# E-Prabandhak

Submitted in partial fulfillment of the requirements of

**PG Diploma in Advanced Computing**

By

| **1. Aniket Nale** | **- 210540381016** |
| --- | --- |
| **2. Jacob Patole** | **- 210540581049** |
| **3. Mahesh Kanojiya** | **- 210540581058** |
| **4. Rajat Yerigeri** | **- 210540381085** |
| **5. Rasika Potdar**  **6. Sameer Patil** | **- 210540381088**  **-210540581088** |

Guides:

**Mr. Chaitanya Mhatre**

Supervisor/Guide

#### Mr.Sohan More

Faculty Supervisor/Guide

# Centre for Development of Advanced Computing

**Kharghar/Juhu MAY- 2021**

**CERTIFICATE**

This is to certify that the project entitled **“E-Prabandhak”** is a bonafide work of **Aniket Nale (PL) (210540381016), Jacob Patole (210540581049), Mahesh Kanojiya(210540581058), Rajat Yerigeri(210540381085), Rasika Potdar(210540381085) and Sameer Patil(210540581088) etc.** submitted to C-DAC Mumbai in partial fulfillment of the requirement for the award of the Post Graduate Diploma in Advanced Computing.

**Mr. Chaitanya Mhatre Mr.Sohan More Supervisor/Guide Faculty Supervisor/Guide**

Declaration

I declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.



Mr. Aniket Nale -210540381016



Mr. Jacob Patole -210540581049



Mr. Mahesh Kanojiya -210540581058



Mr. Rajat Yerigeri -210540381085



Ms. Rasika Potdar -210540381088

--------------------------

Mr. Sameer Patil - 210540581088

Date: 07.09.2021.

Abstract

E-prabandhak is a digital platform which aims to enhance online education. Using this application admin and teachers easily upload information regarding the modules, courses, lectures etc.

Admin, teachers and students can access the data easily whenever required. Institutions or teachers can notify all the students at once.

This will improve the learning experience for students, they can focus more on their studies since they have to spend less time searching for the required information and content. Teachers can spend more time on educative and creative work as they have to spend less time on maintaining the records or finding the data.

### Contents

| **Chapter** | **Contents** | **Page No.** |
| --- | --- | --- |
| **1** | **INTRODUCTION:** | **1** |
| 1.1 | **Description** | **1 - 3** |
| 1.2 | **Problem Formulation** | **3** |
| 1.3 | **Motivation** | **3** |
| 1.4 | **Proposed Solution** | **3** |
| 1.5 | **Scope of the project** | **3 - 4** |
| **2** | **REVIEW OF LITERATURE** | **5** |
| **3** | **SYSTEM ANALYSIS** | **6** |
| 3.1 | **Functional Requirements** | **6** |
| 3.2 | **Non Functional Requirements** | **6 - 7** |
| 3.3 | **3.3 Specific Requirements** | **7 - 8** |

| **4** | **ANALYSIS MODELING** | **9** |
| --- | --- | --- |
|  | * 1. **Use-Case Diagrams and description**   2. **Activity Diagrams**   3. **Class Diagram**   4. **Sequence Diagram** | **9 - 14** |
|  | **4.5 Timeline Chart *(For the project duration) [Gantt]*** | **15** |
| **5** | **DESIGN** | **16** |
|  | **5.1 Data Modeling** | **16** |
|  | **5.2 Architectural Design (*Project Flow /architecture with description)*** | **17-18** |
|  | **5.2 User Interface Design** | **19-23** |
| **6** | **IMPLEMENTATION** | **24** |
|  | **6.1 Algorithms / Methods Used** | **24** |
|  | **6.2 Working of the project *(code for mentioned algorithms) [do not copy paste entire code. Only main snippets]*** | **24 - 32** |
| **7** | **TESTING *(white box /black-box / any testing algorithm used)*** | **33** |
|  | **7.1 Test cases *(conditions on which testing is done)*** | **33 - 35** |

|  | **7.2 Type of Testing used *(explanation and reason of testing method used)*** | **35** |
| --- | --- | --- |
| **8** | **RESULTS AND DISCUSSIONS *(final results or outputs)*** | **36** |
| **9** | **CONCLUSIONS & FUTURE SCOPE** | **37** |

Appendix Literature Cited

Publications by your group (if any) Acknowledgments

### List of Figures

| **Fig. No.** | **Figure Caption** | **Page No.** |
| --- | --- | --- |
| 1 | Use Case Diagram | 9 |
| 2 | Class Diagram | 10 |
| 3 | Activity Diagram for   1. User Registration 2. Admin 3. Faculty 4. Student | 11 - 12 |
| 4 | Sequence Diagram for   1. Admin 2. Faculty 3. Student | 13 - 14 |
| 5 | Timeline Chart | 15 |

**List of Tables**

| **Table No.** | **Table Title** | **Page No.** |
| --- | --- | --- |
| 1. | Registration | 17 |
| 2. | User | 17 |
| 3. | Module | 17 |
| 4. | Study\_Materials | 18 |
| 5. | Recorings | 17 |

**List of Abbreviations**

| **Sr. No.** | **Abbreviation** | **Expanded form** |
| --- | --- | --- |
| 1 | GUI | graphics–based user interface |
| 2 | HTML | Hypertext Markup Language |
| 3 | XML | Extensible Markup Language |
| 4 | SQL | Structured Query Language |
| 5 | ID | Identity Document |

# Chapter 1 Introduction

E-prabandhak is a digital platform which aims to enhance online education.

Since the advent of the internet world is becoming more and more digital in almost every domain of life and this has accelerated enormously since covid.

As we know during the covid times,all the education activity is happening online and all institutions big or small are conducting their batches online.Some major issues faced by students are not being able to find the recording of past lectures, notes given by faculty or some kind of study material.

Not being aware of notices or instructions passed by the institute as the institute uses multiple platforms such as whatsapp ,telegram,signal , etc to communicate with students so that it's easy for them to spread the information.

Even the administration, teachers and trainers/lab-faculties find it difficult to maintain students records of their performance databases, provide study materials, etc.

To solve such issues we are building a common platform E-prabandhak for the institute, teachers and students where students can access everything related to the course at one place and also the institute, teachers can easily update and find the relevant information .

E-prabandhak aims to provide a better learning environment for students, by decreasing the time and effort spent by students in searching for the required information and content. Thereby encouraging students to focus more on learning.

Teachers can spend more time on educative and creative work as they have to spend less time on maintaining the records or finding the data.

#### Description

* + - 1. **Product Perspective**

The software product is a standalone system and not a part of a larger system. The system will be made up of two parts, one running visible directly to the administrator on the server machine and the other visible to the end users, in this case the students and teachers through web pages. The three users of the system, namely the teachers,students and the admin interact with the system in different ways. The admin can add students and teachers .

The students can login and access the dashboard using the web interface provided.

Students can access the module name ,module syllabus and also see the recordings i.e the lecture recordings of all the modules.

Teachers can update the details of his/her module and can also see the list of the students.

#### Product Functions

1. Admin can view/create/update/delete ***teacher or faculty*** details.

2. Admin can view/create/update/delete ***module*** details.

3. Admin can view/create/update/delete ***student*** details.

4. Admin can view/create/update/delete ***recording*** details.

5. Admin can send notifications.

6. Teachers or faculty can view and update ***student*** details.

7. Teachers or faculty can view and update ***module*** details.

8. Teachers or faculty can view/create/update/delete ***recording*** details of the authorized module.

9. Teachers or faculty can send notifications.

10. Students can view ***module*** details.

11. Students can view ***recording*** details.

12. Students can get ***notifications.***

#### User Classes and Characteristics

The users can be divided into three main classes:

1. Admin 2. Teacher/faculty 3. Student

**Admin:** The responsibility of the Admin is to add teachers or faculties details and edit them. After adding Admin teachers or faculties will provide the user Id and password to teachers and faculties . Later on passwords provided by admin may be changed by teachers or faculties in this way the Admin can maintain records of teachers or faculties. It is the responsibility of the Admin to add students and edit their details. Admin has responsibility for adding modules and editing module details.

Admin can send notifications whenever required.

Admin should have a basic knowledge of using a computer and should have enough training on the application so as to navigate easily through different sections of the application. Admin should be aware of its user id and password.

**Teacher/faculty:** Teacher/faculty can view and edit the details of the student. Teacher/faculty can view and edit the details of the authorized module. Teacher/faculty can add/*delete/*edit the recordings. Teachers/faculty can send notifications whenever required.

Teachers/faculty should have a basic knowledge of using a computer and should have enough training on the application so as to navigate easily through different sections of the application. Teacher/faculty should be aware of its user id and password.

**Students:** Students can view module details for the required information. Students can view the recordings and also access the study material uploaded by the teacher/faculty. Notifications will keep students up to date with the recent important instructions or information.

Students should have a basic knowledge of using a computer and should be aware of their user id and password.

#### Operating Environment

The server should have Java installed on the machine, along with Java’s cryptographic packages. The dashboard server runs on a http server, that is “jsp” enabled. The browsers through which the students,teachers access the server should have minimal support for cookies and encrypted transactions.

#### Design and Implementation Constraints

Even though the students can access the lecture recordings at any time, they cannot record the screen or take screenshots. This constraint is imposed to ensure that only the genuine student is allowed to access the recordings.

#### User Documentation

##### Documentation for Admin:

A step by step cross-referenced tutorial like manual should be provided for the Admin in order to help him set-up a server and the batch management system on it. All features and GUI interface details should also be clearly provided in the document.

##### Documentation for Students/Teachers:

The user interface (through standalone application) is easy enough to use. But minimal instructions may be provided at the bottom of each web-page as an aid for the un-introduced.

#### 1.1.6 Assumptions and Dependencies

User side assumptions and dependencies:

PC (Personal Computer) or workstation with GUI. – A web browser with support for cookies Server side assumptions and dependencies:

A web server with GUI, Java and an http server installed.

#### Problem Formulation

The traditional method of managing a batch of students online has following drawbacks.:

* As we know during the covid times,all the education activity is happening online and all institutions (big or small) are conducting their batches online.Some common issues students face are:
* Not being able to find the recording of past lectures,not able to find notes given by faculty,not being aware of notice and instruction passed by institute as institute uses multiple platforms such as whatsapp ,telegram,signal to communicate.

#### Motivation

* As we already know that covid has brought a lot of changes in our lives and one of those changes is the way education is provided.Even though schools and colleges will reopen in future but online education is here to stay because if it’s uncountable benefits.
* So each and every organisation who aims to start it’s online batches will need an amazing portal to manage all its activities and to run its operations smoothly.

#### Scope

E-Prabandhak Online Batch Management System find it’s scope in various spheres of life:

* The project “E-Prabandhak” is based on creating a web application service so as to facilitate students and teachers.
* The project can be extended in providing a secure and efficient portal for the entire school or an entire college.
* Somehow the system is the best choice for conducting batches online.

#### SCOPE FOR FUTURE ENHANCEMENT

The E-Prabandhak platform can be made versatile by adding following functionalities

* Performance tracker
* Feedback
* Attendance Keeper

Performance Tracker is **an ongoing analysis of the students' performance in exams**.

Capturing students’ **attendance** data in real-time will make E-Prabandhak more versatile.

# Chapter 2

**Review of Literature**

A lot of practices are made to introduce the various platforms for online student management systems where different techniques and methodologies are used. Some of them guarantee confidentiality and security to the system to some extent, still the information and process that particular institute follows need to be controlled and managed with advanced systems that will ensure and guarantee the security and privacy of institutes’ as well as students’ information.

**2.1 Basic E-Prabandhak approach/architecture**

The systems that are developed to manage the information of the institutes by means of digital approach using online portals and electronic devices use various encryption and decryption techniques to guarantee the secure data transaction.

**Blackbaud Education Management**

Blackbaud school management software **helps schools deliver on that mission by connecting parents, teachers, and students to drive achievement, accountability, and safety.**

**Vidyalaya School ERP**

Vidyalaya is the powerful and topmost company to provide the solution for the students' management. This is the interactive platform for the students, teachers, parents, administration, and management. Students can submit their homework, assignments, and projects and get every announcements and news

Traditional databases are maintained by a single organization, and that organization has complete control of the database, including the ability to manipulate with the stored data, to censor otherwise valid changes to the data, or to add data fraudulently. For most use cases, this is not a problem since the organization which maintains the database does so for its own benefit, and therefore has no motive to falsify the database’s contents; however, there are other use cases, such as a financial network, where the data being stored is too sensitive and the motive to manipulate it is too enticing to allow any single organization to have total control over the database. Even if it could be guaranteed that the responsible organization would never enact a fraudulent change to the database (an assumption which, for many people, is already too much to ask), there is still the possibility that a hacker could break in and manipulate the database to their own ends.

# Chapter 3 System Analysis

## Functional Requirements

* Students will register to the portal after taking admission in the batch and then get access to the materials.
* If a student misses or wishes to see the recordings of the lectures then he can view the recordings.
* If students want to check the names of subjects and their syllabus then he/she can do that

## Login of Admin

* The system will allow the administrator to select, add, delete, update teachers and students.
* The system will allow the administrator to view/create/update/delete ***module*** details.
* The system will allow administrators to view/create/update/delete ***student*** details..
* The system will allow the administrator to view/create/update/delete ***recording*** details.
* The system will allow the administrator to send notifications.

## Login of Teacher:

* The system will allow the teacher to view and update ***student*** details..
* The system will allow the teacher to view and update ***module*** details
* The system will allow the teacher to view/create/update/delete ***recording*** details of the authorized module.
* Teachers or faculty can send notifications

**3.1.3 Login of Student:**

* The system will allow students to view ***module*** details.
* The system will allow students to view ***recording*** details.
* The system will allow student to view ***notifications***

## Non-functional Requirements

* + 1. **Performance Requirements**

The system should store all the database records of each administrator, teacher/faculty and student. The application should be available for specific users only. Likewise, the application should be user-friendly through an appropriate user interface so that it is easy for users to understand. For the convenience of the user, the option should be placed in an appropriate place and in easily understandable language.

**3.2.2 Safety Requirements**

1. In order to prevent data loss in case of system failure, the user’ information must be saved in the database, for the system to keep information secure.
2. In case the Admin detects any security lapse in the system, he should be able to shut down the server and close all connections immediately while preserving the already stored data.
3. The system should be capable of gracefully recovering from earlier crashes and keep all the information safe.
   * 1. **Security Requirements**

Passwords of the Admin, teachers and students should be protected for privacy using whatever constraints required in the database or the application. User’s password should be saved in encrypted format so that the intruder cannot know the password of the user. All passwords should be stored as a secure hash of the administrator password.

##### Data Security

The education institute will not like that students take screen recordings of the lectures or take screenshots and try to misuse the information or circulate the lectures in their private groups hence stand alone application has been built where students cannot capture the screen while watching the recordings.

Also students cannot share the links of the study materials and notes anywhere as right click is disabled.

* + 1. **Software Quality Attributes**

#### Usability:

The user interface of the application is very easily understandable.

#### Transparency:

Users should be able to possess confidentiality of the institute.

#### Accuracy:

The system shall record and keep all the users and shall do so correctly.

#### Eligibility:

Only authorized users, who are students of the institute, should be able to register.

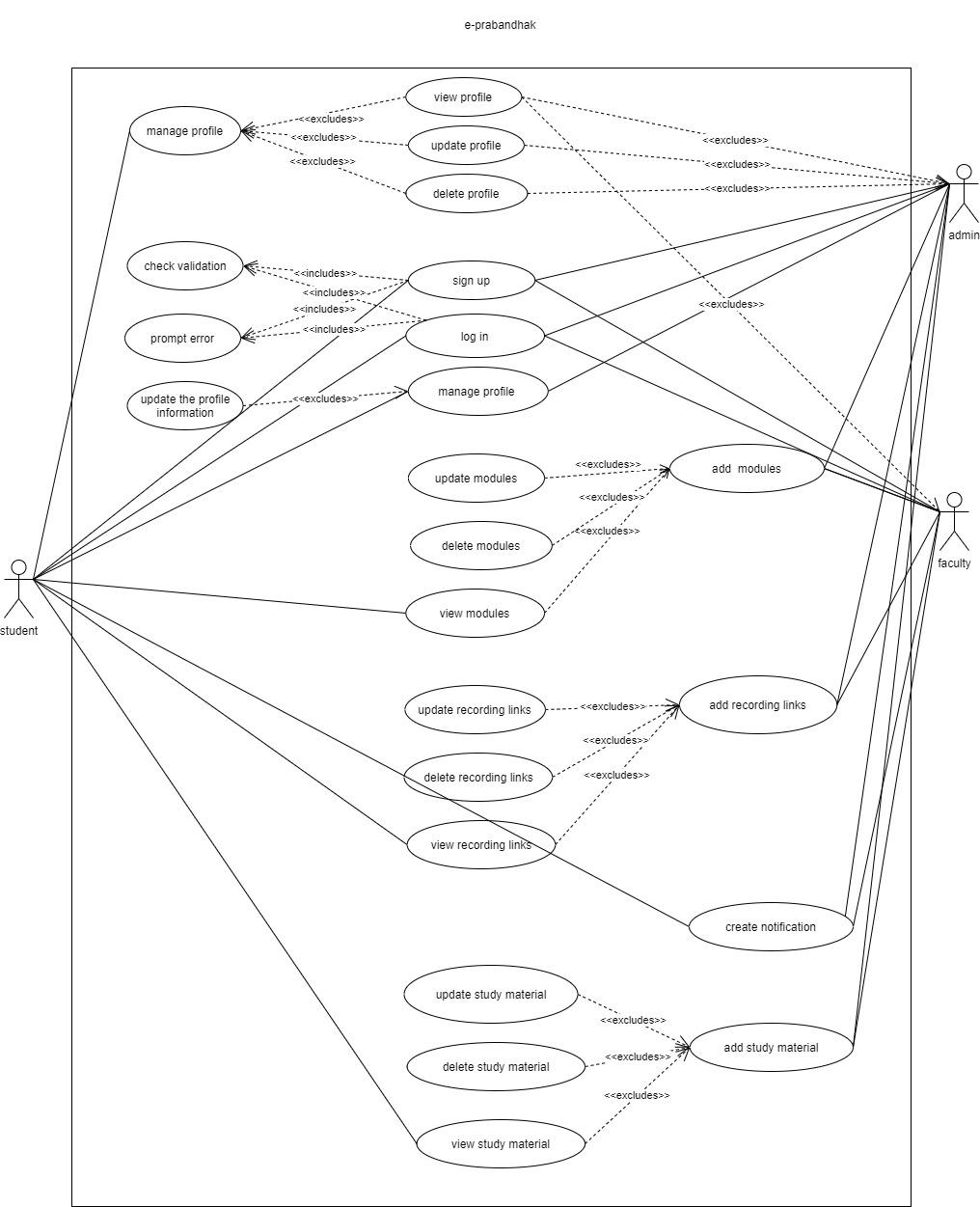
#### Cost-effectiveness:

Student Management systems should be affordable and efficient.

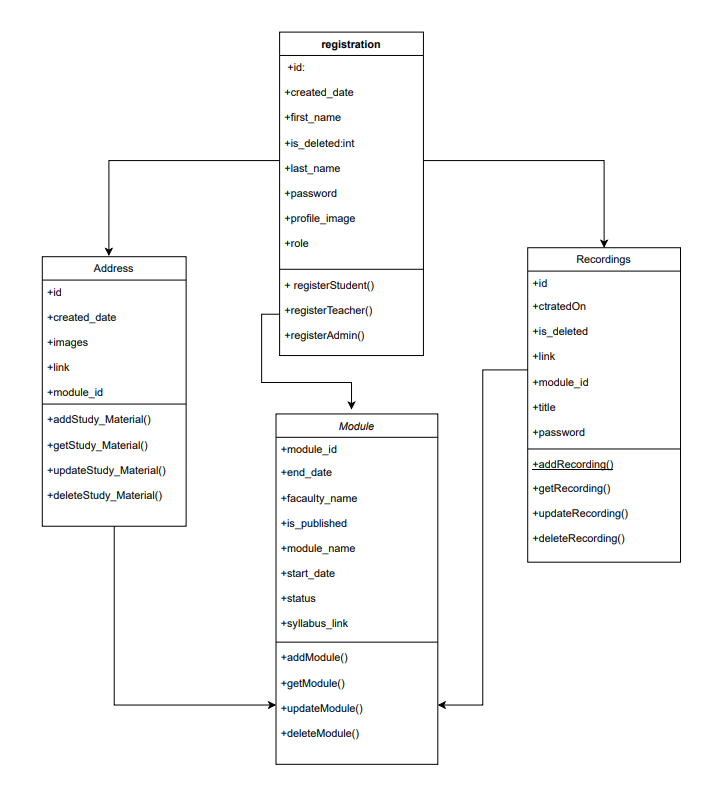
# Chapter 4

# Analysis Modeling

## Use Case Diagram: -

Fig-4.1.Use case diagram of E-Prabandha

## 4.2 Class Diagram : -

Fig.4.2-Class diagram of E-Prabandhak

## 4.3 (A) Activity Diagram For User Registration: -

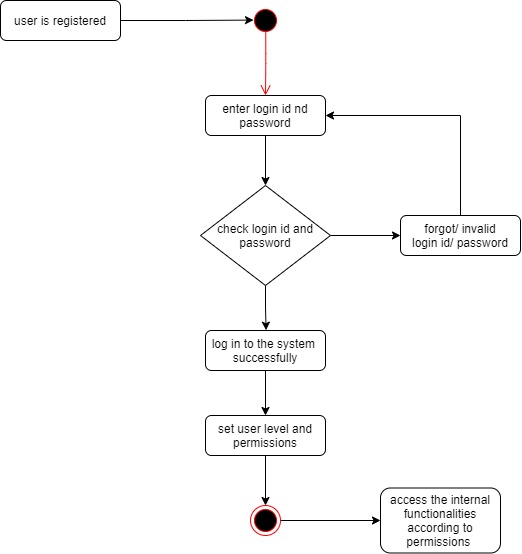
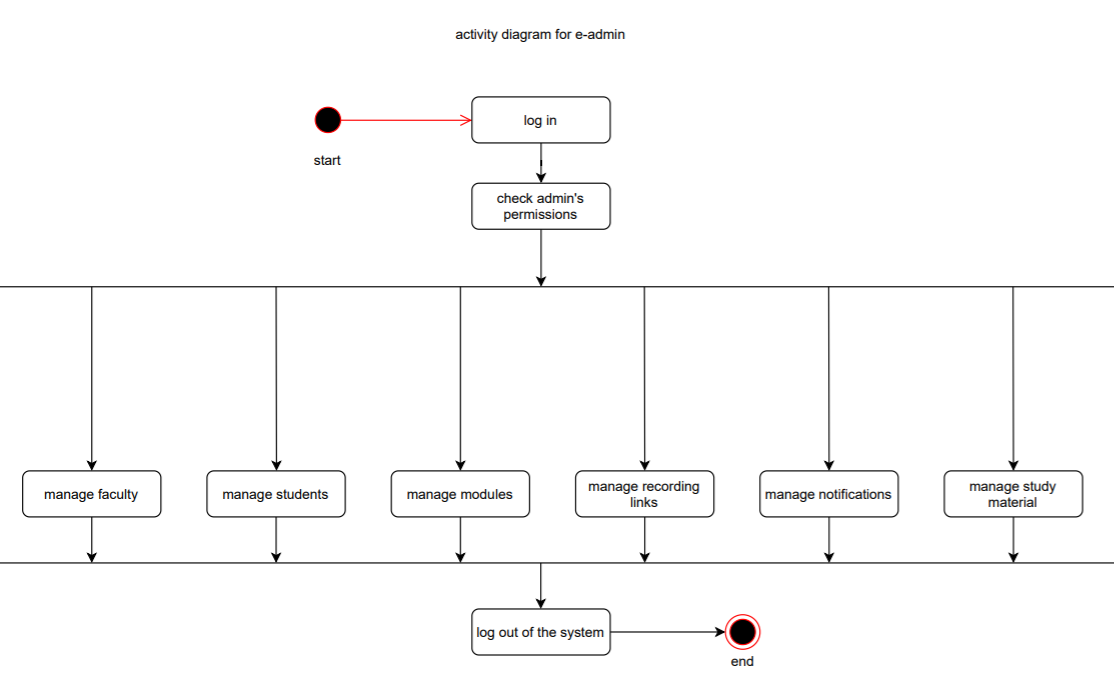


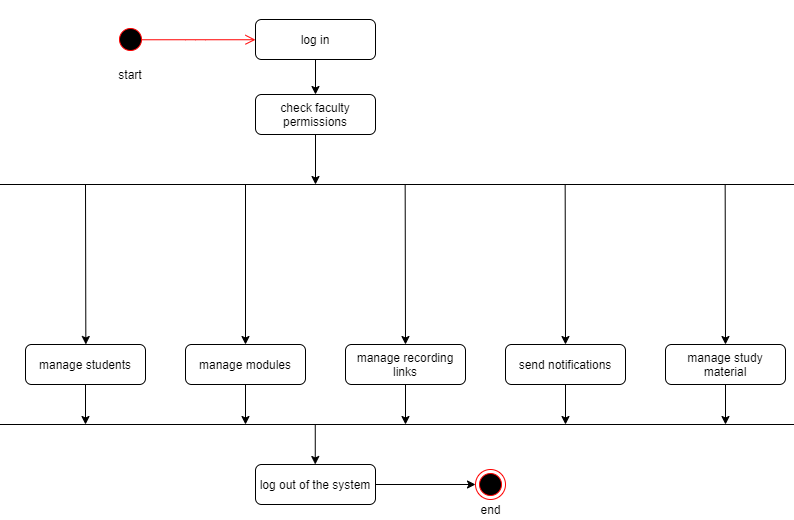
Fig-4.3.(a) Activity Diagram for User Registration

## (B) Activity Diagram For Admin:

Fig-4.3 (b) Activity Diagram for Admin: -

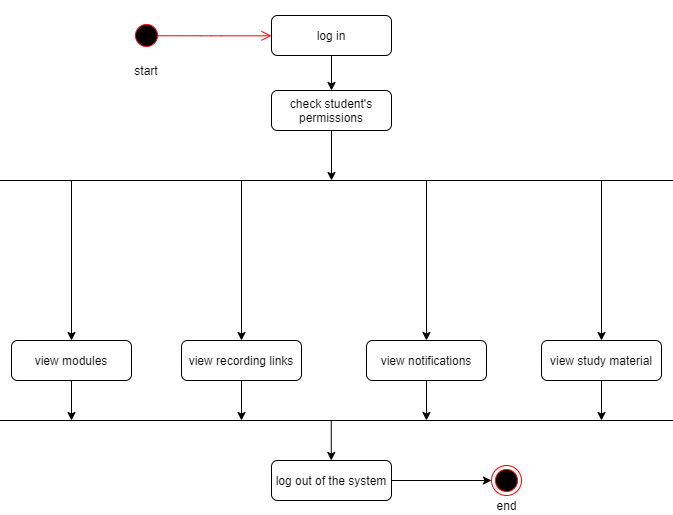
**4.5 (C) Activity diagram for Faculty**

Fig-4.3 (b) Activity Diagram for Faculty: -

****

**4.5 (C) Activity diagram for Students**

Fig-4.3 (c) Activity Diagram for Students: -



## Sequence Diagram : -

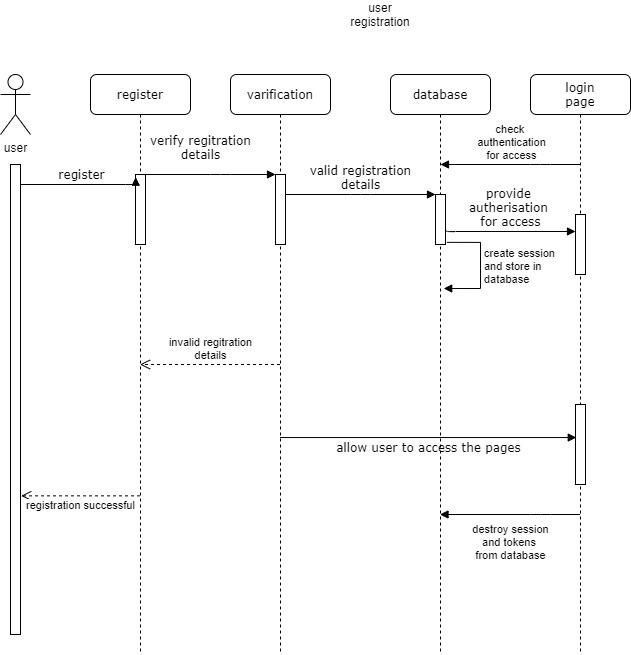
Fig-4.4 (a) Sequence Diagram for registration 

Fig-4.4 b) Sequence Diagram for admin

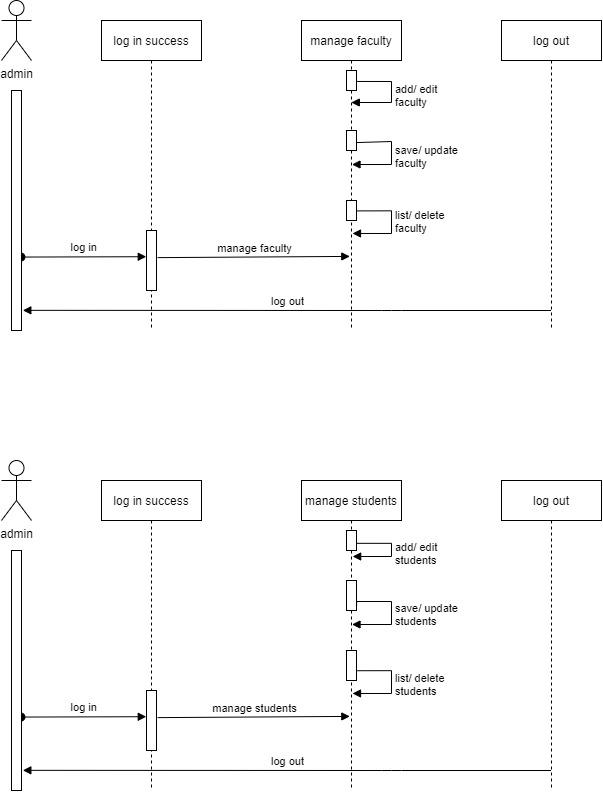


Fig-4.4 (b) Sequence Diagram for admin/ Faculty

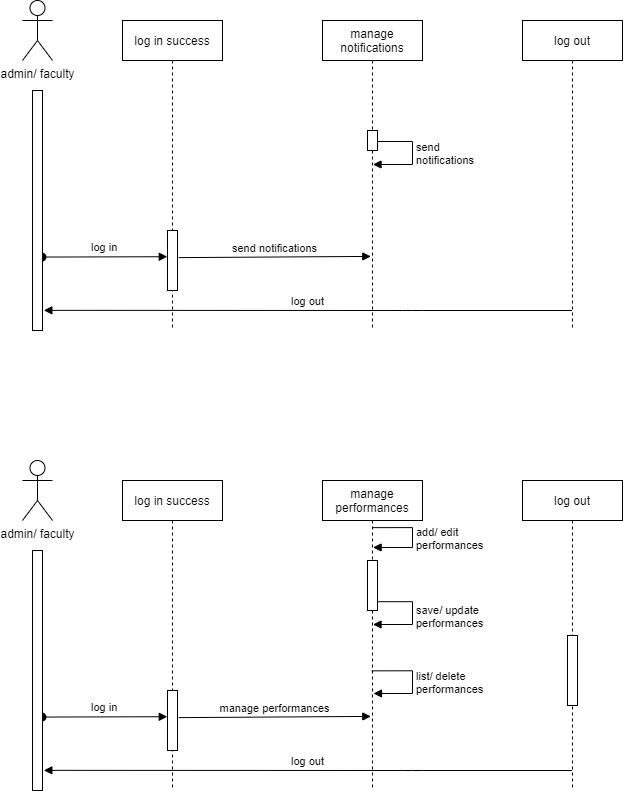


Fig-4.4 (c) Sequence Diagram for admin/ faculty

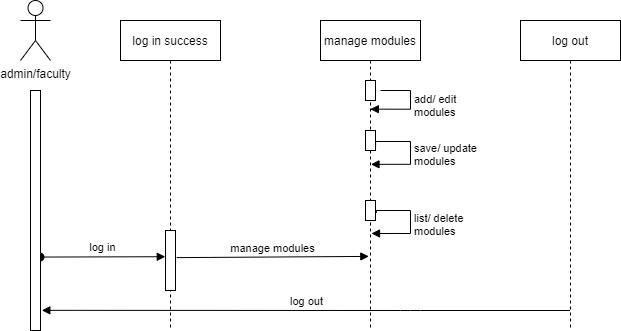


Fig-4.4 (c) Sequence Diagram for admin/ faculty

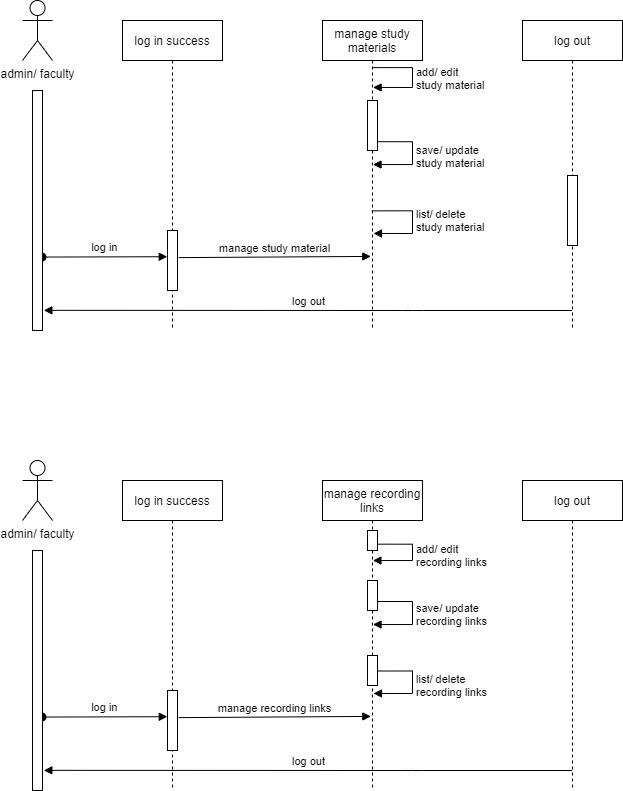


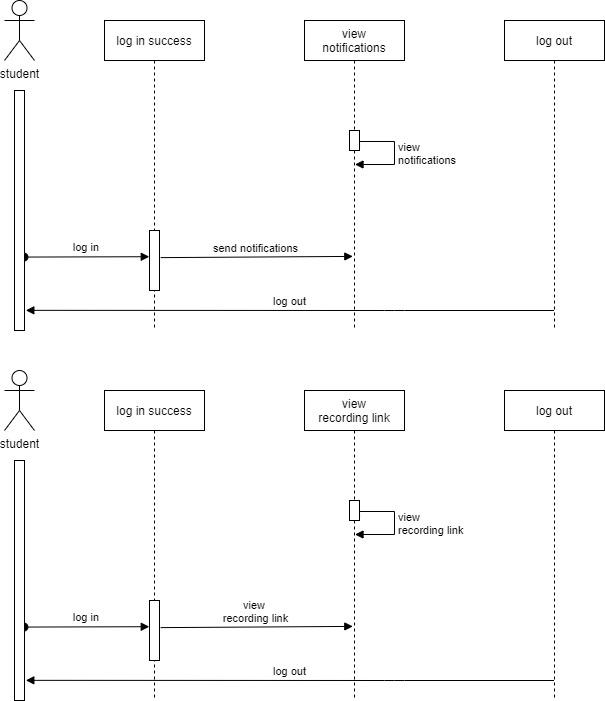
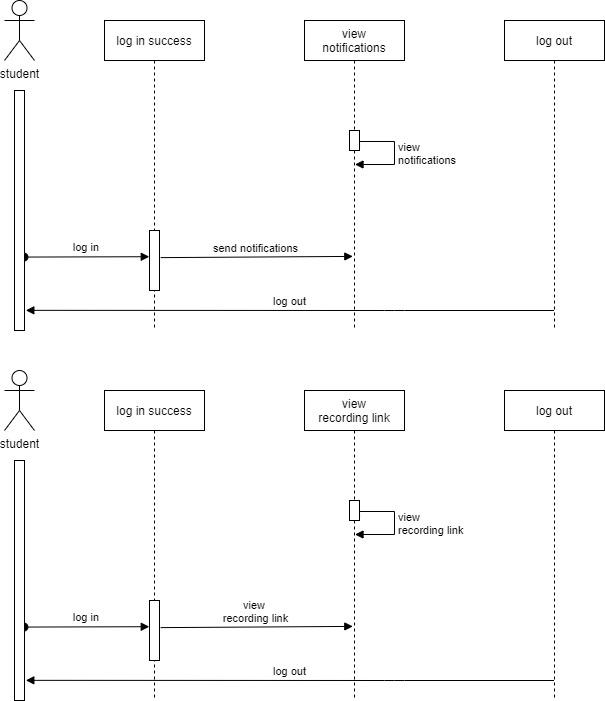
Fig-4.4 (d) Sequence Diagram for students

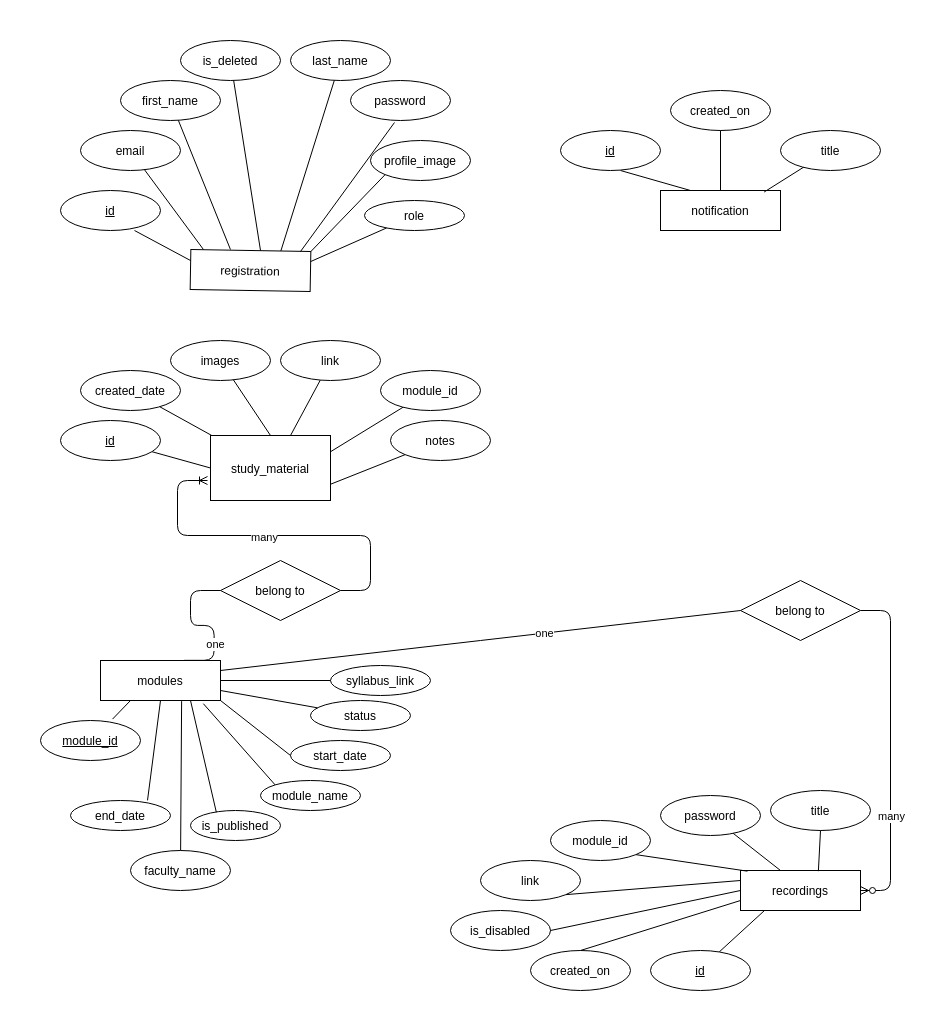
Fig-4.4 (e) Sequence Diagram for students

**4.5 Timeline Chart : -**

| Task Name | Start Date | End Date | No of Days |
| --- | --- | --- | --- |
| Analysis | 18-10-2020 | 19-11-2020 | 32 |
| Evaluation & Recommendations | 18-10-2020 | 25-10-2020 | 8 |
| Requirements Gathering | 25-10-2020 | 02-11-2020 | 8 |
| Learning the technology | 02-11-2020 | 10-11-2020 | 8 |
| Literature Survey | 10-11-2020 | 18-11-2020 | 8 |
| Design | 09-01-2021 | 13-01-2020 | 5 |
| Design Database | 09-01-2021 | 10-01-2020 | 1 |
| Software Design | 10-01-2021 | 11-01-2021 | 1 |
| Revise Project Pin | 11-01-2021 | 12-01-2021 | 1 |
| Interface Design | 12-01-2021 | 13-01-2021 | 1 |
| Software Design Document | 13-10-2021 | 14-01-2021 | 1 |
| Development | 14-01-2021 | 17-01-2021 | 4 |
| Verify & Validate User Requirements | 14-01-2021 | 15-01-2021 | 1 |
| Develop System Module | 15-01-2021 | 16-01-2021 | 1 |
| Integrate System Module | 16-01-2021 | 17-01-2021 | 1 |
| Perform Initial Testing | 17-01-2021 | 18-01-  2021 | 1 |
| Testing | 18-01-2021 | 19-01-2021 | 2 |
| Perform System Testing | 18-01-2021 | 19-01-2021 | 1 |
| Correct issues Found | 19-01-2021 | 20-01-2021 | 1 |
| Implementation & Maintenance | 20-01-2021 | 23-01-2021 | 4 |
| System Maintenance | 20-01-2021 | 21-01-2021 | 1 |
| Document Lessons Learned | 21-01-2021 | 22-01-2021 | 1 |
| Update Files/Records | 22-01-2021 | 24-01-2021 | 1 |
| Evaluation | 24-01-2021 | 25-01-2021 | 1 |

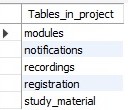
# Chapter 5 DESIGN

* 1. **Data Modeling *:***
* ***E-R Model :***

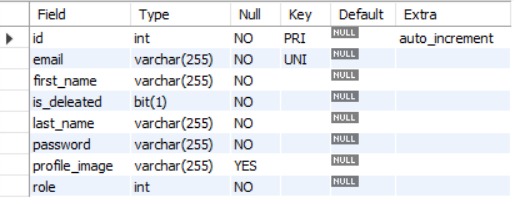


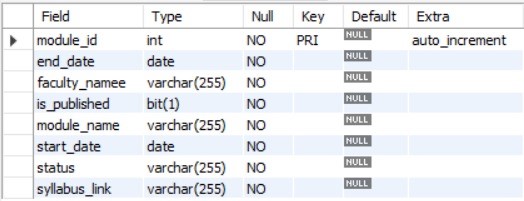
*Fig.6 E-R Diagram*

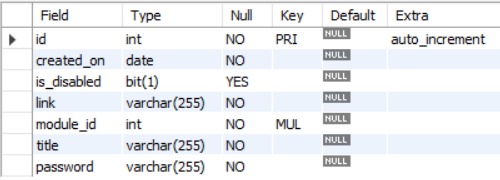
* **All Tables :**

****

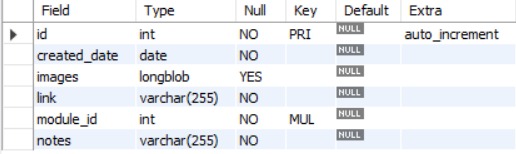
* **For Registration**

****

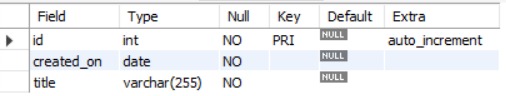
* **For Modules**
* **For Recording Links**



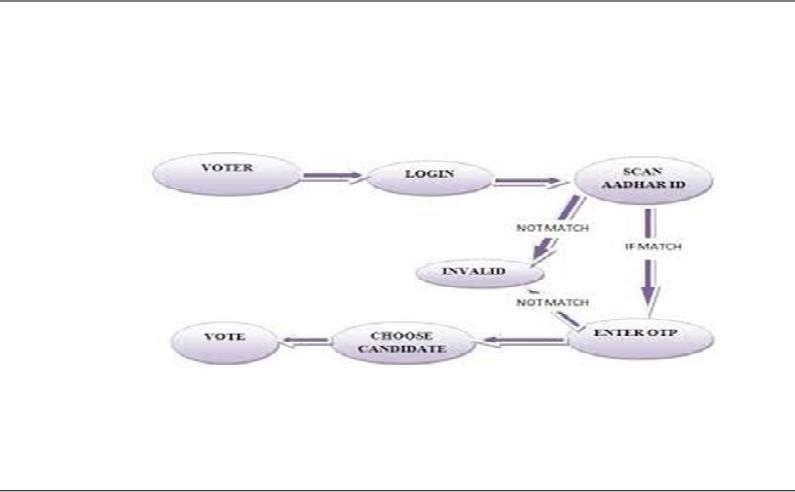
* **For Study Material**

****

* **For Notification**

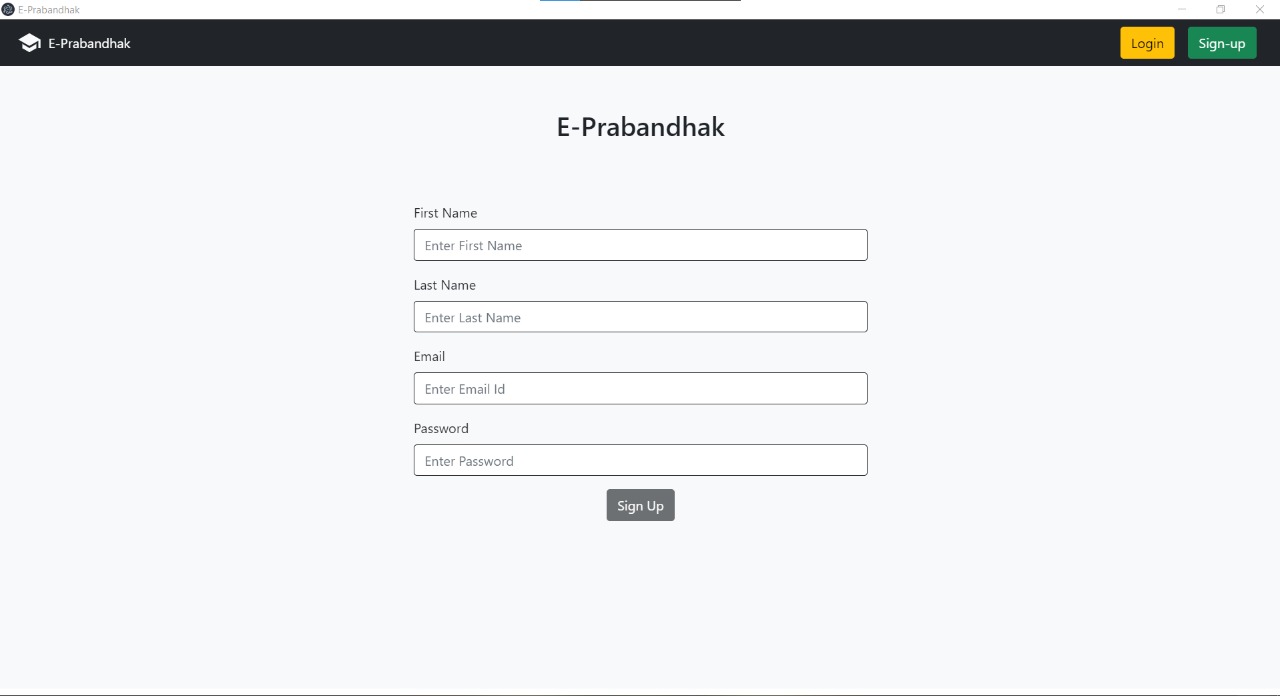
****

* 1. **Architectural Design :**

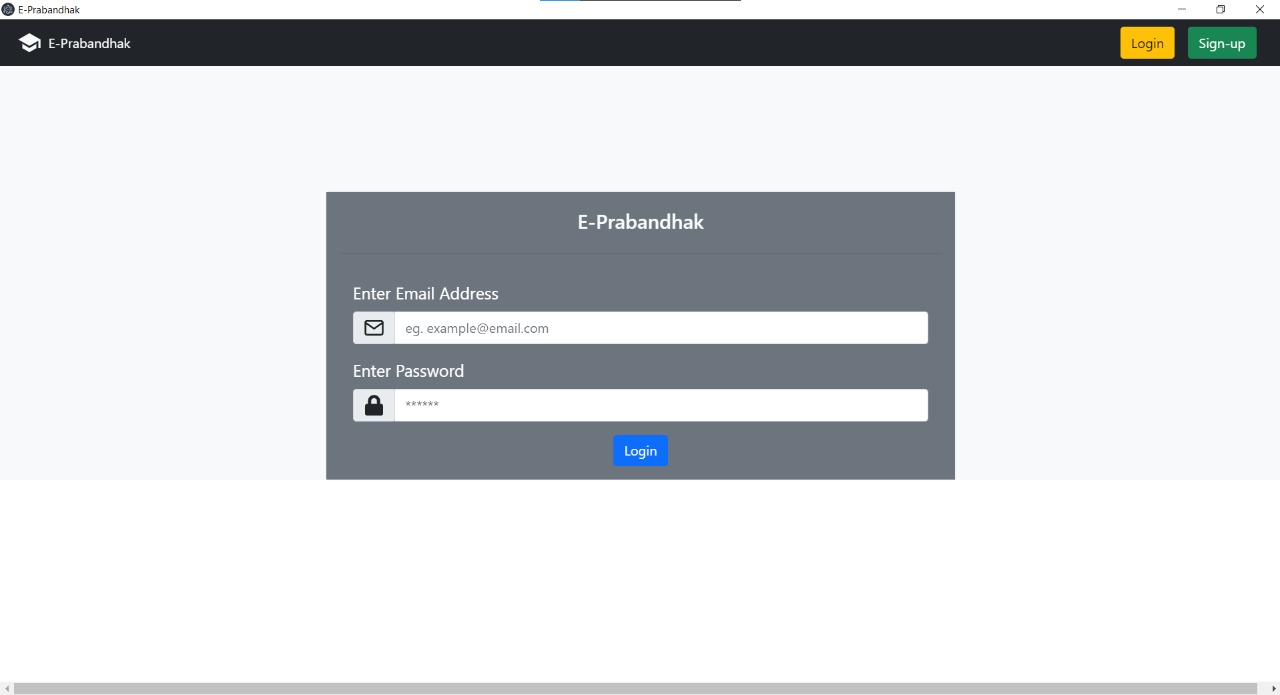


The online voting system is an electronic way of voting through a network-driven application. The voter must enter his Aadhaar number, and he will receive a one-time password message on the registered Email address stored in the database to ensure authorized access to the voting web page.

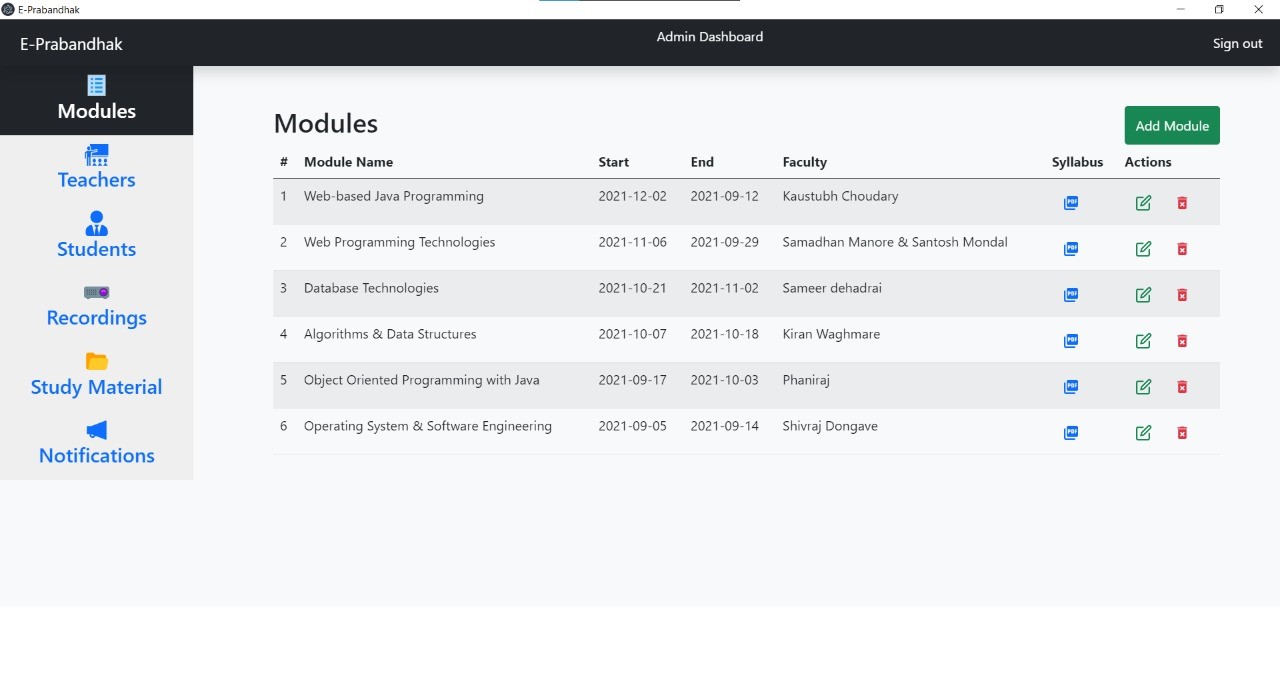
* + 1. Voter to launch the web and try to login.
    2. He/she enters the in web and enter Aadhaar number and clicks submit.
    3. Check whether the internal code Aadhaar is correct, the system sends OTP through the voter's email, and then he enters the to next process.
    4. Incorrect Aadhaar No, it gives an invalid ID message and goes to the login page.
    5. After verification voter select candidate to vote and cast vote.
  1. **User Interface Design** :
     1. **Registration Page**

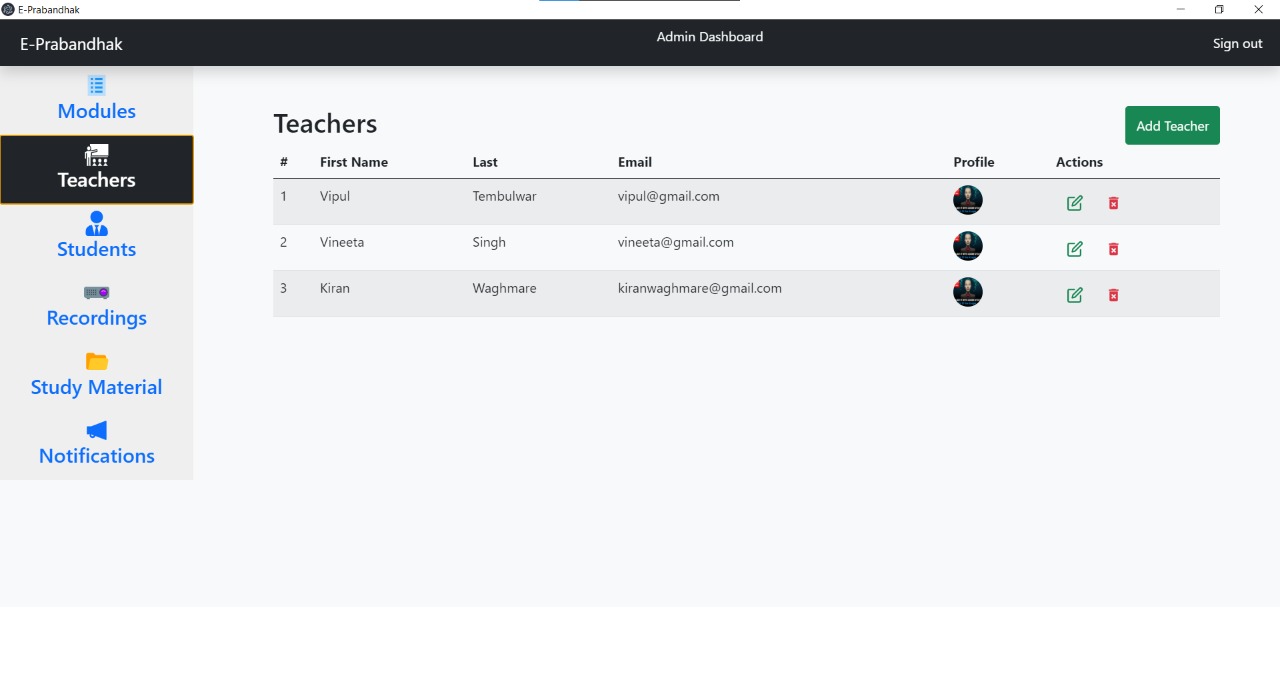


* + 1. **Login**

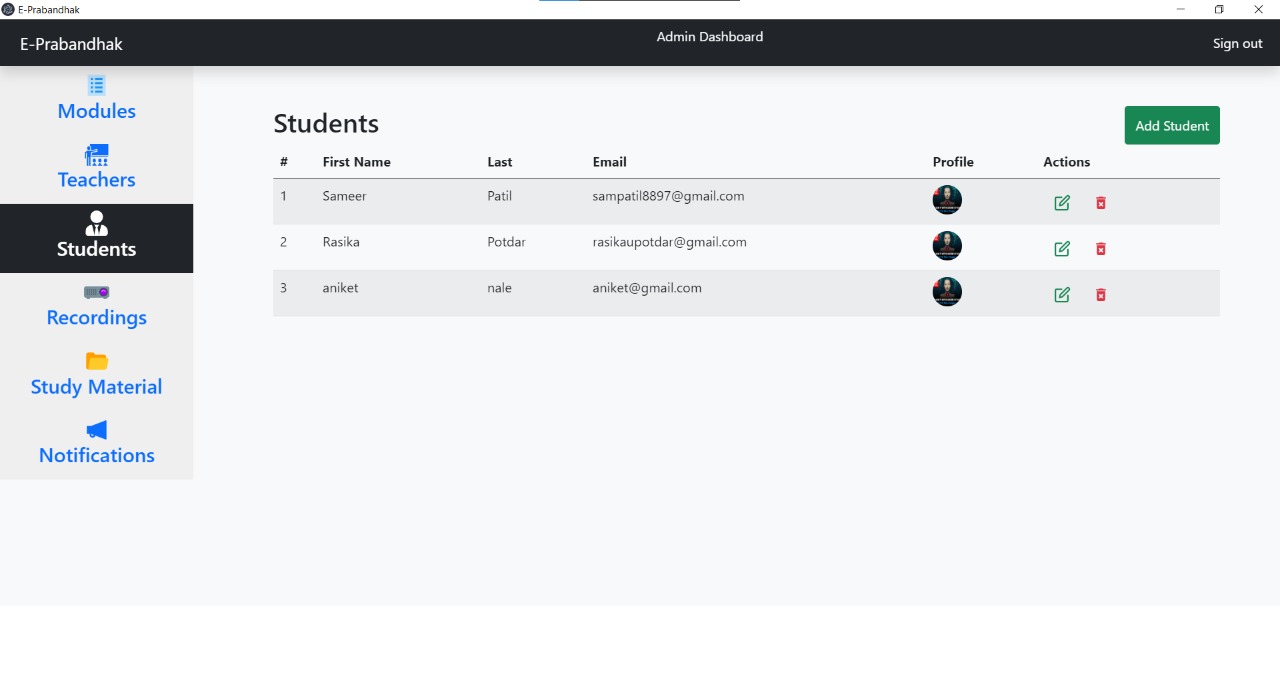


* + 1. **Admin Dashboard-1**

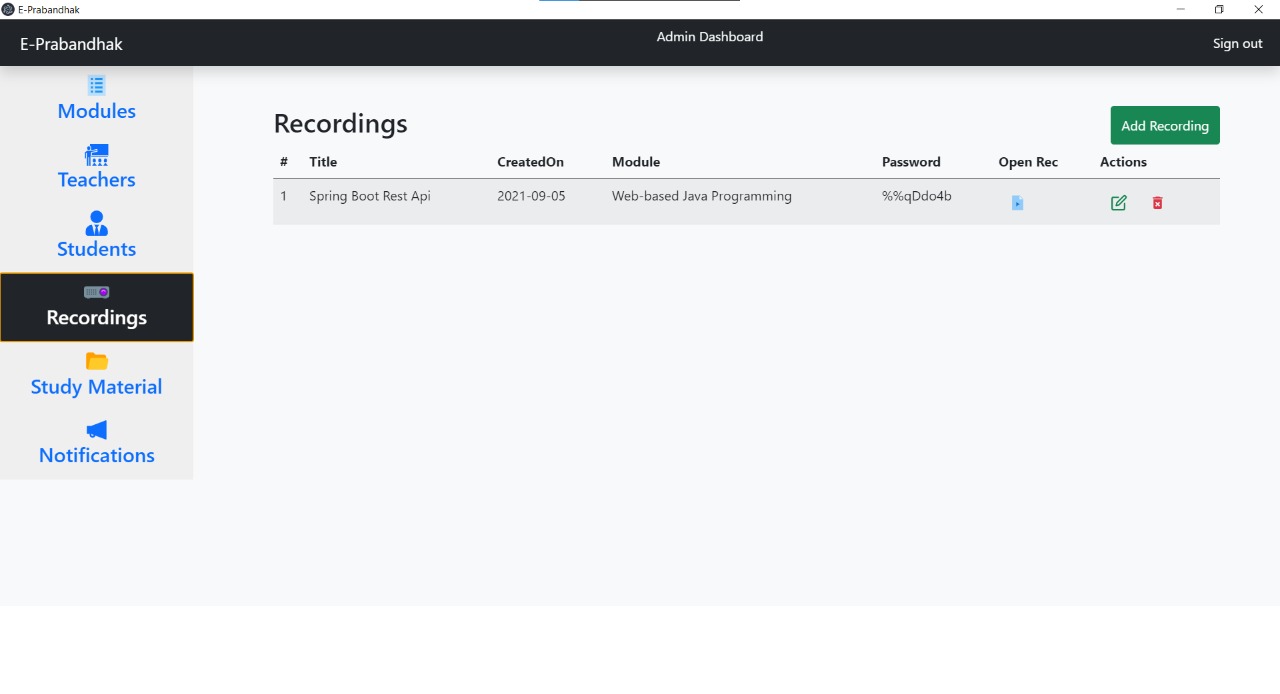
**d) Admin Dashboard-2**



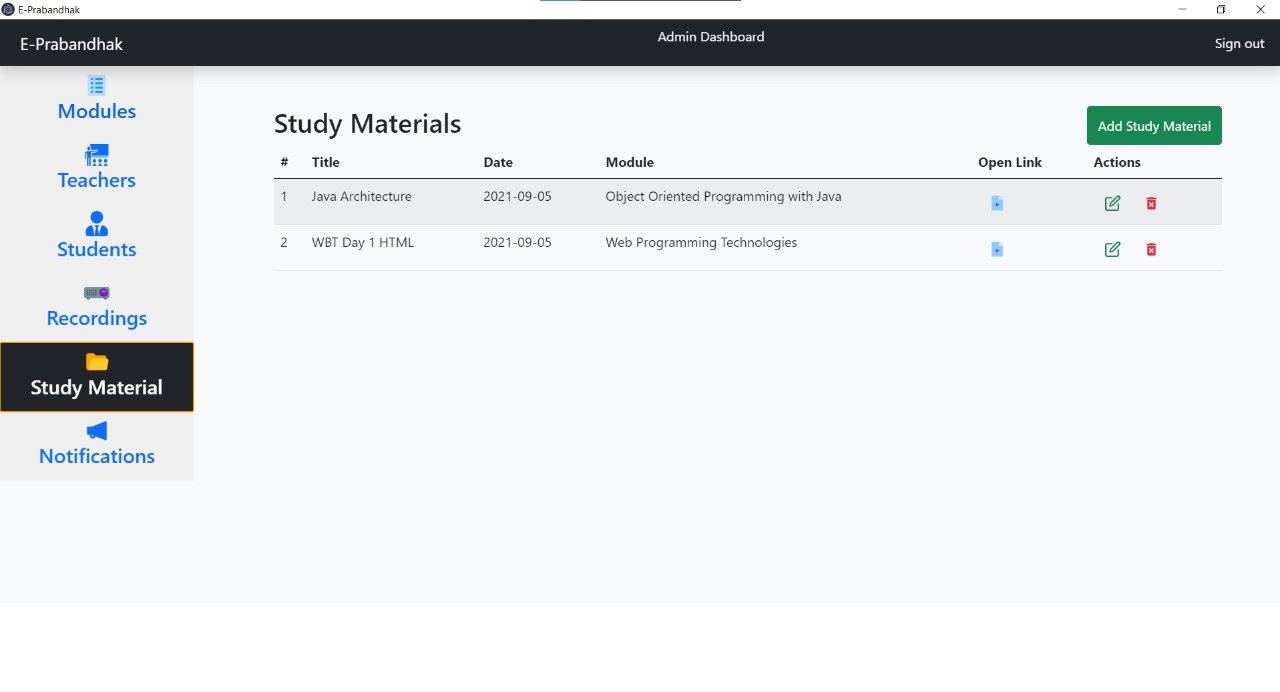
* + 1. **Admin Dashboard-3**



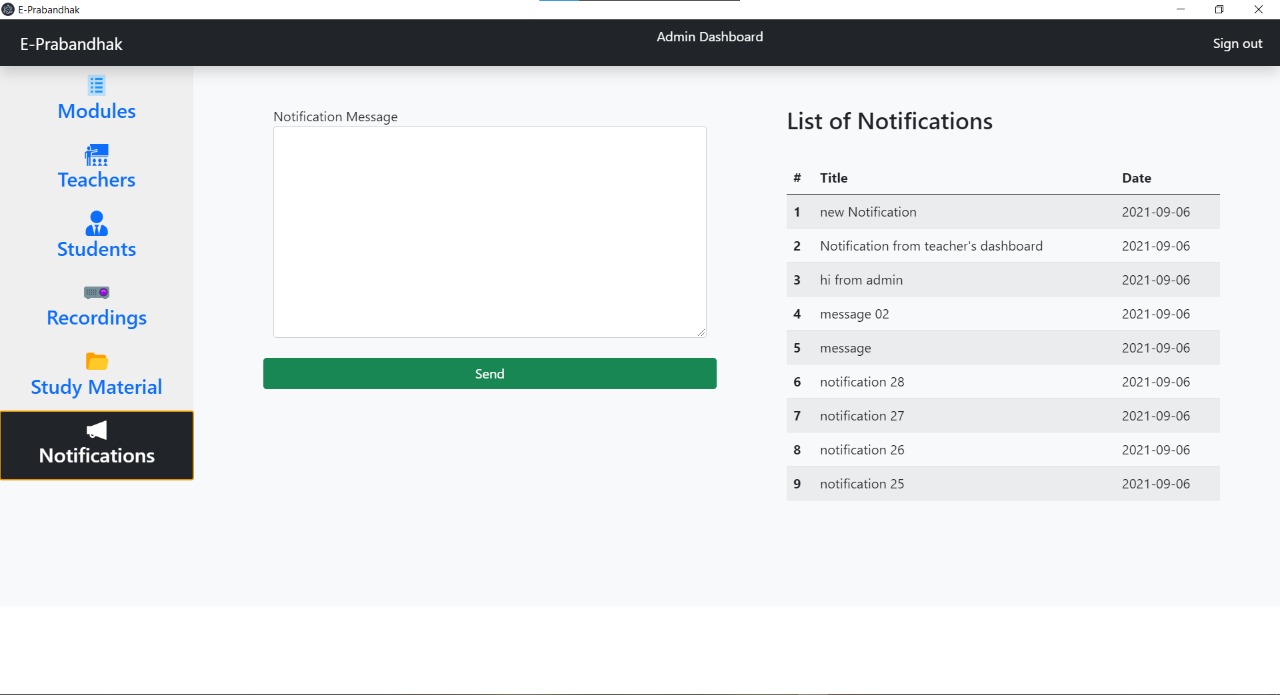
**f) Admin Dashboard-4**



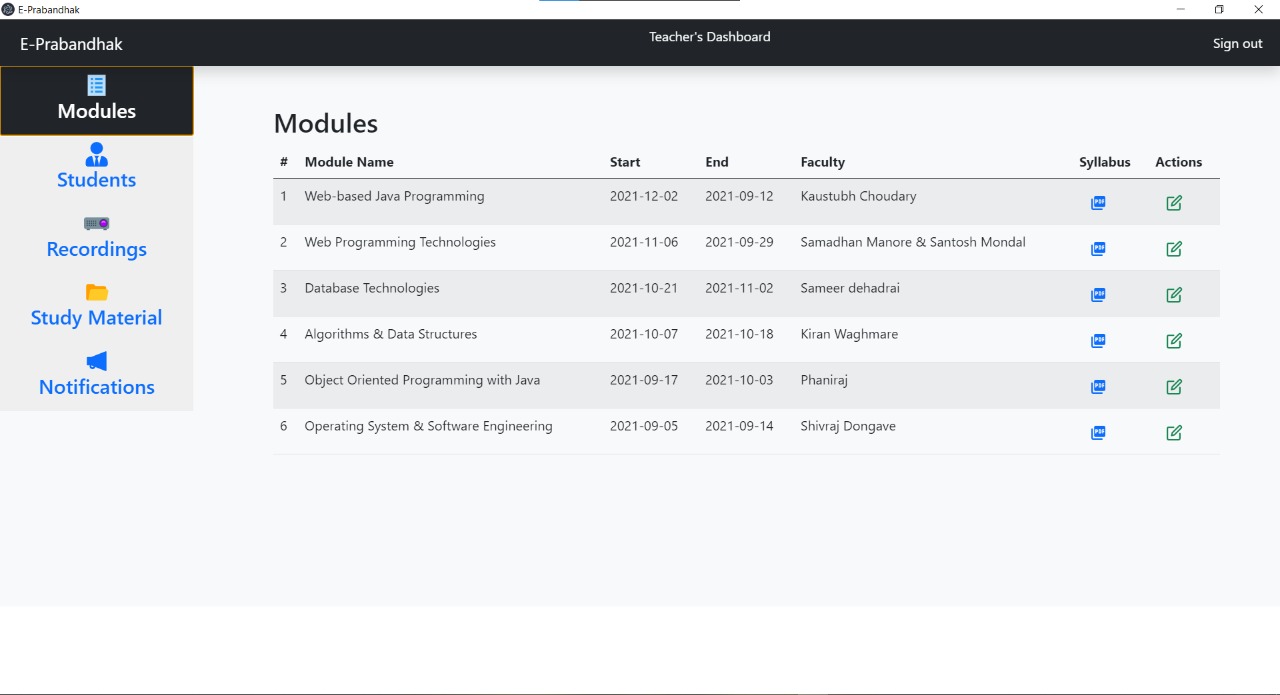
* + 1. **Admin Dashboard-5**



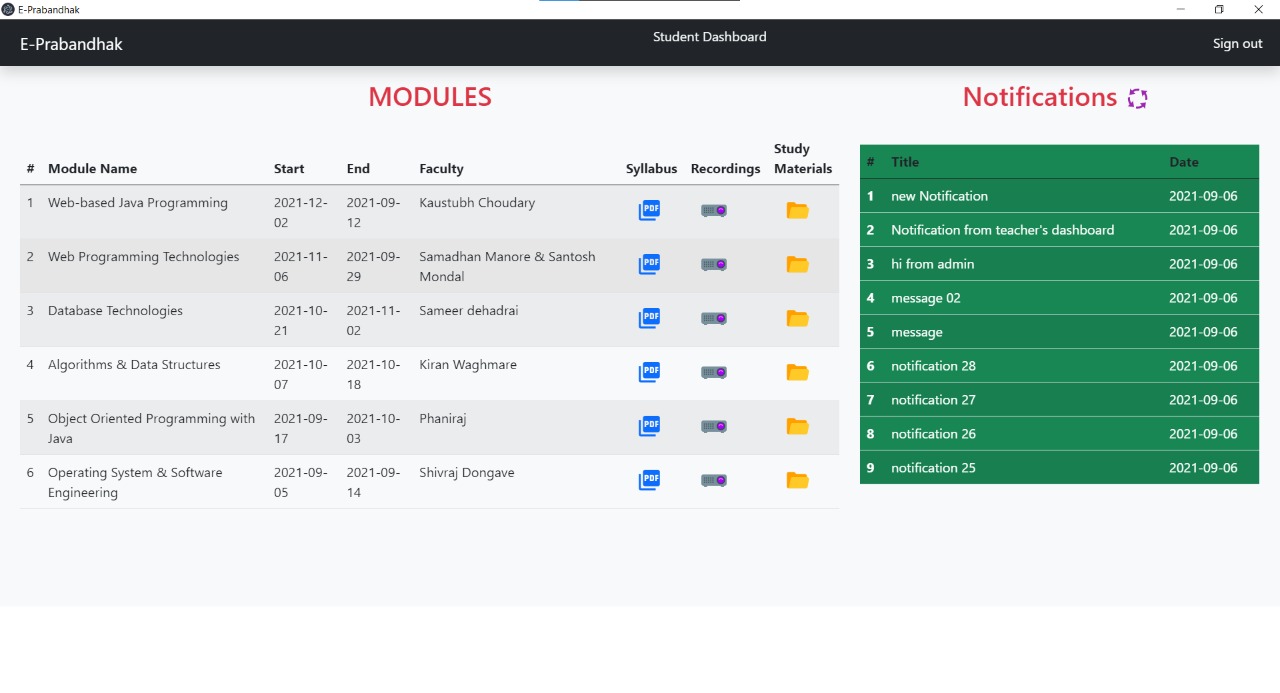
* + 1. **Admin Dashboard-6**



* + 1. **Teacher’s Dashboard**

****

* + 1. **Student’s Dashboard**



# Chapter 6 IMPLEMENTATION

* 1. **Algorithms / Methods Used**
     1. **Add vote and Count vote:-** we had stored the vote details in a database and also the vote count also stored in the database.
     2. **Notifying:-**

Notification of getting a confirmation text has been maintained by the UIN( unique identification number) and password. It can also be formulated by email.

* 1. **Working of the project *(code for mentioned algorithms)***

1. ***Crud Operations for Registration :***

package com.finalProject.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

import org.springframework.security.crypto.password.PasswordEncoder;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RestController;

import com.finalProject.dao.UserDetailsDao;

import com.finalProject.entities.UserDetails;

@CrossOrigin("http://localhost:3000")

//@CrossOrigin("\*")

@RestController

public class RegistrationController {

@Autowired

UserDetailsDao UserDetailsDao;

@Autowired

private BCryptPasswordEncoder passwordEncoder;

@PostMapping("/register")

public UserDetails createUser(@RequestBody UserDetails user) {

System.out.println(user.toString());

if (this.doesEmailExist(user.getemail())) {

user.setPassword("false");

return user;

}

String encodedPassword = passwordEncoder.encode(user.getPassword());

user.setPassword(encodedPassword);

UserDetails retUser = this.UserDetailsDao.save(user);

retUser.setPassword("true");

return retUser;

}

@PutMapping("/update")

public void updateUser(@RequestBody UserDetails user) {

// System.out.println(user.toString());

this.UserDetailsDao.save(user);

}

/\*

\* { "email":"trial@email.com", "firstName" : "aniket", "lastName":"Nale",

\* "isDeleted" : false, "password":"pass", "role":3,

\* "profileImage":"https://cdn4.buysellads.net/uu/1/3386/1525189943-38523.png" }

\*/

@PostMapping("/login")

public UserDetails authUser(@RequestBody UserDetails user) {

String email = user.getemail();

String pass = user.getPassword();

// System.out.println(user.getemail() + " " + user.getPassword());

List<UserDetails> dbUserList = UserDetailsDao.findAllByemail(email);

UserDetails dbUser = null;

if (dbUserList.size() > 0) {

dbUser = dbUserList.get(0);

boolean isValid = passwordEncoder.matches(pass, dbUser.getPassword());

if (isValid) {

dbUser.setPassword("true");

}

} else {

dbUser = new UserDetails();

dbUser.setPassword("false");

}

return dbUser;

}

@PostMapping("/doesEmailExist")

public boolean doesEmailExist(@RequestBody String email) {

List<UserDetails> dbUserList = UserDetailsDao.findAllByemail(email);

return (dbUserList.size() > 0) ? true : false;

}

@DeleteMapping("/user/delete/{id}")

public void deleteUser(@PathVariable String id)

{

System.out.println("Delete Module Request: "+ id);

this.UserDetailsDao.deleteById(Integer.parseInt(id));

}}

1. ***Crud Operations for User :***

package com.finalProject.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RestController;

import com.finalProject.Services.UserServiceImpl;

import com.finalProject.dao.UserDetailsDao;

import com.finalProject.entities.UserDetails;

@CrossOrigin("http://localhost:3000")

@RestController

public class UserController {

@Autowired

UserServiceImpl userServiceImpl;

// get list of all users + teachers + Admin

@GetMapping("/allUsersList")

public List<UserDetails> getAllUserList() {

return this.userServiceImpl.getAllUserList();

}

@GetMapping("/allRoleBasedList/{id}")

public List<UserDetails> getAllTeachersList(@PathVariable int id) {

return this.userServiceImpl.findAllByrole(id);

}

@DeleteMapping("/api/user/delete/{id}")

public void deleteUserById(@PathVariable int id) {

this.userServiceImpl.deleteUserById(id);

}

}

package com.finalProject.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestParam;

import org.springframework.web.bind.annotation.RestController;

import com.finalProject.Services.Study\_MaterialImpl;

import com.finalProject.dao.RecordingsDao;

import com.finalProject.dao.Study\_MaterialsDao;

import com.finalProject.entities.Recordings;

import com.finalProject.entities.Study\_Materials;

import com.finalProject.entities.UserDetails;

//@CrossOrigin("http://localhost:3000")

@RestController

public class Study\_MaterialsController {

@Autowired

Study\_MaterialImpl study\_MaterialImpl;

// get list of recordings + teachers + Admin

@GetMapping("/allStudy\_Materials")

public List<Study\_Materials> getAllStudy\_MaterialsList() {

return this.study\_MaterialImpl.getAllStudy\_MaterialsList();

}

@PostMapping("/addStudy\_Materials")

public void createUser(@RequestBody Study\_Materials study\_Materials) {

this.study\_MaterialImpl.addStudy\_Materials(study\_Materials);//.study\_MaterialsDao.save(study\_Materials);

}

@DeleteMapping("/deleteStudyMaterials/{id}")

public void deleteStudyMaterials(@PathVariable int id) {

this.study\_MaterialImpl.deleteStudy\_Materials(id);//study\_MaterialsDao.deleteById(id);

}

@PutMapping("/updateStudyMaterials")

public void updateUser(@RequestBody Study\_Materials study\_Materials) {

this.study\_MaterialImpl.updateStudy\_Materials(study\_Materials);//study\_MaterialsDao.save(study\_Materials);

}

}

package com.finalProject.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RestController;

import com.finalProject.Services.RecordingImpl;

import com.finalProject.dao.RecordingsDao;

import com.finalProject.dao.UserDetailsDao;

import com.finalProject.entities.Recordings;

import com.finalProject.entities.Study\_Materials;

import com.finalProject.entities.UserDetails;

@RestController

public class RecordingsController {

@Autowired

RecordingImpl recordingImpl;

// get list of recordings + teachers + Admin

@GetMapping("/allRecordings")

public List<Recordings> getAllRecordingsList() {

return this.recordingImpl.getAllRecordingsList();

}

@GetMapping("/getAllRecordingsById/{id}")

public List<Recordings> getAllREcordingsById(@PathVariable int id) {

return this.recordingImpl.getAllREcordingsById(id);

}

@PostMapping("/addRecordings")

public Recordings createUser(@RequestBody Recordings recordings) {

return this.recordingImpl.addRecording(recordings);

}

@DeleteMapping("/deleteRecordings/{id}")

public void deleteRecordings(@PathVariable int id) {

this.recordingImpl.deleteRecordings(id);

}

@PutMapping("/updateRecordings")

public Recordings updateUser(@RequestBody Recordings recordings) {

return this.recordingImpl.updateRecording(recordings);

}

}

# Chapter 7

# TESTING

* 1. **Test Cases**

| **Test Id** | **Item to be Tested** | **Steps** | **Input** | **Actual Output** | **Expected Output** | **Pass/Fail** |
| --- | --- | --- | --- | --- | --- | --- |
| **1** | **Content Protection** | **Tried different Softwares to to if the screen is recording or not** | **E-Prabandhak Application** | **Screen wasn't getting captured** | **Screen must not be captured.** | **Pass** |
| **2** | **System check for proper username and password entered by users** | **System compares the data entered by user and the entered data in database** |  |  |  |  |
|  |  | **If username and password is valid** |  | **Make Connection** | **Make connection** | **Pass** |

|  |  | **If username and password is invalid** |  | **Report invalid Voter id** | **Report error** | **Pass** |
| --- | --- | --- | --- | --- | --- | --- |
| **3** | **System checks whether details of user are entered as per the format** | **System checks if the data entered by the user is in valid form or not.** |  |  |  |  |
|  |  | **If valid** | **user entered data** | **Entered in database** | **Entered in database** | **Pass** |
|  |  | **If invalid** | **user entered data** | **“Invalid Data” message will be printed** | **“Invalid Data” message will be printed** | **Pass** |

* 1. **Type of Testing used**

#### Manual Testing :

In the manual test, we used the machine coding tool to write all the codes. After writing each code, we run the code in the local server of the machine, and then check whether it runs. We have made the API and manually tested it in the postman tool and checked whether all the APIs are working well. It works well, so we add this code to your project file and connect to the front-end code.

#### Unit Testing :

In the unit test, we have tested the back-end code on the local server of the computer, and the code can run, and then we checked whether all APIs are working properly. Here, we connect all the back- end code to the front-end code, and check whether the login, registration, add/delete, and update functions are operating normally after clicking the button.

#### API Testing :

In this test, we checked whether all APIS of students,teachers, and administrators are running well, or what type of exception is given, and we have resolved all exceptions.

# Chapter 8 Results and Discussions

* The proposed system will aim to provide a platform where the Institute can manage the batches online.
* Easy student interface: Students can manage all their activities on a single platform
* SCOPE FOR FUTURE ENHANCEMENT

The E-Prabandhak platform can be made versatile by adding following functionalities

* Performance tracker

Attendance Keeper

Performance Tracker is **an ongoing analysis of the students' performance in exams**.

Capturing students’ **attendance** data in real-time will make E-Prabandhak more versatile.

# Chapter 9 Conclusions

In this paper, we have presented the design of an online student management system. The key issue addressed by our work has been the issue of students not being able to find the recordings of lectures,not being able to find appropriate study materials and not being up to date with the notices passed by the institute.Students from all age groups are now studying online and all the organisations need a stable and good UI for students. Also, our design successfully meets all the previously identified requirements of convenience, transparency, flexibility, support for students.Institutes can also be happy as their content can not be recorded and published elsewhere and thus gives security and privacy.

# Appendix

**Glossary Authorized Voter:**

a voter who is on the enumeration list and who has been given the means to use the Pericles Voting Client software.

**Ballot:**

a means of registering a vote. In the case of the Pericles system, this is a set of information and questions that can be transmitted from a Pericles Elections Server to a Pericles Voting Client and back.

**Client:**

a program that connects to a server. In the case of the Pericles system, the Voting Client is a piece of software that Voters can use to connect to a Pericles Election Server and receive ballots, send their votes and possibly change their votes if this is permitted in an election.

**Code Reviews:**

meetings where software developers review each other’s code as a quality control mechanism.

**Customers:**

peoples or agencies who purchase the Pericles system.

**Database:**

any aggregation data. Files consisting of records (or tables), each of which is constructed of fields (or columns) of a particular type, together with a collection of operations.

**Denial of Service:**

explicit attempts by attackers to prevent legitimate users of a service from using that service. Examples include:

* attempts to "flood" a network, thereby preventing legitimate network traffic
* attempts to disrupt connections between two machines, thereby preventing access to a service
* attempts to disrupt service to a specific system or person

**Development Team:**

a team of people developing software.

**Distributed Network:**

a network in which processing, storage and other functions are handled by separate units rather than by a single main computer

**Elections Database:**

database system used for storing election questions, Elections Officer information, voter information and votes.

**Election Editor:**

Graphical User Interface (GUI) application that allows the Election Officer register an election, change its settings and customize ballots Election Officer: an impartial individual authorized to run an election.

**Elections Server:**

a server that manages an election and that the Pericles Voting Client can Software Requirements Specification for E-Voting System xi | Page connect to. It is responsible for jobs such as database management, user identity checks and reporting of statistics.

**Encryption:**

the process of making information indecipherable to unauthorized readers. This can be used for the storage and transmission of information.

**Enumeration List:**

a list of identification information and other required information for all authorized voters.

**Enumerated Voter:**

a voter that is registered for a specific election.

**GUI:**

a graphics–based user interface that incorporates icons, buttons, pull-down menus and a mouse.

**Hacker:**

a person who secretly invades other people’s computers to inspect or tamper with the programs or data stored on them.

**HTML:**

an acronym for Hypertext Markup Language. The document format used on the World Wide Web. Web pages are

**Literature Cited**

###### IEEE standard Book,

1. T. Kohno, A. Stubblefield, A. D. Rubin and D. S. Wallach, “Analysis of a Student Portal”.

###### Journal Paper,

1. System”, IEEE Symposium on Security and Privacy 2004. IEEE Computer Society Press, May 2004

###### Proceeding paper,

1. Writing Effective Use Cases - Alistair Cockburn (1998)

Thomos M.Buchsbaum, “E-Voting: International developments and lesson learnt”, Technical Report by Australian Federal Ministry for Foreign Affairs, 2004

### Publications

<https://pushy.me/>

<https://dev.to/mandiwise/electron-apps-made-easy-with-create-react-app-and-electron-forge-560e>

https://reactjs.org/

### Acknowledgements

We take this opportunity to express our gratitude to those who directly or indirectly helped us complete the project. First of all, we would like to express our deep gratitude to our guiding Gentleman **Chaitanya Mhatre** for his valuable guidance and encouragement throughout the project. We would also like to express our gratitude to **Mr. Sohan More** of the Project Academy, the course coordinator of CDACJuhu/Kharghar in Mumbai. Finally, I want to thank my parents for their trust in us. I also want to thank all of our blessings. We would also like to thank all the other staff at CDAC Juhu/Kharghar in Mumbai.

**Aniket Nale - 210540381016 KH**

**Jacob Patole -210540581049 JH**

**Rajat Yerigeri -210540381085 KH**

**Rasika Potdar -210540381088 KH**

**Sameer Patil -210540581088 JH**

**Mahesh Kanojiya -210540581058 JH**