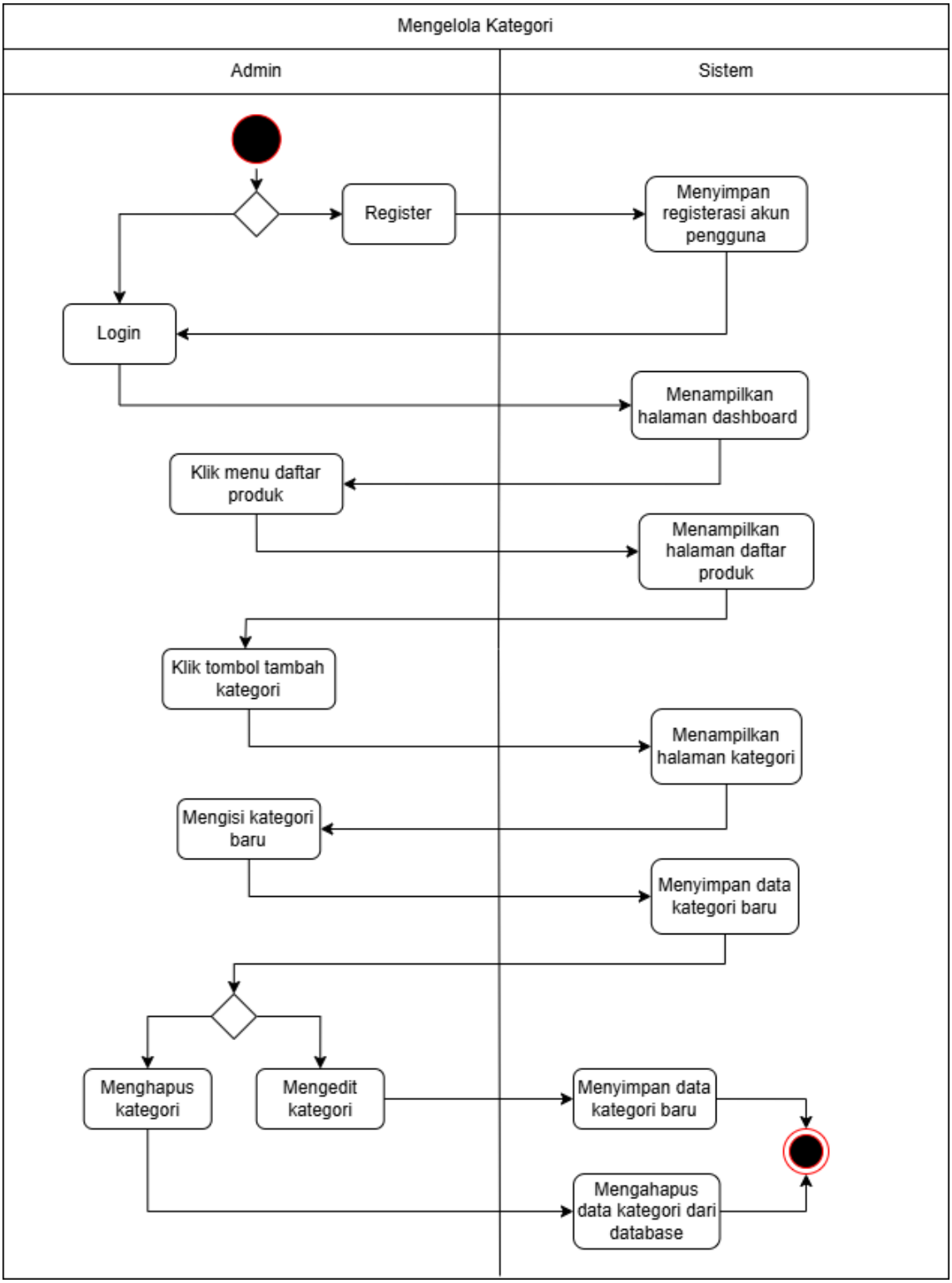
Diagram ini menggambarkan proses aktivitas admin dalam mengelola data kategori produk yang tersedia pada sistem e-commerce. Akses ke fitur ini hanya dapat dilakukan setelah admin berhasil login ke dalam sistem. Jika belum memiliki akun, admin perlu melakukan registrasi terlebih dahulu. Setelah berhasil login, sistem akan menampilkan halaman dashboard sebagai halaman awal bagi admin.

Dari dashboard, admin dapat mengakses menu “Daftar Produk” yang kemudian mengarahkan ke halaman daftar produk. Di dalam halaman ini, admin dapat memilih untuk menambahkan kategori baru dengan mengklik tombol “Tambah Kategori”. Setelah tombol diklik, sistem akan menampilkan halaman kategori yang berisi formulir pengisian data kategori.

Admin mengisi nama kategori baru sesuai kebutuhan. Data yang diinputkan akan diproses dan divalidasi oleh sistem. Apabila data valid, sistem akan menyimpan kategori baru tersebut ke dalam basis data.

Selain menambah kategori, admin juga memiliki kemampuan untuk mengelola kategori yang sudah ada. Admin dapat memilih untuk mengedit kategori tertentu dengan mengubah datanya melalui formulir yang tersedia, lalu sistem akan memperbarui data tersebut di dalam basis data. Admin juga dapat menghapus kategori jika dirasa tidak diperlukan lagi, dan sistem akan secara otomatis menghapus data kategori tersebut dari basis data.

Diagram ini menegaskan peran admin dalam menjaga struktur pengkategorian produk agar tetap rapi dan relevan, sehingga memudahkan pelanggan dalam menelusuri produk berdasarkan kategori yang telah ditentukan. Ilustrasi Activity Diagram mengelola kategori dapat dilihat pada Gambar 4.4.



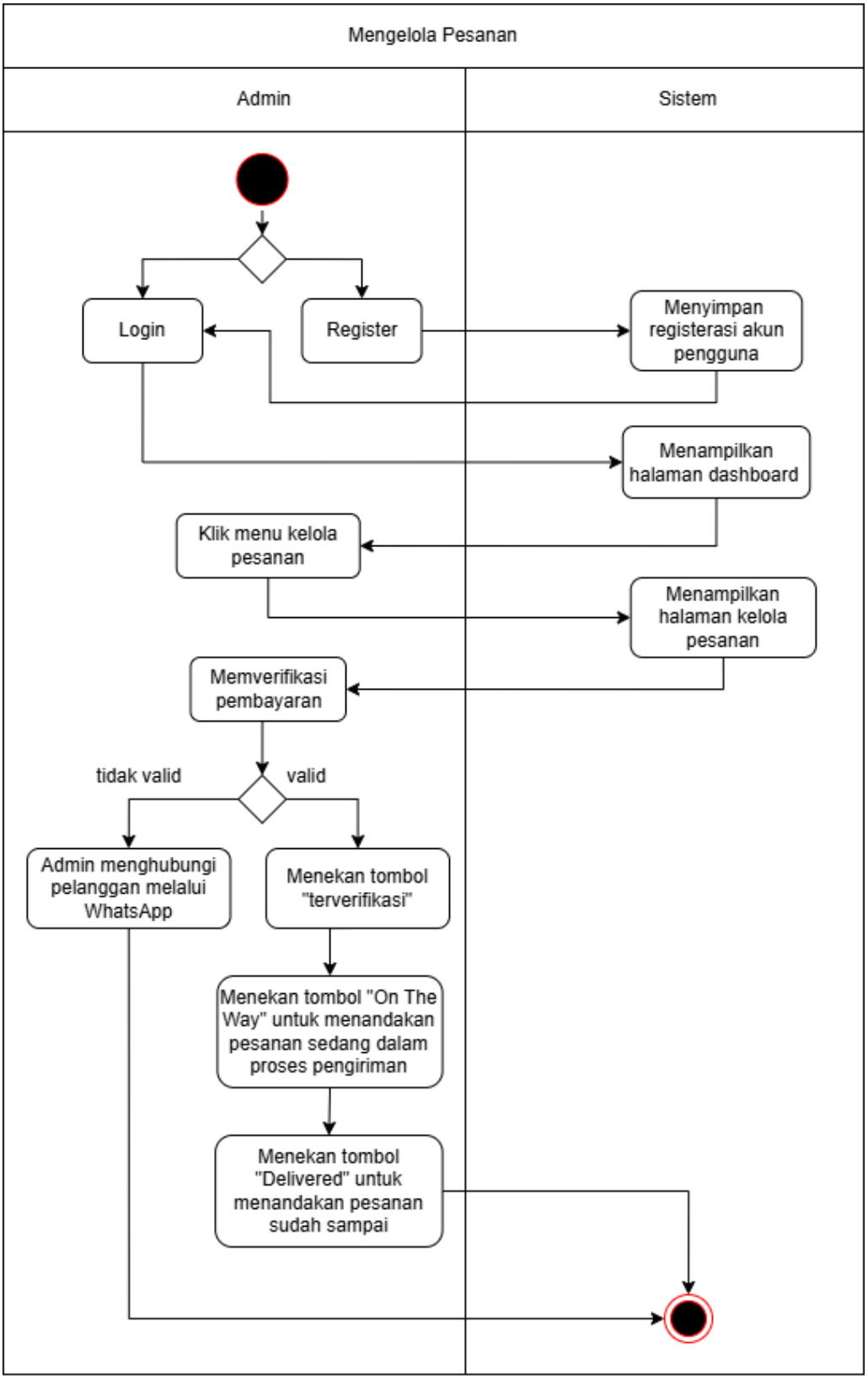
Gambar 4.4 *Activity Diagram* Mengelola Kategori

d) Mengelola Pesanan

Diagram ini menunjukkan proses manajemen pesanan yang dilakukan oleh admin. Setelah login, admin dapat mengakses menu "Kelola Pesanan" yang menampilkan daftar semua pesanan yang masuk dari pelanggan. Dalam daftar ini, admin dapat melihat informasi seperti nama pelanggan, Alamat pelanggan, nomer telepon pelanggan, nama produk yang dipesan, total harga, gambar produk yang dipesan, bukti pembayaran, dan status pesanan.

Admin memiliki kewenangan untuk memperbarui status pesanan berdasarkan tahapan pemrosesan, seperti: "Menunggu konfirmasi", "Sedang diproses", "Dikirim", hingga "Selesai". Misalnya, setelah pelanggan melakukan checkout, status awal pesanan adalah "Menunggu konfirmasi". Setelah admin memverifikasi pesanan dan mulai menyiapkan pengiriman, status bisa diubah menjadi "Sedang diproses".

Selain itu, admin juga dapat membatalkan pesanan apabila ditemukan kesalahan atau kendala tertentu, seperti stok kosong atau pembayaran gagal. Proses ini dilakukan dengan hati-hati dan akan memicu sistem untuk mengirimkan notifikasi status terbaru kepada pelanggan. Diagram ini memperlihatkan peran sentral admin dalam memastikan kelancaran transaksi dan transparansi kepada pelanggan. Ilustrasi *Activity diagram* mengelola pesanan dapat dilihat pada Gambar 4.5.



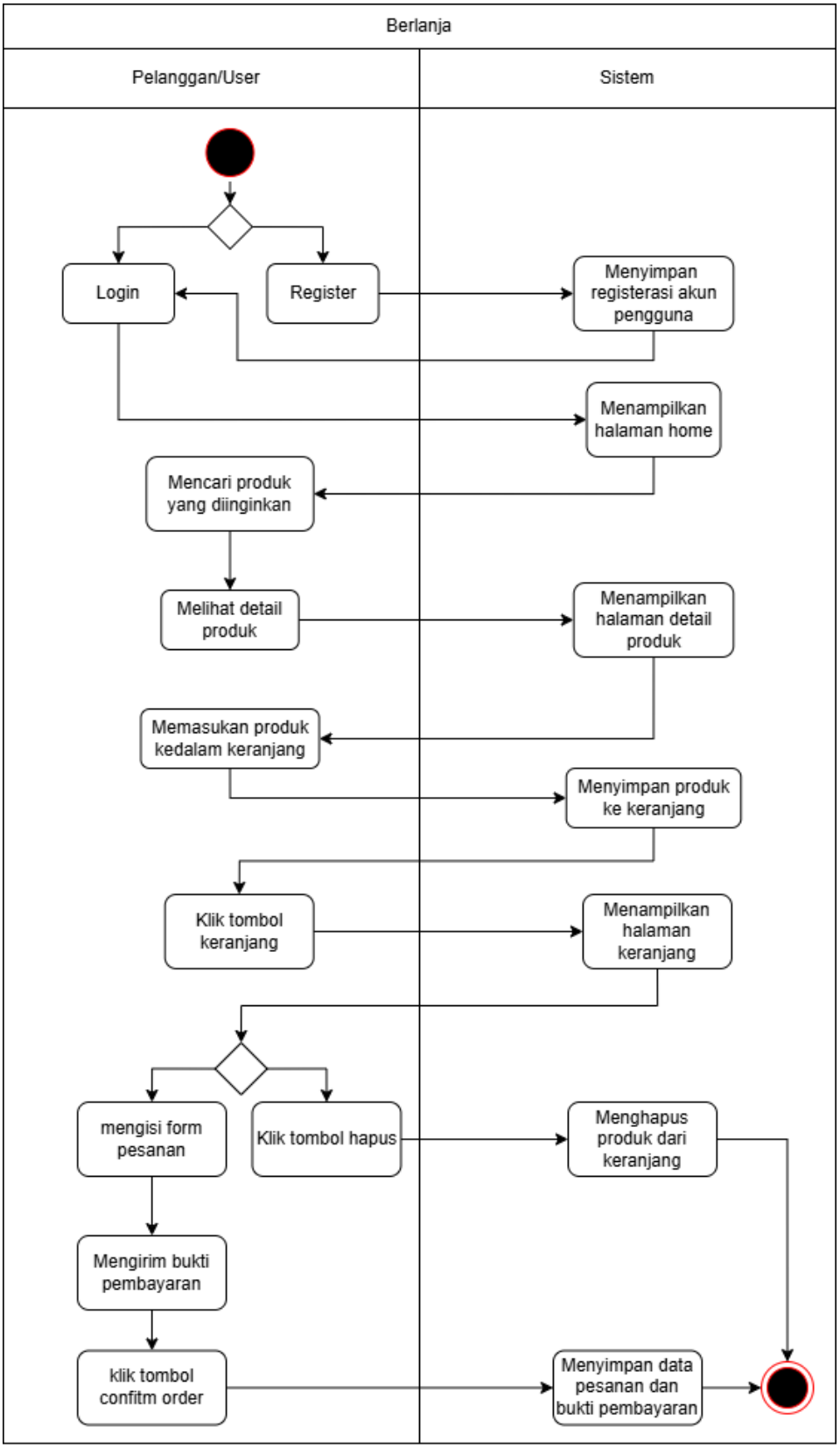
Gambar 4.5 *Activity Diagram* Mengelola Pesanan

e) Berbelanja

Activity diagram ini merupakan inti dari fungsionalitas sistem e-commerce karena menggambarkan seluruh proses transaksi pembelian yang dilakukan oleh pelanggan. Proses dimulai saat pelanggan menelusuri katalog produk. Setelah menemukan produk yang diinginkan, pelanggan dapat melihat deskripsi dengan menekan gambar produk atau jika ingin langsung menyimpan produk ke keranjang dapat langsung menekan tombol "Tambah ke Keranjang".

Produk yang dipilih akan masuk ke dalam daftar keranjang belanja. Pelanggan bisa meninjau ulang keranjang untuk menghapus produk atau melanjutkan ke tahap checkout. Saat checkout, pelanggan akan diminta mengisi informasi pengiriman seperti alamat lengkap, nomor kontak, dan mengupload bukti pembayaran. Terdapat juga intruksi tata cara pembayaran sebelum melakukan konfirmasi pesanan.

Setelah data lengkap dan pelanggan mengkonfirmasi pembelian, sistem akan membuat entri pesanan baru dan menyimpan data tersebut ke basis data. Pelanggan akan diarahkan ke halaman "My Orders" untuk melihat pesanan yang berhasil dibuat. Diagram ini menunjukkan bagaimana sistem memfasilitasi transaksi dari awal hingga akhir secara terstruktur dan aman. Ilustrasi *Activity Diagram* berbelanja dapat dilihat pada Gambar 4.6.



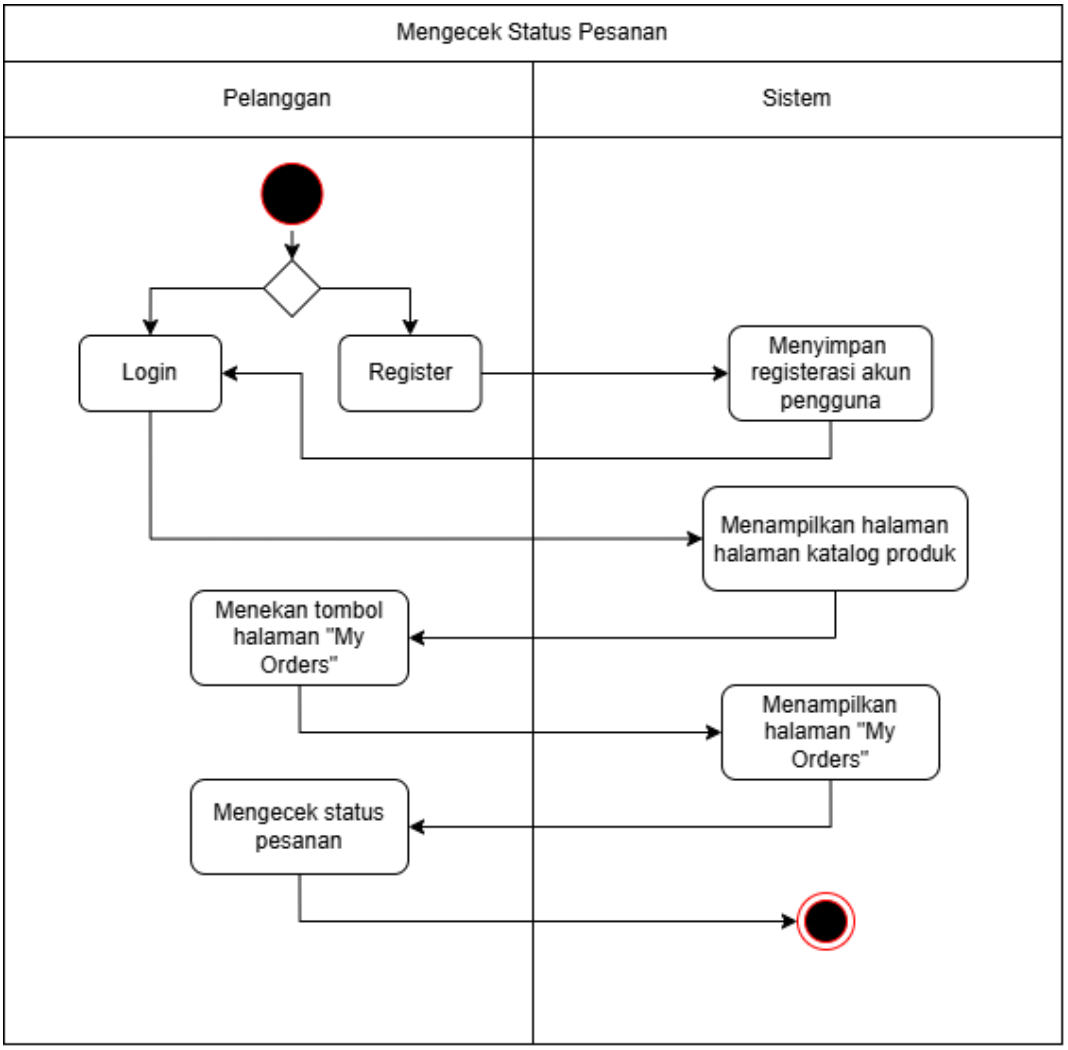
Gambar 4.6 *Activity Diagram* Berbelanja

f) Mengecek Status Pesanan

Activity diagram ini menggambarkan bagaimana pelanggan dapat memantau perkembangan pesanan yang telah mereka buat. Setelah login, pelanggan dapat mengakses halaman "My Orders" yang menampilkan daftar semua transaksi yang pernah mereka lakukan.

Setiap entri pada daftar tersebut memuat informasi penting, seperti nama produk, harga, status pembayaran, status pesanan, dan gambar produk yang dipesan. Status pesanan akan berubah secara dinamis mengikuti update yang dilakukan oleh admin, seperti "Sedang diproses", "Dikirim", atau "Selesai".

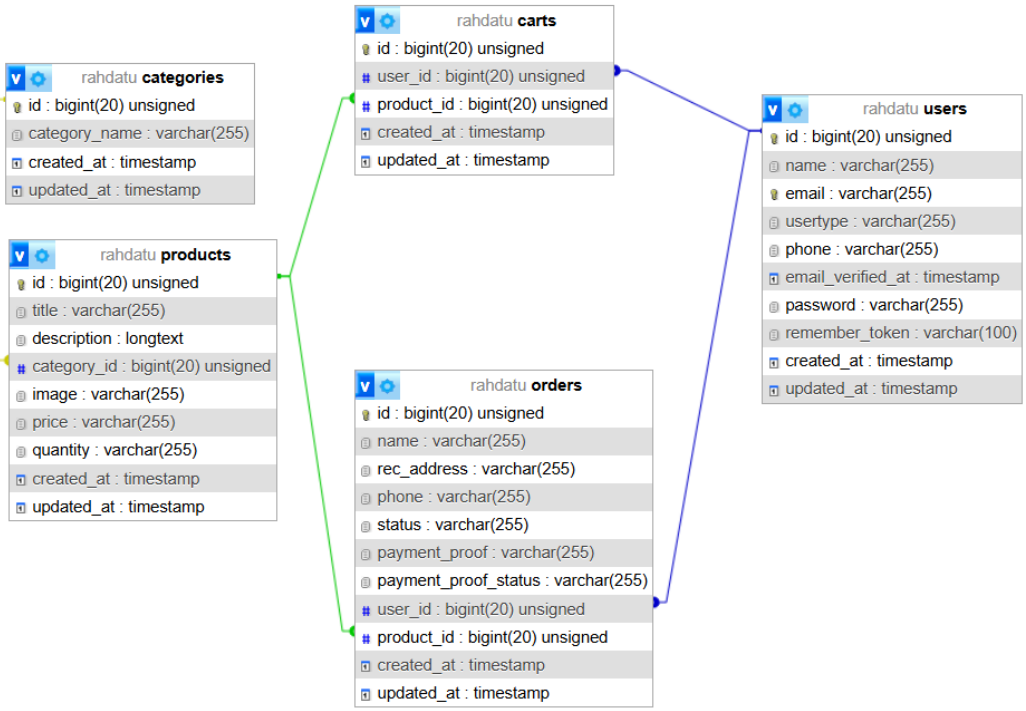
Fitur ini memberikan rasa aman dan transparansi kepada pelanggan, karena mereka bisa mengetahui apakah pesanan mereka sudah diproses atau sedang dalam pengiriman. Jika tersedia, sistem juga bisa menampilkan nomor resi untuk pelacakan secara langsung. Diagram ini mendemonstrasikan interaksi penting antara sistem dan pelanggan untuk membangun kepercayaan dan kepuasan pengguna. Ilustrasi *Activity Diagram* mengecek status pesanan dapat dilihat pada Gambar 4.7.



Gambar 4.7 *Activity Diagram* Mengecek Status Pesanan

4.2.3 Rancangan Basis Data

Rancangan basis data pada sistem *e-commerce* Rahdatu Wooden Furniture disusun untuk mendukung kebutuhan penyimpanan data berdasarkan *use case* yang telah dianalisis sebelumnya. Tujuannya adalah agar setiap fitur utama, baik dari sisi admin maupun pelanggan memiliki struktur data yang tepat, efisien, dan mudah dikelola. Adapun rancangan desain basis data dapat dilihat pada Gambar 4.8.



Gambar 4.8 Rancangan Basis Data

Tabel *users* dibuat untuk menyimpan data seluruh pengguna sistem, baik pelanggan maupun *admin*. Tabel ini berkaitan langsung dengan *use case Login* (UC1 & UC7) dan menjadi identitas dasar dalam setiap aktivitas sistem, termasuk proses pemesanan dan manajemen data oleh *admin*.

Tabel *products* berfungsi untuk menyimpan informasi detail produk yang ditampilkan pada katalog dan dikelola oleh *admin*. Data yang terdapat dalam tabel ini digunakan dalam *use case* Mengelola Produk (UC4), karena semua interaksi pelanggan dan *admin* terhadap produk—seperti melihat katalog, menambahkan ke keranjang, atau mengelola stok—berdasarkan data dari tabel ini.

Tabel *categories* menyimpan informasi tentang klasifikasi atau kategori produk, seperti "Meja", "Kursi", dan sebagainya. Tabel ini sangat penting dalam *use case* Mengelola Kategori (UC3), di mana *admin* dapat menambah, mengubah, atau menghapus kategori produk. Selain itu, karena setiap produk memiliki relasi ke kategori, tabel ini juga mendukung *use case* Mengelola Produk (UC4) sebagai bagian dari pengelompokan produk berdasarkan jenisnya.

Tabel *orders* digunakan untuk merekam setiap transaksi pembelian yang dilakukan oleh pelanggan. Informasi yang disimpan meliputi data pengiriman, status pembayaran, dan bukti pembayaran. Tabel ini mendukung beberapa *use case* penting, yaitu Berbelanja (UC8), Mengecek Status Pesanan (UC9) oleh pelanggan, serta Mengelola Pesanan (UC5) oleh admin.

Tabel *carts* dirancang untuk menampung sementara produk-produk yang dipilih pelanggan sebelum mereka menyelesaikan pembelian (checkout). Fungsinya sangat penting dalam alur pembelian

yang terdapat pada *use case* Berbelanja (UC8), sebagai tahap awal sebelum pelanggan melakukan pemesanan dan pembayaran.

4.3 Implementasi

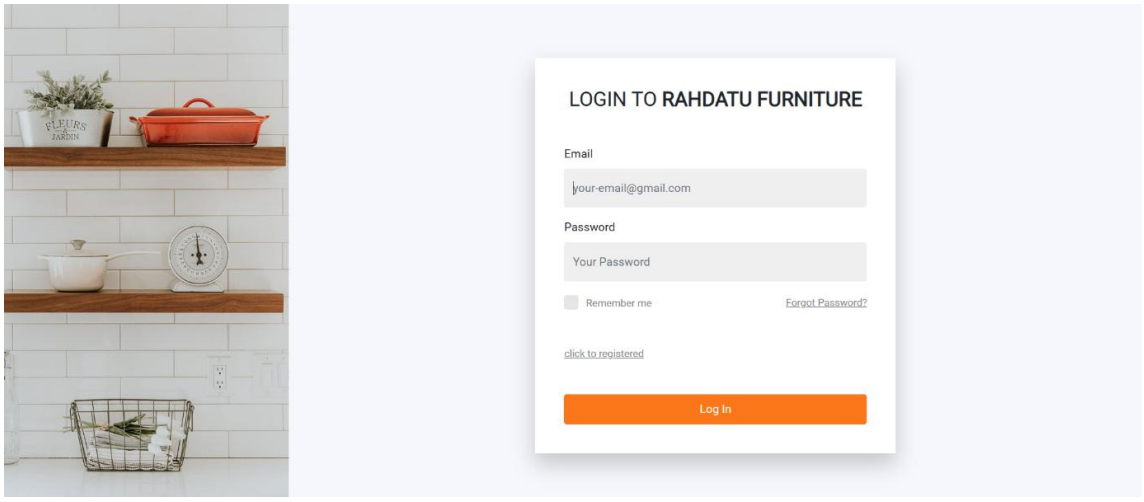
Pada tahap ini dijelaskan hasil rancangan Antar muka dari *website e-commerce* yang telah dikembangkan. Antar muka dirancang untuk memberikan kemudahan dalam navigasi dan penggunaan, baik bagi pelanggan maupun admin. Berikut ini merupakan penjelasan dari masing-masing tampilan Antar muka telah dirancang.

4.3.1 Login dan Register

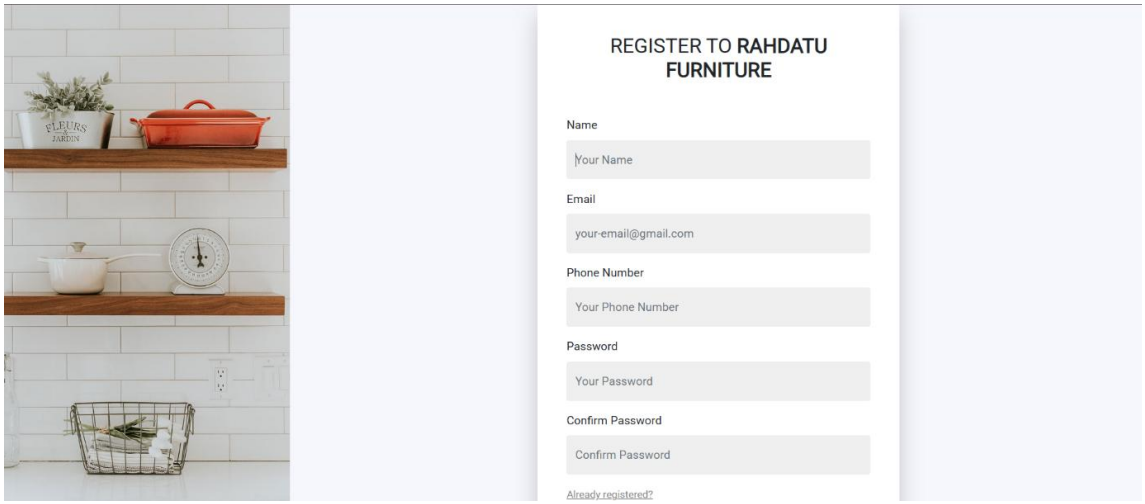
Halaman Login digunakan untuk mengautentikasi pengguna yang telah terdaftar sebelum mengakses fitur-fitur utama dari website Rahdatu Furniture. Antar muka halaman login dirancang secara minimalis dengan formulir isian email dan kata sandi, serta fitur tambahan seperti *checkbox* "*Remember me*", tautan "*Forgot Password*?", dan navigasi ke halaman registrasi.

Halaman Registerasi memungkinkan pengguna baru untuk membuat akun di website Rahdatu Furniture. Formulir pendaftaran mencakup isian nama lengkap, email, nomor telepon, kata sandi, dan konfirmasi kata sandi. Antar muka ini dirancang secara intuitif agar pengguna dapat melakukan registrasi dengan mudah. Tersedia juga tautan navigasi untuk kembali ke halaman login bagi pengguna yang sudah memiliki akun.

Kedua halaman ini berkaitan dengan *use case* Login (UC 1 & UC 7). Tampilan halaman Login dan Registerasi dapat dilihat pada Gambar 4.9 dan Gambar 4.10.



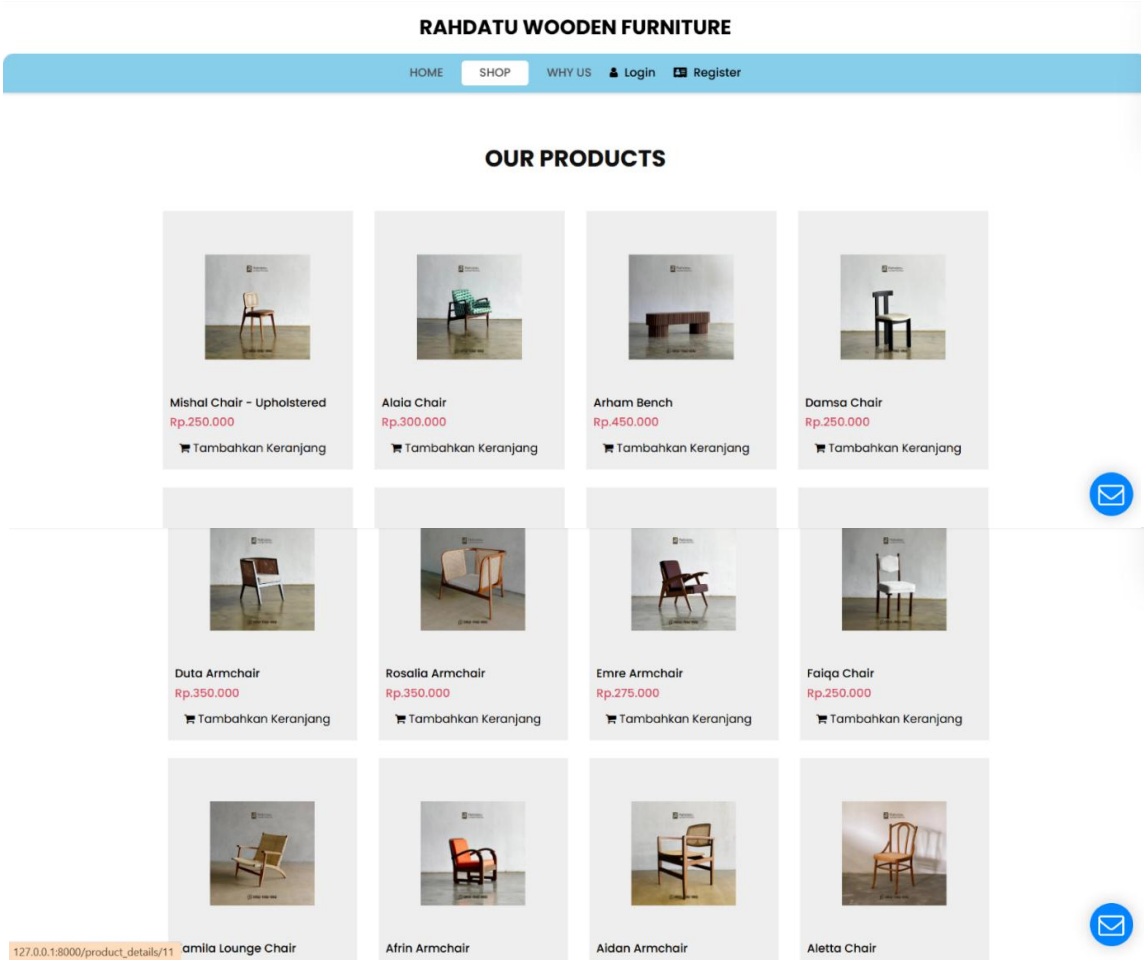
Gambar 4.9 Tampilan Halaman Login



Gambar 4.10 Tampilan Halaman *Register*

4.3.2 Tampilan Halaman Katalog Produk

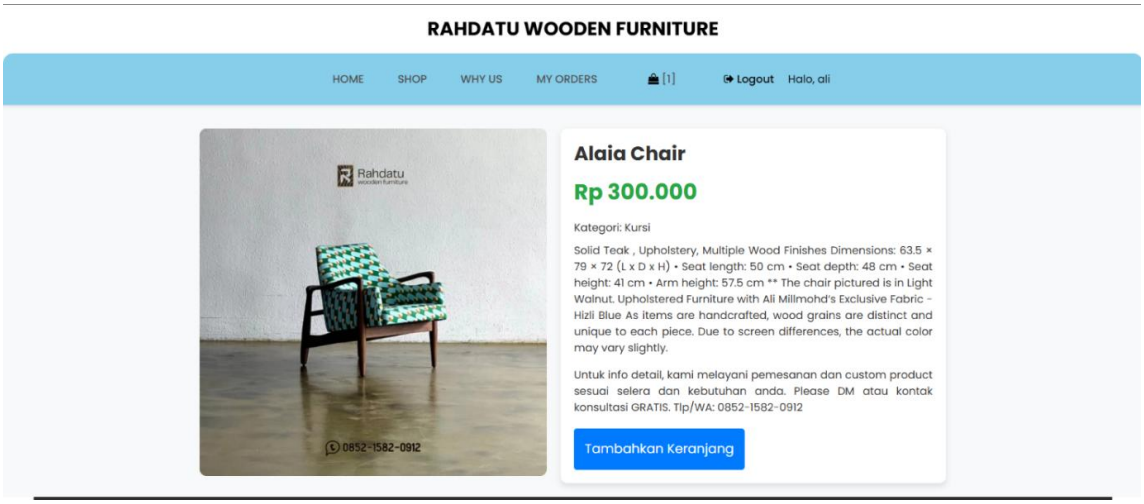
Halaman ini menampilkan seluruh produk furnitur yang tersedia dalam bentuk daftar visual. Masing-masing produk ditampilkan dalam kartu (*card*) yang memuat gambar, nama, harga, dan tombol "Tambahkan Keranjang". Jika pelanggan ingin melihat detail dari produk yang diminati, pelanggan dapat menekan gambar produk untuk berpindah ke halaman Detail Produk. Halaman ini berkaitan dengan *use case* melihat katalog produk (UC6) dan berbelanja (UC8). Tampilan halaman katalog produk dapat dilihat di Gambar 4.11.



Gambar 4.11 Halaman Katalog Produk

4.3.3 Tampilan Halaman Detail Produk

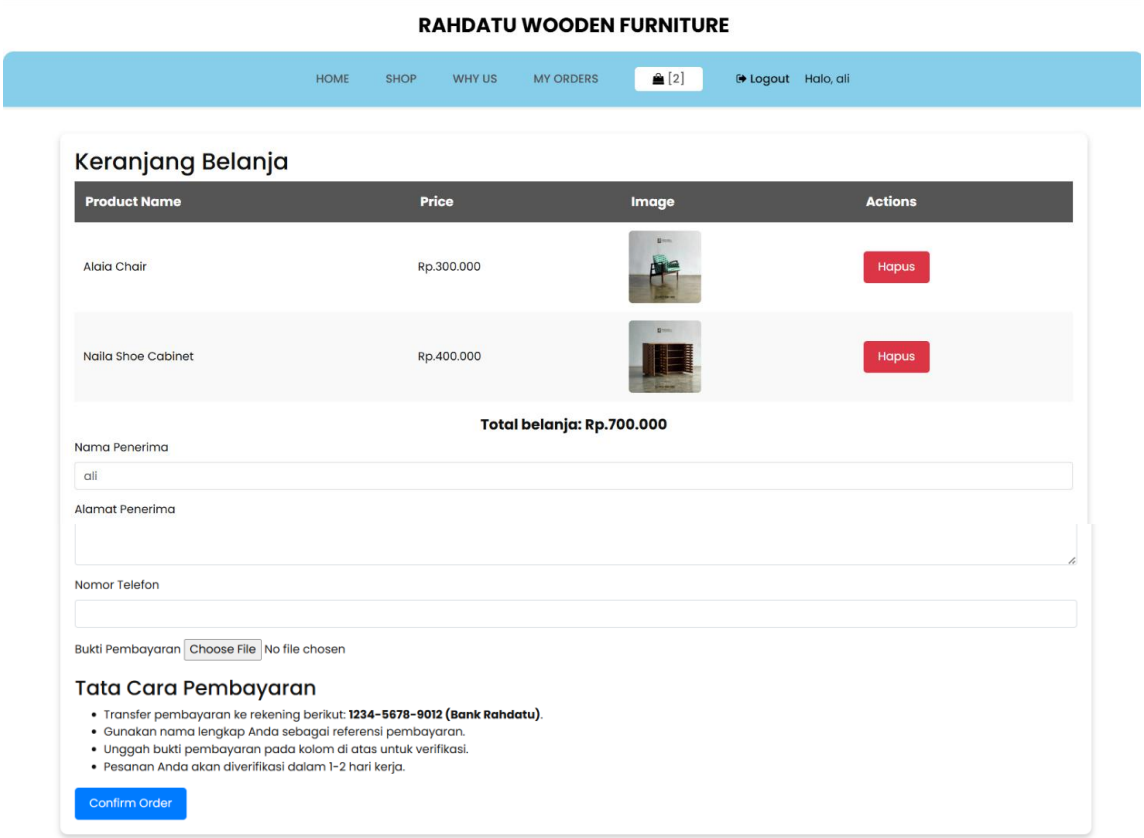
Tampilan ini memberikan informasi detail dari produk yang dipilih. Antar muka menampilkan gambar produk berukuran besar, deskripsi lengkap, harga, serta informasi stok. Disediakan tombol untuk menambahkan produk ke keranjang. Halaman ini berkaitan dengan *use case* melihat katalog produk (UC6) dan berbelanja (UC8). Tampilan halaman detail produk dapat dilihat di Gambar 4.12.



Gambar 4.12 Halaman Detail Produk

4.3.4 Tampilan Halaman Keranjang Belanja

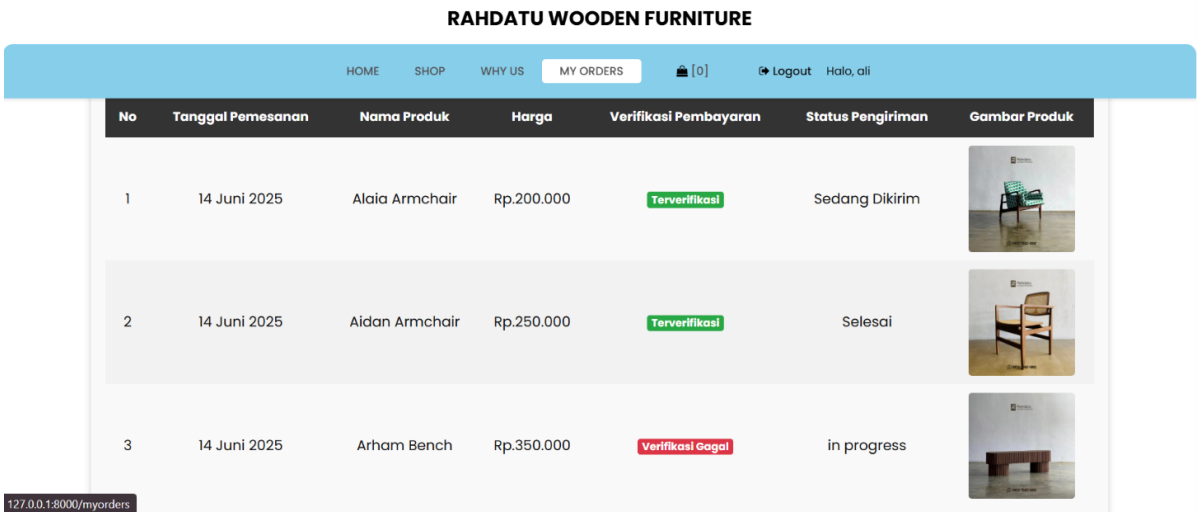
Pada tampilan ini, pengguna dapat melihat daftar produk yang telah dimasukkan ke dalam keranjang belanja. Setiap item menampilkan gambar, nama, kuantitas, harga satuan, dan subtotal. Pengguna dapat menghapus item dari keranjang. Di bagian bawah tabel terdapat informasi total harga pembelian belanja yang harus dibayarkan dan tombol untuk melanjutkan proses *checkout* dengan mengisi form nama penerima, alamat penerima, nomor telefon, dan bukti pembayaran berupa gambar. Halaman ini berkaitan dengan *use case* berbelanja (UC8). Tampilan halaman keranjang belanja dapat dilihat di Gambar 4.13.



Gambar 4.13 Halaman Keranjang Belanja

4.3.5 Tampilan Halaman Riwayat Pesanan

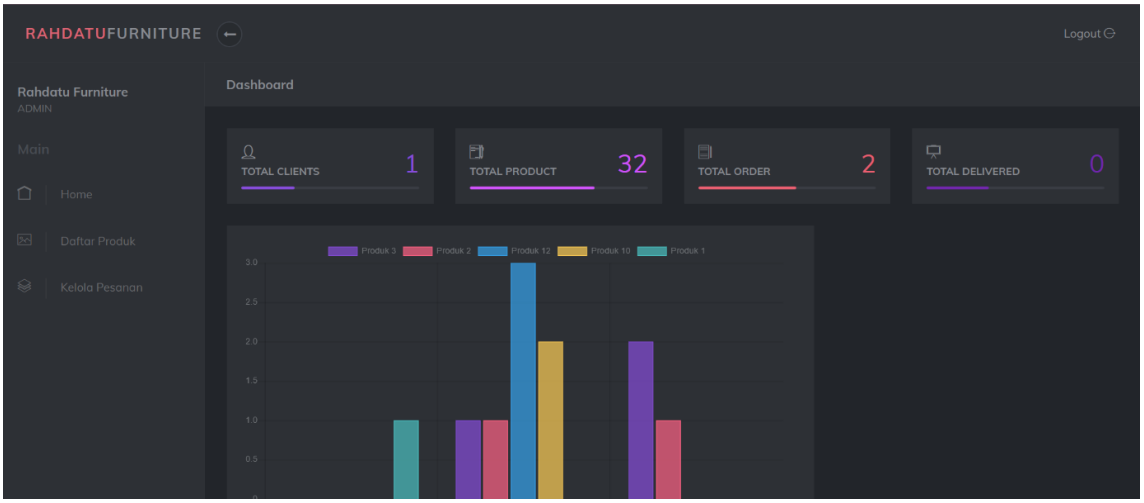
Halaman ini memungkinkan pengguna untuk melihat daftar transaksi yang telah mereka lakukan. Setiap pesanan ditampilkan dalam bentuk tabel yang mencakup Nomor Pesanan, Tanggal Pemesanan, Nama Produk, Harga, Verifikasi Pembayaran, Status Pengiriman, dan Gambar Produk. Setiap baris mewakili satu item pesanan, bukan per transaksi, sehingga pengguna dapat melihat detail setiap produk yang dibeli. Teks “Terverifikasi” ditampilkan dalam bentuk label berwarna hijau pada kolom verifikasi pembayaran, menandakan bahwa pembayaran telah berhasil divalidasi. Sementara itu, status pengiriman ditampilkan dalam teks "*in progress*" sebagai status *default* yang menunjukkan pesanan yang sedang di proses oleh Admin. Di sisi kanan, terdapat gambar mini produk yang memudahkan pengguna untuk mengenali barang yang dipesan. Halaman ini berkaitan dengan *use case* mengecek status pesanan (UC9). Tampilan halaman riwayat pesanan dapat dilihat di Gambar 4.14.



Gambar 4.14 Halaman Riwayat Pesanan

4.3.6 Tampilan Halaman *Dashboard* Admin

Merupakan tampilan utama admin setelah login. *Dashboard* menyajikan informasi ringkasan seperti jumlah akun pelanggan terdaftar, total produk, jumlah pesanan, dan grafik penjualan. Tujuannya adalah memberikan informasi cepat mengenai performa toko *online*. Halaman ini berkaitan dengan *use case* mengakses *dashboard* (UC2). Tampilan halaman *dashboard* admin dapat dilihat di Gambar 4.15.

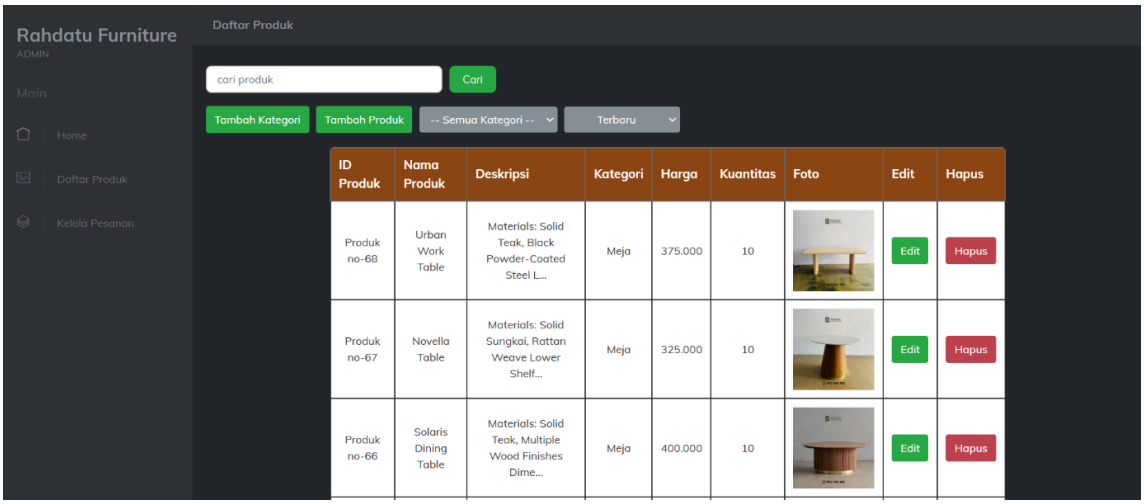


Gambar 4.15 Halaman *Dasboard* Admin

4.3.7 Tampilan Halaman Daftar Produk Admin

Tampilan ini menyajikan seluruh produk yang telah dimasukkan ke dalam sistem. Disusun dalam bentuk tabel, setiap baris mencantumkan ID Produk, Nama Produk, Kategori, Harga, Kuantitas Stok, dan Foto Produk. Terdapat tombol edit dan hapus untuk memudahkan admin

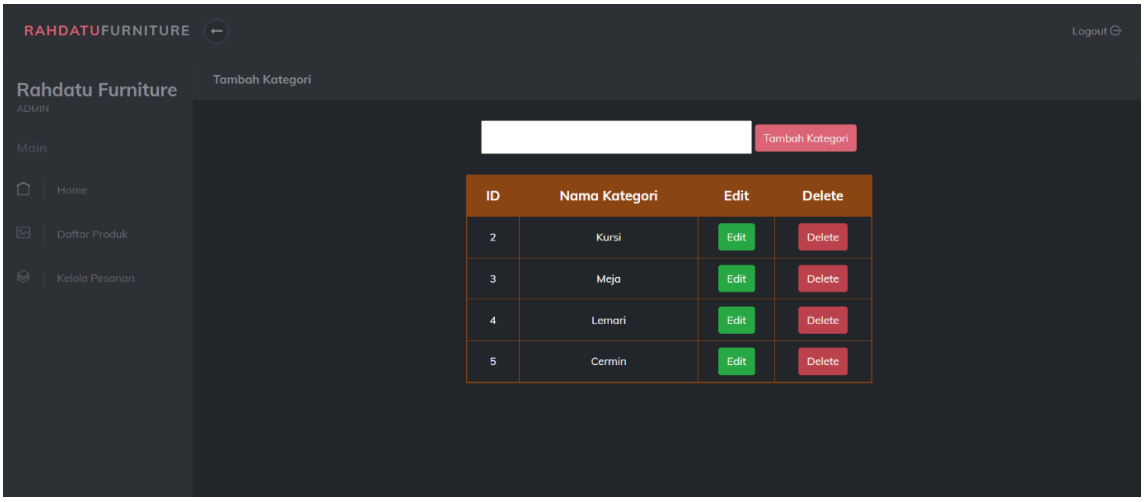
dalam melakukan pengelolaan. Selain itu, tersedia fitur pencarian produk, fitur filter produk berdasarkan kategori serta fitur sorting list yang memungkinkan admin untuk mengurutkan produk berdasarkan kriteria tertentu, seperti nama, harga, dan kategori. Halaman ini berkaitan dengan *use case* mengelola produk (UC4). Tampilan halaman daftar produk dapat dilihat di Gambar 4.16.



Gambar 4.16 Halaman Daftar Produk

4.3.8 Tampilan Halaman Kelola Kategori

Halaman ini memungkinkan admin untuk mengelola data kategori produk yang tersedia di sistem. Di bagian atas, terdapat kolom input dan tombol “Tambah Kategori” yang digunakan untuk menambahkan kategori baru. Sementara itu, tabel di bawahnya menampilkan daftar kategori yang sudah ada, lengkap dengan ID, nama kategori, serta tombol “Edit” dan “Delete” untuk memperbarui atau menghapus kategori sesuai kebutuhan. Fitur ini memudahkan admin dalam menjaga struktur dan pengelompokan produk secara teratur. Halaman ini berkaitan dengan use case mengelola kategori (UC3). Tampilan halaman kategori dapat dilihat pada Gambar 4.17.



Gambar 4.17 Tampilan Halaman Kelola Kategori

4.3.9 Tampilan Halaman Tambah Produk Admin

Tampilan ini memungkinkan admin untuk menambahkan produk baru ke dalam sistem. Formulir input mencakup nama produk, deskripsi, harga, jumlah stok, kategori, dan unggah gambar. Setelah semua data diisi, admin dapat menyimpan produk dengan menekan tombol "Simpan". Halaman ini berkaitan dengan *use case* mengelola produk (UC4). Tampilan halaman tambah produk dapat dilihat di Gambar 4.18.

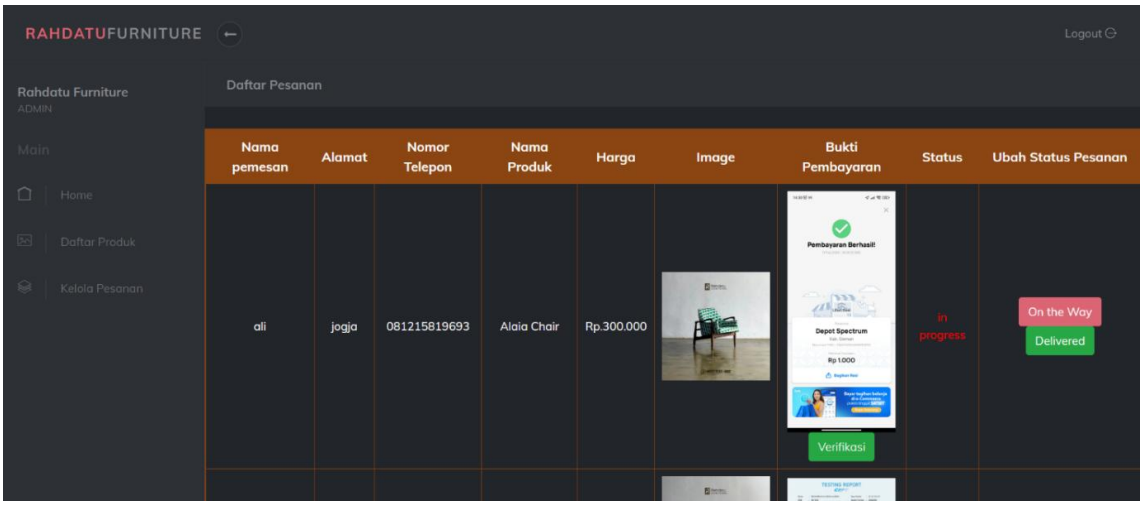


Gambar 4.18 Halaman Tambah Produk

4.3.10 Tampilan Halaman Kelola Pesanan Admin

Pada halaman ini, admin dapat memantau dan mengelola pesanan dari seluruh pengguna. Tersedia daftar pesanan berbentuk tabel yang menampilkan nama pemesan, alamat, nomor

telepon pemesan, nama produk yang dipesan, harga produk, gambar produk, bukti pembayaran, status, dan tombol ubah status pesanan. Admin dapat memverifikasi pembayaran dengan melihat bukti pembayaran yang dilampirkan. Halaman ini berkaitan dengan *use case* mengelola pesanan (UC5). Tampilan halaman kelola pesanan dapat dilihat di Gambar 4.19.



Gambar 4.19 Halaman Kelola Pesanan

4.4 Implementasi Kode Program

Pada tahap ini dijelaskan implementasi kode program untuk fitur-fitur yang telah dikembangkan. Dalam pengembangannya penulis menggunakan *framework* Laravel dan PHP sebagai bahasa pemrogramannya. Laravel dipilih karena mendukung arsitektur Model-View-Controller (MVC) yang memudahkan pengelolaan struktur kode dan pengembangan fitur-fitur sistem. Terdapat dua *Controller* yang menyimpan seluruh fungsi-fungsi fitur, yaitu *AdminController* (untuk menyimpan fungsi-fungsi yang dipakai di halaman Admin) dan *HomeController* (untuk menyimpan fungsi-fungsi yang dipakai di halaman utama).

4.4.1 Fitur Login dan Registrasi

Fitur login dan registrasi pada sistem ini berfungsi sebagai pintu masuk bagi pengguna, baik admin maupun pelanggan, untuk dapat mengakses fitur-fitur sesuai hak akses masing-masing. Proses registrasi memungkinkan pengguna mendaftarkan akun secara mandiri sebagai pelanggan. Setiap data yang diinput seperti nama, email, nomor telepon, dan password akan divalidasi sebelum disimpan ke dalam *database*. Implementasi formulir registerasi pengguna dapat dilihat pada Gambar 4.18.

```
<form method="POST" action="{{ route('register') }}">
    @csrf
    <!-- Name Field -->
    <div class="form-group first">
```

```
        <label for="name">Name</label>
        <input    type="text"    class="form-control"    id="name"    name="name"
placeholder="Your   Name"   value="{{   old('name')   }}"   required   autofocus
autocomplete="name">
        @error('name')
            <span class="text-danger">{{ $message }}</span>
        @enderror
    </div>
    <!-- Email Field -->
    <div class="form-group">
        <label for="email">Email</label>
         <input   type="email"   class="form-control"   id="email"   name="email"
placeholder="your-email@gmail.com"    value="{{    old('email')    }}"    required
autocomplete="username">
        @error('email')
            <span class="text-danger">{{ $message }}</span>
        @enderror
    </div>
    <!-- Phone Number Field -->
    <div class="form-group">
        <label for="phone">Phone Number</label>
             <input   type="text"   class="form-control"   id="phone"   name="phone"
placeholder="Your  Phone  Number"  value="{{  old('phone')  }}"  required  autofocus
autocomplete="phone">
            @error('phone')
              <span class="text-danger">{{ $message }}</span>
            @enderror
    </div>
    <!-- Password Field -->
    <div class="form-group">
        <label for="password">Password</label>
        <input type="password" class="form-control" id="password" name="password"
placeholder="Your Password" required autocomplete="new-password">
        @error('password')
            <span class="text-danger">{{ $message }}</span>
        @enderror
    </div>
    <!-- Confirm Password Field -->
    <div class="form-group">
        <label for="password_confirmation">Confirm Password</label>
          <input  type="password"  class="form-control"  id="password_confirmation"
name="password_confirmation"     placeholder="Confirm     Password"     required
autocomplete="new-password">
        @error('password_confirmation')
```

```
            <span class="text-danger">{{ $message }}</span>
        @enderror
    </div>
    <!-- Login Redirect and Submit Button -->
    <div class="d-sm-flex mb-5 align-items-center">
        <a href="{{ route('login') }}" class="forgot-pass">Already registered?</a>
   </div>
   <input type="submit" value="Register" class="btn btn-block py-2 btn-primary">
</form>
```

Gambar 4.20 Implementasi Formulir Registerasi Pengguna

Form ini terdiri dari beberapa input seperti nama, email, nomor telepon, kata sandi, dan konfirmasi sandi. Komponen @csrf digunakan untuk mencegah serangan CSRF (*Cross Site Request Forgery*), yang merupakan praktik keamanan standar dalam Laravel. Setiap isian diberi validasi *required* agar data yang masuk tidak kosong. Saat pengguna menekan tombol "Register", data dikirim ke *route register*, dan jika valid, akan disimpan ke dalam *database*.

Setelah berhasil mendaftar, sistem mengirimkan akan email verifikasi. Pengguna wajib memverifikasi alamat email terlebih dahulu sebelum dapat login. Hal ini dilakukan untuk menjaga validitas dan keamanan data pengguna. Implementasi penyimpanan data pengguna ke *database* dapat dilihat pada Gambar 4.21.

```
$user = User::create([
            'name' => $request->name,
            'email' => $request->email,
            'phone' => $request->phone,
            'password' => Hash::make($request->password),
        ]);
```

Gambar 4.21 Implementasi Penyimpanan Data Pengguna ke *Database*

Fungsi ini menggunakan metode create() untuk memasukkan data pengguna baru ke dalam tabel *users*. *Password* yang dimasukkan akan dienkripsi menggunakan Hash::make() untuk menjaga keamanan data sensitif. Data ini nantinya akan digunakan untuk proses *login*.

Sementara itu, ketika admin ingin membuat akun baru, maka akun tersebut akan terdaftar sebagai pelanggan secara default dan masuk ke halaman utama. Cara agar admin dapat memasuki halaman admin yaitu menetapkan atribut *usertype* sebagai admin secara manual melalui *database*. Dengan cara ini maka akses ke halaman admin akan lebih aman karena hanya admin yang mempunyai akses menuju *database*.

Setelah berhasil login, sistem akan melakukan pengecekan terhadap peran pengguna. Jika pengguna memiliki peran admin, maka akan diarahkan ke halaman *dashboard* admin. Sebaliknya, jika pengguna adalah *user* atau pelanggan, maka akan diarahkan ke halaman utama

katalog produk. Implementasi fungsi *redirect* berdasarkan peran dapat dilihat pada Gambar 4.21.

```
public function store(LoginRequest $request): RedirectResponse
    {
        $request->authenticate();
        $request->session()->regenerate();
        if($request->user()->usertype === 'admin')
        {
            return redirect('admin/dashboard');
        }
        return redirect()->intended(route('dashboard'));
    }
```

Gambar 4.22 Fungsi *Redirect* Berdasarkan Peran

Kode ini mengecek atribut *usertype* dari pengguna yang login. Jika pengguna bertipe 'admin', maka diarahkan ke dashboard admin. Jika tidak, maka pengguna masuk ke halaman utama pelanggan. Pendekatan ini efektif dalam memisahkan hak akses antara admin dan pelanggan tanpa perlu membuat dua sistem login yang terpisah.

Seluruh proses autentikasi ini diatur oleh sistem *routing* Laravel dan menggunakan *middleware* untuk mengamankan akses ke halaman-halaman tertentu berdasarkan peran pengguna. Dengan mekanisme ini, sistem memastikan bahwa hanya pengguna yang sesuai yang dapat mengakses fitur yang ditentukan. *Route* proses autentikasi dapat dilihat pada Gambar 4.23.

```
Route::middleware('auth')->group(function () {
             Route::get('/profile',      [ProfileController::class,      'edit'])-
>name('profile.edit');
           Route::patch('/profile',     [ProfileController::class,     'update'])-
>name('profile.update');
          Route::delete('/profile',    [ProfileController::class,    'destroy'])-
>name('profile.destroy');
});
require __DIR__.'/auth.php';
```

Gambar 4.23 *Route* Proses Autentikasi

Dengan penggunaan middleware auth, sistem memastikan bahwa hanya pengguna yang telah login yang bisa mengakses halaman tertentu. Ini menjaga integritas dan keamanan data, terutama pada fitur-fitur yang bersifat sensitif seperti manajemen produk atau pesanan.

4.4.2 Fitur Manajemen Kategori Produk

Fitur ini digunakan untuk membantu admin dalam mengelola daftar kategori produk yang tersedia di dalam sistem. Pengelolaan ini mencakup proses menampilkan, menambahkan, mengedit, dan menghapus kategori. Seluruh proses dikendalikan melalui fungsi-fungsi yang terdapat di dalam AdminController.php serta dihubungkan ke tampilan halaman kategori dan halaman edit kategori.

1. Menampilkan Daftar Kategori

Fungsi view_category() di dalam AdminController.php bertugas untuk mengambil seluruh data kategori dari database menggunakan perintah Kategori::all(). Data ini kemudian diteruskan ke tampilan admin.category dengan menggunakan fungsi *compact*, seperti ditunjukkan pada potongan kode pada Gambar 4.24.

```
public function view_category()
{
    $data = Kategori::all();
    return view('admin.category', compact('data'));
}
```

Gambar 4.24 *Function* Fitur Menampilkan Daftar Kategori

Di dalam halaman tampilan, data kategori ditampilkan dalam bentuk tabel. Setiap baris mencantumkan ID, nama kategori, serta tombol untuk melakukan proses edit dan hapus. Implementasi tampilan tabel kategori dapat dilihat pada Gambar 4.25.

```
<table class="table_deg">
    <tr>
        <th>ID</th>
        <th>Nama Kategori</th>
        <th>Edit</th>
        <th>Delete</th>
    </tr>

    @foreach($data as $item)

    <tr>
        <td>{{$item->id}}</td>
        <td>{{$item->category_name}}</td>
        <td><a class="btn btn-success" href="{{url('edit_category', $item-
>id)}}">Edit</a></td>
        <td><a class="btn btn-danger" onclick="confirmation(event)"
href="{{url('delete_category', $item->id)}}">Delete</a></td>
    </tr>
```

```
    @endforeach
</table>
```

Gambar 4.25 Tampilan Tabel Kategori

2. Menambahkan Kategori Baru

Untuk menambahkan kategori baru, admin dapat menggunakan formulir yang tersedia pada halaman yang sama. Formulir ini mengirim data *category* ke URL add_category melalui metode *POST*. Implementasi form tambah kategori dapat dilihat pada Gambar 4.26

```
<form action="{{url('add_category')}}" method="post">
@csrf
   <div>
        <input type="text" name="category">
        <input class="btn btn-primary" type="submit" value="Tambah Kategori">
    </div>
</form>
```

Gambar 4.26 Implementasi Form Tambah Kategori

Formulir ini diproses oleh fungsi add_category() dalam *controller*, yang akan menyimpan data kategori ke dalam database yang dapat dilihat pada Gambar 4.27.

```
public function add_category(Request $request)
{
    $data = new Kategori;
    $data->category_name = $request->category;
    $data->save();

    toastr()->timeOut(10000)->closeButton()->success('Category Added
Successfully');
    return redirect()->back();
}
```

Gambar 4.27 *Function* Menyimpan Data Kategori

3. Menghapus Kategori

Admin dapat menghapus kategori dengan menekan tombol "Delete" pada baris tabel kategori. Tombol ini akan memanggil URL delete_category/{id} yang di-handle oleh fungsi fitur hapus kategori seperti pada Gambar 4.28.

```
public function delete_category($id)
{
    $data = Kategori::find($id);
    $data->delete();

    toastr()->timeOut(5000)->closeButton()->success('Category Deleted
Successfully');
```

```
    return redirect()->back();
}
```

Gambar 4.28 *Function* Fitur Hapus Kategori

Fungsi ini mencari data berdasarkan ID dan kemudian menghapusnya dari tabel kategori.

4. Mengedit Kategori

Untuk memperbarui data kategori, admin dapat menekan tombol "Edit" yang akan mengarahkan ke halaman edit_category/{id}. Fungsi edit_category() akan memuat data kategori tersebut dan meneruskannya ke halaman edit yang dapat dilihat pada Gambar 4.29.

```
public function edit_category($id)
{
    $data = Kategori::find($id);
    return view('admin.edit_category', compact('data'));
}
```

Gambar 4.29 *Function* Edit Kategori

Pada halaman edit, data kategori ditampilkan di dalam input field sehingga admin dapat langsung memperbaruinya seperti pada Gambar 4.30.

```
<form action="{{ url('update_category_post/'.$data->id) }}" method="post">
    @csrf
    <input type="text" name="category" value="{{ $data->category_name }}">
    <input class="btn btn-primary" type="submit" value="Update Kategori">
</form>
```

Gambar 4.30 Tampilan *Input* Edit Data Kategori Baru

Setelah data diperbarui dan dikirim, fungsi update_category_post() akan menyimpan perubahan tersebut ke dalam database yang dapat dilihat pada Gambar 4.31.

```
public function update_category_post(Request $request, $id)
{
    $data = Kategori::find($id);
    $data->category_name = $request->category;
    $data->save();

    toastr()->timeOut(5000)->closeButton()->success('Category Updated
Successfully');
    return redirect('/view_category');
}
```

Gambar 4.31 *Function* Menyimpan Data Perubahan Kategori

4.4.3 Fitur Manajemen Produk

Fitur ini memungkinkan admin untuk mengelola data produk yang tersedia di marketplace. Seluruh proses seperti menambahkan, mengedit, dan menghapus produk dilakukan melalui *AdminController,* dengan dukungan *view* yang terstruktur dan mudah digunakan.

1. Fitur Tambah Produk

Fitur ini memungkinkan admin menambahkan produk baru ke katalog marketplace. Admin mengisi form input yang terdiri dari nama produk, deskripsi, harga, stok, dan gambar produk. Implementasi fitur tambah produk dapat dilihat pada Gambar 4.32.

```
<form action="{{ url('upload_product') }}" method="POST" enctype="multipart/form-
data">
@csrf
  <div class="mb-3">
    <label for="title">Nama Produk</label>
    <input type="text" id="title" name="title" required>
  </div>
  <div class="mb-3">
    <label for="description">Deskripsi</label>
    <textarea id="description" name="description" required></textarea>
  </div>
  <div class="mb-3">
    <label for="price">Harga</label>
    <input type="number" id="price" name="price" required min="0" step="0.01">
  </div>
  <div class="mb-3">
    <label for="qty">Kuantitas</label>
    <input type="number" id="qty" name="qty" required min="0">
  </div>
  <div class="mb-3">
    <label for="category">Kategori</label>
    <select id="category" name="category_id" required>
      <option value="">Pilih Kategori</option>
      @foreach($categories as $category)
        <option    value="{{    $category->id    }}">{{    $category->category_name
}}</option>
      @endforeach
    </select>
  </div>
  <div class="mb-3">
    <label for="image">Foto Produk</label>
    <input type="file" id="image" name="image" required accept="image/*">
  </div>
```

```
  <div>
    <input type="submit" class="btn btn-success" value="Tambah Produk">
  </div>
</form>
```

Gambar 4.32 Implementasi Fitur Tambah Produk

Form ini memuat input dasar produk: nama (title), deskripsi, harga (price), kuantitas (qty), dan gambar produk (image). Atribut enctype="multipart/form-data" diperlukan untuk mendukung pengunggahan file gambar. Semua input divalidasi di sisi frontend agar admin tidak mengirim data kosong.

Gambar 4.33 merupakan *function* untuk tambah produk dan mengupload produk ke dalam *database*.

```
public function add_product()
{
    $categories = Kategori::all();
    return view('admin.add_product');
}
public function upload_product(Request $request)
{
    $data = new Product;
    $data->title = $request->title;
    $data->description = $request->description;
    $data->price = $request->price;
    $data->quantity = $request->qty;
    $data->category_id = $request->category_id;

    $image = $request->image;
    if($image)
    {
        $imagename = time().'.'.$image->getClientOriginalExtension();
        $request->image->move('products',$imagename);
        $data->image = $imagename;
    }
    $data->save();
    toastr()->timeOut(10000)->closeButton()->success('Product Added
Successfully');
    return redirect()->back();
}
```

Gambar 4.33 *Function* Fitur Tambah Produk di *AdminController*

Fungsi upload_product() membuat objek baru dari model *Product*, lalu mengisi atribut produk dari data request. Gambar produk diproses dengan menyimpan nama file berbasis timestamp (time()) agar tidak duplikat, dan dipindahkan ke folder public/products/. Setelah

data disimpan dengan save(), admin akan kembali ke halaman sebelumnya dengan feedback visual melalui toastr().

2. Fitur Edit Produk

Fitur ini memungkinkan admin untuk memperbarui informasi produk yang sudah ada. Admin mengakses halaman edit produk melalui tombol "Edit", lalu mengubah data seperti nama, deskripsi, stok, atau gambar. Implementasi fitur edit produk dapat dilihat pada gambar 4.34.

```
<a class="btn btn-success" href="{{url('update_product', $products->id)}}">Edit</a>
```

Gambar 4.34 Tombol Edit Produk

Gambar 4.35 merupakan *function* untuk menyimpan produk yang telah di edit.

```
public function edit_product(Request $request, $id)
{
    $data = Product::find($id);
    $data->title = $request->title;
    $data->description = $request->description;
    $data->quantity = $request->quantity;
    $image = $request->image;
    if($image)
    {
       $imagename = time().'.'.$image->getClientOriginalExtension();
       $request->image->move('products', $imagename);
       $data->image = $imagename;
     }
     $data->save();
     toastr()->timeOut(5000)->closeButton()->success('Category Updated Successfully');
     return redirect('/view_product');
}
```

Gambar 4.35 *Function* Edit Produk

Fungsi ini mencari produk berdasarkan ID, lalu memperbarui isian yang telah diubah oleh admin. Jika admin juga mengganti gambar produk, sistem akan menyimpan gambar baru dan menimpanya pada entri lama. Tujuan utama proses ini adalah memastikan katalog tetap relevan dan informasi produk selalu diperbarui.

3. Fitur Hapus Produk

Admin dapat menghapus produk yang sudah tidak dijual atau tidak relevan. Proses ini dilakukan melalui tombol "Hapus" pada daftar produk. Implementasi fitur hapus produk dapat dilihat pada gambar 4.36.

```
//tombol hapus produk
<a          class="btn          btn-danger"          onclick="confirmation(event)"
href="{{url('delete_product', $products->id)}}">Hapus</a>
```

Gambar 4.36 Tombol Hapus Produk

Tindakan ini menjalankan *function* delete_product pada Gambar 4.37.

```
public function delete_product($id)
{
    $data = Product::find($id);
    $image_path = public_path('products/'.$data->image);
    if(file_exists($image_path))
    {
        unlink($image_path);
   }
    $data->delete();
    toastr()->timeOut(5000)->closeButton()->success('Category              Deleted
Successfully');
    return redirect()->back();
}
```

Gambar 4.37 *Function* Hapus Produk

Fungsi delete_product() mencari entri produk berdasarkan ID, lalu menghapus gambar dari direktori penyimpanan jika file-nya masih ada. Setelah itu, entri produk dihapus dari *database*. Penghapusan ini bertujuan menjaga integritas data dan mencegah tampilan produk yang sudah tidak valid di sisi pelanggan.

4. Fitur Filter dan *Sorting List* Produk

Halaman daftar produk juga dilengkapi dengan fitur filter berdasarkan kategori produk. Fitur ini memungkinkan admin untuk menampilkan produk secara terkelompok sesuai kategori yang dipilih melalui dropdown menu. Ketika admin memilih salah satu kategori, sistem akan menampilkan hanya produk-produk yang termasuk dalam kategori tersebut. Hal ini memudahkan proses manajemen dan pencarian data produk secara lebih spesifik, terutama jika jumlah produk dalam sistem sudah banyak. Tampilan fitur filter produk dapat dilihat pada Gambar 4.38.

```
<select  name="filter_category"  onchange="this.form.submit()"  class="btn  btn-
secondary">
    <option value="">-- Semua Kategori --</option>
    @foreach($all_categories as $cat)
    <option  value="{{  $cat->id  }}"  {{  request('filter_category')  ==  $cat->id  ?
'selected' : '' }}>
                {{ $cat->category_name }}
    </option>
```

```
    @endforeach
</select>
```

Gambar 4.38 Fitur Filter Produk

Selain fitur filter produk, halaman ini juga dilengkapi dengan fitur *sorting list* yang memudahkan admin untuk mengurutkan produk berdasarkan kriteria tertentu. Fitur ini diimplementasikan dalam bentuk elemen <select> di dalam formulir GET, yang secara otomatis mengirimkan permintaan ke server setiap kali pilihan sorting diubah. Tampilan fitur *sorting list* dapat dilihat pada Gambar 4.39.

```
<select name="sort" onchange="this.form.submit()" class="btn btn-secondary">
    <option value="latest" {{ request('sort') == 'latest' ? 'selected' : ''
}}>Terbaru</option>
    <option value="name_asc" {{ request('sort') == 'name_asc' ? 'selected' : ''
}}>Nama A-Z</option>
   <option value="name_desc" {{ request('sort') == 'name_desc' ? 'selected' : ''
}}>Nama Z-A</option>
   <option value="price_low_high" {{ request('sort') == 'price_low_high' ?
'selected' : '' }}>Harga Termurah</option>
   <option value="price_high_low" {{ request('sort') == 'price_high_low' ?
'selected' : '' }}>Harga Termahal</option>
  <option value="category_asc" {{ request('sort') == 'category_asc' ? 'selected' :
'' }}>Kategori A-Z</option>
   <option value="category_desc" {{ request('sort') == 'category_desc' ? 'selected'
: '' }}>Kategori Z-A</option>
</select>
```

Gambar 4.39 Fitur *Sorting List* Produk

Fitur sorting dan filter pada halaman daftar produk diproses melalui method view_product(Request $request) di controller. Parameter sort digunakan untuk menentukan urutan data produk berdasarkan opsi yang dipilih oleh admin, seperti mengurutkan produk berdasarkan nama (A-Z/Z-A), harga (termurah/termahal), atau berdasarkan nama kategori. Sementara itu, parameter filter_category digunakan untuk menyaring tampilan produk agar hanya menampilkan produk dari kategori tertentu yang dipilih melalui dropdown. Kombinasi kedua fitur ini memungkinkan admin untuk menyesuaikan tampilan daftar produk secara lebih fleksibel dan terstruktur, baik dalam konteks pencarian data maupun pengelompokan produk. Function view_product yang mengimplementasikan logika sorting dan filter dapat dilihat pada Gambar 4.40.

```
public function view_product(Request $request)
    {
        $sort = $request->get('sort', 'latest');
```

```
        $filterCategory = $request->get('filter_category');
        $all_categories = Kategori::all();
        $query = Product::with('kategori');

        if ($filterCategory) {
        $query->where('category_id', $filterCategory);
    }

        switch ($sort) {
            case 'name_asc':
                $query->orderBy('title', 'asc');
                break;
            case 'name_desc':
                $query->orderBy('title', 'desc');
                break;
            case 'price_low_high':
                $query->orderBy('price', 'asc');
                break;
            case 'price_high_low':
                $query->orderBy('price', 'desc');
                break;
            case 'category_asc':
                        $query->join('kategoris',  'products.category_id',  '=',
'kategoris.id')
                    ->orderBy('kategoris.category_name', 'asc')
                    ->select('products.*');
                break;
            case 'category_desc':
                        $query->join('kategoris',  'products.category_id',  '=',
'kategoris.id')
                    ->orderBy('kategoris.category_name', 'desc')
                    ->select('products.*');
                break;
            default:
                $query->orderBy('created_at', 'desc'); // default: terbaru
                break;
        }

        $product = $query->paginate(5)->appends([
            'sort' => $sort,
            'filter_category' => $filterCategory
        ]);
        return     view('admin.view_product',     compact('product',     'sort',
'all_categories'));
```

```
    }
```

Gambar 4.40 *Function* view_product

Data produk yang telah difilter dan diurutkan kemudian ditampilkan pada halaman admin.view_product menggunakan fitur pagination, sehingga memudahkan navigasi antar halaman produk. Parameter sorting dan filter tetap dipertahankan menggunakan method appends(), sehingga status pilihan pengguna tidak hilang saat berpindah halaman. Fitur sorting dan filter ini memberikan fleksibilitas lebih bagi admin dalam menyesuaikan tampilan data produk, seperti meninjau produk berdasarkan harga, nama, maupun kategori tertentu. Dengan adanya fitur ini, proses manajemen produk menjadi lebih efisien, terstruktur, dan mudah dikontrol sesuai kebutuhan administrasi toko.

4.4.4 Fitur Manajemen Pesanan

Fitur ini memungkinkan admin untuk melihat, memproses, dan mengubah status pesanan yang dilakukan oleh pelanggan. Semua fungsi pengelolaan data pesanan dilakukan melalui *AdminController* dan ditampilkan dalam halaman admin. Pada fitur manajemen pesanan terdapat beberapa fitur seperti tampilan tabel pesanan yang masuk, tombol untuk memverifikasi pembayaran, dan tombol untuk mengubah status pesanan.

1. Tampilan Tabel Pesanan

Pada tabel pesanan masuk diperlihatkan beberapa informasi dari pesanan yang masuk, seperti nama, alamat, nomor telepon pemesan, nama produk, harga, gambar produk, gambar bukti pembayaran, dan status. Implementasi tampilan tabel pesanan dapat dilihat pada gambar 4.41.

```
<table>
    <tr>
       <th>Nama pemesan</th>
        <th>Alamat</th>
        <th>Nomor Telepon</th>
        <th>Nama Produk</th>
        <th>Harga</th>
        <th>Image</th>
        <th>Bukti Pembayaran</th>
        <th>Status</th>
    </tr>
    @foreach($data as $data)
    <tr>
         <td>{{$data->name}}</td>
         <td>{{$data->rec_address}}</td>
```

```
          <td>{{$data->phone}}</td>
          <td>{{$data->product->title}}</td>
          <td>Rp.{{$data->product->price}}</td>
          <td>
              <img width="150" src="/products/{{$data->product->image}}" >
          </td>
          <td>
              <img width="150" src="/payment/{{$data->payment_proof}}" >
              <form action="{{ url('terverifikasi', $data->id) }}">
                    @csrf
                   <button          type="submit"          class="btn          btn-
success">Verifikasi</button>
              </form>
              <form action="{{ url('verifikasi_gagal', $data->id) }}">
                     @csrf
                      <button   type="submit"   class="btn   btn-danger">Verifikasi
Gagal</button>
              </form>
         </td>
         <td>
            @if($data->status == 'Sedang Diproses')
            <span style="color: red">{{$data->status}}</span>
            @elseif($data->status == 'Sedang Dikirim')
            <span style="color: yellow">{{$data->status}}</span>
            @else
            <span style="color: green">{{$data->status}}</span>
           @endif
            <div>
                <a   class="btn   btn-primary"   href="{{url('on_the_way',   $data-
>id)}}">Sedang Dikirim</a>
                <a   class="btn   btn-success"   href="{{url('delivered',   $data-
>id)}}">Selesai</a>
            </div>
        </td>
    </tr>
    @endforeach
</table>
```

Gambar 4.41 Implementasi Tampilan Tabel Pesanan

Tabel ini menampilkan relasi antar entitas seperti orders dan products melalui metode Eloquent relationship Laravel ($data->product->title). Dengan demikian, admin dapat melihat data lengkap pesanan beserta bukti pembayaran secara visual.

Gambar 4.42 merupakan *function* untuk menampilkan data pesanan ke dalam tabel.

```
public function view_order()
    {
        $data = Order::all();
        return view('admin.order', compact('data'));
    }
```

Gambar 4.42 *Function* Menampilkan Data Pada Tabel

Fungsi view_order() mengambil semua data pesanan dari *database* menggunakan Order::all() dan mengirimkannya ke halaman admin untuk ditampilkan dalam tabel.

2. Tombol Verifikasi Pembayaran

Pada tabel terdapat dua tombol untuk memverifikasi pembayaran, yaitu “Verifikasi” dan “Verifikasi Gagal”. Implementasi tombol verifikasi pembayaran dapat dilihat pada gambar 4.43.

```
<td>
    <img width="150" src="/payment/{{$data->payment_proof}}" >
              <form action="{{ url('terverifikasi', $data->id) }}">
                       @csrf
        <button type="submit" class="btn btn-success">Verifikasi</button>
    </form>
    <form action="{{ url('verifikasi_gagal', $data->id) }}">
        @csrf
         <button type="submit" class="btn btn-danger">Verifikasi Gagal</button>
    </form>
</td>
```

Gambar 4.43 Implementasi Tombol Verifikasi dan Verifikasi Gagal

Setiap tombol memanggil *route* berbeda yang memicu fungsi pada Gambar 4.44.

```
public function terverifikasi($id)
    {
        $data = Order::find($id);
        $data->payment_proof_status = 'Terverifikasi';
        $data->save();
        return redirect('/view_order');
    }
public function verifikasi_gagal($id)
    {
        $data = Order::find($id);
        $data->payment_proof_status = 'Verifikasi Gagal';
        $data->save();
        return redirect('/view_order');
```

```
    }
```

Gambar 4.44 *Function* Mengubah Status Pembayaran

Kedua fungsi ini bertugas mengubah status verifikasi pembayaran pada entri *orders*. Hal ini penting untuk memberi informasi ke pelanggan apakah bukti pembayaran mereka diterima atau ditolak. Perubahan ini juga akan tercermin pada halaman riwayat pesanan pelanggan.

3. Tombol Mengubah Status Pesanan

Pada tabel secara default status pesanan yaitu "Sedang Diproses*"* yang menandakan bahwa pesanan masih dalam tahap verifikasi pembayaran. Setelah pembayaran terverifikasi dan pesanan siap untuk dikirim, admin mengubah status pesanan dengan menekan tombol "Sedang Dikirim". Setelah pesanan sampai, admin menegubah status dengan menekan tombol "Selesai". Implementasi fitur mengubah status pesanan dapat dilihat pada gambar 4.45.

```
<td>
    @if($data->status == 'Sedang Diproses')
    <span style="color: red">{{$data->status}}</span>
    @elseif($data->status == 'Sedang Dikirim')
    <span style="color: yellow">{{$data->status}}</span>
    @else
    <span style="color: green">{{$data->status}}</span>

    @endif
    <div>
    <a class="btn btn-primary" href="{{url('on_the_way', $data->id)}}">Sedang
Dikirim</a>
    <a class="btn btn-success" href="{{url('delivered', $data->id)}}">Selesai</a>
    </div>
</td>
```

Gambar 4.45 Implementasi Fitur Mengubah Status Pesanan

Gambar 4.46 merupakan *function* untuk menyimpan dan mengubah status pesanan di *database*.

```
public function on_the_way($id)
{
        $data = Order::find($id);
        $data->status = 'Sedang Dikirim';
        $data->save();
        return redirect('/view_order');
}
public function delivered($id)
{
```

```
        $data = Order::find($id);
        $data->status = 'Selesai';
        $data->save();
        return redirect('/view_order');
}
```

Gambar 4.46 *Function* Untuk Menyimpan dan Mengubah Status Pesanan

Setiap fungsi mengubah nilai kolom status dalam tabel orders, yang merepresentasikan progres dari pemrosesan pesanan. Status "Sedang Dikirim" menunjukkan bahwa produk telah dikirim, sedangkan "Selesai" menandakan bahwa pesanan telah sampai dan proses transaksi telah tuntas. Informasi ini juga secara otomatis diperbarui di halaman pelanggan.

Dengan mekanisme ini, admin dapat secara efisien memantau alur transaksi dari awal pemesanan hingga produk dikirim ke pelanggan.

4.4.5 Fitur Katalog Produk

Halaman ini digunakan oleh pelanggan untuk menelusuri seluruh produk yang tersedia di halaman utama. Masing-masing produk ditampilkan dalam kartu (*card*) yang memuat gambar, nama, harga, dan tombol "Tambahkan Keranjang". Fungsi untuk menampilkan katalog produk dilakukan melalui *HomeController.* Implementasi fitur katalog produk dapat dilihat pada gambar 4.47.

```
<div class="row">
    @foreach($product as $products)
    <div class="col-sm-6 col-md-4 col-lg-3">
        <div class="box">
            <a href="{{url('product_details',$products->id)}}">
            <div class="img-box">
                <img src="products/{{$products->image}}" alt="">
            </div>
            <div class="detail-box">
                <h6>{{$products->title}}</h6>
            </div>
            <div class="detail-box">
                <h6>
                   <span>Rp.{{$products->price}}</span>
                </h6>
            </div>
            </a>
            <div class="cart-button">
                <a   class="btn   btn-custom"   href="{{url('add_cart',   $products-
>id)}}">
```

```
                <i class="fa fa-shopping-cart"></i> Tambahkan Keranjang
                </a>
            </div>
        </div>
    </div>
   @endforeach
</div>
```

Gambar 4.47 Implementasi Fitur Katalog Produk

Potongan kode di atas menunjukkan bagaimana setiap produk dirender ke dalam HTML. Informasi yang ditampilkan mencakup gambar ($products->image), judul ($products->title), dan harga ($products->price). Tombol "Tambahkan Keranjang" berfungsi mengarahkan pengguna ke proses penambahan item ke keranjang, yang nantinya akan dijelaskan di subfitur berikutnya.

Function untuk menampilkan katalog produk dilakukan melalui *HomeController* dapat dilihat pada Gambar 4.48.

```
public function shop()
    {
        $product = Product::all();
        return view('home.shop', compact('product'));
    }
```

Gambar 4.48 *Function* Menampilkan Katalog Produk

Fungsi ini menggunakan metode Product::all() untuk mengambil seluruh data dari tabel products. Data tersebut kemudian dikirim ke *view home.shop* menggunakan fungsi compact(). *View* ini kemudian mengiterasi setiap entri produk dan merender-nya ke tampilan menggunakan @foreach.

Penggunaan fungsi ini bersifat statis, yaitu menampilkan seluruh produk tanpa adanya filter atau pagination. Jika sistem ingin dikembangkan lebih lanjut, fungsi ini dapat diperluas dengan logika pencarian, kategori, atau pemfilteran berdasarkan harga.

Fitur pada katalog produk terintegrasi dengan fitur lain, yaitu :

- Fitur Detail Produk, saat pelanggan mengklik gambar produk,
- Fitur Keranjang Belanja, melalui tombol “Tambahkan Keranjang”,
- Manajemen Produk Admin, karena data yang ditampilkan bersumber dari entri yang diinput admin.

Dengan demikian, halaman katalog menjadi simpul utama yang menghubungkan berbagai fitur lainnya dalam sistem *e-commerce* ini.

4.4.6 Fitur Detail Produk

Fitur halaman detail produk merupakan komponen utama bagi pelanggan untuk menelusuri dan memilih produk yang tersedia. Halaman ini menampilkan semua item yang telah diinput oleh admin ke dalam sistem, lengkap dengan gambar, nama, dan harga. Implementasi fitur tampilan detail produk dapat dilihat pada Gambar 4.49.

```
//Tampilan detail produk
<div class="product-container">
        <div class="product-image">
           <img src="/products/{{$data->image}}" alt="">
        </div>
        <div class="product-details">
           <h6 class="product-title">{{ $data->title}}</h6>
           <p class="price">Rp {{$data->price}}</p>
           <p class="description">{{ $data->description }}</p>
           <p class="contact">Untuk info detail, kami melayani pemesanan dan custom
product sesuai selera dan kebutuhan anda. Please DM atau kontak konsultasi GRATIS.
Tlp/WA: 0852-1582-0912</p>
            <div class="detail-box">
               <form action="{{ url('add_cart', $data->id) }}" method="POST">
                   @csrf
                   <button    type="submit"    class="btn    btn-primary">Tambahkan
Keranjang</button>
               </form>
           </div>
      </div>
</div>
```

Gambar 4.49 Implementasi Fitur Tampilan Detail Produk

Komponen yang ditampilkan mencakup gambar berukuran besar, nama produk, harga, dan deskripsi yang diambil dari data $data, yang dikirimkan dari *controller*. Teks tambahan seperti nomor WhatsApp bertujuan untuk memberikan opsi komunikasi langsung bagi pelanggan yang membutuhkan konsultasi lebih lanjut atau ingin melakukan pemesanan kustom.

Tombol “Tambahkan Keranjang” memungkinkan pelanggan langsung memasukkan produk ke dalam keranjang tanpa harus kembali ke katalog. Hal ini mendukung alur belanja yang lebih efisien dan mengurangi *friction* dalam pengalaman pengguna.

Function untuk menampilkan halaman detail produk dilakukan melalui *HomeController* dapat dilihat pada Gambar 4.50

```
public function product_details($id)
    {
        $data = Product::find($id);
```

```
        if(Auth::id())
        {
            $user = Auth::user();
            $userid = $user->id;
            $count = Cart::where('user_id', $userid)->count();
        }
        else
        {
            $count = '';
        }
        return view('home.product_details', compact('data','count'));
    }
```

Gambar 4.50 *Function* Menampilkan Halaman Detail Produk

Fungsi product_details() menerima parameter $id dari produk yang dipilih. Produk kemudian diambil dari *database* menggunakan Product::find($id). Jika pengguna telah login, sistem juga menghitung jumlah item dalam keranjang pengguna saat ini dan mengirimkan nilai tersebut ke *view* untuk ditampilkan (misalnya dalam ikon keranjang di header). Hal ini memberikan informasi real-time tentang jumlah barang yang sudah dikumpulkan pengguna.

Fitur pada halaman detail produk terintegrasi dengan fitur lain, yaitu :

- Fitur Katalog Produk, sebagai pintu masuk,
- Fitur Keranjang Belanja, untuk menyimpan produk yang dipilih,
- *Database* Produk, untuk mendapatkan informasi produk berdasarkan ID.

Dengan menampilkan informasi produk secara rinci dan menyertakan akses cepat ke proses pembelian, fitur ini mendukung keputusan belanja pelanggan secara lebih informatif dan responsif.

4.4.7 Fitur Keranjang Belanja

Setelah melihat detail produk, pelanggan dapat menambahkan produk yang diinginkan ke dalam keranjang belanja. Fitur ini membantu menyimpan daftar produk sementara sebelum pelanggan melanjutkan ke proses pemesanan. Terdapat juga fitur tombol untuk menghapus produk dari keranjang belanja. Implementasi fitur tombol tambah keranjang belanja dapat dilihat pada gambar 4.51.

```
<form action="{{ url('add_cart', $data->id) }}" method="POST">
     @csrf
     <button type="submit" class="btn btn-primary">Tambahkan Keranjang</button>
</form>
```

Gambar 4.51 Implementasi Fitur Tombol Tambah Keranjang

Tombol ini menjalankan metode *POST* ke *route* add_cart/{id}, yang menangani penyimpanan data produk ke dalam tabel keranjang.

Semua produk yang telah dipilih ditampilkan dalam halaman khusus yang menyerupai daftar belanja, sehingga memudahkan pelanggan dalam meninjau ulang dan melakukan pengeditan sebelum memesan. Implementasi tampilan halaman keranjang dapat dilihat pada gambar 4.52.

```
<table>
       <tr>
           <th>Product Name</th>
           <th>Price</th>
           <th>Image</th>
           <th>Actions</th>
       </tr>
       @php $total = 0; @endphp
       @foreach ($cart as $cart)
       <tr>
           <td>{{ $cart->product->title }}</td>
           <td>Rp.{{ $cart->product->price }}</td>
           <td>
                 <img src="/products/{{ $cart->product->image }}" alt="{{ $cart-
>product->title }}" width="120">
           </td>
           <td>
               <form action="{{ url('remove_cart', $cart->id) }}" method="POST">
                   @csrf
                   @method('DELETE')
                   <button type="submit" class="btn btn-danger">Hapus</button>
               </form>
           </td>
       </tr>
       <?php
           $total = $total + ($cart->product->price * 1000);
       ?>
       @endforeach
</table>
```

```
<h3 class="total-price">Total belanja: Rp.{{ number_format($total, 0, ',', '.')
}}</h3>
```

Gambar 4.52 Implementasi Fitur Halaman Keranjang

Tabel ini memanfaatkan relasi Cart -> Product untuk mengambil informasi produk yang telah ditambahkan. Pengguna dapat melihat gambar produk, harga satuan, dan menghapus item dengan menekan tombol "Hapus". Di bagian bawah tabel, sistem secara otomatis menghitung dan menampilkan total harga pembelian berdasarkan seluruh item yang ada di keranjang, sehingga pengguna dapat mengetahui jumlah biaya yang harus dibayarkan sebelum melanjutkan ke proses checkout. Perhitungan total ini dilakukan dengan menjumlahkan hasil perkalian antara harga satuan produk dan kuantitasnya, sebagaimana ditunjukkan pada potongan kode <?php $total = $total + ($cart->product->price * 1000); ?> yang mengakumulasi total harga setiap item ke dalam variabel $total.

Gambar 4.53 terdapat tiga *function* pada *AdminController* yang berfungsi menambahkan produk ke dalam keranjang, menampilkan seluruh isi keranjang dan fungsi untuk menghapus item tertentu dari keranjangnya.

```
Public function add_cart($id)
    {
        $product_id = $id;
        $user = Auth::user();
        $user_id = $user->id;
        $data = new Cart;
        $data->user_id = $user_id;
        $data->product_id = $product_id;
        $data->save();
         toastr()->timeOut(5000)->closeButton()->success('Berhasil Menambahkan Ke
Keranjang');
        return redirect()->back();
    }

public function mycart()
    {
        if(Auth::id())
        {
            $user = Auth::user();
            $userid = $user->id;
            $count = Cart::where('user_id', $userid)->count();
            $cart = Cart::where('user_id', $userid)->get();
        }
        return view('home.mycart', compact('count','cart'));
    }
```

```
public function remove_cart($id)
    {
        $cartItem = Cart::find($id); // Temukan item keranjang berdasarkan ID
        if ($cartItem) {
            $cartItem->delete(); // Hapus item dari keranjang
                toastr()->timeOut(5000)->closeButton()->success('Produk  berhasil
dihapus dari keranjang');
        } else {
            toastr()->error('Produk tidak ditemukan');
        }
        return redirect()->back(); // Kembali ke halaman sebelumnya
    }
```

Gambar 4.53 *Function* Halaman Keranjang

Fungsi add_cart($id) bekerja dengan mencatat ID pengguna dan ID produk ke dalam tabel carts. Pendekatan ini memungkinkan setiap pengguna memiliki keranjang yang bersifat personal. Data disimpan secara langsung menggunakan Eloquent ORM Laravel, sehingga prosesnya efisien dan minim error.

Fungsi mycart() mengambil seluruh isi keranjang berdasarkan ID pengguna yang sedang login. Jumlah item dihitung untuk keperluan tampilan (misalnya, ikon jumlah item di header), dan semua data dikirim ke view home.mycart.

Fungsi remove_cart() melakukan pencarian item keranjang berdasarkan ID, kemudian menghapus entri tersebut dari *database*. Pendekatan ini menjaga agar isi keranjang tetap akurat dan sesuai dengan keinginan pengguna.

4.4.8 Fitur Checkout dan Membuat Pesanan

Fitur checkout memungkinkan pelanggan menyelesaikan transaksi dengan memesan produk yang telah dimasukkan ke dalam keranjang dengan mengisi formulir yang berisi nama,alamat, nomor telepon pemesan, dan tempat untuk mengupload bukti pembayaran. Terdapat juga instruksi tata cara pembayaran. Data pesanan akan tersimpan ke dalam sistem dan dapat diproses lebih lanjut oleh admin. Implementasi fitur *checkout* dan membuat pesanan dapat dilihat pada gambar 4.54.

```
<form action="{{ url('confirm_order') }}" method="POST" enctype="multipart/form-
data">
   @csrf
   <div class="mb-3">
       <label for="name" class="form-label">Nama Penerima</label>
```

```
        <input type="text" name="name" class="form-control" value="{{ Auth::user()->name }}" required>
    </div>
    <div class="mb-3">
        <label for="address" class="form-label">Alamat Penerima</label>
        <textarea name="address" class="form-control" required></textarea>
    </div>
    <div class="mb-3">
        <label for="phone" class="form-label">Nomor Telefon</label>
       <input type="text" name="phone" class="form-control" value="{{ Auth::user()->phone }}" required>
    </div>
    <div class="mb-3">
        <label for="image" class="form-label">Bukti Pembayaran</label>
        <input type="file" id="image" name="image" accept="image/*" required>
    </div>
    <div class="payment-instructions">
        <h3>Tata Cara Pembayaran</h3>
        <ul>
              <li>Transfer pembayaran ke rekening berikut: <strong>1234-5678-9012 (Bank Rahdatu)</strong>.</li>
              <li>Gunakan nama lengkap Anda sebagai referensi pembayaran.</li>
              <li>Unggah bukti pembayaran pada kolom di atas untuk verifikasi.</li>
              <li>Pesanan Anda akan diverifikasi dalam 1-2 hari kerja.</li>
        </ul>
    </div>
    <button type="submit" class="btn btn-primary">Confirm Order</button>
</form>
```

Gambar 4.54 Implementasi Fitur Checkout dan Membuat Pesanan

Gambar 4.55 merupakan *function* yang digunakan untuk memproses checkout pesanan pelanggan. Data seperti nama, alamat, nomor telepon, dan bukti pembayaran dikirim dari form dan disimpan ke dalam *database* untuk setiap item di keranjang pengguna. Setelah pesanan tersimpan, seluruh item di keranjang dihapus, dan pelanggan diarahkan ke halaman riwayat pesanan.

```
public function confirm_order(Request $request)
    {
        $name = $request->name;
        $address = $request->address;
        $phone = $request->phone;
        $userid = Auth::user()->id;
        $cart = Cart::where('user_id',$userid)->get();
        foreach($cart as $carts)
```

```
        {
            $order = new Order;
            $order->name = $name;
            $order->rec_address = $address;
            $order->phone = $phone;
            $order->user_id = $userid;
            $order->product_id = $carts->product_id;
            $image = $request->image;
            if($image)
            {
                $imagename = time().'.'.$image->getClientOriginalExtension();
                $request->image->move('payment',$imagename);
                $order->payment_proof = $imagename;
            }
            $order->save();
        }
        $cart_remove = Cart::where('user_id',$userid)->get();
        foreach($cart_remove as $remove)
        {
            $data = Cart::find($remove->id);
            $data->delete();
        }
       toastr()->timeOut(5000)->closeButton()->success('Produk berhasil dipesan');
        return redirect('myorders');
    }
```

Gambar 4.55 *Function* Proses *Checkout* Pesanan

Fungsi ini memproses data checkout dalam beberapa tahap:

1. Pengambilan Data: Sistem mengambil data pengguna dan semua item dalam keranjang berdasarkan user_id.
2. Pembuatan Pesanan: Untuk setiap item di keranjang, dibuat satu entri baru pada tabel *orders* yang berisi informasi pemesan dan produk.
3. Pengunggahan Bukti Pembayaran: File gambar disimpan di direktori payment/ dengan nama unik berbasis *timestamp*.
4. Penghapusan Keranjang: Setelah semua pesanan berhasil disimpan, isi keranjang dibersihkan agar tidak terjadi duplikasi pemesanan.
5. Pengalihan ke Halaman Riwayat Pesanan: Setelah selesai, pelanggan diarahkan ke halaman myorders untuk melihat status pesanan.

4.4.9 Fitur Melihat Status Pesanan

Pelanggan dapat memantau status dari pesanan yang telah mereka buat, mulai dari tahap pemrosesan hingga pengiriman. Tampilan informasi status ini berupa tabel yang ditampilkan secara *real-time* sesuai dengan perubahan yang dilakukan oleh admin. Implementasi fitur melihat status pesanan dapat dilihat pada gambar 4.56.

```
<table>
       <tr>
           <th>NO</th>
           <th>Tanggal Pemesanan</th>
           <th>Nama Produk</th>
           <th>Harga</th>
           <th>Verifikasi Pembayaran</th>
           <th>Status Pengiriman</th>
           <th>Gambar Produk</th>
      </tr>
     @foreach ($orders as $index => $order)
       <tr>
           <td>{{ $index + 1 }}</td>
           <td>{{ $order->tanggal_pemesanan_indo }}</td>
           <td>{{ $order->product->title }}</td>
           <td>Rp.{{ $order->product->price }}</td>
           <td>
               @if($order->payment_proof_status == 'Terverifikasi')
               <span class="badge badge-success">Terverifikasi</span>
               @elseif($order->payment_proof_status == 'Verifikasi Gagal')
               <span class="badge badge-danger">Verifikasi Gagal</span>
               @else
               <span class="badge badge-warning">Menunggu Verifikasi</span>
               @endif
           </td>
           <td>{{ $order->status }}</td>
           <td>
               <img src="/products/{{ $order->product->image }}" alt="{{ $order-
>product->title }}" width="100">
           </td>
       </tr>
       @endforeach>
</table>
```

Gambar 4.56 Implementasi Fitur Melihat Status Pesanan

Tabel ini menampilkan informasi pesanan berdasarkan relasi antara entitas *orders* dan *products*. Kolom status verifikasi ditampilkan dalam bentuk label warna (badge), sehingga pengguna dapat dengan cepat mengenali kondisi pesanan. Gambar produk juga ditampilkan

untuk memperjelas produk yang dimaksud dalam pesanan. Tampilan ini sangat penting dari sisi pengalaman pengguna karena memberikan transparansi terhadap proses transaksi dan menciptakan rasa percaya terhadap sistem.

Gambar 4.57 merupakan *function* untuk menampilkan data pesanan pelanggan yang telah melakukakn *checkout*.

```
public function myorders()
    {
        $user = Auth::user()->id;
        $count = Cart::where('user_id', $user)->get()->count();
        $order = Order::where('user_id', $user)->get();
        return view('home.order', compact('count', 'order'));
    }
```

Gambar 4.57 *Function* Menampilkan Data Pesanan

Fungsi ini mengambil data pesanan berdasarkan user_id pengguna yang sedang login. Jumlah item dalam keranjang juga dihitung untuk ditampilkan di bagian lain Antar muka. Dengan memanfaatkan *Eloquent*, sistem dengan mudah melakukan filter data berdasarkan pengguna tanpa *query* manual yang kompleks.

Fitur ini memberikan manfaat langsung bagi pelanggan berupa:

- Monitoring transaksi secara mandiri tanpa perlu menghubungi admin,
- Mendeteksi kendala (misalnya: verifikasi gagal atau pesanan belum dikirim),
- Meningkatkan kepercayaan pengguna karena sistem terlihat transparan dan profesional.

Secara fungsional, fitur ini memperkuat alur *e-commerce* dan menjembatani komunikasi tidak langsung antara sistem dan pengguna.

4.4.10 Struktur Routing Laravel

Laravel menggunakan konsep routing untuk mengarahkan permintaan dari pengguna ke *controller* yang sesuai. Subbab ini akan menjelaskan bagaimana *route* disusun dan bagaimana masing-masing fitur terhubung dalam kerangka kerja MVC Laravel. Struktur *routes* dapat dilihat pada gambar 4.58 dan 4.59.

1. *Routes* Untuk Admin

```
Route::get('add_product',[AdminController::class,'add_product'])-
>middleware(['auth','admin']);
Route::post('upload_product',[AdminController::class,'upload_product'])-
>middleware(['auth','admin']);
Route::get('view_product',[AdminController::class,'view_product'])-
>middleware(['auth','admin']);
```

```
Route::get(‘delete_product/{id}’,[AdminController::class,’delete_product’])-
>middleware([‘auth’,’admin’]);
Route::get(‘update_product/{id}’,[AdminController::class,’update_product’])-
>middleware([‘auth’,’admin’]);
Route::post(‘edit_product/{id}’,[AdminController::class,’edit_product’])-
>middleware([‘auth’,’admin’]);
Route::get(‘product_search’,[AdminController::class,’product_search’])-
>middleware([‘auth’,’admin’]);
Route::get(‘view_order’,[AdminController::class,’view_order’])-
>middleware([‘auth’,’admin’]);
Route::get(‘on_the_way/{id}’,[AdminController::class,’on_the_way’])-
>middleware([‘auth’,’admin’]);
Route::get(‘delivered/{id}’,[AdminController::class,’delivered’])-
>middleware([‘auth’,’admin’]);
Route::get(‘terverifikasi/{id}’,[AdminController::class,’terverifikasi’])-
>middleware([‘auth’,’admin’]);
Route::get(‘verifikasi_gagal/{id}’,[AdminController::class,’verifikasi_gagal’])-
>middleware([‘auth’,’admin’]);
Route::get('view_category',[AdminController::class,'view_category'])-
>middleware(['auth','admin']);
Route::post('add_category',[AdminController::class,'add_category'])-
>middleware(['auth','admin']);
Route::get('delete_category/{id}',[AdminController::class,'delete_category'])-
>middleware(['auth','admin']);
Route::get('edit_category/{id}',[AdminController::class,'edit_category'])-
>middleware(['auth','admin']);
Route::post('update_category_post/{id}',[AdminController::class,'update_category_
post'])->middleware(['auth','admin']);
```

Gambar 4.58 *Routes* Untuk Admin

Setiap *route* dibungkus dengan *middleware auth* dan admin. *Middleware auth* memastikan hanya pengguna yang sudah login yang bisa mengakses halaman tersebut. *Middleware* admin membatasi akses hanya untuk admin, yang biasanya ditentukan berdasarkan atribut *usertype* pada data pengguna.

Struktur ini menjaga agar fitur penting seperti manajemen produk, kategori, dan pesanan tidak bisa diakses oleh pelanggan biasa. Ini merupakan bagian dari *access control* yang fundamental dalam aplikasi multi-role.

2. *Routes* Untuk Pelanggan

```
Route::get('product_details/{id}',[HomeController::class,'product_details']);
Route::get('add_cart/{id}',[HomeController::class,'add_cart'])-
>middleware(['auth', 'verified']);
```

```
Route::post('add_cart/{id}', [HomeController::class, 'add_cart'])-
>middleware(['auth', 'verified']);
Route::get('mycart',[HomeController::class,'mycart'])->middleware(['auth',
'verified']);
Route::get('myorders', [HomeController::class, 'myorders'])->middleware(['auth',
'verified']);
Route::delete('remove_cart/{id}', [HomeController::class, 'remove_cart'])-
>middleware(['auth', 'verified']);
Route::post('confirm_order', [HomeController::class, 'confirm_order'])-
>middleware(['auth', 'verified']);
Route::get('shop',[HomeController::class,'shop']);
Route::get('why',[HomeController::class,'why']);
```

Gambar 4.59 *Routes* Untuk Pelanggan

Beberapa *route* seperti *shop* dan product_details dapat diakses tanpa login karena fungsinya hanya untuk melihat. Namun, proses transaksi seperti add_cart, mycart, dan confirm_order dilindungi dengan *middleware auth* dan *verified* untuk memastikan hanya pelanggan yang telah terautentikasi dan diverifikasi email-nya yang dapat melakukan pembelian.

Middleware verified merupakan fitur bawaan Laravel untuk mengaktifkan verifikasi email, dan bisa diaktifkan bila sistem membutuhkan keamanan tambahan terhadap akun pelanggan.

Peran Routing dalam arsitektur MVC Laravel sangat penting. Ia bertindak sebagai jembatan antara tampilan pengguna (*View*) dan logika pemrosesan (*Controller*). Dengan mendefinisikan *route* secara eksplisit, pengembang dapat mengatur alur kontrol akses, mengarahkan pengguna ke *view* yang sesuai, dan menjaga agar alur sistem tetap modular dan aman. Dalam sistem *e-commerce* ini, struktur routing telah diatur sedemikian rupa untuk membedakan peran admin dan pelanggan. Hal ini penting untuk menghindari penyalahgunaan fitur dan menjaga integritas data dalam sistem.

4.5 Pengujian Sistem/Testing

4.5.1 *Black Box Testing*

Pada tahap ini keseluruhan sistem akan diuji apakah sistem berjalan sesuai dengan apa yang diharapkan atau belum. Pengujian akan dilakukan dengan metode *Black Box* yang dapat dilihat pada Tabel 4.2.

Tabel 4.2 Hasil Pengujian *Black Box*

Halaman/Fitur	Test Case	Hasil yang Diharapkan	Hasil Pengujian
Landing page	Pengguna membuka situs website dan melihat landing page	Halaman landing page ditampilkan dengan benar termasuk navigasi menuju login atau register	Berhasil (valid)
Login dan register	Pengguna mendaftarkan pelangganname, email, nomor telepon, dan password pada form lalu menekan tombol register	Sistem berhasil meregisterasi akun pengguna	Berhasil (valid)
	Pengguna memasukkan pelangganname dan password pada form lalu menekan tombol login	Sistem memverifikasi akun pengguna yang telah terdaftar dan masuk ke halaman home	Berhasil (valid)
Mencari produk yang diinginkan (pelanggan)	Pengguna mencari produk yang diinginkan dengan melihat halaman home	Sistem berhasil menampilkan seluruh produk	Berhasil (valid)
Melihat detail produk (pelanggan)	Pengguna menekan produk untuk melihat detail produk	Sistem berhasil berpindah ke halaman detail produk	Berhasil (valid)
Memasukan produk ke keranjang (pelanggan)	Pengguna menekan tombol tambah keranjang	Sistem berhasil menambahkan produk ke keranjang	Berhasil (valid)
Melakukan transaksi (pelanggan)	Pengguna melakukan transaksi dengan mengisi formulir dan mengirimkan bukti pembayaran dengan	Sistem berhasil menyimpan data pesanan pengguna	Berhasil (valid)

	jumlah yg telah ditetapkan		
Melihat status pemesanan (pelanggan)	Pengguna melihat status pembayaran dan status pengiriman produk yang telah dipesan	Sistem berhasil menampilkan status pembayaran dan status pengiriman produk yang telah dipesan pengguna	Berhasil (valid)
Melihat dashboard (admin)	Admin yang telah login diarahkan menuju dashboard penjualan	Sistem mengarahkan admin menuju halaman dashboard penjualan	Berhasil (valid)
Manajemen Kategori (Admin)	Admin menambahkan kategori	Sistem berhasil menambahkan data kategori dan menampilkan kategori di tabel kategori	Berhasil (valid)
	Admin mengedit kategori	Sistem berhasil mengubah data terbaru dari kategori	Berhasil (valid)
	Admin menghapus kategori	Sistem berhasil menghapus kategori	Berhasil (valid)
Manajemen produk (admin)	Admin menambahkan produk	Sistem berhasil menambahkan produk dan menampilkan produk di halaman home pengguna	Berhasil (valid)
	Admin mengedit informasi produk	Sistem berhasil mengubah informasi terbaru dari produk	Berhasil (valid)
	Admin menghapus produk dari daftar produk	Sistem berhasil menghapus produk	Berhasil (valid)
Manajemen Kelola pesanan (admin)	Admin memverifikasi bukti pembayaran pesanan dan menekan	Sistem berhasil menampilkan bukti pembayaran dan	Berhasil (valid)

	tombol verifikasi jika pembayaran berhasil	mengubah status pembayaran dari "Menunggu verifikasi" menjadi "Terverifikasi"	
	Admin mengubah status pesanan pengguna dengan menekan tombol on the way dan delivered	Sistem berhasil mengubah status pesanan dari "in progress" menjadi "on the way" dan "delivered"	Berhasil (valid)

Berdasarkan Tabel 4.2, dapat disimpulkan bahwa seluruh fungsi utama dalam sistem telah diuji menggunakan metode *Black Box Testing* dan menunjukkan hasil yang valid. Pengujian meliputi fitur-fitur penting seperti *landing page*, *login* dan *register*, pencarian produk, pengelolaan keranjang belanja, transaksi pemesanan, monitoring status pesanan, serta manajemen produk dan pesanan oleh admin. Setiap skenario pengujian berhasil dijalankan sesuai dengan hasil yang diharapkan, yang membuktikan bahwa sistem berjalan stabil dan mampu memenuhi kebutuhan fungsional pengguna.

4.5.2 *User Acceptance Test* (UAT)

Selain pengujian teknis menggunakan metode Black Box Testing, penulis juga melakukan pengujian sistem menggunakan pendekatan User Acceptance Testing (UAT) untuk mengevaluasi apakah sistem yang dikembangkan telah sesuai dengan kebutuhan dan harapan pengguna akhir. UAT ini dilakukan secara daring melalui pertemuan video call menggunakan platform Zoom bersama pemilik UMKM Rahdatu Wooden Furniture.

Dalam sesi tersebut, penulis memberikan akses kepada sistem dan membimbing pemilik UMKM untuk mencoba fitur-fitur inti berdasarkan skenario nyata, seperti menambahkan produk, mengelola pesanan, dan melakukan simulasi pemesanan sebagai pelanggan. Seluruh umpan balik dari pemilik UMKM dicatat secara langsung untuk menilai aspek fungsionalitas dan kemudahan penggunaan.

Hasil pengujian UAT pada pemilik UMKM dapat dilihat pada Tabel 4.3.

Tabel 4.3 Hasil Pengujian UAT Terhadap Pemilik UMKM

Test Scenario	Umpan Balik	Evaluasi
Menjelajahi katalog dan melihat detail produk	Tampilan katalog sudah jelas, gambar dan informasi produk lengkap dan menarik.	Sesuai
Simulasi pemesanan produk sebagai pelanggan	Alur pemesanan mudah dipahami, proses checkout berjalan baik. Upload bukti pembayaran lancar.	Sesuai
Menambahkan, mengedit, dan menghapus produk	Form tambah/edit produk mudah diisi, meskipun tampilannya masih standar dan masih bisa disempurnakan.	Sesuai (perlu penyempurnaan visual)
Melihat dan mengelola pesanan pelanggan	Informasi pesanan lengkap dan sistem verifikasi berjalan. Fitur ubah status sangat membantu.	Sesuai
Akses sebagai admin dan pelanggan	Pembagian hak akses admin dan pelanggan berjalan baik dan tidak membingungkan.	Sesuai
Navigasi dan tampilan sistem secara keseluruhan	Navigasi mudah, meski pemilik menyarankan tampilan visual lebih modern agar lebih profesional.	Sesuai (perlu penyempurnaan visual)

Berdasarkan hasil pengujian UAT terhadap pemilik UMKM Rahdatu Wooden Furniture, sistem dinilai telah memenuhi sebagian besar kebutuhan operasional bisnis harian. Pemilik menyatakan bahwa proses pencatatan pesanan dan pengelolaan stok kini menjadi lebih mudah dan efisien, tanpa perlu pencatatan manual. Fitur seperti penambahan, pengeditan, dan penghapusan produk dapat dijalankan dengan lancar melalui antarmuka admin, serta sistem login khusus berhasil membedakan akses admin dan pelanggan dengan baik.

Selain itu, pemilik merasa terbantu dengan adanya fitur pengelolaan pesanan yang memungkinkan verifikasi bukti pembayaran secara langsung dan pengubahan status pesanan secara real time. Website juga mampu menampilkan katalog produk dengan lengkap, mencakup gambar, deskripsi, dan harga, serta memungkinkan pelanggan untuk memesan langsung tanpa harus menghubungi admin. Beberapa saran diberikan terkait penyempurnaan

tampilan visual dan penambahan notifikasi sistem, namun secara fungsional, website sudah layak digunakan dalam kegiatan bisnis.

Selain pemilik UMKM, penulis juga melakukan pengujian langusng kepada tiga orang responden sebagai calon pelanggan. Responden pertama dan kedua diujikan kepada mahasiswa dan responden ketiga diujikan kepada seorang ibu rumah tangga. Mereka diminta untuk mencoba langsung mulai dari proses login hingga proses mengecek riwayat pesanan mereka.

Hasil pengujian UAT terhadap calon pelanggan dapat dilihat pada Tabel 4.4.

Tabel 4.4 Hasil Pengujian UAT Terhadap Calon Pelanggan

Test Scenario	Responden		
	Responden 1	Responden 2	Responden 3
Login dan Register	Tampilan warna halaman tidak sama dengan tema *e-commerce,* desainnya terlalu simple namun sudah fungsional.	Proses pendaftaran dan login berjalan lancar. Formulir registrasi mudah diisi dan validasi inputnya berfungsi.	Awalnya saya agak bingung karena belum biasa buka website seperti ini, tapi setelah diarahkan sebentar, proses daftarnya mudah dan tampilannya jelas.
Kesimpulan	Fitur login dan registrasi berjalan lancar dan mudah digunakan, meskipun desain tampilan masih sederhana.		
Melihat katalog produk	Gambar yang tertampil di katalog produk terlalu kecil.	Katalog produk tampil rapi dan informatif, tetapi akan lebih praktis jika dilengkapi filter kategori atau urutan harga.	Halaman utamanya bagus dan tidak terlalu ramai, saya bisa langsung lihat produk-produk yang dijual tanpa harus klik banyak-banyak.
Kesimpulan	Pengguna dapat melihat katalog produk dengan baik, namun ukuran gambar perlu diperbesar dan fitur filter perlu ditambahkan.		
Melihat detail produk	Tampilan responsive dan menarik.	Informasi di halaman detail produk sudah	Gambarnya besar dan tulisannya jelas,

	Gambar, nama, harga, dan deskripsi produk tertampil dengan jelas.	lengkap dan mudah dipahami, cukup membantu sebelum memutuskan membeli.	jadi saya bisa tahu harga dan jenis barangnya sebelum beli.
Kesimpulan	Informasi produk seperti gambar, deskripsi, dan harga tertampil dengan lengkap dan mudah dipahami oleh pengguna dari berbagai latar belakang.		
Memasukkan produk ke keranjang	Fitur berfungsi dengan baik dan mudah digunakan.	Fitur tambah ke keranjang berfungsi baik, terdapat notifikasi produk berhasil dimasukkan ke dalam keranjang.	Saya suka karena tombolnya besar dan gampang ditemukan, saya tahu produk sudah masuk ke keranjang karena ada tulisan konfirmasi.
Kesimpulan	Fitur berfungsi dengan baik dan dilengkapi notifikasi visual yang membantu.		
Melakukan checkout	Proses checkout terlihat mudah dijalani namun lebih baik ditambahkan fitur pembayaran otomatis.	Proses checkout berjalan lancar dan formnya mudah dipahami, seluruh tahapan dari pengisian data hingga pengiriman pesanan terasa jelas dan tidak membingungkan.	Form checkout-nya kelihatan rapi dan tidak terlalu banyak isian, saya tinggal isi sesuai petunjuk dan langsung kirim pesanan.
Kesimpulan	Checkout berjalan lancar dan mudah dipahami, namun disarankan untuk menambahkan fitur pembayaran otomatis.		
Mengecek status dan riwayat pesanan	Riwayat pesanan tertampil dengan baik namun perlu	Halaman riwayat pesanan sudah membantu mengecek	Setelah pesan, saya bisa lihat di bagian pesanan saya,

	dibuat penomoran pesanan jadi sekali pesan bisa banyak produk dan tidak satu-satu.	status, tapi tampilannya masih terasa polos dan kurang informatif secara visual.	tampilannya sederhana jadi saya tahu pesanan saya masih diproses atau belum.
Kesimpulan	Pengguna dapat memantau status pesanan dengan mudah, tapi tampilan perlu ditingkatkan dan perlu penomoran transaksi.		

Hasil pengujian UAT terhadap tiga calon pelanggan menunjukkan bahwa sistem mampu memberikan pengalaman belanja yang cukup baik dan mudah digunakan. Calon pelanggan dapat melakukan login dan registrasi dengan lancar, melihat katalog produk dengan informasi yang lengkap, serta mengakses detail produk secara jelas. Fitur pemesanan seperti menambahkan produk ke keranjang, checkout, hingga unggah bukti pembayaran telah berfungsi sesuai harapan dan dilengkapi dengan notifikasi yang membantu.

Selain itu, pengguna juga dapat mengakses riwayat pesanan melalui halaman “My Orders” untuk memantau status pembelian mereka. Beberapa umpan balik yang diberikan mencakup saran agar ukuran gambar produk diperbesar, tampilan riwayat pesanan lebih menarik, dan fitur filter katalog ditambahkan.

Berdasarkan umpan balik dan saran yang telah diberikan oleh pemilik UMKM dan para calon pelanggan, sistem telah memenuhi kebutuhan dasar pengguna akhir dari sisi fungsionalitas dan kemudahan penggunaan, meskipun masih terbuka untuk pengembangan lebih lanjut di sisi estetika dan fitur tambahan dan akan dijelaskan lebih lanjut pada Bab V.

Dengan demikian, sistem layak digunakan dan dapat diimplementasikan oleh UMKM sebagai solusi digital dalam mendukung aktivitas bisnis di era industri 4.0.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil dari proses analisis kebutuhan, perancangan, implementasi, dan pengujian sistem yang telah dilakukan dalam pengembangan website e-commerce furnitur berbasis Laravel untuk UMKM Rahdatu Wooden Furniture, maka dapat disimpulkan beberapa poin berikut:

1. Website e-commerce yang dikembangkan berhasil memenuhi kebutuhan utama UMKM dalam hal digitalisasi proses penjualan furnitur, mulai dari penyajian katalog produk hingga sistem pemesanan dan manajemen pesanan.
2. Sistem berhasil diimplementasikan menggunakan *framework* Laravel dengan struktur arsitektur Model-View-Controller (MVC), yang terbukti mendukung pengembangan aplikasi secara modular, efisien, dan mudah dipelihara.
3. Fitur-fitur penting seperti katalog produk, login/register pengguna, manajemen produk, keranjang belanja, manajemen pesanan, dan riwayat transaksi telah berjalan dengan baik dan sesuai dengan kebutuhan pengguna berdasarkan hasil wawancara dan observasi.
4. Pengujian sistem menggunakan Black Box Testing dan User Acceptance Test (UAT) menunjukkan bahwa seluruh fitur fungsional berjalan dengan baik dan dapat diterima oleh pemilik UMKM sebagai solusi yang membantu dalam operasional bisnis.
5. Pengembangan sistem ini membuktikan bahwa pemanfaatan teknologi web dapat memberikan kontribusi nyata dalam mendukung transformasi digital UMKM, khususnya di sektor furnitur, dengan meningkatkan efisiensi operasional dan memperluas jangkauan pemasaran.

5.2 Saran

Agar sistem yang telah dikembangkan dapat lebih optimal dan relevan untuk digunakan dalam jangka panjang, berikut beberapa saran yang dapat dipertimbangkan untuk menjadi bahan penelitian selanjutnya:

1. Integrasi Sistem Pembayaran Otomatis : Untuk meningkatkan kenyamanan pengguna dan efisiensi transaksi, disarankan agar website dikembangkan lebih lanjut dengan integrasi sistem pembayaran otomatis seperti Midtrans, Xendit, atau DANA.

2. Pengembangan Fitur Pelacakan Pengiriman : Penambahan fitur pelacakan pesanan secara real-time akan meningkatkan kepercayaan dan transparansi bagi pelanggan.
3. Optimasi Responsivitas dan Aksesibilitas Mobile : Diperlukan pengembangan lebih lanjut dalam hal optimasi SEO dan desain Antar muka yang mobile-friendly agar website dapat menjangkau audiens yang lebih luas dan memiliki performa baik di perangkat seluler.
4. Penerapan Dashboard Analytics : Penambahan fitur analitik untuk memantau statistik penjualan, produk terlaris, dan perilaku pengguna dapat membantu pemilik UMKM dalam mengambil keputusan strategis berbasis data.
5. Pengembangan Aplikasi Mobile : Untuk meningkatkan kemudahan akses bagi pelanggan, pengembangan aplikasi mobile berbasis Android/iOS menjadi langkah strategis jangka panjang.
6. Pemanfaatan Kecerdasan Buatan (AI) : Integrasi teknologi AI untuk fitur rekomendasi produk atau chatbot akan menambah nilai tambah dan meningkatkan personalisasi layanan pelanggan.
7. Evaluasi dan Umpan Balik Berkala : Disarankan untuk melakukan evaluasi sistem secara berkala dan terus mengumpulkan masukan dari pengguna untuk memastikan sistem tetap relevan dan dapat berkembang mengikuti kebutuhan pasar.

DAFTAR PUSTAKA

Alam, M., & Ahmad, A. (2021). *Software testing methods: A comparative study*. Journal of Software Engineering and Applications, 14(2), 97–105.

Aljawarneh, S., & Al-Omari, S. A. (2018). Software implementation success model: A user-centered approach. *Journal of Systems and Software*, 138, 123–136.

Alam, M., & Ahmad, I. (2021). *Software Testing Techniques: A Literature Review*. International Journal of Computer Applications, **183**(2), 15–20. https://doi.org/10.5120/ijca2021921201

Badan Pusat Statistik. (2023). *Statistik UMKM sektor industri pengolahan*. https://www.bps.go.id/

Booch, G., Rumbaugh, J., & Jacobson, I. (2005). *The unified modeling language user guide* (2nd ed.). Addison-Wesley.

Chaffey, D., & Ellis-Chadwick, F. (2019). *Digital marketing* (7th ed.). Pearson Education.

DuBois, P. (2013). *MySQL* (5th ed.). Addison-Wesley.

Fowler, M. (2003). *UML distilled: A brief guide to the standard object modeling language* (3rd ed.). Addison-Wesley.

Jabar, J. A., & Anubhakti, D. (2021). Pemodelan UML dalam pengembangan sistem informasi. *Jurnal Informatika*, 9(2), 115–124.

Kementerian Komunikasi dan Informatika. (2022). *Transformasi digital UMKM Indonesia*. https://kominfo.go.id/

Kementerian Koperasi dan UKM. (2021). *Laporan tahunan pengembangan UMKM*. https://kemenkopukm.go.id/

Laudon, K. C., & Traver, C. G. (2021). *E-commerce 2021: Business, technology, society* (16th ed.). Pearson.

Laravel Team. (2025). *Laravel documentation*. https://laravel.com/docs

Morville, P., & Rosenfeld, L. (2015). *Information architecture: For the web and beyond* (4th ed.). O'Reilly Media.

Milisavljevic-Syed, M., Peris-Ortiz, M., & Cabrera, E. (2020). *System design thinking and development*. Springer.

Purwanegara, M. S., Andika, M. H., & Widjaja, A. E. (2021). *Digital transformation for SMEs: Evidence from Indonesia*. Journal of Small Business Strategy, 31(2), 55–68.

Patel, R., & Amin, D. (2020). *Black box testing techniques in software engineering*. International Journal of Computer Applications, 175(7), 10–15.

Pressman, R. S. (2010). *Software engineering: A practitioner's approach* (7th ed.). McGraw-Hill.

Royce, W. W. (1970). Managing the development of large software systems. *Proceedings of IEEE WESCON*, 1–9.

Rahayu, L., & Putra, A. D. (2022). Pengembangan platform e-commerce UMKM berbasis Laravel dengan metode waterfall. *Jurnal Teknologi Informasi dan Komputer*, 6(3), 101–110.

Stauffer, M. (2019). *Laravel: Up and running* (2nd ed.). O'Reilly Media.

Sommerville, I. (2011). *Software engineering* (9th ed.). Addison-Wesley.

Sari, F., & Hidayatullah, M. (2021). Pengaruh marketplace dan digital marketing terhadap peningkatan penjualan UMKM. *Jurnal Bisnis dan Pemasaran Digital*, 5(4), 25–33.

Sugiyono. (2018). *Metode penelitian kuantitatif, kualitatif, dan R&D*. Alfabeta.

Sukamto, R., & Shalahuddin, M. (2013). *Rekayasa perangkat lunak: Terstruktur dan berorientasi objek*. Bandung: Informatika.

Wahyudi, T., & Mahendra, R. (2023). Transformasi digital bisnis UMKM di Indonesia setelah masa pandemi. *Jurnal Ekonomi Digital*, 3(1), 1–12.