Class Planning and Management System – Initial Problem Description, Solution and Plan

Medet Murzakhanov

Introduction

Class timetabling is a critical activity within a university setting, requiring the management of multiple constraints such as student experience, staff flexibility, and space availability. An effective scheduling system must balance these constraints while remaining intuitive for the end-users.

My objective with this project is to develop a Class Planning and Management System with a focus on user experience and constraint optimization. The system will provide a modern, web-based interface to manage class schedules efficiently, with a strong emphasis on constraint visibility and conflict resolution. This document reflects the progress I made in first semester, Throughout the first semester, my focus has been primarily on frontend development, where I've worked to create a functional and accessible interface using. This document outlines the progress made so far and the next phases project.

Problem Description

The Class Planning and Management System should address key challenges university face with timetabling by balancing multiple constraints effectively while ensuring the frontend is user-friendly, clear, and accessible. Effective scheduling directly impacts student success, faculty satisfaction, and institutional efficiency, making it essential to develop a system capable of balancing competing needs while ensuring clarity and fairness.

Who:

- Students: I believe students should have a system where they can quickly check their schedules without needing to navigate a complicated platform. If conflicts appear, students should be able to see them instantly, to avoid missing lectures or make last-minute adjustments.
- Staff: Staff need flexibility, especially since teaching loads, research responsibilities, and personal availability vary so much. Scheduling conflicts often result in staff burnout, inconsistent teaching loads, and poor work-life balance. The system should prevent that by giving staff control over preferences while balancing institutional requirements.
- Administrators: Administrators to manage the logistical aspects of scheduling, including room/location/building allocations, managing constraints directly and etc. The lack of conflict

detection mechanisms in existing systems often results in double bookings, last-minute changes, and increased administrative overhead (ScienceDirect, 2024).

What:

- Complex Constraint Management: University scheduling involves balancing constraints (e.g., room capacity, student preferences etc.) into hard and soft. Modern systems often fail to manage both effectively, resulting in unresolved conflicts.
- Resource Underutilization: Poor scheduling often leads to wasted classroom space or over-filled classes, highlighting the importance of balanced resource allocation.
- Limited Flexibility: If last-minute changes like staff absences occur, manual adjustments often take too long. System should provide automated checks to catch these issues before they disrupt events.

When:

- Peak Periods: These issues are most evident during semester planning phases when the most of scheduling activities, such as course registration and staff assignments, take place.
- Ongoing Impact: Problems persist throughout the semester when staff leaves, room reassignments, and class cancellations occur, causing further disruptions in timetable.

Where:

- Since many institutions now offer remote and hybrid learning, I've realized it's important for the system to handle both physical rooms and virtual class links seamlessly. Design should be able to manage both without causing confusion

Why:

- Lack of Optimization Algorithms: Modern tools rarely leverage algorithms like backtracking or genetic algorithms for efficient constraint management.
- Data Disorganization: Poor data handling structures lead to redundant information and scheduling errors.

Goals and Requirements

Since the initial submission, the goals and requirements needed to shift as I progressed, to better address the current challenges identified during the development process and feedback received.

- **Enhanced User Interface:**
    - Continue refining the frontend for accessibility, clarity, and ease of use.
    - Begin backend development to support data management and constraint storage.
    - Improve user interaction through enhanced visual elements and simplified workflows.
- **Real-Time Conflict Detection:**
    - Implement real-time visual markers for conflicts such as overlapping lectures and room clashes (and more).
    - Expand conflict validation to cover complex scheduling constraints using backend algorithms.
    - Ensure conflict alerts are instant and clear during the scheduling process.
- **Multi-User Role Management:**
    - Support role-based access control for students, staff, and administrators, each with distinct permissions and views.
- **Scalable Architecture:**
    - Ensure the system is API-ready for future integrations with Canvas, Microsoft Teams, and other platforms.
    - Design the architecture for easy expansion with constraint-solving algorithms like backtracking and genetic algorithms.
- **Frontend Development:**
    - Enable drag-and-drop functionality for manual schedule adjustments.
    - Incorporate visual conflict detection markers directly in the timetable page.
    - Develop distinct user views for students, faculty, and administrators.
- **Data Handling:**
    - Current: Use temporary data for testing and frontend validation.
    - Planned: Implement PostgreSQL for persistent storage and structured data management.
    - Ensure data security and efficient data retrieval for large datasets.
- **Accessibility:**
    - Ensure full WCAG 2.1 compliance for usability, scan using Lighthouse extension.
    - Enhance to enhance code, add keyboard navigation, screen reader support, and color contrast.

- **User-Configurable Notifications:**
    - Provide customizable notifications for schedule updates, including:
        - Class time changes.
        - Conflict alerts.
        - Reminders for upcoming classes.
    - Allow users to personalize notification preferences for frequency and delivery method

Success Criteria

The following success metrics will be used to evaluate the effectiveness of the CPMS:

- Frontend Completion: A fully functional interface capable of handling core scheduling tasks.
- Conflict Detection: The system should detect and visually represent scheduling conflicts accurately.
- User Testing Feedback: A minimum satisfaction score of 4/5 from user feedback surveys.
- Testing: Completion of unit, integration, and user acceptance tests.

Expected Project Development Plan

Phase	Task Completed/Planned	Timeline
Research planning and Requirement Gathering	Requirement gathering, tool selection, initial scoping	Completed
System Design	Completed initial wireframes, system architecture, planned further changes to be added	Completed
Frontend Development	Built a basic frontend interface with foundational layout. Role Based control and real time conflict detection to be added during work on backend	Week 1-3 Semester 2
Backend Development	To set up PostgreSQL for data storage. Create data schemas for students, staff, courses, and constraints. Develop essential API endpoints for data retrieval	January – February 2025

	and synchronization between frontend and database.	
Constraint Implementation	Develop backtracking and genetic algorithms to handle hard violations (e.g., room capacity) and soft violations (e.g., back-to-back classes). API layer connecting with constraint engine	February 2025
System Testing and Optimisation	Unit testing, integration testing, system validation, resolving constraints, user feedback cycle	March 2025
Refinement & Optimization	Performance optimization, refining UI, resolving bugs based on user feedback	Early April 2025
Final Prototype & Demonstration	Final system presentation, gathering final feedback, making last adjustments	April 2025 (Before 28 April 2025)

References

- ScienceDirect (2024). A survey of university course timetabling and examination timetabling problems. *Operations Research Perspectives*. Retrieved from: https://www.sciencedirect.com/science/article/pii/S0167923624001787
- Hanover Research (January 2018). Best Practices in Course Scheduling. Retrieved from: https://www.sfasu.edu/docs/envisioned/hanover-research-best-practices-in-course-scheduling.pdf