SKILL / JOB RECOMMENDER APPLICATION

A MINI-PROJECT REPORT

Submitted by

ASVITHA K (AC19UC011)

BARKAVI R (AC19UC014)

ESHASWETHA E (AC19UC028)

HIMA VASINI M (AC19UC036)

In partial fulfillment of the award of the degree

of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

ADHIYAMAAN COLLEGE OF ENGINEERING

DR.M.G R NAGAR, HOSUR-635130

ANNA UNIVERSITY: CHENNAI 600025

NOVEMBER 2022

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this mini project report **“SKILL AND JOB RECOMMENDER"** is the bonafide work of **“HIMA VASINI M (AC19UCS036), ASVITHA K (AC19UCS011), BARKAVI R (AC19UCS014), ESHASWETHA E (AC19UCS028)”** who carried out the project under my supervision.

SIGNATURE	**SIGNATURE**
Dr. G. FATHIMA, M.E., Ph.D.,	**Mrs.D.M. VIJAYA LAKSHMI**
HEAD OF THE DEPARTMENT	**M.E., SUPERVISOR**
PROFESSOR,	**PROFESSOR,**
Department of CSE,	Department of CSE,
Adhiyamaan College of Engineering,	Adhiyamaan College of Engineering,
(Autonomous)	(Autonomous)
Dr. M.G.R. Nagar,	Dr. M.G.R. Nagar,
Hosur – 635 130.	Hosur – 635 130.

Submitted for the Mini project VIVA-VOCE Examination held on ________________ at **Adhiyamaan College of Engineering (Autonomous), Hosur-635 130.**

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

It is one of the most efficient tasks in life to choose the appropriate words to express one’s gratitude to the beneficiaries. We are very much grateful to God who helped us all the way through the project and how moulded us into what we are today.

We are grateful to our beloved Principal **Dr. G. RANGANATH, M.E., Ph.D.,** Adhiyamaan College of Engineering (Autonomous), Hosur for providing the opportunity to do this work in premises

We acknowledge our heartful gratitude to **Dr. G. FATHIMA, M.E., Ph.D.,** Professor and Head of the Department, Department of Computer Science and Engineering, Adhiyamaan College of Engineering (Autonomous), Hosur, for her guidance and valuable suggestions and encouragement throughout this project.

We are highly indebted to **Mrs.D.M. VIJAYA LAKSHMI M.E., SUPERVISOR Professor**, Department of Computer Science and Engineering, Adhiyamaan College of Engineering (Autonomous), Hosur, whose immense support, encouragement and valuable guidance made us to complete this project successfully.

We also extent our thanks to Project Coordinator and all Staff Members for their support in complete this project successfully.

Finally, we would like to thank to our parents, without their motivational and support would not have been possible for us to complete this project successfully.

SKILL AND JOB RECOMMENDER
CONTENTS

CHAPTER-1

INTRODUCTION

1.1 PROJECT OVERVIEW

The skill and job recommender project provides a 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.

Dealing with the enormous amount of recruiting information on the Internet, a job seeker always spends hours to find useful ones. To reduce this laborious work, we design and implement a recommendation system for online job-hunting. We also take background information including students resumes and details of recruiting information into consideration.

The main aim of this project is 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.

This system has traditionally been treated as a filter-based match or as recommendation based on the features of jobs and candidates as discrete entities. We have deployed our model in a real-world job recommender system and have achieved the best click-through rate.

.

1.2 PURPOSE

The main purpose the job and skill recommender application is to suggest relevant jobs and skills to the user. To achieve this task, there exist two major categories of methods which include recommendations through chatbot and job search engine.

This recommender application aims to help users in finding jobs and skills that match their personnel interests; it also has a successful usage in e-commerce applications to deal with problems related to information overload efficiently. Basically, job recommender system is designed to retrieve a list of job descriptions to a job applicant based on his/her preferences or to generate a list of job candidates to a recruiter based on the job requirements.

ADMIN: The main role and responsibility of the admin is to take care of the whole recommendation process. Only the admin will be able to track the whole process which include user registration, user login, chat bot recommendations and they will be able to send an alert message to the user when there are current job openings.

USER: The user can register on the website by providing valid information. After registration, they can login and get recommendation by using job search engine or interacting with the chatbot.

An excellent job recommender system not only enables to recommend a higher paying job which is maximally aligned with the skill set of the current job, but also suggests acquiring few additional skills which are required for new roles.

CHAPTER -2

LITERATURE SURVEY

2.1 EXISTING PROMBLEM

Jorge Valverde-Rebaza ,et al.[1] proposed a framework for job recommendation based on professional skills of job seekers, which automatically extracts the skills from the job seeker profiles using a variety of text processing techniques .It performs the job recommendation using TF-IDF (Term Frequency-Inverse Document Frequency) and four different configurations of Word2vec over a dataset of job seeker profiles and job vacancies.TF-IDF assigns weights to the words as a statistical measure and it is used to evaluate the relevance of a word in document of a corpus .To perform job offers scraping ,it creates a list of keywords from the IT industry and used them as search terms .The system also uses text mining approaches to process both profiles and job offers data and selects the work experience, education and competencies/skills from the profiles and, the description and title from the job offers .As a result ,this framework facilitates the understanding of job recommendation process as well as it allows the use of a variety of text processing and recommendation methods according to the preferences of the job recommender system designer.

Giacomo Domeniconi, et al. [2] proposed a system which aims to find out relationships between jobs and people skills, making use of data from LinkedIn users' public profiles. This job recommendation system is based on Latent Semantic Analysis (LSA) for the support in the job seeking process, evaluating its performance through a hierarchical clustering of job positions. It also uses clustering to partition data into previously unknown groups of similar items and is applied in a large variety of contexts. It uses the mined semantics to obtain a hierarchical clustering of job positions and to build a job recommendation system. The outcome proves the effectiveness of proposed method in recommending job positions. To evaluate this method, the system takes LinkedIn as reference scenario because of its widespread use, crawling real public profiles from which it can easily infer information about people skills and current job positions. As a result, these methods have been tested using a set of public profiles extracted from LinkedIn and shows how the same data can be used to automatically build a folksonomy of job position by means of hierarchical clustering, in order to discover groups of related occupations.

D. Mhamdi, et al,[3] proposed a recommender system that aims to help job seekers to find suitable jobs. First, job offers are collected from job search websites then they are prepared to extract meaningful attributes such as job titles and technical skills. Job offers with common features are grouped into clusters. As job seeker like one job belonging to a cluster, the person will probably find other jobs in that cluster that the person will like as well. A list of top n recommendations is suggested after matching data from job clusters and job seeker behavior, which consists of user interactions such as applications, likes and rating. This model aims to extract meaningful data from job postings using text-clustering methods. As a result, job offers are divided into job clusters based on their common features and job offers are matched to job seekers according to their interactions and therefore, the applicants could receive personalized online jobs and recruiters will find the most relevant candidates with skills and qualifications that fit their needs.

Shaha T. Al-Otaibi and Mourad Ykhlef [4] proposed an e-commerce applications joint recommender system in order to expand customer services, increase selling rates and decrease customers search technologies that can help recruiters to handle the information efficiently. The recruiters generate the job description by determining the set of requirements and constraints on skills, expertise levels, and degrees. The job- seeker on the other hand, generates his/her CV by specifying the academic background, previous work experience and skills. It also analyzed the e-recruiting process and the different aspects related to applying the recommender systems in candidates/job matching problem. To construct candidate profiles, the meta-data is extracted from existing resumes. In this system, user profiles are constructed by passively detecting the clickstream and read-time behavior of users. And finally, the proposed system accomplishes recommendation using E-recruitment platforms for both recruiters and jobseekers by reducing the recruitment time and advertisement cost.

2.2 REFERENCE

[1] Schafer J B, Frankowski D, Herlocker J, et al. Collaborative _ltering recommender systems[M]//The adaptive web. Springer Berlin Heidelberg, 2007: 291-324.

[2] Pazzani M J, Billsus D. Content-based recommendation systems[M]//The adaptive web. Springer Berlin Heidelberg, 2007:325-341.

[3] Sarwar B, Karypis G, Konstan J, et al. Item-based collaborative_ltering recommendation algorithms[C]//Proceedings of the 10th international conference on World Wide Web. ACM, 2001: 285-295.

[4] Nikolaos D. Almalis ,Prof. George A. Tsihrintzis , Nikolaos Karagianniset al."FoDRA - A New Content-Based Job Recommendation Algorithm for Job Seeking and Recruiting".

[5] Anika Gupta, Dr. Deepak Garg. "Applying Data Mining Techniques in Job Recommender System for Considering Candidate Job Preferences".

[6] Dunning T. Accurate methods for the statistics of surprise and coincidence[J]. Computational linguistics, 1993, 19(1): 61-74.

[7] Emmanuel Malherbe, Mamadou Diaby , Mario Cataldi et al. "Field Selection for Job Categorization and Recommendation to Social Network Users". 2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2014)

[8] Elsafty, A., Riedl, M., & Biemann, C. (2018, June). Document-based Recommender System for Job Postings using Dense Representations. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,Volume 3 (Industry Papers) (pp. 216-224).

[9] Abel, F., Benczúr, A., Kohlsdorf, D., Larson, M., & Pálovics, R. (2016, September). Recsys challenge 2016: Job recommendations. In Proceedings of the 10th ACM Conference on Recommender Systems (pp. 425- 426). ACM.

2.3 PROBLEM STATEMENT DEFINITION

In some cases, people who lack industry knowledge are unclear about what exactly they need to learn in order to get a suitable job for them. We address the problem of recommending suitable jobs to people who are seeking a new job. Job recommender technology aims to help job seekers in finding jobs that match their skills.

The internet caused a substantial impact on the recruitment process through the creation of e-recruiting platforms that become a primary recruitment channel in most companies. While companies established job positions on these portals, jobseeker uses them to publish their profiles.

E-Recruitment platforms accomplished clear advantages for both recruiters and jobseekers by reducing the recruitment time and advertisement cost. Recommender system technology aims to help users in finding items that match their preferences; it has a successful usage in a wide range of applications to deal with problems related to information overload efficiently.

In current situation, recruitments done manually for lakhs of students in which many talented students may lose their opportunities due to different reasons since it is done manually, and further company also needs the highly talented people from the mass group for their growth. So, we have built a cloud application to do this process in an efficient manner.

CHAPTER - 3

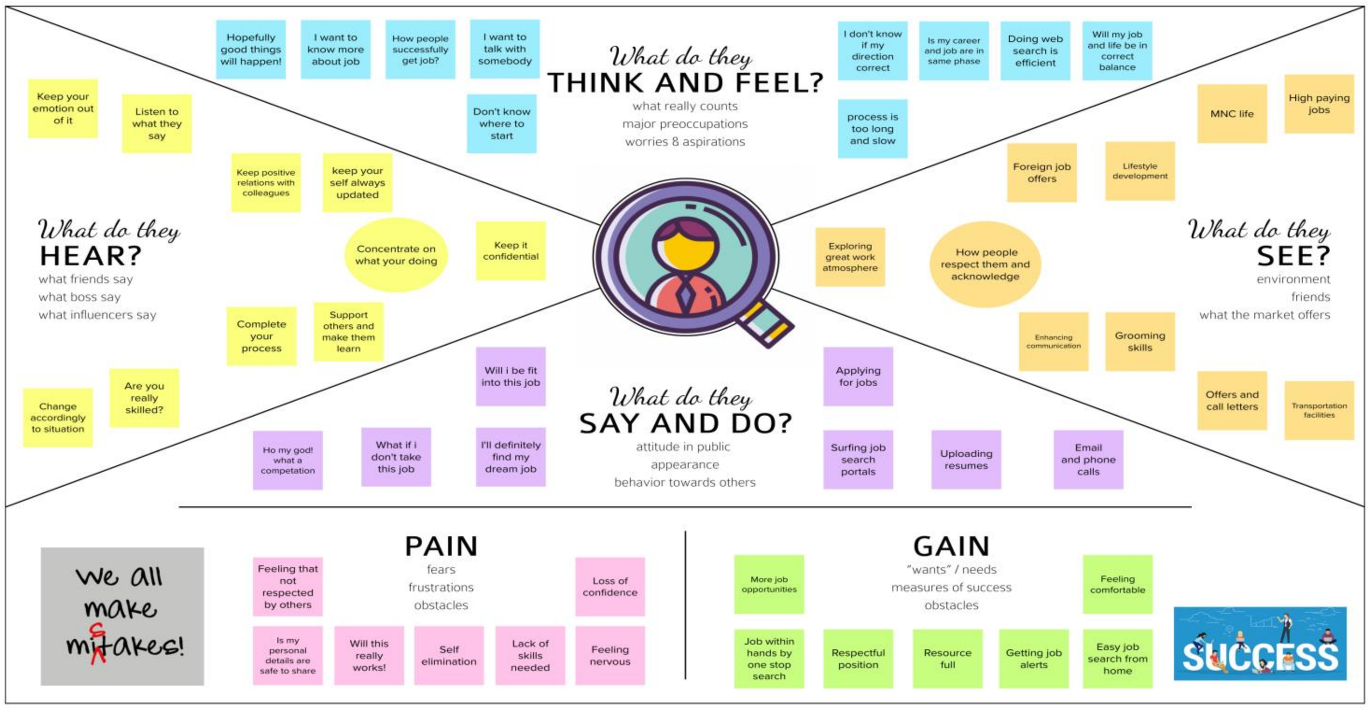
IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviors and attitudes.

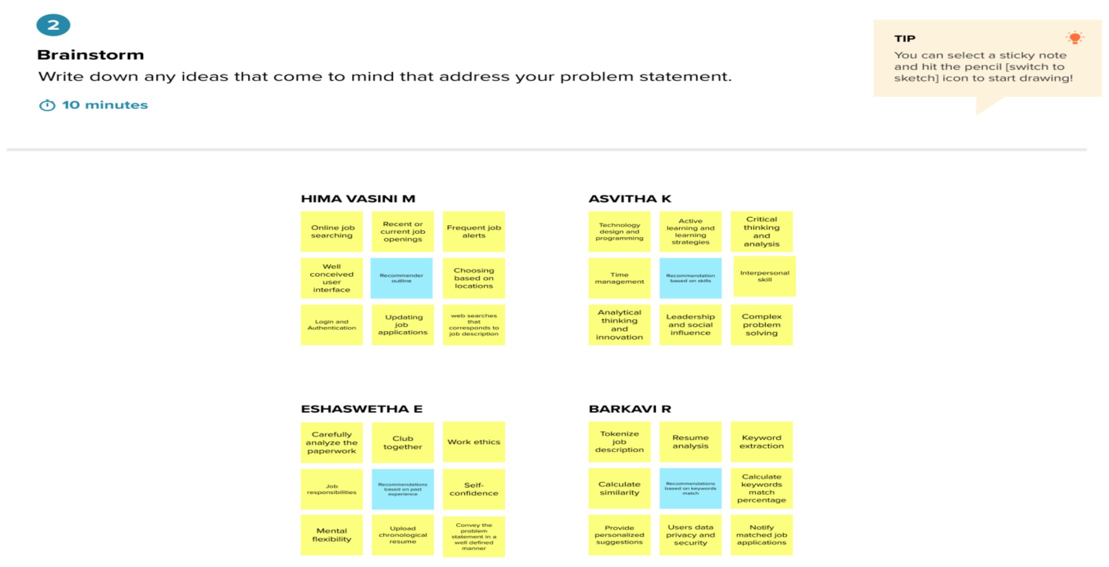
It is a useful tool to helps teams better understand their users.

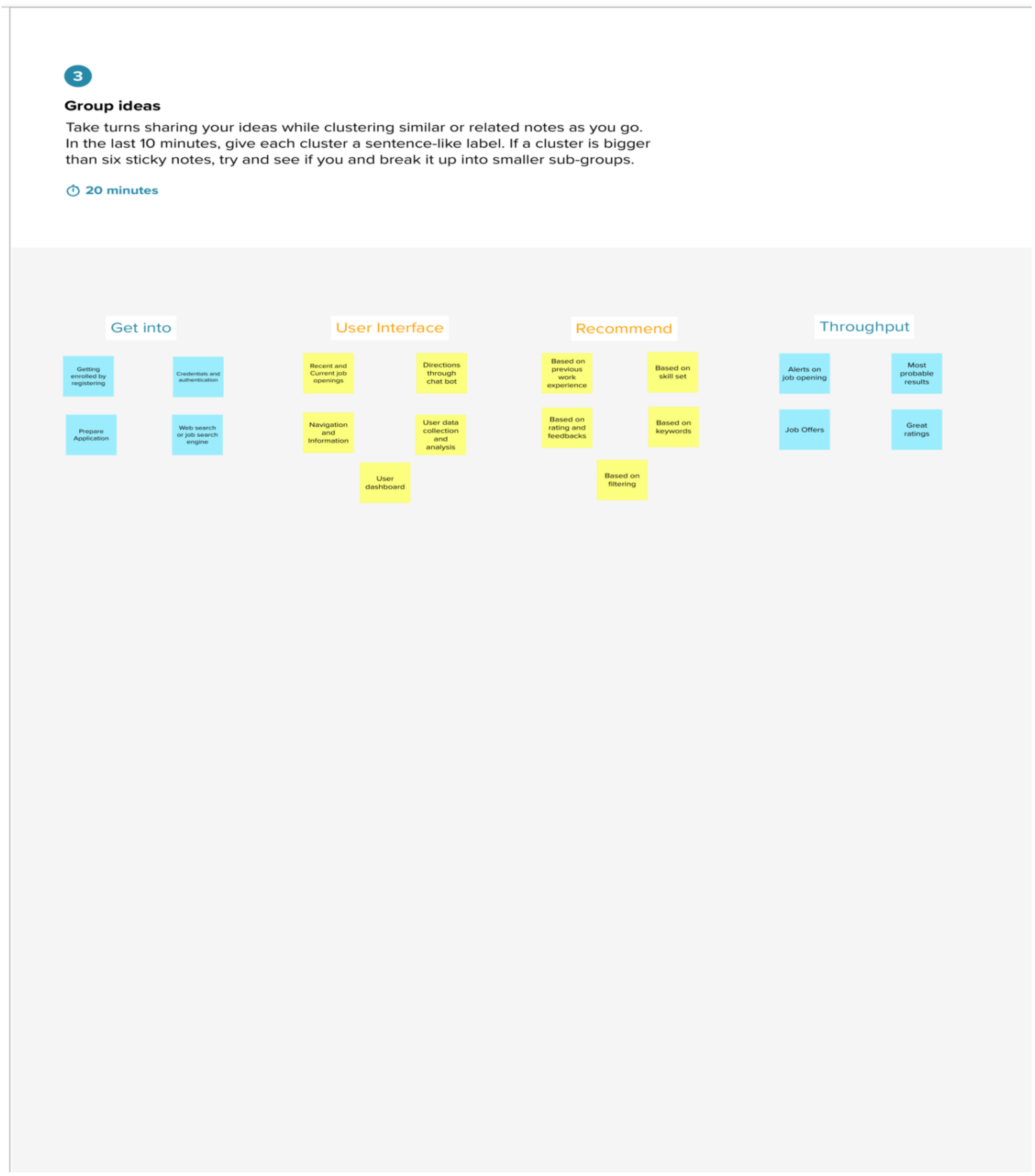
Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.


What do they THINK AND FEEL?
what really counts
major preoccupations
worries & aspirations
Hopefully good things will happen!
I want to know more about job
How people successfully get job?
I want to talk with somebody
Don't know where to start
I don't know if my direction correct
Is my career and job are in same phase
Doing web search is efficient
Will my job and life be in correct balance
process is too long and slow
What do they HEAR?
what friends say
what boss say
what influencers say
Keep your emotion out of it
Listen to what they say
Keep positive relations with colleagues
keep your self always updated
Concentrate on what your doing
Keep it confidential
Complete your process
Support others and make them learn
Change accordingly to situation
Are you really skilled?
What do they SEE?
environment
friends
what the market offers
MNC life
High paying jobs
Foreign job offers
Lifestyle development
Exploring great work atmosphere
How people respect them and acknowledge
Enhancing communication
Grooming skills
Offers and call letters
Transportation facilities
What do they SAY AND DO?
attitude in public
appearance
behavior towards others
Will i be fit into this job
Ho my god! what a competation
What if i don't take this job
I'll definitely find my dream job
Applying for jobs
Surfing job search portals
Uploading resumes
Email and phone calls
We all make mistakes!
PAIN
fears
frustrations
obstacles
Feeling that not respected by others
Loss of confidence
Is my personal details are safe to share
Will this really works!
Self elimination
Lack of skills needed
Feeling nervous
GAIN
"wants" / needs
measures of success
obstacles
More job opportunities
Feeling comfortable
Job within hands by one stop search
Respectful position
Resource full
Getting job alerts
Easy job search from home
SUCCESS

3.2 IDEATION & BRAINSTORMING

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.


2
Brainstorm
Write down any ideas that come to mind that address your problem statement.
10 minutes
TIP
You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!
HIMA VASINI M
Online job searching
Recent or current job openings
Frequent job alerts
Well conceived user interface
Recommender outline
Choosing based on locations
Login and Authentication
Updating job applications
web searches that corresponds to job description
ASVITHA K
Technology design and programming
Active learning and learning strategies
Critical thinking and analysis
Time management
Recommendation based on skills
Interpersonal skill
Analytical thinking and innovation
Leadership and social influence
Complex problem solving
ESHASWETHA E
Carefully analyze the paperwork
Club together
Work ethics
Job responsibilities
Recommendations based on past experience
Self-confidence
Mental flexibility
Upload chronological resume
Convey the problem statement in a well defined manner
BARKAVI R
Tokenize job description
Resume analysis
Keyword extraction
Calculate similarity
Recommendations based on keywords match
Calculate keywords match percentage
Provide personalized suggestions
Users data privacy and security
Notify matched job applications



3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

Get into
Getting enrolled by registering
Credentials and authentication
Prepare Application
Web search or job search engine
User Interface
Recent and Current job openings
Directions through chat bot
Navigation and Information
User data collection and analysis
User dashboard
Recommend
Based on previous work experience
Based on skill set
Based on rating and feedbacks
Based on keywords
Based on filtering
Throughput
Alerts on job opening
Most probable results
Job Offers
Great ratings

3.3 PROPOSED SOLUTION

1.Problem Statement (Problem to be solved)

- Finding a job with Work-LifeBalance.
- Lack of person to assist or guide.
- Finding the right organization whichoffers great Perks and Benefit.
- Lack getting recommendation fromexpertise person.

2.Idea / Solution description

- Chat Bot to recommend right job.
- Embedding Job Search engine.
- Job filters
- A job seeker can broadcast his/herresume to thousands of recruiters.
- Track job application

3.Novelty / Uniqueness

- Sending most appropriate job byintegrating with WhatsApp.
- Creating resume for job seeker basedon his/her choice of template.
- Motivational Videos.
- Foreign job opportunities.
- Network.

4.Social Impact / Customer Satisfaction

- Lower Work and Overhead.
- Finding great jobs.
- Job seeker will get up to date information about what industriesare expecting.
- Effective Response

5.Business Model (Revenue Model)

- Profile enhancement by experts forbetter performance.
- Mock interviews
- Providing seminar and webinar

6.Scalability of the Solution

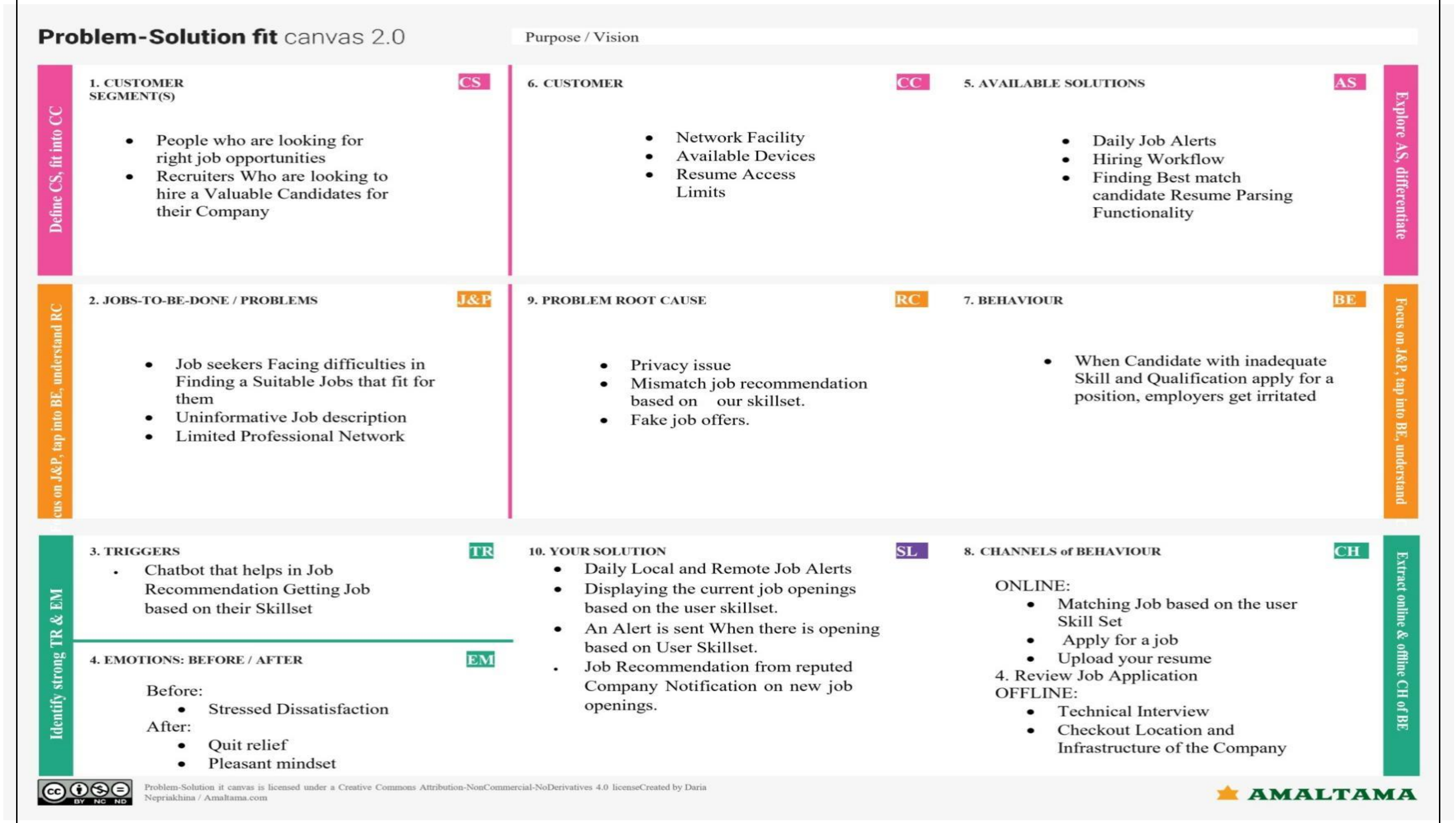
- Our application takes less time to respond.
- Our application will response many requests simultaneously without making any server crashes.
- Providing Good response time between user request and application response.
- Large database to provide access formore job seekers.
- Having high configured hardwareresources.

3.4 PROPOSED SOLUTION FIT

The Problem-Solution Fit simply means that you have found a problem with your customer and that the solution you have realized for it actually solves the customer's problem. It helps entrepreneurs, marketers and corporate innovators identify behavioral patterns and recognize what would work.

PURPOSE:

- 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 touchpoints 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.



Problem-Solution fit canvas 2.0

Purpose / Vision

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS
- People who are looking for right job opportunities
- Recruiters Who are looking to hire a Valuable Candidates for their Company

6. CUSTOMER CC
- Network Facility
- Available Devices
- Resume Access Limits

5. AVAILABLE SOLUTIONS AS
- Daily Job Alerts
- Hiring Workflow
- Finding Best match candidate Resume Parsing Functionality

Explore AS, differentiate

cus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P
- Job seekers Facing difficulties in Finding a Suitable Jobs that fit for them
- Uninformative Job description
- Limited Professional Network

9. PROBLEM ROOT CAUSE RC
- Privacy issue
- Mismatch job recommendation based on our skillset.
- Fake job offers.

7. BEHAVIOUR BE
- When Candidate with inadequate Skill and Qualification apply for a position, employers get irritated

Focus on J&P, tap into BE, understand

Identify strong TR & EM

3. TRIGGERS TR
- Chatbot that helps in Job Recommendation Getting Job based on their Skillset

4. EMOTIONS: BEFORE / AFTER EM

Before:
- Stressed Dissatisfaction

After:
- Quit relief
- Pleasant mindset

10. YOUR SOLUTION SL
- Daily Local and Remote Job Alerts
- Displaying the current job openings based on the user skillset.
- An Alert is sent When there is opening based on User Skillset.
- Job Recommendation from reputed Company Notification on new job openings.

8. CHANNELS of BEHAVIOUR CH

ONLINE:
- Matching Job based on the user Skill Set
- Apply for a job
- Upload your resume

4. Review Job Application

OFFLINE:
- Technical Interview
- Checkout Location and Infrastructure of the Company

Extract online & offline CH of BE

AMALTAMA

CHAPTER-4

REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

Functional requirements define the internal workings of the software: that is, the technical details, data manipulation and processing and other specific functionality that show how the use cases are to be satisfied. They are supported by non-functional requirements, which impose constraints on the design or implementation.

FR-1 User Registration - The user can register through website by providing valid email.

FR-2 User Confirmation - Confirmation via Email Confirmation via OTP.

FR-3 User login - User should login with their respective username and password.

FR-4 User Interaction - All the user can interact and get recommendations through chatbot.

FR-5 Search Jobs - The user can search job through job search API.

FR-6 Job Notification – The website will automatically notify the user when there is current job openings.

Software Required:

Python, Flask, Docker

System Required:

8GB RAM, Intel Core i3,OS-Windows/Linux/MAC ,Laptop or Desktop

4.2 NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements are requirements which specify criteria that can be used to judge the operation of a system, rather than specific behaviours. This should be contrasted with functional requirements that specify specific behaviour or functions. Typical non-functional requirements are reliability, scalability, and cost. Non-functional requirements are often called the ilities of a system. Other terms for non-functional requirements are "constraints", "quality attributes" and "quality of service requirements".

NFR-1 Usability - Everyone can easily install and use from the play store by using the instructions in the app.

NFR-2 Security - The application is very secure and confidential and stored in the IBM cloud.

NFR-3 Performance - The app can work fast and more reliable.

NFR-4 Availability - It is available in all kind of play store and the service available for 24/7.

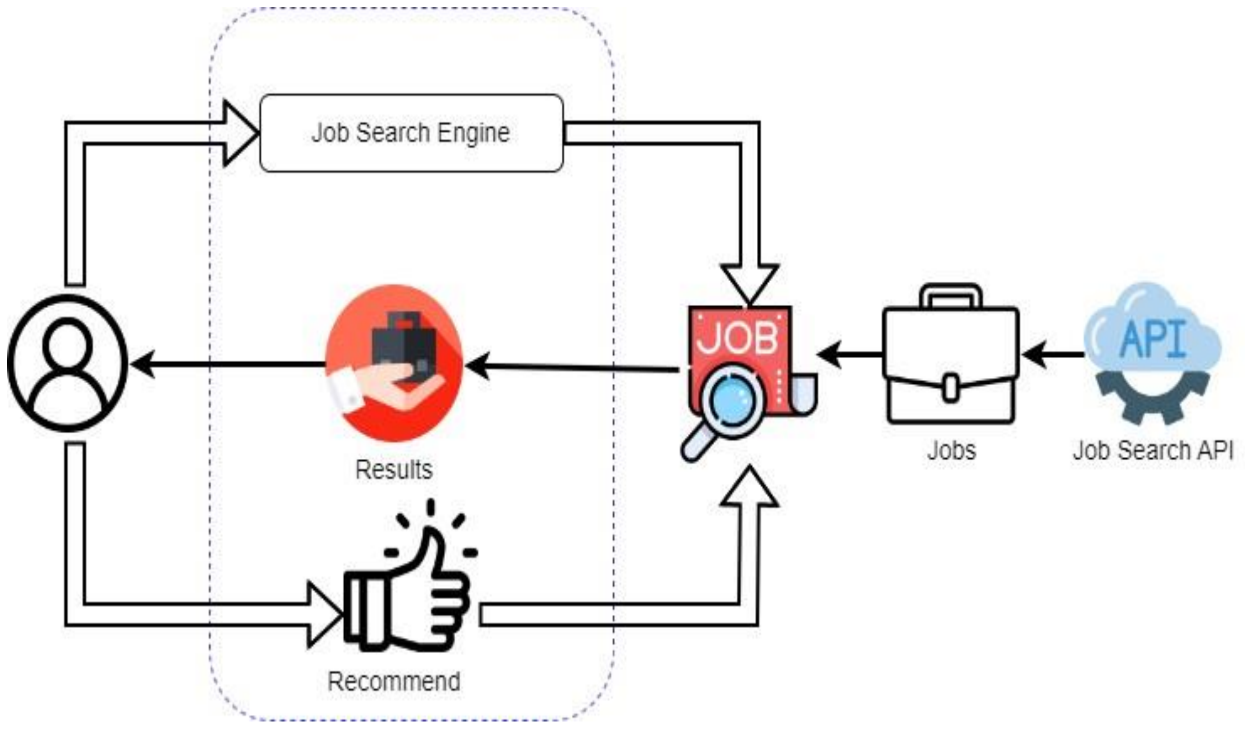
NFR-5 Scalability - Our application takes less time to response many requests simultaneously without making any server crashes, so this will ensure our application will scalable.

CHAPTER-5

PROJECT DESIGN

5.1 DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can graphically depict the right amount of the system requirement. It shows how data enters and leaves the system, what changes the information, and where data is stored.



5.2 SOLUTION & TECHNICAL ARCHITECTURE

Solution Architecture

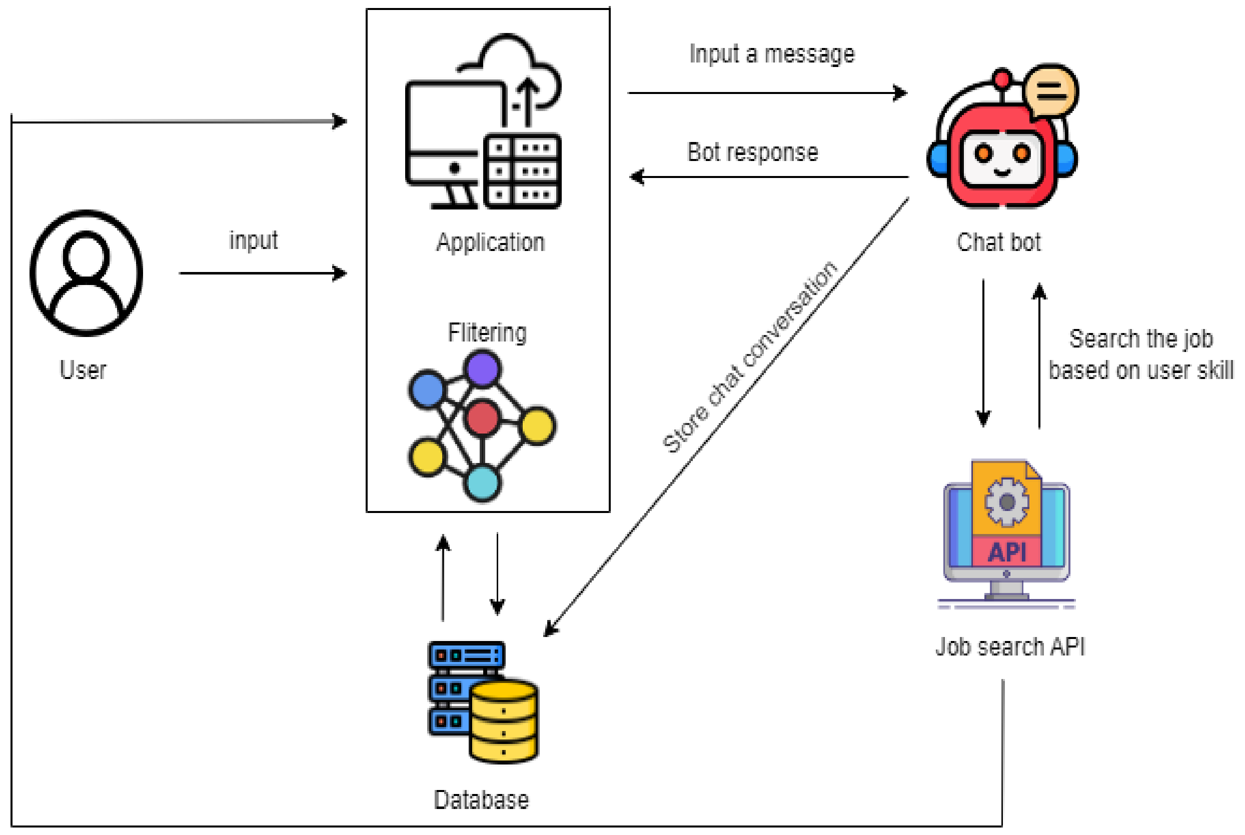
Stores job candidate information like resumes, cover letters, references, and other recruitment and hiring data that HR teams can easily access and organize .

- Tracks job candidates and their application status.

- Recommends the best fit for a position based on the parameters set by HR. Only those on the shortlist are moved to the next stage of the hiring process.

- Automatically sends notification and emails to job candidates and employees.

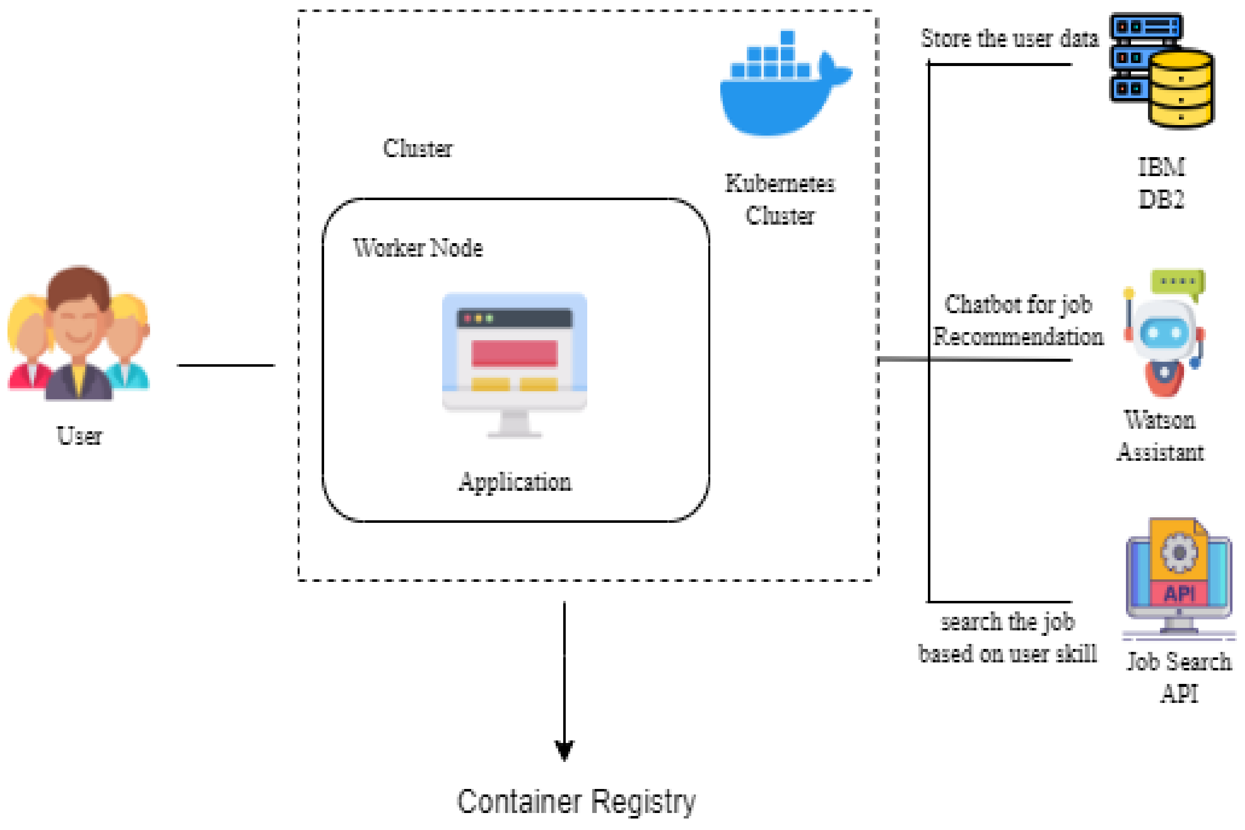
- Filtering uses item features to recommend other items similar to what the user likes, based on their previous actions or explicit feedback



Technical Architecture

Kubernetes is an orchestration tool for containerized applications. Kubernetes can control resource allocation and traffic management for cloud applications , it simplifies many aspects of running a service-oriented application infrastructure.

Worker nodes within the Kubernetes cluster are used to run containerized applications and handle networking to ensure that traffic between applications across the cluster and from outside of the cluster can be properly facilitated.



5.3 USER STRORIES

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	UNS-6	As a user, I can access the dashboard after signing in.		High	Sprint-1
Customer (Web user)	Registration/login	UNS-7	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
Customer Care Executive	Contact with Customers	UNS-8	As a customer care executive, I solve the customer requirements and feedback	I can receive calls from customer	High	Sprint-1

Administrator	Check jobs, requirements of job seeker, update jobs	UNS-9	As an administrator, I can check the database for new job alerts, post jobs, update new jobs	I am the administrator of the Company	High	Sprint-1

CHAPTER-6

PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION

Sprint	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Priority**	**Team Members**
Sprint-1	Registration	USN-1	UI Creation Creating Registration page, Login page	Medium	HIMA VASINI M ASVITHA K
Sprint-1	Database Connectivity	USN-2	Connecting database with UI	High	HIMA VASINI M
Sprint-2	Chatbot Development	USN-3	Building a chatbot	Medium	BARKAVI R ESHASWETHA E
Sprint-2	Integration	USN-4	Integrating chatbot to the HTML page	Low	HIMA VASINI M ESHASWETHA
Sprint-3	View jobs or apply	USN-5	Able to visit application, get recommendations based on job seeker skill and apply if needed.	High	HIMA VASINI M ASVITHA K BARKAVI R ESHASWETHA E
Sprint-3	SendGrid Integration	USN-6	SendGrid Integration with Python Code	Medium	ASVITHA K BARKAVI R

Sprint -4	Upload Image and deployment	USN-7	Upload the image to the IBM Registry and deploy it in the Kubernetes Cluster.	High	ASVITHA K BARKAVI R ESHASWETHA E HIMA VASINI M
Sprint-4	Containerizing and deploying	USN-8	Containerizing app and deploying it in IBM cloud	High	ASVITHA K BARKAVI R ESHASWETHA E HIMA VASINI M

6.2 SPRINT DELIVERY SCHEDULE

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	3 Days	24 Oct 2022	27 Oct 2022	20	29 Oct 2022
Sprint-2	20	1 Days	1 Nov 2022	2 Nov 2022	20	5 Nov 2022
Sprint-3	20	7 Days	3 Nov 2022	10 Nov 2022	20	12 Nov 2022
Sprint-4	20	10 Days	11Nov 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)

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.

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

https://makemycharts.atlassian.net/jira/software/projects/HIMAboards/1

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

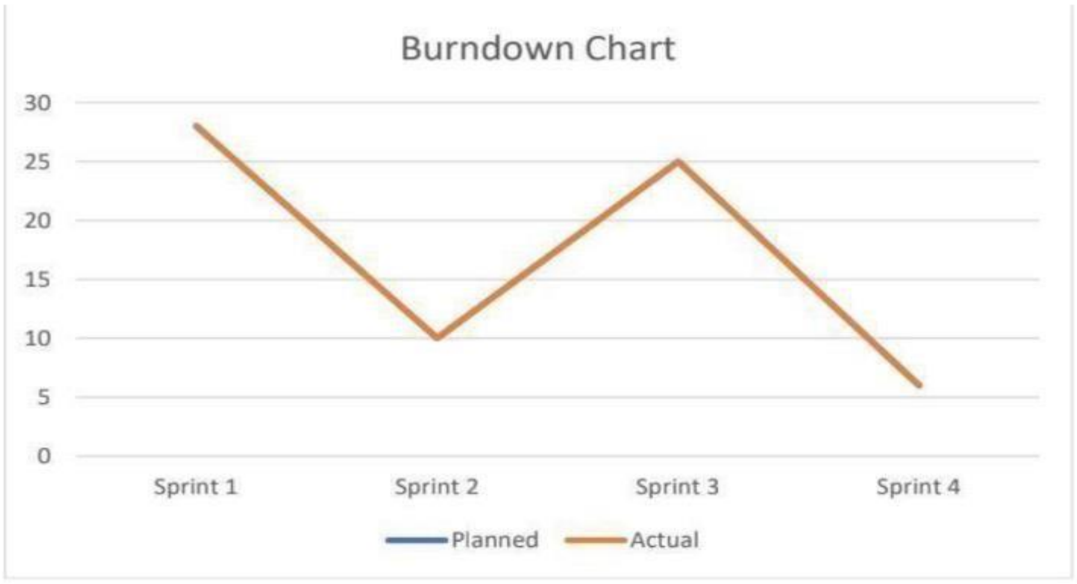
burndown-chart/

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

6.3 REPORTS FROM JIRA

Jira Software is part of a family of products designed to help teams of all types manage work. Originally, Jira was designed as a bug and issue tracker. But today, Jira has evolved into a powerful work management tool for all kinds of use cases, from requirements and test case management to agile software development.

Jira is one of the best open-source tools for planning and tracking in Agile methodology. Development teams use Jira for tracking bugs and projects, managing Scrums, and visualizing workflows with Kanban boards. Workflows in Jira make it easy to plan, track, release, and report on software.



CHAPTER-7

CODING & SOLUTIONING

7.1 FEATURE 1

Index.html

```
<!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">

    <! --<link rel="stylesheet" href="style.css" type="text/css"/>-->

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">

    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">

    <title>Home Page</title>

    <style>

        body{

           background-color: white; }

        nav{

    margin:5px;

    padding:5px;

    height:50px;

    width:100%;

    background-color: rgb(103, 117, 197);
}
.home{
    margin-right:15px;
    margin-top:20px;
    color:rgb(250, 249, 249);
    font-size: 20px;
}
.navlist{
    float:right;
    margin-right:20px;
}
.navlist li{
    display: inline-block;
    list-style-type: none;
    margin:5px;
    text-transform:uppercase;
    font-size:20px;
}
.navlist li a{
    color:white;
}
a{
    cursor:pointer;
}
```

```
#menu{
   display:none;
}
.checkbtn{
   display:none;
}

.reg{
      margin:0;
      position:absolute;
      top:50%;
      left:32%;
      -ms-transform:translateY(-50%);
      transform:translateY(-50%);
   }
@media(max-width: 800px){
   .checkbtn{
      float:right;
      display:block;
      font-size: 40px;
      margin-right:20px;
   }
   .home{
      display:block;
      padding:15px;
```

```
    }
    .navlist {
        display:block;
        color:black;
        position:fixed;
        width:30%;
        height:100vh;
        background: #4133bb;
        top:55px;
        right:0;
        text-align:center;
    }
    .navlist li{
        display:block;
        padding:15px;
    }

}
</style>
</head>
<body>
    <nav>
       <input type="checkbox" id="menu">
        <label for="menu" class="checkbtn">
```

```
            <i class="fa fa-bars"></i>
        </label>
        <a class="home" href="index.html">RECOMMEND ME SKILL/JOB</a>
        <ul class="navlist">
            <li><a href="/about">about</a></li>
            <li><a href="/contact">contact</a></li>
            <li><a href="/login">Login</a></li>
        </ul>
    </nav>


    <div class="reg">
        <a href="/recommend"><button type="button" class="btn btn-primary btn-lg btn-
block">RECOMMEND JOB THAT MATCHES WITH SKILLSET</button></a>
        <a href="/jobseeker"><button type="button" class="btn btn-secondary btn-lg btn-
block">REGISTER AS JOB SEEKER</button></a>
        <a href="/organization"><button type="button" class="btn btn-secondary btn-lg btn-
block">REGISTER AS ORGANIZATION</button></a>
    </div>
</body>
</html>
```

Registerasjobseeker.html

```
<!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>Regiter As Organization</title>

    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">

</head>

<body>

<section class="vh-100" style="background-color: rgb(255, 255, 255);">

  <div class="container h-100">

    <div class="row d-flex justify-content-center align-items-center h-100">

      <div class="col-lg-12 col-xl-11">

        <div class="card text-black" style="border-radius: 25px;">

          <div class="card-body p-md-5">

            <div class="row justify-content-center">

              <div class="col-md-10 col-lg-6 col-xl-5 order-2 order-lg-1">


                <p class="text-center h1 fw-bold mb-5 mx-1 mx-md-4 mt-4">Lets Start!</p>


                <form  method="POST" class="mx-1 mx-md-4">


                  <div class="d-flex flex-row align-items-center mb-4">

                    <i class="fas fa-user fa-lg me-3 fa-fw"></i>

                    <div class="form-outline flex-fill mb-0">

                      <input type="text" id="form3Example1c" class="form-control" name="name"/>

                      <label class="form-label" for="form3Example1c">Your Name</label>

                    </div>
```

```
</div>


<div class="d-flex flex-row align-items-center mb-4">
  <i class="fas fa-lock fa-lg me-3 fa-fw"></i>
  <div class="form-outline flex-fill mb-0">
    <input type="email" id="form3Example4c" class="form-control" name="email"/>
    <label class="form-label" for="form3Example4c">Email</label>
  </div>
</div>


<div class="d-flex flex-row align-items-center mb-4">
  <i class="fas fa-lock fa-lg me-3 fa-fw"></i>
  <div class="form-outline flex-fill mb-0">
    <input type="password" id="form3Example4c" class="form-control"
name="password"/>
    <label class="form-label" for="form3Example4c">password</label>
  </div>
</div>


<div class="d-flex flex-row align-items-center mb-4">
  <i class="fas fa-lock fa-lg me-3 fa-fw"></i>
  <div class="form-outline flex-fill mb-0">
    <input type="number" id="form3Example4c" class="form-control" name="age"/>
    <label class="form-label" for="form3Example4c">Your Age</label>
  </div>
```

```
</div>

<!----- <div class="d-flex flex-row align-items-center mb-4">
  <i class="fas fa-lock fa-lg me-3 fa-fw"></i>
  <div class="form-outline flex-fill mb-0">
    <input type="text" id="form3Example4c" class="form-control" name="address"/>
    <label class="form-label" for="form3Example4c">Your Address</label>
  </div>
</div>-->

<div class="d-flex flex-row align-items-center mb-4">
  <i class="fas fa-lock fa-lg me-3 fa-fw"></i>
  <div class="form-outline flex-fill mb-0">
    <input type="text" id="form3Example4c" class="form-control" name="skills"/>
    <label class="form-label" for="form3Example4c">Your Skills</label>
  </div>
</div>

<div class="d-flex flex-row align-items-center mb-4">
  <i class="fas fa-lock fa-lg me-3 fa-fw"></i>
  <div class="form-outline flex-fill mb-0">
    <input type="text" id="form3Example4c" class="form-control"
name="expectedjob" />
    <label class="form-label" for="form3Example4c">Expected jobRole</label>
  </div>
```

```
                  </div>


                  <div class="d-flex justify-content-center mx-4 mb-3 mb-lg-4">
                    <input type="submit" class="btn btn-primary btn-lg"></input>
                  </div>
                  <!---- <a href="/jobseeker"> <input type="submit" class="btn btn-primary btn-
lg">Register</submit></a>
                  </div>-->


                </form>


              </div>
              <div class="col-md-10 col-lg-6 col-xl-7 d-flex align-items-center order-1 order-lg-2">



                <img src="https://mdbcdn.b-cdn.net/img/Photos/new-templates/bootstrap-
registration/draw1.webp" height="500px"
width="500px" class="img-fluid" alt="Sample image">


              </div>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>
```

```
</section>

</body>
```

Login.html

```
<!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">

    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">

    <title>Login</title>

    <style>

      .space

      {

        position:absolute;

        top:200px;

        left:200px;

      }

.divider:after,

.divider:before {

content: "";

flex: 1;

height: 1px;
background: #eee;
}
.h-custom {
height: calc(100% - 73px);
}
@media (max-width: 450px) {
.h-custom {
height: 100%;
}
}
</style>
</head>
<body>

   <section class="vh-100">
     <div class="container-fluid h-custom">
       <div class="row d-flex justify-content-center align-items-center h-100">
         <div class="col-md-9 col-lg-6 col-xl-5">
           <img src="https://mdbcdn.b-cdn.net/img/Photos/new-templates/bootstrap-login-
form/draw2.webp"
             class="img-fluid" alt="Sample image">
         </div>
         <div class="col-md-8 col-lg-6 col-xl-4 offset-xl-1">
           <form action="/login" method="POST">
```

```
<!-- Email input -->
<div class="space">
<div class="form-outline mb-4">
  <input type="email" id="form3Example3" class="form-control form-control-lg"
    placeholder="Enter a valid email address" />
  <label class="form-label" for="form3Example3">Email address</label>
</div>

<!-- Password input -->
<div class="form-outline mb-3">
  <input type="password" id="form3Example4" class="form-control form-control-lg"
    placeholder="Enter password"  />
  <label class="form-label" for="form3Example4">Password</label>
</div>

<div class="d-flex justify-content-between align-items-center"></div>

<div class="text-center text-lg-start mt-4 pt-2">
  <button type="button" class="btn btn-primary btn-lg"
    style="padding-left: 2.5rem; padding-right: 2.5rem;">Login</button>
  <p class="small fw-bold mt-2 pt-1 mb-0">Don't have an account? <a
href="/jobseeker"
      class="link-danger">Register</a></p>
</div>
</div>
```

```
            </form>
            <p class="message">{{msg}}</p>
          </div>
        </div>
      </div>
      <div
        class="d-flex flex-column flex-md-row text-center text-md-start justify-content-between
py-4 px-4 px-xl-5 bg-primary">
      </div>
    </section>

</body>
</html>
</html>
```

7.2 FEATURE 2

app.py

```
import csv
from datetime import datetime

#from flask_db2 import DB2
#import ibm_db
import ibm_db
import requests
from bs4 import BeautifulSoup
from flask import Flask, render_template, request, url_for

import os

#import sendgrid

#from sendgrid import SendGridAPIClient

#from sendgrid.helpers.mail import *



dsn_hostname = "0c77d6f2-5da9-48a9-81f8-
86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud"

dsn_uid = "why11914"

dsn_pwd = "qMCrggTKEsZ8b54q"

dsn_driver = "{IBM DB2 ODBC DRIVER}"

dsn_database = "bludb"

dsn_port = "31198"

dsn_protocol = "TCPIP"

dsn_security = "SSL"

dsn = (

    "DRIVER={0};"

    "DATABASE={1};"

    "HOSTNAME={2};"

    "PORT={3};"

    "PROTOCOL={4};"

    "UID={5};"

    "PWD={6};"

    "SECURITY={7};").format(dsn_driver, dsn_database, dsn_hostname, dsn_port, dsn_protocol,
dsn_uid, dsn_pwd,dsn_security)
```

```
conn = ibm_db.connect(dsn, "", "")


print(conn)

print("Connection Successful...........")

app=Flask(__name__)


@app.route('/')

def welcome():

    return render_template("index.html")

@app.route('/jobseeker',methods=['POST','GET'])

def jobseeker():

    if request.method == 'POST':

        name = request.form['name']

        email = request.form['email']

        password = request.form['password']

        age = request.form['age']

        skills = request.form['skills']

        expectedjob = request.form['expectedjob']


        #if password!=cp:

            #return render_template('register.html')


        try:

            insert_sql = "INSERT INTO REGISTER_AS_REGISTER VALUES (?,?,?,?,?,?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)

            ibm_db.bind_param(prep_stmt, 1, name)

            ibm_db.bind_param(prep_stmt, 2, email)

            ibm_db.bind_param(prep_stmt, 3, password)

            ibm_db.bind_param(prep_stmt, 4, age)

            ibm_db.bind_param(prep_stmt, 5, skills)

            ibm_db.bind_param(prep_stmt, 6, expectedjob)

            ibm_db.execute(prep_stmt)

            #sql = "insert into REGISTER_AS_REGISTER values ('{}','{}','{}','{}','{}','{}')".format(
name, email, password, age, skills, expectedjob)

            #stmt = ibm_db.exec_immediate(conn,sql)

            print("No of Affected rows: ",ibm_db.num_rows(prep_stmt))

        except:

            print("Error: ",ibm_db.stmt_errormsg())

    return render_template("registerasjobseeker.html")

@app.route('/organization',methods=['POST','GET'])

def organization():

    if request.method == 'POST':

        organization_name = request.form['organization_name']

        job_role = request.form['job_role']

        lpa = request.form['lpa']

        #address = request.form['address']

        city = request.form['city']

        email = request.form['email']

        mobile = request.form['mobile']
        #if password!=cp:
            #return render_template('register.html')


        try:
            insert_sql = "INSERT INTO REGISTER_AS_ORGANIZATION VALUES (?,?,?,?,?,?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prep_stmt, 1, organization_name)
            ibm_db.bind_param(prep_stmt, 2, job_role)
            ibm_db.bind_param(prep_stmt, 3, lpa)
            ibm_db.bind_param(prep_stmt, 4, city)
            ibm_db.bind_param(prep_stmt, 5, email)
            ibm_db.bind_param(prep_stmt, 6, mobile)
            ibm_db.execute(prep_stmt)
            #sql = "insert into REGISTER_AS_REGISTER values ('{}','{}','{}','{}','{}','{}')".format(
name, email, password, age, skills, expectedjob)
            #stmt = ibm_db.exec_immediate(conn,sql)
            print("No of Affected rows: ",ibm_db.num_rows(prep_stmt))
        except:
            print("Error: ",ibm_db.stmt_errormsg())
    return render_template("registerasorganization.html")
@app.route('/login',methods=['POST','GET'])
def login():
    if request.method == 'POST':
        email = request.form.get('email')
        password = request.form.get('password')


        sql = "SELECT * FROM REGISTER_AS_REGISTER email =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, email)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)


        if account:
            if (password == str(account['PASS']).strip()):
                return render_template('index.html')
            else:
                return render_template('index.html', msg="Password is invalid")
        else:
            return render_template('registerasjobseeker.html')
    else:
        return render_template('login.html')
@app.route('/about')
def about():
    return render_template('about.html')
@app.route('/recommend')
def recommend():
    return render_template('recommend.html')
@app.route('/apply_job')
def apply_job():
    return render_template('apply_job.html')
@app.route('/applyforjob',methods=['POST','GET'])
def applyforjob():
    if request.method == 'POST':
        resume = request.form.get('resume')
        email = request.form.get('email')
        message =
Mail(from_email=email,to_emails='mail2himaraj@gmail.com',subject='Resume',html_content=r
esume)

sendgrid.SendGridAPIClient(apikey='SG.zLNCF9gKR2at5dcxotfDZQ.80Hp6z9OXXLf4OsV2s
FUHjrebzUy6J-yydwoFyFAmsk')
        from_email=Email("mail2himaraj@gmail.com")
        to_email=Email("mail2himaraj@gmail.com")
        subject="mail"
        content=Content("text/plain","easy")
        mail=Mail(from_email,subject,to_email,content)
        response=sg.client.mail.send.post(request_body=mail.get())'''
    return render_template('upload_resume.html')
@app.route('/manualrecommendation',methods=['POST','GET'])
def manualrecommendation():
    if request.method == 'POST':
        jobrole= request.form.get('jobrole')
        location= request.form.get('location')
        template='https://in.indeed.com/jobs?q={}&l={}&from=searchOnHP'
        url=template.format(jobrole, location)
        return url
    return render_template('manual.html')
#url=manualrecommendation('jobrole','location')
#print(url)
@app.route('/chatbotrecommendation')
def chatbotrecommendation():
    return render_template("chatbot.html")
@app.route('/chatbotrecon')
def chatbotcon():
    return render_template("chatconversation.html")
if __name__=="__main__":
    port=int(os.environ.get('PORT',5000))
    app.run(port=port,host="0.0.0.0")
```

Applyjob.html

```
<!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>View Jobs</title>
    <style>
        .container{
            display:inline;
        }
        .box{
            height:230px;
            width:100%;
            border:1px solid none;
            background-color: rgb(230, 242, 252);
            border-radius: 10px;
        }
        .btn{
            background-color: rgb(10, 221, 221);
            border:rgb(152, 243, 243);
            border-radius: 5px;
            cursor:pointer;
        }
        </style>
</head>
<body>
    <div class="container">
        <div class="box">
            <h4>Cognizant</h4>
            <h3>Plsql Developer</h3>
            <p>skills:
               Oracle SQL, Advance PLSQL, Unix/Shell scripting
               Oracle 12c /19c
               Complex SQL query writing, Adv. PLSQL programming
```

```
      Hands on SQL/PLSQL/DB Performance Tuning Must for SA level positions .
      Good written/verbal communication skills</p>
    <p>experience:0-2 yrs</p>
    <p>Location:Chennai</p>
    <a href="/applyforjob"><button type="submit" class="btn" >APPLY</button></a>
</div>
<div class="box">
    <h4>Capgemini</h4>
    <h3>Programmer Analyst </h3>
    <p>skills:
      Ability to plan and direct complex technical projects.
      Experience with network protocols, computer hardware, and software.
      Excellent knowledge of data backups, recovery, security, integrity, and SQL.
      Great attention to details.
      Good written/verbal communication skills</p>
    <p>experience:0-2 yrs</p>
    <p>Location:Chennai</p>
    <a href="/upload_resume"><button type="submit" class="btn" >APPLY</button></a>
</div>
<div class="box">
    <h4>Tech Mahindra</h4>
    <h3>Software engineer</h3>
    <p>skills:
      Computer Programming and Coding
      Software development
```

```
              Software Testing and Debugging

              Good written/verbal communication skills</p>

            <p>experience:0-2 yrs</p>

            <p>Location:Chennai</p>

            <a href="/applyforjob"><button type="submit" class="btn" >APPLY</button></a>

        </div>


        </div>

</body>

</html>
```

Chatbot.html

```
<!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">

    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">

    <title>chatbot</title>

    <style>

        .center{

            margin:0;

            position:absolute;

            top:50%;
            left:45%;

            -ms-transform:translateY(-50%);

            transform:translateY(-50%);

        }

        </style>

</head>

<body><!---<div class="center">

    <button type="button" class="btn btn-secondary btn-lg btn-block">CHATBOT</button>

    </div>--->

        <script>

            window.watsonAssistantChatOptions = {

              integrationID: "889e133a-41ab-4db9-b03d-e94022406dc1", // The ID of this
integration.

              region: "au-syd", // The region your integration is hosted in.

              serviceInstanceID: "c72dc204-4442-4727-bf4b-86c09de2b0f6", // The ID of your
service instance.

              onLoad: function(instance) { instance.render(); }

            };

            setTimeout(function(){

              const t=document.createElement('script');

              t.src="https://web-chat.global.assistant.watson.appdomain.cloud/versions/" +
(window.watsonAssistantChatOptions.clientVersion || 'latest') + "/WatsonAssistantChatEntry.js";

              document.head.appendChild(t);

            });

          </script>

</body>
```

</html>

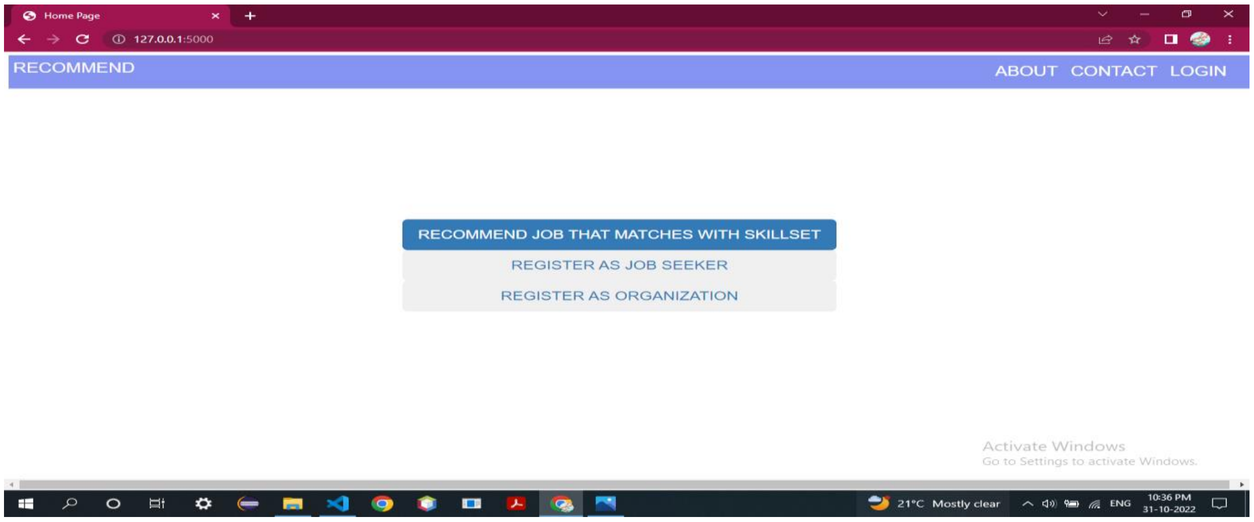
Docker file

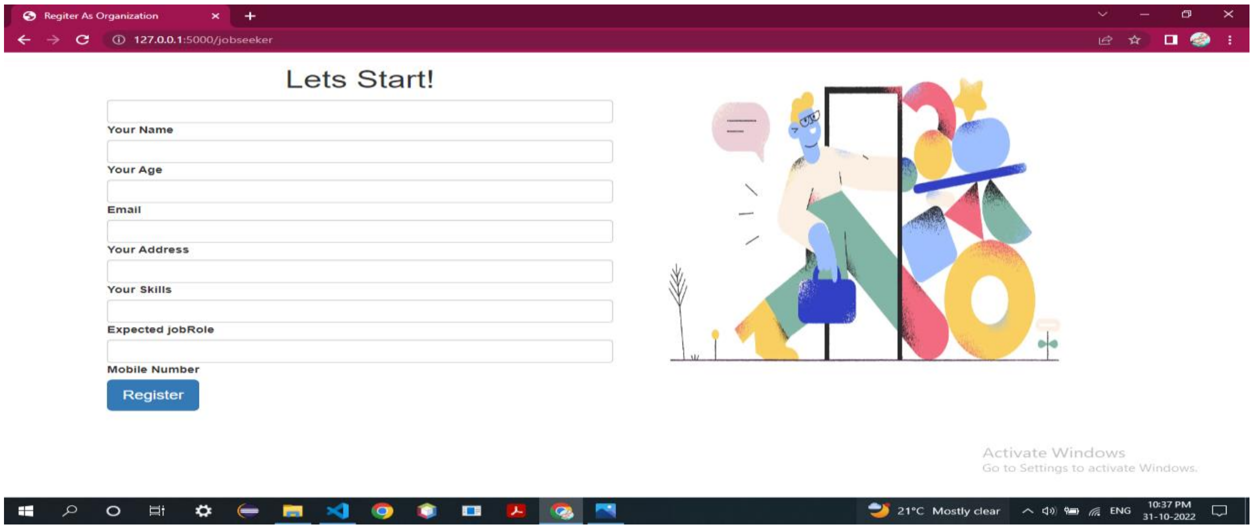
```
FROM python
WORKDIR /app
COPY . .
RUN pip install -r requirements.txt
CMD ["flask","run"]
EXPOSE 5000
```

Requirments.txt

```
flask
ibm_db
sendgrid
bs4
Requests
```

OUTPUT/SOLUTION





Lets Start!

Your Name

Your Age

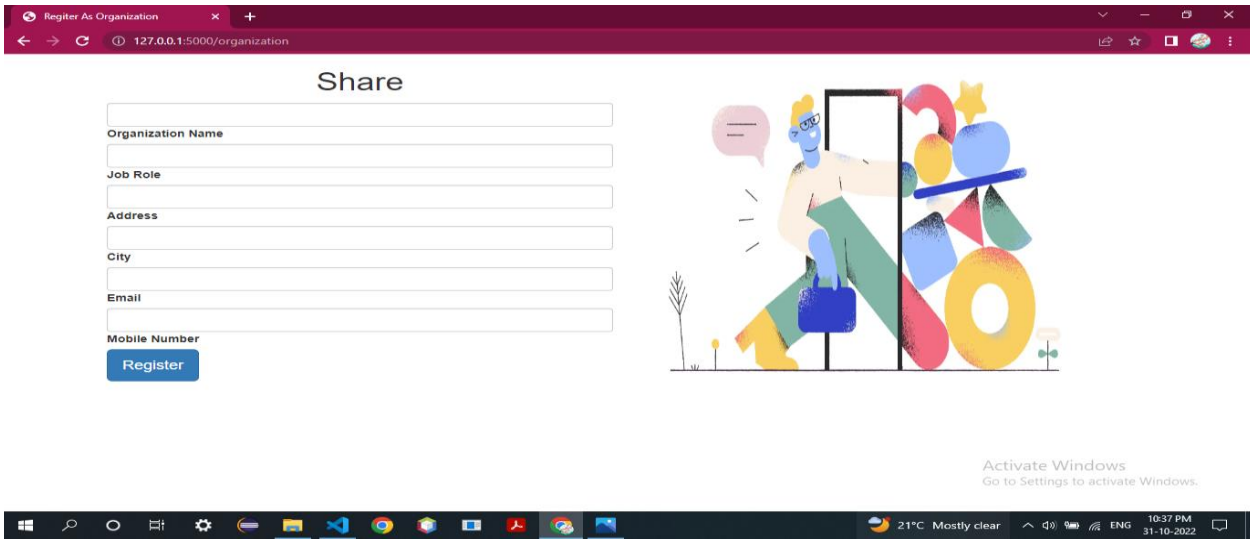
Email

Your Address

Your Skills

Expected jobRole

Mobile Number



Share

Organization Name

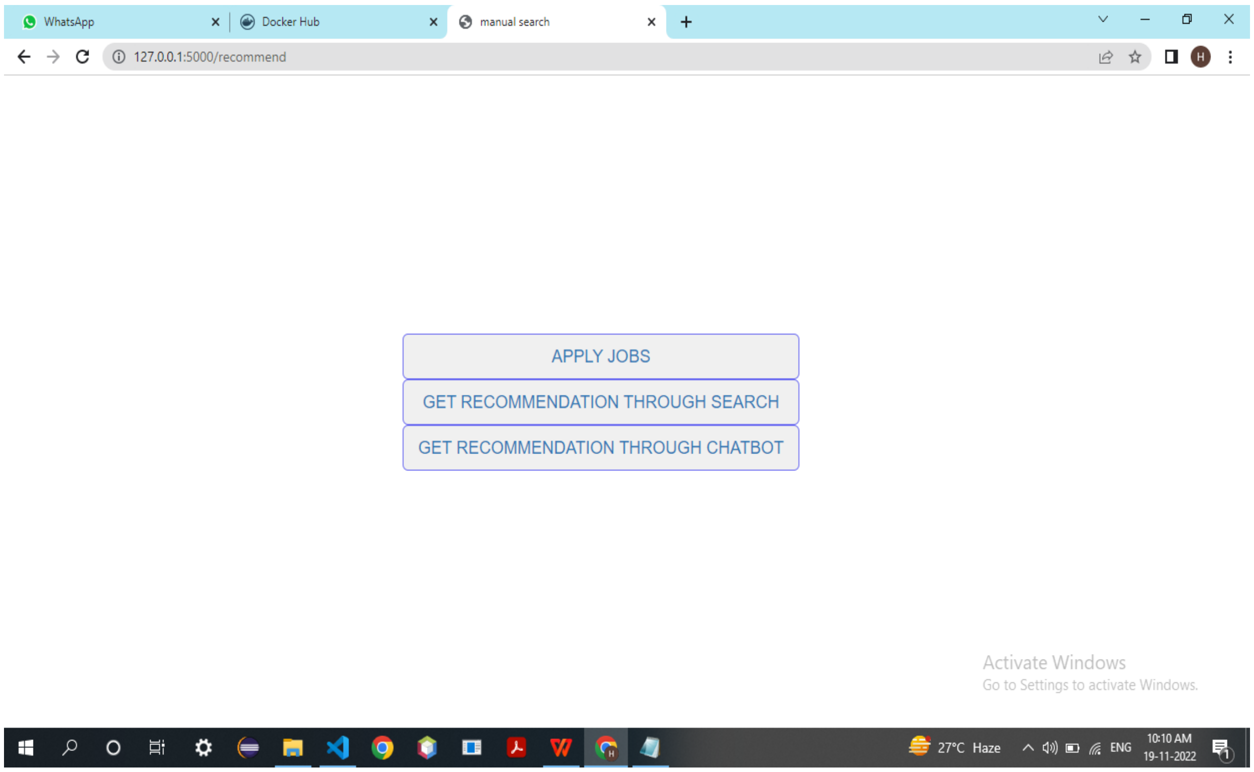
Job Role

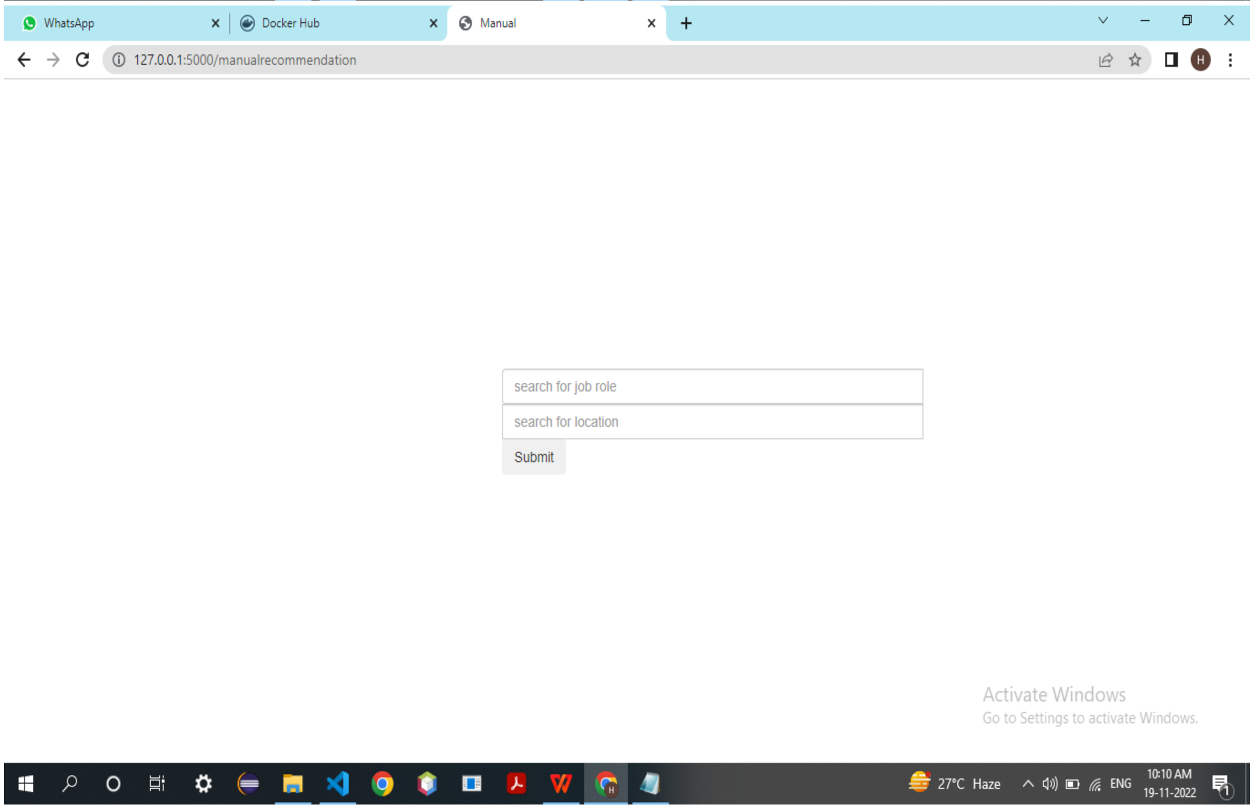
Address

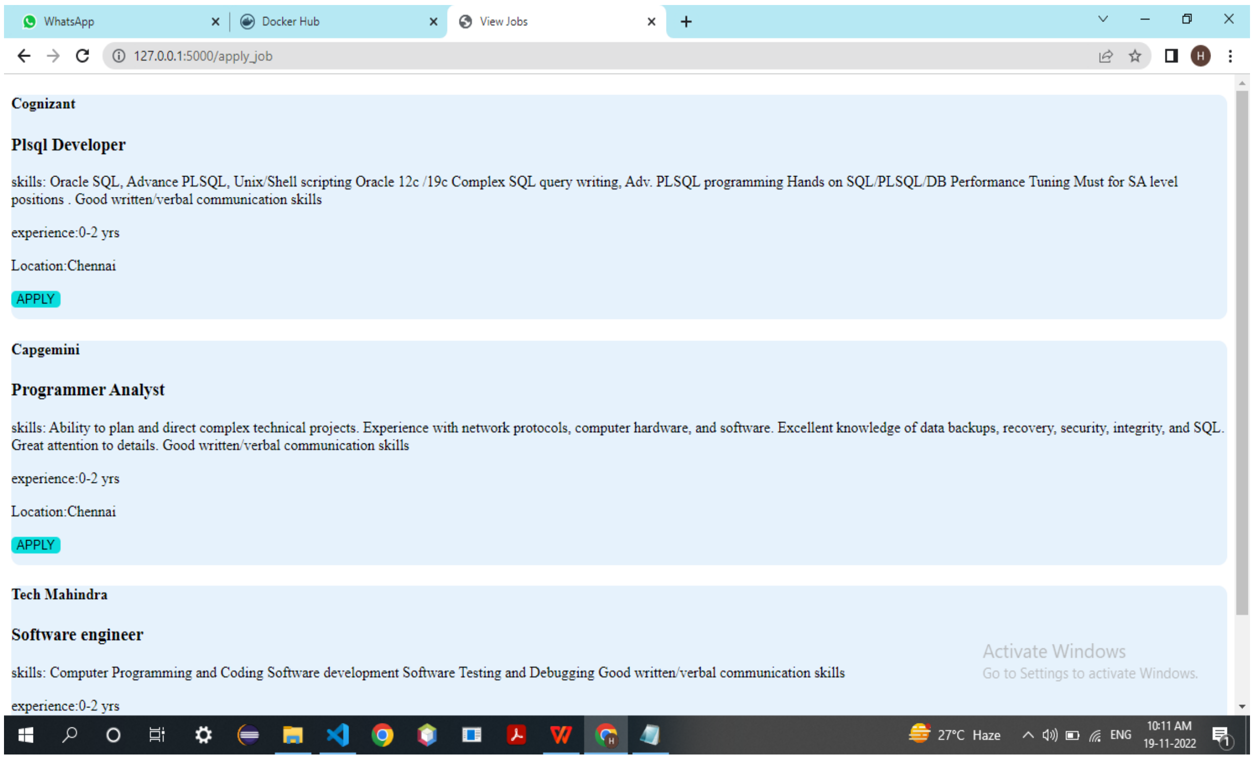
City

Email

Mobile Number


WhatsApp
Docker Hub
manual search
127.0.0.1:5000/recommend
APPLY JOBS
GET RECOMMENDATION THROUGH SEARCH
GET RECOMMENDATION THROUGH CHATBOT
Activate Windows
Go to Settings to activate Windows.
27°C Haze
ENG
10:10 AM
19-11-2022


WhatsApp
Docker Hub
Manual
127.0.0.1:5000/manualrecommendation
search for job role
search for location
Submit
Activate Windows
Go to Settings to activate Windows.
27°C Haze
ENG
10:10 AM
19-11-2022



Cognizant

Plsql Developer

skills: Oracle SQL, Advance PLSQL, Unix/Shell scripting Oracle 12c /19c Complex SQL query writing, Adv. PLSQL programming Hands on SQL/PLSQL/DB Performance Tuning Must for SA level positions . Good written/verbal communication skills

experience:0-2 yrs

Location:Chennai

APPLY

Capgemini

Programmer Analyst

skills: Ability to plan and direct complex technical projects. Experience with network protocols, computer hardware, and software. Excellent knowledge of data backups, recovery, security, integrity, and SQL. Great attention to details. Good written/verbal communication skills

experience:0-2 yrs

Location:Chennai

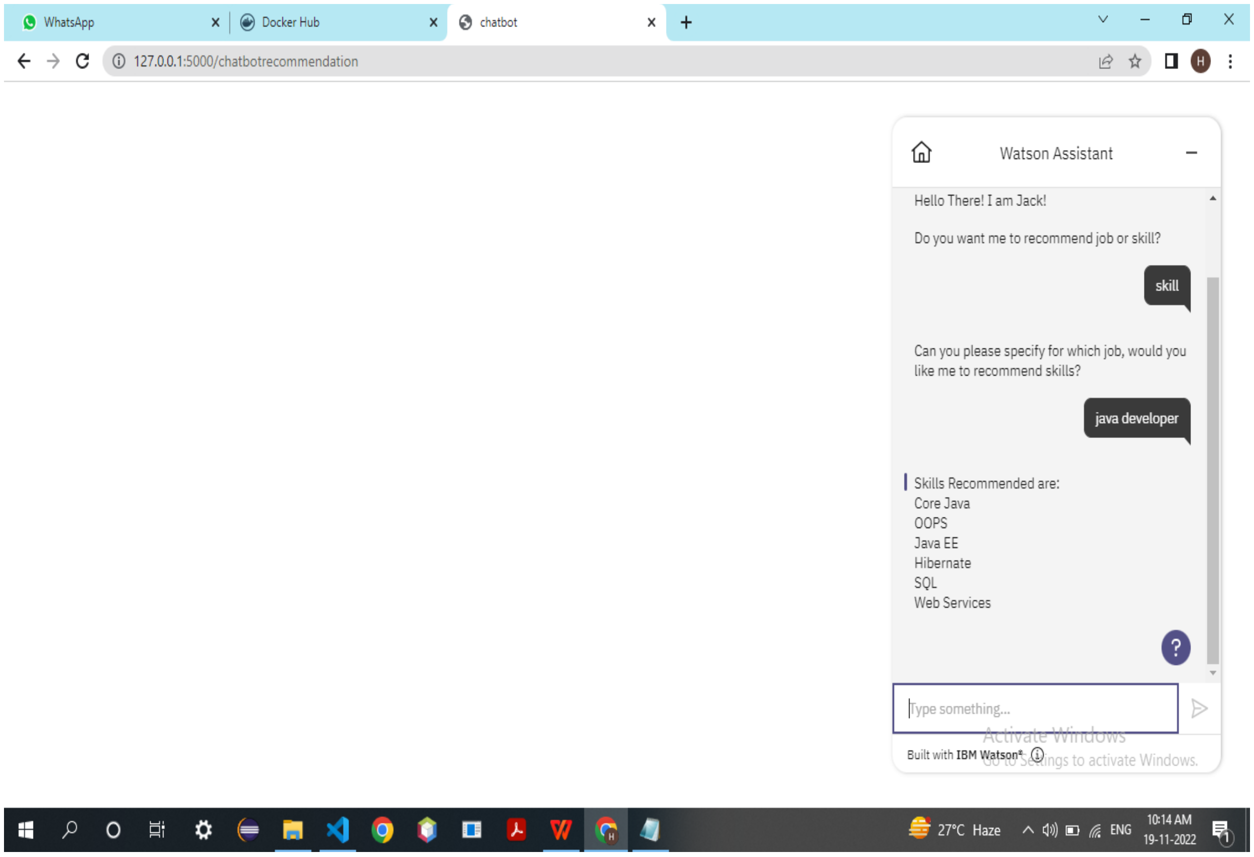
APPLY

Tech Mahindra

Software engineer

skills: Computer Programming and Coding Software development Software Testing and Debugging Good written/verbal communication skills

experience:0-2 yrs



Hello There! I am Jack!

Do you want me to recommend job or skill?

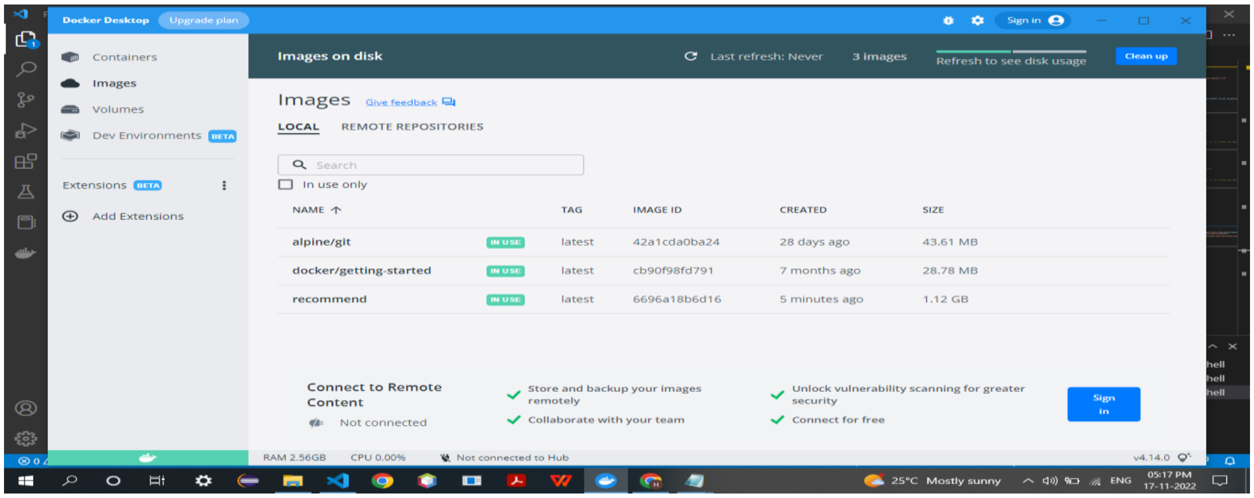
skill

Can you please specify for which job, would you like me to recommend skills?

java developer

Skills Recommended are:
Core Java
OOPS
Java EE
Hibernate
SQL
Web Services

DEPLOYMENT



Images on disk · Last refresh: Never · 3 images · Refresh to see disk usage

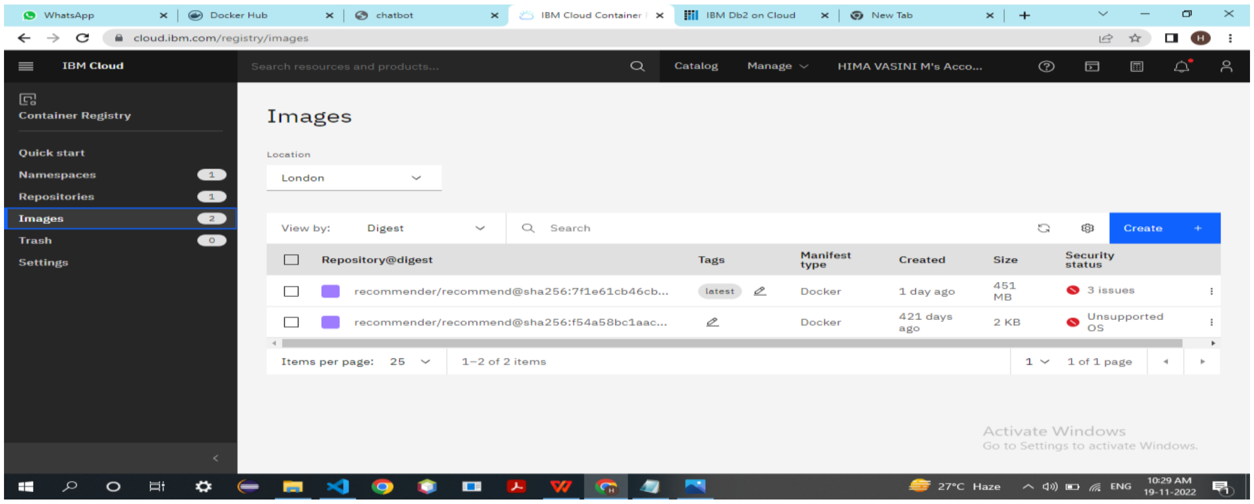
Images

NAME		TAG	IMAGE ID	CREATED	SIZE
alpine/git	IN USE	latest	42a1cda0ba24	28 days ago	43.61 MB
docker/getting-started	IN USE	latest	cb90f98fd791	7 months ago	28.78 MB
recommend	IN USE	latest	6696a18b6d16	5 minutes ago	1.12 GB

Connect to Remote Content · Not connected
✓ Store and backup your images remotely
✓ Collaborate with your team
✓ Unlock vulnerability scanning for greater security
✓ Connect for free
Sign in

RAM 2.56GB CPU 0.00% Not connected to Hub v4.14.0

25°C Mostly sunny ENG 05:17 PM 17-11-2022



cloud.ibm.com/registry/images

Images

Location: London

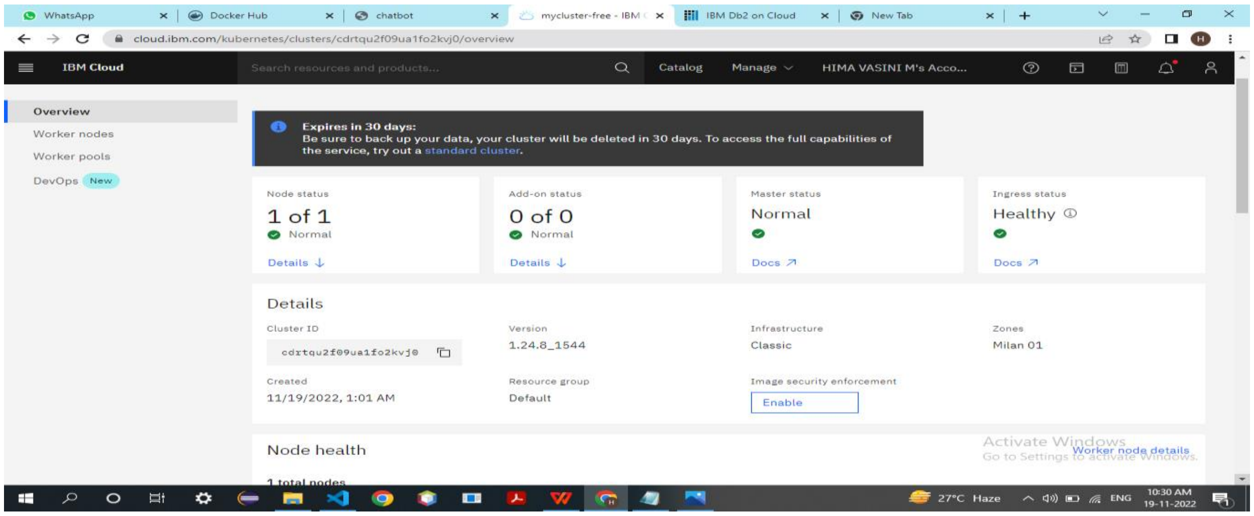
View by: Digest

Repository@digest	Tags	Manifest type	Created	Size	Security status
recommender/recommend@sha256:7f1e61cb46cb...	latest	Docker	1 day ago	451 MB	3 issues
recommender/recommend@sha256:f54a58bc1aac...		Docker	421 days ago	2 KB	Unsupported OS

Items per page: 25 · 1–2 of 2 items · 1 of 1 page

Activate Windows
Go to Settings to activate Windows.

27°C Haze ENG 10:29 AM 19-11-2022



cloud.ibm.com/kubernetes/clusters/cdrtqu2f09ua1fo2kvj0/overview

Expires in 30 days:
Be sure to back up your data, your cluster will be deleted in 30 days. To access the full capabilities of the service, try out a standard cluster.

Node status	Add-on status	Master status	Ingress status
1 of 1 ✓ Normal	0 of 0 ✓ Normal	Normal ✓	Healthy ✓
Details	Details	Docs	Docs

Details

Cluster ID: cdrtqu2f09ua1fo2kvj0	Version: 1.24.8_1544	Infrastructure: Classic	Zones: Milan 01
Created: 11/19/2022, 1:01 AM	Resource group: Default	Image security enforcement: Enable	

Node health · Worker node details

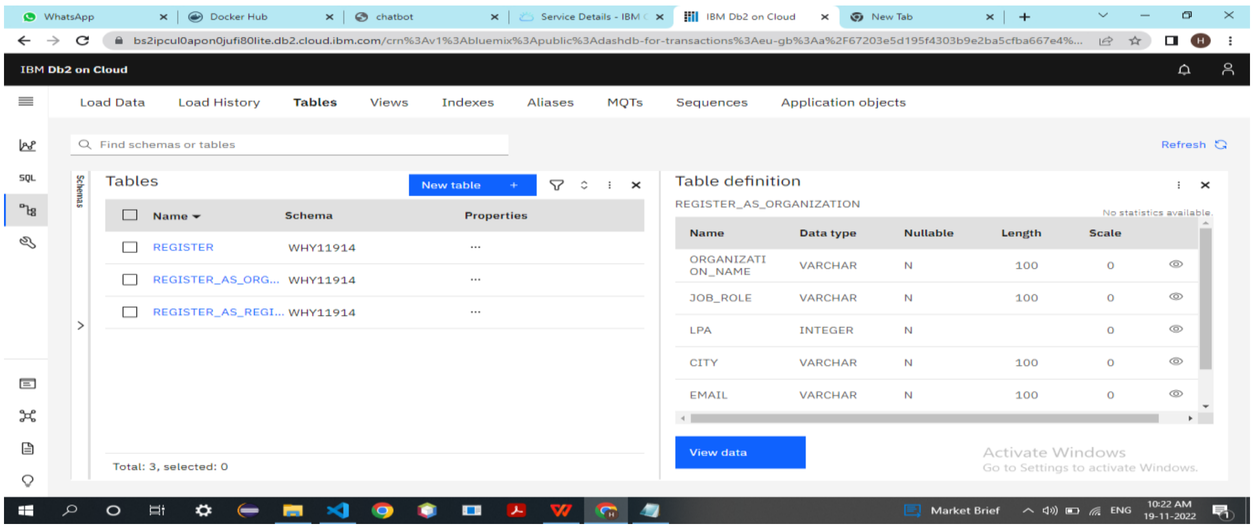
1 total nodes

Activate Windows
Go to Settings to activate Windows.

27°C Haze ENG 10:30 AM 19-11-2022

7.3 DATABASE SCHEMA

```
dsn_hostname = "0c77d6f2-5da9-48a9-81f8-
86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud"

dsn_uid = "why11914"

dsn_pwd = "qMCrggTKEsZ8b54q"

dsn_driver = "{IBM DB2 ODBC DRIVER}"

dsn_database = "bludb"

dsn_port = "31198"

dsn_protocol = "TCPIP"

dsn_security = "SSL"

dsn = (

    "DRIVER={0};"

    "DATABASE={1};"

    "HOSTNAME={2};"

    "PORT={3};"

    "PROTOCOL={4};"

    "UID={5};"

    "PWD={6};"

    "SECURITY={7};").format(dsn_driver, dsn_database, dsn_hostname, dsn_port,
dsn_protocol, dsn_uid, dsn_pwd,dsn_security)


conn = ibm_db.connect(dsn, "", "")


print(conn)
```


IBM Db2 on Cloud

Load Data Load History Tables Views Indexes Aliases MQTs Sequences Application objects

Tables

Name	Schema	Properties
REGISTER	WHY11914	...
REGISTER_AS_ORG...	WHY11914	...
REGISTER_AS_REGI...	WHY11914	...

Total: 3, selected: 0

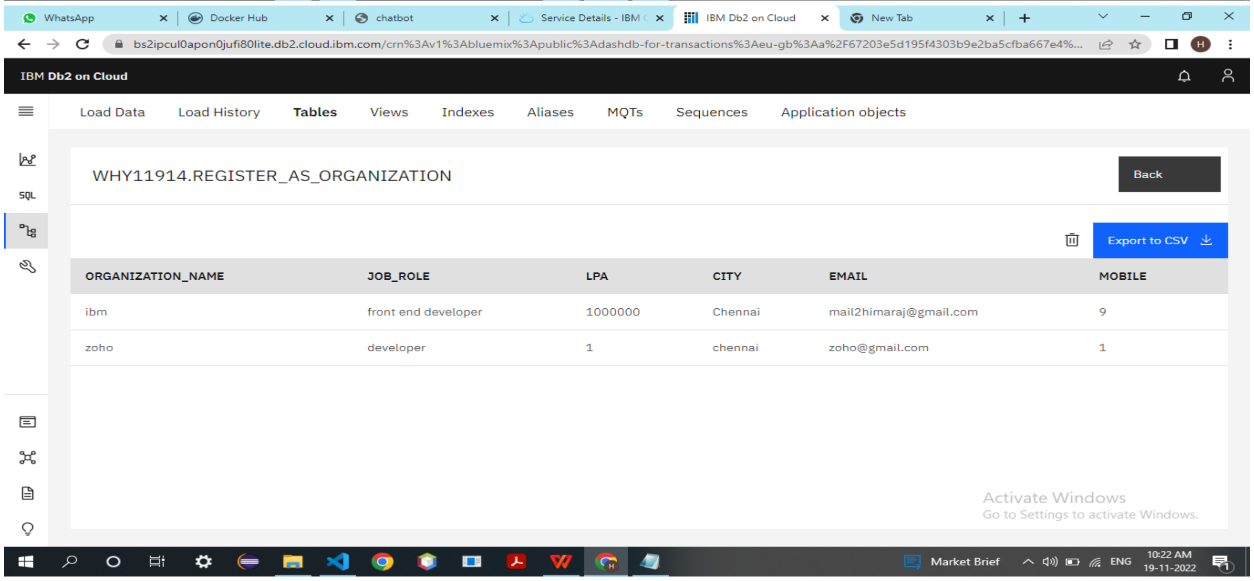
Table definition

REGISTER_AS_ORGANIZATION

No statistics available.

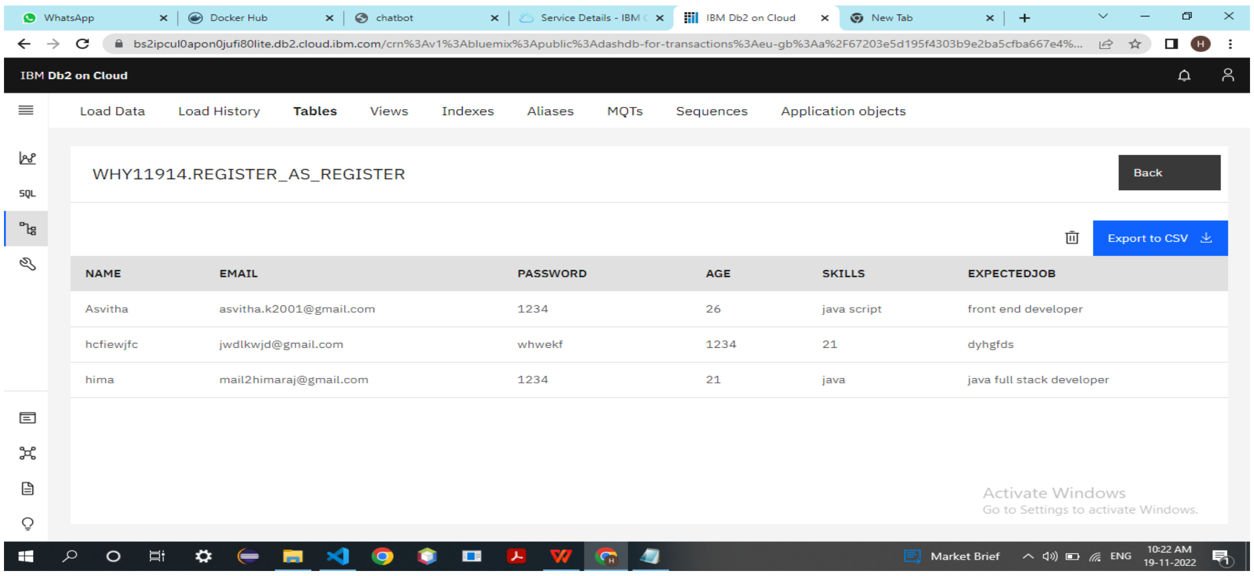
Name	Data type	Nullable	Length	Scale
ORGANIZATI ON_NAME	VARCHAR	N	100	0
JOB_ROLE	VARCHAR	N	100	0
LPA	INTEGER	N		0
CITY	VARCHAR	N	100	0
EMAIL	VARCHAR	N	100	0

View data


IBM Db2 on Cloud

WHY11914.REGISTER_AS_ORGANIZATION

ORGANIZATION_NAME	JOB_ROLE	LPA	CITY	EMAIL	MOBILE
ibm	front end developer	1000000	Chennai	mail2himaraj@gmail.com	9
zoho	developer	1	chennai	zoho@gmail.com	1


IBM Db2 on Cloud

WHY11914.REGISTER_AS_REGISTER

NAME	EMAIL	PASSWORD	AGE	SKILLS	EXPECTEDJOB
Asvitha	asvitha.k2001@gmail.com	1234	26	java script	front end developer
hcfiewjfc	jwdlkwjd@gmail.com	whwekf	1234	21	dyhgfds
hima	mail2himaraj@gmail.com	1234	21	java	java full stack developer

CHAPTER-8

TESTING

8.2 USER ACCEPTANCE TESTING

Test Cases ID	**Feature Type**	**Components**	**Test Scenario**
TC01	functional	home page	verifies, user is able to see the register and recommend button
TC02	UI	home page	verify the UI elements in nav bar such as login, about, redirect to home
TC03	functional	home page	verify user is able to log into application withvalid credentials
TC04	functional	application	verify user is able to get job recommendations
TC05	functional	contact	verify user is able to customer comment system
TC06	functional	chatbot	verify user is able to get right job recommendation

FLASK

Test Case ID	**Pre-Requisite**	**Steps To Execute**	**Test**
TC01	flask, vscode	1.enter URL 2.Click on register as job seeker 3.Click on register as organization	Index.html Registerasjobseeker.html Registerasorganization.html
TC02	flask , vscode	1.enter URL 2. click on login 3.verify login/sign with UI elements: a). email text box b). password **t**c). login button d). new customer? create account	login.html Registerasjobseeker.html
TC03	flask , vscode	1.enter URL 2.click on register 3.enter valid username/email in email text box 4.enter valid password in password text box	username: userid@gmail.com password: @userpassword
TC04	flask, vscode	1.verify working of apply job, recommend job by search, recommend job by chatbot	Apply_job.html Manual.html Chatbot.html

TC05	flask, vscode	Verify apply job pops up relevant jobs. Verify click on apply to start uploading resume.	Upload_resume.html
TC06	flask, vscode	Verify chatbot whether providing right recommendations. Verify relevant jobs for skills. Verify relevant skills for jobs.	Chatbot.html
TC07	Flask, vscode	Verify manual search by entering job role and location Verify whether job fetched from Indeed.com	Manual.html

Test Case ID	**Actual Result**	**Result**	**BUG ID**
TC01	working as expected	pass	no
TC02	working as expected	pass	no
TC03	working as expected	pass	no
TC04	working as expected	pass	no
TC05	working as expected	pass	no
TC06	working as expected	pass	no

CHAPTER-9

RESULTS

9.1 PERFORMANCE METRICS

At its most basic, a performance evaluation is simply providing constructive feedback on whether an employee is underperforming, meeting, or exceeding the goals and objectives of their job. Employees need this feedback so they can feel confident knowing what is expected of them as well as how and where they can improve.

1. CPU usage. CPU usage affects the responsiveness of an application.
2. Memory usage.
3. Requests per minute and bytes per request.
4. Latency and uptime.
5. Average response time
6. Error rates.

CHAPTER-10

ADVANTAGES &DISADVANTAGES

ADVANTAGES:

- Helps in Creating Right Job-Employee Fit
- Helps in Establishing Effective Hiring Practices
- Guides through Performance Evaluation and Appraisal Processes
- Helps in Analyzing Training & Development Needs
- Helps in Deciding Compensation Package for a Specific Job
- Help the users find relevant jobs

DISADVANTAGES:

- Too many choices
- Unpredictable items
- The cold-start problem
- Scalability
- The lack of right data
- Attributes may be incorrect or inconsistent.

CHAPTER-11

CONCLUSION

In this conclusion, we used a literature analysis of many journals and proceedings related to the requiring process and the job recommendation researches. We have seen from our literature review and from the challenges that faced the holistic e-requiring platforms, an increased need for enhancing the quality of candidates/job matching.

The recommender system technologies accomplished significant success in a broad range of applications and potentially a powerful searching and recommending techniques Consequently is a great opportunity for applying these technologies in environment to improve the matching quality.

This survey shows that several approaches for job recommendation have been proposed, an many techniques combined in order to produce the best fit between jobs and candidates. We presented state of the job recommendation as well as a comparative study for its approaches that proposed by literatures.

Additionally, we reviewed typical recommender system techniques and the recruiting process related issues. We conclude that the field of job recommendations is still unripe and require further improvements. As part of our ongoing research, we aim to build a new recommendation approach and test with real data for employee and staffing data from large companies.

CHAPTER-12

FUTURE SCOPE

Our proposed solution uses, LinkedIn API, well in future it will be extended to many jobs search API like Indeed API, Naukri API, etc.…

This application mainly focuses on IT background, in future it will be extended to non IT background as well.

future important skill:

- becoming more adaptable
- adjusting to career changes and new environments
- succeeding in a variety of settings
- improving teamwork skills
- offering greater benefits to companies that hire you

CHAPTER-13

APPENDIX

Source code

Index.html

```
<!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">

   <! --<link rel="stylesheet" href="style.css" type="text/css"/>-->

   <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">

   <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">

   <title>Home Page</title>

   <style>

      body{

        background-color: white; }

      nav{

   margin:5px;

   padding:5px;

   height:50px;

   width:100%;
```

```
    background-color: rgb(103, 117, 197);
}
.home{
    margin-right:15px;
    margin-top:20px;
    color:rgb(250, 249, 249);
    font-size: 20px;
}
.navlist{
    float:right;
    margin-right:20px;
}
.navlist li{
    display: inline-block;
    list-style-type: none;
    margin:5px;
    text-transform:uppercase;
    font-size:20px;
}
.navlist li a{
    color:white;
}
a{
    cursor:pointer;
}
```

```
#menu{
   display:none;
}
.checkbtn{
   display:none;
}

.reg{
      margin:0;
      position:absolute;
      top:50%;
      left:32%;
      -ms-transform:translateY(-50%);
      transform:translateY(-50%);
   }
@media(max-width: 800px){
   .checkbtn{
      float:right;
      display:block;
      font-size: 40px;
      margin-right:20px;
   }
   .home{
      display:block;
```

```
        padding:15px;


    }
    .navlist {
        display:block;
        color:black;
        position:fixed;
        width:30%;
        height:100vh;
        background: #4133bb;
        top:55px;
        right:0;
        text-align:center;
    }
    .navlist li{
        display:block;
        padding:15px;
    }


}
</style>
</head>
<body>
    <nav>
       <input type="checkbox" id="menu">
```

```
        <label for="menu" class="checkbtn">
            <i class="fa fa-bars"></i>
        </label>
        <a class="home" href="index.html">RECOMMEND ME SKILL/JOB</a>
        <ul class="navlist">
            <li><a href="/about">about</a></li>
            <li><a href="/contact">contact</a></li>
            <li><a href="/login">Login</a></li>
        </ul>
    </nav>

    <div class="reg">
        <a href="/recommend"><button type="button" class="btn btn-primary btn-lg btn-
block">RECOMMEND JOB THAT MATCHES WITH SKILLSET</button></a>
        <a href="/jobseeker"><button type="button" class="btn btn-secondary btn-lg btn-
block">REGISTER AS JOB SEEKER</button></a>
        <a href="/organization"><button type="button" class="btn btn-secondary btn-lg btn-
block">REGISTER AS ORGANIZATION</button></a>
    </div>
</body>
</html>
```

Registerasjobseeker.html

```
<!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>Regiter As Organization</title>

    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">

</head>

<body>

<section class="vh-100" style="background-color: rgb(255, 255, 255);">

  <div class="container h-100">

    <div class="row d-flex justify-content-center align-items-center h-100">

      <div class="col-lg-12 col-xl-11">

        <div class="card text-black" style="border-radius: 25px;">

          <div class="card-body p-md-5">

            <div class="row justify-content-center">

              <div class="col-md-10 col-lg-6 col-xl-5 order-2 order-lg-1">


                <p class="text-center h1 fw-bold mb-5 mx-1 mx-md-4 mt-4">Lets Start!</p>


                <form  method="POST" class="mx-1 mx-md-4">


                  <div class="d-flex flex-row align-items-center mb-4">

                    <i class="fas fa-user fa-lg me-3 fa-fw"></i>

                    <div class="form-outline flex-fill mb-0">

                      <input type="text" id="form3Example1c" class="form-control" name="name"/>

                      <label class="form-label" for="form3Example1c">Your Name</label>
                    </div>
                  </div>

                  <div class="d-flex flex-row align-items-center mb-4">
                    <i class="fas fa-lock fa-lg me-3 fa-fw"></i>
                    <div class="form-outline flex-fill mb-0">
                      <input type="email" id="form3Example4c" class="form-control" name="email"/>
                      <label class="form-label" for="form3Example4c">Email</label>
                    </div>
                  </div>

                  <div class="d-flex flex-row align-items-center mb-4">
                    <i class="fas fa-lock fa-lg me-3 fa-fw"></i>
                    <div class="form-outline flex-fill mb-0">
                      <input type="password" id="form3Example4c" class="form-control"
name="password"/>
                      <label class="form-label" for="form3Example4c">password</label>
                    </div>
                  </div>

                  <div class="d-flex flex-row align-items-center mb-4">
                    <i class="fas fa-lock fa-lg me-3 fa-fw"></i>
                    <div class="form-outline flex-fill mb-0">
                      <input type="number" id="form3Example4c" class="form-control" name="age"/>
                      <label class="form-label" for="form3Example4c">Your Age</label>
```

```
                  </div>
                </div>

                <!----- <div class="d-flex flex-row align-items-center mb-4">
                  <i class="fas fa-lock fa-lg me-3 fa-fw"></i>
                  <div class="form-outline flex-fill mb-0">
                    <input type="text" id="form3Example4c" class="form-control" name="address"/>
                    <label class="form-label" for="form3Example4c">Your Address</label>
                  </div>
                </div>-->

                <div class="d-flex flex-row align-items-center mb-4">
                  <i class="fas fa-lock fa-lg me-3 fa-fw"></i>
                  <div class="form-outline flex-fill mb-0">
                    <input type="text" id="form3Example4c" class="form-control" name="skills"/>
                    <label class="form-label" for="form3Example4c">Your Skills</label>
                  </div>
                </div>

                <div class="d-flex flex-row align-items-center mb-4">
                  <i class="fas fa-lock fa-lg me-3 fa-fw"></i>
                  <div class="form-outline flex-fill mb-0">
                    <input type="text" id="form3Example4c" class="form-control"
name="expectedjob" />
                    <label class="form-label" for="form3Example4c">Expected jobRole</label>
```

```
                      </div>
                    </div>


                    <div class="d-flex justify-content-center mx-4 mb-3 mb-lg-4">
                      <input type="submit" class="btn btn-primary btn-lg"></input>
                    </div>
                    <!---- <a href="/jobseeker"> <input type="submit" class="btn btn-primary btn-
lg">Register</submit></a>
                    </div>-->


                  </form>


                </div>
                <div class="col-md-10 col-lg-6 col-xl-7 d-flex align-items-center order-1 order-lg-2">



                  <img src="https://mdbcdn.b-cdn.net/img/Photos/new-templates/bootstrap-
registration/draw1.webp" height="500px"
width="500px" class="img-fluid" alt="Sample image">


                </div>
              </div>
            </div>
          </div>
        </div>
      </div>
```

```
  </div>
</section>
</body>
```

Login.html

```
<!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">
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
    <title>Login</title>
    <style>
      .space
      {
        position:absolute;
        top:200px;
        left:200px;
      }
.divider:after,
.divider:before {
content: "";
flex: 1;
height: 1px;

background: #eee;

}

.h-custom {

height: calc(100% - 73px);

}

@media (max-width: 450px) {

.h-custom {

height: 100%;

}

}

</style>

</head>

<body>


   <section class="vh-100">

     <div class="container-fluid h-custom">

       <div class="row d-flex justify-content-center align-items-center h-100">

         <div class="col-md-9 col-lg-6 col-xl-5">

           <img src="https://mdbcdn.b-cdn.net/img/Photos/new-templates/bootstrap-login-
form/draw2.webp"

             class="img-fluid" alt="Sample image">

         </div>

         <div class="col-md-8 col-lg-6 col-xl-4 offset-xl-1">

           <form action="/login" method="POST">
```

```
<!-- Email input -->
<div class="space">
<div class="form-outline mb-4">
  <input type="email" id="form3Example3" class="form-control form-control-lg"
    placeholder="Enter a valid email address" />
  <label class="form-label" for="form3Example3">Email address</label>
</div>

<!-- Password input -->
<div class="form-outline mb-3">
  <input type="password" id="form3Example4" class="form-control form-control-lg"
    placeholder="Enter password"  />
  <label class="form-label" for="form3Example4">Password</label>
</div>

<div class="d-flex justify-content-between align-items-center"></div>

<div class="text-center text-lg-start mt-4 pt-2">
  <button type="button" class="btn btn-primary btn-lg"
    style="padding-left: 2.5rem; padding-right: 2.5rem;">Login</button>
  <p class="small fw-bold mt-2 pt-1 mb-0">Don't have an account? <a
href="/jobseeker"
      class="link-danger">Register</a></p>
</div>
```

```
                </div>


              </form>
              <p class="message">{{msg}}</p>
            </div>
          </div>
        </div>
        <div
          class="d-flex flex-column flex-md-row text-center text-md-start justify-content-between
py-4 px-4 px-xl-5 bg-primary">
        </div>
      </section>


</body>
</html>
</html>
```

app.py

```
import csv
from datetime import datetime


#from flask_db2 import DB2
#import ibm_db
import ibm_db
import requests
from bs4 import BeautifulSoup
from flask import Flask, render_template, request, url_for

import os

#import sendgrid

#from sendgrid import SendGridAPIClient

#from sendgrid.helpers.mail import *




dsn_hostname = "0c77d6f2-5da9-48a9-81f8-
86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud"

dsn_uid = "why11914"

dsn_pwd = "qMCrggTKEsZ8b54q"

dsn_driver = "{IBM DB2 ODBC DRIVER}"

dsn_database = "bludb"

dsn_port = "31198"

dsn_protocol = "TCPIP"

dsn_security = "SSL"

dsn = (

    "DRIVER={0};"

    "DATABASE={1};"

    "HOSTNAME={2};"

    "PORT={3};"

    "PROTOCOL={4};"

    "UID={5};"

    "PWD={6};"

    "SECURITY={7};").format(dsn_driver, dsn_database, dsn_hostname, dsn_port, dsn_protocol,
dsn_uid, dsn_pwd,dsn_security)
```

```
conn = ibm_db.connect(dsn, "", "")

print(conn)
print("Connection Successful...........")
app=Flask(__name__)

@app.route('/')
def welcome():
    return render_template("index.html")
@app.route('/jobseeker',methods=['POST','GET'])
def jobseeker():
    if request.method == 'POST':
        name = request.form['name']
        email = request.form['email']
        password = request.form['password']
        age = request.form['age']
        skills = request.form['skills']
        expectedjob = request.form['expectedjob']

        #if password!=cp:
            #return render_template('register.html')

        try:
            insert_sql = "INSERT INTO REGISTER_AS_REGISTER VALUES (?,?,?,?,?,?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)

            ibm_db.bind_param(prep_stmt, 1, name)

            ibm_db.bind_param(prep_stmt, 2, email)

            ibm_db.bind_param(prep_stmt, 3, password)

            ibm_db.bind_param(prep_stmt, 4, age)

            ibm_db.bind_param(prep_stmt, 5, skills)

            ibm_db.bind_param(prep_stmt, 6, expectedjob)

            ibm_db.execute(prep_stmt)

            #sql = "insert into REGISTER_AS_REGISTER values ('{}','{}','{}','{}','{}','{}')".format(
name, email, password, age, skills, expectedjob)

            #stmt = ibm_db.exec_immediate(conn,sql)

            print("No of Affected rows: ",ibm_db.num_rows(prep_stmt))

        except:

            print("Error: ",ibm_db.stmt_errormsg())

    return render_template("registerasjobseeker.html")

@app.route('/organization',methods=['POST','GET'])

def organization():

    if request.method == 'POST':

        organization_name = request.form['organization_name']

        job_role = request.form['job_role']

        lpa = request.form['lpa']

        #address = request.form['address']

        city = request.form['city']

        email = request.form['email']

        mobile = request.form['mobile']
        #if password!=cp:
            #return render_template('register.html')


        try:
            insert_sql = "INSERT INTO REGISTER_AS_ORGANIZATION VALUES (?,?,?,?,?,?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prep_stmt, 1, organization_name)
            ibm_db.bind_param(prep_stmt, 2, job_role)
            ibm_db.bind_param(prep_stmt, 3, lpa)
            ibm_db.bind_param(prep_stmt, 4, city)
            ibm_db.bind_param(prep_stmt, 5, email)
            ibm_db.bind_param(prep_stmt, 6, mobile)
            ibm_db.execute(prep_stmt)
            #sql = "insert into REGISTER_AS_REGISTER values ('{}','{}','{}','{}','{}','{}')".format(
name, email, password, age, skills, expectedjob)
            #stmt = ibm_db.exec_immediate(conn,sql)
            print("No of Affected rows: ",ibm_db.num_rows(prep_stmt))
        except:
            print("Error: ",ibm_db.stmt_errormsg())
    return render_template("registerasorganization.html")
@app.route('/login',methods=['POST','GET'])
def login():
    if request.method == 'POST':
        email = request.form.get('email')
        password = request.form.get('password')


        sql = "SELECT * FROM REGISTER_AS_REGISTER email =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, email)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)


        if account:
            if (password == str(account['PASS']).strip()):
                return render_template('index.html')
            else:
                return render_template('index.html', msg="Password is invalid")
        else:
            return render_template('registerasjobseeker.html')
    else:
        return render_template('login.html')
@app.route('/about')
def about():
    return render_template('about.html')
@app.route('/recommend')
def recommend():
    return render_template('recommend.html')
@app.route('/apply_job')
def apply_job():
    return render_template('apply_job.html')
@app.route('/applyforjob',methods=['POST','GET'])
def applyforjob():
    if request.method == 'POST':
        resume = request.form.get('resume')
        email = request.form.get('email')
        message =
Mail(from_email=email,to_emails='mail2himaraj@gmail.com',subject='Resume',html_content=r
esume)

sendgrid.SendGridAPIClient(apikey='SG.zLNCF9gKR2at5dcxotfDZQ.80Hp6z9OXXLf4OsV2s
FUHjrebzUy6J-yydwoFyFAmsk')
        from_email=Email("mail2himaraj@gmail.com")
        to_email=Email("mail2himaraj@gmail.com")
        subject="mail"
        content=Content("text/plain","easy")
        mail=Mail(from_email,subject,to_email,content)
        response=sg.client.mail.send.post(request_body=mail.get())'''
    return render_template('upload_resume.html')
@app.route('/manualrecommendation',methods=['POST','GET'])
def manualrecommendation():
    if request.method == 'POST':
        jobrole= request.form.get('jobrole')
        location= request.form.get('location')
        template='https://in.indeed.com/jobs?q={}&l={}&from=searchOnHP'
        url=template.format(jobrole, location)
        return url
    return render_template('manual.html')
#url=manualrecommendation('jobrole','location')
#print(url)
@app.route('/chatbotrecommendation')
def chatbotrecommendation():
    return render_template("chatbot.html")
@app.route('/chatbotrecon')
def chatbotcon():
    return render_template("chatconversation.html")
if __name__=="__main__":
    port=int(os.environ.get('PORT',5000))
    app.run(port=port,host="0.0.0.0")
```

Applyjob.html

```
<!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>View Jobs</title>
    <style>
        .container{
            display:inline;
        }
        .box{
            height:230px;
            width:100%;
            border:1px solid none;
            background-color: rgb(230, 242, 252);
            border-radius: 10px;
        }
        .btn{
            background-color: rgb(10, 221, 221);
            border:rgb(152, 243, 243);
            border-radius: 5px;
            cursor:pointer;
        }
        </style>
</head>
<body>
    <div class="container">
        <div class="box">
            <h4>Cognizant</h4>
            <h3>Plsql Developer</h3>
            <p>skills:
               Oracle SQL, Advance PLSQL, Unix/Shell scripting
               Oracle 12c /19c
               Complex SQL query writing, Adv. PLSQL programming
        Hands on SQL/PLSQL/DB Performance Tuning Must for SA level positions .
        Good written/verbal communication skills</p>
    <p>experience:0-2 yrs</p>
    <p>Location:Chennai</p>
    <a href="/applyforjob"><button type="submit" class="btn" >APPLY</button></a>
</div>
<div class="box">
    <h4>Capgemini</h4>
    <h3>Programmer Analyst </h3>
    <p>skills:
        Ability to plan and direct complex technical projects.
        Experience with network protocols, computer hardware, and software.
        Excellent knowledge of data backups, recovery, security, integrity, and SQL.
        Great attention to details.
        Good written/verbal communication skills</p>
    <p>experience:0-2 yrs</p>
    <p>Location:Chennai</p>
    <a href="/upload_resume"><button type="submit" class="btn" >APPLY</button></a>
</div>
<div class="box">
    <h4>Tech Mahindra</h4>
    <h3>Software engineer</h3>
    <p>skills:
        Computer Programming and Coding
        Software development
```

```
            Software Testing and Debugging

            Good written/verbal communication skills</p>

           <p>experience:0-2 yrs</p>

           <p>Location:Chennai</p>

           <a href="/applyforjob"><button type="submit" class="btn" >APPLY</button></a>

        </div>


        </div>

</body>

</html>
```

Chatbot.html

```
<!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">

    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">

    <title>chatbot</title>

    <style>

        .center{

            margin:0;

            position:absolute;

            top:50%;
            left:45%;

            -ms-transform:translateY(-50%);

            transform:translateY(-50%);

        }

        </style>

</head>

<body><!---<div class="center">

    <button type="button" class="btn btn-secondary btn-lg btn-block">CHATBOT</button>

    </div>--->

        <script>

            window.watsonAssistantChatOptions = {

              integrationID: "889e133a-41ab-4db9-b03d-e94022406dc1", // The ID of this
integration.

              region: "au-syd", // The region your integration is hosted in.

              serviceInstanceID: "c72dc204-4442-4727-bf4b-86c09de2b0f6", // The ID of your
service instance.

              onLoad: function(instance) { instance.render(); }

            };

            setTimeout(function(){

              const t=document.createElement('script');

              t.src="https://web-chat.global.assistant.watson.appdomain.cloud/versions/" +
(window.watsonAssistantChatOptions.clientVersion || 'latest') + "/WatsonAssistantChatEntry.js";

              document.head.appendChild(t);

            });

          </script>

</body>
```

</html>

Docker file

```
FROM python

WORKDIR /app

COPY . .

RUN pip install -r requirements.txt

CMD ["flask","run"]

EXPOSE 5000
```

Requirments.txt

```
flask

ibm_db

sendgrid

bs4

Requests
```

GitHub

https://github.com/IBM-EPBL/IBM-Project-8324-1658915549

Project Demo Link:

https://drive.google.com/drive/folders/1s6nkCBtQujuPFrXjg63J9Om0n4111exD