AI-Powered Python Code Reviewer - Project Documentation

1. Introduction

This project is an AI-powered code review app built using Python and Streamlit, integrated with Google's Gemini AI model via LangChain. It enables users to paste or upload `.py` files and receive detailed code reviews automatically.

2. Objective

To provide an intuitive web-based interface for Python developers to get instant feedback on their code, covering:

- Bug detection
- Best practices
- Performance optimizations
- Security vulnerabilities
- Beginner-friendly suggestions

3. Libraries & Technologies Used

3.1 Streamlit:

- Used to build the interactive web UI.
- Functions used include text area, file uploader, and buttons.

3.2 LangChain + Google Generative AI:

- ChatGoogleGenerativeAI for accessing Gemini.
- Prompt and output parser to handle input/output.

3.3 dotenv:

- Loads environment variables from a .env file for security.

3.4 OS:

- Accesses environment variables in code.

4. Environment Setup

1. Install required packages:

   pip install streamlit python-dotenv langchain langchain-google-genai

2. Create .code_reviewer_env file:

   GOOGLE_API_KEY=your_google_api_key_here

3. Ensure access to Gemini API via Google.

5. Application Workflow (Step-by-Step)

1. Load API Key using dotenv and os.
2. Set up Streamlit UI: title, input area, and file uploader.
3. Read user code input from file or text box.
4. On button click, validate API key and code presence.
5. Use a structured prompt to instruct the LLM.
6. Initialize Gemini model with LangChain.
7. Chain prompt, model, and output parser.
8. Invoke the model with user code.
9. Display markdown-formatted feedback.

6. Code Structure & Explanation

Code is organized into logical blocks:

- Environment loading
- UI creation with Streamlit
- File handling logic
- LLM pipeline using LangChain
- Exception handling for robustness

7. Running the Application

1. Ensure environment is active and dependencies installed.
2. Run the app:

   streamlit run app.py

3. Open browser and paste or upload your Python code for review.

8. Future Improvements

- Add multi-language support
- Maintain review history
- Use code editors for inline suggestions
- Highlight buggy lines in the code

9. Conclusion

This app simplifies the code review process by leveraging advanced LLM capabilities. It's beginner-friendly, efficient, and easily extensible.