**LAPORAN**

**“APLIKASI RESTOQ”**

****

**Oleh :**

**OKI JAYA GUNA : 22650161**

**RASLAN : 22650178**

**MUHAMMAD ASFAR DANI : 22650168**

**FAUZAN NURFAJAR I. : 22650152**

**ASRIATI AWADIA : 22650157**

**IIS PRATAMI : 22650173**

**RASLAN : 22650178**

**RAHMADIN HIDAYAT APSAHIH : 22650176**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK**

**UNIVERSITAS DAYANU IKHSANUDDIN**

**PASARWAJO**

**2023**

1. **ABSTRAK**

Aplikasi restoran ini dikembangkan untuk mempermudah pengelolaan berbagai aspek operasional dalam sebuah restoran, termasuk manajemen pesanan, pembayaran, inventaris, dan data pengguna. Aplikasi ini dirancang dengan menggunakan framework Django untuk backend dan HTML, CSS, serta JavaScript untuk frontend, yang memberikan antarmuka yang user-friendly dan responsif. Tujuan utama dari aplikasi ini adalah untuk meningkatkan efisiensi dan efektivitas dalam pengelolaan operasional restoran.

Proses pengembangan aplikasi ini melibatkan beberapa tahapan penting, mulai dari analisis kebutuhan, perancangan sistem, implementasi, hingga pengujian dan evaluasi. Dalam tahap perancangan, berbagai diagram seperti use case, class, dan activity digunakan untuk memvisualisasikan alur kerja dan struktur sistem. Implementasi dilakukan dengan mengikuti desain yang telah dibuat, dan pengujian dilakukan untuk memastikan bahwa setiap fungsi aplikasi berjalan sesuai dengan spesifikasi yang telah ditentukan.

Hasil pengujian menunjukkan bahwa aplikasi ini dapat berjalan dengan baik dan memenuhi kebutuhan operasional restoran. Aplikasi ini tidak hanya mempermudah staf restoran dalam mengelola pesanan dan pembayaran, tetapi juga membantu dalam pengelolaan inventaris dan pelaporan pendapatan. Dengan adanya aplikasi ini, diharapkan restoran dapat meningkatkan produktivitas dan memberikan pelayanan yang lebih baik kepada pelanggan. Laporan ini mendokumentasikan seluruh proses pengembangan aplikasi, memberikan wawasan yang mendalam tentang setiap tahap dan keputusan yang diambil selama pengembangan.

1. **PENDAHULUAN**

Manajemen operasional dalam sebuah restoran merupakan proses yang kompleks dan memerlukan efisiensi tinggi untuk memastikan kelancaran layanan serta kepuasan pelanggan. Proses ini mencakup pengelolaan pesanan, pembayaran, inventaris, dan koordinasi antara berbagai peran seperti admin, kasir, koki, pelayan, dan pelanggan. Penggunaan sistem manual seringkali menyebabkan kesalahan dan keterlambatan yang berdampak negatif pada pengalaman pelanggan dan operasional restoran secara keseluruhan.

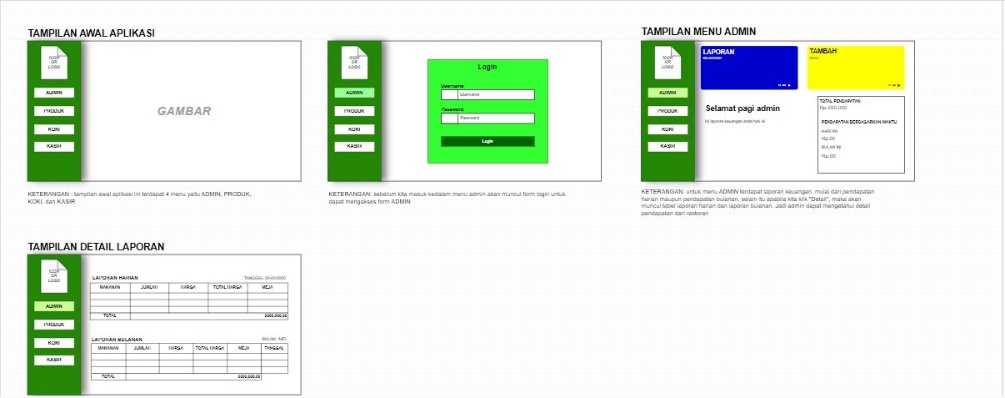
Dalam era digital ini, penggunaan teknologi informasi untuk mengotomatisasi dan mempermudah pengelolaan operasional restoran menjadi sangat penting. Aplikasi restoran yang dikembangkan dalam proyek ini bertujuan untuk menawarkan solusi komprehensif yang dapat meningkatkan efisiensi, mengurangi kesalahan, dan menyediakan data yang akurat dan real-time untuk pengambilan keputusan.

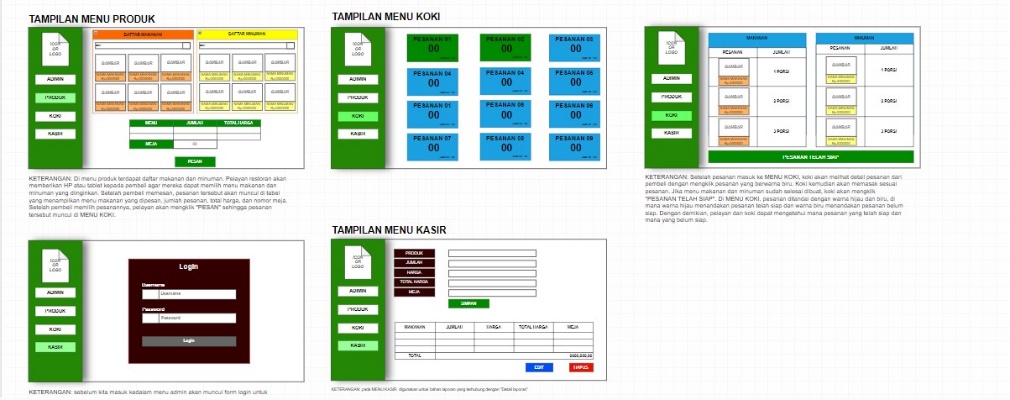
Laporan ini mendokumentasikan seluruh proses pengembangan aplikasi restoran, mulai dari analisis kebutuhan hingga tahap pengujian akhir. Pada bagian awal, laporan ini akan membahas perancangan sistem yang melibatkan berbagai diagram seperti use case, class, dan activity untuk memvisualisasikan alur kerja dan struktur sistem. Selanjutnya, implementasi dari desain yang telah dibuat akan dijelaskan secara rinci, termasuk teknologi dan framework yang digunakan.

Proses pengujian aplikasi juga dijelaskan untuk memastikan setiap fungsi berjalan sesuai dengan spesifikasi dan kebutuhan pengguna. Hasil dari pengujian ini akan memberikan gambaran tentang keandalan dan kinerja aplikasi dalam kondisi operasional sesungguhnya. Pada bagian akhir, laporan ini akan menyimpulkan temuan utama dari pengembangan dan pengujian aplikasi, serta memberikan rekomendasi untuk pengembangan lebih lanjut di masa depan.

Dengan adanya aplikasi restoran ini, diharapkan dapat membantu restoran dalam meningkatkan produktivitas, mengoptimalkan manajemen operasional, dan memberikan pelayanan yang lebih baik kepada pelanggan. Aplikasi ini tidak hanya memberikan manfaat bagi staf restoran tetapi juga meningkatkan pengalaman pelanggan secara keseluruhan.

1. **ANALISIS KEBUTUHAN**
2. **WIREFRAME**





Untuk lebih jelasnya silahkan kunjungi: <https://drive.google.com/file/d/1yeXaxOSpWiL9JisZzV0q4XoYo9-f-G6k/view?usp=sharing>

Wireframe adalah representasi visual dari struktur dan tata letak aplikasi yang membantu dalam merancang antarmuka pengguna (UI) sebelum tahap implementasi dimulai. Wireframe yang digunakan dalam pengembangan aplikasi restoran ini dirancang untuk memvisualisasikan elemen-elemen penting dari setiap halaman aplikasi, memastikan bahwa semua kebutuhan pengguna dan fungsionalitas yang diharapkan telah dipertimbangkan. Berikut adalah penjelasan detail tentang wireframe untuk aplikasi restoran:

1. Halaman Login

Deskripsi: Halaman login digunakan oleh pengguna (admin, kasir, koki, pelayan) untuk mengakses sistem dengan memasukkan kredensial mereka.

1. Elemen Utama:

* Field Username: Tempat untuk memasukkan username.
* Field Password: Tempat untuk memasukkan password.
* Button Login: Tombol untuk mengirimkan kredensial dan masuk ke sistem.
* Link Lupa Password: Opsi untuk pengguna yang lupa password.

1. Alur Kerja:

* Pengguna memasukkan username dan password, lalu menekan tombol login untuk masuk ke dashboard sesuai peran mereka.

1. Dashboard Admin

Deskripsi: Halaman utama setelah login untuk admin, yang berfungsi sebagai pusat navigasi untuk mengelola data dan laporan.

1. Elemen Utama:

* Menu Navigasi: Tautan ke halaman-halaman lain seperti kelola menu, kelola pesanan, kelola inventaris, dan laporan pendapatan.
* Ringkasan Data: Informasi ringkas mengenai jumlah pesanan, pendapatan harian, dan stok inventaris.
* Tabel Data: Menampilkan data-data terbaru atau penting yang perlu diperhatikan.

1. Alur Kerja:

* Admin dapat mengakses berbagai halaman pengelolaan dari menu navigasi dan melihat ringkasan data penting di dashboard.

1. Halaman Kelola Menu

Deskripsi: Halaman ini memungkinkan admin untuk menambah, mengubah, dan menghapus item menu.

1. Elemen Utama:

* Form Tambah/Ubah Menu: Formulir untuk memasukkan atau mengubah informasi item menu seperti nama, harga, dan deskripsi.
* Tabel Menu: Menampilkan daftar item menu yang sudah ada, dengan opsi untuk mengedit atau menghapus masing-masing item.
* Button Simpan: Tombol untuk menyimpan data item menu yang baru atau yang sudah diubah.

1. Alur Kerja:

* Admin dapat menambah item menu baru dengan mengisi formulir dan menekan tombol simpan, atau mengedit/hapus item menu yang sudah ada dari tabel.

1. Halaman Kelola Pesanan

Deskripsi: Halaman ini digunakan oleh kasir untuk mengelola pesanan pelanggan.

1. Elemen Utama:

* Form Tambah/Ubah Pesanan: Formulir untuk memasukkan atau mengubah informasi pesanan seperti item, jumlah, dan status.
* Tabel Pesanan: Menampilkan daftar pesanan yang sudah ada, dengan opsi untuk mengedit atau menghapus masing-masing pesanan.
* Button Simpan: Tombol untuk menyimpan data pesanan yang baru atau yang sudah diubah.

1. Alur Kerja:

* Kasir dapat menambah pesanan baru dengan mengisi formulir dan menekan tombol simpan, atau mengedit/hapus pesanan yang sudah ada dari tabel.

1. Halaman Kelola Inventaris

Deskripsi: Halaman ini memungkinkan admin untuk mengelola stok inventaris.

1. Elemen Utama:

* Form Tambah/Ubah Inventaris: Formulir untuk memasukkan atau mengubah informasi stok inventaris seperti item dan jumlah.
* Tabel Inventaris: Menampilkan daftar stok inventaris yang sudah ada, dengan opsi untuk mengedit atau menghapus masing-masing stok.
* Button Simpan: Tombol untuk menyimpan data stok inventaris yang baru atau yang sudah diubah.

1. Alur Kerja:

* Admin dapat menambah stok inventaris baru dengan mengisi formulir dan menekan tombol simpan, atau mengedit/hapus stok inventaris yang sudah ada dari tabel.

1. Halaman Pelanggan

Deskripsi: Halaman ini digunakan oleh pelanggan untuk melihat dan memesan menu yang tersedia.

1. Elemen Utama:

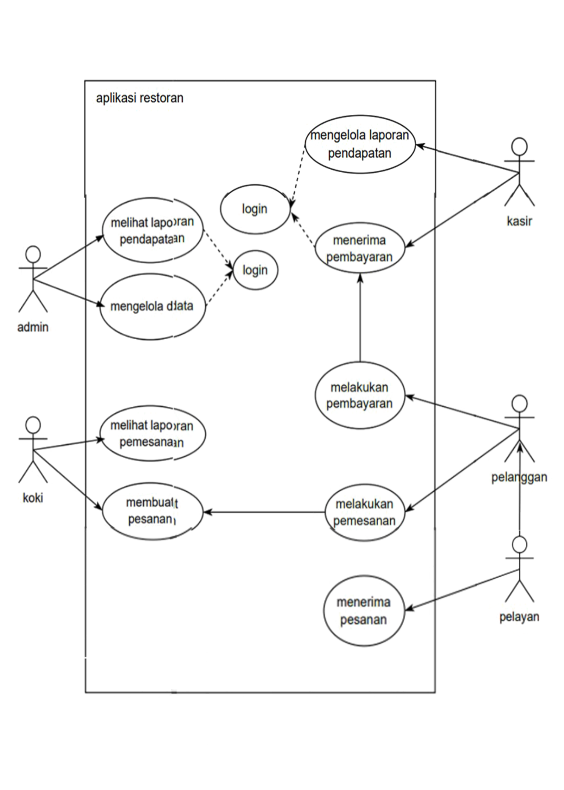
* Daftar Menu: Menampilkan semua item menu yang tersedia beserta harga dan deskripsinya.
* Form Pemesanan: Formulir untuk memilih item menu dan jumlah yang ingin dipesan.
* Button Pesan: Tombol untuk mengirimkan pesanan ke sistem.

1. Alur Kerja:

* Pelanggan dapat melihat daftar menu, memilih item yang diinginkan, mengisi jumlah, dan menekan tombol pesan untuk melakukan pemesanan.

Wireframe yang telah dirancang memberikan gambaran yang jelas tentang struktur dan elemen-elemen yang ada di setiap halaman aplikasi. Ini membantu dalam memastikan bahwa semua kebutuhan pengguna dan fungsionalitas yang diharapkan telah dipertimbangkan sebelum tahap pengembangan dimulai.

1. **USE CASE**

****

Use case diagram untuk aplikasi restoran ini menggambarkan interaksi antara pengguna (aktor) dengan sistem, serta fungsi-fungsi utama yang dapat dilakukan oleh masing-masing aktor. Berikut adalah deskripsi lengkap dari use case diagram:

1. Aktor dan Use Case
2. Admin

* Login: Admin dapat masuk ke dalam sistem dengan menggunakan kredensial mereka.
* Melihat Laporan Pendapatan: Admin dapat melihat laporan pendapatan harian, mingguan, atau bulanan restoran.
* Mengelola Data: Admin dapat mengelola data seperti menu, inventaris, dan pengguna lain.

1. Kasir

* Login: Kasir dapat masuk ke dalam sistem dengan menggunakan kredensial mereka.
* Mengelola Laporan Pendapatan: Kasir dapat mengelola dan melihat laporan pendapatan yang mereka catat.
* Menerima Pembayaran: Kasir dapat memproses dan mencatat pembayaran dari pelanggan.

1. Koki

* Melihat Laporan Pemesanan: Koki dapat melihat pesanan yang masuk untuk dipersiapkan.
* Membuat Pesanan: Koki dapat mencatat status persiapan pesanan.

1. Pelanggan

* Melakukan Pembayaran: Pelanggan dapat melakukan pembayaran untuk pesanan mereka.
* Melakukan Pemesanan: Pelanggan dapat melihat menu dan melakukan pemesanan.

1. Pelayan

* Menerima Pesanan (Pelanggan): Pelayan dapat menerima pesanan dari pelanggan dan memasukkannya ke dalam sistem.

1. Deskripsi Use Case
2. Login

* Deskripsi: Aktor (admin, kasir, koki, pelayan) masuk ke dalam sistem menggunakan kredensial mereka.
* Aktor: Admin, Kasir, Koki, Pelayan
* Alur Utama:
* Aktor memasukkan username dan password.
* Sistem memverifikasi kredensial.
* Jika valid, sistem mengarahkan aktor ke halaman dashboard sesuai peran mereka.
* Jika tidak valid, sistem menampilkan pesan kesalahan.

1. Melihat Laporan Pendapatan

* Deskripsi: Admin dapat melihat laporan pendapatan untuk berbagai periode.
* Aktor: Admin
* Alur Utama:
* Admin memilih periode laporan (harian, mingguan, bulanan).
* Sistem menampilkan laporan pendapatan untuk periode yang dipilih.

1. Mengelola Data

* Deskripsi: Admin dapat mengelola data menu, inventaris, dan pengguna lain.
* Aktor: Admin
* Alur Utama:
* Admin memilih jenis data yang ingin dikelola (menu, inventaris, pengguna).
* Admin dapat menambah, mengubah, atau menghapus data.
* Sistem menyimpan perubahan yang dilakukan oleh admin.

1. Mengelola Laporan Pendapatan

* Deskripsi: Kasir dapat mengelola dan melihat laporan pendapatan yang mereka catat.
* Aktor: Kasir
* Alur Utama:
* Kasir memilih laporan pendapatan yang ingin dilihat atau dikelola.
* Sistem menampilkan laporan pendapatan yang relevan.

1. Menerima Pembayaran

* Deskripsi: Kasir memproses pembayaran dari pelanggan.
* Aktor: Kasir
* Alur Utama:
* Kasir memasukkan detail pembayaran ke dalam sistem.
* Sistem mencatat pembayaran dan memperbarui status pesanan.

1. Melihat Laporan Pemesanan

* Deskripsi: Koki dapat melihat pesanan yang masuk untuk dipersiapkan.
* Aktor: Koki
* Alur Utama:
* Koki mengakses halaman laporan pemesanan.
* Sistem menampilkan daftar pesanan yang perlu dipersiapkan.

1. Membuat Pesanan

* Deskripsi: Koki dapat mencatat status persiapan pesanan.
* Aktor: Koki
* Alur Utama:
* Koki memilih pesanan yang sedang dipersiapkan.
* Koki mencatat status persiapan (misalnya, sedang dipersiapkan, selesai).
* Sistem memperbarui status pesanan.

1. Melakukan Pembayaran

* Deskripsi: Pelanggan melakukan pembayaran untuk pesanan mereka.
* Aktor: Pelanggan
* Alur Utama:
* Pelanggan memilih metode pembayaran.
* Pelanggan memasukkan detail pembayaran.
* Sistem memproses pembayaran dan memperbarui status pesanan.

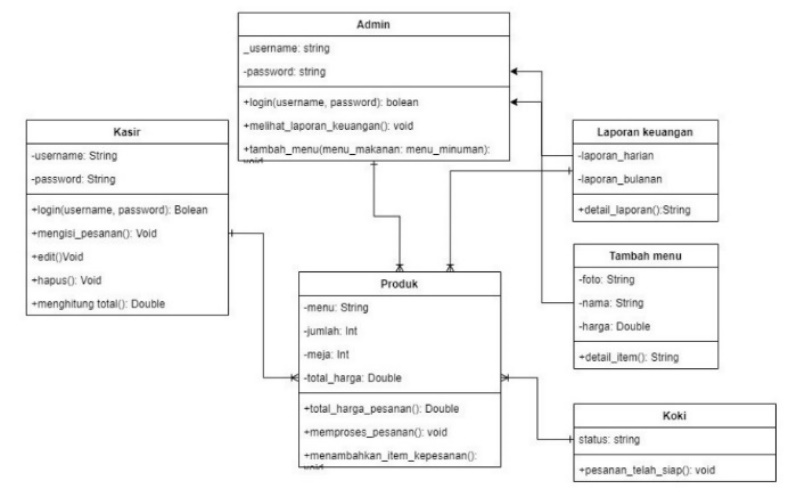
1. Melakukan Pemesanan

* Deskripsi: Pelanggan melihat menu dan melakukan pemesanan.
* Aktor: Pelanggan
* Alur Utama:
* Pelanggan melihat daftar menu.
* Pelanggan memilih item menu dan jumlah.
* Pelanggan mengirimkan pesanan ke sistem.
* Sistem mencatat pesanan dan memberikan konfirmasi.

1. Menerima Pesanan (Pelanggan)

* Deskripsi: Pelayan menerima pesanan dari pelanggan dan memasukkannya ke dalam sistem.
* Aktor: Pelayan
* Alur Utama:
* Pelayan mencatat pesanan dari pelanggan.
* Pelayan memasukkan detail pesanan ke dalam sistem.
* Sistem mencatat pesanan dan mengirimkannya ke dapur.

1. **PERANCANGAN SISTEM**
2. **CLASS DIAGRAM**



Class diagram untuk aplikasi restoran ini menggambarkan struktur kelas-kelas yang ada dalam sistem, termasuk atribut dan metode dari setiap kelas, serta hubungan di antara mereka. Berikut adalah deskripsi lengkap dari class diagram:

1. Kelas dan Relasinya
2. User

* Atribut:
* `username: String`
* `password: String`
* `role: String`
* Metode:
* `login()`
* `logout()`

1. Admin (extends User)

* Metode:
* `viewRevenueReport()`
* `manageData()`

1. Cashier (extends User)

* Metode:
* `manageRevenueReport()`
* `processPayment()`

1. Chef (extends User)

* Metode\*\*:
* `viewOrderReport()`
* `updateOrderStatus()`

1. Customer (extends User)\*\*

* Metode:
* `makePayment()`
* `placeOrder()`

1. Waiter (extends User)

* Metode:
* `takeOrder()`

1. Menu

* Atribut:
* `menuId: int`
* `name: String`
* `description: String`
* `price: double`
* Metode:
* `addMenuItem()`
* `updateMenuItem()`
* `deleteMenuItem()`
* `viewMenu()`

1. Order

* Atribut:
* `orderId: int`
* `orderDate: Date`
* `status: String`
* `totalAmount: double`
* Metode:
* `addOrder()`
* `updateOrder()`
* `viewOrder()`
* `deleteOrder()`

1. Inventory

* Atribut:
* `inventoryId: int`
* `itemName: String`
* `quantity: int`
* Metode:
* `addInventoryItem()`
* `updateInventoryItem()`
* `deleteInventoryItem()`
* `viewInventory()`

1. Payment

* Atribut\*\*:
* `paymentId: int`
* `paymentDate: Date`
* `amount: double`
* `paymentMethod: String`
* Metode:
* `processPayment()`
* `viewPayment()`

1. Relasi Antar Kelas
2. User - Admin, Cashier, Chef, Customer, Waiter:

* Relasi inheritance (pewarisan), di mana `Admin`, `Cashier`, `Chef`, `Customer`, dan `Waiter` mewarisi atribut dan metode dari `User`.

1. User - Order:

* Relasi association, di mana `User` dapat membuat `Order`. Relasi ini digambarkan dengan garis langsung antara `User` dan `Order`.

1. Order - Menu:

* Relasi association, di mana `Order` terdiri dari satu atau lebih `Menu` item. Relasi ini digambarkan dengan garis langsung antara `Order` dan `Menu`.

1. Order - Payment:

* Relasi association, di mana setiap `Order` memiliki satu `Payment`. Relasi ini digambarkan dengan garis langsung antara `Order` dan `Payment`.

1. Admin - Inventory:

* Relasi association, di mana `Admin` mengelola `Inventory`. Relasi ini digambarkan dengan garis langsung antara `Admin` dan `Inventory`.

1. Cashier - Payment:

* Relasi association, di mana `Cashier` memproses `Payment`. Relasi ini digambarkan dengan garis langsung antara `Cashier` dan `Payment`.

1. Chef - Order:

* Relasi association, di mana `Chef` mengelola status `Order`. Relasi ini digambarkan dengan garis langsung antara `Chef` dan `Order`.

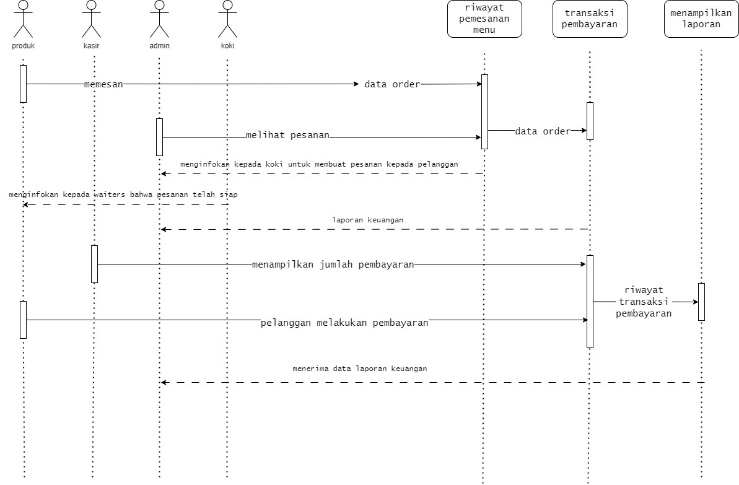
1. Waiter - Order:

* Relasi association, di mana `Waiter` mengambil `Order` dari `Customer`. Relasi ini digambarkan dengan garis langsung antara `Waiter` dan `Order`.

1. Visualisasi Class Diagram

Class diagram ini membantu dalam memahami struktur dan relasi antar kelas dalam sistem aplikasi restoran. Ini menjadi dasar untuk pengembangan kode program dan memastikan bahwa semua kebutuhan fungsional telah dipertimbangkan dalam desain sistem.

1. **DIAGRAM AKTIVITAS**



Sequence diagram di atas menggambarkan alur proses dari pemesanan hingga pembayaran dalam sistem aplikasi restoran. Berikut adalah penjelasan rinci dari diagram tersebut:

1. Pelanggan Memesan Produk:

* Pelanggan memesan produk (makanan/minuman) melalui aplikasi atau langsung kepada pelayan. Pesanan ini dicatat dalam sistem sebagai data order.

1. Data Order Dikirimkan ke Kasir:

* Data order diteruskan ke kasir, yang memiliki akses untuk melihat semua pesanan yang masuk. Kasir dapat memverifikasi dan mengonfirmasi pesanan tersebut.

1. Kasir Menginformasikan Koki:

* Setelah verifikasi, kasir mengirimkan informasi kepada koki untuk memulai proses pembuatan pesanan. Ini memastikan bahwa koki tahu apa yang harus disiapkan.

1. Koki Mempersiapkan Pesanan:

* Koki menyiapkan pesanan berdasarkan data order yang diterima. Setelah pesanan siap, koki menginformasikan kepada pelayan bahwa pesanan sudah dapat diantar kepada pelanggan.

1. Pelayan Menginformasikan Pelanggan:

* Pelayan menerima informasi dari koki bahwa pesanan sudah siap, dan kemudian menginformasikan kepada pelanggan atau langsung mengantarkan pesanan ke meja pelanggan.

1. Kasir Menampilkan Jumlah Pembayaran:

* Setelah pelanggan menikmati makanan/minuman, pelanggan akan melakukan pembayaran. Kasir menampilkan jumlah total pembayaran kepada pelanggan berdasarkan pesanan yang telah dibuat.

1. Pelanggan Melakukan Pembayaran:

* Pelanggan melakukan pembayaran kepada kasir. Kasir kemudian memproses pembayaran tersebut dan mencatatnya dalam sistem sebagai transaksi pembayaran.

1. Sistem Mengupdate Riwayat Transaksi:

* Sistem memperbarui riwayat transaksi pembayaran dengan informasi terbaru dari pembayaran yang dilakukan oleh pelanggan. Data ini kemudian digunakan untuk laporan keuangan.

1. Admin Mengakses Laporan Keuangan:

* Admin dapat mengakses laporan keuangan untuk melihat total pendapatan, transaksi, dan performa penjualan dalam periode tertentu. Laporan ini membantu dalam analisis dan pengambilan keputusan manajemen.

Penjelasan Tambahan:

* Lifeline: Garis vertikal yang merepresentasikan keberadaan aktor atau objek selama waktu tertentu.
* Message: Panah horizontal yang menunjukkan pesan yang dikirim dari satu objek ke objek lainnya. Pesan ini bisa berupa permintaan atau informasi.
* Activation Bar: Blok persegi panjang di lifeline yang menunjukkan periode ketika objek sedang melakukan operasi tertentu.
* Return Message: Panah garis putus-putus yang menunjukkan pengembalian informasi atau hasil dari pesan yang dikirim.

Diagram ini membantu dalam memahami alur kerja dari satu proses ke proses lainnya dalam aplikasi restoran, dan bagaimana setiap aktor (pelanggan, kasir, koki, admin) berinteraksi dengan sistem untuk menyelesaikan tugas mereka. Ini juga memperjelas bagaimana informasi mengalir antara aktor dan sistem, serta bagaimana data dikelola dan diproses pada setiap tahap.

1. **IMPLEMENTASI**
2. Database
3. Struktur Tabel:

* Produk: Menyimpan informasi menu (nama, deskripsi, harga, ketersediaan).
* Pesanan: Mencatat setiap pesanan (ID, waktu, status).
* Transaksi: Menyimpan pembayaran (ID, jumlah, metode).
* Pengguna: Data pengguna (kasir, koki, admin) dengan detail login dan peran.

1. Relasi:

* Tabel pesanan terkait dengan tabel produk.
* Tabel transaksi terkait dengan tabel pesanan.

1. Back-End
2. API Server: Menggunakan Node.js/Express atau PHP/Laravel.
3. Endpoints:

* login: Otentikasi pengguna.
* pesanan: Mengelola pesanan.
* transaksi: Mengelola pembayaran.
* produk: Mengelola menu.
* laporan: Akses laporan keuangan.

1. Front-End
2. Framework: React.js atau Vue.js.
3. Komponen:

* Login: Formulir untuk login pengguna.
* Dashboard Kasir: Melihat dan mengelola pesanan, proses pembayaran.
* Dashboard Koki: Melihat pesanan yang harus disiapkan.
* Dashboard Admin: Melihat laporan keuangan dan statistik.
* Halaman Pemesanan: Untuk pelanggan memesan menu.
* Komunikasi API: Menggunakan Axios atau Fetch API.

1. Keamanan
2. Otentikasi: Menggunakan JWT untuk keamanan akses.
3. Enkripsi Data: Menggunakan HTTPS untuk semua komunikasi data.
4. Pengujian
5. Unit Testing: Menguji fungsi individual.
6. Integration Testing: Menguji interaksi antar komponen.
7. User Acceptance Testing (UAT): Menguji sistem oleh pengguna akhir (kasir, koki, admin).
8. **PENGUJIAN**

Pengujian aplikasi restoran dilakukan untuk memastikan bahwa semua fungsi bekerja sesuai dengan yang diharapkan dan aplikasi bebas dari bug. Berikut adalah beberapa jenis pengujian yang dilakukan:

1. Unit Testing

Unit testing menguji fungsi atau unit terkecil dari aplikasi secara individual untuk memastikan setiap bagian berfungsi dengan benar.

* Alat yang Digunakan: Jest (untuk JavaScript/React) atau PHPUnit (untuk PHP/Laravel).
* Contoh: Menguji fungsi login, validasi pesanan, dan perhitungan total pembayaran.

1. Integration Testing

Integration testing menguji bagaimana berbagai komponen atau modul aplikasi berinteraksi satu sama lain. Pengujian ini memastikan bahwa integrasi antar modul berjalan lancar.

* Alat yang Digunakan: Postman untuk API testing, Selenium untuk mengotomatisasi pengujian front-end.
* Contoh: Menguji alur lengkap dari pemesanan menu oleh pelanggan hingga pembayaran dan pencatatan dalam laporan keuangan.

1. User Acceptance Testing (UAT)

User Acceptance Testing dilakukan oleh pengguna akhir (kasir, koki, admin) untuk memastikan bahwa aplikasi memenuhi kebutuhan dan ekspektasi mereka. Pengujian ini menilai kegunaan dan fungsionalitas dari perspektif pengguna.

Langkah-langkah:

* Pengguna mencoba melakukan berbagai tugas seperti login, membuat pesanan, memproses pembayaran, dan melihat laporan.
* Mengumpulkan umpan balik dari pengguna tentang kemudahan penggunaan dan fungsionalitas aplikasi.

1. Performance Testing

Performance testing dilakukan untuk memastikan aplikasi dapat menangani beban pengguna dan pesanan yang tinggi tanpa kinerja yang menurun.

* Alat yang Digunakan: JMeter atau LoadRunner.
* Contoh: Menguji respons server ketika ada banyak pesanan masuk secara bersamaan.

1. Security Testing

Security testing memastikan bahwa aplikasi terlindungi dari berbagai ancaman keamanan dan data pengguna aman.

* Alat yang Digunakan: OWASP ZAP atau Burp Suite.
* Contoh: Menguji kerentanan terhadap serangan SQL injection, cross-site scripting (XSS), dan man-in-the-middle (MITM).

Hasil Pengujian

Setiap jenis pengujian menghasilkan laporan yang mendokumentasikan temuan, bug, dan perbaikan yang dilakukan. Ini termasuk:

* Laporan Unit Testing: Menyatakan persentase fungsi yang lulus uji.
* Laporan Integration Testing: Menyertakan hasil pengujian integrasi antar modul.
* Laporan UAT: Berisi umpan balik dari pengguna dan perubahan yang diusulkan berdasarkan umpan balik tersebut.
* Laporan Performance Testing: Menyertakan kinerja aplikasi di bawah beban tinggi.
* Laporan Security Testing: Mendokumentasikan kerentanan yang ditemukan dan langkah-langkah mitigasi yang diambil.

Pengujian yang komprehensif ini memastikan aplikasi restoran berjalan dengan baik, aman, dan memenuhi kebutuhan pengguna akhir.

1. **KESIMPULAN**

Aplikasi restoran yang dikembangkan telah melalui berbagai tahap pengembangan dan pengujian untuk memastikan fungsionalitas, kinerja, dan keamanan yang optimal. Melalui proses desain yang komprehensif, termasuk pembuatan use case diagram, class diagram, dan sequence diagram, aplikasi ini mampu memenuhi kebutuhan operasional restoran dengan melibatkan berbagai aktor seperti kasir, koki, admin, dan pelanggan. Setiap modul dan fitur diuji secara menyeluruh melalui unit testing, integration testing, user acceptance testing, performance testing, dan security testing. Hasil pengujian menunjukkan bahwa aplikasi ini berfungsi dengan baik, mampu menangani beban tinggi, dan memiliki tingkat keamanan yang memadai.

Dengan adanya aplikasi ini, proses pemesanan, pengelolaan inventaris, dan pencatatan transaksi menjadi lebih efisien dan terstruktur. Pengguna akhir, termasuk kasir dan koki, memberikan umpan balik positif mengenai kemudahan penggunaan dan fungsionalitas aplikasi. Secara keseluruhan, aplikasi restoran ini diharapkan dapat meningkatkan produktivitas dan kepuasan pelanggan, serta mendukung pengelolaan restoran yang lebih baik. Implementasi dan pengujian yang telah dilakukan menunjukkan bahwa aplikasi ini siap digunakan dalam lingkungan operasional yang sebenarnya.