Software Requirements Specification

for

Digital Exam Platform

Version 1.0

Prepared by

Group Name: ExamMasterPro

<name>	**<student #>**	**<e-mail>**
<name>	**<student #>**	**<e-mail>**
<name>	**<student #>**	**<e-mail>**
<name>	**<student #>**	**<e-mail>**
<name>	**<student #>**	**<e-mail>**

Instructor: ***<place your instructor's name here>***

Course: **<place your course name here>**

Lab Section: ***<place your lab section here>***

Teaching Assistant: ***<place your TA's name here>***

Date: **<place the date of submission here>**

Contents

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Draft Type and Number	Full Name	Information about the revision. This table does not need to be filled in whenever a document is touched, only when the version is being upgraded.	00/00/00

1 Introduction

This document outlines the software requirements for the **Digital Exam Platform**, a web-based system designed to allow professors to create exams and students to securely take them online. It defines the functional and non-functional requirements, user interfaces, and constraints that will guide the development of the system.

1.1 Document Purpose

The purpose of this Software Requirements Specification (SRS) is to clearly define the functional and non-functional requirements for the **Digital Exam Platform**. This document serves as a contract between the development team and stakeholders (instructor, users) and will guide the design, implementation, testing, and deployment of the system. This SRS covers the full scope of the platform, including admin, teacher, and student interfaces.

1.2 Product Scope

The **Digital Exam Platform** is a secure, scalable web application that enables:

- **Professors** to create and manage exams by selecting exam type (e.g., GATE, JEE), subject, topic, and question type (MCQ, MSQ, NAT).
- **Students** to select their exam context and securely take exams with real-time timers, navigation, and result tracking.
- **Admins** to manage user roles and monitor system usage.

Key benefits include:

- Eliminates paper-based exams.
- Supports OTP-based login for students.
- Real-time exam analytics.
- Secure data storage via Firebase.

The system will be implemented using HTML, CSS, JavaScript, and Firebase (Authentication, Firestore).

1.3 Intended Audience and Document Overview

This document is intended for:

- **Development Team**: To guide implementation and testing.
- **Instructor/Client**: To validate requirements and provide feedback.
- **Teaching Assistant**: To evaluate completeness and correctness.
- **Future Developers**: For system maintenance and enhancements.

Document Overview:

- **Section 2**: High-level product description and constraints.
- **Section 3**: Detailed functional requirements and use cases.
- **Section 4**: Non-functional requirements (performance, security, quality).
- **Appendices**: Data dictionary and group activity log.

Recommended reading order: Sections 1 → 2 → 3 → 4 → Appendices.

1.4 Definitions, Acronyms and Abbreviations

SRS - Software Requirements Specification

UI - User Interface
UX - User Experience
MCQ - Multiple Choice Question (single correct answer)
MSQ - Multiple Select Question (multiple correct answers)
NAT - Numerical Answer Type
OTP - One-Time Password
Firebase - Google's BaaS platform (Auth, Firestore)
CRUD - Create, Read, Update, Delete
COMET - Component- and Message-based Software Engineering Techniques
UML - Unified Modeling Language

1.5 Document Conventions

Font: Arial, Size 11 for body, Size 12 for headings.
Spacing: Single-spaced, 1" margins.
Headings: Follow IEEE numbering (e.g., 1, 1.1, 1.1.1).
Emphasis: Italic for definitions and comments.
Diagrams: UML notation for use cases.
Requirements: Numbered as F1, F2, etc.

1.6 References and Acknowledgments

IEEE Std 830-1998, Recommended Practice for Software Requirements Specifications.
COMET Methodology: http://www.comet.unicam.it
UML 2.5 Specification: https://www.omg.org/spec/UML/2.5
Firebase Documentation: https://firebase.google.com/docs
Project Description Document (provided by instructor)

2 Overall Description

2.1 Product Overview

The Digital Exam Platform is a new, self-contained web application designed to modernize exam delivery in academic settings. It replaces manual or paper-based exams with a secure digital alternative. The system supports three user roles: Admin, Teacher, and Student.

The platform integrates with Firebase for authentication and real-time data storage. It is not a subsystem but a standalone product.

2.2 Product Functionality

F1: User role-based access (Admin, Teacher, Student)
F2: OTP-based login for students
F3: Username/password login for teachers and admins
F4: Teachers create questions with subject/topic tagging
F5: Students select exam context and take timed exams
F6: Real-time exam timer and question navigation
F7: Results and score calculation post-exam

2.3 Design and Implementation Constraints

Must use COMET methodology for design.
Must use UML for modeling (use case, sequence, class diagrams).
Frontend: HTML, CSS, JavaScript only (no frameworks like React).
Backend: Firebase (Authentication, Firestore).
Hosting: Firebase Hosting.
Browser Support: Chrome, Firefox, Edge (latest versions).
Must support responsive layout for mobile and desktop.

2.4 Assumptions and Dependencies

Firebase will remain available and within free tier limits.
Users have stable internet access during exams.

Teachers will input correct answers and marks.
Students will not attempt to bypass security.
Dependencies:
Firebase SDK (v10.12.5)
Google reCAPTCHA for OTP
No third-party payment or external APIs

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

Student Login UI
Login Screen Mockup
Role selection (Admin/Teacher/Student)
Username/password or OTP input
Dynamic form based on role

Exam Interface
Left: Question area (text, options, NAT input)
Right: Navigation panel (timer, filters, question grid)
Draggable splitter for resizable layout

3.1.2 Hardware Interfaces

Standard web browser (no special hardware)
Input: Keyboard and mouse/touchscreen

Output: Monitor display
Internet connection required

3.1.3 Software Interfaces

Firebase Authentication: For login (email/password, phone OTP)
Firebase Firestore: Store questions, user data, exam context
Firebase Hosting: Deploy web app
Browser LocalStorage: Store exam context temporarily

3.2 Functional Requirements

F1 - The system shall allow users to select their role (Admin, Teacher, Student).
F2 - The system shall support OTP-based login using Firebase Phone Authentication.
F3 - The system shall support username/password login for teachers and admins.
F4 - Teachers shall be able to create MCQ, MSQ, and NAT questions with correct answers and marks.
F5 - Questions shall be stored in Firestore under exam_questions/{docId}.
F6 - Students shall select exam context (exam, branch, class, etc.) before starting.
F7 - The exam interface shall display a real-time countdown timer.
F8 - Students shall be able to mark questions for review and clear responses.
F9 - The system shall calculate and display the final score after exam submission.
F10 - Admins shall assign teacher roles and manage user access.

3.3 Use Case Model

3.3.1 Use Case #1 (use case name and unique identifier – e.g. U1)

- **3.3.1 Use Case: UC1 - Take Exam**
- **Author**: Maya Patel
- **Purpose**: Allow student to securely take a timed exam.
- **Requirements Traceability**: F6, F7, F8, F9
- **Priority**: High
- **Preconditions**: User is logged in and has selected exam context.
- **Postconditions**: Exam is submitted; score is calculated.
- **Actors**: Student
- **Extends**: None
- **Includes**: UC2 (Login), UC3 (Select Exam Context)

Flow of Events

1. **Basic Flow**
    1. Student selects exam type, branch, class, subject, topic.
    2. System loads questions from Firestore.
    3. Student answers questions, marks for review.
    4. Timer counts down.
    5. Student clicks "Submit Exam".
    6. System calculates and displays score.
2. **Alternative Flow**
    - If time runs out, exam auto-submits.
3. **Exceptions**
    - E1: No internet → Show "Offline" message, retry on reconnect.
    - E2: Invalid exam context → Redirect to selection.

Notes/Issues

- Ensure questions load quickly.
- Prevent back navigation during exam.

4 Other Non-functional Requirements

4.1 Performance Requirements

P1: Questions must load within 3 seconds of exam start.
P2: OTP must be sent within 10 seconds.
P3: Exam timer must update every second without drift.
P4: System must support up to 100 concurrent students.

4.2 Safety and Security Requirements

Database: Firestore collections: users, exam_questions
Legal: No personal data collection beyond phone/email.
Accessibility: Support screen readers and keyboard navigation.
Internationalization: English only (Phase 1).
Reuse: Login and exam components can be reused in future modules.

4.3 Software Quality Attributes

4.3.1 Reliability
The system shall maintain 99% uptime during exam hours.
Auto-save user answers every 30 seconds.
4.3.2 Maintainability
Code shall be modular and well-commented.
Use consistent naming (camelCase for JS, kebab-case for HTML/CSS).
Version control via Git.
4.3.3 Usability
UI shall be intuitive with minimal training.
Responsive design for mobile and desktop.
Clear error messages for login and exam issues.
4.3.4 Portability
Works on Chrome, Firefox, Edge.

No OS-specific dependencies.

5 Other Requirements

Database: Firestore collections: users, exam_questions
Legal: No personal data collection beyond phone/email.
Accessibility: Support screen readers and keyboard navigation.
Internationalization: English only (Phase 1).
Reuse: Login and exam components can be reused in future modules.

Appendix A – Data Dictionary

Appendix B - Group Log

<Please include here all the minutes from your group meetings, your group activities, and any other relevant information that will assist in determining the effort put forth to produce this document>