karthik_proj

Project Overview

This project is a simple Flask application that uses the Gemini 2.0 Flash API to generate text based on user prompts. The primary purpose is to demonstrate a basic integration with Google's Gemini API. The target user is any developer wanting to quickly understand how to interface with and leverage the Gemini language model for text generation.

Tech Stack

- **Programming Language:** Python
- **Framework:** Flask
- **API:** Google Gemini 2.0 Flash API
- **Libraries:** `requests`, `flask_cors`, `dotenv`
- **Environment Variables:** `.env` file for API key management

Installation & Setup

Prerequisites:

1. **Python 3.7+:** Ensure Python is installed on your system.
2. **Pip:** Make sure pip (Python package installer) is installed.
3. **Virtual Environment (Recommended):** Create a virtual environment to isolate project dependencies.

```
python3 -m venv venv
source venv/bin/activate  # On Windows: venv\Scripts\activate
```

Installation Steps:

1. **Clone the repository:**

   ```
   git clone <repository_url>
   ```

2. **Navigate to the project directory:**

   ```
   cd karthik_proj
   ```

3. **Install dependencies:**

   ```
   pip install -r requirements.txt
   ```

4. **Create a `.env` file:** Create a file named `.env` in the project's root directory and add your Gemini API key:

   ```
   GEMINI_API_KEY=YOUR_GEMINI_API_KEY
   ```

   Replace `YOUR_GEMINI_API_KEY` with your actual API key. Obtain this key from the Google Cloud Console after setting up a Gemini project.

Usage Guide

Running the application:

1. **Start the Flask development server:**

   ```
   python app.py
   ```

2. **Send a POST request:** The application accepts POST requests to the `/generate` endpoint with a JSON payload containing the prompt.

   Example using curl:

   ```
   curl -X POST -H "Content-Type: application/json" -d '{"prompt": "Write a short poem about a cat."}'
   http://127.0.0.1:5000/generate
   ```

   Expected Response (JSON):

   ```
   {
     "generated_text": "A furry friend, a purring sound,\nA whiskered face, upon the ground,\nWith emerald eyes, so softly
   bright,\nA feline grace, a wondrous sight."
   }
   ```

Production Deployment: This example is designed for local development. For production, consider using a WSGI server like Gunicorn or uWSGI and a production-ready web server such as Nginx. You will also need to handle environment variables appropriately in a production setting.

Features

- **Gemini API Integration:** Seamlessly integrates with the Google Gemini 2.0 Flash API for text generation.
- **Error Handling (Retry Logic):** Includes retry mechanism for API requests.
- **CORS Enabled:** Allows cross-origin requests for easier integration with front-end applications.
- **Environment Variable Support:** Uses environment variables for API key security.

Codebase Walkthrough

Folder Structure:

- **app.py:** The main application file containing the Flask app and Gemini API interaction logic.
- **.env:** (Hidden file) Stores the Gemini API key. This file should be added to your `.gitignore` file.

- **requirements.txt:** Lists project dependencies for easy installation.

Important Files:

- **`app.py`:** Contains the Flask application setup, route definition (`/generate`), and the function `call_gemini` which makes the request to the Gemini API, includes retry logic, and handles the response.

API Documentation

Endpoint: `/generate`

Method: `POST`

Request:

```
{
  "prompt": "Your text prompt here"
}
```

Response (Success - 200 OK):

```
{
  "generated_text": "The text generated by Gemini API"
}
```

Response (Error - e.g., 500 Internal Server Error):

```
{
  "error": "An error occurred"
}
```

Contribution Guide

1. **Fork the repository:** Create a fork of the project on your GitHub account.
2. **Clone your fork:** Clone your forked repository to your local machine.
3. **Create a branch:** Create a new branch for your changes. For example: `git checkout -b feature/add-new-feature`.
4. **Make your changes:** Implement your changes and commit them.
5. **Push your branch:** Push your branch to your forked repository.
6. **Create a pull request:** Create a pull request from your branch to the original repository's `main` branch.

Coding Style: Follow PEP 8 Python style guidelines.

License & Credits

This project is currently unlicensed. Credits to Google for the Gemini API. The project utilizes the following libraries: `requests`, `flask`, `flask_cors`, and `python-dotenv`.