

**LAB ASSIGNMENT 4**

**GROUP MEMBERS:**

MUHIBBA FATIMA SP24-BSE-138

ASNA AQEEL SP24-BSE-016

RUTABA HAYAT FA23-BSE-220

**SUBMITTED TO:**

SIR SHAHID BHATTI

**DATE:** 12/12/2024

**ATM**

**User class**

package com.example.demo30;

import java.io.Serializable;

import java.io.\*;

import java.util.ArrayList;

import java.util.HashMap;

public class User implements Serializable {

private String username;

private String password;

private double balance;

private ArrayList<String> transactionHistory;

public User(String username, String password, double balance) {

this.username = username;

this.password = password;

this.balance = balance;

this.transactionHistory = new ArrayList<>();

}

public String getUsername() {

return username;

}

public String getPassword() {

return password;

}

public double getBalance() {

return balance;

}

public void setBalance(double balance) {

this.balance = balance;

}

public ArrayList<String> getTransactionHistory() {

return transactionHistory;

}

public void addTransaction(String transaction) {

transactionHistory.add(transaction);

}

public void setPassword(String newPassword) {

this.password = newPassword;

}

}

**UserMnager Class**

package com.example.demo30;

import java.io.\*;

import java.util.HashMap;

public class UserManager {

private HashMap<String, User> users;

private final String dataFilePath = "users.dat";

public UserManager() {

users = readUserDataFromFile();

}

public boolean authenticateUser(String username, String password) {

User user = users.get(username);

return user != null && user.getPassword().equals(password);

}

public void createAccount(String username, String password, double balance) {

if (users.containsKey(username)) {

throw new IllegalArgumentException("Username already exists.");

}

users.put(username, new User(username, password, balance));

saveUserDataToFile();

}

public User getUser(String username) {

return users.get(username);

}

public void deleteAccount(String username) {

users.remove(username);

saveUserDataToFile();

}

private HashMap<String, User> readUserDataFromFile() {

try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(dataFilePath))) {

return (HashMap<String, User>) ois.readObject();

} catch (IOException | ClassNotFoundException e) {

return new HashMap<>();

}

}

public void saveUserDataToFile() {

try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(dataFilePath))) {

oos.writeObject(users);

} catch (IOException e) {

throw new RuntimeException("Error saving user data to file", e);

}

}

}

**Class AtmApp**

package com.example.demo30;

import javafx.animation.PauseTransition;

import javafx.application.Application;

import javafx.geometry.Insets;

import javafx.geometry.Pos;

import javafx.scene.Scene;

import javafx.scene.control.\*;

import javafx.scene.image.Image;

import javafx.scene.image.ImageView;

import javafx.scene.layout.\*;

import javafx.scene.text.Font;

import javafx.stage.Stage;

import javafx.util.Duration;

public class ATMApp extends Application {

private UserManager userManager;

private Stage primaryStage;

private BorderPane mainLayout;

private Label messageLabel;

@Override

public void start(Stage primaryStage) {

this.primaryStage = primaryStage;

this.userManager = new UserManager();

mainLayout = new BorderPane();

mainLayout.setStyle("-fx-background-color: #1c1c1c;");

ToolBar toolBar = createToolBar();

mainLayout.setTop(toolBar);

showMainMenu();

Scene scene = new Scene(mainLayout, 600, 400);

primaryStage.setTitle("ATM");

primaryStage.setScene(scene);

primaryStage.show();

}

private ToolBar createToolBar() {

Button homeButton = new Button();

homeButton.setStyle("-fx-background-color: transparent;");

Image homeIcon = new Image("file:/C:/Users/Admin/Downloads/homeIcon.png");

ImageView homeIconView = new ImageView(homeIcon);

homeIconView.setFitHeight(40);

homeIconView.setPreserveRatio(true);

homeButton.setGraphic(homeIconView);

homeButton.setOnAction(e -> showMainMenu());

Button feedbackButton = new Button();

feedbackButton.setStyle("-fx-background-color: transparent;");

Image feedbackIcon = new Image("file:/C:/Users/Admin/Downloads/feedback.png");

ImageView feedbackIconView = new ImageView(feedbackIcon);

feedbackIconView.setFitHeight(40);

feedbackIconView.setPreserveRatio(true);

feedbackButton.setGraphic(feedbackIconView);

feedbackButton.setOnAction(e -> showFeedbackForm());

Button logoutButton = new Button();

logoutButton.setStyle("-fx-background-color: transparent;");

Image logoutIcon = new Image("file:/C:/Users/Admin/Downloads/logout.png");

ImageView logoutIconView = new ImageView(logoutIcon);

logoutIconView.setFitHeight(40);

logoutIconView.setPreserveRatio(true);

logoutButton.setGraphic(logoutIconView);

logoutButton.setOnAction(e -> System.exit(0));

ToolBar toolBar = new ToolBar(homeButton, feedbackButton, logoutButton);

toolBar.setStyle("-fx-background-color: #a1d70a;");

toolBar.setPrefHeight(70);

return toolBar;

}

private void showMainMenu() {

VBox menu = new VBox(10);

menu.setAlignment(Pos.CENTER);

menu.setPadding(new Insets(20));

Button createAccountButton = new Button("Create an Account");

createAccountButton.setStyle("-fx-background-color: #a1d70a; -fx-text-fill: black; -fx-font-weight: bold; -fx-font-size: 18px; -fx-font-family: 'Times New Roman';");

createAccountButton.setPrefWidth(400);

createAccountButton.setPrefHeight(50);

Button loginButton = new Button("Login");

loginButton.setStyle("-fx-background-color: #a1d70a; -fx-text-fill: black; -fx-font-weight: bold; -fx-font-size: 18px; -fx-font-family: 'Times New Roman';");

loginButton.setPrefHeight(50);

loginButton.setPrefWidth(400);

createAccountButton.setOnAction(e -> showCreateAccountPanel());

loginButton.setOnAction(e -> showLoginPanel());

menu.getChildren().addAll(createAccountButton, loginButton);

mainLayout.setCenter(menu);

messageLabel = new Label("Welcome to the ATM APP");

messageLabel.setStyle("-fx-text-fill : #a1d70a; -fx-font-size: 44px; -fx-font-weight: bold; -fx-font-family: 'Times New Roman';");

mainLayout.setBottom(messageLabel);

BorderPane.setAlignment(messageLabel, Pos.CENTER);

}

private void showCreateAccountPanel() {

GridPane grid = createStyledGridPane();

Label usernameLabel = createStyledLabel("Username:", "#a1d70a");

TextField usernameField = new TextField();

Label passwordLabel = createStyledLabel("Password:", "#a1d70a");

PasswordField passwordField = new PasswordField();

Label balanceLabel = createStyledLabel("Initial Balance:", "#a1d70a");

TextField balanceField = new TextField();

Label accountTypeLabel = createStyledLabel("Account Type:", "#a1d70a");

ComboBox<String> accountTypeComboBox = new ComboBox<>();

accountTypeComboBox.getItems().addAll("Savings Account", "Current Account");

Button createButton = new Button("Create");

createButton.setStyle("-fx-background-color: #a1d70a; -fx-text-fill: black; -fx-font-weight: bold; -fx-font-size: 18px; -fx-font-family: 'Times New Roman';");

createButton.setPrefHeight(50);

createButton.setPrefWidth(200);

Button backButton = new Button("Back");

backButton.setStyle("-fx-background-color: #a1d70a; -fx-text-fill: black; -fx-font-weight: bold; -fx-font-size: 18px; -fx-font-family: 'Times New Roman';");

backButton.setPrefWidth(200);

backButton.setPrefHeight(50);

createButton.setOnAction(e -> {

try {

String username = usernameField.getText();

String password = passwordField.getText();

double balance = Double.parseDouble(balanceField.getText());

userManager.createAccount(username, password, balance);

messageLabel.setText("Account created successfully!");

messageLabel.setStyle("-fx-text-fill: #a1d70a;");

showMainMenu();

} catch (Exception ex) {

messageLabel.setText("Error: " + ex.getMessage());

messageLabel.setStyle("-fx-text-fill: #f44336;");

}

});

backButton.setOnAction(e -> showMainMenu());

grid.add(usernameLabel, 0, 0);

grid.add(usernameField, 1, 0);

grid.add(passwordLabel, 0, 1);

grid.add(passwordField, 1, 1);

grid.add(balanceLabel, 0, 2);

grid.add(balanceField, 1, 2);

grid.add(accountTypeLabel, 0, 3);

grid.add(accountTypeComboBox, 1, 3);

grid.add(createButton, 0, 4);

grid.add(backButton, 1, 4);

mainLayout.setCenter(grid);

mainLayout.setTop(createToolBar());

}

private void showLoginPanel() {

GridPane grid = createStyledGridPane();

Label usernameLabel = createStyledLabel("Username:", "#a1d70a");

TextField usernameField = new TextField();

Label passwordLabel = createStyledLabel("Password:", "#a1d70a");

PasswordField passwordField = new PasswordField();

Button loginButton = new Button("Login");

loginButton.setStyle("-fx-background-color: #a1d70a; -fx-text-fill: black; -fx-font-weight: bold; -fx-font-size: 18px; -fx-font-family: 'Times New Roman';");

loginButton.setPrefHeight(50);

loginButton.setPrefWidth(200);

Button backButton = new Button("Back");

backButton.setStyle("-fx-background-color: #a1d70a; -fx-text-fill: black; -fx-font-weight: bold; -fx-font-size: 18px; -fx-font-family: 'Times New Roman';");

backButton.setPrefWidth(200);

backButton.setPrefHeight(50);

loginButton.setOnAction(e -> {

try {

String username = usernameField.getText();

String password = passwordField.getText();

if (userManager.authenticateUser(username, password)) {

PauseTransition delay = new PauseTransition(Duration.seconds(1));

delay.setOnFinished(event -> {

messageLabel.setText("Hi, " + username + "! Welcome to the ATM.");

messageLabel.setStyle("-fx-text-fill: #a1d70a; -fx-font-size: 40px; -fx-font-weight: bold");

showATMMenu(username);

});

delay.play();

} else {

messageLabel.setText("Invalid username or password.");

messageLabel.setStyle("-fx-text-fill: #f44336; -fx-font-size: 40px; -fx-font-weight: bold");

}

} catch (Exception ex) {

messageLabel.setText("Error: " + ex.getMessage());

messageLabel.setStyle("-fx-text-fill: #f44336; -fx-font-size: 40px; -fx-font-weight: bold");

}

});

backButton.setOnAction(e -> showMainMenu());

grid.add(usernameLabel, 0, 0);

grid.add(usernameField, 1, 0);

grid.add(passwordLabel, 0, 1);

grid.add(passwordField, 1, 1);

grid.add(loginButton, 0, 2);

grid.add(backButton, 1, 2);

mainLayout.setCenter(grid);

mainLayout.setTop(createToolBar());

}

private void showATMMenu(String username) {

GridPane gridPane = new GridPane();

gridPane.setPadding(new Insets(20));

gridPane.setHgap(20);

gridPane.setVgap(20);

gridPane.setAlignment(Pos.CENTER);

Button withdrawButton = new Button("Withdraw");

withdrawButton.setStyle("-fx-background-color: #a1d70a; -fx-text-fill: black; -fx-font-weight: bold; -fx-font-size: 18px; -fx-font-family: 'Times New Roman';");

withdrawButton.setPrefHeight(60);

withdrawButton.setPrefWidth(300);

Button depositButton = new Button("Deposit");

depositButton.setStyle("-fx-background-color: #a1d70a; -fx-text-fill: black; -fx-font-weight: bold; -fx-font-size: 18px; -fx-font-family: 'Times New Roman';");

depositButton.setPrefHeight(60);

depositButton.setPrefWidth(300);

Button balanceButton = new Button("Balance Inquiry");

balanceButton.setStyle("-fx-background-color: #a1d70a; -fx-text-fill: black; -fx-font-weight: bold; -fx-font-size: 18px; -fx-font-family: 'Times New Roman';");

balanceButton.setPrefWidth(300);

balanceButton.setPrefHeight(60);

Button transferButton = new Button("Transfer Funds");

transferButton.setStyle("-fx-background-color: #a1d70a; -fx-text-fill: black; -fx-font-weight: bold; -fx-font-size: 18px; -fx-font-family: 'Times New Roman';");

transferButton.setPrefHeight(60);

transferButton.setPrefWidth(300);

Button deleteButton = new Button("Delete Account");

deleteButton.setStyle("-fx-background-color: #a1d70a; -fx-text-fill: black; -fx-font-weight: bold; -fx-font-size: 18px; -fx-font-family: 'Times New Roman';");

deleteButton.setPrefWidth(300);

deleteButton.setPrefHeight(60);

Button changePassword = new Button("Change Password");

changePassword.setStyle("-fx-background-color: #a1d70a; -fx-text-fill: black; -fx-font-weight: bold; -fx-font-size: 18px; -fx-font-family: 'Times New Roman';");

changePassword.setPrefHeight(60);

changePassword.setPrefWidth(300);

Button logoutButton = new Button("Logout");

logoutButton.setStyle("-fx-background-color: #a1d70a; -fx-text-fill: black; -fx-font-weight: bold; -fx-font-size: 18px; -fx-font-family: 'Times New Roman';");

logoutButton.setPrefWidth(300);

logoutButton.setPrefHeight(60);

withdrawButton.setOnAction(e -> performWithdrawal(username));

depositButton.setOnAction(e -> performDeposit(username));

balanceButton.setOnAction(e -> checkBalance(username));

deleteButton.setOnAction(e -> deleteAccount(username));

changePassword.setOnAction(e -> changePassword(username));

logoutButton.setOnAction(e -> showMainMenu());

gridPane.add(withdrawButton, 0, 0);

gridPane.add(depositButton, 1, 0);

gridPane.add(balanceButton, 2, 0);

gridPane.add(transferButton, 0, 1);

gridPane.add(deleteButton, 1, 1);

gridPane.add(changePassword, 2, 1);

gridPane.add(logoutButton, 1, 2);

mainLayout.setCenter(gridPane);

mainLayout.setTop(createToolBar());

}

private GridPane createStyledGridPane() {

GridPane grid = new GridPane();

grid.setPadding(new Insets(20));

grid.setHgap(15);

grid.setVgap(15);

grid.setAlignment(Pos.CENTER);

return grid;

}

private Label createStyledLabel(String text, String color) {

Label label = new Label(text);

label.setFont(Font.font("Arial", 18));

label.setStyle("-fx-text-fill: " + color + ";");

return label;

}

private void performWithdrawal(String username) {

try {

TextInputDialog dialog = new TextInputDialog();

dialog.setTitle("Withdrawal");

dialog.setHeaderText("Enter withdrawal amount:");

dialog.setContentText("Amount:");

dialog.showAndWait().ifPresent(amountStr -> {

try {

double amount = Double.parseDouble(amountStr);

User user = userManager.getUser(username);

if (amount <= 0 || amount > user.getBalance()) {

messageLabel.setText("Error: Invalid withdrawal amount.");

messageLabel.setStyle("-fx-text-fill: #f44336; -fx-font-size: 40px; -fx-font-weight: bold");

return;

}

user.setBalance(user.getBalance() - amount);

userManager.saveUserDataToFile();

messageLabel.setText("Withdrawal successful. Remaining balance: " + user.getBalance());

messageLabel.setStyle("-fx-text-fill: #a1d70a; -fx-font-size: 40px; -fx-font-weight: bold");

} catch (NumberFormatException ex) {

messageLabel.setText("Error: Invalid input. Please enter a number.");

messageLabel.setStyle("-fx-text-fill: #f44336;");

}

});

} catch (Exception e) {

messageLabel.setText("Error: " + e.getMessage());

messageLabel.setStyle("-fx-text-fill: #f44336;");

}

}

private void performDeposit(String username) {

try {

TextInputDialog dialog = new TextInputDialog();

dialog.setTitle("Deposit");

dialog.setHeaderText("Enter deposit amount:");

dialog.setContentText("Amount:");

dialog.showAndWait().ifPresent(amountStr -> {

try {

double amount = Double.parseDouble(amountStr);

if (amount <= 0) {

messageLabel.setText("Error: Deposit amount must be positive.");

messageLabel.setStyle("-fx-text-fill: #f44336; -fx-font-size: 40px; -fx-font-weight: bold");

return;

}

User user = userManager.getUser(username);

user.setBalance(user.getBalance() + amount);

userManager.saveUserDataToFile();

messageLabel.setText("Deposit successful. New balance: " + user.getBalance());

messageLabel.setStyle("-fx-text-fill: #a1d70a; -fx-font-size: 40px; -fx-font-weight: bold");

} catch (NumberFormatException ex) {

messageLabel.setText("Error: Invalid input. Please enter a number.");

messageLabel.setStyle("-fx-text-fill: #f44336; -fx-font-size: 40px; -fx-font-weight: bold");

}

});

} catch (Exception e) {

messageLabel.setText("Error: " + e.getMessage());

messageLabel.setStyle("-fx-text-fill: #f44336;");

}

}

private void checkBalance(String username) {

try {

User user = userManager.getUser(username);

messageLabel.setText("Current balance: " + user.getBalance());

messageLabel.setStyle("-fx-text-fill: #a1d70a; -fx-font-size: 40px; -fx-font-weight: bold");

} catch (Exception e) {

messageLabel.setText("Error: " + e.getMessage());

messageLabel.setStyle("-fx-text-fill: #f44336; -fx-font-size: 40px; -fx-font-weight: bold");

}

}

private void deleteAccount(String username) {

VBox confirmationBox = new VBox(20);

confirmationBox.setAlignment(Pos.CENTER);

confirmationBox.setPadding(new Insets(20));

Label confirmationLabel = new Label("Are you sure you want to delete your account?");

confirmationLabel.setStyle("-fx-font-size: 18px; -fx-text-fill: #f44336; -fx-font-weight: bold;");

Label warningLabel = new Label("This action cannot be undone.");

warningLabel.setStyle("-fx-font-size: 14px; -fx-text-fill: #f44336;");

Button confirmButton = new Button("Confirm");

confirmButton.setStyle("-fx-background-color: a1d70a; -fx-text-fill: #ff4d4d; -fx-font-weight: bold; -fx-font-size: 18px; -fx-font-family: 'Times New Roman'; ");

confirmButton.setPrefHeight(50);

confirmButton.setPrefWidth(200);

Button cancelButton = new Button("Cancel");

cancelButton.setStyle("-fx-background-color: a1d70a; -fx-text-fill: #ff4d4d; -fx-font-weight: bold; -fx-font-size: 18px; -fx-font-family: 'Times New Roman';");

cancelButton.setPrefWidth(200);

cancelButton.setPrefHeight(50);

confirmButton.setOnAction(e -> {

try {

userManager.deleteAccount(username);

messageLabel.setText("Account deleted successfully.");

messageLabel.setStyle("-fx-text-fill: #a1d70a;");

showMainMenu();

} catch (Exception ex) {

messageLabel.setText("Error: " + ex.getMessage());

messageLabel.setStyle("-fx-text-fill: #f44336;");

}

});

cancelButton.setOnAction(e -> showATMMenu(username));

confirmationBox.getChildren().addAll(confirmationLabel, warningLabel, confirmButton, cancelButton);

mainLayout.setCenter(confirmationBox);

}

private void changePassword(String username) {

try {

TextInputDialog currentPasswordDialog = new TextInputDialog();

currentPasswordDialog.setTitle("Change Password");

currentPasswordDialog.setHeaderText("Enter current password:");

currentPasswordDialog.setContentText("Current Password:");

currentPasswordDialog.showAndWait().ifPresent(currentPassword -> {

try {

User user = userManager.getUser(username);

if (!user.getPassword().equals(currentPassword)) {

messageLabel.setText("Error: Current password is incorrect.");

messageLabel.setStyle("-fx-text-fill: #f44336; -fx-font-size: 40px; -fx-font-weight: bold");

return;

}

TextInputDialog newPasswordDialog = new TextInputDialog();

newPasswordDialog.setTitle("Change Password");

newPasswordDialog.setHeaderText("Enter new password:");

newPasswordDialog.setContentText("New Password:");

newPasswordDialog.showAndWait().ifPresent(newPassword -> {

try {

if (newPassword.isEmpty()) {

messageLabel.setText("Error: Password cannot be empty.");

messageLabel.setStyle("-fx-text-fill: #f44336; -fx-font-size: 40px; -fx-font-weight: bold");

return;

}

user.setPassword(newPassword);

userManager.saveUserDataToFile();

messageLabel.setText("Password changed successfully.");

messageLabel.setStyle("-fx-text-fill: #a1d70a; -fx-font-size: 40px; -fx-font-weight: bold");

} catch (Exception ex) {

messageLabel.setText("Error: " + ex.getMessage());

messageLabel.setStyle("-fx-text-fill: #f44336; -fx-font-size: 40px; -fx-font-weight: bold");

}

});

} catch (Exception ex) {

messageLabel.setText("Error: " + ex.getMessage());

messageLabel.setStyle("-fx-text-fill: #f44336; -fx-font-size: 40px; -fx-font-weight: bold");

}

});

} catch (Exception e) {

messageLabel.setText("Error: " + e.getMessage());

messageLabel.setStyle("-fx-text-fill: #f44336;");

}

}

private void showFeedbackForm() {

VBox feedbackLayout = new VBox(20);

feedbackLayout.setAlignment(Pos.CENTER);

feedbackLayout.setPadding(new Insets(20));

Label feedbackLabel = new Label("We value your feedback!");

feedbackLabel.setStyle("-fx-font-size: 24px; -fx-font-weight: bold; -fx-text-fill: #a1d70a;");

TextArea feedbackTextArea = new TextArea();

feedbackTextArea.setPromptText("Enter your feedback here...");

feedbackTextArea.setWrapText(true);

feedbackTextArea.setPrefHeight(150);

feedbackTextArea.setStyle("-fx-font-size: 16px;");

Button submitButton = new Button("Submit Feedback");

submitButton.setStyle("-fx-background-color: #a1d70a; -fx-text-fill: black; -fx-font-weight: bold; -fx-font-size: 18px; -fx-font-family: 'Times New Roman';");

submitButton.setPrefWidth(200);

submitButton.setOnAction(e -> {

String feedback = feedbackTextArea.getText();

if (feedback.isEmpty()) {

messageLabel.setText("Please provide some feedback.");

messageLabel.setStyle("-fx-text-fill: #f44336;");

} else {

messageLabel.setText("Thank you for your feedback!");

messageLabel.setStyle("-fx-text-fill: #a1d70a;");

}

});

Button backButton = new Button("Back");

backButton.setStyle("-fx-background-color: #a1d70a; -fx-text-fill: black; -fx-font-weight: bold; -fx-font-size: 18px; -fx-font-family: 'Times New Roman';");

backButton.setPrefWidth(200);

backButton.setOnAction(e -> showMainMenu());

feedbackLayout.getChildren().addAll(feedbackLabel, feedbackTextArea, submitButton, backButton);

mainLayout.setCenter(feedbackLayout);

}

public static void main(String[] args) {

launch(args);