1. Real-World Domain (Student Course Registration System)

Universities use course registration systems to help students enroll in courses each semester. Students log into an online portal, select courses, and the system checks several conditions including prerequisites, seat availability, timetable conflicts, and waitlists when sections are full.

1. How Universities Register Students

Students register online. Their details (ID, semester, completed courses) already exist in the system. When selecting a course, the system:

- checks prerequisites
- checks section seat availability
- enrolls the student if seats are open
- adds them to a waitlist if the section is full

2. What Are Prerequisites?

Prerequisites are courses that must be completed before taking more advanced courses. Example: OOP is a prerequisite of DSA.

3. What Is a Waitlist?

If a section is full, the student is placed in a FIFO queue. When an enrolled student drops the course, the first student in the waitlist is automatically enrolled.

4. What Is Section Capacity?

The number of seats available in a course section (e.g., DSA-A: 40 seats). After capacity is reached, all additional students are waitlisted.

5. What Is Timetable Conflict Checking?

A conflict occurs when two classes overlap. Example:
"You cannot add this course due to a clash with CS200 on Monday 10–11 AM."
The system compares the requested section time with the student's existing schedule.

6. How Seat Allocation Works

Universities use FIFO. Earlier registration attempts have priority. If the section is full, the student is placed in the waitlist queue.

7. Why FIFO Is Needed

FIFO ensures fairness. Using a stack (LIFO) would be unfair. A queue ensures the earliest student gets the seat first.

8. How Section Balancing Is Done

Used to distribute students evenly across multiple sections.
Real systems apply:

- capacity limits
- prerequisite checks
- time conflict prevention

2. Functional Requirements

- The system must allow students to view all available courses and sections.
- The system must allow registration only after checking prerequisites.
- The system must detect timetable clashes and block conflicting courses.
- The system must store students, courses, prerequisites, timings, and seats.
- If the section is full, the student must be added to a waitlist.
- When a student drops, the first waitlisted student must be given the seat.
- The system must balance sections when one section is crowded and others have empty seats.
- The system must show clear reasons when registration fails.

- Reports must be generated for:
    - enrolled students
    - waitlisted students
    - available seats

High-Level Architecture (HLA)

1. CourseGraph Module (Prerequisite Management)

- Stores courses and their prerequisites.
- Checks if a student has completed required courses before enrolling in an advanced course.
- Provides prerequisite chains to the Registrar for validation.
- Ensures students cannot register for courses without completing required prerequisites.

2. Registrar Module (Registration & Conflict Checking)

- Handles student registration requests.
- Verifies prerequisites with the CourseGraph.
- Checks timetable conflicts to prevent overlapping courses.
- Coordinates with SectionManager to enroll students or add them to waitlists.
- Manages registration, ensures rules are followed, and informs students why registration fails if necessary.

3. SectionManager Module (Seats, Waitlists, and Balancing)

- Manages course sections and seat availability.
- Adds students to sections if seats are available.
- Adds students to FIFO waitlists if sections are full.
- Automatically promotes students from waitlists when seats open.
- Balances student distribution across multiple sections if one section is crowded.
- Ensures fair seat allocation and smooth section management.

4. Reports Module

- Generates reports for enrolled students, waitlists, and available seats.
- Provides read-only access for monitoring.

4. Chosen Data Structures

Directed Graph / DAG

- Stores course prerequisites.
- DFS/BFS traversal for prerequisite validation.
- Complexity: O(V + E).

Hash Maps

- Fast lookup for student registrations and course-section mappings.
- Ensures constant-time access.

Queues (FIFO)

- Used for waitlists.
- Ensures first-come-first-served fairness.

Trees / Heaps

- Used for priority-based seat allocation.
- Supports dynamic section balancing.

Arrays

- Used for timetable grids.
- Each timeslot mapped to a fixed index for quick clash detection.

5. Project Plan

Project Overview

System will support:

- DAG-based prerequisite checking
- timetable conflict detection
- seat allocation with hashing
- waitlist queues
- section balancing using trees/heaps

Objectives

- Efficient registration system
- DAG validation
- Time Table conflict checking
- fast hashing-based lookups
- FIFO waitlists
- balanced sections
- standard reporting

Scope

Included: registration, prereqs, conflict detection, waitlists, balancing, reporting, CLI-based system
Excluded: payments, instructor management, UI, transcripts, dashboards

Milestones

Milestone 1:

- Implement DAG
- Build hash maps

Milestone 2:

- Implement queue, tree/heap, timetable array

Milestone 3:

- CourseGraph module
- Registrar module

- SectionManager module

Milestone 4:

- Integrate all modules
- Test full registration flow

Milestone 5:

- Unit tests
- Integration tests
- Complexity analysis

Milestone 6:

- Documentation
- Final report
- Presentation