Software Requirements Specification

for

ClassMate

Version 1.0 approved

Prepared by Musaddiq Kamal Khan

Team members: Ahmed Youshay, Anas Ahmed

29/09/2024

Table of Contents

Table of Contents ii

Revision History ii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 3

2.5 Design and Implementation Constraints 3

2.6 User Documentation 3

2.7 Assumptions and Dependencies 3

2.8 Process Model……………………………………………………………………………..3

2.9 Project Plan...……………………………………………………………………………...4

2.10 Feasibility Report…………………………………………………………………………. 4

2.11 Textual UseCase…………………………………………………………………………...5

2.12 Video Demo………………………………………………………………………………..6

3. External Interface Requirements

3.1 User Interfaces 6

3.2 Hardware Interfaces 6

3.3 Software Interfaces 6

3.4 Communications Interfaces 7

4. System Features 8

4.1 Student 8

4.2 Instructor 9

4.3 Admin 10

5. Other Nonfunctional Requirements 11

5.1 Performance Requirements 11

5.2 Safety Requirements 12

5.3 Security Requirements 12

5.4 Software Quality Attributes 12

5.5 Business Rules 13

**6. Appendix** ................................................................................................................................. 14

6.1 Login Request ................................................................................................................................................ 14

6.2 Activity Diagram ............................................................................................................................................. 15

6.3 Class Diagram ..................................................................................................................................................16

6.4 Collaboration Diagram .................................................................................................................................... 17

6.5 Component Diagram ....................................................................................................................................... 18

6.6 Use Case ......................................................................................................................................................... 19

6.7 Deployment Diagram ..................................................................................................................................... 20

6.8 Sequence Diagrams ........................................................................................................................................ 21

6.9 State Chart ....................................................................................................................................................... 24

# Introduction

## Purpose

This SRS outlines the software requirements for the "ClassMate", a Student Management System. The system will handle the key administrative and academic functions of an educational institution, specifically focusing on student, teacher, and admin management. The main entities covered in this SRS include students, admins, teachers, courses, enrollments, assignments, attendance, marks, books, comments, fees, and transcripts. The purpose of this document is to define the scope, requirements, and functionalities of the system to guide the development team and ensure the system meets institutional needs

## Document Conventions

Standard formatting is used in this SRS. Headings are in bold and bigger, and important keywords such as **MUST**, **SHOULD**, and **OPTIONAL** indicate requirement priorities. Each requirement is individually prioritized, and the document follows the IEEE 830-1998 SRS standards.

## Intended Audience and Reading Suggestions

This SRS is made for our teacher(supervisor) and fellow team members acting as developers. It is recommended that all readers start with the overview sections to gain an understanding of the system before diving into more specific areas relevant to their roles.

## Product Scope

ClassMate is designed to improve the efficiency of managing student-related academic tasks. Its purpose is to support educational institutions by providing a user-friendly solution for administrators, teachers, and students to handle course enrollment, assignment submissions, attendance tracking, and grade management. The system aims to enhance operational efficiency while improving the learning experience for students.

## References

 Django **Documentation**: <https://docs.djangoproject.com>

 PostgreSQL **Documentation**: <https://www.postgresql.org/docs/>

 IEEE **Standard for Software Requirements Specifications (IEEE Std 830-1998)**

# Overall Description

## Product Perspective

ClassMate is a web-based application that will integrate seamlessly into the existing educational ecosystem of an institution. It functions as a standalone system. The system aims to reduce manual paperwork and streamline academic processes, making it easier for users to manage their tasks efficiently. The system is built with a focus on user experience, ensuring that students, teachers, and administrators can navigate it intuitively.

## Product Functions

Students can:

* View attendance
* View marks
* Register/withdraw from courses
* Give feedback
* Borrow books from library
* Submit/view assignments
* View fees detail

Teachers can:

* Update attendance
* Update marks
* Manage courses
* Give assignments
* View salary details

Admin can:

* Create, read, update, and delete accounts for other users

## User Classes and Characteristics

**Students:**

* **Frequency of Use**: Daily, as they interact with the system for assignments, grades, and course materials.
* **Subset of Functions**: Use features related to course registration, assignment submissions, attendance tracking, and viewing grades.

**Teachers:**

* **Frequency of Use**: Daily to weekly, depending on course loads and assignments.
* **Subset of Functions**: Use functions for managing courses, grading assignments, tracking attendance, and communicating with students.

**Administrators:**

* **Frequency of Use**: Weekly to daily, depending on their role in the institution.
* **Subset of Functions**: Use comprehensive features for managing users, courses, attendance, and overall system settings.

## Operating Environment

**Hardware**: Any device capable of running a modern web browser, such as laptops, desktops, tablets, and smartphones.

**Software**: Compatible with major web browsers like Chrome, Firefox, Safari, and Edge, ensuring a smooth user experience across different platforms.

## Design and Implementation Constraints

Assignment Submission Challenges:

Variability in students’ ability to submit assignments on time requires robust reminders and tracking features to manage late submissions or incomplete assignments.

Hardware Limitations:

The application must perform well on a range of hardware, which can vary widely among users, potentially limiting resource-intensive features and necessitating optimizations for lower-end devices.

Technological Requirements:

The system will be built using specific technologies, such as Django for backend development and PostgreSQL for data management, which may limit options for developers regarding language or framework choices.

Compliance and Security Regulations:

Adherence to educational regulations (such as FERPA) is crucial for data privacy and security, requiring the system to handle student information in accordance with these standards.

## User Documentation

The ClassMate system will not include formal user documentation components such as user manuals, online help, or tutorials upon delivery.

## Assumptions and Dependencies

Technology Familiarity:

Users are expected to have a basic understanding of web applications and internet navigation. This assumption may affect the design choices made for user interfaces and functionality.

Stable Internet Connectivity:

The application assumes that users will have access to reliable internet connections, which is essential for optimal performance and usability. Any disruptions could significantly impact user experience.

## Process Model

The Incremental Development Model is adopted for the development of the ClassMate system. This approach divides the project into smaller, manageable increments, each delivering a part of the final functionality. The model ensures that critical functionalities are built and delivered early, with additional features integrated in subsequent iterations.  
Each increment involves the following steps:

* Requirements gathering for the increment.
* System design and architecture updates.
* Incremental development and testing.
* Integration with previously developed increments.
* Client feedback and iterative improvements.

## Project Plan

The project plan is organized into distinct phases:

1. **Requirements Gathering:**
   * Develop Software Requirements Specification (SRS).
2. **System Design:**
   * Create diagrams (ERD, domain model, use case, class, sequence, activity, component, deployment).
   * Define system architecture.
3. **Database Design and Setup:**
   * Implement ERD in PostgreSQL, create Django models, and generate SQL scripts.
4. **Implementation:**
   * Set up Django project and apps.
   * Develop backend functionality, frontend interfaces, and implement authentication.
5. **Testing and Validation:**
   * Conduct unit and integration testing.
6. **Deployment Preparation:**
   * Configure hosting, production settings, and deploy the application.
7. **Final Deliverables:**
   * Compile project report, prepare presentation, finalize source code, and demonstrate working software.

## Feasibility Report

**Technical Feasibility:**  
The project leverages widely used technologies like Django for the backend, PostgreSQL for the database, and HTML/CSS/JavaScript for the frontend. This ensures that technical implementation is feasible with available tools.

**Operational Feasibility:**  
The system addresses real-world needs, such as managing courses, assignments, and attendance, which ensures operational relevance and user acceptance.

**Economic Feasibility:**  
The open-source nature of the selected technologies reduces development costs. Minimal additional infrastructure is required, ensuring a cost-effective solution.

**Schedule Feasibility:**  
The incremental model and clear project milestones ensure that the system can be developed within the specified timeline.

## Homogenization Process

The homogenization process in ClassMate ensures uniformity in data and functionality across modules. Key aspects include:

1. **Data Standardization:**
   * Consistent formatting of student, teacher, and course data.
   * Validation checks to prevent errors, such as ensuring marks do not exceed the maximum value.
2. **Feature Consistency:**
   * All user interfaces follow a unified design language for ease of navigation.
   * Similar workflows are implemented for attendance, marks, and assignments.
3. **Integration of Modules:**
   * Seamless interaction between modules (e.g., assignment grades affecting final transcript generation).

## Use Case in Textual Format

Use Case: View Student Profile

Actor(s): Student  
Description: Students can view their personal details, including enrolled courses, attendance, and grades.

Use Case: Register/Withdraw Courses

Actor(s): Student  
Description: Students can add or drop courses, subject to the course availability and prerequisites.

Use Case: View Attendance

Actor(s): Student  
Description: Students can check their attendance records for all enrolled courses.

Use Case: View/Print Transcript

Actor(s): Student  
Description: Students can view and download their academic transcript in PDF format.

Use Case: Submit/View Assignment

Actor(s): Student  
Description: Students can submit assignments online and view feedback from teachers.

Use Case: Upload Course Material

Actor(s): Teacher  
Description: Teachers can upload lecture notes, slides, and other course resources.

Use Case: Update Attendance

Actor(s): Teacher  
Description: Teachers can record attendance for students in their courses.

Use Case: Manage Users

Actor(s): Administrator  
Description: Administrators can add, edit, or remove users (students, teachers).

(You can continue this pattern for the remaining use cases.)

## Video of Working Software

https://drive.google.com/file/d/1sNC6TDvSAMSeCSNIpWb6\_cMAsATPoTCx/view?usp=drive\_link

# External Interface Requirements

## User Interfaces

The ClassMate system will feature user-friendly interfaces tailored for students, teachers, and administrators. Each role will have a distinct layout, with a consistent navigation menu on the left and dynamic content displayed in the main area.

Key characteristics include:

* GUIStandards: Consistent use of standard buttons (e.g., Submit, Cancel) and responsive design for both desktop and mobile.
* ErrorMessages: Clear, actionable error messages that guide users on how to resolve issues.
* AccessibilityFeatures: Support for keyboard navigation and screen readers to ensure usability for all students.

## Hardware Interfaces

The ClassMate system will interface with various hardware components, primarily supporting devices such as desktops, laptops, and tablets. The logical interactions involve data input from user devices through keyboards and touchscreens, while output is displayed on monitors or screens. Communication between the software and hardware will primarily utilize HTTP/HTTPS protocols for web-based functionalities, ensuring secure data transfer over the internet.

## Software Interfaces

The ClassMate system will interact with several key software components, ensuring a cohesive and efficient operation. The primary components include:

1. **Django (Version 4)**:

The web framework will handle server-side logic, user authentication, and database interactions. It communicates with the PostgreSQL database to store and retrieve user data, assignments, and grades.

1. **PostgreSQL (Version 14)**:

This relational database management system will store all application data, including user profiles, assignments, and grades. The communication occurs through SQL queries initiated by Django, with data coming in and out as structured data items.

1. **Front-end Technologies (HTML/CSS/JavaScript)**:

These technologies will be used for the user interface, interacting with the Django backend via RESTful APIs.

1. **Communication Services**:

The application will utilize HTTP/HTTPS for secure data transmission between the client-side and server-side components.

**Data Flow**

* **Incoming Data**:

User input (e.g., assignment submissions, profile updates) is sent to the server for processing.

* **Outgoing Data**:

The system responds with data such as user profiles, assignment statuses, and grade information

**Services and Data Sharing**

The system will require services like user authentication, data retrieval, and notifications. All user-related data will be shared across components, particularly between the Django backend and PostgreSQL database.

**Implementation Constraints**

For effective multitasking and performance, a global data area must be used for shared state management, especially in managing concurrent user sessions.

## Communications Interfaces

The ClassMate system will incorporate several communications functions essential for its operation:

1. **Web Browser Communication**:

The application will be accessible via modern web browsers (e.g., Chrome, Firefox, Safari) using HTTP/HTTPS protocols for secure communication between the client and server.

1. **Security and Encryption**:

All communications over the web will be secured using HTTPS to encrypt data in transit, protecting sensitive information from unauthorized access. Email communications will also follow best practices for security.

1. **Data Transfer Rates**:

The system will be designed to handle typical web data transfer rates, ensuring that user interactions are swift and responsive.

1. **Synchronization Mechanisms**:

To maintain data integrity, the system will implement mechanisms for synchronizing user data across sessions, ensuring that updates are reflected in real-time for all users.

# System Features

## Student

### **View Courses: view all the courses students can take according to the course’s semester and pre-requisite requirements.**

### **Course recommendations**: Provides personalized course suggestions based on the student’s academic performance and interests.

### **Register course**: Allows students to enroll in courses offered by the university for a given term.

### **Withdraw course**: Students can drop or withdraw from courses before a set deadline.

### **Student Profile**: Student can view their profile

### **View assignments**: Enables students to see all their course assignments and submission deadlines.

### **Submit assignments**: Allows students to upload and submit their completed assignments for evaluation.

### **View attendance**: Provides students with the ability to check their attendance records for each course.

### **View marks**: Displays the grades or marks that student have received in their courses.

### **Request book**: Allows students to place a request for borrowing a book from the university library.

### **View book details**: Provides access to book information, including availability, author, and location in the library.

### **Comment on assignments**: Enables students to provide feedback or ask questions about their assignments.

### **Give course feedback**: Allows students to submit feedback on courses and instructors at the end of the term.

### **View fees detail/challan**: Displays the student's current fee balance and allows them to view or download their fee challan for payment.

### **View/print transcripts**: Allows students to access and print their academic transcripts for their records or official use.

### **Track academic progress**: Allows students to monitor their progress towards degree completion by viewing course requirements and completed credits.

### **View library history**: Provides students with a record of books they have borrowed and returned.

### **Set reminders**: Allows students to set reminders for important dates such as assignment deadlines, exam schedules, or course registration periods.

### **Access digital library resources**: The university's digital library system provides students with online access to e-books, research papers, and journals.

### **View course schedules**: Displays class schedules for the student's enrolled courses, including time, location, and instructor details.

### **Receive notifications**: Sends automatic notifications or alerts regarding assignments, exams, upcoming deadlines, or important announcements from the university.

### **Data analytics dashboard**: Provides a comprehensive dashboard with key performance indicators (KPIs) on student performance, course enrollment trends, and financial reports.

## Instructor

### **Manage course offerings**: Allows instructors to view and manage all courses they are teaching, including course materials and schedules.

### **Course recommendations assistance**: Provides tools to review student performance and interests, helping instructors make personalized course recommendations.

### **Approve course registrations**: Allows instructors to approve or reject student requests for course enrollment if needed.

### **View/Manage student course withdrawals**: Enables instructors to track students who withdraw from their courses and update any necessary records.

### **Instructor Profile**: instructors can view their personal profile

### **Post/View assignments**: Instructors can create and post assignments for their courses and also view submitted assignments by students.

### **Grade/Assess assignments**: Allows instructors to view, grade, and provide feedback on assignments submitted by students.

### **Track attendance**: Provides instructors the ability to track and update attendance records for each class session.

### **Enter/View student marks**: Enables instructors to input and manage student grades or marks throughout the course.

### **Respond to book requests**: Instructors can view and respond to student book requests, such as recommending additional reading material from the library.

### **Upload/View course materials**: Allows instructors to upload and manage course resources such as textbooks, reading materials, and reference materials.

### **Respond to assignment comments**: Enables instructors to respond to student questions or feedback on assignments.

### **View course feedback**: Allows instructors to review feedback from students about their courses and teaching methods.

### **View Salary reports**: Instructors may view reports related to their salary

### **Generate academic reports**: Allows instructors to generate and print academic performance reports for their courses.

### **Track student academic progress**: Instructors can monitor the academic progress of their students based on grades, attendance, and assignment submissions.

### **View library usage history**: Enables instructors to view the library resource usage of their students, providing insights into reading habits.

### **Set notifications/reminders**: Allows instructors to set reminders for important course-related events like assignment deadlines, exams, or grading periods.

### **Access digital library resources**: Instructors can access the university’s digital library for e-books, research materials, and journals to share with students.

### **Set/View course schedules**: Enables instructors to set or adjust the class schedule for their courses and view the final schedule.

### **Send notifications**: Instructors can send automatic notifications or alerts to students regarding assignments, exams, or important announcements.

### **Data analytics dashboard**: Provides instructors with a dashboard that tracks student performance, attendance, assignment completion, and overall course trends.

### Request extra Class: Instructors can request to set up an extra class with students if more lectures are required

## Admin

### Control users: Add/delete/manage Users

# Other Nonfunctional Requirements

## Performance Requirements

The ClassMate system must meet the following performance requirements to ensure a smooth user experience:

* **Response Time**: The system should respond to user actions (e.g., submitting assignments, loading grades) within 2 seconds under normal load conditions to maintain user engagement.
* **Concurrent Users**: The application must support at least 100 concurrent users without performance degradation, ensuring reliable access during peak usage times, like assignment deadlines.
* **Data Load Time**: Pages displaying student grades and assignments should load within 3 seconds, even with the maximum data volume expected.

## Safety Requirements

The ClassMate system must adhere to the following safety requirements to prevent loss, damage, or harm from its use:

* **Data Integrity**: Implement safeguards to ensure that student data and grades are not lost or corrupted during processing or storage.
* **User Privacy**: Protect user information by ensuring compliance with data protection regulations preventing unauthorized access or data breaches.
* **Error Handling**: The system must include mechanisms for proper error handling, preventing crashes that could lead to data loss or user frustration.

## Security Requirements

The ClassMate system must ensure the security and privacy of sensitive data, particularly grades. Key security requirements include:

* **Data Protection**: Grades must be encrypted both in transit and at rest.
* **User Authentication**: Implement secure login credentials with options for multi-factor authentication.
* **Access Controls**: Use role-based access to ensure only authorized users can view or modify grades

## Software Quality Attributes

The ClassMate system must meet the following software quality attributes to ensure a robust and reliable user experience:

* **Usability**: The system must be intuitive and easy to navigate for all user classes (students, teachers, admins) with minimal training required. User satisfaction is prioritized over advanced features.
* **Availability**: The system must be available 99.9% of the time, ensuring students and teachers can access it during critical times, like assignment deadlines and grade reviews.
* **Reliability**: Data integrity and system reliability are essential. The system must handle user input without errors and must prevent any loss of data during normal operations.
* **Security**: Protect user data through encryption and robust authentication mechanisms, ensuring compliance with data protection standards.

## Business Rules

The following business rules will govern how different user roles interact with the ClassMate system:

**Role-Based Access**:

* + Students can view their own grades, submit assignments, and view course materials.
  + Teachers can manage their courses, grade assignments, and communicate with students.
  + Admins can access all system data, including student records, teacher data, and system settings.
* **Assignment Submission**:
  + Students can only submit assignments before the specified due date. Late submissions will only be accepted if the teacher extends the deadline.
* **Grade Publication and Editing**:
  + Teachers can assign and modify grades. Once grades are published, only teachers or authorized admins can make changes, with any modifications logged for transparency.
* **Communication Rules**:
  + Teachers and admins can send messages to individual students or the entire class. Students can only send messages to their teacher or admin.
* **System Access**:
  + Only authorized users (students, teachers, admins) can access the system through secure authentication. Unauthorized users will be blocked from accessing sensitive data.

# Appendix

## Login Request

# Login request uml

## Activity Diagram

## Class Diagram

## A screenshot of a diagram Description automatically generated

## A screenshot of a computer program Description automatically generatedCollaboration Diagram

## Component Diagram

## A diagram of a person with many circles Description automatically generatedUse case

## A computer screen shot of a computer network Description automatically generated with medium confidenceDeployment Diagram

## Sequence DiagramsA diagram of a diagram Description automatically generated

A diagram of a software project

Description automatically generated

## State ChartA screenshot of a computer Description automatically generated