LAPORAN UJIAN TENGAH SEMESTER
PRAKTIKUM PEMOGRAMAN BERORIENTASI OBJEK

<u>BOOK SHOP MANAGEMENT SYSTEM</u>

Oleh :

ALVIN ANUGERAH PRATAMA

(22343019)

Dosen Pengampu :

RANDI PROSKA SANDRA, S.Pd., M.Sc.

(Seksi : 202313430043)

PRODI INFORMATIKA

DEPARTEMEN TEKNIK ELEKTRONIKA

FAKULTAS TEKNIK

UNIVERSITAS NEGERI PADANG

2023

A. Latar Belakang Aplikasi

Aplikasi Book Shop Management System adalah sebuah aplikasi yang dibangun agar dapat mempermudah dalam mengelola operasional toko buku. Tujuan dari aplikasi ini adalah membantu penjual dalam meneglola pejualan, jumlah buku, dan pelanggan dengan lebih efisisen. Aplikasi ini dapat digunakan untuk menghitung jumlah penjualan, pelanggan, pemasukan bahkan hingga melacak stok buku.

Book Shop Management System memiliki sejumlah fitur yang berguna bagi pemilik toko buku. Salah satu fitur utamanya adalah kemampuan mengelola inventaris. Sistem ini memungkinkan pemilik toko buku melacak jumlah buku, jumlah eksemplar, penulis, dan penerbit.

Fitur lainnya dari Book Shop Management System adalah kemampuan mengelola penjualan. Sistem ini memungkinkan pemilik toko buku untuk melacak penjualan, termasuk tanggal penjualan, dan buku yang dibeli. Fitur ini membantu pemilik toko buku mengetahui buku mana yang laku di pasaran dan mana yang tidak.

Kesimpulannya, Book Shop Management System adalah aplikasi yang dirancang untuk membantu pemilik toko buku mengelola inventaris, penjualan, dan pelanggan mereka dengan lebih efektif. Sistem ini memiliki sejumlah fitur yang berguna bagi pemilik toko buku. Dengan menggunakan sistem manajemen toko buku, pemilik toko buku dapat menghemat waktu dan meningkatkan operasional bisnisnya.

B. Unsur atau Konsep PBO Yang Dilibatkan

1. Inheritance

   Adalah sebuah konsep dimana sebuah kelas dapat mewarisi properti (variabel dan metode) dari kelas lain. Dalam Book Shop Management System, inheritance digunakan dalam membuat hirarki kelas, seperti pada kelas buku yang mewarisi dari kelas item yang bersifat umum

2. Polymorphism

   Adalah sebuah konsep dimana suatu objek dapat memiliki banyak bentuk atau perilaku. Dalam Book Shop Management System, polymorphism digunakan untuk mengizinkan beberapa jenis objek, seperti objek buku dan pengguna, untuk diperlakukan sebagai instance dari kelas yang lebih umum

3. Encapsulation

   Adalah sebuah konsep yang menggabungkan data (variabel) dan metode yang bberoperasi pada data tersebut ke dalam kelas. Dalam Book Shop Management System, encapsulation digunakan dalam melindungi data yang sensitif, seperti username dan password.

4. Abstraction

   Adalah sebuah konsep yang melakukan penyembunyian detail implementasi dan hanya mengekspos fungsi yang relavan dari suatu objek. Dalam Book Shop Management System, abstraction digunakan untuk membuat kelas yang umum seperti pada kelas buku.

5. Class, Object, Method

   Class adalah blue print yang menciptakan objek pada PBO. Object adalah instansi dari suatu Class. Method adalah untuk mendefinisikan prilaku atau tindakan yang dapat dilakukan objek tersebut. Dalam Book Shop Management System, kelas dapat mencakup buku, customer, pengguna, dll. Dan metode dapat mencakup menambah, menghapu, dan memperbarui data buku.

6. Property

   Adalah variabel atau atribut yang dimiliki suatu objek dalam suatu Class. Dalam Book Shop Management System, properti dapat berupa judul, penerbit, tahun terbit, harga buku, jumlah buku, dll.

7. Construcctor

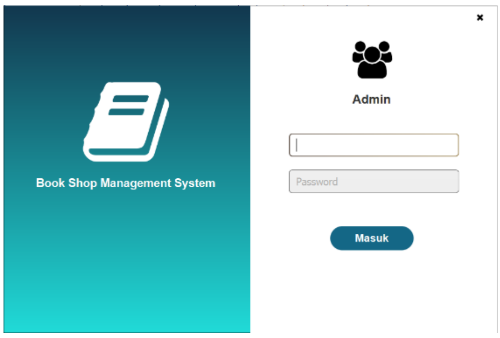
   Adalah sebuah metode khusus yang digunakan untuk menginisialisasi objek saat objek tersebut dibuat. Dalam Book Shop Management System, constructor digunakan dalam membuat objek buku baru dengan properti tertentu.

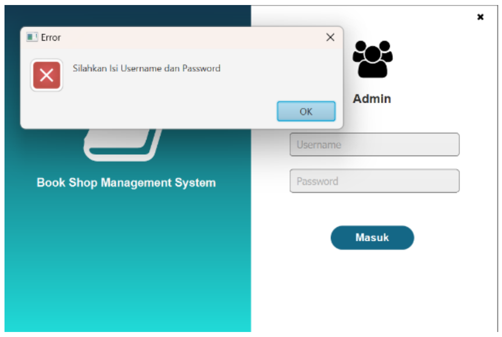
8. Visibility

   Adalah aturan yang mengatur sejauh mana suatu kelas, atribut atau metode dapat diakses oleh kelas-kelas lain dalam program. . Dalam Book Shop Management System, visibilitas digunakan untuk mengontrol akses suatu kelas terhadap atribut atau metode yang ada pada kelas lain.

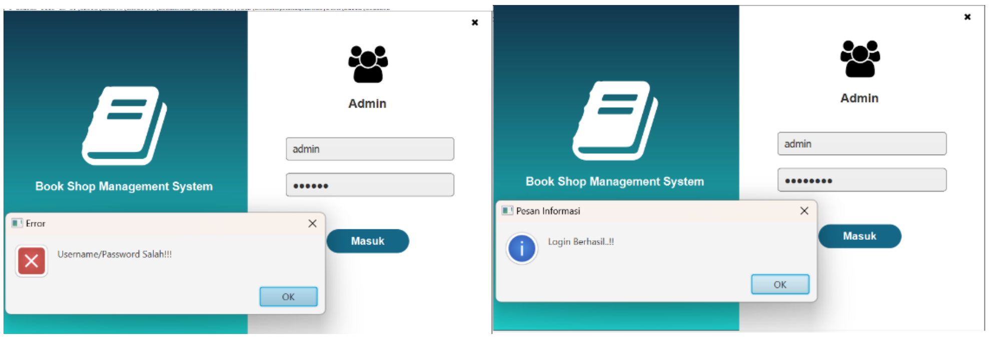
C. Penjelasan Aplikasi

1. Halaman Login



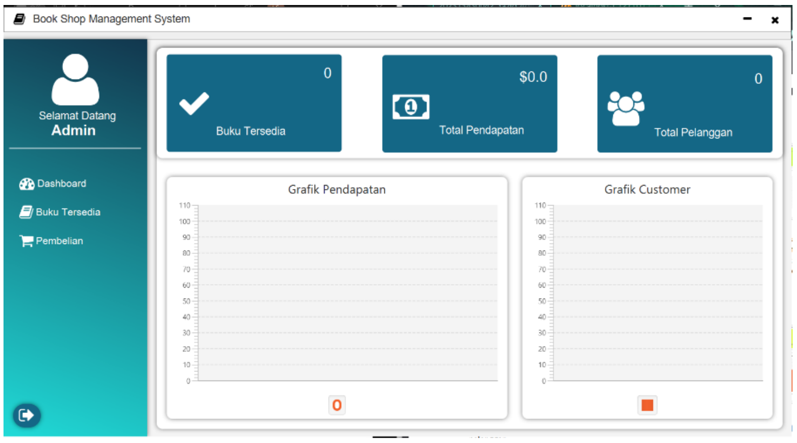


Silahkan Isi Username dan Password



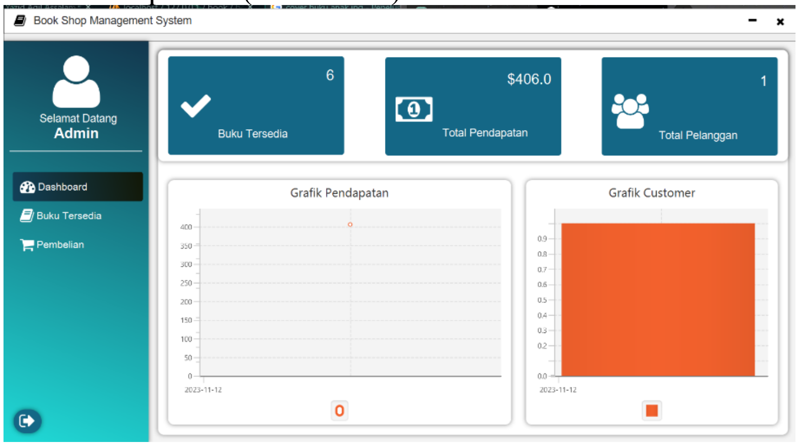
Halaman login adalah halaman awal yang dapat digunakan oleh pemilik toko atau oleh user untuk melakukan akses masuk kedalam aplikasi. Yang dapat melakukannya adalah yang telah terdaftar username dan passwordnya dalam database.

2. Halaman Aplikasi



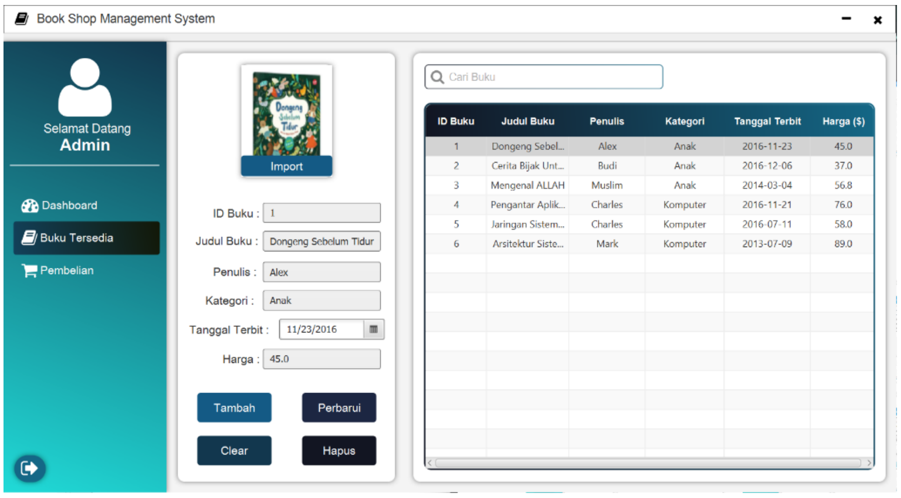
Halaman aplikasi adalah halaman yang akan muncul ketika user telah berhasil memasukkan username dan password dengan benar. Pada halaman ini user dapat melihat jumalh buku yang tersedia, total pendapatan, dan total pelanggan beserta grafiknya. Dan juga pada kiri bawah terdapat tombol log out dari aplikasi.

3. Halaman Aplikasi (Dashboard)



Bagian yang pertama adalah Dashboard / halaman utama dari aplikasi ini. User dapat melihat total buku yang tersedia, total pendapatan, dan total pelanggan serta grafiknya.

4. Halaman Aplikasi (Buku Tersedia)

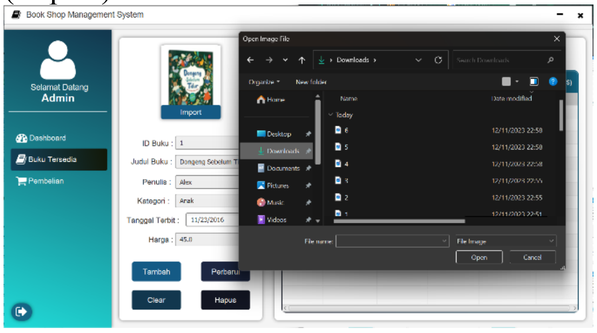


ID Buku : 1
Judul Buku : Dongeng Sebelum Tidur
Penulis : Alex
Kategori : Anak
Tanggal Terbit : 11/23/2016
Harga : 45.0

ID Buku	Judul Buku	Penulis	Kategori	Tanggal Terbit	Harga ($)
1	Dongeng Sebel...	Alex	Anak	2016-11-23	45.0
2	Cerita Bijak Unt...	Budi	Anak	2016-12-06	37.0
3	Mengenal ALLAH	Muslim	Anak	2014-03-04	56.8
4	Pengantar Aplik...	Charles	Komputer	2016-11-21	76.0
5	Jaringan Sistem...	Charles	Komputer	2016-07-11	58.0
6	Arsitektur Siste...	Mark	Komputer	2013-07-09	89.0

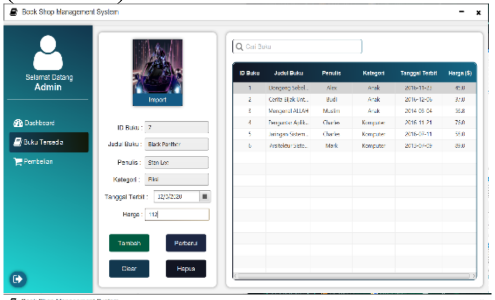
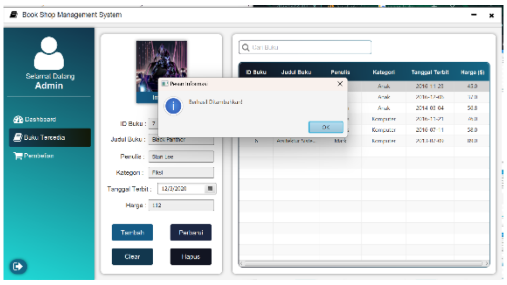
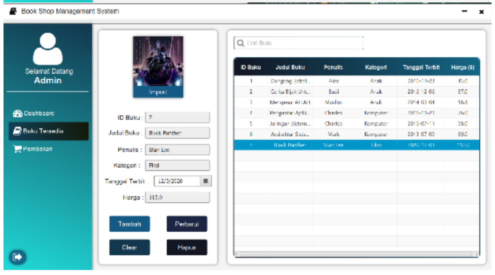
Bagian kedua adalah halaman yang menampilkan daftar buku yang tersedia beserta dengan penulis, judul, kategori tahun terbit hingga harga dari buku tersebut. Terdapat juga beberapa fitur yaitu import, tambah, perbarui, clear dan hapus. Yang mana keseluruhannya digunakan untuk mengatur data buku tersebut.

(Import)



Fitur ini digunakan untuk menambahkan gambar cover buku ke dalam aplikasi

(Tambah)

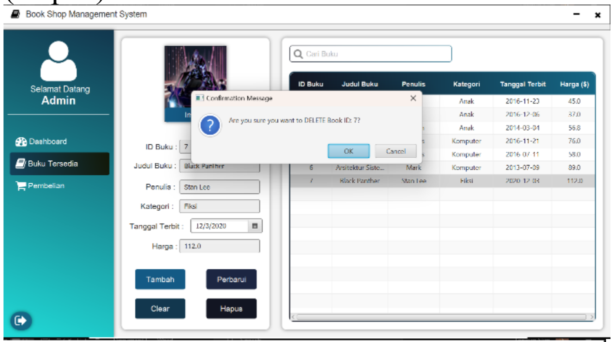
ID Buku : 7
Judul Buku : Black Panther
Penulis : Stan Lee
Kategori : Film
Tanggal Terbit : 12/3/2020
Harga : 112.0

Fitur ini digunakan untuk menambahkan data buku baru

(Perbarui) dan (Clear)

Fitur perbarui digunkan untuk memperbarui data buku yang telah ditambahkan sebekumnya dan fitur clear digunakan untuk membersihkan tempat input data buku

(Hapus)



Book Shop Management System

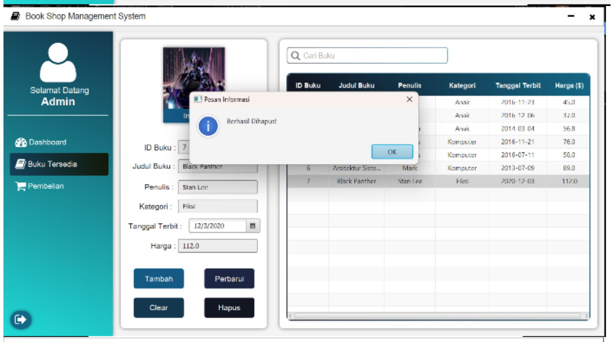
Confirmation Message

Are you sure you want to DELETE Book ID: 7?

OK
Cancel

ID Buku : 7
Judul Buku : Black Panther
Penulis : Stan Lee
Kategori : Fiksi
Tanggal Terbit : 12/3/2020
Harga : 112.0

ID Buku	Judul Buku	Penulis	Kategori	Tanggal Terbit	Harga ($)
			Anak	2016-11-23	45.0
			Anak	2016-12-06	37.0
			Anak	2014-03-04	56.8
			Komputer	2016-11-21	76.0
			Komputer	2016-07-11	58.0
6	Arsitektur Siste...	Mark	Komputer	2013-07-09	89.0
7	Black Panther	Stan Lee	Fiksi	2020-12-03	112.0

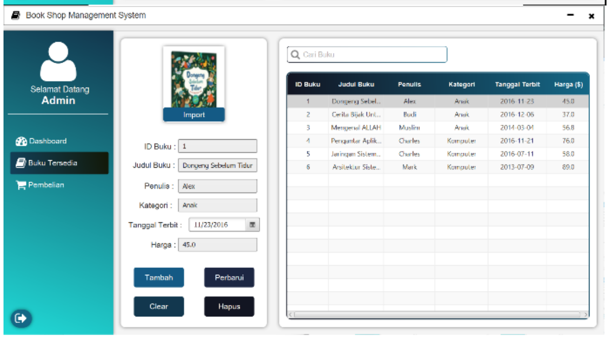


Book Shop Management System

Pesan Informasi

Berhasil Dihapus!

OK



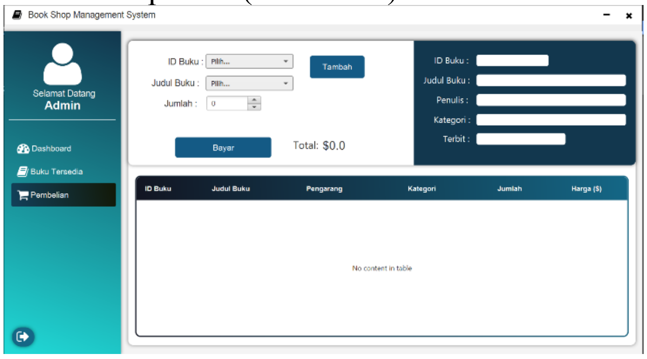
Book Shop Management System

ID Buku : 1
Judul Buku : Dongeng Sebelum Tidur
Penulis : Alex
Kategori : Anak
Tanggal Terbit : 11/23/2016
Harga : 45.0

ID Buku	Judul Buku	Penulis	Kategori	Tanggal Terbit	Harga ($)
1	Dongeng Sebel...	Alex	Anak	2016-11-23	45.0
2	Cerita Bijak Unt...	Budi	Anak	2016-12-05	37.0
3	Mengenal ALLAH	Muslim	Anak	2014-03-04	56.8
4	Pengantar Aplik...	Charles	Komputer	2016-11-21	76.0
5	Jaringan Sistem...	Charles	Komputer	2016-07-11	58.0
6	Arsitektur Siste...	Mark	Komputer	2013-07-09	89.0

Fitur ini digunakan untuk menghapus buku dari daftar buku yang tersedia apabila buku tersebut tidak dibutuhkan lagi

5. Halaman Aplikasi (Pembelian)



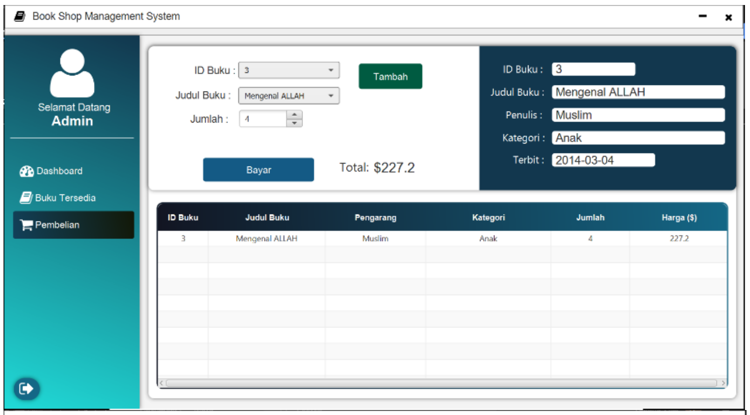
Book Shop Management System

ID Buku : Pilih...
Judul Buku : Pilih...
Jumlah : 0
Total: $0.0

ID Buku	Judul Buku	Pengarang	Kategori	Jumlah	Harga ($)

No content in table

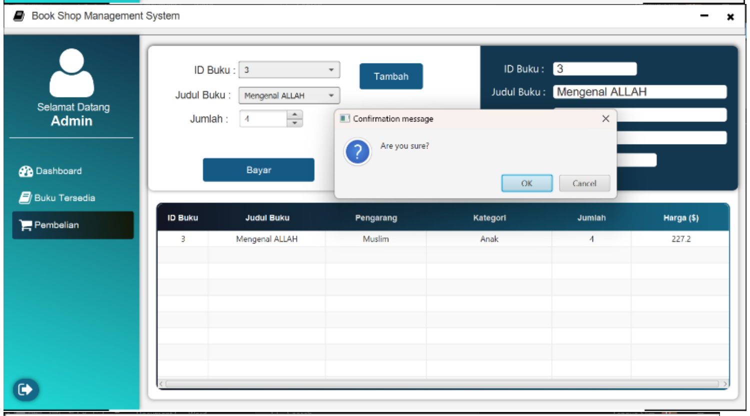
Halaman ini adalah halaman untuk melakukan transakasi pembelian buku yang diinginkan oleh konsumen. Pada halaman ini juga terdapat tota harga yang harus dibayar



ID Buku : 3
Judul Buku : Mengenal ALLAH
Jumlah : 4
Total: $227.2

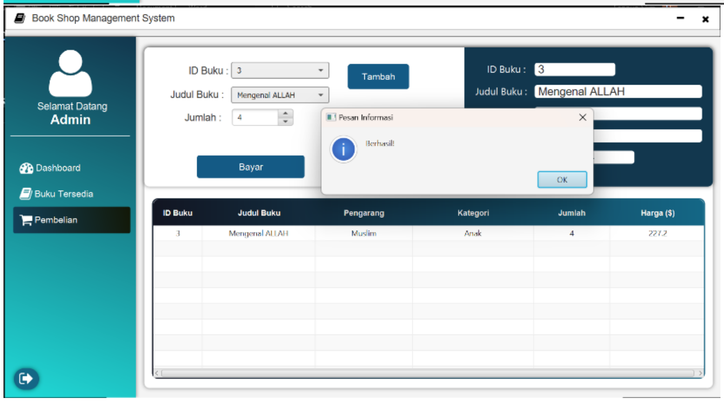
ID Buku : 3
Judul Buku : Mengenal ALLAH
Penulis : Muslim
Kategori : Anak
Terbit : 2014-03-04

ID Buku	Judul Buku	Pengarang	Kategori	Jumlah	Harga ($)
3	Mengenal ALLAH	Muslim	Anak	4	227.2



Are you sure?

ID Buku	Judul Buku	Pengarang	Kategori	Jumlah	Harga ($)
3	Mengenal ALLAH	Muslim	Anak	4	227.2

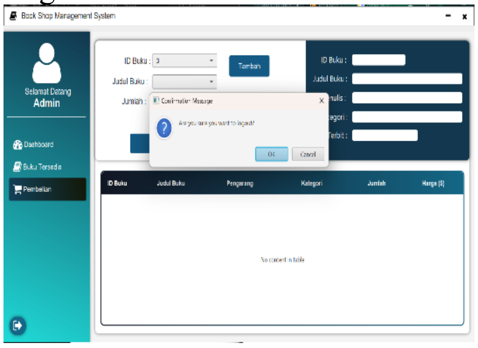


Berhasil!

ID Buku	Judul Buku	Pengarang	Kategori	Jumlah	Harga ($)
3	Mengenal ALLAH	Muslim	Anak	4	227.2

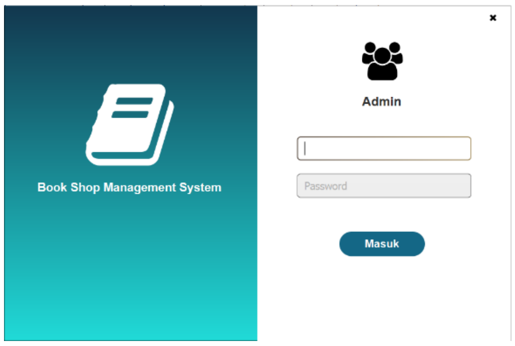
User akan menambahkan buku yang ingin dibeli dan jumlah yang ingin dibeli setelah itu klik tombol tambah, maka akan muncul dalam daftar buku yang dibeli dan untuk melakukan pembayaran klik tombol bayar lalu akan muncul pop up klik OK hingga muncul pesan berhasil yang menandakan transaksi berhasil.

6. Log Out



Are you sure you want to logout?

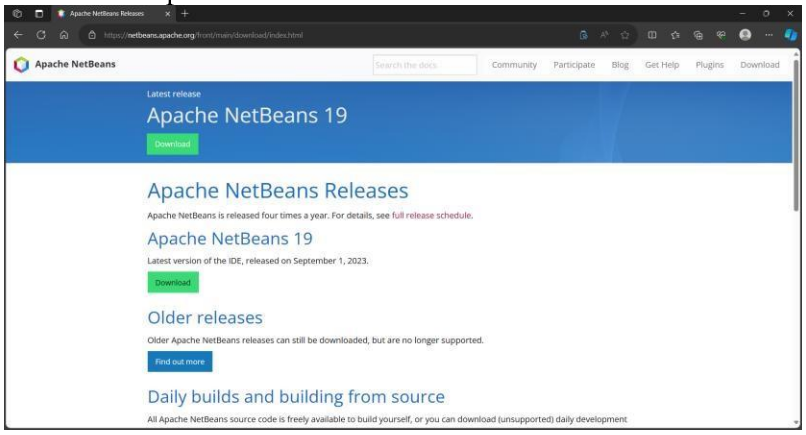
No content in table



Jika ingin keluar dari halaman aplikasi cukup dengan menekan tombol yang ada pada pojok kiri bawah, lalu akan muncul pesan apakah yakin ingin keluar lalu klik OK, maka kita akan kembali ke halaman login

D. Langkah-Langkah Pembuatan Aplikasi

1. Download Apache-neatbenas



https://netbeans.apache.org/front/main/download/index.html

Latest release

Apache NetBeans 19

Apache NetBeans Releases

Apache NetBeans is released four times a year. For details, see full release schedule.

Apache NetBeans 19

Latest version of the IDE, released on September 1, 2023.

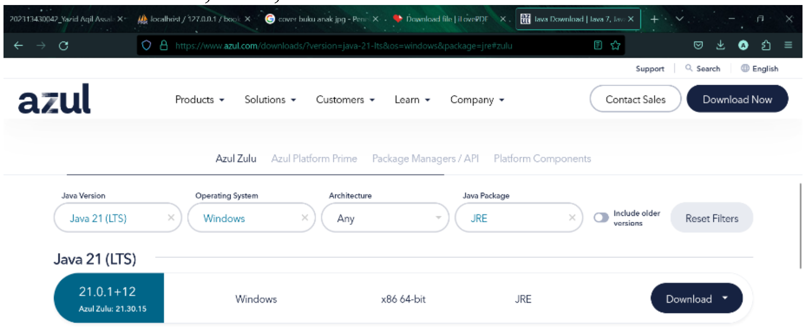
Older releases

Older Apache NetBeans releases can still be downloaded, but are no longer supported.

Daily builds and building from source

All Apache NetBeans source code is freely available to build yourself, or you can download (unsupported) daily development

2. Download JDK, JRE, dan JDK FX

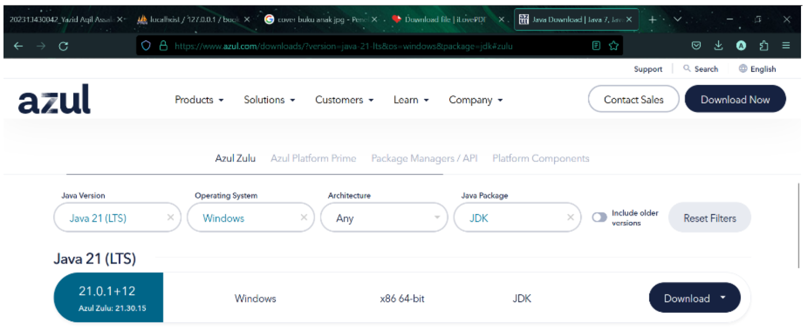


https://www.azul.com/downloads/?version=java-21-lts&os=windows&package=jre#zulu

Java Version: Java 21 (LTS)
Operating System: Windows
Architecture: Any
Java Package: JRE

Java 21 (LTS)

21.0.1+12 Azul Zulu: 21.30.15	Windows	x86 64-bit	JRE

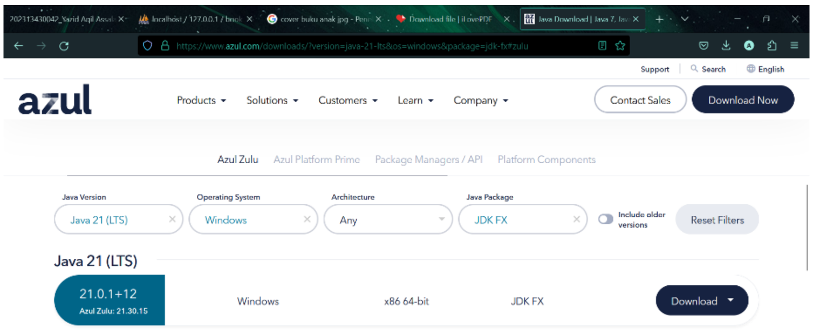


https://www.azul.com/downloads/?version=java-21-lts&os=windows&package=jdk#zulu

Java Version: Java 21 (LTS)
Operating System: Windows
Architecture: Any
Java Package: JDK

Java 21 (LTS)

21.0.1+12 Azul Zulu: 21.30.15	Windows	x86 64-bit	JDK



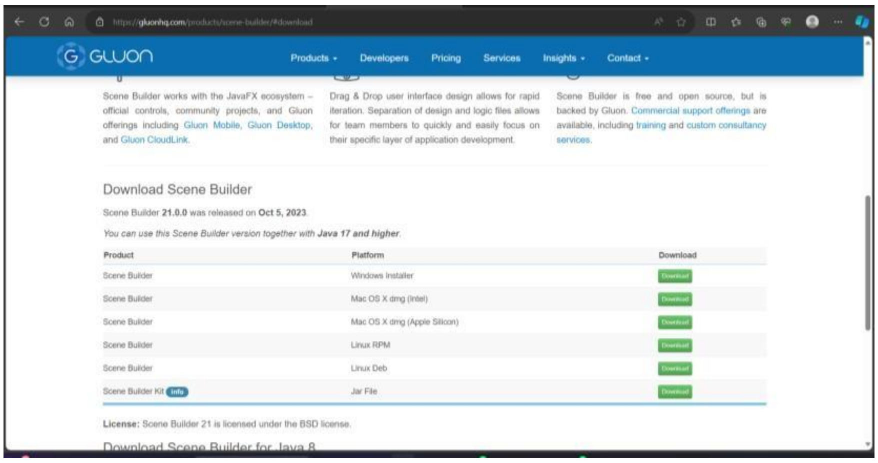
https://www.azul.com/downloads/?version=java-21-lts&os=windows&package=jdk-fx#zulu

Java Version: Java 21 (LTS)
Operating System: Windows
Architecture: Any
Java Package: JDK FX

Java 21 (LTS)

21.0.1+12 Azul Zulu: 21.30.15	Windows	x86 64-bit	JDK FX

3. Download Scene Builder



Scene Builder works with the JavaFX ecosystem – official controls, community projects, and Gluon offerings including Gluon Mobile, Gluon Desktop, and Gluon CloudLink.

Drag & Drop user interface design allows for rapid iteration. Separation of design and logic files allows for team members to quickly and easily focus on their specific layer of application development.

Scene Builder is free and open source, but is backed by Gluon. Commercial support offerings are available, including training and custom consultancy services.

Download Scene Builder

Scene Builder **21.0.0** was released on **Oct 5, 2023**.

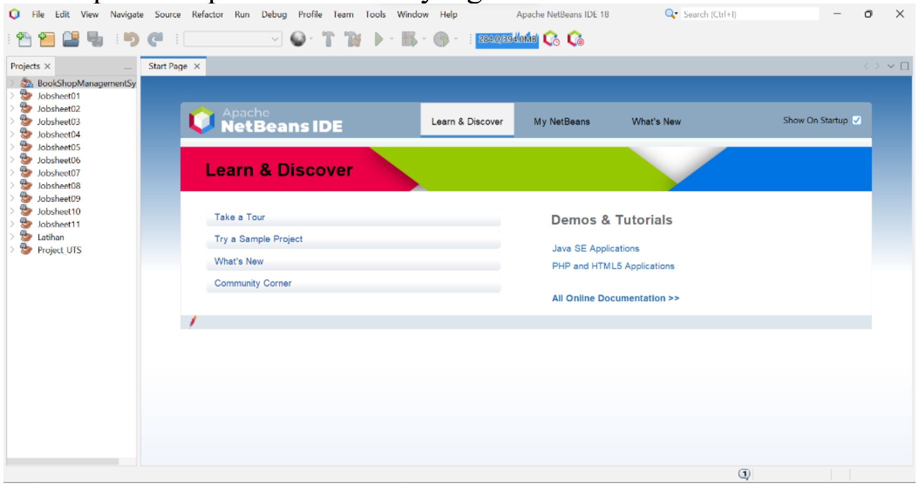
*You can use this Scene Builder version together with **Java 17 and higher**.*

Product	Platform	Download
Scene Builder	Windows Installer	Download
Scene Builder	Mac OS X dmg (Intel)	Download
Scene Builder	Mac OS X dmg (Apple Silicon)	Download
Scene Builder	Linux RPM	Download
Scene Builder	Linux Deb	Download
Scene Builder Kit info	Jar File	Download

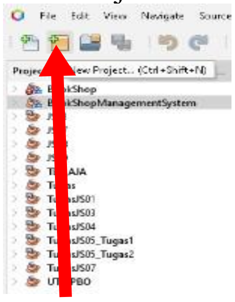
License: Scene Builder 21 is licensed under the BSD license.

Download Scene Builder for Java 8

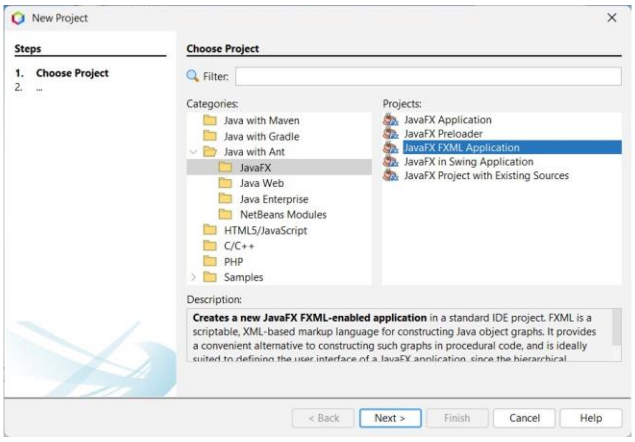
4. Instal semua aplikasi yang telah didownload tadi
5. Buka aplikasi Apache-Netbeans yang telah diinstal



6. Buat Project baru



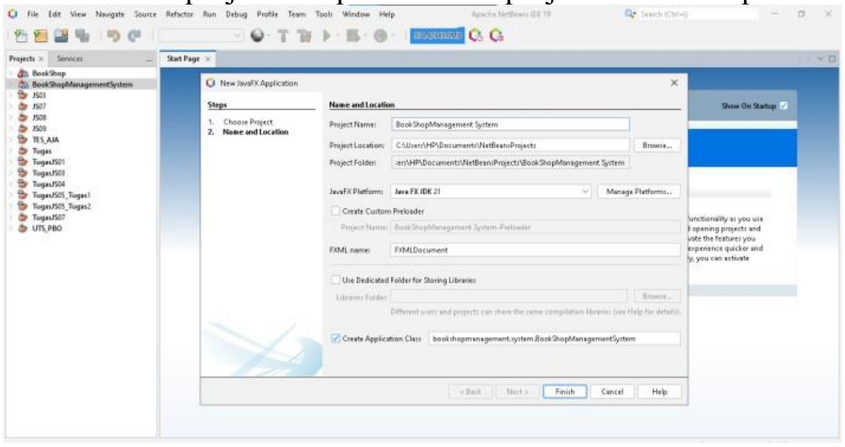
Lalu Pilih Java with Ant > JavaFX > JavaFX FXML Aplication



Description:

Creates a new JavaFX FXML-enabled application in a standard IDE project. FXML is a scriptable, XML-based markup language for constructing Java object graphs. It provides a convenient alternative to constructing such graphs in procedural code, and is ideally suited to defining the user interface of a JavaFX application, since the hierarchical

7. Lalu isi nama project dan pilih dimana project akan disimpan



New JavaFX Application

Steps
1. Choose Project
2. Name and Location

Name and Location

Project Name: BookShopManagement System

Project Location: C:\Users\HP\Documents\NetBeansProjects

Project Folder: ers\HP\Documents\NetBeansProjects\BookShopManagement System

JavaFX Platform: Java FX JDK 21

Create Custom Preloader

Project Name: BookShopManagement System-Preloader

FXML name: FXMLDocument

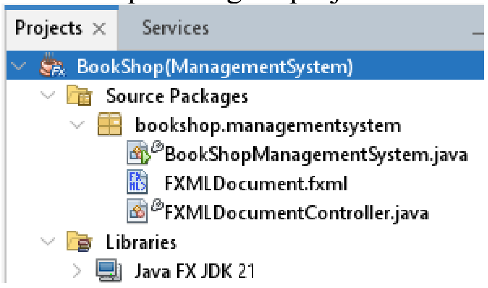
Use Dedicated Folder for Storing Libraries

Libraries Folder:

Different users and projects can share the same compilation libraries (see Help for details).

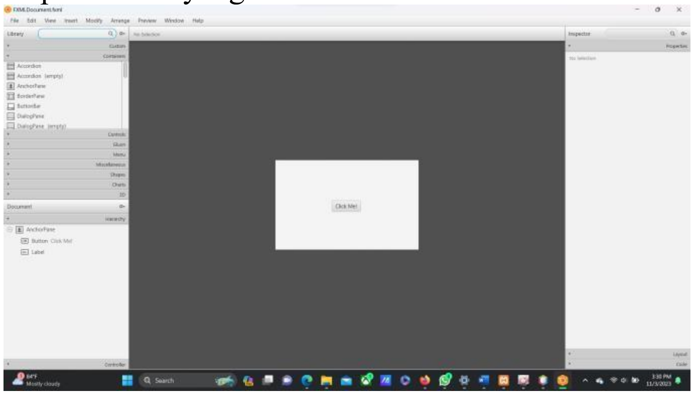
Create Application Class: bookshopmanagement.system.BookShopManagementSystem

8. Buka file pada bagian project

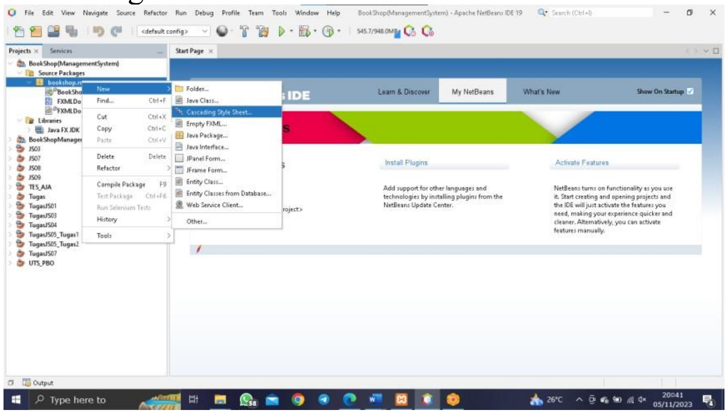


Projects
Services
BookShop(ManagementSystem)
Source Packages
bookshop.managementsystem
BookShopManagementSystem.java
FXMLDocument.fxml
FXMLDocumentController.java
Libraries
Java FX JDK 21

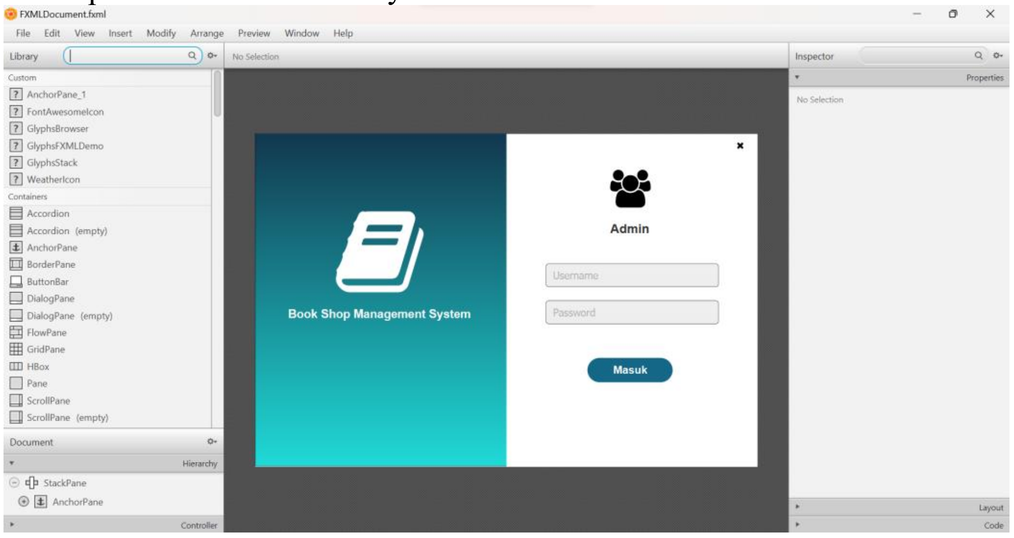
9. Masuk ke FXMLDocument.fxml dengan klik dua kali pada file
10. Hapus semua yang awal ada di scene builder



11. Buatlah file baru yang berupa file CSS baru, lalu buatlah dengan nama DesainLogin.css



12. Buatlah tampilan sesuai dengan yang ditunjukkan dalam video dan tambahkan beberapa kreativitas didalamnya



13. Buatlah seluruh source code yang dibutuhkan dalam pembuatan aplikasi Book Shop Management System sesuai dengan video yang ditonton, seperti berikut:

a. DesainLogin.css > Kodingan CSS untuk tampilan halaman login

```
.form-kiri{
    -fx-background-color:linear-gradient(to top,
#20dbd8, #12374e);
    -fx-border-color: #000;
    -fx-border-width: .4px 0px .4px .4px;
}
.form-kanan{
    -fx-background-color:#fff;
    -fx-border-color: #000;
    -fx-border-width: .4px .4px .4px 0px;
}
.tombol_close{
    -fx-background-color: transparent;
    -fx-cursor:hand;
}
.tombol_close:hover{
    -fx-background-color:#b10c0c;
}
.teks{
    -fx-background-color:linear-gradient(to bottom,
#efefef, #eee);
    -fx-background-radius:4px;
    -fx-font-size:13px;
    -fx-border-color: #000;
    -fx-border-width: .4px;
    -fx-border-radius: 4px;
    -fx-font-family: Tahoma;
}
.teks:focused{
    -fx-background-color: #fff;
    -fx-border-color:linear-gradient(to top right,
#3c2c21, #93773e);
    -fx-border-width: 1px;
}
.tombol_login{
    -fx-background-color:#146786;
    -fx-background-radius: 50px;
    -fx-cursor:hand;
    -fx-text-fill: #fff;
    -fx-font-size: 14px;
    -fx-font-weight: bold;
    -fx-font-family: "Arial";
}
.tombol_login:hover{
    -fx-background-color:#12374e;
}
```

b. FXMLDocumentController.java

```
package bookshopmanagementsystem;

import java.net.URL;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.ResourceBundle;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.fxml.Initializable;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.Alert;
import javafx.scene.control.Alert.AlertType;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.PasswordField;
import javafx.scene.control.TextField;
import javafx.scene.input.MouseEvent;
import javafx.scene.layout.AnchorPane;
import javafx.stage.Stage;
import javafx.stage.StageStyle;


/**
 *
 * @author Lenovo
 */
public class FXMLDocumentController implements
Initializable {

    @FXML
    private AnchorPane main_form;

    @FXML
    private TextField username;

    @FXML
    private PasswordField password;

    @FXML
    private Button loginBtn;

    @FXML
    private Button close;

    private Connection connect;
    private PreparedStatement prepare;
    private ResultSet result;
    private double x = 0;
    private double y = 0;

    public void loginAdmin(){

        connect = database.connectDb();

        String sql = "SELECT * FROM admin WHERE 
username = ? and password = ?"; // admin is our table 
name

        try{
            Alert alert;

            prepare = connect.prepareStatement(sql);
            prepare.setString(1, username.getText());
            prepare.setString(2, password.getText());

            result = prepare.executeQuery();

            if(username.getText().isEmpty() || 
password.getText().isEmpty()){
                alert = new Alert(AlertType.ERROR);
                alert.setTitle("Error");
                alert.setHeaderText(null);
                alert.setContentText("Silahkan Isi 
Username dan Password");
                alert.showAndWait();
            }else{
                if(result.next()){
                    // IF CORRECT USERNAME AND PASSWORD

                    getData.username = 
username.getText();

                    alert = new 
Alert(AlertType.INFORMATION);
                    alert.setTitle("Pesan Informasi");
                    alert.setHeaderText(null);
                    alert.setContentText("Login 
Berhasil..!!");
                    alert.showAndWait();

                    // TO HIDE YOUR LOGIN FORM
                    
loginBtn.getScene().getWindow().hide();

                    // LINK YOUR DASHBOARD FORM : )
                    Parent root = 
FXMLLoader.load(getClass().getResource("dashboard.fxml"
));
                    Stage stage = new Stage();
                    Scene scene = new Scene(root);

                    root.setOnMousePressed((MouseEvent 
event) ->{
                        x = event.getSceneX();
                        y = event.getSceneY();
                    });

                    root.setOnMouseDragged((MouseEvent 
event) ->{
```

```
                        stage.setX(event.getScreenX() -
x);
                        stage.setY(event.getScreenY() -
y);
                    });


stage.initStyle(StageStyle.TRANSPARENT);

                    stage.setScene(scene);
                    stage.show();

                }else{ // IF WRONG USERNAME OR PASSWORD
                    alert = new Alert(AlertType.ERROR);
                    alert.setTitle("Error ");
                    alert.setHeaderText(null);

alert.setContentText("Username/Password Salah!!!");
                    alert.showAndWait();
                }
            }

        }catch(Exception e){e.printStackTrace();}

    }

    public void close(){
        System.exit(0);
    }

    @Override
    public void initialize(URL url, ResourceBundle rb)
{
        // TODO
    }

}
```

c. BookShhopManagementSystem.java

```
package bookshopmanagementsystem;


import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.input.MouseEvent;
import javafx.stage.Stage;
import javafx.stage.StageStyle;

/**
*
@author Lenovo
*/
public class BookShopManagementSystem extends Application {

private double x = 0;
private double y = 0;

@Override
public void start(Stage stage) throws Exception {
```

```
Parent root =
FXMLLoader.load(getClass().getResource("FXMLDocument.fxml"));

Scene scene = new Scene(root);

root.setOnMousePressed((MouseEvent event) ->{
x = event.getSceneX();
y = event.getSceneY();
});

root.setOnMouseDragged((MouseEvent event) ->{
stage.setX(event.getScreenX() - x);
stage.setY(event.getScreenY() - y);

stage.setOpacity(.8);
});

root.setOnMouseReleased((MouseEvent event) ->{
stage.setOpacity(1);
});

stage.initStyle(StageStyle.TRANSPARENT);

stage.setScene(scene);
stage.show();
}

/**
@param args the command line arguments
*/
public static void main(String[] args) {
launch(args);
}
}
```

d. DataBuku.java

```
package bookshopmanagementsystem;
import java.sql.Date;
/**
 *
 * @author Lenovo
 */
public class DataBuku {
    private Integer bookId;
    private String title;
    private String author;
    private String genre;
    private Date date;
    private Double price;
    private String image;
    // MAKE SURE YOU FOLLOWED THE PARAMETERS THAT I PUT
    public DataBuku(Integer bookId, String title, String
author, String genre, Date date, Double price, String
image){
        this.bookId = bookId;
        this.title = title;
        this.author = author;
        this.genre = genre;
        this.date = date;
        this.price = price;
        this.image = image;
    }
    public Integer getBookId(){
        return bookId;
    }
    public String getTitle(){
        return title;
    }
    public String getAuthor(){
        return author;
    }
    public String getGenre(){
        return genre;
    }
    public Date getDate(){
        return date;
    }
    public Double getPrice(){
        return price;
    }
    public String getImage(){
        return image;
    }
}
```

e. DataCustomer.java

```
package bookshopmanagementsystem;
import java.sql.Date;
/**
 *
 * @author Lenovo
 */
public class DataCustomer {
    private Integer customerId;
    private Integer bookId;
    private String title;
    private String author;
    private String genre;
    private Integer quantity;
    private Double price;
    private Date date;

    public DataCustomer(Integer customerId, Integer
bookId, String title, String author
            , String genre, Integer quantity, Double
price, Date date){
        this.customerId = customerId;
        this.bookId = bookId;
        this.title = title;
        this.author = author;
        this.genre = genre;
        this.quantity = quantity;
        this.price = price;
        this.date = date;
    }
    public Integer getCustomerId(){
        return customerId;
    }
    public Integer getBookId(){
        return bookId;
    }
    public String getTitle(){
        return title;
    }
    public String getAuthor(){
        return author;
    }
    public String getGenre(){
        return genre;
    }
    public Integer getQuantity(){
        return quantity;
    }
    public Double getPrice(){
        return price;
    }
    public Date getDate(){
        return date;
    }
}
```

f. dashboardController.java

```
package bookshopmanagementsystem;

import java.io.File;
import java.net.URL;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import java.time.LocalDate;
import java.util.Date;
import java.util.Optional;
import java.util.ResourceBundle;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.collections.transformation.FilteredList;
import javafx.collections.transformation.SortedList;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.fxml.Initializable;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.chart.AreaChart;
import javafx.scene.chart.BarChart;
import javafx.scene.chart.XYChart;
import javafx.scene.control.Alert;
import javafx.scene.control.Alert.AlertType;
import javafx.scene.control.Button;
import javafx.scene.control.ButtonType;
import javafx.scene.control.ComboBox;
import javafx.scene.control.DatePicker;
import javafx.scene.control.Label;
import javafx.scene.control.Spinner;
import javafx.scene.control.SpinnerValueFactory;
import javafx.scene.control.TableColumn;
import javafx.scene.control.TableView;
import javafx.scene.control.TextField;
import javafx.scene.control.cell.PropertyValueFactory;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
import javafx.scene.input.MouseEvent;
import javafx.scene.layout.AnchorPane;
import javafx.stage.FileChooser;
import javafx.stage.FileChooser.ExtensionFilter;
import javafx.stage.Stage;
import javafx.stage.StageStyle;

/**
*
      @author Lenovo
*/
public class dashboardController implements Initializable{

@FXML
private AnchorPane main_form;

@FXML
private Button close;

@FXML
private Button minimize;

@FXML
private Label username;

@FXML
private Button dashboard_btn;

@FXML
private Button availableBooks_btn;

@FXML
private Button purchase_btn;

@FXML
private Button logout;

@FXML
private AnchorPane dashboard_form;

@FXML
private Label dashboard_AB;

@FXML
private Label dashboard_TI;

@FXML
private Label dashboard_TC;

@FXML
private AreaChart<?, ?> dashboard_incomeChart;

@FXML
private BarChart<?, ?> dashboard_customerChart;

@FXML
private AnchorPane availableBooks_form;

@FXML
private ImageView availableBooks_imageView;

@FXML
private Button availableBooks_importBtn;
```

```
    @FXML
    private TextField availableBooks_bookID;

    @FXML
    private TextField availableBooks_bookTitle;

    @FXML
    private TextField availableBooks_author;

    @FXML
    private TextField availableBooks_genre;

    @FXML
    private DatePicker availableBooks_date;

    @FXML
    private TextField availableBooks_price;

    @FXML
    private Button availableBooks_addBtn;

    @FXML
    private Button availableBooks_updateBtn;

    @FXML
    private Button availableBooks_clearBtn;

    @FXML
    private Button availableBooks_deleteBtn;

    @FXML
    private TextField availableBooks_search;

    @FXML
    private TableView<DataBuku> availableBooks_tableView;

    @FXML
    private TableColumn<DataBuku, String>
    availableBooks_col_bookID;

    @FXML
    private TableColumn<DataBuku, String>
    availableBooks_col_bookTItle;

    @FXML
    private TableColumn<DataBuku, String>
    availableBooks_col_author;

    @FXML
    private TableColumn<DataBuku, String>
    availableBooks_col_genre;

    @FXML
    private TableColumn<DataBuku, String>
    availableBooks_col_date;

    @FXML
    private TableColumn<DataBuku, String>
    availableBooks_col_price;

    @FXML
    private AnchorPane purchase_form;
    @FXML
    private ComboBox<?> purchase_bookID;

    @FXML
    private ComboBox<?> purchase_bookTitle;

    @FXML
    private Label purchase_total;

    @FXML
    private Button purchase_addBtn;

    @FXML
    private Label purchase_info_bookID;

    @FXML
    private Label purchase_info_bookTItle;

    @FXML
    private Label purchase_info_author;

    @FXML
    private Label purchase_info_genre;

    @FXML
    private Label purchase_info_date;

    @FXML
    private Button purchase_payBtn;

    @FXML
    private TableView<DataCustomer> purchase_tableView;

    @FXML
    private Spinner<Integer> purchase_quantity;

    @FXML
    private TableColumn<DataCustomer, String>
    purchase_col_bookID;

    @FXML
    private TableColumn<DataCustomer, String>
    purchase_col_bookTitle;

    @FXML
    private TableColumn<DataCustomer, String>
    purchase_col_author;

    @FXML
    private TableColumn<DataCustomer, String>
    purchase_col_genre;

    @FXML
    private TableColumn<DataCustomer, String>
    purchase_col_quantity;

    @FXML
    private TableColumn<DataCustomer, String>
    purchase_col_price;

    private Connection connect;
    private PreparedStatement prepare;
    private Statement statement;
```

```
private ResultSet result;

private Image image;

public void dashboardAB(){

String sql = "SELECT COUNT(id) FROM book";

connect = database.connectDb();
int countAB = 0;
try{
prepare = connect.prepareStatement(sql);
result = prepare.executeQuery();

if(result.next()){
countAB = result.getInt("COUNT(id)");
}

dashboard_AB.setText(String.valueOf(countAB));

}catch(Exception e){e.printStackTrace();}
}

public void dashboardTI(){

String sql = "SELECT SUM(total) FROM customer_info";

connect = database.connectDb();
double sumTotal = 0;
try{
prepare = connect.prepareStatement(sql);
result = prepare.executeQuery();

if(result.next()){
sumTotal = result.getDouble("SUM(total)");
}

dashboard_TI.setText("$" + String.valueOf(sumTotal));

}catch(Exception e){e.printStackTrace();}
}

public void dashboardTC(){
String sql = "SELECT COUNT(id) FROM customer_info";

connect = database.connectDb();
int countTC = 0;
try{
prepare = connect.prepareStatement(sql);
result = prepare.executeQuery();

if(result.next()){
countTC = result.getInt("COUNT(id)");
}

dashboard_TC.setText(String.valueOf(countTC));

}catch(Exception e){e.printStackTrace();}

}

public void dashboardIncomeChart(){
```

```
dashboard_incomeChart.getData().clear();

String sql = "SELECT date, SUM(total) FROM customer_info
GROUP BY date ORDER BY TIMESTAMP(date) ASC LIMIT 6";

connect = database.connectDb();

try{
XYChart.Series chart = new XYChart.Series();

prepare = connect.prepareStatement(sql);
result = prepare.executeQuery();

while(result.next()){
chart.getData().add(new XYChart.Data(result.getString(1),
result.getInt(2)));
}

dashboard_incomeChart.getData().add(chart);

}catch(Exception e){e.printStackTrace();}

}


public void dashboardCustomerChart(){

dashboard_customerChart.getData().clear();

String sql = "SELECT date, COUNT(id) FROM customer_info
GROUP BY date ORDER BY TIMESTAMP(date) ASC LIMIT 4";

connect = database.connectDb();

try{
XYChart.Series chart = new XYChart.Series();

prepare = connect.prepareStatement(sql);
result = prepare.executeQuery();

while(result.next()){
chart.getData().add(new XYChart.Data(result.getString(1),
result.getInt(2)));
}

dashboard_customerChart.getData().add(chart);

}catch(Exception e){e.printStackTrace();}

} // THATS IT FOR THIS VIDEO, THANKS FOR WATCHING! HOPE YOU
LIKE IT : )
// SUBSCRIBE OUR CHANNEL FOR MORE COOL PROJECT TUTORIALS
// THANKS FOR THE SUPPORT! <3

public void availableBooksAdd(){

String sql = "INSERT INTO book (book_id, title, author,
genre, pub_date, price, image) "
+ "VALUES(?,?,?,?,?,?,?)";

connect = database.connectDb();

try{
```

```
Alert alert;

if(availableBooks_bookID.getText().isEmpty()
|| availableBooks_bookTitle.getText().isEmpty()
|| availableBooks_author.getText().isEmpty()
|| availableBooks_genre.getText().isEmpty()
|| availableBooks_date.getValue() == null
|| availableBooks_price.getText().isEmpty()
|| getData.path == null || getData.path == ""){
alert = new Alert(AlertType.ERROR);
alert.setTitle("Error Message");
alert.setHeaderText(null);
alert.setContentText("Please fill all blank fields");
alert.showAndWait();
}else{
// CHECK IF BOOK ID IS ALREADY EXIST
String checkData = "SELECT book_id FROM book WHERE book_id =
'"
+availableBooks_bookID.getText()+"'";

statement = connect.createStatement();
result = statement.executeQuery(checkData);

if(result.next()){
alert = new Alert(AlertType.ERROR);
alert.setTitle("Error Message");
alert.setHeaderText(null);
alert.setContentText("Book ID: " +
availableBooks_bookID.getText() + " was already exist!");
alert.showAndWait();
}else{

prepare = connect.prepareStatement(sql);
prepare.setString(1, availableBooks_bookID.getText());
prepare.setString(2, availableBooks_bookTitle.getText());
prepare.setString(3, availableBooks_author.getText());
prepare.setString(4, availableBooks_genre.getText());
prepare.setString(5,
String.valueOf(availableBooks_date.getValue()));
prepare.setString(6, availableBooks_price.getText());

String uri = getData.path;
uri = uri.replace("\\", "\\\\");

prepare.setString(7, uri);

prepare.executeUpdate();

alert = new Alert(AlertType.INFORMATION);
alert.setTitle("Pesan Informasi");
alert.setHeaderText(null);
alert.setContentText("Berhasil Ditambahkan!");
alert.showAndWait();

// TO BE UPDATED THE TABLEVIEW
availableBooksShowListData();
// CLEAR FIELDS
availableBooksClear();
}
}
}catch(Exception e){e.printStackTrace();}

}
```

```
public void availableBooksUpdate(){

String uri = getData.path;
uri = uri.replace("\\", "\\\\");

String sql = "UPDATE book SET title = '"
+availableBooks_bookTitle.getText()+"', author = '"
+availableBooks_author.getText()+"', genre = '"
+availableBooks_genre.getText()+"', pub_date = '"
+availableBooks_date.getValue()+"', price = '"
+availableBooks_price.getText()+"', image = '"
+uri+"' WHERE book_id =
'"+availableBooks_bookID.getText()+"'";

connect = database.connectDb();

try{
Alert alert;

if(availableBooks_bookID.getText().isEmpty()
|| availableBooks_bookTitle.getText().isEmpty()
|| availableBooks_author.getText().isEmpty()
|| availableBooks_genre.getText().isEmpty()
|| availableBooks_date.getValue() == null
|| availableBooks_price.getText().isEmpty()
|| getData.path == null || getData.path == ""){
alert = new Alert(AlertType.ERROR);
alert.setTitle("Error Message");
alert.setHeaderText(null);
alert.setContentText("Please fill all blank fields");
alert.showAndWait();
}else{
alert = new Alert(AlertType.CONFIRMATION);
alert.setTitle("Confirmation Message");
alert.setHeaderText(null);
alert.setContentText("Are you sure you want to UPDATE Book
ID: " + availableBooks_bookID.getText() + "?");
Optional<ButtonType> option = alert.showAndWait();

if(option.get().equals(ButtonType.OK)){
statement = connect.createStatement();
statement.executeUpdate(sql);

alert = new Alert(AlertType.INFORMATION);
alert.setTitle("Pesan Informasi");
alert.setHeaderText(null);
alert.setContentText("Berhasil Diperbarui!");
alert.showAndWait();

// TO BE UPDATED THE TABLEVIEW
availableBooksShowListData();
// CLEAR FIELDS
availableBooksClear();
}
}
}catch(Exception e){e.printStackTrace();}

}

public void availableBooksDelete(){

String sql = "DELETE FROM book WHERE book_id = '"
```

```
+availableBooks_bookID.getText()+"'";

connect = database.connectDb();

try{
Alert alert;

if(availableBooks_bookID.getText().isEmpty()
|| availableBooks_bookTitle.getText().isEmpty()
|| availableBooks_author.getText().isEmpty()
|| availableBooks_genre.getText().isEmpty()
|| availableBooks_date.getValue() == null
|| availableBooks_price.getText().isEmpty()
|| getData.path == null || getData.path == ""){
alert = new Alert(AlertType.ERROR);
alert.setTitle("Error Message");
alert.setHeaderText(null);
alert.setContentText("Please fill all blank fields");
alert.showAndWait();
}else{
alert = new Alert(AlertType.CONFIRMATION);
alert.setTitle("Confirmation Message");
alert.setHeaderText(null);
alert.setContentText("Are you sure you want to DELETE Book
ID: " + availableBooks_bookID.getText() + "?");
Optional<ButtonType> option = alert.showAndWait();

if(option.get().equals(ButtonType.OK)){
statement = connect.createStatement();
statement.executeUpdate(sql);

alert = new Alert(AlertType.INFORMATION);
alert.setTitle("Pesan Informasi");
alert.setHeaderText(null);
alert.setContentText("Berhasil Dihapus!");
alert.showAndWait();

// TO BE UPDATED THE TABLEVIEW
availableBooksShowListData();
// CLEAR FIELDS
availableBooksClear();
}
}
}catch(Exception e){e.printStackTrace();}

}

public void availableBooksClear(){
availableBooks_bookID.setText("");
availableBooks_bookTitle.setText("");
availableBooks_author.setText("");
availableBooks_genre.setText("");
availableBooks_date.setValue(null);
availableBooks_price.setText("");

getData.path = "";

availableBooks_imageView.setImage(null);
}

public void avaialableBooksInsertImage(){

FileChooser open = new FileChooser();
```

```
open.setTitle("Open Image File");
open.getExtensionFilters().add(new ExtensionFilter("File
Image", "*jpg", "*png"));

File file =
open.showOpenDialog(main_form.getScene().getWindow());

if(file != null){
getData.path = file.getAbsolutePath();

image = new Image(file.toURI().toString(), 112, 137, false,
true);
availableBooks_imageView.setImage(image);
}

}

public ObservableList<DataBuku> availableBooksListData(){

ObservableList<DataBuku> listData =
FXCollections.observableArrayList();
String sql = "SELECT * FROM book";

connect = database.connectDb();

try{
prepare = connect.prepareStatement(sql);
result = prepare.executeQuery();

DataBuku bookD;

while(result.next()){
bookD = new DataBuku(result.getInt("book_id"),
result.getString("title")
, result.getString("author"), result.getString("genre")
, result.getDate("pub_date"), result.getDouble("price")
, result.getString("image"));

listData.add(bookD);
}
}catch(Exception e){e.printStackTrace();}
return listData;
}

private ObservableList<DataBuku> availableBooksList;
public void availableBooksShowListData(){
availableBooksList = availableBooksListData();

availableBooks_col_bookID.setCellValueFactory(new
PropertyValueFactory<>("bookId"));
availableBooks_col_bookTItle.setCellValueFactory(new
PropertyValueFactory<>("title"));
availableBooks_col_author.setCellValueFactory(new
PropertyValueFactory<>("author"));
availableBooks_col_genre.setCellValueFactory(new
PropertyValueFactory<>("genre"));
availableBooks_col_date.setCellValueFactory(new
PropertyValueFactory<>("date"));
availableBooks_col_price.setCellValueFactory(new
PropertyValueFactory<>("price"));

availableBooks_tableView.setItems(availableBooksList);
}
```

```
public void availableBooksSelect(){
DataBuku bookD =
availableBooks_tableView.getSelectionModel().getSelectedItem
();
int num =
availableBooks_tableView.getSelectionModel().getSelectedInde
x();

if((num - 1) < -1){ return; }

availableBooks_bookID.setText(String.valueOf(bookD.getBookId
()));
availableBooks_bookTitle.setText(bookD.getTitle());
availableBooks_author.setText(bookD.getAuthor());
availableBooks_genre.setText(bookD.getGenre());
availableBooks_date.setValue(LocalDate.parse(String.valueOf(
bookD.getDate())));
availableBooks_price.setText(String.valueOf(bookD.getPrice()
));

getData.path = bookD.getImage();

String uri = "file:" + bookD.getImage();

image = new Image(uri, 112, 137, false, true);

availableBooks_imageView.setImage(image);
}

public void availableBooksSeach(){

FilteredList<DataBuku> filter = new
FilteredList<>(availableBooksList, e -> true);

availableBooks_search.textProperty().addListener((Observable
, oldValue, newValue) ->{

filter.setPredicate(predicateBookData -> {

if(newValue == null || newValue.isEmpty()){
return true;
}

String searchKey = newValue.toLowerCase();

if(predicateBookData.getBookId().toString().contains(searchK
ey)){
return true;
}else
if(predicateBookData.getTitle().toLowerCase().contains(searc
hKey)){
return true;
}else
if(predicateBookData.getAuthor().toLowerCase().contains(sear
chKey)){
return true;
}else
if(predicateBookData.getGenre().toLowerCase().contains(searc
hKey)){
return true;
}else
if(predicateBookData.getDate().toString().contains(searchKey
)){
return true;
}else
if(predicateBookData.getPrice().toString().contains(searchKe
y)){
return true;
}else return false;
});
});

SortedList<DataBuku> sortList = new SortedList(filter);
sortList.comparatorProperty().bind(availableBooks_tableView.
comparatorProperty());
availableBooks_tableView.setItems(sortList);

}

private double totalP;
public void purchaseAdd(){
purchasecustomerId();

String sql = "INSERT INTO customer (customer_id, book_id,
title, author, genre, quantity, price, date) "
+ "VALUES(?,?,?,?,?,?,?,?)";

connect = database.connectDb();

try{
Alert alert;

if(purchase_bookTitle.getSelectionModel().getSelectedItem()
== null
|| purchase_bookID.getSelectionModel().getSelectedItem() ==
null){
alert = new Alert(AlertType.ERROR);
alert.setTitle("Error message");
alert.setHeaderText(null);
alert.setContentText("Please choose book first");
alert.showAndWait();
}else{

prepare = connect.prepareStatement(sql);
prepare.setString(1, String.valueOf(customerId));
prepare.setString(2, purchase_info_bookID.getText());
prepare.setString(3, purchase_info_bookTItle.getText());
prepare.setString(4, purchase_info_author.getText());
prepare.setString(5, purchase_info_genre.getText());
prepare.setString(6, String.valueOf(qty));

String checkData = "SELECT title, price FROM book WHERE
title = '"
+purchase_bookTitle.getSelectionModel().getSelectedItem()+"'
";

double priceD = 0;

statement = connect.createStatement();
result = statement.executeQuery(checkData);

if(result.next()){
priceD = result.getDouble("price");
```

```
}

totalP = (qty * priceD);

prepare.setString(7, String.valueOf(totalP));

Date date = new Date();
java.sql.Date sqlDate = new java.sql.Date(date.getTime());

prepare.setString(8, String.valueOf(sqlDate));

prepare.executeUpdate();

purchaseDisplayTotal();
purchaseShowCustomerListData();
}
}catch(Exception e){e.printStackTrace();}
}

public void purchasePay(){

String sql = "INSERT INTO customer_info (customer_id, total,
date) "
+ "VALUES(?,?,?)";

connect = database.connectDb();

try{
Alert alert;
if(displayTotal == 0){
alert = new Alert(AlertType.ERROR);
alert.setTitle("Error message");
alert.setHeaderText(null);
alert.setContentText("Invalid :3");
alert.showAndWait();
}else{
alert = new Alert(AlertType.CONFIRMATION);
alert.setTitle("Confirmation message");
alert.setHeaderText(null);
alert.setContentText("Are you sure?");
Optional<ButtonType> option = alert.showAndWait();

if(option.get().equals(ButtonType.OK)){
prepare = connect.prepareStatement(sql);
prepare.setString(1, String.valueOf(customerId));
prepare.setString(2, String.valueOf(displayTotal));

Date date = new Date();
java.sql.Date sqlDate = new java.sql.Date(date.getTime());

prepare.setString(3, String.valueOf(sqlDate));

prepare.executeUpdate();

alert = new Alert(AlertType.INFORMATION);
alert.setTitle("Pesan Informasi");
alert.setHeaderText(null);
alert.setContentText("Berhasil!");
alert.showAndWait();
}
}
}catch(Exception e){e.printStackTrace();}
```

```
}

private double displayTotal;
public void purchaseDisplayTotal(){
purchasecustomerId();

String sql = "SELECT SUM(price) FROM customer WHERE
customer_id = '"+customerId+"'";

connect = database.connectDb();

try{
prepare = connect.prepareStatement(sql);
result = prepare.executeQuery();

if(result.next()){
displayTotal = result.getDouble("SUM(price)");
}

purchase_total.setText("$" + String.valueOf(displayTotal));

}catch(Exception e){e.printStackTrace();}

}

public void purchaseBookId(){

String sql = "SELECT book_id FROM book";

connect = database.connectDb();

try{
prepare = connect.prepareStatement(sql);
result = prepare.executeQuery();

ObservableList listData =
FXCollections.observableArrayList();

while(result.next()){
listData.add(result.getString("book_id"));
}

purchase_bookID.setItems(listData);
purchaseBookTitle();
}catch(Exception e){e.printStackTrace();}

}

public void purchaseBookTitle(){

String sql = "SELECT book_id, title FROM book WHERE book_id
= '"
+purchase_bookID.getSelectionModel().getSelectedItem()+"'";

connect = database.connectDb();

try{
prepare = connect.prepareStatement(sql);
result = prepare.executeQuery();

ObservableList listData =
FXCollections.observableArrayList();
```

```
while(result.next()){
listData.add(result.getString("title"));
}

purchase_bookTitle.setItems(listData);

purchaseBookInfo();

}catch(Exception e){e.printStackTrace();}

}

public void purchaseBookInfo(){

String sql = "SELECT * FROM book WHERE title = '"
+purchase_bookTitle.getSelectionModel().getSelectedItem()+"'
";

connect = database.connectDb();

String bookId = "";
String title = "";
String author = "";
String genre = "";
String date = "";

try{
prepare = connect.prepareStatement(sql);
result = prepare.executeQuery();

if(result.next()){
bookId = result.getString("book_id");
title = result.getString("title");
author = result.getString("author");
genre = result.getString("genre");
date = result.getString("pub_date");
}

purchase_info_bookID.setText(bookId);
purchase_info_bookTItle.setText(title);
purchase_info_author.setText(author);
purchase_info_genre.setText(genre);
purchase_info_date.setText(date);

}catch(Exception e){e.printStackTrace();}

}

public ObservableList<DataCustomer> purchaseListData(){
purchasecustomerId();
String sql = "SELECT * FROM customer WHERE customer_id =
'"+customerId+"'";

ObservableList<DataCustomer> listData =
FXCollections.observableArrayList();

connect = database.connectDb();

try{
prepare  = connect.prepareStatement(sql);
result = prepare.executeQuery();

DataCustomer customerD;
```

```
while(result.next()){
customerD = new DataCustomer(result.getInt("customer_id")
, result.getInt("book_id")
, result.getString("title")
, result.getString("author")
, result.getString("genre")
, result.getInt("quantity")
, result.getDouble("price")
, result.getDate("date"));

listData.add(customerD);
}

}catch(Exception e){e.printStackTrace();}
return listData;
}

private ObservableList<DataCustomer> purchaseCustomerList;
public void purchaseShowCustomerListData(){
purchaseCustomerList = purchaseListData();

purchase_col_bookID.setCellValueFactory(new
PropertyValueFactory<>("bookId"));
purchase_col_bookTitle.setCellValueFactory(new
PropertyValueFactory<>("title"));
purchase_col_author.setCellValueFactory(new
PropertyValueFactory<>("author"));
purchase_col_genre.setCellValueFactory(new
PropertyValueFactory<>("genre"));
purchase_col_quantity.setCellValueFactory(new
PropertyValueFactory<>("quantity"));
purchase_col_price.setCellValueFactory(new
PropertyValueFactory<>("price"));

purchase_tableView.setItems(purchaseCustomerList);

}

private SpinnerValueFactory<Integer> spinner;

public void purchaseDisplayQTY(){
spinner = new
SpinnerValueFactory.IntegerSpinnerValueFactory(0, 10, 0);
purchase_quantity.setValueFactory(spinner);
}
private int qty;
public void purhcaseQty(){
qty = purchase_quantity.getValue();
}

private int customerId;
public void purchasecustomerId(){

String sql = "SELECT MAX(customer_id) FROM customer";
int checkCID = 0 ;
connect = database.connectDb();

try{
prepare = connect.prepareStatement(sql);
result = prepare.executeQuery();

if(result.next()){
```

```
customerId = result.getInt("MAX(customer_id)");
}

String checkData = "SELECT MAX(customer_id) FROM
customer_info";

prepare = connect.prepareStatement(checkData);
result = prepare.executeQuery();

if(result.next()){
checkCID = result.getInt("MAX(customer_id)");
}

if(customerId == 0){
customerId += 1;
}else if(checkCID == customerId){
customerId = checkCID + 1;
}

}catch(Exception e){e.printStackTrace();}

}

public void displayUsername(){
String user = getData.username;
user = user.substring(0, 1).toUpperCase() +
user.substring(1);
username.setText(user);
}

public void switchForm(ActionEvent event){

if(event.getSource() == dashboard_btn){
dashboard_form.setVisible(true);
availableBooks_form.setVisible(false);
purchase_form.setVisible(false);

dashboard_btn.setStyle("-fx-background-color:linear-
gradient(to top right, #12415b, #121807);");
availableBooks_btn.setStyle("-fx-background-color:
transparent");
purchase_btn.setStyle("-fx-background-color: transparent");

dashboardAB();
dashboardTI();
dashboardTC();
dashboardIncomeChart();
dashboardCustomerChart();

}else if(event.getSource() == availableBooks_btn){
dashboard_form.setVisible(false);
availableBooks_form.setVisible(true);
purchase_form.setVisible(false);

availableBooks_btn.setStyle("-fx-background-color:linear-
gradient(to top right, #12415b, #121807);");
dashboard_btn.setStyle("-fx-background-color: transparent");
purchase_btn.setStyle("-fx-background-color: transparent");

availableBooksShowListData();
availableBooksSeach();

}else if(event.getSource() == purchase_btn){
```

```
dashboard_form.setVisible(false);
availableBooks_form.setVisible(false);
purchase_form.setVisible(true);

purchase_btn.setStyle("-fx-background-color:linear-
gradient(to top right, #12415b, #121807);");
availableBooks_btn.setStyle("-fx-background-color:
transparent");
dashboard_btn.setStyle("-fx-background-color: transparent");

purchaseBookTitle();
purchaseBookId();
purchaseShowCustomerListData();
purchaseDisplayQTY();
purchaseDisplayTotal();

}
}

private double x = 0;
private double y = 0;
public void logout(){
try{
Alert alert = new Alert(AlertType.CONFIRMATION);
alert.setTitle("Confirmation Message");
alert.setHeaderText(null);
alert.setContentText("Are you sure you want to logout?");
Optional<ButtonType> option = alert.showAndWait();

if(option.get().equals(ButtonType.OK)){

// HIDE YOUR DASHBOARD
logout.getScene().getWindow().hide();
// LINK YOUR LOGIN FORM
Parent root =
FXMLLoader.load(getClass().getResource("FXMLDocument.fxml"))
;
Stage stage = new Stage();
Scene scene = new Scene(root);

root.setOnMousePressed((MouseEvent event) ->{
x = event.getSceneX();
y = event.getSceneY();
});

root.setOnMouseDragged((MouseEvent event) ->{
stage.setX(event.getScreenX() - x);
stage.setY(event.getScreenY() - y);

stage.setOpacity(.8);
});

root.setOnMouseReleased((MouseEvent event) ->{
stage.setOpacity(1);
});

stage.initStyle(StageStyle.TRANSPARENT);

stage.setScene(scene);
stage.show();
}

}catch(Exception e){e.printStackTrace();}
```

```
}

public void close(){
System.exit(0);
}

public void minimize(){
Stage stage = (Stage)main_form.getScene().getWindow();
stage.setIconified(true);
}

@Override
public void initialize(URL location, ResourceBundle
resources) {
displayUsername();

dashboardAB();
dashboardTI();
dashboardTC();
dashboardIncomeChart();
dashboardCustomerChart();

// TO SHOW THE DATA ON TABLEVIEW (AVAILABLE BOOKS)
availableBooksShowListData();

purchaseBookId();
purchaseBookTitle();
purchaseShowCustomerListData();
purchaseDisplayQTY();
purchaseDisplayTotal();

}

}
```

g. dashboardDesign.css

```
.top-form{
    -fx-background-color: #fff;
    -fx-border-color: #000;
    -fx-border-width: .4px .4px .2px .4px;
}
.semi-top-form{
    -fx-background-color: #efefef;
    -fx-border-color: #000;
    -fx-border-width: .2px .4px .4px .4px;
}
.close{
    -fx-background-color: transparent;
    -fx-cursor:hand;
}
.close:hover{
    -fx-background-color:#b10c0c;
}
.minimize{
    -fx-background-color: transparent;
    -fx-cursor:hand;
}
.minimize:hover{
    -fx-background-color: #ddd;
}
.nav-form{
    -fx-background-color:linear-gradient(to top
right, #20dbd8, #12374e);
}
.nav-btn{
    -fx-background-color:transparent;
    -fx-cursor:hand;
    -fx-font-size: 14px;
    -fx-text-fill: #fff;
    -fx-font-family: Arial;
    -fx-alignment: CENTER-LEFT;
}
.sign-out{
    -fx-background-color: #146786;
    -fx-cursor:hand;
    -fx-background-radius: 20px;
}
.sign-out:hover{
    -fx-background-color: #121522;
}
.shadow{
    -fx-effect: dropshadow(three-pass-
box,rgba(0,0,0,0.5), 8,0,0,0);
}
.white-bg{
    -fx-background-color: #fff;
    -fx-background-radius: 8px;
}
.card{
    -fx-background-color: linear-gradient(to top,
#146786, #146786);
    -fx-background-radius: 4px;
}
.add-btn{
    -fx-background-color: #145a85;
    -fx-background-radius: 4px;
    -fx-cursor:hand;
    -fx-font-size: 14px;
    -fx-font-family: Arial;
    -fx-text-fill: #fff;
}
.add-btn:hover{
    -fx-background-color: #005B41;
}
.update-btn{
    -fx-background-color: #1c2642;
    -fx-background-radius: 4px;
    -fx-cursor:hand;
    -fx-font-size: 14px;
    -fx-font-family: Arial;
    -fx-text-fill: #fff;
}
.update-btn:hover{
    -fx-background-color: #810CA8;
}
.clear-btn{
    -fx-background-color: #12374e;
    -fx-background-radius: 4px;
    -fx-cursor:hand;
    -fx-font-size: 14px;
    -fx-font-family: Arial;
    -fx-text-fill: #fff;
}
.clear-btn:hover{
    -fx-background-color: #8B9A46;
}
.delete-btn{
    -fx-background-color: #121522;
    -fx-background-radius: 4px;
    -fx-cursor:hand;
    -fx-font-size: 14px;
    -fx-font-family: Arial;
    -fx-text-fill: #fff;
}
.delete-btn:hover{
    -fx-background-color: CD1818;
}
.textfield{
    -fx-background-color: linear-gradient(to
bottom, #efefef, #eee);
    -fx-background-radius: 2px;
    -fx-font-family: Tahoma;
    -fx-border-color:#000;
    -fx-border-radius: 2px;
    -fx-border-width: .4px;
}
.textfield:focused{
    -fx-border-color:linear-gradient(to top right,
#3c2c21, #93773e);
    -fx-background-color: #fff;
    -fx-border-width: 1px;
}
.search{
    -fx-background-color: transparent;
    -fx-font-size: 13px;
    -fx-font-family: Arial;
    -fx-border-color: linear-gradient(to top
right, #121522, #146786);
    -fx-border-radius: 4px;
    -fx-border-width: .8px;
    -fx-padding: 0px 0px 0px 28px;
}
.search:focused{
    -fx-border-width: 1.5px;
}
.table-view{
    -fx-background-color:transparent;
    -fx-border-color: linear-gradient(to top
right, #121522, #146786);
    -fx-border-radius: 8px;
    -fx-border-width: 2px;
    -fx-padding: 0px;
}
.table-view .table-column{
    -fx-alignment: CENTER;
}
.table-view .column-header-background{
    -fx-background-color: linear-gradient(to top
right, #121522, #146786);
    -fx-background-radius: 8px 8px 0px 0px;
    -fx-background-insets: 0 0 0 0;
}
.table-view .column-header, .filter{
    -fx-background-color: transparent;
    -fx-size: 40px;
}
.table-view .column-header .label{
```

```
    -fx-text-fill: #fff;
    -fx-font-family: Arial;
}
.info{
    -fx-background-color:linear-gradient(to top 
right, #12374e, #12374e);
    -fx-background-radius: 0 8px 8px 0;
}
.info-label{
    -fx-background-color: #fff;
    -fx-background-radius: 4px;
    -fx-padding: 0 0 0 5px;
}
```

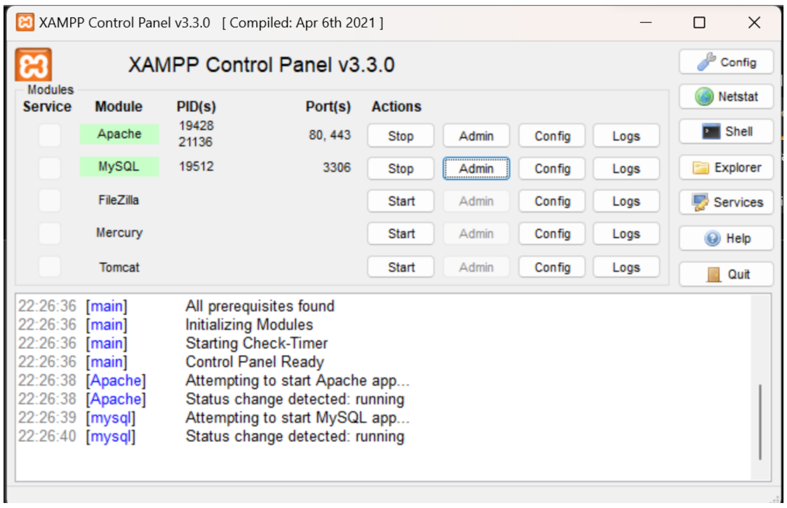
h. getData,java

```
package bookshopmanagementsystem;
public class getData {
    public static String username;
    public static String path;
}
```

i. database.java

```
package bookshopmanagementsystem;
import java.sql.Connection;
import java.sql.DriverManager;

public class database {

    public static Connection connectDb(){

        try{

Class.forName("com.mysql.jdbc.Driver");
            Connection connect = 
DriverManager.getConnection("jdbc:mysql://localh
ost/book", "root", ""); // address, database 
username, database password
            return connect;
        }catch(Exception 
e){e.printStackTrace();}
        return null; // LETS MAKE OUR DATABASE 
: ) book is our database name : )
    }

}
```

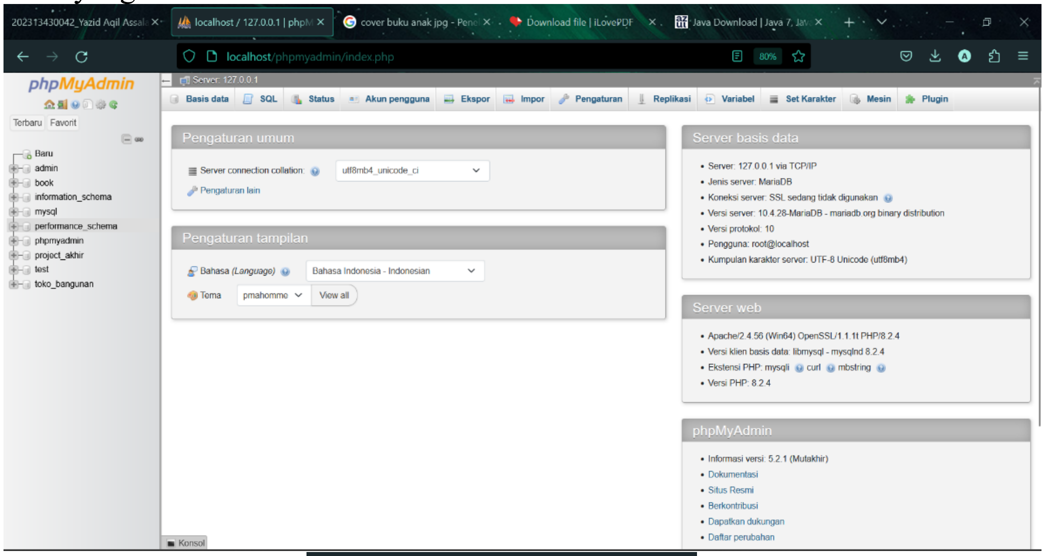
14. Aktifkan XAMPP


XAMPP Control Panel v3.3.0 [Compiled: Apr 6th 2021]
XAMPP Control Panel v3.3.0
Config
Netstat
Shell
Explorer
Services
Help
Quit

Modules Service	Module	PID(s)	Port(s)	Actions			
	Apache	19428 21136	80, 443	Stop	Admin	Config	Logs
	MySQL	19512	3306	Stop	Admin	Config	Logs
	FileZilla			Start	Admin	Config	Logs
	Mercury			Start	Admin	Config	Logs
	Tomcat			Start	Admin	Config	Logs

```
22:26:36  [main]     All prerequisites found
22:26:36  [main]     Initializing Modules
22:26:36  [main]     Starting Check-Timer
22:26:36  [main]     Control Panel Ready
22:26:38  [Apache]   Attempting to start Apache app...
22:26:38  [Apache]   Status change detected: running
22:26:39  [mysql]    Attempting to start MySQL app...
22:26:40  [mysql]    Status change detected: running
```

15. Masuk ke phpMyAdmin lalu buatlah database “book”, dan buatlah beberapa tabel yang dibutuhkan



Baru
admin
book
information_schema
mysql
performance_schema
phpmyadmin
project_akhir
test
toko_bangunan

Pengaturan umum

Server connection collation: utf8mb4_unicode_ci

Pengaturan lain

Pengaturan tampilan

Bahasa (*Language*): Bahasa Indonesia - Indonesian

Tema: pmahomme View all

Server basis data

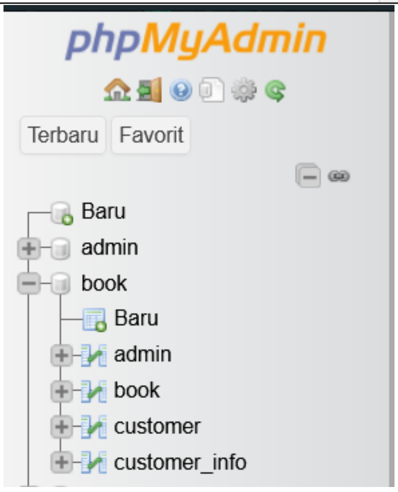
- Server: 127.0.0.1 via TCP/IP
- Jenis server: MariaDB
- Koneksi server: SSL sedang tidak digunakan
- Versi server: 10.4.28-MariaDB - mariadb.org binary distribution
- Versi protokol: 10
- Pengguna: root@localhost
- Kumpulan karakter server: UTF-8 Unicode (utf8mb4)

Server web

- Apache/2.4.56 (Win64) OpenSSL/1.1.1t PHP/8.2.4
- Versi klien basis data: libmysql - mysqlnd 8.2.4
- Ekstensi PHP: mysqli curl mbstring
- Versi PHP: 8.2.4

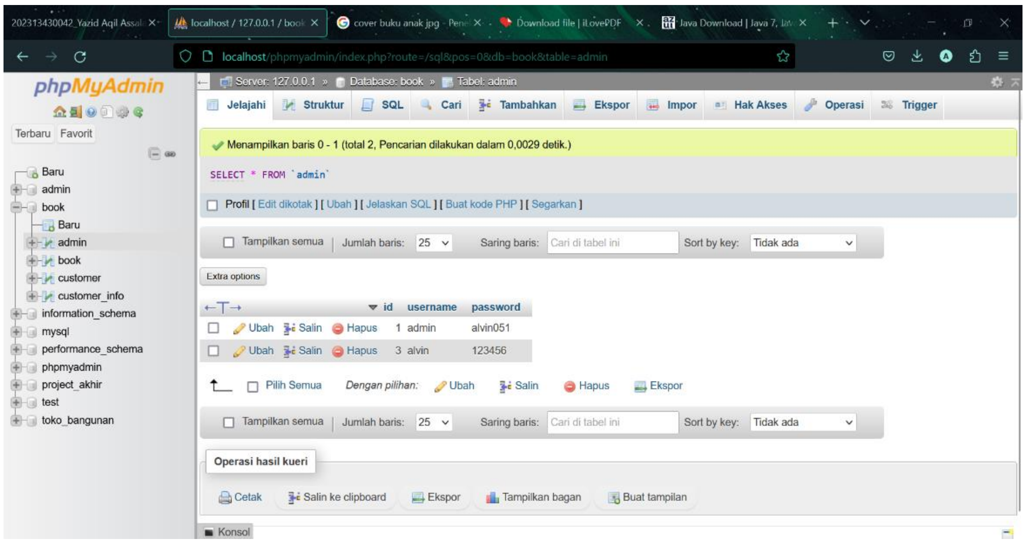
phpMyAdmin

- Informasi versi: 5.2.1 (Mutakhir)
- Dokumentasi
- Situs Resmi
- Berkontribusi
- Dapatkan dukungan
- Daftar perubahan



Baru
admin
book
Baru
admin
book
customer
customer_info

a. Tabel "admin"



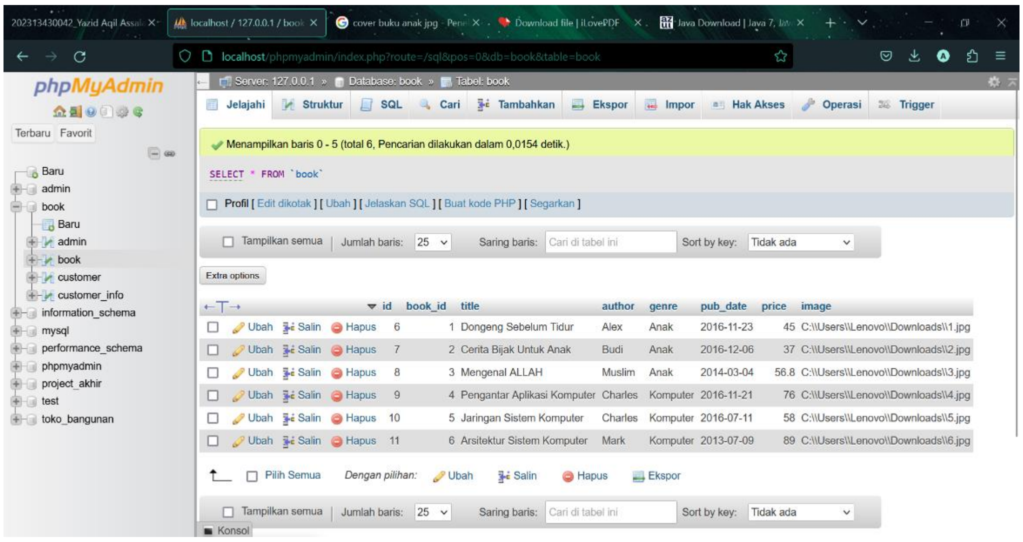
Server: 127.0.0.1 » Database: book » Tabel: admin

Menampilkan baris 0 - 1 (total 2, Pencarian dilakukan dalam 0,0029 detik.)

```
SELECT * FROM `admin`
```

id	username	password
1	admin	alvin051
3	alvin	123456

b. Tabel "book"



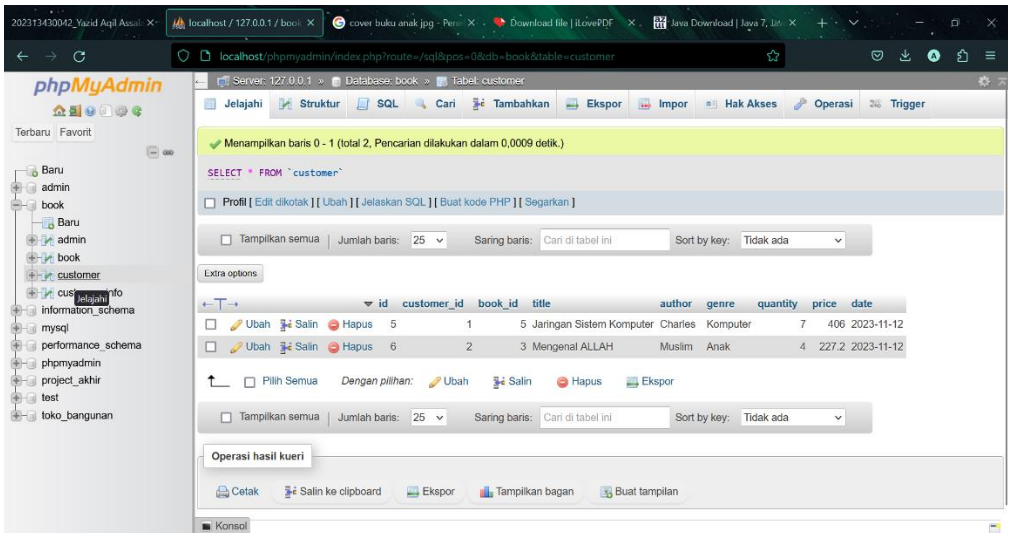
Server: 127.0.0.1 » Database: book » Tabel: book

Menampilkan baris 0 - 5 (total 6, Pencarian dilakukan dalam 0,0154 detik.)

```
SELECT * FROM `book`
```

id	book_id	title	author	genre	pub_date	price	image
6	1	Dongeng Sebelum Tidur	Alex	Anak	2016-11-23	45	C:\\Users\\Lenovo\\Downloads\\1.jpg
7	2	Cerita Bijak Untuk Anak	Budi	Anak	2016-12-06	37	C:\\Users\\Lenovo\\Downloads\\2.jpg
8	3	Mengenal ALLAH	Muslim	Anak	2014-03-04	56.8	C:\\Users\\Lenovo\\Downloads\\3.jpg
9	4	Pengantar Aplikasi Komputer	Charles	Komputer	2016-11-21	76	C:\\Users\\Lenovo\\Downloads\\4.jpg
10	5	Jaringan Sistem Komputer	Charles	Komputer	2016-07-11	58	C:\\Users\\Lenovo\\Downloads\\5.jpg
11	6	Arsitektur Sistem Komputer	Mark	Komputer	2013-07-09	89	C:\\Users\\Lenovo\\Downloads\\6.jpg

c. Tabel "customer"



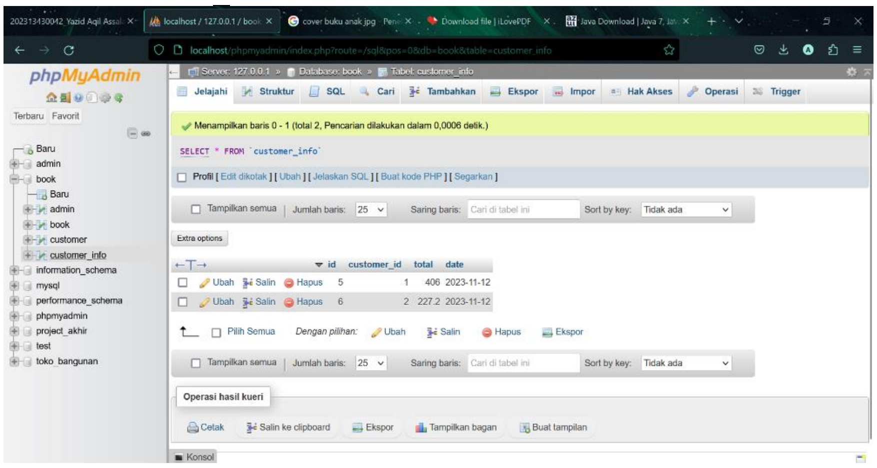
Server: 127.0.0.1 » Database: book » Tabel: customer

Menampilkan baris 0 - 1 (total 2, Pencarian dilakukan dalam 0,0009 detik.)

```
SELECT * FROM `customer`
```

id	customer_id	book_id	title	author	genre	quantity	price	date
5	1	5	Jaringan Sistem Komputer	Charles	Komputer	7	406	2023-11-12
6	2	3	Mengenal ALLAH	Muslim	Anak	4	227.2	2023-11-12

d. Tabel “customer_info”



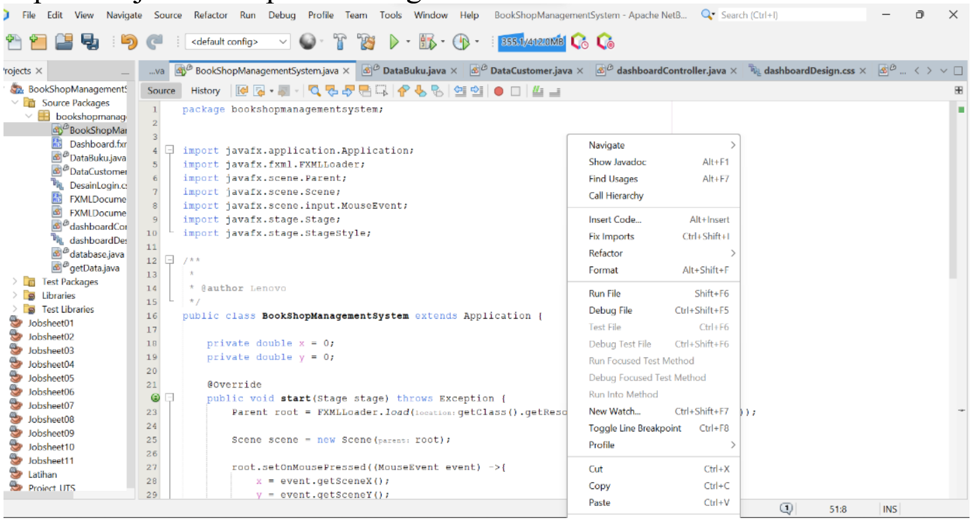
Menampilkan baris 0 - 1 (total 2, Pencarian dilakukan dalam 0,0006 detik.)

```
SELECT * FROM `customer_info`
```

id	customer_id	total	date
5	1	406	2023-11-12
6	2	227.2	2023-11-12

localhost/phpmyadmin/index.php?route=/sql&pos=0&db=book&table=customer_info
Server: 127.0.0.1 » Database: book » Tabel: customer_info
Jelajahi
Struktur
SQL
Cari
Tambahkan
Ekspor
Impor
Hak Akses
Operasi
Trigger
Profil [Edit dikotak] [Ubah] [Jelaskan SQL] [Buat kode PHP] [Segarkan]
Tampilkan semua | Jumlah baris: 25 | Saring baris: Cari di tabel ini | Sort by key: Tidak ada
Extra options
Pilih Semua | Dengan pilihan: Ubah Salin Hapus Ekspor
Operasi hasil kueri
Cetak
Salin ke clipboard
Ekspor
Tampilkan bagan
Buat tampilan
Konsol

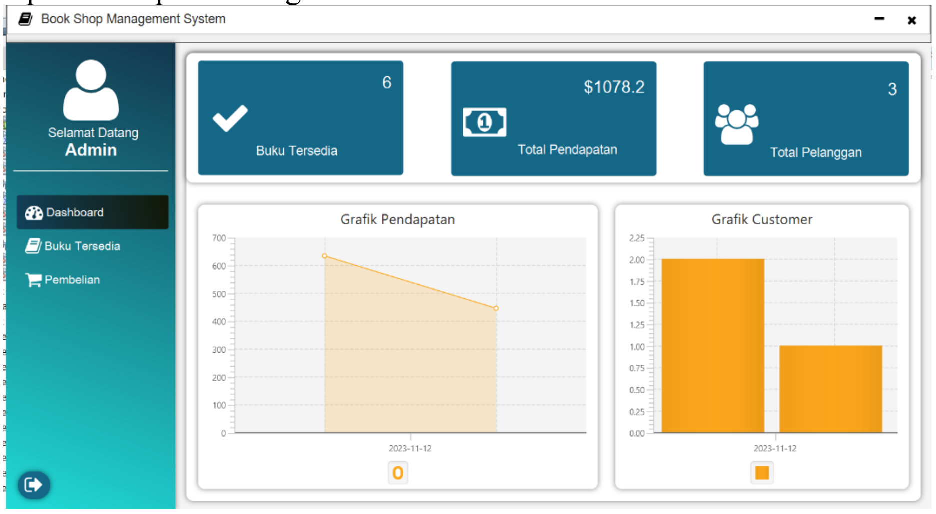
16. Setelah semua kode dan database yang dibutuhkan selesai dibuat maka kita dapat menjalankan aplikasi dengan lancar dan baik



```java
package bookshopmanagementsystem;


import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.input.MouseEvent;
import javafx.stage.Stage;
import javafx.stage.StageStyle;

/**
 *
 * @author Lenovo
 */
public class BookShopManagementSystem extends Application {

    private double x = 0;
    private double y = 0;

    @Override
    public void start(Stage stage) throws Exception {
        Parent root = FXMLLoader.load(location: getClass().getReso
        
        Scene scene = new Scene(parent: root);

        root.setOnMousePressed((MouseEvent event) ->{
            x = event.getSceneX();
            y = event.getSceneY();
```

Klik kanan pada file BookShopManagementSystem.java lalu klik “Run File” atau dapat juga dengan klik segitiga hijau yang ada pada bagian atas halaman

17. Aplikasi siap untuk digunakan



Book Shop Management System
Selamat Datang
Admin
Dashboard
Buku Tersedia
Pembelian
6
Buku Tersedia
$1078.2
Total Pendapatan
3
Total Pelanggan
Grafik Pendapatan
2023-11-12
Grafik Customer
2023-11-12