

SRI SIVASUBRAMANIYA NADAR COLLEGE OF ENGINEERING

(AN AUTONOMOUS INSTITUTION,
AFFILIATED TO ANNA UNIVERSITY)

Rajiv Gandhi Salai (OMR), Kalavakkam - 603 110.

LABORATORY RECORD

NAME : SHREYA SRIRAM
Reg. No. : 195001106
Dept. : CSE Sem. : VI Sec. : B



SRI SIVASUBRAMANIYA NADAR
COLLEGE OF ENGINEERING, CHENNAI
(AN AUTONOMOUS INSTITUTION, AFFILIATED TO ANNA UNIVERSITY)

BONAFIDE CERTIFICATE

Certified that this is the bonafide record of the practical work done in the

..... MINI PROJECT Laboratory by

Name SHREYA SRIRAM


Register Number 195001106


Semester VI

Branch CSE

Sri Sivasubramaniya Nadar College of Engineering, Kalavakkam.

During the Academic year 2021-2022


Faculty


Head of the Department

Submitted for the End Semester Practical Examination held at SSNCE
on 22.6.22


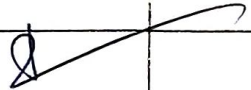

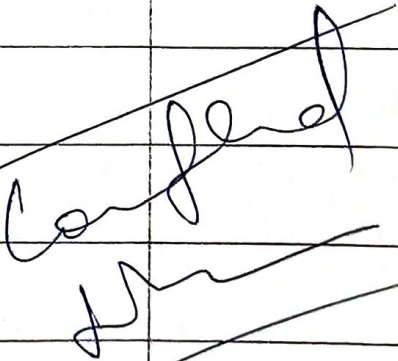
Internal Examiner

External Examiner

INDEX

Name : SHREYA SRIRAM Reg. No. 195001106

Sem : VI Sec : B

Ex. No.	Date of Expt.	Title of the Experiment	Page No.	Signature of the Faculty	Remarks
1.	07/03/22	Problem Statement	1		
2.	14/03/22	SRS Document	3		
3.	21/03/22	Use Case Diagrams	15		
4.	28/3/22	Domain Model and Class Diagrams	24		
5.	4/4/22	Interaction Diagram	30		
6.	11/4/22	State Machine and Activity Diagram	33		
7.	18/4/22	Implementation Demo	35		
8.	23/5/22	Test Cases and Test plans	47		
9.	30/5/22	Applying design pattern	49		
10.	6/6/22	Testing and refactorin	51		

RESOURCE RESERVATION SYSTEM

TEAM MEMBERS:

Shreya Sriram - 195001106
Srivarsha Elangovan - 195001109
Swetha R - 195001114
Tushar Nair - 195001116

DATE: 07.03.2022

ASSIGNMENT 1
PROBLEM STATEMENT

INTRODUCTION:

The classrooms in our department, in addition to being a venue for conducting regular classes, can also serve as examination halls and spaces for club meetings.

In order to facilitate the hassle-free allotment of classrooms, seminar halls, labs, we need to build a user-friendly classroom resources reservation system.

This system can be extended to all departments, schools and workspaces with necessary modifications.

FEATURES:

1. Priority based allotment of classrooms and labs through dynamic allocation.
2. Allotment of classes for examinations.
3. Allotment of seminar halls for events/guest lectures/club meetings, etc.
4. Accommodate changes to the allotment based on class strength.
5. Intimation of changes in the reservation to the concerned faculty and class representative via mail.
6. Reporting issues pertaining to the classroom.
7. Allotment of labs for special purposes, eg: placement tests, student project requirements depending on software requirements.

END USERS:

1) ADMIN:

1. At the beginning of the semester, create initial allotment and time tables for all labs and classrooms.
2. Approve or deny all requests on the basis of priority (priority is determined by purpose of reservation)
3. Only user allowed access before the start of the semester.
4. Incorporates permanent changes to the timetables and allotments. Eg: accommodate a new batch of students to the existing system.
5. View an activity log.

2) PROFESSORS:

1. View their timetables and the ones of the classes handled by them.
2. Reserve classrooms for extra classes by stating the purpose and class strength.
3. Cancellation/extension of class hours.
4. Exchange of classes with another professor.

5. Request to hand over responsibilities to another faculty permanently, subject to admin approval.

3) STUDENTS (GENERAL FUNCTIONS)

1. View timetable for the day and their respective classroom allotment.

4) STUDENTS (REPRESENTATIVES)

1. Report issues pertaining to the classroom.
2. View status of the professors handling their classes.
3. Request professors free during the hour to handle a canceled class.

5) STUDENTS (CLUB HEADS)

1. Reserve free classes for club meetings stating the reason, to be approved by admin.

DATE: 14.03.2022

SOFTWARE REQUIREMENTS SPECIFICATION

1. Introduction

The classrooms in our department, in addition to being a venue for conducting regular classes, can also serve as examination halls and spaces for club meetings.

We have decided to investigate the use of a classroom allocation system System to facilitate hassle free, user friendly and efficient allotment of classroom resources.

This system would be used by members who may be students or professors of that University to check the availability of the classrooms, seminar halls, examination centers and labs. The purpose of this document is to analyze and elaborate on the high-level needs and features of the RRS. It focuses on the capabilities and facilities provided by an RRS. The details of what all are the needs of the RRS and if it fulfills these needs are detailed in the use-case and supplementary specifications.

This system can be extended to all departments, schools and workspaces with necessary modifications.

1.1 Purpose

The purpose of the Software Requirements Specification (SRS) document is to describe the external behavior of the RRS. Requirements Specification defines and describes the operations, interfaces, performance, and quality assurance requirements of the RRS. The document also describes the nonfunctional requirements such as the user interfaces. It also describes the design constraints that are to be considered when the system is to be designed, and other factors necessary to provide a complete and comprehensive description of the requirements for the software. The Software Requirements Specification (SRS) captures the complete software requirements for the system, or a portion of the system. Requirements described in this document are derived from the Vision Document prepared for the SRS.

1.2 Scope

The Software Requirements Specification captures all the requirements in a single document. The RRS that is to be developed provides the members of the department with classroom availability, online blocking of classes and many other facilities. The RRS is supposed to have the following features.

1. The system has priority based allotment of classrooms and labs through dynamic allocation.
2. The system provides allotment of classes for examinations.
3. The system allows allotment of seminar halls for events/guest lectures/club meetings, etc.

4. The product also allows accommodation of changes to the allotment based on class strength.
5. The system allows for intimation of changes in the reservation to the concerned faculty and class representative via mail.
6. The system provides options for reporting issues pertaining to the classroom.
7. The system allows for allotment of labs for special purposes, eg: placement tests, student project requirements depending on software requirements.

The features that are described in this document are used in the future phases of the software development cycle. The features described here meet the needs of all the users. The success criteria for the system is based on the level up to which the features described in this document are implemented in the system.

1.3 Definitions, Acronyms and Abbreviations

RRS – Resource reservation system

PROF – Professor

REP – Representative

SRS – software requirement system

ADMIN – Administrator

SSNCE – SSN COLLEGE OF ENGINEERING

1.4 References

Online Library System Software Requirements Specification – version 1.3

1.5 Overview

The SRS will provide a detailed description of the RRS. This document will provide the outline of the requirements, overview of the characteristics and constraints of the system.

1.5.1 Section 2: This section of the SRS will provide the general factors that affect the product and its requirements. It provides the background for those requirements. The items such as product perspective, product function, user characteristics, constraints, assumptions and dependencies and requirements subsets are described in this section.

1.5.2 Section 3: This section of SRS contains all the software requirements mentioned in section 2 in detail sufficient enough to enable designers to design the system to satisfy the requirements and testers to test if the system satisfies those requirements

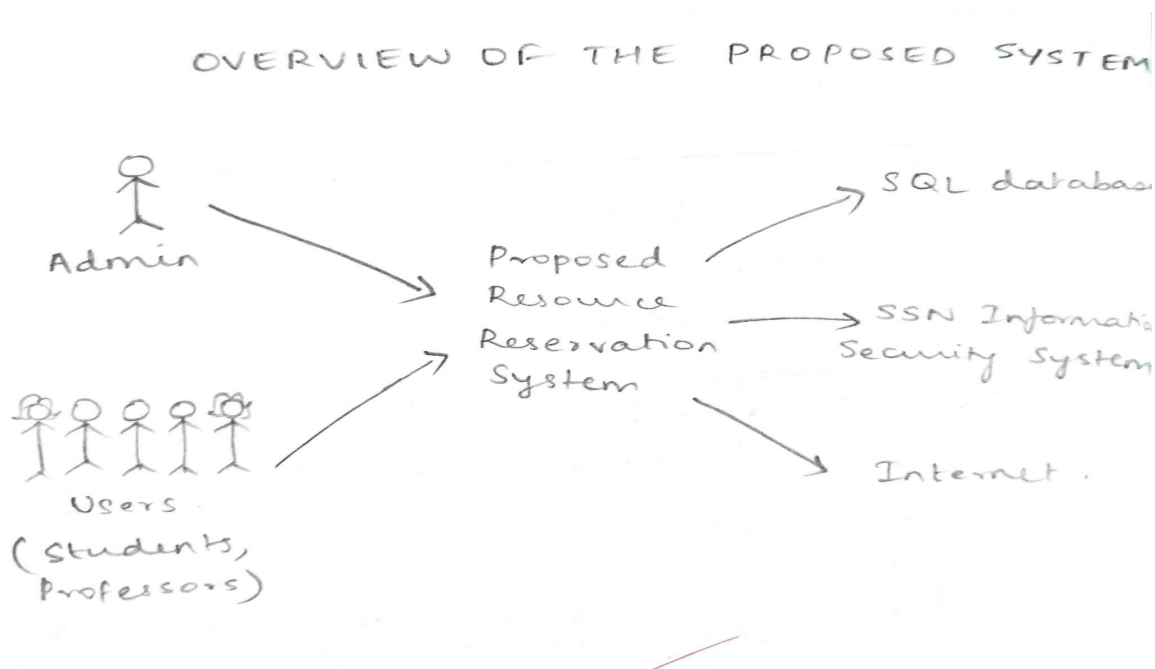
2. Overall Description

2.1 Product Perspective

The Resource Reservation System is an application to be used by the department to improve the efficiency of classroom allocation. The RRS to be developed benefits greatly the students and faculty members of the Computer Science Department, SSNCE. The system provides the daily timetable and current allotment scheme and provides options for modifications as and when needed. The system also includes a mail communication feature such that the members (students and the professors) are updated about the changes immediately.

The complete overview of the system is as shown in the overview diagram below: The product to be developed has interactions with the users: Admin, Members who are the students and professors of the department.

The product has to interact with other systems like: SQL Database, Internet.



2.2 Product Functions

The Resource Reservation System provides information about the current classroom allotment in the department and the timetable information. The Product functions are more or less the same as described in the product perspective. The functions of the system include the system providing different types of services based on the type of users [Admin/Member].

1. Members should be provided with updated information about the allotment scheme.
2. Provision for members to request classrooms, by stating the purpose and seat requirements.

3. Provision for students to view the timetable for the day and for professors to view those of classes handled by them.
4. Faculty members can request to hand over responsibilities to another professor, to be approved by the Admin.
5. The Admin is provided with permissions to make changes to the current allotment, and approve/deny requests from members.
6. The Admin can view a log of all recent changes committed to the allotment scheme.
7. Class representatives are given a provision to view the status of the department faculty.
8. The system makes use of the University information security requirements to provide the login facility to the users.

2.3 User Characteristics

The users of the system are students, professors of the department and the administrators who maintain the system. The members and the Admin are assumed to have basic knowledge of the computers and Internet browsing. The administrators of the system have more knowledge of the internals of the system and are given the provision to approve/deny requests made by members of the department. The proper user interface, users manual, online help and the guide to install and maintain the system must be sufficient to educate the users on how to use the system without any problems.

2.4 Constraints

1. The information of all the users must be stored in a database that is accessible by the Resource Reservation System.
2. The university information security system must be compatible with Internet applications.
3. The Resource Reservation System is connected to the university computer and is running all 24 hours a day.
4. The users access the Resource Reservation System from any computer that has Internet browsing capabilities and an Internet connection.
5. The users must have their correct usernames and passwords to enter into the Resource Reservation System.

2.5 Assumptions and dependencies

1. The users have sufficient knowledge of computers.
2. The University computer should have Internet connection and Internet server capabilities.
3. The users know the English language, as the user interface will be provided in English.

4. The application can access the university resource database.

3. Specific Requirements

This section describes in detail all the functional requirements.

3.1 Functionality

3.1.1 Login Capabilities

The system shall provide the users with login capabilities.

3.1.2 Access through Internet

The RRS can be accessed through the Internet for users with valid login.

3.1.3 Alerts

The system can send a mail to the professors and class reps in case of changes in the allotment.

3.2 Usability

- The system shall allow the users to access it from the Internet using HTML or its derivative technologies. The system uses a web browser as an interface.
- Since all users are familiar with the general usage of browsers, no specific training is required.
- The system is user friendly and self-explanatory.

3.3 Reliability

The system has to be very reliable due to the importance of data and the damages incorrect or incomplete data can do.

3.3.1 Availability

The system is available 100% for the user and is used 9 hrs a day and 365 days a year. The system shall be operational 24 hours a day and 7 days a week.

3.3.2 Mean Time Between Failures (MTBF)

The system will be developed in such a way that it may fail once in a year

3.3.3 Mean Time to Repair (MTTR)

Even if the system fails, the system will be recovered back up within an hour or less.

3.3.4 Accuracy

The accuracy of the system is limited by the accuracy of the speed at which the admin responds to the requests posted by any user.

3.3.5 Maximum Bugs or Defect Rate

Not specified.

3.3.6 Access Reliability

The system shall provide 100% access reliability.

3.4 Performance

3.4.1 Response Time

The user's home page should be rendered within seconds upon entering the correct login credentials. The information is refreshed every two minutes. The system shall respond to the member when the admin has processed and accepted/denied the submitted request. The system shall be allowed to take more time when there is a clash in reservations by any two or more parties and the resources need to be allocated accordingly and dynamically.

3.4.2 Administrator

The system shall take as less time as possible to provide the resource request details to the administrator.

3.4.3 Throughput

The number of reservation requests is directly dependent on the number of users, the users may be the Admin, Professors and also the students of the departments that may want to view the timetable for the day or request for faculty, report issues as in the case of a class representative.

3.4.4 Capacity

The system is capable of handling 25 users at a time.

3.4.5 Resource Utilization

The resources are modified according to the user requirements and also according to the classrooms, seminar halls or laboratories requested by the users.

3.5 Supportability

The system designers shall take into consideration the following supportability and technical limitations.

3.5.1 Internet Protocols

The system shall comply with the TCP/IP protocol standards and shall be designed accordingly.

3.5.2 Information Security Requirement

The system shall support and use the same standard as the SSN information security requirements.

3.5.3 Maintenance

The maintenance of the system shall be done as per the maintenance contract.

3.5.4 Standards

The coding standards and naming conventions will be as per the American standards.

3.6 Design Constraints

3.6.1 Software Language Used

The languages that shall be used for coding the RRS are HTML, CSS, JavaScript and Java. For working on the coding phase of the RRS, the Internet Information Services (IIS) Server needs to be installed.

3.6.2 Development Tools

Will make use of NetBeans IDE, SQL Server and Bootstrap framework. Also will make use of the online references available for developing programs in HTML, CSS, Bootstrap, JavaScript and Java.

3.6.3 Class Libraries

Will make use of the existing libraries in the languages and frameworks specified above.

3.7 On-line User Documentation and Help System Requirements

Online help is provided for each of the features available with the RSS. All the applications provide an online help system to assist the user. Online help is provided for each and every feature provided by the system. The User Manual describes the use of the system to the admin, professors and students. An installation document will be provided that includes the installation instructions and configuration guidelines, which is important to a full solution offering. Also, a ReadMe file is typically included as a standard component. The ReadMe includes a “What’s New With This Release” section, and a discussion of compatibility issues with earlier releases. Most users also appreciate documentation defining any known bugs and workarounds in the ReadMe file.

3.8 Purchased Components

The System Administrator will need to purchase the license for IIS Server. Mostly it is available with Windows Environment. So the system need not purchase any licensing products.

3.9 Interfaces

1) Home Page

HOME

This system is proposed to

.

(description)

LOGIN HERE

2) Navigation bars

Student

Home	View Timetable	Help	Class Alloc
------	-------------------	------	----------------

Student (Rep)

Home	View Timetable	Class Allocation	Request changes	Report	Help
------	-------------------	---------------------	--------------------	--------	------

Professor / Student (Club Head)

Home View Timetable Class Allocation Request charges Help

login Page

LOGIN

Username :

Password :

Forgot Password?

Admin Navigation

Home Add/update details View Allocation View requests Help View log

TimeTable View (Students) and General view

DAILY TIMETABLE

Class _____

	I	II	III	IV	V	VI	VII	VIII
Mon								
Tue								
Wed								
Th								
Fri								

Help Page

A general help view
for ADMIN
STUDENTS
REPS
PROFESSORS

Time Table (Professor View)

PROFESSOR - TIMETABLES.

Select class handled to view timetable.

Drop-Down v
C8
C2
C1
C3

classRoom and Lab Status (select time and view status)

CLASS STATUS

C1	C4	Lab1	C7 v
C2	C5	Lab2	C8
C3	C6	Seminar Hall	C9

v
8:15 - 9:15
9:15 - 10:15
10:15 - 11:15
11:15 - 12:15
12:15 - 1:15

Report Issues (Rep)

REPORT ISSUES

ClassName

Room No.

Your Name.

Issue

Request Changes (Professor)

Name

Type

v
Extend
Exchange
Cancel
Book

Purpose

Strength

No of Hours

SUBMIT

Request changes (Rep)

Name

Call Professor

Class

Number of hours

Request changes (Club Head)

Name

Purpose

Strength

Faculty Incharge

No. of hours

SUBMIT

Admin Details Updation

Update details

Lab softwares

Add Timetable

View reports

Modify timetable

3.10 Licensing Requirements

The usage is restricted to only SSN College Of Engineering.

3.11 Legal, Copyright, and Other Notices

RRS is owned and maintained by the CSE department of SSNCE and cannot be used without its consent.

3.12 Applicable Standards

The ISO/IEC 6592 guidelines for the documentation of computer based application systems will be followed.

4. Supporting Information

The use-case storyboards or the user-interface prototypes are not available. The appendices are not to be considered as part of the requirements.

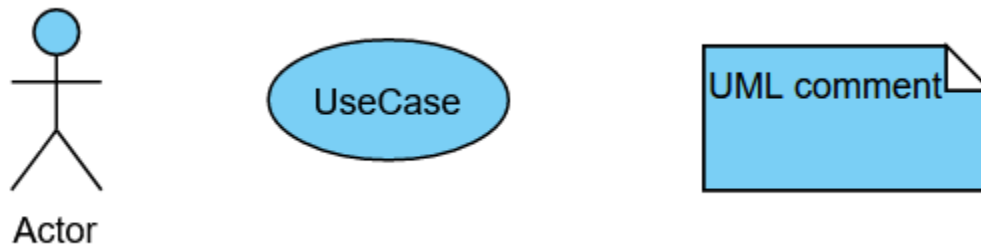
DATE:21.03.2022

ASSIGNMENT 3
USE CASE DIAGRAM

AIM:

Creation of a UML Use case model for resource reservation system.

NOTATIONS:



Actor: An actor in use case modeling specifies a role played by a user or any other system that interacts with the subject.

Use case: A use case is a description of how a person who actually uses that process or system will accomplish a goal.

UML comment: A comment is a textual annotation that can be attached to a set of elements.

IDENTIFICATION OF ACTORS:

1. Professor
2. Student
3. Student Representative (Rep)
4. Club head
5. Administrator (Admin)

IDENTIFICATION OF SCENARIOS:

1. **Success scenarios:**
 - a. Generation of timetable and initial classroom allocation by admin
 - b. View timetable and current classroom allocation
 - c. Login and validate user

- d. Reservation stating reason, cancellation, exchange, handing over responsibility of classes to another professor
- e. Reservation of classroom/seminar halls by club head stating reason.
- f. Request to handle classes by professor, reporting classroom issues by class representatives.
- g. Approval or rejection of requests by admin according to purpose
- h. Accommodate new arriving batches/satisfy special requests into the current allocation scheme
- i. View activity log by admin

2. Failure scenarios:

- a. Rearrangement of classes in event of non availability to satisfy the request.
- b. Rejection of request made by club head for invalid reasons

3. Subfunctions:

- a. Check availability
- b. Get request parameters
- c. View professor status

RELATING USE CASES:

1. Generalization:

A generalization relationship is a relationship that implements the concept of object orientation called inheritance. The generalization relationship occurs between two entities or objects, such that one entity is the parent, and the other one is the child.

2. Association:

a. <<includes>>

In UML modeling, an include relationship is a relationship in which one use case (the base use case) includes the functionality of another use case (the inclusion use case). The include relationship supports the reuse of functionality in a use-case model.

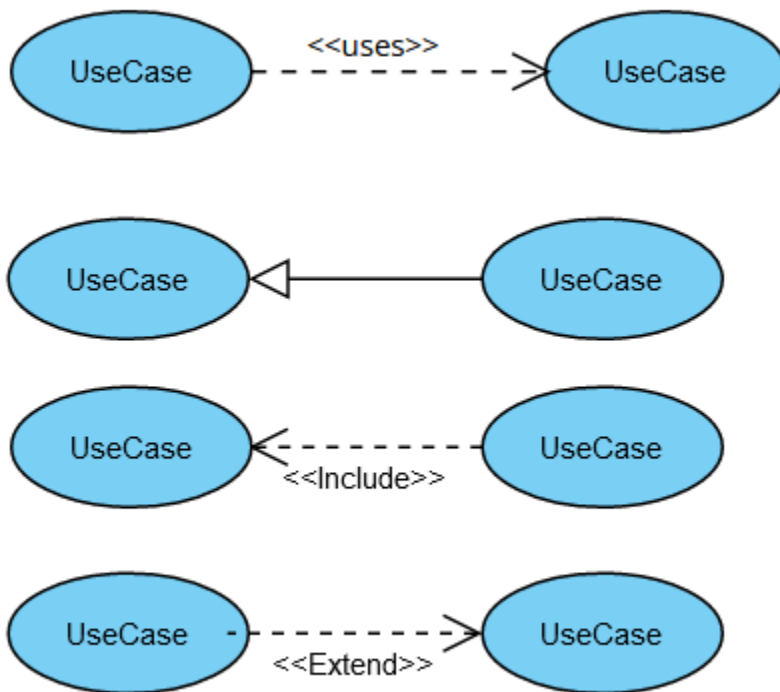
b. <<extends>>

The extension use case consists of one or several behavior sequences (segments) that describe additional behavior that can incrementally augment the behavior of the base use case. Each segment can be inserted into the base use case at a different point, called an extension point.

c. **<<uses>>**

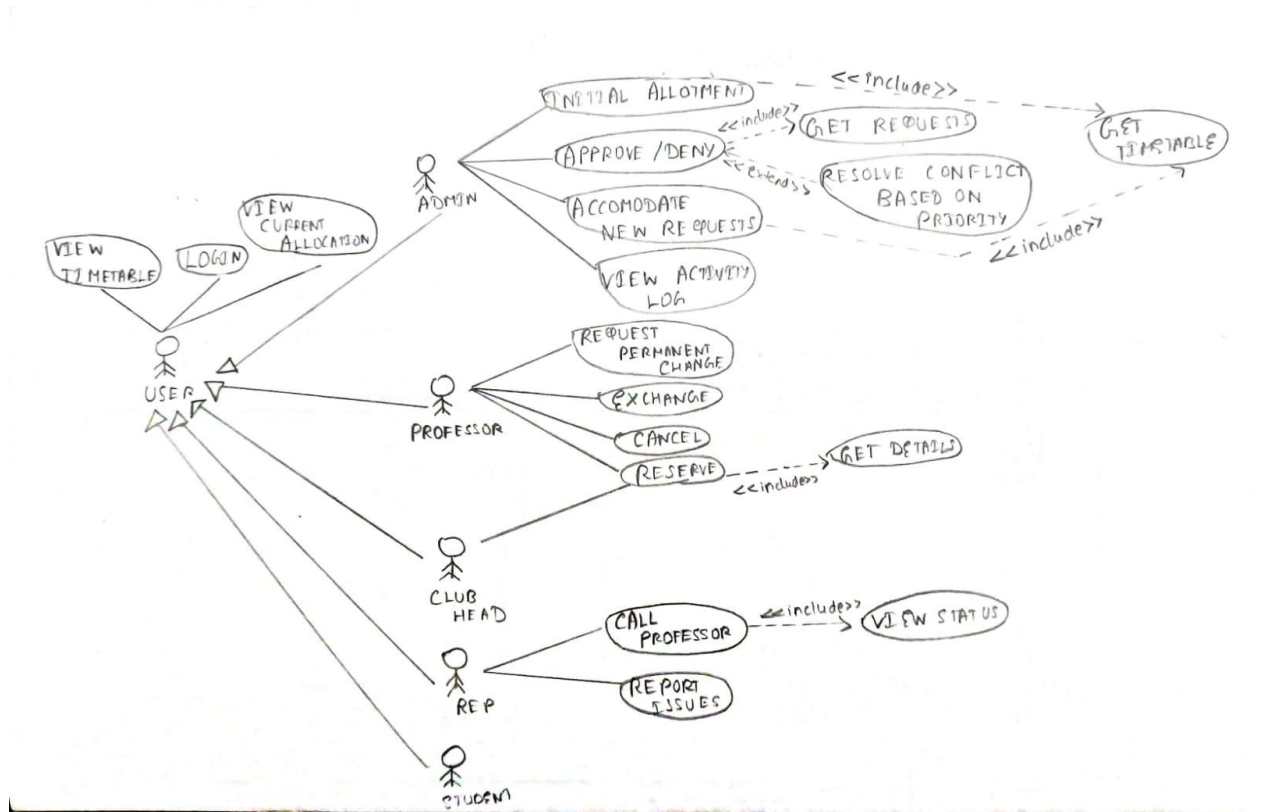
In UML modeling, a usage relationship is a type of dependency relationship in which one model element requires another model element for full implementation or operation.

NOTATIONS FOR RELATING USE CASES

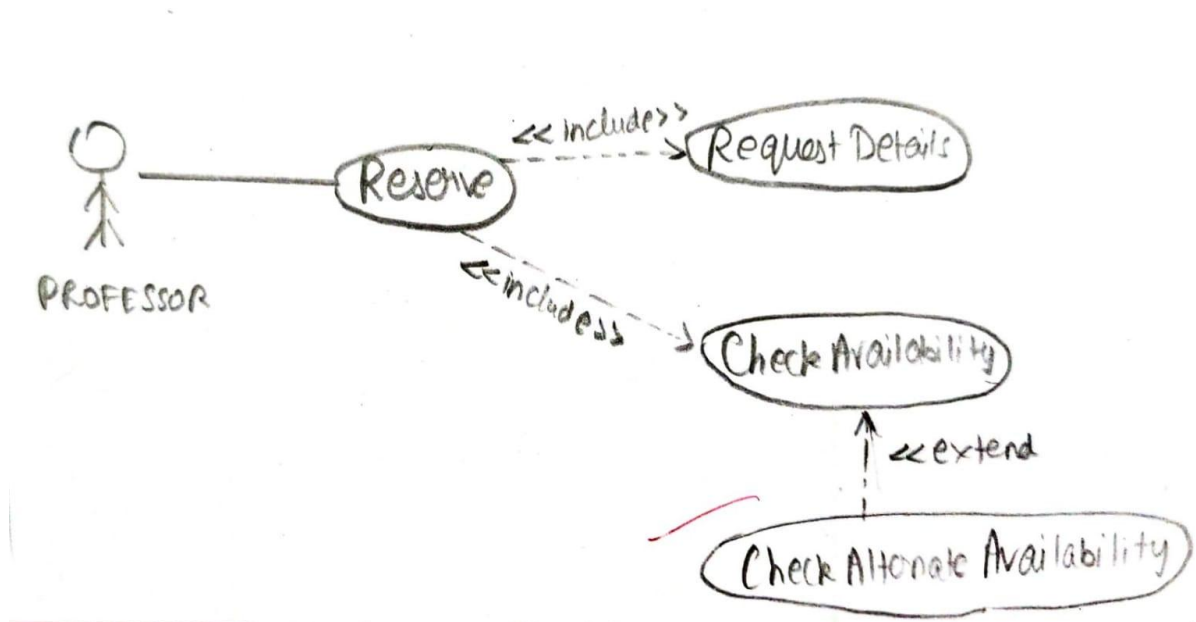


USE CASE DIAGRAMS:

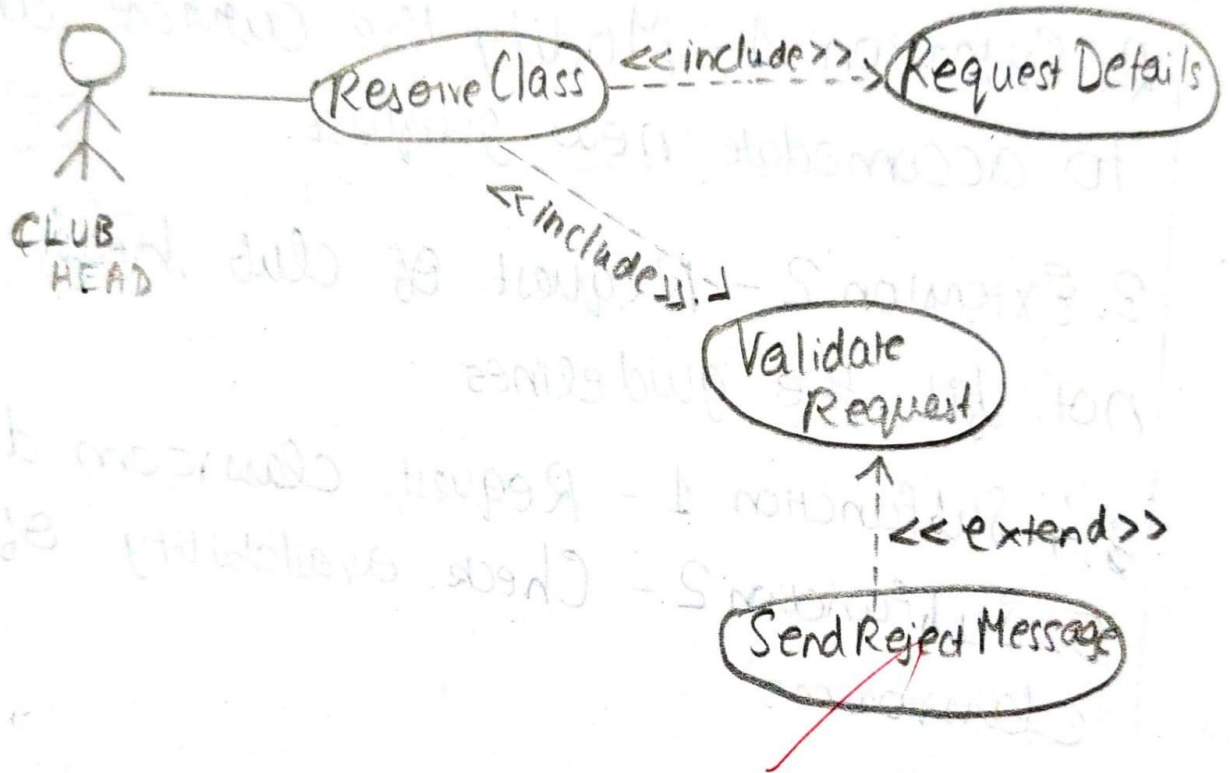
a. Main success scenario



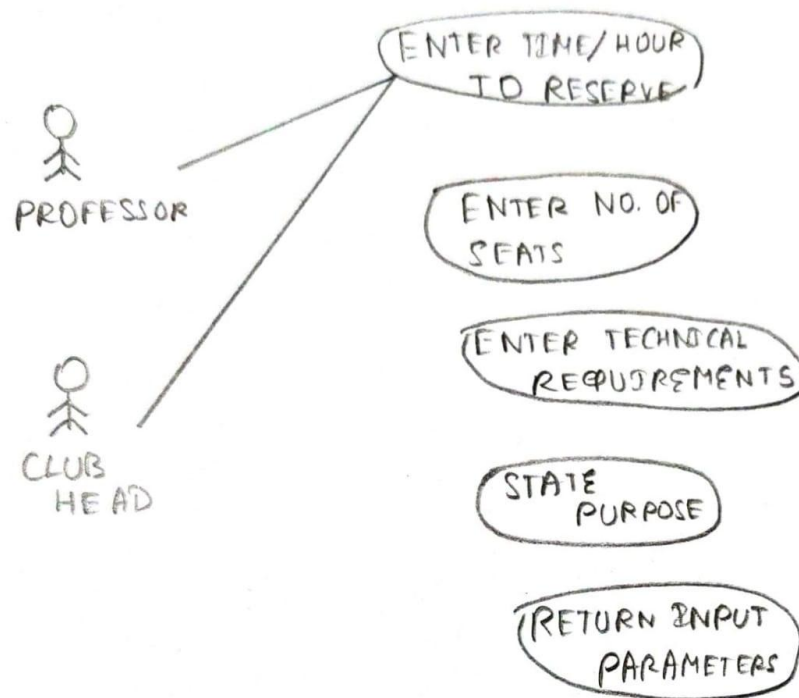
b. Frequent Alternate Success Scenario - 1: Rearrangement of classrooms



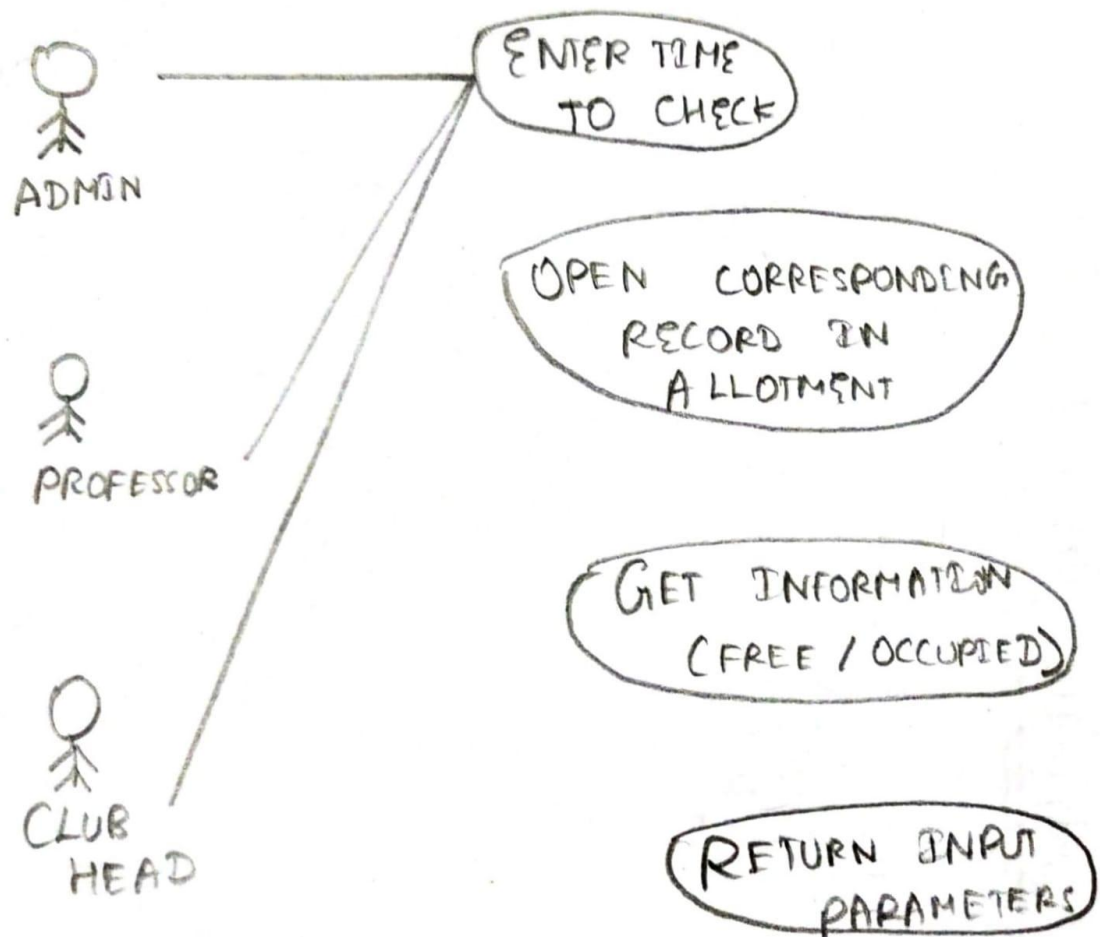
c. Frequent Alternate Success Scenario - 2: Rejection of club head requests



d. Important subfunction - 1: Request classroom details



e. Important subfunction - 2: Check availability



FULLY DRESSED USE CASE DESCRIPTIONS:

Scope: Resource Reservation System

Level: User goal

Primary actor: Admin, Faculty, Students

Stakeholders and interests:

- a. **Admin:** Efficient allocation, overall supervision by viewing activity log, approval and deny of requests based on priority
- b. **Faculty:** Reservation, cancellation, extension and exchange of classes, handover of responsibilities to another faculty member.
- c. **Students:** View timetable and current allocation of classrooms.

- d. **Student (representatives):** Request professor to handle a free class and report issues such as faulty projectors pertaining to classrooms.
- e. **Student (club head):** Request reservation of classrooms/seminar halls for club related activities.
- f. **CSE department:** Efficient and hassle free resource allocation for all users.

Precondition: The users are logged in and authenticated successfully.

Success guarantee: Users can view the timetables and the current classroom allotment. Professor and club head reservation requests are handled successfully and approved/denied accordingly. Representatives can report issues such as faulty equipment which are then acted upon at the earliest. All operations performed on the system can be viewed by the admin in the activity log.

a. Main success scenario:

1. Admin allowed access before the beginning of semester.
2. Admin generates a timetable and allots classrooms and labs for each class at the beginning of the semester.
3. Professor views their timetable.
4. Professor views the timetable of the class handled by them.
5. Professor reserves a classroom on entering the number of hours, equipment required along with the strength of students expected.
6. Professor exchanges the class they handle with another professor.
7. Professor extends class hours.
8. Professor cancels class hours.
9. Professor requests to hand over responsibilities permanently.
10. Admin incorporates permanent changes to timetable and allotments.
11. Student views timetable for the day.
12. Club Head requests classroom reservation on entering the number of hours and a valid purpose.
13. Admin approves/denies requests on the basis of priority.
14. Class Rep reports issues pertaining to the classroom.
15. Class Rep views the status of professors handling their class.
16. Class Rep requests any professor that is free to handle the class.
17. Admin views activity log.

b. Frequent Alternate Success Scenario - 1 (Rearrangement of classes to satisfy the requirement):

1. In the event of non availability of classrooms to satisfy a request, the existing system can be rearranged depending on the class strength.

2. Appropriate alerts are sent to the people concerned with the rearrangement to their mails.

c. Frequent Alternate Success Scenario - 2 (Rejection of a request made by a club head for an invalid request):

1. When a club head provides a reason for reservation that does not follow the Reservation Guidelines and protocols, the request is rejected by admin.
2. The club head is prompted of the rejection in advance through appropriate alerts.

d. Important subfunction - 1 (Check availability of classroom/lab/seminar halls):

1. Enter details of the class like strength, technical requirements if any, number of hours etc.
2. System checks the timeslot and requirements with the existing allotment for a classroom
3. If an empty classroom is found, provide a list of classrooms and allow requests from professors and club heads.

e. Important subfunction - 2 (Request classroom details):

1. Enter time(hour) for reservation of class.
2. Enter no. of seats required.
3. Input technical requirements such as projectors/OS requirements.
4. State purpose of reservation.
5. Return the parameters to the reservation function.

DOCUMENTATION:

On use of the proposed resource reservation system, classrooms are allotted to various classes successfully. As the semester progresses, professors and students (representatives and club heads) can request for changes to the current allotment, which are approved/denied by the admin on the basis of priority. All alternate scenarios including accommodation of newly arriving batches and rearrangement of classes are taken into consideration by the application. Use case diagrams and fully dressed use case descriptions are provided for all possible scenarios, along with definitions for the various notations used.

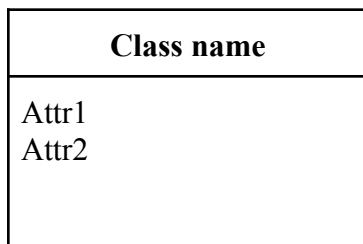
DATE: 28.03.2022

ASSIGNMENT 4
DOMAIN MODEL AND CLASS DIAGRAM

AIM:

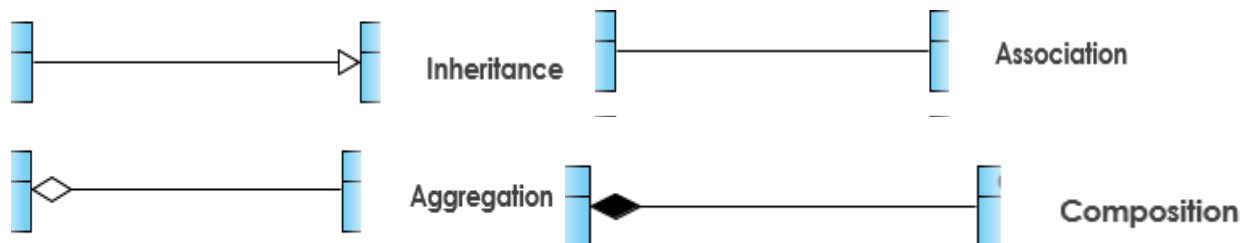
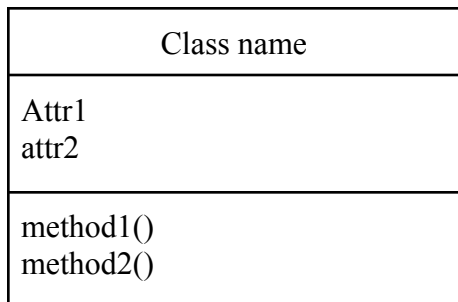
To create a domain model diagram using the noun classes and conceptual classes for Resource Reservation System

UML NOTATIONS FOR DOMAIN MODEL DIAGRAM



1 relationship *

UML NOTATIONS FOR CLASS DIAGRAM



IDENTIFICATION OF CLASSES

A. Conceptual Class Category List

<u>Conceptual class category</u>	<u>Noun phrase</u>
Physical or tangible objects	Classroom, lab, desktop, seat, projectors
Specifications, designs and descriptions	ClassDescription, LabDescription
Places	College, CSEDepartment
Transactions	reserveClass, exchangeClass
Transaction line items	reservationLineItem
Roles of people	Student, professor, club head, rep, admin
Containers of other things	Classroom, lab, seminar halls
Things in the container	Bench, monitor, projectors
Other computer/electromechanical systems external to the system	-
Abstract noun concepts	Knowledge, intelligence
Organizations	University, college
Events	Reserve, report, cancel, extend, crash
Processes	reserveClass, reportClass, cancelClass, extendClass
Rules and policies	reservationPolicy, ExchangePolicy
Catalogs	-
Record of finance, work	activityLog
Financial instruments	-
Manuals, records, papers	User manual

B. Identification Of Noun Phrases

Class, professor, student, rep, clubhead, lab, seminar hall, alert, admin, issues, seats, purpose, semester, initialAllotment, priority, timetables, activityLog, batch

IDENTIFICATION OF ASSOCIATIONS

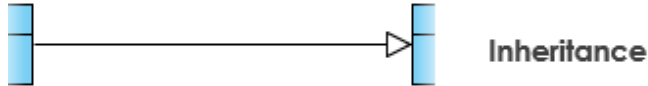
A. Association Category List

<u>Category</u>	<u>Examples</u>
A is a physical part of B	-
A is a logical part of B	strength-class
A is physically contained in B	Projector-classroom, bench-classroom, seat-classroom, monitor-classroom, system-lab, seats-class
A is logically contained in B	timetable-classroom
A is a description of B	classroom-classDescription, lab-LabDescription
A is a line item for B	reserveClass-reservationLineItem
A is known/logged/recorded/reported/captured in B	reservation-ActivityLog
A is a member of B	Professor-college, student-college, admin-department
A is an organizational subunit of B	department-college
A uses or manages B	admin-system
A communicates with B	Professor-admin, rep-professor, clubhead-admin
A is related to transaction B	professor-reserveClass
A is a transaction related to another transaction B	professor-reserveClass
A is next to B	Classroom-classroom, lab-lab
A is owned by B	desktop-department
A is an event related to B	exchangeClass-professor, reserveClass-professor

B. Definition Of Associations And Their Notations (Generalization, Aggregation And Composition)

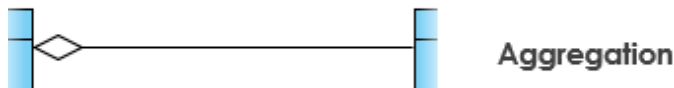
Generalization

A generalization is a relationship that implements the concept of inheritance. (an 'is-a-kind-of' or 'is-a' relationship) .The child class is-a-kind-of a base class.



Aggregation

An aggregation is a relationship between classes that says one class is a part of another class (an 'is a part of' relationship). In aggregation, the part (a constituent) is meaningful without the whole (aggregate).



Composition

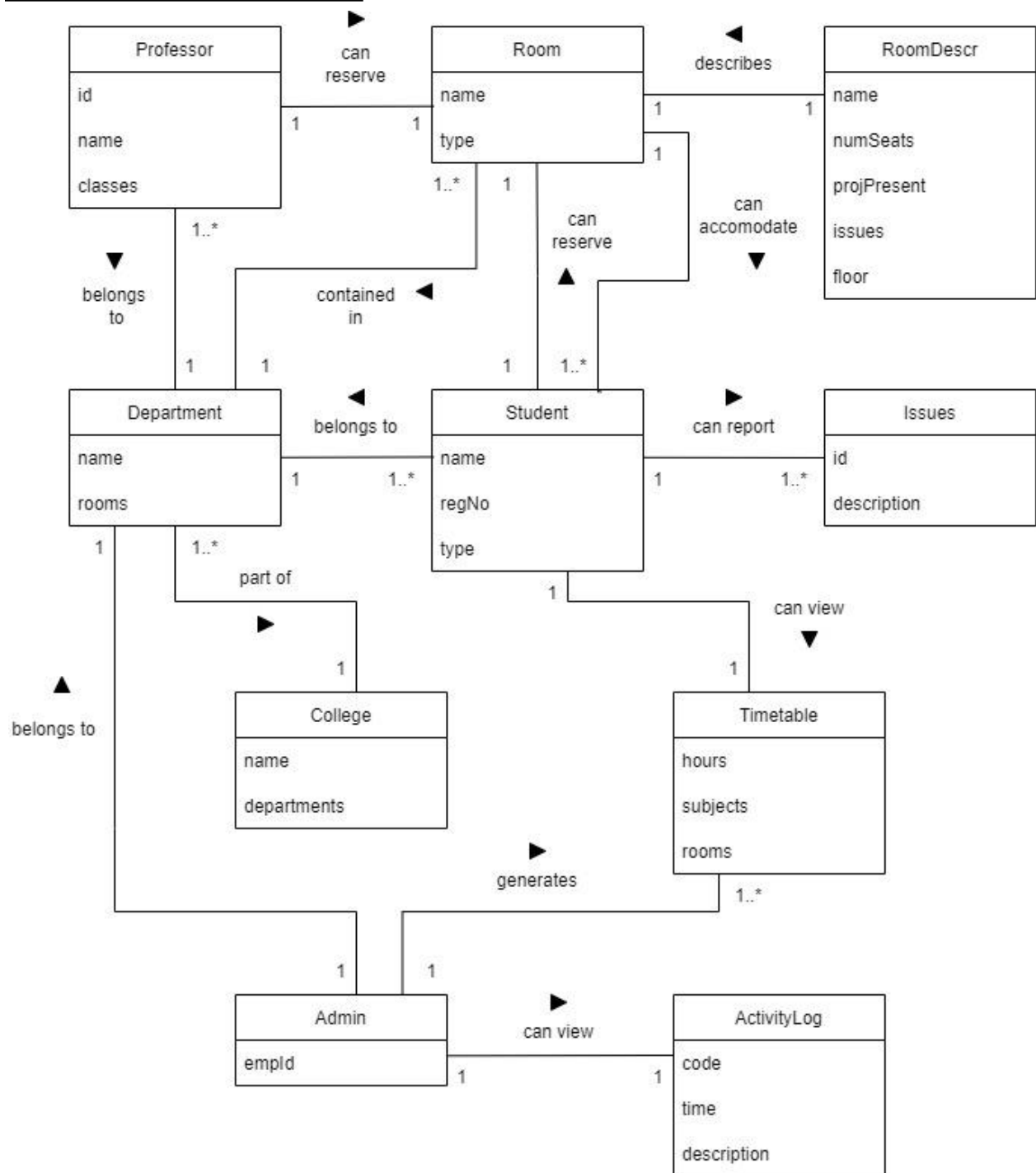
Composition is a relationship between two classes where one class is a part of another (an 'is a part of' relationship). In composition, the part (component) is not meaningful without the whole (container).



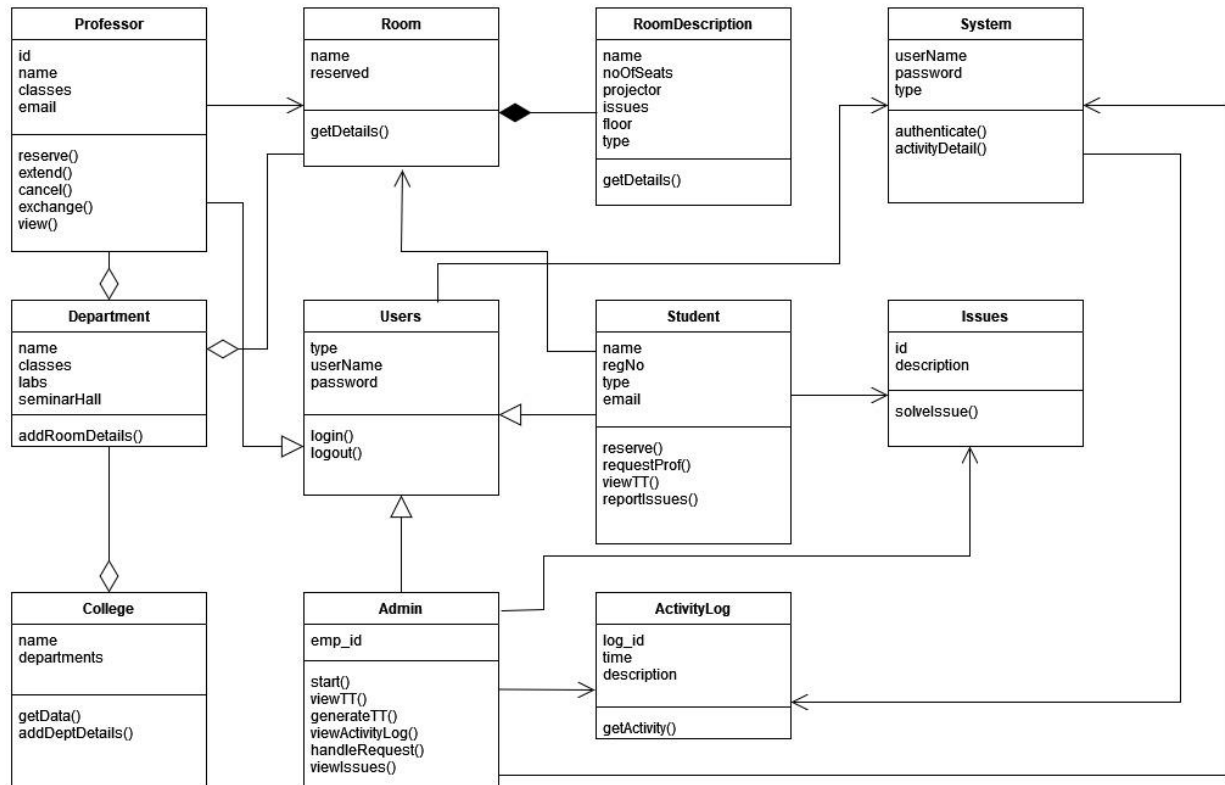
C. Multiplicity Based Associations (List All Possible Multiplicity)

- 1 professor can reserve 1 class
- 1 classroomDescription can describe 1 class
- 1 to * classes are contained in department
- 1 to * professors belong to 1 department
- 1 to * students belong to 1 department
- 1 to * departments belong to 1 college
- 1 to * admins belong to 1 college
- 1 admin can view many activityLogs
- 1 admin generates many timetables
- 1 admin can view many issues
- 1 professor can view many timetables
- 1 student can view one timetable

DOMAIN MODEL DIAGRAM



CLASS DIAGRAM



DOCUMENTATION

On the use of the proposed resource reservation system, classrooms are allotted to various classes successfully. As the semester progresses, professors and students (representatives and club heads) can request changes to the current allotment, which are approved/denied by the admin on the basis of priority. The noun phrases and conceptual classes, associations, and multiplicity of associations have been identified and the domain model diagrams are drawn. The class diagrams are drawn using the nouns and relationships like generalization, aggregation and composition.

DATE:04.04.2022

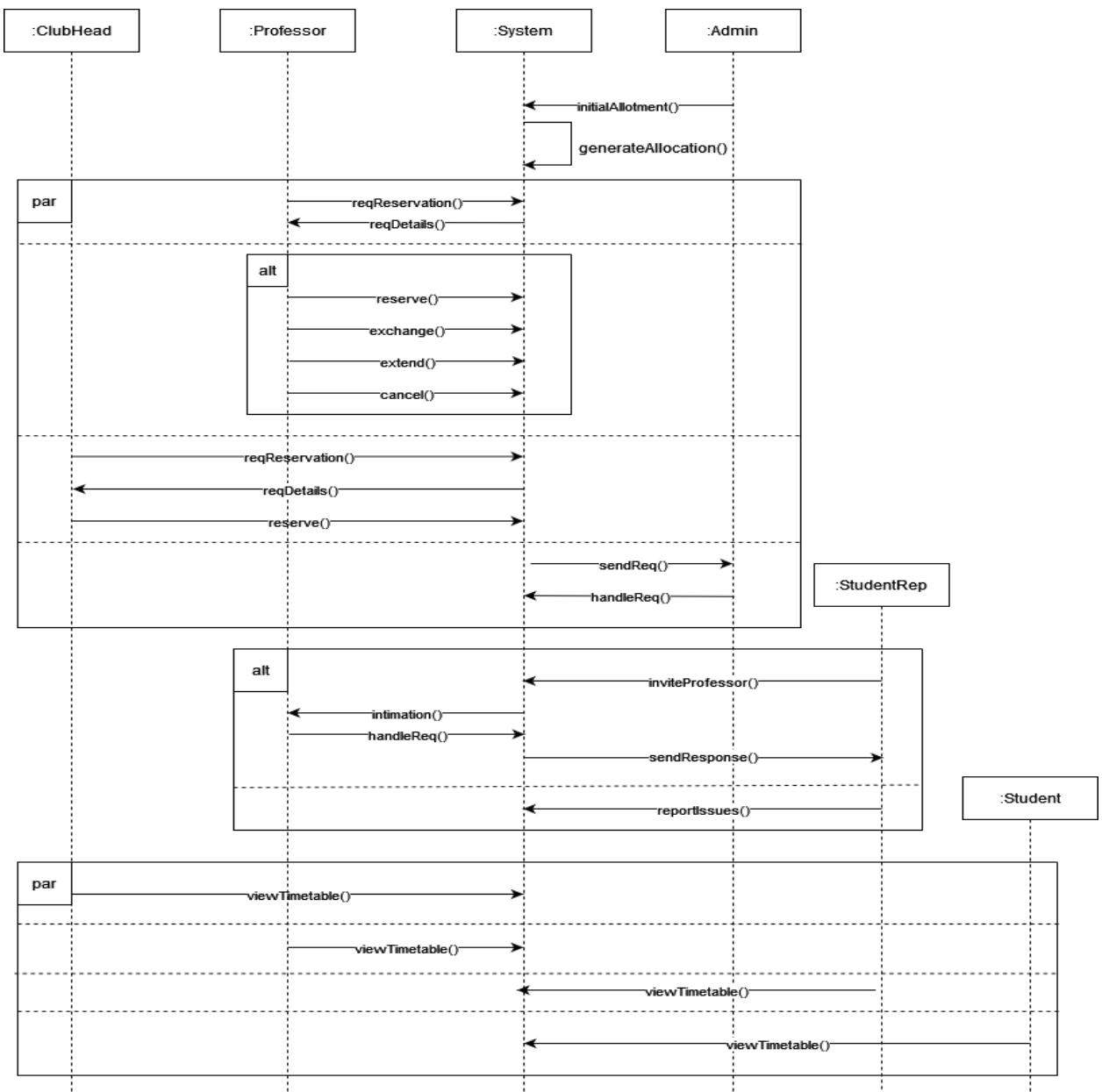
ASSIGNMENT - 5
INTERACTION DIAGRAM

AIM:

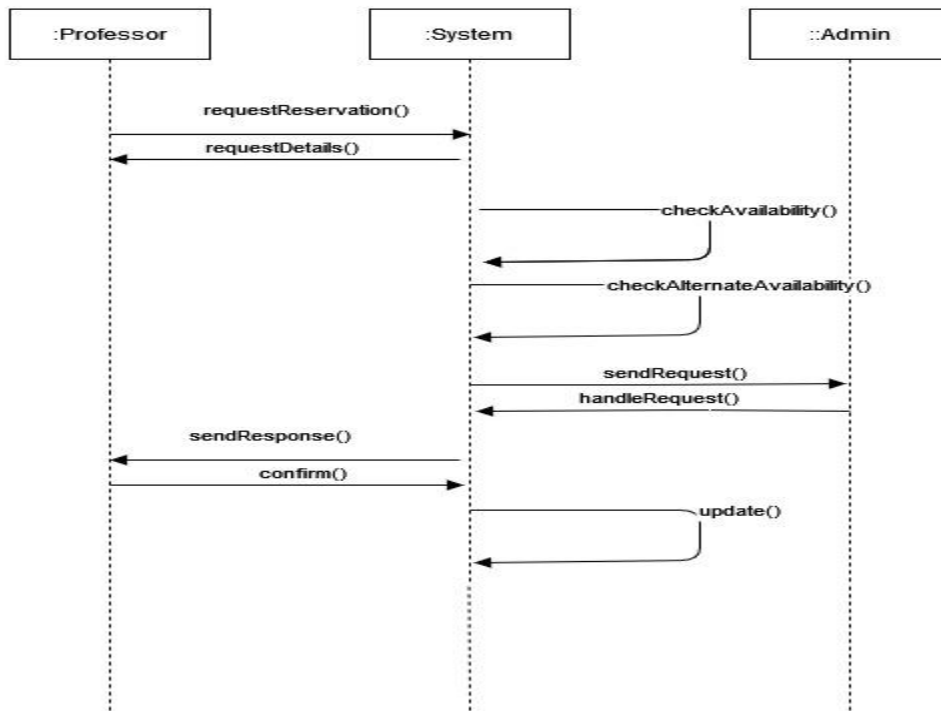
To find the interaction between objects and represent them using UML Sequence diagrams

OUTPUT:

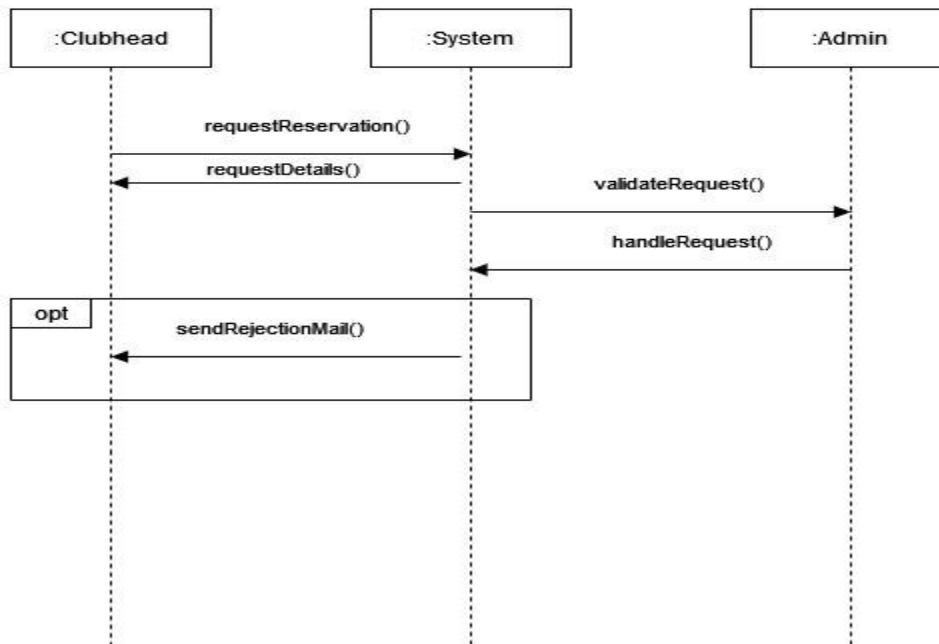
1. Main success scenario



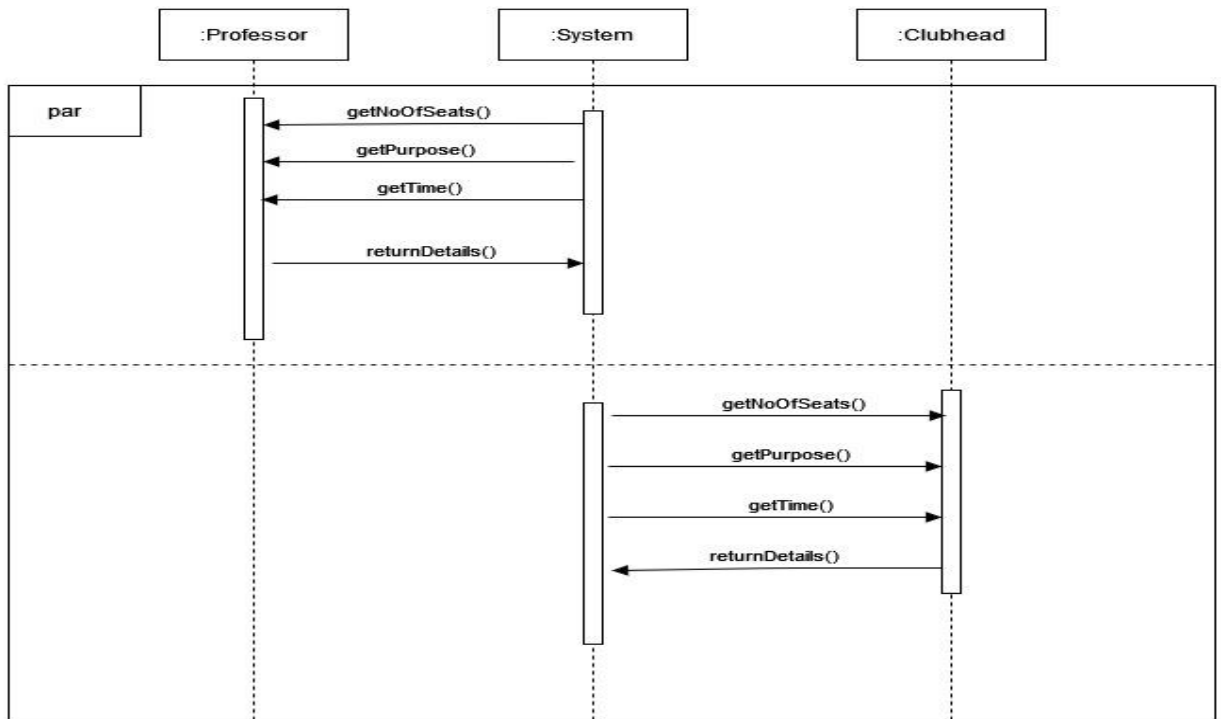
2. Alternate success scenario - 1 (Rearrangement of current configuration request to accomodate new request)



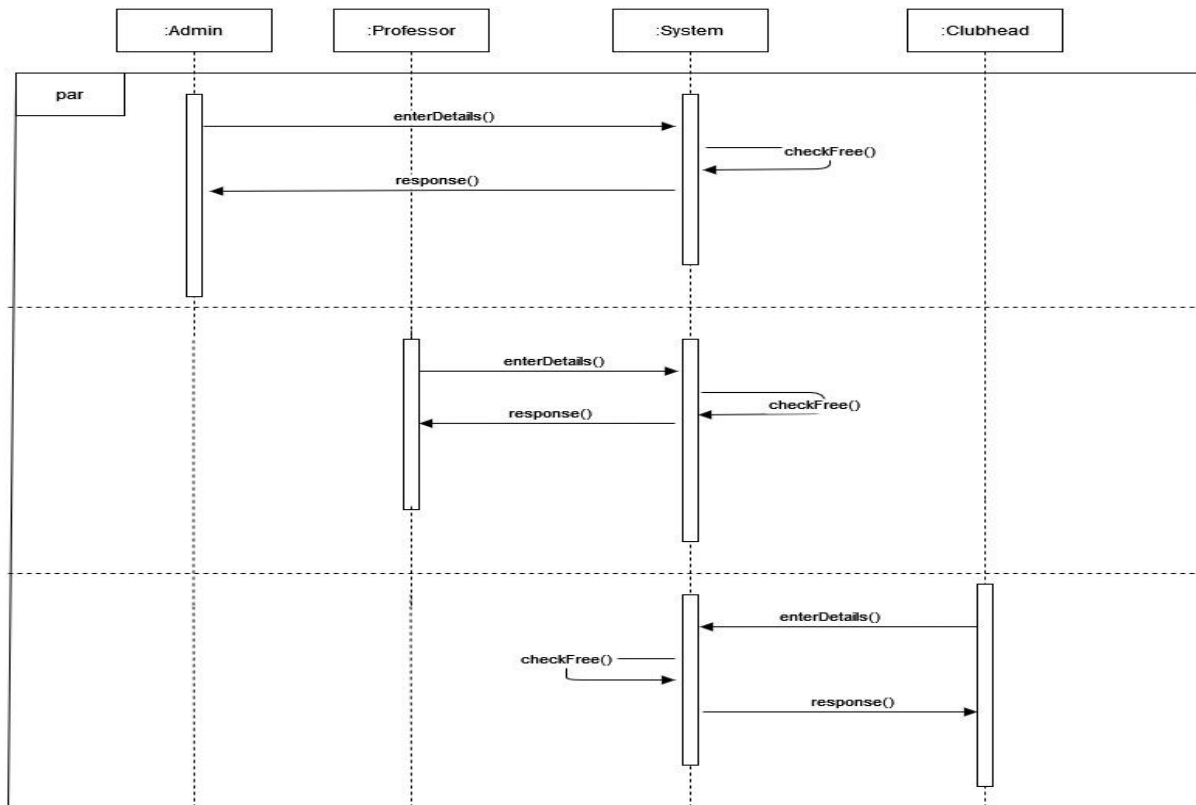
3. Alternate success scenario - 2 (Handle invalid request from clubhead)



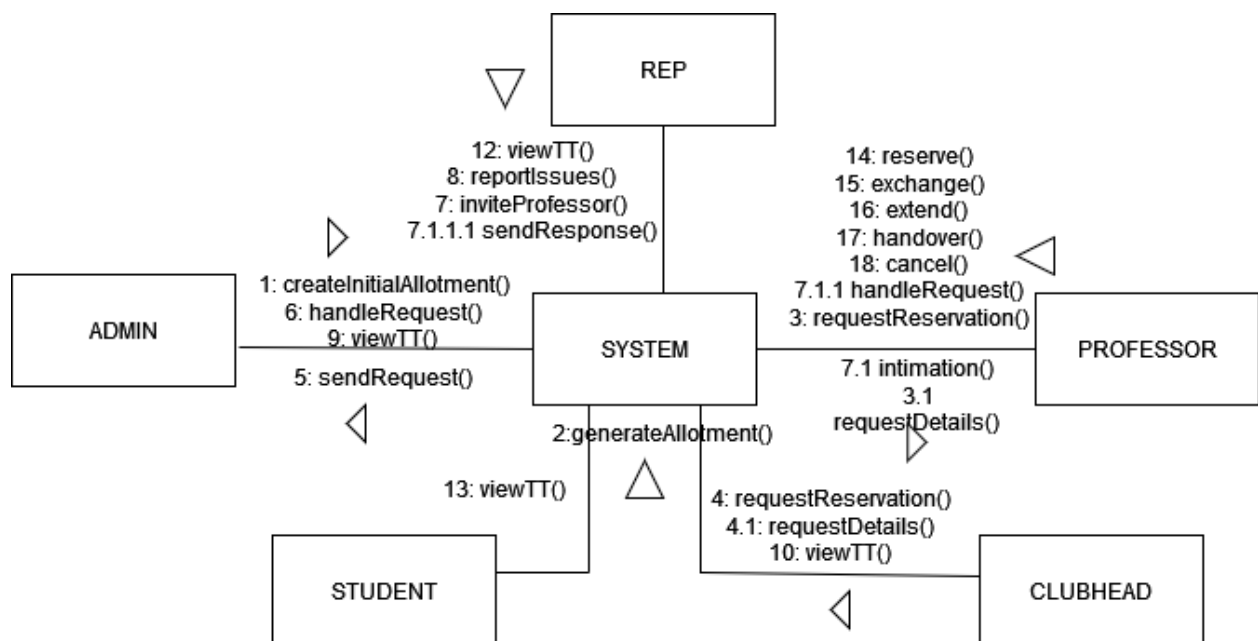
4. Subfunction - 1 (Request classroom details)



5. Subfunction - 2 (Check availability)



6. Collaboration diagram



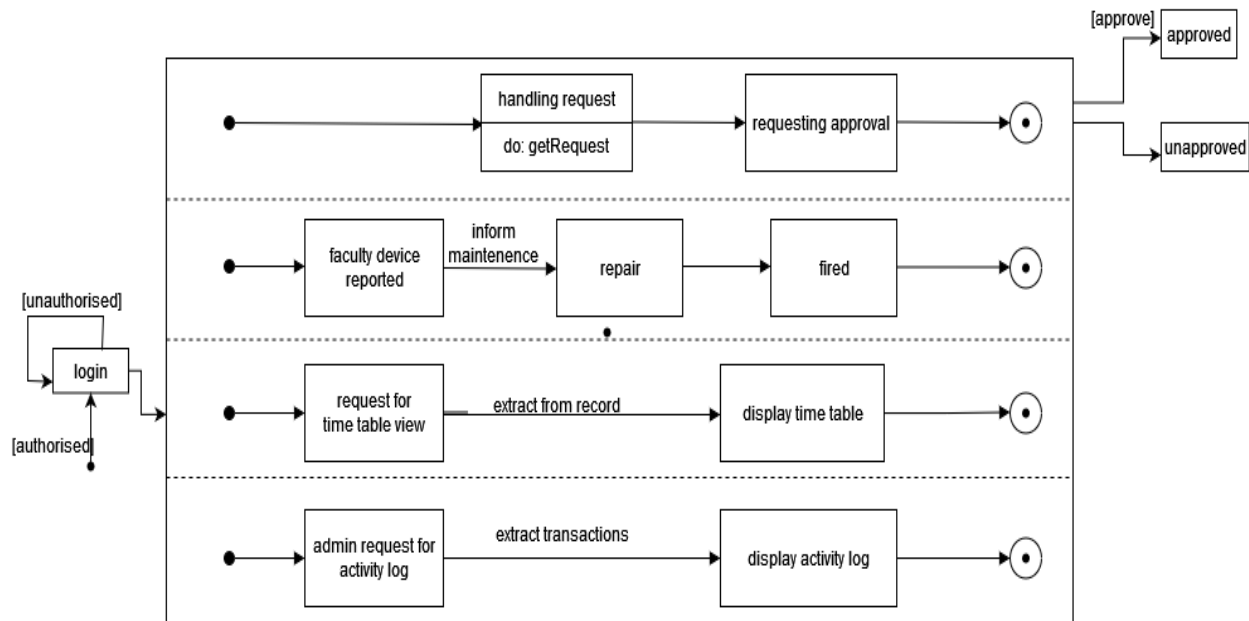
DATE: 11.04.2022

ASSIGNMENT 6
STATE CHART DIAGRAM AND ACTIVITY DIAGRAM

AIM:

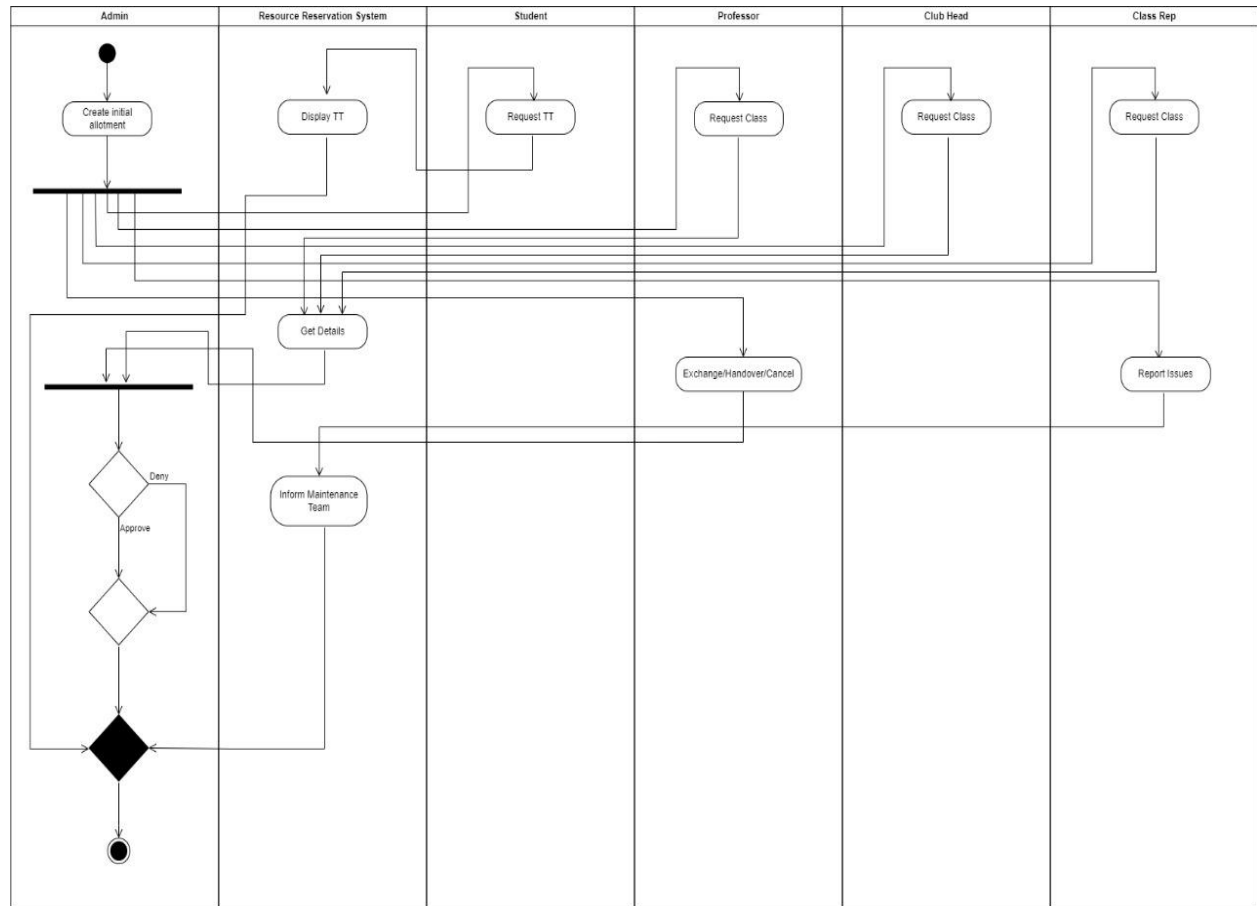
To create a state machine and activity diagram for the Resource Reservation System.

STATE MACHINE DIAGRAM:



state machine diagram

ACTIVITY DIAGRAM:



DATE:18.04.2022

ASSIGNMENT 7
IMPLEMENTATION DEMO

SOURCE CODE FILES:

index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';
// ReactDOM.render(<TTContainer />, document.getElementById("root"))

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
    { /* <TTContainer /> */ }
  </React.StrictMode>
);

reportWebVitals();
```

App.js

```
import React from "react";
import "./App.css";
// importing components from react-router-dom package
import {
  BrowserRouter as Router,
  Routes,
  Route,
} from "react-router-dom";

// import Home component
import HomePage from "./Home";
// import Login component
import Login from "./Login";
//import Class Status component
import ClassStatus from "./ClassStatus";
```

```
//import Report Issue component
import ReportIssue from "../ReportIssue";
//import Report Issue component
import SolveIssue from "../SolveIssue";
//import TT components
import TTContainer from "../TimetableComponents/TTContainer";
// import { render } from "@testing-library/react";
```

```
class App extends React.Component{
  render() {

    return (
      <Router>
        { /* <div className="App">
          <ul className="App-header">
            <li>
              <Link to="/home">Home</Link>
            </li>
            <li>
              <Link to="/login">Login</Link>
            </li>
            <li>
              <Link to="/timetable">View TT</Link>
            </li>
          </ul> */ }
        <Routes>
          <Route exact path="/login" element={<Login/>} />
          <Route path="/home" element={<HomePage/>} />
          <Route path="/timetable" element={<TTContainer/>} />
          <Route path="/checkstat" element={<ClassStatus/>} />
          <Route path="/report" element={<ReportIssue/>} />
          <Route path="/solve" element={<SolveIssue/>} />
        </Routes>
        { /* </div> */ }
      </Router>
    );
  }
}
```

```
export default App;
```

ClassStatus.js

```
import React from 'react';
```

```

import "./Login.css"
import data from "./rooms.json"

class ClassStatus extends React.Component {

  retStatus(event) {
    event.preventDefault();
    var r = document.getElementById("room").value;
    var found=0;
    for(const d of data){
      if(d.room === r) {
        var text = "Status : " + d.status;
        if(text==="Status : Busy"){
          document.getElementById("content").setAttribute("style", "color: red;");
        }
        else{
          document.getElementById("content").setAttribute("style", "color: black;");
        }
        document.getElementById("content").innerHTML=text;
        found=1;
        break;
      }
    }
    if(found===0){
      const text ="Room not found !";
      document.getElementById("content").innerHTML = text;
    }
  }

  render() {
    return (
      <>
      {
        //<Link to="/">Login</Link>
      }
      <form className="login-form">
      <br/>
      <h3>Check Class Availability</h3>
      <input id="room" className="uname" placeholder="Enter class" type="text" />
      <button onClick = {this.retStatus} className="btn">Check Status</button>
      <br/>
      </form>
      <div><p id="content"></p></div>
      </>
    )
  }
}

```

```
}
```

```
export default ClassStatus;
```

Home.js

```
import React from 'react';  
import './App.css';  
import { Link } from 'react-router-dom';
```

```
class HomePage extends React.Component {  
  render() {  
    return (  

```

```
      <>  
      <div className="App">  
        <ul className="App-header">  
          <li>  
            <Link to="/home">Home</Link>  
          </li>  
          <li>  
            <Link to="/checkstat">Check Availability</Link>  
          </li>  
          <li>  
            <Link to="/timetable">View TT</Link>  
          </li>  
          <li>  
            <Link to="/report">Report Issue</Link>  
          </li>  
          <li>  
            <Link to="/login">Logout</Link>  
          </li>  
        </ul>  
      </div>
```

```
      <div id='desc'>  
        <h1>Resource Reservation System</h1>  
        <p className="para">  
          The classrooms in our department, in addition to being a venue for conducting regular  
classes,  
          can also serve as examination halls and spaces for club meetings.  
          We have decided to investigate the use of a classroom allocation system System to  
facilitate  
          hassle free, user friendly and efficient allotment of classroom resources.
```

This system would be used by members who may be students or professors of that University to check the availability of the classrooms, seminar halls, examination centers and labs. The purpose of this document is to analyze and elaborate on the high-level needs and features of the RRS. It focuses on the capabilities and facilities provided by an RRS. The details of what all are the needs of the RRS and if it fulfills these needs are detailed in the use-case and supplementary specifications.

This system can be extended to all departments, schools and workspaces with necessary modifications.

```
</p>
</div>
</>
);
}
}
```

```
export default HomePage;
```

Login.js

```
import React from 'react';
import './Login.css'
import prof from './profile.jpg'
import {Link} from 'react-router-dom'
```

```
class Login extends React.Component {
```

```
  render() {
    return (
      <div>
        {
          //<Link to="/">Login</Link>
        }
        <form className="login-form">
          <br/><br/>
          <img className="prof-img" alt="" src={prof} />
          <input className="uname" placeholder="Username" type="text" />
          <br/>
          <input className="pwd" placeholder="Password" type="password" />
          <br/>
          <Link to="/home"><button className="btn">Login</button></Link>
          <br/><br/>
        </form>
      </div>
    );
  }
}
```



```

        </form>
      </>
    )
  }
}

export default Login;

```

ReportIssues.js

```

import React from 'react';
import './Login.css'

class ReportIssues extends React.Component {

  render() {
    return (
      <>
        {
          //<Link to="/">Login</Link>
        }
        <form className="login-form" method="POST" action="http://localhost:3002/add">
          <br/>
          <h3>Report Issue</h3>
          <input id="room" name="room" className="uname" placeholder="Enter class"
type="text" />
          <input id="issue" name="issue" className="uname" placeholder="Describe issue"
type="text" />
          <input id="criticality" name="criticality" className="uname" placeholder="Enter
criticality (1-5)" type="text" />
          <input type="submit" value="Report Issue" className="btn" />
          <br/>
        </form>
      </>
    )
  }
}

export default ReportIssues;

```

SolveIssue.js

```
import React from 'react';
import './Login.css'

class SolveIssue extends React.Component {

  render() {
    return (
      <>
      {
        //<Link to="/">Login</Link>
      }
      <form className="login-form" method="POST" action="http://localhost:3002/delete">
      <br/>
      <h3>Resolve Issue</h3>
      <input id="room" name="room" className="uname" placeholder="Enter class"
type="text" />
      <input type="submit" value="Solve Issue" className="btn" />
      <br/>
      </form>
      </>
    )
  }
}
```

export default SolveIssue;

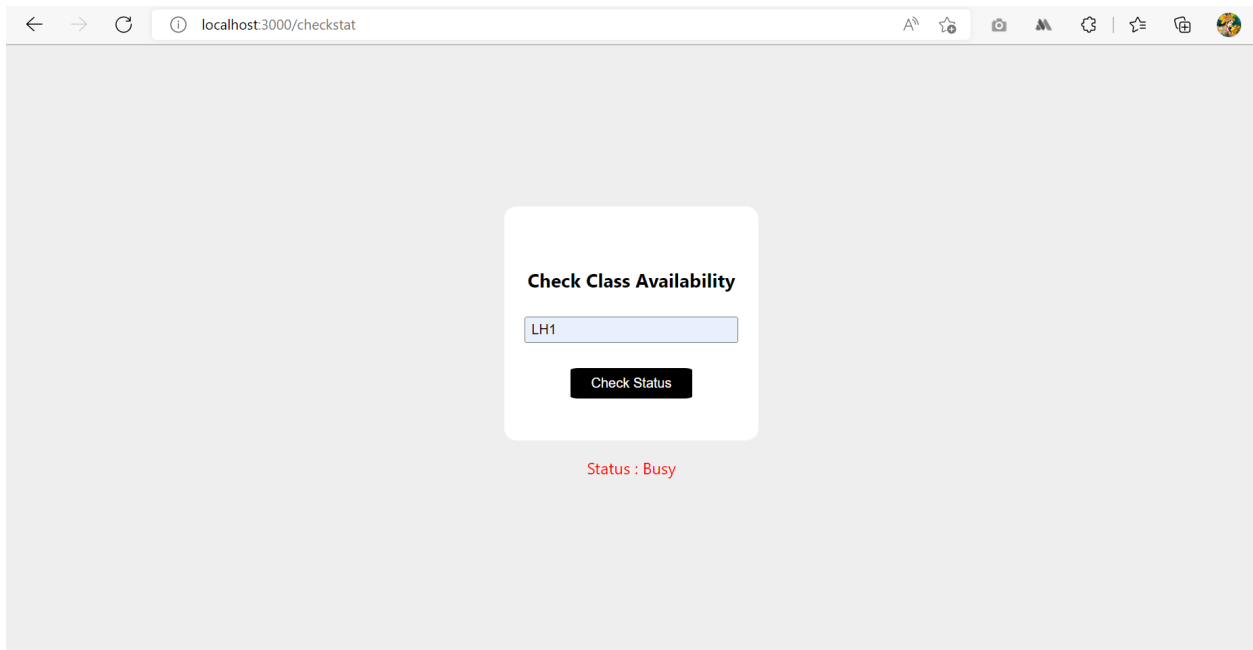
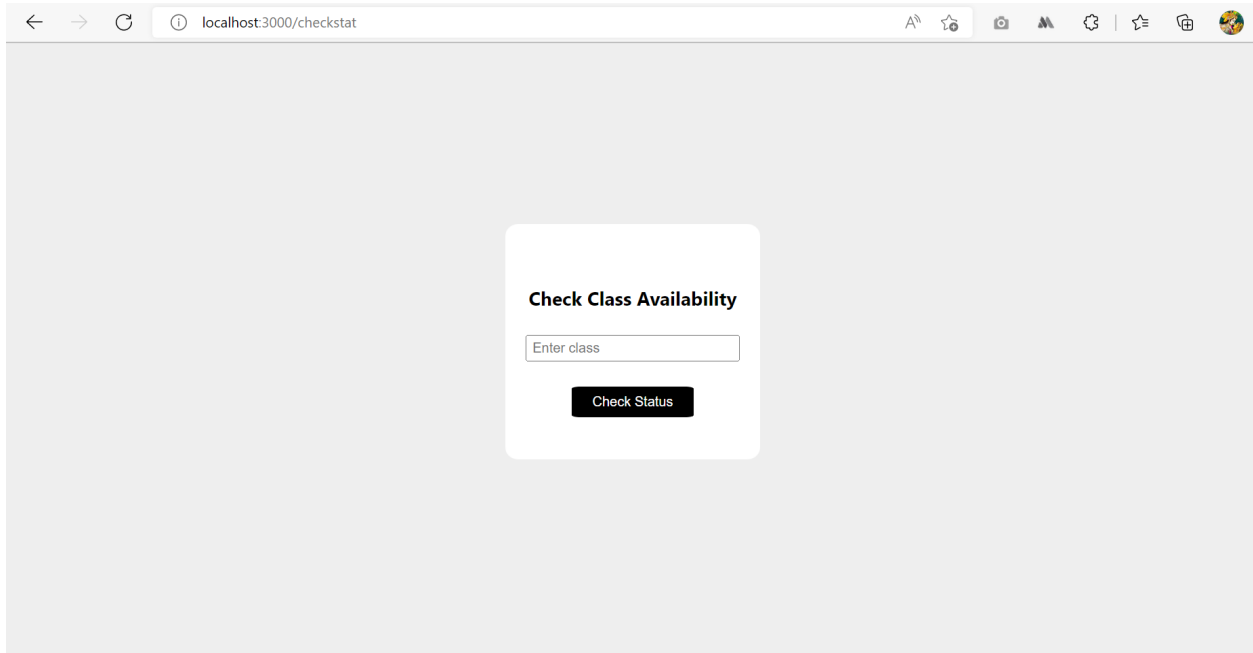
FRONT END OUTPUT FILES:

The image displays two screenshots of a web application running on a local host.

The top screenshot shows the login page at `localhost:3000/login`. It features a white login card centered on a light gray background. The card contains a circular profile icon placeholder, a text input field for "Username", another text input field for "Password", and a black "Login" button.

The bottom screenshot shows the home page at `localhost:3000/home`. It has a dark gray navigation bar with five links: [Home](#), [Check Availability](#), [View TT](#), [Report Issue](#), and [Logout](#). Below the navigation bar, the page title "Resource Reservation System" is displayed in a large, bold, black font. Underneath the title, there is a paragraph of text:

The classrooms in our department, in addition to being a venue for conducting regular classes, can also serve as examination halls and spaces for club meetings. We have decided to investigate the use of a classroom allocation system System to facilitate hassle free, user friendly and efficient allotment of classroom resources. This system would be used by members who may be students or professors of that University to check the availability of the classrooms, seminar halls, examination centers and labs. The purpose of this document is to analyze and elaborate on the high-level needs and features of the RRS. It focuses on the capabilities and facilities provided by an RRS. The details of what all are the needs of the RRS and if it fulfills these needs are detailed in the use-case and supplementary specifications. This system can be extended to all departments, schools and workspaces with necessary modifications.



[Home](#)[Check Availability](#)[View TT](#)[Logout](#)

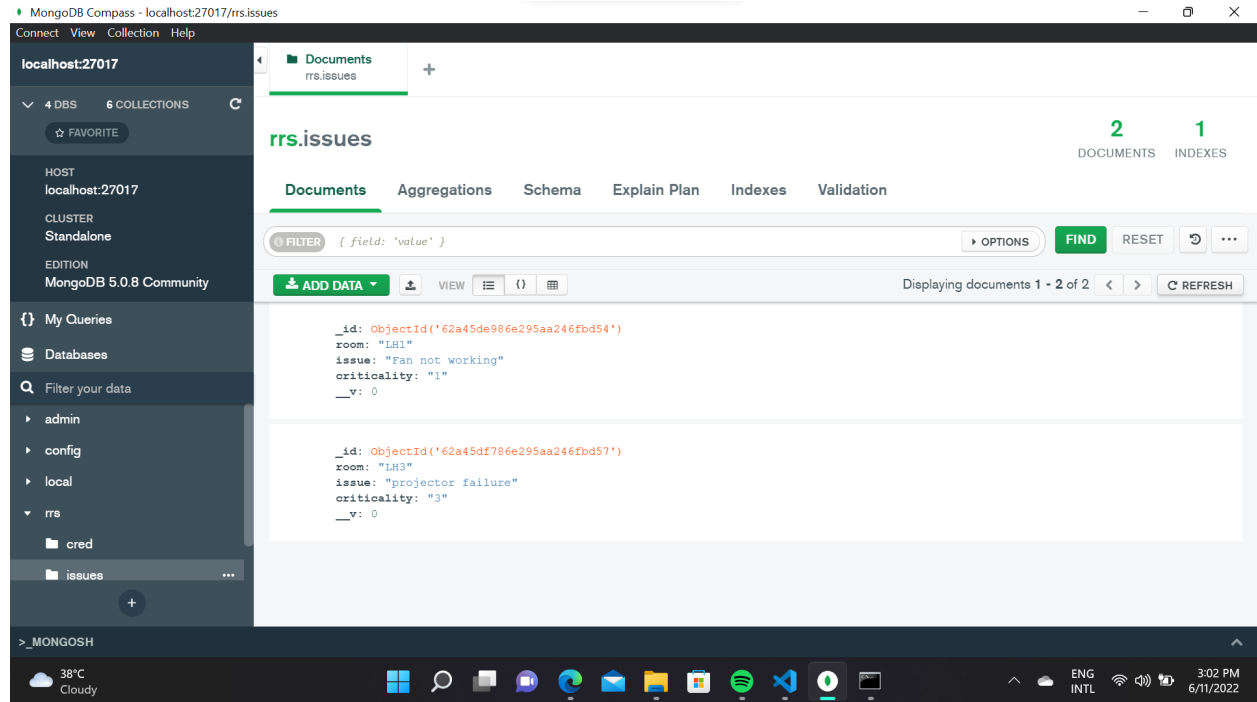
Time Table

Day	Hour 1	Hour 2	Hour 3	Hour 4	Hour 5
Monday	IP	OOAD	FDS	ML	CD
Tuesday	OOAD	ML	IP	CD	FDS
Wednesday	IP	CD	OOAD	ML	FDS
Thursday	CD	ML	OOAD	IP	FDS
Friday	ML	CD	FDS	OOAD	IP

DATABASE OUTPUTS:

Solve issues module

Before solving issue:



The screenshot displays the MongoDB Compass application interface. The left sidebar shows the database structure for 'localhost:27017', including 4 databases and 6 collections. The 'rrs' database is expanded, showing a collection named 'issues'. The main panel displays the 'rrs.issues' collection with 2 documents and 1 index. The 'Documents' tab is active, showing a list of documents. The first document has an issue of 'Fan not working' with a criticality of '1'. The second document has an issue of 'projector failure' with a criticality of '3'. The interface includes a filter bar, a search bar, and a refresh button. The bottom status bar shows the system temperature as 38°C and the time as 3:02 PM on 6/11/2022.

MongoDB Compass - localhost:27017/rrs.issues

Connect View Collection Help

localhost:27017

4 DBS 6 COLLECTIONS

☆ FAVORITE

HOST
localhost:27017

CLUSTER
Standalone

EDITION
MongoDB 5.0.8 Community

My Queries

Databases

Filter your data

admin

config

local

rrs

cred

issues

Documents
rrs.issues

rrs.issues

2 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { field: 'value' }

ADD DATA VIEW

Displaying documents 1 - 2 of 2

REFRESH

```
{ "_id": ObjectId("62a45de986e295aa246fbd54"),  
  "room": "LH1",  
  "issue": "Fan not working",  
  "criticality": "1",  
  "_v": 0 }  
  
{ "_id": ObjectId("62a45df786e295aa246fbd57"),  
  "room": "LH3",  
  "issue": "projector failure",  
  "criticality": "3",  
  "_v": 0 }
```

> _MONGOSH

38°C Cloudy

ENG INTL

3:02 PM 6/11/2022

After solving issue:

MongoDB Compass - localhost:27017/rrs.issues

Connect View Collection Help

localhost:27017

4 DBS 6 COLLECTIONS

☆ FAVORITE

HOST
localhost:27017

CLUSTER
Standalone

EDITION
MongoDB 5.0.8 Community

My Queries

Databases

Filter your data

- admin
- config
- local
- rrs
 - cred
 - issues

+ _MONGOSH

Documents
rrs.issues

2 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { field: 'value' }

ADD DATA VIEW

Displaying documents 1 - 1 of 1

```
{
  "_id": ObjectId('62a45df786e295aa246fbd57'),
  "room": "LH3",
  "issue": "projector failure",
  "criticality": "3",
  "_v": 0
}
```

38°C Cloudy

3:04 PM 6/11/2022

DATE:23.05.2022

ASSIGNMENT 8
TEST CASES AND TEST PLANS

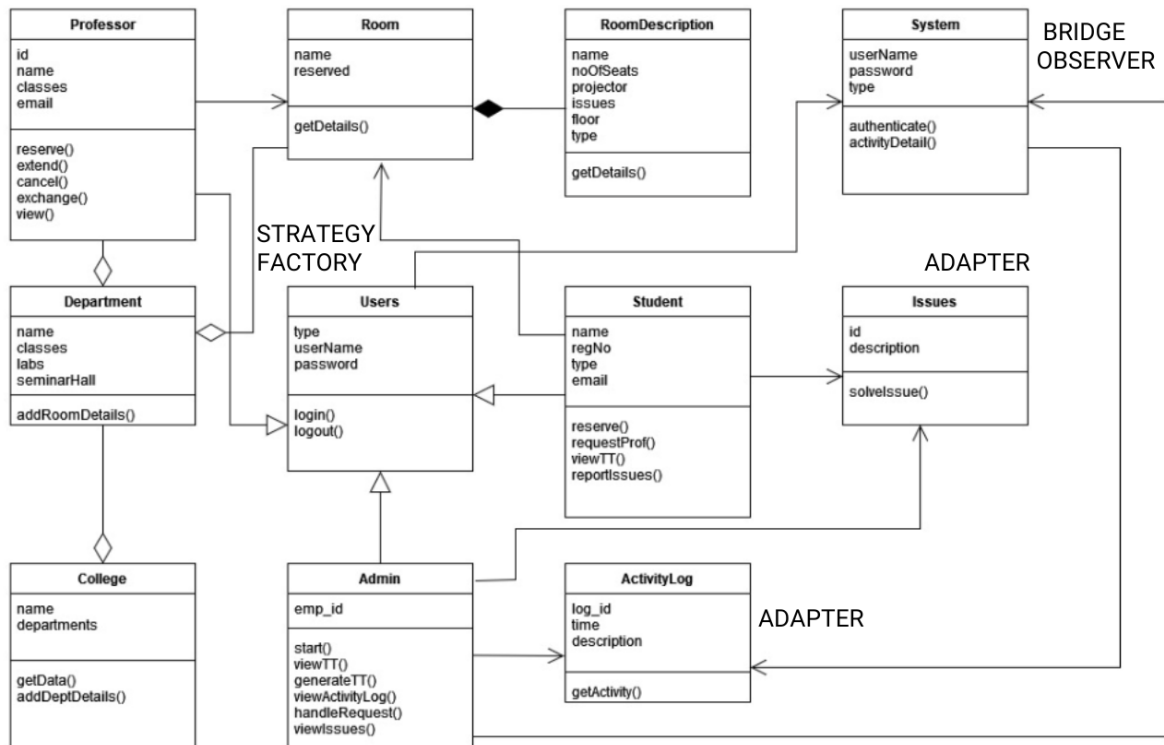
Sl No	Test Scenario	Testing steps	Test data	Expected output	Actual output	pass/fail
1	User authentication	Enter the username and password and submit the login form	Username: ST100 For a student	Shows the student page	Shows the student page	pass
			Username: F100 For a faculty	Shows faculty page	Shows faculty page	pass
			Username: ST101	No user found	Shows student page	fail
			Username: F101 Faculty	No user found	No user found	pass
2	Reserve classroom	Enter the required details to book a classroom and submit the form	Invalid classroom details for prof./club head	Error, redirected	Error, redirected	pass
			Classroom not available with dynamic allocation for prof./club head	Classroom not found	Classroom not found	pass
			Reason not valid for club head	Send alert stating non availability	No mail sent	fail
3	Report issues	The class rep fills the issues form and submits it	Invalid issues reported	Invalid, kindly contact admin	Successfully submitted	fail

4	Approve or deny requests	The admin views the list of requests from the activity log for approval	Invalid requests by prof or club head	Rejection mail is sent	No mail sent	fail
			Valid requests by prof/club head	Approved if available	Approved if available	pass
5	View Timetable	The view time table option is selected from the menu	For a student, request to view multiple class timetables	Can view only your timetable	Can view only your timetable	pass
			For a faculty, viewing other classes timetables	Can view only your classes handled	Can view only your classes handled	pass
			For club head, viewing all timetables	Can only view overall allotment	Can view only your timetable	fail
			For rep, viewing all timetables	Can view only your class timetable	Can view only your class timetable	pass

DATE:30.05.2022

ASSIGNMENT 9
APPLYING DESIGN PATTERNS

Class diagram



The following design patterns have been obtained from the existing class diagram

1) STRUCTURAL PATTERNS:

a) Adapter patterns:

Issues and activity log act as the adapter classes, providing compatibility between the input form and the local file system.

b) Bridge:

Login functionality in the System class abstracts the authentication that takes place behind the scene for each type user. In this way, abstraction and implementation are separated. The User is only concerned with logging into the system and is unaware/ in the dark about the authorisation.

2) **CREATIONAL PATTERNS:**

a) **Factory patterns:**

The System class initializes the base User class which is responsible for creating the different types of system users. Thus, User class implements factory pattern creating Admin, Faculty and Student class.

3) **BEHAVIORAL PATTERNS:**

a) **Strategy pattern:**

View timetable based on the user who wishes to view the timetable details, different strategies/algorithms are used to render the same.

Eg: student is authorized to only view the timetable of his/her class whereas admin can view those of every single class in a department.

b) **Observer pattern:**

In checking available functionality for classroom/lab/seminar hall, the submit button acts as the observer which on clicking retrieves the information stored on the database and displays the current status inside the container in the webpage.

DATE:06.06.2022

ASSIGNMENT 10
TESTING AFTER REFINEMENT- REFACTORING

REUSABLE FUNCTIONS

- Check availability function scans the classrooms as requested and returns the status of the same. The same applies to seminar halls and laboratories. Further, it can be expanded to other departments as well by simply modifying the total number of allocatable resources.
- Login function checks the username and password entered by the user with the credentials already stored in the database and performs the respective action. This is the same no matter who is trying to access the system and hence, reused for any type of user login.

REUSABLE COMPONENTS

The User class is a parent class from which the various users of the system like Admin, Student and Professor inherit from. This class can be implemented by other departments and/or colleges.

APPLICATIONS WHICH CAN MAKE USE OF THE COMPONENTS AND FUNCTIONS

1. Check availability function can be used in a hotel/hostel management system to view the status of booking for various rooms.
2. Reservation of resources can be used in hotel/hostel management systems to book resources in advance.
3. View timetable function can be used in any scenario where a schedule or plan is integral to the working.
4. Report issues module can be used in any school/college/work setup to report maintenance and logistic issues.
5. User classes can be reused in any school/college application to store and retrieve the credentials and personal data of the user.