**SA（系統架構說明）**

**系統目的**  
自動化處理多家銀行對帳單 → 與會計客戶資料庫模糊比對 → 依入帳日輸出 SAP 可上傳之會計憑證 .xls，跨批次避免重複並保留批次歷史（YYYYMMDD-2.xls 等）。

**關鍵元件**

* run\_gui.py：GUI 入口，收檔案＋過帳日，逐檔呼叫核心流程。
* bank.py：管線編排（偵測銀行 → 解析 → 比對 → 去重 → 輸出）。
* parsers.py：各銀行 Parser（如 Citi、CTBC、Fubon…），負責取「客戶名稱、金額、細節」。
* fuzzy\_matcher.py：名稱與客戶庫的模糊比對（必要時人工確認／輸入 ID）。
* utils.py：共用工具（記錄跳過、版本命名、I/O）。
* **外部資源**：會計憑證模板（空白）、客戶資料庫、輸出資料夾 ~/Downloads/Banks。

**資料流程（文字拓撲）**  
GUI → 檔名判斷銀行 → Parser 擷取列 → 模糊比對客戶庫 → 蒐集同日已存在紀錄（含 -2/-3 檔）→ 以 (E posting\_date, U cust\_id, S amount) 去重 → 以模板產生新 .xlsx → 轉 .xls → 刪 .xlsx。

**執行與部署**

* 需求：Python 3.10+、套件 rapidfuzz pandas openpyxl xlrd pywin32。
* 執行：python run\_gui.py（Windows 可雙擊 Run GUI）。
* 產出：~/Downloads/Banks/會計憑證導入模板 - YYYYMMDD(-N).xls。

**安全與稽核**

* 僅處理本機檔案；敏感資料不上傳。
* skipped.csv 紀錄略過原因（供追蹤）。
* 版本檔保留批次痕跡，便於稽核。

**SD（詳細設計／規格說明）**

**輸入**

* 多個銀行對帳單（.xls/.xlsx），由使用者在 GUI 選擇。
* 過帳日（YYYYMMDD；預設今日）。

**核心演算法**

* 銀行偵測：­依檔名關鍵字對應 Parser。
* 解析：Parser 介面輸出結構 {name, amount, details} 列表。
* 模糊比對：以名稱對「客戶資料庫」比對，門檻值可調；必要時人工指定 ID。
* 去重規則：主鍵 = (E posting\_date, U cust\_id, S amount)；聚合同日所有既有檔（含 -2/-3）建立「已寫入鍵集合」，新批次只寫**未出現**的鍵。
* 檔名版本：首次 YYYYMMDD.xls，同日第 n 次 YYYYMMDD-n.xls（只含本批新資料）。

**輸出**

* 先寫 .xlsx（便於 openpyxl 處理），完成後轉 .xls（SAP 相容），刪除中間 .xlsx。
* 模板套版：固定欄位落位（含 E=posting date、U=cust id、S=amount、I=text…），維持雙列（DZ）結構。

**錯誤處理**

* 解析不到關鍵欄位 → 記錄並略過檔。
* 模糊比對分數過低 → 進入人工確認清單或記錄跳過。
* 輸出失敗（檔案鎖定）→ 提示關閉檔案後重試。

**可擴充性（新增銀行）**

1. 在 parsers.py 實作新類別（定義：目標工作表、欄位、終止條件）。
2. 於 PARSER\_REGISTRY 註冊銀行關鍵字 → 類別。
3. 撰寫一組最小測試（見下方「測試」）。

**設定點**

* 模糊比對門檻（建議 80–90）。
* 輸出資料夾（預設 ~/Downloads/Banks）。
* 模板路徑（TEMPLATE\_FILE）。

## Data Model（資料模型）

會計憑證輸出欄位（範例與說明）

* **E（Posting Date / 過帳日期）**
  + 格式：文字型態 YYYYMMDD（例如 20250715）。
  + 來源：GUI 中輸入的過帳日期。
  + 作用：決定該批憑證的入帳日，為主鍵組合之一。
* **U（Customer ID / 客戶代號）**
  + 格式：文字或數字型態（依客戶資料庫）。
  + 來源：經由 **模糊比對** 從客戶資料庫取得的唯一代號。
  + 作用：作為會計分錄的對應客戶，主鍵組合之一。
* **S（Amount / 金額）**
  + 格式：數字（允許正數／負數）。
  + 來源：銀行對帳單交易金額。
  + 作用：決定憑證金額，主鍵組合之一。
* **I（Text / Details / 摘要或細節說明）**
  + 格式：文字。
  + 來源：銀行對帳單的「細節描述」或「備註」。
  + 作用：輔助辨識交易，提供上傳 SAP 後的查詢依據。

**備註**：

* 系統會輸出 **雙列（兩列一組）** 的憑證，每一組代表一筆交易（例如借方／貸方）。
* 其他欄位（如科目代號、幣別、公司代碼等）依模板的固定值填入，或沿用模板既有公式／格式，不由本程式動態產生。

### 客戶資料庫（Customer Master）

* **Customer ID（主鍵）**
  + 唯一識別每個客戶，為會計系統標準代碼。
* **Customer Name（比對主欄位）**
  + 與銀行對帳單中的「客戶名稱／細節描述」進行模糊比對。
  + 支援多種名稱變形（如簡稱、英文名、別名）。
* **其他欄位**
  + 包含與會計憑證相關的資訊（如付款條件、公司代碼、科目等）。
  + 在輸出憑證時直接沿用，不由程式修改。

**說明**：客戶資料庫為核心比對依據，程式會使用 **rapidfuzz** 進行模糊比對；若分數低於閾值，則可能跳過或要求人工確認。

**維護（運維要點）**

* 版本命名維持 YYYYMMDD(-N).xls，勿在外部手動更名。
* 定期清查 skipped.csv 與 GUI Log，針對常見未匹配客戶在客戶庫補充別名。
* 新增銀行時先在測試資料夾建立 2–3 份真實樣式檔做回歸。
* 若模板欄位或 SAP 上傳規範變更，優先更新 TEMPLATE\_FILE 與欄位映射常數。
* 目前程式以 bank\_reconciliation 專案資料夾為核心，建議維持在單一目錄中集中管理。  
  未來若需擴充功能（例如新增銀行 parser、整合其他會計模組），可以在 parsers.py 中新增類別並於 PARSER\_REGISTRY 註冊，或建立新的子資料夾專門存放擴充模組，保持結構清晰。