### SOFTWARE DESIGN

**7.1 USE CASE DIAGRAM:**

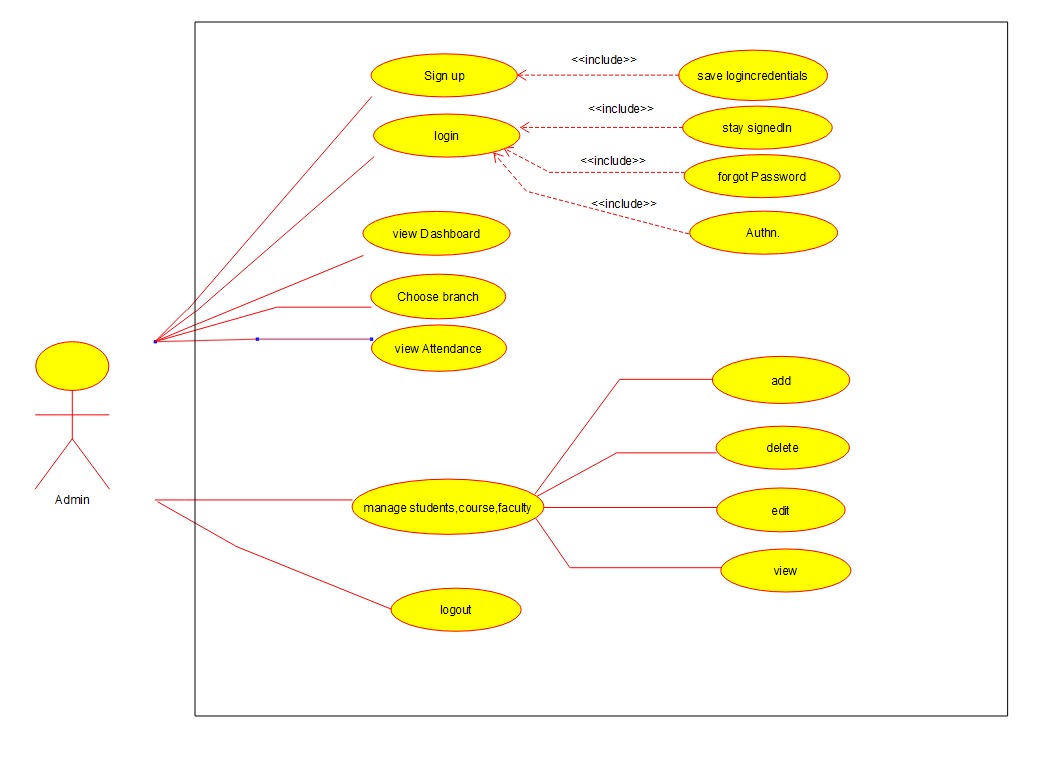


Fig 7.1.1 use case diagram for admin

The actor is a admin here. The admin first needs to register himself/herself. Then can login into the application if appropriate details is provided and it is a single sign in. As a logged-in admin, his dashboard will be choices of manage students, manage faculty, manage course and view attendance. The view attendance option is to view overall attendance of every class in his department so it would include list of classes for him to choose. Then he add, delete, edit or view students, faculty and course details. He can logout whenenver he wants and until he logouts, he will stay signed in even the application is closed.

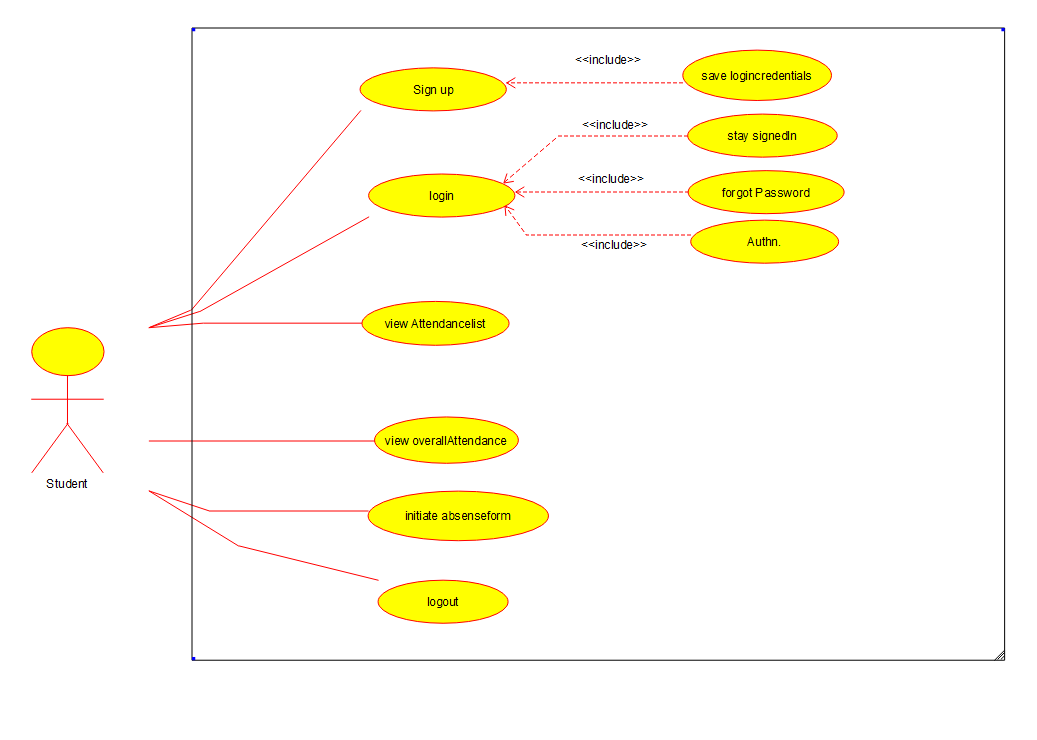
****

Fig. 7.1.2 use case diagram for student

Here actor is a student who is one of the primary stakeholder of the system. The student first needs to register himself/herself. Then can login into the application if appropriate details is provided and it is a single sign in. As a logged-in student, the student will view his overall attendance in his dashboard. Then he can able to view attendancelist provided as a option in the menu bar. He also initiate his absence as a form to the faculty he wants to. He will acquire his feedback in the email. He can logout whenenver he wants and until he logouts, he will stay signed in even the application is closed.

The last use case diagram is for describing the system of the faculty. So here actor is a faculty who is one of the primary stakeholders of the system. The faculty first needs to register himself/herself. Then can login into the application if appropriate details is provided and it is a single sign in. As a logged-in faculty, his dashboard will be list of classes for which he is a instructor. This also includes features of rescheduling class and add course through which he will enrol himself to the course. Then he will choose one of this course and take attendance to the students of the class. This feature also includes listing of absentees so that the faculty could confirm his attendance and submit to store in database. He can also add, delete, edit or view students. Then he can also view overall attendance of each student of a class and also view attendance list through which he can determine when each student can be absent on daily basis. He could also update the attendance later via the edit attendance list feature. The faculty could also view and give feedback on the absence-initiated form which is initiated by his students. He can logout whenenver he wants and until he logouts, he will stay signed in even the application is closed.

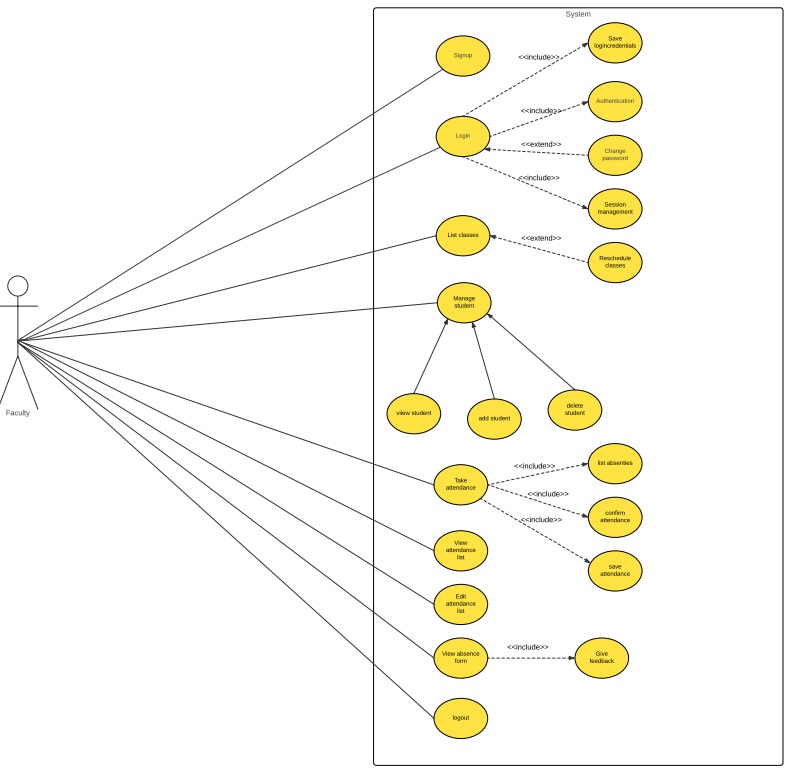
****

Fig. 7.1.2 use case diagram for faculty

**7.2 CLASS DIAGRAM:**

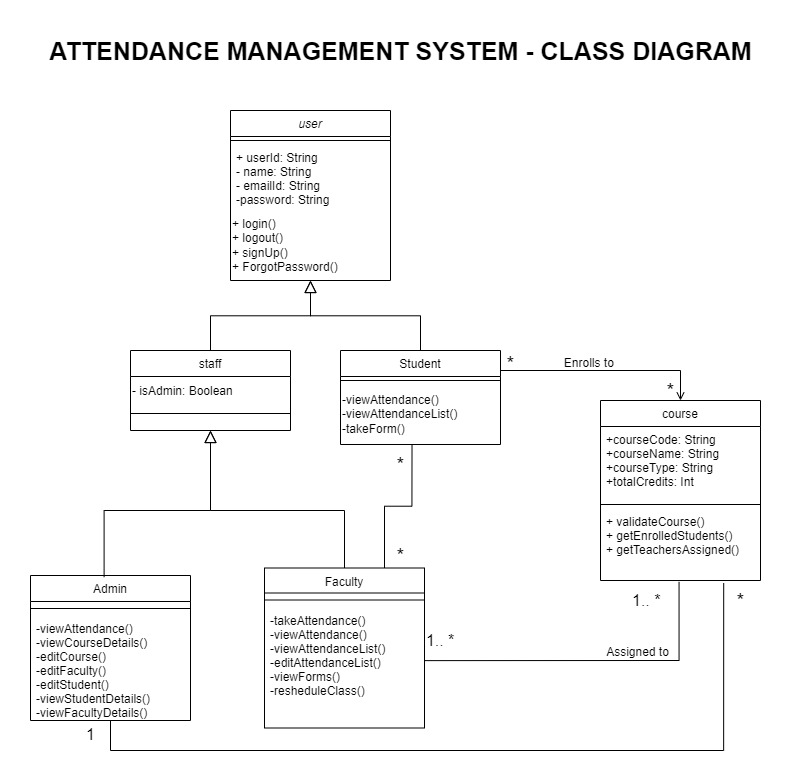


Fig. 7.2 class diagram of the system

DIAGRAM DESCRIPTION

UML class diagrams show the classes of a system, their inter-relationships, and the operations and attributes of the classes. It represents the static view of an application. The class diagram of this project has six different classes: user, staff, student, admin, faculty, course.

User class:

The User class has attributes like userl’, name, emailld, and password and operations such as logi n(), logout(), SignUp(), and ForgotPassword().

Staff class:

The Staff class has an isAdmin attribute.

Student class:

The Student class has operations like viewAttendance(), viewAttendanceList(), and takeForm(), and is associated with a course.

Course class:

The Course class has attributes like courseCode, courseName, courseType, and totalCredits.

Admin class:

The Admin class performs operations like viewAttendance(), viewCourseDetails(), and administrative tasks.

Faculty class:

The Faculty class manages tasks like takeAttendance(), viewAttendance(), viewing and editing attendance lists, view forms and rescheduling classes.

The relationship exists between classes are:

Generalization:

User class generalizes two subclasses staff and student.Staff class generalizes two subclasses admin and faculty.

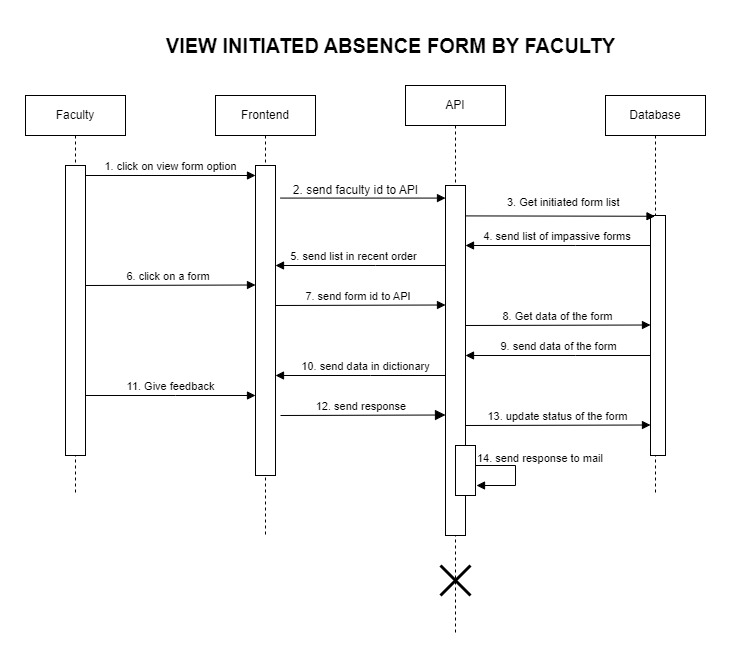
Association:

An association is any kind of relationship between instances of classes. In this student class has many to many association with faculty class.

**7.3 SEQUENCE DIAGRAM:**

7.3.1 SEQUENCE DIAGRAM 1 (VIEWING OF INITIATED FORM BY FACULTY)

When the faculty clicks on view form option, the frontend sends the faculty id to its API and the API gets and sends back name and id of the students who initiated the forms from the database to the course in which this particular faculty is a instructor. Then the user chooses a form which he wants to view or respond to. Then the frontend will send the id of this form to API and then the API will query database to get the details contained in the form and sends back to API. The API will send this detail to frontend to display to faculty. Then the faculty will accept or reject the form, once the feedback is provided, the status is updated in the database by the API and also the feedback to the student is send as email through external web server.

**** Fig. 7.3 sequence diagram 1 (viewing of initiated form by faculty)

7.3.2 SEQUENCE DIAGRAM2 (VIEWING ATTENDANCE ON DAILY BASES BY STUDENT):

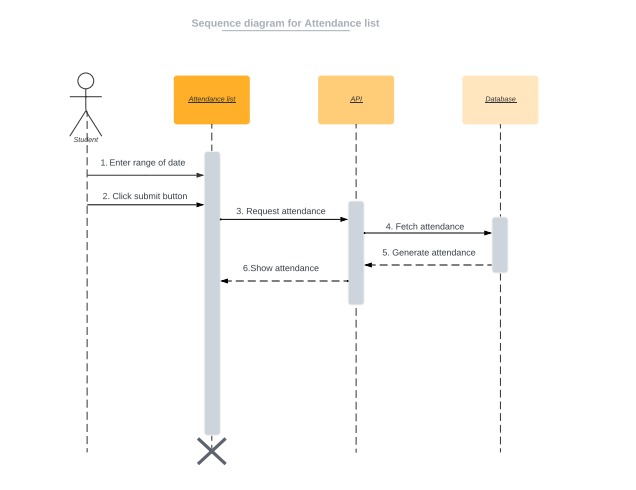


Fig 7.4 sequence diagram2 (viewing attendance on daily bases by student)

The attendance list is for the students to view their attendance status on daily basis. First the student is asked to give a range of date. The API is called to query database to get the attendance of this particular student on the given range of date. Then the attendance is generated and sent to frontend to display.

**7.4 STATE DIAGRAM**

First, once the admin opened the application, he is directed to dashboard where he finds his features. Next, when he click on the view attendance in the dashboard, he will be forwarded to a page where he finds a list of classes of his own department or for which he is an admin. Then when he click on a class, The students of that particular class is retrieved with amount of attendance they hold. Here the attendance is displayed in red when a student lacks the required attendance else in black indicating normal. In each page of the whole process, the admin can find a home button in the top right corner by which he can return back to the dashboard if he desires to.

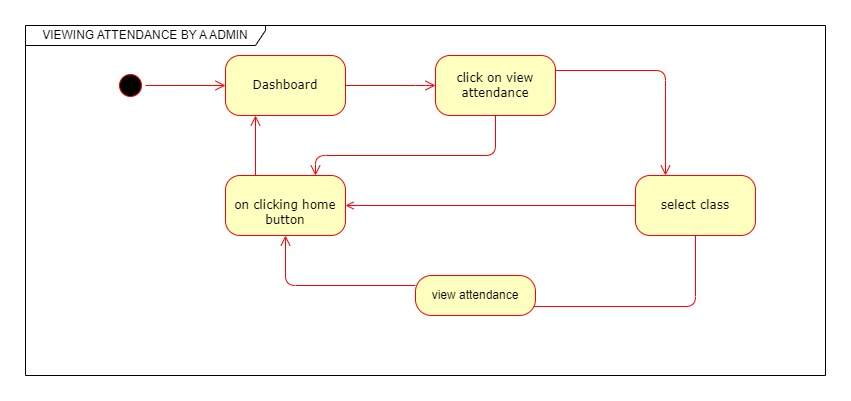


Fig. 7.5 state diagram 1 (viewing attendance of a class feature for admin)

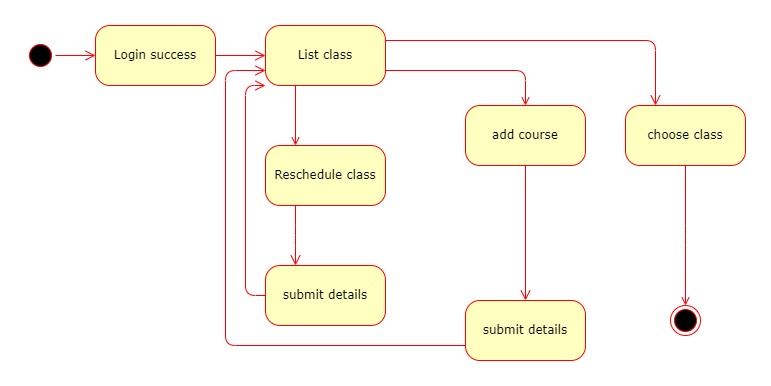


Fig. 7.6 state diagram 2 (viewing listed class for faculty)

The list class option in the faculty side is a page for him/her to choose for which class, he needs to take attendance or view overall attendance. The page begins with a login credentials through which his I'd is forwarded to display a list of classes for which he is a instructor. Then he chooses a class and proceed further. This page also includes a add course option by which the faculty enrolls to the courses for which he is a instructor. On clicking this feature icon in bottom right corner of the page, he is asked to fill the details and once he submitted it, the API throws error indicating there is no such course or the teacher already enrolled to the course, if not he is redirected to list class page indicating that he successfully enrolled to the course.

**7.5 ACTIVITY DIAGRAM**

### DIAGRAM DESCRIPTION:

The activity diagram begins with the user initiating the login process. A decision point follows, determining the authentication's validity. If the login is valid, the flow progresses through a series of activities, including viewing and editing faculty and student details, listing batches, semesters, and courses, managing attendance, and handling course details. In the case of invalid authentication, the system prompts the user to reattempt login. Each activity node represents distinct tasks such as viewing, adding, or editing various information. The diagram encapsulates the sequential steps from login to logout, covering the functionalities within the system.

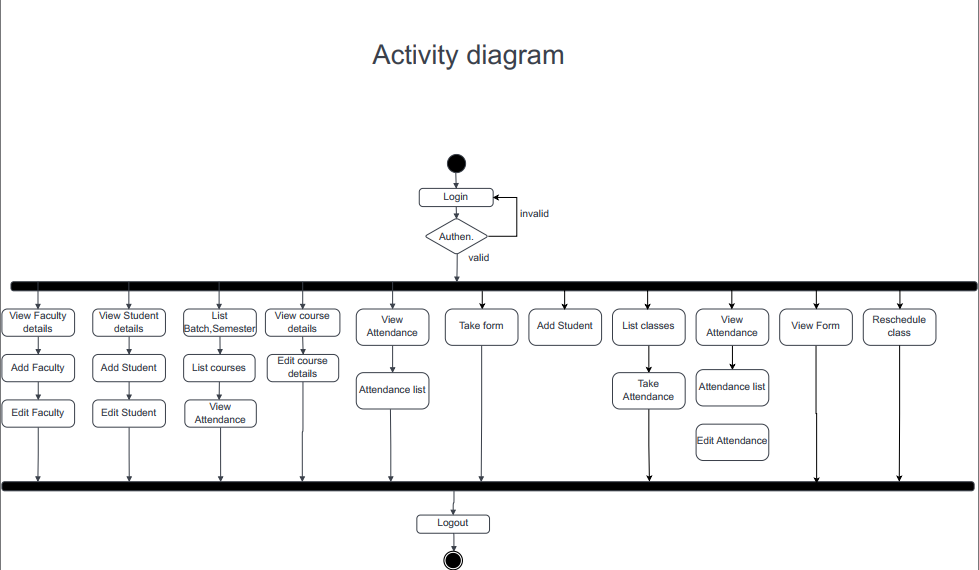


Fig. 7.7 Activity diagram of the system

**7.6 ENTITY RELATIONSHIP DIAGRAM**

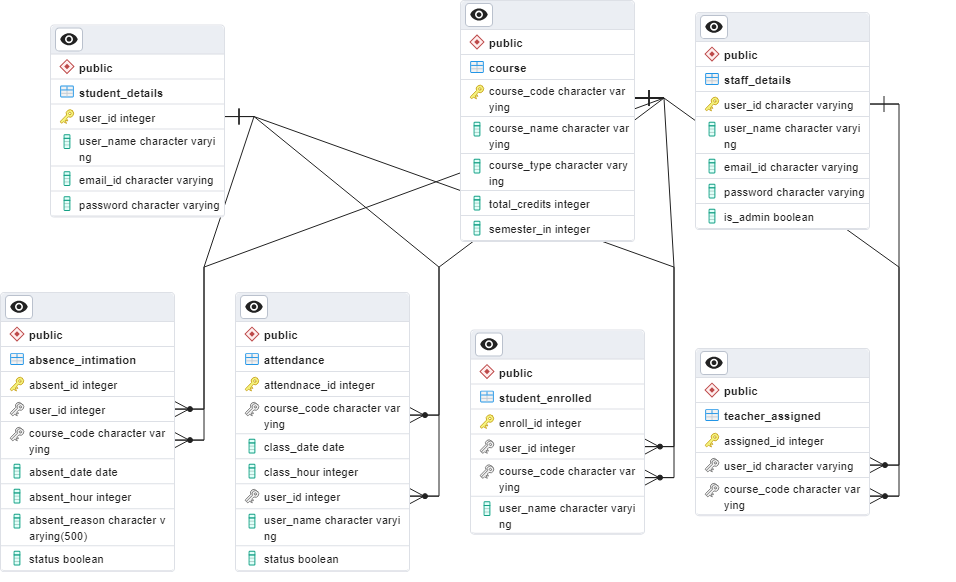
****

Fig. 7.8 ER diagram for the database of the system

### The student details table will contain the details of the students when they register.

### The staff details table will contain the details of the staff with admin attribute to indicate that the staff is a admin or not when a staff registers to the application. Then the course details contains about the details of the course that is taken in the department inserted by admin or faculty. Then the teacher\_assigned table will contain the data about which course is instructed by which faculty. Then the students\_enrolled table will contain the details about the students' registration to the courses he is enrolled in when he processes enrolment. The attendance table will contain the attendance when a faculty mark attendance students of a class with all the required details. Finally the absence intimation table will hold the data given as input by the student when he initiates his absence to a faculty.