FUNCTIONAL SPECIFICATION

Smart Scheduler and Reminder System

Author: Changyu Jia

Student ID: C00292876

Programme: SETU — BSc Software Development (Year 4)

Supervisor: Dr Jamal Tauseef

Date: 1/10/2025

Abstract

This document is the functional specification of the "Intelligent Scheduling and Reminders System", which is a web-based application designed to address the inefficiency of scheduling in educational environments. It goes beyond general calendar tools and is an intelligent interactive center serving administrators, teachers, and students. Its core objective is to reduce missed appointments and simplify communication through automated multi-channel reminders and notifications.

Key features include a robust user role management system, comprehensive schedule and activity management supported by semester-based timetable imports, and a unique two-way calendar visibility system. This system enables students and teachers to view each other's availability, overcoming the limitations of traditional one-way calendar sharing. Other features integrate personal to-do management with the official timetable and provide activity participation tracking to generate on-time rate reports. By customizing its design for academic workflow, this platform aims to enhance schedule transparency, simplify meeting coordination, and provide valuable user engagement insights, ultimately saving time and reducing management costs.

CONTENTS

1. Introduction

This document outlines the functional specifications for the Smart Scheduler & Reminder System, a web-based application designed for educational institutions to manage schedules, events, and tasks efficiently through automated reminders and notifications. The platform moves beyond simple calendar management to become an intelligent engagement hub that reduces missed appointments, simplifies meeting scheduling, and provides valuable insights into user participation.

2. Project Scope

2.1 Problem Statement

Educational institutions and community groups currently rely on a set of generic tools—such as Google Calendar and Notion—for schedule management and communication. While these tools offer general functionality, they are not designed for the specific workflows of academic environments. These solutions either lack direct support for term timetable imports or are not tuned to the roles and workflows of administrators, teachers and students; as a result schedule updates and reminders are often delayed or sent inconsistently, causing missed classes and extra administrative work. Students also face fragmented personal task management, with to-dos and deadlines separated from official timetables.

2.2 Project Objectives

This system is tailored to the specific needs of academic environments, supporting complex scheduling scenarios such as term-based timetables and role-based permissions. Help users reduce missed events and communication gaps by multi-channel reminders while offering simple personal task management for measuring engagement and punctuality.

2.3Target Audience

The system targets educational institutions (schools, colleges, and universities) and their users. It serves administrators, instructors, and students who need to organize academic schedules, communicate changes, and track tasks. The application could also be used by general groups seeking a structured scheduling and reminder tool.

System Context Diagram

It provides a high-level overview of the system's boundaries, showing how user roles communicate with the core scheduling, reminder, and database components:

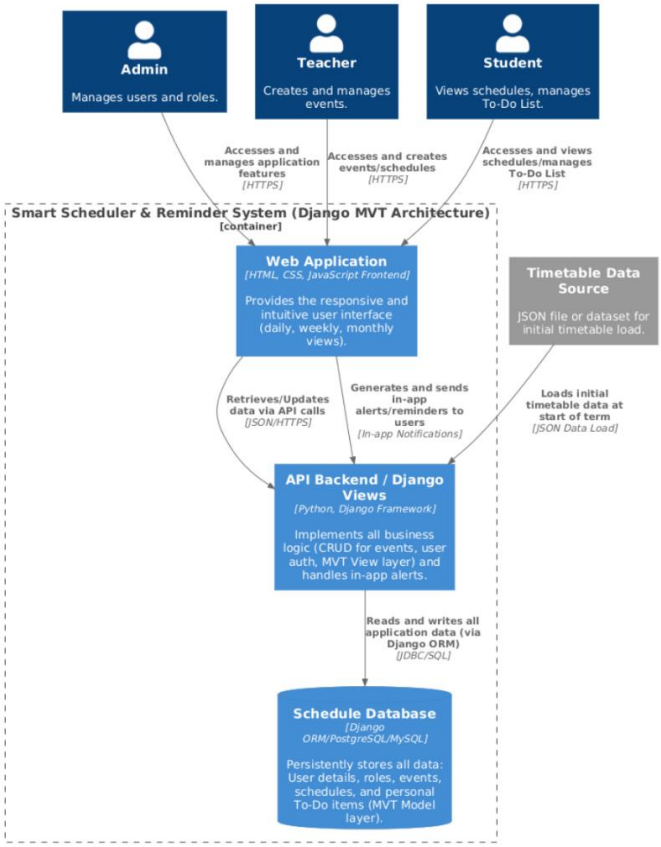

Figure 2.3 System Context Diagram

3. User Roles

- **Admin**: To have control of everything such as to view all users and change their roles.
- **Teacher**: Creates and edits events, sends notifications, manages participant lists.
- **Student**: Views schedules, receives reminders, manages personal to-do lists.

4. Requirements

4.1 Core Features and Additional Features

Core Features:

User Role Management: Secure login and role-based access control (Admin, Teacher, Student) with permission levels for viewing, creating, and editing events[1].

Schedule and Event Management: Ability to create and delete events with details such as title, date, time, location, description, and participant list. Support for daily, weekly, and monthly calendar views. At the start of term or year, the timetable is scheduled based on json file or any data set, the app loads the daily timetable automatically and allows changes by admin.

Two-Way Calendar Visibility System: Students and teachers can view each other's courses, free time slots, and leave arrangements, avoiding time conflicts. This system overcomes the limitation of traditional calendars (such as Google Calendar) that only support one-way sharing, enhancing the schedule transparency in academic scenarios.

Reminders and Notifications: System-generated reminders via email and in-app alerts for upcoming events, changes, or cancellations, any updates are notified to both students and teachers.

To-Do List Integration: Personal task management with fields for title, estimated time and status (pending/completed).

Event Participation Tracking and Reporting:Description: The system tracks whether an event was attended or missed. Users can generate personal reports showing metrics like "Events Completed," "Events Missed," and overall punctuality.

Suggestions Function: Based on popular events or functions , the app will recommended tasks and events.

Additional Features:

AI implementation: This application can integrate artificial intelligence technology to provide time planning suggestions.

4.2Usability

Intuitive Navigation: Clear menu structure with labels

Help Section: Tooltips for first-time users.

4.3Security

Access control: Role-based access to prevent unauthorized actions.

5. Diagram

5.1Sequence diagram

When teachers or administrators create or cancel events, the system will automatically trigger a reliable multi-channel notification process to ensure that students receive all schedule change information.

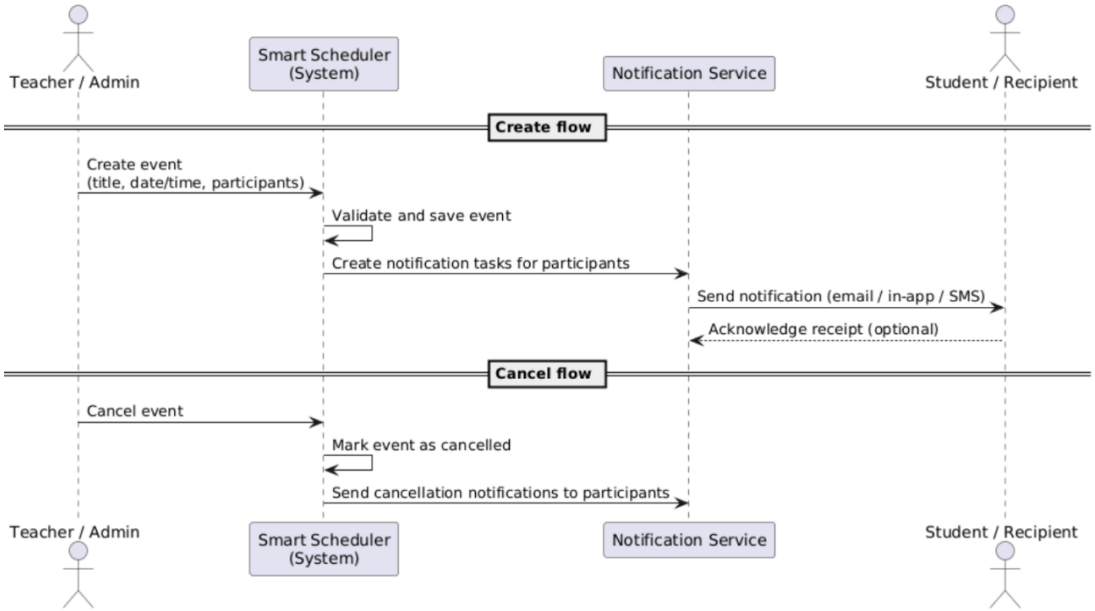

Figure 5.1 Sequence Diagram

5.2 Use Cases

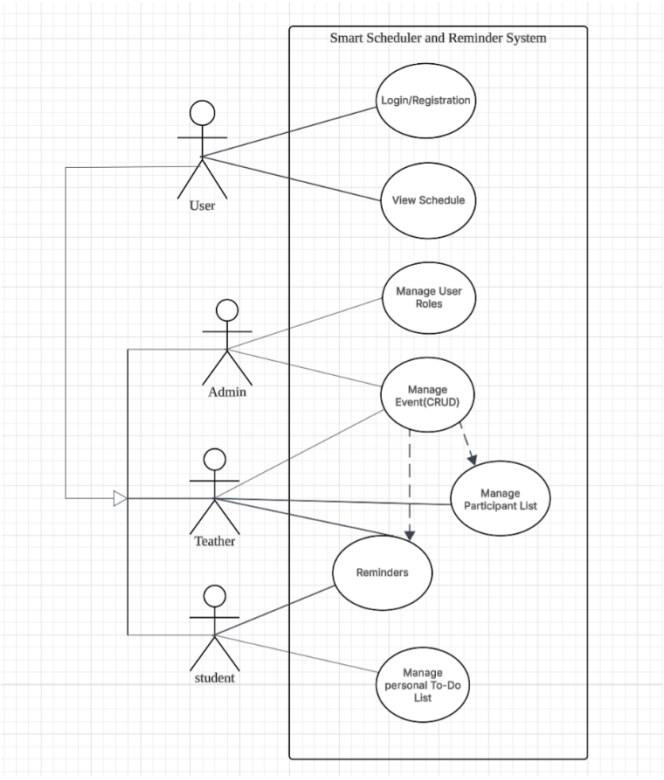

Figure 5.3 Use Cases

5.3 Brief Use case

Daily timetiable web app for any course/class.

At the start of term or year, the timetable is scheduled based on json file or any data set, the app loads the daily timetable automatically and allows changes by admin. Any updates are notified to both students and teachers.

Name	Login/Registration
Actor(s)	User (Admin, Teacher, Student)
Description	The User securely logs in using credentials or registers to gain role-based access to the system features, ensuring access control.

Name	View Schedule
Actor(s)	Admin, Teacher, Student
Description	The User views the calendar to see all scheduled events and the daily timetable, with support for daily, weekly, and monthly views, which is automatically loaded based on a JSON file or dataset at the start of the term.

Name	Manage User Roles
Actor(s)	Admin
Description	The Admin views all users, changes their roles, and manages permission levels, having control of everything in the system.

Name	Manage Events (CRUD)
Actor(s)	Admin, Teacher
Description	The Teacher or Admin creates, edits, or deletes events with details like title, date, time, location, and description, with Admin having the ability to allow changes to the loaded daily timetable.

Name	Manage Personal To-Do List
Actor(s)	Student
Description	The Student manages personal tasks by adding titles, estimated times, and setting the status (pending/completed).

Name	Reminders
Actor(s)	Teacher, Student
Description	The System automatically sends system-generated reminders and notifications via email or in-app alerts for upcoming events, changes, or cancellations, ensuring timely updates for both students and teachers.

5.4Detailed Use case

Use Case Name	Create a New Event
Actor(s)	Admin, Teacher
Description	A Teacher or Admin creates a new event (e.g., a class, meeting, or workshop) by filling in its details. The system saves the event and automatically notifies all participants.
Preconditions	1. The Actor (Teacher/Admin) must be successfully logged into the system. 2. The Actor must have the necessary permissions to create events (inherent to their role).
Main Success Scenario	1. The Actor navigates to the calendar view and selects the option to "Create New Event". 2. The system displays a form for entering event details. 3. The Actor fills in the required fields: Event Title, Date, Start Time, and End Time. 4. The Actor clicks the "Save" or "Create Event" button. 5. The system validates the entered data (e.g., ensures end time is after start time).

	6. The system saves the new event to the database. 7. The system triggers the Automated Event Notification use case, generating in-app alerts for participants. 8. The system displays a confirmation message to the Actor (e.g., "Event created successfully") and updates the calendar view to include the new event.
Alternative Flows	4a. Validation Error: ● At step 5, if the system finds invalid data (e.g., a time in the past), it displays an error message highlighting the problematic field(s) and does not save the event. The Actor can correct the data and resubmit the form. 4c. Delete Existing Event: ● The Actor selects an existing event and chooses "Delete". ● The system requests confirmation for deletion. ● The Actor confirms. ● The system removes the event from the database and triggers notifications to inform participants of the cancellation.
Postconditions	1. A new event is persistently stored in the system's database. 2. All selected participants have been notified via their configured channels. 3. The event is visible on the calendars of the creator and all participants.

6. Metrics

- **Accuracy:** Based on the event verification and automatic notification process in the document, the system should ensure that the delivery accuracy of event reminders is no less than 90%.

- **Usability**: New users should be able to complete the initial login and core operations (such as viewing the schedule or managing the to-do list) within an average of 5 minutes, relying on the intuitive navigation design and help tools in the documentation.

7. References

[1]Sandhu, R., Coyne, E., Feinstein, H., & Youman, C. (1996). Role-Based Access Control Models. IEEE Computer, 29(2), 38-47.