

Software Requirement Specification (SRS) & Model Diagram

December 27, 2025

Prepared By:

Group 03

2112471642, Souvik Pramanik

2121462042, S.M. Riaz Rahman Antu

2132727042, Md Ibrahim Khalil

Course Instructor: Dr. Mohammad Rezwanul Huq

CONTENTS

I	Software Requirements Specifications (SRS)	3
I-A	Introduction	3
	I-A1 Purpose	3
	I-A2 Scope	3
	I-A3 Definitions and Acronyms	3
	I-A4 Overview	3
I-B	Overall Description	3
	I-B1 Product Perspective	3
	I-B2 Product Functions	4
	I-B3 User Characteristics	4
	I-B4 Constraints	4
	I-B5 Assumptions and Dependencies	4
I-C	System Requirements	5
	I-C1 Functional Requirements	5
	I-C2 Non-Functional Requirements	5
	I-C3 Data Requirements	5
	I-C4 Scheduling Constraint Rules	5
I-D	External Interface Requirements	6
	I-D1 User Interface (Conceptual)	6
	I-D2 Hardware Interfaces	6
	I-D3 Software Interfaces	6
	I-D4 Communication Interfaces	6
I-E	Additional Assumptions	6
II	Modeling and Design Diagrams	7
II-A	Use Case Diagram	7
II-B	Use Case Descriptions	8
	II-B1 UC-01: Add Class	8
	II-B2 UC-02: Assign Instructor	8
	II-B3 UC-03: Assign Room	8
	II-B4 UC-04: View Schedule	8
	II-B5 UC-05: Validate Schedule	8
II-C	Sequence Diagram	9
II-D	Activity Diagram	10
II-E	Class Diagram	11
II-F	High-Level Architecture Diagram	12
II-G	UI Wireframe Diagram	13

LIST OF FIGURES

1	Use Case Diagram. (S.M. Riaz Rahman)	7
2	Assign Class & Validation. (Souvik Pramanik)	9
3	Admin Edit. (Souvik Pramanik)	9
4	Create Schedule. (S.M. Riaz Rahman)	10
5	Validation. (S.M. Riaz Rahman)	10
6	Class Diagram. (Souvik Pramanik)	11
7	High Level Architecture Diagram. (Ibrahim Khalil)	12
8	UI Wireframe Diagram. (Ibrahim Khalil)	13

Class Schedule Management System

Abstract—The Class Schedule Management System (CSMS) is a web-based tool designed to assist academic administrators in creating and managing semester-wise class schedules efficiently. It centralizes course, instructor, room, and timeslot information to ensure accurate scheduling with minimal manual effort. The system automatically detects conflicts such as instructor overlaps and room mismatches, helping prevent common scheduling errors. This Software Requirements Specification (SRS) document outlines the system's objectives, functional requirements, constraints, and assumptions that guide its development to support a more organized and reliable scheduling process in educational institutions.

I. SOFTWARE REQUIREMENTS SPECIFICATIONS (SRS)

A. Introduction

1) *Purpose*: The purpose of this Software Requirements Specification (SRS) document is to define and describe the functional and non-functional requirements of the Class Schedule Management System. The platform will be controlled by the administrator to ensure that class scheduling is properly maintained.

2) *Scope*: The Class Schedule Management System is intended for academic administrators who manage semester-wise class schedules. The system will allow:

- Manage courses, sections, instructors, rooms, and time slot patterns.
- Assign instructors and rooms to course sections.
- Validate schedules for conflicts.
- Ensure lab/theory room compatibility.
- Search and view schedules by instructor, room, or timeslot.
- Instructor course load limit: minimum 3 courses (9 credits) and maximum 5 courses (15 credits).
- Assign instructor office hClassroom:ours where no class can be scheduled.

3) Definitions and Acronyms: Definitions

- **Class Schedule Management System** : A software system designed to create, manage, and maintain academic class schedules efficiently.
- **Administrator**: An authorized user responsible for managing courses, instructors, classrooms, time slots, and overall scheduling configuration.
- **Instructor**: A faculty member assigned to teach one or more courses and view assigned class schedules.
- **Student**: A registered user who can view class schedules relevant to their enrolled courses.
- **Course**: An academic subject offered by the institution with a defined syllabus and assigned instructional time.
- **Classroom**: A physical or virtual space allocated for conducting class sessions.
- **Theory Room**: A classroom specifically designated for conducting theory-based courses.
- **Lab Room**: A classroom located in a laboratory building and equipped for practical or lab-based courses.

- **Time Slot**: A predefined time interval during which a class session can be scheduled.
- **Timeslot Pattern**: An encoded pattern defining a combination of class days and a specific time range (e.g., ST1).
 - 1) **ST (Sunday–Tuesday)**: A day grouping representing classes held on Sunday and Tuesday.
 - 2) **MW (Monday–Wednesday)**: A day grouping representing classes held on Monday and Wednesday.
 - 3) **RA (Thursday–Saturday)**: A day grouping representing classes held on Thursday and Saturday.
 - 4) **ST1**: A specific timeslot pattern representing the time range from 8:00 AM to 9:30 AM. It will have 10 minutes of break after each class and classes will end at 5.50 PM.
- **Schedule Conflict**: Any scheduling violation such as overlapping time slots, classroom clashes, instructor overlaps, or mismatches between room type and course type.
- **User Authentication**: The process of verifying a user's identity before granting access to the system.
- **Database**: A structured collection of data used to store, manage, and retrieve scheduling and user information.

Acronyms and Abbreviations

- **CSMS**: Class Schedule Management System
- **SRS**: Software Requirements Specification
- **UI**: User Interface
- **DBMS**: Database Management System
- **ST**: Sunday–Tuesday
- **MW**: Monday–Wednesday
- **RA**: Thursday–Saturday

4) *Overview*: The Class Schedule Management System is a software-based solution developed to automate and streamline the process of creating, managing, and maintaining academic class schedules within an educational institution. This document provides a comprehensive description of the overall system, including its architecture, workflow, and operational scope which is supported by relevant diagrams such as use case, data flow, and entity-relationship diagrams. It can clearly illustrate system interactions and data movement. It defines both functional requirements, such as course scheduling, instructor and classroom allocation, conflict detection, and schedule viewing, as well as non-functional requirements related to system performance, security, usability, and reliability. The document also describes the system's data structures, database design, and user interfaces to ensure accurate data management and ease of use for administrators, faculty, and students. Additionally, constraints in scheduling, such as classroom availability, instructor workload limits, and fixed time slots, as well as system assumptions about the accuracy of data and authorized access, are identified to ensure realistic and efficient schedule generation.

B. Overall Description

1) *Product Perspective*: The Class Schedule Management System is a standalone web-based administrative tool used internally by academic departments. It replaces manual

spreadsheet-based scheduling by enforcing rules, preventing conflicts, and organizing course-related resources. It visualizes the system very smartly and takes less time to organize the schedule. Instructor can view only their schedules by searching.

System Components

Some system components are included here for implementation.

- 1) Course Management Module
- 2) Instructor Management Module
- 3) Timeslot Management Module
- 4) Classroom Management Module
- 5) Schedule Assignment Module
- 6) Conflict Detection Engine
- 7) Schedule Viewing & Search Module
- 8) Office hour Management Module

2) *Product Functions:* The Class Schedule Management System provides academic administrators with a complete set of tools to manage courses, instructors, rooms, timeslots, and semester schedules. The system supports schedule creation, conflict detection, weekly timetable visualization, and office hour management. It ensures that all scheduling rules such as instructor availability, room type matching, and timeslot restrictions are strictly enforced.

1) **Add, edit, delete courses, instructors, rooms, and sections**

This function allows administrators to manage all essential academic entities. They can create new entries, update incorrect information, or remove outdated records.

2) **Assign or update class schedules**

Administrators can select a course section, choose its instructor, timeslot, and room, then create a schedule entry. The system validates all constraints before confirming.

3) **View weekly timetable**

The system provides a structured grid-based representation of the weekly schedule so admins can quickly verify availability and planning.

4) **Search schedules by instructor, room, or timeslot**

Users can filter schedules to find when a specific instructor teaches, what classes take place in a given room, or which classes occur in a timeslot

5) **Review unresolved conflicts**

Administrators can view all pending conflicts such as instructor clashes, room double-booking, or room-type mismatches. This allows them to resolve errors before publishing the final schedule.

6) **Search schedules of office hour**

The system allows admins to view an instructor's office hour blocks and verify that no classes are scheduled during those restricted periods.

3) *User Characteristics:* The Class Schedule Management System is intended to be used by multiple user groups with different roles and levels of technical expertise. The main user types are described below:

Administrators

- Academic and administrative personnel responsible for creating and managing schedules.
- Moderate computer experience, though not necessarily technical experts.
- Require full access to all system functionalities including course setup, instructor assignment, resource allocation, and conflict resolution.
- Need a well-organized interface that supports efficient data entry and updates.

Instructors

- Faculty members responsible for teaching courses.
- Primarily use the system to view assigned schedules, room locations, and time slots.
- May submit availability information or request schedule changes.
- Require limited access with mostly read-only permissions.
- Prefer a lightweight and easy-to-navigate interface.

Students

- End-users who access the system to view course timetables and classroom information.
- Diverse in background and digital literacy levels.
- Expect a simple and responsive interface (both on web and mobile) with clear schedule presentation.
- Read-only access with no privileges to modify data.

4) *Constraints:*

- A class = Course + Section + Timeslot + Instructor + Room
- Lab courses must be placed in lab rooms
- No instructor can teach two classes in the same timeslot
- No room can host more than one class in the same timeslot
- Timeslot patterns are fixed and predefined
- A class repeats for both days in the pattern (e.g., ST = Sun & Tue)

5) *Assumptions and Dependencies:* The Class Schedule Management System operates under the following assumptions and relies on certain external conditions:

- 1) All timeslot definitions, including day patterns and time ranges, will be finalized and provided before the scheduling process begins.
- 2) The accuracy of the schedule depends on the correctness of the information entered for rooms, instructors, and course details; any future changes must be updated manually.
- 3) The academic calendar remains fixed throughout the semester, with no unexpected changes to working days or class durations.
- 4) Internet connectivity is consistently available for accessing the web-based system, especially in multi-user environments.
- 5) Administrators using the system possess basic computer and data-entry skills to properly manage scheduling tasks.

- 6) The institution maintains the necessary server and software infrastructure required for system operation and data storage.
- 7) The availability of instructors and classrooms is confirmed before creating the schedule to avoid additional rescheduling efforts.

C. System Requirements

1) *Functional Requirements:* Functional requirements describe the specific behavior, features, and operations that the system must provide. They define *what* the system should do and specify the tasks, services, or functions it must perform to meet the user needs. In the context of the Class Schedule Management System, functional requirements include course management, instructor assignment, schedule creation, conflict detection, search and view functionalities, and data validation.

Course Management

- FR-1 The system shall allow the administrator to add, edit, and delete courses.
- FR-2 Each course shall include the following fields: Course Code, Course Title, Type (Theory/Lab), Credit, and Department.
- FR-3 The system shall allow defining multiple sections for each course.

Instructor Management

- FR-4 The system shall allow adding, editing, and deleting instructor information.
- FR-5 The system shall allow assigning an instructor to one or more course sections.

Room Management

- FR-6 The system shall store Room Number, Capacity, and Room Type (Theory/Lab).
- FR-7 The system shall prevent lab courses from being assigned to theory rooms.

Timeslot Management

- FR-8 The system shall maintain a predefined list of timeslot codes (e.g., ST1, MW2, RA3).
- FR-9 Each timeslot shall have a defined day pattern and time range.

Schedule Assignment

- FR-10 The system shall allow administrators to assign a schedule entry consisting of Course + Section + Instructor + Timeslot + Room.
- FR-11 The system shall validate the assignment before saving.

Conflict Detection

- FR-12 The system shall detect and prevent the following conflicts:
 - Instructor double-booking
 - Room double-booking
 - Room type mismatch
 - Duplicate class entries
- FR-13 The system shall display detailed error messages for detected conflicts.

Search and View

- FR-14 Administrators shall be able to view schedules by:
 - Instructor
 - Room
 - Timeslot
 - Weekly timetable
- FR-15 The system shall support exporting and printing of timetables (optional).

Data Validation

- FR-16 Course type must match the room type for lab courses.
- FR-17 Empty or inconsistent fields shall trigger validation errors.

2) *Non-Functional Requirements:* Non-functional requirements specify *how* the system performs its functions. They define the quality attributes, constraints, and standards that the system must satisfy. Including the performance, reliability, usability, security, and maintainability. Non-functional requirements ensure that the system operates efficiently, securely, and reliably while providing a satisfactory user experience.

Performance

- NFR-1 Schedule conflict checking shall occur in less than 2 seconds.
- NFR-2 The system shall support at least 300 schedule entries.

Usability

- NFR-3 The user interface shall be simple and intuitive for non-technical users.
- NFR-4 Alerts and warnings shall be clearly visible.

Reliability

- NFR-5 The system shall ensure that incorrect schedules cannot be saved.
- NFR-6 System uptime should be at least 95%.

Security

- NFR-7 Only authorized administrator users can modify schedules.
- NFR-8 All updates shall be logged for audit purposes.

Maintainability

- NFR-9 The codebase shall follow a modular design to facilitate maintenance and updates.

3) Data Requirements: Core Data Entities

- **Course:** Code, Title, Type, Credit, Department
- **Section:** Course Code, Section ID
- **Instructor:** ID, Name, Department
- **Room:** ID, Type, Capacity
- **Timeslot:** Code, Days, Start Time, End Time
- **Schedule Entry:** Course Section, Instructor, Timeslot, Room
- **Office Hour:** Instructor

4) Scheduling Constraint Rules: Timeslot Rules

- Each timeslot code represents both the days and the time range.

- **Example:** ST1 → Sunday–Tuesday, 08:00–09:30; MW2 → Monday–Wednesday, 09:40–11:10

Conflict Rules

- **Instructor Conflict:** An instructor cannot teach two classes at the same timeslot.
- **Room Conflict:** A room cannot host multiple classes at the same timeslot.
- **Room-Type Conflict:** Lab courses must be scheduled only in lab rooms.
- **Duplicate Entry Conflict:** The same course-section cannot be scheduled twice.
- **Office Hour Conflict:** An instructor's office hour cannot coincide with a scheduled class in the same room.

D. External Interface Requirements

1) *User Interface (Conceptual):* The Class Schedule Management System shall provide a user-friendly interface that enables administrators, instructors, and students to interact efficiently with the system. The interface shall include the following components:

- **Dashboard:** Provides a quick overview of courses, instructors, rooms, and schedule summaries.
- **Course Management:** Pages for adding, editing, and deleting courses.
- **Instructor Management:** Pages for adding, editing, and deleting instructor information.
- **Room Management:** Pages for adding, editing, and managing room details.
- **Schedule Assignment:** Page for assigning courses to instructors, rooms, and timeslots.
- **Dropdown Selectors:** Facilitate easy selection of courses, instructors, rooms, and timeslots.
- **Conflict Warnings Panel:** Displays any detected scheduling conflicts in real-time.
- **Weekly Schedule View:** Grid-based visual representation of the complete weekly timetable.
- **Search Functionality:** Allows users to search schedules by instructor, room, or timeslot.
- **Conflict Highlighting:** Visually emphasizes conflicting entries to facilitate resolution.

2) *Hardware Interfaces:* The system shall support standard hardware commonly available in academic environments:

- Desktop or laptop computers with standard display capabilities.
- Mouse and keyboard input devices.

3) *Software Interfaces:* The system shall interact with the following software components:

- Standard web browsers (e.g., Chrome, Firefox, Edge) for client access.
- SQL-based database system for storing schedules, courses, instructors, and room information.

4) *Communication Interfaces:* The system shall support the following communication protocols and network requirements:

- HTTP/HTTPS-based client-server communication for all web interactions.

- Internet or local area network (WLAN) connectivity to support multi-user access.

E. Additional Assumptions

The following assumptions are made to ensure the effective operation of the Class Schedule Management System:

- Academic departments will provide finalized course lists prior to the start of the scheduling process.
- Instructor availability and preferences will be communicated to the system administrators in advance.
- Administrative users possess basic computer literacy and are capable of navigating the system interface.
- All classrooms, resources, and timeslot definitions are available and accurate at the time of scheduling.
- The system will be used within a stable academic calendar, with minimal mid-semester changes.

Conclusion: This Software Requirements Specification (SRS) document defines the complete set of functional and non-functional requirements for the Class Schedule Management System. This document provides a foundation for subsequent phases of the software development lifecycle, including system modeling, detailed design, implementation, testing, and deployment, enabling the creation of an efficient, reliable, and user-friendly scheduling system which can easily maintain by the Admin. Instructors and Student can view the schedule only.

II. MODELING AND DESIGN DIAGRAMS

A. Use Case Diagram

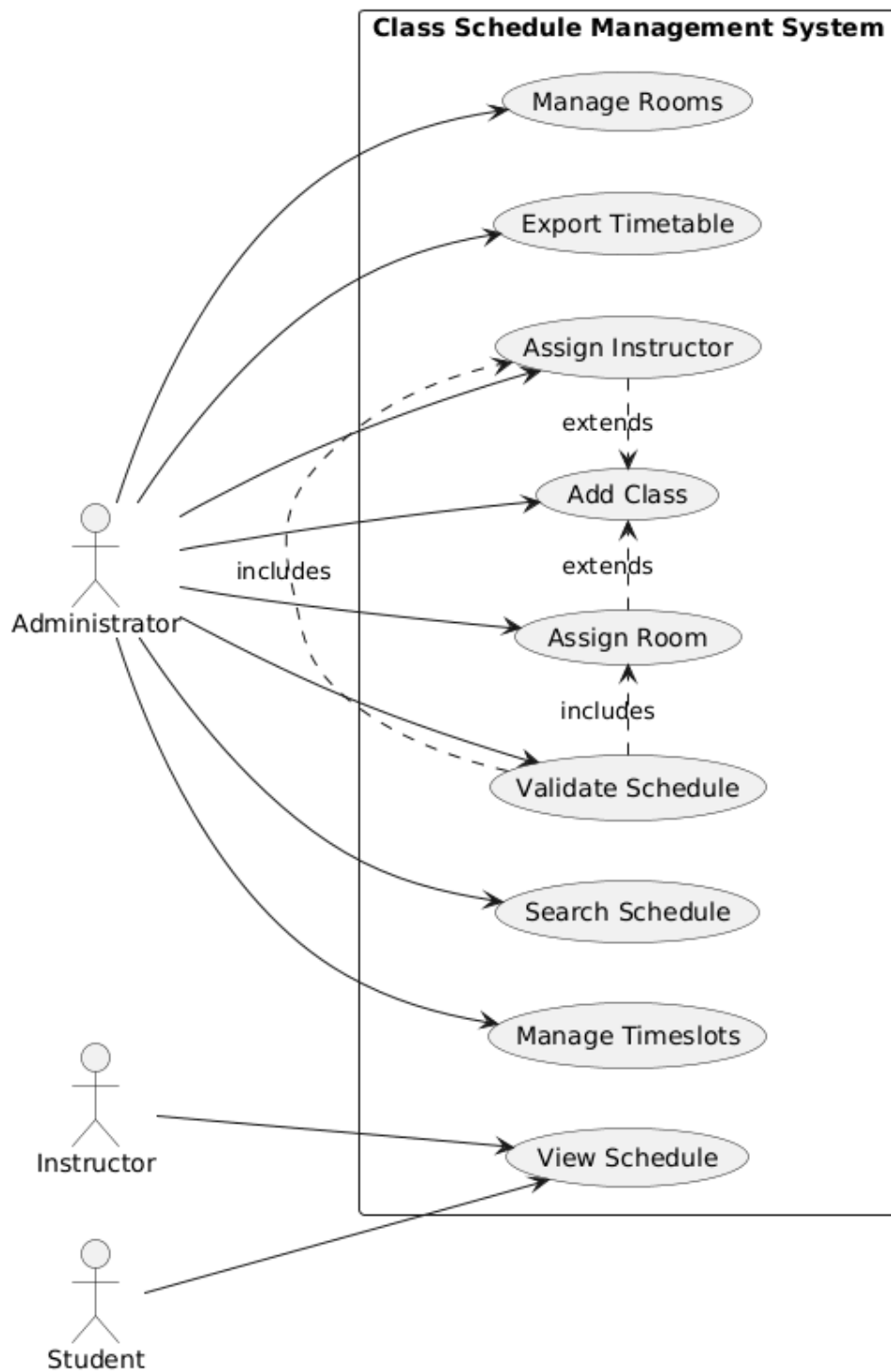


Figure 1. Use Case Diagram. (S.M. Riaz Rahman)

B. Use Case Descriptions

1) UC-01: Add Class: **Actor:** Administrator

Preconditions: Admin is authenticated; Course and Section data are available or being created.

Main Flow:

- 1) Admin opens the "Add Class" page.
- 2) Admin selects a Course Code (or creates a new course).
- 3) Admin enters Section ID, Timeslot, Room, Instructor, and Course Type (Theory/Lab).
- 4) System validates all fields and timeslot patterns.
- 5) System saves the class entry and displays a confirmation.

Alternate Flows:

- A1: Missing mandatory field → System shows a validation error and blocks save.
- A2: Room type mismatch (lab course in theory room) → System rejects entry and suggests a suitable room.

Postconditions: New schedule entry is created and visible in the timetable. Conflict detection flags any conflicts.

2) UC-02: Assign Instructor: **Actor:** Administrator

Preconditions: Instructor exists in the database; course section exists.

Main Flow:

- 1) Admin navigates to the relevant section or schedule entry.
- 2) Admin selects an instructor for the section.
- 3) System checks instructor availability and office hours.
- 4) System saves the assignment (if no conflict) and updates the schedule.

Alternate Flows:

- A1: Instructor double-booked → System shows conflict details and prevents save.

Postconditions: Instructor is assigned to the section or the assignment is blocked due to a conflict.

3) UC-03: Assign Room: **Actor:** Administrator

Preconditions: Rooms and timeslot definitions exist.

Main Flow:

- 1) Admin selects a schedule entry to assign to a room.
- 2) Admin chooses a room from a list (filterable by type/capacity).
- 3) System checks room availability and type match (lab/theory).
- 4) System assigns the room and reflects the change in the weekly timetable.

Alternate Flows:

- A1: Room double-booked → System displays conflict and suggests alternate rooms.

Postconditions: Room is assigned, or admin must select a different room.

4) UC-04: View Schedule: **Actor:** Administrator / Instructor / Student

Preconditions: Schedule entries exist for the relevant term.

Main Flow:

- 1) User selects filter criteria (Instructor / Room / Timeslot / Weekly).

- 2) System retrieves matching schedule entries.

- 3) System displays the weekly grid view; conflicts are highlighted for administrators.

Alternate Flows:

- A1: No results → System shows a "No classes found" message and provides suggestions.

Postconditions: User views the requested timetable.

5) UC-05: Validate Schedule: **Actor:** Administrator (system-triggered)

Preconditions: Schedule assignment operations have been performed.

Main Flow:

- 1) Admin triggers schedule validation or it runs automatically on save.
- 2) Conflict Detection Engine checks: instructor double-booking, room double-booking, room-type mismatch, duplicate entries, office-hour conflicts.
- 3) System lists conflicts with details (instructor, timeslot, room, suggested fixes).
- 4) Admin resolves conflicts (reassign instructor/room or change timeslot).

Alternate Flows:

- A1: Auto-suggested fix accepted → System applies changes and re-validates.

Postconditions: Schedule is validated and either saved as conflict-free or flagged for administrator attention.

Figure 1 is the "Use Case Diagram" of the software. Figure 2 and 3 is the two Sequence Diagrams. Figure 4 and 5 is the two mentioned Activity Diagrams.

The Class Diagram is in Figure 6.

The High Level Architecture is present in Figure 7.

And the Figure 8 is the UI Wireframe Diagram.

C. Sequence Diagram

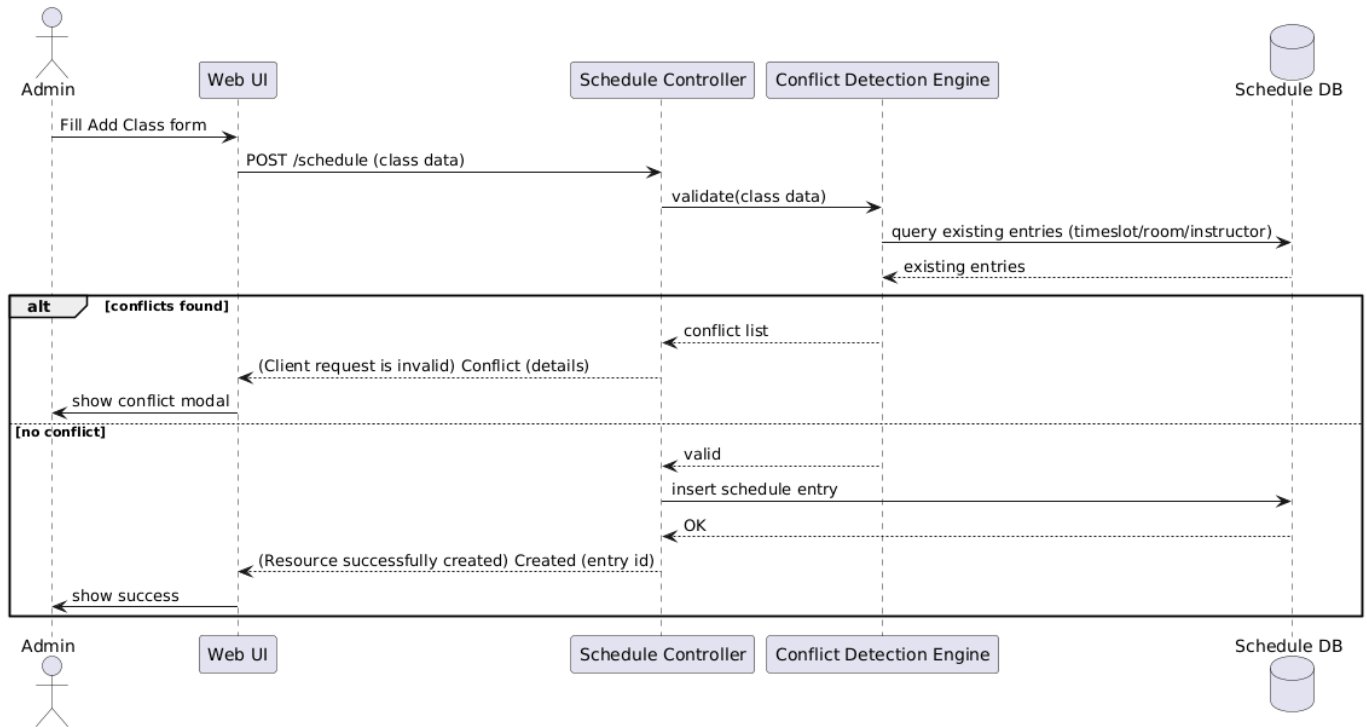


Figure 2. Assign Class & Validation. (Souvik Pramanik)

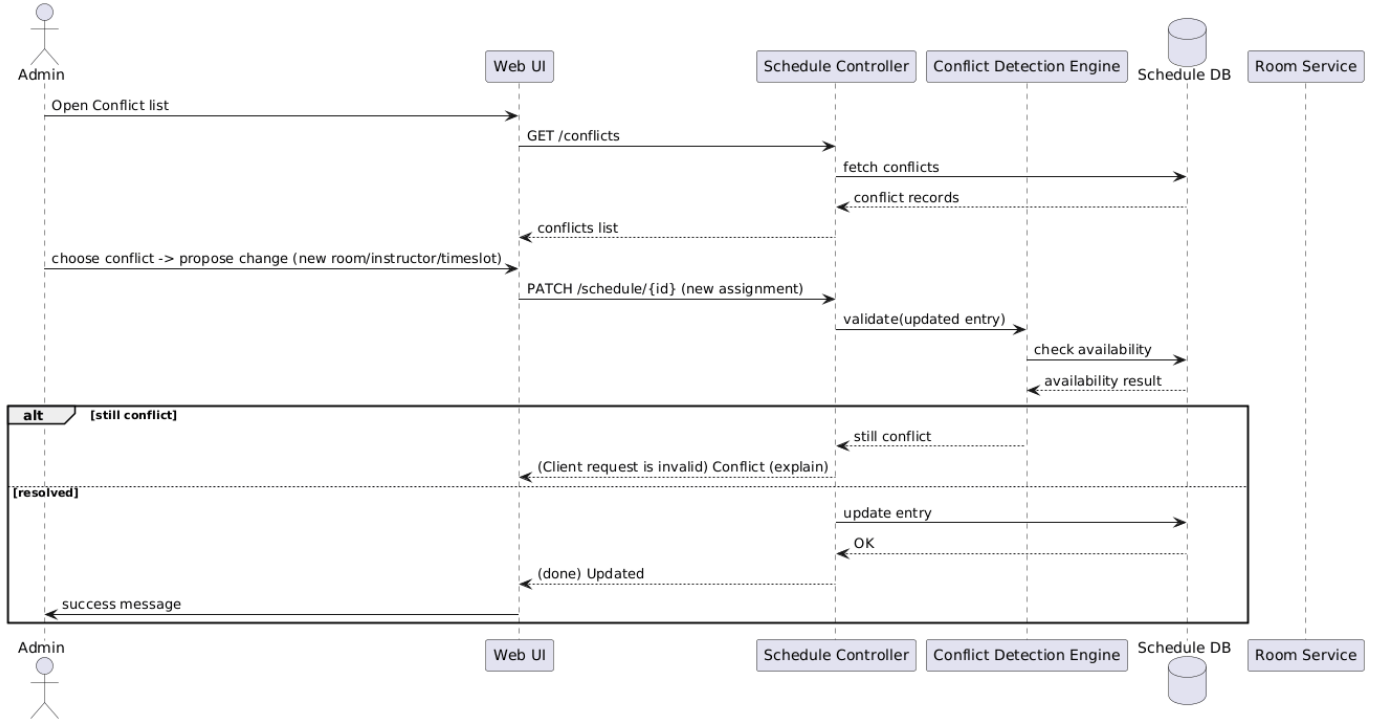


Figure 3. Admin Edit. (Souvik Pramanik)

D. Activity Diagram

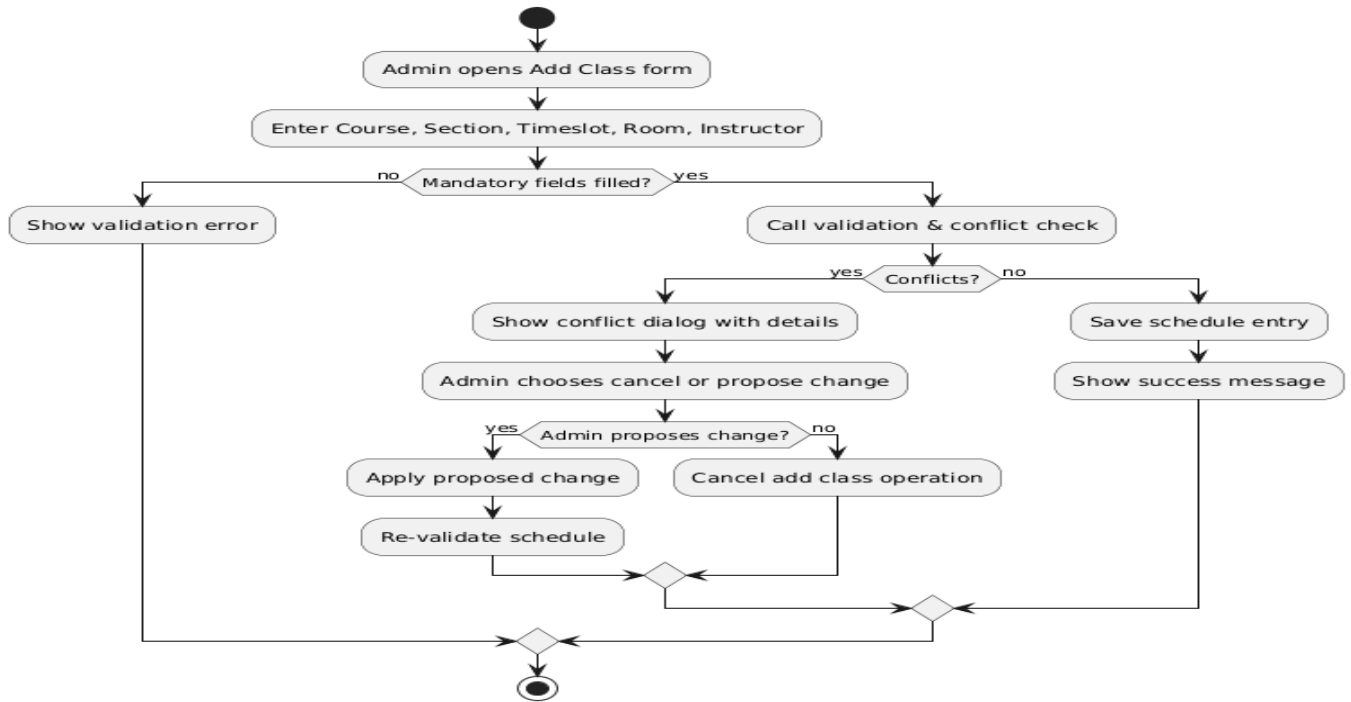


Figure 4. Create Schedule. (S.M. Riaz Rahman)

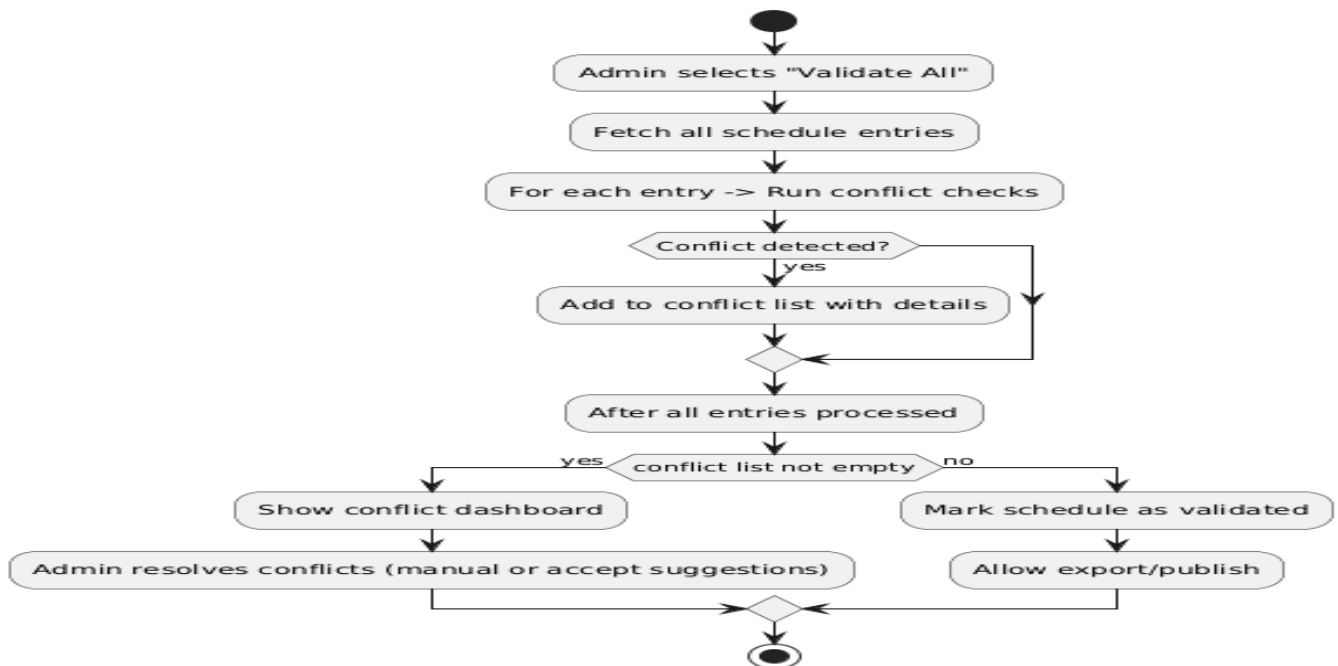


Figure 5. Validation. (S.M. Riaz Rahman)

E. Class Diagram

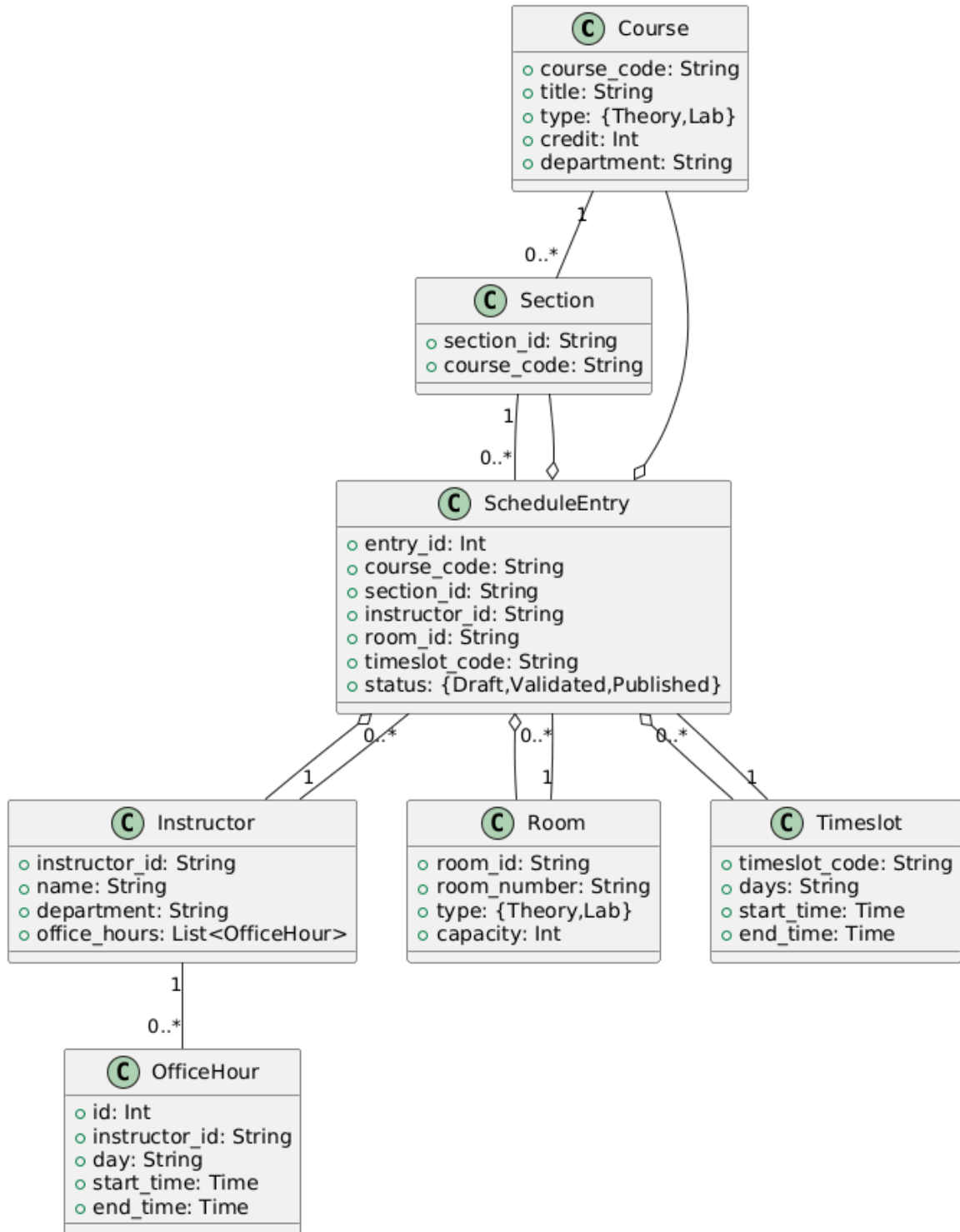


Figure 6. Class Diagram. (Souvik Pramanik)

F. High-Level Architecture Diagram

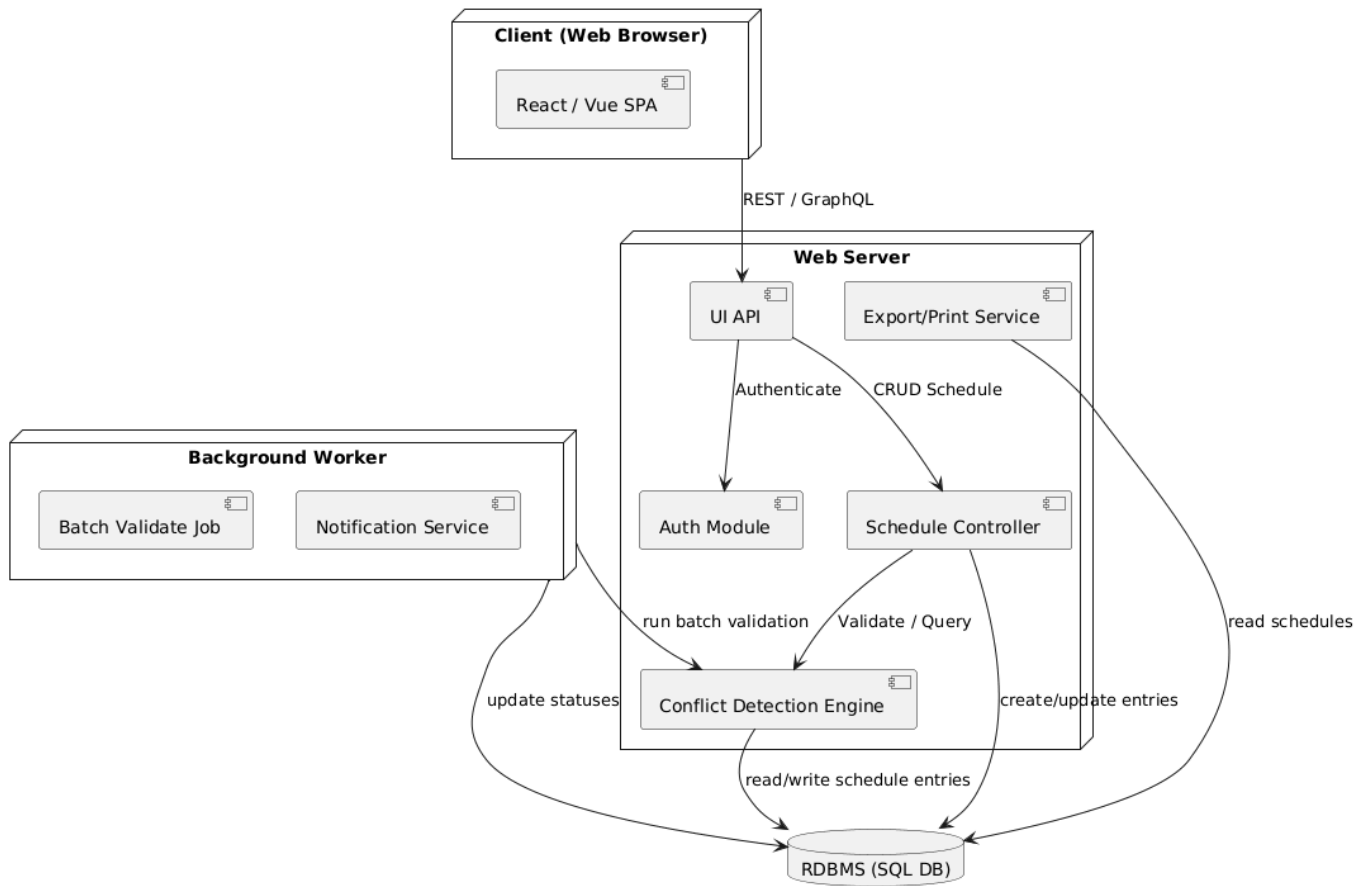


Figure 7. High Level Architecture Diagram. (Ibrahim Khalil)

G. UI Wireframe Diagram



Figure 8. UI Wireframe Diagram. (Ibrahim Khalil)