Report On

Quiz Application

Submitted in partial fulfillment of the requirements of the Course project in

Semester III of **Second Year Artificial Intelligence & Data Science**

By

Yash Kerkar (Roll No. S238267105)
Kunal Rajput (Roll No. S238307104)
Komal Sapatale (Roll No. S238317202)
Ankit Bari (Roll No. S238247105)

Supervisor

Prof. Sneha Yadav

Vidyavardhini's College of Engineering & Technology

Department of Artificial Intelligence & Data science

(2023-24)

Vidyavardhini's College of Engineering & Technology

Department of Artificial Intelligence & Data science

CERTIFICATE

This is to certify that the project entitled "**Quiz Application**" is a bonafide work of "**Yash Kerkar** (Roll No. S238267105), **Kunal Rajput** (Roll No. S238307104), **Komal Sapatale** (Roll No. S238317202), **Ankit Bari** (Roll No. S238247105)" submitted to the University of Mumbai in partial fulfillment of the requirement for the Course project in semester III of Second Year Artificial Intelligence & Data Science.

Supervisor
Prof. Sneha Yadav

Internal Examiner

External Examiner

Prof. Sejal Dmello Assistant HOD of Dept Of AIDS	**Dr. Tatwadarshi P.N.** Head of Department of of AIDS	**Dr. Harish Vankudre** Principal

Abstract

A quiz program using Java and Applet is a web-based educational application designed to create interactive quizzes and tests that can be accessed through a web browser. This program leverages Java's Applet technology to provide a dynamic and engaging way for users to participate in quizzes. The main components and functionalities of this quiz program include:

1. User Interface: The program provides a user-friendly interface where users can select a quiz, view questions, and submit their answers.
2. Quiz Creation: Educators or quiz administrators can create quizzes by inputting questions, answer choices, and correct answers into the system. This data is typically stored in a structured format such as XML.
3. Interactive Quizzes: Users can select a quiz from the available options and interact with the questions. The program displays one question at a time, along with multiple-choice answer options.

Introduction

A quiz application is an applet program designed to facilitate quizzes, tests, and interactive knowledge-based games. It is a versatile tool that can be used for educational purposes, entertainment, or to test and challenge a user's knowledge on a wide range of topics. Quiz applications have become increasingly popular due to their accessibility, interactivity, and the ability to engage users in a fun and informative way.

Key Features of the Quiz Application:

1.User Registration and Profiles: Users can create accounts, personalize their profiles, and track their progress over time. This feature allows for a more engaging and personalized experience.

2.Quiz Creation: Quiz creators can generate quizzes with various question formats, such as multiple-choice, true/false, or open-ended questions. They can also add multimedia elements like images or videos to enhance the questions.

3.Categories and Topics: Quizzes can be categorized into different topics or subjects, making it easy for users to find quizzes related to their interests or areas of expertise.

4.Difficulty Levels: Quizzes can be customized to offer different difficulty levels, from easy to advanced, catering to a wide range of users.

5.Leaderboards: Users can compete with others and view their rankings on leaderboards. This competitive element adds an extra layer of engagement and motivation.

Background

AWT:

1. Creating Windows and Frames: AWT allows you to create the main window or frame for your quiz application where you can place various components, such as questions, answer choices, and buttons.

2. Components: You can use AWT components like Label for displaying questions, Checkbox for answer choices, and Button for actions like submitting answers or moving to the next question.

3. Layout Management: AWT provides layout managers (e.g., FlowLayout, BorderLayout, GridLayout) to help you organize and position the GUI components within the frame.

4. Event Handling: AWT allows you to handle user interactions, such as button clicks, using event listeners. You can define actions to be taken when users submit answers or navigate through questions.

5. Graphics: AWT provides basic graphics capabilities, so you can display images, shapes, and other visual elements in your quiz application.

Swing

Swing is a GUI (Graphical User Interface) toolkit for Java that is an extension of AWT (Abstract Window Toolkit). It provides a more modern and feature-rich set of GUI components and capabilities compared to AWT. Swing is commonly used for developing interactive and visually appealing desktop applications, including quiz applications.

In the context of a quiz application, Swing offers several advantages:

1.Richer Component Set: Swing provides a wide range of GUI components, including buttons, labels, text fields, radio buttons, checkboxes, and more. These components can be used to create a more interactive and visually appealing quiz interface.

2.Customization: Swing components can be easily customized, allowing you to create attractive and user-friendly interfaces. You can apply different fonts, colors, and styles to components.

3.Layout Management: Swing offers layout managers like FlowLayout, BorderLayout, GridLayout, and more, which help you arrange and position components within your application window.

4.Event Handling: Swing supports event handling, allowing you to respond to user interactions, such as button clicks or menu selections.

5.Graphics and Icons: You can use Swing to work with images, icons, and graphics, which can be useful for incorporating images into your quiz application.

6.Advanced Features: Swing provides advanced components like JTable for tabular data, JComboBox for dropdown menus, and JScrollPane for scrollable content, which can be handy in creating feature-rich quiz applications.

AWT Events

1. ActionEvent: Action events occur when the user performs an action, like clicking a button. In a quiz application, you might use ActionEvents to handle actions such as submitting answers.

2. ItemEvent: Item events are often used with checkboxes and choice components. You can use ItemEvents to track user selections, particularly for multiple-choice questions.

3. WindowEvent: Window events are triggered when window-related actions occur, such as opening, closing, resizing, or minimizing the application's window. You can use WindowEvents for managing the quiz application's window.

4. MouseEvent: MouseEvent events handle mouse interactions, including clicks, movements, and dragging. You might use MouseEvent events to enable interactions like selecting answers or navigating through the application.

5. KeyEvent: KeyEvent events deal with keyboard input from the user. These events can be used for keyboard shortcuts or navigation within the quiz application.

Action Listener

In a quiz application, an ActionListener is an important component for handling user interactions, particularly for actions such as submitting answers, moving to the next question, or navigating through the application. The ActionListener interface in Java is used to respond to events triggered by user actions, most commonly by clicking on buttons.

Difference Between AWT and Swing

AWT	Swing
AWT components are heavyweight components	Swing components are lightweight components
AWT doesn't support pluggable look and feel	Swing supports pluggable look and feel
AWT programs are not portable	Swing programs are portable
AWT is old framework for creating GUIs	Swing is new framework for creating GUIs
AWT components require java.awt package	Swing components require javax.swing package
AWT supports limited number of GUI controls	Swing provides advanced GUI controls like Jtable, JTabbedPane etc
More code is needed to implement AWT controls functionality	Less code is needed to implement swing controls functionality
AWT doesn't follow MVC	Swing follows MVC

Literature Review

1. Purpose: The Online Core Java Quiz Application is designed for conducting online quizzes related to core Java concepts. Users can answer a series of multiple-choice questions and receive their scores at the end.

2. User Interface: The project uses Java's Swing library to create a graphical user interface. The interface includes a question label, radio buttons for answer choices, "Next" and "Bookmark" buttons, and buttons for bookmarked questions. The user interface allows users to select answers and navigate through the quiz.

3. Quiz Logic: The quiz logic is implemented within the set() and check() methods. The set() method is responsible for displaying questions and answer choices, and the check() method checks whether the user's selected answer is correct for the current question.

4. Bookmark Feature: Users have the option to bookmark questions for later review. Bookmarked questions are stored in an array, and users can revisit them by clicking on the corresponding "Bookmark" buttons.

5. Flow Control: The "Next" button allows users to move to the next question, and when the user reaches the last question, it changes to a "Show Result" button, which displays the final score when clicked.

6. Result Display: The project uses a JOptionPane to display the user's final score upon completing the quiz.

7. Code Structure: The code is organized into a single class named "OnlineTest." It utilizes instance variables for components like labels, radio buttons, and buttons. The actionPerformed() method handles button clicks and controls the logic for advancing to the next question, bookmarking, and displaying results.

Code:

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class OnlineTest extends JFrame implements ActionListener
{
   JLabel l;
   JRadioButton jb[]=new JRadioButton[5];
   JButton b1,b2;
   ButtonGroup bg;
   int count=0,current=0,x=1,y=1,now=0;
   int m[]=new int[10];
   OnlineTest(String s)
   {
      super(s);
      l=new JLabel();
      add(l);
      bg=new ButtonGroup();
      for(int i=0;i<5;i++)
      {
         jb[i]=new JRadioButton();
         add(jb[i]);
         bg.add(jb[i]);
      }
      b1=new JButton("Next");
      b2=new JButton("Bookmark");
      b1.addActionListener(this);
      b2.addActionListener(this);
      add(b1);
      add(b2);
      set();
      l.setBounds(30,40,450,20);
      jb[0].setBounds(50,80,100,20);
      jb[1].setBounds(50,110,100,20);
      jb[2].setBounds(50,140,100,20);
      jb[3].setBounds(50,170,100,20);
      b1.setBounds(100,240,100,30);
      b2.setBounds(270,240,150,30);
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      setLayout(null);
      setLocation(250,100);
      setVisible(true);
```

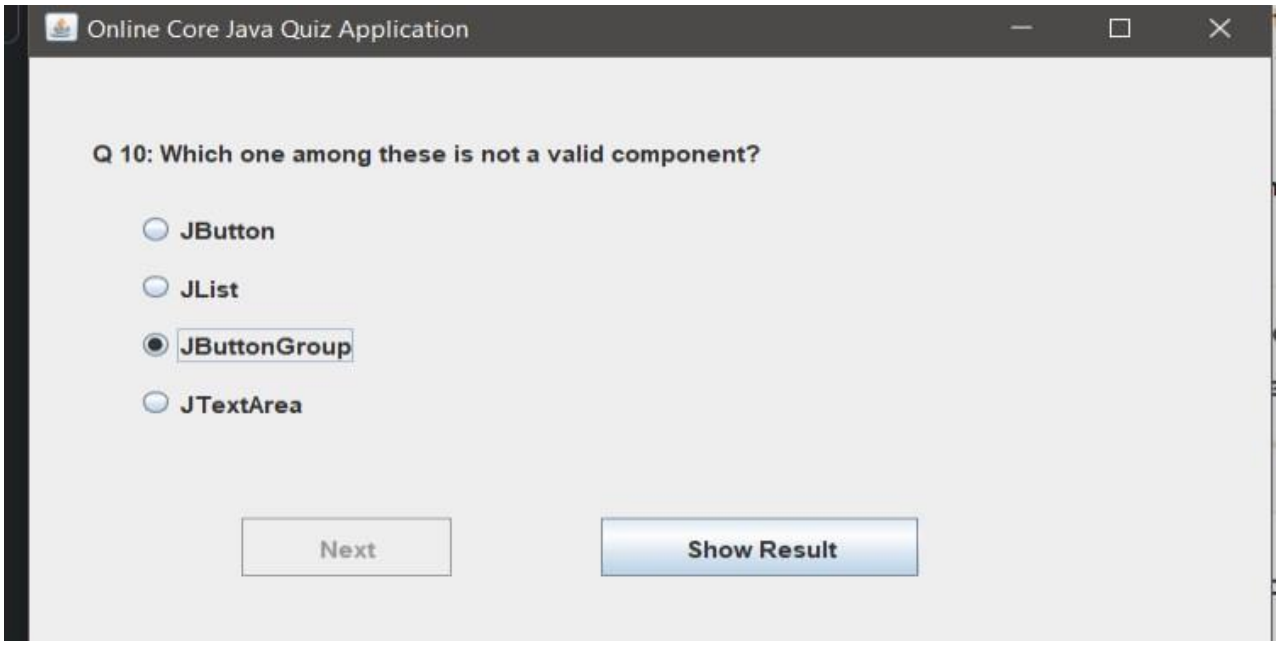
```
    setSize(600,350);
}
public void actionPerformed(ActionEvent e)
{
    if(e.getSource()==b1)
    {
        if(check())
            count=count+1;
        current++;
        set();
        if(current==9)
        {
            b1.setEnabled(false);
            b2.setText("Show Result");
        }
    }
    if(e.getActionCommand().equals("Bookmark"))
    {
        JButton bk=new JButton("Bookmark"+x);
        bk.setBounds(480,20+30*x,100,30);
        add(bk);
        bk.addActionListener(this);
        m[x]=current;
        x++;
        current++;
        set();
        if(current==9)
            b2.setText("Show Result");
        setVisible(false);
        setVisible(true);
    }
    for(int i=0,y=1;i<x;i++,y++)
    {
    if(e.getActionCommand().equals("Bookmark"+y))
    {
        if(check())
            count=count+1;
        now=current;
        current=m[y];
        set();
        ((JButton)e.getSource()).setEnabled(false);
        current=now;
    }
    }
```

```
        if(e.getActionCommand().equals("Show Result"))
        {
            if(check())
                count=count+1;
            current++;
            JOptionPane.showMessageDialog(this,"Correct Answers :"+count);
            System.exit(0);
        }
    }
    void set()
    {
        jb[4].setSelected(true);
        if(current==0)
        {
            l.setText("Q 1: Which one among these is not a primitive datatype?");
            jb[0].setText("int");jb[1].setText("Float");jb[2].setText("boolean");jb[3].setText("
char");
        }
        if(current==1)
        {
            l.setText("Q 2: Which class is available to all the class automatically?");
            jb[0].setText("Swing");jb[1].setText("Applet");jb[2].setText("Object");jb[3].setTe
xt("ActionEvent");
        }
        if(current==2)
        {
            l.setText("Q 3: Which package is directly available to our class without importing
it?");
            jb[0].setText("swing");jb[1].setText("applet");jb[2].setText("net");jb[3].setText("l
ang");
        }
        if(current==3)
        {
            l.setText("Q 4: String class is defined in which package?");
            jb[0].setText("lang");jb[1].setText("Swing");jb[2].setText("Applet");jb[3].setText
("awt");
        }
        if(current==4)
        {
            l.setText("Q 5: Which institute is best for java coaching?");
            jb[0].setText("Utek");jb[1].setText("Aptech");jb[2].setText("SSS
IT");jb[3].setText("jtek");
        }
        if(current==5)
        {
```

```
            l.setText("Q 6: Which one among these is not a keyword?");
            jb[0].setText("class");jb[1].setText("int");jb[2].setText("get");jb[3].setText("if");
        }
        if(current==6)
        {
            l.setText("Q 7: Which one among these is not a class? ");
            jb[0].setText("Swing");jb[1].setText("Actionperformed");jb[2].setText("ActionEv
ent");
                    jb[3].setText("Button");
        }
        if(current==7)
        {
            l.setText("Q 8: which one among these is not a function of Object class?");
            jb[0].setText("toString");jb[1].setText("finalize");jb[2].setText("equals");
                    jb[3].setText("getDocumentBase");
        }
        if(current==8)
        {
            l.setText("Q 9: which function is not present in Applet class?");
            jb[0].setText("init");jb[1].setText("main");jb[2].setText("start");jb[3].setText("des
troy");
        }
        if(current==9)
        {
            l.setText("Q 10: Which one among these is not a valid component?");
            jb[0].setText("JButton");jb[1].setText("JList");jb[2].setText("JButtonGroup");
                    jb[3].setText("JTextArea");
        }
        l.setBounds(30,40,450,20);
        for(int i=0,j=0;i<=90;i+=30,j++)
            jb[j].setBounds(50,80+i,200,20);
    }
    boolean check()
    {
        if(current==0)
            return(jb[1].isSelected());
        if(current==1)
            return(jb[2].isSelected());
        if(current==2)
            return(jb[3].isSelected());
        if(current==3)
            return(jb[0].isSelected());
        if(current==4)
            return(jb[2].isSelected());
        if(current==5)
```

```
            return(jb[2].isSelected());
        if(current==6)
            return(jb[1].isSelected());
        if(current==7)
            return(jb[3].isSelected());
        if(current==8)
            return(jb[1].isSelected());
        if(current==9)
            return(jb[2].isSelected());
        return false;
    }
    public static void main(String s[])
    {
        new OnlineTest("Online Core Java Quiz Application");
    }
}
```

Output:


Online Core Java Quiz Application
Q 10: Which one among these is not a valid component?
JButton
JList
JButtonGroup
JTextArea
Next
Show Result

Conclusion:

The given Java program is an implementation of an online quiz application for testing knowledge on core Java concepts. Here's a brief summary of the program's functionality:

1. The program creates a GUI-based quiz application using the Java Swing framework.
2. It defines a class **OnlineTest** that extends **JFrame** and implements the **ActionListener** interface.
2. The quiz consists of 10 multiple-choice questions, and each question has four options. Users can select one of the options as their answer.
4. The program maintains the user's score and current question number.
5. Users can navigate through the questions using the "Next" button.
6. Users can bookmark a question using the "Bookmark" button, which allows them to come back to it later.

- # References

1. **Java Programming Language:**
    - Oracle Java Documentation: Java SE Documentation
    - Java Tutorials: Oracle Java Tutorials
2. **User Interface (UI):**
    - JavaFX Documentation: JavaFX
    - JavaFX Tutorial: JavaFX Tutorial
3. **Backend Logic:**
    - Java Collections Framework: Java Collections
    - Multithreading in Java: Java Multithreading
4. **Database Integration:**
    - JDBC (Java Database Connectivity): JDBC Tutorial
    - Hibernate (Object-Relational Mapping for Java): Hibernate
5. **Web Development (Optional):**
    - Spring Framework: Spring Framework
    - Spring Boot (for quick application setup): Spring Boot
6. **Testing:**
    - JUnit: JUnit
    - TestNG: TestNG
7. **Version Control:**
    - Git Documentation: Git
8. **Build Tools:**
    - Apache Maven: Maven
    - Gradle: Gradle
9. **Security:**
    - Java Authentication and Authorization Service (JAAS): JAAS
10. **Additional Resources:**
    - Codecademy Java Course: Codecademy - Learn Java
    - Java Brains YouTube Channel: Java Brains