**LAPORAN TUGAS BESAR**

**PEMROGRAMAN BERORIENTASI OBJEK**

**SISTEM TOKO ELEKTRONIK**

# HALAMAN JUDUL



**Disusun Oleh :**

**NAMA : AVENIA ROHMATUN NISA**

**NIM : 32602200047**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS TEKNOLOGI INDUSTRI**

**UNIVERSITAS ISLAM SULTAN AGUNG**

**SEMARANG**

**2024**

# DAFTAR ISI

[HALAMAN JUDUL i](#_Toc154951902)

[DAFTAR ISI ii](#_Toc154951903)

[DAFTAR GAMBAR iii](#_Toc154951904)

[BAB I Pendahuluan 1](#_Toc154951905)

[1.1 Latar belakang 1](#_Toc154951906)

[1.2 Tujuan 1](#_Toc154951907)

[BAB II Konsep Dasar PBO 2](#_Toc154951908)

[2.1 Inheritance 2](#_Toc154951909)

[2.2 Polimorfisme 2](#_Toc154951910)

[2.3 Encapsulation 2](#_Toc154951911)

[2.4 Getter dan Setter 2](#_Toc154951912)

[2.5 Interface 2](#_Toc154951913)

[BAB III Struktur Program 3](#_Toc154951914)

[3.1 File Utama ( Toko Elektronik ) 3](#_Toc154951915)

[3.2 Kelas Produk 4](#_Toc154951916)

[3.3 Kelas Elektronik 5](#_Toc154951917)

[3.4 Interface Kategori 6](#_Toc154951918)

[3.5 Antarmuka utama ( MainForm ) 6](#_Toc154951919)

[3.6 Antarmuka tambah produk ( ElektronikForm ) 9](#_Toc154951920)

[BAB IV Implementasi Program 12](#_Toc154951921)

[4.1 Menjalankan Program 12](#_Toc154951922)

[4.2 Menambahkan Produk 13](#_Toc154951923)

[BAB V Kesimpulan 15](#_Toc154951924)

# DAFTAR GAMBAR

[Gambar 1. 1 File Utama Program Toko Elektronik 3](#_Toc154981526)

[Gambar 1. 2 Kelas Produk 1 4](#_Toc154981527)

[Gambar 1. 3 Kelas Produk 2 4](#_Toc154981528)

[Gambar 1. 4 Kelas Elektronik 1 5](#_Toc154981529)

[Gambar 1. 5 MainForm 1 7](#_Toc154981530)

[Gambar 1. 6 MainForm 2 7](#_Toc154981531)

[Gambar 1. 7 MainForm 3 7](#_Toc154981532)

[Gambar 1. 8 ElektronikForm 1 9](#_Toc154981533)

[Gambar 1. 9 ElektronikForm 2 9](#_Toc154981534)

[Gambar 1. 10 ElektronikForm 3 10](#_Toc154981535)

[Gambar 1. 11 ElektronikForm 4 10](#_Toc154981536)

[Gambar 2. 1 Open Neatbeans 12](#_Toc154981537)

[Gambar 2. 2 Open Projects 12](#_Toc154981538)

[Gambar 2. 3 Arahkan ke project 12](#_Toc154981539)

[Gambar 2. 4 Open struktur project 13](#_Toc154981540)

[Gambar 2. 5 Run program 13](#_Toc154981541)

[Gambar 2. 6 Tunggu program muncul 13](#_Toc154981542)

[Gambar 2. 7 Tambah Elektronik 13](#_Toc154981543)

[Gambar 2. 8 Isi Data, Klick Simpan 14](#_Toc154981544)

[Gambar 2. 9 Keluar Dialog Pesanan 14](#_Toc154981545)

# BAB I Pendahuluan

## **Latar belakang**

Dalam era teknologi modern ini, bisnis toko elektronik semakin berkembang pesat. Dengan banyaknya jenis produk elektronik yang beragam, manajemen toko elektronik menjadi semakin kompleks. Oleh karena itu, diperlukan suatu sistem yang mampu mempermudah pengelolaan dan manajemen produk di dalam toko tersebut.

Latar belakang dibuatnya program Sistem Manajemen Toko Elektronik ini adalah sebagai respons terhadap kebutuhan akan solusi yang efisien dalam mengelola produk elektronik. Seiring dengan kemajuan teknologi, diperlukan suatu sistem yang tidak hanya dapat menyimpan informasi produk dengan baik tetapi juga memanfaatkan konsep-konsep pemrograman berorientasi objek agar dapat diintegrasikan dengan baik ke dalam struktur bisnis toko elektronik.

## **Tujuan**

Tujuan utama dari pembuatan program ini adalah untuk memberikan solusi yang efisien dan mudah digunakan dalam manajemen produk pada sebuah toko elektronik. Dengan menerapkan konsep-konsep dasar pemrograman berorientasi objek, diharapkan program ini dapat memberikan fleksibilitas dan skalabilitas dalam mengelola produk, sekaligus memudahkan pengembangan sistem di masa depan.

Melalui penerapan konsep inheritance, polymorphism, encapsulation, getter dan setter, serta interface, diharapkan program ini dapat menjadi dasar yang kokoh untuk pengembangan lebih lanjut. Dengan kata lain, program ini dirancang not only for the present but also for the future, sehingga dapat terus berkembang sesuai dengan kebutuhan toko elektronik yang terus berubah

# BAB II Konsep Dasar PBO

## **Inheritance**

Konsep inheritance digunakan untuk menunjukkan hubungan hierarki antar kelas pada program. Pada program ini, terdapat kelas Produk sebagai kelas dasar yang diwarisi oleh kelas Elektronik.

## **Polimorfisme**

Polymorphism digunakan untuk memungkinkan suatu objek dapat memiliki banyak bentuk (tipe data) dengan cara yang konsisten. Pada program ini, polymorphism diimplementasikan melalui metode getInfo() pada kelas Produk yang dioverride oleh kelas Elektronik.

## **Encapsulation**

Encapsulation digunakan untuk menyembunyikan implementasi internal suatu objek dan hanya mengekspos fungsionalitas yang diperlukan. Pada program ini, atribut-atribut kelas seperti nama, harga, dan jenis dienkapsulasi dalam kelas Produk dan Elektronik.

## **Getter dan Setter**

Getter dan setter digunakan untuk mengakses dan mengubah nilai atribut suatu objek. Dalam program ini, metode getter dan setter diterapkan pada kelas Produk untuk mengakses dan mengubah nilai atribut.

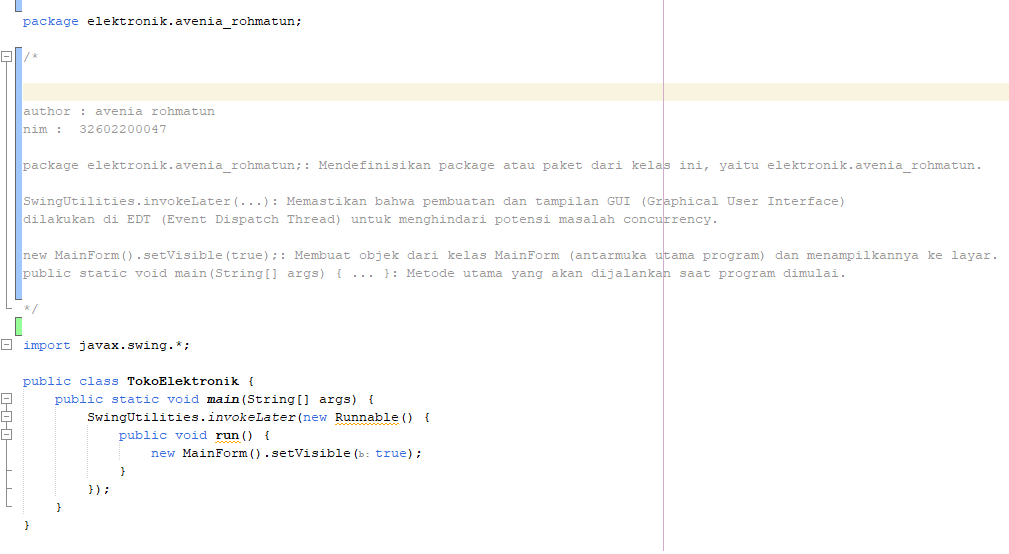
## **Interface**

Interface digunakan untuk menyediakan kontrak untuk kelas-kelas yang mengimplementasikannya. Pada program ini, terdapat interface Kategori yang diimplementasikan oleh kelas Elektronik

# BAB III Struktur Program

## **File Utama ( Toko Elektronik )**

File utama berisi kelas TokoElektronik yang merupakan entry point program dan menampilkan antarmuka utama.



Gambar 1. 1 File Utama Program Toko Elektronik

Penjelasan:

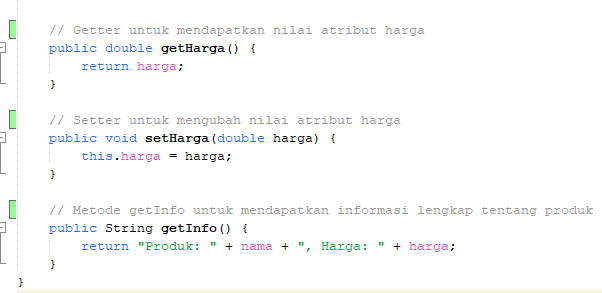
1. /\* ... \*/: Ini adalah komentar blok yang digunakan untuk memberikan penjelasan atau dokumentasi terhadap sejumlah baris kode.
2. package elektronik.avenia\_rohmatun;: Mendefinisikan package atau paket dari kelas ini, yaitu elektronik.avenia\_rohmatun.
3. Author : Avenia Rohmatun: Memberikan informasi tentang penulis atau pembuat program. Berguna untuk dokumentasi dan identifikasi.
4. NIM : 32602200047: Menyertakan NIM (Nomor Induk Mahasiswa) atau identifikasi lainnya. Berguna untuk identifikasi dan dokumentasi pribadi.
5. SwingUtilities.invokeLater(...): Memastikan bahwa pembuatan dan tampilan GUI (Graphical User Interface) dilakukan di EDT (Event Dispatch Thread) untuk menghindari potensi masalah concurrency.
6. new MainForm().setVisible(true);: Membuat objek dari kelas MainForm (antarmuka utama program) dan menampilkannya ke layar.
7. public static void main(String[] args) { ... }: Metode utama yang akan dijalankan saat program dimulai.

## **Kelas Produk**

Kelas Produk sebagai kelas dasar yang merepresentasikan produk dengan atribut nama dan harga. Lebih mudahnya adalah file kelas yang menampung kode tentang data informasi yang ditampilkan di program apa saja, baik kita atur ( set ) atau kita dapatkan ( get ) dari masing – masing atribut.



Gambar 1. 2 Kelas Produk 1



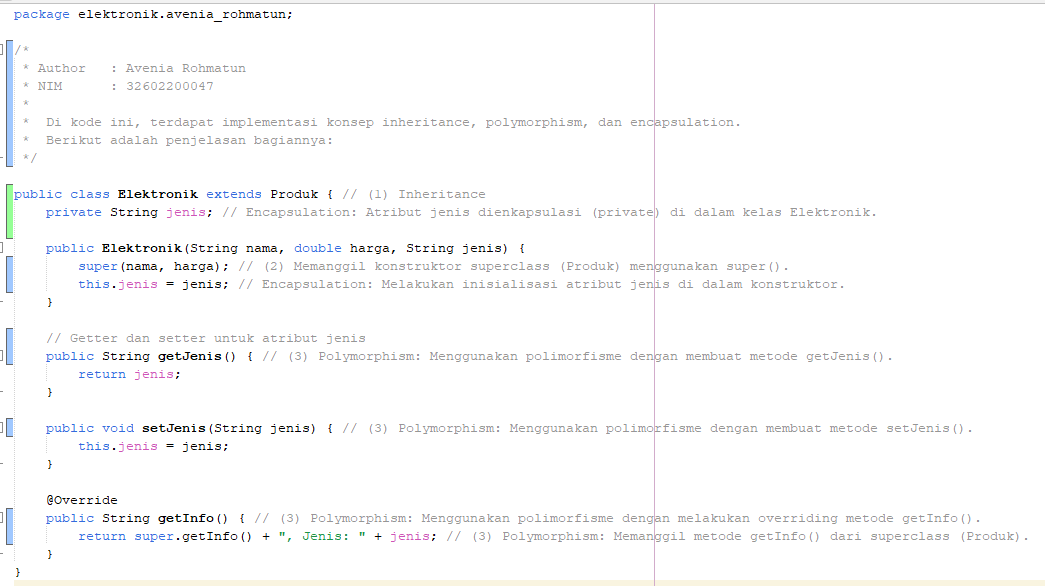
Gambar 1. 3 Kelas Produk 2

Penjelasan:

1. private String nama; dan private double harga;: Mendeklarasikan dua atribut privat (nama dan harga) untuk kelas Produk.
2. public Produk(String nama, double harga) { ... }: Konstruktor untuk kelas Produk yang menerima input nama dan harga saat pembuatan objek.
3. public String getNama() { return nama; }: Getter untuk mendapatkan nilai atribut nama.
4. public void setNama(String nama) { this.nama = nama; }: Setter untuk mengubah nilai atribut nama.
5. public double getHarga() { return harga; }: Getter untuk mendapatkan nilai atribut harga.
6. public void setHarga(double harga) { this.harga = harga; }: Setter untuk mengubah nilai atribut harga.
7. public String getInfo() { return "Produk: " + nama + ", Harga: " + harga; }: Metode getInfo yang memberikan informasi lengkap tentang produk, termasuk nama dan harga.

## **Kelas Elektronik**

Kelas Elektronik merupakan turunan dari kelas Produk dan mewakili produk elektronik dengan tambahan atribut jenis.



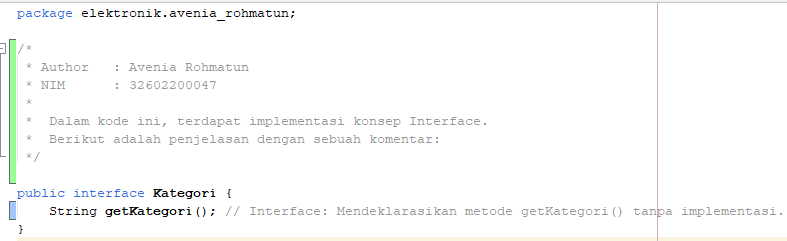
Gambar 1. 4 Kelas Elektronik 1

Penjelasan:

* 1. Inheritance: Kelas Elektronik meng-extend kelas Produk, sehingga mewarisi atribut dan metode dari kelas Produk.
  2. Memanggil konstruktor superclass (Produk) menggunakan super(): Dalam konstruktor Elektronik, menggunakan super(nama, harga) untuk memanggil konstruktor kelas Produk dan menginisialisasi atribut nama dan harga.
  3. Polymorphism: Implementasi polimorfisme dengan membuat metode getJenis(), setJenis(), dan melakukan overriding pada metode getInfo(). Polimorfisme memungkinkan pemanggilan metode dari superclass (dalam hal ini, Produk) melalui objek turunannya (Elektronik).

## **Interface Kategori**

Interface Kategori menyediakan kontrak untuk kelas-kelas yang ingin menentukan kategori produk.

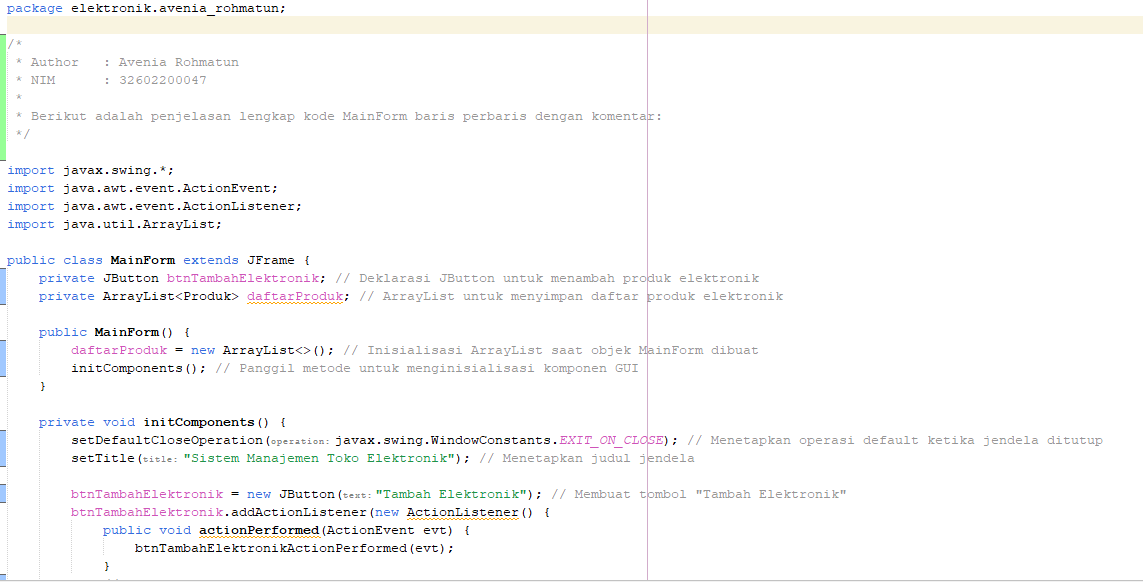


Penjelasan:

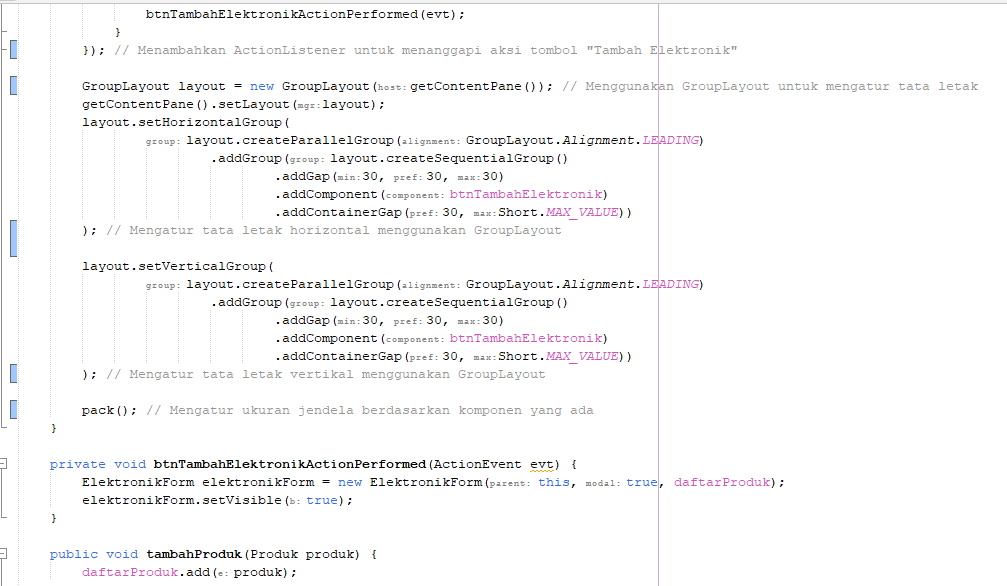
1. Interface: Merupakan sebuah kontrak yang mendeklarasikan metode tanpa memberikan implementasi. Interface digunakan untuk menentukan suatu kontrak yang harus dipenuhi oleh kelas-kelas yang mengimplementasikannya.
2. String getKategori();: Interface Kategori mendeklarasikan metode getKategori() yang harus diimplementasikan oleh kelas-kelas yang menggunakan interface ini. Kelas-kelas yang mengimplementasikan interface Kategori harus menyediakan implementasi konkret untuk metode getKategori().

## **Antarmuka utama ( MainForm )**

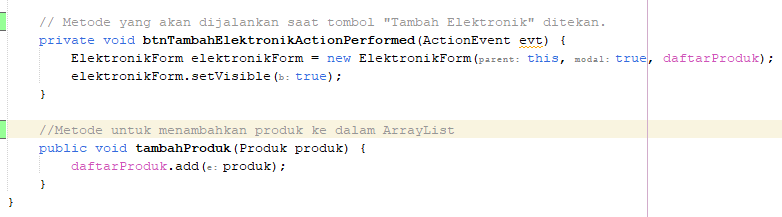
Antarmuka utama program yang memungkinkan pengguna untuk memilih operasi yang ingin dilakukan, seperti menambahkan produk elektronik.



Gambar 1. 5 MainForm 1



Gambar 1. 6 MainForm 2



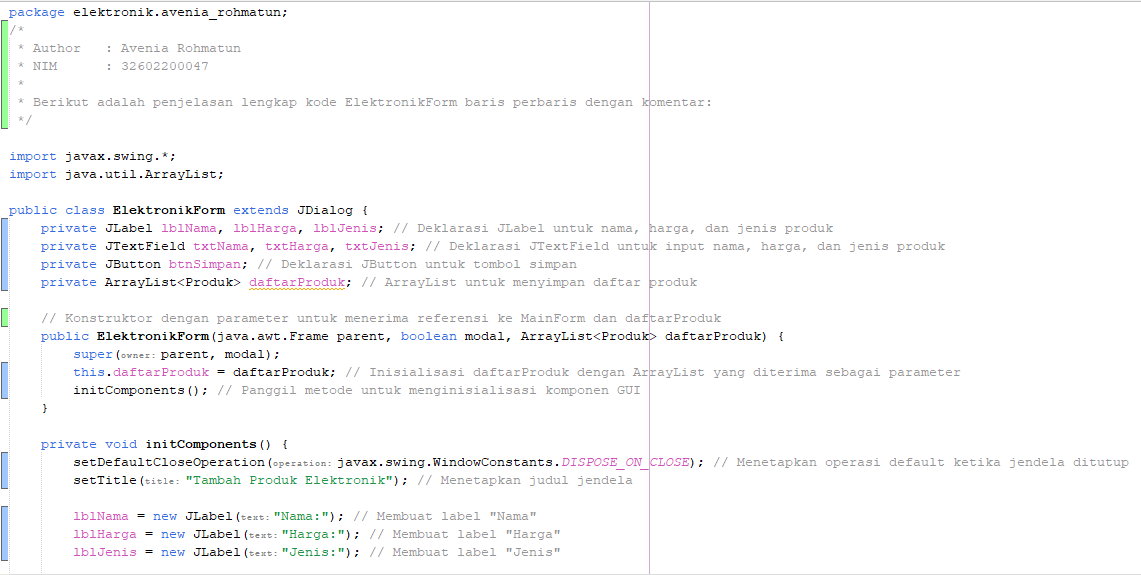
Gambar 1. 7 MainForm 3

Penjelasan:

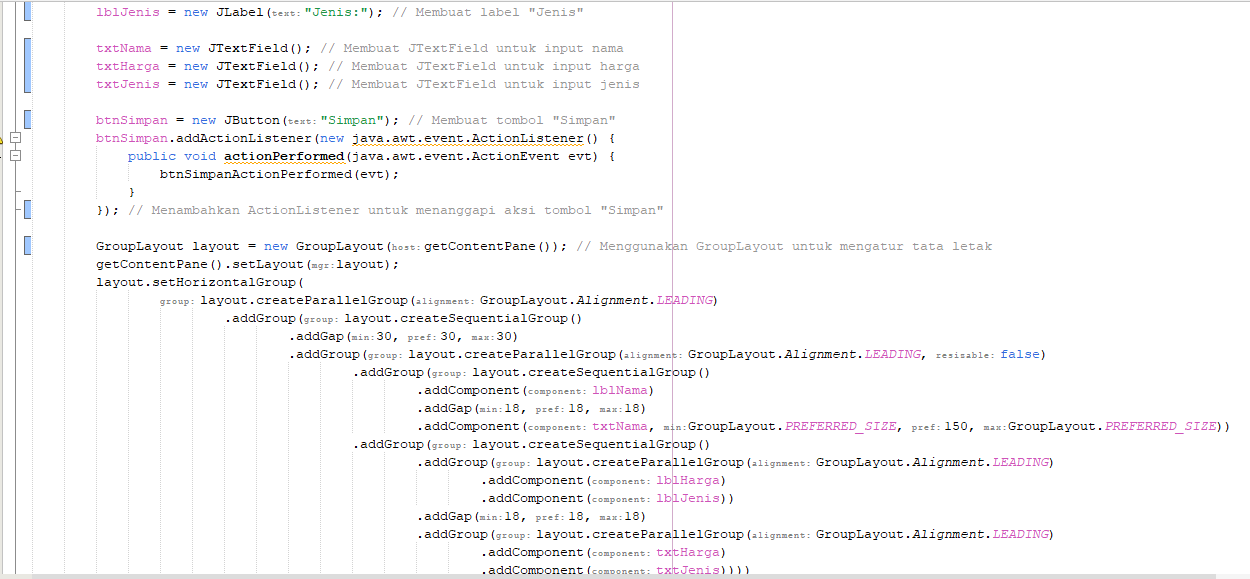
1. private JButton btnTambahElektronik;: Deklarasi tombol "Tambah Elektronik".
2. private ArrayList<Produk> daftarProduk;: Deklarasi ArrayList untuk menyimpan daftar produk elektronik.
3. public MainForm() { ... }: Konstruktor untuk kelas MainForm, di mana ArrayList daftarProduk diinisialisasi.
4. private void initComponents() { ... }: Metode untuk menginisialisasi komponen GUI.
5. setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);: Menetapkan operasi default ketika jendela ditutup.
6. setTitle("Sistem Manajemen Toko Elektronik");: Menetapkan judul jendela.
7. btnTambahElektronik = new JButton("Tambah Elektronik");: Membuat tombol "Tambah Elektronik".
8. btnTambahElektronik.addActionListener(new ActionListener() { ... });: Menambahkan ActionListener untuk menanggapi aksi tombol "Tambah Elektronik".
9. GroupLayout layout = new GroupLayout(getContentPane());: Menggunakan GroupLayout untuk mengatur tata letak komponen.
10. pack();: Mengatur ukuran jendela berdasarkan komponen yang ada.
11. private void btnTambahElektronikActionPerformed(ActionEvent evt) { ... }: Metode yang akan dijalankan saat tombol "Tambah Elektronik" ditekan.
12. ElektronikForm elektronikForm = new ElektronikForm(this, true, daftarProduk);: Membuat objek ElektronikForm dengan meneruskan referensi ke MainForm dan daftarProduk.
13. elektronikForm.setVisible(true);: Menampilkan antarmuka form tambah produk elektronik.
14. public void tambahProduk(Produk produk) { daftarProduk.add(produk); }: Metode untuk menambahkan produk ke dalam ArrayList daftarProduk.

## **Antarmuka tambah produk ( ElektronikForm )**

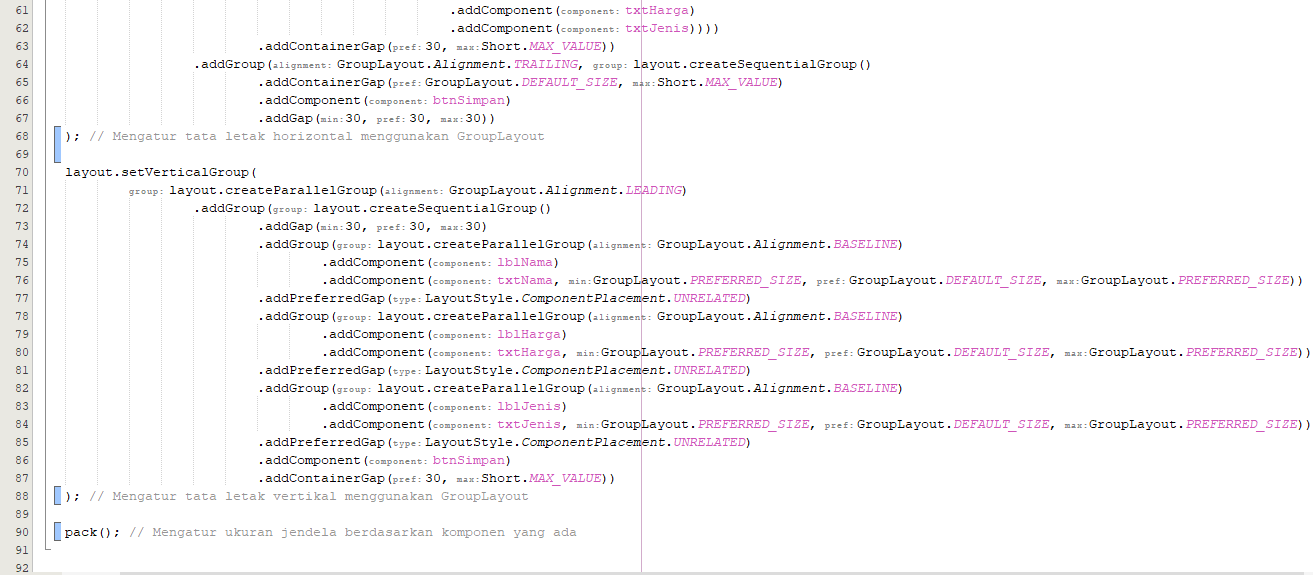
Antarmuka untuk menambah produk elektronik dengan input nama, harga, dan jenis produk.



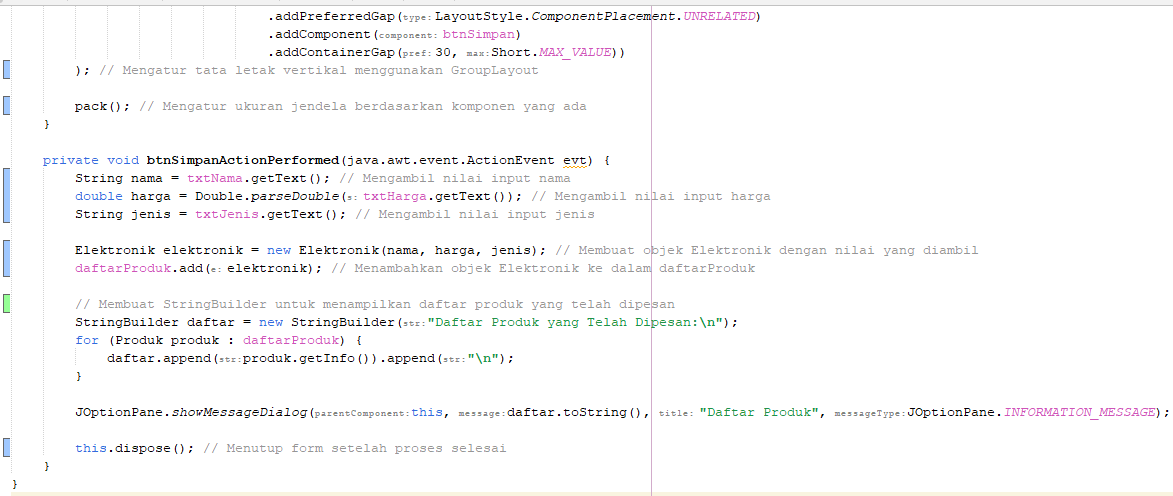
Gambar 1. 8 ElektronikForm 1



Gambar 1. 9 ElektronikForm 2



Gambar 1. 10 ElektronikForm 3



Gambar 1. 11 ElektronikForm 4

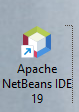
Penjelasan:

1. import javax.swing.\*;: Mengimpor pustaka Swing untuk komponen GUI.
2. import java.util.ArrayList;: Mengimpor pustaka ArrayList untuk menyimpan daftar produk.
3. public class ElektronikForm extends JDialog {: Mendeklarasikan kelas ElektronikForm yang meng-extend JDialog.
4. private JLabel lblNama, lblHarga, lblJenis;: Deklarasi label (JLabel) untuk nama, harga, dan jenis produk.
5. private JTextField txtNama, txtHarga, txtJenis;: Deklarasi JTextField untuk input nama, harga, dan jenis produk.
6. private JButton btnSimpan;: Deklarasi tombol (JButton) "Simpan".
7. private ArrayList<Produk> daftarProduk;: Deklarasi ArrayList untuk menyimpan daftar produk.
8. public ElektronikForm(java.awt.Frame parent, boolean modal, ArrayList<Produk> daftarProduk) {: Konstruktor ElektronikForm dengan menerima referensi ke MainForm dan daftarProduk sebagai parameter.
9. super(parent, modal);: Memanggil konstruktor superclass (JDialog) dengan merujuk ke parent (MainForm) dan menyatakan apakah bersifat modal atau tidak.
10. this.daftarProduk = daftarProduk;: Menginisialisasi daftarProduk dengan ArrayList yang diterima sebagai parameter.
11. initComponents();: Memanggil metode untuk menginisialisasi komponen GUI.
12. private void initComponents() {: Metode untuk menginisialisasi komponen GUI.
13. setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE\_ON\_CLOSE);: Menetapkan operasi default ketika jendela ditutup.
14. setTitle("Tambah Produk Elektronik");: Menetapkan judul jendela.
15. Inisialisasi komponen GUI:
16. lblNama = new JLabel("Nama:");
17. lblHarga = new JLabel("Harga:");
18. lblJenis = new JLabel("Jenis:");
19. txtNama = new JTextField();
20. txtHarga = new JTextField();
21. txtJenis = new JTextField();
22. btnSimpan = new JButton("Simpan")

# BAB IV IMPLEMENTASI PROGRAM

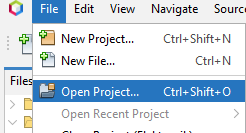
## **Menjalankan Program**

1. Untuk menjalankan program klick neatbeans

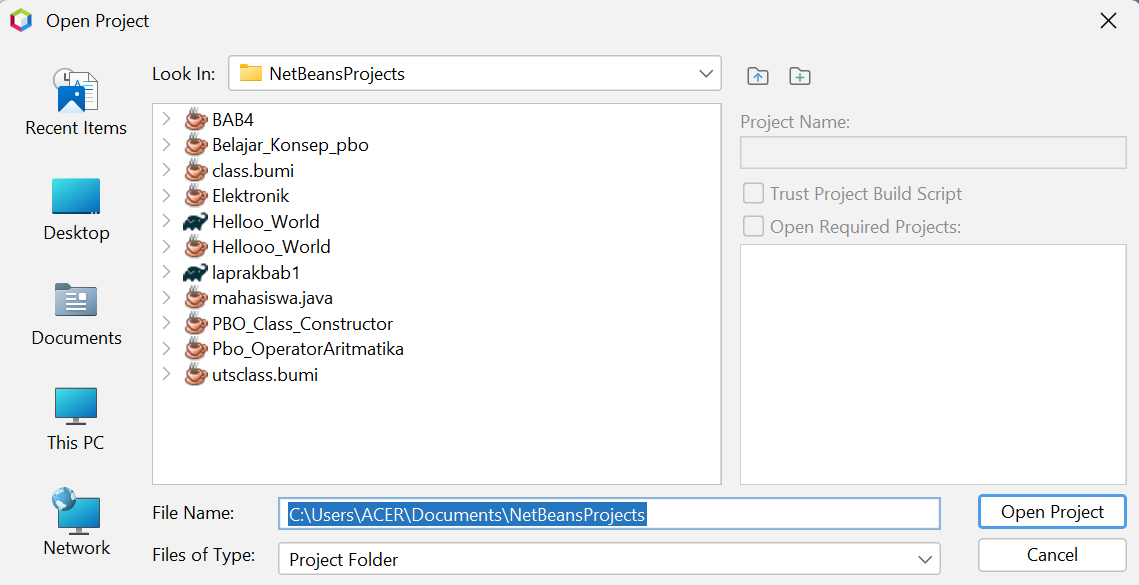


Gambar 2. 1 Open Neatbeans

1. Open new project, arahkan ke elektronik

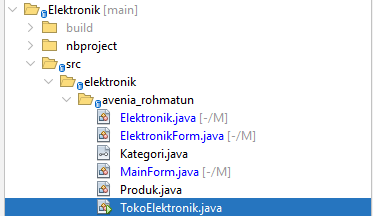


Gambar 2. 2 Open Projects



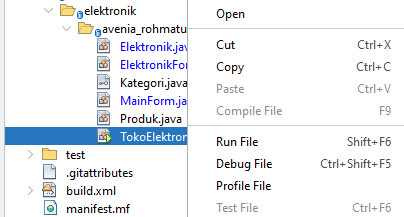
Gambar 2. 3 Arahkan ke project

1. Buka src/elektronik/avenia\_rohmatun



Gambar 2. 4 Open struktur project

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



Gambar 2. 5 Run program

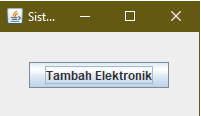
1. Tunggu program muncul



Gambar 2. 6 Tunggu program muncul

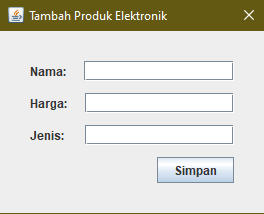
## **Menambahkan Produk**

1. Klick Tombol Tambah Elektronik



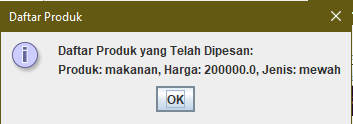
Gambar 2. 7 Tambah Elektronik

1. Isi Data Nama, Harga, Jenis. Lalu klick Simpan



Gambar 2. 8 Isi Data, Klick Simpan

1. Maka akan muncul sebuah dialog konfirmasi daftar produk yang telah dipesan adalah (apa), harga(berapa), jenis(apa).



Gambar 2. 9 Keluar Dialog Pesanan

# BAB V Kesimpulan

Program Java GUI NeatBeans Sistem Manajemen Toko Elektronik ini merupakan implementasi dari beberapa konsep dasar pemrograman berorientasi objek, seperti:

1. Inheritance: Terlihat pada kelas Elektronik yang merupakan turunan dari kelas Produk. Dengan menggunakan inheritance, kelas Elektronik dapat mewarisi properti dan metode dari kelas Produk.
2. Polymorphism: Polymorphism terjadi pada metode getInfo yang di-override di kelas Elektronik. Meskipun metode ini dideklarasikan di kelas induk Produk, namun perilaku (implementasinya) dapat berbeda di kelas anak Elektronik.
3. Encapsulation: Konsep encapsulation diterapkan pada penggunaan akses modifier private pada variabel-variabel di kelas Produk dan Elektronik. Akses ke variabel-variabel ini hanya dapat dilakukan melalui metode getter dan setter.
4. Getter dan Setter: Terdapat metode getter dan setter pada kelas Produk untuk memanipulasi nilai variabel nama dan harga. Penggunaan getter dan setter membantu dalam menjaga kontrol terhadap akses dan manipulasi data.
5. Interface: Konsep interface diimplementasikan pada Kategori yang memiliki satu metode getKategori(). Kelas Elektronik mengimplementasikan interface Kategori, sehingga wajib menyediakan implementasi untuk metode getKategori().

Dalam struktur program ini, terdapat dua frame utama, yaitu MainForm dan ElektronikForm. MainForm berfungsi sebagai tampilan utama yang memungkinkan pengguna untuk menambahkan produk elektronik. Produk yang ditambahkan akan disimpan dalam daftarProduk. ElektronikForm adalah formulir tambah produk elektronik yang memanfaatkan daftarProduk yang diteruskan dari MainForm. Produk yang ditambahkan melalui formulir ini akan langsung ditampilkan dalam pesan dialog.

Keseluruhan program memberikan gambaran bagaimana konsep-konsep dasar OOP diterapkan dalam pembuatan program dengan antarmuka grafis untuk manajemen toko elektronik.

# DAFTAR PUSTAKA

(Rosenberg and Stephens, 2007; Nasseri, Counsell and Shepperd, 2010; Perry, 2010)Nasseri, E., Counsell, S. and Shepperd, M. (2010) ‘Class movement and re-location: An empirical study of Java inheritance evolution’, *Journal of Systems and Software*, 83(2), pp. 303–315. Available at: https://doi.org/10.1016/j.jss.2009.08.011.

Perry, J.S. (2010) *Table of content*, *IEEE Aerospace and Electronic Systems Magazine*. Available at: https://doi.org/10.1109/maes.2010.5638796.

Rosenberg, D. and Stephens, M. (2007) *Use case driven object modeling with UML: Theory and Practice*, *Use Case Driven Object Modeling with UML: Theory and Practice*. Available at: https://doi.org/10.1007/978-1-4302-0369-8\_1.