

---

# FINAL REPORT

for

## Classroom Management System

Prepared by :

Tasnim Ferdous Anan, ID:180041108

Farhan Ishmam, ID:180041120

Sidratul Muntaha, ID:180041118

January 9, 2021

# Table of Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Purpose . . . . .	1
1.2	Product Scope . . . . .	1
1.3	Target Users . . . . .	1
<b>2</b>	<b>Problem Statement, Opportunities And Objectives:</b>	<b>3</b>
2.1	Problem Statement: . . . . .	3
2.2	Issues: . . . . .	3
2.3	Opportunities: . . . . .	4
2.4	Objectives: . . . . .	5
<b>3</b>	<b>User Requirement Analysis:</b>	<b>6</b>
3.1	METHODOLOGY . . . . .	6
3.1.1	Questionnaire . . . . .	6
3.1.2	Interview . . . . .	6
3.1.3	Representative Users . . . . .	6
3.2	RAW DATA . . . . .	7
3.3	PERSONAS AND SCENARIOS . . . . .	7
3.3.1	Persona: James . . . . .	7
3.3.2	Persona: Albert . . . . .	8
3.3.3	Persona: Stacy . . . . .	9
3.3.4	Persona: Ethan . . . . .	9
3.4	CURRENT SYSTEMS: . . . . .	10
3.4.1	Skedda . . . . .	10
3.4.2	Classroombookings . . . . .	10
3.5	REQUIREMENTS: . . . . .	11
3.5.1	Mandatory Requirements: . . . . .	11
3.5.2	Want List: . . . . .	12
3.5.3	Wish List: . . . . .	12
3.5.4	Excluded Features: . . . . .	13
<b>4</b>	<b>SRS</b>	<b>14</b>
4.1	Overall Description . . . . .	14
4.1.1	Product Perspective . . . . .	14
4.1.2	Product Functions . . . . .	14
4.1.3	User Classes and Characteristics . . . . .	15
4.1.4	Operating Environment . . . . .	16

4.1.5	Design and Implementation Constraints . . . . .	16
4.1.6	User Documentation . . . . .	16
4.1.7	Assumptions and Dependencies . . . . .	17
4.2	External Interface Requirements . . . . .	17
4.2.1	User Interfaces . . . . .	17
4.2.2	Hardware Interfaces . . . . .	19
4.2.3	Software Interfaces . . . . .	20
4.2.4	Communications Interfaces . . . . .	20
4.3	System Features . . . . .	20
4.3.1	Book Room without Booking Conflicts . . . . .	20
4.3.2	Search Rooms based on Room Details . . . . .	21
4.3.3	User Login and Authentication . . . . .	22
4.3.4	View Routine with associated Room Number . . . . .	23
4.3.5	View, Modify and Delete Booking Records . . . . .	23
4.3.6	Administration Panel . . . . .	24
4.3.7	Mobile App-based Notification System . . . . .	25
4.4	Other Nonfunctional Requirements . . . . .	26
4.4.1	Performance Requirements . . . . .	26
4.4.2	Safety Requirements . . . . .	26
4.4.3	Security Requirements . . . . .	26
4.4.4	Software Quality Attributes . . . . .	27
4.4.5	Business Rules . . . . .	27
<b>5</b>	<b>Feasibility Analysis:</b>	<b>28</b>
5.1	Technical . . . . .	28
5.2	Economical . . . . .	28
5.3	Operational . . . . .	28
<b>6</b>	<b>Project Timeline</b>	<b>30</b>
6.1	Gantt Chart . . . . .	30
6.2	PERT diagram . . . . .	30
<b>7</b>	<b>Data Flow Diagrams:</b>	<b>31</b>
7.1	Context Diagram . . . . .	31
7.2	Diagram 0 . . . . .	32
7.3	Child Diagram . . . . .	32
7.3.1	Child Diagram 1 . . . . .	32
7.3.2	Child Diagram 2 . . . . .	33
<b>8</b>	<b>UML Diagrams:</b>	<b>34</b>
8.1	Use Case Diagram . . . . .	34
8.2	Activity Diagram . . . . .	35
8.3	Sequence Diagram . . . . .	37
8.3.1	Sequence Diagram (User) . . . . .	37

8.3.2	Sequence Diagram (CR & Faculty) . . . . .	38
8.3.3	Sequence Diagram (Admin) . . . . .	39
8.4	Class Diagram . . . . .	40
8.5	ERD . . . . .	40
<b>9</b>	<b>Prototypes:</b>	<b>41</b>
<b>10</b>	<b>Conclusion and Future work:</b>	<b>44</b>
10.1	Future work: . . . . .	44
10.2	Conclusion: . . . . .	44
<b>Appendix: Glossary</b>		<b>i</b>

# 1 Introduction

## 1.1 Purpose

Our project is an online Classroom Management System to book classrooms according to the users' needs and availability. The primary purpose of our system is to minimize the issues faced when classrooms are booked manually such as booking conflicts and manual errors. Our improved system tends to mitigate problems like schedule inflexibility and improve room resource management. Through this project users will be able to book classrooms on any particular day with their desired facilities and can also unbook the allocated resources if needed. Our intended system is more efficient than the current system and would, hence, require less manpower to operate, easing the pressure on the staff. Thus, time and resources will be saved.

The system also allows us to control accessibility to different categories of user and meets necessary privacy and security requirements. Necessary information of the users and institution is also stored and moderated.

The overall purpose of this project is to provide a **complete, universally accessible, real-time, online management system for the classes of IUT**. All the necessary features required to establish such a management environment will also be provided by our system.

## 1.2 Product Scope

Our project is a real-time **MANAGEMENT INFORMATION SYSTEM (MIS)** for classrooms that can be accessed, updated and administered by multiple categories of users from anywhere. To implement the project we will use a Real Time Cloud-based Database called Firestore and a web-based Graphical User Interface for online access. The User Interface will be designed and stylized in a way to make it intuitive and user-friendly. Standard user authentication and access control will be done using Firebase Authentication. We plan to achieve our objectives by utilizing principles taught in our System Analysis and Design class along with our own dedication and hard work.

## 1.3 Target Users

This SRS document is suitable for the

- Students (primarily Class Representatives)
- Faculty

- Staff
- University Administrators

Currently, our project targets of these users of IUT specifically.

## 2 Problem Statement, Opportunities And Objectives:

### 2.1 Problem Statement:

In IUT, every section has to attend classes taken in different classrooms according to a given schedule. The scheduled classes often get canceled. Extra classes, quizzes or discussion classes are also taken. When such unscheduled classes are taken, it becomes difficult to manage a classroom for a particular group of students. The floor staff has the booking records of the classrooms. The booking records can be accessed only by him. The procedure of booking is done manually - he has to go through all the records to a room and then write that record. On the other hand, when a class is cancelled the floor staff isn't usually informed which eventually causes unavailability of classrooms. The booking procedure gets complicated when classrooms from different floors or buildings are booked. Sometimes, the same class gets booked by two groups of students and they have to wait in front of that particular classroom to resolve the issue. The unscheduled classes often get delayed or cancelled because of these booking conflicts. The teachers also have certain criteria for the rooms, for example – having a projector, being air-conditioned and so on. But these criteria are hardly met because of the inflexible booking system. The CRs also face problems in relaying the information regarding booked classes to the students.

### 2.2 Issues:

1. **Booking Conflicts:** Due to inaccessibility to booking in real-time and miscommunication, the same class gets booked by two groups of students concurrently causing them to wait in front of classrooms to resolve the conflict and/or search and find another classroom.

- *Weight – 10*

2. **Manual Errors:** Errors made by staff or CR during booking. The current booking system is manually performed by the staff and class representatives IUT. The procedure is time-consuming and prone to manual errors.

- *Weight – 10*

3. **Inflexibility:** The system is unable to find and book classes for unscheduled classes. Teachers are reluctant to take extra classes because of the inflexibility of the system.

- *Weight – 8*

4. **Not meeting requirements:** The requirements of a teacher – projectors, air-conditioned, seat capacity, number of boards and so on, aren't always ensured by the current system. Teachers often fail to deliver proper lectures due to unavailable classroom equipment.  
- *Weight – 7*
5. **Poor Resource Management:** Cancelled classes aren't reallocated and the system isn't aware of the vacant classes in real-time. As a result, some of the previously booked classrooms are left unallocated while some groups have to sit idle because of unavailable classrooms.  
- *Weight – 6*
6. **Wastage of Manpower:** A floor staff is responsible for booking classes on each floor. The booking procedure is monotonous and the staff are reluctant to take this role.  
- *Weight – 6*
7. **Time consuming:** The current system takes a considerable amount of time to search and book classrooms since the procedure is done manually.  
- *Weight – 5*
8. **Inaccessibility:** Every time the CR has to book a room, he has to look for the floor staff of that particular floor. A lot of time and effort is wasted by doing so..  
- *Weight – 4*
9. **Relaying Booking Information:** The CR has to spend a lot of time telling his fellow classmates where the unscheduled classes will take place. Social media group chats made the process a bit easier but the students still prefer a single platform to get this information from. Notifications will also help them find classrooms of unscheduled classes quickly.  
- *Weight – 3*

## 2.3 Opportunities:

1. By resolving booking conflicts, the students won't have to wait in front of classes or search for another classroom.
2. Our improved system will be free from manual errors, won't cause unwanted consequences and will, hence, be reliable.
3. The improved system will be more flexible for students and faculty members. The faculty members will be able to take unscheduled classes, discussion classes and also reschedule classes if required.
4. Teachers will be able to find classrooms meeting their requirements and can thus deliver perfect lectures.



5. Resource efficiency will be increased by making the classrooms of cancelled classes accessible to the system.
6. A floor staff won't be required to book rooms; thus saving manpower.
7. Classrooms will be booked instantly saving time for CRs.
8. The improved system will be accessed from anywhere at any time.
9. Students won't miss classes for not knowing where the classes will be held

## 2.4 Objectives:

- Develop a Real-Time system to prevent booking conflicts and reduce resource inefficiency  
*Tackled Issues - 1,5*
- Develop a Computer-based Management Information System to make the system faster, organized and free from manual errors  
*Tackled Issues - 2,6,7*
- Search and book vacant classrooms instantly on any particular date at a particular time  
*Tackled Issues - 3,7*
- Search classes based on certain criteria (capacity, AC, projector and number of boards)  
*Tackled Issues - 4*
- Access and delete previous booking records  
*Tackled Issues - 5*
- Develop a Web-based system for universal accessibility  
*Tackled Issues - 8*
- Simple, easy and intuitive user interface so that users can operate without any prior training or help from staff
- Relay the class schedule and allocated rooms to the students  
*Tackled Issues - 9*
- Create a mobile app for user notification and wider accessibility  
*Tackled Issues - 8,9*
- Automatically delete old records to keep record list clean
- Role-based user access for different categories of users
- Administrative features for moderation and institutional changes

## 3 User Requirement Analysis:

### 3.1 METHODOLOGY

The following methods were used to gather data about the given personas:

- Questionnaire via Google Form
- Interviews via Direct Messages and Phone Call

#### 3.1.1 Questionnaire

By using questionnaires as a form of data gathering method, we were able to reach 54 people in a short period of time. This method proved to be easier, quicker and more convenient. Keeping the pandemic in mind, we used Google Forms the medium to conduct our survey. Using online surveying methods like Google Form allowed us to avoid direct contact with people and also saved a lot of time. We prepared two separate Google Forms to collect data from two categories of users – the students and the class representatives of IUT. The form for students was made short, simple and had more close-ended questions since most students won't invest more than 5 minutes on a single form. Open-ended questions were marked as optional so that redundant and effortless comments were avoided. The Google form for class representatives were more detailed with more mandatory open-ended questions. Both forms followed the pyramid structure.

#### 3.1.2 Interview

For interviews, we used direct messages and phone calls as the medium since face-to-face interviews cannot be performed during the pandemic. We interviewed 15 people from all the categories of our target users which includes students, class representatives, teachers and staff of IUT. By interviewing, we were able to reach users like teachers and staff whom we were not able to reach using questionnaires. The data gathered from interviews were more reliable, descriptive and detailed.

#### 3.1.3 Representative Users

For our system, the representative users are the students, teachers, class representatives and staff of IUT. The primary purpose of our system is to find and book classrooms of IUT and show them to the desired users. The class representatives are responsible for finding and booking classrooms for unscheduled classes. The information about the booked rooms has to be related to teachers and students. The staff should also be able

to book classrooms according to the fixed schedule and play administrative roles in our system. So, all four categories of users have been included in our list of representative users.

## 3.2 RAW DATA

Before gathering data from our users, we had a rough idea of what the users wanted from our system. Most of the findings from the users matched our expectations but with a few exceptions. The following are some of the interesting finds from our survey: • While collecting data from the class representatives using Google forms, we found a response regarding the schedule of the auditorium. This is an interesting find because quizzes are often taken in the auditorium but we didn't consider that in our initial system. We shall now include the auditorium as a special room in our system.

• Before conducting the survey, we assumed that most classes had inadequate seat capacity. But studies showed mixed results regarding seat capacity. Because of the increasing number of students, many of us faced problems regarding the seat capacity of the classroom. But, this problem seems to occur less than expected.

This one's also an interesting find as it didn't meet our expected result. • We asked a lot of open ended questions to our class representatives. We found some great suggestions from those questions and one of them was to include a digital board in the first floor of academic buildings to show the classroom booking schedule. Our team thinks that it is a great idea and with the help of IUT, we can integrate such a system with our web-based and application-based classroom management system.

## 3.3 PERSONAS AND SCENARIOS

### 3.3.1 Persona: James

**Occupation:** Student (Class Representative)

**Age:** 20

**Goals:**

1. Quick search and book classrooms
2. Arrange rooms for quizzes and extra classes
3. Easily relay information about booked classrooms to his fellow classmates and teachers

**Frustrations:**

1. Wasting time looking for classrooms and booking them
2. Unable to arrange classrooms for extra classes and quizzes
3. Doing tedious work of relaying information about booked classrooms

**Scenario:**

James started his life at IUT with a bang, making new friends, going to new places and overall enjoying himself. Due to his outgoing and easy nature, he was selected as the Class Representative, but that was the start of all of his woes.

James could handle most of his work but one problem always seemed to haunt him: the issue of managing classrooms for lectures. Faculty members kept assigning extra classes, discussion classes and quizzes, even though no classrooms could be found to take them. Moreover, the classrooms assigned for regular classes were sometimes even occupied by senior students or faculty members, making the problem even worse. Even after a successful booking, James has to do the work he hates the most: contact his classmates and teachers and tell them about the room number and time for the unscheduled classes. James reaches home everyday being tired and demotivated.

**3.3.2 Persona: Albert**

**Occupation:** Lecturer

**Age:** 27

**Goals:**

1. Deliver perfect lectures to his students
2. Complete the given syllabus
3. Use various methods to teach his students
4. Manage time for his family during special occasions

**Frustrations:**

1. Unavailability of classroom resources for delivering perfect lectures
2. Being unable to take extra classes to complete the syllabus
3. Being unable to take discussion classes to clarify confusions of the students
4. Failing to manage time for his family on special occasions

**Scenario:**

Right after Albert graduated from IUT, he joined the institution as a lecturer. Albert is a perfectionist and tries to deliver the best lectures to his students. As number of students in IUT increased, Albert realized that his allocated classes do not have proper seat capacity. The classes don't have the right number of whiteboards or has projector facilities. Without these resources, Albert can't deliver proper lectures. He also realized that it is difficult to cancel classes and take extra classes because of the inflexibility of the current system. He is often forced to take classes and miss important family occasions. Albert is always hoping for an improved system that will allow rescheduling and provide him classes with adequate resources.

### **3.3.3 Persona: Stacy**

**Occupation:** Student

**Age:** 19

**Goals:**

1. Properly utilize class time
2. Have a clear concept on class lectures
3. Learn as much as possible

**Frustrations:**

1. Waiting in front of classes for the other class to end
2. Can't clarify confusions after classes
3. Unable to learn as much as expected

**Scenario:**

Stacy is who you would call a typical studious student. She has lots of big dreams and works hard to achieve those dreams. She gives full attention during class lectures and attends her classes on time. Unfortunately, she often has to wait in front of the classes when the same class gets booked by two different groups. Often these classes get cancelled and teachers have to compromise by shortening the syllabus. Stacy also gets confused after class lectures but she can hardly find discussion classes to clarify her confusions. She hopes for a system that would be able to provide classrooms easily for more and better interaction among students and teachers.

### **3.3.4 Persona: Ethan**

**Occupation:** University Staff

**Age:** 29

**Goals:**

1. Doing less monotonous work
2. Get a lot of leisure time

**Frustrations:**

1. Stressful job with monotonous work
2. Not getting as much leisure time as the other staff

**Scenario:**

Ethan has been a staff in IUT for three years. His duties as staff have changed throughout the years. Recently, he was appointed as the floor staff and was responsible for booking classrooms. At first he thought that it would be easy. But later on he realized that his work is monotonous and stressful. Every 10 minutes, a student comes to him to look for an empty classroom. And every single time he has to go through his book full of records and has to manually find the empty classroom. Doing this kind of work everyday makes him tired, stressed and demotivated. He wants to have a less stressful work and more leisure time. He hopes for a system that would make his work easier.

## 3.4 CURRENT SYSTEMS:

**Skedda** and **ClassroomBookings** are two popular current systems similar to our project.

### 3.4.1 Skedda

#### Overview:

- Book classrooms and labs online from any institution
- Targets a broader range of audience
- Can provide their own spaces as classrooms
- Can provide lots of extra features and flexibility
- Simple and modern user interface

#### Limitations:

- Skedda does not target a specific institution. So, the system is more generalized
- Most of the provided spaces are available in US only
- Most of the extra features are redundant for simple booking systems in IUT
- The added features and broader reach makes the software more expensive
- Can't be customized for IUT's specific purposes

### 3.4.2 Classroombookings

#### Overview:

- Schedule book classrooms and show it to users
- Simple, light-weight and easy to use
- Open-source and can be customized

### Limitations:

- Can't find classrooms according to a set of requirements
- Does not give access to students and class representatives to find or book classrooms
- Can't relay information to students and show them class schedules
- The user interface is outdated
- Customizable but requires technical knowledge to do so
- Can't perform real-time booking

Both Skedda and Classroombookings are quite similar to our system but needs to be modified in certain ways to fit the user requirements.

## 3.5 REQUIREMENTS:

According to the data gathered from our users, we categorized our mandatory requirements, want list and wish list.

### 3.5.1 Mandatory Requirements:

- *Booking classrooms without booking conflicts*

Booking conflicts occur when the same classroom gets book by two different groups. This feature arises from James' difficulty of successfully booking a classroom and Stacy's frustration of waiting in front of classes due to booking conflicts

- *Multiple bookings based on a schedule*

We included this feature to relieve staff like Ethan from monotonous workload. This also makes it easier for James to book multiple classrooms at different times

- *Finding and booking a classroom based on a set of requirements*

Albert couldn't give perfect features because the classroom didn't have enough resources. We included this feature so that teachers like Albert can now deliver perfect lectures and also take extra classes and discussion classes if required. Students like Stacy can now learn more and have clear concepts on topics.

- *Relaying information about classroom schedule to users*

James frustration about relaying information to his fellow classmates and teachers has been relived by including this automatic information relaying system

- ***Allow certain users to access and modify records***

Any mistakes done while booking can be modified. Both James and Ethan will be benefited by this feature. Cancelled classes can be accessed and this results in efficient resource management.

- ***Access control for multiple categories of users including administrative control***

In our system, James, Albert, Stacy and Ethan will play different roles. Access control is required to let every role do their work properly without interfering with others. Albert or Ethan can play administrative role in the system to access a wide array of added privileges/

- ***Web-based access from any platform***

Our system is developed for a large institution like IUT. Teachers and students come here from their homes in Dhaka. Hence, our system must be accessible from anywhere and the feature of web-based access has been added.

- ***Simple, easy and intuitive user interface***

We plan to give our users the best user experience. None of our representative users would like to invest time in order to start using the system. Our system has been made intuitive so that users can easily use all its features without any prior knowledge.

### **3.5.2 Want List:**

- Separate mobile app for convenience and notification-based features
- Adding specialized rooms like Lab Rooms and Auditoriums along with necessary features to handle them
- Relaying information about class schedules to teachers
- Categorize class types (e.g. regular class, extra class, quiz and so on) on schedules with color codes

### **3.5.3 Wish List:**

- Integrating the attendance, grading and learning management system with this system for a universal system in IUT
- Allow direct communication between class representatives and teachers
- Notifications or emails to students about unscheduled classes and quizzes from class representatives
- Add a feature in mobile app to sync with Google Calendar or any other built-in calendar



- Develop a version for embedded computers so that room the classroom schedule can be shown on digital boards

#### 3.5.4 Excluded Features:

We always decided to keep our program simple and easy to use for the best user experience. So, the program will always be restricted to a certain set of features. We have included a list of features we could have integrated in our system but decided not to do so.

- *Generalized system to book classrooms and spaces outside IUT*

Adds extra layers of complexity. Classrooms and extra spaces aren't that available near IUT. A transaction-based system also needs to be developed to add this feature and this makes the scale of the project bigger and makes the system less feasible to develop.

- *Customizable user interface*

Most users did not want a customizable user interface. We planned to keep the user interface simple. Adding options to customize can make the end user experience worse.

- *Direct communication between students and other users*

Very few users wanted directed communication between every user. Larger and more reliable servers are required for direct communication between every user. This violates economical and operational feasibility.

- *Allowing students to find and book classrooms*

Adding this feature might cause unwanted records by students. This can result in system abuse and/or confusion between the class representatives, students and teachers.

## 4 SRS

### 4.1 Overall Description

#### 4.1.1 Product Perspective

Large institutions don't have a single classroom assigned to a particular group of students. It becomes difficult to manually manage a large number of classrooms and the related booking records. Sometimes, the teachers and students cannot access classrooms due to poor management policies, manual errors and inability to process large amounts of information. To solve the issue CMS stores the information and allows the users to use it in an efficient manner to book classrooms. Such a system does not exist so far. So we hope to serve to a great extent being the foremost to introduce such an efficient system.

A context diagram has been provided in Appendix B to better represent our system.

A typical classroom management system stores the following information:

- **User Details:**

Each and every user, regardless of category will have their name, email, password, phone number and other miscellaneous information stored, along with some necessary information such as department and courses for relevant categories of users for use in booking and/or viewing the correct routine.

- **Location Details:**

The building and rooms are included, identified by their name and code and details such as capacity and equipment are also present.

- **Booking Details:**

It includes date, time, location as well as department and courses

#### 4.1.2 Product Functions

By interviewing the expected users and analyzing the results, we determined the main functions of the CMS, which are :

1. Search and book Classrooms
2. View Booking records
3. Add user, department, location Information by administration

4. Store data for multiple information of the institution
5. View Routine

An ERD (Entity Relationship Diagram) has been provided in Appendix B to give a brief overview of the different functions of the entire system.

#### **4.1.3 User Classes and Characteristics**

As we plan to implement the user interface in the easiest way possible, we hope to get users both with or without technical knowledge. In this case the targeted users for the CMS are Students/Class Representatives, teachers, administrators/staffs of any kind of educational institution. The three specified categories will have different access points, in order to facilitate the entire system. In general all the users will have some common accessibility, such as

1. Login to the system
2. Show user profile
3. Change user information
4. Recover password in case the user forgets

All the students will be able to perform these functions along with the mentioned basic users' functionalities

1. View routine

The class representatives and teachers will be able to perform more specialized functions, such as

1. Search room
2. Book room
3. View booking records
4. Delete records made by them
5. View Routine(s)

The administrators will have the basic user functionalities and also these specialized ones

1. Add user

2. Add building
3. Add room
4. Add department elements
5. Edit department information

A Use Case Diagram has been provided in Appendix B for better understanding of the user classes and their characteristics.

#### **4.1.4 Operating Environment**

Operating environment for the classroom management system is as listed below.

1. distributed database
2. client/server system
3. Operating System: Linux, Windows, iOS, Android
4. database: Firebase
5. platform: Javascript

#### **4.1.5 Design and Implementation Constraints**

- The project targets all the people of the institution which includes – teachers, students, staff and administration.
- The project is limited to online users only. Users must have access to the internet when needed.
- For the time being, our program is only targeting the users of IUT but we are planning to make a system like this available for any institution based on their requirements.

#### **4.1.6 User Documentation**

Since our project has an intuitive interface, we will not be providing any additional documentation but we will maintain a help page in our service for F.A.Q.s and any complex features that may be required to be explained in detail.

### 4.1.7 Assumptions and Dependencies

Let us assume that this is a CMS made only for a specific institution (IUT) and it is used in the following applications:

- A request for booking or cancellation of any lecture in a given classroom that is not a lab or an auditorium or any other room which is specifically designated as a lecture room.
- Buildings, rooms, departments, programmes and courses are changed infrequently and not in the middle of a term/ semester
- Faculties do not change in the middle of a term/ semester

Our application is dependent on the following products:

- free version of google firebase
- free version of firehosting
- a free trial of 1 year for domain hosting

## 4.2 External Interface Requirements

### 4.2.1 User Interfaces

The User Interface is built based on modern, responsive web design. For proper screen scaling on devices of all sizes, Bootstrap framework was chosen as it provides the best experience for responsive designs.

#### Landing Page

The web-app starts with a landing page containing the “Sign In” button along with guidelines on how to use the website for various categories of users. The landing page also contains the team members of the project along with their necessary contact details. In the footer of the web-page, there will be the email address of the team for further communication, any sort of related problem and bug fixes.

#### Sign-in Modal

A sign-in modal will be shown to allow the users to login to the system. The users need to provide their email address and password in two text fields. If an incorrect password is entered an error message will be shown. Or else the user will be logged in to the corresponding user pages.

There will be a forget password option clicking which will open a new window to recover the user password.

## **Admin Panel**

The user interface will then refer the user to their respective webpages. An admin will be referred to the admin panel where he will be granted a variety of privileges including adding, removing and modifying any sort of entity of the system. The entities include users, departments, programs, courses, routines, rooms and buildings. The admin also has access to the list of these entities through which he/she can visually modify the data.

The admin dashboard will have a list of all the entities that the admin can add, delete or modify. By clicking the option, the admin will be shown a list of those entities. There will be an add option to add at the top to add one of those entities. For example - if the admin chooses “Department” then the department list will be shown and by clicking on the add option, a department will be added. A popup window will appear which will ask for all the necessary details to create a Department. There will be a mandatory field for each entity which will be the ID of that field.

Beside each element of the list, there will be a cross icon which will easily let the admin delete a particular element. Clicking on any particular attribute will enable the admin to modify it. For instance - if “Student” is chosen then the admin can click on the cross icon at the right any particular student from the student list. Doing so will open a confirmation box, and by confirming it the admin will be able to delete records of that particular student from the database. Similarly, if the admin wants to change a particular attribute like name, or ID then he can just click on that attribute of a particular student and a text field will appear on the name which will enable the admin to modify it.

## **Student/Staff Panel**

The student panel will contain a single window which will show the routine assigned to the student. There will be a dashboard and on that there is only one tab called “Routine”. The routine will be a generated table which will replicate assigned routines to each section. Along with the course IDs in the routine, there will be room and building numbers as well. If the student is a CR then he/she will be able to see three more tabs in the dashboard. The tabs will be “Search Room”, “Book Room”, and “User Records”.

By clicking on the search room tab, a popup will appear with the title “Search Room”. The popup will allow the user to enter necessary details about a room. The available options of room details are starting time, ending time, capacity, air-conditioning, number of boards and projector facilities. For starting time and ending time, date-picker boxes will be used. For capacity, a text field will be used. The text field will receive number inputs only. Number of boards will have a drop-down menu which will let the user select from 0 to 8+, with a single digit increment between each option. The projector and air-conditioning will have check-in boxes. There will be two other boxes called “Course ID” and “Notes” to provide optional data. All the boxes will be empty by default.

A list of unoccupied rooms will be shown based on the given data. The user can select a room from the list to make a booking by clicking on the add-icon beside the

room list. At the top right corner of the popup, there will be a cross icon to close the popup.

The second option, booking a particular room, will allow the user to enter a room number and building number and book it for a specific time and date. There will be a popup which will show two input fields for selecting the room and building number. Then there will be a calendar (date-picker) popup box to select the time and date visually. After selecting the physical location, date and time, the user will see a tick-icon which denotes confirming his booking. By clicking on the confirmation icon the screen will show a pop-in window “Room booked successfully” There will also be a cross-icon and clicking this icon will cancel the whole event popup. At the top right corner of the popup window there will also be a cross icon to close the popup similarly.

The third option, “User Records”, will open a popup window showing a list of all the user records and the necessary details including the room number, building number, date, starting time, ending time, booking time (the date and time when the room was booked), Course ID and notes. By clicking on the cross button at the side of each record, the user will be able to delete the record. There will be a confirmation box, and by confirming it, the user will be shown the message “Record Deleted Successfully” in a popup. Clicking on any field of the record, except booking time, will enable the user to modify that particular record. After modification, a confirmation icon at the right of each record has to be clicked, which will then open a confirmation popup like the delete record option.

### **User Profile Page**

At the top right corner of the dashboard there will be a “User Profile” button and a sign-out button. This portion will be the same for all categories of signed-in users. The user profile will open a new page that shows all the corresponding user information including but not limited to name, email address, password and phone number. There will be a change option beside each field which will let the user enter updated data. If any field is changed then password verification will be required. If the wrong password is given then a “wrong password” message is shown in red on top of the password field.

### **4.2.2 Hardware Interfaces**

The hardware interfaces required by our system include:

- Any operating system that has support for web-browsers, on desktop or mobile platforms
- A browser that supports CSS, HTML5 and JavaScript

As we will be hosting the server and database on online services, there will be no additional hardware requirements on the client side other than a stable internet connection. Users will be able to access the website through the browser using HTTPS and all formats of input and output are digital in nature.

### 4.2.3 Software Interfaces

Front-end Software:

Markup Language: HTML 5

Styling: CSS 3

Styling Framework: Bootstrap (version 4.5.3)

Scripting Language: Javascript (ECMAScript-6 2020 release)

Scripting Framework - React Native (version 0.63.4)

- Redux (version 4.0.5)

Back-end Software:

Scripting Language: Javascript

Scripting Framework - React Native

Server: Google Cloud Computing Services

Database: Cloud Firestore

Tools Used:

Text Editor: Visual Studio Code (version 1.52)

Backend Development Platform: Firebase (Javascript SDK version 8.1.2)

Web-Hosting: Firehosting

User Authentication Software: Firebase Authentication

Javascript Runtime Environment: Node.js (version 15.2.1)

### 4.2.4 Communications Interfaces

Since we will be hosting our server and database on google's services, security is of minimal concern and users will be able to access the domain using HTTPS socket from any web browser that supports HTML5 and JavaScript. HTTPS itself ensures an added layer of security for the users, ensuring that no data is stolen or modified maliciously.

## 4.3 System Features

The features are listed below in order of priority:

### 4.3.1 Book Room without Booking Conflicts

#### Description and Priority

The booking room feature allows us to select a room from a list of rooms and book that room at a particular time and date. While making the booking, the function checks if the room is booked by another user at the same time and date. By using this feature the user will be able to safely book anyone room without booking conflicts.

This is the core and arguably the most important feature of your system and is of HIGH priority. A relative weight of this feature will be 10.



### **Stimulus/Response Sequences**

Our database contains a record entity with a unique Record ID. Record is fundamentally composed of building number, room number, starting time, ending time and date. For convenience, when we say a room is selected, we mean that a particular physical location (room + building number) is selected. The UI sends that room number to the server which then makes a query to the database to filter all the records under that room number. The UI also sends the date, starting time and ending time required to make the record. Firstly, the filtered records are re-filtered by date. Now, we have all the records of that room in particular. Then a checking is performed by the server to ensure that none starting time isn't between any of the records or is equal to the starting time of any of the records. For example - if the required starting time is 9 AM, all the records of the room must not have 9 AM between the starting and ending time. The function performs another similar check with the ending time. If both the checks return true, then no booking conflict has occurred and the server adds the corresponding data to the records entity along with adding the identity of the user. Otherwise, the server returns an error signal which is interpreted at the UI as booking conflict.

### **Functional Requirements**

REQ - 1: User must be logged in as CR/ Faculty

REQ - 2: The provided date and time must be valid.

REQ - 3: Users are limited to adding a particular number of records to prevent system abuse. Users are hence advised to not add and delete redundant records whimsically.

## **4.3.2 Search Rooms based on Room Details**

### **Description and Priority**

The searching room feature will enable our users to enter room details - date, starting time, ending time, capacity, projector facilities, air-conditioning facilities and number of boards. The feature will show our user a list of rooms from which the user will be able to select and book the particular room using the book room feature.

This will be one of the core features of our webapp and will have HIGH priority. A relative weight of this feature will be 9.

### **Stimulus/Response Sequences**

Details about the room are sent by the UI to the server. The servers will then filter rooms in the database based on those criteria. Then for each room, booking conflict is checked using the date and time provided by the user. The booking conflict is checked by filtering the records with room ID of each room and then performing the conflict checking function. Firebase Function allows us to easily make such back-end functions that can be used on our Firestore Database. Finally the remaining list of rooms without any conflict is sent to the UI which displays the list to the user.

## **Functional Requirements**

REQ - 1: User must be logged in as CR/ Faculty

REQ - 2: The provided date and time must be valid.

REQ - 3: In order to prevent system abuse, only a certain number of searched rooms can be booked.

### **4.3.3 User Login and Authentication**

#### **Description and Priority**

The user login feature gives a form of structure to our program. The login feature allows users to safely access the system using unique credentials. The login also allows role based access control to our system. Along with login, authentication is performed so that no other user can access the system identifying as another user, or be able to access another user's account without external help.

Login and authentication is done in our system with a HIGH priority keeping the user security on mind. User login and authentication also allows us to ensure a stable system and lets us easily identify, recognize and tackle unwanted events caused by the users. The user login has a relative weight of 8.

#### **Stimulus/Response Sequences**

The whole user login procedure can be subdivided into two parts - the user interface login and backend user authentication. The user interface login deals with the users providing their necessary credentials. The credentials are then sent to the database i.e Firestore where the authentication is performed. Before that, the user interface provides the option of password recovery which allows our user to recover their passwords using either Email or Phone Number. For email recovery, a referral email is sent to that particular email address after confirming it. Using the referral email, the user can change the password. Similarly for phone recovery, the phone number will be first verified and then a verification code will be sent to that number. If the verification code is right then the user will be given the ability to change his/her password. If the verification code is wrong then there will be an option to resend the code.

The second part is the user authentication. When the login credentials are successfully entered, a connection will be made with the database. This allows direct communication between the webapp and the server's database. To establish this connection along with verifying the login credentials, Firebase has a special section called Firebase Authentication. This allows the admin to easily add users along with tackle all the issues related to authentication. Firebase authentication has high security, easy to use and can be easily implemented in our system.

## **Functional Requirements**

REQ-1: Users must be assigned an ID or username before logging into the system.

REQ-2: Users must have a working email address or password for the password recovery

methods.

REQ-3: When the password is changed, the user must provide a valid password which meets the password criteria of our system.

REQ-4: The provided new password must not be the same as the old password.

#### **4.3.4 View Routine with associated Room Number**

##### **Description and Priority**

The view routine option is the primary feature of our student user type. When the students are successfully logged in, they are shown the routine which corresponds to the course routine assigned to them in their sections. Under each entry of the routine table, the room number where that course is supposed to take place will be shown. This room number is dynamically updated i.e. if the course is taken in a different room then it will be instantaneously shown in the routine.

Viewing routine is a key feature in enabling real-time access to our system by the users. This feature is of MEDIUM priority and is given a relative weight of 8.

##### **Stimulus/Response Sequences**

Each student has been assigned a routine ID. The routine ID corresponds to a particular routine table. The routine table has 5 rows for each working day - Monday, Tuesday, Wednesday, Thursday, and Friday. The days of the routine can be modified by the admin depending on the working days of the institution. The rows will be divided by the columns into class periods. A class time period is basically a starting time and an ending time. Each class time period will have a course ID, teach

##### **Functional Requirements**

REQ-1: User must be logged in as a student to view the routine

REQ-2: The student must have a routine ID given by the admin

REQ-3: The routine must be manually set by the admin

REQ-4: The routine must refer to valid courses, room and building number.

#### **4.3.5 View, Modify and Delete Booking Records**

##### **Description and Priority**

All the records made by the user can later be accessed using the User Records tab. From the list of records, the user can remove a record or change a particular attribute of the record. This provides additional flexibility to our system and allows us to tackle situations like cancelled class effectively.

The feature is of MEDIUM priority and helps us in achieving the core philosophy of our system. The feature is relatively weighted at 7.

### **Stimulus/Response Sequences**

In order to view the records, a query is made to the database by the server which filters all the records based on the ID of that particular user ordered by building number and room number. The list of records returned by the database is then displayed to our user. When a record is selected then the ID of that record is sent by the user interface to the server. The server then deletes the record with that particular record ID.

Modifying a record is also similar to deletion. The record ID and the attribute that needs to be updated along with the updated value is sent to the server. Server makes a query to select that particular record and then edits that particular attribute to the updated value.

### **Functional Requirements**

REQ - 1: Users must be logged in as CR or Faculty to access the records.

REQ - 2: During record modification, the user must provide information that doesn't result in booking conflicts or else an error message will be shown. This is why modifying records is discouraged. Users are encouraged to delete the record and then search and book a room again using the search room function.

## **4.3.6 Administration Panel**

### **Description and Priority**

In any system, administration options allow high system stability and longer life time. In our case, we want the system to be self-dependent i.e. all the users of the system should be able to collectively make any changes if required without any technical support. To ensure this, an admin panel was added with additional privileges that allows specific users to perform a set of functions which interact with the core entities of our system. Administration panel basically enables direct communication with the database.

The administration panel allows viewing options to entities of our system - department, program, course, routine, user and record. The elements of these entities are listed and shown to the admin. Elements can also be added, removed or modified from this list.

Administration panel is of MEDIUM priority and has a relative weight of 6.

### **Stimulus/Response Sequences**

In order to view a list, first the admin chooses the entity to view in the UI. The name of the entity is then sent to the server and a query is made to the database to see all the elements of the entity along with the details. The details are then sent back to the client side and displayed on the screen. To add a record all the necessary details to add that record is taken from the UI and sent to the server. The server processes a command that inserts the data in that particular entity in the form of a record. Similarly, to remove an element the ID of that element in the entity is taken from the UI and sent to the back-end server. The server then generates a command that will ask the database to

remove that element from that particular entity. Modification is done similarly. All the procedures correspond to 4.5.2

### **Functional Requirements**

REQ - 1: Administration panel is a highly important area. Admins are to be selected wisely as an admin is capable of disrupting the whole system resulting in data loss.

REQ - 2: Data entered by the admin must be double-checked as the system isn't capable of verifying most of the data.

### **4.3.7 Mobile App-based Notification System**

#### **Description and Priority**

A mobile application is planned to be developed to give the project a bigger scope across different platforms. As web-apps are often taxing to mobile operating systems, the mobile application will also allow better performance for certain users. A notification based system can be used to remind students about class time and class room number which helps us achieve higher user satisfaction.

Although this system can be considered to be really important as it part of our core design philosophy, it is difficult to implement this feature since a notification system requires a separate mobile app. For this reason this feature is of LOW priority with a weight of 4.

#### **Stimulus/Response Sequences**

The mobile based app will have users logged in the system and will have auto-start options enabled. The app will always check the data and time of the mobile. Whenever the app encounters a time which is 10 minutes prior to the starting time of a class, the app will send a notification to the user which will contain the course-ID and the room where the course classes are meant to be taken. To do so the app will always keep the routine of the user stored, and will dynamically update the routine when there's internet connection.

### **Functional Requirements**

REQ - 1: An android based mobile system capable of running modern apps (Minimum android version is yet to be determined)

REQ - 2: The app has to be allowed to sent notifications to the phone

REQ - 3: Proper data and time should be set on the phone.

REQ - 4: Frequent internet connection is required to keep the room numbers up to date.

## 4.4 Other Nonfunctional Requirements

### 4.4.1 Performance Requirements

Since we will be using third-party services to keep our service up and running, the only performance improvements we can do are related to the database and the JavaScript API for the web-app.

- Database: For the database, we have designed it in accordance to the 1st Normal Form such that none of attributes in each of the entities can be broken down any further, thus reducing redundancy and keeping null values to a minimum
- JavaScript: As we are designing the web-app from scratch, we are constantly optimizing and making changes to make the code more efficient and easier to run on web-browsers.

### 4.4.2 Safety Requirements

As our web-app is just a Management Information System (MIS), there are no physical safety requirement but we have three measures of security in case of data loss:

- Google has multiple data centers in multiple parts of the world with redundant backups of data so that if any data is lost from a single data center, it can be recovered quite easily.
- If any user cannot complete a session due to a disconnection or blackout, no changes will be made in the database for incomplete sessions to ensure data validity.
- A log file is updated each time any changes are made and this file is constantly synced to an offline storage so that in the case that all google data centers lose data at once at any time, we can restore our data from the log file without any issues whatsoever.

### 4.4.3 Security Requirements

All the users will have to be added manually by the admin, and no user can suddenly create an account, ensuring that only people on the institution can have an account assigned to them. Additional security features such as database and web server protection are handled within the third-party services that we are using and the HTTPS portal will ensure the security of our user's privacy and data when they are accessing our domain. Also, since all activities in the domain will be backed-up in a log file, it will help us analyze any malicious attacks and improve our web-app to better defend against malicious attacks as we face them.

#### **4.4.4 Software Quality Attributes**

- Adaptability: Using a web-app with HTML5 and CSS ensures that our system is scalable in UI as well as cross-platform compatibility, hence we can say that it has great adaptability.
- Availability: Our service can be accessed at any time as long as it is within the monthly limit and bandwidth of the hosting service we are using. These two factors can be improved simply by moving to paid plans.
- Correctness: We have measures in place to verify each and every changes made to the database to ensure that all data entered are complete and correct at all times.
- Maintainability: Maintenance is a non-issue as software revisions of the services we are using come few and far in between with minimal changes so chances of our codebase breaking are very low.
- Reliability: Our system is reliable in the sense that all data maintain correctness thus ensuring that no corrupt data enters the database, ensuring that the database remains error-free and our website is designed in such a way that unprecedented actions by the users will return with an error and prevent any changes from being made that may break the system.
- Usability: We have designed a user-friendly interface for our users, keeping in mind the current navigation methods and conventions so that users can experience an intuitive interface and accomplish their goals without needing to go over any additional resources or documents.

#### **4.4.5 Business Rules**

Our system will be free to use and our code will also be made open-source so that other developers can reuse our code in their projects. The source code will be provided on github for easier sharing.

## 5 Feasibility Analysis:

### 5.1 Technical

The technical issues are developing the website, using the services, gathering institutional information and hosting the website.

1. Our project uses web-based markup languages, scripting languages and services from Firebase. The members of our team are already experienced in web development and related services.
2. Operating the database and authenticating the user will be done by Google's back-end service provider - Firebase. The services of Firebase are easy to learn and have good documentation. Firestore and Firebase authentication are used for database services and user authentication respectively.
3. Information about IUT can be accessed from the staff. Our team members have already talked with the staff and they have ensured that they will provide us information regarding the students, faculties, departments, programs, routines and rooms.
4. Firehosting will be used to host the website which is also relatively easy to learn.

From the current analysis, we have concluded that the system is technically feasible.

### 5.2 Economical

Most of the tools and services used in our project are completely free. Only Firestore has a few constraints being a freemium software.

Firestore allows 0.05 Million read operation, 0.05 Million small operations and 1 GB of stored data. There is a 1 GB incoming and outgoing bandwidth cap. Currently these are the constraints of the free version of Firestore and we are operating well within these constraints. But if the users of our system increase to more than 700-800 (estimated) then we will have to switch from free services of Firebase to a monthly subscription pack depending on the number of extra users of our system.

### 5.3 Operational

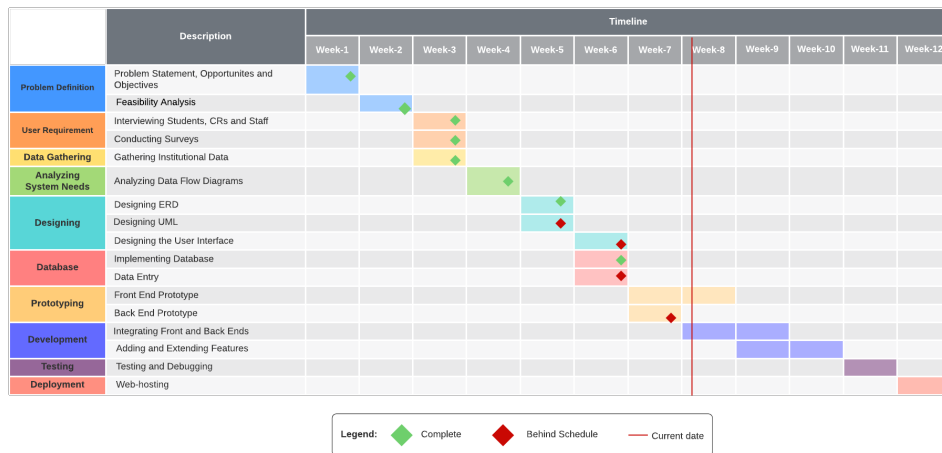
There is no operational cost rather than standard maintenance check performed by the staff. Every year, data of the incoming students has to be manually provided to the



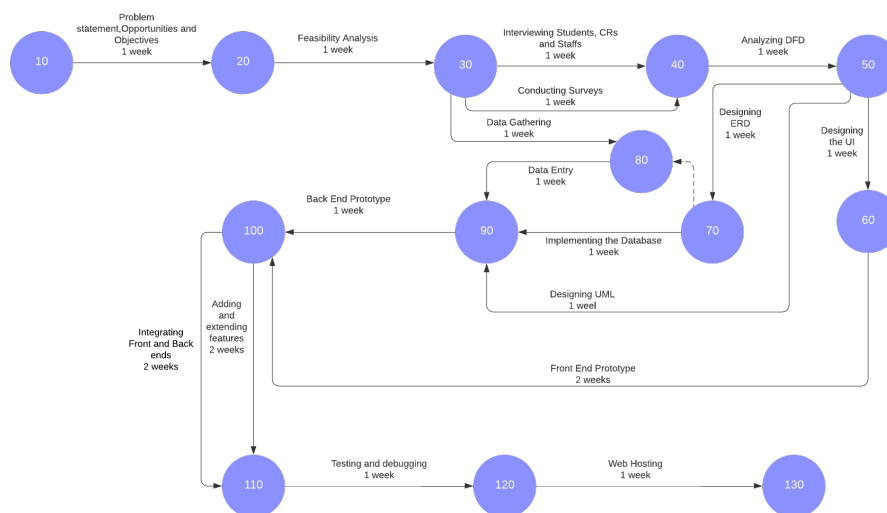
database. The operational can also vary depending on our number of users which has been analyzed in the previous sub-section.

## 6 Project Timeline

### 6.1 Gantt Chart



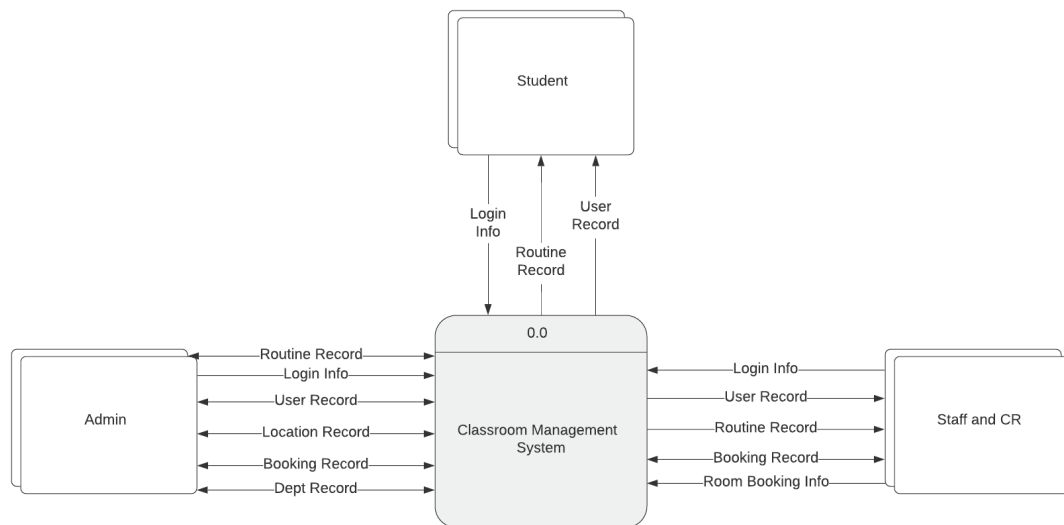
### 6.2 PERT diagram



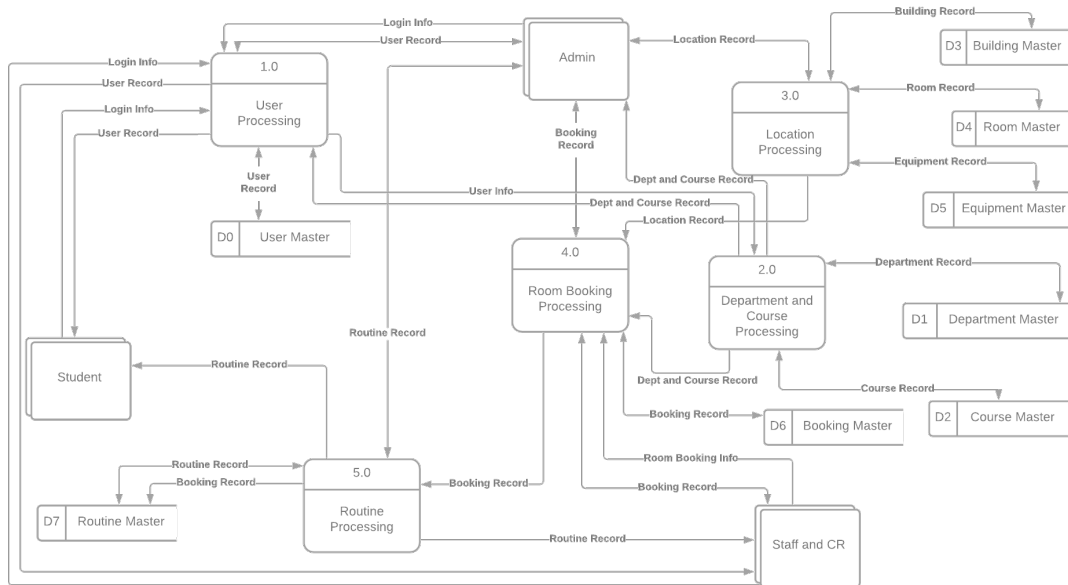
## 7 Data Flow Diagrams:

DFD graphically characterizes data processes and flows in a business system.

### 7.1 Context Diagram

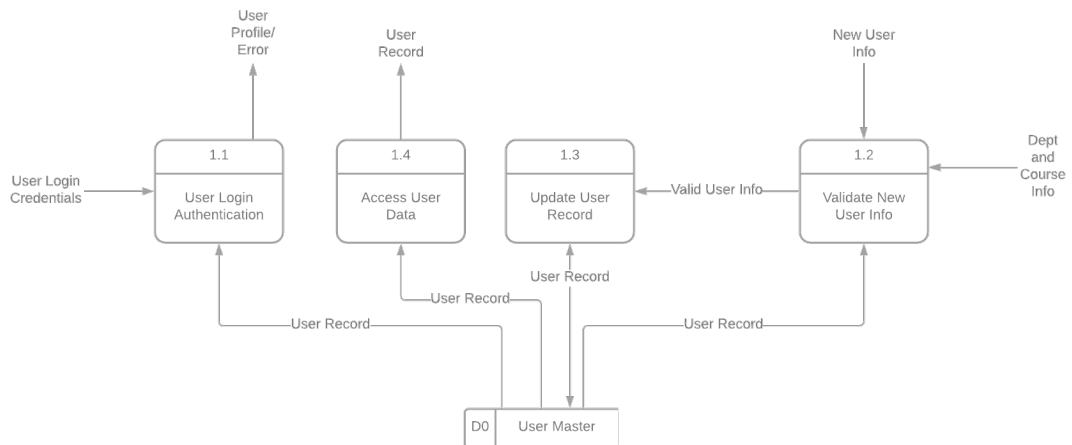


## 7.2 Diagram 0

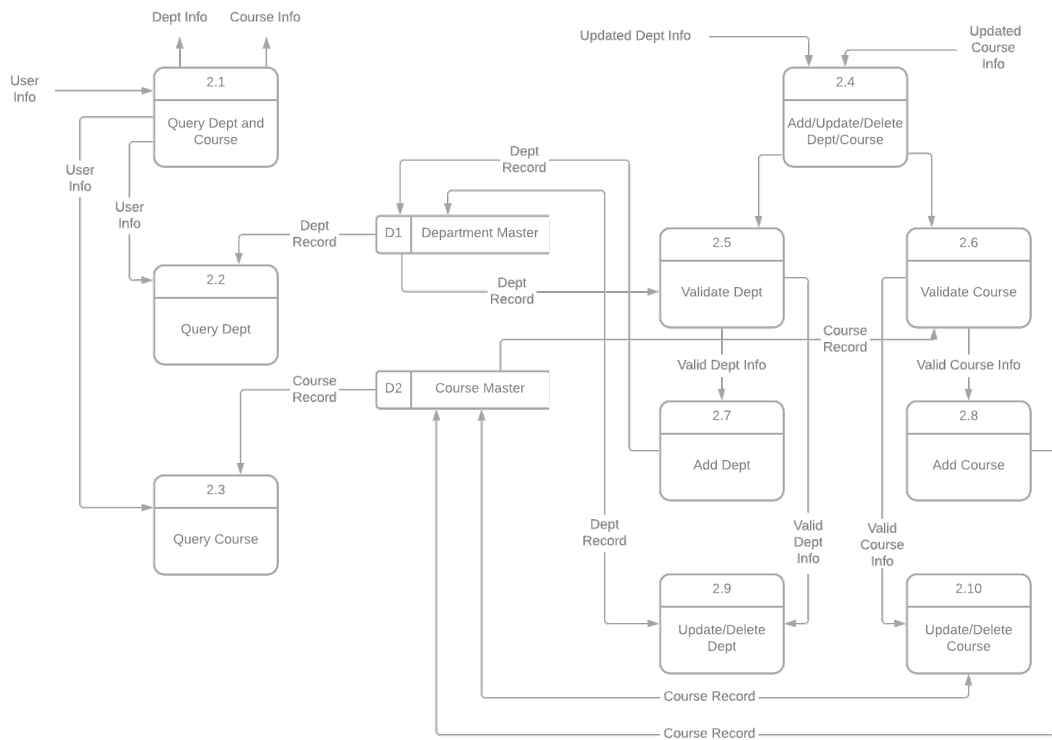


## 7.3 Child Diagram

### 7.3.1 Child Diagram 1



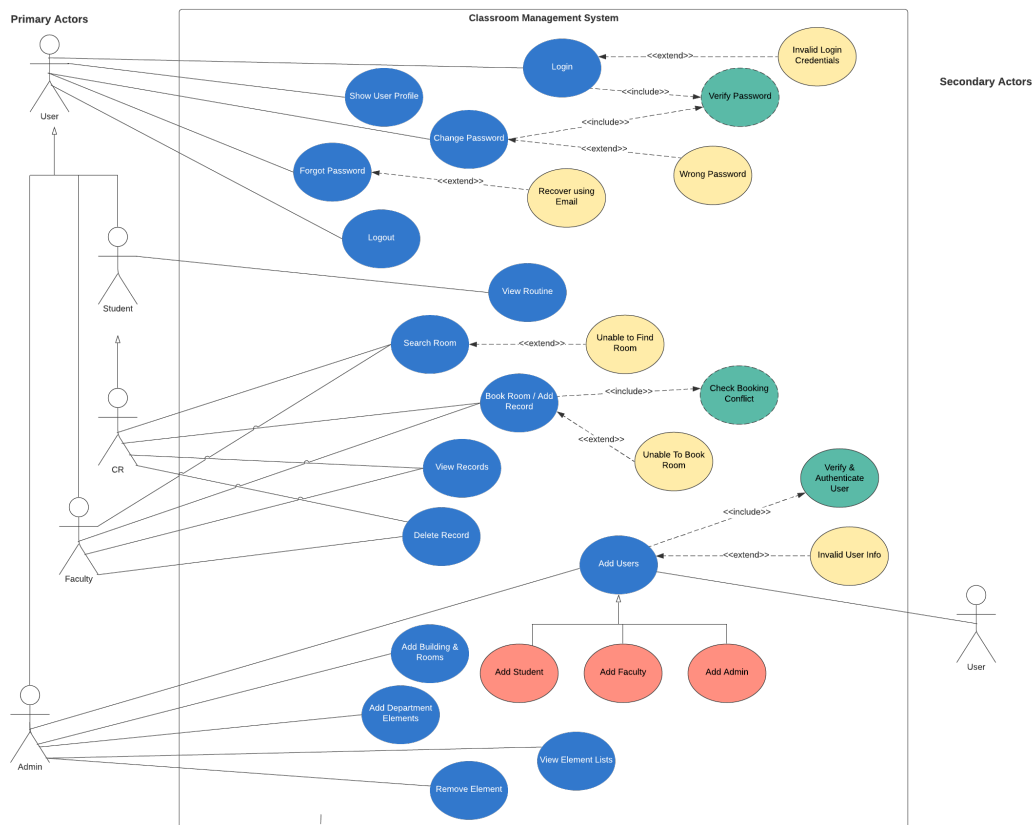
### 7.3.2 Child Diagram 2



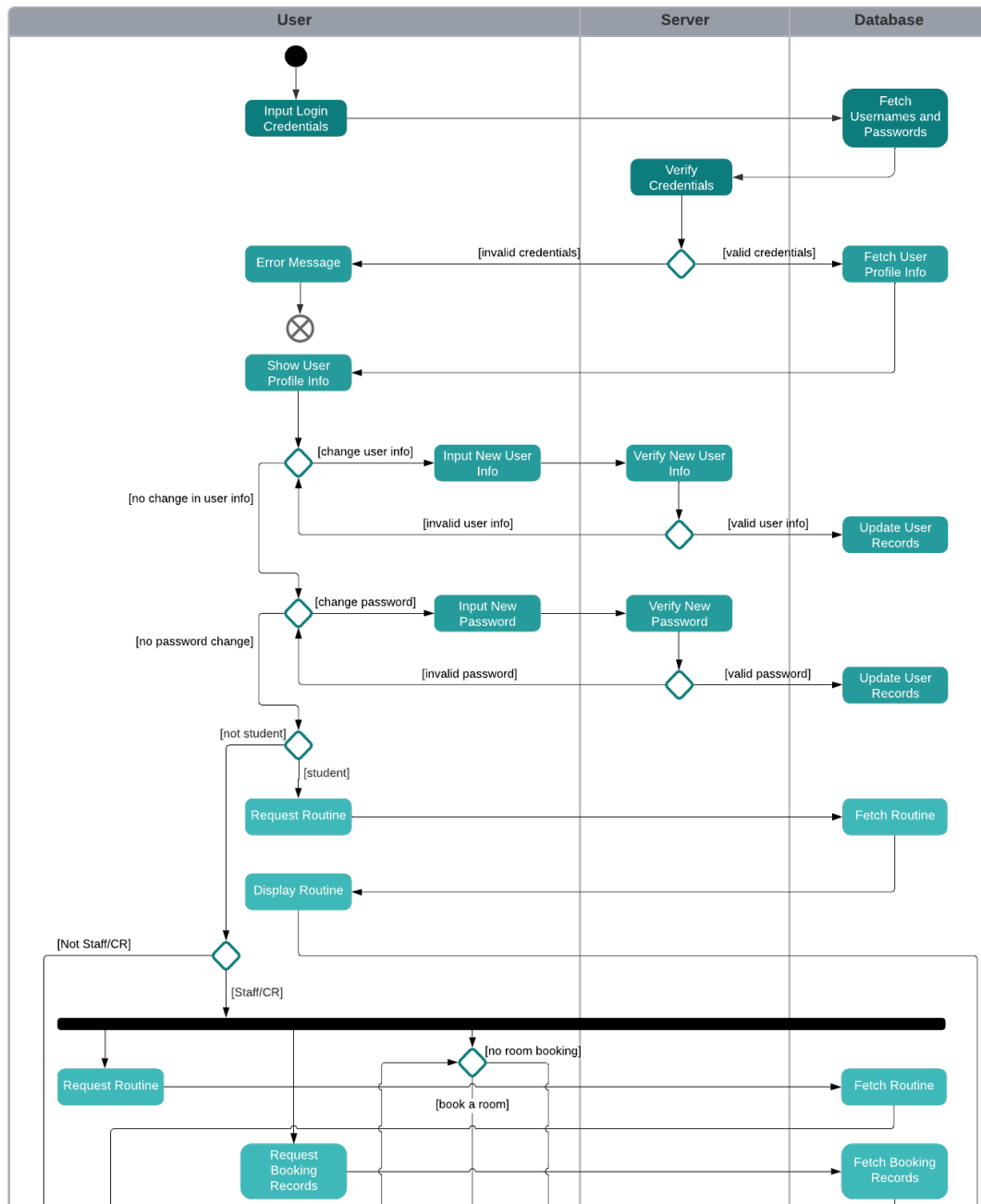
## 8 UML Diagrams:

A UML diagram is a diagram based on the UML (Unified Modeling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system.

### 8.1 Use Case Diagram



## 8.2 Activity Diagram

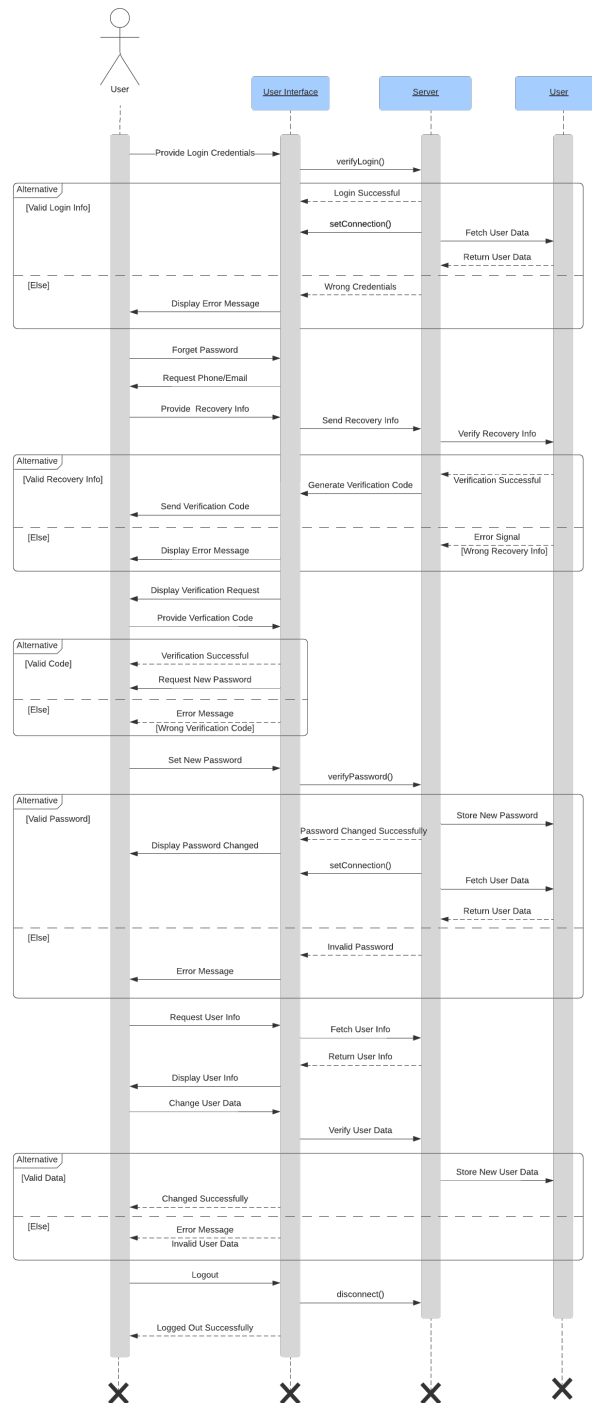




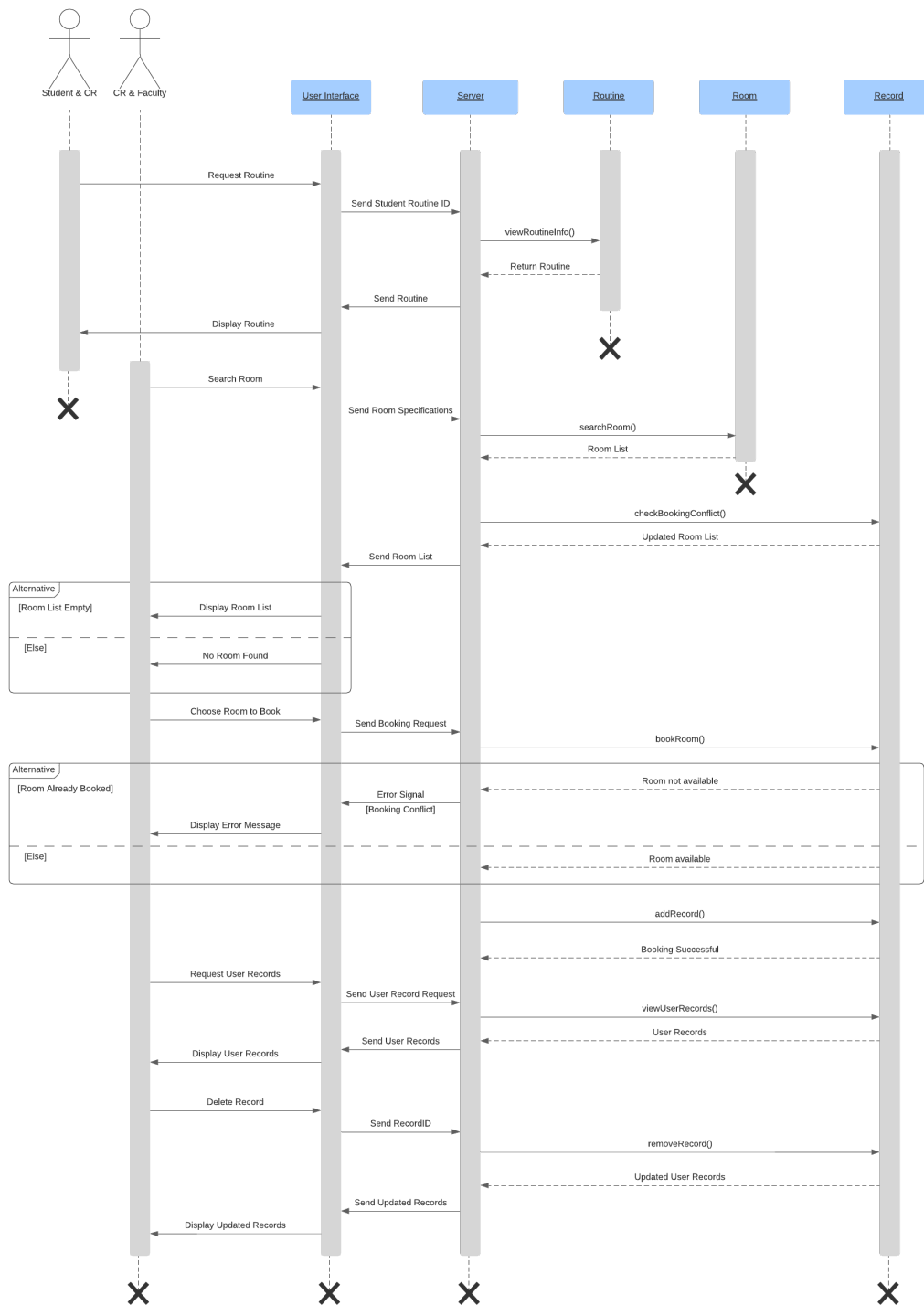


## 8.3 Sequence Diagram

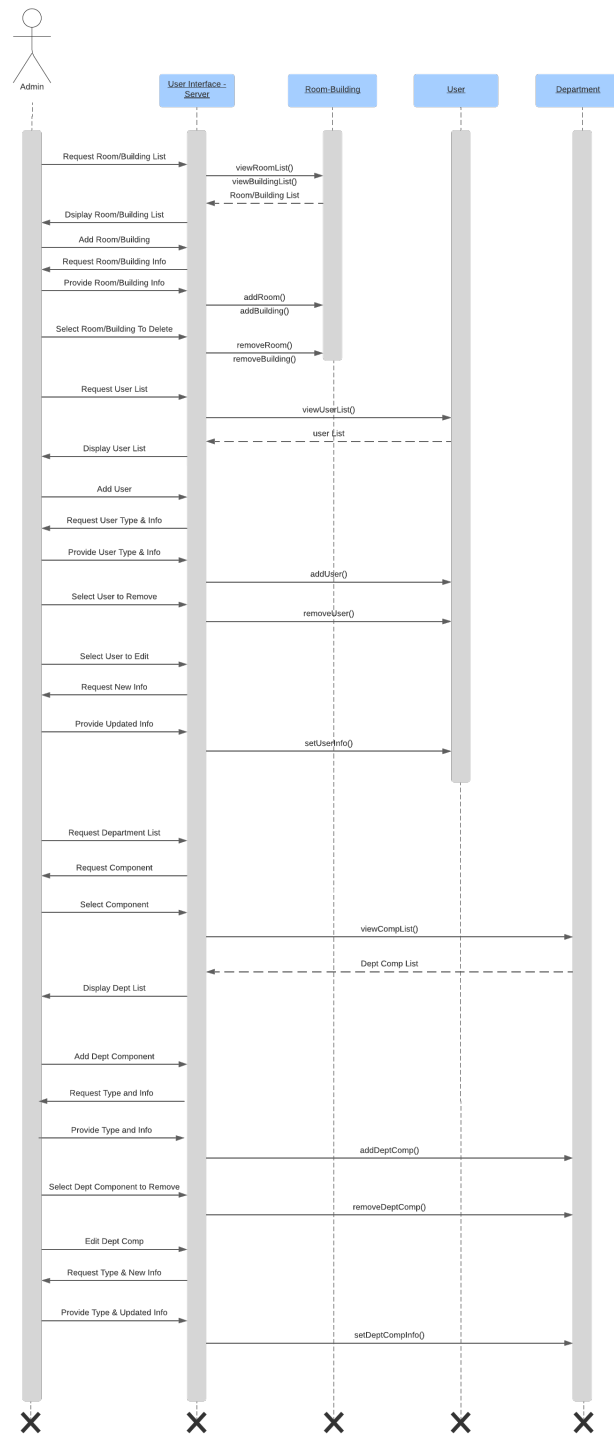
### 8.3.1 Sequence Diagram (User)



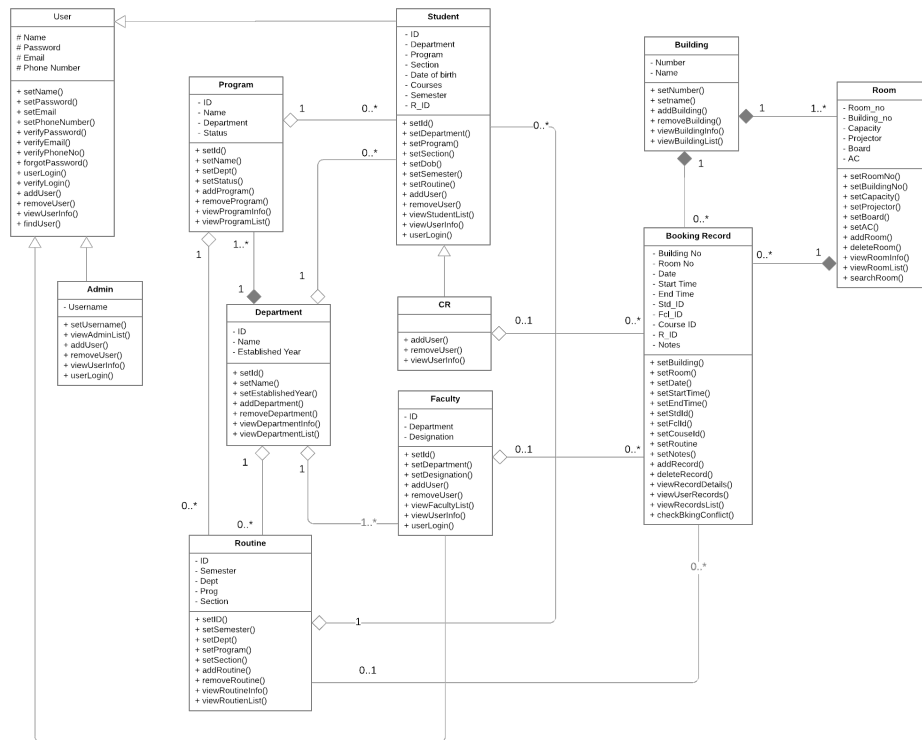
### 8.3.2 Sequence Diagram (CR & Faculty)



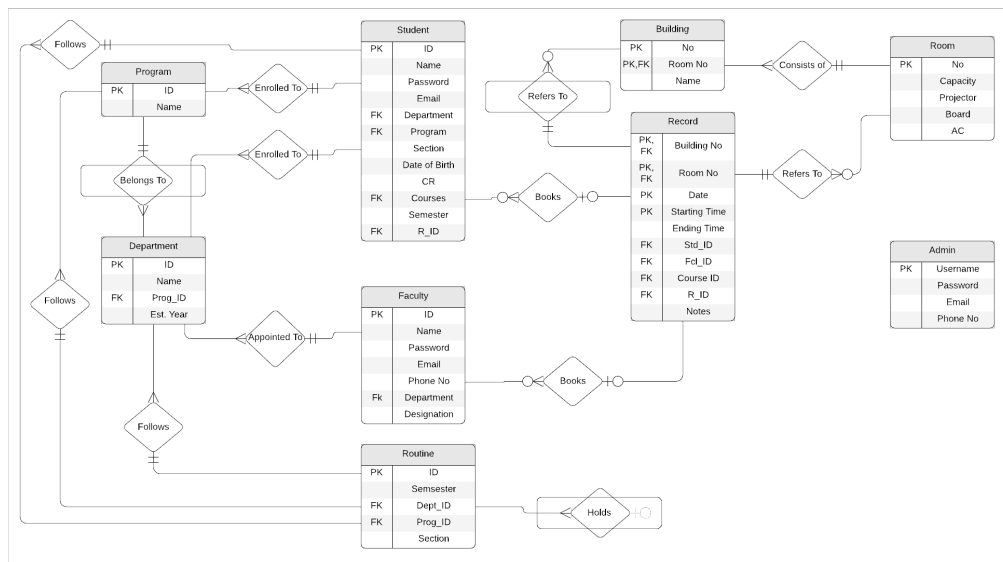
### 8.3.3 Sequence Diagram (Admin)



## 8.4 Class Diagram



## 8.5 ERD



## 9 Prototypes:

We have added screenshots of the prototype we developed for our project. We followed the agile model in developing the base prototype model and extending the features.

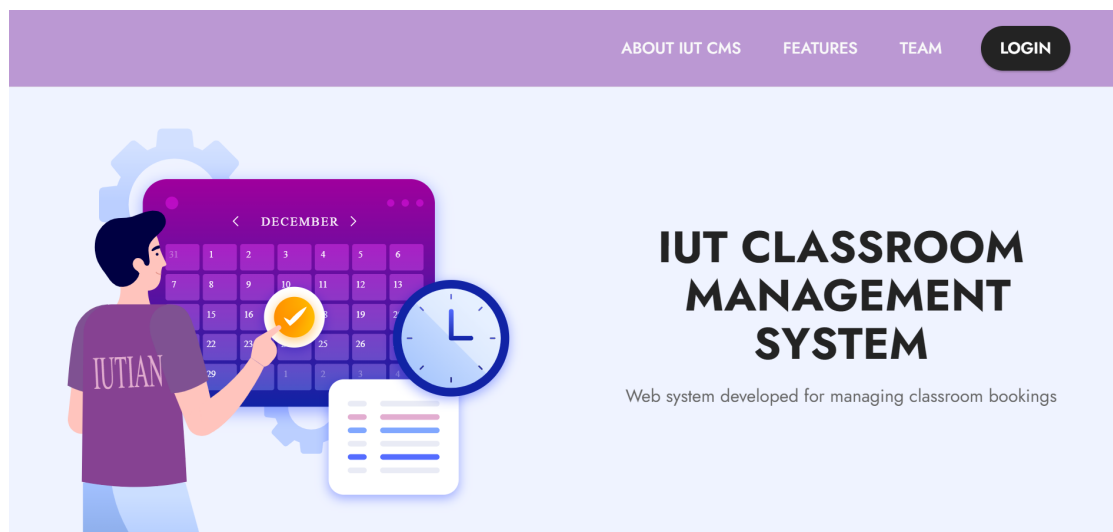




Figure 9.1: A screenshot of the landing page of our web-app

×

---

## User Login

 Email

 Password

LOGIN

Forgot Password?

Figure 9.2: The user login modal

### Search Room

Building Number : Any ▾

Capacity (max 200): 20

Equipments : ☐ AC ☐ Projector

Boards : Low (less than 4) ▾

Date: dd-----yyyy

Time Slot: 9:00 AM - 9:50 AM ▾

SEARCH ROOM

### Book Room

Room Number : ▾

Routine ID : CSE18-1 ▾

Course ID:

Notes:

BOOK ROOM

Figure 9.3: Search room and book room forms

Building No.	Room No.	Date	Starting Time	Ending Time	Course ID	Routine ID	Notes	
1	102	27-01-2021	9:00 AM	9:50 AM		CSE18-1	Come 15 minutes before class	X
2	203	27-01-2021	9:00 AM	9:50 AM		CSE18-1	None	X
3	101	29-01-2021	9:00 AM	9:50 AM		CSE18-1	None	X

Figure 9.4: User Records List

# 10 Conclusion and Future work:

## 10.1 Future work:

We achieved most of the objectives through our current system. We are planning to extend these ideas and integrate our system with other student management systems in IUT. Some of our future plans are:

- **Show room number in the routine:** To relay the information of booked rooms to students and faculty members, we intend to add room numbers of the classes in the schedule.
- **Notification before the classes:** We plan to develop a mobile app that will notify users the room number before the classes start.
- **Faculty Routine:** A similar routine about the faculty members will be shown when users log in as faculty.
- **Course Enrollment:** We are planning to integrate the course enrollment system with our current system to validate course inputs.
- **Attendance:** We plan to add features that will enable faculty members to take attendance for particular courses and relay the results to the students. Students can also give their attendance using the system.
- **Recurring Booking:** Include a feature that allows users to book classroom on multiple days of a week for a set period of time into the future.
- **Grading System:** Integrate the grading system with our current system to relay student grades.

## 10.2 Conclusion:

We, with our knowledge, willpower and hard work accomplished most of our goals for this project. Our main goal was to create something really usable in real life. We were able to do so to some extent by implementing the most wanted features with an user friendly interface and many other flexibilities. The most handy issue in the project is that it is accessible from anywhere with any kind of device just with an internet connection. It is surely to say this project will save a huge amount of time for our valuable users. Nonetheless, we hope to increase the scope of the project in future days.



## Appendix: Glossary

<i>CMS</i>	- Classroom Management System
<i>CR</i>	- Class Representative
<i>CSS</i>	- Cascading Style Sheets
<i>DB</i>	- Database
<i>HTML</i>	- Hypertext Markup Language
<i>HTTPS</i>	- Hypertext Transfer Protocol Secure
<i>JS</i>	- JavaScript
<i>SRS</i>	- Software Requirements Specification
<i>UI</i>	- User Interface