

元智大學資管系
第三十屆應用類畢業專題頂石課程(一)
期末報告

運用 LLM 與 RAG 開發個人化成癮風險評估輔助系統

1111604 胡家璿、1111643 傅詩婷

實習公司：
精誠資訊股份有限公司
宜誠資訊股份有限公司

工作代號：T2、T1
指導教授：林志麟 博士

中華民國 114 年 11 月
November, 2025

Contents

Chapter1. 緒論	3
Chapter2. 相關技術介紹（本學期延伸部分）	4
2.1 建立短期問答記憶池（Short-term Memory Buffer）	4
2.2 引入語義嵌入與相似度分析（Semantic Embedding and Similarity Analysis）	4
2.3 加權風險評分模型（Weighted Risk Scoring Model）	4
Chapter3. 研究方法	5
3.1 更新模型版本	5
3.2 完善 Prompt 語言與邏輯約束	5
3.3 加入語義分析機制（Semantic Analysis）	6
3.4 建立短期問答記憶池（Short-term Memory Buffer）	7
3.5 導入加權風險評分機制	8
3.6 強化輸出結構與回饋內容	9
Chapter4. 系統展示	11
4.1 系統架構	11
4.2 系統回應展示	11
4.3 系統回應錯誤與行為偏差分析	12
4.4 系統優化策略	13
4.5 系統優化成果	14
Chapter5. 結論	16
Reference	17
附錄 A. 實習工作內容	18
➤ 胡家璿	18
➤ 傅詩婷	18
附錄 B. 實習心得與建議	19
➤ 胡家璿	19
➤ 傅詩婷	19

Chapter1. 緒論

在本專題中，我們聚焦於現代社會中常見的成癮問題，期望設計出一個能夠提供個人化建議的系統，協助使用者提升對自身行為的覺察與改善。在實習過程中，我們觀察到不同年齡層與職業背景的人，常因壓力、情緒低落或生活節奏失衡而產生各種成癮行為，例如過度購物、長時間滑手機、頻繁飲酒或無法戒除吸菸等，這些行為不僅影響身心健康，也可能導致焦慮、睡眠障礙、經濟壓力與人際關係惡化。

由於實習部門與保險產業相關，我們進一步發現個人健康狀況與保險條件之間存在高度連動。成癮行為雖常被忽視，實際上卻與慢性病風險及心理健康密切相關，甚至可能影響核保條件與保費評估。因此，本系統希望在保險流程的前期，協助使用者進行初步自我檢測，作為改善生活習慣與健康狀況的參考依據。

上學期的研究以 RAG (Retrieval-Augmented Generation) 架構為核心，結合語言模型與可靠資料來源，讓系統能在提供建議時維持專業性與可信度。透過此架構，模型能根據成癮理論與統計資料回應並協助進行初步的成癮傾向分析。然而實際運作中，我們發現系統仍存在部分不穩定或偏離預期的回應現象。例如，模型偶爾會出現語言輸出不一致、對相同問題反覆追問、未能整合多輪對話資訊形成完整結論，以及缺乏對使用者先前回答的記憶等問題。這些情況不僅降低了互動的流暢度，也使模型在進行風險評估時缺乏連貫的上下文理解，影響整體使用體驗。

因此，本學期的研究以「系統回應錯誤與行為偏差分析」為主要延伸方向，針對上述問題進行結構化分析與功能優化。我們從語言輸出穩定性、語意理解深度與互動一致性三個層面著手，提出以下改進策略：

1. **建立短期問答記憶池 (short-term memory buffer)：**結結合語義相似度檢查機制，使系統能記錄並辨識先前的對話內容，延續上下文語意、避免重複提問與語意錯亂，提升對話連貫性與互動穩定度。
2. **引入自訂化語義解析模組 (Semantic Parsing Module)：**使模型能從使用者輸入中自動擷取行為特徵（如頻率、強度、控制力、影響程度與情緒依附等），並依據臨床上具意義的層面進行語義分析，提升系統對自然語言的理解準確性與一致性。
3. **整合加權風險評分邏輯 (Weighted Risk Scoring Model)：**將臨床常用的判斷指標轉化為可量化的結構化評估，使系統除文字分析外，亦能提供具體的風險分數與等級判斷，進一步提升輸出內容的可讀性與臨床參考價值。

透過上述優化，我們期望讓系統在理解層面更貼近真實對話脈絡，能根據使用者的語意特徵動態生成更具針對性的回饋內容。除此之外，評分結構的引入也讓成癮風險分析具備可比較與可追蹤性，有助於未來進一步發展長期行為監測或健康風險預警等應用。整體而言，本階段的研究不僅強化了系統的語意判斷能力與互動穩定性，也為後續實作更高層次的個人化健康輔助系統奠定基礎。

Chapter2. 相關技術介紹（本學期延伸部分）

本專題中主要使用到三項核心技術，分別是短期問答記憶池、引入語義嵌入與相似度分析，以及整合加權風險評分邏輯，以改善上下文理解、語意判斷準確性與風險評估結構化表達的能力。

2.1 建立短期問答記憶池（Short-term Memory Buffer）

為了改善模型在多輪對話中的上下文理解能力，系統新增了短期問答記憶池，可暫存最近五輪問答內容。當新的輸入進來時系統會透過語義相似度比對，判斷當前問題是否與先前主題相關。若相似度高便自動檢索對應的對話內容作為輔助參考，使模型能延續先前語意而非重複提問。

此設計能有效降低語意錯亂與對話不連貫的情形，提升上下文連貫性，讓系統在互動上更接近具有記憶能力的智能對話輔助工具。

2.2 引入語義嵌入與相似度分析（Semantic Embedding and Similarity Analysis）

為了讓模型能更準確理解使用者的自然語言輸入，本研究導入 embedding 模型進行語意相似度判斷，透過向量化方式分析使用者的描述內容，從中擷取與行為相關的語義資訊。

系統會根據語意層面的相似度計算，辨識輸入中與成癮行為有關的特徵，例如頻率、強度、控制力、影響程度與情緒依附等。

這樣的語義分析方式，使系統不再依賴固定關鍵詞，而能理解不同語氣與表達下的同義概念（如「停不下來」與「克制不住」），從而提升模型對行為敘述的理解能力與解析準確度，為後續的風險評估與行為建議提供更可靠的語意基礎。

2.3 加權風險評分模型（Weighted Risk Scoring Model）

為了讓模型的回覆更具臨床參考價值，本研究建立以五個層面為基礎的加權風險評分架構：頻率、強度、控制力、影響程度與情緒依附。

系統會依照使用者輸入內容為各層面打分，並依臨床邏輯設定不同權重（控制力 1.6、影響程度 1.4、情緒依附 1.3），最終計算加權總分並對應風險等級（高／中／低）。

此設計能將原本主觀的成癮風險判斷轉化為可量化的結構化評估，使系統輸出結果更具一致性、可比較性與臨床解釋價值。

Chapter3. 研究方法

這學期主要針對上學期系統的問題（輸出不穩定、重複提問、缺乏記憶）做功能強化與行為修正，實際完成的部分如下：

1. 更新模型版本
2. 完善 Prompt 語言與邏輯約束
3. 加入語義分析機制（Semantic Analysis）
4. 建立短期問答記憶池（Short-term Memory Buffer）
5. 導入加權風險評分機制
6. 強化輸出結構與回饋內容

3.1 更新模型版本

由於上學期使用的語言模型已停止服務，為確保系統可持續運行，本階段改採用 llama-3.1-8b-instant 作為主要生成模型。此模型在成本與運算效能之間取得良好平衡，能滿足系統在對話生成上的基本需求，確保研究能在可控資源範圍內持續開發與測試其他功能模組。

```
model = "llama-3.1-8b-instant"  
base_url="https://api.groq.com/openai/v1"
```

為確保模型能在正確的運行環境中執行，系統同時在初始化階段載入必要的通用函式庫，包括 re、json 與 numpy。這三個模組是支撐後續語義分析與相似度計算的重要基礎，例如 re 用於關鍵字與數值擷取，json 負責處理模型的輸出格式，而 numpy 則用於語義嵌入向量之間的運算與相似度計算。這些模組的載入確保了模型在更新至 llama-3.1-8b-instant 後，仍能保持穩定的資料處理能力與運算效率。

```
import re  
import json  
import numpy as np
```

3.2 完善 Prompt 語言與邏輯約束

在系統實際運行過程中，發現模型偶爾會以英文或其他語言回覆，導致語言表達不一致的問題。主要是因為 llama-3.1-8b-instant 的預訓練資料同時包含多國語言，在缺乏明確語言約束時，模型可能會因語境模糊或遇到英文 token 而自動切換語言。為解決此問題，我們重新設計並強化系統提示詞（Prompt），在生成邏輯中明確加入語言控制規則：

```
若模型輸出為英文，請自動將回應翻譯為繁體中文後再輸出。若模型輸出中混有英文或其他語言  
詞彙，請自動翻譯成自然的繁體中文並保持語意不變。
```

此設計能在 Groq API 缺乏語言鎖定 (language lock) 參數的情況下，透過提示詞層級的控制，確保模型在所有互動中皆以繁體中文回覆，有效避免多語混雜導致的語意錯誤與閱讀不一致性，並使系統的回應語氣與語言風格更貼近本地使用情境。

3.3 加入語義分析機制 (Semantic Analysis)

在語義分析模組中，`semantic_similarity()` 是核心的語意比對函式，主要負責計算使用者輸入文字與特定關鍵詞之間的語意相似度。其運作原理是透過 embedding 模型 將文字轉換為高維向量，並以餘弦相似度 (cosine similarity) 公式計算兩者語意上的接近程度。當相似度高於設定閾值 (`threshold = 0.7`) 時，系統即判定該輸入與關鍵詞具有語意關聯。

此設計使系統能理解不同表達方式下的同義語句，例如「停不下來」與「克制不住」雖然字面不同，但在語意空間中相似度高，因此都能被辨識為失控傾向相關行為。這種語義層的比對方法有效提升了模型對自然語言的理解能力與彈性，使其能跨越單純關鍵字比對的限制，進一步接近臨床語意判斷的方式。

```
def semantic_similarity(text, keywords, embedding_model,
threshold=0.7):
    """
    用 embedding 模型做語意相似度判斷。
    threshold 建議 0.7~0.8 (越高越嚴格)
    """
    try:
        text_vec = np.array(embedding_model.embed_query(text))
        keyword_vecs = [np.array(embedding_model.embed_query(k)) for k
in keywords]
        sims = [np.dot(text_vec, kv) / (np.linalg.norm(text_vec) *
np.linalg.norm(kv)) for kv in keyword_vecs]
        return max(sims) > threshold
    except Exception as e:
        print(f"[Warning] similarity check failed: {e}")
        return False
```

在語義分析模組中，頻率關鍵字主要用來判斷使用者是否明確提及其行為發生的時間頻率，例如每天、每週、偶爾等描述。此部分利用前述的 `semantic_similarity()` 函式，將使用者輸入與一組代表頻率概念的關鍵詞進行語意比對。

當輸入語句與關鍵詞的語意相似度超過 0.7 時，系統會判定使用者的描述中有頻率描述，反之則標記為未明確提及。這種設計能讓系統辨識出自然語言中不同形式的時間表達，不僅限於單一詞彙的比對，例如能同時理解「幾乎每天」與「一天至少一次」等變化型敘述。

以下為頻率的實作範例程式片段：

```
# 頻率關鍵詞
```

```
freq_keywords = ["每天", "每週", "每月", "常常", "偶爾", "有時", "幾天", "一小時", "早上", "晚上", "一天", "一週", "一個月"]
freq_semantic = semantic_similarity(user_input, freq_keywords,
embedding_model, threshold=0.7)
freq = "有頻率描述" if freq_semantic else "未明確提及"
```

3.4 建立短期問答記憶池 (Short-term Memory Buffer)

為了讓系統能理解多輪對話的語境並避免重複提問，本專題新增短期記憶機制，讓模型能暫存使用者最近五輪的問答內容。在每次新輸入時，系統會先檢查記憶池是否有資料，若存在先前的對話，就會計算目前輸入與過去問題之間的語義相似度，以判斷是否屬於相同主題。

以下這段程式會從記憶池中找出與當前輸入最相似的歷史對話，並依照相似度排序後取出前兩筆作為輔助參考，確保系統回應時能延續先前的語意脈絡。

```
def retrieve_memory_relevant(user_input, top_k=2):
    if not memory_buffer:
        return ""

    query_emb = np.array(embedding_model.embed_query(user_input))
    sims = []
    for past_q, past_a in memory_buffer:
        past_emb = np.array(embedding_model.embed_query(past_q))
        sim = np.dot(query_emb, past_emb) / (np.linalg.norm(query_emb)
* np.linalg.norm(past_emb))
        sims.append(sim)

    top_idx = sorted(range(len(sims)), key=lambda i: sims[i],
reverse=True)[:top_k]
    related = "\n\n".join([
        f"使用者曾說：{memory_buffer[i][0]}\nAI 回覆：
{memory_buffer[i][1]}"
        for i in top_idx
    ])
    return related
```

下列程式則負責初始化記憶池與設定最大容量，讓系統的記憶保持短期性，避免佔用過多運算資源，同時確保只保留最近的對話。

```
memory_buffer = []
max_memory_size = 5
```

而在主要聊天流程中，當使用者輸入新問題時，系統會先檢查記憶池中是否存在相關內容，若有相似對話則自動將其附加至檢索內容，幫助模型在回答時參考舊資料，這樣模型在回覆時能根據語義延伸過去的主題，而不是重新開始。

```
# 加入記憶池中的相關內容
related_memory = retrieve_memory_relevant(user_input)
if related_memory:
    retrieved_chunks = f"{retrieved_chunks}\n\n(以下是過去對話的相關內容，供你參考)\n{related_memory}"
```

最後當回覆生成後，會將這輪問答存入記憶池中，並確保總數不超過五輪。這樣的設計讓系統能保留短期語意連續性，同時避免記憶過載或語義干擾，使對話更自然流暢。

```
# 更新記憶池（維持固定大小）
memory_buffer.append((user_input, reply))
if len(memory_buffer) > max_memory_size:
    memory_buffer.pop(0)
```

3.5 導入加權風險評分機制

此部分主要用於量化分析使用者的成癮風險，讓語言模型的判斷結果更具臨床參考價值。系統設計五個主要層面：「頻率、強度、失控程度、影響程度、情緒依附」，每個層面以 0-5 分制表示嚴重程度，並依照臨床邏輯設定權重，反映其對成癮風險的相對重要性。而後模型根據使用者的語意輸入進行推理，計算加權總分，並判定風險等級為高／中／低／無法判定。

首先，系統以提示詞（prompt）明確定義五個層面的評分方式與臨床權重，讓模型在生成回覆時能根據固定結構進行評估，此設計能確保評分標準一致，避免模型依語境出現不穩定的主觀判斷。

```
scoring_prompt = """
你是一位具臨床經驗的成癮輔導員，請根據使用者的描述，
以以下五個層面分別評估 0-4 分，並依照臨床邏輯推理。
評分時請同時考慮「臨床權重」的重要性：
- 失控程度（權重 1.6）：無法控制行為是成癮核心指標。
- 影響程度（權重 1.4）：若行為干擾生活、人際、健康，屬高危訊號。
- 情緒依附（權重 1.3）：強烈依附行為代表心理依賴風險。
- 強度（權重 1.1）：高使用量或時間過長會增加身體依賴風險。
- 頻率（權重 1.0）：頻繁使用雖不等於成癮，但為早期警訊。

加權總分計算方式（模型需自行推理）：
加權總分 = 頻率×1.0 + 強度×1.1 + 失控程度×1.6 + 影響程度×1.4 + 情緒依附
×1.3
並依以下區間判定風險：
    ≥14 → 高風險
    10-13.9 → 中風險
    <10 → 低風險
"""
```


在實作部分，系統會根據模型的回傳內容自動進行解析。若模型能正確輸出 JSON 格式，系統會直接讀取，若出現非結構化文字（如「頻率：3」），則透過正則表達式自動擷取分數。此外，若模型完全未提供數值，系統會自動以零補齊，避免程式中斷。

以下為加權總分的運算則根據各層面權重計算出整體風險，並在最後輸出對應等級。

```
# 計算加權總分
total = (
    scores[0]*1.0 + # 頻率
    scores[1]*1.1 + # 強度
    scores[2]*1.6 + # 失控程度
    scores[3]*1.4 + # 影響程度
    scores[4]*1.3   # 情緒依附
)

# 判定風險等級
if total >= 14:
    level = "高風險"
elif total >= 10:
    level = "中風險"
elif total > 0:
    level = "低風險"
else:
    level = "無法判定"

return {
    "頻率": scores[0],
    "強度": scores[1],
    "失控程度": scores[2],
    "影響程度": scores[3],
    "情緒依附": scores[4],
    "加權總分": round(total, 1),
    "風險等級": level
}
```

3.6 強化輸出結構與回饋內容

在初期版本中，系統的回覆多以自由生成的文字為主，雖然內容具分析性，但在表達上缺乏一致結構，導致不同回合之間的回應可讀性不一，甚至出現重複陳述或語意不連貫的情況。為了讓輸出內容更具條理與專業度，本研究針對系統的最終輸出層進行了重新設計，將模型生成的語意分析結果與系統自動計算的加權評分結果整合為單一報告格式，使其在資訊呈現上更直觀且具參考價值。

新的輸出機制將成癮風險評估結果與模型建議內容合併輸出，讓使用者在同一段落中即可同時看到數據分析結果與行為建議，提升資訊完整性與實用性。這樣的設計

不僅讓回覆內容結構化，也使系統更接近臨床報告格式，有助於未來延伸至健康風險預警或心理評估等應用場景。

在實作層面，系統會於模型生成回覆後，先呼叫加權評分函式計算各層面分數與風險等級，再將其整合至統一的輸出模板中，確保回應格式一致、資訊完整。

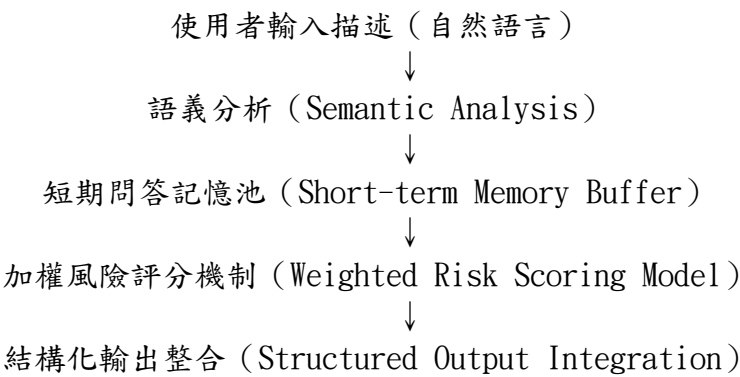
以下為實際輸出結構之程式碼：

```
reply = (  
    f"【成癮風險分析】\n"  
    f"頻率：{score_data['頻率']} 強度：{score_data['強度']} 失控程度：  
{score_data['失控程度']} "  
    f"影響程度：{score_data['影響程度']} 情緒依附：{score_data['情緒依附  
'}}\n"  
    f"加權總分：{score_data['加權總分']} → {score_data['風險等級  
'}}\n\n{answer}"  
)
```

Chapter4. 系統展示

4.1 系統架構

本學期的系統在原有 RAG 架構基礎上，加入語義分析、短期記憶與加權風險評分等模組，以提升對話連貫性與判斷準確度。整體運作流程如下：



系統會先透過語義分析模組解析輸入內容，主動辨識「頻率、強度、失控程度、影響程度、情緒依附」等關鍵層面，並以相似度比對方式確認描述中是否包含具體行為特徵。接著，短期問答記憶池會檢查使用者過去五輪的對話內容，若當前輸入與前文主題相關，系統會自動引用先前的互動紀錄，以維持語意連貫。

在判斷階段，系統透過加權風險評分模型計算出整體分數與風險等級，並將結果整合進最終回覆，使輸出內容同時具備量化結果與語意建議。此流程讓模型的回應更具一致性與臨床參考價值，並有效改善先前版本中回覆模糊與重複提問的問題。

4.2 系統回應展示

RAG	使用者
【成癮風險分析】 頻率:4 強度:3 失控程度:2 影響程度:2 情緒依附:0 加權總分:13.3→中風險 根據您的描述,我們可以初步判斷您可能存在成癮風險的傾向。您提到,每天都滑手機到半夜,但想睡時還是能放下。這表明您的滑手機行為可能有一定的強度和影響程度。	每天都滑手機到半夜,但想睡時還是能放下。

然而, 您也提到, 當您想睡時還是能放下。這意味著您可能對手機的攝入有一定的控制能力。	
【成癮風險分析】 頻率：4 強度：3 失控程度：4 影響程度：3 情緒依附：4 加權總分：23.1 → 高風險 根據用者的描述，我們可以看到有一個關於酒精和焦慮之間的連結。每天都要喝酒不然會感到焦慮，這個情況可能表明使用者存在一定的酒精依賴或成癮風險。尤其是當我們看到有高強度的傾向，這可能指向使用者對酒精有著非常強的情緒依賴。 但是，需要進一步了解的是什麼樣的情況下才能感到焦慮。是只喝一杯時感受到的焦慮，還是沒有喝酒就很難過？這個資訊對於評估成癮風險的程度很重要。 具體來說，我們需要更深入地了解以下幾點： 1. 你每天喝多少酒才會感到焦慮？ 2. 你覺得不喝酒會受到多大的影響，例如工作、社交、日常活動等方面？ 3. 你是否試過嘗試戒除或減少酒精使用？ 4. 你對酒精的依賴程度有多少？ 了解更多關於你的行為和感受，可以幫助我們進一步評估成癮風險和提供合適的建議。	每天都要飲酒，否則會感到焦慮。

4.3 系統回應錯誤與行為偏差分析

在成癮風險評估系統開發過程中，模型於多輪測試階段出現若干回應偏差與邏輯異常。為釐清問題來源與後續優化方向，將主要觀察結果歸納如下：

1. 評分邏輯偏差

模型對部分語意的數量詞與時間詞判斷不精確，例如「一天喝一瓶」的描述在語義分析中被誤判為中低頻率，導致分數低估。此問題反映出模型在中文日常語境下對

「頻率單位」與「量詞」的語意理解能力不足，造成加權總分無法準確反映實際行為強度。

2. 成癮風險分數不穩定

系統在多輪互動過程中出現「分數逐漸下降」的情形。推測主因為模型在後續回合重新評估時傾向保守化推論，加上未採取上次結果校正機制（last_score），導致風險評分逐輪減少。此現象影響了結果一致性與臨床可信度。

3. 結論生成機制失效

部分回覆雖已具備足夠資訊（例如高頻率且明顯影響生活），但模型仍重複產生「資訊不足」類回應，未進入最終結論階段。問題主因為 prompt 結構中未明確規定「風險判斷門檻」或終止條件，導致模型預設選擇安全回應路徑（持續補問）。

4. 反問與提問邏輯不自然

早期版本中，模型在資訊不足情況下的「補問」常出現語意偏差或語氣不合邏輯，例如對已明確陳述的行為再度追問「您是否有這樣的情況？」。此現象顯示模型未能正確辨識已知與未知資訊，導致提問冗餘、體驗不佳。後續版本已透過明確輸出模板及禁用提問關鍵詞（如「請問」「是否」「進一步了解」）加以改善。

4.4 系統優化策略

為改善上述狀況，本研究採取以下三項優化策略，使系統能更精準且穩定地輸出結論。

1. 雙層判斷機制（Scoring + Decision Layer）

系統首先透過 get_semantic_score() 函式進行五構面量化分析（頻率、強度、失控程度、影響程度、情緒依附），依臨床權重計算加權總分後自動判定風險等級（低至高），在此基礎上，新增第二層「決策邏輯（Decision Layer）」。

系統依據總分及關鍵構面組合自動分流為兩種回覆模式：

- A 模式：風險明確 → 直接產出【成癮風險等級】【最終結論】【建議行動】三段式回覆。
- B 模式：資訊不足 → 輸出【初步判斷】【可能原因或觀察】【建議後續觀察方向】三段式內容，不再提問。

此架構確保模型在資訊充足時能主動終止補問，輸出具體結論；而當資料不足時，仍能提供行為導向的觀察建議，避免對話陷入無限追問。

2. 分數穩定化機制（Score Stabilization）

為避免模型在多輪互動中出現評分偏移或過度保守，本系統採用三項穩定化設計：

1. 低溫度取樣（temperature=0.2）：減少隨機性，使同樣的描述維持一致評分。
2. 防漂移校正（last_score 機制）：若新一輪加權分顯著低於前次結果（低於80%），自動採用前次分數，避免系統誤降風險。此機制有效提升評分一致性與臨床判讀可靠度，使系統可於多輪對話中維持邏輯連貫。
3. 輸出格式結構化與防重複提問控制
過去模型常因 prompt 指令衝突，導致在給出結論後仍產生「若需進一步了解」等補問段落。本系統透過 輸出格式明確化（A/B 互斥邏輯）與 禁用提問關鍵詞（如「請問」「是否」「進一步了解」），讓模型在產出結論後不再主動追問，確保對話流程自然收斂，輸出內容一致且具可讀性。

4.5 系統優化成果

以下分別為資訊不足之情況與已明確可判斷風險之範例：

RAG	使用者
【成癮风险分析】 頻率：2 強度：1 失控程度：0 影響程度：0 情緒依附：0 加權總分：3.3 → 低風險 【初步判斷】 目前根據使用者描述，風險傾向為低風險。 【可能原因或觀察】 使用者描述表明只喝酒一次禮拜，這可能是因為對酒精的控制感覺良好，也可能是因為沒有感到酒精對自己的生理需求有強烈的依賴性。 【建議後續觀察方向】 建議觀察使用者的飲酒頻率和飲酒量是否會有任何變化，同時也建議使用者保持健康的生活習慣和飲食習慣，以維持身體和心理的健康。	我一個禮拜喝一次酒。

【成癮風險分析】 頻率：4 強度：3 失控程度：3 影響程度：4 情緒依附：4 加權總分：20.4 → 高風險 【成癮風險等級】 高風險 【最終結論】 使用者描述表明有明顯的成癮傾向，認為每天都要喝酒不然會睡不著，這表明酒精對使用者的生理需求有了強烈的依賴性。 【建議行動】 建議使用者儘快尋求專業的成癮治療和輔導，透過戒酒計畫和替代活動來減少酒精的使用，同時也建議使用者與家人或朋友溝通，共享自己的感受和需求，以獲得支持和鼓勵。	我每天都要喝酒，不然會無法入睡。
---	-------------------------

小結：經此優化後，系統能在單輪互動中完成風險評估、臨床分析與建議生成，達成「**自動化初步臨床判斷、避免無限追問、維持分數穩定**」三項目標。此設計兼顧了臨床邏輯嚴謹度與使用者體驗，使模型能以更接近專業輔導員的方式輸出成癮風險分析報告。

Chapter5. 結論

本研究以「運用 LLM 與 RAG 開發個人化成癮風險評估輔助系統」為核心，結合大型語言模型的語意理解與檢索增強生成架構，讓系統能從自然語言輸入中自動解析行為特徵，並以加權風險模型進行量化分析與臨床建議生成。

透過雙層判斷機制（Scoring + Decision Layer），系統能自動區分「風險明確」與「資訊不足」情境，在前者直接輸出結論與建議，後者則提供觀察方向，避免重複提問。搭配分數穩定化與短期記憶機制，有效提升了評估一致性與對話連貫性，使輸出內容更貼近臨床輔導員的分析邏輯。

實測結果顯示，系統可在單輪互動中完成行為解析與風險判定，成功改善早期版本中出現的重複提問、評分偏差與結論缺失問題，達成「自動化初步臨床判斷、穩定分數輸出、提升互動流暢度」三項主要目標。

此外，本研究針對模型的語意理解與記憶能力進行強化，透過 embedding 模型建立語義相似度判斷，讓系統能從自然語言中自動擷取行為層面的語義特徵，並結合短期問答記憶池避免重複提問。再配合加權風險評分機制，將原本主觀的語言輸出轉化為具臨床參考價值的量化結果，使模型在生成回覆時能同時呈現數據與分析邏輯，顯著提升輸出結構化程度與穩定性。

然而，目前模型在理解含模糊頻率詞的中文語句時仍可能出現低估，且 RAG 檢索速度受限於多模組運算，仍有進一步優化空間。

未來可從以下三個方向延伸研究：

1. **語意理解強化：** 針對繁體中文語料進行微調，提升模型對日常語句與數量詞的理解準確度。
2. **長期行為追蹤模組：** 將使用者多次評估結果時序化，建立行為變化趨勢，以支援健康風險預測。
3. **臨床應用與實測整合：** 與心理輔導或保險產業合作，進行臨床實驗與場域驗證，評估實際使用成效。

綜上所述，本研究透過生成式 AI 技術，建立起能理解語意、分析行為並提供建議的成癮風險評估系統。這套系統讓風險評估變得更即時、更貼近使用者的實際情況，也展現了 AI 在健康輔助領域中的應用潛力。

Reference

<https://solwen.ai/posts/what-is-rag>

訓練文件：

吸菸成癮：<https://hdl.handle.net/11296/2td3a4>

購物成癮：<https://hdl.handle.net/11296/g6vjsx>

藥物成癮：<https://hdl.handle.net/11296/2cz5sd>

咖啡因成癮：

<https://archpublichealth.biomedcentral.com/articles/10.1186/s13690-024-01447-6>

酒精成癮：<https://hdl.handle.net/11296/2b5c96>

數位成癮：<https://hdl.handle.net/11296/9jw44x>

附錄 A. 實習工作內容

➤ 胡家璿

我在精誠資訊進行專業實習，主要參與保險系統專案的功能開發與維護工作。上學期以熟悉企業內部開發環境與架構為主，主要負責現有模組的維護與功能調整，學習專案流程、版本控管與資料處理邏輯，逐步建立系統開發的基礎。進入本學期後，工作內容轉向獨立開發新功能模組，從零開始進行前後端整合、資料流程設計與 API 串接，實際負責完整系統的開發與上線，累積了更深入的專案經驗與問題解決能力。

工作內容包括：

- **系統開發與功能實作：**開發「保險商品配置系統」與「保單資料異動作業」等主要模組，實作商品配置推薦計算、推薦查詢、推薦確認進度及資料搬移功能，建構完整交易流程並確保資料正確性與流程穩定。
- **前後端整合：**使用 HTML、CSS、JavaScript、C# 與 SQL 進行介面開發與邏輯處理，完成資料驗證、查詢、更新與刪除等 CRUD 功能，優化畫面互動體驗與系統效能。
- **API 開發與串接：**依據規格書開發多支後端 API，負責取得保險商品配置回傳資訊、處理外部資料回傳與入庫邏輯，確保跨系統資料交換正確且流程順暢。
- **RDL 報表呈現：**使用 RDL 建構具邏輯控制的視覺化網頁內容，新增進件欄位並實作提領、調降保額等動態邏輯，整合資料呈現與互動流程，強化系統的視覺一致性與操作性。

➤ 傅詩婷

我於精誠集團－宜誠資訊進行專業實習，參與企業內部系統的開發與維護工作，負責從前端介面到後端資料庫的完整建置，採用 MVC.NET 架構進行開發。實作中運用 HTML、Bootstrap 與 JavaScript 建置互動式網頁，透過 C# 控制器串接資料庫，執行 SQL 預存程序，以提升系統效能並降低查詢延遲，確保功能與使用者需求的一致性。

工作內容包括：

- **網頁開發：**以 .NET MVC 架構開發，使用 HTML 與 Bootstrap 製作響應式網頁介面，並透過 JavaScript 及 jQuery 強化使用者互動，由 C# 控制器負責前後端資料串接。

- **資料庫設計與預存程序應用：**負責建立與維護 SQL 資料表結構，撰寫多個 Stored Procedure 以處理資料的新增、查詢、修改與刪除。透過 JavaScript 呼叫程序並即時回傳前端，實現頁面互動與資料即時更新，有效提升資料處理效率與安全性。
- **系統規格書學習與理解：**在實習期間首次接觸系統規格書，學習如何從中解讀功能需求與系統邏輯，培養開發前分析與規劃的能力。
- **AI 工具應用探索：**實地測試並評估 AI 工具在程式開發、除錯與資料分析中的應用，發現其在縮短開發時間、提升準確率與促進團隊協作方面具有明顯效益，進一步優化整體開發流程與效率。

附錄 B. 實習心得與建議

➤ 胡家璿

經過一學期的實習，使我對整個專案開發流程更加熟悉。無論是前後端的邏輯銜接、API 設計、資料處理，或是版本控制與測試流程，都比起剛開始時有更深入的理解。

在實際的開發任務中，我也逐漸學會如何讓程式結構更清晰、命名更一致，並注意到團隊協作時維護性的重要性。

比起上學期主要在熟悉環境與既有系統，這學期更有機會獨立開發新功能、處理實際需求，過程中不僅訓練了問題分析與解決能力，也讓我更理解團隊協作在專案開發中的價值。

整體而言這段實習經驗讓我從原本的單純完成任務，變得能主動思考架構，並溝通需求、預先規劃，讓我對未來職涯的方向更具信心。

目前公司在實習安排與專案協作上都相當完善，整體流程清楚、工作內容也具挑戰性。

如果真的要提出建議，也許可以在任務初期多提供一點整體系統的脈絡說明，讓新進實習生更快了解功能的上下關聯。除此之外，整體環境和指導都讓我收穫很多，沒有其他特別的意見。

➤ 傅詩婷

實習心得：

在大三時，我進入精誠集團實習，在真實企業環境中參與系統開發專案。實習初期，我主要負責前端介面設計與功能優化，學習如何運用 HTML、CSS、JavaScript 結合 Bootstrap 進行響應式設計，確保系統在不同裝置上皆能維

持良好體驗。同時，我也協助後端資料庫連接與 API 整合，理解了資料在前後端之間傳遞與驗證的流程，進一步提升了程式邏輯與整體架構思維。

在中後期，我嘗試建立自動化測試流程，運用 PowerShell 搭配 .NET 測試框架，使系統能自動產出測試報告與截圖，大幅提升測試效率與準確性。此外，我也撰寫操作手冊、系統分析書與測試報告，學習如何以文件形式清楚呈現系統設計邏輯與使用步驟，這讓我深刻體會到文件化管理對專案協作與後續維護的重要性。

這段實習不僅讓我將課堂所學實際應用於企業系統中，更訓練了我在專案中主動解決問題與與團隊溝通的能力。透過與工程師及主管的交流，我學會在開發中平衡效率與品質，並理解企業在導入新系統時面臨的挑戰。

整體而言，精誠集團在實習安排與指導上都相當完善，讓我能在短時間內學習到實務開發經驗。若要提出建議，我認為可以在專案初期提供更多技術文件或系統架構說明，讓實習生更快熟悉整體流程。此外，若能安排跨部門的簡短分享或工作觀摩，將有助於實習生更全面了解企業運作與專案協作方式。