DEV PLUS

Software Requirements Specification (SRS)

1. Introduction

1.1 Purpose

The Online Mock Test System is designed to provide students with a platform to take mock tests for Reading, Listening, Writing, and Speaking skills. It aims to help students assess their proficiency, track progress, and improve their performance through structured testing and feedback mechanisms.

1.2 Scope

This application serves students, teachers, and administrators by facilitating online test-taking, grading, and performance tracking. Key features include:

- User authentication and role-based access
- Mock test management and scheduling
- Automated and manual grading
- Test result notifications and reporting
- Secure data handling and user management

1.3 Definitions, Acronyms, and Abbreviations

- **OTP:** One-Time Password
- **LMS:** Learning Management System
- **GDPR:** General Data Protection Regulation
- **Session Key:** A unique key required to access a test session

1.4 Overview

This document provides a structured breakdown of the Online Mock Test System's requirements, including functional and non-functional aspects, external interfaces, system models, and validation procedures.

---

2. Overall Description

2.1 Product Perspective

The Online Mock Test System is a standalone web-based platform that supports remote test-taking. It may be integrated with LMS platforms for extended functionality in the future.

2.2 Product Functions

- User authentication and role-based access control
- Test creation and management
- Automated grading (Reading & Listening) and manual grading (Writing & Speaking)
- Student test history tracking
- Performance reports and notifications

2.3 User Characteristics

- **Students:** Non-technical users who take tests and track performance.
- **Teachers:** Users responsible for managing test sessions and grading exams.
- **Administrators:** Users handling teacher accounts and system configuration.

2.4 Constraints

- Supports up to 10,000 concurrent users.
- Requires internet access and a web browser.
- Complies with GDPR for data security and privacy.

2.5 Assumptions and Dependencies

- Assumes students have a valid session key to access tests.
- Assumes teachers will manually grade Writing and Speaking tests.
- Depends on email service providers for OTP and result notifications.

---

3. Functional Requirements

3.1 User Authentication

- **FR-1:** Students must register with Name, Email, Password, Phone Number, Student ID, and Class Name.
- **FR-2:** OTP verification is required for account activation.
- **FR-3:** Users must be able to log in with email and password.

- **FR-4:** Password reset functionality via email must be provided.

3.2 Mock Test Features

- **FR-5:** Students can view available test sets.
- **FR-6:** Students must enter a valid session key to start a test.
- **FR-7:** The system must display test instructions before starting.
- **FR-8:** The system must track test progress and remaining time.
- **FR-9:** Test responses must be saved automatically.
- **FR-10:** Test submissions must be confirmed before finalization.
- **FR-11:** Teachers can create test sessions with a session key, valid duration, and max attempts.

3.3 Grading & Results

- **FR-12:** The system auto-grades multiple-choice tests.
- **FR-13:** Teachers manually grade Writing and Speaking tests.
- **FR-14:** The system sends test results via email upon grading completion.
- **FR-15:** Students can view past test scores in their profile.

3.4 Security & Access Control

- **FR-16:** Role-based access control must be enforced.
- **FR-17:** Only administrators can manage teacher accounts.
- **FR-18:** Only teachers can create and manage test sessions.

---

4. Non-Functional Requirements

4.1 Performance Requirements

- **NFR-1:** The system must support up to 10,000 concurrent users.
- **NFR-2:** Test pages must load within 3 seconds.

4.2 Security Requirements

- **NFR-3:** User data must be encrypted at rest and in transit.
- **NFR-4:** The system must implement multi-factor authentication for admin accounts.

4.3 Usability Requirements

- **NFR-5:** The system must be accessible via desktop and mobile devices.
- **NFR-6:** The user interface must support English language input.

4.4 Reliability Requirements

- **NFR-7:** The system must maintain 99.9% uptime.
- **NFR-8:** Automated backups must be performed daily.

4.5 Maintainability and Support Requirements

- **NFR-9:** The system must support software updates without downtime.
- **NFR-10:** Detailed logs must be maintained for debugging and auditing.

---

5. External Interface Requirements

5.1 User Interfaces

- The system must provide a responsive web-based UI.
- The interface must be intuitive for non-technical users.

5.2 Hardware Interfaces

- The system requires standard computing devices (PC, tablet, smartphone) with internet access.

5.3 Software Interfaces

- The system should support integration with email services for notifications.

5.4 Communication Interfaces

- The system must use HTTPS for secure data transmission.

---

6. System Models

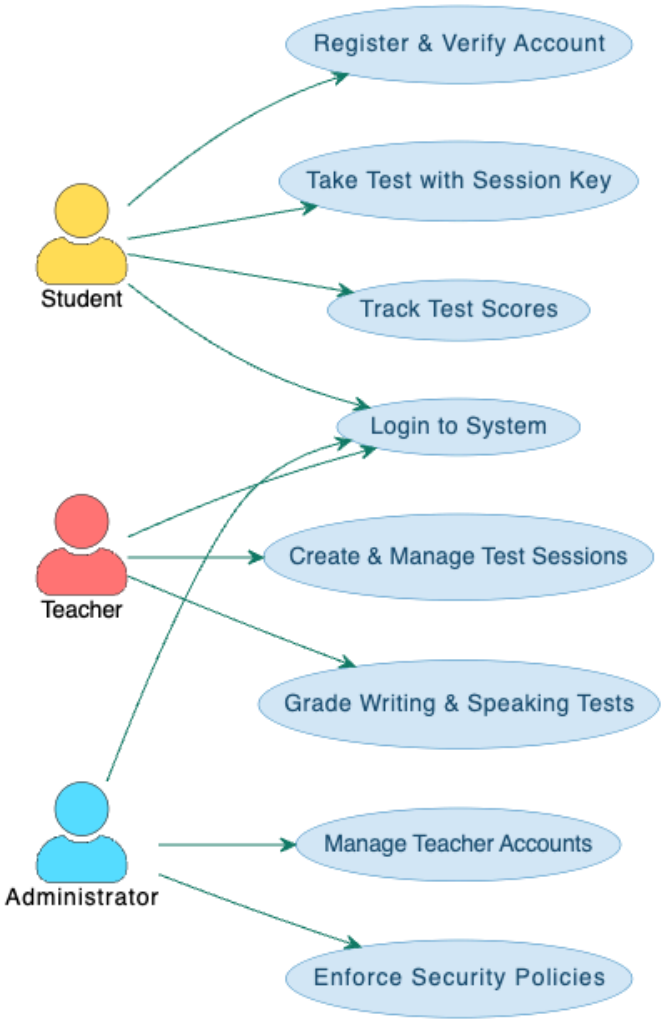
6.1 User Stories

- **As a student,** I want to register and verify my account so that I can access the mock test system.
- **As a student,** I want to log in securely so that I can access my test history and upcoming tests.

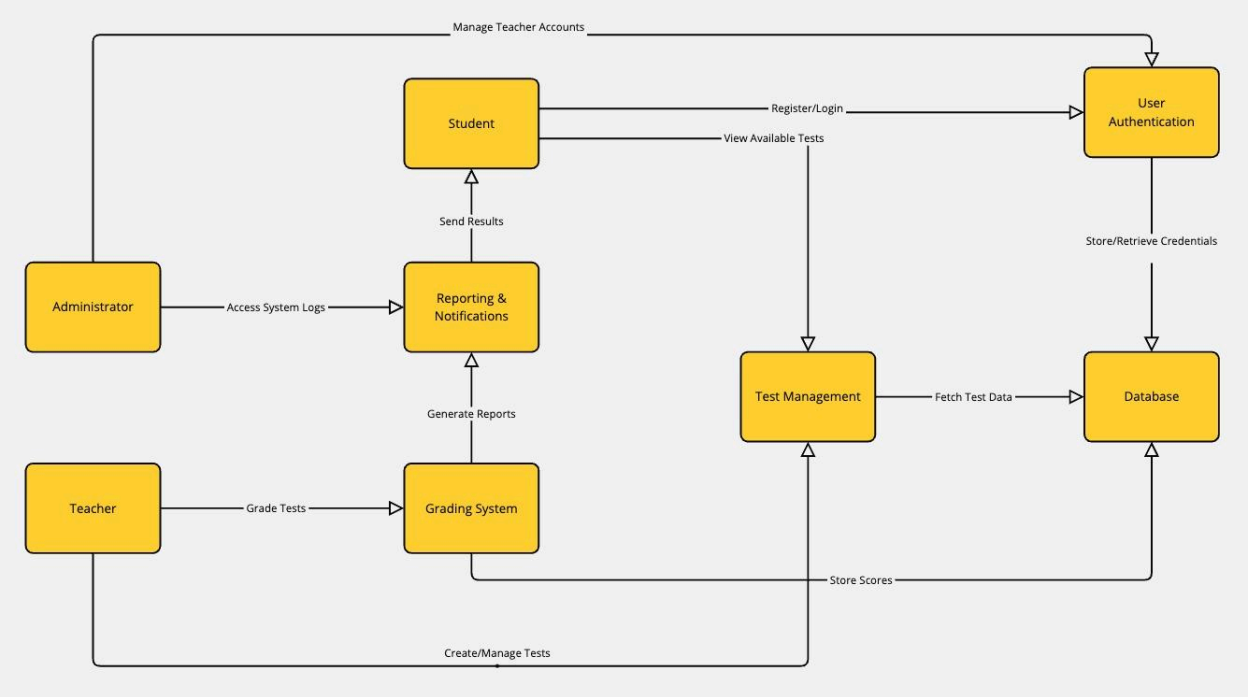
- **As a student,** I want to take tests with a session key so that I can complete my practice exams.
- **As a student,** I want to track my test scores so that I can monitor my progress.
- **As a teacher,** I want to create and manage test sessions so that I can facilitate student practice.
- **As a teacher,** I want to grade Writing and Speaking tests so that I can provide detailed feedback.
- **As an administrator,** I want to manage teacher accounts so that I can ensure only authorized personnel create tests.
- **As an administrator,** I want to enforce security policies so that user data remains protected.

6.2 Diagrams

- **Use Case Diagram:**



- **Data Flow Diagram:** [Link Here]



- **System Architecture Diagram:** Link here

---

7. Technical Specification Document

7.1 Technology Stack

- **Frontend:** React.js
- **Backend:** Node.js
- **Database:** MongoDB or PostgreSQL
- **Third-party services:** Firebase, Supabase, Socket.IO
- **DevOps & Deployment:** AWS, Netlify, Onrender, Heroku
- **CSS Frameworks:** TailwindCSS, BootstrapCSS

---

8. Validation and Verification

- Functional testing for each module.
- Performance testing under heavy loads.
- Security audits to identify vulnerabilities.

9. Appendices

- Future enhancements: AI-based automated grading.
- API documentation (if applicable).