PEMROGRAMAN BERBASIS OBJEK
PROGRAM KASIR WARUNG BAKSO BERBASIS JAVA

Laporan ini diajukan untuk memenuhi tugas besar mata kuliah Pemrograman Berbasis Objek

Disusun oleh :

Muhammad Naufal Nur Irawan (1101204104)
Rabby Fitriana Adawiyah (1101202505)
Rafi Hadi Muthi Wicaksono (1101202169)

Kelas : TT-44-04

PROGRAM STUDI S1 TEKNIK TELEKOMUNIKASI
FAKULTAS TEKNIK ELEKTRO
UNIVERSITAS TELKOM
2022/2023

KATA PENGANTAR

Puji dan syukur kami panjatkan kepada Allah Ta'ala atas karunianya kami dapat menyelesaikan Tugas Besar Pemrograman Berbasis Objek sebagai pertanggungjawaban untuk nilai mata kuliah Pemrograman Berbasis Objek. Laporan yang berjudul "Program Kasir Warung Bakso" ini dapat diselesaikan tepat waktu dengan maksimal dan tanpa kendala.

Untuk itu, kami sangat berterimakasih kepada pihak-pihak yang telah membantu khususnya kepada Bapak Fardan selaku Dosen mata kuliah Pemrograman Berbasis Objek. Kami menyadari masih banyak kekurangan dalam penyelesaian Laporan Tugas Besar kami. Oleh karena itu, kami meminta maaf dan menerima kritik serta saran yang membangun dari pembaca.

Demikian tugas besar ini kami susun, semoga dapat bermanfaat bagi semua pihak serta penulis sendiri. Akhir kata kami ucapkan terimakasih.

Bandung, 3 Januari 2023

Tim Penyusun

DAFTAR ISI

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Seiring kemajuan teknologi yang berkembang saat ini serta kebutuhan bisnis, "Warung Bakso" belum memanfaatkan teknologi sistem informasi untuk mendukung pengelolaan sistem kasir di sana. "Warung Bakso" masih memanfaatkan teknologi kalkulator manual untuk menghitung pesanan pelanggan. Dengan pengelolaan seperti itu sering terjadi kesalahan dalam menghitung total harga yang dipesan pelanggan yang dapat merugikan penjual ataupun pelanggan.

Dari kendala pengelolaan kasir di "Warung Bakso" yaitu kesalahan perhitungan, lamanya dalam menghitung total harga, maka perancangan program ini dibuat untuk mengurangi kendala-kendala tersebut. Perancangan program ini akan dibuat menggunakan bantuan aplikasi Visual Studio Code berbasis Java. Dari latar belakang di atas, akan disusun laporan Tugas Besar dengan judul "PROGRAM KASIR WARUNG BAKSO".

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah di atas maka rumusan masalah pada laporam ini adalah bagaimana merancang Program Kasir Warung Bakso berbasis Java, sehingga bisa memberikan layanan yang baik kepada pelanggan maupun pihak penjual meliputi kemudahan dan keakuratan dalam perhitungan harga, dan transaksi sehinggan dapat meningkatkan efisiensi dan pelayanan pada "Warung Bakso".

1.3 Tujuan dan Manfaat

Perancangan program "Kasir Warung Bakso" memiliki tujuan, antara lain :

1. Untuk menghasilkan sebuah program kasir yang nantinya akan digunakan oleh pihak "Warung Bakso"
2. Merancang program sebuah program kasir yang dapat digunakan untuk menghitung biaya harga sehingga dapat meningkatkan efisiensi dan pelayanan pada "Warung Bakso"
3. Membuat program kasir menggunakan Visual Studio Code berbasis Java.

Manfaat perancangan program "Kasir Warung Bakso", antara lain :

1. Membantu berkembangnya bisnis "Warung Bakso" dalam meningkatkan kualitas pelayanan terhadap pelanggan
2. Memudahkan pegawai "Warung Bakso" dalam melakukan tugasnya sebagai kasir.
3. Menambah wawasan serta sebagai syarat untuk memenuhi nilai mata kuliah Pemrograman Berbasis Objek.

BAB II
LANDASAN TEORI

2.1 Java dan JDK

Java adalah sebuah teknologi yang diperkenalkan oleh Sun Microsystem pada pertengahan tahun 1990. Menurut definisi Sun, Java adalah nama untuk sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada computer ataupun pada lingkungan jaringan. Teknologi Java merupakan sebuah bahasa pemrograman yang lebih lengkap dibanding sebuah bahasa pemrograman konvensional. Teknologi Java memiliki komponen penting, yaitu :

1. Java Development Kit

   Java Development Kit merupakan perangkat lunak yang digunakan untuk melakukan proses kompilasi dari kode Java menjadi bytecode yang dapat dimengerti dan dapat dijalankan oleh Java Runtime Environtment.

2. Java Runtime Environtment

   Java Runtime Environtment merupakan perangkat lunak yang digunakan untuk menjalankan aplikasi yang dibangun menggunakan java. Versi JRE harus sama atau lebih tinggi dari JDK yang digunakan untuk membangun aplikasi agar aplikasi dapat berjalan sesuai dengan yang diharapkan.

3. IDE (*Integrated Development Environment)*

   IDE (Integrated Development Environment) merupakan sofware yang berfungsi untuk membuat suatu program atau aplikasi. Pada bagian ini, kita membahas Bahasa Java yang dipakai untuk membuat suatu aplikasi. Dalam membuat suatu aplikasi yang berbasis Java, beberapa IDE yang bisa digunakan antara lain Netbeans dan Eclipse. Netbeans dan Eclipse mempunyai kelebihan dan kelemahan masing - masing. Tetapi kedua IDE tersebut merupakan IDE handal untuk merancang suatu aplikasi Java.

2.2 GUI *(Graphical User Interface)*

GUI (Graphical User Interface) adalah bentuk antarmuka pengguna untuk memungkinkan user dapat berinteraksi dengan perangkat elektronik. GUI memiliki beberapa elemen, mulai dari elemen windows, menu, icon, widget dan juga tab. Untuk menggunakan elemen ini biasanya GUI akan mendapatkan inputan dari perangkat masukan.

Penggunaan dari GUI ini diimplementasikan pada perangkat elektronik seperti komputer, laptop dan juga smartphone. GUI biasanya digunakan untuk representasi visual untuk melakukan perintah dan fungsi pada operating system Anda juga pada software. Dengan adanya GUI ini maka user akan lebih mudah dalam hal menggunakan fitur-fitur yang penting dalam sebuah perangkat elektronik. Untuk menggunakan fungsi dengan mudah ini, beberapa hal yang biasanya digunakan adalah alat input seperti keyboard, mouse dan masih banyak lagi lainnya.

2.3 UML *(Unified Modelling Language)*

UML (Unified Modeling Language) adalah sekumpulan diagram yang digunakan untuk melakukan abstraksi terhadap sebuah sistem atau perangkat lunak berbasis objek. UML dapat digunakan untuk mempermudah pengembangan aplikasi yang berkelanjutan. UML dapat dikatakan juga sebagai perkembangan, bahasa pemodelan di bidang rekayasa perangkat lunak yang dimaksudkan untuk menyediakan cara standar untuk memvisualisasikan desain sebuah sistem. UML terdiri dari banyak elemen-elemen grafis yang digabungkan dalam bentuk diagram. Tujuan representasi elemen-elemen grafis ke dalam diagram adalah untuk menyajikan beragam sudut pandang dari sebuah sistem berdasarkan fungsi masing-masing diagram tersebut. Kumpulan dari beragam sudut pandang inilah yang disebut sebuah model.

Berikut diagram-diagram dalam UML:

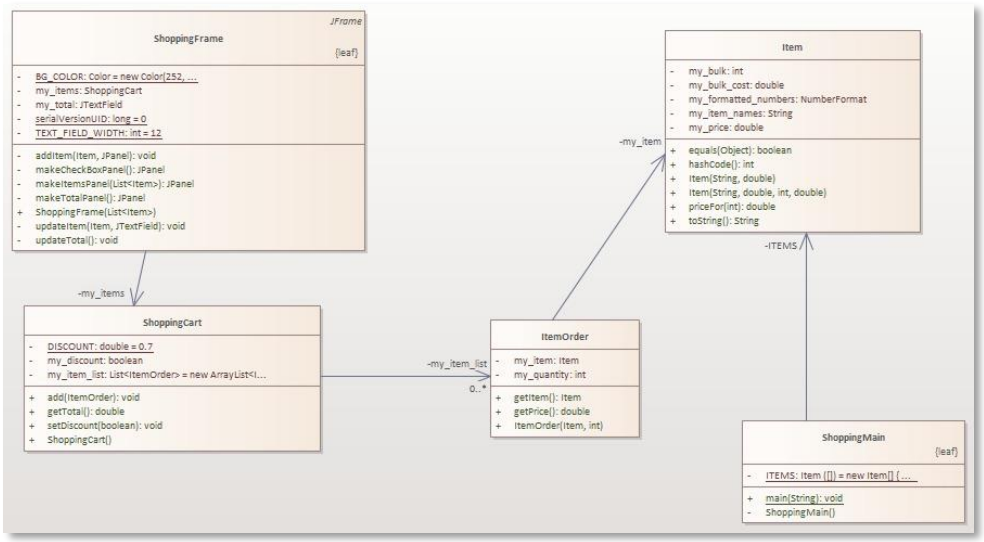
Structural Diagram

- Class Diagram: sebuah diagram yang menjelaskan hubungan antar class dalam sebuah sistem yang sedang dibuat dan menjelaskan bagaimana caranya agar mereka saling berkolaborasi. Class Diagram UML inilah yang digunakan pada laporan kami.
- Object Diagram: sebuah gambaran tentang objek-objek dalam sebuah sistem pada satu titik waktu. Karena lebih menonjolkan perintah-perintah daripada *class*, *object* diagram lebih sering disebut sebagai sebuah diagram perintah.
- Component Diagram: diagram yang menampilkan komponen dalam sistem dan hubungan antara mereka.
- Deployment Diagram: Diagram yang menunjukkan tata letak sebuah sistem secara fisik, menampakkan bagian-bagian *software* yang berjalan pada bagian-bagian *hardware* yang digunakan untuk mengimplementasikan sebuah sistem dan keterhubungan antara komponen-komponen *hardware*.

Behavioral Diagram

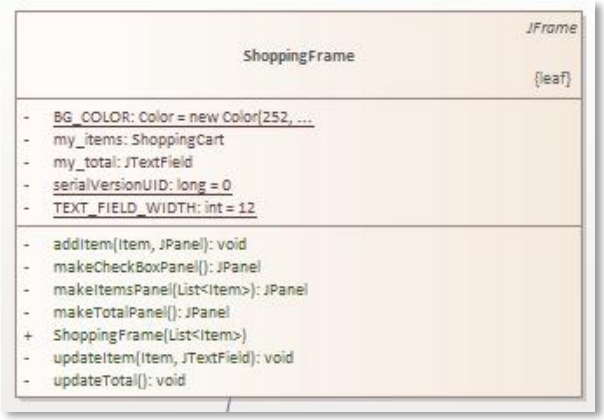
- Activity Diagram: diagram yang menggambarkan alur kerja dari berbagai aktivitas *user* atau sistem, orang yang melakukan aktivitas, dan aliran berurutan dari aktivitas ini.
- Use case Diagram: diagram yang menunjukkan peran *user* dan bagaimana peran tersebut ketika menggunakan sistem. Use case diagram juga dapat digunakan untuk memprepresentasikan interaksi *user* dengan sistem dan menggambarkan spesifikasi kasus penggunaan.
- System Sequence Diagram: diagram yang menggambarkan interaksi *user* dengan sistem secara sekuensial (berurutan).
- Collaboration Diagram: diagram yang merupakan bentuk lain dari *sequence diagram*. Diagram ini menggambarkan struktur organisasi dari sistem dengan pesan yang diterima dan dikirim.
- Statechart Diagram: diagram yang menggambarkan bagaimana sistem dapat bereaksi terhadap suatu kejadian dari dalam atau luar. Kejadian (*event*) ini bertanggung jawab terhadap perubahan keadaan sistem.

BAB III
UML DIAGRAM



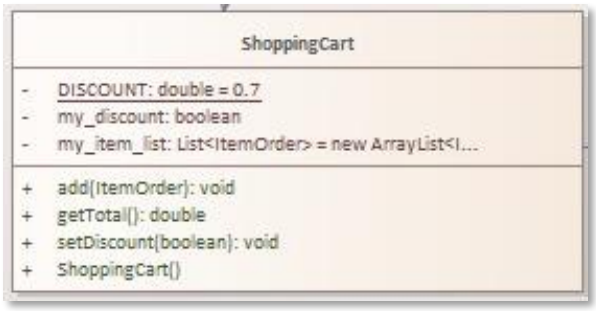
Gambar 1 UML Diagram

3.1 Shopping Frame



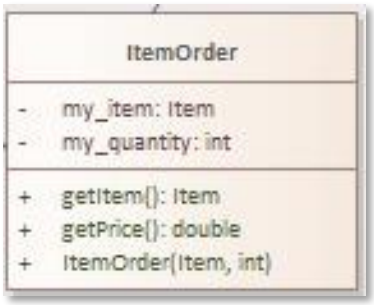
Gambar 2 UML : Shopping Frame

3.2 Shopping Cart



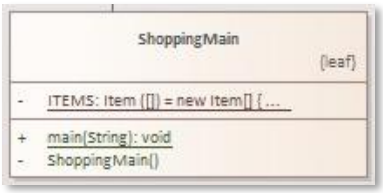
Gambar 3 UML : Shopping Cart

3.3 Item Order



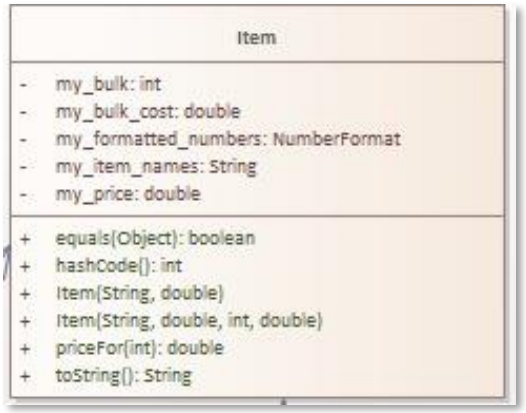
Gambar 4 UML : Item Order

3.4 Shopping Main



Gambar 5 UML : Shopping Main

3.5 Item



Gambar 6 UML : Item

BAB IV
DOKUMENTASI PROGRAM

4.1 *Source Code*

a. Item.java

```java
import java.text.NumberFormat;

//Program memproses barang-barang (items) yang akan ditampilkan

public class Item {

  /**
   * membuat tipe data untuk nama barang (string)
   */
  private final String my_item_names;

  /**
   * membuat tipe data double untuk harga barang
   */
  private final double my_price;

  /**
   * input tipe data untuk harga barang banyak.
   */
  private final int my_bulk;

  /**
   * double untuk tipe data harga barang banyak.
   */
  private final double my_bulk_cost;

  /**
   * NumberFormat my_formatted_numbers untuk menampilkan mata uang yang akan
   * ditampilkan.
   */
  private final NumberFormat my_formatted_numbers;
```

Gambar 7 Source Code : Item.java (1)

```java
  /**
   * @param the_item_name  variable nama barang
   * @param the_item_price variable untuk harga barang
   */
  public Item(final String the_item_name, final double the_item_price) {

    this(the_item_name, the_item_price, the_quantity_in_bulk: 0, the_price_in_bulk: 0.0);

  }

  /**
   * @param the_quantity_in_bulk variable untuk jumlah barang banyak
   * @param the_price_in_bulk    variable untuk harga banyak barang
   */
  public Item(final String the_item_name, final double the_item_price,
      final int the_quantity_in_bulk, final double the_price_in_bulk) {
    my_item_names = the_item_name;
    my_price = the_item_price;
    my_bulk = the_quantity_in_bulk;
    my_bulk_cost = the_price_in_bulk;
    my_formatted_numbers = NumberFormat.getCurrencyInstance();
  }

```

Gambar 8 Source Code : Item.java (2)

```java
/**
 * @param a_quantity varibale jumlah barang yang ditentukan.
 * @return price
 */
public double priceFor(final int a_quantity) {

  final int quantity = a_quantity;
  double bulk_num;

  if (quantity >= my_bulk && my_bulk > 0) {
    final int bulk_quantity = quantity / my_bulk;
    final int reg_quantity = quantity % my_bulk;
    bulk_num = bulk_quantity * my_bulk_cost + reg_quantity * my_price;
  } else {
    bulk_num = quantity * my_price;
  }
  return bulk_num;
}

```

Gambar 9 Source Code : Item.java (3)

```java
  // proses method tipe data string

  public String toString() {
    StringBuilder stringbuilder;
    stringbuilder = new StringBuilder().append(my_item_names);
    stringbuilder.append(str: ", ");
    stringbuilder.append(my_formatted_numbers.format(my_price));
    if (my_bulk > 0) {
      stringbuilder.append(str: " (");
      stringbuilder.append(my_bulk);
      stringbuilder.append(str: " for ");
      stringbuilder.append(my_formatted_numbers.format(my_bulk_cost));
      stringbuilder.append(str: ")");
    }
    return stringbuilder.toString();
  }

```

Gambar 10 Source Code : Item.java (4)

```java
  public boolean equals(final Object the_other) {
    final Item check = (Item) the_other;
    boolean temporary_checker;

    if (check.my_item_names != null &&
        check.my_item_names.equals(my_item_names) &&
        check.my_price == my_price &&
        check.my_bulk == my_bulk &&
        check.my_bulk_cost == my_bulk_cost) {
      temporary_checker = true;
    } else {
      temporary_checker = false;
    }

    return temporary_checker;

  }

  public int hashCode() {

    return my_item_names.hashCode();
  }
}

```

Gambar 11 Source Code : Item.java (5)

b. ItemOrder.java

```java
// Program Pemrosesan barang-barang yang di order
public class ItemOrder {

  private final Item my_item;
  private final int my_quantity;

  /**
   * @param the_item_name         variable untuk nama-nama barang
   * @param the_quantity_of_items variable untuk jumlah(harga) barang
   */
  public ItemOrder(final Item the_item_name, final int the_quantity_of_items) {
    my_item = the_item_name;
    my_quantity = the_quantity_of_items;
  }

  /**
   * Mengembalikan item(barang) untuk mendapatkan harga brang
   *
   * @return my_item.(priceFor(my_quantity)
   */
  public double getPrice() {

    return my_item.priceFor(my_quantity);
  }

  /**
   * dikembalikan ke barang
   *
   * @return my_item
   */
  public Item getItem() {

    return my_item;
  }
}
```

Gambar 12 Source Code : ItemOrder.java

c. ShoppingCart.java

```java
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

/**
 * ini adalah program untuk pembuatan(proses) shooping cart
 */

public class ShoppingCart {

  /**
   * Menentukan program untuk diskon yang diberikan.
   */
  private static final double DISCOUNT = 0.7;

  /**
   * fungsi boolean untuk menentukan apakah program diskon dijalankan/tidak.
   */
  private boolean my_discount;
  private final List<ItemOrder> my_item_list = new ArrayList<ItemOrder>();

  public ShoppingCart() {
    my_discount = false;
  }
```

Gambar 13 Source Code : ShoppingCart.java

d. ShoppingFrame.java

```java
//import libary untuk ploting frame GUI

import java.awt.Color;
import java.awt.FlowLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.FocusAdapter;
import java.awt.event.FocusEvent;
import java.text.NumberFormat;
import java.util.List;

import javax.swing.JCheckBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextField;
import javax.swing.SwingConstants;
```

Gambar 14 Source Code : ShoppingFrame.java (1)

```java

/**
 * ShoppingFrame, Program untuk tampilan GUI aplikasi Shopping Cart.
 */
public final class ShoppingFrame extends JFrame {
  /**
   * Serialization ID, di perlukan untuk merepresentasikan bagian - bagian dari
   * program agar termasuk kedalam objek data.
   */
  private static final long serialVersionUID = 0;

  /**
   * Membuat Bidang untuk text agar bisa ditampilkan di GUI.
   */
  private static final int TEXT_FIELD_WIDTH = 12;

  /**
   * Membuat warna background untuk interface barang-barang di GUI.
   */
  private static final Color BG_COLOR = new Color(r: 252, g: 140, b: 3);

  /**
   * memprogram GUI Shopping Cart.
   */
  private final ShoppingCart my_items;

  /**
   * Membuat bidang kosong untuk total harga yang customer beli.
   */
  private final JTextField my_total;

```

Gambar 15 Source Code : ShoppingFrame.java (2)

```java
/**
 * Membuat GUI shooping cart untuk item yang dijual ke custumer.
 *
 * @param the_items adalah barang - barang yang diinput oleh user.
 */
public ShoppingFrame(final List<Item> the_items) {
  // membuat bingkai (nama TOKO) dan daftar orderan
  super(title: "WARUNG BAKSO");
  setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
  my_items = new ShoppingCart();

  // inputan untuk text di total harga
  my_total = new JTextField(text: "Rp.0.00", TEXT_FIELD_WIDTH);
  add(makeTotalPanel(), constraints: "North");
  add(makeItemsPanel(the_items), constraints: "Center");
  add(makeCheckBoxPanel(), constraints: "South");

  // fit the size untuk tampilan pada GUI
  pack();
  setVisible(b: true);
}
```

Gambar 16 Source Code : ShoppingFrame.java (3)

```java
/**
 * Membuat panel untuk total harga
 */
private JPanel makeTotalPanel() {
  // mengatur bidang agar tidak bisa di edit oleh user
  // dan menentukan warna untuk tulisan

  my_total.setEditable(b: false);
  my_total.setEnabled(enabled: false);
  my_total.setDisabledTextColor(Color.black);

  // Membuat panel untuk tampilan total harga order di GUI

  final JPanel p = new JPanel();
  p.setBackground(Color.red);
  final JLabel l = new JLabel(text: "Total Harga Order");
  l.setForeground(Color.green);
  p.add(l);
  p.add(my_total);
  return p;
}

/**
 * Membuat panel untuk menampung daftar barang yang dibeli.
 */
private JPanel makeItemsPanel(final List<Item> the_items) {
  final JPanel p = new JPanel(new GridLayout(the_items.size(), cols: 1));

  for (Item item : the_items) {
    addItem(item, p);
  }

  return p;
}
```

Gambar 17 Source Code : ShoppingFrame.java (4)

```java
/**
 * Membuat checkbox untuk opsi kalau memasukan harga diskon
 *
 * @param the_event adalah panggilan untuk adanya diskon
 */
private JPanel makeCheckBoxPanel() {
  final JPanel p = new JPanel();
  p.setBackground(Color.yellow);
  final JCheckBox cb = new JCheckBox(text: "Ada Diskon 30% bayar pake PayPay");
  p.add(cb);
  cb.addActionListener(new ActionListener() {
    public void actionPerformed(final ActionEvent the_event) {
      my_items.setDiscount(cb.isSelected());
      updateTotal();
    }
  });
  return p;
}
```

Gambar 18 Source Code : ShoppingFrame.java (5)

```java

/**
 * program untuk menambahkan barang yang dibeli ke panel yang telah ditentukan
 *
 * @param the_item  barang yang dipilih
 * @param the_panel Panel yang dipakai (telah ditentukan).
 */
private void addItem(final Item the_item, final JPanel the_panel) {
  final JPanel sub = new JPanel(new FlowLayout(FlowLayout.LEFT));
  sub.setBackground(BG_COLOR);
  final JTextField quantity = new JTextField(columns: 3);
  quantity.setHorizontalAlignment(SwingConstants.CENTER);
  quantity.addActionListener(new ActionListener() {
    public void actionPerformed(final ActionEvent the_event) {
      updateItem(the_item, quantity);
      quantity.transferFocus();
    }
  });
  quantity.addFocusListener(new FocusAdapter() {
    public void focusLost(final FocusEvent the_event) {
      updateItem(the_item, quantity);
    }
  });
  sub.add(quantity);
  final JLabel l = new JLabel(the_item.toString());
  l.setForeground(Color.BLACK);
  sub.add(l);
  the_panel.add(sub);
}

```

Gambar 19 Source Code : ShoppingFrame.java (6)

```
155   /**
156    * program untuk menjumlahkan (mengupdate) barang barang yang telah ditentukan.
157    *
158    * @param the_item     barang yang diupdate.
159    * @param the_quantity Jumlah barang yang telah ditentukan.
160    */
161   private void updateItem(final Item the_item, final JTextField the_quantity) {
162     final String text = the_quantity.getText().trim();
163     int number;
164     try {
165       number = Integer.parseInt(text);
166       if (number < 0) {
167         // mencegah ada bilangan negatif (barang =! -1)
168         throw new NumberFormatException();
169       }
170     } catch (final NumberFormatException e) {
171       number = 0;
172       the_quantity.setText(t: "");
173     }
174     my_items.add(new ItemOrder(the_item, number));
175     updateTotal();
176   }
```

Gambar 20 Source Code : ShoppingFrame.java (7)

```
177
178   /**
179    * Menampilkan total barang yang dpilih
180    */
181   private void updateTotal() {
182     final double total = my_items.getTotal();
183     my_total.setText(NumberFormat.getCurrencyInstance().format(total));
184   }
185 }
186
187 // END of ShoppingCart GUI
188
```

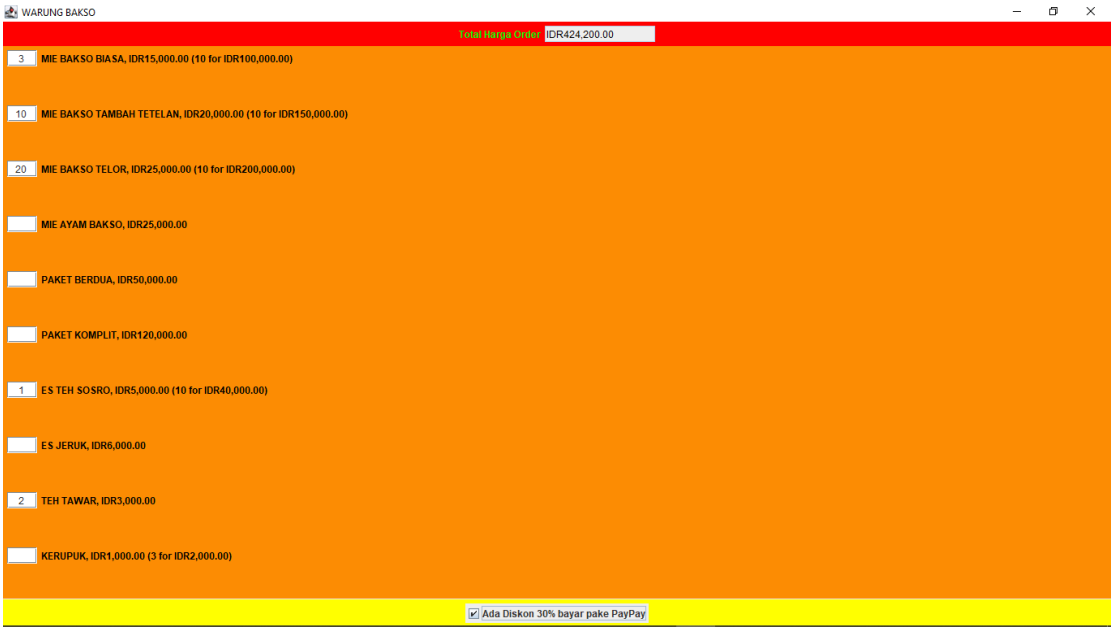
Gambar 21 Source Code : ShoppingFrame.java (8)

e. ShoppingMain.java

```java
import java.util.Arrays;

/**
 * ShoppingMain Program utama dari Shopping Cart dengan GUI
 */

public final class ShoppingMain {
  /**
   * List Item yang ada di toko
   */
  private static final Item[] ITEMS = new Item[] {
      new Item(the_item_name: "MIE BAKSO BIASA", the_item_price: 15000, the_quantity_in_bulk: 10, the_price_in_bulk: 100000),
      new Item(the_item_name: "MIE BAKSO TAMBAH TETELAN", the_item_price: 20000, the_quantity_in_bulk: 10, the_price_in_bulk
      new Item(the_item_name: "MIE BAKSO TELOR", the_item_price: 25000, the_quantity_in_bulk: 10, the_price_in_bulk: 200000),
      new Item(the_item_name: "MIE AYAM BAKSO", the_item_price: 25000),
      new Item(the_item_name: "PAKET BERDUA", the_item_price: 50000),
      new Item(the_item_name: "PAKET KOMPLIT", the_item_price: 120000),
      new Item(the_item_name: "ES TEH SOSRO", the_item_price: 5000, the_quantity_in_bulk: 10, the_price_in_bulk: 40000),
      new Item(the_item_name: "ES JERUK", the_item_price: 6000),
      new Item(the_item_name: "TEH TAWAR", the_item_price: 3000),
      new Item(the_item_name: "KERUPUK", the_item_price: 1000, the_quantity_in_bulk: 3, the_price_in_bulk: 2000) };

  /**
   * Pakai constructor Private, supaya tidak ada instance
   */
  private ShoppingMain() {
  }

  // @param the_args untuk mendefinisikan jika main method merupakan string
  Run | Debug
  public static void main(final String... the_args) {
    new ShoppingFrame(Arrays.asList(ITEMS));
  }
}
```

Gambar 22 Source Code : ShoppingMain.java

4.2 *Running Program*



Gambar 23 Program dengan GUI

BAB V
PENUTUP

5.1 Kesimpulan

Dalam laporan ini dapat disimpulkan mengenai program kasir “Warung Bakso”, diantaranya adalah :

1. Perancangan Program kasir “Warung Bakso” berhasil dibuat dengan menggunakan java dan dapat digunakan.
2. Program yang dibuat dapat mempermudah pegawai kasir dalam melakukan perhitungan pembayaran oleh pelanggan, sehingga dapat meningkatkan efisiensi dan pelayanan pada “Warung Bakso”

5.2 Saran

Dalam bagian ini akan diberikan saran mengenai program ini, antara lain :

1. Tampilan dari program ini masih terbatas, sehingga untuk pengembangan selanjutnya tampilan dari aplikasi dapat dibuat lebih menarik lagi dengan mementingkan keramahan bagi pengguna.
2. Aplikasi ini masih harus dikembangkan ke arah yang lebih sempurna dengan melakukan serangkaian uji coba dalam masa waktu tertentu, sehingga dapat berguna untuk user dan khususnya pegawai kasir “Warung Bakso”

DAFTAR PUSTAKA

A., F. (2022, Desember 14). *Integrated Development Environment (IDE)*. Diambil kembali dari Hostinger Tutorial: https://www.hostinger.co.id/tutorial/integrated-development-environment-adalah

Alexandra, J. (2019, Mei 15). *Model-model Diagram UML*. Diambil kembali dari Binus University School of Information System: https://sis.binus.ac.id/2019/05/15/model-model-diagram-uml/

Arni, U. D. (2021, Januari 18). *Perbedaan JDK dan IDE*. Diambil kembali dari Garuda Cyber Indonesia: https://garudacyber.co.id/artikel/2108-perbedaan-jdk-dan-ide

Gumanti, M. (2012). *Diagram UML*. Diambil kembali dari lengkapku: http://mukti362.blogspot.com/2012/09/diagram-uml.html

Haqi, B. (2019). *Aplikasi SPK Pemilihan Dosen Terbaik Metode SAW dengan Java.* Sleman: DEEPUBLISH.

Kadir, A. (2012). *Algoritma dan Pemrograman menggunakan Java.* Yogyakarta: Andi.

Maulana, I. F., Khotijah, S., & Hapsari, A. T. (2020). PERANCANGAN SISTEM INFORMASI KASIR DI I-WASH CUCI KENDARAAN . *Journal of Informatioan System, Informatics and Computing*, 111-112.

Ritonga, P. (2015). *Pengertian Unified Modeling Language (UML)*. Diambil kembali dari bangpahmi: http://www.bangpahmi.com/2015/04/pengertian-unified-modelling-language-uml-dan-modelnya-menurut-pakar.html

Satzinger, J. W. (2012). *Introduction to Systems Analysis and Design : an agile, interactive approach.* Canada: Course Technology.