此Demo是使用 Flask + bootstrap + PostgreSQL 的註冊登入系統

環境設置

1. 安裝相關套件： 此Demo是使用python 3.11.9版本，進入此Demo目錄，執行
   ```
   cd flask_demo
   python -m venv env
   source env/bin/activate (Linux/Mac) or .\env\Scripts\activate (Windows)
   pip install -r requirements.txt
   ```
2. 創建資料庫： 以PostgresSQL創建名為`mydb`的資料庫，並在`mydb`中建立名為`users`的資料表，
   ```
   CREATE TABLE users (
   id serial PRIMARY KEY,
   username VARCHAR ( 50 ) NOT NULL,
   password VARCHAR ( 255 ) NOT NULL,
   email VARCHAR ( 50 ) NOT NULL,
   registered_date DATE
   );
   ```
3. 啟動：
   ```
   cd backend
   flask run
   ```
   首頁會在本地http://127.0.0.1:5000/ 中呈現

API說明

1. 創建帳號，POST，'/api/users'
   request json:
   ```
   {
   "email": "xxx@xxx.xxx",
   "password": "xxx",
   "username":"xxxx"
   }
   ```
   response:
   檢查是否重複及確認資料皆合法後，回傳創建成功訊息
2. 用API登入，POST，'/api/login/'
   request json:
   ```
   {
   "email": "xxx@xxx.xxx",
   "password": "xxx"
   }
   ```
   response:
   檢查資料是否正確，再回傳登入成功與否的結果
3. 查詢所有帳號列表，GET，'/api/users'
   要是登入狀態才能查詢，已登入的話會得到所有用戶的
   `id`,`username`,`email`,`registered_date`資料

4. 查詢單一帳號，GET，'/api/users/{user_id}'
   要是登入狀態才能查詢，user_id 的type是integer，已登入的話會得到`user_id`該`id`的資料
5. 更新特定帳號的資料，PUT，'/api/users/{user_id}'
   要是登入狀態才能更新，user_id 的type是integer
   request json:
   ```
   {
   "new_username": "xxx",
   "new_password": "xxxx",
   "new_email": "xxxx@xx.xx"
   }
   ```
   根據輸入的request json檔可以更新該`id`用戶的名字、密碼或電子郵件，有輸入的key及value才會去更新
   response:
   檢查資料是否正確，再回傳更新成功與否的結果
6. 刪除特定帳號的資料，DELETE，'/api/users/{user_id}'
   要是登入狀態才能刪除，user_id 的type是integer，
   已登入的話可以刪除特定`user_id`的資料
7. 登入時會印出jwt的token，時效目前設定是3600秒