Automated Time-Table

Scheduler

1. Use Case diagram
2. Activity Diagram
3. Class diagrams
4. State Diagram
5. SRS

# SRS

1. Introduction
   1. Purpose
      1. Automate scheduling for 5-day timetables (8:45 AM – 4:30 PM) with 45-minute sessions.
      2. Allocate subjects based on teacher availability and required session counts.
   2. Product Scope
      1. Web-based scheduler for university timetable managers.
      2. Accepts inputs on teachers, subjects, and required session frequencies.
      3. Outputs schedule in Excel format.
   3. Intended Audience
      1. University timetable managers and academic administrators.
2. Overall Description
   1. Product Perspective
      1. Standalone web application.
      2. Inputs provided by the manager (subject list, teacher counts, session frequency).
      3. Excel export for schedule distribution.
   2. Product Features
      1. Input form for teacher availability and subject details.
      2. Configurable session frequency per subject (e.g., Cloud Computing: 6 times/week).
      3. Intelligent scheduling algorithm (supports combined sessions, avoids >2 consecutive sessions).
      4. Excel output with timeslot allocations.
   3. User Classes and Characteristics
      1. Timetable managers and academic staff with basic tech proficiency.
      2. Users familiar with the university's subject structure.
   4. Operating Environment
      1. Accessible via modern web browsers on desktops and laptops.
   5. Design Implementation
      1. Frontend: HTML, CSS, JavaScript.
      2. Backend: Java-based scheduling logic.
      3. Excel generation library for output.
   6. Design Constraints
      1. Must account for teacher availability (multiple teachers per subject possible).
      2. Adhere to session frequency inputs; no more than 3 sessions grouped consecutively.
   7. Assumptions and Dependencies
      1. Accurate input data (subjects, teachers, session requirements) from the manager.
      2. University provides the official subject list.
      3. Reliable web and internet environment.
3. External Interface Requirement
   1. User Interface
      1. Clean, simple, and user-friendly website.
      2. Forms for inputting teacher availability, subject names, and session counts.
      3. Visual display of generated timetable.
   2. Software Interface
      1. Backend API for processing scheduling logic.
      2. Integration with an Excel generation tool/library.
   3. External Interface
      1. Minimal external systems; standalone tool for university use.
   4. Communication Interface
      1. HTTP/HTTPS protocol for secure web communication.
      2. Standard data exchange formats (JSON for API, XLS/XLSX for Excel output).
4. Functional Features
   1. Input teacher availability and subject details.
   2. Configure the number of sessions required per subject.
   3. Run scheduling algorithm to assign subjects to timeslots.
   4. Group sessions as allowed (single or two consecutive sessions; never three).
   5. Export the final schedule to Excel with designated time and subject slots.
5. Non-Functional Features
   1. Performance:
      1. Fast schedule computation and Excel export.
   2. Usability:
      1. Intuitive and straightforward website interface.
   3. Reliability:
      1. Consistent schedule generation with minimal errors.
   4. Security:
      1. Secure data handling via HTTPS and safe input validation.
   5. Scalability:
      1. Ability to manage an increasing number of subjects and teacher inputs.
   6. Maintainability:
      1. Modular code design for easy updates and feature enhancements.