



# SALDO CERDAS



● FINAL REPORT

# 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

---

<b>01</b>	<b>Chapter I - Preliminary</b>	
	• Background and Objective	5-6
	• Roles and Responsibility	7
	• Timeline	8
<b>02</b>	<b>Chapter II - Business Application</b>	
	• Use Case and Activity Diagram	10-11
	• GUI Design	13
	• DB Design	24
	• Coding explanation for DB Connectivity and GUI	73
	• Question and Answer	87
<b>03</b>	<b>Chapter III - Conclusion</b>	88





# 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



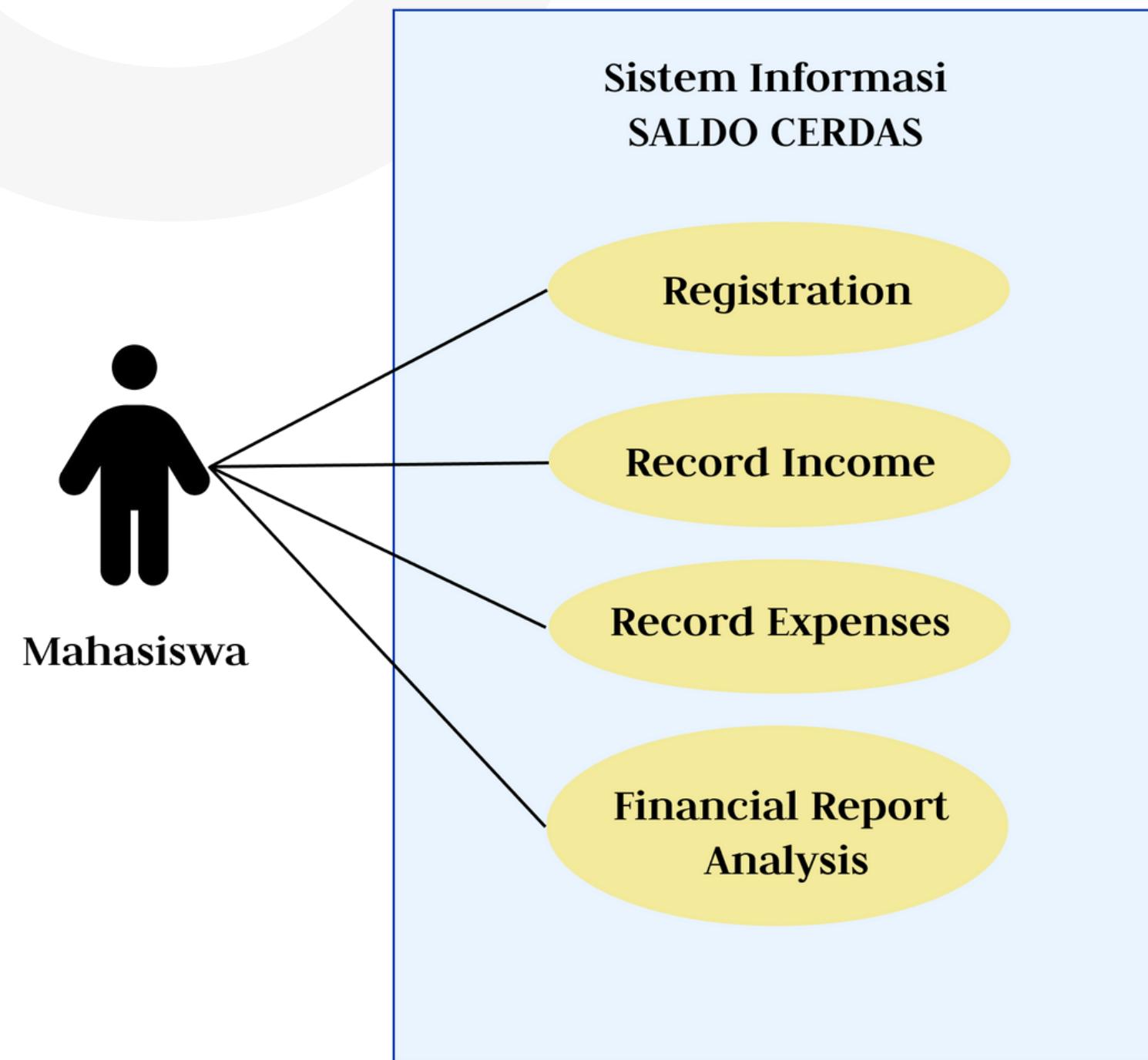
# CHAPTER II

● BUSINESS APPLICATION

---

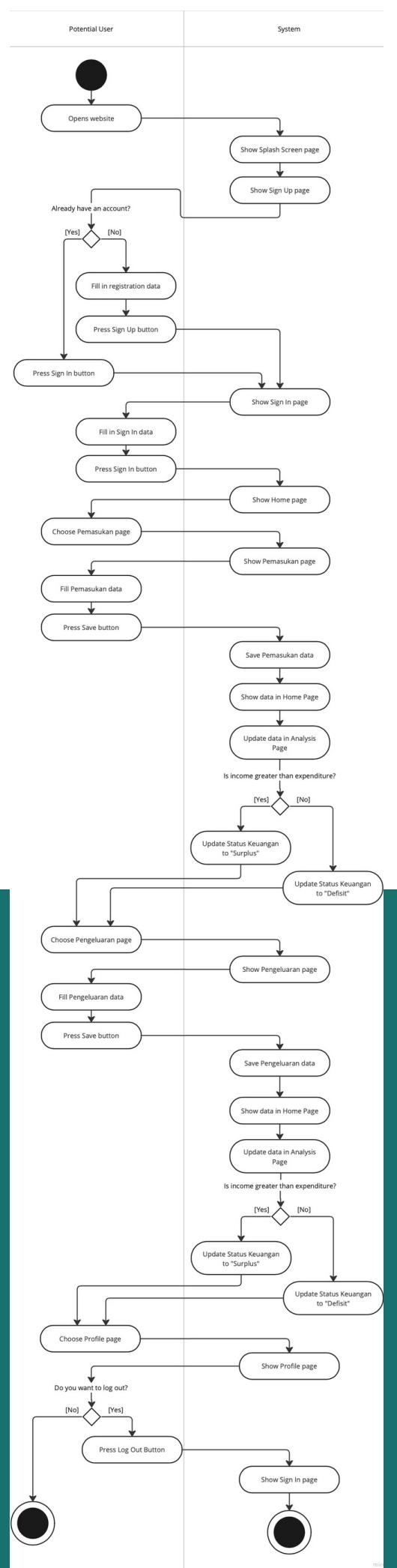
December 2023

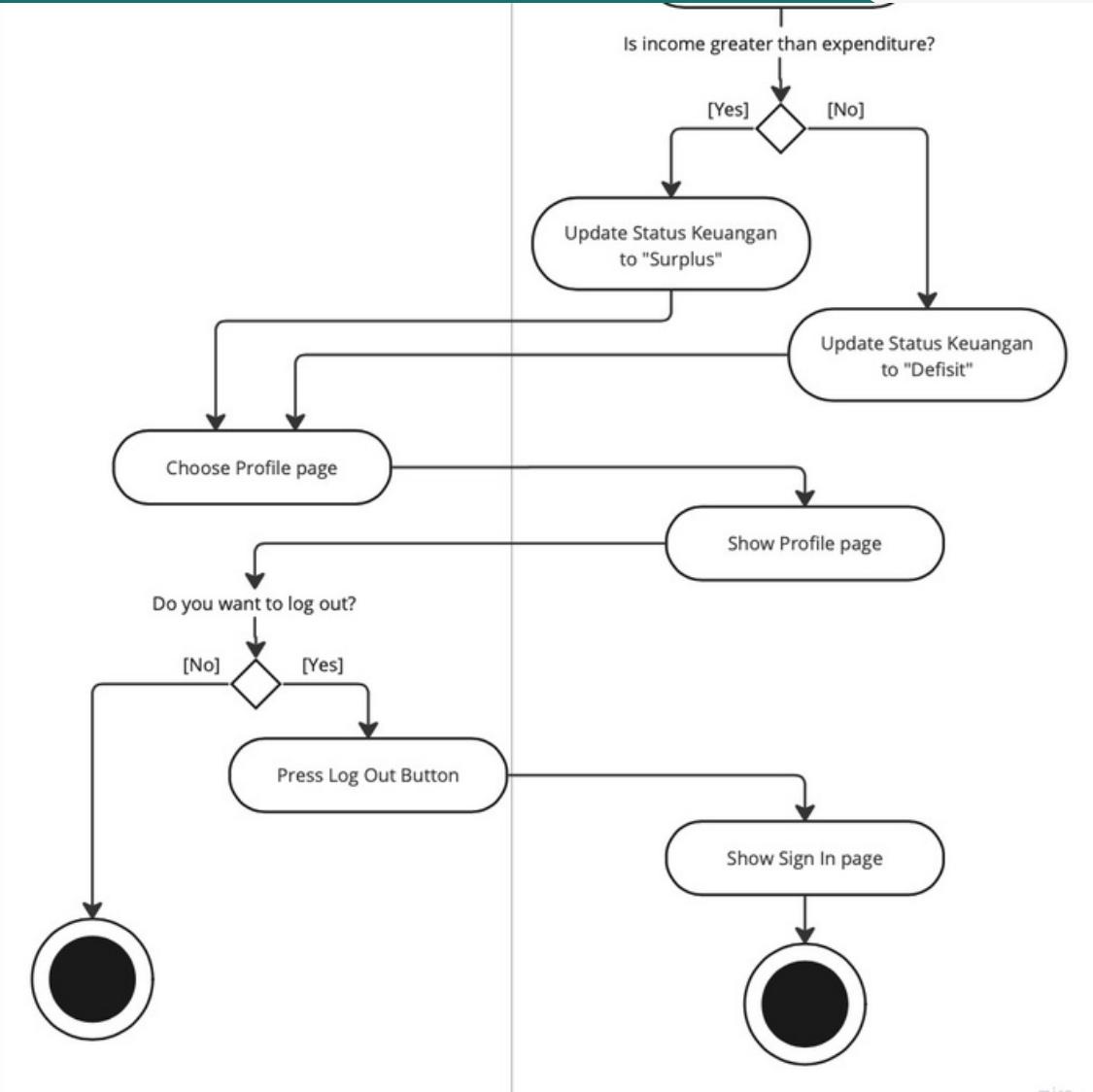
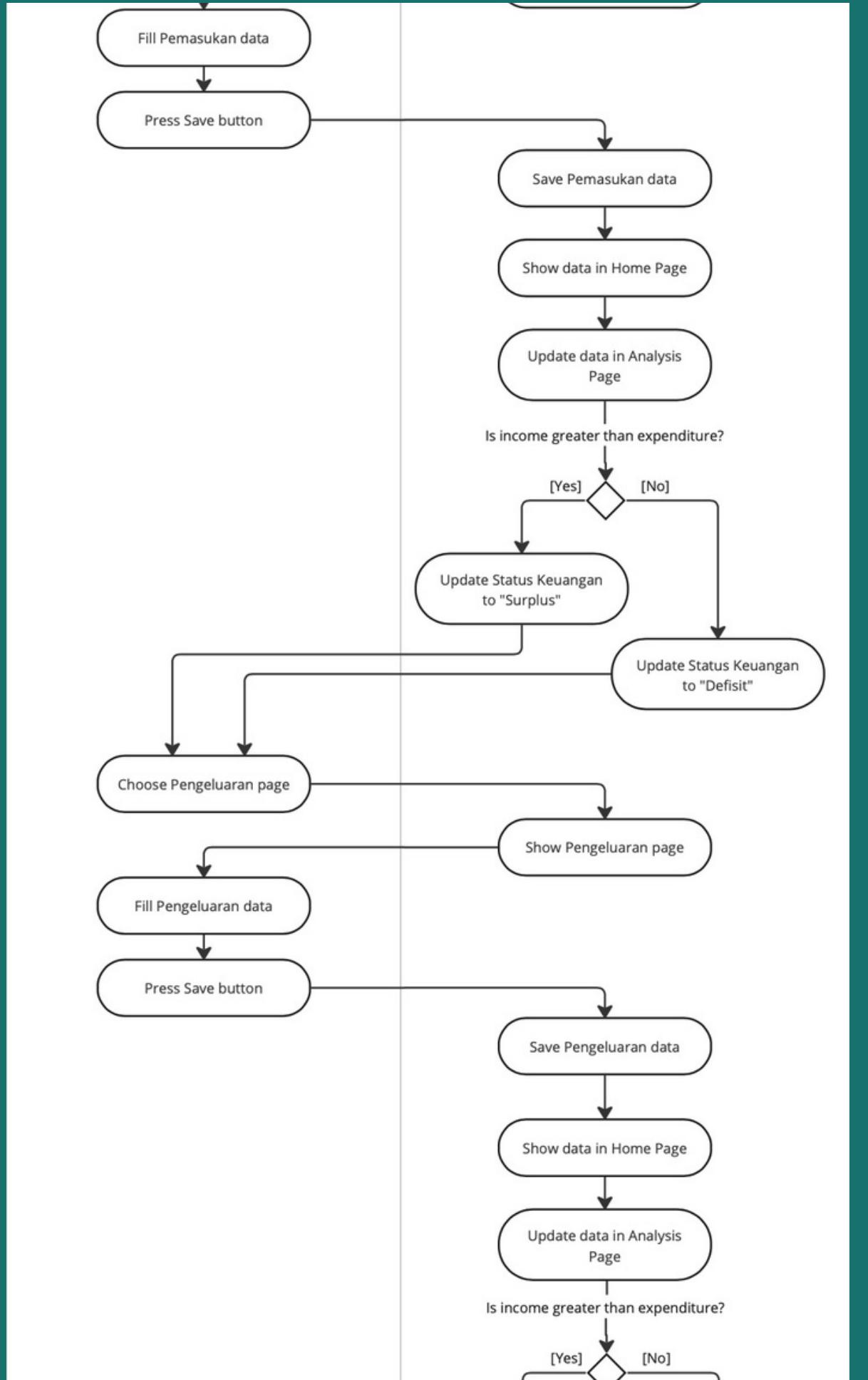
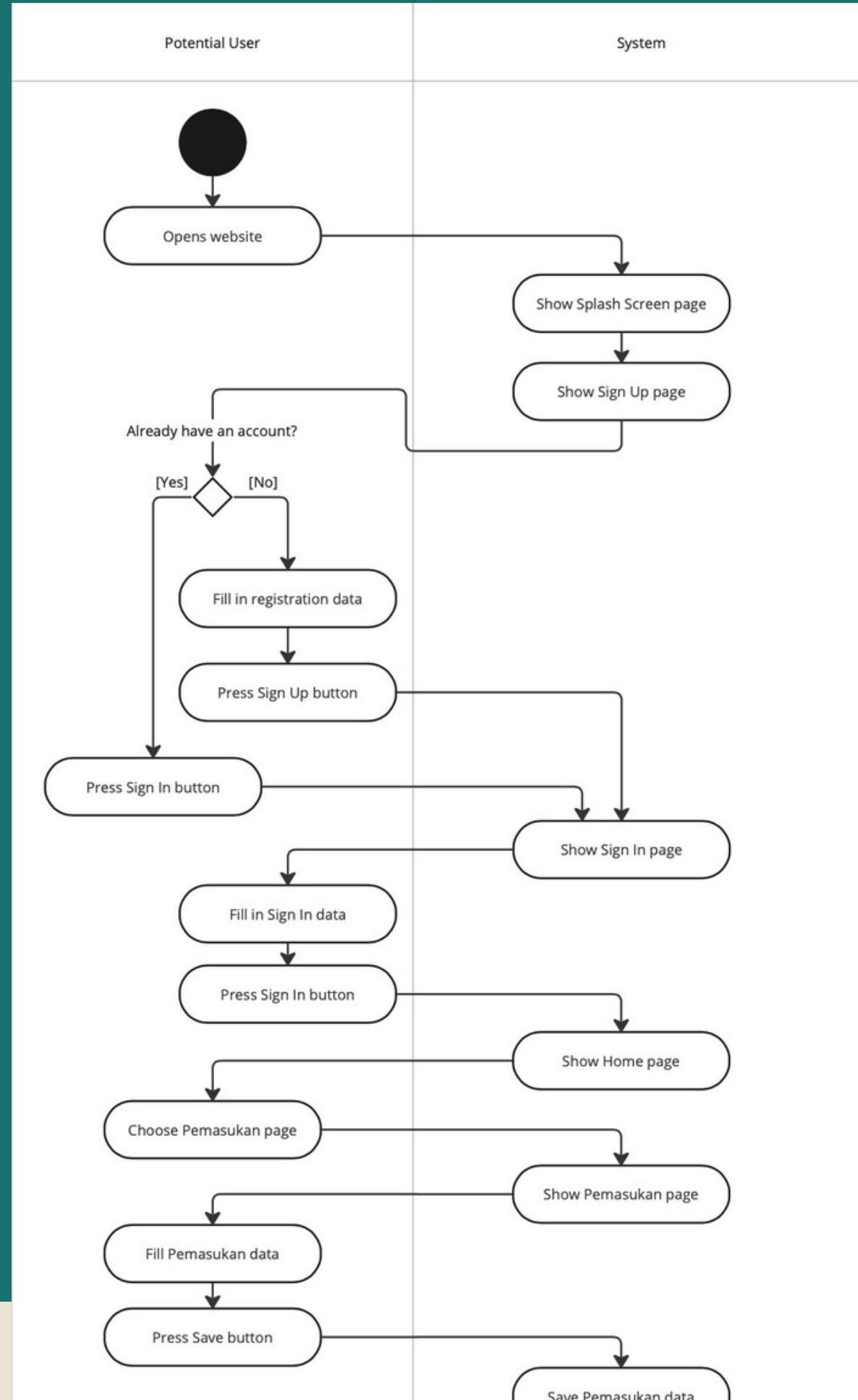
# USE CASE SALDO CERDAS



# ACTIVITY DIAGRAM SALDO CERDAS

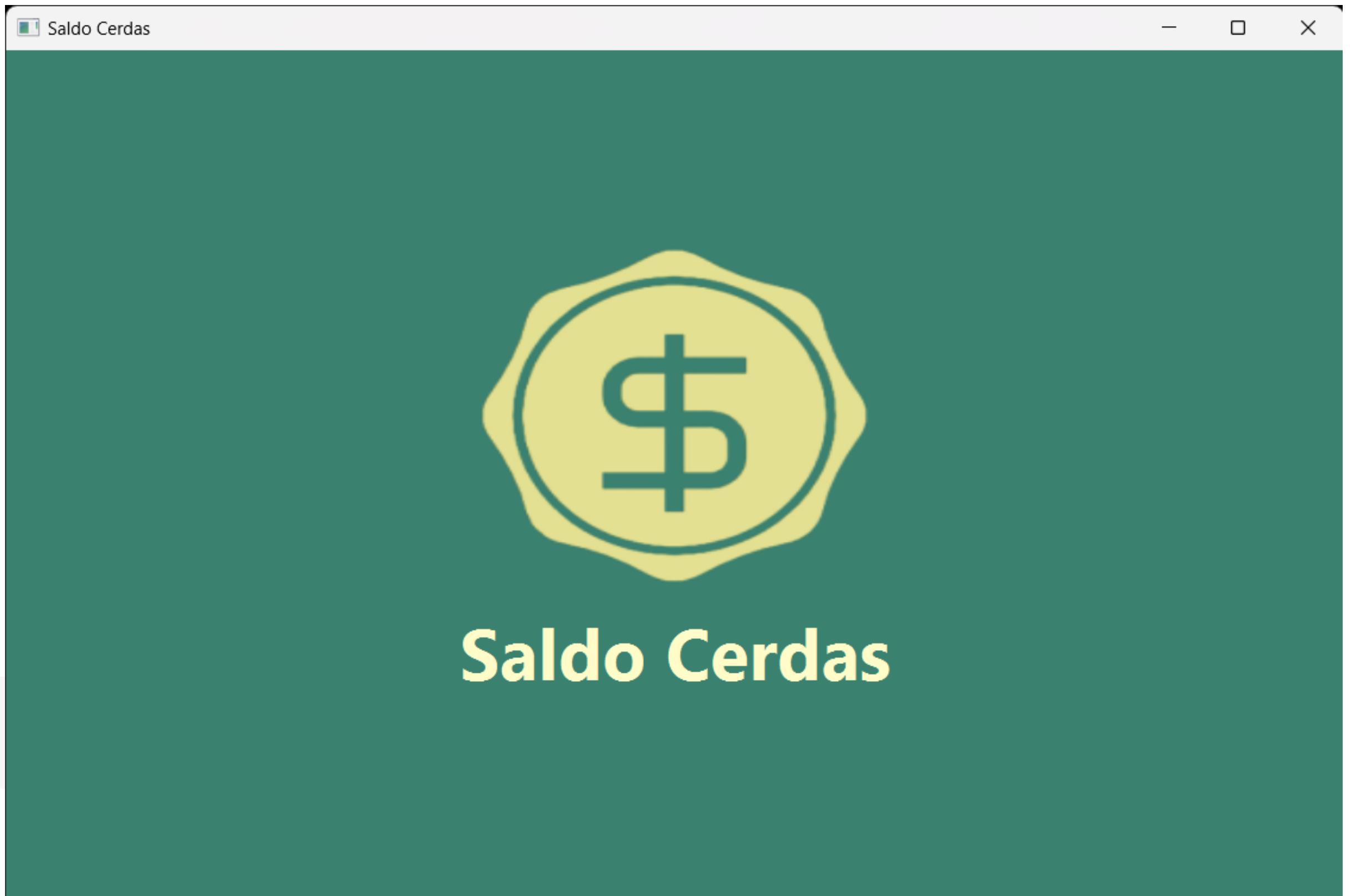
[https://drive.google.com/file/d/10-d38dTtERVzHmFHHQtNmeO3H869XPz1/view?usp=share\\_link](https://drive.google.com/file/d/10-d38dTtERVzHmFHHQtNmeO3H869XPz1/view?usp=share_link)





# GUI DESIGN

# SPLASH SCREEN



# SIGN UP



## Sign Up

Nama Lengkap

Email

Tanggal Lahir

PIN

Sign Up

Sudah Punya Akun? [Masuk](#)

# SIGN IN

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
Dondok	50000	Bawa Babi Babi	2023-12-01

# 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

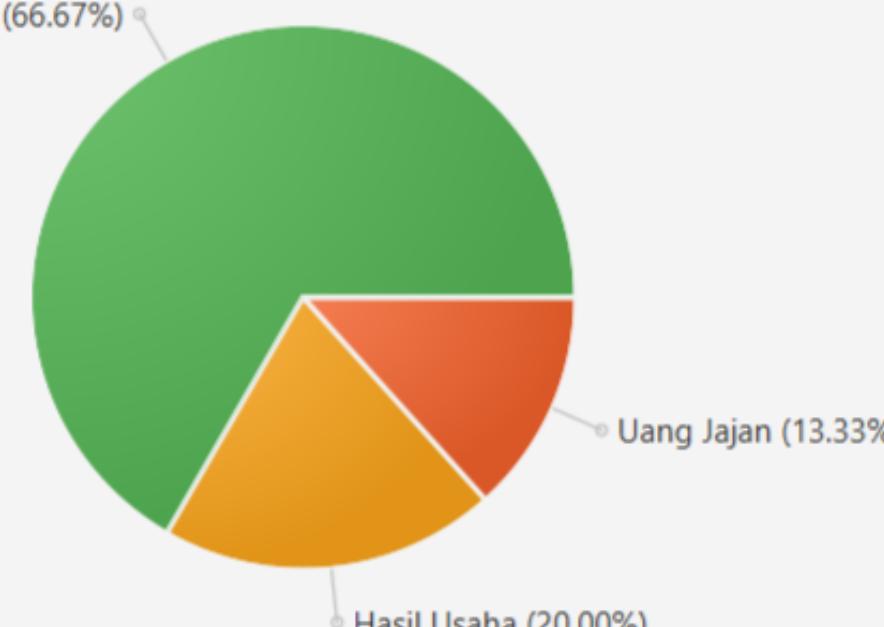
Total Pengeluaran : Rp 100000.0

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

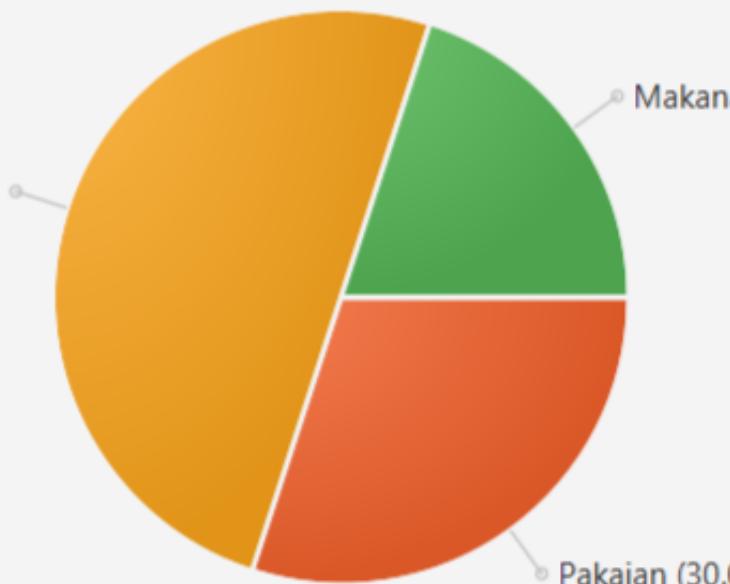
Total Pengeluaran : Rp 100000.0

TOTAL : Rp 1400000.0

Status Keuangan :

**SURPLUS**

**Pie Chart Pengeluaran**



Kategori	Persentase
Pendidikan	50.00%
Pakaian	30.00%
Makanan	20.00%

Legend:

- Pakaian (30.00%)
- Pendidikan (50.00%)
- Makanan (20.00%)

# PEMASUKAN

The image shows a screenshot of a Windows application window titled "Saldo Cerdas". The main title bar has a green background with white text. The left sidebar, also with a green background, lists several menu items: "Home", "Analisis", "Pemasukan" (selected), and "Pengeluaran". The "Pemasukan" item is highlighted with a yellow icon of an upward arrow. The main content area is titled "Pemasukan" in large green font. It contains four input fields: "Kategori" (Category) with a dropdown menu labeled "Pilih Kategori" (Select Category); "Jumlah" (Amount) with a text input field; "Catatan (opsional)" (Optional Note) with a text input field containing placeholder text "Masukkan Catatan" (Enter Note); and "Tanggal" (Date) with a date picker input field labeled "Pilih Tanggal" (Select Date). At the bottom is a large green "Save" button.

Saldo Cerdas

Saldo Cerdas

Home

Analisis

Pemasukan

Pengeluaran

Profile

Pemasukan

Pemasukan

Kategori

Pilih Kategori

Jumlah

Catatan (opsional)

Masukkan Catatan

Tanggal

Pilih Tanggal

Save

# PENGELUARAN

The image shows a mobile application interface for managing finances. On the left is a vertical navigation bar with the following items:

- Saldo Cerdas (with a coin icon)
- Home (with a house icon)
- Analisis (with a chart icon)
- Pemasukan (with an upward arrow icon)
- Pengeluaran (with a downward arrow icon)
- Profile (with a person icon)

The main content area is titled "Pengeluaran" (Expense). It contains four input fields:

- Kategori:** A dropdown menu labeled "Pilih Kategori".
- Jumlah:** An input field with placeholder text "Masukkan jumlah dalam bentuk angka".
- Catatan (opsional):** An input field with placeholder text "Masukkan Catatan".
- Tanggal:** A date picker input labeled "Pilih Tanggal".

A large green "Save" button is located at the bottom of the form.

# PROFILE

Saldo Cerdas

The image shows a screenshot of a mobile application named "Saldo Cerdas". The top navigation bar has a back arrow and the text "Saldo Cerdas". The main content area is titled "Profile". It features a large circular placeholder icon for a user profile picture. Below the icon, the user's information is displayed: Username: Irvin, Tanggal Lahir: 2023-11-06, and Email: Irvin@gmail.com. At the bottom is a green "Log out" button. On the left side, there is a vertical sidebar with the following menu items: Home, Analisis, Pemasukan, Pengeluaran, and Profile. Each item has an associated icon.

- Home
- Analisis
- Pemasukan
- Pengeluaran
- Profile

**Profile**

Username : Irvin

Tanggal Lahir : 2023-11-06

Email : Irvin@gmail.com

Log out

# DATABASE DESIGN

# DB PEMASUKAN

phpMyAdmin

Server: 127.0.0.1 » Database: saldocerdas » Table: pemasukan

Browse Structure SQL Search Insert Export Import Privileges Operations

Table structure Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	idpemasukan	char(10)	utf8mb4_general_ci		No	None			<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>
2	iduser	char(11)	utf8mb4_general_ci		No	None			<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>
3	kategori	varchar(50)	utf8mb4_general_ci		No	None			<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>
4	jumlah	bigint(20)			No	None			<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>
5	catatan	varchar(100)	utf8mb4_general_ci		No	None			<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>
6	tanggal	date			No	None			<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>

Check all With selected: Browse Change Drop Primary Unique Index Spa

Remove from central columns

New cangkir information\_schema mysql performance\_schema phpmyadmin saldocerdas New pemasukan pengeluaran user

# DB PENGELUARAN

phpMyAdmin

Server: 127.0.0.1 » Database: saldocerdas » Table: pengeluaran

Browse Structure SQL Search Insert Export Import Privileges Operations

Table structure Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	idpengeluaran	char(10)	utf8mb4_general_ci		No	None			<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>
2	iduser	char(11)	utf8mb4_general_ci		No	None			<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>
3	kategori	varchar(50)	utf8mb4_general_ci		No	None			<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>
4	jumlah	bigint(20)			No	None			<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>
5	catatan	varchar(100)	utf8mb4_general_ci		No	None			<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>
6	tanggal	date			No	None			<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>

Check all With selected: Browse Change Drop Primary Unique Index Spat

Remove from central columns

New cangkir information\_schema mysql performance\_schema phpmyadmin saldocerdas pemasukan pengeluaran user

# DB USER

phpMyAdmin

Server: 127.0.0.1 » Database: saldocerdas » Table: user

Browse Structure SQL Search Insert Export Import Privileges Operations

Table structure Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	<b>iduser</b>	char(11)	utf8mb4_general_ci		No	None			Change  Drop
2	<b>username</b>	varchar(50)	utf8mb4_general_ci		No	None			Change  Drop
3	<b>email</b>	varchar(100)	utf8mb4_general_ci		No	None			Change  Drop
4	<b>tanggallahir</b>	date			No	None			Change  Drop
5	<b>pin</b>	varchar(4)	utf8mb4_general_ci		No	None			Change  Drop

Check all With selected: Primary Unique Index

New

cangkir

information\_schema

mysql

performance\_schema

phpmyadmin

saldocerdas

New

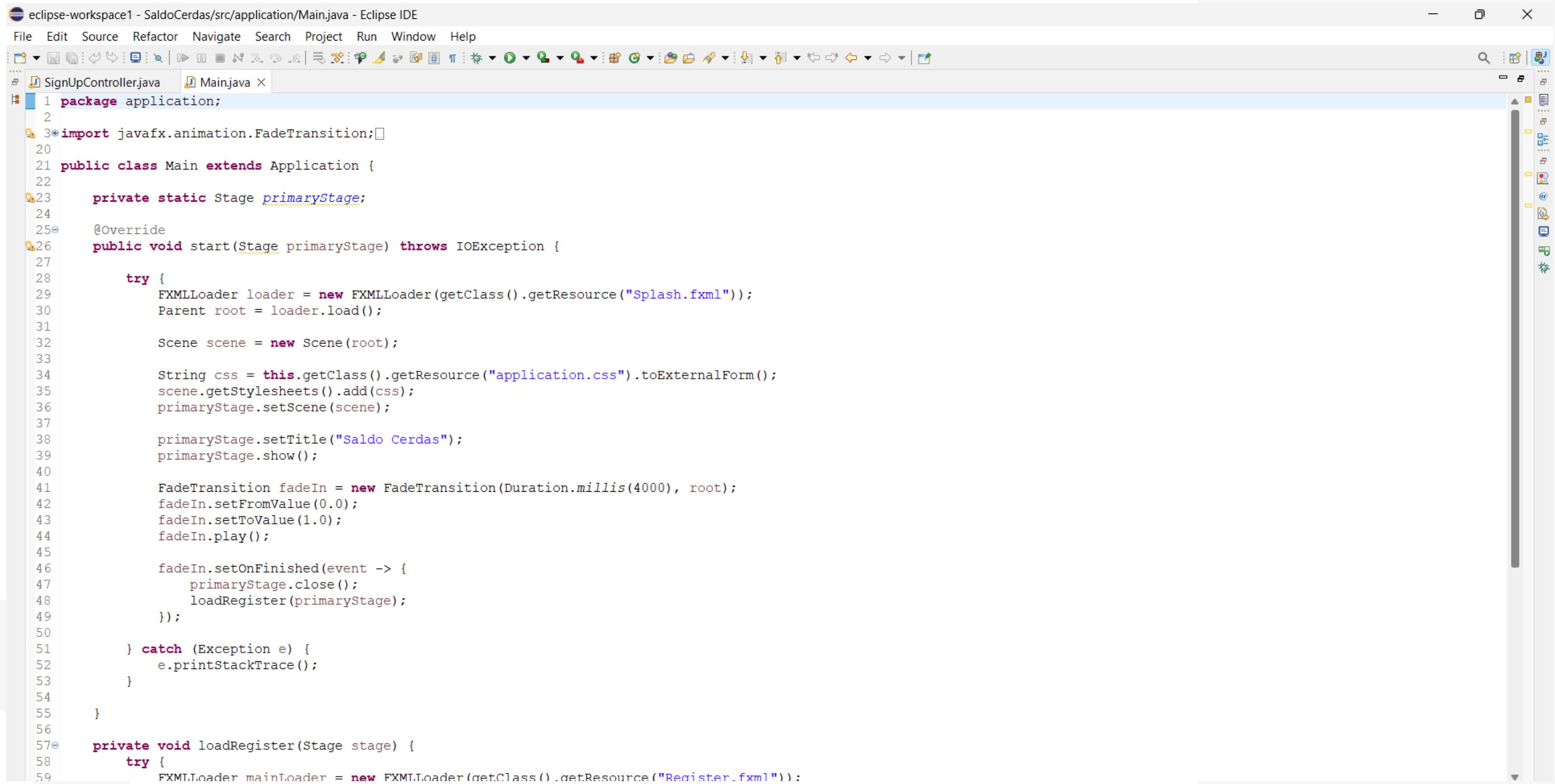
pemasukan

pengeluaran

user

# CODING DOCUMENTATION

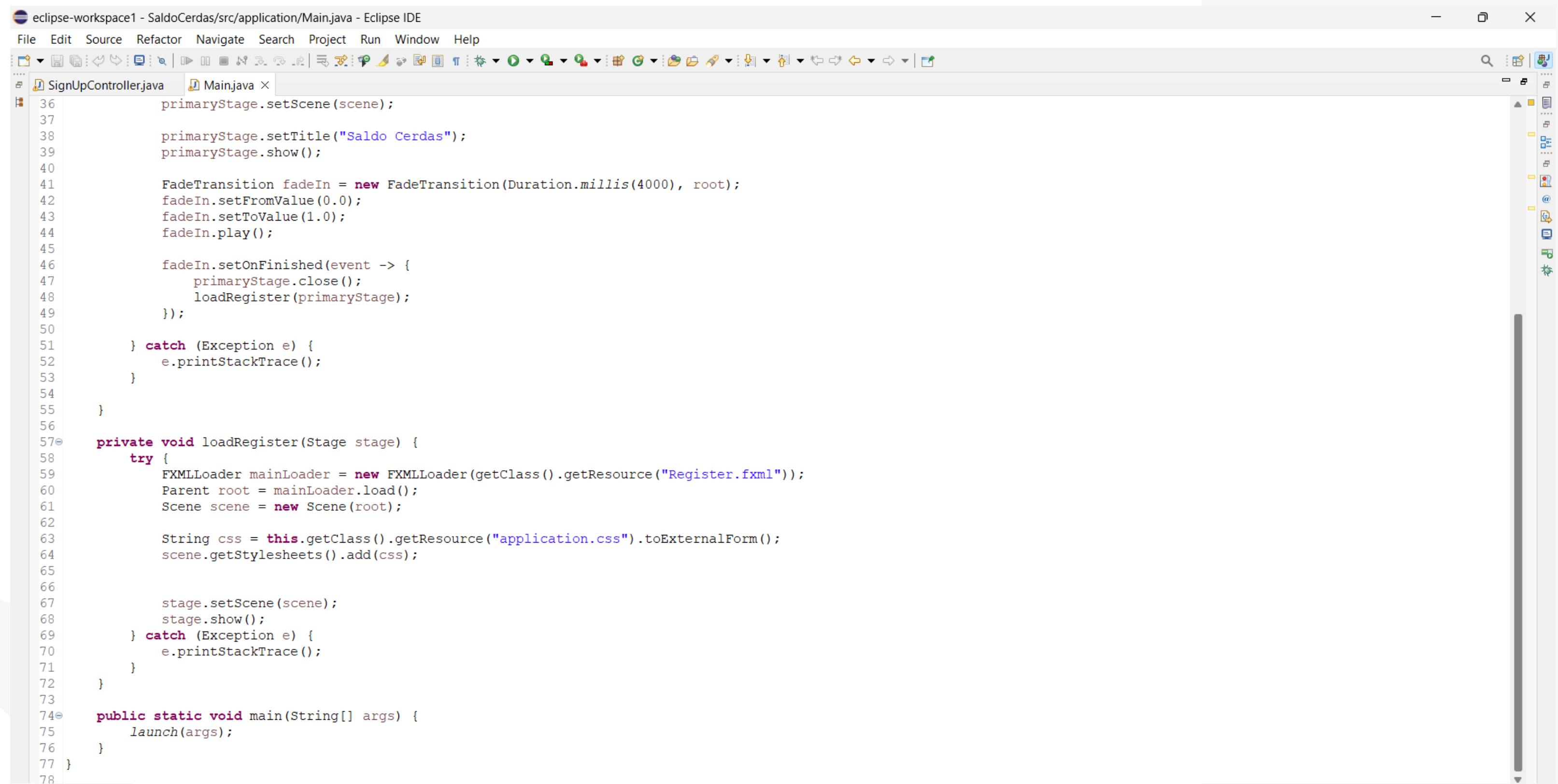
# JAVA (SPLASH SCREEN)



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/Main.java - Eclipse IDE". The main window displays the Java code for a splash screen application. The code uses JavaFX to load a FXML file ("Splash.fxml") and set it as the root node of a scene. It then applies a CSS style sheet ("application.css") and sets the stage title to "Saldo Cerdas". A fade transition is used to show the splash screen for 4000 milliseconds before closing the stage and loading a register screen. The code includes imports for JavaFX and the application class, and defines a Main class that extends Application.

```
1 package application;
2
3 import javafx.animation.FadeTransition;
4
5 public class Main extends Application {
6
7     private static Stage primaryStage;
8
9     @Override
10    public void start(Stage primaryStage) throws IOException {
11
12        try {
13            FXMLLoader loader = new FXMLLoader(getClass().getResource("Splash.fxml"));
14            Parent root = loader.load();
15
16            Scene scene = new Scene(root);
17
18            String css = this.getClass().getResource("application.css").toExternalForm();
19            scene.getStylesheets().add(css);
20            primaryStage.setScene(scene);
21
22            primaryStage.setTitle("Saldo Cerdas");
23            primaryStage.show();
24
25            FadeTransition fadeIn = new FadeTransition(Duration.millis(4000), root);
26            fadeIn.setFromValue(0.0);
27            fadeIn.setToValue(1.0);
28            fadeIn.play();
29
30            fadeIn.setOnFinished(event -> {
31                primaryStage.close();
32                loadRegister(primaryStage);
33            });
34
35        } catch (Exception e) {
36            e.printStackTrace();
37        }
38
39    }
40
41    private void loadRegister(Stage stage) {
42        try {
43            FXMLLoader mainLoader = new FXMLLoader(getClass().getResource("Register.fxml"));
44
```

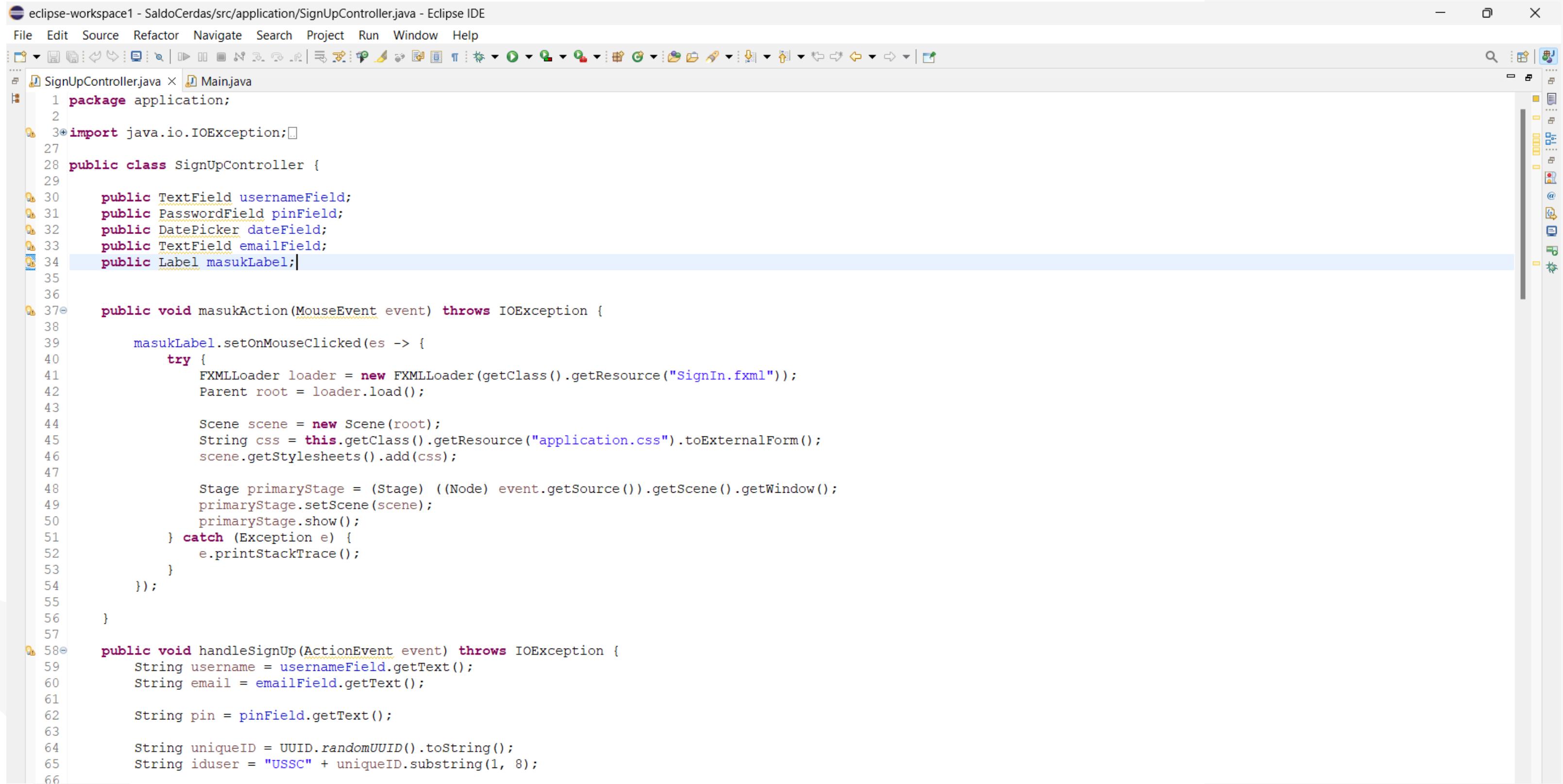
# JAVA (SPLASH SCREEN)



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/Main.java - Eclipse IDE". The main area displays Java code for a splash screen. The code uses JavaFX components like Stage, Scene, and FadeTransition. It includes exception handling and CSS loading. The code is organized into methods like loadRegister and main.

```
36     primaryStage.setScene(scene);
37
38     primaryStage.setTitle("Saldo Cerdas");
39     primaryStage.show();
40
41     FadeTransition fadeIn = new FadeTransition(Duration.millis(4000), root);
42     fadeIn.setFromValue(0.0);
43     fadeIn.setToValue(1.0);
44     fadeIn.play();
45
46     fadeIn.setOnFinished(event -> {
47         primaryStage.close();
48         loadRegister(primaryStage);
49     });
50
51     } catch (Exception e) {
52         e.printStackTrace();
53     }
54
55 }
56
57 private void loadRegister(Stage stage) {
58     try {
59         FXMLLoader mainLoader = new FXMLLoader(getClass().getResource("Register.fxml"));
60         Parent root = mainLoader.load();
61         Scene scene = new Scene(root);
62
63         String css = this.getClass().getResource("application.css").toExternalForm();
64         scene.getStylesheets().add(css);
65
66         stage.setScene(scene);
67         stage.show();
68     } catch (Exception e) {
69         e.printStackTrace();
70     }
71 }
72
73
74 public static void main(String[] args) {
75     launch(args);
76 }
77 }
78 }
```

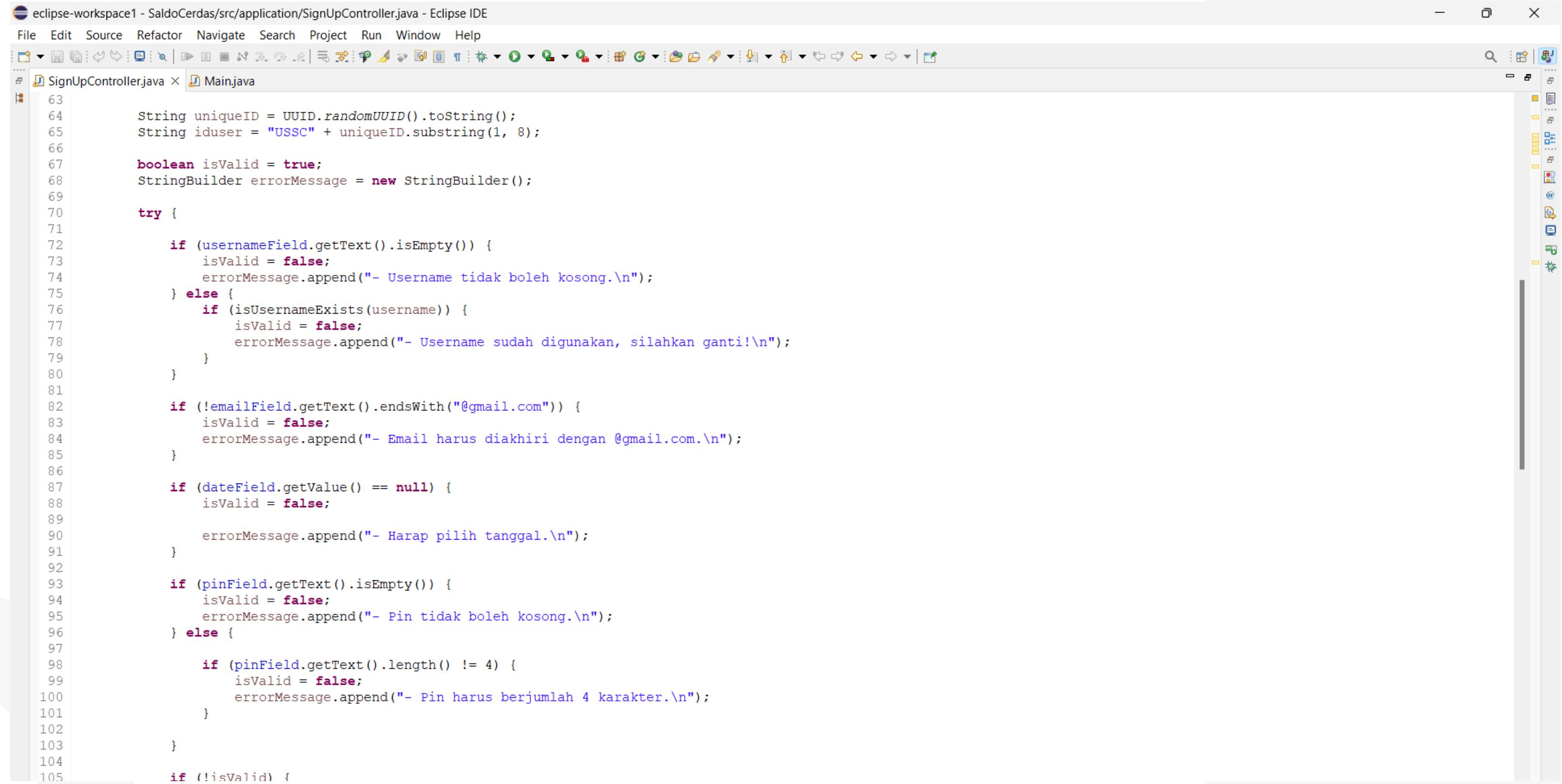
# JAVA (SIGN UP PAGE)



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/SignUpController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows the package structure with "application" selected, and the "Outline" view shows the class "SignUpController". The main editor area displays the Java code for "SignUpController.java".

```
1 package application;
2
3 import java.io.IOException;
4
5 public class SignUpController {
6
7     public TextField usernameField;
8     public PasswordField pinField;
9     public DatePicker dateField;
10    public TextField emailField;
11    public Label masukLabel;
12
13    public void masukAction(MouseEvent event) throws IOException {
14        masukLabel.setOnMouseClicked(es -> {
15            try {
16                FXMLLoader loader = new FXMLLoader(getClass().getResource("SignIn.fxml"));
17                Parent root = loader.load();
18
19                Scene scene = new Scene(root);
20                String css = this.getClass().getResource("application.css").toExternalForm();
21                scene.getStylesheets().add(css);
22
23                Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
24                primaryStage.setScene(scene);
25                primaryStage.show();
26            } catch (Exception e) {
27                e.printStackTrace();
28            }
29        });
30    }
31
32    public void handleSignUp(ActionEvent event) throws IOException {
33        String username = usernameField.getText();
34        String email = emailField.getText();
35
36        String pin = pinField.getText();
37
38        String uniqueID = UUID.randomUUID().toString();
39        String iduser = "USSC" + uniqueID.substring(1, 8);
40
41    }
42
43}
```

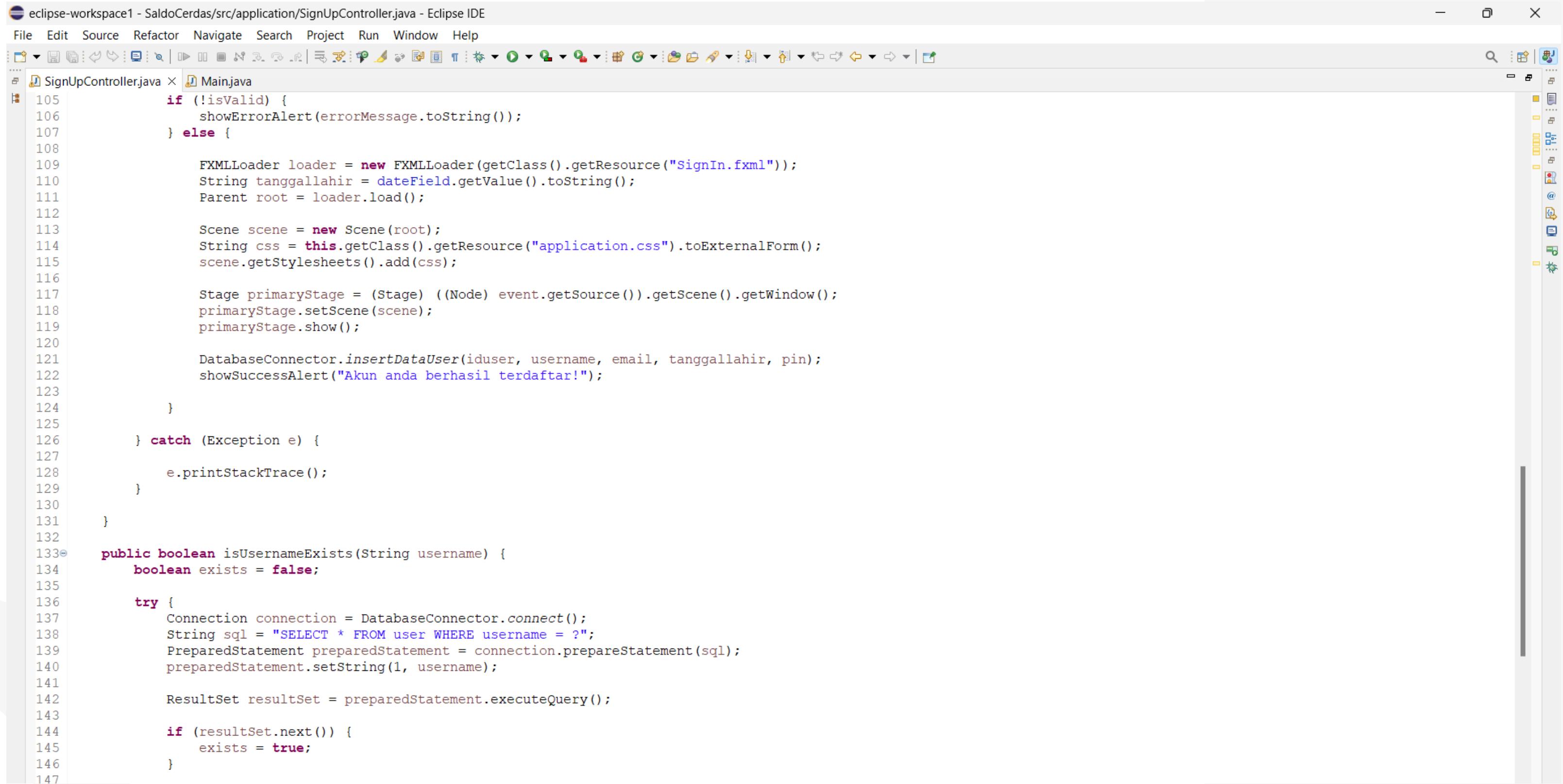
# JAVA (SIGN UP PAGE)



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/SignUpController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows the package structure under "SaldoCerdas" with "src" and "application" packages, and "SignUpController.java" is selected. The right sidebar contains various tool icons. The main editor area displays Java code for a sign-up controller:

```
63     String uniqueID = UUID.randomUUID().toString();
64     String iduser = "USSC" + uniqueID.substring(1, 8);
65
66     boolean isValid = true;
67     StringBuilder errorMessage = new StringBuilder();
68
69     try {
70
71         if (usernameField.getText().isEmpty()) {
72             isValid = false;
73             errorMessage.append("- Username tidak boleh kosong.\n");
74         } else {
75             if (isUsernameExists(username)) {
76                 isValid = false;
77                 errorMessage.append("- Username sudah digunakan, silahkan ganti!\n");
78             }
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) {
```

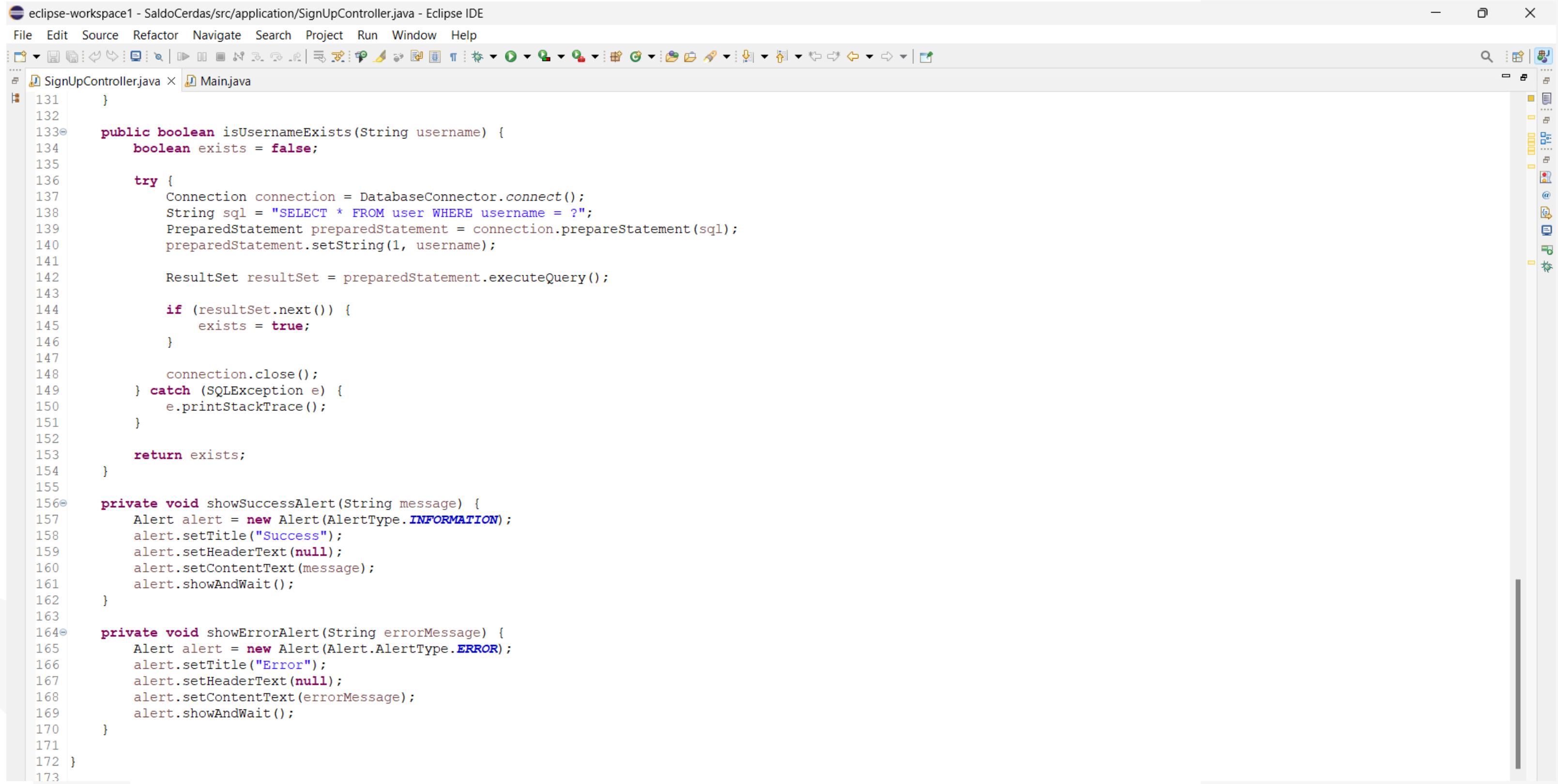
# JAVA (SIGN UP PAGE)



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/SignUpController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows the package structure with " SignUpController.java" selected. The main editor area contains Java code for a sign-up controller. The code handles user input validation, FXMLLoader loading, scene creation, stage setup, and database insertion. It also includes a method to check if a username exists in the database.

```
105     if (!isValid) {
106         showErrorMessage(errorMessage.toString());
107     } else {
108
109         FXMLLoader loader = new FXMLLoader(getClass().getResource("SignIn.fxml"));
110         String tanggallahir = dateField.getValue().toString();
111         Parent root = loader.load();
112
113         Scene scene = new Scene(root);
114         String css = this.getClass().getResource("application.css").toExternalForm();
115         scene.getStylesheets().add(css);
116
117         Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
118         primaryStage.setScene(scene);
119         primaryStage.show();
120
121         DatabaseConnector.insertDataUser(iduser, username, email, tanggallahir, pin);
122         showSuccessAlert("Akun anda berhasil terdaftar!");
123
124     }
125
126 } catch (Exception e) {
127
128     e.printStackTrace();
129 }
130
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     }
```

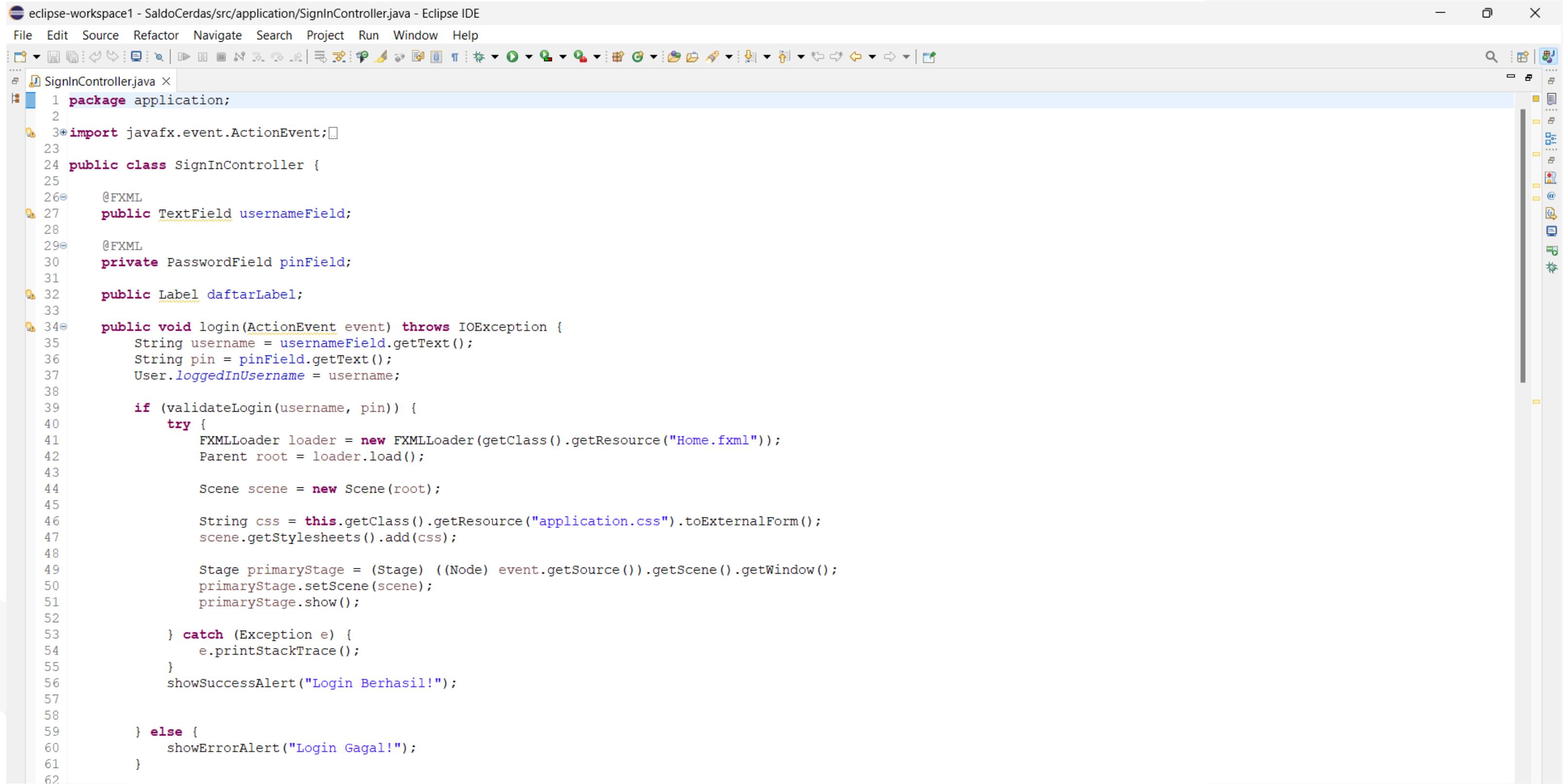
# JAVA (SIGN UP PAGE)



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/SignUpController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows the package structure under "SaldoCerdas" with "src" and "application" folders, and "SignUpController.java" is selected. The right sidebar contains various tool icons. The main editor area displays Java code for a SignUpController class:

```
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 }
```

# JAVA (SIGN IN PAGE)

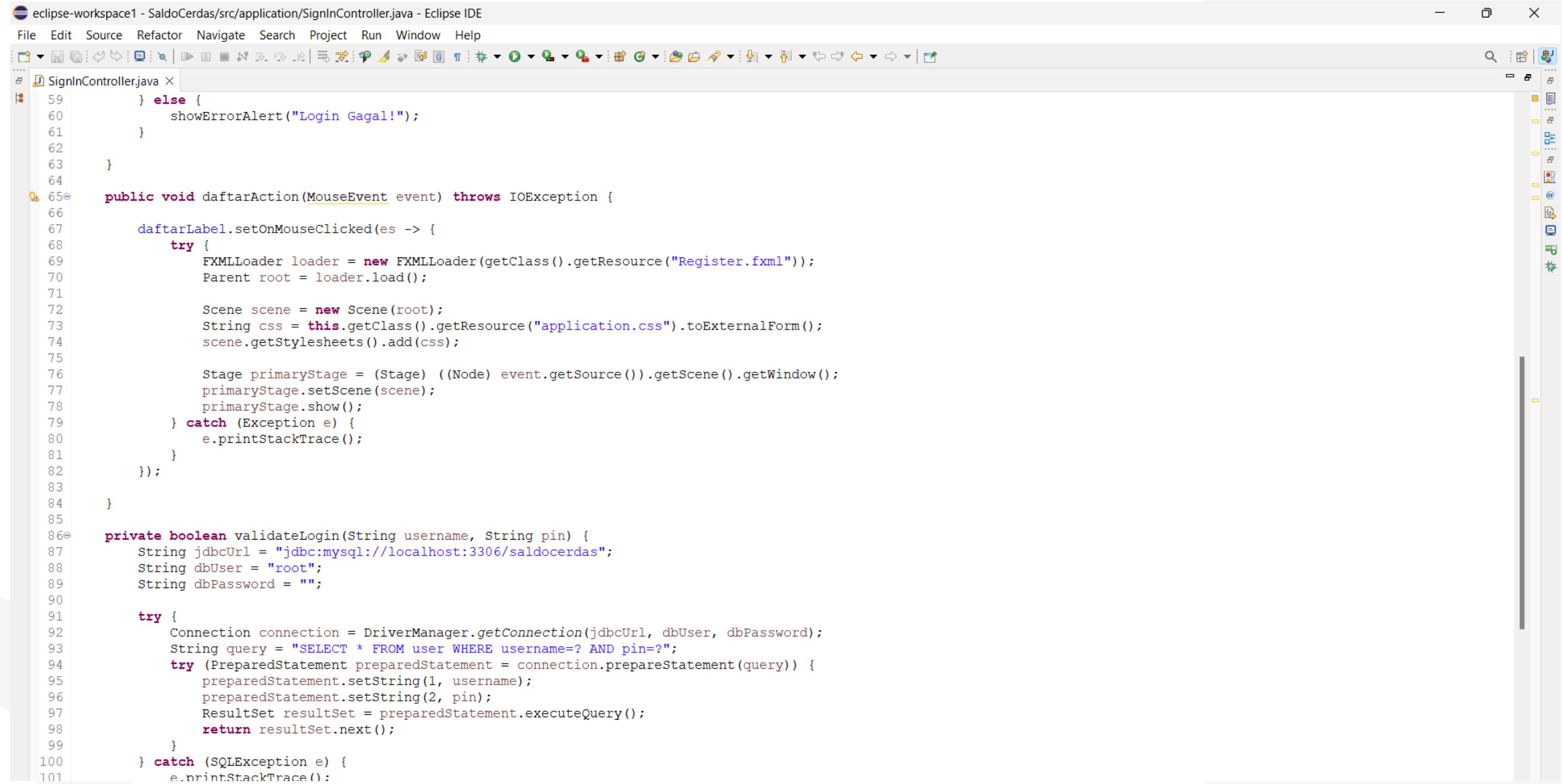


The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace1 - SaldoCerdas/src/application/SignInController.java - Eclipse IDE
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbar:** Standard Eclipse toolbar with various icons for file operations, search, and project management.
- Left Margin:** Line numbers from 1 to 62 on the left side of the code editor.
- Code Editor:** The main area displays Java code for a controller class named `SignInController`. The code includes imports for `javafx.event.ActionEvent`, defines fields for `usernameField` and `pinField` (both annotated with `@FXML`), and implements a `login` method that handles user input and navigates to a new scene. Error handling is provided for potential exceptions.
- Right Margin:** A vertical bar containing several small icons representing different Eclipse features or perspectives.

```
1 package application;
2
3 import javafx.event.ActionEvent;
4
5 public class SignInController {
6
7     @FXML
8     public TextField usernameField;
9
10    @FXML
11    private PasswordField pinField;
12
13    public Label daftarLabel;
14
15    public void login(ActionEvent event) throws IOException {
16        String username = usernameField.getText();
17        String pin = pinField.getText();
18        User.loggedInUsername = username;
19
20        if (validateLogin(username, pin)) {
21            try {
22                FXMLLoader loader = new FXMLLoader(getClass().getResource("Home.fxml"));
23                Parent root = loader.load();
24
25                Scene scene = new Scene(root);
26
27                String css = this.getClass().getResource("application.css").toExternalForm();
28                scene.getStylesheets().add(css);
29
30                Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
31                primaryStage.setScene(scene);
32                primaryStage.show();
33
34            } catch (Exception e) {
35                e.printStackTrace();
36            }
37            showSuccessAlert("Login Berhasil!");
38
39        } else {
40            showErrorAlert("Login Gagal!");
41        }
42
43    }
44
45    private void showSuccessAlert(String message) {
46        Alert successAlert = new Alert(AlertType.INFORMATION);
47        successAlert.setTitle("Success");
48        successAlert.setHeaderText(null);
49        successAlert.setContentText(message);
50        successAlert.showAndWait();
51
52    }
53
54    private void showErrorAlert(String message) {
55        Alert errorAlert = new Alert(AlertType.ERROR);
56        errorAlert.setTitle("Error");
57        errorAlert.setHeaderText(null);
58        errorAlert.setContentText(message);
59        errorAlert.showAndWait();
60
61    }
62}
```

# JAVA (SIGN IN PAGE)

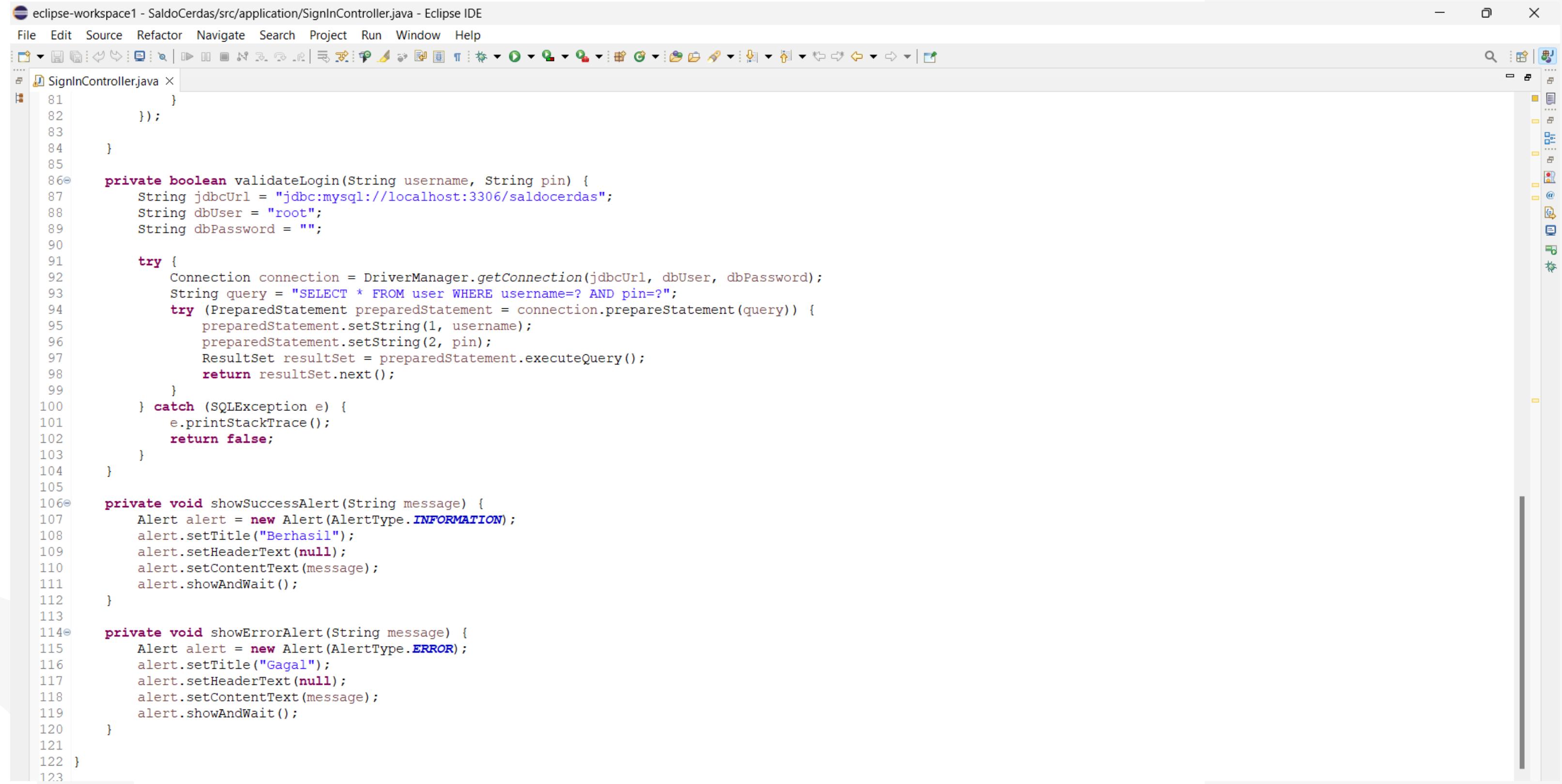


The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace1 - SaldoCerdas/src/application/SignInController.java - Eclipse IDE
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbar:** Standard Eclipse toolbar icons.
- Left Margin:** Line numbers from 59 to 101.
- Code Editor:** The file contains Java code for a sign-in controller. It includes methods for handling login attempts and displaying error alerts, as well as logic for opening a registration FXML window when the "daftar" button is clicked. It also includes a private method for validating the database connection and executing a SQL query to check if a user exists.

```
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();
102    }
103 }
```

# JAVA (SIGN IN PAGE)



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/SignInController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows the package structure under "SaldoCerdas". The main editor window displays Java code for a "SignInController.java" class. The code handles user login validation and alert message display.

```
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();
102        return false;
103    }
104 }
105
106 private void showSuccessAlert(String message) {
107     Alert alert = new Alert(AlertType.INFORMATION);
108     alert.setTitle("Berhasil");
109     alert.setHeaderText(null);
110     alert.setContentText(message);
111     alert.showAndWait();
112 }
113
114 private void showErrorAlert(String message) {
115     Alert alert = new Alert(AlertType.ERROR);
116     alert.setTitle("Gagal");
117     alert.setHeaderText(null);
118     alert.setContentText(message);
119     alert.showAndWait();
120 }
121
122 }
```

# JAVA (HOME PAGE)

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/HomeController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows the project structure with files like SignInController.java and HomeController.java. The main editor area displays the Java code for HomeController.java, which defines a controller class for a home page. The code includes imports for javafx.scene.Node, javafx.scene.control.TableColumn, javafx.scene.control.TableView, and javafx.collections.ObservableList. It defines private fields for tables (pemasukanTable, pengeluaranTable) and columns (kategoriPemasukan, jumlahPemasukan, catatanPemasukan, tanggalPemasukan for pemasukan; kategoriPengeluaran, jumlahPengeluaran, catatanPengeluaran, tanggalPengeluaran for pengeluaran). It also defines public methods for deleting items from both tables. The initialize() method setsCellValueFactories for all columns. A User object is instantiated. The code ends with setting dataPemasukan to an observable array list and pemasukanTable.setItems(dataPemasukan). A status bar at the bottom right indicates "Updates Available" with the message "Software updates have been downloaded. Click to review and install updates".

```
1 package application;
2
3 import javafx.scene.Node;
4
5 public class HomeController {
6
7     @FXML
8     public Label usernameLabel;
9     private ObservableList<Pemasukan> dataPemasukan;
10    private ObservableList<Pengeluaran> dataPengeluaran;
11
12    public TableView<Pemasukan> pemasukanTable;
13    public TableColumn<Pemasukan, String> kategoriPemasukan;
14    public TableColumn<Pemasukan, Integer> jumlahPemasukan;
15    public TableColumn<Pemasukan, String> catatanPemasukan;
16    public TableColumn<Pemasukan, String> tanggalPemasukan;
17
18    public TableView<Pengeluaran> pengeluaranTable;
19    public TableColumn<Pengeluaran, String> kategoriPengeluaran;
20    public TableColumn<Pengeluaran, Integer> jumlahPengeluaran;
21    public TableColumn<Pengeluaran, String> catatanPengeluaran;
22    public TableColumn<Pengeluaran, String> tanggalPengeluaran;
23
24    public Button deletePemasukan;
25    public Button deletePengeluaran;
26
27    User user = new User();
28
29    @FXML
30    public void initialize() {
31        kategoriPemasukan.setCellValueFactory(new PropertyValueFactory<>("kategori"));
32        jumlahPemasukan.setCellValueFactory(new PropertyValueFactory<>("jumlah"));
33        catatanPemasukan.setCellValueFactory(new PropertyValueFactory<>("catatan"));
34        tanggalPemasukan.setCellValueFactory(new PropertyValueFactory<>("tanggal"));
35
36        kategoriPengeluaran.setCellValueFactory(new PropertyValueFactory<>("kategori"));
37        jumlahPengeluaran.setCellValueFactory(new PropertyValueFactory<>("jumlah"));
38        catatanPengeluaran.setCellValueFactory(new PropertyValueFactory<>("catatan"));
39        tanggalPengeluaran.setCellValueFactory(new PropertyValueFactory<>("tanggal"));
40
41        dataPemasukan = FXCollections.observableArrayList();
42        pemasukanTable.setItems(dataPemasukan);
43    }
44}
```

# JAVA (HOME PAGE)

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/HomeController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows the package structure with "SignInController.java" and "HomeController.java" selected. The main editor area displays the Java code for "HomeController.java". The code includes methods for setting up tables, displaying data, and performing database queries to retrieve spending data from a MySQL database. A status bar at the bottom right indicates "Updates Available" with the message "Software updates have been downloaded. Click to review and install updates".

```
72     dataPemasukan = FXCollections.observableArrayList();
73     pemasukanTable.setItems(dataPemasukan);
74
75     dataPengeluaran = FXCollections.observableArrayList();
76     pengeluaranTable.setItems(dataPengeluaran);
77
78     showPemasukanData();
79     showPengeluaranData();
80     deletePemasukanAction();
81     deletePengeluaranAction();
82
83
84     usernameLabel.setText(""+ user.getUsername());
85 }
86
87
88
89
90 public void showPemasukanData() {
91     getPemasukanData();
92     pemasukanTable.setItems(dataPemasukan);
93 }
94
95
96 private void getPemasukanData() {
97     String url = "jdbc:mysql://localhost:3306/saldocerdas";
98     String username = "root";
99     String password = "";
100
101    try {
102        Connection connection = DriverManager.getConnection(url, username, password);
103        String query = "SELECT * FROM pemasukan WHERE iduser = " + ""+ user.getIduser()+"";
104        Statement statement = connection.createStatement();
105        ResultSet resultSet = statement.executeQuery(query);
106
107        while (resultSet.next()) {
108            String kategori = resultSet.getString("kategori");
109            int jumlah = resultSet.getInt("jumlah");
110            String catatan = resultSet.getString("catatan");
111            String tanggal = resultSet.getString("tanggal");
112            String idpemasukan = resultSet.getString("idpemasukan");
113
114            Pemasukan pemasukan = new Pemasukan(kategori, jumlah, catatan, tanggal, idpemasukan);
```

# JAVA (HOME PAGE)

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/HomeController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows two open files: "SignInController.java" and "HomeController.java", with "HomeController.java" currently selected. The main editor area displays the Java code for "HomeController.java". The code handles the addition of new entries to a database and retrieves spending data for a user. A try-catch block manages database connections and exceptions. A message box at the bottom right indicates "Updates Available" with the message "Software updates have been downloaded. Click to review and install updates".

```
114         Pemasukan pemasukan = new Pemasukan(kategori, jumlah, catatan, tanggal, idpemasukan);
115         dataPemasukan.add(pemasukan);
116     }
117
118     connection.close();
119 } catch (SQLException e) {
120     e.printStackTrace();
121 }
122 }
123
124 public void showPengeluaranData() {
125     getPengeluaranData();
126     pengeluaranTable.setItems(dataPengeluaran);
127 }
128
129 private void getPengeluaranData() {
130     String url = "jdbc:mysql://localhost:3306/saldocerdas";
131     String username = "root";
132     String password = "";
133
134     try {
135         Connection connection = DriverManager.getConnection(url, username, password);
136         String query = "SELECT * FROM pengeluaran WHERE iduser = " + "" + user.getIduser() + "";
137         Statement statement = connection.createStatement();
138         ResultSet resultSet = statement.executeQuery(query);
139
140         while (resultSet.next()) {
141
142             String kategori = resultSet.getString("kategori");
143             String idUser = resultSet.getString("iduser");
144             int jumlah = resultSet.getInt("jumlah");
145             String catatan = resultSet.getString("catatan");
146             String tanggal = resultSet.getString("tanggal");
147             String idpengeluaran = resultSet.getString("idpengeluaran");
148
149             Pengeluaran pengeluaran = new Pengeluaran(kategori, idUser, jumlah, catatan, tanggal, idpengeluaran);
150             dataPengeluaran.add(pengeluaran);
151         }
152
153         connection.close();
154     } catch (SQLException e) {
155         e.printStackTrace();
156     }
}
```

# JAVA (HOME PAGE)

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/HomeController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows two open files: "SignInController.java" and "HomeController.java", with "HomeController.java" currently selected. The main editor area displays Java code for the HomeController class, which handles actions for deleting entries from tables and navigating to a FXMLLoader. The code uses try-catch blocks, JDBC connections, and JavaFX components like TableView and Scene. A status bar at the bottom right indicates "Updates Available" with the message "Software updates have been downloaded. Click to review and install updates".

```
153         connection.close();
154     } catch (SQLException e) {
155         e.printStackTrace();
156     }
157 }
158
159 public void deletePemasukanAction() {
160     Pemasukan selectedPemasukan = pemasukanTable.getSelectionModel().getSelectedItem();
161
162     if (selectedPemasukan != null) {
163         DatabaseConnector.hapusPemasukan(selectedPemasukan.getIdpemasukan());
164
165         pemasukanTable.getItems().remove(selectedPemasukan);
166         showSuccessAlert("Delete data pemasukan berhasil");
167         pemasukanTable.refresh();
168     }
169 }
170
171
172 public void deletePengeluaranAction() {
173     Pengeluaran selectedPengeluaran = pengeluaranTable.getSelectionModel().getSelectedItem();
174
175     if (selectedPengeluaran != null) {
176         DatabaseConnector.hapusPengeluaran(selectedPengeluaran.getIdpengeluaran());
177
178         pengeluaranTable.getItems().remove(selectedPengeluaran);
179         showSuccessAlert("Delete data pengeluaran berhasil");
180         pengeluaranTable.refresh();
181     }
182 }
183
184
185 public void pengeluaranAction(ActionEvent event) throws IOException {
186
187     try {
188         FXMLLoader loader = new FXMLLoader(getClass().getResource("Pengeluaran.fxml"));
189         Parent root = loader.load();
190
191         Scene scene = new Scene(root);
192         String css = this.getClass().getResource("application.css").toExternalForm();
193         scene.getStylesheets().add(css);
194
195         Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
```

# JAVA (HOME PAGE)

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/HomeController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows two files: "SignInController.java" and "HomeController.java", with "HomeController.java" currently selected. The main editor area displays Java code for the HomeController class, which handles action events for "pemasukanAction" and "analisisAction". The code uses FXMLLoader to load FXML files ("Pemasukan.fxml" and "Analisis.fxml") and sets them as scenes for a primary stage. Error markers are visible on lines 203, 219, 221, and 237. A status bar at the bottom right indicates "Updates Available" with the message "Software updates have been downloaded. Click to review and install updates".

```
195     Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
196     primaryStage.setScene(scene);
197     primaryStage.show();
198 } catch (Exception e) {
199     e.printStackTrace();
200 }
201 }
202
203 public void pemasukanAction(ActionEvent event) throws IOException {
204
205     try {
206         FXMLLoader loader = new FXMLLoader(getClass().getResource("Pemasukan.fxml"));
207         Parent root = loader.load();
208
209         Scene scene = new Scene(root);
210         String css = this.getClass().getResource("application.css").toExternalForm();
211         scene.getStylesheets().add(css);
212
213         Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
214         primaryStage.setScene(scene);
215         primaryStage.show();
216     } catch (Exception e) {
217         e.printStackTrace();
218     }
219 }
220
221 public void analisisAction(ActionEvent event) throws IOException {
222
223     try {
224         FXMLLoader loader = new FXMLLoader(getClass().getResource("Analisis.fxml"));
225         Parent root = loader.load();
226
227         Scene scene = new Scene(root);
228         String css = this.getClass().getResource("application.css").toExternalForm();
229         scene.getStylesheets().add(css);
230
231         Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
232         primaryStage.setScene(scene);
233         primaryStage.show();
234     } catch (Exception e) {
235         e.printStackTrace();
236     }
237 }
```

# JAVA (HOME PAGE)

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/HomeController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows two files: "SignInController.java" and "HomeController.java", with "HomeController.java" selected. The main editor area displays the Java code for "HomeController.java". The code handles button events to load FXML files ("Analisis.fxml" and "Profile.fxml") into scenes and stages. It also contains a private method "showSuccessAlert" to display an information alert. The right sidebar contains a "Project Explorer" view with several Java files listed. A small "Updates Available" dialog is visible in the bottom right corner.

```
223     try {
224         FXMLLoader loader = new FXMLLoader(getClass().getResource("Analisis.fxml"));
225         Parent root = loader.load();
226
227         Scene scene = new Scene(root);
228         String css = this.getClass().getResource("application.css").toExternalForm();
229         scene.getStylesheets().add(css);
230
231         Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
232         primaryStage.setScene(scene);
233         primaryStage.show();
234     } catch (Exception e) {
235         e.printStackTrace();
236     }
237 }
238
239 public void profileAction(ActionEvent event) throws IOException {
240
241     try {
242         FXMLLoader loader = new FXMLLoader(getClass().getResource("Profile.fxml"));
243         Parent root = loader.load();
244
245         Scene scene = new Scene(root);
246         String css = this.getClass().getResource("application.css").toExternalForm();
247         scene.getStylesheets().add(css);
248
249         Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
250         primaryStage.setScene(scene);
251         primaryStage.show();
252     } catch (Exception e) {
253         e.printStackTrace();
254     }
255 }
256
257 private void showSuccessAlert(String message) {
258     Alert alert = new Alert(AlertType.INFORMATION);
259     alert.setTitle("Success");
260     alert.setHeaderText(null);
261     alert.setContentText(message);
262     alert.showAndWait();
263 }
264 }
```

Updates Available

Software updates have been downloaded.  
Click to review and install updates.

# JAVA (ANALISIS PAGE)

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/AnalisisController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Find, and Run. The left sidebar shows project files: SignInController.java, HomeController.java, and AnalisisController.java (which is currently selected). The right sidebar contains various tool icons. The main editor area displays the Java code for AnalisisController.java:

```
1 package application;
2
3 import java.io.IOException;
4
5 public class AnalisisController {
6
7     public PieChart pieChart;
8     public PieChart pieChart1;
9
10    public Label pemasukanTotal;
11    public Label pengeluaranTotal;
12    public Label totalSemua;
13    public Label statusKeuangan;
14
15    User user = new User();
16
17    private static final String DB_URL = "jdbc:mysql://localhost:3306/saldocerdas";
18    private static final String DB_USER = "root";
19    private static final String DB_PASSWORD = "";
20
21    public void initialize() {
22        updatePieChart();
23        updatePieCharts();
24        totalKeseluruhan();
25    }
26
27    private ObservableList<PieChart.Data> getDataFromDatabase() {
28        ObservableList<PieChart.Data> pieChartData = FXCollections.observableArrayList();
29        double totalAmount = 0.0;
30        Set<String> uniqueCategories = new HashSet<>();
31
32        try (Connection connection = DriverManager.getConnection(DB_URL, DB_USER, DB_PASSWORD)) {
33            String query = "SELECT kategori, jumlah FROM pemasukan WHERE iduser = " + "''" + user.getIduser() + "'";
34            try (PreparedStatement preparedStatement = connection.prepareStatement(query)) {
35                ResultSet resultSet = preparedStatement.executeQuery();
36                while (resultSet.next()) {
37                    String category = resultSet.getString("kategori");
38
39                    if (uniqueCategories.add(category)) {
40                        double amount = resultSet.getDouble("jumlah");
41                        totalAmount += amount;
42                        pieChartData.add(new PieChart.Data(category, amount));
43                    }
44                }
45            }
46        } catch (SQLException e) {
47            e.printStackTrace();
48        }
49    }
50}
```

A small "Updates Available" dialog box is visible in the bottom right corner, stating "Software updates have been downloaded. Click to review and install updates."

# JAVA (ANALISIS PAGE)

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/AnalysisController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows three open files: SignInController.java, HomeController.java, and AnalysisController.java (the active tab). The right sidebar contains various tool icons. A status bar at the bottom right displays "Updates Available" and "Software updates have been downloaded. Click to review and install updates." The main code editor area contains the following Java code:

```
72             pieChartData.add(new PieChart.Data(category, amount));
73         }
74     }
75 }
76 } catch (SQLException e) {
77     e.printStackTrace();
78 }
79
80 for (PieChart.Data data : pieChartData) {
81     double percentage = (data.getPieValue() / totalAmount) * 100;
82     String originalLabel = data.getName();
83     data.setName(originalLabel + " (" + String.format("%.2f%%", percentage) + ")");
84
85 }
86 return pieChartData;
87 }
88
89
90 private ObservableList<PieChart.Data> getDataFromDatabases() {
91     ObservableList<PieChart.Data> pieChartDatas = FXCollections.observableArrayList();
92     double totalAmount = 0.0;
93     Set<String> uniqueCategories = new HashSet<>();
94
95     try (Connection connection = DriverManager.getConnection(DB_URL, DB_USER, DB_PASSWORD)) {
96         String query = "SELECT kategori, jumlah FROM pengeluaran WHERE iduser = " + "" + user.getIduser() + "";
97         try (PreparedStatement preparedStatement = connection.prepareStatement(query);
98             ResultSet resultSet = preparedStatement.executeQuery()) {
99             while (resultSet.next()) {
100                 String category = resultSet.getString("kategori");
101
102                 if (uniqueCategories.add(category)) {
103                     double amount = resultSet.getDouble("jumlah");
104                     totalAmount += amount;
105                     pieChartDatas.add(new PieChart.Data(category, amount));
106                 }
107             }
108         }
109     } catch (SQLException e) {
110         e.printStackTrace();
111     }
112
113     for (PieChart.Data data : pieChartDatas) {
```

# JAVA (ANALISIS PAGE)

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/AnalysisController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Find, and Run. The left sidebar shows three open files: SignInController.java, HomeController.java, and AnalysisController.java (the active tab). The right sidebar contains toolbars for code navigation, search, and project management. A status bar at the bottom right displays "Updates Available" and "Software updates have been downloaded. Click to review and install updates".

```
114     for (PieChart.Data data : pieChartData) {
115         double percentage = (data.getPieValue() / totalAmount) * 100;
116         String originalLabel = data.getName();
117         data.setName(originalLabel + " (" + String.format("%.2f%%", percentage) + ")");
118     }
119
120     return pieChartData;
121 }
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)

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/AnalisisController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows project files: SignInController.java, HomeController.java, and AnalisisController.java (which is currently selected). The right sidebar contains various tool icons. The main editor area displays Java code for the AnalisisController.java class. The code performs database queries to calculate total income and expenses for a user, then determines the financial status (SURPLUS or DEFISIT) based on the difference. It also handles SQL exceptions. The code is annotated with line numbers from 156 to 198. A tooltip "Updates Available" is visible in the bottom right corner.

```
156     ResultSet resultSetPemasukan = statement
157         .executeQuery("SELECT SUM(jumlah) FROM pemasukan WHERE iduser = " + "" + user.getIduser() + "");
158     double totalPemasukan = 0;
159     if (resultSetPemasukan.next()) {
160         totalPemasukan = resultSetPemasukan.getDouble(1);
161         pemasukanTotal.setText("Rp " + totalPemasukan);
162     }
163     resultSetPemasukan.close();
164
165     ResultSet resultSetPengeluaran = statement
166         .executeQuery("SELECT SUM(jumlah) FROM pengeluaran WHERE iduser = " + "" + user.getIduser() + "");
167     double totalPengeluaran = 0;
168     if (resultSetPengeluaran.next()) {
169         totalPengeluaran = resultSetPengeluaran.getDouble(1);
170         pengeluaranTotal.setText("Rp " + totalPengeluaran);
171     }
172     resultSetPengeluaran.close();
173
174     double selisih = totalPemasukan - totalPengeluaran;
175
176     totalSemua.setText("Rp " + selisih);
177
178     if (totalPemasukan > totalPengeluaran) {
179         statusKeuangan.setText("SURPLUS");
180         statusKeuangan.setStyle("-fx-text-fill : #2ebf30");
181     } else if (totalPengeluaran > totalPemasukan) {
182         statusKeuangan.setText("DEFISIT");
183         statusKeuangan.setStyle("-fx-text-fill : #e40000");
184     } else {
185         statusKeuangan.setText("");
186     }
187
188     statement.close();
189     connection.close();
190 } catch (SQLException e) {
191     e.printStackTrace();
192 }
193
194 }
195
196 public void HomeAction(ActionEvent event) throws IOException {
197     trv {
198 }
```

Updates Available

Software updates have been downloaded.  
Click to review and install updates.

# JAVA (ANALISIS PAGE)

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/AnalisisController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows three open files: SignInController.java, HomeController.java, and AnalisisController.java (the active tab). The right sidebar contains a tree view of project files and an "Updates Available" notification. The main editor area displays Java code for the AnalisisController.java class, which handles actions for inserting and withdrawing funds. The code uses FXMLLoader to load FXML files and SceneBuilder to define scenes and stylesheets.

```
198     try {
199         FXMLLoader loader = new FXMLLoader(getClass().getResource("Home.fxml"));
200         Parent root = loader.load();
201
202         Scene scene = new Scene(root);
203         String css = this.getClass().getResource("application.css").toExternalForm();
204         scene.getStylesheets().add(css);
205
206         Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
207         primaryStage.setScene(scene);
208         primaryStage.show();
209     } catch (Exception e) {
210         e.printStackTrace();
211     }
212
213 }
214
215 public void pemasukanAction(ActionEvent event) throws IOException {
216
217     try {
218         FXMLLoader loader = new FXMLLoader(getClass().getResource("Pemasukan.fxml"));
219         Parent root = loader.load();
220
221         Scene scene = new Scene(root);
222         String css = this.getClass().getResource("application.css").toExternalForm();
223         scene.getStylesheets().add(css);
224
225         Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
226         primaryStage.setScene(scene);
227         primaryStage.show();
228     } catch (Exception e) {
229         e.printStackTrace();
230     }
231
232 }
233
234 public void pengeluaranAction(ActionEvent event) throws IOException {
235
236     try {
237         FXMLLoader loader = new FXMLLoader(getClass().getResource("Pengeluaran.fxml"));
238         Parent root = loader.load();
239
240         Scene scene = new Scene(root);
```

Updates Available

Software updates have been downloaded.  
Click to review and install updates.

# JAVA (ANALISIS PAGE)

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/AnalysisController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows three open files: SignInController.java, HomeController.java, and AnalysisController.java (the active tab). The main editor area displays the Java code for AnalysisController.java, which handles FXMLLoader loading and Scene setup for primary stages. A status bar at the bottom right indicates "Updates Available" with the message "Software updates have been downloaded. Click to review and install updates".

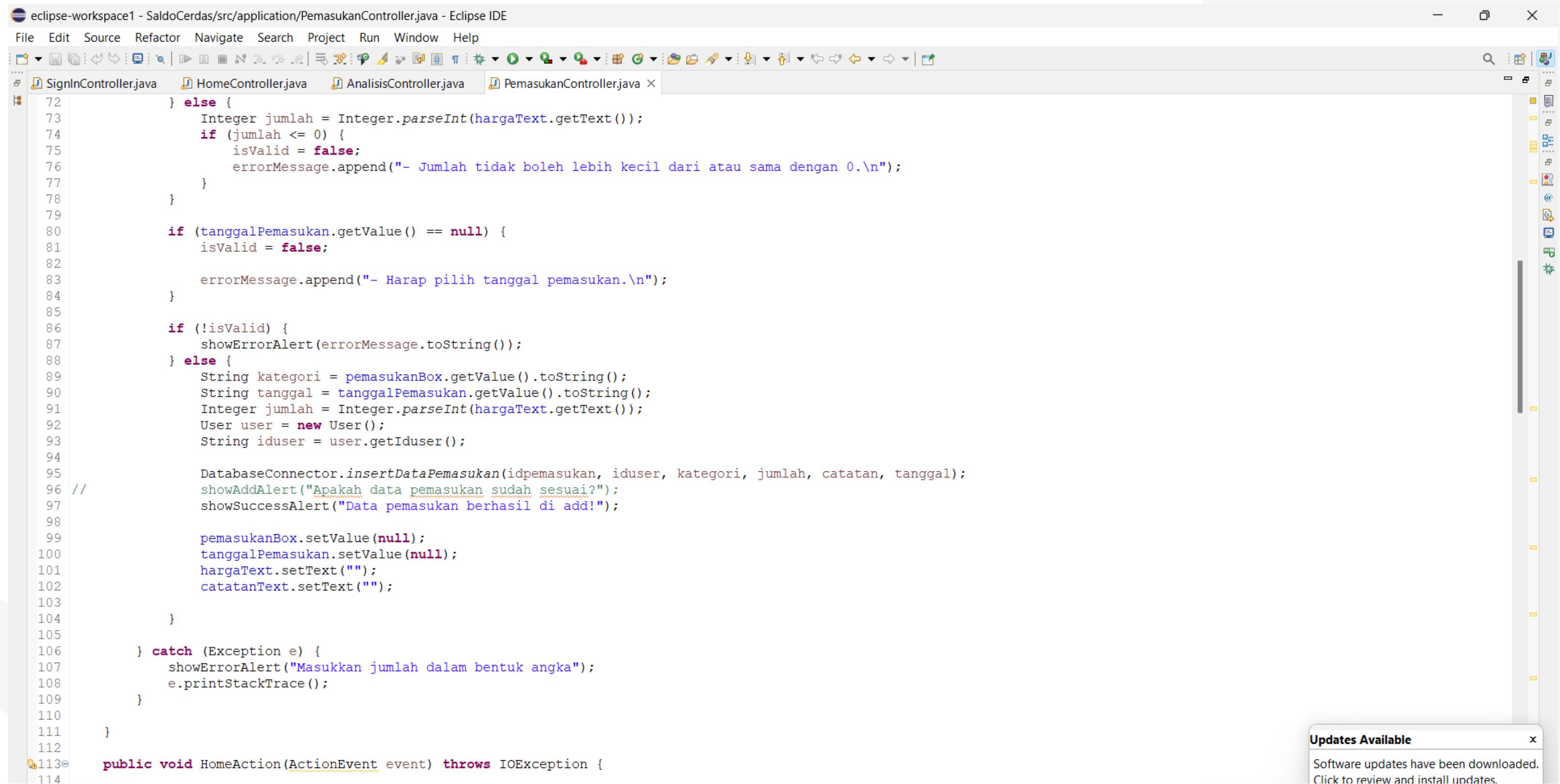
```
238     Parent root = loader.load();
239
240     Scene scene = new Scene(root);
241     String css = this.getClass().getResource("application.css").toExternalForm();
242     scene.getStylesheets().add(css);
243
244     Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
245     primaryStage.setScene(scene);
246     primaryStage.show();
247 } catch (Exception e) {
248     e.printStackTrace();
249 }
250
251 }
252
253 public void profileAction(ActionEvent event) throws IOException {
254
255     try {
256         FXMLLoader loader = new FXMLLoader(getClass().getResource("Profile.fxml"));
257         Parent root = loader.load();
258
259         Scene scene = new Scene(root);
260         String css = this.getClass().getResource("application.css").toExternalForm();
261         scene.getStylesheets().add(css);
262
263         Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
264         primaryStage.setScene(scene);
265         primaryStage.show();
266     } catch (Exception e) {
267         e.printStackTrace();
268     }
269 }
270
271 private void showSuccessAlert(String message) {
272     Alert alert = new Alert(AlertType.INFORMATION);
273     alert.setTitle("Info Data");
274     alert.setHeaderText(null);
275     alert.setContentText(message);
276     alert.showAndWait();
277 }
278
279 }
```

# JAVA (PEMASUKAN PAGE)

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/PemasukanController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Find, and Run. The left sidebar shows project files: SignInController.java, HomeController.java, AnalisisController.java, and PemasukanController.java (which is currently selected). The main editor area contains the Java code for PemasukanController.java, which handles input validation and database insertion. A status bar at the bottom right indicates "Updates Available" with the message "Software updates have been downloaded. Click to review and install updates".

```
1 package application;
2
3 import javafx.application.Application;
4
5 public class PemasukanController implements Initializable {
6
7     @FXML
8     private ComboBox<String> pemasukanBox;
9
10    public TextField hargaText;
11    public TextArea catatanText;
12    public DatePicker tanggalPemasukan;
13
14    @Override
15    public void initialize(URL url, ResourceBundle resourceBundle) {
16        ObservableList<String> itemsPemasukan = FXCollections.observableArrayList("Gaji", "Kiriman", "Uang Jajan",
17            "Hasil Usaha", "Investasi", "Bonus", "Lainnya");
18        pemasukanBox.setItems(itemsPemasukan);
19    }
20
21    public void OnAdd(ActionEvent event) {
22
23        String catatan = catatanText.getText();
24
25        boolean isValid = true;
26        StringBuilder errorMessage = new StringBuilder();
27
28        String uniqueID = UUID.randomUUID().toString();
29        String idpemasukan = "PMS" + uniqueID.substring(1, 8);
30
31        try {
32
33            if (pemasukanBox.getValue() == null) {
34                isValid = false;
35
36                errorMessage.append("- Harap pilih kategori pemasukan.\n");
37            }
38
39            if (hargaText.getText().isEmpty()) {
40                isValid = false;
41                errorMessage.append("- Jumlah tidak boleh kosong.\n");
42            } else {
43
44                if (!isValid) {
45                    errorMessage.append("Input tidak valid.\n");
46                }
47
48                // Insert logic here
49            }
50        } catch (Exception e) {
51            errorMessage.append("Terjadi kesalahan dalam proses pemasukan.\n");
52        }
53
54        if (!isValid) {
55            JOptionPane.showMessageDialog(null, errorMessage.toString());
56        } else {
57            // Insert logic here
58        }
59    }
60
61
62
63
64
65
66
67
68
69
70
71
72 }
```

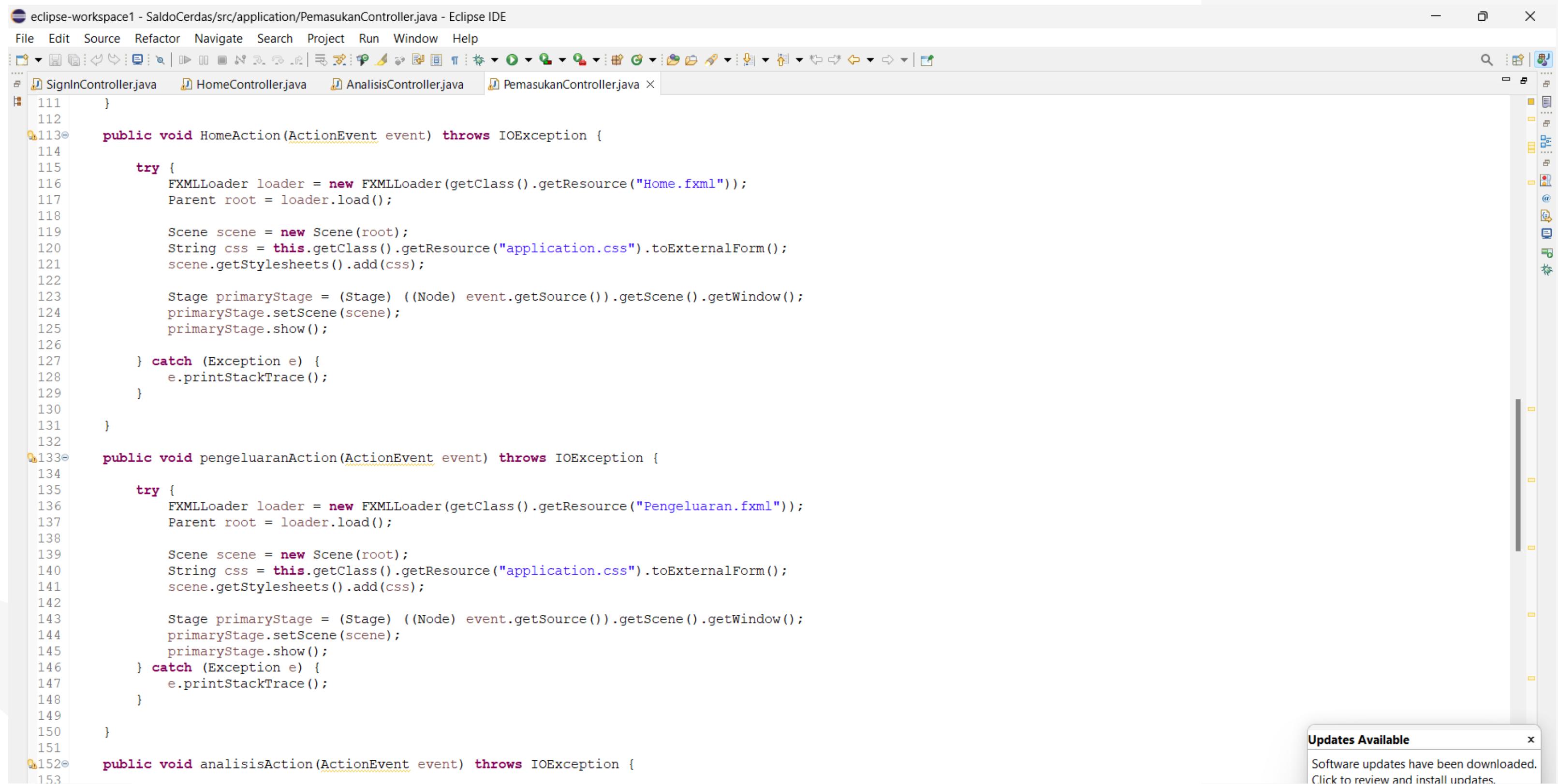
# JAVA (PEMASUKAN PAGE)



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/PemasukanController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows project files: SignInController.java, HomeController.java, AnalisisController.java, and PemasukanController.java (the active tab). The right sidebar contains toolbars for Java, XML, CSS, and JS. A status bar at the bottom right says "Updates Available" with a message: "Software updates have been downloaded. Click to review and install updates." The main editor area displays the Java code for PemasukanController.java, which handles input validation and database insertion.

```
72     } else {
73         Integer jumlah = Integer.parseInt(hargaText.getText());
74         if (jumlah <= 0) {
75             isValid = false;
76             errorMessage.append("- Jumlah tidak boleh lebih kecil dari atau sama dengan 0.\n");
77         }
78     }
79
80     if (tanggalPemasukan.getValue() == null) {
81         isValid = false;
82
83         errorMessage.append("- Harap pilih tanggal pemasukan.\n");
84     }
85
86     if (!isValid) {
87         showErrorAlert(errorMessage.toString());
88     } else {
89         String kategori = pemasukanBox.getValue().toString();
90         String tanggal = tanggalPemasukan.getValue().toString();
91         Integer jumlah = Integer.parseInt(hargaText.getText());
92         User user = new User();
93         String iduser = user.getIduser();
94
95         DatabaseConnector.insertDataPemasukan(idpemasukan, iduser, kategori, jumlah, catatan, tanggal);
96     // showAddAlert("Apakah data pemasukan sudah sesuai?");
97     // showSuccessAlert("Data pemasukan berhasil di add!");
98
99     pemasukanBox.setValue(null);
100    tanggalPemasukan.setValue(null);
101    hargaText.setText("");
102    catatanText.setText("");
103
104    }
105
106 } catch (Exception e) {
107     showErrorAlert("Masukkan jumlah dalam bentuk angka");
108     e.printStackTrace();
109 }
110
111 }
112
113 public void HomeAction(ActionEvent event) throws IOException {
114 }
```

# JAVA (PEMASUKAN PAGE)



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/PemasukanController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Find, and Run. The left sidebar shows project files: SignInController.java, HomeController.java, AnalisisController.java, and PemasukanController.java (which is the active tab). The right sidebar contains various tool icons. The main editor area displays Java code for the PemasukanController class, which handles three action events: HomeAction, pengeluaranAction, and analisisAction. The code uses FXMLLoader to load FXML files and SceneBuilder to create scenes and stages. A tooltip "Updates Available" is visible in the bottom right corner.

```
111     }
112
113     public void HomeAction(ActionEvent event) throws IOException {
114         try {
115             FXMLLoader loader = new FXMLLoader(getClass().getResource("Home.fxml"));
116             Parent root = loader.load();
117
118             Scene scene = new Scene(root);
119             String css = this.getClass().getResource("application.css").toExternalForm();
120             scene.getStylesheets().add(css);
121
122             Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
123             primaryStage.setScene(scene);
124             primaryStage.show();
125
126         } catch (Exception e) {
127             e.printStackTrace();
128         }
129     }
130
131
132
133     public void pengeluaranAction(ActionEvent event) throws IOException {
134         try {
135             FXMLLoader loader = new FXMLLoader(getClass().getResource("Pengeluaran.fxml"));
136             Parent root = loader.load();
137
138             Scene scene = new Scene(root);
139             String css = this.getClass().getResource("application.css").toExternalForm();
140             scene.getStylesheets().add(css);
141
142             Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
143             primaryStage.setScene(scene);
144             primaryStage.show();
145         } catch (Exception e) {
146             e.printStackTrace();
147         }
148     }
149
150
151
152     public void analisisAction(ActionEvent event) throws IOException {
153 }
```

Updates Available

Software updates have been downloaded.  
Click to review and install updates.

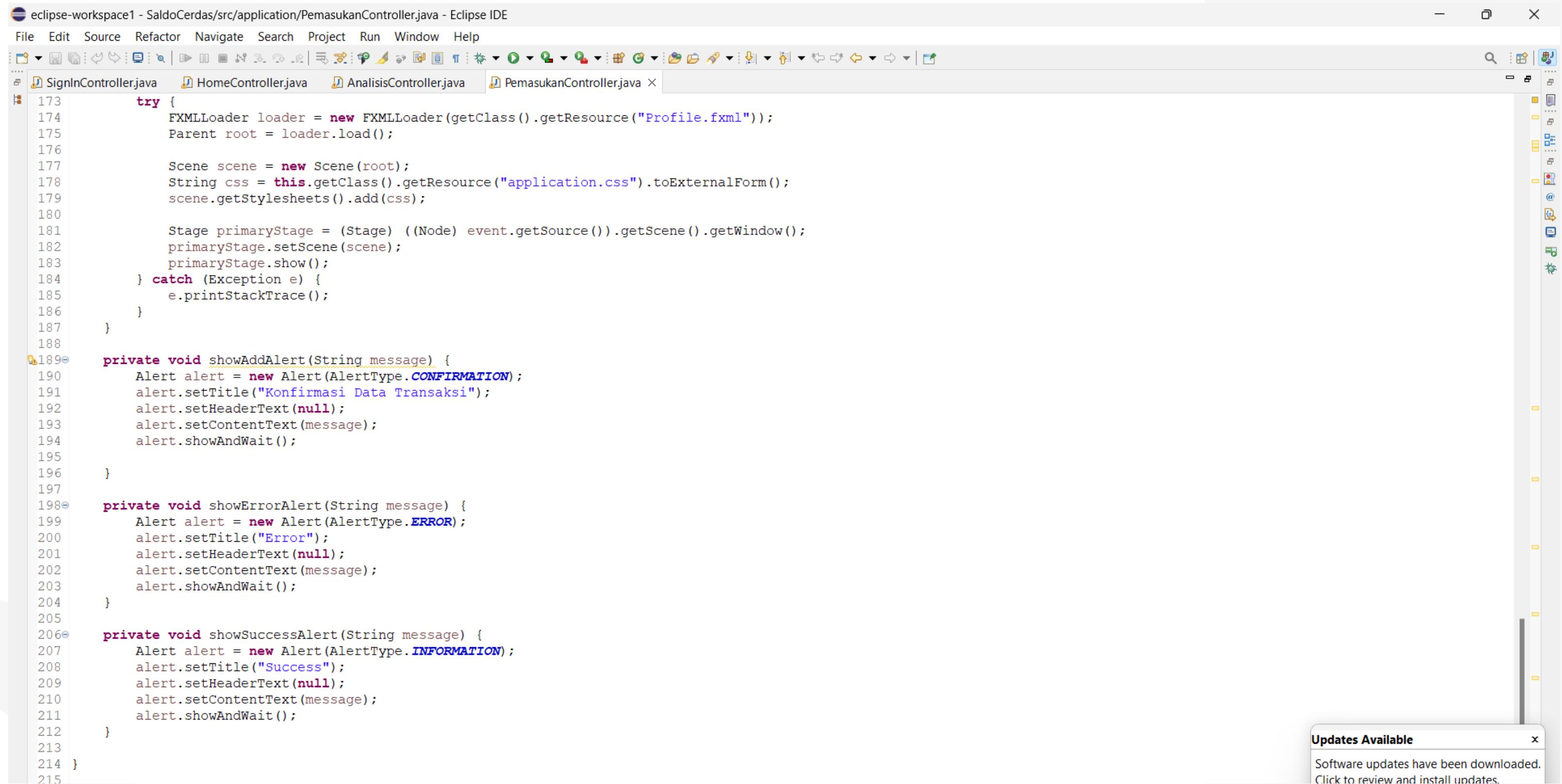
# JAVA (PEMASUKAN PAGE)

```
eclipse-workspace1 - SaldoCerdas/src/application/PemasukanController.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
SignInController.java HomeController.java AnalisisController.java PemasukanController.java
132
133     public void pengeluaranAction(ActionEvent event) throws IOException {
134
135         try {
136             FXMLLoader loader = new FXMLLoader(getClass().getResource("Pengeluaran.fxml"));
137             Parent root = loader.load();
138
139             Scene scene = new Scene(root);
140             String css = this.getClass().getResource("application.css").toExternalForm();
141             scene.getStylesheets().add(css);
142
143             Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
144             primaryStage.setScene(scene);
145             primaryStage.show();
146         } catch (Exception e) {
147             e.printStackTrace();
148         }
149
150     }
151
152     public void analisisAction(ActionEvent event) throws IOException {
153
154         try {
155             FXMLLoader loader = new FXMLLoader(getClass().getResource("Analisis.fxml"));
156             Parent root = loader.load();
157
158             Scene scene = new Scene(root);
159             String css = this.getClass().getResource("application.css").toExternalForm();
160             scene.getStylesheets().add(css);
161
162             Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
163             primaryStage.setScene(scene);
164             primaryStage.show();
165         } catch (Exception e) {
166             e.printStackTrace();
167         }
168
169     }
170
171     public void profileAction(ActionEvent event) throws IOException {
172
173         try {
174             FXMLLoader loader = new FXMLLoader(getClass().getResource("Profile.fxml")):
```

Updates Available

Software updates have been downloaded.  
Click to review and install updates.

# JAVA (PEMASUKAN PAGE)



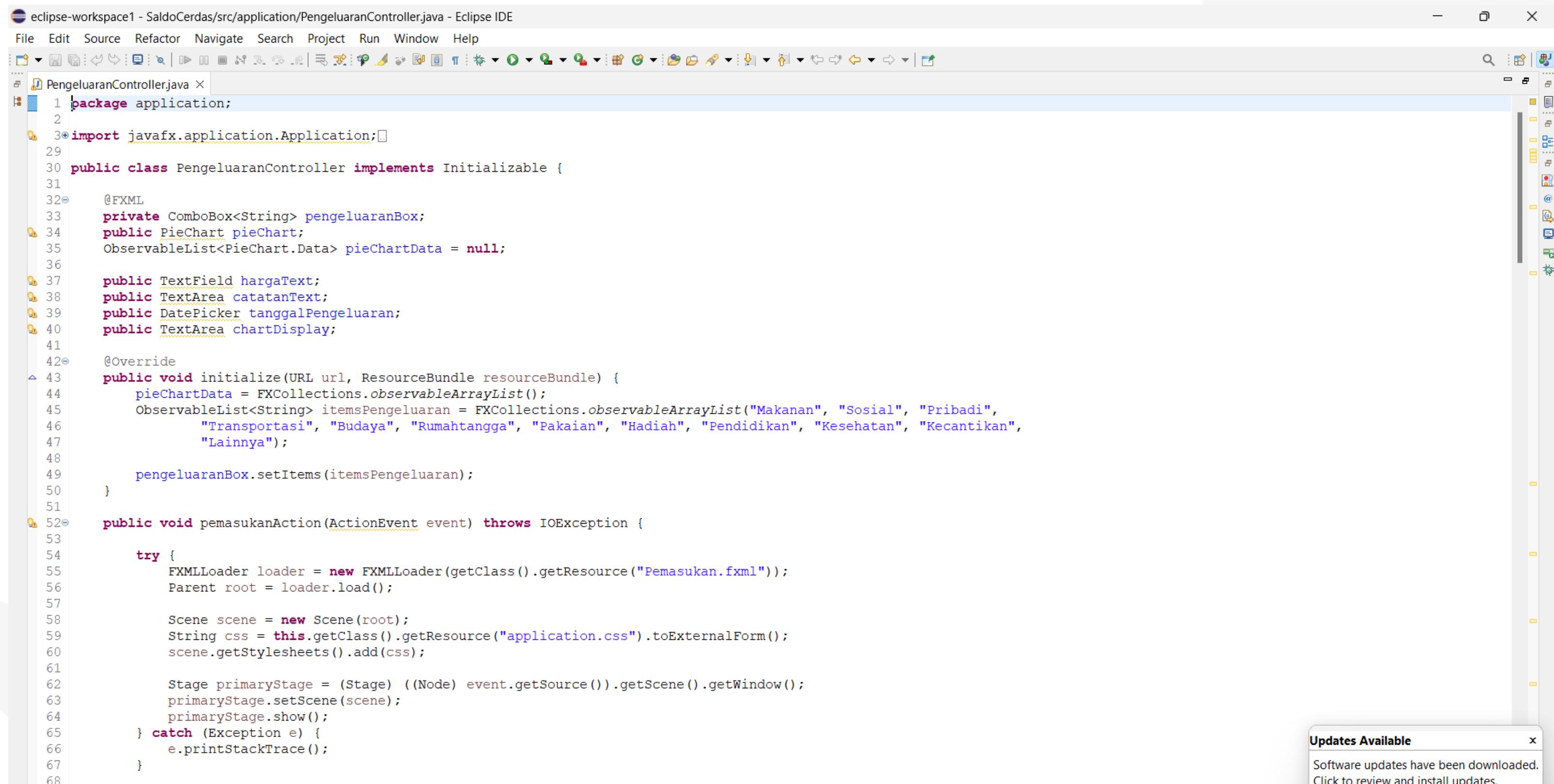
The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/PemasukanController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows project files: SignInController.java, HomeController.java, AnalisisController.java, and PemasukanController.java (which is the active tab). The right sidebar contains a tree view of the project structure and a "Updates Available" notification. The main editor area displays the Java code for PemasukanController.java, which handles FXML loading and alert dialog creation for adding, erroring, and success messages.

```
173     try {
174         FXMLLoader loader = new FXMLLoader(getClass().getResource("Profile.fxml"));
175         Parent root = loader.load();
176
177         Scene scene = new Scene(root);
178         String css = this.getClass().getResource("application.css").toExternalForm();
179         scene.getStylesheets().add(css);
180
181         Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
182         primaryStage.setScene(scene);
183         primaryStage.show();
184     } catch (Exception e) {
185         e.printStackTrace();
186     }
187 }
188
189 private void showAddAlert(String message) {
190     Alert alert = new Alert(AlertType.CONFIRMATION);
191     alert.setTitle("Konfirmasi Data Transaksi");
192     alert.setHeaderText(null);
193     alert.setContentText(message);
194     alert.showAndWait();
195 }
196
197 private void showErrorAlert(String message) {
198     Alert alert = new Alert(AlertType.ERROR);
199     alert.setTitle("Error");
200     alert.setHeaderText(null);
201     alert.setContentText(message);
202     alert.showAndWait();
203 }
204
205 private void showSuccessAlert(String message) {
206     Alert alert = new Alert(AlertType.INFORMATION);
207     alert.setTitle("Success");
208     alert.setHeaderText(null);
209     alert.setContentText(message);
210     alert.showAndWait();
211 }
212
213 }
214 }
```

Updates Available

Software updates have been downloaded.  
Click to review and install updates.

# JAVA (PENGELUARAN PAGE)



The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace1 - SaldoCerdas/src/application/PengeluaranController.java - Eclipse IDE
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbar:** Standard Eclipse toolbar icons.
- Left Margin:** Line numbers from 1 to 68.
- Code Area:** Java code for PengeluaranController.java. The code initializes a ComboBox with categories like Makanan, Sosial, Pribadi, etc., and handles a pemasukanAction event.
- Right Margin:** Updates Available dialog box: Software updates have been downloaded. Click to review and install updates.
- Sidebar:** Project Explorer, Package Explorer, and other Eclipse toolbars.

```
1 package application;
2
3 import javafx.application.Application;
4
5 public class PengeluaranController implements Initializable {
6
7     @FXML
8     private ComboBox<String> pengeluaranBox;
9     public PieChart pieChart;
10    ObservableList<PieChart.Data> pieChartData = null;
11
12    public TextField hargaText;
13    public TextArea catatanText;
14    public DatePicker tanggalPengeluaran;
15    public TextArea chartDisplay;
16
17    @Override
18    public void initialize(URL url, ResourceBundle resourceBundle) {
19        pieChartData = FXCollections.observableArrayList();
20        ObservableList<String> itemsPengeluaran = FXCollections.observableArrayList("Makanan", "Sosial", "Pribadi",
21            "Transportasi", "Budaya", "Rumahtangga", "Pakaian", "Hadiah", "Pendidikan", "Kesehatan", "Kecantikan",
22            "Lainnya");
23
24        pengeluaranBox.setItems(itemsPengeluaran);
25    }
26
27    public void pemasukanAction(ActionEvent event) throws IOException {
28
29        try {
30            FXMLLoader loader = new FXMLLoader(getClass().getResource("Pemasukan.fxml"));
31            Parent root = loader.load();
32
33            Scene scene = new Scene(root);
34            String css = this.getClass().getResource("application.css").toExternalForm();
35            scene.getStylesheets().add(css);
36
37            Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
38            primaryStage.setScene(scene);
39            primaryStage.show();
40        } catch (Exception e) {
41            e.printStackTrace();
42        }
43    }
44}
```

# JAVA (PENGELUARAN PAGE)

eclipse-workspace1 - SaldoCerdas/src/application/PengeluaranController.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

PengeluaranController.java

```
65     } catch (Exception e) {
66         e.printStackTrace();
67     }
68 }
69 }
70 }
71 public void OnAdd(ActionEvent event) {
72
73     String catatan = catatanText.getText();
74
75     boolean isValid = true;
76     StringBuilder errorMessage = new StringBuilder();
77
78     String uniqueID = UUID.randomUUID().toString();
79     String idpengeluaran = "PNG" + uniqueID.substring(1, 8);
80
81     try {
82         if (pengeluaranBox.getValue() == null) {
83             isValid = false;
84
85             errorMessage.append("- Harap pilih kategori pengeluaran.\n");
86         }
87
88         if (hargaText.getText().isEmpty()) {
89             isValid = false;
90             errorMessage.append("- Jumlah tidak boleh kosong.\n");
91         } else {
92             Integer jumlah = Integer.parseInt(hargaText.getText());
93             if (jumlah <= 0) {
94                 isValid = false;
95                 errorMessage.append("- Jumlah tidak boleh lebih kecil dari atau sama dengan 0.\n");
96             }
97         }
98
99         if (tanggalPengeluaran.getValue() == null) {
100            isValid = false;
101
102            errorMessage.append("- Harap pilih tanggal pengeluaran.\n");
103        }
104
105        if (!isValid) {
106            showErrorAlert(errorMessage.toString());
107        } else {
```

# JAVA (PENGELUARAN PAGE)

```
eclipse-workspace1 - SaldoCerdas/src/application/PengeluaranController.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
PengeluaranController.java X
107     } else {
108         String kategori = pengeluaranBox.getValue().toString();
109         String tanggal = tanggalPengeluaran.getValue().toString();
110         Integer jumlah = Integer.parseInt(hargaText.getText());
111         User user = new User();
112         String iduser = user.getIduser();
113
114         DatabaseConnector.insertDataPengeluaran(idpengeluaran, iduser, kategori, jumlah, catatan, tanggal);
115         showSuccessAlert("Data pengeluaran berhasil di add!");
116
117         pengeluaranBox.setValue(null);
118         tanggalPengeluaran.setValue(null);
119         hargaText.setText("");
120         catatanText.setText("");
121
122     }
123
124 } catch (Exception e) {
125     showErrorAlert("Masukkan jumlah dalam bentuk angka");
126     e.printStackTrace();
127 }
128
129
130
131 public void HomeAction(ActionEvent event) throws IOException {
132
133     try {
134         FXMLLoader loader = new FXMLLoader(getClass().getResource("Home.fxml"));
135         Parent root = loader.load();
136
137         Scene scene = new Scene(root);
138         String css = this.getClass().getResource("application.css").toExternalForm();
139         scene.getStylesheets().add(css);
140
141         Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
142         primaryStage.setScene(scene);
143         primaryStage.show();
144     } catch (Exception e) {
145
146         e.printStackTrace();
147     }
148 }
```

Updates Available

Software updates have been downloaded.  
Click to review and install updates.

# JAVA (PENGELUARAN PAGE)

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace1 - SaldoCerdas/src/application/PengeluaranController.java - Eclipse IDE
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbar:** Standard Eclipse toolbar icons.
- Left Margin:** Line numbers from 146 to 188.
- Code Area:** Java code for `PengeluaranController.java`. The code handles two main actions: `analisisAction` and `profileAction`, both of which load FXML files and show them in a Stage. It also includes a private method `showAddAlert`.
- Right Margin:** Updates Available dialog box indicating software updates have been downloaded.

```
146         e.printStackTrace();
147     }
148
149 }
150
151 public void analisisAction(ActionEvent event) throws IOException {
152     try {
153         FXMLLoader loader = new FXMLLoader(getClass().getResource("Analisis.fxml"));
154         Parent root = loader.load();
155
156         Scene scene = new Scene(root);
157         String css = this.getClass().getResource("application.css").toExternalForm();
158         scene.getStylesheets().add(css);
159
160         Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
161         primaryStage.setScene(scene);
162         primaryStage.show();
163     } catch (Exception e) {
164         e.printStackTrace();
165     }
166 }
167
168 }
169
170 public void profileAction(ActionEvent event) throws IOException {
171     try {
172         FXMLLoader loader = new FXMLLoader(getClass().getResource("Profile.fxml"));
173         Parent root = loader.load();
174
175         Scene scene = new Scene(root);
176         String css = this.getClass().getResource("application.css").toExternalForm();
177         scene.getStylesheets().add(css);
178
179         Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
180         primaryStage.setScene(scene);
181         primaryStage.show();
182     } catch (Exception e) {
183         e.printStackTrace();
184     }
185 }
186
187
188 private void showAddAlert(String message) {
```

# JAVA (PENGELUARAN PAGE)

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace1 - SaldoCerdas/src/application/PengeluaranController.java - Eclipse IDE
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbar:** Standard Eclipse toolbar with various icons for file operations, search, and project management.
- Left Margin:** Line numbers from 171 to 213 on the left side of the code editor.
- Code Editor:** The main area contains Java code for a controller class. It includes methods for handling FXML loading, displaying alerts (CONFIRMATION, INFORMATION, ERROR), and managing scenes and stages.
- Right Margin:** A vertical bar with several small icons representing different tools or perspectives.
- Status Bar:** Updates Available (x) - Software updates have been downloaded. Click to review and install updates.

```
171     try {
172         FXMLLoader loader = new FXMLLoader(getClass().getResource("Profile.fxml"));
173         Parent root = loader.load();
174
175         Scene scene = new Scene(root);
176         String css = this.getClass().getResource("application.css").toExternalForm();
177         scene.getStylesheets().add(css);
178
179         Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
180         primaryStage.setScene(scene);
181         primaryStage.show();
182     } catch (Exception e) {
183         e.printStackTrace();
184     }
185 }
186
187
188 private void showAddAlert(String message) {
189     Alert alert = new Alert(AlertType.CONFIRMATION);
190     alert.setTitle("Konfirmasi Data Transaksi");
191     alert.setHeaderText(null);
192     alert.setContentText(message);
193     alert.showAndWait();
194 }
195
196 private void showSuccessAlert(String message) {
197     Alert alert = new Alert(AlertType.INFORMATION);
198     alert.setTitle("Success");
199     alert.setHeaderText(null);
200     alert.setContentText(message);
201     alert.showAndWait();
202 }
203
204 private void showErrorAlert(String message) {
205     Alert alert = new Alert(AlertType.ERROR);
206     alert.setTitle("Error");
207     alert.setHeaderText(null);
208     alert.setContentText(message);
209     alert.showAndWait();
210 }
211
212 }
```

# JAVA (PROFILE PAGE)

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/ProfileController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows the package structure with "application" selected. The main editor area displays the Java code for ProfileController.java:

```
1 package application;
2
3 import java.io.IOException;
4
5 public class ProfileController {
6
7     public Label usernameLabel;
8     public Label emailLabel;
9     public Label tanggalLabel;
10
11     public void initialize() {
12         ubahProfile();
13     }
14
15     public void ubahProfile() {
16         User user = new User();
17         usernameLabel.setText(" " + user.getUsername());
18         tanggalLabel.setText(" " + user.getTanggallahir());
19         emailLabel.setText(" " + user.getEmail());
20     }
21
22     public void HomeAction(ActionEvent event) throws IOException {
23
24         try {
25             FXMLLoader loader = new FXMLLoader(getClass().getResource("Home.fxml"));
26             Parent root = loader.load();
27
28             Scene scene = new Scene(root);
29             String css = this.getClass().getResource("application.css").toExternalForm();
30             scene.getStylesheets().add(css);
31
32             Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
33             primaryStage.setScene(scene);
34             primaryStage.show();
35
36         } catch (Exception e) {
37             e.printStackTrace();
38         }
39
40     }
41
42     public void pengeluaranAction(ActionEvent event) throws IOException {
43
44     }
45
46 }
47
48
49
50     public void pengeluaranAction(ActionEvent event) throws IOException {
51 }
```

A small "Updates Available" dialog is visible in the bottom right corner, stating "Software updates have been downloaded. Click to review and install updates."

# JAVA (PROFILE PAGE)

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/ProfileController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Find, and Run. The left sidebar shows the package structure with "PengeluaranController.java" and "ProfileController.java" selected. The main editor area contains Java code for "ProfileController.java". The code defines three methods: pengeluaranAction, pemasukanAction, and analisisAction. Each method uses FXMLLoader to load FXML files ("Pengeluaran.fxml", "Pemasukan.fxml", and "Analisis.fxml") and creates a Stage to display them. An "Updates Available" dialog is visible in the bottom right corner.

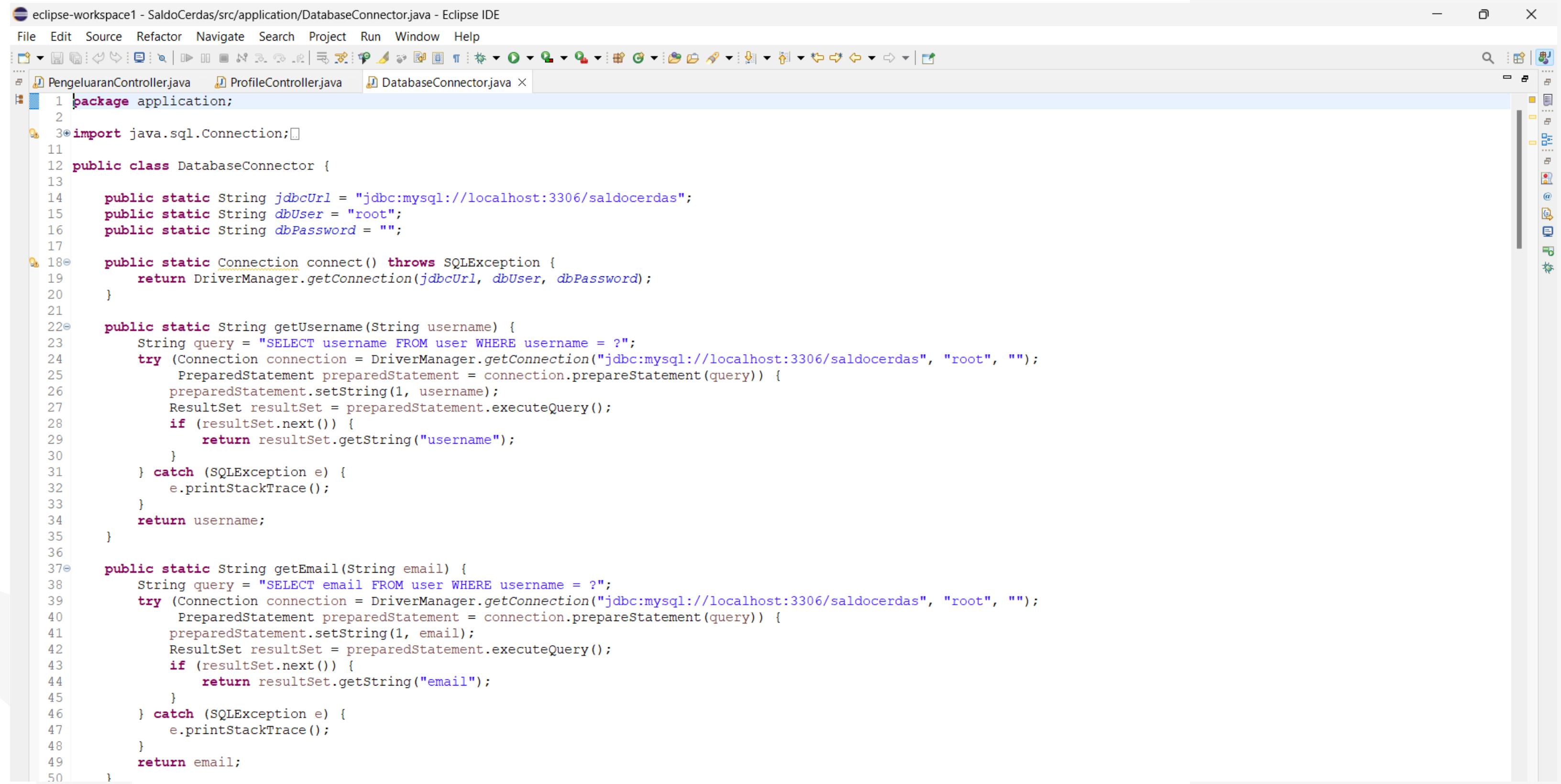
```
48     }
49
50     public void pengeluaranAction(ActionEvent event) throws IOException {
51
52         try {
53             FXMLLoader loader = new FXMLLoader(getClass().getResource("Pengeluaran.fxml"));
54             Parent root = loader.load();
55
56             Scene scene = new Scene(root);
57             String css = this.getClass().getResource("application.css").toExternalForm();
58             scene.getStylesheets().add(css);
59
60             Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
61             primaryStage.setScene(scene);
62             primaryStage.show();
63         } catch (Exception e) {
64             e.printStackTrace();
65         }
66     }
67
68     public void pemasukanAction(ActionEvent event) throws IOException {
69
70         try {
71             FXMLLoader loader = new FXMLLoader(getClass().getResource("Pemasukan.fxml"));
72             Parent root = loader.load();
73
74             Scene scene = new Scene(root);
75             String css = this.getClass().getResource("application.css").toExternalForm();
76             scene.getStylesheets().add(css);
77
78             Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
79             primaryStage.setScene(scene);
80             primaryStage.show();
81         } catch (Exception e) {
82             e.printStackTrace();
83         }
84     }
85
86     public void analisisAction(ActionEvent event) throws IOException {
87
88         try {
89             FXMLLoader loader = new FXMLLoader(getClass().getResource("Analisis.fxml"));
90             Parent root = loader.load();
```

# JAVA (PROFILE PAGE)

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/ProfileController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows the package structure with "PengeluaranController.java" and "ProfileController.java" selected. The main editor area displays Java code for "ProfileController.java". The code handles two actions: "analisisAction" and "logoutAction". Both actions attempt to load FXML files ("Analisis.fxml" and "SignIn.fxml") using FXMLLoader, set them as the scene for the primary stage, and show the stage. If an exception occurs, it prints the stack trace. A status bar at the bottom right indicates "Updates Available" with the message "Software updates have been downloaded. Click to review and install updates".

```
82         e.printStackTrace();
83     }
84 }
85
86 public void analisisAction(ActionEvent event) throws IOException {
87
88     try {
89         FXMLLoader loader = new FXMLLoader(getClass().getResource("Analisis.fxml"));
90         Parent root = loader.load();
91
92         Scene scene = new Scene(root);
93         String css = this.getClass().getResource("application.css").toExternalForm();
94         scene.getStylesheets().add(css);
95
96         Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
97         primaryStage.setScene(scene);
98         primaryStage.show();
99     } catch (Exception e) {
100         e.printStackTrace();
101     }
102 }
103
104 public void logoutAction(ActionEvent event) throws IOException {
105
106     try {
107         FXMLLoader loader = new FXMLLoader(getClass().getResource("SignIn.fxml"));
108         Parent root = loader.load();
109
110         Scene scene = new Scene(root);
111         String css = this.getClass().getResource("application.css").toExternalForm();
112         scene.getStylesheets().add(css);
113
114         Stage primaryStage = (Stage) ((Node) event.getSource()).getScene().getWindow();
115         primaryStage.setScene(scene);
116         primaryStage.show();
117     } catch (Exception e) {
118         e.printStackTrace();
119     }
120 }
121
122 }
123 }
```

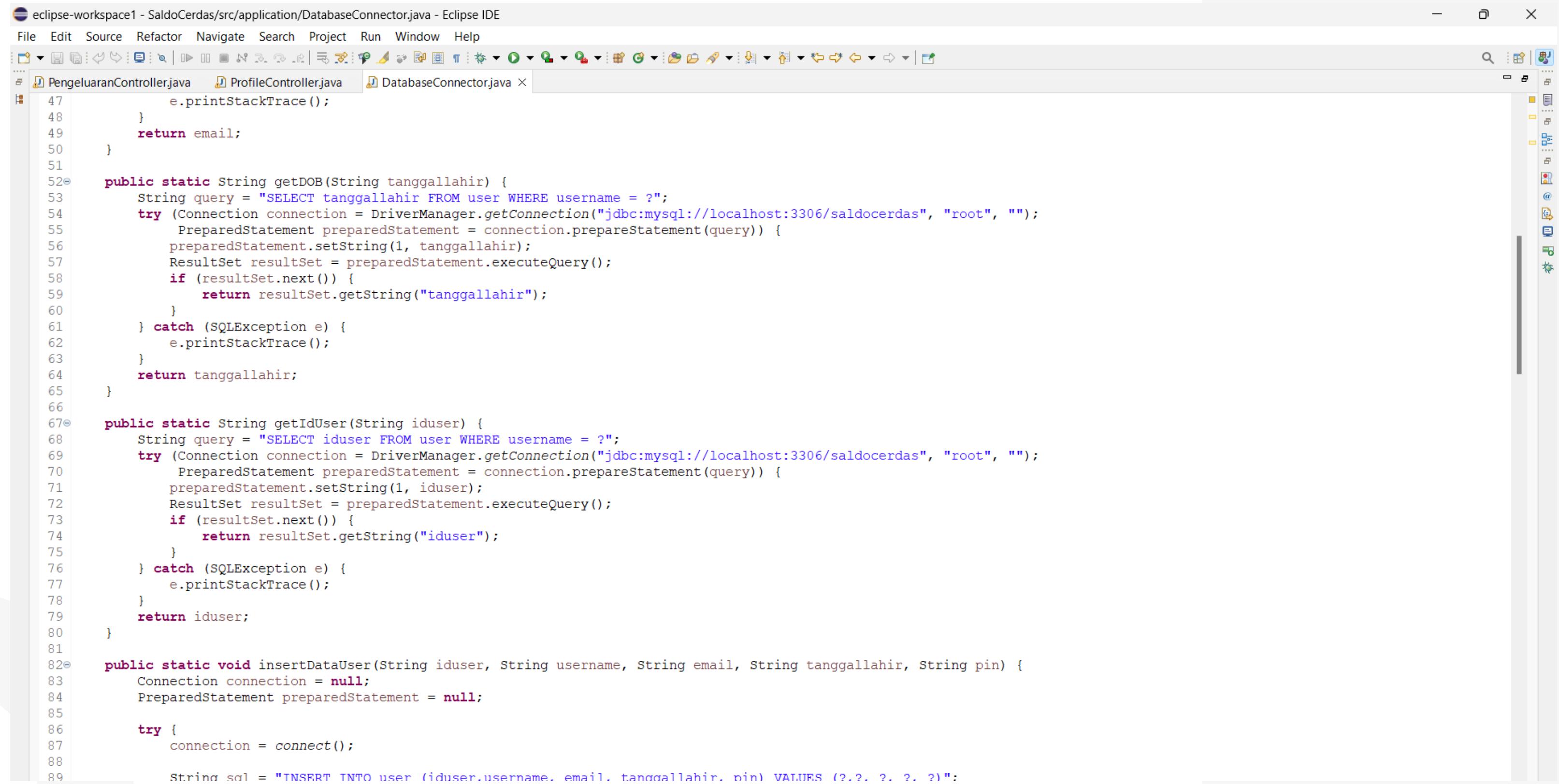
# JAVA (DATABASE CONNECTOR)



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/DatabaseConnector.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows project files: PengeluaranController.java, ProfileController.java, and DatabaseConnector.java (the active tab). The right sidebar contains various tool icons. The main editor area displays the Java code for DatabaseConnector.java:

```
1 package application;
2
3 import java.sql.Connection;
4
5 public class DatabaseConnector {
6
7     public static String jdbcUrl = "jdbc:mysql://localhost:3306/saldocerdas";
8     public static String dbUser = "root";
9     public static String dbPassword = "";
10
11    public static Connection connect() throws SQLException {
12        return DriverManager.getConnection(jdbcUrl, dbUser, dbPassword);
13    }
14
15    public static String getUsername(String username) {
16        String query = "SELECT username FROM user WHERE username = ?";
17        try (Connection connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/saldocerdas", "root", ""));
18        PreparedStatement preparedStatement = connection.prepareStatement(query);
19        preparedStatement.setString(1, username);
20        ResultSet resultSet = preparedStatement.executeQuery();
21        if (resultSet.next()) {
22            return resultSet.getString("username");
23        }
24        } catch (SQLException e) {
25            e.printStackTrace();
26        }
27        return username;
28    }
29
30    public static String getEmail(String email) {
31        String query = "SELECT email FROM user WHERE username = ?";
32        try (Connection connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/saldocerdas", "root", ""));
33        PreparedStatement preparedStatement = connection.prepareStatement(query);
34        preparedStatement.setString(1, email);
35        ResultSet resultSet = preparedStatement.executeQuery();
36        if (resultSet.next()) {
37            return resultSet.getString("email");
38        }
39        } catch (SQLException e) {
40            e.printStackTrace();
41        }
42        return email;
43    }
44}
```

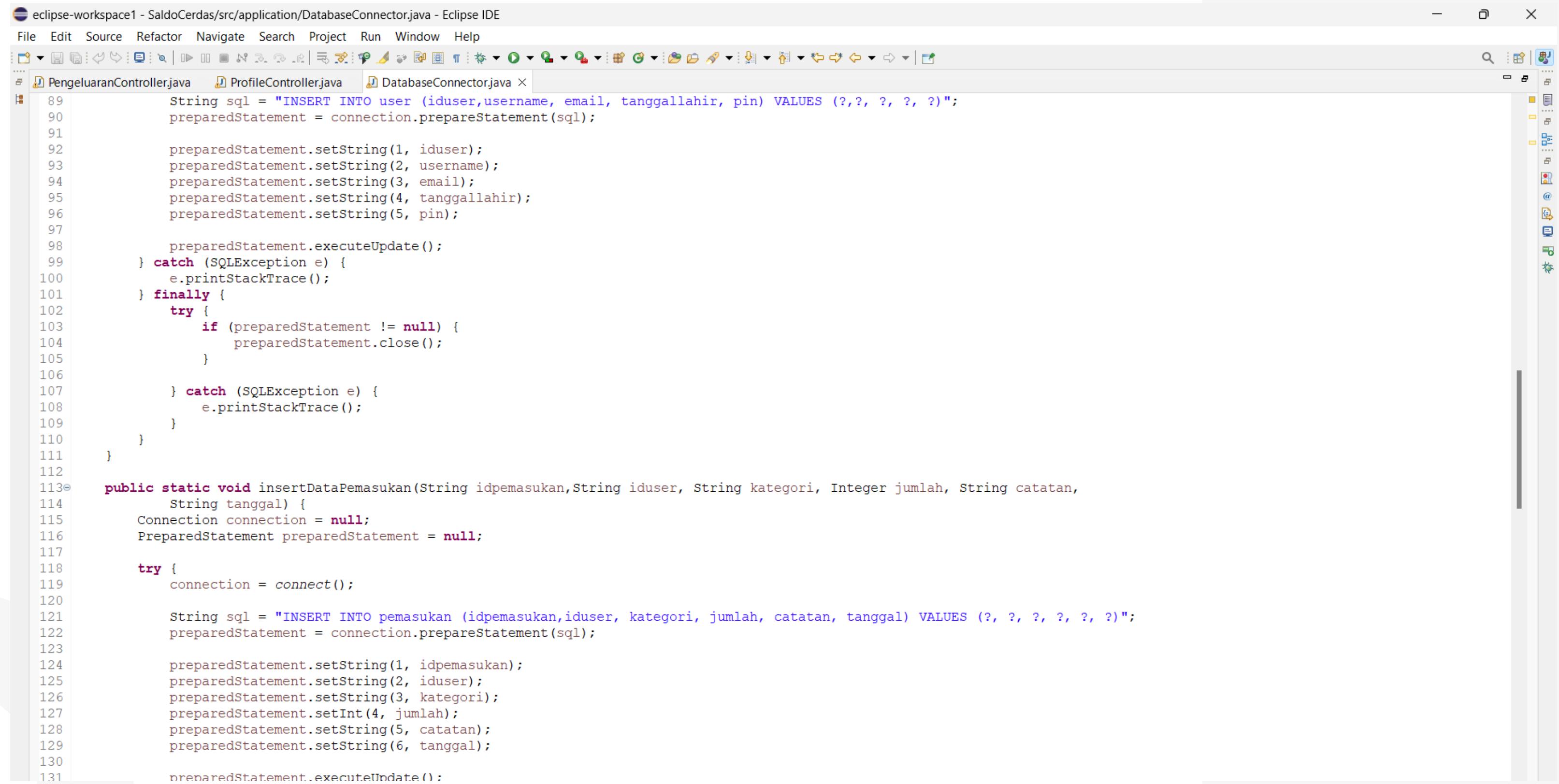
# JAVA (DATABASE CONNECTOR)



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/DatabaseConnector.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows project files: PengeluaranController.java, ProfileController.java, and DatabaseConnector.java (the active tab). The right sidebar contains various tool icons. The main editor area displays the Java code for DatabaseConnector.java, which includes methods for getting DOB and ID from the database and inserting data into the user table.

```
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     {
56         PreparedStatement preparedStatement = connection.prepareStatement(query);
57         preparedStatement.setString(1, tanggallahir);
58         ResultSet resultSet = preparedStatement.executeQuery();
59         if (resultSet.next()) {
60             return resultSet.getString("tanggallahir");
61         }
62     } catch (SQLException e) {
63         e.printStackTrace();
64     }
65     return tanggallahir;
66 }
67
68 public static String getIdUser(String iduser) {
69     String query = "SELECT iduser FROM user WHERE username = ?";
70     try (Connection connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/saldocerdas", "root", ""));
71     {
72         PreparedStatement preparedStatement = connection.prepareStatement(query);
73         preparedStatement.setString(1, iduser);
74         ResultSet resultSet = preparedStatement.executeQuery();
75         if (resultSet.next()) {
76             return resultSet.getString("iduser");
77         }
78     } catch (SQLException e) {
79         e.printStackTrace();
80     }
81     return iduser;
82 }
83
84 public static void insertDataUser(String iduser, String username, String email, String tanggallahir, String pin) {
85     Connection connection = null;
86     PreparedStatement preparedStatement = null;
87
88     try {
89         connection = connect();
90
91         String sql = "INSERT INTO user (iduser,username,email,tanggallahir,pin) VALUES (?, ?, ?, ?, ?)";
92         preparedStatement = connection.prepareStatement(sql);
93         preparedStatement.setString(1, iduser);
94         preparedStatement.setString(2, username);
95         preparedStatement.setString(3, email);
96         preparedStatement.setString(4, tanggallahir);
97         preparedStatement.setString(5, pin);
98
99         preparedStatement.executeUpdate();
100    } catch (SQLException e) {
101        e.printStackTrace();
102    } finally {
103        if (preparedStatement != null) {
104            try {
105                preparedStatement.close();
106            } catch (SQLException e) {
107                e.printStackTrace();
108            }
109        }
110        if (connection != null) {
111            try {
112                connection.close();
113            } catch (SQLException e) {
114                e.printStackTrace();
115            }
116        }
117    }
118 }
```

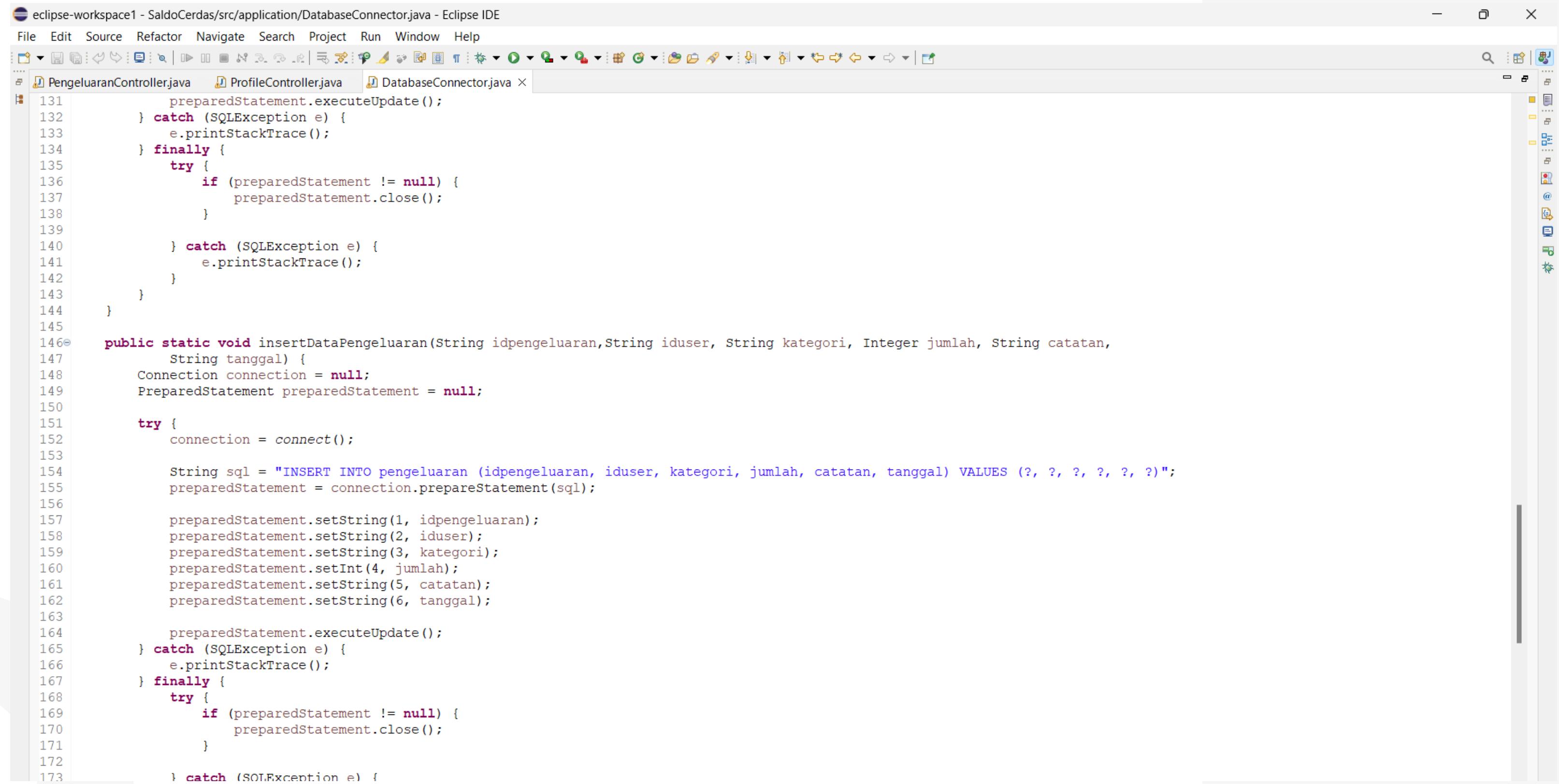
# JAVA (DATABASE CONNECTOR)



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/DatabaseConnector.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Find, and Run. The left sidebar shows three open files: PengeluaranController.java, ProfileController.java, and DatabaseConnector.java (the active tab). The DatabaseConnector.java code is displayed in the main editor area:

```
89     String sql = "INSERT INTO user (iduser,username, email, tanggallahir, pin) VALUES (?,?,?,?,?)";
90     preparedStatement = connection.prepareStatement(sql);
91
92     preparedStatement.setString(1, iduser);
93     preparedStatement.setString(2, username);
94     preparedStatement.setString(3, email);
95     preparedStatement.setString(4, tanggallahir);
96     preparedStatement.setString(5, pin);
97
98     preparedStatement.executeUpdate();
99 } catch (SQLException e) {
100     e.printStackTrace();
101 } finally {
102     try {
103         if (preparedStatement != null) {
104             preparedStatement.close();
105         }
106     } catch (SQLException e) {
107         e.printStackTrace();
108     }
109 }
110 }
111 }
112
113 public static void insertDataPemasukan(String idpemasukan, String iduser, String kategori, Integer jumlah, String catatan,
114                                         String tanggal) {
115     Connection connection = null;
116     PreparedStatement preparedStatement = null;
117
118     try {
119         connection = connect();
120
121         String sql = "INSERT INTO pemasukan (idpemasukan,iduser, kategori, jumlah, catatan, tanggal) VALUES (?,?,?,?,?,?)";
122         preparedStatement = connection.prepareStatement(sql);
123
124         preparedStatement.setString(1, idpemasukan);
125         preparedStatement.setString(2, iduser);
126         preparedStatement.setString(3, kategori);
127         preparedStatement.setInt(4, jumlah);
128         preparedStatement.setString(5, catatan);
129         preparedStatement.setString(6, tanggal);
130
131         preparedStatement.executeUpdate();
```

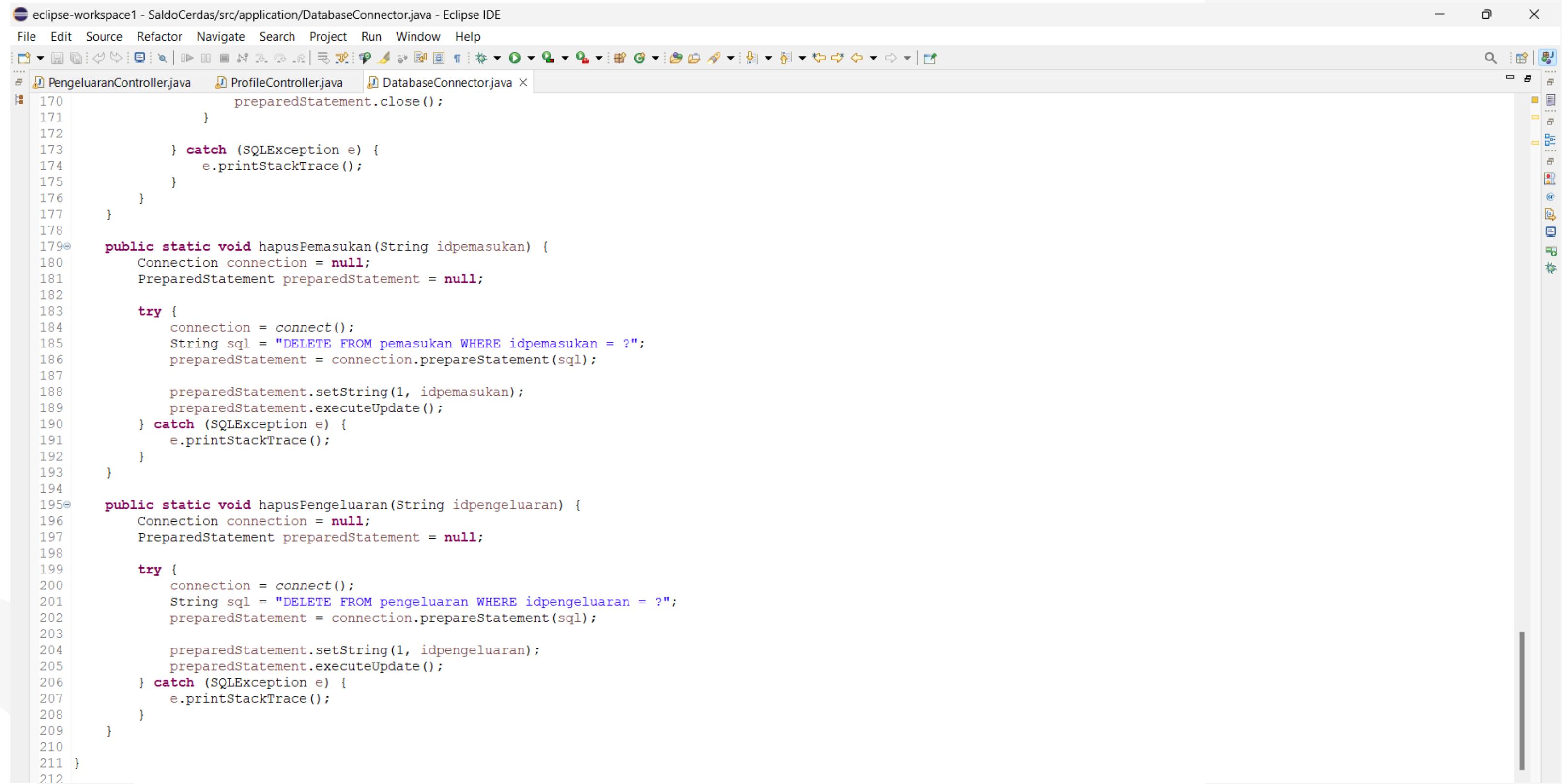
# JAVA (DATABASE CONNECTOR)



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/DatabaseConnector.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Find, and Run. The left sidebar shows three open files: PengeluaranController.java, ProfileController.java, and DatabaseConnector.java (the active tab). The right sidebar contains various toolbars and views. The main editor area displays Java code for a DatabaseConnector class, specifically the insertDataPengeluaran method, which handles SQL statements for inserting data into a pengeluaran table.

```
131     preparedStatement.executeUpdate();
132 } catch (SQLException e) {
133     e.printStackTrace();
134 } finally {
135     try {
136         if (preparedStatement != null) {
137             preparedStatement.close();
138         }
139     } catch (SQLException e) {
140         e.printStackTrace();
141     }
142 }
143 }
144 }
145
146 public static void insertDataPengeluaran(String idpengeluaran, String iduser, String kategori, Integer jumlah, String catatan,
147     String tanggal) {
148     Connection connection = null;
149     PreparedStatement preparedStatement = null;
150
151     try {
152         connection = connect();
153
154         String sql = "INSERT INTO pengeluaran (idpengeluaran, iduser, kategori, jumlah, catatan, tanggal) VALUES (?, ?, ?, ?, ?, ?)";
155         preparedStatement = connection.prepareStatement(sql);
156
157         preparedStatement.setString(1, idpengeluaran);
158         preparedStatement.setString(2, iduser);
159         preparedStatement.setString(3, kategori);
160         preparedStatement.setInt(4, jumlah);
161         preparedStatement.setString(5, catatan);
162         preparedStatement.setString(6, tanggal);
163
164         preparedStatement.executeUpdate();
165     } catch (SQLException e) {
166         e.printStackTrace();
167     } finally {
168         try {
169             if (preparedStatement != null) {
170                 preparedStatement.close();
171             }
172         } catch (SQLException e) {
```

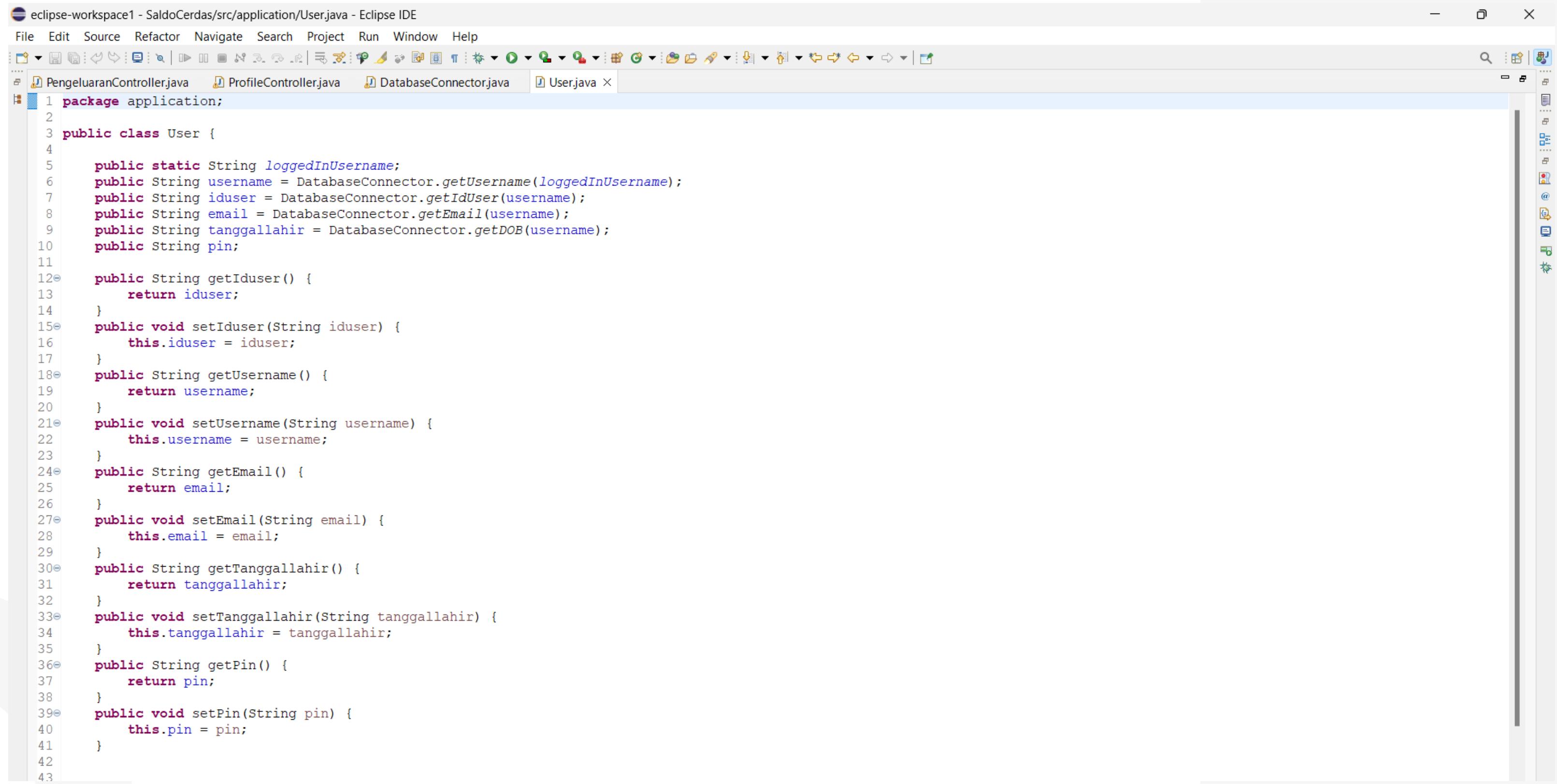
# JAVA (DATABASE CONNECTOR)



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/DatabaseConnector.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Find, and Run. The left sidebar shows the package structure under "SaldoCerdas" with files PengeluaranController.java, ProfileController.java, and DatabaseConnector.java. The main editor area displays the Java code for DatabaseConnector.java, which contains methods for deleting entries from a database. The code uses JDBC to connect to a database and execute SQL DELETE statements. Error handling is provided using try-catch blocks to manage SQLExceptions.

```
170         preparedStatement.close();
171     }
172 
173     } catch (SQLException e) {
174         e.printStackTrace();
175     }
176 }
177 }
178
179 public static void hapusPemasukan(String idpemasukan) {
180     Connection connection = null;
181     PreparedStatement preparedStatement = null;
182 
183     try {
184         connection = connect();
185         String sql = "DELETE FROM pemasukan WHERE idpemasukan = ?";
186         preparedStatement = connection.prepareStatement(sql);
187 
188         preparedStatement.setString(1, idpemasukan);
189         preparedStatement.executeUpdate();
190     } catch (SQLException e) {
191         e.printStackTrace();
192     }
193 }
194
195 public static void hapusPengeluaran(String idpengeluaran) {
196     Connection connection = null;
197     PreparedStatement preparedStatement = null;
198 
199     try {
200         connection = connect();
201         String sql = "DELETE FROM pengeluaran WHERE idpengeluaran = ?";
202         preparedStatement = connection.prepareStatement(sql);
203 
204         preparedStatement.setString(1, idpengeluaran);
205         preparedStatement.executeUpdate();
206     } catch (SQLException e) {
207         e.printStackTrace();
208     }
209 }
210
211 }
```

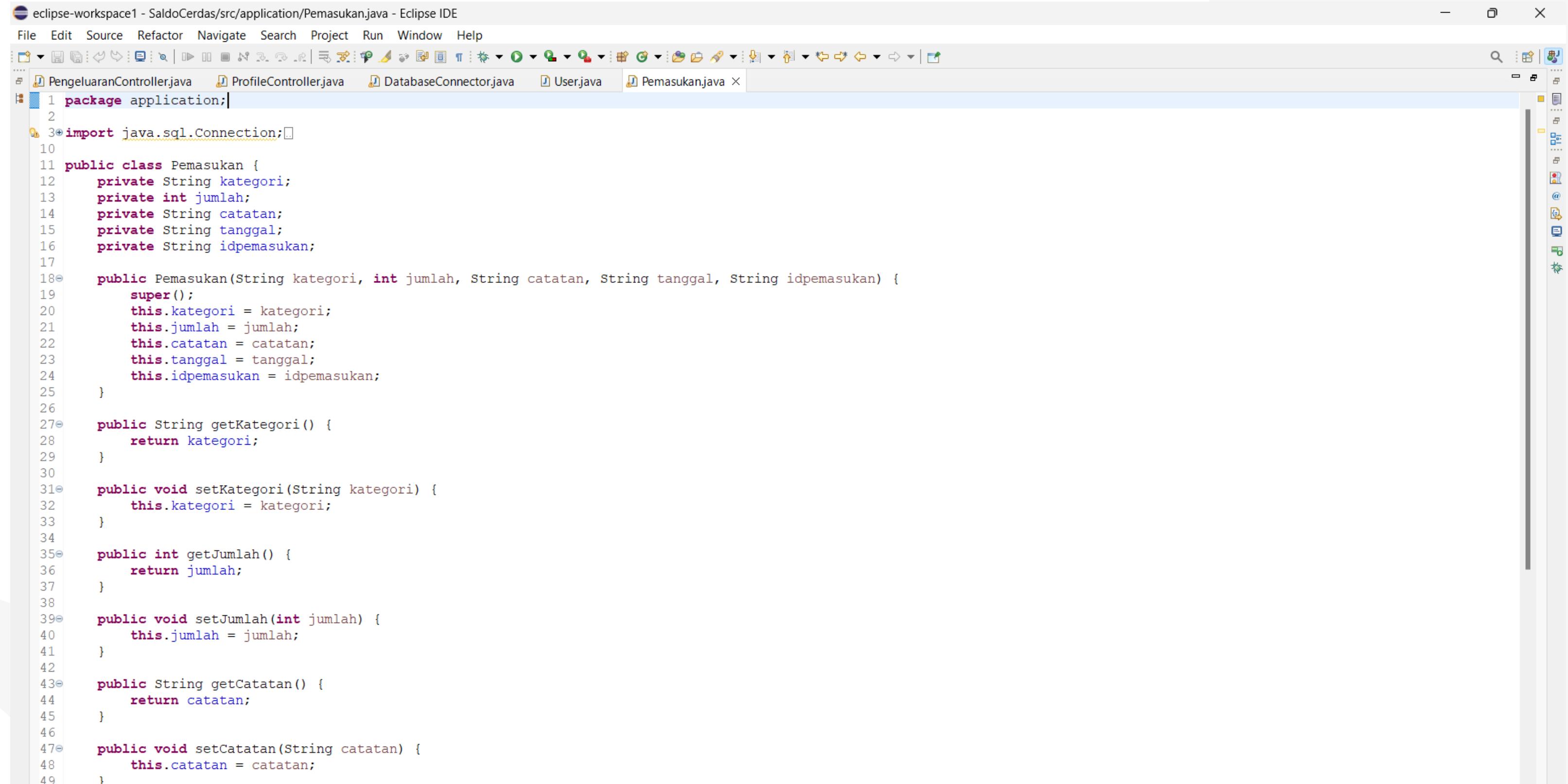
# JAVA (USER)



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/User.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows project files: PengeluaranController.java, ProfileController.java, DatabaseConnector.java, and User.java (the current file). The main editor area displays the following Java code:

```
1 package application;
2
3 public class User {
4
5     public static String loggedInUsername;
6     public String username = DatabaseConnector.getUsername(loggedInUsername);
7     public String iduser = DatabaseConnector.getIdUser(username);
8     public String email = DatabaseConnector.getEmail(username);
9     public String tanggallahir = DatabaseConnector.getDOB(username);
10    public String pin;
11
12    public String getIduser() {
13        return iduser;
14    }
15    public void setIduser(String iduser) {
16        this.iduser = iduser;
17    }
18    public String getUsername() {
19        return username;
20    }
21    public void setUsername(String username) {
22        this.username = username;
23    }
24    public String getEmail() {
25        return email;
26    }
27    public void setEmail(String email) {
28        this.email = email;
29    }
30    public String getTanggallahir() {
31        return tanggallahir;
32    }
33    public void setTanggallahir(String tanggallahir) {
34        this.tanggallahir = tanggallahir;
35    }
36    public String getPin() {
37        return pin;
38    }
39    public void setPin(String pin) {
40        this.pin = pin;
41    }
42}
43
```

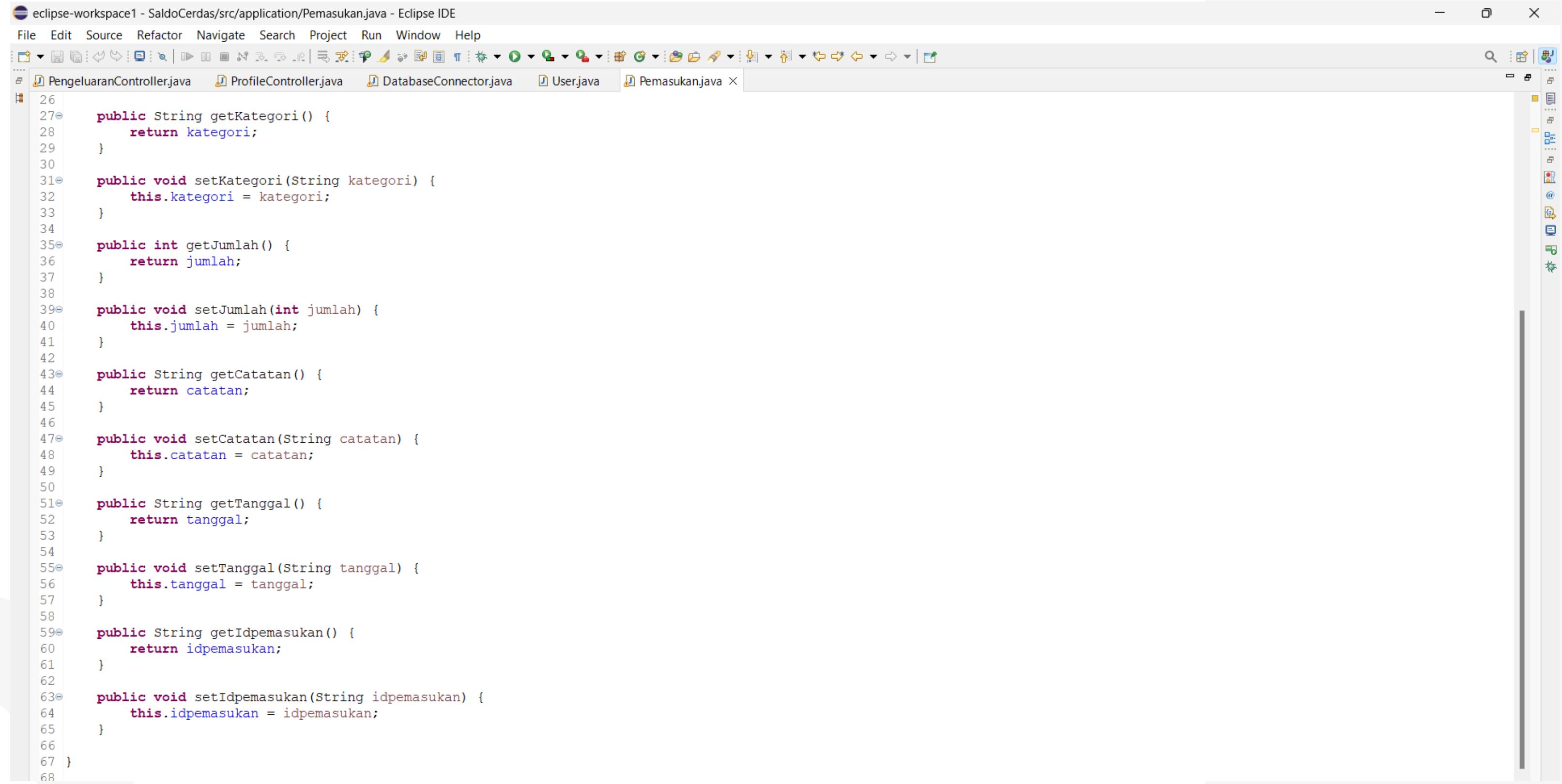
# JAVA (PEMASUKAN)



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/Pemasukan.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows project files: PengeluaranController.java, ProfileController.java, DatabaseConnector.java, User.java, and Pemasukan.java (the active file). The right sidebar contains various tool icons. The main editor area displays the Java code for the Pemasukan class:

```
1 package application;
2
3 import java.sql.Connection;
4
5 public class Pemasukan {
6     private String kategori;
7     private int jumlah;
8     private String catatan;
9     private String tanggal;
10    private String idpemasukan;
11
12    public Pemasukan(String kategori, int jumlah, String catatan, String tanggal, String idpemasukan) {
13        super();
14        this.kategori = kategori;
15        this.jumlah = jumlah;
16        this.catatan = catatan;
17        this.tanggal = tanggal;
18        this.idpemasukan = idpemasukan;
19    }
20
21    public String getKategori() {
22        return kategori;
23    }
24
25    public void setKategori(String kategori) {
26        this.kategori = kategori;
27    }
28
29    public int getJumlah() {
30        return jumlah;
31    }
32
33    public void setJumlah(int jumlah) {
34        this.jumlah = jumlah;
35    }
36
37    public String getCatatan() {
38        return catatan;
39    }
40
41    public void setCatatan(String catatan) {
42        this.catatan = catatan;
43    }
44
45    public String getTanggal() {
46        return tanggal;
47    }
48
49    public void setTanggal(String tanggal) {
50        this.tanggal = tanggal;
51    }
52
53    public String getIdpemasukan() {
54        return idpemasukan;
55    }
56
57    public void setIdpemasukan(String idpemasukan) {
58        this.idpemasukan = idpemasukan;
59    }
60}
```

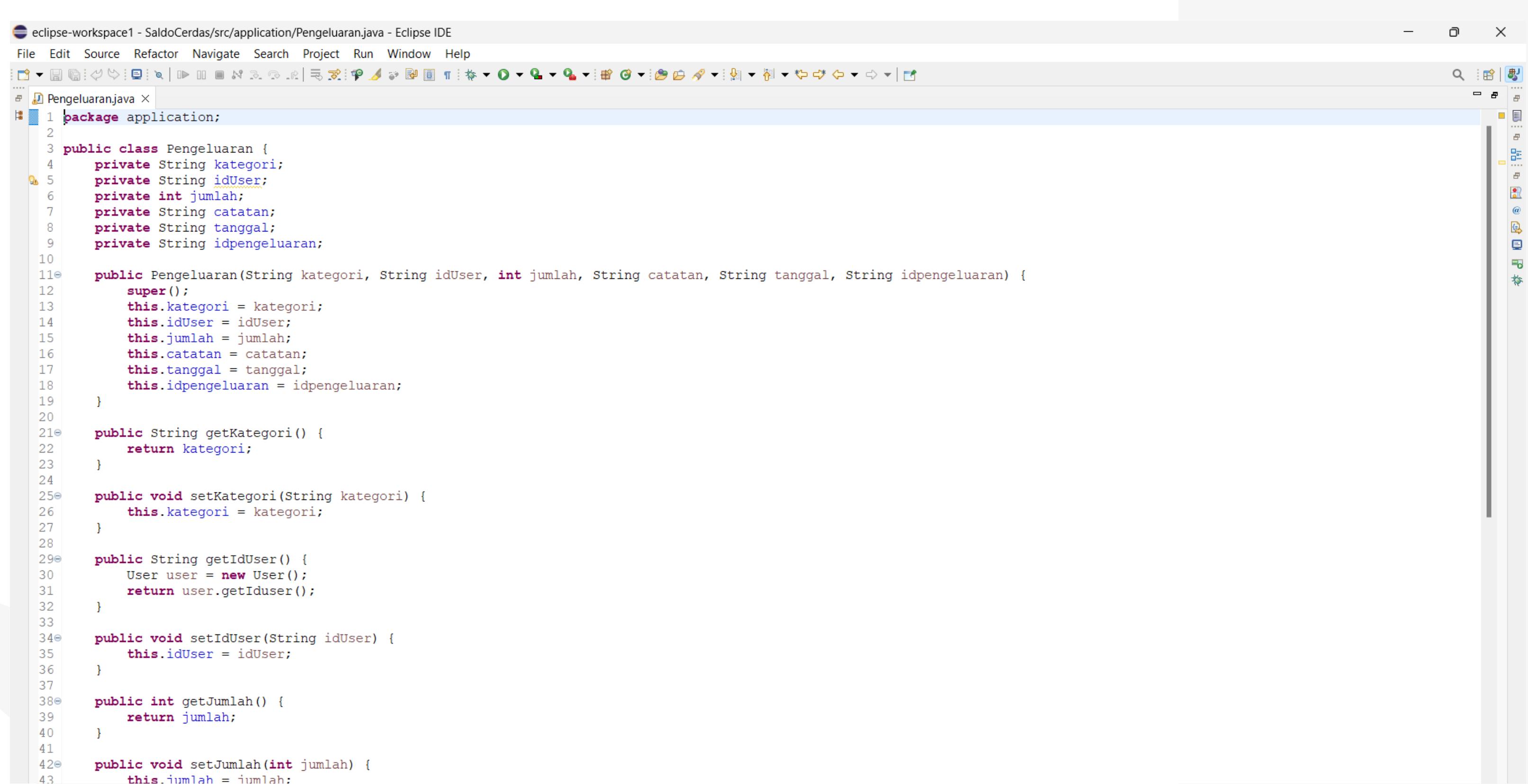
# JAVA (PEMASUKAN)



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace1 - SaldoCerdas/src/application/Pemasukan.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows project files: PengeluaranController.java, ProfileController.java, DatabaseConnector.java, User.java, and Pemasukan.java (the active file). The right sidebar contains icons for search, refresh, and other tools. The main editor area displays the Java code for the Pemasukan class:

```
26
27     public String getKategori() {
28         return kategori;
29     }
30
31     public void setKategori(String kategori) {
32         this.kategori = kategori;
33     }
34
35     public int getJumlah() {
36         return jumlah;
37     }
38
39     public void setJumlah(int jumlah) {
40         this.jumlah = jumlah;
41     }
42
43     public String getCatatan() {
44         return catatan;
45     }
46
47     public void setCatatan(String catatan) {
48         this.catatan = catatan;
49     }
50
51     public String getTanggal() {
52         return tanggal;
53     }
54
55     public void setTanggal(String tanggal) {
56         this.tanggal = tanggal;
57     }
58
59     public String getIdpemasukan() {
60         return idpemasukan;
61     }
62
63     public void setIdpemasukan(String idpemasukan) {
64         this.idpemasukan = idpemasukan;
65     }
66
67 }
68 }
```

# JAVA (PENGELUARAN)

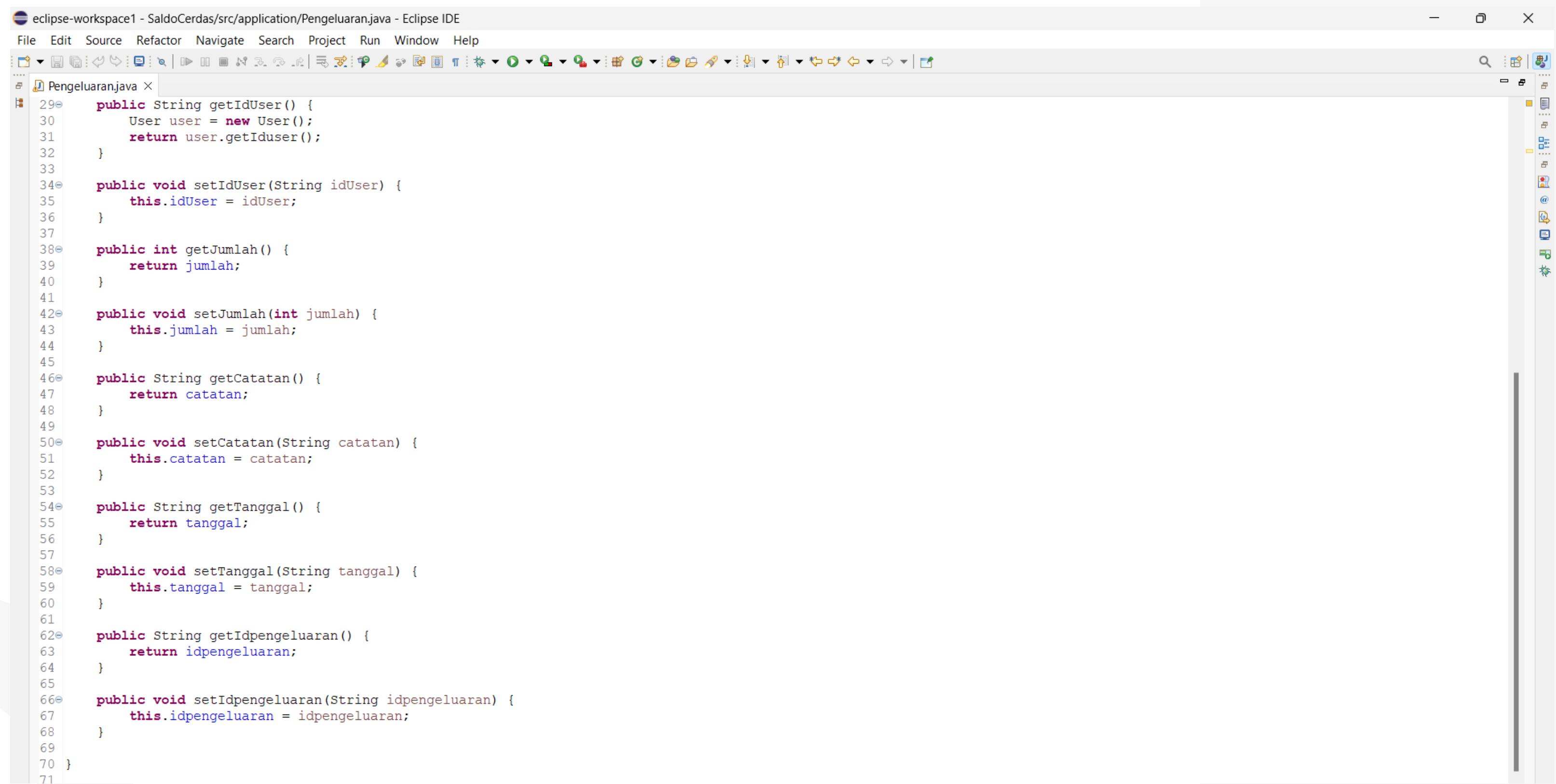


The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace1 - SaldoCerdas/src/application/Pengeluaran.java - Eclipse IDE
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbar:** Standard Eclipse toolbar with various icons for file operations.
- Left Margin:** Shows line numbers from 1 to 43.
- Code Editor:** Displays the Java code for the Pengeluaran class. The code defines a class with private attributes for category, user ID, amount, note, date, and transaction ID, along with their corresponding getters and setters.
- Right Margin:** Shows the Java code completion suggestions.
- Bottom Status Bar:** Shows the current file path: SaldoCerdas/src/application/Pengeluaran.java

```
1 package application;
2
3 public class Pengeluaran {
4     private String kategori;
5     private String idUser;
6     private int jumlah;
7     private String catatan;
8     private String tanggal;
9     private String idpengeluaran;
10
11    public Pengeluaran(String kategori, StringidUser, int jumlah, String catatan, String tanggal, String idpengeluaran) {
12        super();
13        this.kategori = kategori;
14        this.idUser = idUser;
15        this.jumlah = jumlah;
16        this.catatan = catatan;
17        this.tanggal = tanggal;
18        this.idpengeluaran = idpengeluaran;
19    }
20
21    public String getKategori() {
22        return kategori;
23    }
24
25    public void setKategori(String kategori) {
26        this.kategori = kategori;
27    }
28
29    public String getIdUser() {
30        User user = new User();
31        return user.getIduser();
32    }
33
34    public void setIdUser(String idUser) {
35        this.idUser = idUser;
36    }
37
38    public int getJumlah() {
39        return jumlah;
40    }
41
42    public void setJumlah(int jumlah) {
43        this.jumlah = jumlah;
```

# JAVA (PENGELUARAN)



The screenshot shows the Eclipse IDE interface with the following details:

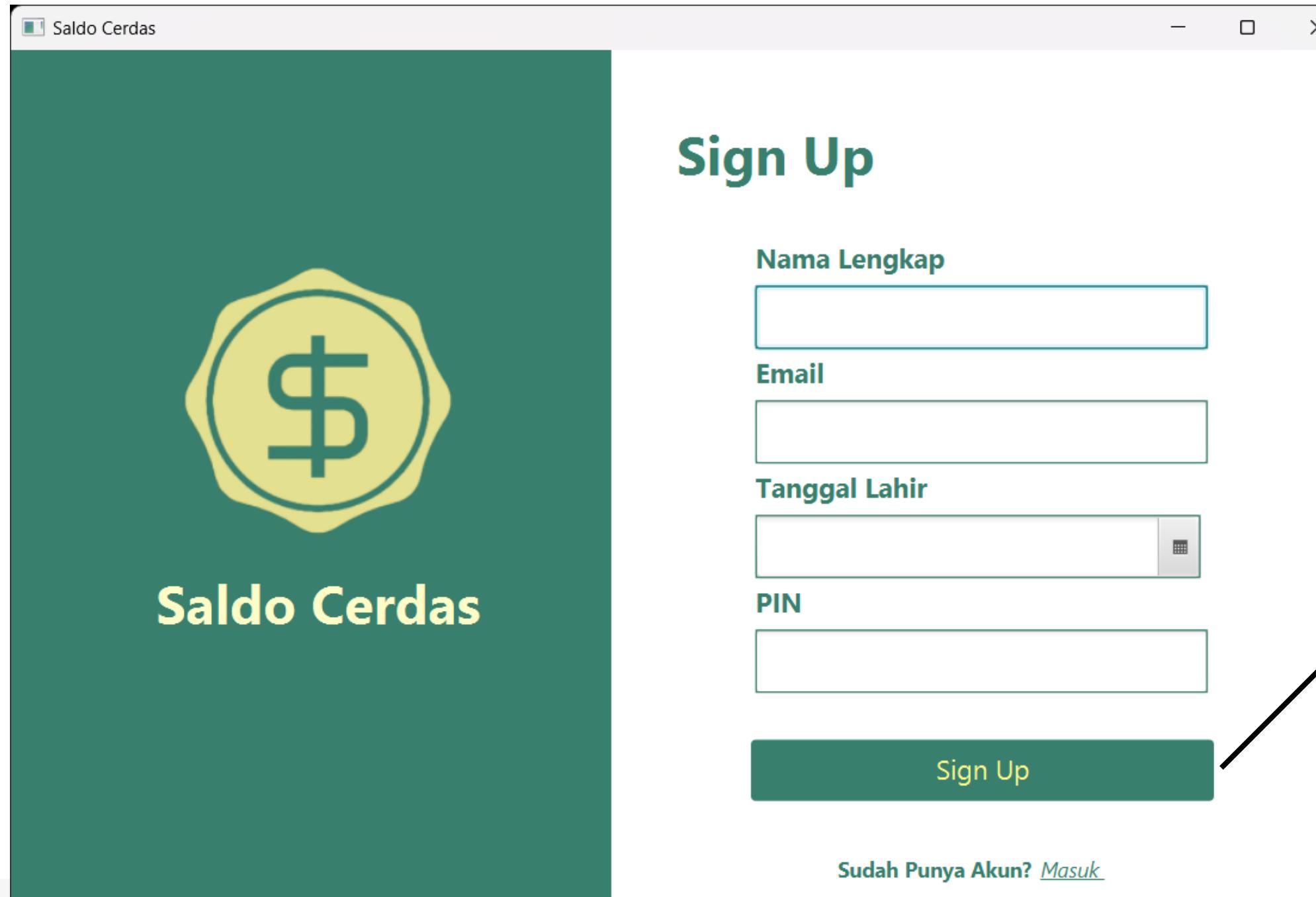
- Title Bar:** eclipse-workspace1 - SaldoCerdas/src/application/Pengeluaran.java - Eclipse IDE
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbar:** Standard Eclipse toolbar with various icons for file operations, search, and project management.
- Left Margin:** Shows line numbers from 29 to 71.
- Code Editor:** The main area displays the Java code for the Pengeluaran class. The code includes methods for getting and setting user ID, amount, note, and date, along with an additional method for getting the transaction ID.

```
29  public String getIdUser() {
30     User user = new User();
31     return user.getIduser();
32 }
33
34  public void setIdUser(String idUser) {
35     this.idUser = idUser;
36 }
37
38  public int getJumlah() {
39     return jumlah;
40 }
41
42  public void setJumlah(int jumlah) {
43     this.jumlah = jumlah;
44 }
45
46  public String getCatatan() {
47     return catatan;
48 }
49
50  public void setCatatan(String catatan) {
51     this.catatan = catatan;
52 }
53
54  public String getTanggal() {
55     return tanggal;
56 }
57
58  public void setTanggal(String tanggal) {
59     this.tanggal = tanggal;
60 }
61
62  public String getIdpengeluaran() {
63     return idpengeluaran;
64 }
65
66  public void setIdpengeluaran(String idpengeluaran) {
67     this.idpengeluaran = idpengeluaran;
68 }
69
70 }
71 }
```

- Right Margin:** Shows the Java code completion (Quick Assist) palette with suggestions like `User`, `idUser`, `jumlah`, `catatan`, `tanggal`, and `idpengeluaran`.

# CODING EXPLANATION FOR DB CONNECTIVITY & GUI

# SIGN UP



```
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);

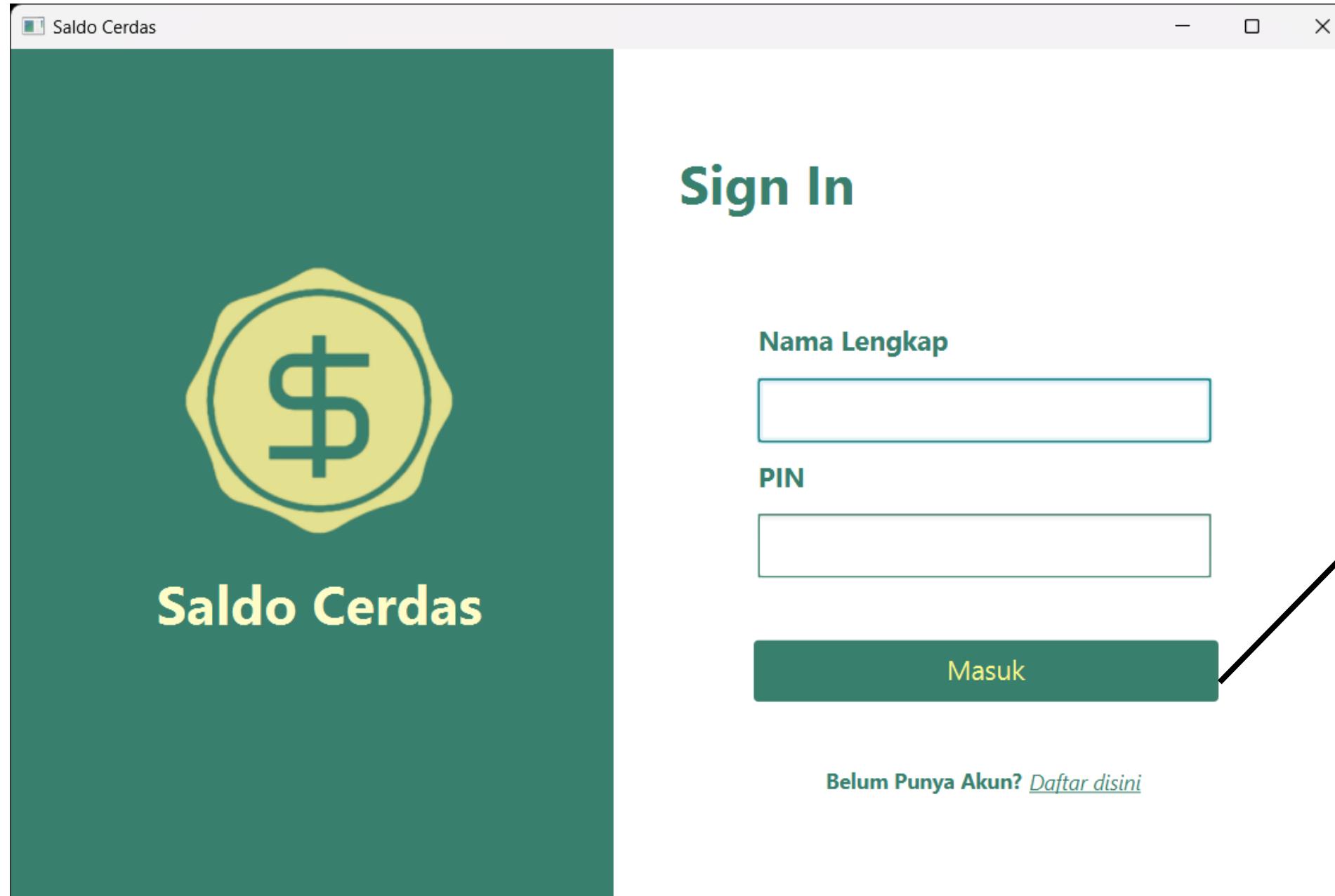
        preparedStatement.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            if (preparedStatement != null) {
                preparedStatement.close();
            }
        } catch (SQLException e) {
            e.printStackTrace();
     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

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

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

The screenshot shows the Saldo Cerdas application interface. On the left is a sidebar with icons for Home, Analisis, Pemasukan, Pengeluaran, and Profile. The main area displays two tables: 'Pemasukan' (Income) and 'Pengeluaran' (Expense). The 'Pemasukan' table has three rows with data: Uang Jajan (200000), Hasil Usaha (300000), and Gaji (1000000). The 'Pengeluaran' table is currently empty. Below the tables are 'Add Pemasukan' and 'Delete Pemasukan' buttons. Callout arrows point from these buttons to their corresponding Java code snippets.

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

Kategori	Jumlah	Catatan	Tanggal
----------	--------	---------	---------

```
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



```
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();
    }
}
```

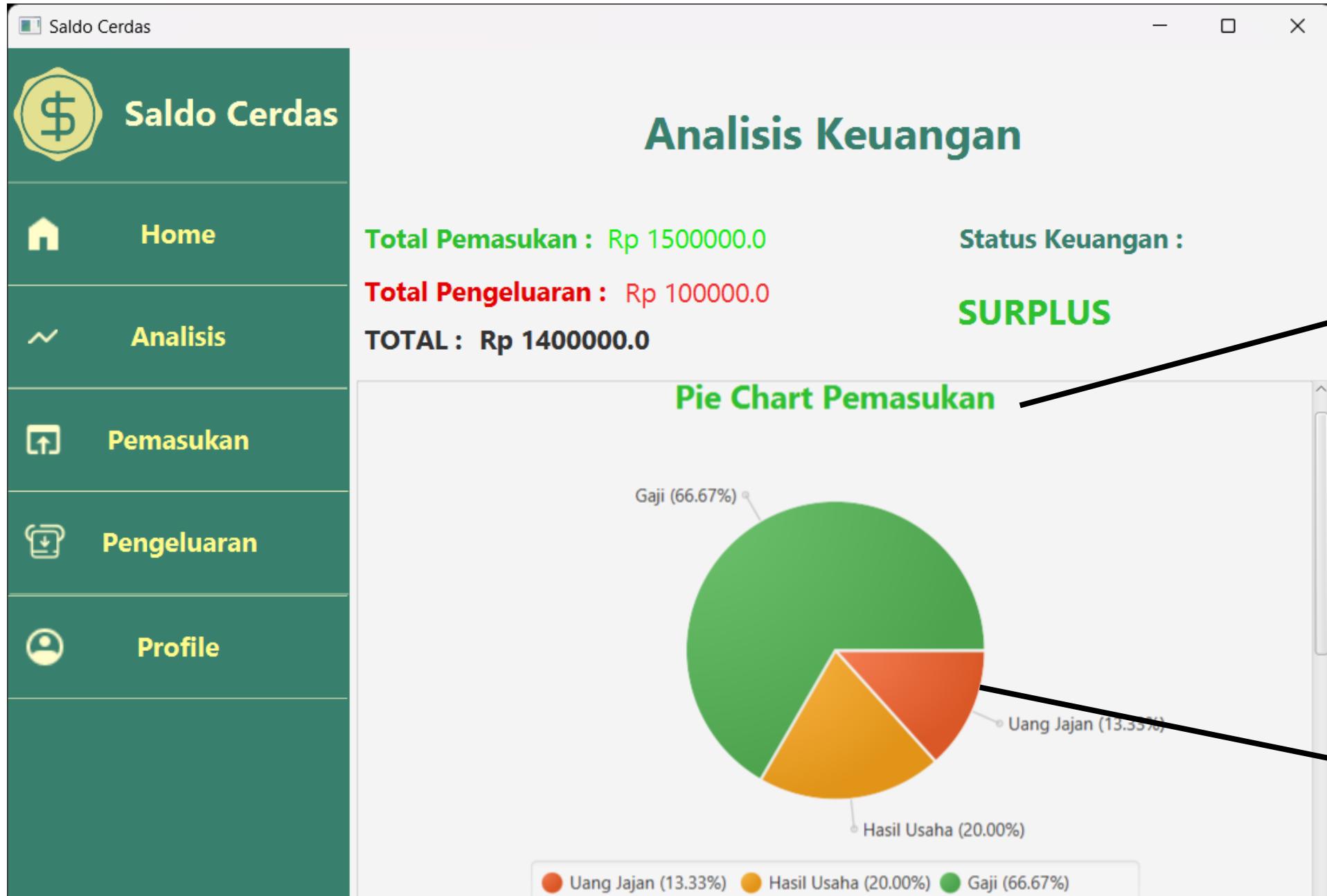
```
public static void hapusPemasukan(String idpemasukan) {
    Connection connection = null;
    PreparedStatement preparedStatement = null;

    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



```
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;
}
```

```
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");  
80         }  
81     } catch (SQLException e) {  
82         e.printStackTrace();  
83     }  
84 }
```

```
85     if (totalPemasukan > totalPengeluaran) {  
86         statusKeuangan.setText("SURPLUS");  
87         statusKeuangan.setStyle("-fx-text-fill : #2ebf30");  
88     } else if (totalPengeluaran > totalPemasukan) {  
89         statusKeuangan.setText("DEFISIT");  
90         statusKeuangan.setStyle("-fx-text-fill : #e40000");  
91     } else {  
92         statusKeuangan.setText("");  
93     }  
94  
95     statement.close();  
96     connection.close();  
97 } catch (SQLException e) {  
98     e.printStackTrace();  
99 }  
100 }
```

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) {
    showAlertDialog(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();

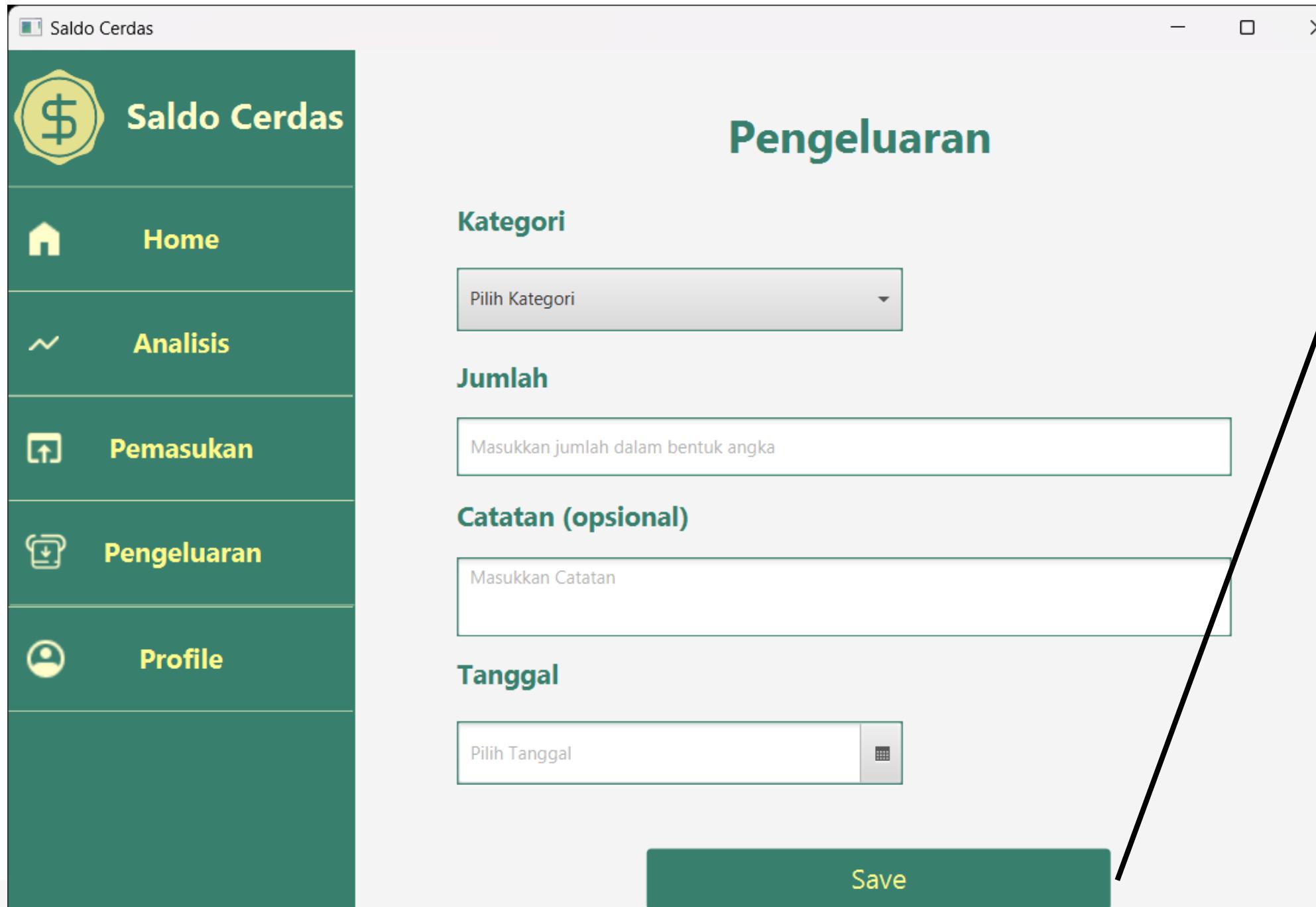
        String sql = "INSERT INTO pemasukan (idpemasukan, iduser, kategori, jumlah, catatan, tanggal) VALUES (?, ?, ?, ?, ?, ?)";
        preparedStatement = connection.prepareStatement(sql);

        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



```
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();

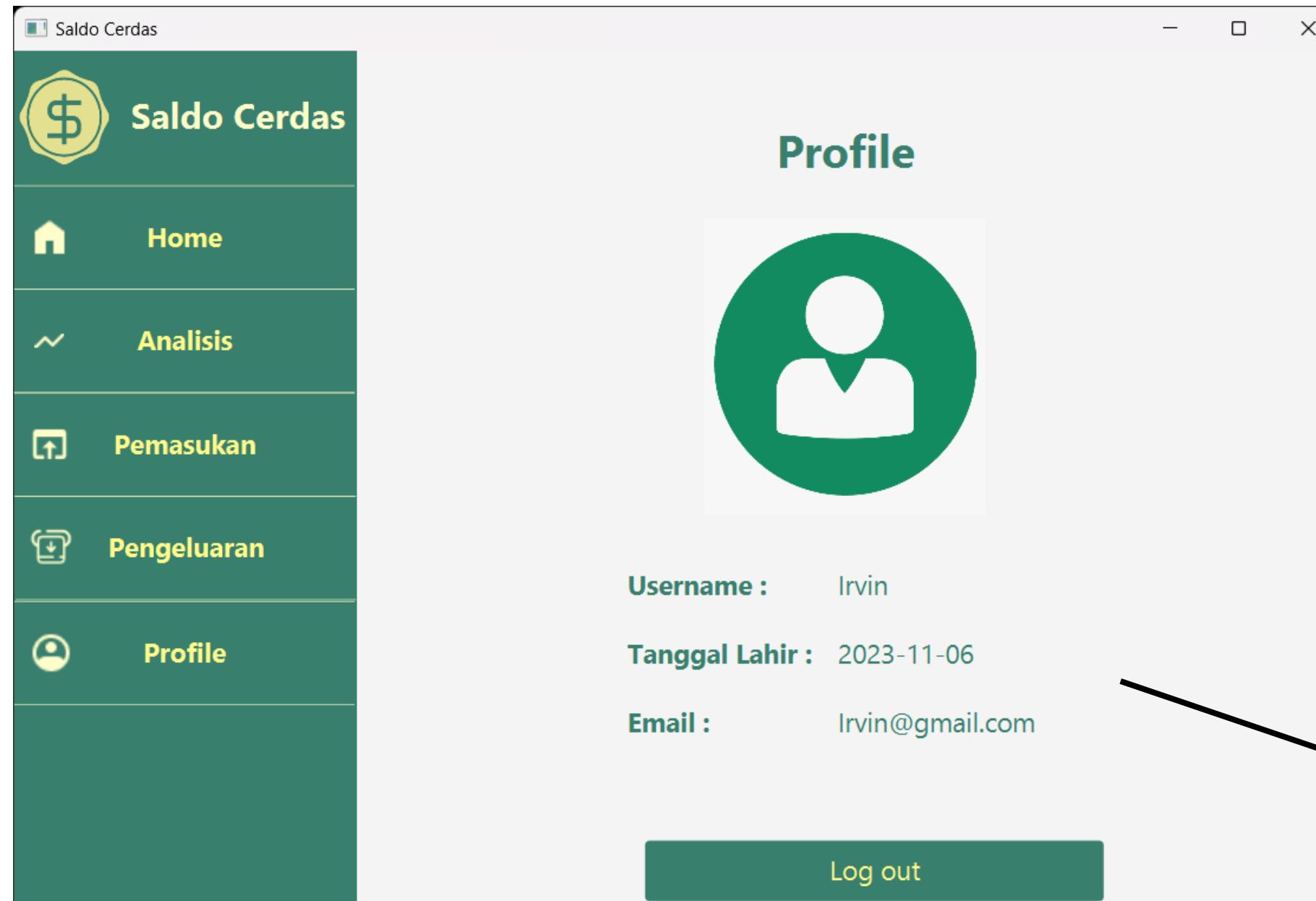
        String sql = "INSERT INTO pengeluaran (idpengeluaran, iduser, kategori, jumlah, catatan, tanggal) VALUES (?, ?, ?, ?, ?, ?)";
        preparedStatement = connection.prepareStatement(sql);

        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

```
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;
}
```

```
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 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 tanggal 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