結合UCAN職能評量與生成式AI之課程選修與職涯路徑推薦系統
--以T大學資管系為例

專題生:呂柏葦、林芯瑜、郭致君
指導教授:謝明哲 博士

目錄

一、前言
二、文獻探討
三、解決方案設計
四、解決方案實現與發展
五、案例展示
六、測試結果評估
七、平台與UCAN連結及模擬課程推薦
八、討論與結論

一、前言

研究背景

學生在課程選擇中常面臨迷惘與動機不足的困境。

現況問題

- 學生選課盲目、學習動機低落
- 課程與未來職涯缺乏清晰連結
- 缺乏有效的課程規劃工具與支持

研究動機：

- 結合生成式AI與UCAN職涯平台
- 提供清晰、動態且職涯導向的課程推薦方案，提升學生學習動機與自主學習能力。

二、文獻探討

1.AI工具提升自主學習與動機 (Gervacio, 2024)

- 生成式AI（如:ChatGPT）使用提升學習參與與動機，這兩者再正向預測學生的研究能力發展與自主規劃能力。

2.利用生成式 AI 建立適應性學習路徑(Dhagare, 2024)

- AI 能根據學生數據生成「彈性學習模組」，改善固定式教學設計造成的學習落差，促進學習成就與動機。

3.AI 生成個性化教材對學生參與影響(Pesovski et al, 2024)

- 與興趣相連結的教材可延長注意力維持時間與學習投入，AI 客製教材可創造「沉浸式學習經驗」。

二、文獻探討

4.個性化課程順序推薦(2016)
(Personalized Course Sequence Recommendations)

- 一種向學生提供個人化課程序列推薦的系統方法。可以最佳的選擇課程序列以減少學生畢業所需的時間

5.通過使用具有遺傳優化的混合多標準推薦系統來幫助大學生選擇選修課程(2024)

- 幫助學生選擇適合他們個人興趣和學習成績的課程的工具，使用與學生和課程資訊相關的多個標準向學生推薦最合適的課程。

參考文獻APA

AI工具提升自主學習與動機 (Gervacio, 2024)🔗 論文連結（ResearchGate）

作者 :Yi Lii/Ghulfam Sadiq/Ghulam Qambar/Pengyu Zhen
發表時間： 2024/8

利用生成式 AI 建立適應性學習路徑(Dhagare, 2024) 🔗 論文連結（IJRASET）

作者： Rahul Prabhakar Dhagare 先生
發表時間: 2024/11

AI 生成個性化教材對學生參與影響(Pesovski et al, 2024) 🔗 論文連結（ResearchGate）

作者 : Ivica Pesovski/Ricardo Santos/Roberto Henriques/Vladimir Trajkovik
發表時間 : 2024/8

參考文獻APA

個性化課程順序推薦　　[1512.09176] 個性化課程順序推薦

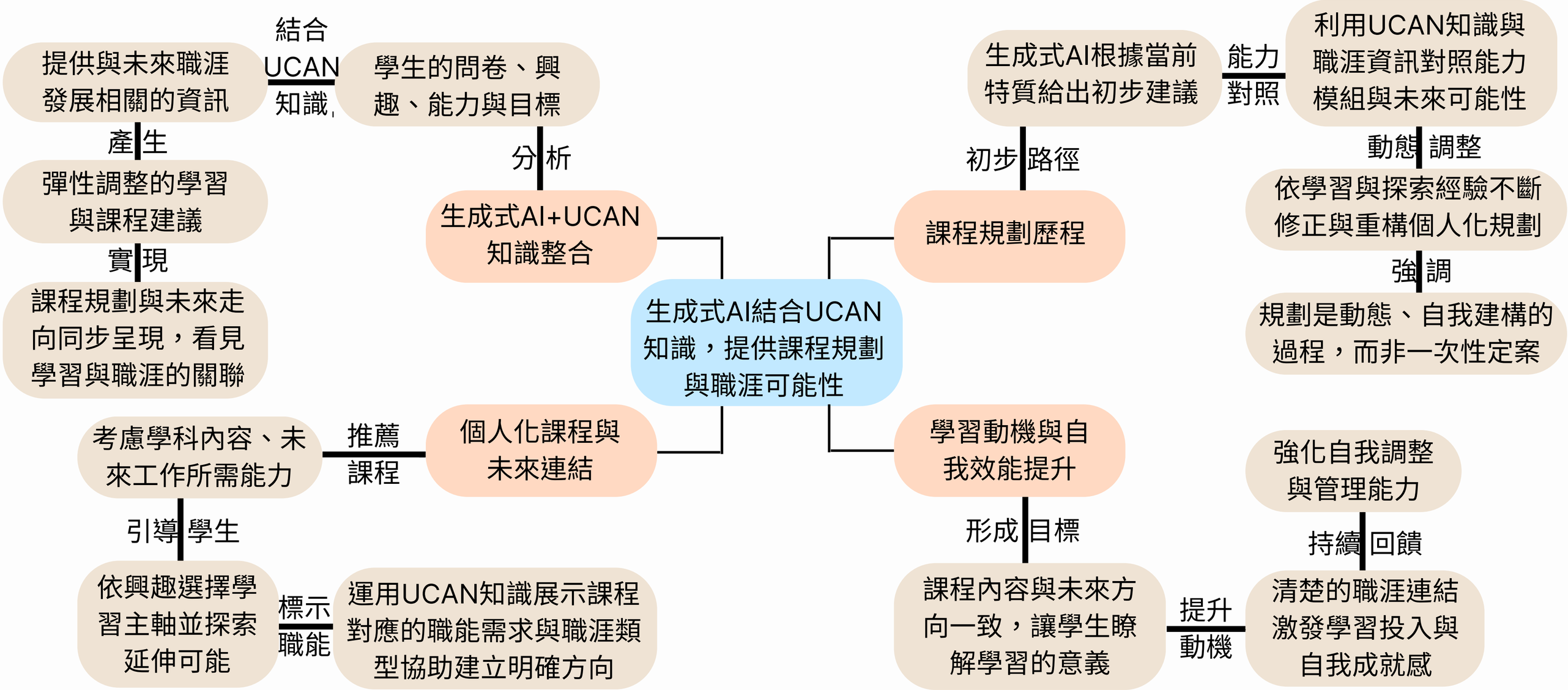
作者 : Jie Xu， Tianwei Xing， Mihaela van der Schaar
發表時間 : 2016/1

通過使用具有遺傳優化的混合多標準推薦系統來幫助大學生選擇選修課程

作者 : A. Esteban, A. Zafra, C. Romeror
發表時間 : 2024/2

[2402.08371] 通過使用具有遺傳優化的混合多標準推薦系統來幫助大學生選擇選修課程

三、解決方案設計


生成式AI結合UCAN知識，提供課程規劃與職涯可能性
生成式AI+UCAN知識整合
結合UCAN知識
學生的問卷、興趣、能力與目標
分析
提供與未來職涯發展相關的資訊
產生
彈性調整的學習與課程建議
實現
課程規劃與未來走向同步呈現，看見學習與職涯的關聯
課程規劃歷程
初步路徑
生成式AI根據當前特質給出初步建議
能力對照
利用UCAN知識與職涯資訊對照能力模組與未來可能性
動態調整
依學習與探索經驗不斷修正與重構個人化規劃
強調
規劃是動態、自我建構的過程，而非一次性定案
個人化課程與未來連結
推薦課程
考慮學科內容、未來工作所需能力
引導學生
依興趣選擇學習主軸並探索延伸可能
標示職能
運用UCAN知識展示課程對應的職能需求與職涯類型協助建立明確方向
學習動機與自我效能提升
形成目標
課程內容與未來方向一致，讓學生瞭解學習的意義
提升動機
強化自我調整與管理能力
持續回饋
清楚的職涯連結激發學習投入與自我成就感

三之一、系統組成解析：AI與UCAN功能角色

1. 生成式AI

→ **分析** → 學生的興趣、能力、學習風格與職涯目標

→ **應用** → 學習歷程檔案、問卷數據與課程回饋資料

→ **進行** → 課程與職涯資料的比對與推論

→ **產生** → 個人化的學習建議與課程推薦

2. UCAN平台整合

→ **提供** → 能力指標與職涯探索工具

→ **支援** → AI建議的驗證與補充

→ **建立** → 模組化、多元化的學習路徑

三之二、應用策略設計：學習支持與動機提升

3. 個人化課程推薦

→ **對應** → 不同學生類型（迷茫型／專注型）

→ **引導** → 迷茫型學生探索學習方向

→ **深化** → 專注型學生的專業能力與實務經驗

4.課程與職涯連結

→ **呈現** → 課程與未來職涯的對應關係

→ **說明** → 學習內容的實務應用情境

→ **強化** → 課程選擇的意義與學習動機

5. 學生學習動機提升

→ **來自於** → 興趣與職涯目標的匹配

→ **建立** → 明確的學習方向與自信心

→ **促進** → 主動學習與持續探索

三之三、學習特徵分類：二分法問卷設計與分析

我目前已經確定了自己的未來目標或方向。 *

- 是
- 否

入學至今，我的學習動力相比剛入學時有所下降。 *

- 是
- 否

我目前所學的專業內容與我的興趣密切相關。 *

- 是

我曾經考慮過轉系或轉換學習方向。 *

- 是
- 否

我對未來畢業後要做什麼樣的工作感到迷茫。 *

- 是
- 否

我在學習中經常感到不知道自己該專注於哪個領域。 *

- 是

三之三、學習特徵分類：二分法問卷設計與分析

我認為資訊管理領域的學習與我的興趣不符。 *

- 是
- 否

我有清晰的學習計劃，並能按照計劃執行 *

- 是
- 否

我曾經參加過課外活動（如:社團類型活動或運動等、競賽或專題）來探索自己的興趣。 *

- 是
- 否

我會制定學習目標並將其分解為短期目標然後進行反思。 *

- 是
- 否

我認為與同學討論學習內容對提升我的學習興趣有幫助。 *

- 是
- 否

我經常根據自己的興趣或未來需求自主選擇學習內容。 *

- 是
- 否

三之三、學習特徵分類：二分法問卷設計與分析

問卷目的:

透過問卷分析，**判斷**學生在課程選擇與學習行為中的**主動性**與**規劃能力**，進而分類為：

- 專注型學生
- 迷茫型學生

透過這樣的分類，我們能**明確辨識**學生在目標設定、學習動機、自我管理、困難處理與學習參與度方面的差異，進一步提供**個別化的學習建議**。

分析面向：

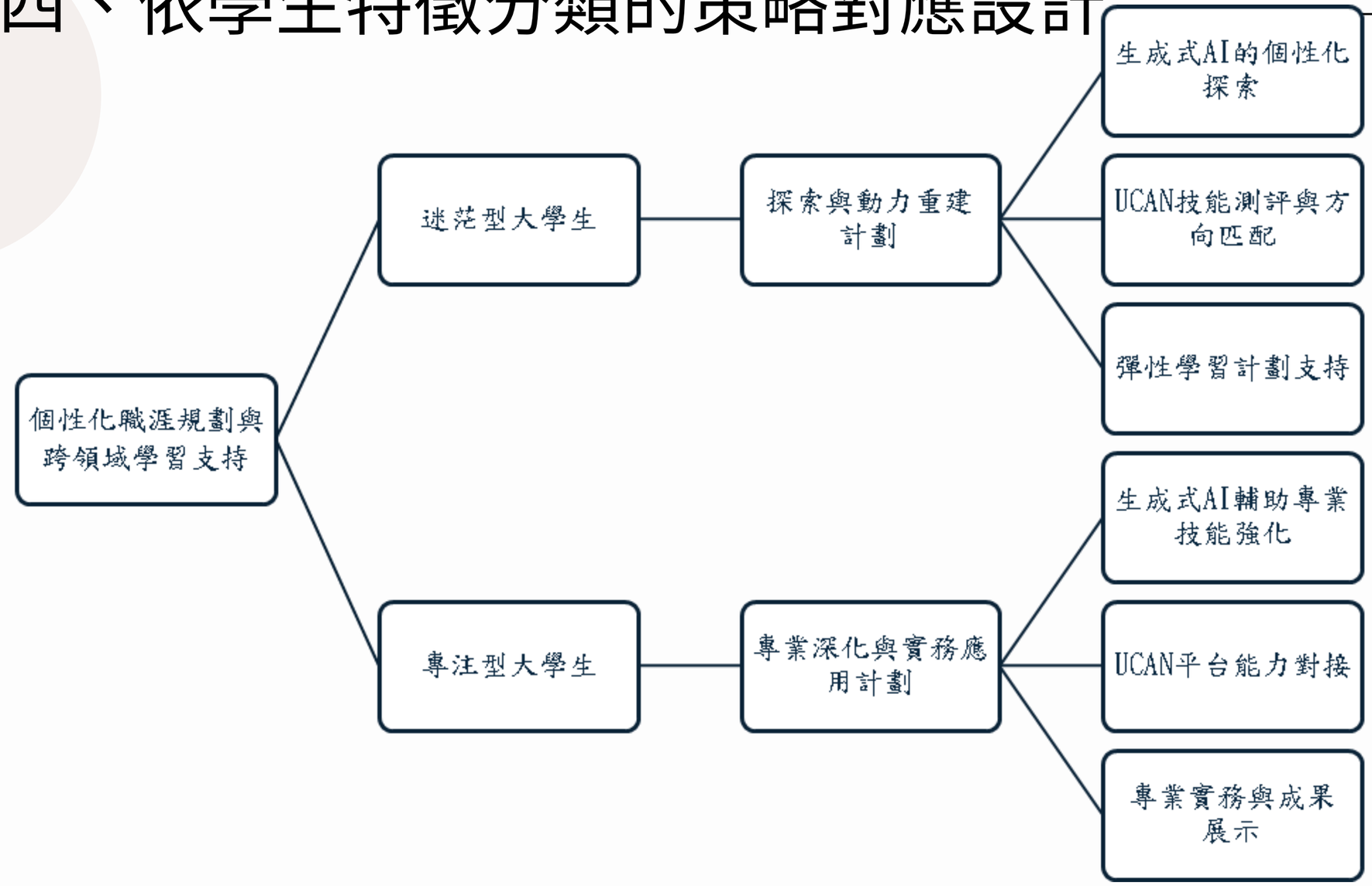
共分為「四大主題」

1. 基本學習情況
2. 學習動力與方向感
3. 學科興趣與專業探索
4. 學習規劃與行動

三之三、學習特徵分類：二分法問卷設計與分析

分類	**專注型學生**	**迷茫型學生**
目標設定	已確定學習目標、依未來發展選課	學習目標不明確、受同儕或課程難易度影響選課
學習動機	對學習有興趣、願意主動學習新知識	缺乏學習興趣、學習態度消極
自我管理	有固定學習計畫、能安排學習進度	無明確計畫、容易拖延
困難處理	遇到問題會搜尋資料或請教他人	遇到困難傾向放棄、不尋求協助
學習參與度	積極參與課外活動、競賽、實習	較少參與學習相關活動

三之四、依學生特徵分類的策略對應設計



三之四、依學生特徵分類的策略對應設計

針對**迷茫型**學生:**探索與動力重建計畫**

興趣探索階段：

- 使用生成式AI進行興趣分析，生成課程推薦報告與個性化發展建議，學生可基於AI生成的課程推薦報告進行選課規劃及最終學習目標擬定。

學習計畫制定：

- 依據AI和UCAN結果，制定個人學習計畫。
- 每月檢視進度，調整計畫以保持學習動力。

動力重建活動：

- 透過參加學長姐經驗分享會，展示不同興趣與專業結合的成功案例；讓學生能夠對自己的未來有更多想法及方向以此重建學習動力與自信。

三之四、依學生特徵分類的策略對應設計

針對**專注型**學生：**專業深化與實務應用計畫**

技能提升階段：

- 推薦生成式AI輔助工具，用於專業技能學習（如：程式代碼生成工具、數據分析模擬）。

實務應用階段：

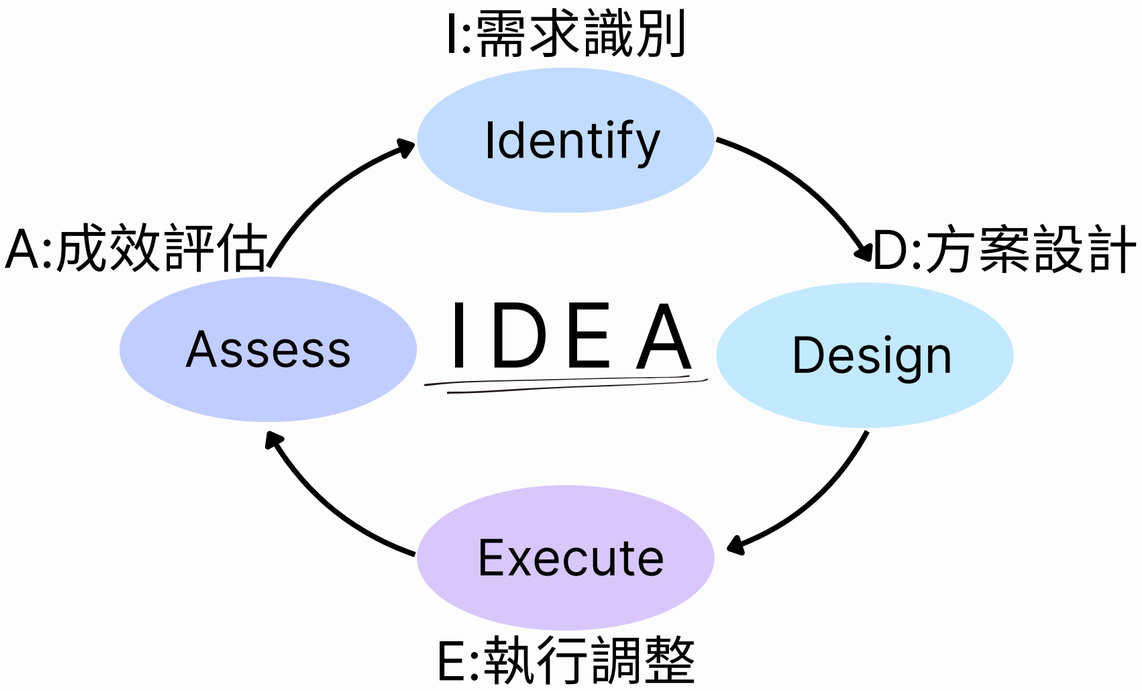
- 引入專案實作檢測，讓學生在實際場景中應用所學。
- 推薦學生參加資管相關比賽（如：黑客松、專案競賽）。

成果展示：

- 舉辦期末專案發表會，學生展示基於AI和UCAN建議完成的專案，藉此展現學生的專業成長與應用能力。

三之五、理論應用設計：I-D-E-A循環模型

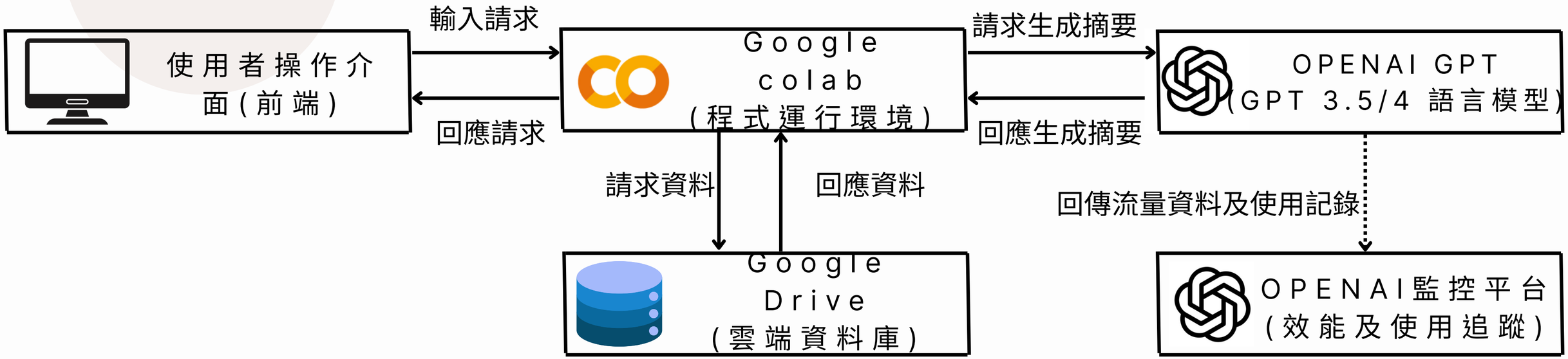
流程階段	功能與描述
Identify (需求識別)	學生資料蒐集（問卷、興趣）、UCAN能力分析、生成式AI進行學生類型判斷
Design (方案設計)	針對學生類型，設計迷茫型與專注型專屬學習與職涯路徑
Execute (執行調整)	學生根據AI建議，進行課程學習、專案參與與技能訓練，期間允許彈性調整
Assess (成效評估)	學習成效分析、職涯準備度評估與回饋，提供後續精進建議



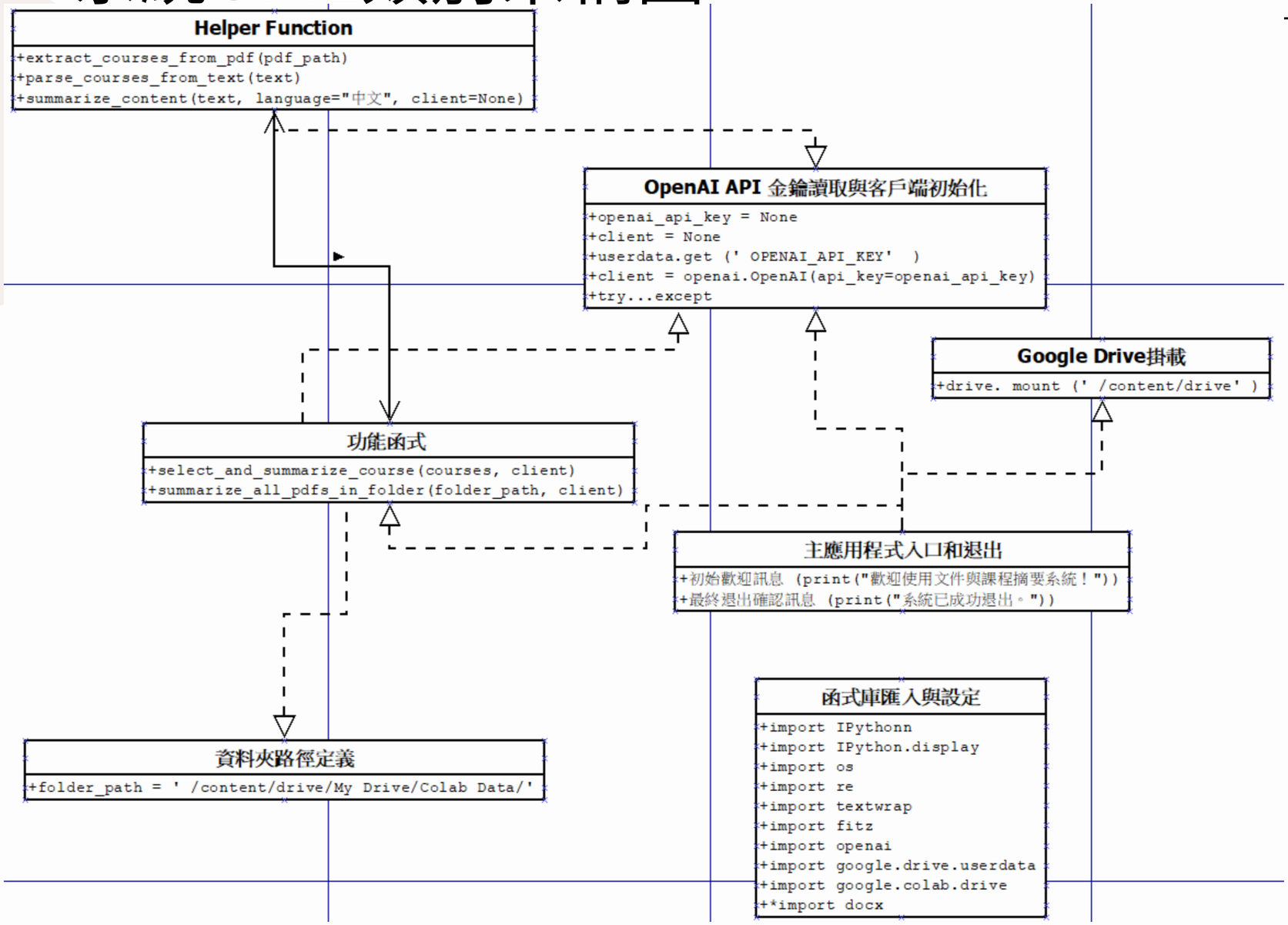
四、解決方案實現與發展

為實現個人化課程文件摘要與輔助功能，本系統採用三層式系統架構設計，包含**「使用者介面層」**、**「應用邏輯層」**以及**「資料與 AI 模型層」**。整體設計以 **Google Colab 作為後端開發環境，結合 Google Drive** 雲端硬碟作為相關應用資料儲存位置，並**串接 OpenAI GPT-4 / GPT-3.5 語言模型**進行自然語言生成及回應。透過此架構，可彈性整合輸入、分析與輸出流程，實現有效且具擴充性的輔助教學系統。

四之二、系統架構圖


使用者操作介面(前端)
輸入請求
回應請求
Google colab (程式運行環境)
請求生成摘要
回應生成摘要
OPENAI GPT (GPT 3.5/4 語言模型)
請求資料
回應資料
回傳流量資料及使用記錄
Google Drive (雲端資料庫)
OPENAI監控平台 (效能及使用追蹤)

四之三、系統UML類別架構圖



四之四、原始碼說明(套件匯入與模組載入)

```
from IPython import get_ipython
from IPython.display import display
import os
import re # Import re for regular expressions
import textwrap # Import textwrap for text wrapping

# 將 fitz 的匯入移到前面並加強安裝檢查
try:
    import fitz  # PyMuPDF for PDF processing
    print("✅ fitz (PyMuPDF) 已匯入。")
except ImportError:
    print("⚠️ 無法匯入 fitz (PyMuPDF)。正在嘗試安裝 pymupdf...")
    try:
        # 使用最新的穩定版本或您需要的版本
        !pip install pymupdf==1.23.12
        print("✅ pymupdf 安裝完成。")
        # 安裝後必須再次嘗試匯入
        import fitz
        print("✅ fitz (PyMuPDF) 安裝並成功匯入。")
    except Exception as e:
        print(f"❌ 安裝 pymupdf 失敗: {e}")
        print("請嘗試手動執行 !pip install pymupdf 或檢查網路連接。")
        # 如果安裝或匯入仍失敗，程式後續依賴 fitz 的部分會出錯


import openai # Assuming openai library is already installed in your environment
from google.colab import userdata # 匯入 userdata 模組

# 確保 python-docx 已安裝
try:
    import docx
    print("✅ python-docx 已匯入。")
except ImportError:
    print("⚠️ 無法匯入 python-docx。正在嘗試安裝 python-docx...")
    try:
        !pip install python-docx==1.1.0
        print("✅ python-docx 安裝完成。")
        import docx # 安裝後重新匯入
        print("✅ python-docx 安裝並成功匯入。")
    except Exception as e:
        print(f"❌ 安裝 python-docx 失敗: {e}")
        print("請嘗試手動執行 !pip install python-docx 或檢查網路連接。")
```

```
# 確保 openai 已安裝 (如果需要)
try:
    import openai
    print("✅ openai 已匯入。")
except ImportError:
    print("⚠️ 無法匯入 openai。正在嘗試安裝 openai...")
    try:
        !pip install openai==1.2.3
        print("✅ openai 安裝完成。")
        import openai
        print("✅ openai 安裝並成功匯入。")
    except Exception as e:
        print(f"❌ 安裝 openai 失敗: {e}")
        print("請嘗試手動執行 !pip install openai 或檢查網路連接。")


# 掛載 Google Drive (如果需要讀取 Colab_Data 資料夾中的檔案)
try:
    import google.colab.drive as drive
    drive.mount('/content/drive')
    print("✅ Google Drive 掛載成功。")
except ImportError:
    print("⚠️ 無法匯入 google.colab.drive，跳過掛載 Google Drive。")
except Exception as e:
    print(f"⚠️ 掛載 Google Drive 失敗: {e}")
```

四之四、原始碼說明(雲端硬碟掛載)

```
# 掛載 Google Drive (如果需要讀取 Colab_Data 資料夾中的檔案)
try:
    import google.colab.drive as drive
    drive.mount('/content/drive')
    print("✅ Google Drive 掛載成功。")
except ImportError:
    print("⚠️ 無法匯入 google.colab.drive，跳過掛載 Google Drive。")
except Exception as e:
    print(f"⚠️ 掛載 Google Drive 失敗: {e}")
```

四之四、原始碼說明(API KEY讀取及API Client建立)

```
#  設定資料夾路徑
#  可以根據需要修改此路徑。
folder_path  =  '/content/drive/My  Drive/Colab_Data/'

#  從  Secret  讀取  API  金鑰並建立  Client
openai_api_key  =  None
client  =  None

print("\n嘗試從  Colab  Secrets  讀取  OpenAI  API  金鑰...")

try:
        openai_api_key  =  userdata.get('OPENAI_API_KEY')  #  從  Secret  讀取
        if  openai_api_key:
                client  =  openai.OpenAI(api_key=openai_api_key)
                print("✅  OpenAI  client  建立成功。")
        else:
                print("⚠️  Colab  Secret  中未找到名稱為  'OPENAI_API_KEY'  的金鑰。")
                print("請確認您已在  Colab  Secrets  中設定此金鑰。")
except  Exception  as  e:
        print(f"⚠️  從  Colab  Secrets  讀取  API  金鑰失敗:  {e}")
        print("請確認您已開啟此  Notebook  對  Secrets  的訪問權限。")
```

四之四、原始碼說明(PDF文字摘要)

```
# Helper Function to Extract Text from PDF
def extract_text_from_pdf(pdf_path):
    """
    Extracts text content from a PDF, limiting the output size.
    """
    if not os.path.exists(pdf_path):
        print(f"找不到檔案: {pdf_path}")
        return ""

    text = ""
    try:
        doc = fitz.open(pdf_path)
        for page in doc:
            # 使用 get_text() 提取文本
            text += page.get_text()
        doc.close() # Close the document after processing
    except Exception as e:
        print(f"⚠️ 檔案 {pdf_path} 讀取失敗: {e}")
        return "" # Return empty string if reading fails

    # Limit text length to avoid exceeding token limits.
    # Adjusted limit based on previous runs
    max_chars = 4000 # Decreased the limit to reduce token usage
    if len(text) > max_chars:
        print(f"ℹ️ PDF 內容已截斷至 {max_chars} 字元以避免 token 過多。")
        return text[:max_chars]
    else:
        return text
```

四之四、原始碼說明(資料分割處理與OPENAI摘要功能連結)

```python
# Helper Function to Parse Courses from Text
def parse_courses_from_text(text):
    """
    Parses course information (name and description) from the extracted text.
    """
    courses = []
    if not text:
        print("⚠️ 沒有文本內容可供解析。")
        return courses

    # Based on previous parsing logic
    current_course_name = None
    current_course_description_lines = []

    # Split the text by '○ ' which seems to be the start of each course entry.
    course_entries = text.split('○ ')

    for entry in course_entries:
        entry = entry.strip()
        if not entry:
            continue

        parts = entry.split('■ 教學目標:')

        if len(parts) > 1:
            course_name_part = parts[0].strip()
            description_part = parts[1].strip()

            name_lines = course_name_part.splitlines()
            if name_lines:
                potential_name = name_lines[-1].strip()
                if len(potential_name) > 1 and any(c.isalnum() for c in potential_name):
                    current_course_name = potential_name
                else:
                    current_course_name = course_name_part # Fallback

            current_course_description_lines = description_part.splitlines()

            if current_course_name:
                description_text = "\n".join(current_course_description_lines).strip()
                courses.append({'name': current_course_name, 'description': description_text})

            current_course_name = None
            current_course_description_lines = []
    return courses
```

```python
# Function to Summarize Text using OpenAI GPT
def summarize_content(text, language="中文", client=None):
    """
    Calls the OpenAI GPT API to generate a summary of the input text.
    Requires an initialized OpenAI client object.
    """
    if not client:
        return "⚠️ 錯誤：OpenAI client 未初始化。請檢查 API 金鑰。"

    if not text:
        return "⚠️ 錯誤：沒有內容可以摘要。"

    system_msg = f"你是一位擅長撰寫{language}摘要的助理。請精簡地摘述以下內容。"
    prompt = f"請幫我以{language}摘述以下內容：\n{text}"

    messages = [
        {"role": "system", "content": system_msg},
        {"role": "user", "content": prompt}
    ]

    try:
        response = client.chat.completions.create(
            model="gpt-4",    # Or another suitable model like "gpt-3.5-turbo"
            messages=messages,
            temperature=0.7,
            max_tokens=500 # Adjust max_tokens based on desired summary length
        )
        return response.choices[0].message.content
    except openai.APIError as e:
        print(f"⚠️ OpenAI API 發生錯誤: {e}")
        if 'context_length_exceeded' in str(e):
            return "⚠️ 錯誤：輸入內容太長，超過模型限制。請嘗試縮減 PDF 內容。"
        return f"⚠️ OpenAI API 錯誤: {e}"
    except Exception as e:
        return f"⚠️ 發生未知錯誤: {e}"
```

四之四、原始碼說明(資料輸出整理)

```
# Helper function to wrap text by word count
def wrap_by_word_count(text, word_limit):
    """
    Wraps text so that each line has approximately word_limit words.
    """
    words = text.split()
    wrapped_lines = []
    current_line_words = []

    for word in words:
        current_line_words.append(word)
        if len(current_line_words) >= word_limit:
            wrapped_lines.append(" ".join(current_line_words))
            current_line_words = []

    if current_line_words:  # Add any remaining words as the last line
        wrapped_lines.append(" ".join(current_line_words))

    return "\n".join(wrapped_lines)
```

四之四、原始碼說明(資料輸出整理)

```python
# Function to Select and Summarize a Course
def select_and_summarize_course(courses, client):
    """
    Displays the list of courses, prompts the user to select one,
    and provides the name and summary of the selected course.
    Includes an option to exit.
    Returns True if a course was selected and summarized, False otherwise.
    """
    if not courses:
        print("⚠️ 沒有課程資訊可供選擇。")
        return False # Indicate that the operation was not completed

    print("\n── 請選擇所要顯示之課程摘要 ──")

    for i, course in enumerate(courses):
        print(f"    {i+1}. {course['name']}")

    while True:
        try:
            user_selection = input(f"\n請輸入課程編號 (1-{len(courses)}) 或輸入 '退出' 以返回: ").strip().lower()

            if user_selection == '退出':
                print("取消課程選擇。")
                return False # Indicate user wants to exit

            selected_index = int(user_selection) - 1

            if 0 <= selected_index < len(courses):
                selected_course = courses[selected_index]
                print(f"\n您已選擇課程: {selected_course['name']}")

                # Ask for the language for the summary
                lang = input("請選擇摘要語言（中文/英文，預設中文）: ").strip().lower()
                if lang not in ["中文", "英文"]:
                    lang = "中文"
                    print("使用預設語言: 中文")

                # Generate the summary of the selected course description
                print("✅ 正在生成課程摘要...")
                summary = summarize_content(selected_course['description'], language=lang, client=client)

                print(f"\n📄 課程 '{selected_course['name']}' 的摘要: \n")
                # Wrap the summary text based on the selected language
                if lang == "中文":
                    elif lang == "英文":
                        # Wrap by word count for English (approx 20 words per line)
                        wrapped_summary = wrap_by_word_count(summary, 15) # Changed word limit to 15
                    else:
                        # Fallback to default wrapping if language is unexpected
                        wrapped_summary = textwrap.fill(summary, width=30)

                    print(wrapped_summary)

                    return True # Indicate successful completion

                else:
                    print("⚠️ 無效的課程編號，請重新輸入。")
            except ValueError:
                print("⚠️ 無效的輸入，請輸入課程編號或 '退出'。")
```

四之四、原始碼說明(資料庫文本資料讀取(文件摘要))

```python
def summarize_all_pdfs_in_folder(folder_path, client):
    """
    Lists PDF files in the specified folder, prompts the user to select one,
    and summarizes the content of the selected file.
    Includes an option to exit.
    Returns True if a summary was generated, False otherwise.
    """
    if not client:
        print("⚠️ 警告：OpenAI client 未初始化，無法執行文件摘要。")
        return False

    if not os.path.isdir(folder_path):
        print(f"⚠️ 錯誤：找不到資料夾或路徑無效：{folder_path}")
        return False

    print(f"\n—— 資料夾 '{folder_path}' 中的 PDF 檔案 ——")

    files = os.listdir(folder_path)
    pdf_files = [f for f in files if os.path.isfile(os.path.join(folder_path, f)) and f.lower().endswith('.pdf')]

    if not pdf_files:
        print("ℹ️ 在指定資料夾中沒有找到任何 PDF 檔案。")
        return False

    for i, filename in enumerate(pdf_files):
        print(f"  {i+1}. {filename}")

    while True:
        try:
            user_selection = input(f"\n請輸入要摘要的檔案編號 (1-{len(pdf_files)}) 或輸入 '退出' 以返回主選單：").strip().lower()

            if user_selection == '退出':
                print("取消文件摘要。")
                return False # Indicate user wants to exit

            selected_index = int(user_selection) - 1

            if 0 <= selected_index < len(pdf_files):
                selected_filename = pdf_files[selected_index]
                selected_file_path = os.path.join(folder_path, selected_filename)

                print(f"\n您已選擇檔案：'{selected_filename}'")

                # 提取 PDF 文本
                pdf_text = extract_text_from_pdf(selected_file_path)

                if pdf_text:
                    # For file summary, let's ask for the language as well
                    lang = input("請選擇摘要語言（中文/英文，預設中文）：").strip().lower()
                    if lang not in ["中文", "英文"]:
                        lang = "中文"
                        print("使用預設語言：中文")

                    print("✅ 內容已提取，正在生成摘要...")
                    # 呼叫 summarize_content 函式進行摘要
                    summary = summarize_content(pdf_text, language=lang, client=client)

                    print(f"\n📄 檔案 '{selected_filename}' 的摘要：\n")
                    # Wrap the summary text based on the selected language
                    if lang == "中文":
                        wrapped_summary = textwrap.fill(summary, width=30) # Wrap by character count for Chinese
                    elif lang == "英文":
                        # Wrap by word count for English (approx 20 words per line)
                        wrapped_summary = wrap_by_word_count(summary, 15) # Changed word limit to 15
                    else:
                        # Fallback to default wrapping if language is unexpected
                        wrapped_summary = textwrap.fill(summary, width=30)

                    print(wrapped_summary)
                    return True # Indicate successful completion
                else:
                    print(f"⚠️ 無法從檔案 '{selected_filename}' 中提取內容，跳過摘要。")
                    return False # Indicate that the operation was not completed

            else:
                print("⚠️ 無效的檔案編號，請重新輸入。")
        except ValueError:
            print("⚠️ 無效的輸入，請輸入檔案編號或 '退出'。")
```

四之四、原始碼說明(主迴圈(使用者介面))

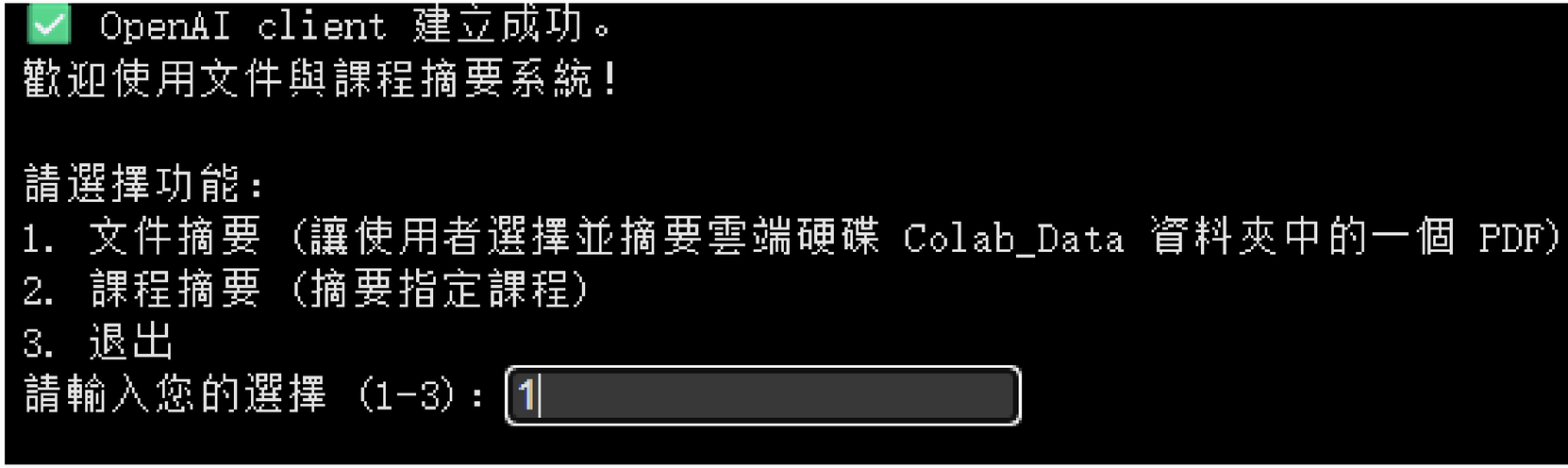
```
# Main application loop
print("歡迎使用文件與課程摘要系統！")

while True:
    print("\n請選擇功能：")
    print("1. 文件摘要 (讓使用者選擇並摘要雲端硬碟 Colab_Data 資料夾中的一個 PDF)")
    print("2. 課程摘要 (摘要指定課程)")
    print("3. 退出")

    choice = input("請輸入您的選擇 (1-3)：").strip()

    if choice == '1':
        print("\n您選擇了文件摘要功能。")
        if client:
            # Call the modified function that handles file listing, selection, and summarization
            summarize_all_pdfs_in_folder(folder_path, client)
            # The summarize_all_pdfs_in_folder function already handles its own exit option.
            # If it returns False (user exited file selection), the loop continues, showing the main menu again.
            # If it returns True (summary generated) or if there was an error before selection,
            # the loop also continues, showing the main menu again.
        else:
            print("\n❌ OpenAI client 未初始化，無法執行文件摘要。請檢查 API 金鑰。")
    elif choice == '2':
        print("\n您選擇了課程摘要功能。")
        if client:
            target_pdf_filename = "國立台東大學資訊管理學系學生所學課程.pdf"
            target_pdf_path = os.path.join(folder_path, target_pdf_filename)

            if os.path.exists(target_pdf_path):
                # 1. 提取 PDF 文本
                pdf_full_text = extract_text_from_pdf(target_pdf_path)

                # 2. 解析 PDF 文本以提取課程資訊
                courses = parse_courses_from_text(pdf_full_text)

                # 3. 顯示課程列表並讓使用者選擇和摘要
                if courses:
                    # select_and_summarize_course function includes its own exit handling
                    select_and_summarize_course(courses, client)
                else:
                    print("⚠️ 未能從 PDF 文本中提取課程資訊。")
            else:
                print(f"\n❌ 找不到目標檔案：'{target_pdf_filename}'。請確認檔案路徑是否正確。")
        else:
            print("\n❌ OpenAI client 未初始化，無法執行課程摘要。請檢查 API 金鑰。")
    elif choice == '3':
        print("\n謝謝您的使用，系統即將退出。")
        break # Exit the loop
    else:
        print("\n無效的選擇，請輸入 1, 2 或 3。")

print("系統已成功退出。")
```

五、案例展示(情境一)

首先我們以「某欲就讀台東大學資管系學生，**想了解**關於東大資管系的**畢業準則、相關規範**以及在學期間應具備之能力等」作為模擬情境之設定，且利用該生成式系統進行詢問(以文件摘要作為主要之使用功能)，最後得知相關資訊。

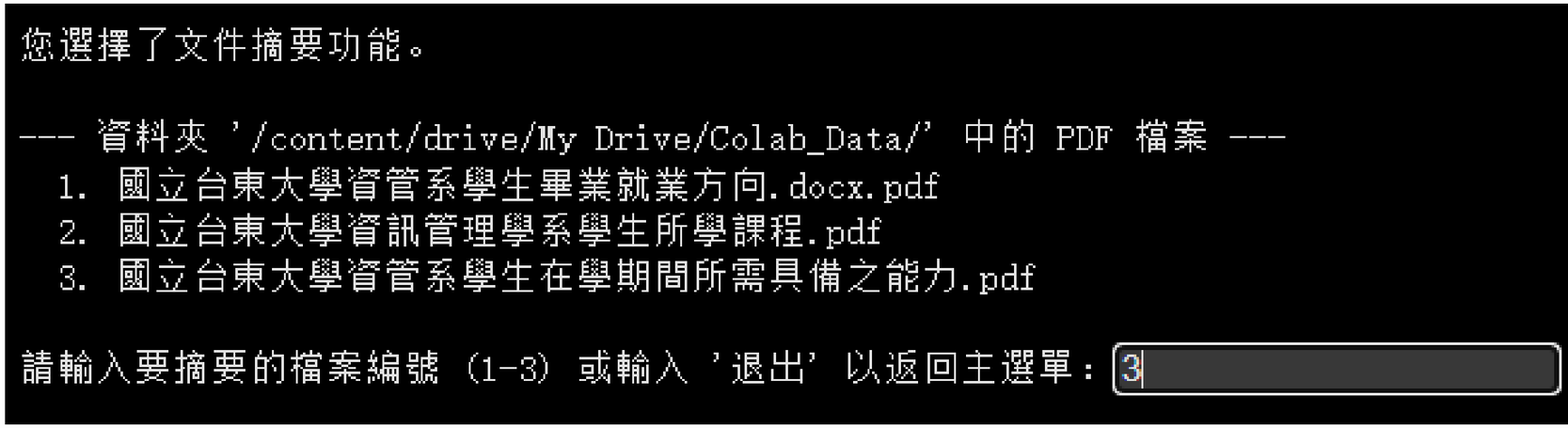
五、案例展示(實作畫面1-1)

首先使用者先選擇欲使用之功能，有文件摘要，課程摘要等功能作為選擇。我們以假設情境為主，**選擇「文件摘要」功能** (輸入" 1 ")，如右圖所示。



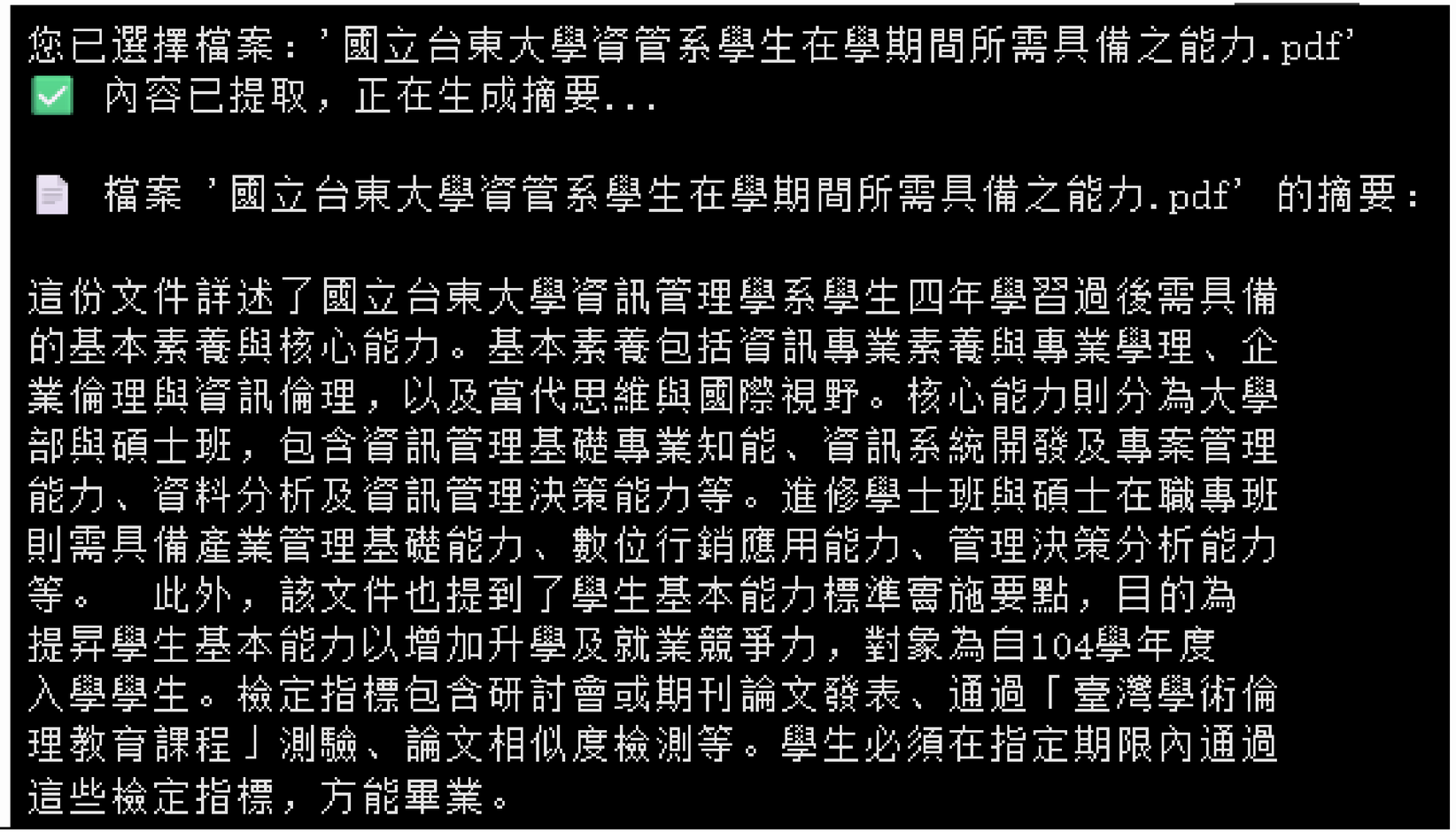
五、案例展示(實作畫面1-2)

選擇完功能後，接著使用者可以**選擇欲摘要之文件**(國立台東大學資管系學生在學期間所需具備之能力.pdf)，輸入"3"，如右圖所示。



五、案例展示(實作畫面1-3)

輸出請求，等待GPT回應，右圖為GPT作出之回應及文件摘要內容。



五、案例展示(情境一輸出說明)

根據上述操作生成的**摘要**:

這份文件詳述了國立台東大學資訊管理學系學生在學期間需要具備的基本素養與核心能力。基本素養包括資訊專業素養與學理、企業與資訊倫理、當代思維與國際視野。

核心能力則依學制不同，大學部需具備資訊管理基礎專業知能、資訊系統開發及專案管理能力等五項；碩士班需具備資訊系統開發能力、資訊管理決策能力及整合應用能力三項。

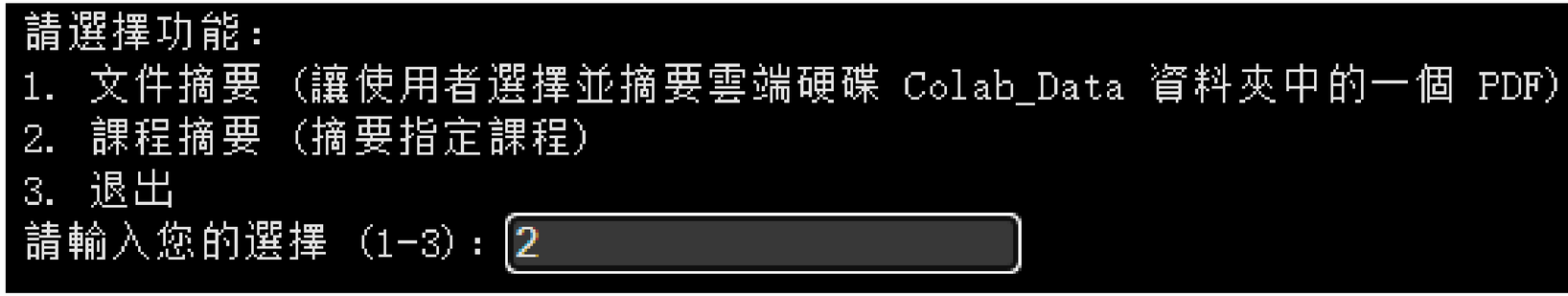
此外，也提供了實施要點，包括實施目的、對象、檢定指標及方式等，目的是提升學生基本能力與升學及就業競爭力，並針對碩士與學士班學生設置不同檢定指標，如發表論文、通過測驗、取得證照等。

五、案例展示(情境二)

再來我們以「某欲就讀台東大學資管系學生，想了解關於東大資管系的相關課程」作為情境二之假設情況，**欲瞭解之課程**包括「程式設計」、「作業系統原理與實務」、「管理資訊系統」等，透過摘要回應，學生可以了解特定課程之教學內容及概述。

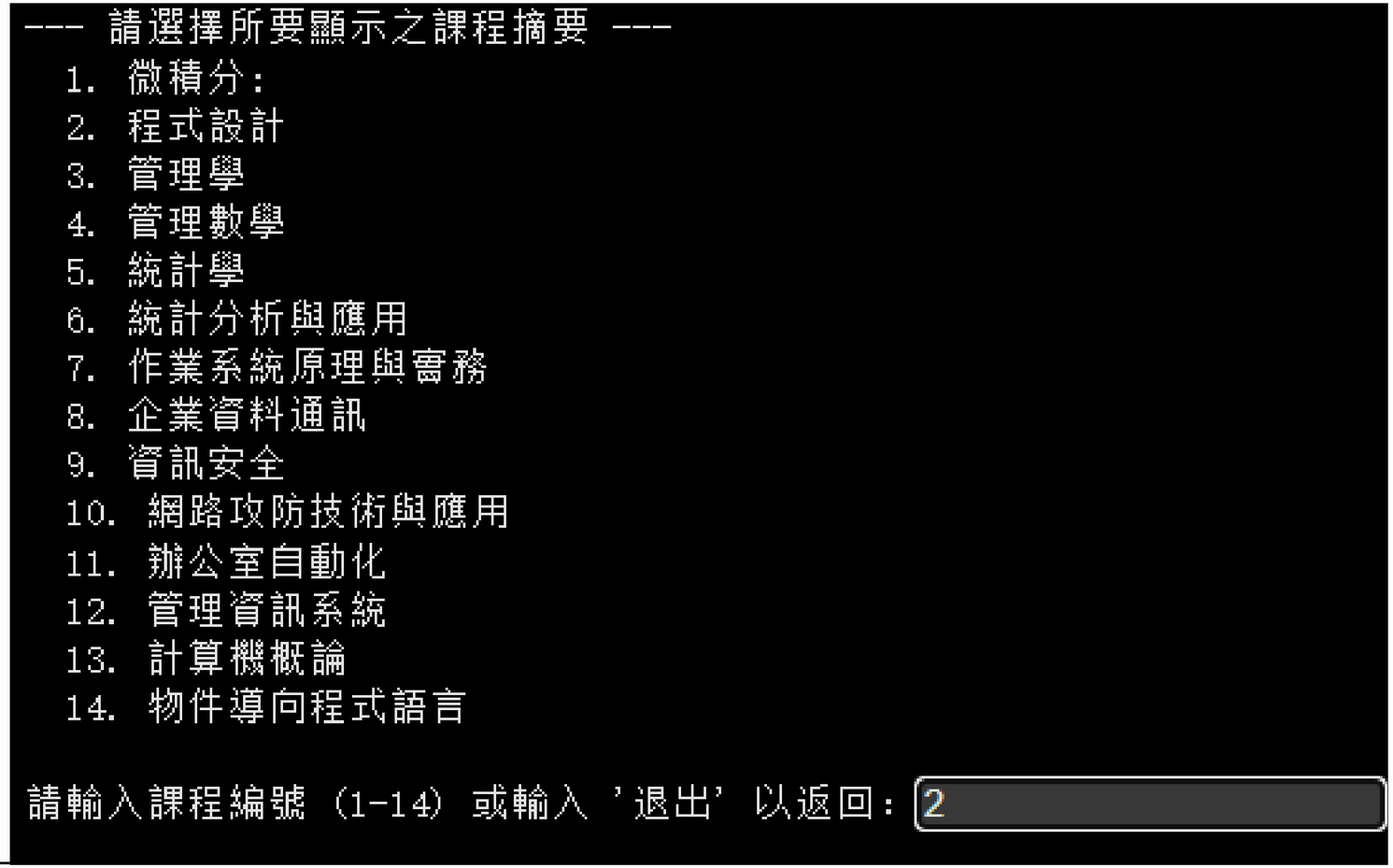
五、案例展示(實作畫面2-1)

使用者首先根據情境設定**選擇「課程摘要」**功能(輸入"2")，如右圖所示。



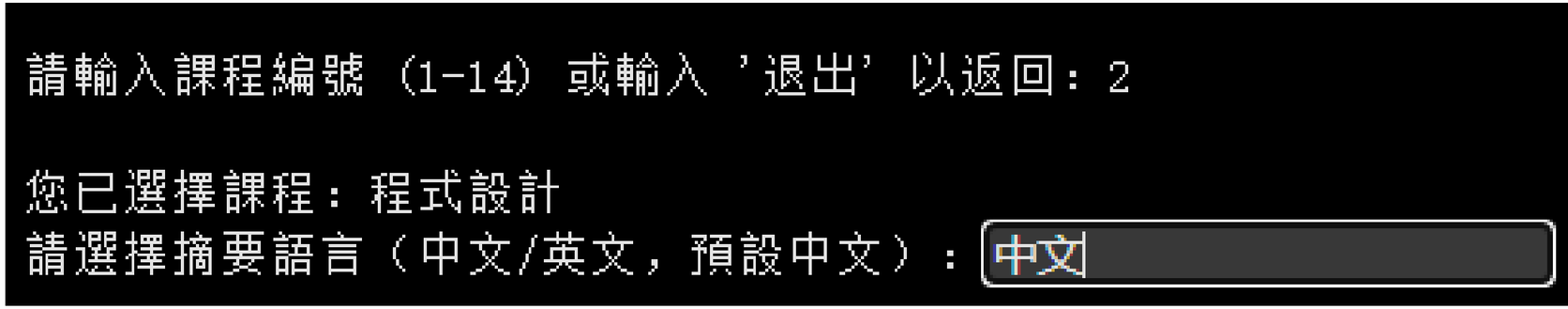
五、案例展示(實作畫面2-2)

系統收到指令後，會**列出課程**檔案中所摘要到的課程名稱並列出，**使用者輸入編號**，根據有興趣之課程內容進一步了解，右圖為條列之課程，並以「程式設計」課程作為範例選擇(註:因目前系統尚未擴充，故課程尚未完全匯入)。



五、案例展示(實作畫面2-3)

生成摘要**可依據使用者習慣調整語言**，目前提供中文及英文兩種語言供使用者應用，預設則為中文形式輸出如右圖所示。



五、案例展示(實作畫面2-4)

此為「程式設計」課程」之**中英文之課程摘要生成展示**。

生成的摘要：

此內容主要涵蓋了對作業系統的全面認識，包括其組織架構、抽象化與虛擬化設計概念，並能掌握各種作業系統（如Linux、Windows、Android）的具體實例與發展趨勢。同時，也明瞭硬體介面、程式介面、及使用者介面的分工關係，以及個人電腦硬體組裝和簡易檢修程序。此外，還能在實體機器和虛擬機器上完成Linux作業系統的安裝，並深入學習電腦節能方法與作業系統的節能管理。對於Linux檔案系統與磁碟管理，以及Linux使用者與群組管理，也有深入的了解和實際操作能力，並能應用至其他作業系統。另外，也瞭解處理程序結構、多執行緒結構，以及實體記憶體與虛擬記憶體的組織，並能透過API進行程式設計。最後，還能執行Linux程序控制與管理，以及Linux記憶體與系統資源觀察，並能安裝Apache和IIS伺服器，撰寫簡單的PHP和ASP.NET動態網頁程式。

生成的摘要：

The summary focuses on understanding the basic principles of programming language evolution and compiler linking, as well as developing foundational skills in structured program design. This includes the ability to declare variables and input data using C++ syntax, output calculation results on the screen, apply arithmetic expressions to solve computational problems, use conditional branching to handle decision-making, and use loops for repetitive work. Other abilities cultivated include handling arrays, strings, and pointers, differentiating variable scope and lifespan, processing large amounts of data using arrays, and passing variable addresses using pointer variables. It also emphasizes modular problem-solving and object-oriented design skills, using functions to modularize problems, utilizing existing functions, understanding C++ abstract data types, defining classes, declaring objects, and simplifying design with objects. Through small programming projects, it fosters potential for innovative information service design and cultivates teamwork and problem-solving skills.

六、測試結果評估

👤 **受測者資料（S01）**

- 年級／性別：大三／男
- UCAN興趣類型：事務型(C)、實用型(R)、社會型(S)
- 興趣:傾向為資料處理、系統操作與邏輯推理
- 核心職能擅長：資訊組織、工具操作、邏輯推演

💻 操作過程

- **輸入課程**：「程式設計」、「作業系統原理與實務」以及「管理資訊系統」

📄 **生成摘要**：

1.程式設計：學習 C++ 程式語法與邏輯，掌握陣列、字串、指標與物件導向觀念，透過微專題實作強化解題與團隊合作能力。

2.作業系統原理與實務：介紹作業系統架構與功能，實作 LINUX 安裝與系統管理，學習多執行緒與記憶體管理，並能建置伺服器與動態網頁。

3.管理資訊系統:了解資訊系統在企業中的角色與應用，探討科技如何提升決策效率與競爭力，並關注資訊管理的最新發展趨勢。

六、測試結果評估

💬 使用者回饋

- 使用者表示 AI 摘要幫助他**快速理解課程內容**，能排除與其職涯規劃不符的課程，尤其在選課初期提供具參考價值的篩選依據。

🔍 研究者觀察

- 約 5 分鐘內完成 3 門課摘要敘述，效率高。
- AI 描述語句清晰、準確性與實用性高。
- 未來可再加入職涯職位資料庫，強化學用對接建議。

六、測試結果評估

👤 **受測者資料（S02）**

年級／性別：大三／女

UCAN興趣類型：社會型（S）＋企業型（E）

興趣傾向：團隊協作、溝通管理與實務操作

核心職能擅長：組織溝通、計畫推動、人際互動

💻 操作過程

輸入課程：「管理學」、「管理數學」、「辦公室自動化」

📄 **生成摘要**:

1. 管理學：結合理論與實務，認識管理功能、效率與效能，透過案例研討提升邏輯思考與表達力，培養專業管理素養。

2. 管理數學：建立數學與邏輯思維基礎，強化在管理與資訊應用中的數學解題能力，為進階課程奠定應用基礎。

3. 辦公室自動化：學習以 Python 自動處理 Excel、Word、簡報與郵件，提高辦公效率並強化資訊處理與決策能力。

六、測試結果評估

💬 使用者回饋

- 使用者表示 AI 摘要讓她快速掌握課程重點，並**理解各門課程與她興趣在溝通與管理的連結**。
- 她特別認為辦公室自動化的摘要實用，感受到數位時代對於現代辦公及日常性或例行性之作業處理的影響與助益。

🔍 研究者觀察

- 使用者在選課過程中，明顯偏好與團隊管理相關的課程。
- AI 系統的摘要有效**輔助學生快速評估課程內容與興趣匹配度**，提升選課效率。

六、測試結果評估

👤 **受測者資料（S03）**

年級／性別：大三／男

UCAN興趣類型：企業型（E）＋實用型(R)＋社會型（S）

興趣傾向：偏向策略規劃、執行管理及實務操作

核心職能擅長：組織溝通、計畫推動、人際互動

💻 操作過程

輸入課程「管理資訊系統」、「統計分析與應用」、「辦公室自動化」。

📄 **系統生成摘要**：

1. 管理資訊系統： 了解資訊系統在企業中的角色與應用，探討科技如何提升決策效率與競爭力，並關注資訊管理的最新發展趨勢。
2. 統計分析與應用：本課程教授使用開源軟體JASP進行敘述統計、假設檢定、變異數分析、因素分析、迴歸分析及結構方程模型等統計分析方法與資料視覺化的應用，以培養學生進行學術研究所需的資料分析與視覺化能力。
3. 辦公室自動化：學習以 Python 自動處理 Excel、Word、簡報與郵件，提高辦公效率並強化資訊處理與決策能力。

六、測試結果評估

💬 使用者回饋

使用者認為系統摘要精準且重點突出，有助於他**判斷課程是否符合職涯規劃**，並且他特別感興趣「辦公室自動化」這門課程，覺得能幫助未來職場效率及作為有效之工作輔助。

🔍 研究者觀察

- 受測者展現明確的職涯目標，偏好實務與策略結合的課程。
- AI摘要功能**有效縮短資訊搜尋時間**，促使他更快做出選擇。

七、平台與UCAN連結及模擬課程推薦

目前平台已架構出有助於學生了解課程內容介紹及教學內容摘要，後續將以模擬方式進一步與該生所做的UCAN職涯問卷結果做連結及整合，並推薦相關合適課程給該生，並做後續評估。

七、平台與UCAN連結及模擬課程推薦

我們將以S01受測者作為例，該生在有興趣之課程中選擇「程式設計」、「作業系統原理與實務」、「管理資訊系統」，並搭配其所完成之UCAN「職業興趣探索診斷」結果，其在職涯類型「資訊科技」、「科學、技術、工程、數學」之興趣強度皆得到滿分，首先初步判斷該生之興趣與資管系教學內容相符(如下頁兩圖所示)。

七、平台與UCAN連結及模擬課程推薦

16職涯類型	0 興趣強度分數 5	PR值
建築營造	4.40	99
天然資源、食品與農業	3.80	98
藝文與影音傳播	3.50	93
企業經營管理	4.40	99
教育與訓練	4.70	99
金融財務	4.10	97
政府公共事務	4.70	99
醫療保健	4.40	99
個人及社會服務	4.70	99
休閒與觀光旅遊	4.70	99
資訊科技	5.00	99
司法、法律與公共安全	4.40	98
行銷與銷售	4.70	99
製造	4.40	99
科學、技術、工程、數學	5.00	99
物流運輸	4.70	99

1. 全體PR值是將您的測驗結果與UCAN平台全體學生進行比較，例如您在建築營造的百分等級是PR92，意思是您對於建築營造職涯類型的興趣程度在100人裡面位居92。
2. 深色項目為興趣度相對高分的4~6種職涯類型，若同分的職涯類型過多，則僅取最高分的類型。

職業性格	職業性格代碼0	職業性格傾向 12	PR值
事務型	C	12	99
實用型	R	12	99
社會型	S	12	99
企業型	E	11	98
研究型	I	10	94
藝術型	A	8	88

1. 全體PR值是將您的測驗結果與UCAN平台全體學生進行比較，例如您在藝術型的百分等級是PR92，意思是平台中做UCAN興趣探索測驗的100人中，您的職業興趣傾向於藝術型性格高於92人。
2. 職業性格代碼是分數較高的2到4種職業性格。透過2到4種職業性格類型的結合，能夠較為完善地描述個人的職業性格與所屬類型。

七、平台與UCAN連結及模擬課程推薦

經過該生進行課程摘要及相關文件閱讀後，加上其已完成UCAN職業興趣探索診斷，結果顯示其對**「資訊科技」**、**「科學、技術、工程、數學」**等領域感興趣，且Holland碼(職業性格)為CRS，並已對**「程式設計」**、**「作業系統原理與實務」**、**「管理資訊系統」**等課程產生興趣，我們模擬其在學士班課程修習的推薦(如下頁所示)。

七、平台與UCAN連結及模擬課程推薦

👨‍🎓 大一上
- 微積分（院共同）3
- 程式設計（院共同）3
- 經濟學（基礎必修）3
- 計算機概論（核心必修）3
- 辦公室自動化（基礎選修）3

👨‍🎓 大一下
- 管理數學（基礎必修）3
- 會計學（基礎必修）3
- 作業系統原理與實務（基礎必修）3
- 物件導向程式語言（核心必修）3
- 管理資訊系統（基礎必修）3

👨‍🎓 大二上
- 管理學（基礎必修）3
- 資料結構（核心必修）3
- 企業資料通訊（基礎必修）3
- 行銷管理（專業模組1選修）3
- 網頁程式設計（核心選修）3

👨‍🎓 大二下
- 統計學（基礎必修）3
- 系統分析與設計（核心必修）3
- 資料庫管理系統（核心選修）3
- 電子商務（專業模組2選修）3
- 財務管理（專業模組1選修）3
- 企業資源規劃（專業模組1選修）3

👨‍🎓 大三上
- 軟體工程（核心必修）3
- 統計分析與應用（基礎選修）3
- 生產與作業管理（專業模組1選修）3
- 雲端運算與應用（專業模組2選修）3
- 資訊管理專題A（核心必修）1
- 消費者行為（專業模組2選修）3

👨‍🎓 大三下
- 資訊系統專案管理（核心必修）3
- 決策支援系統（專業模組1選修）3
- 網路行銷（專業模組2選修）3
- 資訊管理專題B（核心必修）1
- 自由學分2（可再選其他課程或跨領域）

七、平台與UCAN連結及模擬課程推薦成果評估

此修課與學習架構高度契合該生對**「程式設計」、「作業系統原理與實務」、「管理資訊系統」**及STEM領域的興趣，亦呼應其Holland碼CRS職業性格。

課程設計涵蓋基礎程式語言與系統分析與設計、資料結構、雲端運算等實作性強的科目，能滿足R（**實用型**）對具體技術操作的需求；搭配ERP、決策支援系統及行銷管理等商業應用課程，強化C（**事務型**）在資訊管理流程中的敏銳度；而S（**社會型**）面則透過網路行銷與消費者行為等課程，**培養使用者導向思維**。

整體而言，該架構有助於**建立紮實的資訊專業基礎**，並**強化跨域整合與實務應用能力**，為未來進入數位轉型、商業智慧或資訊服務相關職涯奠定良好基礎。

八、討論與結論

討論研究價值

- AI與UCAN平台整合，有效**提升學生學習動機**
- 提供動態、職涯導向的**個人化課程規劃**
- 顯現出教育科技創新之潛力

研究貢獻與成果

- 建構初步可行之智慧課程推薦系統原型
- 明確展示AI技術於高等教育應用的**潛力與價值**

結論與未來建議

- 進一步提升AI模型的**精確性**與職涯資料整合
- **擴展**至其他系所與學校進行更多使用者測試
- 探討**跨領域**與更多資料型態的整合可能性

Thanks!