BAB IV

RANCANGAN SISTEM DAN PROGRAM USULAN

4.1 Analisis Kebutuhan Software

Sistem manajemen laundry merupakan sebuah aplikasi berbasis web yang dirancang untuk membantu pengelolaan operasional usaha laundry secara digital dan terintegrasi. Sistem ini berfungsi sebagai solusi pengganti dari metode pencatatan manual yang seringkali rawan kesalahan, kurang efisien, serta sulit dalam pelacakan data transaksi dan riwayat pelanggan.

Dengan adanya sistem ini, proses bisnis pada usaha laundry menjadi lebih sistematis dan terstruktur. Proses transaksi seperti pencatatan pelanggan, jenis layanan yang digunakan, perhitungan total harga, status pembayaran, serta status cucian dapat dikelola secara real-time oleh pihak kasir. Sementara itu, admin memiliki akses untuk memantau dan mengelola seluruh aktivitas yang berlangsung di sistem, termasuk pengelolaan kasir, layanan, serta laporan transaksi.

Sistem ini dibangun menggunakan framework Laravel, yaitu sebuah kerangka kerja PHP yang populer dan mendukung arsitektur MVC (*Model-View-Controller*). Laravel dipilih karena memiliki fitur keamanan yang baik, struktur pengembangan yang terorganisir, serta mendukung berbagai fungsionalitas modern seperti validasi form, otentikasi pengguna, dan integrasi database yang efisien.

Aplikasi ini dirancang agar dapat diakses melalui berbagai web browser, seperti Google Chrome, Mozilla Firefox, dan sejenisnya. Antarmuka sistem dibuat sederhana, responsif, serta intuitif, sehingga memudahkan pengguna tanpa latar belakang teknis yang kuat, khususnya bagi kasir laundry yang bertugas dalam pencatatan transaksi harian.

UNIVERSITAS NUSA MANDIRI

Berikut adalah spesifikasi kebutuhan sistem manajemen laundry:

1. Kebutuhan Fungsional Admin
    1) Admin dapat login ke dalam sistem menggunakan akun yang valid.
    2) Admin dapat menambahkan, mengubah, dan menghapus data kasir.
    3) Admin dapat menambahkan dan mengelola layanan (jenis cucian dan harga).
    4) Admin dapat melihat seluruh transaksi laundry.
    5) Admin dapat melihat laporan dan riwayat transaksi berdasarkan filter tanggal.
    6) Admin dapat mengelola promo (kode diskon dan kuota).

2. Kebutuhan Fungsional Kasir
    1) Kasir dapat login ke dalam sistem menggunakan akun yang valid.
    2) Kasir dapat mencatat transaksi baru (nama pelanggan, layanan, jumlah, harga, promo).
    3) Kasir dapat memilih apakah pembayaran dilakukan lunas atau belum.
    4) Kasir dapat memperbarui status cucian menjadi “Proses”, “Selesai”, atau “Diambil”.
    5) Kasir dapat mencetak nota pembayaran setelah transaksi disimpan.
    6) Kasir dapat melihat riwayat transaksi.

3. Kebutuhan Fungsional Pelanggan
    1) Pelanggan tidak mengakses sistem secara langsung, tetapi data mereka dicatat oleh kasir.

UNIVERSITAS NUSA MANDIRI

2) Data pelanggan tersimpan untuk digunakan ulang dalam transaksi berikutnya.
3) Nama pelanggan digunakan dalam laporan dan pencetakan nota.

4. Kebutuhan Fungsional Sistem Pembayaran
   1) Sistem menyediakan opsi metode pembayaran: tunai, transfer, atau QRIS.
   2) Sistem secara otomatis menghitung total pembayaran berdasarkan layanan, berat, promo, dan biaya tambahan.
   3) Sistem dapat memvalidasi kode promo jika tersedia dan mengurangi kuota promo yang digunakan.

5. Kebutuhan Non-Fungsional
   1) Sistem dapat diakses melalui browser modern seperti Chrome dan Firefox.
   2) Sistem menggunakan otentikasi berbasis sesi agar aman dari akses tidak sah.
   3) Aplikasi dirancang responsif dan dapat diakses di berbagai ukuran layar.
   4) Sistem dapat di-*deploy* secara lokal (localhost) maupun online (web hosting).
   5) Sistem menyimpan data secara terstruktur menggunakan database MySQL.

4.2 Desain

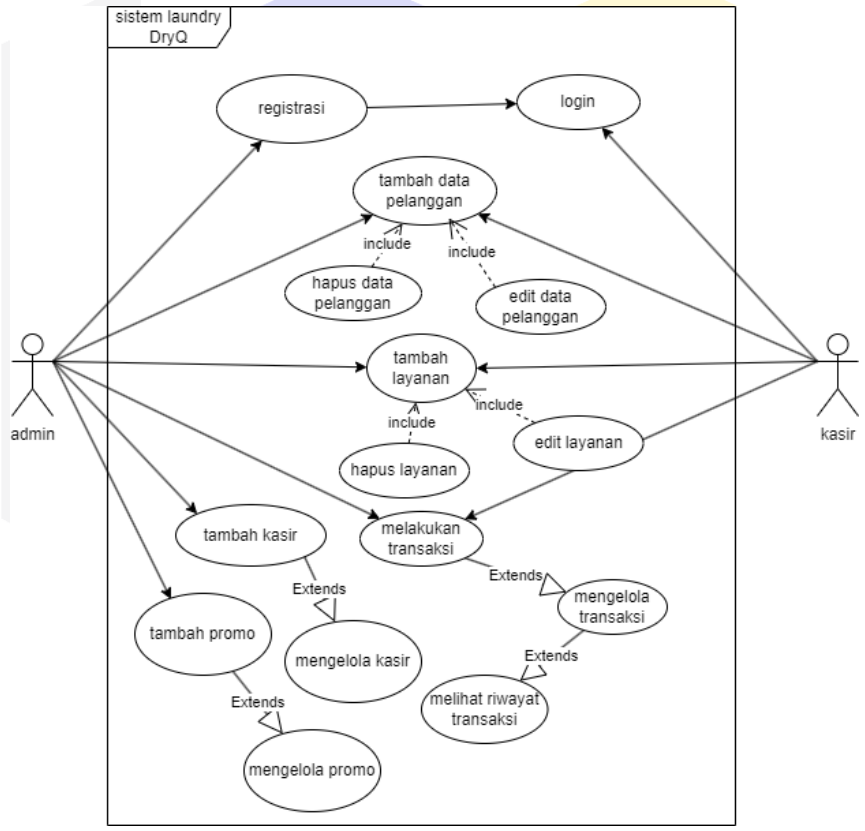
Perancangan sistem dilakukan untuk menggambarkan bagaimana sistem usulan akan bekerja, mulai dari proses bisnis, alur interaksi antar pengguna, hingga struktur data dan tampilan antarmuka. Pada bagian ini, penulis memodelkan sistem dari sisi fungsional (proses bisnis), struktur data (database), dan user interface (tampilan pengguna).

UNIVERSITAS NUSA MANDIRI

4.2.1 Desain Pemodelan Sistem

1. Pemodelan *Use Case Diagram*

Use Case Diagram digunakan untuk menggambarkan interaksi antara aktor (pengguna sistem) dengan fitur-fitur utama dari sistem manajemen laundry. Aktor dalam sistem ini terdiri dari dua jenis, yaitu Admin dan Kasir, yang masing-masing memiliki hak akses dan tanggung jawab yang berbeda.



Sumber : Penelitian

Gambar IV. 1 Use Case Diagram Interaksi Aktor Dengan Sistem

1) Deskripsi *Use Case Diagram*: Registrasi Admin

Tabel IV. 1 Use Case Diagram: Registrasi

Use Case Description	Proses registrasi akun admin utama sistem laundry untuk pertama kali agar dapat mengelola sistem.		
Actors	Pengguna (Calon Admin)		
Pre-Condition	1. Halaman registrasi diakses.		
	2. Sistem belum memiliki admin utama.		
Post-Condition	Sistem berhasil menyimpan akun admin dan pengguna dapat login sebagai admin.		
Fault Condition	Jika form tidak lengkap, sistem menampilkan pesan kesalahan.		
	Jika email sudah digunakan, registrasi gagal.		
Main Scenarios		Extensions	
Serial No.	**Step**	**No.**	**Kondisi**
1	Pengguna membuka halaman registrasi.		
2	Pengguna mengisi data registrasi (nama, email, password, nama usaha).	2a	Jika email tidak valid, sistem menampilkan pesan error.
		2b	Jika password kurang dari 6 karakter, sistem menampilkan pesan kesalahan.
3	Sistem melakukan validasi input.		
4	Jika valid, sistem menyimpan data dan membuat akun admin.		

Sumber : Penelitian

2) Deskripsi *Use Case Diagram*: Login Sistem

Tabel IV. 2 Use Case Diagram: Login

Use Case Name	**Login Sistem**	
Use Case Description	Admin dan kasir dapat login ke dalam sistem dengan akun yang sudah terdaftar untuk mengakses fitur sesuai hak aksesnya.	
Actors	Admin, Kasir	
Pre-Condition	1. Sistem terhubung ke internet/server lokal.	
	2. Halaman login diakses melalui browser.	
	3. Username dan password dimasukkan.	
Post-Condition	Jika valid, pengguna diarahkan ke halaman dashboard sesuai rolenya.	
Fault Condition	Jika password salah atau username tidak ditemukan, sistem menolak login.	
	Jika server down, proses login gagal.	
Main Scenarios		Extensions

Serial No.	Step	No.	Kondisi
1	Pengguna memasukkan username dan password.		
2	Sistem memverifikasi data login di database.	2a	Jika username tidak ditemukan, sistem menampilkan error "Username tidak ditemukan".
		2b	Jika password salah, sistem menampilkan error "Password salah".
3	Jika valid, pengguna diarahkan ke dashboard.		

Sumber : Penelitian

3) Deskripsi *Use Case Diagram*: Kelola Layanan

Tabel IV. 3 Use Case Diagram: Kelola Layanan

Use Case Name	Kelola Layanan		
Use Case Description	admin dan kasir dapat menambah, mengubah, dan menghapus jenis layanan laundry yang tersedia di sistem.		
Actors	Admin dan kasir		
Pre-Condition	1. telah login ke sistem.		
	2. mengakses menu layanan.		
Post-Condition	Sistem akan menyimpan perubahan data layanan ke dalam database.		
Fault Condition	1. Jika input tidak valid (misal: harga kosong), sistem menolak penyimpanan.		
	2. Jika koneksi gagal, data tidak tersimpan.		
Main Scenarios		**Extensions**	
Serial No.	**Step**	**No.**	**Kondisi**
1	membuka halaman daftar layanan.		
2	memilih aksi: tambah, edit, atau hapus layanan.		
3	mengisi atau mengubah nama layanan dan harga per kg.	3a	Jika harga tidak valid (misalnya huruf), sistem menampilkan error.
		3b	Jika nama layanan kosong, sistem menolak penyimpanan.
4	menekan tombol simpan.		
5	Sistem menyimpan perubahan ke database.		

Sumber : Penelitian

4) Deskripsi *Use Case Diagram*: Tambah Transaksi Cucian

Tabel IV. 4 Use Case Diagram: Tambah Transaksi

Use Case Name	**Tambah Transaksi Cucian**		
Use Case Description	Kasir menambahkan data transaksi baru berupa nama pelanggan, jenis layanan, jumlah berat, status pembayaran, dan total harga. Jika tersedia, kasir dapat memasukkan kode promo dan menambahkan catatan.		
Actors	Kasir		
Pre-Condition	1. Kasir harus login terlebih dahulu.		
	2. Halaman tambah transaksi sudah diakses.		
	3. Data layanan dan pelanggan tersedia.		
Post-Condition	Data transaksi baru berhasil disimpan ke dalam sistem dengan status awal "Baru".		
Fault Condition	1. Jika input tidak valid, maka transaksi tidak dapat disimpan.		
	2. jika koneksi database gagal, proses simpan dibatalkan.		
	3. Jika kode promo tidak valid, sistem menampilkan pesan kesalahan.		
Main Scenarios		Extensions	
Serial No.	**Step**	**No.**	**Kondisi**
1	Kasir memasukkan nama pelanggan dan memilih layanan.		
2	Sistem menghitung total berdasarkan berat dan harga layanan.	2a	Jika berat tidak diisi atau nol, sistem menampilkan pesan kesalahan.
		2b	Jika kode promo tidak valid atau kuota habis, sistem menolak kode dan menampilkan pesan.
		2c	Jika total belum dihitung, sistem tidak mengizinkan penyimpanan.
3	Kasir mengisi status pembayaran dan menambahkan catatan (opsional).		
4	Kasir menekan tombol "Simpan".		
5	Sistem menyimpan transaksi dengan status "Baru".		

Sumber : Penelitian

5) Deskripsi *Use Case Diagram*: Selesaikan Cucian

Tabel IV. 5 Use Case Diagram: Selesaikan Cucian

Use Case Name	**Selesaikan Cucian**
Use Case Description	Kasir dapat mengubah status cucian dari “proses” menjadi “selesai” jika transaksi sudah lunas.
Actors	Kasir
Pre-Condition	1. Kasir telah login.2. Status pembayaran transaksi adalah “lunas”.

Post-Condition	Status cucian berubah menjadi “selesai” dan dapat dilihat di halaman riwayat.		
Fault Condition	- Jika status pembayaran belum lunas, tombol “selesaikan” tidak aktif.- Jika koneksi database gagal, status tidak berubah.		
Main Scenarios		Extensions	
Serial No.	**Step**	**No.**	**Kondisi**
1	Kasir membuka halaman daftar cucian.		
2	Kasir mencari cucian dengan status “proses”.		
3	Kasir klik tombol “Selesai” pada transaksi yang sudah lunas.	3a	Jika status pembayaran = belum lunas, sistem menonaktifkan tombol “Selesai”.
4	Sistem meminta konfirmasi.	4a	Jika kasir membatalkan konfirmasi, proses dibatalkan.
5	Kasir menekan “Ya”.		
6	Sistem mengubah status cucian menjadi “selesai”.		

Sumber : Penelitian

6) Deskripsi *Use Case Diagram*: Tambah Kasir

Tabel IV. 6 Use Case Diagram: Tambah Kasir

Use Case Name	**Tambah Kasir**		
Use Case Description	Admin menambahkan akun kasir baru yang dapat digunakan untuk mengakses dan mencatat transaksi laundry.		
Actors	Admin		
Pre-Condition	Admin sudah login dan berada di halaman manajemen kasir.		
Post-Condition	Data kasir baru tersimpan dan bisa digunakan untuk login.		
Fault Condition	1. Jika email kasir sudah terdaftar, sistem menolak penyimpanan.		
	2. Jika password terlalu pendek, sistem menampilkan pesan kesalahan.		
Main Scenarios		Extensions	
Serial No.	**Step**	**No.**	**Kondisi**
1	Admin membuka halaman tambah kasir.		
2	Admin mengisi form data kasir (nama, email, password).	2a	Jika email tidak valid, sistem menampilkan pesan kesalahan.
		2b	Jika password kurang dari 6 karakter, sistem menampilkan error.
3	Sistem melakukan validasi.		
4	Sistem menyimpan data dan menampilkan pesan berhasil.		

Sumber : Penelitian

7) Deskripsi *Use Case Diagram*: Tambah Promo

Tabel IV. 7 Use Case Diagram: Tambah Promo

Use Case Name	**Tambah Promo**		
Use Case Description	Admin menambahkan kode promo yang bisa digunakan oleh kasir saat transaksi, untuk memberikan potongan harga ke pelanggan.		
Actors	Admin		
Pre-Condition	Admin login dan mengakses halaman manajemen promo.		
Post-Condition	Promo baru berhasil disimpan dan aktif.		
Fault Condition	1. Jika kode promo duplikat, sistem menolak penyimpanan		
	2. Jika diskon tidak valid, sistem menampilkan pesan error.		
Main Scenarios		**Extensions**	
Serial No.	**Step**	**No.**	**Kondisi**
1	Admin membuka halaman tambah promo.		
2	Admin mengisi data promo (kode, deskripsi, diskon %, kuota).	2a	Jika diskon lebih dari 100%, sistem menolak penyimpanan.
		2b	Jika kuota tidak diisi, sistem menampilkan pesan kesalahan.
3	Sistem memvalidasi data.		
4	Sistem menyimpan promo dan menampilkannya dalam daftar.		

Sumber : Penelitian

2. Pemodelan *Activity Diagram*

Activity Diagram adalah diagram yang digunakan untuk menggambarkan alur proses (*workflow*) dari satu *use case* tertentu secara terperinci. Dalam sistem manajemen laundry, *activity diagra*m digunakan untuk memahami bagaimana suatu aktivitas dimulai, dijalankan, bercabang, hingga selesai.

Dalam pembuatan *activity diagram*, terdapat dua pendekatan:

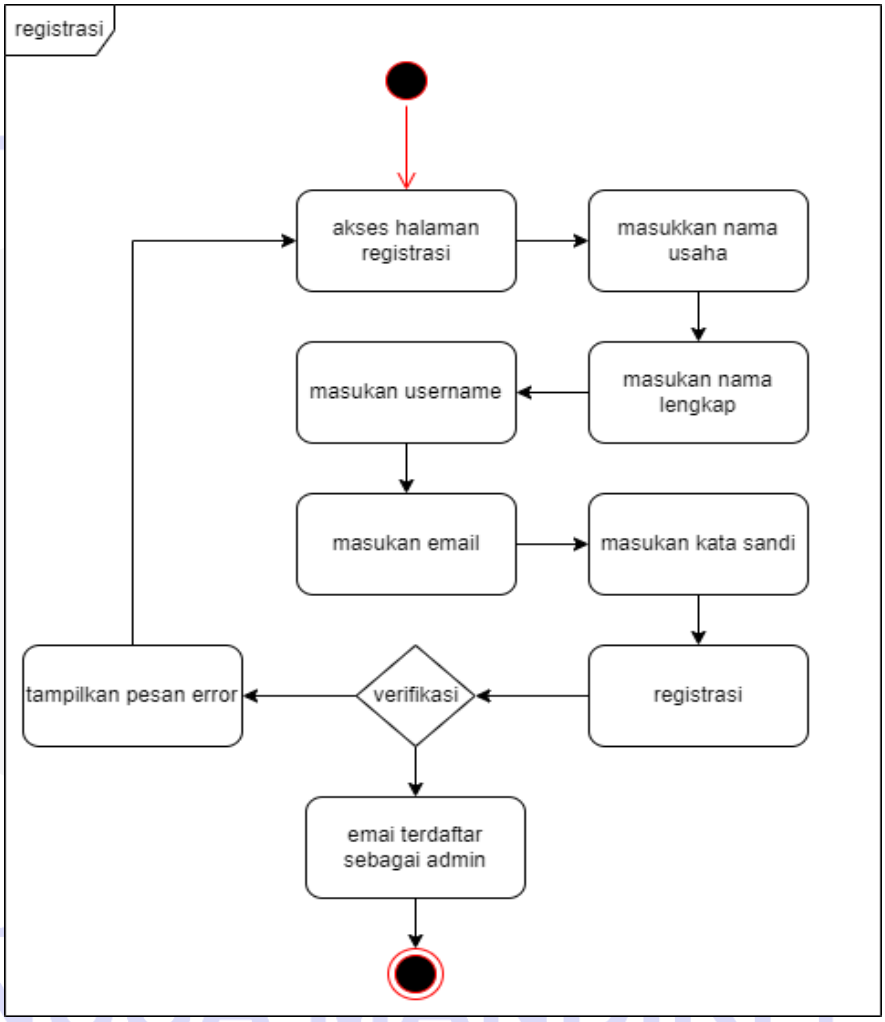
1) Menggunakan *swimlane* untuk memisahkan aktivitas berdasarkan aktor (contohnya admin, kasir).

2) Tanpa *swimlane*, cukup menampilkan alur logika satu *use case* secara umum.

UNIVERSITAS NUSA MANDIRI

Dalam dokumen ini, penggambaran *activity diagram* dilakukan berdasarkan satu *use case* per *diagram*, untuk menjaga kejelasan dan fokus terhadap satu proses bisnis utama.

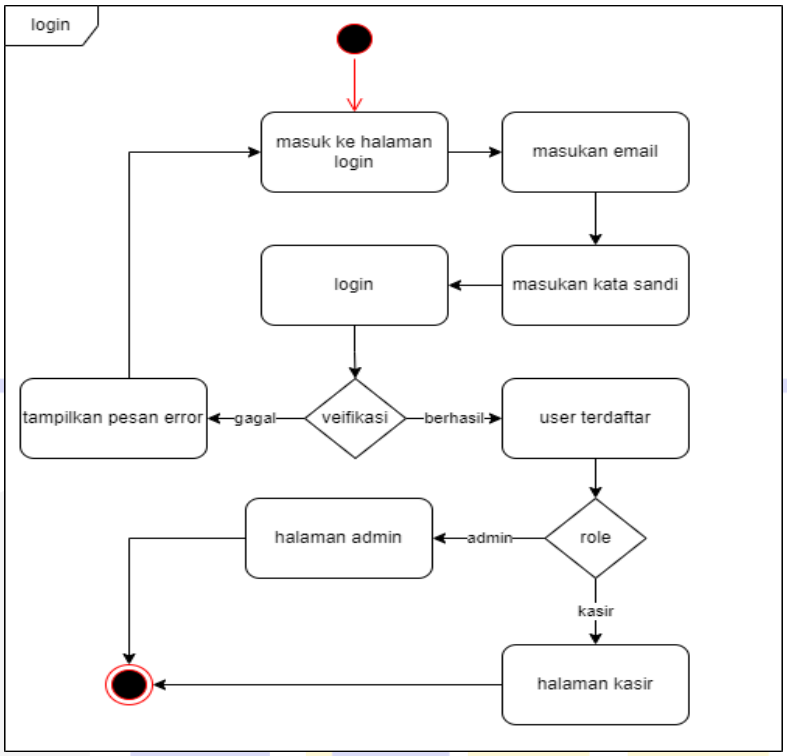
1) *Activity Diagram* registrasi (admin)



Sumber : Penelitian

Gambar IV. 2 Activity Diagram Registrasi

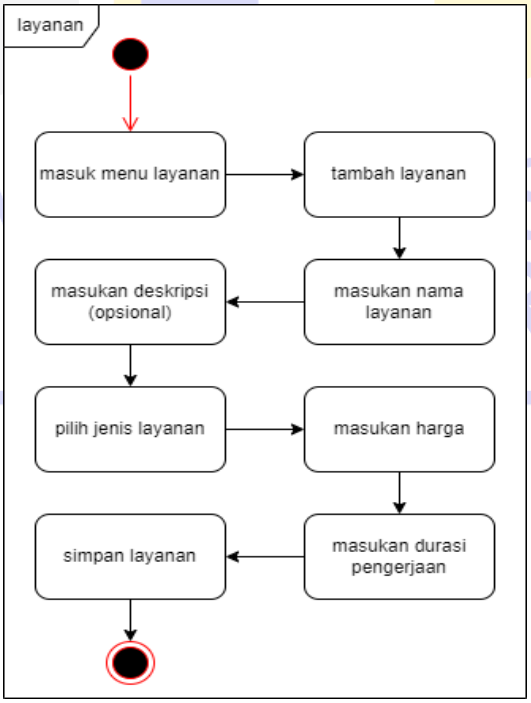
2) *Activity Diagram* login (admin dan kasir)



Sumber : Penelitian

Gambar IV. 3 Activity Diagram Login

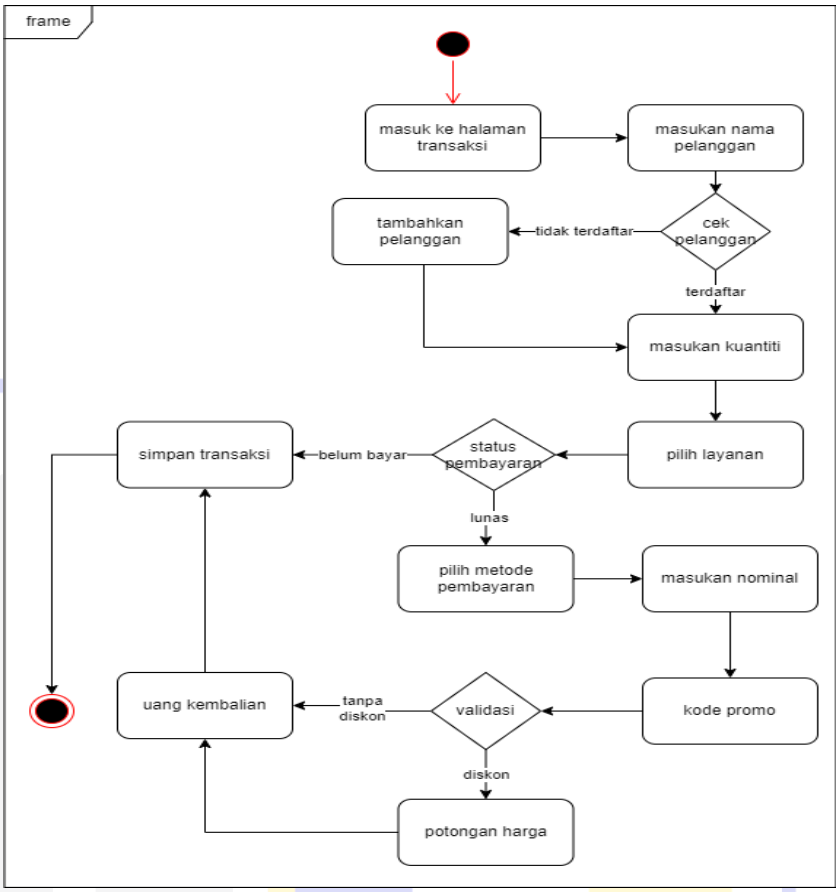
3) *Activity Diagram* tambah layanan (admin dan kasir)



Sumber : Penelitian

Gambar IV. 4 Activity Diagram Tambah Layanan

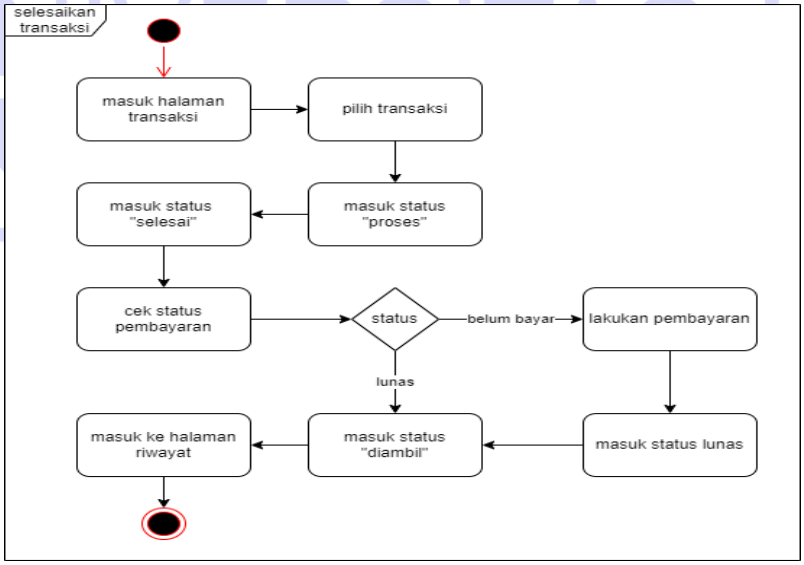
4) *Activity Diagram* transaksi (admin dan kasir)



Sumber : Penelitian

Gambar IV. 5 Activity Diagram Transaksi

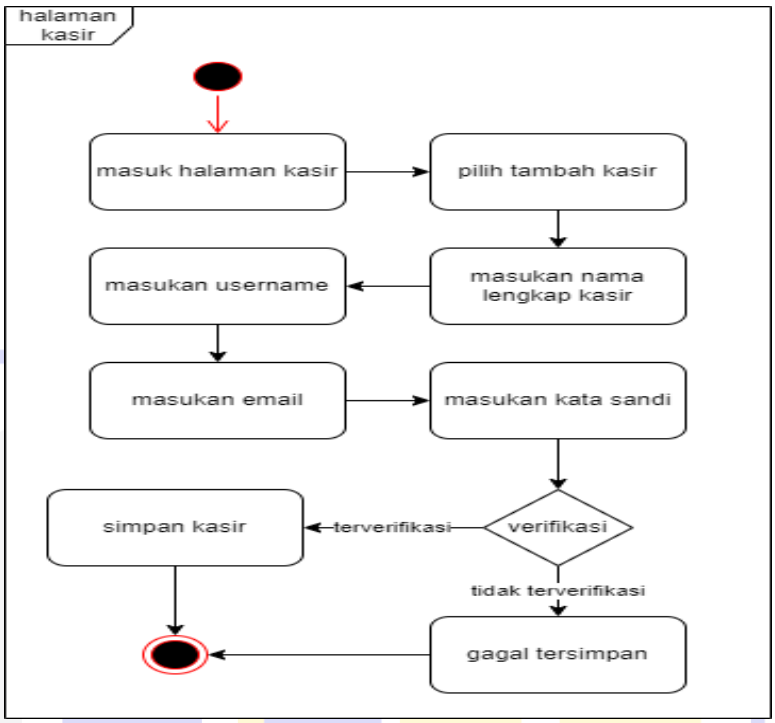
5) *Activity Diagram* selesaikan transaksi (admin dan kasir)



Sumber : Penelitian

Gambar IV. 6 Activity Diagram Selesaikan Transaksi

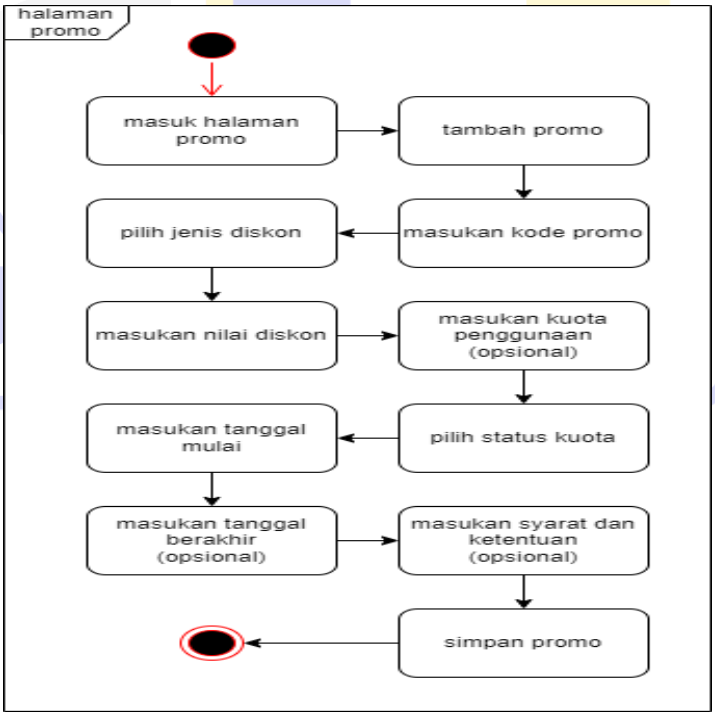
6) *Activity Diagram* tambah kasir (admin)



Sumber : Penelitian

Gambar IV. 7 Activity Diagram Tambah Kasir

7) *Activity Diagram* tambah promo (admin)

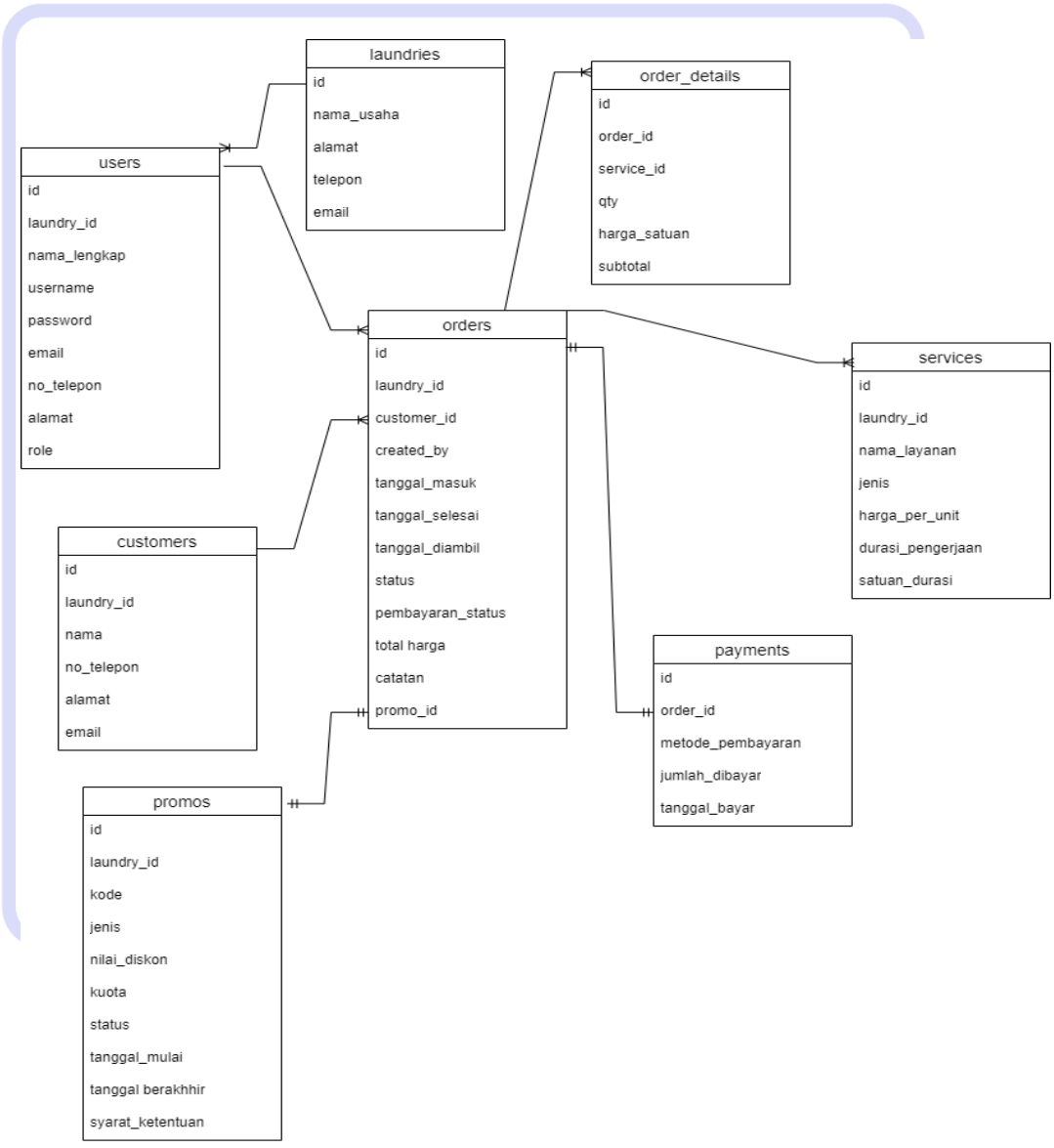


Sumber : Penelitian

Gambar IV. 8 Activity Diagram Tambah Promo

3. Pemodelan *Class Diagram*

Pemodelan *Class Diagram* berfungsi untuk menggambarkan struktur objek dalam sistem beserta hubungan antar objek (*class*), atribut, dan operasi (*method*) yang dimilikinya. *Class Diagram* merupakan bagian dari pemodelan berorientasi objek dan cocok untuk sistem kamu yang berbasis Laravel.



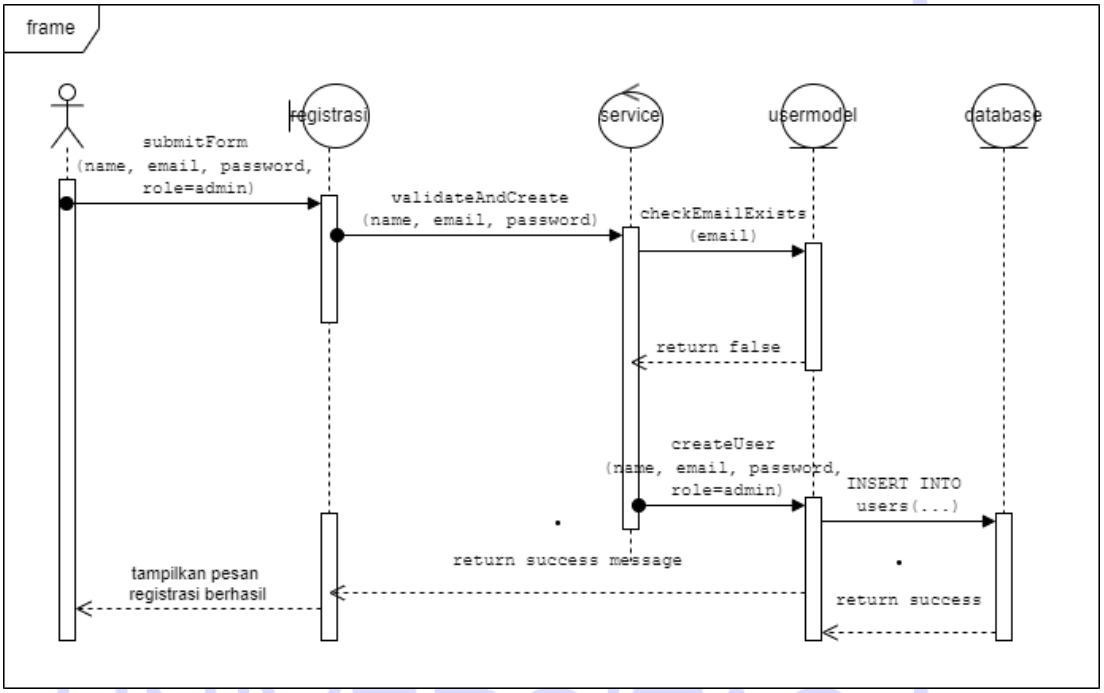
Sumber : Penelitian

Gambar IV. 9 Class Diagram

4. Pemodelan *Sequence Diagram*

Sequence Diagram digunakan untuk menggambarkan alur komunikasi antar objek pada satu *use case* tertentu secara berurutan. Diagram ini menunjukkan bagaimana alur kerja dari login dimulai dari user hingga sistem melakukan autentikasi dan memberikan respon, baik sukses maupun gagal.

1) *Sequence Diagram* Registrasi

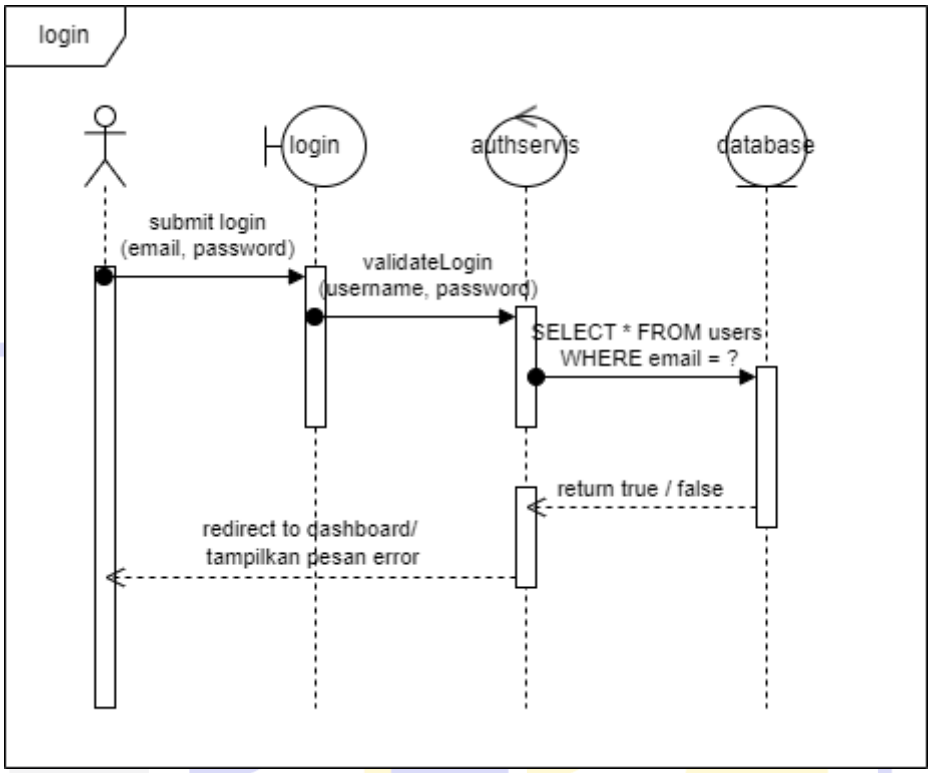


Sumber : Penelitian

Gambar IV. 10 Sequence Diagram Registrasi

UNIVERSITAS NUSA MANDIRI

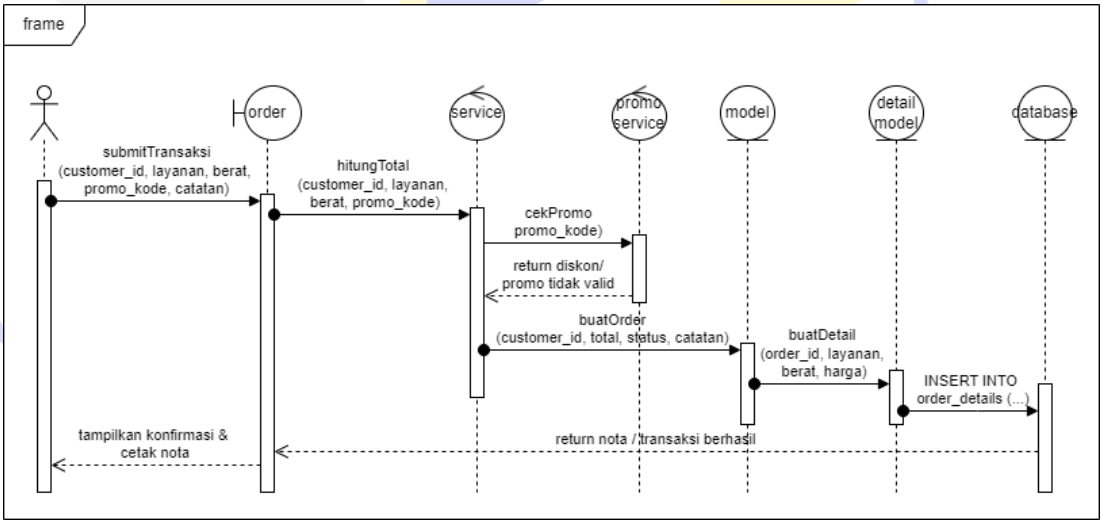
2) *Sequence Diagram* login



Sumber : Penelitian

Gambar IV. 11 Sequence Diagram Login

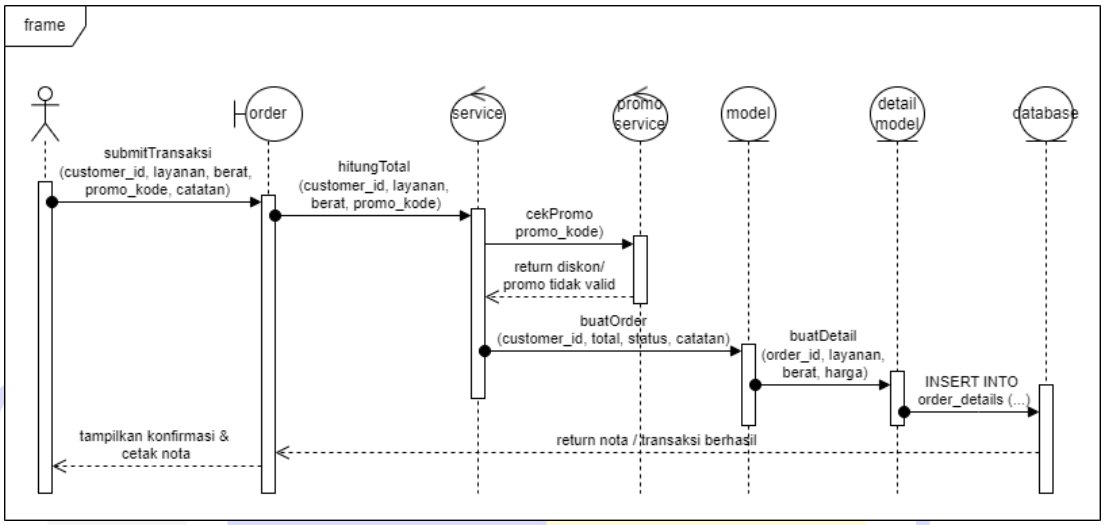
3) *Sequence Diagram* Tambah Transaksi



Sumber : Penelitian

Gambar IV. 12 Sequence Diagram Tambah Layanan

4) *Sequence Diagram* Selesaikan Transaksi

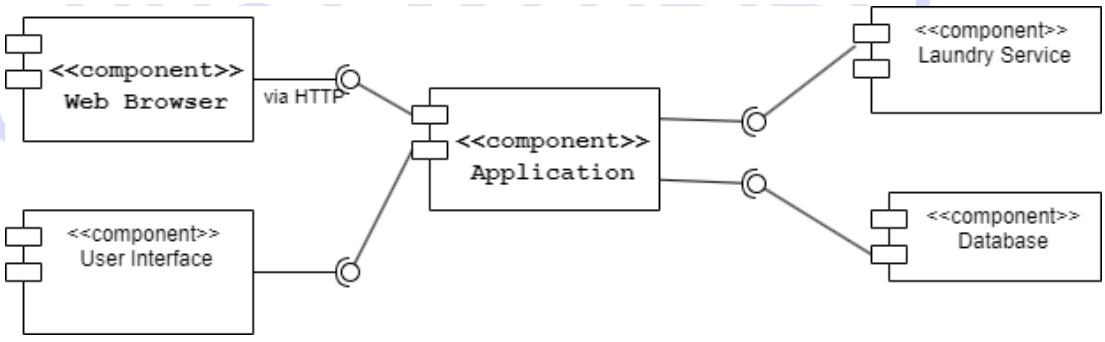


Sumber : Penelitian

Gambar IV. 13 Sequence Diagram Selesaikan Transaksi

5. *Component Diagram*

Component Diagram pada sistem ini menggambarkan bagaimana komponen-komponen perangkat lunak berinteraksi untuk membentuk aplikasi manajemen laundry. Setiap bagian seperti tampilan antarmuka (*Frontend*), pengendali logika aplikasi (*Controller*), model basis data (*Model*), dan basis data itu sendiri direpresentasikan sebagai komponen yang saling terhubung.

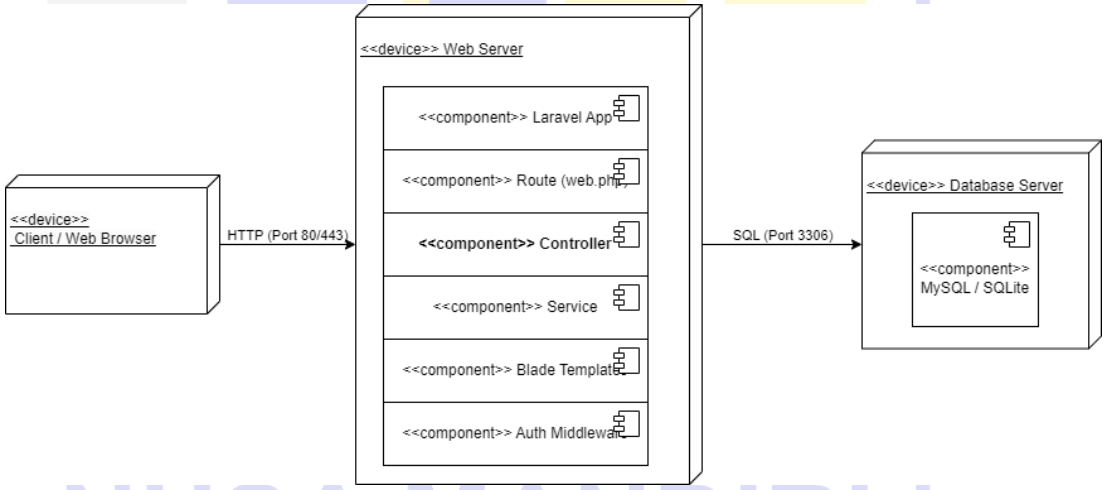


Sumber : Penelitian

Gambar IV. 14 Component Diagram

6. *Deployment Diagram*

Deployment Diagram menggambarkan konfigurasi fisik dari sistem manajemen laundry berbasis Laravel. Diagram ini terdiri dari beberapa node, seperti client browser, web server, dan database server. Web server menjalankan komponen Laravel seperti *controller*, model, dan view, sementara *database* server menangani penyimpanan data melalui MySQL atau SQLite. Komunikasi antar node dilakukan melalui HTTP dan koneksi basis data standar.



Sumber : Penelitian

Gambar IV. 15 Deployment Diagram

4.2.2 Desain Pemodelan Data

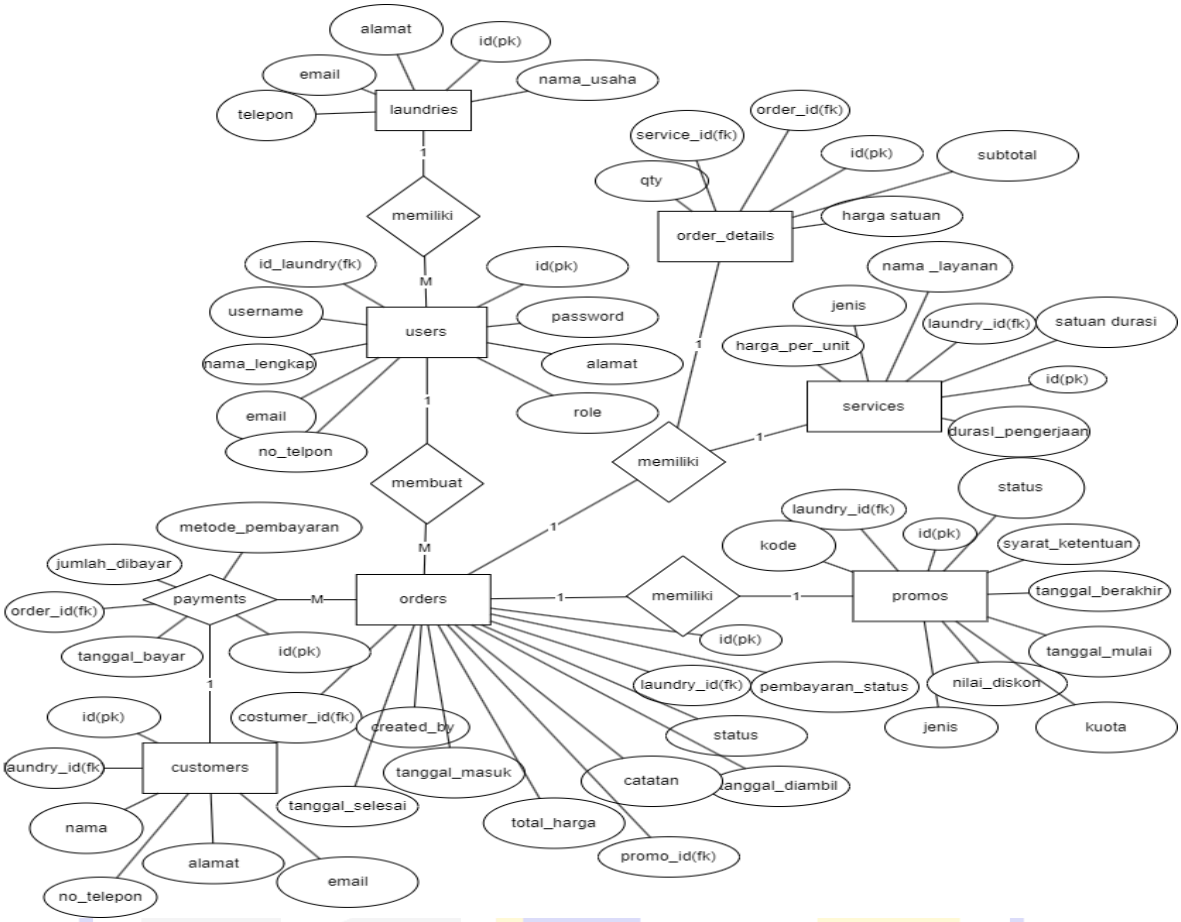
1. *Entity Relationship Diagram* (ERD)

Entity Relationship Diagram (ERD) digunakan sebagai alat bantu untuk memodelkan struktur data dan hubungan antar entitas dalam sistem manajemen laundry. Dengan ERD, hubungan antar tabel dalam basis data dapat

divisualisasikan dengan jelas dan sistematis, sehingga memudahkan proses pengembangan sistem, implementasi database, serta menjaga integritas data. Dalam sistem manajemen laundry ini, terdapat beberapa entitas utama, antara lain: *Users, Customers, Orders, Order_Details, Services, Payments, dan Promos*. Masing-masing entitas memiliki atribut kunci *(Primary Key)* yang bersifat unik, dan atribut penghubung *(Foreign Key)* yang merepresentasikan relasi antar entitas.

Dengan pendekatan ini, ERD tidak hanya menggambarkan struktur tabel dan relasi, namun juga menjadi acuan penting dalam proses normalisasi data dan pembuatan query SQL. ERD membantu memastikan bahwa basis data yang digunakan pada sistem ini berjalan secara optimal, efisien, dan bebas dari duplikasi data.

UNIVERSITAS
NUSA MANDIRI



Sumber : Penelitian

Gambar IV. 16 Entity Relationship Diagram

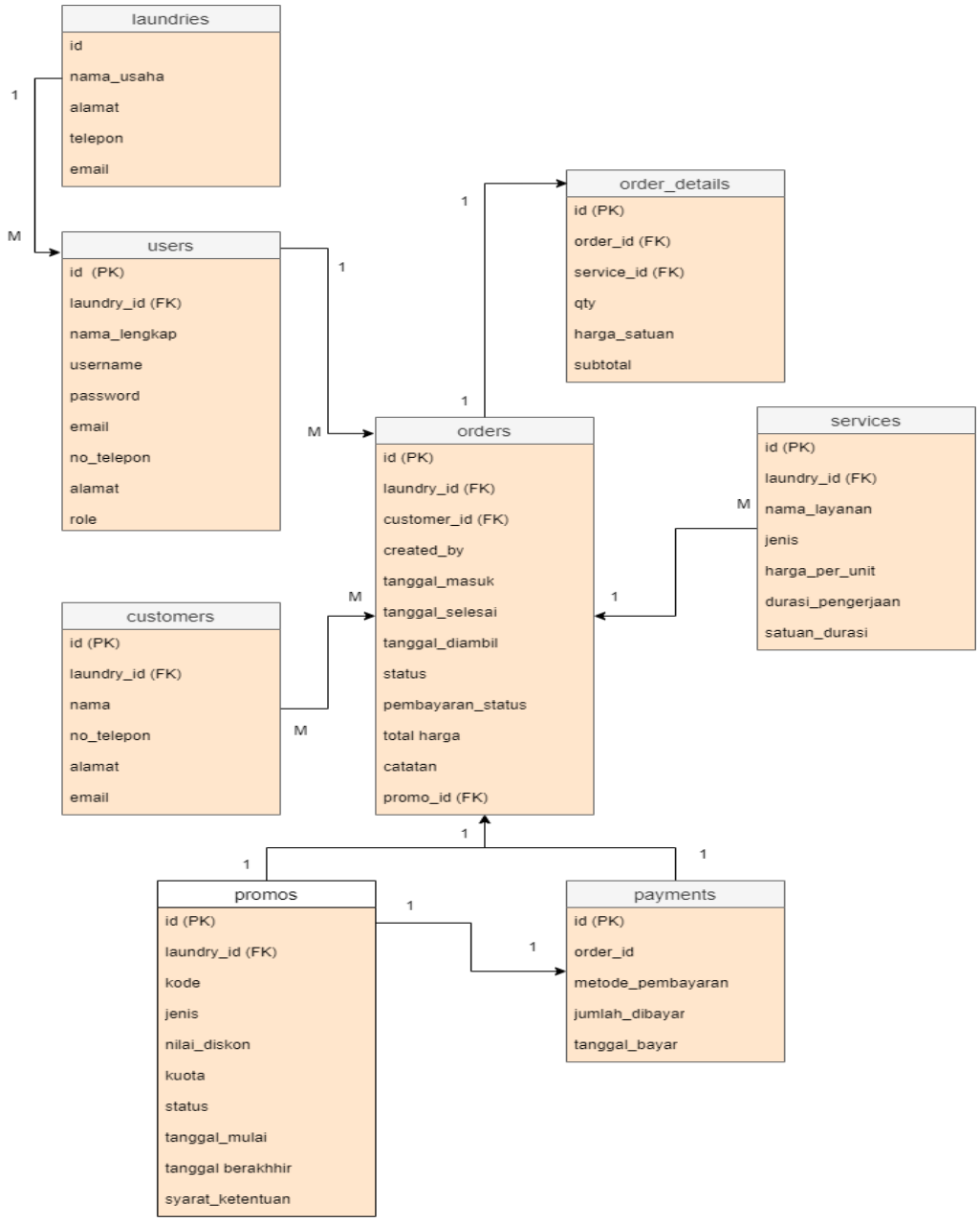
2. *Logical Record Structure*.

Logical Record Structure (LRS) merupakan bentuk representasi logis dari struktur data yang digunakan dalam pengembangan sistem. LRS menggambarkan masing-masing entitas yang terdapat dalam sistem dalam bentuk tabel lengkap dengan atribut, tipe data, ukuran *field*, dan jenis kunci (*key*) yang digunakan.

Struktur ini bertujuan untuk menjembatani antara model konseptual (*Entity Relationship Diagram*) dan model fisik dari *database* yang akan diimplementasikan. Dengan adanya LRS, perancang sistem dapat memastikan bahwa kebutuhan data dari setiap entitas telah dirancang secara jelas, konsisten, dan siap untuk diimplementasikan ke dalam sistem basis data.

UNIVERSITAS NUSA MANDIRI

Pada sistem informasi manajemen laundry yang dikembangkan, terdapat beberapa entitas utama seperti *users, laundries, customers, orders, order_details, services, promos, dan payments*. Setiap entitas memiliki sejumlah *field* yang mewakili data yang disimpan, dengan tipe data dan panjang *field* yang telah disesuaikan dengan kebutuhan sistem.



Sumber : Penelitian

Gambar IV. 17 Logical Record Structure

3. Spesifikasi File Database.

1) Spesifikasi File Tabel Laundry *(Laundries)*

Nama File : Tabel Laundries

Akronim : tblaundry.myd

Tipe File : File Master

Akses File : Random

Panjang Record : ±108 Byte

Kunci Field : id_laundry

Tabel IV. 8 Tabel Laundries

No	Nama Field	Tipe Data	Ukuran Field	Jenis Key
1	id_laundry	Char	8	Primary
2	nama_usaha	Varchar	50	
3	alamat	Text	-	
4	no_telp	Varchar	20	

Sumber : Penelitian

2) Spesifikasi File Tabel *User (Users)*

Nama File : Tabel *Users*

Akronim : tbusers.myd

Tipe File : File Master

Akses File : Random

Panjang Record : ±138 Byte

Kunci Field : id_user

UNIVERSITAS NUSA MANDIRI

Tabel IV. 9 Tabel Users

No	Nama Field	Tipe Data	Ukuran Field	Jenis Key
1	id_user	Char	8	Primary
2	id_laundry	Char	8	Foreign
3	nama	Varchar	50	
4	email	Varchar	100	Unique
5	password	Varchar	100	
6	role	Enum	('admin','kasir')	

Sumber : Penelitian

3) Spesifikasi File Tabel Pelanggan *(Customers)*

Nama File : Tabel *Customers*

Akronim : tbpelanggan.myd

Tipe File : File Master

Akses File : Random

Panjang Record : 198 Byte

Kunci Field : id_pelanggan

Tabel IV. 10 Tabel Customers

No	Nama Field	Tipe Data	Ukuran Field	Jenis Key
1	id_pelanggan	Char	8	Primary
2	nama	VarChar	50	
3	no_hp	VarChar	20	
4	alamat	Text	-	

Sumber : Penelitian

4) Spesifikasi File Tabel Transaksi *(Orders)*

Nama File : Tabel *Orders*

Akronim : tborder.myd

Tipe File : File Transaksi

Akses File : Random

Panjang Record : 178 Byte

Kunci Field : id_order

Tabel IV. 11 Tabel Orders

No	Nama Field	Tipe Data	Ukuran Field	Jenis Key
1	id_order	Char	8	Primary
2	id_pelanggan	Char	8	Foreign
3	tanggal_masuk	Date	-	
4	status	Varchar	20	
5	pembayaran	Varchar	20	
6	total	Decimal	-	

Sumber : Penelitian

5) Spesifikasi File Tabel Layanan *(Services)*

Nama File : Tabel *Services*

Akronim : tblayanan.myd

Tipe File : File Master

Akses File : Random

Panjang Record : 78 Byte

Kunci Field : id_layanan

UNIVERSITAS NUSA MANDIRI

Tabel IV. 12 Tabel Services

No	Nama Field	Tipe Data	Ukuran Field	Jenis Key
1	id_layanan	Char	8	Primary
2	nama_layanan	Varchar	50	
3	harga_per_kg	Decimal	-	

Sumber : Penelitian

6) Spesifikasi File Tabel Detail *Order (Order_Details)*

Nama File : Tabel *Detail_Order*

Akronim : tbdetailorder.myd

Tipe File : File Transaksi

Akses File : Random

Panjang Record : 88 Byte

Kunci Field : id_detail

Tabel IV. 13 Tabel Order_Details

No	Nama Field	Tipe Data	Ukuran Field	Jenis Key
1	id_detail	Char	8	Primary
2	id_order	Char	8	Foreign
3	id_layanan	Char	8	Foreign
4	berat	Float	-	
5	subtotal	Decimal	-	

Sumber : Penelitian

7) Spesifikasi File Tabel Pembayaran *(Payments)*

Nama File : Tabel Payments

UNIVERSITAS NUSA MANDIRI

Akronim : tbpembayaran.myd

Tipe File : File Transaksi

Akses File : Random

Panjang Record: 88 Byte

Kunci Field : id_pembayaran

Tabel IV. 14 Tabel Payments

No	Nama Field	Tipe Data	Ukuran Field	Jenis Key
1	id_pembayaran	Char	8	Primary
2	id_order	Char	8	Foreign
3	metode	Varchar	20	
4	jumlah_bayar	Decimal	-	
5	tanggal_bayar	Date	-	

Sumber : Penelitian

8) Spesifikasi File Tabel Promo *(Promos)*

Nama File : Tabel *Promos*

Akronim : tbpromo.myd

Tipe File : File Master

Akses File : Random

Panjang Record: 88 Byte

Kunci Field : id_promo

UNIVERSITAS NUSA MANDIRI

Tabel IV. 15 Tabel Promos

No	Nama Field	Tipe Data	Ukuran Field	Jenis Key
1	id_promo	Char	8	Primary
2	kode	Varchar	20	
3	diskon	Int	-	
4	deskripsi	Varchar	50	
5	kuota	Int	-	

Sumber : Penelitian

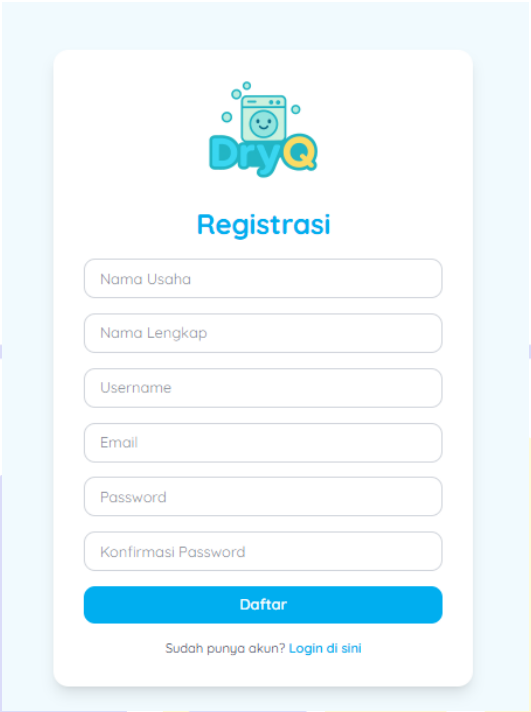
4.2.3 Desain *User Interface*

Desain *User Interface* (UI) merupakan bagian penting dalam pengembangan sistem karena secara langsung berhubungan dengan pengalaman pengguna *(user experience)*. Antarmuka pengguna dalam sistem manajemen laundry ini dirancang secara sederhana, intuitif, dan mudah digunakan baik oleh admin maupun kasir.

Desain UI dibuat secara nyata menggunakan *framework* Laravel dengan integrasi *Blade Template* dan didukung oleh TailwindCSS untuk tampilan yang responsif dan modern. Semua komponen antarmuka telah diuji untuk memastikan dapat dijalankan dengan baik di berbagai perangkat melalui browser.

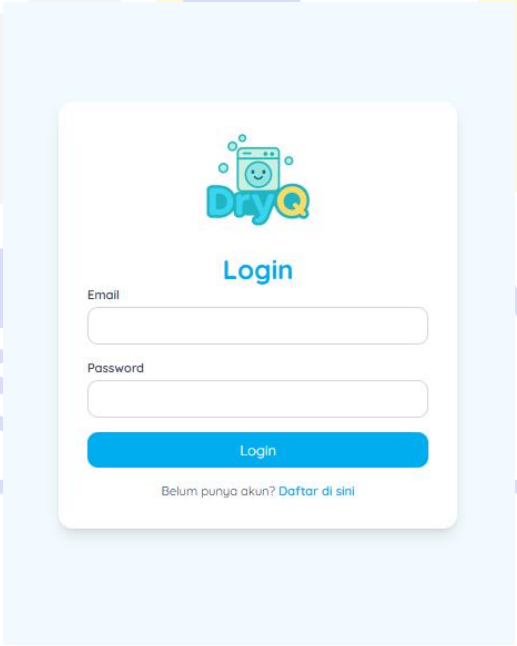
1. Tujuan Perancangan UI:
    1) Mempermudah pengguna dalam mengakses fitur utama sistem
    2) Menyediakan alur kerja yang efisien dan terarah
    3) Memberikan tampilan visual yang profesional dan ringan
2. Tampilan Desain *User Interface*

1) Desain Halaman Registrasi dan Login



Sumber : Penelitian

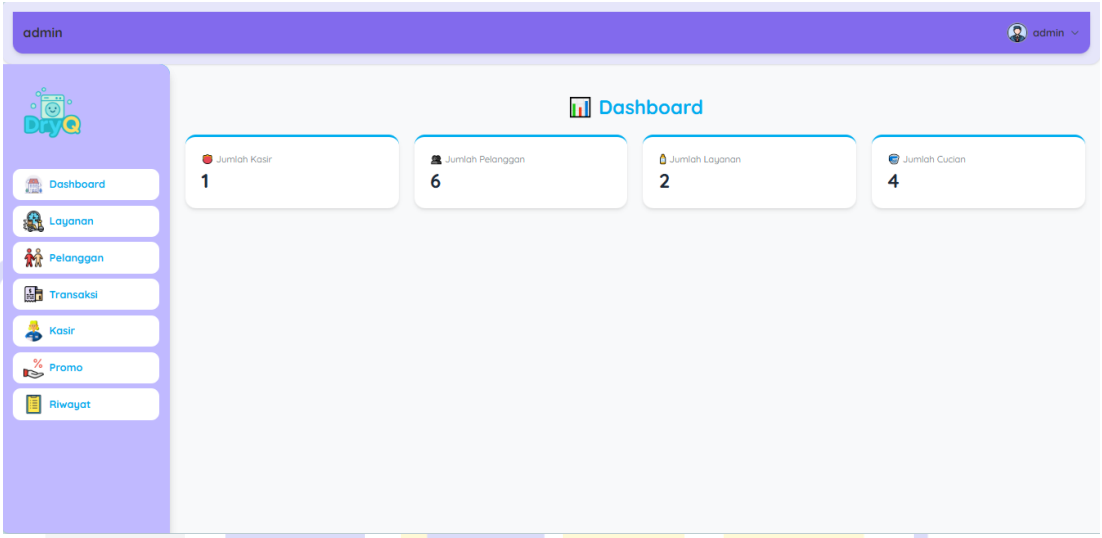
Gambar IV. 18 Halaman Registrasi



Sumber : Penelitian

Gambar IV. 19 Halaman Login

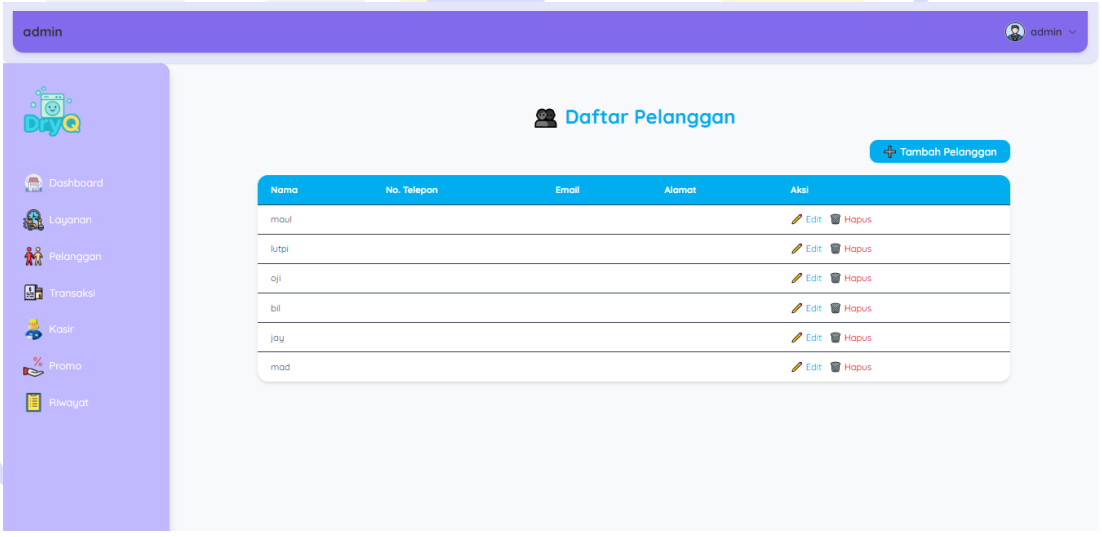
2) Desain Halaman *Dashboard*



Sumber : Penelitian

Gambar IV. 20 Halaman Dashboard

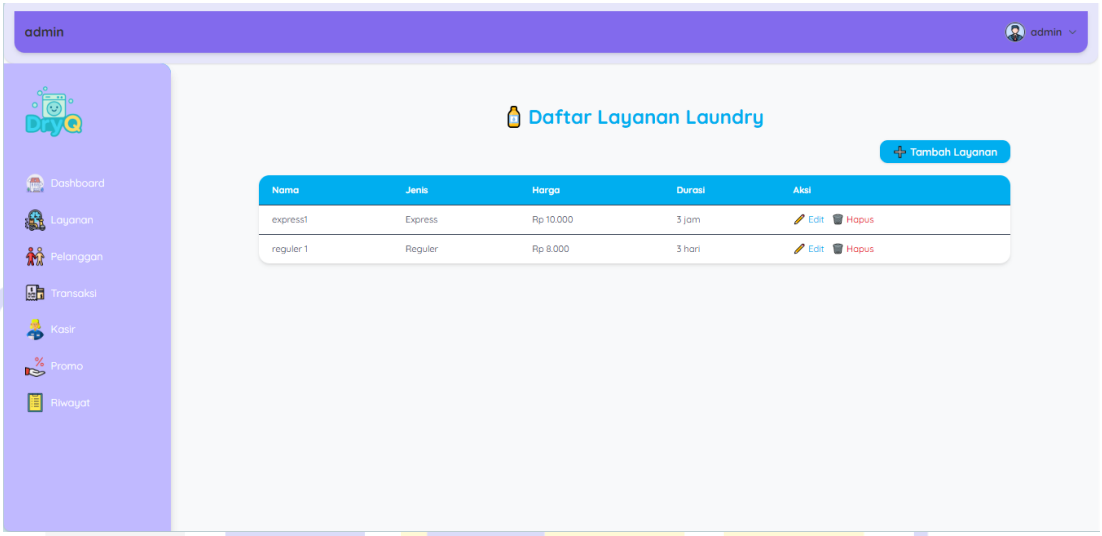
3) Desain Halaman Pelanggan



Sumber : Penelitian

Gambar IV. 21 Halaman Pelanggan

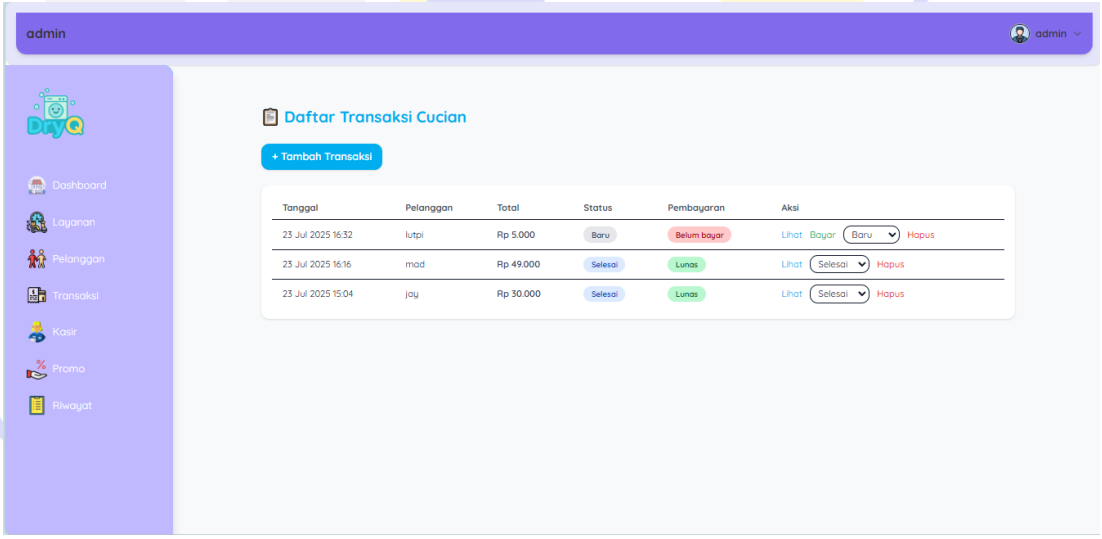
4) Desain Halaman Layanan



Sumber : Penelitian

Gambar IV. 22 Halaman Pelanggan

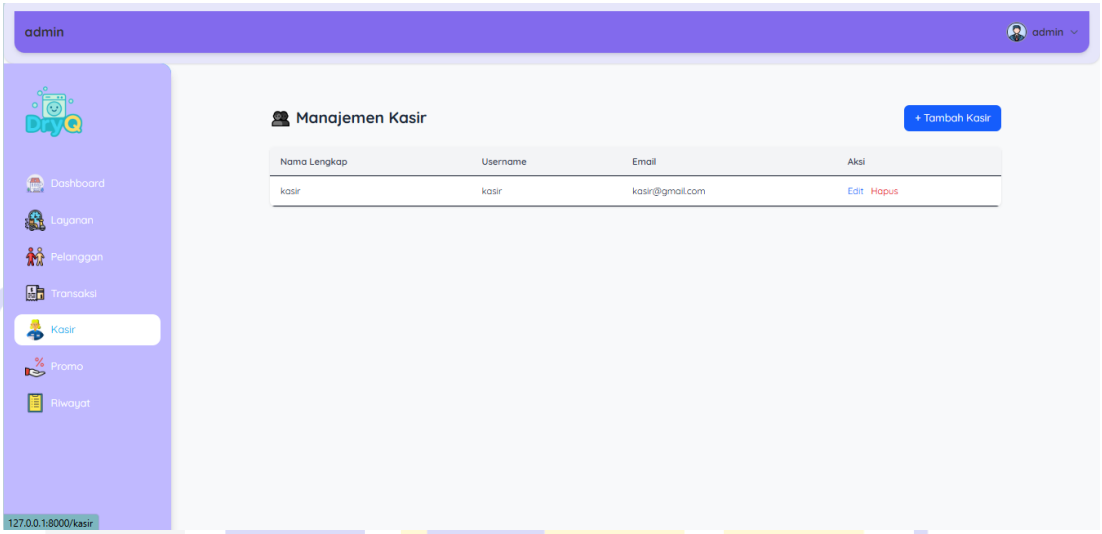
5) Desain Halaman Transaksi



Sumber : Penelitian

Gambar IV. 23 Halaman Transaksi

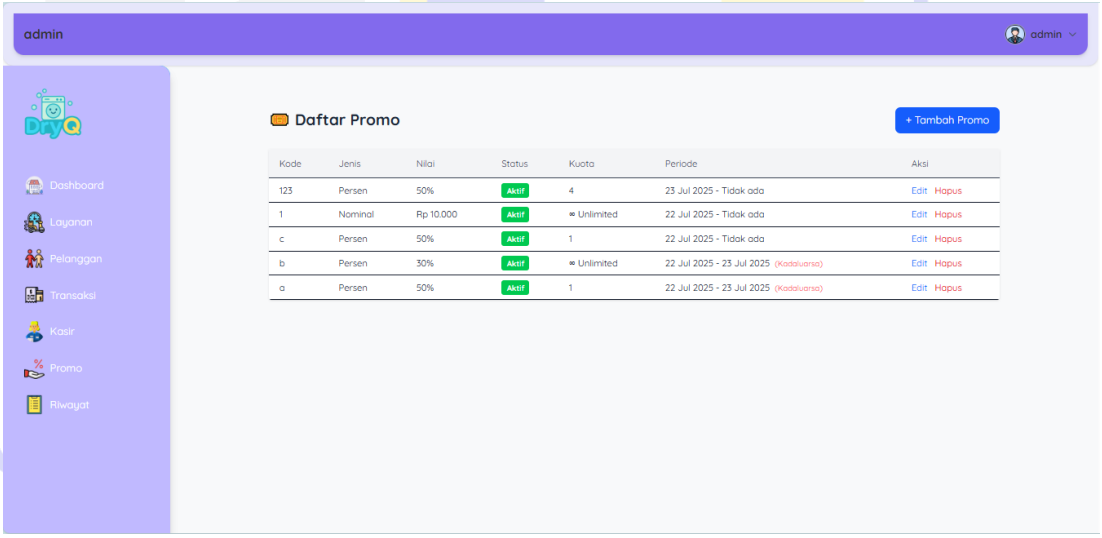
6) Desain Halaman Kasir



Sumber : Penelitian

Gambar IV. 24 Halaman Kasir

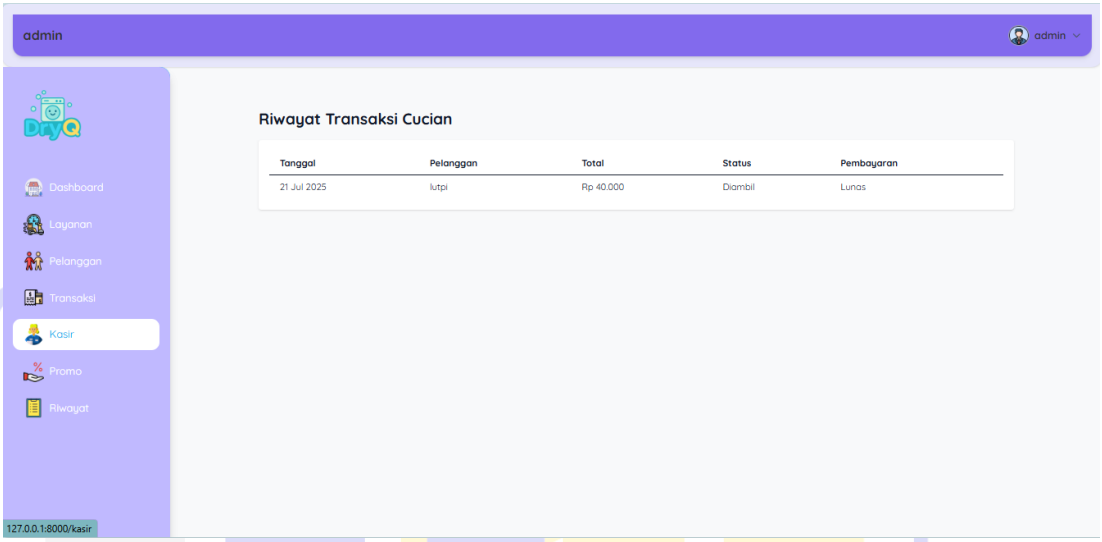
7) Desain Halaman Promo



Sumber : Penelitian

Gambar IV. 25 Halaman Promo

8) Desain Halaman Riwayat



Sumber : Penelitian

Gambar IV. 26 Halaman Riwayat

4.3 *Code Generation*

Tahap *code generation* merupakan proses penerapan desain sistem ke dalam bentuk kode program nyata menggunakan bahasa pemrograman dan *framework* tertentu. Dalam sistem manajemen laundry ini, pengembangan dilakukan menggunakan *framework* Laravel (PHP) yang bersifat berorientasi objek serta menerapkan konsep MVC *(Model-View-Controller)*.

Code generation bertujuan untuk merealisasikan fungsionalitas sistem sesuai dengan kebutuhan yang telah dirancang sebelumnya, seperti proses login, pendaftaran pelanggan, pengelolaan transaksi laundry, hingga pencatatan pembayaran.

Berikut ini adalah contoh beberapa potongan kode *(source code)* dari fungsi-fungsi penting dalam sistem:

```php
<?php

namespace App\Http\Controllers;
```

```
use App\Models\{Order, Customer, Service, OrderDetail, Promo,
Payment};
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Session;
use Illuminate\Support\Facades\DB;
use Carbon\Carbon;

class OrderController extends Controller
{
    public function index()
    {
        $orders = Order::where('laundry_id', Session::get('user')-
>laundry_id)
            ->where('status', '!=', 'diambil')
            ->with(['customer', 'payments'])
            ->latest()
            ->get();

        return view('orders.index', compact('orders'));
    }

    public function create()
    {
        $laundryId = Session::get('user')->laundry_id;
        $customers = Customer::where('laundry_id', $laundryId)-
>get();
        $services  = Service::where('laundry_id', $laundryId)->get();
        $promos    = Promo::where('laundry_id', $laundryId)
            ->where('status', 'aktif')
            ->whereDate('tanggal_mulai', '<=', now())
            ->whereDate('tanggal_berakhir', '>=', now())
            ->get();

        return view('orders.create', compact('customers', 'services',
'promos'));
    }

    public function store(Request $request)
    {
        $rules = [
            'items'             => 'required|array|min:1',
            'items.*.qty'       => 'required|numeric|min:0.1',
            'items.*.service_id' => 'required|exists:services,id',
            'kode_promo'        => 'nullable|string|max:255',
            'pembayaran_status' => 'nullable|in:belum bayar,lunas',
            'metode_bayar'      => 'nullable|in:cash,qris,transfer',
            'diskon'            => 'nullable|numeric|min:0',
            'total_setelah_diskon' => 'nullable|numeric|min:0',
```

```
        ];

        if ($request->pembayaran_status === 'lunas') {
            $rules['jumlah_dibayar'] = 'required|numeric|min:1';
        }
        if ($request->customer_combined === '__baru__' && !$request-
>filled('nama_pelanggan_baru')) {
    return back()->withErrors(['nama_pelanggan_baru' => 'Nama
pelanggan baru harus diisi'])->withInput();
}


        $request->validate($rules);

        $laundryId = Session::get('user')->laundry_id;
        $statusPembayaran = $request->pembayaran_status ?? 'belum
bayar';

        if ($request->filled('nama_pelanggan_baru')) {
            $existing = Customer::where('nama', $request-
>nama_pelanggan_baru)
                ->where('laundry_id', $laundryId)->first();
            $customerId = $existing
                ? $existing->id
                : Customer::create([
                    'nama' => $request->nama_pelanggan_baru,
                    'laundry_id' => $laundryId
                ])->id;
        } elseif ($request->filled('customer_combined') && $request-
>customer_combined !== '__baru__') {
            $customerId = $request->customer_combined;
        } else {
            return back()->withErrors(['customer_id' => 'Pelanggan
harus dipilih atau ditambahkan.']);
        }

        DB::beginTransaction();
        try {
            $order = Order::create([
                'laundry_id'        => $laundryId,
                'customer_id'       => $customerId,
                'created_by'        => Session::get('user')->id,
                'tanggal_masuk'     => now(),
                'status'            => 'baru',
                'pembayaran_status' => $statusPembayaran,
                'total_harga'       => 0,
            ]);
```

```
            $total = 0;
            foreach ($request->items as $item) {
                $service  = Service::findOrFail($item['service_id']);
                $qty      = (float) $item['qty'];
                $subtotal = $service->harga_per_unit * $qty;

                OrderDetail::create([
                    'order_id'     => $order->id,
                    'service_id'   => $service->id,
                    'qty'          => $qty,
                    'harga_satuan' => $service->harga_per_unit,
                    'subtotal'     => $subtotal,
                ]);

                $total += $subtotal;
            }

            $promoId = null;
            $diskon = 0;

            if ($request->filled('kode_promo') && $request-
>input('promo_validated') != 1) {
                return back()->withErrors(['kode_promo' => 'Silakan
tekan tombol Cek terlebih dahulu untuk validasi promo.']);
            }

            if ($request->filled('kode_promo')) {
                $promo = Promo::where('kode', $request->kode_promo)
                    ->where('laundry_id', $laundryId)
                    ->where('status', 'aktif')
                    ->whereDate('tanggal_mulai', '<=', now())
                    ->whereDate('tanggal_berakhir', '>=', now())
                    ->first();

                if ($promo) {
                    $promoDigunakan = Order::where('customer_id',
$customerId)
                        ->where('promo_id', $promo->id)
                        ->exists();

                    if ($promoDigunakan) {
                        return back()->withErrors(['kode_promo' =>
'Promo ini sudah pernah digunakan oleh pelanggan ini.']);
                    }

                    if ($promo->kuota !== null && $promo->kuota <= 0)
{
```

```
                        return back()->withErrors(['kode_promo' =>
'Kuota promo sudah habis.']);
                    }

                    $promoId = $promo->id;
                    $diskon = (float) $request->input('diskon', 0);
                }
            }

            $totalSetelahDiskon = $request-
>filled('total_setelah_diskon')
                ? (float) $request->total_setelah_diskon
                : round($total - $diskon, -3, PHP_ROUND_HALF_UP);

            $order->update([
                'total_harga'       => $totalSetelahDiskon,
                'promo_id'          => $promoId,
                'pembayaran_status' => $statusPembayaran,
            ]);

            if ($statusPembayaran === 'lunas') {
                if ($request->jumlah_dibayar < $totalSetelahDiskon) {
                    return back()->withErrors(['jumlah_dibayar' =>
'Jumlah dibayar kurang dari total.']);
                }
                $order->payments()->create([
                    'jumlah_dibayar'    => $request->jumlah_dibayar,
                    'tanggal_bayar'     => now(),
                    'metode_pembayaran' => $request->metode_bayar ??
'cash',
                ]);
            }

            if (isset($promo) && $promo->kuota !== null) {
                $promo->decrement('kuota');
            }

            DB::commit();
            return redirect()->route('orders.index')->with('success',
'Transaksi berhasil. Total: Rp ' . number_format($total, 0, ',', '.')
. ($diskon > 0 ? ', Diskon: Rp ' . number_format($diskon, 0, ',',
'.') . ', Total Bayar: Rp ' . number_format($totalSetelahDiskon, 0,
',', '.') : ''));
        } catch (\Exception $e) {
            DB::rollBack();
            return back()->withErrors(['error' => 'Terjadi kesalahan:
' . $e->getMessage()]);
        }
```

```
        logger()->debug('PROMO CHECK', [
    'kode_promo' => $request->kode_promo,
    'promo_validated' => $request->promo_validated,
    'diskon' => $diskon,
    'promo_id' => $promoId,
    'total_setelah_diskon' => $totalSetelahDiskon,
]);

    }

    public function edit(Order $order)
    {
        $this->authorizeOrder($order);

        if ($order->pembayaran_status === 'lunas') {
            return redirect()->route('orders.index')->with('info',
'Transaksi ini sudah dibayar.');
        }

        $order->load(['promo', 'payments', 'customer',
'orderDetails.service']);

        return view('orders.edit', compact('order'));
    }

    public function update(Request $request, Order $order)
    {
        $this->authorizeOrder($order);

        if ($order->pembayaran_status === 'lunas') {
            return back()->with('info', 'Transaksi ini sudah
dibayar.');
        }

        $request->validate([
            'metode_bayar' => 'required|in:cash,qris,transfer',
            'uang_dibayar' => 'required|numeric|min:0',
            'kode_promo'   => 'nullable|string|max:255',
        ]);

        DB::beginTransaction();

        try {
            $totalSebelumDiskon = $order->orderDetails-
>sum('subtotal');
            $promoId = null;
            $diskon = 0;
            $promo = null;
```

```
            // Cek apakah ada promo baru
            if ($request->filled('kode_promo')) {
                $promo = Promo::where('kode', $request->kode_promo)
                    ->where('laundry_id', $order->laundry_id)
                    ->where('status', 'aktif')
                    ->whereDate('tanggal_mulai', '<=', now())
                    ->where(function ($q) {
                        $q->whereNull('tanggal_berakhir')->orWhereDate('tanggal_berakhir', '>=', now());
                    })
                    ->first();

                if (!$promo) {
                    return back()->withErrors(['kode_promo' => 'Kode promo tidak valid.']);
                }

                // Promo hanya bisa dipakai 1x per customer, kecuali promo yang sedang dipakai
                $sudahPernahDigunakan = Order::where('customer_id', $order->customer_id)
                    ->where('promo_id', $promo->id)
                    ->where('id', '!=', $order->id)
                    ->exists();

                if ($sudahPernahDigunakan) {
                    return back()->withErrors(['kode_promo' => 'Promo ini sudah pernah digunakan pelanggan ini.']);
                }

                // Cek kuota promo
                if ($promo->kuota !== null && $promo->kuota <= 0) {
                    return back()->withErrors(['kode_promo' => 'Kuota promo sudah habis.']);
                }

                // Hitung diskon
                $diskon = $promo->jenis === 'persen'
                    ? ($promo->nilai_diskon / 100) * $totalSebelumDiskon
                    : $promo->nilai_diskon;

                $diskon = min($diskon, $totalSebelumDiskon);
                $promoId = $promo->id;
            }

            // Hitung total akhir
```

```
            $totalSetelahDiskon = round($totalSebelumDiskon -
$diskon, -3, PHP_ROUND_HALF_UP);

            // Validasi uang cukup
            if ($request->uang_dibayar < $totalSetelahDiskon) {
                return back()->withErrors(['uang_dibayar' => 'Jumlah
dibayar kurang dari total.']);
            }

            // Kembalikan kuota promo lama jika diganti
            if ($order->promo_id && $order->promo_id !== $promoId) {
                $promoLama = Promo::find($order->promo_id);
                if ($promoLama && $promoLama->kuota !== null) {
                    $promoLama->increment('kuota');
                }
            }

            // Simpan data
            $order->update([
                'total_harga'       => $totalSetelahDiskon,
                'pembayaran_status' => 'lunas',
                'promo_id'          => $promoId,
            ]);

            $order->payments()->create([
                'jumlah_dibayar'    => $request->uang_dibayar,
                'tanggal_bayar'     => now(),
                'metode_pembayaran' => $request->metode_bayar,
            ]);

            // Kurangi kuota promo baru jika digunakan
            if ($promo && $promo->kuota !== null) {
                $promo->decrement('kuota');
            }

            DB::commit();

            $kembalian = $request->uang_dibayar -
$totalSetelahDiskon;

            return redirect()->route('orders.index')->with('success',
'Pembayaran berhasil. Kembalian: Rp ' . number_format($kembalian, 0,
',', '.'));
        } catch (\Throwable $e) {
            DB::rollBack();
            return back()->withErrors(['error' => 'Terjadi kesalahan:
' . $e->getMessage()]);
        }
```

```
    }

    public function show(Order $order)
    {
        $this->authorizeOrder($order);
        $order->load(['customer', 'creator', 'orderDetails.service',
'payments', 'promo', 'laundry']);
        return view('orders.show', compact('order'));
    }

    public function updateStatus(Order $order, Request $request)
{
    $this->authorizeOrder($order);

    $request->validate(['status' =>
'required|in:baru,proses,selesai,diambil']);

    $urutan = ['baru', 'proses', 'selesai', 'diambil'];
    if (array_search($request->status, $urutan) <
array_search($order->status, $urutan)) {
        return back()->withErrors(['status' => 'Status tidak dapat
kembali ke sebelumnya.']);
    }

    $dataUpdate = ['status' => $request->status];

    if ($request->status === 'selesai' && $order->tanggal_selesai ===
null) {
        $dataUpdate['tanggal_selesai'] = now();
    }

    if ($request->status === 'diambil' && $order->tanggal_diambil ===
null) {
        $dataUpdate['tanggal_diambil'] = now();
    }

    $order->update($dataUpdate);
    return back()->with('success', 'Status transaksi diperbarui
menjadi ' . ucfirst($request->status));
}

    public function destroy(Order $order)
    {
        $this->authorizeOrder($order);
        $order->orderDetails()->delete();
        $order->delete();
        return redirect()->route('orders.index')->with('success',
'Transaksi berhasil dihapus.');
```

```
    }

    public function riwayat()
    {
        $orders = Order::where('laundry_id', Session::get('user')-
>laundry_id)
            ->where('status', 'diambil')
            ->with('customer')
            ->latest()->get();

        return view('riwayat.index', compact('orders'));
    }

    private function authorizeOrder(Order $order)
    {
        if ($order->laundry_id !== Session::get('user')->laundry_id)
{
            abort(403);
        }
    }
}
```

Penjelaasan setiap fungsi yang ada pada *OrderController*

1. Fungsi index()

   Berfungsi untuk menampilkan semua data transaksi laundry yang statusnya belum diambil oleh pelanggan. Data yang diambil difilter berdasarkan ID laundry user yang sedang login dan ditampilkan melalui view orders.index.

2. Fungsi *create*()

   Menyiapkan data yang dibutuhkan untuk menampilkan form input transaksi baru, termasuk daftar pelanggan, layanan, dan promo aktif. *View* yang dituju adalah *orders.create.*

3. Fungsi *store*(Request $request)

UNIVERSITAS NUSA MANDIRI

Merupakan fungsi inti untuk menyimpan transaksi laundry baru ke dalam database:

1) Melakukan validasi input (termasuk layanan, promo, dan pelanggan)
2) Menyimpan data pelanggan baru jika belum terdaftar
3) Menghitung total layanan dan diskon promo
4) Menyimpan transaksi cucian dan detail layanan (order & order_details)
5) Menyimpan data pembayaran jika statusnya "lunas"
6) Mengurangi kuota promo jika digunakan
7) Transaksi dilakukan menggunakan DB::beginTransaction() untuk memastikan integritas data.

4. Fungsi edit(Order $order) dan update(Request $request, Order $order)

Digunakan untuk mengedit transaksi yang belum dibayar dan melakukan pembayaran. Sistem akan menghitung ulang diskon jika pengguna mengubah kode promo dan memverifikasi bahwa jumlah uang yang dibayar cukup. Status transaksi akan diubah menjadi lunas dan pembayaran dicatat ke dalam tabel payments.

5. Fungsi *show*(Order $order)

Digunakan untuk menampilkan detail lengkap transaksi berdasarkan ID, termasuk data pelanggan, layanan, promo, metode pembayaran, dan total harga.

6. Fungsi updateStatus(Order $order, Request $request)

UNIVERSITAS NUSA MANDIRI

Digunakan untuk mengubah status cucian sesuai urutan: “baru”, “proses”, “selesai”, “diambil”. Sistem juga mencatat waktu selesai dan waktu pengambilan otomatis jika status berubah ke selesai atau diambil.

7. Fungsi *destroy*(Order $order)

   Digunakan untuk menghapus transaksi dan semua detail layanannya. Fungsi ini hanya bisa dijalankan oleh pemilik laundry terkait untuk menjaga keamanan data.

8. Fungsi riwayat()

   Digunakan untuk menampilkan riwayat transaksi yang sudah selesai dan diambil. Halaman ini berfungsi sebagai laporan histori bagi admin atau kasir.

9. Fungsi authorizeOrder(Order $order)

   Fungsi pelindung (*privat*) untuk memastikan bahwa setiap transaksi yang diakses memang milik laundry user yang login. Jika bukan, akses ditolak dengan *abort*(403).

4.4 Testing

Tahap pengujian dilakukan untuk mengetahui sejauh mana sistem aplikasi dapat berjalan sesuai dengan yang dirancang. Pengujian dalam sistem ini mencakup pengujian performa dan pengujian penerimaan pengguna, dengan tujuan memastikan bahwa aplikasi dapat bekerja secara optimal dan diterima dengan baik oleh pengguna akhir.

Pengujian dilakukan setelah seluruh fitur utama sistem selesai dikembangkan dan dapat diakses secara daring (online). Aplikasi diuji menggunakan perangkat lunak

UNIVERSITAS NUSA MANDIRI

pengujian performa berbasis web untuk menilai kualitas dari segi kecepatan, struktur, dan pengalaman pengguna.

4.4.1 Tahap Pengujian Aplikasi

1) Pengujian Performa

Pengujian performa bertujuan untuk mengukur seberapa cepat dan efisien aplikasi dapat diakses oleh pengguna melalui browser. Pada pengujian ini, penulis menggunakan GTmetrix yang mengadopsi Lighthouse Engine dari Google untuk menganalisis kinerja halaman website berdasarkan metrik-metrik Web Vitals.

Pengujian dilakukan pada alamat aplikasi http://dryq.najasaka.com/ menggunakan browser Chrome versi 125.0 dengan lokasi server pengujian di Vancouver, Kanada. Hasil pengujian menunjukkan bahwa aplikasi memiliki performa yang sangat baik.

Berikut adalah ringkasan hasil pengujian performa:

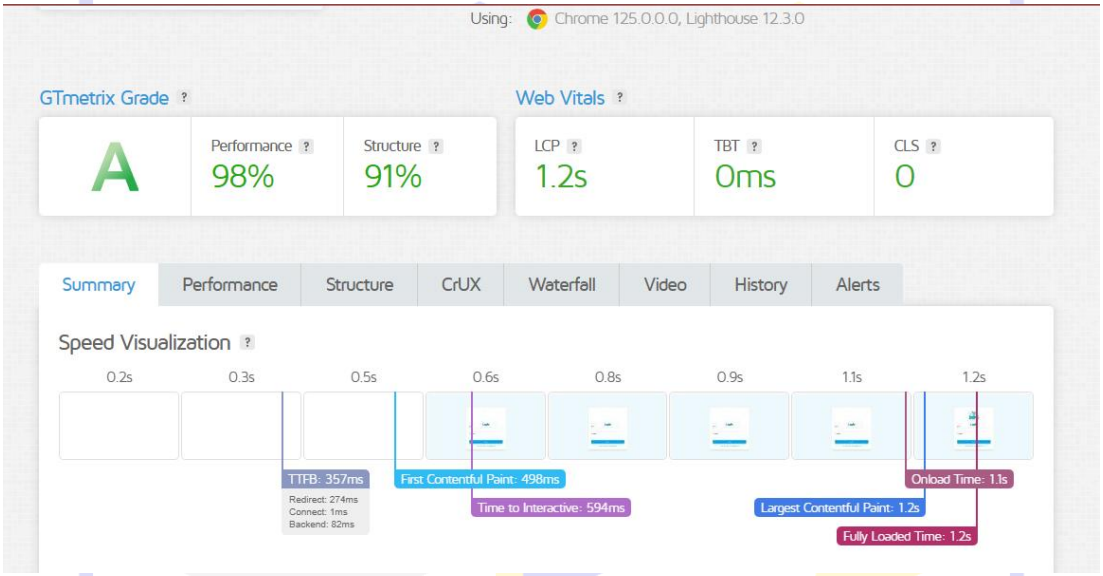
Tabel IV. 16 Tabel Peforma

Parameter	Nilai
GTmetrix Grade	A
Performance Score	98%
Structure Score	91%
Fully Loaded Time	1.2 s
Time to First Byte (TTFB)	357 ms
First Contentful Paint (FCP)	497 ms
Time to Interactive	593 ms

UNIVERSITAS NUSA MANDIRI

Largest Contentful Paint (LCP)	1.2 s
Cumulative Layout Shift (CLS)	0
Total Page Size	590 KB
Total Page Requests	10

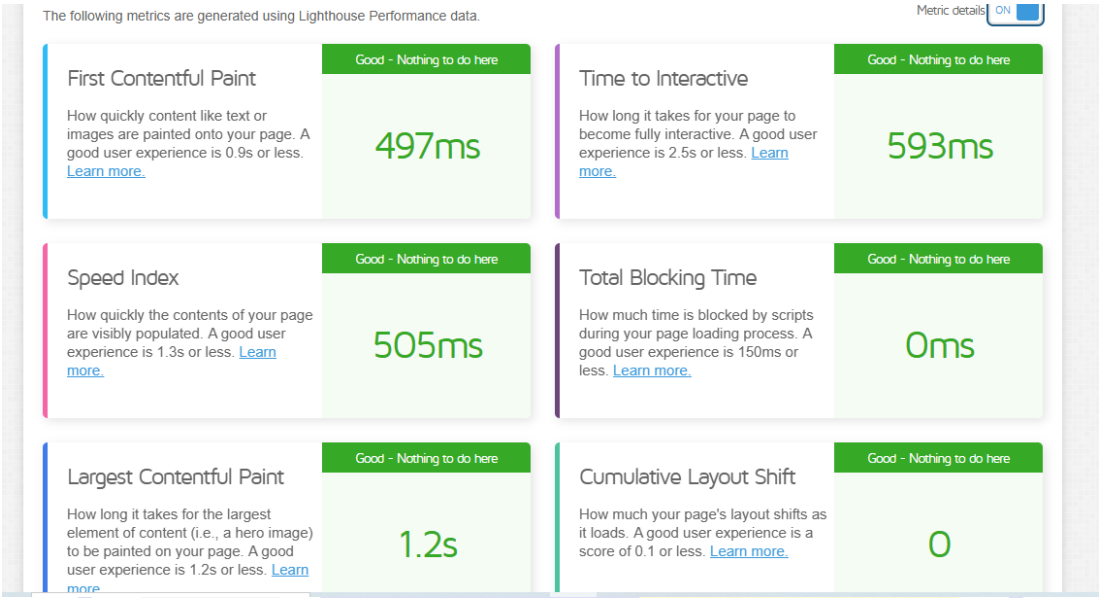
Sumber : Penelitian



Sumber : Penelitian

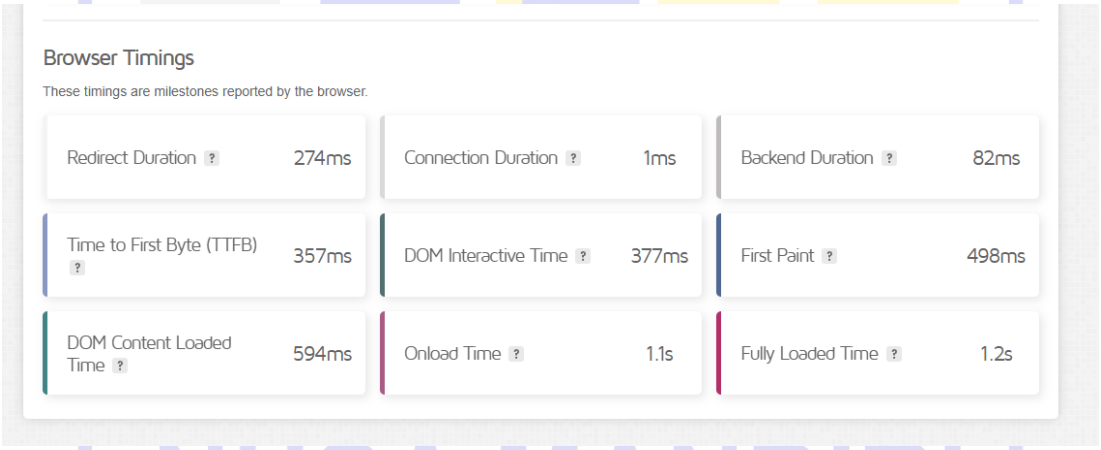
Gambar IV. 27 Hasil Peforma menggunakan GTmetrix

UNIVERSITAS
NUSA MANDIRI



Sumber : Penelitian

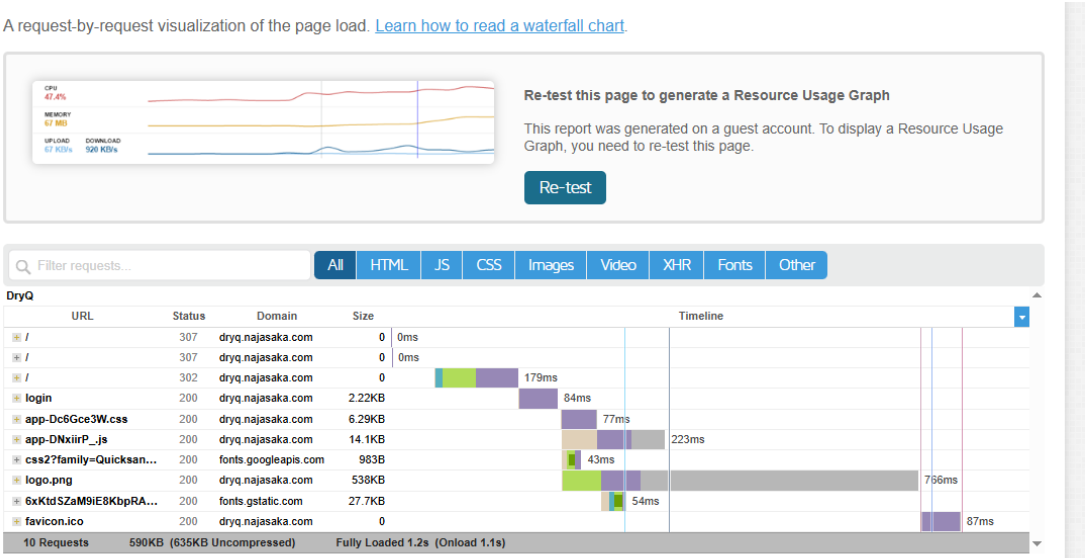
Gambar IV. 28 Hasil Peforma menggunakan GTmetrix



Sumber : Penelitian

Gambar IV. 29 Hasil Peforma menggunakan GTmetrix

NUSA MANDIRI



Sumber : Penelitian

Gambar IV. 30 Hasil Peforma

2) Pengujian Keamanan Website

Pengujian keamanan dilakukan untuk mengetahui potensi celah (*vulnerability*) yang terdapat pada sistem berbasis web. Pengujian ini penting untuk mencegah serangan atau penyalahgunaan oleh pihak tidak bertanggung jawab. Dalam tahap ini, alat yang digunakan adalah Acunetix Vulnerability Scanner, salah satu tools otomatis yang dapat mendeteksi berbagai jenis kerentanan umum pada aplikasi web.

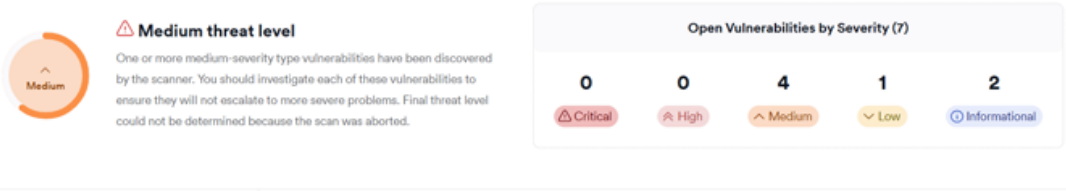
Hasil pengujian terhadap sistem http://dryq.najasaka.com/ menunjukkan tingkat ancaman pada kategori *Medium Threat* Level, dengan total 7 temuan kerentanan yang diklasifikasikan sebagai berikut:

Tabel IV. 17 Tabel Tingkat Kerentanan

Severity	**Jumlah**
Critical	0
High	0

Medium	**4**
Low	1
Informational	2

Sumber : Penelitian



Sumber : Penelitian

Gambar IV. 31 Hasil Pengujian Keamanan

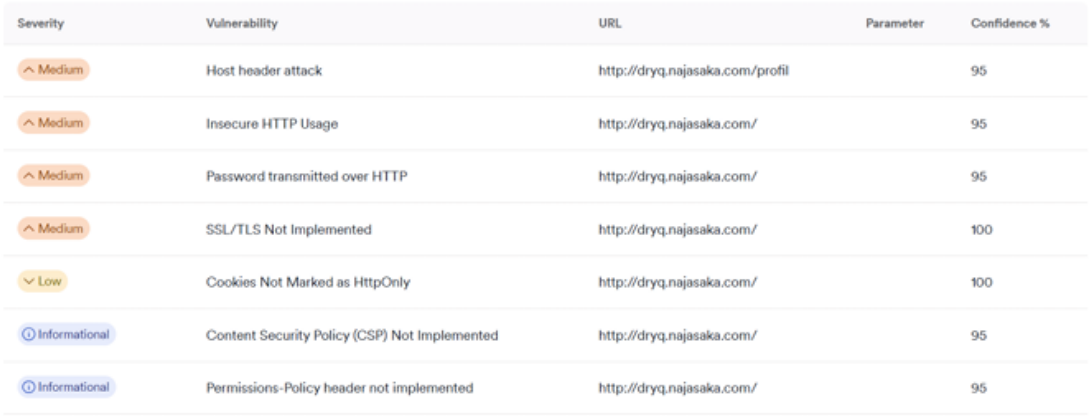
Berikut adalah daftar kerentanan yang terdeteksi:

Tabel IV. 18 Tabel Tingkat Kerentanan

Tingkat	Jenis Kerentanan	URL	Confidence
Medium	Host Header Attack	/profil	95%
Medium	Insecure HTTP Usage	/	95%
Medium	Password Transmitted Over HTTP	/	95%
Medium	SSL/TLS Not Implemented	/	100%
Low	Cookies Not Marked as HttpOnly	/	100%
Informational	Content Security Policy (CSP) Not Implemented	/	95%
Informational	Permissions-Policy Header Not Implemented	/	95%

Sumber : Penelitian

UNIVERSITAS NUSA MANDIRI



Severity	Vulnerability	URL	Parameter	Confidence %
Medium	Host header attack	http://dryq.najasaka.com/profil		95
Medium	Insecure HTTP Usage	http://dryq.najasaka.com/		95
Medium	Password transmitted over HTTP	http://dryq.najasaka.com/		95
Medium	SSL/TLS Not Implemented	http://dryq.najasaka.com/		100
Low	Cookies Not Marked as HttpOnly	http://dryq.najasaka.com/		100
Informational	Content Security Policy (CSP) Not Implemented	http://dryq.najasaka.com/		95
Informational	Permissions-Policy header not implemented	http://dryq.najasaka.com/		95

Sumber : Penelitian

Gambar IV. 32 Hasil Pengujian Keamanan

4.4.2 Tahap Pengujian Keterimaan Sistem

Pengujian keterimaan sistem atau *User Acceptance Testing* (UAT) dilakukan untuk mengetahui sejauh mana aplikasi yang dikembangkan dapat diterima dan digunakan dengan baik oleh pengguna akhir sesuai dengan kebutuhan operasional. UAT menjadi tahap penting sebelum sistem benar-benar digunakan secara penuh di lingkungan kerja atau operasional.

Tabel IV. 19 Tabel Pengujian Keterimaan Sistem

No	Use Case	Deskripsi Uji	Hasil Uji	Penguji	Tanggal	Catatan Penguji
1	Login Sistem	Mengakses sistem menggunakan akun admin dan kasir	Berhasil	Maulido	28 Juli 2025	Sistem membedakan akses berdasarkan role
2	Input Transaksi	Menambahkan pesanan cucian dari pelanggan baru/lama	Berhasil	Hendri	28 Juli 2025	Sistem menghitung total otomatis dan menyimpan
3	Gunakan Kode Promo	Menginput dan memvalidasi kode promo saat transaksi	Berhasil	Maulido	29 Juli 2025	Promo hanya dapat digunakan satu kali

4	Selesaikan Transaksi	Mengubah status pesanan menjadi "selesai" jika sudah dibayar lunas	Berhasil	Maulido dan Hendri	29 Juli 2025	Tombol “Selesai” hanya aktif jika sudah lunas
5	Cetak Nota	Mencetak atau mengunduh nota transaksi setelah input	Berhasil	Maulido dan Hendri	30 Juli 2025	Nota tampil dalam format yang dapat diunduh
6	Tambah Kasir	Admin menambahkan user kasir baru melalui menu pengguna	Berhasil	Maulido	30 Juli 2025	Akses kasir baru berhasil login

Sumber : Penelitian

Dikarenakan aplikasi ini dirancang untuk digunakan secara umum oleh siapa saja yang memiliki usaha laundry, bukan untuk perusahaan atau instansi tertentu, maka proses pengujian keterimaan sistem (*User Acceptance Testing*) dilakukan oleh pengguna acak yang mewakili calon pengguna akhir, seperti pemilik dan kasir laundry dari lingkungan sekitar

4.5 *Support*

Perangkat lunak yang telah dikembangkan dalam sistem manajemen laundry ini dirancang untuk digunakan secara umum oleh pelaku usaha laundry, tanpa terbatas pada satu mitra atau perusahaan tertentu. Sistem dapat diinstal dan dijalankan secara mandiri oleh pemilik usaha laundry melalui perangkat komputer atau server hosting. Perubahan atau dukungan pengembangan (*support*) kemungkinan besar akan dilakukan setelah perangkat lunak digunakan secara luas. Beberapa penyebab umum perubahan meliputi:

1. Adanya bug atau kesalahan logika yang baru ditemukan.
2. Perluasan fitur dan peningkatan performa.

UNIVERSITAS NUSA MANDIRI

3. Adaptasi sistem terhadap lingkungan baru, seperti perangkat lunak server yang diperbarui atau penggunaan pada perangkat mobile.
4. Permintaan pengguna untuk penambahan fitur tertentu.

Sistem ini dibangun menggunakan framework Laravel dan bersifat responsif serta dapat diakses melalui berbagai perangkat. Pemilik usaha laundry cukup memiliki akses ke server hosting dan koneksi internet untuk mengoperasikan sistem ini.

4.5.1 Publikasi Web

Sistem manajemen laundry ini didesain untuk dapat dipasang di layanan hosting sendiri *(self-hosted)* yang mendukung PHP dan MySQL/MariaDB. Aplikasi ini bersifat web-based sehingga dapat diakses dari mana saja melalui browser.

Dengan sistem ini, pemilik laundry dapat:

1) Mencatat transaksi cucian secara digital.
2) Mengelola pelanggan dan layanan.
3) Mengatur status cucian (proses, selesai, diambil).
4) Melihat dan mencetak laporan transaksi.
5) Mengatur pengguna (admin dan kasir).

UNIVERSITAS NUSA MANDIRI

4.5.2 Spesifikasi *Hardware* dan *Software*

1. Kebutuhan *Hardware Server*

Tabel IV. 20 Spesifikasi Hardware

Komponen	Spesifikasi Minimum
Disk Space	≥ 1 GB
Storage	SSD

Bandwidth	Unlimited
Sistem Operasi	Linux / CloudLinux
Protokol	HTTP/2 + QUIC Support

Sumber : Penelitian

Catatan: Sistem tidak membutuhkan perangkat keras khusus dan dapat berjalan di layanan shared hosting dengan spesifikasi dasar.

2. Kebutuhan *Software Server*

Tabel IV. 21 Spesifikasi Software

Komponen	Rincian
Framework	Laravel 12
Perintah Framework	Laravel Artisan (PHP Artisan)
Bahasa Pemrograman	PHP 8.1 atau lebih baru
Sistem Manajemen Database (DBMS)	MySQL / MariaDB
Perangkat Administrasi Database	phpMyAdmin

Sumber : Penelitian

4.6 Spesifikasi Dokumen Sistem Usulan

Dokumen sistem merupakan komponen penting dalam sistem informasi yang berfungsi sebagai bukti, referensi, dan arsip data transaksi maupun operasional sistem. Dalam sistem manajemen laundry ini, dokumen-dokumen yang terlibat meliputi bukti transaksi cucian, laporan riwayat, dan dokumen pelanggan.

UNIVERSITAS NUSA MANDIRI

Berikut adalah spesifikasi dari dokumen sistem yang dihasilkan dalam aplikasi:

Nama Dokumen : Nota Transaksi Cucian

Fungsi : Sebagai bukti transaksi cucian yang dilakukan pelanggan

Sumber : Kasir

Tujuan : Pelanggan

Media : Tampilan & Cetak (PDF atau Thermal Print)

Frekuensi : Setiap terjadi transaksi layanan laundry

Format : Lampiran B3

Nama Dokumen : Riwayat Transaksi cucian

Funsi : Sebagai bukti riwayat transaksi

Sumber : Admin

Tujuan : Usaha laundry

Media : Tampilan & Cetak (PDF atau Thermal Print)

Frekuensi : Harian, bulanan, atau sesuai tanggal yang dibutuhkan

Format : Lampiran B2 dan B2

UNIVERSITAS NUSA MANDIRI