

QUIZ APPLICATION



A PROJECT REPORT

Submitted by

RAJAKARTHIKEYAN.V (2303811710421129)

in partial fulfillment of requirements for the award of the course

CGB1201 - JAVA PROGRAMMING

In

COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM – 621 112

NOVEMBER- 2024

SAMAYAPURAM – 621 112

Certified that this project report on “ **QUIZ APPLICATION**” is the bonafide work of **RAJAKARTHIKEYAN.V (2303811710421129)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

CGB1201-JAVA PROGRAMMING
Mr. MAHARAMAN A, M.E.,
ASSISTANT PROFESSOR

SIGNATURE

SUPERVISOR

ASSISTANT PROFESSOR

Department of CSE

K.Ramakrishnan College of Technology
(Autonomous)

Samayapuram-621112.

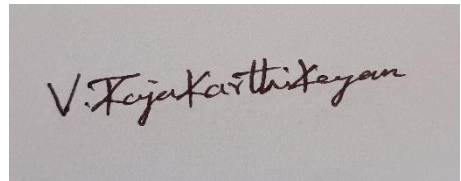
CGB1201-JAVA PROGRAMMING
Mr.R. KARTHICK, M.E.,
EXTERNAL EXAMINER
ASSISTANT PROFESSOR
8138-SCE, TRICHY.

EXTERNAL EXAMINER

DECLARATION

I declare that the project report on “**QUIZ APPLICATION**” is the result of original work done by us and best of our knowledge, similar work has not been submitted to “**ANNA UNIVERSITY CHENNAI**” for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **CGB1201 - JAVA PROGRAMMING**.

Signature:

A rectangular box containing a handwritten signature in black ink. The signature is written in a cursive style and reads "V. Rajakarthikeyan".

RAJAKARTHIKEYAN.V

Place: Samayapuram

Date: 06.12.2024

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution “**K.Ramakrishnan College of Technology (Autonomous)**”, for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN, B.E.**, for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.**, for forwarding to our project and offering adequate duration in completing our project.

I would like to thank **Dr. N. VASUDEVAN, M.Tech., Ph.D.**, Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. A. DELPHIN CAROLINA RANI, M.E.,Ph.D.**, Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project supervisor **MR. A. MALARMANNAN, M.E.**, Department of **COMPUTER SCIENCE AND ENGINEERING**, for his incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

VISION OF THE INSTITUTION

To serve the society by offering top-notch technical education on par with global standards

MISSION OF THE INSTITUTION

- Be a center of excellence for technical education in emerging technologies by exceeding the needs of the industry and society.
- Be an institute with world class research facilities
- Be an institute nurturing talent and enhancing the competency of students to transform them as all-round personality respecting moral and ethical values

VISION OF DEPARTMENT

To be a center of eminence in creating competent software professionals with research and innovative skills.

MISSION OF DEPARTMENT

M1: Industry Specific: To nurture students in working with various hardware and software platforms inclined with the best practices of industry.

M2: Research: To prepare students for research-oriented activities.

M3: Society: To empower students with the required skills to solve complex technological problems of society.

PROGRAM EDUCATIONAL OBJECTIVES

1. PEO1: Domain Knowledge

To produce graduates who have strong foundation of knowledge and skills in the field of Computer Science and Engineering.

2. PEO2: Employability Skills and Research

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

3. PEO3: Ethics and Values

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO 1: Domain Knowledge

To analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.

PSO 2: Quality Software

To apply software engineering principles and practices for developing quality software for scientific and business applications.

PSO 3: Innovation Ideas

To adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing/novel problems

PROGRAM OUTCOMES (POs)

Engineering students will be able to:

- 1. Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
- 2. Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences
- 3. Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations
- 4. Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions

5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

ABSTRACT

The Quiz Application is a user-friendly and interactive software designed to enhance learning and self-assessment. Built using Java's AWT and Swing libraries, the application allows users to attempt quizzes by selecting from various categories such as General Knowledge, Maths, and Movies. The application reads questions dynamically from an external text file, filters them based on the selected category, and presents them in a well-designed GUI. A robust scoring mechanism ensures that users receive real-time feedback on their performance. By integrating OOP concepts, file handling, and event-driven programming, the project provides a complete example of Java's versatility in application development.

ABSTRACT WITH POs AND PSOs MAPPING

CO 5 : BUILD JAVA APPLICATIONS FOR SOLVING REAL-TIME PROBLEMS.

ABSTRACT	POs MAPPED	PSOs MAPPED
<p>The Quiz Application is a user-friendly and interactive software designed to enhance learning and self-assessment. Built using Java's AWT and Swing libraries, the application allows users to attempt quizzes by selecting from various categories such as General Knowledge, Maths, and Movies. The application reads questions dynamically from an external text file, filters them based on the selected category, and presents them in a well-designed GUI. A robust scoring mechanism ensures that users receive real-time feedback on their performance. By integrating OOP concepts, file handling, and event-driven programming, the project provides a complete example of Java's versatility in application development.</p>	<p>PO1 -3 PO2 -3 PO3 -3 PO4 -3 PO5 -3 PO6 -3 PO7 -3 PO8 -3 PO9 -3 PO10 -3 PO11-3 PO12 -3</p>	<p>PSO1 -3 PSO2 -3 PSO3 -3</p>

Note: 1- Low, 2-Medium, 3- High

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	vii
1	INTRODUCTION	1
	1.1 Objective	1
	1.2 Overview	1
	1.3 Java Programming concepts	1
2	PROJECT METHODOLOGY	3
	2.1 Proposed Work	3
	2.2 Block Diagram	3
3	MODULE DESCRIPTION	4
	3.1 Question Loading	4
	3.2 User Interface	4
	3.3 Randomization and Selection	4
	3.4 Score Calculation	5
	3.5 Result Display	5
4	CONCLUSION & FUTURE SCOPE	6
	4.1 Conclusion	6
	4.2 Future Scope	6
	APPENDIX A (SOURCE CODE)	7
	APPENDIX B (SCREENSHOTS)	11
	REFERENCES	13

CHAPTER 1

INTRODUCTION

1.1 Objective

The primary aim of this project is to design and develop a quiz application that provides an interactive and educational experience. Users can select a category, answer a series of questions, and receive a score at the end, encouraging learning and self-assessment.

1.2 Overview

It is a Java-based GUI quiz application designed to allow users to select a category, answer questions, and view their scores at the end of the quiz. It reads questions from an external file, filters them based on the selected category, and shuffles them to ensure a random selection of questions for each session. The application uses various Java GUI components like Frame, Label, Checkbox, and Button to create an interactive and user-friendly interface. Events such as button clicks are handled using the ActionListener interface to ensure smooth navigation through the quiz. The program is modular, with distinct functionality for loading questions, displaying them, validating answers, and calculating the score. At the end of the quiz, the user is presented with their final score and performance summary. The code effectively demonstrates core Java concepts, including file handling with BufferedReader, collections with ArrayList, event-driven programming, and object-oriented principles, making it a comprehensive example of a small-scale Java application.

1.3 Java Programming Concepts

1. Object-Oriented Programming (OOP):

Java is built around OOP principles, including encapsulation, inheritance, polymorphism, and abstraction. These concepts promote modular design, code reusability, and easier maintenance of applications.

2. Exception Handling:

Java provides mechanisms to handle runtime errors gracefully using try, catch, and finally. It ensures the program continues executing smoothly without abrupt terminations, even when exceptions occur.

3. Collections Framework:

This framework provides ready-to-use data structures like ArrayList, HashMap, and LinkedList. It helps in efficiently storing, retrieving, and manipulating groups of objects, making it essential for data management.

4. Multithreading:

Java supports concurrent execution of multiple threads, allowing tasks to run in parallel. This improves the efficiency and responsiveness of applications, especially in multitasking environments.

5. Java Swing/GUI:

Java offers the Swing library to create graphical user interfaces with components like Button, Label, and TextField. It simplifies building interactive and visually appealing desktop applications.

6. Generics:

Generics ensure type safety and reusability in collections and methods. They reduce runtime errors by enforcing type checks during compilation.

7. File Handling:

Java facilitates reading and writing data to files using classes like FileReader, BufferedReader, and FileWriter, essential for managing external data storage and retrieval.

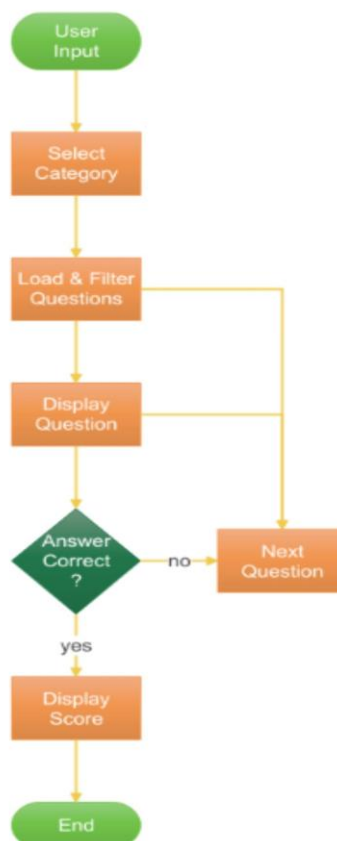
CHAPTER 2

PROJECT METHODOLOGY

2.1 Proposed Work

The proposed work for the quiz application involves several key enhancements aimed at improving its functionality and user experience. First, integrating a timer for each question would make the quiz more challenging and time-bound, encouraging quicker decision-making. Additionally, enabling dynamic loading of question files from various sources, such as user uploads or databases, would make the app more flexible and adaptable. Expanding the quiz to support more categories and a larger pool of questions would increase the variety of the quiz and keep it engaging for users over multiple sessions. Another important addition would be the implementation of a user profile system, allowing users to track their performance and scores over time, providing a personalized experience. Finally, upgrading the GUI using JavaFX would enhance the visual appeal and interactivity of the application, making it more modern and intuitive for users to navigate. These improvements would contribute to a more comprehensive and user-friendly quiz application.

2.2 Block Diagram



CHAPTER 3

MODULE DESCRIPTION

3.1 Module 1: Question Loading

The Question Loading module is responsible for reading and processing the quiz questions from an external file, typically formatted with questions, options, and the correct answer. The questions are categorized based on a predefined set of categories like General Knowledge, Maths, etc. The module filters the questions according to the selected category, ensuring that only relevant questions are loaded for that quiz session. Each question, along with its answer options and correct answer, is stored in an object of the Question class, allowing the program to easily manage and access them. This module ensures that all questions are ready and structured for the quiz.

3.2 Module 2: User Interface

The User Interface (UI) module is a critical component of the quiz application, as it handles all graphical elements and user interactions. It uses Java Swing components such as Label, Checkbox, and Button to display the questions, answer options, and navigation buttons. The module is responsible for the layout, ensuring that the quiz is easy to read and navigate. It updates dynamically with each question and answer option, allowing users to select their answers and proceed through the quiz. In addition, it handles user actions such as clicking the "Next" button to move to the next question. The UI module aims to provide an intuitive, engaging, and smooth interaction experience for the user.

3.3 Module 3: Randomization and Selection

The Randomization and Selection module ensures that each quiz session is unique and engaging by randomly selecting and presenting a different set of questions. Once all questions are loaded into the system, the module shuffles them to prevent a predictable order. It then picks a subset of questions (e.g., five questions) from the shuffled list, ensuring that the quiz is different each time a user takes it. This randomization helps keep the experience fresh and prevents users from memorizing questions. Additionally, this module ensures that questions are selected without repetition, creating a fair quiz environment for the user.

3.4 Module 4: Score Calculation

The Score Calculation module tracks the user's responses to each question and calculates the total score based on correct answers. As the user progresses through the quiz, the module continuously checks the selected answer against the correct answer for each question. If the answer is correct, the score is incremented. This module ensures that the score is updated in real-time, providing immediate feedback to the user on their performance. Once all questions are answered, the module calculates the final score, which reflects the user's performance throughout the quiz, allowing for an accurate assessment of their knowledge in the chosen category.

3.5 Module 5: Result Display

The Result Display module is responsible for showing the final score to the user once the quiz is completed. After the user answers all the questions, the module hides the question options and the navigation buttons and displays a message indicating that the quiz has ended. The user's total score is shown, along with a performance summary that may include the number of correct and incorrect answers. This module offers feedback to the user, allowing them to reflect on their performance. Additionally, it may provide options for the user to start a new quiz or exit the application. The Result Display module concludes the quiz session by providing the user with their final results in a clear and concise manner.

CHAPTER 4

CONCLUSION

The Quiz Application successfully integrates the principles of Java programming to create an engaging and educational platform. The use of dynamic question loading, category-based filtering, and real-time score tracking demonstrates the project's practical utility. This project serves as an excellent example of Java's capabilities for developing desktop-based educational tools while adhering to modular and scalable design principles.

FUTURE SCOPE:

The future scope for a quiz application is promising. It includes AI-based personalized quizzes, AR/VR integration, and gamification features like leaderboards and rewards. Multilingual support ensures accessibility, while offline mode enhances usability. The app can serve corporate training and education, with adaptive difficulty for better engagement. Data analytics and social sharing features can boost user insights and community growth.

APPENDIX - A

(Project Source Code)

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

class Question {
    String category;
    String questionText;
    String[] options;
    int correctAnswer;

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

    public boolean isCorrect(int answer) {
        return answer == correctAnswer;
    }
}

public class QuizApplication extends Frame implements ActionListener {
    ArrayList<Question> questions = new ArrayList<>();
    ArrayList<Question> selectedQuestions = new ArrayList<>();
    int currentQuestionIndex = 0;
    int score = 0;

    Label questionLabel;
    CheckboxGroup optionsGroup;
    Checkbox option1, option2, option3, option4;
    Button nextButton;
    Label scoreLabel;
    String selectedCategory;

    public QuizApplication() {
        setLayout(new FlowLayout());

        questionLabel = new Label("", Label.CENTER);
        add(questionLabel);

        optionsGroup = new CheckboxGroup();
        option1 = new Checkbox("", optionsGroup, false);
        option2 = new Checkbox("", optionsGroup, false);
```

```

option3 = new Checkbox("", optionsGroup, false);
option4 = new Checkbox("", optionsGroup, false);

add(option1);
add(option2);
add(option3);
add(option4);

nextButton = new Button("Next");
nextButton.addActionListener(this);
add(nextButton);

scoreLabel = new Label("");
add(scoreLabel);

setTitle("Quiz Application");
setSize(400, 300);
setVisible(true);
}

public void loadQuestions(String fileName, String category) {
    try (BufferedReader br = new BufferedReader(new FileReader(fileName))) {
        String line;
        while ((line = br.readLine()) != null) {
            String[] parts = line.split("\\|");
            if (parts.length != 7) continue;

            String fileCategory = parts[0];
            String questionText = parts[1];
            String[] options = Arrays.copyOfRange(parts, 2, 6);
            int correctAnswer = Integer.parseInt(parts[6]);

            if (fileCategory.equalsIgnoreCase(category)) {
                questions.add(new Question(fileCategory, questionText, options,
correctAnswer));
            }
        }
    } catch (IOException e) {
        System.out.println("Error reading file: " + e.getMessage());
    }
}

public void startQuiz(String category) {
    selectedCategory = category;

    // Shuffle and pick 5 questions
    Collections.shuffle(questions);
    selectedQuestions = new ArrayList<>(questions.subList(0, Math.min(5,
questions.size())));

```

```

        loadQuestion();
    }

    private void loadQuestion() {
        if (currentQuestionIndex >= selectedQuestions.size()) {
            displayScore();
            return;
        }

        Question currentQuestion = selectedQuestions.get(currentQuestionIndex);

        questionLabel.setText(currentQuestion.questionText);
        option1.setLabel(currentQuestion.options[0]);
        option2.setLabel(currentQuestion.options[1]);
        option3.setLabel(currentQuestion.options[2]);
        option4.setLabel(currentQuestion.options[3]);
    }

    private void displayScore() {
        questionLabel.setText("Quiz Completed!");
        option1.setVisible(false);
        option2.setVisible(false);
        option3.setVisible(false);
        option4.setVisible(false);
        nextButton.setVisible(false);

        scoreLabel.setText("Your score: " + score + "/" + selectedQuestions.size());
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        if (currentQuestionIndex < selectedQuestions.size()) {
            Question currentQuestion = selectedQuestions.get(currentQuestionIndex);

            // Check the selected answer
            int selectedOption = -1;
            if (option1.getState()) selectedOption = 1;
            if (option2.getState()) selectedOption = 2;
            if (option3.getState()) selectedOption = 3;
            if (option4.getState()) selectedOption = 4;

            if (currentQuestion.isCorrect(selectedOption)) {
                score++;
            }

            currentQuestionIndex++;
            loadQuestion();
        }
    }
}

```

```

public static void main(String[] args) {
    QuizApplication quizApp = new QuizApplication();

    // Choose a category
    String[] categories = {"General Knowledge", "Maths", "Games", "Movies", "Bikes"};
    String selectedCategory = (String) JOptionPane.showInputDialog(
        null,
        "Choose a category:",
        "Quiz Categories",
        JOptionPane.QUESTION_MESSAGE,
        null,
        categories,
        categories[0]);

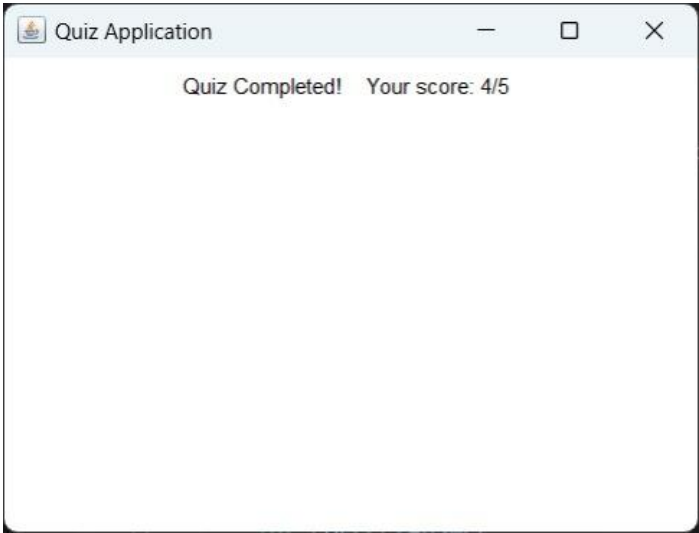
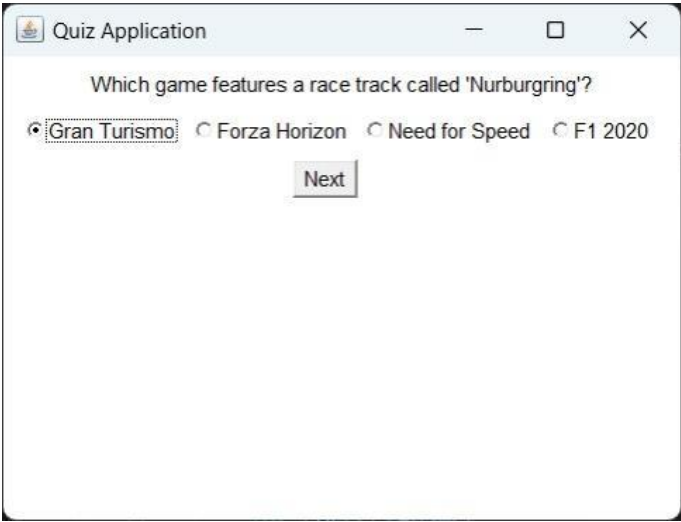
    if (selectedCategory == null || selectedCategory.isEmpty()) {
        System.exit(0);
    }

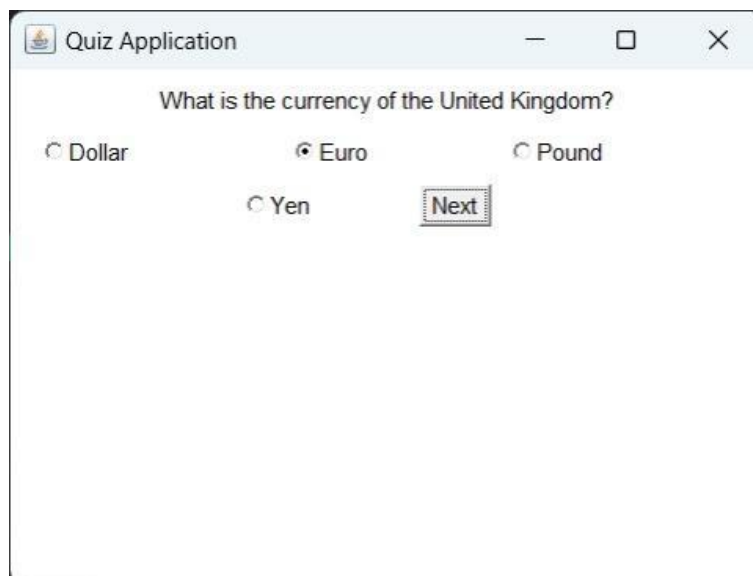
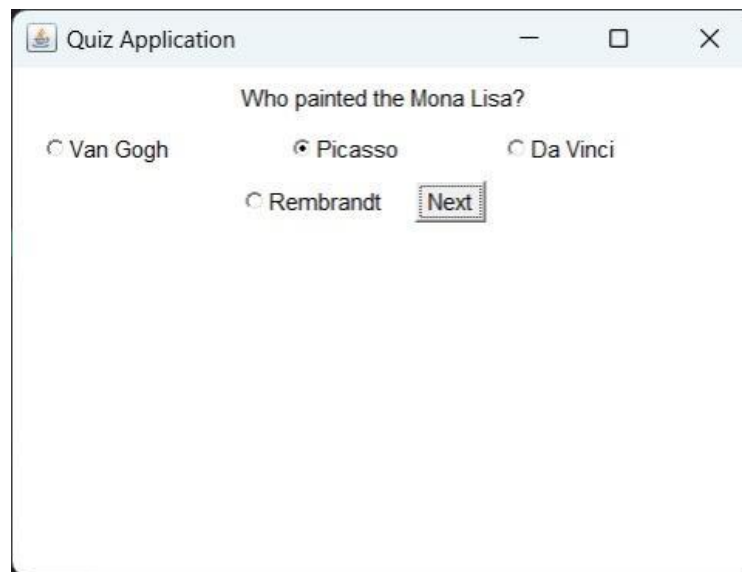
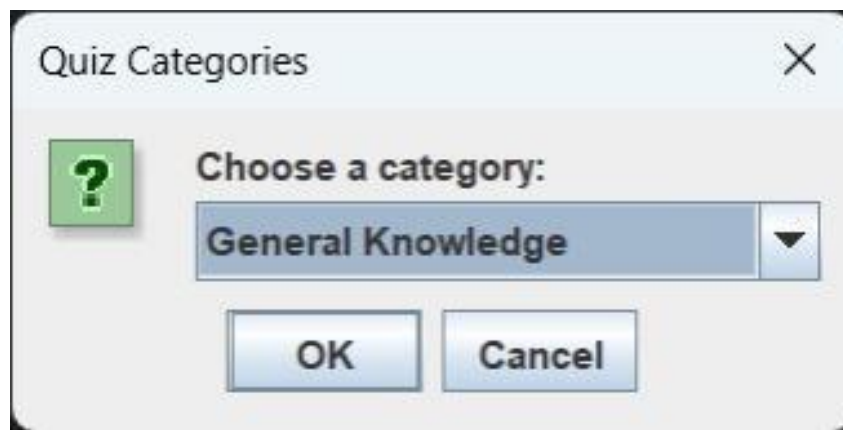
    quizApp.loadQuestions("questions.txt", selectedCategory);
    quizApp.startQuiz(selectedCategory);
}
}

```

APPENDIX - B SCREEN SHOTS

QUIZ APPLICATION





REFERENCES

1. Herbert Schildt, Java: The Complete Reference, McGraw Hill Education, 11th Edition 2019.
2. Cay S. Horstmann, Core Java Volume I – Fundamentals, Pearson Education, 11th Edition, 2018.
3. Oracle Java Documentation: <https://docs.oracle.com/en/java>
4. GeeksforGeeks Java Tutorials: <https://www.geeksforgeeks.org/java>
5. Telusko – Java Programming Tutorials
6. Programming with Mosh – Java for Beginners