# Log Analyzer Using OpenAI API and LangChain Framework

This project analyzes log files by utilizing the OpenAI API to identify patterns, anomalies, or critical errors. It uses the LangChain framework to integrate with the OpenAI Large Language Model (LLM) for natural language processing.

## Project Setup

### Prerequisites

Before running the Log Analyzer, ensure you have the following installed:

* Python 3.7 or higher
* pip (Python package installer)
* OpenAI API Key (you can sign up and get an API key from [OpenAI's platform](https://platform.openai.com/signup))

### Step 1: Install Required Libraries

Install the necessary Python packages using pip.

pip install langchain openai unittest.mock

### Step 2: Set Your OpenAI API Key

You will need your OpenAI API key to access the API. You can set it as an environment variable to keep it secure.

openai\_api\_key = "your-openai-api-key"

### Step 3: Add Your Log Files

Place your .log files into a folder. For example, create a folder logs/ in the project directory and place your log files there.

In the LogAnalyzer.py script, update the log\_folder\_path variable to point to the directory where your logs are stored.

log\_folder\_path = "path\_to\_your\_log\_folder" # Example: "logs/"

### Step 4: Toggle Between Mock and Real API Usage

The project allows you to switch between **mock testing** (to avoid API rate limits) and **real API usage**.

#### For Mock Testing (No API Calls)

Set testing\_mode = True in the LogAnalyzer.py file.

python

if \_\_name\_\_ == "\_\_main\_\_":

testing\_mode = True # Use mock API calls

#### For Real API Calls

Set testing\_mode = False in the LogAnalyzer.py file to use the OpenAI API for real analysis.

python

if \_\_name\_\_ == "\_\_main\_\_":

testing\_mode = False # Use real OpenAI API

### Step 5: Run the Log Analyzer

Run the Python script to analyze the logs. The script will either simulate the analysis (in mock mode) or use the real OpenAI API.

python LogAnalyzer.py

### Step 7: Check the Analysis Report

After the script completes, it will generate an analysis report and save it to log\_analysis\_report.txt in the project directory.

### Key Components

* **Log Analyzer Code**: Located in LogAnalyzer.py.
* **Log Files**: Place your logs in the folder specified by log\_folder\_path.
* **Mocking**: In testing mode, mocked API calls simulate OpenAI responses.
* **Exponential Backoff**: The real API calls implement retry logic with delays to handle rate limits effectively.