JOB SEEKING APPLICATION

UCS2201 – Fundamentals and Practice of Software Development

A PROJECT REPORT

Submitted By

KARTHIK RAJA C 3122 22 5001 055

KARTHIKEYAN.S 3122 22 5001 056

G KAVIN RAJAN 3122 22 5001 057

Department of Computer Science and Engineering

Sri Sivasubramaniya Nadar College of Engineering

(An Autonomous Institution, Affiliated to Anna University)

Kalavakkam – 603110

July 2023

Sri Sivasubramaniya Nadar College of Engineering

(An Autonomous Institution, Affiliated to Anna University)

BONAFIDE CERTIFICATE

Certified that this project report titled "**JOB SEEKING APPLICATION**" is the bonafide work of "Karthik Raja C (3122 22 5001 055), Karthikeyan.S (3122 22 5001 056), Kavin Rajan G (3122 22 5001 057)" who carried out the project work in the UCS2201 – Fundamentals and Practice of Software Development during the academic year 2022-23.

Internal Examiner External Examiner

Date:

TABLE OF CONTENTS

ABSTRACT:

It is difficult to find jobs which satisfy the job-seeker and the employer, meeting all criteria on both sides. Our job-seeking application is a platform that connects job-seekers with potential employers. It makes the job-seeking process easy and efficient. This application helps job-seekers easily access an extensive collection of job vacancies.

Our application also enables Job Providers to look for potential candidates

Finding the right candidate with desired qualifications to fill their current job openings is an important task for the recruiters of any organization. A job-seeker can register in the applications. By entering the details, skills, education, etc, he/she is recommended with the top job offers satisfying his/her constraints.

Similarly, an employer is recommended with the top job-seekers satisfying his requirements.

This helps in making the hiring process unbiased and non-discriminatory.

INTRODUCTION:

Our job-seeking application is a platform which helps job-seekers find available job offers suited to their requirements, and employers to identify the most qualified job-seekers from the vast number of potential candidates.

Traditional recruitment methods are no longer enough to acquire top talent in this technological era. They may put employers at risk of making prejudiced hiring decisions which will result in inconsistent hiring. As traditional recruitment methods rely heavily on human opinion, it creates the risk of making unconsciously biased decisions.

To help avoid this, job-seeking platforms can provide employers with a mechanism to screen and filter through job applications to ensure the risk of a biased hiring decision is significantly reduced.

A modern approach such as a job-seeking application means hiring teams and job-seekers can make more effective decisions with the assistance of technology. Platforms such as Naukri.com and LinkedIn have helped countless people find jobs, hence benefiting job-seekers and employers.

By taking advantage of these platforms, hiring managers can reduce their hiring time, lower recruitment cost, and find the best suitable candidate for the job.

PROBLEM STATEMENT

Design and develop a mediating platform that caters to the needs of both job seekers and job providers, providing a comprehensive solution for job exploration and matching. The platform should address the following challenges:

- Job seekers can explore the open positions which match their profiles.
- Talent acquisition managers can search for deserving applicants who could qualify for their job requirements.

EXTENDED EXPLORATION OF THE PROBLEM STATEMENT

There are two entities: Job Seeker (who is looking for jobs) and Job Provider/HR (who recruits suitable people for respective jobs).

The Job Seeker can look for jobs based on their preference of constraint (Job Title/Company Title/Location/Salary).

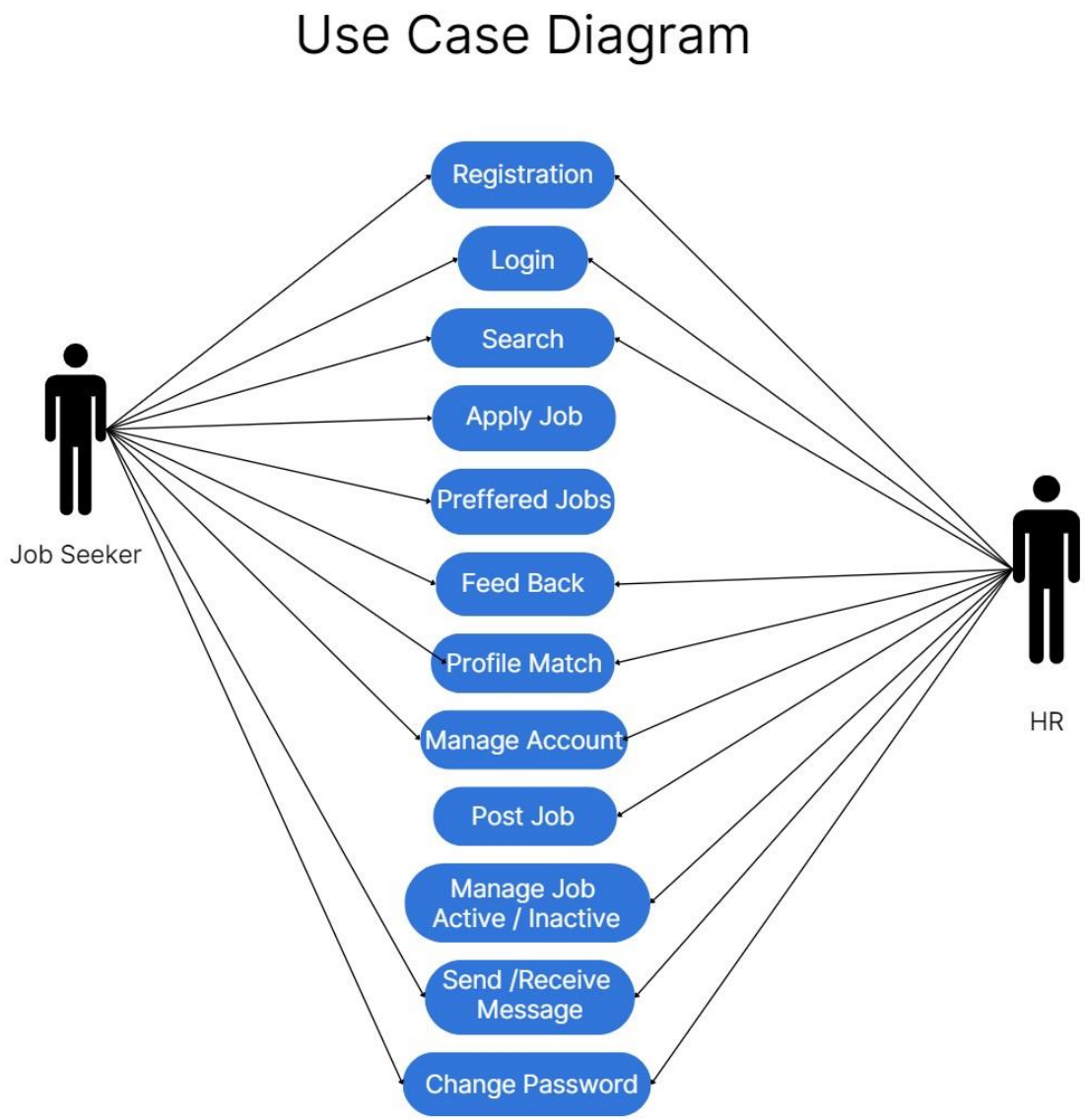
The Job Providers can look for potential candidates based on a score metric.

The Job Seeker profile consists of details like education, job preference, company preference, skills, experience, expected salary, languages known.

The available jobs include Job code, company code, job title, weights, minimum score (fixed by company), salary offered.

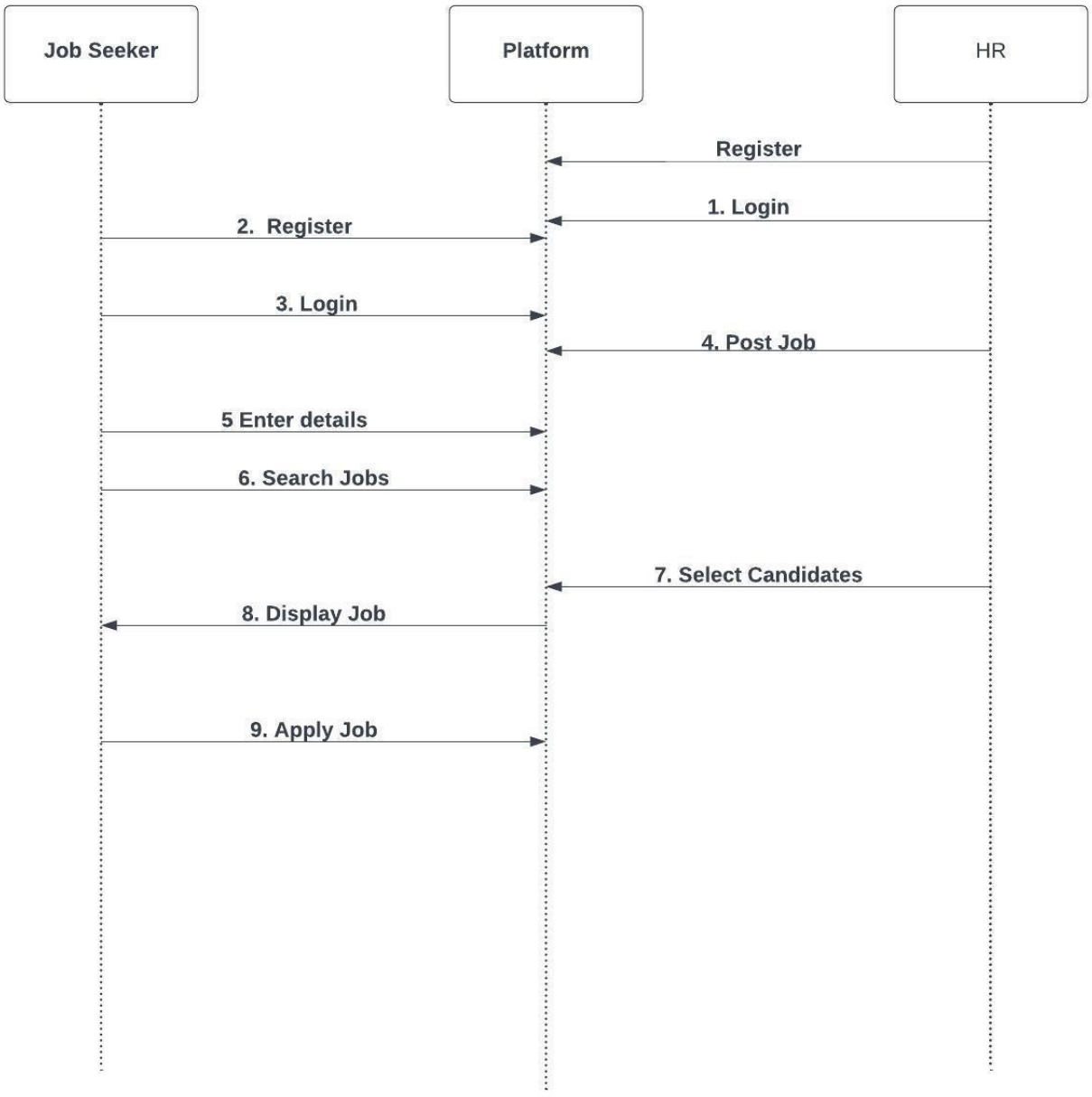
ANALYSIS:

Use Case Diagram



The use case diagram has 2 main entities: **Job Seeker and HR (Job Provider).** Job Seeker should register (sign up) first before logging in. After logging in, the job seeker should enter the necessary details. Then the job seeker can look for jobs based on either Job Preference, Company Name or Location. The job provider can also filter candidates (job seekers). The jobs which match the preference of the job seeker will be displayed and vice versa.

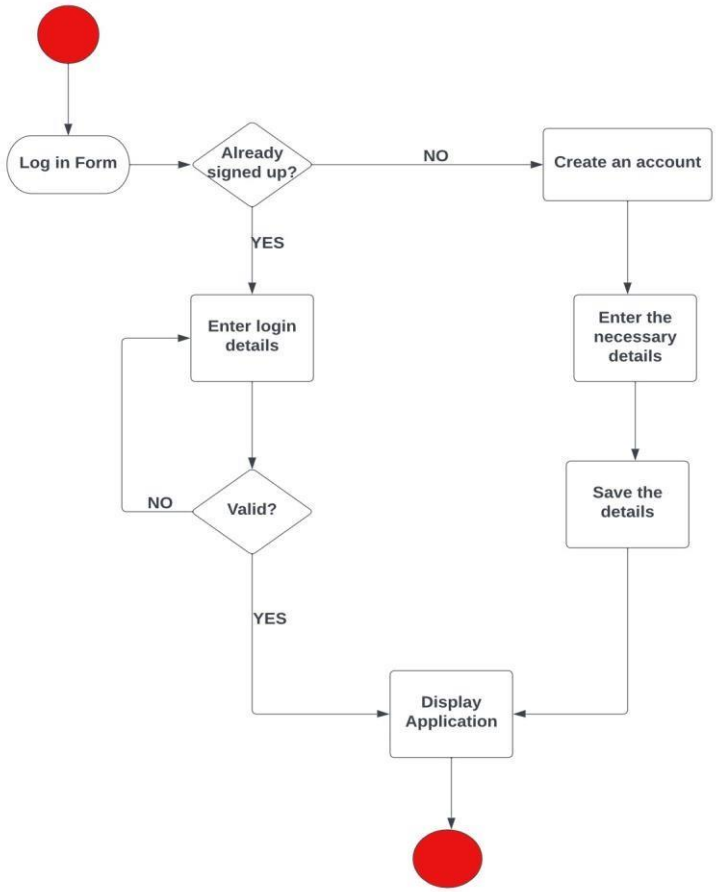
Sequence diagram:



The sequence diagram shows the sequence of steps of the job application system.
The first step involves the sign-in or login of the user/admin.
After the login, Job seekers are asked to enter necessary details and the Job providers posts the job. The Job seeker can now search for jobs which match their profile.
Job providers can look for potential candidates and then shortlist candidates based on their scores.

Activity diagram:

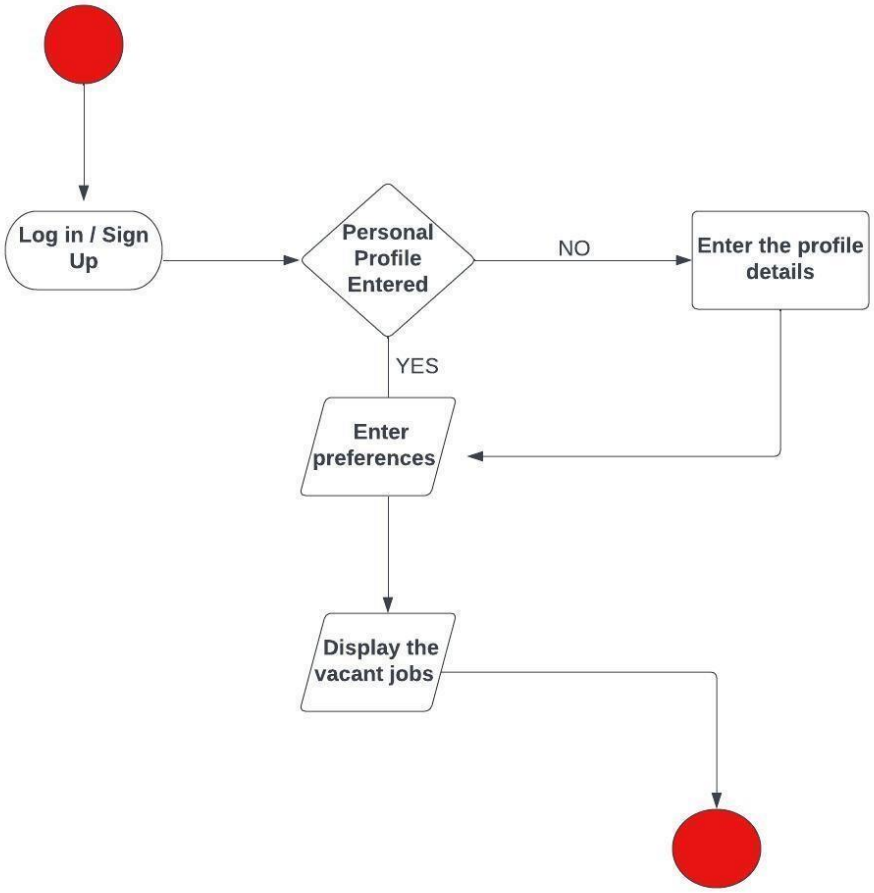
Login:



The above is the activity diagram for the login page.

The log-in form takes into consideration whether the user has an account or not. If the user has an account, the user can simply login by entering correct username and password. Else, the user is asked to enter the sign-in credentials to create an account. After the user has entered the basic details for sign up in the application, they can again run the application and user can login.

Job Seeker Information and Job Selection:

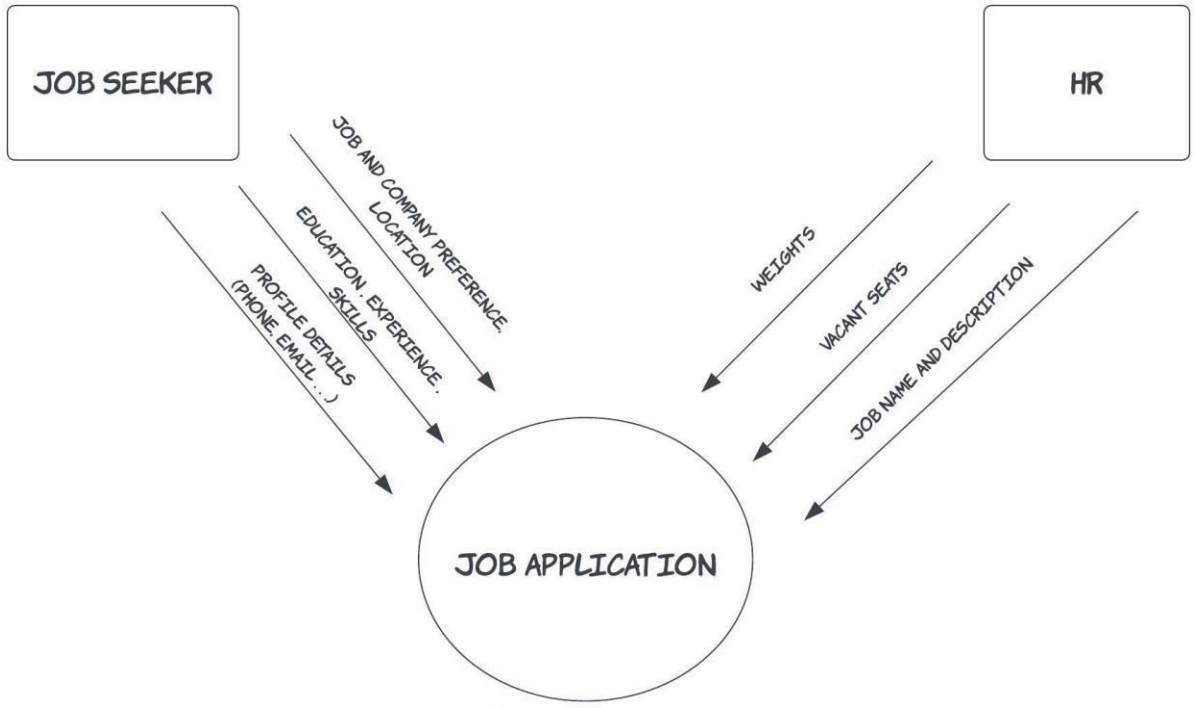


Above diagram is about job seeker information and job selection.

The Job seeker is required to enter necessary details like education, programming skills, location, expected salary, work experience etc. and various other constraints. Based on this, potential job/s are filtered out and displayed.

Data Flow Diagram

Level 0:



The above is the DFD of level 1.

Job Seeker:

(Login Details) profile details like email, phone no., address.

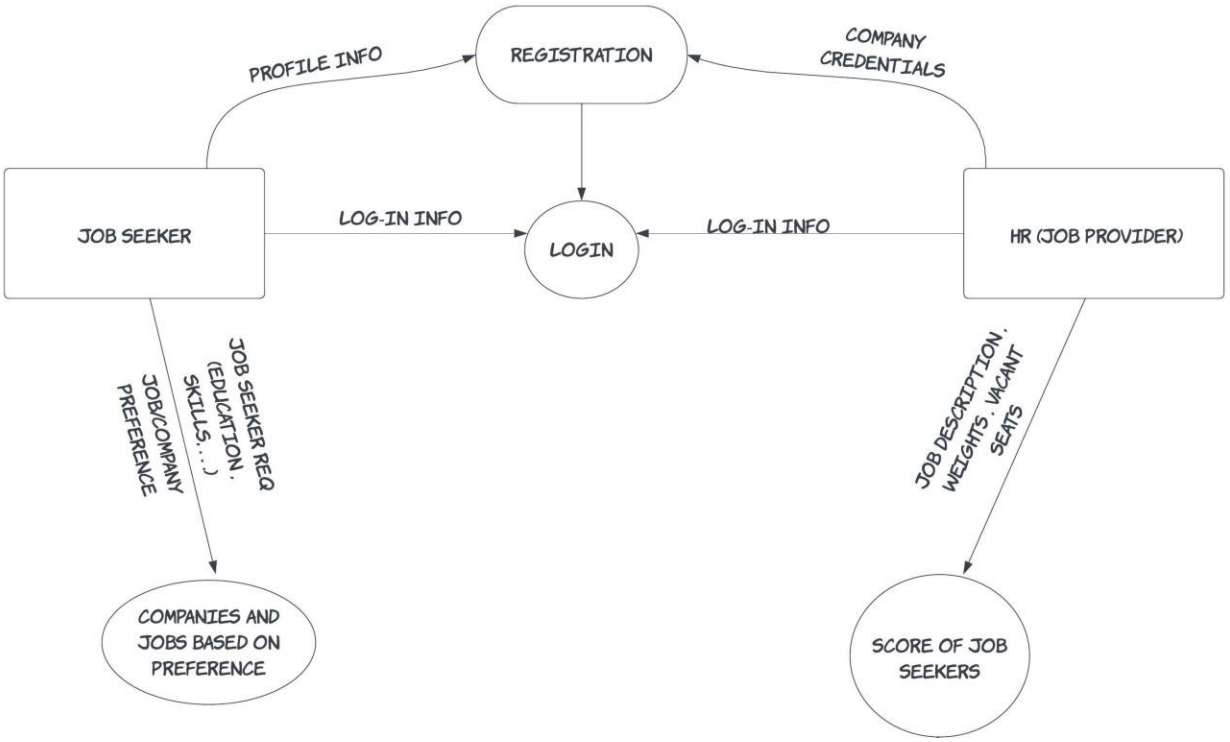
(Skill details) education, experience, skills, Job preferences, location.

Job Provider/HR

Vacant job titles and their respective codes along with number of vacant seats

Weights for each skill.

Level 1:



This the level 1data flow diagram level 1

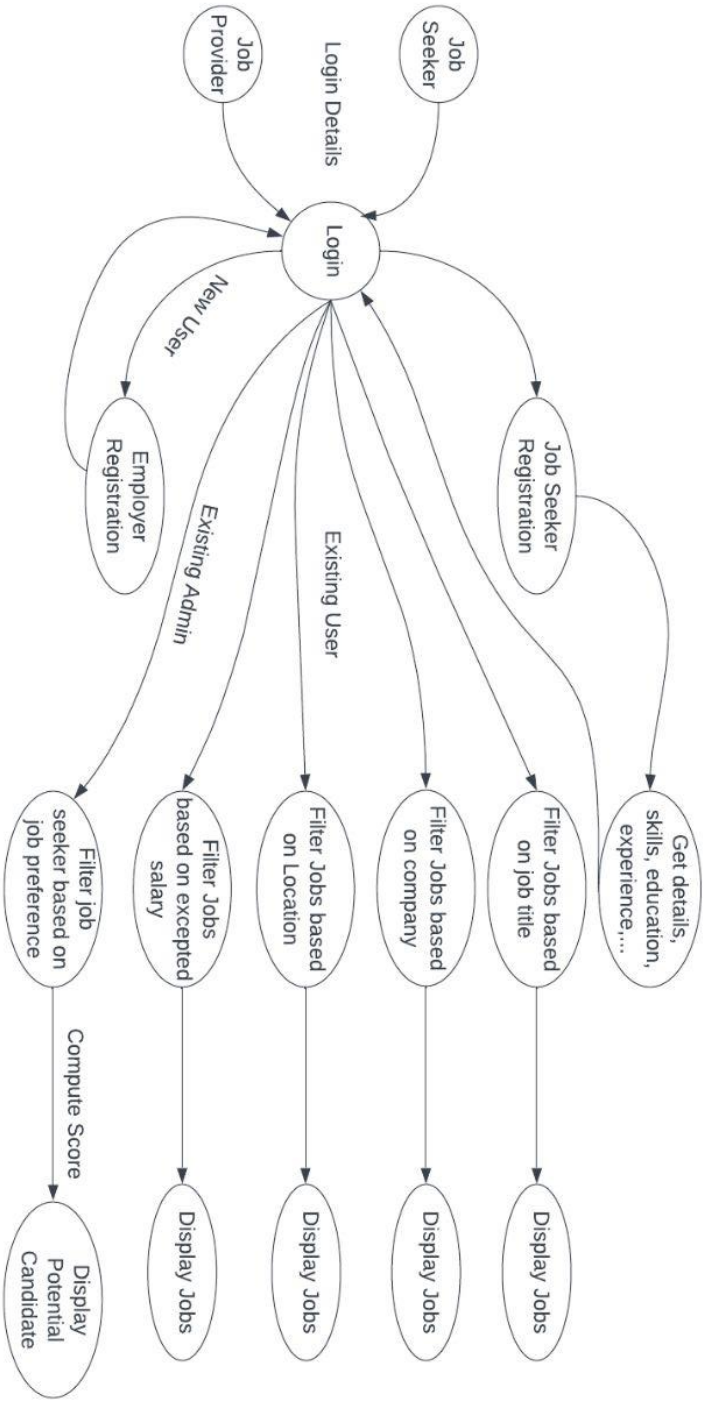
Job seeker:

Register (Sign-Up) and then login to the page.
Filter the jobs according to their constraints.

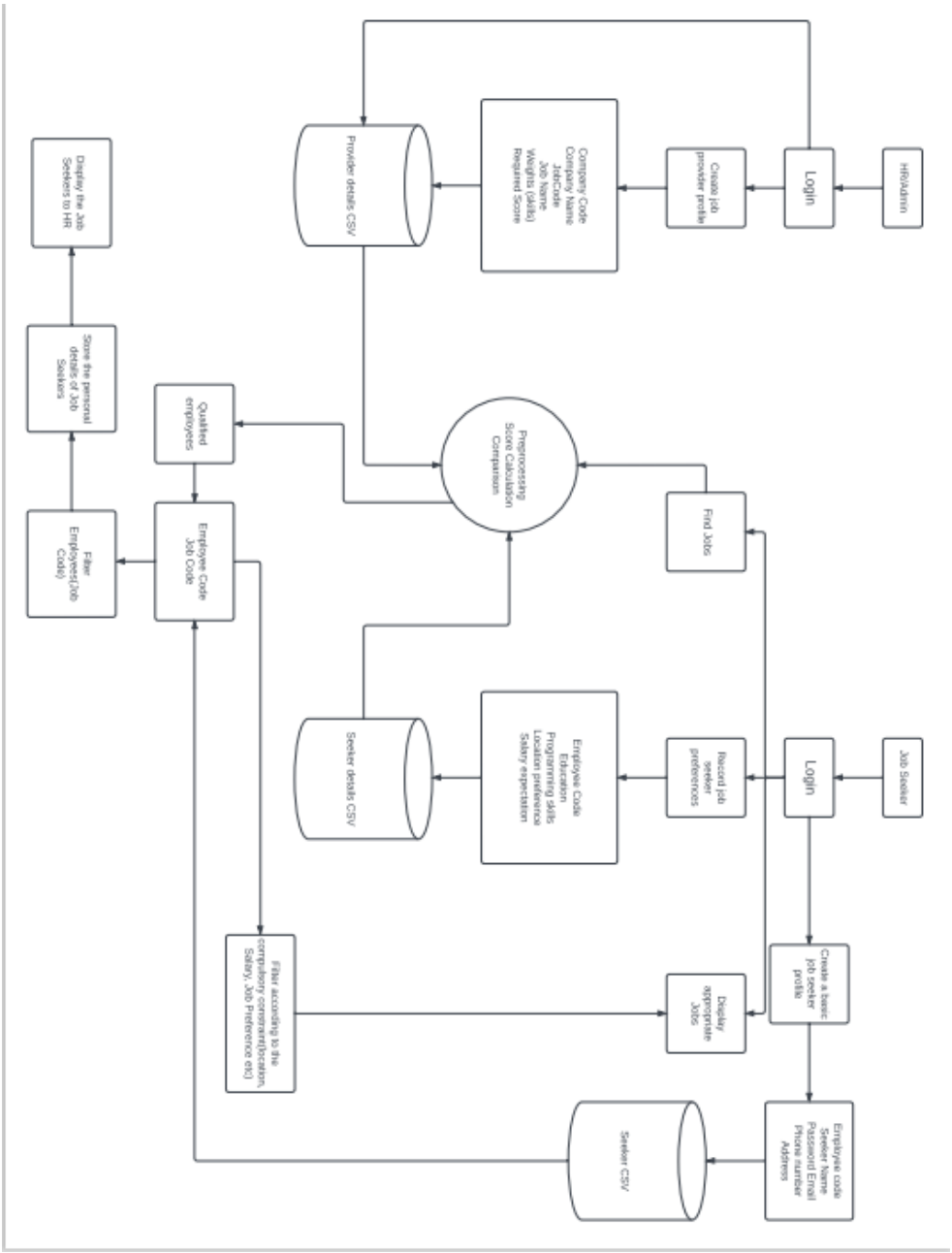
Job Provider:

Register and login with their company credentials.
List of vacant jobs and the weights for each skill.

Level 2:


Job Provider
Job Seeker
Login Details
Login
New User
Employer Registration
Existing Admin
Existing User
Job Seeker Registration
Filter job seeker based on job preference
Filter Jobs based on excepted salary
Filter Jobs based on Location
Filter Jobs based on company
Filter Jobs based on job title
Get details, skills, education, experience,...
Compute Score
Display Potential Candidate
Display Jobs
Display Jobs
Display Jobs
Display Jobs

ARCHITECTURE DIAGRAM:



STRUCTURES:

1. This structure is a template of the login/sign-up credentials of the User.

```
struct {
   int employeecode;
  char username[MAX_LENGTH];
  char password[MAX_LENGTH];
  char email[MAX_LENGTH];
  char dob[MAX_LENGTH];
  char phoneNumber[MAX_LENGTH];
  char address[MAX_LENGTH];
} User;
```

2. This structure is a template of the Company code and the password.

```
struct {
  char code[MAX_LENGTH];
  char password[MAX_LENGTH];
} Admin;
```

3. This template is used to store the usernames and their respective passwords.

```
typedef struct {
  char username[MAX_LENGTH];
  char password[MAX_LENGTH];
} user_pass;
```

4. This structure is a template of the Employee skills (Job Seeker)

```
struct Employee_all {
```

```
  int empid;
  char education[50];
  char job_preference[50];
  char company_preference[50];
  char pro_skill1[50];
  char pro_skill2[50];
  char pro_skill3[50];
  char pro_skill4[50];
  char pro_skill5[50];
  int pro_score;
  char location[50];
  int workex;
  char lang1[50];
  char lang2[50];
  char lang3[50];
  char salary[50];
};
```

5. This structure is a template of the Job Provider details

```
struct JobProvider {
  char jobcode[MAX_LENGTH];
  char companycode[MAX_LENGTH];
  char companyname[MAX_LENGTH];
  int vacantseats;
  float weight1;
  float weight2;
```

```
  float weight3;
  float weight4;
  float weight5;
  // float weight6;
  float requiredscore;
  char job_title[MAX_LENGTH];
  char location[MAX_LENGTH];
  char salary[MAX_LENGTH];
};
```

6. This is used as a template while deriving the programming languages and the corresponding score.

```
struct LanguageScore {
  char language[20];
  int score;
};
```

DESCRIPTION OF EACH MODULE:

1.struct JobProvider *filterJobProviderByPreference(const char *jobPreference, int *numMatches) ;

This module/function is used to filter the Job Providers based on the Job Preference entered by the Job Seeker.

INPUT: jobPreference , numMatches

RETURN TYPE: structure

2. struct JobProvider *filterJobProviderByLocation(const char *location, int *numMatches);

This module/function is used to filter the Job Providers based on the Location entered by the Job Seeker.

INPUT: location, numMatches

RETURN TYPE: structure

3. struct JobProvider *filterJobProviderByName(const char *name1,

int *numMatches);

This module/function is used to filter the Job Providers based on the Company Name entered by the Job Seeker.

INPUT: name1, numMatches

RETURN TYPE: structure

4. struct JobProvider *filterJobProviderBySalary(const char *salary1,

int *numMatches);

This module/function is used to filter the Job Providers based on the Expected Salary entered by the Job Seeker.

INPUT: salary1, numMatches

RETURN TYPE: structure

5. struct Employee_all *filterJobSeekerByPreference(const char *jobPreference, int *numMatches);

This module/function is used to filter the Job Seekers based on the Job Title entered by the Job Provider.

INPUT: jobPreference, numMatches

RETURN TYPE: structure

6. double calculateScore(struct Employee_all *employee,

struct JobProvider *provider);

This module/function is used in the calculation of the scores of each Job Seeker.

INPUT: structures Employee_all and JobProvider

RETURN TYPE: double

7. void saveMatchingEmployees(const char *jobCode, int employeeCode);

This module/function is used in storing of the matching employees of a respective Job.

All the employees who have scored above a threshold score (fixed by job provider) are potential candidates.

INPUT: jobCode and employeeCode

RETURN TYPE: None

8. void adminSignup();

This module/function is used in the sign-up of the Admins/HR.

The company code and the password are to be entered by the admin.

9. int adminLogin();

This module/function is used in the login of the Admin.

(1 if the login is successful, else 0)

RETURN TYPE: Integer

10. int generateUserCode();

This module/function is used in the generation of a random employee ID.

11. int validatePhoneNumber(char phoneNumber);

This module is used to validate the phoneNumber entered by the user.

VALIDATION KEYS : i) 10 digits

ii) No other charcaters

(1 if invalid length, 2 if non-digit character found and 0 if number is valid)

12. int checkUsernameExists(const char *username);

This module/function is used to ensure that the usernames entered during sign-up are unique and non-repetitive.

(0 if username is unique, else 1)

13. void writeUsersToFile(User users[], int numUsers);

This module/function is used to write the users into the file user.csv

INPUT: users and numUsers

RETURN TYPE: None

14.void writeUsersToFile1(user_pass users1[], int numUsers);

This module/function is used to store the usernames and the passwords in a file user_pass.csv

INPUT: Structure users1 and numUsers

15 void saveJobSeekerDetails(struct JobSeeker *jobSeeker);
This module/function is used to store the Job Seeker details into a file "seekerdetails.csv".

INPUT: structure jobSeeker

RETURN TYPE: None

16. void read_csv(struct LanguageScore *scores, int max_scores);

This module/function is used to read the language_score file which has the scores for each programming language.

17. void read_csv(struct LanguageScore *scores, int max_scores);

This module/function is used to read the programming langauges and the score.

INPUT: Array of structures and max_scores

IMPLEMENTATION

DATA ORGANISATION:

Structures

Structures were used as a template either for retrieving data from a file or writing data into a file. They are useful when we have multiple data attributes associated with one entity.

For example, the User structure is used as a template to hold attributes like employeecode, username, password, etc for each employee,

```
typedef struct {
  int employeecode;
  char username[MAX_LENGTH];
  char password[MAX_LENGTH];
  char email[MAX_LENGTH];
  char dob[MAX_LENGTH];
  char phoneNumber[MAX_LENGTH];
  char address[MAX_LENGTH];
} User;
```

Files

Files were used to store the data permanently. CSV files were opted because of the tabular nature of the data. They are needed when there is a need of storage of large data over a large period of time and to be retained across different program runs.

For example, “seekerdetails.csv” is used to store the Job Seeker details like education, Job preference, Company preference, skills etc.

```
1  90345,SSN,Data_Scientist,ABC_Company,Python,Java,C,SQL,HTML,83,Ch
   ennai,3,Tamil,English,NaN,32000,
2  54053,SSN,Data_Analyst,XYZ_Company,Python,C,SQL,HTML,C#,83,Chenna
   i,3,Tamil,Hindi,NaN,43000,
3  13197,SSN,Data_Scientist,ABC_Company,Python,C,NaN,NaN,NaN,33,New_
   Delhi,2,Hindi,English,NaN,43000,
4  1676,SNU,Data_Scientist,XYZ_Company,Python,C,C#,SQL,Ruby,77,New_Y
   ork,4,English,NaN,NaN,32000,
5  10336,SSN,Data_Analyst,Google,C,Python,Ruby,NaN,NaN,45,,3,Tamil,H
   indi,NaN,43000,
6  10336,SSN,Data_Scientist,Google,C,Python,NaN,NaN,NaN,33,Chennai,3
   ,Tamil,NaN,NaN,43000,
```

Arrays of structures

They combine the benefits of arrays and structures. They were used to store multiple amount of similar-template data.

For example, User structure is defined and then array of User is declared. This is stored in such a way because there are many number of users with common attributes.

```
typedef struct {
    int employeecode;
    char username[MAX_LENGTH];
    char password[MAX_LENGTH];
    char email[MAX_LENGTH];
    char dob[MAX_LENGTH];
    char phoneNumber[MAX_LENGTH];
    char address[MAX_LENGTH];
} User;
```

```
User users[MAX_USERS];
```

LIBRARIES USED:

#include <ctype.h>

The <ctype.h> header file declares a set of functions to classify (and transform) individual characters.

It defines functions that can perform operations on individual characters, such as checking if a character is an alphabetic character, converting characters to uppercase or lowercase, and testing for various character properties.

#include <stdbool.h>

The <stdbool.h> header file defines the constants true and false, which represent boolean values in C. Including this header file allows us to use boolean variables and expressions in the program.

#include <stdio.h>

The <stdio.h> header file stands for "standard input-output header" and provides various input and output functions in C. It includes function declarations that are used for performing input and output operations on files or the standard input/output streams.

#include <stdlib.h>

<stdlib.h> provides functions for dynamic memory allocation and deallocation, such as malloc, calloc, realloc, and free. These functions are used to allocate memory during runtime and manage memory resources.

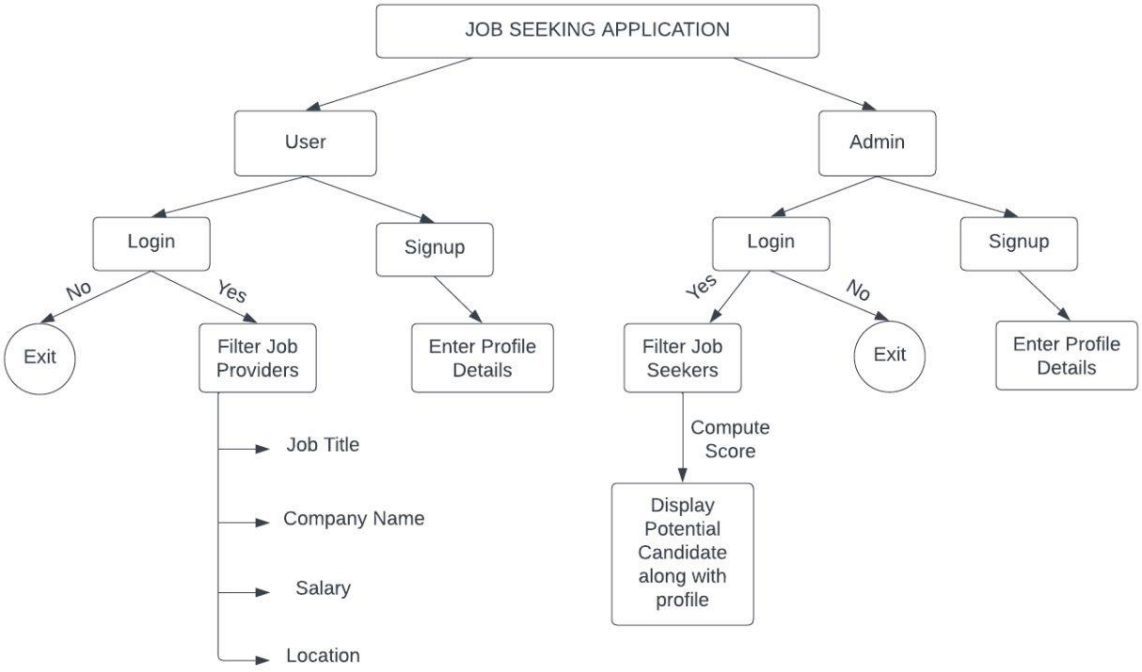
#include <string.h>

The <string.h> header file provides various functions for manipulating strings in C. It includes function declarations for string operations such as copying, concatenation, comparison, searching, and manipulation.

#include <time.h>

The <time.h> header file provides functions and structures for working with date and time in C. This file was used in the seeding of the random number generator.

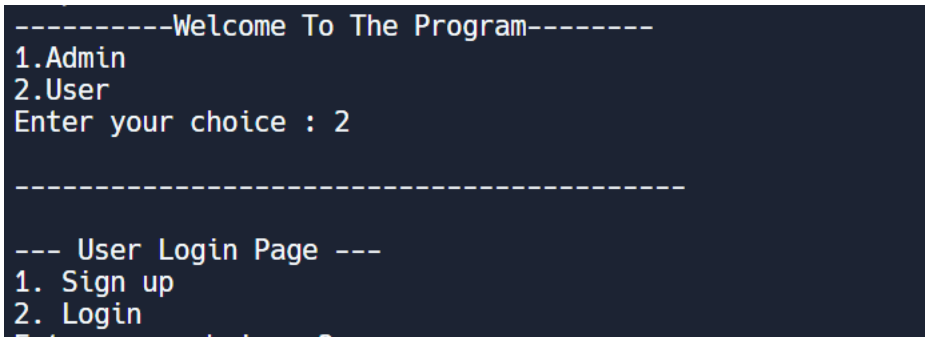
USER INTERFACE DESIGN:



Due to the limitations of working with the C programming language and the challenges involved in implementing a graphical user interface (GUI), we have opted to focus on creating an enhanced terminal-based interface.

Our goal is to provide a user-friendly experience within the terminal environment.

To achieve this, we have implemented clear and descriptive titles for each stage of the process. These titles serve to provide users with a better understanding of the current stage they are in, facilitating their navigation through the application.



Platform Used for Coding: Replit

Replit offers several advantages as a coding platform.

Firstly, it provides online accessibility, allowing users to access their code and work on projects from anywhere with an internet connection. This eliminates the need for local installations or setup, making it highly convenient and accessible.

Secondly, Replit includes collaboration features that enable multiple users to work on the same code simultaneously. This promotes real-time collaboration and seamless code sharing, making it ideal for team projects, pair programming, or coding workshops.

Additionally, Replit provides an integrated development environment (IDE) with a user-friendly interface. The IDE includes features such as code editing, debugging tools, version control, and terminal access, providing a comprehensive coding environment within a single platform.

VALIDATION THROUGH TEST CASES

I.USER:

1.SIGNUP:

If the username already exits, the program asks to re-enter a new unique username.

```
--- User Login Page ---
1. Sign up
2. Login
Enter your choice: 1
Here is your  employee id :10336
Enter your username: SK
Username already exists.Please enter a different username: hello
Username is available!
Re-Enter the username :
```

If the phone number is invalid (incorrect number of digits or presence of characters), the programs asks us to enter a new valid phone number.

```
Enter a phone number (10 digits): 9876
Invalid phone number length! Please try again.
Enter a phone number (10 digits): 9876y
Invalid phone number length! Please try again.
Enter a phone number (10 digits): yut
Invalid phone number length! Please try again.
Enter a phone number (10 digits): 9940495512
Valid phone number!
Enter your address:
```

Successful sign-up and a unique employee id!

```
--- User Login Page ---
1. Sign up
2. Login
Enter your choice: 1
Here is your  employee id :10336
Enter your username: SK
Username already exists.Please enter a different username: hello
Username is available!
Re-Enter the username : hello
Enter your password: bye
Re-Enter your password : bye
Enter your email: hello@gmail.com
Enter your date of birth (DOB): 22-01-2001
Enter a phone number (10 digits): 9876
Invalid phone number length! Please try again.
Enter a phone number (10 digits): 9876y
Invalid phone number length! Please try again.
Enter a phone number (10 digits): yut
Invalid phone number length! Please try again.
Enter a phone number (10 digits): 9940495512
Valid phone number!
Enter your address: Velachery
Sign up successful!
```

2.LOGIN:

If the username doesn’t exist (i.e username hasn’t signed up)

```
--- User Login Page ---
1. Sign up
2. Login
Enter your choice: 2
Enter your username: hello1
User not found.
```

If the username exits but password is incorrect

```
--- User Login Page ---
1. Sign up
2. Login
Enter your choice: 2
Enter your username: hello
Enter your password: 123
Incorrect password.
```

Successful Login

```
--- User Login Page ---
1. Sign up
2. Login
Enter your choice: 2
Enter your username: hello
Enter your password: bye
Login successful!
Welcome, !
==================================
```

2.ENTERING DETAILS (NECESSARY)

If the employee id doesn't exist,

```
--- Job Seeker Details ---
Enter Job Seeker details:
Employee ID: 31224
Invalid employee id
EXITING....>
```

Successful entry of data,

```
--- Job Seeker Details ---
Enter Job Seeker details:
Employee ID: 10336
Education: SSN
Job Preference: Data_Analyst
Company Preference: Google
Programming Languages (up to 5, separated by new lines):
C
Python
Ruby


Location:
Experience (in years): 3
Languages (up to 3, separated by new lines):
Tamil
Hindi

Salary: 43000
Job Seeker details saved to file.
```

3.FILTERING JOB PROVIDERS

i) Based on Job Title

```
Enter the compulsory constraint
1.Job Preference   2. Company Name   3.Location   4.Expected Sala
ry
Enter any other number to exit
Enter your choice : 1
Enter your Job Preference : Data_Analyst
------Matching Job Providers:-------
Job Code: JC002
Company Code: CC002
Company Name: XYZ_Corporation
Job Title: Data_Analyst
Location : New_Delhi

Job Code: JC010
Company Code: CC010
Company Name: XYZ_Corporation
Job Title: Data_Analyst
Location : Chennai
```

ii) Based on Company Name

```
1.Job Preference   2. Company Name   3.Location   4.Exp
ry
Enter any other number to exit
Enter your choice : 2
Enter the Company Name : ABC_Company
------Matching Job Providers:-------
Job Code: JC001
Company Code: CC001
Company Name: ABC_Company
Job Title: Software_Engineer
Location : New_York

Job Code: JC009
Company Code: CC009
Company Name: ABC_Company
Job Title: Software_Engineer
Location : New_York
```

iii) Based on Location

```
Enter the compulsory constraint
1.Job Preference   2. Company Name   3.Location   4.Expected Sala
ry
Enter any other number to exit
Enter your choice : 3
Enter the Location : New_York
-----Matching Job Providers:-----
Job Code: JC001
Company Code: CC001
Company Name: ABC_Company
Job Title: Software_Engineer
Location : New_York

Job Code: JC009
Company Code: CC009
Company Name: ABC_Company
Job Title: Software_Engineer
Location : New_York
```

iv) Based on Expected Salary of the job

```
Enter the compulsory constraint
1.Job Preference   2. Company Name   3.Location   4.Expected Sala
ry
Enter any other number to exit
Enter your choice : 4
Enter your Expected Salary : 4000
------Matching Job Providers:-------
Job Code: JC006
Company Code: CC006
Company Name: JKL_Systems
Job Title: Data_Scientist
Location : Seoul
Salary : 4000
```

v) Any other choice to exit the program

```
Enter the compulsory constraint
1.Job Preference   2. Company Name   3.Location   4.Expected Salary
Enter any other number to exit
Enter your choice : 5
```

II.ADMIN

1.SIGN-UP

```
----------------------------------------

--- Admin Login Page ---

1. Admin Signup
2. Admin Login
3. Exit
Enter your choice: 1

Admin Signup:
Company Code: CC003
Password: cc
Admin signup successful!
```

2.LOGIN

i) Invalid employee code

```
Admin Login:
Company Code: CC009
Password: sew
Admin login failed.
```

ii) Invalid Password

```
--- Admin Login Page ---

1. Admin Signup
2. Admin Login
3. Exit
Enter your choice: 2

Admin Login:
Company Code: CC003
Password: sew
Admin login failed.
```

iii) Admin login successful

```
--- Admin Login Page ---

1. Admin Signup
2. Admin Login
3. Exit
Enter your choice: 2

Admin Login:
Company Code: CC003
Password: cc
Admin login successful!
========================================================
```

3.FILTERING JOB SEEKERS

i)Displaying all the Job Seekers

```
Re-enter the Company Code :CC003
Enter the Job Preference by which you want to filter the job seekers: Data
_Scientist
-------Matching Job seekers:-------
Emp Code: 90345
Job preference: Data_Scientist
Company preference: ABC_Company
Programming Skill 1: Python
Programming Skill 2: Java
Programming Skill 3: C
Programming Skill 4: SQL
Programming Skill 5: HTML
Location: Chennai
Language1: Tamil
Language2: English
Language3: NaN
Expected Salary : 32000
```

```
Emp Code: 13197
Job preference: Data_Scientist
Company preference: ABC_Company
Programming Skill 1: Python
Programming Skill 2: C
Programming Skill 3: NaN
Programming Skill 4: NaN
Programming Skill 5: NaN
Location: New_Delhi
Language1: Hindi
Language2: English
Language3: NaN
Expected Salary : 43000

Emp Code: 1676
Job preference: Data_Scientist
Company preference: XYZ_Company
Programming Skill 1: Python
Programming Skill 2: C
Programming Skill 3: C#
Programming Skill 4: SQL
Programming Skill 5: Ruby
Location: New_York
Language1: English
Language2: NaN
Language3: NaN
Expected Salary : 32000
```

```
Emp Code: 10336
Job preference: Data_Scientist
Company preference: Google
Programming Skill 1: C
Programming Skill 2: Python
Programming Skill 3: NaN
Programming Skill 4: NaN
Programming Skill 5: NaN
Location: Chennai
Language1: Tamil
Language2: NaN
Language3: NaN
Expected Salary : 43000
```

ii)SCORING OF THE JOB SEEKERS

The profiles of the potential candidates (who have scores more than the minimum score of that company) are only displayed

```
-------SCORING--------
Score: 68.5000, Required Score: 60.0000
!!!Potential Candidate :
Company Code : CC003 ,Job Name: Data_Scientist, Employee Code: 90345

---User Details----
ID: 90345
Username: Karthik
EMAIL ID : karthik@gmail.com
DATE OF BIRTH : 22-01-2005
Address: Velachery
Phone Number: 9176319990

Score: 27.8000, Required Score: 60.0000
Employee Code : 13197 Did Not meet the minimum score

Score: 64.4000, Required Score: 60.0000
!!!Potential Candidate :
Company Code : CC003 ,Job Name: Data_Scientist, Employee Code: 1676

---User Details----
ID: 1676
Username: Priya
EMAIL ID : priya@gmail.com
DATE OF BIRTH : 13-09-1997
Address: New_York
Phone Number: 9987654432
```

```
Score: 28.5000, Required Score: 60.0000
Employee Code : 10336 Did Not meet the minimum score
```

LIMITATION OF THE SOLUTION PROVIDED

- While this can save lot of time, it may lack the personal touch that comes from direct interaction with employers.

- Not every jobs are listed in the application some may want to choose the different but that can't be able to choose.

- Job posting on this platform with basic information about the role, but not the detailed about the company's culture, work environment or team dynamics

- Online application may give way to wider pool of candidates with unqualified and unsustainable candidates applying for particular job and that increase the more time on sorting process

OBSERVATIONS:

1.SOCIETAL

- Access and inclusivity
- Diversity and representation
- Skills development and lifelong learning

2.ENVIRONMENTAL

- Reduced carbon footprint
- Remote work and commuting
- Reduced paper usage

3.LEGAL

- Data privacy and protection
- Equal opportunity and anti – discrimination
- Intellectual property rights

4.ETHICAL

- The job-seeking application does not share the username and password of any of the users elsewhere.
- The job-seeking application is accessible to all people. On no grounds will any user be barred from signing up.
- The process of recruitment is not biased, since personal details like name, religion, etc. are not considered.

LEARNING OUTCOMES:

- Through this project, we have deepened my understanding of C programming concepts, including **data types**, **control structures**, **functions**, and **pointers**.
- Working with manual **memory management** in C using pointers has given me hands-on experience in managing memory efficiently and understanding memory-related issues.
- Throughout the development process, we encountered and resolved various bugs and issues, enhancing my **debugging** and **troubleshooting skills**.
- We have learned how to **organize** my code into **functions**, **modules**, and header files, ensuring that my projects are **well-structured**, **maintainable**, and **scalable**.
- Integrating **external libraries** and APIs into my application has taught me how to leverage existing code and resources to **enhance** the **functionality**.

REFERENCES:

https://www.w3schools.com/c/

https://www.geeksforgeeks.org/c-programming-language/

Linkedin (One of the popular website to find job providers based on the required constraint)

https://www.learn-c.org/en/Dynamic_allocation

https://www.csir.res.in/intellectual-property-directorate-ipd-0

(Here in this website we utilised the data flow that was happening between servers and application through filtering process. We tried to implement the same using csv file and structure filtration).