Software Requirement and Design Specifications

Hostel Management System

Version (1)

Course Code	CS-3004
Instructor	Miss Nida Munawar
Project Team	Fabiha Atique 20k-0369 Muhammad Usama 20k-0190
Submission Date	7/December/2022

[Instructions]

- *No section of template should be deleted. You can write ‘Not applicable’ if a section is not applicable to your project. But all sections must exist in the final document.*

- *All comments/examples mentioned in square brackets ([]) are in the template for explanation purposes and must be replaced / removed in final document.*

- *This’ Instruction’ section should also be removed from the final document.*

Table of Contents

1. Introduction

1.1. Purpose of Document

The purpose of this document is to present our academic project is an effective way and give a tour to our audience about how a hostel management system words.

1.2. Intended Audience

Course instructor, Instructor's Assistant, students enrolled in the course.

1.3 Definition of Terms, Acronyms and Abbreviations

Not applicable

Term	**Description**
ASP	Active Server Pages
DD	Design Specification

1.4 Document Convention

Font size is 10 for the normal paragraphs and 12 for headings.

Font: Areal

2. Overall System Description

2.1. Project Background

Many fellow students at our university are living in hostels, it is important to know about the management system of the institutions we belong to. Hence, we have presented a model to demonstrate how a normal hostel works.

2.2. Project Scope

This system can be used by students to enhance their knowledge about how management systems work in practical life, . Also, it can be used by the actual hostel owners to design and perform their management using our system.

2.3. Not In Scope

Chat box for students' assistance, salary management of warden and rent management of the whole hostel building.

2.4. Project Objectives

To Give an idea to about of how management is done, how many functionalities are involved in seemingly simple tasks.

2.5. Stakeholders

Users (students and warden) admin and database developer.

2.6. Operating Environment

Web browsers on laptops and computers.

2.7. System Constraints

The following are the few constraints due to which it gets harder for the team to provide the best of the services.

- *Web server limitations.*
- *Internet not working*
- *Many students who do not live in hostels or are not in favor of living in hostels.*
- *The Authorities are not in favor of the hostels to be managed privately.*
- *The environment where our system is deployed do not have skilled people, for example people do not know how to manage the database etc.*
- *User constraints, the students and the wardens are non-serious and do not cooperate with the system.*

2.8. Assumptions & Dependencies

- *We are dependent on our database system, due to which we can store and retrieve so much data.*
- *We are dependent on the internet for this system to work.*

3. External Interface Requirement

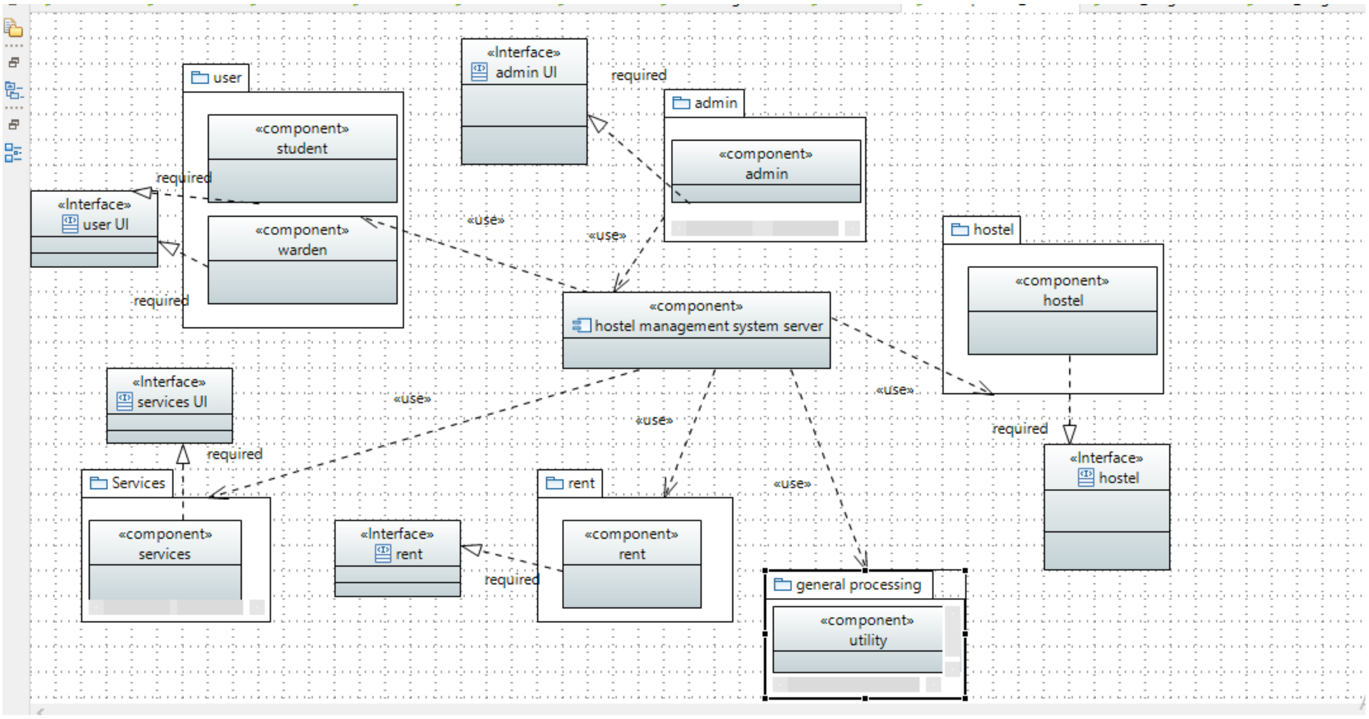
external requirements are shown in the deployment diagrams below.

3.1 Hardware interfaces

We will be needing a computer to manage and view the database.

3.1. Software Interfaces

The following component diagram describes the components of the classes, their connections, their dependencies and uses.



3.2. Communications Interfaces

[non applied as of now]

4. Functional Requirements

4.1. Functional Hierarchy

The functionalities are either by the student or the warden.

For warden

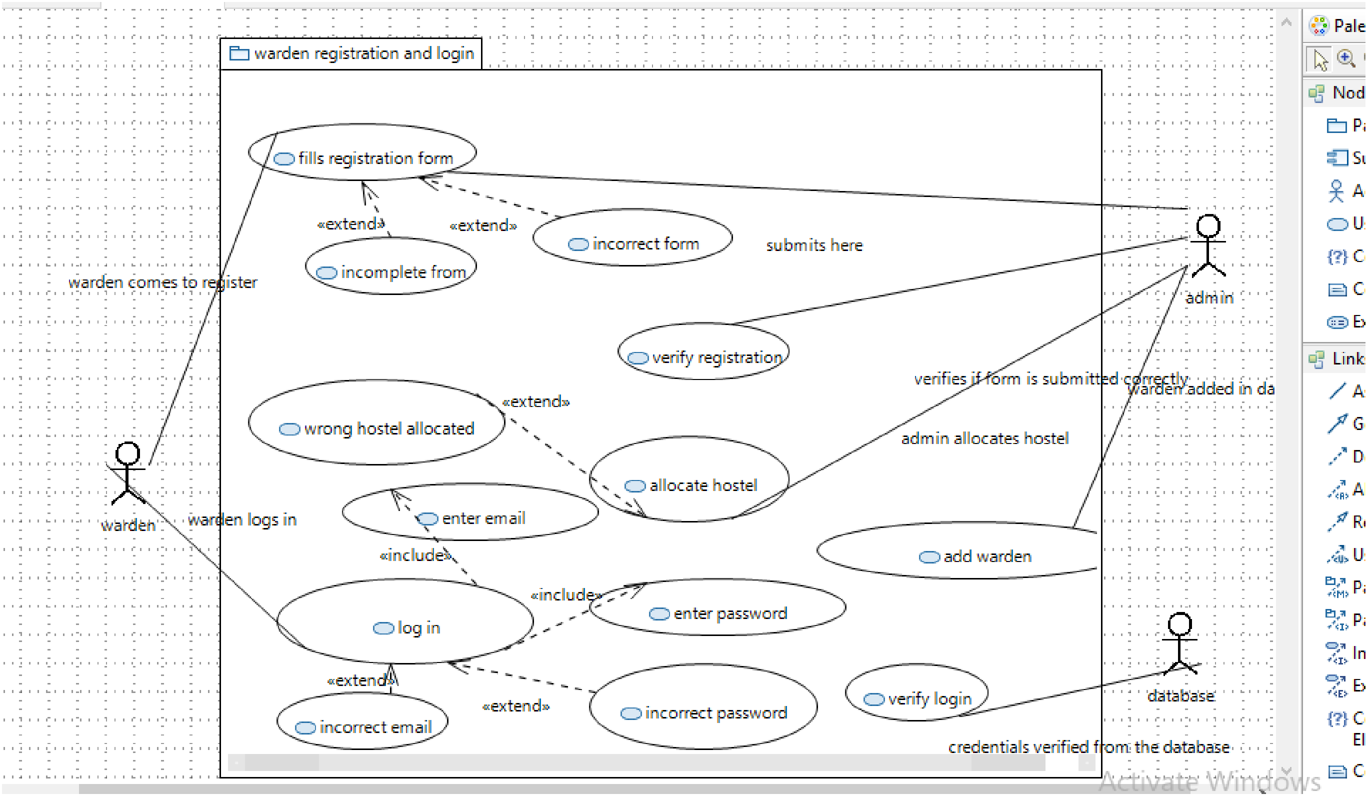
- *Registration*
- *Log in*
- *Add student*
- *Add facility in hostel*
- *Delete facility in hostel*
- *Check student rent status*
- *Generate rent voucher*
- *Validate student payments*

For student

- *Registration*
- *Log in*
- *Avail facility*
- *Discontinue facility*
- *Complain regarding a facility*
- *Check rent voucher*
- *Pay rent*

4.2. Use Cases

4.2.1. Warden registration and login



Id	UC1
USE CASE NAME	Warden registration form
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	The system will add the warden in the system
PRIMARY ACTOR	Warden
SUPPORTING ACTOR	Student, database
PRECONDITION	The user is connected to the internet.
POSTCONDITION	The Warden has successfully registered.
MAIN SUCCESS SCENERIOS	1. The warden fills a form in which he enters his information. 2. The warden submits his form.
EXTENSIONS	*a. At any time, the system crashes.

	1a. The warden does not fill a necessary column. 2a. The warden forgets to submit the form.

Id	UC2
USE CASE NAME	Verify registration form
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	The admin checks if the form is submitted correctly
PRIMARY ACTOR	Admin
SUPPORTING ACTOR	Warden, student
PRECONDITION	The admin staff is connected to the system The warden has filled the registration form
POSTCONDITION	The form is approved and the warden is added
MAIN SUCCESS SCENERIOS	1. The system checks all the details which are provided in the form. 2. The system approves the form. 3. The system allocates a hostel to the warden. 4. The system enters the warden details in the database.
EXTENSIONS	*a. At any time, the system crashes. 1a. The system is unable to detect an invalid form. 2a. the system is unable to approve the correct form. 3a. the system does not provide hostel to the warden. 3b. The system gives a hostel to the warden which already has a warden. 4a. the system enters incomplete information in the database. 4b. the system enters incorrect information in the database.

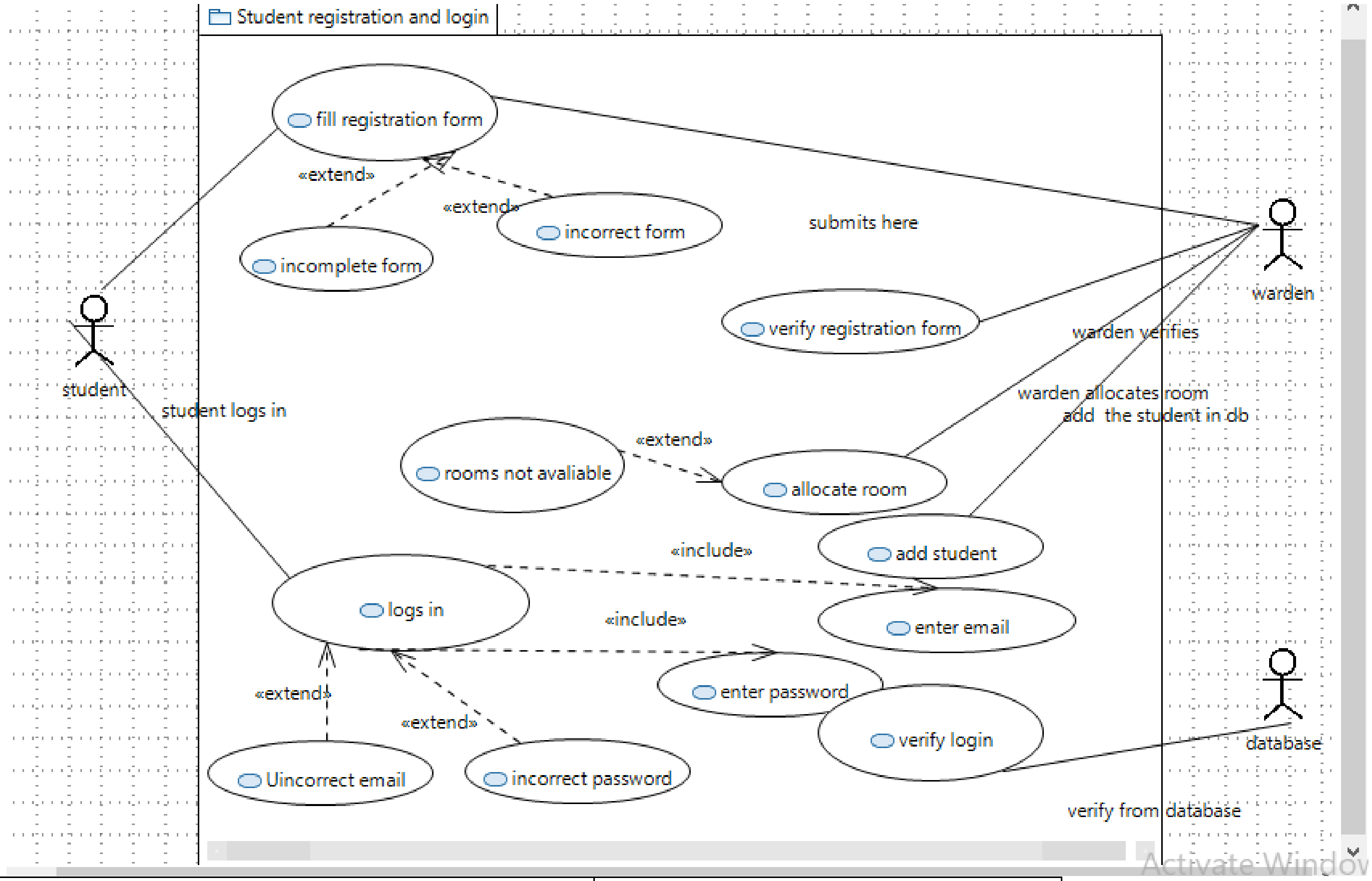
Id	UC3
USE CASE NAME	Warden log in

AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	The warden logs in the system
PRIMARY ACTOR	Warden
SUPPORTING ACTOR	Database, student
PRECONDITION	The user is connected to the internet. The user has already registered in the system as a warden.
POSTCONDITION	The warden successfully logs in.
MAIN SUCCESS SCENERIOS	1 The warden clicks on the sign in button. 2 The warden enters his credentials 3 The warden clicks on the submit button.
EXTENSIONS	*a. At any time, the system crashes. 2a the warden enters invalid credentials. 3a. The warden forgets to click on the submit button.

Id	UC4
USE CASE NAME	Verify warden login
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	The warden’s credentials are verified from the database.
PRIMARY ACTOR	Database
SUPPORTING ACTOR	Admin, student, warden
PRECONDITION	The user is connected to the internet. The warden is registered in the system. The warden attempts to login to the system.
POSTCONDITION	The student is successfully logged In and can see the current status and facilities provided by the system.
MAIN SUCCESS SCENERIOS	1. The warden’s id is verified. 2. The warden’s password is verified. 3. Logged in successfully.
EXTENSIONS	*a. At any time, the system crashes.

	1a. The warden enters invalid email. 2a. the warden enters an invalid password. 3a. even after correct credentials, the warden is not logged in.

4.2.2. Student registration and login



Id	UC5
USE CASE NAME	Student fills registration form
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	The student comes and registers himself in the system so he can live in the hostel.
PRIMARY ACTOR	Student
SUPPORTING ACTOR	Admin, Warden
PRECONDITION	The user is connected to the internet.

POSTCONDITION	The user successfully fills the registration form
MAIN SUCCESS SCENERIOS	1. The student clicks on the registration button. 2. The student enters his information in the form for registration. 3. The student clicks the submit button.
EXTENSIONS	*a. At any time, the system crashes. 2a. The student by mistake does not enter some information, incomplete form gives an error. 2b. The student enters some wrong information.

Id	UC6
USE CASE NAME	Verify student registration
AUTHOR	FABIHA ATIQUE,MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	The warden checks if the student has filled the registration form correctly.
PRIMARY ACTOR	Warden
SUPPORTING ACTOR	Admin, student
PRECONDITION	The user is connected to the internet. The student has filled the form.
POSTCONDITION	The registration has been verified.
MAIN SUCCESS SCENERIOS	1. The warden checks the details entered in the form. 2. The warden approves the registration. 3. The warden allocates a room to the student. 4. The warden adds the student information in the database.
EXTENSIONS	*a. At any time, the system crashes. 1a. The admin fails to check the registration form correctly. 2a. The form is checked and is ok but the warden forgets to approve the registration. 3a. The warden forgets to allocate the room to the student.

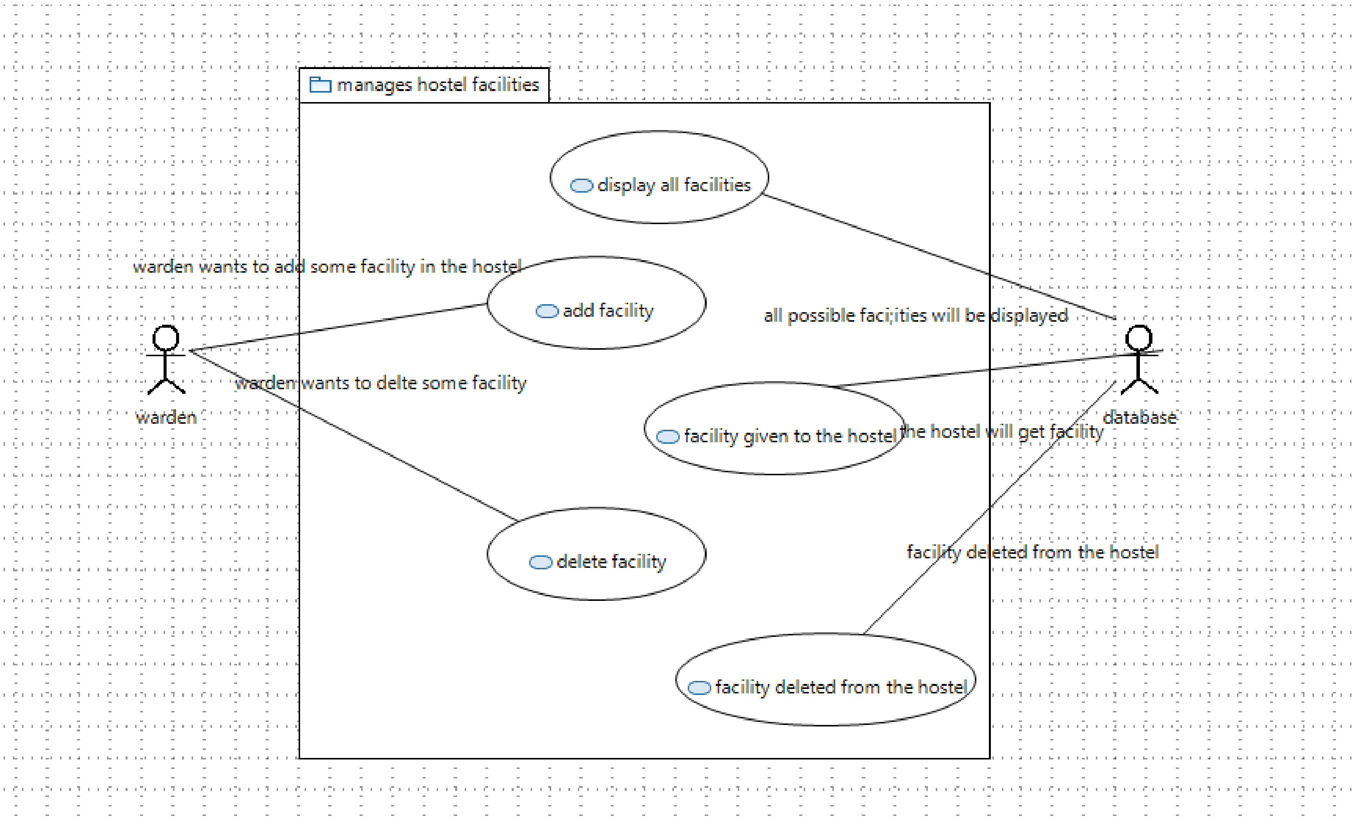
	3b. The warden allocates room which is already taken. 4a. The warden forgets to add the student in the database. 4b. the warden adds incorrect information in the database.

Id	UC7
USE CASE NAME	Student Log in
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	The student logs in to the system to check the facilities provided by the hostel and check status.
PRIMARY ACTOR	Student
SUPPORTING ACTOR	Admin, warden
PRECONDITION	The user is connected to the internet. The student is registered in the system.
POSTCONDITION	The student is successfully logged In and can see the status and facilities provided by the system.
MAIN SUCCESS SCENERIOS	1. The student enters the ID and password to login. 2. The student clicks on the login button.
EXTENSIONS	*a. At any time, the system crashes. 1a. The student enters invalid log in credentials. 1b. the credentials are valid, but the student is unable to log in.

Id	UC8
USE CASE NAME	Verify log in
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	The login credentials are verified from the database.
PRIMARY ACTOR	Database
SUPPORTING ACTOR	Admin, warden, student
PRECONDITION	The user is connected to the internet.

	The student is registered in the system. The user has attempted to login.
POSTCONDITION	The student is successfully logged In and can see the status and facilities provided by the system.
MAIN SUCCESS SCENERIOS	1. The email is verified from the database. 2. The password is verified from the database. 3. If both matches, then the user is allowed to login.
EXTENSIONS	*a. At any time, the system crashes. 1a. the email was incorrect. 2a. the password was invalid.

4.2.3 Manage hostel facilities


manages hostel facilities
display all facilities
warden wants to add some facility in the hostel
add facility
all possible faci;ities will be displayed
warden
warden wants to delte some facility
database
facility given to the hostel
the hostel will get facility
delete facility
facility deleted from the hostel
facility deleted from the hostel

Id	UC9
USE CASE NAME	Display facilities
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022

DESCRIPTION	The system displays the facilities a warden can provide in his hostel.
PRIMARY ACTOR	Database
SUPPORTING ACTOR	Student, Admin, warden
PRECONDITION	The system is connected to the internet
POSTCONDITION	The facilities will be displayed.
MAIN SUCCESS SCENERIOS	1. The facilities button will be present on the front end. 2. The facilities will be displayed on the screen.
EXTENSIONS	*a. At any time, the system crashes. 2a. No facilities are displayed on screen. 2b. incomplete facilities list is displayed on screen.

Id	UC10
USE CASE NAME	Add facilities
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	The warden adds the facilities that can be provided to the students.
PRIMARY ACTOR	Warden
SUPPORTING ACTOR	Admin, student,
PRECONDITION	The user is connected to the internet.
POSTCONDITION	The warden successfully adds the facilities that his branch of hostel will have.
MAIN SUCCESS SCENERIOS	1. The warden will click on the add facilities option. 2. The warden adds the facilities he wants from the list of available.
EXTENSIONS	*a. At any time, the system crashes. 2a. There is not a list of present. 2a. The warden tries to add the facilities that are not present on the table.

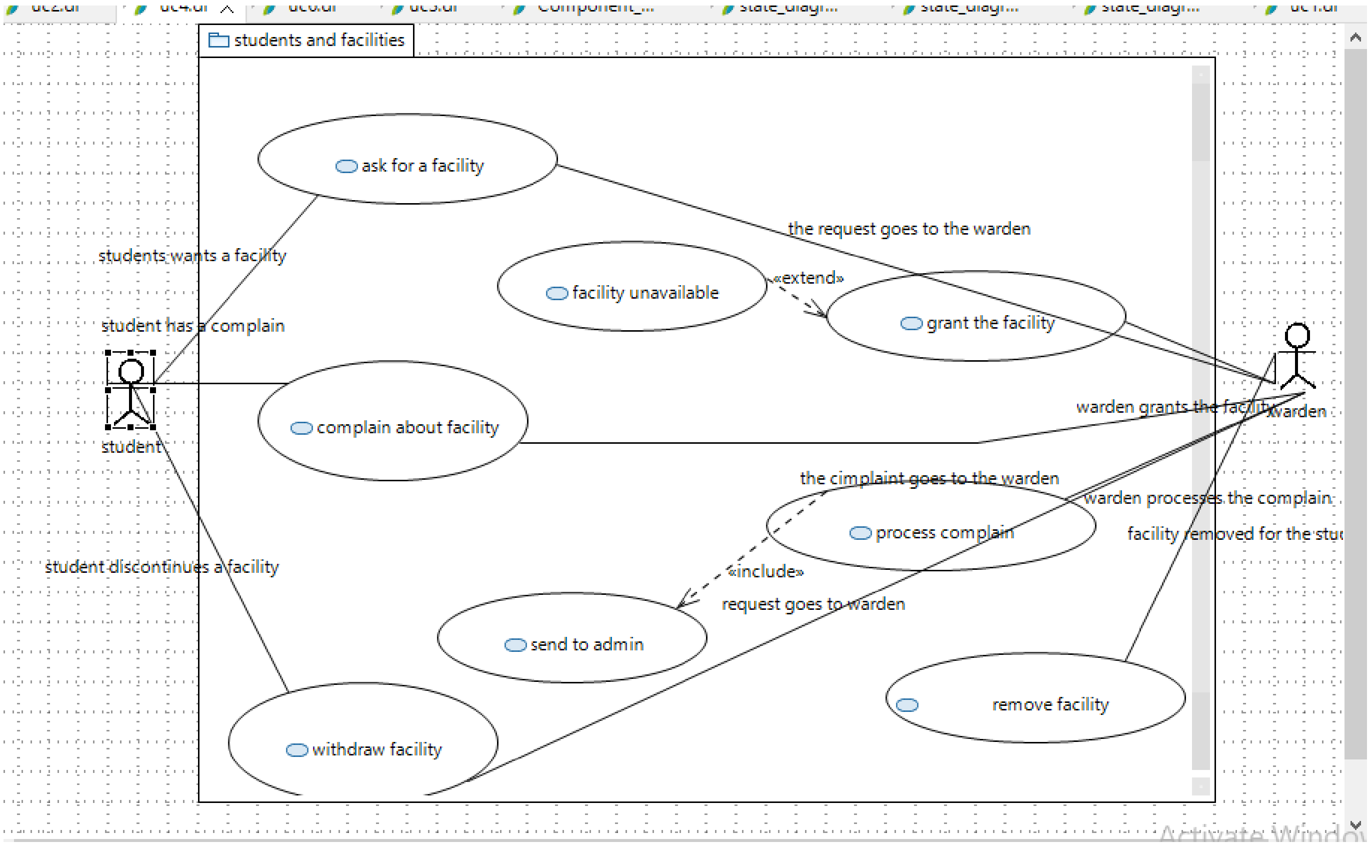
Id	UC11
USE CASE NAME	Drop facilities
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA

LAST REVISED	26/11/2022
DESCRIPTION	The warden drops the facilities that were originally added.
PRIMARY ACTOR	Warden
SUPPORTING ACTOR	Student, Admin
PRECONDITION	The user has already registered in the system. The warden has added a few facilities for the hostel.
POSTCONDITION	The facilities will be deleted.
MAIN SUCCESS SCENERIOS	1. The warden selects the facilities he wants to delete. 2. The facilities will no longer be available for the students.
EXTENSIONS	*a. At any time, the system crashes. 1a. The warden is trying to delete the facility that was never added. 2a. The facility is deleted but still is showing available.

Id	UC12
USE CASE NAME	Facilities updated in the database
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	The facilities added and deleted will be updated in the database.
PRIMARY ACTOR	Database
SUPPORTING ACTOR	Student, Admin, warden
PRECONDITION	The system is connected to the internet. The warden adds and deletes facilities in their hostel.
POSTCONDITION	The data related to facilities will be updated in the database.
MAIN SUCCESS SCENERIOS	1. The facilities that the warden adds are displayed in the database. 2. The facilities deleted will be updated in the database too.
EXTENSIONS	*a. At any time, the system crashes. 1a. the facilities that are added are not shown in the database. 1b. the facilities that are not added are shown in the database.

	2a. The facilities that are deleted are shown in the database. 2b. the facilities that are not deleted will not be shown in the database/.

4.2.4 Student and facilities



Id	UC13
USE CASE NAME	Asks for facility
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	The student asks for a particular facility provided by the hostel
PRIMARY ACTOR	Student
SUPPORTING ACTOR	Warden, admin, database
PRECONDITION	The student has already registered in the system. The student is logged in.
POSTCONDITION	The request is sent to the warden.
MAIN SUCCESS SCENERIOS	1. The student applies for facilities he wants to avail. 2. The warden allows the facilities if available.

EXTENSIONS	*a. At any time, the system crashes. 2a. The student has asked for a facility which is not available in the hostel.

Id	UC14
USE CASE NAME	Grant facility
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	The warden grants the facility asked by the student
PRIMARY ACTOR	Warden
SUPPORTING ACTOR	student, admin, database
PRECONDITION	The warden has registered in the system The warden is logged in The student has asked for a facility
POSTCONDITION	The student successfully gets the facility.
MAIN SUCCESS SCENERIOS	1. The warden checks what facility is asked by the student. 2. The warden checks if that facility is regulated in the hostel. 3. If it is, then it is allowed to the student
EXTENSIONS	*a. At any time, the system crashes. 2a. The facility is not present in the hostel. 3a. The facility is present in the hostel but still the student doesn’t get it.

Id	UC25
USE CASE NAME	Complain about facilities
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	The student can complain about the facilities he is using if there is some problems in it.
PRIMARY ACTOR	Student
SUPPORTING ACTOR	Database, warden
PRECONDITION	The student is connected to the internet. The student is logged in. The student is already using that facility he is about to complain for.
POSTCONDITION	The complaint is successfully submitted
MAIN SUCCESS SCENERIOS	1. The student clicks on the lodge complain button. 2 The student enters the facility about which he wants to complain. 3. The student writes down what is the exact issue with the facility. 4. The student submits the complaint.
EXTENSIONS	*a. At any time, the system crashes. 2a. The student selects an item for complaint which is not currently in his use. 3a. the student writes invalid or incomplete description. 4a. The student forgets to submit the complaint.

Id	UC16
USE CASE NAME	Processes the complain
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	The complain about the facility is
PRIMARY ACTOR	Student
SUPPORTING ACTOR	Warden, admin, database
PRECONDITION	The warden is logged in. The student has complaint about a facility he is using.

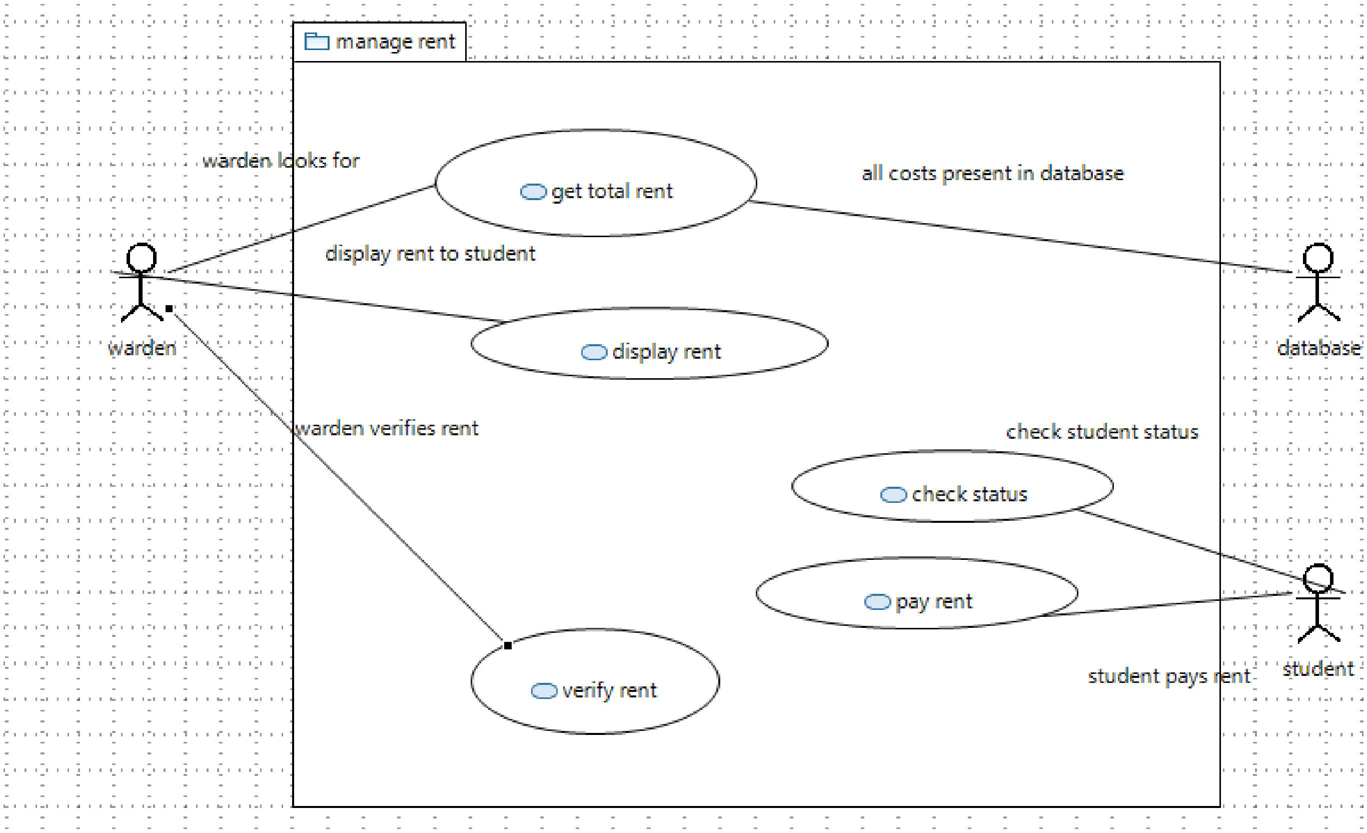
POSTCONDITION	The complaint is forwarded to the admin.
MAIN SUCCESS SCENERIOS	1. The warden receives a complaint about a facility the student was using. 2. The warden reads the complaint. 3. The warden forwards the complaint to the admin.
EXTENSIONS	*a. At any time, the system crashes. 3a. the warden receives the complaint but forgets to forward it to the admin.

Id	UC17
USE CASE NAME	Request to discontinue facility
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	The student wants to delete the facility he was using earlier.
PRIMARY ACTOR	Student
SUPPORTING ACTOR	Database, warden
PRECONDITION	The student is connected to the internet. The student is logged in. The student is already using that facility he wants to discontinue
POSTCONDITION	The facility is discontinued for the student.
MAIN SUCCESS SCENERIOS	1 The student clicks on the facility he wants to discontinue. 2 The student clicks on the discontinues button.
EXTENSIONS	*a. At any time, the system crashes. 2a. the student clicks on a facility he is not even using.

Id	UC18
USE CASE NAME	Discontinue facility
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	The warden discontinues the facility for a student.
PRIMARY ACTOR	Warden

SUPPORTING ACTOR	Database, student, admin
PRECONDITION	The warden is connected to the internet. The student has requested to discontinue the facility.
POSTCONDITION	The facility is discontinued for the student
MAIN SUCCESS SCENERIOS	1 The warden sees the student request to discontinue the facility. 2 The warden updates in the database \\ 3 The facility is removed for that student.
EXTENSIONS	*a. At any time, the system crashes. 2a. The warden discontinues the facilities for some other student by mistake. 2b. the warden forgets to discontinue the facility for that student.

4.2.5 Manages student fees



Id	UC19
USE CASE NAME	Get total rent
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	Warden checks the room's rent plus the facility rent for each student from the database.
PRIMARY ACTOR	Warden
SUPPORTING ACTOR	Student, database
PRECONDITION	The warden is connected to the internet. The warden is logged in.
POSTCONDITION	The total amount is checked from the database.
MAIN SUCCESS SCENERIOS	1. The warden checks the database. 2. The warden first sees the hostel's rent for the student. 3. The warden then sees the rent of all the facilities the student is using. 4. The warden adds up the rent.
EXTENSIONS	*a. At any time, the system crashes. 2a. the warden reads incorrect rent from the database. 3a. the warden reads incorrect facility rent from the database. 4a. the rent is added incorrectly.

Id	UC19
USE CASE NAME	Display rent
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	The warden displays the total rent on screen.
PRIMARY ACTOR	Warden
SUPPORTING ACTOR	Database, student
PRECONDITION	The warden is registered in the system. The warden is logged in. The warden has calculated the total bill.

POSTCONDITION	The warden has printed the rent on the screen.
MAIN SUCCESS SCENERIOS	1. The warden displays the rent calculated on the screen.
EXTENSIONS	*a. At any time, the system crashes. 1a. the wrong rent is displayed on screen. 1a. no bill is displayed on screen.

Id	UC20
USE CASE NAME	Check status
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	The student will check his status to view his fee status.
PRIMARY ACTOR	Student
SUPPORTING ACTOR	Database, warden
PRECONDITION	The student is connected to the internet. The student is logged in.
POSTCONDITION	The student is able to view his rent
MAIN SUCCESS SCENERIOS	2. The student will view on the fee details.
EXTENSIONS	*a. At any time, the system crashes.

Id	UC21
USE CASE NAME	Pay rent
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	The student will pay the fees manually.
PRIMARY ACTOR	Student
SUPPORTING ACTOR	Database, warden
PRECONDITION	The student is connected to the internet. The student is logged in. The student has seen the rent he needs to pay/
POSTCONDITION	The student pays the fees.
MAIN SUCCESS SCENERIOS	1. The student pays the fees manually.

EXTENSIONS	*a. At any time, the system crashes. 1a. the student does not pay the fees. 1b. the student pays the wrong fee.

Id	UC22
USE CASE NAME	Verify rent
AUTHOR	FABIHA ATIQUE, MOHAMMAD USAMA
LAST REVISED	26/11/2022
DESCRIPTION	The warden will verify whether the student has paid the rent or not
PRIMARY ACTOR	Warden
SUPPORTING ACTOR	Database, student
PRECONDITION	The warden is connected to the internet. The warden is logged in The student has paid the fees.
POSTCONDITION	The warden has updated the database
MAIN SUCCESS SCENERIOS	1. The warden checks manually if the student has paid the fees or not 2. The warden updates the database.
EXTENSIONS	*a. At any time, the system crashes. 1a. the warden makes some mistake in checking. 2a. the fee is paid but warden forgets to update in the database.

5. Non-functional Requirements

5.1. Performance Requirements

The system requires concurrency to manage both the requests of the student and the warden. In the system.

5.2. Safety Requirements

As it's a public hostel, the system requires us to ensure the safety of the users' data in our database. There should be complete confidentiality and no data leaks are affordable.

5.3. Security Requirements

For any user to enter the system. Login credentials are needed to ensure the security of our users. We as developers need to ensure that the data is secure with us.

5.4. User Documentation

The user will be provided with a complete user guide to use and manage our system at the time of deployment.

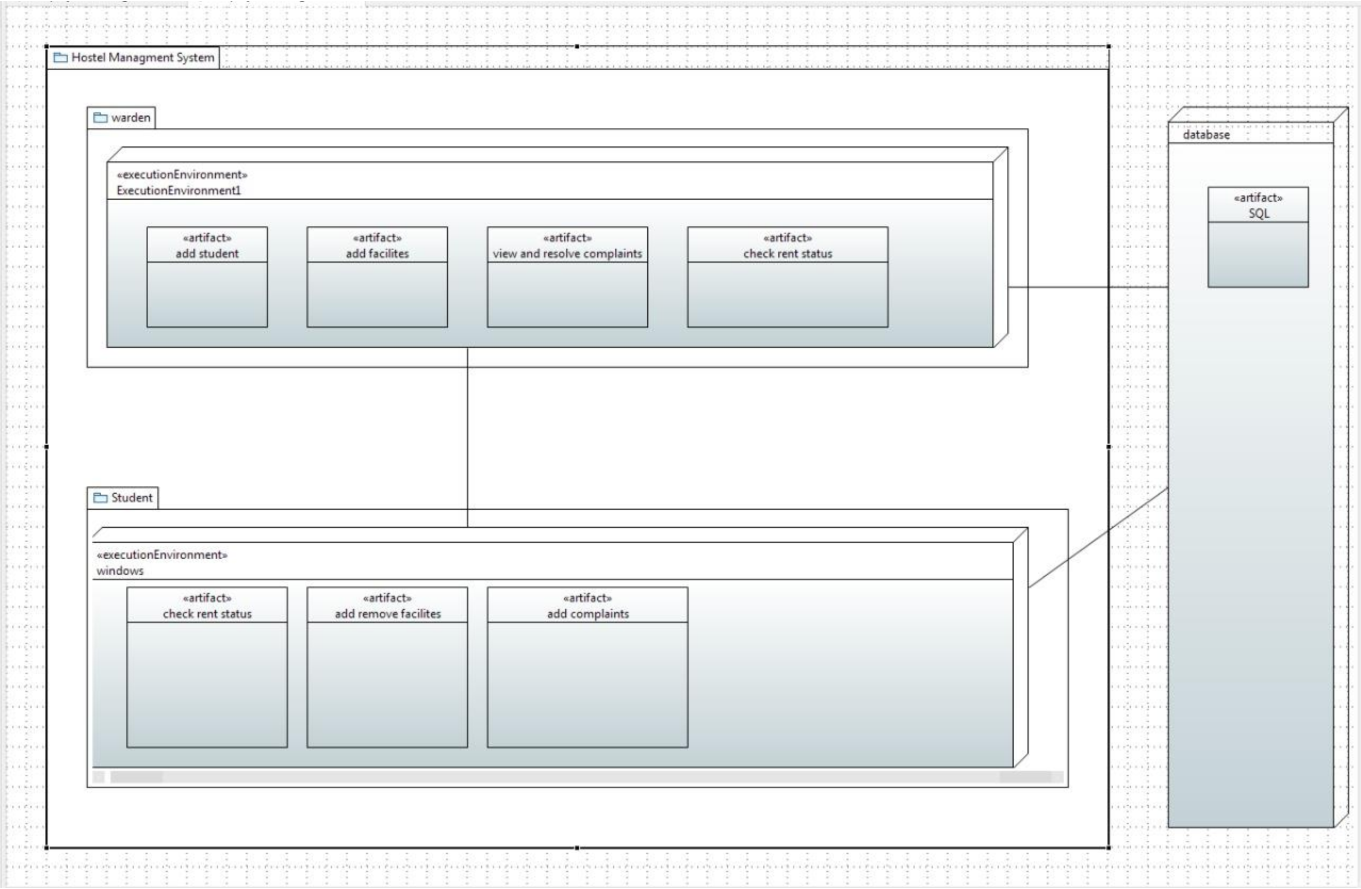
SDS

6. System Architecture

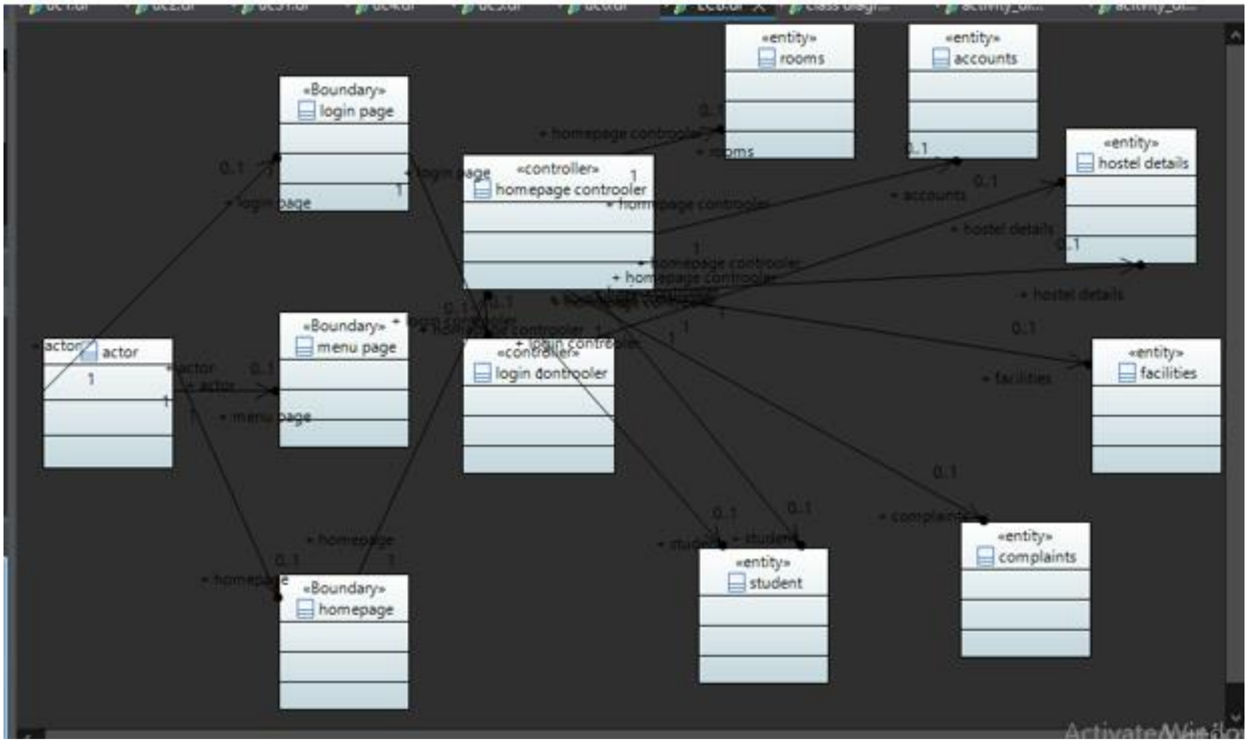
The hostel management system is done by both maintaining the frontend and the backend. There are two users in our system, the student, and the warden, for both of these users we need to maintain the whole system.

6.1. System Level Architecture

Deployment diagram



6.2. Software Architecture

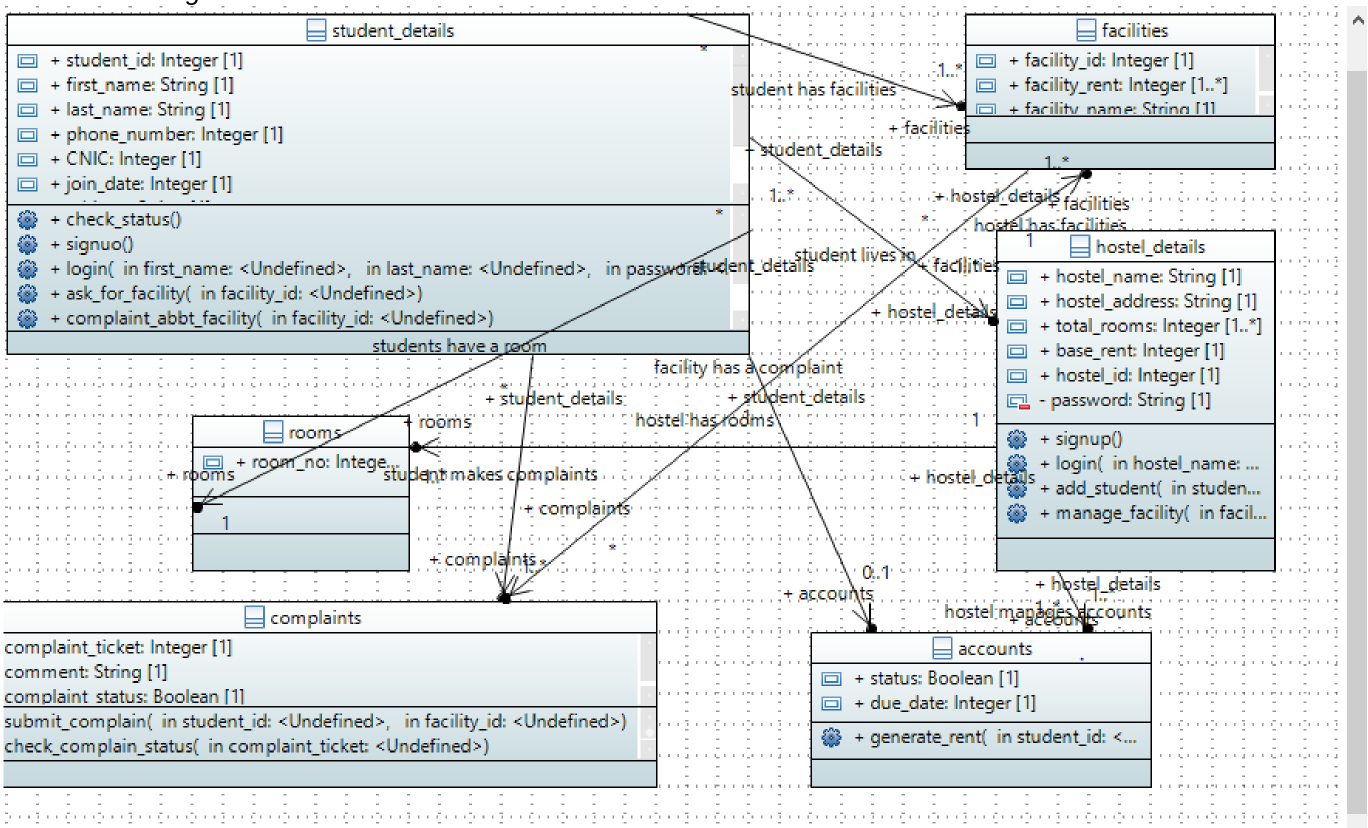


7. Design Strategy

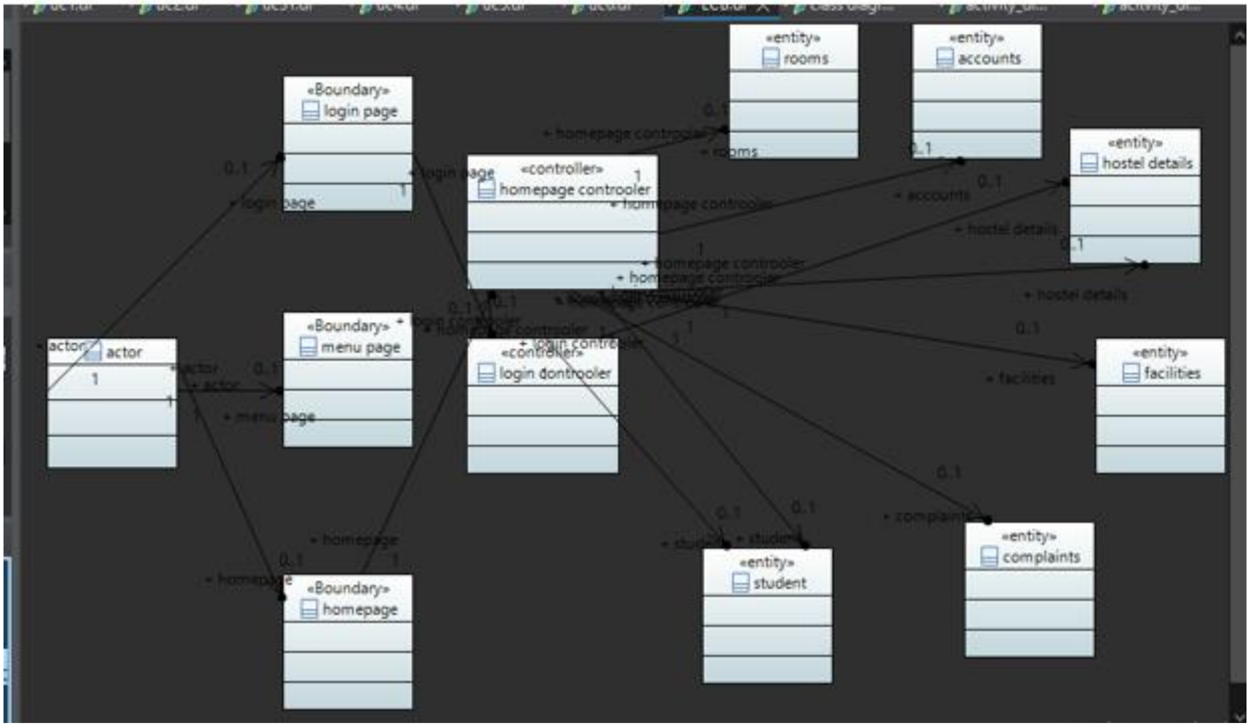
Beginning with the structure of the hostel management, the plan is to provide a portal for both the student and the warden to manage the hostel in a convenient way. To ensure that our system t
In our design phase, we have

8. Detailed System Design

Class diagram


student_details
+ student_id: Integer [1]
+ first_name: String [1]
+ last_name: String [1]
+ phone_number: Integer [1]
+ CNIC: Integer [1]
+ join_date: Integer [1]
+ check_status()
+ signuo()
+ login(in first_name: <Undefined>, in last_name: <Undefined>, in password...
+ ask_for_facility(in facility_id: <Undefined>)
+ complaint_abbt_facility(in facility_id: <Undefined>)
students have a room
facilities
+ facility_id: Integer [1]
+ facility_rent: Integer [1..*]
+ facility_name: String [1]
hostel_details
+ hostel_name: String [1]
+ hostel_address: String [1]
+ total_rooms: Integer [1..*]
+ base_rent: Integer [1]
+ hostel_id: Integer [1]
- password: String [1]
+ signup()
+ login(in hostel_name: ...
+ add_student(in studen...
+ manage_facility(in facil...
rooms
+ room_no: Intege...
complaints
complaint_ticket: Integer [1]
comment: String [1]
complaint_status: Boolean [1]
submit_complain(in student_id: <Undefined>, in facility_id: <Undefined>)
check_complain_status(in complaint_ticket: <Undefined>)
accounts
+ status: Boolean [1]
+ due_date: Integer [1]
+ generate_rent(in student_id: <...
student has facilities
+ facilities
+ student_details
student lives in
facility has a complaint
hostel has rooms
hostel has facilities
student makes complaints
hostel manages accounts
+ hostel_details
+ rooms
+ complaints
+ accounts
1..*
1
0..1
*

ECB class diagram

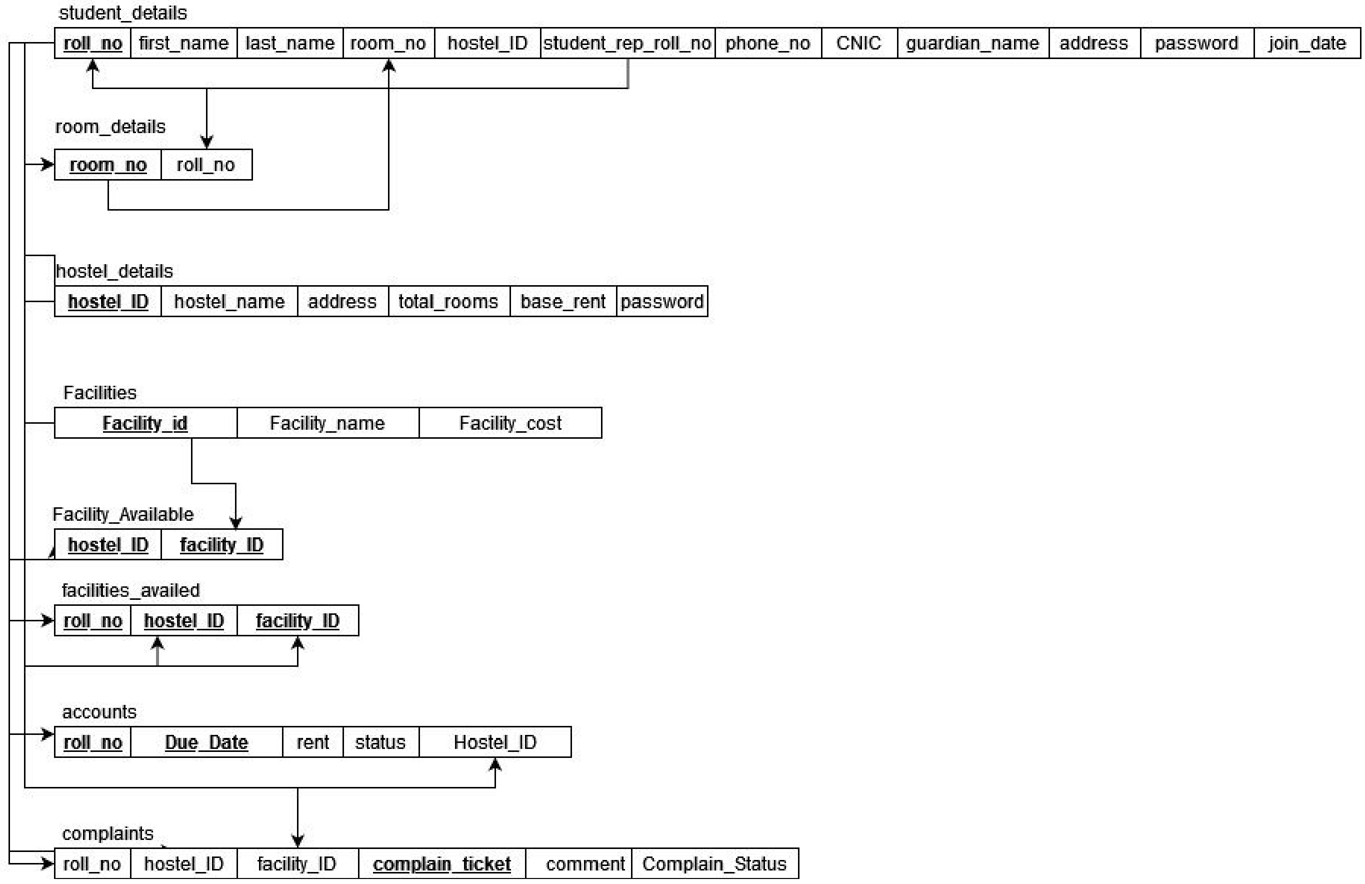


```
public class student_details {
                          /**
                           * 
                           */
                          public String first_name;
                          /**
                           * 
                           */
                          public String last_name;
                          /**
                           * 
                           */
                          public int phone_number;
                          /**
                           * 
                           */
                          public int CNIC;
                          /**
                           * 
                           */
                          public String guardian_name;
                          /**
                           * 
                           */
                          public String address;
                          /**
                           * 
                           */
                          public int join_date;
                          /**
                           * 
                           */
                          private String password;
                          /**
                           * 
                           */
                          public int student_id;
                          /**
                           * 
                           */
                          public hostel_details[] hostel_details;
                          /**
                           * 
                           */
                          public facilities[] facilities;
                          /**
                           * 
                           */
                          public complaints[] complaints;
                          /**
                           * 
                           */
                          public rooms rooms;
                          /**
                           * 
                           */
```

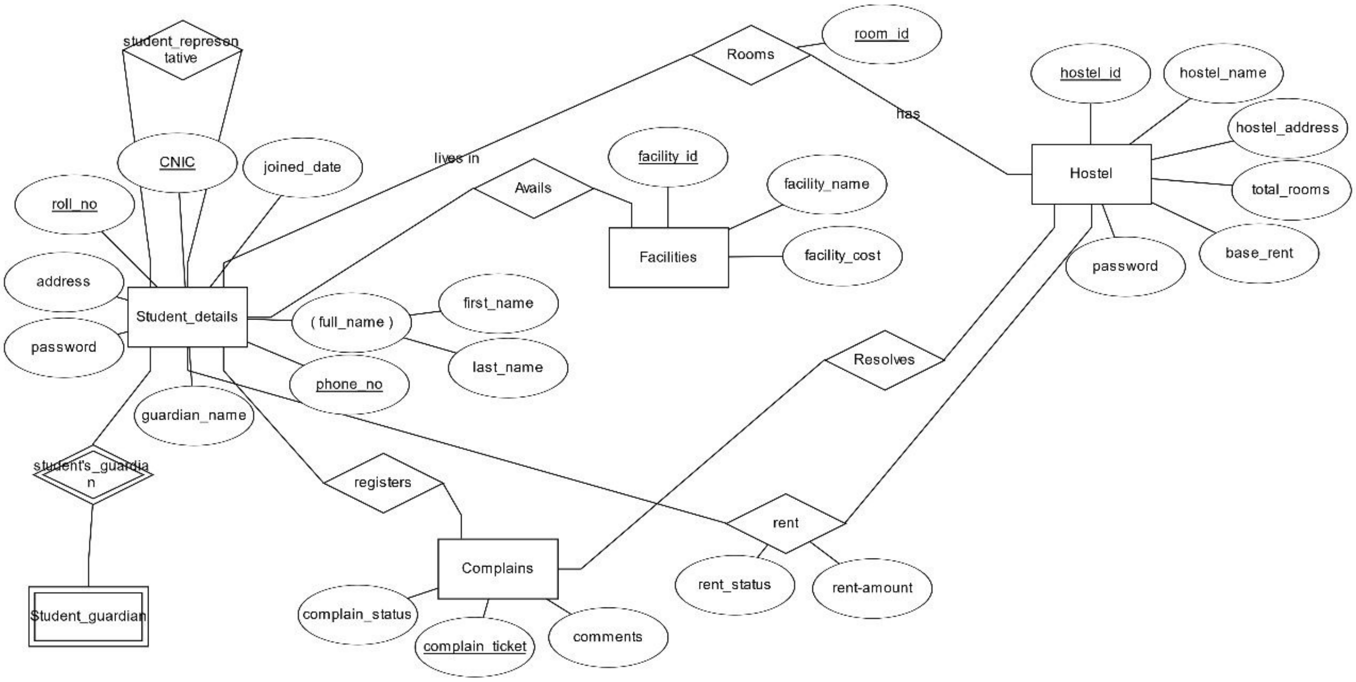
```
                                public accounts accounts;
        /**
         * 
         */
        public void signuo() {
        }
        /**
         * 
         * @param first_name
         * @param last_name
         * @param password
         */
        public void login(undef first_name, undef last_name, undef password) {
        }
        /**
         * 
         */
        public void check_status() {
        }
        /**
         * 
         * @param facility_id
         */
        public void ask_for_facility(undef facility_id) {
        }
        /**
         * 
         * @param facility_id
         */
        public void complaint_abbt_facility(undef facility_id) {
        }
        /**
         * 
         * @param facility_id
         */
        public void withdraw_facility(undef facility_id) {
        }
        /**
         * 
         */
        public void check_status() {
        }
}


public class rooms {
                                /**
                                 * 
                                 */
                                public int room_no;
}
```

```
public class hostel_details {
                          /**
                           * 
                           */
                          public int hostel_id;
                          /**
                           * 
                           */
                          public String hostel_name;
                          /**
                           * 
                           */
                          public String hostel_address;
                          /**
                           * 
                           */
                          public int[] total_rooms;
                          /**
                           * 
                           */
                          public int base_rent;
                          /**
                           * 
                           */
                          private String password;
                          /**
                           * 
                           */
                          public facilities[] facilities;
                          /**
                           * 
                           */
                          public rooms[] rooms;
                          /**
                           * 
                           */
                          public accounts[] accounts;
      /**
       * 
       */
      public void signup() {
      }
      /**
       * 
       * @param hostel_name
       * @param password
       */
      public void login(undef hostel_name, undef password) {
      }
      /**
       * 
       * @param student_id
       */
      public void add_student(undef student_id) {
       }
       /**
        *
        * @param facility_id
        */
       public void manage_facility(undef facility_id) {
       }
}


public class facilities {
                          /**
                           *
                           */
                          public int facility_id;
                          /**
                           *
                           */
                          public String facility_name;
                          /**
                           *
                           */
                          public int[] facility_rent;
                          /**
                           *
                           */
                          public complaints[] complaints;
}


public class complaints {
                          /**
                           *
                           */
                          public int complaint_ticket;
                          /**
                           *
                           */
                          public String comment;
                          /**
                           *
                           */
                          public boolean complaint_status;
       /**
        *
        * @param student_id
        * @param facility_id
        */
       public void submit_complain(undef student_id, undef facility_id) {
       }
       /**
        *
        * @param complaint_ticket
        */
        public void check_complain_status(undef complaint_ticket) {
        }
}


public class accounts {
                                /**
                                 *
                                 */
                                public int due_date;
                                /**
                                 *
                                 */
                                public boolean status;
        /**
         *
         * @param student_id
         */
        public void generate_rent(undef student_id) {
        }
}
```

8.1. Database Design


student_details
roll_no | first_name | last_name | room_no | hostel_ID | student_rep_roll_no | phone_no | CNIC | guardian_name | address | password | join_date
room_details
room_no | roll_no
hostel_details
hostel_ID | hostel_name | address | total_rooms | base_rent | password
Facilities
Facility_id | Facility_name | Facility_cost
Facility_Available
hostel_ID | facility_ID
facilities_availed
roll_no | hostel_ID | facility_ID
accounts
roll_no | Due_Date | rent | status | Hostel_ID
complaints
roll_no | hostel_ID | facility_ID | complain_ticket | comment | Complain_Status

8.1.1. ER Diagram



8.1.2. Data Dictionary

[The convention recommended for writing the data dictionary is as follows.]

8.1.2.1 Data 1

Student

The student logs in and does multiple jobs like managing the facilities he is using and see his rent and amount he has to pay.

8.1.2.2. Data 2

Hostel details

The hostel details are managed by the warden; the warden manages the facilities for the whole hostel and checks the rent status for each student and the issue them the fee challan so that the student can pay the fee.

8.1.2.2 data 3

rooms

the rooms are the essential data is to be present in any hostel. The room can be issued by the warden to a student.

8.1.2.4 data 4

Facilities

The facilities are managed by the students as well as the warden. The student will manage for himself only while the warden will be managing for the whole hostel

8.1.2.5 data 5

Accounts

Accounts data is important to keep track of what charges are due on each student. Since each student is paying different fees because they are availing different facilities. It is also important to know which student has paid the fees and which has not.

8.1.2.6 data 6

Complaints

Complaints data is important to keep, the complaints regarding the facilities are stored and managed there.

8.1.2.7 data 7

Facility_available

The facilities available are stored in a table so that the warden knows which facilities are allowed by the admin.

8.1.2.8 data 8

Facility _availed

The facilities which are available by a hostel are kept in a separate table.

< Data 1>	
Name	*Student*
Alias	*Not applicable*
Where-used/how- used	*The student is an entity class. The student data will be provided by the student and inserted in the database by the admin after verified by the warden.*
Content description	*The student data is to maintain the records of each student living in our hostel.*

Column Name	**Description**	**Type**	**Length**	**Null able**	**Default Value**	**Key Type**
Roll_no	*The unique ID assigned to each student*	*Integer*	*Standard*	*No]*	*1 greater than the roll_no of previous student*	*PK*
First_name	*The first name of the student*	*String*	*20 characters*	*No*	*Abc*	*None*
Last_name	The last name of the student	String	20 characters	No	Abc	None
Room_no	The room number assigned to the student	Integer	2 digits	No	None	FK
Hostel_ID	The hostel ID of the hostel	Integer	2 digits	No	1 greater than the previous hostel's Id	FK
Phonr_no	The phone number of the student	Long int	Standard	No	None	None
CNIC	The student cnic number	Integer	standard	No	None	Unique
Guardian_name	The name of the guardian of the student	String	20 characters	No	Abc	None
Address	The home address of the student	String	20 characters	No	None	None
Password	The password of the student	String	standard	No	None	None
Voin date	The date on which the student joined	Date	Standard	No	none	None

< Data 2>	
Name	*Hostel_details*
Alias	*Not applicable*
Where-used/how- used	*The hostel details is an entity class, it will hold the information related to the hostel, it will be managed by the warden*
Content description	*The hostel_details will hold all the details related to the hostel.*

Column Name	**Description**	**Type**	**Length**	**Null able**	**Default Value**	**Key Type**
Hostel_ID	*The unique ID assigned to each hostel*	*Integer*	*Standard*	*No*	*1 greater than the Id assigned to the previous hostel*	*PK*
Hostel_name	*The name of the hostel*	*String*	*20 characters*	*No*	*Abc*	*None*
Address	The address of the hostel	String	20 characters	No	None	None
Total_rooms	The number of rooms in a hostel	Integer	2 digits	No	None	None
Base_rent	The rent of the hostel	Integer	Standard	No	None	None
Password	The password of the warden to enter in the system	String	Standard	No	None	None

< Data 3>	
Name	*Rooms*
Alias	*Not applicable*

Where-used/how- used	*The rooms are an entity class in our system. It contains all the information about the rooms in the hostel*
Content description	*The rooms are specific to each hostel,, these are managed by the warden and the admin.*

Column Name	***Description***	***Type***	***Length***	***Null able***	***Default Value***	***Key Type***
Room_no	*The room number assigned to every room*	*Intege r*	*Standard*	*No*	*1 greater than the Id assigned to the previous room*	*PK*
Roll_no	*The student roll number*	*intege r*	*Standard*	*No*	*None*	*FK*

< Data 4>	
Name	*Facilities*
Alias	*Not applicable*
Where-used/how- used	*The facilities are the entity class in our system, these classes are maintained by the student and the warden.*
Content description	*The facilities table consists of the number of facilities and their names*

Column Name	**Description**	**Type**	**Length**	**Null able**	**Default Value**	**Key Type**
Facility_ID	*The unique id assigned to each facility*	*Integer*	*Standard*	*No*	*1 greater than the previous facility in the table*	*PK*
Facility_name	*The name of the facility*	*String*	*Standard*	*No*	*None*	*PK*
Cost	*The cost of that facility*	*integer*	*standard*	*No*	*None*	*None*

< Data 5>	
Name	*Accounts*
Alias	*Not applicable*
Where-used/how- used	*The accounts table is used to save the accounts details and rent related information of the students.*
Content description	*The entity contains the rent details of the students*

Column Name	**Description**	**Type**	**Length**	**Null able**	**Default Value**	**Key Type**
Roll_no	*The toll number of a student*	*Integer*	*Standard*	*No*	*NONE*	*FK, PK*
Hostel_id	*The hostel id*	*integer*	*Standard*	*No*	*None*	*FK*
Due_date	*The date when the rent is due*	*Date*	*standard*	*No*	*None*	*None*
Rent	*The amount of rent to be paid*	*Integer*	*Standard*	*no*	*None*	*None*
status	*The status of the rent being paid or not*	*Boolean*	*Standard*	*no*	*None*	*None*

< Data 6>

Name	*Complaints*
Alias	*Not applicable*
Where-used/how- used	*The entity class complaints is there to maintain the records of all the complaints regarding a facility. These complaints are registered by the students*
Content description	*All the details regarding complaints are stored there*

Column Name	**Description**	**Type**	**Length**	**Null able**	**Default Value**	**Key Type**
Roll_no	*The toll number of a student*	*Intege r*	*Standard*	*No*	*NONE*	*FK*
Hostel_id	*The hostel id*	*integer*	*Standard*	*No*	*None*	*FK*
Facility_id	*The id of the facility the student is using*	*Integer*	*standard*	*No*	*None*	*FK*
Complain _ticket	*The id of the complain*	*Integer*	*Standard*	*no*	*None*	*PK*
Comment	*The comment left by the student for the facility*	*String*	*Standard*	*no*	*None*	*None*

Status	*The status of the complaint being registered*	*Boolean*	*Standard*	*No*	*None*	*None*

< Data 7>	
Name	*Facilities_available*
Alias	*Not applicable*
Where-used/how- used	*This class is used by the warden to know what facilities are available in the list, only these facilities can be added in a hostel.*
Content description	*All the details related to a facility is present in the tale.*

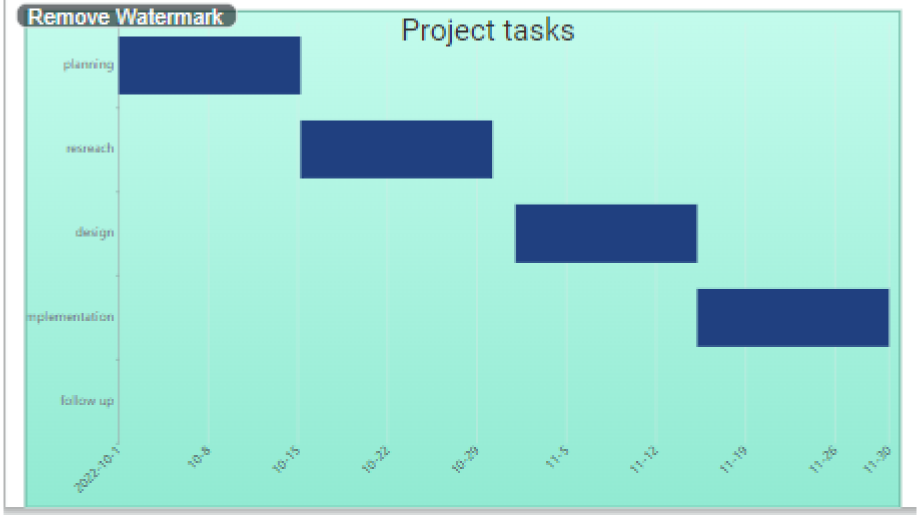
Column Name	***Description***	***Type***	***Length***	***Null able***	***Default Value***	***Key Type***
Hostel_id	*The id of the hostel which is using the facility*	*Integer*	*Standard*	*No*	*NONE*	*FK. PK*
Facility_id	*The Id of the facility.*	*integer*	*Standard*	*No*	*None*	*FK ,PK*

< Data 8>	
Name	Facilities_avaied
Alias	Not applicable
Where-used/how- used	This class is used to view what facilities are availed by the hostel.
Content description	All the details related to a facility is present in the tale.

Column Name	Description	Type	Length	Null able	Default Value	Key Type
Roll_no	The toll number of a student	Intege r	Standard	No	NONE	FK,PK
Hostel_id	The hostel id	integer	Standard	No	None	FK, PK
Facility_id	The id of the facility the student is using	Integer	standard	No	None	FK,PK

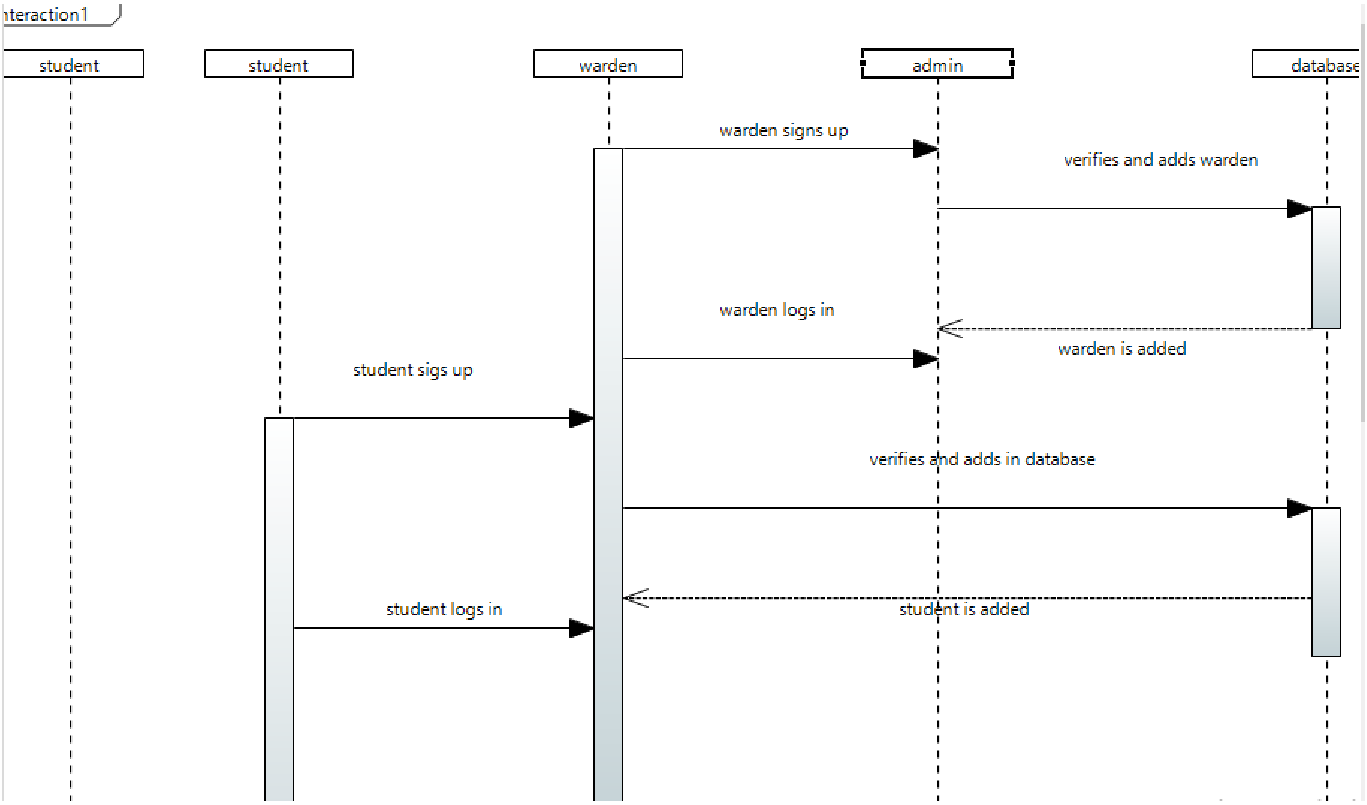
9. Application Design

The application was designed in the following stages.

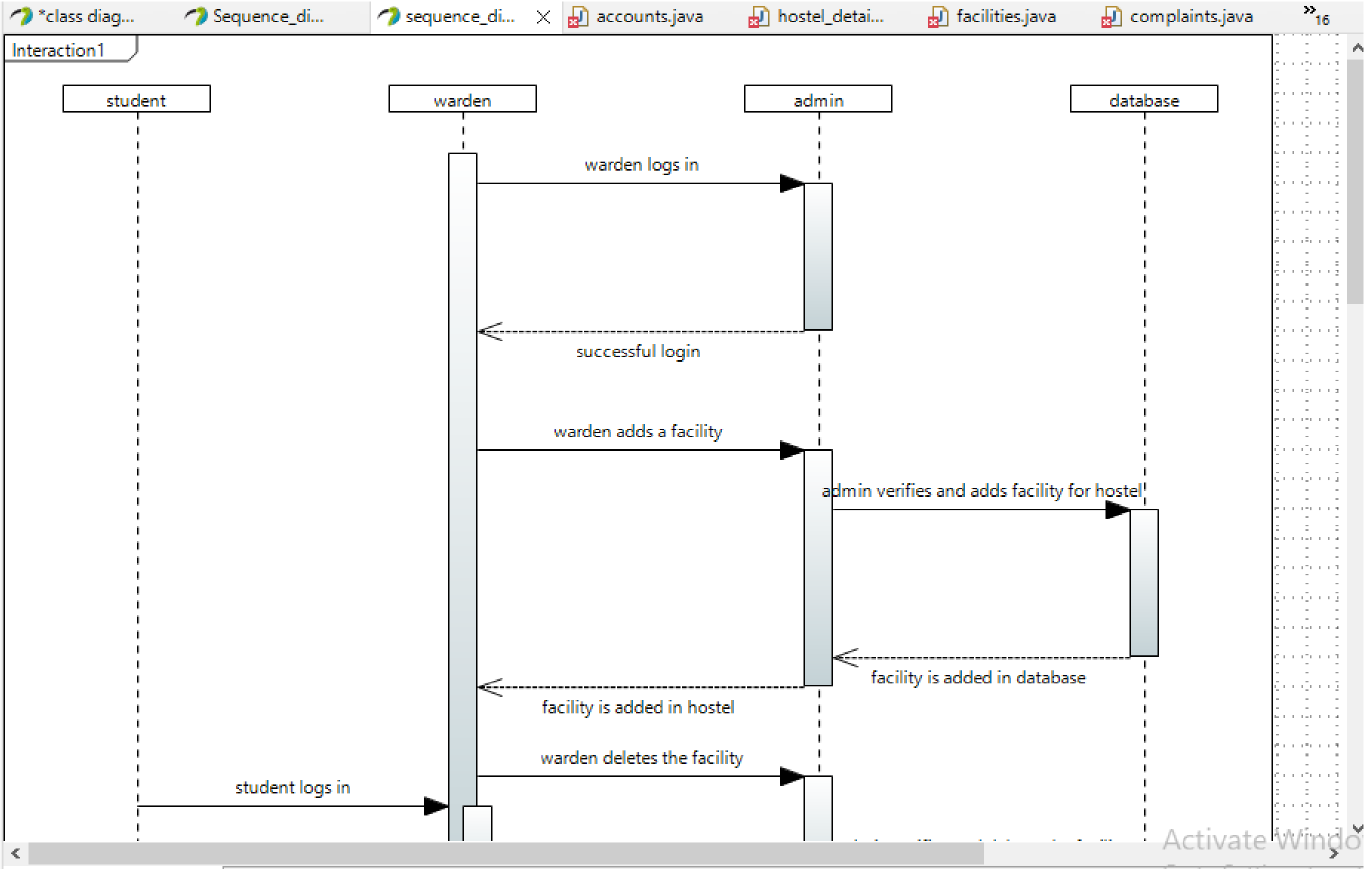


9.1.2. Sequence Diagram

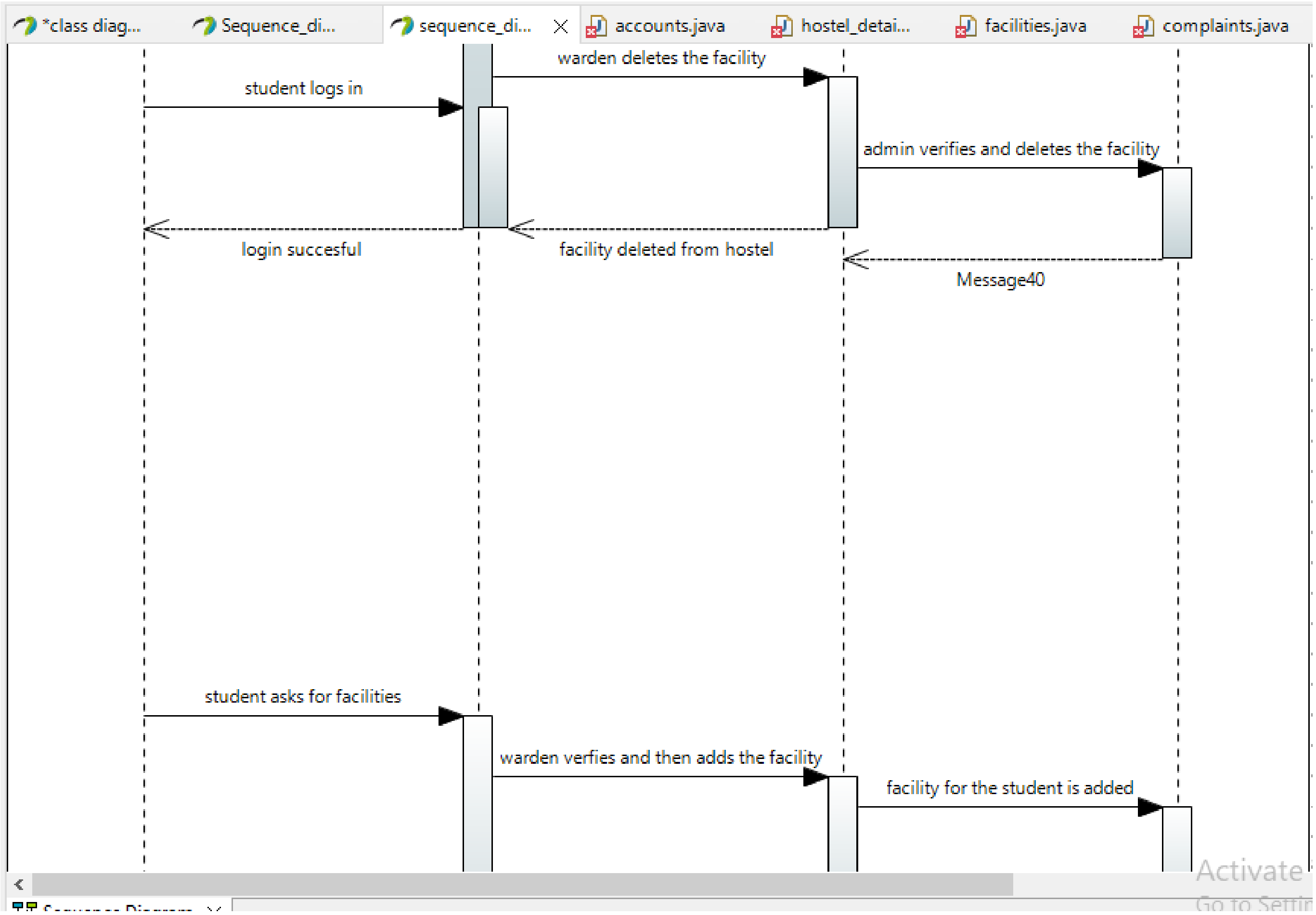
9.1.2.1 <Sequence Diagram 1>

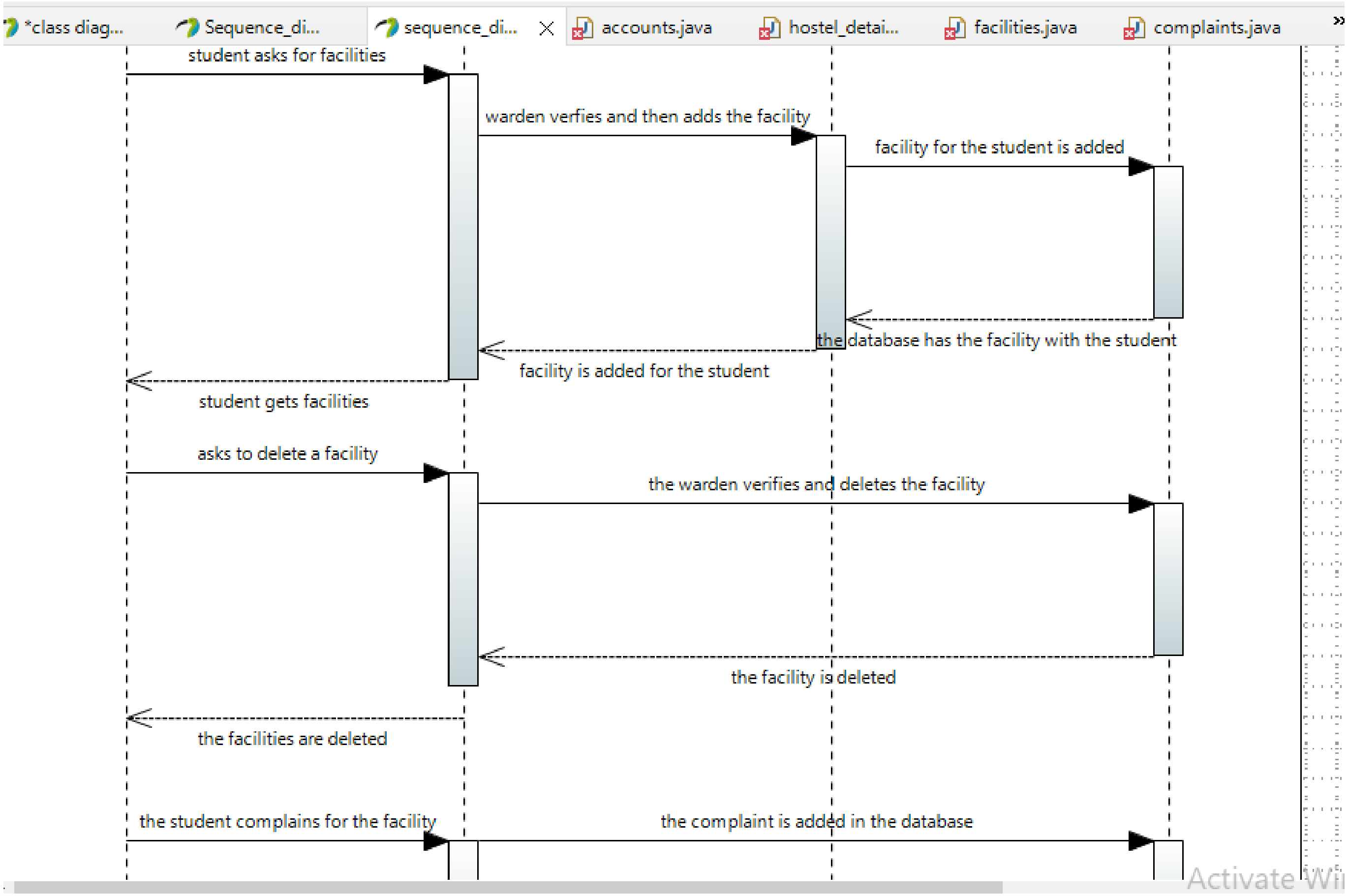


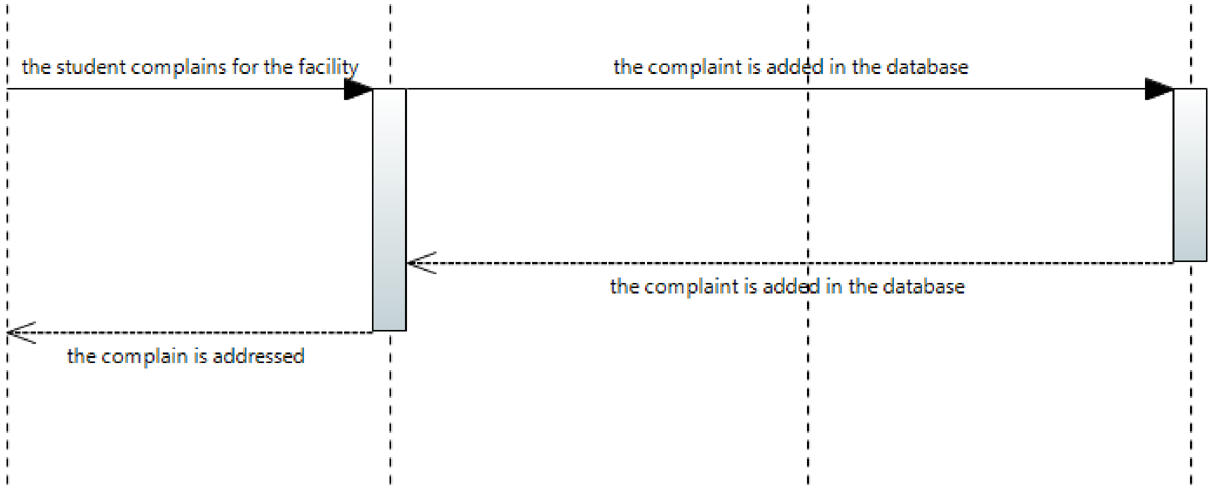
9.1.2.2 <Sequence Diagram 2>


*class diag...
Sequence_di...
sequence_di...
accounts.java
hostel_detai...
facilities.java
complaints.java
16
Interaction1
student
warden
admin
database
warden logs in
successful login
warden adds a facility
admin verifies and adds facility for hostel
facility is added in database
facility is added in hostel
warden deletes the facility
student logs in

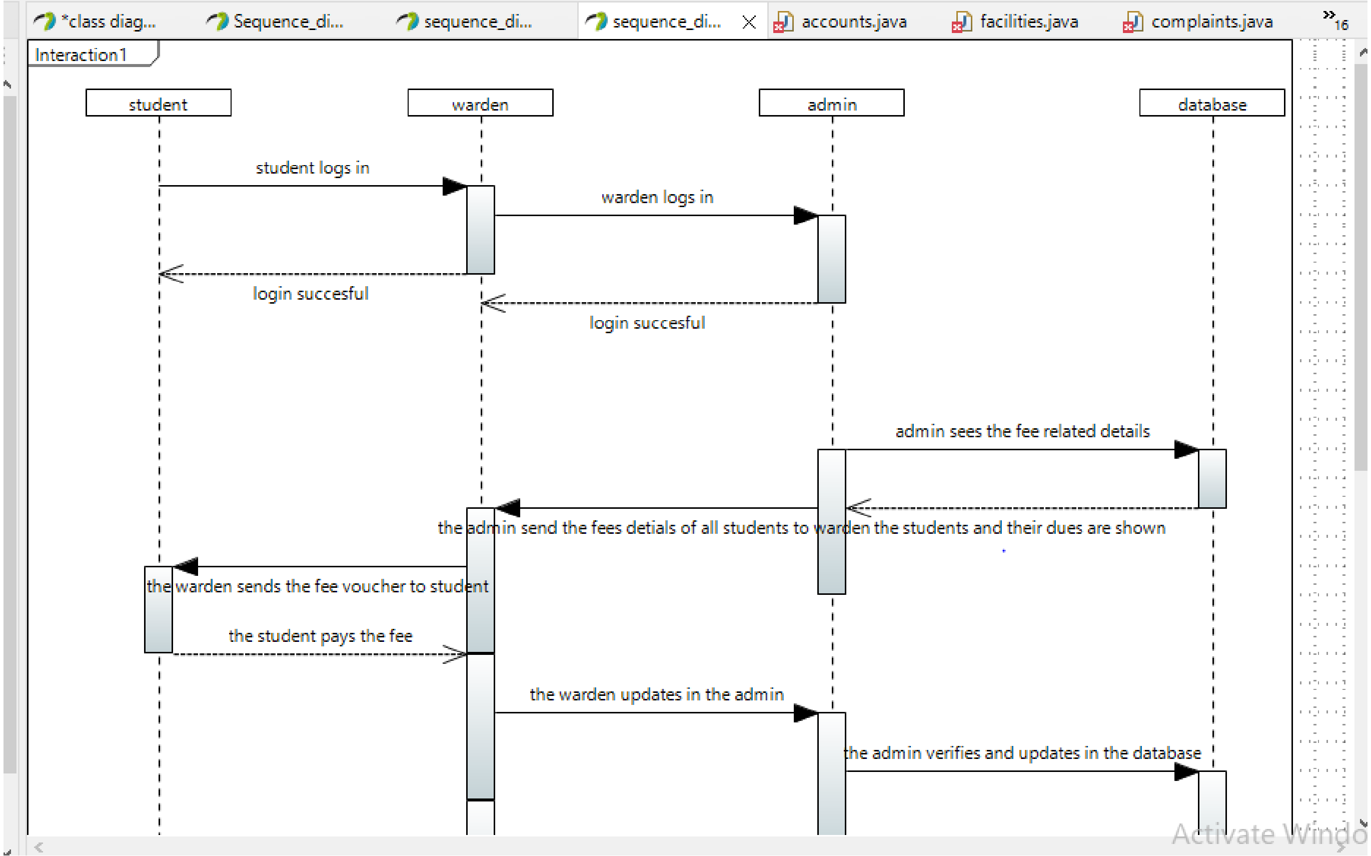
.

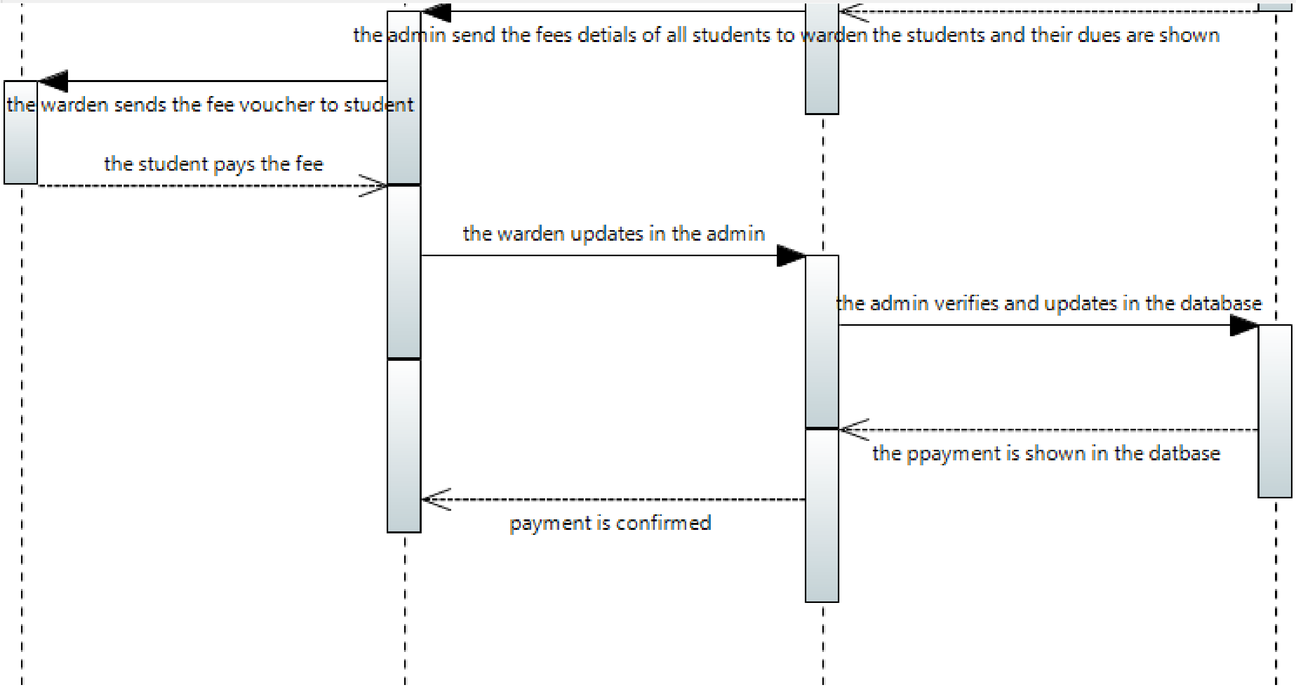

*class diag...
Sequence_di...
sequence_di...
accounts.java
hostel_detai...
facilities.java
complaints.java
student logs in
warden deletes the facility
admin verifies and deletes the facility
login succesful
facility deleted from hostel
Message40
student asks for facilities
warden verfies and then adds the facility
facility for the student is added
Activate
Go to Settin
Sequence Diagram


*class diag...
Sequence_di...
sequence_di...
accounts.java
hostel_detai...
facilities.java
complaints.java
student asks for facilities
warden verfies and then adds the facility
facility for the student is added
the database has the facility with the student
facility is added for the student
student gets facilities
asks to delete a facility
the warden verifies and deletes the facility
the facility is deleted
the facilities are deleted
the student complains for the facility
the complaint is added in the database
Activate Wi


the student complains for the facility
the complaint is added in the database
the complaint is added in the database
the complain is addressed

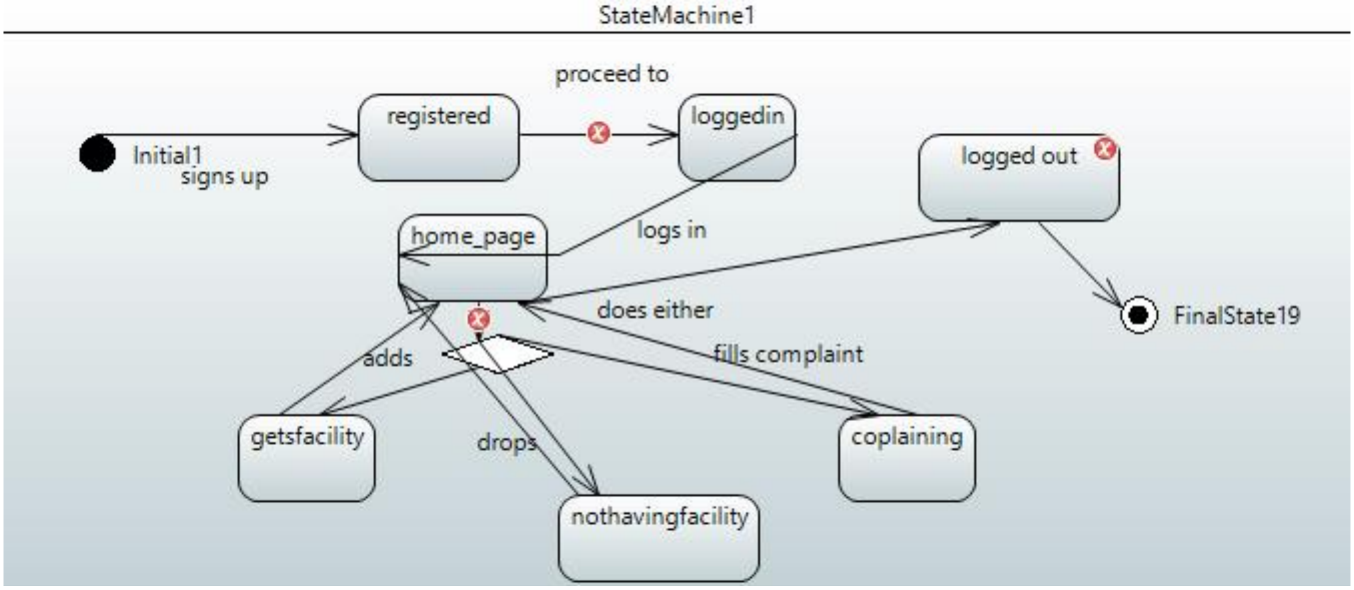
9.1.2.3 <Sequence Diagram n>


*class diag...
Sequence_di...
sequence_di...
sequence_di...
accounts.java
facilities.java
complaints.java
»16
Interaction1
student
warden
admin
database
student logs in
warden logs in
login succesful
login succesful
admin sees the fee related details
the admin send the fees detials of all students to warden
the students and their dues are shown
the warden sends the fee voucher to student
the student pays the fee
the warden updates in the admin
the admin verifies and updates in the database
Activate Windo


the admin send the fees detials of all students to warden
the students and their dues are shown
the warden sends the fee voucher to student
the student pays the fee
the warden updates in the admin
the admin verifies and updates in the database
the ppayment is shown in the datbase
payment is confirmed

9.1.3. State Diagrams

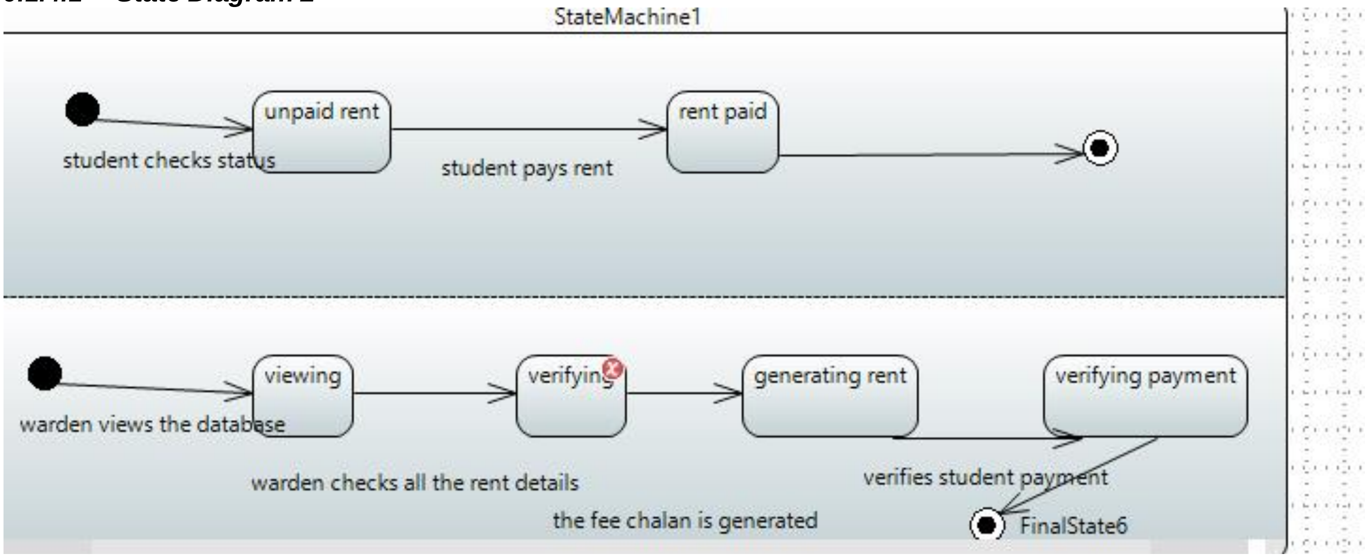
9.1.4. <state diagram 1>



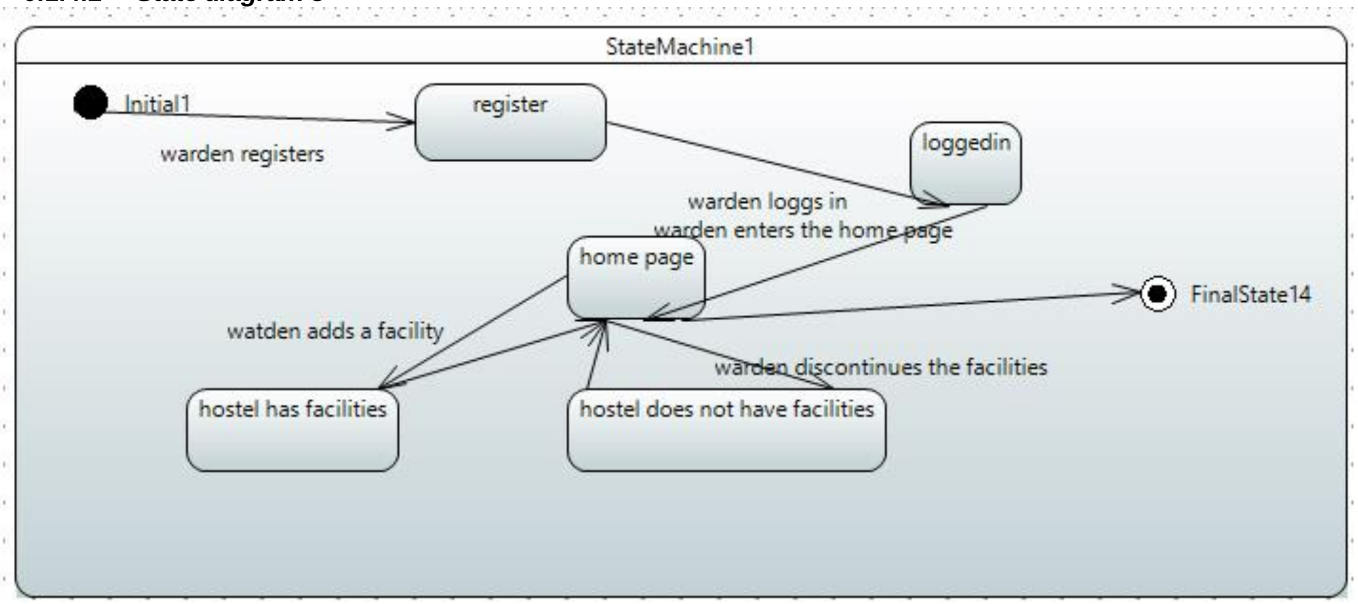
.

.

9.1.4.1 <State Diagram 2>

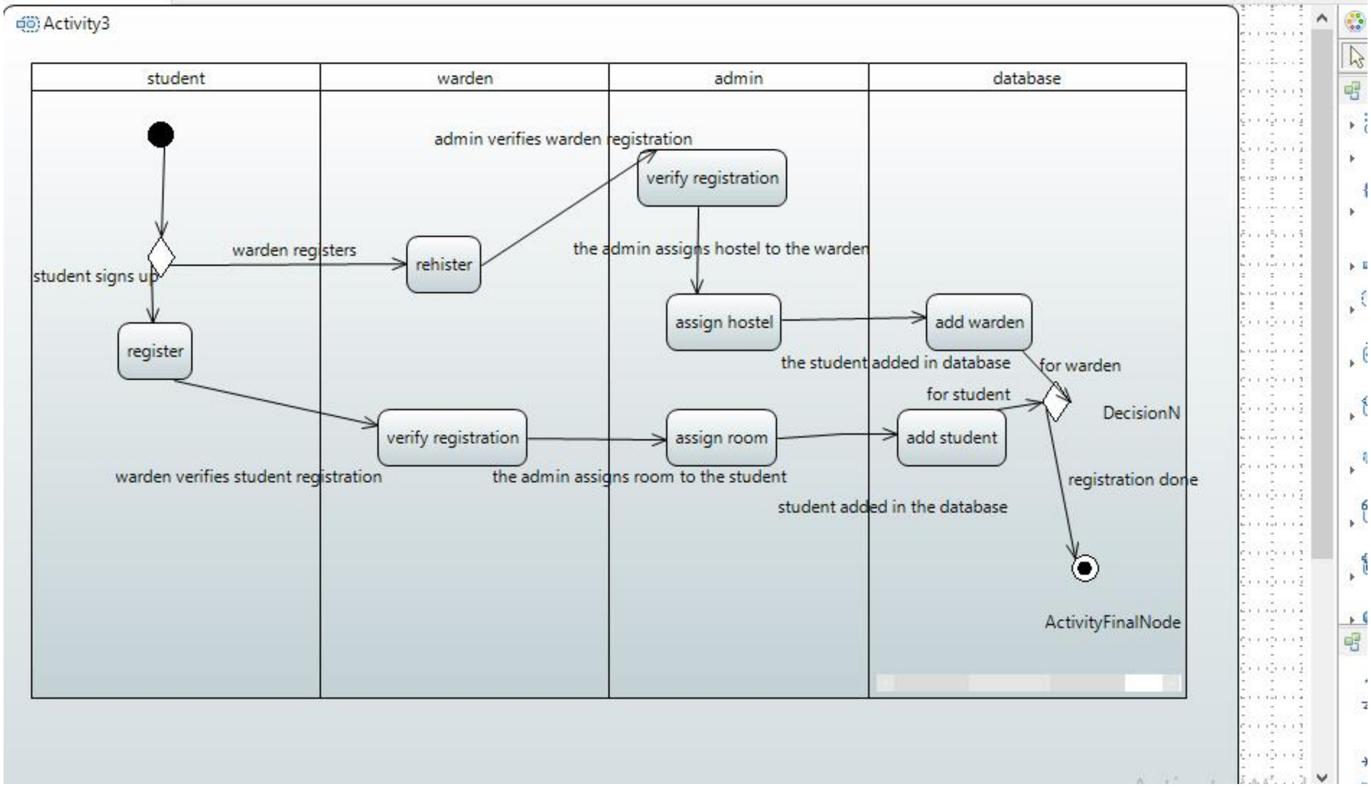


9.1.4.2 <State diagram 3>

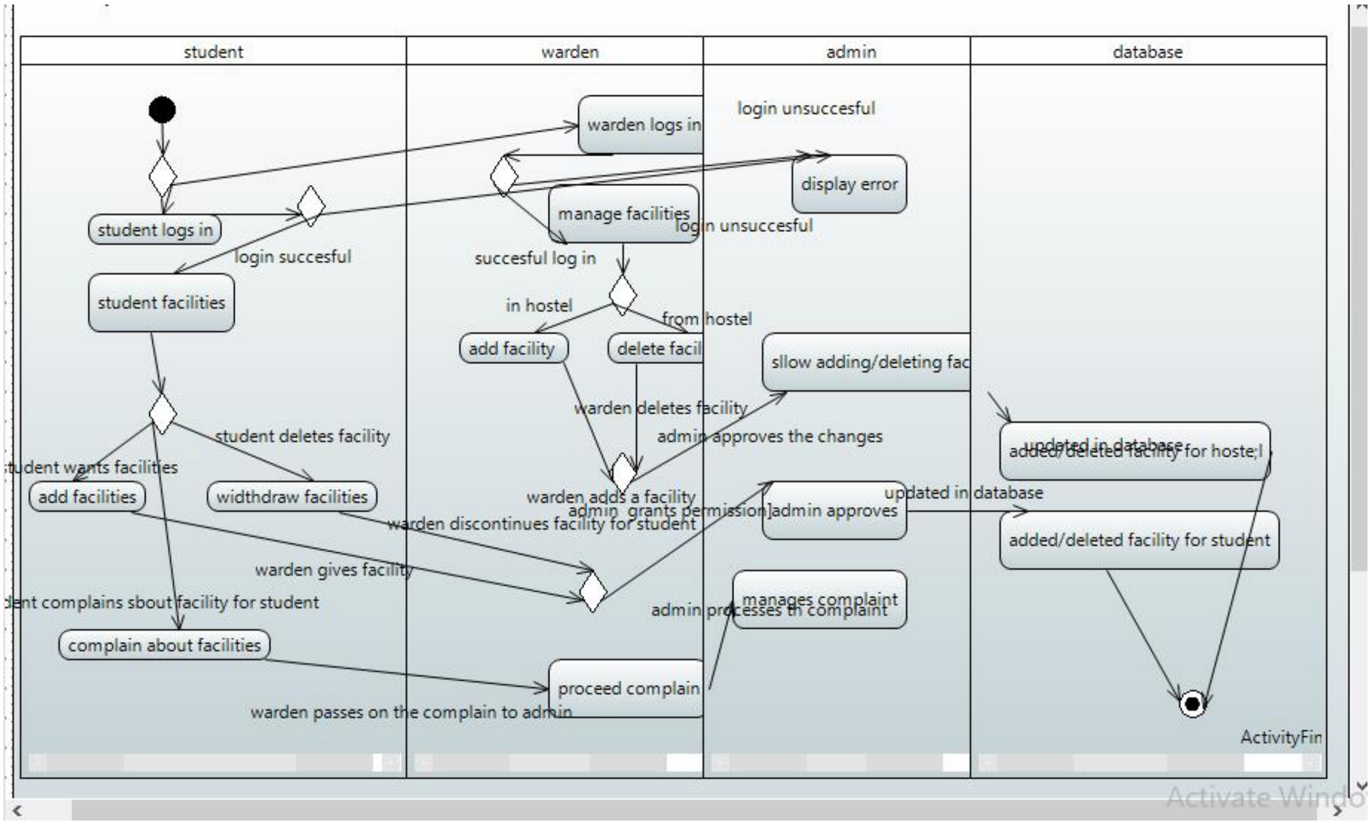


9.1.5. Activity Diagram

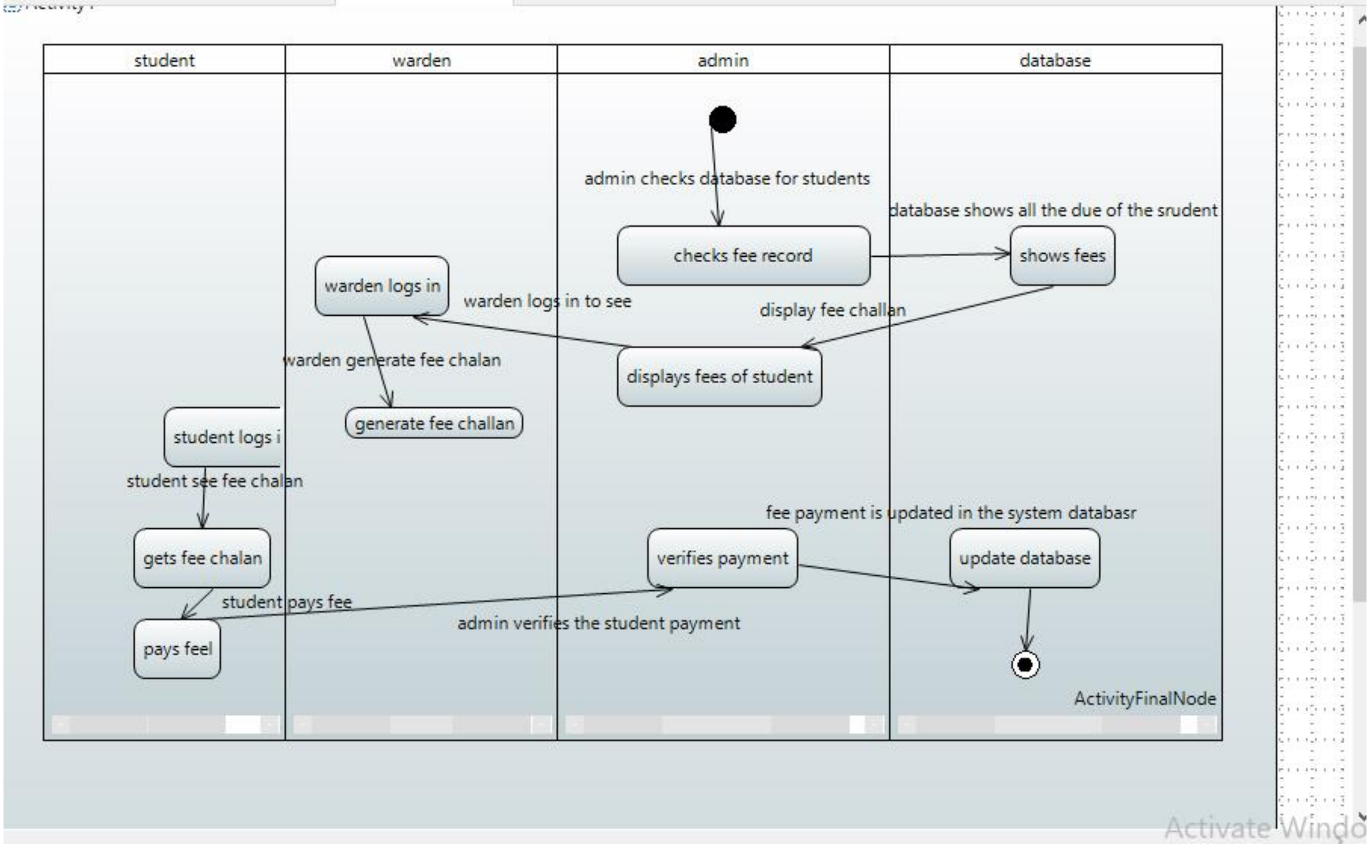
9.1.5.1 <Activity Diagram 1>



9.1.5.2 <Activity Diagram 2>


student
warden
admin
database
warden logs in
login unsuccesful
display error
student logs in
manage facilities
login unsuccesful
login succesful
succesful log in
student facilities
in hostel
from hostel
add facility
delete facil
sllow adding/deleting fac
warden deletes facility
admin approves the changes
student deletes facility
tudent wants facilities
add facilities
widthdraw facilities
warden adds a facility
admin grants permission]admin approves
updated in database
updated in database
added/deleted facility for hoste;l
warden discontinues facility for student
added/deleted facility for student
warden gives facility
dent complains sbout facility for student
admin processes th complaint
manages complaint
complain about facilities
proceed complain
warden passes on the complain to admin
ActivityFin
Activate Windo

9.1.5.3 Activity Diagram 3>



10. References

No references taken specifically, just used a few YouTube video tutorials to understand a few things.

11. Appendices

Not applicable