**「final\_exam\_report」**

**軟體開發文件**

課程名稱：Python程式設計與實務應用\_ZZZ002

組名：Pythonary -第 2組

組長： 112221476譚培成

組員： 112270042李能安、113194172許佳語、113170501謝順隆、113294207于樂洋113170134董亦辰

指導老師：賴素純 老師

中華民國114年5月

**目錄**

**「final\_exam\_report」作業分工說明 .............................................................. 2**

**1. 專案簡介 .................................................................................................. 3**

**2. 開發環境 .................................................................................................. 3**

**3. 系統開發流程 ........................................................................................... 3**

**4. 程式設計主要技巧 ................................................................................... 5**

**5. 原始碼重點解說 ....................................................................................... 6**

**6. 套件使用 .................................................................................................. 7**

**7. 系統文件結構概述 ................................................................................... 7**

**「Pythonary Lookup」作業分工說明**

**1. 開發程式 .............................................................................................. 于樂洋**

**2. 成員協調................................................................................................ 譚培成**

**3. 開發文件、外觀設計 ........................................................................... 許佳語**

**4. 使用手冊、交互邏輯規劃 .................................................................... 董亦辰**

**5. 投影片製作 ……………………………………………………………………………………..… 李能安**

**6. 視訊報告 .............................................................................................. 謝順隆**

**1. 專案簡介**

• 專案名稱：互動式格林童話

• 專案目標：開發一個能夠根據使用者輸入，提供童話主角進行選擇分支任務，促進閱讀童話興趣的應用程式。

**2. 開發環境**

IDE：Visual Studio Code  
Python版本：3.13

**3. 系統開發流程**

**• (1) 訂定主題**

這是開發的第一步，旨在確定應用的主要目標和功能。我們集思廣益，提出了多種功能想法，並根據用戶需求選定了最優解。這個過程中我們還需要考慮應用的使用場景和潛在用戶。

**• (2) 初期規劃**

在此階段，我們制定了項目的整體計劃，包含主要的時間線、開發分工及資源分配等。同時確立了技術棧（如選擇 Python 和 tkinter），並評估應用程式的技術可行性。

**• (3) 設計互動邏輯**

設計互動邏輯是為了確保使用者操作時，能更有興趣的投入童話故事情節中。例如，當使用者選擇了不同分支時可能發生的不同結局，應用程式如何將選擇結果從OpenAI獲取並展示給使用者。我們針對各種使用情境進行了流程圖的繪製，確保使用者體驗流暢無誤。

**• (4) 外觀設計**

使用 tkinter 套件設計應用的外觀，包括按鈕、內容顯示區域等，旨在讓使用者界面簡單直觀，方便所有年齡層的用戶使用。色彩搭配和佈局設 計也在此階段進行，以提升視覺效果和使用者的舒適度。

**• (5) 程式開發**

程式開發是根據前面的需求與設計文件，將每個模組逐步實作出來。這裡分別進行資料模組（Model）、視圖模組（View）及邏輯處理（ViewModel） 的實作，同時確保模組間的協作運作正常。

**• (6) 功能驗收**

每個功能完成後，我們進行功能驗收，以確保各個模組達到預期的效果。例如，確認選擇結果的連貫性及顯示的正確性等。

**• (7) 測試**

在開發完成後，進行全面的測試，包括單元測試和整合測試。單元測試確保每個模組獨立運作正常，而整合測試則確保各模組之間協作順暢。用戶體驗測試也是其中的重要一環，以確認應用程式符合使用者的需求。

**• (8) 修正錯誤**

測試過程中發現的錯誤會立即進行修正，並重新測試以確保問題解決。我們採用迭代式的方法持續修正問題，直到系統穩定。

**• (9) 開發完成**

完成所有功能的開發和測試，確保系統的每個部分都能穩定運行後，我們進行最終的驗收，並準備部署。

**4. 程式設計主要技巧**

**• (1) 外觀部分使用tkinter套件：**

tkinter 是 Python 內建的 GUI 工具包，用來設計使用者界面。透過 tkinter，我們能快速構建應用程式的視覺部分，例如選擇按鈕、顯示區域等，由於 tkinter 是內建的，所以它具有高度兼容性，適合快速建立和測試 GUI。

**• (2) 資料部分使用MVVM架構：**

A. MVVM（Model-View-ViewModel）架構將應用程式分為三個主要部分，以提高可維護性和可擴充性。這種架構的主要目的是將應用程式的邏輯與視圖分離，使代碼更加易於管理和測試。

(A) Model（模型）： Model 層負責管理應用程式的數據和商業邏輯。它與應用的資料來源互動，例如從 API 獲取數據，並將數據處理後提供給 ViewModel。這樣的設計確保了數據與應用的顯示分離，方便數據邏輯的獨立測試和修改。

(B) View（視圖）： View 是應用程式的使用者界面，負責顯示資料並與使用者互動。它接收來自 ViewModel 的數據並進行展示，所有與界面 的互動（例如按鈕點擊）都會由 View 來處理，並 傳送給 ViewModel。這確保了界面的變更不會影響底層邏輯。

(C) ViewModel（視圖模型）： ViewModel 作為 View 與 Model 之間的橋樑，負責處理來自 Model 的資料並進行格式化，以便 View 能夠顯示。它還處理 使用者的輸入，並將相應的變更回饋給 Model。ViewModel 通 常會透過資料綁定與 View 進行互動，這意味著當 Model 中的 數據變動時，View 也會自動更新，實現了更好的動態互動。

B. View 和 ViewModel 使用 CallBack 來綁定資料變動事件 這種綁定方式確保了在資料變更時，界面可以即時更新，用戶的操作 也能及時反饋到數據層。

**5. 原始碼重點解說**

**(1) main.py：**

程式主要為啟動主畫面及綁定 ViewModel 到 View

**(2) apikey.py：**

儲存api 金鑰

**(3) ApiService.py：**

這模組主要功能為根據格林童話（排除小紅帽）隨機生成一個文字冒險故事，並將故事分為五段，每段提供三個選項讓使用者互動選擇，透過 OpenAI 的 GPT 模型生成 JSON 格式的回應，並持續接續故事直到結尾為止（最後一段不再提供選項）。

**(4) MainView.py：**

這個程式是用 Tkinter 建立的童話文字介面，具有以下功能：顯示故事內容並提供三個選項讓使用者選擇，根據使用者的選擇更新故事內容，直到故事結束，當故事結束後，顯示一個“下一個故事”按鈕，讓使用者開始新的故事。

**(5) Message.py：**

這模組主要為定義了 OpenAI的角色（系統、助手、使用者）。Message 類別用來表示一條訊息，包括角色和內容，並提供將訊息轉換為字典的功能，還可以從 GPT 回應中創建 Message 物件。

**(6) StoryScene.py：**

這段模組為定義 StoryScene 類別，用來表示故事的一個場景：StoryScene 包含標題、內容、選項列表和是否結束的狀態。提供一個靜態方法 from\_json，可以將 JSON 字串轉換為 StoryScene 物件，方便處理故事數據。

**(7) ViewModel.py：**

這模組定義 ViewModel 類別，負責管理遊戲故事的邏輯：bind\_to 方法將視圖的回調函數綁定到模型，當故事更新時觸發回調。start\_story 方法開始新故事並呼叫 API 獲取故事內容。select\_index 方法處理用戶選擇，將選項發送給 API 並更新故事。故事的狀態（如結束與否）會在每次更新後通知視圖。

**6. 套件使用**

• tkinter：Python內置的UI套件，不需要額外安裝，用於構建應用程式的用戶界面。這個套件簡單且功能強大，適合進行快速的UI開發。

• dataclass\_wizard：用於JSON物件的解析和處理，尤其在處理API返回的複雜結構時，可以將其轉化為Python的資料類別，便於操作和管理。

• requests：用於發送HTTP請求，以與OpenAI的API互動。requests是一個簡單而強大的HTTP請求工具，可以輕鬆發送GET和POST請求，並處理返回的資料。

• 其他依賴套件：根據應用的需要，還安裝了一些額外的依賴套件，以確保所有功能模組的正常運作。

**7. 系統文件結構概述**

**• (1) 功能需求規劃：**

• 確定應用程式的主要功能，包括選項按鈕點擊、與API的交互等。

• 通過多次會議，列出了應用程式的核心功能需求，並確保這些需求能夠滿足使用者的主要使用場景。

**• (2) 系統架構設計：**

• 使用MVVM架構來劃分系統的各個模組，這樣可以有效分離資料層與顯示層，並確保各部分之間的責任單一且清晰。

• 這樣的架構使得代碼的可維護性大大提升，尤其在後期功能擴充和修改時，能夠保持高效和簡便。

**• (3) 視覺與互動設計：**

• 視覺設計部分主要通過tkinter來完成。

• 確保每一個UI元素都符合直觀性和易用性的原則，例如查詢按鈕的大小和位置、結果顯示的清晰度等，這些都是經過多次測試和優化後確定的。

• 此外，還考慮了使用者的不同需求，增加了多個分支路線的選擇功能，讓用戶能更有參與的的感受。

**• (4) 測試與驗收：**

• 在測試過程中，進行了多層次的測試，包括功能測試、用戶體驗測試以及性能測試。

• 功能測試確保應用的每個部分按預期工作，用戶體驗測試則重點在於使用者操作的便捷性和整體使用感受。

• 性能測試主要是檢測應用在不同網絡環境下的表現，特別是在與API交互時的響應速度。

**• (5) 部署與維護：**

• 在應用完成開發和測試之後，進行了部署。

• 部署過程中選擇了一個穩定的Python運行環境，並對可能的依賴問題進行了測試和解決。

• 未來的維護計劃包括持續的功能更新和錯誤修復，並根據使用者的反饋對應用進行改進和優化。