SALDO CERDAS

FINAL REPORT

Business Application Development

ANGGOTA

01 Tiffany Angelia - 2602050734

02 Yosua Lieandi - 2602065074

03 Irvin Lolo - 2602109662

04 Brigitta Marcella - 2602137325

05 Naswa Winagati Rahhadi Rukma - 2602195060

06 Rubennito Sugata - 2602071543

07 Estevan Tannaka - 2602091204

08 Ramadhan Rizky - 2602181465

09 Nandito Adrian - 2602167025

10 Mayzha Silva Andhini - 2602172542

11 Kyla Aleitta - 2602099661

12 Muhammad Bayu - 2602080415

Saldo Cerdas

TABLE OF CONTENT

CHAPTER I

PRELIMINARY

December 2023

BACKGROUND

OF SALDO CERDAS

Di masa sekarang ini banyak sekali orang yang sering lupa dengan keuangannya sendiri, baik itu pengeluaran maupun pemasukan juga dengan Utang-piutang mereka. Oleh karena itu, kami membuat Saldo Cerdas.

Saldo Cerdas merupakan sebuah aplikasi pencatat keuangan individu, baik itu pemasukan, pengeluaran, dan utang-piutang. Aplikasi ini juga dapat memberikan analisis keuangan User dalam periode tertentu.

OBJECTIVE

OF SALDO CERDAS

Pencatatan Keuangan yang Akurat

Memungkinkan user untuk dengan mudah mencatat setiap transaksi keuangan mereka

Analisis Keuangan

Memberikan visualisasi grafis yang jelas dan mudah dimengerti untuk membantu pengguna memahami pola pengeluaran dan pemasukan mereka.

ROLES & RESPONSIBILITY

Team Leader
Tiffany Angelia

System Designer & UI Designer
Brigitta Marcella
Nandito Adrian
Kyla Aleitta

Application Developer / Programmer
Yosua Lieandi
Irvin Lolo

Tester
Naswa Winagati
Mayzha Silva
Ramadhan Rizky

Presenter
Estevan Tannaka
Muhammad Bayu
Rubennito Sugata

TIMELINE

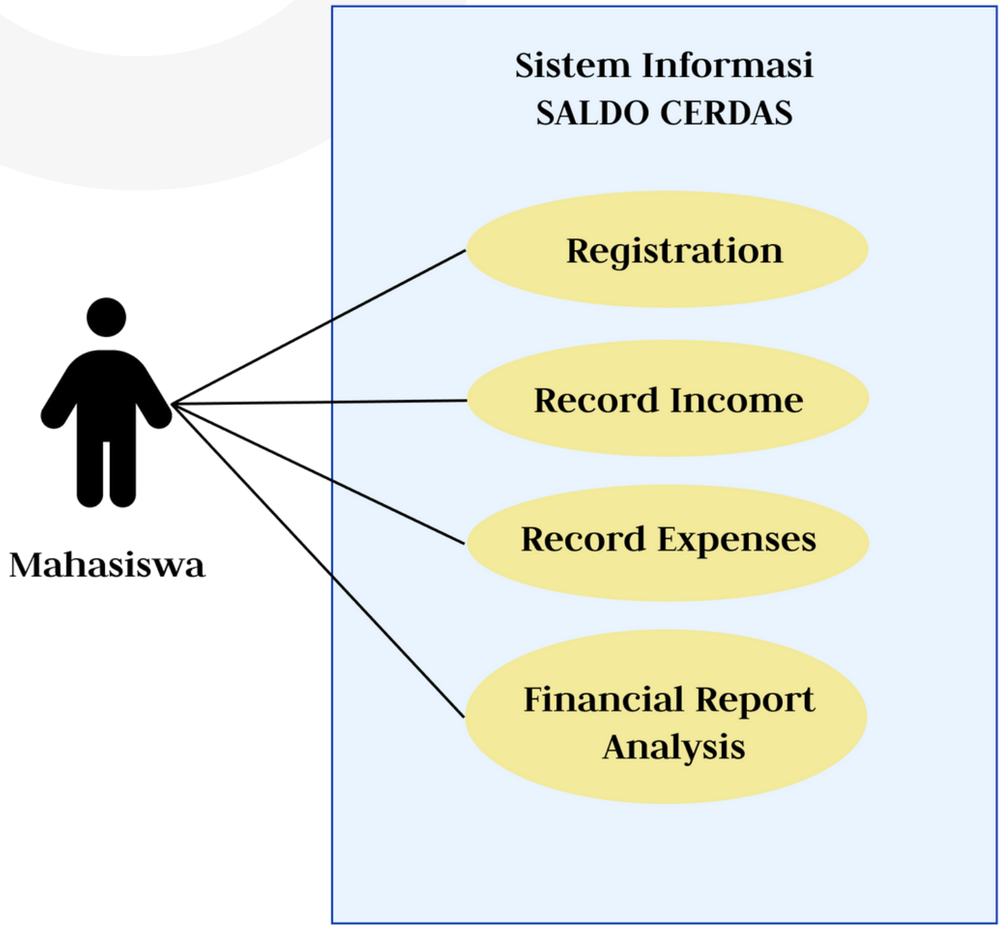
PIC	Task	Plan		Action		Status
		Start	End	Start	End	
Semua Anggota	Menentukan ide aplikasi	18-Sep		18-Sep		Tepat Waktu
Semua Anggota	Menentukan use case aplikasi	18-Sep		18-Sep		Tepat Waktu
Semua Anggota	Membuat use case diagram	18-Sep		18-Sep		Tepat Waktu
Semua Anggota	Pembagian tugas pembuatan ERD dan Mock Up	26-Sep		26-Sep		Tepat Waktu
Semua Anggota	Penentuan deadline tugas	26-Sep		26-Sep		Tepat Waktu
Tiffany	Mock Up (regist page)	27-Sep-23		27-Sep-23		Tepat Waktu
Yosua	Mock Up (analysis, income - expense page)	27-Sep-23		27-Sep-23		Tepat Waktu
Brigitta	Mock Up (analysis, income - expense page)	27-Sep-23		27-Sep-23		Tepat Waktu
Mayzha	Mock Up (profile page)	27-Sep-23		27-Sep-23		Tepat Waktu
Kyla	Mock Up (form utang, peminjaman page)	27-Sep-23		27-Sep-23		Tepat Waktu
Naswa	Mock Up (home page)	27-Sep-23		27-Sep-23		Tepat Waktu
Dito, Estevan, Irvin, Rama, Ruben, Bayu	ERD	28-Sep		29-Sep		Telat
Irvin, Yosua, Tiffany	Pengerjaan Coding Sprint 1	2 Oktober	5 Oktober	14 Oktober		Telat
Irvin, Yosua, Tiffany	Pengerjaan Coding Sprint 2	9 Oktober	12 Oktober	15 Oktober		Telat
Rama, Naswa, Mayzha, Tiffany	Testing Sprint 1	6 Oktober	8 Oktober	17 Oktober	20 Oktober	Telat
Rama, Naswa, Mayzha, Tiffany	Testing Sprint 2	13 Oktober	15 Oktober	17 Oktober	20 Oktober	Telat
Yosua, Irvin	Pengerjaan Coding Sprint 3	16 Oktober	19 Oktober	19-Nov	26-Nov	Telat
Rama, Naswa, Mayzha	Testing sprint 3	20 Oktober	22 Oktober	1 Desember	2 Desember	Telat
Yosua, Irvin	Pengerjaan Coding Sprint 4	23 Oktober	26 Oktober	27-Nov	03-Dec	Telat
Rama, Naswa, Mayzha	Testing sprint 4	27 Oktober	29 Oktober	3 Desember		Telat

CHAPTER II

BUSINESS APPLICATION

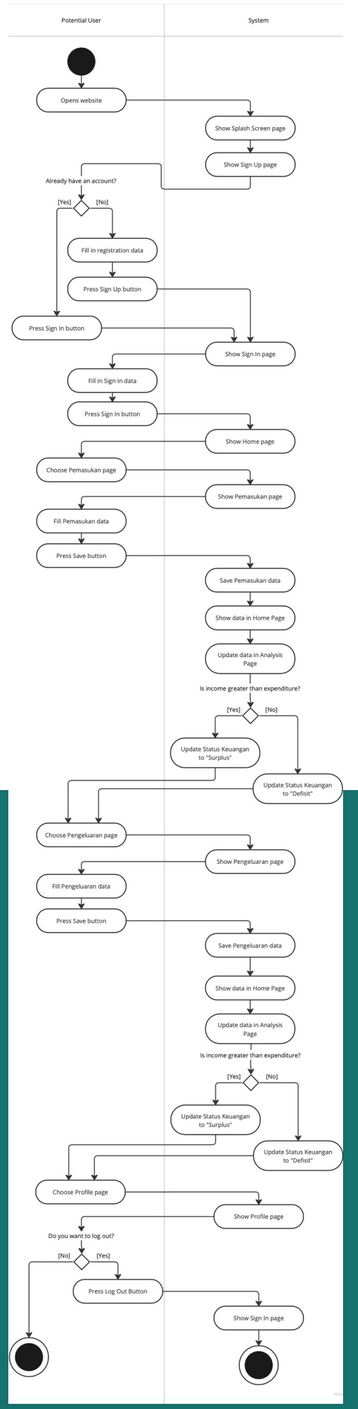
December 2023

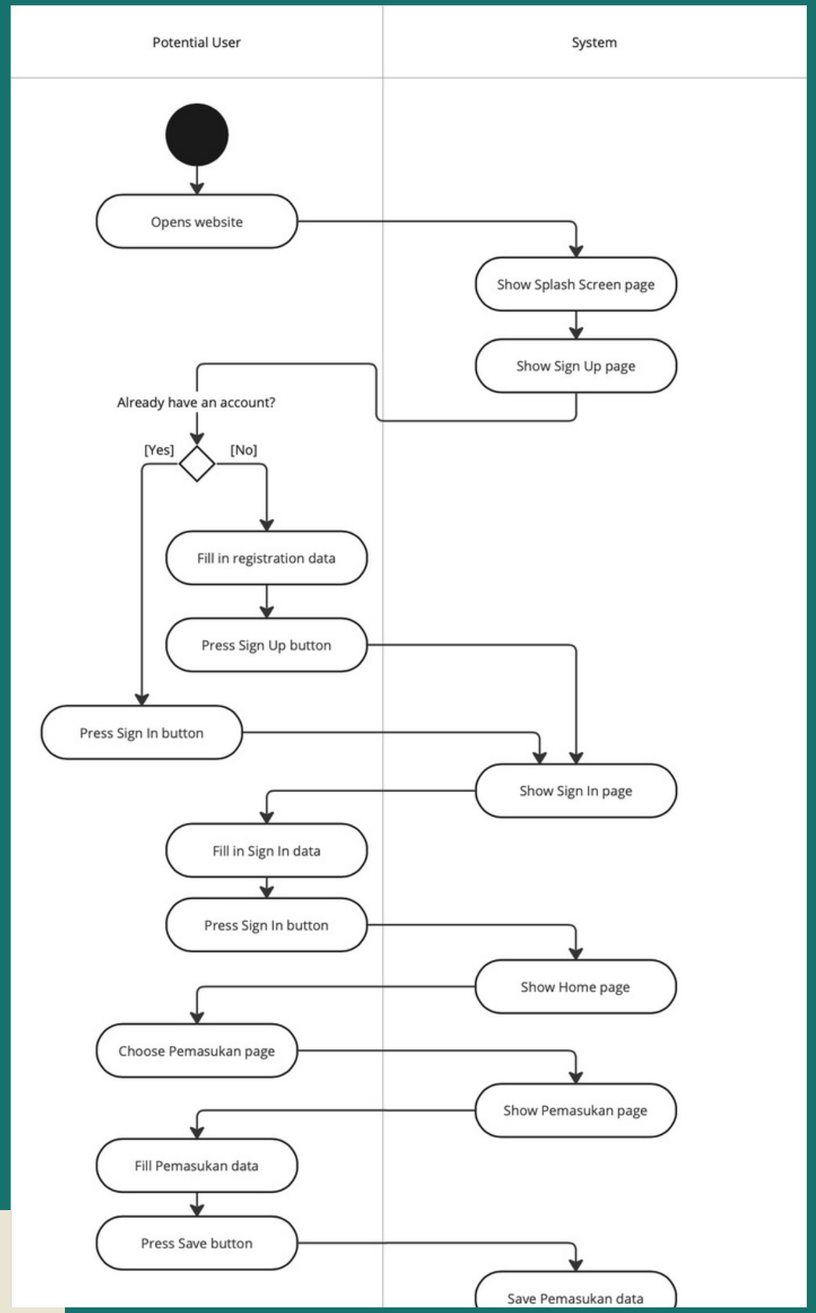
USE CASE SALDO CERDAS

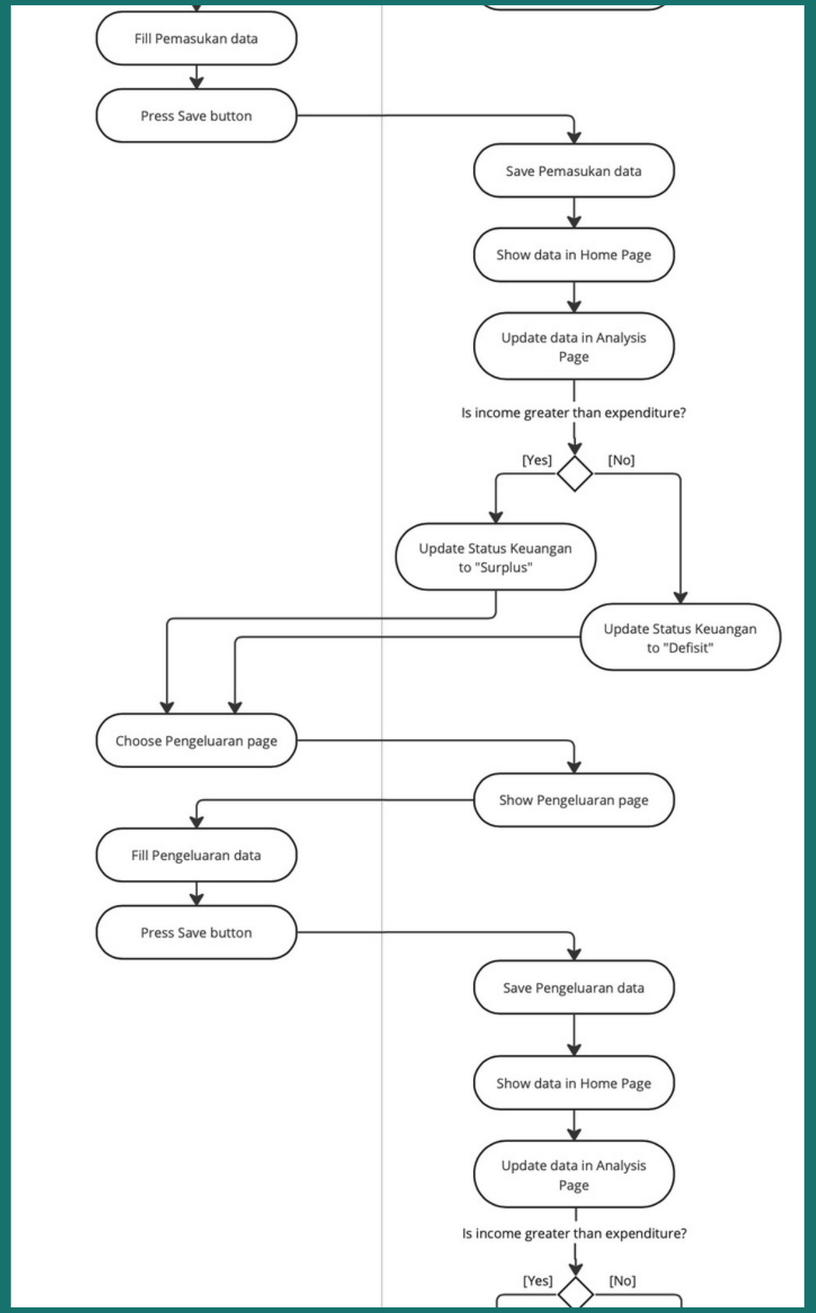


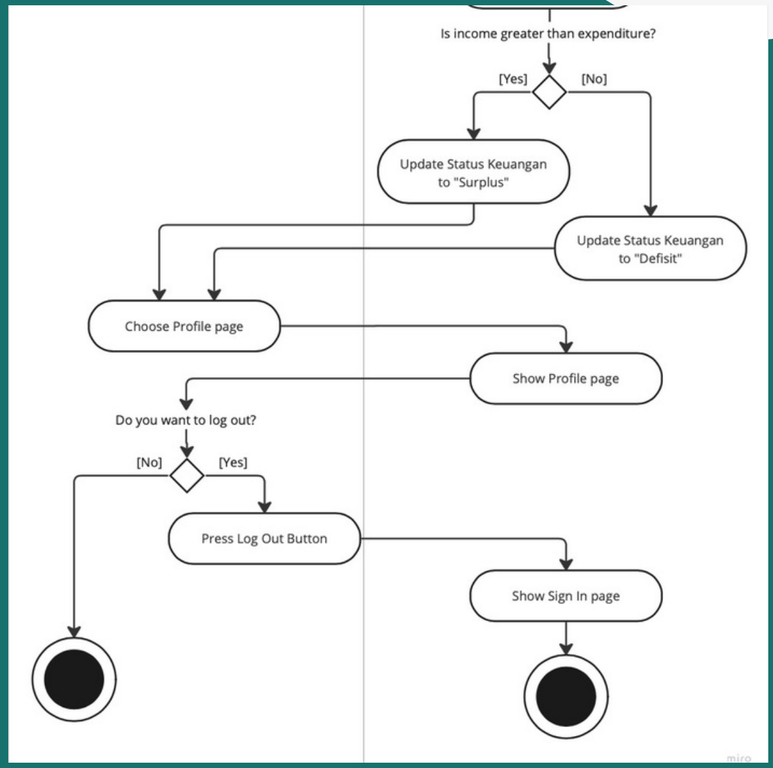
ACTIVITY DIAGRAM SALDO CERDAS

https://drive.google.com/file/d/10-d38dTtERVzHmFHHQtNmeO3H869XPz1/view?usp=share_link




Potential User
System
Opens website
Show Splash Screen page
Show Sign Up page
Already have an account?
[Yes]
[No]
Fill in registration data
Press Sign Up button
Press Sign In button
Show Sign In page
Fill in Sign In data
Press Sign In button
Show Home page
Choose Pemasukan page
Show Pemasukan page
Fill Pemasukan data
Press Save button
Save Pemasukan data

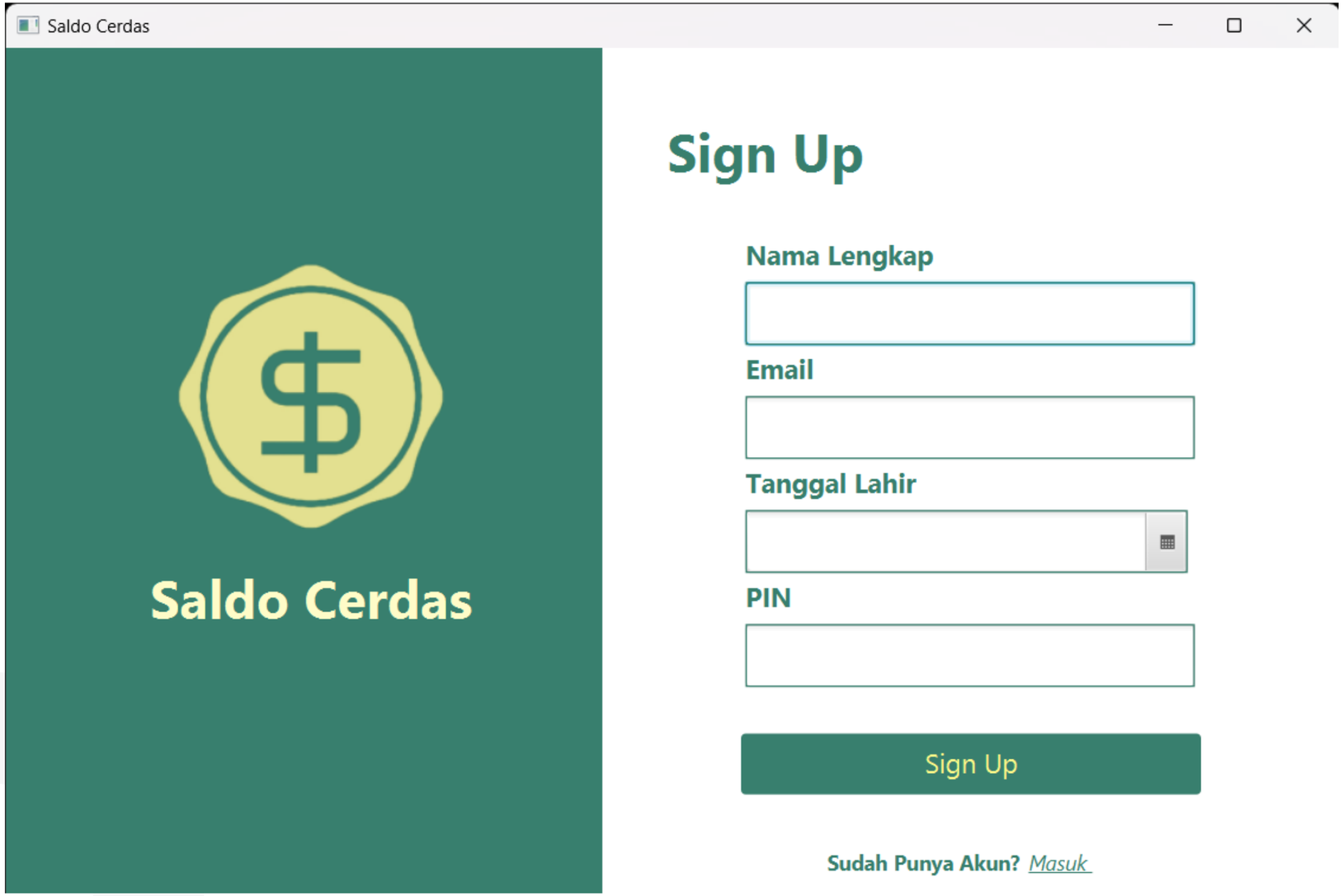

Fill Pemasukan data
Press Save button
Save Pemasukan data
Show data in Home Page
Update data in Analysis Page
Is income greater than expenditure?
[Yes]
[No]
Update Status Keuangan to "Surplus"
Update Status Keuangan to "Defisit"
Choose Pengeluaran page
Show Pengeluaran page
Fill Pengeluaran data
Press Save button
Save Pengeluaran data
Show data in Home Page
Update data in Analysis Page
Is income greater than expenditure?
[Yes]
[No]


Is income greater than expenditure?
[Yes]
[No]
Update Status Keuangan to "Surplus"
Update Status Keuangan to "Defisit"
Choose Profile page
Show Profile page
Do you want to log out?
[No]
[Yes]
Press Log Out Button
Show Sign In page
miro

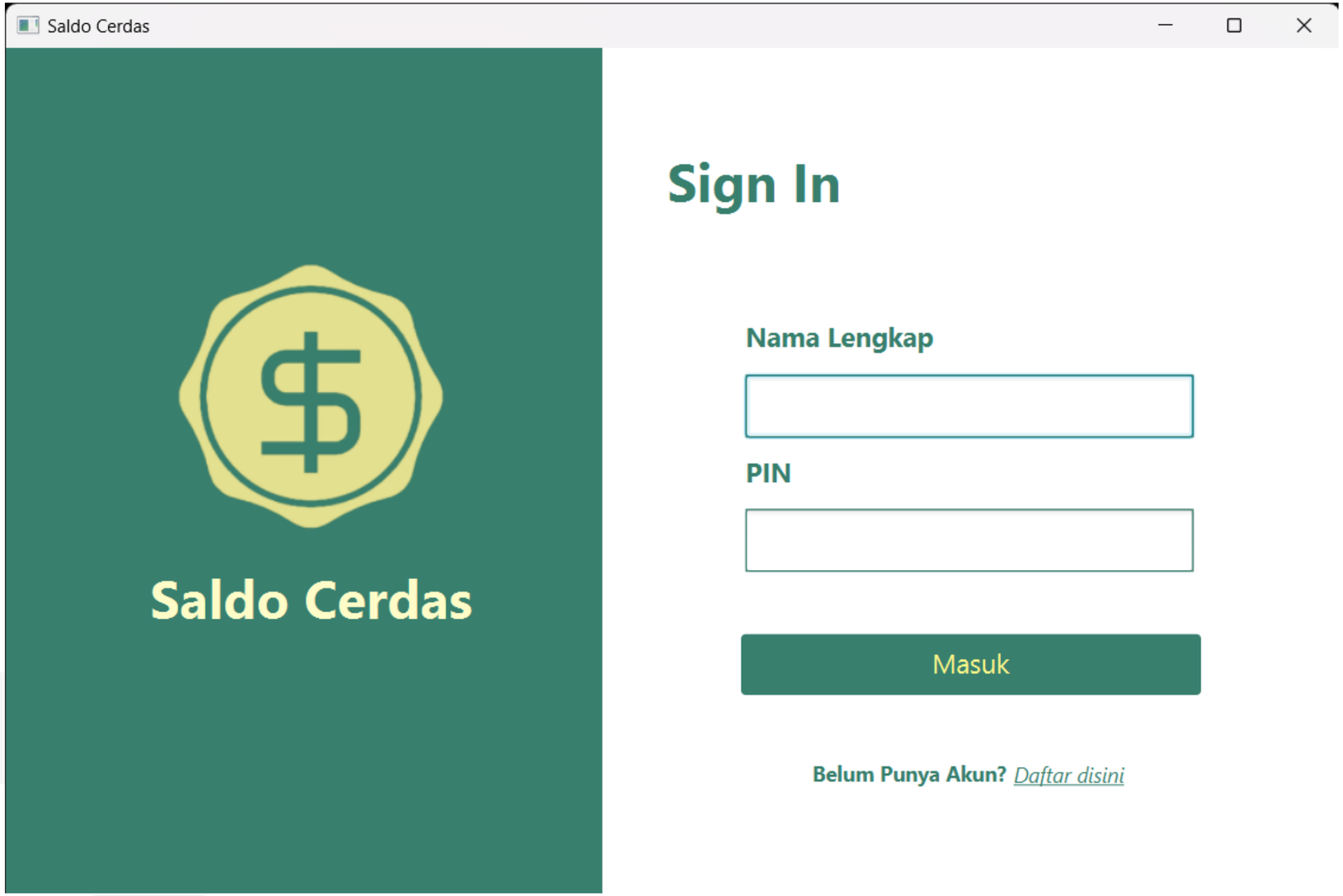
GUI DESIGN

SPLASH SCREEN

SIGN UP


Saldo Cerdas
Saldo Cerdas
Sign Up
Nama Lengkap
Email
Tanggal Lahir
PIN
Sign Up
Sudah Punya Akun? Masuk

SIGN IN


Saldo Cerdas
Saldo Cerdas
Sign In
Nama Lengkap
PIN
Masuk
Belum Punya Akun? Daftar disini

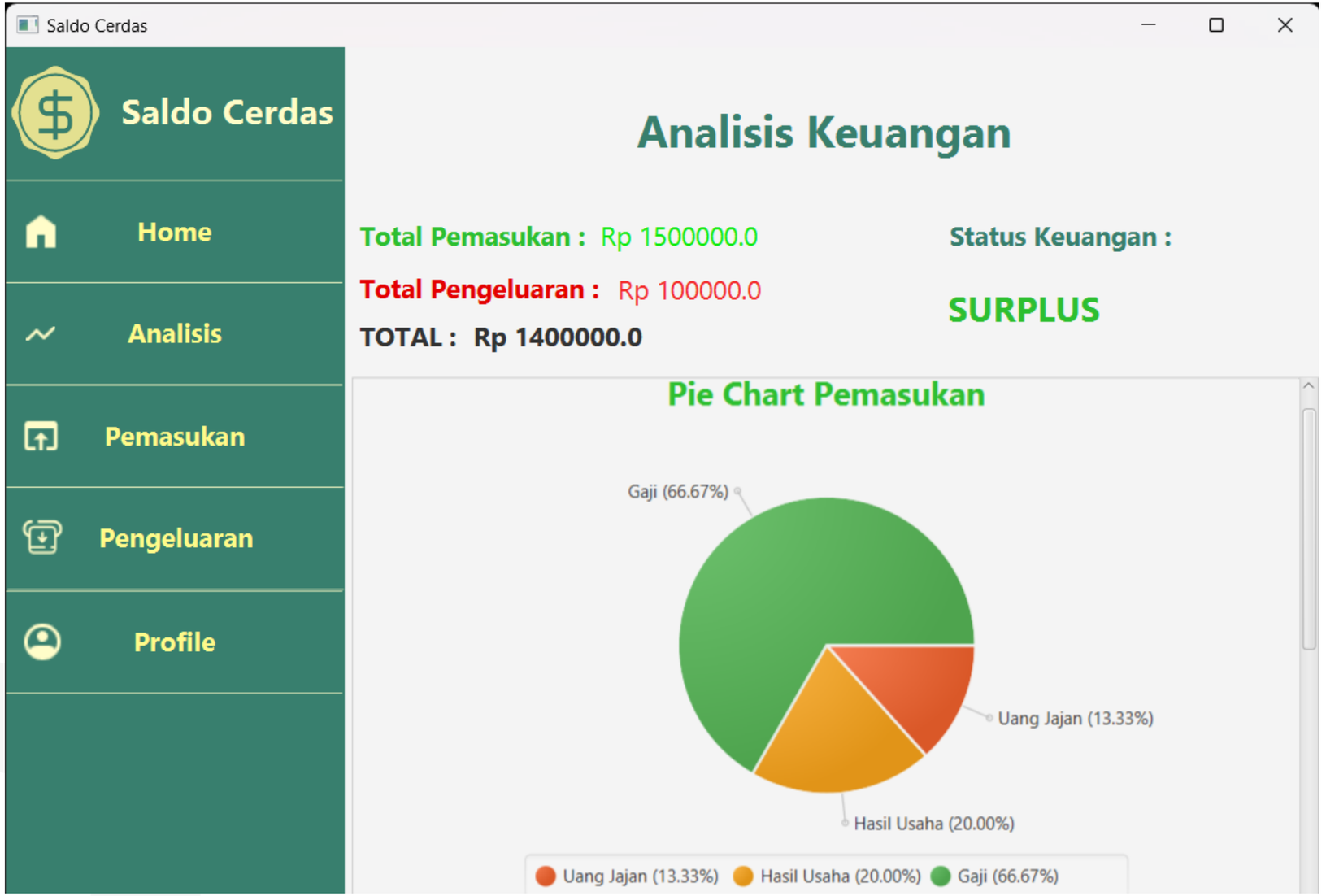
HOME


Saldo Cerdas
Saldo Cerdas
Home
Analisis
Pemasukan
Pengeluaran
Profile
Selamat Datang, Irvin
Pemasukan
Kategori
Jumlah
Catatan
Tanggal
Uang Jajan
200000
kiriman ortu
2023-12-20
Hasil Usaha
300000
Kerja
2023-12-13
Gaji
1000000
Gaji dari bos
2023-12-19
Add Pemasukan
Delete Pemasukan
Pengeluaran
Kategori
Jumlah
Catatan
Tanggal
Pakaian
30000
Pembelian Baju
2023-11-21

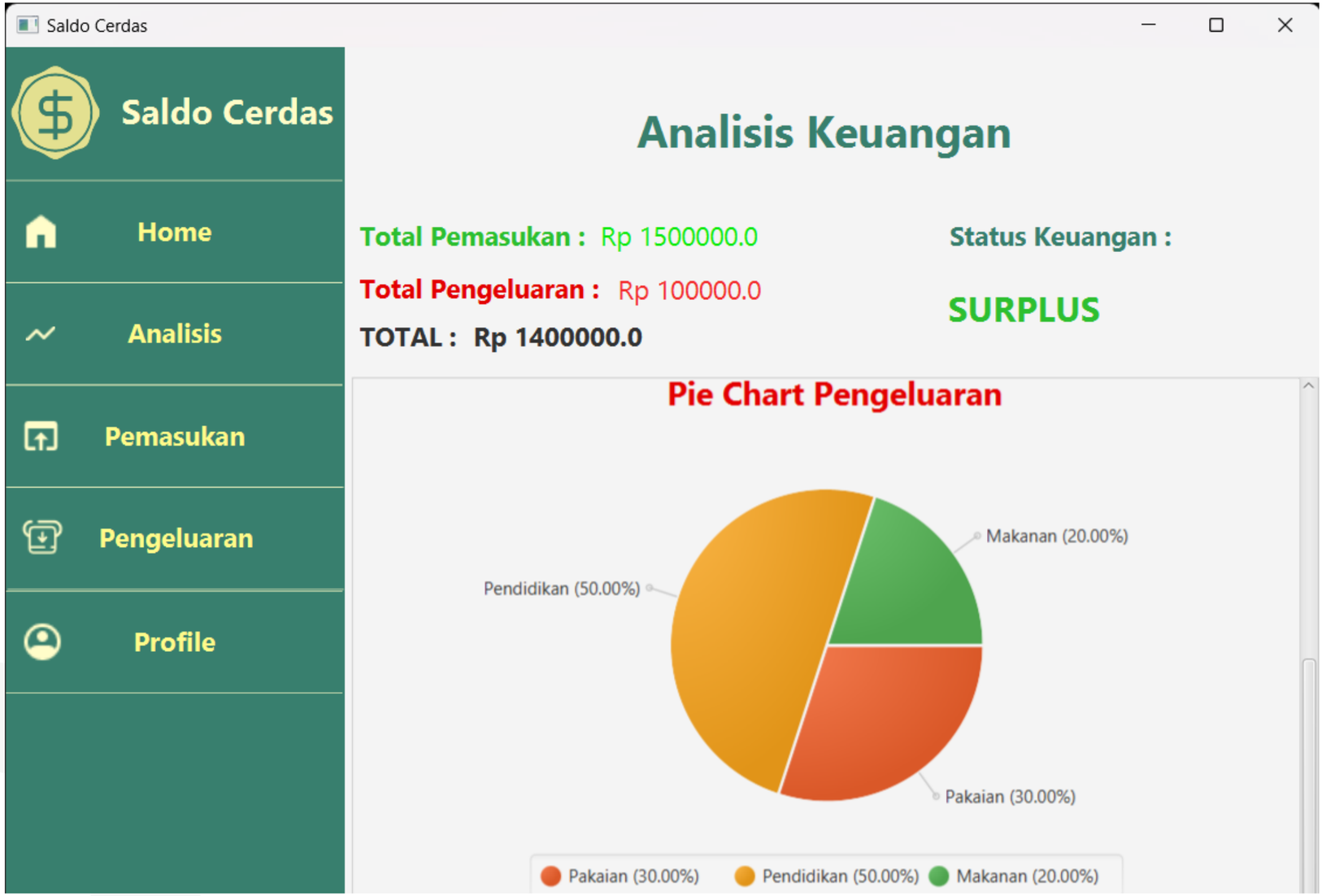
HOME


Saldo Cerdas
Saldo Cerdas
Home
Analisis
Pemasukan
Pengeluaran
Profile
Add Pemasukan
Delete Pemasukan
Pengeluaran
Kategori
Jumlah
Catatan
Tanggal
Pakaian
30000
Pembelian Baju
2023-11-21
Pendidikan
50000
Biaya Beli Buku
2023-12-01
Makanan
20000
Warteg
2023-12-18
Add Pengeluaran
Delete Pengeluaran

ANALISIS


Saldo Cerdas
Saldo Cerdas
Home
Analisis
Pemasukan
Pengeluaran
Profile
Analisis Keuangan
Total Pemasukan : Rp 1500000.0
Total Pengeluaran : Rp 100000.0
TOTAL : Rp 1400000.0
Status Keuangan :
SURPLUS
Pie Chart Pemasukan
Gaji (66.67%)
Uang Jajan (13.33%)
Hasil Usaha (20.00%)
Uang Jajan (13.33%)
Hasil Usaha (20.00%)
Gaji (66.67%)

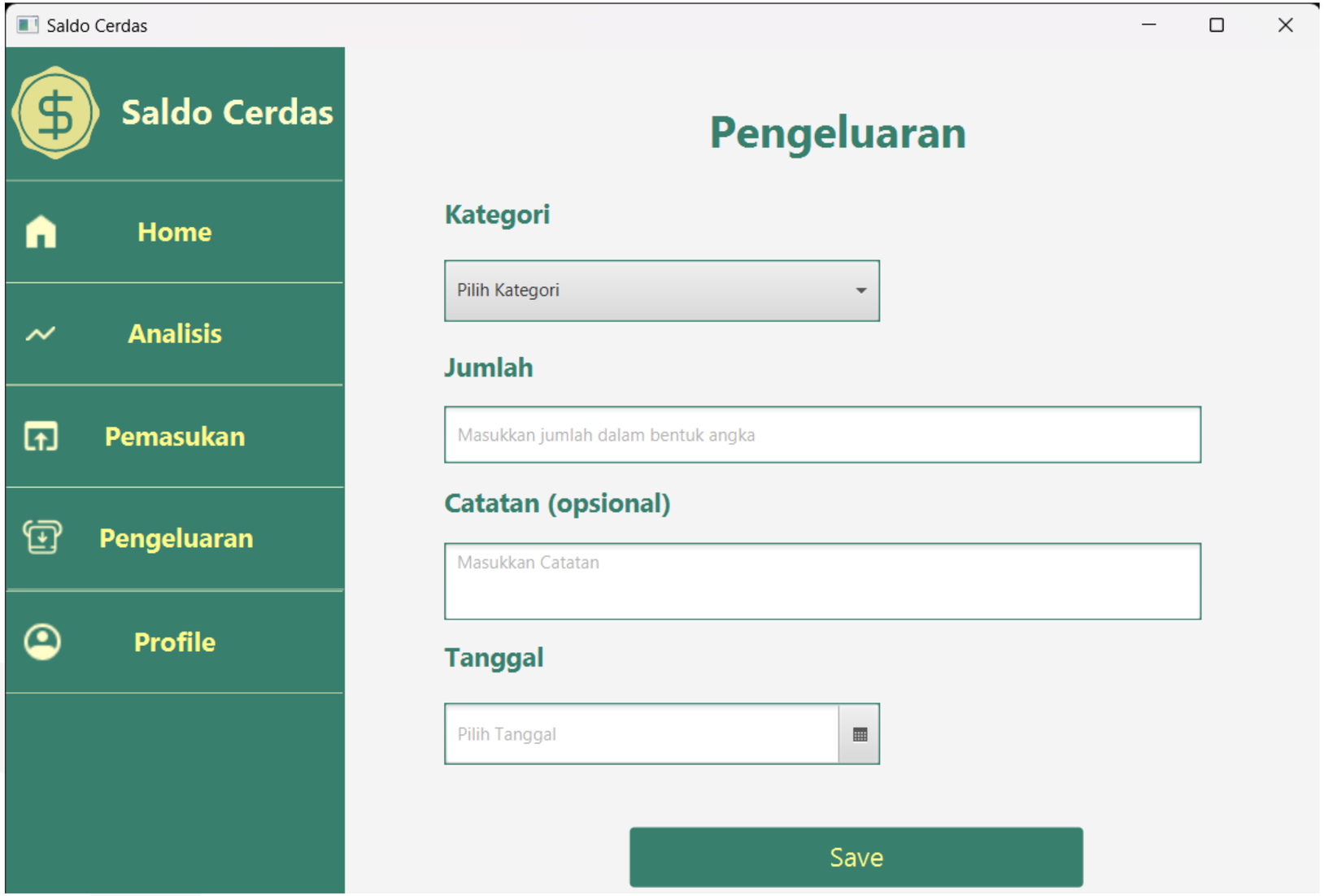
ANALISIS


Saldo Cerdas
Saldo Cerdas
Home
Analisis
Pemasukan
Pengeluaran
Profile
Analisis Keuangan
Total Pemasukan : Rp 1500000.0
Total Pengeluaran : Rp 100000.0
TOTAL : Rp 1400000.0
Status Keuangan :
SURPLUS
Pie Chart Pengeluaran
Makanan (20.00%)
Pendidikan (50.00%)
Pakaian (30.00%)
Pakaian (30.00%)
Pendidikan (50.00%)
Makanan (20.00%)

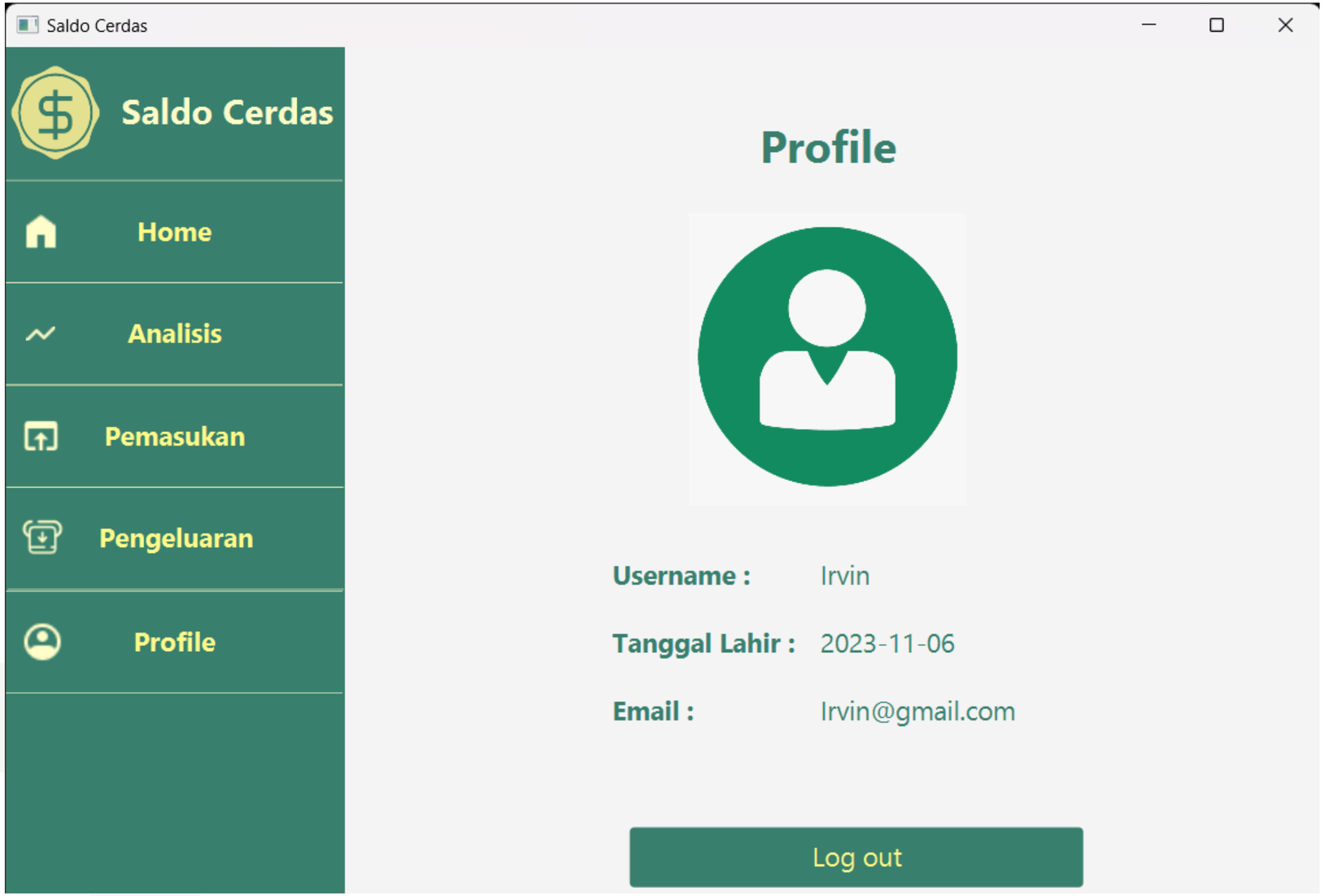
PEMASUKAN


Saldo Cerdas
Saldo Cerdas
Home
Analisis
Pemasukan
Pengeluaran
Profile
Pemasukan
Kategori
Pilih Kategori
Jumlah
Catatan (opsional)
Masukkan Catatan
Tanggal
Pilih Tanggal
Save

PENGELUARAN

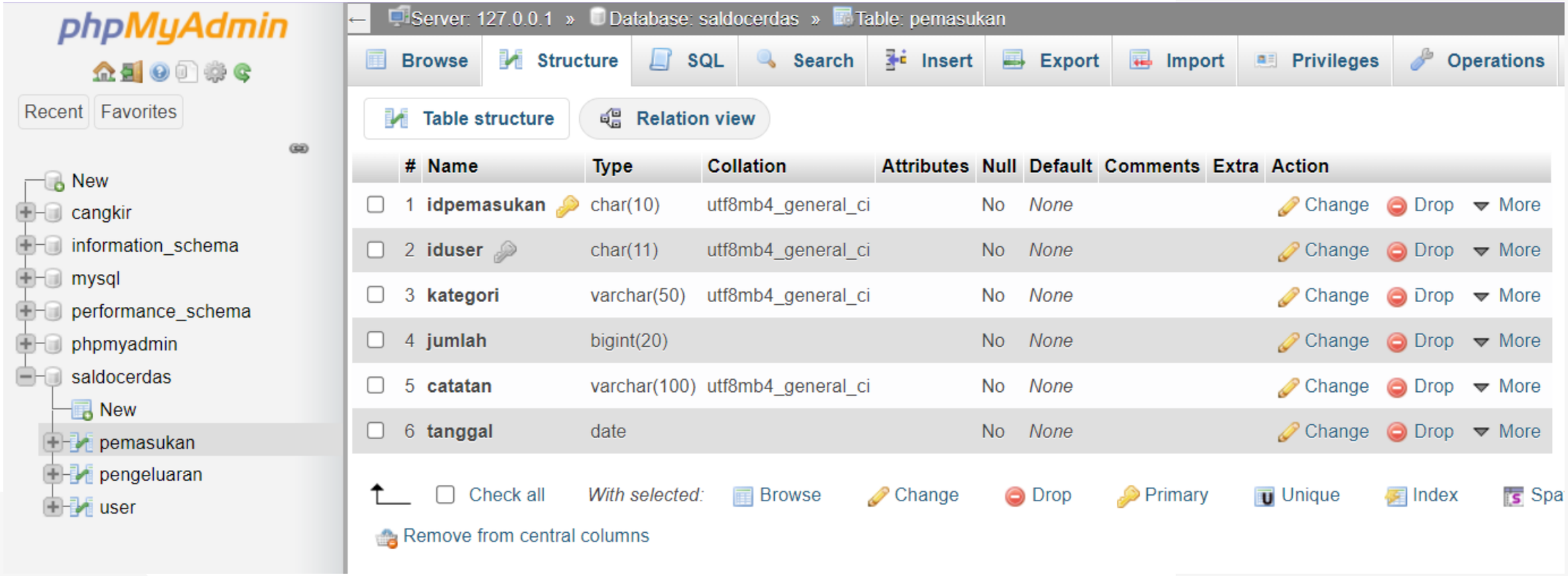


PROFILE

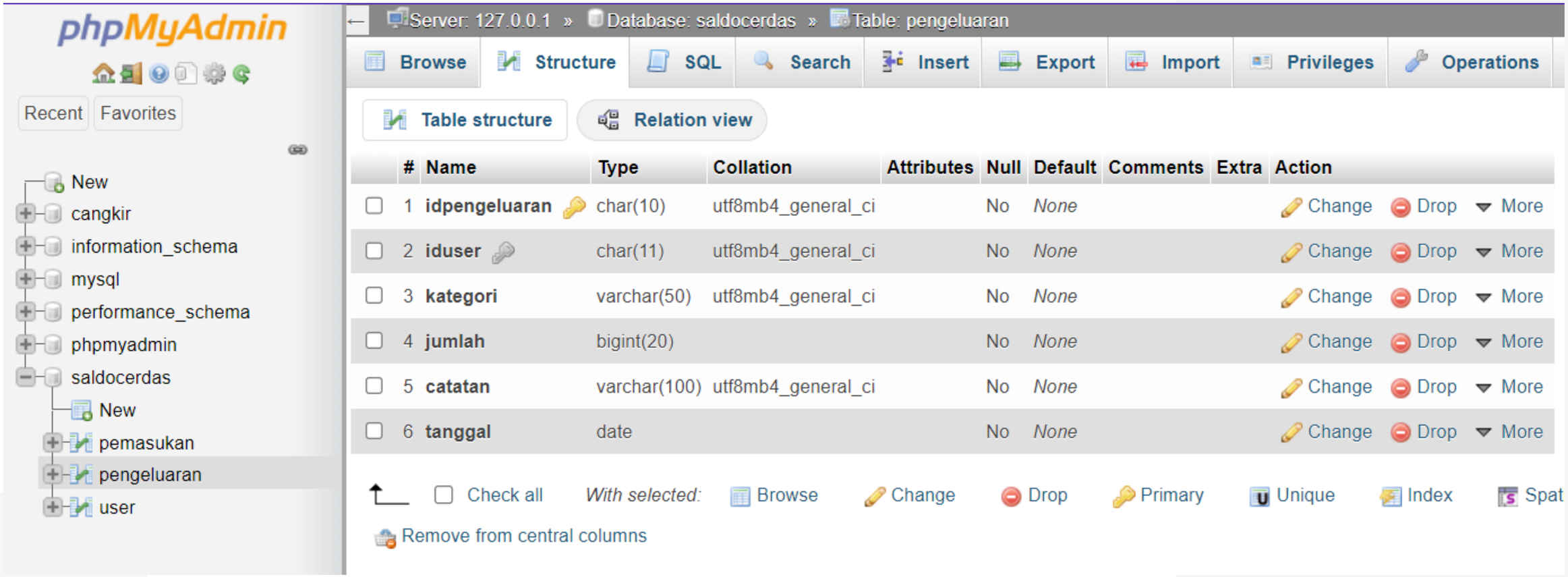

Saldo Cerdas
Saldo Cerdas
Home
Analisis
Pemasukan
Pengeluaran
Profile
Profile
Username : Irvin
Tanggal Lahir : 2023-11-06
Email : Irvin@gmail.com
Log out

DATABASE DESIGN

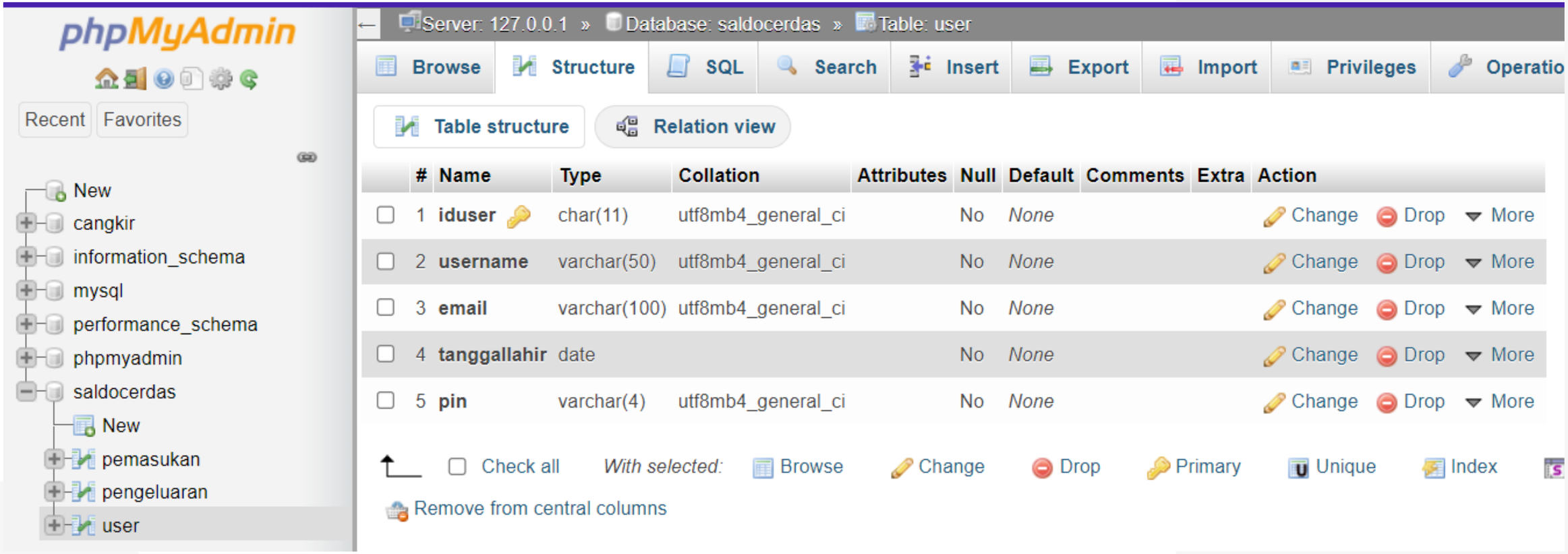
DB PEMASUKAN



DB PENGELUARAN

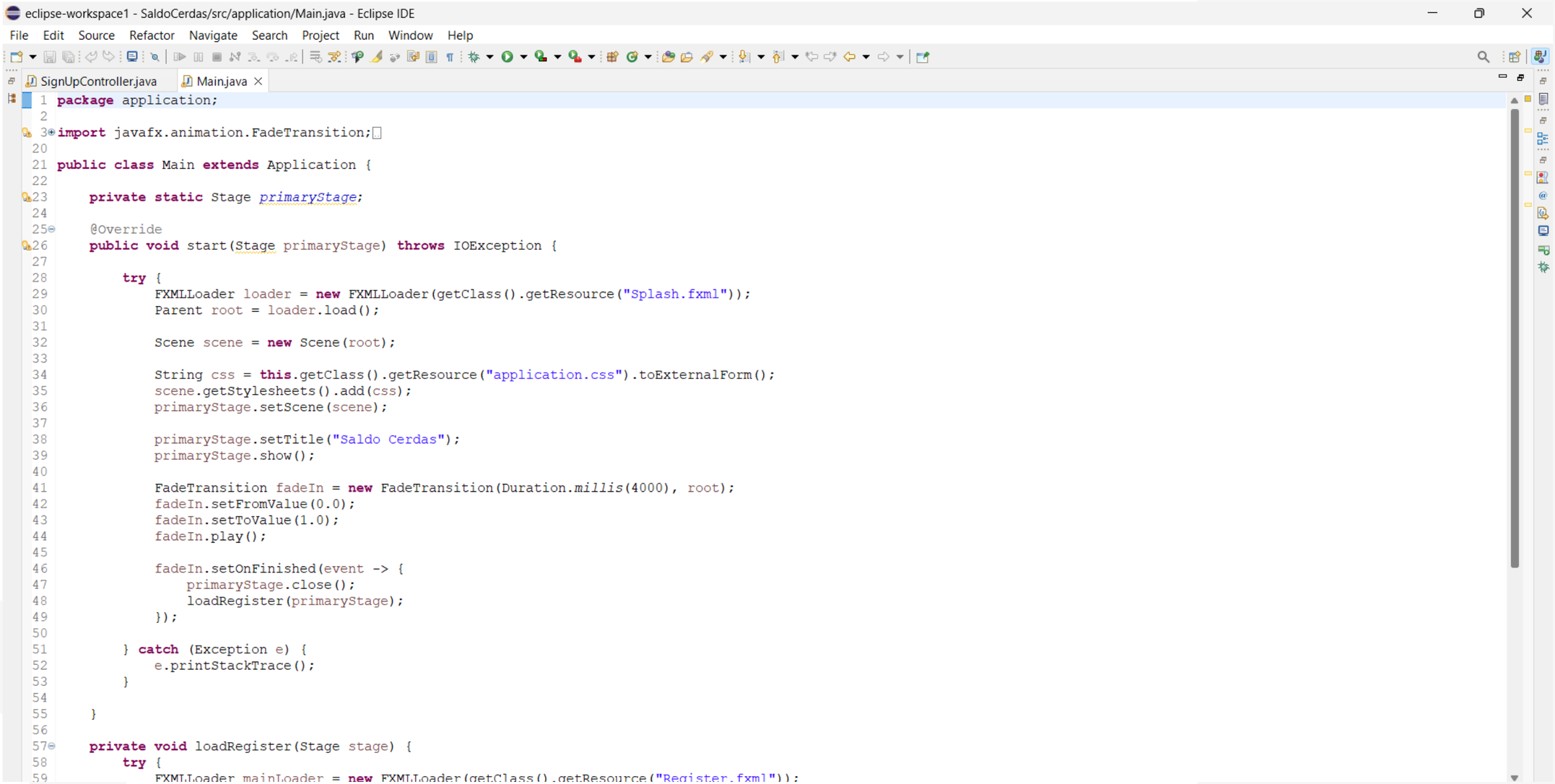


DB USER


phpMyAdmin
Recent
Favorites
New
cangkir
information_schema
mysql
performance_schema
phpmyadmin
saldocerdas
New
pemasukan
pengeluaran
user
Server: 127.0.0.1 » Database: saldocerdas » Table: user
Browse
Structure
SQL
Search
Insert
Export
Import
Privileges
Operatio
Table structure
Relation view
Name Type Collation Attributes Null Default Comments Extra Action
1 iduser char(11) utf8mb4_general_ci No None Change Drop More
2 username varchar(50) utf8mb4_general_ci No None Change Drop More
3 email varchar(100) utf8mb4_general_ci No None Change Drop More
4 tanggallahir date No None Change Drop More
5 pin varchar(4) utf8mb4_general_ci No None Change Drop More
Check all
With selected:
Browse
Change
Drop
Primary
Unique
Index
Remove from central columns

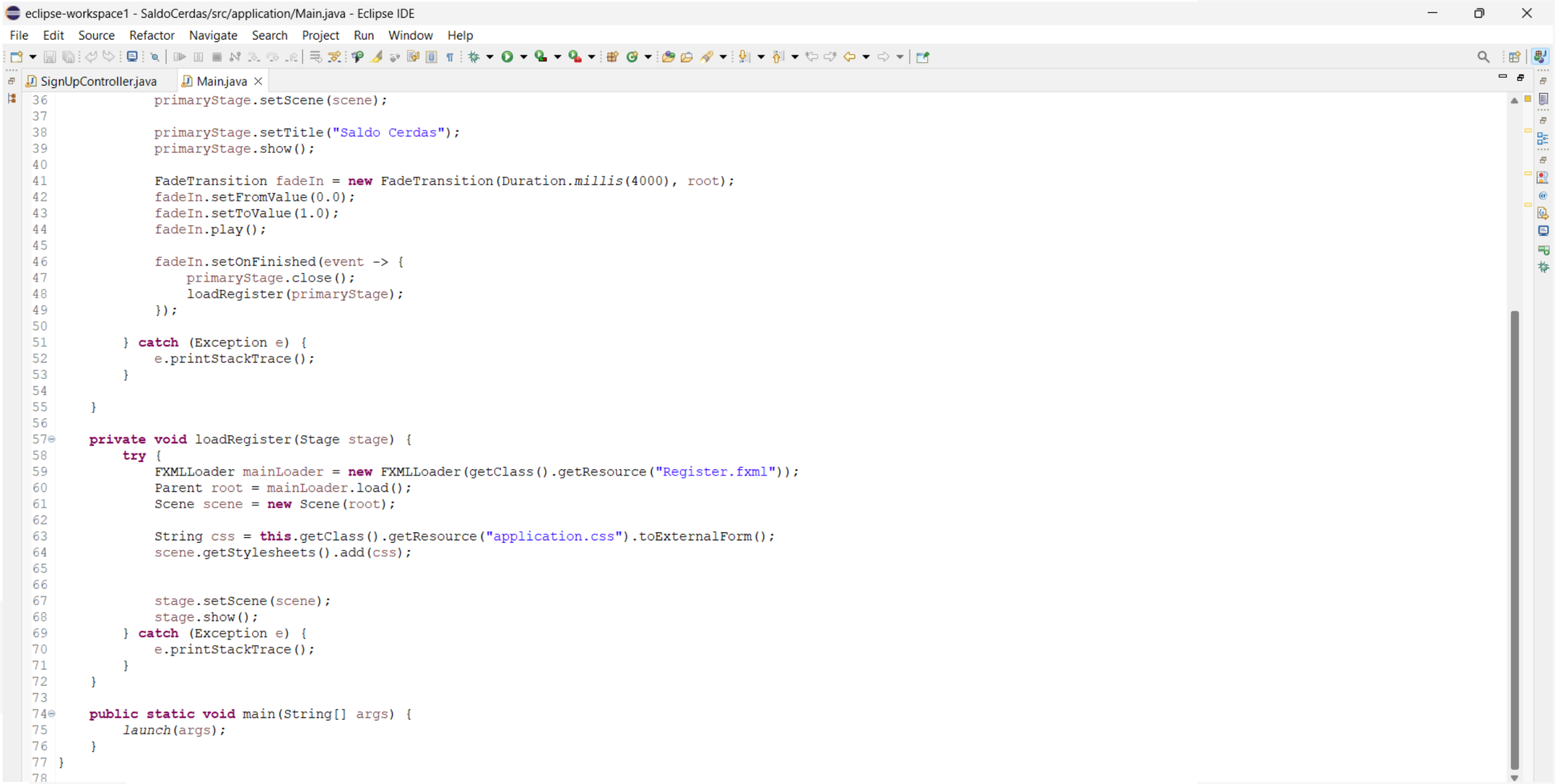
CODING DOCUMENTATION

JAVA (SPLASH SCREEN)



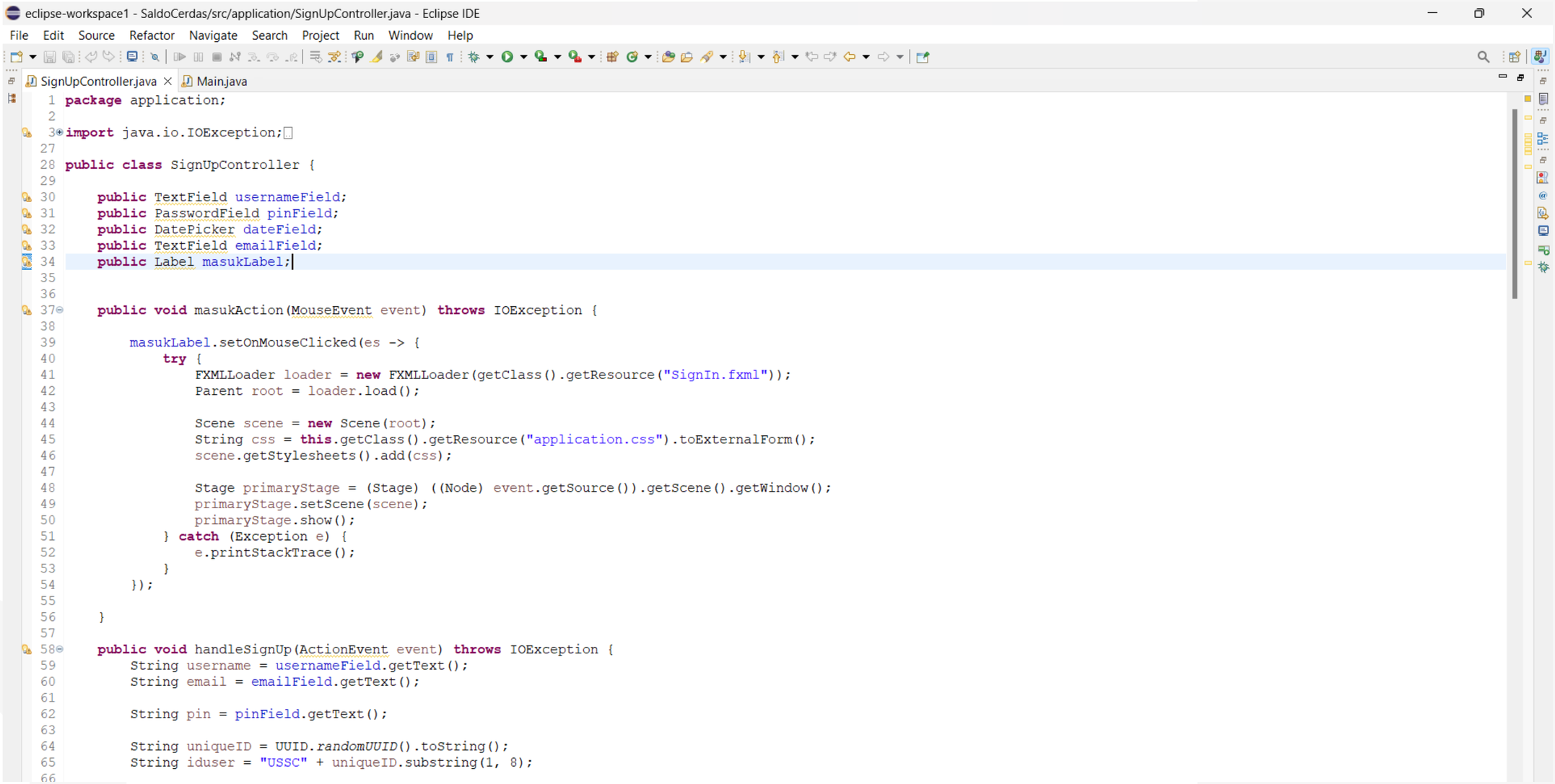

```java
package application;

import javafx.animation.FadeTransition;

public class Main extends Application {

    private static Stage primaryStage;

    @Override
    public void start(Stage primaryStage) throws IOException {

        try {
            FXMLLoader loader = new FXMLLoader(getClass().getResource("Splash.fxml"));
            Parent root = loader.load();

            Scene scene = new Scene(root);

            String css = this.getClass().getResource("application.css").toExternalForm();
            scene.getStylesheets().add(css);
            primaryStage.setScene(scene);

            primaryStage.setTitle("Saldo Cerdas");
            primaryStage.show();

            FadeTransition fadeIn = new FadeTransition(Duration.millis(4000), root);
            fadeIn.setFromValue(0.0);
            fadeIn.setToValue(1.0);
            fadeIn.play();

            fadeIn.setOnFinished(event -> {
                primaryStage.close();
                loadRegister(primaryStage);
            });

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

    }

    private void loadRegister(Stage stage) {
        try {
            FXMLLoader mainLoader = new FXMLLoader(getClass().getResource("Register.fxml"));
```

JAVA (SPLASH SCREEN)



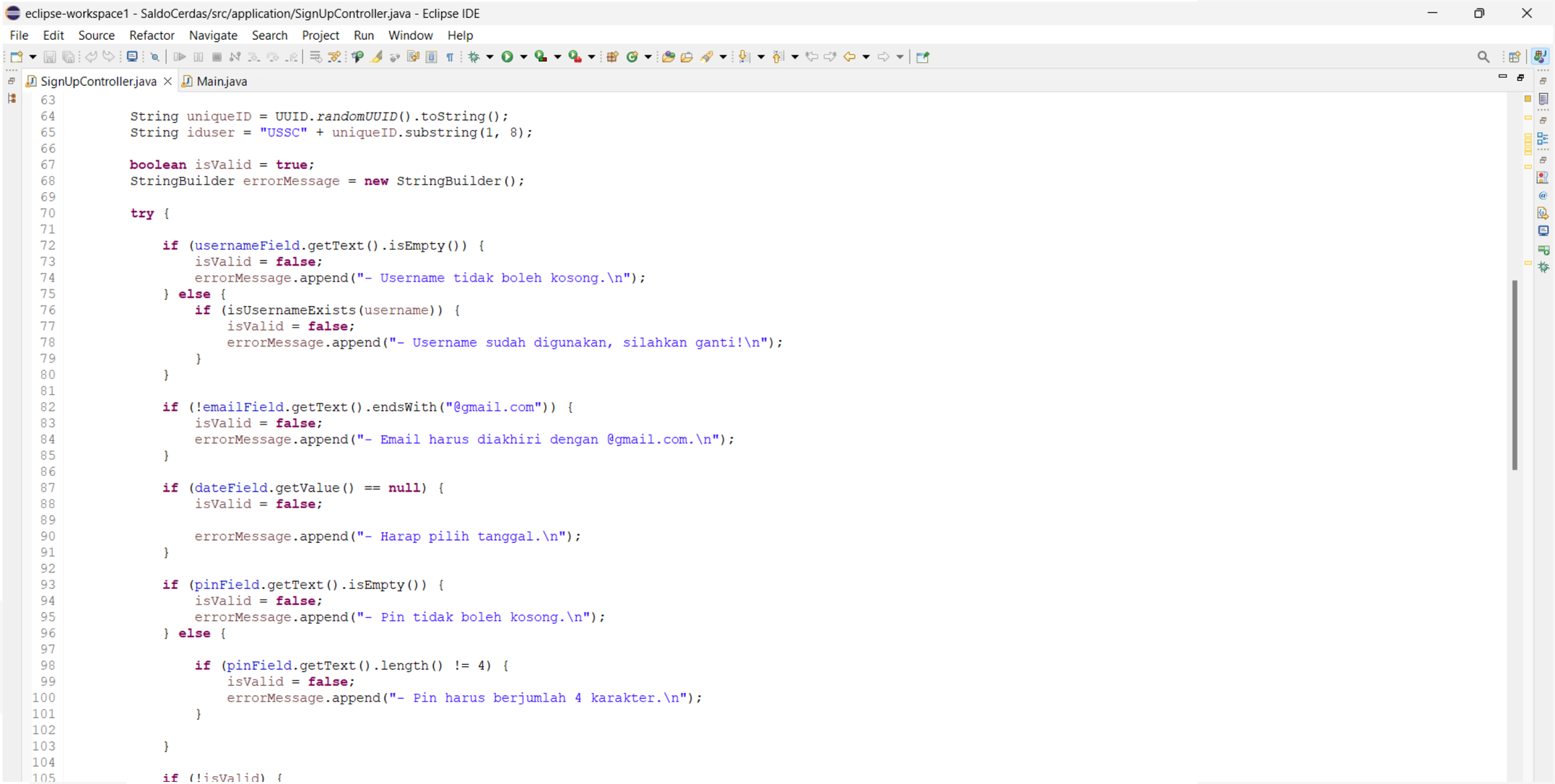

```
            primaryStage.setScene(scene);

            primaryStage.setTitle("Saldo Cerdas");
            primaryStage.show();

            FadeTransition fadeIn = new FadeTransition(Duration.millis(4000), root);
            fadeIn.setFromValue(0.0);
            fadeIn.setToValue(1.0);
            fadeIn.play();

            fadeIn.setOnFinished(event -> {
                primaryStage.close();
                loadRegister(primaryStage);
            });

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

    }

    private void loadRegister(Stage stage) {
        try {
            FXMLLoader mainLoader = new FXMLLoader(getClass().getResource("Register.fxml"));
            Parent root = mainLoader.load();
            Scene scene = new Scene(root);

            String css = this.getClass().getResource("application.css").toExternalForm();
            scene.getStylesheets().add(css);


            stage.setScene(scene);
            stage.show();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public static void main(String[] args) {
        launch(args);
    }
}
```

JAVA (SIGN UP PAGE)



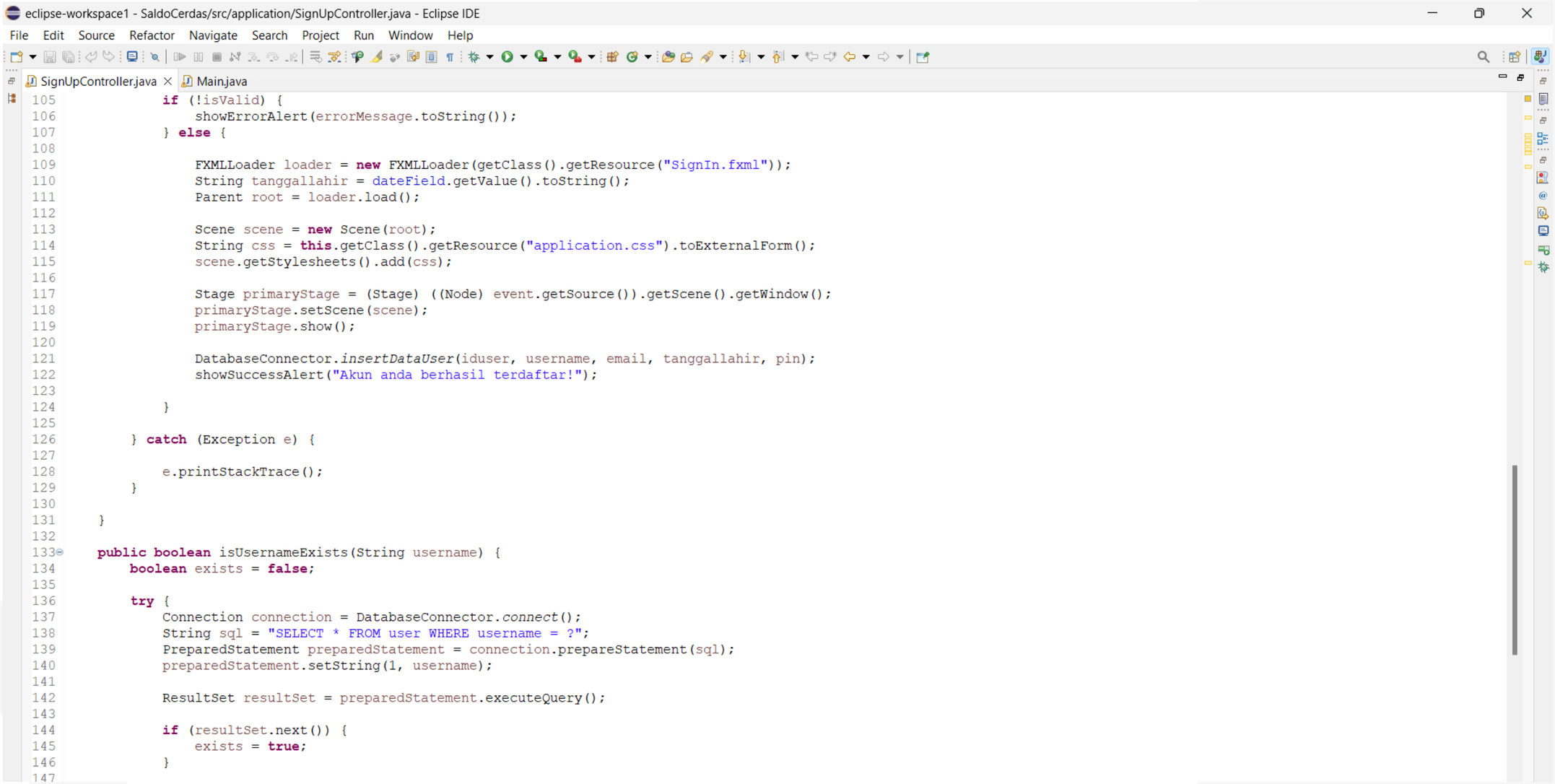

```java
package application;

import java.io.IOException;

public class SignUpController {

    public TextField usernameField;
    public PasswordField pinField;
    public DatePicker dateField;
    public TextField emailField;
    public Label masukLabel;


    public void masukAction(MouseEvent event) throws IOException {

        masukLabel.setOnMouseClicked(es -> {
            try {
                FXMLLoader loader = new FXMLLoader(getClass().getResource("SignIn.fxml"));
                Parent root = loader.load();

                Scene scene = new Scene(root);
                String css = this.getClass().getResource("application.css").toExternalForm();
                scene.getStylesheets().add(css);

                Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
                primaryStage.setScene(scene);
                primaryStage.show();
            } catch (Exception e) {
                e.printStackTrace();
            }
        });

    }

    public void handleSignUp(ActionEvent event) throws IOException {
        String username = usernameField.getText();
        String email = emailField.getText();

        String pin = pinField.getText();

        String uniqueID = UUID.randomUUID().toString();
        String iduser = "USSC" + uniqueID.substring(1, 8);
```

JAVA (SIGN UP PAGE)



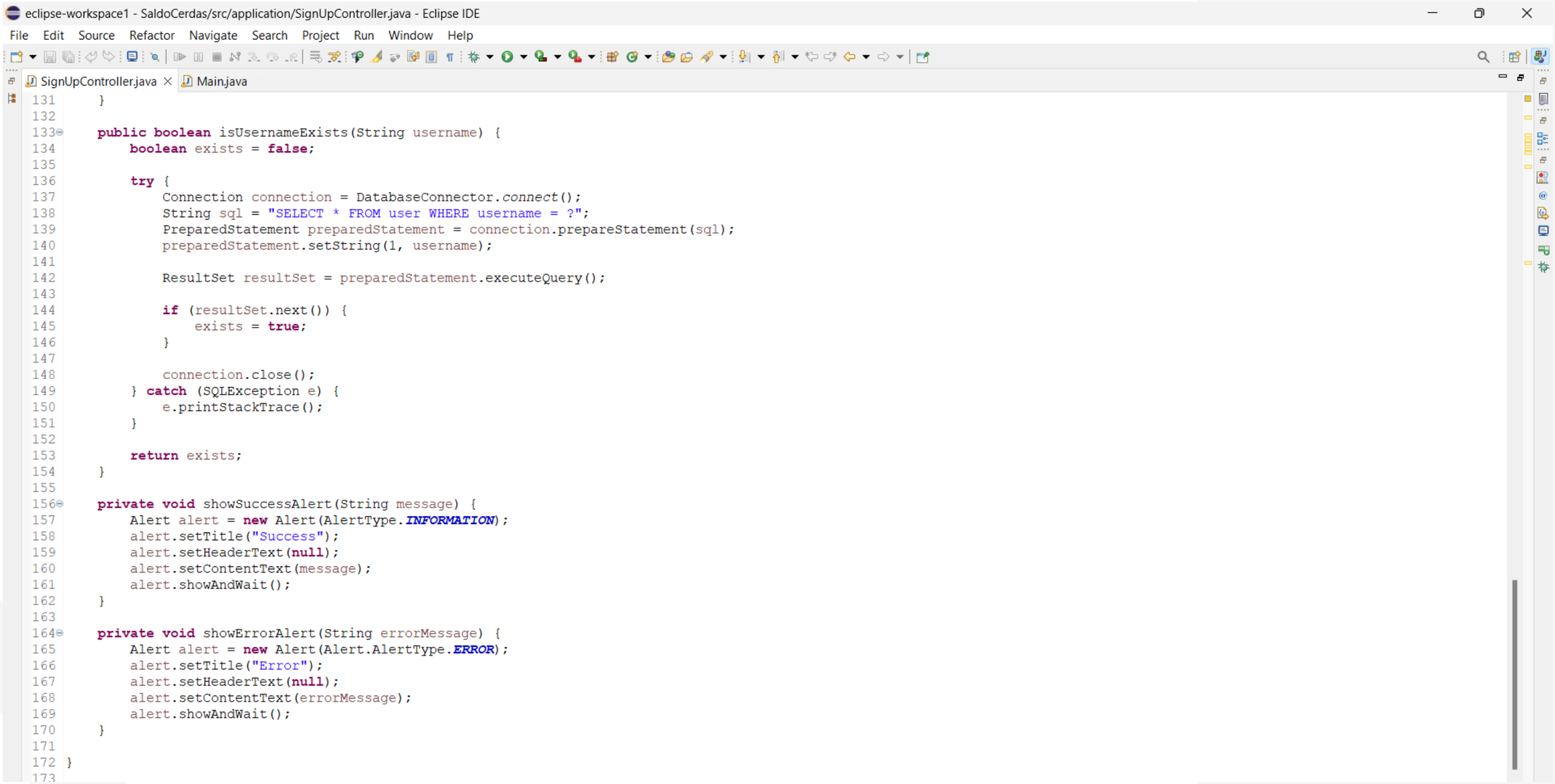

```
63
64          String uniqueID = UUID.randomUUID().toString();
65          String iduser = "USSC" + uniqueID.substring(1, 8);
66
67          boolean isValid = true;
68          StringBuilder errorMessage = new StringBuilder();
69
70          try {
71
72              if (usernameField.getText().isEmpty()) {
73                  isValid = false;
74                  errorMessage.append("- Username tidak boleh kosong.\n");
75              } else {
76                  if (isUsernameExists(username)) {
77                      isValid = false;
78                      errorMessage.append("- Username sudah digunakan, silahkan ganti!\n");
79                  }
80              }
81
82              if (!emailField.getText().endsWith("@gmail.com")) {
83                  isValid = false;
84                  errorMessage.append("- Email harus diakhiri dengan @gmail.com.\n");
85              }
86
87              if (dateField.getValue() == null) {
88                  isValid = false;
89
90                  errorMessage.append("- Harap pilih tanggal.\n");
91              }
92
93              if (pinField.getText().isEmpty()) {
94                  isValid = false;
95                  errorMessage.append("- Pin tidak boleh kosong.\n");
96              } else {
97
98                  if (pinField.getText().length() != 4) {
99                      isValid = false;
100                     errorMessage.append("- Pin harus berjumlah 4 karakter.\n");
101                 }
102
103             }
104
105             if (!isValid) {
```

JAVA (SIGN UP PAGE)



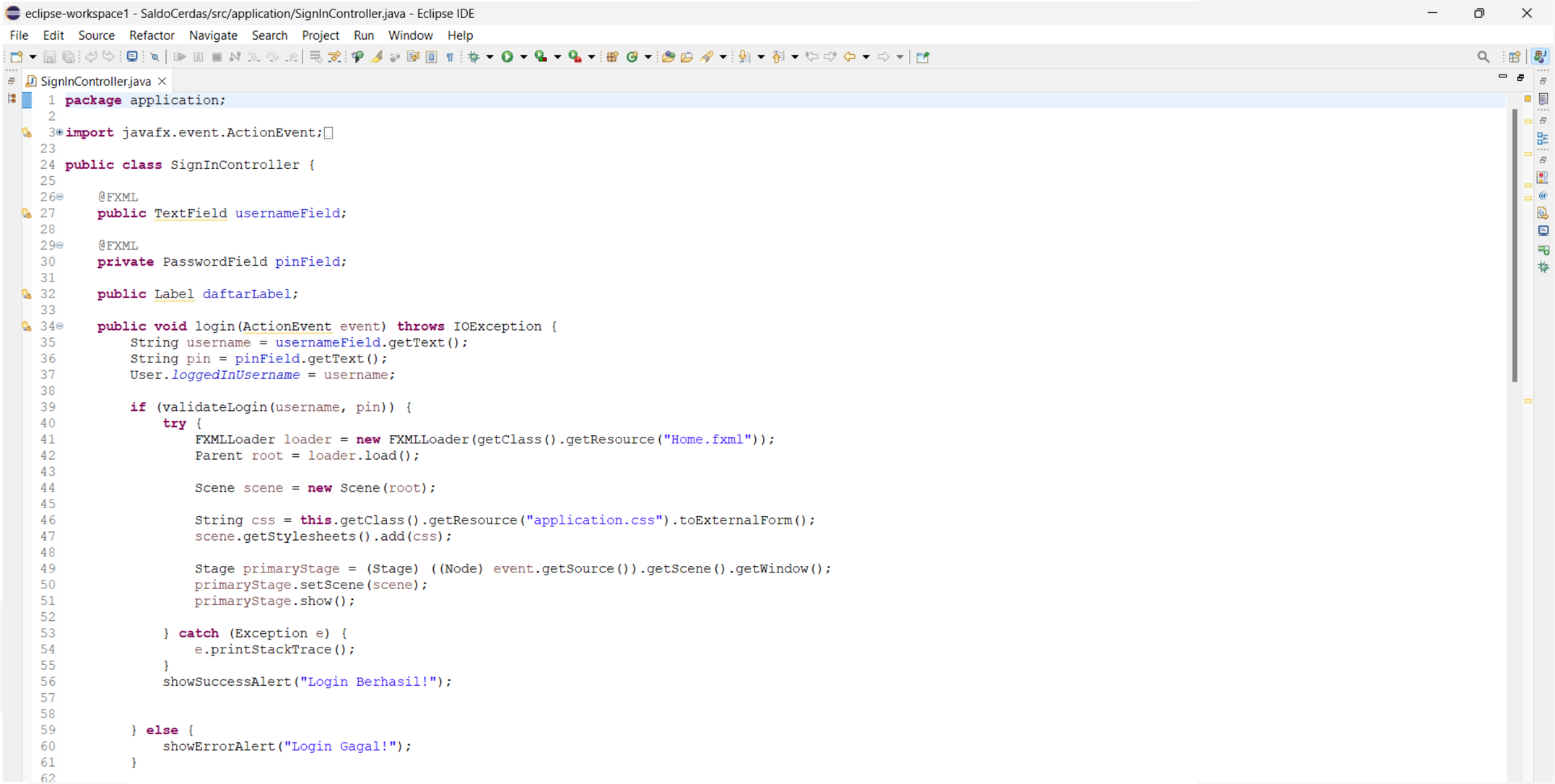

```
            if (!isValid) {
                showErrorAlert(errorMessage.toString());
            } else {

                FXMLLoader loader = new FXMLLoader(getClass().getResource("SignIn.fxml"));
                String tanggallahir = dateField.getValue().toString();
                Parent root = loader.load();

                Scene scene = new Scene(root);
                String css = this.getClass().getResource("application.css").toExternalForm();
                scene.getStylesheets().add(css);

                Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
                primaryStage.setScene(scene);
                primaryStage.show();

                DatabaseConnector.insertDataUser(iduser, username, email, tanggallahir, pin);
                showSuccessAlert("Akun anda berhasil terdaftar!");

            }

        } catch (Exception e) {

            e.printStackTrace();
        }

    }

    public boolean isUsernameExists(String username) {
        boolean exists = false;

        try {
            Connection connection = DatabaseConnector.connect();
            String sql = "SELECT * FROM user WHERE username = ?";
            PreparedStatement preparedStatement = connection.prepareStatement(sql);
            preparedStatement.setString(1, username);

            ResultSet resultSet = preparedStatement.executeQuery();

            if (resultSet.next()) {
                exists = true;
            }
```

JAVA (SIGN UP PAGE)



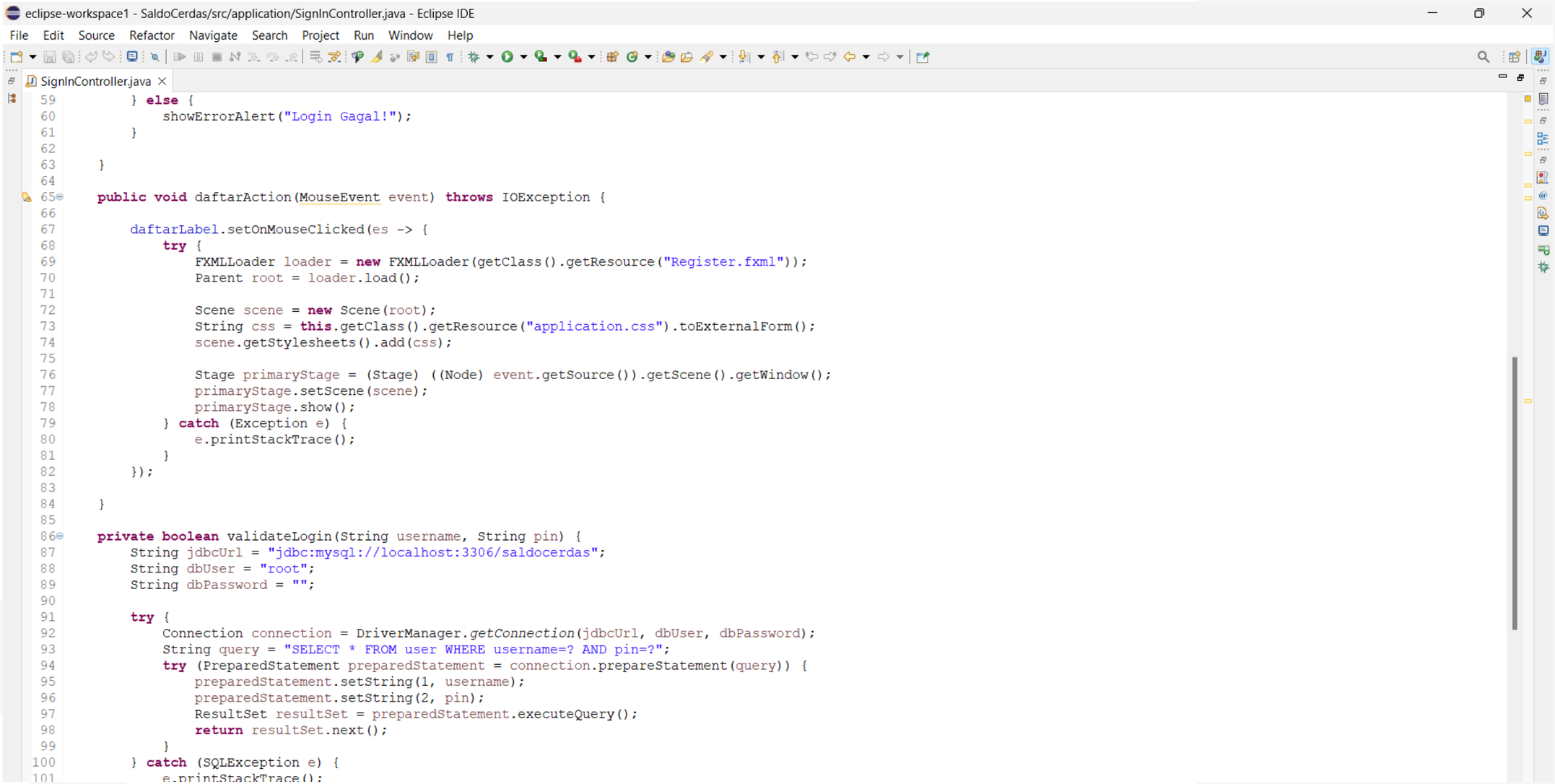

```
131     }
132
133     public boolean isUsernameExists(String username) {
134         boolean exists = false;
135
136         try {
137             Connection connection = DatabaseConnector.connect();
138             String sql = "SELECT * FROM user WHERE username = ?";
139             PreparedStatement preparedStatement = connection.prepareStatement(sql);
140             preparedStatement.setString(1, username);
141
142             ResultSet resultSet = preparedStatement.executeQuery();
143
144             if (resultSet.next()) {
145                 exists = true;
146             }
147
148             connection.close();
149         } catch (SQLException e) {
150             e.printStackTrace();
151         }
152
153         return exists;
154     }
155
156     private void showSuccessAlert(String message) {
157         Alert alert = new Alert(AlertType.INFORMATION);
158         alert.setTitle("Success");
159         alert.setHeaderText(null);
160         alert.setContentText(message);
161         alert.showAndWait();
162     }
163
164     private void showErrorAlert(String errorMessage) {
165         Alert alert = new Alert(Alert.AlertType.ERROR);
166         alert.setTitle("Error");
167         alert.setHeaderText(null);
168         alert.setContentText(errorMessage);
169         alert.showAndWait();
170     }
171
172 }
173
```

JAVA (SIGN IN PAGE)



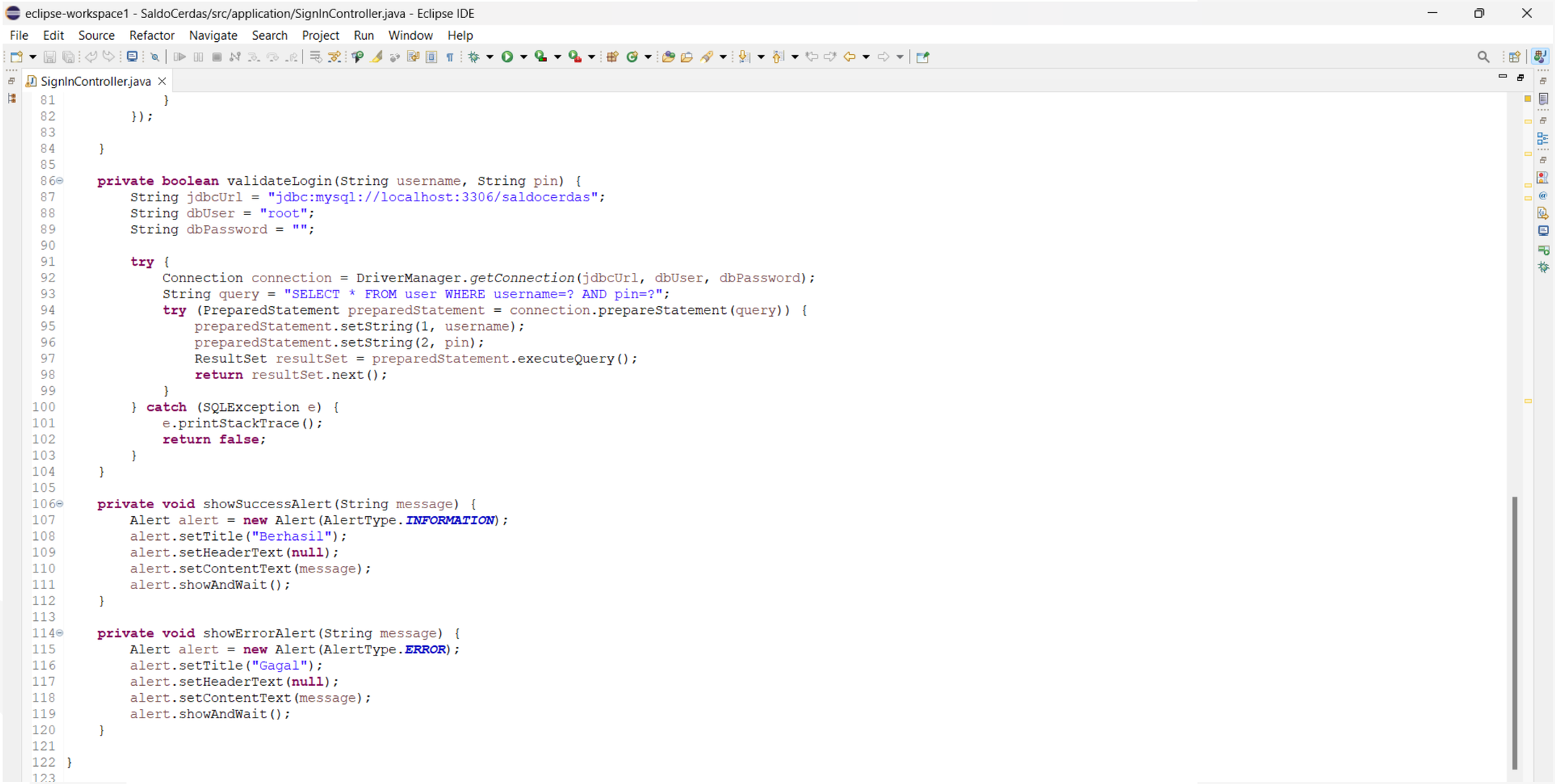

```java
package application;

import javafx.event.ActionEvent;

public class SignInController {

    @FXML
    public TextField usernameField;

    @FXML
    private PasswordField pinField;

    public Label daftarLabel;

    public void login(ActionEvent event) throws IOException {
        String username = usernameField.getText();
        String pin = pinField.getText();
        User.loggedInUsername = username;

        if (validateLogin(username, pin)) {
            try {
                FXMLLoader loader = new FXMLLoader(getClass().getResource("Home.fxml"));
                Parent root = loader.load();

                Scene scene = new Scene(root);

                String css = this.getClass().getResource("application.css").toExternalForm();
                scene.getStylesheets().add(css);

                Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
                primaryStage.setScene(scene);
                primaryStage.show();

            } catch (Exception e) {
                e.printStackTrace();
            }
            showSuccessAlert("Login Berhasil!");


        } else {
            showErrorAlert("Login Gagal!");
        }

```

JAVA (SIGN IN PAGE)



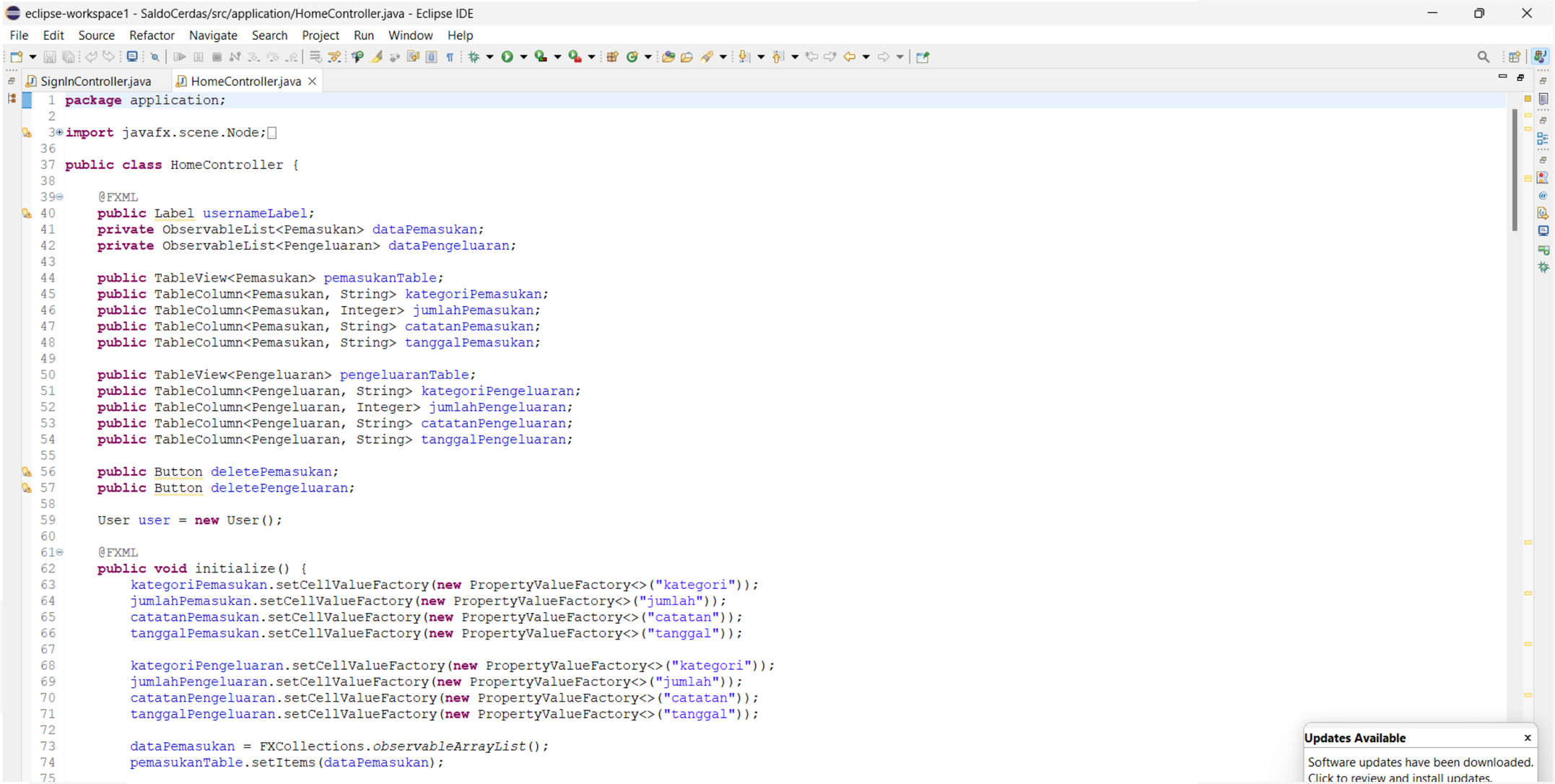

```
59          } else {
60              showErrorAlert("Login Gagal!");
61          }
62
63      }
64
65      public void daftarAction(MouseEvent event) throws IOException {
66
67          daftarLabel.setOnMouseClicked(es -> {
68              try {
69                  FXMLLoader loader = new FXMLLoader(getClass().getResource("Register.fxml"));
70                  Parent root = loader.load();
71
72                  Scene scene = new Scene(root);
73                  String css = this.getClass().getResource("application.css").toExternalForm();
74                  scene.getStylesheets().add(css);
75
76                  Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
77                  primaryStage.setScene(scene);
78                  primaryStage.show();
79              } catch (Exception e) {
80                  e.printStackTrace();
81              }
82          });
83
84      }
85
86      private boolean validateLogin(String username, String pin) {
87          String jdbcUrl = "jdbc:mysql://localhost:3306/saldocerdas";
88          String dbUser = "root";
89          String dbPassword = "";
90
91          try {
92              Connection connection = DriverManager.getConnection(jdbcUrl, dbUser, dbPassword);
93              String query = "SELECT * FROM user WHERE username=? AND pin=?";
94              try (PreparedStatement preparedStatement = connection.prepareStatement(query)) {
95                  preparedStatement.setString(1, username);
96                  preparedStatement.setString(2, pin);
97                  ResultSet resultSet = preparedStatement.executeQuery();
98                  return resultSet.next();
99              }
100         } catch (SQLException e) {
101             e.printStackTrace();
```

JAVA (SIGN IN PAGE)

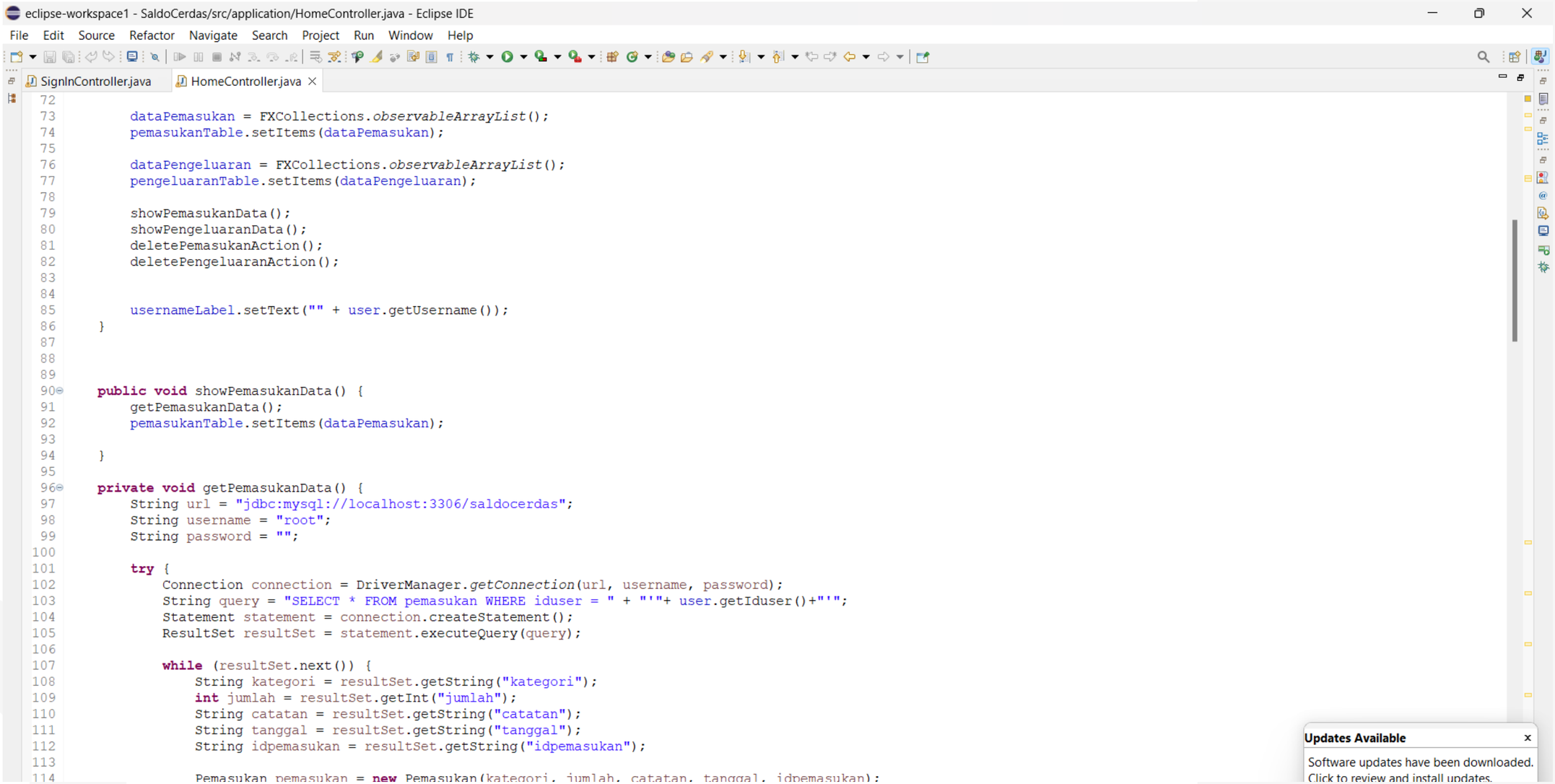



```java
            }
        });

    }

    private boolean validateLogin(String username, String pin) {
        String jdbcUrl = "jdbc:mysql://localhost:3306/saldocerdas";
        String dbUser = "root";
        String dbPassword = "";

        try {
            Connection connection = DriverManager.getConnection(jdbcUrl, dbUser, dbPassword);
            String query = "SELECT * FROM user WHERE username=? AND pin=?";
            try (PreparedStatement preparedStatement = connection.prepareStatement(query)) {
                preparedStatement.setString(1, username);
                preparedStatement.setString(2, pin);
                ResultSet resultSet = preparedStatement.executeQuery();
                return resultSet.next();
            }
        } catch (SQLException e) {
            e.printStackTrace();
            return false;
        }
    }

    private void showSuccessAlert(String message) {
        Alert alert = new Alert(AlertType.INFORMATION);
        alert.setTitle("Berhasil");
        alert.setHeaderText(null);
        alert.setContentText(message);
        alert.showAndWait();
    }

    private void showErrorAlert(String message) {
        Alert alert = new Alert(AlertType.ERROR);
        alert.setTitle("Gagal");
        alert.setHeaderText(null);
        alert.setContentText(message);
        alert.showAndWait();
    }

}
```

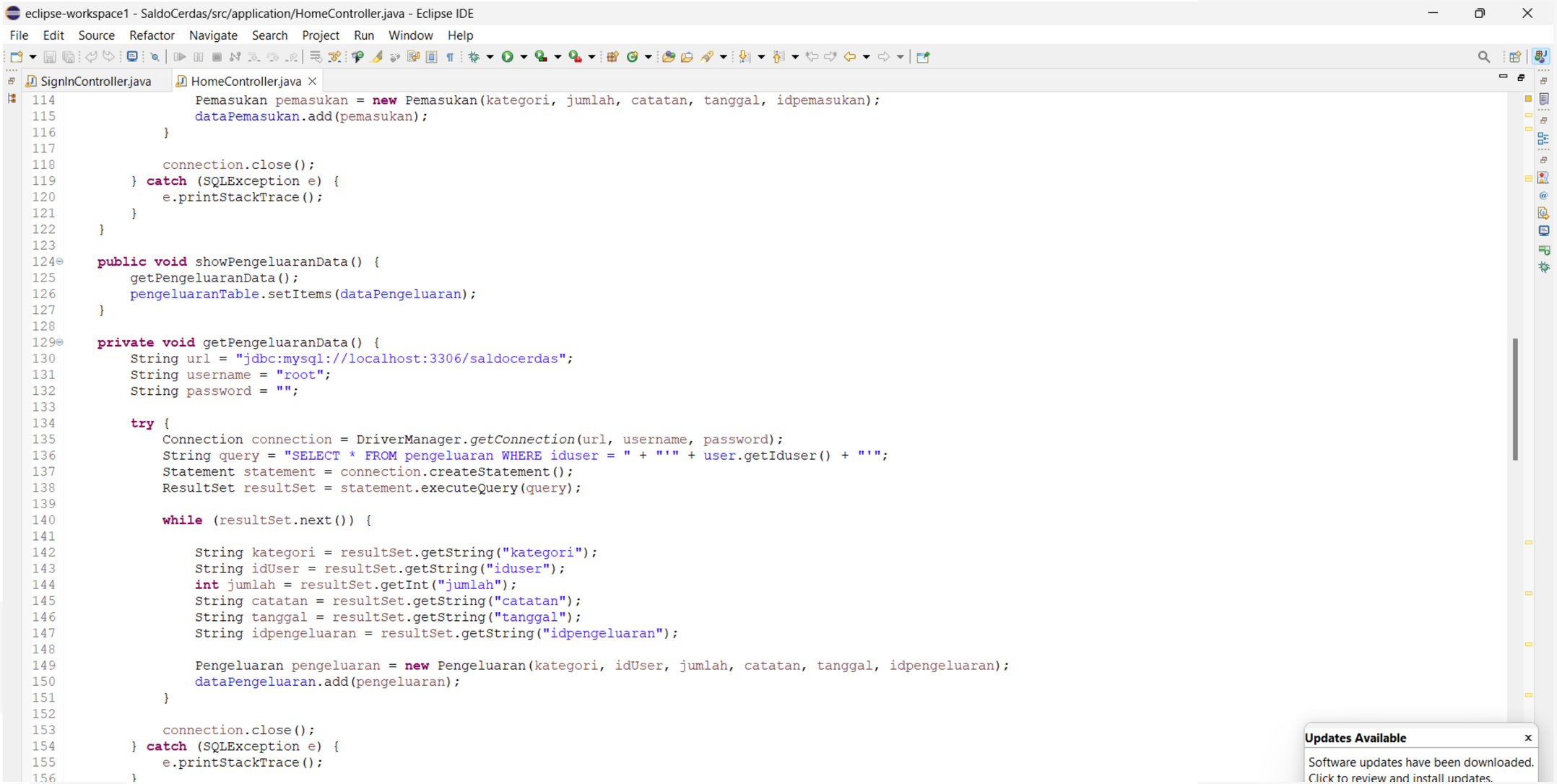
JAVA (HOME PAGE)



JAVA (HOME PAGE)

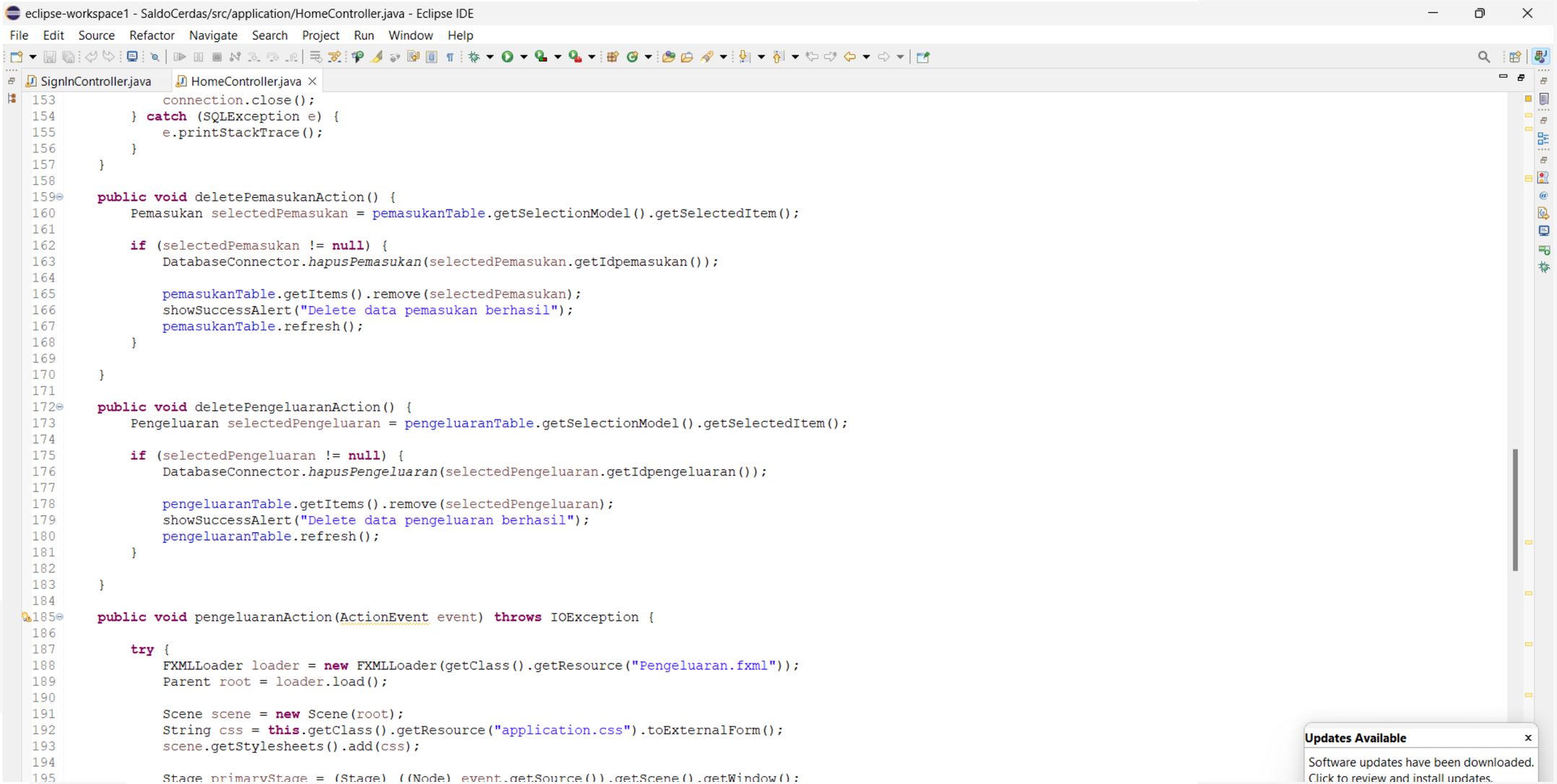


JAVA (HOME PAGE)

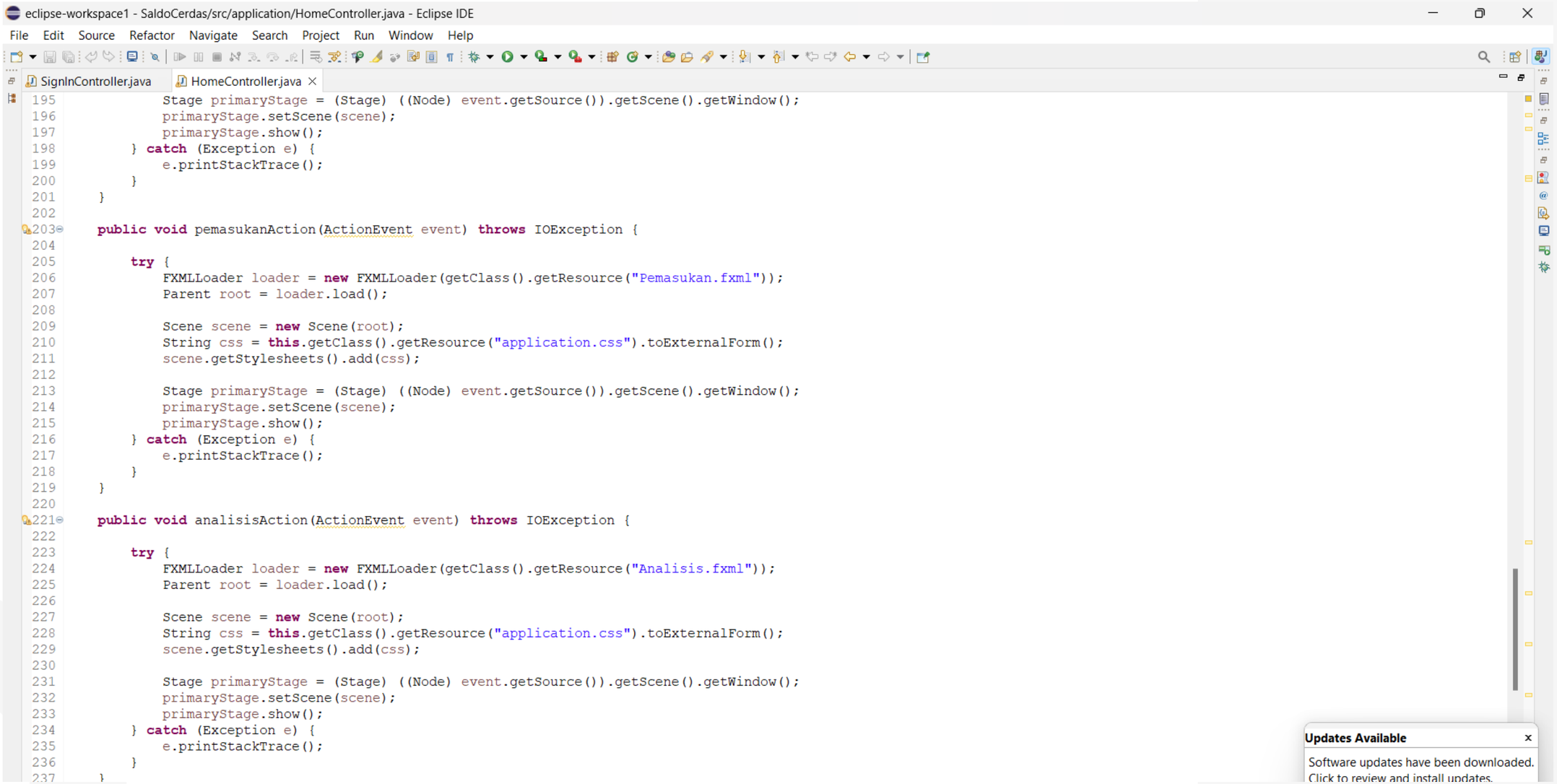



```
                Pemasukan pemasukan = new Pemasukan(kategori, jumlah, catatan, tanggal, idpemasukan);
                dataPemasukan.add(pemasukan);
            }

            connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public void showPengeluaranData() {
        getPengeluaranData();
        pengeluaranTable.setItems(dataPengeluaran);
    }

    private void getPengeluaranData() {
        String url = "jdbc:mysql://localhost:3306/saldocerdas";
        String username = "root";
        String password = "";

        try {
            Connection connection = DriverManager.getConnection(url, username, password);
            String query = "SELECT * FROM pengeluaran WHERE iduser = " + "'" + user.getIduser() + "'";
            Statement statement = connection.createStatement();
            ResultSet resultSet = statement.executeQuery(query);

            while (resultSet.next()) {

                String kategori = resultSet.getString("kategori");
                String idUser = resultSet.getString("iduser");
                int jumlah = resultSet.getInt("jumlah");
                String catatan = resultSet.getString("catatan");
                String tanggal = resultSet.getString("tanggal");
                String idpengeluaran = resultSet.getString("idpengeluaran");

                Pengeluaran pengeluaran = new Pengeluaran(kategori, idUser, jumlah, catatan, tanggal, idpengeluaran);
                dataPengeluaran.add(pengeluaran);
            }

            connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
```

JAVA (HOME PAGE)

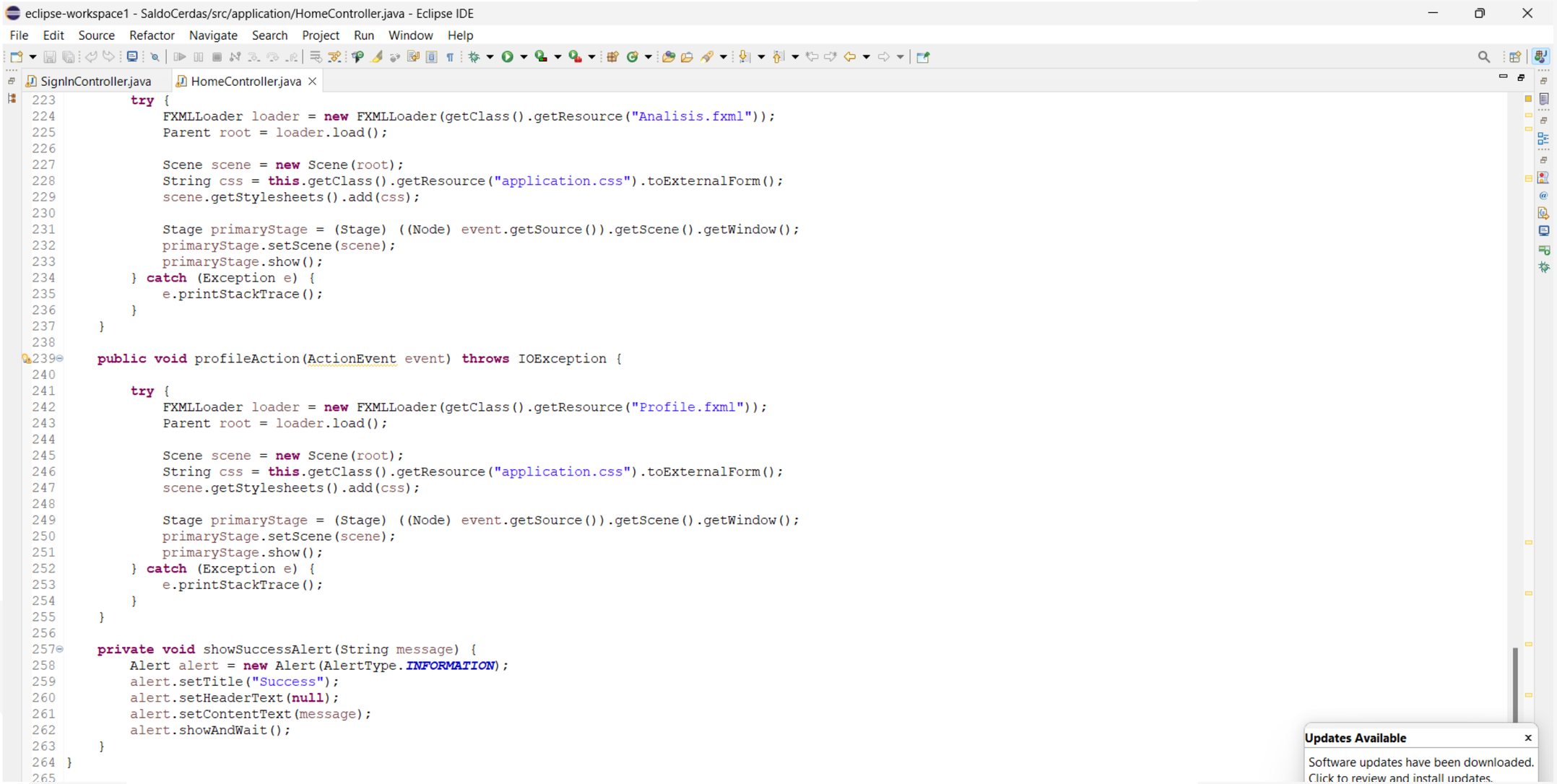


JAVA (HOME PAGE)

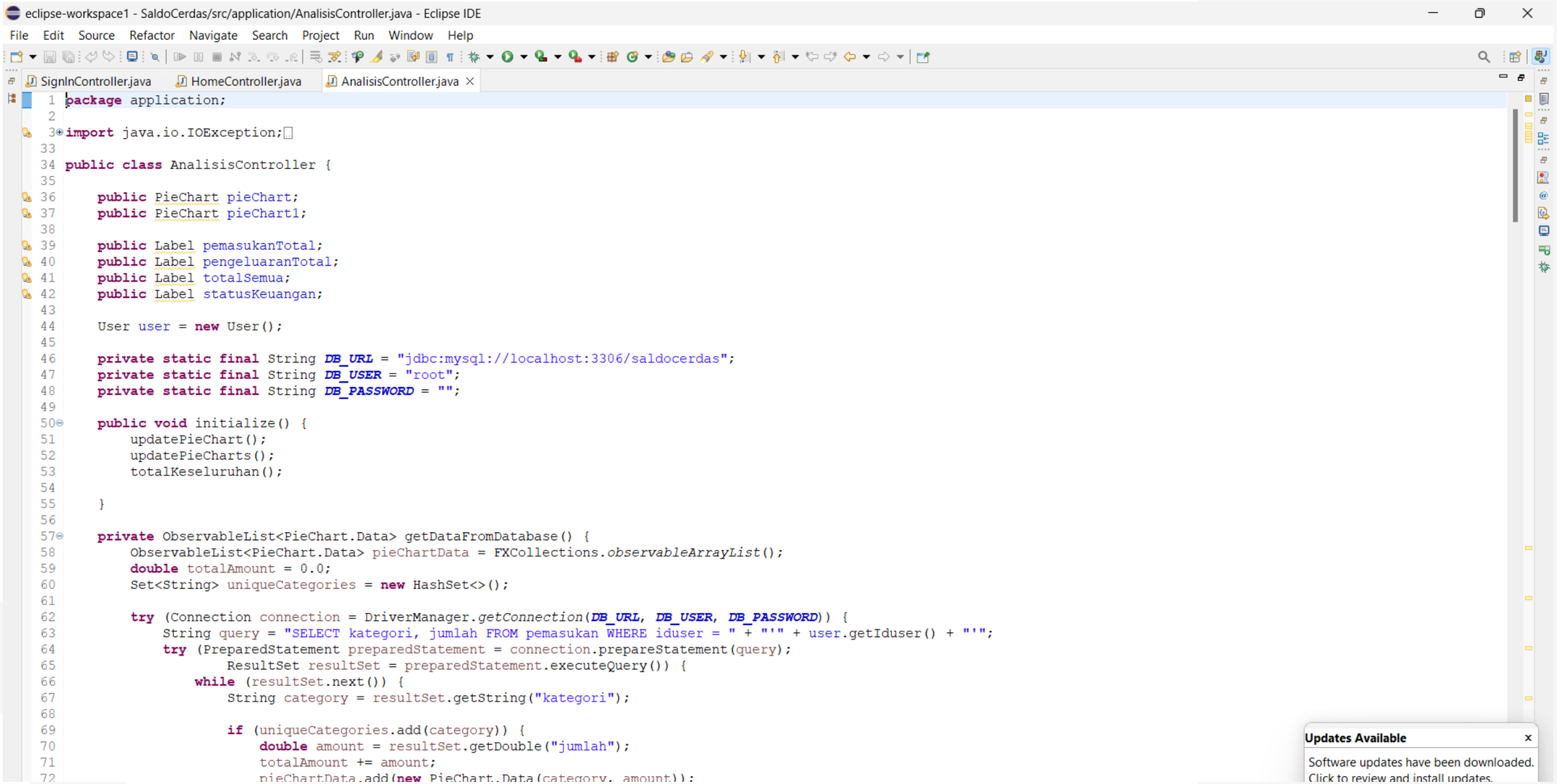



```
            Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
            primaryStage.setScene(scene);
            primaryStage.show();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public void pemasukanAction(ActionEvent event) throws IOException {

        try {
            FXMLLoader loader = new FXMLLoader(getClass().getResource("Pemasukan.fxml"));
            Parent root = loader.load();

            Scene scene = new Scene(root);
            String css = this.getClass().getResource("application.css").toExternalForm();
            scene.getStylesheets().add(css);

            Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
            primaryStage.setScene(scene);
            primaryStage.show();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public void analisisAction(ActionEvent event) throws IOException {

        try {
            FXMLLoader loader = new FXMLLoader(getClass().getResource("Analisis.fxml"));
            Parent root = loader.load();

            Scene scene = new Scene(root);
            String css = this.getClass().getResource("application.css").toExternalForm();
            scene.getStylesheets().add(css);

            Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
            primaryStage.setScene(scene);
            primaryStage.show();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

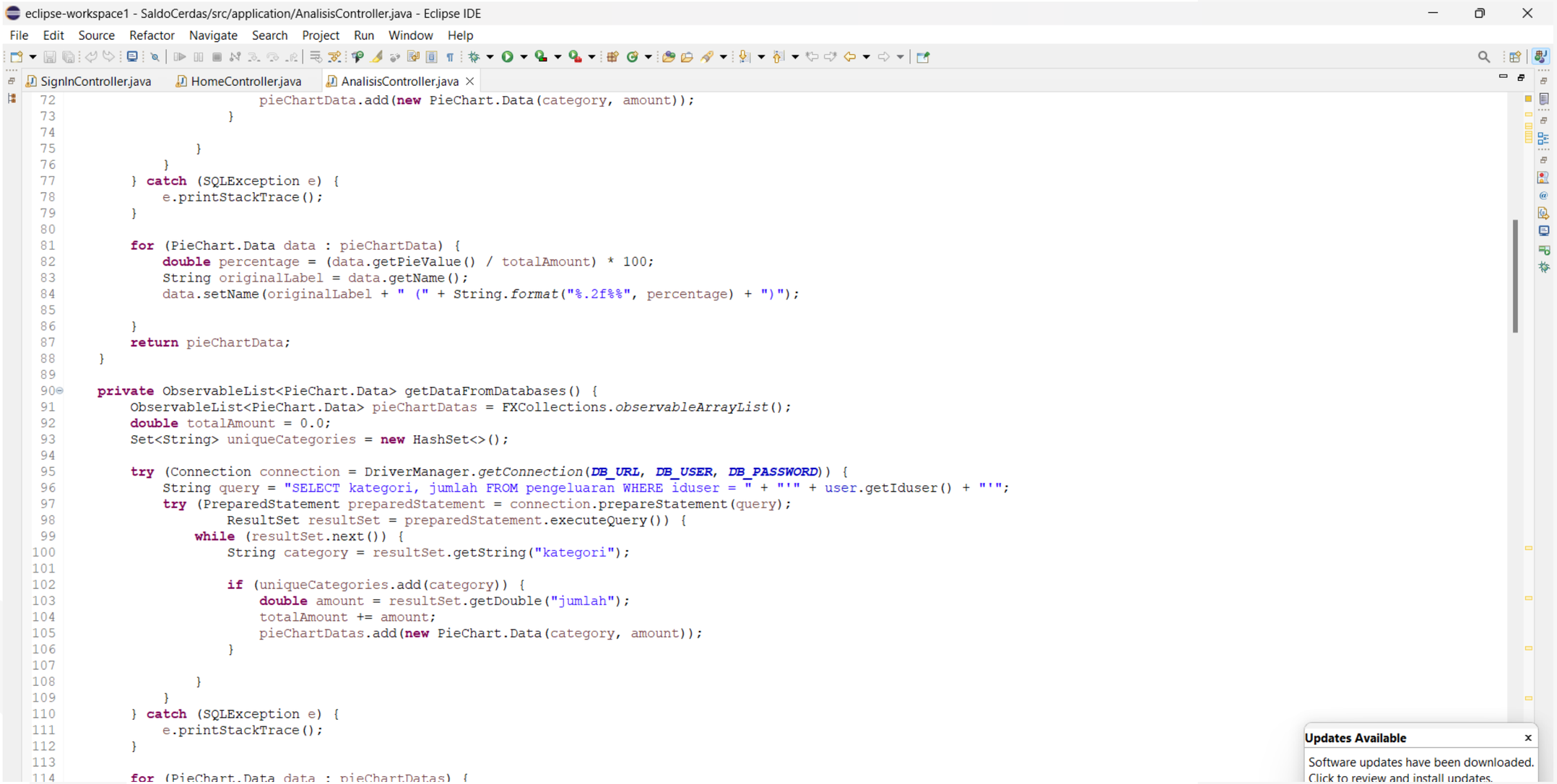
JAVA (HOME PAGE)



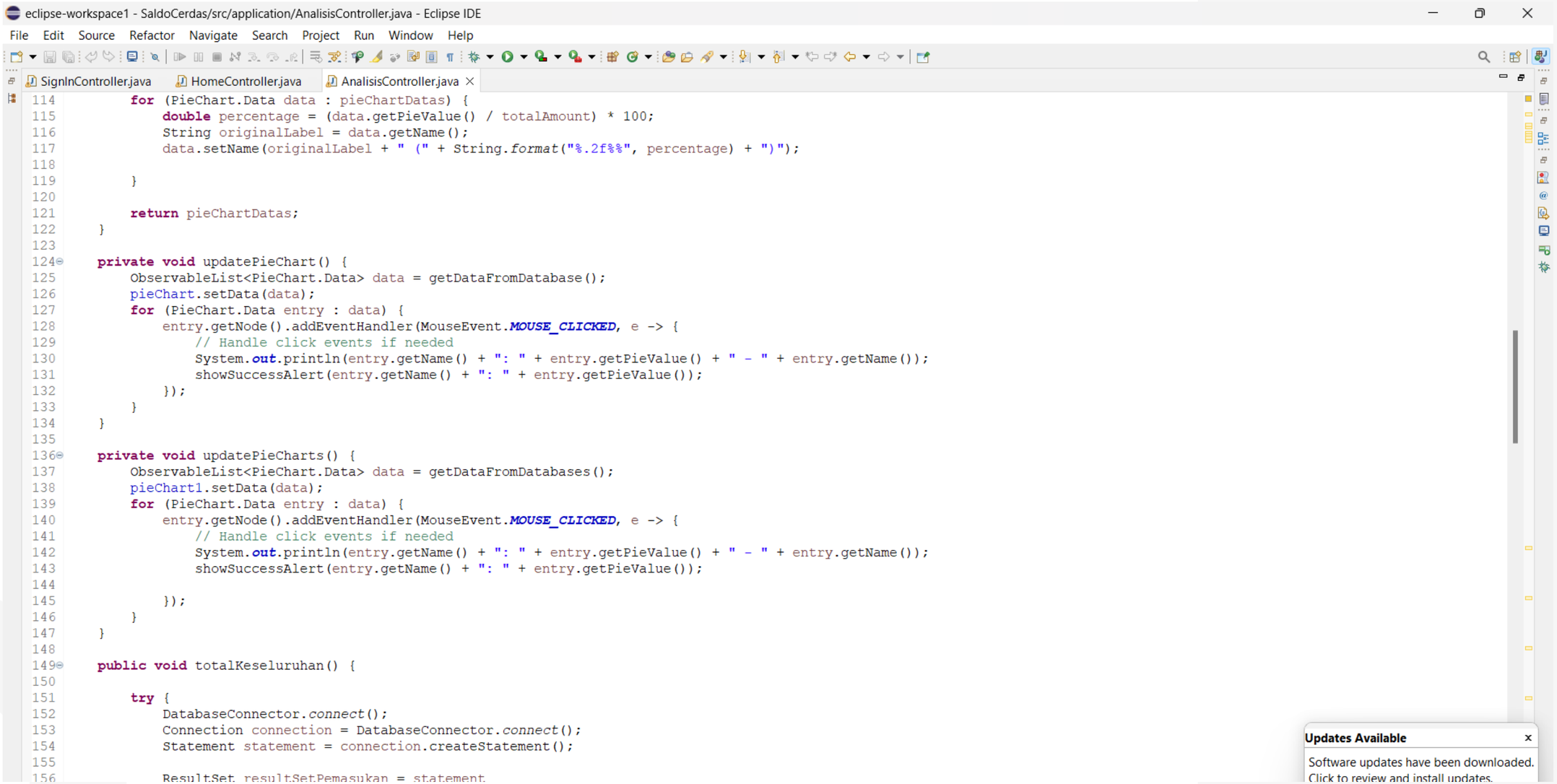
JAVA (ANALISIS PAGE)



JAVA (ANALISIS PAGE)

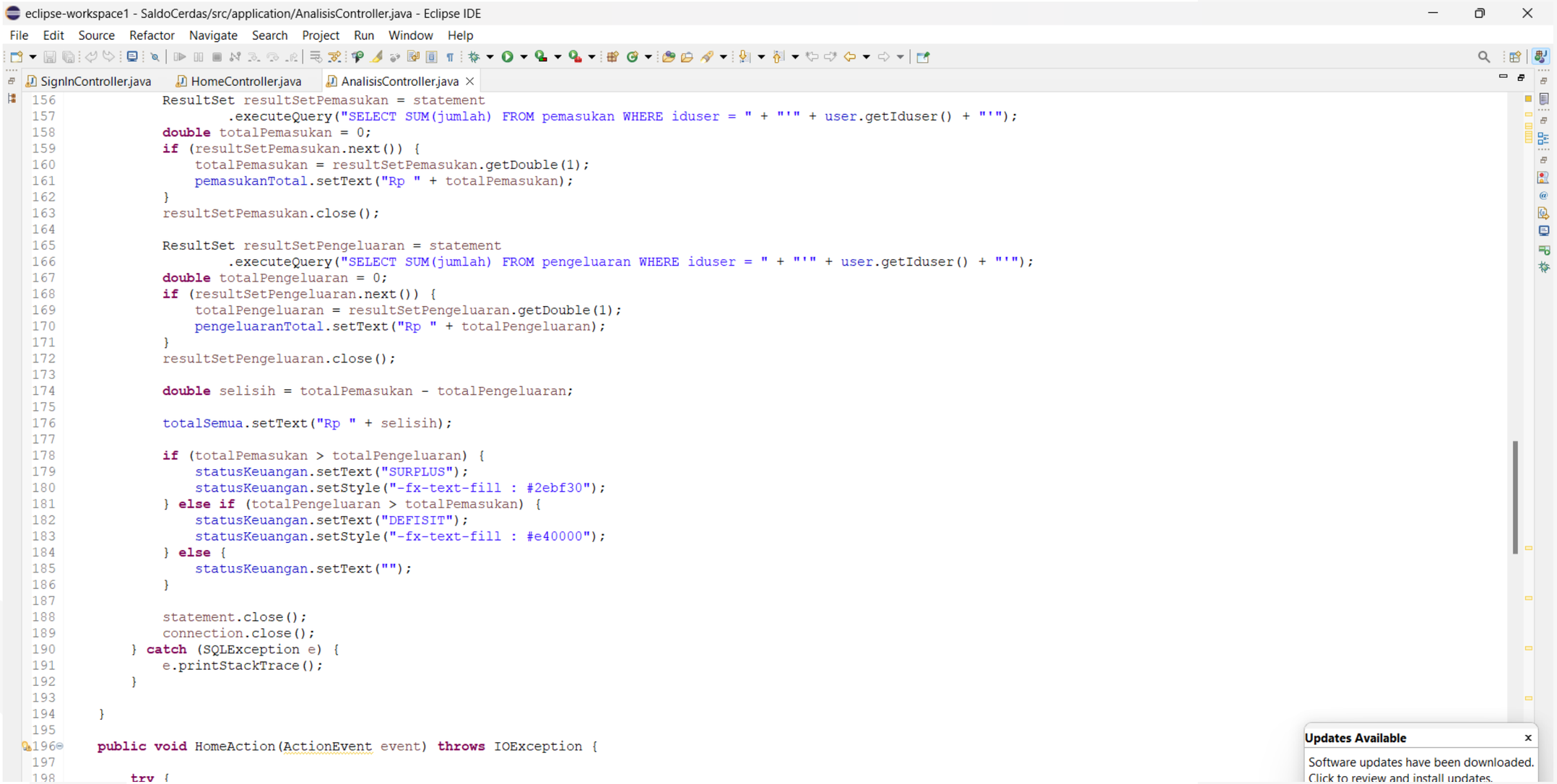


JAVA (ANALISIS PAGE)

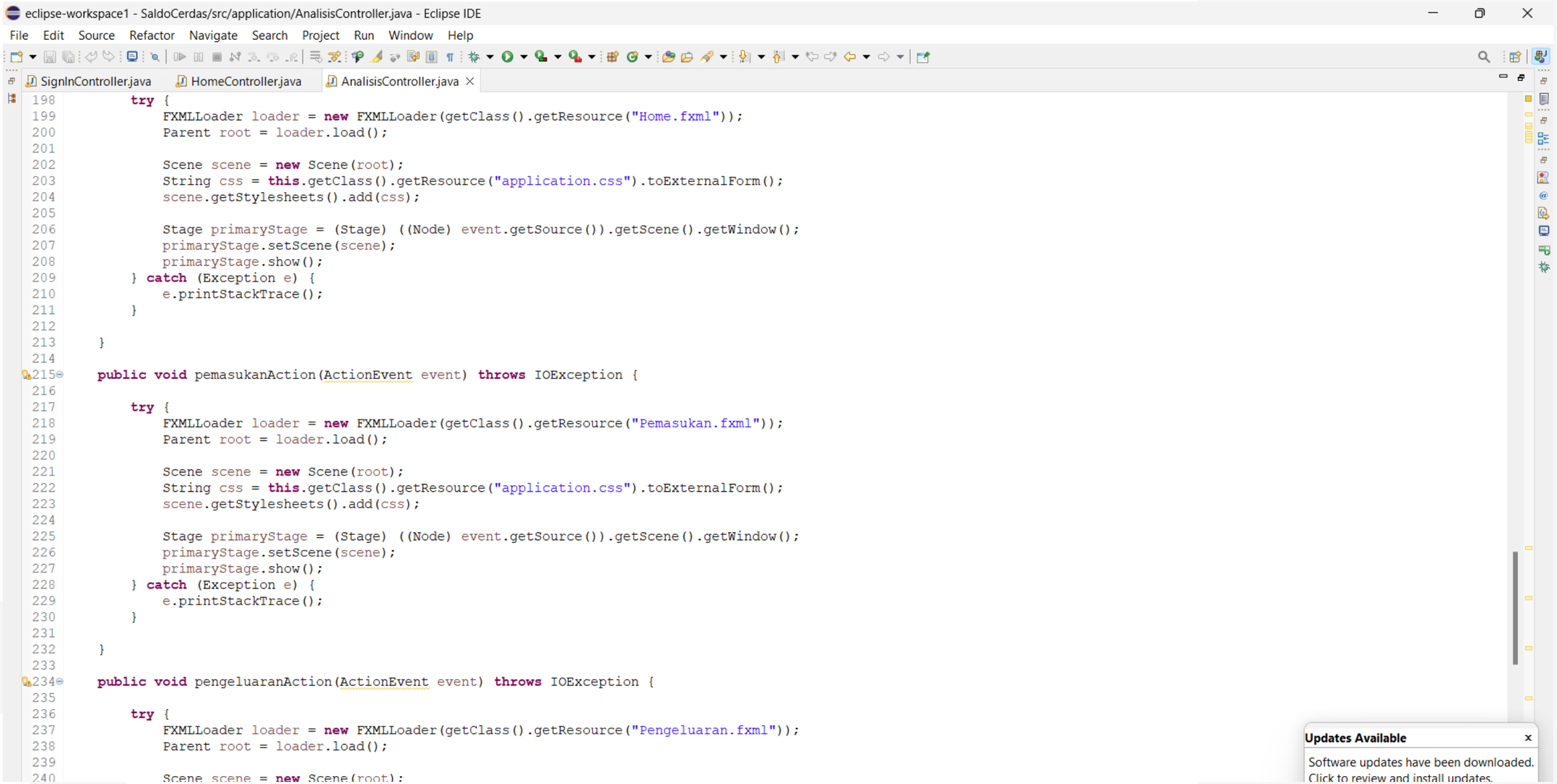



```
114         for (PieChart.Data data : pieChartDatas) {
115             double percentage = (data.getPieValue() / totalAmount) * 100;
116             String originalLabel = data.getName();
117             data.setName(originalLabel + " (" + String.format("%.2f%%", percentage) + ")");
118 
119         }
120 
121         return pieChartDatas;
122     }
123 
124     private void updatePieChart() {
125         ObservableList<PieChart.Data> data = getDataFromDatabase();
126         pieChart.setData(data);
127         for (PieChart.Data entry : data) {
128             entry.getNode().addEventHandler(MouseEvent.MOUSE_CLICKED, e -> {
129                 // Handle click events if needed
130                 System.out.println(entry.getName() + ": " + entry.getPieValue() + " - " + entry.getName());
131                 showSuccessAlert(entry.getName() + ": " + entry.getPieValue());
132             });
133         }
134     }
135 
136     private void updatePieCharts() {
137         ObservableList<PieChart.Data> data = getDataFromDatabases();
138         pieChart1.setData(data);
139         for (PieChart.Data entry : data) {
140             entry.getNode().addEventHandler(MouseEvent.MOUSE_CLICKED, e -> {
141                 // Handle click events if needed
142                 System.out.println(entry.getName() + ": " + entry.getPieValue() + " - " + entry.getName());
143                 showSuccessAlert(entry.getName() + ": " + entry.getPieValue());
144 
145             });
146         }
147     }
148 
149     public void totalKeseluruhan() {
150 
151         try {
152             DatabaseConnector.connect();
153             Connection connection = DatabaseConnector.connect();
154             Statement statement = connection.createStatement();
155 
156             ResultSet resultSetPemasukan = statement
```

JAVA (ANALISIS PAGE)

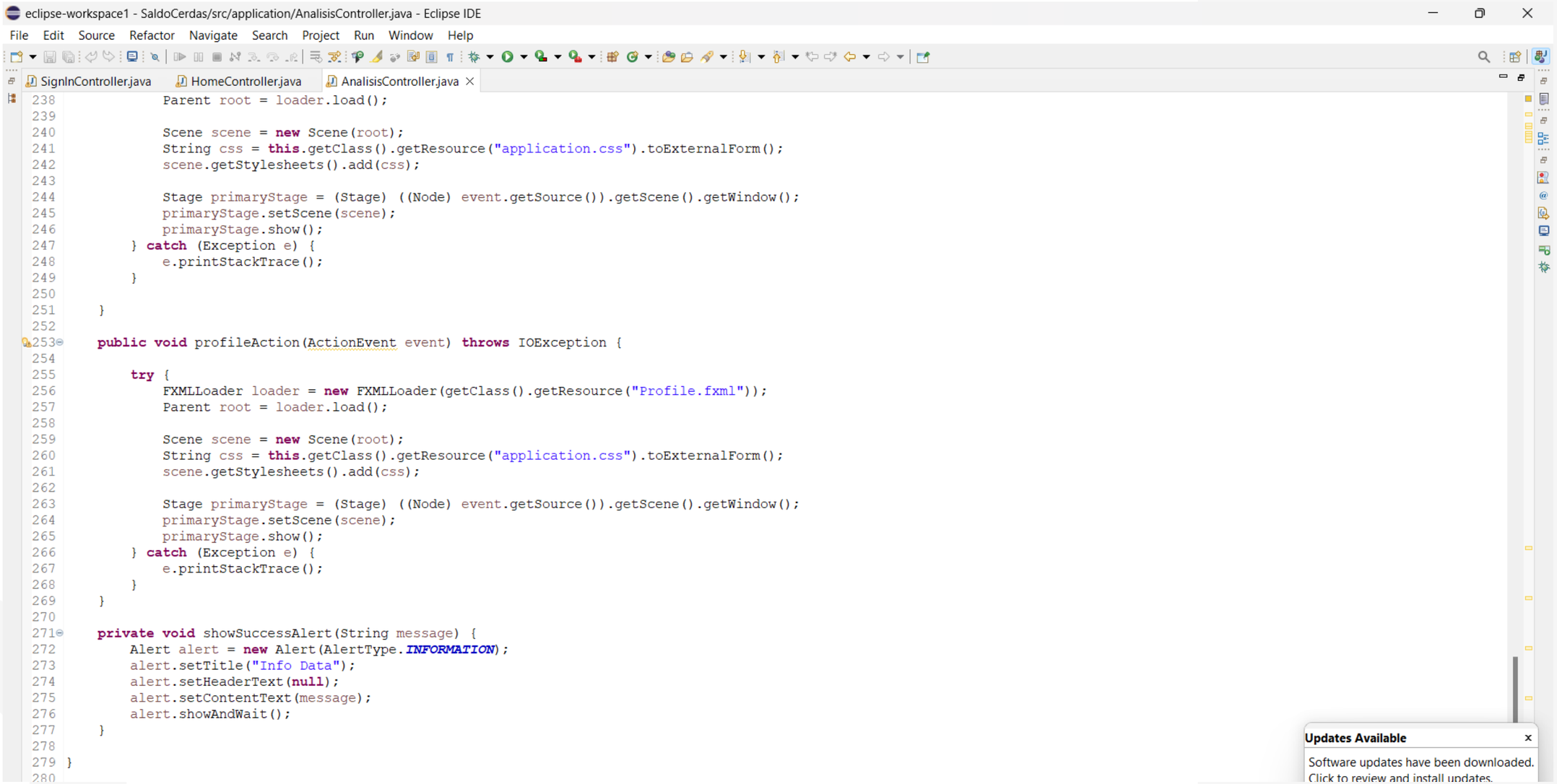


JAVA (ANALISIS PAGE)



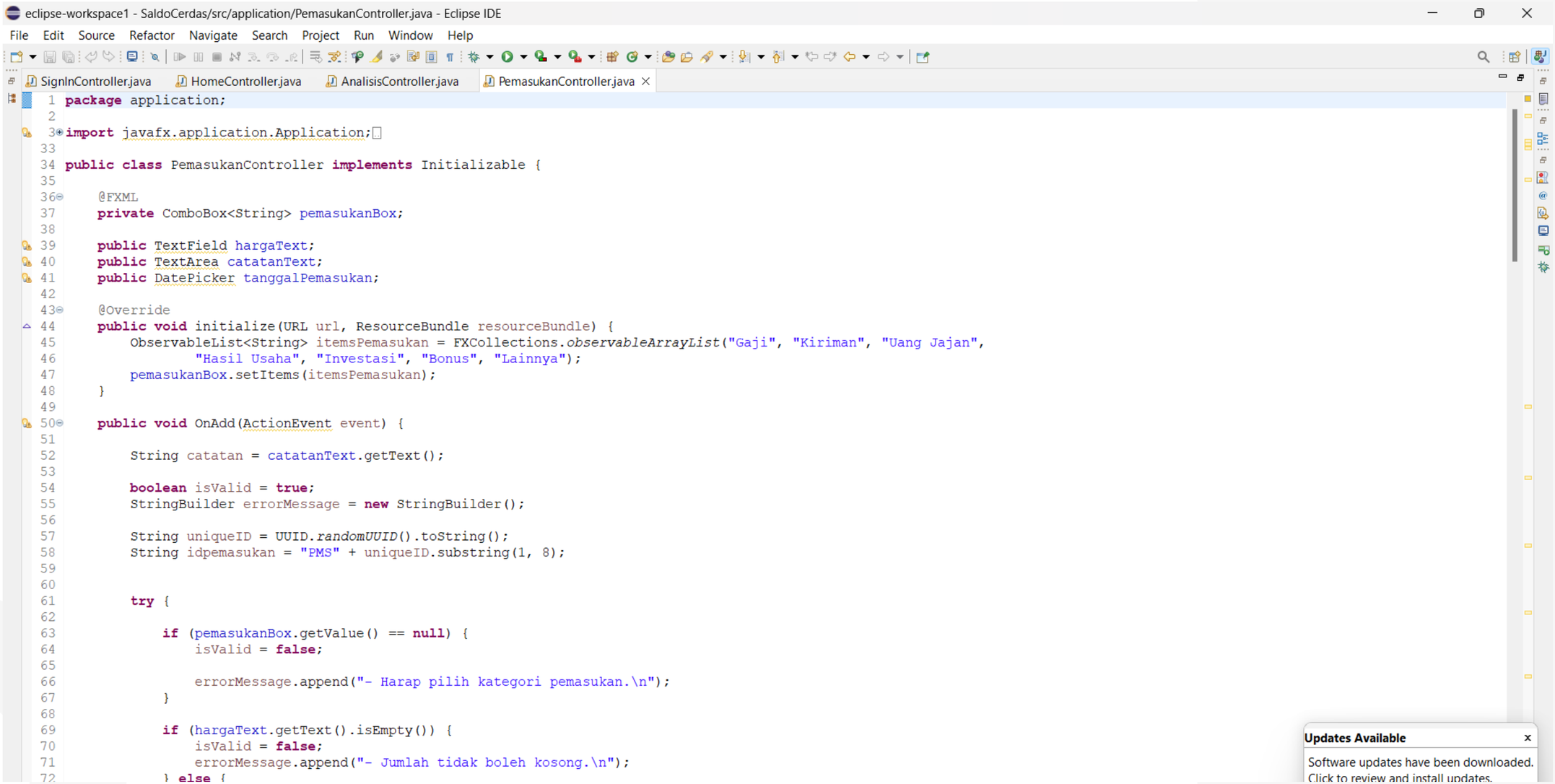

```java
        try {
            FXMLLoader loader = new FXMLLoader(getClass().getResource("Home.fxml"));
            Parent root = loader.load();

            Scene scene = new Scene(root);
            String css = this.getClass().getResource("application.css").toExternalForm();
            scene.getStylesheets().add(css);

            Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
            primaryStage.setScene(scene);
            primaryStage.show();
        } catch (Exception e) {
            e.printStackTrace();
        }

    }

    public void pemasukanAction(ActionEvent event) throws IOException {

        try {
            FXMLLoader loader = new FXMLLoader(getClass().getResource("Pemasukan.fxml"));
            Parent root = loader.load();

            Scene scene = new Scene(root);
            String css = this.getClass().getResource("application.css").toExternalForm();
            scene.getStylesheets().add(css);

            Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
            primaryStage.setScene(scene);
            primaryStage.show();
        } catch (Exception e) {
            e.printStackTrace();
        }

    }

    public void pengeluaranAction(ActionEvent event) throws IOException {

        try {
            FXMLLoader loader = new FXMLLoader(getClass().getResource("Pengeluaran.fxml"));
            Parent root = loader.load();

            Scene scene = new Scene(root);
```

JAVA (ANALISIS PAGE)

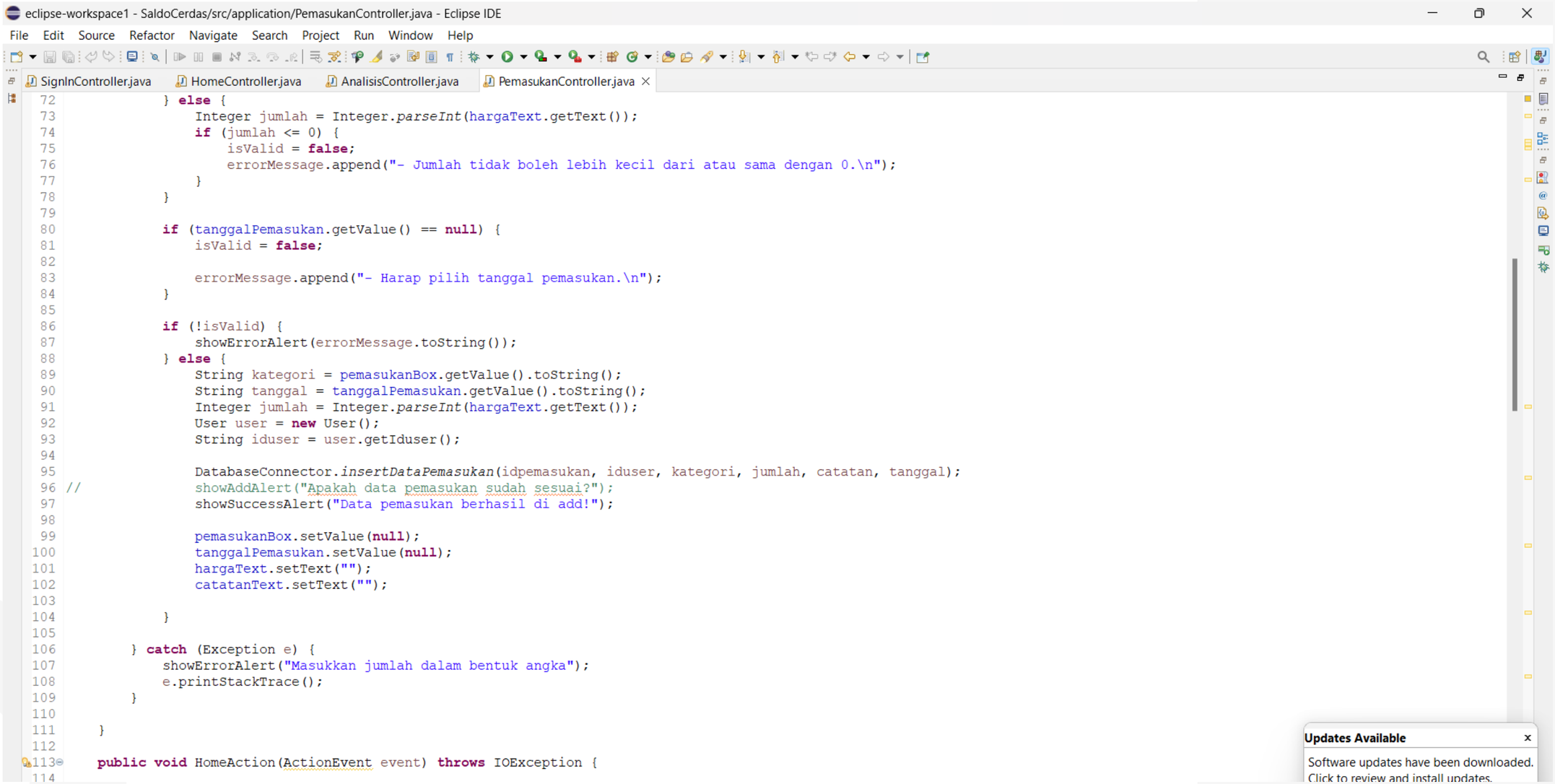



```
            Parent root = loader.load();

            Scene scene = new Scene(root);
            String css = this.getClass().getResource("application.css").toExternalForm();
            scene.getStylesheets().add(css);

            Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
            primaryStage.setScene(scene);
            primaryStage.show();
        } catch (Exception e) {
            e.printStackTrace();
        }

    }

    public void profileAction(ActionEvent event) throws IOException {

        try {
            FXMLLoader loader = new FXMLLoader(getClass().getResource("Profile.fxml"));
            Parent root = loader.load();

            Scene scene = new Scene(root);
            String css = this.getClass().getResource("application.css").toExternalForm();
            scene.getStylesheets().add(css);

            Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
            primaryStage.setScene(scene);
            primaryStage.show();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private void showSuccessAlert(String message) {
        Alert alert = new Alert(AlertType.INFORMATION);
        alert.setTitle("Info Data");
        alert.setHeaderText(null);
        alert.setContentText(message);
        alert.showAndWait();
    }

}
```

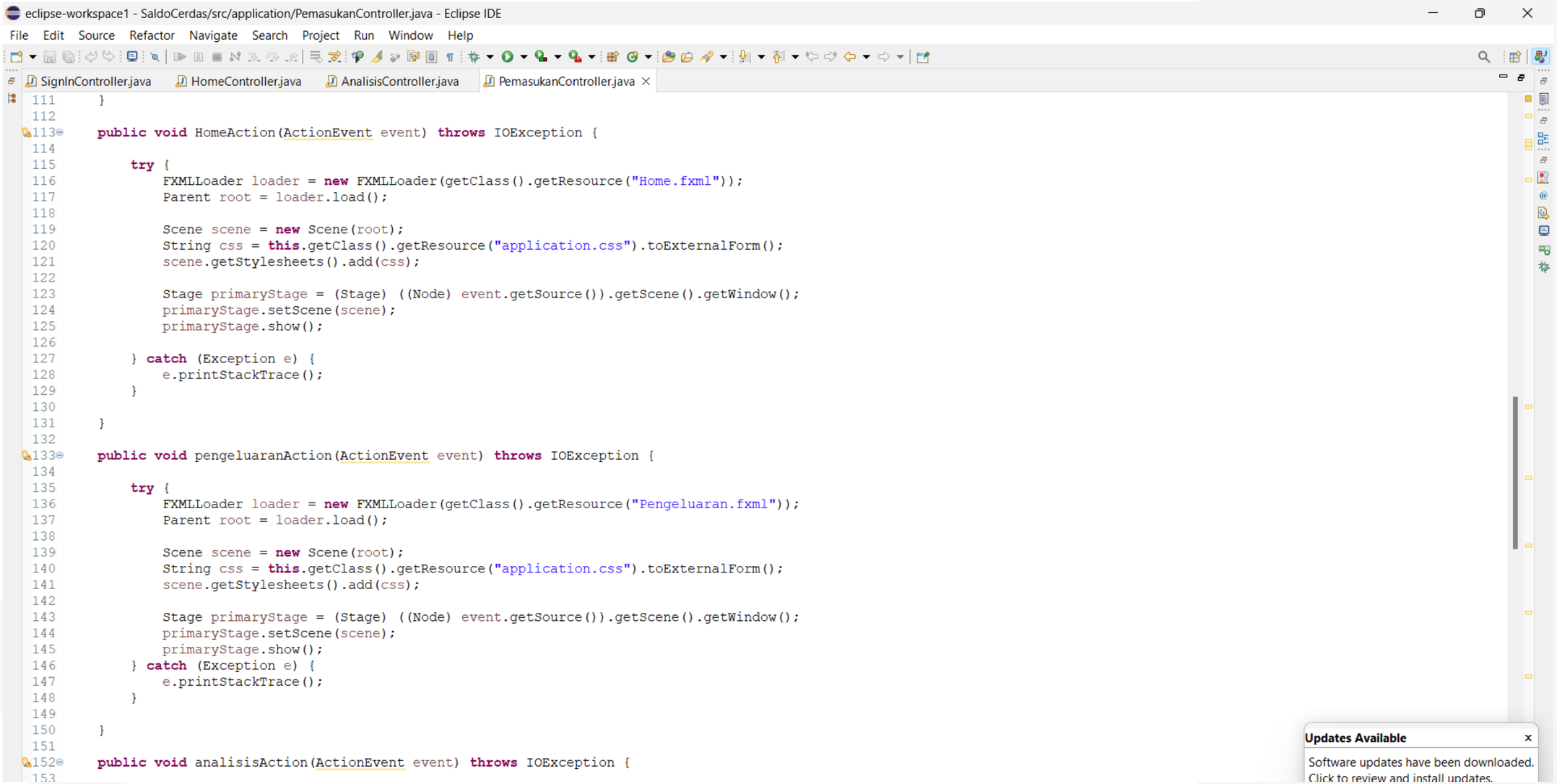
JAVA (PEMASUKAN PAGE)



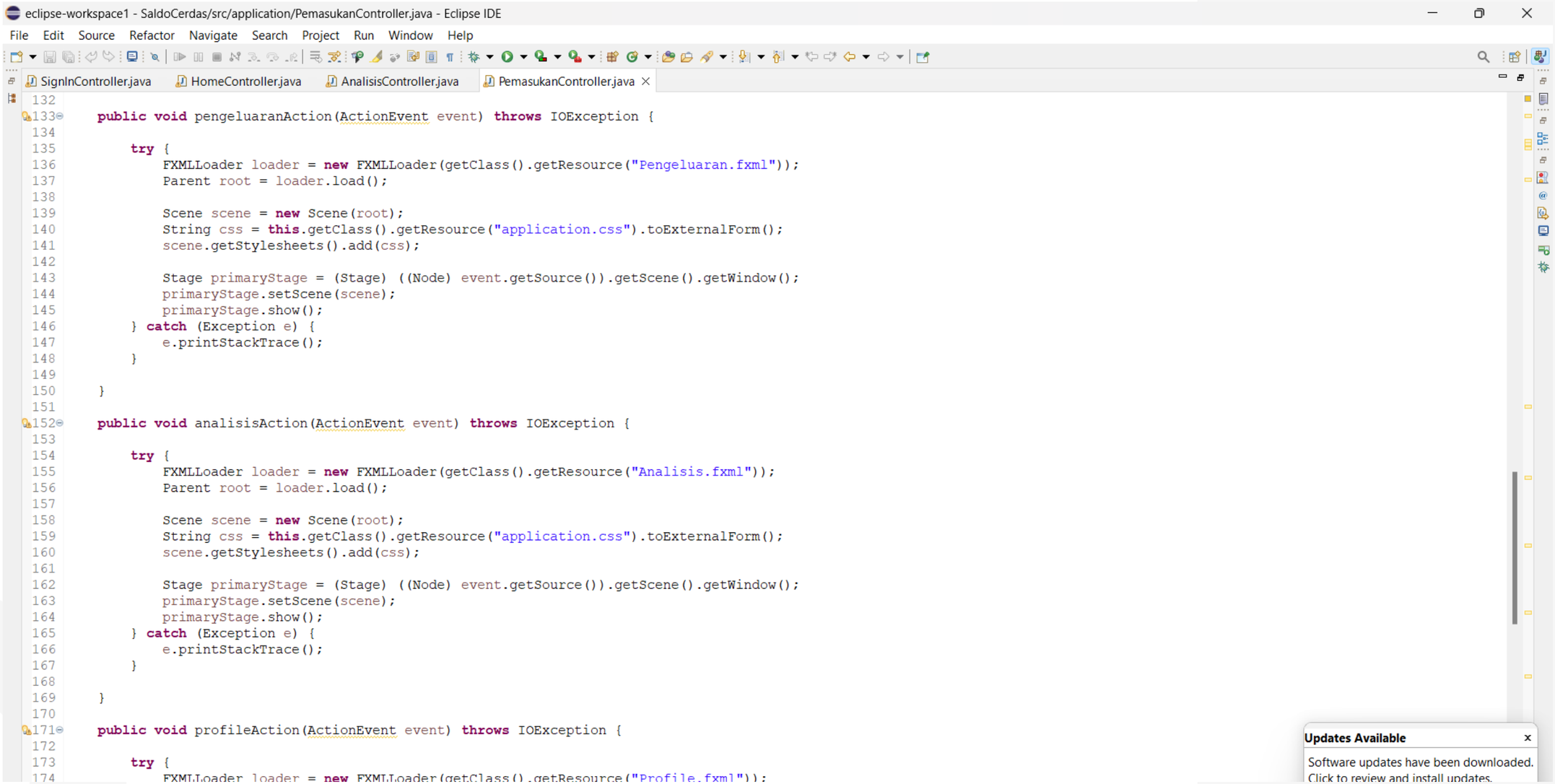
JAVA (PEMASUKAN PAGE)



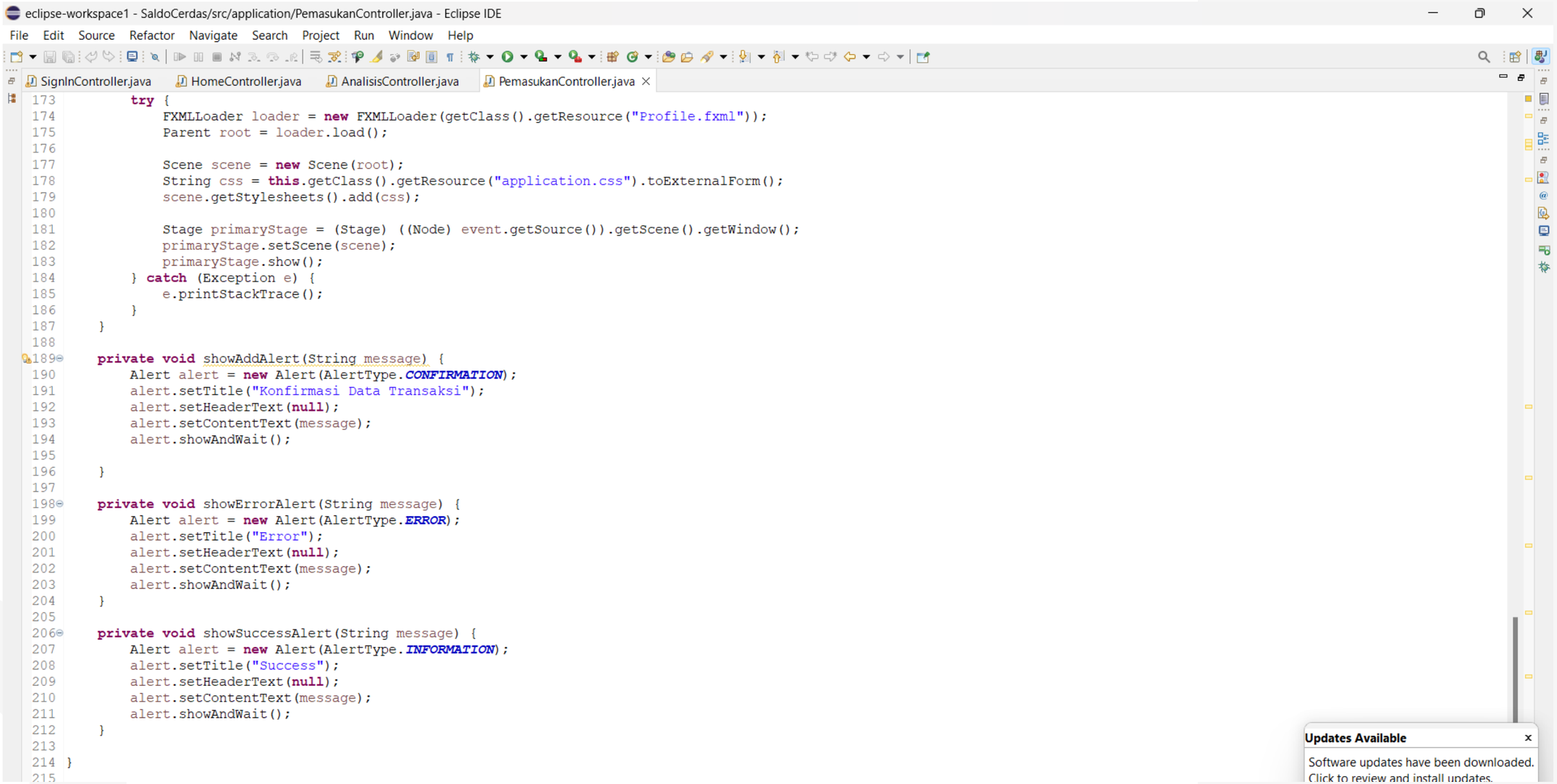
JAVA (PEMASUKAN PAGE)



JAVA (PEMASUKAN PAGE)

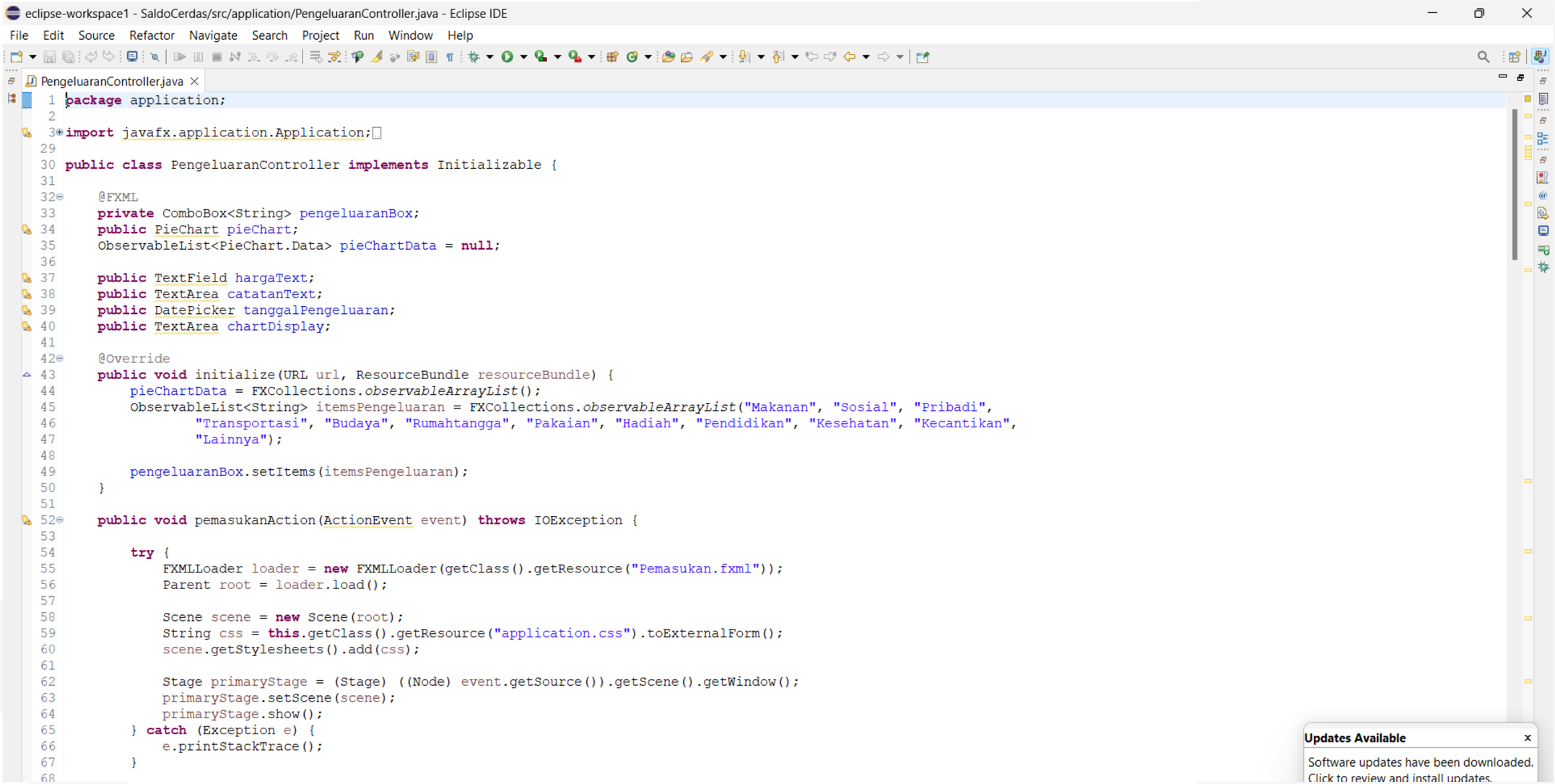


JAVA (PEMASUKAN PAGE)

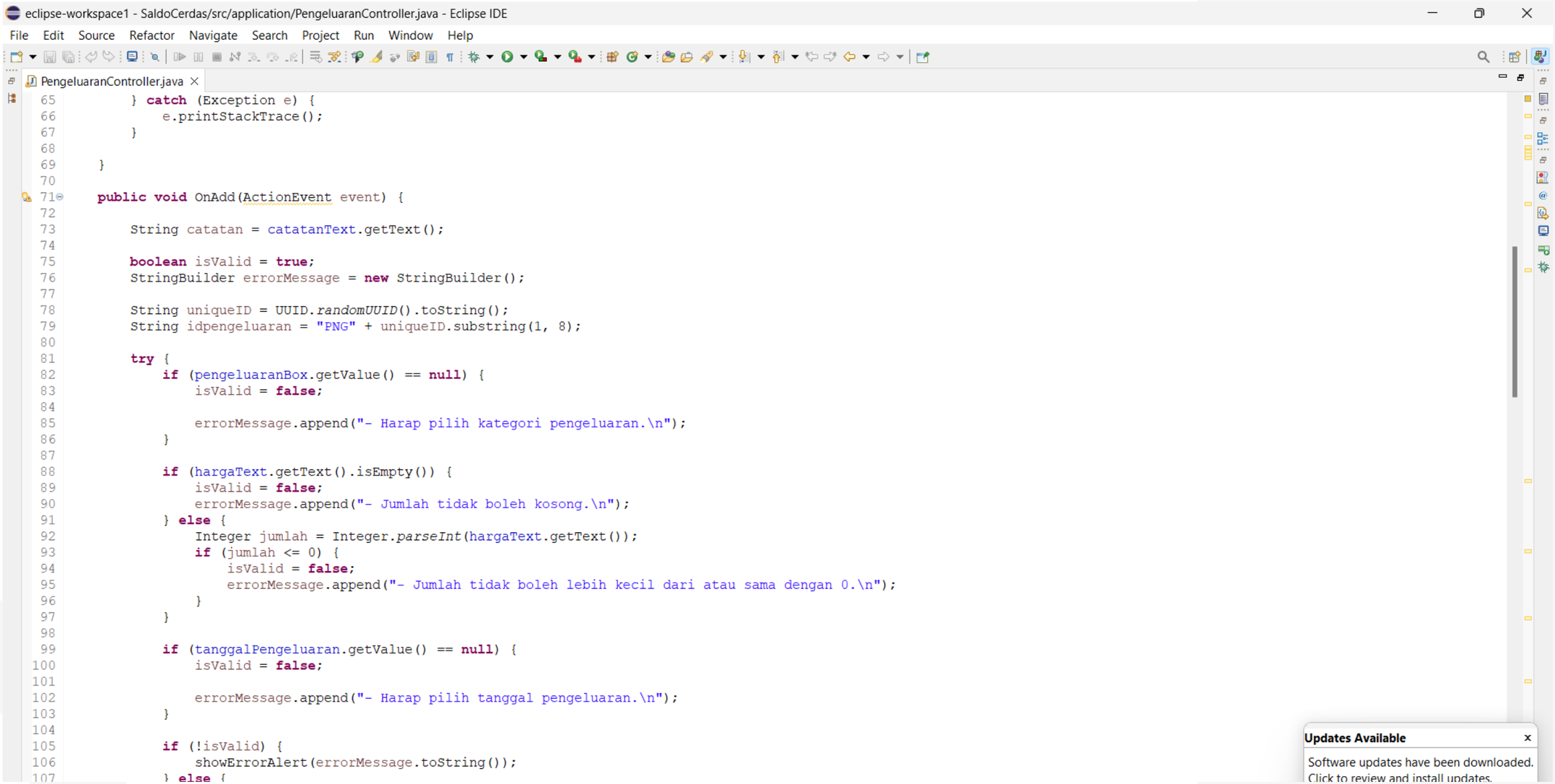



```java
        try {
            FXMLLoader loader = new FXMLLoader(getClass().getResource("Profile.fxml"));
            Parent root = loader.load();

            Scene scene = new Scene(root);
            String css = this.getClass().getResource("application.css").toExternalForm();
            scene.getStylesheets().add(css);

            Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
            primaryStage.setScene(scene);
            primaryStage.show();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private void showAddAlert(String message) {
        Alert alert = new Alert(AlertType.CONFIRMATION);
        alert.setTitle("Konfirmasi Data Transaksi");
        alert.setHeaderText(null);
        alert.setContentText(message);
        alert.showAndWait();

    }

    private void showErrorAlert(String message) {
        Alert alert = new Alert(AlertType.ERROR);
        alert.setTitle("Error");
        alert.setHeaderText(null);
        alert.setContentText(message);
        alert.showAndWait();
    }

    private void showSuccessAlert(String message) {
        Alert alert = new Alert(AlertType.INFORMATION);
        alert.setTitle("Success");
        alert.setHeaderText(null);
        alert.setContentText(message);
        alert.showAndWait();
    }

}
```

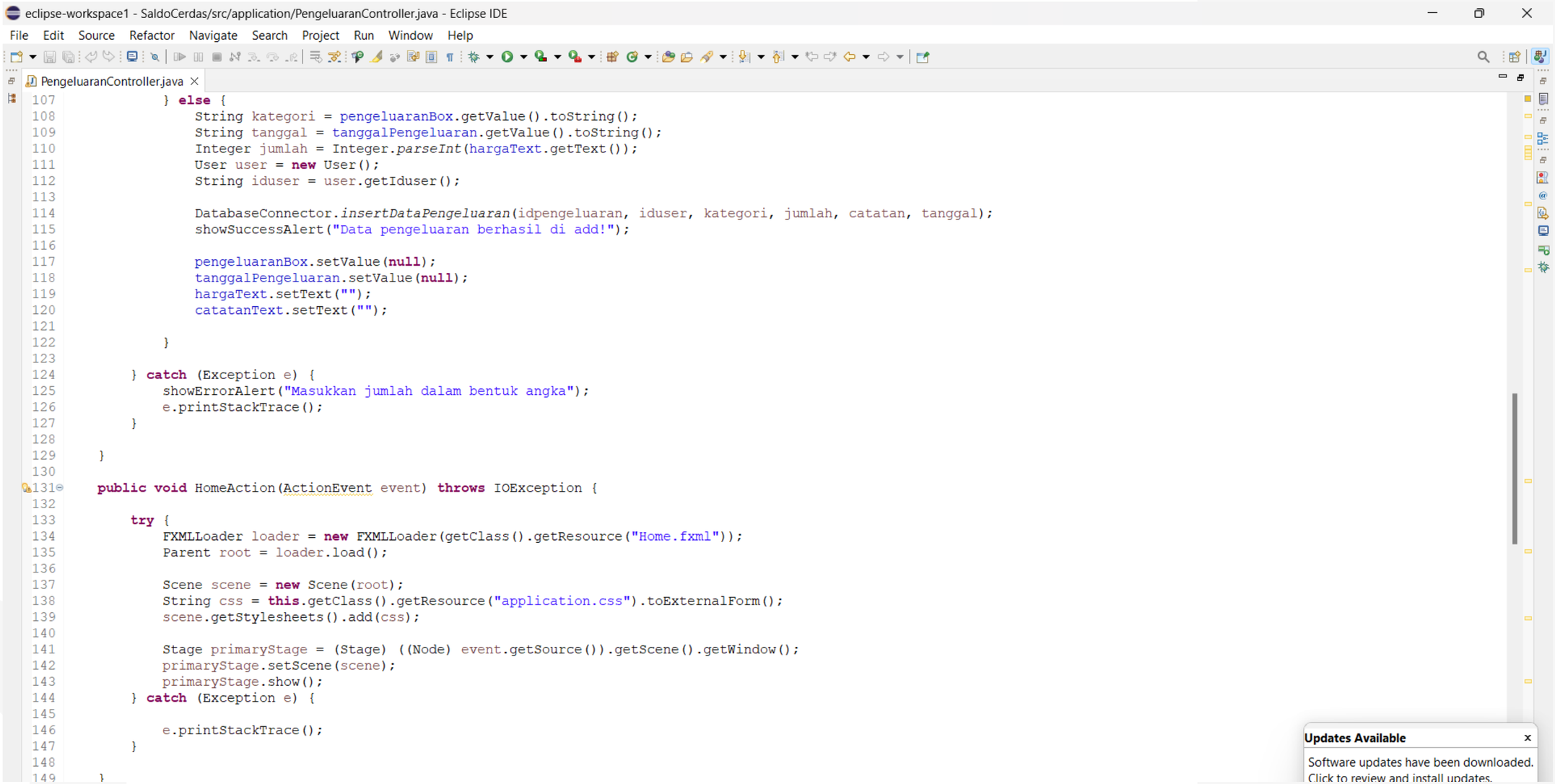
JAVA (PENGELUARAN PAGE)



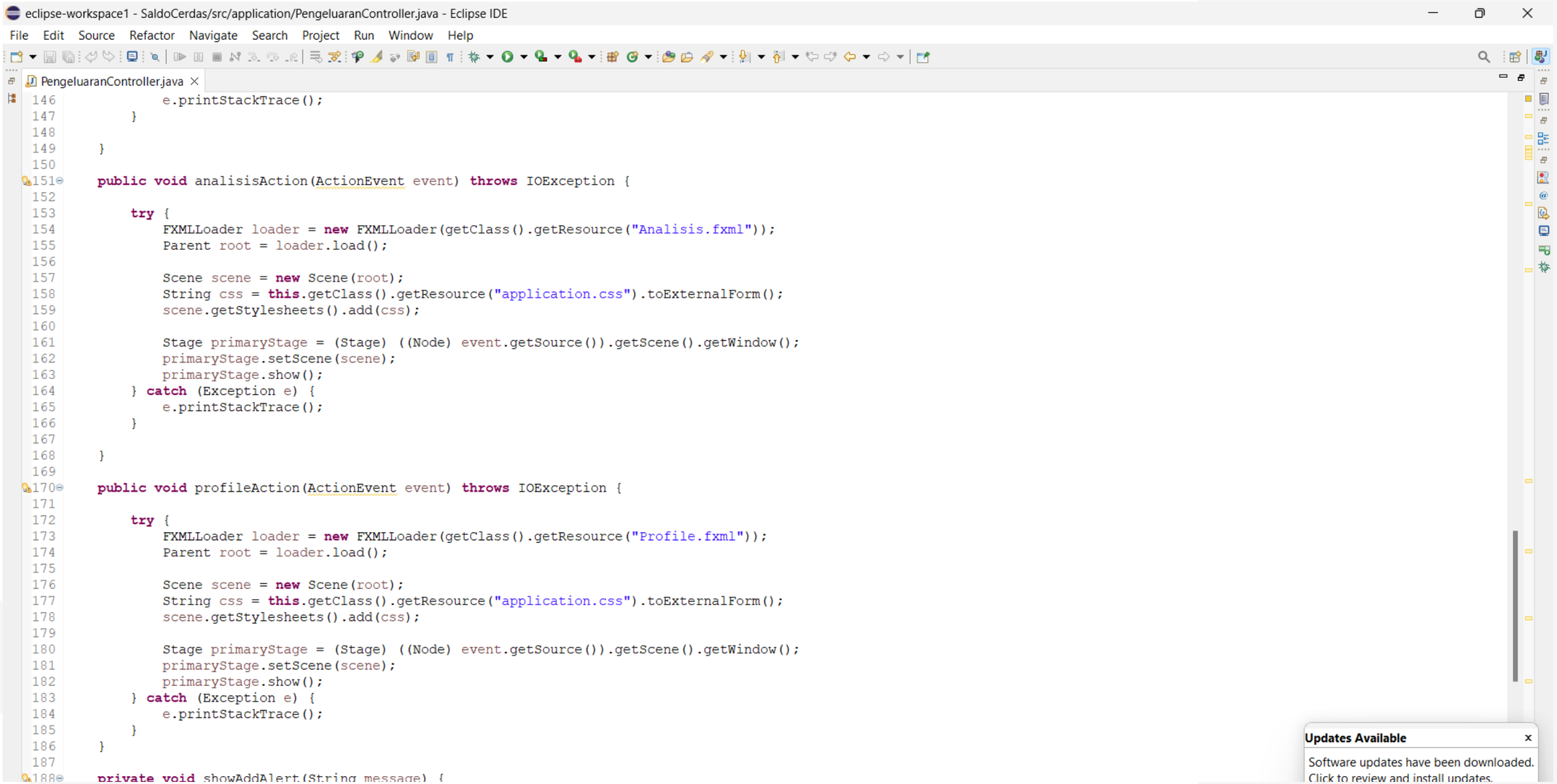
JAVA (PENGELUARAN PAGE)



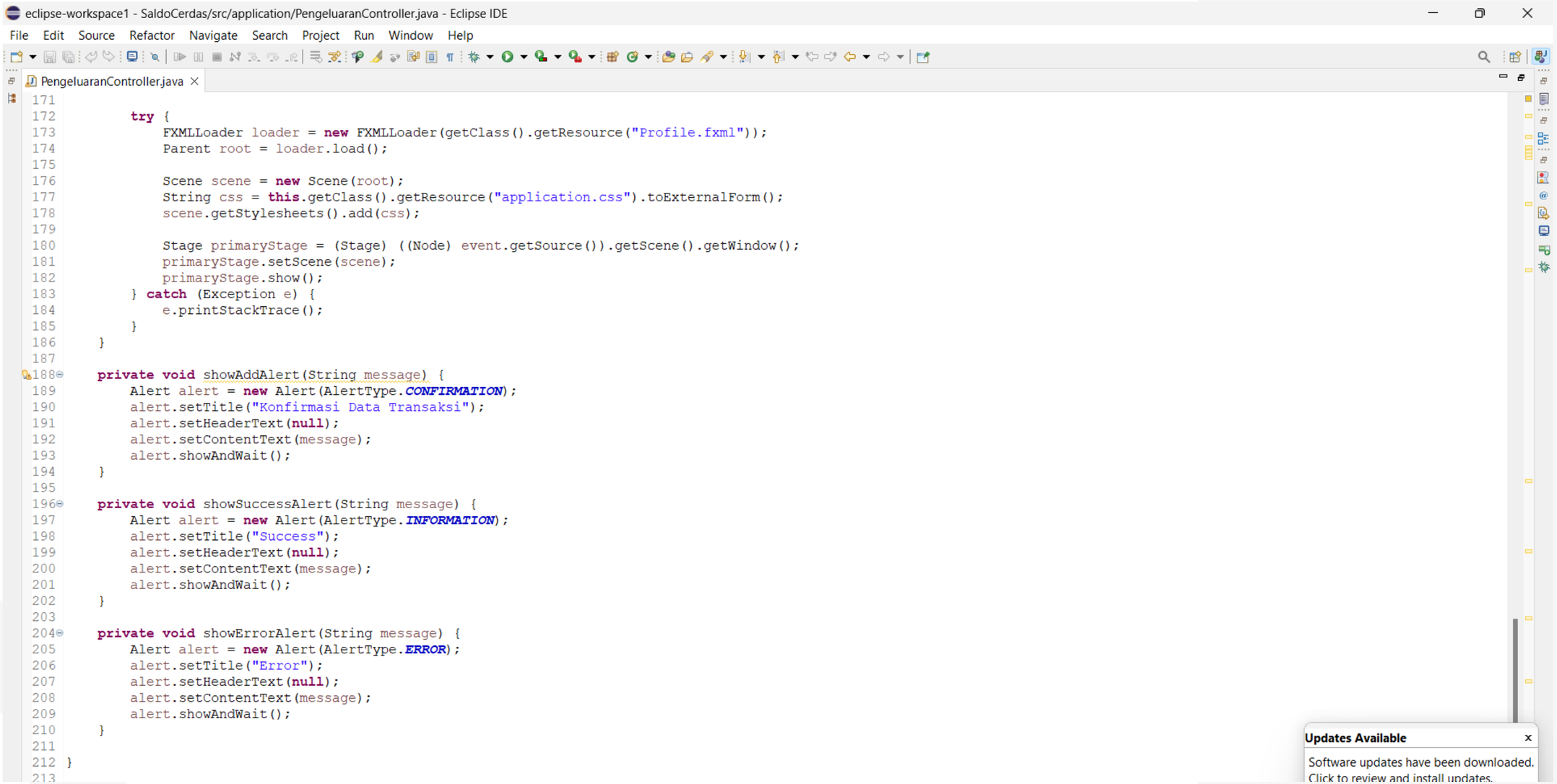
JAVA (PENGELUARAN PAGE)



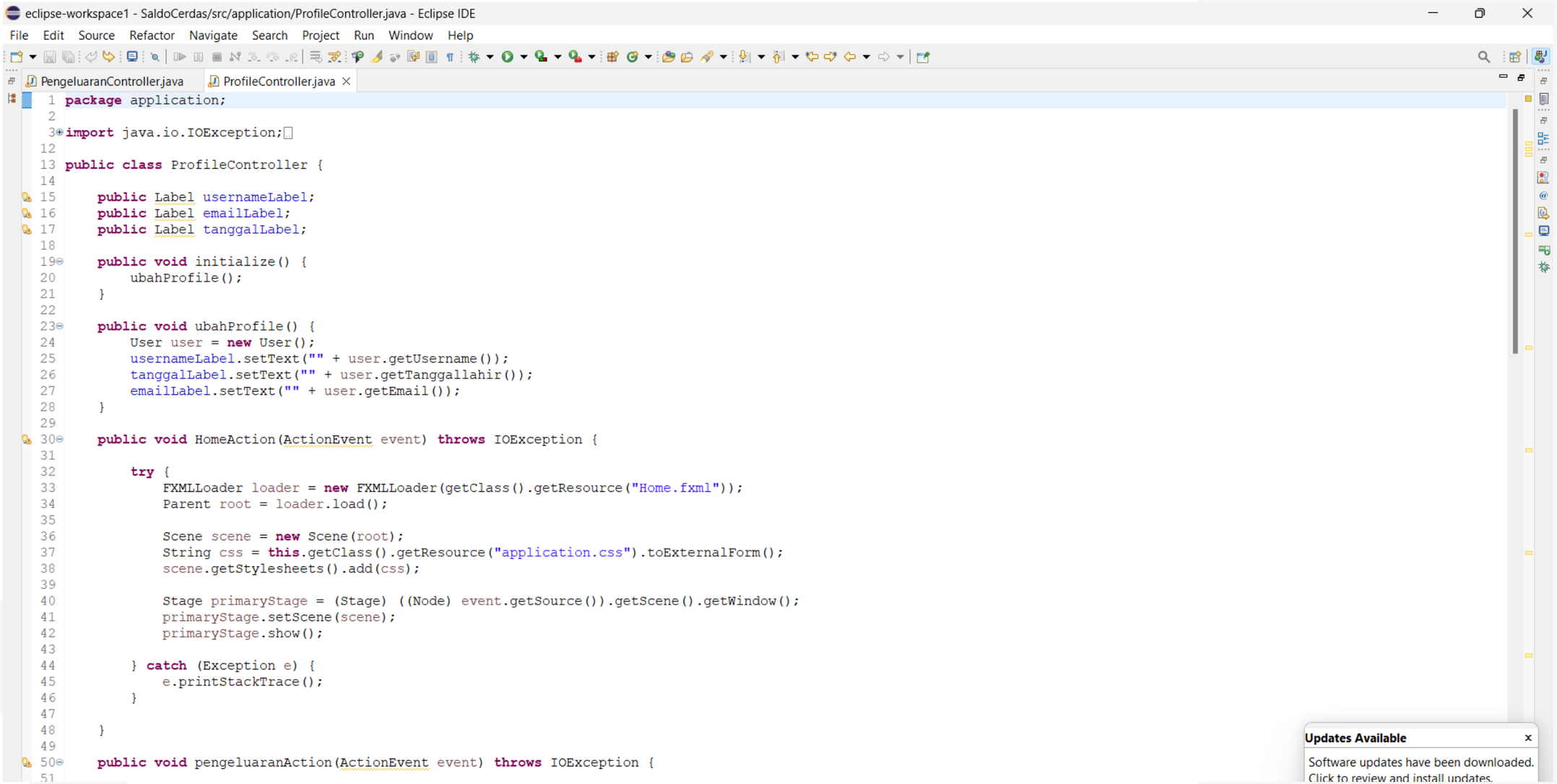
JAVA (PENGELUARAN PAGE)



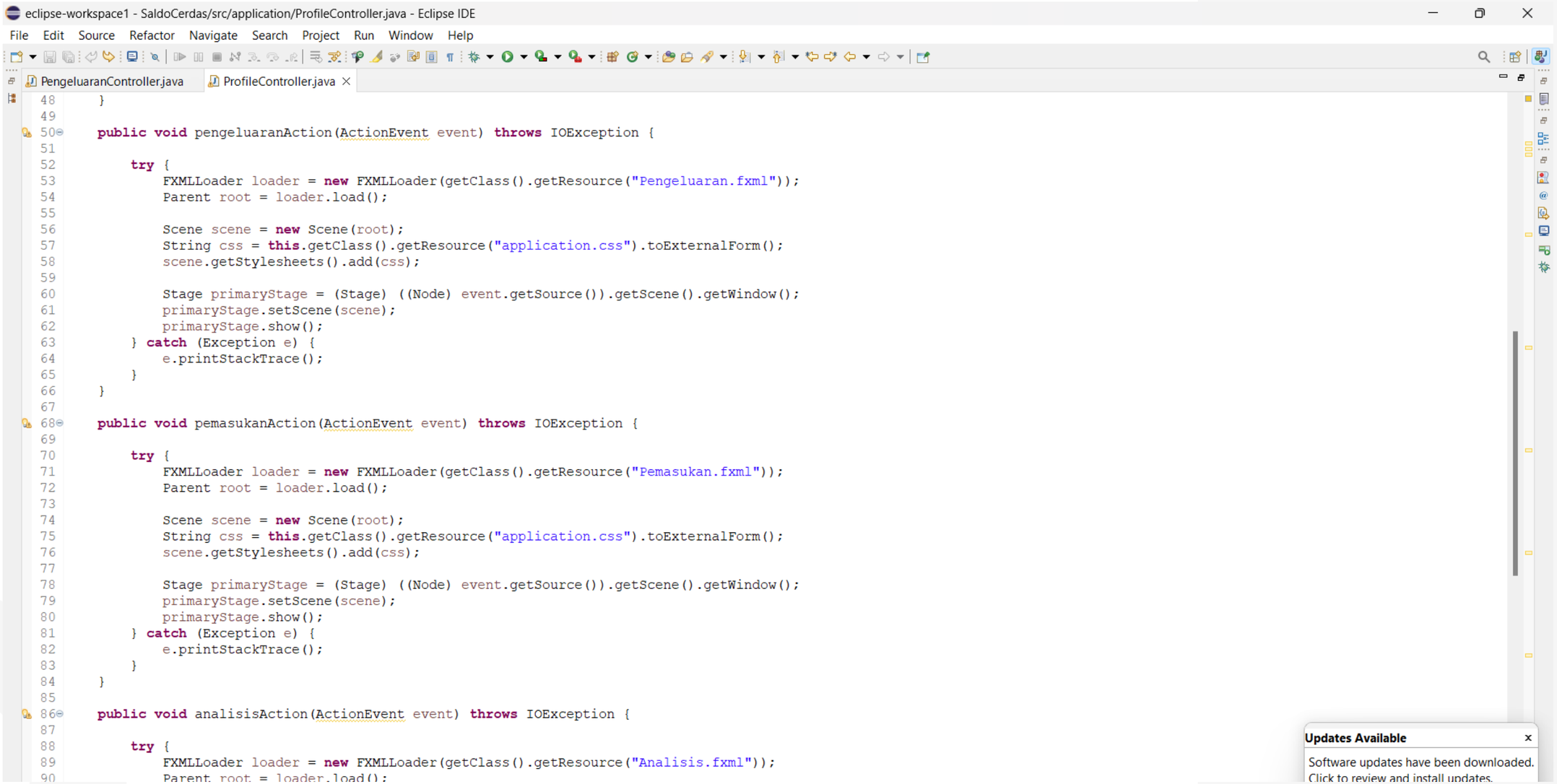
JAVA (PENGELUARAN PAGE)



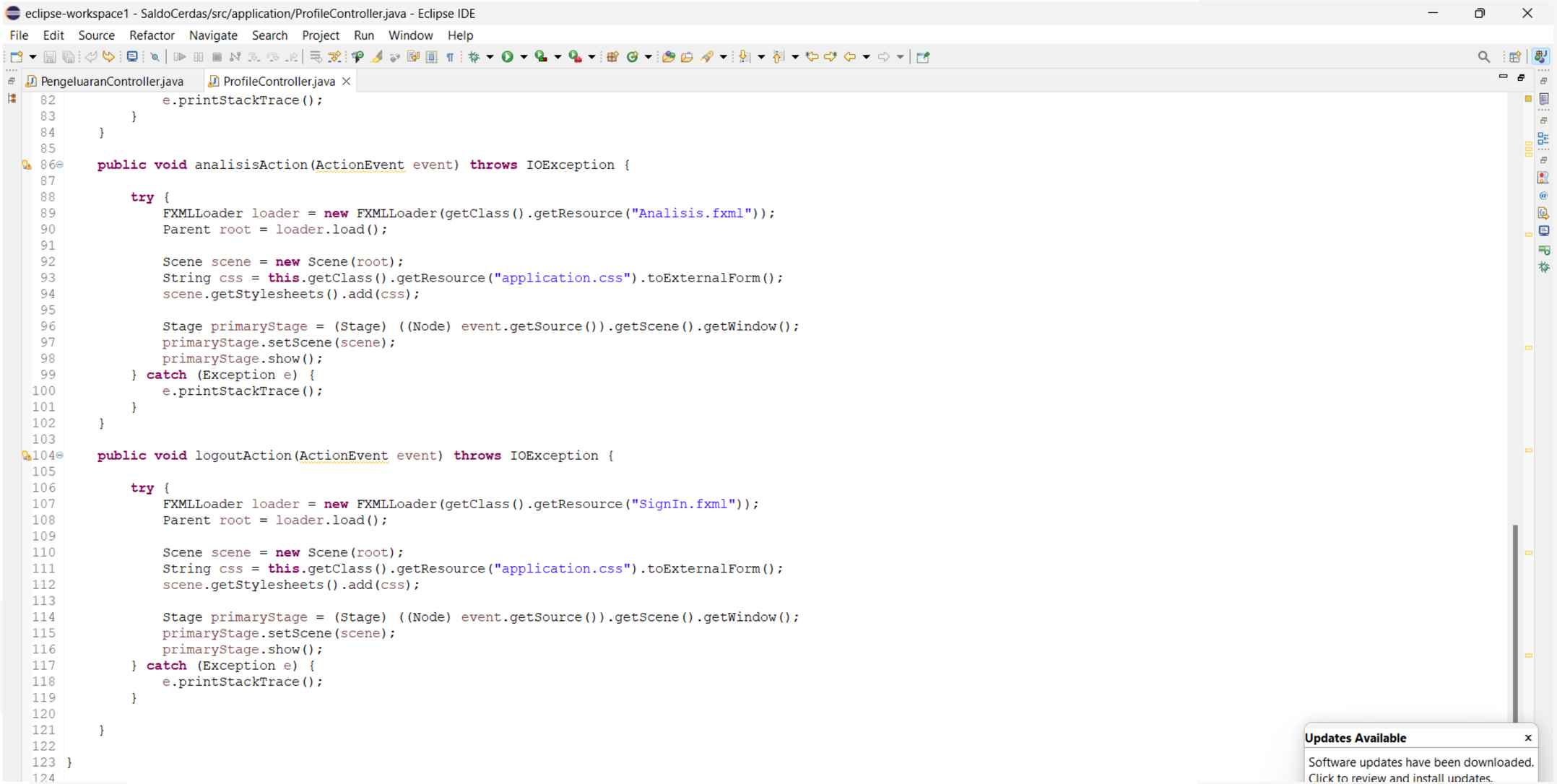
JAVA (PROFILE PAGE)



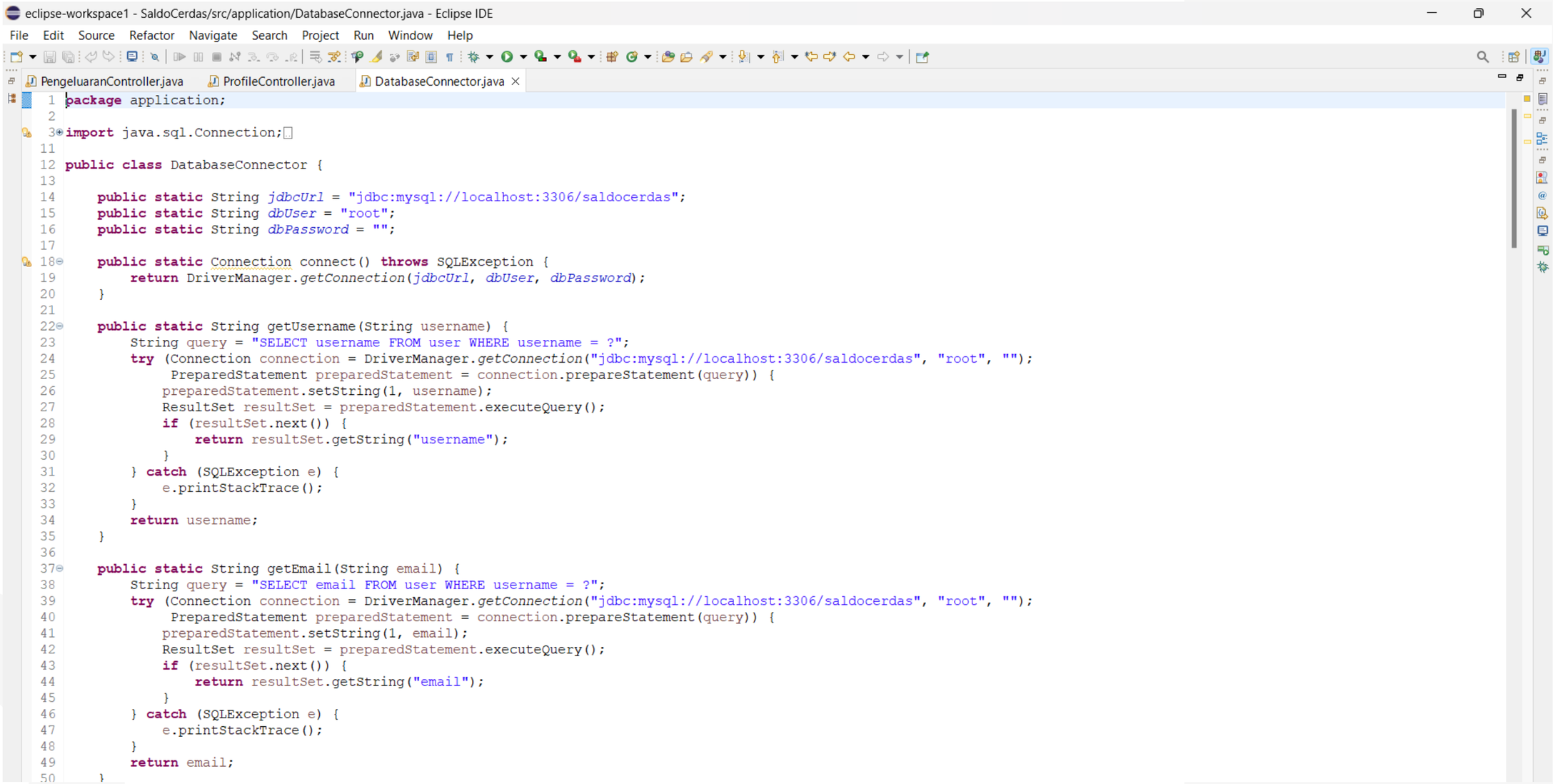
JAVA (PROFILE PAGE)



JAVA (PROFILE PAGE)

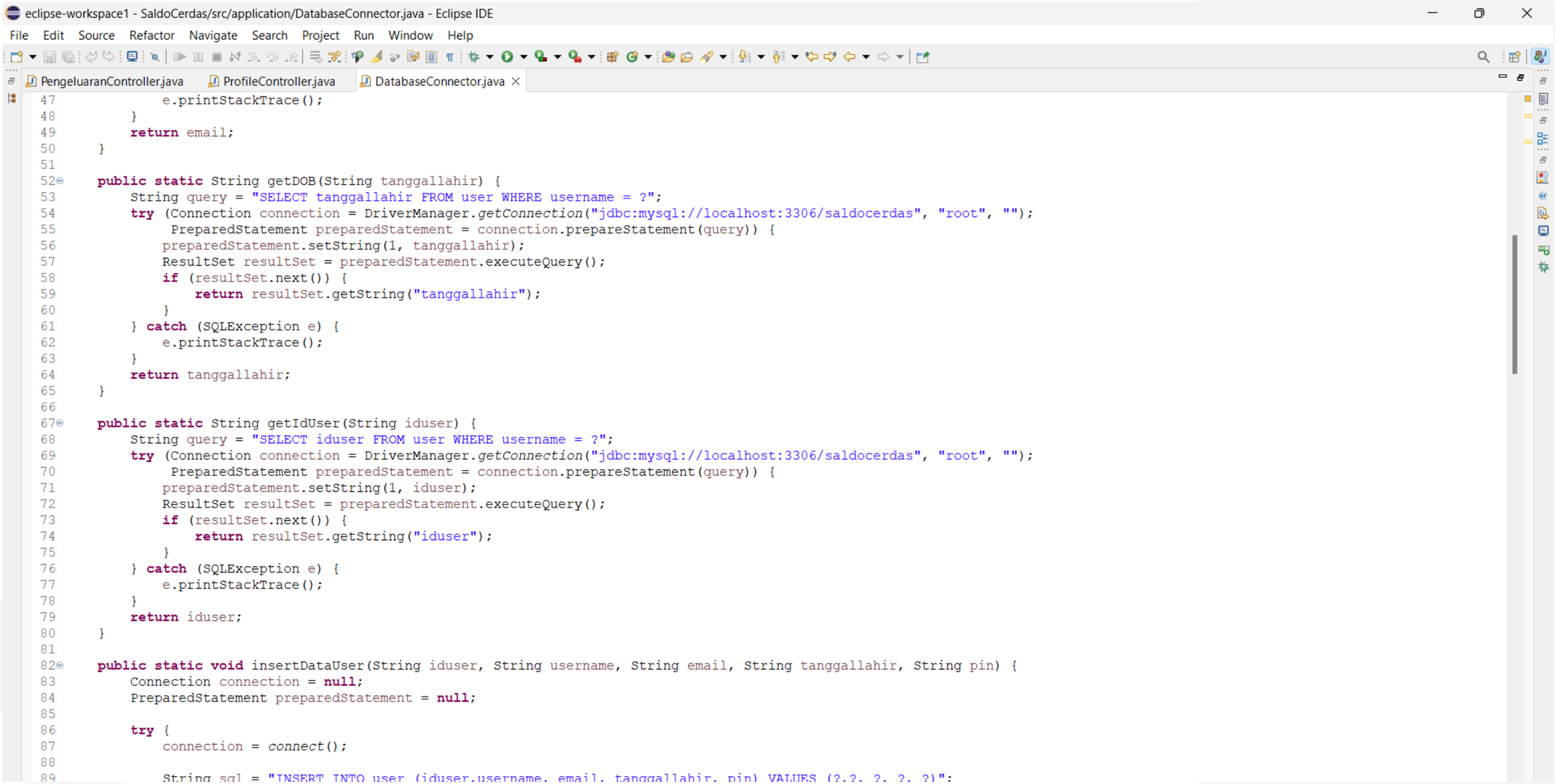


JAVA (DATABASE CONNECTOR)



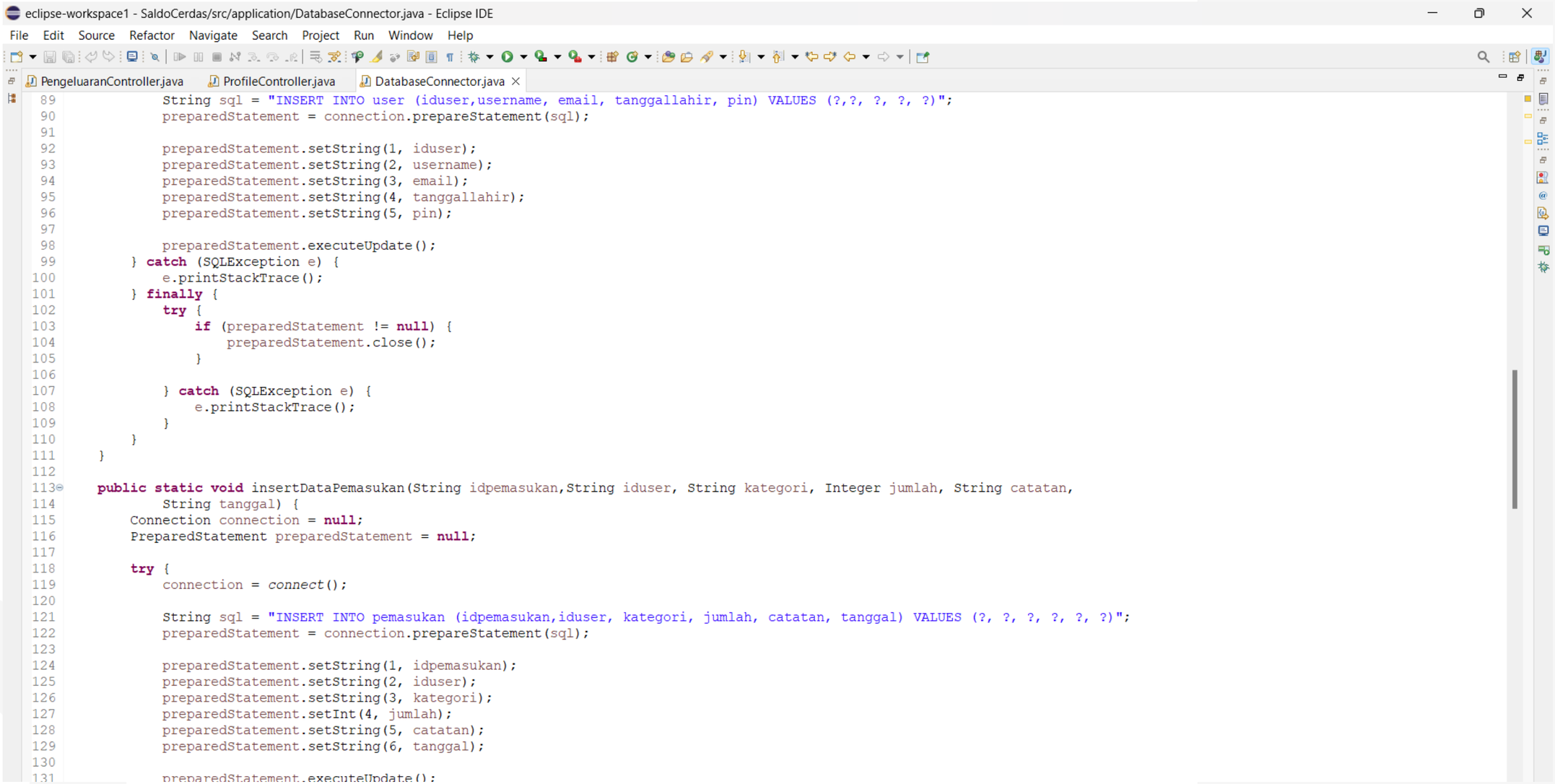

```java
package application;

import java.sql.Connection;

public class DatabaseConnector {

    public static String jdbcUrl = "jdbc:mysql://localhost:3306/saldocerdas";
    public static String dbUser = "root";
    public static String dbPassword = "";

    public static Connection connect() throws SQLException {
        return DriverManager.getConnection(jdbcUrl, dbUser, dbPassword);
    }

    public static String getUsername(String username) {
        String query = "SELECT username FROM user WHERE username = ?";
        try (Connection connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/saldocerdas", "root", "");
             PreparedStatement preparedStatement = connection.prepareStatement(query)) {
            preparedStatement.setString(1, username);
            ResultSet resultSet = preparedStatement.executeQuery();
            if (resultSet.next()) {
                return resultSet.getString("username");
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return username;
    }

    public static String getEmail(String email) {
        String query = "SELECT email FROM user WHERE username = ?";
        try (Connection connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/saldocerdas", "root", "");
             PreparedStatement preparedStatement = connection.prepareStatement(query)) {
            preparedStatement.setString(1, email);
            ResultSet resultSet = preparedStatement.executeQuery();
            if (resultSet.next()) {
                return resultSet.getString("email");
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return email;
    }
```

JAVA (DATABASE CONNECTOR)

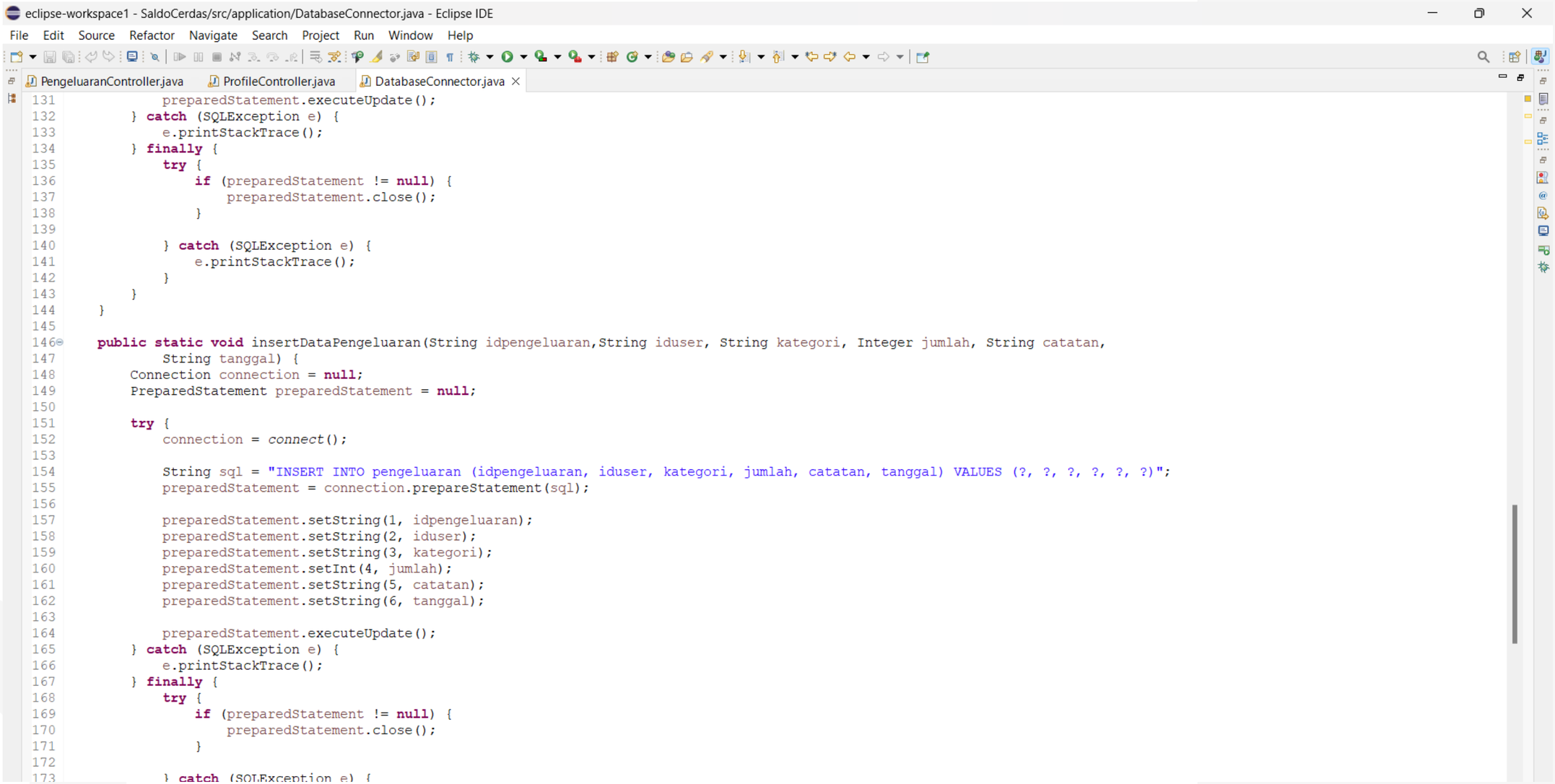



```
47              e.printStackTrace();
48          }
49          return email;
50      }
51
52      public static String getDOB(String tanggallahir) {
53          String query = "SELECT tanggallahir FROM user WHERE username = ?";
54          try (Connection connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/saldocerdas", "root", "");
55               PreparedStatement preparedStatement = connection.prepareStatement(query)) {
56              preparedStatement.setString(1, tanggallahir);
57              ResultSet resultSet = preparedStatement.executeQuery();
58              if (resultSet.next()) {
59                  return resultSet.getString("tanggallahir");
60              }
61          } catch (SQLException e) {
62              e.printStackTrace();
63          }
64          return tanggallahir;
65      }
66
67      public static String getIdUser(String iduser) {
68          String query = "SELECT iduser FROM user WHERE username = ?";
69          try (Connection connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/saldocerdas", "root", "");
70               PreparedStatement preparedStatement = connection.prepareStatement(query)) {
71              preparedStatement.setString(1, iduser);
72              ResultSet resultSet = preparedStatement.executeQuery();
73              if (resultSet.next()) {
74                  return resultSet.getString("iduser");
75              }
76          } catch (SQLException e) {
77              e.printStackTrace();
78          }
79          return iduser;
80      }
81
82      public static void insertDataUser(String iduser, String username, String email, String tanggallahir, String pin) {
83          Connection connection = null;
84          PreparedStatement preparedStatement = null;
85
86          try {
87              connection = connect();
88
89              String sql = "INSERT INTO user (iduser,username, email, tanggallahir, pin) VALUES (?,?, ?, ?, ?)";
```

JAVA (DATABASE CONNECTOR)

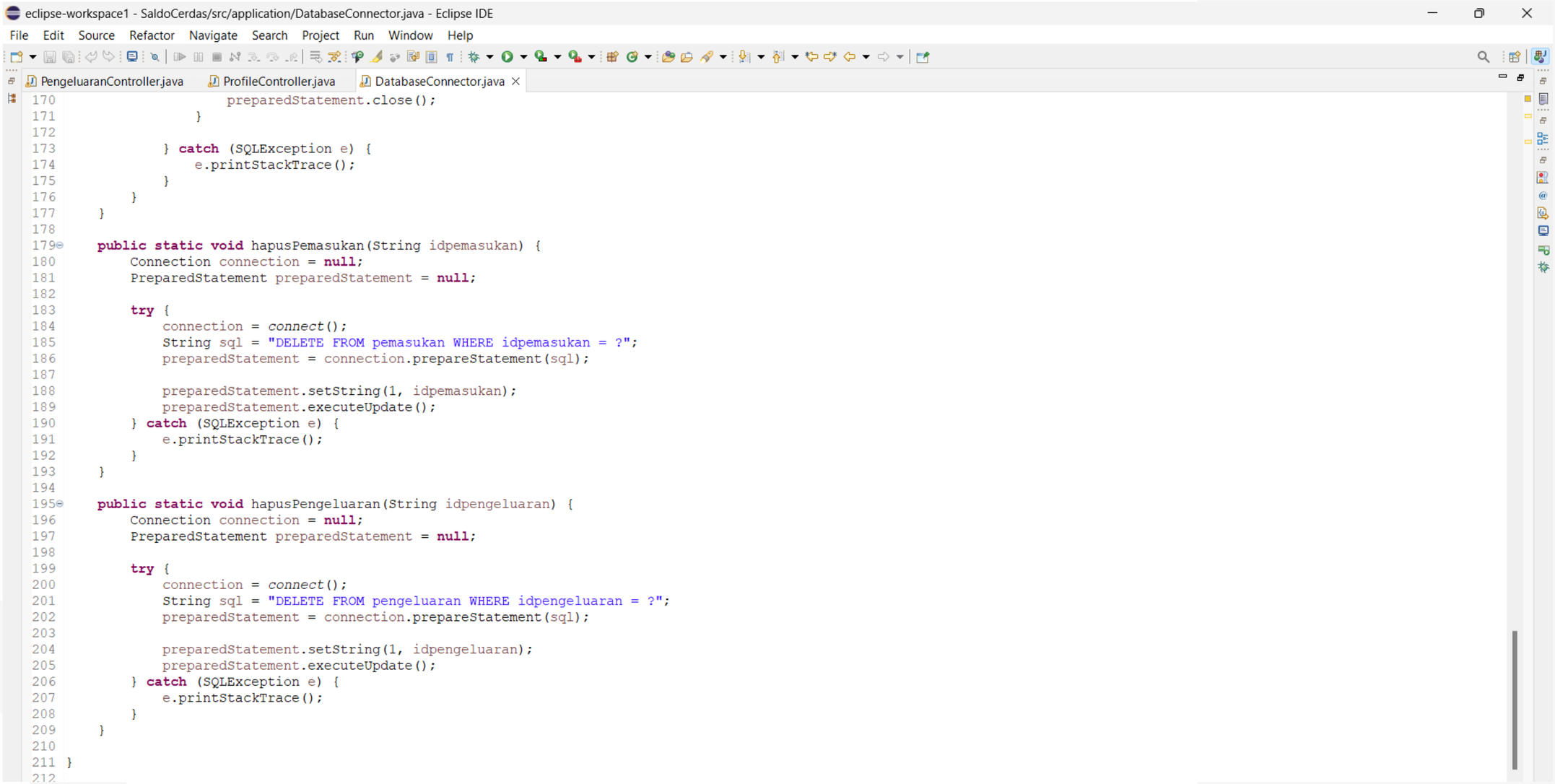


JAVA (DATABASE CONNECTOR)



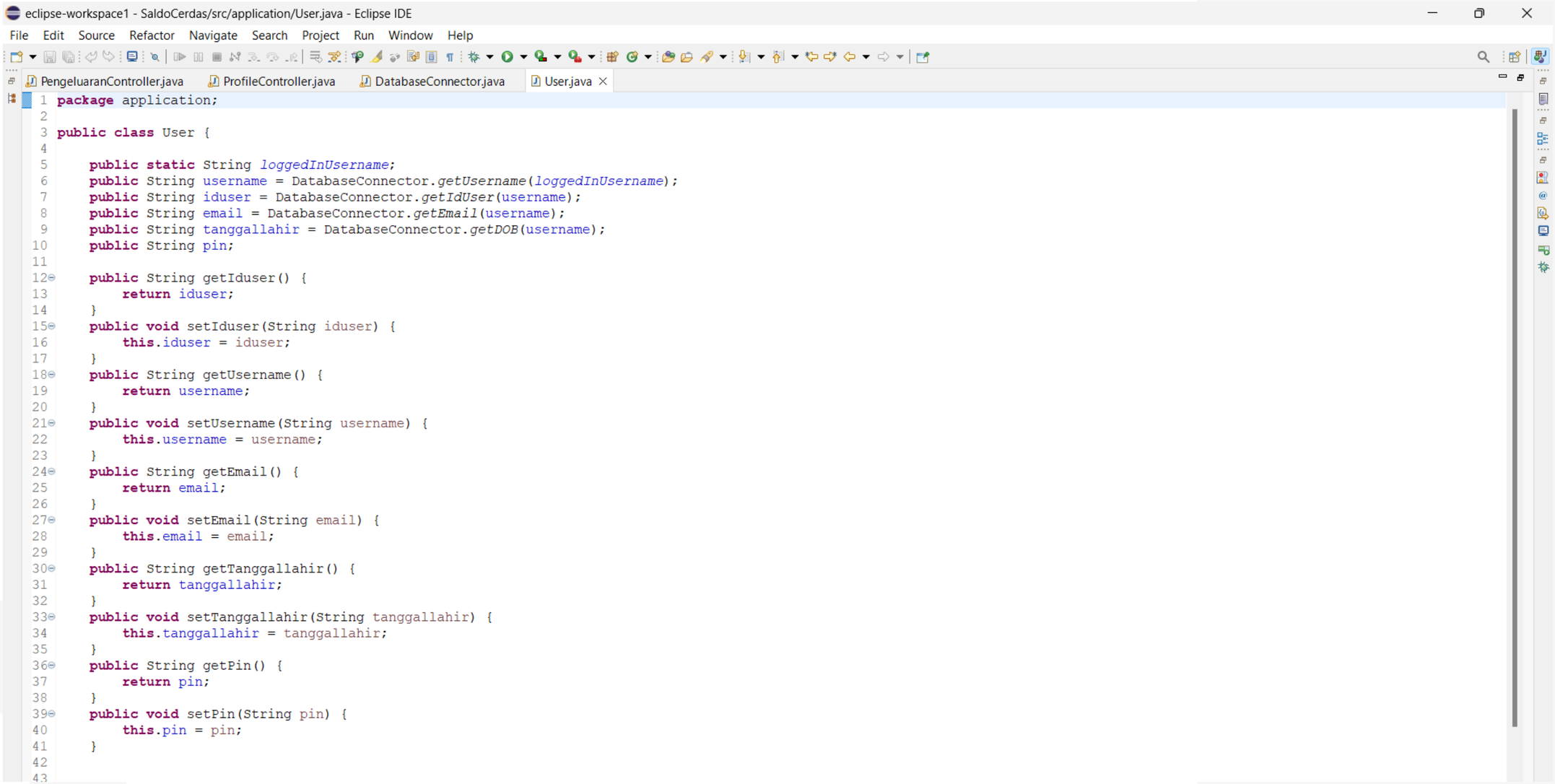

```
            preparedStatement.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        } finally {
            try {
                if (preparedStatement != null) {
                    preparedStatement.close();
                }

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

    public static void insertDataPengeluaran(String idpengeluaran,String iduser, String kategori, Integer jumlah, String catatan,
            String tanggal) {
        Connection connection = null;
        PreparedStatement preparedStatement = null;

        try {
            connection = connect();

            String sql = "INSERT INTO pengeluaran (idpengeluaran, iduser, kategori, jumlah, catatan, tanggal) VALUES (?, ?, ?, ?, ?, ?)";
            preparedStatement = connection.prepareStatement(sql);

            preparedStatement.setString(1, idpengeluaran);
            preparedStatement.setString(2, iduser);
            preparedStatement.setString(3, kategori);
            preparedStatement.setInt(4, jumlah);
            preparedStatement.setString(5, catatan);
            preparedStatement.setString(6, tanggal);

            preparedStatement.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        } finally {
            try {
                if (preparedStatement != null) {
                    preparedStatement.close();
                }

            } catch (SQLException e) {
```

JAVA (DATABASE CONNECTOR)

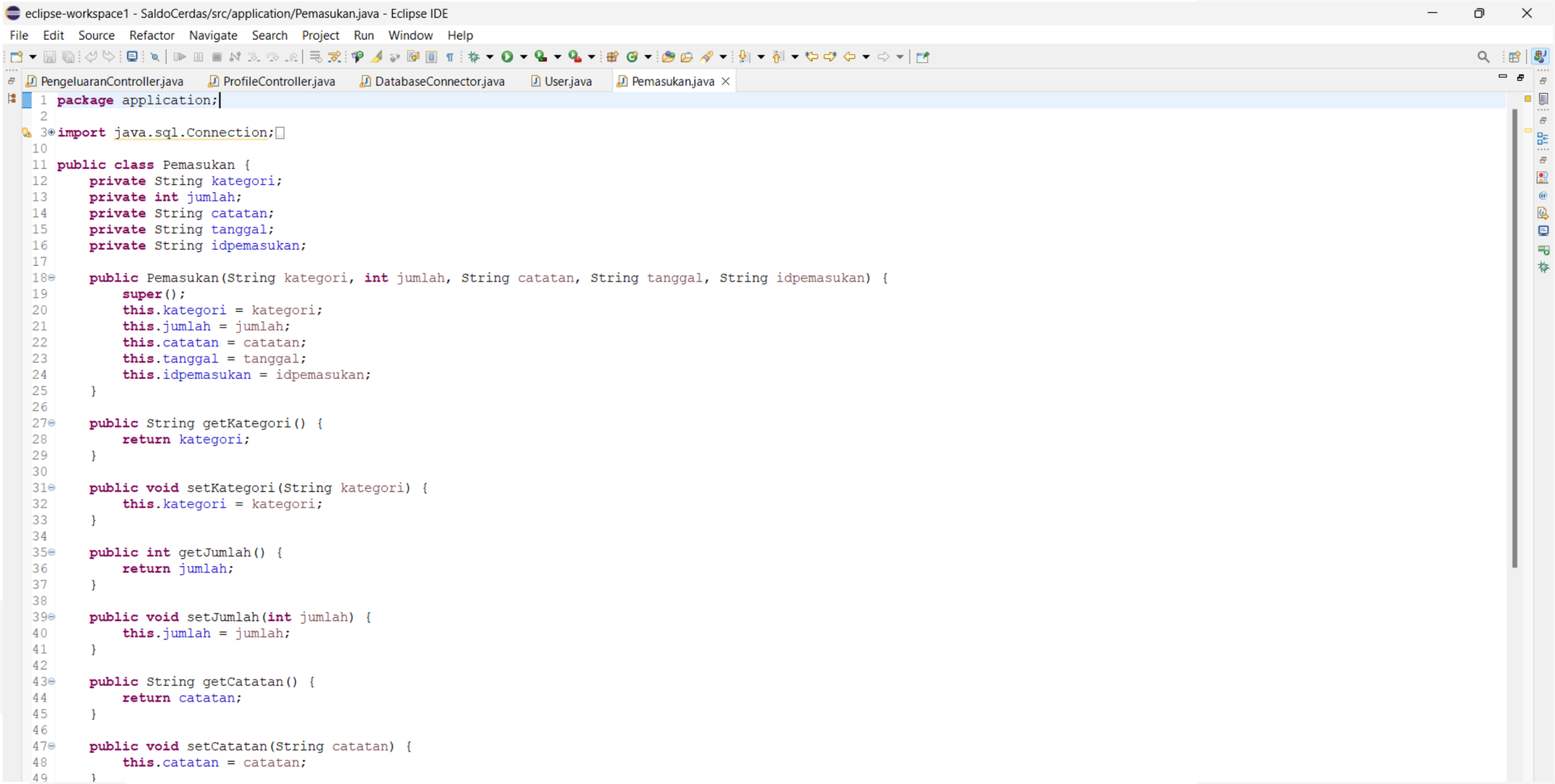



```
                    preparedStatement.close();
                }

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

    public static void hapusPemasukan(String idpemasukan) {
        Connection connection = null;
        PreparedStatement preparedStatement = null;

        try {
            connection = connect();
            String sql = "DELETE FROM pemasukan WHERE idpemasukan = ?";
            preparedStatement = connection.prepareStatement(sql);

            preparedStatement.setString(1, idpemasukan);
            preparedStatement.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public static void hapusPengeluaran(String idpengeluaran) {
        Connection connection = null;
        PreparedStatement preparedStatement = null;

        try {
            connection = connect();
            String sql = "DELETE FROM pengeluaran WHERE idpengeluaran = ?";
            preparedStatement = connection.prepareStatement(sql);

            preparedStatement.setString(1, idpengeluaran);
            preparedStatement.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

}
```

JAVA (USER)

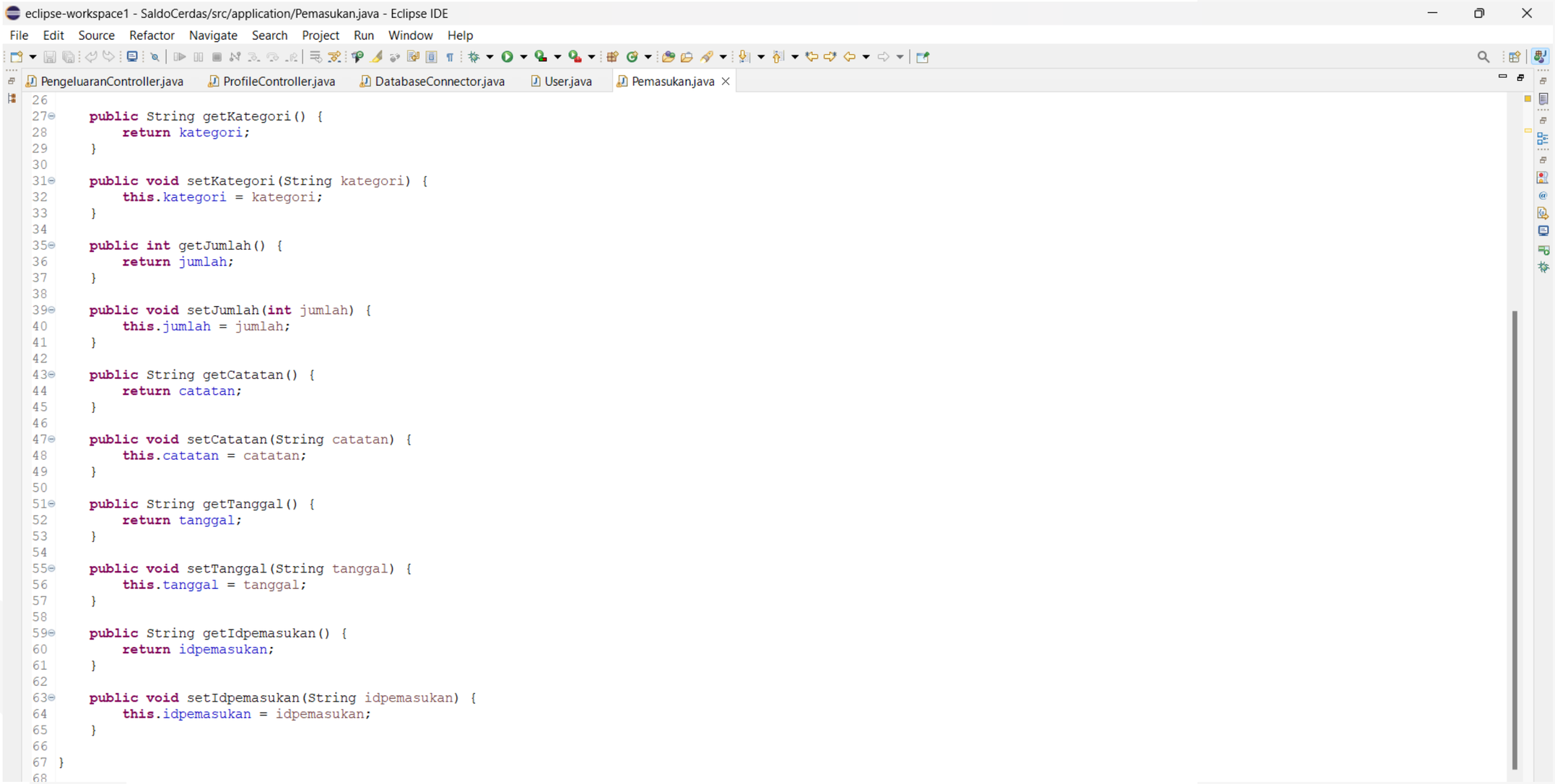

eclipse-workspace1 - SaldoCerdas/src/application/User.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
PengeluaranController.java
ProfileController.java
DatabaseConnector.java
User.java
package application;
public class User {
public static String loggedInUsername;
public String username = DatabaseConnector.getUsername(loggedInUsername);
public String iduser = DatabaseConnector.getIdUser(username);
public String email = DatabaseConnector.getEmail(username);
public String tanggallahir = DatabaseConnector.getDOB(username);
public String pin;
public String getIduser() {
return iduser;
}
public void setIduser(String iduser) {
this.iduser = iduser;
}
public String getUsername() {
return username;
}
public void setUsername(String username) {
this.username = username;
}
public String getEmail() {
return email;
}
public void setEmail(String email) {
this.email = email;
}
public String getTanggallahir() {
return tanggallahir;
}
public void setTanggallahir(String tanggallahir) {
this.tanggallahir = tanggallahir;
}
public String getPin() {
return pin;
}
public void setPin(String pin) {
this.pin = pin;
}

JAVA (PEMASUKAN)

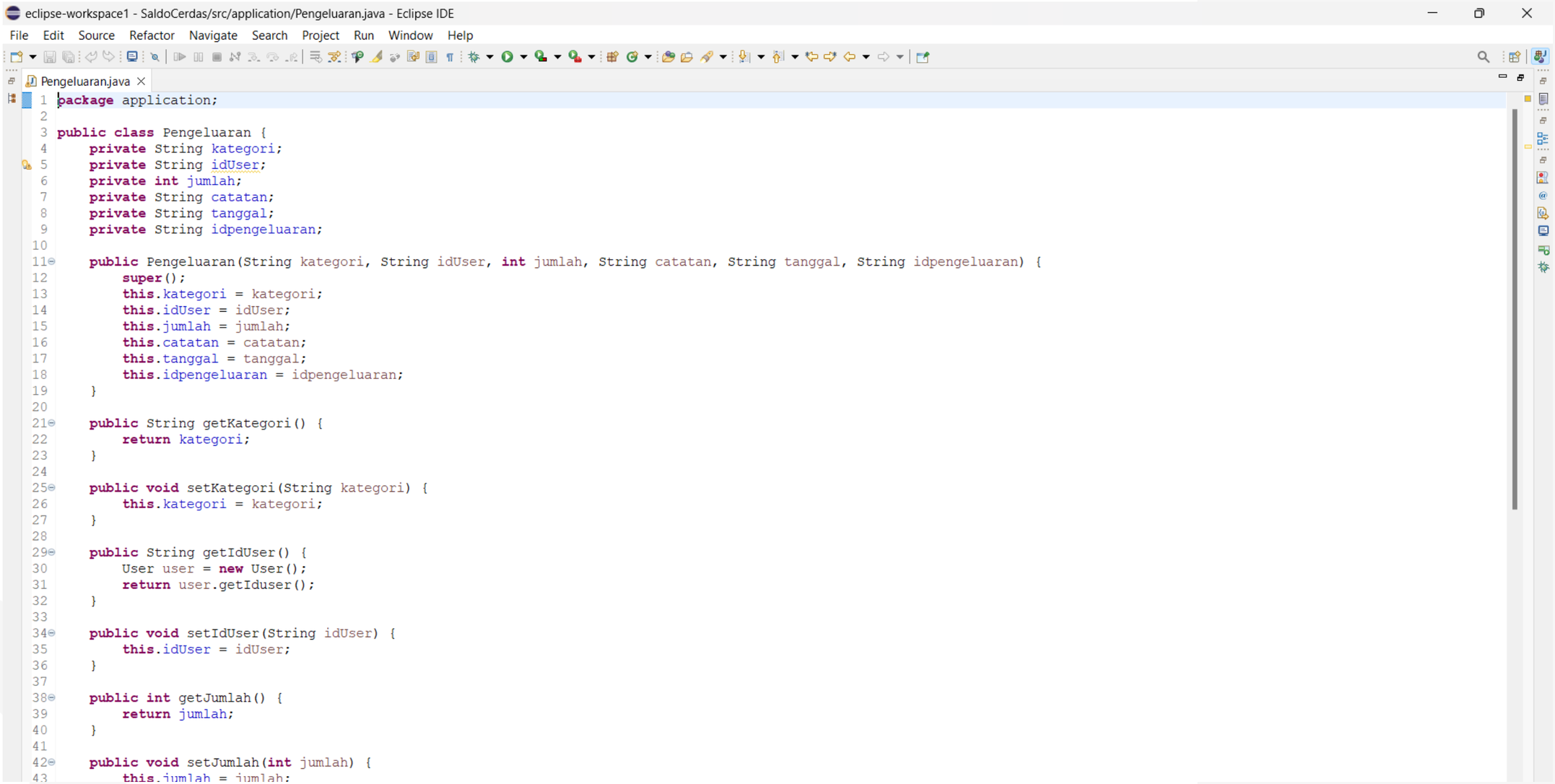



```java
package application;

import java.sql.Connection;

public class Pemasukan {
    private String kategori;
    private int jumlah;
    private String catatan;
    private String tanggal;
    private String idpemasukan;

    public Pemasukan(String kategori, int jumlah, String catatan, String tanggal, String idpemasukan) {
        super();
        this.kategori = kategori;
        this.jumlah = jumlah;
        this.catatan = catatan;
        this.tanggal = tanggal;
        this.idpemasukan = idpemasukan;
    }

    public String getKategori() {
        return kategori;
    }

    public void setKategori(String kategori) {
        this.kategori = kategori;
    }

    public int getJumlah() {
        return jumlah;
    }

    public void setJumlah(int jumlah) {
        this.jumlah = jumlah;
    }

    public String getCatatan() {
        return catatan;
    }

    public void setCatatan(String catatan) {
        this.catatan = catatan;
    }
```

JAVA (PEMASUKAN)

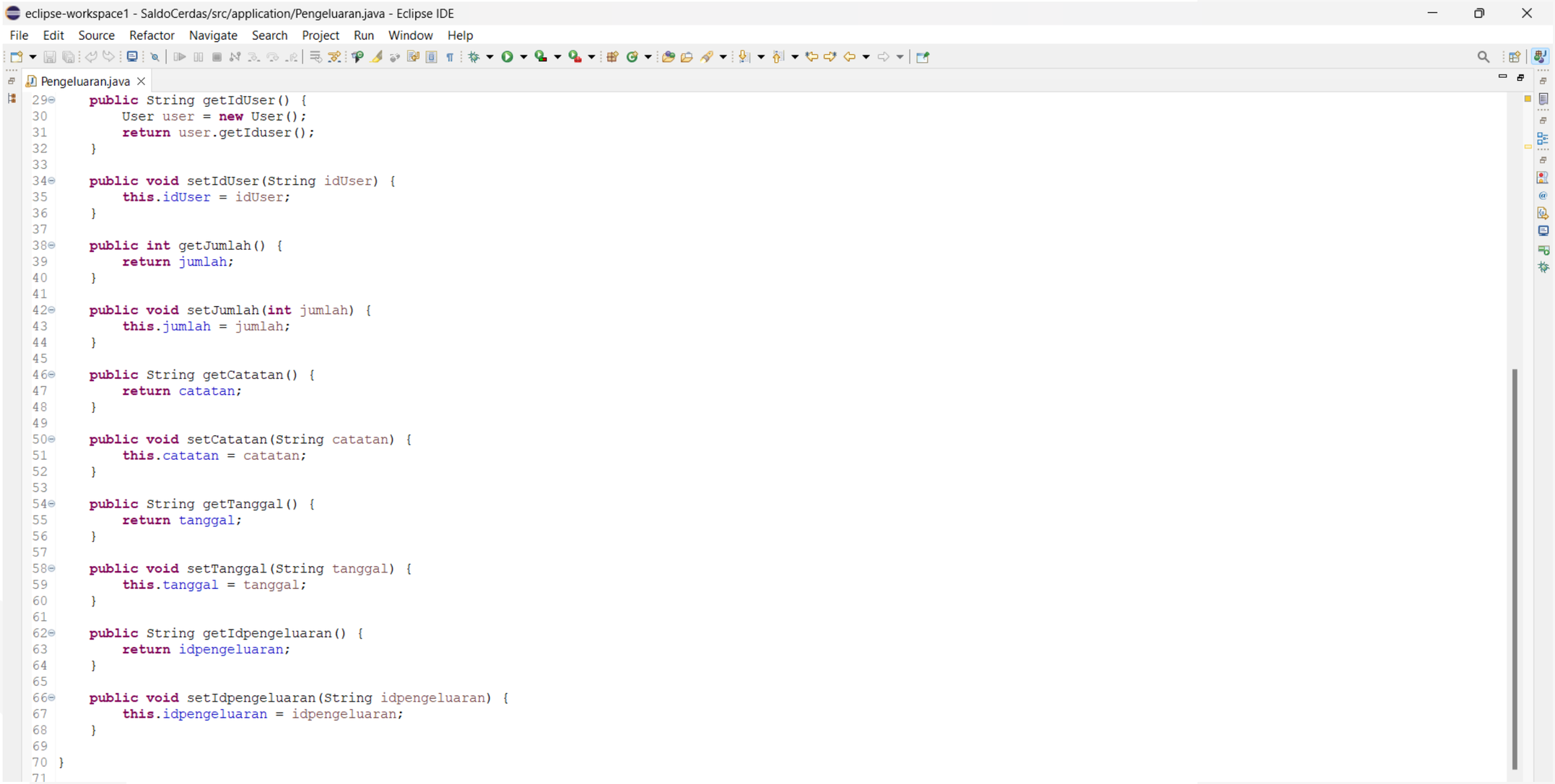


JAVA (PENGELUARAN)




```java
package application;

public class Pengeluaran {
    private String kategori;
    private String idUser;
    private int jumlah;
    private String catatan;
    private String tanggal;
    private String idpengeluaran;

    public Pengeluaran(String kategori, String idUser, int jumlah, String catatan, String tanggal, String idpengeluaran) {
        super();
        this.kategori = kategori;
        this.idUser = idUser;
        this.jumlah = jumlah;
        this.catatan = catatan;
        this.tanggal = tanggal;
        this.idpengeluaran = idpengeluaran;
    }

    public String getKategori() {
        return kategori;
    }

    public void setKategori(String kategori) {
        this.kategori = kategori;
    }

    public String getIdUser() {
        User user = new User();
        return user.getIduser();
    }

    public void setIdUser(String idUser) {
        this.idUser = idUser;
    }

    public int getJumlah() {
        return jumlah;
    }

    public void setJumlah(int jumlah) {
        this.jumlah = jumlah;
```

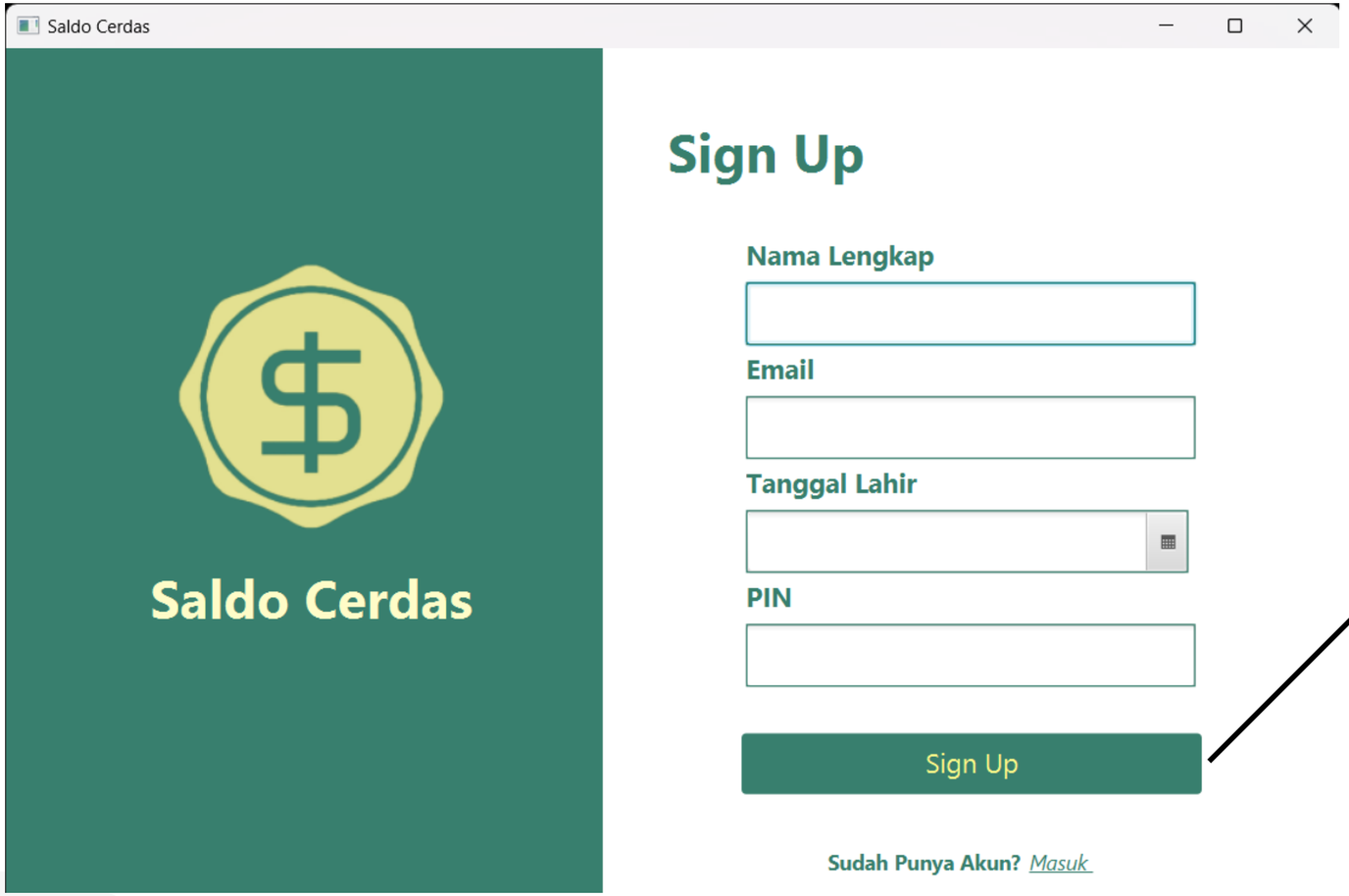
JAVA (PENGELUARAN)




```java
    public String getIdUser() {
        User user = new User();
        return user.getIduser();
    }

    public void setIdUser(String idUser) {
        this.idUser = idUser;
    }

    public int getJumlah() {
        return jumlah;
    }

    public void setJumlah(int jumlah) {
        this.jumlah = jumlah;
    }

    public String getCatatan() {
        return catatan;
    }

    public void setCatatan(String catatan) {
        this.catatan = catatan;
    }

    public String getTanggal() {
        return tanggal;
    }

    public void setTanggal(String tanggal) {
        this.tanggal = tanggal;
    }

    public String getIdpengeluaran() {
        return idpengeluaran;
    }

    public void setIdpengeluaran(String idpengeluaran) {
        this.idpengeluaran = idpengeluaran;
    }

}
```

CODING EXPLANATION FOR DB CONNECTIVITY & GUI

SIGN UP




```java
public void handleSignUp(ActionEvent event) throws IOException {
    String username = usernameField.getText();
    String email = emailField.getText();

    String pin = pinField.getText();

    String uniqueID = UUID.randomUUID().toString();
    String iduser = "USSC" + uniqueID.substring(1, 8);

    boolean isValid = true;
    StringBuilder errorMessage = new StringBuilder();

    try {

        if (usernameField.getText().isEmpty()) {
            isValid = false;
            errorMessage.append("- Username tidak boleh kosong.\n");
        } else {
            if (isUsernameExists(username)) {
                isValid = false;
                errorMessage.append("- Username sudah digunakan, silahkan ganti!\n");
            }
        }

        if (!emailField.getText().endsWith("@gmail.com")) {
            isValid = false;
            errorMessage.append("- Email harus diakhiri dengan @gmail.com.\n");
        }

        if (dateField.getValue() == null) {
            isValid = false;

            errorMessage.append("- Harap pilih tanggal.\n");
        }

        if (pinField.getText().isEmpty()) {
            isValid = false;
            errorMessage.append("- Pin tidak boleh kosong.\n");
        } else {

            if (pinField.getText().length() != 4) {
                isValid = false;
```

Ketika user menekan tombol “Sign Up” maka program akan menjalankan method eventHandler SignUpAction dan memvalidasi dengan melihat if else yang ada di dalam method tersebut, seperti textfield tidak boleh kosong, tanggal lahir wajib dipilih, panjang dari PIN dan lain-lain

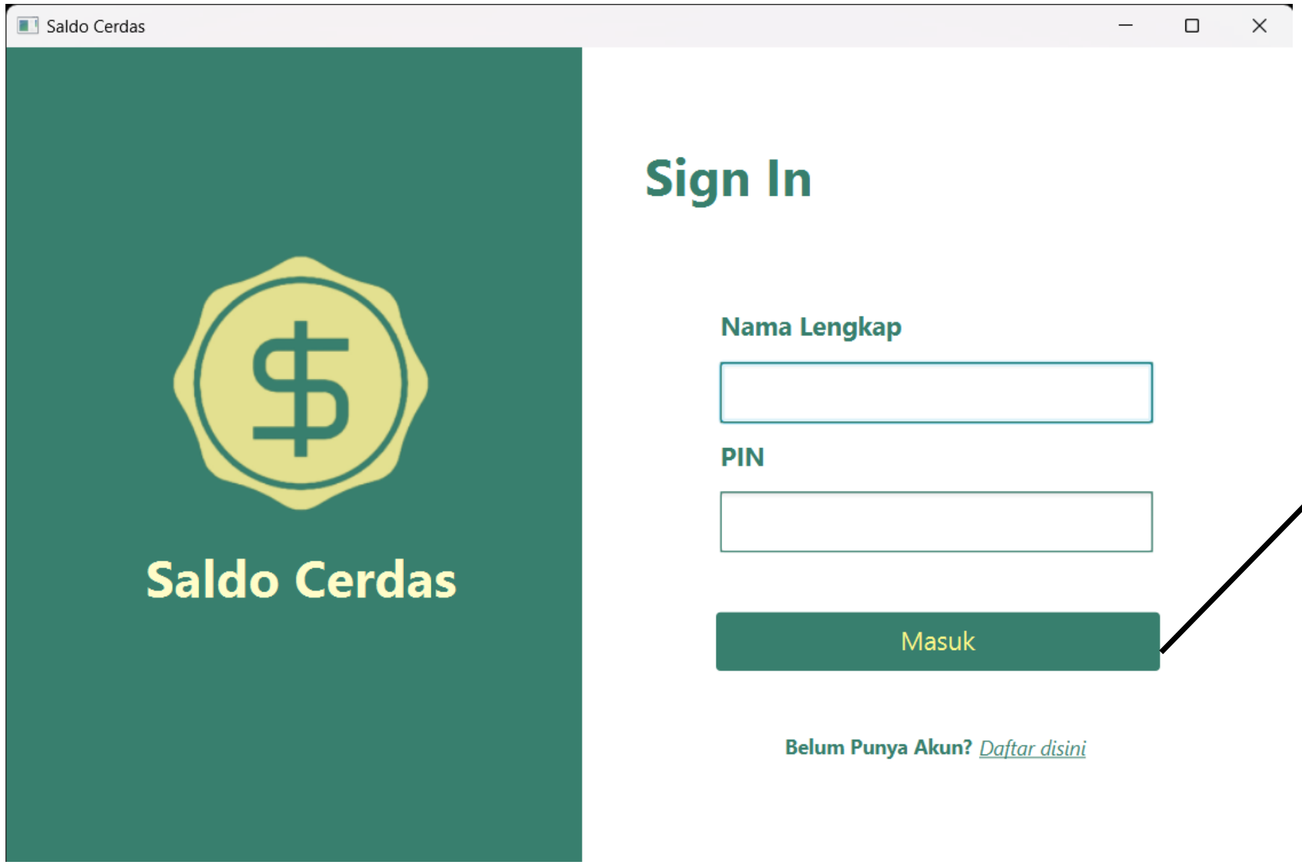
SIGN IN

```
        if (pinField.getText().isEmpty()) {
            isValid = false;
            errorMessage.append("- Pin tidak boleh kosong.\n");
        } else {

            if (pinField.getText().length() != 4) {
                isValid = false;
                errorMessage.append("- Pin harus berjumlah 4 karakter.\n");
            }

        }

        if (!isValid) {
            showErrorAlert(errorMessage.toString());
        } else {

            FXMLLoader loader = new FXMLLoader(getClass().getResource("SignIn.fxml"));
            String tanggallahir = dateField.getValue().toString();
            Parent root = loader.load();

            Scene scene = new Scene(root);
            String css = this.getClass().getResource("application.css").toExternalForm();
            scene.getStylesheets().add(css);

            Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
            primaryStage.setScene(scene);
            primaryStage.show();

            DatabaseConnector.insertDataUser(iduser, username, email, tanggallahir, pin);
            showSuccessAlert("Akun anda berhasil terdaftar!");

        }

    } catch (Exception e) {

        e.printStackTrace();
    }

}

public boolean isUsernameExists(String username) {
    boolean exists = false;
```

```
public static void insertDataUser(String iduser, String username, String email, String tanggallahir, String pin) {
    Connection connection = null;
    PreparedStatement preparedStatement = null;

    try {
        connection = connect();

        String sql = "INSERT INTO user (iduser,username, email, tanggallahir, pin) VALUES (?,?, ?, ?, ?)";
        preparedStatement = connection.prepareStatement(sql);

        preparedStatement.setString(1, iduser);
        preparedStatement.setString(2, username);
        preparedStatement.setString(3, email);
        preparedStatement.setString(4, tanggallahir);
        preparedStatement.setString(5, pin);

        preparedStatement.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            if (preparedStatement != null) {
                preparedStatement.close();
            }

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

public static void insertDataPemasukan(String idpemasukan,String iduser, String kategori, Integer jumlah, String catatan,
        String tanggal) {
    Connection connection = null;
    PreparedStatement preparedStatement = null;

    try {
        connection = connect();
```

Setelah sistem selesai memvalidasi seluruh input user dan apabila validasi tersebut bernilai True maka sistem akan menjalankan method insertDataUser() untuk memasukkan input pengguna berupa nama, email, tanggal lahir, dan juga PIN ke dalam database dengan menggunakan query INSERT. Pada program java ini, setiap pembuatan method untuk menghubungkannya ke database yang ada cukup dengan memanggil method connect() sehingga akan mengembalikan koneksi ke database tersebut

SIGN IN




```
private boolean validateLogin(String username, String pin) {
    String jdbcUrl = "jdbc:mysql://localhost:3306/saldocerdas";
    String dbUser = "root";
    String dbPassword = "";

    try {
        Connection connection = DriverManager.getConnection(jdbcUrl, dbUser, dbPassword);
        String query = "SELECT * FROM user WHERE username=? AND pin=?";
        try (PreparedStatement preparedStatement = connection.prepareStatement(query)) {
            preparedStatement.setString(1, username);
            preparedStatement.setString(2, pin);
            ResultSet resultSet = preparedStatement.executeQuery();
            return resultSet.next();
        }
    } catch (SQLException e) {
        e.printStackTrace();
        return false;
    }
}
```

Ketika user menekan tombol “Masuk” maka program akan menjalankan method validateLogin() dan memvalidasi dengan apakah input pengguna pada textField Nama Lengkap dan PasswordField PIN terdapat di dalam database tabel user, dan apabila benar maka sistem akan memasukkan user ke dalam aplikasi saldocerdas pada halaman Home

HOME


```
public void pemasukanAction(ActionEvent event) throws IOException {

    try {
        FXMLLoader loader = new FXMLLoader(getClass().getResource("Pemasukan.fxml"));
        Parent root = loader.load();

        Scene scene = new Scene(root);
        String css = this.getClass().getResource("application.css").toExternalForm();
        scene.getStylesheets().add(css);

        Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
        primaryStage.setScene(scene);
        primaryStage.show();
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

```
public void deletePemasukanAction() {
    Pemasukan selectedPemasukan = pemasukanTable.getSelectionModel().getSelectedItem();

    if (selectedPemasukan != null) {
        DatabaseConnector.hapusPemasukan(selectedPemasukan.getIdpemasukan());

        pemasukanTable.getItems().remove(selectedPemasukan);
        showSuccessAlert("Delete data pemasukan berhasil");
        pemasukanTable.refresh();
    }

}
```

Ketika user menekan tombol “Add Pemasukan” maka program akan menjalankan method pemasukanAction() dan sistem akan langsung mengarahkan user ke halaman Pemasukan yang ada pada menu Pemasukan, begitu pula juga berlaku sama seperti tombol “Add Pengeluaran” maka program akan menjalankan method pengeluaranAction() dan sistem akan langsung mengarahkan user ke halaman Pengeluaran yang ada pada menu Pengeluaran.

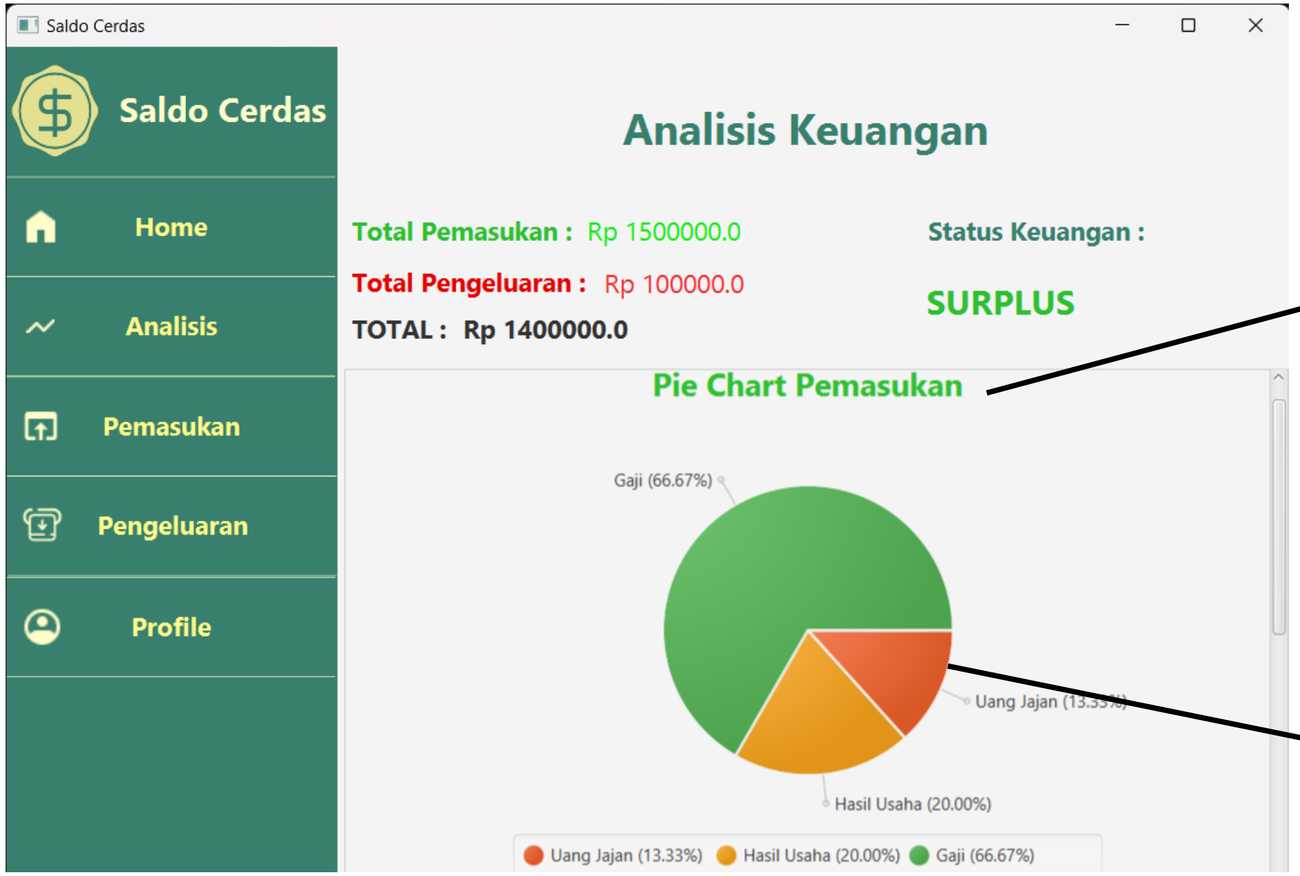
HOME


```java
public void showPemasukanData() {
    getPemasukanData();
    pemasukanTable.setItems(dataPemasukan);

}

private void getPemasukanData() {
    String url = "jdbc:mysql://localhost:3306/saldocerdas";
    String username = "root";
    String password = "";

    try {
        Connection connection = DriverManager.getConnection(url, username, password);
        String query = "SELECT * FROM pemasukan WHERE iduser = " + "'"+ user.getIduser()+"'";
        Statement statement = connection.createStatement();
        ResultSet resultSet = statement.executeQuery(query);

        while (resultSet.next()) {
            String kategori = resultSet.getString("kategori");
            int jumlah = resultSet.getInt("jumlah");
            String catatan = resultSet.getString("catatan");
            String tanggal = resultSet.getString("tanggal");
            String idpemasukan = resultSet.getString("idpemasukan");

            Pemasukan pemasukan = new Pemasukan(kategori, jumlah, catatan, tanggal, idpemasukan);
            dataPemasukan.add(pemasukan);
        }

        connection.close();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

```java
public static void hapusPemasukan(String idpemasukan) {
    Connection connection = null;
    PreparedStatement preparedStatement = null;

    try {
        connection = connect();
        String sql = "DELETE FROM pemasukan WHERE idpemasukan = ?";
        preparedStatement = connection.prepareStatement(sql);

        preparedStatement.setString(1, idpemasukan);
        preparedStatement.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    }
```

Pada halaman Home, user akan ditampilkan sebuah TableView dari semua transaksi pemasukan dan pengeluaran yang sudah dimasukkan pengguna ke dalam database dan kemudian dipanggil ke dalam tabelView tersebut yaitu kategori, jumlah, catatan dan tanggal transaksi.
Kemudian terdapat juga tombol “Delete Pemasukan” yang dimana jika pengguna menekan salah satu data yang ada di dalam TableView dan kemudian menekan tombol delete maka data yang dipilih akan terhapus dari dalam database dan akan menghapus juga data tersebut dari tampilan pada halaman Home

ANALISIS




```java
private ObservableList<PieChart.Data> getDataFromDatabase() {
    ObservableList<PieChart.Data> pieChartData = FXCollections.observableArrayList();
    double totalAmount = 0.0;
    Set<String> uniqueCategories = new HashSet<>();

    try (Connection connection = DriverManager.getConnection(DB_URL, DB_USER, DB_PASSWORD)) {
        String query = "SELECT kategori, jumlah FROM pemasukan WHERE iduser = " + "'" + user.getIduser() + "'";
        try (PreparedStatement preparedStatement = connection.prepareStatement(query);
                ResultSet resultSet = preparedStatement.executeQuery()) {
            while (resultSet.next()) {
                String category = resultSet.getString("kategori");

                if (uniqueCategories.add(category)) {
                    double amount = resultSet.getDouble("jumlah");
                    totalAmount += amount;
                    pieChartData.add(new PieChart.Data(category, amount));
                }

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

    for (PieChart.Data data : pieChartData) {
        double percentage = (data.getPieValue() / totalAmount) * 100;
        String originalLabel = data.getName();
        data.setName(originalLabel + " (" + String.format("%.2f%%", percentage) + ")");

    }
    return pieChartData;
}
```

```java
private void updatePieChart() {
    ObservableList<PieChart.Data> data = getDataFromDatabase();
    pieChart.setData(data);
    for (PieChart.Data entry : data) {
        entry.getNode().addEventHandler(MouseEvent.MOUSE_CLICKED, e -> {
            // Handle click events if needed
            System.out.println(entry.getName() + ": " + entry.getPieValue() + " - " + entry.getName());
            showSuccessAlert(entry.getName() + ": " + entry.getPieValue());
        });
    }
}
```

Ketika user memilih menu Analisis maka sistem akan menampilkan data-data yang ada di dalam database berdasarkan input pengguna saat itu dan menampilkannya dalam bentuk pie chart dengan menggunakan method getDataFromDatabase dan dimasukkan ke dalam variable pie chart yaitu pieChartData. Kemudian apabila user menambahkan transaksi pemasukan atau pengeluaran maka sistem akan menjalankan method updatePieChart() untuk merefresh pie chart dengan data terbaru yang ada di dalam database setelah terjadi perubahan.

ANALISIS

```
49    public void totalKeseluruhan() {
50
51        try {
52            DatabaseConnector.connect();
53            Connection connection = DatabaseConnector.connect();
54            Statement statement = connection.createStatement();
55
56            ResultSet resultSetPemasukan = statement
57                    .executeQuery("SELECT SUM(jumlah) FROM pemasukan WHERE iduser = " + "'" + user.getIduser() + "'");
58            double totalPemasukan = 0;
59            if (resultSetPemasukan.next()) {
60                totalPemasukan = resultSetPemasukan.getDouble(1);
61                pemasukanTotal.setText("Rp " + totalPemasukan);
62            }
63            resultSetPemasukan.close();
64
65            ResultSet resultSetPengeluaran = statement
66                    .executeQuery("SELECT SUM(jumlah) FROM pengeluaran WHERE iduser = " + "'" + user.getIduser() + "'");
67            double totalPengeluaran = 0;
68            if (resultSetPengeluaran.next()) {
69                totalPengeluaran = resultSetPengeluaran.getDouble(1);
70                pengeluaranTotal.setText("Rp " + totalPengeluaran);
71            }
72            resultSetPengeluaran.close();
73
74            double selisih = totalPemasukan - totalPengeluaran;
75
76            totalSemua.setText("Rp " + selisih);
77
78            if (totalPemasukan > totalPengeluaran) {
79                statusKeuangan.setText("SURPLUS");
```

```
            if (totalPemasukan > totalPengeluaran) {
                statusKeuangan.setText("SURPLUS");
                statusKeuangan.setStyle("-fx-text-fill : #2ebf30");
            } else if (totalPengeluaran > totalPemasukan) {
                statusKeuangan.setText("DEFISIT");
                statusKeuangan.setStyle("-fx-text-fill : #e40000");
            } else {
                statusKeuangan.setText("");
            }

            statement.close();
            connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }

    }
```

Ketika user memilih menu Analisis maka sistem akan menampilkan data-data yang ada di dalam database berdasarkan input pengguna saat itu, terdapat section total pemasukan, pengeluaran, selisih dan juga status keuangan yang dimana angka-angka tersebut didapatkan melalui query SUM(jumlah) yang ada di dalam database dan kemudian akan dihitung jumlah selisih antara totalPemasukan - totalPengeluaran, dan melalui nilai ini maka akan menampilkan juga status keuangannya, apabila positif maka statusnya “SURPLUS” dan jika negatif maka statusnya “DEFISIT”

PEMASUKAN


```
public void OnAdd(ActionEvent event) {

    String catatan = catatanText.getText();

    boolean isValid = true;
    StringBuilder errorMessage = new StringBuilder();

    String uniqueID = UUID.randomUUID().toString();
    String idpemasukan = "PMS" + uniqueID.substring(1, 8);


    try {

        if (pemasukanBox.getValue() == null) {
            isValid = false;

            errorMessage.append("- Harap pilih kategori pemasukan.\n");
        }

        if (hargaText.getText().isEmpty()) {
            isValid = false;
            errorMessage.append("- Jumlah tidak boleh kosong.\n");
        } else {
            Integer jumlah = Integer.parseInt(hargaText.getText());
            if (jumlah <= 0) {
                isValid = false;
                errorMessage.append("- Jumlah tidak boleh lebih kecil dari atau sama dengan 0.\n");
            }
        }

        if (tanggalPemasukan.getValue() == null) {
            isValid = false;

            errorMessage.append("- Harap pilih tanggal pemasukan.\n");
        }

        if (!isValid) {
            showErrorAlert(errorMessage.toString());
        } else {
            String kategori = pemasukanBox.getValue().toString();
            String tanggal = tanggalPemasukan.getValue().toString();
            Integer jumlah = Integer.parseInt(hargaText.getText());
```

Ketika user memilih menu pemasukan, terdapat beberapa data yang harus dimasukkan oleh user diantaranya adalah Kategori dalam bentuk combo box, jumlah dalam bentuk textfield, catatan dalam bentuk textfield dan juga tanggal dalam bentuk date picker. Ketika semua data telah terisi dan sudah sesuai dengan ketentuan validasi. Maka user dapat menekan tombol save lalu aplikasi akan menampilkan notifikasi sukses yang berarti data sudah berhasil diinsert ke dalam database.

PEMASUKAN

```
        if (tanggalPemasukan.getValue() == null) {
            isValid = false;

            errorMessage.append("- Harap pilih tanggal pemasukan.\n");
        }

        if (!isValid) {
            showErrorAlert(errorMessage.toString());
        } else {
            String kategori = pemasukanBox.getValue().toString();
            String tanggal = tanggalPemasukan.getValue().toString();
            Integer jumlah = Integer.parseInt(hargaText.getText());
            User user = new User();
            String iduser = user.getIduser();

            DatabaseConnector.insertDataPemasukan(idpemasukan, iduser, kategori, jumlah, catatan, tanggal);
//          showAddAlert("Apakah data pemasukan sudah sesuai?");
            showSuccessAlert("Data pemasukan berhasil di add!");

            pemasukanBox.setValue(null);
            tanggalPemasukan.setValue(null);
            hargaText.setText("");
            catatanText.setText("");

        }

    } catch (Exception e) {
        showErrorAlert("Masukkan jumlah dalam bentuk angka");
        e.printStackTrace();
    }

}
```

```
public static void insertDataPemasukan(String idpemasukan,String iduser, String kategori, Integer jumlah, String catatan,
        String tanggal) {
    Connection connection = null;
    PreparedStatement preparedStatement = null;

    try {
        connection = connect();

        String sql = "INSERT INTO pemasukan (idpemasukan,iduser, kategori, jumlah, catatan, tanggal) VALUES (?, ?, ?, ?, ?, ?)";
        preparedStatement = connection.prepareStatement(sql);

        preparedStatement.setString(1, idpemasukan);
        preparedStatement.setString(2, iduser);
        preparedStatement.setString(3, kategori);
        preparedStatement.setInt(4, jumlah);
        preparedStatement.setString(5, catatan);
        preparedStatement.setString(6, tanggal);

        preparedStatement.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            if (preparedStatement != null) {
                preparedStatement.close();
            }

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

Kemudian pada class DatabaseConnector terdapat sebuah public void insertDataPemasukan yang akan dipanggil pada class Pemasukan untuk melakukan query ke dalam database berupa INSERT TO sesuai dengan data yang telah dimasukkan oleh user. Dan disini juga terdapat idpemasukan yang akan digenerate secara random dan diberikan untuk setiap pencatatan pemasukan yang telah diinput.

PENGELUARAN


```java
public void OnAdd(ActionEvent event) {

    String catatan = catatanText.getText();

    boolean isValid = true;
    StringBuilder errorMessage = new StringBuilder();

    String uniqueID = UUID.randomUUID().toString();
    String idpengeluaran = "PNG" + uniqueID.substring(1, 8);

    try {
        if (pengeluaranBox.getValue() == null) {
            isValid = false;

            errorMessage.append("- Harap pilih kategori pengeluaran.\n");
        }

        if (hargaText.getText().isEmpty()) {
            isValid = false;
            errorMessage.append("- Jumlah tidak boleh kosong.\n");
        } else {
            Integer jumlah = Integer.parseInt(hargaText.getText());
            if (jumlah <= 0) {
                isValid = false;
                errorMessage.append("- Jumlah tidak boleh lebih kecil dari atau sama dengan 0.\n");
            }
        }

        if (tanggalPengeluaran.getValue() == null) {
            isValid = false;

            errorMessage.append("- Harap pilih tanggal pengeluaran.\n");
        }

        if (!isValid) {
            showErrorAlert(errorMessage.toString());
        } else {
            String kategori = pengeluaranBox.getValue().toString();
            String tanggal = tanggalPengeluaran.getValue().toString();
            Integer jumlah = Integer.parseInt(hargaText.getText());
            User user = new User();
            String iduser = user.getIduser();
```

Ketika user memilih menu pengeluaran, terdapat beberapa data yang harus dimasukkan oleh user diantaranya adalah Kategori dalam bentuk combo box, jumlah dalam bentuk textfield, catatan dalam bentuk textfield dan juga tanggal dalam bentuk date picker. Ketika semua data telah terisi dan sudah sesuai dengan ketentuan validasi. Maka user dapat menekan tombol save lalu aplikasi akan menampilkan notifikasi sukses yang berarti data sudah berhasil diinsert ke dalam database.

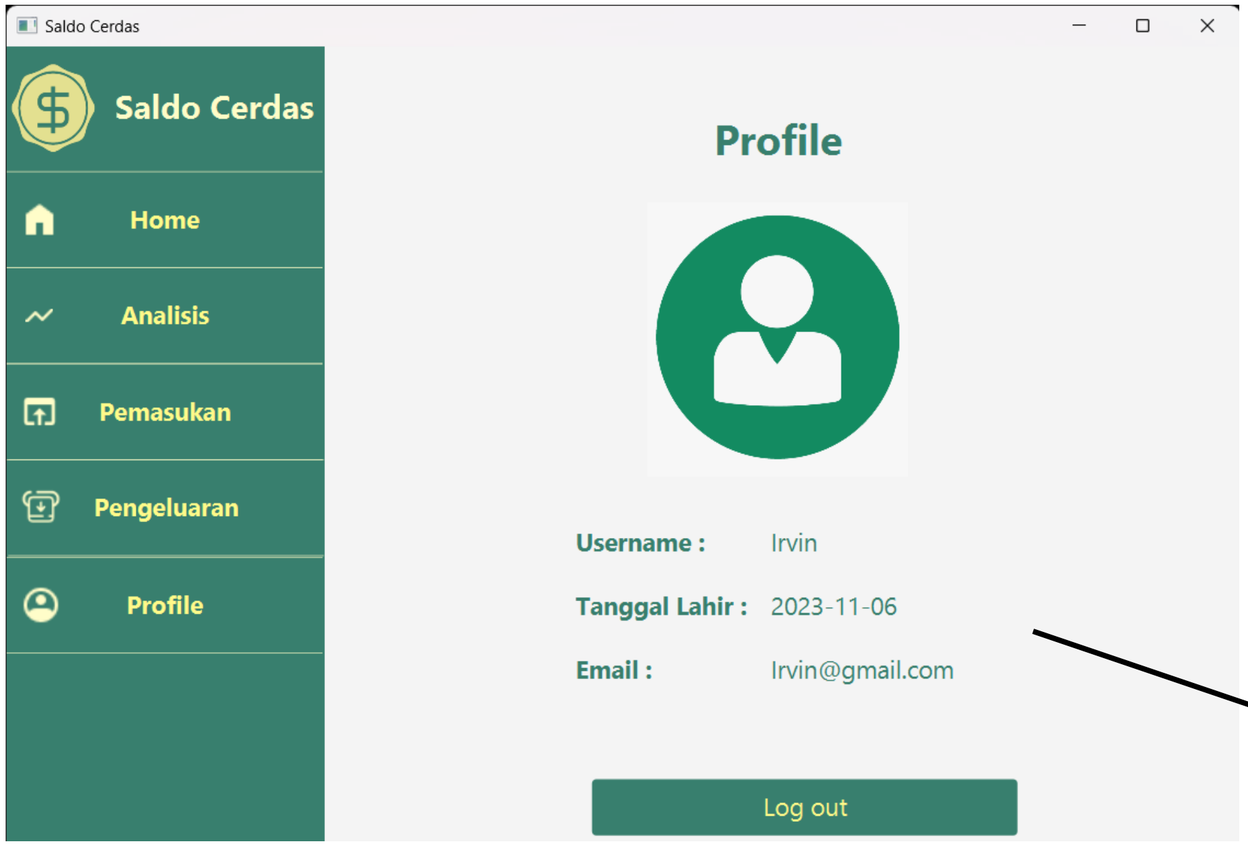
PEMASUKAN

```
    if (!isValid) {
        showErrorAlert(errorMessage.toString());
    } else {
        String kategori = pengeluaranBox.getValue().toString();
        String tanggal = tanggalPengeluaran.getValue().toString();
        Integer jumlah = Integer.parseInt(hargaText.getText());
        User user = new User();
        String iduser = user.getIduser();

        DatabaseConnector.insertDataPengeluaran(idpengeluaran, iduser, kategori, jumlah, catatan, tanggal);
        showSuccessAlert("Data pengeluaran berhasil di add!");

        pengeluaranBox.setValue(null);
        tanggalPengeluaran.setValue(null);
        hargaText.setText("");
        catatanText.setText("");

    }

} catch (Exception e) {
    showErrorAlert("Masukkan jumlah dalam bentuk angka");
    e.printStackTrace();
}
```

```
public static void insertDataPengeluaran(String idpengeluaran,String iduser, String kategori, Integer jumlah, String catatan,
        String tanggal) {
    Connection connection = null;
    PreparedStatement preparedStatement = null;

    try {
        connection = connect();

        String sql = "INSERT INTO pengeluaran (idpengeluaran, iduser, kategori, jumlah, catatan, tanggal) VALUES (?, ?, ?, ?, ?, ?)";
        preparedStatement = connection.prepareStatement(sql);

        preparedStatement.setString(1, idpengeluaran);
        preparedStatement.setString(2, iduser);
        preparedStatement.setString(3, kategori);
        preparedStatement.setInt(4, jumlah);
        preparedStatement.setString(5, catatan);
        preparedStatement.setString(6, tanggal);

        preparedStatement.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            if (preparedStatement != null) {
                preparedStatement.close();
            }

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

Kemudian pada class DatabaseConnector terdapat sebuah public void insertDataPengeluaran yang akan dipanggil pada class Pengeluaran untuk melakukan query ke dalam database berupa INSERT TO sesuai dengan data yang telah dimasukkan oleh user. Dan disini juga terdapat idpengeluaran yang akan digenerate secara random dan diberikan untuk setiap pencatatan pengeluaran yang telah diinput.

PROFILE




```
public class User {

    public static String loggedInUsername;
    public String username = DatabaseConnector.getUsername(loggedInUsername);
    public String iduser = DatabaseConnector.getIdUser(username);
    public String email = DatabaseConnector.getEmail(username);
    public String tanggallahir = DatabaseConnector.getDOB(username);
    public String pin;

    public String getIduser() {
        return iduser;
    }
    public void setIduser(String iduser) {
        this.iduser = iduser;
    }
    public String getUsername() {
        return username;
    }
    public void setUsername(String username) {
        this.username = username;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public String getTanggallahir() {
        return tanggallahir;
    }
    public void setTanggallahir(String tanggallahir) {
        this.tanggallahir = tanggallahir;
    }
```

```
public void ubahProfile() {
    User user = new User();
    usernameLabel.setText("" + user.getUsername());
    tanggalLabel.setText("" + user.getTanggallahir());
    emailLabel.setText("" + user.getEmail());
}
```

Ketika user memilih menu Profile maka sistem akan menampilkan data user yang sedang login di dalam aplikasi tersebut dengan cara memanggil getter setter yang ada di class User dan kemudian sistem akan menjalankan method ubahProfile() pada halaman profile ini dan akan tertampil data-datanya berupa username, tanggal lahir dan email, pada class User, username, iduser, email dan tanggallahir di set nilainya dengan menggunakan method yang ada pada DatabaseConnector sehingga akan menampilkan data user berdasarkan user yang sedang login saat itu juga

PROFILE

```java
public static String getEmail(String email) {
    String query = "SELECT email FROM user WHERE username = ?";
    try (Connection connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/saldocerdas", "root", "");
         PreparedStatement preparedStatement = connection.prepareStatement(query)) {
        preparedStatement.setString(1, email);
        ResultSet resultSet = preparedStatement.executeQuery();
        if (resultSet.next()) {
            return resultSet.getString("email");
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return email;
}
```

```java
public static String getUsername(String username) {
    String query = "SELECT username FROM user WHERE username = ?";
    try (Connection connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/saldocerdas", "root", "");
         PreparedStatement preparedStatement = connection.prepareStatement(query)) {
        preparedStatement.setString(1, username);
        ResultSet resultSet = preparedStatement.executeQuery();
        if (resultSet.next()) {
            return resultSet.getString("username");
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return username;
}
```

```java
public static String getDOB(String tanggallahir) {
    String query = "SELECT tanggallahir FROM user WHERE username = ?";
    try (Connection connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/saldocerdas", "root", "");
         PreparedStatement preparedStatement = connection.prepareStatement(query)) {
        preparedStatement.setString(1, tanggallahir);
        ResultSet resultSet = preparedStatement.executeQuery();
        if (resultSet.next()) {
            return resultSet.getString("tanggallahir");
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return tanggallahir;
}
```

Berikut ini adalah tampilan method-method yang ada untuk menentukan nilai dari data user berupa username, email dan tanggil lahirnya dengan menggunakan query SELECT ... WHERE username = ?
dan ketika method ini dijalankan maka sistem akan menampilkan data sesuai dengan username yang sedang login

QUESTION & ANSWER

Siapa target market dari Saldo Cerdas?

Target market dari Saldo Cerdas adalah mahasiswa karena memiliki penghasilan atau pengeluaran yang belum terlalu besar.

Kenapa perlu registrasi?

Memberikan akses eksklusifitas kepada user

Jika melakukan log out, apakah data yang sudah di input akan hilang?

Tidak, data yang disimpan akan tetap tersimpan sesuai akun pengguna meskipun sudah log out.

CHAPTER III

CONCLUSION

December 2023

CONSLUSION

Dengan aplikasi Saldo Cerdas ini, kita bisa dengan mudahnya mengatur, mencatat, dan melihat keuangan pribadi kita secara akurat, mulai dari pemasukan, pengeluaran, utang-piutang dan juga analisis keuangan.

LESSON LEARNED

Melalui project ini, kami belajar bahwa progress lebih penting dari pada hasil akhir. Kami juga belajar bahwa kontribusi aktif dari teman-teman sekelompok sangat berpengaruh dalam kinerja tim.

THANK YOU

FOR YOUR NICE ATTENTION

Business Application Development