Project Title: **Campus Placement Management**

Course Name:

Code Query: The Ultimate PL/SQL and Data Science Bootcamp

Institution Name:

Lovely Professional University (LPU)

Submitted To: Dr. Avinash Kaur

Date of Submission: 14 -7 -2025

Submitted by:

Yashika malik (reg no: 12324446)

Chesta (reg no:12301284)

z

Acknowledgment

I take this opportunity to express my sincere gratitude to my instructor, **[Dr. Avinash Kaur]**, for their expert guidance, encouragement, and valuable insights throughout the course of this project. Their continuous support and constructive feedback were instrumental in shaping the direction and outcome of this work.

I would also like to extend my thanks to the faculty and staff for providing the necessary academic environment and resources that enabled me to undertake and complete this project successfully.

The structured guidance and support from my institution have greatly contributed to the learning process and the professional execution of this project. I deeply appreciate the opportunity to explore and apply theoretical concepts to a practical implementation.

Lastly, I am grateful to my peers and well-wishers for their encouragement and cooperation throughout this journey.

z

CERTIFICATE

This is to certify that Yashika Malik, Chesta bearing Registration no. 12324446, 12301284 has completed PL/SQL project titled, **"Campus Placement Management System"** under my guidance and supervision. To the best of my knowledge, the present work is the result of his/her original development, effort and study.

Dr. Avinash Kaur

Designation of the Supervisor

School of computer Science and Engineering

Lovely Professional University

Phagwara, Punjab.

z

Introduction

The **Campus Placement Management System** is a comprehensive database-driven application developed to streamline, organize, and automate the entire campus recruitment lifecycle. With the increasing scale of recruitment activities across academic institutions, there arises a strong need for an efficient, reliable, and centralized system that can effectively manage student registrations, company interactions, eligibility checks, and interview schedules. This project is aimed at addressing those needs through a well-structured and user-friendly platform.

The system facilitates the seamless registration of students with personal, academic, and professional details, including resume uploads. It maintains detailed records of companies visiting the campus, the job roles offered, eligibility criteria, application deadlines, and interview schedules. One of the core functionalities of the system is the automated eligibility checking module, which ensures that only qualified students, based on predefined conditions such as branch, CGPA, and skill set, are shortlisted for relevant opportunities. This reduces manual filtering work and ensures accuracy in the selection process.

Furthermore, the system empowers placement coordinators by providing a centralized interface to track all placement activities. From viewing the list of registered students and applied companies to managing schedules and tracking placement status — every essential function is efficiently handled within the application. The end goal is to significantly reduce the administrative workload of the placement cell, minimize human errors, and enhance the overall transparency and speed of the recruitment process.

The project has been developed using **Structured Query Language (SQL)** and **PostgreSQL**, a powerful open-source object-relational database system. These technologies offer robust data integrity, high performance, scalability, and security — making them ideal for handling the complex relational data involved in placement activities.

By implementing this system, academic institutions can ensure a more organized, digital-first approach to campus placements, benefitting both students and recruiters alike.

Purpose of the Application

he primary objective of the **Campus Placement Management System** is to facilitate the smooth and effective handling of all placement-related activities within an academic institution by leveraging database automation. The system is designed to simplify and enhance the efficiency of the placement process for both students and administrators. The following points highlight the detailed purpose of this project:

- **Automate Student Registration Process:**
  To allow students to register on the platform by entering their personal, academic, and contact details. The system stores this data securely and makes it easily accessible for placement-related operations.
- **Centralized Management of Student Data:**
  To maintain a well-organized repository of student profiles, resumes, and application history, enabling quick retrieval and monitoring by placement officers.
- **Efficient Company Data Handling:**
  To record and manage company visits, job profiles, eligibility requirements, offered packages, job locations, and other recruitment details in a structured and accessible format.
- **Reduce Manual Errors and Administrative Workload:**
  To eliminate redundant paperwork and manual eligibility checks, ensuring accuracy and reducing the burden on the placement cell staff.
- **Interview Scheduling and Management:**
  To help schedule interviews for eligible candidates, send automated updates, and maintain logs of interview outcomes and final placement status.
- **Enhance Transparency and Communication:**
  To promote clear communication between students, placement officers, and companies through systematic updates on application status, interview dates, and selection outcomes.
- **Real-Time Access to Placement Statistics:**
  To provide analytical insights and dashboards for tracking placement records, student participation, and company engagement trends.

z

- **Create a Scalable and Secure Digital Platform:**
  To design a reliable, scalable, and secure application using SQL and PostgreSQL technologies that can handle large datasets efficiently.
- **Support Decision Making:**
  To assist placement coordinators in making informed decisions using structured data, eligibility filters, and company requirements.

z

Scope of the System:

The **Campus Placement Management System** is designed with flexibility, scalability, and real-world usability in mind. Its implementation provides a practical and efficient solution for managing end-to-end campus placement activities in educational institutions. The scope of the system includes the following:

- **Applicable to Colleges and Universities:**
  The system is suitable for a wide range of academic institutions, including engineering colleges, management schools, and universities that conduct regular campus recruitment drives. It can be customized to meet the specific requirements of any institution.
- **Scalable for Large Student and Company Data:**
  Built using robust database technologies like SQL and PostgreSQL, the system is capable of handling thousands of student records and multiple company profiles concurrently, without performance degradation. This makes it highly scalable for institutions with large student enrollments and frequent recruitment events.
- **Real-Time and Accurate Eligibility Checks:**
  The system ensures that only eligible students are allowed to apply for company drives by checking real-time data against criteria such as CGPA, academic branch, backlog status, and other conditions specified by the recruiter. This reduces errors and manual filtering, enhancing accuracy and trust.
- **Resume Upload and Document Handling:**
  The platform can be extended to support the upload and storage of student resumes and supporting documents, making them readily accessible to recruiters and placement coordinators during the hiring process.
- **Feedback Collection and Result Tracking:**
  Post-placement, the system can be enhanced to include modules for collecting feedback from companies about students, as well as tracking the outcome of each drive, including selected candidates and offered packages.
- **Secure and Role-Based Access Control:**
  The system can incorporate user roles such as Administrator, Placement Officer, Student, and Recruiter to ensure controlled and secure access to relevant modules and data.

z

- **Adaptable for Future Enhancements:**
  The modular architecture allows for future upgrades, such as automated notifications via email/SMS, analytics dashboards for placement trends, integration with external job portals, and AI-based recommendations.
- **Web-Based Interface for Remote Access:**
  The system can be deployed with a web-based frontend to allow students, companies, and placement teams to access and manage placement-related information anytime, from anywhere.
- **Paperless and Time-Efficient Operations:**
  By digitizing the entire placement process, the system reduces paperwork, shortens the time taken for student-company matching, and enhances the efficiency of the overall recruitment workflo

Technology Used:

- Database: PostgreSQL (accessed via pgAdmin4)
- Backend: Node.js (Express)
- Frontend: React (VS Code as IDE)

z

Problem Statement

Traditional placement procedures are mostly manual and inefficient. Institutions face challenges in:

- Maintaining accurate student records.
- Tracking eligibility based on company criteria.
- Coordinating company visits and interview schedules.
- Ensuring data consistency across multiple stakeholders.

  This project addresses the need for a **centralized, automated system** that streamlines these processes using SQL.

Objectives of the Project

The primary goals of this project are:

- To register students with academic and personal details.
- To store and manage company details including eligibility criteria.
- To automate the process of checking which students are eligible for which companies.
- To schedule and manage interviews between students and companies.
- To maintain logs using triggers and handle exceptions gracefully.

System Requirements

Hardware Requirements:

- Processor: Intel Core i5 or above
- RAM: 8 GB
- Hard Disk: 500 GB or more
- Display: 1024x768 resolution monitor

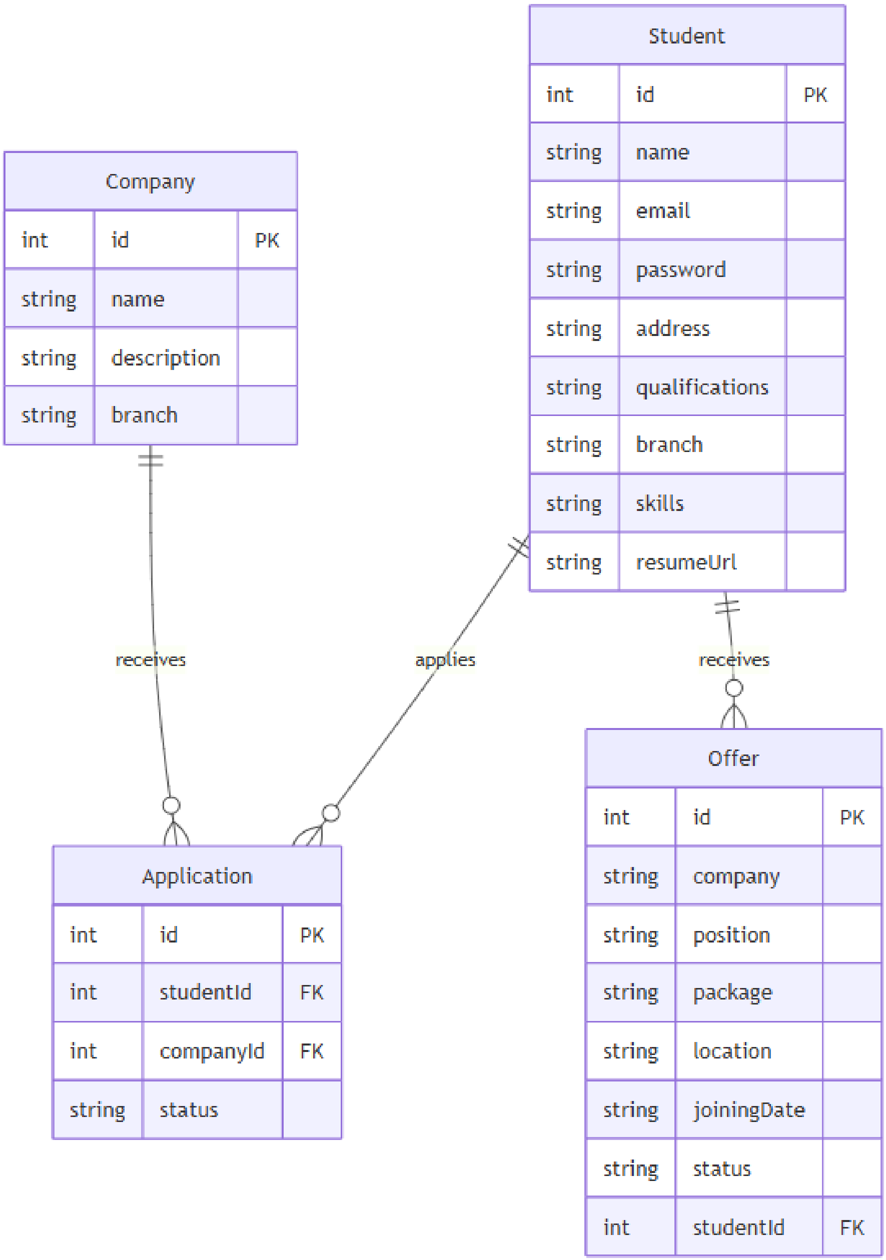
Software Requirements:

- Operating System: Windows 10 / Linux

- Database: PostgreSQL (accessed via pgAdmin4)
- Development Tools: VS Code, Node.js, React, Prisma ORM

System Design:

ER Diagram: Represents the relationship between Students, Companies, Interviews, and Eligibility.



Company		
int	id	PK
string	name	
string	description	
string	branch	

Student		
int	id	PK
string	name	
string	email	
string	password	
string	address	
string	qualifications	
string	branch	
string	skills	
string	resumeUrl	

Application		
int	id	PK
int	studentId	FK
int	companyId	FK
string	status	

Offer		
int	id	PK
string	company	
string	position	
string	package	
string	location	
string	joiningDate	
string	status	
int	studentId	FK

z

Schema : Includes tables like STUDENTS, COMPANIES, ELIGIBILITY, INTERVIEWS.

```
generator client {
provider = "prisma-client-js"
}

datasource db {
provider = "postgresql"
url      = env("DATABASE_URL")
}

model Student {
id            Int         @id @default(autoincrement())
name          String
email         String      @unique
password      String
address       String
qualifications String
branch        String
skills        String?
resumeUrl     String?

// Relations
applications  Application[]
offers        Offer[]
}

model Company {
z
id          Int          @id @default(autoincrement())
name        String
description  String?
branch      String

// Relations
applications Application[]
}

model Application {
id        Int      @id @default(autoincrement())
student    Student  @relation(fields: [studentId], references: [id])
studentId  Int
company    Company  @relation(fields: [companyId], references: [id])
companyId  Int
status     String   @default("pending")
}
model Offer {
id         Int      @id @default(autoincrement())
company     String
position    String
package     String
location    String
joiningDate String
status      String
student     Student  @relation(fields: [studentId], references: [id])
studentId   Int
}
```

z

Database Design:

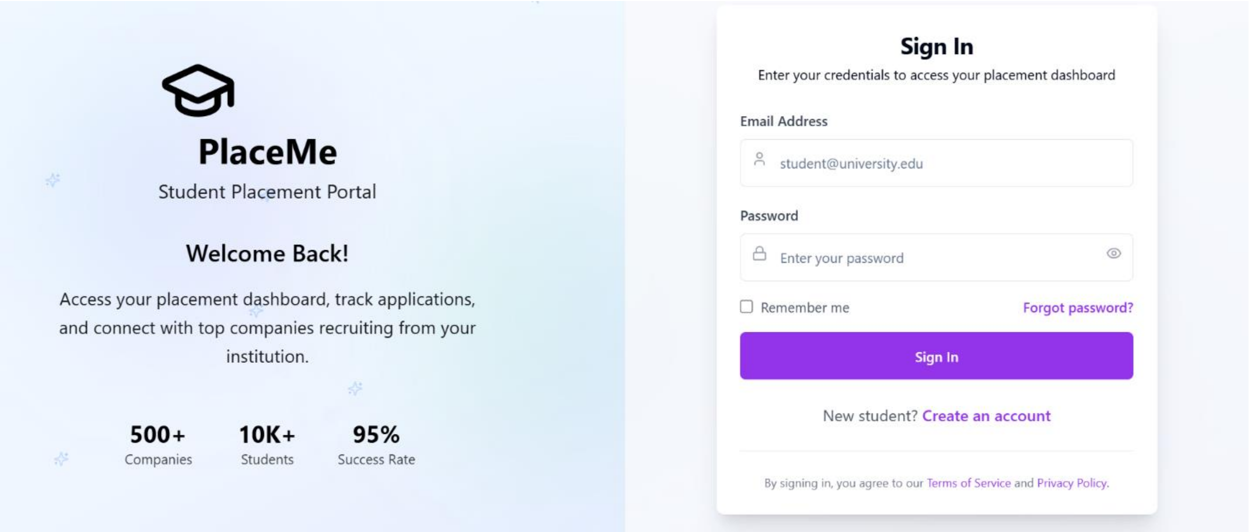
Main Tables:

- **STUDENTS** (student_id, name, branch, cgpa, email, CV)
- **COMPANIES** (company_id, name, deadline, location, logo designation, package, Status)
- **Application** (interview_id, student_id, company_id, interview_date, status)

Constraints:

- Primary Keys: student_id, company_id, interview_id
- Foreign Keys: student_id and company_id as foreign keys in application and Company table

Output Screen:



PlaceMe

Student Placement Portal

Welcome Back!

Access your placement dashboard, track applications, and connect with top companies recruiting from your institution.

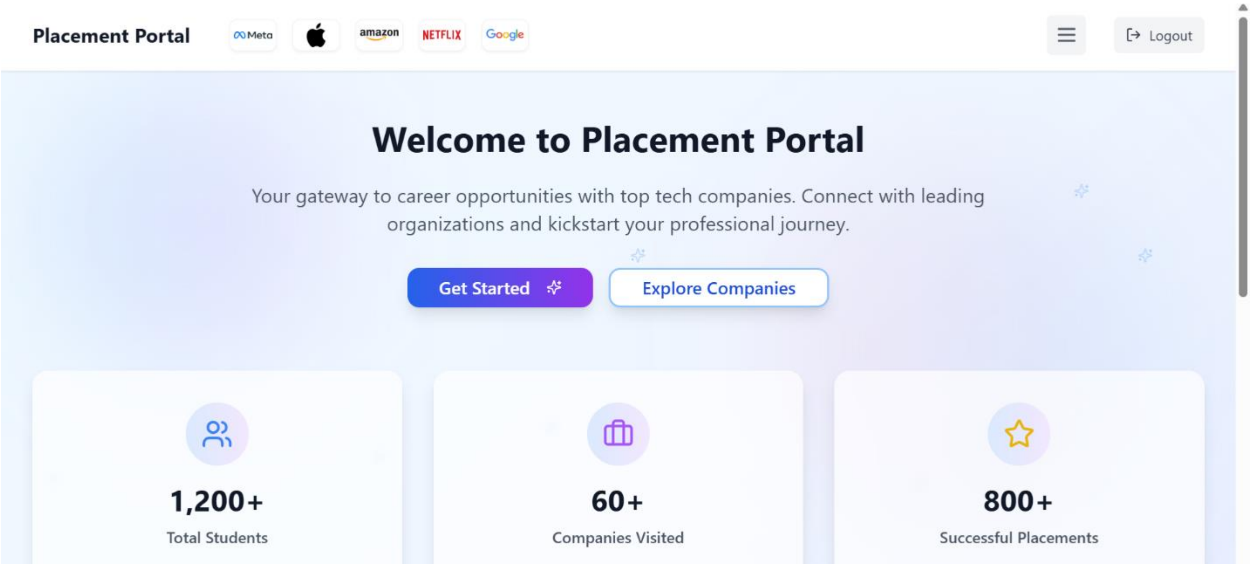
Sign In

Enter your credentials to access your placement dashboard

New student? Create an account

By signing in, you agree to our Terms of Service and Privacy Policy.

z



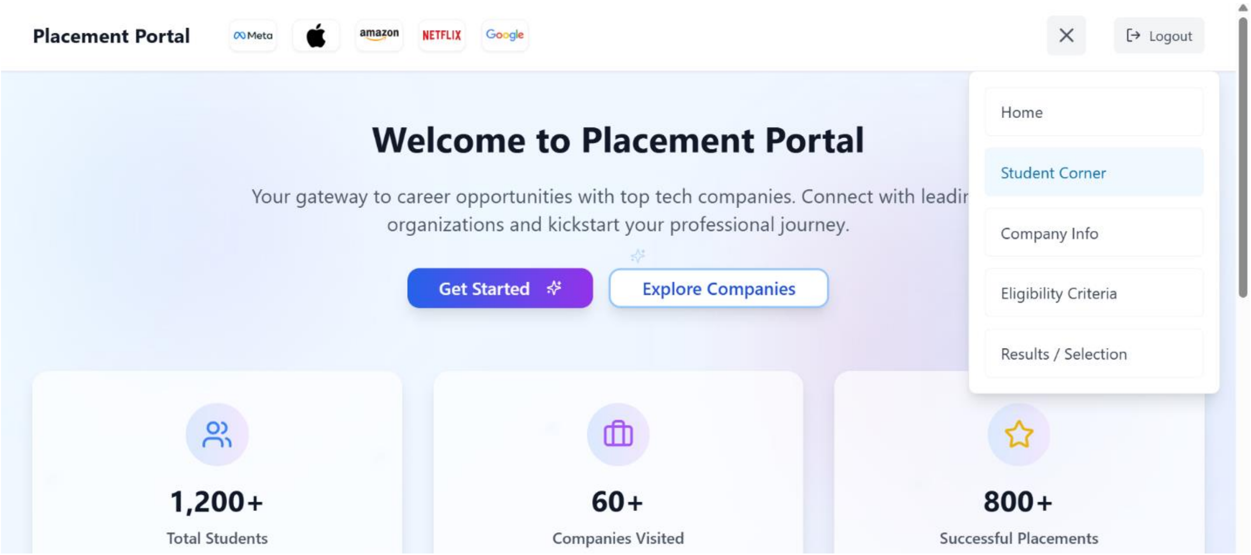
Welcome to Placement Portal

Your gateway to career opportunities with top tech companies. Connect with leading organizations and kickstart your professional journey.

1,200+ Total Students

60+ Companies Visited

800+ Successful Placements



Welcome to Placement Portal

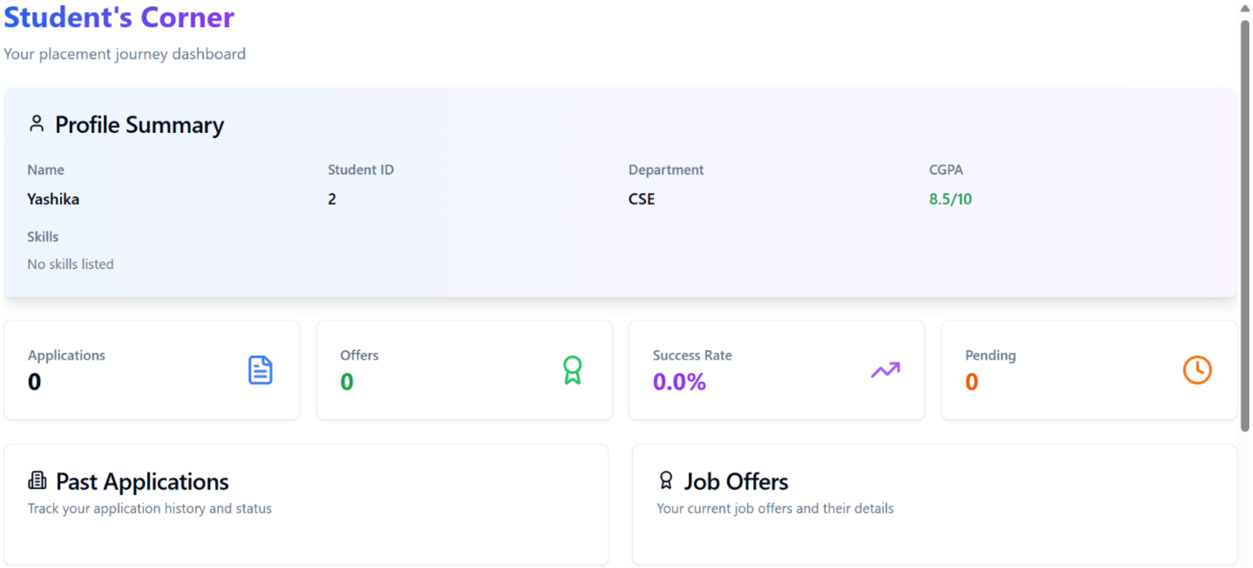
Your gateway to career opportunities with top tech companies. Connect with leadir organizations and kickstart your professional journey.

1,200+ Total Students

60+ Companies Visited

800+ Successful Placements

Z



Student's Corner

Your placement journey dashboard

Profile Summary

Name	Student ID	Department	CGPA
Yashika	2	CSE	8.5/10

Skills

No skills listed

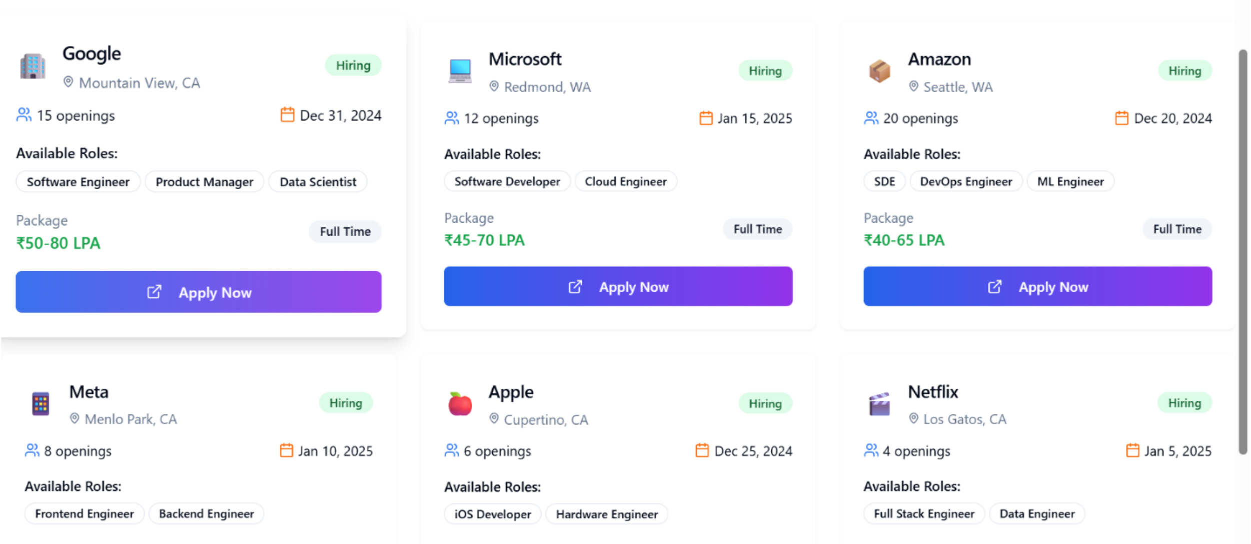
Applications	Offers	Success Rate	Pending
0	0	0.0%	0

Past Applications

Track your application history and status

Job Offers

Your current job offers and their details



Google
Mountain View, CA

15 openings
Dec 31, 2024
Available Roles:
Software Engineer, Product Manager, Data Scientist
Package
₹50-80 LPA
Full Time

Microsoft
Redmond, WA

12 openings
Jan 15, 2025
Available Roles:
Software Developer, Cloud Engineer
Package
₹45-70 LPA
Full Time

Amazon
Seattle, WA

20 openings
Dec 20, 2024
Available Roles:
SDE, DevOps Engineer, ML Engineer
Package
₹40-65 LPA
Full Time

Meta
Menlo Park, CA

8 openings
Jan 10, 2025
Available Roles:
Frontend Engineer, Backend Engineer

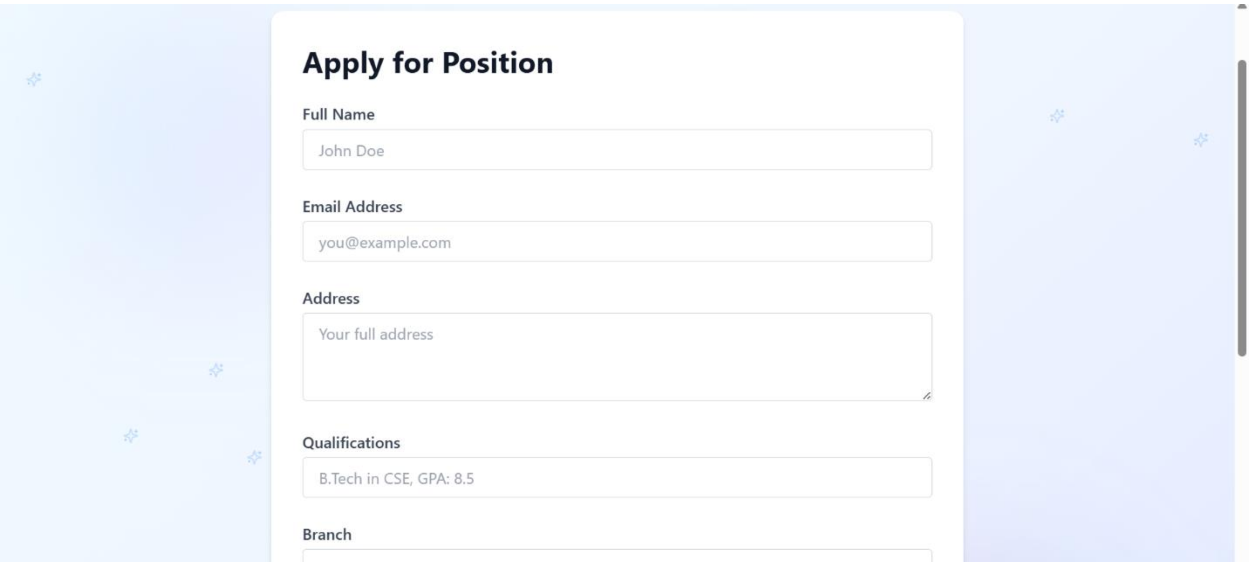
Apple
Cupertino, CA

6 openings
Dec 25, 2024
Available Roles:
iOS Developer, Hardware Engineer

Netflix
Los Gatos, CA

4 openings
Jan 5, 2025
Available Roles:
Full Stack Engineer, Data Engineer

Z



Apply for Position

Full Name

John Doe

Email Address

you@example.com

Address

Your full address

Qualifications

B.Tech in CSE, GPA: 8.5

Branch

Testing and Validation

Test Case	Description	Expected Result	Actual Result
TC1	Add new student	Student added	Passed
TC2	Student login with valid credentials	Login successful	Passed
TC3	Same student applies to multiple companies	Multiple applications recorded correctly	Passed
TC4	View number of applications by a student	Correct application count displayed	Passed

z

Bugs/Issues

- React pages connectivity
- Connecting database

Conclusion

This project successfully automates the campus placement process using Web Dev + Data Science Integration. It simplifies operations, minimizes manual errors, and ensures data consistency. During development, challenges such as managing trigger behavior and handling exceptions were encountered and resolved effectively.

Future Scope:

- Add student resume upload functionality.
- Allow companies to provide feedback post-interview.
- Integrate with email/SMS notifications for interview alerts.
- Use AI for predicting placement probabilities.
- Mobile-friendly web interface for students and companies.

Table Creation Scripts

```
CREATE TABLE Student (
   id SERIAL PRIMARY KEY,
   name TEXT NOT NULL,
   email TEXT UNIQUE NOT NULL,
   password TEXT NOT NULL,
   address TEXT NOT NULL,
   qualifications TEXT NOT NULL,
   branch TEXT NOT NULL,
   skills TEXT,
   resumeUrl TEXT
);
```

z

```
CREATE TABLE Company (
   id SERIAL PRIMARY KEY,
   name TEXT NOT NULL,
   description TEXT,
   branch TEXT NOT NULL
);

CREATE TABLE Application (
   id SERIAL PRIMARY KEY,
   studentId INT REFERENCES Student(id) ON DELETE CASCADE,
   companyId INT REFERENCES Company(id) ON DELETE CASCADE,
   status TEXT DEFAULT 'pending'
);
```

Reference

- W3Schools – SQL Tutorials
- GeeksforGeeks – DBMS Concepts
- YouTube: Telusko, ProgrammingWithMosh
- College Lecture Notes and LMS

z