Project Profile

* **Project Profile**

**Project Name:** Campus Placement Recruitment System

**Type of Application:** Web Application

**Project Description:** This Project is aimed at developing an web application for the training and Placement Department of the College. The system is a web application that can be accessed throughout the organization with proper login provide.

**Team Size:3 PEOPLE**

**Front End: HTML,CSS,JAVASCRIPT**

**Back End: DJANGO**

**Database: SQLITE 3**

**Tools used: VISUAL STUDIO CODE**

Introduction to tools

* **Introduction to Tools**
* **Front End Tool:**



* HTML:-

**HTML**stands for Hypertext Markup Language. It is the most basic language, and simple to learn and modify. It is a combination of both hypertext and markup language. It contains the elements that can change/develop a web page’s look and the displayed contents. Or we can say that HTML creates or defines the structure of web pages. We can create websites using HTML which can be viewed on internet-connected devices like laptops, android mobile phones, etc. It was created by Tim Berners-Lee in 1991. The first version of HTML is HTML 2.0 which was published in 1999, and the latest version is HTML 5. We can save HTML files with an extension .html.

HTML, or **HyperText Markup Language** is the standard markup language used to create web pages. It is a combination of Hypertext and Markup language. The Hypertext defines the link between web pages, and Markup defines the text document within tags to structure the web pages.



* CSS:-
* Cascading Style Sheets (CSS) is a style sheet language used for specifying the presentation and styling of a document
* written in a markup language such as HTML or XML (including XML dialects such as SVG, MathML or XHTML).
* CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.
* CSS is designed to enable the separation of content and presentation, including layout,colors, and fonts.
* CSS file, which reduces complexity and repetition in the structural content; and enable the.
* CSS file to be cached to improve the page load speed between the pages that share the file and its formatting
* CSS files can help define font, size, colour, spacing, border and location of HTML information on a web page, and can also be used to create a continuous look throughout multiple pages of a website



* JAVA SCRIPT:-

JavaScript is an open-source programming language designed for creating web-centric applications. It is lightweight and interpreted which makes it much faster than other languages and is integrated with[HTML](https://www.simplilearn.com/html-interview-questions-and-answers-article) making it easier to implement in web applications.

In this Introduction to JavaScript article, you will learn all about JavaScript, the backbone of web development, and understand what exactly this language is and why and how this language is used across various fields.

JavaScript is critical for web development, and if you’ve ever thought about choosing that career path, you’d surely have come across this language. And probably, that’s why you are here in the first place.

* HTML is the structure of your page like the headers, the body text, any images you want to include. It basically defines the contents of a web page.
* CSS controls how that page looks (it’s what you’ll use to customize fonts, background colors, etc.).
* **Back End Tool:**



* Django:-
* Django is a Python framework that makes it easier to create web sites using Python.
* Django takes care of the difficult stuff so that you can concentrate on building your web applications.
* Django emphasizes reusability of components, also referred to as DRY (Don't Repeat Yourself), and comes with ready-to-use features like login system, database connection and CRUD operations (Create Read Update Delete).
* Django is a high-level Python web framework designed for rapid, secure, and scalable web application development. It follows the Model-Template-View (MTV) architecture, allowing developers to focus on writing clean, reusable code. Django comes with powerful features like an admin interface, object-relational mapping (ORM) for database handling, robust security measures, URL routing, and built-in authentication. It's popular for creating complex websites quickly and efficiently, used by major companies like Instagram and Pinterest.
* **Database:-**



* **SQLITE 3:-**

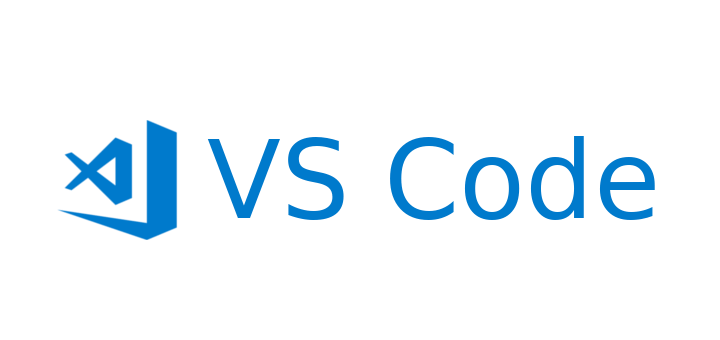
Oracle welcomes and encourages MySQL database users to download and display on their websites any of our Conditional Use Logos below. We take pride in our many enthusiastic users, and this is one of the most important ways to spread our message and software.

You do not need written permission from Oracle to use the MySQL Conditional Use logos on your website. The use is however subject to the [Third Party Usage Guidelines for Oracle Trademarks](http://www.oracle.com/html/3party.html), [Third Party Usage Guidelines for Oracle Logos](http://www.oracle.com/html/3partylogo.html) and the terms specifically applying to use of [MySQL Conditional Use logos](https://www.mysql.com/about/legal/trademark.html).

A written agreement with MySQL is needed before:

* displaying any MySQL logos except the Conditional Use Logos
* displaying any MySQL logos on printed products or elsewhere except on a website

**Tools used:**



* VISUAL STUDIO CODe:-

Studio Code (VS Code) is a source code editor that helps developers build, debug, and run code. It's available for Windows, Linux, and macOS, and can also run in web browsers. Some of its features include:

* Built-in support: VS Code comes with built-in support for JavaScript, TypeScript, and Node.js.
* Extensions: VS Code has a large ecosystem of extensions for other languages and runtimes, such as C++, C#, Java, Python, PHP, and Go.
* Integrated development environment: VS Code is an integrated development environment (IDE) that includes features like syntax highlighting, intelligent code completion, and snippets.
* Cross-platform: VS Code runs on Windows, Linux, and macOS.
* Version control: VS Code has embedded version control with Git.

System Study

* 1. **ExistingSystem:-**
* The existing system for managing the placement process is a manual process that involves a lot of paperwork, time, and effort. The placement department at the college typically manages this process by collecting and maintaining student data manually. This includes information about the student's personal details, academic background, and any other relevant information.
* The manual process is not only time-consuming, but it is also prone to errors and inaccuracies. Due to the volume of data being handled, the manual process increases the chances of errors, leading to inaccuracies and delays in the placement process.
* Moreover, managing the placement process manually makes it challenging to maintain confidentiality, as sensitive information can easily fall into the wrong hands. Also, it becomes difficult to track and monitor the progress of the placement process, resulting in missed opportunities for both the students and potential employers.
  1. **ProposedSystem:-**

To design and implement a web-based placement management system. High-quality placements bring good benefits and positive impacts on students as well as for the colleges. During this process college finds it difficult and time-consuming to collect data from each student. In most cases they collect data manually. Working in a manual system in the colleges requires a lot of manpower and time.

The placement management system is an online application that can be accessed throughout the college and outside with proper login details. This system can be used as an application for the placement officer, Hod, faculty coordinator of the college to manage the student information with regards to placement. The student is able to upload their information. The web application is developed in laravel framework with Model-ViewTemplate (MVT) pattern [5].

The system will be having different types of accounts for different types of users suchas Principal, HOD, placement officers and coordinators, and students. A profile for each student is created with the necessary credentials for the portal. The system uses MySQL for database management and will sort the data of the student based on eligibility criteria demanded by the respective companies.

Also a link will be broadcasted to all eligible candidates via an email for them to choose if they are interested to attend the particular drive or test. The main feature of the system is a machine learning application that can predict placement probabilities of a particular student in different companies based on the previous placement frequency of his/her department, his skill set, marks, and other data available.

* 1. **ScopeoftheProposedSystem:-**

The Student Campus Placement Recruitment System (CPRS) aims to automate the entire placement process for students in a college or university. The system will provide a secure and efficient platform for students to upload and manage their personal and educational information, ensuring fast and easy access to placement procedures and related activities while maintaining the confidentiality of student data.

The primary objective of CPRS is to provide a one-stop solution for managing the entire placement process, from registration to job offers. One of the key features of the project is its one-time registration enabled system, which will help to reduce repetitive tasks and streamline the entire placement process. The system will allow students to register once and then manage their personal and educational information throughout their academic career.

CPRS is a web-based system accessible to students from any device with an internet connection. It features a user-friendly interface for easy navigation. The system automates eligibility checks, streamlining the placement process and saving time. It also automates the entire placement process, from job postings to offers, ensuring efficiency, accuracy, and security. The placement department has a real-time dashboard for updates and tracking student performance. Students can access job listings, monitor application status, and receive career counseling. SIPMS aims to streamline the placement process, automate tasks, and provide students with career guidance.

* 1. **Aim and Objective of Propose System:-**

Create a centralized platform to manage the entire placement process, from job postings to final placement offers.

Maintain an organized database of students’ academic qualifications, resumes, and other relevant details for easy access by the placement cell and recruiters.

Enable students to browse job openings and apply for them directly through the system, making the process more efficient.

Provide recruiters with tools to filter candidates based on qualifications, schedule interviews, and manage recruitment tasks within the platform.

Establish a communication platform where students, placement officers, and recruiters can interact easily, receive notifications, and be informed about updates.

Generate detailed reports on placement metrics, such as the number of students placed, company participation, student performance, andother key statistics.

Provide students with instant updates on job postings, deadlines, interview schedules, and results through notifications and alerts.

* **Feasibility Study:-**

1. **Operational Feasibility:**

The system should be user-friendly and intuitive for students, placement coordinators, and recruiters.

Placement officers and recruiters should be actively involved in defining the system requirements to ensure the system aligns with their needs.

Proper training sessions should be provided to students and staff to familiarize them with the system’s features.

Adequate support should be in place for technical troubleshooting, maintenance, and system updates.

1. **TechnicalFeasibility:-**

Modern technologies such as web development frameworks, databases (MySQL), and cloud services are readily available to build and host the system.

The system should be compatible with the educational institution’s existing IT infrastructure, such as servers, networks, and devices (PCs, smartphones, etc.).

The development team should have the technical expertise required to implement and maintain the system, covering areas like web development, database management, and cybersecurity.

1. **EconomicalFeasibility:-**

The costs of developing the system, including hardware, software, licensing, and developer fees.

Ongoing costs, such as server maintenance, system updates, and support staff salaries.

Increased efficiency in the recruitment process, reduced paperwork, faster placement cycle, improved placement rates, and potential savings on manual processes.

System Analysis

Requirement Specification:-

* **Hardware Specification :-**
* **Processor**: Intel(R) Core(TM) i5-1035G1 CPU @ 1.00GHz 1.19 GHz
* **Installed RAM**: 8.00 GB (7.77 GB usable)
* **Hard Disk**: [Specify type, e.g., SSD or HDD] with a capacity of [Specify capacity, e.g., 256 GB or 512 GB].
* **System Modules :-**
* **Student Module:-**

Students can create and update their profiles, including personal details, academic qualifications, skills, and upload their resumes.Students can search and filter job opportunities posted by recruiters based on criteria such as industry, role, location, and salary.

* **Recruiter Module:-**

View and manage student profiles, approve profiles for placement activities, and track student progress during the recruitment process. recruiters to post job opportunities, set application deadlines, and manage job descriptions.

* **Coordinator Module:-**

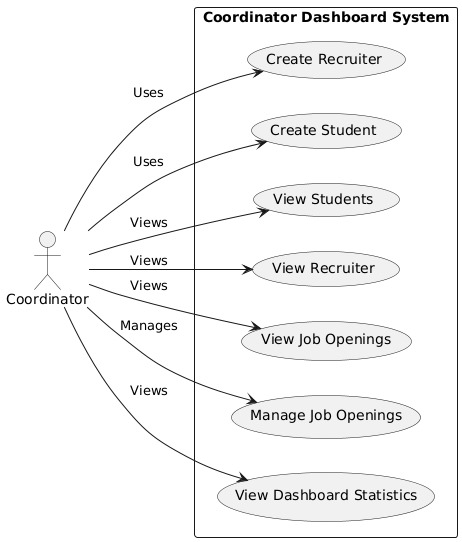
Coordinator can create and manage Recruiter and Student Profiles. and also add and manage job openings.

* **Authentication and Authorization Module:-**

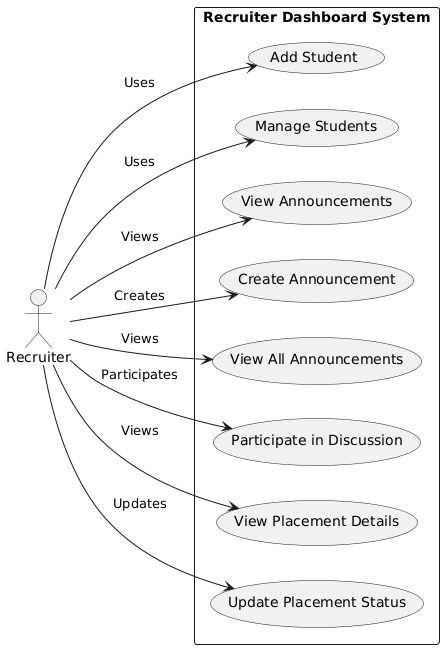
Different roles (student, admin/placement officer, recruiter) have different levels of access to the system’s features and data.Secure login system with options for password recovery and multi-factor authentication for enhanced security.

Use Case Diagram:-

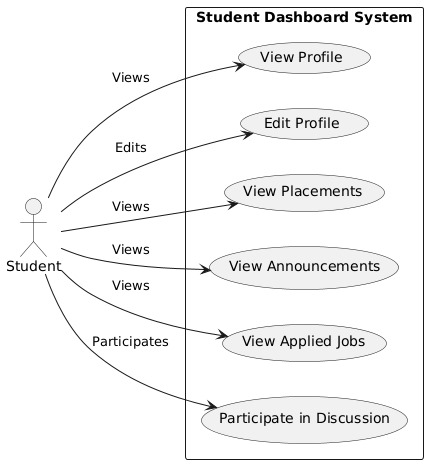
**Coordinator :-**



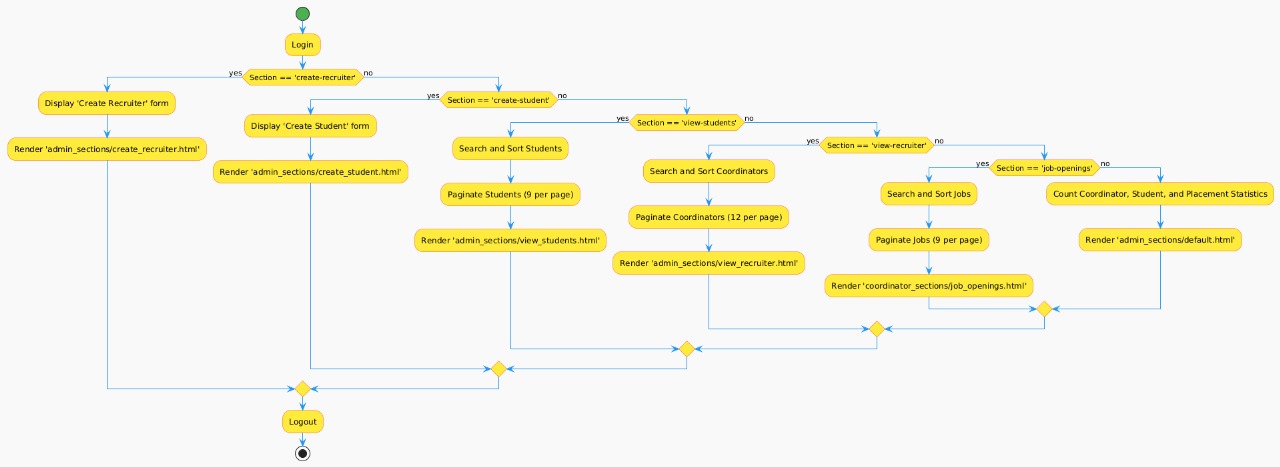
**Recruiter:-**

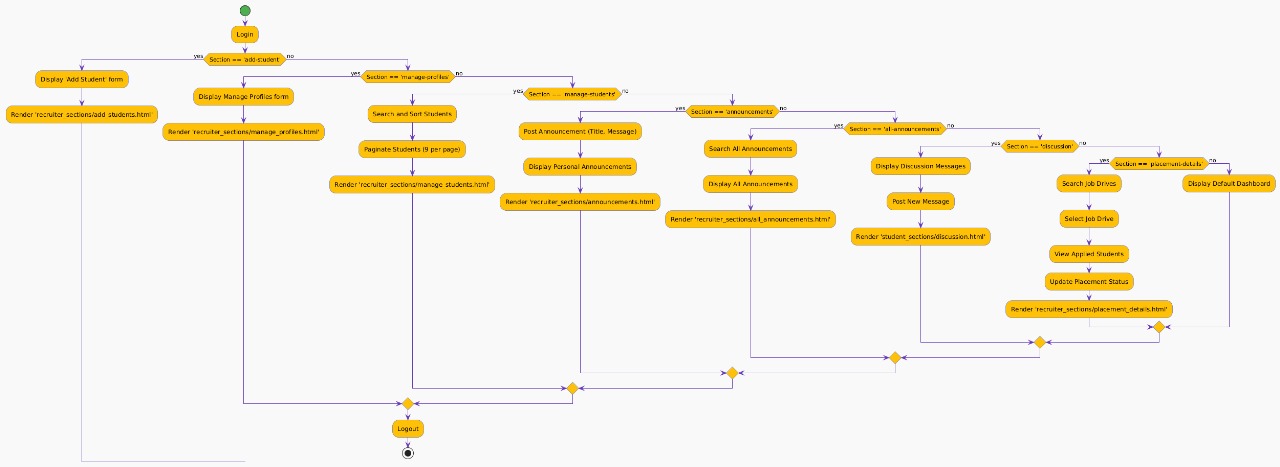


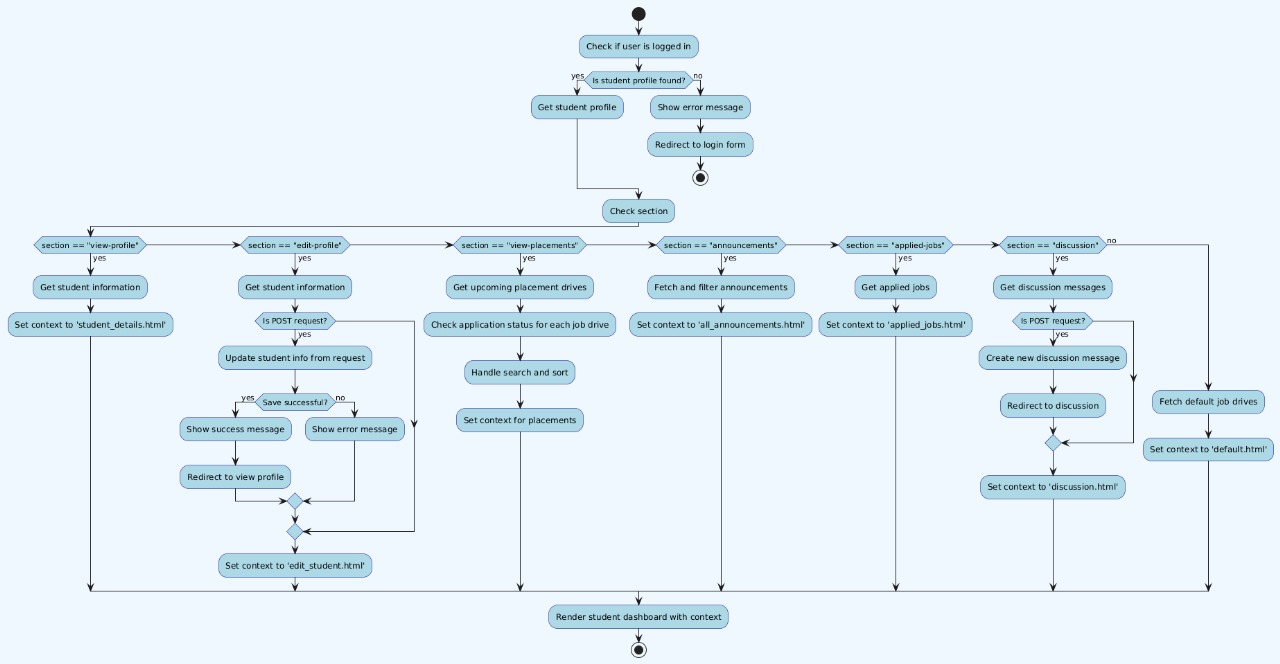
**Student:-**



Activity Diagram:-







System Design

**Data Dictionary:**

**1. CoordinatorProfile**

| **Field** | **Type** | **Description** | **Constraints** |
| --- | --- | --- | --- |
| coordinator\_name | CharField | Name of the coordinator (max 20 characters). | Blank allowed |
| email | EmailField | Unique email address of the coordinator. | Unique, max length 254 |
| department | CharField | Department of the coordinator (max 50 characters). | Null and blank allowed |
| password | CharField | Password for the coordinator (max 100 characters). | Default set to None |
| profile\_photo | ImageField | Profile photo of the coordinator. | Null and blank allowed |
| mobile\_number | CharField | Mobile number of the coordinator (max 15 characters). | Null and blank allowed |
| address | TextField | Address of the coordinator. | Null and blank allowed |
| date\_of\_birth | DateField | Birth date of the coordinator. | Null and blank allowed |
| gender | CharField | Gender of the coordinator (choices: Male, Female, Other). | Null and blank allowed |
| hire\_date | DateField | Date the coordinator was hired. | Null and blank allowed |
| reset\_code | CharField | Reset code for password recovery (max 6 characters). | Null and blank allowed |
| reset\_code\_expires\_at | DateTimeField | Expiration date and time of the reset code. | Null and blank allowed |

**2. StudentProfile**

| **Field** | **Type** | **Description** | **Constraints** |
| --- | --- | --- | --- |
| student\_name | CharField | Name of the student (max 100 characters). | Default set to None |
| email | EmailField | Unique email address of the student. | Unique, null allowed |
| department | CharField | Department of the student (max 50 characters). | Default set to None |
| password | CharField | Password for the student (max 128 characters). | Default set to None |
| reset\_code | CharField | Reset code for password recovery (max 6 characters). | Null and blank allowed |
| reset\_code\_expires\_at | DateTimeField | Expiration date and time of the reset code. | Null and blank Allowed |

**3. StudentInformation**

| **Field** | **Type** | **Description** | **Constraints** |
| --- | --- | --- | --- |
| student\_profile | OneToOneField | One-to-one relationship with StudentProfile. | On delete cascade |
| student\_id | CharField | Unique student ID (max 50 characters). | Unique, null and blank allowed |
| college\_email | EmailField | Unique college email of the student. | Unique |
| full\_name | CharField | Full name of the student (max 200 characters). | Required |
| personal\_email | EmailField | Personal email address of the student. | Null and blank allowed |
| phone\_number | CharField | Phone number of the student (max 15 characters). | Null and blank allowed |
| date\_of\_birth | DateField | Birth date of the student. | Null and blank allowed |
| gender | CharField | Gender of the student (choices: Male, Female, Other). | Null and blank allowed |
| profile\_photo | ImageField | Profile photo of the student. | Null and blank allowed |
| permanent\_address | TextField | Permanent address of the student. | Null and blank allowed |
| current\_address | TextField | Current address of the student. | Null and blank allowed |
| department | CharField | Department of the student (max 100 characters). | Required |
| degree\_program | CharField | Degree program of the student (max 100 characters). | Null and blank allowed |
| year\_of\_enrollment | IntegerField | Year the student enrolled. | Null and blank allowed |
| year\_of\_graduation | IntegerField | Year the student is expected to graduate. | Null and blank allowed |
| tenth\_percentage | FloatField | Percentage in 10th standard. | Null and blank allowed |
| tenth\_passing\_year | IntegerField | Passing year of 10th standard. | Null and blank allowed |
| tenth\_school | CharField | School attended for 10th (max 200 characters). | Null and blank allowed |
| twelfth\_stream | CharField | Stream in 12th standard (choices: Science, Commerce, Arts, Other). | Null and blank allowed |
| twelfth\_percentage | FloatField | Percentage in 12th standard. | Null and blank allowed |
| twelfth\_passing\_year | IntegerField | Passing year of 12th standard. | Null and blank allowed |
| twelfth\_school | CharField | School attended for 12th (max 200 characters). | Null and blank allowed |
| ug\_course | CharField | Undergrad course (max 100 characters). | Null and blank allowed |
| ug\_passing\_year | IntegerField | Passing year of undergrad. | Null and blank allowed |
| ug\_cgpa | FloatField | CGPA in undergrad. | Null and blank allowed |
| ug\_college\_university | CharField | College/University attended for undergrad (max 200 characters). | Null and blank allowed |
| other\_course | CharField | Other courses (max 100 characters). | Null and blank allowed |
| other\_passing\_year | IntegerField | Passing year of other courses. | Null and blank allowed |
| other\_cgpa | FloatField | CGPA in other courses. | Null and blank allowed |
| other\_college\_university | CharField | College/University attended for other courses (max 200 characters). | Null and blank allowed |
| backlogs | IntegerField | Number of backlogs. | Default set to 0 |
| skills | TextField | Skills of the student. | Null and blank allowed |
| certifications | TextField | Certifications earned by the student. | Null and blank allowed |
| internship\_details | TextField | Details of internships undertaken. | Null and blank allowed |
| projects | TextField | Details of projects completed. | Null and blank allowed |
| achievements | TextField | Achievements of the student. | Null and blank allowed |
| extracurricular\_activities | TextField | Extracurricular activities participated in. | Null and blank allowed |
| resume | FileField | Student's resume file. | Null and blank allowed |
| linkedin\_profile | URLField | URL to the student's LinkedIn profile. | Null and blank allowed |
| github\_profile | URLField | URL to the student's GitHub profile. | Null and blank allowed |
| placement\_preferences | TextField | Job placement preferences of the student. | Null and blank allowed |
| placed\_status | BooleanField | Status indicating if the student is placed. | Default set to False |
| job\_offers | TextField | Details of job offers received. | Null and blank allowed |
| date\_of\_last\_update | DateTimeField | Last update date and time of student information. | Default set to timezone.now |
| nationality | CharField | Nationality of the student (max 50 characters). | Null and blank allowed |

**4. JobDrive**

| **Field** | **Type** | **Description** | **Constraints** |
| --- | --- | --- | --- |
| title | CharField | Title of the job drive (max 255 characters). | Required |
| company | CharField | Company name (max 255 characters). | Required |
| location | CharField | Job location (max 255 characters). | Required |
| job\_description | TextField | Description of the job. | Required |
| skills\_required | CharField | Required skills for the job (max 255 characters). | Required |
| salary | CharField | Salary offered for the job (max 50 characters). | Required |
| vacancies | IntegerField | Number of vacancies available. | Default set to 0 |
| last\_date\_to\_apply | DateField | Last date to apply for the job. | Required |
| tentative\_drive\_date | DateField | Tentative date for the job drive. | Null and blank allowed |
| created\_at | DateTimeField | Creation date and time of the job drive. | Auto now add |

**5. Announcement**

| **Field** | **Type** | **Description** | **Constraints** |
| --- | --- | --- | --- |
| title | CharField | Title of the announcement (max 255 characters). | Required |
| message | TextField | Announcement message. | Required |
| created\_at | DateTimeField | Creation date and time of the announcement. | Default set to timezone.now |
| posted\_by | ForeignKey | Coordinator who posted the announcement. | On delete cascade |

**6. AppliedJob**

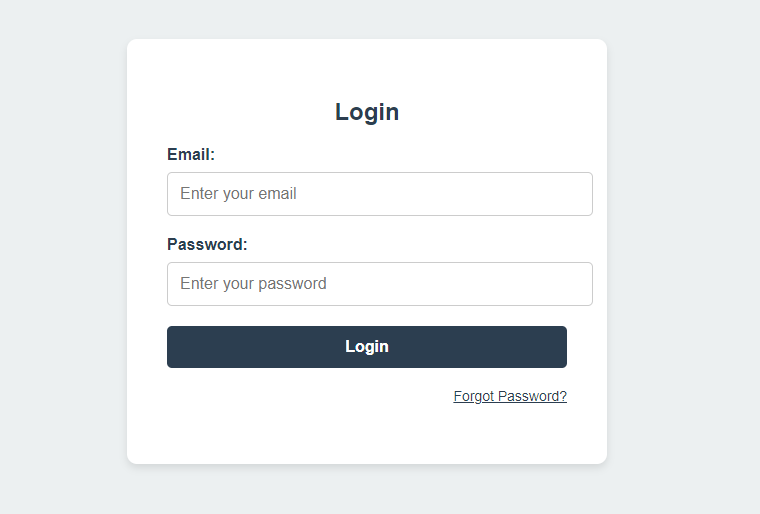
| **Field** | **Type** | **Description** | **Constraints** |
| --- | --- | --- | --- |
| student | ForeignKey | Student applying for the job. | On delete cascade |
| job\_drive | ForeignKey | Job drive the student applied to. | On delete cascade |
| applied\_at | DateTimeField | Date and time when the student applied. | Auto now add |
| status | CharField | Current status of the application (choices: Applied, Shortlisted, Selected, Rejected). | Default set to 'applied' |

**7. DiscussionMessage**

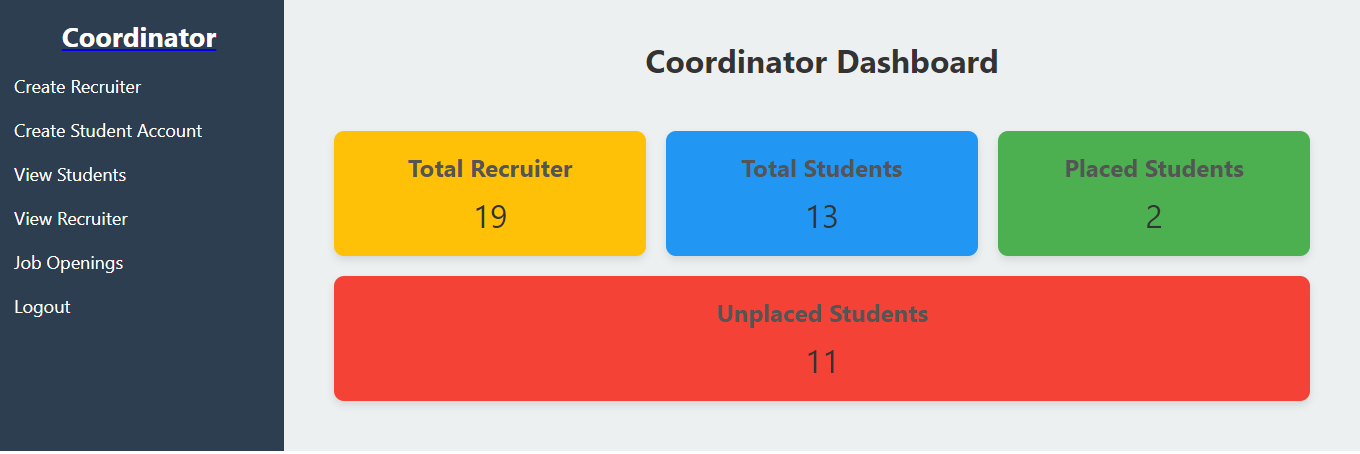
| **Field** | **Type** | **Description** | **Constraints** |
| --- | --- | --- | --- |
| username | CharField | Username of the person posting the message (max 100 characters). | Required |
| content | TextField | Content of the message. | Required |
| timestamp | DateTimeField | Date and time when the message was posted. | Auto now add |

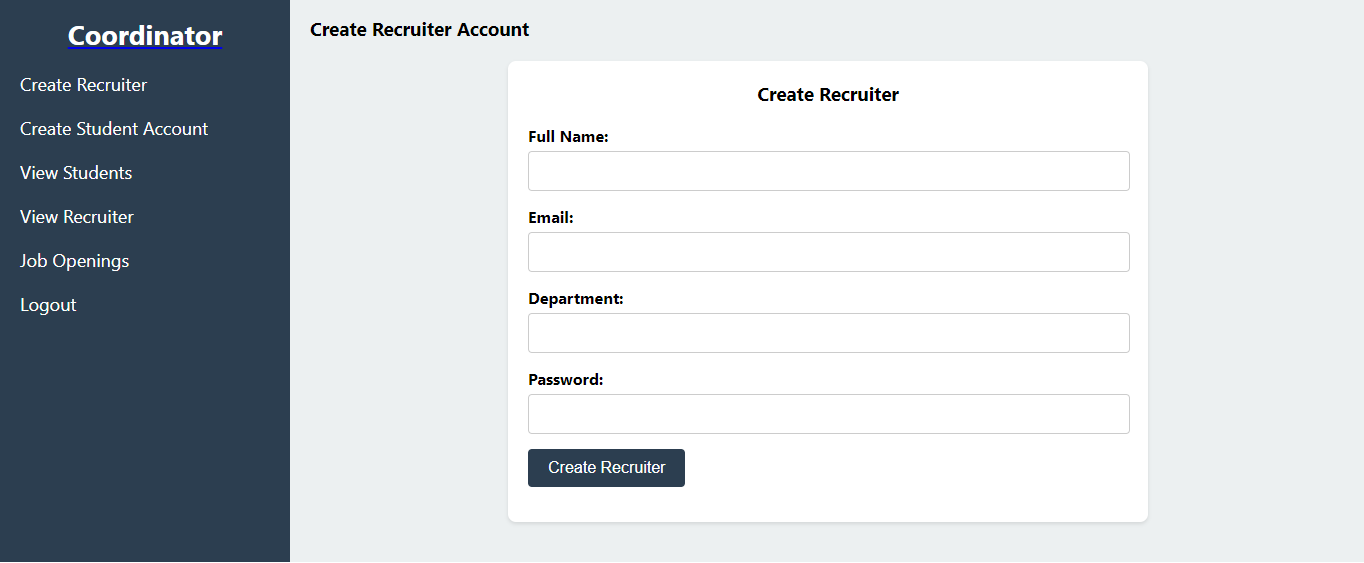
Screen Layout:

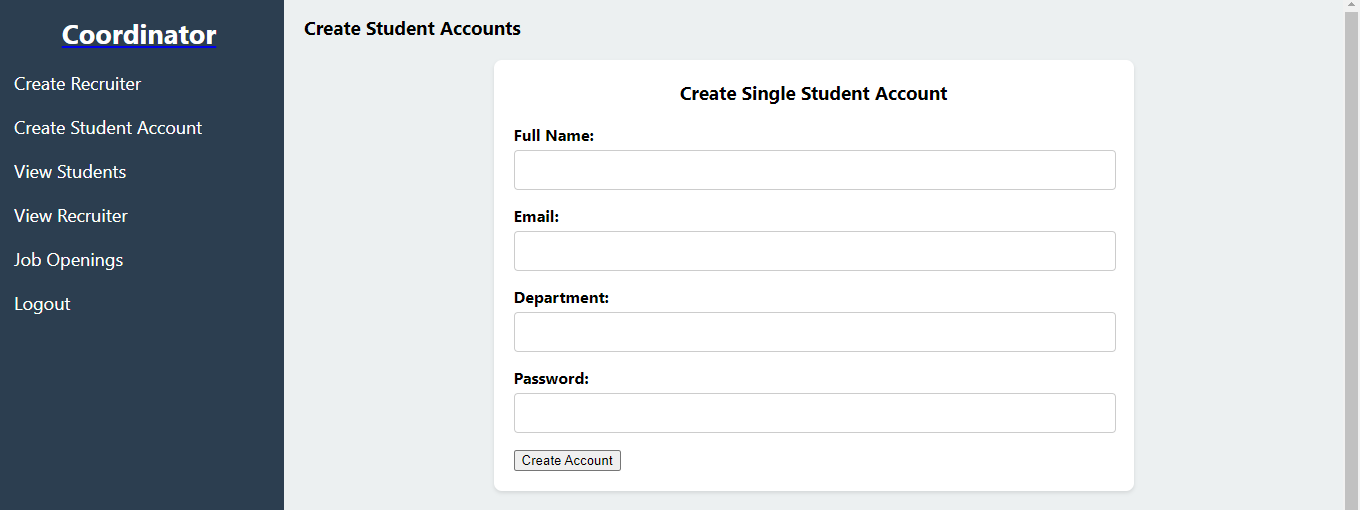
1.Login Form:

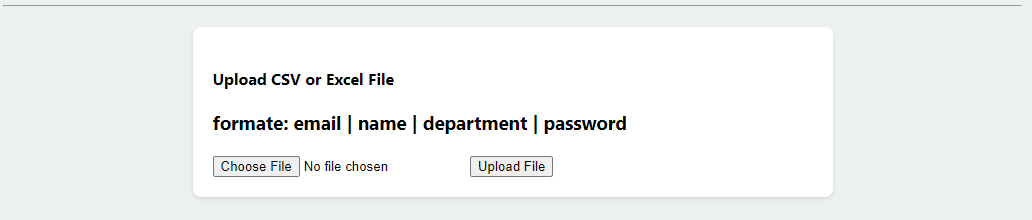


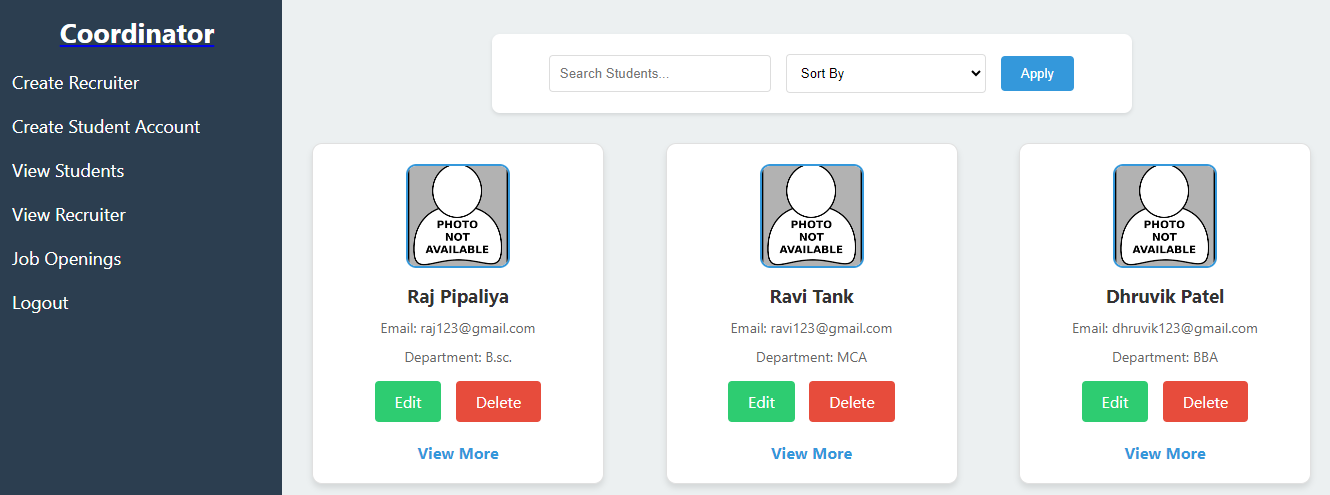
2.Coordinator Dashboard:

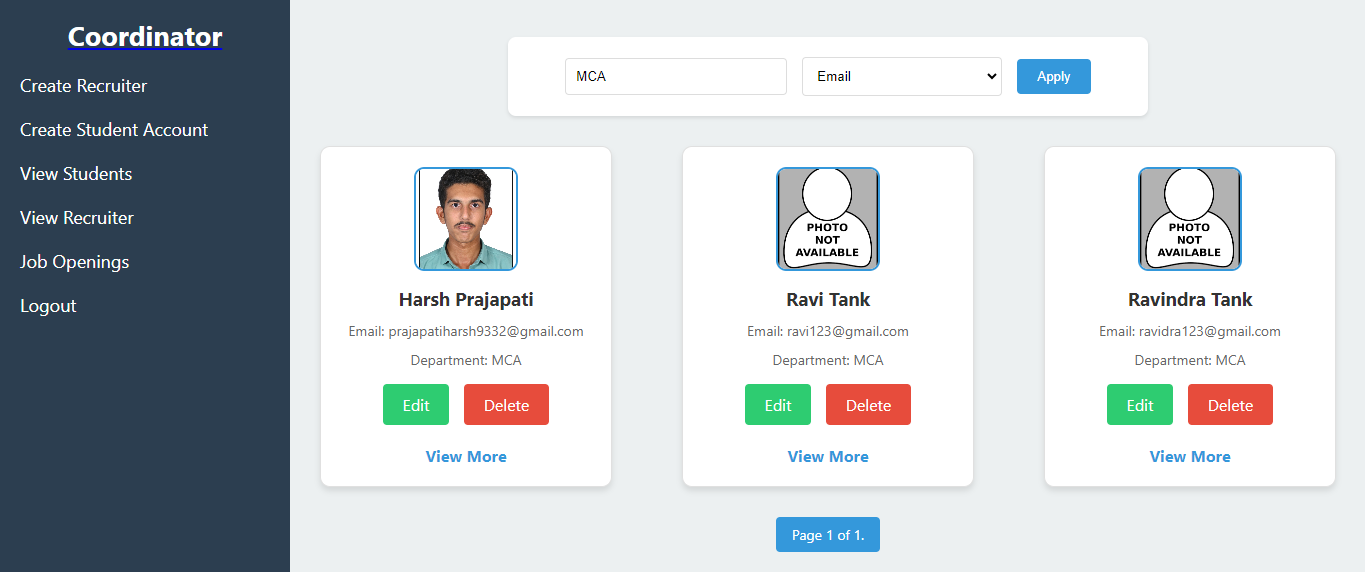


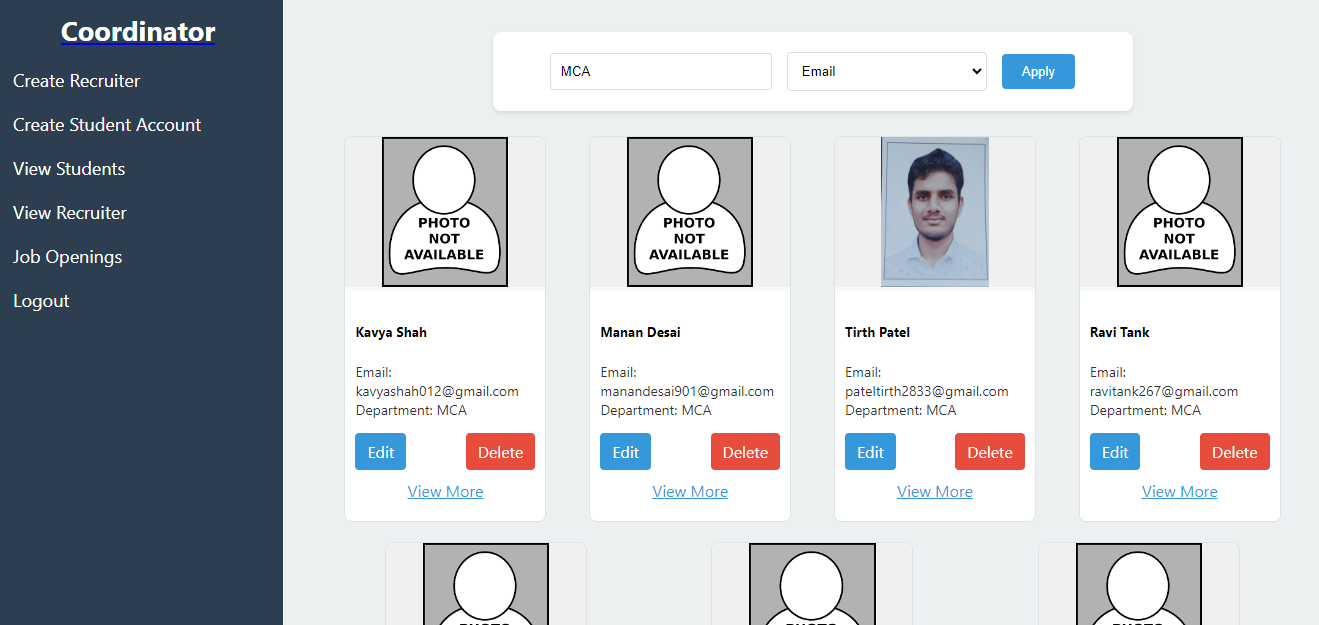
3.Create Recruiter:  


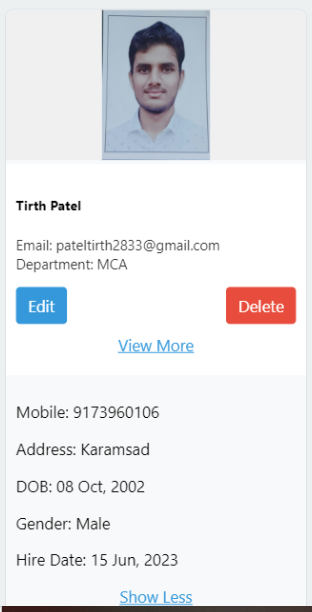
4.Create Student Account:  


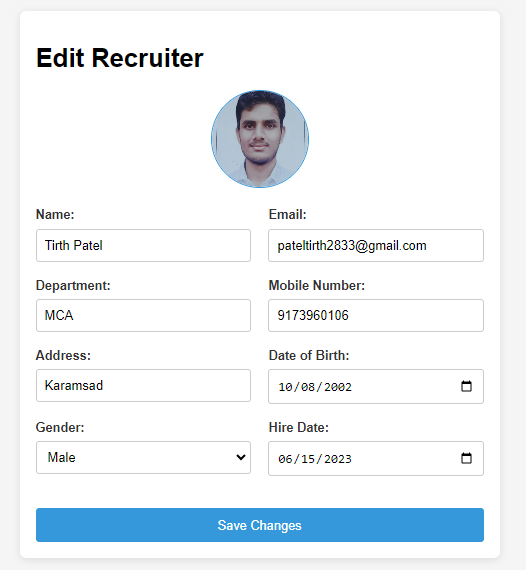


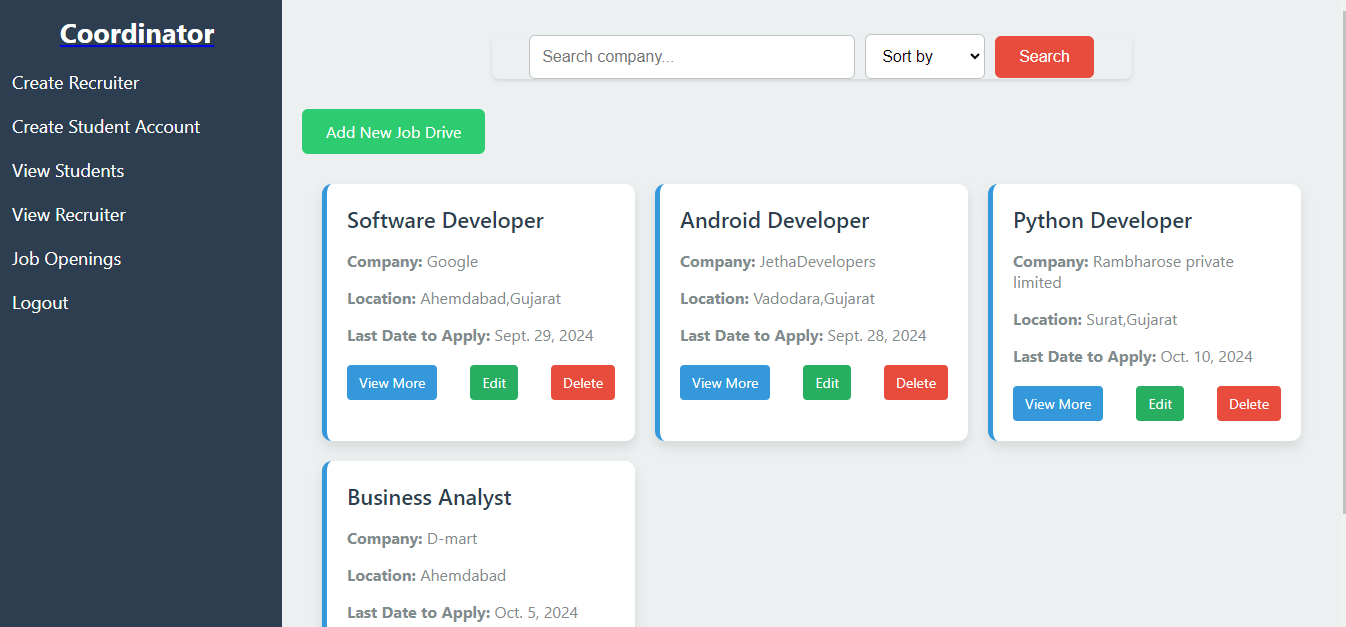
5.view student:  


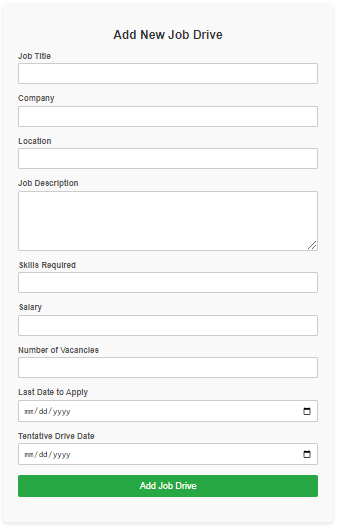


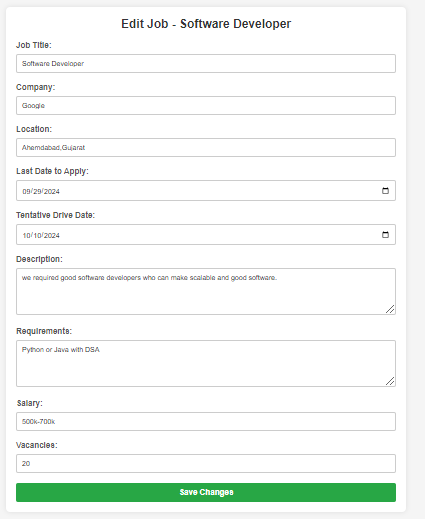
6. Recruiter view:

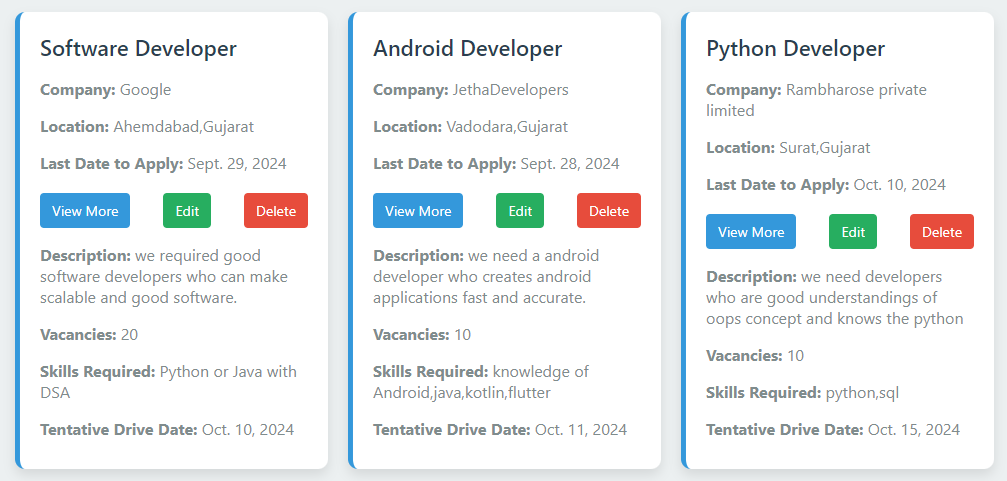
7.view more:

8.Edit Recruiter:  
 

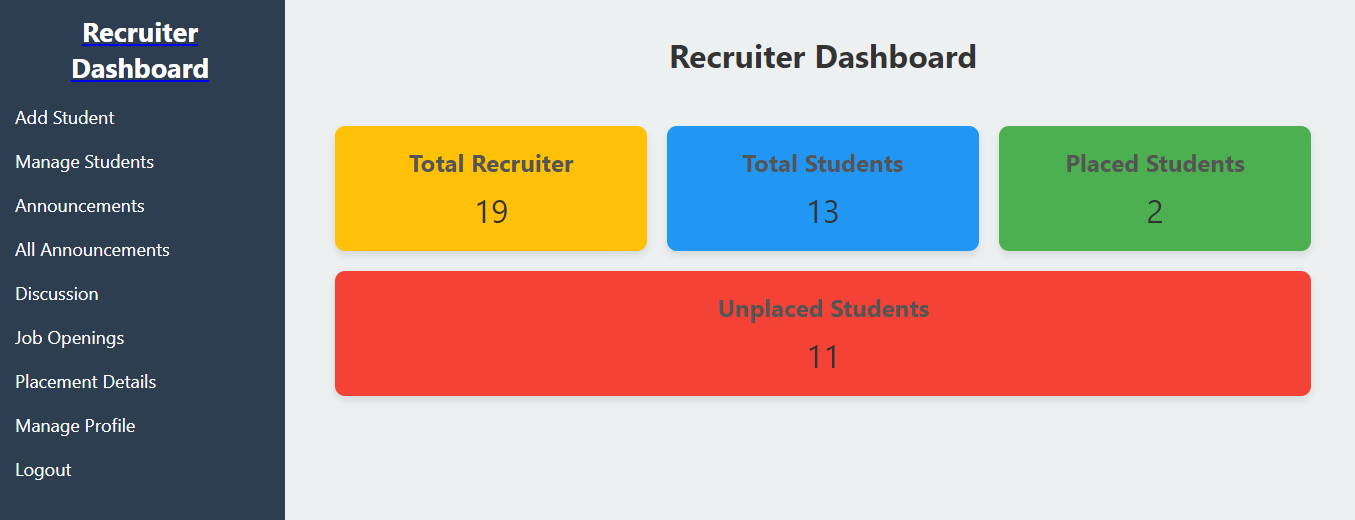
9.Job Openings:  


10.Add new job:

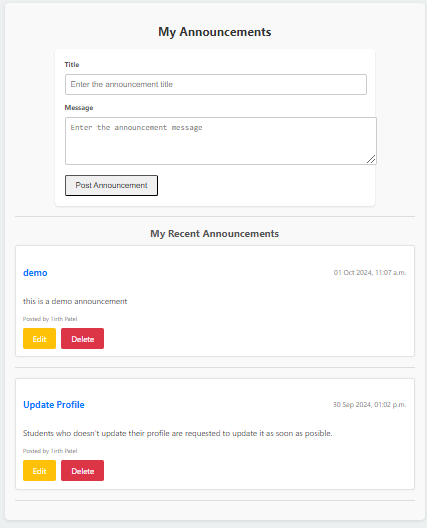
11. Edit job

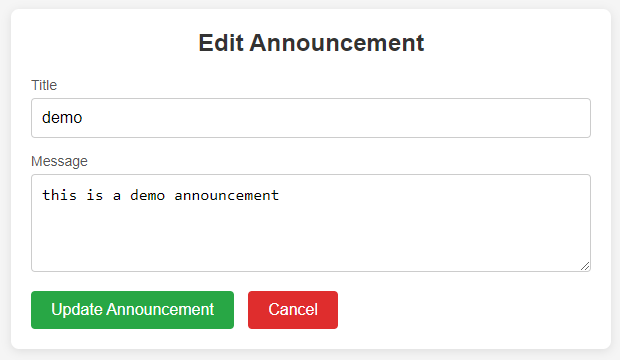
12.view more:  


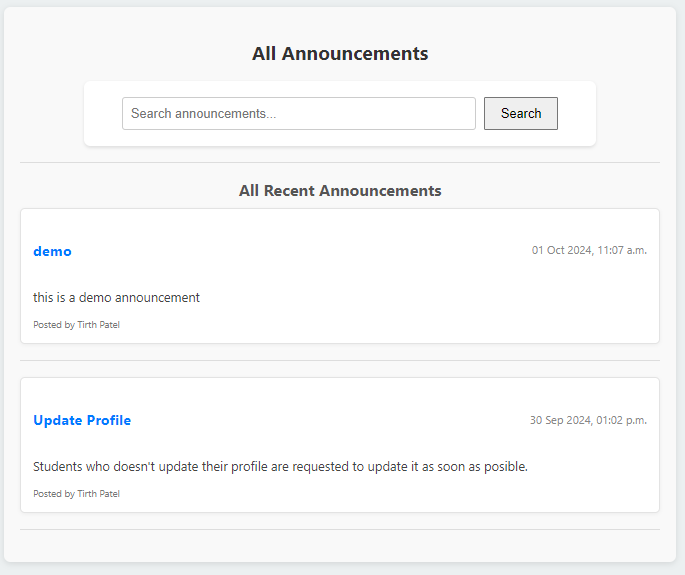
13.Recruiter Dashboard:



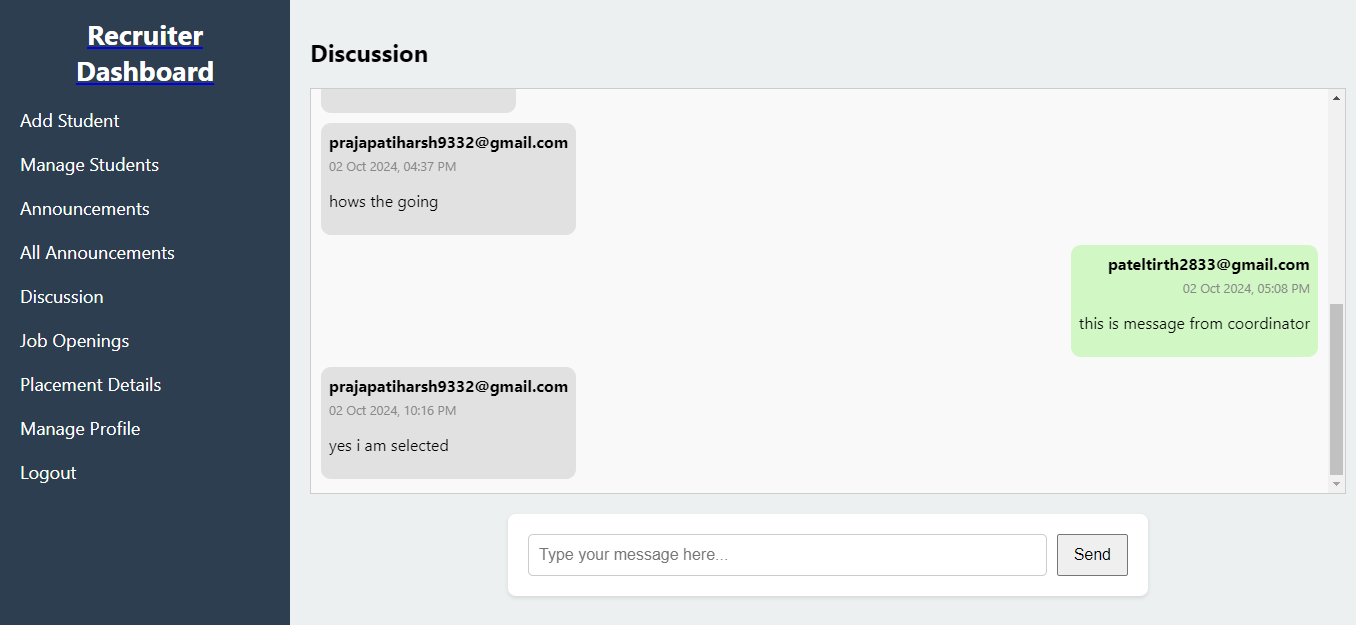
14.Announcements:



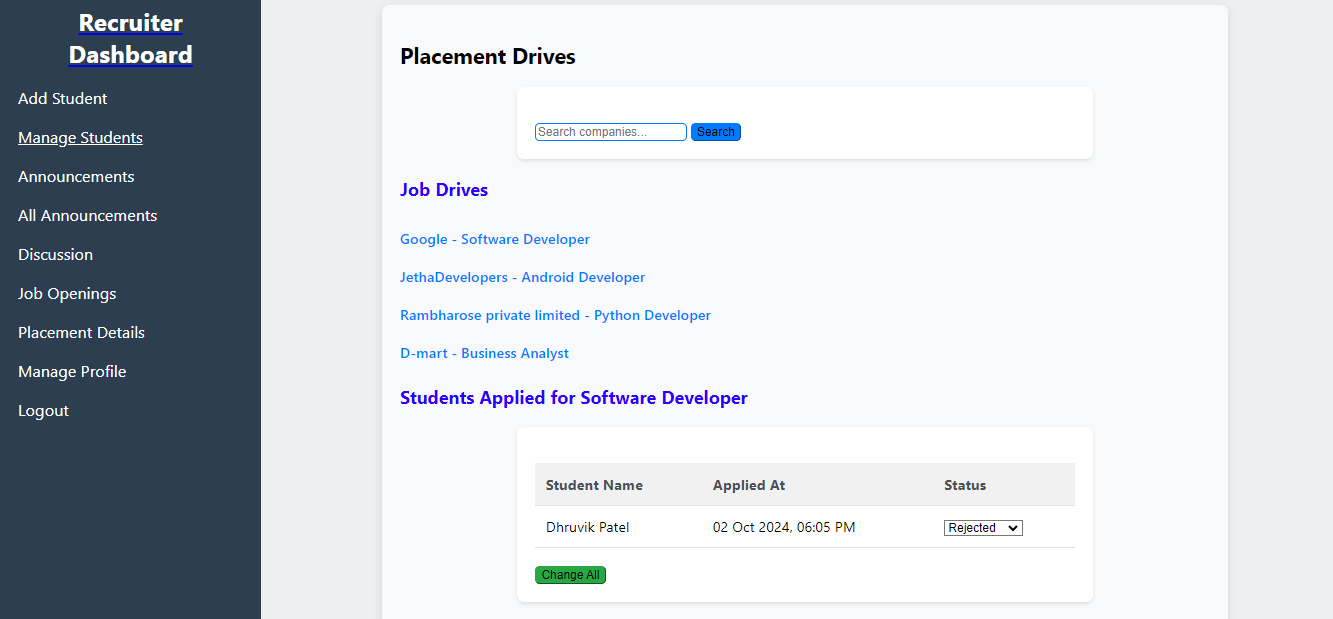
15.Edit Announcement:  


16. All Announcement:

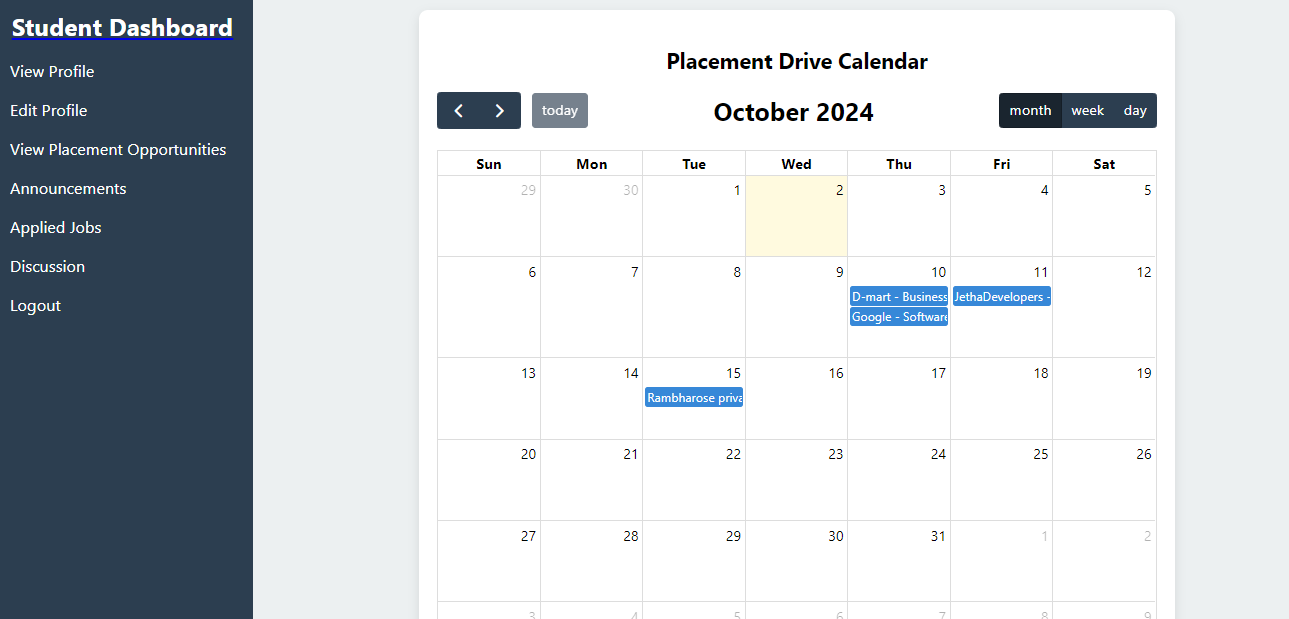
17.Discussion:

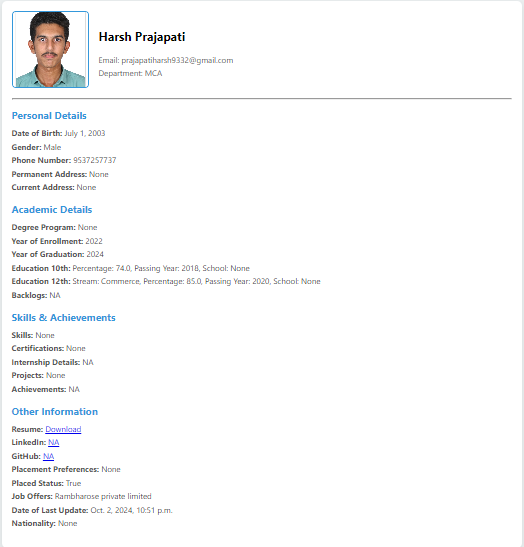


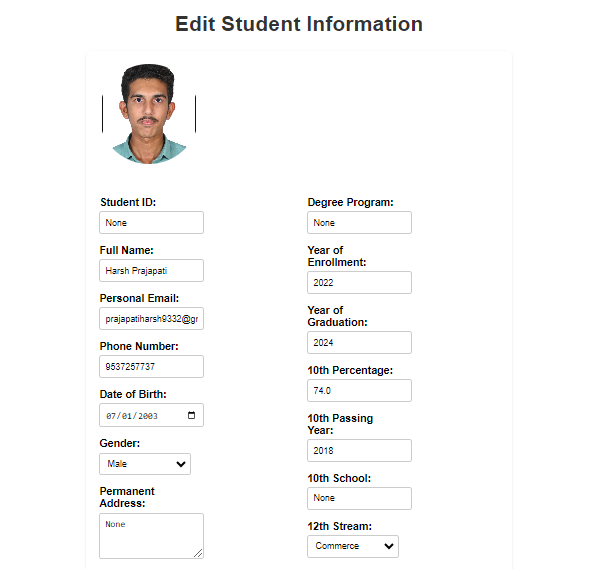
18.Placement Drive:

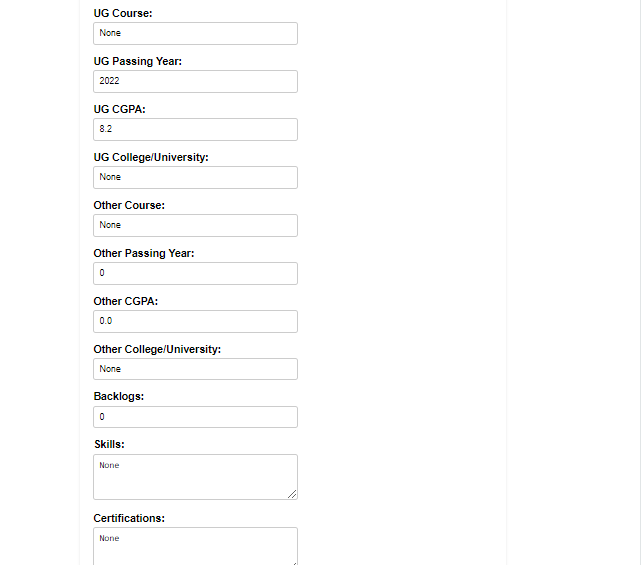


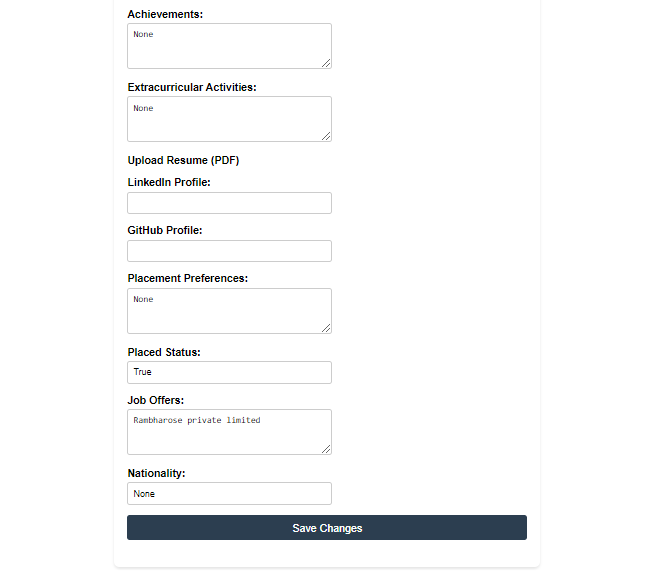
19.Student Dashboard:

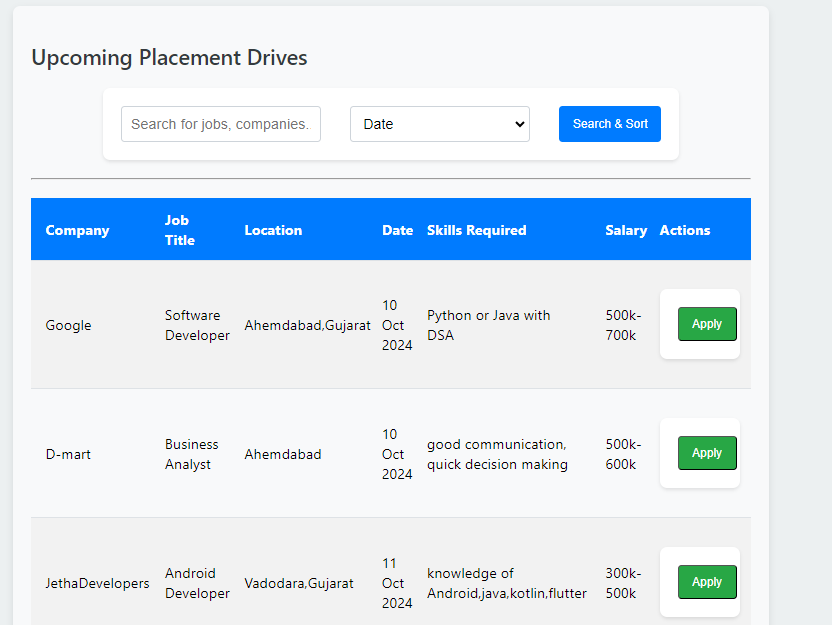


20.view profile:  


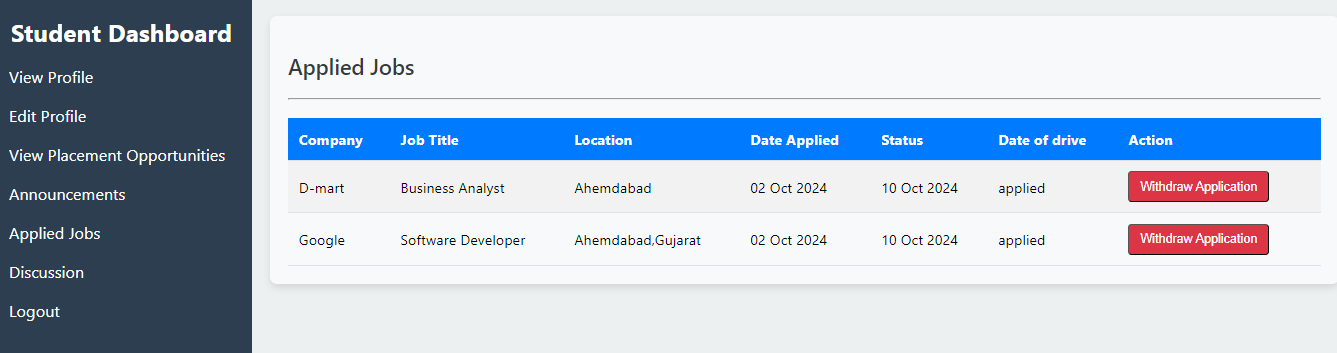
21.edit profile:



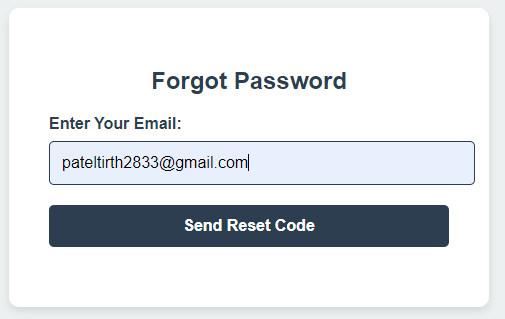


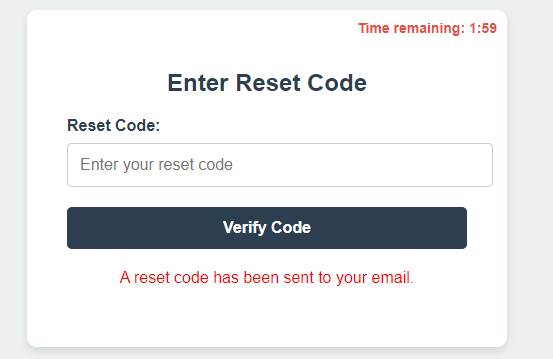
22.view placement opportunities:  


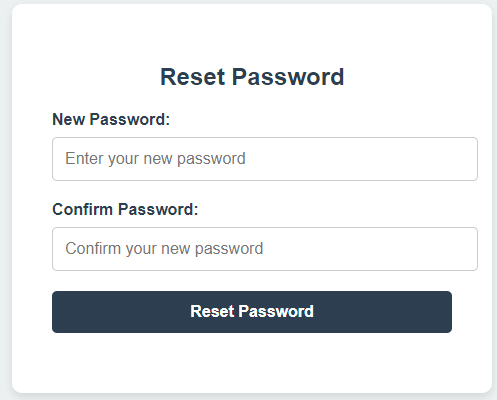
23.applied jobs:



24.Forgot Password:







System Testing

### 1. **Coordinator Profile Testing:-**

* **Test Case 1: Creating a new coordinator profile.**
  + Precondition: Access to coordinator creation form.
  + Steps: Submit the form with valid data for all fields.
  + Expected Result: Coordinator profile should be saved and appear in the list of coordinators.
* **Test Case 2: Updating coordinator profile details.**
  + Precondition: Coordinator profile already exists.
  + Steps: Update fields such as email, password, or profile photo.
  + Expected Result: Updated coordinator profile should reflect changes in the system.
* **Test Case 3: Password reset functionality.**
  + Precondition: Coordinator wants to reset the password.
  + Steps: Trigger password reset functionality (generate and verify reset code).
  + Expected Result: Reset code should be generated, and the new password should work upon reset.

### 2. **Student Profile and Information Testing:-**

* **Test Case 1: Creating a student profile.**
  + Precondition: Valid data for a new student.
  + Steps: Create a student profile and associated student information.
  + Expected Result: Student profile and information should be saved and accessible in the database.
* **Test Case 2: Editing student information (including educational details, contact, etc.).**
  + Precondition: Student profile exists.
  + Steps: Edit fields such as phone number, address, and education details.
  + Expected Result: The changes should be reflected and saved in the database.
* **Test Case 3: Handling file uploads (profile photos and resumes).**
  + Precondition: Student profile exists.
  + Steps: Upload profile photo and resume in the form.
  + Expected Result: Uploaded files should be saved and linked to the student profile.

### 3. **Job Drive Testing:-**

* **Test Case 1: Creating a job drive.**
  + Precondition: Job drive creation form.
  + Steps: Submit a new job drive with valid details such as title, skills required, and last date to apply.
  + Expected Result: Job drive should be listed and available for students to apply.
* **Test Case 2: Displaying a list of job drives.**
  + Precondition: Job drives already exist in the system.
  + Steps: Access the job drive listing page.
  + Expected Result: All job drives should be listed, and clicking on a drive should display its details.
* **Test Case 3: Searching for job drives.**
  + Precondition: Job drives exist with various company names.
  + Steps: Use the search feature to filter job drives by company or title.
  + Expected Result: Job drives that match the search criteria should be displayed.

### 4. **Job Application Testing (AppliedJob):-**

* **Test Case 1: Applying for a job drive.**
  + Precondition: Student is logged in, and job drives are available.
  + Steps: Apply for a job drive.
  + Expected Result: Job application status should be "applied," and the application should appear in the list of applied jobs.
* **Test Case 2: Updating job application status.**
  + Precondition: Student has applied for a job.
  + Steps: Change the application status to "shortlisted," "selected," or "rejected."
  + Expected Result: The application status should be updated correctly in the system.

### 5. **Announcement Testing:-**

* **Test Case 1: Creating a new announcement.**
  + Precondition: Coordinator is logged in.
  + Steps: Create a new announcement.
  + Expected Result: Announcement should be visible to all relevant users (students or coordinators).
* **Test Case 2: Displaying announcements.**
  + Precondition: Announcements exist in the system.
  + Steps: Access the announcement listing page.
  + Expected Result: All announcements should be displayed, with the latest at the top.

### 6. **Discussion Message Testing:-**

* **Test Case 1: Adding a new discussion message.**
  + Precondition: Access to the discussion board.
  + Steps: Submit a new message to the discussion.
  + Expected Result: The message should appear on the board with the correct timestamp and username.
* **Test Case 2: Displaying discussion messages.**
  + Precondition: Discussion messages already exist.
  + Steps: View the discussion board.
  + Expected Result: All messages should be listed in chronological order.

### 7. **General System Tests:-**

* **Test Case 1: User authentication (login/logout).**
  + Precondition: Valid coordinator and student profiles exist.
  + Steps: Perform login and logout actions.
  + Expected Result: Users should be able to log in and out with valid credentials.
* **Test Case 2: Handling validation errors (empty fields, invalid data).**
  + Precondition: Forms are filled out incorrectly.
  + Steps: Submit forms with missing required fields or invalid data (e.g., invalid email).
  + Expected Result: Validation errors should be displayed, and data should not be saved.
* **Test Case 3: Data integrity and uniqueness checks (e.g., unique email for students).**
  + Precondition: Attempt to create multiple students with the same email or job drives with the same title.
  + Steps: Submit duplicate entries.
  + Expected Result: The system should reject duplicate entries with appropriate error messages.

### 8. **UI/UX Testing:-**

* **Test Case 1: Responsiveness of the web pages.**
  + Precondition: Access the website from various devices (desktop, mobile, tablet).
  + Steps: Navigate through the website and interact with forms and data.
  + Expected Result: The site should adjust correctly for different screen sizes without UI glitches.
* **Test Case 2: Cross-browser compatibility.**
  + Precondition: Different browsers (Chrome, Firefox, Edge, etc.).
  + Steps: Access the site from different browsers.
  + Expected Result: The design and functionality should remain consistent across all browsers.

Future enhancement

**1. AI-Powered Candidate Matching**

* **AI Algorithms**: Implement AI algorithms to match students with job opportunities based on skills, academic performance, interests, and extracurricular activities.
* **Smart Recommendations**: Provide smart job recommendations to students and help recruiters identify the best-fit candidates.

**2. Automated Interview Scheduling**

* **AI Scheduling**: Automate the interview scheduling process by considering both recruiter and candidate availability, reducing human intervention and minimizing conflicts.
* **Notifications and Reminders**: Integrated notifications and reminders via email or SMS to remind candidates and recruiters about upcoming interviews or important deadlines.

**3. Mobile Application**

* **Mobile-Friendly Interface**: Create a mobile application that allows students and recruiters to access the system on the go, track recruitment activities, and receive push notifications.
* **On-the-Go Updates**: Students can get real-time updates on job postings, placement drives, and interview results directly on their smartphones.

**4. Social Media Integration**

* **LinkedIn Integration**: Allow students to directly link their LinkedIn profiles to the system to showcase recommendations, project work, and professional connections.
* **Employer Branding**: Enable companies to use the platform to share posts, videos, and news, making it easier for students to understand company culture and expectations.

bibliography/

references

Bibliography:-

The **Campus Placement Recruitment System** project references The Definitive Guide to Django by Pollock & Holovaty (2009) and academic articles like Reddy’s "A Comparative Study on Campus Placement Systems" (2019). Key resources include Django Documentation (2023), MDN Web Docs (2023), and tutorials like Traversy Media’s Django Crash Course (2022).

References:-

* **For Python :-**

<https://docs.python.org/3/>

<https://www.w3schools.com/python/python>

* **For Django :-**

<https://docs.djangoproject.com/en/5.1/>

<https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django>

<https://www.geeksforgeeks.org/django-tutorial/>

* **HTML ,CSS ,JavaScript ,Bootstrap.**

<https://getbootstrap.com/>

<https://www.w3schools.com/css/default.asp>

<https://www.w3schools.com/js/default.asp>

<https://www.w3schools.com/html/default.asp>