INTERNSHIP PROJECT-1 DOCUMENT
ON "EXPENSE TRACKER APPLICATION (JAVA)"

Submitted by:	**Submitted To:-**
Navpreet Singh (INTERN)	**Kanduri Abhinay (FOUNDER)**
	RITHIN VERMA (CTO)

INDEX

INTRODUCTION:-

The Online Quiz System is a Java-based graphical application developed to provide users with an interactive platform for answering multiple-choice questions (MCQs). This project uses Java's Swing library to create a user-friendly graphical interface, calculates quiz scores in real time, and persistently stores user results for future reference. The solution demonstrates the integration of user input, automated scoring, and simple file handling techniques, making it ideal for educational and self-assessment purposes.

SOFTWARE REQUIREMENTS:-

Programming Language: Java (JDK 8 or higher)

IDE: VS Code, IntelliJ IDEA, Eclipse, or any Java-compatible editor

Libraries: Java Standard Library (especially javax.swing and java.io)

Operating System: Cross-platform (Windows, Linux, Mac)

Other Requirements: No external dependencies; runs as a standalone application with GUI.

DESIGN:-

The system is organized into the following core components:

Question Bank: Stores MCQ text, possible options, and correct answers in program arrays or lists.

QuizApp Main Class: Handles GUI construction, quiz logic, user navigation, and score tracking.

ResultStorage Class: Responsible for saving quiz results (user name and score) to a text file for persistence.

User Interface: Built using Swing components (JLabel, JRadioButton, JButton, JPanel) for smooth, interactive user experience.

All navigation and actions are performed via buttons and dialog boxes in the graphical window.

INPUT:-

```
+---------------------------+

|    Online Quiz System    |

+---------------------------+
|  [Question displayed]    |
|  ( ) Option A            |
|  ( ) Option B            |
|  ( ) Option C            |
|  ( ) Option D            |
|  [Next]     [Submit]     |
+--------------------------+
```

The user selects answers using radio buttons for each displayed question.

Inputs are processed through button clicks ("Next" to proceed, "Submit" to finish).

IMPLEMENTATION:-

1. The application maintains arrays/lists for questions, answer choices, and correct response indices.

2. On startup, the QuizApp class initializes the graphical interface and displays the first question.

3. Each selected answer is recorded as the user progresses.

4. The score is automatically calculated by comparing the user's choice to the correct answer for each question.

5. Upon completion, the application displays the final score, shows correct answers for review, and prompts for the user's name.

6. User scores (with name and score) are saved to a local text file (results.txt) using file handling methods in the ResultStorage class.

7. The application gracefully exits after displaying results and storing the outcome.

TESTING:-

The following testing methodology was applied:

Manual Functional Testing: Used a variety of answer selections to ensure correct score calculation and answer review.

Result Storage Validation: Verified that each quiz attempt appends a new result (user name and score) to the results.txt file.

Boundary Cases: Checked behavior when no option is selected, at last question, and when entering empty or unusual user names.

Cross-Platform Runs: Tested on Windows and Linux for consistent GUI and file interactions.

Error Handling: Confirmed robust handling when the results file is missing or cannot be written.

ADVANTAGES:-

Intuitive GUI: User-friendly interface for all experience levels.

Automated Scoring: Instant feedback and score calculation upon quiz completion.

Result Persistence: Stores user names and scores for historical tracking or review.

Customization: Easily add or modify question sets as needed.

No External Libraries: Runs on any basic Java installation without additional setup.

Educational Value: Facilitates practice, learning, and assessment in a standalone package.

CONCLUSION:-

The Online Quiz System project demonstrates the practical merging of Java GUI programming, interactive user logic, and persistent data storage. The application provides a full workflow from question delivery to result feedback and storage. Through its clear interface and robust back-end, the system serves as a strong example of how Java can be used to build real-world educational tools. It is highly adaptable for further extension, such as supporting user log-in, category selection, or web-based deployment.

REFERENCES:-

GeeksForGeeks: Java Swing Tutorials and GUI Programming
(https://www.geeksforgeeks.org/java-swing-tutorial/)

JavaTPoint: Java File Handling and GUI
(https://www.javatpoint.com/java-file-handling)
(https://www.javatpoint.com/java-swing)

Oracle Documentation: Standard Edition – Swing and I/O documentation
(https://docs.oracle.com/en/java/javase/)