IBM – NALAIYA THIRAN PROJECT

SKILL AND JOB RECOMMENDER APPLICATION

INDUSTRY MENTOR : KRISHNA CHAITANYA

FACULTY MENTOR : SUJARITHA M

TEAM ID : PNT2022TMID02825

TEAM LEAD : SUDHARSAN H

TEAM MEMBER : SURYA SIDHESHVAR R

TEAM MEMBER : VIGNESH A

TEAM MEMBER : SRI VARSHAN M

TABLE OF CONTENT

CHAPTER 1
INTRODUCTION

1.1 Project Overview

Skill and Job Recommender web application is capable of displaying the current job openings based on the skillset of the users. Having lots of Skills but wondering which job will suits you ? Don't need to worry we had to come up with a skill recommender solution throught which the freshers or the skilled person can login and find the jobs by using search option or they can directly interact with the chatbot and get their dream job. To develop an end to end web application capable of displaying the current jobs openings based on the skillset of the users. The users and their information are stored in the Database.An alert is sent when there is a opening based on the user skillset. User will interact with the chatbot and can get the recommendations based on his skills.We can use job search API to get current job openings in the market which will fetch the directly from the webpage.

1.2 Purpose

Having lots of Skills but wondering which job will suits you ? Don't need to worry we had to come up with a skill recommender solution throught which the freshers or the skilled person can login and find the jobs by using search option or they can directly interact with the chatbot and get their dream job. To develop an end to end web application capable of displaying the current jobs openings based on the skillset of the users. The users and their information are stored in the Database.An alert is sent when there is a opening based on the user skillset. User will interact with the chatbot and can get the recommendations based on his skills.We can use job search API to get current job openings in the market which will fetch the directly from the webpage.

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING PROBLEM

Job recommender systems have become popular since they successfully reduce information overload by generating personal-ized 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 recommending job vacancies that fit properly to the job seekers 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; and 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.

2.2 REFERENCE

1) AlO12] Shaha T Al-Otaibi and Mourad Ykhlef. “A survey of job recommender systems”. In: International

2) Journal of the Physical Sciences 7.29 (2012), pp. 5127–5142. issn: 19921950. doi: 10.5897/IJPS12.

482.

3) [Den15] 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 207 (2015), pp. 369–376.

4) [Dia13] M Diaby, E Viennet, and T Launay. “Toward the next generation of recruitment tools: An online social network-based job recommender system”. In: Proc. of the 2013 IEEE/ACM Int. Conf. on

5) Advances in Social Networks Analysis and Mining, ASONAM 2013 (2013), pp. 821–828.

2.3 PROJECT STATEMENT DEFINITION

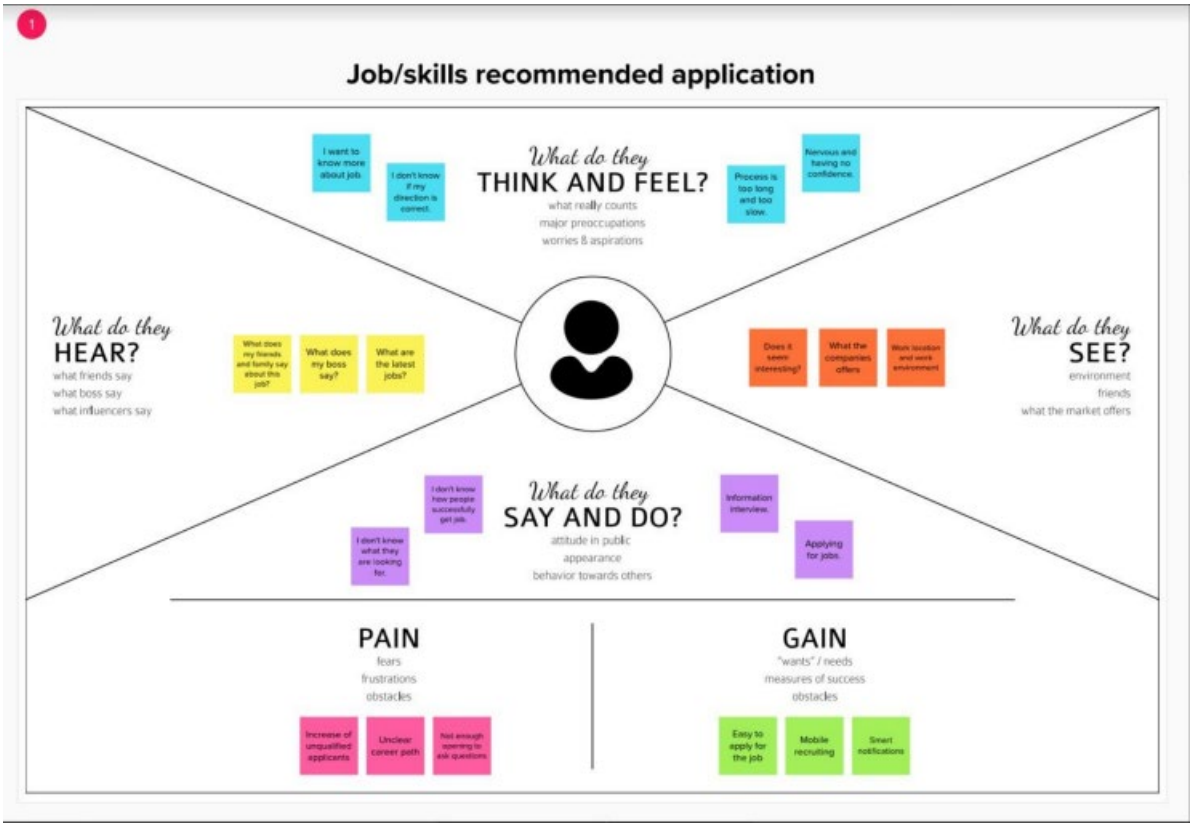
Having lots of skills but wondering which job will best suit you? Don’t need to worry! We have come up with a skill recommender solution through which the fresher or the skilled person can log in and find the jobs by using the search option or they can directly interact with the chatbot and get their dream job.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.

CHAPETER – 3

IDEATION AND PROPOSED SOLUTION

3.1 Empathy Map Canvas

Empathy Map Canvas: An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes. It is a useful tool to helps teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.



3.2 Ideation & Brainstorming

Brainstorming is a problem-solving activity where students build on or develop higher order thinking skills. **Encourages creative thought**. Brainstorming encourages students to think creatively (out of the box), encouraging all students to share their ideas, no matter how far "out there" they may seem.

3.3 Proposed Solution

S. No.	Parameter	Description
1.	Problem Statement (Problem to besolved)	Having lots of skills but wondering whichjob will best suit you? Don't need to worry! We have come up with a skill recommender solution through which thefresher or the skilled person can log in and find the jobs by using the search option or they can directly interact with the chatbot and get their dream job. To develop an end-to-end web applicationcapable 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 canget the recommendations based on their skills. We can use a job search API to get the current job openings in the market which will fetch the data directly from the webpage.

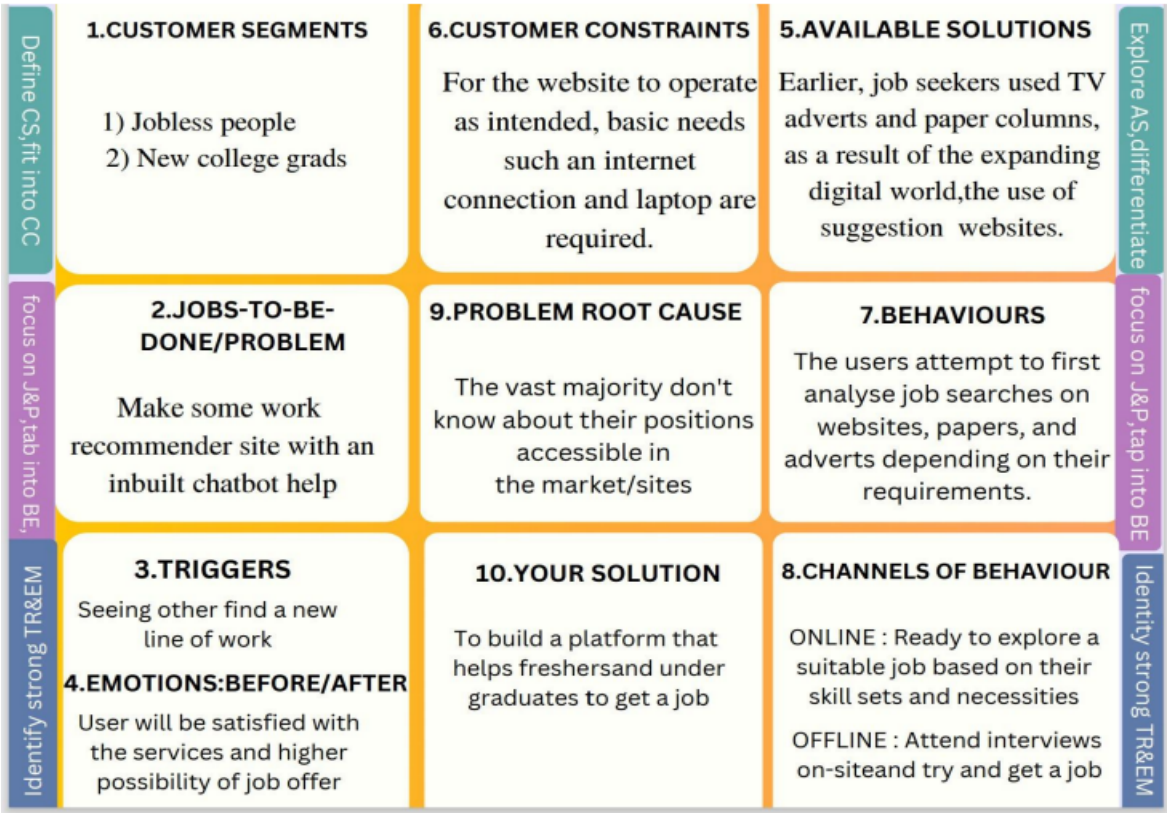
2.	Idea / Solution description	The contributions of this work are threefold, we: i) made publicly available a new datasetformed by a set of job seekers profiles and aset of job vacancies collected from different job search engine sites ii) put forward the proposal of a framework for job recommendation based on professional skills of job seekers iii) carried out an evaluation to quantify recommendation abilities of two state-of-the art methods, considering different configurations, within the proposed framework. We thus present ageneral panorama of job recommendation task aiming to facilitate research and real- world application design regarding this important issue.
3.	Novelty / Uniqueness	The best position are suggested to any person according to her skills. While the position of known profiles are assumed
		should be noted that there are usually multiple advisable positions corresponding to a set of skills.A recommendation system should return a set of most likely positions and all of them can be equally valid. The recommendation method we use is simply based on representing both positions and profiles as comparable vectors and seeking for each profile the positions with the most similar vectors.
4.	Social Impact / Customer Satisfaction	Students will be benefited as they will get to know which job suits them based on their skill set and therefore Lack of Unemployment can be reduced.
5.	Business Model (Revenue Model)	We can provide the application for job seekers in a subscription based and we can share the profiles with companies and generate the revenue by providing them best profiles.
6.	Scalability of the Solution	Data can be scaled up and scaled down according to number of current job openings available.

3.4 Problem Solution fit

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

Purpose:

1) Solve complex problems in a way that fits the state of your customers.
2) Succeed faster and increase your solution adoption by tapping into existing mediums and channels of behavior.
3) Sharpen your communication and marketing strategy with the right triggers and messaging.
4) Increase touch-points with your company by finding the right problem-behavior fit and building trust by solving frequent annoyances, or urgent or costly problems.
5) Understand the existing situation in order to improve it for your target group



CHAPTER – 4
REQUIREMENT ANALYSIS

4.1 Functional requirement

S. No	FUNCTIONAL REQUIREMENT (Epic)	SUB REQUIREMENT (Story)
1)	Sign In / Login	Register with username, password
2)	Profile Registration	Register with username, password, email, qualification, skills. This data will be stored in a database.
3)	Job profile display	Display job profiles based on availability, location ,skills
4)	Chatbot	A chat on the webpage to solve user queries and issues
5)	Job registration	A copy of the company the user applied for with its registration/description details willbe sent to the registered email id.
6)	Logout	

4.2 Non-Functional requirements

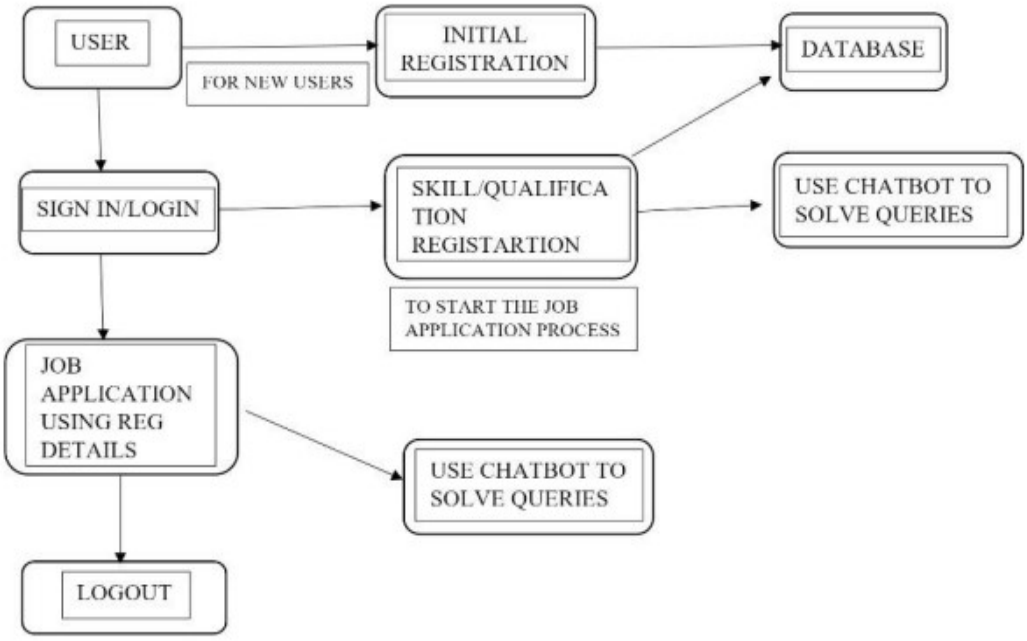
S. No	NON-FUNTIONAL REQUIREMENT	DESCRIPTION
1)	Usability	The webpage will be designed in such away that any non-technical user can easily navigate through it and completethe job registration work. (Easy and Simple design.)
2)	Security	Using of SSL certificate (Python Flaskto Cloud connect) will provide securityto the project. Database will be safely stored in DB2.
3)	Reliability	To make sure the webpage doesn't go down due to network traffic.
4)	Availability	This webpage will be available to all users (network connectivity is necessary) at any given point of time.
5)	Scalability	Increasing the storage space of databasecan increase the number of users. Add some features in future to make the webpage unique and attractive.
6)	Performance	Focus on loading the webpage as quickly as possible irrespective of thenumber of user/integrator traffic.

CHAPTER 5

PROJECT DESIGN

5.1 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

FUNTIONAL AND NON-FUNCTIONAL REQUIREMNTS:
Following are the functional requirements of the proposed solution

S. No	FUNCTIONAL REQUIREMENT (Epic)	SUB REQUIREMENT (Story)
1)	Sign In / Login	Register with username, password
2)	Profile Registration	Register with username, password, email, qualification, skills. This data will be stored in a database.
3)	Job profile display	Display job profiles based on availability, location ,skills
4)	Chatbot	A chat on the webpage to solve user queries and issues
5)	Job registration	A copy of the company the user applied for with its registration/description details willbe sent to the registered email id.

Following are the non-functional requirements of the proposed solution

S. No	NON-FUNTIONAL REQUIREMENT	DESCRIPTION
1)	Usability	The webpage will be designed in such away that any non-technical user can easily navigate through it and complete the job registration work. (Easy and Simple design.)
2)	Security	Using of SSL certificate (Python Flask to Cloud connect) will provide security to the project. Database will be safely stored in DB2.
3)	Reliability	To make sure the webpage doesn't go down due to network traffic.

5)	Availability	This webpage will be available to all users (network connectivity is necessary) at any given point of time.
6)	Scalability	Increasing the storage space of databasecan increase the number of users. Add some features in future to make the webpage unique and attractive.
4)	Performan ce	Focus on loading the webpage as quickly as possible irrespective of the number of user/integrator traffic.

5.3 USER STORY

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	I can receive confirmation email & click confirm	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	I can access my account / dashboard	High	Sprint-1
	Dashboard	USN-6	Create a model set that contains those models, then assign it to a role.	Assign that group to the appropriate roles on the Roles page	High	Sprint-1
Customer (Web user)	Identity-Aware	USN-7	Open, public access, User-authenticated access, Employee-restricted access.	Company public website. App running on the company intranet. App with access to customer private information.	High	Sprint-1
Customer Care Executive	Communication	USN-8	A customer care executive is a professional responsible for communicating the how's and why's regarding service expectations within a company.	For how to tackle customer queries.	Medium	Sprint-1
Administrator	Device management	USN-9	You can Delete/Disable/Enable devices in Azure Active Directory but you cannot Add/Remove Users in the directory.	Ease of use.	Medium	Sprint-1

CHAPTER 6

PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION

Sprints are the backbone of any good Agile development team. And the better prepared you are before a sprint, the more likely you are to hit your goals. Spring planning helps to refocus attention, minimize surprises, and (hopefully) guarantee better code gets shipped.

But maybe more than that, sprint planning aligns the development team with the product owner. Like any relationship, the one between you and your team requires communication and clarity. And taking the time to sit down and make sure that your expectations are understood and *can* be done by your team is key to keeping everyone motivated and productive.

Sprint planning comes down to a few key steps, from making sure your product backlog is properly groomed to framing the sprint, and running an effective sprint planning meeting. In this guide, we'll run you through everything you need to know (plus give you a few additional resources to help you through your own sprint planning session).

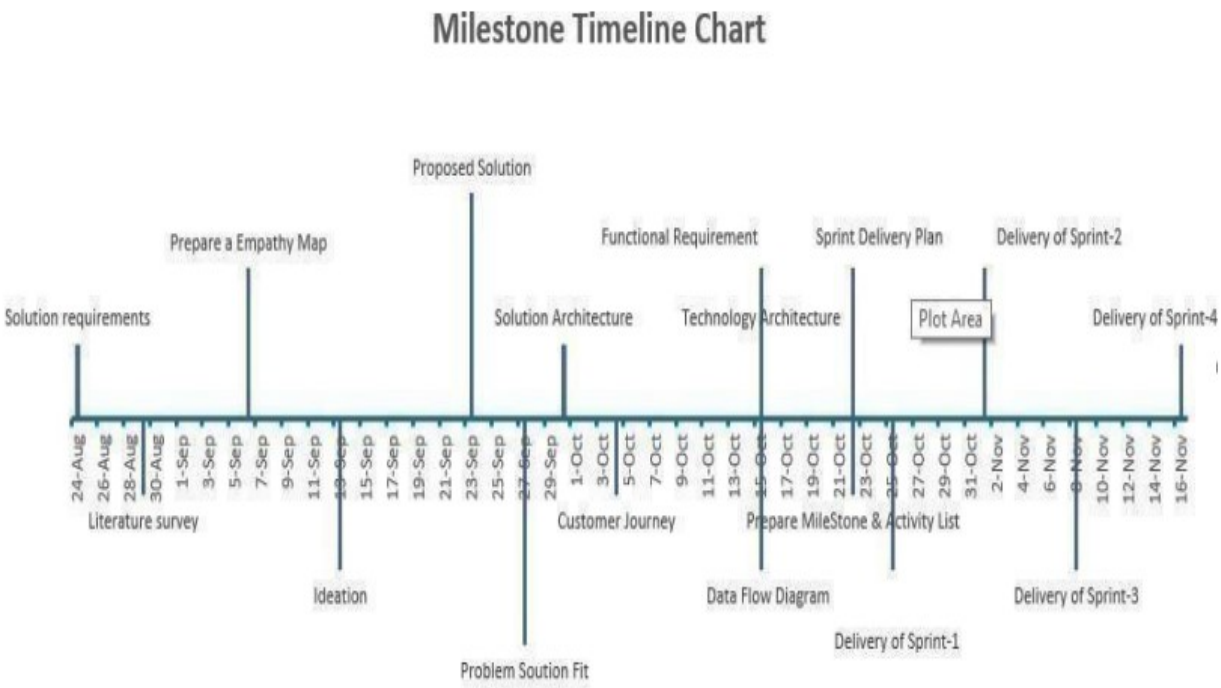
6.2 SPRINT DELIVERY SCHEDULE

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Team Members
Sprint-1	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	Sudharsan Sri varshan
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	Surya vignesh
Sprint-2		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sudharsan Sri varshan
Sprint-3		USN-4	As a user, I can register for the application through Gmail	I can receive confirmation email & click confirm	Medium	Surya vignesh
Sprint-2	Login	USN-5	As a user, I can log into the application by entering email & password	I can access my account / dashboard	High	Sudharsan Sri varshan
Sprint-2	Dashboard	USN-6	Create a model set that contains those models, then assign it to a role.	Assign that group to the appropriate roles on the Roles page	High	Surya vignesh
Sprint-4	Identity-Aware	USN-7	Open, public access, User-authenticated access, Employee-restricted access.	Company public website. App running on the company intranet. App with access to customer private information.	High	Sudharsan Sri varshan
Sprint-1	Communication	USN-8	A customer care executive is a professional responsible for communicating the how's and why's regarding service expectations within a company.	For how to tackle customer queries.	Medium	Surya vignesh
Sprint-3	Device management	USN-9	You can Delete/Disable/Enable devices in Azure Active Directory but you cannot Add/Remove Users in the directory.	Ease of use.	Medium	Sudharsan Sri varshan

6.3 REPORTS FROM JIRA

When project begins then it is expected that project related activities must be initiated. In project planning, series of milestones must be established. Milestone can be defined as recognizable endpoint of software project activity. At each milestone, report must be generated.

Milestone is distinct and logical stage of the project. It is used as signal post for project start and end date, need for external review or input and for checking budget, submission of the deliverable, etc. It simply represents clear sequence of events that are incrementally developed or build until project gets successfully completed. It is generally referred to as task with zero-time duration because they are used to symbolize an achievement or point of time in project. It helps in signifying change or stage in development.



CHAPTER 7

CODING AND SOLUTIONING

7.1 FEATURE 1

```
from flask import Flask,render_template,request, redirect,url_for
import ibm_db
app = Flask(__name__)


app.secret_key='vy@ur434'
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=19af6446-6171-4641-8aba-9dcff8e1b6ff.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=30699;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=mrv82327;PWD=lUMrTlKQrEqCFg7b",",")
print(conn)
print("connection successful...")


@app.route('/')
def home():
    return render_template('index.html')


@app.route('/contacts')
def contacts():
    return render_template('contacts.html')


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



```
@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)
        #ibm_db.bind_param(stmt, 1, name)
        #ibm_db.bind_param(stmt, 2, email)
        #ibm_db.bind_param(stmt, 3, phone)
        #ibm_db.bind_param(stmt, 4, password)
        #ibm_db.execute(stmt)
    return render_template('signup.html')

@app.route('/login', methods=['POST','GET'])
def login():
    #if request.method == 'POST':
        #email = request.form['email']
        #password = request.form['password']



        #sql = "SELECT * FROM users WHERE email=%s AND password=%s"
        #stmt = ibm_db.prepare(conn, sql)
        #ibm_db.bind_param(stmt,1,email)
        #ibm_db.bind_param(stmt,2,password)
       # user = ibm_db.execute(stmt).fetchone()
    return render_template('login.html' ,msg="success")
```



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

7.2 FEATURE 2

```
from flask import Flask, request, render_template, redirect, url_for, session
import ibm_db
import re
import json
import requests
# from sendgrid import SendGridAPIClient
# from sendgrid.helpers.mail import Mail
import os

app = Flask(__name__,template_folder='template',static_folder='template/static')
app.config['SESSION_TYPE'] = 'memcached'
app.config['SECRET_KEY'] = 'super secret key'

session_username = ""

hostname = ''
uid = ''
pwd = ''
driver = "{}"
db = ''
port = ''
protocol = ''
cert = ''

dsn = (
        "DATABASE = {0};"
        "HOSTNAME = {1};"
        "PORT = {2};"
```

```
        "UID = {3};"
        "SECURITY = SSL;"
        "SSlServerCertificate = {4};"
        "PWD = {5};"
).format(db, hostname, port, uid, cert, pwd)



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



@app.route("/")
def home_page():
        return render_template('home.html')



@app.route("/login", methods=["GET", "POST"])
def login_page():
        global userid
        msg_value = ""
        msg = ""
        if request.method == 'POST':
                username = request.form["username"]
                password = request.form["password"]
                # username = request.form.get("username")
                # password = request.form.get("password")
                print(username)
                sql = "SELECT username,email,password from user where username = ? and password = ? "
                stmt = ibm_db.prepare(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']
                # msg_value = "Logged in successfully"
                global session_username
                session_username = username
                return render_template('dashboard_2.html', msg_value=msg_value)
            else:
                msg = "Incorrect Username or Password"
                return render_template('login.html', msg=msg)
    return render_template('login.html')


@app.route("/register", methods=["GET", "POST"])
def register_page():
    msg = ''
    if request.method == 'POST':
        username = request.form["username"]
        password = request.form["password"]
        email = request.form["email"]
        sql = "SELECT * from user where username = ?"
        stmt = ibm_db.prepare(conn,sql)
        ibm_db.bind_param(stmt,1,username)
        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]',username):
                msg = "Name must contain only characters and numbers!"
            else:
                insert_sql = "INSERT INTO user (USERNAME,EMAIL,PASSWORD) values (?,?,?)"
                prep_stmt = ibm_db.prepare(conn, insert_sql)
                ibm_db.bind_param(prep_stmt,1,username)
                ibm_db.bind_param(prep_stmt,2,email)
                ibm_db.bind_param(prep_stmt,3,password)
                ibm_db.execute(prep_stmt)
                msg='You have successfully registered !'
                # return render_template('login.html')
        elif request.method == 'POST':
            msg = 'Please fill out the form!'
        return render_template('register.html', msg=msg)


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

            skillname = request.form["skillname"]

            url = "https://jsearch.p.rapidapi.com/search"
```



```
            querystring = {"query":f"{skillname}","num_pages":"1"}
```

```
        headers = {
        "X-RapidAPI-Key": "",
        "X-RapidAPI-Host": "jsearch.p.rapidapi.com"
        }

        response = requests.request("GET", url, headers=headers, params=querystring)

        # print(response.text)
        json_value = response.text



        json_value = json.loads(json_value,strict=False)

        jobs_data = json_value["data"]

        msg = """""""
        i = 0

        for i in range(len(jobs_data)):
                job_title = json_value["data"][int(i)]["job_title"]
                employer_website = json_value["data"][i]["employer_website"]
                msg = msg + f"""
                        <div class="col-lg-3 ml-5" id="focus-third">
                        <div class="card" style="width: 20rem;">
                                <!-- <img src="assets/img/health camp.jpg" class="card-img-
top" alt="..."> -->
                                <div class="card-body" style="border: 1px solid #000000">
                                <h5 class="card-title">{job_title}</h5>
                                <p class="card-text">{employer_website}</p>
```

```
                                        <a
href="apply?job_title={job_title}&employer_website={employer_website}" class="btn btn-
primary">Apply Now</a>
                                    </div>
                                </div>
                                </div>
                                <br>
                                <div class="space"></div>
                            """


                print(len(jobs_data))
                return render_template('dashboard_2.html',msg=msg)
        else:
                return render_template('dashboard_2.html')



@app.route('/apply', methods=['GET','POST'])
def apply():
        msg_value = ''
        if request.method == 'POST':
                username = request.form["username"]
                email = request.form["email"]
                qualification = request.form["qualification"]
                skills = request.form["skills"]
                job_title = request.form["job_title"]
                employer_website = request.form["employer_website"]
                sql = 'SELECT * FROM user where username =?'
                insert_sql = "INSERT INTO jobs
(USERNAME,EMAIL,QUALIFICATION,SKILLS,JOBTITLE,COMPANY) VALUES
(?,?,?,?,?,?)"
                prep_stmt = ibm_db.prepare(conn, insert_sql)
                ibm_db.bind_param(prep_stmt,1,username)
```

```
            ibm_db.bind_param(prep_stmt,2,email)
            ibm_db.bind_param(prep_stmt,3,qualification)
            ibm_db.bind_param(prep_stmt,4,skills)
            ibm_db.bind_param(prep_stmt,5,job_title)
            ibm_db.bind_param(prep_stmt,6,employer_website)
            ibm_db.execute(prep_stmt)
            msg_value = 'You have successfully applied for job !'
            session['loggedin']=True
        elif request.method == ['POST']:
            msg_value='Please fill out the form !'
            return render_template("apply_2.html", remaining_form=remaining_form)
        else:
            print("out")
            get_job_title = request.args.get('job_title', default = "Developer", type = str)
            get_employer_website = request.args.get('employer_website', default = 'Company Name Not Provided', type = str)
            if get_employer_website == None:
                get_employer_website = 'Company Name Not Provided'
            # print(get_job_title)
            # print(get_employer_website)
            remaining_form = f"""
                <input type="text" name="job_title" id="job_title" value="{get_job_title}" required></br></br>
                <input type="text" name="employer_website" id="employer_website" value="{get_employer_website}" required></br></br>
            """
            return render_template("apply_2.html", remaining_form=remaining_form)
        return render_template('dashboard_2.html',msg_value=msg_value)
```



```
@app.route('/display')
```

```
def display():
    # print(session["username"],session['id'])
    # cursor = mysql.connetion.cursor()
    # cursor.execute('SELECT * FROM job where userid = %s', (session['id']))
    # account = cursor.fetchone()
    # print("accountdisplay",account)
    jobs_list_sql = "SELECT * FROM jobs where username = ? "
    prep_stmt = ibm_db.prepare(conn, jobs_list_sql)
    print(session_username)
    ibm_db.bind_param(prep_stmt,1,session_username)
    ibm_db.execute(prep_stmt)
    list_of_jobs = ibm_db.fetch_assoc(prep_stmt)
    jobs_table = """""""
    while list_of_jobs != False:
        # print("The ID is : " + str(list_of_jobs["JOBTITLE"]))
        # print("The name is : ", str(list_of_jobs["COMPANY"]))
        jobs_name_value = str(list_of_jobs["JOBTITLE"])
        company_name_value = str(list_of_jobs["COMPANY"])
        skills_name_value = str(list_of_jobs["SKILLS"])
        list_of_jobs = ibm_db.fetch_assoc(prep_stmt)
        jobs_table = jobs_table + f"""
            <tr>
                <td>{jobs_name_value}</td>
                <td>{company_name_value} Chang</td>
                <td>{skills_name_value}</td>
            </tr>
        """
    print(list_of_jobs)
    return render_template('display.html', jobs_table=jobs_table)
```

CHAPTER 8

TESTING

8.1 TEST CASES

A test case is a set of actions performed on a system to determine if it satisfies software requirements and functions correctly. The purpose of a test case is to determine if different features within a system are performing as expected and to confirm that the system satisfies all related standards, guidelines and customer requirements. The process of writing a test case can also help reveal errors or defects within the system.

Test cases are typically written by members of the quality assurance (QA) team or the testing team and can be used as step-by-step instructions for each system test. Testing begins once the development team has finished a system feature or set of features. A sequence or collection of test cases is called a test suite.

A test case document includes test steps, test data, preconditions and the postconditions that verify requirements.

Types of test cases

- Functionality Test Case
- User Interface Test Case
- Performance Test Case
- Integration Test Case
- Usability Test Case
- Database Test Case
- Security Test Case
- User Acceptance Test Case

7.2 USER ACCEPTANCE TESTING

Acceptance testing can take many forms, such as user acceptance testing, operational acceptance testing, contract acceptance testing and others. In this article, the focus is on user acceptance testing.

When people ask, “What is UAT testing?” One commonly cited definition of user acceptance testing is:

“Formal testing with respect to user needs, requirements, and business processes conducted to determine whether or not a system satisfies the acceptance criteria and to enable the user, customers or other authorized entity to determine whether or not to accept the system.” (ISTQB Glossary V3.1)

There are other definitions as well, which highlight the need of understanding the context of user acceptance testing.

For example, in agile methods “acceptance testing” is defined as:

“An acceptance test is a formal description of the behavior of a software product, generally expressed as an example or a usage scenario…..Teams mature in their practice of agile use acceptance tests as the main form of functional specification and the only formal expression of business requirements. Other teams use acceptance tests as a complement to specification documents containing uses cases or more narrative text.” (Agile Alliance)

In addition, the Agile Alliance also adds:

“The terms “functional test”, “acceptance test” and “customer test” are used more or less interchangeably. A more specific term “story test”, referring to user stories is also used, as in the phrase “story test driven development”.

From these differences, we can see that traditional acceptance testing is seen in a validation context while agile methods tend to view acceptance testing as verification. In the agile context, acceptance criteria are basically described along with a user story to show specific cases that should be tested to show the user story has been implemented correctly and are basically the same as functional tests.

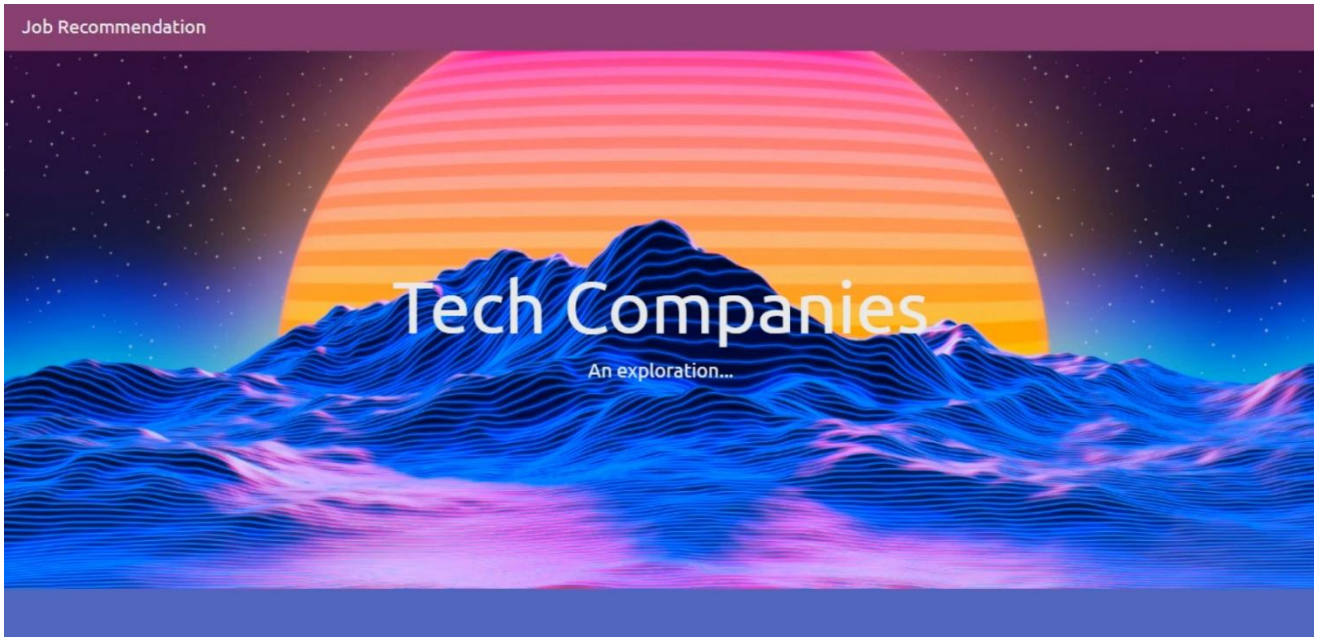
I often say that user acceptance testing is one of the most valuable levels of testing, but often performed at the worst possible time. That is because if process gaps or other major flaws are discovered in UAT testing, there is little time to fix them before release. Also, the options available to fix late-stage defects may be very limited at the end of a project.

CHAPTER 9

RESULTS

9.1 PERFORMANCE METRICS

Front page:



User search view:

Job recommending page:

CHAPTER 10

ADVANTAGES AND DISADVANTAGES

ADVANTAGES

- **Revenue** — With years of research, experiments and execution primarily driven by Amazon, not only is there less of a learning curve for online customers today
- **Customer Satisfaction** — Many a time customers tend to look at their product recommendation from their last browsing. Mainly because they think they will find better opportunities for good products.
- **Personalization** — The user will be placed in a better mood to purchase your products or serviceS.
- **Discovery** — People generally like to be suggested things which they would like, and when they use a site which can relate to his/her choices extremely perfectly then he/she is bound to visit that site again.
- **Provide Reports** — Giving the client accurate and up to the minute reporting allows him to make solid decisions about his site and the direction of a campaign.

DISADVANTAGES

- Significant investments required
- Too many choices
- The complex on boarding process
- Lack of data analytics capability
- The ‘cold start’ problem
- Inability to capture changes in use behaviour
- Privacy concerns

CHAPTER 11

CONCLUSION

The proposed skill and job recommendation application has been implemented in the visual studio for the web development. The tasks involved in work are dived into modules. The data is approached and shared by visual studio. The proposed system is efficient and user friendly.

CHAPTER 12

FUTURE SCOPE

- Skill and Job Recommender web application is capable of displaying the current job openings based on the skillset of the users.
- Don’t need to worry we had to come up with a skill recommender solution throught which the freshers or the skilled person can login and find the jobs by using search option or they can directly interact with the chatbot and get their dream job.
- To develop an end to end web application capable of displaying the current jobs openings based on the skillset of the users.
- The users and their information are stored in the Database.An alert is sent when there is a opening based on the user skillset.
- User will interact with the chatbot and can get the recommendations based on his skills.
- We can use job search API to get current job openings in the market which will fetch the directly from the webpage.

CHAPTER 13

APPENDIX

https://github.com/IBM-EPBL/IBM-Project-17320-1659634088

SOURCE CODE

```
/* @extend display-flex; */
display-flex, .display-flex, .display-flex-center, .signup-content, .signin-content, .social-login, .socials {
  display: flex;
  display: -webkit-flex; }

/* @extend list-type-ulli; */
list-type-ulli, .socials {
  list-style-type: none;
  margin: 0;
  padding: 0; }

/* poppins-300 - latin */
@font-face {
  font-family: 'Poppins';
  font-style: normal;
  font-weight: 300;
  src: url("../fonts/poppins/poppins-v5-latin-300.eot");
  /* IE9 Compat Modes */
  src: local("Poppins Light"), local("Poppins-Light"), url("../fonts/poppins/poppins-v5-latin-300.eot?#iefix") format("embedded-opentype"), url("../fonts/poppins/poppins-v5-latin-300.woff2") format("woff2"), url("../fonts/poppins/poppins-v5-latin-300.woff") format("woff"), url("../fonts/poppins/poppins-v5-latin-300.ttf") format("truetype"), url("../fonts/poppins/poppins-v5-latin-300.svg#Poppins") format("svg");
```

```
  /* Legacy iOS */ }
/* poppins-300italic - latin */
@font-face {
  font-family: 'Poppins';
  font-style: italic;
  font-weight: 300;
  src: url("../fonts/poppins/poppins-v5-latin-300italic.eot");
  /* IE9 Compat Modes */
  src: local("Poppins Light Italic"), local("Poppins-LightItalic"), url("../fonts/poppins/poppins-v5-latin-300italic.eot?#iefix") format("embedded-opentype"), url("../fonts/poppins/poppins-v5-latin-300italic.woff2") format("woff2"), url("../fonts/poppins/poppins-v5-latin-300italic.woff") format("woff"), url("../fonts/poppins/poppins-v5-latin-300italic.ttf") format("truetype"), url("../fonts/poppins/poppins-v5-latin-300italic.svg#Poppins") format("svg");
  /* Legacy iOS */ }
/* poppins-regular - latin */
@font-face {
  font-family: 'Poppins';
  font-style: normal;
  font-weight: 400;
  src: url("../fonts/poppins/poppins-v5-latin-regular.eot");
  /* IE9 Compat Modes */
  src: local("Poppins Regular"), local("Poppins-Regular"), url("../fonts/poppins/poppins-v5-latin-regular.eot?#iefix") format("embedded-opentype"), url("../fonts/poppins/poppins-v5-latin-regular.woff2") format("woff2"), url("../fonts/poppins/poppins-v5-latin-regular.woff") format("woff"), url("../fonts/poppins/poppins-v5-latin-regular.ttf") format("truetype"), url("../fonts/poppins/poppins-v5-latin-regular.svg#Poppins") format("svg");
  /* Legacy iOS */ }
/* poppins-italic - latin */
@font-face {
  font-family: 'Poppins';
  font-style: italic;
  font-weight: 400;
  src: url("../fonts/poppins/poppins-v5-latin-italic.eot");
```

```
  /* IE9 Compat Modes */
  src: local("Poppins Italic"), local("Poppins-Italic"), url("../fonts/poppins/poppins-v5-latin-italic.eot?#iefix") format("embedded-opentype"), url("../fonts/poppins/poppins-v5-latin-italic.woff2") format("woff2"), url("../fonts/poppins/poppins-v5-latin-italic.woff") format("woff"), url("../fonts/poppins/poppins-v5-latin-italic.ttf") format("truetype"), url("../fonts/poppins/poppins-v5-latin-italic.svg#Poppins") format("svg");
  /* Legacy iOS */ }
/* poppins-500 - latin */
@font-face {
  font-family: 'Poppins';
  font-style: normal;
  font-weight: 500;
  src: url("../fonts/poppins/poppins-v5-latin-500.eot");
  /* IE9 Compat Modes */
  src: local("Poppins Medium"), local("Poppins-Medium"), url("../fonts/poppins/poppins-v5-latin-500.eot?#iefix") format("embedded-opentype"), url("../fonts/poppins/poppins-v5-latin-500.woff2") format("woff2"), url("../fonts/poppins/poppins-v5-latin-500.woff") format("woff"), url("../fonts/poppins/poppins-v5-latin-500.ttf") format("truetype"), url("../fonts/poppins/poppins-v5-latin-500.svg#Poppins") format("svg");
  /* Legacy iOS */ }
/* poppins-500italic - latin */
@font-face {
  font-family: 'Poppins';
  font-style: italic;
  font-weight: 500;
  src: url("../fonts/poppins/poppins-v5-latin-500italic.eot");
  /* IE9 Compat Modes */
  src: local("Poppins Medium Italic"), local("Poppins-MediumItalic"), url("../fonts/poppins/poppins-v5-latin-500italic.eot?#iefix") format("embedded-opentype"), url("../fonts/poppins/poppins-v5-latin-500italic.woff2") format("woff2"), url("../fonts/poppins/poppins-v5-latin-500italic.woff") format("woff"), url("../fonts/poppins/poppins-v5-latin-500italic.ttf") format("truetype"), url("../fonts/poppins/poppins-v5-latin-500italic.svg#Poppins") format("svg");
  /* Legacy iOS */ }
/* poppins-600 - latin */
```

```
@font-face {
  font-family: 'Poppins';
  font-style: normal;
  font-weight: 600;
  src: url("../fonts/poppins/poppins-v5-latin-600.eot");
  /* IE9 Compat Modes */
  src: local("Poppins SemiBold"), local("Poppins-SemiBold"), url("../fonts/poppins/poppins-v5-latin-600.eot?#iefix") format("embedded-opentype"), url("../fonts/poppins/poppins-v5-latin-600.woff2") format("woff2"), url("../fonts/poppins/poppins-v5-latin-600.woff") format("woff"), url("../fonts/poppins/poppins-v5-latin-600.ttf") format("truetype"), url("../fonts/poppins/poppins-v5-latin-600.svg#Poppins") format("svg");
  /* Legacy iOS */ }
/* poppins-700 - latin */
@font-face {
  font-family: 'Poppins';
  font-style: normal;
  font-weight: 700;
  src: url("../fonts/poppins/poppins-v5-latin-700.eot");
  /* IE9 Compat Modes */
  src: local("Poppins Bold"), local("Poppins-Bold"), url("../fonts/poppins/poppins-v5-latin-700.eot?#iefix") format("embedded-opentype"), url("../fonts/poppins/poppins-v5-latin-700.woff2") format("woff2"), url("../fonts/poppins/poppins-v5-latin-700.woff") format("woff"), url("../fonts/poppins/poppins-v5-latin-700.ttf") format("truetype"), url("../fonts/poppins/poppins-v5-latin-700.svg#Poppins") format("svg");
  /* Legacy iOS */ }
/* poppins-700italic - latin */
@font-face {
  font-family: 'Poppins';
  font-style: italic;
  font-weight: 700;
  src: url("../fonts/poppins/poppins-v5-latin-700italic.eot");
  /* IE9 Compat Modes */
```

```
  src: local("Poppins Bold Italic"), local("Poppins-BoldItalic"), url("../fonts/poppins/poppins-v5-latin-700italic.eot?#iefix") format("embedded-opentype"), url("../fonts/poppins/poppins-v5-latin-700italic.woff2") format("woff2"), url("../fonts/poppins/poppins-v5-latin-700italic.woff") format("woff"), url("../fonts/poppins/poppins-v5-latin-700italic.ttf") format("truetype"), url("../fonts/poppins/poppins-v5-latin-700italic.svg#Poppins") format("svg");

  /* Legacy iOS */ }

/* poppins-800 - latin */

@font-face {

  font-family: 'Poppins';

  font-style: normal;

  font-weight: 800;

  src: url("../fonts/poppins/poppins-v5-latin-800.eot");

  /* IE9 Compat Modes */

  src: local("Poppins ExtraBold"), local("Poppins-ExtraBold"), url("../fonts/poppins/poppins-v5-latin-800.eot?#iefix") format("embedded-opentype"), url("../fonts/poppins/poppins-v5-latin-800.woff2") format("woff2"), url("../fonts/poppins/poppins-v5-latin-800.woff") format("woff"), url("../fonts/poppins/poppins-v5-latin-800.ttf") format("truetype"), url("../fonts/poppins/poppins-v5-latin-800.svg#Poppins") format("svg");

  /* Legacy iOS */ }

/* poppins-800italic - latin */

@font-face {

  font-family: 'Poppins';

  font-style: italic;

  font-weight: 800;

  src: url("../fonts/poppins/poppins-v5-latin-800italic.eot");

  /* IE9 Compat Modes */
```