**CLASSROOM ALLOCATION**

**SYSTEM**

**SE MINI PROJECT**

SUBMITTED BY:

SANDHIYA K 220701244

SANJEEV KANTH S 220701250

SHAJINA A 220701257

SHARAN KUMAR D 220701264

MOHAMMED REHAN 220701515

SHARIEF M T

**CLASSROOM ALLOCATION**

**SYSTEM**

**INDEX**

|  |  |  |  |
| --- | --- | --- | --- |
| S.NO | DATE | CONTENT | PAGE NO |
|  |  | OVERVIEW |  |
|  |  | BUSINESS ARCHITECTURE |  |
|  |  | USER STORY AND NFR |  |
|  |  | ARCHITECTURAL PATTERN |  |
|  |  | USECASE DIAGRAM |  |
|  |  | CLASS DIAGRAM |  |
|  |  | SEQUENCE DIAGRAM |  |
|  |  | TEST STRATEGY |  |
|  |  |  |  |

**OVERVIEW**

In educational institutions, the efficient allocation and management of classrooms is critical to ensure that both teaching and learning activities are conducted smoothly. However, many institutions face challenges in effectively managing classroom schedules, handling allocation requests, and resolving conflicts arising from overlapping bookings. This project aims to address these issues by developing a comprehensive Classroom Allocation System that streamlines the process of allocating and managing classrooms for students, professors, and the Head of Department (HOD).

#### Problems Being Addressed

1. Inefficient Classroom Management:
   * Current Situation: The manual or semi-automated methods used for managing classroom allocations are time-consuming and prone to errors.
   * Problem: This often leads to double bookings, underutilization of classroom space, and scheduling conflicts.
2. Lack of Transparency:
   * Current Situation: Students and professors often do not have real-time access to the classroom schedule.
   * Problem: This can result in confusion and miscommunication regarding where and when classes or events are scheduled.
3. Cumbersome Request Process:
   * Current Situation: Requesting classrooms for extra classes, group studies, or events usually involves a lengthy process of filling out forms and awaiting approval.
   * Problem: Delays in the approval process can hinder the planning and execution of academic and extracurricular activities.
4. Conflict Resolution:
   * Current Situation: Resolving conflicts arising from classroom allocations requires significant manual intervention and communication.
   * Problem: This can be inefficient and often leads to last-minute changes that disrupt planned activities.
5. Administrative Overload:
   * Current Situation: The HOD or administrative staff must manually manage and allocate classrooms, which is a resource-intensive task.
   * Problem: This administrative burden can divert attention from other critical responsibilities, reducing overall efficiency.

#### Proposed Solution

The Classroom Allocation System will provide a digital platform to automate and streamline the process of classroom allocation. The system will have distinct functionalities for students, professors, and the HOD, addressing the specific needs and roles of each user type:

1. For Students:
   * Ability to view current classroom allocations.
   * Option to request classrooms for group studies or project work.
   * Receive notifications regarding the status of their requests.
2. For Professors:
   * Access to the schedule of classroom allocations.
   * Capability to request additional classrooms for extra classes or events.
   * Receive updates on the approval status of their requests.
3. For the HOD:
   * Comprehensive view of all classroom allocation requests.
   * Tools to allocate classrooms efficiently based on availability and priority.
   * Functionality to edit and reallocate classrooms as needed to resolve conflicts.
4. General Features:
   * A calendar view to visualize classroom schedules.
   * Search functionality to quickly find specific allocation information.
   * Automated notifications to keep all users informed about the status of their requests.

By addressing the issues of inefficiency, lack of transparency, cumbersome request processes, conflict resolution, and administrative overload, the Classroom Allocation System will enhance the overall management of classroom resources, leading to a more organized and effective academic environment.

**USER BENEFITS:**

1.Efficiency and Time-Saving:

* + Automated Scheduling: Eliminates manual processes, reducing the time required to allocate classrooms.
  + Quick Requests: Streamlined process for students and professors to request classrooms, speeding up approvals.

2.Transparency:

* Real-Time Access: Students, professors, and the HOD can view up-to-date classroom allocations, ensuring everyone is informed of the current schedule.
* Notifications: Immediate notifications for request approvals or rejections keep users updated without delays.

3.User-Friendly Interface:

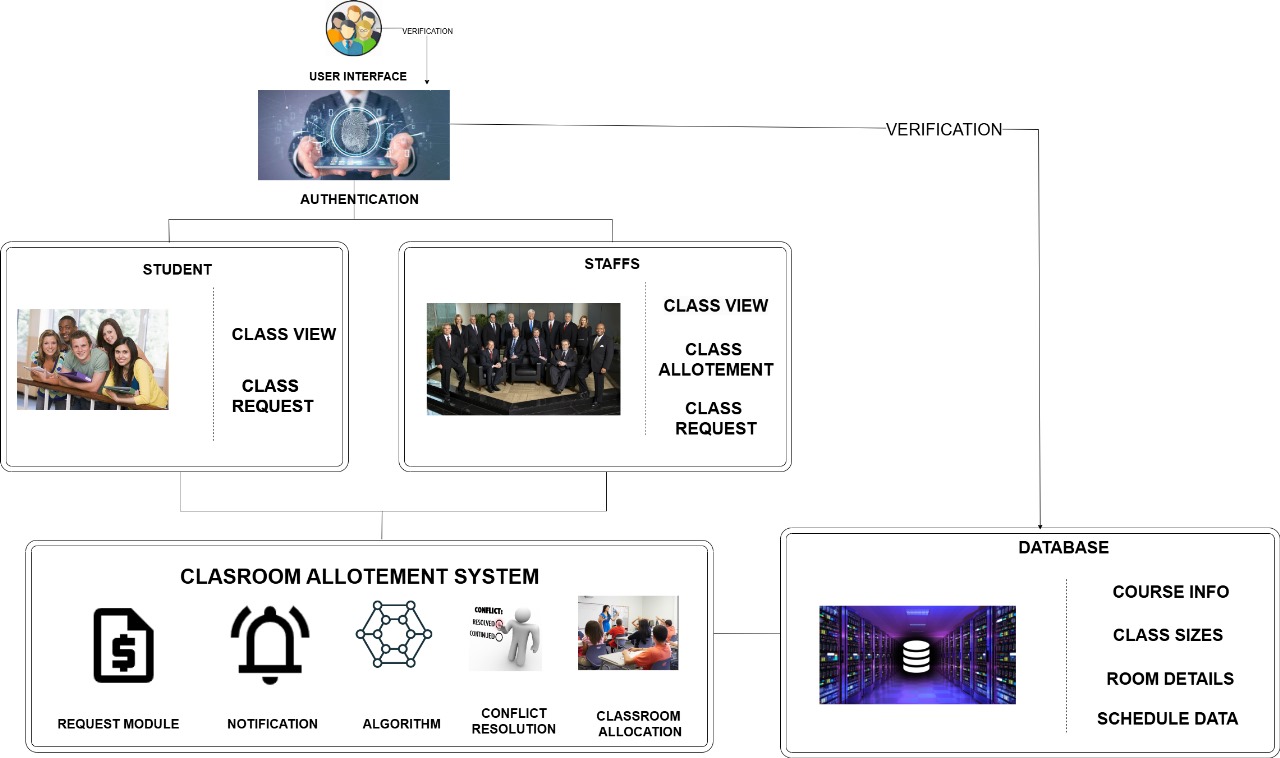
* + Unified Interface: A single, easy-to-use interface for students and professors to interact with the system.
  + Dedicated Dashboard: A specialized HOD dashboard provides comprehensive control over classroom management.

4.Enhanced Communication:

* + Clear Communication Channels: Facilitates clear communication between students, professors, and the HOD regarding classroom allocations and requests.
  + Centralized System: All classroom allocation-related activities are managed in one place, reducing miscommunication and errors.

5.Better Resource Management:

* + Optimized Usage: Ensures that classroom resources are used effectively, avoiding underutilization or overbooking.
  + Data-Driven Decisions: Provides the HOD with data and insights to make informed decisions about classroom allocations.

**BUSINESS ARCHITECTURE**

**BUSINESS ARCHITECTURE EXPLANATION**

This architecture ensures a streamlined and efficient process for managing classroom allocations. The system integrates user requests, uses algorithms to optimize allocation, and resolves conflicts, all while keeping users informed through notifications. By centralizing data in a database, the system supports informed decision-making and effective resource management.

#### Users

* Types of Users:
  + Students:
    - Class View: Allows students to view the current classroom allocations.
    - Class Request: Enables students to request classrooms for their activities or needs.
  + Staff:
    - Class View: Staff members can also view the current classroom allocations.
    - Class Allotment: Staff, particularly the HOD, can allocate classrooms based on requests and availability.
    - Class Request: Staff can request classrooms for their sessions or activities.

#### Classroom Allocation System

* Core Functionalities:
  + Request Module: Manages the requests for classrooms from both students and staff.
  + Notification: Sends notifications to users about the status of their requests and any changes in classroom allocations.
  + Algorithm: Utilizes algorithms to efficiently allocate classrooms based on various criteria such as availability, class size, and priority.
  + Conflict Resolution: Handles conflicts in classroom allocations, such as double bookings, by suggesting alternative options or re-allocating rooms.
  + Classroom Allocation: The process of assigning available classrooms to different requests.

#### Database

* Components:
  + Course Info: Contains details about the courses being taught, which helps in classroom allocation based on the needs of each course.
  + Class Sizes: Information on the number of students in each class, ensuring that classrooms are allocated according to capacity.
  + Room Details: Data about the available rooms, including their sizes, equipment, and current status.
  + Schedule Data: Information on the schedules of classes and other events to avoid conflicts in room allocation.

**USER STORY**

### Student User Stories

1. As a student, I want to view the current classroom allocation, so that I can know where my classes are scheduled.
   * Acceptance Criteria:
     + Student can log in to the system.
     + Student can navigate to a section where all allocated classrooms are listed.
     + Student can see the schedule for each allocated classroom.
2. As a student, I want to request a classroom for group study or project work, so that I can have a space to collaborate with my peers.
   * Acceptance Criteria:
     + Student can log in to the system.
     + Student can navigate to a request form for classroom allocation.
     + Student can fill out and submit the form with details such as date, time, and purpose of use.

### Professor User Stories

1. As a professor, I want to view the classroom allocations, so that I can know where my classes and others are scheduled.
   * Acceptance Criteria:
     + Professor can log in to the system.
     + Professor can navigate to a section where all allocated classrooms are listed.
     + Professor can see the schedule for each allocated classroom.
2. As a professor, I want to request a classroom for an extra class or event, so that I can conduct sessions outside regular class timings.
   * Acceptance Criteria:
     + Professor can log in to the system.
     + Professor can navigate to a request form for classroom allocation.
     + Professor can fill out and submit the form with details such as date, time, and purpose of use.

### HOD User Stories

1. As the HOD, I want to view all classroom allocation requests from students and professors, so that I can manage and approve them.
   * Acceptance Criteria:
     + HOD can log in to the system.
     + HOD can navigate to a section listing all pending requests.
     + HOD can view details of each request including requester, date, time, and purpose.
2. As the HOD, I want to allocate classrooms based on the requests, so that classroom resources are efficiently managed.
   * Acceptance Criteria:
     + HOD can log in to the system.
     + HOD can view available classrooms and time slots.
     + HOD can allocate classrooms to specific requests and update the system.
3. As the HOD, I want to edit or reallocate classrooms if conflicts or changes arise, so that the schedule remains accurate and up-to-date.
   * Acceptance Criteria:
     + HOD can log in to the system.
     + HOD can view the current allocations.
     + HOD can make changes to allocations and notify relevant parties of changes.

### General User Stories

1. As any user, I want to receive notifications for the status of my classroom request, so that I know if my request has been approved or denied.
   * Acceptance Criteria:
     + User can receive email or system notifications.
     + Notification includes details of the request status (approved, denied, or pending).
2. As any user, I want to view a calendar or timetable of all classroom allocations, so that I can see the overall schedule.
   * Acceptance Criteria:
     + User can log in to the system.
     + User can access a calendar view displaying all classroom allocations.
     + Calendar view can be filtered by date, classroom, or requester.
3. As any user, I want to search for classroom allocations based on date, time, or purpose, so that I can quickly find specific information.
   * Acceptance Criteria:
     + User can log in to the system.
     + User can use a search function to filter classroom allocations.
     + Search results display relevant allocations based on the entered criteria.

**NON FUNCTIONAL REQUIREMENTS:**

**Performance Requirements:**

The system should be able to handle more number of users (>8000)concurrent users without performance degradation.   
 Classroom allocation processing time should not exceed 2 seconds for any operation.  
   
**S**

The system must be scalable to accommodate a growing number of users, classrooms, and allocation requests.  
 It should support horizontal scaling by adding more servers to handle increased load.

**Reliability:**

The system should have an uptime of 99.9%, ensuring minimal downtime.  
 Data integrity must be maintained at all times, preventing any loss of allocation data.

**Availability:**

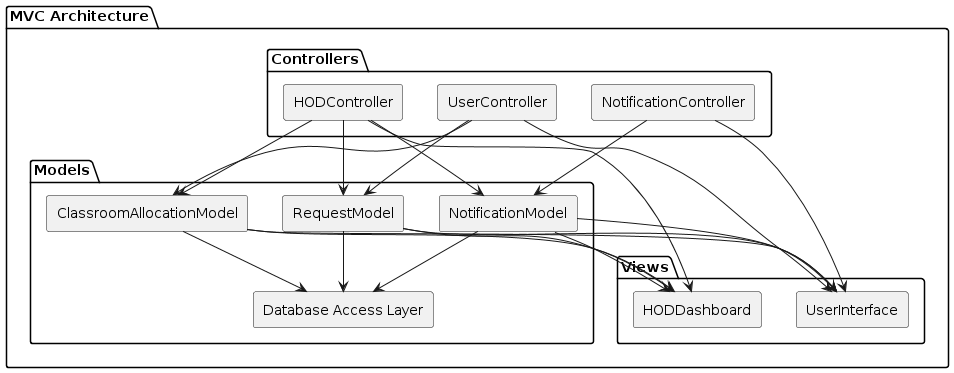
The system should be accessible 24/7, with scheduled maintenance windows communicated at least 48 hours in advance.  
 A failover mechanism should be in place to ensure continuous availability during server failures.

**Security:**

User authentication and authorization mechanisms should be robust, ensuring that only authorized personnel can access the allocation functionalities.

**Maintainability:**  
 The system codebase should be modular, allowing for easy updates and maintenance.  
 Comprehensive documentation should be provided for both users and developers.

**ARCHITECTURAL PATTERN**

**MVC ARCHITECTURE**

**EXPLANATION:**

Controllers Package:

* + UserController handles interactions for students and professors.
  + HODController handles interactions for the HOD.
  + NotificationController manages notifications.
* Models Package:
  + ClassroomAllocationModel, RequestModel, NotificationModel, and Database Access Layer manage data and business logic.
* Views Package:
  + UserInterface serves as the interface for students and professors.
  + HODDashboard serves as the interface for the HOD.

### Relationships:

* Controllers to Views:
  + UserController renders the UserInterface.
  + HODController renders the HODDashboard.
  + NotificationController renders the UserInterface.
* Controllers to Models:
  + Each controller uses the appropriate models to fetch and manipulate data.
* Models to Database Access Layer:
  + Each model interacts with the DAL for database operations.
* Models to Views:
  + Each model updates both the UserInterface and HODDashboard when there are changes in data.

**DESIGN PRINCIPLES**

**Design Principles:**

**Single Responsibility Principle (SRP):**

Each module, class, or function is designed to have a single responsibility, promoting clarity, maintainability, and reusability.

**Open/Closed Principle (OCP):**

Modules are open for extension but closed for modification, allowing for the addition of new features or functionalities without altering existing code.

**Dependency Inversion Principle (DIP):**

High-level modules depend on abstractions, not concrete implementations, facilitating loose coupling and enabling easier integration of components.

**Don't Repeat Yourself (DRY):**

Common functionalities are encapsulated into reusable components, reducing redundancy and ensuring consistency t hroughout the system.

**Reasons for Using Design Principles Modularity:**

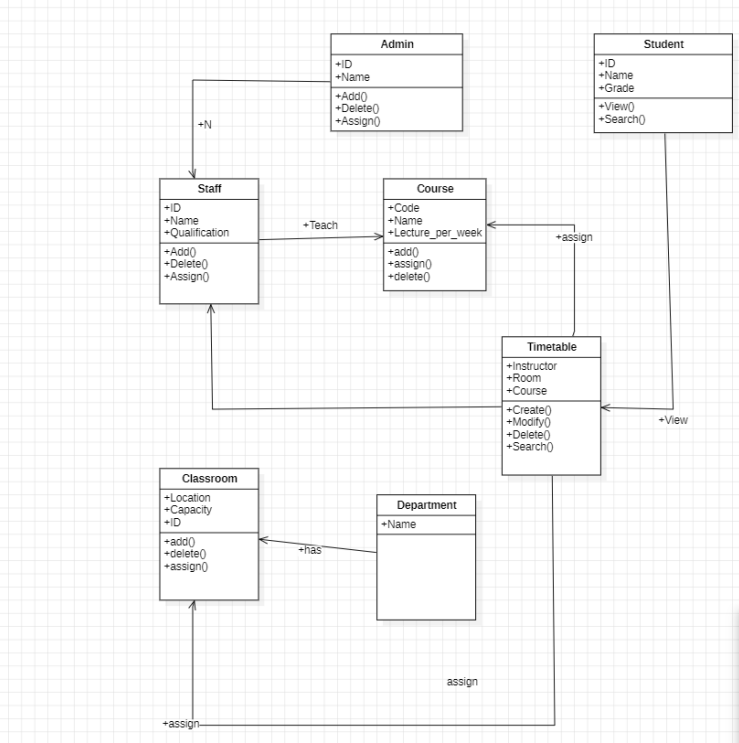
Design principles help in breaking down the system into smaller, manageable components, making it easier to develop, test, and maintain.

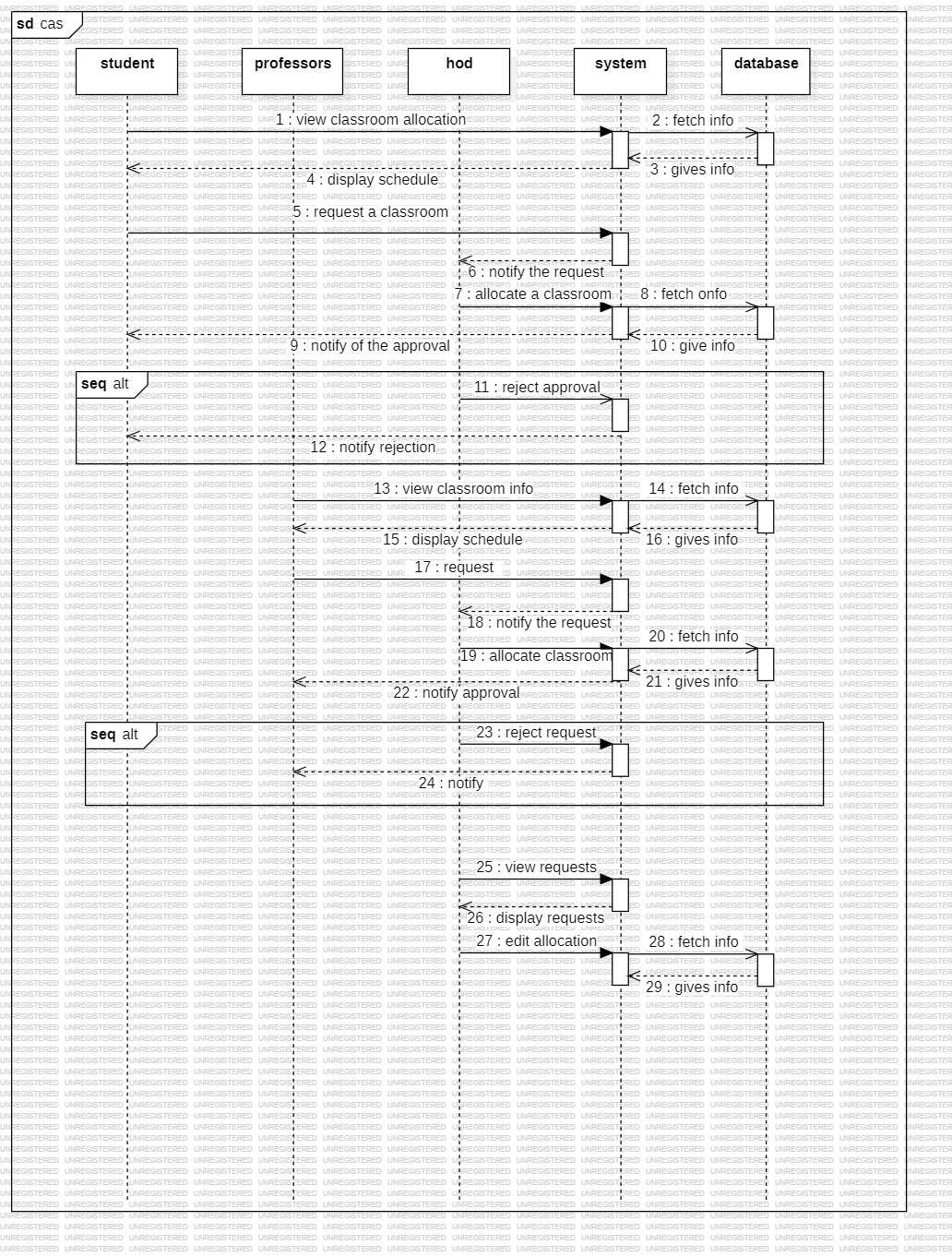
**Flexibility:**

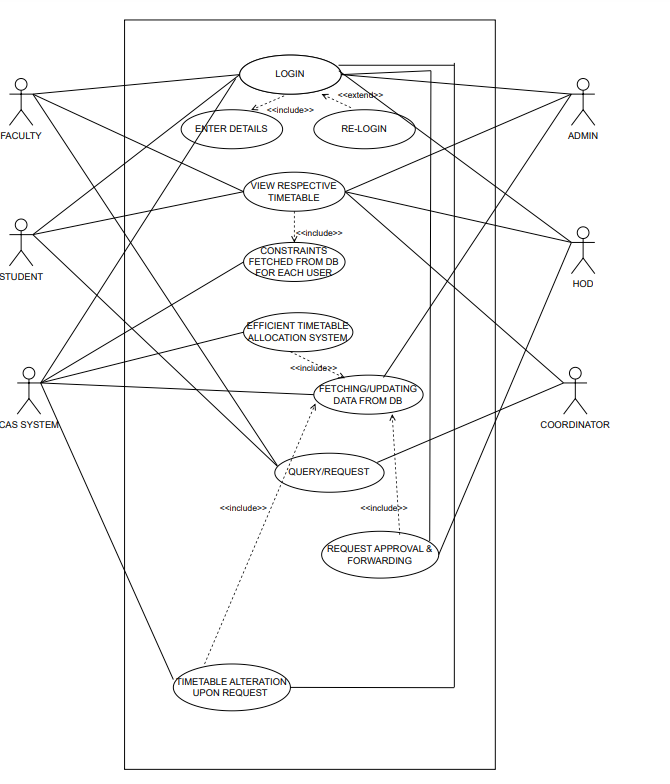
By adhering to design principles, the system becomes more adaptable to changes in requirements, technology, and business rules.

**Readability and Maintainability**:

Following design principles enhances code readability and maintainability, making it easier for developers to understand, modify, and extend the system over time.

**CLASS DIAGRAM**

**SEQUENCE DIAGRAM**

**USECASE DIAGRAM**

**TEST STARTEGY**

To ensure the Classroom Allocation System is robust, reliable, and meets the requirements of all user roles (students, professors, and HOD), a comprehensive testing strategy is essential. This strategy will cover various levels and types of testing to ensure all functionalities are thoroughly verified.

### Testing Levels and Types

#### 1. Unit Testing

* Objective: Verify the functionality of individual components in isolation.
* Scope:
  + Controllers: Test methods in UserController, HODController, NotificationController.
  + Models: Validate logic in ClassroomAllocationModel, RequestModel, NotificationModel.
  + Database Access Layer: Ensure correct implementation of CRUD operations.

#### 2. Integration Testing

* Objective: Ensure that components interact correctly.
* Scope:
  + Controllers and Models: Verify interactions between controllers and models.
  + Models and Database: Check data flow between models and the database access layer.
  + Notification System: Test integration with the notification system.

#### 3. Functional Testing

* Objective: Validate that the system performs its intended functions based on user requirements.
* Scope:
  + User Authentication and Authorization: Ensure secure login and role-based access.
  + Classroom Requests: Test functionalities for requesting classrooms by students and professors.
  + Classroom Allocation: Verify the HOD’s ability to allocate classrooms and resolve conflicts.
  + View Allocations: Ensure students and professors can view classroom allocations.
  + Notifications: Confirm notifications are sent and received correctly.

#### 4. System Testing

* Objective: Validate the complete and integrated system against the requirements.
* Scope:
  + End-to-End Scenarios: Test workflows from user login to classroom allocation and notifications.
  + Load Testing: Assess performance under high user load.
  + Stress Testing: Test system behavior under extreme conditions.

#### 5. User Acceptance Testing (UAT)

* Objective: Ensure the system meets end-user requirements and is ready for deployment.
* Scope:
  + Students: Test viewing and requesting classrooms.
  + Professors: Test viewing and requesting classrooms.
  + HOD: Test viewing requests, allocating classrooms, and resolving conflicts.
  + Notifications: Ensure all users receive appropriate notifications.

#### 6. Regression Testing

* Objective: Ensure that new changes do not negatively impact existing functionality.
* Scope:
  + Retesting: Execute previously passed tests to confirm no new defects.
  + Automated Regression Suite: Develop automated tests for critical functionality to speed up regression testing.

#### 7. Performance Testing

* Objective: Validate the system’s performance characteristics.
* Scope:
  + Response Time: Measure time taken for operations like viewing allocations and submitting requests.
  + Scalability: Ensure the system can handle increasing user and data loads without performance degradation.

#### 8. Security Testing

* Objective: Ensure the system is secure against vulnerabilities.
* Scope:
  + Authentication: Test for secure login mechanisms.
  + Authorization: Verify role-based access control.
  + Data Protection: Ensure secure storage and transmission of sensitive data.

### Test Plan

1. Test Planning and Preparation:
   * Identify test cases based on user stories and system requirements.
   * Set up test environments that mimic the production environment.
2. Test Case Development:
   * Write detailed test cases for each scenario.
   * Develop automated tests for regression and performance testing.
3. Test Execution:
   * Execute unit tests followed by integration, functional, system, UAT, and regression tests.
   * Log and track defects, and re-test after fixes.
4. Test Reporting:
   * Generate reports after each test phase.
   * Summarize test coverage, defects found, and overall system readiness.
5. Continuous Testing:
   * Incorporate continuous integration/continuous deployment (CI/CD) for automated testing.
   * Regularly update tests based on new features and changes.

### Tools and Technologies

* Unit Testing: JUnit, NUnit, or equivalent.
* Integration Testing: Postman, SoapUI, or custom scripts.
* Functional Testing: Selenium, TestComplete, or similar.
* Performance Testing: JMeter, LoadRunner.
* Security Testing: OWASP ZAP, Burp Suite.
* Continuous Integration: Jenkins, Travis CI.

**TEST CASES**

### Test Case 1: Student Views Current Classroom Allocation

Title: Student views current classroom allocation.

Preconditions:

* Student is logged into the system.
* Classrooms have been allocated and the schedule is available in the system.

Test Steps:

1. Student navigates to the classroom allocation section.
2. Student requests to view the current classroom allocation.

Expected Result:

* The system displays the current classroom schedule including the details of each allocated classroom (classroom number, date, time, and purpose).

### Test Case 2: Student Requests a Classroom

Title: Student requests a classroom.

Preconditions:

* Student is logged into the system.

Test Steps:

1. Student navigates to the classroom request form.
2. Student fills in the request form with details such as date, time, and purpose.
3. Student submits the request.

Expected Result:

* The system sends the request to the HOD.
* The student receives a notification that the request has been submitted successfully.
* The HOD receives a notification of the new request.

### Test Case 3: HOD Allocates Classrooms Based on Requests

Title: HOD allocates classrooms based on requests.

Preconditions:

* HOD is logged into the system.
* There are pending classroom requests from students and professors.

Test Steps:

1. HOD navigates to the pending requests section.
2. HOD reviews the details of the requests.
3. HOD selects an appropriate classroom for each request and allocates it.
4. HOD confirms the allocation.

Expected Result:

* The system updates the classroom schedule with the new allocations.
* The requesters (students or professors) receive notifications about the approval and details of the allocated classroom.