Final Report: Flask Keyword Extraction API

Submitted by: Bisma Tahir

Project Overview:

This project is a Flask-based web API designed for keyword extraction from user-submitted text. It uses TextBlob to extract noun phrases as keywords. The API is deployed serverlessly on Vercel, ensuring ease of access and scalability. It supports both command-line interface (CLI) and HTML form-based input, making it flexible for different user preferences and testing environments.

Requirements

- Python 3.9
- Flask
- TextBlob
- requirements.txt
- vercel.json for deployment config

Deployment

- Platform: Vercel
- GitHub used as codebase source
- Deployed via CLI using

  vercel --prod

Features

- POST endpoint: /api/v1/keywords
  - Input: JSON with "text" key
  - Output: JSON list of extracted keywords

- GET endpoint: / returns an HTML form for manual testing

Results

- The deployed API successfully extracts noun phrases from user input using TextBlob.
- Works via:
    - **Postman** (tested with JSON body)
    - **CLI client**
    - **Browser form**

Example input:

json

{"text": "Machine learning helps computers learn from data."}

Example output:

{"keywords": ["machine learning"]}

Improvements Made

1. **Refactored code structure** for better readability and logging.
2. **Logging system** added with timestamps for easier debugging.
3. **Static and template folder setup** aligned for better form rendering.
4. **API key management** via .env using dotenv.
5. **Removed explicit nltk downloads** and fallbacks to avoid deployment issues.

Bugs Encountered & Fixes

During development and deployment, several bugs were encountered and resolved. The first issue was that the **NLTK resource was not found** when running the app on Vercel. This occurred because Vercel's serverless environment does not persist local NLTK corpora. To fix this, the project was modified to **remove any manual nltk.data.path.append() logic**, relying instead on TextBlob's built-in default behavior which gracefully handles missing corpora.

Another issue was a **500 Internal Server Error**, triggered when the "text" key was missing in the incoming JSON body. This was resolved by **adding input validation** to check for the presence of the "text" field and returning a clear error message if it was missing.

A separate bug arose when **static files and templates were not loading properly**. This was traced back to an incorrect folder structure. The fix involved explicitly **defining the template_folder and static_folder paths** when initializing the Flask app to ensure Flask could correctly locate these resources.

Finally, there was a **deployment crash on Vercel**, caused because Vercel couldn't find the correct entry point to start the app. This was resolved by **updating the vercel.json file to correctly point to runserver.py as the entry point**, ensuring that the app launched correctly in the serverless environment.

Conclusion:

This project successfully demonstrates the deployment of a Flask-based keyword extraction API using serverless architecture via Vercel. Through the integration of TextBlob, the system efficiently identifies relevant keywords from user input. Key challenge such as missing NLTK resources, deployment errors, and input validation issues were identified and resolved during development. The final application is stable, user-friendly, and accessible both via CLI and a web interface. This deployment highlights the feasibility of lightweight NLP services in a serverless environment.