



IT644 WSSOA - System Design Document

Project: Placemate - Campus Recruitment Management System

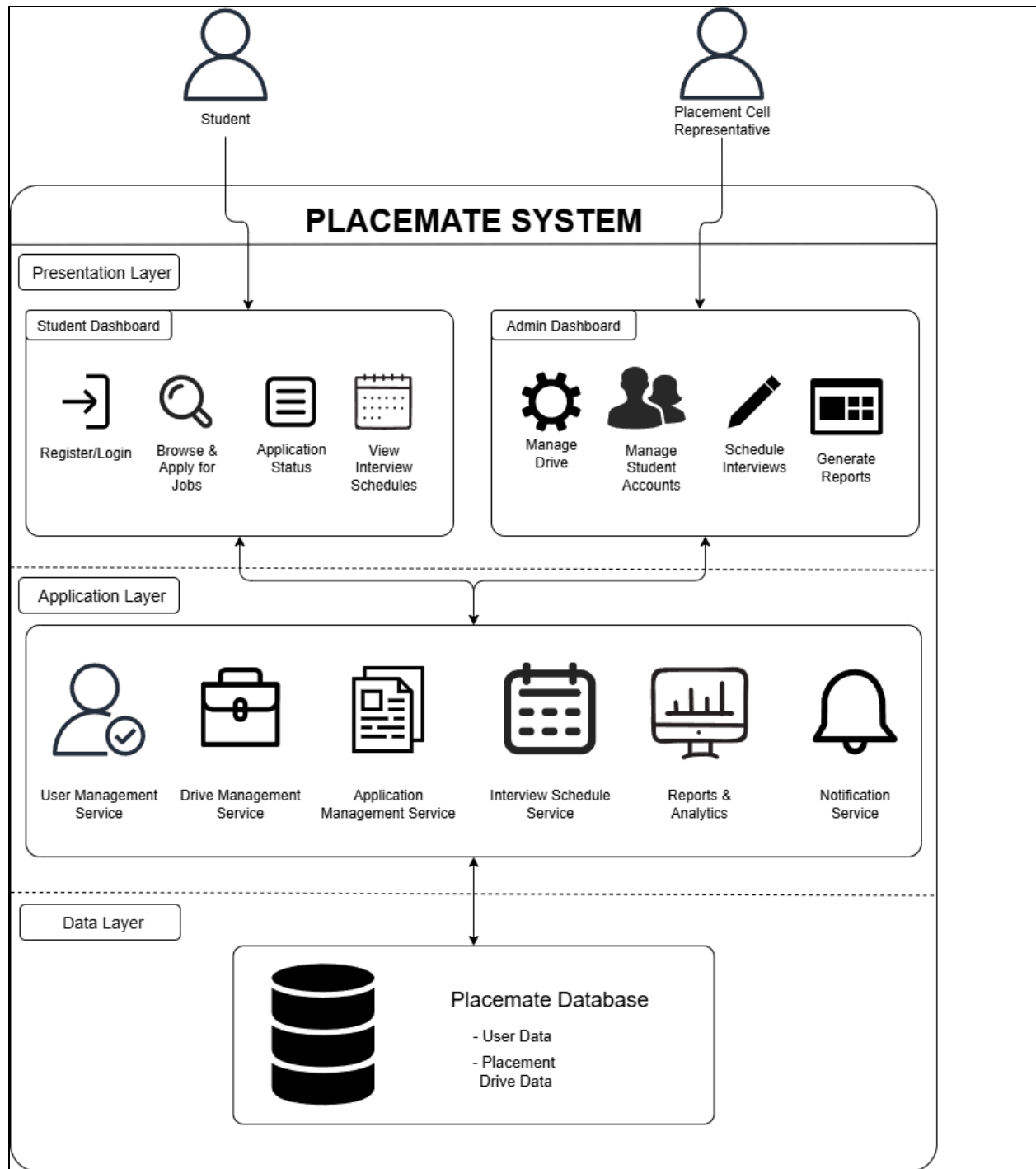
Project Name:	Placemate
Version:	1.1
Date:	September 14, 2025
Group:	9
Programme:	MSc IT

Table of Contents

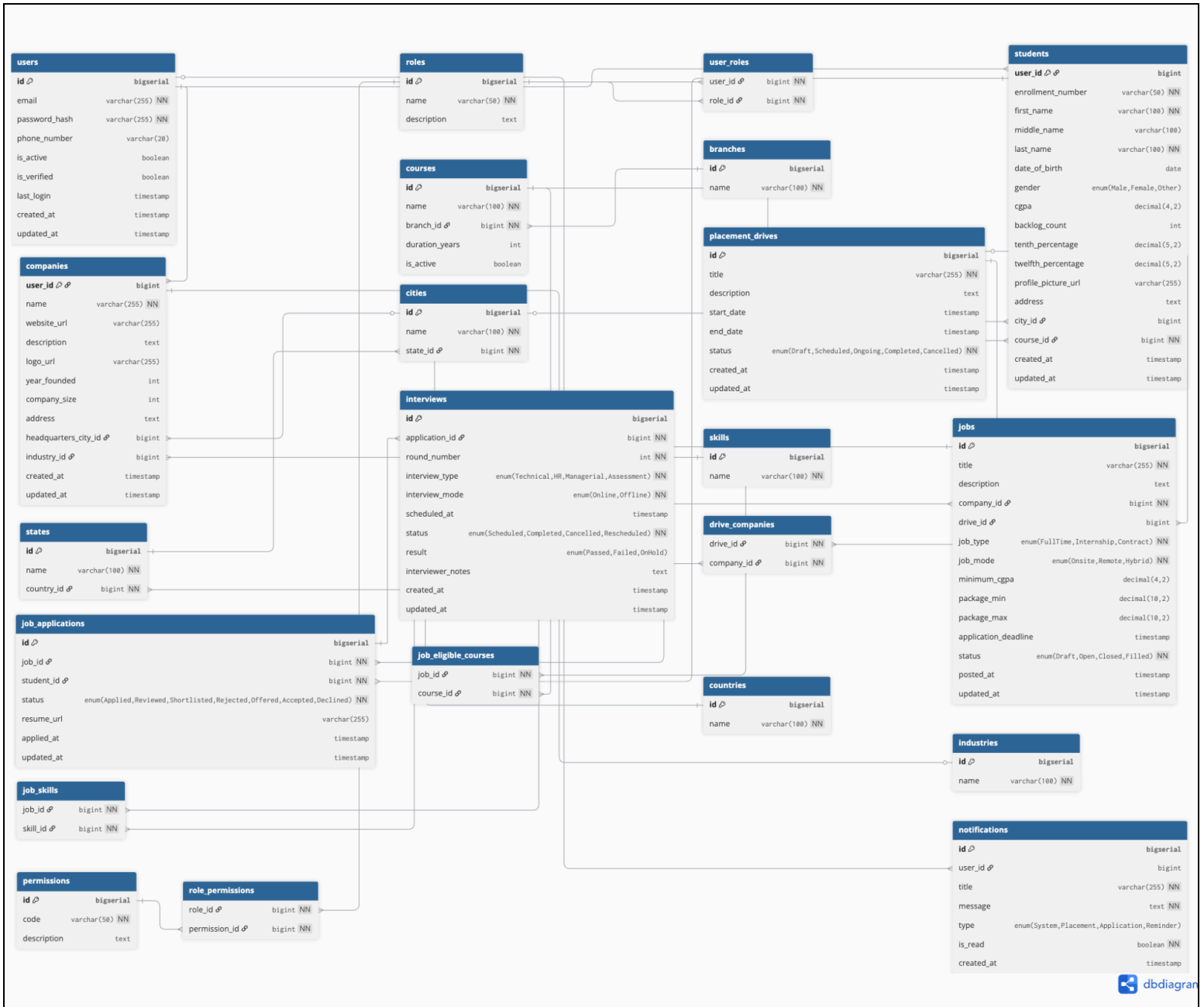
IT644 WSSOA - System Design Document.....	1
Project: Placemate - Campus Recruitment Management System.....	1
Table of Contents.....	2
1. System Architecture.....	2
1.1. Architecture Diagram.....	3
1.2. ER Diagram.....	4
1.3. Class Diagram.....	5
2. Placemate REST API Documentation.....	5
2.1. Introduction.....	6
2.2. Authentication.....	6
2.3. API Reference.....	6
2.3.1. Authentication & Authorization.....	7
2.3.2. Students.....	9
2.3.3. Companies & Placement Drives.....	10
2.4. Error Handling.....	12
2.5. Example Workflow: A Student Applies for a Job.....	13
3. Wireframes / UI UX Mockups.....	15
3.1. General.....	15
3.2. Admin Role Access.....	16
3.3. Student Role Access.....	29

1. System Architecture

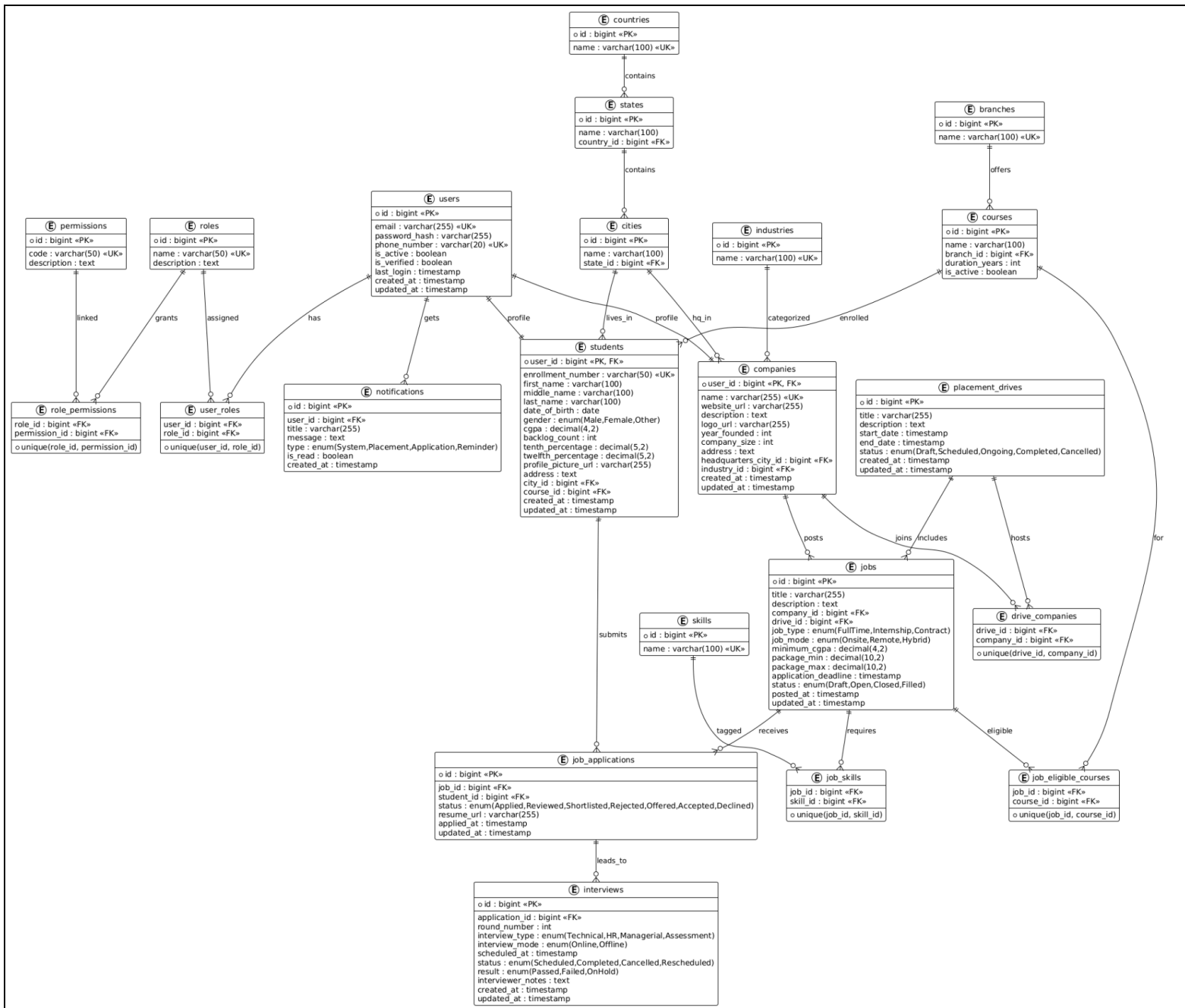
1.1. Architecture Diagram



1.2. ER Diagram



1.3. Class Diagram



2. Placemate REST API Documentation

Version: 1.0

2.1. Introduction

This document provides a complete reference for the Placemate REST API. The Placemate API is designed to facilitate and manage the entire campus placement process, providing a seamless interface for both students and placement cell administrators. The API offers resources for user management, company and job postings, student applications, interview scheduling, and offer management.

Intended Audience: This documentation is intended for developers, students, and administrators who will be interacting with the Placemate platform programmatically.

Base URL: All API endpoints are relative to the following base URL: <https://api.placemate.com/>

2.2. Authentication

The Placemate API uses JSON Web Tokens (JWT) for authentication. All requests to protected endpoints must include a valid JWT access token in the Authorization header.

Authentication Flow

- 1. Register an Account:** A new user first registers using the POST `/api/auth/register/` endpoint.
- 2. Log In to Get Tokens:** After successful registration, the user logs in with their credentials using the POST `/api/auth/login/` endpoint. This request returns an access token and a refresh token.
- 3. Authenticate Requests:** The access token must be included in the Authorization header of all subsequent requests to protected endpoints, using the Bearer scheme.
 - **Example Header:** Authorization: Bearer <your_jwt_access_token>
- 4. Refresh Token:** The access token has a limited lifetime. Once it expires, the client must use the refresh token with the POST `/api/auth/refresh/` endpoint to obtain a new access token without requiring the user to log in again.

2.3. API Reference

This section details every available endpoint, including required parameters, request bodies, and example responses.

2.3.1. Authentication & Authorization

Endpoints for user identity management.

Register User

- **Endpoint:** POST /api/auth/register/
- **Description:** Creates a new user account (Student or Admin).
- **Permissions:** Public
- **Request Body:**

```
{  
  "email": "test@example.com",  
  "password": "SecretPassword123",  
  "role": "Student"  
}
```

- **Success Response (201 Created):**

```
{  
  "id": 1,  
  "email": "test@example.com",  
  "role": "Student"  
}
```

Login User

- **Endpoint:** POST /api/auth/login/
- **Description:** Authenticates a user and returns a JWT token pair.
- **Permissions:** Public
- **Request Body:**

```
{  
  "email": "test@example.com",  
  "password": "SecretPassword123"  
}
```

- **Success Response (200 OK):**

```
{  
  "access": "<jwt_access_token>",  
  "refresh": "<jwt_refresh_token>"  
}
```

Refresh Access Token

- **Endpoint:** POST /api/auth/refresh/
- **Description:** Generates a new access token using a valid refresh token.
- **Permissions:** Public
- **Request Body:**

```
{  
  "refresh": "<jwt_refresh_token>"  
}
```

- **Success Response (200 OK):**

```
{  
  "access": "<new_jwt_access_token>"  
}
```

Forgot Password

- **Endpoint:** POST /api/auth/forgot-password/
- **Description:** Initiates the password reset process by sending an OTP to the user's email.
- **Permissions:** Public
- **Request Body:**

```
{  
  "email": "test@example.com"  
}
```

- **Success Response (200 OK):**

```
{  
  "message": "OTP sent to email"  
}
```


Reset Password

- **Endpoint:** POST /api/auth/reset-password/
- **Description:** Sets a new password for a user using a valid email and OTP.
- **Permissions:** Public
- **Request Body:**

```
{  
  "email": "test@example.com",  
  "otp": "123456",  
  "new_password": "NewSecretPassword123"  
}
```
- **Success Response (200 OK):**

```
{  
  "message": "Password reset successful"  
}
```

2.3.2. Students

Endpoints for managing student profiles.

Get My Profile

- **Endpoint:** GET /api/students/me/
- **Description:** Retrieves the profile of the currently logged-in student.
- **Permissions:** Student
- **Success Response (200 OK):**

```
{  
  "id": 1,  
  "email": "student@example.com",  
  "full_name": "John Doe",  
  "resume_url": "[https://example.com/resume.pdf](https://example.com/resume.pdf)",  
  "skills": ["Python", "JavaScript", "SQL"]  
}
```

Update My Profile

- **Endpoint:** PUT /api/students/me/

- **Description:** Updates the profile of the currently logged-in student.
- **Permissions:** Student
- **Request Body:**

```
{
  "full_name": "John A. Doe",
  "resume_url":
    "[https://example.com/new_resume.pdf](https://example.com/new_resume.pdf)",
  "skills": ["Python", "JavaScript", "SQL", "React"]
}
```
- **Success Response (200 OK):** The updated student profile object.

List All Students

- **Endpoint:** GET /api/students/
- **Description:** Retrieves a list of all student profiles.
- **Permissions:** Admin only
- **Success Response (200 OK):**

```
[
  {
    "id": 1,
    "email": "student1@example.com",
    "full_name": "John Doe"
  },
  {
    "id": 2,
    "email": "student2@example.com",
    "full_name": "Jane Smith"
  }
]
```

View Specific Student Profile

- **Endpoint:** GET /api/students/{id}/
- **Description:** Retrieves the profile of a specific student by their ID.
- **Permissions:** Admin only
- **URL Parameters:** id (integer, required): The ID of the student.

- **Success Response (200 OK):** A single student profile object.

2.3.3. Companies & Placement Drives

Endpoints for managing companies and placement events.

List Companies

- **Endpoint:** GET /api/companies/
- **Description:** Retrieves a list of all companies.
- **Permissions:** Authenticated Users (Student, Admin)
- **Success Response (200 OK):** An array of company objects.

Add Company

- **Endpoint:** POST /api/companies/
- **Description:** Adds a new company record to the database.
- **Permissions:** Admin only
- **Request Body:**

```
{
  "name": "Tech Solutions Inc.",
  "description": "A leading provider of cloud services.",
  "website": "[https://techsolutions.com](https://techsolutions.com)"
}
```
- **Success Response (201 Created):** The newly created company object.

List Placement Drives

- **Endpoint:** GET /api/placement-drives/
- **Description:** Retrieves a list of all placement drives.
- **Permissions:** Authenticated Users (Student, Admin)
- **Success Response (200 OK):** An array of placement drive objects.

Create Placement Drive

- **Endpoint:** POST /api/placement-drives/
- **Description:** Creates a new placement drive.
- **Permissions:** Admin only
- **Request Body:**

```
{
  "title": "Fall 2025 Placement Drive",
```

```
"start_date": "2025-10-01T09:00:00Z",  
"end_date": "2025-10-15T17:00:00Z"  
}
```

- **Success Response (201 Created):** The newly created placement drive object.

Note: Endpoints for Jobs, Applications, Interviews, Offers, and Notifications would follow the same detailed format.

2.4. Error Handling

The Placemate API uses conventional HTTP status codes to indicate the success or failure of an API request.

Code	Status	Meaning
200	OK	The request was successful.
201	Created	The resource was created successfully.
400	Bad Request	The server could not understand the request due to invalid syntax (e.g., malformed JSON, missing required fields).
401	Unauthorized	The request was not authenticated. The Authorization header is either missing or contains an invalid token.
403	Forbidden	The authenticated user does not have the necessary permissions to

		access this resource.
404	Not Found	The requested resource (e.g., a specific job or student) does not exist.
500	Internal Server Error	An unexpected error occurred on the server.

2.5. Example Workflow: A Student Applies for a Job

This section demonstrates a typical workflow for a student using the API.

Step 1: Log In

The student authenticates to get an access token.

- **Request:** POST /api/auth/login/


```
{
  "email": "student1@example.com",
  "password": "StudentPassword123"
}
```
- **Response:**

```
{
  "access": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "refresh": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."
}
```

The student's application now stores the access token for future requests.

Step 2: Find an Available Job

The student fetches the list of all jobs.

- **Request:** GET /api/jobs/ (with Authorization: Bearer <access_token>)
- **Response:**

```
[
  {
    "id": 101,
```

```

    "title": "Software Engineer Intern",
    "company_name": "Tech Solutions Inc.",
    "description": "...",
  },
  {
    "id": 102,
    "title": "Data Analyst",
    "company_name": "Data Insights LLC",
    "description": "...",
  }
]

```

The student decides to apply for the "Software Engineer Intern" position (job ID 101).

Step 3: Apply for the Job

The student submits an application for the chosen job.

- **Request:** POST /api/job-applications/ (with Authorization: Bearer <access_token>)

```

{
  "job_id": 101
}

```

- **Response (201 Created):**

```

{
  "id": 55,
  "job_id": 101,
  "student_id": 1,
  "status": "Applied",
  "applied_at": "2025-09-13T12:30:00Z"
}

```

Step 4: Check Application Status

The student can later view all their submitted applications.

- **Request:** GET /api/job-applications/me/ (with Authorization: Bearer <access_token>)
- **Response:**

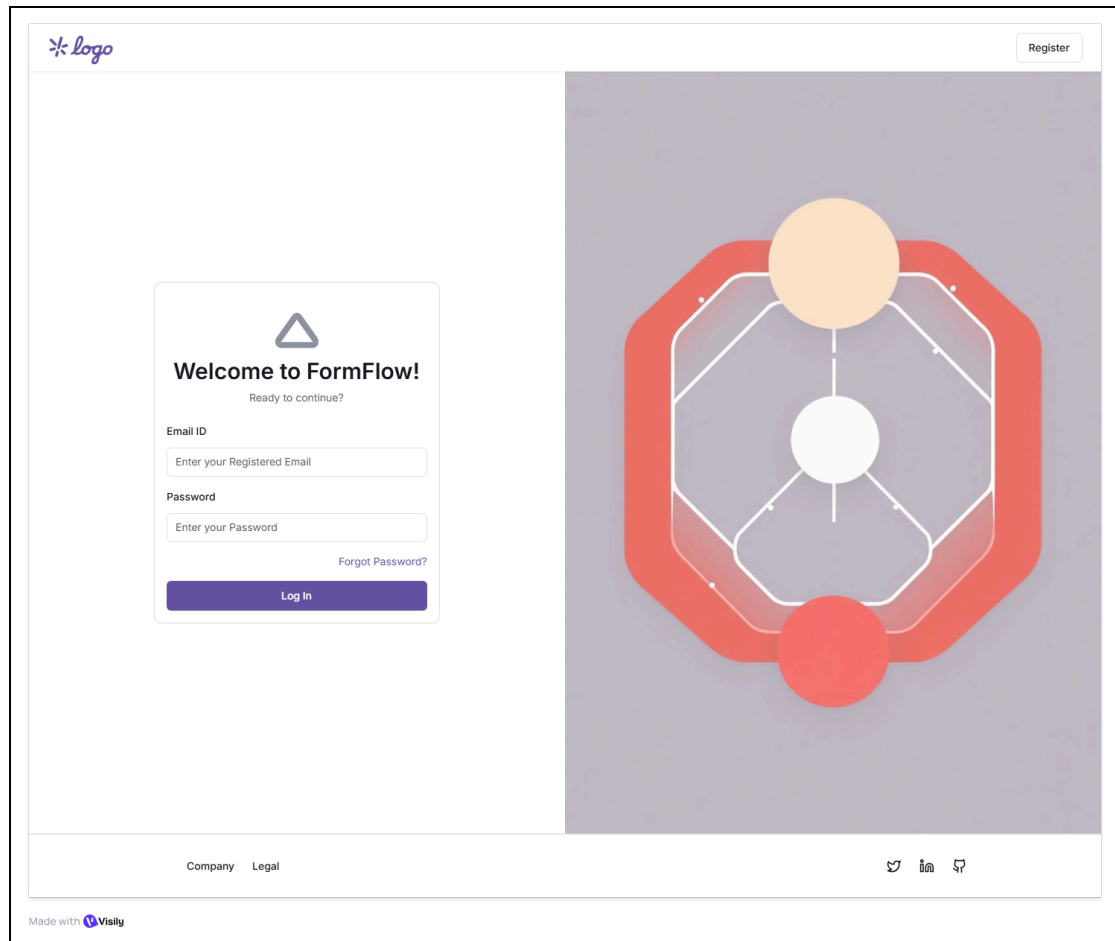
```
[
  {
    "id": 55,
    "job_title": "Software Engineer Intern",
    "company_name": "Tech Solutions Inc.",
    "status": "Applied"
  }
]
```

This confirms the application was successful and is now being processed.

3. Wireframes / UI UX Mockups

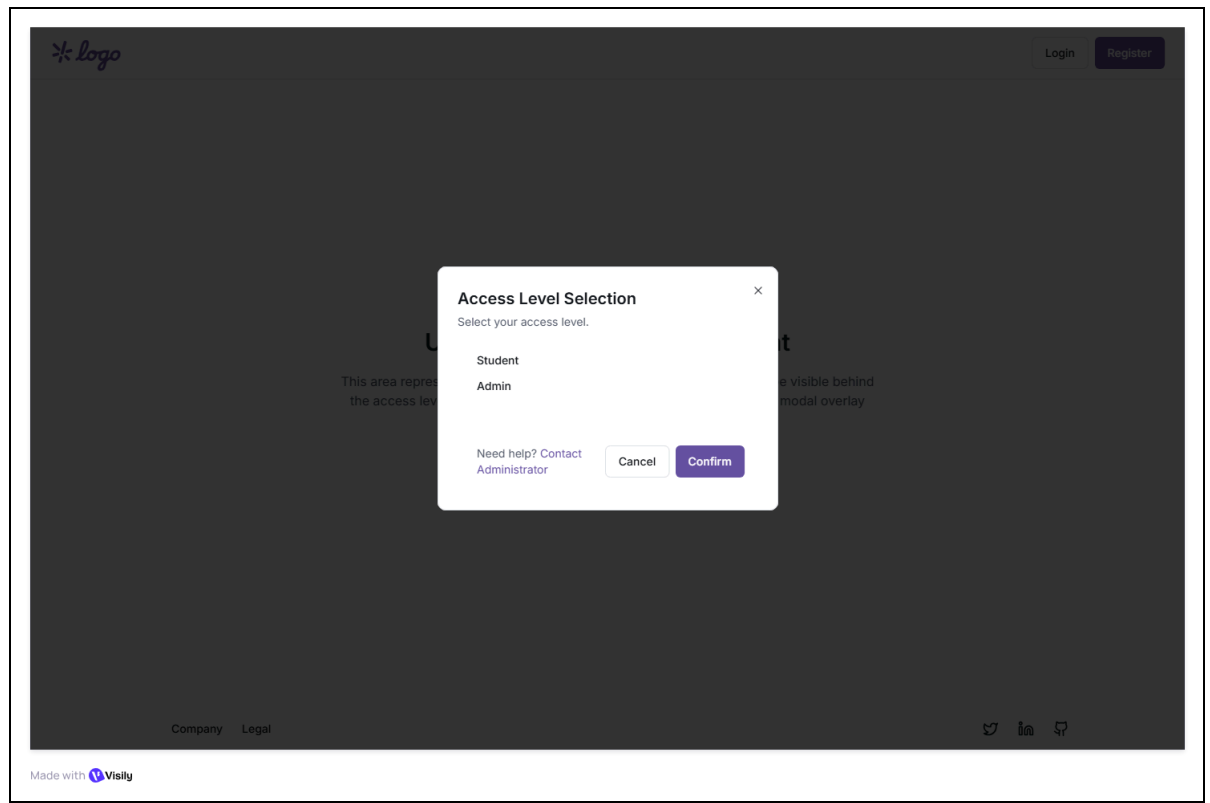
3.1. General

- Login Screen



[Image: Login page with fields for email, password, and a login button.]

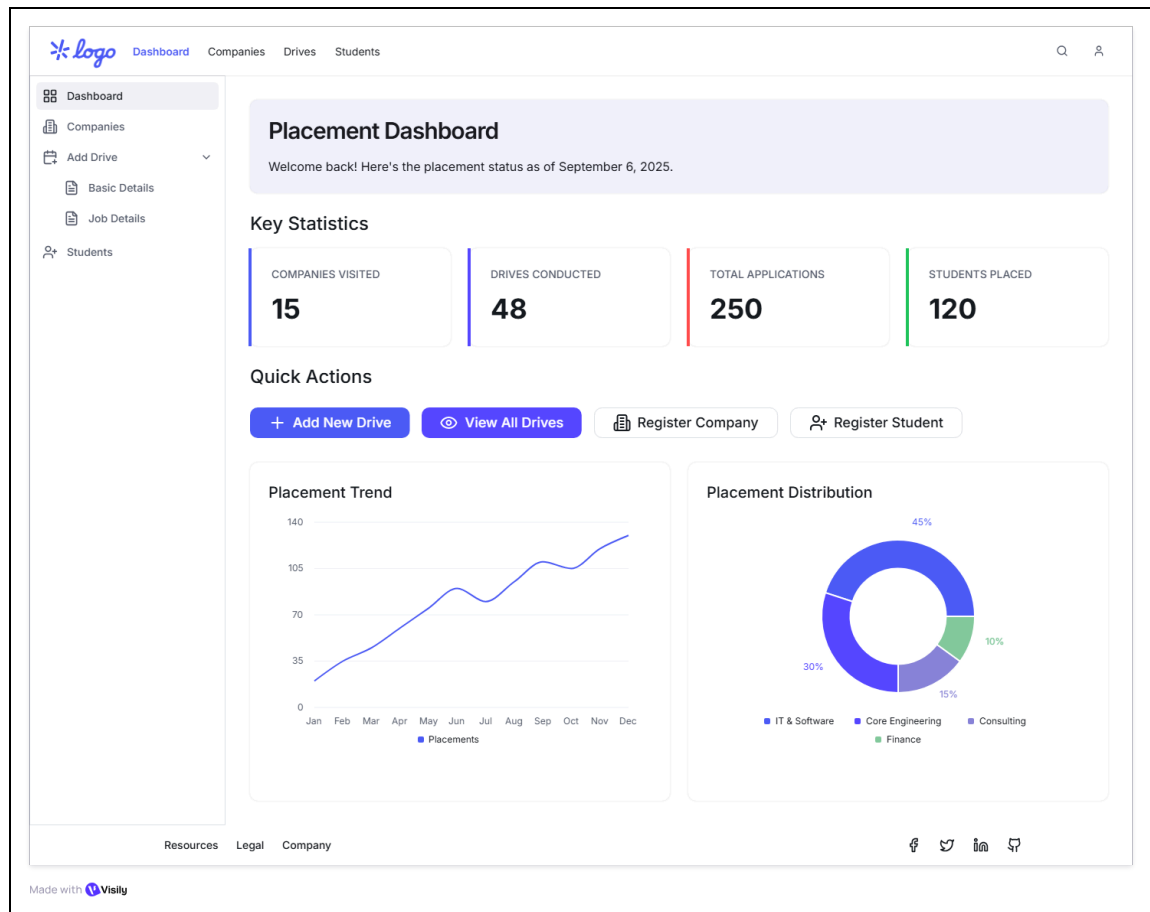
- **Role-Based Access**



[Image: Diagram or mockup showing different views/permissions for Admin vs. Student roles.]

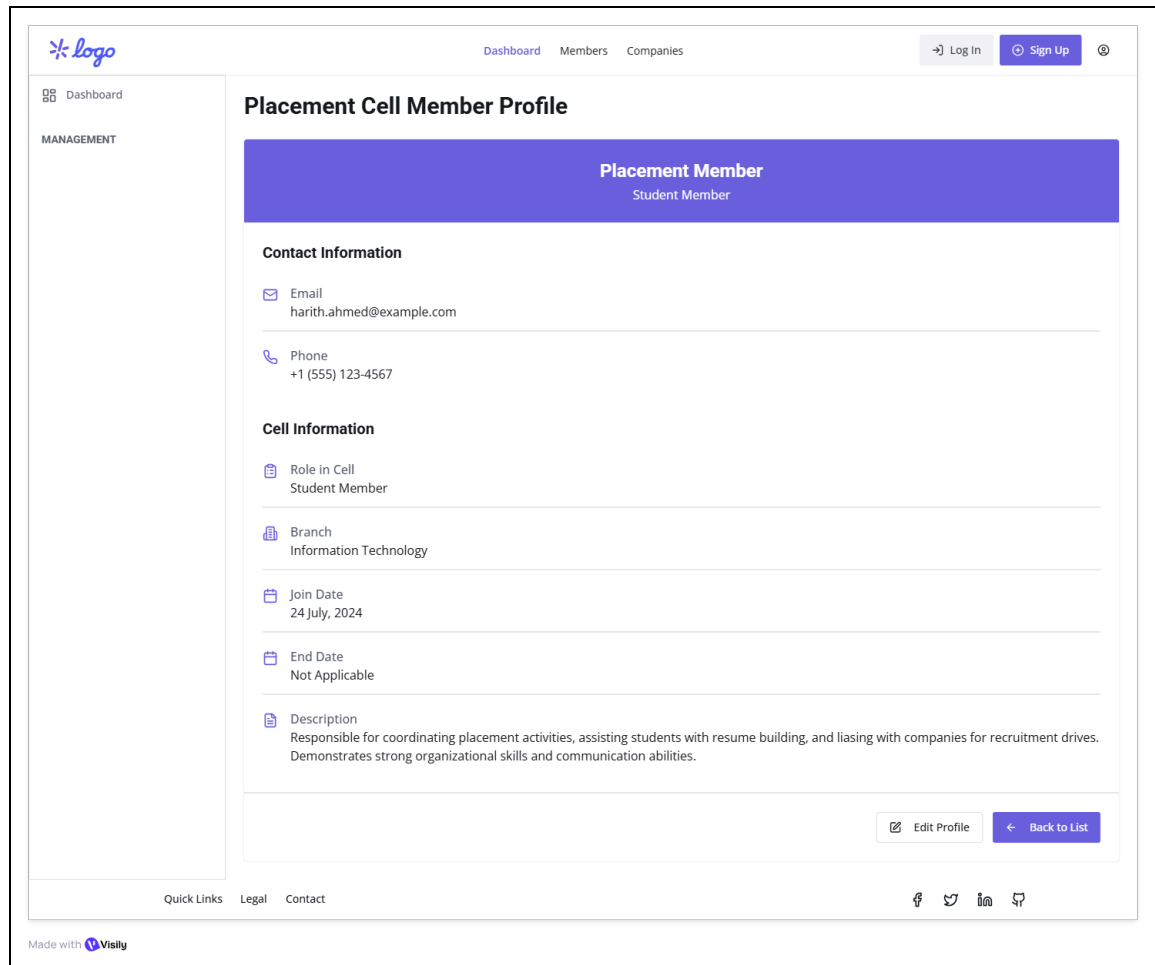
3.2. Admin Role Access

- 3.2.1. Placement Dashboard



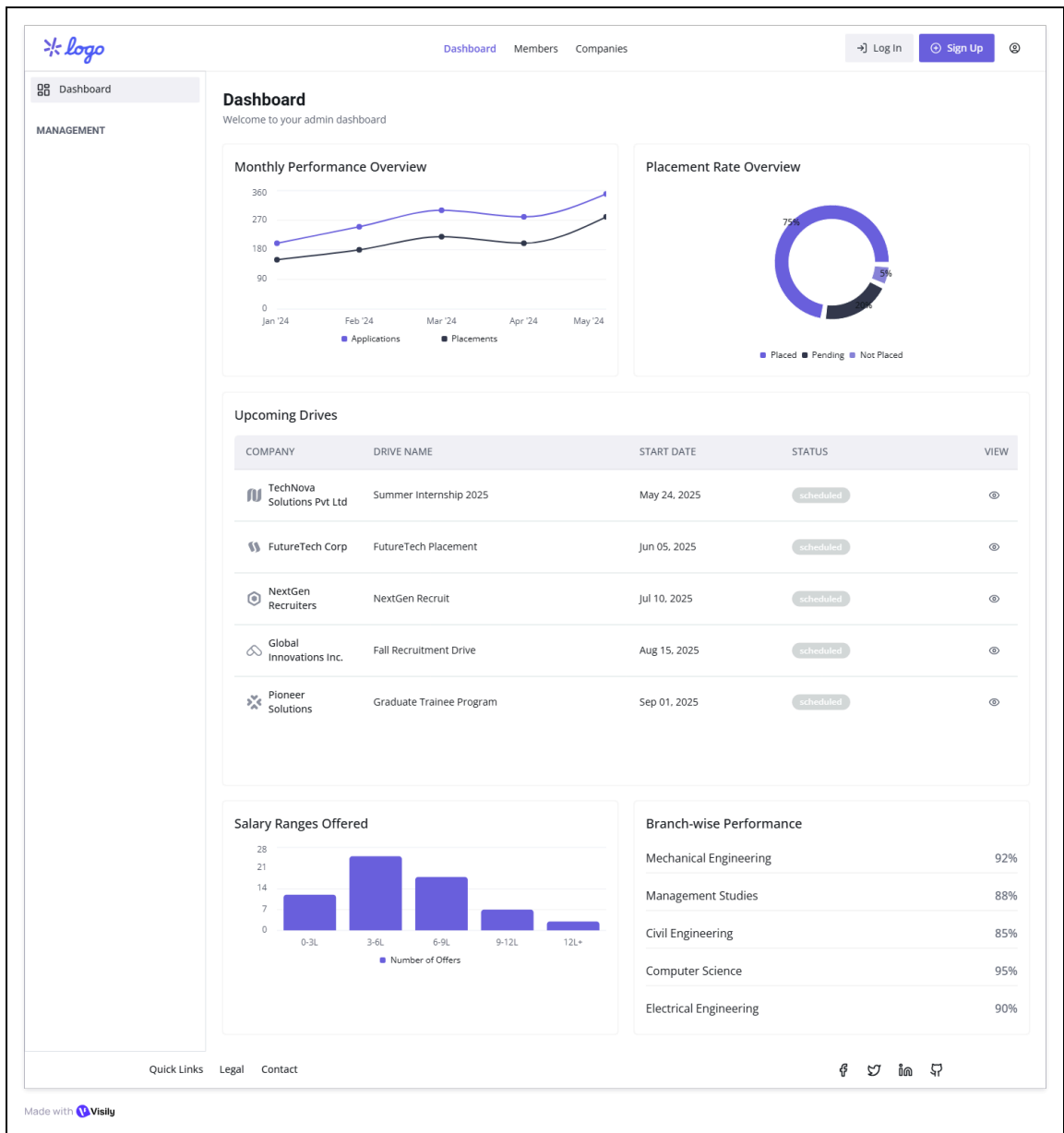
[Image: Admin dashboard showing key metrics like total companies, active drives, applications received, and offers made.]

- 3.2.2. Member Profile



[Image: Profile page for an admin/cell member.]

● 3.2.3. Dashboard (Alternative View)



[Image: A different view of the admin dashboard.]

- 3.2.4. Student Registration

Student Manual Registrations

Add student details

Student Information
Enter student details to register them in the portal.

First Name *
Enter first name

Middle Name
Enter middle name

Last Name *
Enter last name

Email *
example@college.com

Phone Number *
e.g., +1 (555) 123-4567

Date of Birth *
dd/mm/yyyy

Gender
Select Gender

Enrollment No. *
e.g., 2021BTECHCS001

Address
Enter full address

Joining Year *
e.g., 2021

Current CGPA
e.g., 8.5

Course *
Select Course

Placement Status *
Select Placement Status

Graduation Status *
Select Graduation Status

Company Placed In
e.g., Google

Job Role
e.g., Software Engineer

Package (LPA)
e.g., 12.5


Register Student

Company Legal Support

Made with Visiily

[Image: Form for an admin to manually register a new student.]

- 3.2.5. Register Cell Members


 Login

Register Cell Member


Add placement cell member in placemate.

Personal Information

Email Address *

 student117@example.com

Phone Number *

 1234567812

Cell Information


Role in Cell *

Student Member


Branch

Information Technology

Join Date *

 2024-07-24

End Date

 dd/mm/yyyy




Leave blank if no end date


Additional Information

Description / Notes

Enter any additional information about the member


Cancel Add Member

[Company](#) [Legal](#)   

Made with  Visily

[Image: Form for an admin to register other placement cell members.]

- 3.2.6. Registered Students List



[Student Manual Registrations](#)
[Registered Students](#)
[Applications Status](#)
[Detailed Application View](#)

Registered Students

Manage Registered Students for placement management.

Select Course

Select Batch

Select Placement Status

Reset

S.No	Enrollment No	Full Name	Batch	Course	Email	Placement Status	Action
1	2021001	John Doe	2020-2026	M.Sc IT	john.doe@example.com	Placed	View Edit Delete
2	2021002	Jane Smith	2020-2022	B.Sc IT	jane.smith@example.com	Internship	View Edit Delete
3	2021003	Michael Brown	2019-2021	Diploma in Information Technology	michael.brown@example.com	Not Placed	View Edit Delete
4	2021004	Aarav Patel	2020-2022	M.E Electrical Engineering	student61@example.com	Placed	View Edit Delete
5	2021005	Sanya Verma	2019-2021	B.E Electronics and Communication	student62@example.com	Not Placed	View Edit Delete
6	2021006	Dev Singh	2021-2023	M.Sc Data Science	student63@example.com	Internship	View Edit Delete
7	2021007	Rhea Shah	2020-2022	Diploma in Information Technology	student64@example.com	Job Offer Received	View Edit Delete
8	2021008	Kabir Khan	2018-2020	B.Tech Computer Science	student65@example.com	Placed	View Edit Delete
9	2021009	Meera Gupta	2010-2016	M.E Mechanical Engineering	student66@example.com	Not Placed	View Edit Delete
10	2021010	Arjun Reddy	2019-2021	B.E Electrical Engineering	student67@example.com	Placed	View Edit Delete
11	2021011	Priya Sharma	2020-2026	M.Sc IT	priya.sharma@example.com	Internship	View Edit Delete
12	2021012	Rajesh Kumar	2020-2022	B.Sc IT	rajesh.kumar@example.com	Placed	View Edit Delete

Showing 1 to 12 of 12 students





Previous


1

Next

10 records

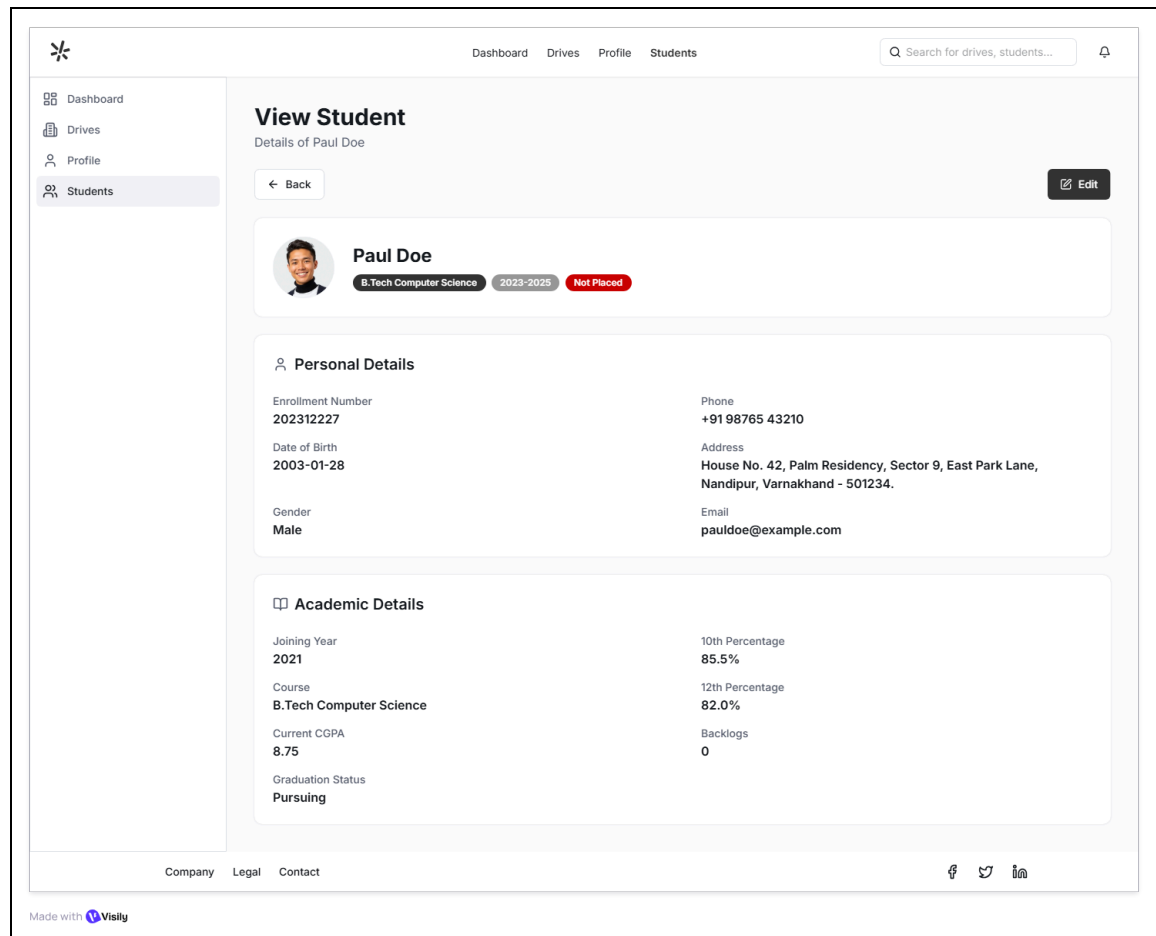
[Company](#)
[Legal](#)
[Support](#)

Made with  Visily

[Image: A table view listing all registered students with key details.]

- 3.2.7. Student Details View



[Image: Detailed view of a single student's profile, accessible by the admin.]

- 3.2.8. Register Company

Register New Company

Company Information

Company Logo
PNG, JPG, GIF up to 5MB

Company Name*
NexGen Innovations Pvt Ltd

Company Website*
https://nexgeninnovations.com

Founded Year
2020

Industry*
Technology

Company Size*
51-200 employees

Company Type*
Private Limited

Company Category*
Software Development

Company Description
NexGen Innovations Pvt Ltd specializes in developing cutting-edge software solutions and providing digital transformation services. We focus on cloud-native applications, AI-driven analytics, and robust cybersecurity frameworks to empower businesses globally.

Contact Information

Contact Person Name*
Priya Sharma

Contact Person Email*
priya.sharma@nexgen.com

Contact Person Position*
Head of Talent Acquisition

Address Information

Address Line*
Block 7, Cyber Hub, Sector 21, Infinity Towers, Gurgaon

Country*
India

State*
Haryana

Headquarter City*
Gurgaon

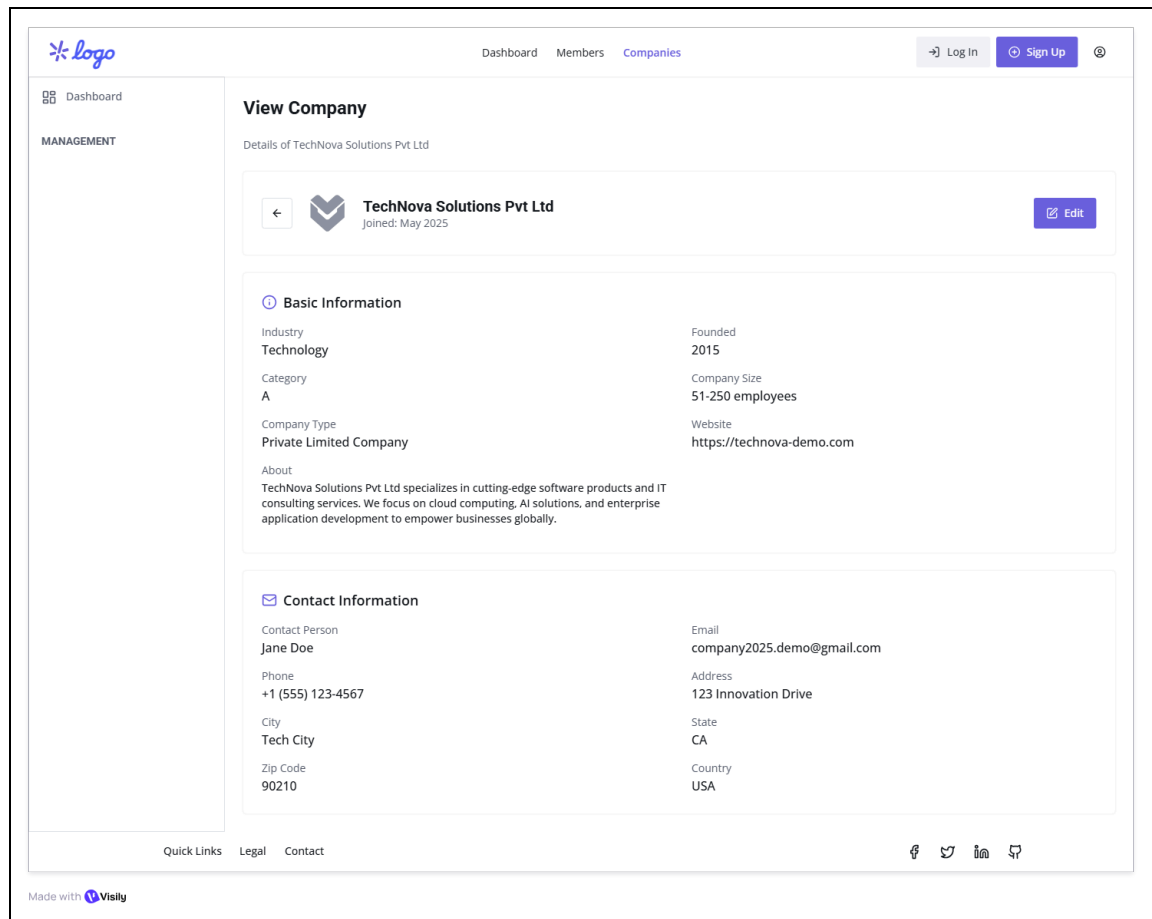
[Register Company](#)

Quick Links Legal Contact

Made with Visally

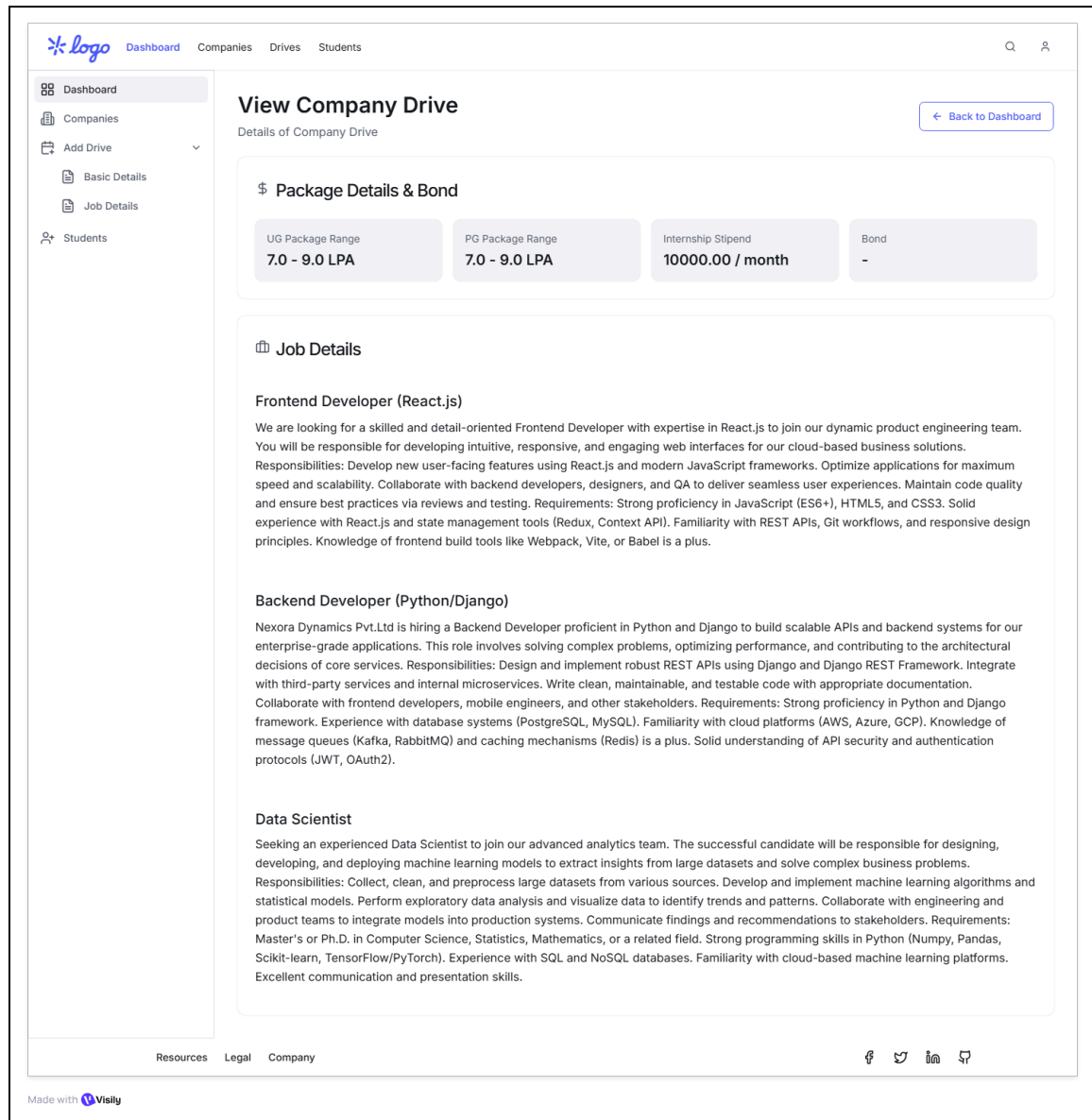
[Image: Form for an admin to add a new company to the system.]

- 3.2.9. View Company Details



[Image: Detailed view of a registered company's profile.]

- **3.2.10. Company Drive Details**



[Image: View showing the details of a specific placement drive organized by a company.]

- 3.2.11. Add Drive - Basic Details

The screenshot displays a web application interface for adding a new placement drive. The top navigation bar includes a logo, a search icon, and a user profile icon. The main navigation menu on the left lists 'Dashboard', 'Companies', 'Add Drive' (selected), 'Basic Details' (active), 'Job Details', and 'Students'. The 'Add Drive' section is titled 'Add Drive' with the subtitle 'Add company drive in placemate.' The form is divided into two main sections: 'Basic Drive Details' and 'Academic Eligibility Criteria'. The 'Basic Drive Details' section contains fields for 'Company' (a dropdown menu), 'Drive Name' (a text input with a placeholder 'e.g., Software Development Internship 2024'), 'Job Type' (a dropdown menu), 'Job Mode' (a dropdown menu), 'Minimum CGPA' (a text input with a placeholder 'e.g., 7.5'), 'Eligible Courses' (a text input with a placeholder 'Type to filter courses and select eligible courses'), 'Required Skills' (a text input with a placeholder 'Type to filter skills and select required skills'), and 'Posting locations' (a text input with a placeholder 'Type to filter cities and select posting locations'). The 'Academic Eligibility Criteria' section contains four fields: 'Minimum 10th Percentage' (a text input with a placeholder 'e.g., 60' and a percentage sign), 'Minimum 12th Percentage' (a text input with a placeholder 'e.g., 60' and a percentage sign), 'Minimum Diploma Percentage' (a text input with a placeholder 'e.g., 60' and a percentage sign), and 'Minimum UG CGPA' (a text input with a placeholder 'e.g., 7.5 or 75' and a slash followed by '10'). At the bottom right of the form are 'Cancel' and 'Next' buttons. The footer of the application includes 'Resources', 'Legal', 'Company', and social media icons, along with a 'Made with Visually' watermark.

Dashboard Companies Drives Students

Dashboard Companies Add Drive Basic Details Job Details Students

Add Drive

Add company drive in placemate.

Basic Drive Details

Company *
Select Company

Drive Name *
e.g., Software Development Internship 2024

Job Type *
Select Job Type

Job Mode *
Select Job Mode

Minimum CGPA
e.g., 7.5

Eligible Courses
Type to filter courses and select eligible courses

Required Skills
Type to filter skills and select required skills

Posting locations
Type to filter cities and select posting locations

Academic Eligibility Criteria

Minimum 10th Percentage
e.g., 60 %

Minimum 12th Percentage
e.g., 60 %

Minimum Diploma Percentage
e.g., 60 %

Minimum UG CGPA
e.g., 7.5 or 75 / 10

Cancel Next

Resources Legal Company

Made with Visually

[Image: First step of a form for creating a new placement drive, asking for basic info like title and dates.]

- 3.2.12. Add Drive - Job Details

The screenshot shows a web application interface for adding a company drive. The top navigation bar includes a logo, 'Dashboard', 'Companies', 'Drives', and 'Students'. A sidebar on the left contains links to 'Dashboard', 'Companies', 'Add Drive' (selected), 'Basic Details', 'Job Details', and 'Students'. The main content area is titled 'Add Drive' with the subtitle 'Add company drive in placement.' Below this is a 'Job Details' section containing two job entries. Each entry has a 'Job Title' field and a 'Job Description' field. The first job is 'Job #1' with title 'Frontend Developer (React.js)' and description 'Familiarity with REST APIs, Git workflows, and responsive design principles. Knowledge of frontend build tools like Webpack, Vite, or Babel is a plus.' The second job is 'Job #2' with title 'Backend Developer (Python/Django)' and description 'Familiarity with Docker, Git, and API versioning practices. Knowledge of authentication protocols (JWT, OAuth2) is desirable.' Below the job entries is a button 'Add Another Job'. At the bottom right are 'Back' and 'Create Drive' buttons. The footer includes 'Resources', 'Legal', 'Company', social media icons, and 'Made with Visily'.

Dashboard Companies Drives Students

Dashboard Companies Add Drive Basic Details Job Details Students

Add Drive

Add company drive in placement.

Job Details

Job #1

Job Title *

Frontend Developer (React.js)

Job Description *

Familiarity with REST APIs, Git workflows, and responsive design principles. Knowledge of frontend build tools like Webpack, Vite, or Babel is a plus.

Job #2

Job Title *

Backend Developer (Python/Django)

Job Description *

Familiarity with Docker, Git, and API versioning practices. Knowledge of authentication protocols (JWT, OAuth2) is desirable.

Add Another Job

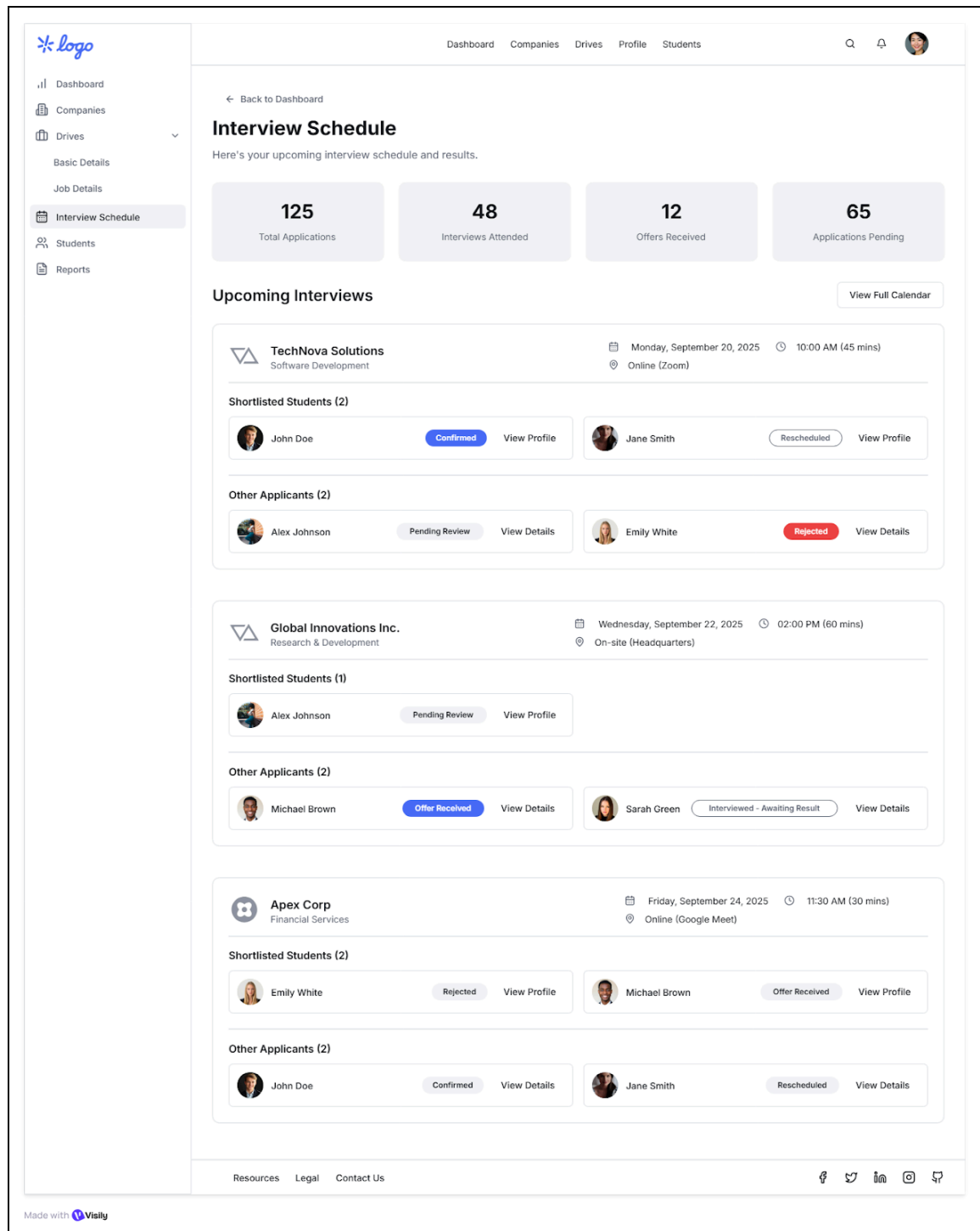
Back Create Drive

Resources Legal Company

Made with Visily

[Image: Second step of the form for creating a drive, focused on adding job roles, descriptions, and requirements.]

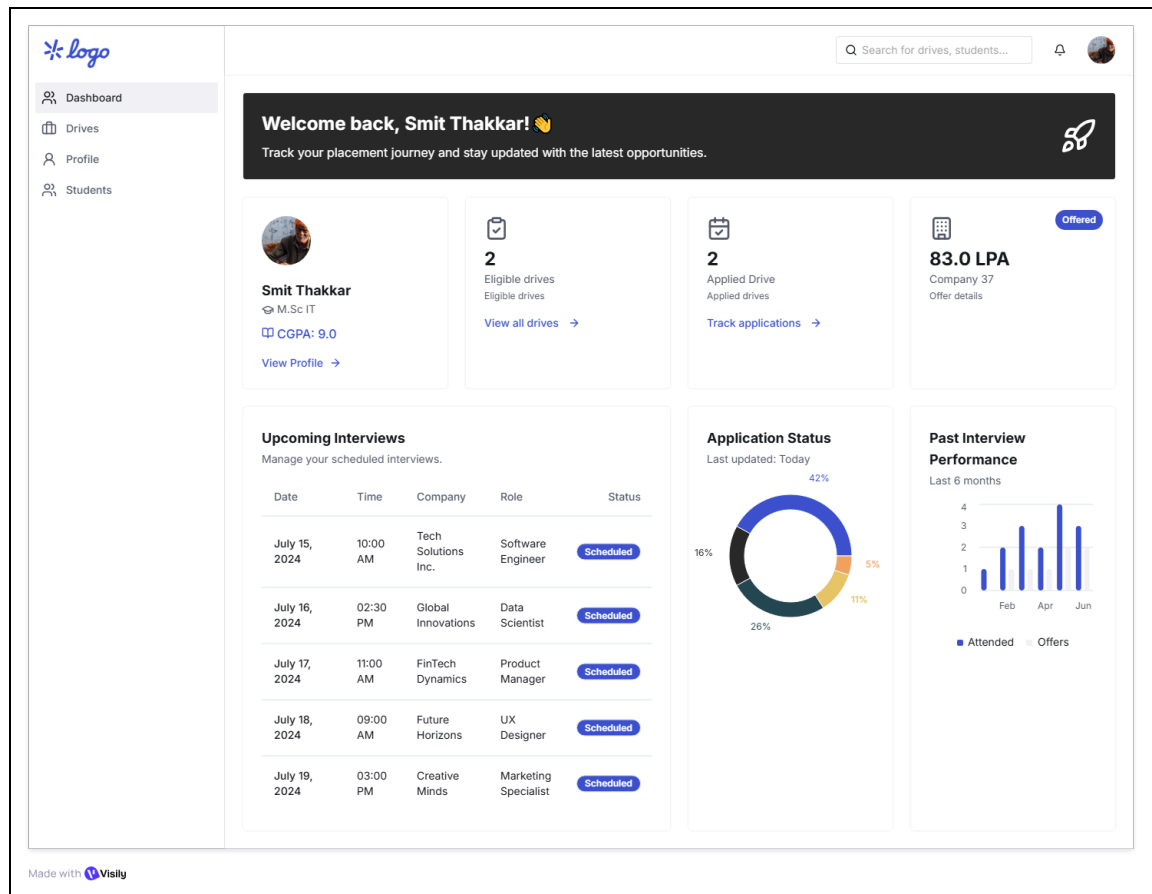
- 3.2.13. Admin Interview Scheduling



[Image: Interface for an admin to schedule interviews between students and companies.]


3.3. Student Role Access

- 3.3.1. Student Dashboard



[Image: Student's main dashboard showing upcoming interviews, application statuses, and available drives.]


- 3.3.2. Edit Profile



Edit Profile

You are allowed to update your phone number, address, profile picture, 10th percentage, and 12th percentage.

Profile Picture



Personal Information

Phone Number *

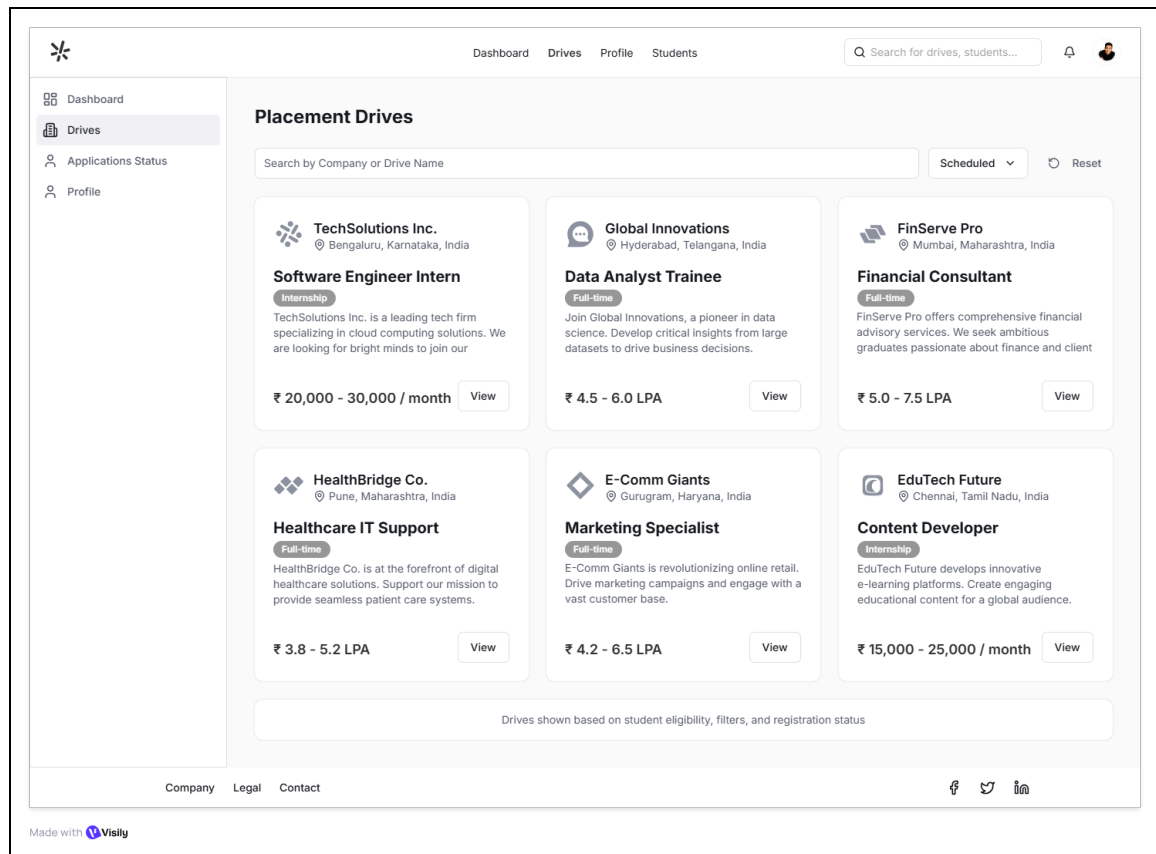
Address *

10th Percentage *

12th Percentage *

[Image: Form for a student to edit their personal details, skills, and upload a resume.]

- 3.3.3. View Drive



[Image: Page showing the details of an active placement drive, including participating companies and job roles.]

- 3.3.4. Detailed Application View

Application Details [← Back to Applications](#)

Job Description

Senior Frontend Developer (React.js)
Tech Innovators Inc.

About the Role:
We are seeking a highly skilled and passionate Senior Frontend Developer with extensive experience in React.js to join our innovative team. You will be responsible for designing, developing, and maintaining high-performance, scalable, and user-friendly web applications. This role requires a strong understanding of modern JavaScript practices, component-based architecture, and state management.

Key Responsibilities:
Develop and maintain high-quality user interfaces using React.js and TypeScript.
Collaborate with UX/UI designers to translate wireframes and mockups into interactive web experiences.
Optimize applications for maximum speed and scalability.
Participate in code reviews to ensure code quality and adherence to best practices.
Mentor junior developers and contribute to a culture of continuous learning and improvement.

Required Skills:
React.js TypeScript JavaScript (ES6+) Tailwind CSS RESTful APIs
Git Redux/Zustand Unit Testing (Jest/RTL)

Salary/Package:
12 - 18 LPA

Add/Update Resume

Please provide a link to your updated resume that matches the job role before opting in.

To ensure the best fit for your application, kindly update your resume according to the specific requirements of the job role. This will help in aligning your skills with the position and increase your chances of success.

Resume Link
🔗 Enter your resume link (Google Drive, Dropbox, etc.)
🔒 Make sure the link has proper sharing permissions.

Cover Letter (Optional)
Tell us why you are a good fit for this role...

Submit Application

Company Legal Support

Made with Visily

[Image: A detailed view of a job posting that the student can apply to.]

- 3.3.5. Application Status List

Company	Drive Date	Stipend	Status	Submitted On	Submitted Resume	View Drive
Company 37	May 10, 2025	₹15000.00/month	Selected	April 20, 2025	View Resume	View Drive
Company 42	July 10, 2025	₹5000.00/month	Applied	May 23, 2025	View Resume	View Drive
Company 34	May 10, 2025	₹12000.00/month	Shortlisted	May 04, 2025	View Resume	View Drive
Company 20	May 10, 2025	₹12000.00/month	Rejected	May 04, 2025	View Resume	View Drive
Apple	April 20, 2025	₹50000.00/month	Applied	April 19, 2025	View Resume	View Drive
TechNova Solutions Pvt Ltd	May 24, 2025	₹10000.00/month	Applied	May 24, 2025	View Resume	View Drive
InnovateX Group	June 01, 2025	₹20000.00/month	Interviewing	May 28, 2025	View Resume	View Drive

[Image: A list of all jobs the student has applied for, along with the current status of each application (Applied, Shortlisted, Interviewing, Offered).].