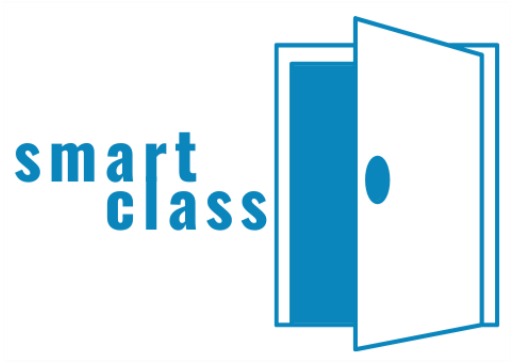
King Saud University

College of Computer and Information Sciences

Department of Computer Science

CSC343: System Analysis & Design

Final Report



Prepared by

Name	ID	Task Assigned
Shatha Alabbad	444200515	REQ-03, REQ-08, scenario 2 System structural diagram +SD for UC-2+Class diagram+ rest is equally disturbed
Nouf Aljuried	444201018	REQ-0, REQ-06, scenario 3 + SD for UC-5+block diagram + rest is equally disturbed
Maram Alsheddi	444200628	REQ-02, REQ-07, scenario 1 + SD for UC-1+Architectural Styles+ rest is equally disturbed
Reema Alsamrani	444201193	REQ-04, REQ-09, scenario 4+ SD for UC-7 System behavioral diagram + rest is equally disturbed

Table of contents

1. Project Proposal

In universities, both faculty members and students often struggle to find vacant classrooms quickly, especially in urgent situations or during free time. This issue results in wasted time and effort searching for suitable rooms, which impacts the efficiency of academic activities.

Universities often lack an efficient way to identify available classrooms at any given moment. Currently, faculty members and students rely on random inquiries or outdated paper schedules, making it difficult to quickly determine room availability. This problem affects:

• Faculty members: who may need an extra classroom on short notice.

• Students: who are looking for an available classroom to accommodate large groups of 4 or more for activities such as group study sessions or meetings.

Proposed Solution:

This app is a smart application designed to help users find available classrooms. It will be integrated with the university's system to ensure accurate updates on classroom schedules, course assignments, detailed map of each building, and room availability, allowing users to:

• Search for available classrooms by building, floor, room number, and capacity, as well as identifying the available technological resources.

• View the occupancy schedule for each classroom

• Receive real-time updates on room availability (busy/free)

• Facilitate advance booking of classrooms

Key Features:

• Role-based access:
Faculty members can register for any available classroom.
Students can only view classroom availability but cannot book rooms directly unless if the groups meet the limitations (specified below).
Users are limited to booking a classroom for a maximum of one hour, with the option to extend (if available).

• Classroom schedule tracking:
Both students and faculty can view the classrooms assigned to them for the semester.
Students receive notifications if their instructor changes their assigned classroom.

• Search functionality:
Students can search for any section's classroom using the section number, course name, or the instructor's name.

• Waitlist system:
If a classroom is fully booked, users can join a waitlist and receive notifications when it becomes available.

• Booking confirmation:
Users must confirm their reservation before the scheduled time; otherwise, it will be automatically canceled to free up the room for others.

• Group reservations with limitations:
Students can join a group on each course with a minimum of
4 students for each group.
Each group is allowed to book a classroom once per week.
Any member of the group can book on behalf of the group.

• Room evaluation and feedback:
Students and facilities can rate the room (cleanliness, lighting, seating comfort, internet speed ...etc).
report issues within the room (equipment malfunction, no board, lack of ventilation ...etc).

• Real-time room updates:
Displays whether a classroom is occupied or available at any given moment.

Expected Benefits:

• **Time efficiency**: Easily locate available classrooms instead of searching manually.

• **Improved academic efficiency**: Enables faculty members to find classrooms quickly with the desired technological resources and necessary amenities.

• **Better utilization of free classrooms**: Provides students with a suitable study environment and enhances their learning experience.

• **Minimized room conflicts**: Offers accurate information on room availability preventing any overlaps or confusion.

• **Enhanced collaboration:** Allows students to book classrooms for group study sessions or meetings, promoting teamwork and collaborative learning.

2. Customer Statement of Requirements (CSR)

A. *Problem Statement*

As a faculty member or student at the university, locating available classrooms quickly and efficiently poses a significant challenge. The current method of finding vacant classrooms relies on either random inquiries or outdated paper schedules, which consumes valuable time and energy. This lack of an efficient classroom availability system often leads to delays in academic activities and missed opportunities for collaborative learning.

Faculty members frequently need additional classrooms on short notice, whether for holding extra sessions, conducting meetings, or organizing academic events. Without a streamlined system, finding an appropriate room is cumbersome and frustrating. Students also face similar issues when looking for available classrooms for group study sessions or project meetings, especially for larger groups of four or more. The absence of a unified platform to check room availability forces students to rely on word-of-mouth information or manual searches, which are both time-consuming and unreliable.

A solution that provides up-to-date information about classroom availability would significantly improve the efficiency of academic activities. Such a system would enable users to easily find rooms based on their needs, saving time and effort. It would also allow users to express their preferences and provide feedback on room conditions. The ability to book rooms in advance or join a waitlist for fully occupied rooms would help ensure that everyone has a fair chance to access university facilities.

By providing a more organized and transparent way to access available classrooms, this solution would promote better utilization of free spaces, encourage collaborative learning among students, and enhance the overall academic environment at the university.

B. Glossary of Terms

Term	Definition
Classroom Availability	The status of a classroom indicating whether it is free or occupied at a given time
Faculty member	A university professor or lecturer who can book classrooms for lectures or meetings
Student	A user who can search for classroom availability but cannot directly book rooms (except under certain conditions)
Section	Specific class offering of a course, identified by a unique section number
Course Name	The title of an academic course
Booking	The process by which a faculty member or Student reserves a classroom for a specific date and time
Classroom Details	Information about a specific classroom, including capacity, location, technology, and scheduled courses
Issue Reporting	A feature allowing users to report problems with a classroom, such as broken equipment or poor ventilation
Waitlist	A queue system where users can request a room when it is fully booked and get notified if it becomes available
Notifications	Alerts sent to Students regarding classroom changes, bookings, waitlist updates, or maintenance issues.
System Administrator	The system administrator responsible for managing user roles, classroom data, and resolving technical issues

3. System Requirements

A. *Enumerated Functional Requirements*

REQ-x	Description
REQ-01	The system shall allow users to book classrooms. • The system shall allow faculty members to book available classrooms directly. • Students can book a classroom only if they are in a group. • Each student group can only book one classroom per week. • If a classroom is fully booked, students can join a waitlist and receive notifications when it becomes available. • Bookings must be confirmed within 10 minutes, or they will be automatically canceled.
REQ-02	The system shall be integrated with Edugate • User Authentication: • The system shall authenticate users via Edugate Single Sign-On (SSO). • Classroom Schedule Synchronization: • The system shall fetch real-time class schedules, exams, and special reservations from Edugate. • Faculty Assignment Retrieval: • Faculty members shall be able to see their assigned classrooms for each lecture.
REQ-03	The system shall allow user to be able to search for available classroom: • The system shall allow students to search for any section's assigned classroom using the section number and course name, enabling them to quickly search for scheduled rooms without manual inquiries. • The system shall enable Students or Faculty member to search for available classrooms based on building, floor, and seating capacity. The system shall display real-time availability, allowing users to find suitable classrooms efficiently.
REQ-04	The system shall users to Issue Reporting and receiving Notification • The system shall allow students and faculty members to report issues in classrooms, such as broken equipment or ventilation problems, through an online interface. • The system shall provide a form where users can describe the issue, select the affected classroom, and attach optional images. • Upon submission, the system shall generate a unique issue ID and send a confirmation notification to the reporter. • The system shall automatically notify the maintenance team via email or an internal notification system.

	• The system shall allow maintenance staff to update the status of reported issues (e.g., "In Progress," "Resolved"). • The system shall provide a dashboard for tracking reported issues, categorized by status and location.
REQ-05	The system shall allow students to join a group. • displaying available groups based on the student's enrolled courses. • allow the student to select a group. • verify if the group meets the minimum member requirement (4 students). • add the student to the group and notify members once the group meets the required number of members.

B. Enumerated Nonfunctional Requirements

REQ-x	Description
REQ-06	*The system must provide real-time updates on classroom availability with* : • The system shall respond to search queries within 2 seconds. • Classroom availability should be updated in real-time (every 5 seconds). • The system must handle at least 2,000 concurrent users without performance degradation.
REQ-07	*The system shall provide an intuitive user interface with:* • Easy navigation for searching and booking classrooms. • Minimal learning curve for first-time users. • Consistent design across different devices. • Access all features within 3 clicks or less.
REQ-08	*Security (Role-Based Access Control):* • Faculty members shall have exclusive access to book and manage classroom reservations • Students shall only have viewing access to classroom availability and schedules, however, under certain conditions they may be allowed to book classrooms • The system shall prevent unauthorized bookings, ensuring that only eligible users can make reservations
REQ-09	*System Reliability (Availability & Downtime):* • The system shall maintain a 99.9% uptime, ensuring that classroom availability data remains accessible at all times.

	• Scheduled maintenance shall not exceed 43.2 minutes per month to meet uptime requirements. • The system shall implement failover mechanisms and redundancy strategies to minimize unexpected downtime. • In case of system failure, recovery procedures shall restore functionality within 5 minutes for critical services. • System availability shall be monitored continuously, with automated alerts for any downtime incidents.

C. On-Screen Appearance Requirements

Fig1. entry page to accessing the SmartClass system.

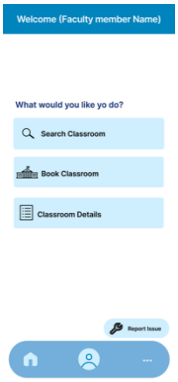


Fig2. Faculty Dashboard

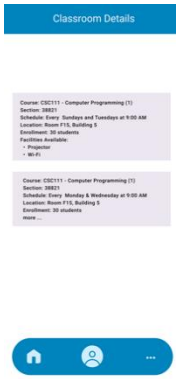


Fig3. Faculty Dashboard (classsroom Details)

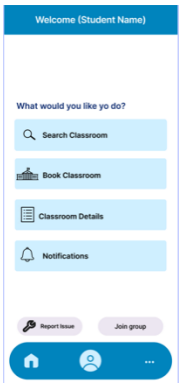


Fig4. Student Dashboard

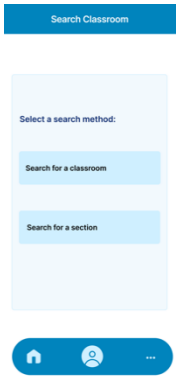


Fig5. Student Dashboard (Search Classroom)

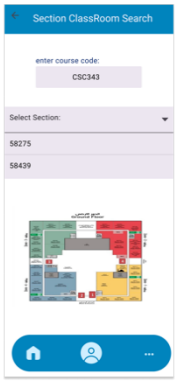


Fig6. Search for classroom by Section

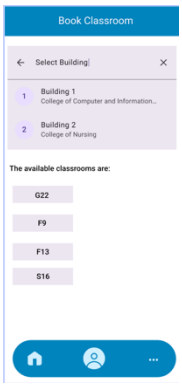


Fig7. Student Dashboard (Classroom details)

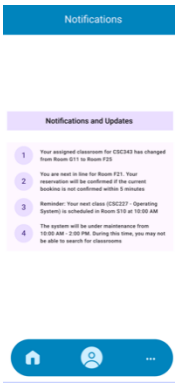


Fig8. Student Dashboard (Notifications)

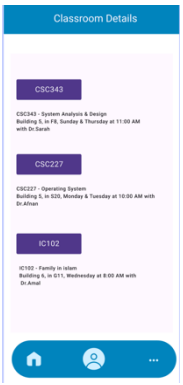


Fig9. User Dashboard (Book Classroom)

Fig10. User Dashboard (Book Classroom) Classroom)

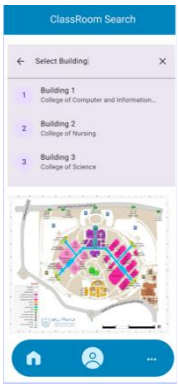


Fig11. User Dashboard (Search Classroom)

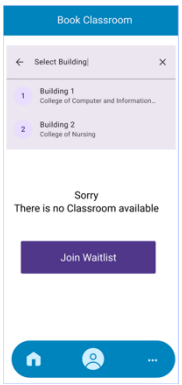


Fig12. User Dashboard (Search

4. Functional Requirements Specification

A. Stakeholders

Faculty Members: Primary users who search for or need to book classrooms for lectures, meetings, or additional sessions.
Students: Primary users who search for available classrooms or need to book classrooms for group study sessions or project meetings.
University Administration: Responsible for managing classroom schedules, system policies, and resolving conflicts.
IT Department: Manages system maintenance, security, and technical support.
Assistant: Participates in reporting issues related to classroom conditions.

B. *Actors and Goals*

Faculty Member (Initiating Actor): The primary goal is to book classrooms for lectures, extra sessions, and meetings efficiently without delays.

Student (Initiating Actor): The primary goal is to find and book available classrooms for group study sessions or project meetings, provided the group meets the minimum member requirements.

University Administration (Participating Actor): Manages classroom schedules, resolves conflicts, and approves exceptional booking requests.

IT System (Participating Actor): Provides real-time data on room availability, booking confirmations, waitlist notifications, and security features.

Assistant (Participating Actor): Assists in reporting classroom issues and ensuring maintenance requests are submitted

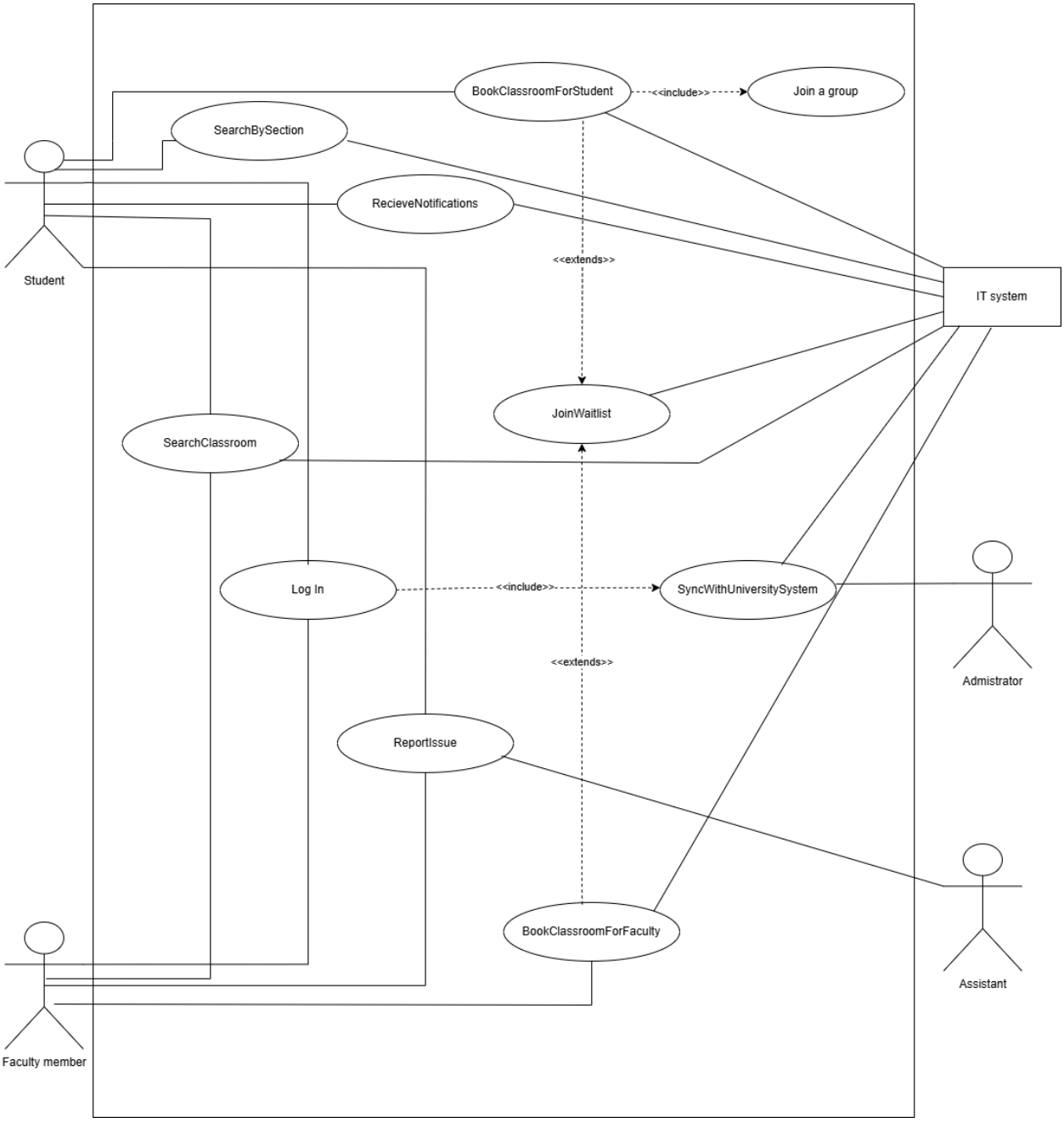
C. Use Cases

I. *Casual Description*

UC-X	Use Case Name	Description	Corresponding REQ ID
UC-1	Log In	Users can log in to the system using their username and password	REQ-02
UC-2	Search for Classroom	Student or faculty member searches for an available classroom using filters such as building, floor, and seating capacity. The system displays real-time classroom availability	REQ-03
UC-3	Search by section (student)	Student can search for classroom by entering section number and course code. The system retrieves and displays the classroom details	REQ-03
UC-4	Book a Classroom for faculty	faculty member searches for an available classroom and submits a booking request	REQ-01
UC-5	Book a Classroom for student	A student group searches for and books a classroom for study sessions or meetings. The system verifies that the group meets the minimum member requirement before allowing the booking	REQ-01, REQ-05

UC-6	Receive Notifications	A student receives a real-time notification when their assigned classroom is changed	REQ-04
UC-7	Report Issues	Students or faculty can report classroom issues (e.g., broken projector, missing chairs, ventilation problems**)** through the system	REQ-04
UC-8	Sync with University system	The system integrates with the university scheduling system to fetch real-time schedules, exams, and reservations**,** ensuring classroom availability remains accurate. If synchronization fails, administrators are notified	REQ-02
UC-9	Join a group	Students can join an available group for their enrolled courses if the group meets the minimum member requirement	REQ-05
UC-10	Join a Waitlist	If a classroom is fully booked, students or faculty members can join a waitlist to be notified when a classroom becomes available.	REQ-01

II. Use Case Diagram



III. *Fully-Dressed Description*

Use case ID	Use case name
UC-1	Log In
Initiating actor:	Student, Faculty Member
Actor's goal:	To authenticate and gain access to the system's features based on their role (student, faculty)
Participating actor:	Administrator, IT System
Trigger	Student/ Faculty Member clicks log in button
Pre-condition:	Post-condition:
• The user must be registered in the university system. • The system must be connected to the authentication database.	• The user is successfully authenticated and redirected to their respective dashboard. • The system grants access based on user roles (Student, Faculty).
Flow of event for success scenario:	• → The user navigates to the Log In page. • ← The system displays fields for Username and Password. • → The user enters their credentials and clicks "Log In". • ← The system validates the credentials with the university database. • ← system retrieves the user's role (Student, Faculty), if the credentials are correct. • → The user successfully logs in and can access system functionalities.
Flow of event for extension (alternative scenario):	If the user forgets their password: • The system provides a "Forgot Password" option. • The user can reset their password via email verification.

Use case ID	Use case name
UC-2	Search for Classroom
Initiating actor:	Student, Faculty Member
Actor's goal:	To find an available classroom based on building, floor, or seating capacity
Participating actor:	IT system
Trigger	The user clicks on Search Classroom button
Pre-condition:	Post-condition:
• user must be logged into the system	• The system displays a map of available classrooms based on the search criteria • The user successfully finds a suitable classroom for their needs
Flow of event for success scenario:	1. → The user selects the Search Classroom option 2. ← The user selects a building from the list 3. → The system fetches classrooms within the selected building 4. ← The user selects a floor, and an interactive map displays the layout 5. → The system highlights occupied and available classrooms based on real-time data 6. ← The user selects a classroom to view details
Flow of event for extension (alternative scenario):	1. If the user does not select a building or floor, the system prompts the user to select at least one search filter 2. If the system is unable to fetch real-time classroom availability, the system provides an option to retry or return to the main menu 3. If there are no available classrooms that match the selected building and floor, The system displays a message (please try again or select different building/floor)

Use case ID	Use case name
UC-5	Book a Classroom for student
Initiating actor:	Student
Actor's goal:	to reserve an available classroom for group study sessions or project meetings.
Participating actor:	IT System
Trigger	The student clicks on Book classroom button
Pre-condition:	Post-condition:
1. Student must be logged into the system. 2. Student must be a member of a group with at least 4 members.	1. The classroom is successfully booked. 2. The student group receives a booking confirmation notification. 3. If no rooms are available, the group is added to the waitlist.
Flow of event for success scenario:	1. → Student searches for available classrooms. 2. ← The system displays real-time classroom availability. 3. → Student selects a classroom and provides booking details. 4. ← The system verifies group eligibility and temporarily reserves the room. 5. → Student confirms the booking. 6. ← The system finalizes the booking and sends a confirmation notification.
Flow of event for extension (alternative scenario):	1. If the group doesn't meet the minimum requirement, the system prevents booking. 2. If the classroom is fully booked, the system prompts the student to join the waitlist. 3. If no rooms are available, the system sends a notification when a room becomes available.

Use case ID	Use case name
UC-7	Report Issues
Initiating actor:	Student, Faculty Member
Actor's goal:	To report issues in classrooms, such as broken equipment or ventilation problems, through an online interface to ensure timely maintenance and resolution.
Participating actor:	Assistant
Trigger	Student/Faculty Member clicks the report issue button
Pre-condition:	Post-condition:
• The user (student or faculty member) must be logged into the system. • The classroom issue reporting feature must be accessible.	1. The reported issue is recorded in the system with a unique issue ID. 2. The maintenance team is notified. 3. The reporter receives a confirmation notification.
Flow of event for success scenario:	1. → The actor logs into the system. 2. → The actor navigates to the "Report Issues" section. 3. → The actor selects the affected classroom. 4. → The actor describes the issue and optionally attaches an image. 5. → The actor submits the issue report. 6. ← The system generates a unique issue ID and stores the report. 7. ← The system sends a confirmation notification to the reporter. 8. ← The system notifies the maintenance team via email or internal notification. 9. → The maintenance team updates the issue status when work begins and upon resolution. 10. → The actor can track the issue status through a dashboard.

Flow of event for extension (alternative scenario):	1. If the actor does not fill in the required fields, the system prompts them to complete the missing information before submission. 2. If the system fails to generate an issue ID due to a technical problem, an error message is displayed, and the actor is prompted to try again later. 3. If the maintenance team does not update the status within a set period, the system sends a follow-up notification to the team.

5. Interaction Diagrams

A. SSD

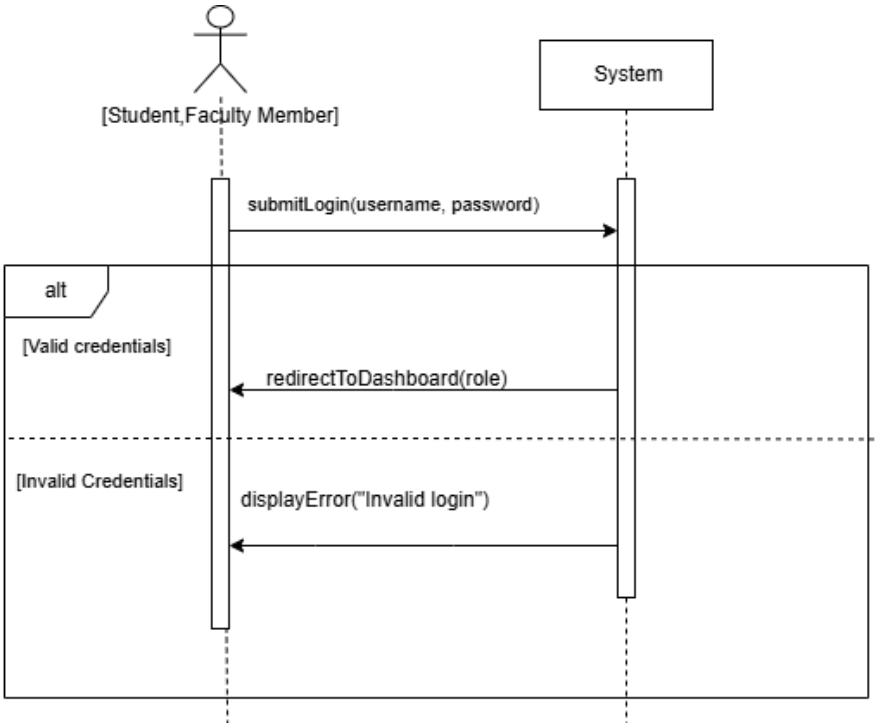


Fig1. Log in

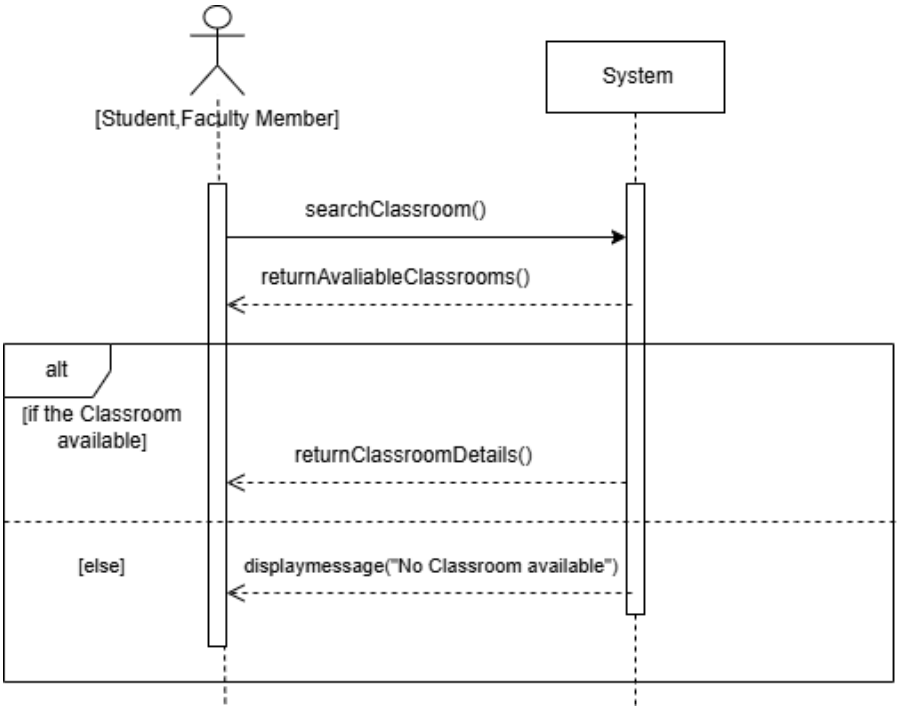


Fig2.Search for classroom

B. SD

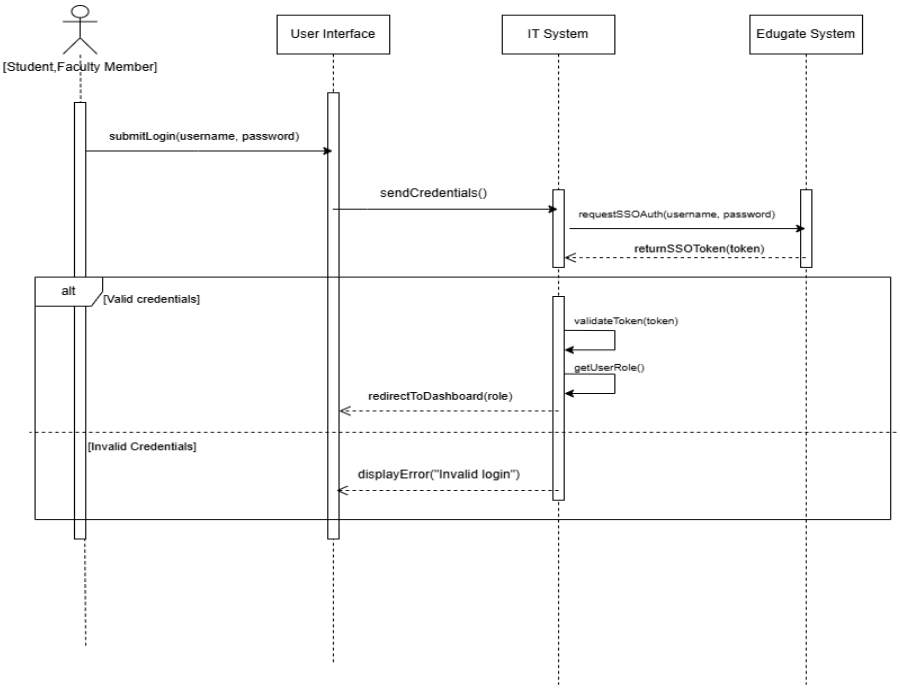


Fig3. UC-1, Log in

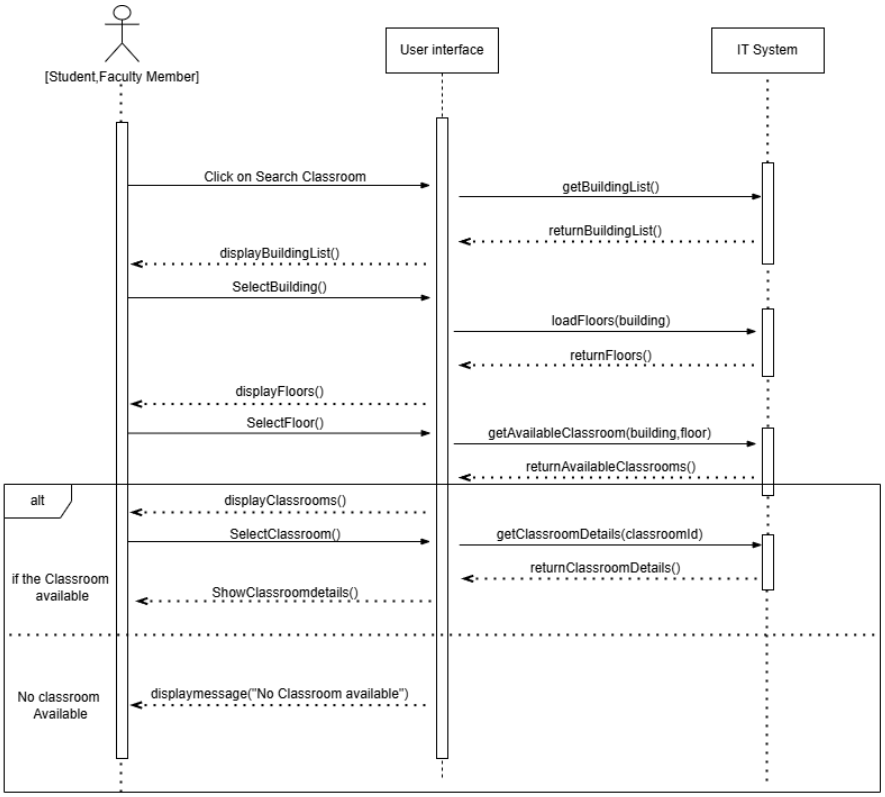


Fig4. UC-2 Search for Classroom

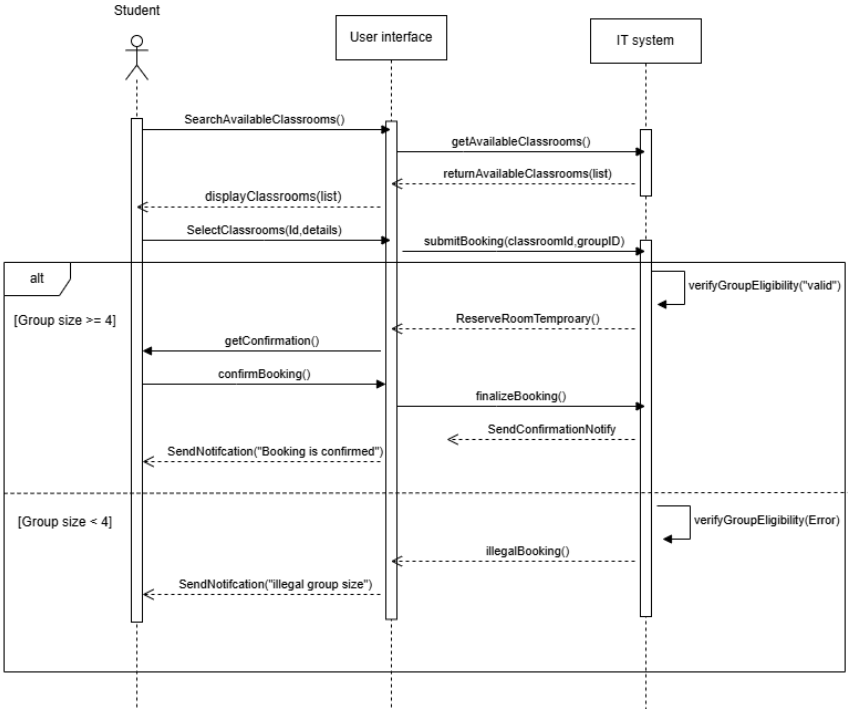


Fig5. UC-5, Book Classroom

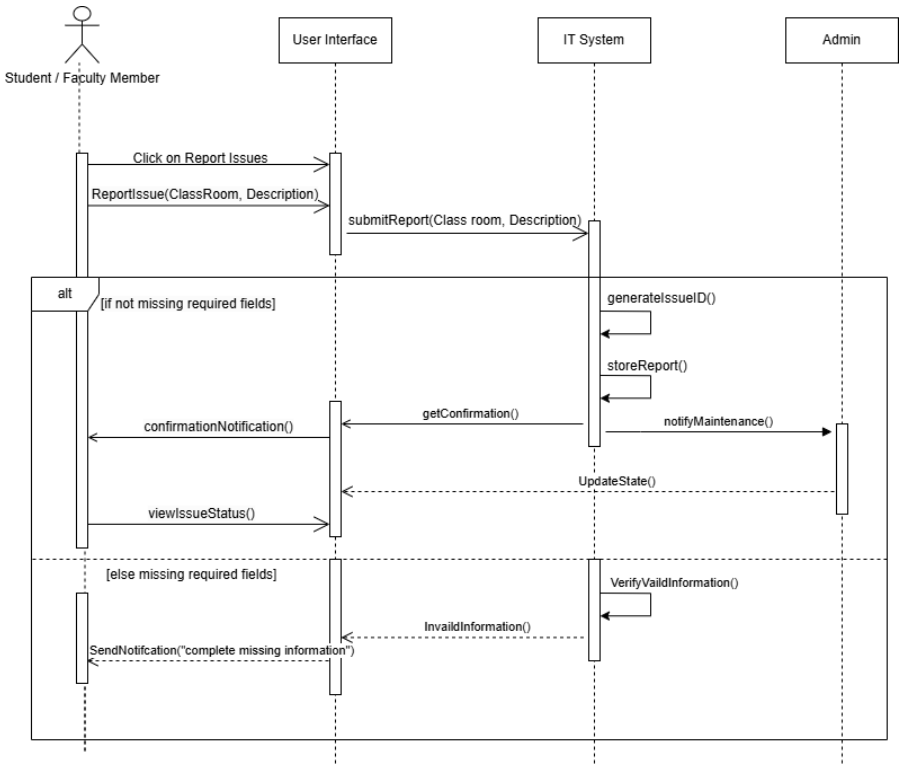
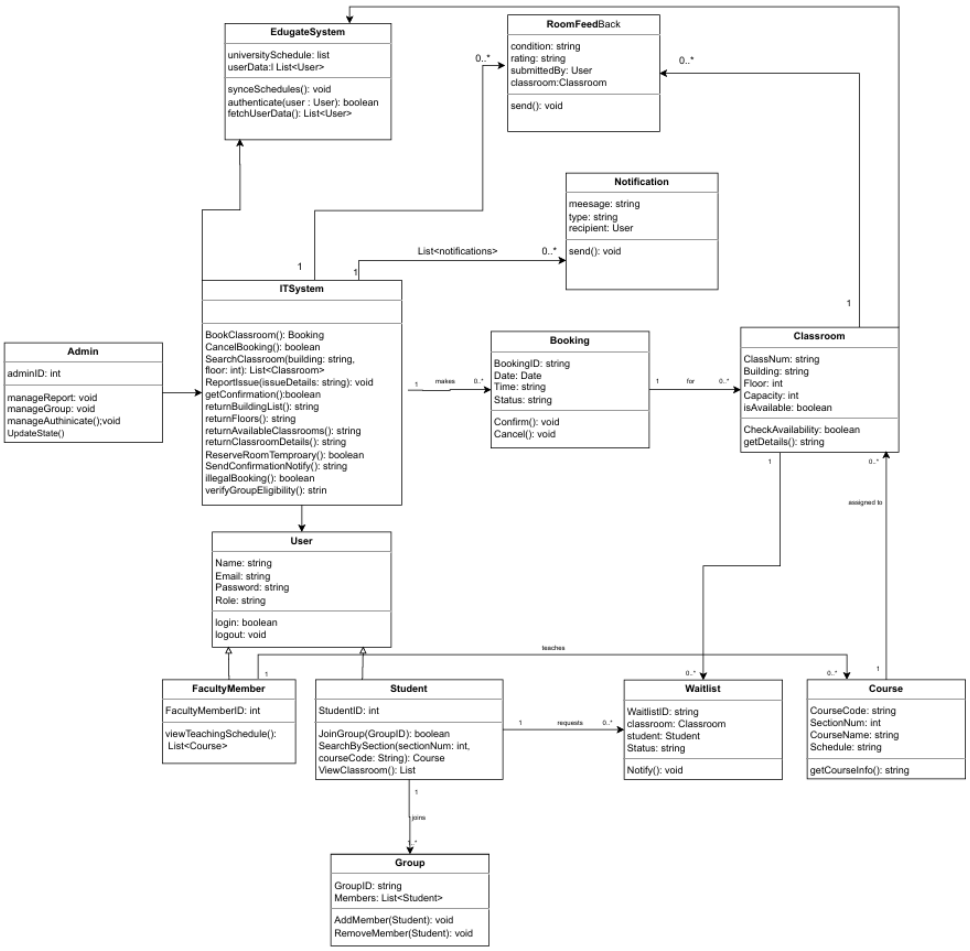


Fig6. UC-7 Report Issue

6. System Architecture and System Design

A. System structural diagram:

I. Class Diagram



II. Class Diagram Description

Class User:

Attributes:

- **Name:** name of the user
- **Email:** user's email address
- **Password:** password for login
- **Role:** either Student or Faculty

Methods:

- **login(): boolean:** logs the user into the system
- **logout(): void:** logs the user out

Class Student:

Attributes:

- **StudentID:** a unique ID for each student

Methods:

- **JoinGroup(GroupID): boolean:** joins the student into a group

- **SearchBySection(sectionNum: int, courseCode: String): Course:** returns the course classroom based on section number and course code
- **ViewClassroom(): List:** returns assigned or available classrooms for the student

Class FacultyMember:

Attributes:

- **FacultyMemberID:** a unique ID for each faculty member

Methods:

- **viewTeachingSchedule(): List<Course>:** returns the list of assigned courses

Class Booking:

Attributes:

- **BookingID:** unique identifier for the booking
- **Date:** date of the booking
- **Time:** time of the booking
- **Status:** current status of booking (pending, confirmed, canceled)

Methods:

- **Confirm(): void:** confirms the booking
- **Cancel(): void:** cancels the booking

Class Classroom:

Attributes:

- **ClassNum:** classroom number
- **Building:** building name
- **Floor:** floor number
- **Capacity:** maximum number of students
- **isAvailable:** boolean to check if classroom is free

Methods:

- **CheckAvailability(): boolean:** returns classroom availability
- **getDetails(): string:** returns a description of the classroom

Class Group:

Attributes:

- **GroupID:** unique identifier for a group
- **Members:** list of student members

Methods:

- **AddMember(Student): void:** adds a student to the group
- **RemoveMember(Student): void:** removes a student from the group

Class Waitlist:

Attributes:

- **WaitlistID:** unique ID of the waitlist entry
- **classroom:** the requested classroom
- **student:** the student who made the request
- **Status:** current status of waitlist (pending, notified, expired)

- **Methods:**
- **Notify(): void:** notifies the student when the classroom becomes available

Class Course:

Attributes:

- **CourseCode:** unique course code
- **SectionNum:** section number of the course
- **CourseName:** course title
- **Schedule:** days/times the course is held

Methods:

- **getCourseInfo(): string:** returns the full description of the course

Class Admin:

Attributes:

• **adminID:** unique ID for the admin

Methods:

• **manageReport(): void:** allows the admin to manage and track submitted issue reports

• **manageGroup(): void:** enables the admin to oversee and organize user groups

• **manageAuth(): void:** controls authentication and access permissions for the system

• **updateSchedule(): void:** updates or modifies the class and course schedule

Class ITSystem:

Methods:

- **BookClassroom(): Booking:** allows the user to book a classroom
- **CancelBooking(): boolean:** cancels the user's booking
- **SearchClassroom(building: string, floor: int): List<Classroom>:** returns available classrooms based on criteria
- **ReportIssue(issueDetails: string): void:** allows user to report issues in the room
- **SendConfirmationNotify(): string:** sends confirmation messages to users
- **SendNotification(): string:** sends general system notifications
- **requestSSOAuth(username, password): void:** sends login request to Edugate for verification
- **validateToken(token): void:** verifies the session token
- **getUserRole(): string:** retrieves the user role after login
- **redirectToDashboard(role): void:** navigates the user to the correct dashboard
- **displayError(): void:** shows an error if login fails
- **generateIssueID(): void:** creates a unique ID for a submitted issue
- **storeReport(): void:** saves the issue report in the system
- **notifyMaintenance(): void:** informs the admin about reported issues
- **UpdateState(): void:** updates the current status of the issue

- **VerifyValidInformation(): boolean:** checks if issue submission has required fields

Class RoomFeedback:

Attributes:

- **condition:** feedback description or note
- **rating:** numerical or textual rating of the classroom
- **student:** reference to the student submitting the feedback
- **classroom:** reference to the classroom the feedback is about

Methods:

- **send(): void:** submits the feedback to the system

Class EdugateSystem:

Attributes:

- **universitySchedule: list:** stores current schedule information
- **userData: List<User>:** stores data of authenticated user

Methods:

- **syncSchedule(): void:** synchronizes university schedule data
- **authenticateUser(u: User): boolean:** authenticates a user attempting to log in
- **fetchUserData(): List<User>:** retrieves a list of user data from the system

Class Notification:

Attributes:

- **message:** content of the notification
- **type:** category of the message (info, alert, error, etc.)
- **recipient:** the user who will receive the message

Methods:

- **send(): void:** delivers the notification to the specified recipient

A. Architectural diagram

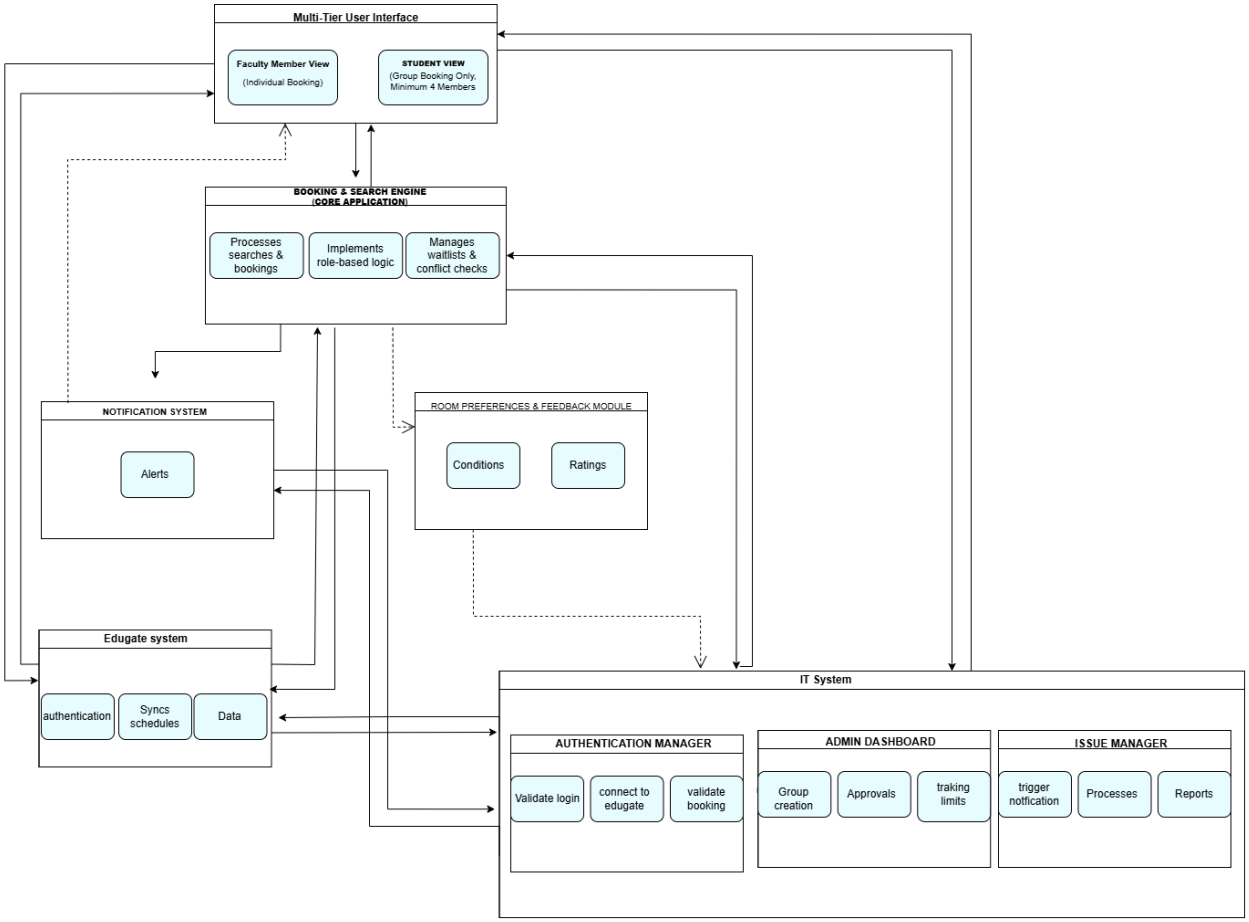
I. *Architectural Style*

Our system is designed using the Client–Server Architectural Style, where the system is divided into two main components:
Client Side: This includes the user interfaces used by students, faculty members, and assistants.
Server Side: Responsible for processing logic like login, validation, classroom booking, issue reporting
The rationale behind this design is as follows:

Clear Separation of Responsibilities: The architecture clearly separates user interaction from internal logic and data processing, which improves maintainability and simplifies the development process.
Seamless Integration with External Systems: The server can communicate directly with third-party systems like Edugate using APIs, without exposing sensitive logic to the client side.
Scalability: The client–server model enables the system to handle a large number of concurrent users efficiently. It also supports horizontal scalability for future enhancements.
Centralized Security and Control: Authentication, authorization, and access control are all managed on the server side, ensuring consistent enforcement of security policies.
Modularity and Maintainability: Features such as login, booking, group management, and issue reporting are structured as independent modules, allowing for easier updates and future expansion.

This architectural style ensures a robust, secure, and maintainable system design suitable for academic environments involving multiple user roles and system integrations.

II. Block diagram



Client Side (Multi-Tier User Interface)

- **Faculty Member View**: Enables individual room booking.
- **Student View**: Allows group bookings (minimum of 4 members).
- Both views interact with the core system via the **Booking & Search Engine**.

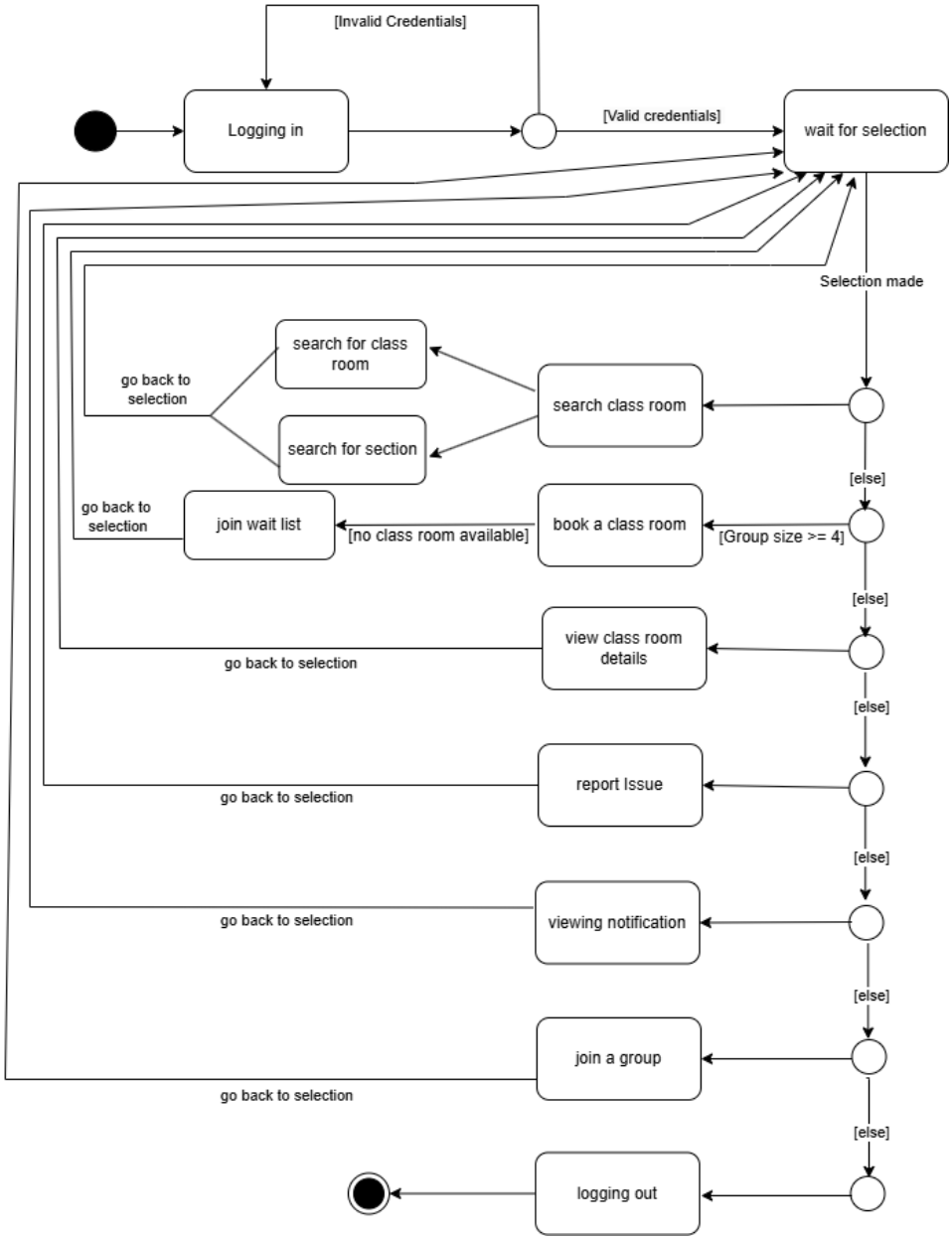
Server Side (IT System & Core Modules)

1. **Booking & Search Engine (Core Application)**
    - **Processes searches & bookings**
    - **Implements role-based logic** (distinguishing between faculty and student behavior)
    - **Manages waitlists & conflict checks**
2. **Notification System**
    - Sends **alerts** to users based on booking events, issues, or updates.
3. **Room Preferences & Feedback Module**
    - Collects and processes **room condition** and **ratings** from users for better room management.
4. **Edugate System Integration**
    - Provides **authentication**, **schedule syncing**, and **data exchange** with the university's academic platform.
5. **Authentication Manager**
    - Validates user login
    - Connects to Edugate for authentication
    - Validates booking requests
6. **Admin Dashboard**
    - Supports **group creation**, **booking approvals**, and **tracking usage limits**
7. **Issue Manager**

- Handles **user-reported problems**, triggers **notifications**, and generates **reports** for administrative follow-up.

B. System behavioral diagram

I. State Diagram



7. Design of Tests

I. Unit Testing

Test Case ID	TC-1
Use case tested	Log in
Assumptions	The user is on the login screen and has valid login credentials (username and password)
Unit to test	Login form, user account system
Steps to be executed	1. Enter valid username and password 2. Click the "Log In" button
Expected result	○ The system validates the credentials ○ The user is redirected to the main dashboard based on their role
Pass/Fail	Login successful and user dashboard shown/ Login failed or error message displayed

Test Case ID	TC-2
Use case tested	Search for Classroom
Assumptions	The user has logged in the system and cicked the "search for classroom" button on the main screen then "search classroom"
Unit to test	classroom availability system
Steps to be executed	1. Click on "Search Classroom" 2. Select a building from the list 3. Select a floor 4. Wait for system to display available/occupied classrooms 5. Click on a classroom to view details
Expected result	○ The system responds to each action with the appropriate data ○ Available classrooms are shown clearly on the map ○ Upon selection, detailed information for the classroom is displayed
Pass/Fail	Classroom search and details displayed correctly/ Search or details not displayed

Test Case ID	TC-3
Use case tested	Search by section (student)
Assumptions	The user has logged in the system and cicked the "search for classroom" button on the main screen then "Search by Section"
Unit to test	course schedule system
Steps to be executed	1. Click on "Search by Section" 2. Enter course code and section number 3. Submit the search request 4. Wait for system to display classroom assigned to that section
Expected result	○ System displays the correct classroom for the entered section ○ Classroom details are retrieved and shown accurately
Pass/Fail	Classroom location shown correctly/ Invalid section input or no result returned

Test Case ID	TC-4
Use case tested	Book a Classroom for faculty member
Assumptions	The user has logged in the system and cicked the "Book Classroom" button on the main screen
Unit to test	Booking system, time slot validation
Steps to be executed	1. Click on “Book Classroom” 2. Select classroom and time 3. Submit booking request
Expected result	○ Classroom is successfully booked ○ Confirmation notification is sent
Pass/Fail	Booking completed/Booking blocked or failed

Test Case ID	TC-5
Use case tested	Book a Classroom for student
Assumptions	Student is logged in and is a member of a group with at least 4 members
Unit to test	Group validation, notification system, Booking system, time slot validation
Steps to be executed	1. Click on “Book Classroom” 2. Select classroom and time 3. Submit booking request 4. Confirm the booking
Expected result	○ Booking is confirmed ○ Confirmation notification is sent ○ Group eligibility is verified
Pass/Fail	Booking completed/Booking blocked or failed

Test Case ID	TC-6
Use case tested	Receive Notifications
Assumptions	The student/faculty member is logged in and has enabled notifications
Unit to test	Notification system, classroom assignment system
Steps to be executed	1. Assign a classroom to a student/faculty member 2. Change the assigned classroom 3. System sends notification to student
Expected result	○ The system detects the change ○ The student receives a real-time notification with updated classroom info

Pass/Fail	Notification received and correct info displayed / Notification not received or incorrect

Test Case ID	TC-7
Use case tested	Report Issues
Assumptions	User is logged in and the issue reporting interface is accessible
Unit to test	Issue reporting form, issue tracking system
Steps to be executed	1. navigate to the “Report Issue” section 2. Select issue type (e.g., broken projector) 3. Submit the report
Expected result	○ The issue is saved in the system ○ Admin or support team is notified or can view the issue
Pass/Fail	ssue successfully submitted and tracked / Submission fails or issue not saved

Test Case ID	TC-8
Use case tested	Sync with University System
Assumptions	The university system provides API or data access; system is configured for scheduled sync
Unit to test	Sync service, error handling, admin alerts
Steps to be executed	1. Trigger the sync process 2. Observe data import or update 3. Simulate a failure (e.g., no response from university API)
Expected result	○ If sync successful: data is updated ○ If sync fails: admin receives notification

Pass/Fail	Data synced correctly or admin notified on failure / Sync silently fails or incorrect update

Test Case ID	TC-9
Use case tested	Join a Group
Assumptions	Student is enrolled in a course with available groups
Unit to test	Group management system
Steps to be executed	1. View available groups for enrolled course 2. Select a group that meets member requirements 3. Click “Join Group”
Expected result	○ Student is successfully added to group ○ Group member count updates
Pass/Fail	Student added to group successfully / Join request denied or error occurs

Test Case ID	TC-10
Use case tested	Join a Waitlist
Assumptions	Classroom is fully booked and waitlist feature is active
Unit to test	Waitlist management, notifications
Steps to be executed	1. Attempt to book a full classroom 2. Select “Join Waitlist” 3. Wait for a slot to open and observe notification
Expected result	○ Student is added to waitlist ○ System notifies user when a slot becomes available

Pass/Fail	Waitlist joined and notification sent / Failed to join waitlist or no notification received

II. Integration Test Cases

Test Case IT-1: Faculty Login and Individual Booking

- **Use Case:** UC-1, UC-4
- **Modules Involved:** Faculty View → Booking & Search Engine → Edugate → Admin Dashboard
- **Steps:**
  1. Faculty logs in via SSO (Edugate).
  2. Searches for available room.
  3. Submits a booking.
- **Expected:**
  - Auth via Edugate is successful
  - Booking is created and recorded
  - Admin sees booking in dashboard

Pass/Fail: Booking flow complete / Any module fails

-

Test Case IT-2: Student Login and Group Booking

- **Use Case:** UC-1, UC-5, UC-9
- **Modules Involved:** Student View → Booking & Search Engine → Group Validation → Edugate
- **Steps:**
  1. Student logs in.
  2. Creates or joins a group (≥4 members).
  3. Books room via group.
- **Expected:**
  - Group size validated
  - Booking approved only if valid
  - Edugate confirms enrollment
- **Pass/Fail:** Group booked / Booking blocked or error shown

---

Test Case IT-3: Search Classroom and Section Mapping

- **Use Case:** UC-2, UC-3
- **Modules Involved:** Student/Faculty View → Booking & Search Engine → Edugate
- **Steps:**
  1. User uses filters or enters section number.
  2. Classroom details retrieved.

- **Expected:**
  - Filters/search respond correctly
  - Data fetched from Edugate
- **Pass/Fail:** Correct room shown / Mismatch or not found

Test Case IT-4: Notifications on Change

- **Use Case:** UC-6
- **Modules Involved:** Booking Engine → Notification System
- **Steps:**
  1. Book a classroom.
  2. Change the assigned room.
  3. Check for real-time notification.
- **Expected:**
  - Change is logged
  - Notification is sent instantly
- **Pass/Fail:** Alert received / No alert or incorrect info

Test Case IT-5: Report Issue and Admin Response

- **Use Case:** UC-7
- **Modules Involved:** Student/Faculty View → Feedback Module → Admin Dashboard
- **Steps:**
  1. User reports issue (e.g., broken chair).
  2. Admin views it in dashboard.
- **Expected:**
  - Issue recorded with metadata
  - Admin sees real-time updates
- **Pass/Fail:** Issue visible / No admin view or tracking

Test Case IT-6: University Sync and Admin Alert

- Use Case: UC-8
- **Modules Involved:** Edugate → Booking Engine → Notification/Admin
- Steps:
  - System syncs with Edugate.
  - Simulate sync failure.
- Expected:
  - Data updated if successful
  - Admin alerted if failure
- **Pass/Fail:** Correct handling / Sync fails silently

Test Case IT-7: Waitlist Flow

- **Use Case:** UC-10
- **Modules Involved:** Booking Engine → Waitlist System → Notification System
- **Steps:**
  1. Try booking a full room.
  2. Join waitlist.
  3. Wait for slot and check notification.
- **Expected:**
  - Waitlist entry confirmed
  - Alert sent when slot opens
- **Pass/Fail:** Waitlist and notify works / Missed or late notification

III. Acceptance test

During the acceptance testing phase of our classroom booking system, we engaged both students and faculty members to evaluate core features such as classroom search, booking, reporting, and group management. Their feedback helped identify improvement areas. Below is a summary of their comments and the actions we plan to take:

- Search for Classroom

Users appreciated the interactive map and filtering by building and floor. However, in cases of slow responses, some were unsure if the system was working.

We plan to add clearer loading indicators and error messages.

- Book Classroom (student)

A few students attempted to book without meeting the group size requirement and received error messages. They suggested adding a visible warning earlier in the process.

We plan to add a real-time group size check and a tooltip alert before booking confirmation.

- Book Classroom (faculty)

Faculty members requested better visibility of their current teaching schedules to avoid time conflicts when booking classrooms.

We are integrating teaching schedules into the booking form view for faculty users.

- Join a group

Students found the group-joining process simple, but some tried to join already full groups without knowing.

We are planning to enhance it by adding live group member counts and availability status to the join screen.

- Report Issues

Users appreciated the ability to report problems with classrooms, but some mistakenly submitted incomplete forms and received errors.

We updated the system to display specific field validation messages and added a preview feature so users can review their report before submission.

8. References

[1] https://draw.io

[2] https://www.figma.com

[3] https://eceweb1.rutgers.edu/~marsic/books/SE/projects/HealthMonitor/2019-g2-report3.pdf

[4] https://eceweb1.rutgers.edu/~marsic/books/SE/projects/HealthMonitor/2013-g7-report3.pdf

[5] https://eceweb1.rutgers.edu/~marsic/books/SE/projects/HealthMonitor/2014-g12-report3.pdf

[6] Behavioral Diagrams.pdf

[7] Object Domain Modelling.pdf