**COMSATS University Islamabad, Abbottabad Campus**

**Department of Computer Science**

**Project Proposal**

**TIMETABLE MANAGEMENT SYSTEM**

**CSC392 Object-Oriented Software Engineering**

Submitted on: <04-04-2023>

Group Members:

Saud Khan (FA21-BSE-033) Leader

Usama Ijaz (FA21-BSE-158)

Zakariya Anwer (FA21-BSE-042)

Table of Contents

[**CHAPTER 1 PROJECT PROPOSAL** 4](#_Toc135770905)

[**Introduction** 4](#_Toc135770906)

[**Vision and Business Case** 4](#_Toc135770907)

[**Use-Case Model** 5](#_Toc135770908)

[**Supplementary Specification** 5](#_Toc135770909)

[**Glossary** 6](#_Toc135770910)

[**Risk list and risk management plan** 6](#_Toc135770911)

[**CHAPTER 2 USE CASES** 9](#_Toc135770912)

[**Use Case Diagram** 9](#_Toc135770913)

[**Use Cases Distribution** 10](#_Toc135770914)

[**Brief Level Use Cases** 10](#_Toc135770915)

[**Saud Khan (FA21-BSE-033)** 10](#_Toc135770916)

[**Use Case: Create slot** 10](#_Toc135770917)

[**Use Case: Update Slot** 10](#_Toc135770918)

[**Use Case: Delete slot** 11](#_Toc135770919)

[**Use Case: Add slot** 11](#_Toc135770920)

[**Use Case: Give course preferences** 11](#_Toc135770921)

[**Use Case: Select course preferences** 11](#_Toc135770922)

[**Zakariya Anwar (FA21-BSE-042)** 11](#_Toc135770923)

[**Use Case: Login** 11](#_Toc135770924)

[**Use Case: View Timetable** 12](#_Toc135770925)

[**Use Case: Faculty timetable view** 12](#_Toc135770926)

[**Use Case: Student timetable view** 12](#_Toc135770927)

[**Use Case: Send notification** 12](#_Toc135770928)

[**Use Case: Generate request to modify** 12](#_Toc135770929)

[**Usama Ijaz (FA21-BSE-158)** 12](#_Toc135770930)

[**Use Case: Create Timetable** 12](#_Toc135770931)

[**Use Case: Clash detection** 12](#_Toc135770932)

[**Use Case: Set Courses** 13](#_Toc135770933)

[**Use Case: Allot rooms** 13](#_Toc135770934)

[**Use Case: Change course room** 13](#_Toc135770935)

[**Use Case: Modify course timing** 13](#_Toc135770936)

[**Fully Dressed Use Cases** 14](#_Toc135770937)

[**Usama Ijaz (FA21-BSE-158)** 14](#_Toc135770938)

[**Use Case UC 1: Create Timetable** 14](#_Toc135770939)

[**Use Case UC 2: Clash Detection** 15](#_Toc135770940)

[**Use Case UC 3: Set Courses** 17](#_Toc135770941)

[**Use Case UC 4: Allot Rooms** 19](#_Toc135770942)

[**Use Case UC 5: Change Course Room** 21](#_Toc135770943)

[**Use Case UC 6: Modify Course Timing** 23](#_Toc135770944)

[**Saud Khan (FA21-BSE-033)** 24](#_Toc135770945)

[**Use Case UC 7: Create Slot** 24](#_Toc135770946)

[**Use Case UC 8: Update Slot** 26](#_Toc135770947)

[**Use Case UC 9: Delete Slot** 28](#_Toc135770948)

[**Use Case UC 10: Add Slot** 29](#_Toc135770949)

[**Use Case UC 11: Give course preferences** 31](#_Toc135770950)

[**Use Case UC 12: Select Course Preferences** 32](#_Toc135770951)

[**Zakariya Anwar Khan (FA21-BSE-042)** 34](#_Toc135770952)

[**Use Case UC 13: Login** 34](#_Toc135770953)

[**Use Case UC 14: View Timetable** 35](#_Toc135770954)

[**Use Case UC15: Faculty Timetable View** 36](#_Toc135770955)

[**Use Case UC16: Student Timetable View** 37](#_Toc135770956)

[**Use Case UC17: Send Notification** 39](#_Toc135770957)

[**Use Case UC18: Generate Request to Modify** 40](#_Toc135770958)

[**CHAPTER 3 DOMAIN MODEL** 42](#_Toc135770959)

[**CHAPTER 4 SYSTEM SEQUENCE DIAGRAM** 43](#_Toc135770960)

[**Saud Khan:** 43](#_Toc135770961)

[**UC1: Create Slot** 43](#_Toc135770962)

[**UC2: Update Slot** 44](#_Toc135770963)

[**UC3: Delete Slot** 44](#_Toc135770964)

[**UC4: Add Slot** 45](#_Toc135770965)

[**UC3: Give Course Preferences** 45](#_Toc135770966)

[**UC6: Select Course Preferences** 46](#_Toc135770967)

# **CHAPTER 1 PROJECT PROPOSAL**

## **Introduction**

A timetable management system is a software application that helps users create, manage, and organize their schedules. It can be used for a variety of purposes, such as managing academic timetables.

The system allows users to input their schedules, and then generates a visual representation of the timetable that can be easily viewed and modified as needed. Admin can add, delete, or modify, and the system will automatically update the timetable accordingly.

## **Vision and Business Case**

The high-level goals of the timetable management system for a university are to improve scheduling efficiency, reduce conflicts, and increase overall productivity. The system should be able to handle a large number of users, including students, faculty, and staff, and be able to handle complex scheduling requirements, such as room and resource allocation.

The constraints of the system include the need to integrate with existing university systems, such as student and faculty databases, and the need to adhere to strict privacy and security protocols to protect sensitive data.

**Business Case:**

The timetable management system for a university can provide significant benefits, including:

1. Increased efficiency: The system can automate many of the scheduling processes, reducing the time and effort required to create and manage schedules.
2. Reduced conflicts: The system can detect and resolve scheduling conflicts, reducing the likelihood of double-bookings or other issues.
3. Improved communication: The system can provide a centralized location for all scheduling information, making it easier for students, faculty, and staff to stay informed and up-to-date.
4. Enhanced productivity: By streamlining scheduling processes and reducing conflicts, the system can help increase overall productivity and effectiveness.

**Executive Summary:**

The timetable management system for a university is a software application that helps improve scheduling efficiency, reduce conflicts, and increase productivity. The system should be able to handle complex scheduling requirements and integrate with existing university systems while adhering to strict privacy and security protocols. The system can provide significant benefits, including increased efficiency, reduced conflicts, improved communication, and enhanced productivity. By implementing a timetable management system, the university can improve its scheduling processes and better serve its students, faculty, and staff.

## **Use-Case Model**

The use-case model for a timetable management system for a university may include the following functional requirements:

1. **Login:** Allow users to log in to the system using their university credentials.
2. **Create Schedule:** Allow admin to create a new schedule for a particular semester or academic term.
3. **Modify Schedule:** Allow admin to modify an existing schedule by adding or removing classes or changing rooms.
4. **View Schedule:** Allow users (student, faculty & other staff) to view their own schedules or timetable.
5. **Room Allotting:** Allow authorized users to allot room or do changing in it.
6. **Resource Allocation:** Allow authorized users to allocate resources (such as projectors, microphones, or laptops) for classes or events.
7. **Notification:** Notify users of any changes to their schedules or scheduling conflicts.
8. **Report Generation:** Allow users to generate reports on scheduling information, such as attendance reports or room utilization reports.

## **Supplementary Specification**

1. **Performance:** The system should be able to handle a large number of users and data without significant delays or downtime.
2. **Scalability**: The system should be scalable, able to accommodate an increasing number of users and data as the university grows.
3. **Availability**: The system should be available to users at all times, with minimal downtime for maintenance or upgrades.
4. **Security**: The system should be designed with robust security features to protect sensitive data and prevent unauthorized access.
5. **Usability**: The system should be easy to use, with an intuitive interface and clear instructions for users.
6. **Accessibility**: The system should be accessible to users with disabilities, conforming to accessibility standards and guidelines.
7. **Reliability**: The system should be reliable, with minimal errors or failures.
8. **Compatibility**: The system should be compatible with various browsers and devices, to ensure all users can access the system.
9. **Compliance**: The system should comply with relevant laws, regulations, and policies, such as data protection laws and university policies on information security and privacy.
10. **Maintenance**: The system should be easy to maintain and upgrade, with minimal disruption to users.

During the inception phase, it is useful to have some idea of the key non-functional requirements that will have a major impact on the architecture. This will help ensure that the system is designed to meet these requirements, and that any trade-offs between different requirements are identified and addressed. As the project progresses, additional non-functional requirements may be identified and added to the supplementary specification as needed.

## **Glossary**

1. **Timetable Management System:** A software application that allows users to create, modify, and manage schedules for classes and events at a university.
2. **Schedule**: A list of classes or events that occur during a specific time period, such as a semester or academic term.
3. **Scheduling Conflict:** An issue that occurs when two or more classes or events are scheduled at the same time or in the same location.
4. **Resource**: Any physical item required for a class or event, such as projectors, microphones, or laptops.
5. **Room**: A physical space used for classes or events, such as a classroom or lecture hall.
6. **User**: Any individual who interacts with the timetable management system, including students, faculty, and staff.

## **Risk list and risk management plan**

**Risk List**

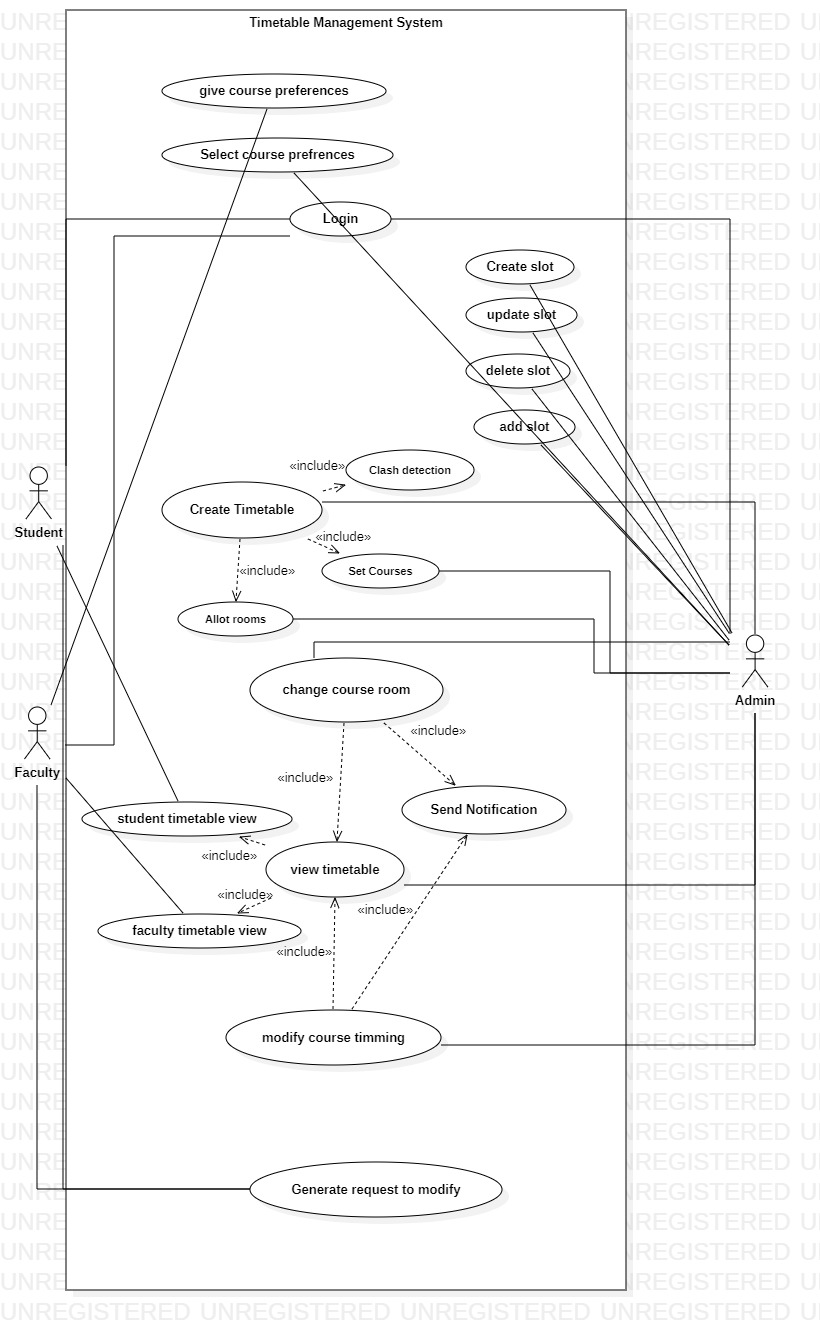
1. **Inadequate System Performance:** There is a risk that the timetable management system will not perform adequately, resulting in delays or downtime for users.
2. **Security Breach:** There is a risk that the system will be compromised by a security breach, resulting in unauthorized access to sensitive data.
3. **Insufficient Budget:** There is a risk that the project budget will be insufficient to meet all project requirements, resulting in incomplete or suboptimal system functionality.
4. **Lack of Skilled Personnel:** There is a risk that the project team will lack the necessary skills and expertise to complete the project successfully.
5. **Project Delays:** There is a risk that the project will be delayed due to unforeseen circumstances, such as technical issues or resource constraints.
6. **Resistance to Change:** There is a risk that users may resist using the new system, resulting in low adoption rates and limited system effectiveness.

**Risk Management Plan:**

1. **Inadequate System Performance:** Mitigation strategies could include performance testing and optimization during development, as well as ongoing monitoring and maintenance to ensure system performance meets expectations.
2. **Security Breach:** Mitigation strategies could include implementing robust security features such as encryption, firewalls, and user authentication, as well as ongoing monitoring and updating of security protocols.
3. **Insufficient Budget:** Mitigation strategies could include careful budget planning and prioritization, as well as seeking additional funding sources if necessary.
4. **Lack of Skilled Personnel:** Mitigation strategies could include identifying skill gaps early in the project and addressing them through training, hiring additional staff with necessary expertise, or outsourcing certain tasks to third-party providers.
5. **Project Delays:** Mitigation strategies could include developing a realistic project timeline with contingency plans, monitoring progress regularly, and adjusting timelines as needed to ensure the project stays on track.
6. **Resistance to Change:** Mitigation strategies could include involving users in the development process through user testing and feedback, providing clear communication and training materials, and highlighting the benefits of the new system to encourage user adoption.

Top of Form

# **CHAPTER 2 USE CASES**

**Use Case Diagram**

## **Use Cases Distribution**

|  |  |  |
| --- | --- | --- |
| S#. | Group Member | Assigned Use Cases |
| 1 | Usama Ijaz  FA21-BSE-158 | UC 1: Create Timetable  UC 2: Clash detection  UC 3: Set Courses  UC 4: Allot rooms  UC 5: Change course room  UC 6: Modify course timing |
| 2 | Saud Khan  FA21-BSE-033 | UC 7: Create slot  UC 8: Update slot  UC 9: Delete slot  UC 10: Add slot  UC 11: Give Course preferences  UC 12: Select course preferences |
| 3 | Zakariya Anwar  FA21-BSE-042 | UC 13: Login  UC 14: View timetable  UC 15: Faculty timetable view  UC 16: Student timetable view  UC 17: Send notification  UC 18: Generate request to modify |
|  |  |  |

## **Brief Level Use Cases**

### **Saud Khan (FA21-BSE-033)**

#### **Use Case: Create slot**

The academic administrator can only create slot for timetable which will be of specific interval of time assigning course and teacher to it. The system would need to ensure that there are no scheduling conflicts with other courses or events already scheduled, and that the instructor is available during the specified time. Once the slot is created, it would be visible to students who can then register for the course.

#### **Use Case: Update Slot**

The admin can only update the slot in timetable. If there are any conflicts with other slots or users, the system alerts the administrator and provides options for resolving the conflicts. The administrator can review the updated access settings and make further adjustments as needed before finalizing the changes. Once the changes are finalized, the timetable management system restricts access to the slot to only users with administrator privileges, ensuring the slot is secure and protected from unauthorized access.

#### **Use Case: Delete slot**

The Delete Slot use case allows an admin to remove a scheduled slot from a timetable in the Timetable Management System. This subfunction supports the primary use cases of creating and modifying a timetable, providing flexibility in managing the schedule of classes. The deletion of a slot may be necessary due to changes in availability, cancellations, or rescheduling. The Delete Slot function ensures that the timetable remains accurate and up-to-date, enabling users to plan their activities effectively.

#### **Use Case: Add slot**

The Add Slot use case allows an admin to schedule a new slot in a timetable in the Timetable Management System. This subfunction supports the primary use cases of creating and modifying a timetable, providing flexibility in managing the schedule of classes. The addition of a slot may be necessary due to new timings, changes in availability. The Add Slot function ensures that the timetable remains accurate and up-to-date, enabling users to plan their activities effectively.

#### **Use Case: Give course preferences**

The Give Course Preferences use case allows a faculty member to provide their course preferences for scheduling in the Timetable Management System. This primary use case is essential in the process of creating and managing a timetable, as it enables the system to consider faculty availability and preferences when assigning courses. The Give Course Preferences function ensures that the timetable can be optimized to accommodate the needs of both the faculty and the students, ensuring efficient use of resources and better alignment with the academic goals of the institution.

#### **Use Case: Select course preferences**

The Select Course Preferences use case allows an authorized administrator to specify the preferences for the courses in the Timetable Management System. The administrator can set the preferences for each course, such as preferred times, room types, and faculty availability. The preferences are used by the system to generate the timetable, optimizing the use of resources and minimizing scheduling conflicts. The Select Course Preferences function ensures that the timetable meets the needs of the courses and the students, enhancing the quality of education and improving the efficiency of the scheduling process.

### **Zakariya Anwar (FA21-BSE-042)**

#### **Use Case: Login**

The admin, faculty and student can interact with login use case. This use case allows the user to interact with the system through their personal username and password. The user provides their login credentials (username and password) on the system's login page. The system verifies the credentials and grants access if they are correct. If the credentials are incorrect, the system displays an error message and prompts the user to try again.

#### **Use Case: View Timetable**

View timetable use case allows the user navigates to the timetable page of the system. The system displays the user's timetable in a clear and organized format. The user can view their timetable for a specific day and time having room no. The user can scroll or navigate through the timetable to view different periods or days. The main actors are admin, faculty and student.

#### **Use Case: Faculty timetable view**

The faculty timetable view use case allows faculty members to view their schedules for the week, including details such as class timings, room, and the courses they will be teaching. This use case helps faculty members to stay organized and manage their time effectively, ensuring they can provide the best possible education for their students.

#### **Use Case: Student timetable view**

The Student Timetable View use case allows a student to view their personal class schedule, including courses, teachers, and locations, in a visual format. sThis use case enhances the student's ability to plan and manage their time effectively, ensuring they attend all necessary classes and avoid scheduling conflicts.

#### **Use Case: Send notification**

The Send Notification use case involves the system monitoring for any modifications made to a specific entity or object. If a modification is detected, the system sends a notification to relevant users (students and faculty), informing them of the change. The notification may contain details about change of time or room changes etc. This use case helps keep users and stakeholders informed and up-to-date about changes in the system.

#### **Use Case: Generate request to modify**

The Generate Request to Modify use case involves a user requesting to modify an existing record or data within the system. The user fills out a modification request form, which includes details such as the reason for the modification and the proposed changes. The system generates a modification request that is sent to an administrator for review and approval. Once approved, the modifications are made to the record or data within the system, and the user is notified of the update.

### **Usama Ijaz (FA21-BSE-158)**

#### **Use Case: Create Timetable**

* Actors: Admin
* Brief: This use case allows the Admin to create a new timetable for a particular semester. The Admin will select the courses, faculties, and timeslots for the timetable.

#### **Use Case: Clash detection**

* Actors: Admin
* Brief: This use case allows the Admin to detect any clashes between the courses in the timetable. The system will check for conflicts in course timings, faculty availability, and room availability.

#### **Use Case: Set Courses**

* Actors: Admin
* Brief: This use case allows the Admin to add or remove courses from the timetable. The Admin will select the courses and specify the details such as the course code, name, and credit hours.

#### **Use Case: Allot rooms**

* Actors: Admin
* Brief: This use case allows the Admin to assign rooms to the courses in the timetable. The Admin will select the courses and rooms and specify the details such as the room number and capacity.

#### **Use Case: Change course room**

* Actors: Admin
* Brief: This use case allows the Admin to change the room assigned to a particular course. The Admin will select the course and the new room and update the details in the system.

#### **Use Case: Modify course timing**

* Actors: Admin
* Brief: This use case allows the Admin to modify the timings of a particular course. The Admin will select the course and specify the new timings such as the start time and end time. The system will automatically update the timetable accordingly.

## **Fully Dressed Use Cases**

### **Usama Ijaz (FA21-BSE-158)**

#### **Use Case UC 1: Create Timetable**

**Scope:** Timetable Management System

**Level:** User goal

**Primary Actor**: Admin

**Stakeholders and Interests:**

• Admin: Wants to create a clear and conflict-free timetable efficiently.

• Faculty: Wants to have a well-defined timetable for their courses.

• Students: Want to have an easy-to-follow and conflict-free timetable.

• System: Wants to ensure that the timetable is created without any discrepancies and conflicts.

• Administration: Wants to ensure the system remains efficient and well-managed.

**Preconditions:**

The Admin is authenticated and authorized to create a new timetable.

**Success Guarantee (or Postconditions):**

The new timetable is created and published to all stakeholders without any conflicts.

**Main Success Scenario (or Basic Flow):**

1. The Admin navigates to the "Create Timetable" feature in the system.

2. The system presents a form for entering the timetable details, including semester, course codes, faculty, and timings.

3. The Admin enters the required details in the form.

4. The system validates the entered details and checks for any conflicts with existing timetables or courses.

5. If there are no conflicts, the system generates the new timetable and publishes it to all stakeholders.

6. If there are conflicts, the system alerts the Admin and provides suggestions for resolving the conflicts.

7. The Admin either accepts the suggested changes or makes manual changes to resolve the conflicts.

8. The system generates the new timetable and publishes it to all stakeholders.

**Extensions (or Alternative Flows):**

If the Admin enters incorrect or invalid details in the form:

1. The system displays an error message to the Admin, prompting them to correct the errors and re-submit the form.

2. The system does not create a new timetable until the entered details are validated and corrected.

**Special Requirements:**

• The system should be able to handle conflicts and suggest solutions efficiently.

• The system should be able to validate the entered details accurately.

• The system should be able to publish the new timetable to all stakeholders efficiently.

• The system should be able to handle errors and exceptions gracefully.

**Technology and Data Variations List:**

• The form for entering timetable details may vary based on the system design and user interface.

• The validation and conflict-checking algorithms may vary based on the system design and implementation.

• The publishing system may vary based on the system design and implementation.

Frequency of Occurrence: The creation of new timetables could happen multiple times a year, depending on the frequency of course schedules and timetable updates.

**Open Issues:**

• The system should be able to handle multiple users creating new timetables simultaneously without conflicts.

• The system should be able to integrate with other modules of the timetable management system for seamless updates.

#### **Use Case UC 2: Clash Detection**

**Scope:** Timetable Management System

**Level**: User Goal

**Primary Actor:** Admin

**Stakeholders and Interests:**

• Admin: Wants to detect and resolve conflicts between courses and slots efficiently.

• Faculty: Wants to have a well-defined and conflict-free timetable for their courses.

• Students: Want to have an easy-to-follow and conflict-free timetable.

• System: Wants to ensure that conflicts between courses and slots are detected and resolved without any discrepancies.

• Administration: Wants to ensure the system remains efficient and well-managed.

**Preconditions:**

• The Admin is authenticated and authorized to detect and resolve conflicts.

• The timetable, course, and slot data is updated and accurate.

**Success Guarantee (or Postconditions):**

• Conflicts between courses and slots are detected and resolved without any discrepancies.

**Main Success Scenario (or Basic Flow):**

1. The Admin navigates to the "Clash Detection" feature in the system.

2. The system presents a list of all courses and slots.

3. The Admin selects a course or slot to check for conflicts.

4. The system checks for any overlaps between the selected course/slot and other courses/slots.

5. If there are no conflicts, the system notifies the Admin that there are no clashes.

6. If there are conflicts, the system presents a list of conflicting courses/slots.

7. The Admin selects a conflicting course/slot to resolve the conflict.

8. The system presents options for resolving the conflict, such as changing the course/slot timing or room.

9. The Admin selects an option to resolve the conflict.

10. The system updates the timetable with the resolved conflict and notifies the Admin of the successful resolution.

**Extensions (or Alternative Flows):**

- If the system encounters an error during clash detection:

1. The system displays an error message to the Admin and prompts them to retry the operation.

- If the Admin selects multiple courses/slots to check for conflicts:

1. The system checks for overlaps between the selected courses/slots and other courses/slots.

2. The system presents a list of conflicting courses/slots.

3. The Admin selects a conflicting course/slot to resolve the conflict.

4. The system presents options for resolving the conflict, such as changing the course/slot timing or room.

5. The Admin selects an option to resolve the conflict.

6. The system updates the timetable with the resolved conflict and notifies the Admin of the successful resolution.

**Special Requirements:**

• The system should be able to detect conflicts accurately and efficiently.

• The system should provide options to resolve conflicts.

• The system should be able to update the timetable with resolved conflicts.

• The system should be able to handle errors and exceptions gracefully.

**Technology and Data Variations List:**

• The system design and user interface may vary for clash detection feature.

• The clash detection algorithm may vary based on the system design and implementation.

• The notification system may vary based on the system design and implementation.

Frequency of Occurrence:

• Clash detection could happen multiple times a day, depending on the frequency of course schedules and timetable updates.

**Open Issues:**

• The system should be able to handle multiple users detecting and resolving conflicts simultaneously without conflicts.

• The system should be able to integrate with other modules of the timetable management system for seamless updates.

#### **Use Case UC 3: Set Courses**

**Scope**: Timetable Management System

**Level**: User Goal

**Primary Actor:** Admin

**Stakeholders and Interests:**

• Admin: Wants to set up courses and their schedules efficiently.

• Faculty: Wants to have a well-defined and accurate schedule for their courses.

• Students: Want to have access to the course schedule and enroll in courses.

• System: Wants to ensure that courses and schedules are set up without any discrepancies.

• Administration: Wants to ensure the system remains efficient and well-managed.

**Preconditions:**

• The Admin is authenticated and authorized to set up courses and schedules.

• The course data is available, including the course name, code, description, and credit hours.

• The faculty data is available, including the faculty name, code, and department.

• The room data is available, including the room number, capacity, and type.

**Success Guarantee (or Postconditions):**

• Courses and their schedules are set up accurately and efficiently without any discrepancies.

**Main Success Scenario (or Basic Flow):**

1. The Admin navigates to the "Set Courses" feature in the system.

2. The system presents a form to add a new course.

3. The Admin fills in the course details, including the course name, code, description, and credit hours.

4. The Admin selects the faculty member responsible for the course.

5. The system presents a list of available rooms.

6. The Admin selects a room for the course.

7. The Admin sets the course schedule, including the day and time for the course.

8. The Admin saves the course and its schedule.

9. The system confirms the successful creation of the course and schedule.

**Extensions (or Alternative Flows):**

- If the Admin wants to modify an existing course schedule:

1. The Admin navigates to the "Set Courses" feature in the system.

2. The system presents a list of all courses and schedules.

3. The Admin selects the course to modify.

4. The system presents the existing schedule for the selected course.

5. The Admin modifies the schedule for the course.

6. The Admin saves the modified course schedule.

7. The system confirms the successful modification of the course schedule.

**Special Requirements:**

• The system should allow the Admin to add, modify, and delete courses and their schedules.

• The system should validate the course data, faculty data, and room data before allowing the Admin to save the course and schedule.

• The system should prevent scheduling conflicts between courses and rooms.

• The system should be able to handle errors and exceptions gracefully.

**Technology and Data Variations List:**

• The system design and user interface may vary for course setup and scheduling feature.

• The course and faculty data may be imported from other systems or databases.

• The room data may be imported or entered manually.

Frequency of Occurrence:

• Course setup and scheduling could happen multiple times a day, depending on the frequency of new courses and updates.

**Open Issues:**

• The system should be able to handle multiple users adding, modifying, and deleting courses and schedules simultaneously without conflicts.

• The system should be able to integrate with other modules of the timetable management system for seamless updates.

#### **Use Case UC 4: Allot Rooms**

**Scope:** Timetable Management System

**Level**: User Goal

**Primary Actor:** Admin

**Stakeholders and Interests:**

• Admin: Wants to efficiently assign rooms to courses.

• Faculty: Wants to have appropriate rooms assigned to their courses.

• Students: Want to have well-defined and easy-to-follow class schedules.

• System: Wants to ensure that rooms are assigned accurately and efficiently.

• Administration: Wants to ensure the system remains efficient and well-managed.

**Preconditions:**

• The Admin is authenticated and authorized to allot rooms.

• The course and room data is updated and accurate.

**Success Guarantee (or Postconditions):**

• Courses are assigned to appropriate rooms efficiently.

**Main Success Scenario (or Basic Flow):**

1. The Admin navigates to the "Allot Rooms" feature in the system.

2. The system presents a list of all courses that require rooms to be assigned.

3. The Admin selects a course to assign a room.

4. The system presents a list of available rooms that meet the course requirements.

5. The Admin selects a room to assign to the course.

6. The system updates the timetable with the assigned room and notifies the Admin of the successful assignment.

7. The Admin repeats steps 3-6 for all courses that require rooms to be assigned.

**Extensions (or Alternative Flows):**

- If the system encounters an error during room allotment:

1. The system displays an error message to the Admin and prompts them to retry the operation.

- If there are no available rooms that meet the course requirements:

1. The system displays a message to the Admin that no rooms are available for the course.

**Special Requirements:**

• The system should be able to efficiently assign rooms to courses.

• The system should provide a list of available rooms that meet the course requirements.

• The system should be able to update the timetable with assigned rooms.

• The system should be able to handle errors and exceptions gracefully.

**Technology and Data Variations List:**

• The system design and user interface may vary for room allotment feature.

• The room allotment algorithm may vary based on the system design and implementation.

• The notification system may vary based on the system design and implementation.

**Frequency of Occurrence:**

• Room allotment could happen multiple times a day, depending on the frequency of course schedules and timetable updates.

**Open Issues:**

• The system should be able to handle multiple users assigning rooms simultaneously without conflicts.

• The system should be able to integrate with other modules of the timetable management system for seamless updates.

#### **Use Case UC 5: Change Course Room**

**Scope:** Timetable Management System

**Level**: User Goal

**Primary Actor:** Admin

**Stakeholders and Interests:**

• Admin: Wants to efficiently modify course rooms.

• Faculty: Wants to have appropriate rooms assigned to their courses.

• Students: Want to have well-defined and easy-to-follow class schedules.

• System: Wants to ensure that course rooms are modified accurately and efficiently.

• Administration: Wants to ensure the system remains efficient and well-managed.

**Preconditions:**

• The Admin is authenticated and authorized to modify course rooms.

• The course and room data is updated and accurate.

**Success Guarantee (or Postconditions):**

• Course rooms are modified efficiently.

**Main Success Scenario (or Basic Flow):**

1. The Admin navigates to the "Change Course Room" feature in the system.

2. The system presents a list of all courses and their assigned rooms.

3. The Admin selects a course to modify the room.

4. The system presents a list of available rooms that meet the course requirements.

5. The Admin selects a new room to assign to the course.

6. The system updates the timetable with the new room assignment for the selected course.

The system notifies the Admin of the successful room change.

**Extensions (or Alternative Flows):**

- If the system encounters an error during room change:

1. The system displays an error message to the Admin and prompts them to retry the operation.

- If the Admin selects a course with conflicting schedule in the new room:

1. The system displays a warning message to the Admin.

2. The Admin may choose to select a different room or modify the course schedule to resolve the conflict.

**Special Requirements:**

• The system should be able to efficiently present a list of available rooms that meet the course requirements.

• The system should be able to update the timetable with the new room assignment accurately and efficiently.

• The system should be able to handle errors and exceptions gracefully.

• The system should ensure that room changes do not cause conflicts with other courses or schedules.

**Technology and Data Variations List:**

• The system design and user interface may vary for the Change Course Room feature.

• The available room selection algorithm may vary based on the system design and implementation.

• The notification system may vary based on the system design and implementation.

Frequency of Occurrence:

• Room changes could happen multiple times a day, depending on the frequency of course schedules and timetable updates.

**Open Issues:**

• The system should be able to handle multiple users modifying course rooms simultaneously without conflicts.

• The system should be able to integrate with other modules of the timetable management system for seamless updates.

#### **Use Case UC 6: Modify Course Timing**

**Scope:** Timetable Management System

**Level:** User Goal

**Primary Actor:** Faculty

**Stakeholders and Interests:**

• Faculty: Wants to modify the timing of their courses according to their availability and preferences.

• Students: Want to have well-defined and easy-to-follow class schedules.

• Admin: Wants to efficiently manage course timings and ensure a balanced timetable.

• System: Wants to ensure that course timings are modified accurately and efficiently.

• Administration: Wants to ensure the system remains efficient and well-managed.

**Preconditions:**

• The faculty member is authenticated and authorized to modify course timings.

• The course and room data is updated and accurate.

• The faculty member has access to their course schedules.

**Success Guarantee (or Postconditions):**

• Course timings are modified efficiently.

**Main Success Scenario (or Basic Flow):**

The faculty member navigates to the "Modify Course Timing" feature in the system.

The system presents a list of all courses assigned to the faculty member.

The faculty member selects a course to modify the timing.

The system presents the current course timing.

The faculty member modifies the course timing according to their availability and preferences.

The system updates the timetable with the new course timing.

The system notifies the faculty member of the successful course timing modification.

**Extensions (or Alternative Flows):**

- If the system encounters an error during course timing modification:

1. The system displays an error message to the faculty member and prompts them to retry the operation.

**Special Requirements**:

• The system should be able to efficiently present the current course timing and allow modifications.

• The system should be able to update the timetable with the new course timing accurately and efficiently.

• The system should be able to handle errors and exceptions gracefully.

**Technology and Data Variations List:**

• The system design and user interface may vary for the Modify Course Timing feature.

• The notification system may vary based on the system design and implementation.

Frequency of Occurrence:

• Course timing modifications could happen multiple times a day, depending on faculty availability and preferences.

**Open Issues:**

• The system should be able to handle multiple users modifying course timings simultaneously without conflicts.

• The system should be able to integrate with other modules of the timetable management system for seamless updates.

### **Saud Khan (FA21-BSE-033)**

#### **Use Case UC 7: Create Slot**

**Scope:** Timetable Management System

**Level:** User goal

**Primary Actor:** Admin

**Stakeholders and Interests:**

* Admin: Wants to create a new time slot accurately and efficiently without any errors.
* Faculty: Wants to have a clear and well-defined timetable.
* Students: Want to have a conflict-free and easy-to-follow timetable.
* System: Wants to ensure that the new time slot is created without any discrepancies and conflicts.
* Administration: Wants to ensure the system remains efficient and well-managed.

**Preconditions:** The Admin is authenticated and authorized to create a new slot.

Success Guarantee (or Postconditions): The new time slot is created and added to the timetable without any conflicts.

**Main Success Scenario (or Basic Flow):**

1. The Admin navigates to the "Create Slot" feature in the system.
2. The system presents a form for entering the slot details, including start and end time, course name, course code, section, and instructor name.
3. The Admin enters the required details in the form.
4. The system validates the entered details and checks for any conflicts with existing slots.
5. If there are no conflicts, the system adds the new slot to the timetable and notifies the Admin of the successful creation of the new slot.
6. If there are conflicts, the system alerts the Admin and provides suggestions for resolving the conflicts.
7. The Admin either accepts the suggested changes or makes manual changes to resolve the conflicts.
8. The system updates the timetable with the new slot and notifies the Admin of the successful creation of the new slot.

**Extensions (or Alternative Flows):**

If the Admin enters incorrect or invalid details in the form:

1. The system displays an error message to the Admin, prompting them to correct the errors and re-submit the form.
2. The system does not create a new slot until the entered details are validated and corrected.

**Special Requirements:**

* The system should be able to handle conflicts and suggest solutions efficiently.
* The system should be able to validate the entered details accurately.
* The system should be able to provide notifications to the Admin upon successful creation of a new slot.
* The system should be able to handle errors and exceptions gracefully.

**Technology and Data Variations List:**

* The form for entering slot details may vary based on the system design and user interface.
* The validation and conflict-checking algorithms may vary based on the system design and implementation.
* The notification system may vary based on the system design and implementation.

Frequency of Occurrence: The creation of new slots could happen multiple times a day, depending on the frequency of course schedules and timetable updates.

**Open Issues:**

* The system should be able to handle multiple users creating new slots simultaneously without conflicts.
* The system should be able to integrate with other modules of the timetable management system for seamless updates.

#### **Use Case UC 8: Update Slot**

**Scope:** Timetable Management System

**Level:** User goal

**Primary Actor:** Admin

**Stakeholders and Interests:**

* Administrator: Wants to update an existing slot with accurate and fast entry, and no errors, to avoid any scheduling conflicts.
* Lecturer: Wants to be informed of any changes made to their teaching schedules.
* Students: Want to be informed of any changes made to their course schedules.
* Department: Wants to ensure that all course schedules are up-to-date and accurate.
* System: Wants to ensure that all updates are recorded accurately and updated in real-time.

**Preconditions:**

* Administrator is authenticated and authorized to make updates to the timetable management system.
* The slot to be updated exists in the system.

**Success Guarantee (or Postconditions):**

* The slot is updated with the new information.
* The system sends notifications to the relevant stakeholders, including lecturers and students.
* The system updates the timetable with the new information.

**Main Success Scenario (or Basic Flow):**

1. Administrator selects the slot to be updated from the timetable.
2. The system displays the existing details of the selected slot.
3. Administrator modifies the slot details as required (e.g. course code, venue, time, lecturer).
4. The system validates the updated details.
5. Administrator confirms the changes.
6. The system updates the slot with the new details.
7. The system sends notifications to the relevant stakeholders.
8. The system updates the timetable with the new information.

**Extensions (or Alternative Flows):**

Administrator cancels the update process:

1. The system cancels the update process.
2. The system displays the original details of the slot.

**Special Requirements:**

* The system must ensure that there are no scheduling conflicts with the updated slot.
* The system must have real-time updates to ensure that stakeholders receive notifications promptly.
* The system must have a user-friendly interface for the administrator to make updates.

**Technology and Data Variations List:**

* The system must be able to handle different course codes, venues, times, and lecturers.
* The system must be able to send notifications via email or text message.
* The system must be able to handle different scheduling rules and constraints.

Frequency of Occurrence:

* The frequency of occurrence will depend on the number of changes made to the timetable.

**Open Issues:**

* The system should have a log of all updates made to the timetable for auditing purposes.
* The system should have a backup and recovery plan in case of system failure during the update process.

#### **Use Case UC 9: Delete Slot**

**Scope:** Timetable Management System

**Level:** User goal

**Primary Actor:** Admin

**Stakeholders and Interests:**

* Administrator: Wants to delete a slot from the timetable without errors or inconsistencies.
* Lecturer: Wants to ensure that the timetable is up-to-date.
* Student: Wants an accurate timetable that reflects the available classes.
* Department: Wants to ensure that the timetable is consistent with the course offerings and the availability of faculty members.
* System: Wants to maintain the integrity of the timetable by deleting the slot without affecting any other part of the system.

**Preconditions:**

* Administrator is identified and authenticated.
* A slot exists in the timetable.

**Success Guarantee (or Postconditions):**

* Slot is deleted from the timetable.

**Main Success Scenario (or Basic Flow):**

1. Administrator selects the slot to be deleted from the timetable.
2. System verifies that the slot can be deleted and shows a confirmation message to the Administrator.
3. Administrator confirms the deletion of the slot.
4. System deletes the slot from the timetable and updates the database.
5. System shows a success message to the Administrator.

**Extensions (or Alternative Flows):**

If the slot cannot be deleted:

1. System shows an error message to the Administrator.
2. Administrator chooses to either select another slot or cancel the operation.

**Special Requirements:**

* The system must prevent the deletion of a slot that is already in use by a class or an instructor.
* The system must ensure that the deletion of a slot does not affect any other part of the timetable.

**Technology and Data Variations List:**

* The system can be accessed through a web interface or a mobile application.
* The database can be hosted on a local server or a cloud-based platform.

Frequency of Occurrence: Depends on the need to update the timetable.

.

**Open Issues:**

* Should the system allow the deletion of a slot that is part of a repeating schedule?
* How will the system handle the deletion of a slot that has already been assigned to a class or an instructor?

#### **Use Case UC 10: Add Slot**

**Scope:** Timetable Management System

**Level:** User goal

**Primary Actor:** Admin

**Stakeholders and Interests:**

* Administrator: Wants to add a new slot in the timetable accurately and efficiently.
* Faculty: Wants to know the updated timetable and course offerings.
* Students: Want to know the updated timetable and available slots for courses.
* Company: Wants to ensure that the timetable is up-to-date and that there are no conflicts in the schedule.

**Preconditions:**

Administrator is identified and authenticated. The timetable system is accessible.

**Success Guarantee (or Postconditions):**

New slot is added to the timetable without conflicts. The updated timetable is accessible to all relevant stakeholders.

**Main Success Scenario (or Basic Flow):**

1. Administrator accesses the timetable system.
2. Administrator selects the option to add a new slot.
3. System prompts Administrator to input the course code, date, time, venue, and other necessary details.
4. Administrator enters the required information.
5. System checks for conflicts with existing slots.
6. If there are no conflicts, the new slot is added to the timetable.
7. System displays a confirmation message that the new slot has been added.
8. System updates the timetable accessible to all relevant stakeholders.

**Extensions (or Alternative Flows):**

If there are conflicts with existing slots:

1. System displays an error message indicating the conflict.
2. Administrator is prompted to modify the slot details.
3. Administrator modifies the details until there are no conflicts.
4. System adds the new slot to the timetable and displays a confirmation message.

**Special Requirements:**

* System should have the ability to detect conflicts in the timetable.
* System should have a user-friendly interface to input new slot details..

**Technology and Data Variations List:**

* Date and time formats may vary depending on the location.
* Venue details may vary depending on the campus and room availability.

Frequency of Occurrence: Occurs regularly when new courses or changes in schedules are implemented

**Open Issues:**

* The system should have a feature to delete a slot if necessary.
* The system should have the ability to notify relevant stakeholders of any changes in the timetable.

#### **Use Case UC 11: Give course preferences**

**Scope:** Timetable Management System

**Level:** User goal

**Primary Actor:** Faculty

**Stakeholders and Interests:**

* Faculty: Wants to give course preferences accurately and efficiently.
* Department Head: Wants to collect course preferences from faculty members in a timely and organized manner.
* Timetable Coordinator: Wants to create a timetable with minimum conflicts and optimize room allocation.
* Students: Want a timetable with minimal conflicts and an organized schedule.
* University Administration: Wants to ensure efficient scheduling and use of resources.

**Preconditions:**

* Faculty member is identified and authenticated.
* The timetable system is operational.

**Success Guarantee (or Postconditions):**

* Course preferences are saved accurately.
* The department head is notified of the faculty member’s course preferences.

**Main Success Scenario (or Basic Flow):**

1. Faculty member logs into the timetable system.
2. System presents the faculty member with a list of courses to choose from.
3. Faculty member selects their preferred courses from the list.
4. System saves the selected course preferences.
5. System sends a notification to the department head confirming the successful submission of course preferences.

**Extensions (or Alternative Flows):**

Faculty member encounters an issue while submitting course preferences:

1. Faculty member contacts the Timetable Coordinator for assistance.
2. Timetable Coordinator provides assistance to resolve the issue.

**Special Requirements:**

* The system should be user-friendly and easy to navigate.
* The list of courses should be up-to-date and accurate.

**Technology and Data Variations List:**

* The system should be accessible via desktop or mobile devices.
* The list of courses should be updated regularly.

Frequency of Occurrence: Once per semester.

**Open Issues:**

* What happens if a faculty member fails to submit their course preferences on time?
* How will the department head and Timetable Coordinator handle conflicts between faculty member’s course preferences?

#### **Use Case UC 12: Select Course Preferences**

**Scope:** Timetable Management System

**Level:** User goal

**Primary Actor:** Admin

**Stakeholders and Interests:**

* Admin: Wants to select the course preferences accurately and efficiently.
* Faculty: Wants to ensure that the course preferences are aligned with the available teaching staff.
* Students: Wants the courses to be scheduled at convenient times.
* Company: Wants to have an optimized schedule of courses that satisfies the interests of students and faculty.
* Manager: Wants to be able to monitor and ensure the smooth functioning of the course scheduling process.

**Preconditions:**

Admin is identified and authenticated.

**Success Guarantee (or Postconditions):**

Course preferences are saved. The course schedule is optimized and aligned with the available teaching staff. The interests of students and faculty are taken into account.

**Main Success Scenario (or Basic Flow):**

1. Admin accesses the course preference selection interface.
2. System presents the available courses and the relevant parameters such as the number of students, course duration, and available teaching staff.
3. Admin selects the preferred courses based on the given parameters.
4. System validates the selected course preferences.
5. System generates the course schedule based on the selected preferences and the available teaching staff.
6. System presents the optimized course schedule for review and approval by the Admin.
7. Admin approves the course schedule.
8. System saves the course preferences and the approved course schedule.

**Extensions (or Alternative Flows):**

At any time, Admin requests to edit the course preferences:

1. System allows Admin to edit the selected course preferences.
2. System validates the edited course preferences.
3. System generates the course schedule based on the edited preferences and the available teaching staff.
4. System presents the optimized course schedule for review and approval by the Admin.
5. Admin approves the course schedule.
6. System saves the edited course preferences and the approved course schedule.

**Special Requirements:**

* User-friendly interface for course preference selection.
* The system should be able to handle a large number of courses and parameters.
* The system should be able to optimize the course schedule based on the available teaching staff.
* The system should take into account the interests of students and faculty.
* The system should allow for easy editing of course preferences.

**Technology and Data Variations List:**

* The system should be able to handle different course parameters such as number of students, course duration, and available teaching staff.
* The system should be able to handle different course scheduling algorithms.
* The system should be able to integrate with the faculty and student database.

Frequency of Occurrence: Once per semester.

**Open Issues:**

* How to handle conflicts between course preferences and available teaching staff?
* How to handle course preferences for online or hybrid courses?

### **Zakariya Anwar Khan (FA21-BSE-042)**

#### **Use Case UC 13: Login**

**Scope:**

Timetable Management System

**Level:**

User goal

**Primary Actor:**

User

**Stakeholders and Interest:**

* **User:** Wants to access the Timetable Management System and perform various tasks.
* **Admin:** Wants to ensure that only authorized users can access the system.

**Preconditions:**

* The Timetable Management System is installed and running.
* The user has a valid username and password.

**Success Guarantee:**

* The user is authenticated and granted access to the system.

**Main Success Scenario:**

* The user opens the login page of the Timetable Management System
* The system presents a form with fields for the username and password.
* The user enters their username and password.
* The system verifies the entered credentials with the stored user data.
* If the credentials are correct, the system grants the user access to the system and redirects them to the home page.

**Extensions:**

* If the user enters an incorrect username or password, the system displays an error message and asks the user to re-enter their credentials.
* If the user enters an invalid username or password multiple times, the system locks their account for a specified period.

**Special Requirements:**

* The system should use encryption to securely store user credentials.
* The system should have a password recovery mechanism in case the user forgets their password.

**Technology and Data Variation List:**

* The login process may vary depending on the type of authentication mechanism used, such as LDAP or OAuth.
* The system may use different databases to store user data.

**Open Issues:**

The Login use case is executed every time a user wants to access the Timetable Management System.

#### **Use Case UC 14: View Timetable**

**Scope:**

Timetable Management System

**Level:**

User-goal

**Primary Actor:**

Student or Faculty

**Stakeholders and Interest:**

* Student and Faculty: need to view their class schedule to manage their time and attend classes on time.
* Admin: responsible for managing the timetable and ensuring it is up to date.

**Preconditions:**

* The user must be logged in to the system.
* The timetable must be created and published by the administrator.

**Success Guarantee:**

* The user can view their timetable without any errors and the timetable is up to date.

**Main Success Scenario:**

* The user navigates to the "Timetable" section of the system.
* The system displays the current week's timetable.
* The user selects a different week or month to view the timetable.
* The system displays the selected timetable.

**Extensions:**

* If the user is not logged in, the system prompts them to log in.
* If there is no timetable available for the selected week or month, the system displays a message indicating that there are no classes scheduled.
* If there is an error in loading the timetable, the system displays an error message and prompts the user to try again.

**Special Requirements:**

* The system should display the timetable in a clear and easy-to-read format.
* The system should allow users to filter the timetable by course, instructor, or location.

**Technology and Data Variation List:**

* The timetable data may vary based on the academic calendar or any changes made by the administrator.
* The system should be accessible on desktop and mobile devices.

**Open Issues:**

The view timetable use case may occur multiple times per day for each user, depending on their schedule and need to access the timetable.

#### **Use Case UC15: Faculty Timetable View**

**Scope:**

Timetable Management System

**Level:**

User-goal

**Primary Actor:**

Faculty

**Stakeholders and Interest:**

* Faculty Member: Wants to view their timetable to know their schedule.
* Admin: Wants to ensure that faculty members can view their timetable accurately and easily.

**Preconditions:**

* Faculty member must be registered in the system.
* The timetable must have already been created.
* The faculty member must be logged in to the system.

**Success Guarantee:**

The faculty member can view their timetable.

**Main Success Scenario:**

* The faculty member logs in to the system.
* The faculty member navigates to the "Timetable" section of the system.
* The system displays the faculty member's timetable, including the courses they are teaching and the times of those courses.
* The faculty member can filter the timetable based on different parameters such as date, course name, and course time.
* The faculty member can print or download their timetable if they wish to have a physical copy.

**Extensions:**

* If the faculty member is not registered in the system, an error message is displayed, and the faculty member is prompted to register.
* If the timetable has not been created yet, an error message is displayed, and the faculty member is prompted to contact the timetable administrator.
* If the faculty member is not logged in to the system, they are prompted to log in before they can view their timetable.

**Special Requirements:**

None

**Technology and Data Variation List:**

* The timetable may vary depending on the courses and times offered each semester.
* The system should be accessible via a web browser on a computer or mobile device.

**Open Issues:**

The faculty member may view their timetable multiple times throughout the semester.

#### **Use Case UC16: Student Timetable View**

**Scope:**

Timetable Management System

**Level:**

User-goal

**Primary Actor:**

Student

**Stakeholders and Interest:**

* Student: Want to view their personal timetable.
* Lecturers: Want to ensure correct timetable information is displayed to students
* Admin: Want to ensure the system is functioning correctly and that the correct timetables are displayed to students.

**Preconditions:**

* The student must have an account in the system and be logged in.
* The student must have enrolled in at least one course.
* The timetable for the current academic session must have been created and uploaded into the system.

**Success Guarantee:**

The student can view their personal timetable accurately and without errors.

**Main Success Scenario:**

* The student logs into the system.
* The system displays the student's dashboard.
* The student clicks on the "Timetable" button.
* The system displays the student's personal timetable for the current academic session.
* The student can view the courses they are enrolled in and the corresponding class schedules for each course.
* The student can view any changes made to the timetable and is notified of any cancellations or rescheduling of classes.

**Extensions:**

* If the student is not logged in, the system displays an error message and prompts the student to log in.
* If the student has not enrolled in any courses, the system displays an error message informing the student to enroll in at least one course.
* If there is no timetable uploaded for the current academic session, the system displays an error message informing the student to contact the administrator.
* If there are any errors in the timetable, such as scheduling conflicts or missing information, the system displays an error message and prompts the student to contact the administrator.

**Special Requirements:**

* The timetable must be updated in real-time to reflect any changes made to class schedules.
* The system must be secure to ensure the privacy of student information.

**Technology and Data Variation List:**

* The system must be compatible with various web browsers and operating systems.
* The system must be able to handle a large amount of data for multiple students and courses.

**Open Issues:**

The use case will occur frequently, as students will need to check their timetables regularly throughout the academic session.

#### **Use Case UC17: Send Notification**

**Scope:**

Timetable Management System

**Level:**

User-goal

**Primary Actor:**

Admin

**Stakeholders and Interest:**

* Admin: Wants to send timely and relevant notifications to all users of the system
* Students: Want to receive important notifications related to class schedules and changes in the timetable.
* Faculty : Want to receive timely notifications related to class schedules, cancellations and changes in the timetable.

**Preconditions:**

* The Administrator has a valid login to the Timetable Management System.
* There are at least one or more users of the system who have subscribed to notifications.

**Success Guarantee:**

All subscribed users receive timely and relevant notifications without delay

**Main Success Scenario:**

* The Administrator logs in to the Timetable Management System and navigates to the 'Send Notification' page
* The Administrator selects the target recipients for the notification (e.g., students, teachers, or both)
* The Administrator composes the notification message, including relevant information (e.g., class name, date, time, and reason for the notification)
* The Administrator verifies the content of the notification and selects the send option.
* The System sends the notification to all subscribed users in the selected recipient group without delay.

**Extensions:**

* If there are no subscribed users in the selected recipient group, the System displays an error message and prompts the Administrator to select a different recipient group.
* If the notification message contains inappropriate content or violates the policies of the Timetable Management System, the System displays an error message and does not send the notification.

**Special Requirements:**

* The System must support multiple notification channels (e.g., email, SMS, or push notifications) to cater to the preferences of different users.
* The notification message should be concise and relevant to avoid overwhelming users with unnecessary information.

**Technology and Data Variation List:**

* The System must support various data formats (e.g., text, images, and videos) to enhance the clarity and effectiveness of the notification.
* The System must be able to handle a large volume of notifications without affecting the performance or stability of the System.

**Open Issues:**

The frequency of notification sending will depend on the availability of new information or updates that need to be communicated to users. It can range from multiple times per day to several times per week.

#### **Use Case UC18: Generate Request to Modify**

**Scope:**

Timetable Management System

**Level:**

User-goal

**Primary Actor:**

Faculty

**Stakeholders and Interest:**

* Faculty: They need to modify the timetable for a course(s) due to various reasons such as personal commitments, changes in course content, etc.
* Students: They are affected by any modifications to the timetable as it can impact their schedule and availability.
* Admin: They are responsible for reviewing and approving the requests for modifications made by the Course Instructors.

**Preconditions:**

* The Course Instructor must be logged into the Timetable Management System.
* The Course Instructor must have the necessary permissions to generate a request to modify the timetable.

**Success Guarantee:**

* A request to modify the timetable will be generated and sent to the Timetable Administrator for review and approval.

**Main Success Scenario:**

* The Course Instructor logs into the Timetable Management System.
* The Course Instructor selects the course or courses for which they want to generate a request to modify the timetable.
* The Course Instructor provides the necessary details for the modification request, such as the reason for modification, the preferred date and time for the class, etc.
* The Course Instructor submits the modification request.
* The Timetable Management System generates a notification to the Timetable Administrator regarding the modification request.
* The Timetable Administrator reviews the modification request and either approves or rejects it.
* If the modification request is approved, the Timetable Management System updates the timetable accordingly and notifies the Course Instructor and Students of the changes.
* If the modification request is rejected, the Timetable Management System sends a notification to the Course Instructor with the reason for rejection.

**Extensions:**

* If the Course Instructor is not logged into the Timetable Management System, they will be prompted to log in before they can generate a request to modify the timetable.
* If the Course Instructor does not have the necessary permissions to generate a request to modify the timetable, they will receive an error message and will not be able to proceed with the request.
* If the modification request conflicts with another course or event on the timetable, the Timetable Administrator may reject the request and ask for an alternative date and time

**Special Requirements:**

NONE

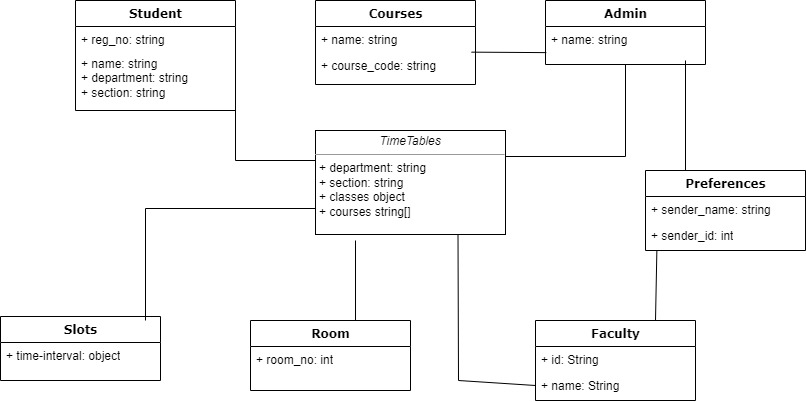
**Technology and Data Variation List:**

* The Timetable Management System must be accessible via a web browser or a mobile app.
* The modification request form must include fields for course name, date, time, reason for modification, and any additional comments.
* The Timetable Management System must be able to generate notifications and send emails to the Course Instructor, Students, and Timetable Administrator.

**Open Issues:**

This use case can occur multiple times a semester or academic year, depending on the frequency of course modifications requested by the Course Instructors.

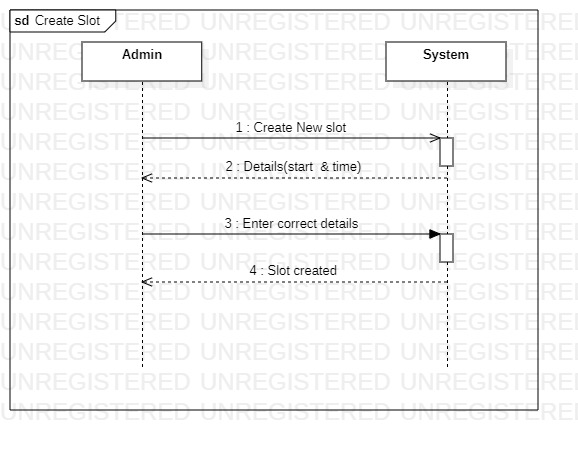
# **CHAPTER 3 DOMAIN MODEL**



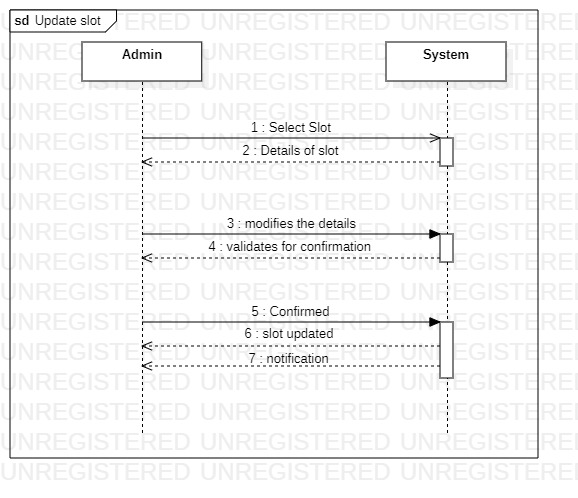
# **CHAPTER 4 SYSTEM SEQUENCE DIAGRAM**

## **Saud Khan:**

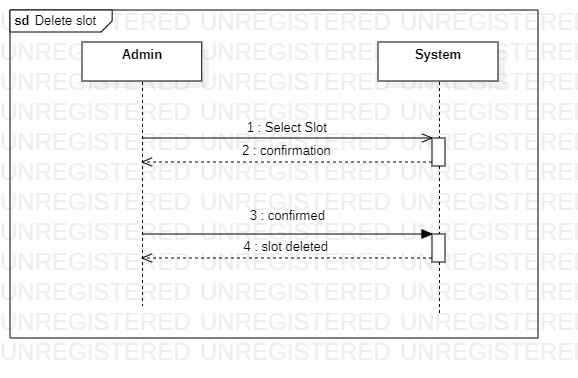
### **UC1: Create Slot**



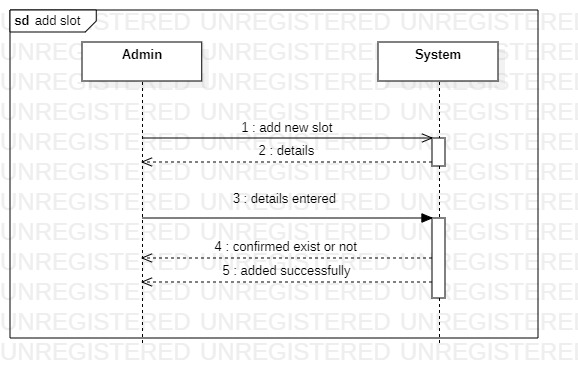
### **UC2: Update Slot**



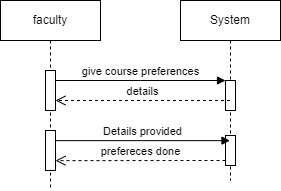
### **UC3: Delete Slot**



### **UC4: Add Slot**



### **UC3: Give Course Preferences**



### **UC6: Select Course Preferences**

