Alumni Information Management System

Submitted in partial fulfillment of the requirements of

PG Diploma in Advanced Computing

By

1. Shreyash Amberkhane	**- 210540581097**
2. Sanket Shinde	**- 210540381098**
3. Alisha Nagvekar	**- 210540381012**
4. Onkar Vangutte	**- 210540581068**
5. Sonali Chavan	**- 210540581103**

Guides:

Ms. Anuja Shirsath
Supervisor/Guide

Mr. Sohan More
Faculty Supervisor/Guide

Centre for Development of Advanced Computing

Mumbai

May 2021

CERTIFICATE

This is to certify that the project entitled **"Alumni Information Management System"** is a bonafide work of **Shreyash Amberkhane (PL) (210540581097), Sanket Shinde (210540381098), Alisha Nagvekar (210540381012), Onkar Vangutte (210540581068), Sonali Chavan (210540581103),** submitted to C-DAC Mumbai in partial fulfillment of the requirement for the award of the Post Graduate Diploma in Advanced Computing.

Ms. Anuja Shirsath	**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. Shreyash Amberkhane – 210540581097

Mr. Sanket Shinde–210540381098

Ms. Alisha Nagvekar – 210540381012

Mr. Onkar Vangutte – 210540581068

Ms. Sonali Chavan – 210540581103

Date: 06.09.2021

Abstract

The main aim of the project is to build an interaction between alumni, admin and the students; a system that will be able to manage alumni data of a college and provide easy access to the same. The alumni will also be interested to maintain relations with their institutions. The alumni and the student can communicate only through the admin permission. A system that will be able to manage alumni data of a college and provide easy access to the system. Access to the system can help them in building connections to their projects or for placements. The system will automatically list all Alumni information (name, passing year, company currently working in, contact number) and their status will be transferred from the student module to the alumni module.

Contents

List of Figures

List of Tables

List of Abbreviations

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

Chapter 1

Introduction

The greatest asset any institution can have is the Alumni system. Alumni are the people who represent the institution in the real world. Alumni website is created for the students that have graduated from the institution. This is an online website that allows former students to take advantage of the benefits and services that an institution offers after graduation.

The alumni network is becoming important in the development of the institution because of their vast potential that benefits both the institution and the students. There are many benefits for being an alumni member of a college or institution . Some of these benefits are : keeping a person informed on the events that are organized by the institution, and when some important events are going to be held in the institution. Another benefit is that the information concerning a former student can easily be received and other members of the alumni community can be located without much stress.

1. Description

Overall description consists of background of the entire specific requirement. It also gives explanation about actor and function which is used. It gives explanation about architecture diagram and it also gives what we are assumed and dependencies. It also support specific requirement and also it support functional requirement, supplementary requirement other than actor which is used. It also gives index and appendices. It also gives explanation about any doubt and queries.

Once a student graduates from the institute, his/her professional life or career begins, with higher education playing an important role in establishing himself/herself in the profession. In respect of college, it has been our experience that from the very beginning, the alumni have maintained personal contacts with one another, rather than use the channel of Alumni Association.

The advancements in information technology have certainly helped in creating new resources such as alumni web pages, list servers etc., so as to permit greater interactions between the alumni.

2. Problem Statement

Nowadays, the education became one of the desired need of the people, but unfortunately what if the student is sincere but unaware about the current market status and the market requirement.

This is the major problem with the students that they are unable to get more and more information about the surrounding "What is going on..?" The Problem occurred with the students at the time of career guidance, Project title selection, Internships, Domain related queries, any query related to the syllabus or something which is very new concept for them.

Because of the lack of knowledge the students decided on their own ways may be wrong or right. If luckily they get the desired goal then no issue but imagine what if they have selected something very wrong. After the wrong choice the thing which has been done is only the wastage of time. Always remember "Your golden advice can save others precious times."

The proposed architecture offers a solution to those problems. Problem can be solved by reducing the distance between the students with their alumni and the faculties so that they can get the correct guidance or the advice and can utilise their time in best way.

The implemented Alumni Portal will provide the features for all the Alumni and the Students.

3. Motivation

The motivation to develop this project is to solve some problems that are currently occurring in every colleges/university institutions. Many challenges exist for building the desired alumni systems. Most people are habituated to social networking sites and spending time on these sites has become an indispensable activity in the daily routine. As such, it may be quite a difficult task to make alumni and current students to get habituated to SAS. A factor which could attract more members is the current member database. If more users are currently using the system, it is more likely that it would attract new users. Further, if the current students are allowed to be a part of the SAS while enrolled at the university, the university need not put any additional effort in attracting them or getting them involved with alumni activities once they graduate. A simple change of status from current students to alumni can be performed which correspondingly changes the privileges of the particular member.

Several issues need to be considered while expanding the alumni system to include current students. The principal logistics to be considered are: the hardware to support larger websites; simultaneous handling of privacy and alumni concerns. It should be kept in mind that not all alumni might be interested in sharing their details with people whom they don‘t know. The increase in data volumes resulting from larger website system also adds to questions about the privacy of the data. This can be taken care of by specifying selective permissions to members in the system. Based on the type of user the access can be defined. One key aspect in protecting privacy can be accomplished by requiring all the members in the system to have a valid email id, provided by the university. The guest users would have an exception to this rule, and will be allotted restricted access. Another rule which can be incorporated is the validation of information provided by a user during registration to check for consistency. This information typically involves details such as student number, department, year of graduation etc. and a combination of these values are unique. Hence, only authenticated members can get access to the system.

Although our website is quite attractive but it lacks the feature of communication with the alumnus. This creates a problem for the current students in terms of guidance, internships, industrial knowledge. It also lacks the instant solution to the problem which are related to education and career. It does not provide any idea about the news and current status of jobs in different companies and industries.

4. Scope For Future Enhancement

It is not possible to develop a system that makes all the requirements of the user. User requirements keep changing as the system is being used. Some of the future enhancements that can be done to this system are:

- As the technology emerges, it is possible to upgrade the system and can be adaptable to desired environment.
- Because it is based on object-oriented design, any further changes can be easily adaptable.
- Based on the future security issues, security can be improved using emerging technologies.
- Attendance module can be added
- sub admin module can be added
- Students also can see the campus interview details.

Chapter 2

Literature Review

The Existing system is a computerized system but which is maintained at individual databases i.e in excels sheets, it’s a time delay process. And maintaining all the records in Excel sheets is difficult. If they want any record they have to search all the records. It doesn’t provide multiple user accessibility and also doesn’t have different user privileges. So the system is not accessible for all the employees of the organization.

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.

The Proposed system is a computerized system but which is maintained at Centralized databases i.e. in automated forms it’s a very fast process. And maintaining all the records in online systems database which makes it very easy to access and retrieve data from the database. If they want any record they can easily search all the records. It provides multiple user accessibility and also has different user privileges. So the system is accessible for all the employees of the organization.

Chapter 3

3.1 System Analysis

Functional Requirements

Once the user credentials are entered, all data will be accepted and processed by controller (Servlet) layer. Then this data is getting stored into database. The page now redirected to login page for login. Now user has to select his/her status (Alumni/Student). The data entered into login page accepted and processed by controller, compares with the data which already stored in database. Once the data matched user can able to enter into respective home page. Now user has functionalities to see ongoing/future events, able to see their data in profile section and free to leave by logout functionality. Once logout from website user needs to enter his/her credentials again to enter into website.

3.1.1 Login of Admin

o The system will allow the administrator to select, add, delete, update students and alumni's.
o The system will allow the administrator to select " students, alumni's " and check their details.
o The system will allow administrators to select, add, delete, update events.
o The system will allow the administrator to view the list of students and alumni's.

3.1.2 Login of Student

o The system will allow the students to enter the Name, Email, Phone no, Password, PRN No, Branch etc.
o The system will allow the students see current and future events.

3.2 Non-functional Requirements

3.2.1 Performance Requirements

The system should store all the database records of each students, alumni's and administrator, and the application should be available through the server (local host). 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 location.

3.2.2 Safety Requirements

1. In order to prevent data loss in case of system failure, the data of students, alumni's will be safely saved in the database.

2. In case the admin detects any security issues in the system, he should able to shut down the user data and close all connections of their immediately.

3. The system should be capable of gracefully recovering from earlier crashes and continuing the process safely and inefficient way.

3.2.3 Security Requirements

Passwords of the Admin, students and alumni 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 intruder cannot know the password of user. All passwords should be stored as a secure hash of the administrator password.

1. **Communications Security** All communications between the admin and alumni's must be encrypted to ensure the privacy of data. Encryption of communications will also ensure that anyone packet sniffing over the network will be unable to extract any usable information from the data that is sent between the Client and Server.

2. Storage Security All students, alumni's must be able to record their data anonymously without anybody being able to determine how they stored or change their data's. If Admin is going to be able to change their data's, the system must store their identification information along with their data's. This means that the file storing their data's must be encrypted so that nobody can read it directly at any stage.

3.2.4 Software Quality Attributes

3.2.4.1. Mobility:

The admin can able to add, delete, and update user details, event details. Alumni can see and can able to contact with the institutes and vice versa.

3.2.4.2. Transparency:

User can easily register and log into website and get daily updates of activities, events.

3.2.4.3. Accuracy:

The system shall manage, create, update and delete all records correctly.

3.2.4.4. Eligibility:

Only authorized students, Alumni's who are registered with institute, should be able to login.

3.2.4.5. No Repeat-Entry:

The User shall be prevented from choosing more than one registration with the same credentials.

3.2.4.6. No Null-Entry:

The User may receive a warning of not entering null information into credentials fields or not to leave blank credentials.

Chapter 4

Analysis Modeling

4.1 Use Case Diagram:

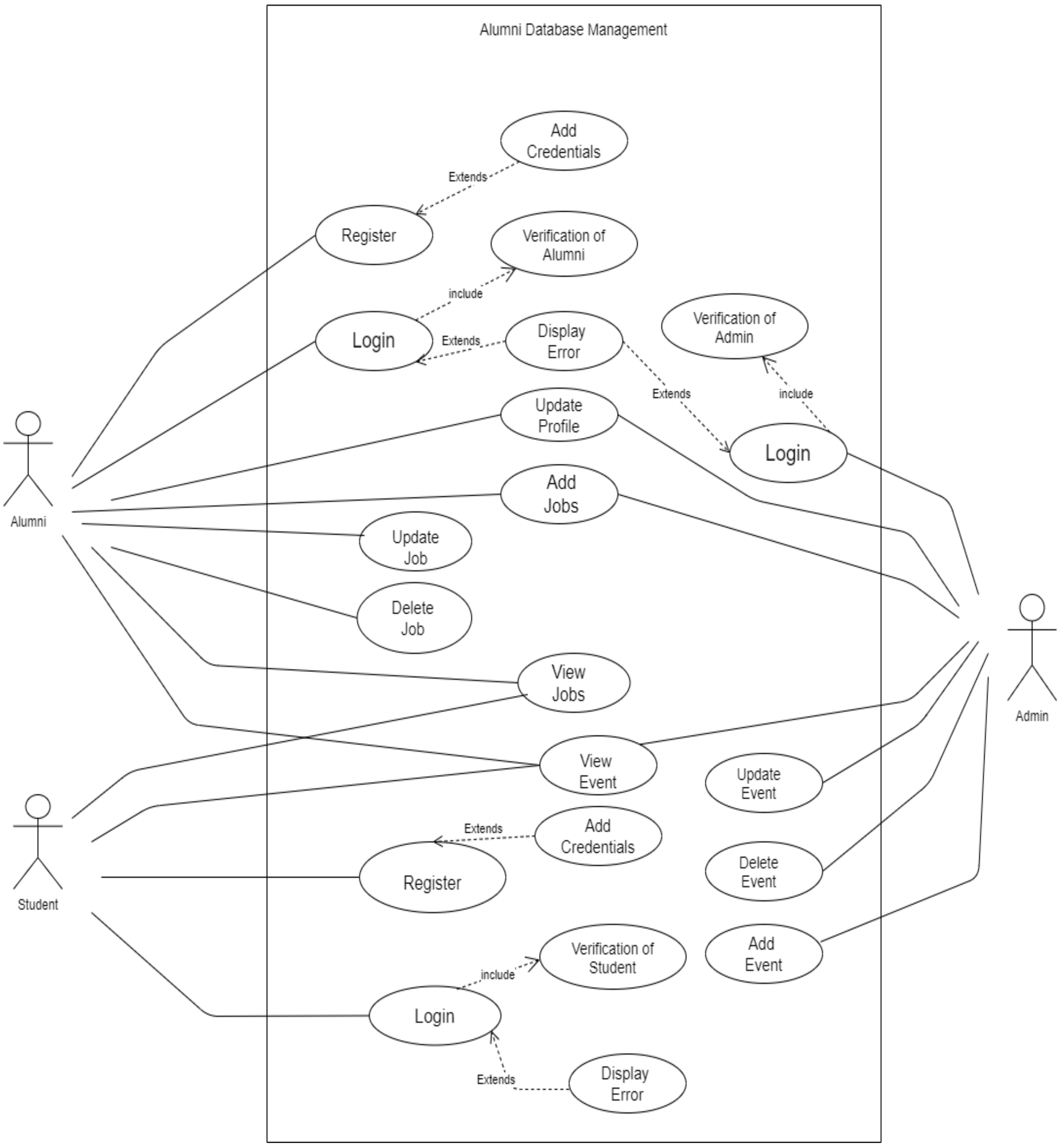


Fig. 4.1: Use Case Diagram

4.2 Class Diagram:

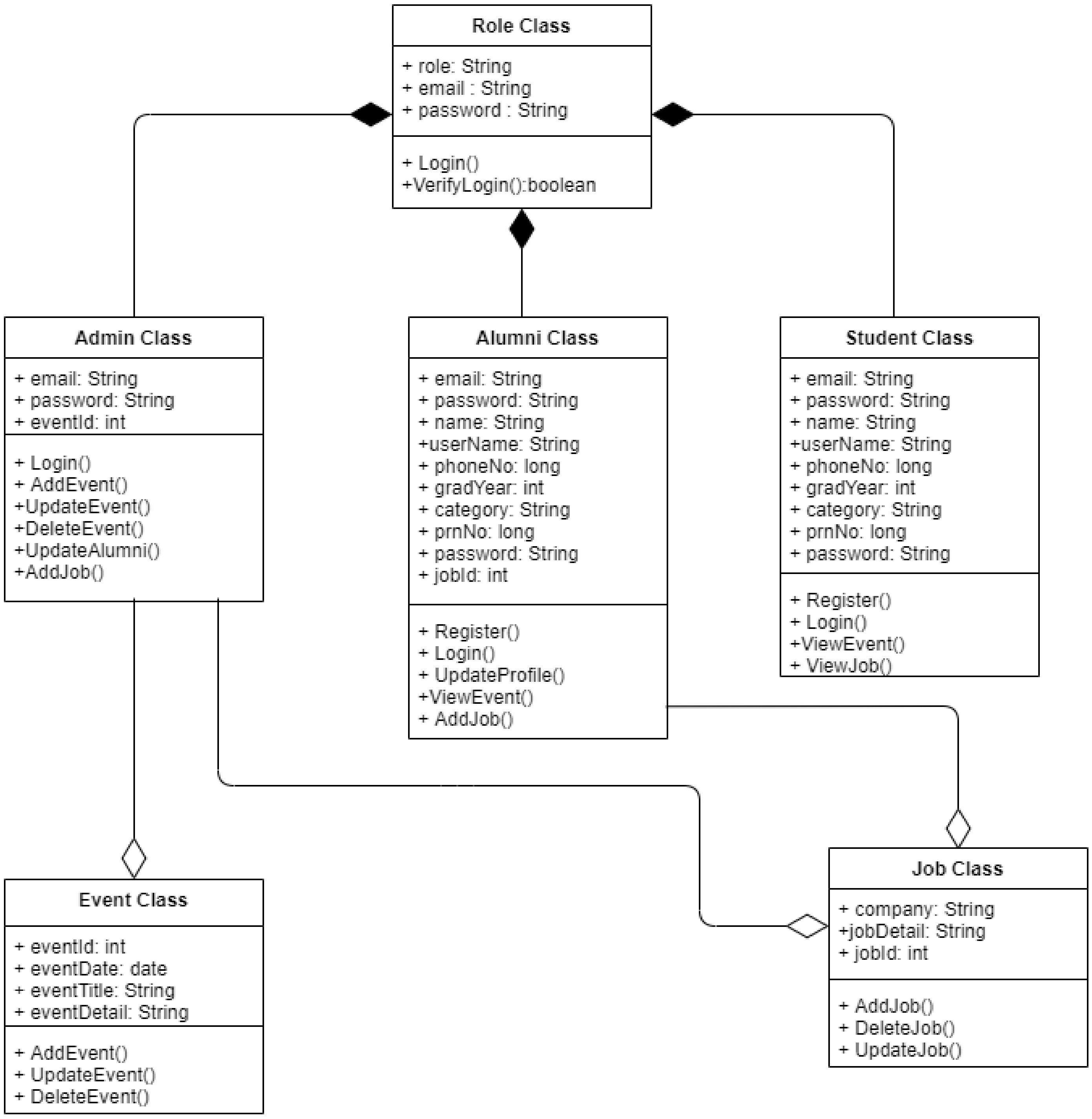


Fig. 4.2: Class Diagram

4.3 Activity Diagrams:

4.3.1 Admin Activity Diagram:

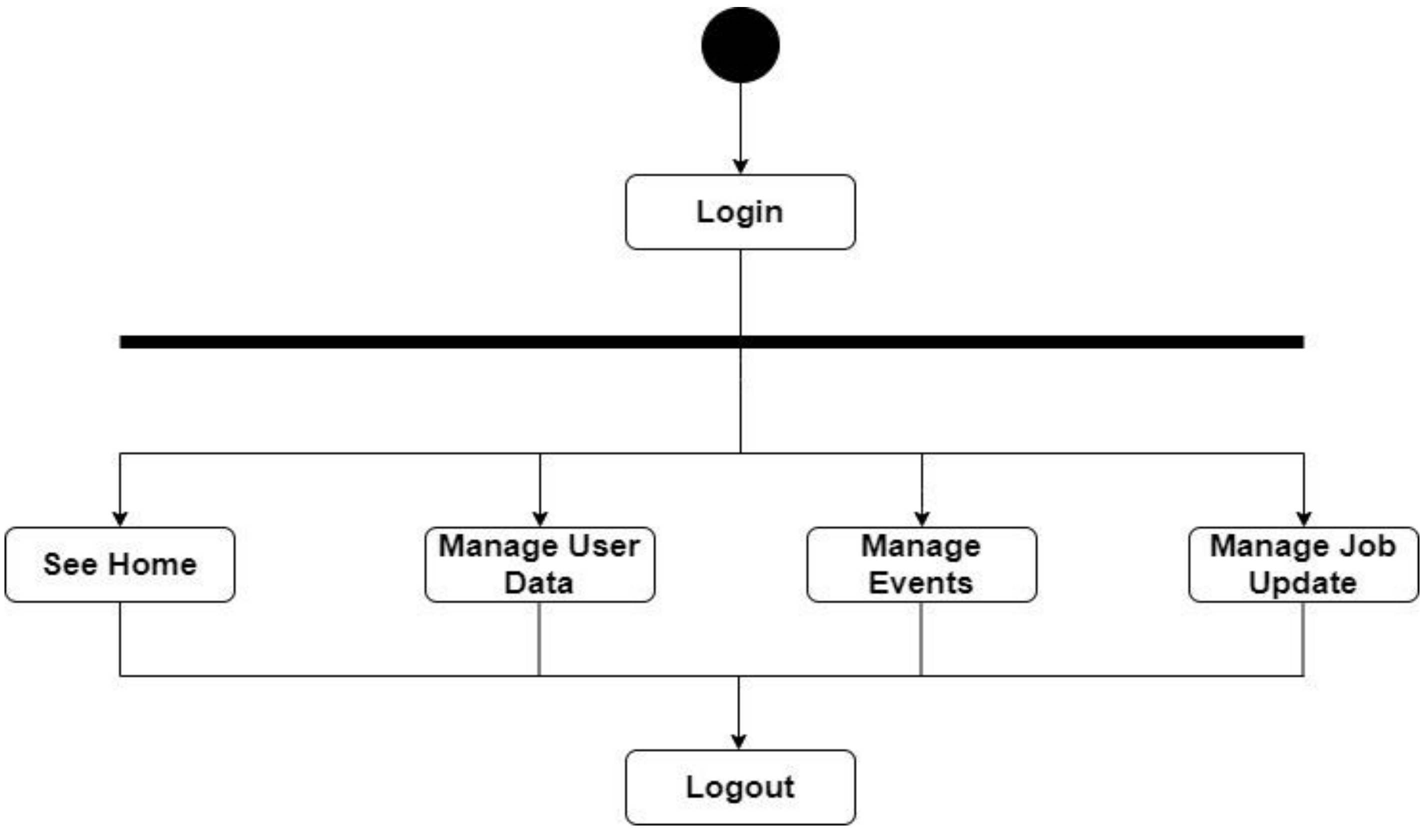


Fig. 4.3.1: Admin Activity Diagram

4.3.2: Alumni Activity Diagram:

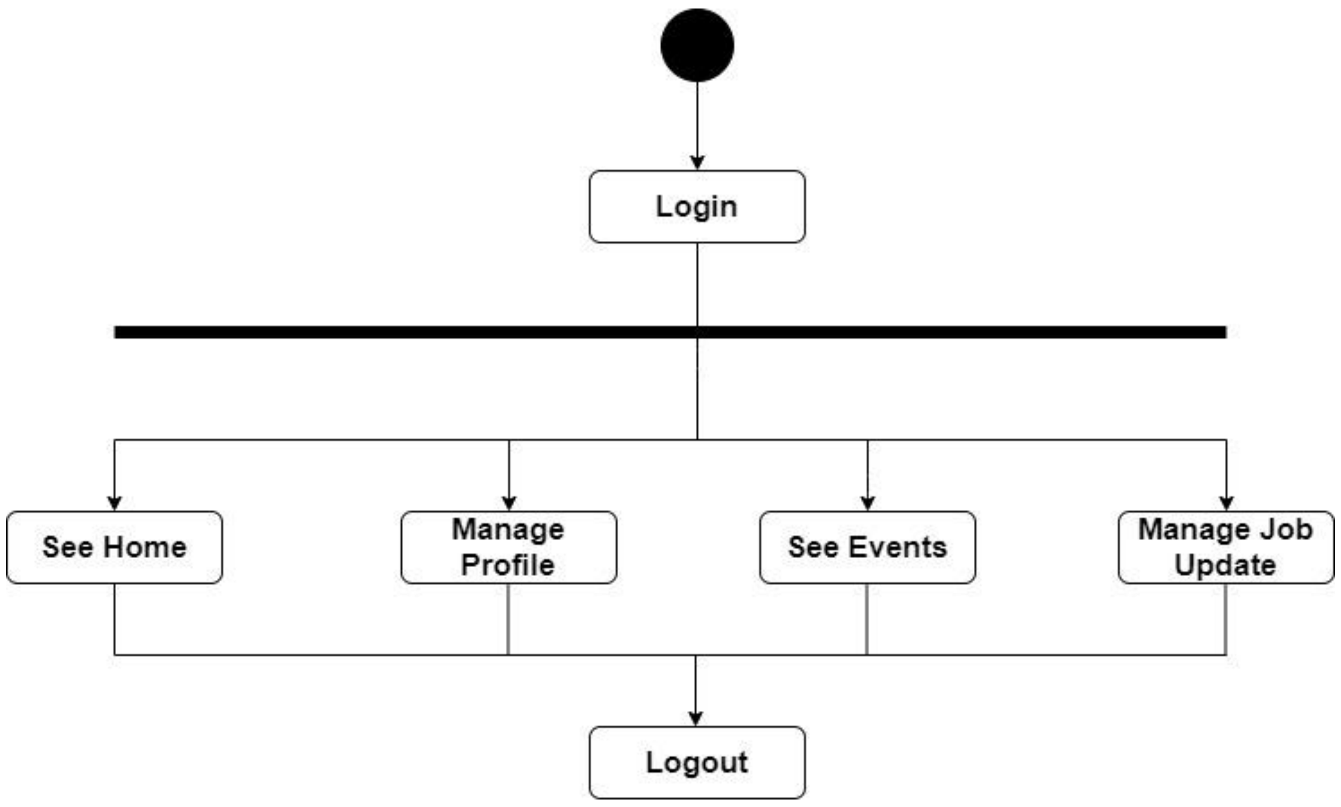


Fig. 4.3.2: Alumni Activity Diagram

4.3.3: Student Activity Diagram:

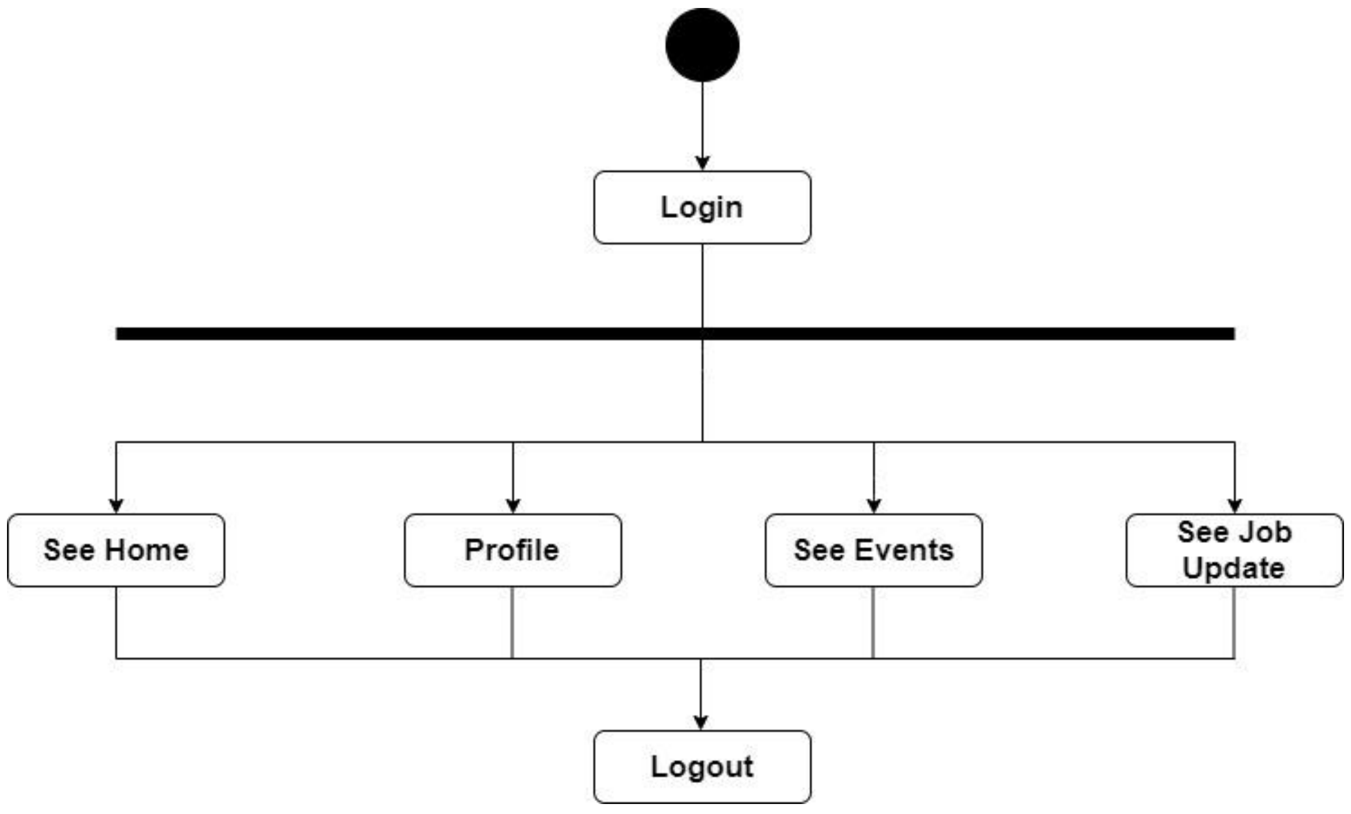


Fig. 4.3.3: Student Activity Diagram

4.4 Sequence Diagram:

4.4.1: Admin Sequence Diagram:

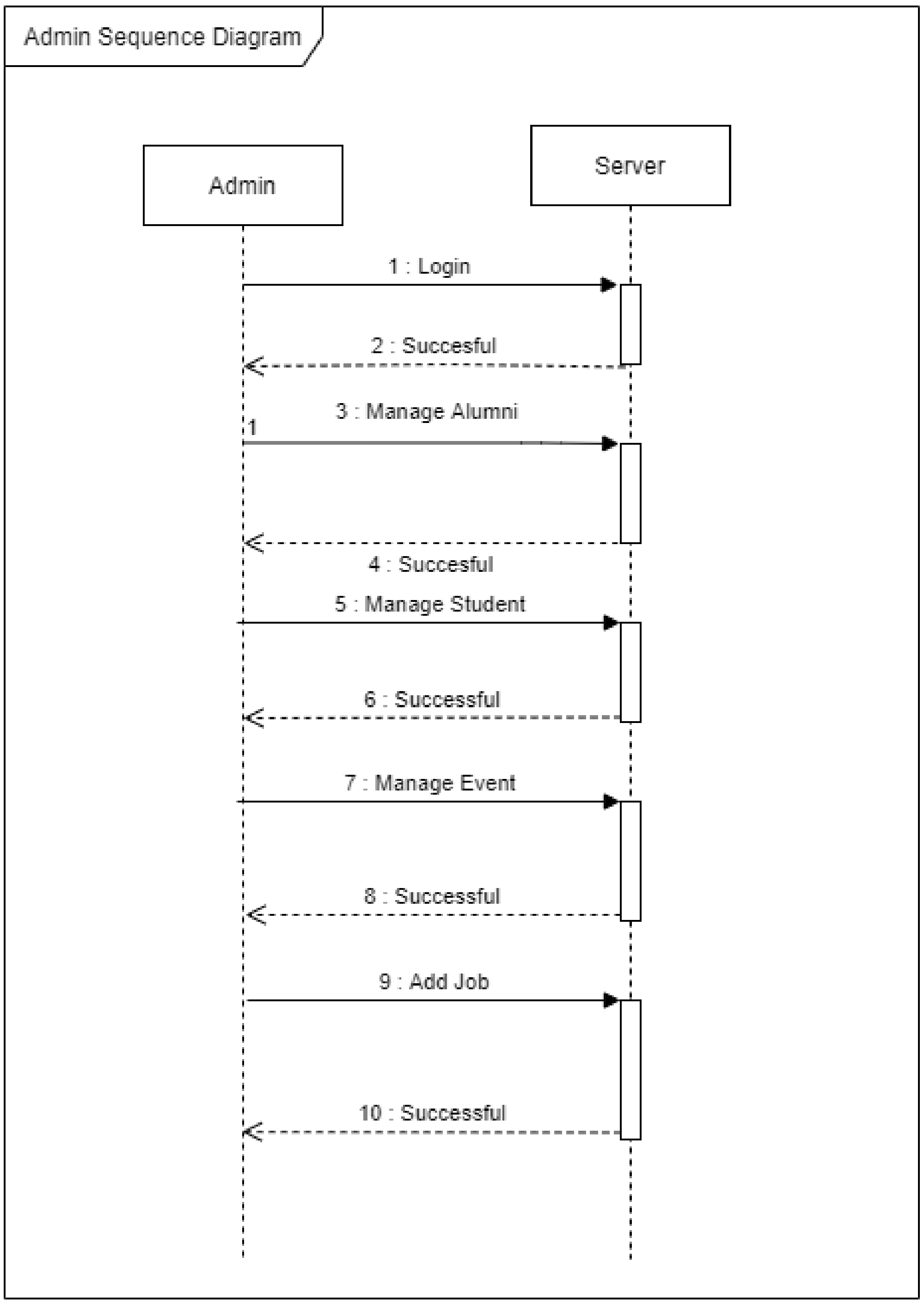


Fig. 4.4.1: Admin Sequence Diagram.

4.4.2: Alumni Sequence Diagram:

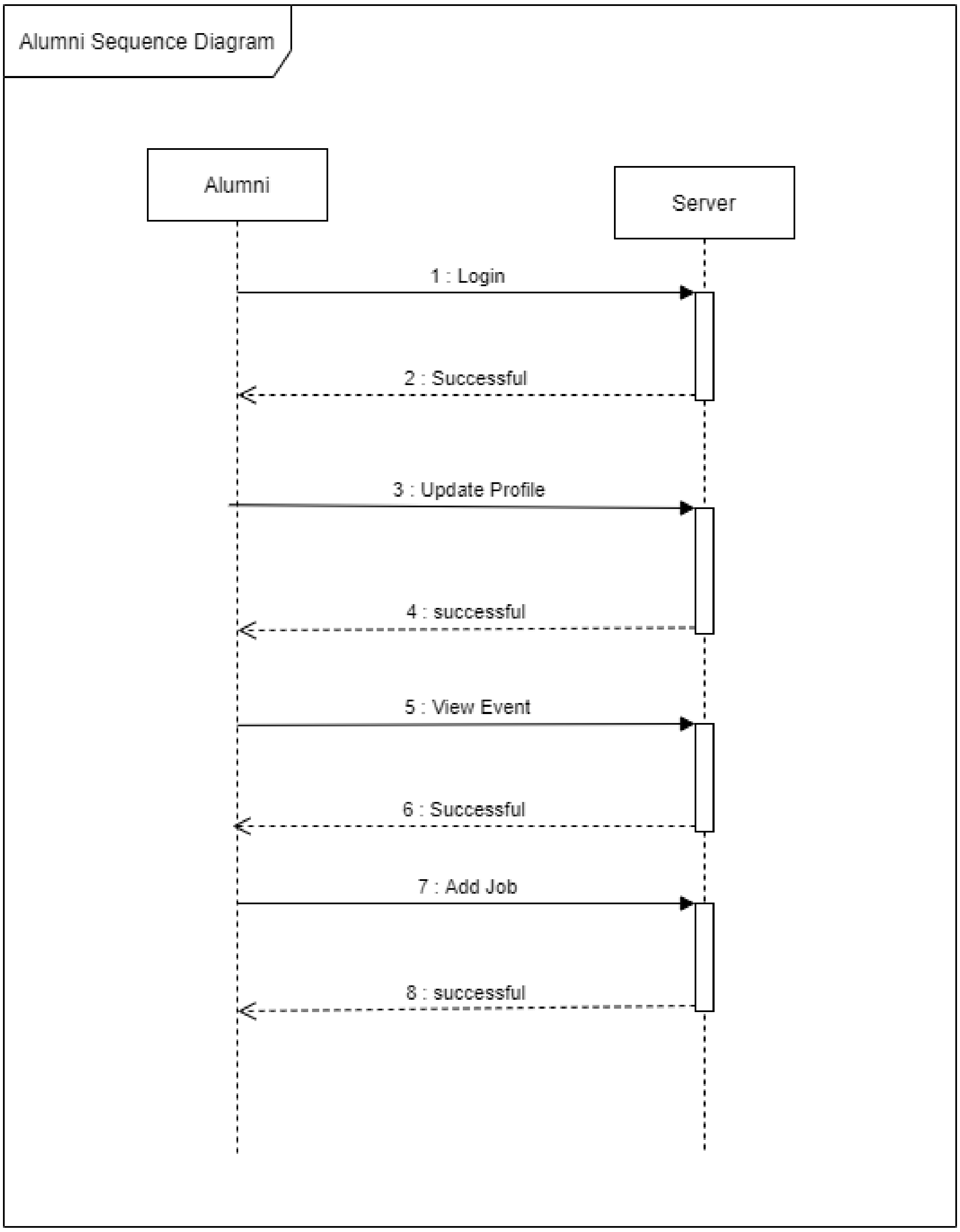


Fig. 4.4.2: Alumni Sequence Diagram

4.4.3: Student Sequence Diagram:

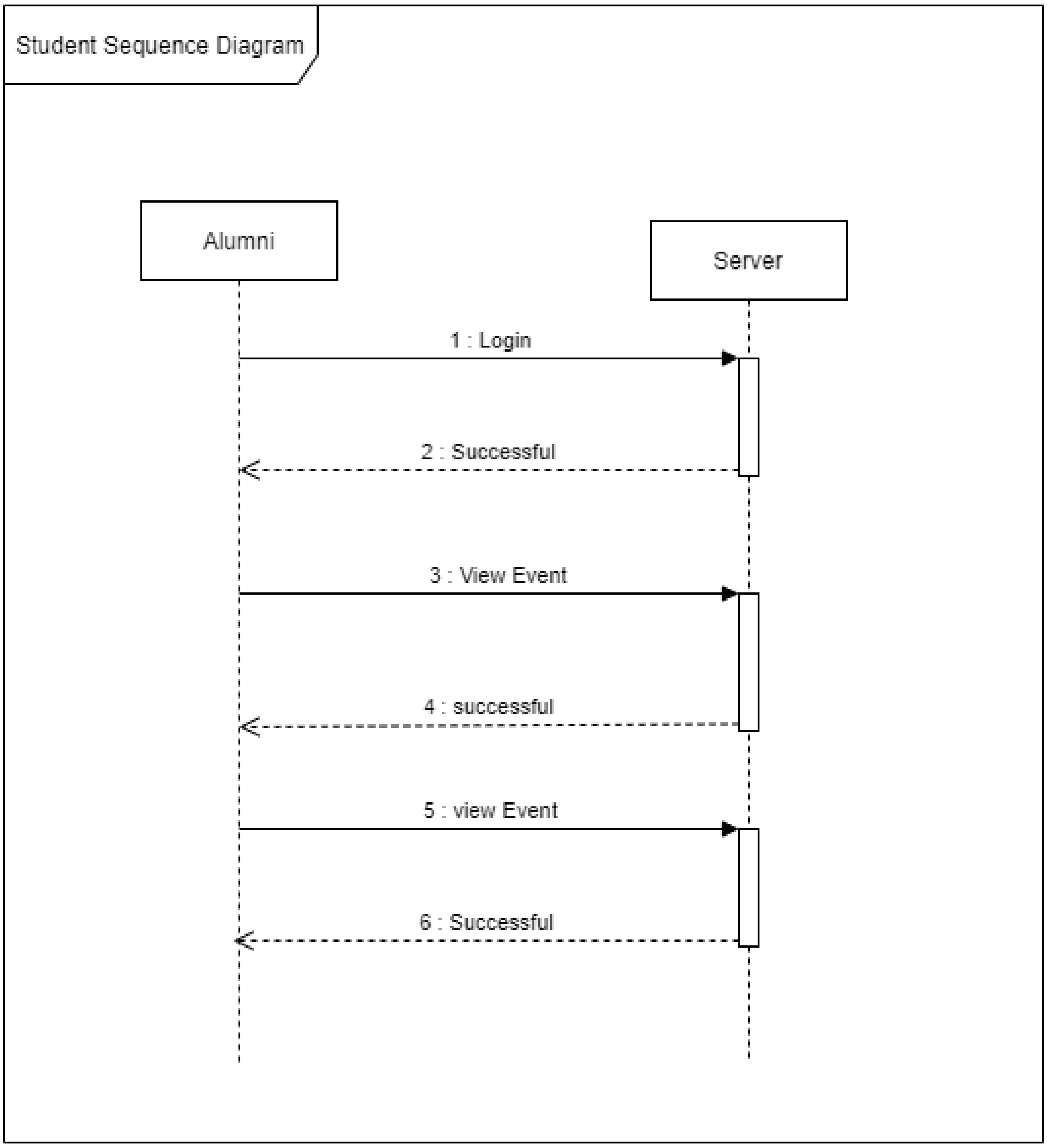


Fig. 4.4.3: Student Sequence Diagram

4.5 Timeline Chart:

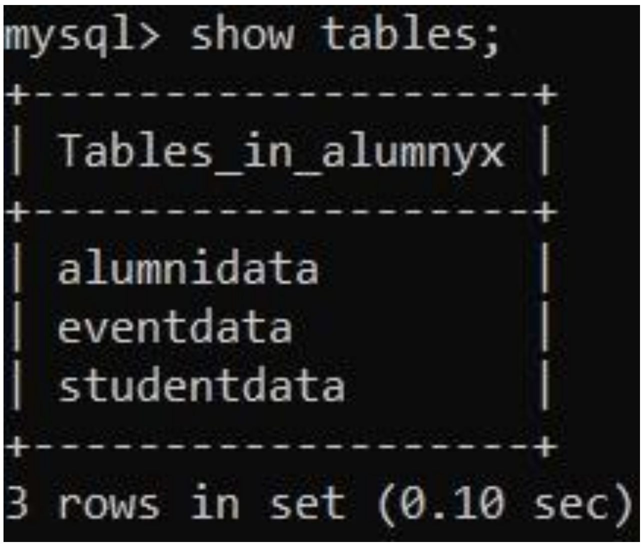
Task Name	Start Date	End Date	No of Days
Analysis	23/08/2021	05/09/2021	14
Evaluation & Recommendations	23/08/2021	26/08/2021	4
Requirements Gathering	27/08/2021	30/08/2021	4
Learning the technology	30/08/2021	02/09/2021	4
Literature Survey	04/09/2021	05/09/2021	2
Design	23/08/2021	27/08/2021	5
Design Database	23/08/2021	24/08/2021	2
Software Design	24/08/2021	25/08/2021	2
Software Design Document	25/08/2021	26/08/2021	1
Development	26/08/2021	28/08/2021	3
Verify & Validate user Requirement	28/08/2021	29/08/2021	1
Develop System Module	29/08/2021	30/08/2021	1
Perform initial Testing	30/08/2021	31/08/2021	1
Testing	31/08/2021	01/09/2021	2
Perform System Testing	01/09/2021	02/09/2021	1
Correct issued Found	02/09/2021	03/09/2021	1
Implementation & Maintenance	03/09/2021	05/09/2021	4
System Maintenance	03/09/2021	04/09/2021	1
Document Lessons Learned	04/09/2021	05/09/2021	1
Update Files/Records	05/09/2021	06/09/2021	1
Evaluation	06/09/2021		1

Chapter 5

Design

5.1 Data Modeling:

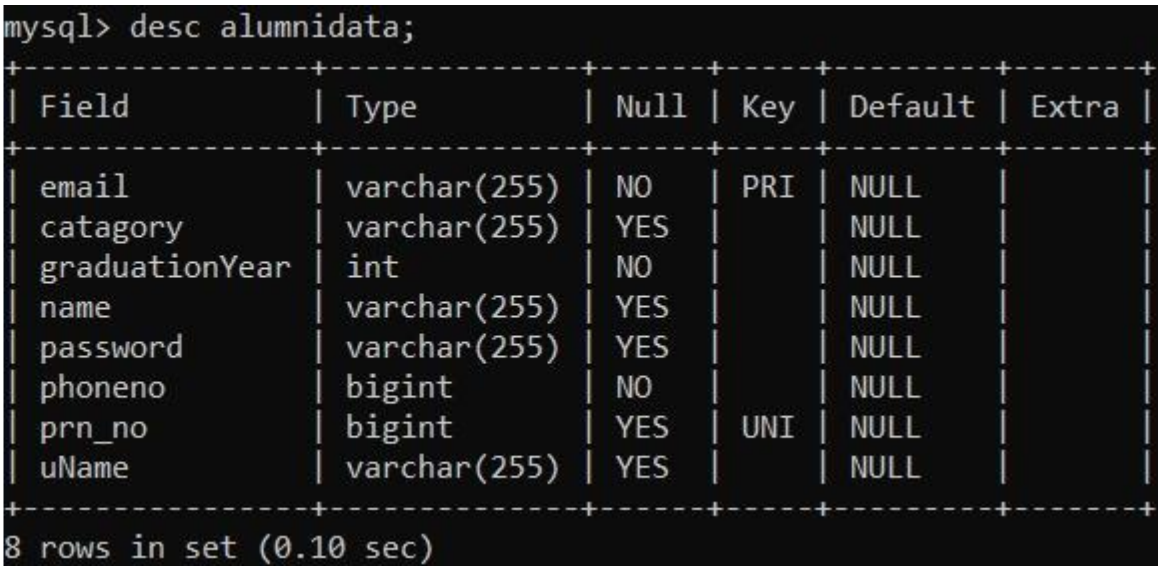
5.1.1: All Tables:



```
mysql> show tables;
+-------------------+
| Tables_in_alumnyx |
+-------------------+
| alumnidata        |
| eventdata         |
| studentdata       |
+-------------------+
3 rows in set (0.10 sec)
```

Fig. 5.1.1: All Tables.

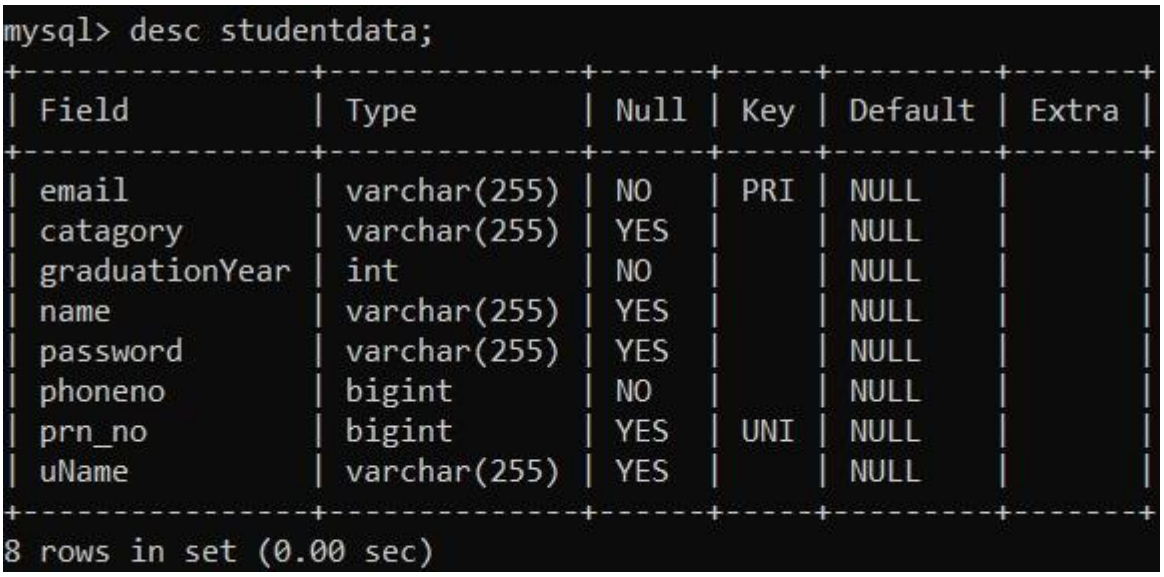
5.1.2: Alumni Data Table:



```
mysql> desc alumnidata;
+----------------+--------------+------+-----+---------+-------+
| Field          | Type         | Null | Key | Default | Extra |
+----------------+--------------+------+-----+---------+-------+
| email          | varchar(255) | NO   | PRI | NULL    |       |
| catagory       | varchar(255) | YES  |     | NULL    |       |
| graduationYear | int          | NO   |     | NULL    |       |
| name           | varchar(255) | YES  |     | NULL    |       |
| password       | varchar(255) | YES  |     | NULL    |       |
| phoneno        | bigint       | NO   |     | NULL    |       |
| prn_no         | bigint       | YES  | UNI | NULL    |       |
| uName          | varchar(255) | YES  |     | NULL    |       |
+----------------+--------------+------+-----+---------+-------+
8 rows in set (0.10 sec)
```

Fig. 5.1.2: Alumni Data Table.

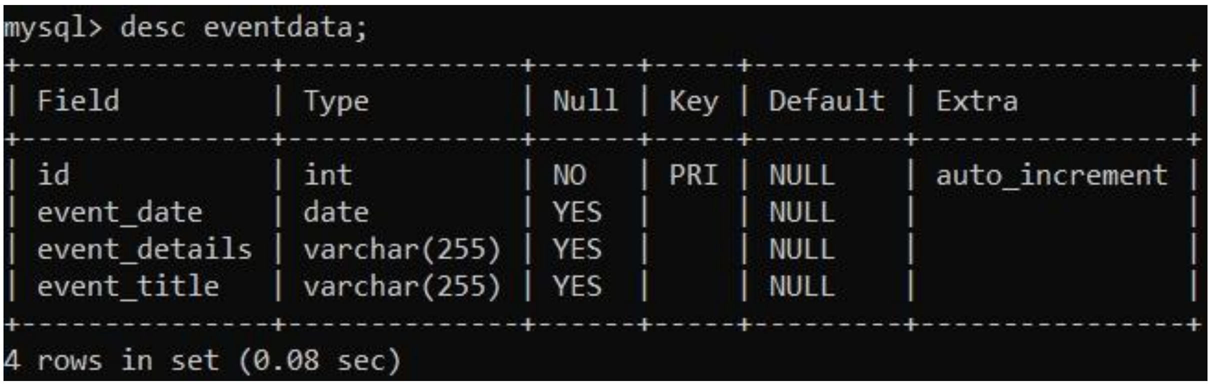
5.1.3: Student Data Table:



```
mysql> desc studentdata;
+----------------+--------------+------+-----+---------+-------+
| Field          | Type         | Null | Key | Default | Extra |
+----------------+--------------+------+-----+---------+-------+
| email          | varchar(255) | NO   | PRI | NULL    |       |
| catagory       | varchar(255) | YES  |     | NULL    |       |
| graduationYear | int          | NO   |     | NULL    |       |
| name           | varchar(255) | YES  |     | NULL    |       |
| password       | varchar(255) | YES  |     | NULL    |       |
| phoneno        | bigint       | NO   |     | NULL    |       |
| prn_no         | bigint       | YES  | UNI | NULL    |       |
| uName          | varchar(255) | YES  |     | NULL    |       |
+----------------+--------------+------+-----+---------+-------+
8 rows in set (0.00 sec)
```

Fig. 5.1.3: Student Data Table.

5.1.4: Event Data Table:

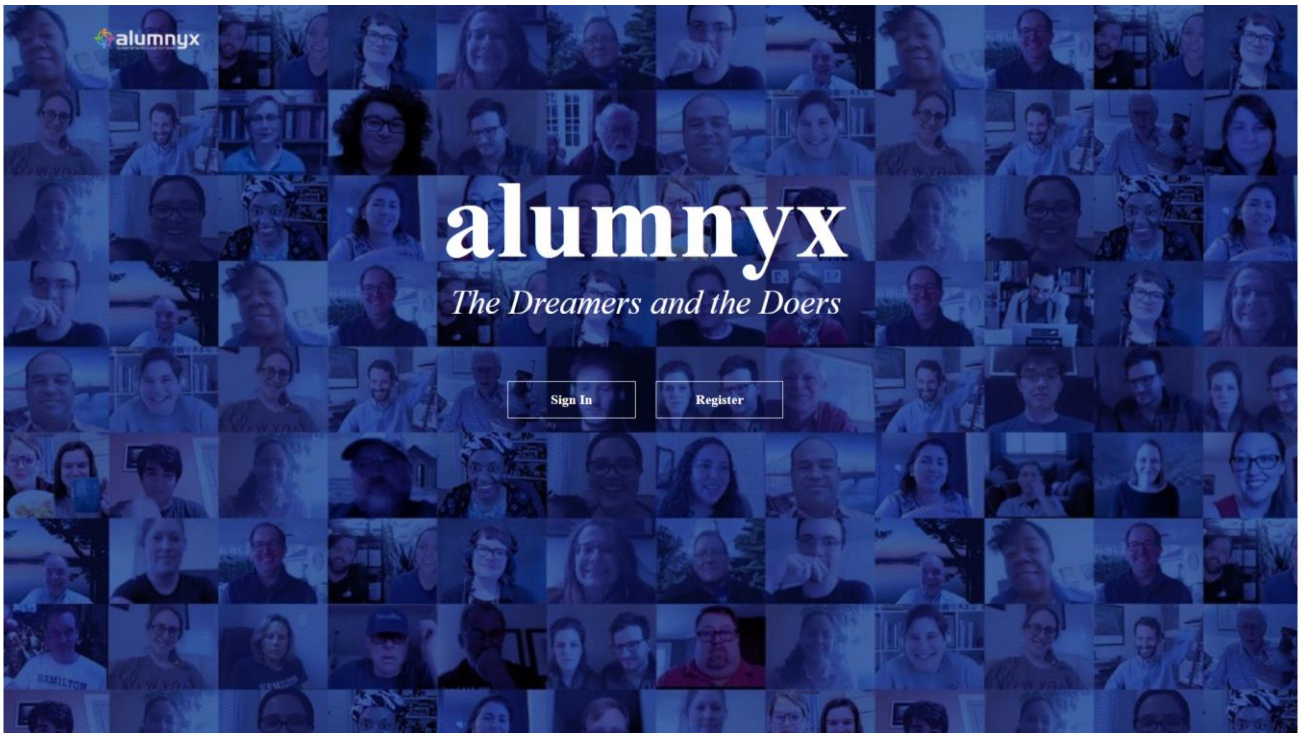


```
mysql> desc eventdata;
+---------------+--------------+------+-----+---------+----------------+
| Field         | Type         | Null | Key | Default | Extra          |
+---------------+--------------+------+-----+---------+----------------+
| id            | int          | NO   | PRI | NULL    | auto_increment |
| event_date    | date         | YES  |     | NULL    |                |
| event_details | varchar(255) | YES  |     | NULL    |                |
| event_title   | varchar(255) | YES  |     | NULL    |                |
+---------------+--------------+------+-----+---------+----------------+
4 rows in set (0.08 sec)
```

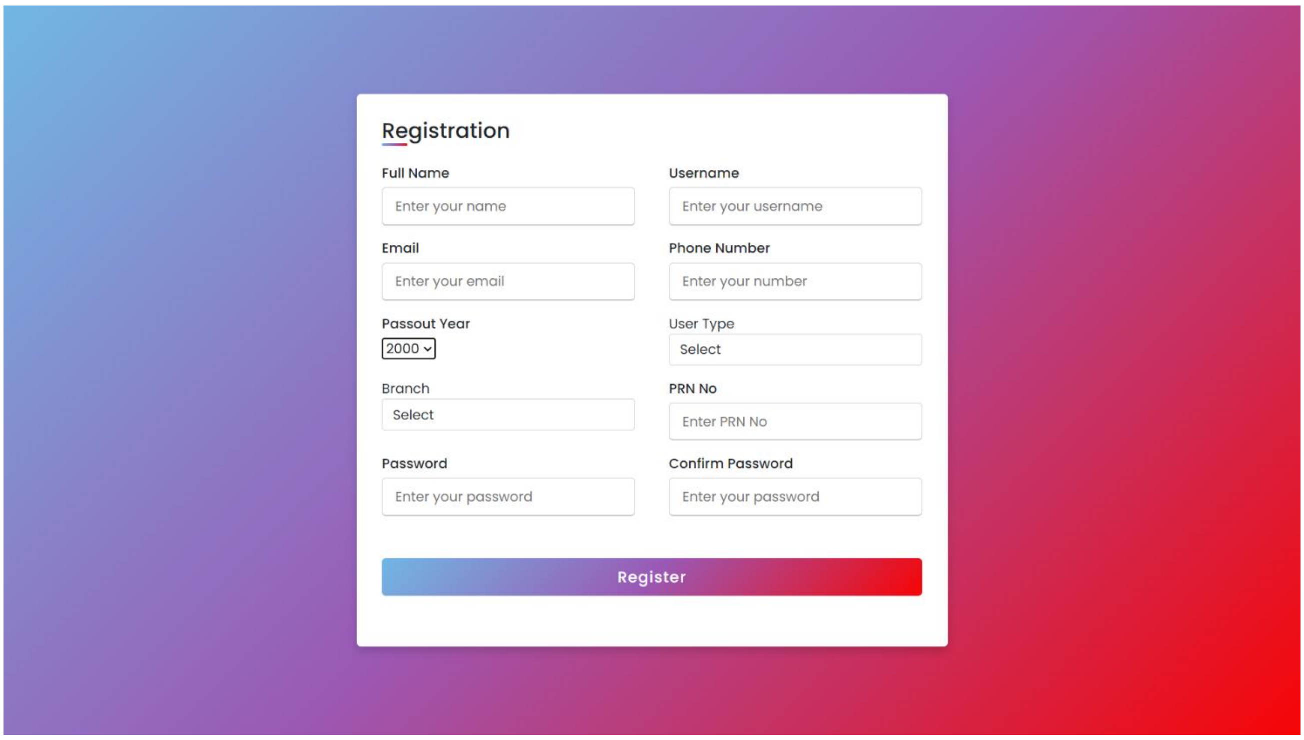
Fig. 5.1.4: Event Data Table.

5.2 User Interface Design:

5.2.1 Welcome Page:



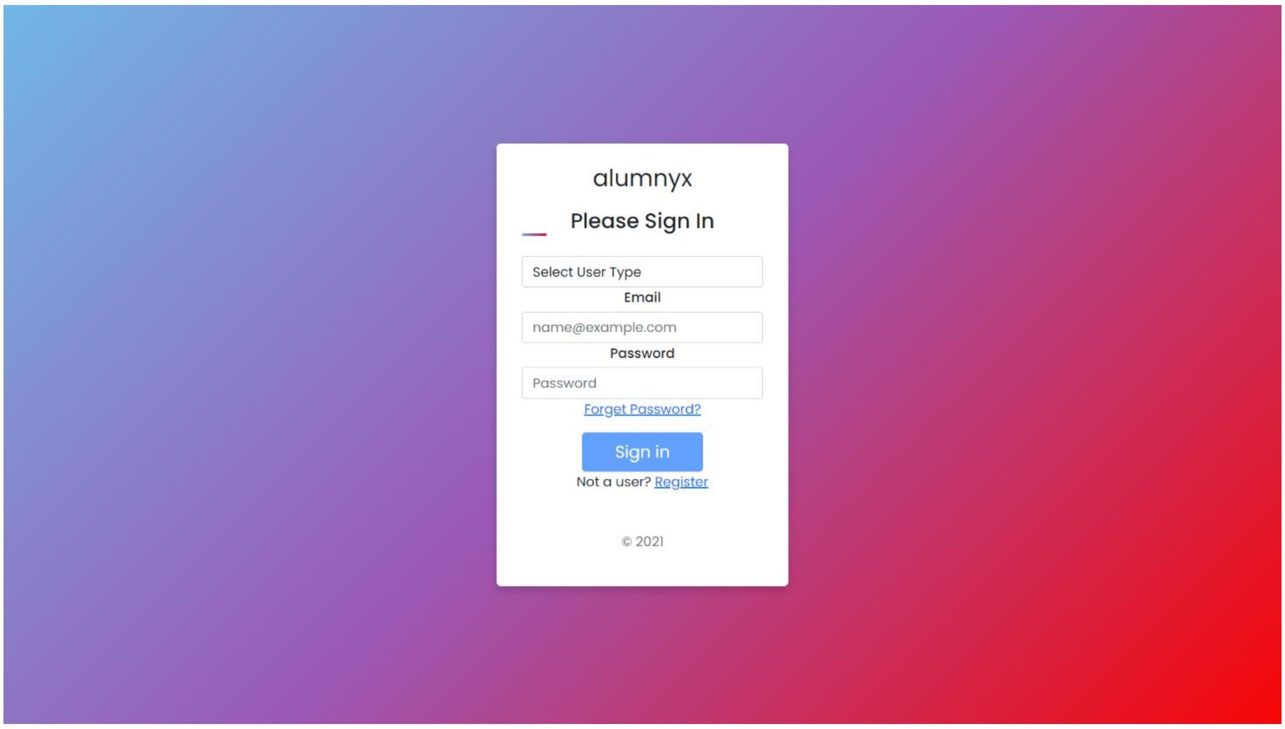
5.2.1 Registration Page:



Registration

Full Name	Username
Enter your name	Enter your username
Email	Phone Number
Enter your email	Enter your number
Passout Year	User Type
2000	Select
Branch	PRN No
Select	Enter PRN No
Password	Confirm Password
Enter your password	Enter your password

5.2.3 Login Page:



alumnyx

Please Sign In

Select User Type

Email

name@example.com

Password

Password

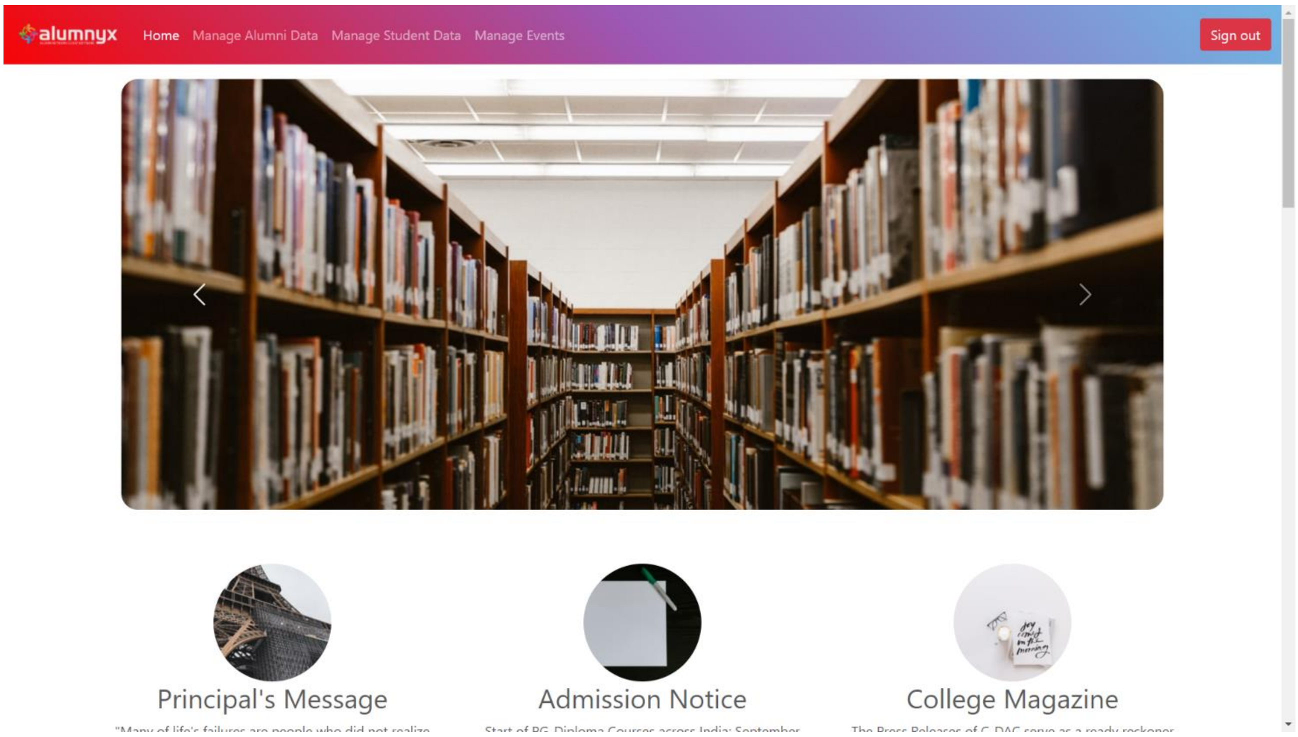
Forget Password?

Sign in

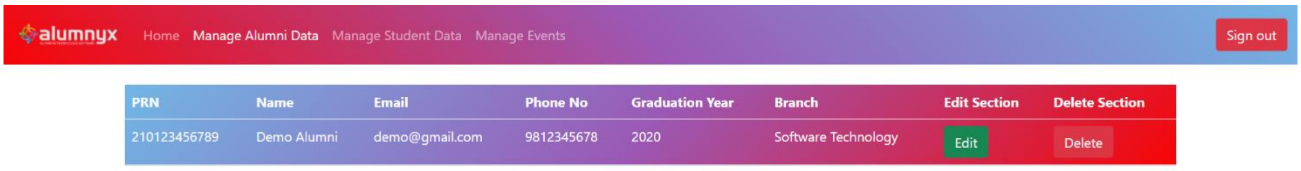
Not a user? Register

© 2021

5.2.4: Admin Home Page:

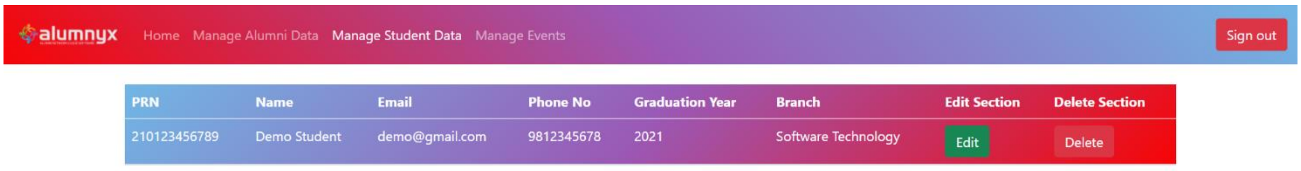

alumnyx
Home
Manage Alumni Data
Manage Student Data
Manage Events
Sign out
Principal's Message
"Many of life's failures are people who did not realize
Admission Notice
Start of PG Diploma Courses across India: September
College Magazine
The Press Releases of C-DAC serve as a ready reckoner

5.2.5: Manage Alumni Data Page:



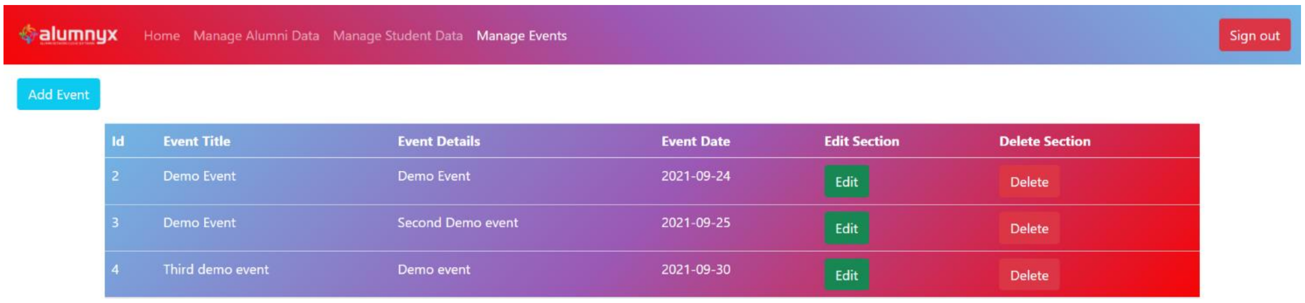
PRN	Name	Email	Phone No	Graduation Year	Branch	Edit Section	Delete Section
210123456789	Demo Alumni	demo@gmail.com	9812345678	2020	Software Technology	Edit	Delete

5.2.6: Manage Student Data Page:



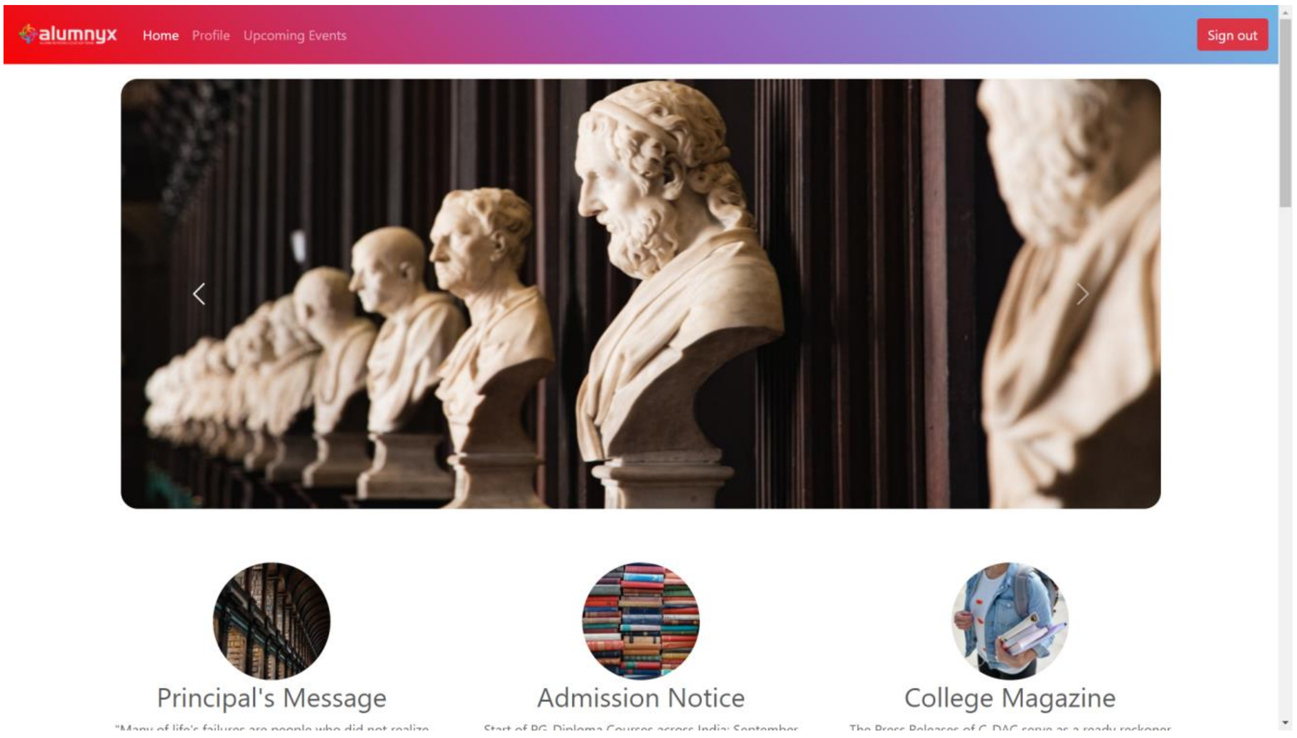
PRN	Name	Email	Phone No	Graduation Year	Branch	Edit Section	Delete Section
210123456789	Demo Student	demo@gmail.com	9812345678	2021	Software Technology	Edit	Delete

5.2.7: Manage Events Page:



Id	Event Title	Event Details	Event Date	Edit Section	Delete Section
2	Demo Event	Demo Event	2021-09-24	Edit	Delete
3	Demo Event	Second Demo event	2021-09-25	Edit	Delete
4	Third demo event	Demo event	2021-09-30	Edit	Delete

5.2.8: Alumni & Student Home Page:



Principal's Message

"Many of life's failures are people who did not realize

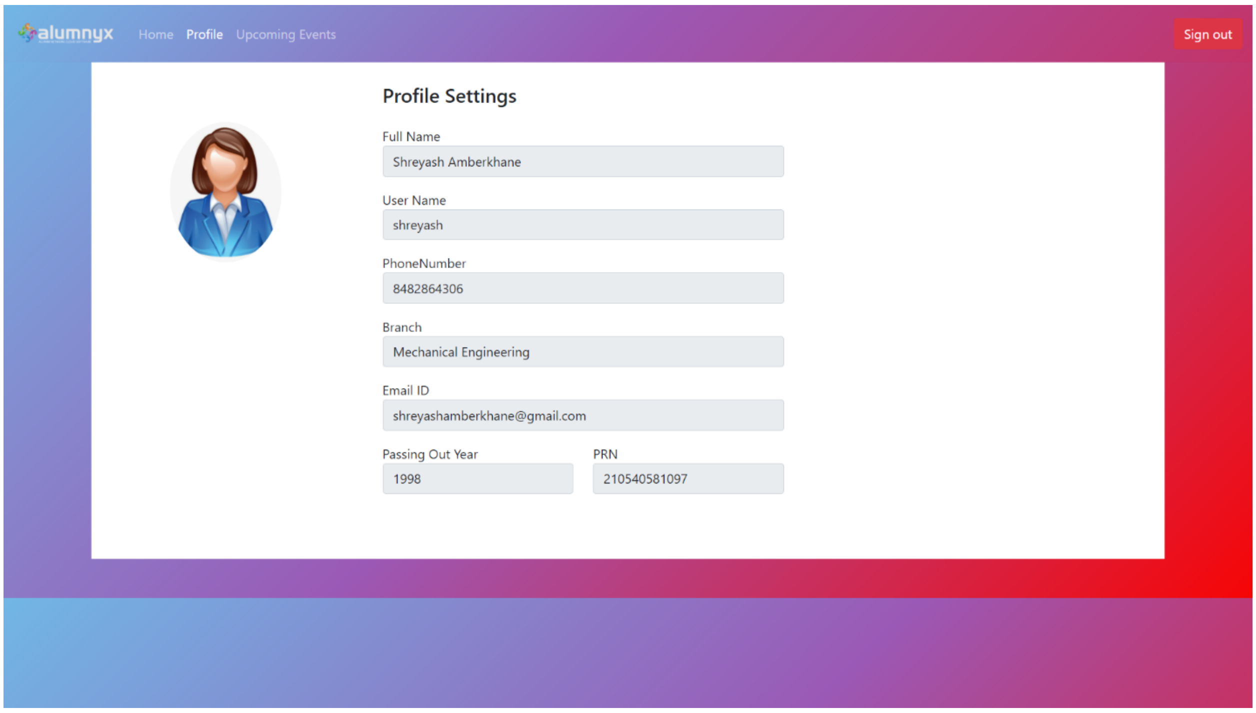
Admission Notice

Start of PG Diploma Courses across India: September

College Magazine

The Press Releases of C-DAC serve as a ready reckoner

5.2.9: Alumni Profile Page:



Profile Settings

Full Name
Shreyash Amberkhane

User Name
shreyash

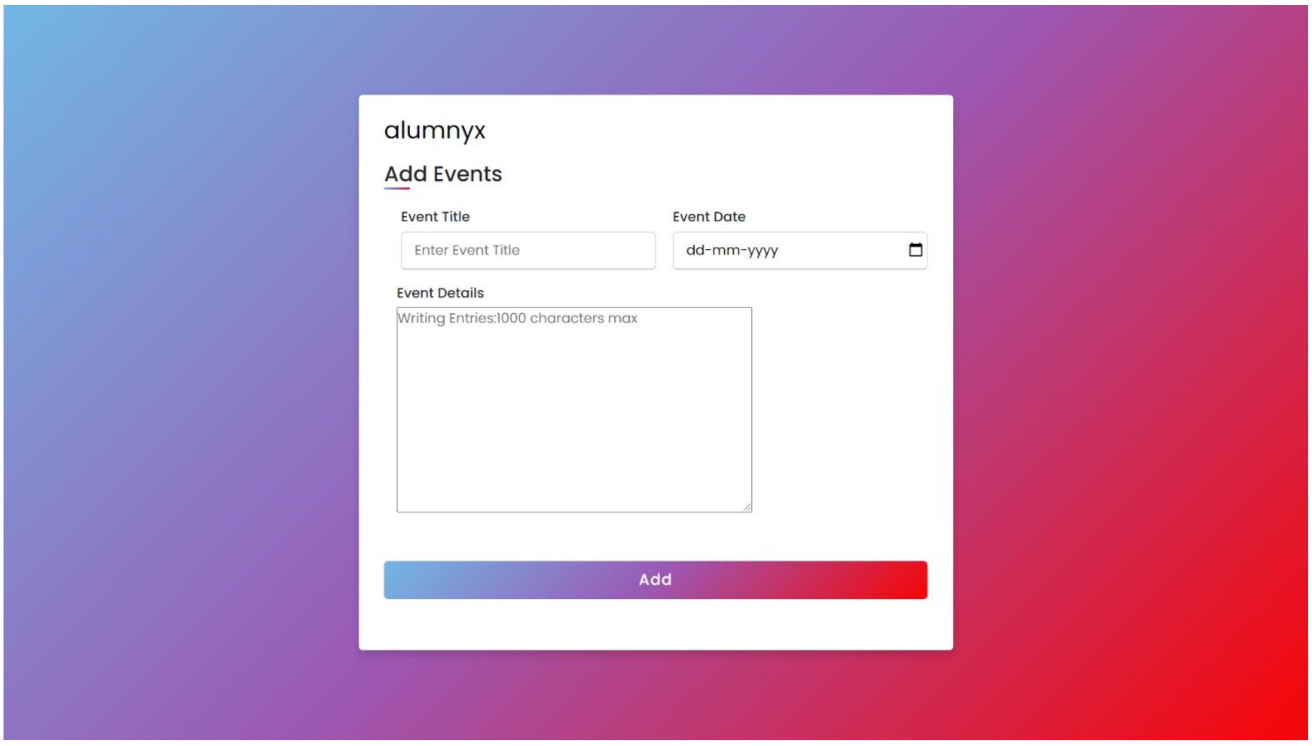
PhoneNumber
8482864306

Branch
Mechanical Engineering

Email ID
shreyashamberkhane@gmail.com

Passing Out Year	PRN
1998	210540581097

5.2.10: Add Events Page:



alumnyx

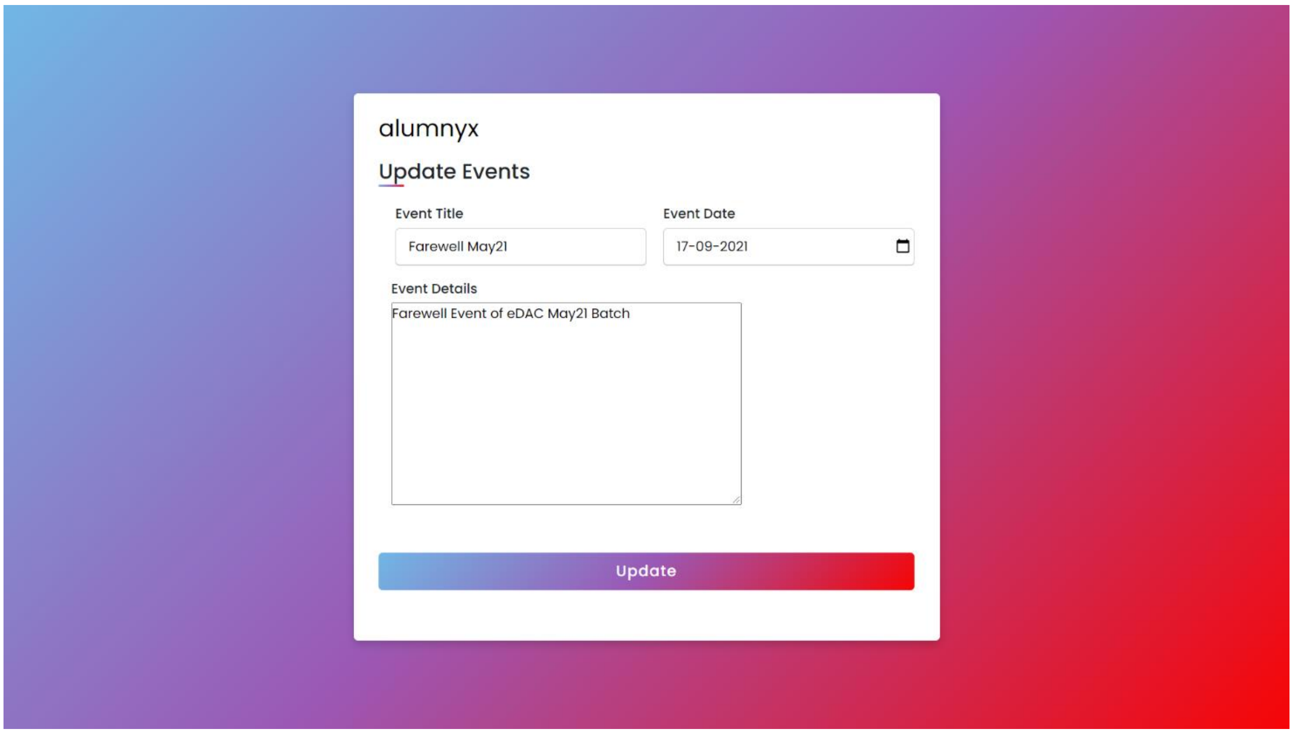
Add Events

Event Title
Enter Event Title

Event Date
dd-mm-yyyy

Event Details
Writing Entries:1000 characters max

5.2.11: Update Events Page:



alumnyx

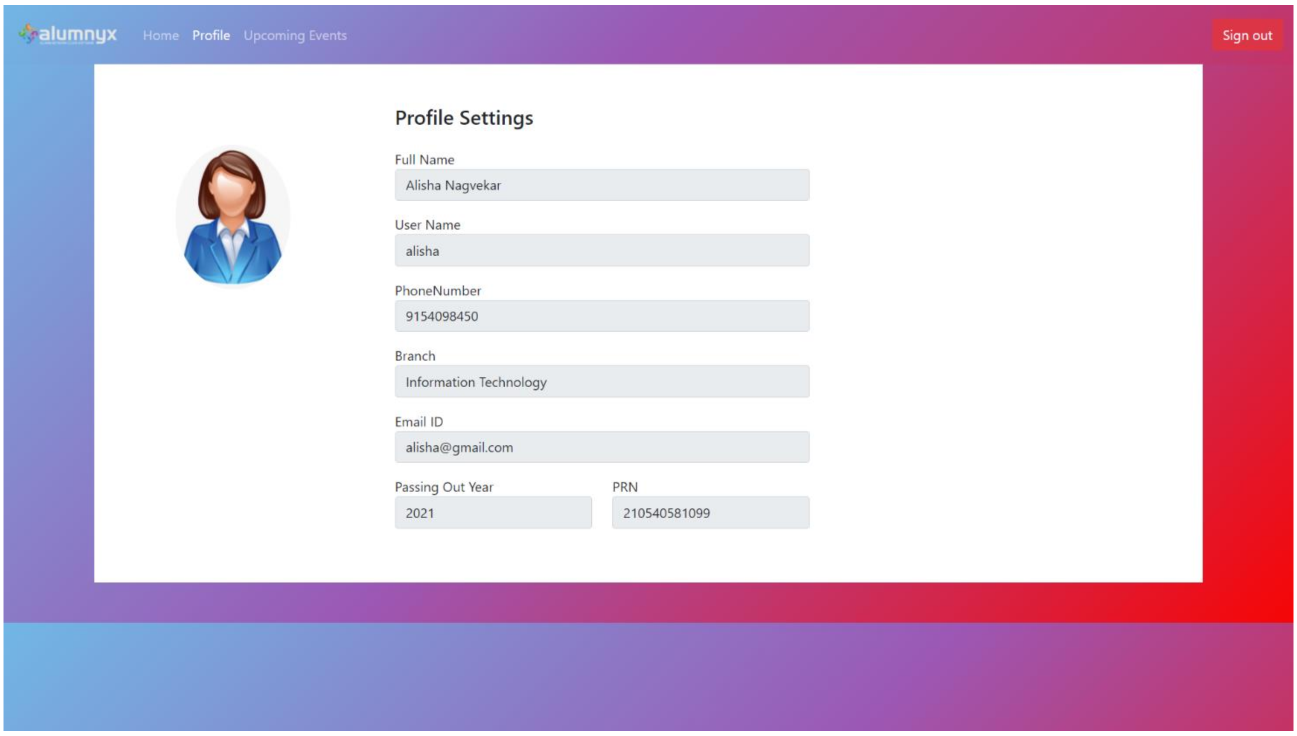
Update Events

Event Title: Farewell May21

Event Date: 17-09-2021

Event Details: Farewell Event of eDAC May21 Batch

5.2.12: Student Profile Page:



Profile Settings

Full Name: Alisha Nagvekar

User Name: alisha

PhoneNumber: 9154098450

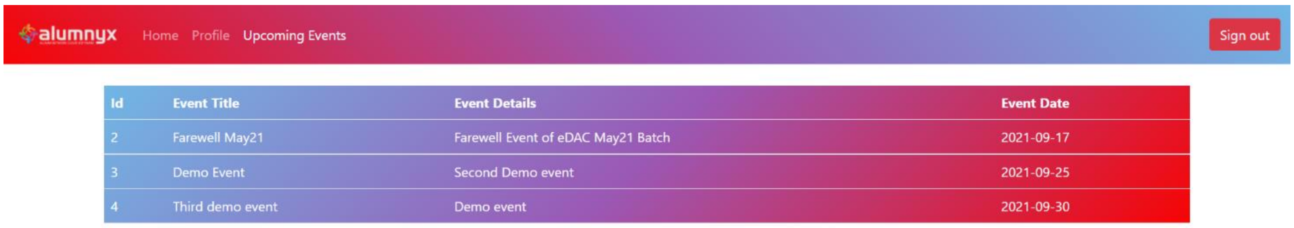
Branch: Information Technology

Email ID: alisha@gmail.com

Passing Out Year: 2021

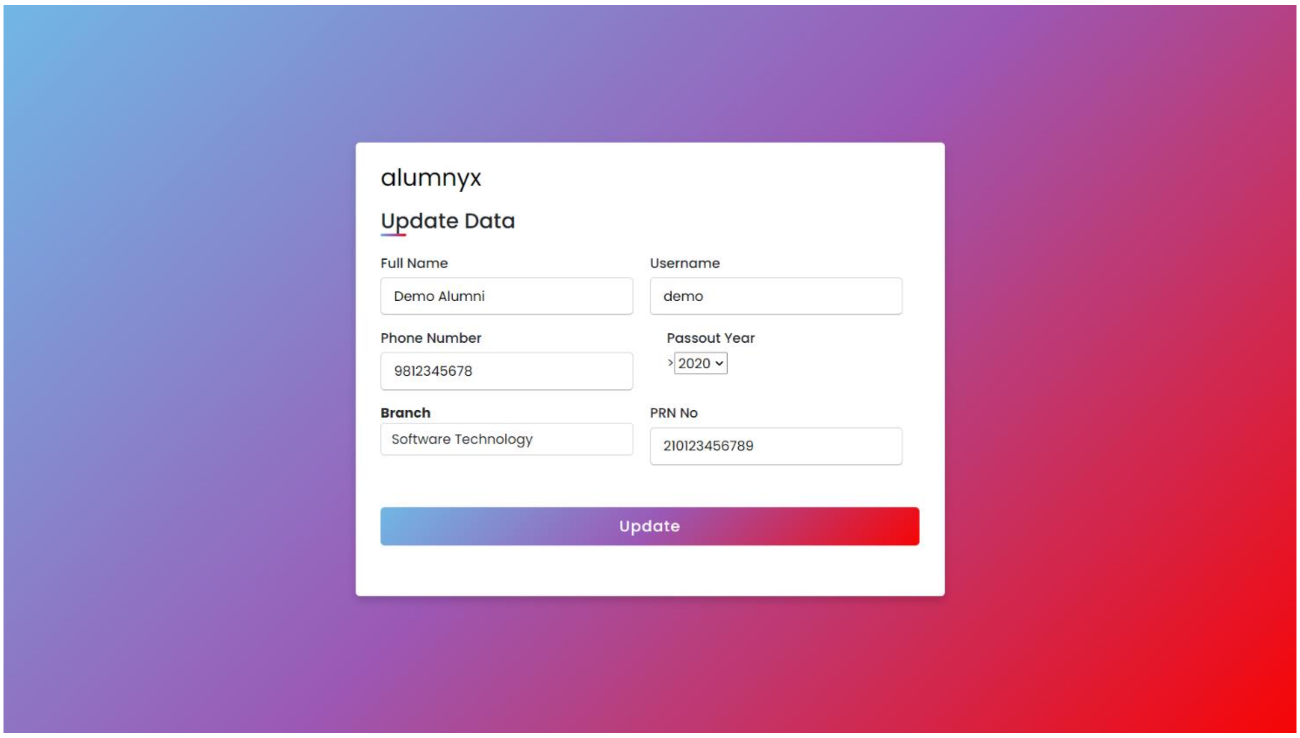
PRN: 210540581099

5.2.13: Upcoming Events Page:



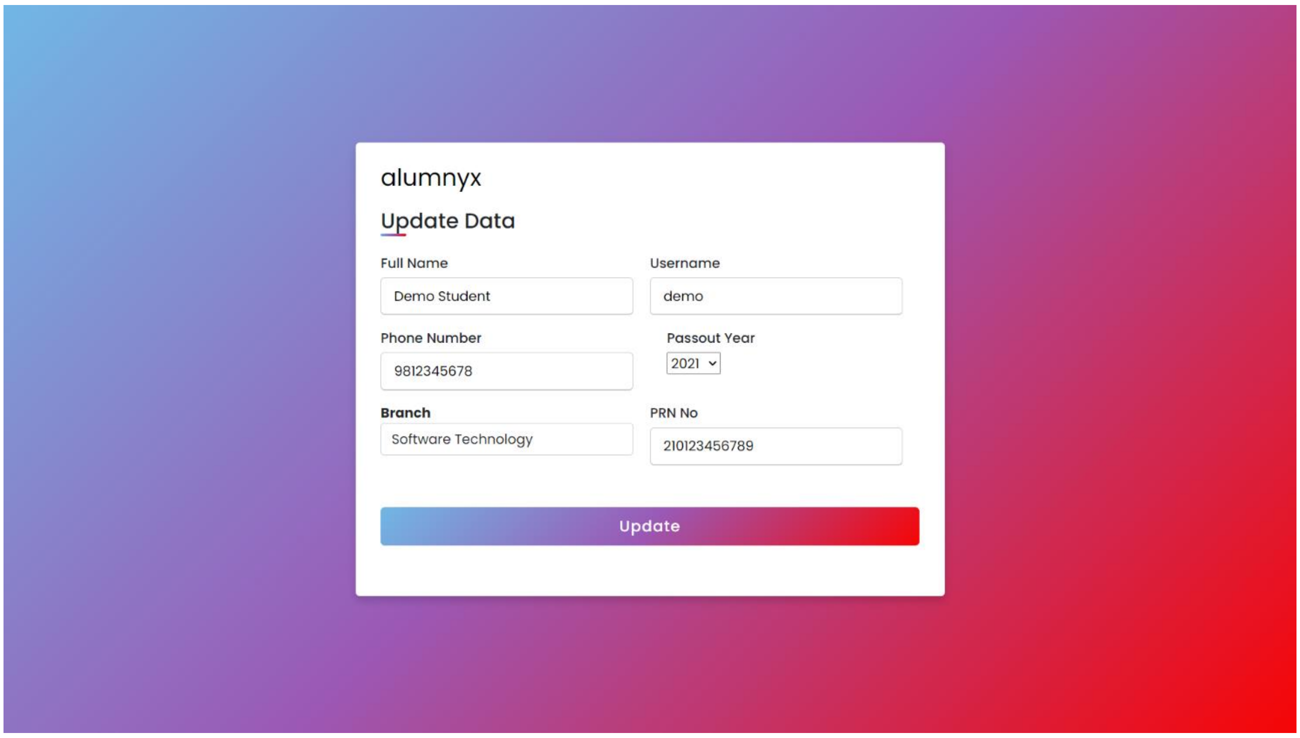
Id	Event Title	Event Details	Event Date
2	Farewell May21	Farewell Event of eDAC May21 Batch	2021-09-17
3	Demo Event	Second Demo event	2021-09-25
4	Third demo event	Demo event	2021-09-30

5.2.14: Update Alumni Page:



Full Name: Demo Alumni	Username: demo
Phone Number: 9812345678	Passout Year: 2020
Branch: Software Technology	PRN No: 210123456789

5.2.15: Update Student Data:



alumnyx

Update Data

Full Name: Demo Student

Username: demo

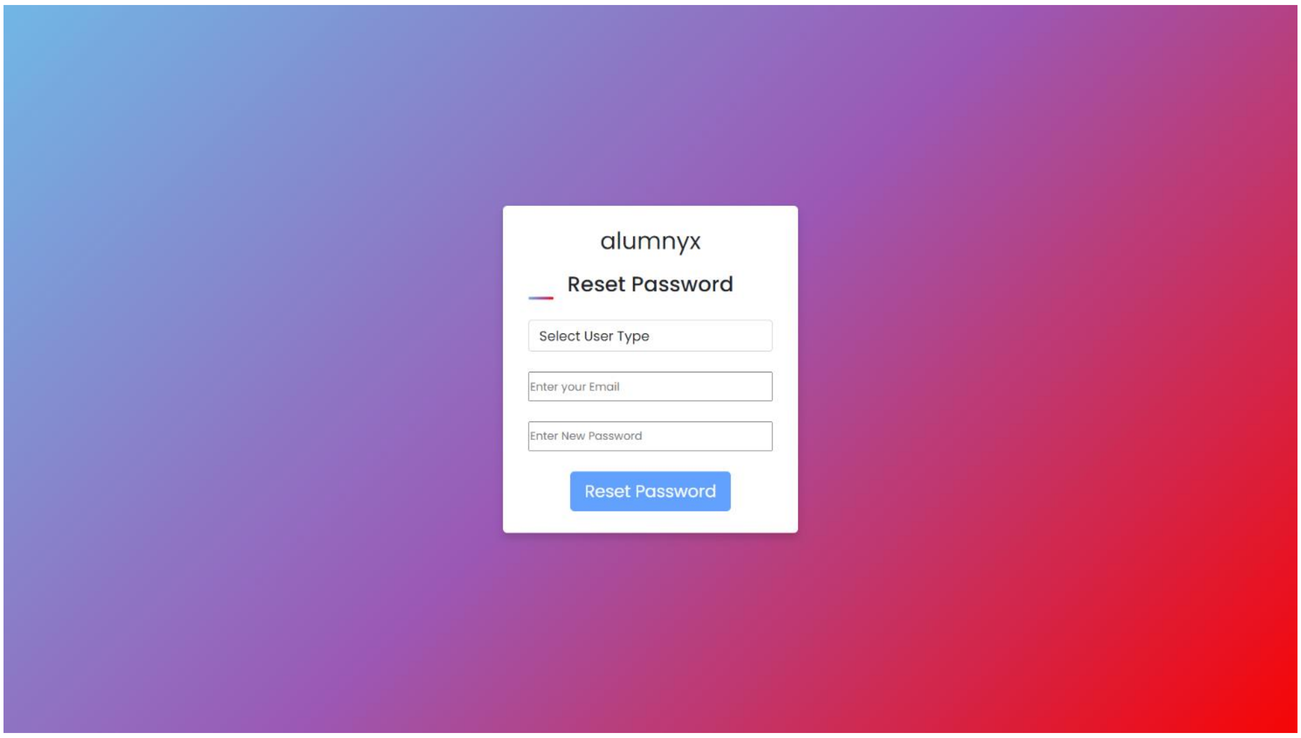
Phone Number: 9812345678

Passout Year: 2021

Branch: Software Technology

PRN No: 210123456789

5.2.15: Forget Password:



alumnyx

Reset Password

Chapter 6

Implementation

6.1 Algorithms/ Methods Used:

6.1.1 Register User:
We had stored the user details in database viz. alumni & student data.

6.1.2: Manage User Data:
Admin can perform CRUD operations on the stored user data.

6.1.3: Forget Password:
If a user forgets his/her password, our system has provisions for resetting the password.

6.2 Working of Project (code for mentioned algorithms)

6.2.1 CRUD operations forAdmin:

a) On Alumni Data:

```
package model;

import java.util.List;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.Configuration;


import entities.AlumniData;

import sessionUtils.SessionUtils;

public class AlumniDaoImpl implements AlumniDao {

	private Session getSession() {
		Configuration config = new Configuration();
		config.configure("hibernate.cfg.xml");
		SessionFactory sf = config.buildSessionFactory();
		Session session = sf.openSession();
		return session;
	}
	public AlumniData insertData(AlumniData record) {
		Session session = getSession();
		Transaction txn = session.beginTransaction();

		session.save(record);
		txn.commit();
		session.close();
		return record;
	}

	public AlumniData updateData(AlumniData alumni) {
		Session session = getSession();
		Transaction txn = session.beginTransaction();
		session.update(alumni);
		txn.commit();
		session.close();
		return alumni;
	}

	public AlumniData deleteData(AlumniData alumni) {
		Session session = getSession();
		Transaction txn = session.beginTransaction();

		AlumniData data = session.get(AlumniData.class, alumni.getEmail());

		session.delete(data);
		txn.commit();
		session.close();
		return alumni;
	}

	public List<AlumniData> getAllData() {
		Session session=getSession();
		Transaction tr = session.beginTransaction();
		@SuppressWarnings("unchecked")
		List<AlumniData> list=session.createQuery("from
AlumniData").getResultList();
		tr.commit();
		session.close();
		return list;
	}

	public AlumniData aluminiLogin(AlumniData record) {
		Session session = SessionUtils.getFactory().openSession();
		Transaction txn = session.beginTransaction();

		AlumniData data = session.get(AlumniData.class, record.getEmail());
		if (data.getEmail().equals(record.getEmail()) &&
data.getPassword().equals(record.getPassword())) {
			txn.commit();
			session.close();
			return data;
		}
		return null;
	}

	public AlumniData getAlumnibyEmail(AlumniData record) {
		Session session = SessionUtils.getFactory().openSession();
		Transaction txn = session.beginTransaction();

		AlumniData data = session.get(AlumniData.class, record.getEmail());
		txn.commit();
		session.close();
		return data;
	}


	public AlumniData changePassword(AlumniData a) {
		Session session = getSession();
		Transaction txn = session.beginTransaction();
//		session.update(a);
		AlumniData data = session.get(AlumniData.class, a.getEmail());
		data.setPassword(a.getPassword());
		txn.commit();
		session.close();
		return a;

	}

}
```

b) On Student Data:

```
package model;

import java.util.List;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.Configuration;

import entities.StudentData;
import sessionUtils.SessionUtils;
```

```
public class StudentDaoImpl implements StudentDao {

	private Session getSession() {
		Configuration config = new Configuration();
		config.configure("hibernate.cfg.xml");
		SessionFactory sf = config.buildSessionFactory();
		Session session = sf.openSession();
		return session;
	}

	public StudentData insertData(StudentData record) {
		Session session = SessionUtils.getFactory().openSession();
		Transaction txn = session.beginTransaction();

		session.save(record);
		txn.commit();
		session.close();
		return record;
	}

	public StudentData updateStudentData(StudentData s) {
		Session session = getSession();
		Transaction txn = session.beginTransaction();
		session.update(s);
		txn.commit();
		session.close();
		return s;
	}

	public StudentData deleteData(StudentData student) {
		Session session = getSession();
		Transaction txn = session.beginTransaction();
		StudentData data = session.get(StudentData.class, student.getEmail());
		session.delete(data);
		txn.commit();
		session.close();
		return student;
	}

	public List<StudentData> getAllData() {
		Session session=getSession();
		Transaction tr = session.beginTransaction();
		@SuppressWarnings("unchecked")
		List<StudentData> list1=session.createQuery("from
StudentData").getResultList();
		tr.commit();
		session.close();
		return list1;
	}

	public StudentData studentLogin(StudentData record) {
		Session session = SessionUtils.getFactory().openSession();
		Transaction txn = session.beginTransaction();

		StudentData data = session.get(StudentData.class, record.getEmail());

		if (data.getEmail().equals(record.getEmail()) &&
data.getPassword().equals(record.getPassword())) {
			txn.commit();
			session.close();
			return data;
		}
		return null;
	}

	public StudentData getStudentbyEmail(StudentData record) {
		Session session = SessionUtils.getFactory().openSession();
		Transaction txn = session.beginTransaction();

		StudentData data = session.get(StudentData.class, record.getEmail());
		txn.commit();
		session.close();
		return data;
	}

	public StudentData updateData(StudentData data) {
		Session session = getSession();
		Transaction txn = session.beginTransaction();
		session.update(data);
		txn.commit();
		session.close();
		return data;
	}

	public StudentData changePassword(StudentData a) {
		Session session = getSession();
		Transaction txn = session.beginTransaction();
//		session.update(a);
		StudentData data = session.get(StudentData.class, a.getEmail());
		data.setPassword(a.getPassword());
		txn.commit();
		session.close();
		return a;
	}
```

```
}
```

6.2.2 Forget Password:

```
package controller;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import entities.AlumniData;
import entities.StudentData;
import model.AlumniDao;
import model.AlumniDaoImpl;
import model.StudentDao;
import model.StudentDaoImpl;

public class ForgetPassword extends HttpServlet {
	private static final long serialVersionUID = 1L;

	protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {

		HttpSession session = request.getSession();
		String email = request.getParameter("email");
		String password = request.getParameter("password");
		String role = request.getParameter("role");

		if (role.equals("Alumini")) {

			AlumniDao aluminiDao = new AlumniDaoImpl();

			AlumniData aluminiRecord = new AlumniData(email, password);
			AlumniData data = aluminiDao.changePassword(aluminiRecord);

			if(data!=null) {
				session.setAttribute("SuccessMsg", "Password has
Changed");
				response.sendRedirect("login.jsp");
			}
			else
			{
				session.setAttribute("failedMsg", "Password Updation is
failed!!! Try Again");
```

```
                    response.sendRedirect("ForgetPassword.jsp");
                }

            } else if (role.equals("Student")) {
                StudentDao studentDao = new StudentDaoImpl();

                StudentData studentRecord = new StudentData(email, password);
                StudentData data = studentDao.changePassword(studentRecord);
                if(data!=null) {
                    session.setAttribute("SuccessMsg", "Password has 
Changed");
                    response.sendRedirect("login.jsp");
                }
                else
                {
                    session.setAttribute("failedMsg", "Password Updation is 
failed!!! Try Again");
                    response.sendRedirect("ForgetPassword.jsp");
                }
            }
        }
}
```

Chapter 7

Testing

7.1 Test Cases:

Test Id	Item to be Tested	Steps	Input	Actual Output	Expected Output	Pass/Fail
1	Roll-Based User Registration (Name , Email Address, Password, PRN No)	User enter their credentials, Clicks on submit.	Name , Email Address, Password, PRN No etc.	User Successfully Registered !	User Successfully Registered !	Pass
2	Roll-Based User Login (System check for proper Email and password entered by User)	System compares the data entered by User and the entered data in database	Email, Password	Login Successful!	Login Successful!	Pass
		If Email and password is valid		Make Connection	Make Connection	Pass
		If Email and password is invalid		Invalid username or password	Invalid username or password	Pass
3	System checks whether details of User are entered as per the	System checks the data entered by User is in valid Form or				

	format	not.				
		If valid format	User entered data	Data Entered in database	Data Entered in database	Pass
		If invalid format	User entered data	“Invalid Data” message will be printed	“Invalid Data” message will be printed	Pass
4	Forget Password	Click on Forget, Enter Credentials, Click On Submit.				
		If Provided Email Matched with the Email stored in database	User Entered Email and new Password.	Password changed successfully!	Password changed successfully!	Pass
		If Provided Email Do Not Matched with the Email stored in database	User Entered Email and new Password.	No Entry Available!	No Entry Available!	Pass
5	CRUD (Create, Retrieve, Update, Delete)	Data entry allowed by view layer, Data accepted and processed	Event Details, User Details.	Data Successfully created, Data Successfully Retrieved, Data Successfull	Data Successfully created, Data Successfully Retrieved, Data Successfull	Pass

		by controller, Operations on data performed by model layer.		y Updated, Data Successfully Deleted.	y Updated, Data Successfully Deleted.	
6	**Session Management**	**User create session object, create logout controller, create httpsession, invalidate user object.**	**HttpSession object, user object.**	**User logout Successfully!**	**User logout Successfully!**	**Pass**

7.1 Type of Testing used

1) Manual Testing:

In the manual test, we used the machine coding tool to write all the codes. After writing each code, we have 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.

2) 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 checked whether the login, registration, add/delete, and update functions are operating normally after clicking the button.

3) API Testing:

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

Chapter 8

Results and Discussions

It would be very useful if the members of the alumni web site which are former student of college could directly contact the alumni officer through the web site. The system that was implemented does not offer this functionality. However, it is easy to find the email address of the alumni officer because it will be placed on the home page of the online community. The contact alumni officer functionality could be easily implemented using asp, which is also used in order to implement the broadcast email functionality that the alumni web site offers. Another useful functionality from which the alumni members could benefit would be if web site had a forum where any discussion could be opened that the related to a person’s field of study.

Many universities around the world have a forum on their alumni website. The forum could also be used to ask some questions. Some people would not like the idea that their information could be seen by everybody that is member of the alumni web site. That is why it would be useful to be able to set some information, such as contact details as private or public. This could be done very easy with the use of radio buttons. Right now, alumni web site offer only inserting details, later it can be modified to update information. The alumni web site is used to maintain data of alumni and to provide platform where alumni can see the progress of an institute and also participate in improving institution condition with the help of donation.

Chapter 9

Conclusion

So the Alumni Information Database is mainly used to share the views between the users of the application which is very useful to upgrade the knowledge of everyone. The application is also serve as a useful site to know what is going on in our in our college and can also know about the various opportunities of the outer world. The application can be further expanded by following the future Enhancements mentioned above.

- Reduces manual work.
- It is time saving.
- Efficient way of managing the Alumni data.
- No need of paperwork as everything is online.

References

https://www.javatpoint.com/java-tutorial

https://www.geeksforgeeks.org/

https://www.tutorialspoint.com/index.html

https://www.iiscalumni.com/