SMARTRESUME GENERATOR: CUSTOMIZED RESUMES FOR EVERY OPPORTUNITY

INTRODUCTION

The Resume Generator project aims to develop an AI-driven tool for automating the creation of professional resumes. The objective is to build a generative model that can craft tailored resumes based on user inputs such as personal information, job experience, and career goals. By analyzing these details, the model produces well-structured resumes that effectively highlight the candidate's skills, achievements, and qualifications. This tool streamlines the resume creation process, enabling users to generate polished and personalized resumes quickly, thereby enhancing their ability to present themselves effectively to potential employers and improve their job application success.

Scenario 1: University Career Services

A university's career services department offers a resume generator to assist students in creating polished resumes tailored to specific industries. Students input details such as their academic achievements, internships, and extracurricular activities, and the tool generates resumes highlighting the most relevant skills and experiences for fields like finance, engineering, or marketing. This service helps students stand out in competitive job markets, similar to how career advisors provide tailored guidance based on individual career goals.

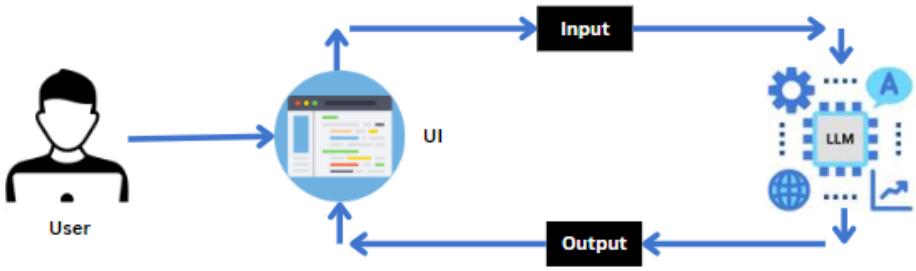
Scenario 2: Job Placement Agencies

A job placement agency uses the resume generator to streamline the job application process for clients. Candidates provide their work history, skills, and job preferences, and the tool produces multiple resume versions optimized for different job roles. This ensures that clients can quickly apply to various positions with resumes tailored to each opportunity, enhancing their chances of securing interviews. The approach is akin to personalized job coaching, where advisors help candidates, craft resumes for specific roles.

Scenario 3: Freelancers and Gig Workers

Freelancers and gig workers use the resume generator to create dynamic resumes that reflect their diverse project experience and skill sets. By inputting details of past projects, skills, and client testimonials, the tool generates resumes suited for different types of gigs, such as web development, graphic design, or writing. This allows freelancers to quickly customize their resumes when bidding for new projects, similar to how they might adjust their portfolios to match the needs of potential clients.

Architecture:



Project Flow

1. User Input via Streamlit UI:

- Users input a prompt (e.g., topic, keywords) and specify parameters such as the desired length, tone, or style through the Streamlit interface.

2.Backend Processing with Generative AI Model:

- The input data is sent to the backend, where it interfaces with the selected Generative AI model (e.g., GPT-4, Gemini, etc.).
- The model processes the input, generating text based on the specified parameters and user input.

3.Content Generation:

- The AI model autonomously creates content tailored to the user's specifications. This could be a blog post, poem, article, or any other form of text.

4.Return and Display Generated Content:

- The generated content is sent back to the frontend for display on the Streamlit app.
- The app presents the content to the user in an easily readable format.

5.Customization and Finalization:

- Users can further customize the generated content through the Streamlit UI if desired. This might include editing text, adjusting length, or altering tone.

6.Export and Usage:

- Once satisfied, users can export or copy the content for their use, such as saving it to a file or directly sharing it.

Requirements Specification

Install the libraries

- pip install streamlit
- pip install google.generativeai

Initializing the Models

Link: https://ai.google.dev/gemini-api

Enable the Gemini API

- Once your project is created, navigate to the API & Services Dashboard.
- Click on Enable APIs and Services at the top.
- Search for "Gemini API" in the API library.
- Select the Gemini API and click Enable.
- After initialising the model,building the codes to run the streamlit.

1.BACKEND.PY

```
import google.generativeai as genai


genai.configure(api_key="AIzaSyDPjM3nY9f6RsvgXve4D4lUUSfH7aIr4yU")  # Replace with your actual API key


# Use the correct model name
model = genai.GenerativeModel("gemini-1.5-pro")


def generate_resume(name, job_title):
    context = f"Generate a professional resume for {name}, applying for {job_title}. Include summary, skills, experience, and education."
    response = model.generate_content(context)
    return response.text if isinstance(response.text, str) else response.text[0].text
```

2.APP.PY

```
import streamlit as st
from backend import generate_resume


# Streamlit UI
st.title("Smart Resume Generator")
name = st.text_input("Enter your Name")
job_title = st.text_input("Enter your Job Title")
```

```
if st.button("Generate Resume"):
    if name and job_title:
        resume = generate_resume(name, job_title)
        st.markdown(resume)
    else:
        st.warning("Please enter both Name and Job Title")
```

3.IMPORT STREAMLIT AS ST.PY

```
import streamlit as st
import google.generativeai as genai


# Configure Google Generative AI
genai.configure(api_key="AIzaSyDPjM3nY9f6RsvgXve4D4lUUSfH7aIr4yU")  # Replace with your actual API key


# Initialize Model
model = genai.GenerativeModel("gemini-1.0-pro")  # Use the correct model name


# Function to generate resume
def generate_resume(name, job_title):
    context = f"Generate a professional resume for {name}, applying for {job_title}. Include summary, skills, experience, and education."
    response = model.generate_content(context)
    return response.text if isinstance(response.text, str) else response.text[0].text


# Streamlit UI
st.title("Smart Resume Generator")
name = st.text_input("Enter your Name")
job_title = st.text_input("Enter your Job Title")
```

```
if st.button("Generate Resume"):
    if name and job_title:
        resume = generate_resume(name, job_title)
        st.markdown(resume)
    else:
        st.warning("Please enter both Name and Job Title")
```

4.The app would run using the code streamlit run app.py

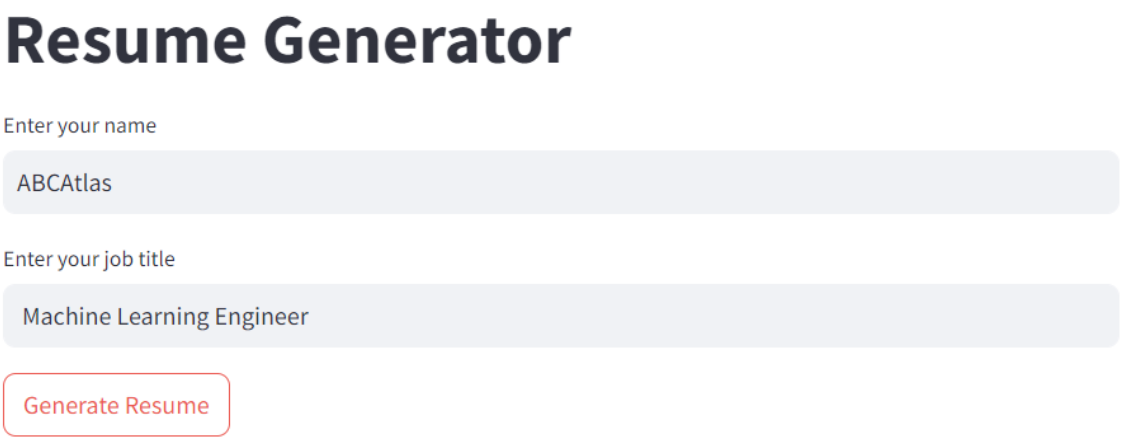
Running the web application

```
You can now view your Streamlit app in your browser.

Local URL: http://localhost:8501
Network URL: http://192.168.0.6:8501
```

The application is now running and can be accessed locally through the provided URL. It is also available on the network via the network URL, allowing access from other devices on the same network.

OUTPUT



Generated Resume

ABCAtlas

Machine Learning Engineer

Contact: [Your Email Address] | [Your Phone Number] | [Add LinkedIn Profile URL (Optional)]

Summary

Highly motivated and results-oriented Machine Learning Engineer with [Number] years of experience in designing, developing, and deploying machine learning models to solve real-world problems. Proven ability to analyze complex datasets, build predictive models, and deliver actionable insights. Passionate about staying at the forefront of advancements in machine learning and applying them to create innovative solutions.

Experience

[Company Name], [City, State] - Machine Learning Engineer | [Start Date] - [End Date]

- Developed and deployed a deep learning model using TensorFlow that improved [Specific Metric] by [Percentage] for [Project/Problem Description].
- Designed and implemented a machine learning pipeline to automate [Specific Task] which reduced processing time by [Percentage].
- Collaborated with cross-functional teams to integrate machine learning models into existing systems and workflows.

Projects

- **[Project Title]:** Developed a [Model Type] model for [Project Description]. Achieved [Specific Metric] of [Value]. Utilized [Tools/Technologies]. ([Add Project Link if applicable])
- **[Project Title]:** Built a [Model Type] model to [Project Description]. Improved [Specific Metric] by [Percentage]. Leveraged [Tools/Technologies]. ([Add Project Link if applicable])

Skills

Programming Languages: Python, R, SQL

Machine Learning Libraries/Frameworks: TensorFlow, PyTorch, Scikit-learn, Pandas, NumPy

Cloud Computing Platforms: AWS (Amazon SageMaker, EC2, S3), Google Cloud Platform (GCP)

Algorithms: Regression, Classification, Clustering, Deep Learning, NLP

Databases: SQL, NoSQL

Tools: Git, Jupyter Notebook, Docker

Other: Data Visualization, Statistical Modeling, Feature Engineering, Model Deployment, A/B Testing

Education

[Your University Name], [City, State] | [Degree] in [Major] | [Your Graduation Year]

[Relevant Coursework or Certifications (Optional)]

For the inputs "ABCAtlas" as the name and "Machine Learning Engineer" as the job title, the output resume would be a professionally formatted document that highlights relevant skills and experiences. It would include sections such as a summary of qualifications, relevant work experience, educational background, and skills tailored to machine learning. The resume might list dummy projects and achievements, formatted in Markdown for easy readability. For example, it could feature sections like "Professional Summary," "Experience," and "Education," with each section populated with generic but relevant content to showcase the expertise and fit for a machine learning engineering role.