SKILL / JOB RECOMMENDER APPLICATION

PROJECT REPORT

Submitted by

NIVETHA A (TL)	**812719104028**
PARTHASARATHI P (TM 1)	**812719104030**
RAHUL RAJ S (TM 2)	**812719104032**

TABALE OF CONTENTS

CHAPTER-I

INTRODUCTION

The world has seen an important increase in the demand for Cloud-based applications. This has in turn increased the demand for Cloud application development.As a result, the past few years have had a consolidation of the Cloud computing market.Cloud apps and services are used,directly or indirectly,by almost everyone.Businesses have also increased their use of Cloud-based applications and services,even if they sometimes don't know it.If you use SaaS tools, you are surely using a Cloud app.However,Cloud apps are more than just that.Cloud-based applications,also known as Cloud apps,seem to be taking over.In theory,a Cloud app is one that uses Cloud-based services. So, whether an app is mobile or web, they probably use some sort of Cloud service.What really differentiates a Cloud app from a native one is the extent to which they use Cloud services.Increased dependence on the Cloud's processing power is the result of companies building innovative and creative solutions to all sorts of problems that use technology to do things that were previously impossible.Thanks to the ability to process large amounts of data (Big Data) through third party-owned IT infrastructure, companies can perform massive calculations and deliver top services. Cloud services have opened the possibility for many web-based Cloud applications, also known as web apps. A web app is one where most of the computation occurs in the Cloud, not on the device itself, and is usually built with the use of Cloud application development services. A new form of a web app, known as a Progressive Web App (PWA), is also seeing an increase in popularity. Cloud application development is the process through which a Cloudbased app is built.It involves different stages of software development,each of which prepares your app to go live and hit the market. The best Cloud app development teams use DevOps practices and tools like Kubernetes. which means being able to build your Cloud app using any technology you prefer.Most apps built using the Cloud are highly dependent on the Cloud to operate.

1.1 PROJECT OVERVIEW

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.

1.2 Purpose

The main aim of the project is to building a model with a skill recommender solution through which the fresher or the skilled person can log in and find the jobs by using the search option or they can directly interact with the chatbot and get their dream job.

CHAPTER-2

LITERATURE SURVEY

2.1 Existing problem

Skill -based recommendation services platforms wants to turn into a leading source of the job recommender platform. The job and skill recommender system is expected to reduce unemployment and improve the skills of job seekers to boost the country's economy. Job recommender is giving guidance for job seekers to get quick contact for those jobs. Immediate announcement of the latest job openings, and walk-in interviews with different locations and offers you full details of recruitment such as the number of job vacancies, eligibility criteria, interview date, required qualifications etc.

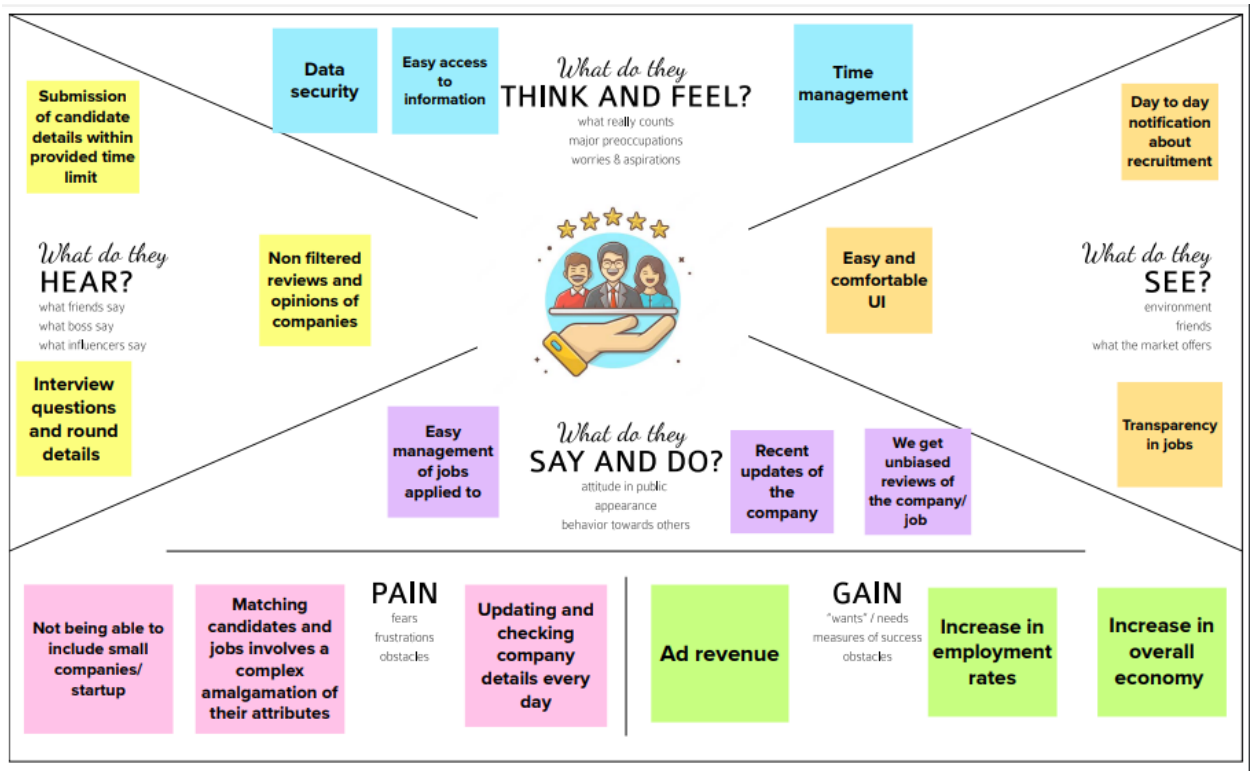
1.4 Problem Statement Definition

The main aim of the project is to develop an end-to-end web application capable of displaying the current job openings based on the user skill set. Users will interact with the chat-bot and can get recommendations based on their skills. This Application uniquely identifies the user's skills recommend the job according to the user's interest.

CHAPTER-3

IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming

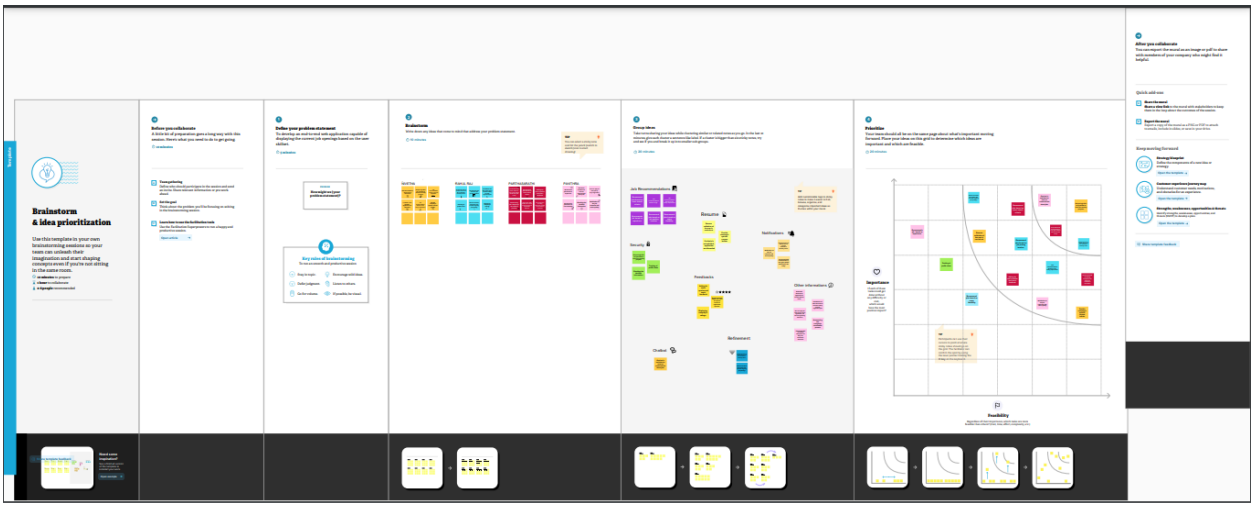
Idea 1: Ideation is often closely related to the practice of brainstorming, a specific technique that is utilized to generate new ideas. A principal difference between ideation and brainstorming is that ideation is commonly more thought of as being an individual pursuit, while brainstorming is almost always a group activity. Brainstorming is usually conducted by getting a group of people together to come up with either general new ideas or ideas for solving a specific problem or dealing with a specific situation.

Idea 2: For example, a major corporation that recently learned it is the object of a major lawsuit may want to gather together top executives for a brainstorming session on how to publicly respond to the lawsuit being filed.

Idea 3: Participants in a brainstorming session are encouraged to freely toss out whatever ideas may occur to them. The thinking is that by generating a large number of ideas, the brainstorming group is likely to come up with a suitable solution for whatever issue they are addressing.

Idea 4: The lines between ideation and brainstorming have become a bit more blurred with the development of several brainstorming software programs, such as Bright idea and Idea wake. These software programs are designed to encourage employees of companies to generate new ideas for improving the companies' operations and, ultimately, bottom-line profitability.

Brainstorming:



3.3 proposed solution

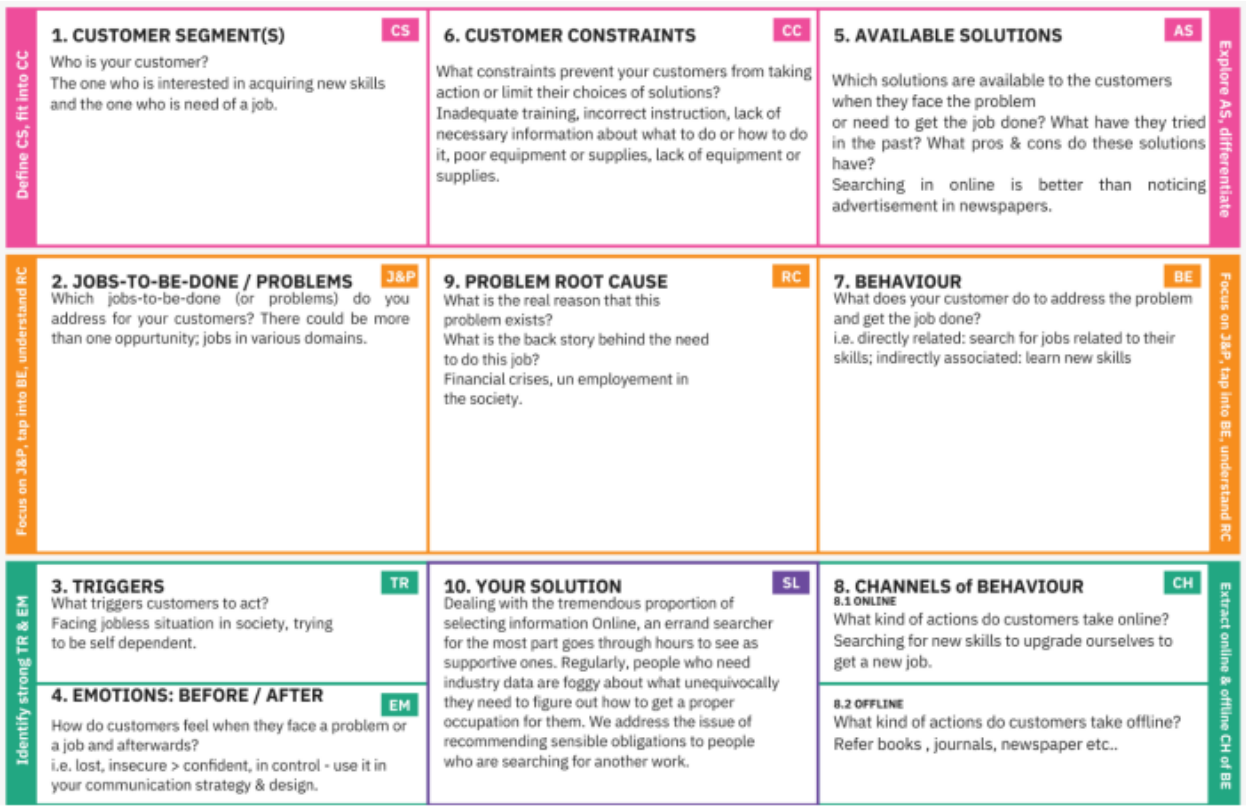
S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Every industry has a lot of career opportunities, but job seekers are unaware of them. The unemployability crisis can be solved if every job seeker receives the right career guidance and proper job role training. So, to eradicate the unemployment crisis, for the job seekers to

		find a job they desire, match their qualifications and skills.
2.	Idea / Solution description	To develop an end-to-end web application capable of displaying the current job openings based on the user skill set. Users will interact with the chat-bot and can get recommendations based on their skills.
3.	Novelty / Uniqueness	This Application uniquely identifies the user's skills recommend the job according to the user's interest.
4.	Social Impact / Customer Satisfaction	The job and skill recommender system is expected to reduce unemployment and improve the skills of job seekers to boost the country's economy. The customer satisfaction can be measured by customer loyalty and customer reviews after deployment of the project.
5.	Business Model (Revenue Model)	A subscription model can be provided for both employees and employers with additional costs for features along with recurring monthly or yearly costs.
6.	Scalability of the Solution	It is lifelong recommender app. Once the user has login to this application, he will be notified about the job up to date.

3.4 Problem 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.



CHAPTER-4

REQUIREMENT ANALYSIS

4.1 Functional Requirement:

- Sign In / Login
- Profile Registration
- Job profile display
- Chatbot
- Job registration
- Logout

4.2 Non-Functional requirements:

- Usability
- Security
- Reliability
- Performance
- Availability
- Scalability

CHAPTER-5
PROJECT DESIGN

5.1 Data flow diagram:

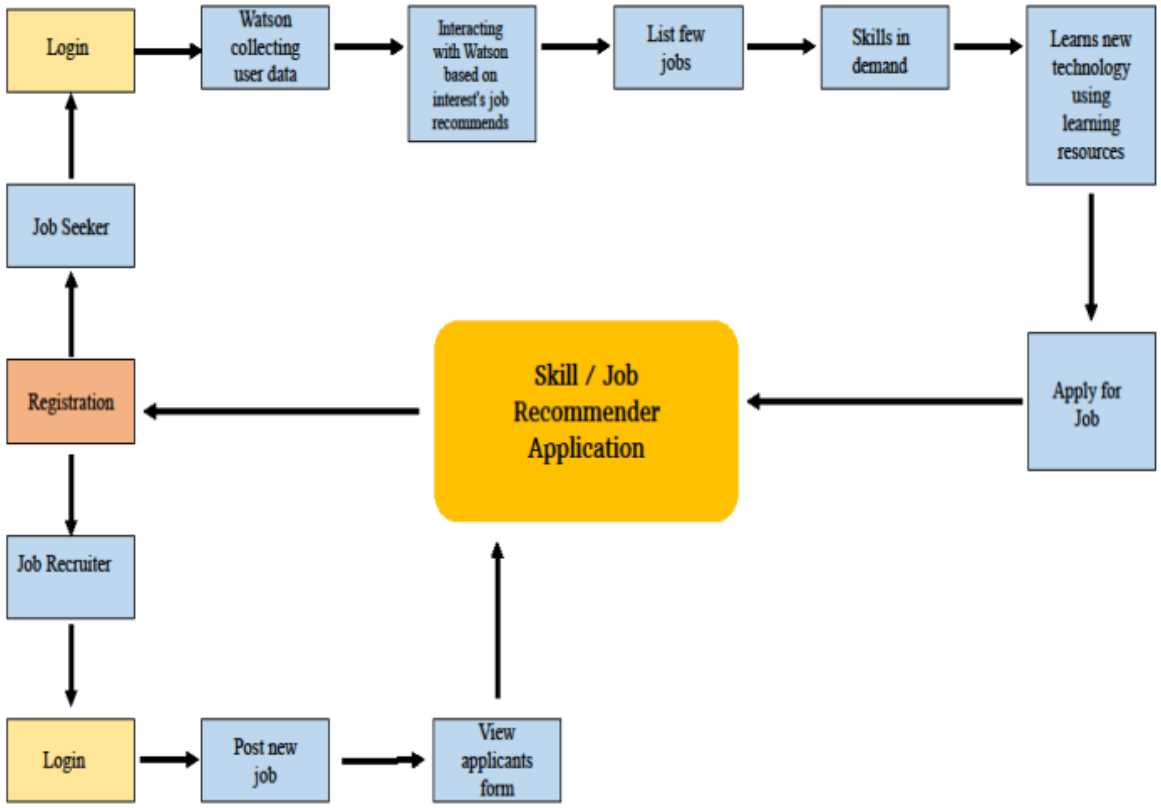
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both.

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

The objective of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communication tool between a system analyst and any person who plays a part in the order that acts as a starting point for redesigning a system. The DFD is also called as a data flow graph or bubble chart.

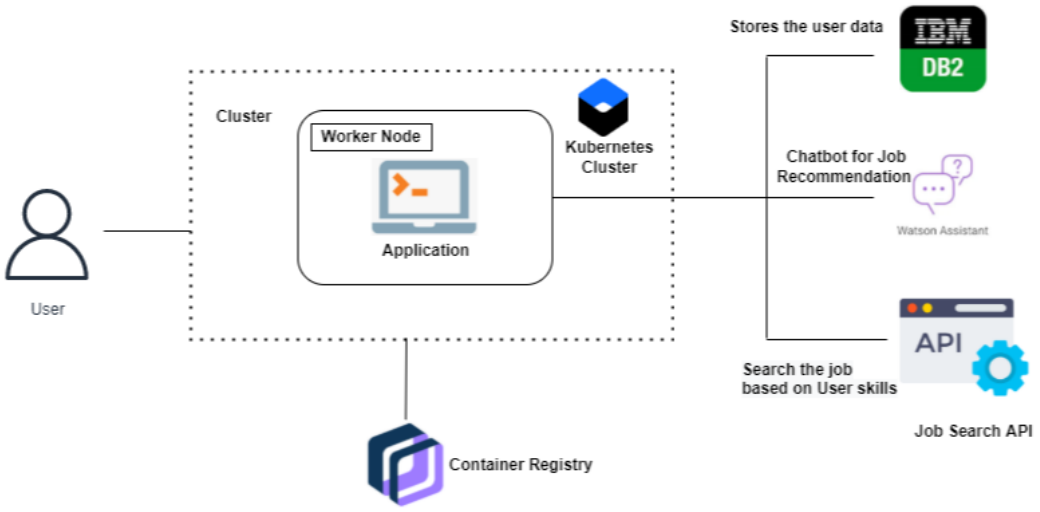
A set of parallel lines shows a place for the collection of data items. A data store indicates that the data is stored which can be used at a later stage or by the other processes in a different order. The data store can have an element or group of elements. The DFD may be used to perform a system or software at any level of abstraction. DFDs may be partitioned into levels that represent increasing information flow and functional detail. Then the system is decomposed and described as a DFD with multiple bubbles. Parts of the system represented by each of these bubbles are then decomposed and documented as more and more detailed DFDs.

DATA FLOW DIAGRAM



5.2 Solution & Technical Architecture

Solution Architects are most similar to project managers, ensuring that all parties, including stakeholders, are on the same page and moving in the right direction at all stages. Technical architects manage all activities leading to the successful implementation of a new application. They propose a combination of building blocks that provides the best possible fix. This process is very detailoriented and serves as a connecting piece between enterprise architecture and technical architecture. It also requires a breadth of knowledge in the technical and business inner workings of the company.



5.3 User Stories

A user story is the smallest unit of work in an agile framework. It's an end goal, nota feature, expressed from the software user's perspective.

A user story is an informal, general explanation of a software feature written from the perspective of the end user or customer.

The purpose of a user story is to articulate how a piece of work will deliver a particular value back to the customer. Note that "customers" don't have to be external end users in the traditional sense, they can also be internal customers or colleagues within your organization who depend on your team.

User stories are a few sentences in simple language that outline the desired outcome. They don't go into detail. Requirements are added later, once agreed upon by the team.

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

CHAPTER-6

PROJECT PLANNING & SCHEDULING

Planning - Planning pertains to the process of creating a plan of which materials and resources will be required to fulfil incoming and forecasted demand. This step is crucial to ensure that you have enough materials and resource capacity available to produce your orders on time. This component pertains to the ‘what’ and ‘how’ of any project: what exactly needs to be achieved and how it will be accomplished.

Scheduling - Scheduling pertains to establishing the timing of the use of specific resources of that organization. In production, scheduling involves developing schedules for workers, equipment, and materials. It reflects on the ‘when’ of a project, by assigning the appropriate resources to get the production plan completed within a period of time. Creating optimized production schedules ensures that your facility is able to reduce costs, increase productivity, and deliver goods to customers on time.

In order to create accurate and realistic production plans that allow manufacturers to react quickly to changes, it is important to have a production plan that is aligned with the resource and material scheduling process. Having any discrepancy or divergence between the planning and scheduling process creates inefficiencies that can be costly for your business. The bigger the divergence, the larger the cost.

6.1 Sprint planning and estimation

Planning:

In Sprint Planning, the team decides what it will build in the upcoming Sprint and how they will build it. The team commits to the Sprint goal after breaking down user stories into tasks and doing task-level estimation. Sprint Planning is done by the Product Owner, Scrum Master, and the Team. In Scrum, every project is broken into time blocks called sprints, usually 2-4 weeks long. A sprint planning meeting is when the team (including the Scrum Master, Scrum Product Manager, and Scrum Team) meets to determine which backlog items will be handled in the next sprint.

Estimation

In Scrum Projects, Estimation is done by the entire team during Sprint

Planning Meeting. The objective of the Estimation would be to consider the User

Stories for the Sprint by Priority and by the Ability of the team to deliver during the Time Box of the Sprint.

Product Owner ensures that the prioritized User Stories are clear, can be subjected to estimation, and they are brought to the beginning of the Product Backlog.

As the Scrum Team in total is responsible for the delivery of the product increment, care would be taken to select the User Stories for the Sprint based on the size of the Product Increment and the effort required for the same.

The size of the Product Increment is estimated in terms of User Story Points. Once the size is determined, the effort is estimated by means of

the past data, i.e., effort per User Story Point called Productivity.

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Priority	Acceptance criteria	Team Members
Sprint-1	UI / UX Design	USN-1	As a user, I need to interact with websites. So, this will be the interface between me and website.	High	I can interact with website	rahul raj,nivetha, parthasarathi
Sprint-1	Registration	USN-2	As a user, I can register for the application by entering my email, password, and confirming my password.	High	I can register with email	rahul raj,nivetha, parthasarathi
Sprint-1		USN-3	As a user, I will receive confirmation email once I have registered for the application	High	I can receive confirmation mail	rahul raj,nivetha, parthasarathi
Sprint-1	Login	USN-4	As a user, I can log into the application by entering email & password	Low	I can login to the application by entering login credentials	rahul raj,nivetha, parthasarathi

Sprint-1	Login	USN-4	As a user, I can log into the application by entering email & password	Low	I can login to the application by entering login credentials	rahul raj,nivetha, parthasarathi

Sprint-1	**Flask**	**USN-5**	**As a user, I can access the website in quick span of time.**	**High**	**I can access website quickly**	rahul raj,nivetha, parthasarathi
Sprint-1	**Dashboard**	**USN-6**	**As a user, I can access various tools and services on the website through the help of dashboard.**	**High**	**I can access various tools and services**	rahul raj,nivetha, parthasarathi
			Submission of sprint 1			
Sprint-2	**Database**	**USN-7**	**As a user, I can store the data am providing in the portal.**	**High**	**I can store my data**	rahul raj,nivetha, parthasarathi
Sprint-2	**User Profile**	**USN-8**	**As a user, I can update my profile in the user**	**Medium**	**I can modify my details**	rahul raj,nivetha, parthasarathi
			profile section			
Sprint-2	**SendGrid Integration**	**USN-9**	**As a user, I can get notification through mail services**	**Medium**	**I can get mail**	rahul raj,nivetha, parthasarathi
Sprint-2	**Learning Resources**	**USN-10**	**As a user, I can access resources and knowledge which is useful in developing the skills**	**Medium**	**I can access learning resources**	rahul raj,nivetha, parthasarathi
Sprint-2	**Link to python code**	**USN-11**	**As a user, I can access website fast**	**High**	**I can access website**	rahul raj,nivetha, parthasara

						thi
			Submission of sprint 2			

Sprint-3	**Cloud Storage**	**USN-12**	**As a user, I can store my photos, resumes and much more media that are supported in the webpage.**	**High**	**I can store the medias**	rahul raj,nivetha, parthasarathi
Sprint-3	**Chat-Bot**	**USN-13**	**As a user, I can access the chatbot which is very much useful in knowing several things like recent available jobs and jobs opening etc..**	**High**	**I can interact with chatbot to solve any queries**	rahul raj,nivetha, parthasarathi
Sprint-3	**Integrate to App**	**USN-14**	**As a user, I can interact with chatbot after the integration of chat-bot and webpage.**	**High**	**Easy access with chat-bot**	rahul raj,nivetha, parthasarathi
			Submission of Sprint 3			
Sprint-4	**Docker**	**USN-15**	**As a user, I can access the webpage in any device.**	**High**	**I can access my account in any device**	rahul raj,nivetha, parthasarathi
Sprint-4	**Kubernetes**	**USN-16**	**As a user, I can access the webpage in any device.**	**High**	**I can access my account in any device**	rahul raj,nivetha, parthasarathi
Sprint-4	**Deployment in cloud**	**USN-17**	**As a user, I can access the website any device.**	**High**	**I can access my account in any device**	rahul raj,nivetha, parthasara

						thi

Sprint-4	**Technical support**	**USN-18**	**As a user, I can get a customer care support on the website when am having any queries**	**Medium**	**Helps me to solve queries**	rahul raj,nivetha, parthasarathi
Sprint-4	**Unit Testing**	**USN-19**	**As a user, I can access the website without any interruptions.**	**High**	**I can access the website without any interruptions**	rahul raj,nivetha, parthasarathi
Sprint-4	**Integration Testing**	**USN-20**	**As a user, I can access the website without any interruptions.**	**High**	**I can access the website without any interruptions**	rahul raj,nivetha, parthasarathi
Sprint-4	**System Testing**	**USN-21**	**As a user, I can access the website without any interruptions.**	**High**	**I can access the website without any interruptions**	rahul raj,nivetha, parthasarathi
Sprint-4	**Correction**	**USN-22**	**As a user, I can access the website without any interruptions.**	**High**	**I can access the website without any interruptions**	rahul raj,nivetha, parthasarathi
Sprint-4	**Acceptance Testing**	**USN-23**	**As a user, I can access the website without any interruptions.**	**High**	**I can access the website without any interruptions**	rahul raj,nivetha, parthasara

						thi
			Submission of Sprint 4			

6.2 Sprint delivery schedule

Since sprints take place over a fixed period of time, it's critical to avoid wasting time during planning and development. And this is precisely where sprint scheduling enters the equation.

In case you're unfamiliar, a sprint schedule is a document that outlines sprint planning from end to end. It's one of the first steps in the agile sprint planning process—and something that requires adequate research, planning, and communication.

Teams often run into trouble when they create more than a few schedules. This can create conflict and derail projects midway through their cycles. To ensure things stay on track, one schedule makes sense.

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

Velocity:

We have a 6-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 = sprint duration / velocity

AV = 20 / 6

AV = 3.33

SPRINT DELIVERY SCHEDULE

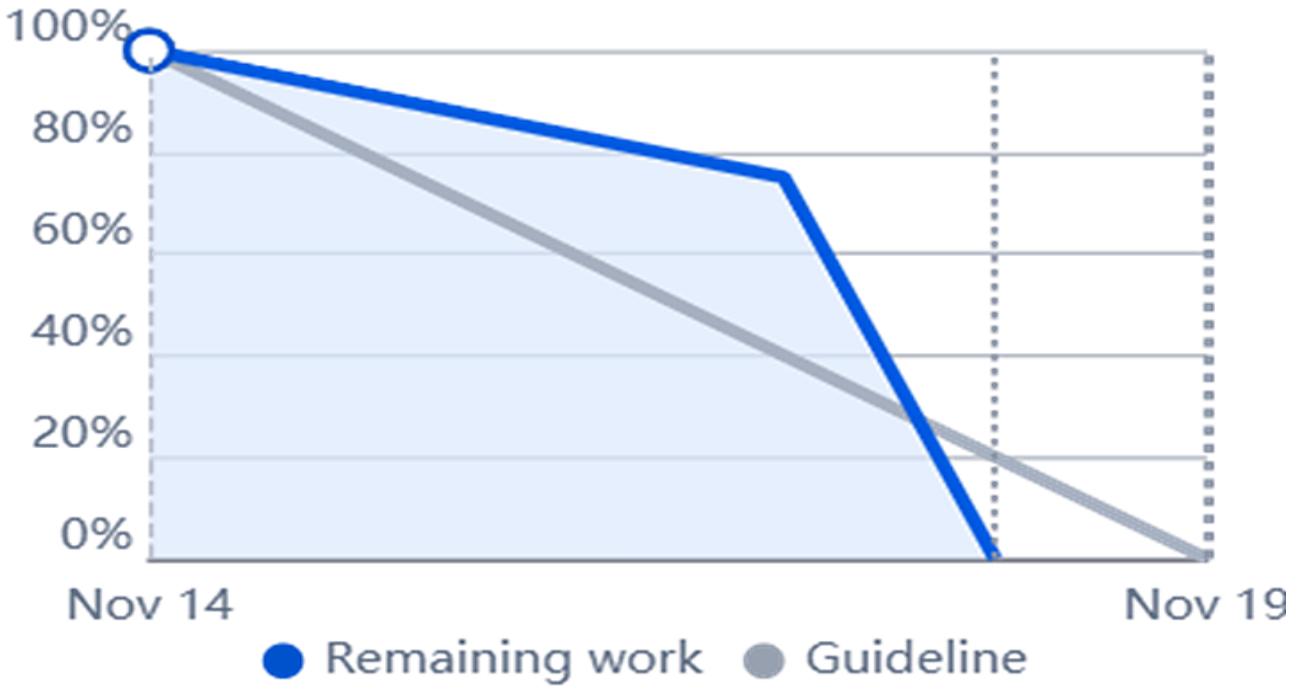
6.3 Reports from JIRA

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.

Typically, in a burn down chart, the outstanding work is often on the vertical axis,

with time along the horizontal. It is useful for predicting when all of the work will be completed. In the Daily Scrum the Development Team updates the Sprint Burn Down and plots the remaining work of the day.

Burndown Chart(Sprint 4):



chapter 7

CODING AND SOLUTIONING

7.1 Feature 1

```
import configparser

import sendgrid
from sendgrid.helpers.mail import Mail

config                                  =
configparser.ConfigParser()
import base64

config.read('mail.env')
APIKEY = config.get('API', 'APIKEY')
api =
sendgrid.SendGridAPIClient(APIKEY)
FROM_EMAIL =
config.get('API','FROM_EMAIL') def
sendemail(user,content):
   TO_EMAIL = user

   # if type == 'Account_creation':
   mail =
Mail(from_email=FROM_EMAIL,to_emails=TO_EMAIL,subject='H
ey there! We heard from
you!',html_content=f'<strong>{content}</strong>')     # if type ==
'complaint_creation':
   #     mail =
Mail(from_email=FROM_EMAIL,to_emails=TO_EMAIL,subject='Complai
nt
Created successfully',html_content='<strong>Compliant created
Successfully</strong
>')     response =
api.send(mail)
print(response.status_
code)
```

```
  print(response.headers)
```

Feature 2

```
{% extends 'base.html' %}

{%  block head %}
   <title>Features - Dream Team
Recruitment™</title>     <link
type="image/png" sizes="16x16" rel="icon"
href="static/img/job_logo.png">
   <link rel="stylesheet" href="static/css/bootstrap.min.css">
   <link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Raleway:300italic,400italic,
600italic,
700italic,800italic,400,300,600,700,800&amp;display=swap">
   {% endblock %}


   {% block content %}
   <script>
      window.watsonAssistantChatOptions = {          integrationID:
"a93c8bfc-cda0-49cd-a3d0-b68581f7ef06", // The ID of this
integration.          region: "eu-gb", // The region your integration is
hosted in.
        serviceInstanceID: "1ab40042-c8f9-455b-b469-fa7435694735",
// 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>
```

Coding:

App.py

```
import ibm_db
from flask import Flask, flash, redirect, render_template,
request, url_for from flask_mail import Mail, Message from
sendgrid import SendGridAPIClient
from sendgrid.helpers.mail import Mail

from sendmailer import *
import pandas as pd

app = Flask(__name__)



# app.secret_key="
# configure the mail settings
```

```
SENDGRID_API_KEY =
"SG.bnGBaY6cSGeU106QGq_H5Q.YhqfT29UYDRV9yWp3Rfn73LQy
kmE455
Zckt_qyJSR2U"
app.config['SECRET_KEY'] = 'top-secret!'
app.config['MAIL_SERVER'] =
'smtp.sendgrid.net'
app.config['MAIL_PORT'] = 587
app.config['MAIL_USE_TLS'] = True
app.config['MAIL_USERNAME'] = 'apikey'
#                app.config['MAIL_PASSWORD']                =
os.environ.get('SENDGRID_API_KEY')
#
app.config['MAIL_DEFAULT_SEN
DER'] =
os.environ.get('rrbg33@gmail.com')
mail = Mail(app)

conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=b0aebb68-
94fa-
46ec-a1fc-
1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud;PO
RT=3124
9;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;U
ID=pfd0
3782;PWD=dnhtwrcfkZlhkhAO","") # type: ignore
print(conn)
print("connection successful...")

@app.route('/sendgrid') def
sendgrid():    return
render_template('sendgrid.htm
l')



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

```
loginpage():     if request.method ==
'POST':         email =
request.form['email']         password =
request.form['password']         if not
email or not password:
          return render_template('login.html',error='Please fill all fields')


           query = "SELECT * FROM USERS WHERE EMAIL=? AND
PASSWORD=?"
       stmt = ibm_db.prepare(conn, query) #
type:ignore
ibm_db.bind_param(stmt,1,email) #
type:ignore
ibm_db.bind_param(stmt,2,password) #
type:ignore         ibm_db.execute(stmt) #
type:ignore         isUser =
ibm_db.fetch_assoc(stmt) # type:ignore
print(isUser,password)

       if not isUser:
          return render_template('login.html',error='Invalid Credentials')


       return redirect(url_for('home'))

   return render_template('login.html',name='Home')



@app.route('/signup') def
registration():     return
render_template('signup.htm
l')



@app.route('/signup',
methods=['POST','GET']) def signup():
   if request.method ==
'POST':         name =
```

```
request.form['name']
email =
request.form['email']
phone =
request.form['phone']
        password = request.form['password']

        sql ="INSERT INTO USERS VALUES
(?,?,?,?)"          stmt = ibm_db.prepare(conn,sql)
# type: ignore          ibm_db.bind_param(stmt, 1,
name)  # type: ignore
ibm_db.bind_param(stmt, 2, email)  # type:
ignore          ibm_db.bind_param(stmt, 3, phone)
# type: ignore          ibm_db.bind_param(stmt, 4,
password)  # type: ignore
ibm_db.execute(stmt)  # type: ignore
sendemail(email,")          return
redirect(url_for('home'))

    return render_template('signup.html')



@app.route('/')
@app.route('/login') def
login():     return
render_template('login.htm
l')




@app.route('/stats') def
stats():     return
render_template('stats.htm
l')
```

```
@app.route('/co
ntacts')        def
requester():
   return render_template('contacts.html')



@app.route('/tech',
methods=['POST'])              def
by_tech():
   jobs =
pd.read_csv('jobs.csv')
input = request.get_json()
tech_name =
input['techName']
      page = int(input['page']) if input['page']
else 0            sorting = input['sorting'] if
input['sorting'] else 0

       filtered_jobs = jobs.loc[jobs['Tech Stack'].str.contains(tech_name,
na=False)]

   if sorting:
      filtered_jobs = filtered_jobs.sort_values(by=[sorting])

         return filtered_jobs[page*10:page*10+10].drop(['Unnamed: 0'],
axis=1).to_json(orient='records')



@app.route('/location',
methods=['POST']) def by_location():
jobs = pd.read_csv('jobs.csv')     input
= request.get_json()     location =
input['location']     page =
int(input['page']) if input['page'] else 0
sorting = input['sorting'] if
input['sorting'] else 0
```

```
    filtered_jobs = jobs.loc[jobs['Location'] ==
location]     if sorting:
       filtered_jobs = filtered_jobs.sort_values(by=[sorting])

          return   filtered_jobs[page*10:page*10+10].drop(['Unnamed:   0'],
axis=1).to_json(orient='records')

@app.route('/
forgot')     def
reques():
   return render_template('forgotten-password.html')

@app.route('/forgot',methods=['PO
ST','GET']) def forgot():     if
request.method == 'POST':
email = request.form['email']
       query = "SELECT * FROM USERS WHERE
EMAIL=?"         stmt = ibm_db.prepare(conn,
query) # type:ignore
ibm_db.bind_param(stmt,1,email)
ibm_db.execute(stmt) # type:ignore
       isUser = ibm_db.fetch_assoc(stmt) # type:ignore
       # print(isUser,password)
        print(isUser)
print(stmt)
       sendemail(email,'We have recieved your email! from your email
address to reset the password, we will send you a link to reset your
password')         return render_template('login.html')
   return render_template('forgotten-password.html')

@app.route('/fe
atures')        def
features():
   return render_template('features.html')
```

```
@app.route('/home')      def
home():                  return
render_template('index.htm
l')

#
@app.r
oute('/')
#     def
home(
):
#     return render_template('index.html')

@app.route('/contacts'
,methods=['POST']) def
contacts():     email =
request.form['email']
       sendemail(email,'We  have  recieved  your
email!')                                  return
render_template('contacts.html')



@app.route('/l
ogout')      def
logout():

   session.pop('email', None) # type:ignore
   return redirect(url_for('login'))



if                __name__=='__main__':
app.run(debug=True)
```

Output:

connection successful...

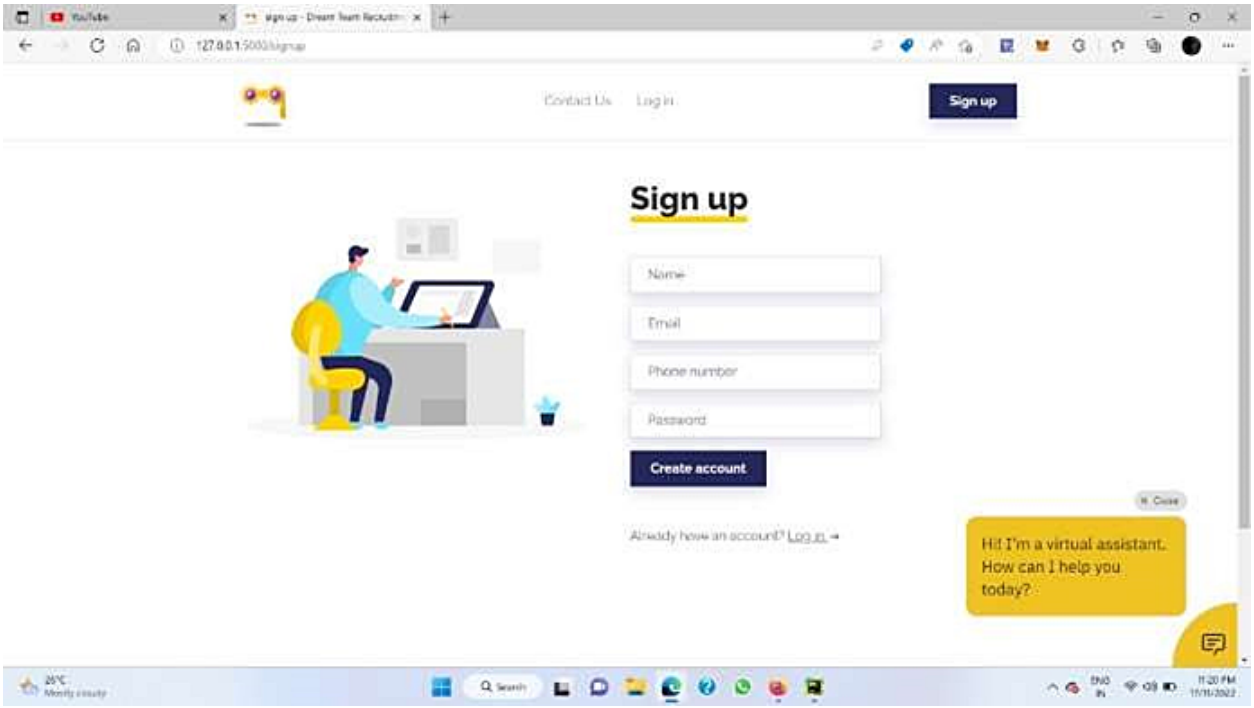
1. Serving Flask app 'app' (lazy loading)
2. Environment: production

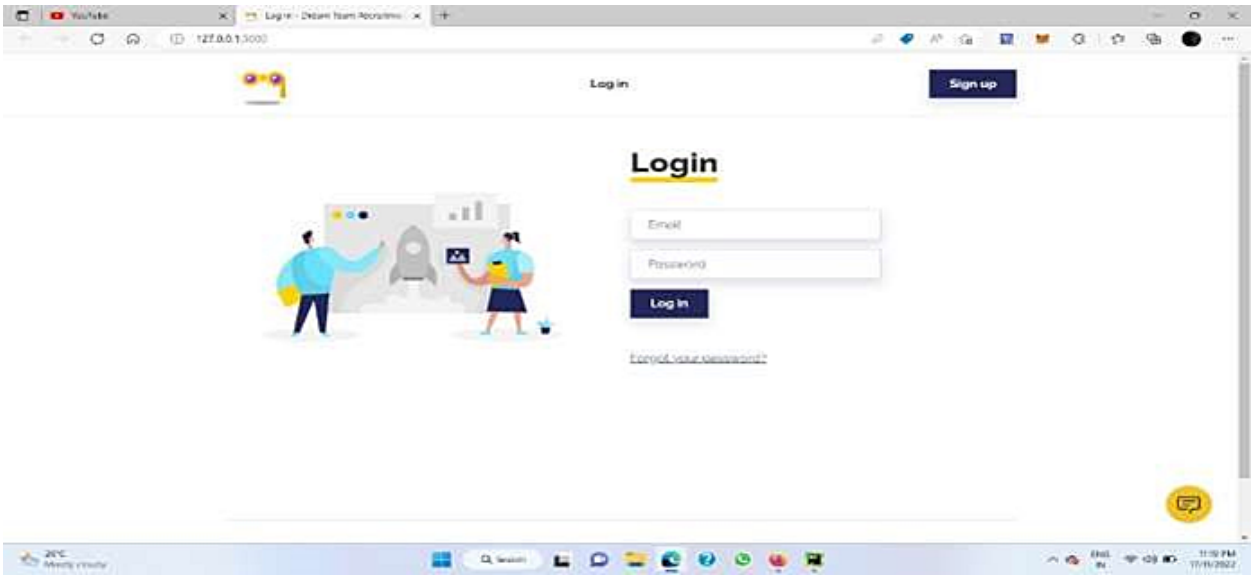
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.

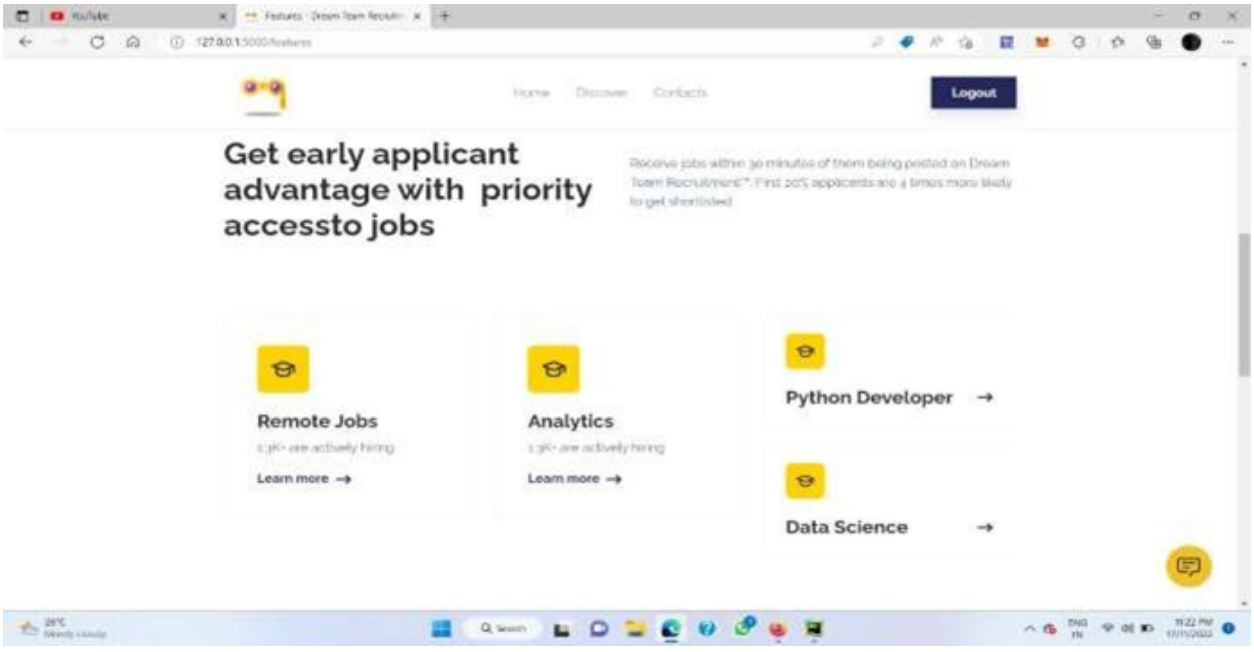
3. Debug mode: on

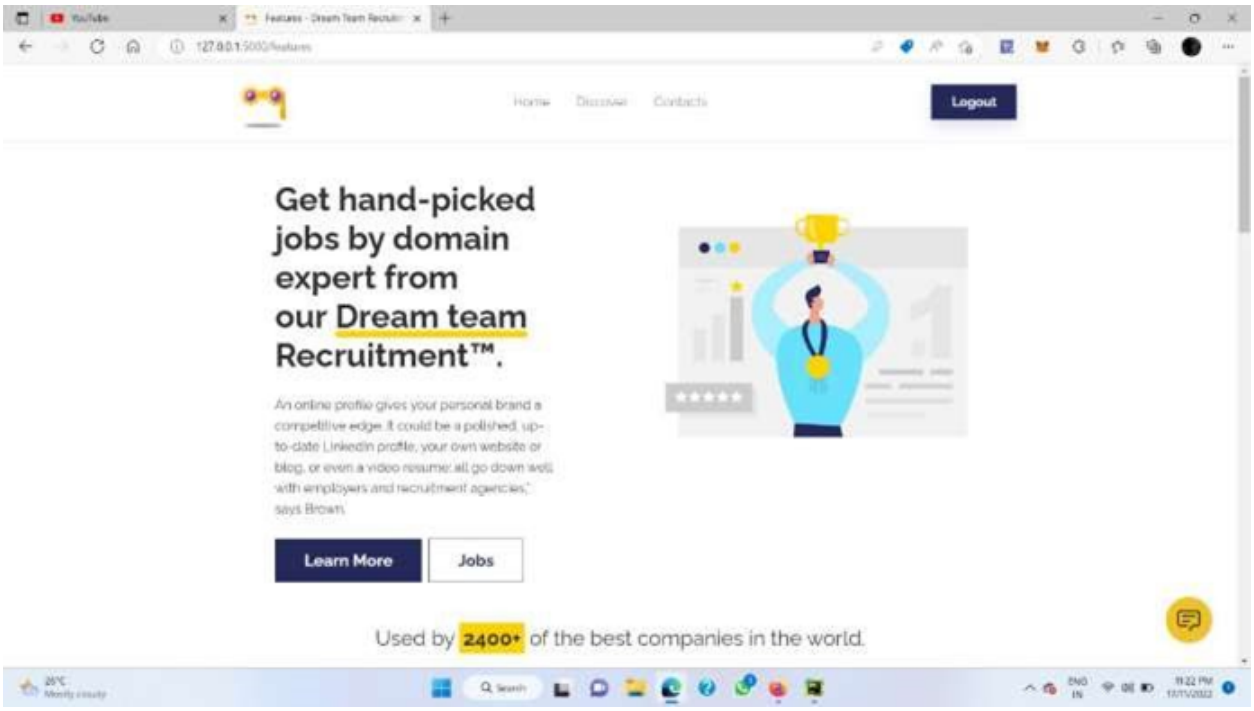
   WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
4. Running on http://127.0.0.1:5000

OUTPUT IMAGES:




Log in
Sign up
Login
Email
Password
Log in
Forgot your password?


Home
Discover
Contacts
Logout
Get early applicant advantage with priority accessto jobs
Remote Jobs
Learn more
Analytics
Learn more
Python Developer
Data Science


Home
Discover
Contacts
Logout
Get hand-picked jobs by domain expert from our Dream team Recruitment™.
An online profile gives your personal brand a competitive edge. It could be a polished, up-to-date LinkedIn profile, your own website or blog, or even a video resume; all go down well with employers and recruitment agencies,' says Brown.
Learn More
Jobs
Used by 2400+ of the best companies in the world.

chapter 8
TESTING

Software testing is used to assess the quality of the product. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation.

8.1 Test Cases:

Testcase1: Does the flask application is perfectly created and in works in very good condition?

Testcase2: Does the Send-Grid integration is working correctly?

Testcase3: Does the db2 is perfectly connected to the application?**Testcase4:** Can the chat-bot which is created using Watson assistant is recommending correctly the job for the end users?

Testcase 5: Whether the application is working correctly without any interruptions?

8.2 User Acceptance Testing:

User acceptance testing is a type of testing that is used to determine whether or not a software system is suitable for use by end users. It is the process of verifying that a system meets the requirements of the user and that the user is able to use the system for its intended purpose. User acceptance testing (UAT) is a process of verifying that a system meets the needs of the end users and that they are able to use it. This can be done through a variety of methods, such as

interviews, surveys, or observation. UAT is important in water quality analysis and prediction because it helps ensure that the system being developed will be useful to those who will be using it. By testing with act users, developers can get feedback on the system and make sure it is meeting the needs of the users.

Chapter 9
RESULTS

9.1 Performance Metrics

Implementation of web application:

To create the web application to interact with the users. The users here is commonly job seeker and job provider. Login, Signup, Job searching have separate pages where we can access into different work functions.

SendGrid Integration:

The flask application that we created is to get integrated with sendgrid which provide the e-mail interface for communication purpose.

Developing chatbot:

To develop a chat-bot so that, that can be very interactive to the users who are using the application and to recommend the jobs based on the job seekers interests.

Deployment of Application:

Finally the developed application is to deployed in the cloud.

1 .Accuracy

The accuracy metric is one of the simplest Classification metrics to implement, and it can be determined as the number of correct predictions to the total number of predictions.

To implement an accuracy metric, we can compare ground truth and predicted value in a loop.

Chapter 10
ADVANTAGES AND DISADVANTAGES

Advantages:

1. The main advantage of our application is that there is a direct one way communication between the job seeker and the job recruiter.
2. There is a chat-bot which gives the directions to the users what to do and not to do and also it recommends the jobs based on the job seekers interests.
3. The application is an open source one which doesn’t asks for the money.

Disadvantage:

4. One disadvantage of the application is that it is not a full paced one.
5. Another disadvantage of skill / job recommender application is that it is unaware of machine language stack. No AI is implemented here.
6. Skill / Job Recommender Application is used in many domains despite of professionals.

Chapter 11
CONCLUSION

In this work, we have presented our proposal for the automatic recommendation of job offers. Our goal here is being able to build methods being able to deliver appropriate job offers to those job seekers that could be potentially interested on them. To do that, we have based our research efforts on two wellknown classification methods: random forests (RF) and support vector machines (SVM). Our empirical evaluation shows us interesting facts. For example, RF are more likely to be interpreted although they do no present a particularly good performance in relation to SVM. On the other hand, SVM are more accurate, although they work with a model being much harder to interpret by human. What it is clear is, that in both cases, we have shown that these two methods are quite

appropriate for accurately working in the context of automatic job recommendation.

There is unemployment only because of lack of skill set in their

domain or fear of missing out on a job. When a job seeker is afraid of getting the desired job, he might losses the job which he really deserved for it. The main purpose of the job recommender application is to provide job opportunities for each and every single person. The only thing the job seeker wants to do is just to approach the application and apply for the job. He will be provided with the login credentials with the confirmation email. There he can find numerous job opportunities. He will be guided with the in-built chat bot, which guides the job seekers to apply for the job and recommends the availability of jobs based on their interest. The chat bot is built with IBM Watson Assistant that is very much helpful in collecting the job seeker interests and also guides them to apply for it.

Chapter 12

FUTURE SCOPE

As future work, we propose to design novel computational methods being able to process the textual description from the job offers. At that point, we were using just the quantitative information that is advertised. However, we think that the way an offer is written can help attracting potential candidates as well, maybe new methods for natural language processing using neural networks could help in this task. We also would like to explore the possibilities to work with expert knowledge via kernel mapping in the case of SVM as we mentioned earlier. Finally, it is also necessary to study how to integrate this technology with existing web information systems so that these two methods can be put into operation by the industry.

chapter 13

Github link: view link

Demo Video Link:
https://drive.google.com/file/d/1pIBG16PN9EDz1Dh7OBrlkwSKphMVcHZY/view?usp=share_link