RAJALAKSHMI ENGINEERING COLLEGE

RAJALAKSHMI NAGAR, THANDALAM – 602 105

Laboratory Record Note Book

Name: . . . . . .

Year / Branch / Section: . . . . . .

University Register No. : . . . . . .

College Roll No. : . . . . . .

Semester: . . . . . .

Academic Year: . . . . . .

RAJALAKSHMI ENGINEERING COLLEGE

RAJALAKSHMI NAGAR, THANDALAM – 602 105

BONAFIDE CERTIFICATE

Name:_____

Academic Year:_____ Semester : _____Branch :_____

Register No:

Certified that this is the bonafide record of work done by the above student

in the CS19442 – *Software Engineering Laboratory during the year 2023-2024*

Signature of Faculty in-charge

Submitted for the Practical Examination held on .....

Internal Examiner External Examiner

A REPORT ON PROJECT AUTOMATIC TIMETABLE GENERATION SYSTEM SOFTWARE

Submitted by

Barath Raj	220701038
Bhargavi	220701044
Chandeep Roshen	220701050
Deeksha	220701056
Dhanyaa Vanisha	220701062
Saran Raj	220701510

INDEX

AUTOMATIC TIMETABLE GENERATION SYSTEM

OVERVIEW OF THE PROJECT

Managing and optimizing schedules is a significant challenge for educational institutions, with many suffering from inefficiencies and resource misallocation. Research indicates that 60% of institutions experience scheduling conflicts, and 50% struggle with resource allocation. Traditional timetable creation is time-consuming, error-prone, and inflexible. To address these issues, an automated timetable generation system is essential.

The project aims to resolve inefficient schedule management, resource misallocation, inflexibility, and time consumption. It collects data on courses, instructors, classrooms, and constraints, processes this data using optimization algorithms, and detects conflicts to provide efficient schedules. Data visualization tools highlight conflicts and resource allocation.

Implementing this system will enhance efficiency by automating timetable creation, reduce conflicts, and optimize resource allocation. It offers flexibility to adapt to changes, improves satisfaction for students and instructors, and provides scalable, data-driven solutions for educational institutions.

Software Requirements Specifications

EXP_NO: 1 **DATE: 20-02-24**

Introduction

An automatic timetable management system is a software solution designed to streamline the process of creating and managing schedules for various activities or resources. Whether for schools, universities, businesses, or other organizations, this system aims to simplify the often complex task of organizing timetables efficiently.

1.1 Purpose

The purpose of this document is to outline the requirements for the development of the Automatic Timetable Generation System for College. It serves as a guide for the development team and stakeholders to understand the system's functionalities, constraints, and user interactions.

1.2 Scope

The system will facilitate the generation of timetables for courses, classrooms, and faculty members within the college. It will consider various constraints such as room availability, faculty preferences, and student course loads. The system will provide an intuitive interface for administrators to input data and generate optimized timetables.

2. Overall Description

2.1 Product Perspective

The Automatic Timetable Generation System for College will operate as a standalone application integrated with the college's existing infrastructure. It will interact with databases containing information about courses, faculty, classrooms, and student schedules.

2.2 Product Functions

- Create and manage course schedules
- Allocate classrooms and resources based on availability
- Consider faculty preferences and availability

- Resolve conflicts and optimize timetables
- Provide notifications for changes or updates

2.3 User Classes and Characteristics • **Administrators**: Responsible for managing course schedules and resources.
- **Faculty**: Provide input on their availability and preferences.
- **Students**: View and access their class schedules.

2.4 Operating Environment

The system will be web-based and accessible through modern web browsers. It will require a stable internet connection and compatibility with common operating systems (Windows, macOS, Linux).

3. Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces: The system will have an intuitive Graphical User Interface (GUI) for administrators, faculty, and students. It will include forms for data input, interactive calendars for scheduling, and notifications for updates.

3.1.2 Hardware Interfaces: The system will require standard computing hardware, including servers for hosting the application and client devices (computers, tablets, smartphones) for user access.

3.1.3 Software Interfaces: The system will interact with databases containing information about courses, faculty, classrooms, and student schedules. It will use web technologies for frontend development and APIs for backend integration.

3.1.4 Communication Interfaces: The system will communicate with users via email notifications and in-app messaging for important updates and announcements.

3.2 Functional Requirements

3.2.1 Authentication and Authorization:

- Users must authenticate before accessing the system.
- Different levels of access (admin, faculty, student) will be defined with appropriate permissions.

3.2.2 Course Management:

- Admins can create, update, and delete course schedules.
- Courses will have attributes such as name, code, duration, and prerequisites.

3.2.3 Resource Allocation:

- The system will allocate classrooms and resources based on availability and capacity.
- It will consider factors such as room size, equipment availability, and accessibility.

3.2.4 Conflict Resolution:

- The system will detect and resolve conflicts such as overlapping schedules or resource shortages.
- Admins will be alerted to conflicts and provided with options for resolution.

3.2.5 Optimization:

- The system will optimize timetables to minimize gaps and maximize resource utilization.
- It will consider preferences, constraints, and historical data to generate efficient schedules.

3.2.6 Notification System:

- Users will receive notifications for schedule changes, updates, and announcements.
- Notifications will be delivered via email and in-app messages.

3.3 Non-functional Requirements

3.3.1 Performance:

- The system must be responsive and capable of handling multiple concurrent users.

- Timetable generation should be fast and efficient, even for large datasets.

3.3.2 Reliability:

- The system must be reliable and available during college operating hours.
- Backup and recovery mechanisms should be in place to prevent data loss.

3.3.3 Security:

- User data must be encrypted and protected from unauthorized access.
- Access controls and permissions should be implemented to prevent data tampering.

3.3.4 Usability:

- The GUI should be intuitive and easy to use, even for non-technical users.
- Help documentation and tooltips should be provided for guidance.

3.3.5 Scalability:

- The system should be scalable to accommodate future growth in data and user base.
- It should support additional features and modules as required.

4. Productivity Improvement

The automatic timetable generation system significantly reduces the time and effort required for timetable creation compared to manual methods. By automating the process, administrators can allocate resources more efficiently, allowing them to focus on other critical tasks. The system optimizes schedules to minimize gaps and conflicts, resulting in smoother operations and better utilization of resources. This optimization leads to increased productivity by ensuring that classrooms, faculty, and students are utilized effectively.

5. Enhanced Customer Experience

The intuitive GUI of the system makes it easy for administrators, faculty, and students to interact with the platform, enhancing the user experience and reducing the learning curve. Timely notifications for schedule changes ensure

that all stakeholders are informed promptly, improving customer satisfaction by keeping them updated and minimizing disruptions.

6. Cost Reduction

By automating timetable generation and optimization, the system reduces the time spent by administrators on manual scheduling tasks. This time-saving translates into cost savings as administrators can allocate their time to other value-added activities. Additionally, the system optimizes resource allocation, ensuring efficient utilization of classrooms and faculty, minimizing wastage, and reducing the need for additional resources, leading to cost savings for the institution.

Conclusion

The Automatic Timetable Generation System for College is a comprehensive solution designed to streamline and optimize the scheduling process. By addressing inefficiencies, conflicts, and resource misallocation, the system enhances productivity, user experience, and cost-efficiency. Its robust functionalities and user-friendly interface make it an essential tool for educational institutions seeking to improve their scheduling operations.

AGILE-SCRUM METHODOLOGY

EXP_NO: 2 **DATE: 20-02-24**

1. Product Backlog

The product backlog defines the different features and functionalities the Automatic Timetable Generation System intends to achieve. It outlines the value that the system would add to the users.

1.1 User Authentication and Access

1.2 Course Management

1.3 Resource Allocation

1.4 Conflict Resolution

1.5 Optimization of Timetables

1.6 Notification System

2. Scrum Backlog

Sprint 1 (2 weeks)

Sprint Goal: Implement basic user authentication and course management functionalities.

User Authentication and Authorization

- **Task 1**: Set up user registration form and backend logic.
- **Task 2**: Design and implement role-based access control system.

Sprint 2 (3 weeks)

Sprint Goal: Set up Course Management requirements.

Course Management

- **Task 3**: Create database schema for storing course details.
- **Task 4**: Implement course creation functionality for administrators.
- **Task 5**: Develop UI for adding and updating course details.

Sprint 3 (4 weeks)

Sprint Goal: Implementation of Resource Allocation and Conflict Resolution.

Resource Allocation

- **Task 6**: Develop UI for defining and managing classroom and faculty availability.
- **Task 7**: Implement backend logic for allocating resources based on availability.

Conflict Resolution

- **Task 8**: Integrate logic for detecting scheduling conflicts.
- **Task 9**: Implement notifications for detected conflicts and provide ion options.

Sprint 4 (3 weeks)

Sprint Goal: Implementation of Optimization of Timetables and Notification System.

Optimization of Timetables

- **Task 10**: Develop algorithm for optimizing timetable schedules.
- **Task 11**: Implement UI for displaying optimized timetables.

Notification System

- **Task 12**: Implement notification system for schedule changes.
- **Task 13**: Develop functionality for email and in-app notifications for updates.

USER STORIES

EXP.NO:3 DATE:12-03-24

1. User Stories as Functional Requirements

1.1 User Authentication and Access

1. As a user, I want to log in securely with my credentials.
2. As an administrator, I want to grant access to faculty and students.

Acceptance Criteria:

- The login page should be accessible from the main navigation or a dedicated login link.
- The page should include fields for entering a username/email and password.
- The system should validate the entered username/email and password.

1.2 Course Management

1. As an administrator, I want to create new courses with detailed information.
2. As a faculty member, I want to view and manage the courses I am assigned to.

Acceptance Criteria:

- The administrator should have access to a course creation form through the admin dashboard.
- The form should include fields for entering the course name, code, duration, and prerequisites.
- Upon submission, the system should create a new course with the provided details.

1.3 Resource Allocation

1. As an administrator, I want to allocate classrooms and resources based on availability.
2. As a faculty member, I want to view the resources allocated to my courses.

Acceptance Criteria:

- The administrator should have access to a resource allocation form within the course management page.
- The form should include options for selecting available classrooms and resources.
- Upon submission, the system should allocate the selected resources to the specified courses.

1.4 Conflict Resolution

1. As an administrator, I want the system to detect and resolve scheduling conflicts automatically.
2. As a faculty member, I want to receive alerts about any conflicts in my schedule.

Acceptance Criteria:

- The system should automatically detect conflicts such as overlapping schedules or resource shortages.
- Administrators should be alerted to conflicts and provided with resolution options.
- Faculty members should receive notifications about any conflicts affecting their schedules.

1.5 Optimization of Timetables

1. As an administrator, I want the system to optimize timetables to minimize gaps and maximize resource utilization.
2. As a student, I want to have a timetable that minimizes idle time between classes.

Acceptance Criteria:

- The system should use algorithms to optimize timetables based on input constraints and preferences.
- The optimized timetable should aim to minimize gaps between classes and efficiently utilize available resources.
- Students should be able to view their optimized timetables through their user interface.

1.6 Notification System

1. As a faculty member, I want to receive notifications for schedule changes.
2. As a student, I want to be alerted about any updates to my timetable.

Acceptance Criteria:

- The system should send notifications via email and in-app messages for any schedule changes.
- Users should be able to view notifications through their user dashboard.

2.User Stories as Non-Functional Requirements:

2.1. User Authentication and Access:

- As a user, I want to securely log in to the automatic timetable management system using my credentials, ensuring that only authorized individuals can access and modify timetable data.

2.2. Timetable Generation:

- As an administrator, I want the system to automatically generate timetables for classes based on predefined constraints such as class timings, teacher availability, room capacities, and other relevant factors.

2.3. Customization Options:

- As a faculty member, I want the ability to customize the generated timetables by adjusting class timings, assigning specific rooms for classes, rescheduling classes to accommodate special events, and blocking off times for maintenance or other purposes.

2.4. Conflict Resolution:

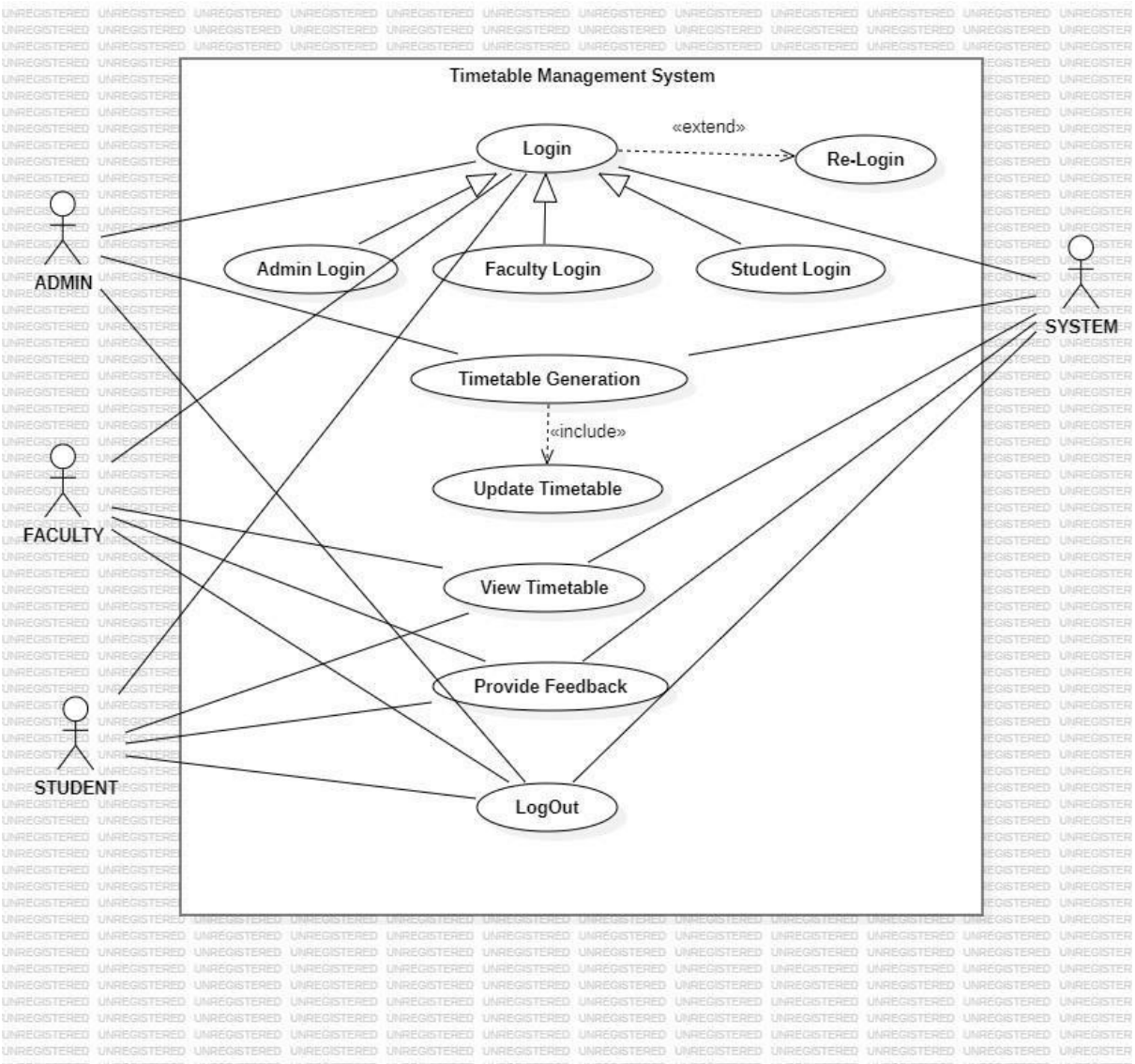
- As a timetable administrator, I want the system to detect and resolve scheduling conflicts automatically, ensuring that no classes overlap, resources are optimally utilized, and all constraints are met while generating timetables.

2.5. Reporting and Analytics:

- As an administrator or stakeholder, I want access to comprehensive reports and analytics that provide insights into timetable schedules, resource utilization, any conflicts encountered during timetable generation, and trends in scheduling patterns over time. This information will help in making data-driven decisions and optimizing the timetable management process.

USERCASE DIAGRAM

EXP.NO:4 DATE:19-03-24



NFR-NON FUNCTIONAL REQUIREMENT

EXP.NO:5 DATE:29-03-24

1 Performance:

- The system must be responsive and capable of handling multiple concurrent users.
- Timetable generation should be fast and efficient, even for large datasets.

2 Reliability:

- The system must be reliable and available during college operating hours.
- Backup and recovery mechanisms should be in place to prevent data loss.

3.Security:

- User data must be encrypted and protected from unauthorized access.
- Access controls and permissions should be implemented to prevent data tampering.

4.Usability:

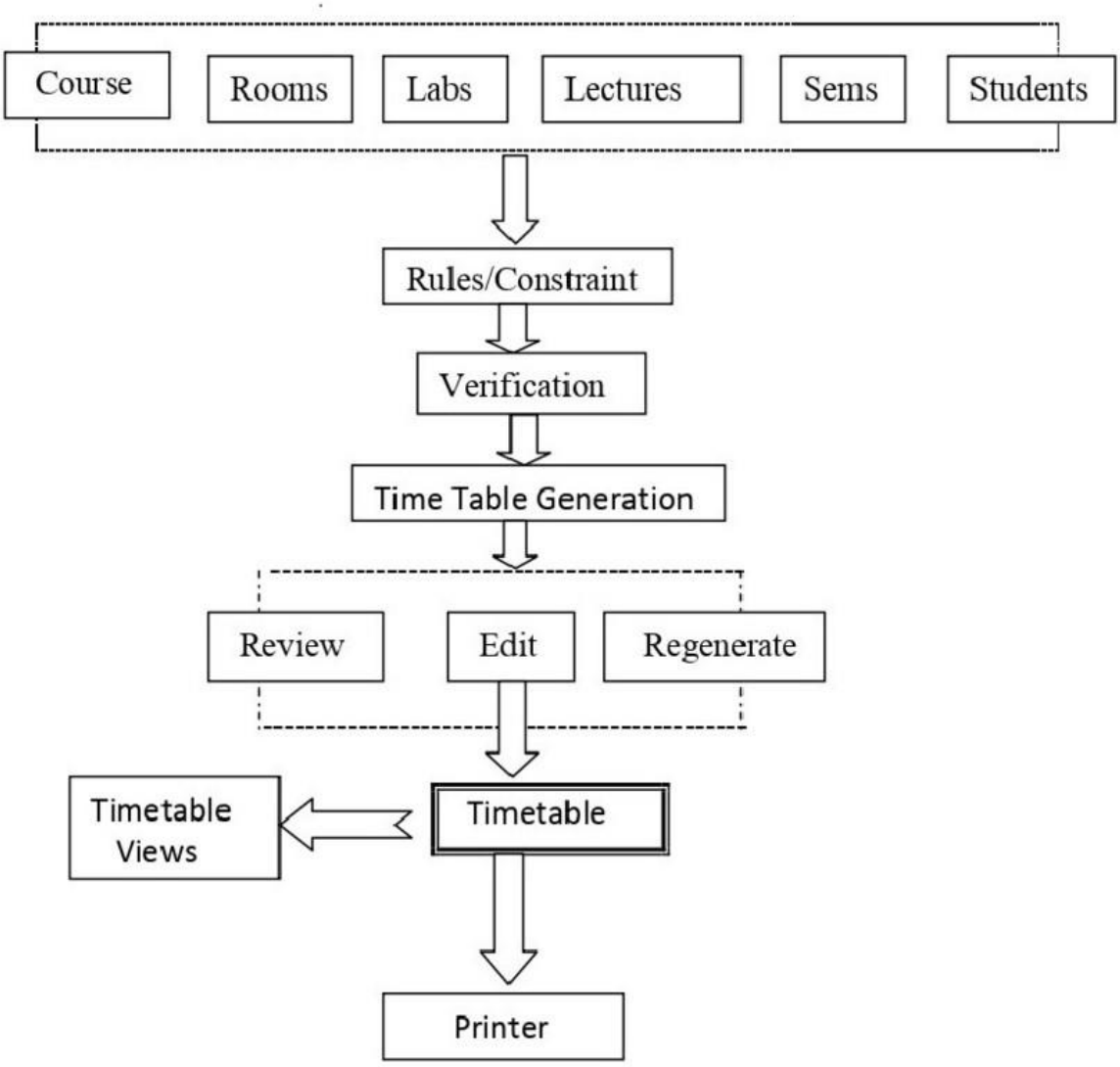
- The GUI should be intuitive and easy to use, even for non-technical users.
- Help documentation and tooltips should be provided for guidance.

5.Scalability:

- The system should be scalable to accommodate future growth in data and user base.
- It should support additional features and modules as required.

Overall Architecture Diagram

EXP.NO:6 DATE:09-04-24



Business Architecture

EXP.NO:7 DATE:19-04-24

Strategy

1. **Efficient Resource Allocation**
    - Ensure optimal use of available resources (e.g., classrooms, teachers, equipment) to meet the scheduling needs without conflicts.
2. **Maximize Student and Teacher Satisfaction**
    - Take into account preferences and constraints of both students and teachers to create a schedule that minimizes stress and maximizes productivity.
3. **Adaptability and Flexibility**
    - Implement a system that can easily adapt to changes and accommodate special requests or unforeseen circumstances, ensuring the timetable remains effective and efficient.

Business Capabilities

1. **Plan**
    - Determine scheduling needs and constraints.
2. **Develop**
    - Create and optimize timetable algorithms.
3. **Manage**
    - Continuously improve scheduling processes.

Corporate Support

1. **Manage Performance**
    - Monitor and adjust timetable effectiveness.
2. **Manage Information and Knowledge**
    - Ensure accurate data on resources and constraints.
3. **Manage Finances**

- Optimize costs related to scheduling.

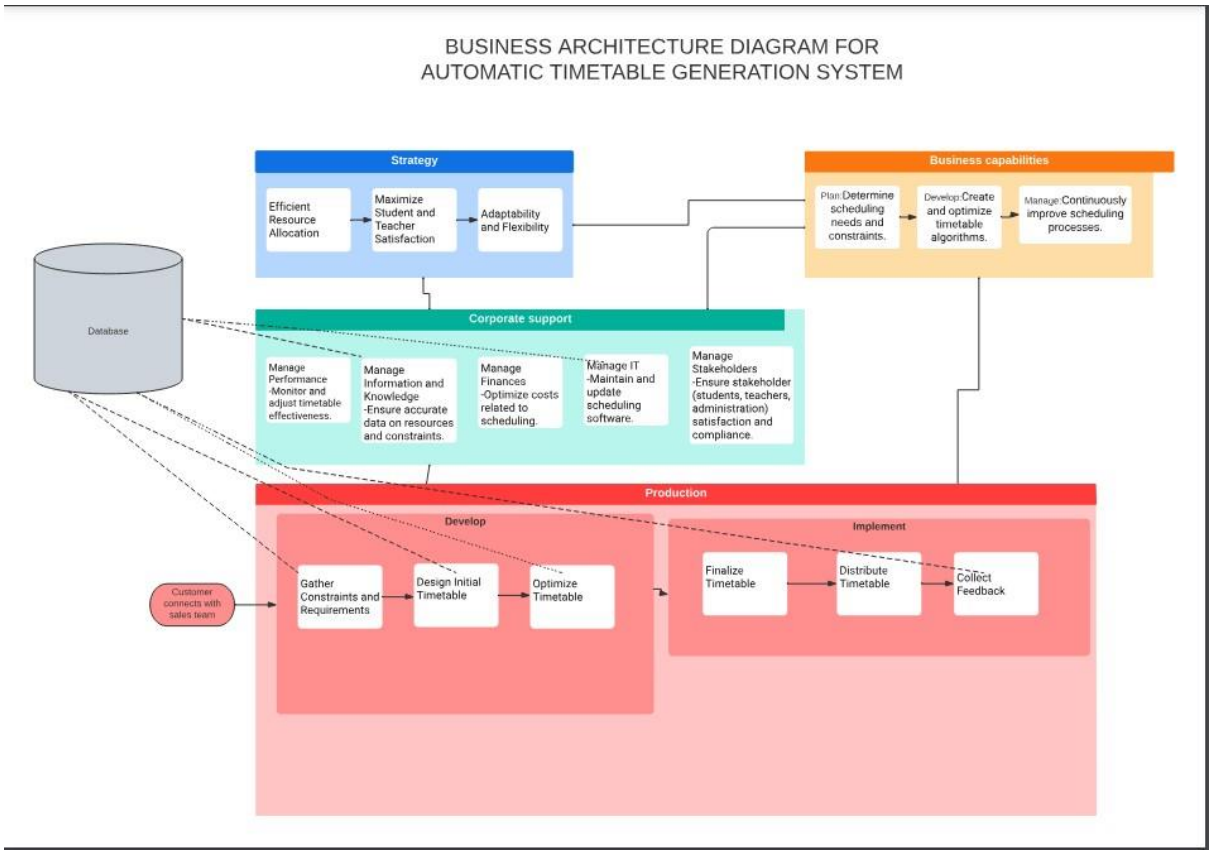
4. **Manage IT**
    - Maintain and update scheduling software.
5. **Manage Stakeholders**
    - Ensure stakeholder (students, teachers, administration) satisfaction and compliance.

Develop

1. **Gather Constraints and Requirements**
    - Collect all necessary data including classroom availability, teacher schedules, and student preferences.
2. **Design Initial Timetable**
    - Create an initial draft of the timetable considering the gathered constraints and requirements.
3. **Optimize Timetable**
    - Refine the timetable to resolve conflicts and improve efficiency and satisfaction for all stakeholders.

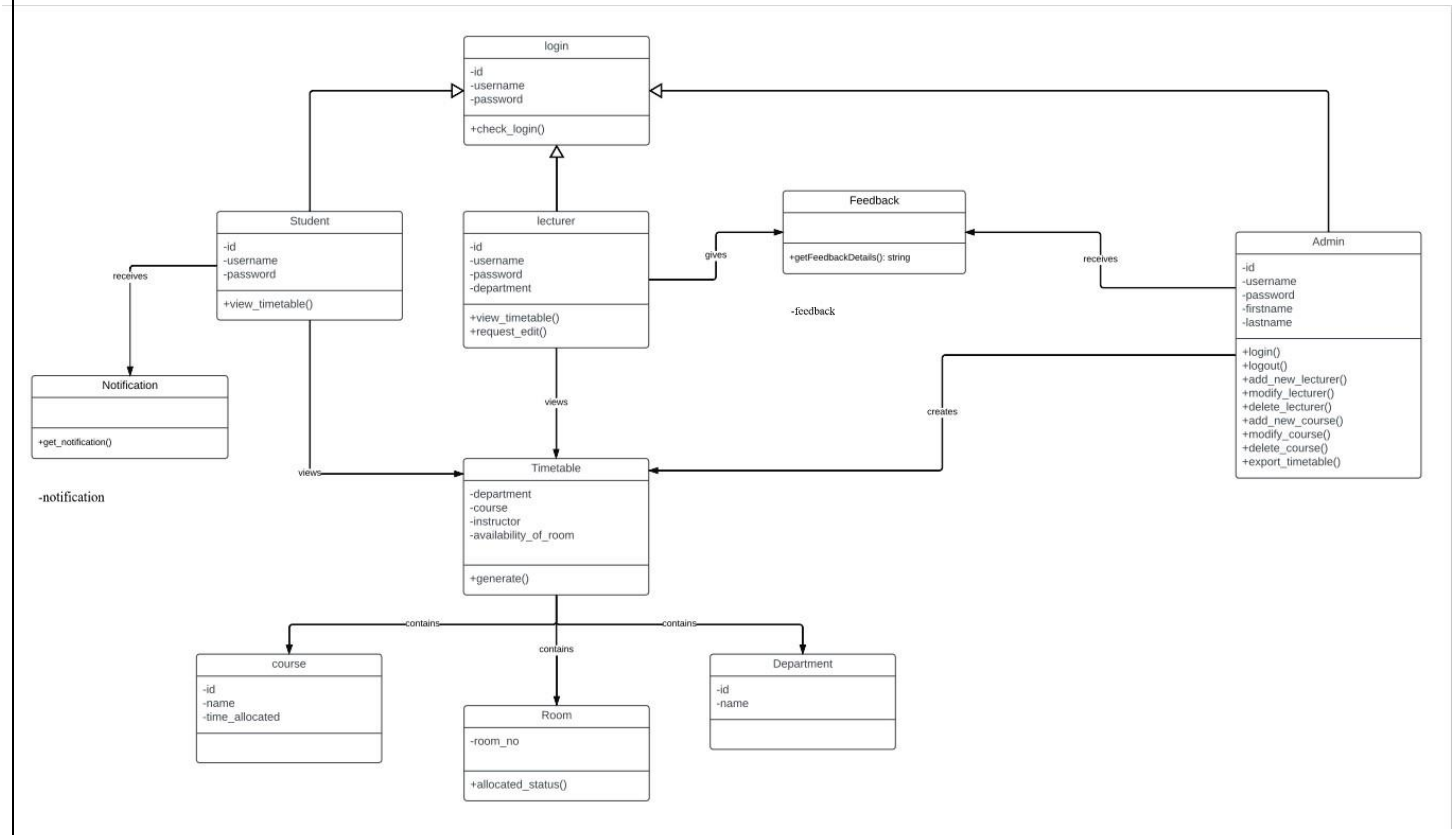
Implement

1. **Finalize Timetable**
    - Review and finalize the timetable for implementation, ensuring accuracy and feasibility.
2. **Distribute Timetable**
    - Distribute the finalized timetable to students, teachers, and administrative staff through appropriate channels.
3. **Collect Feedback**
    - Gather feedback from stakeholders on the implemented timetable to identify areas for improvement.


BUSINESS ARCHITECTURE DIAGRAM FOR
AUTOMATIC TIMETABLE GENERATION SYSTEM
Strategy
Efficient Resource Allocation
Maximize Student and Teacher Satisfaction
Adaptability and Flexibility
Business capabilities
Plan:Determine scheduling needs and constraints.
Develop:Create and optimize timetable algorithms.
Manage:Continuously improve scheduling processes.
Database
Corporate support
Manage Performance -Monitor and adjust timetable effectiveness.
Manage Information and Knowledge -Ensure accurate data on resources and constraints.
Manage Finances -Optimize costs related to scheduling.
Manage IT -Maintain and update scheduling software.
Manage Stakeholders -Ensure stakeholder (students, teachers, administration) satisfaction and compliance.
Production
Develop
Implement
Customer connects with sales team
Gather Constraints and Requirements
Design Initial Timetable
Optimize Timetable
Finalize Timetable
Distribute Timetable
Collect Feedback

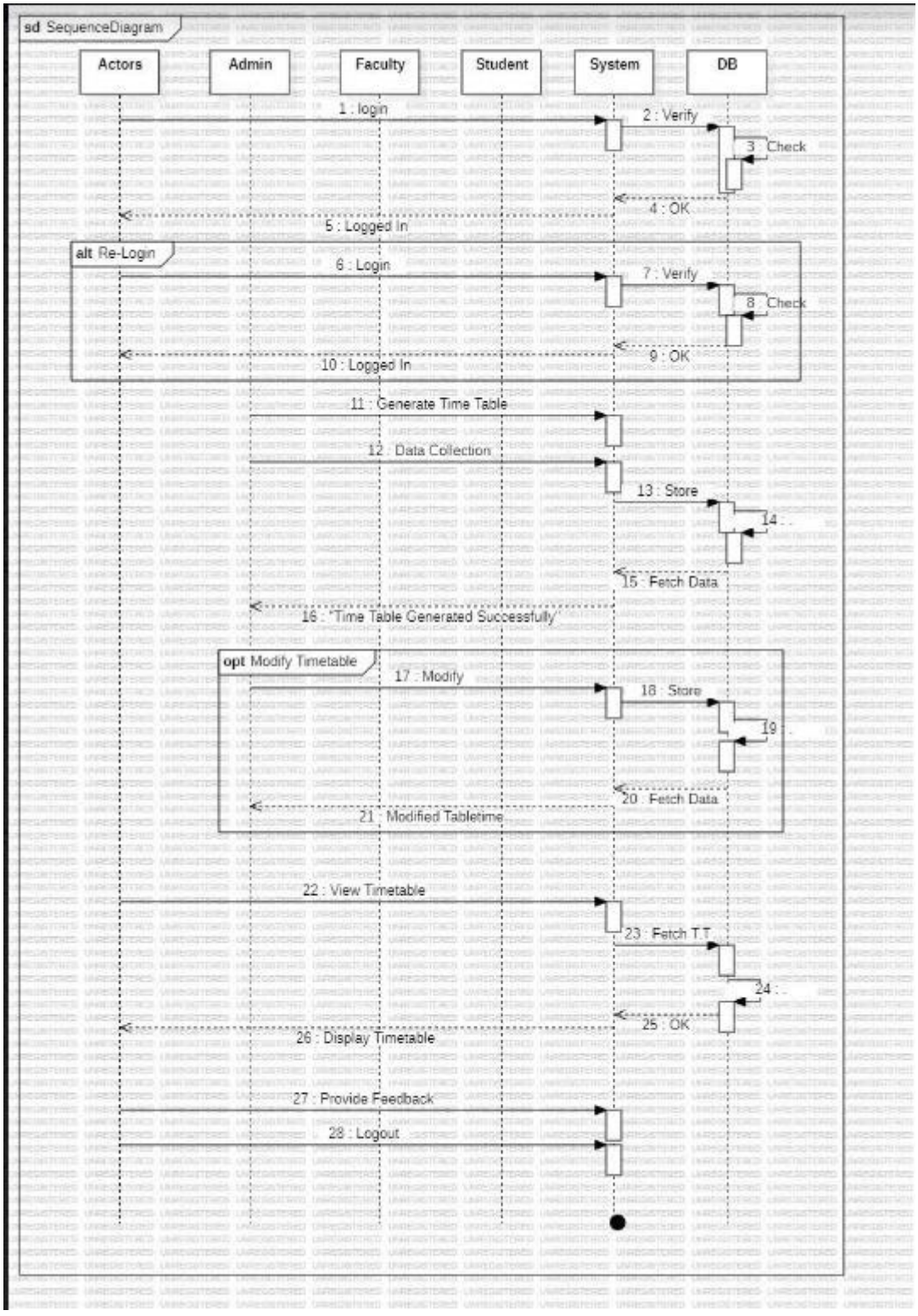
Class Diagram

EXP.NO:8 DATE:30-04-24



Sequence Diagram

EXP.NO:9 DATE:10-05-24



Architectural Pattern Model View Controller

EXP.NO:10 DATE:17-05-24

Automatic Timetable Generation System using MVC Architecture

1. Model

The Model represents the data and business logic of the application. It manages the data, logic, and rules of the timetable generation system.

Components:

- **Lecturer:** Represents lecturer data, including lecturer ID, name, department, and available time slots.
- **Course:** Represents course data, including course ID, name, department, and duration.
- **Timetable:** Represents timetable data, including timetable ID, date, time slot, course, and assigned lecturer.
- **Room:** Represents room data, including room ID, name, capacity, and available time slots.
- **User:** Represents user data, including user ID, name, role (admin, lecturer, student), and credentials.
- **Feedback:** Represents feedback data, including feedback ID, type, content, user ID, and timestamp.
- **Notification:** Represents notification data, including notification ID, type, content, recipient, and timestamp.

2. View

The View represents the UI components of the application. It displays data to the user and sends user commands to the controller.

Components:

- **Login View:** UI for user authentication, allowing users to log in securely.
- **Dashboard View:** UI for displaying an overview of timetables, tailored to user roles (admin, lecturers, students).
- **Timetable View:** UI for viewing and printing timetables, allowing users to select specific dates or courses.

- **Lecturer Management View:** UI for admins to add, edit, and view lecturer information.
- **Course Management View:** UI for admins to add, edit, and view course information.
- **Room Management View:** UI for admins to add, edit, and view room information.
- **Feedback View:** UI for managing and viewing feedback from lecturers and students.
- **Notification View:** UI for managing and viewing notifications.

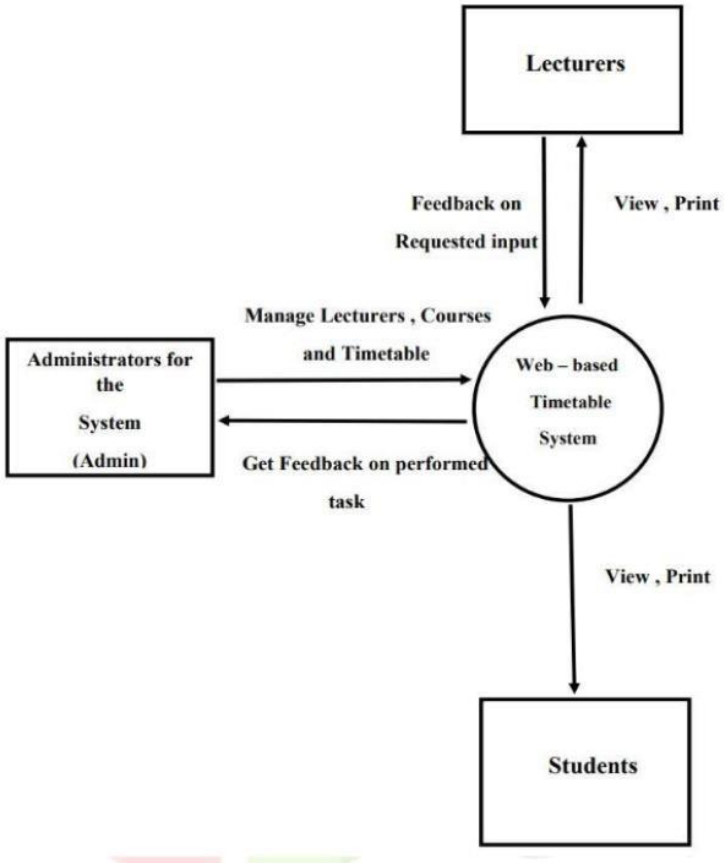
3. Controller

The Controller handles user input and updates the Model and View accordingly. It processes all business logic and incoming requests, manipulates data using the Model, and interacts with the Views to render the final output.

Components:

- **Authentication Controller:** Manages user login, logout, and session management.
- **Lecturer Controller:** Handles creation, updating, and retrieval of lecturer data.
- **Course Controller:** Manages creation, updating, and retrieval of course data.
- **Timetable Controller:** Handles automatic generation of timetables, ensuring no conflicts between courses, lecturers, and rooms.
- **Room Controller:** Manages creation, updating, and retrieval of room data.
- **Feedback Controller:** Handles submission, retrieval, and management of feedback.
- **Notification Controller:** Sends notifications to users based on timetable updates, feedback submissions, and other relevant events.

Automatic Timetable Generation System using MVC Architecture diagram



Best 5 Test Cases for Automatic Timetable Generation System

Test Case 1: Creating a New Timetable

Description: Verify that an admin can create a new timetable from scratch.

Preconditions:

- User is logged into the automatic timetable generation application with admin privileges.

Test Steps:

1. Navigate to the "Create New Timetable" section.
2. Enter basic timetable information (Term, Department, Start Date, End Date).
3. Add courses, resources, and faculty to the timetable.
4. Click the "Generate Timetable" button.
5. Review the generated timetable.
6. Click the "Save" button.

Expected Results:

- The application should save the entered information.
- The user should see a confirmation message indicating the timetable has been saved successfully.
- The newly created timetable should appear in the list of timetables for the admin.

Test Case 2: Editing an Existing Timetable

Description: Verify that an admin can edit an existing timetable.

Preconditions:

- User is logged into the automatic timetable generation application with admin privileges.
- User has at least one saved timetable.

Test Steps:

1. Navigate to the "My Timetables" section.
2. Select an existing timetable to edit.
3. Modify the timetable information (e.g., update course schedules, resources).
4. Click the "Generate Timetable" button to regenerate the timetable.

5. Review the updated timetable.

6. Click the "Save" button.

Expected Results:

- The application should update the timetable with the new information.

- The user should see a confirmation message indicating the timetable has been updated successfully.

- The modified timetable should reflect the changes made.

Test Case 3: Conflict Resolution

Description: Verify that the system can detect and resolve scheduling conflicts.

Preconditions:

- User is logged into the automatic timetable generation application with admin privileges.

- User has at least one saved timetable with potential conflicts.

Test Steps:

1. Navigate to the "My Timetables" section.

2. Select an existing timetable with conflicts.

3. Click the "Resolve Conflicts" button.

4. Review the detected conflicts and suggested resolutions.

5. Choose the appropriate resolution options.

6. Click the "Apply Changes" button.

Expected Results:

- The system should detect conflicts such as overlapping schedules or resource shortages.

- The admin should be alerted to conflicts and provided with options for resolution.

- The resolved timetable should reflect the changes made to resolve conflicts.

Test Case 4: Notification System

Description: Verify that users receive notifications for schedule changes.

Preconditions:

- User is logged into the automatic timetable generation application.

- User has at least one assigned timetable.

Test Steps:

1. Admin makes changes to an existing timetable (e.g., updates course schedules).

2. The system generates notifications for the affected users.

3. Users check their email and in-app messages.

Expected Results:

- The system should send notifications via email and in-app messages for any schedule changes.

- Users should be able to view notifications through their user dashboard.

- Notifications should include details of the changes made to the timetable.

Test Case 5: Teacher Feedback for Timetable Adjustment

Description: Verify that a teacher can provide feedback on the timetable and the system generates a new timetable based on this feedback.

Preconditions:

- User is logged into the automatic timetable generation application with teacher privileges.

- User has at least one assigned timetable.

Test Steps:

1. Navigate to the "My Timetables" section.

2. Select an existing timetable to provide feedback on.

3. Click the "Provide Feedback" button.

4. Enter feedback details (e.g., preferred class times, resource requirements, issues with the current timetable).

5. Click the "Submit Feedback" button.

6. Admin reviews and approves the feedback.

7. The system generates a new timetable based on the feedback.

Expected Results:

- The feedback should be submitted successfully.

- The admin should be notified of the new feedback submission.

- After admin approval, the system should generate a new timetable incorporating the teacher's feedback.

- The new timetable should appear in the list of timetables for the teacher and should reflect the feedback provided.

Test Case 6: Downloading Timetable Details

Description: Verify that a user can download timetable details in PDF format.

Preconditions:

- User is logged into the automatic timetable generation application.

- User has at least one assigned timetable.

Test Steps:

1. Navigate to the "My Timetables" section.

2. Select an existing timetable to download.

3. Click the "Download" button.

4. Choose "PDF" as the format.

Expected Results:

- The application should generate a PDF file of the timetable details.

- The user should see a prompt to download the PDF file.

- The downloaded PDF should contain the correct and formatted information of the timetable.

Test Case 7: Using a Template for Timetable Creation

Description: Verify that an admin can create a timetable using a predefined template.

Preconditions:

- User is logged into the automatic timetable generation application with admin privileges.

Test Steps:

1. Navigate to the "Create New Timetable" section.

2. Select a predefined template from the available options.

3. Enter timetable information, courses, resources, and faculty assignments.

4. Click the "Preview" button to see the timetable with the selected template.

5. Click the "Generate Timetable" button.

6. Review the generated timetable.

7. Click the "Save" button.

Expected Results:

- The application should save the entered information with the selected template.

- The user should see a confirmation message indicating the timetable has been saved successfully.

- The preview should display the timetable with the correct template formatting.

- The newly created timetable should appear in the list of timetables with the applied template.

Test Case 8: Deleting a Timetable

Description: Verify that an admin can delete an existing timetable.

Preconditions:

- User is logged into the automatic timetable generation application with admin privileges.

- User has at least one saved timetable.

Test Steps:

1. Navigate to the "My Timetables" section.

2. Select a timetable to delete.

3. Click the "Delete" button.

4. Confirm the deletion in the confirmation dialog.

Expected Results:

- The application should delete the selected timetable.

- The user should see a confirmation message indicating the timetable has been deleted successfully.

- The deleted timetable should no longer appear in the list of timetables for the user.

Test Case 9: Viewing Timetable History

Description: Verify that a user can view the history of changes made to a timetable.

Preconditions:

- User is logged into the automatic timetable generation application.

- User has at least one assigned timetable with a change history.

Test Steps:

1. Navigate to the "My Timetables" section.

2. Select a timetable to view its history.

3. Click the "View History" button.

Expected Results:

- The application should display a history of changes made to the timetable.

- The history should include details such as changes made, dates of changes, and the user who made the changes.

Test Case 10: Assigning Multiple Roles to a User

Description: Verify that a user can be assigned multiple roles (e.g., admin and faculty) and access the respective functionalities.

Preconditions:

- User is logged into the automatic timetable generation application with admin privileges.

- User management functionality is available.

Test Steps:

1. Navigate to the "User Management" section.

2. Select a user to assign multiple roles.

3. Assign the user both "Admin" and "Faculty" roles.

4. Log in as the user with multiple roles.

5. Navigate to both admin and faculty functionalities (e.g., create timetable, view assigned courses).

Expected Results:

- The user should have access to both admin and faculty functionalities.

- The user should be able to perform tasks related to both roles without any issues.

These additional test cases cover a range of functionalities necessary for ensuring the robustness and usability of the automatic timetable generation system.