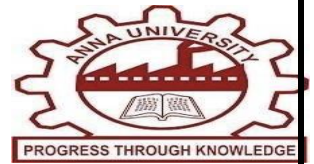


IT - ITes SSC
NASSCOM



SKILL/JOB RECOMMENDATION APPLICATION



PROJECT REPORT

UNDER THE GUIDANCE OF

INDUSTRY MENTOR(S) NAME : Krishna Chaitanya

FACULTY MENTOR(S) NAME : S Hemamalini

TEAM ID: PNT2022TMID25936

TEAM MEMBERS

VENKATESH P	211519104180
YOGESH BALAN	211519104187
ADI KRISHNA K	211519104301
AKASH ARUMUGAM	

APPLICATION DOMAIN: Cloud App Development

College name: PANIMALAR INSTITUTE OF
TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

TABLE OF CONTENTS

1. INTRODUCTION

- 1.1. Project Overview
- 1.2. Purpose

2. LITERATURE SURVEY

- 2.1. Existing problem
- 2.2. References

- 2.3. Problem Statement Definition

3. IDEATION & PROPOSED SOLUTION

- 3.1. Empathy Map Canvas
- 3.2. Ideation & Brainstorming
- 3.3. Proposed Solution
- 3.4. Problem Solution fit

4. REQUIREMENT ANALYSIS

- 4.1. Functional requirement
- 4.2. Non-Functional requirements

5. PROJECT DESIGN

- 5.1. Data Flow Diagrams
- 5.2. Solution & Technical Architecture
- 5.3. User Stories

6. PROJECT PLANNING & SCHEDULING

- 6.1. Sprint Planning & Estimation
- 6.2. Sprint Delivery Schedule
- 6.3. Reports from JIRA

7. CODING & SOLUTIONING

- 7.1. Feature 1
- 7.2. Feature 2

8. TESTING

- 8.1. Test Cases
- 8.2. User Acceptance Testing

9. RESULTS

- 9.1. Performance Metrics

10. ADVANTAGES & DISADVANTAGES

11. CONCLUSION

12. FUTURE SCOPE

13. APPENDIX

Project Report Format

1. INTRODUCTION

1.1. Project Overview

Job recommender systems have become popular since they successfully reduce information overload by generating personalized job suggestions. Although in the literature exists a variety of techniques and strategies used as part of job recommender systems, most of them fail to recommend job vacancies that fit properly to the jobseekers profiles. Thus, the contributions of this work are threefold, we:

- i) made publicly available a new dataset formed by a set of job seekers profiles and a set of job vacancies collected from different job search engine sites;
- ii) put forward the proposal of a framework for job recommendation based on professional skills of job seekers;
- iii) carried out an evaluation to quantify empirically the recommendation abilities of two state-of-the-art methods, considering different configurations within the proposed framework. We thus present a general panorama of job recommendation task aiming to facilitate research and real-world application design regarding this important issue

Keywords: Job matching, job seeking, job search, job recommender systems, person-job fit, LinkedIn, word embedding
Keywords: Job matching, job seeking, job search, job recommender systems, person-job fit, LinkedIn, word embedding

1.2. Purpose

The recommender systems could use historical rating information to determine which type of job required which type of candidate characteristics in the past in order to be rated positively by the recruiter. This information could then be used to predict the match between job and previously not rated candidates. The need of applying the recommender system techniques for selection process can be motivated from different perspectives. While we interested in how people find an appropriate job

2. LITERATURE SURVEY

2.1. Existing problem

The fast growth of the Internet caused a matching growth of the amount of available online information that increased the need to expand the ability of users to manage all this information. This encourages a substantial interest in specific research fields and technologies that could benefit the management of this information overload. The most important fields are Information retrieval and Information filtering. Information retrieval deals with automatically matching users information and Information filtering aims to assist users eliminating unwanted information

2.2. References

Shaha T Al-Otaibi and Mourad Ykhlef. "A survey of job recommender systems". In: International Journal of the Physical Sciences 7.29 (2012), pp. 5127–5142. issn: 19921950. doi:10.5897/IJPS12.482.

N Deniz, A Noyan, and O G Ertosun. "Linking Person-job Fit to Job Stress: The Mediating Effect of Perceived Person-organization Fit". In: Procedia - Social and Behavioral Sciences 20.7.(2015), pp. 369–376

M Diaby and E Viennet. "Taxonomy-based job recommender systems on Facebook and LinkedIn profiles". In: Proc. of Int. Conf. on Research Challenges in Information Science (2014), pp. 1–6. issn: 21511357. doi:10.1109/RCIS.2014.6861048.

MKusneretal. "Fromwordembeddingstodocumentdistances". In: Proc. of the 32nd Int. Conf. on Machine Learning, ICML'15. 2015, pp. 957–966

MDiaby, EViennet, and TLaunay. "Towardthenextgenerationofrecruitment tools: An online social network-based job recommender system". In: Proc. of the 2013 IEEE/ACM Int. Conf. on Advances in Social Networks Analysis and Mining, ASONAM 2013 (2013) pp. 821–828. doi:10.1145/2492517.2500266

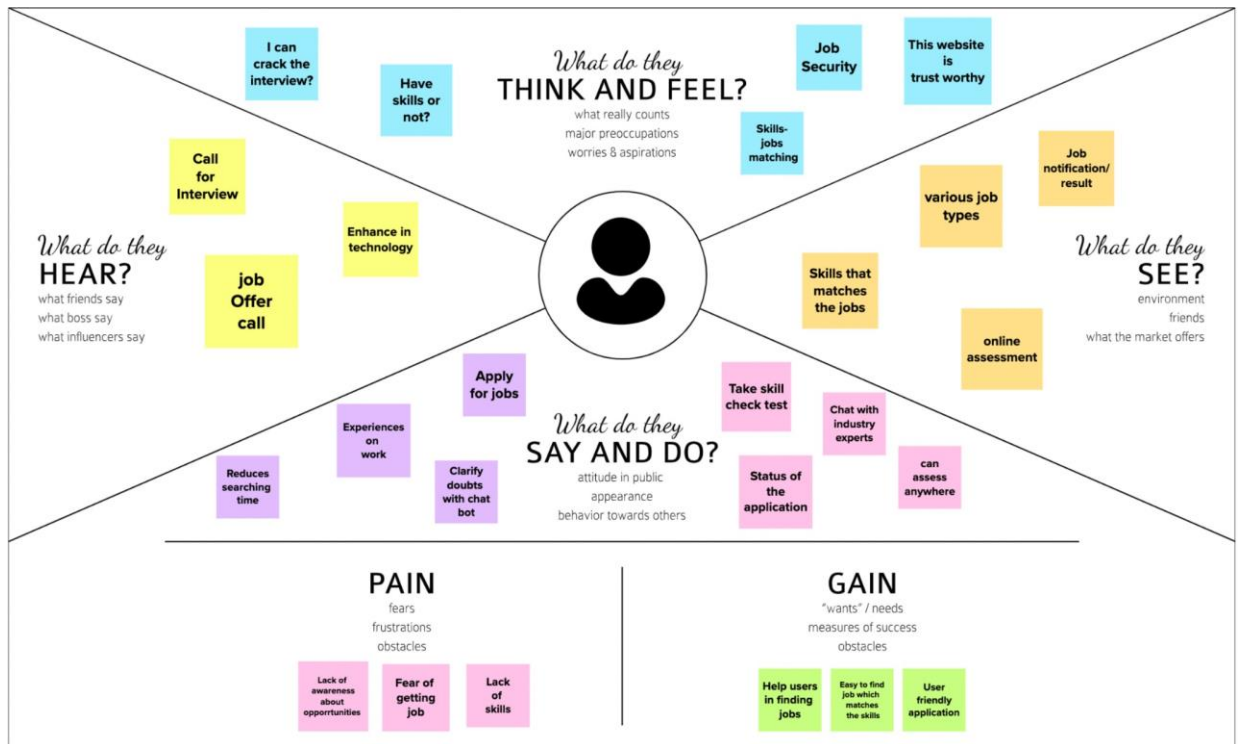
2.3. Problem Statement Definition

We have a new innovation to directly choose your job related to your skills without someone's help. You can chat with chatbot for getting of list of jobs related to your skillset. We developed a end to end web application showing the current job opening based on your skillset. The user's information will be stored in the database. The alert is sent to the

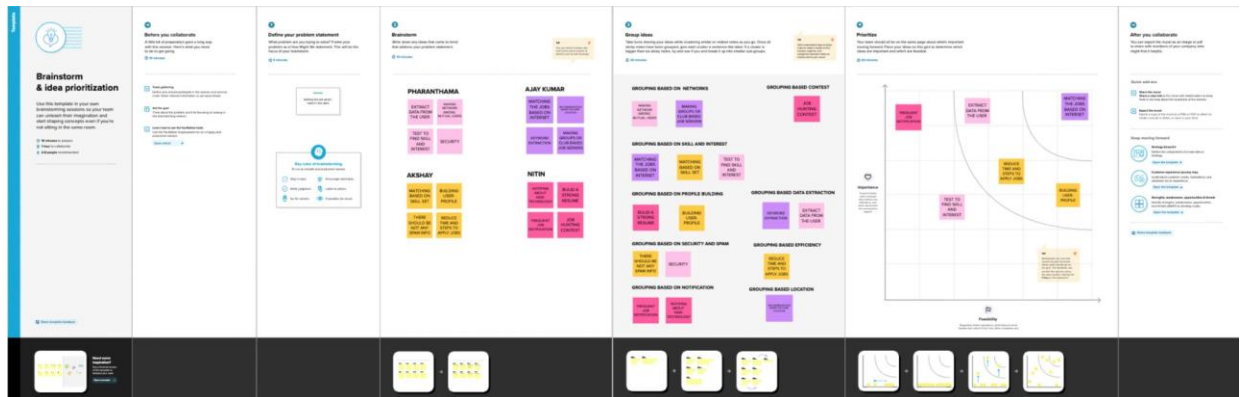
user when there is opening for the user based on their skillset. User can interact with the chatbot to get the recommendations. This application updates the user regarding latest jobs. Once user enter skills into chatbot it search related job in database than it shows various jobs related to the users skillset.

3. IDEATION & PROPOSED SOLUTION

3.1. Empathy Map Canvas



3.2. Ideation & Brainstorming



3.3. Proposed Solution


S. No.	Parameter	Description
1.	Problem Statement	Now a days many graduates situation is unemployed. To solve this Problem we have to recommend job opportunities to the graduates through online that can be accessed anywhere. We assure this application will help them to search the job easily and quickly and also to gain new skills.
2.	Idea/Solution Description	We have an idea to recommend you many Jobs openings from many companies based on the role you want and also it shows you the number of vacancies available in that companies. This application can be used by all domain graduates to find there perfect
3.	Novelty/Uniqueness	This skill and job recommender application is very useful to the user to find their role and job based on their unique skills.
4.	Social Impact/Customer Satisfaction	By using this application user gain more details about which and how to crack that interview by using our skill recommender.
5.	Business model/Revenue model	This model is cost effective. which makes the user to spend less.
6.	Scalability of the solution	This application is a lifelong recommender application. We can access this for lifelong. Once we installed this application it will update you day by day. So it will be very useful to the user to be updated

3.4. Problem Solution fit

Problem-Solution Fit canvas Purpose / Vision: Skill/Job recommender Version: 2.0

Define CS, fit into CL	1. CUSTOMER SEGMENT(S) CS The main customers of our project are: 1. Candidates who are seeking for jobs 2. Recruiters who recruits candidates	6. CUSTOMER LIMITATIONS EG. BUDGET, DEVICES CL 1. By avoiding misuse of personal information from third person 2. Avoiding potential scam 3. Time consuming 4. To concentrate on unreliable connections	5. AVAILABLE SOLUTIONS PROS & CONS AS <table border="1"> <tr> <th>PROS</th> <th>CONS</th> </tr> <tr> <td>Promotion skillset</td> <td>False information</td> </tr> <tr> <td>company infrastructure</td> <td>fraudulent activity</td> </tr> <tr> <td>Commercial relationship</td> <td>intense competition</td> </tr> </table>	PROS	CONS	Promotion skillset	False information	company infrastructure	fraudulent activity	Commercial relationship	intense competition	Explore AS, differentiate
	PROS	CONS										
Promotion skillset	False information											
company infrastructure	fraudulent activity											
Commercial relationship	intense competition											
2. PROBLEMS / PAINS + ITS FREQUENCY PR 1. To create a platform to facilitate job searching 2. To make the job-filtering process simpler 3. To keep profile personal data 4. Identify people with necessary skills	9. PROBLEM ROOT / CAUSE RC 1. Jobs that are listed on unreliable platforms maybe fraudulent 2. Some companies are failed to disclose their true infrastructure 3. Some users post false credentials 4. Users pretended to have expertise in a skillset they lack 5. Some jobs want payment first, so some are Money minded	7. BEHAVIOR + ITS INTENSITY BE 1. When users apply for fraudulent jobs, they get unhappy due to wasted time 2. Users were not satisfied when platforms allowed hirers to post jobs that were not real 3. Cheating during online recruitment process	Focus on PR, tap into BE, understand RC									
3. TRIGGERS TO ACT TR 1. Employment opportunities 2. Branding 3. Endorsement and connections 4. Get job alerts	10. YOUR SOLUTION SL To develop an end to end web application Which in default have a lot of current job Openings through job search API out of Which appropriate job will be recommended Based on user skill set. At the same time Students can develop their skills side by side With various courses and webinars offered by Reputed organizations. In addition to this a smart chat Bot will be available for 24*7 which can help users in finding the Right job.	8. CHANNELS of BEHAVIOR CH ONLINE 1. Apply for jobs 2. Review job applications 3. Attend initial level assessments OFFLINE 1. Final level interview 2. Checkout location and infrastructure of company 3. Finalize paperwork		Extract online & offline CH of BE								
4. EMOTIONS BEFORE / AFTER EM <table border="1"> <tr> <th>Emotions before</th> <th>Emotions after</th> </tr> <tr> <td>Lack of knowledge about job vacancy</td> <td>User receive updates on job vacancies</td> </tr> <tr> <td>No proper platform to showcase skillset</td> <td>Exhibit skillset in profile</td> </tr> <tr> <td>More paperwork during recruitment</td> <td>Easy recruitment process</td> </tr> </table>	Emotions before	Emotions after	Lack of knowledge about job vacancy		User receive updates on job vacancies	No proper platform to showcase skillset	Exhibit skillset in profile	More paperwork during recruitment	Easy recruitment process			
Emotions before	Emotions after											
Lack of knowledge about job vacancy	User receive updates on job vacancies											
No proper platform to showcase skillset	Exhibit skillset in profile											
More paperwork during recruitment	Easy recruitment process											

Problem Solution fit canvas is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License. Designed by Daria Neprikulina / ideahackers.net - we tailor ideas to customer behaviour and increase solution adoption probability.

 IdeaHackers .NET

4. REQUIREMENT ANALYSIS

4.1. Functional requirement

Functional Requirements:

Following are the functional requirements of the proposed solution.

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	Web site	"SpinZ.co" is our through user can access our platform
FR-4	Mobile app	Functional requirements for a mobile app include the specific features that help users navigate through the application, access links and view the application from their mobile device
FR-5	User management system	User management system allow companies to interact with users, store user information and share information within a company's network
FR-6	Reporting guideline	It may explain how users can gather and search for specific data or information

4.2. Non-Functional requirements

Non-functional Requirements:

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

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	How easy is it for a user to use the system
NFR-2	Security	To protect sensitive data, you may consider developing non -functional security features
NFR-3	Reliability	The system run without a failure for a given period of time under predefined conditions
NFR-4	Performance	How fast a system can responds to certain user's actions under a certain workload
NFR-5	Availability	It describes how likely the system is accessible to a user at a given point in time
NFR-6	Scalability	It assesses the highest workloads under which the system will still meet the performance requirements

5. PROJECT DESIGN

5.1. Data Flow Diagrams

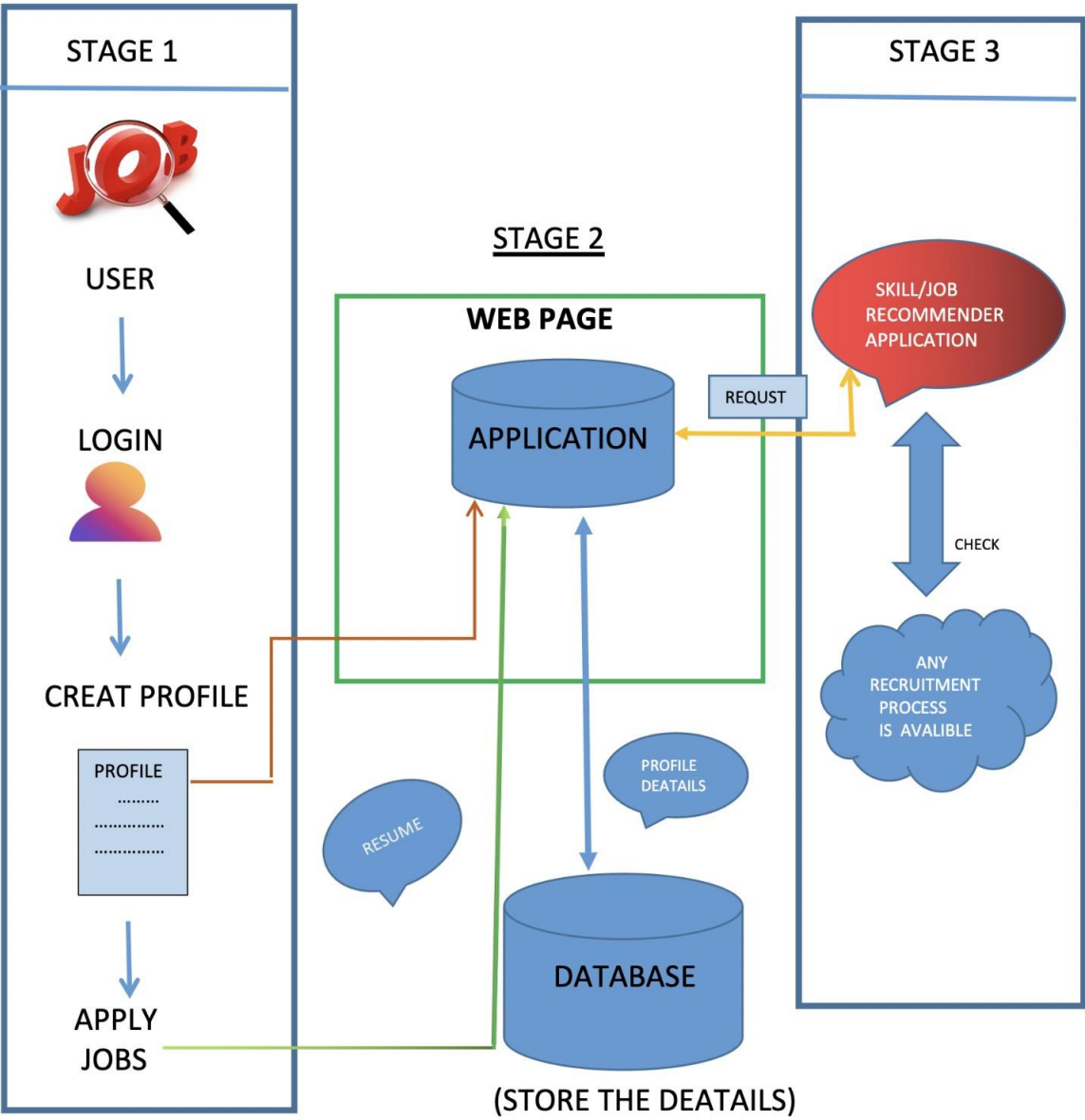
Project Design Phase-II Data Flow Diagram & User Stories

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



5.2. Solution & Technical Architecture



Technology Architecture:

Project shall full fill the following information in this technology architecture .

S.No.	Parameter	Description
1.	Is the System Robust?	Yes, it is partially buildable platform as the budget required will be more as cloud is a pay per use model and time taken will be quite.
2.	Is it highly modifiable?	Indeed, the framework is modifiable and it can own up to the progressions by recognizing blunders that requirements to be fixed and new functionalities. It is exceptionally receptive to the progressions.
3.	Is it Scalable?	Indeed, the framework proposed is exceptionally versatile as it can deal with the developing responsibility where great execution is likewise expected to effectivelywork. Organization of the stage has been finished utilizing different OS virtualization stage it will deal with the responsibility genuinely.

5.3. User Stories

Use the below template to list all the user stories for the product.

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-5	As a user, I can access my dashboard after signing in.	I can access my account / dashboard	High	Sprint-1
Customer (Web user)	Access	USN-6	As a user, I can setup a profile, and basic details by signing in.			
		USN-7	As a user, I will upload my resume, certificates, and other requirements.	I can perform several task in the application	Medium	Sprint-1

Customer Care Executive	Chatbot	USN-8	As a user, I can seek guidance from the customer care executive.		High	Sprint-1
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.	High	Sprint-1

6. PROJECT PLANNING & SCHEDULING

6.1. Sprint Planning & Estimation

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	Priority	Acceptance criteria	Team Members
Sprint-1	UI Design	USN-1	As a user, I can see and experience an awesome user interface in the website	Medium	Better Impression about awebsite	4
Sprint-1	Registration	USN-2	As a user, I can register for the applicationby entering my email, password.	High	I can access my account / dashboard	4
Sprint-1		USN-3	As a user, I will receive confirmation emailonce I have registered for the application	High	I can receive confirmationemail & click confirm	4
Sprint-1	Login	USN-6	As a user, I can log into the application byentering email & password	High	I can access my account / dashboard	4
Sprint-1	Flask	USN-7	As a user, I can access the website in a second.	High	I can access my account / dashboard	4
Sprint-1	Dashboard	USN-8	As a user, If I Logged in correctly, I can view my dashboard and I can navigate. to any pages which are already listed there.	High	I can access all the pages/dashboard	4

Submission Of Sprint-1

Sprint-2	User Profile	USN-9	As a user, I can view and update mydetails	Medium	I can modify my details/data	4
Sprint-2	Database	USN-10	As a user, I can store my details anddata in the website w	Medium	I can store my data	4
Sprint-2	Cloud Storage	USN-11	As a user, I can upload my resume in the website.	Medium	I can Upload my documentsanddetails	4
Sprint-2	Chatbot	USN-12	As a user, I can ask the Chatbot about latest job openings, which will help me and show the recent job openings based on my profile	High	I can know the recentjobopenings	4
Sprint-2	Identity-Aware	USN-13	As a User, I can access my account by entering by correct login credentials. Myuser credentials is only displayed to me.	High	I can have my accountsafely	4

Submission Of Sprint-2

Sprint	Functional Requirement (Epic)	User Story Number	User Story/ Task	Priority	Acceptance criteria	Team Members
Sprint-3	Sendgrid service	USN-14	As a user, I can get a notification or mail about a job opening with the help of sendgrid service.	Medium	I can get a notification in a second.	4
Sprint-3	Docker	USN-16	As a user, I can access the website in any device	High	I can access my account in any device	4
Sprint-3	Kubernetes	USN-17	As a user, I can access the website in any device	High	I can access my account in any device	4
Sprint-3	Deployment in cloud	USN-18	As a user, I can access the website in any device	High	I can access my account in any device	4

Sprint-3	Technical support	USN-19	As a user, I can get a customer care support from the website which will solve my queries.	Medium	I can tackle my problem & queries.	4
----------	-------------------	--------	--	--------	------------------------------------	---

Submission Of Sprint-3

Sprint	Functional Requirement (Epic)	User Story Number	User Story/ Task	Priority	Acceptance criteria	Team Members
Sprint-4	Unit Testing	USN-15	As a user, I can access the website without any interruption	High	I can access the website without any interruption	4
Sprint-4	Integration testing	USN-16	As a user, I can access the website without any interruption	High	I can access the website without any interruption	4
Sprint-4	System testing	USN-17	As a user, I can access the website without any interruption	High	I can access the website without any interruption	4
Sprint-4	Correction	USN-18	As a user, I can access the website without any interruption	High	I can access the website without any interruption	4

Sprint-4	Acceptance testing	USN-19	As a user, I can access the website without any interruption	High	I can access the website without any interruption	
----------	--------------------	--------	--	------	---	--

Submission Of Sprint-4

6.2. Sprint Delivery Schedule

Sprint Delivery planning:

Project Tracker & Velocity : (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	06 Nov 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	10 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	15 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

Velocity:

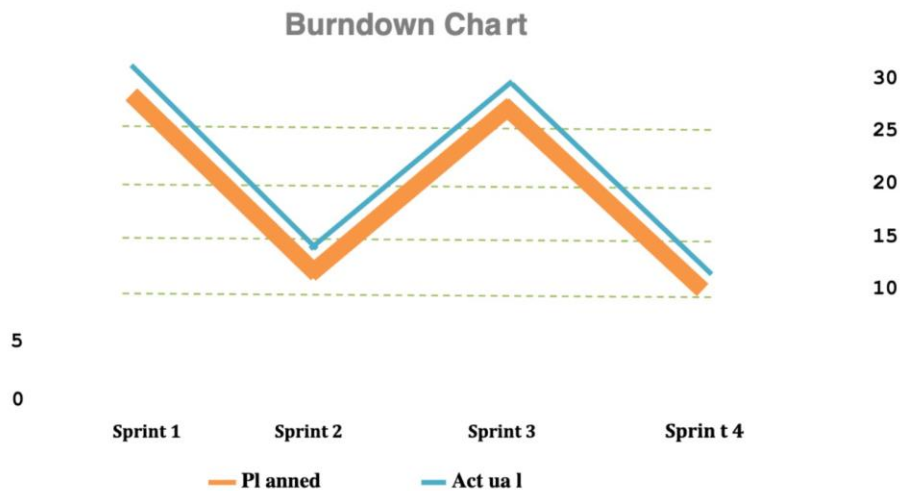
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \text{sprint duration} / \text{Velocity} = 20 / 2 \text{ velocity} = 10$$

6.3. Reports from JIRA

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.



7. CODING & SOLUTIONING

7.1 Feature 1

The software has an In-built “Chat Bot” which can help assist with ongoing queries and provide fast and effective solutions to user problems which may occur and also redirect to management attention if need be there any complications the customer service will be available 24*7 to assist in case of any controversial issues arise

6.2 Feature 2

In this project we have created the dashboard page to view the jobs available and to make ease to access the website

- They communicate information quickly.
- They display information clearly and efficiently.
- They show trends and changes in data over time.
- They are easily customizable.
- The most important widgets and data components are effectively presented in a limited space.

8. TESTING

8.1. Test Cases

Software testing is the process of evaluating and verifying that a software product or application does what it is supposed to do. The benefits of testing include preventing bugs, reducing development costs and improving performance.

This Software is tested and evaluated successfully.

8.2. User Acceptance Testing

Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the Inventory Management System project at the time of the release

to User Acceptance Testing (UAT)

User Acceptance Testing is carried out in a separate testing environment. A change, an update, or a new feature is requested and developed. Unit and integration tests are run. All seems to be in order. But then, after it is released to the public, serious problems appear. Rework and retesting are not the most expensive consequences when that happens. Loss of reputation is.

9. RESULTS

9.1. Performance Metrics

Based on the two types of user recommendations mentioned above, we analyze the performance of all the techniques mentioned above. The resultant jobs recommended to each new user are then checked with the job that the user is originally in as per the test dataset. If the original user job is recommended in the model result, then the model appends 1 for yes else, it appends 0 for no.

This array of 0's and 1's thus received is then checked for accuracy by computing the count of 1's from the total user predictions

Among all the models made with the incorporation of different similarity metrics, the cosine similarity based job recommendation system model outperformed rest of them all. The metrics used to analyse the model performance are: accuracy, precision, recall and

F1-score. This is because cosine considers the existence of duplicate terms while computing similarity. Also, computationally, cosine has low complexity and ease over handling sparse data vectors since only non-zero dimensions are considered.

Upon analyzing the result table we observe that the short-comings of some

similarity measures upon recommending top 5 and highest-score based job recommendations as even upon achieving high . similarity scores is due to the fact that users are seen to have different jobs than the ones recommended by the models, thus resulting in 6–10% error rates.

10. ADVANTAGES & DISADVANTAGES

ADVANTAGES :

- Time efficient.
- Key matching technique
- Easy apply
- The model can capture the specific interests of a user, and can recommend niche items that very few other users are interested in.
- This makes it easier to scale to a large number of users.

DISADVANTAGES:

- Since the feature representation of the items are hand-engineered to some extent, this technique requires a lot of domain knowledge. Therefore, the model can only be as good as the hand-engineered features.
- The model can only make recommendations based on existing interests of the user. In other words, the model has limited ability to expand on the users' existing interests

11. CONCLUSION

In this project, Content-Based Filtering and Collaborative Filtering of recommendations have been compared. Additionally, an aggregation plus recommender system has been devised.

Content-Based Filtering recommends the results based on matching the personal

preferences of the user with the given document whereas collaborative filtering recommends based on the preferences of fellow users. On evaluating both of these methods, it was concluded that a hybrid system of both of these overcomes the limitations of both of them and increases the efficiency of ranking. Problems of cold start, sparse database, scalability, and lack of trend recommendation have been eliminated. The proposal is to design a Job Recommender system that prioritizes quality over quantity. While there are websites and job listing portals already recommending jobs to job seekers based on their profiles, this research on aggregate quality recommendations has been achieved by crawling selectively, overcoming the limitations. A fully functioning user interface was developed to combine everything together to give the user a seamless experience.

12. FUTURE SCOPE

Future works in the case of Personalized Job Recommendation Systems are the utilization of the user-preferred location to get job recommendations based on jobs in organizations established in nearby areas. This can be done by extracting the latitudes and longitudes of the user-preferred location and computing the euclidean distances between the latitudes and longitudes of the organization location.

This filters out other jobs that fall far from the user-preferred location and gives a more accurate job recommender. As part of the future work, we plan to use features of similar candidates and jobs in sequence information. As of now, recommendation using similar candidates and jobs forms part of non-machine learning based recommendations and the initial result seems promising. Finally, it would be interesting to extend our methodology to other recommender systems.

13. APPENDIX

Source Code

login.html

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>My Page Title</title>
    <link rel="icon" type="image/x-icon" href="images/favicon.ico">
    <link rel="stylesheet" href="{ { url_for('static', filename='css/style.css') } }">
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/css/bootstrap.min.css"
integrity="sha384-xOoIHfLEh07PJGoPkLv1IbcEPTNtaed2xpHsD9ESMhqIYd0nLMwNLD69Npy4HI+N" crossorigin="anonymous">

  </head>
  <body >
    <div class="tit">
      <h2>Jobs Map</h2>
    </div>
    <form class="login-form" action="/dashbord" method="get">

      <div class="body-img">

        <div class="">
          <div class="heading">
            <P class="title-txt"><span class=".search"></span> <br><span class="for"></span>
<br><span class="job"> </span> </P>
          </div>
          <div class="botton-heading">
            <P class="bottom-txt">GET HIRED !!</P>
          </div>
        </div>
      </div>
    </form>
  </body>
</html>
```

```

<div class="login-box">
  <div class="title-box">
    <h1>Sign in</h1>

  </div>
  <div class="mail-box">
    <input class="txt-box form-control" type="email" name="username" placeholder=" Email
id" title="Email or user name">
  </div>
  <div class="password">
    <input class="txt-box form-control" type="password" name="password"
placeholder="password" title="password">
  </div>
  <div class="btu-login">
    <button class="log-btn btn-primary" type="login"
name="login" title="login">login</button>

  </div>
  <hr class="line">
  <div class="create-acc">
    <p>don't have account ? <a href="register" title="create new account">create
account</a><span>{{ msg }}</span>

  </p>
</div>
</div>
</div>
</div>
</div>

</form>
<script>
window.watsonAssistantChatOptions = {

```

```

integrationID: "1e37e3f8-b996-4518-935d-5d45ea53a543", // The ID of this integration.
region: "au-syd", // The region your integration is hosted in.
serviceInstanceID: "6a00d3de-c85d-45d1-ac93-3662219a5b2b", // 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>
```

register.html

```

<!DOCTYPE html>
<html lang="en" dir="ltr">
<head>
<meta charset="utf-8">
<title>My Page Title</title>
<link rel="icon" type="image/x-icon" href="images/favicon.ico">
<link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/css/bootstrap.min.css"
integrity="sha384-xOoIHfLEh07PJGoPkLv1IbcEPTNtaed2xpHsD9ESMhqIYd0nLMwNLD69Npy4HI+N"
crossorigin="anonymous">

```

```

<link rel="stylesheet" href="{{ url_for('static', filename='css/register.css') }}">
</head>
<body>

```

```
<div class="tit">
```

```
    <h2                                style="color:white;margin-left:-87%"><a                                href="#"
style="text-decoration:none;color:white">Jobs Map</a></h2>
```

```
</div>
```

```
<form class="" action="/register" method="post">
```

```
<div class="body-img">
```

```
    <div class="register-box">
```

```
        <div class="re-Title">
```

```
            <h2>Create a new account</h2>
```

```
            <hr>
```

```
        </div>
```

```
        <div class="lastFirst-name">
```

```
            <div class="first-name">
```

```
                <input class="txt-box form-control" type="text" name="firstname" placeholder="First
name" title="First name">
```

```
            </div>
```

```
            <div class="last-name">
```

```
                <input class="txt-box form-control" type="text" name="lastname" placeholder="Last
name" title="Last name">
```

```
            </div>
```

```
        </div>
```

```
    <div class="email">
```

```
        <input class="txt-box form-control" type="email" name="email" placeholder=" Email
id" title="Email ">
```

```
    </div>
```

```
    <div class="password">
```

```
        <input class="txt-box form-control" type="password" name="password"
placeholder="Password" title="password">
```

```
    </div>
```

```

        <div class="signup">
            <button class="signup-btn btn-primary" type="submit" name="login" title="login">Sign
up</button>
        </div>
    </div>
</div>

```

```

</form>
<div class="msg">{{ msg }}</div>
<script>
window.watsonAssistantChatOptions = {
integrationID: "1e37e3f8-b996-4518-935d-5d45ea53a543", // The ID of this integration.
region: "au-syd", // The region your integration is hosted in.
serviceInstanceID: "6a00d3de-c85d-45d1-ac93-3662219a5b2b", // 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>

```

set profile.html

```

<!DOCTYPE html>
<html lang="en" dir="ltr">
    <head>

```



```
<meta charset="utf-8">
<title>SET ACCOUNT</title>

<link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/css/bootstrap.min.css"
integrity="sha384-xOoIHfLEh07PJGoPkLv1IbcEPTNtaed2xpHsD9ESMhqIYd0nLMwNLD69Npy4HI+N"
crossorigin="anonymous">

<link rel="stylesheet" href="{{ url_for('static', filename='css/setaccount.css') }}">
<link rel="icon" type="image/x-icon" href="images/favicon.ico">
</head>
<body>
<div class="tit">
<h2 style="color:white;margin-left:-87%"><a href="/login"
style="text-decoration:none;color:white">Jobs Map</a></h2>
</div>
<div class="msg">{{ msg }}</div>
<form class="setAccount" action="/setprofile" method="post" enctype="multipart/form-data">
<div class="profile">
<h2>SETUP PROFILE</h2>
<div class="name">

<div class="firstname">
<label>First name</label>
<input class="fname txt-box form-control" type="text" name="firstname"
placeholder="First name" title="First name">
</div>

<div class="lastname">
<label>Last name</label>
<input class="lname txt-box form-control" type="text" name="lastname" placeholder="Last
name" title="First name">
</div>
```

```
</div>
<div class="email">
  <label>Email</label>
  <input class="email-box txt-box form-control" type="email" name="email" title="Email"
placeholder="Email">
</div>
<div class="college">
<label>College Name</label>
<input class="clgname txt-box form-control" type="text" name="college" placeholder="College
Name">
</div>
<div class="education">
  <label>Qulification</label>
  <input class="Qulification txt-box form-control" type="text" name="qulification"
placeholder="Qulification">

</div>

<div class="resume">
  <label>Upload resume </label>

  <input class="resume-Upload" type="file" name="resume" title="Upload resume">

</div>
<div class="description">
  <label>Descriptions</label>
  <textarea class="decription-box" name="decription" rows="3" cols="60" placeholder="Add
description"></textarea>

</div>
<div class="description">
```

```
<label>Technical Skills</label>
<textarea class="decription-box" name="skills" rows="2" cols="60" placeholder="Add skill
sets"></textarea>
</div>
```

```
<div class="intrest">
<label>Filed of intrest</label>
<select class="interest-opt" name="interest">
  <option value="Select">select</option>
  <option value="web development">web developmen</option>
  <option value="Cloud computing">Cloud computing</option>
  <option value="Data sciencet">Data science</option>
  <option value="Data analytics">Data analytics</option>
  <option value="Blockchin">Blockchin</option>
</select>
</div>
```

```
<div class="next">
  <button class ="next-btn btn-primary" type="submit" name="button">Finish </button>
</div>
```

```
</div>
```

```
</form>
```

```
<script>
```

```
window.watsonAssistantChatOptions = {
integrationID: "1e37e3f8-b996-4518-935d-5d45ea53a543", // The ID of this integration.
region: "au-syd", // The region your integration is hosted in.
serviceInstanceID: "6a00d3de-c85d-45d1-ac93-3662219a5b2b", // 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>

```

display.html

```

<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">

    <link rel="stylesheet" href="{ { url_for('static', filename='css/display.css') } }">
    <link rel="icon" type="image/x-icon" href="images/favicon.ico">
    <title></title>
  </head>
  <body>
    <div class="tit">
      <div class="heading">
        <h2><a href="#">Jobs Map</a></h2>
      </div>

      <div class="row ">
        <a class="btn " href="/dashbord">FIND JOBS </a>
        <a class="btn " href="/logout">LOGOUT</a>
      </div>

```

</div>

<div class="details-box">

<h2><u>Your details</u></h2>

<table class="tab-box">

<tr>

<td>firstname:</td>

<td>Prasath</td>

</tr>

<tr>

<td>lastname:</td>

<td>JS</td>

</tr>

<tr>

<td>Email ID:</td>

<td>prasathjs@gmail.com</td>

</tr>

<tr>

<td>College:</td>

<td>ABC college</td>

</tr>

<tr>

<td>Qualification:</td>

<td>BE-CSE</td>

</tr>

<tr>

<td>Skills:</td>

<td>JAVA,C++</td>

</tr>

<tr>

<td>Intrest:</td>

<td>Web Development</td>

</tr>

</div>

```

<script>
window.watsonAssistantChatOptions = {
  integrationID: "1e37e3f8-b996-4518-935d-5d45ea53a543", // The ID of this integration.
  region: "au-syd", // The region your integration is hosted in.
  serviceInstanceID: "6a00d3de-c85d-45d1-ac93-3662219a5b2b", // 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>

```

dashboard.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="icon" type="image/x-icon" href="images/favicon.ico">
  <title>JOBPORTAL | HOME</title>
  <meta charset="UTF-8">

  <link rel="icon" type="image/png" sizes="16x16" href="/assets/img/favicon-32x32.png">

  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css"

```

```
integrity="sha384-JcKb8q3iqJ61gNV9KGb8thSsNjpSL0n8PARn9HuZOnIxN0hoP+VmmDGMN5t9UJ0
Z" crossorigin="anonymous">
```

```
<link rel="stylesheet" href="{{ url_for('static', filename='css/dash.css') }}">
```

```
</head>
```

```
<body>
```

```
<nav class="tit navbar sticky-top navbar-expand-lg navbar-light">
```

```
<div class="container-fluid">
```

```
<h2>Jobs Map</h2>
```

```
<div class="row donate-sponsor">
```

```
<a class="btn" href="/display">Profile</a>
```

```
<a class="btn" href="/logout">Logout</a>
```

```
</div>
```

```
</div>
```

```
</nav>
```

```
<!-- navbar ends -->
```

```
<!-- what we focus on -->
```

```
<section class="our-focus">
```

```
<div class="container">
```

```
<h2 class="text-center mt-3">Available Jobs</h2>
```

```
<div class="row ml-3 mt-3">
```

```
<div class="col-lg-3 mr-5" id="focus-first">
```

```
<div class="card" style="width: 19rem;">
```

```
<!-- 
```

```
<div class="card-body">
```

```
<h5 class="card-title">Amazon</h5>
```

```
<p class="card-text">Software Development Engineer-Test
```


</p>

<a

href="https://www.amazon.jobs/en/jobs/2166286/software-development-engineer-test"

target="_blank" class="btn btn-primary">Apply Now

</div>

</div>

</div>

<div class="col-lg-3 mr-5" id="focus-second">

<div class="card" style="width: 20rem;">

<!--

<div class="card-body">

<h5 class="card-title">Cognizant</h5>

<p class="card-text">Programmer Analyst Trainee </p>

Apply

Now

</div>

</div>

</div>

<div class="col-lg-3 ml-5" id="focus-third">

<div class="card" style="width: 20rem;">

<!--

<div class="card-body">

<h5 class="card-title">Canonical</h5>

<p class="card-text"> Security Engineer- Ubuntu

</p>

<a

href="https://boards.greenhouse.io/canonicaljobs/jobs/4698780?gh_src=4aa335b81us"

target="_blank"

class="btn btn-primary">Apply Now

</div>

</div>

</div>

</div>

```
</div>
</section>
<!-- focus section ends -->
<section class="our-focus">
  <div class="container">

    <div class="row ml-3 mt-3">
      <div class="col-lg-3 mr-5" id="focus-first">
        <div class="card" style="width: 19rem;">
          <!--  -->
          <div class="card-body">
            <h5 class="card-title">Paypal</h5>
            <p class="card-text">Software Engineer
          </p>
```

Apply Now

```
    </div>
  </div>
</div>
<div class="col-lg-3 mr-5" id="focus-second">
  <div class="card" style="width: 20rem;">
    <!--  -->
    <div class="card-body">
      <h5 class="card-title">Amazon</h5>
      <p class="card-text">Data Scientist Intern</p>
      <a href="https://www.amazon.jobs/en/jobs/2213292/data-scientist-intern"
target="_blank" class="btn btn-primary">Apply Now</a>
```

```
    </div>
  </div>
</div>
<div class="col-lg-3 ml-5" id="focus-third">
  <div class="card" style="width: 20rem;">
```

```
<!-- 
```

```
<div class="card-body">
```

```
<h5 class="card-title">Conxai</h5>
```

```
<p class="card-text">Deep Learning Engineers </p>
```

```
<a
```

```
href="https://join.com/companies/conxai/6339429-deep-learning-engineers-internship?pid=24a1b46991e3de1fbcf0s" target="_blank" class="btn btn-primary">Apply Now</a>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
</section>
```

```
<section class="our-focus">
```

```
<div class="container">
```

```
<div class="row ml-3 mt-3">
```

```
<div class="col-lg-3 mr-5" id="focus-first">
```

```
<div class="card" style="width: 19rem;">
```

```
<!-- 
```

```
<div class="card-body">
```

```
<h5 class="card-title">Hexaware</h5>
```

```
<p class="card-text">Application Front end Developer
```

```
</p>
```

```
<a
```

```
href="https://jobs.hexaware.com/job/application-front-end-developer-175397?cp=74&p=9&utm_source=Linkedin&utm_medium=referral&utm_campaign=wrap" target="_blank" class="btn btn-primary">Apply Now</a>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
<div class="col-lg-3 mr-5" id="focus-second">
```

```

        <div class="card" style="width: 20rem;">
            <!-- 
            <div class="card-body">
                <h5 class="card-title">Amazon</h5>
                <p class="card-text">Data Scientist I</p>
                <a href="https://www.amazon.jobs/en/jobs/1744525/data-scientist-i"
target="_blank" class="btn btn-primary">Apply Now</a>
            </div>
        </div>
    </div>
</div>
<div class="col-lg-3 ml-5" id="focus-third">
    <div class="card" style="width: 20rem;">
        <!-- 
        <div class="card-body">
            <h5 class="card-title">Microsoft</h5>
            <p class="card-text">Software Engineer II</p>
            <a
href="https://careers.microsoft.com/us/en/job/1482841/Software-Engineer-II" target="_blank" class="btn
btn-primary">Apply Now</a>
        </div>
    </div>
</div>
</div>
</div>
</div>
</section>

```

```

<section class="our-focus">
    <div class="container">

```

```

        <div class="row ml-3 mt-3">
            <div class="col-lg-3 mr-5" id="focus-first">
                <div class="card" style="width: 19rem;">
                    <!-- 

```

```
<div class="card-body">
  <h5 class="card-title">Amazon</h5>
  <p class="card-text">Business Analyst - I
</p>
<a
href="https://www.amazon.jobs/en/jobs/2299370/business-analyst-i?cmpid=SPLICX0248M&ss=paid&utm_campaign=cxro&utm_content=job_posting&utm_medium=social_media&utm_source=linkedin.com"
class="btn btn-primary">Apply Now</a>
```

```
</div>
</div>
</div>
<div class="col-lg-3 mr-5" id="focus-second">
  <div class="card" style="width: 20rem;">
    <!-- 
```

-->

```
  <div class="card-body">
    <h5 class="card-title">Amazon</h5>
    <p class="card-text">Robotics - Electrical Engineer Intern</p>
<a
href="https://www.amazon.jobs/en/jobs/2259702/amazon-robotics-electrical-engineer-intern-summer-2023" class="btn btn-primary">Apply Now</a>
```

```
</div>
</div>
</div>
<div class="col-lg-3 ml-5" id="focus-third">
  <div class="card" style="width: 20rem;">
    <!--  -->
    <div class="card-body">
      <h5 class="card-title">Wipro</h5>
      <p class="card-text">Business Analyst</p>
```

```

<a
href="https://careers.wipro.com/careers-home/jobs/2931947?lang=en-us&utm_source=Linkedin"
class="btn btn-primary">Apply Now</a>
</div>
```

```
        </div>
    </div>
</div>
</div>
</section>
<!-- footer starts -->
<!-- Site footer -->

</body>
</html>
```

style.css

```
*{
    margin: 0;
    padding: 0;
}
h2{
    margin-left: 3%;
    margin-top: 2%;
    color: white;
}
.tit{
    height: 50px;
    width: 100%;
    background: #66b3ff;
}
body{
    margin: 0;
    background: #ccebff;
}
.body-img{

    background-image:url("loginpage.png");
```

```
height: 100%;  
width: 100%;  
background-position: center;  
background-size: cover;  
position: relative;  
}
```

```
.heading{  
height: 100px;  
width: 430px;  
position: relative;  
left: 4%;  
top: 140 px;  
}
```

```
.bottom-heading{  
height: 100px;  
width: 430px;  
position: relative;  
left: 12%;  
top: 500px;  
}
```

```
.login-box{  
height: 500px;  
width: 370px;  
background: rgba(255, 255, 255, 0.6);  
position: relative;  
bottom: 150px;  
left: 60%;  
border-radius: 5%;  
  
}
```



```
.mail-box{
    position: relative;
    height: 10%;
    width: 80%;
    left: 10%;
    top:5%;
}

.password{
    position: relative;
    height: 10%;
    width: 80%;

    left: 10%;
    top:15%;
}

.title-box{
    position:relative ;

    height: 100px;
    padding-top: 10%;
    padding-left: 30%;
    padding-bottom: 0;
}

.btu-login{
    position: relative;
    height: 10%;
    width: 80%;
    left: 10%;
    top:25%;
}

.log-btn{
    position: relative;
    height: 100%;
    width:100%;
```

```
text-align: center;
border-radius: 12px;
font-size: 100%;
}

.txt-box{
height: 100%;
width: 100%;
border-radius: 12px;
text-align: left;
padding-left: 5%;
}

.line{
position: relative;
margin-left: 20%;
margin-right: 20%;
top: 30%;
}

.create-acc{
height: 10%;
width: 80%;
left: 10%;
position: relative;
top: 30%;
font-size: 12px;
text-align: center;
}

.title-txt{
font-weight: bold;
font-size: 40px;
font-family: serif;
}

.bottom-txt{
```

```
font-size:40px;
position: relative;
bottom: 100%;
left: 10%;
font-weight: bolder;
font-family:serif;
color: #fff;
}
.for{
position: relative;
left: 20%;
}
.job{
position: relative;
left: 30%;
}
.search{

position: relative;
right: 30px;
}
```

app.py

```
import ibm_boto3
from ibm_botocore.client import ClientError, Config
from flask import Flask, render_template, request, redirect, url_for, session
import ibm_db
import re
```

```
# Constants for IBM COS values
```

```
COS_ENDPOINT = "https://s3.jp.tok.cloud-object-storage.appdomain.cloud" # Current list available at
https://control.cloud-object-storage.cloud.ibm.com/v2/endpoints
```

```
COS_API_KEY_ID      =      "LWGNzbVTji5V5MG0Doq4jy-Lr4CaAntqoeX6Eh9LMf7T"      #      eg
"W00YixxxxxxxxxxMB-odB-2ySfTrFBIQQWanc--P3byk"
```

```
COS_INSTANCE_CRN      =
"crn:v1:bluemix:public:iam-identity::a/c2c7b865995e48f082d98cc5bb07e3e1::serviceid:ServiceId-3921a
51c-3155-4709-b889-8b111dfd3532"
"crn:v1:bluemix:public:cloud-object-storage:global:a/3bf0d9003xxxxxxxxxx1c3e97696b71c:d6f04d83-6
c4f-4a62-a165-696756d63903::"
```

```
# Create resource
```

```
cos = ibm_boto3.resource("s3",
    ibm_api_key_id=COS_API_KEY_ID,
    ibm_service_instance_id=COS_INSTANCE_CRN,
    config=Config(signature_version="oauth"),
    endpoint_url=COS_ENDPOINT
)
```

```
app = Flask(__name__)
app.secret_key='a';
```

```
conn      =
ibm_db.connect("DATABASE=bludb;HOSTNAME=2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io9010
8kqb1od8lcg.databases.appdomain.cloud;PORT=32328;SECURITY=SSL;SSLServerCertificate=DigiCer
tGlobalRootCA.crt;UID=yhz29794;PWD=9CZ26CLv18tpSPkT",";")
```

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

```
def login():
```

```
    global userid
```

```
    msg="
```

```
if request.method == 'POST':
```

```
    username=request.form['username']
```

```
    password=request.form['password']
```

```
    sql="SELECT * FROM user WHERE usernam=? AND password=?"
```

```

stmt=ibm_db.prepar(conn,sql)
ibm_db.bind_param(stmt,1,username)
ibm_db.bind_param(stmt,2,password)
ibm_db.execute(stmt)
account = ibm_db.fetch_assoc(stmt)
print(account)
if account:
    session['loggedin']=True
    session['id']=account['USERNAME']
    userid=account['USERNAME']
    session['username']=account['USERNAME']
    return render_template('dashboard.html', msg = msg)
else:
    msg='incorrect id/password'
return render_template('login.html',msg=msg)

```

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

```
def register():
```

```

    msg = "
    if request.method == 'POST' :
        fitstname = request.form['firstname']
        lastname = request.form['lastname']
        email = request.form['email']
        password = request.form['password']
        sql = "SELECT * FROM user WHERE email =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,email)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print(account)
        if account:
            msg = 'Account already exists !'
        elif not re.match(r'^@]+@^[^@]+\.[^@]+', email):

```

```

        msg = 'Invalid email address !'
    elif not re.match(r'[A-Za-z0-9]+', password):
        msg = 'password must contain characters and numbers !'
    else:
        insert_sql = "INSERT INTO user VALUES (?, ?, ?, ?)"
        prep_stmt = ibm_db.prepare(conn, insert_sql)
        ibm_db.bind_param(prepare_stmt, 1, firstname)
        ibm_db.bind_param(prepare_stmt, 2, lastname)
        ibm_db.bind_param(prepare_stmt, 3, email)
        ibm_db.bind_param(prepare_stmt, 4, password)
        ibm_db.execute(prepare_stmt)
        return render_template('set profile.html',)

    elif request.method == 'POST':
        msg = 'Please fill out the form !'
        return render_template('register.html', msg = msg)

def multi_part_upload(bucket_name,iteam_name,file_path):
    try:
        part_size=1024 * 1024 *5

        file_threshold=1024 * 1024 *15

        transfre_config=ibm_boto3.s3.transfer.TransferConfig(

            multipart_threshold=file_threshold,
            multipart_chunksize=part_size
        )

        with open(file_path,"rb") as file_data:
            cos.Object(bucket_name,iteam_name).upload_fileobj(
                Fileobj=file_data,
                Config=transfre_config

            )

```

```

except ClientError as be:
    print("client error :[0]\n".format(be))
except Exception as e:
    print("unable to complete upload")
@app.route('/setprofile',methods=['GET','POST'])
def setprofile():
    msg = "
    if request.method == 'POST' :
        fitstname = request.form['firstname']
        lastname = request.form['lastname']
        email = request.form['email']
        college = request.form['college']
        qulification = request.form['qulification']
        decription = request.form['decription']
        skills= request.form['skills']
        interest= request.form['interest']

        bucket='resumes-storage'
        file_name=request.form['email']
        file=request.files['resume']
        multi_part_upload(bucket,file_name,file.filename)

    sql = "SELECT * FROM profile WHERE email =?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,email)
    ibm_db.execute(stmt)
    account = ibm_db.fetch_assoc(stmt)
    print(account)
    if account:
        msg = 'Account already exists !'

```

```

elif not re.match(r'^@]+@[^@]+\.[^@]+', email):
    msg = 'Invalid email address !'

else:
    insert_sql = "INSERT INTO profile VALUES (?, ?, ?, ?, ?, ?, ?, ?)"
    prep_stmt = ibm_db.prepare(conn, insert_sql)
    ibm_db.bind_param(prepare_stmt, 1, firstname)
    ibm_db.bind_param(prepare_stmt, 2, lastname)
    ibm_db.bind_param(prepare_stmt, 3, email)
    ibm_db.bind_param(prepare_stmt, 4, college)
    ibm_db.bind_param(prepare_stmt, 5, qualification)
    ibm_db.bind_param(prepare_stmt, 6, decription)
    ibm_db.bind_param(prepare_stmt, 7, skills)
    ibm_db.bind_param(prepare_stmt, 8, interest)
    ibm_db.execute(prepare_stmt)
    return render_template('dashbord.html')

elif request.method == 'POST':
    msg = 'Please fill out the form !'
    return render_template('set profile.html', msg = msg)
@app.route('/display')
def display():

    return render_template('display.html')

@app.route('/logout')

def logout():
    session.pop('logged_in', None)
    session.pop('email', None)

    return render_template('login.html')

@app.route('/dashbord')

```



```
def dash():
```

```
    return render_template('dashbord.html')
```

```
if __name__ == '__main__':
```

```
    app.run(debug = True)
```

14. GitHub & Project Demo Link

Demo link-

<https://youtu.be/V51sIbw6QeM>

GitHub link <https://github.com/IBM-EPBL/IBM-Project-4254-1658726285.git>