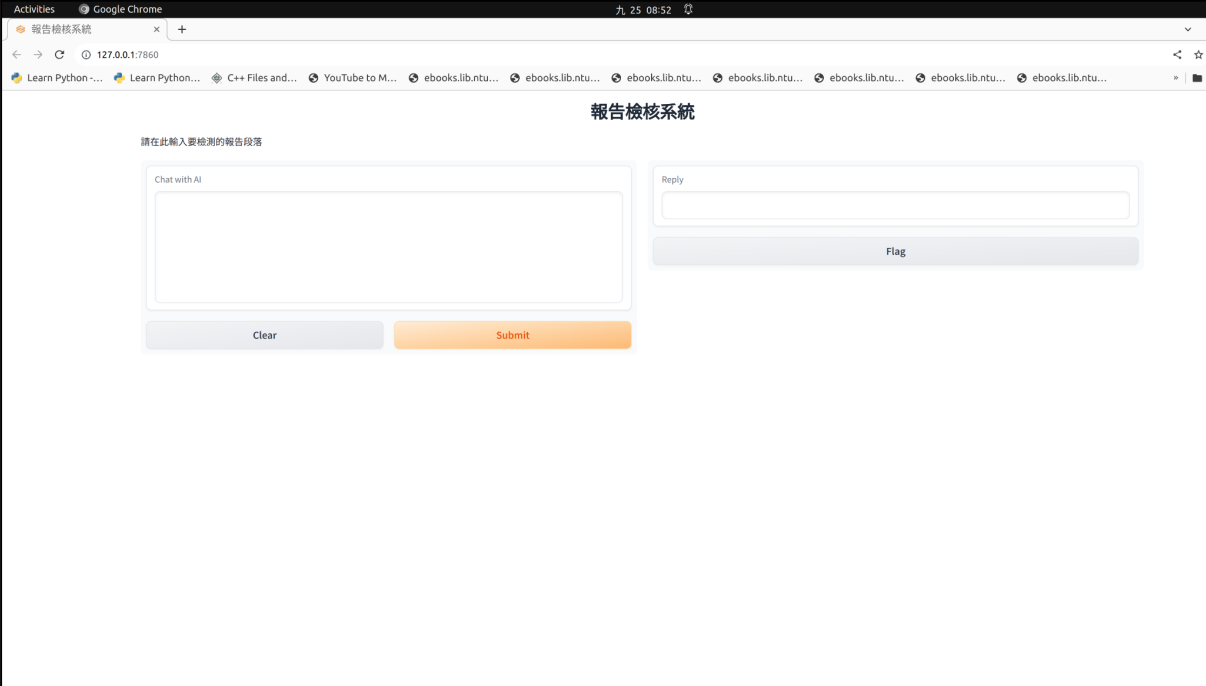


## 網頁說明




報告檢核系統，使用者從左方輸入一段待檢核的文字，確認後【Submit】。接下來等待 chatgpt 做回覆。對話框隨著文字多寡，長度大小會有所改變。

【Flag】為使用者對 openAI 回覆有所疑問時，可使用按鈕將雙方對話內容傳送至 server 端，從 flagged/log.csv 中可以查看。

目前報告檢核系統為獨立存在，需將程式碼合併至系統頁面 competition/view.py 最底下部分。



## 程式說明

<input type="checkbox"/> 名稱	修改日期	類型	大小
 flagged	2023/6/20 下午 01:41	檔案資料夾	
 api	2023/5/26 下午 10:02	檔案	1 KB
 test	2023/9/25 上午 01:23	PY 檔案	1 KB

api: OpenAI 的 API 金鑰

flagged: log.csv 為存取對話框內容文件

`test.py`: 利用 OpenAI 的 GPT-3.5-turbo 模型來進行對話生成

- `import gradio as gr` 一個用於建立網頁界面的 library
- `openai.api_key` 設定 OpenAI 的 API 金鑰, 向 OpenAI 提交請求
- `messages` 對 openAI 設定人設, 或是設定 openAI 的回覆方式
- `def chatbot()`: 處理使用者的輸入並生成回覆, 使用 GPT-3.5-turbo 模型回覆
- `inputs = ...` 一個文字框, 用於接收使用者的輸入
- `outputs = ...` 一個文字框, 為 openAI 生成的回覆
- `gr.Interface()` 用此套件 Gradio 來生成介面
- `launch()` 啟動 Gradio 介面並可以分享介面

```
1 import openai
2 import gradio as gr
3
4 #openai.api_key = ""
5 #openai.api_key = ""
6 openai.api_key = "sk-X2oSWmHbq39fWAvGzhNrT3B1bkFJ0kDef16VWAQYkL7VTSvR"
7
8 messages = [
9     {"role": "system", "content": "Counting his current word count and warn if it is less that 500 words."},
10    {"role": "system", "content": "Using traditional chinese to warning."},
11    #{"role": "system", "content": "Giving some feedback on user content."},
12]
13
14 def chatbot(input):
15     if input:
16         messages.append({"role": "user", "content": input})
17         chat = openai.ChatCompletion.create(
18             | model="gpt-3.5-turbo", messages=messages
19         )
20
21         reply = chat.choices[0].message.content
22         messages.append({"role": "assistant", "content": reply})
23
24         return reply
25
26 inputs = gr.inputs.Textbox(lines=7, label="Chat with AI")
27 outputs = gr.outputs.Textbox(label="Reply")
28
29 gr.Interface(fn=chatbot, inputs=inputs, outputs=outputs, title="報告檢核系統", description="請在此輸入要檢測的報告段落", theme="compact").la
30 #compact
```