順路經濟平台

姓名：陳睿廷

學號：110208059

指導老師：蔡子傑教授

Contents

Abstract

本研究旨在開發一個針對偏遠地區的順路經濟平台，製作響應式網頁架設平台，具有角色選擇、訂單填寫和媒合功能、訂單查詢、即時通知、傳送訊息以及路徑和預計時間等功能，以促進資源的有效配置和運輸效率的提升。

1 研究動機

2024 寒假參訪桃園雪霧鬧部落，是人口大約 200 人的小部落，當地目前以農業和民宿還有露營爲主，對外交通聯絡不易，地形陡峭，山路蜿蜒難以到達。因爲交通問題，沒有物流會願意上山一趟，所以發展出順路經濟。例如居民到山下採買順便把一些物資帶上山，讓其他居民可以不用下山一趟，但可能都是彼此認識居民互相幫忙，我希望能擴大規模，讓居民到山下採買，還可以幫助到更多居民，同時也可以賺錢，更加活絡山區物資運輸。其中有三種角色，買家、賣家、司機，核心功能是可以讓這三個角色提出需求，利用後台配對系統，滿足彼此的需求，詳細內容會在之後提到，想要做一個可以幫助偏鄉的資訊平台，希望可以用資訊的力量來幫助部落，未來想推廣到其他原住民部落使用。

2 研究目標

本研究的主要目標爲：

- 設計並實現一個針對不同角色（賣家、買家、司機）的角色選擇和管理功能。
- 開發一個訂單填寫和媒合系統，滿足不同角色的需求和供給。
- 實現通知功能，確保各方資訊可以即時傳遞。
- 設計路徑和預計時間功能，提高運輸效率。
- 設計登入註冊、訂單查詢、訊息、個人檔案功能。
- 驗證所設計系統的有效性和可行性，促進偏遠地區社區經濟的發展。
- 設計前端介面和結合上述功能，打造出一個完整的資訊平台。

3 功能列表

- 選擇角色和媒合功能
- 填寫訂單和媒合功能
- 通知功能
- 路徑和預計時間功能
- 訂單查詢功能
- 訊息功能
- 個人檔案功能
- 記帳功能

4 功能實現

順路經濟平台的設計包括前端和後端兩部分。前端主要負責使用者界面的設計和互動功能的實現，後端則負責資料處理和邏輯運算，以下是各個功能的實現細節：

4.1 選擇角色功能

- **前端：**設置角色選擇頁面，包含賣家、買家、司機三個角色選擇，使用 React 收集使用者的選擇。頁面使用動態表單來根據不同角色顯示相應的選項。
- **後端：**設置 API 來處理使用者角色的選擇和資料儲存，根據角色不同，將使用者導向相應的頁面或功能模組。使用 Express 框架設計 API 路由，處理使用者資料並儲存於 PostgreSQL。

4.2 填寫訂單和媒合功能

- **前端：**設置訂單填寫頁面，根據使用者角色顯示不同的表單，使用動態表單元件來收集訂單信息。使用 React 狀態管理來動態顯示和更新表單內容，並通過 AJAX 請求將資料發送到後端。

- **後端：**設置訂單 API，處理訂單的創建、修改和刪除，設置媒合邏輯，根據買家和賣家的需求進行媒合，並將結果推送給司機。使用 PostgreSQL 儲存訂單資料，設計相應的資料表和 CRUD 操作 API。

4.3 通知功能

- **前端：**使用 WebSocket 或 Push Notification 實現即時通知，設置通知中心，顯示使用者的所有通知。使用 React 建立通知中心元件，動態顯示和更新通知信息。
- **後端：**設置通知服務，處理通知的創建和發送，使用 Socket.IO 實現即時通知。後端服務器監聽訂單狀態變更事件，並即時推送通知給相關使用者。

4.4 路徑和預計時間功能

- **前端：**使用 Google Maps API 顯示路徑和預計時間，設置地圖和路徑顯示元件，提供路徑規劃和時間預估。使用 React 集成地圖元件，根據使用者選擇的起點和終點動態更新路徑和時間。
- **後端：**設置路徑計算 API，根據起點和終點計算路徑和預計時間，使用 Google Maps API 提供的路徑規劃服務。設計 API 路由，接收前端請求並返回計算結果。

4.5 支線功能

以下是支線功能的實現細節：

4.5.1 登入註冊功能

- **前端：**設計登入和註冊頁面，使用 React 處理表單提交，進行使用者輸入驗證，並通過 AJAX 將資料發送至後端 API。
- **後端：**設計 API 處理使用者資料儲存和驗證，使用 JWT 進行身份驗證。資料儲存於 PostgreSQL。

4.5.2 訂單查詢功能

- **前端：**設置訂單查詢頁面，顯示使用者的歷史訂單和當前訂單狀態，使用 React 和 AJAX 請求動態獲取和顯示訂單資料。

- **後端：**設計 API 提供訂單資料查詢和過濾功能，根據使用者請求返回相應的訂單資訊。資料儲存於 PostgreSQL，設計查詢和過濾邏輯。

4.5.3 訊息功能

- **前端：**設置訊息中心，顯示系統通知和使用者資訊，使用 WebSocket 實現實時資訊更新，使用 React 動態顯示和更新資訊內容。
- **後端：**設計 API 處理資訊的儲存和傳遞，使用 Socket.IO 實現即時資訊傳遞，後端服務器監聽資訊事件並推送給相關使用者。

4.5.4 個人檔案功能

- **前端：**設置個人檔案頁面，使用者可以編輯個人信息和查看歷史活動，使用 React 處理表單提交和資料綁定，通過 AJAX 將更新資料發送至後端。
- **後端：**設計 API 處理個人資料的讀取和更新，資料儲存於 PostgreSQL，設計資料模型和相應的 CRUD 操作 API。

4.5.5 記帳功能

- **前端：**設置記帳頁面，使用者可以查看和管理所有的交易記錄，使用 React 處理表單提交和資料綁定，通過 AJAX 將交易資料發送至後端，並顯示交易明細和總額。
- **後端：**設計 API 處理交易資料的創建、讀取、更新和刪除（CRUD），資料儲存於 PostgreSQL，設計交易資料模型，並提供相應的 CRUD 操作 API。系統會自動計算並更新使用者的總支出。

5 資料庫設計 (暫時)

資料庫設計會再根據需求狀況調整

資料表	欄位	描述
users（使用者）		
id	INT, PRIMARY KEY, AUTO_INCREMENT	使用者 ID (User ID)
username	VARCHAR(50), UNIQUE, NOT NULL	使用者名 (Username)
email	VARCHAR(100), UNIQUE, NOT NULL	使用者郵箱 (User email)
password_hash	VARCHAR(255), NOT NULL	密碼雜湊值 (Password hash)
role_id	INT, FOREIGN KEY to roles(id)	角色 ID (Role ID)
created_at	TIMESTAMP, DEFAULT CURRENT_TIMESTAMP	創建時間 (Created at timestamp)
updated_at	TIMESTAMP, DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP	更新時間 (Updated at timestamp)
roles（角色）		
id	INT, PRIMARY KEY, AUTO_INCREMENT	角色 ID (Role ID)

資料表	欄位	描述
role_name	VARCHAR(50), UNIQUE, NOT NULL	角色名稱 (Role name)
orders（訂單）		
id	INT, PRIMARY KEY, AUTO_INCREMENT	訂單 ID (Order ID)
user_id	INT, FOREIGN KEY to users(id)	使用者 ID (User ID)
total_price	DECIMAL(10, 2), NOT NULL	總價 (Total price)
status	VARCHAR(20), NOT NULL	訂單狀態 (Order status)
created_at	TIMESTAMP, DEFAULT CURRENT_TIMESTAMP	創建時間 (Created at timestamp)
updated_at	TIMESTAMP, DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP	更新時間 (Updated at timestamp)
order_items（訂單項目）		
id	INT, PRIMARY KEY, AUTO_INCREMENT	訂單項目 ID (Order item ID)
order_id	INT, FOREIGN KEY to orders(id)	訂單 ID (Order ID)
product_name	VARCHAR(255), NOT NULL	產品名稱 (Product name)
quantity	INT, NOT NULL	數量 (Quantity)

資料表	欄位	描述
unit_price	DECIMAL(10, 2), NOT NULL	單價 (Unit price)
total_price	DECIMAL(10, 2), NOT NULL	總價 (Total price)
notifications（通知）		
id	INT, PRIMARY KEY, AUTO_INCREMENT	通知 ID (Notification ID)
user_id	INT, FOREIGN KEY to users(id)	使用者 ID (User ID)
message	TEXT, NOT NULL	通知信息 (Notification message)
is_read	BOOLEAN, DEFAULT FALSE	閱讀狀態 (Read status)
created_at	TIMESTAMP, DEFAULT CURRENT_TIMESTAMP	創建時間 (Created at timestamp)
routes（路徑）		
id	INT, PRIMARY KEY, AUTO_INCREMENT	路徑 ID (Route ID)
order_id	INT, FOREIGN KEY to orders(id)	訂單 ID (Order ID)
start_location	VARCHAR(255), NOT NULL	起點位置 (Start location)
end_location	VARCHAR(255), NOT NULL	終點位置 (End location)
estimated_time	VARCHAR(50), NOT NULL	預計時間 (Estimated time)
distance	DECIMAL(10, 2), NOT NULL	距離 (Distance)

資料表	欄位	描述
created_at	TIMESTAMP, DEFAULT CURRENT_TIMESTAMP	創建時間 (Created at timestamp)
transactions（交易）		
id	INT, PRIMARY KEY, AUTO_INCREMENT	交易 ID (Transaction ID)
user_id	INT, FOREIGN KEY to users(id)	使用者 ID (User ID)
order_id	INT, FOREIGN KEY to orders(id)	訂單 ID (Order ID)
amount	DECIMAL(10, 2), NOT NULL	金額 (Transaction amount)
transaction_type	VARCHAR(20), NOT NULL	交易類型 (Transaction type)
transaction_date	TIMESTAMP, DEFAULT CURRENT_TIMESTAMP	交易日期 (Transaction date)
messages（訊息）		
id	INT, PRIMARY KEY, AUTO_INCREMENT	訊息 ID (Message ID)
sender_id	INT, FOREIGN KEY to users(id)	發送者 ID (Sender ID)
receiver_id	INT, FOREIGN KEY to users(id)	接收者 ID (Receiver ID)
message_content	TEXT, NOT NULL	訊息內容 (Message content)
created_at	TIMESTAMP, DEFAULT CURRENT_TIMESTAMP	創建時間 (Created at timestamp)

資料表	欄位	描述
user_profiles（使用者檔案）		
id	INT, PRIMARY KEY, AUTO_INCREMENT	檔案 ID (Profile ID)
user_id	INT, FOREIGN KEY to users(id)	使用者 ID (User ID)
name	VARCHAR(50), NOT NULL	姓名 (name)
phone_number	VARCHAR(20)	電話號碼 (Phone number)
address	VARCHAR(255)	地址 (Address)
profile_picture_url	VARCHAR(255)	個人資料圖片 URL (Profile picture URL)
created_at	TIMESTAMP, DEFAULT CURRENT_TIMESTAMP	創建時間 (Created at timestamp)
updated_at	TIMESTAMP, DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP	更新時間 (Updated at timestamp)

6 主要功能說明

- 首頁：放置三種角色（賣家、買家、司機），每個人都可以選擇這三個角色中的任意一個。
- 角色選擇：設置不同情況，方向和角色的排列組合。
  - 賣家（有批發大量的東西的需求）
  - 買家（想要物資）
  - 司機（負責採買然後運送）

- 假設山下集散地、山上集散地各有一個，並且假設具有能夠滿足需求者的冷藏的生鮮食品和生活必需品，並且交通方便的位置。不考慮山下山上集散地的進貨，還沒想金流的部分。

6.1 情景設計

- **情景 1：山上居民團購來自山下的東西**
  - 位置：山下-> 山上（山下賣家發出請求 → 累積到一定數量的買家 → 判斷需要多少司機）
  - 流程：山下賣家把貨物放到山下集散地 → 司機在山下集散地準備好貨物 → 運送到山上集散地
- **情景 2：山上水蜜桃果農想要賣到山下**
  - 位置：山上-> 山下（山上賣家發出請求 → 累積到一定數量的買家 → 判斷需要多少司機）
  - 流程：山上賣家把貨物放到山上集散地 → 司機在山上集散地準備好貨物 → 運送到山下集散地
- **情景 3：山上居民想要山下物資**
  - 位置：山下 → 山上（司機在山下集散地準備好貨物-> 運送到山上集散地）
- **情景 4：山下居民想要山上物資**
  - 位置：山上 → 山下（買家 + 司機）司機在山上集散地準備好貨物 → 運送到山下集散地

7 不同角色的條件設置

7.1 司機條件設置

- 方向：山下往山上還是山上往山下
- 方便運送的時間
- 車子種類：轎車、休旅車、3.5 噸貨車

7.2 買家條件設置

- 方向：山下往山上還是山上往山下
- 時間限制：最晚司機要什麼時候接單，否則棄單
- 是否緊急：接單後需要當天到達（例如生鮮類商品）
- 取貨地點（目前就只有一個地方）
- 想要購買的商品資訊

7.3 賣家條件設置

- 方向：山下往山上還是山上往山下
- 到什麼時候累積到多少量才出貨
- 貨物詳細資訊

8 預計行程表

日期	任務	備註
6/25-7/24	預計製作 prototype	實現平台核心和基本功能
7/4-7/7	去雪霧鬧實際調查	了解部落實際狀況
9 月	優化和測試核心功能	確保核心功能正常
10 月	優化和測試支線功能	確保支線功能正常
11 月	整合測試並研究平台部署	確保平台能夠正常部屬運行
12 月	最後測試，製作海報	準備專題成果展

Table 2: 預計行程表

9 遇到的問題與解決方案

在需求分析過程中，我們遇到了一些挑戰並提出了解決方案：

- 通知的問題：網站如何推播，手機上的震動或聲音通知。解決方案是使用 WebSocket 或 Push Notification 實現即時通知。
- 平台黏著性的問題：必須有中間的服務員轉接，讓不太會用手機的小孩或長輩也能夠使用服務。

- 設計出來的網頁實用性：了解部落的實際狀況和需求來進行調整。
- 司機接單後放棄的問題：設計系統來確保單子可以接到，並且有機制處理司機中途棄單的情況。

10 程式開發流程

程式開發流程可以參考 GitHub，可以根據 commit 紀錄來看每次的修改。我有加上詳細的 commit 紀錄，可以知道每一次的變更。
6/26 完成基本環境設定，詳見 GitHub Repository。

11 結論

本研究通過設計和開發順路經濟平台，實現了針對不同角色的功能，有效地提高了偏遠地區資源的配置和運輸效率，但還需要通過使用者測試和反饋，提升系統的可行性和實用性。未來可以進一步優化系統功能，擴展到更多偏遠地區，推廣順路經濟。