LAPORAN PROJECT UAS (APLIKASI)

PEGAWAI RESTORANT

Dosen Pengampu:Arief Ichwani, M.T

Disusun Oleh:

Hanif Ardiansyah (20230801397)

Bagas Yoas Sibagariang (20230801254)

Nakhwah Alfikry (20230801244)

Dwi Abdul Kholiq (20230801214)

Raja Indera Supriadi (20230810265)

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS ESA UNGGUL

TANGERANG

2024

LAPORAN PROJECT UAS (APLIKASI)

PEGAWAI RESTORANT

PENDAHULUAN

Proyek aplikasi restoran yang dikembangkan oleh kelompok kami bertujuan untuk meningkatkan efisiensi operasional dalam pengelolaan restoran dengan sistem digital yang terintegrasi. Aplikasi ini dirancang untuk membantu manajer dalam mengelola manajemen restoran, termasuk pengaturan jadwal kerja dan pemantauan kinerja pegawai. Selain itu, aplikasi ini juga mendukung pelayan dalam melayani pelanggan dan menyajikan pesanan dengan lebih cepat dan akurat. Satpam dapat menggunakan sistem ini untuk memantau keamanan restoran secara real-time, sementara kasir dapat melakukan transaksi dengan pelanggan secara efisien, mencatat pembayaran, dan mengelola laporan keuangan. Dengan adanya aplikasi ini, diharapkan operasional restoran dapat berjalan lebih efektif dan meningkatkan kepuasan pelanggan.

METODE APLIKASI DALAM KONSEP PBO

Pada aplikasi yang kelompok kami buat pengembangan aplikasi restoran ini, konsep Pemrograman Berorientasi Objek (PBO) diterapkan melalui penggunaan **i**nterface dan polimorfisme untuk meningkatkan fleksibilitas serta modularitas sistem. Interface digunakan untuk mendefinisikan standar perilaku yang harus dimiliki oleh setiap peran dalam sistem, seperti PegawaiInterface yang mendeklarasikan metode umum yang harus diimplementasikan oleh Manager, Pelayan, Satpam**,** dan Kasir**.** Sementara itu, konsep polimorfisme diterapkan agar setiap objek yang berbeda dapat merespons metode yang sama dengan cara yang berbeda sesuai dengan tugasnya. Misalnya, metode kerja yang dideklarasikan dalam interface akan diimplementasikan secara berbeda oleh setiap kelas sesuai dengan fungsinya, seperti kerja pada Pelayan untuk menyajikan pesanan, sementara pada Kasir untuk melakukan transaksi. Dengan pendekatan ini, sistem menjadi lebih fleksibel, mudah dikembangkan, serta memungkinkan penambahan peran baru tanpa mengubah struktur utama kode.

INHERINTANCE

Konsep di mana sebuah kelas (subclass) mewarisi properti (atribut) dan perilaku (metode) dari kelas lain (superclass). Dengan ini, subclass bisa menggunakan atau mengubah sifat-sifat yang dimiliki oleh superclass.

POLIMORFISME

Konsep di mana objek dari kelas yang berbeda dapat diperlakukan secara seragam menggunakan tipe kelas induk, namun menjalankan perilaku yang berbeda sesuai dengan kelas aslinya.

OVERRRIDE

Konsep di mana subclass mengganti implementasi metode yang ada di superclass dengan implementasi baru di subclass. Ini memungkinkan subclass memberikan perilaku yang lebih spesifik.

SUPERCLASS & SUBCLASS

Pegawai sebagai superclass menyimpan data dan metode yang digunakan oleh semua pegawai.
Manajer, Pelayan, Satpam, dan Kasir sebagai subclass dapat menggunakan fitur dari Pegawai dan menambahkan fitur khusus.

Dengan konsep inheritance, Polimorfisme kode menjadi lebih efisien dan mudah dikelola.

LAPORAN PROJECT UAS (APLIKASI)

PEGAWAI RESTORANT

A.Code Pegawai (Super Class)

Class Pegawai adalah **superclass** yang berfungsi sebagai dasar bagi semua jenis pegawai di restoran, seperti **Manajer, Pelayan, Satpam, dan Kasir**. Class ini memiliki atribut umum yang dimiliki oleh semua pegawai, serta metode yang dapat digunakan kembali oleh subclass.

- Atribut dalam Class Pegawai
  - id → ID unik pegawai
  - nama → Nama pegawai
  - nip → Nomor Induk Pegawai
  - gaji → Gaji pegawai dalam bentuk string
  - alamat → Alamat pegawai
  - profesi → Profesi pegawai (seperti "Kasir", "Pelayan", dll.)

- **Konstruktor dalam Class Pegawai**
  Terdapat tiga jenis konstruktor untuk fleksibilitas dalam pembuatan objek pegawai:

```
public Pegawai(int id, String nama, String nip, String gaji, String alamat, String profesi) {
    this.id = id;
    this.nama = nama;
    this.nip = nip;
    this.gaji = gaji;
    this.alamat = alamat;
    this.profesi = profesi;
}

public Pegawai(int id, String nama, String nip, String gaji, String alamat) {
    this.id = id;
    this.nama = nama;
    this.nip = nip;
    this.gaji = gaji;
    this.alamat = alamat;
    this.profesi = "";
}

public Pegawai() {
    this.id = 0;
    this.nama = "";
    this.nip = "";
    this.gaji = "";
    this.alamat = "";
    this.profesi = "";
}
```

LAPORAN PROJECT UAS (APLIKASI)

PEGAWAI RESTORANT

- Getter dan Setter
  Getter dan setter digunakan untuk mengambil dan mengubah nilai atribut.

```
public String getAlamat() { return alamat; }
public void setAlamat(String alamat) { this.alamat = alamat; }

public String getProfesi() { return profesi; }
public void setProfesi(String profesi) { this.profesi = profesi; }

public void tugas() {
    System.out.println("Tugas umum sebagai pegawai restoran.");
}

public void setIdPegawai(int idPegawai) {
    this.id = idPegawai;
}

public void setNamaProfesi(String namaProfesi) {
    this.profesi = namaProfesi;
}
```

- Inheritance
  terjadi ketika sebuah kelas (Manager, Kasir, Koki, dll.) mewarisi atribut dan metode dari kelas Pegawai. Kelas lain dapat mewarisi atribut seperti id, nama, nip, gaji, alamat, dan metode seperti getters, setters,
- Polimorfisme
  Method tugas() didefinisikan dalam superclass Pegawai, lalu di-**override** di setiap subclass agar memiliki tugas yang sesuai dengan profesinya. Di dalam class Pegawai, method tugas() memiliki implementasi umum untuk semua pegawai.

```
public void tugas() {
    System.out.println("Tugas umum sebagai pegawai restoran.");
}
```

  Jika tidak ada overriding di subclass, maka semua pegawai akan memiliki tugas yang sama.
- Override
  Metode tugas() di kelas Manager (atau kelas lainnya) menggantikan implementasi dari kelas induk Pegawai dengan implementasi yang lebih spesifik.

```
@Override
public String toString() {
    return "Pegawai{" +
            "id=" + id +
            ", nama='" + nama + '\'' +
            ", nip='" + nip + '\'' +
            ", gaji='" + gaji + '\'' +
            ", alamat='" + alamat + '\'' +
            ", profesi='" + profesi + '\'' +
            '}';
}
```

LAPORAN PROJECT UAS (APLIKASI)

PEGAWAI RESTORANT

B. Code Kasir

Class Kasir merupakan **subclass** dari **superclass** Pegawai. Artinya, class Kasir mewarisi **a**tribut dan method dari class Pegawai, tetapi juga bisa memiliki atribut dan method tambahan yang spesifik untuk kasir.

- Atribut class kasir
  Atribut mengambil dari kelas pegawai sebagai superclass, dan tugas sebagai atribut unik dari subclass kasir.

- Konstruktor dalam Class Kasir
  Class Kasir memiliki dua konstruktor:
  Konstruktor dengan parameter, digunakan saat semua informasi pegawai kasir tersedia.
  Konstruktor default, digunakan jika data kasir belum lengkap.

```
// Constructor dengan parameter
public Kasir(int id, String nama, String nip, String gaji, String alamat) {
    super(id, nama, nip, gaji, alamat, "Kasir");
    this.tugas = "Melakukan semua transaksi penjualan dengan pembeli";
}

// Constructor tanpa parameter (default)
public Kasir() {
    super(); // Panggil constructor default dari superclass
    this.setProfesi("Kasir");
    this.tugas = "Melakukan semua transaksi penjualan dengan pembeli";
}
```

- Overriding Method tugas() (Polimorfisme)

```
@Override
public void tugas() {
    System.out.println(tugas); // Outputkan tugas
}
```

- Setter dan Getter

```
// Setter untuk tugas
public void setTugas(String tugas) {
    this.tugas = tugas;
}

// Getter untuk tugas (opsional jika diperlukan)
public String getTugas() {
    return this.tugas;
}
```

LAPORAN PROJECT UAS (APLIKASI)

PEGAWAI RESTORANT

C. Code Koki

Class Koki adalah subclass dari superclass Pegawai. Artinya, Koki mewarisi atribut dan method dari Pegawai, tetapi memiliki beberapa tambahan yang spesifik untuk profesi koki.

- Atribut dalam Class Koki
  Selain atribut yang diwarisi dari class Pegawai (id, nama, nip, gaji, alamat, profesi), class Koki memiliki atribut tambahan:

- Konstruktor dalam Class Koki
  Class Koki memiliki dua jenis konstruktor:
  Konstruktor dengan parameter, digunakan saat semua informasi koki tersedia.
  Konstruktor default, digunakan jika data koki belum lengkap.

```
// Constructor dengan parameter
public Koki(int id, String nama, String nip, String gaji, String alamat) {
    super(id, nama, nip, gaji, alamat, "Koki");
    this.tugas = "Memasak makanan dan membuat minuman";
}

// Constructor tanpa parameter (default)
public Koki() {
    super();
    this.setProfesi("Koki");
    this.tugas = "Memasak makanan dan membuat minuman";
}
```

- Overriding Method tugas() (Polimorfisme)
  Class Koki **meng-override method tugas() dari superclass Pegawai**, sehingga perilakunya berubah sesuai peran koki.

```
@Override
public void tugas() {
    System.out.println(tugas);
}
```

- Setter dan Getter

```
public void setTugas(String tugas) {
    this.tugas = tugas;
}

public String getTugas() {
    return this.tugas;
}
```

LAPORAN PROJECT UAS (APLIKASI)

PEGAWAI RESTORANT

D. Code Manager

Class Manager merupakan subclass dari superclass Pegawai. Dengan kata lain, Manager mewarisi atribut dan method dari class Pegawai tetapi juga memiliki fitur tambahan yang spesifik untuk seorang manajer restoran.

- Atribut dalam Class Manager
  Selain atribut yang diwarisi dari class Pegawai (id, nama, nip, gaji, alamat, profesi), class Manager memiliki atribut tambahan: yaitu Menyimpan deskripsi tugas seorang manajer, dengan default: Mengelola seluruh aktivitas restoran
- Konstruktor dalam Class Manager
  Class `Manager` memiliki dua jenis konstruktor:
  Konstruktor dengan parameter, digunakan saat semua informasi manager tersedia.
  Konstruktor default, digunakan jika data manager belum lengkap.

```
// Constructor dengan parameter
public Manager(int id, String nama, String nip, String gaji, String alamat) {
    super(id, nama, nip, gaji, alamat, "Manager");
    this.tugas = "Mengelola seluruh aktivitas restoran";
}

// Constructor tanpa parameter (default)
public Manager() {
    super();
    this.setProfesi("Manager");
    this.tugas = "Mengelola seluruh aktivitas restoran";
}
```

- Overriding Method tugas() (Polimorfisme)
  Polimorfisme memungkinkan objek Manager diperlakukan sebagai Pegawai, tetapi tetap menjalankan method tugas() versi Manager.

```
@Override
public void tugas() {
    System.out.println(tugas);
}
```

- Setter dan Getter

```
public void setTugas(String tugas) {
    this.tugas = tugas;
}

public String getTugas() {
    return this.tugas;
}
```

LAPORAN PROJECT UAS (APLIKASI)

PEGAWAI RESTORANT

E. Code Pelayan

Class Pelayan adalah **subclass dari superclass Pegawai**. Artinya, Pelayan mewarisi atribut dan method dari Pegawai, tetapi memiliki tambahan fitur yang spesifik untuk profesi pelayan.

- Atribut dalam Class Pelayan
  Selain atribut yang diwarisi dari class Pegawai (id, nama, nip, gaji, alamat, profesi), class Pelayan memiliki atribut tambahan:yaitu Menyimpan deskripsi tugas pelayan
- Konstruktor dalam Class Pelayan
  Class Pelayan memiliki dua jenis konstruktor:
  Konstruktor dengan parameter, digunakan saat semua informasi pelayan tersedia.
  Konstruktor default, digunakan jika data pelayan belum lengkap.

```
// Constructor dengan parameter
public Pelayan(int id, String nama, String nip, String gaji, String alamat) {
    super(id, nama, nip, gaji, alamat, "Pelayan");
    this.tugas = "Melayani dan menyajikan pesanan pembeli";
}

// Constructor tanpa parameter (default)
public Pelayan() {
    super();
    this.setProfesi("Pelayan");
    this.tugas = "Melayani dan menyajikan pesanan pembeli";
}
```

- Overriding Method tugas() (Polimorfisme)
  Polimorfisme memungkinkan objek Pelayan diperlakukan sebagai Pegawai, tetapi tetap menjalankan method tugas() versi Pelayan.

```
@Override
public void tugas() {
    System.out.println(tugas);
}
```

- Setter dan Getter

```
public void setTugas(String tugas) {
    this.tugas = tugas;
}

public String getTugas() {
    return this.tugas;
}
```

LAPORAN PROJECT UAS (APLIKASI)

PEGAWAI RESTORANT

F.Code Satpam

Class Satpam adalah **subclass dari superclass Pegawai**, yang berarti Satpam mewarisi semua atribut dan method dari Pegawai, tetapi memiliki tambahan fitur spesifik sesuai dengan peran satpam di restoran.

- Atribut dalam Class Satpam
  Selain atribut yang diwarisi dari class Pegawai (id, nama, nip, gaji, alamat, profesi), class Satpam memiliki atribut tambahan:yaitu Menyimpan deskripsi tugas satpam
- Konstruktor dalam Class Satpam
  Class Satpam memiliki dua jenis konstruktor:
  Konstruktor dengan parameter, digunakan saat semua informasi satpam sudah tersedia.
  Konstruktor default, digunakan jika data satpam belum lengkap.

```
// Constructor dengan parameter
public Satpam(int id, String nama, String nip, String gaji, String alamat) {
    super(id, nama, nip, gaji, alamat, "Satpam");
    this.tugas = "Menjaga keamanan di dalam dan di luar restoran";
}

// Constructor tanpa parameter (default)
public Satpam() {
    super();
    this.setProfesi("Satpam");
    this.tugas = "Menjaga keamanan di dalam dan di luar restoran";
}
```

- Overriding Method tugas() (Polimorfisme)
  Class Satpam meng-override method **tugas()** dari superclass Pegawai, sehingga tugas yang dijalankan spesifik untuk peran satpam.

```
@Override
public void tugas() {
    System.out.println(tugas);
}
```

- Setter dan Getter

```
public void setTugas(String tugas) {
    this.tugas = tugas;
}

public String getTugas() {
    return this.tugas;
}
```

LAPORAN PROJECT UAS (APLIKASI)

PEGAWAI RESTORANT

PROJECT DATA KARYAWAN

Berfungsi sebagai penanggung jawab untuk melakukan operasi CRUD (Create, Read, Update, Delete) terhadap data pegawai dalam sebuah database menggunakan JDBC (Java Database Connectivity). Mari kita jelaskan inti dari kode ini beserta fungsinya:

1. Koneksi Database
   Kelas tersebut memiliki atribut koneksi ke database yaitu koneksi Database. Koneksi ini dibangun dalam konstruktor kelas dengan memanggil metode koneksi.koneksiDB.

```
private final Connection koneksiDatabase;

public ProjectDataPegawai() {
    this.koneksiDatabase = koneksi.koneksiDB();
}
```

2. Menampilkan Semua Pegawai
   Metode tampil Semua mengembalikan daftar semua pegawai dengan mengambil data dari database dengan melakukan JOIN antara tabel pegawai dan profesi.Hasil dari query dipetakan menjadi objek Pegawai dengan fungsi create Pegawai Instance.

```
public List<Pegawai> tampilSemua() {
    String query = "SELECT p.id_pegawai, p.Nama, p.NIP, p.gaji, p.Alamat, pr.nama_profesi " +
                   "FROM pegawai p JOIN profesi pr ON p.id_profesi = pr.id_profesi";
    List<Pegawai> list = new ArrayList<>();
```

3. Mendapatkan Pegawai Berdasarkan ID
   Metode byId(int id pegawai) mengambil data pegawai tertentu menggunakan ID dan mengembalikan objek Pegawai yang sesuai.

```
public Pegawai byId(int id_pegawai) {
    String query = "SELECT p.*, pr.nama_profesi FROM pegawai p JOIN profesi pr ON p.id_profesi = pr.id_profesi WHERE id_
    Pegawai model = null;
```

4. Menambahkan Data Pegawai
   Metode insert(String nama, String nip, String gaji, String alamat, int idProfesi) digunakan untuk menambahkan pegawai baru ke database.

```
public boolean insert(String nama, String nip, String gaji, String alamat, int idProfesi) {
    String query = "INSERT INTO pegawai (Nama, NIP, gaji, Alamat, id_profesi) VALUES (?, ?, ?, ?, ?)";
```

5. Memperbarui Data Pegawai
   Metode update(String nama, String nip, String gaji, String alamat, int idProfesi, int id_pegawai) untuk memperbarui informasi pegawai yang ada berdasarkan ID.

```
public boolean update(String nama, String nip, String gaji, String alamat, int idProfesi, int id_pegawai) {
    String query = "UPDATE pegawai SET nama = ?, nip = ?, gaji = ?, alamat = ?, id_profesi = ? WHERE id_pegawai = ?";
```

6. Menghapus Data Pegawai
   Metode delete(int id_pegawai) digunakan untuk menghapus pegawai dari database berdasarkan ID-nya.

```
public boolean delete(int id_pegawai) {
    String query = "DELETE FROM pegawai WHERE id_pegawai = ?";
```

7. Membuat Instance Pegawai
   Fungsi createPegawaiInstance(...) digunakan untuk membuat objek Pegawai baru dari data yang diambil dari hasil query.

```
private Pegawai createPegawaiInstance(int idPegawai, String nama, String nip, String gaji, String alamat, String namaProfesi
    Pegawai pegawai = new Pegawai();
    pegawai.setIdPegawai(idPegawai);
    pegawai.setNama(nama);
    pegawai.setNip(nip);
    pegawai.setGaji(gaji);
    pegawai.setAlamat(alamat);
    pegawai.setNamaProfesi(namaProfesi);
    return pegawai;
}
```

8. Manajemen Berdasarkan Jabatan
   terdapat metode untuk menampilkan, mendapatkan berdasarkan ID, dan membuat instance untuk beberapa jabatan pegawai lainnya, seperti Manager, Koki, Kasir, Pelayan, dan Satpam.
   Fungsi Utama code:
   - Kelas ini berfungsi sebagai lapisan model untuk mengelola data pegawai, yang berinteraksi dengan database.
   - Mencakup semua fungsi dasar yang diperlukan untuk mengelola pegawai restaurant dalam sistem, dari menampilkan semua pegawai hingga mengelola pegawai berdasarkan profesi tertentu.

LAPORAN PROJECT UAS (APLIKASI)

PEGAWAI RESTORANT

CONTROLER (KONEKSI JAVA)

```
public class koneksi {
    private static Connection mysqlkonek;

    public static Connection koneksiDB() {
        if(mysqlkonek==null){
            try {
                String nama_database = "pegawai_restoran"; // nama database
                String DB="jdbc:mysql://localhost:3306/"+nama_database;
                String user="root"; // user
                String pass=""; // password database
                DriverManager.registerDriver(new com.mysql.cj.jdbc.Driver());
                mysqlkonek = (Connection) DriverManager.getConnection(DB,user,pass);
            } catch (SQLException e) {
                e.printStackTrace();
                JOptionPane.showMessageDialog(null,"Koneksi Gagal, pastikan MySQL berjalan !");
            }
        }
        return mysqlkonek;
    }
}
```

Kelas koneksi berfungsi untuk mengatur koneksi ke database MySQL dengan menggunakan JDBC. Metode statis koneksiDB() pertama-tama memeriksa apakah koneksi sudah ada, dan jika belum, maka akan membuat koneksi baru ke database pegawai_restoran yang ada di localhost dengan menggunakan kredensial root dan tanpa password.

URL koneksi dibentuk dengan format jdbc:mysql://localhost:3306/pegawai_restoran. Driver MySQL didaftarkan melalui DriverManager, dan koneksi diatur dengan DriverManager.getConnection(). Jika koneksi gagal, pesan kesalahan ditampilkan kepada pengguna. Pola Singleton diterapkan di sini untuk memastikan bahwa hanya ada satu koneksi yang dibuka selama aplikasi berjalan. Dengan demikian, kelas ini mempermudah pengelolaan dan penggunaan koneksi database dalam aplikasi.

HASIL DAN PEMBAHASAN

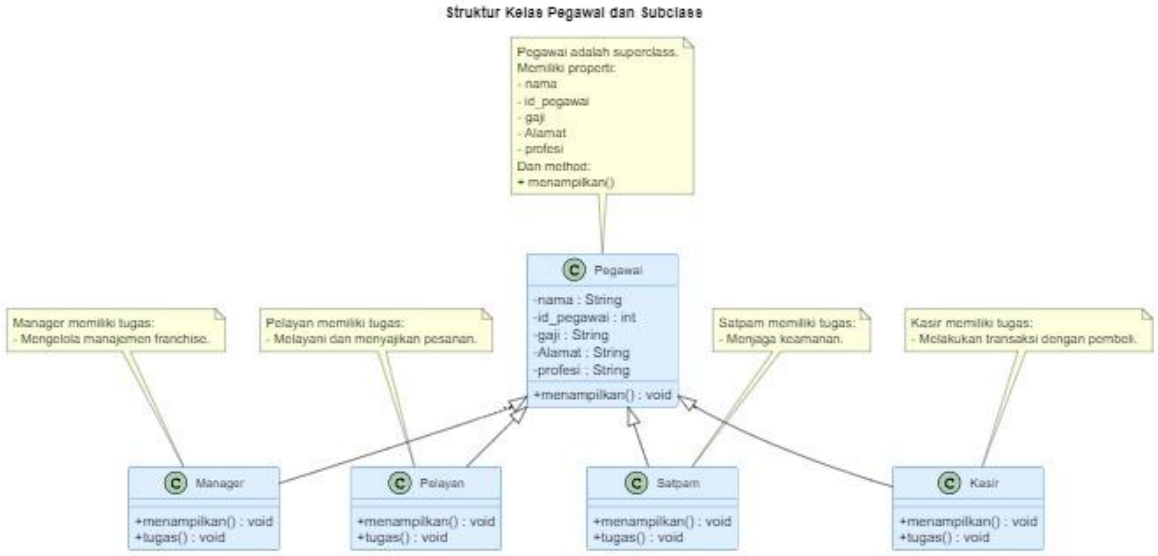
A. Class Diagram

Struktur kelas yang digambarkan dalam diagram ini sangat efektif untuk pengorganisasian dan manajemen berbagai peran pegawai dalam sebuah sistem aplikasi, seperti pengelolaan pegawai di restoran atau tempat usaha lainnya. Dengan menggunakan konsep inheritance (pewarisan), kelas induk Pegawai menyediakan properti dan metode dasar yang umum untuk semua pegawai, seperti nama, id pegawai, dan gaji, yang kemudian diwariskan kepada kelas-kelas turunan seperti Manager, Pelayan, Satpam, dan Kasir. Setiap subclass ini memiliki tugas khusus yang diimplementasikan melalui metode tugas dan menampilkan, yang dapat diubah sesuai dengan kebutuhan spesifik dari peran masing-masing pegawai. Misalnya, kelas Pelayan berfokus pada layanan dan penyajian pesanan, sementara kelas Kasir menangani transaksi pembayaran, dan Satpam bertanggung jawab menjaga keamanan. Konsep polimorfisme memungkinkan setiap kelas untuk memberikan implementasi berbeda dari metode yang sama, tergantung pada tugas yang harus dijalankan, sehingga memudahkan pemeliharaan dan pengembangan aplikasi yang skalabel dan terstruktur. Dengan demikian,

LAPORAN PROJECT UAS (APLIKASI)

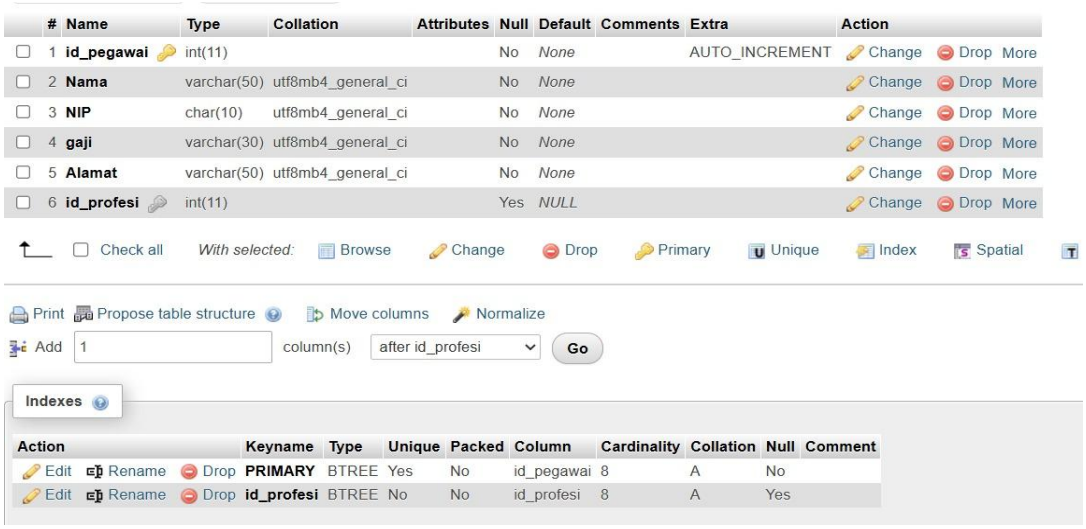
PEGAWAI RESTORANT

aplikasi yang dikembangkan dapat secara efisien mengelola berbagai peran pegawai, mempercepat proses kerja, dan meningkatkan efektivitas operasional.



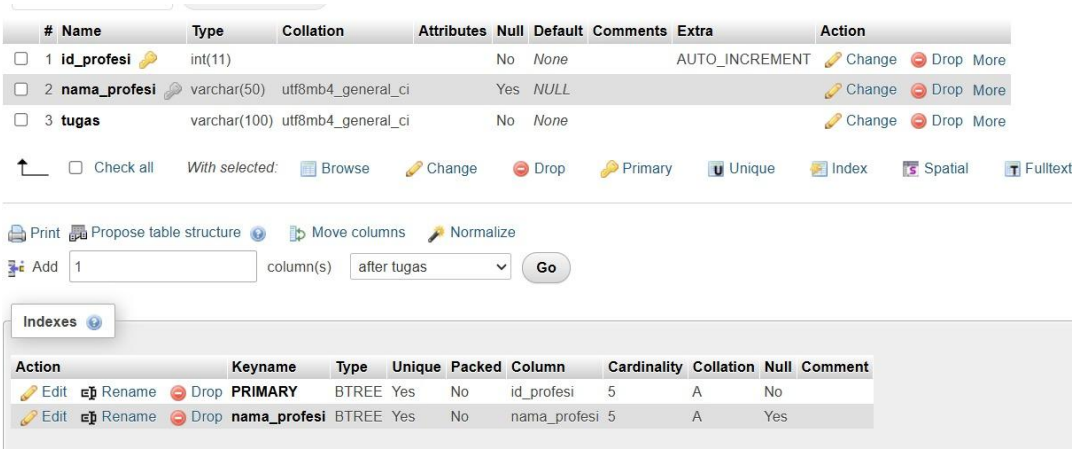
B. Tampilan pada XAMPP

Xampp ini berfungsi sebagai penyimpanan data/database yang telah diisi melaui aplikasi yang telah dibuat, lalu data akan tersimpan pada sebuah database secar otomatis dan juga unutk merelasikan hubungan antar tabel



LAPORAN PROJECT UAS (APLIKASI)

PEGAWAI RESTORANT



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id_profesi	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	nama_profesi	varchar(50)	utf8mb4_general_ci		Yes	NULL			Change Drop More
3	tugas	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More

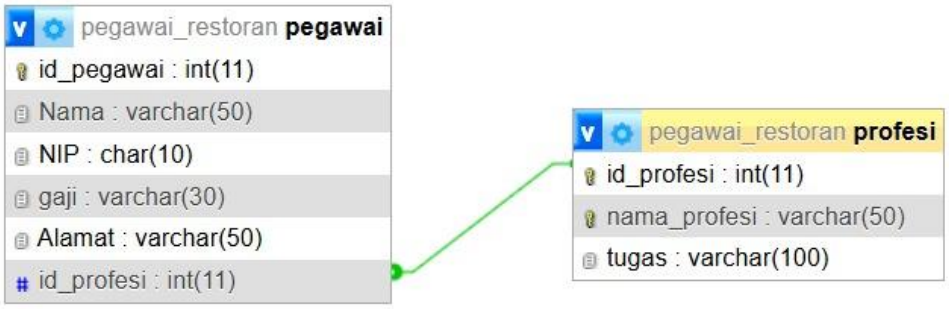
Check all With selected: Browse Change Drop Primary Unique Index Spatial Fulltext

Print Propose table structure Move columns Normalize

Add 1 column(s) after tugas Go

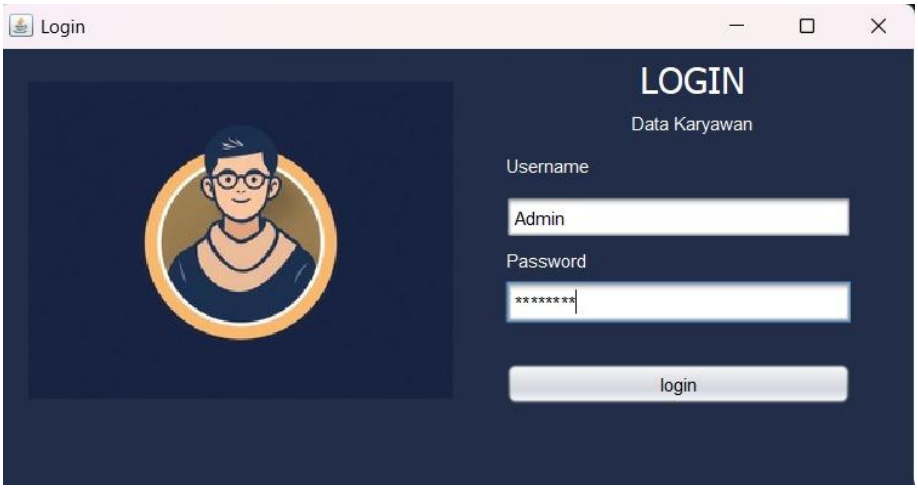
Indexes

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Rename Drop	PRIMARY	BTREE	Yes	No	id_profesi	5	A	No	
Edit Rename Drop	nama_profesi	BTREE	Yes	No	nama_profesi	5	A	Yes	



C. Jalanya Program Dan Penerapanya

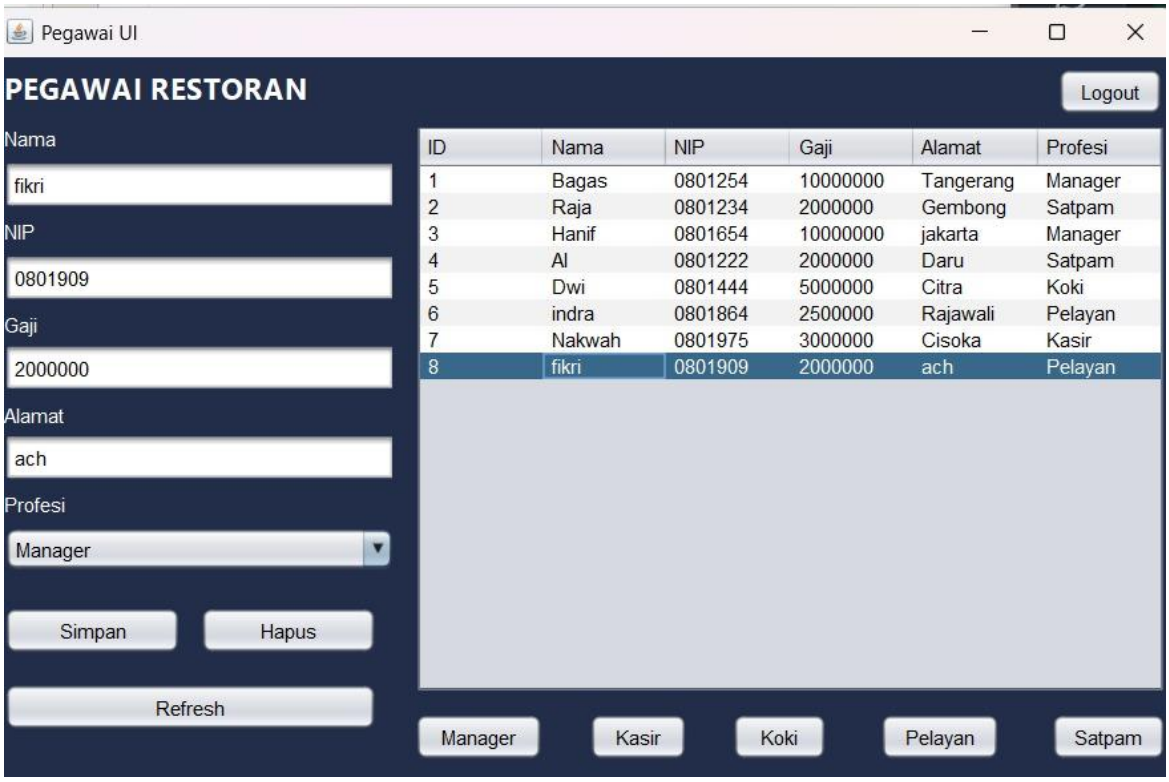
Pada tahapan awal pengguna diminta unutk memasukan username serta sandi unutk mengakses apliksi tersebut



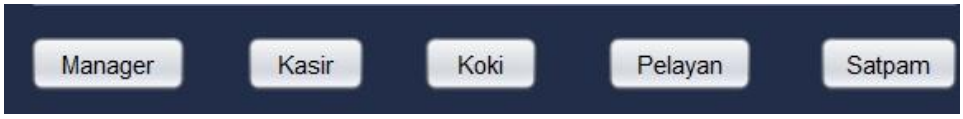
Lalu selanjutnya, dari menu login pengguna akan mengarah pada menu utama setelah setelah memasukan username dan password untuk login. Pada tampilan ini user menginputkan data. Lalu selanjutnya klik login untuk menyimpan data pada databasenya Sehingga data otomatis tersimpan.

LAPORAN PROJECT UAS (APLIKASI)

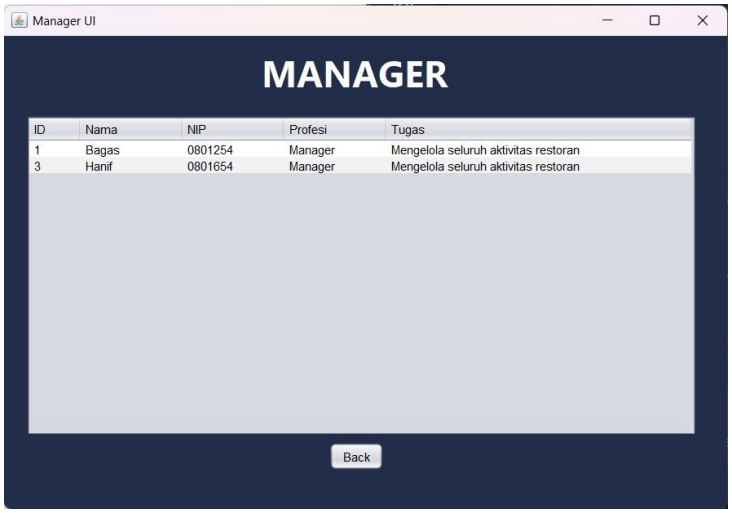
PEGAWAI RESTORANT



Pada tampilan ini pengguna diminta memasukan nama, nip, gaji alamat serta profesi. Pada menu profesi ini pengguna dapat memilih inggin memilih profesi yang ada terdapat pada menu seperti Manager, Satpam, Koki, Pelayan dan Kasir. Setelah pengguna menggisi data tersebut selanjutnya ada pilihan simpan yaitu untuk menyimpan data yang semula kita pilih atau hapus jika ingin merubah pilihan dan juga ada refresh. Jika sudah disimpan data akan muncul otomatis pada database

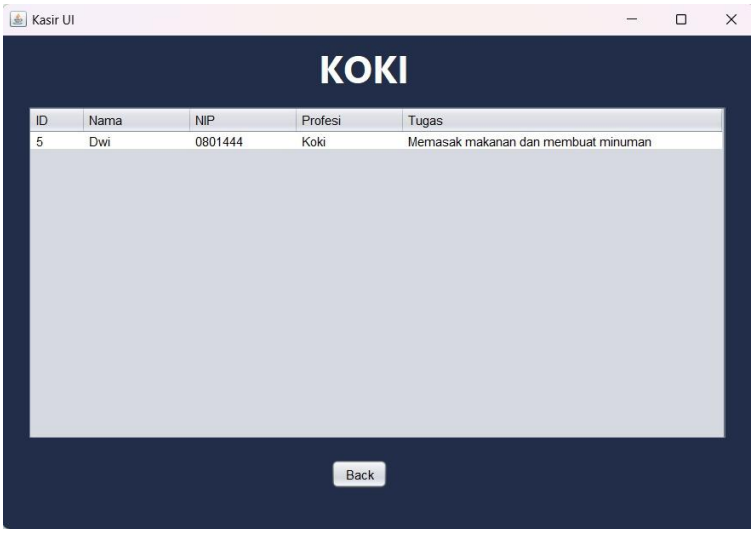


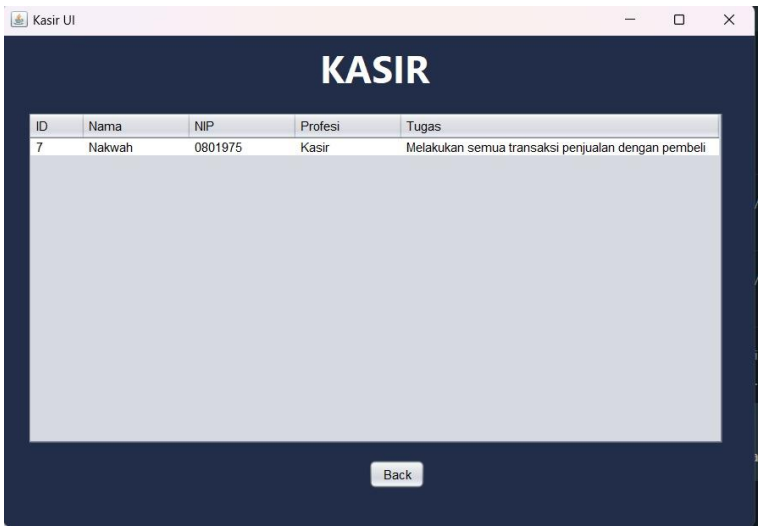
Jika pengguna memilih salah satu pilihan tersebut, maka data akan otomatis tersimpan sesuai dengan profesi yang dipilih oleh pengguna



LAPORAN PROJECT UAS (APLIKASI)

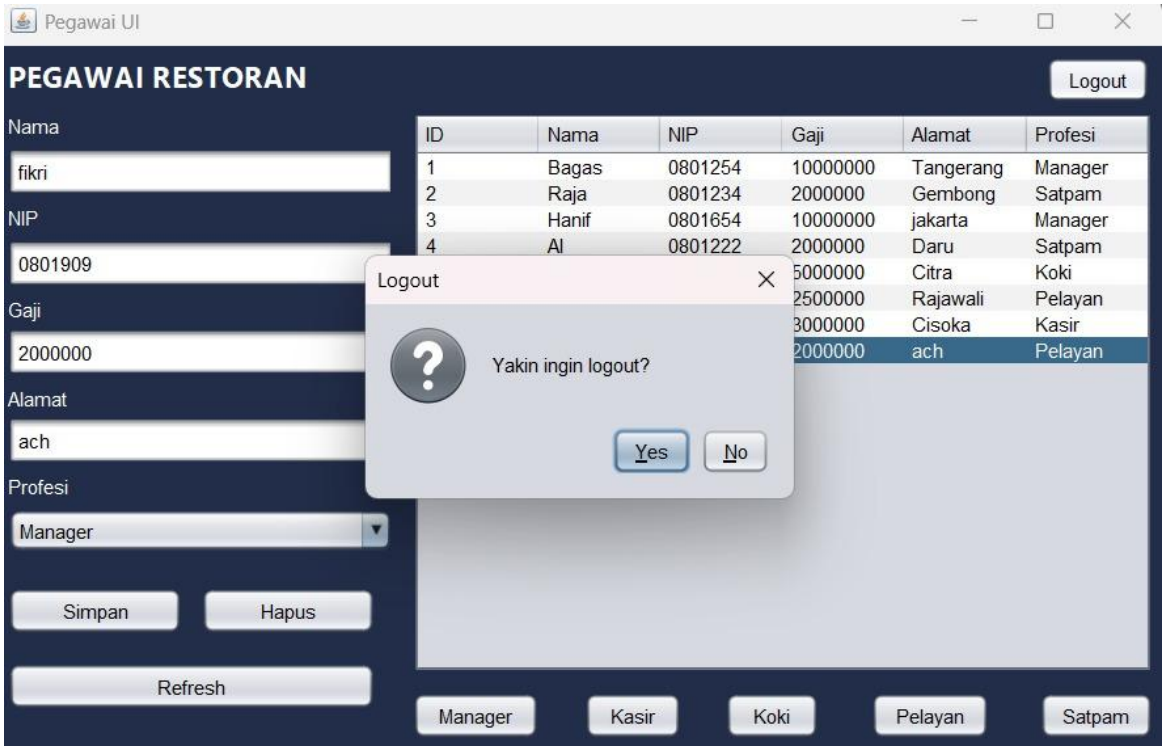
PEGAWAI RESTORANT





LAPORAN PROJECT UAS (APLIKASI)

PEGAWAI RESTORANT



Tampilan tersebut adalah menu Logout jika pengguna ingin keluar adri aplikasi

KESIMPULAN

Konsep OOP atau Objected Oriented Programming pada java ini, memiliki banyak sekali kegunaan untuk menciptakan sebuah program dari masalah kehidupan sehari-hari ini. Tidak salah memang bahwa banyak hal yang bisa diimplementasikan dari kehidupan sehari hari dengan OOP java ini. Berdasarkan hasil uji coba, aplikasi berhasil menyimpan data dengan baik sesuai yang dinginkan yaitu nama, nip, gaji, alamat dan profesi. Sehingga dapat disimpulkan bahwa aplikasi pembukuan keuangan penjual jus berbasis java ini layak untuk digunakan. Program sistem pembukuan ini dibuat dengan bahasa pemrograman java dan sistem OOP sehingga mudah dikembangkan untuk fitur dan menu lainnya

SARAN

Pada penelitian berikutnya disarankan untuk dapat menggunakan metode pengembangan lain yang lebih detail sehingga dalam perencanaan sistem dapat lebih terinci.