Software Requirements Specification (SRS)

Online Quiz Application

Introduction

1.1 Purpose

This document describes the software requirements for the Online Quiz Application.
It is intended for developers, testers, instructors, and evaluators to understand the system's functionality, constraints, and design goals.

1.2 Scope

The Online Quiz Application is a Python-based system that allows instructors to create quiz questions and allows students to take quizzes through both a Command Line Interface (CLI) and a Graphical User Interface (GUI).
The system supports quiz reporting, analytics tracking, and exporting results to CSV format.

1.3 Definitions, Acronyms, and Abbreviations

- **CLI**: Command Line Interface
- **GUI**: Graphical User Interface
- **TA**: Teaching Assistant
- **SRS**: Software Requirements Specification

---

2. Overall Description

2.1 Product Perspective

The application is a standalone desktop-based system developed using Python.
It follows a modular architecture where question management, quiz execution, reporting, and analytics are separated into independent components.

2.2 User Classes and Characteristics

- **Instructor / Teaching Assistant**
    - Adds quiz questions
    - Assigns categories to questions
- **Student**
    - Takes quizzes
    - Views quiz results
    - Views analytics statistics

2.3 Operating Environment

- Programming Language: Python 3.x
- GUI Framework: Tkinter
- Platform: Windows / Linux / macOS
- Storage: In-memory data and CSV files

2.4 Design and Implementation Constraints

- The system runs locally and does not use a database.
- Data persistence is limited to runtime memory and CSV export.
- The GUI is implemented using Tkinter.

---

3. Functional Requirements

FR1 – Add Questions

The system shall allow instructors to add multiple-choice quiz questions.

FR2 – Assign Question Categories

The system shall allow instructors to assign a category to each question.

FR3 – Select Quiz Category

The system shall allow students to select a quiz category before starting the quiz.

FR4 – Take Quiz

The system shall allow students to take quizzes through both GUI and CLI interfaces.

FR5 – Calculate Score

The system shall automatically calculate the quiz score based on correct answers.

FR6 – Generate Quiz Report

The system shall generate a quiz report displaying score and percentage.

FR7 – Track Quiz Analytics

The system shall track quiz attempts and store scores for analytics purposes.

FR8 – View Analytics Dashboard

The system shall allow users to view analytics statistics such as number of attempts and average score.

FR9 – Export Analytics Data

The system shall allow users to export analytics data to a CSV file.

---

4. Non-Functional Requirements

4.1 Usability

- The system shall provide a simple and intuitive GUI.
- Menu options shall be clearly labeled.

4.2 Performance

- Quiz responses shall be processed instantly.
- The system shall handle multiple quiz attempts during runtime.

4.3 Reliability

- The system shall not crash due to invalid user input.
- Input validation shall be applied where necessary.

4.4 Maintainability

- The system shall follow modular design principles.
- Code shall be readable and well-documented.

---

5. Assumptions and Dependencies

- Users have Python installed on their machines.
- The application runs in a local environment.
- No internet connection is required during runtime.

---

6. Future Enhancements

- Database integration for persistent storage
- User authentication system
- Timed quizzes
- Question randomization by difficulty

Developers

- Shady
- Abdallah

- Yassin