Project Report: Resume Analyzer with Flask on Google Colab

1. Introduction

In this project, we built a simple web application called Resume Analyzer using Flask inside Google Colab.

The main goal was to allow users to upload their resume in PDF or DOCX format, extract useful information like name, email, phone number, major, and skills, and then match those skills with suitable job profiles.

Because Google Colab is not a native web server environment, extra steps were needed to expose the Flask app online.

2. Code Description

- **Step 1: Environment Setup**

First, it installs all the needed libraries:

- **flask**: to create the web app.
- **pyngrok**: to create a public link for the Flask app inside Colab.
- **PyPDF2**: to read text from PDF files.
- **python-docx**: to read text from DOCX (Word) files.

After installing, the code sets up ngrok using an authentication token.

This token allows ngrok to open a secure tunnel from Colab to the internet.

Step 2: Creating the Flask Application

- A Flask app is created using **Flask(__name__)**.
- One main route (**/**) is defined:
- If the user visits the page, they see a simple upload form.
- If the user uploads a resume, the server reads and processes the file.
- The **HTML page** is directly included inside the Python code using **render_template_string()**.
- It has a nice design using basic CSS.
- The page has an upload button and displays results after submission.

Step 3: Resume Processing

When a user uploads a resume:

1. The server saves it temporarily.

If the file is a PDF, it uses **PyPDF2** to extract the text.

If the file is a DOCX, it uses **python-docx** to extract the text.

Information Extraction:

It uses regular expressions (regex) to search the text for:

- **Name**: Looks for words after “Name” or “Full Name”.
- **Email**: Searches for any email pattern (e.g., user@example.com).
- **Phone Number**: Searches for Saudi-style numbers (starting with "05").
- **Major Field**: Detects words like Computer Science, Engineering, etc.
- **Skills**: Looks for known skills (e.g., Python, SQL, Machine Learning).

Step 4: Analyzing the Resume

A ResumeAnalyzer class is created.

- It has a list of important skills.
- It also has predefined job profiles (Data Scientist, Web Developer, DevOps Engineer) with required skills.

Matching Logic:

- The analyzer uses TF-IDF Vectorizer to turn the skills into mathematical vectors.
- Then, it uses Cosine Similarity to compare the resume’s skills with each job's skills.
- It calculates a percentage match score for each job.

Step 5: Displaying the Output

After processing, the extracted information and match scores are shown back on the website.

For example:

- Name: Saeed
- Email: saeed24uj@gmail.com
- Phone: 0508149143
- Major: Computer Science
- Skills: Python, SQL, Data Analysis

Matching Jobs:

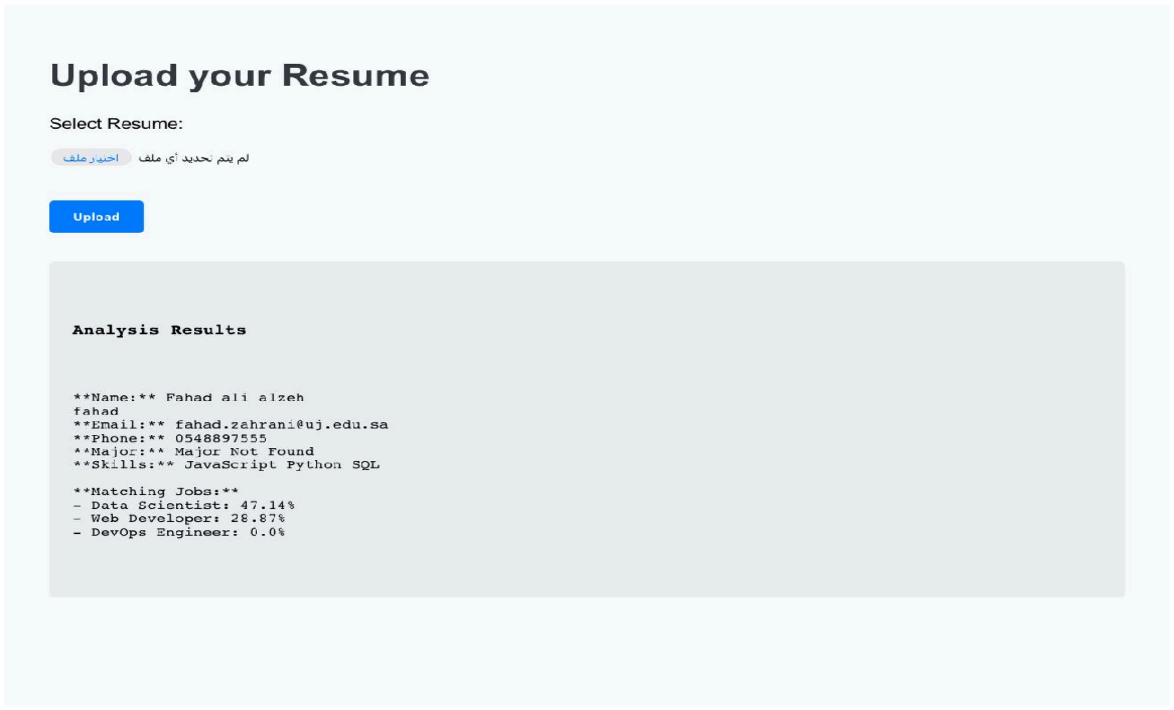
Data Scientist: 95%___Web Developer: 10%___DevOps Engineer: 5%

4. Outputs

The final output of the project is a screenshot of the working application.

When a user uploads their resume, the web app displays:

- The extracted Name, Email, Phone, Major, and Skills.
- A list of matching job profiles with a percentage score showing how closely the user's skills match the jobs.



```
Analysis Results


**Name:** Fahad ali alzeh
fahad
**Email:** fahad.zahrani@uj.edu.sa
**Phone:** 0548897555
**Major:** Major Not Found
**Skills:** JavaScript Python SQL

**Matching Jobs:**
- Data Scientist: 47.14%
- Web Developer: 28.87%
- DevOps Engineer: 0.0%
```

3. Challenges

While working on this project, the following challenges were encountered:

- There were difficulties in running flask directly from colap, and it turned out that it needed autokun from the ngrok website to start running it. There was also an error, and before that we had to prepare the colap website and add html and it worked.

Solution:

- We needed to download and configure the Authtoken for ngrok.
- By installing pyngrok and setting the auth token, we were able to create a public tunnel that allowed the Flask app to be accessed via an external link.

5. Conclusion

This project demonstrated how to deploy a simple Flask application inside a Colab notebook using pyngrok.

It also showed how basic resume parsing and job matching logic can be implemented using regular expressions, text extraction libraries, and machine learning concepts like TF-IDF and Cosine Similarity.

Despite initial technical challenges, especially related to environment setup, the project was successfully completed with a functional web application and a visible public link.