TITLE: SKILL AND JOB RECOMMENDER DOCUMMENDATION

TEAMID:PNT2022TMID42796

1.INTRODUCTION:

Finding a new job in today's era is at the stroke of a pen. However, finding relevant jobs based on one's existing profile/area of interest/skill set is going to be a bit tricky. People will keep changing their careers from one field to another, but it can be difficult to find that one ideal position.

Although there are many career opportunities outside, being able to see suggestions based on one's preferences will ease the job search process. An individual's expectations will change on a regular basis. One might have preferences to start their career in a Start-up, take up a work from home, or choose a job based on their unique skill set. The recommendation system will serve the purpose of refining and showing job profiles based on one's ever-changing interests/profile/skill set.

1.1 Project Overview:

The personalized recommender system is proposed to solve the problem of information overload and widely applied in many domains. The job recommender systems for job recruiting domain have emerged and enjoyed explosive growth in the last decades. User profiles and recommendation technologies in the job recommender system have gained attention and investigated in academia and implemented for some application cases in industries. In this paper, we introduce some basic concepts of user profile and some common recommendation technologies based on the existing research. Finally, we survey some typical job recommender systems which have been achieved and have a general comprehension of job recommender systems

1.2 Purpose:

our goal is to recommend the most relevant jobs for a user. Mathematically, the relevance can be the silimiarity of two skillsets (user's skills and job's skills). The most naive idea, is to count the number of skills appear in both sets.

❑ Solve complex problems in a way that fits the state of your customers.

❑ Succeed faster and increase your solution adoption by tapping into existing mediums and channels of behavior.

❑ Sharpen your communication and marketing strategy with the right triggers and messaging.

❑ Increase touch-points with your company by finding the right problem-behavior fit and building trust by solving frequent annoyances, or urgent or costly problems.

❑ Understand the existing situation in order to improve it for your target group

2. LITERATURE SURVEY:

A literature survey is a guide that helps a researcher to find, identify and define a problem. This is the survey of the various reports, books, journals, articles that are related to your project work, which helps in the justification of your work. Here are a few survey templates that are available which you can use as a framework for your report.

2.1 Existing problem:

 The project aims at building an application of skill recommender solution through which the fresher or the skilled person can log in and find the jobs by using the search option or they can directly interact with the chatbot and get their dream job.

 To develop an end-to-end web application capable of displaying the current job openings based on the user skillset. The user and their information are stored in the Database. An alert is sent when there is an opening based on the user skillset. Users will interact with the chatbot and can get the recommendations based on their skills. We can use a job search API to get the current job openings in the market which will fetch the data directly from the webpage.

2.2 References:

Journal 1:

Authors: W.Shalaby, B. AlAila, M. Korayem, L. Pournajaf, K.

AlJadda, S. Quinn

Title: Help Me Find a Job :A Graph-basedApproach for Job

Recommendation at Scale

Publisher: IEEE

Year of Publication : December 2017

DOI: 10.1109/BigData.2017.8258088

Abstract : Existing systems are mostly focused on content analysis of resumes and job descriptions, relying heavily on the accuracy and coverage of the semantic analysis and modeling of the content inwhich case, they end up usually suffering from rigidity and lack of implicit semantic relations that are uncovered from users behaviour and could be captured by Collaborative Filtering methods (CF).

Critical view :

 CareerBuilder serves job seekers in more than 24 countries with different spoken languages.

 Extending the content-based deep learning matcher to languages other than English in order to effectively bring the GBR to serve non_english speaking countries is a priority.

 Deep learning approaches pave the way towards language agnostic NLP tools, so looking to train more models to capture the similarity between job postings for different languages.

Journal 2:

Authors: G. Domeniconi, G. Moro, A. Pagliarani, K. Pasini and R.

Pasolini

Title: Job Recommendation From Semantic Similarity of LinkedIn

User’s Skills

Publisher: SCITEPRESS - Science and Technology Publications,

LdaSetubal, Portugal

Year of Publication: July 2016

DOI: 10.5220/0005702302700277

Abstract: Until recently job seeking has been a tricky, tedious and time consuming process, because people looking for a new position had to collect information from many different sources.Semantic associations arise by applying Latent Semantic Analysis. They use the mined semantics to obtain a hierarchical clustering of job positions and to build a job recommendation system.

Critical view :

□ To increase accuracy of recommendation, for example by testing other machine learning methods such as nearest neighbour classifiers are even exploiting the generated hierarchy.

□ The vector representations of profiles, skills and positions could possibly be improved, for example by borrowing suitable weighting schemes from text categorization.

Journal 3:

Authors: Miao Jiang, Yi Fang, Huangming Xie and Jike Chong

Title: User click prediction for personalized job recommendation

Publisher: Kluwer Academic Publishers United States

Year of Publication: April 2018

DOI: https://doi.org/10.1007/s11280-018-0568-z

Abstract : Major job search engines aggregate tens of millions of job postings online to enable job seekers to find valuable employment opportunities. Predicting the probability that a given user clicks on jobs is crucial to job search engines as the predictions can be used to provide personalized job recommendations for job seekers.

Critical view :

□ Beyond logistic regression and use other L2R models as the prediction model in the proposed clustering-prediction process.

□ Consequently, the proposed models would become pair wise or list wise PCP instead of point wise PCP presented in this paper.

□ The proposed approach can also be applied to other recommendation tasks in which the data presents multimodality behaviors.

□ Another research direction is to develop more personalization features such as dwell time, query reformulation, and the sequence of clicks, which can be aggregated and exploited in personalized job recommendation.

2.3 Problem Statement Definition:

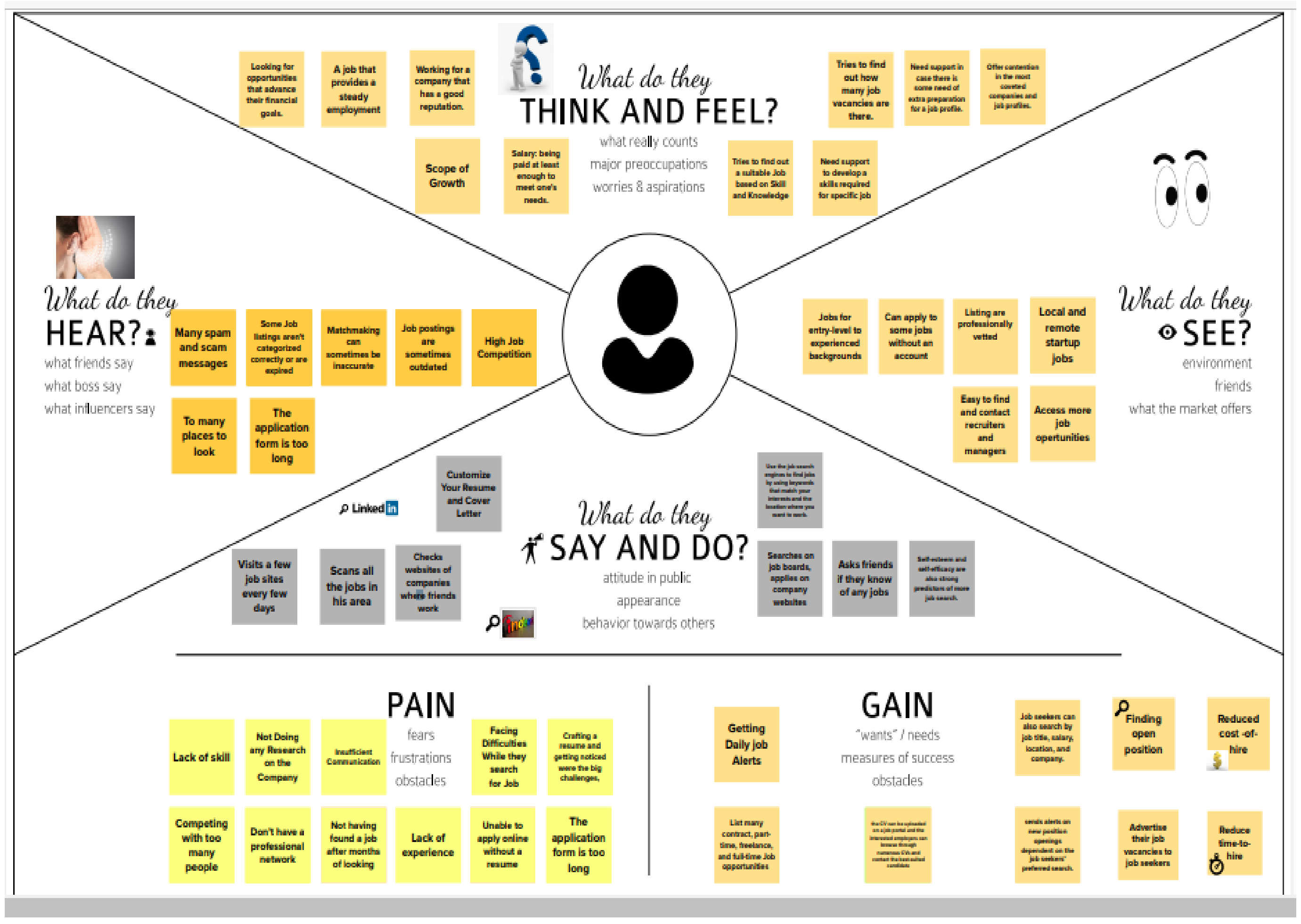
□The job seeker needs a way to efficiently search for a job that suits their skills and lies in their domain of interest by utilizing the accurate search and personalized recommendation of the proposed system.

□The job recruiters need a way to find the most eligible candidates for the offered role. The system helps the recruiter choose suitable candidates for the opening by resume parsing. Resume parsing allows the recruiter to selectcandidates whose skills match the required skills, making the recruiting process easier and quicker.

3. IDEATION & PROPOSED SOLUTION:

Ideation is the third and important step in the process of design thinking. Design Thinking is not the kingdom of designers. It's a systematic process to empathize with human problems and design the right solutions to them. Ideation is about challenging assumptions and exploring hidden territories. It helps to learn user problems and come up with new solutions. If ideation is carried out correctly, you never know which idea becomes the next innovatory solution. It's about going beyond the norm ideas and solutions. It's to explore new perspectives, diversifying ideas to innovate in a never before seen, and novel way.

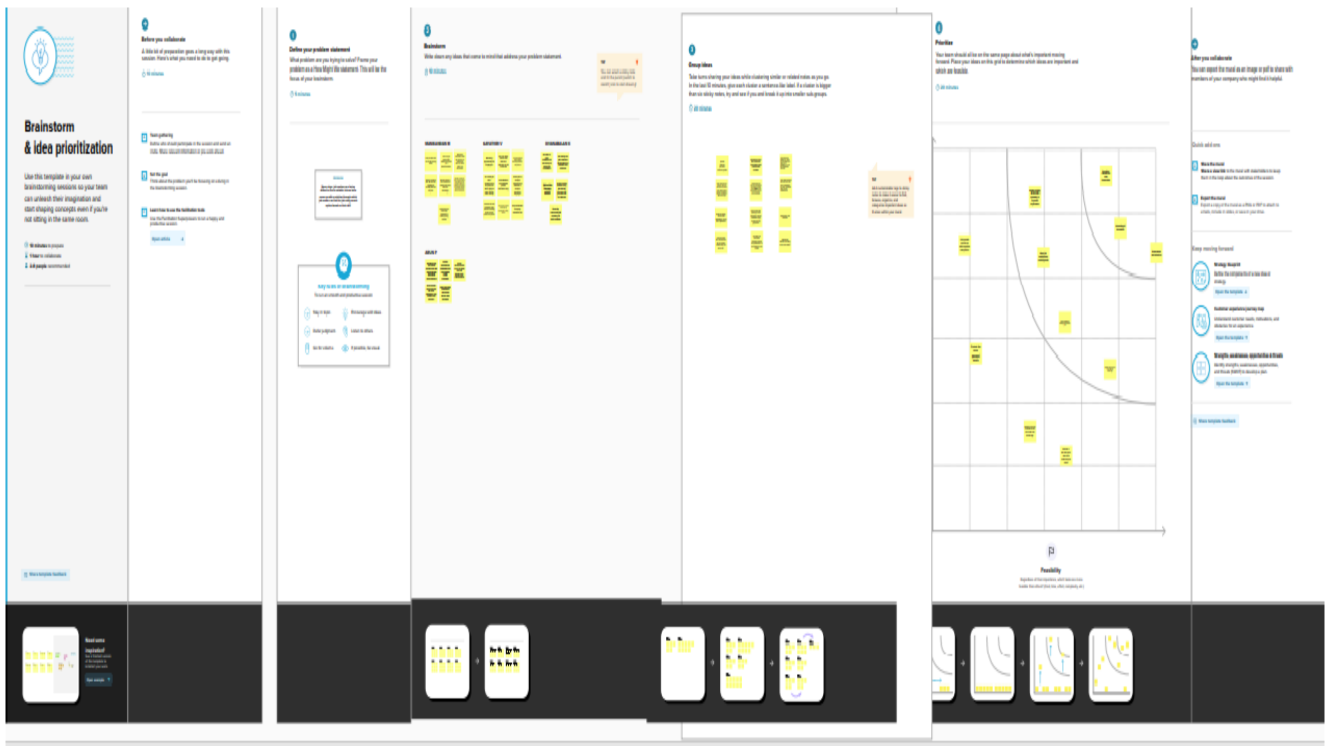
3.1 Empathy Map Canvas:



What do they THINK AND FEEL?
what really counts
major preoccupations
worries & aspirations
Looking for opportunities that advance their financial goals.
A job that provides a steady employment
Working for a company that has a good reputation.
Scope of Growth
Salary: being paid at least enough to meet one's needs.
Tries to find out how many job vacancies are there.
Need support in case there is some need of extra preparation for a job profile.
Offer contention in the most coveted companies and job profiles.
Tries to find out a suitable Job based on Skill and Knowledge
Need support to develop a skills required for specific job
What do they HEAR?
what friends say
what boss say
what influencers say
Many spam and scam messages
Some Job listings aren't categorized correctly or are expired
Matchmaking can sometimes be inaccurate
Job postings are sometimes outdated
High Job Competition
To many places to look
The application form is too long
What do they SEE?
environment
friends
what the market offers
Jobs for entry-level to experienced backgrounds
Can apply to some jobs without an account
Listing are professionally vetted
Local and remote startup jobs
Easy to find and contact recruiters and managers
Access more job opertunities
What do they SAY AND DO?
attitude in public
appearance
behavior towards others
Linkedin
Customize Your Resume and Cover Letter
Visits a few job sites every few days
Scans all the jobs in his area
Checks websites of companies where friends work
Use the job search engines to find jobs by using keywords that match your interests and the location where you want to work.
Searches on job boards, applies on company websites
Asks friends if they know of any jobs
Self-esteem and self-efficacy are also strong predictors of more job search.
PAIN
fears
frustrations
obstacles
Lack of skill
Not Doing any Research on the Company
Insufficient Communication
Facing Difficulties While they search for Job
Crafting a resume and getting noticed were the big challenges.
Competing with too many people
Don't have a professional network
Not having found a job after months of looking
Lack of experience
Unable to apply online without a resume
The application form is too long
GAIN
"wants" / needs
measures of success
obstacles
Getting Daily job Alerts
Job seekers can also search by job title, salary, location, and company.
Finding open position
Reduced cost -of- hire
List many contract, part-time, freelance, and full-time Job opportunities
sends alerts on new position openings dependent on the job seekers' preferred search.
Advertise their job vacancies to job seekers
Reduce time-to-hire

3.2 Ideation & Brainstorming:

Brainstorming is a group creativity technique by which efforts are made to find a conclusion for a specific problem by gathering a list of ideas spontaneously contributed by its members. In other words, brainstorming is a situation where a group of people meet to generate new ideas and solutions around a specific domain of interest by removing inhibitions. People are able to think more freely and they suggest as many spontaneous new ideas as possible. All the ideas are noted down without criticism and after the brainstorming session the ideas are evaluated.



3.3 Proposed Solution:

• We provide an alert based on their recent skill set and their preference.

• It reduces the loss of time.

• We provide a chatbot to solve communication problem.

• We provide enough information in the job portal

We have come up with a solution. In order to resolve this issue:

• We provide a job notification alert from the reputed compan

3.4 Problem Solution fit:

Problem:

Unemployment gives rise to poverty, causing a decrease in production and less consumption of goods and services, contributing to the nation's economic loss. Every industry has a lot of career opportunities, but job seekers are unaware of them. The unemployability crisis can be solved if every job seeker receives the right career guidance and proper job role training. On the other hand, recruiters are finding a way to make the hiring procedure easier for choosing potential candidates. Job recruiters also search for a medium to reach out to many job seekers to promote their firm's name. So, to eradicate the unemployment crisis, for the job seekers to find a job they desire, match their qualifications and skills, train themselves for their expected job roles and help the job recruiters find the perfect candidates, we need to develop a skill and job recommendation engine.

solution:

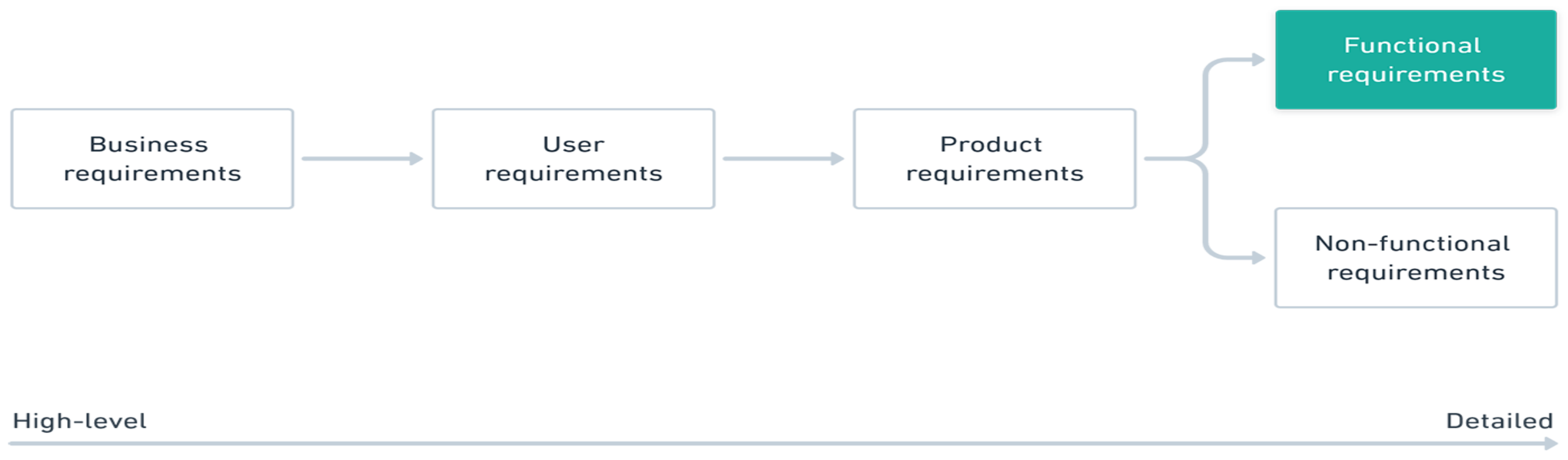
The skills (basic features) are extracted from the job seeker’s resume using the TF-IDF technique.The job seeker’s profile may get outdated sometimes as they fail to update the resume regularly. The dynamic behaviour of the job seeker is noted by observing the jobs he applied for. So, the dynamic features are extracted,which are an updated version of basic features,by making a statistic at regular intervals. The dynamic recommendation engine works as follows: A collaborative user-based filtering algorithm is used initially to overcome the cold-start problem. It takes the features extracted from the job seeker’s profile and the features extracted from the job description,computes the similarity between the two using Euclidean distance, and recommends the top similar jobs applied to generate the initial recommendation jobs. The system provides the initial recommendation to the job seeker and records his behaviour. Thus, we will be able to arrive at a set of jobs in which the job seeker is interested and a set of jobs in which he is not interested. The extended new basic features help in updating the job seeker’s profile. Thus, the job applicant is provided with new recommendations. Similarly, the same recommendation system helps provide job applicant recommendations to the job recruiters to find the most eligible candidates for their firm. Training programmes and certification courses are also recommended to job seekers based on their job interests to grow their skills.

4.REQUIREMENT ANALYSIS:

Requirements analysis or requirements engineering is a process used to determine the needs and expectations of a new product. It involves frequent communication with the stakeholders and end-users of the product to define expectations, resolve conflicts, and document all the key requirements.

The results of the requirements elicitation and the analysis activities are documented in the Requirements Analysis Document (RAD). This document completely describes the system in terms of functional and nonfunctional requirements and serves as a contractual basis between the customer and the developer. The RAD must be written in the language of the customer's domain of business/expertise. Under no circumstances should any "computerese" terminology creep into this document.

The audience for the RAD includes the customer, the users, the project management, the system analysts (i.e., the developers who participate in the requirements), and the system designers (i.e., the developers who participate in the system design). The first part of the document, including use cases and nonfunctional requirements, is written during requirement elicitation. Even for a relatively small system, you are likely to end up with scores of requirements. To understand how they relate to each other, and to effectively deal with them later on in the process, it is necessary to sepa-rate them into categories, logically grouping the requirements according to related business functions or organizational boundaries. Finally the requirement analysis subdivided to functional and non



4.1 Functional Requirement:

Functional requirementsare product features that developers must implement to enable the users to achieve their goals. They define the basic system behavior under specific conditions. Functional requirements should not be confused with other types of requirements in product management.

<u>Functional requirements:</u>

The system must allow users to log into their account by entering their email and password.

The system must allow users to log in with their Google accounts.

The system must allow users to reset their password by clicking on "I forgot my password" and receiving a link to their verified email address.

Functional Requirements:

<u>Following are the functional requirements of the proposed solution.</u>

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)

FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIN
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Chat Bot	A Chat Bot will be there in website to solve user queriesand problems related to applying a job, search for a job and much more
FR-4	User Login	Login through Form Login through Gmail
FR-5	User Profile	Updation of the user profile through the login credentials
FR-6	User Search	Exploration of Jobs based on job filters and skill recommendations.

4.2 Non-Functional requirements:

Users must change the initially assigned login password immediately after the first successful login. ...

Employees never allowed to update their salary information. ...

Every unsuccessful attempt by a user to access an item of data shall be recorded on an audit trail.

A website should be capable enough to handle 20 million users with affecting its performance

The software should be portable.

Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	This application can be used by the job seekers to login and search for the job based on her Skills set.
NFR-2	**Security**	This application is secure with separate login for Job Seekers as well as Job Recruiters
NFR-3	**Reliability**	This application is open-source and feel free to use,without need to pay anything. The enormous job openings will be provided to all the job seekers without any limitation
NFR-4	**Performance**	The performance of this application is quicker response and takes lesser time to do any process.
NFR-5	**Availability**	This application provides job offers and recommends Skills for a Particular Job opening
NFR-6	**Scalability**	The Response time of the application is quite faster compared to any other application.

5. PROJECT DESIGN:

Project design is a major first step toward a successful project. A project design is a strategic organization of ideas, materials and processes for the purpose of achieving a goal. Project managers rely on a good design to avoid pitfalls and provide parameters to maintain crucial aspects of the project, like the schedule and the budget.

5.1 Data Flow Diagrams:

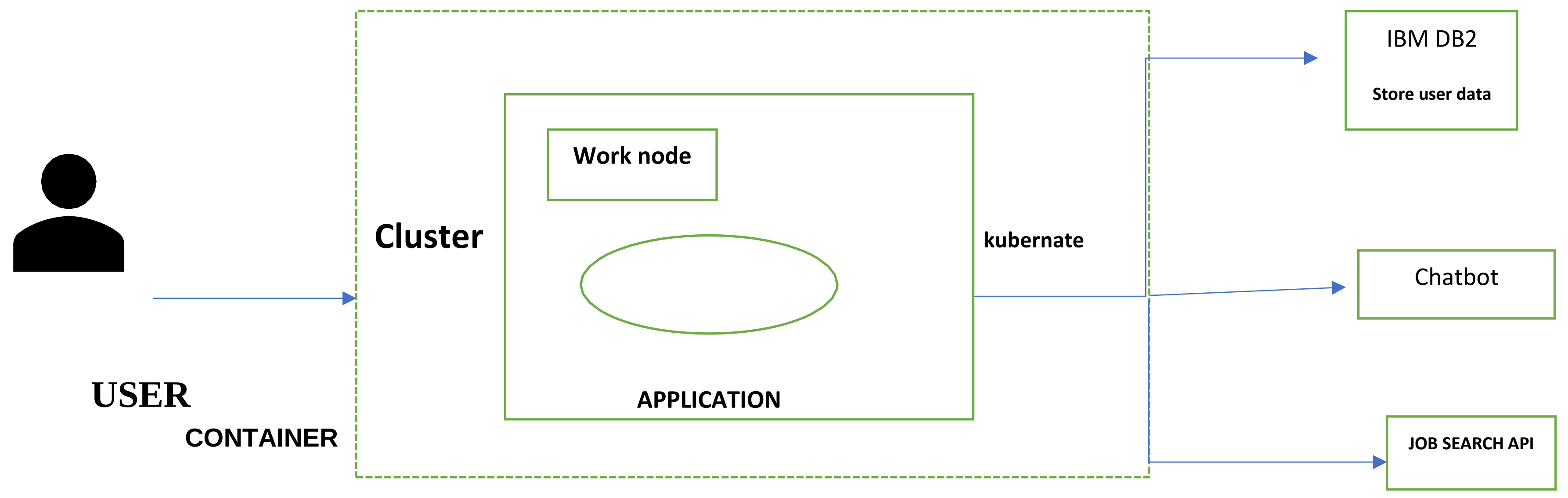
Online Job Portal Data flow diagram is often used as a preliminary step to create an overview of the Job Portal without going into great detail, which can later be elaborated.it normally consists of overall application dataflow and processes of the Job Portal process.

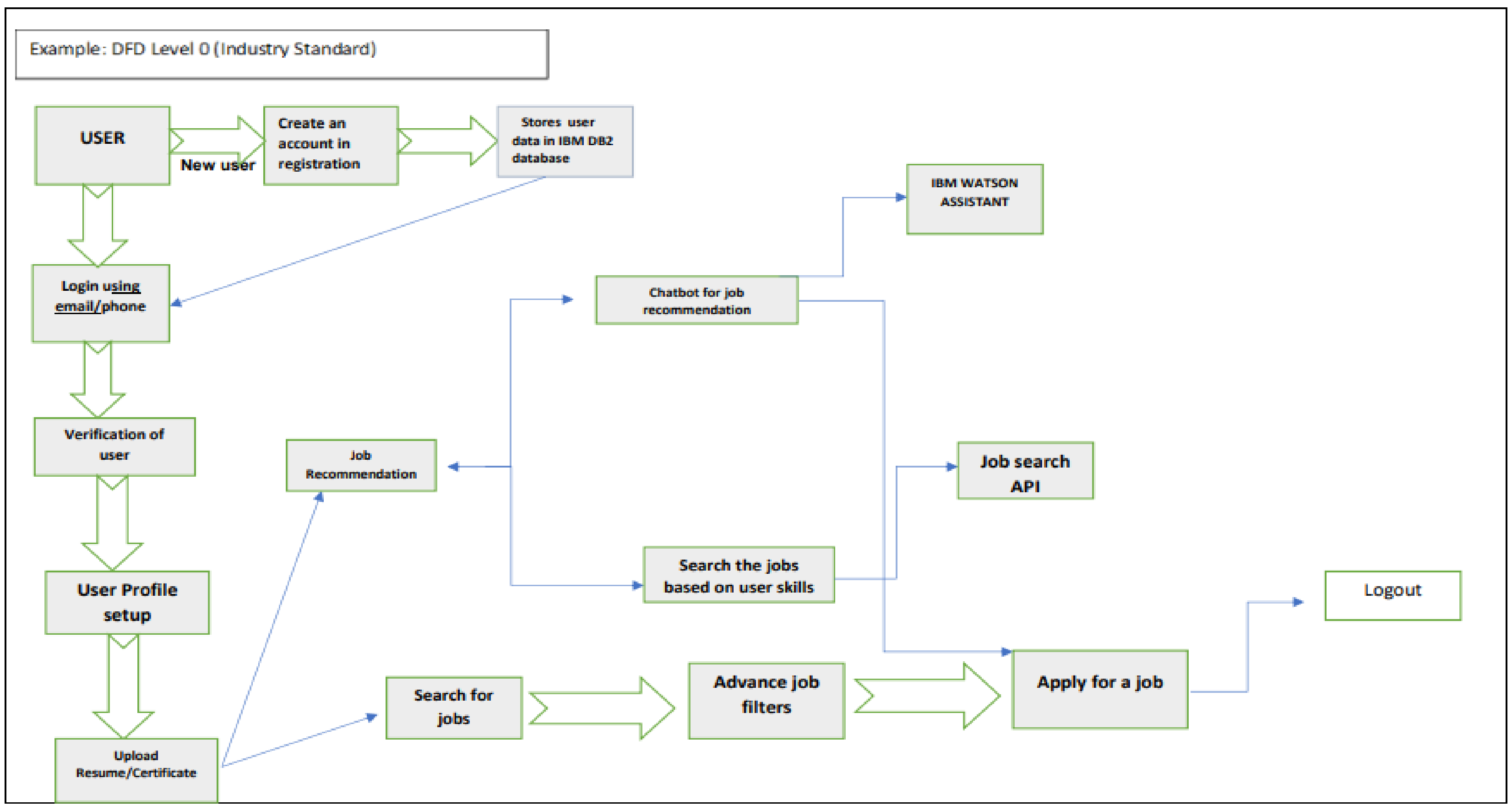
Data Flow Diagrams:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right

amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

(Simplified)




Example: DFD Level 0 (Industry Standard)
USER
New user
Create an account in registration
Stores user data in IBM DB2 database
IBM WATSON ASSISTANT
Login using email/phone
Chatbot for job recommendation
Verification of user
Job Recommendation
Job search API
User Profile setup
Search the jobs based on user skills
Logout
Apply for a job
Advance job filters
Search for jobs
Upload Resume/Certificate

5.2 Solution & Technical Architecture:

Solution architecture is a complex process – with many sub-processes – that bridges

the gap between business problems and technology solutions. Its goals are to:

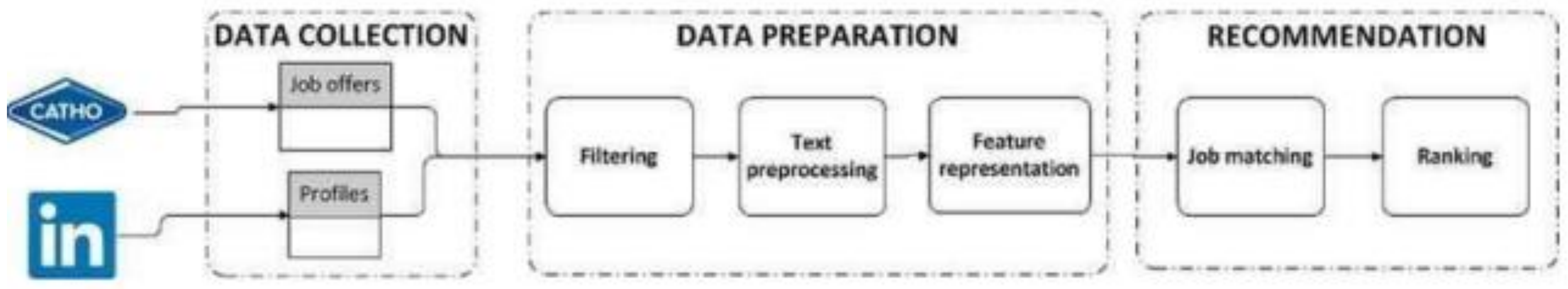
□ Find the best tech solution to solve existing business problems.

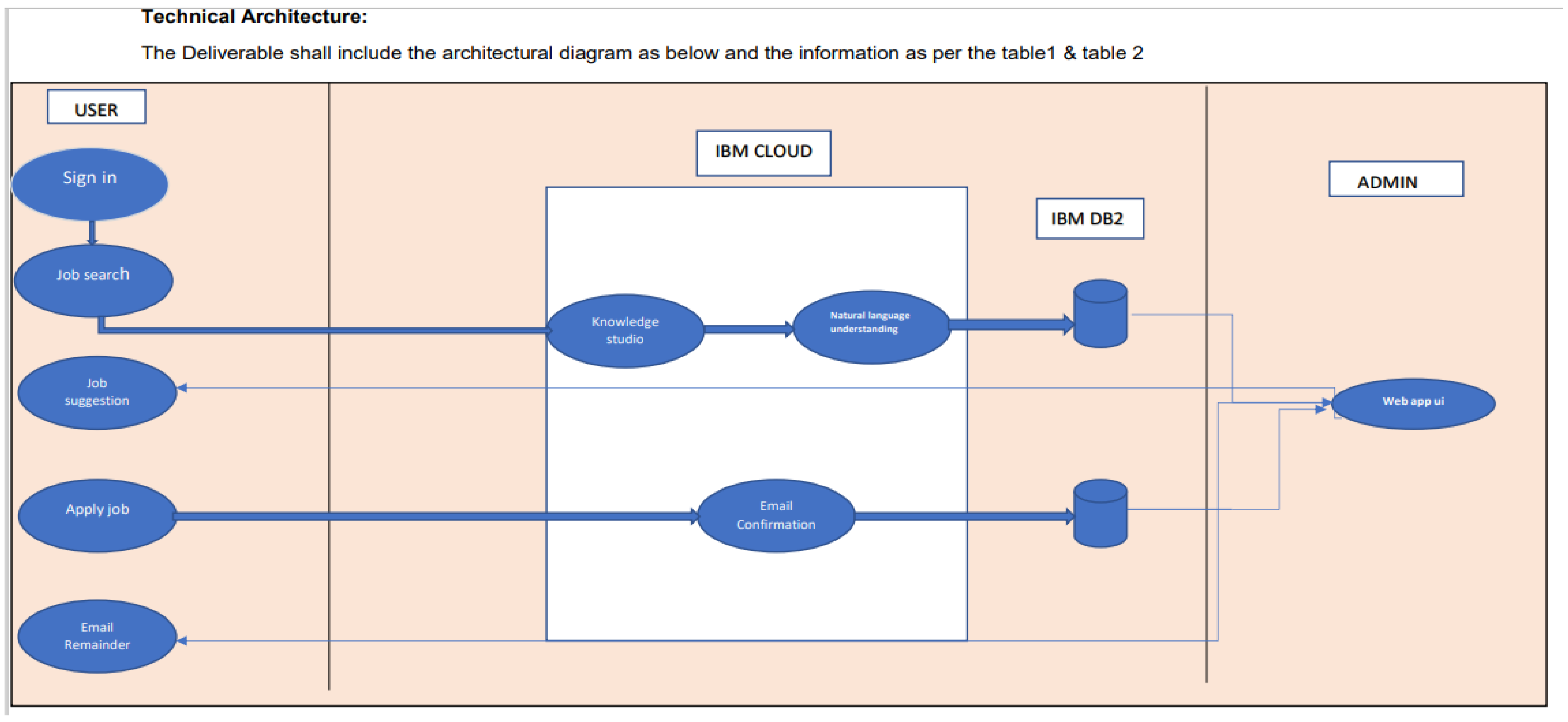
□ Describe the structure, characteristics, behavior, and other aspects ofthe

software to project stakeholders.

□ Define features, development phases, and solution requirements.

□Provide specifications according to which the solution is defined,managed,and delivered.




Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2

5.3 User Stories:

<u>Use the below template to list all the user stories for the product.</u>

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail		Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password		High	Sprint-1
	Dashboard	USN-6	As a user, I can access my dashboard after signing in.	I can access my account / dashboard	Medium	Sprint-1
Customer (Web user)	Access	USN-7	As a user, I will upload my resume, certificates, and other requirements.		Medium	Sprint-2
			As a user, I can setup a profile, and basic details by signing in	I can perform several task in the application	High	Sprint-1

Customer Care Executive	Chatbot	USN-8	As a user, I can seek guidance from the customer care executive.	.Contact customer care for support	Medium	Sprint-2
Administrator	DBMS	USN-9	As a administrator, I can keep the applications of your organization relies on running	I can perform various modifications in the applications	Hight	Sprint-1

6. PROJECT PLANNING & SCHEDULING:

□The basis of project planning is the entire project. Unlikely, project scheduling focuses only on the project-related tasks, the project start/end dates and project dependencies. Thus, a ‘project plan’ is a comprehensive document that contains the project aims, scope, costing, risks, and schedule. And a project schedule includes the estimated dates and sequential project tasks to be executed.

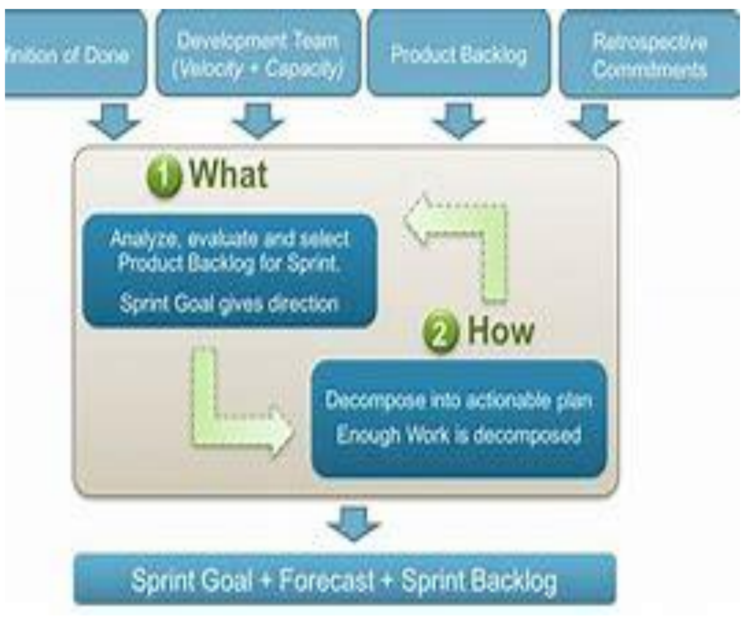
□Scheduling is the process of arranging, controlling and optimizing work and workloads in a production process or manufacturing process. Scheduling is used to allocate plant and machinery resources, plan human resources, plan production processes and purchase materials. However, Project schedule management is a management plan that views all project ...

6.1 Sprint Planning & Estimation:

□Sprint planning is an essential process that an organization needs to adapt to be successful. It indicates the roadmap for the next two to four weeks when stakeholders and team members decide as a group what they need to complete and deliver before the next sprint review meeting.

□Sprint planning is the first step in an agile project and is crucial to project success. A high-level view of the sprint backlog is created where the scrum team discusses, creates a plan for completing their work, establishes dependencies, and identifies risks that need to be addressed.

□Sprint planning is an open forum where everyone comes together, appreciates each other’s work, and gets more clarity about the sprint goals and objectives. That makes every member of the team accountable and re-enforces healthy communication


finition of Done
Development Team (Velocity + Capacity)
Product Backlog
Retrospective Commitments
1 What
Analyze, evaluate and select Product Backlog for Sprint.
Sprint Goal gives direction
2 How
Decompose into actionable plan
Enough Work is decomposed
Sprint Goal + Forecast + Sprint Backlog

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Use the below template to create product backlog and sprint schedule

Sprint	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Story Points**	**Priority**	**Team Members**
Sprint-1	Registration	USN-1	Creating Login page Creating Registration page	10	High	Manikandan, Arun, Gayathri
Sprint-1	Database Connectivity	USN-2	To Store details of the customer Connecting UI with Database	10	Medium	Vijayalakshmi, Dhanabalan
Sprint-2	SendGrid Integration	USN-3	SendGrid Integration with python code	10	Low	Vijayalakshmi, Dhanabalan
Sprint-2	Chatbot Development	USN-4	Building Chatbot Using IBM Watson assistant	10	High	Manikandan, Arun, Gayathri
Sprint-3	Job Recommender UI	USN-5	Building UI for Skill and Job Recommender Application	10	High	Manikandan, Arun, Gayathri
Sprint-3	API	USN-6	Connecting UI with indeed API	10	Medium	Vijayalakshmi, Dhanabalan
Sprint-4	Integration and Containerisation	USN-7	Integrating Chatbot to Web UI and Containerising the app	10	High	Manikandan, Arun, Dhanabalan, Gayathri
Sprint -4	Upload image and deployment	USN-8	Upload image to the IBM Registry and deploy it in the Kubernate Cluster	10	High	Manikandan, Arun, Gayathri,Vijayalakshmi

Burndown Chart:

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such

as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

https://www.visual-paradigm.com/scrum/scrum-burndown-chart/

https://www.atlassian.com/agile/tutorials/burndown-charts

Reference:

https://www.atlassian.com/agile/project-management

https://www.atlassian.com/agile/tutorials/how-to-do-scrum-with-jira-software

https://www.atlassian.com/agile/tutorials/epics

https://www.atlassian.com/agile/tutorials/sprints

https://www.atlassian.com/agile/project-management/estimation

6.2 Sprint Delivery Schedule:

Project Tracker, Velocity & Burndown Chart: (4 Marks)

Sprint	**Total Story Points**	**Duration**	**Sprint Start Date**	**Sprint End Date (Planned)**	**Story Points Completed (as on Planned End Date)**	**Sprint Release Date (Actual)**
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

7. CODING & SOLUTIONING (Explain the features added in the project along with code):

7.1 Feature 1:

Job Seeker Tools:

To make the job portal find the great job opportunities, in the most traditional form, to attract more candidates apply for relevant jobs. Job seekers demands, means learning more about the different vacancies open across the job market you can implement. In this article, we explore the most valuable Job Seekers' features to encourage with to engage more and more and interacts with ease on your job portal as below

Job Search:

The Job seekers appreciate for being able to search job listings and reach towards the desired and best match of openings, you help keep your job board audiences with maximum commitment. You job portal should consent job seekers to search by desired post with such personalize options to give them the best chances of finding job listings that meet their needs.

*save search

*View Similar Jobs

The key feature of a job board is to enable guest users and registered applicant can refine their search and easily identify the exact roles they yearning for and get the most desired job for them.

Home Page

It has an all-encompassing user-specific Home Page, step-by-step user guide, and job search button all in one place.Hence, job portal website development embedded with such a feature makes it easy for Job Seekers to post resumes on jobs of their choices.

Candidate profile creation:

Candidate can create his profile and provide his information here about his expertise and experiences.

USER LOGIN AND REGISTRATION:

HTML CODE:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<script
src="https://kit.fontawesome.com/64d58efce2.js"
crossorigin="anonymous"
></script>
<link rel="stylesheet" href="{{ url_for('static',
filename='Styles/style.css') }}" / >
<title>Sign in & Sign up Form</title>
</head>
<body>
<script src="{{ url_for('static', filename='Js/hide.js') }}"></script>
<div class="container">
<div class="forms-container">
<div class="signin-signup">
<form action="/login" class="sign-in-form" method="post">
<h2 class="title">LOG IN
</h2>
<div class="cont">
<h6 id="box">
{{mesg}}
</h6>
</div>
<div class="input-field">
```

```
<i class="fas fa-envelope"></i>
<input type="email" placeholder="Email" / name="email" required>
</div>
<div class="input-field">
<i class="fas fa-lock"></i>
<input type="password" placeholder="Password" /
name="password"required>
</div>
<input type="submit" value="Login" class="btn solid"/>
</form>
<form action="/register" class="sign-up-form" method="post"
name="sign" onsubmit="return validateform()">
<h2 class="title">REGISTER</h2>
<div class="input-field">
<i class="fas fa-user"></i>
<input type="text" placeholder="Username" / name="username"
required>
</div>
<div class="input-field">
<i class="fas fa-envelope"></i>
<input type="email" placeholder="Email" / name="email" required>
</div>
<div class="input-field">
<i class="fa fa-phone"></i>
<input type="tel" placeholder="Phone" / name="phone" required>
</div>
<div class="input-field">
<i class="fas fa-lock"></i>
<input type="password" placeholder="Password" / name="password"
required>
</div>
<input type="submit" class="btn" value="Sign up" />
</form>
</div>
</div>
<div class="panels-container">
<div class="panel left-panel">
<div class="content">
<h3 id="changeText">For Register?
<script src="{{ url_for('static', filename='Js/head.js')
}}"></script></h3><br>
<button class="btn transparent" id="sign-up-btn">
Sign up
</button>
</div>
<img src="{{ url_for('static', filename='img/log.svg') }}"
class="image" alt="" />
</div>
<div class="panel right-panel"`>
<div class="content">
<h3 id="changeText1">
<script src="{{ url_for('static', filename='Js/reg.js')
}}"></script>
</h3> <br>
<p style="font-weight:600;">
Already have an account?
</p>
<button class="btn transparent" id="sign-in-btn">
Sign in
</button>
</div>
<img src="{{ url_for('static', filename='img/register.svg') }}"
class="image" alt="" />
</div>
</div>
</div>
<script src="{{ url_for('static', filename='Js/app.js') }}"></script>
<script src="{{ url_for('static', filename='Js/sign.js') }}"></script>
</body>
</html>
```

CSS CODE: @import

```
url("https://fonts.googleapis.com/css2?family=Poppins:wght@200;300;400;500;600
;700;800&display=swap");
* {
margin: 0;
padding: 0;
box-sizing: border-box;
}
body,
input {
```

```
font-family: "Poppins", sans-serif;
}
.container {
position: relative;
width: 100%;
background-color:aliceblue;
min-height: 100vh;
overflow: hidden;
}
.forms-container {
position: absolute;
width: 100%;
height: 100%;
top: 0;
left: 0;
}
.signin-signup {
position: absolute;
top: 50%;
transform: translate(-50%, -50%);
left: 75%;
width: 50%;
transition: 1s 0.7s ease-in-out;
display: grid;
grid-template-columns: 1fr;
z-index: 5;
}
form {
display: flex;
align-items: center;
justify-content: center;
flex-direction: column;
padding: 0rem 5rem;
transition: all 0.2s 0.7s;
overflow: hidden;
grid-column: 1 / 2;
grid-row: 1 / 2;
}
form.sign-up-form {
opacity: 0;
```

```
z-index: 1;
}
form.sign-in-form {
z-index: 2;
}
.title {
font-size: 2.2rem;
color: #444;
margin-bottom: 10px;
}
.title1 {
font-size: 1.6rem;
color: #444;
margin-bottom: 10px;
}
.input-field {
max-width: 380px;
width: 100%;
background-color: #f0f0f0;
margin: 10px 0;
height: 55px;
border-radius: 55px;
display: grid;
grid-template-columns: 15% 85%;
padding: 0 0.4rem;
position: relative;
}
.input-field i {
text-align: center;
line-height: 55px;
color:black;
transition: 0.5s;
font-size: 1.1rem;
}
.input-field input {
background: none;
outline: none;
border:none;
line-height: 1;
font-weight: 600;
```

```
font-size: 1.1rem;
color: #333;
}
.input-field input::placeholder {
color: #aaa;
font-weight: 500;
}
.btn {
width: 150px;
background-color: #5995fd;
border: none;
outline: none;
height: 49px;
border-radius: 49px;
color: #fff;
text-transform: uppercase;
font-weight: 600;
margin: 10px 0;
cursor: pointer;
transition: 0.5s;
}
.btn:hover {
background-color: #4d84e2;
}
.panels-container {
position: absolute;
height: 100%;
width: 100%;
top: 0;
left: 0;
display: grid;
grid-template-columns: repeat(2, 1fr);
}
.container:before {
content: "";
position: absolute;
height: 2000px;
width: 2000px;
top: -10%;
right: 48%;
```

```
transform: translateY(-50%);
background-image: linear-gradient(-45deg, #4481eb 0%, #04befe 100%);
transition: 1.8s ease-in-out;
border-radius: 50%;
z-index: 6;
}
.image {
width: 100%;
transition: transform 1.1s ease-in-out;
transition-delay: 0.4s;
}
.panel {
display: flex;
flex-direction: column;
align-items: flex-end;
justify-content: space-around;
text-align: center;
z-index: 6;
}
.left-panel {
pointer-events: all;
padding: 3rem 17% 2rem 12%;
}
.right-panel {
pointer-events: none;
padding: 3rem 12% 2rem 17%;
}
.panel .content {
color: #fff;
transition: transform 0.9s ease-in-out;
transition-delay: 0.6s;
}
.panel h3 {
font-weight: 600;
line-height: 1;
font-size: 1.5rem;
}
.panel p {
font-size: 0.95rem;
padding: 0.7rem 0;
```

```css
}
.btn.transparent {
margin: 0;
background: none;
border: 2px solid #fff;
width: 130px;
height: 41px;
font-weight: 600;
font-size: 0.8rem;
}
.right-panel .image,
.right-panel .content {
transform: translateX(800px);
}
/* ANIMATION */
.container.sign-up-mode:before {
transform: translate(100%, -50%);
right: 52%;
}
.container.sign-up-mode .left-panel .image,
.container.sign-up-mode .left-panel .content {
transform: translateX(-800px);
}
.container.sign-up-mode .signin-signup {
left: 25%;
}
.container.sign-up-mode form.sign-up-form {
opacity: 1;
z-index: 2;
}
.container.sign-up-mode form.sign-in-form {
opacity: 0;
z-index: 1;
}
.container.sign-up-mode .right-panel .image,
.container.sign-up-mode .right-panel .content {
transform: translateX(0%);
}
.container.sign-up-mode .left-panel {
pointer-events: none;
```

```
}
.container.sign-up-mode .right-panel {
pointer-events: all;
}
@media (max-width: 870px) {
.container {
min-height: 800px;
height: 100vh;
}
.signin-signup {
width: 100%;
top: 95%;
transform: translate(-50%, -100%);
transition: 1s 0.8s ease-in-out;
}
.signin-signup,
.container.sign-up-mode .signin-signup {
left: 50%;
}
.panels-container {
grid-template-columns: 1fr;
grid-template-rows: 1fr 2fr 1fr;
}
.panel {
flex-direction: row;
justify-content: space-around;
align-items: center;
padding: 2.5rem 8%;
grid-column: 1 / 2;
}
.right-panel {
grid-row: 3 / 4;
}
.left-panel {
grid-row: 1 / 2;
}
.image {
width: 200px;
height: 200px;
transition: transform 0.9s ease-in-out;
transition-delay: 0.6s;
}
.panel .content {
padding-right: 15%;
transition: transform 0.9s ease-in-out;
transition-delay: 0.8s;
}
.panel h3 {
font-size: 1.2rem;
}
.panel p {
font-size: 0.7rem;
padding: 0.5rem 0;
}
.btn.transparent {
width: 110px;
height: 35px;
font-size: 0.7rem;
background:red;
}
.container:before {
width: 1500px;
height: 1500px;
transform: translateX(-50%);
left: 30%;
bottom: 68%;
right: initial;
top: initial;
transition: 2s ease-in-out;
}
.container.sign-up-mode:before {
transform: translate(-50%, 100%);
bottom: 32%;
right: initial;
}
.container.sign-up-mode .left-panel .image,
.container.sign-up-mode .left-panel .content {
transform: translateY(-300px);
}
.container.sign-up-mode .right-panel .image,
.container.sign-up-mode .right-panel .content {
transform: translateY(0px);
}
.right-panel .image,
.right-panel .content {
transform: translateY(300px);
}
.container.sign-up-mode .signin-signup {
top: 5%;
transform: translate(-50%, 0);
}
}
@media (max-width: 570px) {
form {
padding: 0 1.5rem;
}
.image {
display: none;
}
.panel .content {
padding: 0.5rem 1rem;
}
.container {
padding: 1.5rem;
}
.container:before {
bottom: 72%;
left: 50%;
}
.container.sign-up-mode:before {
bottom: 28%;
left: 50%;
}
}
#changeText1,#changeText{
width: 600px;
}
#box{
color: red;
}
```

```
.cont{
width: 500px;
height: 10px;
padding-left: 100px;
}
```

HOME PAGE:

HTML &CSS CODE:

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
<head>
<meta charset="UTF-8">
<link href='https://unpkg.com/boxicons@2.0.7/css/boxicons.min.css'
rel='stylesheet'>
<script src="https://code.iconify.design/iconify-icon/1.0.1/iconifyicon.min.js"></script>
<style>
/* Googlefont Poppins CDN Link */
@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@200;300;400;500;600
;700&display=swap');
*{
margin: 0;
padding: 0;
box-sizing: border-box;
font-family: 'Poppins', sans-serif;
}
.sidebar{
position: fixed;
height: 100%;
width: 240px;
background: #0A2558;
transition: all 0.5s ease;
}
.sidebar.active{
width: 60px;
}
.sidebar .logo-details{
height: 80px;
display: flex;
align-items: center;
}
```

```css
.sidebar .logo-details iconify-icon {
font-size: 28px;
font-weight: 500;
color: #fff;
min-width: 60px;
text-align: center
}
.sidebar .logo-details .logo_name{
color: #fff;
font-size: 24px;
font-weight: 500;
}
.sidebar .nav-links{
margin-top: 10px;
}
.sidebar .nav-links li{
position: relative;
list-style: none;
height: 50px;
}
.sidebar .nav-links li a{
height: 100%;
width: 100%;
display: flex;
align-items: center;
text-decoration: none;
transition: all 0.4s ease;
}
.sidebar .nav-links li a.active{
background: #081D45;
}
.sidebar .nav-links li a:hover{
background: #081D45;
}
.sidebar .nav-links li i{
min-width: 60px;
text-align: center;
font-size: 18px;
color: #fff;
}
```

```
.sidebar .nav-links li a .links_name{
color: #fff;
font-size: 15px;
font-weight: 400;
white-space: nowrap;
}
.sidebar .nav-links .log_out{
position: absolute;
bottom: 0;
width: 100%;
}
.home-section{
position: relative;
background: #f5f5f5;
min-height: 100vh;
width: calc(100% - 240px);
left: 240px;
transition: all 0.5s ease;
}
.sidebar.active ~ .home-section{
width: calc(100% - 60px);
left: 60px;
}
.home-section nav{
display: flex;
justify-content: space-between;
height: 80px;
background: #fff;
display: flex;
align-items: center;
position: fixed;
width: calc(100% - 240px);
left: 240px;
z-index: 100;
padding: 0 20px;
box-shadow: 0 1px 1px rgba(0, 0, 0, 0.1);
transition: all 0.5s ease;
}
.sidebar.active ~ .home-section nav{
left: 60px;
```

```
width: calc(100% - 60px);
}
.home-section nav .sidebar-button{
display: flex;
align-items: center;
font-size: 24px;
font-weight: 500;
}
nav .sidebar-button i{
font-size: 35px;
margin-right: 10px;
}
.home-section nav .search-box{
position: relative;
height: 50px;
max-width: 550px;
width: 100%;
margin: 0 20px;
}
form{
display:flex;
flex-direction:row;
}
button {
position: absolute;
height: 50px;
width: 50px;
background: #2697FF;
right: 1px;
top: 50%;
transform: translateY(-50%);
border-radius: 4px;
line-height: 40px;
text-align: center;
color: #fff;
border-color: #2697FF;
font-size:12px;
}
button :hover{
background:red;
```

```
}
nav .search-box input{
height: 100%;
width: 100%;
outline: none;
background: #F5F6FA;
border: 2px solid #EFEEF1;
border-radius: 6px;
font-size: 18px;
padding: 0 15px;
}
nav .search-box .bx-search{
position: absolute;
height: 50px;
width: 50px;
background: #2697FF;
top: 50%;
transform: translateY(-50%);
border-radius: 4px;
line-height: 40px;
text-align: center;
color: #fff;
font-size: 22px;
transition: all 0.4 ease;
}
.home-section nav .profile-details{
display: flex;
align-items: center;
background: #F5F6FA;
border: 2px solid #EFEEF1;
border-radius: 6px;
height: 50px;
min-width: 190px;
padding: 0 15px 0 2px;
}
nav .profile-details .admin_name{
font-size: 15px;
font-weight: 500;
color: #333;
margin: 0 10px;
```

```
white-space: nowrap;
}
nav .profile-details i{
font-size: 25px;
color: #333;
}
.boxed {
width: 100vh;
padding: 50px;
margin-left:150px;
height:80vh;
top: 0%;
height: auto;
background-color: rgb(255, 255, 255);
border-radius: 8px;
box-shadow: 0 0 10px 1px #0d5dcc;
margin-bottom: 5%;
overflow:hidden;
}
.iconclr {
color: rgb(66, 66, 130);
padding-right: 5px;
}
.companylogo {
position: static;
margin-left: 5%;
margin-top: 5%;
margin-bottom: 5%;
width: 30%;
}
.companylogooutbox {
position: relative;
left: -5px;
width: 30%;
height: 62px;
margin-bottom: 2%;
}
.content {
padding: 13px 16px;
display: flex;
```

```
border: 2px solid #eae2e1;
border-radius: 5px;
cursor: pointer;
}
.text {
font-size: 21px;
margin-left: 30px;
color: grey;
}
@keyframes animate {
0% {
background-position: left;
}
100% {
background-position: right;
}
}
.boxed_2 {
padding: 8px;
width: 1000px;
top: 0%;
height:550px;
border-radius: 8px;
}
.container{
padding-left:100px;
}
.image{
width: 200px;
transition: transform 0.9s ease-in-out;
transition-delay: 0.6s;
}
</style>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<script>
function companydetails(obj) {
var jobid = document.getElementById('jobid')
var cname = document.getElementById('cname')
var role = document.getElementById('role')
var salary = document.getElementById('salary')
var skill = document.getElementById('skill')
var ex = document.getElementById('ex')
var job_location = document.getElementById('job_location')
var vacancy = document.getElementById('vacancy')
var link = document.getElementById('link')
var logo = document.getElementById('logo')
var date = document.getElementById('d')
var date1 = document.getElementById('e')
var f = document.getElementById('f')
var json = JSON.parse(decodeHtml('{{companies}}'));
var id = obj.getAttribute('id');
var clickedcompany = json.find(function (obj) {
return obj.cname === id;
});
jobid.innerHTML = clickedcompany.jobid
cname.innerHTML = clickedcompany.cname
role.innerHTML = clickedcompany.role
salary.innerHTML = clickedcompany.salary
skill.innerHTML = clickedcompany.skill
ex.innerHTML = clickedcompany.ex
job_location.innerHTML = clickedcompany.job_location
vacancy.innerHTML = clickedcompany.vacancy
link.href = clickedcompany.link
logo.src = clickedcompany.logo
}
</script>
</head>
<body>
<div class="sidebar">
<div class="logo-details">
<iconify-icon icon="cib:skillshare"></iconify-icon>
<span class="logo_name" style="font-weight: 500; font-size: medium;">JOB
RECOMMENDER</span>
</div>
<ul class="nav-links">
<li class="admin">
<a href="/admin2">
<i class='bx bx-user' ></i>
<span class="links_name">Admin Login</span>
</a>
```

```
</li>
<li class="admin">
<a href="/employer.html">
<i class='bx bx-user' ></i>
<span class="links_name">Employer Login</span>
</a>
</li>
<li class="log_out">
<a href="/logout" >
<i class='bx bx-log-out'></i>
<span class="links_name">Log out</span>
</a>
</li>
</ul>
</div>
<section class="home-section">
<nav>
<div class="sidebar-button">
<i class='bx bx-menu sidebarBtn'></i>
<span class="dashboard">Find Jobs</span>
</div>
<div class="search-box">
<form action="/home" method="post">
<input type="text" placeholder="Job title" style="height:50px;textalign: center;" name="search">
<i class='bx bx-search' ></i>
<button type="submit" class="search">search</button>
</form>
</div>
<div class="profile-details">
<!--<img src="images/profile.jpg" alt="">-->
<span class="admin_name">{{data.EMAIL}}</span>
<i class='bx bx-chevron-down' ></i>
</div>
</nav>
<div class="info">
{%if arr!=[]%}
{% for i in arr %}
<div class="boxed">
<div class="boxedcontent">
<spam>
```

```
<img class="companylogo" src='{{i.logo}}' >
</spam>
<li style="list-style:none ;">
{{i.cname}}
</li>
<ul style="list-style:none;">
<img src="{{ url_for('static', filename='images/3.svg') }}"
class="image" alt="" />
<li>Role : {{i.role}}</li>
<li>Experience: {{i.ex}}</li>
<li>Skill : {{i.skill}}</li>
<li>Vacancy : {{i.vacancy}}</li>
<li>Stream : {{i.stream}}</li>
<li>Job Location : {{i.job_location}}</li>
<li>Salary : {{i.salary}}</li>
<li>APPLICATION LINK: <a href={{i.link}}>{{i.link}}</a>
<li>DATE POSTED: {{i.d}}</li>
<li>LAST DATE TO APPLY: {{i.e}}</li>
<li>JOB DESCRIPTION LINK: <a href={{i.link}}>{{i.f}}</a>
</li>
</ul>
</div>
</div>
</div>
{% endfor %}
<script type="module"
src="https://unpkg.com/ionicons@5.5.2/dist/ionicons/ionicons.esm.js"></script>
<script nomodule
src="https://unpkg.com/ionicons@5.5.2/dist/ionicons/ionicons.js"></script>
</div>
{% else %}
<div class="boxed_2">
<p style="margin:100px; margin-top: 100px; margin-left: 350px;">Oops!
Currently there are no such job openings</p>
</div>
{%endif%}
</div>
</section>
<script>
let sidebar = document.querySelector(".sidebar");
let sidebarBtn = document.querySelector(".sidebarBtn");

sidebarBtn.onclick = function() {

sidebar.classList.toggle("active");

if(sidebar.classList.contains("active")){

sidebarBtn.classList.replace("bx-menu" ,"bx-menu-alt-right");

}else

sidebarBtn.classList.replace("bx-menu-alt-right", "bx-menu");

}

</script>

</body>

</html>
```

7.2 Feature 2:

Employer Portal

Your job board software should comprise all job posting companies to create their profile as an Employer, this branding is a vital component for any job board to promote for the best talent pool and to create the mammoth job seekers community. Your job portal should enable all the significant employer tools with the following functionalities to manage and save time and help companies to find and select the right person for the right place without ease:

Job Posting:

Employers advertise on job openings is a process that may have been complicated without streamline with Job Posting feature, as tons of postings happens and most of are automatically. The more successful job portal is the more comfort for employers to add a new Job Post, with the essential meta data he can add on the go, keywords, tags and more comprehensive information making the job listing more precise and relevant to the viewers. With the pre-ordained Questionnaire mechanism, the system can bring the perfect match and more qualified applicants.

*Job Publishing

*Job Approval

CODING:

EMPLOYER LOGIN:

HTML & CSS CODE:

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>admin login</title>

<style>

@import

url("https://fonts.googleapis.com/css2?family=Poppins:wght@200;300;400;500;600
;700;800&display=swap");
* {
margin: 0;
padding: 0;
box-sizing: border-box;
}
body,
input {
font-family: "Poppins", sans-serif;
}
.container {
position: relative;
width: 100%;
min-height: 100vh;
}
.forms-container {
position: absolute;
width: 100%;
height: 100%;
top: 0;
left: 0;
}
.signin-signup {
position: absolute;
top: 70%;
transform: translate(-50%, -50%);
left: 75%;
width: 50%;
transition: 1s 0.7s ease-in-out;
display: grid;
grid-template-columns: 1fr;
z-index: 5;
}
form {
display: flex;
align-items: center;
justify-content: center;
flex-direction: column;
padding: 5rem 5rem;
transition: all 0.2s 0.7s;
```

```
grid-column: 1 / 2;
grid-row: 1 / 2;
}
form.sign-up-form {
opacity: 0;
z-index: 1;
}
form.sign-in-form {
z-index: 2;
}
.title {
font-size: 2.2rem;
color: #444;
margin-bottom: 10px;
}
.title1 {
font-size: 1.6rem;
color: #444;
margin-bottom: 10px;
}
.input-field {
max-width: 380px;
width: 100%;
background-color: #f0f0f0;
margin: 10px 0;
height: 55px;
border-radius: 55px;
display: grid;
grid-template-columns: 15% 85%;
padding: 0 0.4rem;
position: relative;
}
.input-field i {
text-align: center;
line-height: 55px;
color:black;
transition: 0.5s;
font-size: 1.1rem;
}
.input-field input {
```

```
background: none;
outline: none;
border:none;
line-height: 1;
font-weight: 600;
font-size: 1.1rem;
color: #333;
}
.input-field input::placeholder {
color: #aaa;
font-weight: 500;
}
.btn {
width: 150px;
background-color: #5995fd;
border: none;
outline: none;
height: 49px;
border-radius: 49px;
color: #fff;
text-transform: uppercase;
font-weight: 600;
margin: 10px 0;
cursor: pointer;
transition: 0.5s;
}
.btn:hover {
background-color: #4d84e2;
}
.panels-container {
position: absolute;
height: 100%;
width: 100%;
top: 0;
left: 0;
display: grid;
grid-template-columns: repeat(2, 1fr);
}
.container:before {
content: "";
```

```
position:fixed;
height: 2000px;
width: 2000px;
top: -10%;
right: 48%;
transform: translateY(-50%);
background-image: linear-gradient(-45deg, #4481eb 0%, #04befe 100%);
transition: 1.8s ease-in-out;
border-radius: 50%;
z-index: 6;
}
.image {
width: 100%;
transition: transform 1.1s ease-in-out;
transition-delay: 0.4s;
}
.panel {
display: flex;
flex-direction: column;
align-items: flex-end;
justify-content: space-around;
text-align: center;
z-index: 6;
}
.left-panel {
pointer-events: all;
padding: 3rem 17% 2rem 12%;
}
.right-panel {
pointer-events: none;
padding: 3rem 12% 2rem 17%;
}
.panel .content {
color: #fff;
transition: transform 0.9s ease-in-out;
transition-delay: 0.6s;
}
.panel h3 {
font-weight: 600;
line-height: 1;
```

```
font-size: 1.5rem;
}
.panel p {
font-size: 0.95rem;
padding: 0.7rem 0;
}
.btn.transparent {
margin: 0;
background: none;
border: 2px solid #fff;
width: 130px;
height: 41px;
font-weight: 600;
font-size: 0.8rem;
}
.right-panel .image,
.right-panel .content {
transform: translateX(800px);
}
/* ANIMATION */
.container.sign-up-mode:before {
transform: translate(100%, -50%);
right: 52%;
}
.container.sign-up-mode .left-panel .image,
.container.sign-up-mode .left-panel .content {
transform: translateX(-800px);
}
.container.sign-up-mode .signin-signup {
left: 25%;
}
.container.sign-up-mode form.sign-up-form {
opacity: 1;
z-index: 2;
}
.container.sign-up-mode form.sign-in-form {
opacity: 0;
z-index: 1;
}
.container.sign-up-mode .right-panel .image,
.container.sign-up-mode .right-panel .content {
transform: translateX(0%);
}
.container.sign-up-mode .left-panel {
pointer-events: none;
}
.container.sign-up-mode .right-panel {
pointer-events: all;
}
@media (max-width: 870px) {
.container {
min-height: 800px;
height: 100vh;
}
.signin-signup {
width: 100%;
top: 95%;
transform: translate(-50%, -100%);
transition: 1s 0.8s ease-in-out;
}
.signin-signup,
.container.sign-up-mode .signin-signup {
left: 50%;
}
.panels-container {
grid-template-columns: 1fr;
grid-template-rows: 1fr 2fr 1fr;
}
.panel {
flex-direction: row;
justify-content: space-around;
align-items: center;
padding: 2.5rem 8%;
grid-column: 1 / 2;
}
.right-panel {
grid-row: 3 / 4;
}
.left-panel {
grid-row: 1 / 2;
}
.image {
width: 50px;
height: 50px;
transition: transform 0.9s ease-in-out;
transition-delay: 0.6s;
}
.panel .content {
padding-right: 15%;
transition: transform 0.9s ease-in-out;
transition-delay: 0.8s;
}
.panel h3 {
font-size: 1.2rem;
}
.panel p {
font-size: 0.7rem;
padding: 0.5rem 0;
}
.btn.transparent {
width: 110px;
height: 35px;
font-size: 0.7rem;
background:red;
}
.container:before {
width: 1500px;
height: 1500px;
transform: translateX(-50%);
left: 30%;
bottom: 68%;
right: initial;
top: initial;
transition: 2s ease-in-out;
}
.container.sign-up-mode:before {
transform: translate(-50%, 100%);
bottom: 32%;
right: initial;
}
```

```
.container.sign-up-mode .left-panel .image,
.container.sign-up-mode .left-panel .content {
transform: translateY(-300px);
}
.container.sign-up-mode .right-panel .image,
.container.sign-up-mode .right-panel .content {
transform: translateY(0px);
}
.right-panel .image,
.right-panel .content {
transform: translateY(300px);
}
.container.sign-up-mode .signin-signup {
top: 5%;
transform: translate(-50%, 0);
}
}
@media (max-width: 570px) {
form {
padding: 0 1.5rem;
}
.image {
display: none;
}
.panel .content {
padding: 0.5rem 1rem;
}
.container {
padding: 1.5rem;
}
.container:before {
bottom: 72%;
left: 50%;
}
.container.sign-up-mode:before {
bottom: 28%;
left: 50%;
}
}
#changeText1,#changeText{
```

```
width: 600px;
}
#box{
color: red;
}
.cont{
width: 500px;
height: 10px;
padding-left: 100px;
}
</style>
<script
src="https://kit.fontawesome.com/64d58efce2.js"
crossorigin="anonymous"
></script>
</head>
<body>
<script src="{{ url_for('static', filename='Js/hide.js') }}"></script>
<div class="container">
<div class="forms-container">
<div class="signin-signup">
<form action="/job" class="sign-in-form" method="post">
<h3 class="title">POST JOB</h3>
<div class="cont">
<h6 id="box">
{{mesg}}
</h6>
</div>
<div class="in">
<div class="input-field">
<i class="fas fa-id-card"></i>
<input type="text" placeholder="JOBID" / name="job"
required>
</div>
<div class="input-field">
<i class="fas fa-building"></i>
<input type="text" placeholder="COMPANY" / name="cname"
required>
</div>
<div class="input-field">
```

```
<i class="fas fa-clipboard"></i>

<input type="text" placeholder="ROLE" / name="role" required>

</div>

<div class="input-field">

<i class="fas fa-chart-bar"></i>

<input type="text" placeholder="EXPERIENCE" / name="ex"

required>

</div>

<div class="input-field">

<i class="fas fa-chart-area"></i>

<input type="text" placeholder="SKILL REQUIRED" / name="skill"

required>

</div>

<div class="input-field">

<i class="fas fa-tasks"></i>

<input type="text" placeholder="VACANCY" / name="vacancy" required>

</div>

<div class="input-field">

<i class="fas fa-stream"></i>

<input type="text" placeholder="STREAM" / name="stream" required>

</div>

<div class="input-field">

<i class="fa fa-map-marker"></i>

<input type="text" placeholder="LOCATION" / name="location" required>

</div>

<div class="input-field">

<i class="fas fa-money-bill-alt"></i>

<input type="text" placeholder="SALARY" / name="salary" required>

</div>

<div class="input-field">

<i class="fa fa-globe"></i>

<input type="text" placeholder="APPLICATION LINK" / name="website"

required>

</div>

<div class="input-field">

<i class="fa fa-check-square"></i>

<input type="text" placeholder="LOGO" / name="logo" required>

</div>

<div class="input-field">

<i class="fa fa-calendar"></i>
```

```
<input type="datetime-local" placeholder="POSTED DATE" / name="d"
required>
</div>
<div class="input-field">
<i class="fa fa-calendar"></i>
<input type="datetime-local" placeholder="APLLICATION END DATE" / name="e"
required>
</div>
<div class="input-field">
<i class="fa fa-globe"></i>
<input type="text" placeholder="JOB DESCRIPTION LINK" / name="f" required>
</div>
</div>
<input type="submit" class="btn" value="POST JOB"
style="margin-left:80px;" />
</div>
</form>
</div>
</div>
<div class="panels-container">
<div class="panel left-panel">
<div class="content" style="position:fixed;padding-top:300px;">
<h3 id="changeText">POST JOB
</h3><br>
</div>
<img src="{{ url_for('static', filename='img/hire.svg') }}"
class="image"alt="" style="position:fixed; height:500px; width: 500px;
padding-bottom:150px; padding-right:100px;" />
</div>
</div>
</body>
</html>
```

ADMIN

Company details:

Html & css code:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
```

```
<title>user details</title>
<link rel="stylesheet" href="{{url_for('static',filename='style.css')}}"
/>
<link
rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css
"
integrity="sha384-
HSMxcRTRxnN+Bdg0JdbxYKrThecOKuH5zCYotlSAcp1+c8xmyTe9GYg1l9a69psu"
crossorigin="anonymous"
/>
<link
rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/3.4.1/css/bootstraptheme.min.css"
integrity="sha384-
6pzBo3FDv/PJ8r2KRkGHifhEocL+1X2rVCTTkUfGk7/0pbek5mMa1upzvWbrUbOZ"
crossorigin="anonymous"
/>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></scrip
t>
<script
src="https://stackpath.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"
integrity="sha384-
aJ21OjlMXNL5UyIl/XNwTMqvzeRMZH2w8c5cRVpzpU8Y5bApTppSuUkhZXN0VxHd"
crossorigin="anonymous"
></script>
<link
rel="stylesheet"
href="https://pro.fontawesome.com/releases/v5.10.0/css/all.css"
integrity="sha384-
AYmEC3Yw5cVb3ZcuHtOA93w35dYTsvhLPVnYs9eStHfGJvOvKxVfELGroGkvsg+p"
crossorigin="anonymous"
/>
<style>
button{
width: 150px;
background-color: #5995fd;
border: none;
outline: none;
height: 49px;
border-radius: 49px;
color: #fff;
text-transform: uppercase;
font-weight: 600;
margin: 10px 0;
cursor: pointer;
transition: 0.5s;
padding-left:48;
margin-left:1000px;
}
button:hover{
background-color: #4d84e2;
}
</style>
</head>
<body>
<div class="container">
<h3 style="color:#0071F2">User details</h3>
<table class="table table-bordered">
<thead>
<tr>
<th>UserID</th>
<th>Companyname</th>
<th>Email</th>
<th>Phone</th>
<th>Password</th>
<th>Delete</th>
</tr>
</thead>
<tbody>
{% for x in range(0,rows, 1):%}
<tr>
{%for y in range(0,5,1):%}
<td>{{r[x][y]}}</td>
{%endfor%}
<td><a class="btn btn-danger"
href="{{url_for('dele',id=r[x][0])}}">Delete</a></td>
{%endfor%}
</tr>
```

```
</tbody>
</table>
</div>
<form action="\">
<button type="submit">LOGOUT</button>
</form>
<form action="\admin">
<button type="submit">BACK</button>
</form>
</body>
</html>
```

Job details:

```
@app.route('/create', methods=['GET', 'POST'])
def create():
if request.method == 'GET':
fields = json.loads(request.args.get('fields').replace("'", '"'))
return render_template_string('''
<html>
<head>
<!-- Bootstrap CDN -->
<link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/css/bootstrap.min.css"
/>
</head>
<body>
<div class="container mt-5 text-center">
<h3>Post new Job:</h3>
<form class="mt-4" method="POST">
{% for field in fields %}
<div class="col mt-2">
<div class="row mx-auto" style="width: 300px">
<input name="{{field}}" type="text" class="formcontrol" placeholder="{{field}}">
</div>
</div>
{% endfor %}
<!-- Submit form button -->
<input type="submit" class="btn btn-success mt-4"
value="Submit"/>
</form>
</div>
```

```
</body>
</html>
''', fields=fields)
elif request.method == 'POST':
data = dict(request.form)
with open('data.csv', 'a',newline='') as f:
writer = csv.DictWriter(f, fieldnames=data.keys())
writer.writerow(data)
return redirect('/admin')
@app.route('/admin')
def read():
data = []
with open('data.csv') as f:
reader = csv.DictReader(f)
[data.append(dict(row)) for row in reader]
return render_template_string('''
<html>
<head>
<!-- Bootstrap CDN -->
<link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/css/bootstrap.min.css"
/>
<style>
button{
width: 130px;
background-color: #5995fd;
border: none;
outline: none;
height: 49px;
border-radius: 49px;
color: #fff;
text-transform: uppercase;
font-weight: 600;
margin: 10px 0;
cursor: pointer;
transition: 0.5s;
padding-left:48;
margin-left:1000px;
padding-right:50px;
}
```

```
button:hover{
background-color: #4d84e2;
}
</style>
</head>
<body>
<div class="container">
<!-- CRUD operations -->
<div class="col">
<!--a class="btn btn-danger mt-5" href="/create">Delete</a-->
</div>
<!-- CSV data -->
<table class="table table-striped mt-2" style="width: 100%;
border: 1px solid black">
<thead>
<tr class="bg-secondary text-white">
{% for header in data[0].keys() %}
<th scope="col">
{% if header == list(data[0].keys())[0] %}
<a class="btn btn-outline-light"
href="/create?fields={{str(list(data[0].keys()))}}" style="margin-right:
5px;">+</a>
{% endif%}
{{ header}}
</th>
{% endfor %}
</tr>
</thead>
<tbody>
{% for row in range(0, len(data)) %}
<tr id="{{row}}">
{% for col in range(0, len(list(data[row].values()))) %}
<td style="word-break:break-all;">
{% if col == 0 %}
<a class="btn btn-outline-danger"
href="/delete?id={{row}}" style="margin-right: 5px;">x</a>
<a href="/update?id={{row}}">{{
list(data[row].values())[col] }}</a>
{% else %}
{{ list(data[row].values())[col] }}
```

```
{% endif %}
{% endfor %}
{% endfor%}
</tbody>
</table>
</div>
<form action="/home.html">
<button type="submit">Back</button>
</form>
<form action="/detail">
<button type="submit">NEXT</button>
</form>
</body>
</html>
''', data=data, list=list, len=len, str=str)
@app.route('/delete')
def delete():
with open('data.csv') as rf:
data = []
temp_data = []
reader = csv.DictReader(rf)
[temp_data.append(dict(row)) for row in reader]
[
data.append(temp_data[row])
for row in range(0, len(temp_data))
if row != int(request.args.get('id'))
]
with open('data.csv', 'w',newline='') as wf:
writer = csv.DictWriter(wf, fieldnames=data[0].keys())
writer.writeheader()
writer.writerows(data)
return redirect('/admin')
```

BACKEND CODE FOR LOGIN,HOME ,ADMIN,EMPLOYER PAGE:

```
from flask import Flask,render_template,request,redirect,session
from flask_mail import Mail, Message
from flask import render_template_string
import csv
import m
import uuid
import json
```

```
from random import *
import db
import os
from db import ibm_db
app = Flask(__name__)
mail = Mail(app)
# configuration of mail
app.config['MAIL_SERVER']='smtp.gmail.com'
app.config['MAIL_PORT'] = 465
app.config['MAIL_USERNAME'] = 'mani567459@gmail.com'
app.config['MAIL_PASSWORD'] = 'olnlgwvxsxmchrps'
app.config['MAIL_USE_TLS'] = False
app.config['MAIL_USE_SSL'] = True
mail = Mail(app)
otp = randint(000000,999999)
print(otp)
app.secret_key = b'_5#y2L"F4Q8z\n\xec]/'
@app.route('/')
def welcome():
global row
return render_template("index.html")
@app.route('/detail')
def detail():
if db.conn:
sql="SELECT * FROM EMPLOYER"
smt= ibm_db.exec_immediate(db.conn,sql)
res = ibm_db.fetch_both(smt)
result=[]
while res != False:
result.append(res)
row=len(result)
res = ibm_db.fetch_both(smt)
return render_template("del.html",r=result,rows=row)
@app.route('/index')
def homepage():
return render_template("index.html")
@app.route('/admin2')
def admin2():
return render_template("admin.html")
@app.route('/employer.html')
```

```
def employer():
return render_template("employer.html")
@app.route('/home.html')
def postjob():
sql = "SELECT * FROM USERDETAILS WHERE ID='{0}' "
smt = ibm_db.prepare(db.conn, sql.format(session["UID"]))
ibm_db.execute(smt)
res=ibm_db.fetch_assoc(smt)
return render_template("home.html",data=res)
@app.route('/login', methods = ['GET', 'POST'])
def login():
global userid
global email
mesg = ''
if request.method == 'POST':
email = request.form['email']
password = request.form['password']
sql = "SELECT * FROM USERDETAILS WHERE email = ? AND password = ?"
stmt = ibm_db.prepare(db.conn,sql)
ibm_db.bind_param(stmt,1,email)
ibm_db.bind_param(stmt,2,password)
ibm_db.execute(stmt)
res = ibm_db.fetch_assoc(stmt)
print(res)
if res:
session["UID"]=res['ID']
return render_template("home.html",data=res)
else:
mesg = 'Invalid details. Please check the Email ID - Password
combination.!'
return render_template("index.html",mesg=mesg)
@app.route('/login1', methods = ['GET', 'POST'])
def login1():
global userid
mesg = ''
if request.method == 'POST':
email = request.form['email']
password = request.form['password']
sql = "SELECT * FROM EMPLOYER WHERE email = ? AND password = ?"
stmt = ibm_db.prepare(db.conn,sql)
```

```
ibm_db.bind_param(stmt,1,email)
ibm_db.bind_param(stmt,2,password)
ibm_db.execute(stmt)
res = ibm_db.fetch_assoc(stmt)
print(res)
if res:
session["UID"]=res['ID']
return render_template("job.html")
else:
mesg = 'Invalid details. Please check the Email ID - Password
combination.!'
return render_template("employer.html",mesg=mesg)
@app.route('/register1', methods=['GET', 'POST'])
def register1():
global username
global email
global phone
global password
if request.method == 'POST':
username = request.form['username']
email = request.form['email']
msg = Message('OTP',sender = 'mani567459@gmail.com', recipients =
[email])
msg.body = str(otp)
phone = request.form['phone']
password = request.form['password']
sql = "SELECT * FROM EMPLOYER WHERE email = ?"
stmt = ibm_db.prepare(db.conn, sql)
ibm_db.bind_param(stmt,1,email)
ibm_db.execute(stmt)
account = ibm_db.fetch_assoc(stmt)
print(account)
if account:
mesg='Job Recommender Employer Account Already exist.kindly
login!'
return render_template("employer.html",mesg=mesg)
else:
messg=mail.send(msg)
return
render_template("verify1.html",username=username,email=email,phone=phone,passw
ord=password,data=account)
@app.route('/admin1', methods = ['GET', 'POST'])
def adminlogin():
global userid
mesg = ''
if request.method == 'POST':
username = request.form['name']
password = request.form['password']
sql = "SELECT * FROM ADMIN WHERE username = ? AND password = ?"
stmt = ibm_db.prepare(db.conn,sql)
ibm_db.bind_param(stmt,1,username)
ibm_db.bind_param(stmt,2,password)
ibm_db.execute(stmt)
res = ibm_db.fetch_assoc(stmt)
print(res)
if res:
return redirect('/admin')
else:
mesg = 'Invalid details. Please check the USERNAME - Password
combination.!'
return render_template("admin.html",mesg=mesg)
@app.route("/job", methods=['POST', 'GET'])
def job():
if request.method == 'POST':
id = request.form['job']
company = request.form['cname']
role = request.form['role']
ex = request.form['ex']
skill = request.form['skill']
vacancy = request.form['vacancy']
stream = request.form['stream']
location = request.form['location']
salary = request.form['salary']
website= request.form['website']
logo= request.form['logo']
d= request.form['d']
e= request.form['e']
f= request.form['f']
m.home(id,company,role,ex,skill,vacancy,stream,location,salary,website,logo,d
,e,f)
```

```
return render_template("home.html",data='Job Recommender')
@app.route('/register', methods=['GET', 'POST'])
def register():
global username
global email
global phone
global password
if request.method == 'POST':
username = request.form['username']
email = request.form['email']
msg = Message('OTP',sender = 'mani567459@gmail.com', recipients =
[email])
msg.body = str(otp)
phone = request.form['phone']
password = request.form['password']
sql = "SELECT * FROM USERDETAILS WHERE email = ?"
stmt = ibm_db.prepare(db.conn, sql)
ibm_db.bind_param(stmt,1,email)
ibm_db.execute(stmt)
account = ibm_db.fetch_assoc(stmt)
print(account)
if account:
mesg='Job Recommender Account Already exist.kindly login!'
return render_template("index.html",mesg=mesg)
else:
messg=mail.send(msg)
return
render_template("verify.html",username=username,email=email,phone=phone,passwo
rd=password,data=account)
@app.route('/validate',methods=["POST"])
def validate():
user_otp = request.form['otp']
if otp == int(user_otp):
sql ="INSERT INTO USERDETAILS(ID,USERNAME,EMAIL,PHONE,PASSWORD)
VALUES('{0}','{1}','{2}','{3}',{4})"
res =
ibm_db.exec_immediate(db.conn,sql.format(uuid.uuid4(),username,email,phone,pas
sword))
mesg = "Your Job Recommender account successfully registered!"
msg = Message('Registered Sucessfully',sender =
'mani567459@gmail.com', recipients = [email])
msg.body = 'your Job Recommender account registered
successfully\nLogin id:\nemail:'+email+'\npassword:'+password
mail.send(msg)
return render_template("index.html",mesg=mesg)
else:
mesg ="Invalid otp.Kindly Enter the valid otp!"
return render_template("verify.html",mesg=mesg)
@app.route('/validate1',methods=["POST"])
def validate1():
user_otp = request.form['otp']
if otp == int(user_otp):
sql ="INSERT INTO EMPLOYER(USERNAME,EMAIL,PHONE,PASSWORD)
VALUES('{0}','{1}','{2}','{3}')"
res =
ibm_db.exec_immediate(db.conn,sql.format(username,email,phone,password))
mesg = "Your Job Recommender EMPLOYER account successfully
registered!"
msg = Message('Registered Sucessfully',sender =
'mani567459@gmail.com', recipients = [email])
msg.body = 'your Job Recommender EMPLOYER account registered
successfully\nLogin id:\nemail:'+email+'\npassword:'+password
mail.send(msg)
return render_template("employer.html",mesg=mesg)
else:
mesg ="Invalid otp.Kindly Enter the valid otp!"
return render_template("verify1.html",mesg=mesg)
@app.route("/home", methods=['POST', 'GET'])
def home():
if "UID" in session:
if request.method == 'POST':
sql = "SELECT * FROM USERDETAILS WHERE ID='{0}' "
smt = ibm_db.prepare(db.conn, sql.format(session["UID"]))
ibm_db.execute(smt)
res=ibm_db.fetch_assoc(smt)
user_search = request.form.get('search')
arr = []
with open("data.csv", 'r') as file:
csvreader = csv.reader(file)
for i in csvreader:
```

```
if i[2].casefold() == user_search.casefold():
dict = {
'jobid': i[0], 'cname': i[1], 'role': i[2], 'ex':
i[3], 'skill': i[4], 'vacancy': i[5], 'stream': i[6], 'job_location': i[7],
'salary': i[8], 'link': i[9], 'logo': i[10],'d':i[11],'e':i[12],'f':i[13]
}
arr.append(dict)
companies = json.dumps(arr)
return render_template("home.html", companies=companies,
arr=arr,data=res)
else:
sql = "SELECT * FROM USERDETAILS WHERE ID='{0}' "
smt = ibm_db.prepare(db.conn, sql.format(session["UID"]))
ibm_db.execute(smt)
res=ibm_db.fetch_assoc(smt)
user_search = 'Web developer'
arr = []
with open("data.csv", 'r') as file:
csvreader = csv.reader(file)
for i in csvreader:
if i[2].casefold() == user_search.casefold():
dict = {
'jobid': i[0], 'cname': i[1], 'role': i[2], 'ex':
i[3], 'skill': i[4], 'vacancy': i[5], 'stream': i[6], 'job_location': i[7],
'salary': i[8], 'link': i[9], 'logo': i[10]
}
arr.append(dict)
companies = json.dumps(arr)
return render_template("home.html", companies=companies,
arr=arr,data=res)
else:
return render_template("index.html")
@app.route('/create', methods=['GET', 'POST'])
def create():
if request.method == 'GET':
fields = json.loads(request.args.get('fields').replace("'", '"'))
return render_template_string('''
<html>
<head>
<!-- Bootstrap CDN -->
```

```
<link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/css/bootstrap.min.css"
/>
</head>
<body>
<div class="container mt-5 text-center">
<h3>Post new Job:</h3>
<form class="mt-4" method="POST">
{% for field in fields %}
<div class="col mt-2">
<div class="row mx-auto" style="width: 300px">
<input name="{{field}}" type="text" class="formcontrol" placeholder="{{field}}">
</div>
</div>
{% endfor %}
<!-- Submit form button -->
<input type="submit" class="btn btn-success mt-4"
value="Submit"/>
</form>
</div>
</body>
</html>
''', fields=fields)
elif request.method == 'POST':
data = dict(request.form)
with open('data.csv', 'a',newline='') as f:
writer = csv.DictWriter(f, fieldnames=data.keys())
writer.writerow(data)
return redirect('/admin')
@app.route('/admin')
def read():
data = []
with open('data.csv') as f:
reader = csv.DictReader(f)
[data.append(dict(row)) for row in reader]
return render_template_string('''
<html>
<head>
<!-- Bootstrap CDN -->
<link rel="stylesheet"
```

```
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/css/bootstrap.min.css"
/>
<style>
button{
width: 130px;
background-color: #5995fd;
border: none;
outline: none;
height: 49px;
border-radius: 49px;
color: #fff;
text-transform: uppercase;
font-weight: 600;
margin: 10px 0;
cursor: pointer;
transition: 0.5s;
padding-left:48;
margin-left:1000px;
padding-right:50px;
}
button:hover{
background-color: #4d84e2;
}
</style>
</head>
<body>
<div class="container">
<!-- CRUD operations -->
<div class="col">
<!--a class="btn btn-danger mt-5" href="/create">Delete</a-->
</div>
<!-- CSV data -->
<table class="table table-striped mt-2" style="width: 100%;
border: 1px solid black">
<thead>
<tr class="bg-secondary text-white">
{% for header in data[0].keys() %}
<th scope="col">
{% if header == list(data[0].keys())[0] %}
<a class="btn btn-outline-light"
```

```
href="/create?fields={{str(list(data[0].keys()))}}" style="margin-right:
5px;">+</a>
{% endif%}
{{ header}}
</th>
{% endfor %}
</tr>
</thead>
<tbody>
{% for row in range(0, len(data)) %}
<tr id="{{row}}">
{% for col in range(0, len(list(data[row].values()))) %}
<td style="word-break:break-all;">
{% if col == 0 %}
<a class="btn btn-outline-danger"
href="/delete?id={{row}}" style="margin-right: 5px;">x</a>
<a href="/update?id={{row}}">{{
list(data[row].values())[col] }}</a>
{% else %}
{{ list(data[row].values())[col] }}
{% endif %}
{% endfor %}
{% endfor%}
</tbody>
</table>
</div>
<form action="/home.html">
<button type="submit">Back</button>
</form>
<form action="/detail">
<button type="submit">NEXT</button>
</form>
</body>
</html>
''', data=data, list=list, len=len, str=str)
@app.route('/delete')
def delete():
with open('data.csv') as rf:
data = []
temp_data = []
```

```
reader = csv.DictReader(rf)
[temp_data.append(dict(row)) for row in reader]
[
data.append(temp_data[row])
for row in range(0, len(temp_data))
if row != int(request.args.get('id'))
]
with open('data.csv', 'w',newline='') as wf:
writer = csv.DictWriter(wf, fieldnames=data[0].keys())
writer.writeheader()
writer.writerows(data)
return redirect('/admin')
@app.route('/logout')
def logout():
session.pop("UID",None)
return redirect("/index")
@app.route("/dele/<string:id>",methods=['POST','GET'])
def dele(id):
sql="DELETE FROM EMPLOYER WHERE id='{0}'"
smt=ibm_db.exec_immediate(db.conn,sql.format(id))
return redirect("/detail")
if(__name__=='__main__'):
port = os.environ.get("PORT",5000)
app.run(port=port,host='0.0.0.0',debug=True)
```

7.3 Database Schema (if Applicable):

8. TESTING :

Testing is the final stage of our skill and job recommender project.In the testing process we can test a developed portal by us to check its working or not .

8.1 Test Cases:

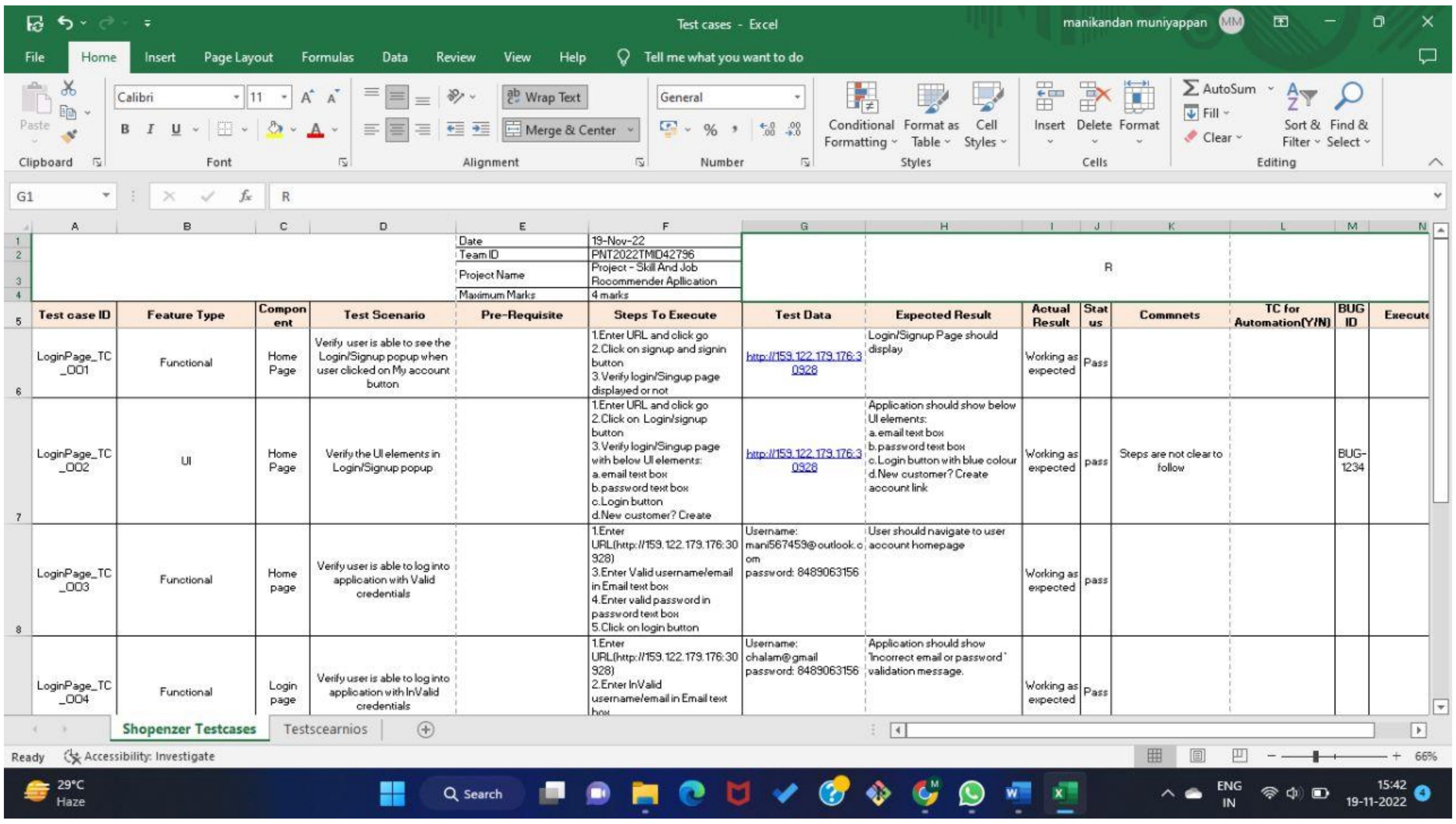
User Login : we have check the sign in /sign up icon and fill the register page to login and also verify a OTP to their email.And finally login to the home page

Employer login: We have to click the emoployer the icon on the home page and the we have to check the sign up/sign in icon and also the fill register page for the employer

And also we want to check the employer have to option to post a job

Admin login: The admin can also add /delete a post and also the admin have a option to delete a fake employer details.

8.2 User Acceptance Testing:

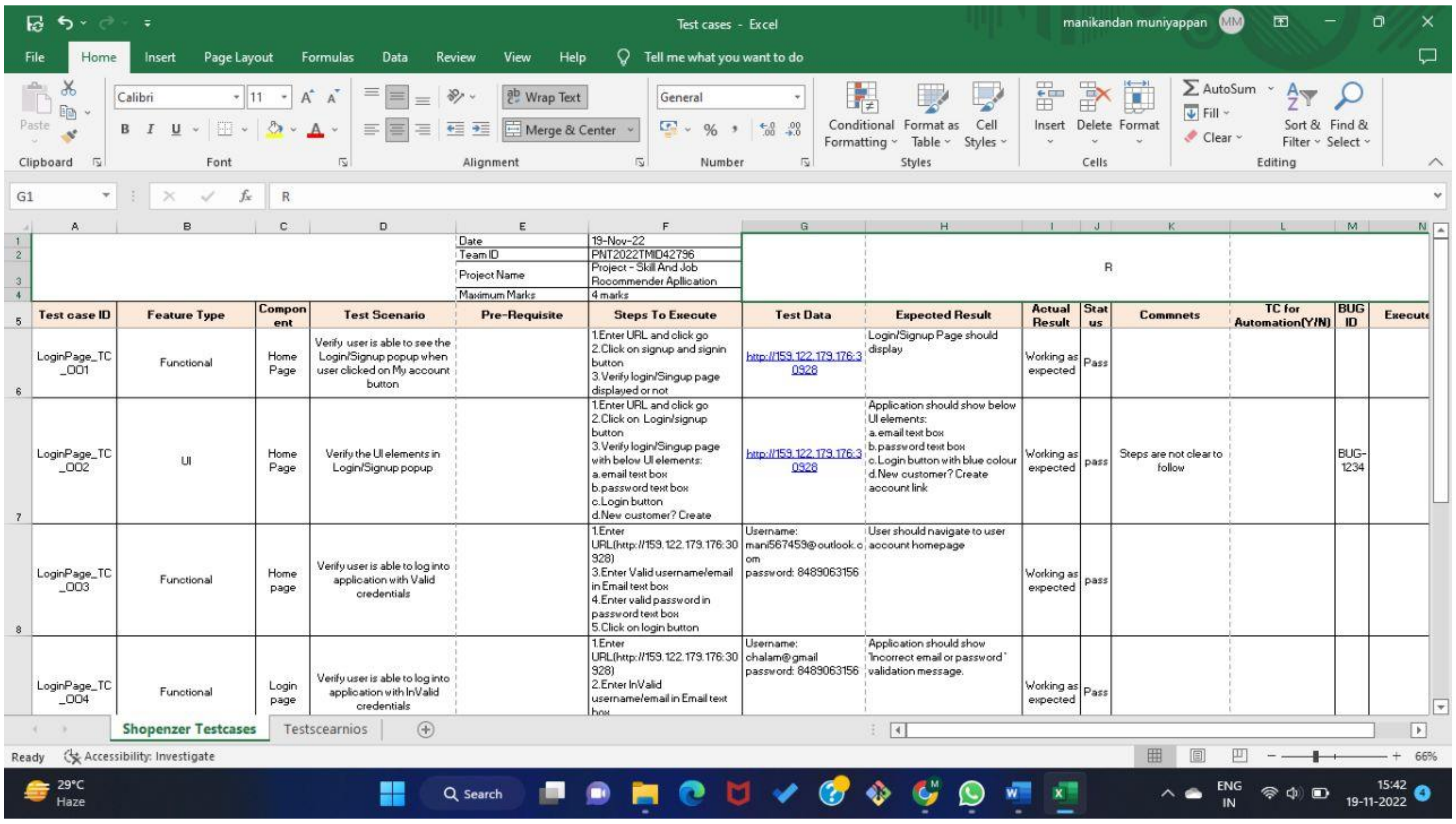

Test cases - Excel

G1 R

				Date	19-Nov-22								
				Team ID	PNT2022TMID42796								
				Project Name	Project - Skill And Job Rocommender Apllication		R						
				Maximum Marks	4 marks								
Test case ID	**Feature Type**	**Component**	**Test Scenario**	**Pre-Requisite**	**Steps To Execute**	**Test Data**	**Expected Result**	**Actual Result**	**Status**	**Commnets**	**TC for Automation(Y/N)**	**BUG ID**	**Execute**
LoginPage_TC_001	Functional	Home Page	Verify user is able to see the Login/Signup popup when user clicked on My account button		1.Enter URL and click go 2.Click on signup and signin button 3.Verify login/Singup page displayed or not	http://159.122.179.176:30928	Login/Signup Page should display	Working as expected	Pass				
LoginPage_TC_002	UI	Home Page	Verify the UI elements in Login/Signup popup		1.Enter URL and click go 2.Click on Login/signup button 3.Verify login/Singup page with below UI elements: a.email text box b.password text box c.Login button d.New customer? Create	http://159.122.179.176:30928	Application should show below UI elements: a.email text box b.password text box c.Login button with blue colour d.New customer? Create account link	Working as expected	pass	Steps are not clear to follow		BUG-1234	
LoginPage_TC_003	Functional	Home page	Verify user is able to log into application with Valid credentials		1.Enter URL(http://159.122.179.176:30928) 3.Enter Valid username/email in Email text box 4.Enter valid password in password text box 5.Click on login button	Username: mani567459@outlook.com password: 8489063156	User should navigate to user account homepage	Working as expected	pass				
LoginPage_TC_004	Functional	Login page	Verify user is able to log into application with InValid credentials		1.Enter URL(http://159.122.179.176:30928) 2.Enter InValid username/email in Email text box	Username: chalam@gmail password: 8489063156	Application should show 'Incorrect email or password ' validation message.	Working as expected	Pass				

Shopenzer Testcases | Testscearnios

Ready Accessibility: Investigate

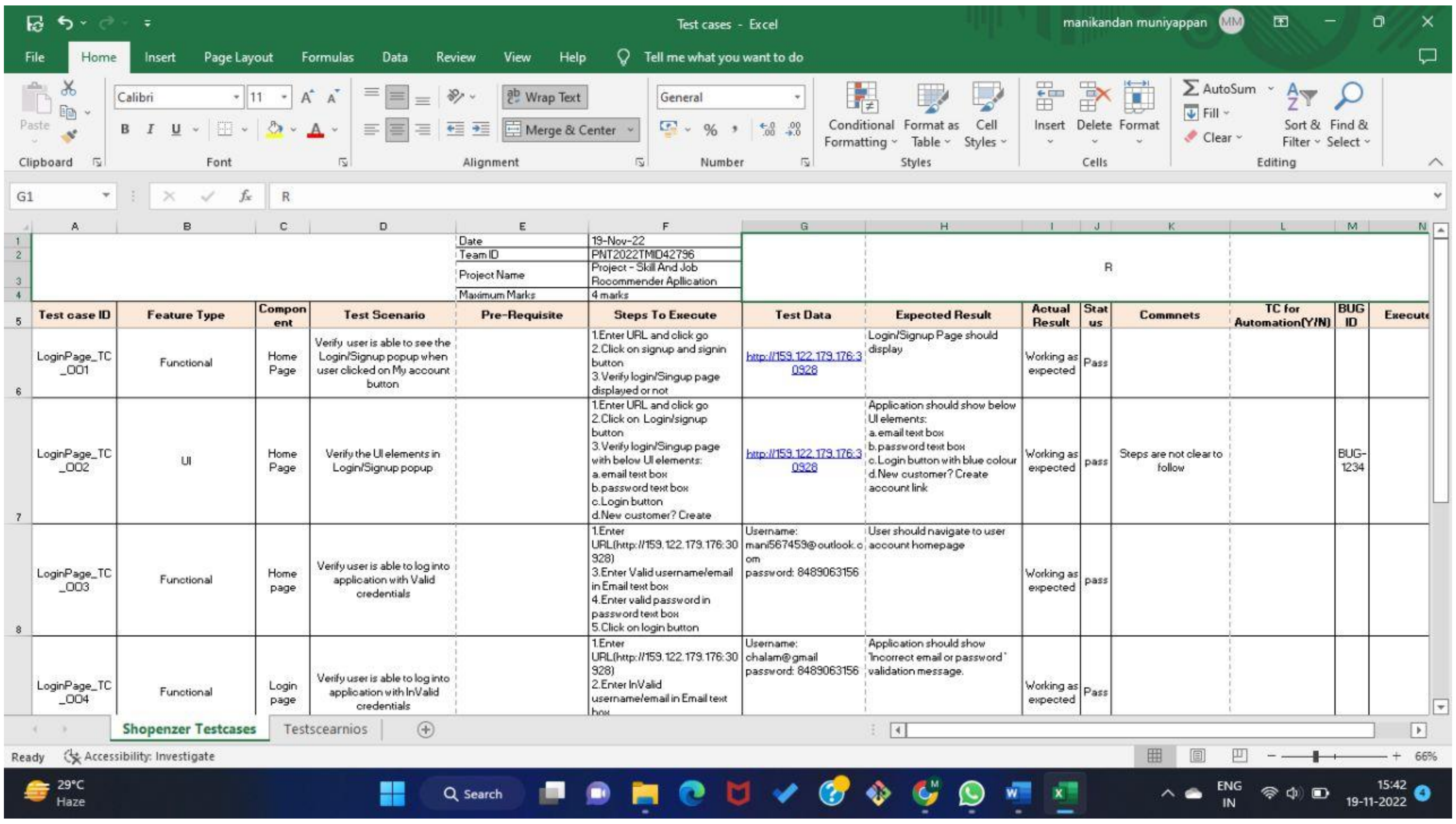

Test cases - Excel

G1 R

				Date	19-Nov-22								
				Team ID	PNT2022TMID42796								
				Project Name	Project - Skill And Job Rocommender Apllication		R						
				Maximum Marks	4 marks								
Test case ID	**Feature Type**	**Component**	**Test Scenario**	**Pre-Requisite**	**Steps To Execute**	**Test Data**	**Expected Result**	**Actual Result**	**Status**	**Commnets**	**TC for Automation(Y/N)**	**BUG ID**	**Execute**
LoginPage_TC_001	Functional	Home Page	Verify user is able to see the Login/Signup popup when user clicked on My account button		1.Enter URL and click go 2.Click on signup and signin button 3.Verify login/Singup page displayed or not	http://159.122.179.176:30928	Login/Signup Page should display	Working as expected	Pass				
LoginPage_TC_002	UI	Home Page	Verify the UI elements in Login/Signup popup		1.Enter URL and click go 2.Click on Login/signup button 3.Verify login/Singup page with below UI elements: a.email text box b.password text box c.Login button d.New customer? Create	http://159.122.179.176:30928	Application should show below UI elements: a.email text box b.password text box c.Login button with blue colour d.New customer? Create account link	Working as expected	pass	Steps are not clear to follow		BUG-1234	
LoginPage_TC_003	Functional	Home page	Verify user is able to log into application with Valid credentials		1.Enter URL(http://159.122.179.176:30928) 3.Enter Valid username/email in Email text box 4.Enter valid password in password text box 5.Click on login button	Username: mani567459@outlook.com password: 8489063156	User should navigate to user account homepage	Working as expected	pass				
LoginPage_TC_004	Functional	Login page	Verify user is able to log into application with InValid credentials		1.Enter URL(http://159.122.179.176:30928) 2.Enter InValid username/email in Email text box	Username: chalam@gmail password: 8489063156	Application should show 'Incorrect email or password ' validation message.	Working as expected	Pass				

Shopenzer Testcases | Testscearnios

Ready Accessibility: Investigate



Date	19-Nov-22
Team ID	PNT2022TMID42796
Project Name	Project - Skill And Job Rocommender Apllication
Maximum Marks	4 marks

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation(Y/N)	BUG ID	Execute
LoginPage_TC_001	Functional	Home Page	Verify user is able to see the Login/Signup popup when user clicked on My account button		1.Enter URL and click go 2.Click on signup and signin button 3.Verify login/Singup page displayed or not	http://159.122.179.176:30928	Login/Signup Page should display	Working as expected	Pass				
LoginPage_TC_002	UI	Home Page	Verify the UI elements in Login/Signup popup		1.Enter URL and click go 2.Click on Login/signup button 3.Verify login/Singup page with below UI elements: a.email text box b.password text box c.Login button d.New customer? Create	http://159.122.179.176:30928	Application should show below UI elements: a.email text box b.password text box c.Login button with blue colour d.New customer? Create account link	Working as expected	pass	Steps are not clear to follow		BUG-1234	
LoginPage_TC_003	Functional	Home page	Verify user is able to log into application with Valid credentials		1.Enter URL(http://159.122.179.176:30928) 3.Enter Valid username/email in Email text box 4.Enter valid password in password text box 5.Click on login button	Username: mani567459@outlook.com password: 8489063156	User should navigate to user account homepage	Working as expected	pass				
LoginPage_TC_004	Functional	Login page	Verify user is able to log into application with InValid credentials		1.Enter URL(http://159.122.179.176:30928) 2.Enter InValid username/email in Email text box	Username: chalam@gmail password: 8489063156	Application should show 'Incorrect email or password' validation message.	Working as expected	Pass				

9. RESULTS:

This trait **makes it easier for job seekers to decide on a specific career**. Equally, online job search sites can help employers pace up their hiring process. It would be easy for recruiters or hiring managers to verify job applications online than physically scan printed resumes

10. ADVANTAGES & DISADVANTAGES:

Advantages:

Branding opportunity for employers

Reduced cost-of-hire

Reduced time-to-hire

Allows for database build-up and tracking.

finding open positions, researching potential employers, applying online, comparing salaries and marketing yourself as a top-notch candidate.

advertise their job vacancies to job seekers. Job seekers can use job boards to search for new job opportunities in their area and profession.

Disadvantage:

Too many places to look. There are dozens of places that job seekers can look for open positions.

Uninformative job descriptions.

Not enough information.

Lengthy, confusing hiring processes. .

No feedback.

11. CONCLUSION:

The final section of our thesis is the conclusion. conclusion is nothing the summarize of our over-all project for the job seekers based on their skills. In our project we are creating a web portal for the job seekers with a cloud base data storage . In these we are using some web tools for make a front end development .Then the back end developed with python and we have access to use the IBM cloud for the data storage. Using kubernets we make the portal into public access for the job seekers .We think it will user friendly to the job seekers based upon their skills in their location itself

12. FUTURE SCOPE:

❑ Solve complex problems in a way that fits the state of your customers.

❑ Succeed faster and increase your solution adoption by tapping into existing

mediums and channels of behavior.

❑ Sharpen your communication and marketing strategy with the right triggers

and messaging.

❑ Increase touch-points with your company by finding the right

problem-behavior fit and building trust by solving frequent annoyances, or

urgent or costly problems.

❑ Understand the existing situation in order to improve it for your target group

13. APPENDIX:

GitHub & Project Demo Link

https://github.com/IBM-EPBL/IBM-Project-15112-1659594271