Milestone 3 - **AI-powered Job Seek Tool**

– A conversational agent

By – Swetha Gendlur Nagarajan

Note to Grader: Dual Interface Implementation

As part of this milestone, I have developed two distinct user interfaces for the **Theta Career Assistant** to demonstrate both the **flexibility of the system's logic** and its **usability across different application contexts**.

1. **Normal Interface (Form-Based Layout)**

   The initial version of the system was built using a structured, form-driven layout. This allowed for clearer debugging, precise control of output stages, and validation of all backend components such as resume parsing, skill extraction, role matching, and course/job recommendations.

2. **Chatbot Interface (Streamlit Conversational UI)**

   After confirming the core logic worked reliably in the normal layout, I transitioned the same logic into a chat-style interface using Streamlit's chat API. This conversational version mimics a real-world AI assistant, providing a more engaging and intuitive user experience with context-aware prompts and live updates.

   The reason for implementing the **normal UI first** was to ensure that the **functional pipeline logic (resume scoring, skill gap analysis, profile tier prediction, etc.) was correct and reliable**. Once validated, I transferred the logic to the chat-based UI to explore how the assistant performs in a more interactive, real-world simulation.

   Both versions use the **same backend datasets, predictive models, and Gemini-powered content generation**. This dual-interface approach illustrates the modularity of the system and its readiness for different deployment environments—from internal tools to user-facing career guidance platforms.

Thank you for reviewing both interfaces as part of this submission

1. Stage 1: Resume Parsing Logic : Skill Extraction and Role Matching

As part of the AI-powered Job Seek Tool, this module implements an intelligent resume parsing pipeline that extracts, cleans, and standardizes candidate skill information from unstructured resume PDFs. The parsed data is then used to compute a role-specific match score by comparing the extracted skills against role-based requirements. This lays the foundation for providing personalized career insights, recommendations, and skill-gap interventions.

The resume parsing logic implemented using pdfplumber, re, and custom text sectioning forms the foundational step of our AI-powered career recommendation system. It plays a crucial role in transforming raw, unstructured resume PDFs into structured, analyzable skill data — enabling personalized job-role alignment and actionable recommendations.

1. 1 Parsing Logic Overview

The resume parser is built using Python's pdfplumber library for PDF text extraction and regular expressions (re) for content segmentation and normalization. The key components of this logic are:

Text Extraction: The function extract_text_from_pdf(file_path) reads each page of a resume PDF and concatenates the textual content.

Section Detection: A dictionary of predefined section headers (e.g., Skills, Experience, Education, Certifications) is used to identify and segment the resume content into logical parts. The extract_sections() function detects these headers using regex-based matching and organizes the resume text accordingly.

S*kill Extraction*: The extract_skills_from_section() function specifically processes the "Skills" section of the resume. It performs the following transformations:

Removes redundant prefixes (e.g., "Libraries:", "Languages:")

- Normalizes text to lowercase and strips special characters
- Expands compound expressions (e.g., *ResNet18/50* becomes *ResNet18, ResNet50*)
- Tokenizes the string into distinct skills using multiple delimiters (comma, slash, 'and', etc.)
- Filters out irrelevant entries (digits, single-character tokens)

1.2 Use Cases:

- **High Score:** If the match score exceeds a predefined threshold (e.g., 40%), the user is marked as a strong candidate. The system:
  - Suggests job openings
  - Provides recruiter contacts
  - Generates a personalized cold email template

- **Low Score:** If the user has multiple missing skills:
  - The missing skills are reported
  - Targeted Coursera course recommendations are provided using the Coursera popularity prediction model

2. Stage 2: Model Evaluation : Role-Based Skills Classification

The purpose of this evaluation is to assess the performance of a trained **Random Forest Classifier** on the Role-Based Skills dataset. The model is tasked with classifying job roles into two categories:

- **Entry-Level** (≤ 2 years of experience)
- **Experienced** (> 2 years of experience)

This binary classification supports downstream decision-making within the AI-powered job search tool by helping determine user-job alignment and tailoring skill-gap analysis.

Metric	**Value**
Accuracy	1.00
ROC AUC	1.00
Precision (Entry)	1.00
Recall (Entry)	1.00
F1-score (Entry)	1.00
Precision (Experienced)	1.00
Recall (Experienced)	1.00
F1-score (Experienced)	1.00

3. Stage 3: Career Assistant: Backend Logic and System Design

The **Gemini Career Assistant** functions as an intelligent, personalized job-matching chatbot that guides users through career planning, resume evaluation, and learning path recommendations. This system integrates resume parsing, machine learning, and large language model (LLM) generation to provide tailored, data-driven guidance.

1. User Interaction and Role Intent

The conversation begins by gathering the user’s name, current role, and the role they aspire to pursue. This information is used to contextualize further interactions and enables the system to retrieve role-specific job market trends and skill requirements. By distinguishing between students, experienced professionals, and career switchers, the assistant dynamically adjusts the information and recommendations it delivers.

2. Role-Specific Market Insights and Skill Requirements

The chatbot uses the **Job Description Dataset** to identify top hiring locations for the user's desired role. This dataset includes structured fields such as job titles, countries, and industry sectors, which allow the assistant to highlight where demand is geographically concentrated. Simultaneously, the assistant prompts **Google's Gemini API** to generate a list of key technical and soft skills for the selected role. This real-time LLM-generated skill list simulates job description analysis and primes the resume comparison logic.

3. Entry-Level Guidance and Skill Development

For students and those entering a new domain, the assistant recommends entry-level learning paths. It filters the **Coursera Course Dataset** to find relevant beginner-friendly courses based on the target role's core keywords. Additionally, it taps into the **LinkedIn Dataset** to identify professionals in similar roles and suggests them as potential mentors. To support professional outreach, Gemini is used to generate a short, context-aware message template that the user can send when requesting career advice.

4. Resume Upload and Skill Extraction

Experienced users are prompted to upload a PDF version of their resume. The assistant utilizes the pdfplumber library to extract raw text from the uploaded document. A rule-based function then segments the text into logical sections using common headers such as "Skills," "Work Experience," and "Certifications." Skills are tokenized, cleaned, and standardized using regex, forming a set of core competencies extracted from the resume. This forms the basis for role matching.

5. Resume-to-Role Matching

The extracted skills are matched against the role-specific required skills from the **Role-Based Skills Dataset**. The system computes:

- The number of matched skills (intersection),
- Missing skills (difference),
- A resume match score calculated as the percentage of required skills the candidate possesses.

This enables a transparent evaluation of the user's preparedness for the chosen role.

6. Optional Profile Richness Evaluation

If the user opts in, the assistant computes an additional set of features from the resume such as:

- Estimated years of experience (based on textual patterns),
- Certification counts,

- Use of seniority-indicative language (e.g., "lead", "manager"),
- Experience-to-skill ratio.

These features are passed to a pretrained **Random Forest or Logistic Regression model**, loaded from a joblib file (ll_model), to predict the user's professional tier (e.g., Beginner, Intermediate, Expert) based on similar profiles from the **LinkedIn Dataset**. This step provides a more nuanced profile strength analysis.

7. Skill Gap-Based Recommendations

If the resume score reveals more missing skills than matched, the assistant recommends relevant courses from the Coursera dataset based on overlapping keywords. Additionally, it prompts Gemini to suggest courses from other platforms. The assistant again identifies professionals working in the same role and generates a customized message asking for skill development advice.

8. Personalized Outreach and Job Applications

For users with a strong resume match, the system facilitates next steps. It generates a referral email tailored to the user's matched skills and the target job role using Gemini. The assistant also lists active job titles and companies from the **Job Description Dataset** and displays recruiter contact information from the **LinkedIn Dataset**. This enables users to confidently initiate professional outreach and job applications.

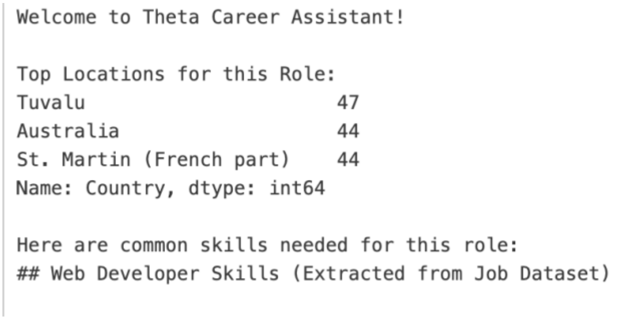
9. Tools, Models, and Datasets Used

The backend system integrates multiple components:

- **Datasets:** Coursera courses (df_cc), Job descriptions (df_jd), LinkedIn profiles (df_ll), Role-based required skills (df_rs)
- **Models:** A pretrained professional-tier classifier (ll_model), Google's Gemini LLM (for skill generation, email drafting)
- **Libraries:** pdfplumber for PDF text extraction, re and pandas for processing, and joblib for model deployment.

The Gemini Career Assistant is an end-to-end intelligent job assistant that blends structured dataset insights with natural language generation and predictive modeling. By dynamically adapting to user roles, experience levels, and resume content, it offers actionable guidance for learning, networking, and job applications. Its modular logic ensures a robust backend capable of supporting a wide range of career navigation scenarios.

Sample Output:



```
Welcome to Theta Career Assistant!

Top Locations for this Role:
Tuvalu                      47
Australia                   44
St. Martin (French part)    44
Name: Country, dtype: int64

Here are common skills needed for this role:
## Web Developer Skills (Extracted from Job Dataset)
```

4. Streamlit-Enhanced Interactive Chatbot

To improve user experience and accessibility, the Gemini Career Assistant was enhanced using **Streamlit**, a Python-based web framework for building interactive, data-driven applications. This upgrade transformed the previously terminal-based AI assistant into a visually engaging, browser-accessible **chat-style interface**. The interface mimics a natural conversation, guiding users through each step of the job discovery and resume analysis pipeline while enabling file uploads and dynamic content rendering.

Conversational Chat Flow Logic

The chatbot adopts a multi-stage interaction flow, implemented using st.session_state to persist user inputs and system state across turns. The assistant sequentially prompts users to provide their name, current role, and target job role. Based on the responses, it branches into different interaction paths, catering separately to students, experienced professionals, and those changing career domains. Each stage of the conversation is reactive, meaning the assistant dynamically adjusts follow-up questions and outputs based on user data and logic outcomes.

Dataset-Driven Recommendations

- **Job Descriptions (df_jd)**: Provides top hiring locations and detailed job specifications per role.
- **Coursera Courses (df_cc)**: Used to recommend beginner and advanced learning paths based on skill gaps.
- **LinkedIn Profiles (df_ll)**: Identifies professionals in relevant roles and surfaces contact details for networking.
- **Role-Based Skills (df_rs)**: Contains structured skills required per position title, used for resume-role matching.

Resume Analysis and Skill Matching

Upon reaching the resume evaluation stage, the assistant prompts the user to upload a PDF file using Streamlit’s built-in uploader. The file is parsed using pdfplumber, and the “Skills” section is extracted through header-based segmentation and regex tokenization. Extracted skills are then compared with role-specific requirements from the Role-Based Skills dataset to compute:

- **Matched Skills**
- **Missing Skills**
- **Resume Score (% match)**

These results are displayed in the chat interface in real time, offering clear feedback to the user on their readiness for the desired role.

For users interested in a deeper analysis, the assistant optionally computes a **profile richness vector** using features like experience mentions, certification frequency, skill density, and leadership language. These features are passed into a pre-trained **LinkedIn profile tier classifier (ll_model)**, which predicts the user's professional tier (e.g., Beginner, Intermediate, Expert). The tier is used to tailor subsequent recommendations, such as course difficulty and email templates for outreach.

Course and Career Suggestions

When significant skill gaps are found, the assistant recommends personalized courses:

- **From Coursera Dataset**: Skill-matching courses with links for immediate access.
- **Gemini Suggestions**: Real-time course ideas generated by Gemini, focused on filling exact gaps.

Cold Email and Referral Templates

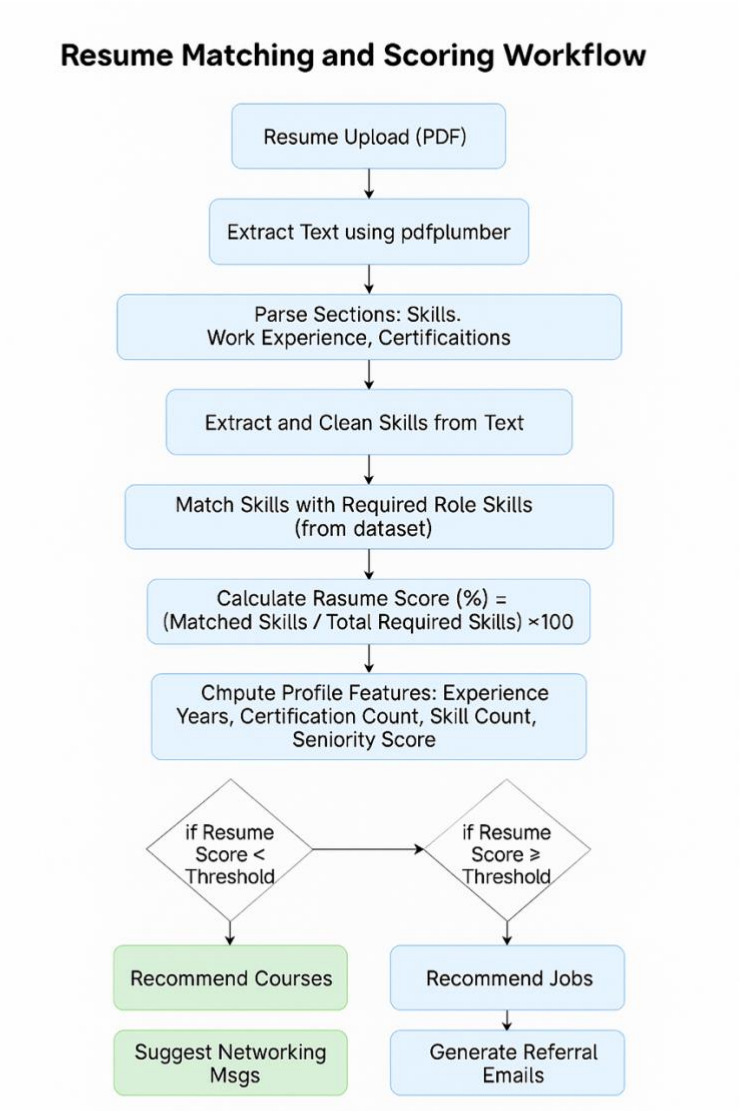
For users who have a strong resume match, Gemini is prompted to generate a tailored cold email template. This message references the job role, employer, and matched skills, and is optimized for professional outreach. In parallel, a list of professionals from the LinkedIn dataset is displayed, giving the user concrete people to reach out to for networking or referrals.

Evaluation

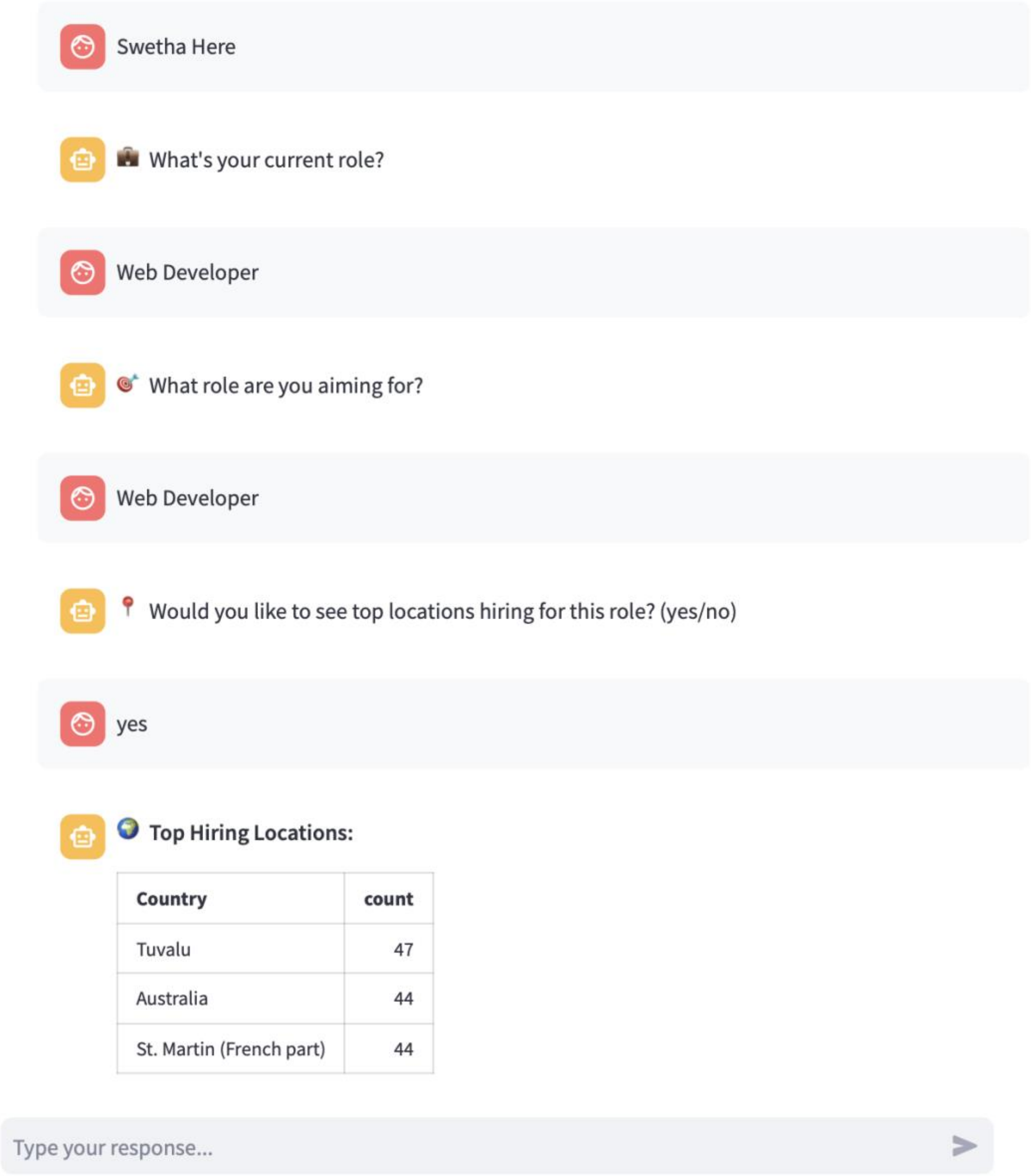
The AI-powered Job Seek Tool was evaluated through multiple quantitative and qualitative measures. The Role-Based Skills Classifier, a Random Forest model, achieved 100% across Accuracy, ROC AUC, Precision, Recall, and F1-Score, demonstrating excellent ability to distinguish between entry-level and experienced job roles. Additionally, resume-to-role matching was quantitatively assessed through a computed resume match score, indicating the percentage of required skills a candidate possessed. Clear thresholds were established: high-scoring users received immediate job and networking recommendations, while low-scoring users were guided toward skill development. The consistent use of classification metrics and real-time resume scoring ensured the system's predictive accuracy and reliability.

Bias & Limitations

While the system performed robustly, certain biases and limitations were identified. The resume parsing pipeline relies on standard section headers and regular expressions, meaning resumes with non-traditional formats or creative layouts might not be parsed accurately. Skill matching is heavily dependent on the completeness of the Role-Based Skills dataset and the specificity of the extracted resume skills, which could lead to over- or under-estimation of match scores. Additionally, course recommendations and professional outreach suggestions are limited to the datasets available at the time of development, and real-world job markets may evolve beyond the captured data. These factors highlight the need for continuous dataset updates and expansion of parsing robustness.


Resume Matching and Scoring Workflow
Resume Upload (PDF)
Extract Text using pdfplumber
Parse Sections: Skills. Work Experience, Certificaitions
Extract and Clean Skills from Text
Match Skills with Required Role Skills (from dataset)
Calculate Rasume Score (%) = (Matched Skills / Total Required Skills) ×100
Chmpute Profile Features: Experience Years, Certification Count, Skill Count, Seniority Score
if Resume Score < Threshold
if Resume Score ⩾ Threshold
Recommend Courses
Suggest Networking Msgs
Recommend Jobs
Generate Referral Emails

5. Output Snippets:



Swetha Here

💼 What's your current role?

Web Developer

🎯 What role are you aiming for?

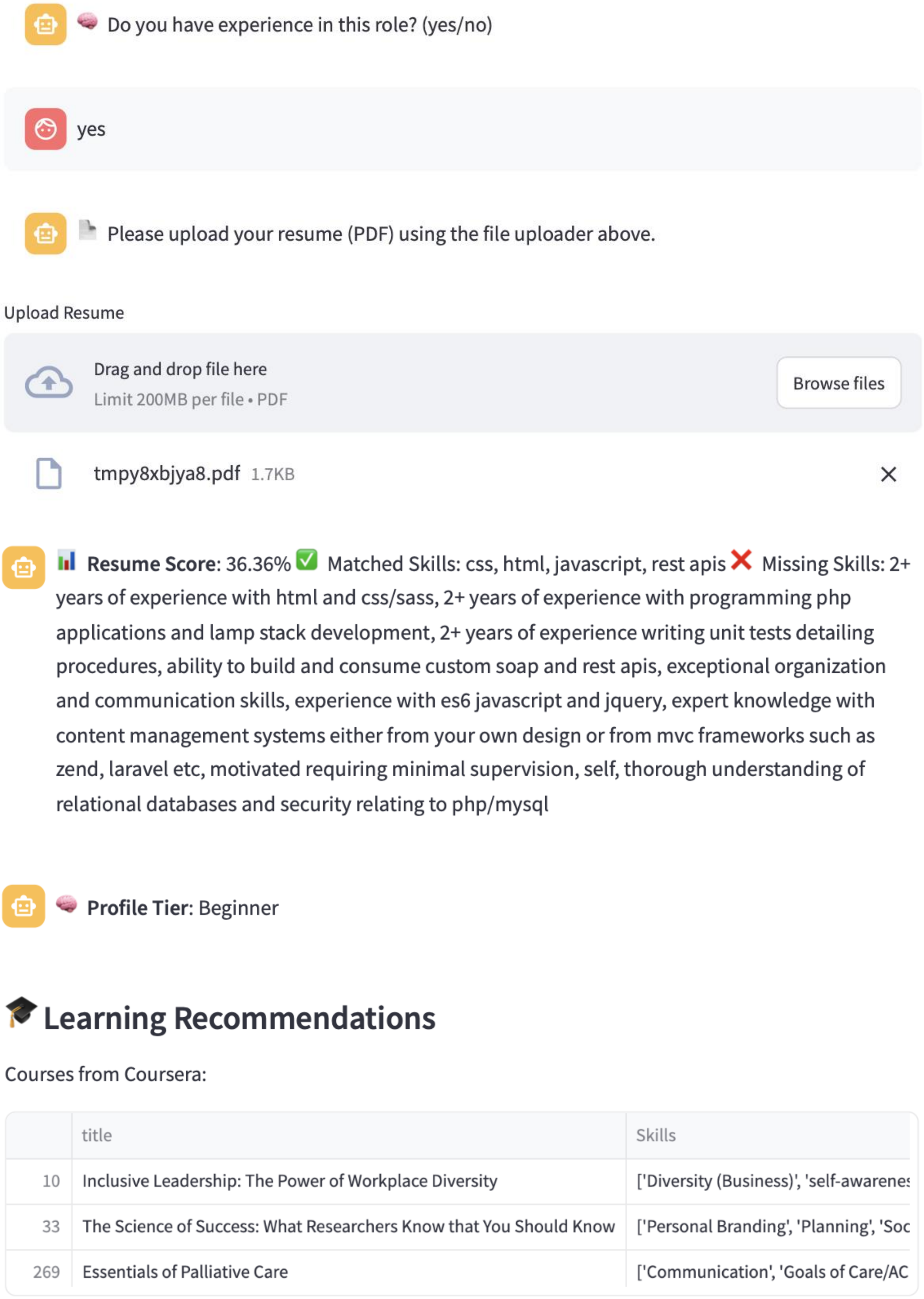
Web Developer

📍 Would you like to see top locations hiring for this role? (yes/no)

yes

🌍 **Top Hiring Locations:**

Country	count
Tuvalu	47
Australia	44
St. Martin (French part)	44



🧠 Do you have experience in this role? (yes/no)

yes

📄 Please upload your resume (PDF) using the file uploader above.

Upload Resume

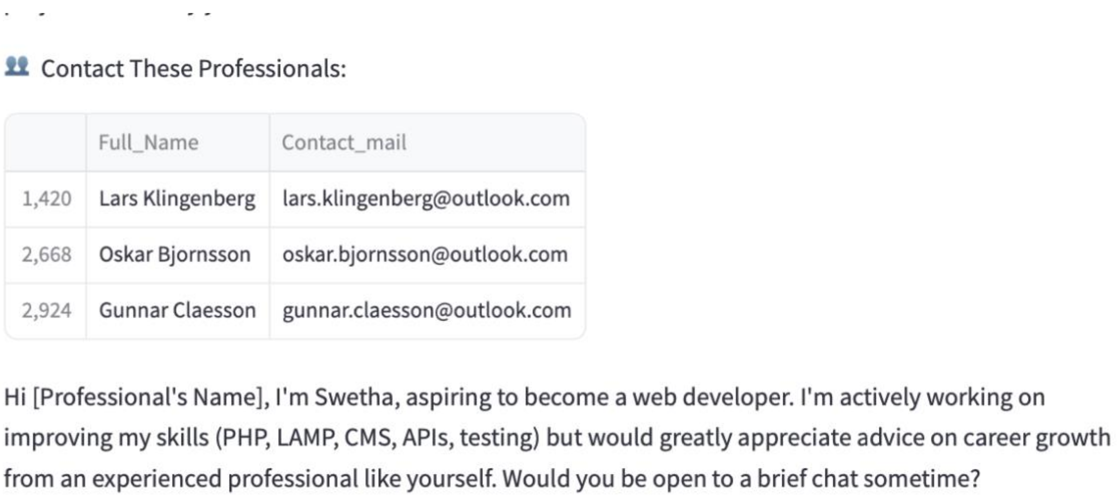
📊 **Resume Score**: 36.36% ✅ Matched Skills: css, html, javascript, rest apis ❌ Missing Skills: 2+ years of experience with html and css/sass, 2+ years of experience with programming php applications and lamp stack development, 2+ years of experience writing unit tests detailing procedures, ability to build and consume custom soap and rest apis, exceptional organization and communication skills, experience with es6 javascript and jquery, expert knowledge with content management systems either from your own design or from mvc frameworks such as zend, laravel etc, motivated requiring minimal supervision, self, thorough understanding of relational databases and security relating to php/mysql

🧠 **Profile Tier**: Beginner

🎓 Learning Recommendations

Courses from Coursera:

	title	Skills
10	Inclusive Leadership: The Power of Workplace Diversity	['Diversity (Business)', 'self-awarenes
33	The Science of Success: What Researchers Know that You Should Know	['Personal Branding', 'Planning', 'Soc
269	Essentials of Palliative Care	['Communication', 'Goals of Care/AC



👥 Contact These Professionals:

	Full_Name	Contact_mail
1,420	Lars Klingenberg	lars.klingenberg@outlook.com
2,668	Oskar Bjornsson	oskar.bjornsson@outlook.com
2,924	Gunnar Claesson	gunnar.claesson@outlook.com

Hi [Professional's Name], I'm Swetha, aspiring to become a web developer. I'm actively working on improving my skills (PHP, LAMP, CMS, APIs, testing) but would greatly appreciate advice on career growth from an experienced professional like yourself. Would you be open to a brief chat sometime?

Aspect	**normal_output.pdf(Form-Based Output)**	**output.pdf (Chat-Style Output)**
Interface Style	Normal form-style web app (structured sections)	Chatbot-like conversation (dynamic flow)
Start Interaction	Immediately collects: Name → Current Role → Target Role	Greets user like a conversation: "What's your name?" "What's your current role?"
Top Hiring Locations	Displayed with no user confirmation (automatically shown)	Asks the user **"Would you like to see top locations?"** before showing
Skills Display	Shows detailed list: Technical + Soft Skills with realistic job data simulation (with %s like "75%")	Also shows detailed list but more **chatty**, phrases as "hypothetical" dataset simulation
Resume Upload	Upload area shown in structured page directly	Upload requested through chat prompts ("Please upload your resume.")
Resume Match Result	Shows: Extracted Skills → Match Score (36.36%) → Matched & Missing Skills	Also shows: Extracted Skills → Match Score (36.36%) → Matched & Missing Skills
Profile Tier Prediction	Shows: "Predicted Tier: Beginner" clearly after resume analysis	Also shows: "Profile Tier: Beginner" after resume analysis

Course Recommendations	Lists Coursera courses categorized by skill gap clusters (e.g., "Organization & Communication Skills") with practical advice	Gives very **advanced course search strategies** (e.g., "Advanced REST API patterns," "OWASP security in PHP") and URLs
Professional Contacts	Provides a **sample cold message** to send to professionals	Lists **real names and emails** (like Lars Klingenberg, etc.) to contact
Overall Tone	**Structured**, like a professional application walkthrough	**Conversational**, like chatting with a career counselor
Technical Details	Slightly deeper listing of missing skills	Slightly richer advice for **next skill-building steps** with URLs

Authorship & Use of Generative AI

This project was fully implemented by me, Swetha Gendlur Nagarajan, including all coding, model integration, dataset preparation, and UI development (both form-based and chatbot interfaces). The logic and flow were designed from scratch, based on my understanding of applied data science, natural language processing, and job-matching systems.

I used large language models (LLMs) like ChatGPT and Gemini at specific points to support my learning process and improve productivity. Their use was limited to:

- Understanding the basic structure of a Streamlit app, especially the initial skeleton required to create a multi-step chatbot interface and implement st.session_state.

- Improving the clarity of technical documentation (README formatting, function naming ideas, etc.).

- Identifying common industry skills, course keywords, and job-market phrasing for realism.

While I referred to LLMs to understand Streamlit's layout, all final code, logic, and customization were written by me. No generated code was copied without deep modification, validation, and testing. The entire project reflects my own implementation and understanding and complies with the UF Student Honor Code.