CS 387: Database and Information Systems Lab

Final Report: **Placement Portal**

Saturday 30th April, 2022

Contents

1 Team members

1. Aditya Badola (190050006)
2. Akshat Gupta (190050010)
3. Liza Dahiya (190050063)
4. Tanay Sharma (190050122)

2 Github Repository

Link: https://github.com/Liza23/placement-portal.git

We have added umesh@cse.iitb.ac.in as a developer to the repository.

3 Overview: Placement Portal

We plan to build a database information system (web application) that comprehensively deals with the placement-related activities and logistics. The **Placement Portal** is a one-stop platform for students, company recruiters and placement coordinators.

The system maintains the data of all the students along with their program, department, CPI, etc. A recruiter can login on the portal, create job roles and define the eligibility criterion for that role. Henceforth, the portal also stores the data of all the firms interested in placements via the portal.

The recruiter opens the job roles (a.k.a. *JAFs*) for students to apply. Students will see the JAFs on their interface that are currently open. Students can apply to the JAF with one of their uploaded resumes if they meet the eligibility criterion for the JAF. Once the JAF closes, the recruiter shortlists the students based on various parameters including but not limited to CPI, resume, etc.

The portal allows the recruiter to add/remove the students for interviews. Once the interviews are conducted, the recruiter releases the final offers. In case of multiple offers, students are allocated one job based on their preference. Thereafter, they are not allowed to apply for any further JAFs.

Furthermore, each recruiter will be appointed a coordinator to help him/her throughout the process. Similarly, a coordinator will be assigned various firms. He/she can see the firms allotted to him/her and associated features to manage the JAFs, shortlists and selections.

There will be a Superuser (a.k.a Admin) who has access to information of all students, firms and JAFs, and can edit them. This has to be backed up by secure authentication mechanism.

The portal will show various analytics such as department-wise/program-wise student strength, profile-wise average CTC, top K popular JAFs, department-wise unplaced students, etc **selectively** to various users as mentioned in the next section.

We plan to use a relational database for the application. Relational databases are an obvious choice since there are a set of well defined entities and a set of well defined relations between the entities. Also we want the data to follow a set of constraints while being inserted/updated/retrieved from the database. Hence we use a relational database (postgres).

4 ER Diagram

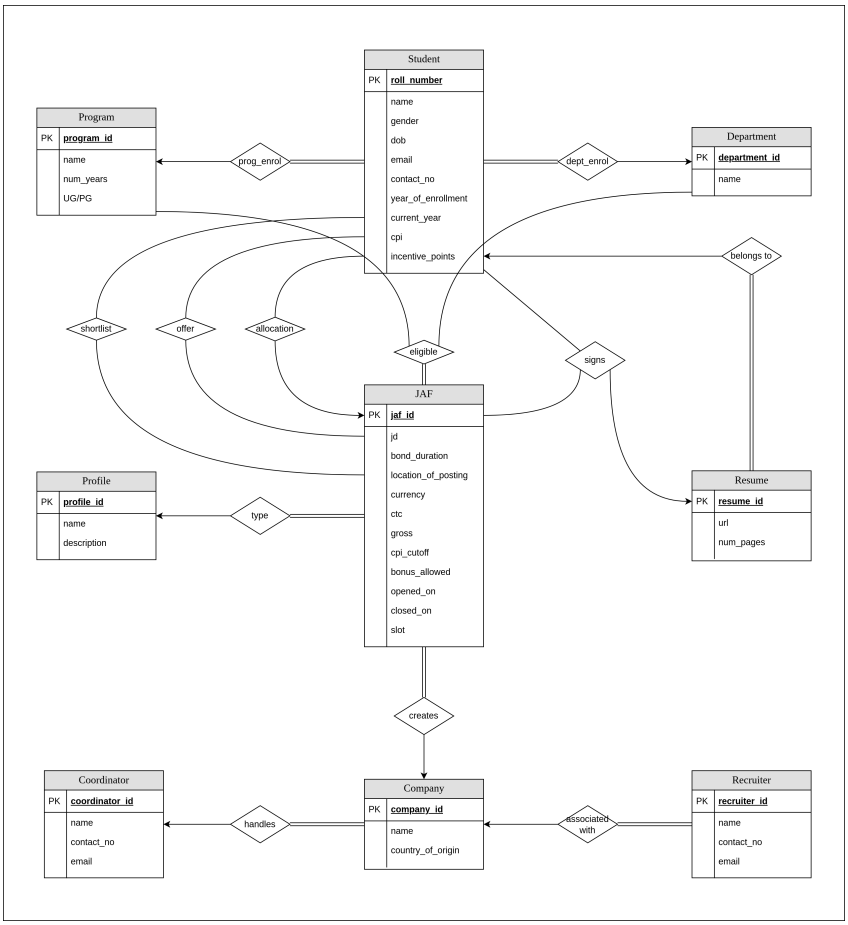

Figure 1: Entity-Relation Diagram

5 Schema Diagram

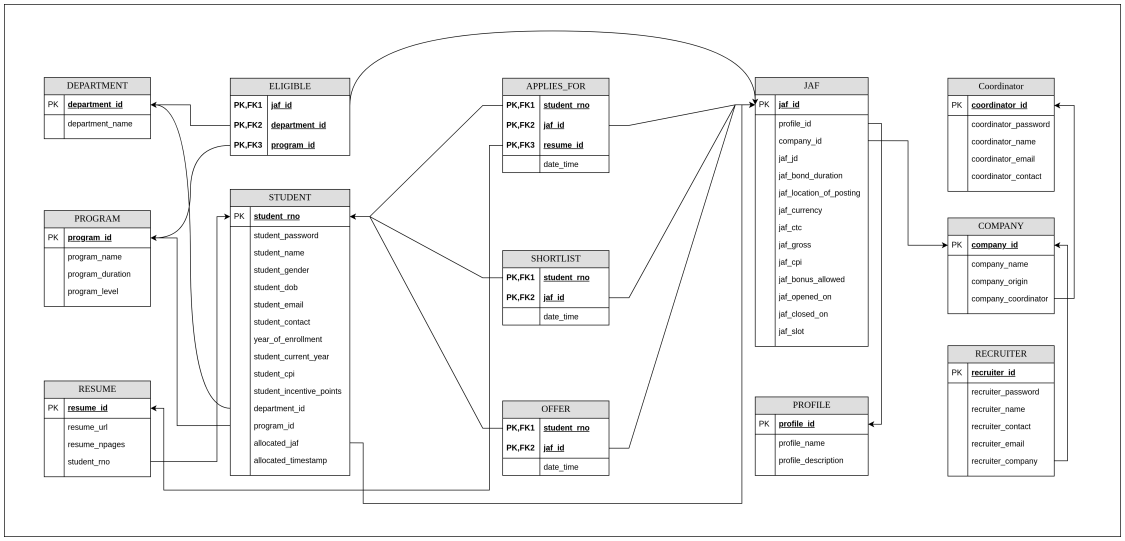

Figure 2: Schema Diagram

6 Users and Use Cases

Usecase	Implemented currently?
Student: Registering on the Portal	No
Student: Logging on to the Portal	Yes
Student: Viewing Personal Details	Yes
Student: Editing Personal Details	Yes
Student: Viewing Resume	Yes
Student: Uploading Resume	No
Student: View list of open JAFs with eligibility	Yes
Student: Apply to open and eligible JAF	Yes
Student: View list of applied JAFs with status	Yes
Recruiter/Coordinator: Registering on the Portal	No
Recruiter/Coordinator: Logging on to the Portal	Yes
Recruiter/Coordinator: View Personal Details	Yes
Recruiter: View Company Details	Yes
Recruiter/Coordinator: Edit Personal Details	Yes
Recruiter/Coordinator: Create new JAF	No
Recruiter/Coordinator: Edit a JAF	No
Recruiter/Coordinator: View list of created JAFs	Yes
Recruiter/Coordinator: View list of applicants to a JAF	No
Recruiter/Coordinator: View Student Resume	Yes
Recruiter/Coordinator: Change the status of a Student	No
Recruiter/Coordinator: Delete a JAF	Yes
Coordinator: View list Of Alloted Companies	Yes
Coordinator: Open JAF	No
Coordinator: Close JAF	No

6.1 Existence of Superuser

We did not implement Superuser and the additional privileges he/she has.

6.2 Placement Statistics

We could not implement the analytics of the placement data due to time constraints.

7 Backend Roles and Authentication

We implemented json token based authentication (jwt) for all the three roles viz. student, recruiter, coordinator. To ensure security, some api endpoints are made specific for a specific role and the users need to verify themselves (token verification) before using it.

8 How is the data generated?

We have written a python script that generates all the `.csv` files randomly. The corresponding hyper-parameters can be adjusted to specify the number of students, recruiters, etc. The script `generate_data.py` can be found in the github repo. The data is then inserted in the database, once the DDL is loaded. This is done by interpreting `insert_data.py` file.

9 What were we unable to do?

9.1 SQL Constraints

We have constrained the data to follow the primary and foreign-key constraints. The data also has some "cannot be NULL" attributes. Those constraints are also tooled in the code. However, the DDL does not incorporate the other comparisional constraints in section **3.3** of *Design Plan*.

9.2 Functionality, Routing and Data Rendering

There are still some bugs in the data rendering from backend to frontend, due to which data is not visible in many fields. This has barriered us to test the portal exhaustively.

9.3 Indexing for Query Optimization

We have not explored the effect of indexing for query processing. This is solely because we were majorly involved in debugging and functional testing, hence load testing was not the priority.