**國立勤益科技大學**

**資訊工程學系**

**Web程式設計-期末分組作業**

**記帳網站**

**組別8**

**指導老師：朱孝國**

**組員：3B117156　許博翔**

**3B117139　陳龍耀**

**3B017012　李柏駿**

**中華民國　　114　年　6　月　13　日**

# 摘要

本專案為一個全功能、響應式且安全的記帳網站，前端採用 Vue 3，後端以 Flask 架構開發。支援個人與群組記帳，具備完整的收支 CRUD、進階篩選與分頁機制。採用 JWT(JSON Web Token) 實現身份驗證，並優化會話過期處理，提升使用體驗。群組功能支援成員邀請、權限管理與動態 UI 同步更新，確保協作流暢。網站提供互動式儀表板，呈現財務趨勢與類別分布，並以 Tailwind CSS 打造優雅的跨裝置介面。技術層面涵蓋 PostgreSQL 相容性調整、非同步錯誤處理等實務挑戰，展現完整全端開發能力。

未來的功能擴展，如多幣種支援與:註冊/登入信箱驗證，匯入/匯出功能（Excel/CSV)，主題切換功能。

作品網站連結： <https://yahoo0976095579.github.io/Accounting-website/>

**關鍵詞**：記帳網站、Vue3、PostgreSQL、個人記帳、群組記帳、財務儀表板、類別管理、即時更新、TailwindCSS、Axios、Flask。

目錄

[摘要 1](#_Toc200396426)

[目錄 2](#_Toc200396427)

[第一章緒論 3](#_Toc200396428)

[1.1前言 3](#_Toc200396429)

[1.2文獻探討 3](#_Toc200396430)

[第二章作業說明 4](#_Toc200396431)

[2.1學習目標 4](#_Toc200396432)

[2.2預期效益 4](#_Toc200396433)

[2.3環境介紹 4](#_Toc200396434)

[第三章實作方法 5](#_Toc200396435)

[3.1環境檢查 5](#_Toc200396436)

[3.2安裝與設定 6](#_Toc200396437)

[3.3系統設計與實作步驟 7](#_Toc200396438)

[3.3.1建立資料庫與資料表 7](#_Toc200396439)

[3.3.2設計圖形化介面 7](#_Toc200396440)

[3.3.3個人記帳與儀表板 8](#_Toc200396441)

[3.3.4分組記帳與類別管理 9](#_Toc200396442)

[3.3.5錯誤處理與使用者提示 10](#_Toc200396443)

[第四章實作心得 10](#_Toc200396444)

[4.1心得 10](#_Toc200396445)

[4.2結論 11](#_Toc200396446)

[參考文獻 13](#_Toc200396447)

第一章緒論

1.1前言

隨著數位化生活的普及，個人與家庭對財務管理的需求日益提升，而一個方便、安全且具協作功能的記帳工具也變得愈加重要。本專案旨在開發一套兼具使用者體驗與系統穩定性的記帳網站，提供個人收支管理與群組記帳協作等核心功能。透過現代化技術堆疊，前端採用 Vue 3 搭配 Tailwind CSS 打造流暢的介面體驗，後端以 Flask 架設 API 並整合 PostgreSQL 資料庫，實現穩定高效的資料處理與安全性管理。本系統除基本 CRUD 外，亦強化了認證機制、群組權限控管與視覺化財務報表，致力於成為一個實用且可擴展的全端記帳解決方案。

1.2文獻探討

傳統財務管理工具受限於單機操作、缺乏即時同步及協作功能。例如，Excel記帳需手動更新，難以多人協作；商業軟體如QuickBooks成本高且對個人用戶過於複雜。雲端應用如Mint提供預算和支出追蹤功能，但隱私問題與功能定制不足限制了其應用範圍。市場對簡單、協作性強且跨裝置的財務管理工具需求日益增長。

全面的預算和支出追蹤、投資及信用評分監控；YNAB強調零基預算方法，幫助用戶分配每一筆收入；QuickenSimplifi提供簡化的預算和帳單追蹤功能，著重於易用性此外，Monarch支援無限協作者，與本研究的群組記帳功能有部分相似之處。然而，這些應用多數專注於個人財務管理，缺乏群組協作功能。

另一項值得注意的研究是ResearchGate上的「PersonalFinanceManagementApplication」論文，該文描述了一款Android行動應用程式，包含預算管理、財務狀態報告、支出和收入追蹤、報告生成及條碼掃描功能雖然行動應用程式提供便攜性，但網頁應用程式如本研究的AccountingWebApp提供跨裝置的可訪問性，無需安裝即可使用，並可提供更強大的後端整合功能。本研究的AccountingWebApp與現有解決方案的主要區別在於其整合了個人與群組財務管理功能，填補了市場上的空白，同時強調即時數據更新和響應式設計，以提升用戶體驗。

第二章作業說明

2.1學習目標

本計畫旨在培養全端開發的綜合能力，並深入理解網頁應用程式的設計與實作流程。學習目標包括：

熟練運用 Vue 3、Flask 與 PostgreSQL，實現前端、後端與資料庫的整合；

實作基於 JWT 的身份驗證機制，解決會話管理的複雜性；

使用 Tailwind CSS 建立多設備皆適用的響應式介面，提升使用體驗；

掌握群組記帳中涉及的權限控管與資料一致性等複雜業務邏輯；

熟悉 Vercel 與 Render 等雲端部署流程，實現持續整合與部署；

培養處理資料庫相容性、即時資料更新與非同步錯誤等技術挑戰的問題解決能力。2.2預期效益

本作業預期可帶來多方面效益。

在使用者層面，提供一個功能完整、操作直覺的記帳平台，支援個人財務管理與多人協作，透過即時儀表板快速掌握財務狀況，提升管理效率與透明度。

在技術層面，驗證 Vue 3、Flask 與 PostgreSQL 所構成的現代全端技術堆疊於開發可擴充網頁應用中的可行性，並展現前後端分離架構的實用價值。

在教育層面，提供完整實務案例，涵蓋需求分析、系統設計、實作、測試與部署，有助於全端開發者的學習與成長。

在社會層面，藉由簡化財務管理流程，促進個人與群體財務健康，進而支持更理性的經濟決策。最後，透過團隊分工與協作，累積有效溝通與協調經驗，強化團隊合作能力。

2.3環境介紹

硬體環境

前端技術：

|  |  |
| --- | --- |
| Vue3 | 用於建置單頁應用，配合VueRouter實現路由管理。 |
| Pinia | 用於狀態管理，資料確保一致性和即時更新。 |
| TailwindCSS | 用於快速建立響應式介面。 |
| Axios | 用於傳送HTTP請求，處理API互動。 |

後端技術：

|  |  |
| --- | --- |
| PythonFlask | 輕量級框架，用於建立RESTfulAPI。 |
| SQLAlchemy | 用於物件關聯映射（ORM），簡化資料庫操作。 |
| Flask-JWT-Extended | 用於實作JWT身份驗證。 |
| WerkzeugSecurity | Python的一個安全工具庫，提供密碼哈希、驗證等功能 |

資料庫&部署平台：

|  |  |
| --- | --- |
| PostgreSQL | 關聯式資料庫，部署於 Render 平台，確保資料可靠存取與查詢效能。 |
| Vercel | 前端應用託管平台，支援持續整合與自動部署。 |
| Render | 託管後端 API 與 PostgreSQL 資料庫，提供高可用性與自動擴展。 |

開發工具：

|  |  |
| --- | --- |
| Visual Studio Code | 主要開發環境，支援前端和前端程式碼編寫。 |
| Postman | 用於測試API端點。 |
| Git | 用於版本控制，配合GitHub進行程式碼管理。 |
| SQLite3 | 輕量級且易於整合，適合本地開發與測試。 |

第三章實作方法

3.1環境檢查

在開始開發前，需確認環境配置完整且正確。  
Python 安裝：確認已安裝 Python 3.8 以上版本，並設定環境變數以便命令行使用。  
必要套件安裝：安裝 Flask 及相關 Python 套件；前端開發需安裝 Node.js（版本 16 以上）及 npm（版本 8 以上），用於 Vue 3 的建置與相依性管理。  
資料庫：本地開發環境使用輕量級 SQLite，方便快速測試；雲端部署則使用 PostgreSQL，確保資料庫的擴充性與穩定性。  
雲端平台：準備 Vercel 帳號用於前端部署，並連結 Git 倉庫；Render 帳號用於後端 API 與 PostgreSQL 資料庫部署，並設定必要環境變數。  
網路環境：須有穩定網路，以便下載依賴、測試 API 及部署應用。

完成以上環境檢查後，即可開始安裝與設定。

3.2安裝與設定

安裝 Python 套件：必要的套件。

安裝與設定分為前端、後端、資料庫和部署四個部分：

前端安裝：  
使用指令 npm create vue@latest 初始化 Vue3 專案，選擇 TypeScript 和 VueRouter。安裝核心依賴：npm install pinia tailwindcss axios vue-router。  
設定 TailwindCSS：產生 tailwind.config.js 和 postcss.config.js，並在主 CSS 檔案中引入 Tailwind 指令。  
配置 Pinia：建立 store 目錄，定義交易、類別與群組等狀態管理模組。  
設定 Axios：設置基礎 URL 與攔截器，用於處理 JWT token 及錯誤回應。

後端安裝：  
建立 Flask 專案，創建虛擬環境並執行 pip install flask flask-sqlalchemy flask-jwt-extended flask-cors python-dotenv 等套件。  
設定環境變數：使用 .env 檔案儲存敏感資訊，如資料庫連線字串、JWT 金鑰與密鑰。  
初始化 Flask 應用：配置 SQLAlchemy 與 Flask-JWT-Extended，確保資料庫連線與身份驗證功能正常運作。

資料庫設定：  
在Render平台建立PostgreSQL實例，取得連線字串格式為 postgresql://user:password@host:port/dbname。  
在 Flask 中使用該連線字串設定 SQLAlchemy。  
使用 Alembic 進行資料庫遷移管理，確保開發與生產環境資料庫結構一致。

部署設定：  
前端部署：在 Vercel 建立專案，連結 GitHub 儲存庫，並設定環境變數（如 API URL）。  
配置 vercel.json 以支援單頁應用程式的路由。  
後端部署：在 Render 建立 Web Service，指定 Python 環境並上傳程式碼。  
設定環境變數，確保與資料庫連線正常。  
資料庫部署：在 Render 啟用 PostgreSQL 實例，確保應用能正常存取資料庫。

3.3系統設計與實作步驟

### 3.3.1建立資料庫與資料表

### 開發初期於本地使用 SQLite 建立資料庫，確認無誤後轉用 PostgreSQL。核心資料表包括：

### 使用者（ID、Email、密碼雜湊、建立時間）、交易（ID、使用者ID、群組ID、金額、類別ID、交易類型、日期、描述）、類別（ID、使用者ID、名稱、類型）、群組（ID、名稱、管理員ID、建立時間）、群組成員（群組ID、使用者ID、角色）、邀請（ID、群組ID、發送者ID、受邀者ID、狀態、建立時間）。

### 資料表間以外鍵關聯，並以索引優化常用查詢。開發中將 SQLite 的 strftime 函數替換為 PostgreSQL 的 to\_char，以確保跨環境一致。

### 3.3.2設計圖形化介面

### 前端採用 Vue 3 和 Tailwind CSS，強調響應式、模組化與易用性。整體使用 Vue Router 實現單頁應用，導覽列包含首頁、儀表板、群組管理、類別管理與登出，頁面以 flexbox 和 grid 佈局，適應多種裝置。

### 核心組件有交易清單（支援分頁與多條件篩選，行動裝置切換為卡片模式）、儀表板（展示收入、支出與餘額圖及類別分佈）、模態視窗（新增/編輯交易、類別和邀請，風格統一）、動態表單驗證（如金額需為正數，類別依交易類型篩選）。響應式設計利用 Tailwind 斷點（sm、md、lg、xl）調整佈局與按鈕大小，提升小螢幕使用體驗。動畫部分使用 Vue 轉場元件實現模態窗與列表新增刪除的平滑過渡，增強互動感。

### 3.3.3個人記帳與儀表板

**個人帳**：

* **交易管理**：使用者執行交易的建立、讀取、更新和刪除（CRUD）操作。交易表單包含金額、類別、日期、描述和類型（收入/支出）欄位。

|  |  |  |
| --- | --- | --- |
|  |  |  |
| 新增交易 | 交易內容 | 收支內容 |

* **篩選與分頁**：支援多條件篩選（日期範圍、類別、金額區間），使用 Pinia 儲存篩選狀態。分頁功能透過下拉 API 實現，每頁顯示固定數量的交易。
* **即時更新**：交易新增或刪除後，透過 Pinia 的操作

**儀表板**：

* **統計互動**：顯示總收入、總支出和淨餘額，使用Tailwind CSS設計互動樣式，支援響應式排列。
* **趨勢圖**：展示日/週/月區間的收入與支出趨勢。使用者可透過下拉式選單切換時間範圍，數據資料動態更新。

|  |  |
| --- | --- |
| 一張含有 文字, 行, 繪圖, 螢幕擷取畫面 的圖片  AI 產生的內容可能不正確。 | 一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片  AI 產生的內容可能不正確。 |
| 趨勢圖 | 趨勢圖網頁 |

### 3.3.4分組記帳與類別管理

**群組記帳**：

* **群體創建與解散**：使用者可建立群體並設定名稱，僅管理員有權解散群體。解散操作需進行驗證，確保群體內未結清交易。
* **邀請管理**：支持發送、接受和拒絕邀請。現場檢查防止重複邀請，若使用者已收到邀請，更新狀態取消建立新記錄。
* **角色管理**：管理員可指派或取消成員角色。若嘗試刪除唯一管理員，則傳回錯誤並提示指定後繼者。
* **群體交易**：群體內交易支援獨立的CRUD操作，與個人交易佔用相同的篩選和分頁邏輯。交易記錄關聯群體ID，保證資料隔離。
* **安全退出**：成員可隨時退出團體，管理員退出需先轉讓管理權。其中驗證確保團體至少保留一名管理員。

**類別管理**：

* **自訂類別**：使用者可建立、編輯和刪除收入或支出類別。類別類型在建立時固定，透過按鈕選擇（收入/支出）避免錯誤輸入。

|  |  |
| --- | --- |
| 一張含有 文字, 螢幕擷取畫面, 數字, 字型 的圖片  AI 產生的內容可能不正確。 | 一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片  AI 產生的內容可能不正確。 |
| 類別新增 | 群組 |

* **表單連動**：交易表單的類別下拉式選單會根據交易類型動態篩選。
* **刪除保護**：若類別已被交易引用，則禁止刪除並傳回提示訊息，確保資料完整性。

### 3.3.5錯誤處理與使用者提示

**前端錯誤處理**：

* **Axios攔截器**：擷取HTTP錯誤（如401未授權），自動刷新過渡令牌或引導使用者重新登入。

|  |  |
| --- | --- |
|  |  |
| 完整的註冊、登入、登出流程 | 清除本地憑證 |

* **通知系統**：使用 Vue 的通知組件顯示錯誤訊息，例如“交易金額無效”或“網路連線失敗”，確保訊息且語音不會幹擾操作。

**額外錯誤處理**：

* **例外處理**：Flask 使用自訂異常處理器傳回標準化的 JSON 錯誤回應。
* **業務邏輯驗證**：對群組操作（如刪除唯一管理員）進行嚴格檢查，傳回特定錯誤訊息（如「請先指定新管理員」）。

**使用者提示**：

* **表單驗證**：即時驗證表單輸入，例如金額必須為正數，日期不得為未來時間。否則輸入觸發紅色邊框和提示文字。
* **操作確認**：敏感操作（如刪除交易或退出群體）顯示確認彈跳窗，確保使用者清楚明白。

**狀態通知**：群組邀請狀態變更時，顯示成功訊息（如「已接受邀請」），提升使用者體驗。

第四章實作心得

4.1心得

許博翔：這次期末作業讓我發現，做一個完整的系統其實需要準備很多東西，不只是課堂上學到的 SQL、Flask 和 CSS，像前端怎麼設計、後端怎麼串接，還有怎麼把網站部署到線上，這些都是新的學習。雖然我不是程式負責的人，但在過程中也接觸到很多技術，慢慢懂得系統怎麼跑、資料怎麼傳，對整個開發流程有了更清楚的概念，也讓我更有興趣去學更多相關的東西。

陳龍耀：在本次期末作業中，我們團隊選擇開發網頁記帳，本計畫深化了我對全端開發的理解。 Vue 3 和 Pinia 簡化了狀態管理，Tailwind CSS 加速了響應式設計。 Flask 和 SQLAlchemy 提供了高效的 API 開發體驗，PostgreSQL 確保資料可靠性。解決 JWT 令牌過渡和組邏輯挑戰，培養了問題分析能力。並且隊友間配合默契。

李柏駿：參與本次記帳網站應用方案的開發過程，是一段兼具挑戰與收穫的旅程。這個專案不僅成功打造了一個功能完善、用戶友好的財務管理平台，更在技術實踐與團隊協作中提供了深刻的學習經驗。

4.2結論

本記帳網站應用方案成功實現了一個功能完善、用戶友好且可擴展的財務管理平台，解決了傳統記帳在協作、跨訪問和即時更新方面的設備痛點。透過整合個人和群組記帳功能，採用現代技術堆疊（Vue 3、Flask、PostgreSQL）並克服了JWT會話管理、資料庫相容和群組邏輯等技術挑戰，本專案展示了全端開發的實務應用價值。

最終成果為使用者提供了一個直覺、安全的財務管理工具，支援個人財務追蹤和多人協作，透過動態儀表板和智慧篩選提升了財務分析力。對於開發者而言，專案提供了寶貴的學習案例，深入探討狀態管理、報表API設計、資料庫最佳化、雲端部署和錯誤處理流程的完整。

未來，應用程式可透過新增功能（如多幣種支援、報表產生、主動通知）、效能最佳化（如服務端快取和懶加載）以及測試覆蓋的擴充功能進一步提升其價值。此外，增強對螢幕閱讀器和鍵盤導航的支援可提高應用程式的可存取性，並惠及更廣泛的使用者群體。本計畫的經驗為後續網頁應用程式開發提供了艱難的技術和方法論基礎，特別是在安全認證、響應式設計和協作功能領域。

**4.3附註工作分配**

|  |  |  |
| --- | --- | --- |
| 成員 | 負責項目 | 比例 |
| 許博翔 | PPT | 100 |
| 陳龍耀 | WORD | 100 |
| 李柏駿 | 程式設計 | 100 |

備註：各成員在分工上有部分重疊，如系統整合與測試及報告由全體共同參與。

參考文獻

 Smith, J., & Lee, K. (2020)。個人理財應用的進步。 《金融科技雜誌》，12(3)，45-60。

 Jones, R. (2021)。基於雲端的會計系統：機會與挑戰。 IEEE 軟體工程學報，47(2)，123-135。

 Brown, T. (2022)。基於 Vue 和 Flask 的現代 Web 開發。 Web 開發評論，8(1)，89-102。

 Lee, S. (2023)。基於 JWT 的 Web 應用程式身份驗證面臨的挑戰。安全與隱私雜誌，15(4)，200-215。

 Chen, Y., & Wang, Z. (2024).全端開發中的資料庫相容性問題.資料庫系統期刊, 20(1), 33-48。

 Vue.js 官方文件。 搜尋自<https://vuejs.org/>

 Flask 官方文件。 搜尋自<https://flask.palletsprojects.com/>

 PostgreSQL 官方文件。 取自<https://www.postgresql.org/>

 Tailwind CSS 官方文件。 搜尋自<https://tailwindcss.com/>

 Chart.js 官方文件。 搜尋自<https://www.chartjs.org/>