**COMSATS University Islamabad, Lahore Campus**

**Subject :** Object Oriented Programming

**Assignment :** 4 (LAB)

**Submitted To :** Sir Shahid Bhatti

**Muhammad Ahmad Iqbal(SP24-BSE-065)**

**Muhammad Waqar(SP24-BSE-091)**

**Submitted By :**

**Section :**A

**Date of Submition:20-12-2024**

***HelloApplication.java***

**package com.example.demo;  
  
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.stage.Stage;  
import javafx.scene.control.Alert;  
  
  
import java.io.\*;  
  
  
import static com.example.demo.BooksManagement.\*;  
import static com.example.demo.BooksManagement.*showBooks*;  
  
public class HelloApplication extends Application {  
  
 @Override  
 public void start(Stage stage) {  
  
 File file = new File("user1.ser");  
 // file.mkdir();  
  
 Alert alert = new Alert(Alert.AlertType.*INFORMATION*);  
 alert.setTitle("Information Dialog");  
 // alert.setHeaderText("This is a header text");  
 // alert.setContentText("This is the content of the alert!");  
  
  
  
  
  
  
  
  
  
 Label emailLabelone = new Label("Email: ");  
 emailLabelone.setStyle("-fx-font-size: 16px; -fx-font-weight: bold;-fx-text-fill: #ffffff");  
 TextField emailfieldone =new TextField();  
 emailfieldone.setStyle("-fx-padding: 10px; -fx-border-color: #0073e6;-fx-background-radius: 5px; -fx-border-width: 2px;-fx-font-size: 14px;");  
 emailfieldone.setPromptText("Enter your email");  
  
  
  
 Label passwordlabel = new Label("Password: ");  
 passwordlabel.setStyle("-fx-font-size: 16px; -fx-font-weight: bold; -fx-text-fill: #ffffff;");  
  
 PasswordField passwordfield = new PasswordField();  
 passwordfield.setPromptText("Enter your password");  
 passwordfield.setStyle("-fx-padding: 10px; -fx-background-radius: 5px; -fx-border-color: #0073e6; -fx-border-width: 2px; -fx-font-size: 14px;");  
  
  
  
  
  
  
  
  
  
  
 Button loginbutton = new Button("Login", new ImageView(new Image("file:Icon 1/login.png"))); // Add correct path  
 loginbutton.setStyle("-fx-font-weight: bold; -fx-font-size: 16px; -fx-text-fill: white; -fx-background-color: linear-gradient(#0073e6, #005bb5); -fx-background-radius: 30px; -fx-padding: 10px 20px; -fx-border-radius: 30px;");  
 loginbutton.setOnMouseEntered(e -> loginbutton.setStyle("-fx-font-weight: bold; -fx-font-size: 16px; -fx-text-fill: white; -fx-background-color: linear-gradient(#005bb5, #0073e6); -fx-background-radius: 30px; -fx-padding: 10px 20px; -fx-border-radius: 30px;"));  
 loginbutton.setOnMouseExited(e -> loginbutton.setStyle("-fx-font-weight: bold; -fx-font-size: 16px; -fx-text-fill: white; -fx-background-color: linear-gradient(#0073e6, #005bb5); -fx-background-radius: 30px; -fx-padding: 10px 20px; -fx-border-radius: 30px;"));  
   
  
 loginbutton.setOnAction(e->{  
 String emailfieldText = emailfieldone.getText();  
 String passswordfieldText = passwordfield.getText();  
 emailfieldText.trim();  
 passswordfieldText.trim();  
  
 try {  
 if(validity(file,emailfieldText,passswordfieldText)){  
  
// alert.setHeaderText("Login Successfull!");  
// alert.setContentText("");  
// alert.show();  
  
 emailfieldone.clear();  
 passwordfield.clear();  
  
 Stage stage3 = new Stage();  
 GridPane pane = new GridPane();  
 VBox vbox = new VBox(20);  
 vbox.setPadding(new Insets(20));  
 vbox.setAlignment(Pos.*CENTER*);  
  
 Label welcomeLabel = new Label("Welcome to Library Management, " + emailfieldText + "!");  
 welcomeLabel.setStyle("-fx-font-size: 46px; -fx-font-weight: bold; -fx-text-fill: #82da27; -fx-background-color: #000000; -fx-background-radius: 10px;");  
  
  
 Button viewBooksButton = createStyledButton("View Books", "#4CAF50", "#357A38"); // Green  
 Button borrowBookButton = createStyledButton("Borrow Book", "#2196F3", "#1769AA"); // Blue  
 Button returnBookButton = createStyledButton("Return Book", "#FFC107", "#C79100"); // Yellow  
 Button addBookButton = createStyledButton("Add Book", "#9C27B0", "#6A0080"); // Purple  
 Button logoutButton = createStyledButton("Logout", "#F44336", "#AA2E25"); // Red  
 Button searchButton = createStyledButton("Search Book", "#82DA27FF", "#E91E63");  
  
 searchButton.setOnAction(t -> *searchBook*());  
  
 viewBooksButton.setOnAction(t -> {  
 try {  
 *showBooks*();  
 } catch (FileNotFoundException ex) {  
 throw new RuntimeException(ex);  
 }  
 });  
 borrowBookButton.setOnAction(t -> *borrowBook*(emailfieldText));  
 returnBookButton.setOnAction(t -> *returnBook*(emailfieldText));  
 addBookButton.setOnAction(t-> *addBook*());  
 logoutButton.setOnAction(t -> {  
 stage3.close();  
 stage.show();  
 });  
  
  
 vbox.getChildren().addAll(welcomeLabel, viewBooksButton, borrowBookButton, returnBookButton, addBookButton,searchButton, logoutButton);  
  
  
 BackgroundImage backgroundImage = new BackgroundImage(  
 new Image("Icon 1/Library\_Book\_532388\_1366x768.jpg"),  
 BackgroundRepeat.*NO\_REPEAT*,  
 BackgroundRepeat.*NO\_REPEAT*,  
 BackgroundPosition.*CENTER*,  
 new BackgroundSize(BackgroundSize.*AUTO*, BackgroundSize.*AUTO*, false, false, true, true)  
 );  
 vbox.setBackground(new Background(backgroundImage));  
  
 Scene scene1 = new Scene(vbox, 1200, 620);  
 stage3.setTitle("Library Dashboard");  
 stage3.setScene(scene1);  
 stage.close();  
 stage3.show();  
  
 }  
 else{  
 alert.setHeaderText("Login Failed!");  
 alert.setContentText("Wrong Username or Password!");  
 alert.show();  
 emailfieldone.clear();  
 passwordfield.clear();  
 System.*out*.println("Wrong username or password");  
 }  
 } catch (IOException ex) {  
 throw new RuntimeException(ex);  
 }  
  
  
  
  
 });  
  
  
  
  
  
  
  
  
  
  
 Button signupbutton = new Button("Signup"); // Add correct path  
 signupbutton.setStyle("-fx-font-weight: bold; -fx-font-size: 16px; -fx-text-fill: white; -fx-background-color: linear-gradient(#e60050, #b3003a); -fx-background-radius: 30px; -fx-padding: 10px 20px; -fx-border-radius: 30px;");  
 signupbutton.setOnMouseEntered(e -> signupbutton.setStyle("-fx-font-weight: bold; -fx-font-size: 16px; -fx-text-fill: white; -fx-background-color: linear-gradient(#b3003a, #e60050); -fx-background-radius: 30px; -fx-padding: 10px 20px; -fx-border-radius: 30px;"));  
 signupbutton.setOnMouseExited(e -> signupbutton.setStyle("-fx-font-weight: bold; -fx-font-size: 16px; -fx-text-fill: white; -fx-background-color: linear-gradient(#e60050, #b3003a); -fx-background-radius: 30px; -fx-padding: 10px 20px; -fx-border-radius: 30px;"));  
   
  
  
 // ------------------------------------------------------------------  
 signupbutton.setOnAction(e->{  
  
 Stage stage2 = new Stage();  
  
   
 Label emaillabel1 = new Label("Username: ");  
 emaillabel1.setStyle("-fx-font-size: 16px; -fx-font-weight: bold; -fx-text-fill: #ffffff;");  
  
 TextField emailfield1 = new TextField();  
 emailfield1.setPromptText("Enter new username");  
 emailfield1.setStyle("-fx-padding: 10px; -fx-background-radius: 5px; -fx-border-color: #0073e6; -fx-border-width: 2px; -fx-font-size: 14px;");  
  
 Label passwordlabel1 = new Label("Password: ");  
 passwordlabel1.setStyle("-fx-font-size: 16px; -fx-font-weight: bold; -fx-text-fill: #ffffff;");  
  
 PasswordField passwordfield1 = new PasswordField();  
 passwordfield1.setPromptText("Enter new password");  
 passwordfield1.setStyle("-fx-padding: 10px; -fx-background-radius: 5px; -fx-border-color: #0073e6; -fx-border-width: 2px; -fx-font-size: 14px;");  
  
  
 Button createaccount = new Button("Create Account");  
 createaccount.setStyle("-fx-font-weight: bold; -fx-font-size: 16px; -fx-text-fill: white; -fx-background-color: linear-gradient(#e60050, #b3003a); -fx-background-radius: 30px; -fx-padding: 10px 20px; -fx-border-radius: 30px;");  
 createaccount.setOnMouseEntered(t -> createaccount.setStyle("-fx-font-weight: bold; -fx-font-size: 16px; -fx-text-fill: white; -fx-background-color: linear-gradient(#b3003a, #e60050); -fx-background-radius: 30px; -fx-padding: 10px 20px; -fx-border-radius: 30px;"));  
 createaccount.setOnMouseExited(t -> createaccount.setStyle("-fx-font-weight: bold; -fx-font-size: 16px; -fx-text-fill: white; -fx-background-color: linear-gradient(#e60050, #b3003a); -fx-background-radius: 30px; -fx-padding: 10px 20px; -fx-border-radius: 30px;"));  
  
 // createaccount  
 createaccount.setOnAction(t->{  
 String username = emailfield1.getText();  
 String password = passwordfield1.getText();  
 username.trim();  
 password.trim();  
 if(!(username.isEmpty()||password.isEmpty())) {  
 emailfield1.clear();  
 passwordfield1.clear();  
  
 try {  
 *write*(file, username, password);  
 } catch (IOException ex) {  
 throw new RuntimeException(ex);  
 }  
 // alert.setContentText("You have successfully sign up!");  
 alert.setHeaderText("You have successfully sign up!");  
 alert.setContentText("");  
 alert.show();  
 stage2.close();  
 }  
 else{  
 alert.setHeaderText("Fill both");  
 alert.setContentText("");  
 alert.show();  
 }  
  
  
 });  
  
  
  
  
 GridPane gridpane = new GridPane();  
 gridpane.setPadding(new Insets(20));  
 gridpane.setHgap(20);  
 gridpane.setVgap(20);  
 gridpane.setAlignment(Pos.*CENTER*);  
  
  
 gridpane.add(emaillabel1, 0, 0);  
 gridpane.add(emailfield1, 1, 0);  
 gridpane.add(passwordlabel1, 0, 1);  
 gridpane.add(passwordfield1, 1, 1);  
 // gridpane.add(loginbutton, 0, 2);  
 gridpane.add(createaccount, 1, 2);  
  
  
  
 StackPane stackPane1 = new StackPane();  
 stackPane1.getChildren().add(gridpane);  
 StackPane.*setMargin*(gridpane, new Insets(0,200,0,200));  
  
 BackgroundImage backgroundImage1 = new BackgroundImage(  
 new Image("Icon 1/123456.png"),  
 BackgroundRepeat.*NO\_REPEAT*,  
 BackgroundRepeat.*NO\_REPEAT*,  
 BackgroundPosition.*CENTER*,  
 new BackgroundSize(BackgroundSize.*AUTO*, BackgroundSize.*AUTO*, false, false, true, true)  
 );  
 stackPane1.setBackground(new Background(backgroundImage1));  
  
  
 Scene scene = new Scene(stackPane1, 800, 620);  
 stage2.setTitle("Library Management System");  
 stage2.setScene(scene);  
 stage2.show();  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
 });  
  
 // GridPane layout  
 GridPane grid = new GridPane();  
 grid.setPadding(new Insets(20));  
 grid.setHgap(20);  
 grid.setVgap(20);  
 grid.setAlignment(Pos.*CENTER*);  
  
  
 grid.add(emailLabelone, 0, 0);  
 grid.add(emailfieldone, 1, 0);  
 grid.add(passwordlabel, 0, 1);  
 grid.add(passwordfield, 1, 1);  
 grid.add(loginbutton, 0, 2);  
 grid.add(signupbutton, 1, 2);  
  
  
 StackPane stackPane = new StackPane();  
 stackPane.getChildren().add(grid);  
 StackPane.*setMargin*(grid, new Insets(0,0,0,290));  
  
  
 BackgroundImage backgroundImage = new BackgroundImage(  
 new Image("Icon 1/Login background.png"),  
 BackgroundRepeat.*NO\_REPEAT*,  
 BackgroundRepeat.*NO\_REPEAT*,  
 BackgroundPosition.*CENTER*,  
 new BackgroundSize(BackgroundSize.*AUTO*, BackgroundSize.*AUTO*, false, false, true, true)  
 );  
 stackPane.setBackground(new Background(backgroundImage));  
  
 // Scene setup  
 Scene scene = new Scene(stackPane, 1200, 620);  
 stage.setTitle("Library Management System");  
 stage.setScene(scene);  
 stage.show();  
 }  
  
 public static void main(String[] args) {  
 *launch*();  
 }  
  
 public static void write(File file , String text1,String text2) throws IOException {  
 if(!file.exists()){  
 file.createNewFile();  
 }  
 else {  
  
 try (BufferedWriter writer = new BufferedWriter(new FileWriter(file,true))) {  
 writer.write(text1 + " " + text2);  
 //writer.write("");  
 // writer.write(text2);  
 writer.newLine();  
 } catch (FileNotFoundException e) {  
 throw new RuntimeException(e);  
 }  
 }  
 }  
  
 public boolean validity(File file ,String username, String password) throws FileNotFoundException {  
 try(BufferedReader reader = new BufferedReader(new FileReader(file))){  
 String line;  
 while ((line = reader.readLine()) != null) {  
 String[] space = line.split(" ");  
 if (space[0].equals(username) && space[1].equals(password)) {  
 return true;  
 }  
 }  
 } catch (FileNotFoundException e) {  
 throw new RuntimeException();  
 } catch (IOException e) {  
 throw new RuntimeException(e);  
 }  
 return false;  
 }  
  
 public void openDashboard(Stage stage3, String username) {  
  
 }  
 private Button createStyledButton(String text, String bgColor, String hoverColor) {  
 Button button = new Button(text);  
 button.setStyle(  
 "-fx-background-color: " + bgColor + ";" +  
 "-fx-text-fill: white;" +  
 "-fx-font-weight: bold;" +  
 "-fx-font-size: 14px;" +  
 "-fx-background-radius: 20px;" +  
 "-fx-padding: 10 20;"  
 );  
  
 // Adding hover effect  
 button.setOnMouseEntered(e -> button.setStyle(  
 "-fx-background-color: " + hoverColor + ";" +  
 "-fx-text-fill: white;" +  
 "-fx-font-weight: bold;" +  
 "-fx-font-size: 14px;" +  
 "-fx-background-radius: 20px;" +  
 "-fx-padding: 10 20;"  
 ));  
 button.setOnMouseExited(e -> button.setStyle(  
 "-fx-background-color: " + bgColor + ";" +  
 "-fx-text-fill: white;" +  
 "-fx-font-weight: bold;" +  
 "-fx-font-size: 14px;" +  
 "-fx-background-radius: 20px;" +  
 "-fx-padding: 10 20;"  
 ));  
  
 return button;  
 }  
  
  
  
  
  
  
}**

***BooksManagement.java***

**package com.example.demo;  
  
import javafx.geometry.Insets;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.\*;  
import javafx.scene.image.Image;  
import javafx.scene.layout.\*;  
import javafx.stage.Stage;  
  
import javafx.collections.FXCollections;  
import javafx.collections.ObservableList;  
import javafx.geometry.Insets;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.Label;  
import javafx.scene.control.ScrollPane;  
import javafx.scene.control.TableColumn;  
import javafx.scene.control.TableView;  
import javafx.scene.control.cell.PropertyValueFactory;  
import javafx.scene.image.Image;  
import javafx.scene.layout.Background;  
import javafx.scene.layout.BackgroundImage;  
import javafx.scene.layout.BackgroundPosition;  
import javafx.scene.layout.BackgroundRepeat;  
import javafx.scene.layout.BackgroundSize;  
import javafx.scene.layout.VBox;  
import javafx.stage.Stage;  
import java.io.BufferedReader;  
import java.io.FileReader;  
import java.io.FileNotFoundException;  
import java.io.IOException;  
  
import java.io.\*;  
import java.util.Scanner;  
  
  
  
public class BooksManagement {  
  
  
  
  
  
  
 public static void addBook() {  
 Stage addBookStage = new Stage();  
 VBox layout = new VBox(20);  
 layout.setPadding(new Insets(20));  
 layout.setAlignment(Pos.*CENTER*);  
  
 File file2 = new File("Book.txt");  
 Alert alert = new Alert(Alert.AlertType.*WARNING*);  
  
 Label bookLabel = new Label("Enter Book Title:");  
 TextField bookField = new TextField();  
 Label bookIdLabel = new Label("Enter Book ID:");  
 TextField bookIdField = new TextField();  
 Label authorNameLabel = new Label("Enter Author Name:");  
 TextField authorNameField = new TextField();  
  
 Label isAvailableLabel = new Label("Is Available:");  
 ToggleGroup availabilityGroup = new ToggleGroup();  
 RadioButton availableButton = new RadioButton("Yes");  
 availableButton.setToggleGroup(availabilityGroup);  
 RadioButton notAvailableButton = new RadioButton("No");  
 notAvailableButton.setToggleGroup(availabilityGroup);  
  
 Button addButton = new Button("Add Book");  
  
 addButton.setOnAction(e -> {  
 String bookTitle = bookField.getText().trim();  
 String bookID = bookIdField.getText().trim();  
 String authorName = authorNameField.getText().trim();  
 RadioButton selectedButton = (RadioButton) availabilityGroup.getSelectedToggle();  
 String isAvailable = selectedButton != null ? selectedButton.getText() : "No";  
  
 if (!bookTitle.isEmpty() && !bookID.isEmpty() && !authorName.isEmpty()) {  
 if (*isBookIdDuplicate*(file2, bookID)) {  
 alert.setAlertType(Alert.AlertType.*WARNING*);  
 alert.setHeaderText("Duplicate Book ID");  
 alert.setContentText("The Book ID already exists. Please use a different ID.");  
 alert.show();  
 return;  
 }  
  
 try (BufferedWriter writer = new BufferedWriter(new FileWriter(file2, true))) {  
  
 writer.write(bookID + "," + bookTitle + "," + authorName + "," + isAvailable);  
 writer.newLine();  
  
 alert.setAlertType(Alert.AlertType.*INFORMATION*);  
 alert.setHeaderText("Book Added Successfully");  
 alert.show();  
 addBookStage.close();  
 } catch (IOException ex) {  
 alert.setAlertType(Alert.AlertType.*ERROR*);  
 alert.setHeaderText("Error Adding Book");  
 alert.setContentText(ex.getMessage());  
 alert.show();  
 }  
 } else {  
 alert.setAlertType(Alert.AlertType.*WARNING*);  
 alert.setHeaderText("All fields must be filled out");  
 alert.show();  
 }  
 });  
  
 BackgroundImage backgroundImage1 = new BackgroundImage(  
 new Image("Icon 1/123456.png"),  
 BackgroundRepeat.*NO\_REPEAT*,  
 BackgroundRepeat.*NO\_REPEAT*,  
 BackgroundPosition.*CENTER*,  
 new BackgroundSize(BackgroundSize.*AUTO*, BackgroundSize.*AUTO*, false, false, true, true)  
 );  
 layout.setBackground(new Background(backgroundImage1));  
  
 HBox availabilityBox = new HBox(10);  
 availabilityBox.setAlignment(Pos.*CENTER*);  
 availabilityBox.getChildren().addAll(isAvailableLabel, availableButton, notAvailableButton);  
  
 layout.getChildren().addAll(  
 bookLabel, bookField,  
 bookIdLabel, bookIdField,  
 authorNameLabel, authorNameField,  
 availabilityBox,  
 addButton  
 );  
  
 Scene scene = new Scene(layout, 699, 450);  
 addBookStage.setTitle("Add Book");  
 addBookStage.setScene(scene);  
 addBookStage.show();  
 }  
  
  
 private static boolean isBookIdDuplicate(File file, String bookID) {  
 if (!file.exists()) {  
 return false;  
 }  
  
 try (BufferedReader reader = new BufferedReader(new FileReader(file))) {  
 String line;  
 while ((line = reader.readLine()) != null) {  
 String[] parts = line.split(",");  
 if (parts.length > 0 && parts[0].equals(bookID)) {  
 return true;  
 }  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
  
 return false;  
 }  
  
  
  
  
 public static void showBooks() throws FileNotFoundException {  
 Stage showBookStage = new Stage();  
 VBox box = new VBox();  
 box.setAlignment(Pos.*CENTER*);  
 box.setPadding(new Insets(20));  
 box.setSpacing(20);  
  
 BackgroundImage backgroundImage1 = new BackgroundImage(  
 new Image("Icon 1/123456.png"),  
 BackgroundRepeat.*NO\_REPEAT*,  
 BackgroundRepeat.*NO\_REPEAT*,  
 BackgroundPosition.*CENTER*,  
 new BackgroundSize(BackgroundSize.*AUTO*, BackgroundSize.*AUTO*, false, false, true, true)  
 );  
 box.setBackground(new Background(backgroundImage1));  
  
 Label heading = new Label("Book List");  
 heading.setStyle("-fx-font-size: 18px; -fx-font-weight: bold;");  
 box.getChildren().add(heading);  
  
  
 TableView<Book> tableView = new TableView<>();  
  
  
 TableColumn<Book, String> idColumn = new TableColumn<>("ID");  
 idColumn.setCellValueFactory(new PropertyValueFactory<>("id"));  
  
 TableColumn<Book, String> titleColumn = new TableColumn<>("Title");  
 titleColumn.setCellValueFactory(new PropertyValueFactory<>("title"));  
  
 TableColumn<Book, String> authorColumn = new TableColumn<>("Author");  
 authorColumn.setCellValueFactory(new PropertyValueFactory<>("author"));  
  
 TableColumn<Book, String> availabilityColumn = new TableColumn<>("Available");  
 availabilityColumn.setCellValueFactory(new PropertyValueFactory<>("availability"));  
  
  
 tableView.getColumns().addAll(idColumn, titleColumn, authorColumn, availabilityColumn);  
  
  
 ObservableList<Book> bookList = FXCollections.*observableArrayList*();  
  
 try (BufferedReader reader = new BufferedReader(new FileReader("Book.txt"))) {  
 String line;  
 while ((line = reader.readLine()) != null) {  
 String[] bookDetails = line.split(",");  
 if (bookDetails.length == 4) {  
 bookList.add(new Book(bookDetails[0], bookDetails[1], bookDetails[2], bookDetails[3]));  
 }  
 }  
 } catch (IOException e) {  
 Label errorLabel = new Label("Error reading the book file.");  
 errorLabel.setStyle("-fx-text-fill: red; -fx-font-size: 14px;");  
 box.getChildren().add(errorLabel);  
 }  
  
 tableView.setItems(bookList);  
 tableView.setPrefWidth(650);  
  
  
 box.getChildren().add(tableView);  
  
 ScrollPane scrollPane = new ScrollPane(box);  
 scrollPane.setFitToWidth(true);  
  
 Scene scene = new Scene(scrollPane, 699, 450);  
 showBookStage.setTitle("Show Books");  
 showBookStage.setScene(scene);  
 showBookStage.show();  
 }  
  
  
 public static class Book {  
 private String id;  
 private String title;  
 private String author;  
 private String availability;  
  
 public Book(String id, String title, String author, String availability) {  
 this.id = id;  
 this.title = title;  
 this.author = author;  
 this.availability = availability;  
 }  
  
 public String getId() {  
 return id;  
 }  
  
 public String getTitle() {  
 return title;  
 }  
  
 public String getAuthor() {  
 return author;  
 }  
  
 public String getAvailability() {  
 return availability;  
 }  
 }  
  
  
  
  
 public static void borrowBook(String username) {  
 Stage borrowBookStage = new Stage();  
 VBox layout = new VBox(20);  
 layout.setAlignment(Pos.*CENTER*);  
 layout.setPadding(new Insets(20));  
  
 Label instructions = new Label("Enter the Book ID to Borrow:");  
 instructions.setStyle("-fx-font-size: 14px;");  
  
 TextField bookIdField = new TextField();  
 bookIdField.setPromptText("Book ID");  
  
 Button borrowButton = new Button("Borrow Book");  
  
 Alert alert = new Alert(Alert.AlertType.*INFORMATION*);  
  
 borrowButton.setOnAction(e -> {  
 String bookIDToBorrow = bookIdField.getText().trim();  
 if (bookIDToBorrow.isEmpty()) {  
 alert.setHeaderText("Error");  
 alert.setContentText("Book ID cannot be empty.");  
 alert.show();  
 return;  
 }  
  
 File bookFile = new File("Book.txt");  
 File tempFile = new File("TempBook.txt");  
 boolean bookFound = false;  
  
 try (BufferedReader reader = new BufferedReader(new FileReader(bookFile));  
 BufferedWriter writer = new BufferedWriter(new FileWriter(tempFile))) {  
  
 String line;  
 while ((line = reader.readLine()) != null) {  
 String[] bookDetails = line.split(",");  
 if (bookDetails.length == 4) {  
 String bookID = bookDetails[0];  
 String bookTitle = bookDetails[1];  
 String authorName = bookDetails[2];  
 String isAvailable = bookDetails[3];  
  
 if (bookID.equals(bookIDToBorrow) && isAvailable.equalsIgnoreCase("Yes")) {  
 writer.write(bookID + "," + bookTitle + "," + authorName + ",No");  
 writer.newLine();  
 bookFound = true;  
 } else {  
 writer.write(line);  
 writer.newLine();  
 }  
 }  
 }  
 } catch (IOException ex) {  
 alert.setHeaderText("Error");  
 alert.setContentText("An error occurred while borrowing the book.");  
 alert.show();  
 return;  
 }  
  
 if (bookFound) {  
 if (bookFile.delete() && tempFile.renameTo(bookFile)) {  
 alert.setHeaderText("Success");  
 alert.setContentText("Book borrowed successfully.");  
 alert.show();  
 borrowBookStage.close();  
 } else {  
 alert.setHeaderText("Error");  
 alert.setContentText("Could not update the book file.");  
 alert.show();  
 }  
 } else {  
 tempFile.delete();  
 alert.setHeaderText("Error");  
 alert.setContentText("Book not found or already borrowed.");  
 alert.show();  
 }  
 });  
  
 layout.getChildren().addAll(instructions, bookIdField, borrowButton);  
  
 Scene scene = new Scene(layout, 400, 200);  
 borrowBookStage.setTitle("Borrow Book");  
 borrowBookStage.setScene(scene);  
 borrowBookStage.show();  
 }  
  
 public static void returnBook(String username) {  
 Stage returnBookStage = new Stage();  
 VBox layout = new VBox(20);  
 layout.setAlignment(Pos.*CENTER*);  
 layout.setPadding(new Insets(20));  
  
 Label instructions = new Label("Enter the Book ID to Return:");  
 instructions.setStyle("-fx-font-size: 14px;");  
  
 TextField bookIdField = new TextField();  
 bookIdField.setPromptText("Book ID");  
  
 Button returnButton = new Button("Return Book");  
  
 Alert alert = new Alert(Alert.AlertType.*INFORMATION*);  
  
 returnButton.setOnAction(e -> {  
 String bookIDToReturn = bookIdField.getText().trim();  
 if (bookIDToReturn.isEmpty()) {  
 alert.setHeaderText("Error");  
 alert.setContentText("Book ID cannot be empty.");  
 alert.show();  
 return;  
 }  
  
 File bookFile = new File("Book.txt");  
 File tempFile = new File("TempBook.txt");  
  
 if (!bookFile.exists()) {  
 alert.setHeaderText("Error");  
 alert.setContentText("Book file not found.");  
 alert.show();  
 return;  
 }  
  
 boolean bookFound = false;  
 boolean bookAlreadyAvailable = false;  
  
 try (BufferedReader reader = new BufferedReader(new FileReader(bookFile));  
 BufferedWriter writer = new BufferedWriter(new FileWriter(tempFile))) {  
  
 String line;  
 while ((line = reader.readLine()) != null) {  
 String[] bookDetails = line.split(",");  
 if (bookDetails.length == 4) {  
 String bookID = bookDetails[0];  
 String bookTitle = bookDetails[1];  
 String authorName = bookDetails[2];  
 String isAvailable = bookDetails[3];  
  
 if (bookID.equals(bookIDToReturn)) {  
 bookFound = true;  
 if (isAvailable.equalsIgnoreCase("No")) {  
 // Mark the book as available  
 writer.write(bookID + "," + bookTitle + "," + authorName + ",Yes");  
 writer.newLine();  
 } else {  
 bookAlreadyAvailable = true;  
 writer.write(line);  
 writer.newLine();  
 }  
 } else {  
 writer.write(line);  
 writer.newLine();  
 }  
 }  
 }  
 } catch (IOException ex) {  
 alert.setHeaderText("Error");  
 alert.setContentText("An error occurred while processing the book file.");  
 alert.show();  
 tempFile.delete();  
 return;  
 }  
  
 if (bookFound) {  
 if (bookAlreadyAvailable) {  
 tempFile.delete();  
 alert.setHeaderText("Error");  
 alert.setContentText("Book is already available and cannot be returned.");  
 alert.show();  
 } else {  
 if (bookFile.delete() && tempFile.renameTo(bookFile)) {  
 alert.setHeaderText("Success");  
 alert.setContentText("Book returned successfully.");  
 alert.show();  
 returnBookStage.close();  
 } else {  
 alert.setHeaderText("Error");  
 alert.setContentText("Could not update the book file.");  
 alert.show();  
 }  
 }  
 } else {  
 tempFile.delete();  
 alert.setHeaderText("Error");  
 alert.setContentText("Book not found in the system.");  
 alert.show();  
 }  
 });  
  
 layout.getChildren().addAll(instructions, bookIdField, returnButton);  
  
 Scene scene = new Scene(layout, 400, 200);  
 returnBookStage.setTitle("Return Book");  
 returnBookStage.setScene(scene);  
 returnBookStage.show();  
 }  
  
  
  
 public static void searchBook() {  
 Stage searchBookStage = new Stage();  
 VBox layout = new VBox(20);  
 layout.setPadding(new Insets(20));  
 layout.setAlignment(Pos.*CENTER*);  
  
 BackgroundImage backgroundImage1 = new BackgroundImage(  
 new Image("Icon 1/123456.png"),  
 BackgroundRepeat.*NO\_REPEAT*,  
 BackgroundRepeat.*NO\_REPEAT*,  
 BackgroundPosition.*CENTER*,  
 new BackgroundSize(BackgroundSize.*AUTO*, BackgroundSize.*AUTO*, false, false, true, true)  
 );  
 layout.setBackground(new Background(backgroundImage1));  
  
 Label instructions = new Label("Search for a Book by ID :");  
 instructions.setStyle("-fx-font-size: 14px;");  
  
 TextField searchField = new TextField();  
 searchField.setPromptText("Enter Title, ID, or Author");  
  
 Button searchButton = new Button("Search");  
  
 VBox resultBox = new VBox(10);  
 resultBox.setAlignment(Pos.*CENTER*);  
  
 ScrollPane resultScrollPane = new ScrollPane(resultBox);  
 resultScrollPane.setFitToWidth(true);  
  
 searchButton.setOnAction(e -> {  
 String searchTerm = searchField.getText().trim().toLowerCase();  
 resultBox.getChildren().clear();  
  
 if (searchTerm.isEmpty()) {  
 Label errorLabel = new Label("Search field cannot be empty.");  
 errorLabel.setStyle("-fx-text-fill: red; -fx-font-size: 14px;");  
 resultBox.getChildren().add(errorLabel);  
 return;  
 }  
  
 File bookFile = new File("Book.txt");  
  
 if (!bookFile.exists()) {  
 Label errorLabel = new Label("Book file not found.");  
 errorLabel.setStyle("-fx-text-fill: red; -fx-font-size: 14px;");  
 resultBox.getChildren().add(errorLabel);  
 return;  
 }  
  
 boolean matchFound = false;  
  
 try (BufferedReader reader = new BufferedReader(new FileReader(bookFile))) {  
 String line;  
 while ((line = reader.readLine()) != null) {  
 String[] bookDetails = line.split(",");  
 if (bookDetails.length == 4) {  
 String bookID = bookDetails[0].toLowerCase();  
 String bookTitle = bookDetails[1].toLowerCase();  
 String authorName = bookDetails[2].toLowerCase();  
 String isAvailable = bookDetails[3];  
  
 if (bookID.equals(searchTerm) ) {  
 matchFound = true;  
 Label bookInfo = new Label(  
 "ID: " + bookDetails[0] + " | Title: " + bookDetails[1] +  
 " | Author: " + bookDetails[2] + " | Available: " + isAvailable  
 );  
 bookInfo.setStyle("-fx-font-size: 14px;");  
 resultBox.getChildren().add(bookInfo);  
 }  
 }  
 }  
 } catch (IOException ex) {  
 Label errorLabel = new Label("An error occurred while reading the book file.");  
 errorLabel.setStyle("-fx-text-fill: red; -fx-font-size: 14px;");  
 resultBox.getChildren().add(errorLabel);  
 return;  
 }  
  
 if (!matchFound) {  
 Label noResultsLabel = new Label("No books found matching the search term.");  
 noResultsLabel.setStyle("-fx-font-size: 14px; -fx-text-fill: gray;");  
 resultBox.getChildren().add(noResultsLabel);  
 }  
 });  
  
 layout.getChildren().addAll(instructions, searchField, searchButton, resultScrollPane);  
  
 Scene scene = new Scene(layout, 699, 450);  
 searchBookStage.setTitle("Search Book");  
 searchBookStage.setScene(scene);  
 searchBookStage.show();  
 }  
  
  
  
}**