

**ACKNOWLEDGEMENT**

We take this opportunity to thank my respected **Prof. Pujan Jani**. We would like to forward my thanking tribute to **Prof. Prashant Vaghela**, Head of Department of BCA and my **J.M.Chaudhary** College to whom we own pleasurable debit for his splendid support, inspiration and thought production.

This Project is Completed Nicely because of our teacher’s tremendous efforts.

Without their support and encouragement this report could not have been completed. We also take this opportunity to thank the authorities of my college and company for providing us with sufficient resources that has helped us make the project valuable.

We also thank all those who have shared their extended help to me, and we may have not mentioned them.

Nehal Patel

Nelvi Talaviya

Sonali Patel

**INDEX**

**Sr.No. Topic Page No.**

1.Introduction……………………………………………7

2.Project Profile………………………………………….8

2.1 Existing System……………….………………8

2.2 Proposed System………………………………9

2.3 Modules……………………..……………..…10

2.3.1 Admin…………………………..….11

2.3.2 Faculty………………………...…...12

2.3.3 Student…………………….……….13

2.4 Tools and technology…………………...…….14

3.System Flow……………….…………………………..15

4.UML Diagram………………………………………….17

4.1 Use Case Diagram……..……………………….17

4.1.1 Admin…..……………………….…..19

4.1.2 Faculty…..…………………………..20

4.1.3 Student………..……………………..21

4.2 Activity Diagram………………………………22

4.2.1 Admin……………….……………..25

4.2.2 Faculty……………………………...26

4.2.3 Student…………………..…………27

4.3 Class Diagram…………………….…………28

4.4 Sequence Diagram….…………….…………33

4.4.1 Login.…..……………………...…..35

4.4.2 Registration…………....……..……36

4.4.3 Application…………....…….…......37

4.4.4 Review…………………………...…38

5.Entity Relationship Diagram………………………….39

6.Data Dictionary……………………...……………...…41

**1. INTRODUCTION**

* College management system is Web application that developed for Students who already admitted in college , faculty.
* This Web Application is developed, which will help Students & faculty to done some activity in digital form.
* Our project helps Students & faculty to saving time & paper work.

**2. PROJECT PROFILE**

2.1 EXISTING SYSTEM

* In the existing system All paper work is done manually.
* That process is very time consuming & lengthy.
* Faculty didn’t had accurate information about Students.
* If some times faculty is not available then process can’t be done.

2.2 PROPOSED SYSTEM

* The proposed system is solving some problem from existing system.
* Students can write and submit reports or assignment directly with this system.
* Faculty can response to the Students reports without delay.

2.3 MODULES

There are three Modules.

1. Admin

2. Faculty

3. Student

2.3.1 ADMIN

* Login.
* Manage Profile.
* Registration Faculty.
* Registration Student.
* Manage Faculty.
* Manage Student.
* View Review.
* Logout.

2.3.2 FACULTY

* Registration.
* Login.
* Manage Profile.
* Manage Attendance.
* Manage Schedule.
* Manage Assignment.
* Manage Application.
* Manage Report.
* Manage Result.
* Manage Notes.
* View Student details.
* Give Review.
* Logout.

2.3.3 STUDENT

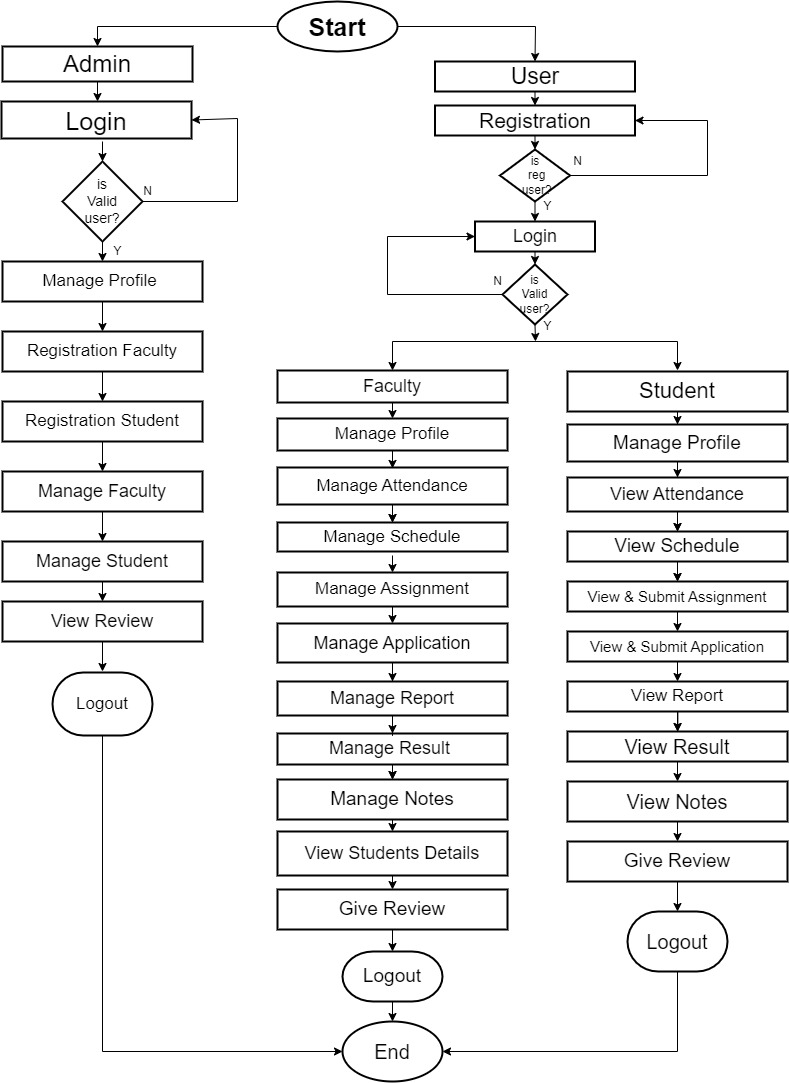
* Registration.
* Login.
* Manage Profile.
* View Attendance.
* View Schedule.
* View & submit Assignment.
* View & submit Application.
* View Report.
* View Result.
* View Notes.
* Give Review.
* Logout.

2.4 Tools and Technology

* **Project name :** College Management System
* **Front end :** Html5 , Css3 , Python
* **Back end :** My Sql
* **Database :** My Sql
* **Supporting :** VS code , Sublime text editor

**3. SYSTEM FLOW DIAGRAM**

* System Flow chart is the way of displaying how data flow in a system and how decisions are made of control events.
* To illustrate this , symbol are used.
* They are connected together to show what happens to data and where it goes.



**4. UML Diagrams**

4.1 Use Case Diagram

4.2 Activity Diagram

4.3 Class Diagram

4.4 Sequence Diagram

4.1 Use Case Diagram

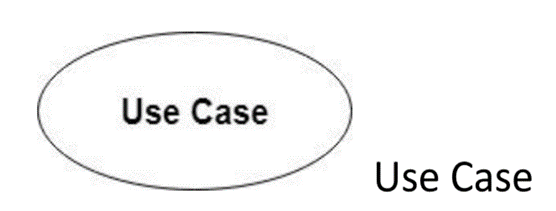
4.1.1 Admin

4.1.2 Faculty

4.1.3 Student

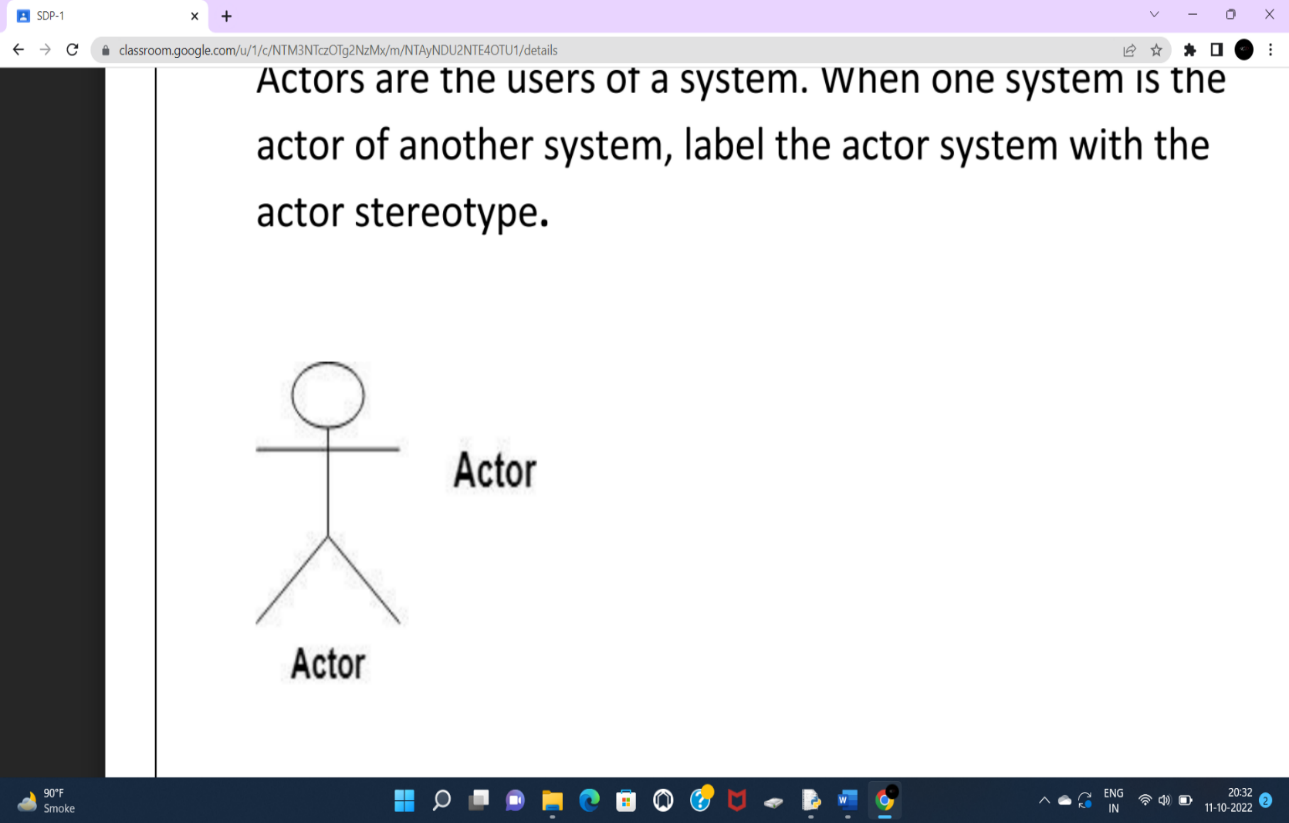
* **Use Case :**

Draw Use cases using the ovals with verbs that represent the system’s functions.

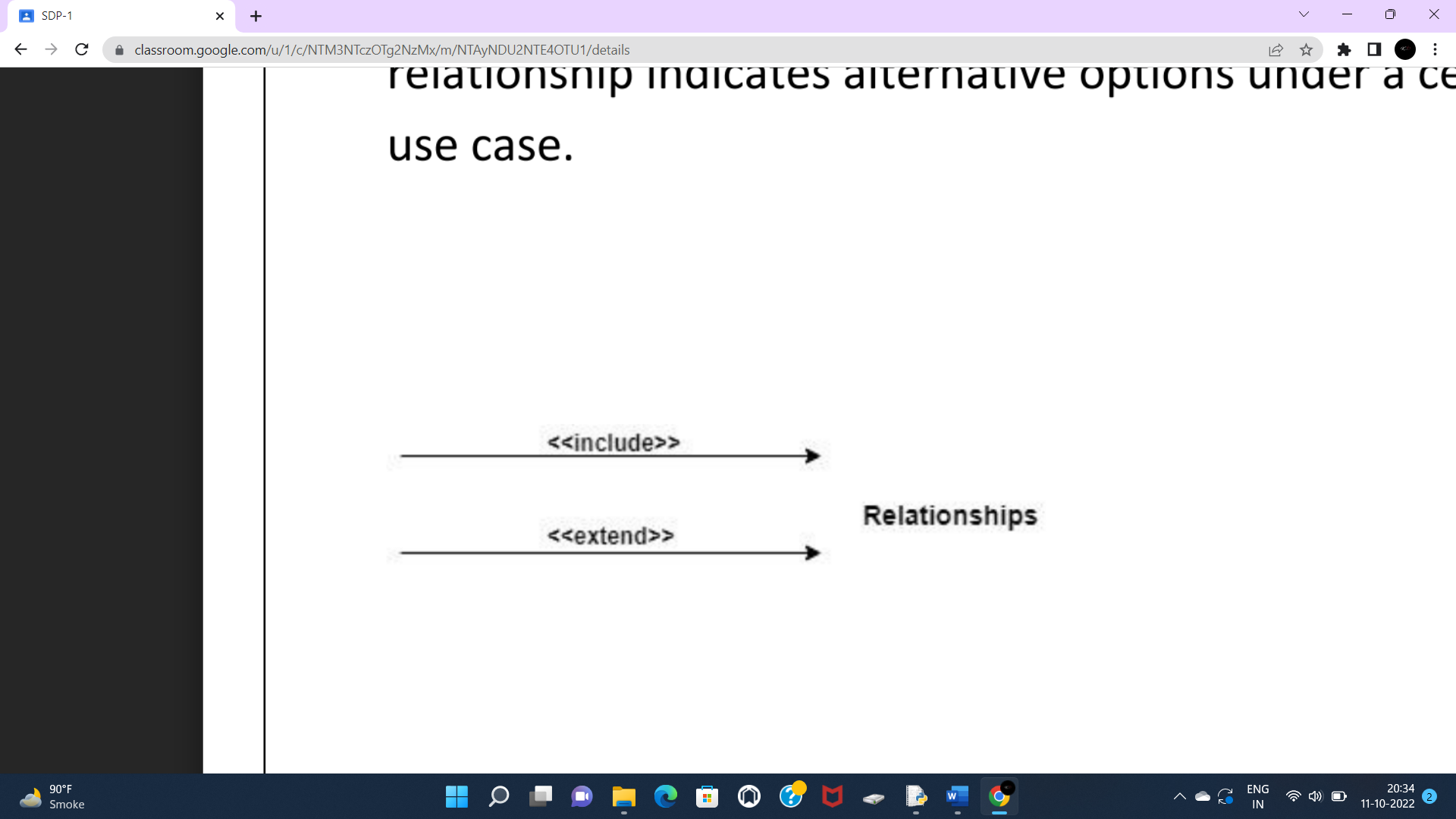
****

* **Actors :**

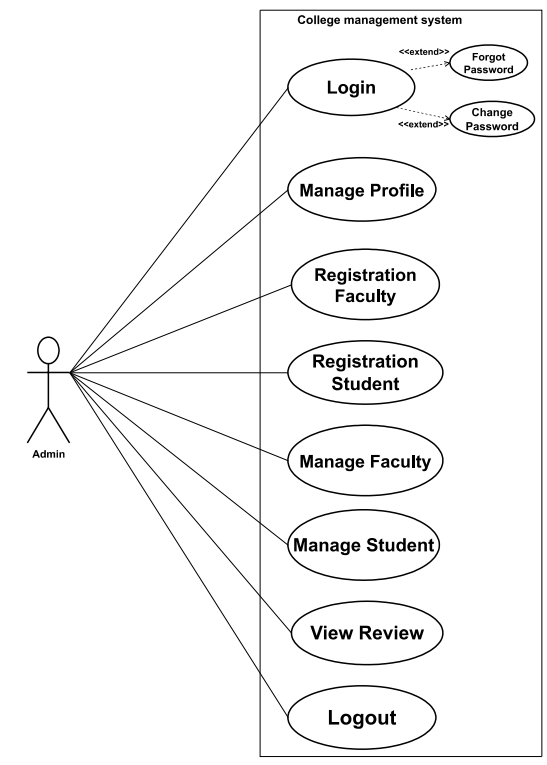
Actors are the users of a system. When one system is the actor of another system , label the actor system with the actor stereotype.



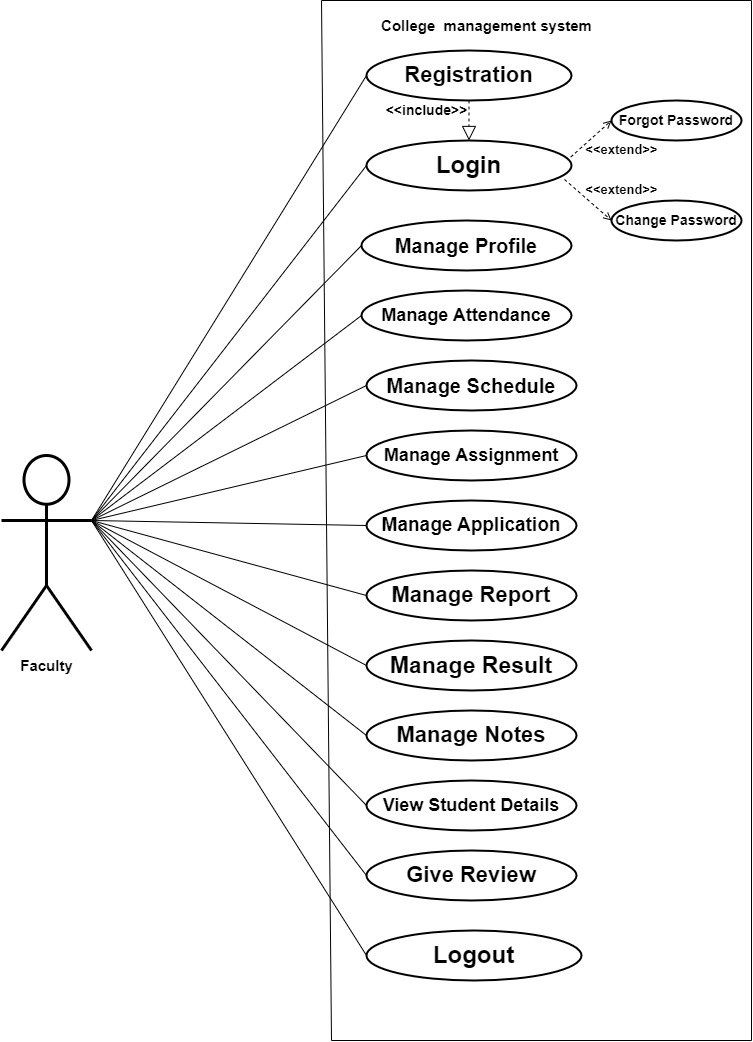
* **Relationships:**
* illustrate relationships between an actor and a use case with a simple line. For relationships among use cases, use arrows labeled either “uses” or “extend”.
* A “uses” relationships indicates that one use case is needed by another in order to perform a task.
* An “extends” relationship indicates alternative options under a certain use case.



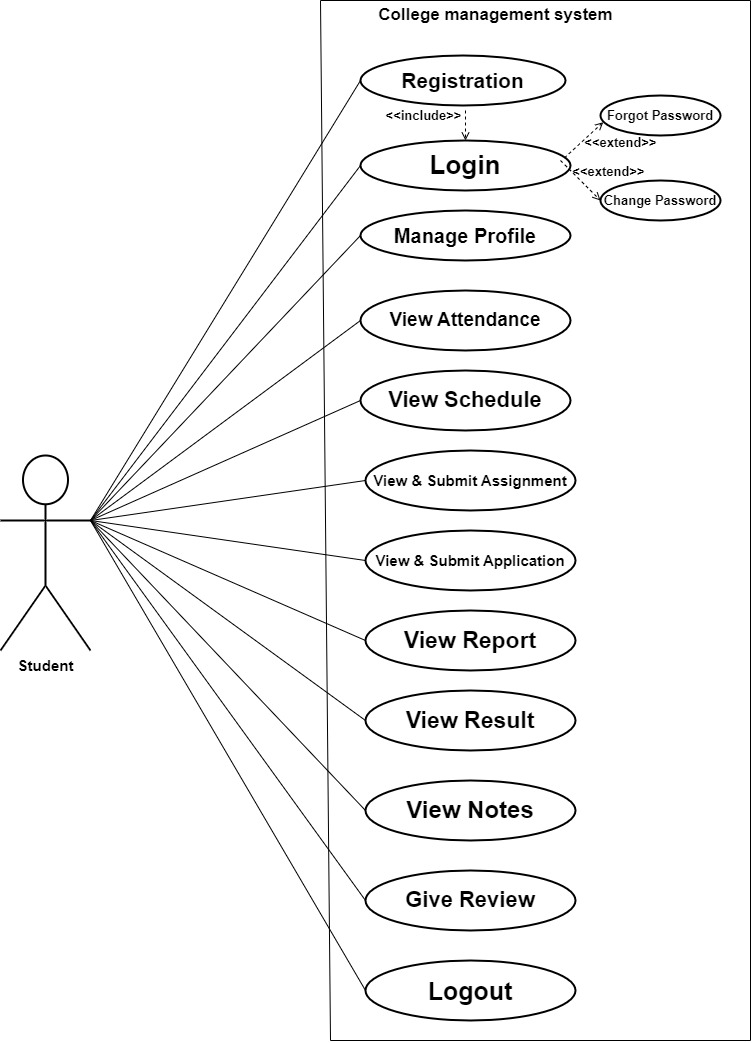
4.1.1 Admin



4.1.2 Faculty



4.1.3 Student



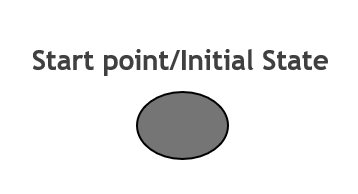
4.2 Activity Diagram

4.2.1 Admin

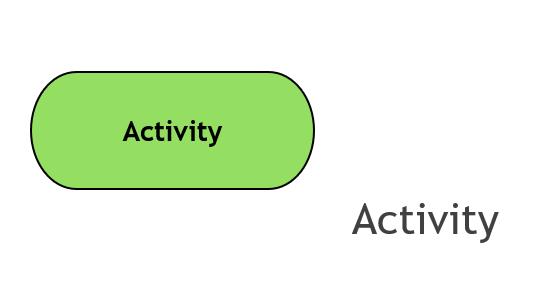
4.2.2 Faculty

4.2.3 Student

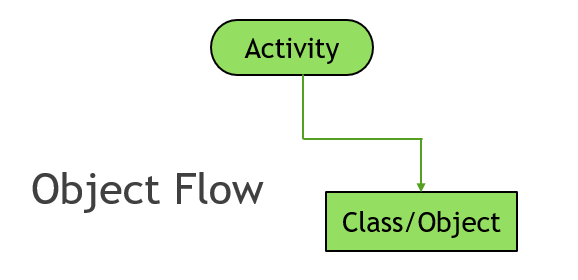
* + **Initial State or Start Point :**
    - A small filled circle followed by an arrow represents the initial action state or the start point for any activity diagram.
    - For activity diagram using swim lanes, make sure the start point is placed in the left corner of the first column.



* + **Activity or Action State:**
  + An action state represents the non-interruptible action of object .
  + You can draw an action state in Smart Draw using a rectangle with rounded corners .



* + **Object Flow:**
  + Object flow refers to the creation and modification of object by activities.
  + An object flow arrow from an action to an object means that the action creates or influences the object.
  + An object flow arrow from an object to an action indicates that the action state uses the object.



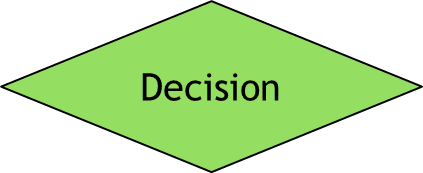
* + **Action Flow**
  + Action flows, also called edges and paths , illustrate the transitions from one action state to another .
  + They are usually drawn with an arrowed line.



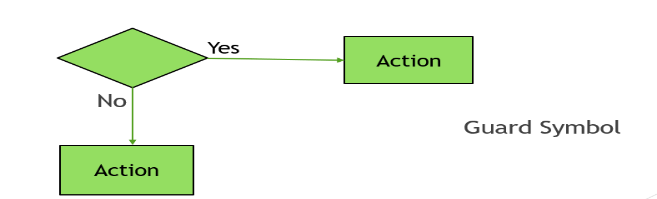
* + **Decisions :**

* + A diamond represents a decision with alternate paths .
  + When an activity requires a decision prior to moving on to the next activity , add a diamond between the two activities.
  + The outgoing alternates should be labelled with a condition or guard expression. You can also label one of the paths “else”.

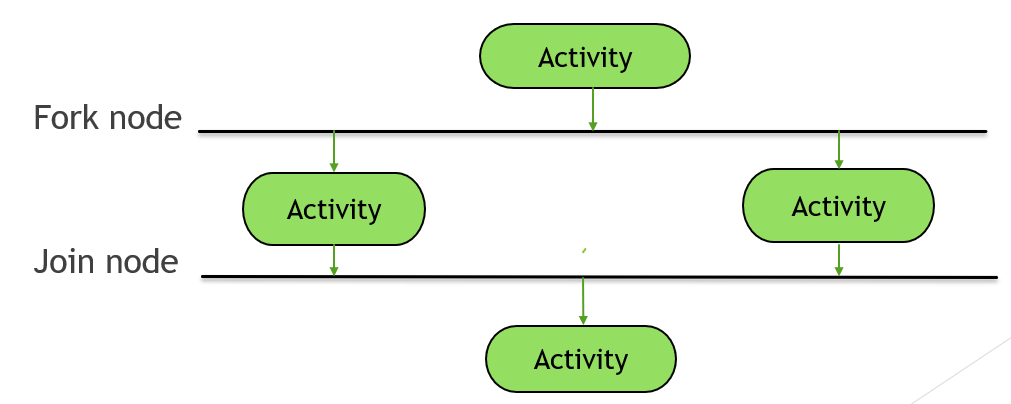
**Decision Symbol**



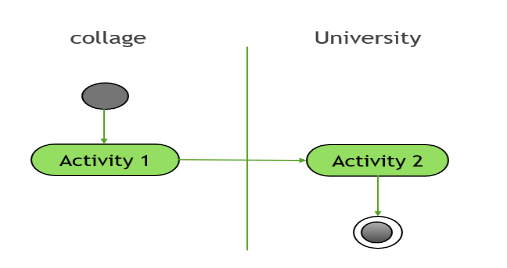
* + **Guards:**
  + An arrow pointing to a filled circle nested inside another circle represents the final action state .



* + **Synchronization:**
  + A fork node is used to split a single incoming flow into multiple concurrent flows. It is represented as a straight, slightly thicker line in an activity diagram.
  + A join node joins multiple concurrent flow back into a single outgoing flow.
  + A fork and join node used together are often referred to as synchronization.

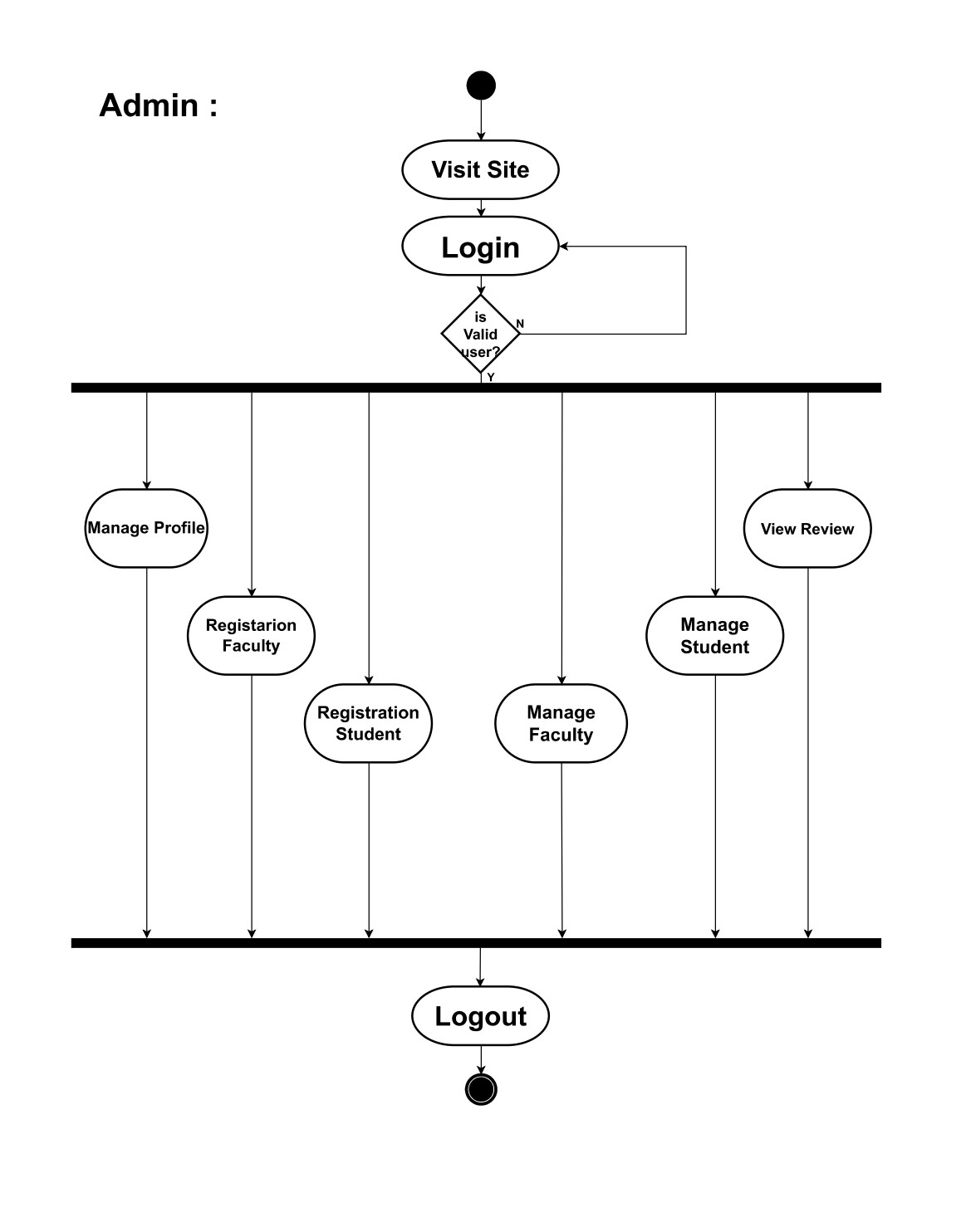


* + **Swim lanes:**
* Swim lanes group related activities into one column.
* An action transition takes place between two swim lanes.

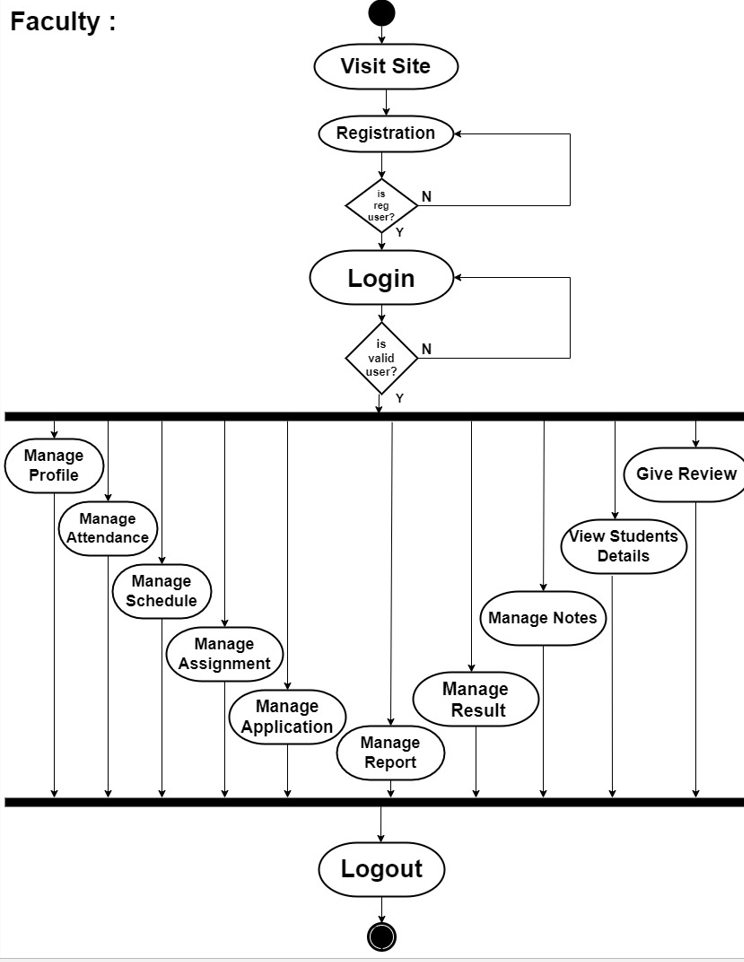


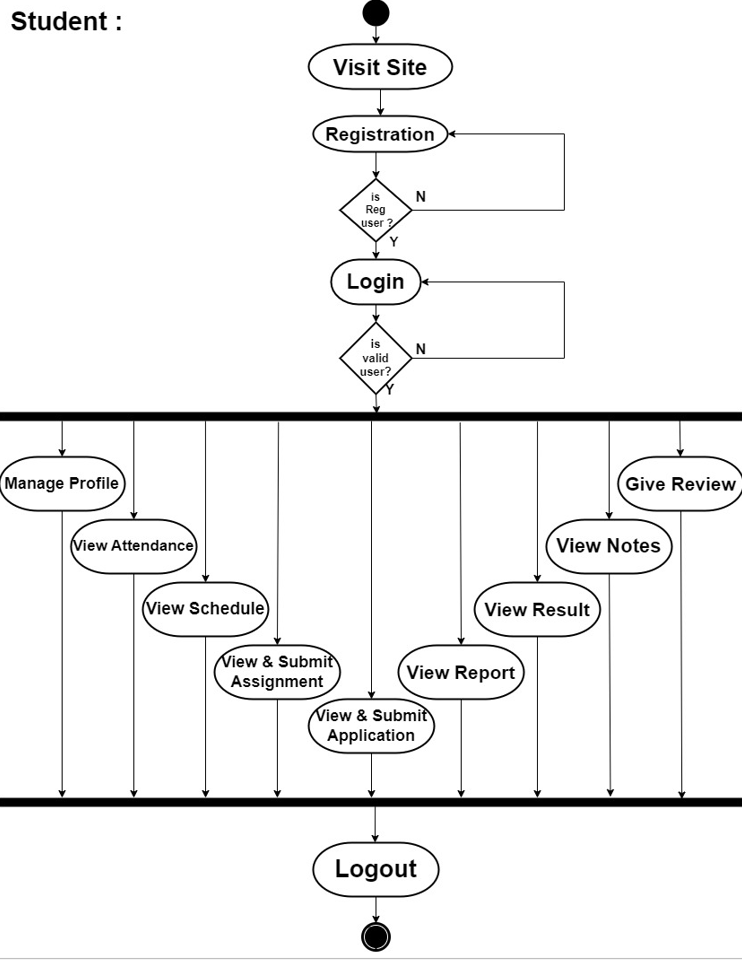
* + **Final State or End Point**
  + An arrow pointing to a filled circle nested inside another circle represents the final action state .



4.2.1 Admin

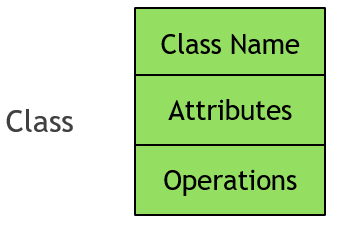
4.2.2Faculty:



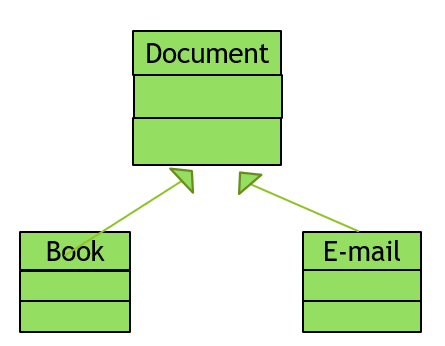
4.2.3 Student:

4.3 Class Diagram

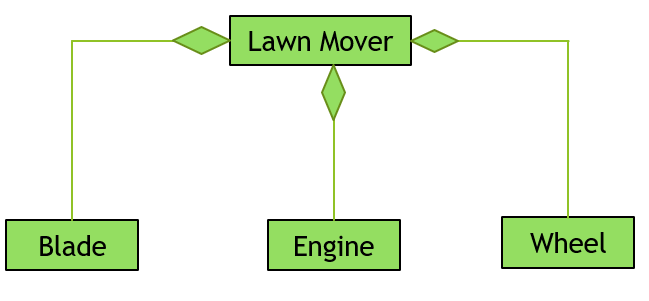
* **Classes :**
* classes represent an abstraction of entities with common characteristics. Association represent the relationships between classes.
* Class divided into three parts :top most part contain a class name. The middle part contain list of attributes and the last part contains list of operation.



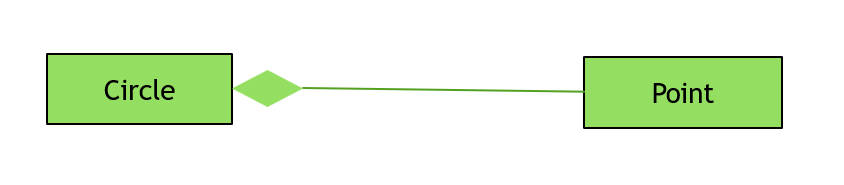
* **Relationship**
* **Association** :
* An association represents a conceptual connection between two or more classes .
* Association is the “glue” that ties a system and classes together .
* Association is denoted by a solid line .
* **Generalization**:
* Generalization is a relationship between a super class and one or more sub classes .It is used to indicate inheritance.
* Generalization is also known as “is - a” relationship .
* It is denoted by a solid line with hollow arrowhead .



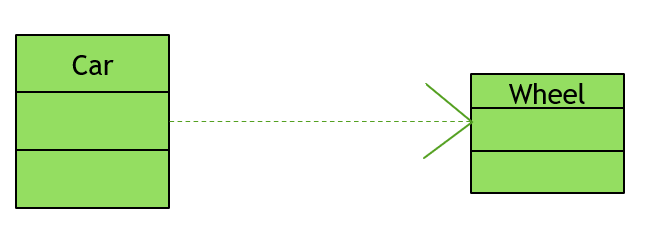
* **Aggregation:**
* Aggregation is a kind of association that represents a part-whole or part-of relationship.
* When a class is formed as a collection of other classes then it is called an aggregation. It is also known as “has-a” relationship.
* It is denoted by a solid line with hollow diamond which indicates the assembly end.



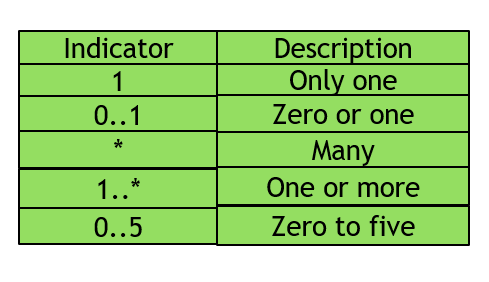
* **Composition**
* Composition is a stronger of aggregation. It is used to allocate and release memory.
* It has two constraints. It is denoted by black diamond.
* The arrowhead on the other end of the relationship denotes the relationship is possible in only one direction.



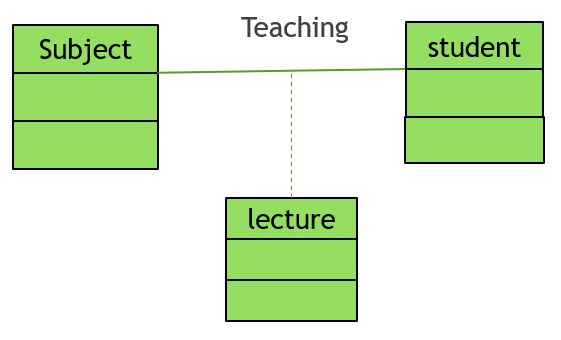
* **Dependency**
* Dependency indicated that one class is dependent on the other class.
* It is a week relationship between two classes.
* It is denoted by dotted line with open arrowhead .



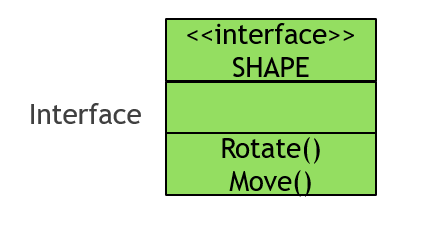
* **Multiplicity**
  + Multiplicity indicates the number of instances of one class linked to one instance of the other class.
  + Multiplicity notations are placed near the ends of an association .



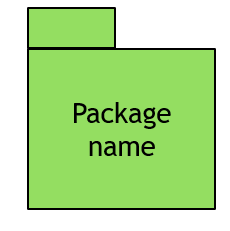
* **Association**
* An association class is a class that is part of an association relationship between two other classes.
* An association relationship to provide additional information about the relationship.

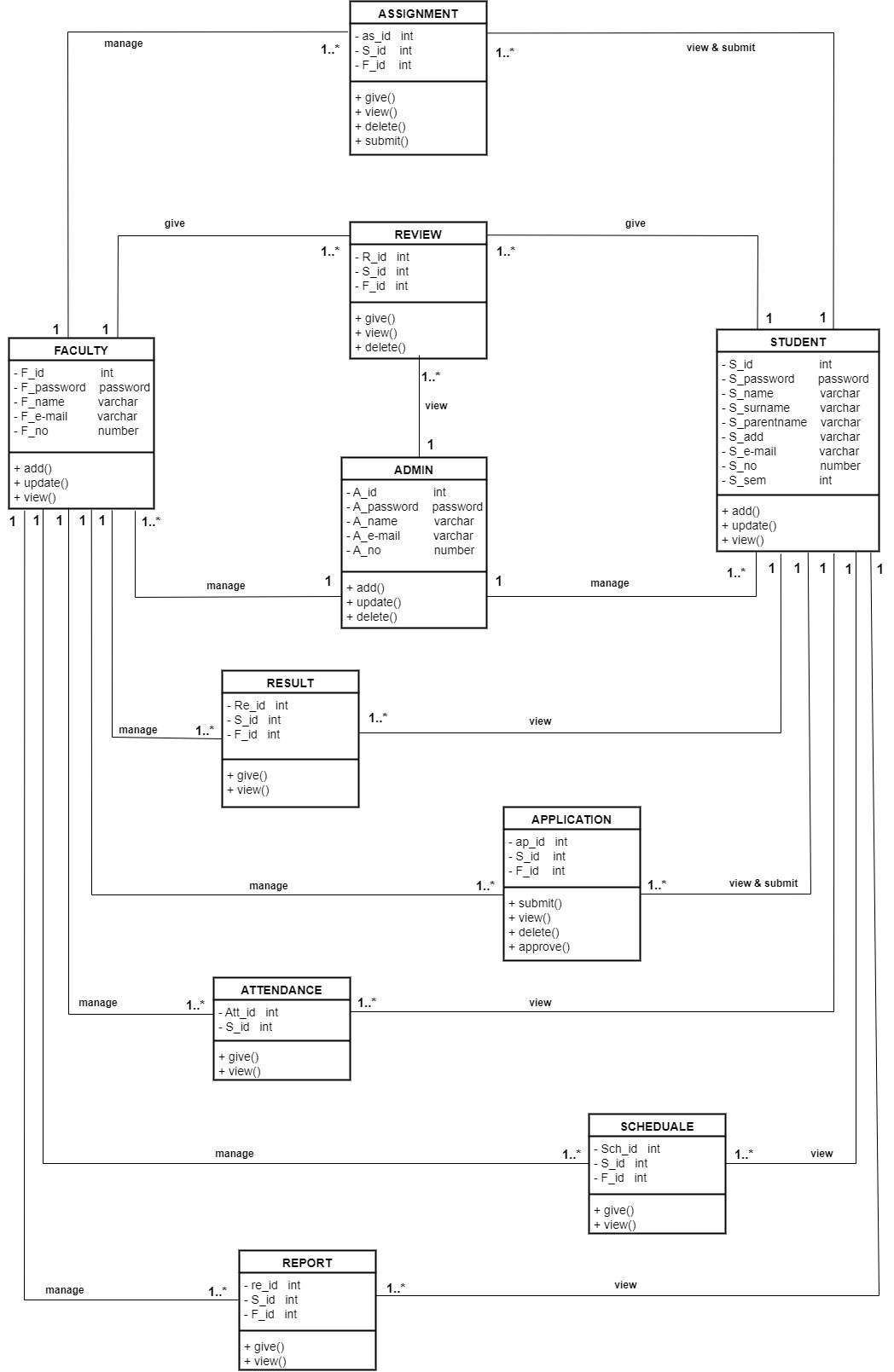


* **interface**
* An interface provide only definition of business functionality of a system.
* An interface share the same feature as a class like attribute and operations.
* An interface is denoted same as class except that it has a special denotation called a stereotype(<< >>).



* **Package**
* A package is a UML construct that enables you to organize model element, such as use cases or classes , into groups.
* A package provides ability to group together classes or interface or both that is related.





4.4 Sequence Diagram

4.4.1 Login

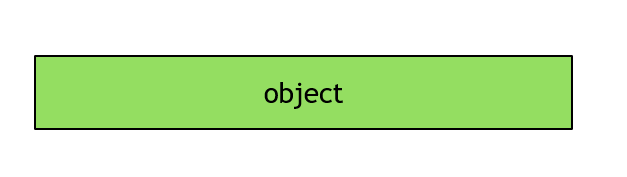
4.4.2 Registration

4.4.3 Application

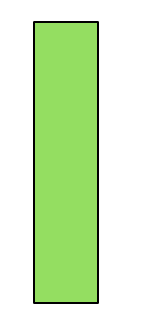
4.4.4 Review

***Sequence diagram notations:***

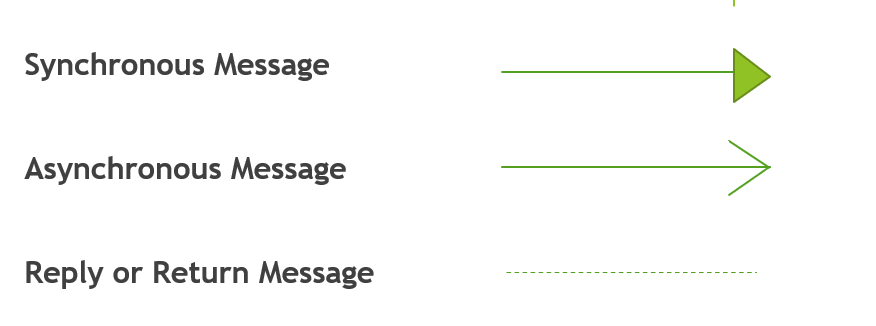
* **Class roles or participants:**
* Class roles describe the way an object will behave in context.
* Use the UML object symbol to illustrate class role ,but doesn’t list object attributes.



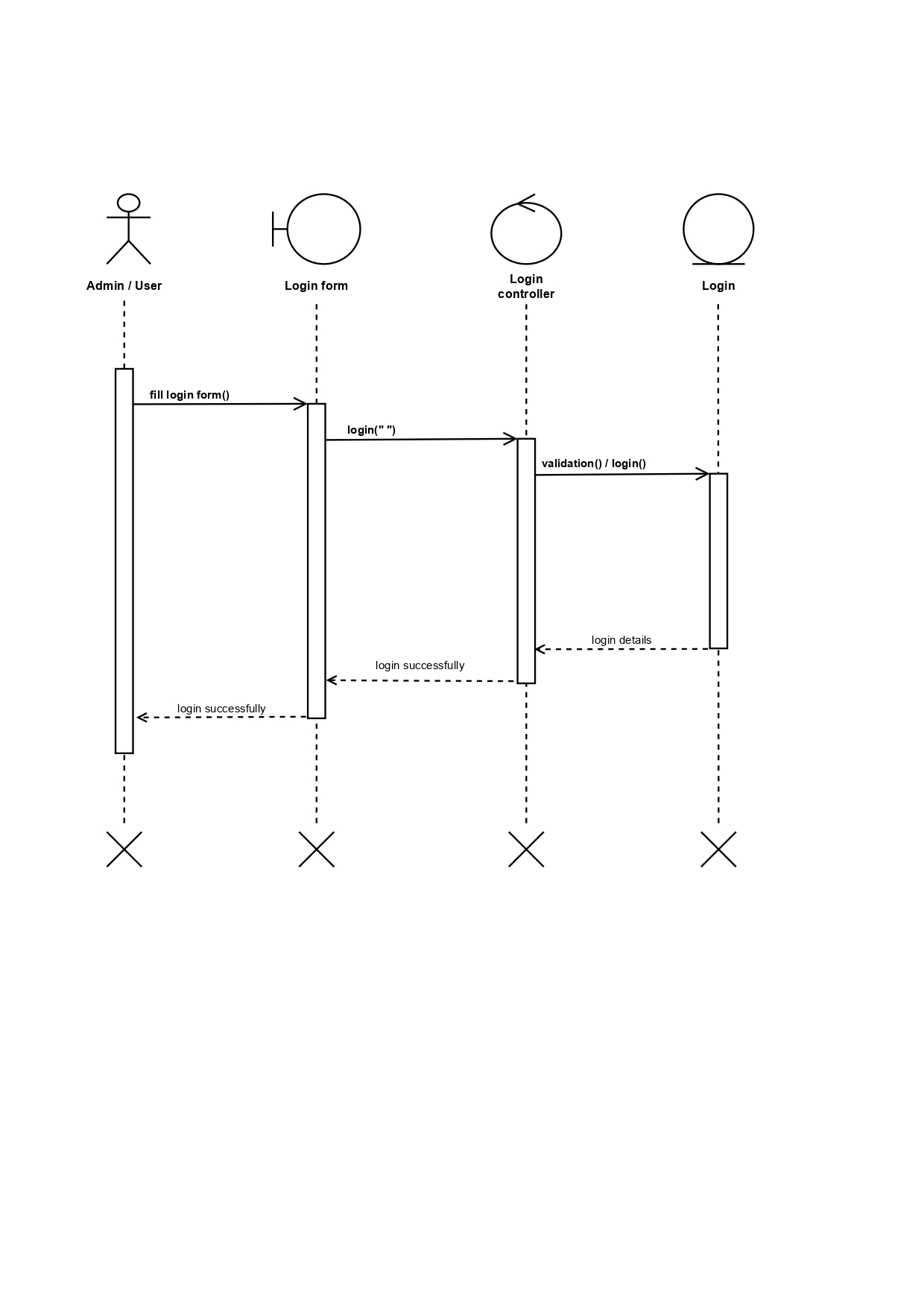
* **Activation or Execution Occurrence**
* Activation boxes represent the time an object needs to complete a task.
* When an object is busy executing a process or waiting for a reply message, use a thin gray rectangle placed vertically on its lifeline.

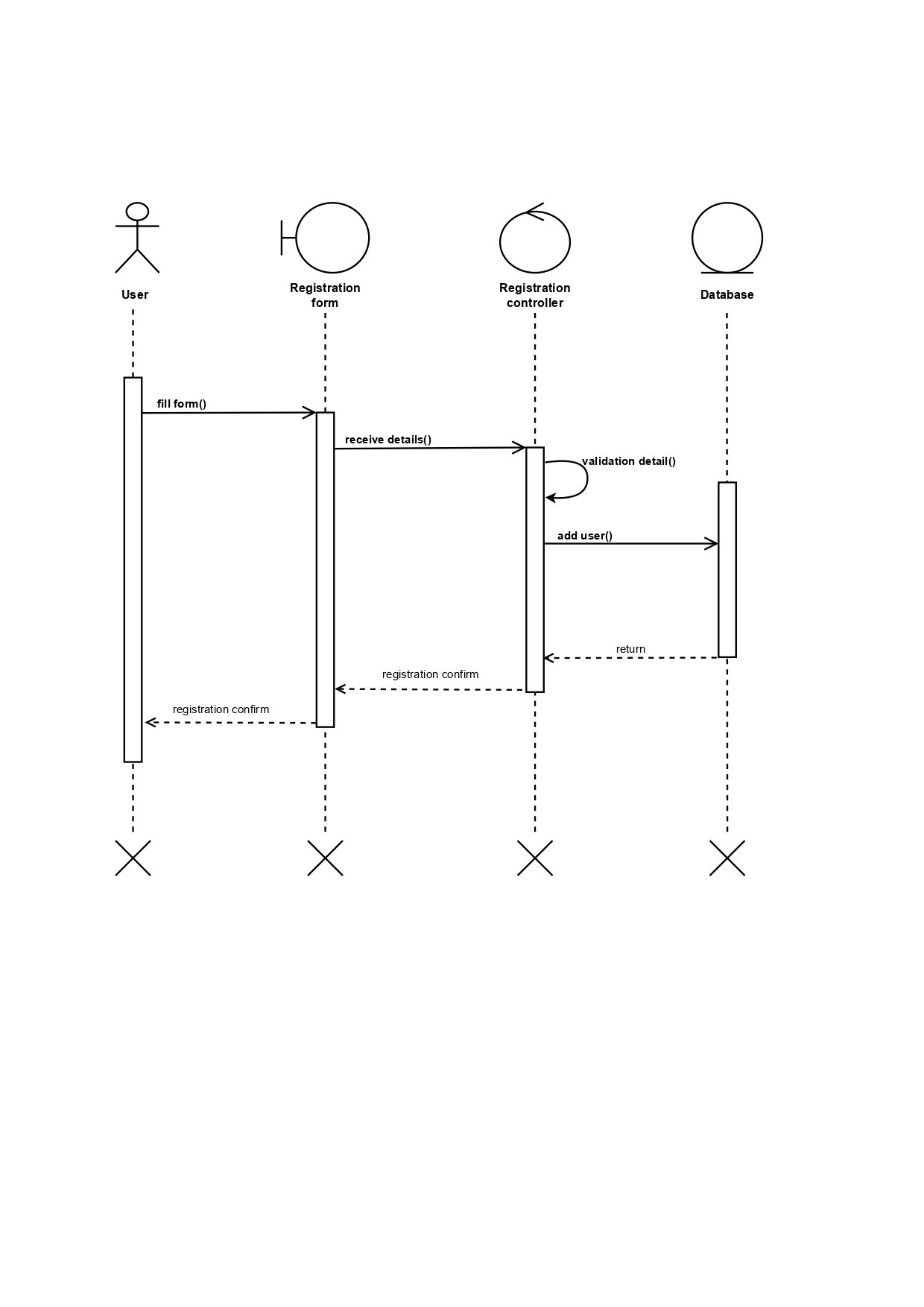


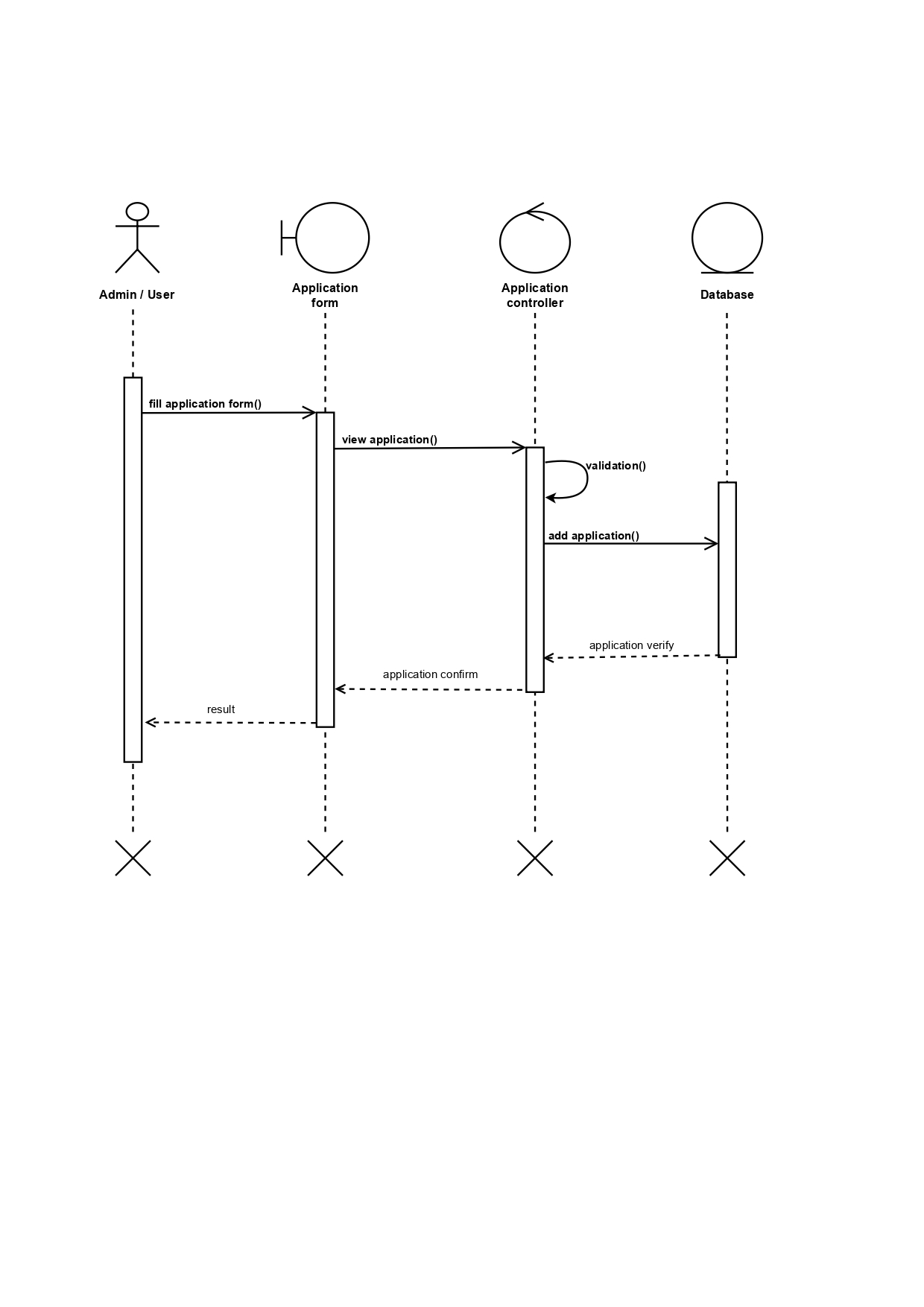
* **Message**
  + Message are arrows that represent communication between objects.
  + Use half-arrowed lines to represent asynchronous messages.
  + Asynchronous messages are sent from an object that will not wait for a response from the receiver before continuing its tasks.

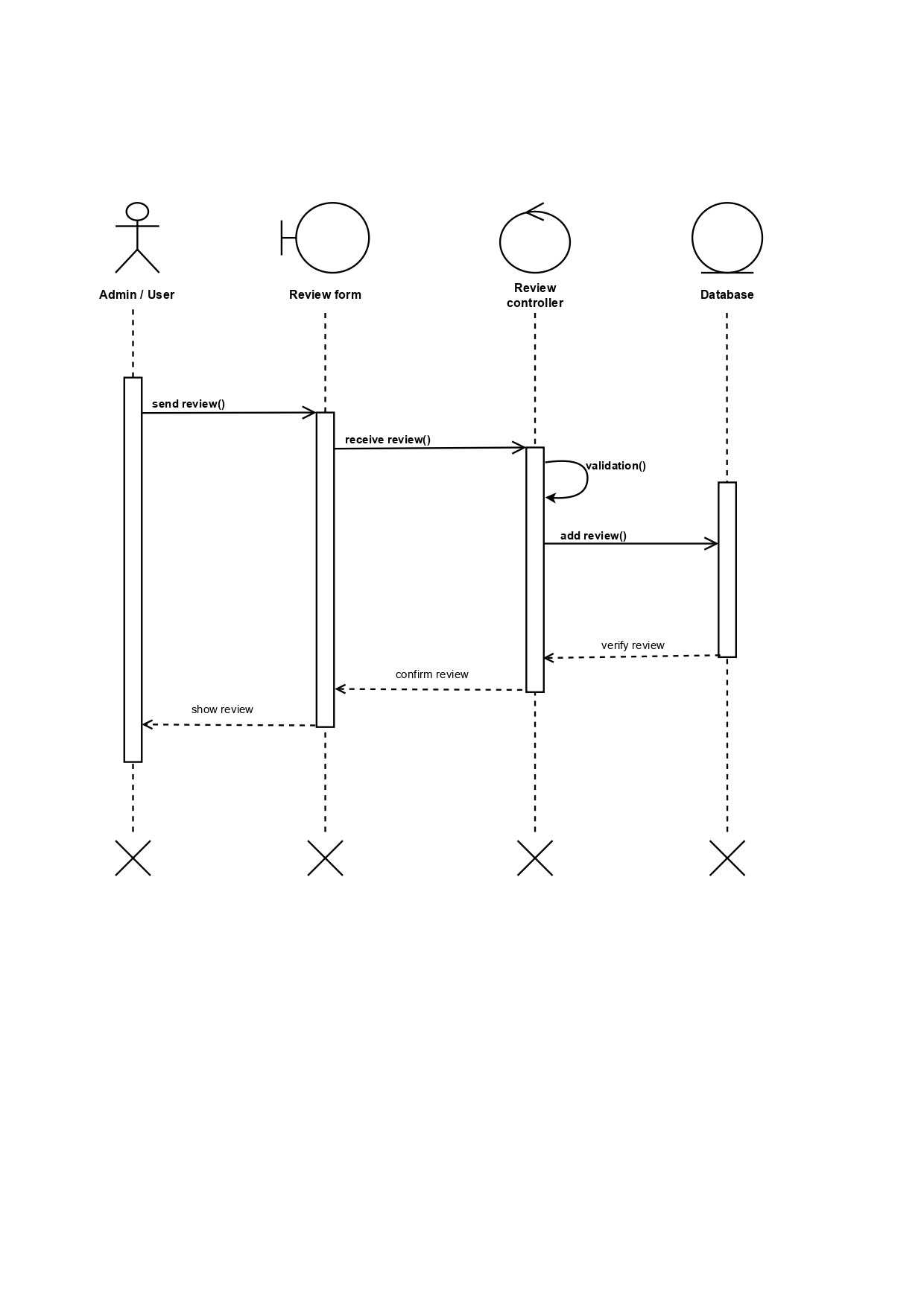


* **Loops**
* A repetition or loop within a sequence diagram is depicted as a rectangle.
* Place the condition for exiting the loop at the bottom left corner in square brackets [ ] .
* **Lifelines** :
* Lifelines are vertical dashed lines that indicate the object’s presence over time.

4.4.1 Login

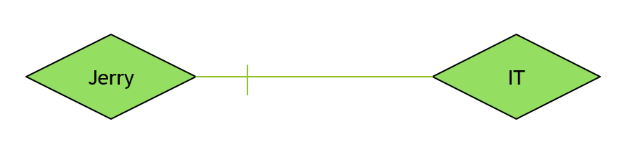
4.4.2 Registration

4.4.3 Application

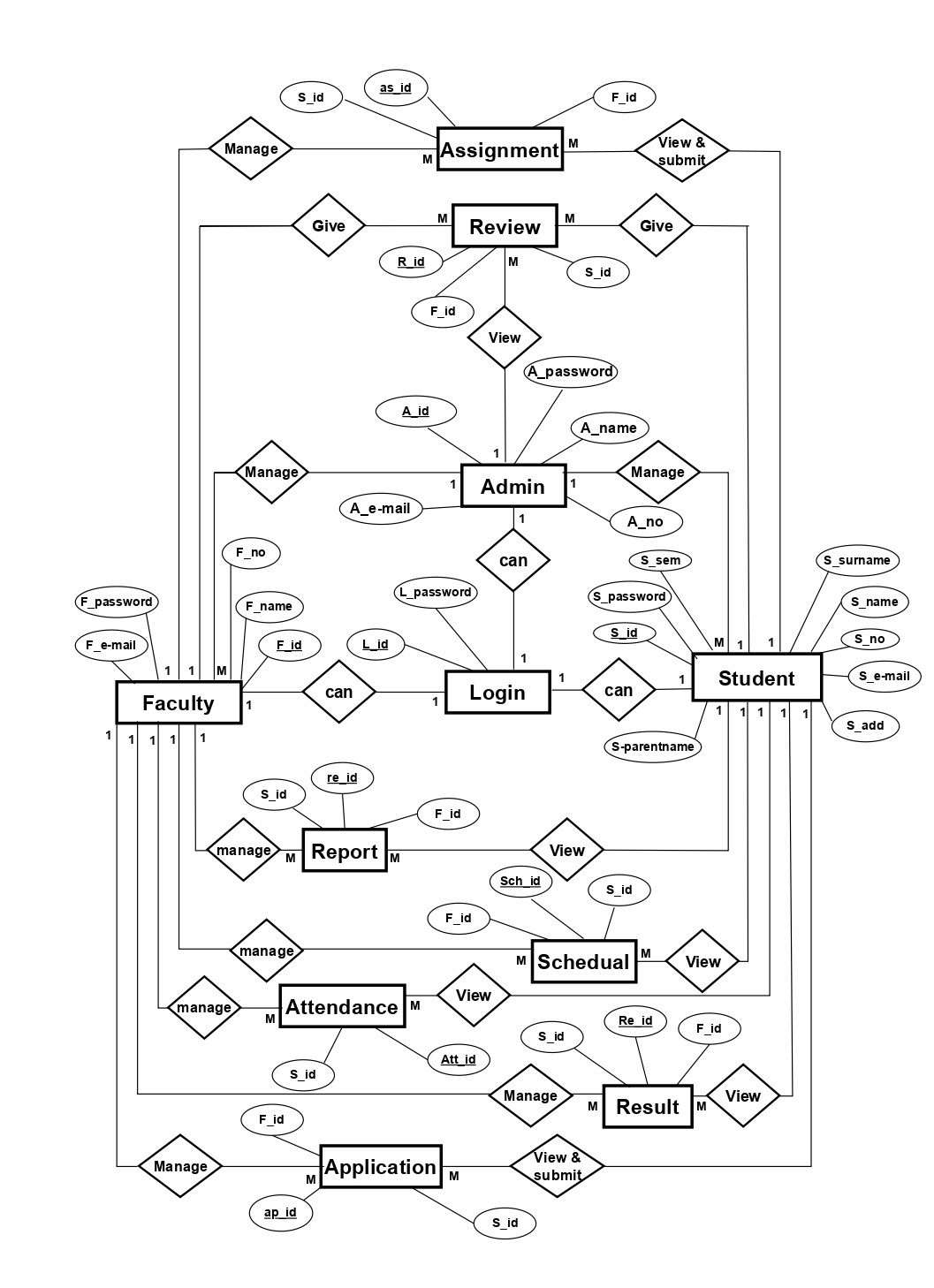
4.4.4 Review 

**5.Entity Relationship Diagram**

* **Entity:**
* This shape are independent from other entities, and are often called Parent Entity , since they will often have weak entities that depend on them.
* They will also have a primary key, Distinguishing each occurrence of the entity.
* **Relationships:**
* Relationships are association between or among entities.
* E.g., Jerry works in the IT department.



* **connectivity**
* The connectivity of a relationship describes the mapping of associated entity instances in the relationship.
* The values of connectivity are “one” or “many”.
* **Attributes**
* Attributes are characteristics of an entities , a many-to many relationships ,or a one-to-one relationship.
* An entity may contain any number of attributes .



**6.Data Dictionary**

1. **Admin :**

Table name : Admin

Table Description : Display the details about admin

Primary key : A\_id

Foreign key : -

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **NO**  **No** | **Field** | **Data type** | **Constrain** | **Size** | **Description** | **Example** |
| 1 | A\_id | Int | P.K. | 4 | Display admin id | 1 |
| 2 | A\_password | Varchar | Not null | 10 | Display admin email password | \*\*\*\*\*\*\* |
| 3 | A\_name | Varchar | Not null | 20 | Display admin name | Nehal |
| 4 | A\_no | Number | Not null | 10 | Display admin contact no | 1234567890 |
| 5 | A\_email | Unique | Not null | 20 | Display admin email id | Abc123@gmail.com |

**2)Faculty :**

Table name : Faculty

Table Description : Display the details about Faculty

Primary key : F\_id

Foreign key : --

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field** | **Data type** | **Constrain** | **Size** | **Description** | **Example** |
| **1** | F\_id | Int | P.K. | 4 | Display faculty id | 34 |
| **2** | F\_password | Varchar | Not null | 10 | Display faculty email password | \*\*\*\*\*\*\* |
| **3** | F\_name | Varchar | Not null | 20 | Display faculty name | Sonali |
| **4** | F\_email | Unique | Not null | 20 | Display faculty email id | Sonu12@gmail.com |
| **5** | F\_no | Number | Not null | 10 | Display faculty contact no | 4536789435 |

**3) Student :**

Table name : Student

Table Description : Display the details about Student

Primary key : S\_id

Foreign key : -

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **NO** | **Field** | **Data type** | **Constrain** | **Size** | **Description** | **Example** |
| 1 | S\_id | Int | P.K. | 4 | Display student id | 2 |
| 2 | S\_password | Varchar | Not null | 10 | Display student email password | \*\*\*\*\*\*\* |
| 3 | S\_sem | Int | Not null | 4 | Display student sem | 5th |
| 4 | S\_surname | Varchar | Not null | 20 | Display student surname | Patel |
| 5 | S\_name | varchar | Not null | 20 | Display student name | Nelvi |
| 6 | S\_parentname | Varchar | Not null | 20 | Display student parentname | Bhilmjibhai |
| 7 | S\_email | Unique | Not null | 20 | Display student email id | Aeg45@gmail.com |
| 8 | S\_no | Number | Not null | 10 | Display student contact no | 7865652314 |
| 9 | S\_add | Varchar | Not null | 50 | Display student address | 19,sahajanand baug soc,vastral,mahadevna-gar ahmedabad. |

**4) Application :**

Table name : Application

Table Description : Display the details about Application

Primary key : ap\_id

Foreign key : S\_id , F\_id

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field** | **Data type** | **Constrain** | **Size** | **Description** | **Example** |
| **1** | ap\_id | Int | P.K. | 4 | Display application id | 14 |
| **2** | S\_id | Int | F.K. | 4 | Display student id | 58 |
| **3** | F\_id | Int | F.K. | 4 | Display faculty id | 83 |
| **4** | Application | Text | Not null | - | Display application | abcdef@1324 |

**5) Assignment :**

Table name : Assignment

Table Description : Display the details about Assignment

Primary key : as\_id

Foreign key : S\_id , F\_id

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field** | **Data type** | **Constrain** | **Size** | **Description** | **Example** |
| **1** | as\_id | Int | P.K. | 4 | Display assignment id | 45 |
| **2** | S\_id | Int | F.K. | 4 | Display student id | 89 |
| **3** | F\_id | Int | F.K. | 4 | Display faculty id | 65 |
| **4** | Assignment | Text | Not null | - | Display assignment | @ghi3452 |

**6) Review :**

Table name : Review

Table Description : Display the details about Review

Primary key : R\_id

Foreign key : S\_id , F\_id

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field** | **Data type** | **Constrain** | **Size** | **Description** | **Example** |
| **1** | R\_id | Int | P.K. | 4 | Display review id | 23 |
| **2** | S\_id | Int | F.K. | 4 | Display student id | 45 |
| **3** | F\_id | Int | F.K. | 4 | Display faculty id | 67 |
| **4** | Review | Text | Not null | - | Display Review | Bhufd25612@ |

**7) Attendance :**

Table name : Attendance

Table Description : Display the details about Attendance

Primary key : Att\_id

Foreign key : S\_id

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field** | **Data type** | **Constrain** | **Size** | **Description** | **Example** |
| **1** | Att\_id | Int | P.K. | 4 | Display Attendance id | 12 |
| **2** | S\_id | Int | F.K. | 4 | Display student id | 30 |

**8) Schedule :**

Table name : Schedule

Table Description : Display the details about Schedule

Primary key : Sch\_id

Foreign key : S\_id , F\_id

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field** | **Data type** | **Constrain** | **Size** | **Description** | E**xample** |
| **1** | Sch\_id | Int | P.K. | 4 | Display schedule id | 8 |
| **2** | S\_id | Int | F.K | 4 | Display student id | 5 |
| **3** | F\_id | Int | F.k | 4 | Display Facluty id | 12 |

**9) Result :**

Table name : Result

Table Description : Display the details about Result

Primary key : Re\_id

Foreign key : S\_id , F\_id

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field** | **Data type** | **Constrain** | **Size** | **Description** | **Example** |
| **1** | Re\_id | Int | P.K. | 4 | Display result id | 1 |
| **2** | S\_id | Int | F.K | 4 | Display student id | 20 |
| **3** | F\_id | Int | F.K | 4 | Display Facluty id | 5 |

**10) Report :**

Table name : Report

Table Description : Display the details about Report

Primary key : re\_id

Foreign key : S\_id , F\_id

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field** | **Data type** | **Constrain** | **Size** | **Description** | **Example** |
| **1** | re\_id | Int | P.K. | 4 | Display Report id | 2 |
| **2** | S\_id | Int | F.K | 4 | Display Student id | 5 |
| **3** | F\_id | Int | F.K | 4 | Display Faculty id | 25 |

**THANK YOU**