Automatic Timetable Generator

**Project Advisor**

Muhammad Fahad

**Project Manager**

Fahad Maqbool

**Project Team**

Samia Amin

{16-uglc-404}

Azka Abubakar

{16-uglc-401}

**Submission Date**

1. april-2020

**Table of Contents**

[CHAPTER 1 4](#_Toc25499)

[Abstract: 4](#_Toc15897)

[Objectives: 4](#_Toc12877)

[1. Course features 4](#_Toc21400)

[2. Staff features 4](#_Toc6962)

[3. Time Table Allocation features 4](#_Toc4712)

[4. Subject features 4](#_Toc4458)

[5. Room features 5](#_Toc14951)

[6. Class Allocation 5](#_Toc4503)

[7. Time period features 5](#_Toc2308)

[Description: 5](#_Toc4466)

[2: Background and Justification 6](#_Toc15093)

[3: Project Scope 7](#_Toc2362)

[CHAPTER 2 8](#_Toc17519)

[Document Information 8](#_Toc19831)

[1. Introduction 9](#_Toc15453)

[2: Overall System Description 10](#_Toc25564)

[Soft Constraints may include following: 11](#_Toc6709)

[3. External Interface Requirements 12](#_Toc5212)

[4. Functional requirements for the Admin: 12](#_Toc9028)

[5. Non-functional Requirements: 12](#_Toc16290)

[6. Business Rules 13](#_Toc3001)

[7. Assumptions and Dependencies 13](#_Toc3726)

[CHAPTER 3 14](#_Toc12472)

[1: Introduction 14](#_Toc27241)

[2: Design Considerations 15](#_Toc15411)

[3: System Architecture 15](#_Toc27631)

[ADMIN 17](#_Toc18526)

[STAFF: 19](#_Toc8026)

[STUDENT: 20](#_Toc2042)

[Insertion Use Case Diagram : 24](#_Toc11795)

[Login Use Case Diagram: 25](#_Toc25037)

[Sign up Use Case Diagram: 26](#_Toc17086)

[Add Faculty Use Case Diagram: 27](#_Toc8447)

[Add Room Use Case Diagram: 27](#_Toc25263)

[Add Subject Use Case Diagram: 28](#_Toc10549)

[Add semester Use Case Diagram: 30](#_Toc4115)

[Add Time slot Use Case Diagram: 30](#_Toc21658)

[Update Subject Use Case Diagram: 31](#_Toc25166)

[Update Time Sot Use Case Diagram: 32](#_Toc4932)

[Update Room Use Case: 33](#_Toc28255)

[Update Semester Sequence Diagram: 37](#_Toc24202)

[Update Time Sots Sequence Diagram: 38](#_Toc26438)

[Update Room Sequence Diagram: 39](#_Toc26698)

[Login Sequence Diagram: 39](#_Toc24260)

[40](#_Toc27959)

[Timetable View Sequence Diagram: 40](#_Toc4331)

[40](#_Toc2715)

[Logout Sequence Diagram: 40](#_Toc14677)

[3.3: Design Strategies 41](#_Toc15161)

[4: Detailed System Design 42](#_Toc9986)

[5: References 42](#_Toc31797)

**CHAPTER 1**

**Abstract:**

Most departments have a number of different courses and each course has a number of subjects. Now there are bounded faculties, each faculty teaching more than one subjects. So now the time table require to schedule the faculty at provided time slots in a manner that their timings do not overlap and the time table schedule makes best use of all faculty subject requirements. We use a genetic algorithm for many purpose like To solve this problem, a genetic algorithm keep up a population of chromosomes, each of which represents a possible solution (timetable). In every generation, a new population of chromosomes is created using bits and parts of the fittest of the old generation. The main tasks of fixing a genetic algorithm to resolve a problem are encoding the solution as chromosomes, generating a fitness evaluation function, choosing genetic operators and run parameters.In general a university course timetabling problem contain in finding the exact time allocation within a limited time period (e.g. a week), of a number of events (courses-lectures) and also assign to them a number of resources (a teacher, a room, etc.) in such a way that a number of constraints are satisfied. Usually courses are arranged in a number of semesters . The constraints that have to be satisfied by a timetable are usually separated into two categories: hard constraints and soft ones.

**Objectives:**

1. Course features
2. Staff features
3. Time Table Allocation features
4. Subject features
5. Room features
6. Class Allocation
7. Time period features

**Description:**

1. **Course:**

In this system both the entrance and view of the course features can be complete. If the entrance should be done, then the features will be checked. After that the features will be added in the database and a message will be displayed to confirm the entry.

1. **Staff Detail:**

In this system both the entry and view of the staff features can be done. If the entry should be done, then the features must be entered and checked for duplicate. Then the features will be added in the database and a message will be displayed to confirm the entry.

1. **Time Table Allocation:**

In the Time Table Allocation features the staff, subject of the Corresponding staff will be entered, then the entered staff Name will be checked against the database. If the match is found, then the Time Table Allocation features of the corresponding staff will be generated and stored in database, Else an error message will be displayed.

1. **Subject features:**

The subject features master includes subject name, id and the staff handling the subject will be entered and stored in database. In time table generation the features will be retrieved and used as per the requirement.

1. **Room features:**

Enter the room features like room id,type of the room(lab or class or multimedia class),No of rooms for theory classes and for practical classes. Make sure the suitable room is assigned to a suitable lecture.

1. **Class Allocation:**

In the Class Allocation features the staff, subject of the Corresponding staff will be entered, Corresponding class room will be entered, then the entered staff Name will be checked against the database. If the match is found, then the class Allocation features of the corresponding

# 2: Background and Justification

The system is designed to be more efficient than the actual manual system. It call on all base performance that are now carried out manually, such as the forms transactions and reports which increase advantage. Description:

* Most and schools have a number of different courses and each course has ‘n’ number of subjects.
* Now there are bounded faculties, and each faculty might be teaching more than one subjects.
* So now the time table needed to schedule all the faculty at provided time slots in such a way that their timings do not interviewing the time table schedule will make the best use of all faculty subject demands.
* We use a generate algorithm for this purpose.
* In our Timetable Generation algorithm we generate use to to utilize a timetable object.
* This object comprises of Classroom objects and the timetable for them likewise a fitness score for the timetable.
* Fitness score relates to the quantity of crashes the timetable has regarding alternate calendars for all the different classes.
* Classroom object comprises of week objects. Week objects comprise of Days, Days comprises of Timeslots.
* Timeslot has an address in which a subject, student gathering going to that particular address and educator showing to the subject it is related will be shown.
* Also further on discussing, we have utilized composite configuration (design), which make it well extendable numerous obligations.
* In every obligation class the condition as determined in our inquiry is now checked between both the timetable objects. On the off chance that condition is fulfilled, there is a crash is available then score is augmented by one.

# 3: Project Scope

Automatic Timetable manger is a Java based software used to

generate timetable automatically.

* Will help you to manage all the periods spontaneously and also will be helpful for faculty who will get timetable in their phone as a notification.
* It will also manage timetable when any teacher is absent late coming or early going.
* Proposed system will help to generate it spontaneously also helps save the time.
* There is no need for Faculty to be worry about their timetable any more.
* It is a comprehensive timetable management solution for Colleges which help to complete the challenges in current system.

**CHAPTER 2**

Document Information

| Category | Information |
| --- | --- |
| Customer | UOS |
| Project | <Automatic Timetable Generator> |
| Document | Requirement Specifications |
| Document Version | 1.0 |
| Identifier | PGBH01-2003-RS |
| Status | Draft |
| Author(s) | <Samia Amin, Azka Abubakar > |
| Approver(s) | PM |
| Issue Date | Nov.18, 2019 |
| Document Location |  |
| Distribution | Advisor  PM  Project Office |

# Introduction

Automatic Timetable Generator is a Java based software used to create

Timetable spontaneously. Currently timetable is managed manually. It will help to

maintain all the periods automatically and also will be helpful for faculty to get

timetable in their phone by using application. It will also maintain timetable when

any teacher is absent , late coming or early going or may not come. Maximum and minimum work pressure for a Faculty for a day, week and month will be specified for the efficient generation of timetable. By using Automatic timetable generator software users can apply for leave by providing leave required date, reason and also with substitute faculty. When selecting a faculty as substitute it manage to view timetable of that faculty for ensure that the faculty is free at that particular period. Substitute can approve or reject request.

Principal can approve / reject request. It is a comprehensive timetable maintenance solutions for Colleges which help to complete the challenges in manually setting the timetable. By using this software it will be very easy or safe for faculty to get timetable in their phones and laptops.

## 1.1 Purpose of Document

The main purpose of this document is to provide the whole description about the project so who will read this document can understand the overall system of the project where the purpose of this is to encapsulate the working and functionality of this project.

## 1.2 Project Overview

Automatic Timetable Generator is a Java based software used to create

timetable spontaneously. Currently timetable is maintain manually. It will help to

manage all the periods spontaneously and also will be helpful for faculty to get

timetable in their phone or laptops by using application. Maximum and minimum work load for a Faculty for a day, week and month will be specified for the efficient generation of timetable. By using this software users can apply for leave by giving leave required date, reason and also with substitute faculty. When selecting a faculty as substitute it order to view timetable of that faculty for ensure that the faculty is free at that particular period. Substitute can approve or reject request. Principal can also view the request send by faculty and can also view substitute response. Principal can approve / reject request.It is a comprehensive timetable maintenance solutions for Colleges which help to overcome the challenges in manually setting the timetable.

## 1.3 Scope

1: Faculty need not worry for time clashes.

2: Authority now does not need to perform permutation and combination

3: Authority can concentrate on other things rather than wasting their time on preparing Time-Table.

4: Gives accurate information

5: Simplifies the manual work

6: It minimizes the documentation related work

7: Provides up to date information

8: Friendly Environment by providing warning messages

# 2: Overall System Description

Trying to create a software which manage to create Timetable for an

Institution spontaneously. By knowing at the existing system we can understand

that timetable generation is done manually. Manually maintain the timetable when

any of the faculty is absent, and this is the big challenge for project Generator that managing the timetable automatically when any of the faculty is absent.As we know all institutions have its own timetable, managing and maintaining these will not be difficult. Considering workload with this scheduling will make it more complex. As mentioned , when Timetable generation is being complete, it should assume the maximum and minimum workload that is in a college. In those cases timetable generation will become more difficult. Also , it is a time consuming work.

## 2.1 User characteristics

The users of this system are:

* Admin

## 2.2 Operating environment

This is a Web base environment which will be used to create automatic Time table. It does not require an internet connection.

**Platforms:**

Web Base System

## 2.3 System Constraints

There are two type of constraints

* Hard constraints
* Soft constraints

**Hard Constraints may include the following:**

1: No resource (teacher or room) may be assigned to different events at the same time.

2: Events of the same semester should not be given at the same time slot when both events are of type “theory” or when one event is “theory” and one event is “lab practice”. Same-semester events can run equally only if they are both of type “lab practice”, as for each course 4..6 “lab practice” classes are scheduled within the week, each attended by a different group of students.

3: There is a maximum number of time periods per day (13), that may not be exceeded.

4: One lecture may be held in a room belonging to a certain set of valid rooms for the lecture.

5: Each room may have its own schedule .

6: Each lecture may be given to a teacher that belongs to a specific set of teachers that can give the lecture.

7: Some lectures must be rigidly given to specific teachers.

8: A class (student) must have only one lecture in a given time slot.

**Soft Constraints may include following**:

1: Every teacher has his/her own availability schedule or submits a plan with desirable time periods that suits him/her best.

2: Every teacher has a minimum and a maximum limit of weekly work-hours.

3: Minimize the travel time of teachers and students between rooms within the campus.

4: Minimize the time gaps within the schedule of each teacher.

5: Minimize the time gaps within the schedule of each room .

6: Minimize the time gaps within the schedule of students

# External Interface Requirements

System requirements are what is necessary for users to install the application in their systems and be used hopefully without any difficulties. The aim of this to help the users make sure they have all required tools and equipment. With analysis there are some objectives for hardware, software and communication.

## Hardware Interfaces

* Core i3 Processor Based Computer
* 1 GB-RAM
* 50 GB Hard Disk

## 3.2 Software Interfaces

* Windows 7 or higher
* Visual Studio 2010.

## 4. Functional requirements for the Admin:

* Admin can log in to his/her account
* Admin can update, delete, retrieve data
* Admin can keep records of all activities on the system
* If a technical problem occurs, someone has to report them to the vendor so that the problem is fixed and that is the work of an admin.

# 

# Non-functional Requirements:

1. **Load Balancing:**

Since the system will be present only the admin logs in the amount of load on server will be limited to time period of admin use.

1. **Easy Accessibility:**

Records can be comfortable accessed and store and other information similarly

1. **User Friendly:**

The Website will be giving a very user friendly approach for all users.

1. **Efficient and reliable:**

Accessing the all secured and database on the server which will be accessible according the user requirement without any generating cost will be very efficient as compared to storing or saving the customer all the data on the spreadsheet or in physically in the record books.

1. **Easy maintenance:**

Automated College Timetable Generator Website is design as easy way. So maintenance is also easy.

# Business Rules

Admin’s Role:

* Admins can login in to the account.
* Admin s can keep records of all activities on the system.
* Admin can change there free time.
* If a technical problem occurs, someone has to report them to the vendor so that the problem is fixed and that is the work of admin.

# 7. Assumptions and Dependencies

## 7.1 Assumptions:

First assumption is that the user must be using laptop or any computer system.

The factor is not system constraint, but areas where future changes might drive changes in the requirements. The project could be affected if these assumptions are incorrect, are not shared, or changed.

## 7.2 Dependencies:

The main and most important dependencies are that the system must be

* Core i3 Processor Based Computer
* 1 GB-RAM

**CHAPTER 3**

# **1: Introduction**

**1.1 Purpose of Document**

The main purpose of this document is to provide the whole description about the project so who will read this document can understand the overall system of the project where the purpose of this is to encapsulate the working and functionality of this project.

**1.2 Project Overview**

Automatic Timetable Generator is a Java based software used to generate

timetable automatically. Currently timetable is managed manually. It will help to

manage all the periods automatically and also will be helpful for faculty to get

timetable in their phone by using application. It will also manage timetable when

any teacher is absent , late coming or early going.

# **2: Design Considerations**

 This section describes many of the issues and conflicts which need to be resolved before attempting to devise a complete design answer. In other words, this section is used to formally set the groundwork for the system design.

**2.1: Assumptions and Dependencies**

Assumptions and dependencies for the system and project are already captured in the FS document. This section should not repeat those issues. Instead it should deal with previously stated issues in the context of design, if appropriate, or bring up new issues that are only relevant to design.

**3: System Architecture**

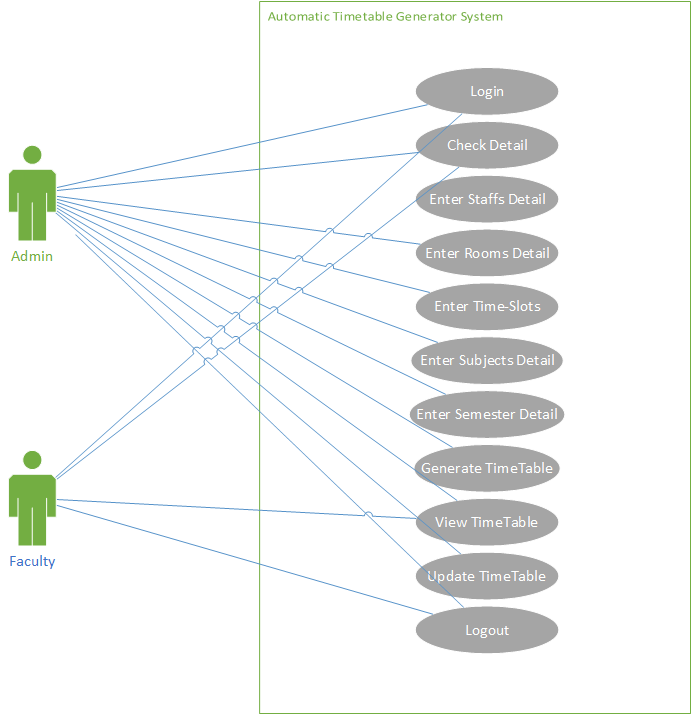
This section should provide a high-level overview of how the functionality and responsibilities of the system are partitioned and then assigned to subsystems or components. The main purpose is to achieve a general understanding of how and when the system is decomposed, and how the individual parts work together to provide the desired functionality.

**3.1 System Level Architecture**

The architecture should decompose the system at a top level in a way that provides a foundation for more detailed design work. The architecture discusses relationships and roles of system elements (subsystems, components, modules, etc.), but does not provide internal features. Areas for consideration are:

* The relationship between the elements
* Interfaces to external systems
* Major physical design issues such as where elements will execute
* Global design strategies such as error handling.

**3.1.1: System-Use Case Diagram**



**ADMIN**

|  |  |
| --- | --- |
| Use Case Name: | Automated Timetable Generator |
| Use Case ID: | ATG-1 |
| Description: | This Use case allows Admin to create an account, Enter detail of department,semester,room and staff and generate a timetable then logout. |
| Primary Actor: | Admin |
| Secondary Actor: | None |
| Precondition: | Create and Login Account |
| Post condition: | Timetable generated correctly |
| Main Flow: | 1. Admin Enter Id and password and System validate it. 2. Admin add or delete user 3. Enter department features or semester features 4. Enter staff features 5. Enter room features 6. Allocate subjects to staff 7. Allocate room to subject 8. Enter willingness of staff 9. Check willingness 10. Generate Timetable and logout 11. Use Case ends. |
| Alternative Flow: | **1a Invalid Admin Id and/or Password:**   1. System displays message”Invalid Id and password” 2. System prompts for id and password. 3. Use Case returns main flow step 1   **6a Staff detail mismatch:**   1. System display message”Mismatch staff detail”. 2. Use case returns to main flow step 4.   **7a Room detail mismatch:**  1.System display message”Mismatch Room detail”.  2.Use case returns to main flow step 5.  **9a Willingness not fulfilled:**  1.System show message”Here is some problem check for other options ”  2.System prompts for willingness.  3.Use case returns to main flow step 8. |
| Special Requirements | * Admin should have department features and semester features * Admin should have features of room to assign * Admin should have staff features to allocate them subjects |
| Exception Condition | * Network down * Department, semester features not available * Subject features not available * Staff not available * Rooms not Exist. |

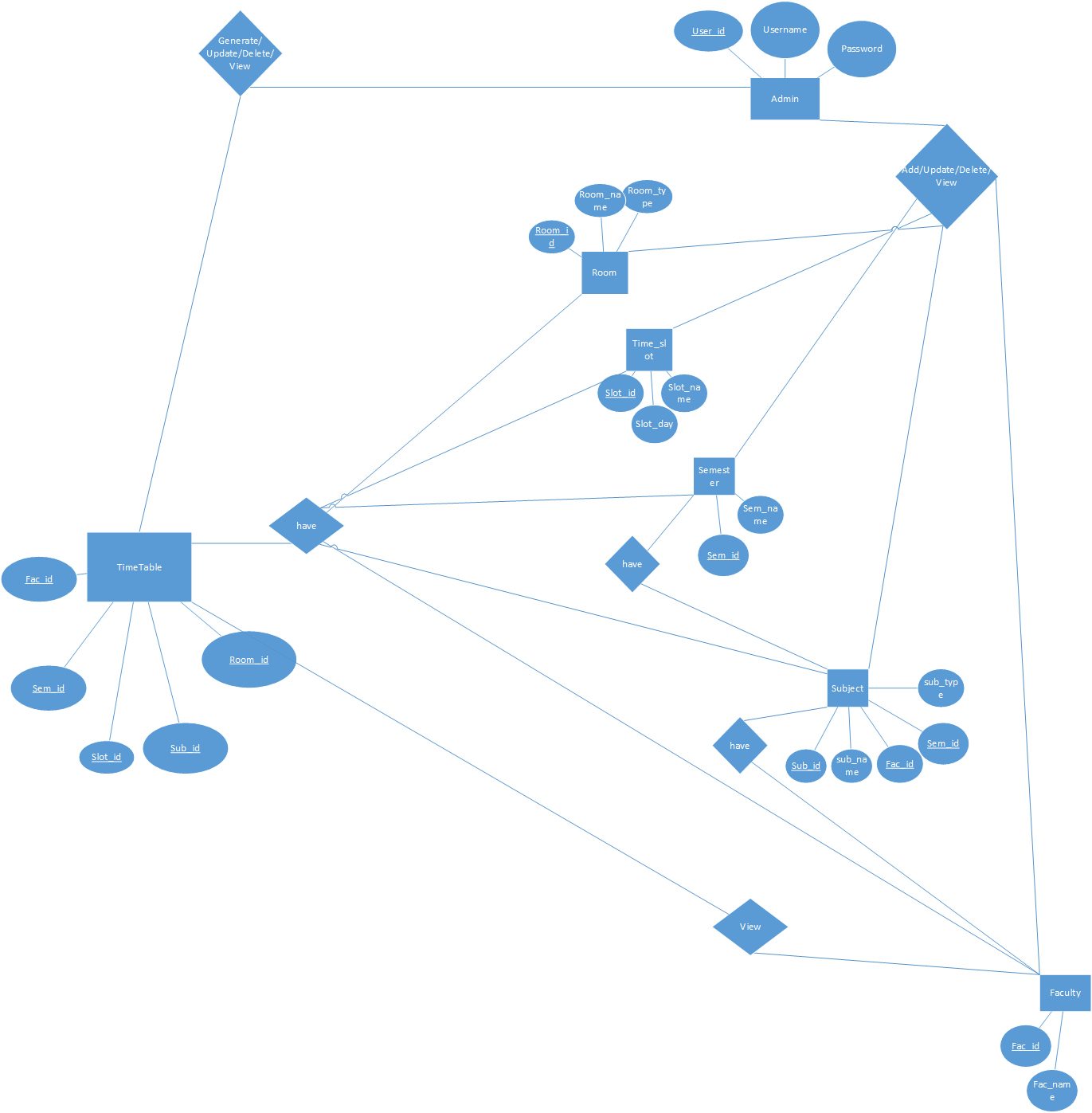
**STAFF:**

|  |  |
| --- | --- |
| Use Case Name: | Automated Timetable Generator |
| Use Case ID: | ATG-2 |
| Description: | This Use case allows Staff to create an account, Enter detail of department and semester to view their Timetable and f any semester timetable also ask for their own willingness then logout. |
| Primary Actor: | Staff |
| Secondary Actor: | None |
| Precondition: | Create and Login Account |
| Post condition: | Timetable viewed correctly. |
| Main Flow: | 1. Staff Enter Id and password and validate account 2. Enter department features or semester features 3. View their own timetable 4. View semester timetable 5. Enter their willingness 6. Check updated timetable 7. logout 8. Use Case ends. |
| Alternative Flow: | **1a Invalid Staff Id and/or Password:**   1. System displays message”Invalid Id and password” 2. System prompts for id and password. 3. Use Case returns main flow step 1   **2a Department or semester detail mismatch:**   1. System display message”Mismatch department and semester detail”. 2. Use case returns to main flow step 2.   6**a Willingness not fulfilled:**  1.System show message”Here is some problem check for other options ”  2.System prompts for willingness.  3.Use case returns to main flow step 5. |
| Special Requirements: | * Staff should have department features and semester features * Staff should have features of room to enter willing room. |
| Exception Condition: | * Network down * Department, semester features not available * Staff not Exist. |

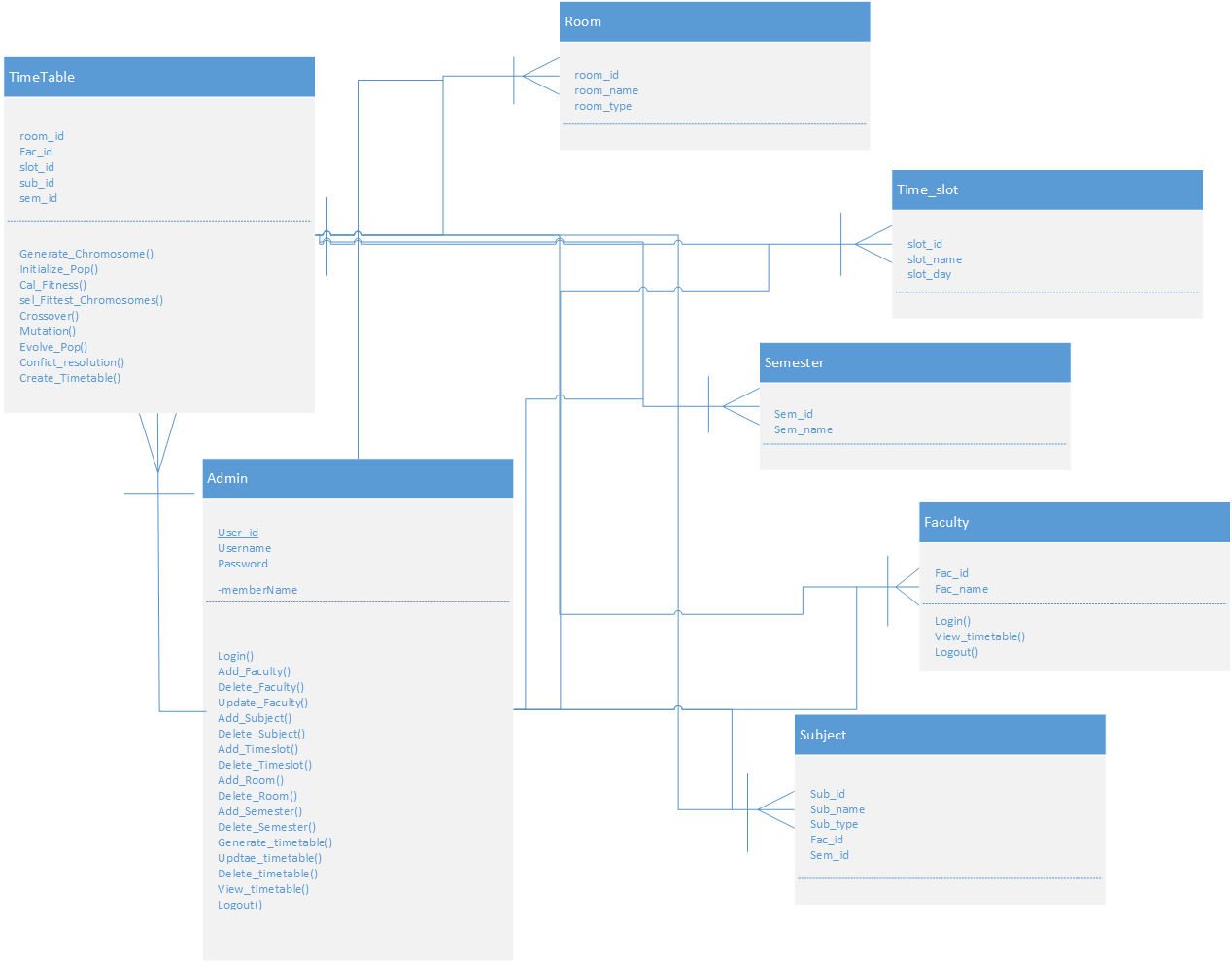
**STUDENT:**

|  |  |
| --- | --- |
| Use Case Name: | Automated Timetable Generator |
| Use Case ID: | ATG-3 |
| Description: | This Use case allows Student to create an account, Enter detail of department and semester to view timetable then logout. |
| Primary Actor: | Student |
| Secondary Actor: | None |
| Precondition: | Login Account |
| Post condition: | Timetable viewed correctly |
| Main Flow: | 1. Admin Enter Id and password and System validate it. 2. Enter department features or semester features 3. View Timetable and logout 4. Use Case ends. |
| Alternative Flow: | **1a Invalid student Id and/or Password:**   1. System displays message”Invalid Id and password” 2. System prompts for id and password. 3. Use Case returns main flow step 1   **2a Department and semester detail mismatch:**   1. System displays message”Mismatch staff detail”. 2. Use case returns to main flow step 2 |
| Special Requirements | * Student should have department features and semester features * Student should have account features. |
| Exception Condition | * Network down * Department, semester features not available * Account not exist.. |

**3.1.2: System-ER Diagram**



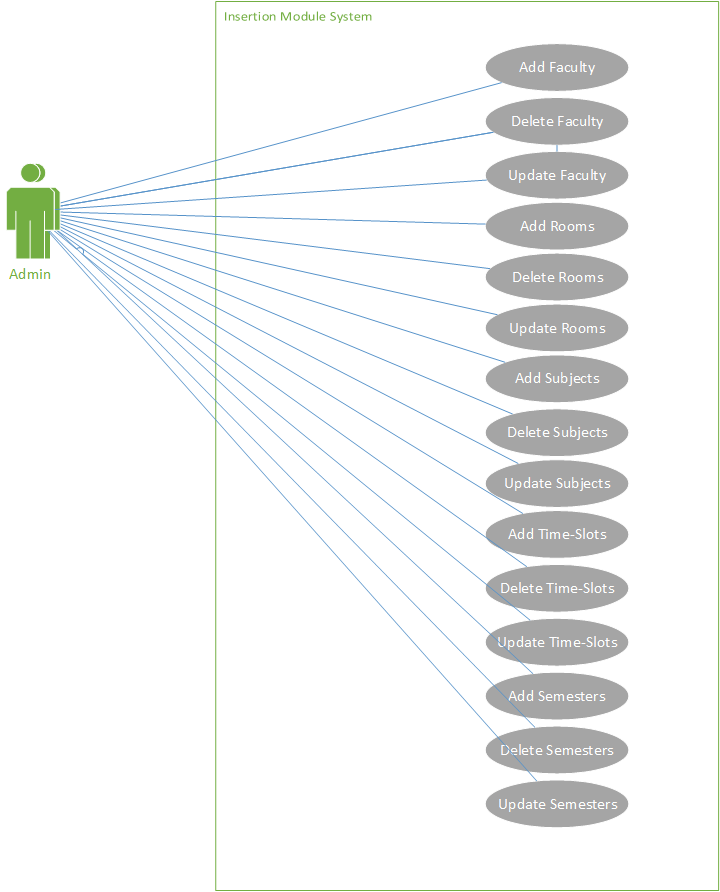
**3.1.3 Class Diagram:**

****

**3.2: Sub-System / Component / Module Level Architecture**

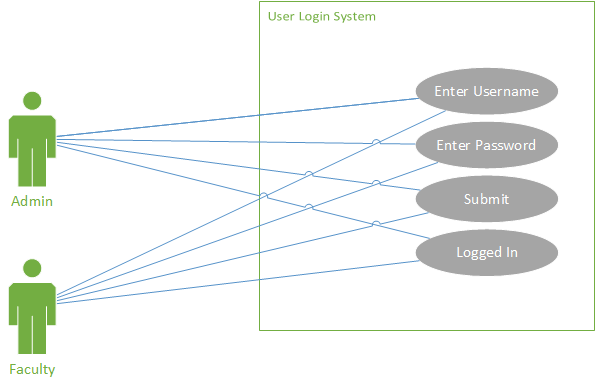
Identify all the sub components or sub modules (if any) of the already identified modules and components. Provide their diagrammatic view using appropriate detailed architecture diagram presenting how those sub systems, modules and components are further divided into sub components and sub modules and how they interact with each other.

**Insertion Use Case Diagram :**



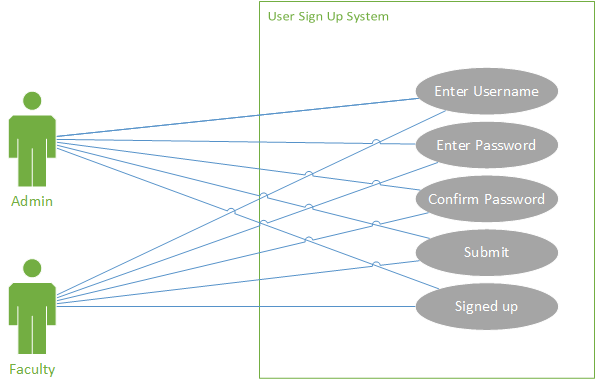
|  |  |
| --- | --- |
| Use Case Name: | Insertion module |
| Actor(s): | Admin |
| Purpose: | Update Boot , allow admin to enter data |
| Overview: | Data Inserted |

**Login Use Case Diagram:**

****

|  |  |
| --- | --- |
| Use Case Name: | Login Module |
| Actor(s): | Admin |
| Purpose: | Log in to the account |
| Overview: | This allows admin to Logged in |

**Sign up Use Case Diagram:**



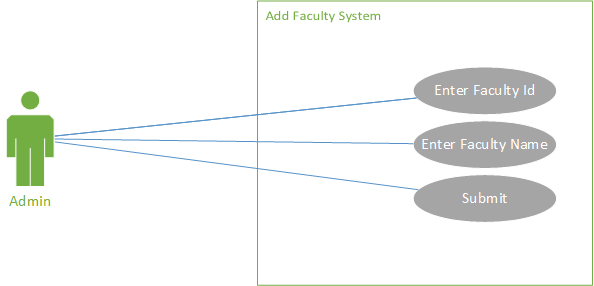
|  |  |
| --- | --- |
| Use Case Name: | Sign up modue |
| Actor(s): | Admin |
| Purpose: | Sign up in to the account |
| Overview: | This allows admin to signed up |

**[3.3 Sub-Component / Sub-Module Level Architecture (1…n)](https://docs.google.com/document/d/1RNnaoq8e2ebNcZbRRINz91HskCB1bnNAuaVNweBXaH8/edit" \l "heading=h.1ksv4uv)**

Identify all the sub components or sub modules (if any) of the already identified modules and components. Provide their diagrammatic view using appropriate detailed architecture diagram presenting how those sub systems, modules and components are further divided into sub components and sub modules and how they interact with each other.

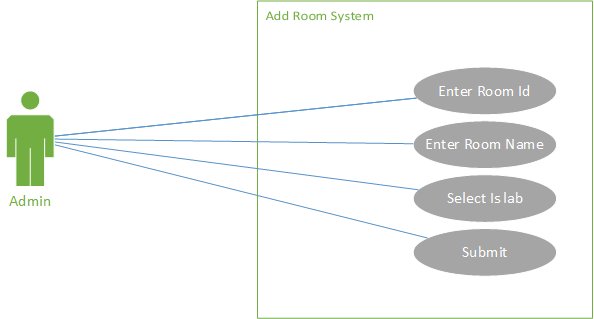
**3.3.1 Use Case Diagrams**

**Add Faculty Use Case Diagram:**



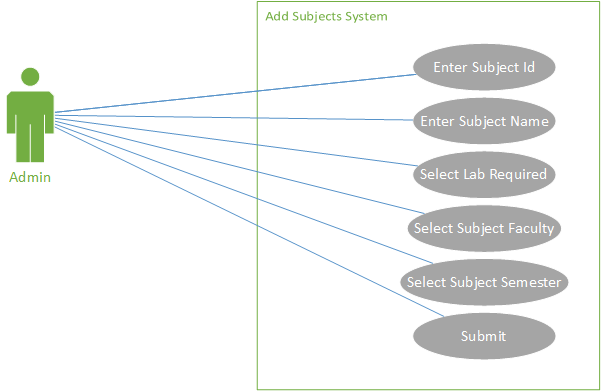
|  |  |
| --- | --- |
| Use Case Name: | Add faculty detail |
| Actor(s): | Admin |
| Purpose: | Enter faculty |
| Overview: | This allows admin to add faculty |

**Add Room Use Case Diagram:**



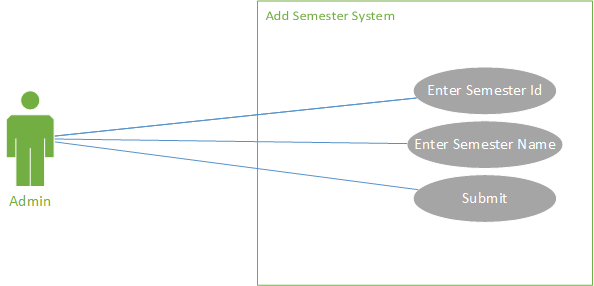
|  |  |
| --- | --- |
| Use Case Name: | Add Room detail |
| Actor(s): | Admin |
| Purpose: | Add room features |
| Overview: | This allows admin to Submit rooms |

**Add Subject Use Case Diagram:**

****

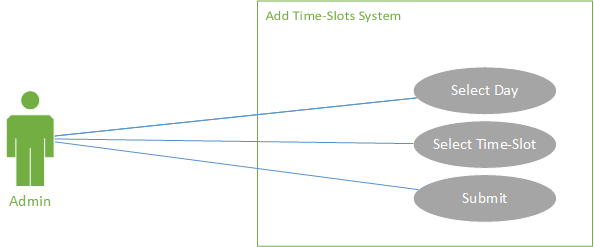
|  |  |
| --- | --- |
| Use Case Name: | Add subject detail |
| Actor(s): | Admin |
| Purpose: | Adding subject information |
| Overview: | This allows admin submit subjects. |

**Add semester Use Case Diagram:**

****

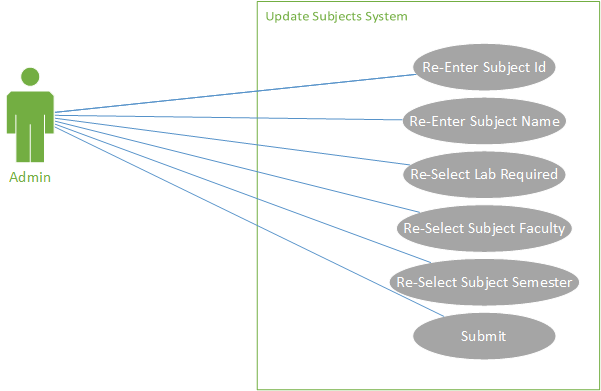
|  |  |
| --- | --- |
| Use Case Name: | Add semester features |
| Actor(s): | Admin |
| Purpose: | Enter semester information |
| Overview: | This allows admin to enter semester detail. |

**Add Time slot Use Case Diagram:**

****

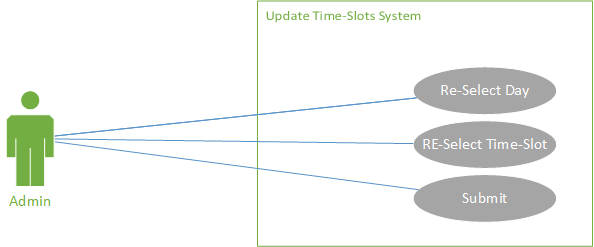
|  |  |
| --- | --- |
| Use Case Name: | Add time slot features |
| Actor(s): | Admin |
| Purpose: | Add time slots information |
| Overview: | This allows admin to enter time slot features. |

**Update Subject Use Case Diagram:**

****

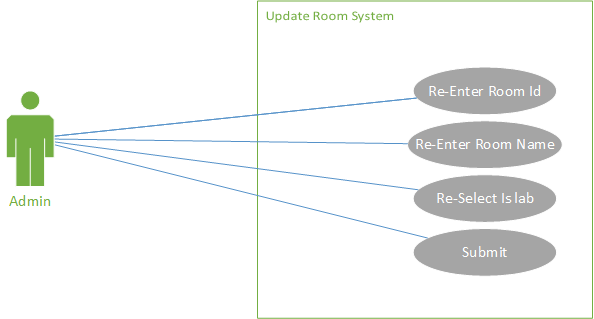
|  |  |
| --- | --- |
| Use Case Name: | Update subject detail |
| Actor(s): | Admin |
| Purpose: | Re enter subject information |
| Overview: | This allows admin to update subject features |

**Update Time Sot Use Case Diagram:**



|  |  |
| --- | --- |
| Use Case Name: | Update time slot features |
| Actor(s): | Admin |
| Purpose: | Re select time slot info |
| Overview: | This allows admin to update time slot features. |

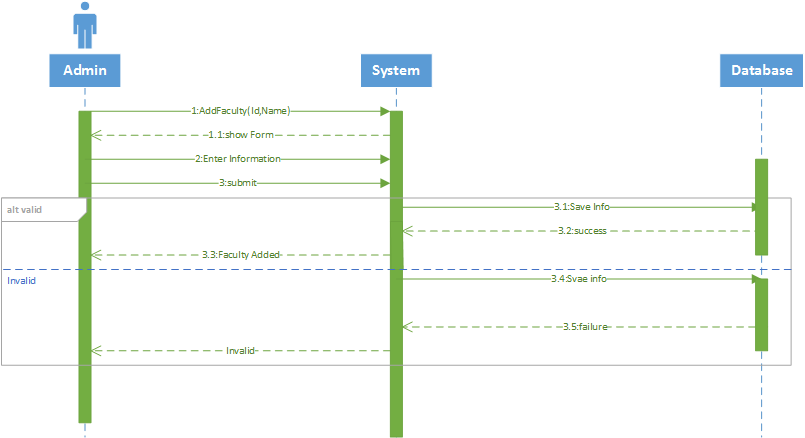
**Update Room Use Case:**



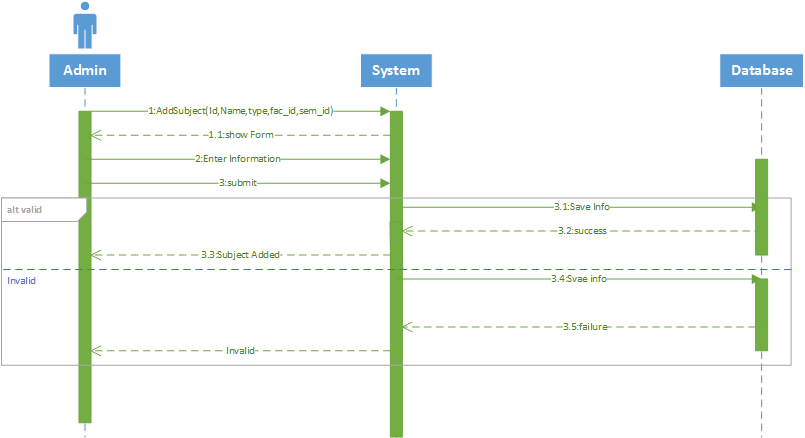
|  |  |
| --- | --- |
| Use Case Name: | Update room features |
| Actors : | Admin |
| Purpose: | Re enter rooms information |
| Overview: | This allows admin to update room features |

**3.2 Sequence Diagram**

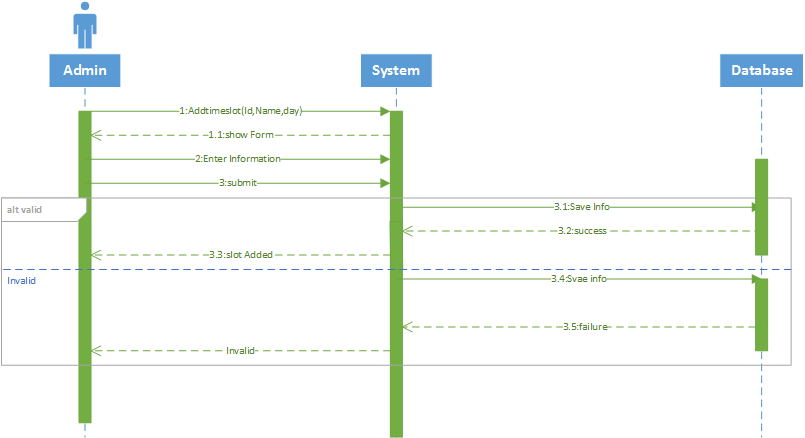
**Add Faculty Sequence Diagram:**

****

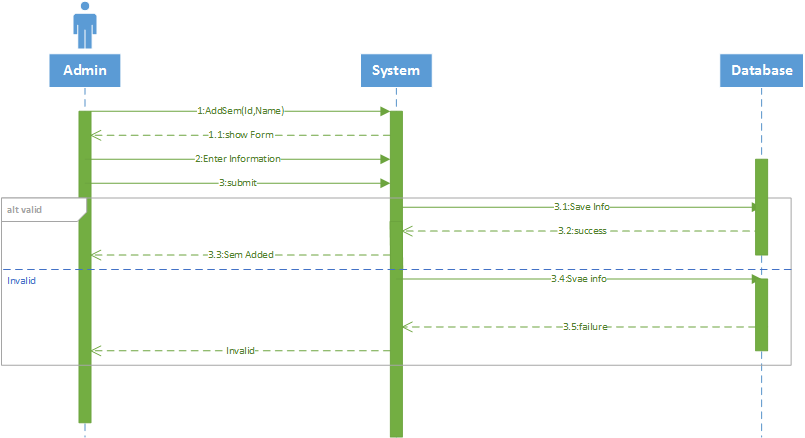
**Add subject Sequence Diagram:**



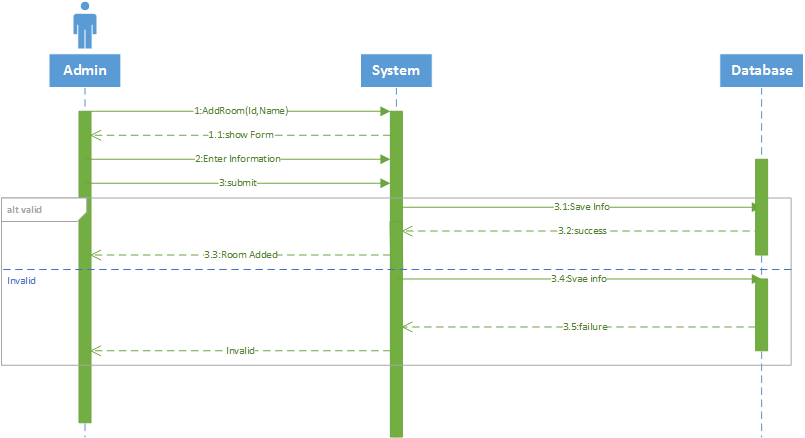
**Add Time Slot Sequence Diagram:**



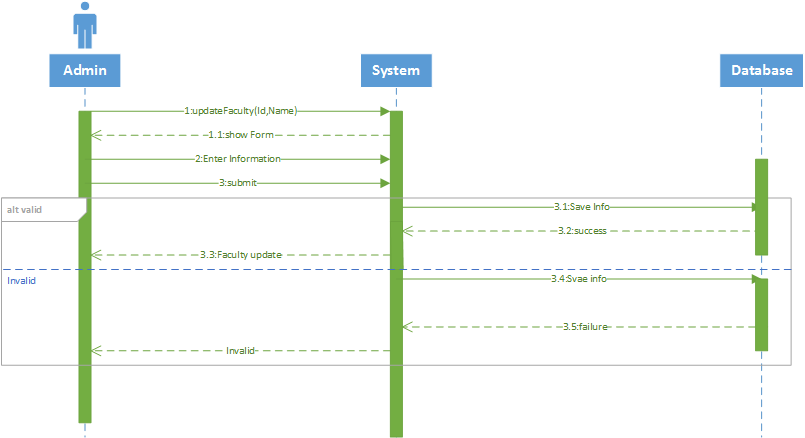
**Add Semester Sequence Diagram:**



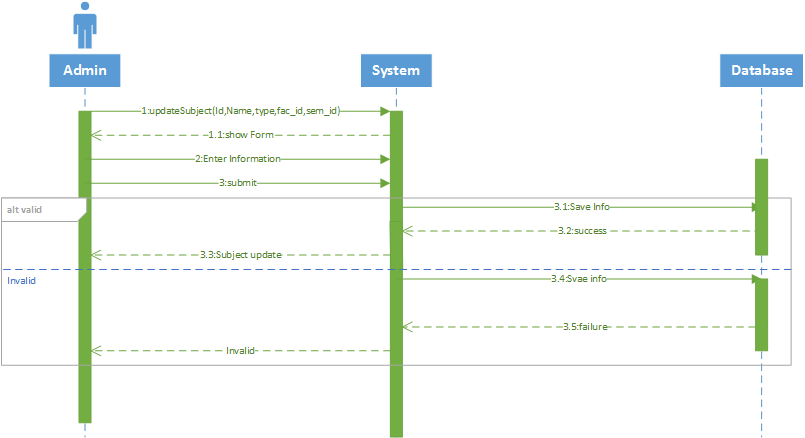
**Add Room Sequence Diagram:**



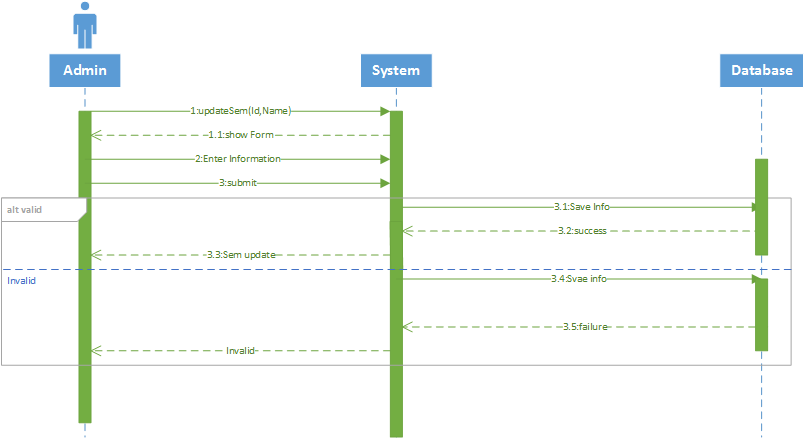
**Update Faculty Sequence Diagram:**

****

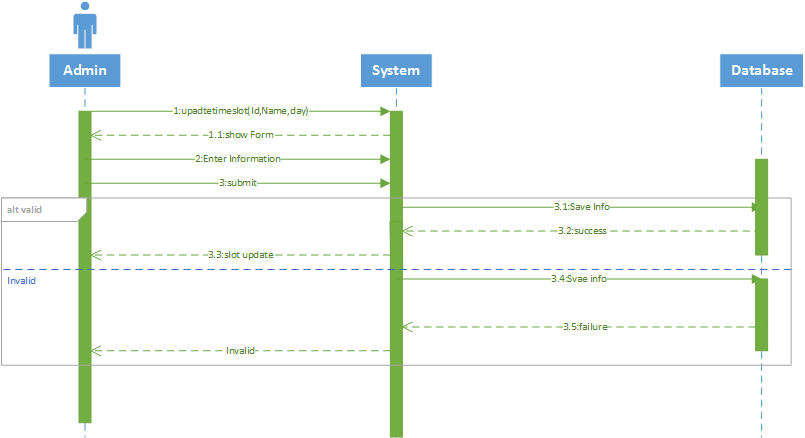
**Update Subject Sequence Diagram:**

****

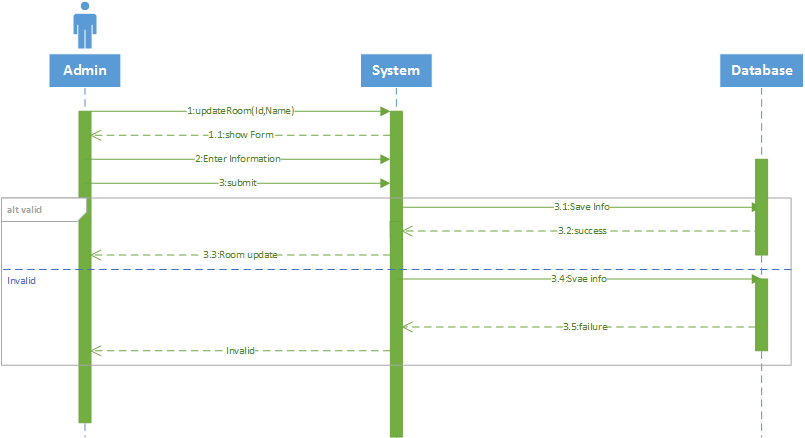
# **Update Semester Sequence Diagram:**



# **Update Time Sots Sequence Diagram:**



# **Update Room Sequence Diagram:**



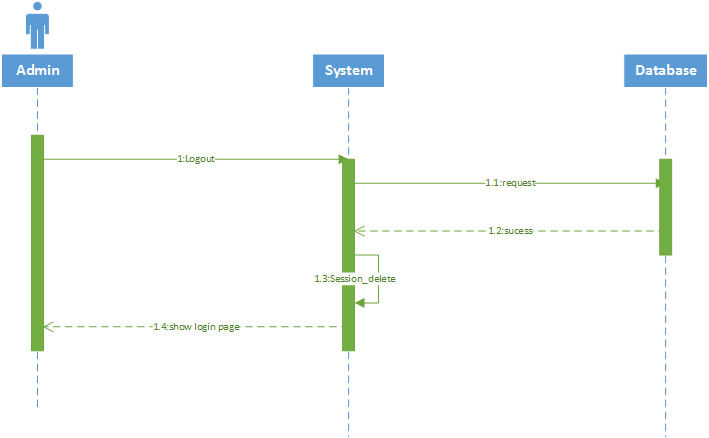
# **Login Sequence Diagram:**

# **login (1)**

# **Timetable View Sequence Diagram:**

# **Viewtimetable**

# **Logout Sequence Diagram:**

****

# **3.3: Design Strategies**

Describe the design strategies or decisions that impact the overall organization of the system and its high-level structures. This information should provide the reader with insights into the key abstractions and mechanisms used in the system architecture.

**3.2: Strategy 1…n**

For each strategy, discuss the reasoning employed (possibly referring to previously stated design goals and principles) and any trade-offs. Areas for consideration include:

* Future system extension or enhancement
* System reuse
* User interface paradigms
* Data management (storage, distribution, persistence)
* Concurrency and synchronization

**4: Detailed System Design**

A detailed design should include the following:

* Detailed class diagram along with a detailed description of all attributes, functions or methods specifying interactions between different classes/modules.
* Detailed Sequence diagram with parameter list
* Detailed use case diagrams
* ER System diagram

**5: References**

**[1]**<https://www.business2community.com/tech-gadgets/importance-information-technology-business-today-01393380>

**[2]** [https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5425771/#ref1](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5425771/" \l "ref1)

**[3]** <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1705904/>

**[4]** <https://www.slideshare.net/SamihaHaque2/online-appointment-system>

**[5]** <https://bus206.pressbooks.com/chapter/chapter-10-information-systems-development/>

**[6]** <https://www.guru99.com/agile-scrum-extreme-testing.html>

**[7]** <https://hbr.org/2013/10/the-strategy-that-will-fix-health-care>