**LAPORAN TUGAS BESAR**

**PEMROGRAMAN BERORIENTASI OBJEK**

**SISTEM PEGAWAI**

# HALAMAN JUDUL



**Disusun Oleh :**

**NAMA : ARFIANA MAULIDIYAH**

**NIM : 32602200012**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS TEKNOLOGI INDUSTRI**

**UNIVERSITAS ISLAM SULTAN AGUNG**

**SEMARANG**

**2022**

# DAFTAR ISI

[HALAMAN JUDUL 1](#_Toc154989973)

[DAFTAR ISI 2](#_Toc154989974)

[DAFTAR GAMBAR 3](#_Toc154989975)

[BAB I Pendahuluan 4](#_Toc154989976)

[**1.1** **Latar belakang** 4](#_Toc154989977)

[**1.2** **Tujuan** 4](#_Toc154989978)

[BAB II Konsep Dasar PBO 5](#_Toc154989979)

[**2.1** **Inheritance** 5](#_Toc154989980)

[**2.2** **Polimorfisme** 5](#_Toc154989981)

[**2.3** **Encapsulation** 5](#_Toc154989982)

[**2.4** **Getter dan Setter** 5](#_Toc154989983)

[**2.5** **Interface** 5](#_Toc154989984)

[BAB III Struktur Program 6](#_Toc154989985)

[**3.1** **File Utama ( ManajemenPegawaiGUI )** 6](#_Toc154989986)

[**3.2** **Kelas Pegawai** 9](#_Toc154989987)

[**3.3** **Kelas Manager dan Staff** 10](#_Toc154989988)

[**3.4** **Interface PengelolaPegawai** 13](#_Toc154989989)

[**3.5** **Kelas ManajemenPegawaiGUI ( Main Class )** 14](#_Toc154989990)

[BAB IV Implementasi Program 18](#_Toc154989991)

[**4.1** **Menjalankan Program** 18](#_Toc154989992)

[**4.2** **Menambahkan Produk** 19](#_Toc154989993)

[BAB V Kesimpulan 22](#_Toc154989994)

[DAFTAR PUSTAKA 24](#_Toc154989995)

# DAFTAR GAMBAR

[Gambar 1. 1 File Utama Program Main 1 6](#_Toc154989800)

[Gambar 1. 2 File Utama Program Main 2 6](#_Toc154989801)

[Gambar 1. 3 File Utama Program Main 3 7](#_Toc154989802)

[Gambar 1. 4 Kelas Pegawai 1 9](#_Toc154989803)

[Gambar 1. 5 Kelas Pegawai 2 9](#_Toc154989804)

[Gambar 1. 6 Kelas Manager 1 11](#_Toc154989805)

[Gambar 1. 7 Kelas Manager 2 11](#_Toc154989806)

[Gambar 1. 8 Kelas Staff 1 12](#_Toc154989807)

[Gambar 1. 9 Kelas Staff 2 12](#_Toc154989808)

[Gambar 1. 10 Interface Pengelola Pegawai 13](#_Toc154989809)

[Gambar 1. 11 MainPegawaiGUI 1 14](#_Toc154989810)

[Gambar 1. 12 MainPegawaiGUI 2 15](#_Toc154989811)

[Gambar 1. 13 MainPegawaiGUI 3 15](#_Toc154989812)

[Gambar 1. 14 MainPegawaiGUI 4 15](#_Toc154989813)

[Gambar 2. 1 Open Neatbeans 18](#_Toc154989814)

[Gambar 2. 2 Open Projects 18](#_Toc154989815)

[Gambar 2. 3 Arahkan ke project 18](#_Toc154989816)

[Gambar 2. 4 Open struktur project 19](#_Toc154989817)

[Gambar 2. 5 Run program 19](#_Toc154989818)

[Gambar 2. 6 Tunggu program muncul 19](#_Toc154989819)

[Gambar 2. 7 Menu Pegawai 19](#_Toc154989820)

[Gambar 2. 8 Menu Staff / Manager 20](#_Toc154989821)

[Gambar 2. 9 Isi Nama Pegawai 20](#_Toc154989822)

[Gambar 2. 10 Dialog sukses add pegawai 20](#_Toc154989823)

[Gambar 2. 11 Back to menu main 20](#_Toc154989824)

[Gambar 2. 12 Dialog show pegawai 21](#_Toc154989825)

[Gambar 2. 13 Back to menu main 21](#_Toc154989826)

# BAB I Pendahuluan

## **Latar belakang**

Pengelolaan pegawai merupakan aspek krusial dalam suatu organisasi. Dalam rangka meningkatkan efisiensi dan keteraturan data pegawai, sistem manajemen pegawai berbasis pemrograman berorientasi objek (PBO) menjadi solusi yang tepat. Pada laporan ini, akan dibahas implementasi sistem manajemen pegawai dengan menggunakan konsep dasar PBO.

## **Tujuan**

Tujuan utama dari laporan ini adalah memberikan pemahaman mengenai konsep dasar PBO yang diterapkan dalam pengelolaan pegawai. Selain itu, laporan ini bertujuan untuk memberikan panduan langkah-langkah dalam menjalankan program dan menambahkan data pegawai.

# BAB II Konsep Dasar PBO

## **Inheritance**

Konsep inheritance diterapkan dalam pembuatan kelas Pegawai, Manager, dan Staff. Kelas Manager dan Staff merupakan turunan dari kelas Pegawai, mewarisi atribut dan metode dari kelas induknya.

## **Polimorfisme**

Polimorfisme diimplementasikan melalui penggunaan metode info() pada kelas Pegawai, yang kemudian di-override oleh kelas Manager dan Staff sesuai dengan kebutuhan masing-masing.

## **Encapsulation**

Encapsulation diaplikasikan dengan menyembunyikan implementasi internal kelas Pegawai dan memberikan akses melalui metode getter dan setter.

## **Getter dan Setter**

Metode getter dan setter digunakan untuk mengakses dan mengubah nilai atribut pada kelas Pegawai, Manager, dan Staff dengan prinsip encapsulation.

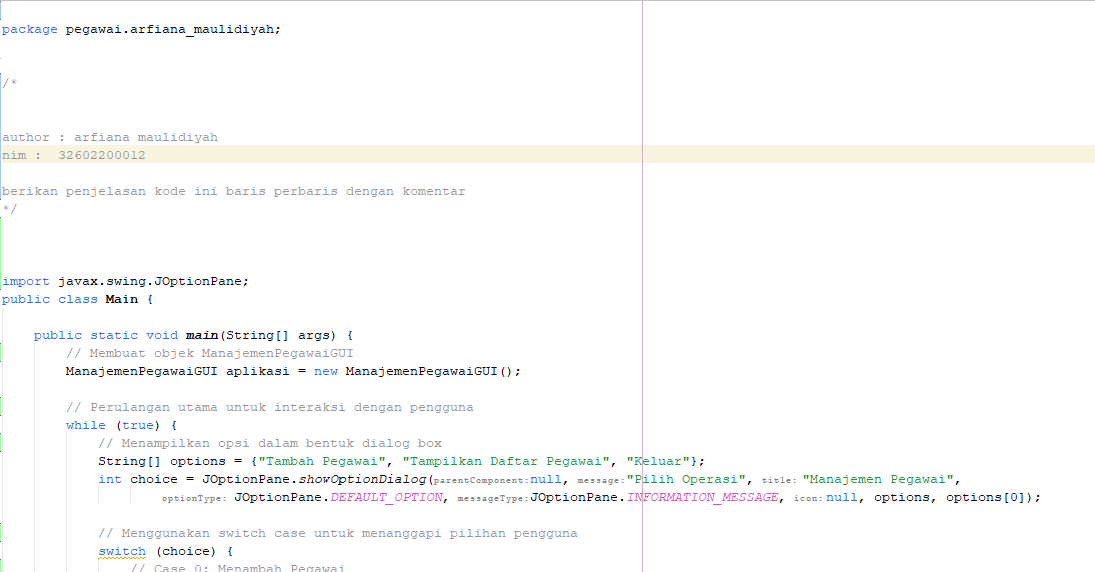
## **Interface**

Interface PengelolaPegawai diimplementasikan pada kelas ManajemenPegawaiGUI untuk memastikan bahwa metode untuk menambah pegawai dan menampilkan daftar pegawai telah diimplementasikan.

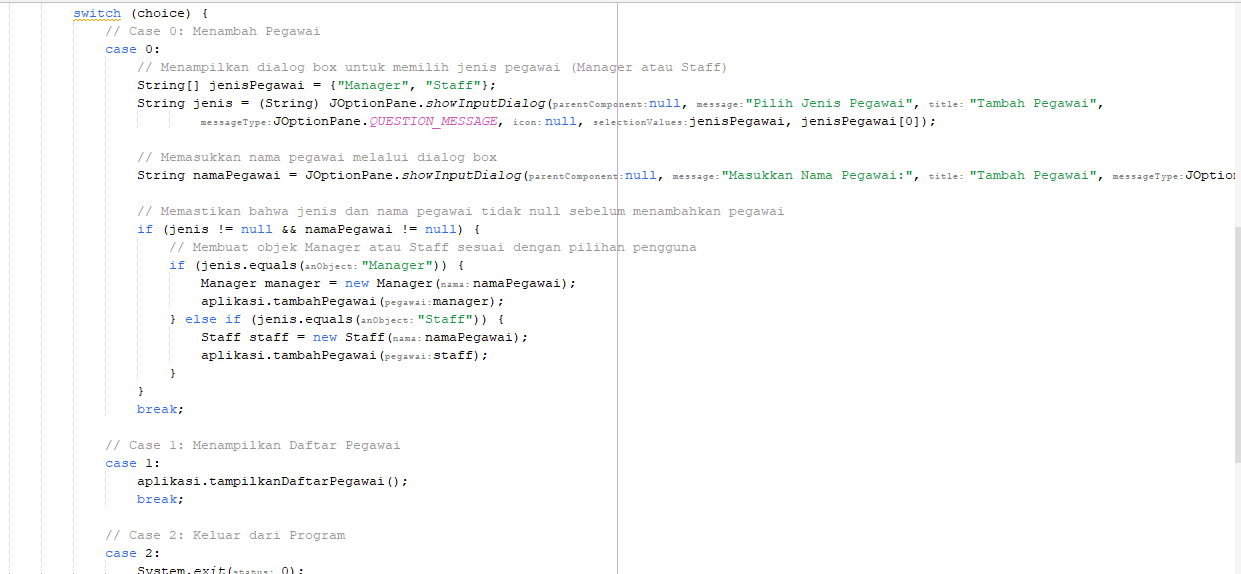
# BAB III Struktur Program

## **File Utama ( ManajemenPegawaiGUI )**

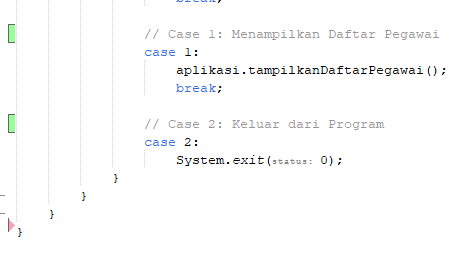
ManajemenPegawaiGUI merupakan kelas utama yang berfungsi sebagai antarmuka pengguna. Dalam kelas ini, user dapat menambahkan pegawai dan menampilkan daftar pegawai..



Gambar 1. 1 File Utama Program Main 1



Gambar 1. 2 File Utama Program Main 2



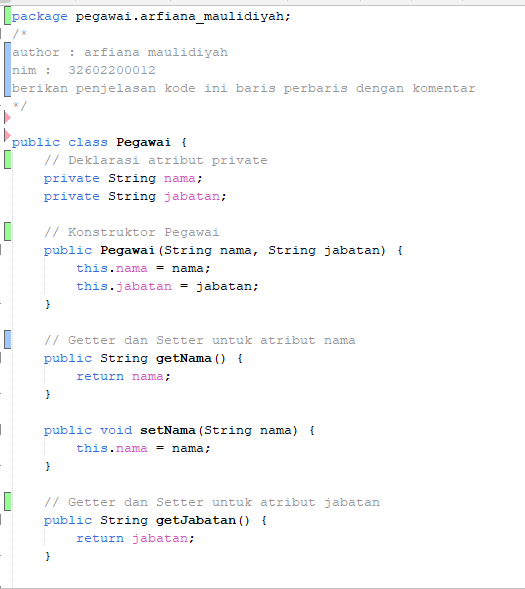
Gambar 1. 3 File Utama Program Main 3

Penjelasan Kode :

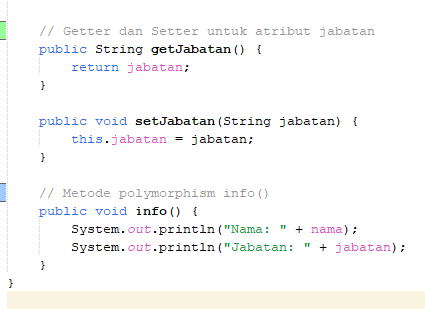
1. Package dan Import
2. package pegawai.arfiana\_maulidiyah;: Mendefinisikan package untuk kelas-kelas dalam program.
3. import javax.swing.JOptionPane;: Mengimpor kelas JOptionPane dari paket javax.swing untuk penggunaan dialog box.
4. Deklarasi Kelas Main
5. public class Main {: Mendeklarasikan kelas utama dengan nama Main.
6. Metode main
7. public static void main(String[] args) {: Mendeklarasikan metode main sebagai titik masuk utama program.
8. Inisialisasi Objek ManajemenPegawaiGUI
9. ManajemenPegawaiGUI aplikasi = new ManajemenPegawaiGUI();: Membuat objek ManajemenPegawaiGUI untuk mengelola pegawai.
10. Perulangan Utama
11. while (true) {: Membuat perulangan tanpa batas untuk terus berinteraksi dengan pengguna.
12. Opsi Pilihan Pengguna
13. String[] options = {"Tambah Pegawai", "Tampilkan Daftar Pegawai", "Keluar"};: Membuat array string berisi opsi pilihan pengguna.
14. Dialog Box Pilihan Operasi
15. int choice = JOptionPane.showOptionDialog(...);: Menampilkan dialog box opsi dan mengambil pilihan pengguna.
16. Switch Case untuk Memproses Pilihan Pengguna
17. switch (choice) {: Memulai struktur kontrol switch case untuk menanggapi pilihan pengguna.
18. Case 0: Menambah Pegawai
19. case 0:: Bagian untuk menambah pegawai.
20. String[] jenisPegawai = {"Manager", "Staff"};: Membuat array string berisi opsi jenis pegawai.
21. String jenis = (String) JOptionPane.showInputDialog(...);: Menampilkan dialog box untuk memilih jenis pegawai.
22. String namaPegawai = JOptionPane.showInputDialog(...);: Menampilkan dialog box untuk memasukkan nama pegawai.
23. if (jenis != null && namaPegawai != null) {: Memastikan input jenis dan nama pegawai tidak null.
24. if (jenis.equals("Manager")) { ... } else if (jenis.equals("Staff")) { ... }: Membuat objek Manager atau Staff berdasarkan pilihan pengguna.
25. Case 1: Menampilkan Daftar Pegawai
26. case 1:: Bagian untuk menampilkan daftar pegawai.
27. aplikasi.tampilkanDaftarPegawai();: Memanggil metode untuk menampilkan daftar pegawai.
28. Case 2: Keluar dari Program
29. case 2:: Bagian untuk keluar dari program.
30. System.exit(0);: Mengakhiri program.
31. Penutup Metode main
32. } // Akhir switch case: Menutup blok switch case.
33. } // Akhir while loop: Menutup blok while loop.
34. } // Akhir main method: Menutup blok main method.
35. } // Akhir kelas Main: Menutup blok kelas Main.

## **Kelas Pegawai**

Kelas Pegawai sebagai superclass berisi atribut dan metode umum yang digunakan oleh kelas turunannya, yaitu Manager dan Staff.



Gambar 1. 4 Kelas Pegawai 1



Gambar 1. 5 Kelas Pegawai 2

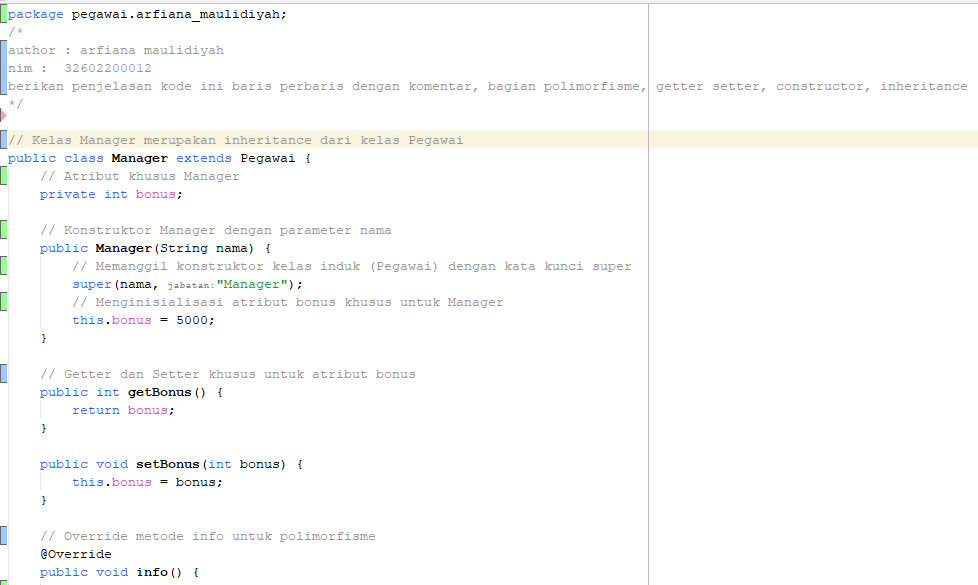
Penjelasan Kode :

1. Deklarasi Package dan Komentar Header
2. package pegawai.arfiana\_maulidiyah;: Mendefinisikan package tempat kelas Pegawai berada.
3. Komentar berisi informasi tentang penulis dan NIM.
4. Deklarasi Kelas Pegawai
   1. public class Pegawai {: Mendeklarasikan kelas Pegawai.
5. Atribut Kelas Pegawai
   1. private String nama;: Atribut private untuk menyimpan nama pegawai.
   2. private String jabatan;: Atribut private untuk menyimpan jabatan pegawai.
6. Konstruktor Kelas Pegawai
   1. public Pegawai(String nama, String jabatan) {: Konstruktor untuk inisialisasi objek Pegawai dengan nama dan jabatan.
7. Getter dan Setter
   1. public String getNama() { return nama; }: Getter untuk mendapatkan nilai atribut nama.
   2. public void setNama(String nama) { this.nama = nama; }: Setter untuk mengubah nilai atribut nama.
   3. public String getJabatan() { return jabatan; }: Getter untuk mendapatkan nilai atribut jabatan.
   4. public void setJabatan(String jabatan) { this.jabatan = jabatan; }: Setter untuk mengubah nilai atribut jabatan.
8. Metode Polymorphism info()

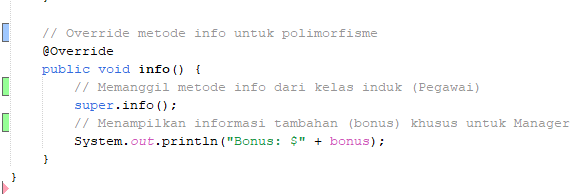
a. public void info() { ... }: Metode polymorphism yang menampilkan informasi nama dan jabatan pegawai.

## **Kelas Manager dan Staff**

Kelas Manager dan Staff sebagai subclass dari Pegawai, mewarisi atribut dan metode dari kelas Pegawai dan menambahkan atribut khusus masing-masing..



Gambar 1. 6 Kelas Manager 1

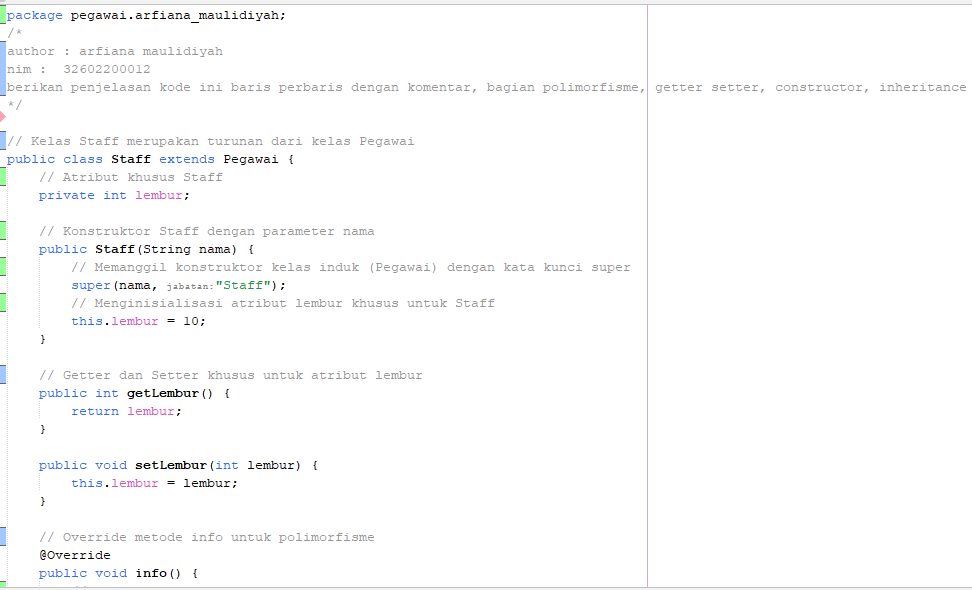


Gambar 1. 7 Kelas Manager 2

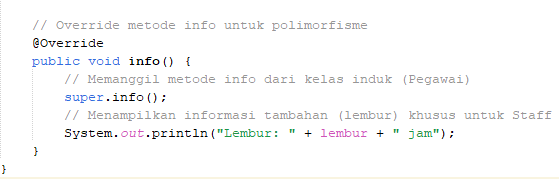
Penjelasan Kode :

* 1. Package dan Komentar Header

1. Komentar berisi informasi tentang penulis dan NIM.
   1. Deklarasi Kelas Manager dan Inheritance
2. public class Manager extends Pegawai {: Mendeklarasikan kelas Manager yang merupakan turunan dari kelas Pegawai.
   1. Atribut Kelas Manager
3. private int bonus;: Atribut khusus untuk Manager.
   1. Konstruktor Kelas Manager
4. public Manager(String nama) { ... }: Konstruktor Manager dengan parameter nama.
5. super(nama, "Manager");: Memanggil konstruktor kelas induk (Pegawai) dengan kata kunci super.
6. this.bonus = 5000;: Menginisialisasi atribut bonus khusus untuk Manager.
   1. Getter dan Setter untuk Atribut Bonus
7. public int getBonus() { return bonus; }: Getter untuk mendapatkan nilai atribut bonus.
8. public void setBonus(int bonus) { this.bonus = bonus; }: Setter untuk mengubah nilai atribut bonus.
   1. Override Metode info() untuk Polimorfisme
9. @Override: Menandakan bahwa metode berikut ini di-override dari kelas induk.
10. super.info();: Memanggil metode info dari kelas induk (Pegawai).
11. System.out.println("Bonus: $" + bonus);: Menampilkan informasi tambahan (bonus) khusus untuk Manager.



Gambar 1. 8 Kelas Staff 1



Gambar 1. 9 Kelas Staff 2

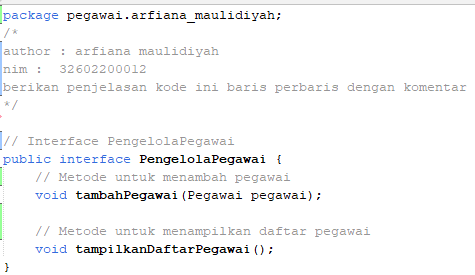
Penjelasan Kode :

* 1. Package dan Komentar Header

1. Komentar berisi informasi tentang penulis dan NIM.
   1. Deklarasi Kelas Staff dan Inheritance
2. public class Staff extends Pegawai {: Mendeklarasikan kelas Staff yang merupakan turunan dari kelas Pegawai.
   1. Atribut Kelas Staff
3. private int lembur;: Atribut khusus untuk Staff.
   1. Konstruktor Kelas Staff
4. public Staff(String nama) { ... }: Konstruktor Staff dengan parameter nama.
5. super(nama, "Staff");: Memanggil konstruktor kelas induk (Pegawai) dengan kata kunci super.
6. this.lembur = 10;: Menginisialisasi atribut lembur khusus untuk Staff.
   1. Getter dan Setter untuk Atribut Lembur
7. public int getLembur() { return lembur; }: Getter untuk mendapatkan nilai atribut lembur.
8. public void setLembur(int lembur) { this.lembur = lembur; }: Setter untuk mengubah nilai atribut lembur.
   1. Override Metode info() untuk Polimorfisme
9. @Override: Menandakan bahwa metode berikut ini di-override dari kelas induk.
10. super.info();: Memanggil metode info dari kelas induk (Pegawai).
11. System.out.println("Lembur: " + lembur + " jam");: Menampilkan informasi tambahan (lembur) khusus untuk Staff.

## **Interface PengelolaPegawai**

Interface yang memuat metode untuk menambah pegawai dan menampilkan daftar pegawai..



Gambar 1. 10 Interface Pengelola Pegawai

Penjelasan Kode :

1. Package dan Komentar Header

Komentar berisi informasi tentang penulis dan NIM.

1. Deklarasi Interface PengelolaPegawai

public interface PengelolaPegawai {: Mendeklarasikan sebuah interface dengan nama PengelolaPegawai.

1. Metode tambahPegawai

void tambahPegawai(Pegawai pegawai);: Metode untuk menambahkan objek Pegawai ke dalam suatu implementasi.

1. Metode tampilkanDaftarPegawai

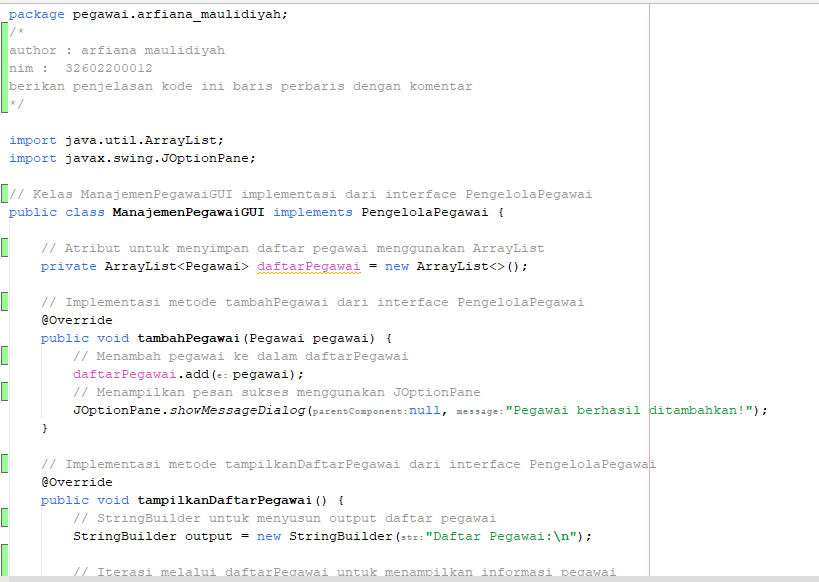
void tampilkanDaftarPegawai();: Metode untuk menampilkan daftar pegawai dalam suatu implementasi.

1. Penutup Interface

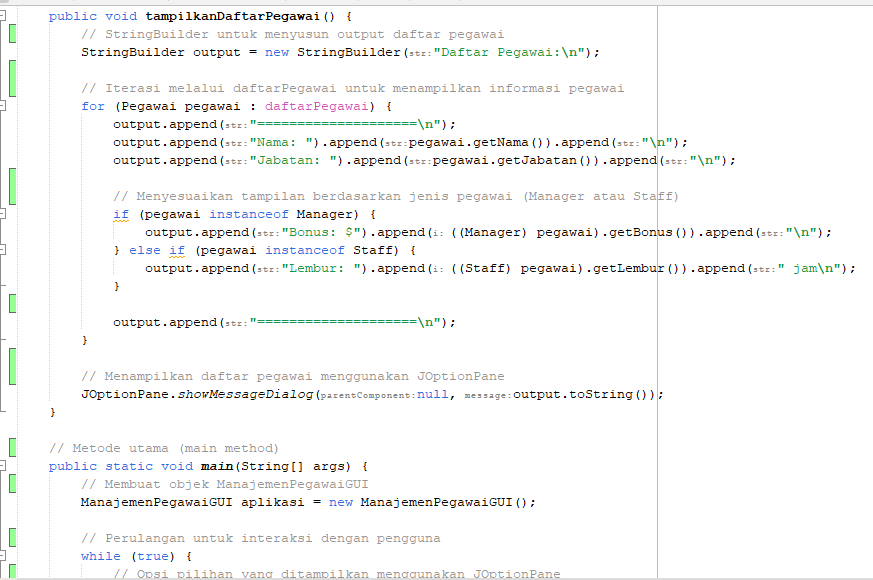
}: Menutup deklarasi interface PengelolaPegawai.

## **Kelas ManajemenPegawaiGUI ( Main Class )**

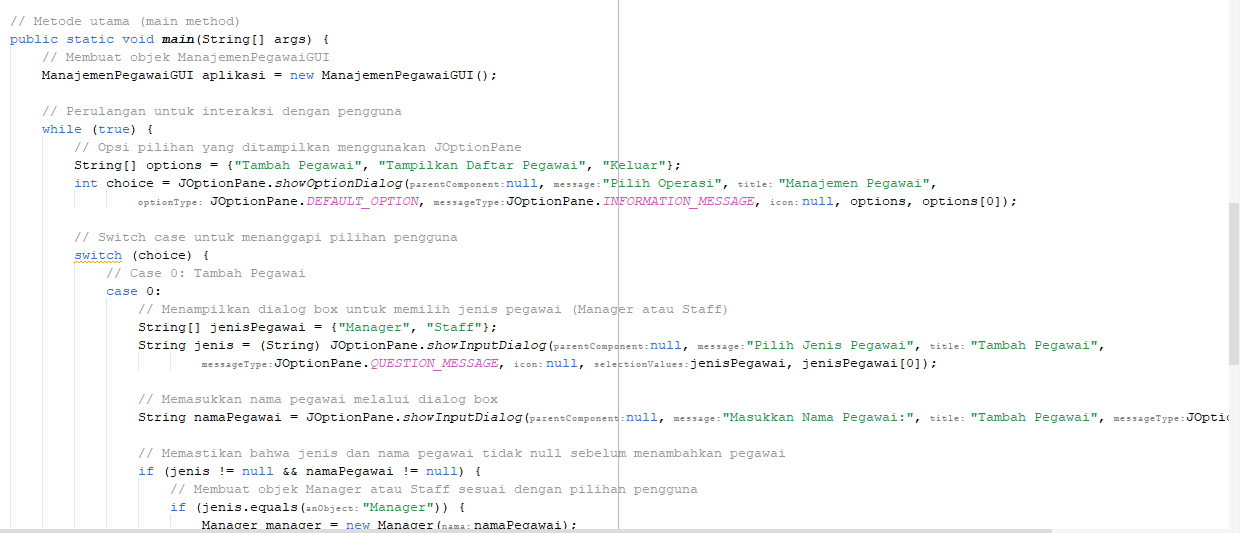
Kelas ini berfungsi sebagai antarmuka utama yang mengintegrasikan seluruh sistem manajemen pegawai. Pengguna dapat memilih operasi untuk menambah pegawai atau menampilkan daftar pegawai..



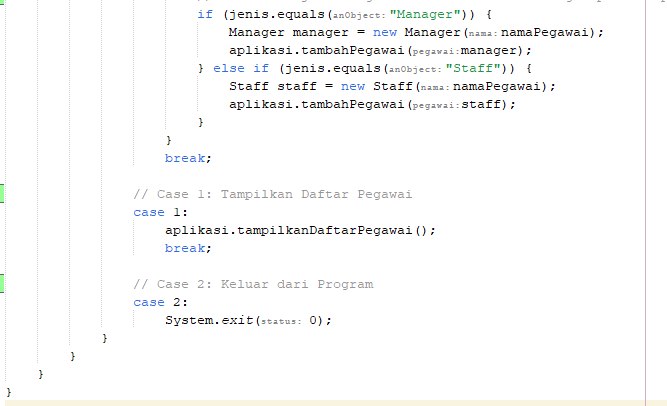
Gambar 1. 11 MainPegawaiGUI 1



Gambar 1. 12 MainPegawaiGUI 2



Gambar 1. 13 MainPegawaiGUI 3



Gambar 1. 14 MainPegawaiGUI 4

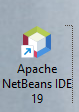
Penjelasan Kode :

1. Package dan Komentar Header
   1. Komentar berisi informasi tentang penulis dan NIM.
2. Import Paket
3. import java.util.ArrayList;: Mengimpor kelas ArrayList dari paket java.util.
4. import javax.swing.JOptionPane;: Mengimpor kelas JOptionPane dari paket javax.swing.
5. Deklarasi Kelas ManajemenPegawaiGUI
6. public class ManajemenPegawaiGUI implements PengelolaPegawai {: Mendeklarasikan kelas ManajemenPegawaiGUI yang mengimplementasi interface PengelolaPegawai.
7. Atribut Kelas ManajemenPegawaiGUI
   1. private ArrayList<Pegawai> daftarPegawai = new ArrayList<>();: Membuat ArrayList untuk menyimpan daftar pegawai.
8. Implementasi Metode dari Interface
9. @Override: Menandakan bahwa metode berikut diimplementasi dari interface.
10. void tambahPegawai(Pegawai pegawai) { ... }: Menambahkan pegawai ke dalam daftar dan menampilkan pesan dengan JOptionPane.
11. void tampilkanDaftarPegawai() { ... }: Menampilkan daftar pegawai dengan informasi sesuai jenis pegawai menggunakan JOptionPane.
12. Metode Main
13. public static void main(String[] args) { ... }: Metode utama untuk menjalankan aplikasi.
14. ManajemenPegawaiGUI aplikasi = new ManajemenPegawaiGUI();: Membuat objek ManajemenPegawaiGUI.
15. Perulangan while (true) untuk terus berinteraksi dengan pengguna.
16. String[] options = {"Tambah Pegawai", "Tampilkan Daftar Pegawai", "Keluar"};: Opsi pilihan yang ditampilkan menggunakan JOptionPane.
17. int choice = JOptionPane.showOptionDialog(...);: Meminta pengguna untuk memilih operasi menggunakan JOptionPane.
18. Switch case untuk menanggapi pilihan pengguna.
19. Case 0: Menambah Pegawai
20. String[] jenisPegawai = {"Manager", "Staff"};: Menampilkan dialog box untuk memilih jenis pegawai.
21. if (jenis.equals("Manager")) { ... } else if (jenis.equals("Staff")) { ... }: Membuat objek Manager atau Staff sesuai dengan pilihan pengguna.
22. Case 1: Menampilkan Daftar Pegawai
23. aplikasi.tampilkanDaftarPegawai();: Memanggil metode tampilkanDaftarPegawai() untuk menampilkan daftar pegawai.
24. Case 2: Keluar dari Program
25. System.exit(0);: Keluar dari program jika pengguna memilih opsi keluar.

# BAB IV Implementasi Program

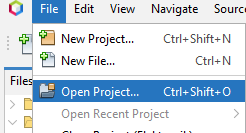
## **Menjalankan Program**

1. Untuk menjalankan program klick neatbeans

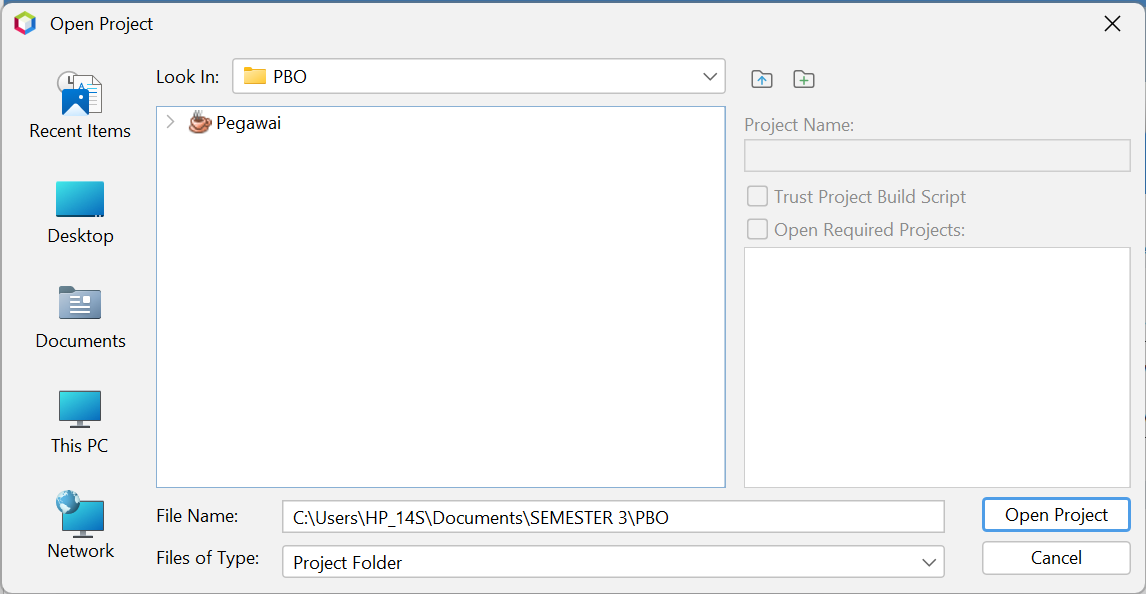


Gambar 2. 1 Open Neatbeans

1. Open new project, arahkan ke pegawai

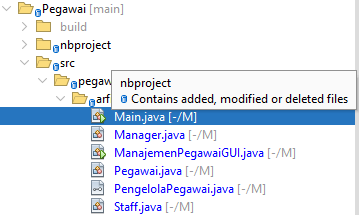


Gambar 2. 2 Open Projects



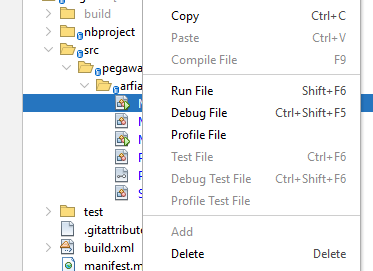
Gambar 2. 3 Arahkan ke project

1. Buka src/pegawai/arfiana\_maulidiyah



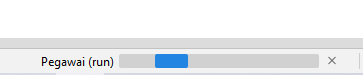
Gambar 2. 4 Open struktur project

1. Klick file Main.java, lalu klick kanan run file atau shift + f6



Gambar 2. 5 Run program

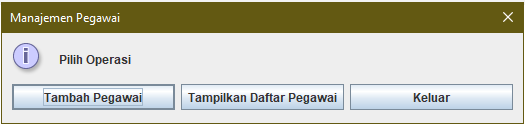
1. Tunggu program muncul



Gambar 2. 6 Tunggu program muncul

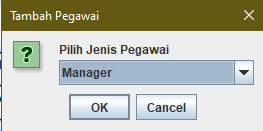
## **Menambahkan Pegawai**

1. Klick Tombol Tambah Pegawai



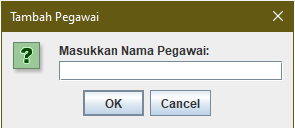
Gambar 2. 7 Menu Pegawai

1. Pilih Manajer / Staff, lalu klick ok



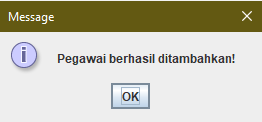
Gambar 2. 8 Menu Staff / Manager

1. Masukan isi nama pegawai, klick ok.



Gambar 2. 9 Isi Nama Pegawai

1. Lalu akan muncul dialog sukses berhasil menambahkan. Klick Ok

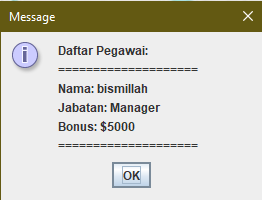


Gambar 2. 10 Dialog sukses add pegawai

1. Akan Kembali ke menu awal, klick tampilkan daftar pegawai, Jika diklick ok, maka akan Kembali menu awal.



Gambar 2. 11 Back to menu main



Gambar 2. 12 Dialog show pegawai

1. Klick keluar, program selesai berhenti



Gambar 2. 13 Back to menu main

# BAB V Kesimpulan

Dari kode yang diberikan, kita dapat menyimpulkan beberapa hal:

* 1. Struktur Kode:

1. Kode terstruktur dengan baik, menggunakan konsep OOP (Object-Oriented Programming) dengan inheritance dan interface.
2. Dibuat hierarki kelas Pegawai, Manager, dan Staff yang menggunakan konsep inheritance.
3. Menggunakan interface PengelolaPegawai untuk memastikan implementasi metode tertentu pada kelas ManajemenPegawaiGUI.
   1. Polimorfisme:
4. Menerapkan polimorfisme dengan menggunakan metode info() yang di-override pada kelas Manager dan Staff.
5. Polimorfisme memungkinkan pemanggilan metode info() dari objek Pegawai, tanpa harus mengetahui jenis sebenarnya (Manager atau Staff).
   1. Encapsulation:
6. Menggunakan encapsulation dengan mendeklarasikan atribut kelas sebagai private dan menyediakan getter dan setter untuk mengakses dan mengubah nilai atribut tersebut.
   1. Getter dan Setter:
7. Setiap kelas (Pegawai, Manager, Staff) memiliki getter dan setter untuk mengakses dan mengubah nilai atribut yang bersifat private.
   1. Inheritance:
8. Memanfaatkan inheritance untuk membuat kelas Manager dan Staff yang merupakan turunan dari kelas Pegawai.
9. Inheritance memungkinkan kelas turunan untuk mewarisi sifat dan perilaku kelas induknya.
   1. Antarmuka Pengguna (GUI):
10. Menerapkan antarmuka pengguna sederhana menggunakan JOptionPane untuk berinteraksi dengan pengguna.
11. Menyediakan opsi untuk menambah pegawai, menampilkan daftar pegawai, dan keluar dari program.
    1. Dinamika Program:
12. Program memiliki dinamika interaktif di mana pengguna dapat memilih operasi yang ingin dilakukan.
13. Pengguna dapat menambahkan Manager atau Staff, dan melihat daftar pegawai yang telah ditambahkan.

Dengan demikian, keseluruhan implementasi menunjukkan penggunaan konsep-konsep dasar Pemrograman Berorientasi Objek (OOP) dengan baik, seperti inheritance, polimorfisme, encapsulation, getter dan setter, serta antarmuka pengguna (GUI).

# DAFTAR PUSTAKA

(Czajkowski and Von Eicken, 1998; Lumpe, Mahmud and Vasa, 2010; Ari Yuana, S.Si, M.Kom, 2015)

Ari Yuana, S.Si, M.Kom, R. (2015) ‘Pemrograman Java’, *Informatika*, pp. 1–77. Available at: https://d1wqtxts1xzle7.cloudfront.net/33519053/JavaIndo.pdf?1398090002=&response-content-disposition=inline%3B+filename%3DDitulis\_oleh.pdf&Expires=1644744989&Signature=hOHnqB2lSAg3O37OjxRlo8mzhmE2pUyaB6R8UbZ75XMmwTHll9mbtKfvvdS4S7MgZTY9R46hVzgrAIej4uG5YJK.

Czajkowski, G. and Von Eicken, T. (1998) ‘JRes: A Resource Accounting Interface for Java’, *SIGPLAN Notices (ACM Special Interest Group on Programming Languages)*, 33(10), pp. 21–35. Available at: https://doi.org/10.1145/286942.286944.

Lumpe, M., Mahmud, S. and Vasa, R. (2010) ‘On the use of properties in Java applications’, *Proceedings of the Australian Software Engineering Conference, ASWEC*, pp. 235–244. Available at: https://doi.org/10.1109/ASWEC.2010.35.