Software Requirements Specification

For

Quiz App

Version 1.0 approved

Prepared by

23DCS138 (Vasu Vaghasia)

Charotar University of Science and Technology

25th August , 2025

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) is to provide a detailed description of the requirements for the "Placement Preparation Quiz App." This application is a mobile-based platform developed using Flutter and Dart, designed to help students prepare for aptitude and technical rounds of placement interviews. This document will serve as the foundation for the project by outlining the system's features, functionalities, and constraints for reference by developers and evaluators.

1.2 Document Conventions

- IEEE 830-1998 standard for SRS
- The terms "app," "system," and "platform" are used interchangeably.
- Date format: DD-MM-YYYY

1.3 Intended Audience and Reading Suggestions

- **Developers :** To understand the technical requirements, features, and constraints to build the application.
- **Project Manager :** To oversee the project scope, timeline, and resource allocation.
- **Testers :** To create test cases and ensure the application meets the specified requirements.
- **Client/Stakeholders :** To review and confirm that the proposed system aligns with their vision and goals.

1.4 Product Scope

The Placement Preparation Quiz App is a comprehensive tool for students targeting company placements. It provides a platform for practicing aptitude and technical questions through various quizzes.

Key Features:

- **Admin Panel:** Allows administrators to create, manage, and monitor quizzes and student performance.
- **Student Panel:** Enables students to register, join quizzes, track their progress, and compete on a leaderboard.
- **Quiz Creation Methods:** Admins can add questions manually or upload them in bulk using an Excel sheet.
- **Quiz Types:** Includes quizzes created by admins that can be joined via a code, as well as built-in practice quizzes for general aptitude and technical skills.
- **Performance Tracking:** Students can view their scores from past quizzes to monitor their improvement.
- **Gamification:** A scoring system and a leaderboard to motivate students and foster healthy competition.

Technologies:

- **Frontend:** Flutter & Dart
- Backend: Firebase (or a similar BaaS)
- Database: Firestore/Realtime DatabaseOverall Description

2. Overall Description

2.1 Product Perspective

The application will be a self-contained, client-server system. The frontend will be a cross-platform mobile application for Android and iOS. The backend will be a cloud-based service responsible for user authentication, data storage (quizzes, questions, results), and business logic.

2.2 Product Functions

The major functions of the application are:

- **User Authentication:** Secure registration and login for both admins and students.
- **Quiz Management:** Creation, modification, and deletion of quizzes and questions.
- **Quiz Participation:** A seamless interface for students to take quizzes.
- **Real-time Results:** Instant feedback and score calculation upon quiz completion.
- **Historical Data:** Storing and retrieving past performance data for students.
- **Leaderboard:** Displaying the top-performing students based on their cumulative scores.

2.3 User Classes and Characteristics

There are two primary user classes:

1. **Administrator (Admin):**

- o **Characteristics:** A technically proficient user responsible for content management. Has full control over the quizzes and user data.
- o **Responsibilities:**
- ▪ Creating and managing quizzes.
- ▪ Adding questions manually or via Excel upload.
- ▪ Viewing the results of all students for any given quiz.
- ▪ Managing the application's content.

2. **Student:**

- o **Characteristics:** The end-user of the application, typically a college student preparing for placements.

- **Responsibilities:**
  - Registering and logging into the app.
  - Joining specific quizzes using a unique code.
  - Attempting built-in practice quizzes.
  - Viewing their own quiz scores and history.
  - Viewing their rank and score on the global leaderboard.

2.4 Operating Environment

- **Mobile Application:** The app will be compatible with:
  - Android 6.0 (Marshmallow) and above.
  - iOS 12.0 and above.
- **Backend:** The backend will be hosted on a cloud platform (e.g., Firebase) and will not have specific OS dependencies for the end-user.
- **Network:** A stable internet connection is required for all functionalities.

2.5 Design and Implementation Constraints

- The mobile application must be developed using the Flutter framework and Dart programming language.
- The backend **must** use a scalable cloud-based solution like Firebase to handle real-time data synchronization.
- The system must rely on a third-party library for parsing Excel files.
- The user interface should be intuitive and follow modern mobile UI/UX design principles.

2.6 Assumptions and Dependencies

- Users (both Admins and Students) have access to a smartphone with a reliable internet connection.
- The format of the Excel sheet for bulk question upload will be predefined and fixed.
- Third-party services (like Firebase authentication, database) will be available and reliable.

3. External Interface Requirements

3.1 User Interfaces

The application will feature a clean, modern, and user-friendly interface. Key screens will include:

- **Splash Screen:** App branding on launch.
- **Login/Registration Screen:** For user authentication.
- **Admin Dashboard:** Main screen for admins with options to create quizzes, view results, etc.
- **Student Dashboard:** Main screen for students to join a quiz, access practice quizzes, and view the leaderboard.
- **Quiz Interface:** Screen for attempting the quiz with questions, options, a timer, and navigation buttons.
- **Results Screen:** Displays the score, correct/incorrect answers immediately after quiz completion.
- **Past Results Screen:** A list of previously attempted quizzes and the scores obtained.

Leaderboard Screen: Shows the top 10 students and the current user's rank.

3.2 Hardware Interfaces

No special hardware interfaces are required. The application will use standard smartphone components like the screen and network adapter.

3.3 Software Interfaces

- **Database:** A NoSQL database like Firebase Firestore will be used for data storage.
- **Authentication:** Firebase Authentication for handling user sign-up and sign-in.
- **Excel Parsing Library:** A Dart/Flutter package (e.g., excel) will be used to read data from .xlsx files.

3.4 Communications Interfaces

- The mobile application will communicate with the backend server via secure RESTful APIs or a direct WebSocket connection (if using Firebase).
- Data will be exchanged in JSON format.
- All communication will be encrypted using HTTPS/SSL.

4. System Features

4.1 Admin Management

- **4.1.1. Admin Authentication:**
  - **Description:** The admin must be able to log in securely to access the admin panel.
  - **Functional Requirements:**
    - FR1: The system shall provide a secure login interface for admins.
    - FR2: The system shall validate admin credentials against the database.

- **4.1.2. Quiz Creation:**
  - **Description:** The admin can create new quizzes.
  - **Functional Requirements:**
    - FR3: The system shall allow the admin to create a quiz by specifying a title, duration, and generating a unique join code.
    - FR4: The system shall allow the admin to add questions to a quiz manually, providing the question text, multiple-choice options, and specifying the correct answer.
    - FR5: The system shall provide an option to upload an Excel file (.xlsx) containing questions, options, and the correct answer in a predefined format.
    - FR6: The system shall parse the uploaded Excel file and populate the quiz with the questions.

- **4.1.3. Result Viewing:**
  - **Description:** The admin can view the performance of students for any quiz.
  - **Functional Requirements:**
    - FR7: The system shall display a list of all quizzes created by the admin.
    - FR8: Upon selecting a quiz, the system shall display a list of all students who attempted it, along with their scores.

4.2 Student Management

- **4.2.1. Student Authentication:**

- **Description:** Students must be able to register and log in to the application.
- **Functional Requirements:**
  - FR9: The system shall provide an interface for new students to register with a username, email, and password.
  - FR10: The system shall provide a secure login interface for existing students.

4.3 Quiz Management

- **4.3.1. Join Quiz:**
  - **Description:** Students can join a specific quiz created by an admin.
  - **Functional Requirements:**
    - FR11: The system shall provide an option for students to enter a unique quiz code.
    - FR12: The system shall validate the code and allow the student to start the quiz.

- **4.3.2. Attempt Quiz:**
  - **Description:** Students can attempt quizzes in a timed environment.
  - **Functional Requirements:**
    - FR13: The system shall display one question at a time with multiple-choice options.
    - FR14: The system shall display a countdown timer for the quiz duration.
    - FR15: The system shall automatically submit the quiz when the timer runs out.
    - FR16: The system shall allow students to navigate between questions (if permitted by the quiz settings).

- **4.3.3. Built-in Quizzes:**
  - **Description:** Students can take pre-loaded quizzes for practice.
  - **Functional Requirements:**
    - FR17: The system shall provide a section with built-in quizzes for aptitude and technical topics.
    - FR18: Students can attempt these quizzes anytime without a code.

4.4 Result and Scoring Management

- **4.4.1. View Instant Results:**
  - **Description:** Students can see their results immediately after finishing a quiz.
  - **Functional Requirements:**
    - FR19: Upon quiz submission, the system shall calculate and display the student's score.
    - FR20: The system shall provide a summary of correct and incorrect answers.

- **4.4.2. View Past Results:**
  - **Description:** Students can track their performance over time.
  - **Functional Requirements:**
    - FR21: The system shall maintain a history of all quizzes attempted by the student.
    - FR22: The system shall display a list of past quizzes with the scores obtained in each.

- **4.4.3. Scoreboard:**
  - **Description:** A leaderboard to foster competition among students.
  - **Functional Requirements:**
    - FR23: The system shall calculate a cumulative score for each student based on their quiz performances.
    - FR24: The system shall display a leaderboard with the top 10 students based on their cumulative scores.
    - FR25: The system shall display the current user's rank and score on the leaderboard.

5. Other Nonfunctional Requirements

- **5.1. Performance Requirements**
- The application should load within 3 seconds on a stable 4G connection.
- Quiz questions should load instantaneously.
- The backend should support at least 100 concurrent users without significant degradation in performance.

- **5.2. Security Requirements**
- All user passwords must be hashed and salted before being stored in the database.
- All data transmission between the client and server must be encrypted using SSL/TLS.
- The system should be protected against basic security threats like SQL injection and Cross-Site Scripting (XSS).
- Admins should only be able to access admin-specific functionalities.

- **5.3. Usability Requirements**
- The user interface should be intuitive and easy to navigate for non-technical users.
- The application should provide clear feedback to users for actions (e.g., successful login, quiz submission).
- The design should be responsive and adapt to different screen sizes and orientations.

- **5.4. Maintainability**
- The source code should be well-documented, modular, and follow standard coding conventions to facilitate future updates and maintenance.

- **5.5. Portability**
- As the application is being developed using Flutter, it must be fully functional on both Android and iOS platforms from a single codebase.