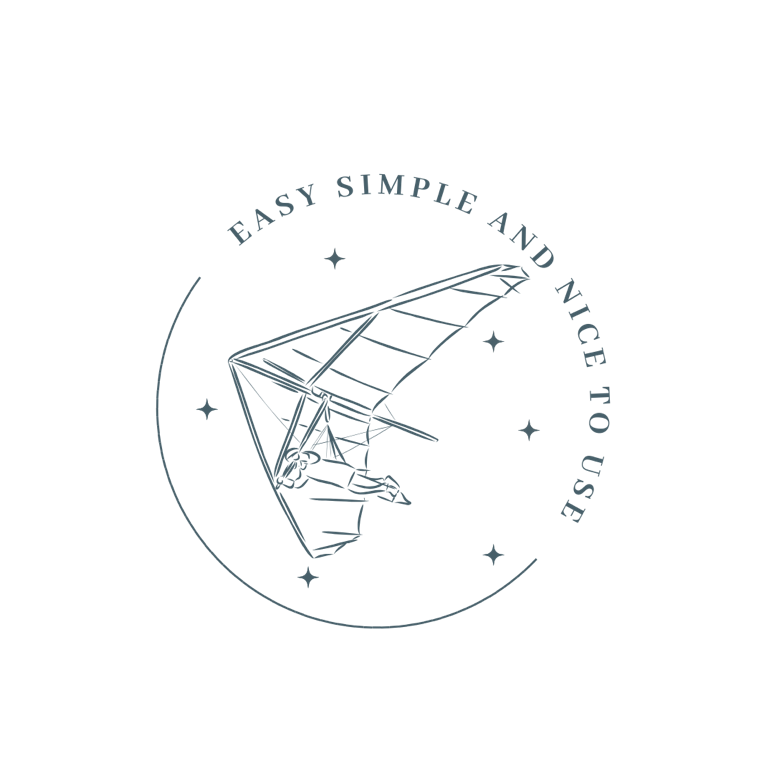
國立臺北商業大學

資 訊 管 理 系

113資訊系統專案設計

系統手冊



組 別：第113507組

題 目：履人

指導老師：葉明貴老師

組 長：10956023 廖翊丞

組 員：10956005林東建 10956015 張庭瑋

10956029李宜庭 10956041 林依誼

中華民國113年5月27日

目錄

目錄 i

圖目錄 iii

表目錄 iv

第 1 章 前言 1

1-1 背景介紹…………………………………………………………………………1

1-2 動機………………………………………………………………………………1

1-3 系統目的與目標…………………………………………………………………1

1-4 預期成果…………………………………………………………………………1

第 2 章 營運計畫 2

2-1 可行性分析……………………………………………………………………2

2-2 商業模式-Businessmodel………………………………………………………4

2-3 市場分析-STP…………………………………………………………………4

2-4 競爭力分析SWOT-TOWS分析………………………………………………6

第 3 章 系統規格 7

3-1 系統架構………………………………………………………………………….7

3-2 系統軟、硬體需求與技術平台…………………………………………………7

3-3 開發標準與使用工具……………………………………………………………8

第 4 章 專案時程與組織分工 10

4-1 專案時程………………………………………………………………………10

4-2 專案組織與分工………………………………………………………………11

第 5 章 需求模型 14

5-1 功能分解圖(Functionaldecompositiondiagram)……………………..………….14

5-2 需求清單……………………………..…………..…………………..………….15

第 6 章 程序模型 16

6-1 資料流程圖(Dataflowdiagram)………………….…………………..………….16

6-2 程序規格書(Processspecification)…...………….…………………..………….21

第 7 章 資料模型 25

7-1 實體關聯圖(Entityrelationshipdiagram)………….…………………..………...25

7-2 資料字典(Datadictionary)………………..……….…………………..………...26

第 8 章 資料庫設計 27

8-1 資料庫關聯圖 ……………………………………….………………..………...27

8-2 表格及其Metadata …………………………………….………………………..28

第 9 章 程式 27

9-1 資料庫關聯圖 ……………………………………….…………………….…...27

9-2 程式規格描述 ……………………………………….…………………….…...27

第 10 章 測試模型 27

10-1 測試計畫：說明採用之測試方法及其進行方式…….……………………….27

10-2 測試個案與測試結果…….………………………..……….……….………….27

第 11 章 操作手冊 27

第 12 章 使用手冊 27

第 13 章 感想 27

第 14 章 參考資料 27

附錄 27

圖目錄

圖2-1-1 系統可行性分析…………………………………………………………… 2

圖2-2-1商業模式………………………………………………………………4

圖2-3-1 市場分析 STP……………………………………………………………4

圖3-1-1 系統架構圖(Json response)………………………………………………… 7

圖4-1-1 專案時程……………………………………………………………………10

圖 5-1-1 功能分解圖…………………………………………………………………14

圖6-1-1系統環境圖…………………………………………………………………16

圖6-1-2 DFD圖0……………………………………………………………………17

圖6-1-3 DFD圖1…………………………………………………………………… 18

圖6-1-4 DFD圖2…………………………………………………………………… 19

圖6-1-5 DFD圖3………………………………………………………………………20

圖7-1-1 實體關聯圖………………………………………………………………… 25

圖8-1-1 資料關聯圖………………………………………………………………… 27

表目錄

表2-4-1 SWOT-TOWS 分析……………………………………………………………6

表3-2-1 技術平台與軟體需求…………………………………………………………7

表 3-2-2伺服器硬體……………………………………………………………………8

表3-2-3 使用者硬體……………………………………………………………………8

表3-3-1 開發工具……………………………………………………………………8

表4-2-1 組織分工表…………………………………………………………………11

表5-1-1 功能性需求…………………………………………………………………14

表6-2-1 程序規格……………………………………………………………………21

表7-2-1 字典清單……………………………………………………………………26

表8-2-1 使用者資料…………………………………………………………………28

表8-2-2 專案…………………………………………………………………………29

表8-2-3 學生………………………………………………………………………29

表8-2-4 評分…………………………………………………………………………30

表8-2-5 協作者………………………………………………………………………30

表8-2-6 學生分配……………………………………………………………………31

表 8-2-7 自傳……………………………………………………………………32

表8-2-8 履歷…………………………………………………………………………33

表 8-2-9 學生相關內容備註……………………………………………………34

表8-2-10 使用者異動……………………………………………………………35

附錄表一…………………………………………………………………………

附錄表二……………………………………………………………………………

第 1 章 前言

1-1 背景介紹

隨著科技迅速發展，大專院校書審流程卻依舊涉及大量手動作業，其中助教和老師需花費大量時間歸納和審核求學者的資料。這個過程不僅效率低下，也容易出錯，尤其是在高壓力和時間壓縮的環境中。為了解決這些問題，我們開發了「履人」，幫助審核過程的管理變得更為簡便和高效。

1-2 動機

現在專四，我們不久後即將畢業，在開始需要準備書審資料的時候，想到評審們要面對大量四面八方的書面資料，如何使自己在眾多申請者中脫穎而出成為一個重要的課題。在這個階段，老師們也面臨著如何有效地找到申請者的姓名以及相關內容的挑戰。在思考解決這個題目的過程中，我們意識到程式設計的目的是為了在有限的時間內讓人們能夠更有效率地完成任務。因此，我們計畫開發一個軟體，以解決老師們在面試過程中遇到的問題。

1-3 系統目的與目標

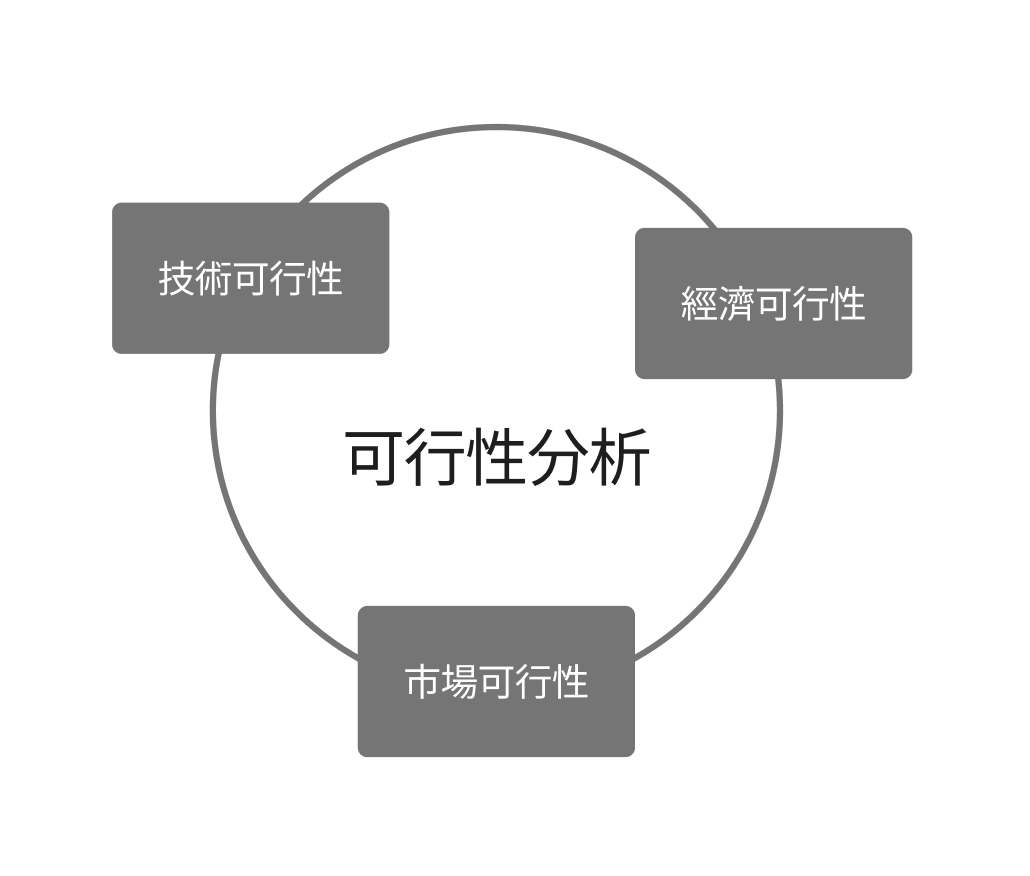
* 可以更有效地協助處理大量的備審資料
* 成為老師們在招生過程中的得力助手，使整個過程更加順暢、高效。

1-4 預期成果

* 提升審核流程的方便性，讓助教與老師不必花費大量時間在無關審核的流程上
* 協助目前資管系的審核流程

第 2 章 營運計畫

2-1 可行性分析



▲ 圖2-1-1 系統可行性分析

技術可行性：

本專題涉及多種技術來實現開發，目前所具備的相關技能有：了解評估系統架構、繪製 DFD、FDD，也了解用不同角色去看待系統的實用性(專二上下學期)；學習html及css從使用者角度去完善整個系統的UI、UX(專三上學期、專四上學期)；學習SQL程式語言連接MySQL資料庫，處理相關資料內容(專四上學期)；以及學習python程式語言，用於實現系統後端功能(專一上下學期)。目前還接觸了新的技術，AI深度學習語言模型，學習lama3的中文修改版模型，使用於本系統中。

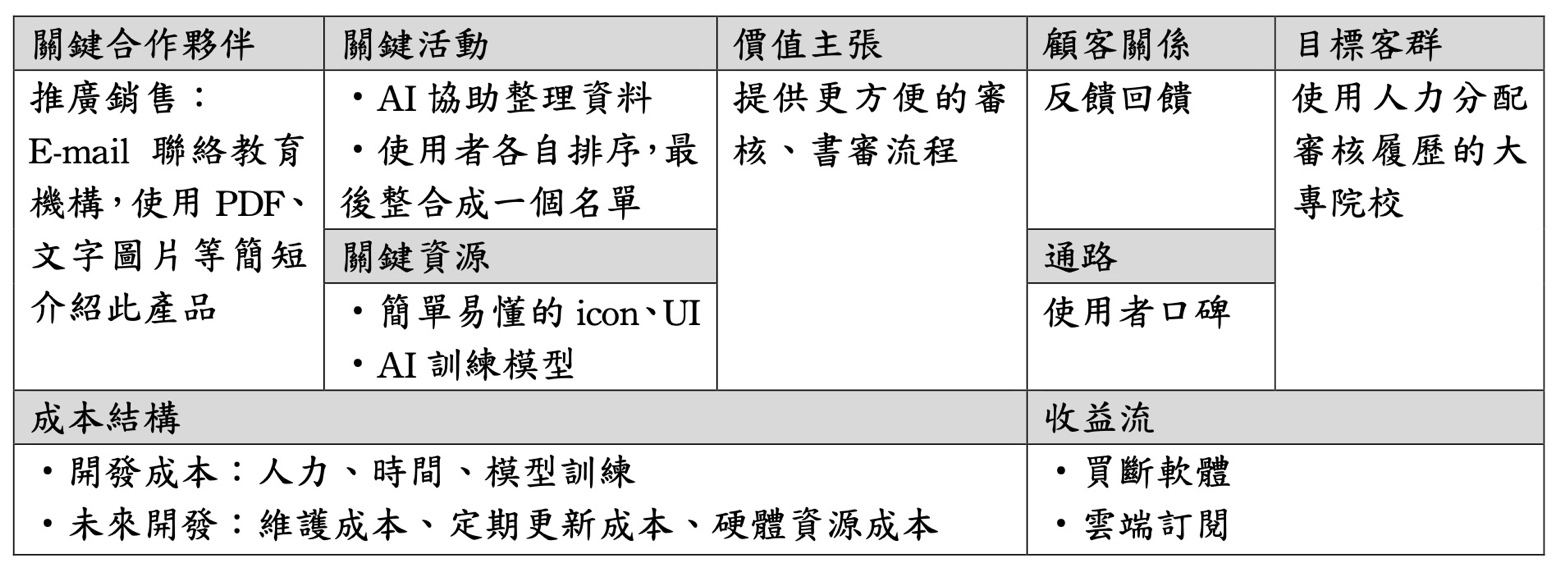
經濟可行性：

每一套系統都是獨立銷售，此做法是為了確保所有數據和資料都能在安全的環境下處理，也能提供買方些微客製化設計和售後問題處理。而此套裝式系統是給予行政人員進行更簡潔的文書管理。無論是小規模教育機構還是大學，需要改善招生流程的機構都適用於此系統。

市場可行性：

每年學校各學制的招生工作通常在相近的時間內進行，因此系上老師需要投入大量時間來審閱並評分學生的履歷、自傳等文件，為了簡化這一過程，我們設計了一套系統，旨在幫助老師們節省時間並提高文件排序的便利性。此系統不僅讓老師在審查和排序工作上更加便捷，也使助教在分享資料、整理和評分方面的工作更為高效，省去了複雜的流程及時間，從而提升整體工作效率。主要的推廣客群是各學校的招生相關人員，他們將直接受益於我們系統帶來的便利和效率提升。

2-2 商業模式－Business model



▲圖2-2-1商業模式

2-3 市場分析－STP



▲圖2-3-1 市場分析 STP

市場區隔：

* + 需要面試審核履歷的大專院校，打造可以幫助老師們的軟體

目標市場：

* + 使用人力整理和分配履歷且需要AI小助手的科系和學校

產品定位：

* + 強調方便性，幫助使用者在審核履歷、面試等方面有更高的便利性、可閱讀性

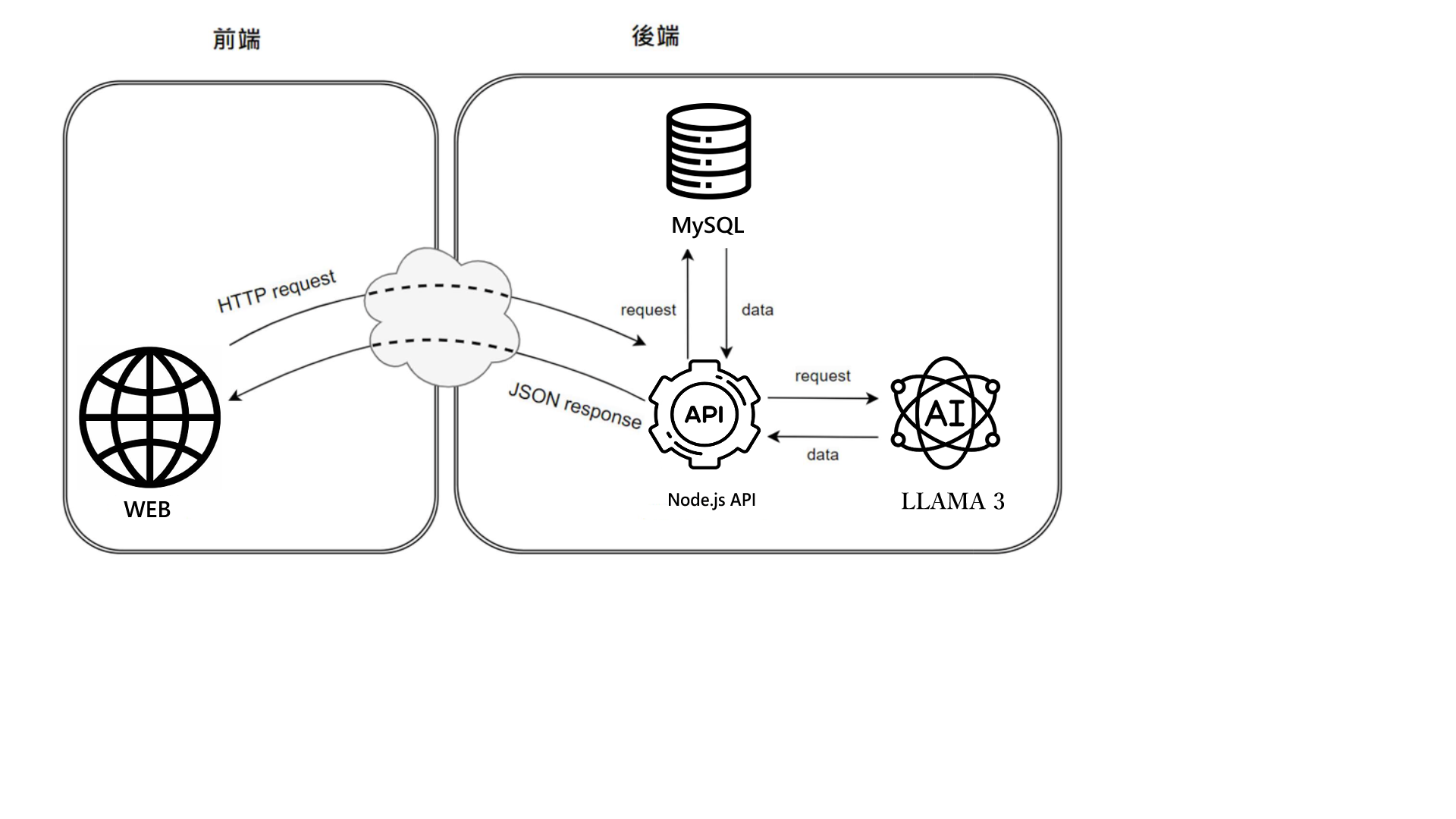
2-4 競爭力分析SWOT-TOWS

▼表 2-4-1 SWOT-TOWS 分析

|  |  |  |
| --- | --- | --- |
| 內部因素  外部因素 | Strengths 優勢   * 專屬的AI小助手 * 精簡的操作流程 | Weaknesses 劣勢   * 客群單一 |
| Opportunities 機會   * 目前市場未出現的管理網站 * 可套用於多所學校或是公司 * 多學制的情況下可以拓展許多履歷模板或是履歷專案樣式 | SO  新型的網站開發能夠套用於多所學校或公司的解決方案，擴大客戶群。 | WO  利用多學制的情況，開發更多履歷模板或專案樣式，可以吸引更多客戶 |
| Threats 威脅   * 網站規模較小容易複製 * 若有體制上得選材改革，會有來不急應對的可能 * 多方軟體文件AI判讀已高度成熟 | ST  利用精美的畫面和簡單的操作流程，吸引客戶增加競爭力 | WT  繼續養成AI 將他訓練成符合需求且穩定 |

第 3 章 系統規格

3-1 系統架構



* 圖3-1-1 系統架構圖(Json response)

3-2 系統軟、硬體需求與技術平台

▼ 表3-2-1 技術平台與軟體需求

|  |  |
| --- | --- |
| 項目 | 描述 |
| 開發語言 | Node.js |
| 機器學習框架 | PyTorch |
| 模型 | LAMA3（來自GitHub） |
| 服務器平台 | WSL 或 Linux |

▼ 表3-2-2伺服器硬體

|  |  |
| --- | --- |
| 項目 | 描述 |
| CPU | 8核心以上 |
| GPU | 6GB VRAM以上 |
| RAM | 32GB以上 |

▼ 表3-2-3 使用者硬體

|  |  |
| --- | --- |
| 項目 | 描述 |
| 網路 | 可連接網際網路的電子產品 |
| 型別 | 桌上型電腦、筆記型電腦、平板 |

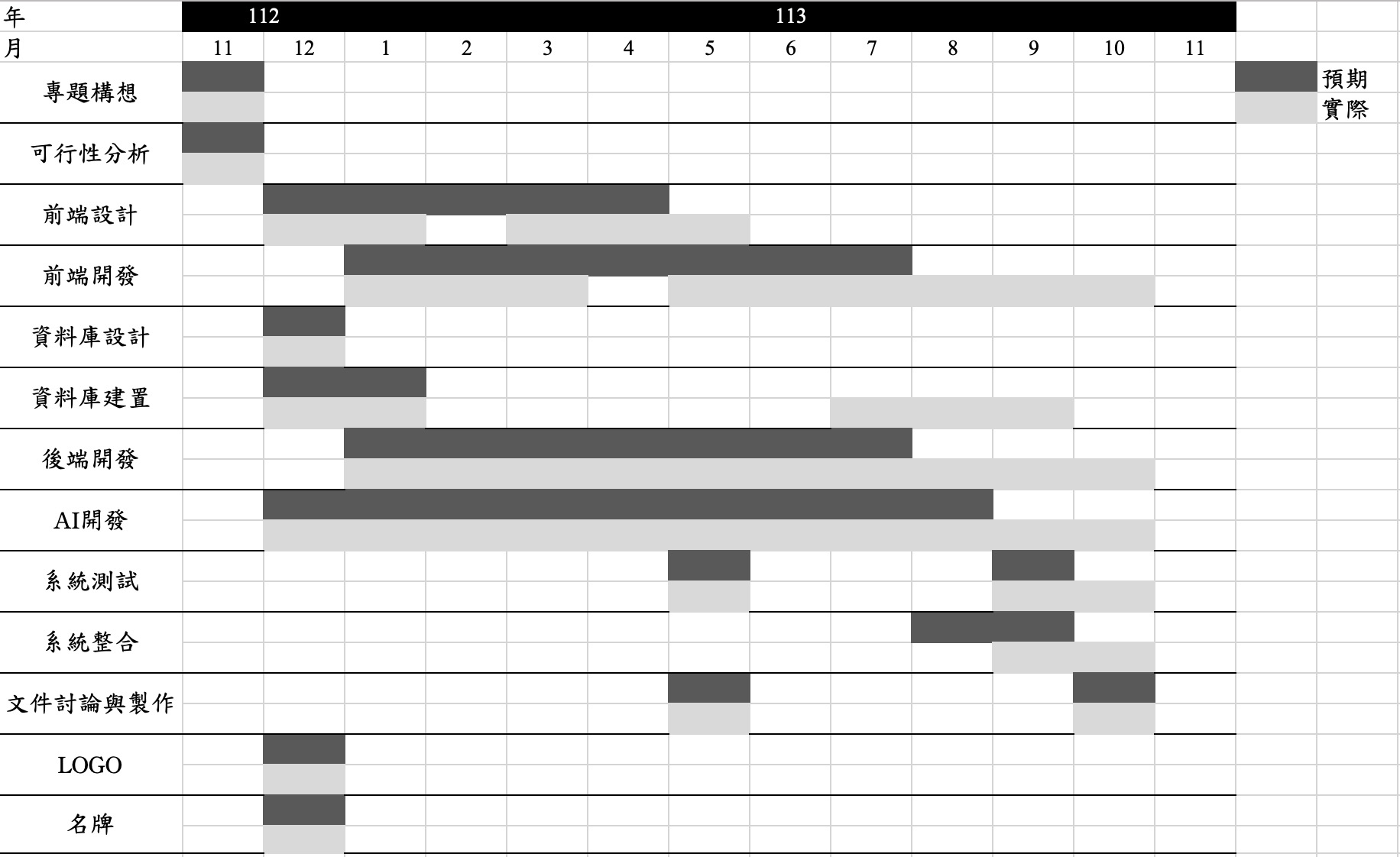
3-3 開發標準與使用工具

▼ 表3-3-1 開發工具

|  |  |  |
| --- | --- | --- |
| 系統開發環境 | | |
| 作業系統 |  | Windows |
| 撰寫工具 | Icon change · Issue #29430 · microsoft/vscode · GitHub Mysql workbench - Social media & Logos Icons | Visual Studio Code、My SQL Workbench |
| 程式開發語言 | | |
| 前端 | react" Icon - Download for free – Iconduck Material Icons - Material-UI Next.js App Router: Setting Up next-intl | React icon、MUI套件、Next.js、typeScript |
| 後端 | Free Python Logo Icon - Free Download Logos Logo Icons | IconScout Free Postman Logo Icon | Node.js、python、Postman |
| 資料庫 | Mysql Logo Vector SVG Icon - SVG Repo | MySQL |
| AI模型 | Llama - Free animals icons | LLAMA3 |
| 介面及插圖繪製工具 | | |
| 介面 | Figma - Free logo icons | Figma |
| 插圖 | Canva icon PNG and SVG Vector Free Download | Canva |
| 文件及美化工具 | | |
| 文件 | Microsoft Word icon PNG and SVG Vector Free Download ChatGPT icon PNG and SVG Vector Free Download | Microsoft Word、Excel、Chat GPT |
| 圖表 | Figma - Free logo icons | Figma |
| 簡報 | Microsoft Powerpoint icon PNG and SVG Vector Free Download | Microsoft Power Point |
| 專案管理及版本控制工具 | | |
| 應用程式 | Fork - a fast and friendly git client for Mac and Windows | Fork |
| 版本控制 |  | Git |
| 系統規劃 | notion" Icon - Download for free – Iconduck | Notion |

第 4 章 專案時程與組織分工

4-1 專案時程



* 圖4-1-1 專案時程

4-2 專案組織與分工

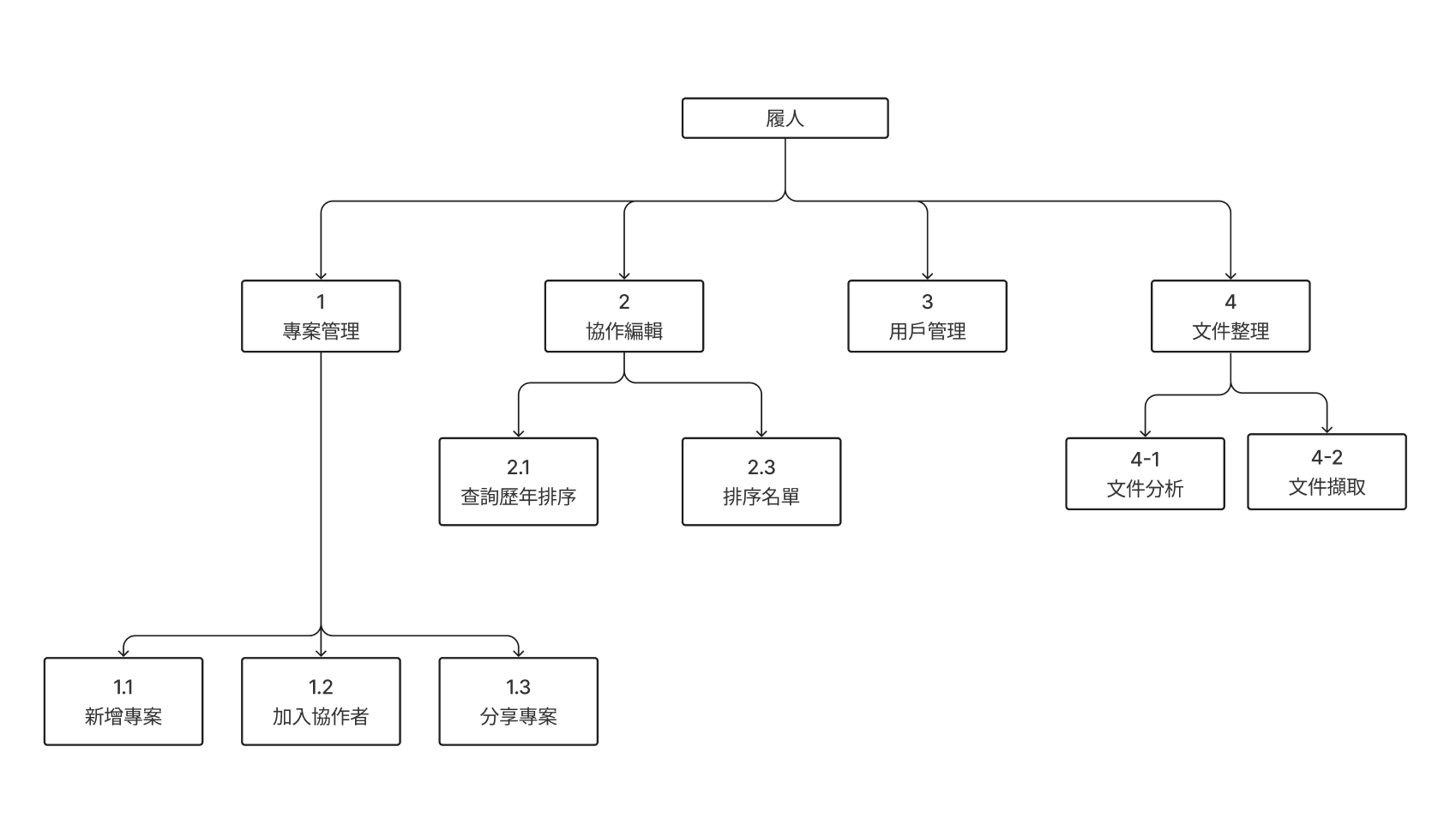
* 表4-2-1 組織分工表

●主要負責人 〇次要負責人

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 項目/組員 | | 10956023/  廖翊丞 | 10956005/  林東建 | 10956015/  張庭瑋 | 10956029/  李宜庭 | 10956041/  林依誼 |
| 後端開發 | 資料庫建置 |  |  | ● |  |  |
| 伺服器架設 |  |  | ● |  |  |
| 登入 |  |  | ● |  |  |
| 專案管理 |  |  | ● |  |  |
| 加入協作者 |  |  |  |  | ● |
| 分享專案 |  |  |  |  | ● |
| 查詢歷年排序 |  |  |  |  | ● |
| 用戶管理 |  |  | ● |  |  |
| 文件備註 |  |  |  |  | ● |
| 排序名單 |  |  | ● |  |  |
| 文件整理 |  | ● |  |  |  |
| AI文件建置 |  | ● |  |  |  |
| 前端開發 | 登入 | ● |  |  |  |  |
| 專案管理 | ● |  |  |  |  |
| 加入協作者 |  |  |  | ● |  |
| 分享專案 | ● |  |  |  |  |
| 查詢歷年排序 |  |  |  | ● |  |
| 用戶管理 |  |  |  | ● |  |
| 文件備註 | ● |  |  |  |  |
| 排序名單 | ● |  |  |  |  |
| 文件整理 | ● |  |  |  |  |
| 美術設計 | UI/ UX | 〇 |  |  | ● |  |
| Web介面設計 | 〇 |  |  | ● |  |
| 色彩設計 |  |  |  | ● |  |
| Logo設計 |  |  |  | ● |  |
| 素材設計 | 〇 |  |  | ● |  |
| 文件撰寫 | 統整 | 〇 |  |  |  | ● |
| 第1章 前言 |  |  |  | ● |  |
| 第2章 營運計畫 |  |  |  |  | ● |
| 第3章 系統規格 | 〇 | ● |  |  |  |
| 第4章 專題時程與組織分工 | ● |  |  |  |  |
| 第5章 需求模型 | ● |  |  |  |  |
| 第6章 程序或設計模型 |  |  |  |  | ● |
| 第7章 資料或實作模型 |  |  | ● |  |  |
| 第8章 資料庫設計 |  |  | ● |  |  |
| 第9章 程式 | ● |  |  |  |  |
| 第10章 測試模型 | ● |  |  |  |  |
| 第11章 操作手冊 | ● |  |  |  |  |
| 第12章 使用手冊 | ● |  |  |  |  |
| 報告 | 簡報製作 |  |  |  |  | ● |

第 5 章 需求模型

5-1 功能分解圖（Functional decomposition diagram）



* 圖5-1-1 功能分解圖

5-2 需求清單

* 表5-1-1 功能性需求

|  |  |
| --- | --- |
| 功能名稱 | 功能敘述 |
| 1. 專案管理 | * 設定專案條件 * 匯入學生資料 * 加入協作者 * 更改專案設定 * 分享專案 |
| 1. 協作編輯 | * 查詢歷年錄取名單 * 排序名單 |
| 1. 用戶管理 | * 使用gmail註冊及登入 * 更改帳號設定 |
| 1. 文件整理 | * AI處理文件顯示給使用者內容摘要 |

非功能性需求

1. 定期備份資料，防止資料丟失

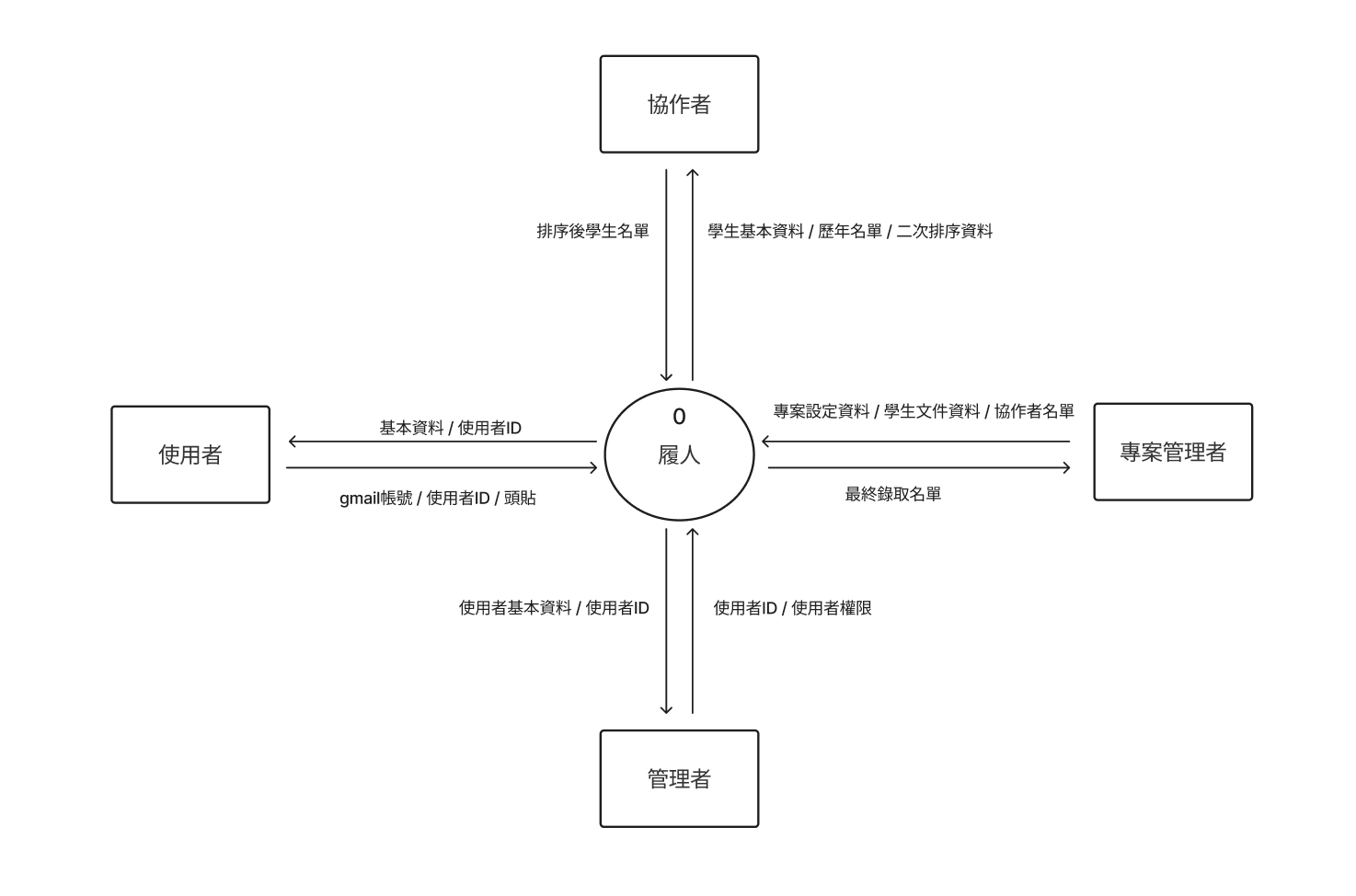
2. 確保適當的權限管理，以防資料丟失

3. 系統應能支援多人同時協作編輯而不降低性能

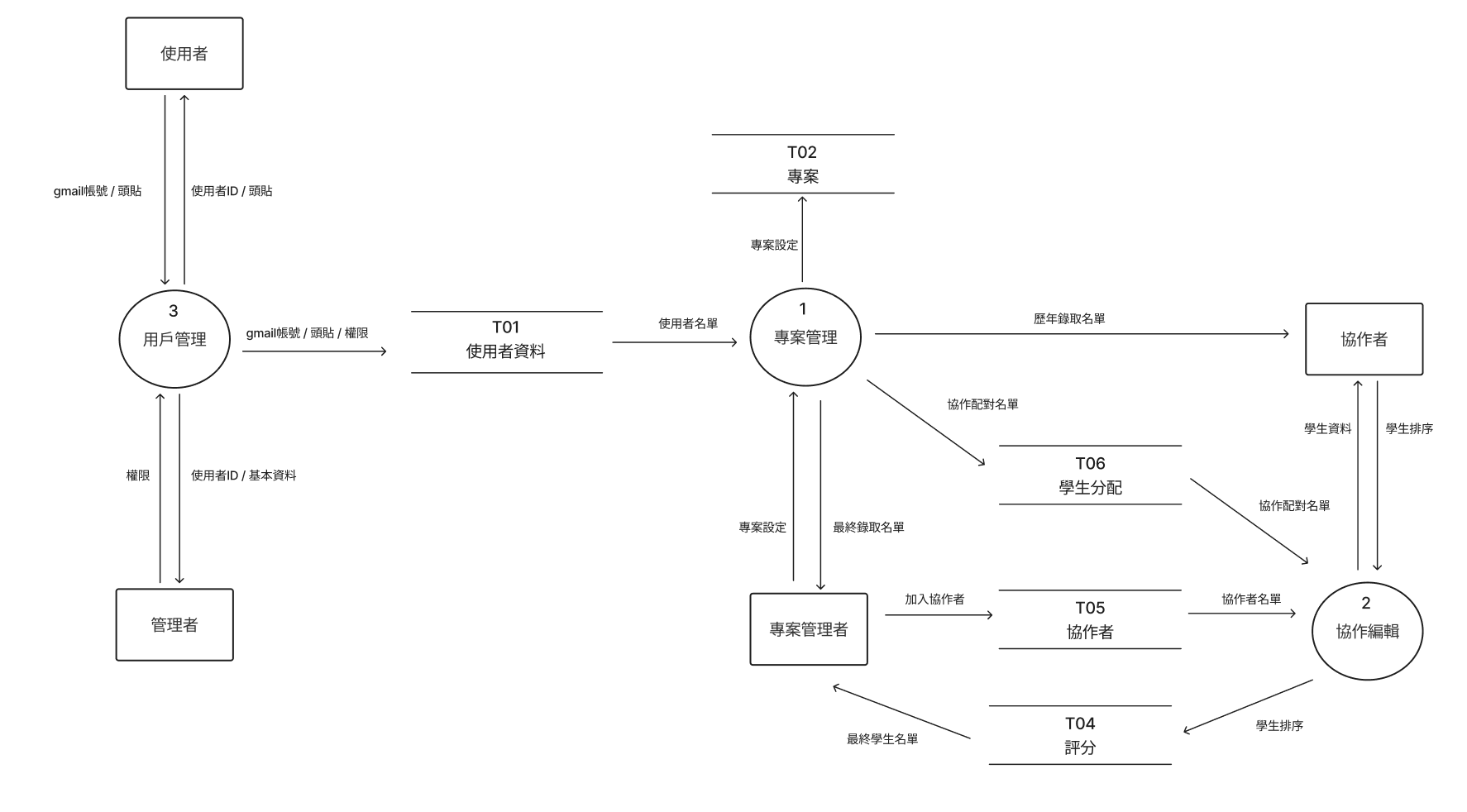
4. 支援特定格式輸入學生資料

第 6 章 程序模型

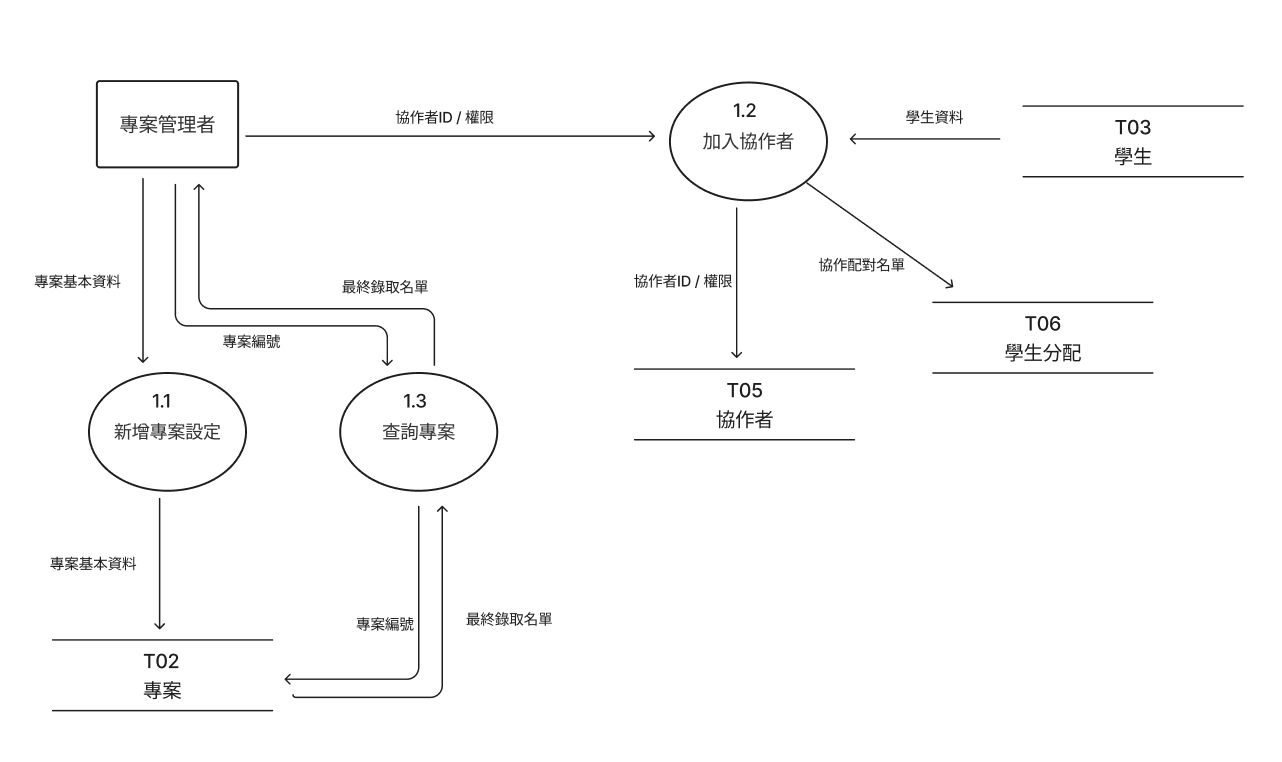
6-1 資料流程圖(Data flow diagram)



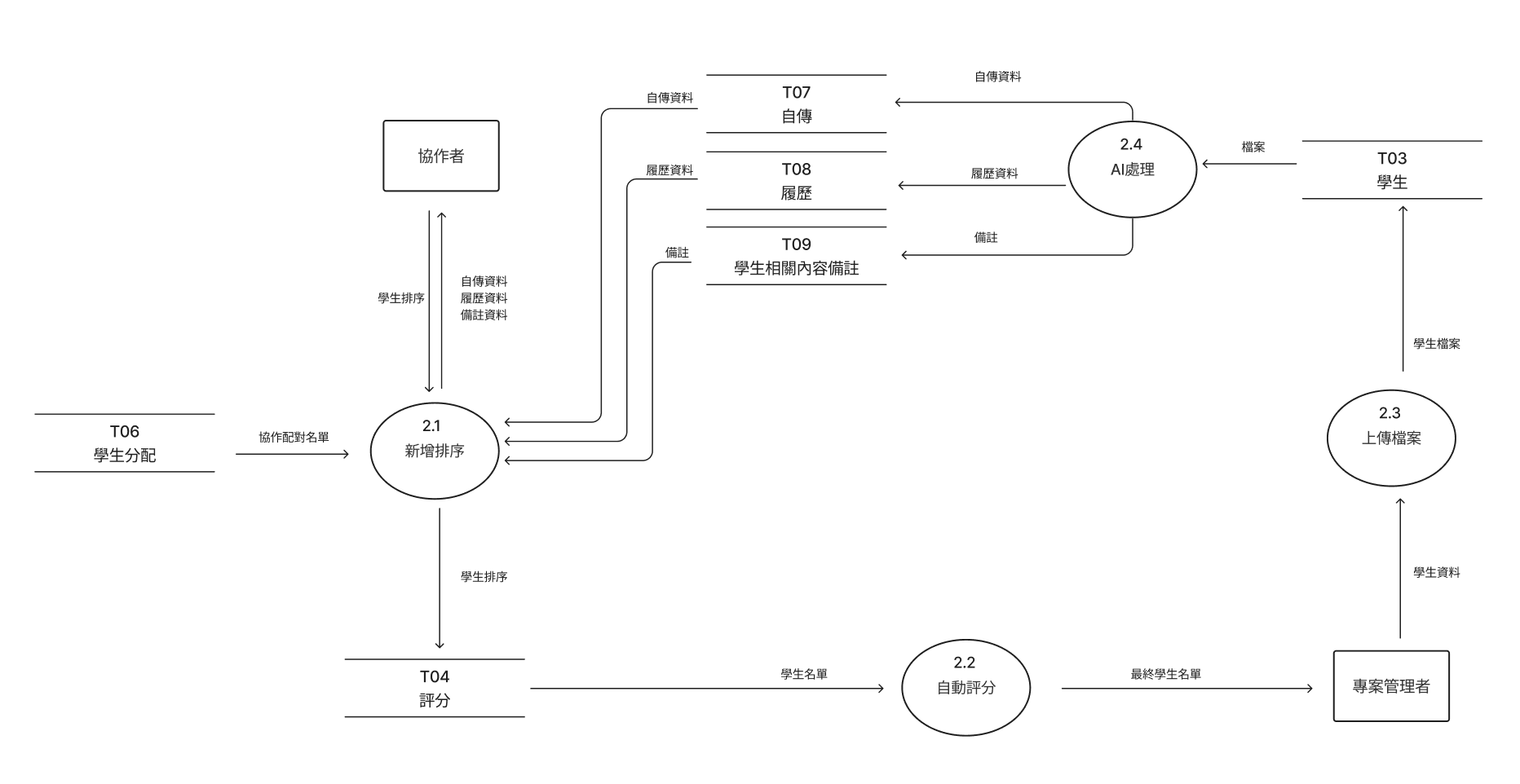
▲ 圖6-1-1 系統環境圖



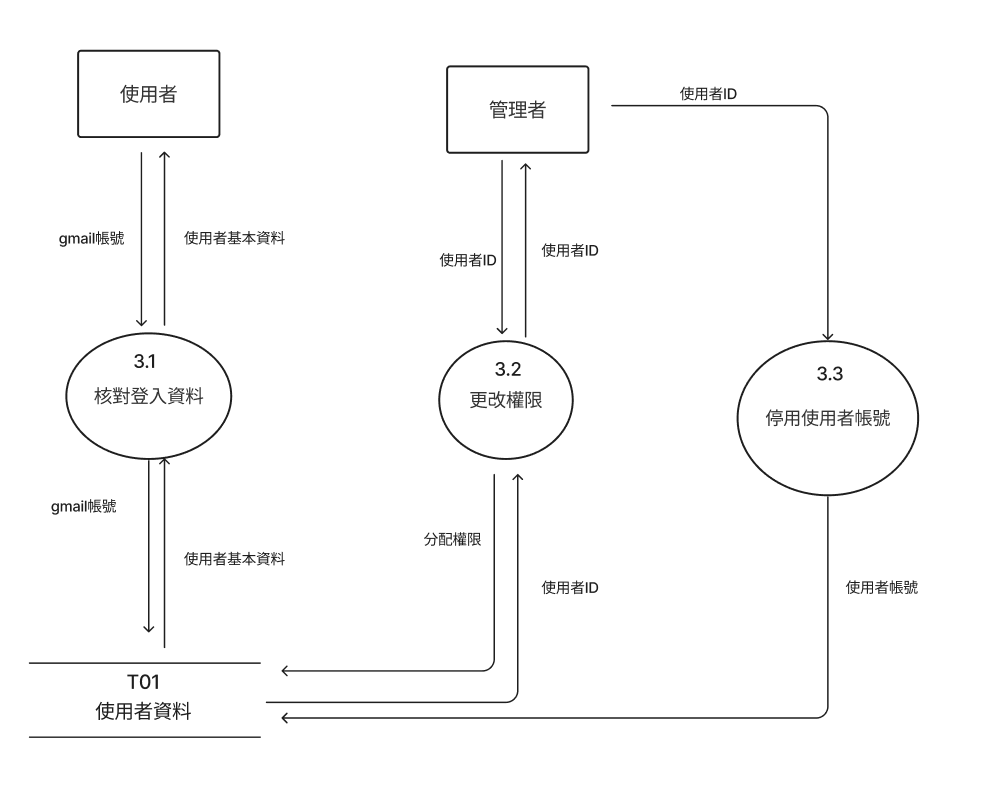
▲ 圖6-1-2 DFD圖0



▲ 圖6-1-3 DFD圖1



▲ 圖6-1-4 DFD圖2



▲ 圖6-1-5 DFD圖3

6-2 程序規格書(Process specification)

▼ 表6-2-1 程序規格

|  |  |  |  |
| --- | --- | --- | --- |
| 編號 | 1 | 功能名稱 | 專案功能 |
| 操作說明 | 輸入專案設定，新增專案，查詢專案 | | |
| 輸入值 | 專案設定、使用者名單 | | |
| 輸出值 | 專案設定、歷年排序、最終錄取名單 | | |
| 編號 | 1.1 | 功能名稱 | 新增專案設定 |
| 操作說明 | 新增專案細項 | | |
| 輸入值 | 專案基本資料 | | |
| 輸出值 | 專案基本資料 | | |
| 編號 | 1.2 | 功能名稱 | 加入協作者 |
| 操作說明 | 加入使用者排序名單 | | |
| 輸入值 | 協作者ID、權限 | | |
| 輸出值 | 協作者ID、權限 | | |
| 編號 | 1.3 | 功能名稱 | 查詢專案 |
| 操作說明 | 查詢專案進度 | | |
| 輸入值 | 專案編號 | | |
| 輸出值 | 專案編號、最終錄取名單 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 編號 | 2 | 功能名稱 | 協作編輯 |
| 操作說明 | 排序錄取學生的名單 | | |
| 輸入值 | 協作者名單、學生排序 | | |
| 輸出值 | 學生名單、學生排序 | | |
| 編號 | 2.1 | 功能名稱 | 新增排序 |
| 操作說明 | 在瀏覽各個學生資料後，排序學生錄取順序 | | |
| 輸入值 | 學生排序 | | |
| 輸出值 | 學生排序 | | |
| 編號 | 2.2 | 功能名稱 | 自動評分 |
| 操作說明 | 在最終排序完成後，系統自動平分給每個人不重複的分數 | | |
| 輸入值 | 學生名單 | | |
| 輸出值 | 最終學生名單 | | |
| 編號 | 2.3 | 功能名稱 | 上傳檔案 |
| 操作說明 | 專案管理者上傳所有的學生資料，系統自動切割內容後加入資料庫 | | |
| 輸入值 | 學生資料 | | |
| 輸出值 | 學生檔案 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 編號 | 2.4 | 功能名稱 | AI處理 |
| 操作說明 | AI整理學生檔案的內容，顯示內容摘要 | | |
| 輸入值 | 檔案 | | |
| 輸出值 | 學生資料 | | |
| 編號 | 3 | 功能名稱 | 用戶管理 |
| 操作說明 | 新增使用者帳號 | | |
| 輸入值 | gmail帳號、頭貼 | | |
| 輸出值 | 使用者ID、頭貼 | | |
| 編號 | 3.1 | 功能名稱 | 核對登入資料 |
| 操作說明 | 核對資料庫是否有使用者輸入的帳號。若有帳號，從資料庫傳回使用者遊戲資訊；若無帳號，則自動註冊。 | | |
| 輸入值 | gmail帳號、頭貼 | | |
| 輸出值 | 使用者ID、頭貼 | | |
| 編號 | 3.2 | 功能名稱 | 更改權限 |
| 操作說明 | 系統後台管理者可以更改使用者對整個系統的權限 | | |
| 輸入值 | 使用者ID、權限 | | |
| 輸出值 | 使用者ID、頭貼、權限 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 編號 | 3.3 | 功能名稱 | 停用使用者帳號 |
| 操作說明 | 系統管理者可以停用使用者的系統權限與帳號 | | |
| 輸入值 | 使用者ID | | |
| 輸出值 | 使用者ID | | |

第 7 章 資料模型

7-1 實體關聯圖(Entity relationship diagram)



* 圖7-1-1 實體關聯圖

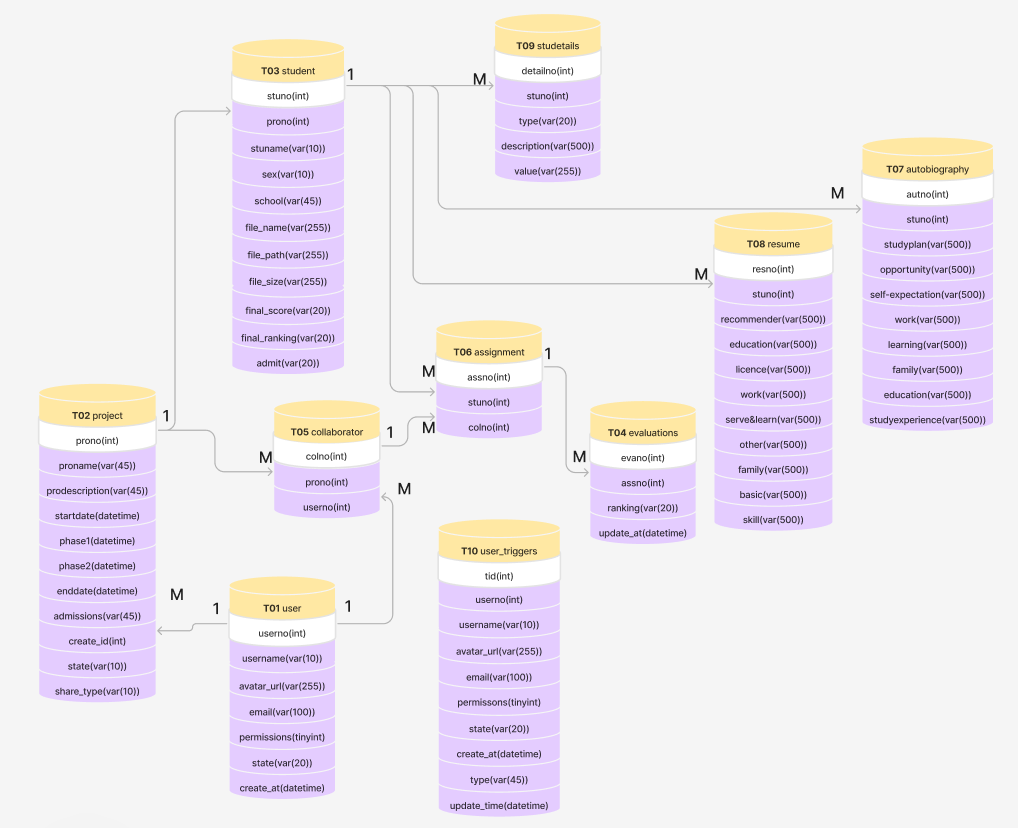
7-2資料字典(Data dictionary)

* 表7-2-1 字典清單

|  |  |  |
| --- | --- | --- |
| 資料表編號 | 資料表名稱 | 資料表中文名稱 |
| T01 | user | 使用者資料 |
| T02 | project | 專案 |
| T03 | student | 學生 |
| T04 | evaluations | 評分 |
| T05 | collaborator | 協作者 |
| T06 | assignment | 學生分配 |
| T07 | autobiography | 自傳 |
| T08 | resume | 履歷 |
| T09 | studetails | 學生相關內容備註 |
| T10 | user\_triggers | 使用者異動 |

第 8 章 資料庫設計

8-1 資料庫關聯圖



▲ 圖8-1-1 資料關聯圖

8-2 表格及其Meta data

* 表8-2-1 使用者資料

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| T01 user 使用者資料 | | | | | | |
| 欄位名稱 | 資料型態 | 長度 | 唯一性 | 允許空值 | 中文 | 備註 |
| userno | Int |  | 是 | 否 | 使用者編號 | PK |
| username | Varchar | 10 | 是 | 否 | 使用者名稱 |  |
| avatar\_url | Varchar | 255 | 是 | 否 | 頭像連結 |  |
| email | Varchar | 100 | 是 | 否 | 電子信箱 |  |
| permission | Tinyint |  | 否 | 否 | 權限 |  |
| state | Varchar | 20 | 否 | 否 | 帳號狀態 |  |
| create\_at | Datetime |  | 否 | 否 | 建立時間 |  |

* 表8-2-2 專案

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| T02 project 專案 | | | | | | |
| 欄位名稱 | 資料型態 | 長度 | 唯一性 | 允許空值 | 中文 | 備註 |
| prono | Int |  | 是 | 否 | 專案編號 | PK |
| proname | Varchar | 45 | 否 | 否 | 專案名稱 |  |
| prodescription | Varchar | 45 | 否 | 否 | 專案描述(學制) |  |
| startdate | Datetime |  | 否 | 否 | 開始日期 |  |
| phase1 | Datetime |  | 否 | 否 | 第一階段開始日期 |  |
| phase2 | Datetime |  | 否 | 否 | 第二階段開始日期 |  |
| enddate | Datetime |  | 否 | 否 | 結束日期 |  |
| admissions | Varchar | 45 | 否 | 否 | 錄取人數 |  |
| create\_id | Int |  | 否 | 否 | 創建者編號 | FK |
| state | Varchar | 10 | 否 | 否 | 專案狀態 |  |

* 表8-2-3 學生

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| T03 student 學生 | | | | | | |
| 欄位名稱 | 資料型態 | 長度 | 唯一性 | 允許空值 | 中文 | 備註 |
| stuno | Int |  | 是 | 否 | 學生編號 | PK |
| prono | Int |  | 否 | 否 | 專案編號 | FK |
| stuname | Varchar | 10 | 是 | 否 | 專案名稱 |  |
| sex | Varchar | 10 | 否 | 是 | 性別 |  |
| school | Varchar | 45 | 否 | 是 | 學校 |  |
| file\_name | Varchar | 255 | 否 | 是 | 檔案名稱 |  |
| file\_path | Varchar | 255 | 否 | 是 | 檔案路徑 |  |
| file\_size | Varchar | 255 | 否 | 是 | 檔案大小 |  |
| final\_score | Varchar | 20 | 否 | 是 | 最終評分 |  |
| final\_ranking | Varchar | 20 | 否 | 是 | 最終排名 |  |
| admit | Varchar | 20 | 否 | 是 | 錄取 |  |

* 表8-2-4 評分

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| T04 evaluations 評分 | | | | | | |
| 欄位名稱 | 資料型態 | 長度 | 唯一性 | 允許空值 | 中文 | 備註 |
| evano | Int |  | 是 | 否 | 評分編號 | PK |
| assno | Int |  | 否 | 否 | 分配編號 | FK |
| ranking | Varchar | 20 | 否 | 是 | 排名 |  |
| update\_at | Datetime |  | 否 | 是 | 編輯時間 |  |

* 表8-2-5 協作者

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| T05 collaborator 協作者 | | | | | | |
| 欄位名稱 | 資料型態 | 長度 | 唯一性 | 允許空值 | 中文 | 備註 |
| colon | Int |  | 是 | 否 | 協作編號 | PK |
| prono | Int |  | 否 | 否 | 專案編號 | FK |
| userno | Int |  | 否 | 否 | 使用者編號 | FK |

* 表8-2-6 學生分配

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| T06 assignment 學生分配 | | | | | | |
| 欄位名稱 | 資料型態 | 長度 | 唯一性 | 允許空值 | 中文 | 備註 |
| assno | Int |  | 是 | 否 | 分配編號 | PK |
| colon | Int |  | 否 | 否 | 協作編號 | FK |
| stuno | Int |  | 否 | 否 | 學生編號 | FK |

* 表8-2-7 自傳

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| T07 autobiography 自傳 | | | | | | |
| 欄位名稱 | 資料型態 | 長度 | 唯一性 | 允許空值 | 中文 | 備註 |
| autno | Int |  | 是 | 否 | 自傳編號 | PK |
| stuno | Int |  | 否 | 否 | 學生編號 | FK |
| studyplan | Varchar | 500 | 否 | 是 | 讀書計畫 |  |
| opportunity | Varchar | 500 | 否 | 是 | 機會 |  |
| self-expectation | Varchar | 500 | 否 | 是 | 自我期許 |  |
| work | Varchar | 500 | 否 | 是 | 工作背景 |  |
| learning | Varchar | 500 | 否 | 是 | 多元學習 |  |
| family | Varchar | 500 | 否 | 是 | 家庭背景 |  |
| education | Varchar | 500 | 否 | 是 | 教育背景 |  |
| studyexperience | Varchar | 500 | 否 | 是 | 求學經歷 |  |

* 表8-2-8 履歷

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| T08 resume履歷 | | | | | | |
| 欄位名稱 | 資料型態 | 長度 | 唯一性 | 允許空值 | 中文 | 備註 |
| resno | Int |  | 是 | 否 | 履歷編號 | PK |
| stuno | Int |  | 否 | 否 | 學生編號 | FK |
| recommender | Varchar | 500 | 否 | 是 | 推薦人 |  |
| education | Varchar | 500 | 否 | 是 | 學歷 |  |
| licence | Varchar | 500 | 否 | 是 | 證照 |  |
| work | Varchar | 500 | 否 | 是 | 工讀 |  |
| serve&learn | Varchar | 500 | 否 | 是 | 服務及學習 |  |
| other | Varchar | 500 | 否 | 是 | 其他資訊 |  |
| family | Varchar | 500 | 否 | 是 | 家庭狀況 |  |
| basic | Varchar | 500 | 否 | 是 | 基本資料 |  |
| skill | Varchar | 500 | 否 | 是 | 技能 |  |

* 表8-2-9 學生相關內容備註

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| T09 studetails 學生相關內容備註 | | | | | | |
| 欄位名稱 | 資料型態 | 長度 | 唯一性 | 允許空值 | 中文 | 備註 |
| detailno | Int |  | 是 | 否 | 備註編號 | PK |
| stuno | Int |  | 否 | 否 | 學生編號 | FK |
| type | Varchar | 20 | 否 | 否 | 內容類型 |  |
| description | Varchar | 500 | 否 | 否 | 詳細資料 |  |
| value | Varchar | 255 | 否 | 否 | 實際值 |  |

* 表8-2-10 使用者異動

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| T10 user\_triggers 使用者異動 | | | | | | |
| 欄位名稱 | 資料型態 | 長度 | 唯一性 | 允許空值 | 中文 | 備註 |
| tid | Int |  | 是 | 否 | 流水號 | PK |
| userno | Int |  | 否 | 否 | 使用者編號 |  |
| username | Varchar | 10 | 否 | 否 | 使用者名稱 |  |
| google\_id | Varchar | 255 | 否 | 否 | 谷歌登入編號 |  |
| avatar\_url | Varchar | 255 | 否 | 否 | 頭像連結 |  |
| email | Varchar | 100 | 否 | 否 | 電子信箱 |  |
| permission | Tinyint |  | 否 | 否 | 權限 |  |
| state | Varchar | 20 | 否 | 否 | 帳號狀態 |  |
| create\_at | Datetime |  | 否 | 否 | 建立時間 |  |
| type | Varchar | 45 | 否 | 否 | 修改類型 |  |
| update\_time | Datetime |  | 否 | 否 | 修改時間 |  |

第 9 章 程式

9-1 軟體架構與程式清單

* 表9-1-1前端程式清單

|  |  |  |
| --- | --- | --- |
| 編號 | 檔案名稱 | 功能名稱 |
| 1 | components/common/NavBar.tsx | 導覽列相關 |
| 2 | components/common/SideBar.tsx | 未登入導覽列 |
| 3 | components/Layout/Layout.tsx | UI相關 |
| 4 | components/popup/addChoose.tsx | 新增審核方式 |
| 5 | components/popup/addFile.tsx | 上傳檔案 |
| 6 | components/popup/addName.tsx | 新增專案名稱 |
| 7 | components/popup/addPeople.tsx | 新增錄取人數 |
| 8 | components/popup/addTime.tsx | 新增專案時程 |
|  | components/popup/projectChoosePopup.tsx | 編輯審核方式 |
|  | components/popup/projectFilePopup.tsx | 重新上傳檔案 |
|  | omponents/popup/projectNamePopup.tsx | 編輯專案名稱 |
|  | omponents/popup/projectPeoplePopup.tsx | 編輯錄取人數 |
|  | omponents/popup/projectTimeSet.tsx | 編輯專案時程 |
|  | components/popup/checkPopup.tsx | 登入成功 |
|  | components/popup/confirmPopup.tsx | 確定執行 |
|  | confirmPopup/popup/failPopup.tsx | 登入失敗 |
|  | pages/index.tsx | 主畫面相關 |
|  | pages/privateRoute.tsx | 登入驗證 |
|  | pages/aboutUser/list.tsx | 使用者相關 |
|  | pages/project/[prono].tsx | 歷年專案內容 |
|  | pages/project/list.tsx | 歷年專案列表 |
|  | pages/project/project.tsx | 歷年專案相關 |
|  | pages/projectManage/[prono]/edit.tsx | 編輯專案 |
|  | pages/projectManage/[prono]/index.tsx | 專案主畫面相關 |
|  | pages/projectManage/[prono]/teacher.tsx | 新增協作老師 |
|  | pages/projectManage/add.tsx | 編輯畫面相關 |
|  | pages/projectManage/list.tsx | 專案列表 |
|  | pages/projectManage/project.tsx | 專案相關 |
|  | pages/user/about.tsx | 帳號相關 |
|  | pages/user/login.tsx | 登入相關 |
|  | pages/user/loginFail.tsx | 登入失敗 |
|  | pages/user/loginSuccess.tsx | 登入成功 |
|  | pages/userManage/list.tsx | 權限相關 |
|  | pages/userManage/user.tsx | 編輯帳號權限 |

* 表9-1-2後端程式清單

|  |  |  |
| --- | --- | --- |
| 編號 | 檔案名稱 | 功能名稱 |
| 1 | app.js | Api相關 |
| 2 | routes/auth.js | 登入相關 |
| 3 | routes/user/data.js | 使用者相關 |
| 4 | routes/user/getRole.js | 權限相關 |
| 5 | routes/user/list.js | 帳號管理相關 |
| 6 | routes/user/page.js | 分頁相關 |
| 7 | routes/user/roleMap.js | 權限相關 |
| 8 | routes/user/update.js | 編輯使用者資料 |
| 9 | routes/project/data.js | 我的專案列表 |
| 10 | routes/project/dataPage.js | 列表分頁相關 |
| 11 | routes/project/delete.js | 刪除專案 |
| 12 | routes/project/insert.js | 新增專案相關 |
| 13 | routes/project/item.js | 歷史專案相關 |
| 14 | routes/project/list.js | 歷史專案列表 |
| 15 | routes/project/page.js | 分頁相關 |
| 16 | routes/project/update.js | 編輯專案 |
| 17 | routes/score/finalist.js | 最終排序 |
| 18 | routes/score/merge.js | 合併排序 |
| 19 | routes/score/sort.js | 排序相關 |
| 20 | routes/score/student.js | 學生資料 |

9-2 程式規格描述

* 表9-2-1 tsx部分程式描述

|  |  |
| --- | --- |
| 程式名稱 | Layout.tsx |
| 目的 | 主要UI畫面佈局 |
| import Head from "next/head";  import { useEffect, useState } from 'react';  import NavBar from "@/components/common/NavBar";  import styles from "@/styles/components/layout.module.scss";  interface LayoutProps {  children: React.ReactNode;  user?: { username: string, avatar\_url: string } | undefined;  }  export default function Layout({ children, user }: LayoutProps) {  const [userData, setUserData] = useState<{ username: string, avatar\_url: string } | undefined>(user);  const fetchUser = async () => {  const res = await fetch('http://localhost:5000/api/auth/user', {  credentials: 'include', // 確保發送請求時會附帶 cookie  });  if (res.ok) {  const data = await res.json();  setUserData(data.user);  }  };  useEffect(() => {  // 檢查用戶是否已經存在於狀態中  if (!userData) {  fetchUser();  }  }, []);  return (  <>  <Head>  <title>履人</title>  <meta charSet="UTF-8" />  <meta name="description" content="NTUB IMD ESN" />  <meta name="viewport" content="width=device-width, initial-scale=1.0" />  <link rel="icon" href="/favicon.ico" />  </Head>  <NavBar user={userData} />  <div className={styles.layout}>  {children}  </div>  </>  );  } | |
| 程式名稱 | projectFilePopup.tsx |
| 目的 | 編輯上傳檔案 |
| import styles from "@/styles/components/popup/project/projectFilePopup.module.scss";  import React, { useState } from "react";  import { MdOutlineCancel, MdFileUpload } from "react-icons/md";  import { FaCheck } from "react-icons/fa";  interface ProjectFileInputProps {  onClose: () => void;  }  const ProjectFileInput: React.FC<ProjectFileInputProps> = ({ onClose }) => {  const [files, setFiles] = useState<FileList | null>(null);  const [filesName, setFilesName] = useState<string>('點擊或拖動文件到這裡上傳');  const handleFileChange = (event: React.ChangeEvent<HTMLInputElement>) => {  setFiles(event.target.files);  setFilesName(event.target.files ? event.target.files[0].name : '');  };  return (  <div className={styles.popupBG}>  <div className={styles.mainShow}>  <section className={styles.title}>  <h4>學生文件檔案(請壓成.zip檔案)</h4>  <a onClick={onClose} className={styles.closeButton}>  <MdOutlineCancel />  </a>  </section>  <section className={styles.inputArea}>  <input  type="file"  id="file"  name="file"  onChange={handleFileChange}  multiple  accept=".zip"  className={styles.hiddenInput}  />  <label htmlFor="file" className={styles.fileArea}>  <MdFileUpload className={styles.donloadicon} />  {filesName}  </label>  </section>  <a className={`${styles.button} ${styles.check}`} onClick={onClose}>  <FaCheck />  </a>  </div>  </div>  );  }  export default ProjectFileInput; | |
| 程式名稱 | teacher.tsx |
| 目的 | 新增協作老師 |
| import Layout from '@/components/Layout/Layout';  import styles from '@/styles/page/project/teacher.module.scss'  import React, { useState, useEffect } from 'react';  import { FaCheck, FaSignOutAlt,FaTimes } from "react-icons/fa";  import { useRouter } from 'next/router';  import PrivateRoute from '@/pages/privateRoute';  import userI from '@/type/userI';  import axios from 'axios';  import CheckPopup from '@/components/popup/checkPopup';  type userType = {  colno?:string;  userno?:string;  username?:string;  permissions?:string;  }  function SetTeacher({ user }: { user: userI | undefined }) {  const router = useRouter();  const { prono } = router.query;  const [ready,setReady] = useState<{ready:userType[]}>();  const [changeUser,setChangeUser] = useState<string[]>([]);  const [users,setUsers] = useState<{teacherDB:userType[]}>();  const [showPopup, setShowPopup] = useState(false);  useEffect(() => {  const fetchProjectData = async () => {  try {  const response = await axios.get(`/api/project/update/search/teacher`, {  params: { prono }  });  setReady(response.data[0])  setChangeUser(readyID(response.data[0]))  setUsers(response.data[1])  } catch (error) {  console.error('Error fetching project data:', error);  }  };  setChangeUser(readyID(ready as {ready:userType[]}))  fetchProjectData();  }, [prono,]);  const readyID = ( data: {ready:userType[]}) => {  const readyList: string[] = [];  data ?  data.ready.map((readys, index) => (  readyList.push(readys.userno as string)  )) : ''  return readyList;  }  const handleCheckboxChange = (teacherID: string) => {  setChangeUser(prev =>  prev.includes(teacherID)  ? prev.filter(id => id !== teacherID)  : [...prev, teacherID]  );  };  const handSubmit = async () => {  try {  await axios.post(`/api/project/update/update/teacher`, {  changeUser  }, {  params: { prono }  });  setShowPopup(true);  setTimeout(() => {  setShowPopup(false);  // 3 秒後跳轉頁面  router.push(`/projectManage/${prono}/edit`);  }, 3000);  } catch (error) {  console.error('Error fetching project data:', error);  }  };  return (  <Layout user={user}>  {showPopup && <CheckPopup title={'成功修改'} />}  <main className={styles.teacherArea}>  <h2>編輯專案 - 選擇協作老師</h2>  <div className={styles.allArea}>  {users?.teacherDB.map((teacher, index) => (  <div key={index} className={styles.nameArea}>  <input  type="checkbox"  id={`checkbox-${index}`}  checked={changeUser.includes(teacher.userno as string)}  onChange={() => handleCheckboxChange(teacher.userno as string)} // 使用 onChange  />  <label htmlFor={`checkbox-${index}`} className={`${styles.name}  ${changeUser.includes(teacher.userno as string) ? styles.colorTRUE : styles.colorFALSE}`}>  {changeUser.includes(teacher.userno as string) ? <FaTimes /> : ''}  {teacher.username}  </label>  </div>  ))}  </div>  <section className={styles.chooseArea}>  <p>建議選擇教師人數:5位</p>  <h5>已選擇教師:{changeUser.length}位</h5>  <h6>每位教師分配學生:{changeUser.length}位</h6>  </section>  <section className={styles.setingButton}>  <a className={`${styles.button} ${styles.check}`} onClick={handSubmit}><FaCheck /></a>  <a onClick={() => router.push(`/projectManage/${prono}/edit`)} className={`${styles.button} ${styles.delete}`}><FaSignOutAlt /></a>  </section>  </main>  </Layout>  );  }  export default function Init() {  const [user, setUser] = useState<userI>();  return (  <PrivateRoute>  <SetTeacher user={user} />  </PrivateRoute>  );  } | |
| 程式名稱 | project/list.tsx |
| 目的 | 歷年專案列表 |
| import Layout from '@/components/Layout/Layout';  import styles from '@/styles/page/project/list.module.scss'  import { Box, FormControl, InputLabel, MenuItem, Pagination, PaginationItem, Select, SelectChangeEvent, Stack } from '@mui/material';  import React, { useEffect, useState } from 'react';  import { FaAngleLeft, FaAngleRight } from "react-icons/fa";  import { useRouter } from 'next/router';  import Project from './project';  import projectInterface from '../../type/projectinterface'  import axios from 'axios';  import PrivateRoute from '../privateRoute';  import userI from '../../type/userI';  function ProjectList({ user }: { user: userI | undefined }) {  const [year, setYear] = React.useState('');  const [academic, setAcademic] = React.useState('');  const [page, setPage] = React.useState<number>(1);  const [pageSize, setPageSize] = React.useState<number>(1);  const [projects, setProjects] = useState<projectInterface[]>([]);  useEffect(() => {  const getList = async () => {  try {  const [listResponse, sizeResponse] = await Promise.all([  axios.get('/api/project/list', {  params: {  year: year === 'all' ? '' : year,  academic: academic === 'all' ? '' : academic,  page  }  }),  axios.get('/api/project/page', {  params: {  year: year === 'all' ? '' : year,  academic: academic === 'all' ? '' : academic,  }  })  ]);  setProjects(listResponse.data);  setPageSize(Number(sizeResponse.data.page));  } catch (error) {  console.error("Error fetching projects:", error);  }  };    getList();  }, [year, academic, page]);    const handleChange = (event: SelectChangeEvent) => {  const { name, value } = event.target;  if (name === 'year') {  setYear(value); // 保持为 'all' 或者具体的年份  console.log('setYear:', value);  }  if (name === 'academic') {  setAcademic(value); // 保持为 'all' 或者具体的学制  console.log('setAcademic:', value);  }  };  const handlePageChange = (event: React.ChangeEvent<unknown>, value: number) => {  setPage(value);  console.log('Current Page:', value);  };  return (  <Layout>  <main className={styles.listArea}>  <h2>歷年專案</h2>  <section className={styles.dropdownArea}>  <Box sx={{ minWidth: 120 }}>  <FormControl sx={{ m: 2, minWidth: 120, height: 20, }} size="small">  <InputLabel id="demo-simple-select-label">學制</InputLabel>  <Select  sx={{ borderRadius: 20 }}  labelId="demo-simple-select-label"  id="demo-simple-select"  value={academic}  label="academic"  name="academic"  onChange={handleChange}  >  <MenuItem value={'all'}>全部</MenuItem>  <MenuItem value={'01'}>博士</MenuItem>  <MenuItem value={'02'}>二技</MenuItem>  <MenuItem value={'03'}>碩士</MenuItem>  <MenuItem value={'04'}>四技</MenuItem>  </Select>  </FormControl>  <FormControl sx={{ m: 2, minWidth: 120 }} size="small">  <InputLabel id="demo-simple-select-label">年度</InputLabel>  <Select  sx={{ borderRadius: 20 }}  labelId="demo-simple-select-label"  id="demo-simple-select"  value={year}  label="year"  name="year"  onChange={handleChange}  >  <MenuItem value={'all'}>全部</MenuItem>  <MenuItem value={'112'}>112</MenuItem>  <MenuItem value={'113'}>113</MenuItem>  <MenuItem value={'114'}>114</MenuItem>  </Select>  </FormControl>  </Box>  </section>  <section className={styles.projectList}>  {projects.map((project, index) => (  <React.Fragment key={project.prono}>  <Project project={project} />  {index < projects.length-1 && <hr/>}    </React.Fragment>  ))}  </section>  <section className={styles.projectPage}>  <Stack spacing={2}>  <Pagination  count={pageSize}  onChange={handlePageChange}  renderItem={(item) => (  <PaginationItem  slots={{ previous: FaAngleLeft, next: FaAngleRight }}  {...item}  />  )}  />  </Stack>  </section>  </main>  </Layout>  );  }  export default function Init() {  const [user, setUser] = useState<userI>();  return (  <PrivateRoute>  <ProjectList user={user} />  </PrivateRoute>  );  } | |
| 程式名稱 | project/project.tsx |
| 目的 | 歷年專案列表內容 |
| import router from 'next/router';  import projectInterface from '../../type/projectinterface';  import styles from '@/styles/page/project/list.module.scss';  import { useEffect, useState } from 'react';  interface proProps {  project: projectInterface;  }  function Project({ project }: proProps) {  const [edate, setEdata] = useState<Date>();  const [proItem, setProItem] = useState<projectInterface>();  useEffect(() => {  if (router.isReady) {  if (project) {  setEdata(new Date(project?.enddate));  setProItem(project);  } else {  console.log('No project found in query.');  }  }  }, [project]);  return (  <div onClick={() => router.push(`/project/${proItem?.prono}`)} className={styles.projectMain}>  <div className={styles.projectItem}>  <article>  <div className={styles.projectLogo}>  <p>{proItem?.prono\_prefix}</p>  </div>  <div className={styles.projectContent}>  <b>{proItem?.proname}</b>  <p>專案建立者: <span>{proItem?.username}</span></p>  <p>排序進度: {proItem?.state}</p>  </div>  </article>  <div className={styles.projectTime}>  <p>結案日期：{formatDate(edate as Date)}</p>  </div>  </div>  </div>  );  }  function formatDate(date: Date) {  const year = date?.getFullYear();  const month = String(date?.getMonth() + 1).padStart(2, '0'); // 月份是從 0 開始的  const day = String(date?.getDate()).padStart(2, '0');  const hours = String(date?.getHours()).padStart(2, '0');  const minutes = String(date?.getMinutes()).padStart(2, '0');  const seconds = String(date?.getSeconds()).padStart(2, '0');  return `${year}/${month}/${day} ${hours}:${minutes}:${seconds}`;  }  export default Project; | |

* 表9-2-2 js部分程式描述

|  |  |
| --- | --- |
| 程式名稱 | auth.js |
| 目的 | 登入辨識 |
| const express = require('express');  const router = express.Router();  const pool = require('../lib/db');  const { OAuth2Client } = require('google-auth-library');  const { GOOGLE\_CLIENT\_ID, GOOGLE\_SECRET\_KEY, HOST } = process.env;  const client = new OAuth2Client({  clientId: GOOGLE\_CLIENT\_ID,  clientSecret: GOOGLE\_SECRET\_KEY,  redirectUri: `${HOST}/api/auth/callback`,  });  // 查看資料庫內有無紀錄  const getUserFromDb = async (email) => {  const [rows] = await pool.query('SELECT \* FROM `student-project`.`user` WHERE email = ?', [email]);  return rows[0];  };  const createUserInDb = async (userInfo) => {  // 動態獲取當前年份和月份  const currentDate = new Date();  const currentYear = currentDate.getFullYear() - 1911; // 將西元年轉為民國年  const currentMonth = String(currentDate.getMonth() + 1).padStart(2, '0'); // 獲取月份並補齊兩位  // 獲取當前年份內最新的 userno  const [result] = await pool.query(  'SELECT userno FROM `student-project`.`user` WHERE userno LIKE ? ORDER BY userno DESC LIMIT 1',  [`${currentYear}%`]  );  let newUserNo;  if (result.length > 0) {  const latestUserNo = result[0].userno;  const latestSequence = latestUserNo % 1000 + 1; // 提取最新的序號部分並加1  newUserNo = parseInt(`${currentYear}${currentMonth}${String(latestSequence).padStart(3, '0')}`);  } else {  newUserNo = `${currentYear}${currentMonth}001`;  }  await pool.query('INSERT INTO `student-project`.`user` (userno, username, avatar\_url, email, permissions, state) VALUES (?, ?, ?, ?, ?, ?)', [  newUserNo, userInfo.name, userInfo.picture, userInfo.email, 0, '1'  ]);  return {  ...userInfo,  userno: newUserNo,  permissions: 0,  state: '1',  };  };  router.get('/login', (req, res) => {  const authorizeUrl = client.generateAuthUrl({  access\_type: 'offline',  scope: [  'https://www.googleapis.com/auth/userinfo.profile',  'https://www.googleapis.com/auth/userinfo.email',  ],  });  res.redirect(authorizeUrl);  });  router.get('/callback', async (req, res) => {  const { code } = req.query;  try {  const { tokens } = await client.getToken(code);  client.setCredentials(tokens);  const userInfoResponse = await client.request({  url: 'https://www.googleapis.com/oauth2/v3/userinfo'  });  const userInfo = userInfoResponse.data;  const userEmail = userInfo.email;  if (!userEmail.endsWith('@ntub.edu.tw')) {  return res.redirect('http://localhost:3000/user/loginFail');  }  let user = await getUserFromDb(userEmail);  if (!user) {  user = await createUserInDb(userInfo);  }  req.session.user = user;  console.log(user);  await req.session.save()  res.redirect('http://localhost:3000/user/loginSuccess');  } catch (error) {  console.error('Error in authentication flow:', error);  res.status(400).send('Error fetching Google user info');  }  });  router.get('/logout', (req, res) => {  req.session.destroy();  res.redirect('http://localhost:3000/user/login');  });  router.get('/user', async (req, res) => {  if (req.session && req.session.user) {  res.json({ user: req.session.user });  } else {  res.status(404).send('User not found');  }  });  module.exports = router; | |
| 程式名稱 | project/list.js |
| 目的 | 歷年專案列表 |
| const express = require('express');  const router = express.Router();  const pool = require('../../lib/db');  router.get('/', async (req, res) => {  const { year, academic, page = 1 } = req.query;  try {  // 基本查詢語句  let query = `SELECT p.\*, u.username, LEFT(p.prono, 3) AS prono\_prefix  FROM \`student-project\`.\`project\` p  JOIN user u ON p.create\_id = u.userno  WHERE p.state = '已關閉'  `;  let params = [];  // 根據是否有 year 來構造查詢條件  if (year) {  query += ' AND LEFT(prono, 3) = ?';  params.push(year);  }  // 根據是否有 academic 來構造查詢條件  if (academic) {  query += ' AND SUBSTRING(prono, 4, 2) = ?';  params.push(academic);  }  // 分頁設定  const pageSize = 5;  const offset = (page - 1) \* pageSize;  query += ' LIMIT ? OFFSET ?';  params.push(pageSize, offset);  // 執行查詢  const [results] = await pool.query(query, params);  // 返回結果  res.json(results);  } catch (error) {  console.error('Error in database query:', error);  res.status(500).send('Error fetching project data');  }  });  module.exports = router; | |
| 程式名稱 | item.js |
| 目的 | 歷年專案列表內容 |
| const express = require('express');  const router = express.Router();  const pool = require('../../lib/db');  // 根據 prono 獲取項目及相關學生資料  router.get('/:prono', async (req, res) => {  try {  const prono = req.params.prono;  const [rows] = await pool.query(`  SELECT \*  FROM \`student-project\`.project p  JOIN student s ON p.prono = s.prono  LEFT JOIN studetails sd ON s.stuno = sd.stuno  JOIN autobiography au ON s.stuno = au.stuno  JOIN \`resume\` r ON s.stuno = r.stuno  WHERE p.prono = ?;  `, [prono]);  // 返回查詢結果  res.json(rows); // 返回合併後的資料  } catch (error) {  console.error(error);  res.status(500).json({ error: '獲取資料失敗' });  }  });  module.exports = router; | |
| 程式名稱 | merge.js |
| 目的 | 合併排序 |
| const express = require('express');  const router = express.Router();  const pool = require('../../lib/db');  // 等距分配分數函數  function distributeScores(min, max, numberOfPeople) {  const scores = [];  const step = (max - min) / (numberOfPeople - 1); // 每個人之間的分數差  for (let i = 0; i < numberOfPeople; i++) {  const score = max - i \* step; // 根據順序，從高到低分配  scores.push(parseFloat(score.toFixed(2))); // 保存分數，保留兩位小數(轉換為浮點數)  }  return scores;  }  // 合併排序函數：按輪流取用的方式合併多個排序列表  const mergeRankings = (lists) => {  const merged = [];  const maxLength = Math.max(...lists.map(list => list.length));  for (let i = 0; i < maxLength; i++) {  lists.forEach(list => {  if (i < list.length) {  merged.push(list[i]);  }  });  }  return merged;  };  // 檢查錄取公平性  const checkAdmissionFairness = async (prono, admissionCount) => {  // 1. 獲取協作老師總數  const collaboratorQuery = 'SELECT COUNT(\*) AS collaboratorCount FROM `student-project`.collaborator WHERE prono = ?';  const collaboratorResult = await pool.query(collaboratorQuery, [prono]);  const collaboratorCount = collaboratorResult[0][0].collaboratorCount;  // 2. 計算每位老師應該錄取的學生數量  const remainderStudents = admissionCount % collaboratorCount; // 剩下的學生數  // 3. 確認是否有老師要多評分一個學生  if (remainderStudents > 0) {  return {  status: 'unfair',  message: `有${remainderStudents}名學生分配不公平，請重新評分這些學生。`,  extraStudentsCount: remainderStudents  };  } else {  return {  status: 'fair',  message: '分配公平，無需重新評分。'  };  }  };  function weightedRanking(allEvaluations) {  const scores = {};  allEvaluations.forEach(collaborator => {  collaborator.allEvaluations.forEach(({ stuno, ranking }) => {  // 初始化每位學生的分數  if (!scores[stuno]) {  scores[stuno] = { totalScore: 0, count: 0 }; // 初始化總分和計數  }  // 根據排名計算分數  const adjustedScore = 1 / ranking; // 使用排名的反轉值作為分數  scores[stuno].totalScore += adjustedScore; // 累加分數  scores[stuno].count += 1; // 增加計數  });  });  // 計算平均分數並排序  const weightedScores = Object.entries(scores).map(([stuno, { totalScore, count }]) => ({  stuno,  finalScore: totalScore / count // 計算平均分數  }));  // 按照 finalScore 降序排序  return weightedScores.sort((a, b) => b.finalScore - a.finalScore);  }  // 處理 share\_type 為 1 的邏輯  async function handleShareType1(prono) {  const sortedLists = [];  const collaborators = await pool.query('SELECT colno FROM collaborator WHERE prono = ?', [prono]);  for (const collaborator of collaborators[0]) {  const evaluationsQuery = `  SELECT s.stuno  FROM \`student-project\`.evaluations e  JOIN assignment a ON e.assno = a.assno  JOIN collaborator c ON a.colno = c.colno  JOIN student s ON a.stuno = s.stuno  WHERE c.prono = ?  ORDER BY e.ranking ASC`;    const evaluations = await pool.query(evaluationsQuery, [prono]);  const studentList = evaluations[0].map(e => e.stuno); // 獲取學生編號  sortedLists.push(studentList);  }  // 合併排序  return mergeRankings(sortedLists);  }  // 處理 share\_type 為 2 的邏輯  async function handleShareType2(prono) {  const allEvaluations = [];  // 查詢協作老師的編號  const [collaborators] = await pool.query('SELECT colno FROM collaborator WHERE prono = ?', [prono]);  for (const collaborator of collaborators) {  const evaluationsQuery = `  SELECT c.colno, e.evano, e.ranking, s.stuno  FROM \`student-project\`.evaluations e  JOIN assignment a ON e.assno = a.assno  JOIN collaborator c ON a.colno = c.colno  JOIN student s ON a.stuno = s.stuno  WHERE c.prono = ?  ORDER BY e.ranking ASC`;    // 查詢評估結果  const [evaluations] = await pool.query(evaluationsQuery, [prono]);  // 確保每個評估的 ranking 為數字，並確保 stuno 存在  const validEvaluations = evaluations.map(evaluation => ({  stuno: evaluation.stuno,  ranking: parseInt(evaluation.ranking, 10) || 0 // 將 ranking 轉換為數字，如果無效則設為 0  }));  if (validEvaluations.length > 0) {  allEvaluations.push({  colno: collaborator.colno,  allEvaluations: validEvaluations  });  }  }  // 確保加權排名的函數存在並正確使用  if (typeof weightedRanking === 'function') {  return weightedRanking(allEvaluations).map(evaluation => evaluation.stuno);  } else {  throw new Error('weightedRanking function is not defined.');  }  }  // 提交合併排序並分配分數  router.post('/', async (req, res) => {  const { prono } = req.body; // 從前端獲取 prono 和分數範圍  try {  // 1. 獲取錄取人數  const projectQuery = 'SELECT admissions, share\_type FROM `student-project`.project WHERE prono = ?';  const projectResult = await pool.query(projectQuery, [prono]);  const admissionCount = projectResult[0][0].admissions;  // 2. 檢查錄取公平性  const fairnessCheck = await checkAdmissionFairness(prono, admissionCount);  // 如果不公平，返回錯誤提示並要求重新評分  if (fairnessCheck.status === 'unfair') {  return res.json({ success: false, message: fairnessCheck.message });  }  // 處理合併排序  const share\_type = projectResult[0][0].share\_type; // 獲取 share\_type  let finalRankingList = [];  if (share\_type == 1) {  finalRankingList = await handleShareType1(prono);  } else if (share\_type == 2) {  finalRankingList = await handleShareType2(prono);  }  // 5. 分配分數（假設最高分 90，最低分 70）  const scores = distributeScores(70, 90, finalRankingList.length);  // 6. 插入或更新合併排序結果及其分數  const insertPromises = finalRankingList.map((stuno, index) => {  const final\_rank = index + 1;  const score = scores[index]; // 根據排序位置分配的分數  const isAdmitted = final\_rank <= admissionCount ? '錄取' : '未錄取'; // 根據錄取人數決定是否錄取  const updateQuery = 'UPDATE `student-project`.`student` SET final\_ranking = ?, final\_score = ?, admit = ? WHERE stuno = ?';  // 使用 final\_rank, score, 和 isAdmitted 更新資料庫  return pool.query(updateQuery, [final\_rank, score, isAdmitted, stuno]);  });  await Promise.all(insertPromises);  res.json({ success: true, message: '合併排序和分數分配成功儲存' });  } catch (error) {  console.error(error);  res.status(500).json({ error: '伺服器錯誤' });  }  });  module.exports = router; | |
| 程式名稱 | student.js |
| 目的 | 學生資料 |
| const express = require('express');  const router = express.Router();  const pool = require('../../lib/db');  // 根據 prono 獲取項目及相關學生資料  router.post('/:prono', async (req, res) => {  try {  const prono = req.params.prono;  const userno = req.body.userno || req.session.user.userno; // 從 session 中取得 userno  const [rows] = await pool.query(`  SELECT \*  FROM \`student-project\`.project p  JOIN collaborator c ON c.prono = p.prono AND c.userno = ?  JOIN assignment a ON a.colno = c.colno  JOIN student s ON s.stuno = a.stuno  LEFT JOIN resume r ON r.stuno = s.stuno  LEFT JOIN autobiography au ON au.stuno = s.stuno  LEFT JOIN studetails sd ON sd.stuno = s.stuno  JOIN evaluations e ON e.assno = a.assno  WHERE p.prono = ?  order by e.evano ASC;  `, [userno, prono]);  // 查詢協作者已完成評分的學生數量  const [completedEvaluations] = await pool.query(`  SELECT COUNT(e.evano) AS completed\_count  FROM \`student-project\`.evaluations e  JOIN assignment a ON a.assno = e.assno  JOIN collaborator c ON a.colno = c.colno  WHERE e.ranking IS NOT NULL AND c.prono = ?;  `, [prono]);  // 查詢協作者負責的總學生數  const [totalStudents] = await pool.query(`  SELECT COUNT(a.stuno) AS total\_count  FROM \`student-project\`.assignment a  JOIN collaborator c ON a.colno = c.colno  WHERE c.prono = ?;  `, [prono]);  // 完成比例計算  const completionRate = (completedEvaluations[0].completed\_count / totalStudents[0].total\_count) \* 100;  // 返回查詢結果  res.json({  rows,  completed\_count: completedEvaluations[0].completed\_count,  total\_count: totalStudents[0].total\_count,  completion\_rate: completionRate.toFixed(2) + '%'  });  } catch (error) {  console.error(error);  res.status(500).json({ error: '獲取資料失敗' });  }  });  module.exports = router; | |

第 10 章 測試模型

10-1 測試計畫

1. 登入：在登入介面點選「GOOGLE LOGO」按鈕，連動學校帳號登入。
2. 帳號設定：點選頭像姓名。
   1. 停用：點選「停用」的圖示按鈕，並確認停用此帳號。
   2. 登出：點選「登出」按鈕。
3. 帳號管理：點選「帳號管理」按鈕，此功能僅提供給管理者使用。
   1. 分配權限：先點選「鉛筆」的圖示按鈕後，再點選「職稱」的下拉式選項更改該使用者為管理者、助教、老師，最後按打勾的圖示按鈕儲存設定。
   2. 強制停用：點選「停用」的圖示按鈕，並確認停用此帳號。
4. 歷史專案：點選「觀看歷年專案」按鈕。
   1. 篩選學制：點選「學制」的下拉式選單，選擇全部學制、博士、二技、碩士、四技各學制。
   2. 篩選學年：點選「學年」的下拉式選單，選擇全部學年度或是各學年。
   3. 查看專案內容：點選清單上的專案名稱。
5. 管理專案：點選「我的專案」按鈕。
   1. 新增：點選「＋」圖示按鈕，按步驟新增專案名稱、學制、選擇階段日期、結束日期、上傳檔案、選擇錄取人數、審核方式，此功能僅提供給助教使用。
   2. 編輯：點選「齒輪」圖示按鈕，可修改專案名稱、學制、審核方式、錄取人數、第一階段日期、結束日期、重新上傳檔案、新增協作老師，此功能僅提供給助教使用。
   3. 刪除：點選「垃圾桶」圖示按鈕，並確認刪除此專案，此功能僅提供給助教使用。
6. 排序：點選清單中的專案名稱。
   1. 查看摘要：點選學生，右邊摘要欄顯示系統為該學生的自傳履歷所整理出相關備注。
   2. 查看該學生PDF檔案：點選「展開PDF」按鈕。
   3. 排序學生名次：在清單列上下拖曳學生資料，更改學生錄取名次，點選「完成」按鈕即儲存設定。
7. 合併排序：點選專案上「合併排序」產生合併後名單，此功能僅提供給助教使用。
8. 二次排序：在清單列上下拖曳學生資料，更改學生錄取名次，點選「完成」按鈕即儲存設定。
9. 產生最終名單：點選「產生名單」按鈕，此功能僅提供給助教使用。

10-2 測試個案與測試結果資料

* 表 10-2-1 登入測試結果

|  |  |
| --- | --- |
| 功能名稱 | 1. 登入 |
| 測試流程 | 1. 在預設的登入介面上點選GOOGLE的LOGO圖示。 2. 使用學校的GOOGLE帳號登入。 3. 畫面彈跳出登入成功後，自動跳轉到首頁。 |
| 預期結果 | 自動將此帳號註冊，同時在資料庫中成功新增使用者資料。 |
| 執行結果 | 測試成功，執行結果與預期結果一致。 |

* 表 10-2-2 帳號設定測試結果

|  |  |
| --- | --- |
| 1. 帳號設定 | |
| 功能名稱 | 2.1 停用 |
| 測試流程 | 1. 點選姓名頭貼按鈕。 2. 點選靠右的停用圖示按鈕。 3. 確定停用此帳號。 |
| 預期結果 | 使用者成功停用此帳號，資料庫的使用者狀態改為已停用。 |
| 執行結果 | 測試成功，執行結果與預期結果一致。 |
| 功能名稱 | 2.2 登出 |
| 測試流程 | 1. 點選姓名頭貼按鈕。 2. 點選「登出」按鈕。   或   1. 在首頁點選右上角離開的圖示按鈕。 |
| 預期結果 | 成功登出並跳轉至登入畫面。 |
| 執行結果 | 測試成功，執行結果與預期結果一致。 |

* 表 10-2-3 帳號管理測試結果

|  |  |
| --- | --- |
| 1. 帳號管理 | |
| 功能名稱 | 3.1 分配權限 |
| 測試流程 | 1. 系統管理者點選「帳號管理」按鈕。 2. 點選鉛筆圖示按鈕進入編輯畫面。 3. 選擇要更改的使用者，並點選職稱的下拉式選項。 4. 選擇更改的權限職稱。 5. 點選打勾的圖示按鈕儲存本次編輯。 6. 若要放棄編輯，則點選「X」的圖示按鈕。 |
| 預期結果 | 該使用者可以完整瀏覽權限看的畫面。 |
| 執行結果 | 測試成功，執行結果與預期結果一致。 |
| 功能名稱 | 3.2 強制停用 |
| 測試流程 | 1. 系統管理者點選鉛筆圖示按鈕進入編輯畫面。 2. 點選使用者清單列的停用圖示按鈕。 3. 確認停用此使用者帳號。 |
| 預期結果 | 使用者無法使用此帳號，且資料庫的使用者狀態改為已停用。 |
| 執行結果 | 測試成功，執行結果與預期結果一致。 |

* 表 10-2-4 歷史專案測試結果

|  |  |
| --- | --- |
| 1. 歷史專案 | |
| 功能名稱 | 4.1 篩選學制 |
| 測試流程 | 1. 點選「觀看歷年專案」按鈕。 2. 點選「學制」下拉式選單。 3. 點選希望看見的學制。 |
| 預期結果 | 清單只顯示有關該學制的所有專案。 |
| 執行結果 | 測試成功，執行結果與預期結果一致。 |
| 功能名稱 | 4.2 篩選學年 |
| 測試流程 | 1. 點選「學年」下拉式選單。 2. 點選希望看見的年度。 |
| 預期結果 | 清單只顯示該年度範圍的所有專案。 |
| 執行結果 | 測試成功，執行結果與預期結果一致。 |
| 功能名稱 | 4.3 查看專案內容 |
| 測試流程 | 1. 點選該專案名稱。 2. 畫面跳轉至該專案錄取名單。 |
| 預期結果 | 顯示該專案最終完成的所有排序。 |
| 執行結果 | 測試成功，執行結果與預期結果一致。 |

* 表 10-2-5 管理專案測試結果

|  |  |
| --- | --- |
| 1. 管理專案 | |
| 功能名稱 | 5.1 新增 |
| 測試流程 | 1. 助教點選「我的專案」按鈕。 2. 點選「＋」圖示按鈕。 3. 根據彈跳視窗的步驟去新增各細項。 4. 畫面挑轉到編輯畫面，確認新增資料有無錯誤。 |
| 預期結果 | 資料庫成功新增各欄位資料及相關聯欄位。 |
| 執行結果 | 測試與預期結果有偏差，因硬體設備或程式優化不足，造成AI生成時間過長。 |
| 功能名稱 | 5.2 編輯 |
| 測試流程 | 1. 助教點選齒輪圖示按鈕。 2. 修改錯誤欄位內容。 3. 新增協作老師。 |
| 預期結果 | 1. 欄位在資料庫更改成功。 2. 將專案分配給已新增的協作老師。 |
| 執行結果 | 測試成功，執行結果與預期結果一致。 |
| 功能名稱 | 5.3 刪除 |
| 測試流程 | 1. 助教點選垃圾桶圖示按鈕。 2. 確認刪除此專案。 |
| 預期結果 | 資料庫所有與此專案相關內容都刪除成功。 |
| 執行結果 | 測試成功，執行結果與預期結果一致。 |

* 表 10-2-6 排序測試結果

|  |  |
| --- | --- |
| 1. 排序 | |
| 功能名稱 | 6.1 查看摘要 |
| 測試流程 | 1. 點選專案名稱。 2. 在排序名單上點選學生。 3. 右邊摘要欄顯示該學生相關摘要。 |
| 預期結果 | 從資料庫找出AI生成的備注，顯示在畫面上供老師參考。 |
| 執行結果 | 測試成功，執行結果與預期結果一致。 |
| 功能名稱 | 6.2 查看該學生的PDF檔案 |
| 測試流程 | 1. 點選學生姓名。 2. 點選「展開PDF」按鈕。 3. 畫面跳轉至該學生所上傳的PDF檔案。 |
| 預期結果 | 從資料庫抓取出學生上傳的原檔案。 |
| 執行結果 | 測試成功，執行結果與預期結果一致。 |
| 功能名稱 | 6.3 排序學生名次 |
| 測試流程 | 1. 點選要改變名次的學生姓名。 2. 拖曳向上或向下。 3. 排序結束後，點選「完成」按鈕儲存設定。 |
| 預期結果 | 新的排序名單成功儲存至資料庫。 |
| 執行結果 | 測試成功，執行結果與預期結果一致。 |

* 表 10-2-7 合併排序測試結果

|  |  |
| --- | --- |
| 功能名稱 | 1. 合併排序 |
| 測試流程 | 1. 當第一階段結束後，所有協作老師完成排序。 2. 點選專案上的「合併排序」按鈕。 3. 系統自動合併所有老師的排序名單。 4. 產生新的排序名單。 |
| 預期結果 | 1. 將所有學生名單合併成功。 2. 產生錄取臨界值。 |
| 執行結果 | 測試成功，執行結果與預期結果一致。 |

* 表 10-2-8 二次排序測試結果

|  |  |
| --- | --- |
| 功能名稱 | 1. 二次排序 |
| 測試流程 | 1. 顯示合併排序後產生臨界值名單。 2. 各老師討論後。 3. 點選要改變名次的學生姓名。 4. 拖曳向上或向下。 5. 排序結束後，點選「完成」按鈕儲存設定。 |
| 預期結果 | 最終錄取排序成功回傳存取至資料庫。 |
| 執行結果 | 測試成功，執行結果與預期結果一致。 |

* 表 10-2-9 產生最終名單測試結果

|  |  |
| --- | --- |
| 功能名稱 | 1. 產生最終名單 |
| 測試流程 | 1. 所有排序皆完成後。 2. 點選專案上的「產生名單」按鈕。 3. 畫面跳轉至最終錄取名單及所有學生分數。 |
| 預期結果 | 1. 成功產生最終名單 2. 系統平均給予所有學生評分 |
| 執行結果 | 測試成功，執行結果與預期結果一致。 |

第 11 章 操作手冊

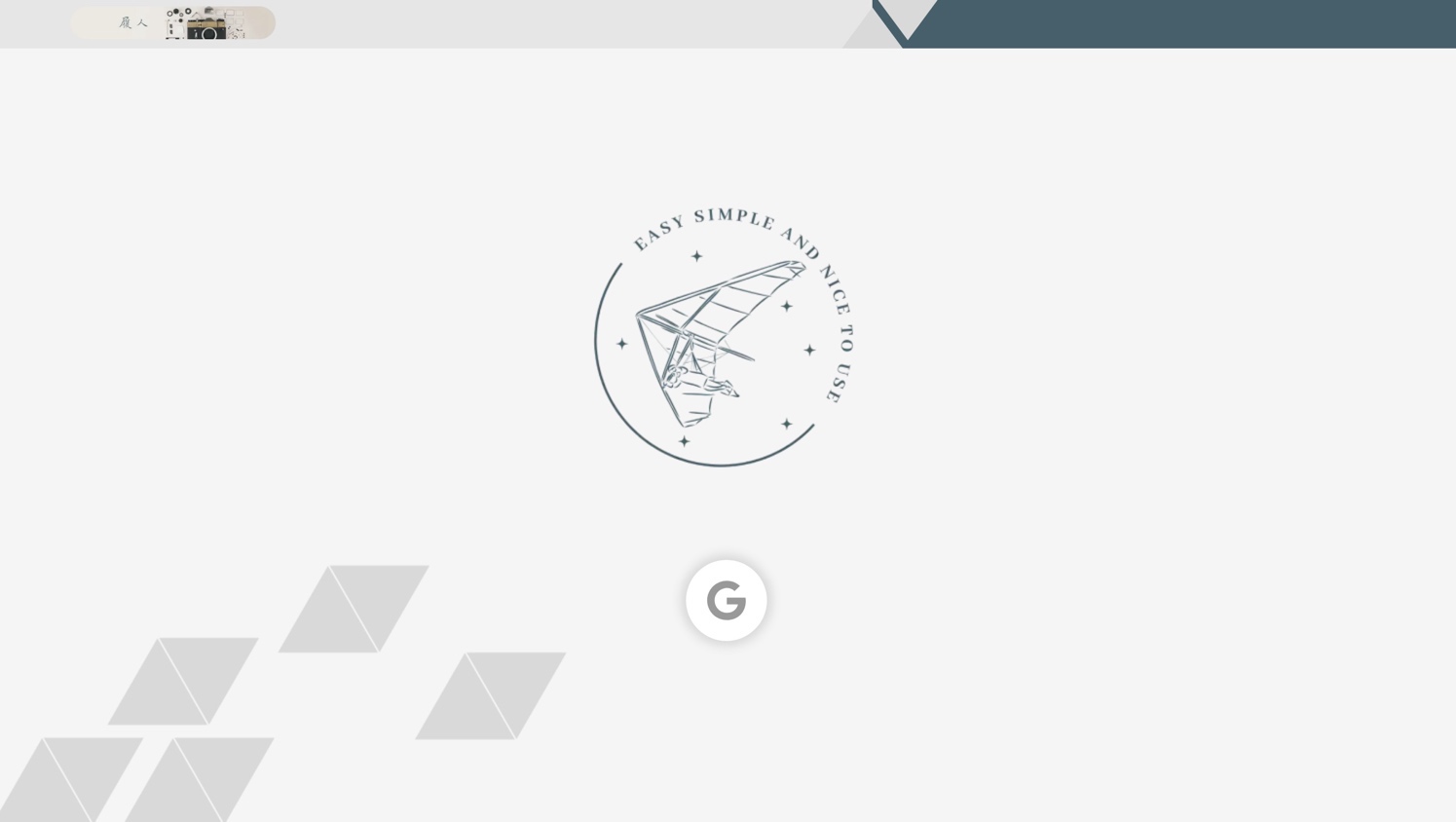
* 表 11-1 系統安裝元件資訊

|  |  |
| --- | --- |
| 系統資訊 | |
| 系統名稱 | 履人 |
| 版本 | 第一版 |
| 軟體系統要求 | Linux 系統 |
| 硬體系統要求 | ram 48以上 輝達gpu(支援cuda) |

第 12 章 使用手冊

* 表12-1 登入

|  |
| --- |
| 使用學校Google帳號登入 |





* 12-2 登入失敗

|  |
| --- |
| 未使用學校Google帳號登入，重新跳回登入畫面 |



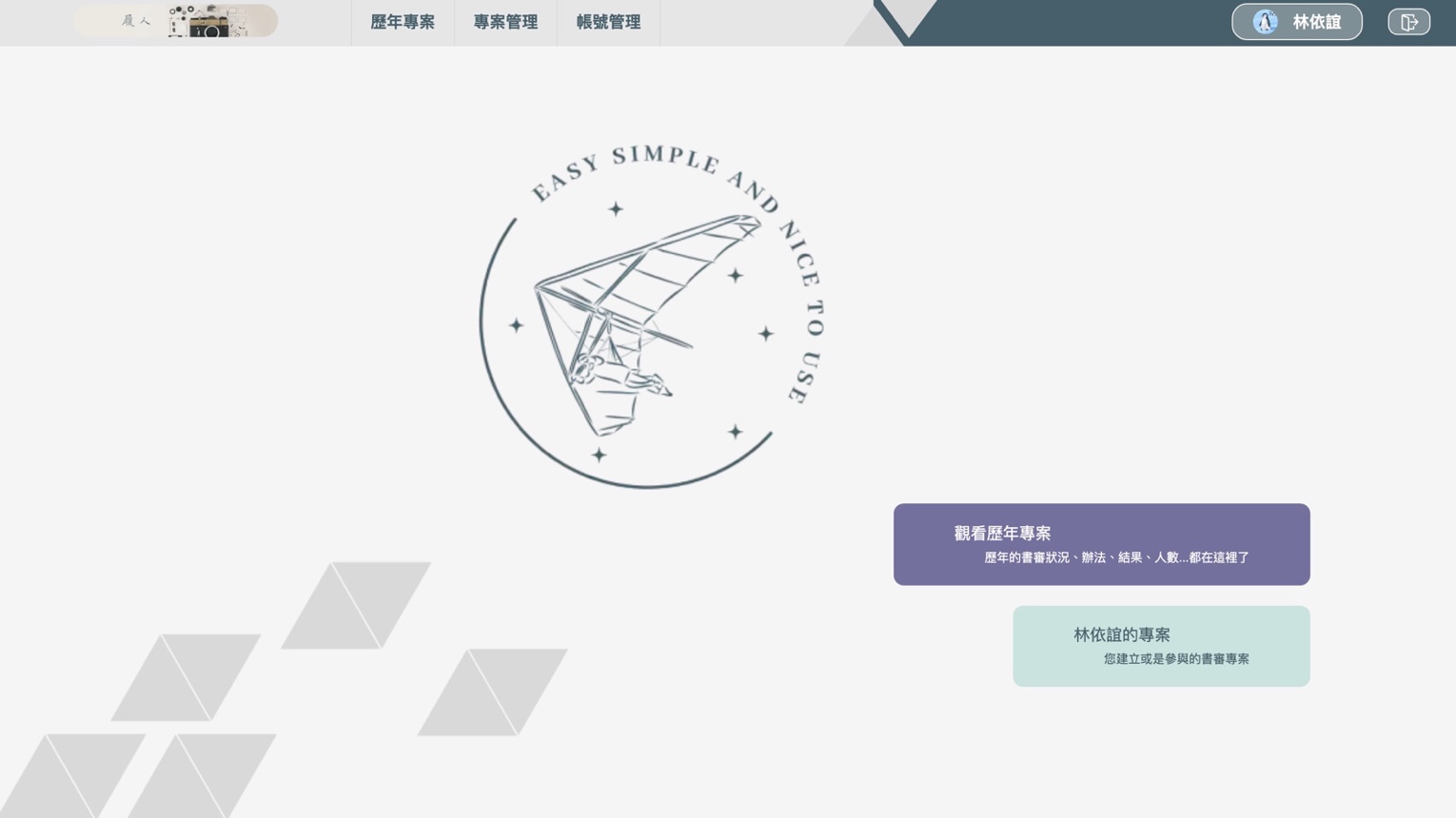
* 表12-3 登入成功

|  |
| --- |
| 已成功處理登入請求 |



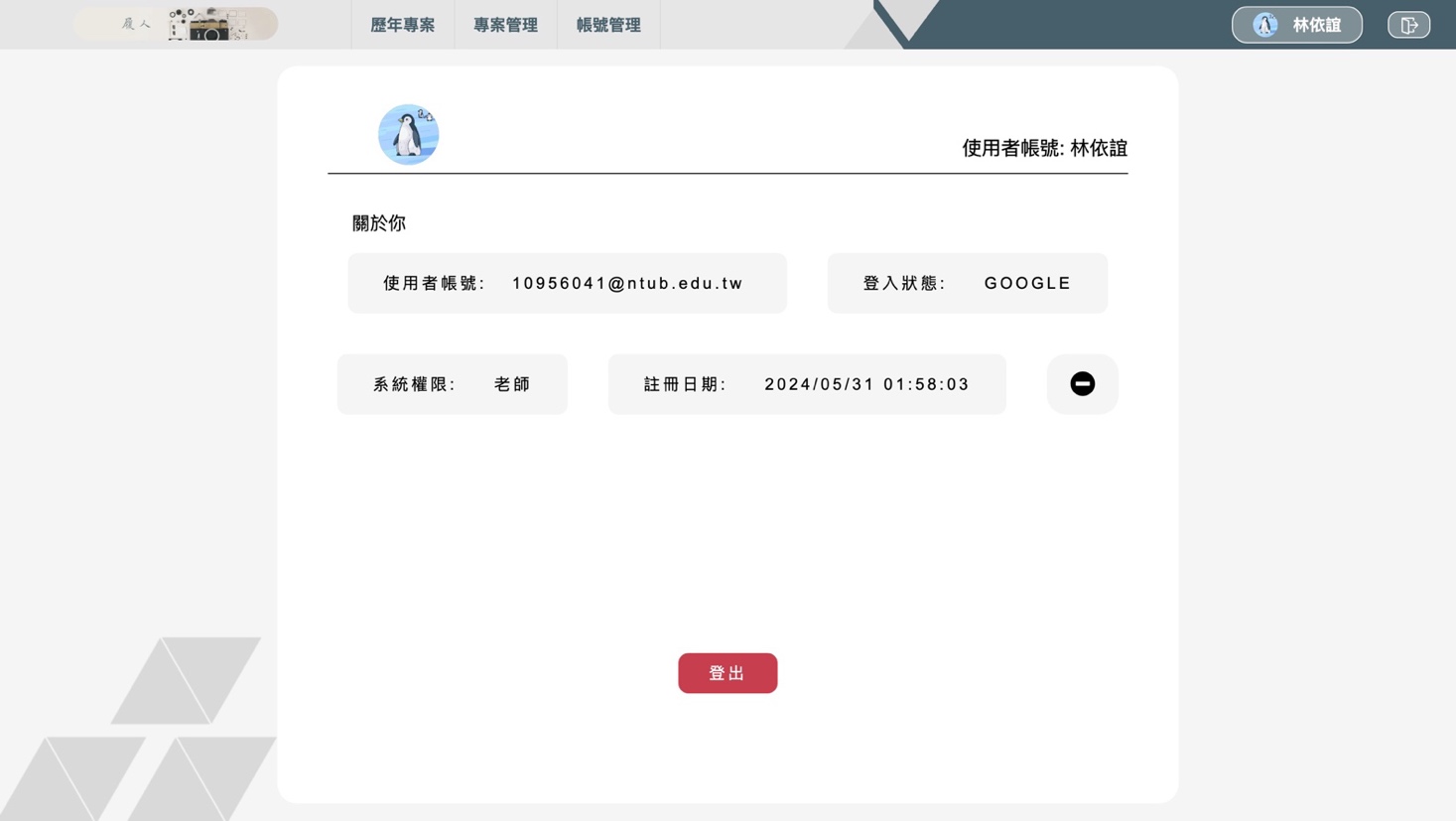
* 表12-4 主畫面

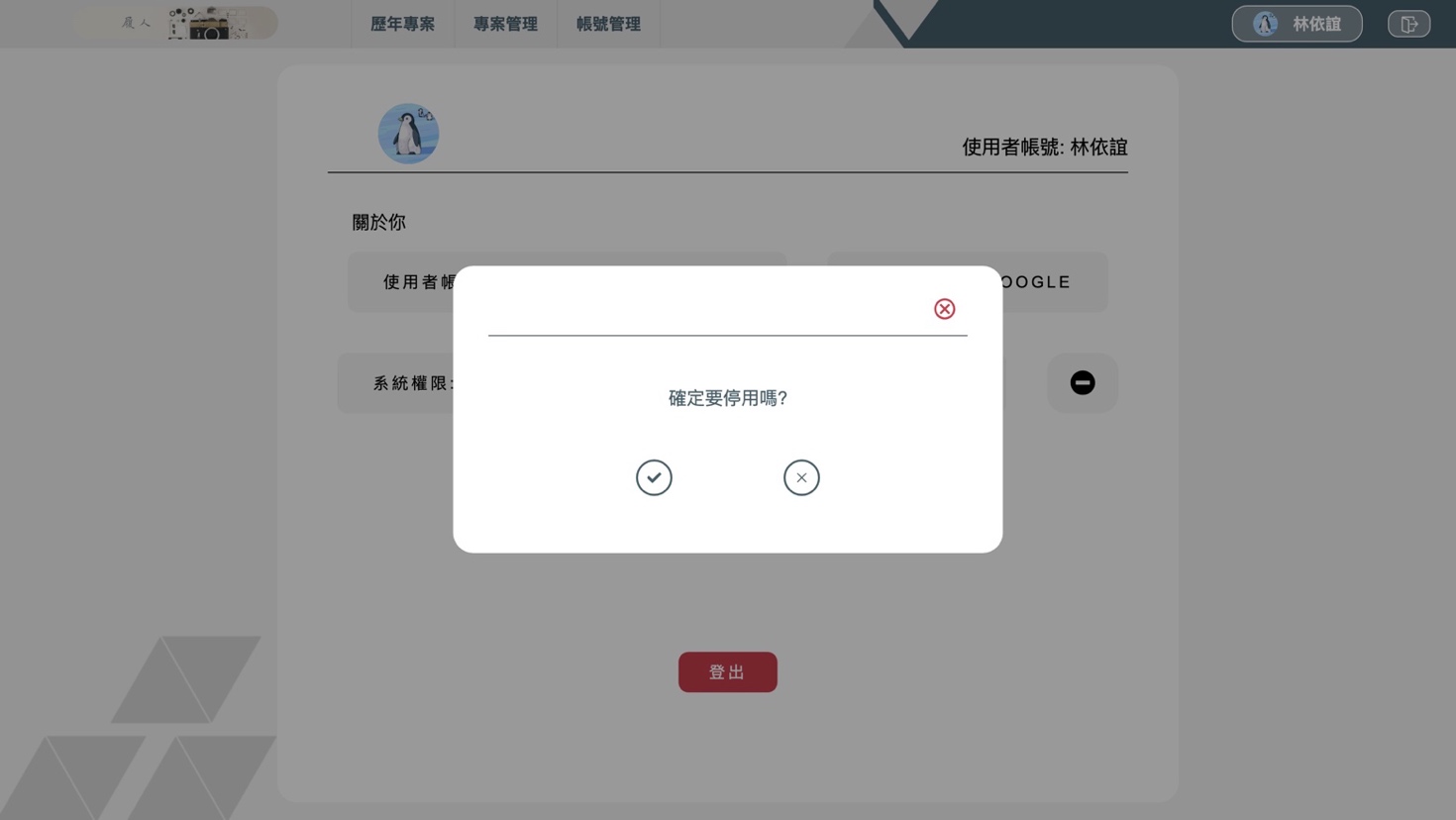
|  |
| --- |
| 登入成功後會跳到主畫面，選擇要使用的功能 |



* 表12-5 帳號畫面

|  |
| --- |
| 可以查看自己帳號的相關資料，  點選註冊日期旁的圖示按鈕可以自行停用此帳戶；  點選右上角離開的圖示按鈕或是紅色的登出按鈕都能成功登出系統。 |





* 表12-6 歷年專案

|  |
| --- |
| 查看歷年的專案評分，  可以點選學制、年度的下拉式選單，選擇自己想看到的專案範圍；  點選專案清單中的名稱就能看見此專案的最終排序，  點選展示內容會在右邊展示區顯示該學生的備注內容，  點選展示PDF則是該學生所上傳的所有檔案內容。 |





* 表12-7 我的專案

|  |
| --- |
| 權限為助教的使用者可以在此畫面點選編輯按鈕及刪除按鈕，  在第一階段過後會顯示合併排序的按鈕，  在第二階段過後會顯示產生名單的按鈕；  權限為老師的使用者在此畫面只能點選專案名稱進入排序畫面。 |



* 表12-8 新增專案

|  |
| --- |
| 權限為助教點選專案總覽「＋」的圖示按鈕，  依照步驟新增專案細項，直到最後跳出新增完成，  點選完成後，畫面會跳轉至專案設定，  檢查已新增內容及新增協作老師。 |











* 表12-9 編輯專案

|  |
| --- |
| 權限為助教點選專案列表的齒輪圖示按鈕，進入編輯畫面；  點選圖示按鈕可進入各個項目的編輯彈跳視窗，  點選右下角協作老師的圖示按鈕後會跳出所有權限為老師的名單，  點擊老師名字並點選完成按鈕即可新增協作老師。 |















* 表12-10 刪除專案

|  |
| --- |
| 權限為助教，點選垃圾桶的圖示按鈕。 |



* 表12-11 排序學生

|  |
| --- |
| 權限為老師點選專案列表的專案名稱，  左側部分透過拖移方式進行學生排序，  點選學生姓名會在右側顯示AI為該學生文件所生成的備注內容，  再點選展開PDF按鈕會顯示該學生所上傳的原文件內容，  最後排序完成，點選完成按鈕，即完成此次專案評分。 |



* 表12-12 帳戶管理

|  |
| --- |
| 權限為管理者點選帳號管理，  點選職稱、狀態的下拉式選單即可篩選人員，  在右邊的搜尋欄位輸入姓名或電子郵件也可找到該人員。 |

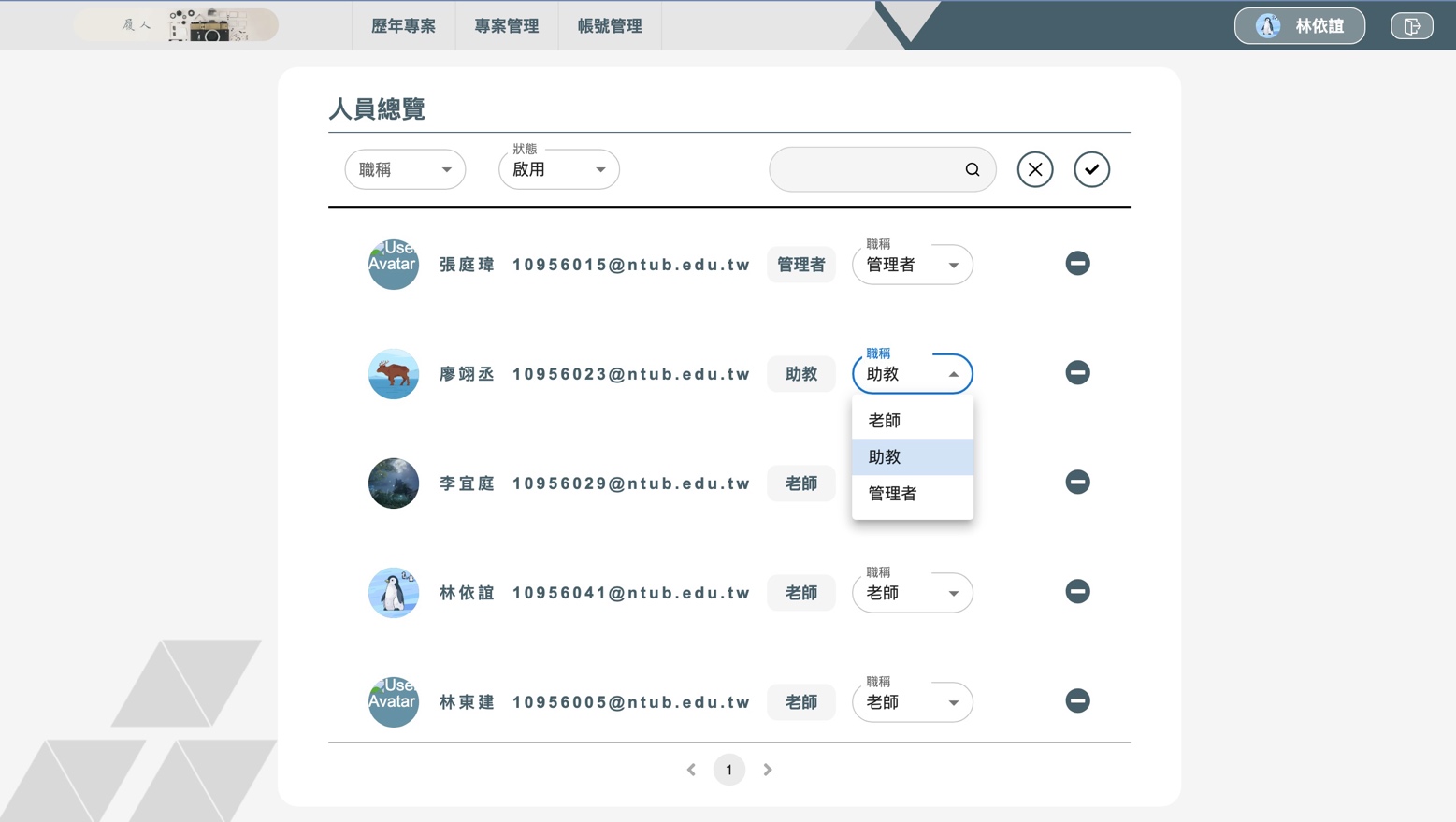


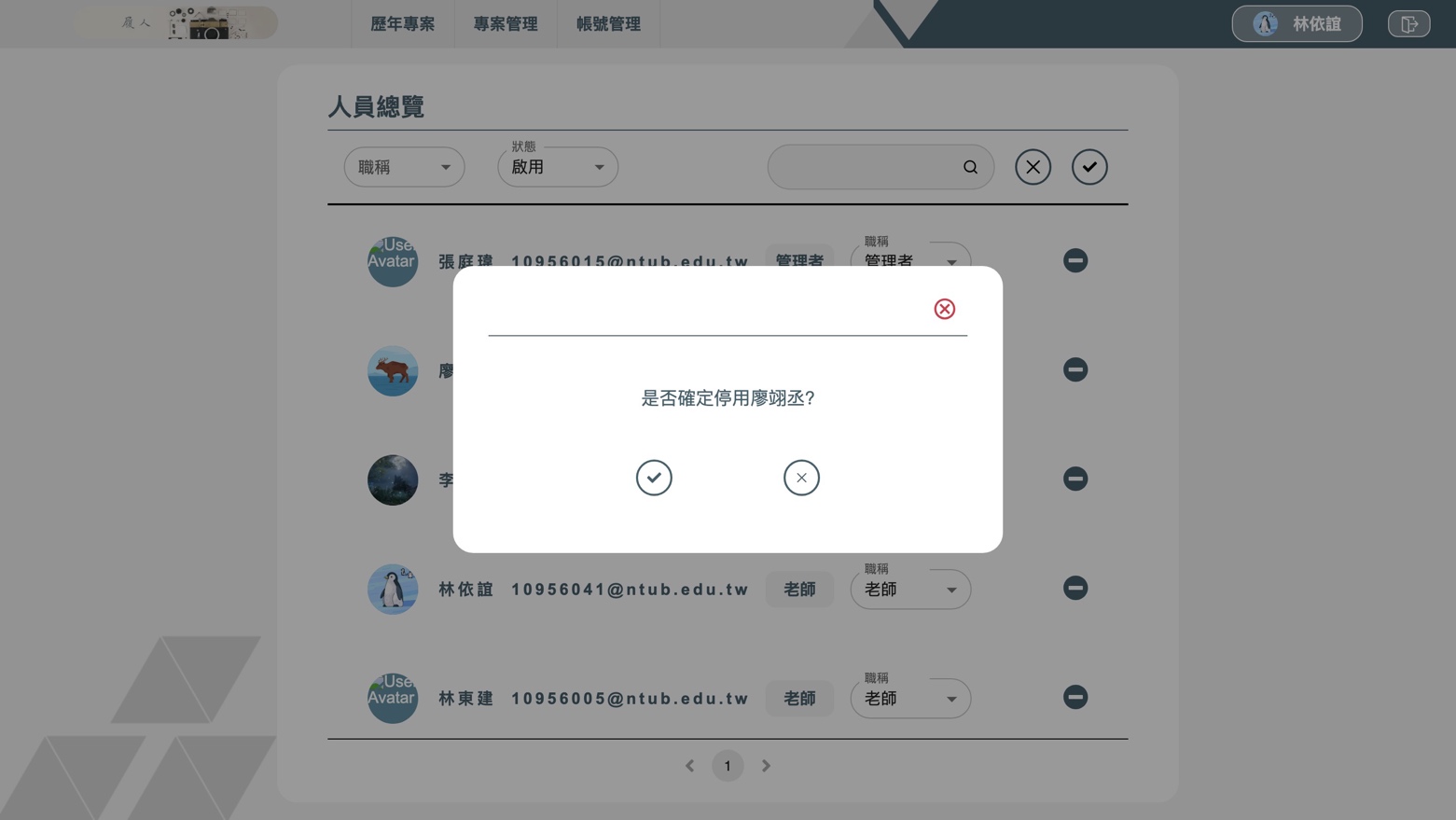




* 表12-13 編輯權限

|  |
| --- |
| 權限為管理者點選右上角的編輯圖示按鈕，  可以透過下拉式選單更改該使用者的職稱，  點選右邊的停用圖示按鈕可以停用該帳戶。 |





第 13 章 感想

10956005 林東建

在這次專題製作過程中，我負責了與 AI 技術相關的部分。雖然我們專題成員都是同一科系的學生，但由於我對 AI 技術不甚熟悉，還是花了很多時間去學習與研究。為了選擇適合專題需求的模型，我比對了許多開源模型，最終決定採用 Llama3 並進行進一步訓練與優化。

這段過程中，我學會了如何調整模型參數以適應特定場景的需求，同時也逐漸理解了模型訓練與測試的精細度與複雜性。雖然過程中經常遇到困難，但每當成功解決問題時，那份成就感讓我對 AI 領域充滿了興趣與信心。

透過這次專題，我提升了 AI 技術的知識與應用能力。這次經驗讓我意識到 AI 技術對未來科技發展的重要性，並激發了我持續深入學習的動力。我也學會以更全面的視角看待問題，並能更加靈活地運用所學知識去解決實際問題。

10956015 張庭瑋

這次的專題製作過程，讓我深刻體會到團隊合作的重要性。整個開發過程中，我們依靠團隊間的彼此配合，發揮了各自的專長。每個人負責的部分相互補足，使我們順利完成這個充滿挑戰的專題。

在進行過程中，溝通扮演了至關重要的角色。無論是面對技術難題，還是設計方向的調整，我們都透過會議和即時溝通，快速解決問題，確保專題的進度不會落後。遇到不同的意見時，我們也學會了透過理性的討論找到最佳解決方案，讓專題朝著正確的方向前進，也讓我學到了如何更好地協調與他人合作。

此外，這次專題讓我能夠將過去所學的知識和技能，像是系統設計和資料庫管理，實際應用到專案中。然而，我也學習了不少新技術，像是新的程式語言和 Git 版本控制的使用。這些技術讓我們的開發過程更加順暢。

總結來說，這次專題製作不僅讓我深刻體驗到團隊合作與溝通的重要性，也讓我學會了新的技術，並成功將其應用到實際專案中。這段經歷對我來說真是充滿挑戰且意義非凡的成長歷程。

10956023 廖翊丞

10956029 李宜庭

在這次為期兩學期的專題中，我學到了很多課堂上無法接觸到的寶貴經驗，無論是人際溝通、組員協作，還是技術上的挑戰，這些都讓我獲益匪淺。

專題初期，我們與多位老師討論需求。在這個過程中，我們遇到了一些困難，例如老師們對資安有顧慮，並質疑系統的實際應用價值。這些挑戰促使我學會如何傾聽用戶的需求，並從中過提取出關鍵的系統功能。

隨著專題的進行，組員之間的摩擦逐漸浮現，特別是在一評臨近的壓力下，衝突變得頻繁。我很慶幸我們能夠坦誠溝通，也感謝組長的鞭策，讓我認識到自身的不足。同時，也深深感受到每位組員對團隊的貢獻是不可或缺的，缺少任何一人，專題都無法順利完成。

在設計UI時，非常感謝組員們的信任，讓我有機會放手一試，並且在過程中給予意見。這讓我學會了如何在美觀與UX之間取得平衡。

在撰寫程式的過程中，我也意識到自己在架構設計、程式語言掌握以及debug方面的不足。這次專題並非學習的終點，而是一個嶄新的開始，還有很多值得我深入學習的知識等著我去學習。

10956041 林依誼

在製作專題的過程中，我們面臨了許多交叉路口。每次危機的解決，都再一次證明了團隊合作的重要性。不論是一起討論問題並合作解決，還是耐心聆聽組員們的想法與困難，我們都在技術上互補成長。這段旅程同時也是對自我的一項新挑戰，它讓我能夠將前三年所學與這一年來的技術精進相結合，產生新的突破。

具有挑戰性的是我接觸了前三年未深入學習的程式語言。在撰寫程式的過程中，我遇到了許多錯誤和挑戰，迫使我在技術上快速學習，還促使我深入思考每一個解決方案。

在進行專題的過程中，我們不僅要完成自己的進度，還需要在團隊中相互幫助。透過交流技術問題以及溝通細節的修改，促使專題逐漸成長。

相信有了這次專題難忘的經驗，能使我們未來面對團隊合作時，更有耐心和條理地進行溝通。同時，也讓自己在上台分享成果時更有自信及勇氣。

第 14 章 參考資料

https://www.youtube.com/watch?v=LEwi44cWBu8&list=PLpZ8gOBZmTy5p5l55u-Kup74LLH8cJCnB

https://www.youtube.com/watch?v=DeGpnhod\_nU&t=2s

https://www.youtube.com/watch?v=aBTiZfShe-4

https://www.youtube.com/watch?v=7kXh2EvFlLs&t=1131s

https://www.youtube.com/watch?v=l789xaPU1WM&t=464s

附錄

|  |  |
| --- | --- |
| 評審建議事項 | 修正情形 |
| 1. 表3-3-1是否應將使用到的模型LAMA2 列入。目前已釋出LAMA3，可評估使用。 2. AI工具可升級用GPT3.5或4等新版 3. 提供競賽等級資料，供評審老師參考 4. 未明確提到轉入分析的學生資料檔案格式為何？若為多頁PDF 或 PDF 內容為圖片呈現文字資料。 5. 商業模式9大區塊有對應位置，請修正文件表2-2-1。 6. AI分析完後，建議加上審核機制，確認分析資料的正確性，以確保提供老師資料的完整無誤。 7. 建議將需求訪談納入說明與整理。 | 1. 已加入表格中程式開發語言的AI模型部分。目前AI訓練已轉換使用LAMA3。 2. 參閱附錄一。 3. 後端有使用Google Custom Search API，從外部抓取競賽等級相關資料，提供參考資料輔助老師。 4. 我們目前支援的格式包括多頁PDF及由圖片形式呈現的PDF內容。我們訓練的模型能夠有效辨識檔案中的文字或圖像中正楷字體的文字部分，而針對一些特殊字體的辨識，目前仍存在些許限制。 5. 已修正 6. AI所提供的備註資料僅供老師參考。 7. 參閱附錄二。 |

附錄一

* 附錄表一AI模型比較

|  |  |  |
| --- | --- | --- |
| 項目 | Llama 3 | GPT-3.5/4 |
| 訓練模型架構 | Llama是開源模型，允許自行下載並訓練。 | GPT是OpenAI的專有模型，不開放源代碼。 |
| 資料隱私保護 | 可完全自行管理模型訓練數據，確保個資安全。 | 使用GPT需經過OpenAI伺服器，存在資料傳輸風險。 |
| 模型部署方式 | 可本地部署，不需依賴外部伺服器。 | 主要基於雲端部署，無法本地化控制。 |
| 調整靈活性 | 本地訓練，高訂製化。 | 預訓練，低訂製化。 |
| 模型表現 | 依訓練數據和參數設計，表現可靈活調整。 | GPT經過大量訓練，表現穩定，不易針對專案需求調整。 |
| 應用範圍 | 適合對資料隱私極敏感的應用。 | 適合快速開發和部署，但有資料外部處理風險。 |

附錄二

訪談內容：了解老師們在審核學生資料時所遇到的困難。

* 附錄表二 問題及解決

|  |  |
| --- | --- |
| 提供/發現問題 | 我們已提供的解決方式 |
| 1. 老師在審核時需一次閱讀大量學生文件。 2. 一次大量閱讀後，難以記住每個學生的文件內容。 3. 評分及閱讀文件時，需來回切換程式，操作不便。 4. 助教需統整所有老師的評分資料並平均分數，且分數不能重複。 | 1. 系統隨機分配學生文件，讓每位老師平均審閱相同數量的文件。 2. 使用AI生成內容備注供老師參考。 3. 系統將拖曳排序和審閱文件功能合併為單一畫面，提升操作便捷性。 4. 系統自動整合所有老師的評分資料，並在規定範圍內平均分數，直接篩選錄取學生。 |