Sistem Pemesanan dan Pembayaran Mandiri di Restoran: Pendekatan Berorientasi Objek dengan Java

ARIA FIKRI MAHADINATA

AHMAD ZIDAN TAMIMY



**PENERBIT CV.EUREKA MEDIA AKSARA**

Sistem Pemesanan dan Pembayaran Mandiri di Restoran: Pendekatan Berorientasi Objek dengan Java

**Penulis :** Aria Fikri Mahadinata  
 Ahmad Zidan Tamimy

**Editor :**

**Desain Sampul :**

**Tata Letak :**

**ISBN :**

Diterbitkan oleh : **EUREKA MEDIA AKSARA, JULI 2022  
 ANGGOTA IKAPI JAWA TENGAH  
 NO. 255/JTE/2021**

**Redaksi :**

Jalan Banjaran, Desa Banjaran RT 20 RW 10 Kecamatan Bojongsari Kabupaten Purbalingga Telp. 0858-5343-1992

Surel : eurekamediaaksara@gmail.com

Cetakan Pertama : 2022

**All right reserved**

Hak Cipta dilindungi undang-undang

Dilarang memperbanyak atau memindahkan sebagian atau seluruh isi buku ini dalam bentuk apa pun dan dengan cara apa pun, termasuk memfotokopi, merekam, atau dengan teknik perekaman lainnya tanpa seizin tertulis dari penerbit.

KATA PENGANTAR

Penulis berterima kasih kepada Allah SWT karena berkat dan rahmat-Nya, penulis dapat menyelesaikan tugas ini dengan sukses. Program "Sistem Pemesanan dan Pembayaran Mandiri di Restoran Berbasis Java" adalah hasil dari penelitian dan upaya yang dilakukan. Penulis sadar bahwa menyelesaikan proyek ini akan sangat sulit jika tidak ada bantuan, dukungan, dan bimbingan dari berbagai pihak. Oleh karena itu, penulis mengucapkan terima kasih kepada Ketua STTI NIIT, Ketua Program Studi, Dosen Pembimbing, dan semua orang yang telah membantu menyusun laporan ini dan memberikannya kepada pembaca.

Sektor layanan makanan adalah salah satu dari banyak bidang kehidupan yang sangat dipengaruhi oleh perkembangan teknologi informasi yang pesat. "Program Tiket Mesin Restoran" dibuat untuk memudahkan pemesanan dan pembayaran di restoran. Melalui sistem ini, pelanggan dapat memesan makanan dan minuman secara mandiri melalui mesin tiket otomatis. Sistem ini meningkatkan efisiensi layanan dan memberikan fleksibilitas dan kenyamanan bagi pelanggan dan pihak restoran.

Saat ini, layanan mandiri semakin diminati oleh masyarakat yang menginginkan kemudahan dan kecepatan dalam berbagai kegiatan sehari-hari. Dengan adanya sistem pemesanan dan pembayaran mandiri ini, diharapkan interaksi antara pelanggan dan restoran dapat berlangsung lebih efisien, diakses dari berbagai tempat, serta mengurangi kesalahan manual yang sering terjadi dalam sistem pemesanan tradisional. Program ini dirancang untuk memberikan solusi yang lebih praktis dan modern dalam memfasilitasi transaksi di restoran.

Penulis menyadari bahwa laporan ini tidak sempurna. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang bermanfaat untuk membantu mereka memperbaiki. Akhir kata, penulis mengucapkan terima kasih kepada semua orang yang telah membantu dan mendukung. Semoga laporan ini dapat membantu kemajuan ilmu pengetahuan dan teknologi di bidang layanan restoran.

Tim Penyusun

DAFTAR ISI

[KATA PENGANTAR iii](#_Toc189933756)

[DAFTAR ISI v](#_Toc189933757)

[BAB 1 1](#_Toc189933758)

[BAB 2 3](#_Toc189933759)

[A. Perancangan 3](#_Toc189933760)

[B. Aplikasi 3](#_Toc189933761)

[C. Unified Modeling Language (UML) 4](#_Toc189933762)

[1. Use Case Diagram 5](#_Toc189933763)

[2. Sequence Diagram 5](#_Toc189933764)

[3. Class Diagram 6](#_Toc189933765)

[D. *Graphical User Interface*(GUI) 6](#_Toc189933766)

[E. Pemrograman Berorientasi Objek (OOP) 7](#_Toc189933767)

[F. Java 7](#_Toc189933768)

[G. NetBeans 8](#_Toc189933769)

[H. Flowchart 8](#_Toc189933770)

[BAB 3 10](#_Toc189933771)

[A. Spesifikasi Kebutuhan Sistem 10](#_Toc189933772)

[B. Kebutuhan Fungsional 10](#_Toc189933773)

[C. Kebutuhan Non Fungsional 11](#_Toc189933774)

[BAB 4 12](#_Toc189933775)

[A. Diagram Perancangan Sistem 12](#_Toc189933776)

[B. *Use* *Case* Diagram 13](#_Toc189933777)

[*C.* *Use Case Scenario* 14](#_Toc189933778)

[1. Use case User Memilih Menu 14](#_Toc189933779)

[2. Use case User Kelola Meja 14](#_Toc189933780)

[3. Use case User Konfirmasi Pesanan 15](#_Toc189933781)

[4. Use case Melakukan Pembayaran 16](#_Toc189933782)

[5. Use case Memverifikasi Pembayaran (Include dari "Melakukan Pembayaran") 16](#_Toc189933783)

[D. Activity Diagram 18](#_Toc189933784)

[**1.** **Activity Diagram Customer** 18](#_Toc189933785)

[**2.** **Activity Diagram User Kelola Pesanan** 19](#_Toc189933786)

[**3.** **Activity Diagram User Kelola Meja** 19](#_Toc189933787)

[**4.** **Activity Diagram User kelola Pembayaran** 20](#_Toc189933788)

[*E.* *Sequence Diagram* 21](#_Toc189933789)

[**1.** **Sequence Diagram Customer** 21](#_Toc189933790)

[**2.** **Sequence Diagram User Kelola Pesanan** 22](#_Toc189933791)

[**3.** **Sequence Diagram User Kelola Meja** 22](#_Toc189933792)

[**4.** **Sequence Diagram User kelola Pembayaran** 23](#_Toc189933793)

[*F.* *Flowchart* 24](#_Toc189933794)

[BAB 5 25](#_Toc189933795)

[A. Rancangan Halaman Pengguna 25](#_Toc189933796)

[**1.** Perancangan Halaman Selamat Datang 25](#_Toc189933797)

[**2.** Perancangan Halaman Menu Makanan 25](#_Toc189933798)

[**3.** Perancangan Halaman Menu *Dessert* 26](#_Toc189933799)

[**4.** Perancangan Halaman Menu Minuman 27](#_Toc189933800)

[**5.** Perancangan Halaman Pemilihan Tempat Duduk 28](#_Toc189933801)

[**6.** Perancangan Halaman Kelola Pesanan 29](#_Toc189933802)

[**7.** Perancangan Halaman Pembayaran 29](#_Toc189933803)

[**8.** Perancangan Halaman Cetak Resi 31](#_Toc189933804)

[**9.** Perancangan Halaman Penutup 32](#_Toc189933805)

[BAB 6 33](#_Toc189933806)

[A. Tampilan Aplikasi 33](#_Toc189933807)

[**1.** Tampilan Halaman Selamat Datang 33](#_Toc189933808)

[**2.** Tampilan Halaman Menu Makanan 33](#_Toc189933809)

[**3.** Tampilan Halaman Menu *Dessert* 34](#_Toc189933810)

[**4.** Tampilan Halaman Menu Minuman 34](#_Toc189933811)

[**5.** Tampilan Halaman Pemilihan Tempat Duduk 35](#_Toc189933812)

[**6.** Tampilan Halaman Kelola Pesanan 35](#_Toc189933813)

[**7.** Tampilan Halaman Pembayaran 36](#_Toc189933814)

[**8.** Tampilan Halaman Cetak Resi 37](#_Toc189933815)

[**9.** Tampilan Halaman Penutup 38](#_Toc189933816)

[B. Testing (Pengujian) 39](#_Toc189933817)

[**1.** **Pengujian Halaman Selamat Datang** 39](#_Toc189933818)

[**2.** **Pengujian Halaman Menu Makanan** 39](#_Toc189933819)

[**3.** **Pengujian Halaman Menu *Dessert*** 40](#_Toc189933820)

[**4.** **Pengujian Halaman Menu Minuman** 40](#_Toc189933821)

[**5.** **Pengujian Pemilihan Tempat Duduk** 41](#_Toc189933822)

[**6.** **Pengujian Halaman Kelola Pesanan** 42](#_Toc189933823)

[**7.** **Pengujian Halaman Pembayaran** 42](#_Toc189933824)

[**8.** **Pengujian Halaman Cetak Resi** 43](#_Toc189933825)

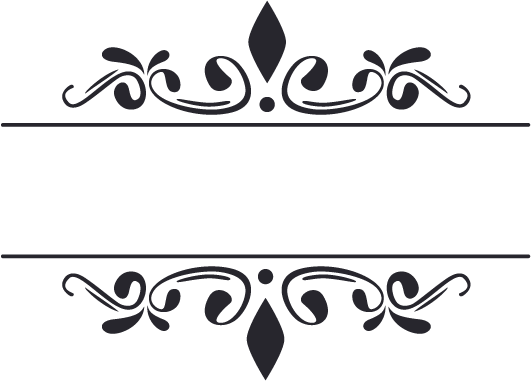
[**9.** **Pengujian Halaman Penutup** 43](#_Toc189933826)

[BAB 7 45](#_Toc189933827)

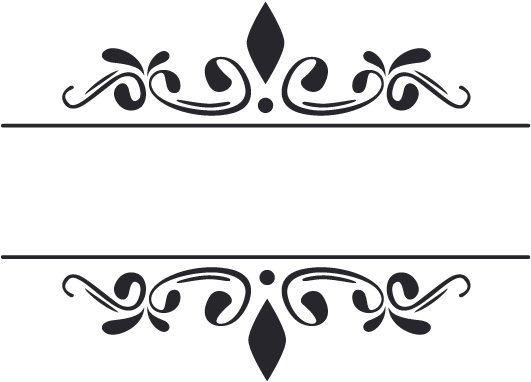
[A. Kesimpulan 45](#_Toc189933828)

[DAFTAR PUSTAKA 46](#_Toc189933829)

[TENTANG PENULIS 48](#_Toc189933830)

****

Sistem Pemesanan dan Pembayaran Mandiri di Restoran: Pendekatan Berorientasi Objek dengan Java

****

# BAB 1

**PENDAHULUAN**

Karena perkembangan teknologi yang semakin pesat, sektor layanan makanan dan minuman telah mengalami banyak perubahan. Saat ini, proses bisnis dapat dilakukan dengan lebih cepat, efisien, dan akurat berkat bantuan teknologi informasi. Selain sektor ekonomi, teknologi informasi memengaruhi bidang pendidikan, hiburan, dan banyak lagi. Penggunaan teknologi informasi mencakup beraneka macam, dari anak-anak hingga orang dewasa, yang semakin bergantung pada berbagai inovasi untuk mempermudah aktivitas sehari-hari mereka.

Sistem *self-serving* atau layanan mandiri, yang memungkinkan pelanggan memesan dan melakukan transaksi tanpa berbicara dengan karyawan, adalah salah satu inovasi yang muncul sebagai hasil dari kemajuan teknologi ini. Salah satu penggunaan teknologi ini adalah Sistem Pemesanan dan Pembayaran Mandiri di Restoran, yang memungkinkan pelanggan restoran memesan makanan, minuman dan pembayaran dengan mudah dan mandiri melalui mesin tiket otomatis. Proses ini tidak hanya mengurangi waktu antrean, tetapi juga meningkatkan efisiensi operasi restoran dan akurasi pemesanan.

Dalam dunia yang semakin terhubung secara digital, konsep otomatisasi seperti ini sangat penting. Dengan menggunakan sistem pemesanan mandiri, manajemen restoran menjadi lebih efisien karena kesalahan manusia berkurang selama proses pemesanan dan pembayaran. Selain itu, pelanggan memiliki pengalaman yang lebih nyaman karena mereka dapat melakukan semua transaksi tanpa perlu berbicara satu sama lain, yang menghemat waktu dan tenaga.

Pemesanan makanan menjadi lebih cepat, akurat, dan fleksibel dengan sistem pemesanan mandiri di restoran. Pemilik restoran juga memperoleh keuntungan dalam hal manajemen karyawan dan memberikan layanan yang lebih terorganisir dan lebih baik. Layanan mandiri seperti ini mendukung tren teknologi yang terus berubah, di mana kecepatan dan efisiensi sangat penting untuk memberikan pengalaman pelanggan terbaik.

# BAB 2

**KONSEP DAN ANALISIS SISTEM**

## Perancangan

Perancangan adalah proses membuat atau mengubah konsep untuk memenuhi kebutuhan manusia dengan cara yang efektif. Dalam proses desain, tahap pertama adalah desain konseptual, yang memerlukan banyak inovasi dan sering kali melibatkan pertimbangan yang tidak jelas. Saat ini, desainer mencari solusi untuk berbagai ide. Setelah itu, desain prototipe dilanjutkan untuk memilih material, dimensi, dan geometri, serta menentukan risiko kegagalan yang mungkin.

Proses perancangan yang lebih lanjut berfokus pada pembuatan desain detail untuk memastikan bahwa semua komponen produk berfungsi dengan baik dan aman sebelum tahap produksi. Faktor emosional pengguna adalah komponen desain yang harus dipertimbangkan. *Kansei Engineering* adalah contoh teknik yang memadukan faktor emosional pengguna dengan parameter desain produk.

Tujuan utama dari perancangan adalah untuk memastikan bahwa produk tidak hanya berguna dan memenuhi standar teknis, tetapi juga memberikan pengalaman pengguna yang menyenangkan. Ini dilakukan dengan mempertimbangkan aspek estetika dan ergonomi yang sesuai dengan preferensi orang-orang.

## Aplikasi

Aplikasi perangkat lunak sistem pemesanan restoran terdiri dari banyak komponen penting. Aplikasi dirancang untuk meningkatkan pengalaman pengguna dengan memungkinkan pengguna mengakses menu, memesan, dan menggunakan berbagai metode pembayaran. Selama proses ini, antarmuka aplikasi harus responsif dan mudah digunakan di berbagai perangkat, seperti ponsel dan desktop. Aplikasi diuji selama tahap pengujian untuk memastikan bahwa itu berfungsi dengan baik pada berbagai platform dan dapat memproses data transaksi dengan cepat dan akurat, sehingga meningkatkan efisiensi operasional restoran.

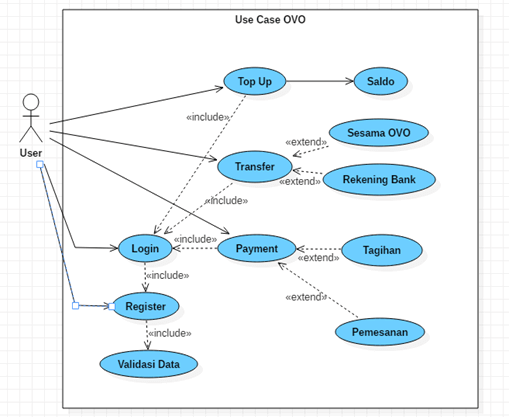
Aplikasi integrasi dengan sistem manajemen restoran, akan membantu mengelola pesanan dan menu dengan lebih baik. Aplikasi diuji pada berbagai perangkat, seperti telepon pintar Android dan sistem berbasis web, untuk memastikan bahwa mereka bekerja dengan baik dalam situasi tertentu, seperti ketika internet tidak selalu tersedia.

## Unified Modeling Language (UML)

Untuk mendokumentasikan, memvisualisasikan, dan merancang sistem berbasis perangkat lunak, bahasa pemodelan *Unified Modeling Language (UML)* menawarkan serangkaian diagram yang memudahkan pengembang perangkat lunak untuk memahami dan merancang komponen sistem. Seperti Java, diagram ini sangat membantu dalam desain sistem berbasis pemrograman berorientasi objek (PBO).

UML memungkinkan pengembang untuk memodelkan sistem secara terstruktur dengan menampilkan interaksi antar objek dan alur kerja. Pada akhirnya, ini meningkatkan kerja sama tim pengembangan dan mengurangi kesalahan desain. Pengembangan perangkat lunak dan model sistem berbasis objek adalah dua contoh aplikasi UML. UML sekarang menjadi standar terbuka yang umum di industri perangkat lunak. Berikut adalah beberapa diagram UML yang umum digunakan.

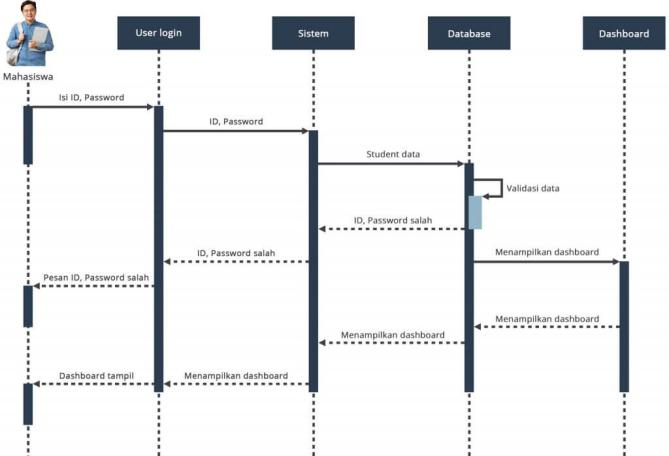
* + 1. Use Case Diagram



**Gambar 1.** Contoh *Use Case* *diagram* pembayaran OVO

Salah satu jenis diagram UML (*Unified Modelling Language*) yang digunakan untuk menunjukkan hubungan antara sistem dan aktor adalah *Use Case Diagram*. Diagram ini dapat menjelaskan jenis interaksi yang terjadi antara pengguna sistem dan sistemnya. Contoh penggunaan adalah sesuatu yang mudah dipelajari. Untuk memulai pemodelan, Anda harus membuat diagram yang dapat menggambarkan tindakan aktor dengan tindakan sistem itu sendiri, seperti yang terlihat dalam *Use Case*.

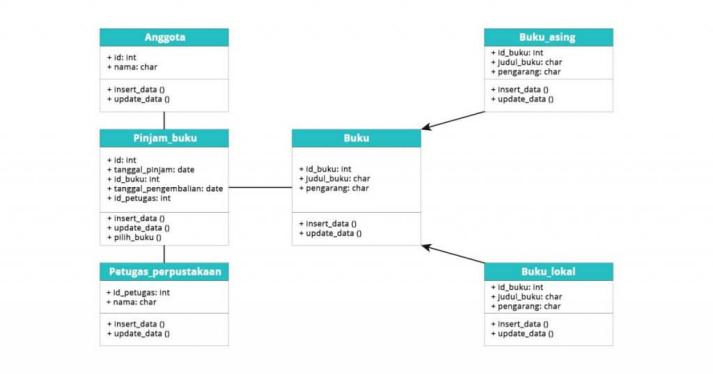
* + 1. Sequence Diagram



**Gambar 2.** Contoh *Sequence* *diagram* *User Login*

*Sequence diagram* dikenal juga diagram urutan, adalah sebuah diagram yang digunakan untuk menunjukkan dan menjelaskan interaksi antar objek-objek dalam sebuah sistem secara menyeluruh. Diagram urutan juga menunjukkan pesan atau perintah yang dikirim, serta waktu yang dibutuhkan untuk menyelesaikannya. Biasanya, item yang berkaitan dengan proses operasi diurutkan dari kiri ke kanan.

* + 1. Class Diagram



**Gambar 3.** Contoh *Class* *diagram* Sistem Perpustakaan

Salah satu jenis diagram struktur yang tersedia dalam UML adalah diagram kelas, yang menampilkan struktur serta deskripsi class, atribut, metode, dan hubungan dari setiap objek. Namun, diagram kelas tidak menjelaskan apa yang terjadi jika kelas-kelasnya berhubungan, tetapi menjelaskan hubungan yang terjadi.

## *Graphical User Interface*(GUI)

GUI adalah antarmuka pengguna grafis yang bisa membuat pengguna berinteraksi dengan perangkat lunak dengan elemen visual seperti ikon, tombol, dan menu. Komponen utama GUI termasuk jendela (untuk menampilkan aplikasi), tombol (untuk menjalankan perintah), kolom teks (untuk memasukkan teks), menu (untuk navigasi), dan ikon (untuk simbol aplikasi atau fungsi tertentu).

Untuk menciptakan pengalaman pengguna yang positif bagi pengguna, antarmuka GUI dirancang dengan prinsip-prinsip dasar seperti konsistensi, keterbacaan, dan aksesibilitas. Keuntungan dari penggunaan GUI termasuk pengalaman pengguna yang lebih mudah, visualisasi data yang lebih baik, dan pengurangan kesalahan penggunaan yang sering terjadi pada antarmuka berbasis teks.

## Pemrograman Berorientasi Objek (OOP)

Pemrograman berorientasi objek (OOP) adalah paradigma pengembangan perangkat lunak yang sangat baik untuk membuat sistem yang lebih kompleks. Tiga konsep utama OOP adalah *encapsulation*, *inheritance*, dan *polymorphism*.

1. *Encapsulation* adalah cara untuk menyembunyikan rincian implementasi objek dan hanya mengekspos *interface* yang diperlukan untuk berinteraksi dengan objek tersebut.
2. *Inheritance* memungkinkan satu kelas untuk mewarisi atribut dan metode dari kelas lain, memungkinkan *reuse* kode yang lebih baik.
3. *Polymorphism* memungkinkan metode yang sama untuk digunakan pada objek yang berbeda, memberikan fleksibilitas lebih dalam pengembangan.

OOP membuat kode lebih mudah dipelihara dan lebih terstruktur. Misalnya, dalam proyek seperti *vending* mesin restoran, elemen seperti produk, transaksi, dan menu dapat dimodelkan secara terpisah dengan fungsi dan data yang terikat dalam kelas-kelas tertentu, meningkatkan *modularitas* dan kemampuan kode untuk dibaca.

## Java

Java adalah bahasa pemrograman yang populer di seluruh dunia, termasuk di Indonesia. Keunggulan utama Java adalah prinsip "*Write Once, Run Anywhere*" (WORA), yang memungkinkan kode yang ditulis dalam Java untuk digunakan secara permanen pada berbagai platform. Ini terjadi karena adanya Java Virtual *Machine* (JVM), yang berfungsi sebagai perantara antara kode Java dan sistem operasi yang digunakan. Karena banyaknya fitur, Java sangat cocok untuk pengembangan aplikasi yang bisa berjalan di berbagai perangkat dan sistem operasi. Selain itu, Java memiliki kemampuan untuk mendukung pengembangan berbasis Pemrograman Berorientasi Objek (OOP), yang sangat penting untuk membuat aplikasi yang modular dan mudah dikelola.

Kemampuan Java untuk berfungsi di berbagai perangkat, dari desktop hingga aplikasi berbasis web, memungkinkan aplikasi untuk dijalankan di proyek sistem pemesanan mandiri restoran tanpa perlu mengubah kode. Ini sangat membantu dalam pengembangan aplikasi yang dapat diskalakan dan portabel.

## NetBeans

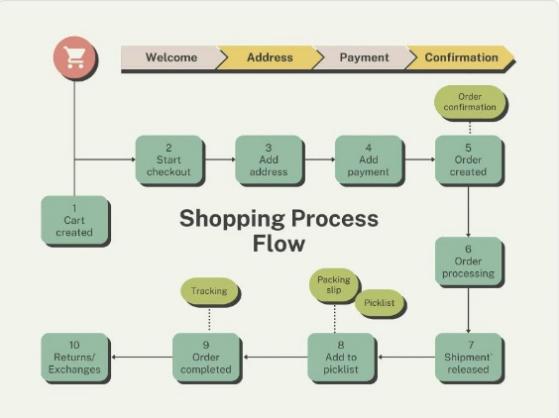
*NetBeans* adalah salah satu IDE (*Integrated Development Environment*) yang paling populer di Indonesia untuk membuat aplikasi berbasis Java. IDE ini memiliki banyak fitur yang sangat membantu pengembang, seperti pengeditan kode otomatis, proposal *sintaks*, dan alat pengelolaan proyek yang memudahkan pengembangan perangkat lunak. Setiawan (2020) menyatakan bahwa *NetBeans* mendukung integrasi dengan berbagai sistem pengelolaan basis data, seperti *MySQL* dan *PostgreSQL*. Ini membantu proyek aplikasi seperti sistem pemesanan mandiri di restoran.

Selain itu, *NetBeans* bagus karena mendukung pengembangan berbasis Pemrograman Berorientasi Objek (PBO) dan mudah digunakan untuk pemeliharaan dan *debugging*. IDE ini memungkinkan integrasi dengan kontrol versi seperti *Git*, yang sangat penting bagi tim pengembang untuk mengelola perubahan kode secara efektif.

## Flowchart

Diagram alur, juga dikenal sebagai diagram alur, adalah representasi grafis dari langkah-langkah dalam suatu proses atau sistem. Diagram alur menunjukkan urutan keputusan, eksekusi, dan aliran data atau informasi yang digunakan dalam algoritma atau proses bisnis. Biasanya, *flowchart* terdiri dari berbagai bentuk geometris yang menunjukkan langkah-langkah tertentu dalam proses; contohnya, persegi panjang menunjukkan proses, berlian menunjukkan keputusan, dan panah menunjukkan arah alur.

Dalam pengembangan perangkat lunak, diagram alur sangat penting untuk merancang algoritma dan menunjukkan alur program secara keseluruhan. *Flowchart* dalam pemrograman membantu pengembang memahami logika program dengan lebih baik dan mempermudah mereka menemukan masalah potensial. Analisis bisnis juga menggunakan *flowchart* untuk memetakan proses kerja dan memaksimalkan efisiensi.



**Gambar 4.** Contoh *Flowchart* pembelian barang *online*

Beberapa simbol yang umum digunakan dalam *flowchart* antara lain:

1. Oval: Menandakan awal atau akhir dari sebuah proses (Start/*End*).
2. Persegi panjang: Digunakan untuk menggambarkan langkah-langkah atau proses dalam alur.
3. Belah ketupat: Menunjukkan titik keputusan, seperti kondisi *True* atau *False*.
4. Panah: Menunjukkan arah aliran proses dari satu langkah ke langkah berikutnya.
5. Paralelogram: Digunakan untuk *input* atau *output* data.

Manfaat *Flowchart*

1. Mempermudah Pemahaman Proses: *Flowchart* memberikan gambaran visual yang jelas tentang bagaimana suatu proses berfungsi.
2. Mendokumentasikan Proses: *Flowchart* digunakan untuk mendokumentasikan proses, sehingga mudah dipahami oleh semua anggota tim atau pemangku kepentingan.
3. Membantu dalam Desain Sistem: Sebelum menulis kode, *flowchart* dapat membantu dalam merancang sistem atau algoritma dengan jelas dan terstruktur.

# BAB 3

**KEBUTUHAN FUNGSIONAL DAN NON FUNGSIONAL**

1. Spesifikasi Kebutuhan Sistem

Dalam menentukan kebutuhan perangkat lunak, analisis kebutuhan perangkat lunak harus dilakukan. Untuk mempermudah tahap analisis kebutuhan perangkat lunak, kebutuhan sistem dibagi menjadi dua kategori: kebutuhan fungsional dan non-fungsional.

1. Kebutuhan Fungsional

Spesifikasi fungsi-fungsi yang harus dimiliki sistem disebut sebagai kebutuhan fungsional. Sistem pemesanan mandiri di restoran harus memenuhi beberapa persyaratan fungsional berikut:

1. **Daftar Produk**

Sistem harus dapat menampilkan semua produk yang tersedia, termasuk nama, deskripsi, harga, dan gambar.

1. **Pemilihan Produk**

Pengguna harus dapat memilih produk dengan mudah melalui antarmuka pengguna yang sederhana. Setelah pemilihan, sistem akan menampilkan konfirmasi pilihan.

1. **Pembayaran**

Sistem harus menyediakan berbagai metode pembayaran, seperti pembayaran tunai, debit dan pembayaran digital. Proses pembayaran harus cepat dan aman.

1. **Pengelolaan Stok**

Sistem harus secara otomatis memperbaharui jumlah stok produk setelah transaksi. Jika stok sedikit, sistem harus memberikan notifikasi bahwa produk habis kepada *user.*

1. **Pencetakan *Struk*/Resi**

Setelah pembayaran selesai, sistem harus mencetak *struk* transaksi sebagai bukti pembelian untuk pengguna.

1. Kebutuhan Non Fungsional

Kebutuhan yang berfokus pada sifat perilaku sistem disebut kebutuhan non-fungsional. Batasan layanan atau fungsi yang diberikan sistem, seperti standarisasi, waktu, dan pengembangan proses, adalah beberapa contoh batasan fungsional. Analisis kebutuhan non-fungsional termasuk analisis perangkat keras, perangkat lunak, dan analisis pengguna.

* + 1. Perangkat Lunak (*Software*)

Untuk membuat *vending* *machine* untuk tiket restoran, perangkat lunak (*software*) yang diperlukan adalah sebagai berikut:

a. Microsoft Windows 10 (64-bit)

b. *Apache NetBeans* IDE 23

c. Browser Google *Chrome* atau Ubuntu

d. Program-program tambahan untuk membantu penyelesaian *Vending Machine* untuk Tiket Restoran: Pendekatan Berorientasi Objek dengan Java

* + 1. Perangkat Keras (*Hardware*)

Perangkat keras yang digunakan untuk membuat *vending* *machine* untuk tiket restoran adalah sebagai berikut:

1. Lenovo Thinkpad T49 AMD Ryzen 5 PRO 3500U w/ Radeon Vega Mobile Gfx 2.10 GHz
2. *Memory*: 8 GB
3. SSD: 254 GB
4. Monitor 14 Inch

# BAB 4

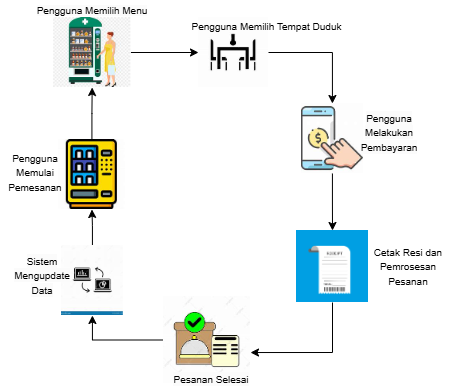
**PEMODELAN PERANGKAT LUNAK**

* + - 1. Diagram Perancangan Sistem

Setelah melewati beberapa tahapan dalam menganalisis berbagai kebutuhan aplikasi sistem, dapat dilakukan berbagai perancangan aplikasi, seperti perancangan diagram blok sistem, alur kerja sistem, dan perancangan model dalam bahasa pemrograman *Unified Modeling Language* (UML), yang terdiri dari *use case diagram, activity diagram,* dan *sequence diagram*.

1. **Melakukan Analisa Perancangan Sistem Berbasis Diagram Blok Sistem**

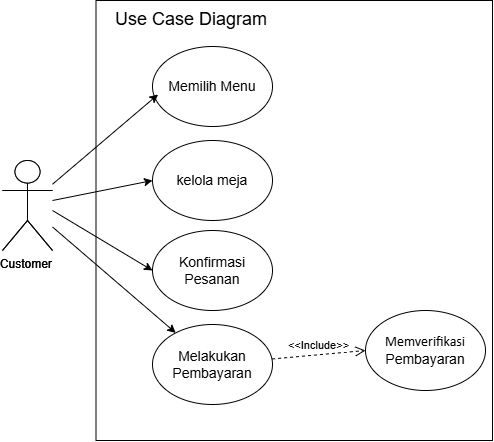
Rancangan diagram blok sistem pemesanan mandiri untuk tiket restoran ditunjukkan di bawah ini:

****

**Gambar 5.** Rancangan Diagram Blok

1. ***Customer Interface* (Layar Sentuh)**
2. Menampilkan menu makanan, minuman, *dessert* dan pilihan meja untuk pelanggan.
3. Pelanggan dapat memilih pesanan dan meja, kemudian melakukan pembayaran melalui layar sentuh.
4. **Sistem Pemesanan Mandiri**
5. Menerima pesanan yang dilakukan oleh pelanggan dan memproses pembayaran.
6. Sistem ini juga mengatur stok produk serta menampilkan makanan atau minuman sesuai pesanan.
   * + 1. *Use* *Case* Diagram

*Use case* diagram dibagi menjadi dua, yaitu *use case user* (pengguna), untuk mendefinisikan interaksi antara satu atau lebih aktor dengan sistem yang akan dibangun. *Use case* diagram juga digunakan untuk mengetahui fungsi sistem, siapa yang ada di dalamnya, dan siapa yang berhak menggunakan fungsi-fungsi tersebut.

****

**Gambar 6.** Rancangan *Use Case Diagram*

* + - 1. *Use Case Scenario*

Gambar *Use Case Diagram* di atas dijelaskan di sini.

* + - 1. Use case User Memilih Menu

**Tabel 1. Memilih Menu**

|  |  |
| --- | --- |
| *Use case* | Memilih Menu |
| Aktor | *Customer* |
| Pra Kondisi | 1. *Customer* sudah memulai pemesanan di sistem restoran. 2. Sistem menampilkan daftar menu makanan, minuman, dan *dessert* yang tersedia. |
| Post Kondisi | 1. Pesanan tersimpan sementara dalam sistem. 2. *Customer* bisa melanjutkan ke pemilihanmeja. |
| Tujuan | Memungkinkan *customer* memilih makanan, minuman, dan *dessert* sebelum memilih meja. |
| Skenario | Sistem menampilkan daftar menu makanan, minuman, dan *dessert*.  *Customer* memilih item yang diinginkan.  *Customer* bisa menambah jumlah atau menghapus item dari daftar pesanan.  *Customer* menekan tombol **"Lanjutkan"** untuk memilih meja. |

* + - 1. Use case User Kelola Meja

**Tabel 2. Kelola Meja**

|  |  |
| --- | --- |
| *Use case* | Kelola Meja |
| Aktor | *Customer* |
| Pra Kondisi | *Customer* telah memilih item makanan, minuman dan *dessert*. |
| Post Kondisi | Meja yang dipilih tercatat pada sistem. |
| Tujuan | Untuk memungkinkan *customer* memilih meja tempat mereka akan menikmati pesanan. |
| Skenario | *Customer* memilih opsi **"Pilih Meja"**.  Sistem menampilkan pilihan tempat duduk **(*Indoor*/*Outdoor*)**.  *Customer* memilih lokasi tempat duduk.  Sistem menampilkan daftar meja yang tersedia.  *Customer* memilih meja yang diinginkan.  Sistem mengonfirmasi pilihan meja.  *Customer* melanjutkan ke tahap **Konfirmasi Pesanan**. |

* + - 1. Use case User Konfirmasi Pesanan

**Tabel 3. Konfirmasi Pesanan**

|  |  |
| --- | --- |
| *Use case* | Konfirmasi Pesanan |
| Aktor | *Customer* |
| Pra Kondisi | *Customer* telah memilih pesanan menu dan meja. |
| Post Kondisi | Pesanan sudah final dan siap untuk pembayaran. |
| Tujuan | Memastikan *customer* bisa mengecek kembali pesanannya sebelum melakukan pembayaran. |
| Skenario | Sistem menampilkan daftar pesanan *customer*.  *Customer* dapat meninjau ulang pesanan.  Jika ada kesalahan, *customer* bisa kembali ke pemilihan menu atau pemilihan meja untuk mengedit pesanan.  Jika sudah benar, *customer* menekan **"Lanjut ke Pembayaran"**. |

* + - 1. ***Use case*** Melakukan Pembayaran

**Tabel 4.** Melakukan Pembayaran

|  |  |
| --- | --- |
| *Use case* | Melakukan Pembayaran |
| Aktor | *Customer* |
| Pra Kondisi | 1. Pesanan telah dikonfirmasi. 2. Sistem menampilkan total harga pesanan. |
| Post Kondisi | 1. Pembayaran berhasil diverifikasi oleh sistem. 2. *Customer* selesai melakukan transaksi. |
| Tujuan | Memungkinkan *customer* menyelesaikan pembayaran menggunakan metode yang tersedia. |
| Skenario | *Customer* memilih metode pembayaran **(QRIS, Debit, Tunai)**.  Jika memilih QRIS, sistem menampilkan QR Code untuk dipindai.  Jika memilih Debit, sistem akan menampilkan pilihan bank dan meminta No. rekening *customer*.  *Customer* melakukan pembayaran.  **Sistem memverifikasi pembayaran** (<> relationship di diagram).  Jika pembayaran berhasil, sistem menampilkan pesan sukses.  *Customer* selesai melakukan transaksi. |

* + - 1. Use case Memverifikasi Pembayaran (Include dari "Melakukan Pembayaran")

**Tabel 5.** Memverifikasi Pembayaran

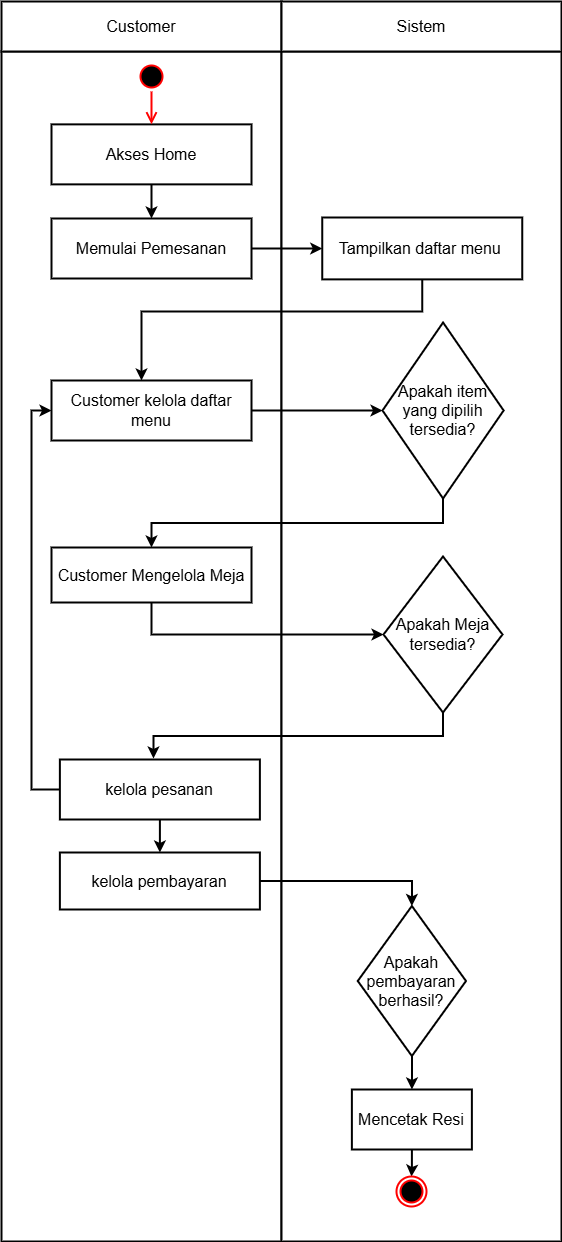
|  |  |
| --- | --- |
| *Use case* | Memverifikasi Pembayaran |
| Aktor | Sistem (dijalankan otomatis) |
| Pra Kondisi | *Customer* telah melakukan pembayaran.( baik melalui *Qris*, Debit, Tunai). |
| Post Kondisi | 1. Sistem memberikan status pembayaran (berhasil atau gagal). 2. Jika pembayaran berhasil, sistem mencetak resi. 3. Jika pembayaran gagal, sistem meminta *customer* untuk mencoba metode lain. |
| Tujuan | Memastikan pembayaran yang dilakukan oleh *customer* sah sebelum transaksi selesai. |
| Skenario | Sistem menerima data pembayaran dari *customer*.  Sistem memverifikasi status pembayaran.  Jika berhasil, sistem mencetak resi.  Jika gagal, sistem meminta *customer* untuk mencoba kembali. |

* + - 1. Activity Diagram

Gambar berikut adalah aktivitas diagram yang dirancang, yang digunakan untuk menjalankan proses alur kerja operasional satu demi satu dari komponen suatu sistem.

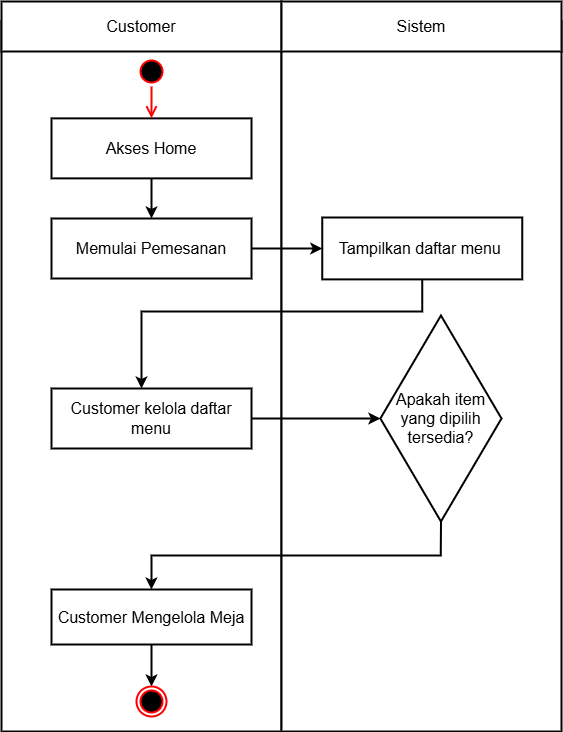
1. **Activity Diagram Customer**

Ini adalah diagram aktivitas pelanggan. Klien dapat mengunjungi halaman menu makanan dan minuman, halaman memilih meja, dan halaman pembayaran .



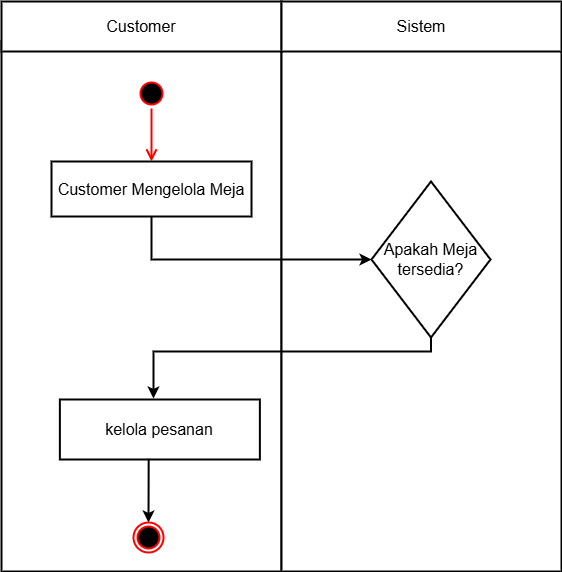
**Gambar 7.** *Activity Diagram Customer*

1. **Activity Diagram User Kelola Pesanan**



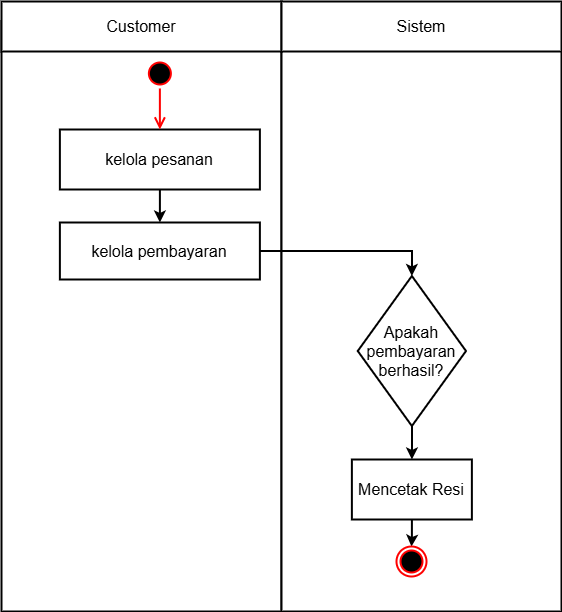
**Gambar 8.** *Activity Diagram User* Kelola Pesanan

1. **Activity Diagram User Kelola Meja**



**Gambar 9.** *Activity Diagram User* Kelola Meja

1. **Activity Diagram User kelola Pembayaran**



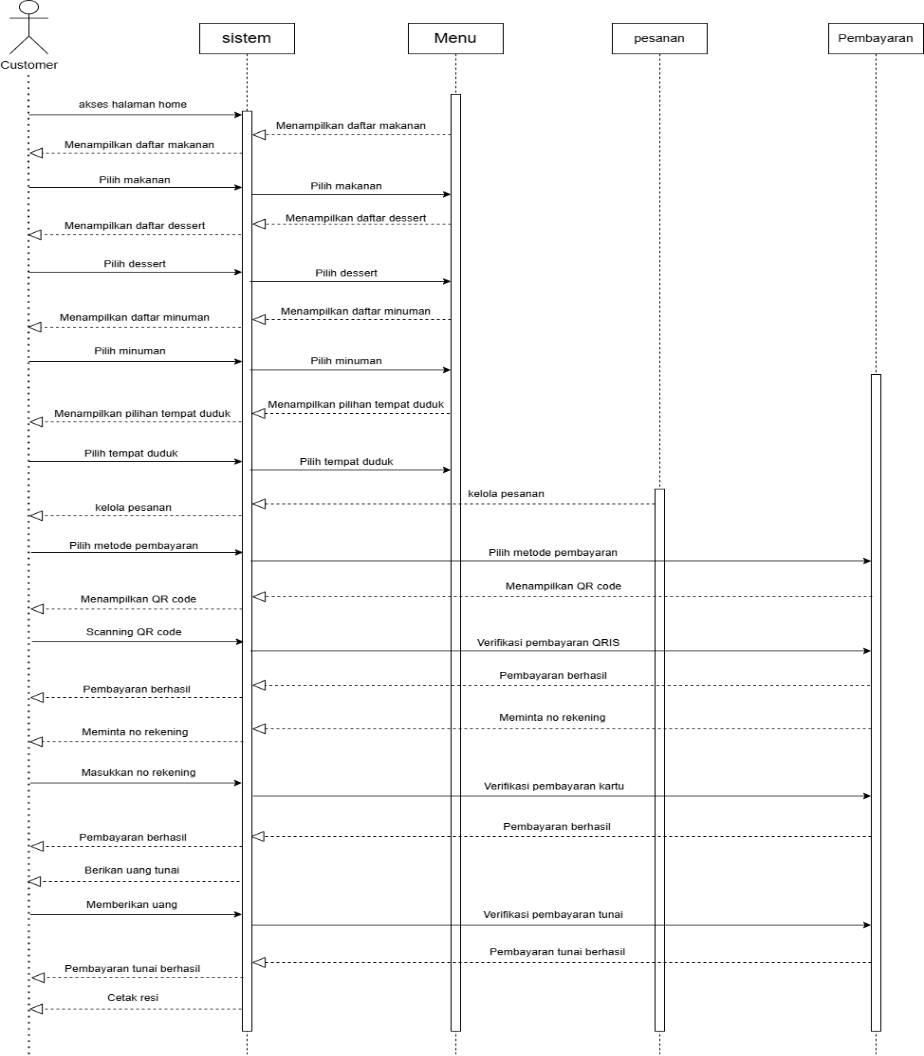
**Gambar 10.** *Activity Diagram User* kelola Pembayaran

* + - 1. *Sequence Diagram*

*Sequence diagram* menunjukkan kerja sama dinamis antara sejumlah objek. Diagram ini berguna untuk menunjukkan rangkaian pesan yang dikirim antara objek dan interaksi mereka satu sama lain.

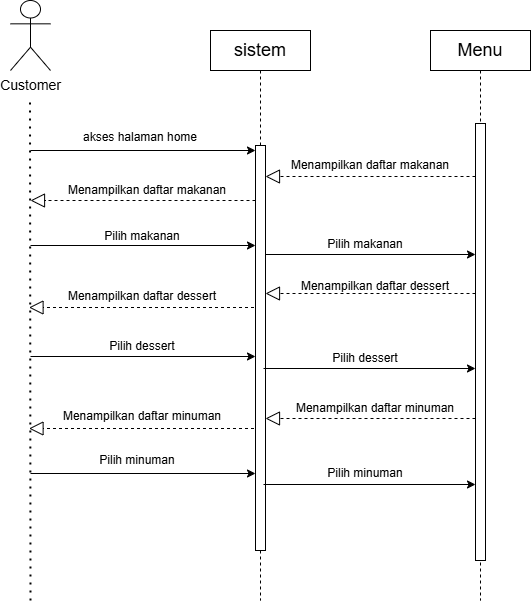
1. **Sequence Diagram Customer**

Berikut ini adalah *sequence diagram customer*, pada *sequence* ini dijelaskan bagaimana langkah-langkah *customer* untuk memilih makanan, minuman, *dessert*, tempat duduk dan metode pembayaran.



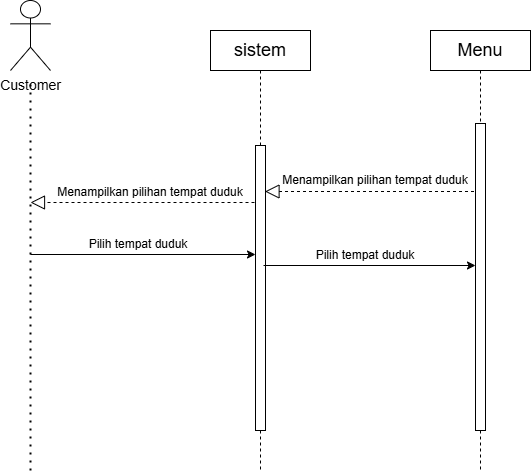
**Gambar 11.** *Sequence Diagram Customer*

1. **Sequence Diagram User Kelola Pesanan**

****

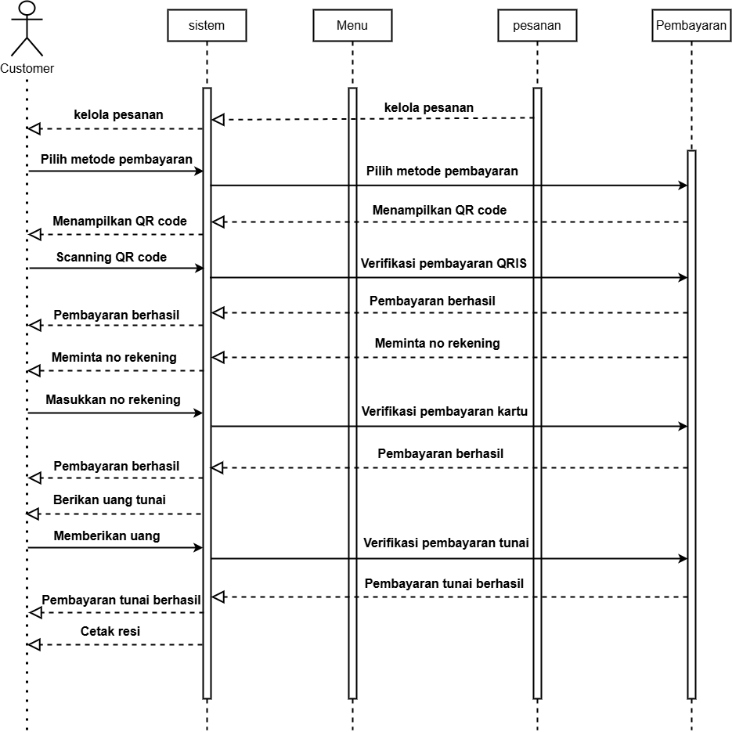
**Gambar 12.** *Sequence Diagram* *User* Kelola Pesanan

1. **Sequence Diagram User Kelola Meja**

****

**Gambar 13.** *Sequence Diagram* *User* Kelola Meja

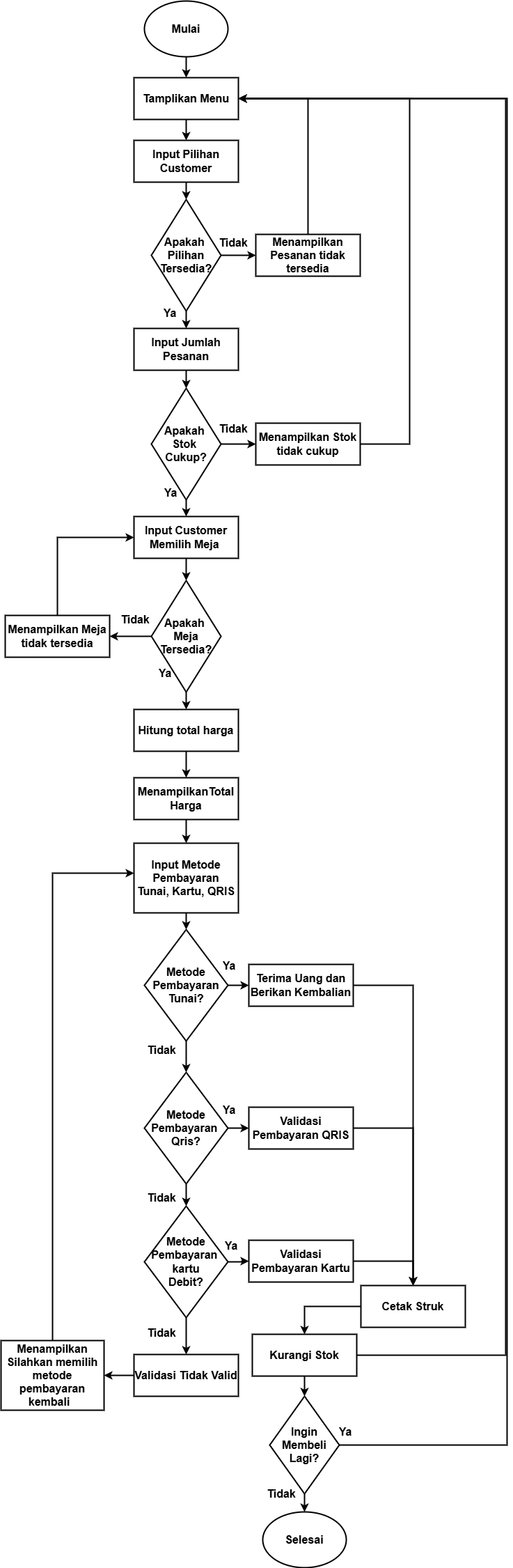
1. **Sequence Diagram User kelola Pembayaran**

****

**Gambar 14.** *Sequence* *Diagram* *User* Kelola Pembayaran

* + - 1. *Flowchart*

Adapun *flowchart* yang digunakan dalam membangun sistem pemesanan mandiri untuk tiket restoran: Pendekatan Berorientasi Objek dengan Java dapat dilihat di bawah ini.:

****

**Gambar 15.** *Flowchart*

# BAB 5

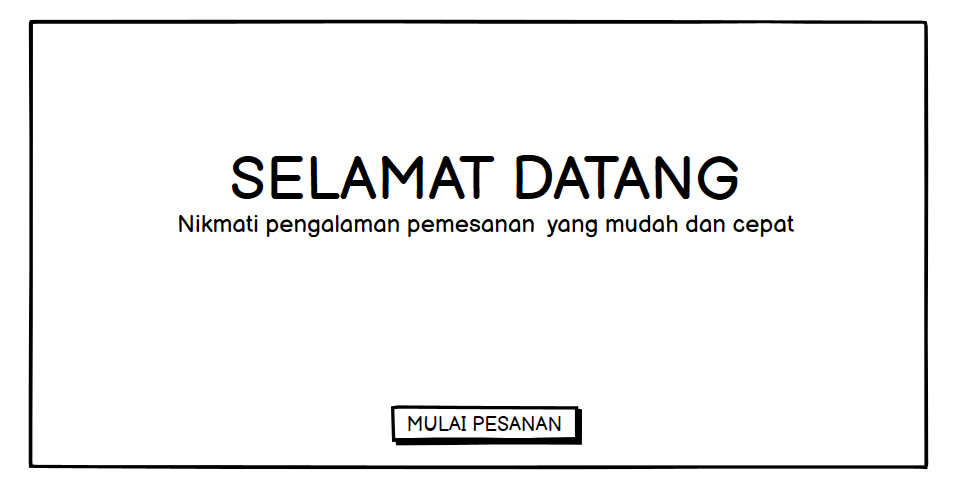
**PERANCANGAN ANTAR MUKA (USER INTERFACE )**

1. Rancangan Halaman Pengguna

Berikut ini merupakan rancangan untuk halaman pengguna yang akan menggunakan sistem pemesanan mandiri untuk tiket restoran: Pendekatan Berorientasi Objek dengan Java.

* + - 1. Perancangan Halaman Selamat Datang

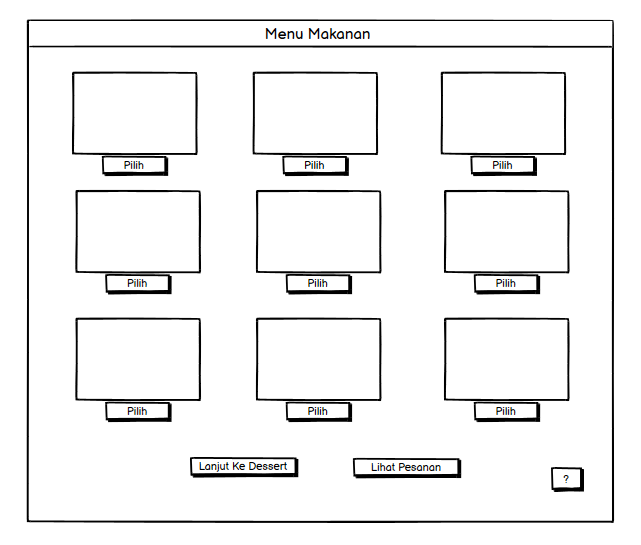
Pada saat pengguna ingin memulai pemesanan, mereka akan dihadapkan pada halaman selamat datang. Halaman ini berisi tombol "Mulai Pesanan" untuk melanjutkan ke proses berikutnya. Berikut perancangan halaman selamat datang.



**Gambar 16.** Desain Halaman Selamat Datang

* + - 1. Perancangan Halaman Menu Makanan

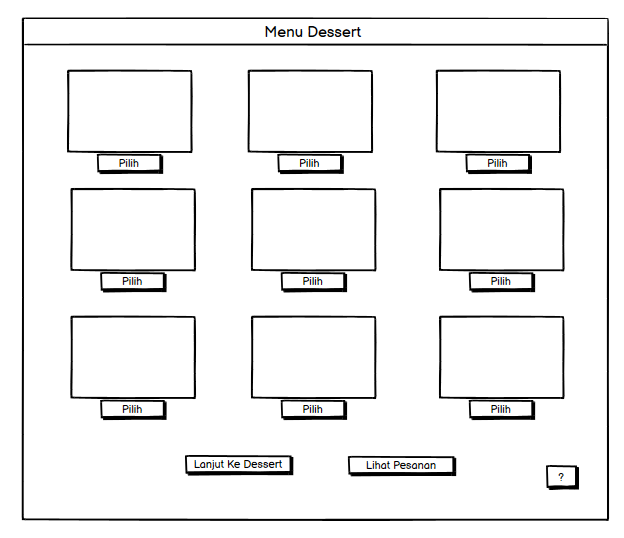
Pada saat pengguna memulai pesanan, mereka akan dihadapkan pada halaman menu makanan. Halaman ini menampilkan daftar makanan utama yang tersedia. Berikut perancangan halaman menu makanan**.**



**Gambar 17.** Desain Halaman Menu Makanan

* + - 1. Perancangan Halaman Menu *Dessert*

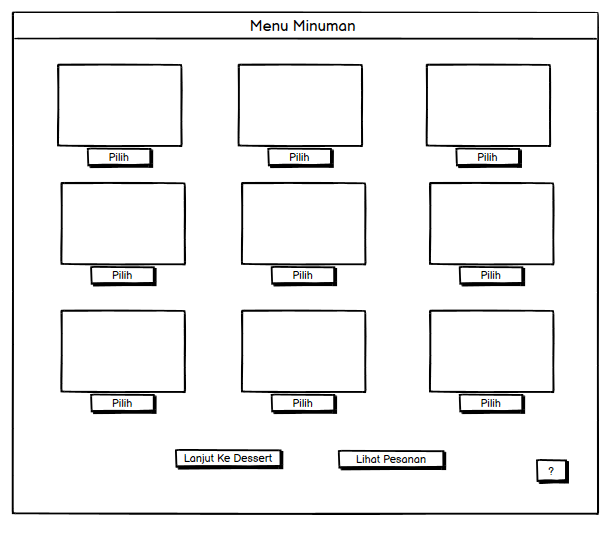
Pada saat pengguna selesai memilih makanan utama, mereka akan diarahkan ke halaman menu *dessert*. Halaman ini memungkinkan pengguna memilih hidangan penutup.Berikut perancangan halaman menu *dessert*.



**Gambar 18.** Desain Halaman Menu *Dessert*

* + - 1. Perancangan Halaman Menu Minuman

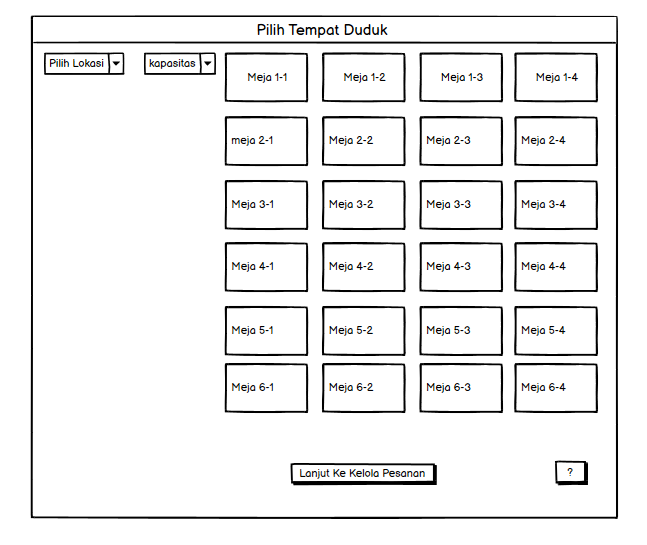
Pada saat pengguna selesai memilih *dessert*, mereka akan diarahkan ke halaman menu minuman. Halaman ini memungkinkan pengguna memilih minuman. Berikut perancangan halaman menu minuman.



**Gambar 19.** Desain Halaman Menu Minuman

* + - 1. Perancangan Halaman Pemilihan Tempat Duduk

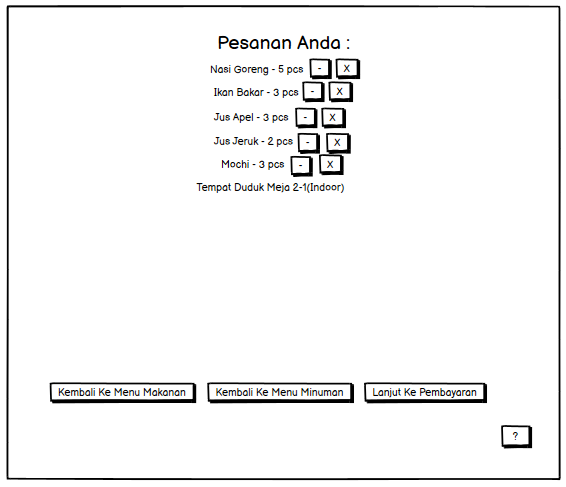
Setelah memilih makanan, minuman, dan *dessert*, pelanggan akan dibawa ke halaman pemilihan tempat duduk, di mana mereka dapat memilih jenis tempat duduk yang diinginkan. Berikut adalah desain halaman ini.



**Gambar 20.** Desain Halaman Pemilihan Tempat Duduk

* + - 1. Perancangan Halaman Kelola Pesanan

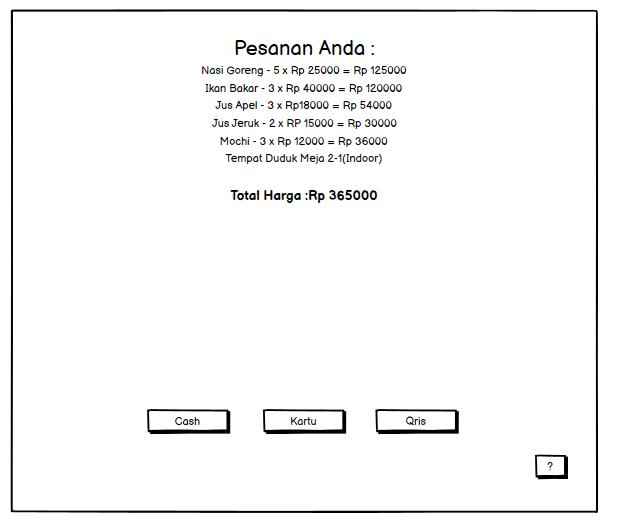
Saat pengguna selesai memilih tempat duduk, mereka akan dibawa ke halaman kelola pesanan. Di halaman ini, mereka akan menemukan daftar menu yang telah mereka pilih dan memiliki kemampuan untuk mengedit pesanan mereka. Ini adalah desain halaman kelola pesanan.



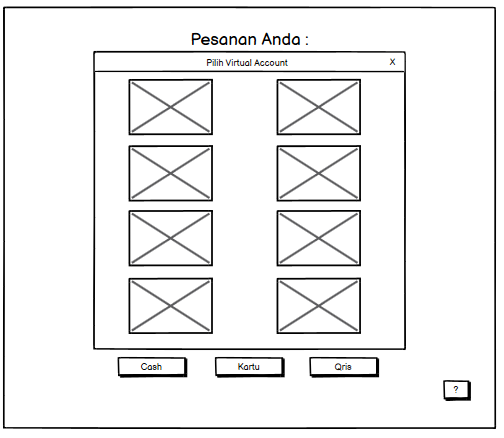
**Gambar.21** Rancangan Halaman Kelola Pesanan

* + - 1. Perancangan Halaman Pembayaran

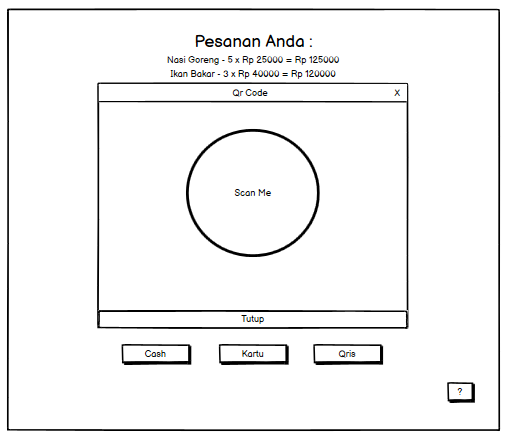
Pada saat pengguna selesai mengatur pesanan, mereka akan diarahkan ke halaman pembayaran. Halaman ini memungkinkan pengguna untuk menyelesaikan pembayaran. Berikut perancangan halaman pembayaran.



**Gambar 22.** Desain Halaman Pembayaran



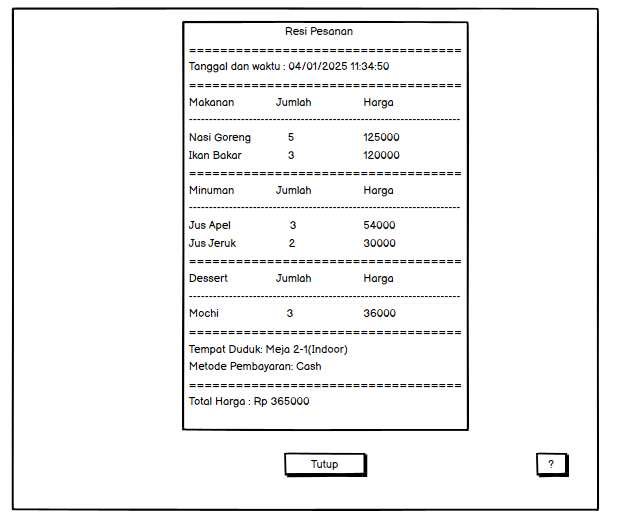
**Gambar 23.** Desain Halaman Pembayaran dengan debit



**Gambar 24.** Desain Halaman Pembayaran dengan *Qris*

* + - 1. Perancangan Halaman Cetak Resi

Pada saat pengguna selesai melakukan pembayaran, mereka akan diarahkan ke halaman cetak resi. Halaman ini menampilkan resi pesanan lengkap setelah pembayaran berhasil. Berikut perancangan halaman cetak resi.



**Gambar 25.** Desain Halaman cetak resi

* + - 1. Perancangan Halaman Penutup

Pengguna akan dibawa ke halaman penutup, yang berisi pesan terima kasih, saat mereka selesai mencetak resi. Ini adalah desain halaman penutup.



**Gambar 26.** Desain Halaman Penutup

# BAB 6

**IMPLEMENTASI APLIKASI**

1. Tampilan Aplikasi

Tampilan halaman program saat dijalankan akan dijelaskan di sini.

1. Tampilan Halaman Selamat Datang

Halaman selamat datang, yang ditunjukkan pada gambar di bawah ini, adalah halaman pertama yang dilihat ketika aplikasi dimulai. Pada halaman ini, Anda dapat menekan tombol "Mulai Pesanan" untuk melanjutkan proses pemesanan.

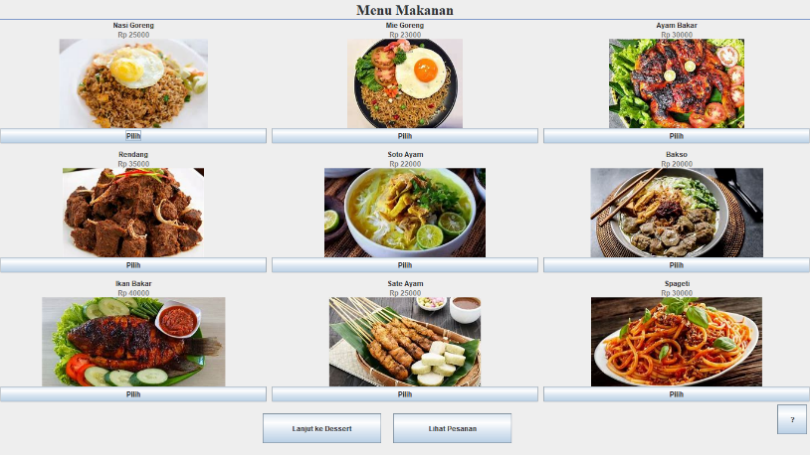


**Gambar 27.** Tampilan Halaman Selamat Datang

1. Tampilan Halaman Menu Makanan

Pengguna dapat memilih makanan dengan lebih mudah di halaman menu makanan. Daftar makanan utama yang tersedia untuk dipesan juga ditampilkan, bersama dengan gambar dan harganya.

.



**Gambar 28.** Tampilan Halaman Dengan Menu Makanan

1. Tampilan Halaman Menu *Dessert*

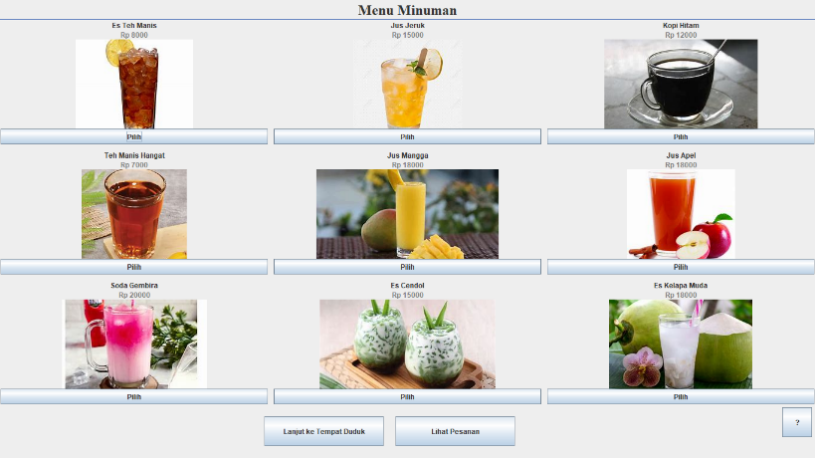
Pelanggan dapat memilih hidangan penutup setelah memilih makanan utama di halaman menu dessert, di mana setiap menu memiliki gambar dan harga.



**Gambar 29.** Tampilan Halaman Dengan Menu *Dessert*

1. Tampilan Halaman Menu Minuman

Pelanggan dapat menambahkan minuman ke dalam pesanan mereka dengan melihat daftar minuman yang tersedia di halaman menu minuman.



**Gambar 30.** Tampilan Halaman Dengan Menu Minuman

1. Tampilan Halaman Pemilihan Tempat Duduk

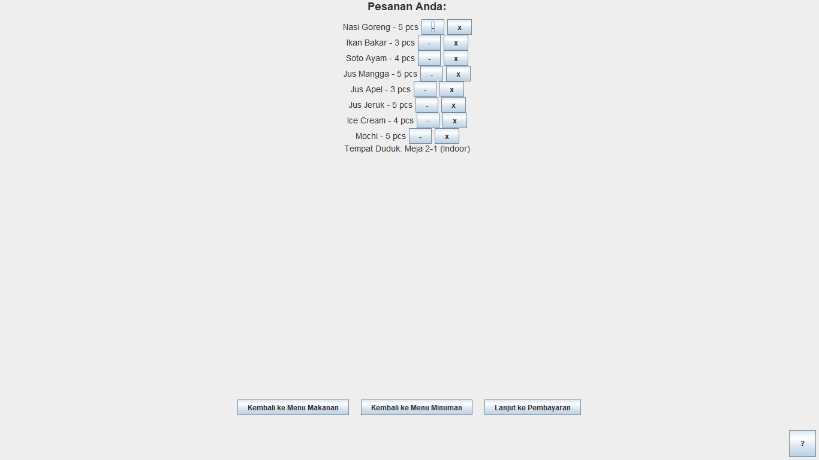
Halaman pemilihan tempat duduk memungkinkan pengguna memilih jenis tempat duduk, yaitu *indoor* atau *outdoor*. Setelah memilih jenis tempat duduk, pengguna dapat memilih nomor meja yang tersedia.



**Gambar 31.** Tampilan Halaman Pemilihan Tempat Duduk

1. Tampilan Halaman Kelola Pesanan

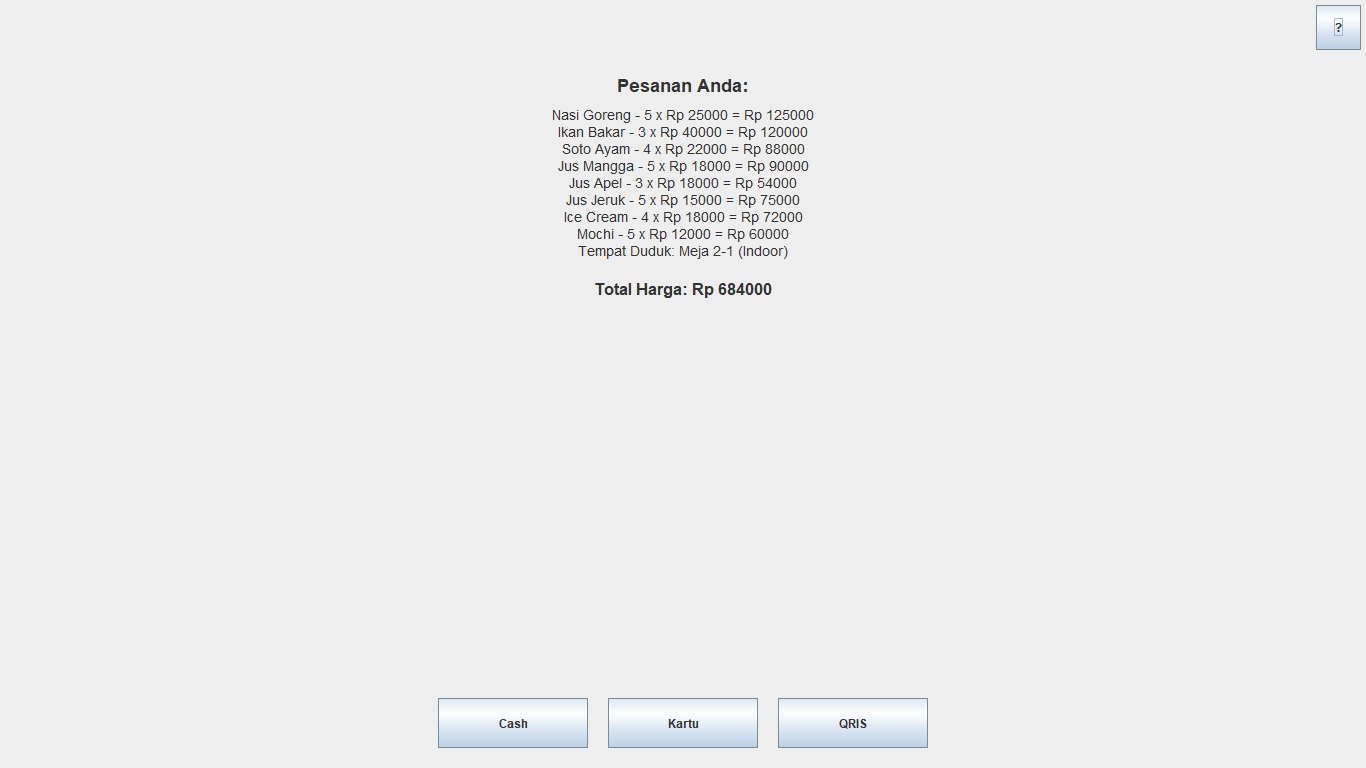
Halaman kelola pesanan menampilkan daftar menu yang telah pengguna pilih. Pada halaman ini, pengguna dapat, Mengurangi jumlah pesanan tertentu, Menghapus pesanan dari daftar, Kembali ke halaman menu untuk menambah pesanan baru, Melanjutkan ke halaman pembayaran.



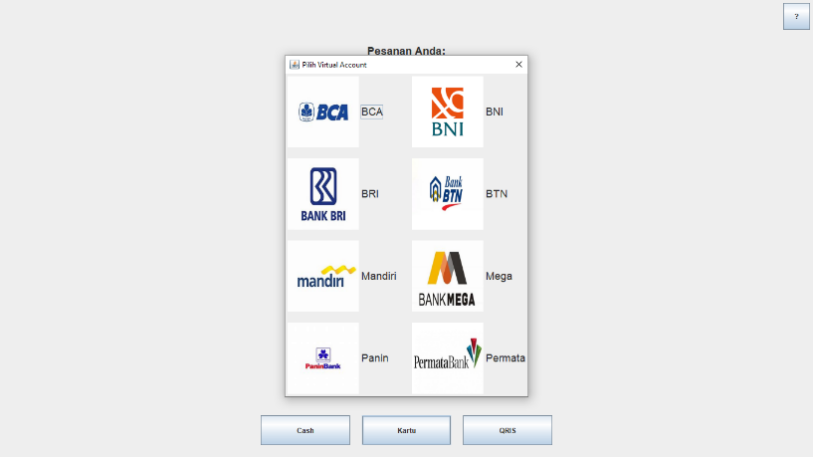
**Gambar 32.** Tampilan Halaman Kelola Pesanan

1. Tampilan Halaman Pembayaran

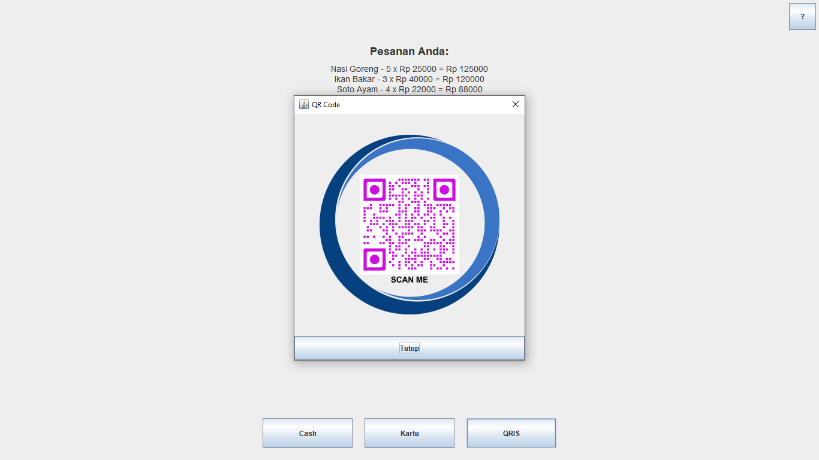
Halaman pembayaran menampilkan ringkasan pesanan lengkap dengan harga dan jumlah. Pengguna dapat memilih metode pembayaran, Cash, Kartu debit (dengan memilih *virtual* *account* bang kemudian memasukkan nomor rekening), Dompet Digital/*QRIS* (dengan *QR* *Code*). Setelah memilih metode pembayaran, pengguna dapat menyelesaikan transaksi.



**Gambar 33.** Tampilan Halaman Pembayaran



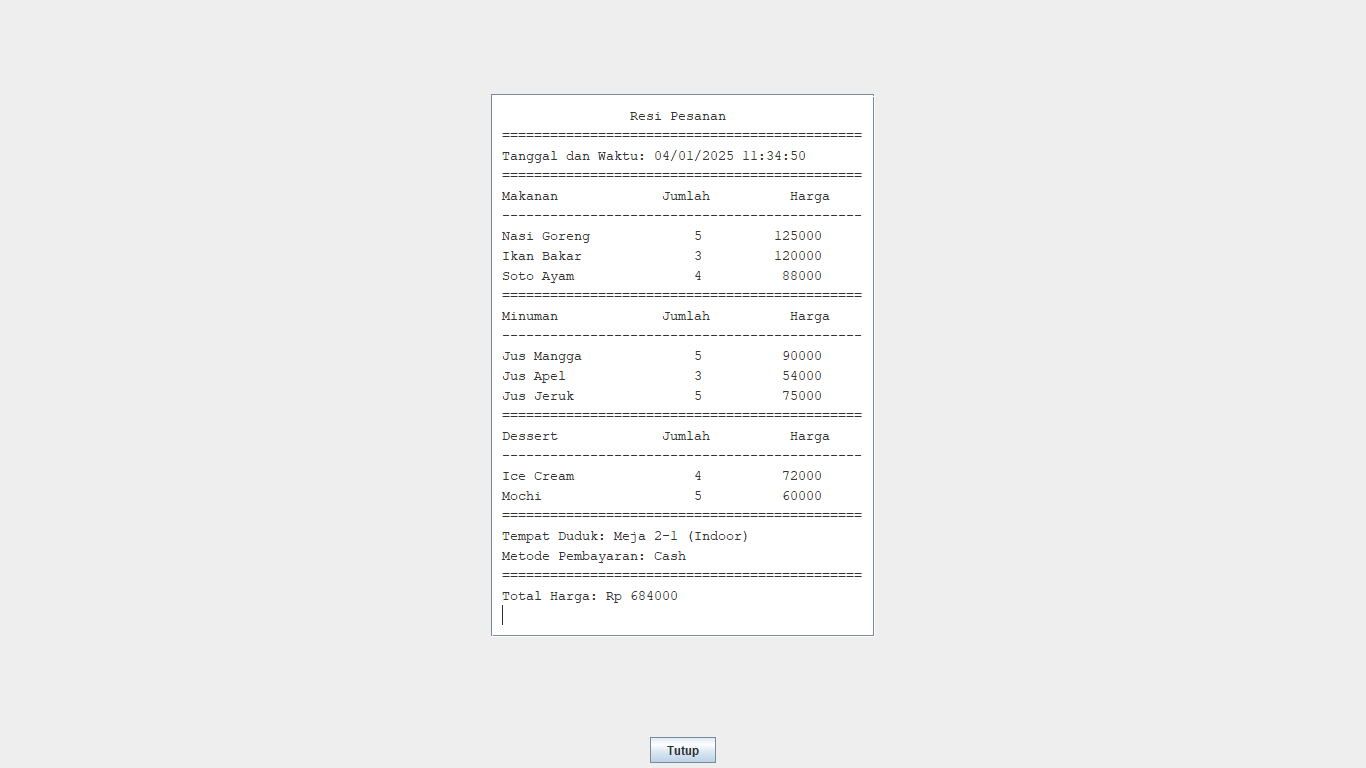
**Gambar 34.** Tampilan Halaman Pembayaran dengan kartu debit



**Gambar 35.** Tampilan Halaman Pembayaran dengan *Qris*

1. Tampilan Halaman Cetak Resi

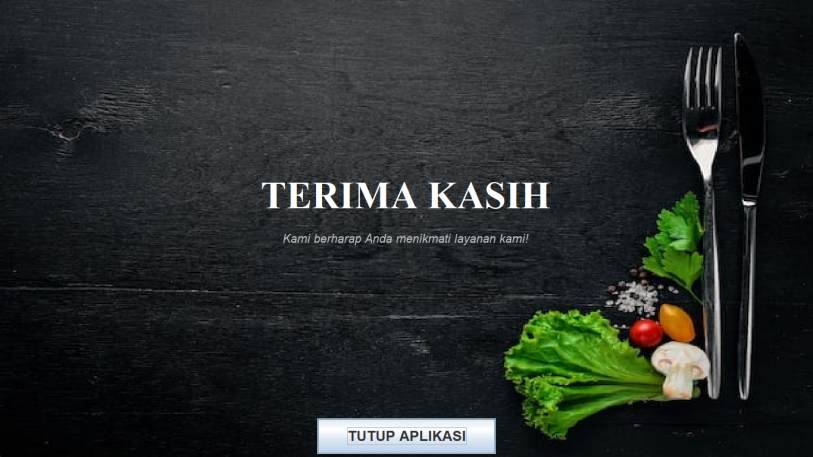
Halaman ini menampilkan resi pesanan lengkap setelah pembayaran berhasil dilakukan. Resi berisi, Daftar pesanan, Total harga, Metode pembayaran yang digunakan. Pengguna dapat mencetak resi dengan menekan tombol "Cetak Resi."



**Gambar 36.** Tampilan Halaman Cetak Resi

1. Tampilan Halaman Penutup

Halaman penutup menampilkan pesan terima kasih kepada pengguna atas penggunaan sistem pemesanan mandiri di restoran. Pengguna dapat menekan tombol 'Selesai' untuk menutup aplikasi.



Gambar 37. Tampilan Halaman Penutup

1. Testing (Pengujian)

Pengujian *black box*—juga dikenal sebagai pengujian perilaku yang digunakan untuk melakukan tahap pengujian manajemen sistem. Pengujian tersebut menguatkan bagaimana mengidentifikasi kesalahan dan kegagalan sistem dari seluruh kebutuhan fungsional sistem pemesanan mandiri untuk tiket restoran: Pendekatan Berorientasi Objek dengan Java.

* + - 1. **Pengujian Halaman Selamat Datang**

Tabel 6. Pengujian Halaman Selamat Datang

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Kasus Pengujian | Hasil yang Diharapkan | Hasil yang Diperoleh | Keterangan |
| 1 | Pengguna Membuka Aplikasi | Sistem menampilkan halaman selamat datang dengan tombol "Mulai Pesanan" | Halaman utama tampil dengan tombol "Mulai Pesanan" | Berhasil |
| 2 | Pengguna menekan tombol "Mulai Pesanan" | Sistem mengarahkan pengguna ke halaman menu makanan | Pengguna diarahkan ke halaman menu makanan | Berhasil |

* + - 1. **Pengujian Halaman Menu Makanan**

Tabel 7. Pengujian Halaman Menu Makanan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| NO | Kasus Pengujian | Hasil yang Diharapkan | Hasil yang Diperoleh | Keterangan |
| 1 | Pengguna melihat daftar makanan | Sistem menampilkan daftar makanan beserta harga dan gambar | Daftar makanan tampil sesuai data menu | Berhasil |
| 2 | Pengguna memilih makanan | Sistem menambahkan item ke daftar pesanan | Item makanan berhasil ditambahkan ke pesanan | Berhasil |
| 3 | Pengguna menekan "Lanjut" | Sistem mengarahkan ke halaman menu *dessert* | Pengguna diarahkan ke halaman menu *dessert* | Berhasil |

* + - 1. **Pengujian Halaman Menu *Dessert***

Tabel 8. Pengujian Halaman Menu *Dessert*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| NO | Kasus Pengujian | Hasil yang Diharapkan | Hasil yang Diperoleh | Keterangan |
| 1 | Pengguna melihat daftar *dessert* | Sistem menampilkan daftar *dessert* beserta harga dan gambar | Daftar *dessert* tampil sesuai data menu | Berhasil |
| 2 | Pengguna memilih *dessert* | Sistem menambahkan item ke daftar pesanan | Item *dessert* berhasil ditambahkan ke pesanan | Berhasil |
| 3 | Pengguna menekan "Lanjut" | Sistem mengarahkan ke halaman menu minuman | Pengguna diarahkan ke halaman menu minuman | Berhasil |

* + - 1. **Pengujian Halaman Menu Minuman**

Tabel 9. Pengujian Halaman Menu Minuman

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Kasus Pengujian | Hasil yang Diharapkan | Hasil yang Diperoleh | Keterangan |
| 1 | Pengguna melihat daftar minuman | Sistem menampilkan daftar minuman beserta harga dan gambar | Daftar minuman tampil sesuai data menu | Berhasil |
| 2 | Pengguna memilih minuman | Sistem menambahkan item ke daftar pesanan | Item minuman berhasil ditambahkan ke pesanan | Berhasil |
| 3 | Pengguna menekan "Lanjut" | Sistem mengarahkan ke halaman pemilihan tempat duduk | Pengguna diarahkan ke halaman pemilihan tempat duduk | Berhasil |

* + - 1. **Pengujian Pemilihan Tempat Duduk**

Tabel 10. Pengujian Halaman Pemilihan Tempat Duduk

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| NO | Kasus Pengujian | Hasil yang Diharapkan | Hasil yang Diperoleh | Keterangan |
| 1 | Pengguna melihat daftar meja yang tersedia | Sistem menampilkan daftar meja *indoor* dan *outdoor* | Daftar meja tampil dengan status ketersediaan | Berhasil |
| 2 | Pengguna memilih meja | Sistem menambahkan pilihan meja ke dalam pesanan | Nomor meja tersimpan dalam sistem | Berhasil |
| 3 | Pengguna menekan "Lanjut" | Sistem mengarahkan ke halaman kelola pesanan | Pengguna diarahkan ke halaman kelola pesanan | Berhasil |

* + - 1. **Pengujian Halaman Kelola Pesanan**

Tabel 11. Pengujian Halaman Kelola Pesanan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| NO | Kasus Pengujian | Hasil yang Diharapkan | Hasil yang Diperoleh | Keterangan |
| 1 | Pengguna melihat daftar pesanan yang telah dipilih | Sistem menampilkan daftar pesanan yang telah dipilih oleh pengguna | Daftar pesanan tampil dengan benar | Berhasil |
| 2 | Pengguna mengubah jumlah pesanan | Sistem memperbarui jumlah pesanan sesuai pilihan pengguna | Jumlah pesanan diperbarui di sistem | Berhasil |
| 3 | Pengguna menghapus item dari daftar pesanan | Sistem menghapus item yang dipilih dari daftar pesanan | Item berhasil dihapus dari daftar | Berhasil |
| 4 | Pengguna menekan "Lanjut ke Pembayaran" | Sistem mengarahkan pengguna ke halaman pembayaran | Pengguna diarahkan ke halaman pembayaran | Berhasil |

* + - 1. **Pengujian Halaman Pembayaran**

Tabel 12. Pengujian Halaman Pembayaran

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| NO | Kasus Pengujian | Hasil yang Diharapkan | Hasil yang Diperoleh | Keterangan |
| 1 | Pengguna memilih metode pembayaran (tunai/kartu/*Qris*) | Sistem menampilkan metode pembayaran yang dipilih | Metode pembayaran tampil sesuai pilihan pengguna | Berhasil |
| 2 | Pengguna memasukkan data pembayaran | Sistem memproses pembayaran | Pembayaran diproses sesuai metode | Berhasil |
| 3 | Pembayaran berhasil | Sistem menampilkan notifikasi sukses dan mencetak resi | Notifikasi muncul dan resi siap dicetak | Berhasil |

* + - 1. **Pengujian Halaman Cetak Resi**

Tabel 13. Pengujian Halaman Cetak Resi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| NO | Kasus Pengujian | Hasil yang Diharapkan | Hasil yang Diperoleh | Keterangan |
| 1 | Pengguna melihat resi pesanan | Sistem menampilkan detail pesanan beserta harga total dan metode pembayaran | Resi tampil sesuai pesanan yang dipilih | Berhasil |
| 2 | Pengguna menekan "Cetak Resi" | Sistem mencetak resi dalam format yang sesuai | Resi berhasil dicetak | Berhasil |

* + - 1. **Pengujian Halaman Penutup**

Tabel 14. Pengujian Halaman Penutup

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| NO | Kasus Pengujian | Hasil yang Diharapkan | Hasil yang Diperoleh | Keterangan |
| 1 | Pengguna melihat halaman penutup setelah mencetak resi | Sistem menampilkan pesan "Terima Kasih" dan tombol "Selesai" | Halaman penutup muncul dengan pesan sesuai | Berhasil |
| 2 | Pengguna menekan "Selesai" | Sistem menutup aplikasi atau kembali ke halaman utama | Aplikasi kembali ke halaman utama | Berhasil |

# BAB 7

**PENUTUP**

1. Kesimpulan

Berikut kesimpulan yang dapat diambil dari perancangan sistem pemesanan mandiri untuk tiket restoran: Pendekatan Berorientasi Objek dengan Java:

* + - 1. **Efisiensi Layanan**: Sistem berhasil meningkatkan efisiensi layanan restoran melalui otomatisasi proses pemesanan, pengelolaan stok, dan pembayaran.
      2. **Kemudahan Penggunaan**: Antarmuka pengguna dirancang sederhana dan mudah digunakan, memungkinkan pelanggan untuk memesan secara mandiri tanpa bantuan staf.
      3. **Pemrograman Berorientasi Objek (OOP)**: Dengan menerapkan prinsip OOP, sistem memiliki struktur kode yang modular, memudahkan pengembangan dan pemeliharaan.
      4. **Integrasi Fitur**: Fitur tambahan seperti pemilihan meja, pembayaran QRIS, dan pencetakan resi membantu memenuhi kebutuhan pelanggan dan meningkatkan pengalaman pengguna.

Dengan laporan ini, diharapkan sistem pemesanan mandiri untuk tiket restoran: Pendekatan Berorientasi Objek dengan Java dapat memberikan kontribusi nyata dalam bidang teknologi layanan restoran dan menjadi acuan bagi pengembangan sistem serupa di masa depan.

DAFTAR PUSTAKA

CodePolitan. (2023). Apa Itu *Object Oriented Programming* (OOP): Pengertian dan Contohnya!. Diakses dari <https://www.codepolitan.com>

Codepolitan. (2023). Pemrograman Berorientasi Objek dengan UML. Diakses dari <https://www.codepolitan.com>

Developer AI Indonesia. (2023). *Graphical User Interface* (GUI). APTIKMA Teknologi Indonesia. <https://aptikma.co.id>

Dicoding. (2024). Apa itu OOP pada Java?. Diakses dari <https://www.dicoding.com>

Jakaria, R. B., & Sukmono, T. (2021). Buku Ajar Mata Kuliah Perencanaan Dan Perancangan Produk. Umsida Press, Sidoarjo, Indonesia.

Local Startup Fest. (2023). Apa Itu GUI dan Cara Kerjanya: Panduan Lengkap. https://www.localstartupfest.id

Nugroho, H., & Arifin, M. (2018). Penerapan *Flowchart* dalam Pengembangan Aplikasi Sistem Informasi. Jurnal Ilmu Komputer, 14(3), 45-52.

Onero.id. (2023). GUI: Pengertian, Fungsi, dan Keunggulan Antarmuka Grafis. <https://onero.id>

Peranginangin, E. (2021). *Kansei Engineering*: Metode Desain Produk berbasis Rekayasa Emosi. Binus University, Jakarta, Indonesia.

Petani Kode. (2023). Mengenal Java: Fitur dan Aplikasinya dalam Pengembangan Aplikasi. Petani Kode, Indonesia.

Santosa, B., & Saputro, I. (2019). Pemodelan Sistem Berbasis *Flowchart* pada Aplikasi Perbankan. Jurnal Teknik Informatika, 6(2), 112-118.

Setiawan, A. (2020). Pengembangan Aplikasi Java Menggunakan *NetBeans* IDE. Jurnal Teknologi Informasi dan Komunikasi, 11(2), 45-56.

Tompoh, J. F., Sentinuwo, S. R., & Sinsuw, A. A. E. (2016). Pengujian Sistem Aplikasi Pemesanan Restoran Berbasis Web dan Android. *E-journal* Teknik Informatika, 9(1). Manado: Universitas Sam Ratulangi.

Universitas Multimedia Nusantara. (2023). Bahasa Pemrograman Java: Keunggulan dan Penggunaan di Industri. Universitas Multimedia Nusantara, Indonesia.

Wikipedia bahasa Indonesia. (2024). *Unified Modeling Language*. Diakses dari https://id.wikipedia.org/wiki/Unified\_Modeling\_Language

TENTANG PENULIS

**PENULIS 1**

**(Aria Fikri Mahadinata)**

Penulis yang bernama Aria Fikri Mahadinata adalah laki-laki kelahiran Jakarta pada tanggal 20 September 2004 Merupakan anak Tunggal. Menghabiskan masa kecil di kota Jakarta, tinggal Bersama orang tua hingga saat ini. Riwayat pendidikan dimulai dari SDN Cilandak Barat 08 (2011-2017), SMPN 37 Jakarta (2017-2020), SMAN 66 Jakarta (2020-2023), dan Sekolah Tinggi Teknologi Informasi NIIT Jakarta (2023-Sekarang).

Setelah menyelesaikan pendidikan wajib saya melanjutkan kuliah di Sekolah Tinggi Teknologi Informasi NIIT pada Program Studi Teknik Informatika pada tahun 2023 Selama berkuliah saya aktif mengikuti berbagai organisasi kemahasiswaan. Dalam mengisi waktu luang selama perkuliahan, saya sering mencari kesibukan untuk menambah ilmu dan pengalaman saya, serta bekerja untuk mencari penghasilan tambahan.

**TENTANG PENULIS**

**PENULIS 2**

**(Ahmad Zidan Tamimy)**

Penulis yang bernama Ahmad Zidan Tamimy adalah Laki-laki kelahiran Jakarta pada tanggal 19 Februari 2005 Merupakan anak kedua dari dua bersaudara Menghabiskan masa kecil di kota Jakarta tinggal Bersama orang tua hingga saat ini. Riwayat pendidikan dimulai dari SDN Cipete Utara 01 (2011-2017), SMPN 240 Jakarta (2017-2020), SMK Keluarga Widuri (2020-2023), dan Sekolah Tinggi Teknologi Informasi NIIT Jakarta (2023-Sekarang).

Setelah menyelesaikan pendidikan wajib saya melanjutkan kuliah di Sekolah Tinggi Teknologi Informasi NIIT pada Program Studi Teknik Informatika pada tahun 2023 Selama berkuliah saya aktif mengikuti berbagai sertifikasi nasional maupun internasional, salah satunya Sertifikasi Microsoft SC-900. Dalam mengisi waktu luang selama perkuliahan, saya sering mencari kesibukan untuk menambah ilmu dan pengalaman saya.