# Fast Hostel Reservation



***Version: [1.0]***

|  |  |
| --- | --- |
| *Course Code* | *CS 3004* |
| *Instructor* | *Nida Munawar* |
| *Project Team* | *Shahryar 21k-3411*  *Israr Ayoub 21k-4521*  *Sanaullah 21k-3412* |
| *Submission Date* | *12-03-2023* |

## [Instructions]

* *No section of template should be deleted. You can write ‘Not applicable’ if asection is not applicable to your project. But all sections must exist in the final document.*
* *All comments/examples mentionedinsquarebrackets([])areinthetemplateforexplanationpurposesandmust bereplaced/removedinfinaldocument.*
* *This’Instruction’sectionshouldalsoberemovedinfinal document.*

*Table of Contents*

1. [Introduction 5](#_bookmark0)
   1. [Purpose ofDocument 5](#_bookmark1)
   2. [Intended Audience 5](#_TOC_250002)
2. [OverallSystemDescription 6](#_bookmark2)
   1. [ProjectBackground 6](#_bookmark3)
   2. [ProjectScope 6](#_bookmark4)
   3. [NotInScope 6](#_bookmark5)
   4. [ProjectObjectives 6](#_TOC_250001)
   5. [Stakeholders 6](#_bookmark6)
   6. [Operating Environment 6](#_bookmark7)
   7. [SystemConstraints 6](#_TOC_250000)
   8. [Assumptions&Dependencies 6](#_bookmark8)
3. [ExternalInterfaceRequirements 7](#_bookmark9)
   1. [HardwareInterfaces 7](#_bookmark10)
   2. [SoftwareInterfaces 7](#_bookmark11)
   3. [CommunicationsInterfaces 7](#_bookmark12)
4. [FunctionalRequirements 8](#_bookmark13)
   1. [FunctionalHierarchy 8](#_bookmark14)
   2. [UseCases 8](#_bookmark15)
      1. [[Admin] 8](#_bookmark16)
      2. [Student]… 8
5. [Non-functionalRequirements 9](#_bookmark17)
   1. [PerformanceRequirements 9](#_bookmark18)
   2. [SafetyRequirements 9](#_bookmark19)
   3. [SecurityRequirements 9](#_bookmark20)
   4. [UserDocumentation 9](#_bookmark21)

[SDS 10](#_bookmark22)

1. [SYSTEMARCHITECTURE 11](#_bookmark23)
   1. [System LevelArchitecture 11](#_bookmark24)
   2. [Software Architecture 11](#_bookmark25)
2. [DESIGNSTRATEGY 12](#_bookmark26)
3. [DETAILEDSYSTEM DESIGN 13](#_bookmark27)
   1. [Database Design 13](#_bookmark28)
4. [ApplicationDesign 15](#_bookmark29)
5. [References 15](#_bookmark30)
6. [Appendices 17](#_bookmark31)

## Introduction

##### PurposeofDocument

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

##### Intended Audience

[user(Admin),Student.]

* 1. **DefinitionofTerms,AcronymsandAbbreviations**

Not applicable

*.*

|  |  |
| --- | --- |
| ***Term*** | ***Description*** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

##### DocumentConvention

Font: Arial for all content.

Font size: 10(Regular content/Paragraph) 12 for heading

## OverallSystemDescription

##### ProjectBackground

Many fellow students of 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.

##### ProjectScope

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

##### NotIn Scope

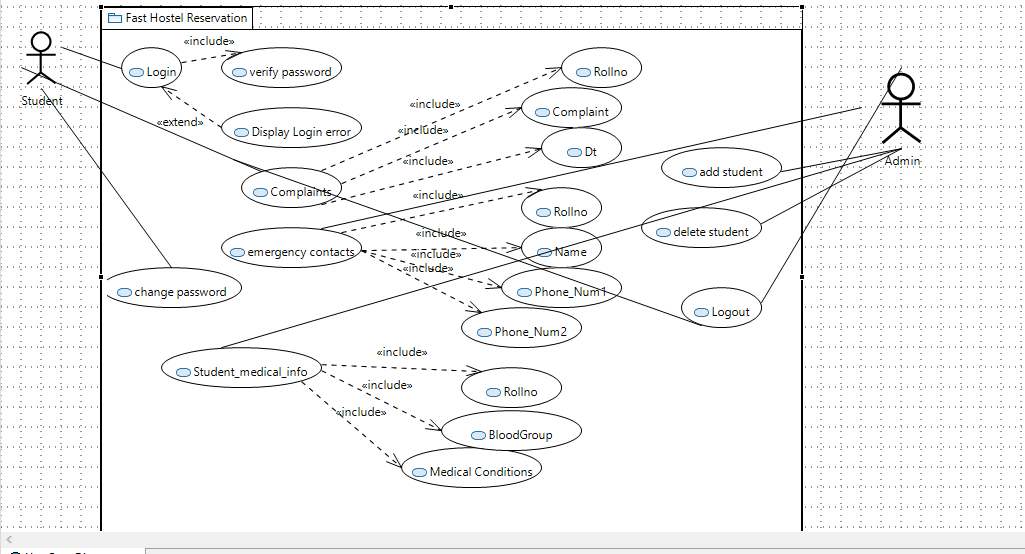
Chat box for students’ assistance, salary management of warden and rent management of the whole hostel building. Can be implemented in future.

##### ProjectObjectives

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

##### Stakeholders

Users (students and warden(or university admin)) admin and database developer



##### OperatingEnvironment

Web Browser on pcs ,Smartphones,etc

##### SystemConstraints

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.*

##### Assumptions&Dependencies

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

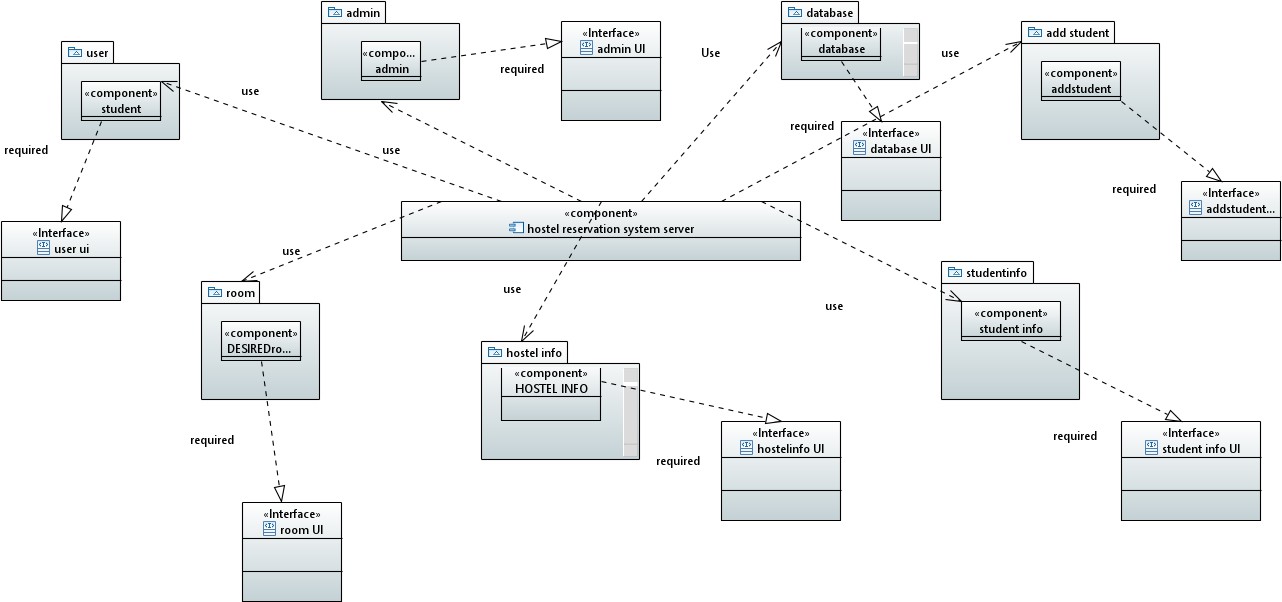
## ExternalInterfaceRequirements

##### HardwareInterfaces

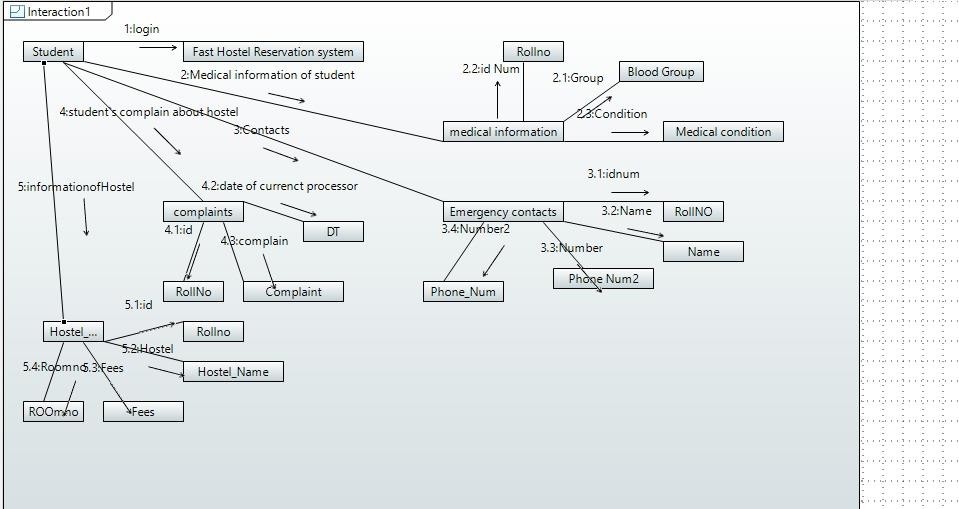
We will be needing a computer to manage and view the database

##### SoftwareInterfaces

Following component diagram describes the component of the classes their connection,dependencies,uses.



##### CommunicationsInterfaces



## FunctionalRequirements

##### FunctionalHierarchy

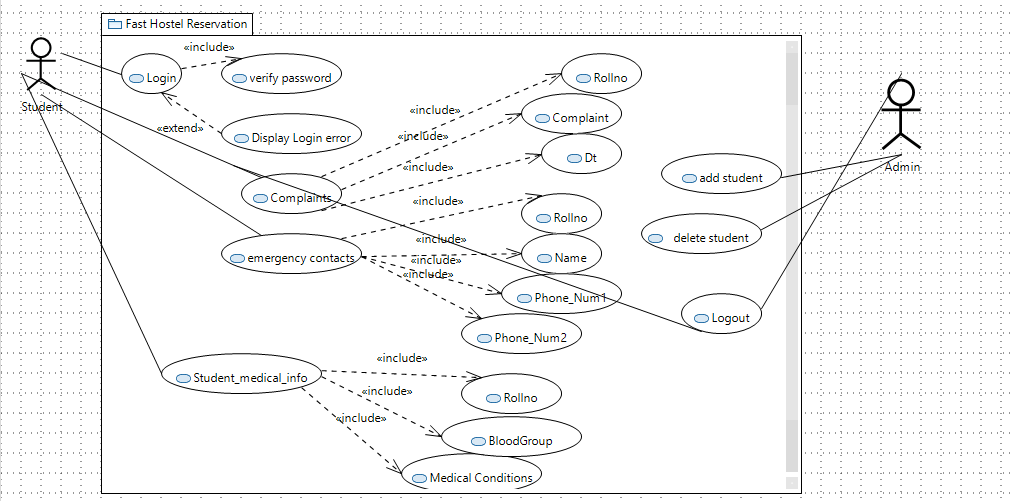
The functionalities are either by the student or the warden. For warden

* *Registration*
* *Log in*
* *Add student*
* *Add facility in hostel*
* *Check student rent status For student*
  + *Registration*
  + *Log in*
  + *Avail facility*
  + *Complain regarding a facility*

##### UseCases

#### [Admin student usecase]

[Use Case Diagram]



[UseCaseDescription]

|  |  |
| --- | --- |
| ***UseCasename:*** | Fast Hostel Reservation system |

|  |  |
| --- | --- |
| ***UseCaseDescription****:Itshowswhatfunctionalitieswillbeusedbywhichuserandwhatfunctionalitiesareincludedorex cluded.* | |
| ***Primaryactor:****Student* | ***Otheractors:****Admin or user* |
| ***Stakeholders:*** | *Admins* ,*developers* |
| ***Relationships***   * ***Includes:*** *Add students, view details, medical\_information, Hostel\_information ,Delete student* * ***Extends:****invalid password/username* | |
| ***Pre-conditions:***   * *Requirement for registration* | |
| ***FlowofEvents:***  ***1.****Admin register*  ***4.*** | |
| ***Alternativeandexceptionalflows:4. 1****….* | |
| ***Post-conditions:***   * None | |

|  |  |
| --- | --- |
| Id | UC1 |
| USE CASE NAME | Warden registration form |
| AUTHOR | Shahryar, Israr Ayoub, Sanaullah |
| LAST REVISED | 12/03/2023 |
| 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 | Shahryar, Israr Ayoub, Sanaullah |
| LAST REVISED | 12/03/2023 |
| 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. |

|  |  |
| --- | --- |
|  | 1. The system allocates a hostel to the warden. 2. 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 | Shahryar, Israr Ayoub, Sanaullah |
| LAST REVISED | 12/03/2023 |
| 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 | Shahryar, Israr Ayoub, Sanaullah |
| LAST REVISED | 12/03/2023 |
| 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 invalid password.  3a. even after correct credentials, the warden is not logged in. |

## Non-functionalRequirements

##### PerformanceRequirements

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

##### SafetyRequirements

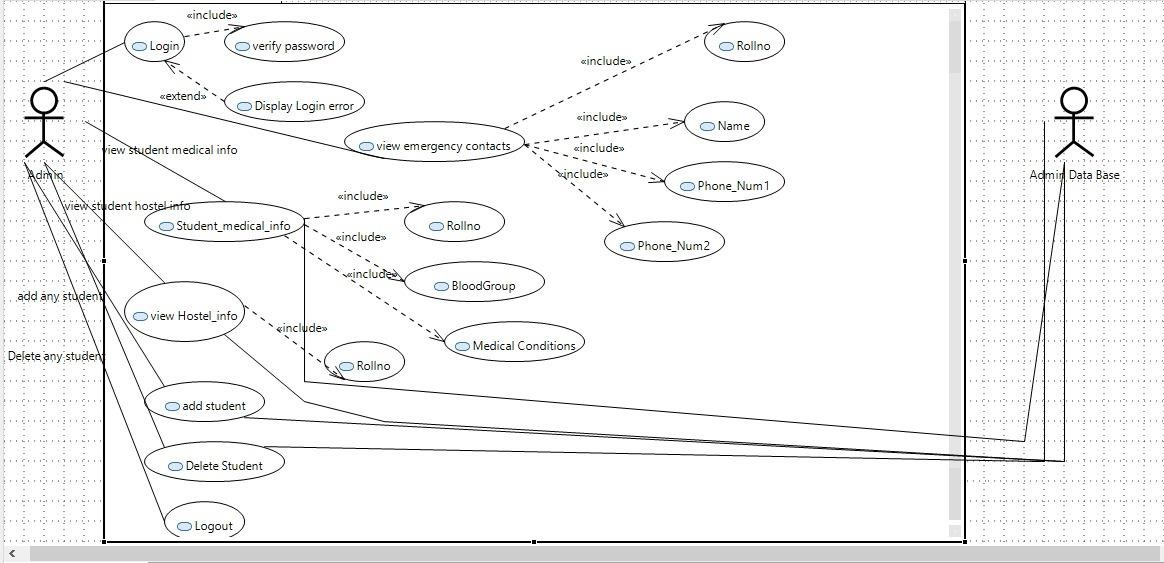
As it’s a public hostel, the system requires to ensure the safety of the users’ data in our database. There should be complete confidentiality and no data leaks are affordable

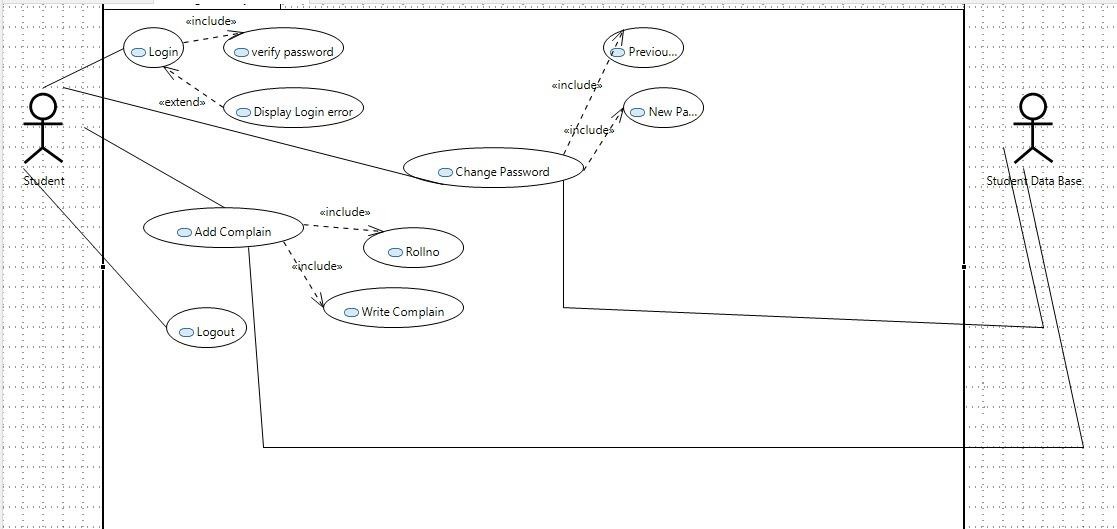
##### SecurityRequirements

For any user to enter in 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

##### UserDocumentation

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





# SDS

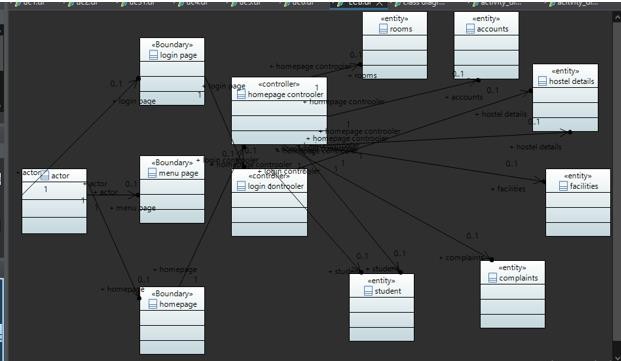
## SystemArchitecture

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

### SystemLevelArchitecture

Deployment diagram

### SoftwareArchitecture

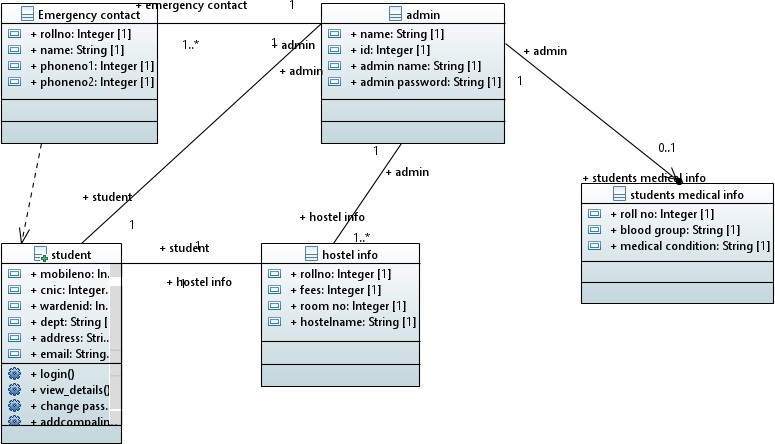


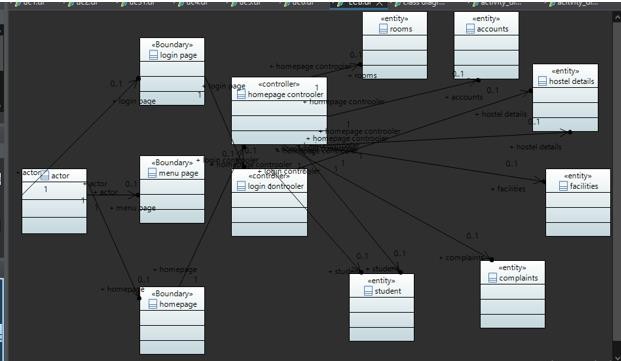
## DesignStrategy

The gantt chart below shows the design strategy

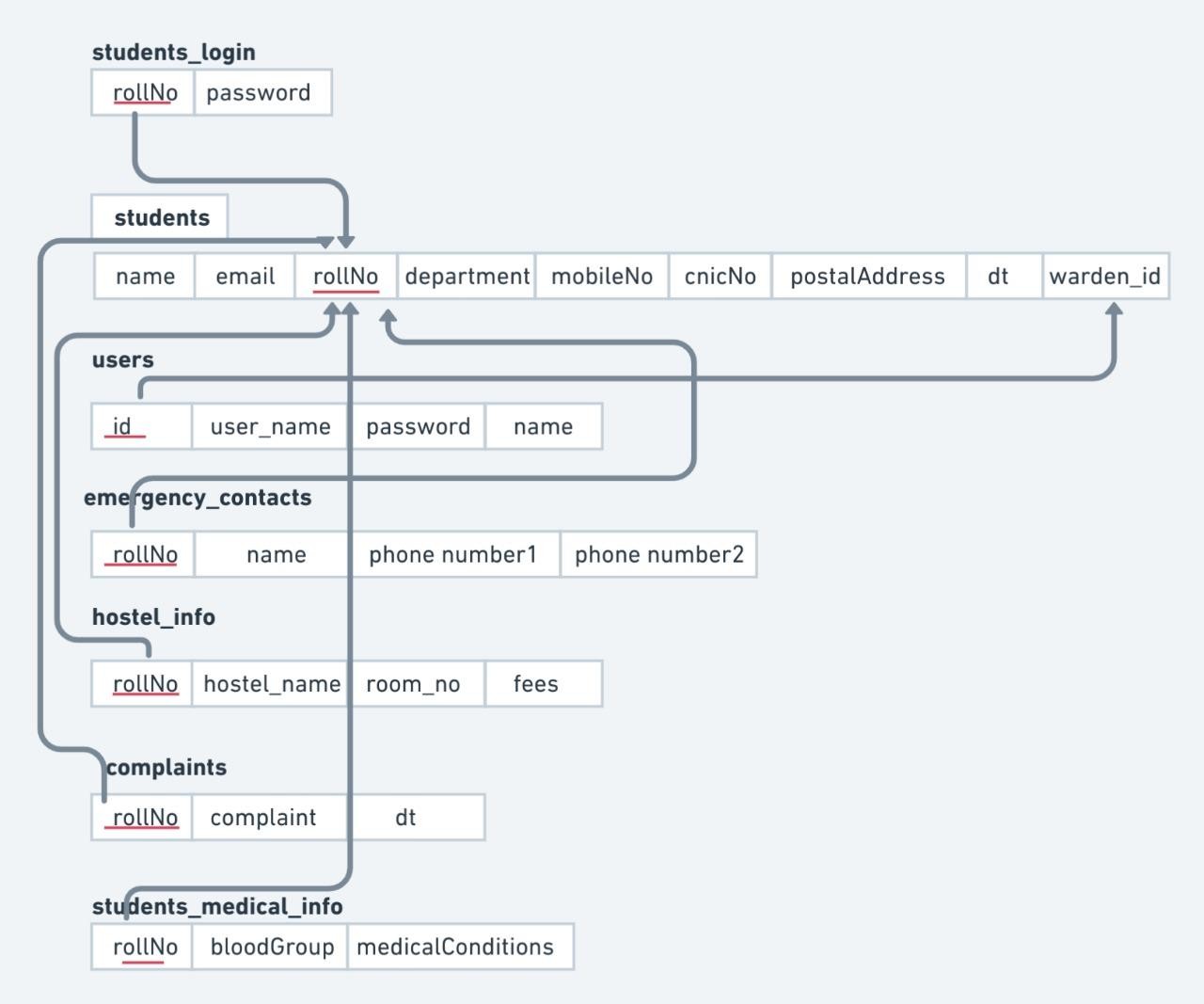
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | | | | |
| **TASK NAME** | August | September | October | November | December |
| **PLANNING** | Yes | Yes |  |  |  |
| **RESEARCH** |  | Yes | yes |  |  |
| **DESIGN** | Yes | | | | |
| **IMPLEMENTATION** | Yes | | | | |
| **GOLLLOW UP** | Yes | | | | |

## DetailedSystemDesign

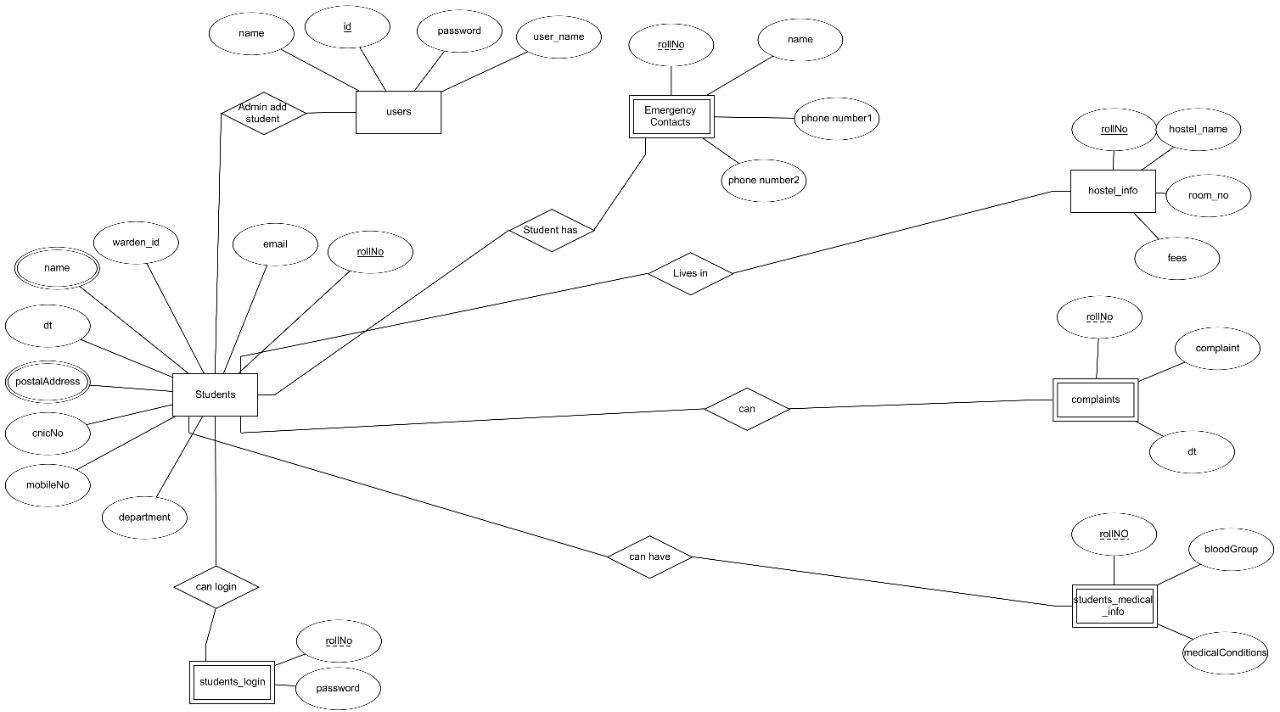


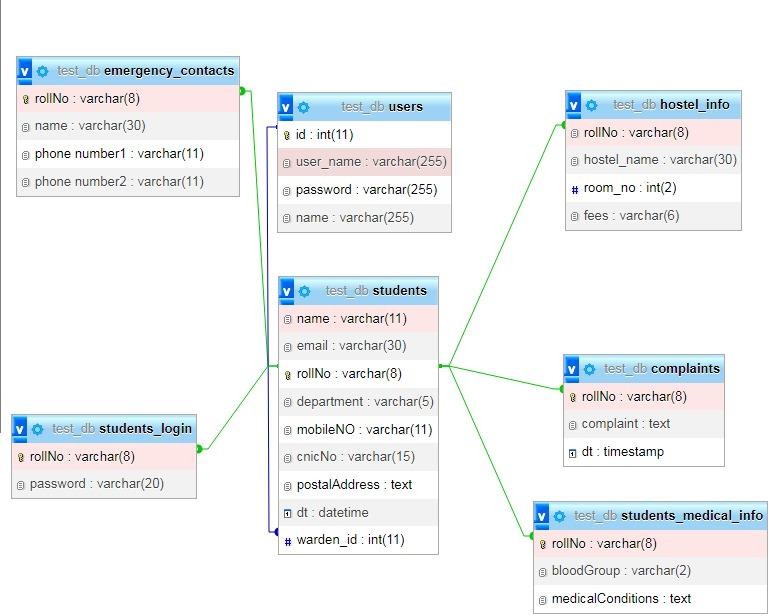
ECB Class Diagram

### Database Design



#### ERDiagram





* + 1. ***DataDictionary***

[Theconventionrecommendedforwritingthedatadictionaryisasfollows.]

* + - 1. ***Data1***

Student

Main users/Admin of the model, all of them can add student, Delete student, view student detail,Medical information of students,view\_Hostelinformation,watch complains such as that a student want to live in Hostel He/she will be allotted to room if room has any free bad and that bad will be given to student from Admin and can view all details of studens.

***8.1.2.2.Data2***

*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 complains.

***8.1.2.2.Data3***

Rooms

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

***8.1.2.3.Data4***

Complaints

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

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **<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.* | | | | | |
| ***Contentdescription*** | | *The student data is to maintain the records of each student living in our hostel.* | | | | | |
|  | | | | | | | |
| ***Column Name*** | ***Description*** | | ***Type*** | ***Length*** | ***Nullable*** | ***Default Value*** | ***KeyType*** |
| *Roll\_no* | *The unique ID assigned to each student* | | *Intege r* | *Standard* | *No ]* | *1 greater than the roll\_no of previous student* | *PK* |
| *First\_nam e* | *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* | | | | | |
| ***Contentdescription*** | | | *The hostel\_details will hold all the details related to the hostel.* | | | | | |
|  | | | | | | | | |  |
| ***Column Name*** | | ***Description*** | ***Type*** | | ***Length*** | ***Nullable*** | ***Default Value*** | ***KeyType*** |
| *Hostel\_ID* | | *The unique ID assigned to each hostel* | *Intege r* | | *Standard* | *No* | *1 greater than the Id assigned to the previous hostel* | *PK* |
| *Hostel\_na me* | | *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* | | | | | |
| ***Contentdescription*** | | | *The rooms are specific to each hostel,, these are managed by the warden and the admin.* | | | | | |
|  | | | | | | | | |  |
| ***Column Name*** | | ***Description*** | ***Type*** | | ***Length*** | ***Nullable*** | ***Default Value*** | ***KeyType*** |
| *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* |

|  |  |
| --- | --- |
| **<Data4>** | |
| ***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* |
| ***Contentdescription*** | *All the details regarding complaints are stored there* |

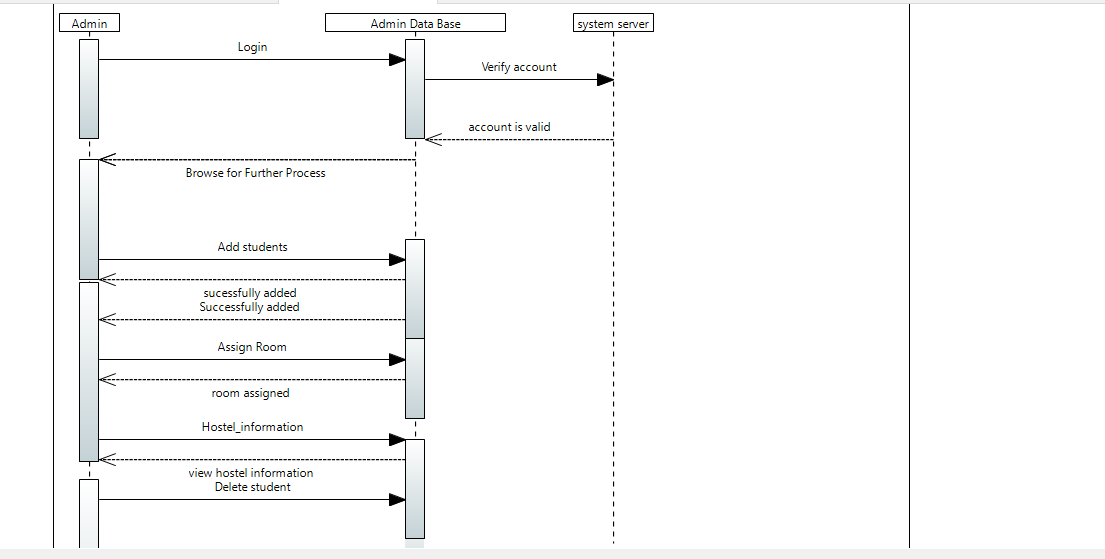
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | |
| ***Column Name*** | ***Description*** | ***Type*** | ***Length*** | ***Nullable*** | ***Default Value*** | ***KeyType*** |
| *Roll\_no* | *The toll number of a student* | *Intege r* | *Standard* | *No* | *NONE* | *FK* |
| *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* |

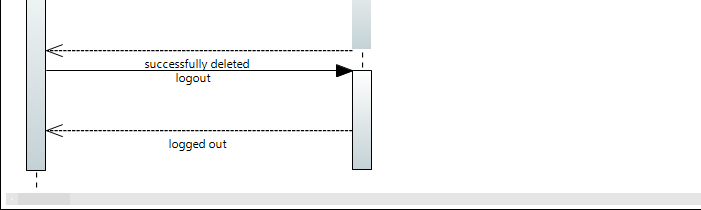
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *Status* | *The status of the complaint being registered* | *Boolea n* | *Standard* | *No* | *None* | *None* |

### ApplicationDesign

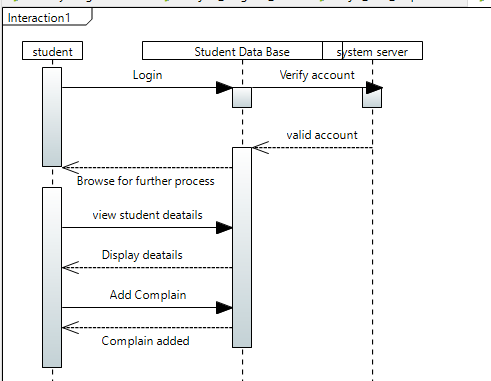
#### SequenceDiagram

* + - 1. ***<SequenceDiagram1> Admin***





* + - 1. ***<SequenceDiagram2>***

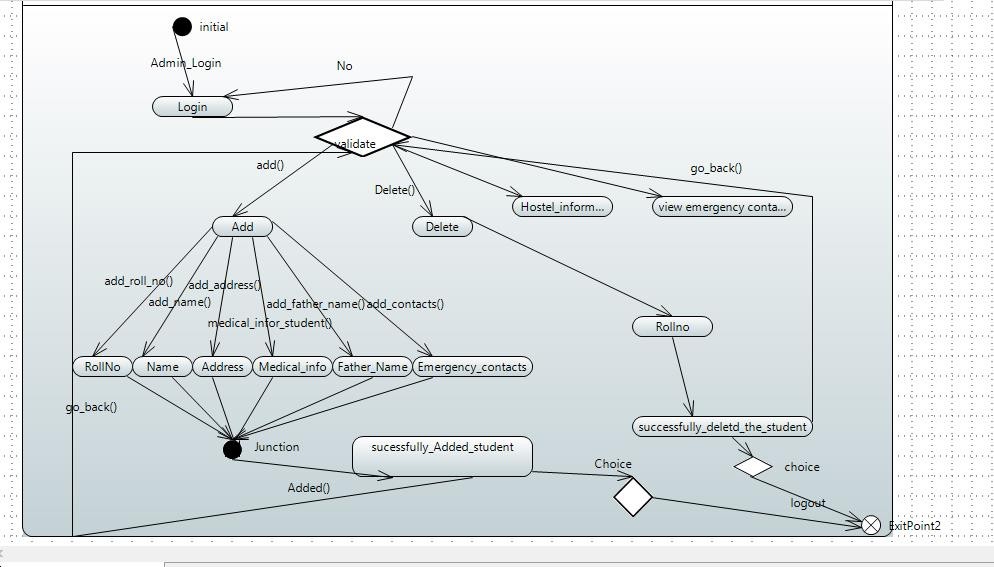
Student

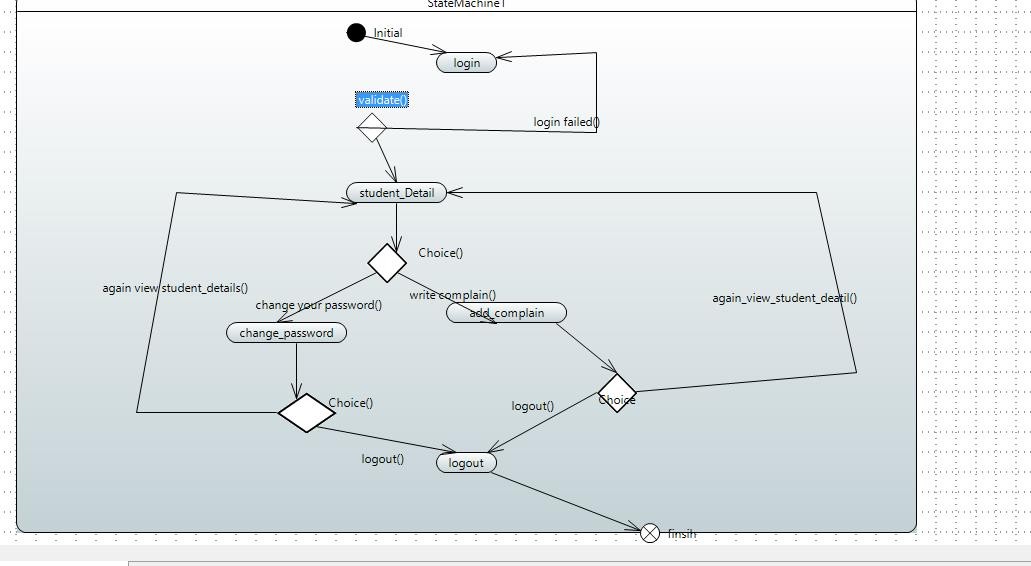
*.*

*.*

#### StateDiagram

* + - 1. ***<State Diagram of Admin>***



* + - 1. ***<state Diagram of student>***

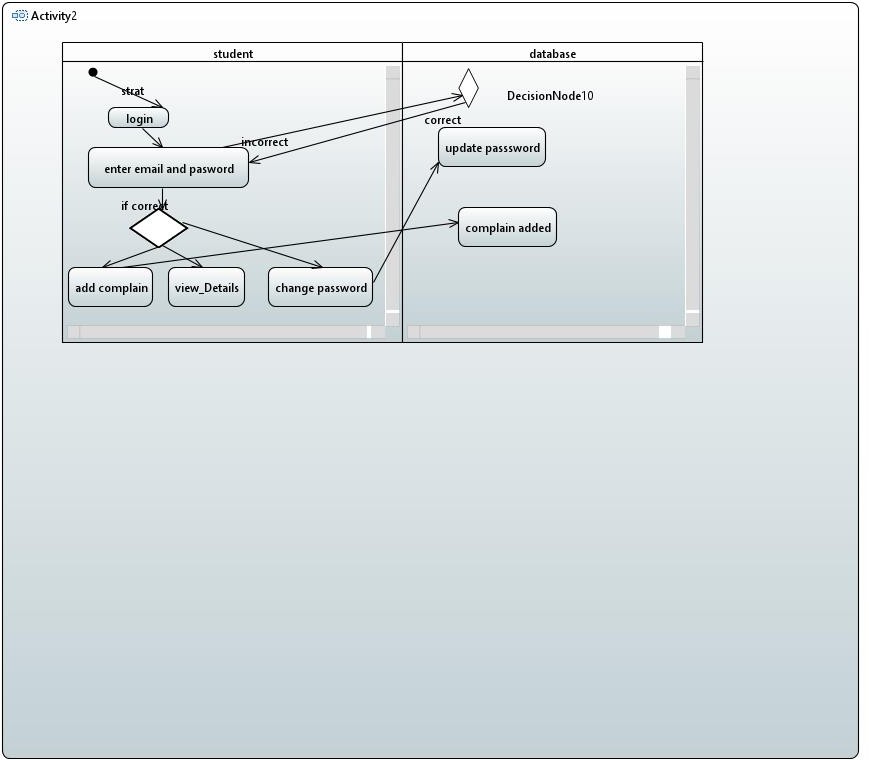
*.*

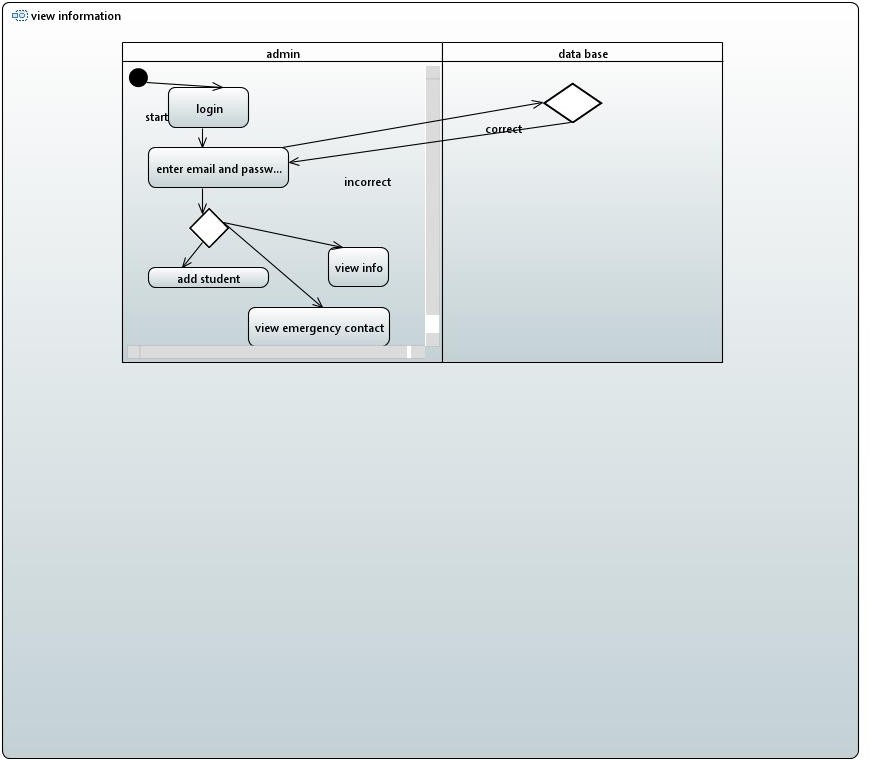
*.*

*.*

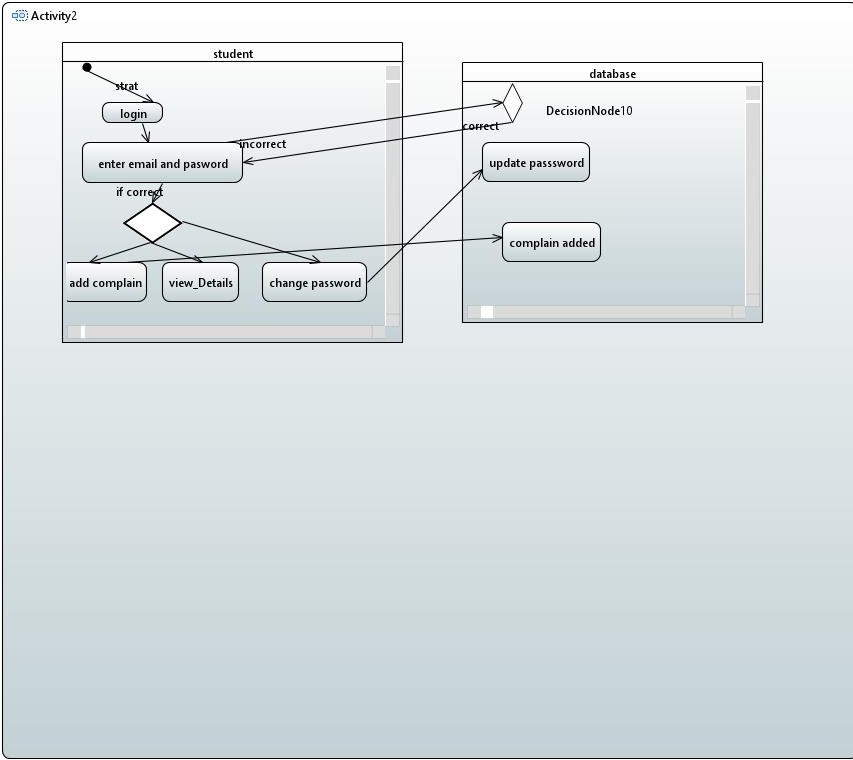
#### ActivityDiagram

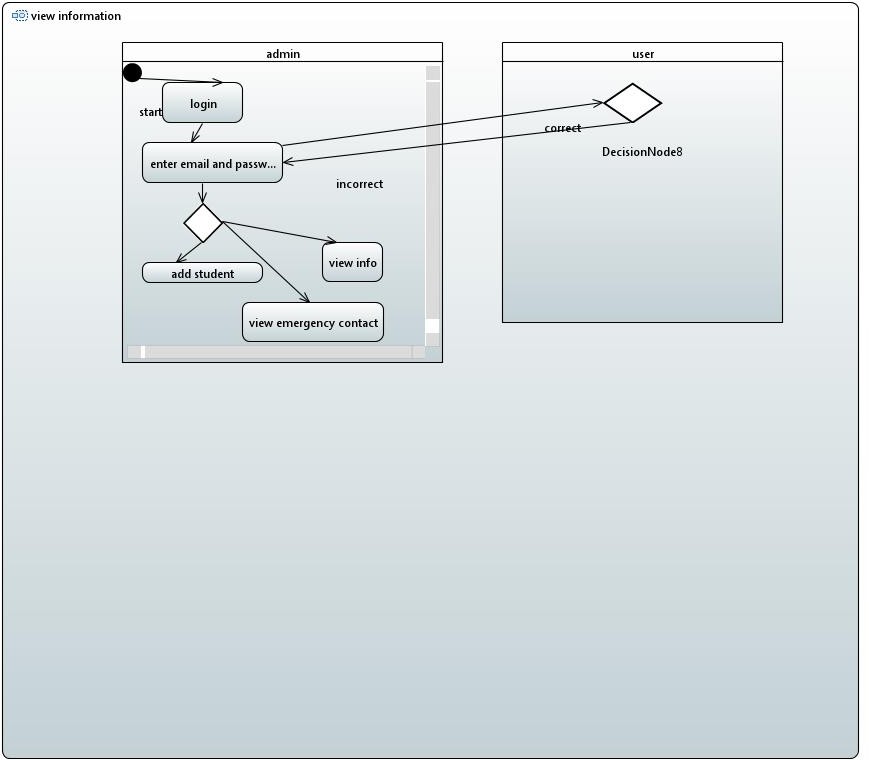
* + - 1. ***<ActivityDiagram1>***





1. ***References( Activity diagram of Student, Database and admin,database)***





***Deployment Diagram***

