QuizApp: An Interactive GUI-Based Quiz Application

IA-3

Subject: Object-Oriented Programming (DSE 2123)

<u>Mini Project</u>

Student Name	Prajjnaa Ray Choudhury
Reg No	**230968348**
Assignment No.	IA-3
Subject Code	DSE 2123
Subject	OOP with Java
Marks	10M

Contents

Problem Statement 5: Develop a GUI-Based Quiz Application

Requirements:

1. Quiz Setup: Create a GUI for administrators to add, edit, and delete quiz questions with multiple choice answers.

2. Quiz Interface: Users should be able to take quizzes, selecting answers from multiple-choice questions.

3. Timer Functionality: Implement a timer for each quiz, with the option to start and stop the quiz automatically based on the timer.

4. Result Display: At the end of the quiz, display the user's score and provide feedback (correct/incorrect answers).

The goal of QuizApp is to create an engaging, user-friendly GUI based quiz application that allows users to answer multiple-choice questions. The primary aim of this application is to make learning more engaging and accessible by incorporating real-time feedback, score tracking, time management and a user-friendly interface.

Methodology

1. Requirements Gathering

- **Objective:** To develop a GUI-based quiz application QuizApp, that caters to two types of users: **Admins** and **Users**. The requirements were gathered through brainstorming sessions focused on providing an interactive quiz experience with features like question management, score tracking, time management, and feedback.
- **Admin Requirements:** Admins can add, edit, and delete questions.
- **User Requirements:** Users experience a simple interface to answer questions, receive instant feedback, option to pause the timer and view their scores at the end of each quiz.
- **Non-functional Requirements:** Ensures user-friendliness, responsiveness, and fast performance for an optimal quiz experience.

2. Design Phase

- **System Architecture:** A modular, object-oriented structure was planned to separate different aspect of the application, like question management and other UI elements.
- **Class Structure**
  - **QuizApp Class:** Manages the overall quiz flow, with a login panel, an admin panel where the admin can manage the questions and a quiz panel where the user attempts the timed multiple-choice questions.
  - **Question Class:** Represents each question with its text, options and the correct answer, making it easier to add or modify questions.
- **File Handling with `users.txt` and `questions.txt`**
  - **users.txt:** Stores user login information, such as usernames and passwords.
  - **questions.txt:** Contains quiz questions, options and correct answers. By using this file, admins can update the question bank externally without modifying the core code, and the app can dynamically loas questions each time a quiz is launched.
- **Swing Libraries for User Interface(UI)**
  - **JFrame:** Used as the main window for the app, serving as the container for all UI elements, including the quiz display, question navigation, and score summary.
  - **JLabel:** Displays static text on the screen, such as instructions, question prompts, and feedback messages.
  - **JButton:** Provides clickable buttons for user actions

    - **JTextField and JPasswordField:** Used for user input fields in the login and registration screens, with JPasswordField providing hidden text for secure password entry.
    - **JOptionPane:** Used for displaying pop-up dialogs to show alerts, confirmations, and score summaries.
    - **JRadioButton:** Used for multiple-choice answer options, allowing users to select a single answer per question.
    - **JPanel:** Serves as a container to organize other UI components.
    - **JList and JScrollPane:** Together, they allow for a scrollable list view, used in admin sections for managing or selecting questions in a list format.
- **User Interface (UI) Design:** A straightforward, user-friendly interface is created to guide users seamlessly through the quiz, displaying questions, feedback and scores clearly.

3. Development Phase

- **Programming Language:** Java, used due to its flexibility. Object-oriented structure, and cross-platform compatibility.
- **Core Functionality**
    - **Quiz Logic:** Implemented in Java using Swing components for question navigation, answer selection, and score tracking.
    - **Question Management and File Handling:** Used questions.txt to load questions at runtime and update data without altering the code.
    - **User Interface:** Swing components create an engaging and organised quiz interface.
    - **Score Calculation:** Score updates occur in real time as each question is answered, giving users instant feedback on their progress.

4. Implementation of Key Features

- **Question Management:** Created an admin interface to manage questions in questions.txt. Admins can edit the file to update questions without modifying the code.
- **Score Tracking and Feedback:** Final score summaries with feedbacks are implemented.
- **User Authentication:** User login and registration, interacting with users.txt for secure access.
- **Timed Questions:** A timer is added to each question for a challenge, which can be paused anytime by the user.

5. Testing and Debugging

- **Unit Testing**: Tested each component and feature individually, including file handling for users.txt and questions.txt.

- **UI Testing**: Ensured all Swing components worked as intended, testing for ease of use, accessibility, and smooth interaction.
- **Performance Testing**: Assessed response times and file handling to maintain an efficient, fast experience.

6. Git link

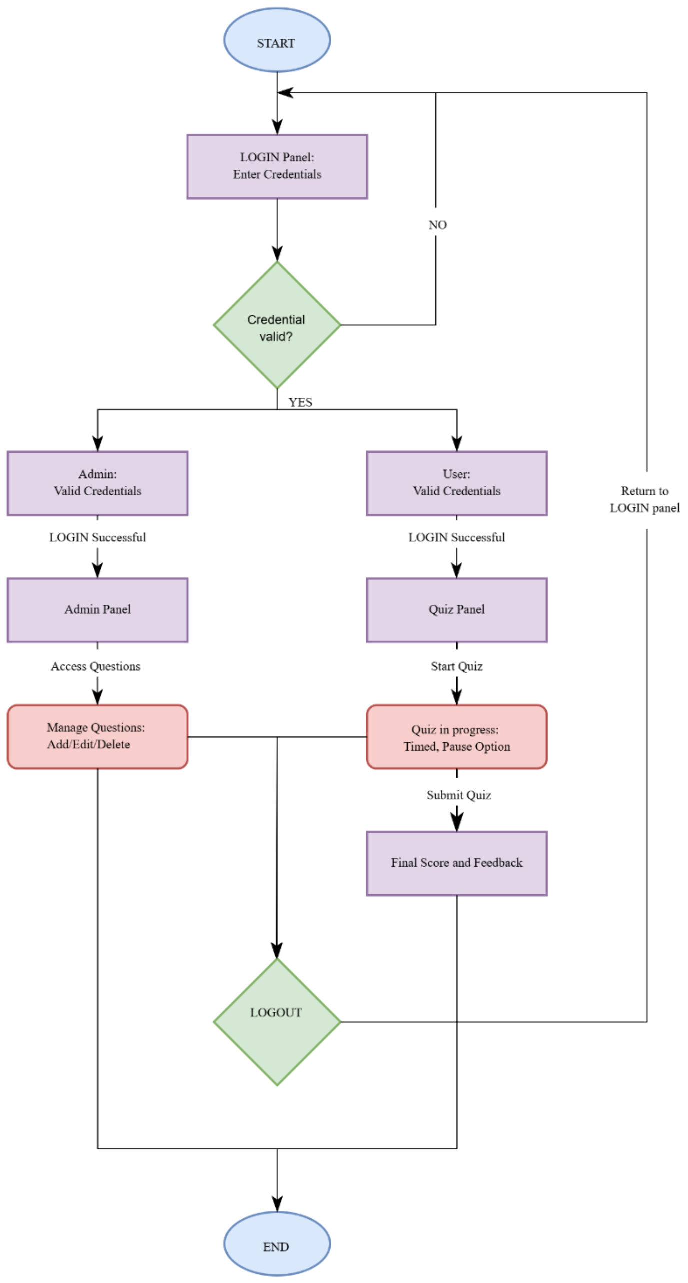
- https://github.com/ankshirc/GUI-Based-Quiz-Application.git

Results and Screenshot

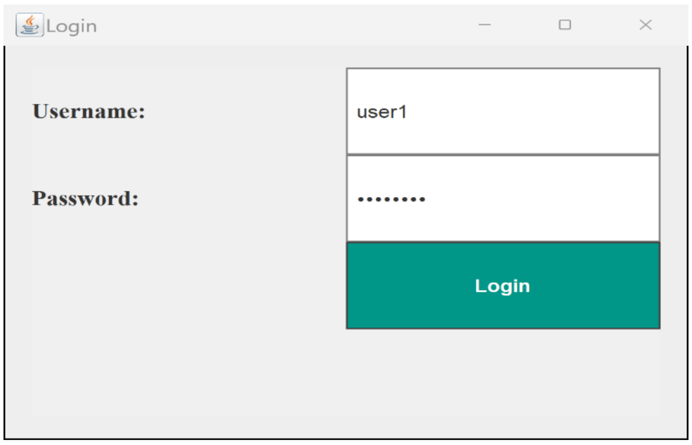
Events and Actions

1. Login panel appears. Enter credentials.
2. Entered as user.
    a. If valid credentials, then login successful. Quiz panel appears. Timed quiz with multiple choice questions. Timer can be paused as well as per user's desire. Display final score and feedback after quiz. Program terminates.
    b. If invalid credentials, then login unsuccessful. Login panel reappears to try again.
    c. Once entered as user, panel also contains a logout button to logout of quiz and login panel reappears.
3. Entered as admin.
    a. If valid credentials, then login successful. Admin panel appears. Admin can add, edit or delete questions, as per their desire.
    b. If invalid credentials, then login unsuccessful. Login panel reappears to try again.
    c. Once entered as admin, panel also contains a logout button to logout of admin panel and login panel reappears.

Flowchart

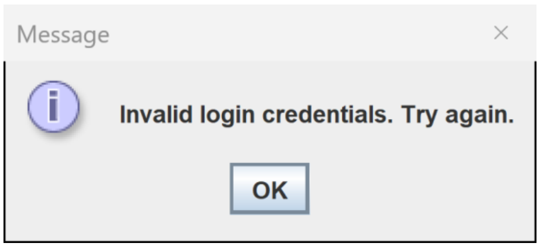

START
LOGIN Panel:
Enter Credentials
Credential valid?
NO
YES
Return to LOGIN panel
Admin:
Valid Credentials
User:
Valid Credentials
LOGIN Successful
LOGIN Successful
Admin Panel
Quiz Panel
Access Questions
Start Quiz
Manage Questions:
Add/Edit/Delete
Quiz in progress:
Timed, Pause Option
Submit Quiz
Final Score and Feedback
LOGOUT
END

Logging in as user

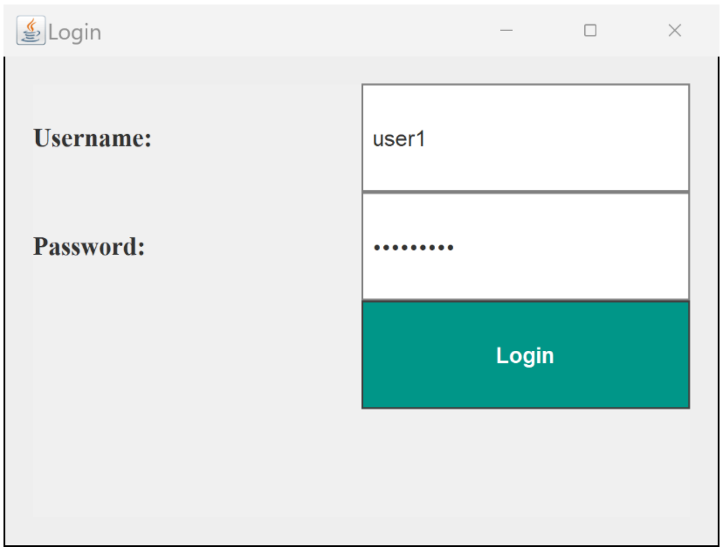

Login

Username: user1

Password: ••••••••

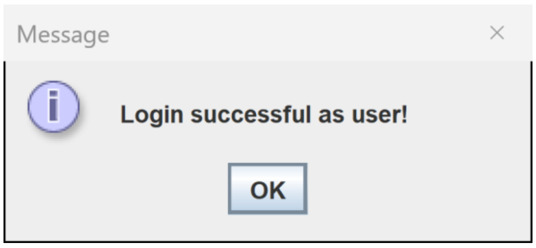

Message

Invalid login credentials. Try again.

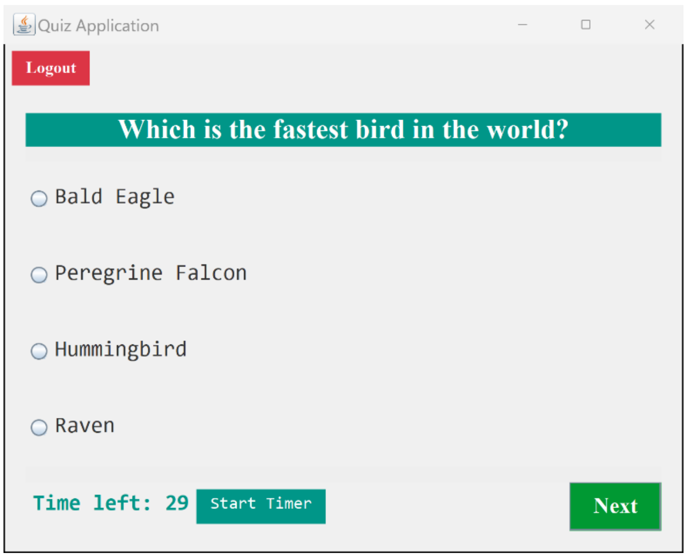

Login

Username: user1

Password: •••••••••


Message

Login successful as user!

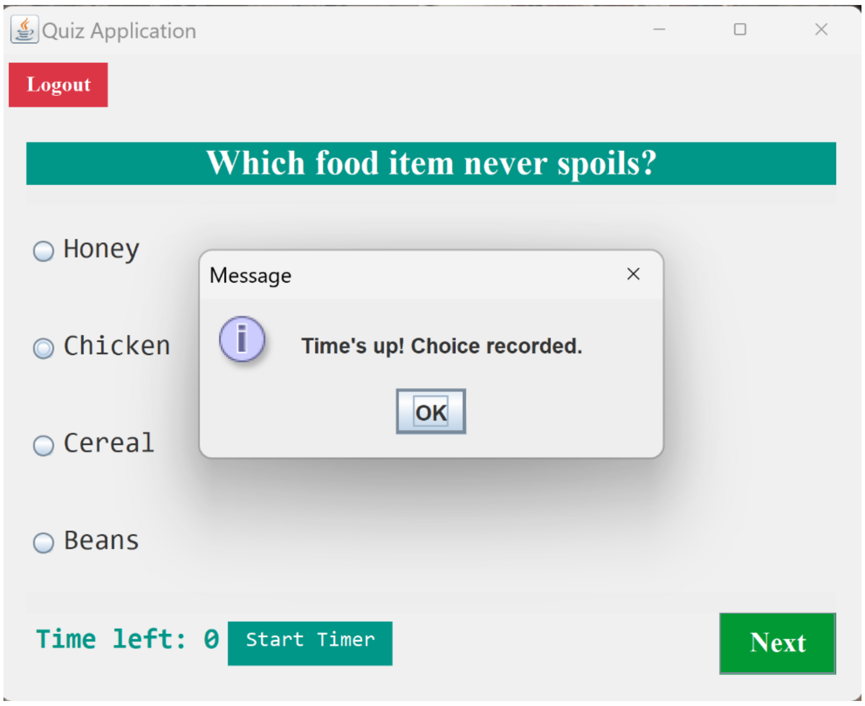

Quiz Application

Logout

Which is the fastest bird in the world?

- ○ Bald Eagle
- ○ Peregrine Falcon
- ○ Hummingbird
- ○ Raven

Time left: 29 Start Timer Next


Quiz Application

Logout

Which food item never spoils?

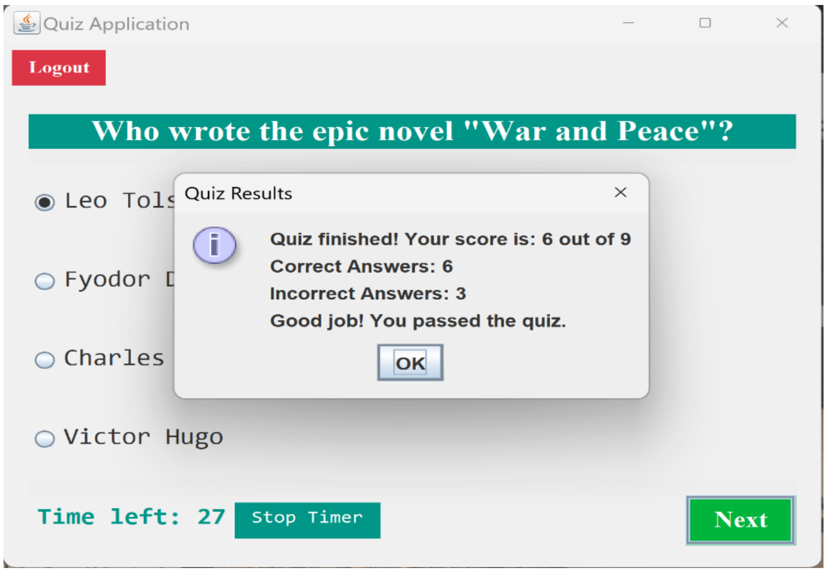
- ○ Honey
- ○ Chicken
- ○ Cereal
- ○ Beans

Message

Time's up! Choice recorded.

OK

Time left: 0 Start Timer Next


Quiz Application

Logout

Who wrote the epic novel "War and Peace"?

- ● Leo Tols
- ○ Fyodor D
- ○ Charles
- ○ Victor Hugo

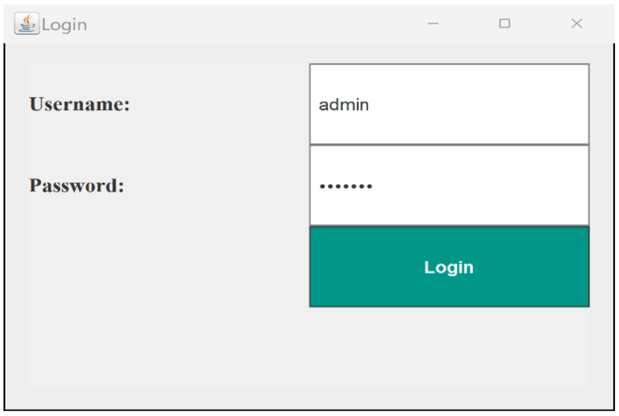
Quiz Results

Quiz finished! Your score is: 6 out of 9
Correct Answers: 6
Incorrect Answers: 3
Good job! You passed the quiz.

OK

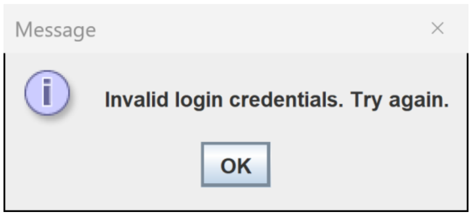
Time left: 27 Stop Timer Next

Logging in as admin

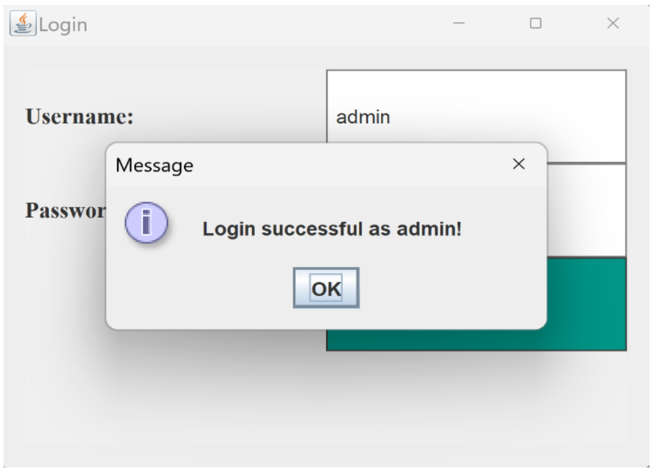

Login

Username: admin

Password: •••••••


Message

Invalid login credentials. Try again.

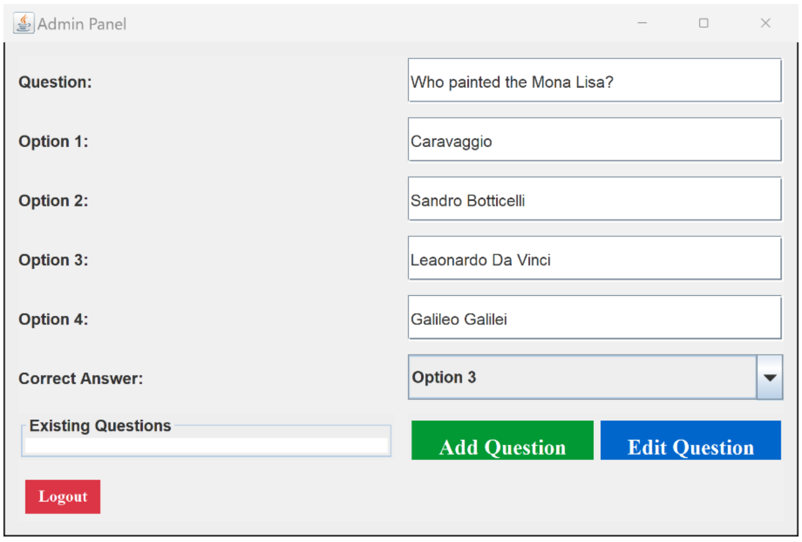

Login

Username: admin

Password:

Message

Login successful as admin!


Admin Panel

Question: Who painted the Mona Lisa?

Option 1: Caravaggio

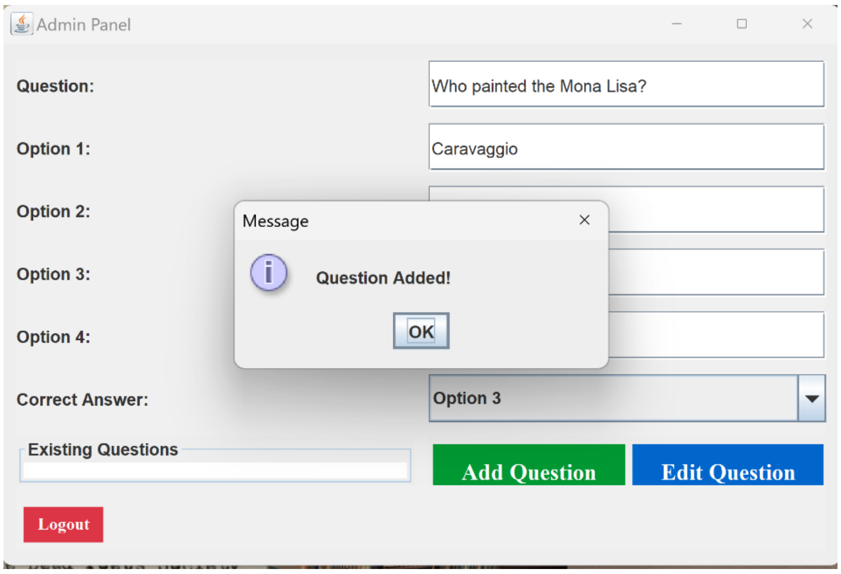
Option 2: Sandro Botticelli

Option 3: Leaonardo Da Vinci

Option 4: Galileo Galilei

Correct Answer: Option 3

Existing Questions



Question: Who painted the Mona Lisa?

Option 1: Caravaggio

Option 2:

Option 3:

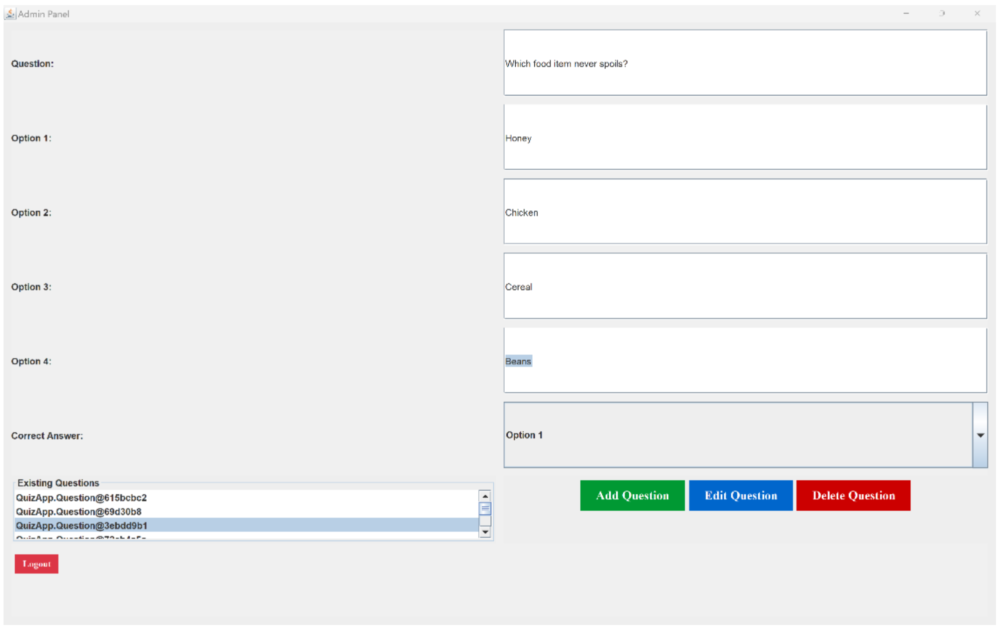
Option 4:

Correct Answer: Option 3

Message

Question Added!

OK



Question: Which food item never spoils?

Option 1: Honey

Option 2: Chicken

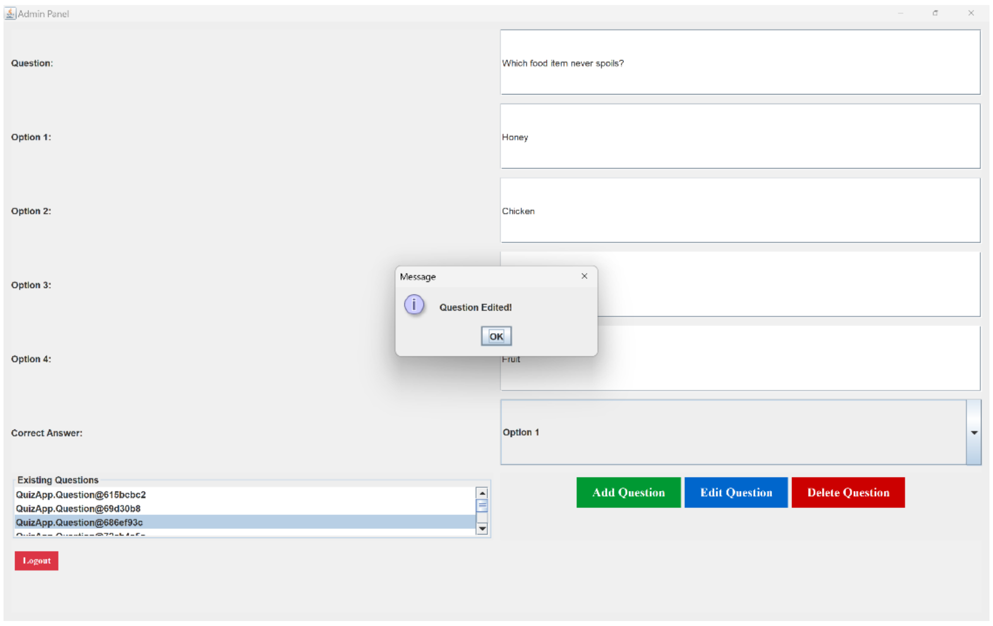
Option 3: Cereal

Option 4: Beans

Correct Answer: Option 1

Existing Questions

QuizApp.Question@615bcbc2
QuizApp.Question@69d30b8
QuizApp.Question@3ebdd9b1



Question: Which food item never spoils?

Option 1: Honey

Option 2: Chicken

Option 3:

Option 4: Fruit

Correct Answer: Option 1

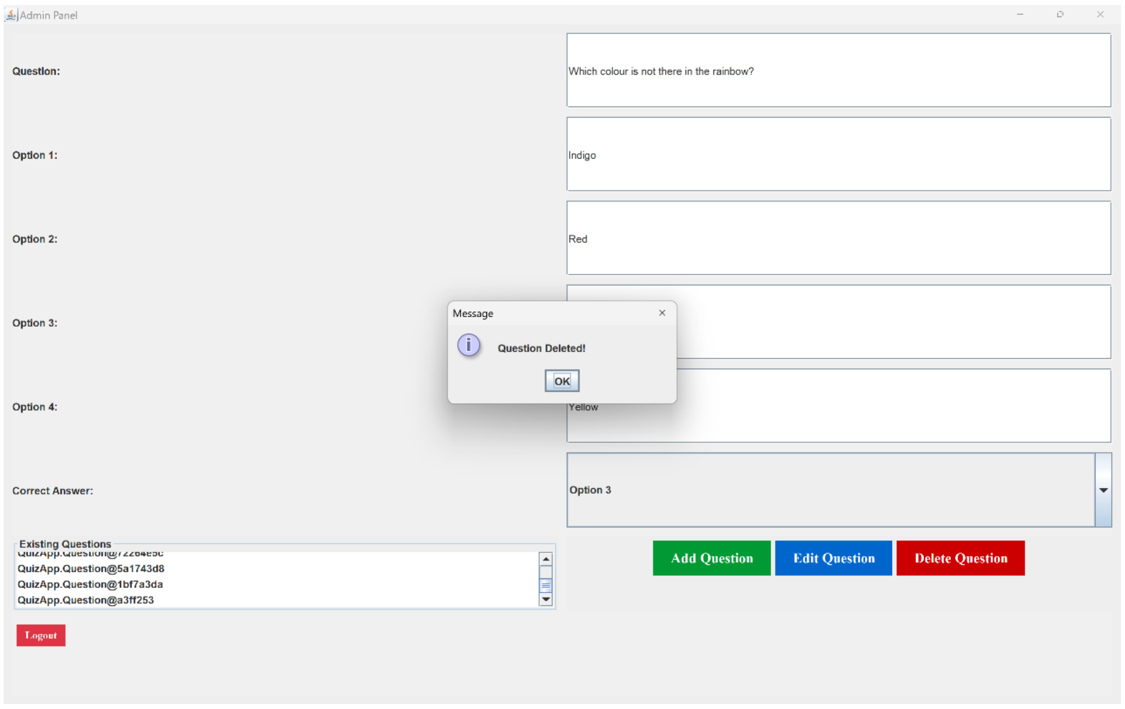
Message

Question Edited!

OK

Existing Questions

QuizApp.Question@615bcbc2
QuizApp.Question@69d30b8
QuizApp.Question@686ef93c


Admin Panel

Question:	Which colour is not there in the rainbow?
Option 1:	Indigo
Option 2:	Red
Option 3:	
Option 4:	Yellow
Correct Answer:	Option 3

Message

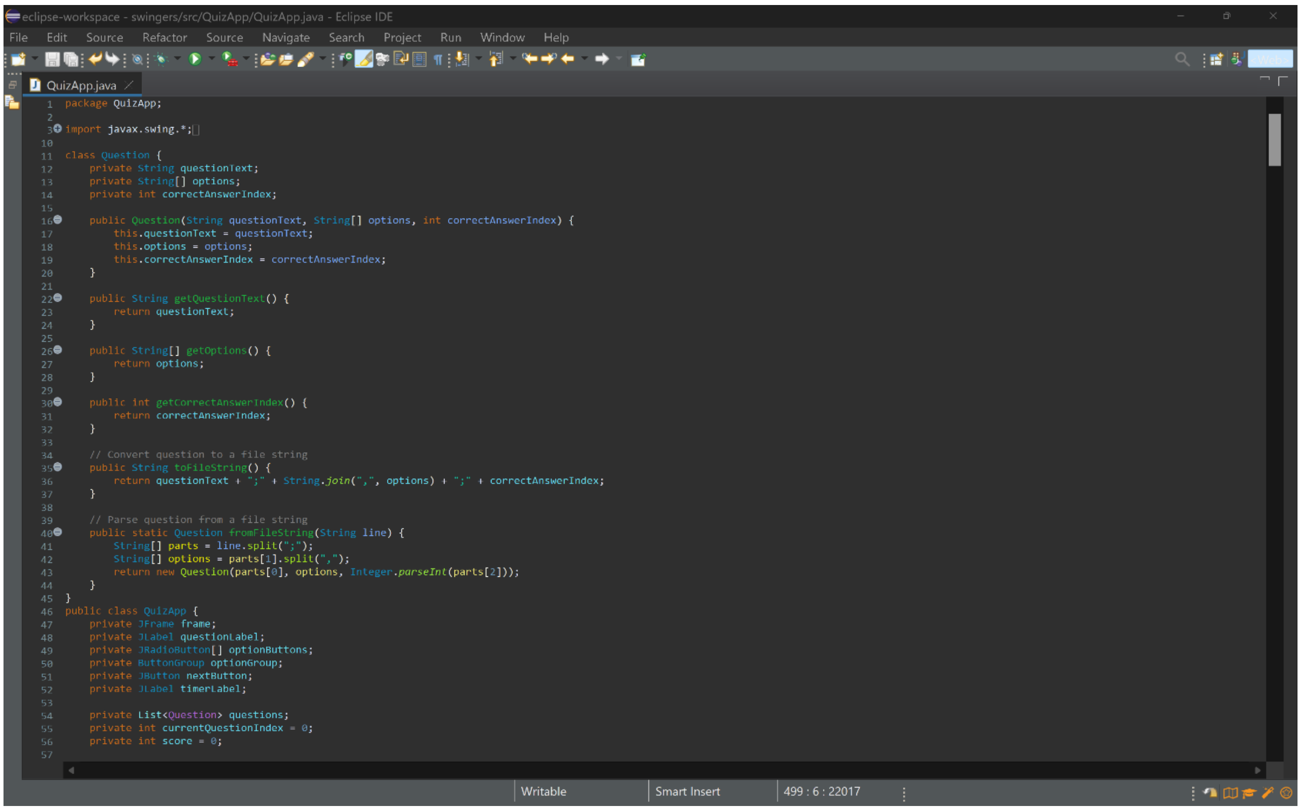
Question Deleted!

Existing Questions

```
QuizApp.Question@72264e5c
QuizApp.Question@5a1743d8
QuizApp.Question@1bf7a3da
QuizApp.Question@a3ff253
```

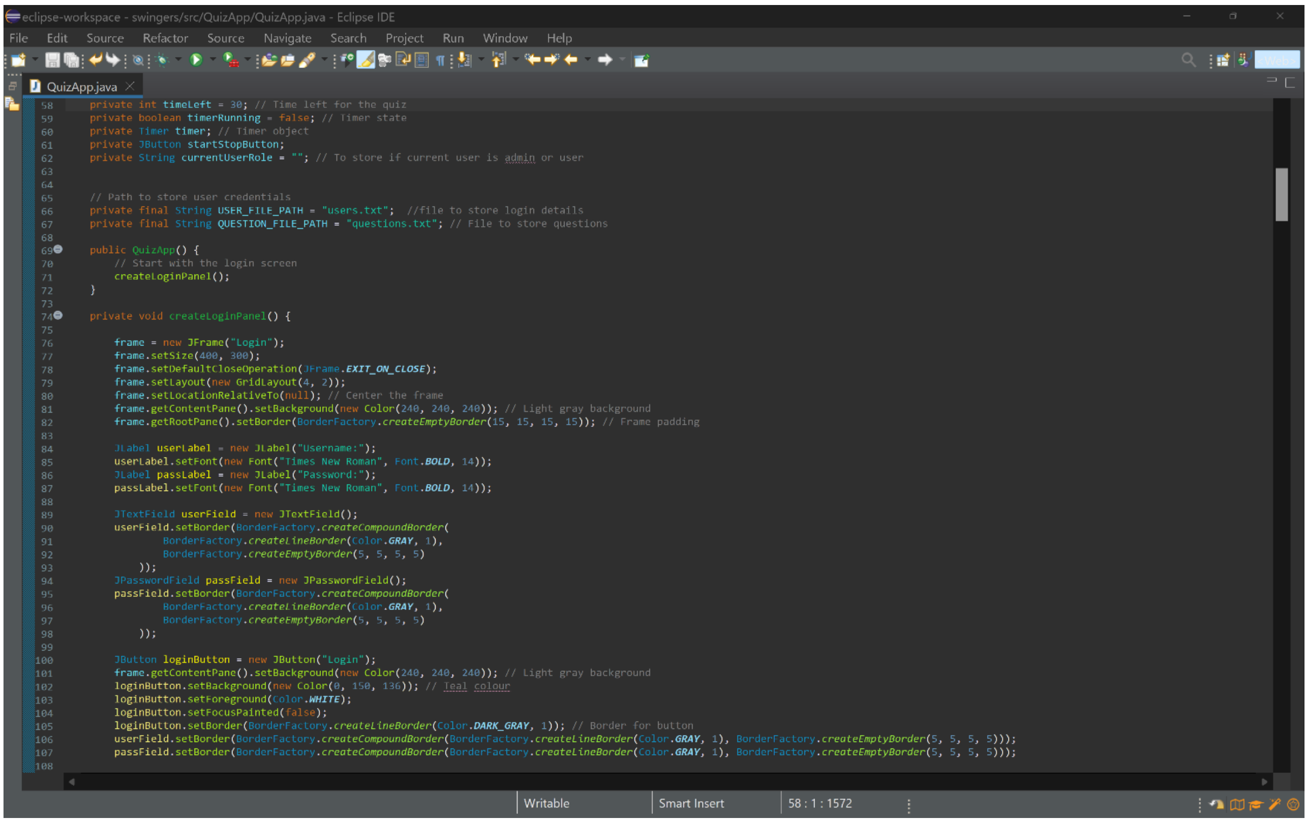
Logout button works in Quiz panel as well as the Admin panel.

Program code



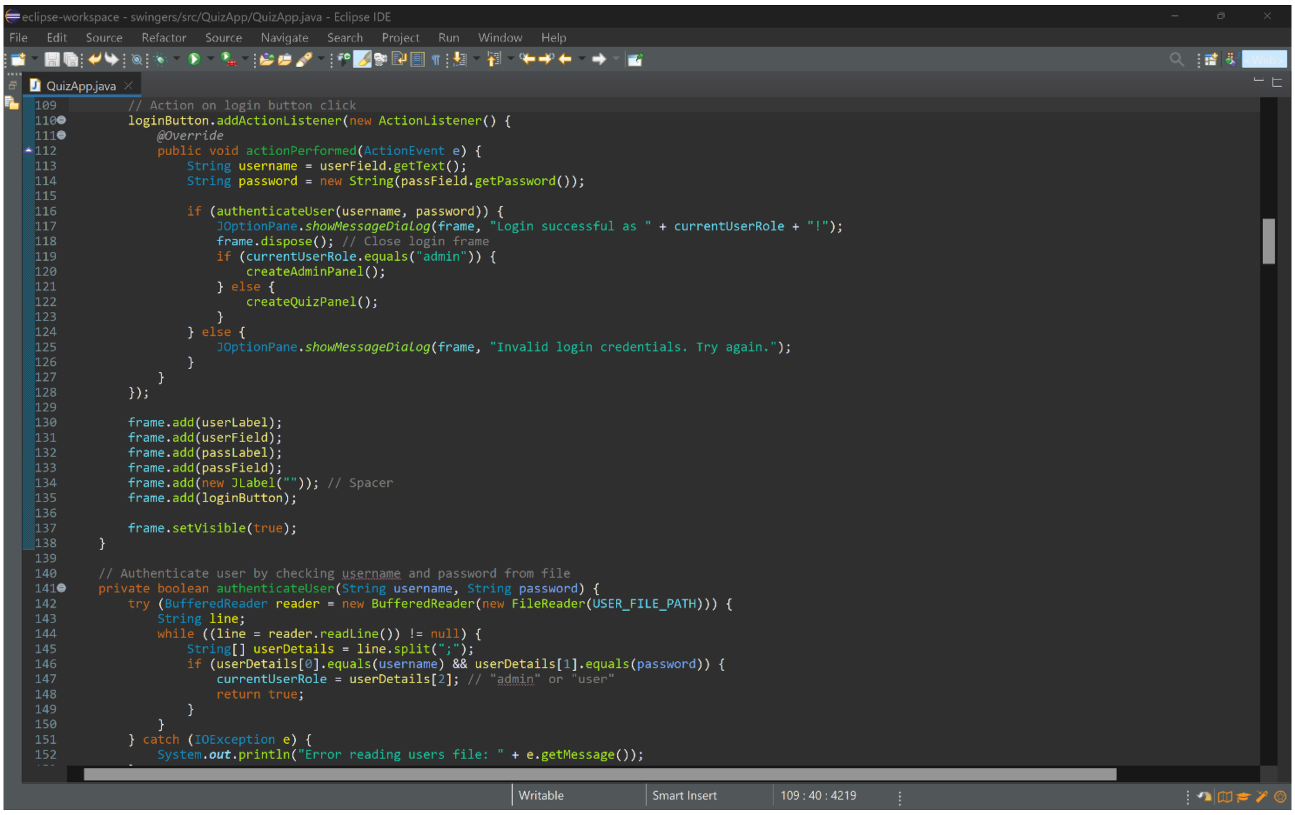
eclipse-workspace - swingers/src/QuizApp/QuizApp.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
QuizApp.java
Writable
Smart Insert
499 : 6 : 22017

```java
package QuizApp;

import javax.swing.*;

class Question {
    private String questionText;
    private String[] options;
    private int correctAnswerIndex;

    public Question(String questionText, String[] options, int correctAnswerIndex) {
        this.questionText = questionText;
        this.options = options;
        this.correctAnswerIndex = correctAnswerIndex;
    }

    public String getQuestionText() {
        return questionText;
    }

    public String[] getOptions() {
        return options;
    }

    public int getCorrectAnswerIndex() {
        return correctAnswerIndex;
    }

    // Convert question to a file string
    public String toFileString() {
        return questionText + ";" + String.join(",", options) + ";" + correctAnswerIndex;
    }

    // Parse question from a file string
    public static Question fromFileString(String line) {
        String[] parts = line.split(";");
        String[] options = parts[1].split(",");
        return new Question(parts[0], options, Integer.parseInt(parts[2]));
    }
}
public class QuizApp {
    private JFrame frame;
    private JLabel questionLabel;
    private JRadioButton[] optionButtons;
    private ButtonGroup optionGroup;
    private JButton nextButton;
    private JLabel timerLabel;

    private List<Question> questions;
    private int currentQuestionIndex = 0;
    private int score = 0;

```

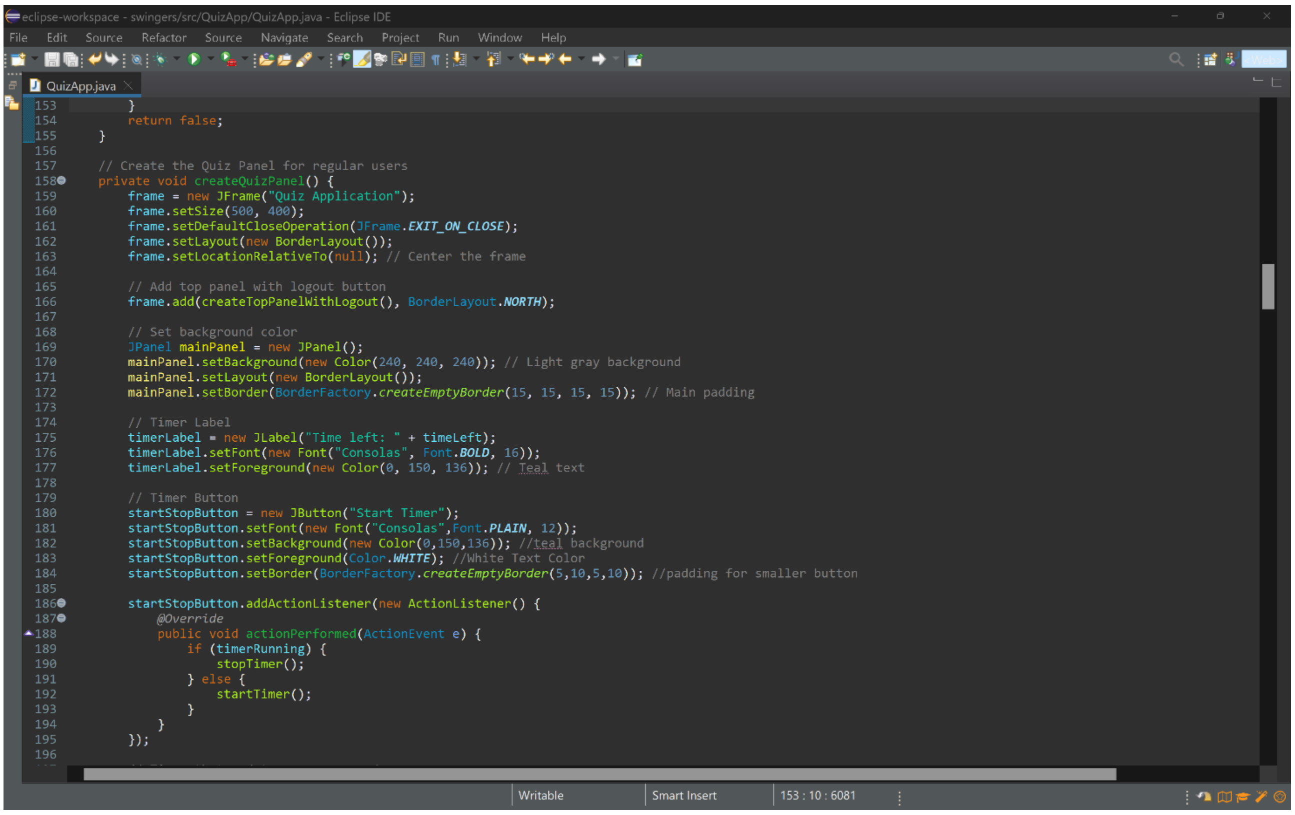


eclipse-workspace - swingers/src/QuizApp/QuizApp.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
QuizApp.java
Writable
Smart Insert
58 : 1 : 1572

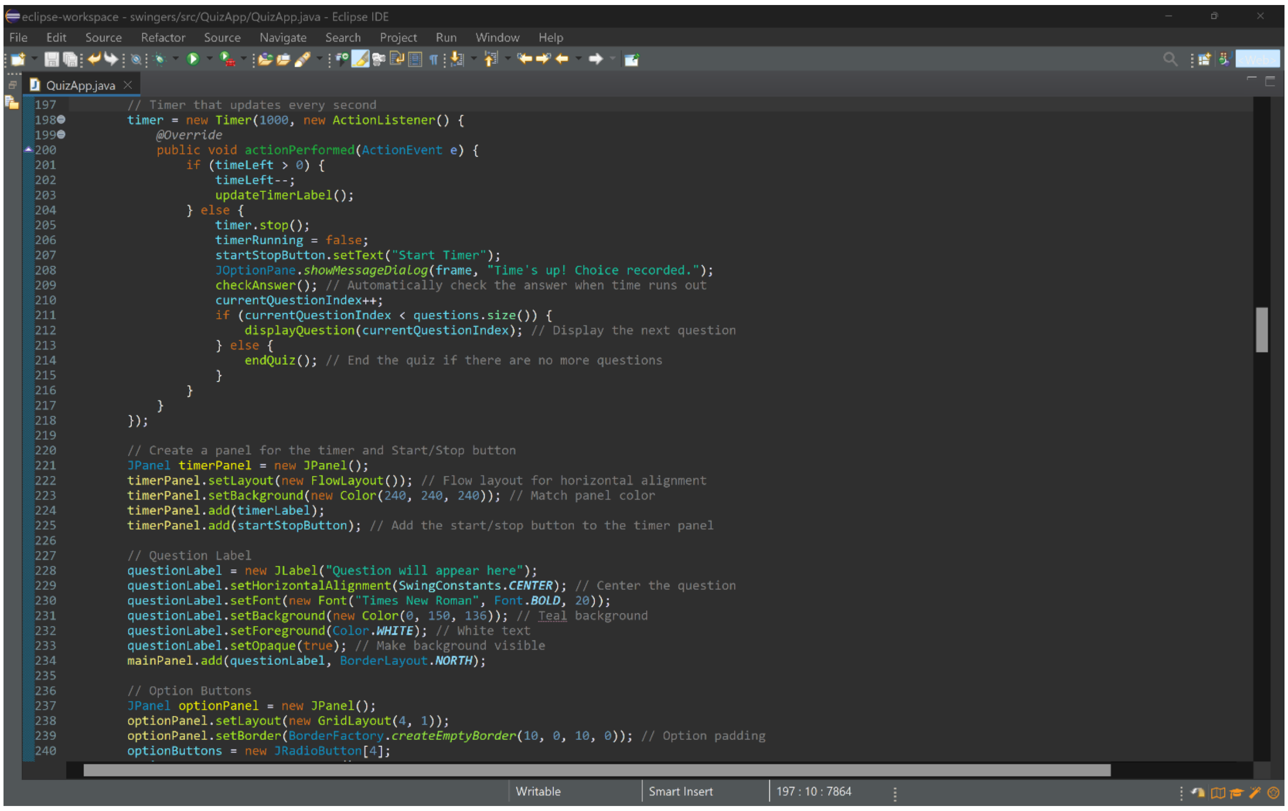
```java
    private int timeLeft = 30; // Time left for the quiz
    private boolean timerRunning = false; // Timer state
    private Timer timer; // Timer object
    private JButton startStopButton;
    private String currentUserRole = ""; // To store if current user is admin or user


    // Path to store user credentials
    private final String USER_FILE_PATH = "users.txt";  //file to store login details
    private final String QUESTION_FILE_PATH = "questions.txt"; // File to store questions

    public QuizApp() {
        // Start with the login screen
        createLoginPanel();
    }

    private void createLoginPanel() {

        frame = new JFrame("Login");
        frame.setSize(400, 300);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLayout(new GridLayout(4, 2));
        frame.setLocationRelativeTo(null); // Center the frame
        frame.getContentPane().setBackground(new Color(240, 240, 240)); // Light gray background
        frame.getRootPane().setBorder(BorderFactory.createEmptyBorder(15, 15, 15, 15)); // Frame padding

        JLabel userLabel = new JLabel("Username:");
        userLabel.setFont(new Font("Times New Roman", Font.BOLD, 14));
        JLabel passLabel = new JLabel("Password:");
        passLabel.setFont(new Font("Times New Roman", Font.BOLD, 14));

        JTextField userField = new JTextField();
        userField.setBorder(BorderFactory.createCompoundBorder(
                BorderFactory.createLineBorder(Color.GRAY, 1),
                BorderFactory.createEmptyBorder(5, 5, 5, 5)
            ));
        JPasswordField passField = new JPasswordField();
        passField.setBorder(BorderFactory.createCompoundBorder(
                BorderFactory.createLineBorder(Color.GRAY, 1),
                BorderFactory.createEmptyBorder(5, 5, 5, 5)
            ));

        JButton loginButton = new JButton("Login");
        frame.getContentPane().setBackground(new Color(240, 240, 240)); // Light gray background
        loginButton.setBackground(new Color(0, 150, 136)); // Teal colour
        loginButton.setForeground(Color.WHITE);
        loginButton.setFocusPainted(false);
        loginButton.setBorder(BorderFactory.createLineBorder(Color.DARK_GRAY, 1)); // Border for button
        userField.setBorder(BorderFactory.createCompoundBorder(BorderFactory.createLineBorder(Color.GRAY, 1), BorderFactory.createEmptyBorder(5, 5, 5, 5)));
        passField.setBorder(BorderFactory.createCompoundBorder(BorderFactory.createLineBorder(Color.GRAY, 1), BorderFactory.createEmptyBorder(5, 5, 5, 5)));

```


eclipse-workspace - swingers/src/QuizApp/QuizApp.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
QuizApp.java
Writable
Smart Insert
109 : 40 : 4219

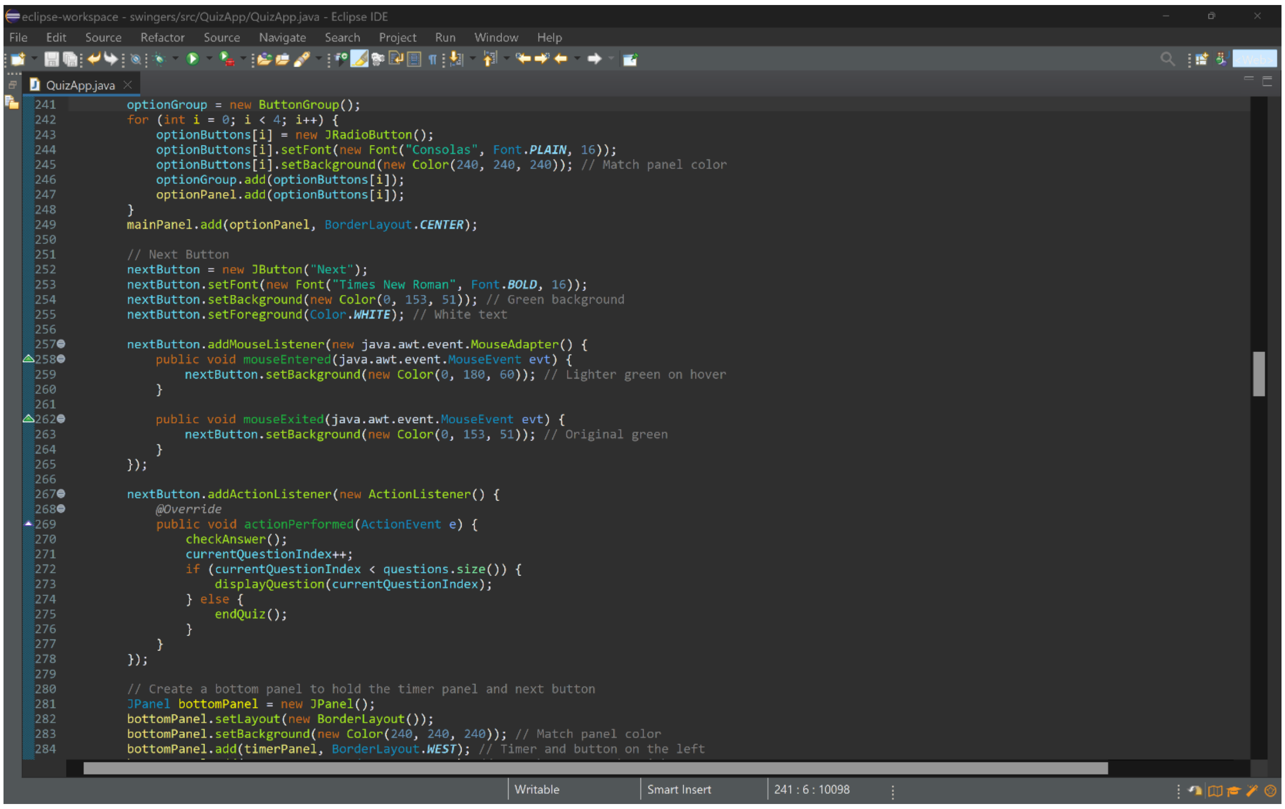
```java
        // Action on login button click
        loginButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                String username = userField.getText();
                String password = new String(passField.getPassword());

                if (authenticateUser(username, password)) {
                    JOptionPane.showMessageDialog(frame, "Login successful as " + currentUserRole + "!");
                    frame.dispose(); // Close login frame
                    if (currentUserRole.equals("admin")) {
                        createAdminPanel();
                    } else {
                        createQuizPanel();
                    }
                } else {
                    JOptionPane.showMessageDialog(frame, "Invalid login credentials. Try again.");
                }
            }
        });

        frame.add(userLabel);
        frame.add(userField);
        frame.add(passLabel);
        frame.add(passField);
        frame.add(new JLabel("")); // Spacer
        frame.add(loginButton);

        frame.setVisible(true);
    }

    // Authenticate user by checking username and password from file
    private boolean authenticateUser(String username, String password) {
        try (BufferedReader reader = new BufferedReader(new FileReader(USER_FILE_PATH))) {
            String line;
            while ((line = reader.readLine()) != null) {
                String[] userDetails = line.split(";");
                if (userDetails[0].equals(username) && userDetails[1].equals(password)) {
                    currentUserRole = userDetails[2]; // "admin" or "user"
                    return true;
                }
            }
        } catch (IOException e) {
            System.out.println("Error reading users file: " + e.getMessage());
```


eclipse-workspace - swingers/src/QuizApp/QuizApp.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
QuizApp.java
Writable
Smart Insert
153 : 10 : 6081

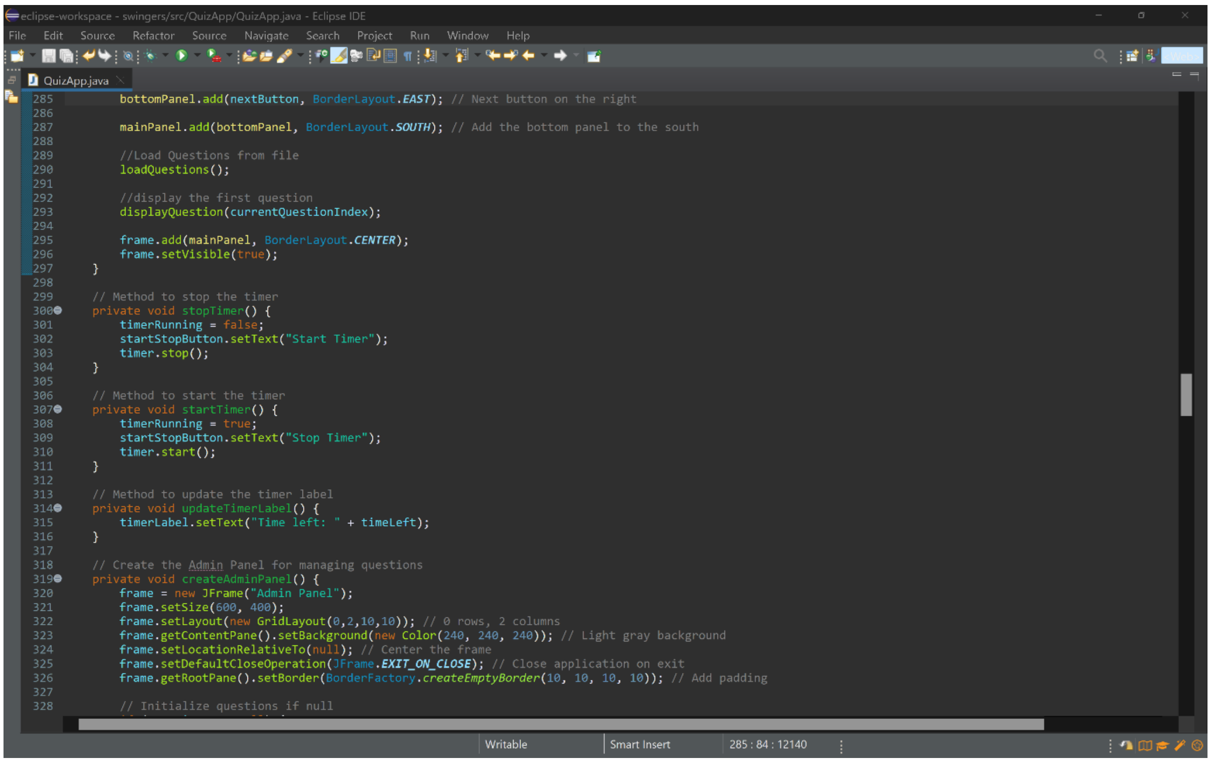
```java
        }
        return false;
    }

    // Create the Quiz Panel for regular users
    private void createQuizPanel() {
        frame = new JFrame("Quiz Application");
        frame.setSize(500, 400);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLayout(new BorderLayout());
        frame.setLocationRelativeTo(null); // Center the frame

        // Add top panel with logout button
        frame.add(createTopPanelWithLogout(), BorderLayout.NORTH);

        // Set background color
        JPanel mainPanel = new JPanel();
        mainPanel.setBackground(new Color(240, 240, 240)); // Light gray background
        mainPanel.setLayout(new BorderLayout());
        mainPanel.setBorder(BorderFactory.createEmptyBorder(15, 15, 15, 15)); // Main padding

        // Timer Label
        timerLabel = new JLabel("Time left: " + timeLeft);
        timerLabel.setFont(new Font("Consolas", Font.BOLD, 16));
        timerLabel.setForeground(new Color(0, 150, 136)); // Teal text

        // Timer Button
        startStopButton = new JButton("Start Timer");
        startStopButton.setFont(new Font("Consolas",Font.PLAIN, 12));
        startStopButton.setBackground(new Color(0,150,136)); //teal background
        startStopButton.setForeground(Color.WHITE); //White Text Color
        startStopButton.setBorder(BorderFactory.createEmptyBorder(5,10,5,10)); //padding for smaller button

        startStopButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                if (timerRunning) {
                    stopTimer();
                } else {
                    startTimer();
                }
            }
        });

```


eclipse-workspace - swingers/src/QuizApp/QuizApp.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
QuizApp.java
Writable
Smart Insert
197 : 10 : 7864

```java
        // Timer that updates every second
        timer = new Timer(1000, new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                if (timeLeft > 0) {
                    timeLeft--;
                    updateTimerLabel();
                } else {
                    timer.stop();
                    timerRunning = false;
                    startStopButton.setText("Start Timer");
                    JOptionPane.showMessageDialog(frame, "Time's up! Choice recorded.");
                    checkAnswer(); // Automatically check the answer when time runs out
                    currentQuestionIndex++;
                    if (currentQuestionIndex < questions.size()) {
                        displayQuestion(currentQuestionIndex); // Display the next question
                    } else {
                        endQuiz(); // End the quiz if there are no more questions
                    }
                }
            }
        });

        // Create a panel for the timer and Start/Stop button
        JPanel timerPanel = new JPanel();
        timerPanel.setLayout(new FlowLayout()); // Flow layout for horizontal alignment
        timerPanel.setBackground(new Color(240, 240, 240)); // Match panel color
        timerPanel.add(timerLabel);
        timerPanel.add(startStopButton); // Add the start/stop button to the timer panel

        // Question Label
        questionLabel = new JLabel("Question will appear here");
        questionLabel.setHorizontalAlignment(SwingConstants.CENTER); // Center the question
        questionLabel.setFont(new Font("Times New Roman", Font.BOLD, 20));
        questionLabel.setBackground(new Color(0, 150, 136)); // Teal background
        questionLabel.setForeground(Color.WHITE); // White text
        questionLabel.setOpaque(true); // Make background visible
        mainPanel.add(questionLabel, BorderLayout.NORTH);

        // Option Buttons
        JPanel optionPanel = new JPanel();
        optionPanel.setLayout(new GridLayout(4, 1));
        optionPanel.setBorder(BorderFactory.createEmptyBorder(10, 0, 10, 0)); // Option padding
        optionButtons = new JRadioButton[4];
```

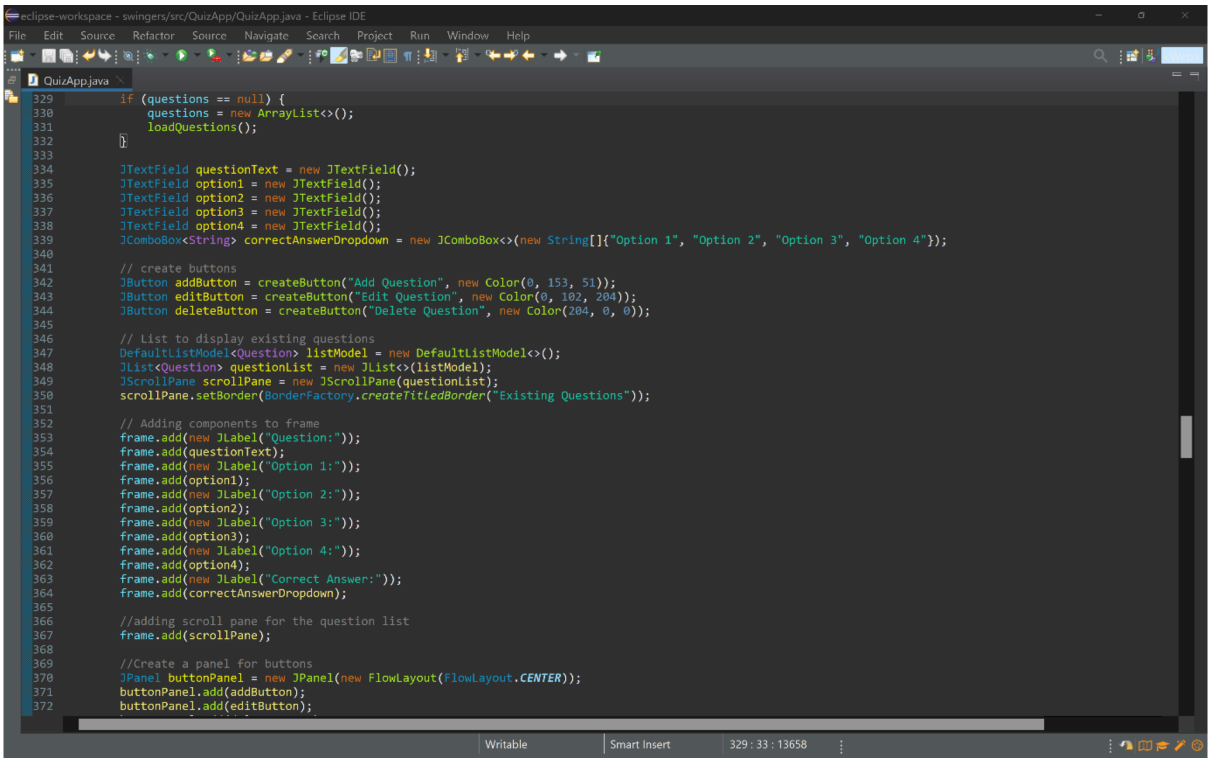

eclipse-workspace - swingers/src/QuizApp/QuizApp.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
QuizApp.java
Writable
Smart Insert
241 : 6 : 10098

```java
        optionGroup = new ButtonGroup();
        for (int i = 0; i < 4; i++) {
            optionButtons[i] = new JRadioButton();
            optionButtons[i].setFont(new Font("Consolas", Font.PLAIN, 16));
            optionButtons[i].setBackground(new Color(240, 240, 240)); // Match panel color
            optionGroup.add(optionButtons[i]);
            optionPanel.add(optionButtons[i]);
        }
        mainPanel.add(optionPanel, BorderLayout.CENTER);

        // Next Button
        nextButton = new JButton("Next");
        nextButton.setFont(new Font("Times New Roman", Font.BOLD, 16));
        nextButton.setBackground(new Color(0, 153, 51)); // Green background
        nextButton.setForeground(Color.WHITE); // White text

        nextButton.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseEntered(java.awt.event.MouseEvent evt) {
                nextButton.setBackground(new Color(0, 180, 60)); // Lighter green on hover
            }

            public void mouseExited(java.awt.event.MouseEvent evt) {
                nextButton.setBackground(new Color(0, 153, 51)); // Original green
            }
        });

        nextButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                checkAnswer();
                currentQuestionIndex++;
                if (currentQuestionIndex < questions.size()) {
                    displayQuestion(currentQuestionIndex);
                } else {
                    endQuiz();
                }
            }
        });

        // Create a bottom panel to hold the timer panel and next button
        JPanel bottomPanel = new JPanel();
        bottomPanel.setLayout(new BorderLayout());
        bottomPanel.setBackground(new Color(240, 240, 240)); // Match panel color
        bottomPanel.add(timerPanel, BorderLayout.WEST); // Timer and button on the left
```

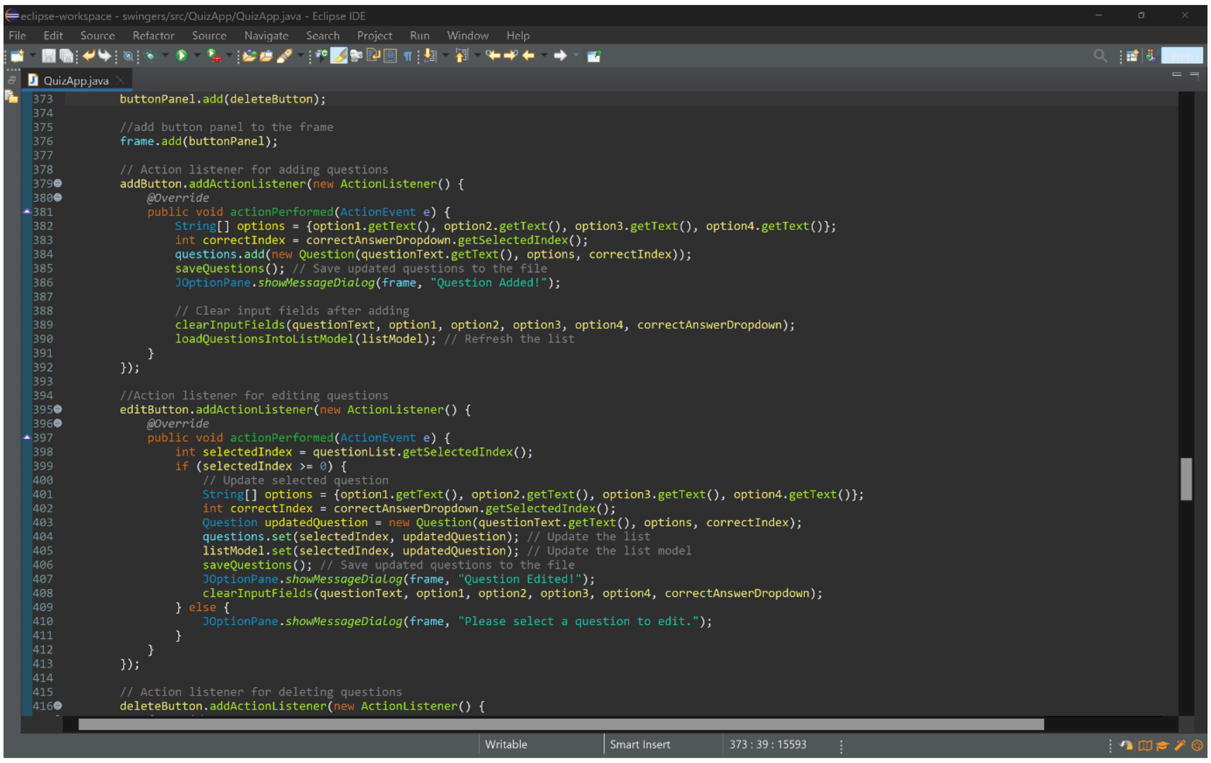

eclipse-workspace - swingers/src/QuizApp/QuizApp.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
QuizApp.java

```java
        bottomPanel.add(nextButton, BorderLayout.EAST); // Next button on the right

        mainPanel.add(bottomPanel, BorderLayout.SOUTH); // Add the bottom panel to the south

        //Load Questions from file
        loadQuestions();

        //display the first question
        displayQuestion(currentQuestionIndex);

        frame.add(mainPanel, BorderLayout.CENTER);
        frame.setVisible(true);
    }

    // Method to stop the timer
    private void stopTimer() {
        timerRunning = false;
        startStopButton.setText("Start Timer");
        timer.stop();
    }

    // Method to start the timer
    private void startTimer() {
        timerRunning = true;
        startStopButton.setText("Stop Timer");
        timer.start();
    }

    // Method to update the timer label
    private void updateTimerLabel() {
        timerLabel.setText("Time left: " + timeLeft);
    }

    // Create the Admin Panel for managing questions
    private void createAdminPanel() {
        frame = new JFrame("Admin Panel");
        frame.setSize(600, 400);
        frame.setLayout(new GridLayout(0,2,10,10)); // 0 rows, 2 columns
        frame.getContentPane().setBackground(new Color(240, 240, 240)); // Light gray background
        frame.setLocationRelativeTo(null); // Center the frame
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); // Close application on exit
        frame.getRootPane().setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10)); // Add padding

        // Initialize questions if null
```

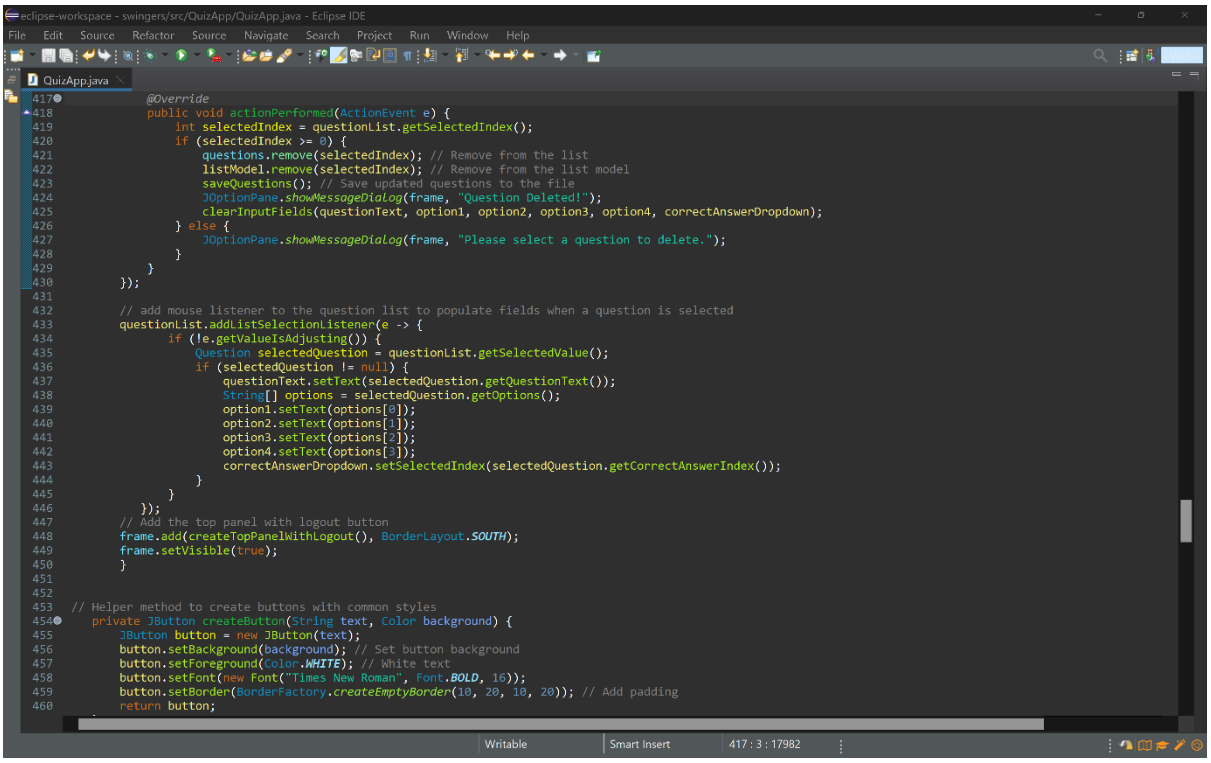
Writable
Smart Insert
285 : 84 : 12140


eclipse-workspace - swingers/src/QuizApp/QuizApp.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
QuizApp.java

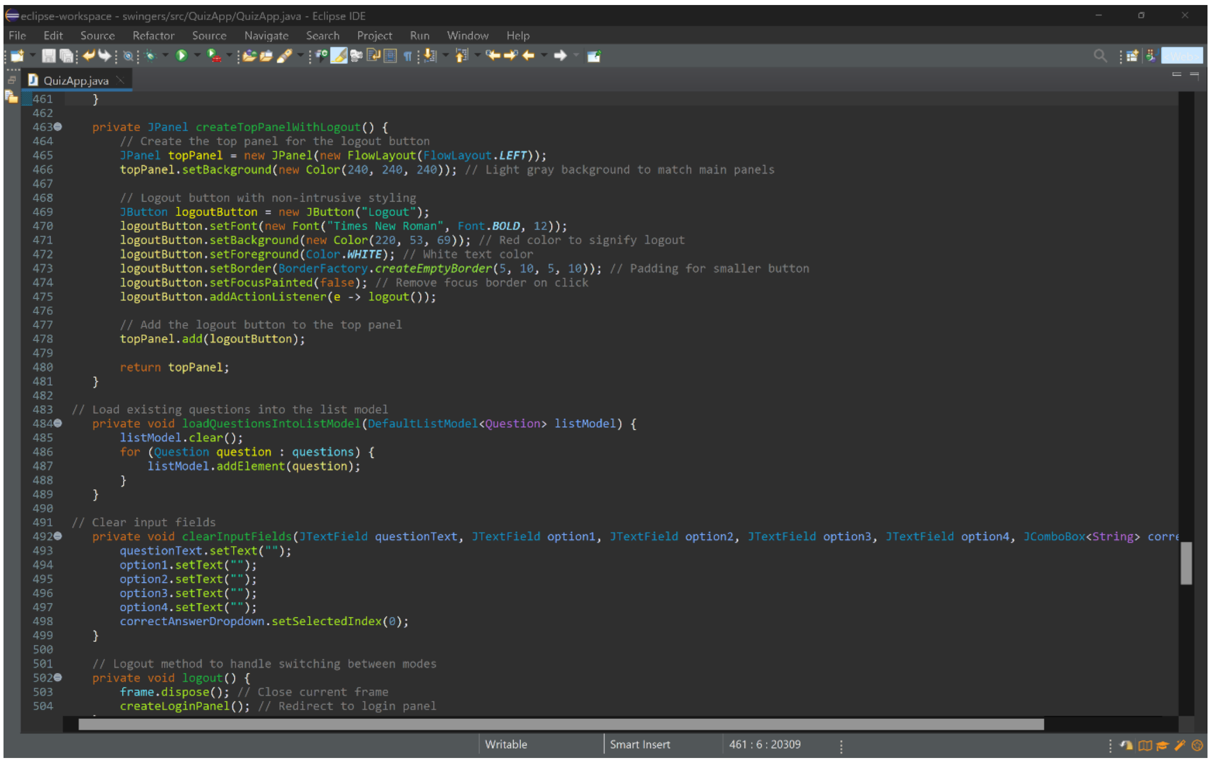
```java
        if (questions == null) {
            questions = new ArrayList<>();
            loadQuestions();
        }

        JTextField questionText = new JTextField();
        JTextField option1 = new JTextField();
        JTextField option2 = new JTextField();
        JTextField option3 = new JTextField();
        JTextField option4 = new JTextField();
        JComboBox<String> correctAnswerDropdown = new JComboBox<>(new String[]{"Option 1", "Option 2", "Option 3", "Option 4"});

        // create buttons
        JButton addButton = createButton("Add Question", new Color(0, 153, 51));
        JButton editButton = createButton("Edit Question", new Color(0, 102, 204));
        JButton deleteButton = createButton("Delete Question", new Color(204, 0, 0));

        // List to display existing questions
        DefaultListModel<Question> listModel = new DefaultListModel<>();
        JList<Question> questionList = new JList<>(listModel);
        JScrollPane scrollPane = new JScrollPane(questionList);
        scrollPane.setBorder(BorderFactory.createTitledBorder("Existing Questions"));

        // Adding components to frame
        frame.add(new JLabel("Question:"));
        frame.add(questionText);
        frame.add(new JLabel("Option 1:"));
        frame.add(option1);
        frame.add(new JLabel("Option 2:"));
        frame.add(option2);
        frame.add(new JLabel("Option 3:"));
        frame.add(option3);
        frame.add(new JLabel("Option 4:"));
        frame.add(option4);
        frame.add(new JLabel("Correct Answer:"));
        frame.add(correctAnswerDropdown);

        //adding scroll pane for the question list
        frame.add(scrollPane);

        //Create a panel for buttons
        JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.CENTER));
        buttonPanel.add(addButton);
        buttonPanel.add(editButton);
```


eclipse-workspace - swingers/src/QuizApp/QuizApp.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
QuizApp.java
Writable
Smart Insert
373 : 39 : 15593

```java
        buttonPanel.add(deleteButton);

        //add button panel to the frame
        frame.add(buttonPanel);

        // Action listener for adding questions
        addButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                String[] options = {option1.getText(), option2.getText(), option3.getText(), option4.getText()};
                int correctIndex = correctAnswerDropdown.getSelectedIndex();
                questions.add(new Question(questionText.getText(), options, correctIndex));
                saveQuestions(); // Save updated questions to the file
                JOptionPane.showMessageDialog(frame, "Question Added!");

                // Clear input fields after adding
                clearInputFields(questionText, option1, option2, option3, option4, correctAnswerDropdown);
                loadQuestionsIntoListModel(listModel); // Refresh the list
            }
        });

        //Action listener for editing questions
        editButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                int selectedIndex = questionList.getSelectedIndex();
                if (selectedIndex >= 0) {
                    // Update selected question
                    String[] options = {option1.getText(), option2.getText(), option3.getText(), option4.getText()};
                    int correctIndex = correctAnswerDropdown.getSelectedIndex();
                    Question updatedQuestion = new Question(questionText.getText(), options, correctIndex);
                    questions.set(selectedIndex, updatedQuestion); // Update the list
                    listModel.set(selectedIndex, updatedQuestion); // Update the list model
                    saveQuestions(); // Save updated questions to the file
                    JOptionPane.showMessageDialog(frame, "Question Edited!");
                    clearInputFields(questionText, option1, option2, option3, option4, correctAnswerDropdown);
                } else {
                    JOptionPane.showMessageDialog(frame, "Please select a question to edit.");
                }
            }
        });

        // Action listener for deleting questions
        deleteButton.addActionListener(new ActionListener() {
```

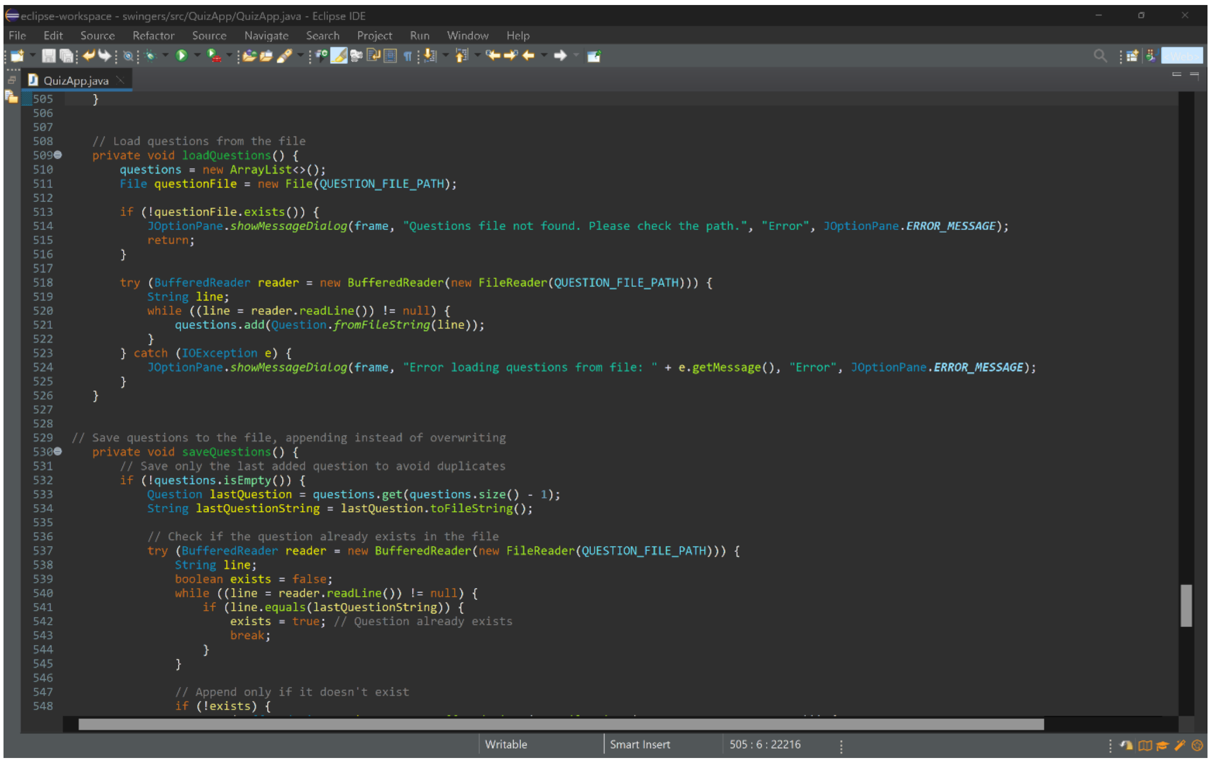

eclipse-workspace - swingers/src/QuizApp/QuizApp.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
QuizApp.java
Writable
Smart Insert
417 : 3 : 17982

```java
            @Override
            public void actionPerformed(ActionEvent e) {
                int selectedIndex = questionList.getSelectedIndex();
                if (selectedIndex >= 0) {
                    questions.remove(selectedIndex); // Remove from the list
                    listModel.remove(selectedIndex); // Remove from the list model
                    saveQuestions(); // Save updated questions to the file
                    JOptionPane.showMessageDialog(frame, "Question Deleted!");
                    clearInputFields(questionText, option1, option2, option3, option4, correctAnswerDropdown);
                } else {
                    JOptionPane.showMessageDialog(frame, "Please select a question to delete.");
                }
            }
        });

        // add mouse listener to the question list to populate fields when a question is selected
        questionList.addListSelectionListener(e -> {
               if (!e.getValueIsAdjusting()) {
                   Question selectedQuestion = questionList.getSelectedValue();
                   if (selectedQuestion != null) {
                       questionText.setText(selectedQuestion.getQuestionText());
                       String[] options = selectedQuestion.getOptions();
                       option1.setText(options[0]);
                       option2.setText(options[1]);
                       option3.setText(options[2]);
                       option4.setText(options[3]);
                       correctAnswerDropdown.setSelectedIndex(selectedQuestion.getCorrectAnswerIndex());
                   }
               }
           });
        // Add the top panel with logout button
        frame.add(createTopPanelWithLogout(), BorderLayout.SOUTH);
        frame.setVisible(true);
        }


 // Helper method to create buttons with common styles
    private JButton createButton(String text, Color background) {
        JButton button = new JButton(text);
        button.setBackground(background); // Set button background
        button.setForeground(Color.WHITE); // White text
        button.setFont(new Font("Times New Roman", Font.BOLD, 16));
        button.setBorder(BorderFactory.createEmptyBorder(10, 20, 10, 20)); // Add padding
        return button;
```


eclipse-workspace - swingers/src/QuizApp/QuizApp.java - Eclipse IDE

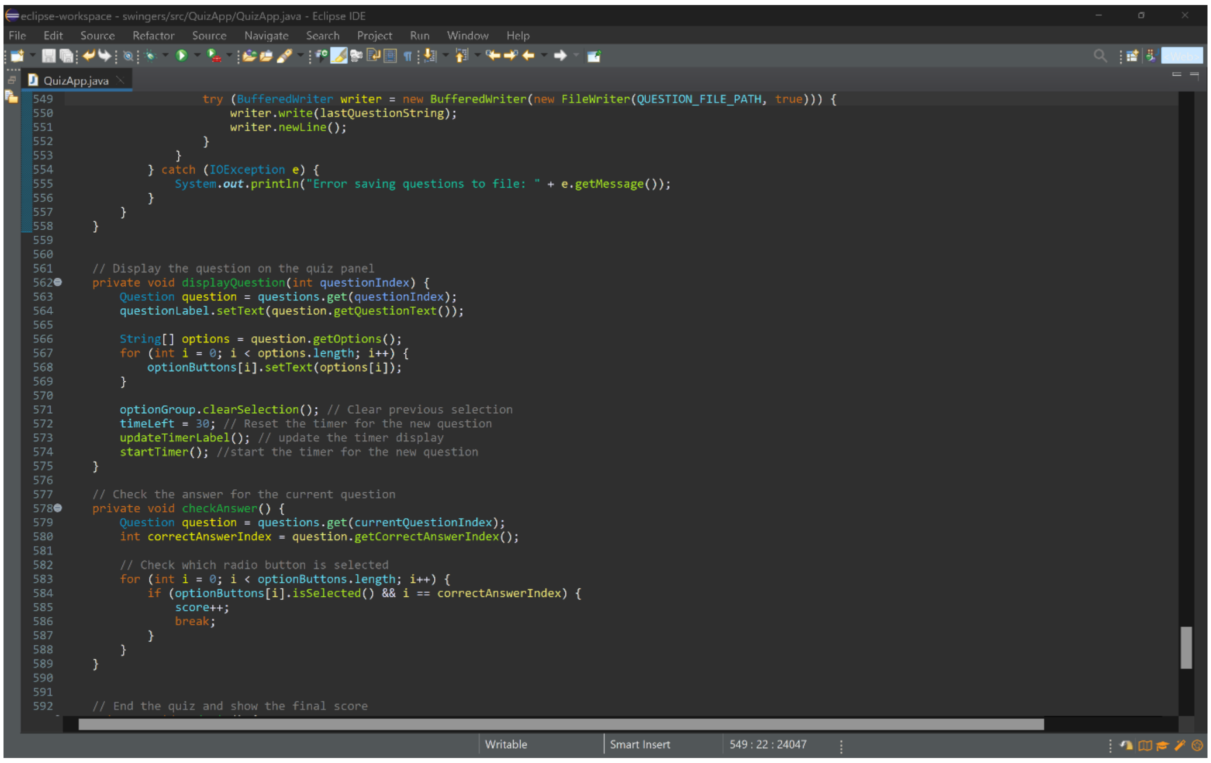
```java
    }

    private JPanel createTopPanelWithLogout() {
        // Create the top panel for the logout button
        JPanel topPanel = new JPanel(new FlowLayout(FlowLayout.LEFT));
        topPanel.setBackground(new Color(240, 240, 240)); // Light gray background to match main panels

        // Logout button with non-intrusive styling
        JButton logoutButton = new JButton("Logout");
        logoutButton.setFont(new Font("Times New Roman", Font.BOLD, 12));
        logoutButton.setBackground(new Color(220, 53, 69)); // Red color to signify logout
        logoutButton.setForeground(Color.WHITE); // White text color
        logoutButton.setBorder(BorderFactory.createEmptyBorder(5, 10, 5, 10)); // Padding for smaller button
        logoutButton.setFocusPainted(false); // Remove focus border on click
        logoutButton.addActionListener(e -> logout());

        // Add the logout button to the top panel
        topPanel.add(logoutButton);

        return topPanel;
    }

 // Load existing questions into the list model
    private void loadQuestionsIntoListModel(DefaultListModel<Question> listModel) {
        listModel.clear();
        for (Question question : questions) {
            listModel.addElement(question);
        }
    }

 // Clear input fields
    private void clearInputFields(JTextField questionText, JTextField option1, JTextField option2, JTextField option3, JTextField option4, JComboBox<String> corre
        questionText.setText("");
        option1.setText("");
        option2.setText("");
        option3.setText("");
        option4.setText("");
        correctAnswerDropdown.setSelectedIndex(0);
    }

    // Logout method to handle switching between modes
    private void logout() {
        frame.dispose(); // Close current frame
        createLoginPanel(); // Redirect to login panel
```

Writable   Smart Insert   461 : 6 : 20309


eclipse-workspace - swingers/src/QuizApp/QuizApp.java - Eclipse IDE

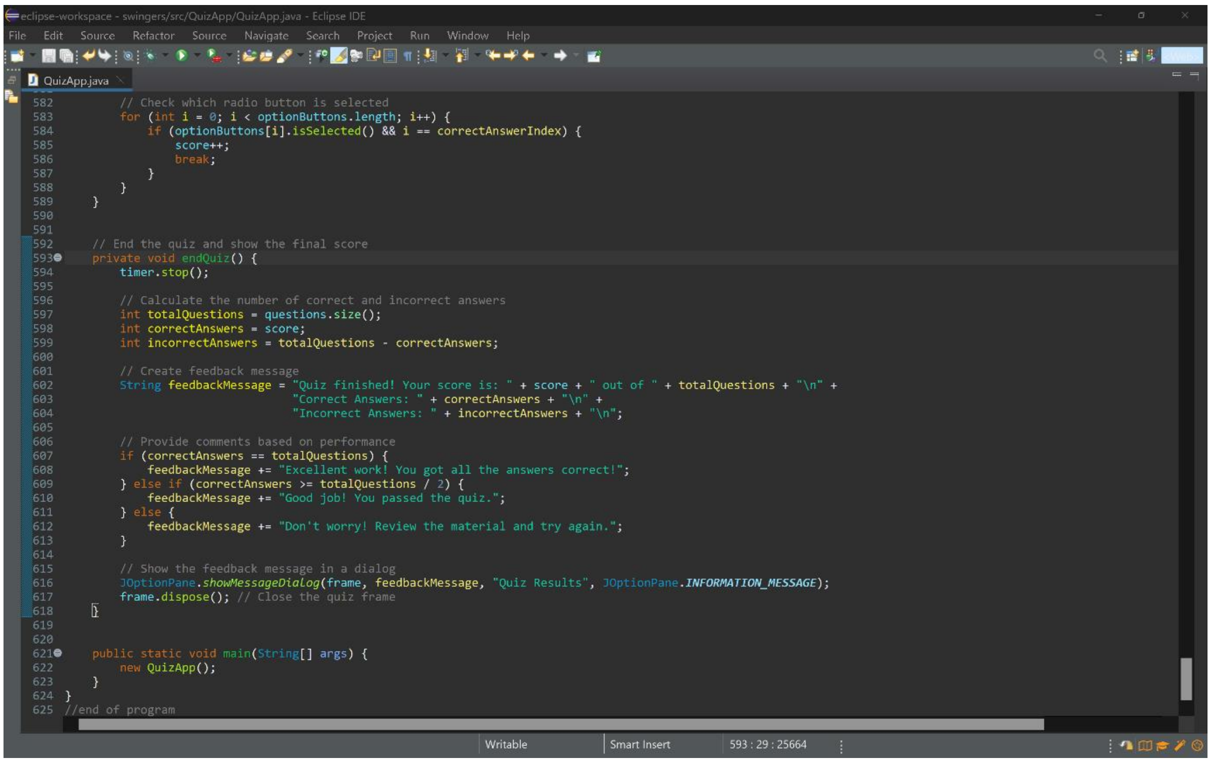
```java
    }


    // Load questions from the file
    private void loadQuestions() {
        questions = new ArrayList<>();
        File questionFile = new File(QUESTION_FILE_PATH);

        if (!questionFile.exists()) {
            JOptionPane.showMessageDialog(frame, "Questions file not found. Please check the path.", "Error", JOptionPane.ERROR_MESSAGE);
            return;
        }

        try (BufferedReader reader = new BufferedReader(new FileReader(QUESTION_FILE_PATH))) {
            String line;
            while ((line = reader.readLine()) != null) {
                questions.add(Question.fromFileString(line));
            }
        } catch (IOException e) {
            JOptionPane.showMessageDialog(frame, "Error loading questions from file: " + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
        }
    }


 // Save questions to the file, appending instead of overwriting
    private void saveQuestions() {
        // Save only the last added question to avoid duplicates
        if (!questions.isEmpty()) {
            Question lastQuestion = questions.get(questions.size() - 1);
            String lastQuestionString = lastQuestion.toFileString();

            // Check if the question already exists in the file
            try (BufferedReader reader = new BufferedReader(new FileReader(QUESTION_FILE_PATH))) {
                String line;
                boolean exists = false;
                while ((line = reader.readLine()) != null) {
                    if (line.equals(lastQuestionString)) {
                        exists = true; // Question already exists
                        break;
                    }
                }

                // Append only if it doesn't exist
                if (!exists) {
```

Writable   Smart Insert   505 : 6 : 22216


eclipse-workspace - swingers/src/QuizApp/QuizApp.java - Eclipse IDE

```java
                    try (BufferedWriter writer = new BufferedWriter(new FileWriter(QUESTION_FILE_PATH, true))) {
                        writer.write(lastQuestionString);
                        writer.newLine();
                    }
                }
            } catch (IOException e) {
                System.out.println("Error saving questions to file: " + e.getMessage());
            }
        }
    }


    // Display the question on the quiz panel
    private void displayQuestion(int questionIndex) {
        Question question = questions.get(questionIndex);
        questionLabel.setText(question.getQuestionText());

        String[] options = question.getOptions();
        for (int i = 0; i < options.length; i++) {
            optionButtons[i].setText(options[i]);
        }

        optionGroup.clearSelection(); // Clear previous selection
        timeLeft = 30; // Reset the timer for the new question
        updateTimerLabel(); // update the timer display
        startTimer(); //start the timer for the new question
    }

    // Check the answer for the current question
    private void checkAnswer() {
        Question question = questions.get(currentQuestionIndex);
        int correctAnswerIndex = question.getCorrectAnswerIndex();

        // Check which radio button is selected
        for (int i = 0; i < optionButtons.length; i++) {
            if (optionButtons[i].isSelected() && i == correctAnswerIndex) {
                score++;
                break;
            }
        }
    }


    // End the quiz and show the final score
```

Writable   Smart Insert   549 : 22 : 24047

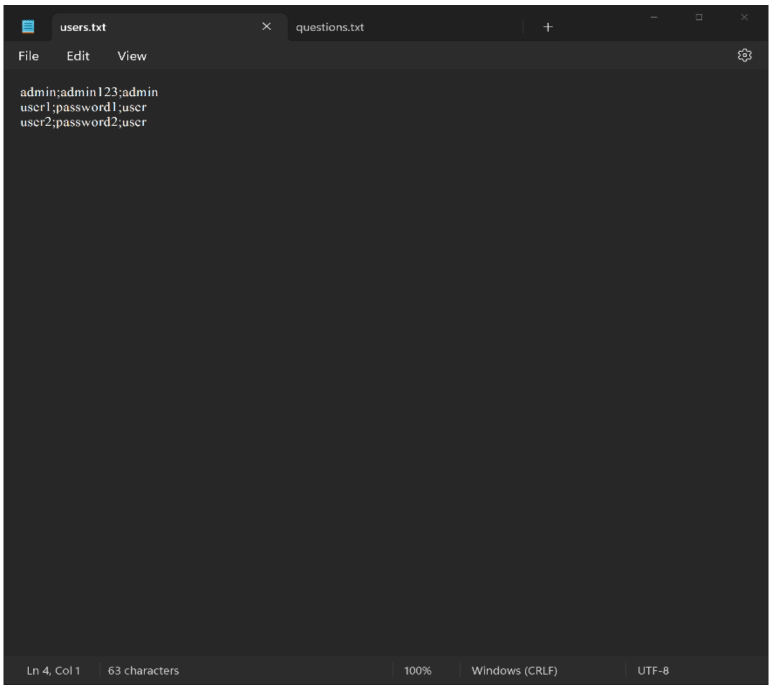

eclipse-workspace - swingers/src/QuizApp/QuizApp.java - Eclipse IDE

```java
        // Check which radio button is selected
        for (int i = 0; i < optionButtons.length; i++) {
            if (optionButtons[i].isSelected() && i == correctAnswerIndex) {
                score++;
                break;
            }
        }
    }


    // End the quiz and show the final score
    private void endQuiz() {
        timer.stop();

        // Calculate the number of correct and incorrect answers
        int totalQuestions = questions.size();
        int correctAnswers = score;
        int incorrectAnswers = totalQuestions - correctAnswers;

        // Create feedback message
        String feedbackMessage = "Quiz finished! Your score is: " + score + " out of " + totalQuestions + "\n" +
                                 "Correct Answers: " + correctAnswers + "\n" +
                                 "Incorrect Answers: " + incorrectAnswers + "\n";

        // Provide comments based on performance
        if (correctAnswers == totalQuestions) {
            feedbackMessage += "Excellent work! You got all the answers correct!";
        } else if (correctAnswers >= totalQuestions / 2) {
            feedbackMessage += "Good job! You passed the quiz.";
        } else {
            feedbackMessage += "Don't worry! Review the material and try again.";
        }

        // Show the feedback message in a dialog
        JOptionPane.showMessageDialog(frame, feedbackMessage, "Quiz Results", JOptionPane.INFORMATION_MESSAGE);
        frame.dispose(); // Close the quiz frame
    }


    public static void main(String[] args) {
        new QuizApp();
    }
}
//end of program
```

Writable   Smart Insert   593 : 29 : 25664

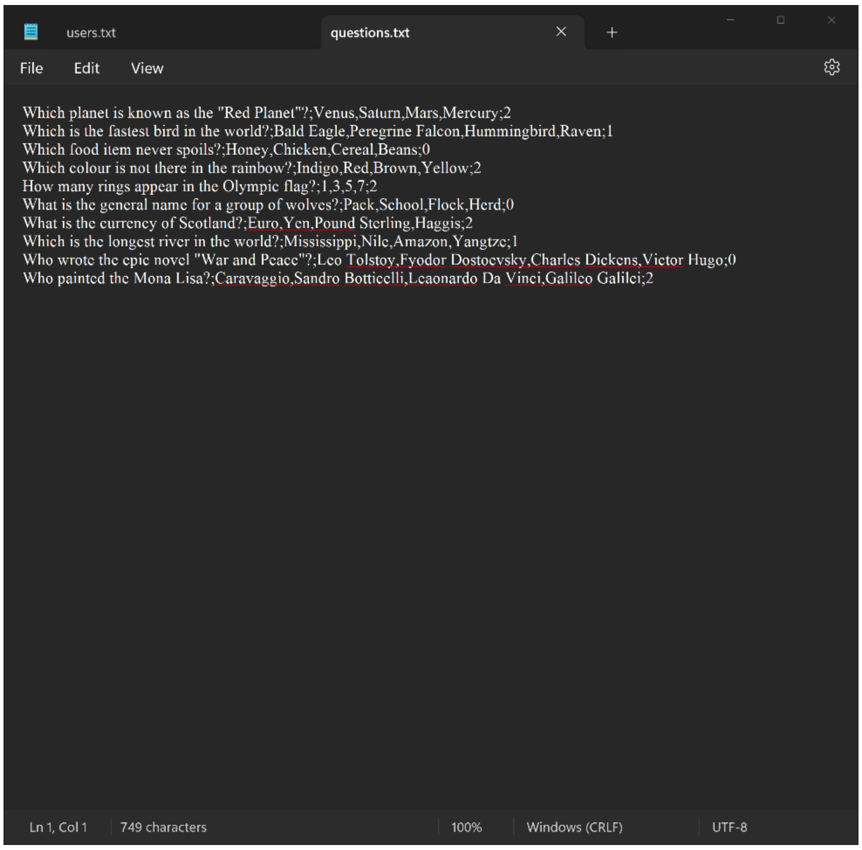
Files used for file handling

users.txt



```
admin;admin123;admin
user1;password1;user
user2;password2;user
```

questions.txt



```
Which planet is known as the "Red Planet"?;Venus,Saturn,Mars,Mercury;2
Which is the fastest bird in the world?;Bald Eagle,Peregrine Falcon,Hummingbird,Raven;1
Which food item never spoils?;Honey,Chicken,Cereal,Beans;0
Which colour is not there in the rainbow?;Indigo,Red,Brown,Yellow;2
How many rings appear in the Olympic flag?;1,3,5,7;2
What is the general name for a group of wolves?;Pack,School,Flock,Herd;0
What is the currency of Scotland?;Euro,Yen,Pound Sterling,Haggis;2
Which is the longest river in the world?;Mississippi,Nile,Amazon,Yangtze;1
Who wrote the epic novel "War and Peace"?;Leo Tolstoy,Fyodor Dostoevsky,Charles Dickens,Victor Hugo;0
Who painted the Mona Lisa?;Caravaggio,Sandro Botticelli,Leaonardo Da Vinci,Galileo Galilei;2
```

References

- https://github.com/ankshirc/GUI-Based-Quiz-Application.git
- https://www.coursera.org/learn/writing-java-code-for-applications/home/info
- https://www.coursera.org/projects/build-java-gui-apps
- Lesson: Using Swing Components (The Java™ Tutorials > Creating a GUI With Swing)
- Java Swing Tutorial - javatpoint

THANK YOU