The Islamia University of Bahawalpur

Software Requirement Specifications

By

Student Name: Marab Shahzad

Roll No: F22BINFT1M01153

Session: 2022 – 2026

Supervisor: Sir Muzammil Ur Rehman

Bachelor of Science in Information Technology

Project Title

Virtual Classroom: An Online Education Platform

Summary

This Software Requirements Specification (SRS) outlines the requirements for an e-classroom web application created to support online education for students, teachers, and administrators. The platform provides virtual classroom functionalities, resource sharing, assignment handling, and performance monitoring. It utilizes modern web technologies such as React.js, Node.js, and SQL to deliver a smooth and secure user experience. A review of current e-learning systems highlights various limitations that this project seeks to overcome by offering improved usability, flexibility, and enhanced features. This document details the essential functional and non-functional requirements that ensure the website is safe, scalable, responsive, and easy to use. The SRS acts as a blueprint for developing a reliable and efficient digital learning environment.

Table of Contents

1. Introduction

The e-Classroom system is developed to offer a smooth and interactive online learning experience for educational organizations, including schools, colleges, and training institutes. It allows teachers and learners to collaborate in a virtual academic environment using features such as classroom creation, assignment distribution, online discussions, and file sharing. Teachers can upload learning materials, schedule lessons, grade assignments, and interact with students in real time. Students, on the other hand, can access course content, upload coursework, check grades, and communicate with teachers.

The platform's frontend is built using HTML, CSS, Tailwind CSS, and React.js to ensure responsiveness and usability. The backend, developed with Node.js, connects with an SQL database to store and retrieve user information, class data, and submitted assignments. This project aims to support remote education with a system that is adaptable, scalable, and suitable for modern digital learning needs.

1.1 Purpose

The purpose of this e-Classroom system is to provide a convenient and intuitive online learning platform for instructors and students. It is designed to support virtual teaching by allowing teachers to manage courses, share educational content, assign tasks, and evaluate student work. Students can view learning materials, submit assignments, and participate in discussions.

The SQL-based database ensures secure and structured data handling, while the frontend—created using HTML, CSS, Tailwind CSS, and React.js—delivers a dynamic and responsive interface. Node.js powers the backend to provide fast processing and system scalability. This project seeks to enhance the online learning experience by improving accessibility, flexibility, and support for remote education.

1.2 Scope

The scope of the e-Classroom platform includes essential features needed for effective online learning. The system will allow students, teachers, and administrators to register, log in, and manage their profiles. Teachers will be able to create classes, upload lesson materials, and schedule activities, while students will access course resources, complete assignments, and upload their work.
Additionally, the platform will offer chat and discussion tools for real-time communication. User and class data will be stored securely using an SQL database.
The scope **does not** include features such as video conferencing integration, advanced performance analytics, or a dedicated mobile app. The primary focus is on building a fully functional web-based platform for educational institutions.

1.3 Product Perspective

The e-Classroom website serves as a core component for delivering online education within an institution. It consists of several interconnected modules that operate together to provide a unified experience.
The frontend, built with HTML, CSS, Tailwind CSS, and React.js, interacts with a Node.js backend, which communicates with an SQL database.

Core modules include:

• User Management

Handles account creation, role assignments, and login processes for students, teachers, and administrators. This module forms the entry point for accessing system features.

• Class Management

Allows teachers to set up digital classrooms, upload learning content, and organize lesson schedules. This module shapes how students engage with course materials.

• Assignment Handling

Manages assignment creation, submission, and evaluation. Students upload their work, and teachers provide grades and feedback.

• Real-Time Messaging

Supports communication through chat to enable discussions and clarify doubts within a virtual environment.

1.4 User Characteristics

The platform caters to various user groups, each with their own responsibilities and permissions:

1. Administrator

- Has full authority over system operations
- Manages user accounts, roles, schedules, and system performance
- Can perform updates, backups, and database maintenance

2. Teacher

- Creates and manages courses
- Uploads learning materials and assigns tasks
- Reviews, grades, and provides feedback on student submissions
- Participates in and manages chat discussions

3. Student

- Creates a personal profile and joins classes
- Accesses study materials and views assignments
- Submits completed work
- Participates in real-time discussions

1.5 Similar Apps and Systems / Literature Review

To understand the current landscape of e-classroom platforms, a review of existing systems was conducted. The following are some notable examples of similar apps and systems, their features, and their shortcomings:

1. Google Classroom

Advantages:

- Simple interface
- Easy assignment management

- Integration with Google tools
- Real-time notifications

Limitations:

- Few customization options
- Not suited for non-Google environments
- Basic analytics

2. Moodle

Advantages:

- Highly customizable and open-source
- Supports plugins and themes
- Includes quizzes, forums, and collaboration tools
- Has a mobile app

Limitations:

- Difficult for beginners
- Requires technical expertise
- Outdated design

3. Edmodo

Advantages:

- Familiar social-media-like interface
- Tools for parents, teachers, and students
- Built-in quizzes and resource sharing

Limitations:

- Less scalable for large institutions
- Mainly targeted at K-12 education

1.6 Proposed Technologies

Frontend

- **HTML (HyperText Markup Language):** Used for structuring the web pages of the system.
- **CSS (Cascading Style Sheets):** Used for styling the user interface to ensure a visually appealing and responsive design.

- **Tailwind CSS:** A utility-first CSS framework that simplifies the process of building modern, responsive designs with pre-defined styles.
- **React.js:** A JavaScript library for building user interfaces, enabling the creation of reusable components and efficient rendering.

2. Backend Technologies

- **Node.js:** A JavaScript runtime environment that enables the development of scalable and high-performance server-side applications.

3. Database Technologies

- **SQL:** A relational database management system used to store, retrieve, and manage data efficiently.

2. Requirements

The e-classroom website aims to provide a comprehensive platform for online education, catering to the needs of students, teachers, and administrators. The system will enable teachers to create and manage virtual classrooms where they can share educational resources, assign tasks, and evaluate student performance. Students will be able to access course materials, submit assignments, and track their progress. The platform will also facilitate real-time communication between teachers and students through discussion forums and chat features.

The project will include a user authentication system to ensure secure access, with separate roles for administrators, teachers, and students. Administrators will have the ability to manage users, monitor system usage, and generate reports. Teachers will be able to create quizzes, grade assignments, and provide feedback to students, while students will have access to their grades and personalized learning resources.

2.1 Function Requirements

Authors will provide definite number of functional requirements in standard format. These requirements directly in the with functions which are already provided. The requirement format as following.

2.1.1. Sign Up

- **Name:** FR001
- **Purpose:** Sign-up functionality is required for users to register and become members of the system.

- **User(s):** Administrator, Teacher, Student
- **Input:**
  - Name: Full name as per the user's official ID.
  - Email: A valid and unique email address.
  - Password: Must be at least 8 characters long, include uppercase letters, lowercase letters, numbers, and special characters.
  - Role: User type (Administrator, Teacher, Student).
  - Phone: A working phone number.
- **Output:** Registered user can log in to the system with their credentials.

2.1.2. Login

- **Name:** FR002
- **Purpose:** Allow registered users to access the system.
- **User(s):** Administrator, Teacher, Student
- **Input:**
  - Email: User's registered email address.
  - Password: User's account password.
- **Output:** Authenticated user is granted access to their respective dashboard.

2.1.3. Create Virtual Classroom

- **Name:** FR003
- **Purpose:** Enable teachers to create and manage virtual classrooms for specific courses.
- **User(s):** Teacher
- **Input:**
  - Classroom Name: Unique name of the class.
  - Course Description: Details about the course.
  - Enrollment Key: A unique code to allow students to join the class.
- **Output:** A virtual classroom is created and accessible to the teacher and enrolled students.

2.1.4. Upload Course Material

- **Name:** FR004
- **Purpose:** Allow teachers to upload educational resources for students.
- **User(s):** Teacher
- **Input:**
  - File: Educational resource file (PDF, Word, Video, etc.).
  - Description: Brief description of the uploaded material.
- **Output:** Course material is available for students to download or view.

2.1.5. Submit Assignment

- **Name:** FR005
- **Purpose:** Allow students to submit their assignments.
- **User(s):** Student
- **Input:**
  - File: Assignment file in the accepted format (PDF, Word, etc.).
  - Comments: Optional remarks or notes regarding the submission.
- **Output:** The assignment is successfully submitted and marked for review by the teacher.

2.1.6. Grade Assignments

- **Name:** FR006
- **Purpose:** Enable teachers to evaluate and grade students' assignments.
- **User(s):** Teacher
- **Input:**
  - Assignment: Submitted assignment.
  - Grade: Assigned grade or marks.
  - Feedback: Optional comments for the student.
- **Output:** Graded assignments and feedback are visible to the students.

1.1. Non-Functional Requirements

The non-functional requirements for the e-classroom website define the quality attributes, performance benchmarks, and operational constraints of the system. These requirements ensure that the system is reliable, efficient, and user-friendly.

2.2.1. Performance

- The system must handle up to 500 concurrent users without noticeable degradation in response time.
- The average page load time should not exceed 3 seconds under normal operating conditions.
- Queries and reports should execute within 2 seconds for up to 10,000 records.

2.2.2. Scalability

- The system should be scalable to support additional users, classrooms, and courses as the user base grows.
- Infrastructure should support easy horizontal scaling of the backend and database components.

2.2.3. Security

- User passwords must be encrypted using industry-standard hashing algorithms.
- The system should implement role-based access control to ensure data privacy and prevent unauthorized access.
- All data exchanges must use HTTPS to ensure secure communication.

2.2.4. Usability

- The platform should have an intuitive interface, ensuring users can perform core functions (e.g., sign up, join a class, upload assignments) within three clicks.
- All features must be accessible to users with basic computer literacy.
- The system should provide comprehensive help documentation and tooltips for user guidance.

2.2.5. Availability

- The system should have an uptime of at least 99.5% per month, excluding planned maintenance.
- Scheduled maintenance should occur during off-peak hours and notify users in advance.

2.2.6. Compatibility

- The website must be responsive and compatible with modern web browsers, including Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari.
- The platform should be accessible on both desktop and mobile devices with consistent functionality.

2.2.7. Maintainability

- The codebase should follow modular design principles to simplify updates and debugging.
- All code must be documented and follow standard naming conventions for better maintainability.
- The system should allow developers to integrate future features without disrupting existing functionalities.

2.2.8. Backup and Recovery

- The system must perform automatic daily backups of all critical data, including user profiles, classroom data, and uploaded files.
- Data recovery in the event of a system failure should not take more than 1 hour.

3.Use Cases and Flow of Processes

Use cases are the formal representation of process flow defined by functional requirements. Authors should provide a flow of events in this. There should be system level use case which is directly influenced to project functions with their user(s) (called actor(s)) which are described in requirements chapters

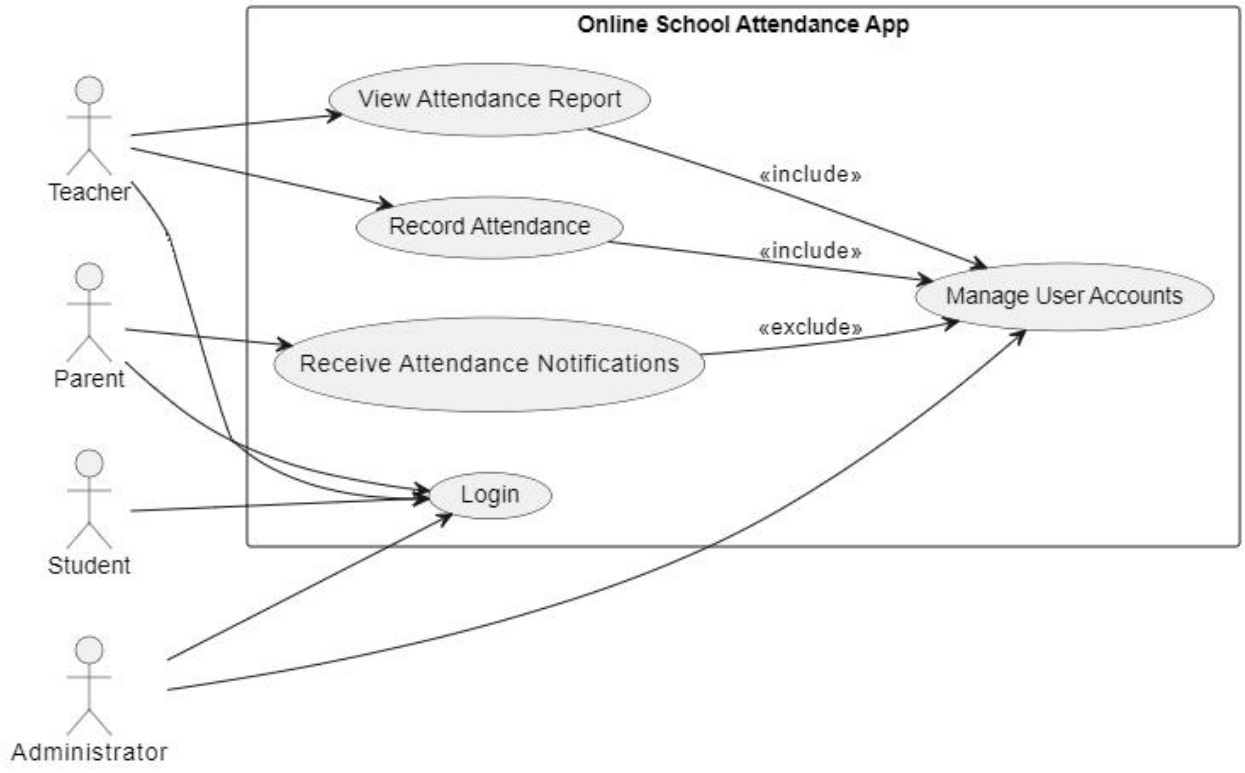

Figure 1: System Level Use Case Diagram

1.2. Use Case Description

1. Login:

Use Case ID:	UC-1
Use Case Name:	Login
Actors:	Actor: Teacher, Student, parents
Description:	Allows users to login for an account on app by providing necessary personal information.
Trigger:	When a new user launches the app for the first time and wishes to create an account to access its features.
Preconditions:	User has launched the app.
Postcondition:	User's account is created, allowing access to other features of the app.
Normal Flow:	1. User launches the app and selects the "Login" option. 2. User fills in the required details and submits the registration form. 3. The app validates the entered information for completeness and accuracy 4. Upon successful validation, the app creates a new user account. 5. The user can log in to the app using their registered email and password.
Alternative Flows:	User opts to login using social media credentials instead of traditional email.
Exceptions:	User registration fails due to invalid or incomplete information provided during the registration process.
Business Rules:	None
Assumptions:	Users provide accurate and valid information during the registration process.

1. Manage User Accounts

Use Case ID:	UC-2
Use Case Name:	Manage user accounts
Actors:	Primary Actor: Administrator
Description:	The Administrator manages user accounts within the system, including creating new accounts, modifying existing accounts, and deactivating or deleting accounts as needed.
Trigger:	The login action triggers the "Manage Users" use case, allowing the Administrator to perform user management tasks.
Preconditions:	Administrator is logged into the system.
Postconditions:	User account changes are saved in the system.
Normal Flow:	The normal flow of the "Manage Users" use case involves the Administrator logging into the system, accessing the user management interface, and

	performing actions such as creating, modifying, or deactivating user accounts as needed.
Alternative Flows:	An alternative flow for the "Manage Users" use case could be the Administrator updating user permissions or resetting passwords.
Exceptions:	If the Administrator lacks appropriate permissions, access to user management functionalities will be denied.
Business rules:	Only administrators have the authority to create, modify, or delete user accounts within the system.
Assumption:	The Administrator has the necessary permissions and authority to perform user management tasks within the system

2. Record Attendance:

Use Case ID:	UC-3
Use Case Name:	Record Attendance
Actors:	Primary Actor: Teacher
Description:	The Teacher records student attendance for each class session, marking students as present, absent, or tardy.
Trigger:	Initiation of a class session by a Teacher.
Postconditions:	Teacher is logged into the system and has access to the attendance recording functionality.
Preconditions:	Attendance records are updated in the system for the specified class session.
Normal flow:	The normal flow of the "Record Attendance" use case involves the Teacher logging into the system, selecting the class session, marking students as present, absent, or tardy, and saving the attendance records.
Business rule:	Attendance must be recorded for each class session within a specified timeframe to ensure accurate tracking of student attendance.
Assumption:	Teachers have access to class rosters and accurate student information to accurately record attendance.

3. View attendance report:

Use Case ID:	UC-4
Use Case Name:	View attendance reports
Actors:	Primary Actor: Teacher,Student,Parent
Description:	Teachers can view attendance reports that summarize student attendance data for specific classes, time periods, or individual students to monitor attendance trends and patterns.
Trigger:	The actor selects the option to view attendance records from the system menu.

Preconditions:	Actor can logged into the system and has access to the viewing attendance report functionality.
Postconditions:	The actor successfully views the attendance record based on the specified criteria (e.g., date range, student, class).
Normal flow:	The normal flow of viewing attendance records involves users logging into the system, navigating to the attendance report section, selecting the desired parameters (such as date range or student name), and generating the report for viewing.
Business rule:	none
Assumption:	Attendance data is accurately recorded and up-to-date in the system for viewing meaningful reports.

4. Receive attendance notifications:

Use Case ID:	UC-5
Use Case Name:	Record Attendance
Actors:	Primary Actor: Teacher
Description:	Parents receive automated notifications (e.g., via email, SMS) when their child is marked as absent or tardy for a class session, keeping them informed about their child's attendance status.
Trigger:	The attendance system detects a change in the attendance status of subscribed students (e.g., marked as absent or tardy).
Postconditions:	Teacher is logged into the system and has access to the attendance recording functionality.
Preconditions:	The Parent receives attendance notifications for the subscribed students based on their attendance status (e.g., absent or tardy).
Normal flow:	It involves parents receiving automated attendance notifications for their child's absences or tardiness.
Business rule:	none
Assumption:	Parents opt in to receive attendance notifications for their child.

4.References

- **IEEE Computer Society.** (1998). *IEEE Recommended Practice for Software Requirements Specifications (IEEE Std 830-1998).*

- **Sommerville, I.** (2016). *Software Engineering* (10th ed.). Pearson Education.

- **Pressman, R. S., & Maxim, B. R.** (2014). *Software Engineering: A Practitioner's Approach* (8th ed.). McGraw-Hill.

- **Dhawan, S.** (2020). Online Learning: A Panacea in the Time of COVID-19. *Journal of Educational Technology Systems,* 49(1), 5–22.

- **React.js Documentation.** (2024). *React – A JavaScript Library for Building User Interfaces.* Available at: https://react.dev/

- **Node.js Documentation.** (2024). *Node.js JavaScript Runtime.* Available at: https://nodejs.org/

- **Moodle Documentation.** (2024). *Moodle Learning Management System.* Available at: https://docs.moodle.org/