8089

房價預測器

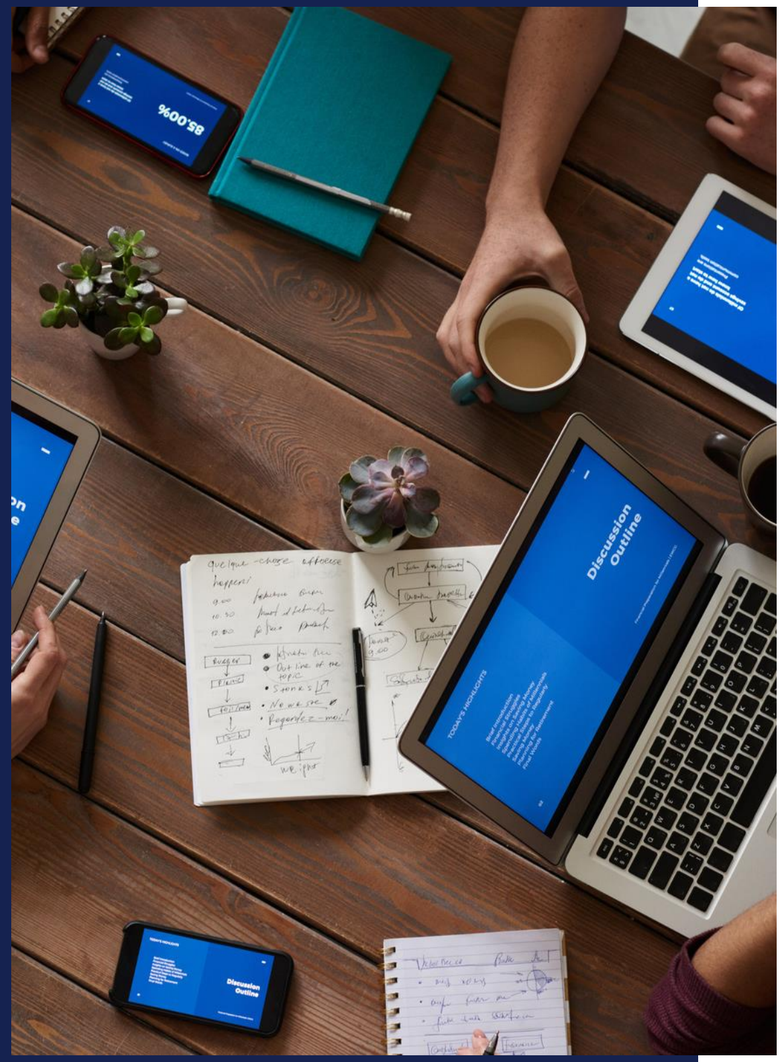
課程：Python程式設計與實務應用

班級：ZZZ002

組長：詹秉蒼

組員：蔡婷羽、蔡惠婷、陳銘泓
余誼姍、 張心齡

目錄



小組分工概況

工作項目	人員
題目發想及撰寫提案	蔡婷羽
開發文件	詹秉蒼、陳銘泓、蔡婷羽、蔡惠婷
使用手冊、海報製作	余誼姍
程式碼撰寫	GUI介面：陳銘泓 資料整理：余誼姍 資料表設計：詹秉蒼 資料撈取：蔡惠婷 預測模型的調用或公式模型的開發：蔡婷羽 整體程式碼整合串接：詹秉蒼 編譯windows執行檔：陳銘泓 編譯macOS執行檔：蔡婷羽
投影片報告製作及面授報告	張心齡

軟體介紹

專案簡介

軟體操作說明

相關套件

專案簡介

專案主題：房價預測器

簡介：

本軟體透過爬取內政部實價登錄資料，以最真實的成交數據為基礎，運用統計學方法，科學的預測房價行情。

軟體操作說明

1. 選取交易地點以及交易標的(門牌/社區可選填)
2. 選取交易期間
3. 選取欲篩選之單價與總面積範圍
4. 選取屋齡範圍
5. 選取預期購買年月
6. 選取預期購買之面積條件
7. 點擊生成資料計算結果
8. 結果輸出區
9. 切換GUI顏色(深/淺色)

相關套件及資源

- Tkinter&Customtkinter：GUI介面
- MySQL：資料庫
- Requests：下載公開交易資料
- Pandas：結構化數據
- Numpy：模型預測公式

軟體開發流程

流程圖

資料下載

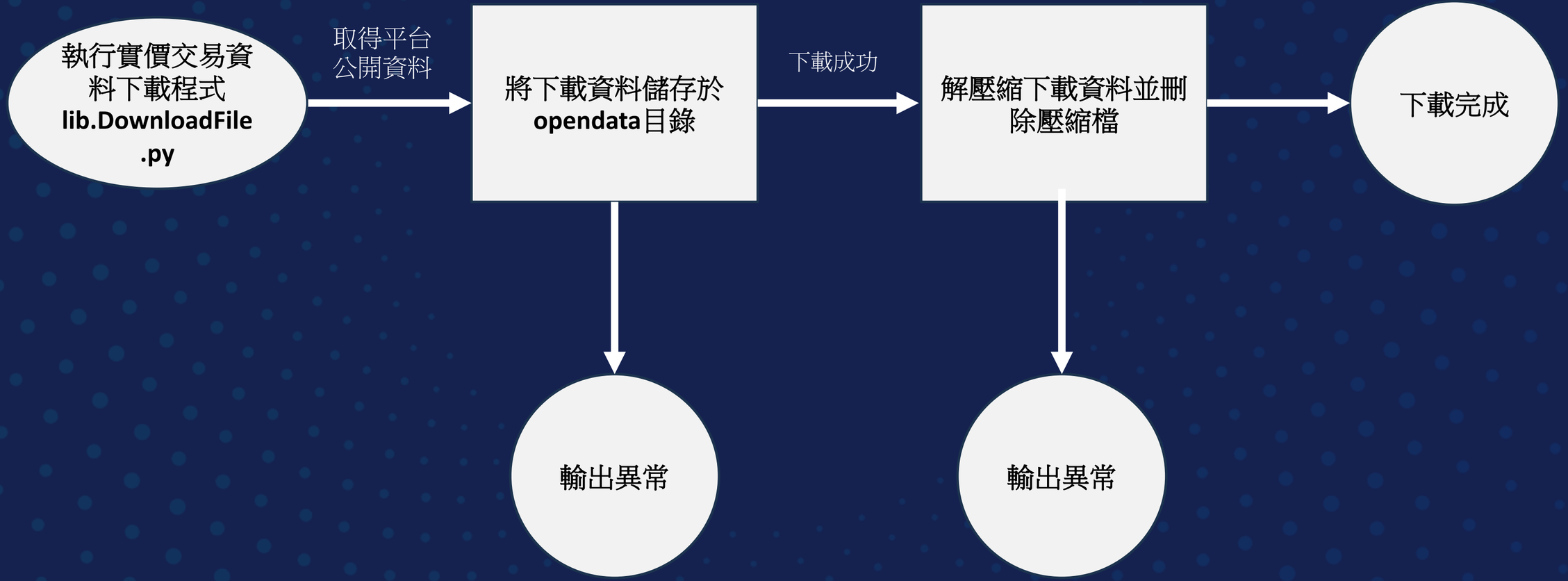
資料彙整與儲存

應用程式

流程圖


GUI介面
提供用戶可輸入的圖形介面
傳遞用戶輸入資料
預測結果回傳
預測模型
根據輸入數據計算出預期房價
提供預測模型數據
資料下載
下載內政部實價登陸的公開數據
下載資料
資料儲存
將資料彙整並寫入資料庫

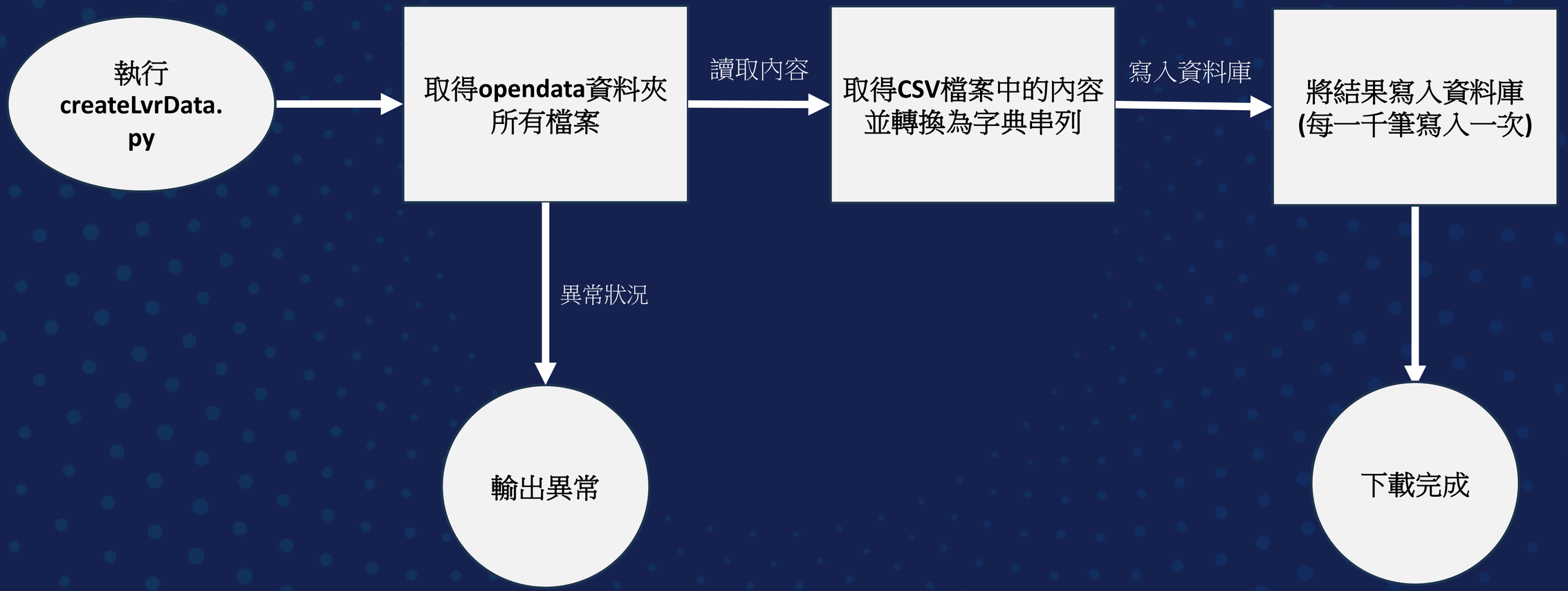
流程圖-資料下載


執行實價交易資料下載程式 lib.DownloadFile.py
取得平台公開資料
將下載資料儲存於opendata目錄
下載成功
解壓縮下載資料並刪除壓縮檔
下載完成
輸出異常
輸出異常

流程圖-資料下載

```
# 壓縮檔名稱
zipPath = f"{outputDir}/data{i}.zip"

# 下載檔案，將資料流寫到本地檔案中
result = downloadFile(zipPath, f"{URL}/{item['path']}", item['params'])

if result == True:
    print(f"{i}-下載成功")
    # 使用檔名作為資料夾將資料解壓縮
    unzipResult = UnZip(zipPath, zipPath.replace(".zip", ""))
    if unzipResult :
        print(f"{i}-解壓縮完成")
        if os.path.exists(zipPath):
            os.remove(zipPath)
            print(f"{i}-已刪除下載的ZIP檔")
    else:
        print(f"{i}-解壓縮失敗")
else:
    print(f"{i}-下載失敗")
i += 1
```

流程圖-資料彙整與儲存


執行
createLvrData.
py
取得opendata資料夾
所有檔案
讀取內容
取得CSV檔案中的內容
並轉換為字典串列
寫入資料庫
將結果寫入資料庫
(每一千筆寫入一次)
異常狀況
輸出異常
下載完成

流程圖-資料彙整與儲存

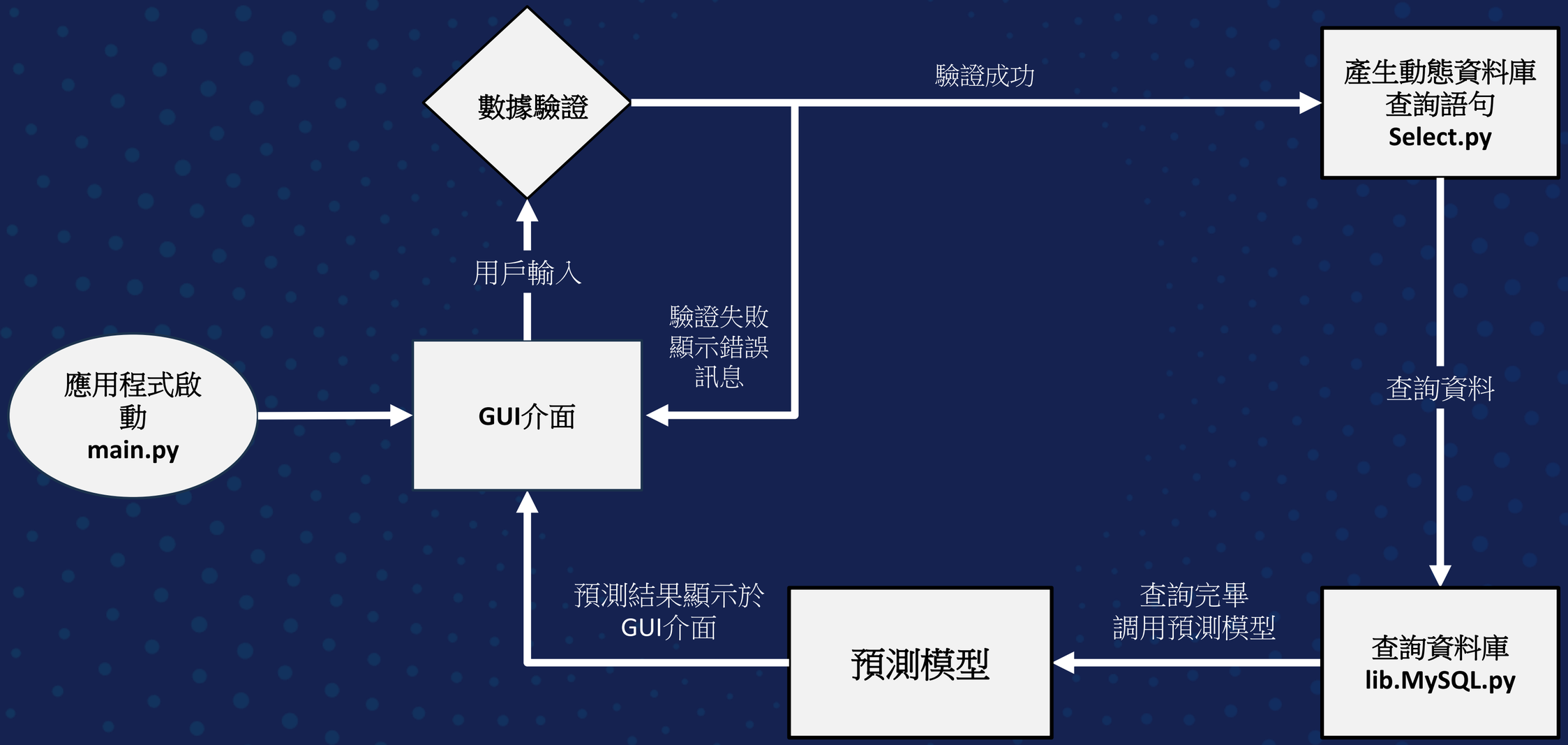
```python
def insertSQL(self, row = 1000):
    '''
    將數據寫入資料庫每1000比進行一次批次寫入
    '''
    print("資料庫已完成上傳，請勿重複執行")

    result = self.getData(1)
    total = len(result)
    print(f"共有{total // row + 1}批資料要寫入每批資料有{row}筆")
    with MySQL.MySQL() as db:
        sql = []
        sql.append("INSERT INTO omeiliau_nou.lvr_lnd (city_code, city_name, town_code, town_name,
                   trade_sign, address, trade_date, price_total, price_nuit, total_area, code, age)")
        sql.append(f"VALUES ({(", ".join(["%s"] * 12))})")
        data_to_insert = []
        for idx,item in enumerate(result):
            #將字典轉元組
            data_to_insert.append((item["city_code"],item["city_name"],item["town_code"],
                                   item["town_name"],item["trade_sign"],
                                   item["address"],item["trade_date"],item["price_total"],
                                   item["price_nuit"],item["total_area"],item["code"],item["age"]))
            # db.insert_many(" ".join(sql), data_to_insert)
            if len(data_to_insert) >= row or idx == total - 1:
                db.insert_many(" ".join(sql), data_to_insert)
                data_to_insert = []  # 清空資料，為下一批次做準備
                print(f"已批次寫入資料：第 {idx // row + 1} 批")
```

流程圖-資料彙整與儲存

資料表與結構

csv檔案中的欄位名稱	資料庫欄位名稱	資料型別	預設值	備註
	city_code	string		縣市代號 （參照params.py的city字典Key）
	city_name	string		縣市名稱 （參照params.py的city字典Key）
	town_code	string		鄉鎮市區參照params.py的town字典key）
鄉鎮市區	town_name	string		鄉鎮市區參照params.py的town字典value
交易標地	trade_sign	int		（房地 => 1 建物 => 2 土地=> 3 車位=>4, 房地+車位 => 5）
土地位置建物門牌	address	strnig		
交易年月日	trade_date	int		
成交總價(元)	price_total	int		
單價元平方公尺	price_nuit	int		
建物移轉總面積平方公尺	total_area	float		
編號	code	string		
屋齡	age	int	0	此處比較特特別， 假測現在開啟的檔案是a_lvr_land_a.csv要取這份檔案中的「編號」欄位的值，然後到a_lvr_land_a_build.csv這份檔案對應的「編號」取得屋齡，如果是土地就沒有屋齡問題預設值為0

流程圖-應用程式


數據驗證
驗證成功
產生動態資料庫
查詢語句
Select.py
用戶輸入
驗證失敗
顯示錯誤
訊息
應用程式啟
動
main.py
GUI介面
查詢資料
預測結果顯示於
GUI介面
預測模型
查詢完畢
調用預測模型
查詢資料庫
lib.MySQL.py

流程圖-應用程式：GUI

```
# I/O 、 處理函式 及 變數部分 ==========================================================================================
# [I/O]輸入資料表
user_input_list = {
    "city":"",                  # str :     縣市,                                   必填：是 (預設:第一筆的key)
    "town":"",                  # str :     鄉鎮市區,                               必填：是 (預設:第一筆的key)
    "ptype":[],                 # list :    1房地、2建物、3土地、4車位、5房地+車位,  必填：是 (預設:[1])
    "p_build": None,            # str :     門牌地址 :                              必填：否 (預設:)
    "p_startY": None,           # int :     交易期間(年起)101-113,                  必填：是 (預設:101)
    "p_startM": None,           # int :     交易期間(月起)1-12,                     必填：是 (預設:1)
    "p_endY": None,             # int :     交易期間(年迄)101-113,                  必填：是 (預設:113)
    "p_endM": None,             # int :     交易期間(月迄)1-12,                     必填：是 (預設:12)
    "pmoney_unit": None,        # int :     單位(1 => 萬元 , 2 => 元)               必填：是 (預設:1)
    "minp": None,               # int :     最小值(單價),                           必填：否 (預設:)
    "maxp": None,               # int :     最大值(單價),                           必填：否 (預設:)
    "unit": None,               # int :     面積單位(1 => M^2 ,2 => 坪),            必填：是 (預設:2)
    "mins": None,               # int :     最小值(坪數),                           必填：否 (預設:)
    "maxs": None,               # int :     最大值(坪數),                           必填：否 (預設:)
    "avg_var": None,            # int :     屋齡,                                   必填：否 (預設:)

    # 計算部分(我們自訂的)
    "calculate_Y": None,        # int :     目標期間(年),                           必填：是 (預設:)
    "calculate_M": None,        # int :     目標期間(月),                           必填：是 (預設:)
    "calculate_unit": None,     # int :     面積單位(1 => M^2 ,2 => 坪),            必填：是 (預設:2)
    "calculate_area": None      # int :     面積,                                   必填：是 (預設:)
}
```

流程圖-應用程式：預測公式

```
# 應傳入之參數origin_data（字典）範例：
# {
#     11110: 116730, # 111年10月之平均成交價為每平方公尺116730元
#     11205: 121716,
#     11307: 124736
# }
# key為「交易年月」，value為「各該月平均單價（元/平方公尺）」（資料按交易時間由舊到新排序）
```

流程圖-應用程式：預測公式

```
# 產生預測公式（最小平方迴歸法擬合）
data = pd.DataFrame({'X': trade_time, 'Y': house_price_per_square_meter_month_average})
# pd.DataFrame(字典資料)
# 範例：trade_time = [0, 7, 21]
# 範例：house_price_per_square_meter_month_average = [116730, 121716, 124736]

cofficients = np.polyfit(data['X'], data['Y'], 1)
# 使用numpy的polyfit函式 進行 最小平方法擬合
# 1 表示擬合 「一次」多項式模型（直線）
# np.polyfit(字典[對應自變數的key], 字典[對應應變數的key], 想要擬合幾次多項式)
# cofficients = [斜率, 截距]

slope = cofficients[0] # 斜率
intercept = cofficients[1] # 截距
```

流程圖-應用程式：預測公式

1. 斜率 m：

$$m = \frac{n\sum(x_i y_i) - \sum x_i \sum y_i}{n\sum x_i^2 - (\sum x_i)^2}$$

2. 截距 b：

$$b = \frac{\sum y_i - m\sum x_i}{n}$$

- x_i 和 y_i 是數據點的座標。
- n 是數據點的數量。

實測結果

實測結果

鄉鎮市區：台北市大安區

交易標的：房地

交易期間：101年1月 - 113年1月

單價：無

面積：無

屋齡：不拘

預期購買時間：120年1月

預期購買面積：無

實測結果

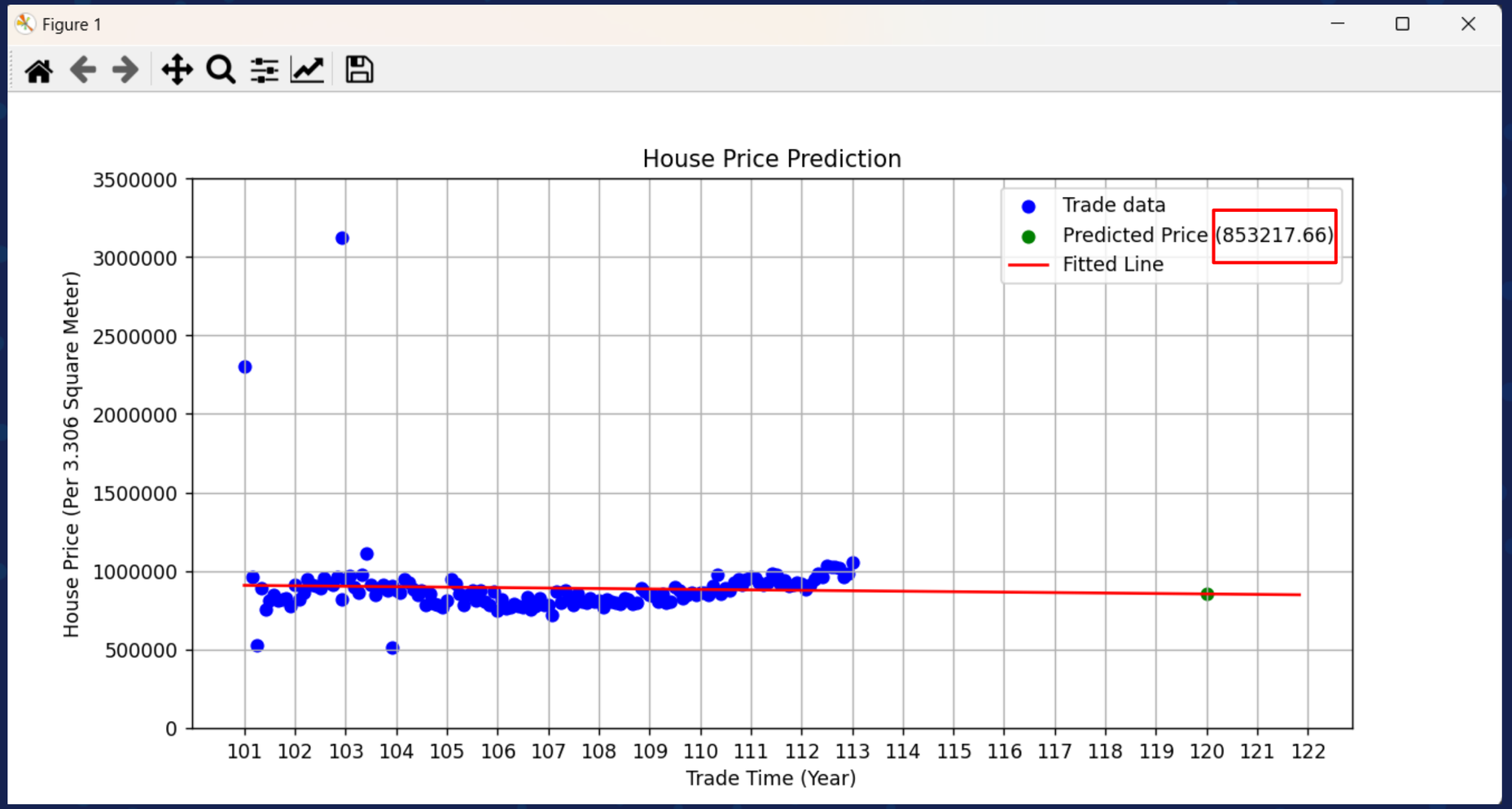


Figure 1
House Price Prediction
Trade data
Predicted Price (853217.66)
Fitted Line
House Price (Per 3.306 Square Meter)
Trade Time (Year)
3500000
3000000
2500000
2000000
1500000
1000000
500000
0
101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122

報告完畢
感謝您的聆聽