

國立臺北商業大學

資訊管理系

112，資訊系統專案設計

系統手冊



組 別：第 112208 組

題 目：A movie I recommend

指導老師：許晉龍老師

組 長：11136025 馬振閔

組 員：11136020 王元平

11136033 伍以晨

中 華 民 國 1 1 2 年 1 1 月 2 2 日

目錄

第一章 前言	1
1-1 背景介紹	1
1-2 動機	2
1-3 系統目的與目標	3
1-4 預期成果	4
第二章 營運計畫	5
2-1 可行性分析	5
2-2 商業模式－BUSINESS MODEL	6
2-3 市場分析－STP	7
2-4 競爭力分析 SWOT-TOWS 或五力分析	8
第三章 系統規格	12
3-1 系統架構	12
3-2 系統軟、硬體需求與技術平台	13
3-3 使用標準與工具	13
第四章 專案時程與組織分工	15
4-1 專案時程	15
4-2 專案組織與分工	16

第五章 需求模型	18
5-1 使用者需求	18
5-2 使用個案圖	19
5-3 使用個案描述	20
5-4 分析類別圖	22
第六章 設計模型	23
6-1 循序圖	23
6-2 設計類別圖	28
第七章 實作模型	29
7-1 佈署圖	29
7-2 套件圖	29
7-3 元件圖	30
7-4 狀態機	31
第八章 資料庫設計	33
8-1 資料庫關聯表	33
8-2 表格及其 META DATA	34
第九章 程式	36
9-1 元件清單及其規格描述	36
9-2 其他附屬之各種元件	59

第十章 測試模型	70
10-1 測試計畫	70
10-2 測試個案與測試結果資料	72
第十一章 操作手冊	79
介紹系統之元件及其安裝及系統管理	79
第十二章 使用手冊	81
第十三章 感想	87
第十四章 參考資料	90
附錄	91

圖目錄

1-1-1 系統 LOGO 圖	1
3-1-1 系統架構圖	12
4-1-1 甘特圖	15
5-2-1 使用個案圖	19
5-3-1 登入活動圖	20
5-3-2 調查活動圖	20
5-3-3 評論活動圖	21
5-4-1 分析類別圖	22
6-1-1 登入循序圖	23
6-1-2 忘記密碼循序圖	23
6-1-3 調查循序圖	24
6-1-4 電影介紹循序圖	24
6-1-5 電影評分循序圖	25
6-1-6 搜尋循序圖	25
6-1-7 收藏片單循序圖	26
6-1-8 修改收藏片單循序圖	26
6-1-9 推薦電影循序圖	27
6-2-1 設計類別圖	28
7-1-1 佈署圖	29
7-2-1 套件圖	29

7-3-1 元件圖	30
7-4-1 註冊狀態機	31
7-4-2 登入狀態機	31
7-4-3 留言評論狀態機	31
7-4-4 修改密碼狀態機	32
7-4-5 變更大頭貼狀態機	32
7-4-6 登出狀態機	32
8-1-1 資料庫關聯圖	33

表目錄

2-4-1 SWOT 分析表	8
3-2-1 軟硬體需求表	13
3-3-1 系統開發環境表	13
3-3-2 程式開發技術表	13
3-3-3 資料庫管理	14
3-3-4 版本控制管理工具	14
3-3-5 設計工具	14
4-1-1 甘特圖	15
4-2-1 分工表	16
4-2-2 專題成果工作內容與貢獻度表	17
8-2-1 使用者興趣 USERINTERESTS 資料表	34
8-2-2 使用者資料 USERS 資料表	34
8-2-3 電影評論 MOVIE_COMMENT 資料表	34
8-2-4 類型 CATEGORY 資料表	35
8-2-5 電影清單 MOVIELIST 資料表	35
8-2-6 電影星等 MOVIE_SCORE 資料表	35
9-1-1 KOTLIN 元件清單	36
9-1-2 規格功能描述-LOGINACTIVITY	37
9-1-3 規格功能描述-MAINACTIVITY	40

9-1-4 規格功能描述- REGISTERACTIVITY.....	42
9-1-5 規格功能描述- CHOOSEACTIVITY	44
9-1-6 規格功能描述- SCOREACTIVITY.....	48
9-1-7 規格功能描述- FRAGMENT_USER.....	49
9-1-8 規格功能描述- FRAGMENT_HOME.....	52
9-1-9 規格功能描述- APISERVICE	53
9-1-10 PYTHON 元件清單	54
9-1-11 規格功能描述-電影評論_BERT-正負評	55
9-1-12 規格功能描述- CONTENT-BASED RECOMMENDATION.....	57
9-2-1 XML 規格功能描述表	59
9-2-2 規格功能描述- ACTIVITY_LOGIN.XML	60
9-2-3 規格功能描述- ACTIVITY_CHOOSE.XML	63
9-2-4 規格功能描述-FRAGMENT_USER.XML	68
10-2-1 註冊會員測試個案與測試結果資料	72
10-2-2 登入會員測試個案與測試結果資料	72
10-2-3 興趣選擇測試個案與測試結果資料	73
10-2-4 主頁功能測試個案與測試結果資料	73
10-2-5 下方功能列測試個案與測試結果資料	74
10-2-6 電影資訊測試個案與測試結果資料	74
10-2-7 留言按鈕測試個案與測試結果資料	75
10-2-8 留言測試個案與測試結果資料	75

10-2-9 我的個人檔案頁面測試個案與測試結果資料	76
10-2-10 編輯個人檔案測試個案與測試結果資料	76
10-2-11 修改密碼測試個案與測試結果資料	77
10-2-12 更換大頭貼測試個案與測試結果資料	77
10-2-13 登出測試個案與測試結果資料	78
11-1 系統安裝元件資訊表	79
11-2 系統安裝步驟	79
12-1 系統使用說明-登入與註冊	81
12-2 系統使用說明-興趣選擇	83
12-3 系統使用說明-觀看電影資訊並評分留言	84
12-4 系統使用說明-個人資料修改	85
12-5 系統使用說明-登出	86
附錄-初評評審意見之修正情形	91
附錄-複評評審意見之修正情形	92
附錄-海報評審意見	92

第一章 前言

1-1 背景介紹

在線上影音平台普及的現在，人們對於電影消費需求的不斷增加，選擇一部好電影也變成許多人的困擾，在大數據和人工智能等技術的發展，個性化推薦系統的應用已經成為了一個重要趨勢。而在這個趨勢下，通過我們的 APP 系統作為一種推薦方式，具有相當高的價值，隨著技術的不斷發展，我們相信這個系統的前景將會更加廣闊，為人們的生活帶來更多便利和樂趣。



▲圖 1-1-1 系統 logo 圖

1-2 動機

1. 提供個人化的電影推薦：

幫助用戶更容易找到他們可能喜歡的電影。這可以提高觀眾的滿意度，增加他們對 App 的使用和忠誠度。

2. 節省時間：

觀眾經常需要花費時間和努力尋找新電影觀看。一個自動推薦系統可以幫助他們迅速找到符合他們品味的電影，節省時間。

3. 支持電影產業：

這樣的 App 有助於推廣電影，增加票房收入，並為製片公司和影片流媒體平台提供更多機會來推廣他們的作品。

4. 電影產業的市場分析和改進：

收集用戶評分和觀看數據，可以提供有關觀眾偏好的有價值信息，數據分析和市場研究有助於電影產業的市場分析和改進。

1-3 系統目的與目標

系統目的：

1. 提供個人化的電影推薦：

許多人喜歡觀看電影，但由於電影種類繁多，有時候難以選擇。一個自動推薦系統可以根據使用者的喜好和觀看歷史，提供個人化的電影建議，幫助他們更輕鬆地找到感興趣的影片。

2. 增加用戶參與度：

一個好的電影推薦系統可以使 App 更具吸引力，並增加用戶的參與度。用戶可能會花更多時間在 App 上，尋找新的電影，評論和分享他們的觀看經驗。

3. 促進電影推廣：

製片公司和影片流媒體平台可能希望通過這種 App 來推廣他們的電影，提高曝光度和觀看率。這樣的 App 可以幫助他們更好地推廣新作品，並吸引更多的觀眾。

4. 收集用戶數據：

這種 App 可以收集用戶的觀看偏好和行為數據，這些數據對於市場分析和改進內容推薦算法非常有價值。這些數據可以用於更好地了解觀眾，並優化內容推薦。

5. 創造商業機會：

這樣的 App 可以成為一個潛在的商業機會，通過廣告、訂閱模型、合作夥伴推廣等方式來盈利。

系統目標：

1. 提供個性化推薦：

開發一個強大的推薦算法，能夠根據用戶的觀看歷史、評分、喜好等因素，為他們推薦適合的電影。

2. 提高用戶參與度：

開發吸引人的功能，例如用戶評論、社交分享、用戶生成內容等，以增加用戶參與度。

3. 提供多平台支援：

確保 App 可以適用於不同的操作系統和設備，以擴大受眾。

4. 增加電影曝光度：

合作製片公司和影片流媒體平台，提供合作和宣傳機會，以增加電影的曝光度和觀看率。

5. 保護用戶隱私：

確保用戶數據的隱私和安全，遵守相關的數據保護法律和規定。

6. 實現盈利能力：

開發可持續的盈利模型，例如廣告收入、訂閱模型、付費功能等，以維持 App 的運營和發展。

1-4 預期成果

希望通過我們的系統，提高使用者的觀影體驗，也讓使用者更容易找到符合自己喜好的電影，同時也能夠不斷改進推薦算法，提高推薦結果的準確性和效果。

第二章 營運計畫

2-1 可行性分析

1. 技術可行性：

實現電影推薦系統需要先進的技術支持，例如機器學習、自然語言處理、推薦算法等。目前，這些技術已經得到了很好的發展，有許多的開源庫和框架可以供開發人員使用，因此在技術方面，這個專案是可行的。

2. 市場需求：

隨著人們生活水平的提高，觀影的需求也在不斷增加，電影推薦系統的市場需求也因此不斷增加。而且，這個專案可以利用網絡平臺，快速擴展用戶基礎，吸引更多的潛在用戶，這意味著這個專案在市場上有很好的前景。

3. 競爭分析：

在這個領域，已經有許多的電影推薦系統，例如 Netflix、IMDb 等。因此，要想在市場上脫穎而出，需要獨特的技術支持、優良的用戶體驗以及積極的市場推廣等。通過不斷地改進和完善，這個專案也有機會與其他競爭對手區分開來。

2-2 商業模式

1. 廣告收入：

提供免費使用的 App，通過顯示廣告來賺取收入。廣告可以在 App 的界面中顯示，也可以在觀看電影時插播。廣告商可以根據用戶數據進行定向廣告，提高廣告效益。

2. 合作夥伴推廣：

合作製片公司和影片流媒體平台，為他們的新電影提供宣傳和推廣服務。為成功的合作推廣活動收取費用。

3. 數據銷售和分析：

將收集的用戶數據出售給市場研究公司、製片公司或廣告商，以進行市場分析和廣告定向。

4. 贊助和贊助商合作：

與贊助商合作，為他們的產品或服務提供曝光機會，並收取贊助費用。

2-3 市場分析

1. 競爭環境：

在台灣市場上，已經有許多的電影推薦服務，如觀眾電影推薦系統、愛奇藝電影推薦系統、Netflix 等。這些競爭對手在市場上具有一定的知名度和品牌優勢，對於新進入市場的專案會帶來挑戰。

2. 目標用戶：

這個專案的目標用戶是那些喜愛觀看電影的人群，特別是年輕人和城市居民。這些人對於新興科技和數字化產品的接受度較高，並且樂於在網路上尋找自己喜愛的電影。

3. 相關政策：

在台灣市場，數據隱私保護和數位安全等政策非常重要。因此，對於這個專案而言，需要了解相關政策法規，並且制定適合的隱私保護措施，保障用戶的數據安全。

2-4 競爭力分析

1. 競爭對手分析：

● Netflix：

作為一個領先的在線視頻平臺，Netflix 已經建立了強大的電影推薦系統，並且擁有龐大的用戶基礎和品牌效應，是這個專案的強大競爭對手。

● IMDb：

作為一個致力於電影、電視節目和名人的信息平臺，IMDb 擁有豐富的電影數據庫和評論系統，是這個專案的競爭對手之一。

● 其他同類型的推薦系統：

除了 Netflix 和 IMDb 之外，還有許多類似的電影推薦系統，例如 Amazon Prime Video、Hulu、Rotten Tomatoes 等。

2. SWOT 分析：

▼表 2-4-1 SWOT 分析表

優勢 (Strengths)	劣勢 (Weaknesses)
個性化推薦 良好的用戶體驗 廣泛的電影數據庫	新進入市場 需要豐富的數據支持 機器學習準確度
機會 (Opportunities)	威脅 (Threats)
與製片公司合作 提供社交功能	競爭激烈 版權問題

- 優勢：

- (1) . 個性化推薦：

- 該 APP 能夠通過分析使用者的觀看歷史、評分和喜好等數據，為用戶提供個性化的影片推薦，提高用戶體驗和滿意度。

- (2) . 良好的用戶體驗：

- 這個 APP 的 UI 界面設計簡單易用，操作流暢，用戶體驗良好。

- (3) . 廣泛的電影數據庫：

- 我們可以集成廣泛的電影數據庫，從而能夠為用戶提供更多的選擇。

- 劣勢：

- (1) . 新進入市場：

- 我們是一個新進入市場的產品，需要時間來建立品牌知名度和用戶基礎。

- (2) . 需要豐富的數據支持：

- 這個專案需要豐富的電影數據來支持推薦算法的準確性，需要不斷地收集和整理數據。

- (3) . 機器學習準確度：

- 該 APP 的個性化推薦演算法是基於機器學習實現的，因此，其準確度可能會受到使用者行為的變化和數據的不完整性等因素的影響。

- 機會：

- (1) . 與製片公司合作：

- 可以與製片公司合作，通過獨家發行權來提供獨特的影片內容，吸引更多用戶和提高盈利能力。

- (2) . 提供社交功能：

- 可以增加社交功能，使用戶可以與其他用戶互動和分享影片

- 威脅：

- (1) . 競爭激烈：

- 該 APP 所在的市場競爭激烈，存在許多其他影片推薦 APP 的競爭對手，需要通過優化算法、提高用戶體驗和提供多樣化的影片內容來保持市場競爭力。

- (2) . 版權問題：

- 如果遇到版權問題，該 APP 的內容庫將受到限制，進而影響用戶體驗和市場競爭力。

以下是經過 SWOT 分析後運用 TOWS 產生的解決方案：

- S-O 策略：

利用獨特的推薦算法和良好的用戶體驗，與製片公司合作，增加影片內容的多樣性，提高推薦算法的準確度和用戶滿意度。

- W-O 策略：

通過豐富的數據支持和機器學習技術，進一步優化推薦算法和提供更多樣化的影片內容，以提高用戶滿意度和市場競爭力。

- S-T 策略：

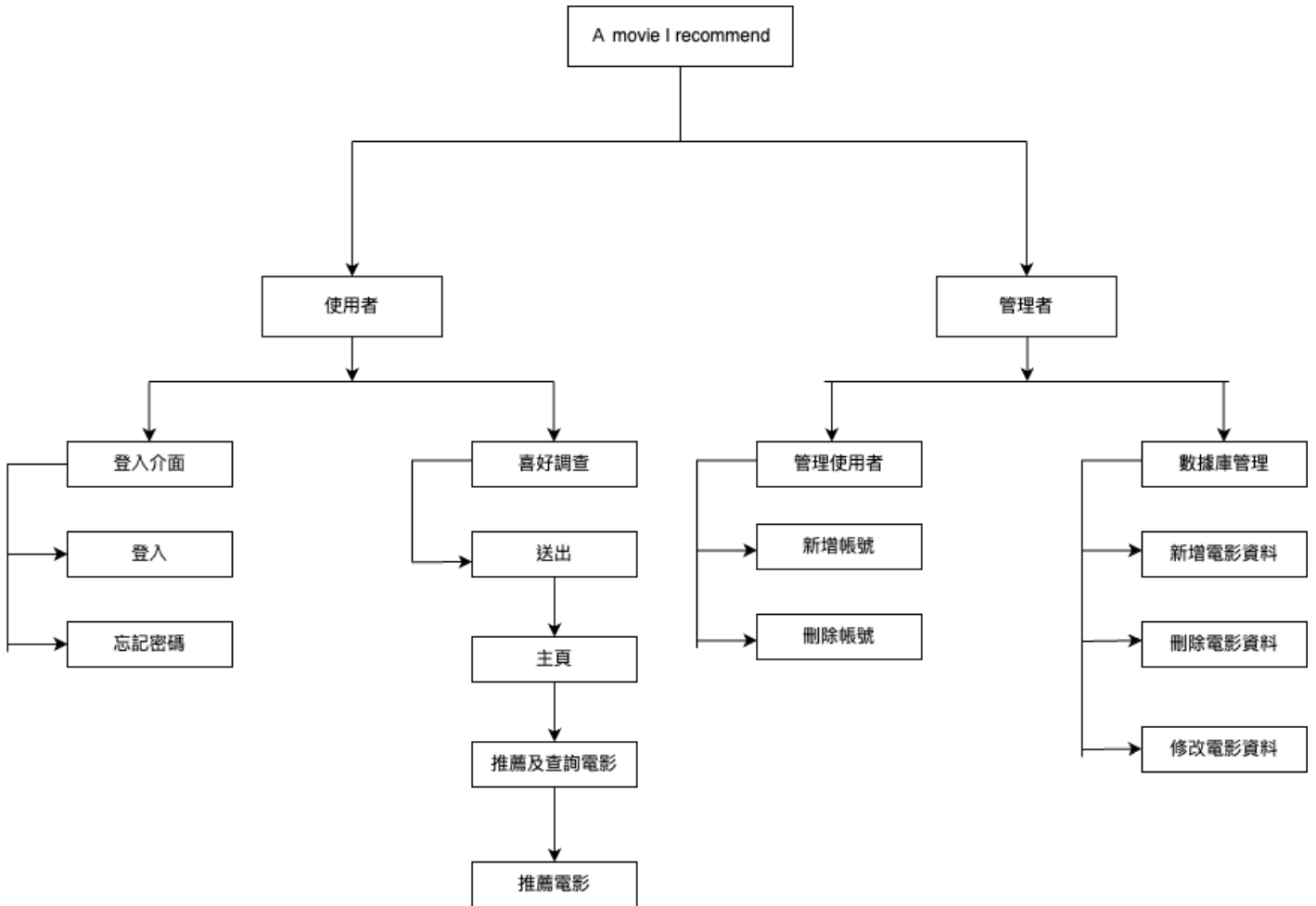
加強與用戶的互動，提供社交功能，擴大用戶群體，提高市場競爭力和品牌知名度。

- W-T 策略：

積極處理版權問題，加強與相關監管機構的合作，提高市場信譽度。

第三章 系統規格

3-1 系統架構



▲圖 3-1-1 系統架構圖

3-2 系統軟、硬體需求與技術平台

▼表 3-2-1 軟硬體需求表

軟硬體需求	
作業系統版本	Android 10 +
行動需求	
網路需求	Wi-Fi 無線網路

3-3 使用標準與工具

▼表 3-3-1 系統開發環境表

系統開發環境	
作業系統	Windows、Mac os、學校虛擬機
資料庫伺服器	Sqlite3.44.0
伺服器	學校系上虛擬機

▼表 3-3-2 程式開發技術表

程式開發技術	
前端整合開發環境	Android Studio
程式語言	Kotlin、python
Web 框架	Flask
遠端伺服器	Windows Servar
資料庫	Sqlite

▼表 3-3-3 資料庫管理

版本控制管理工具	
資料庫	Sqlite
資料庫管理介面	Visual Studio Code

▼表 3-3-4 版本控制管理工具

版本控制管理工具	
版本控制	GitHub
專案管理	Fork

▼表 3-3-5 設計工具

設計工具	
文件製作	Microsoft Office Word 2019
簡報製作	Canva
UML 製作	Drawio
LOGO 設計	Canva
海報製作	Canva
影片製作	imovie

第四章 專案時程與組織分工

4-1 專案時程

工作項目	111 年		112 年											
	11 月	12 月	1 月	2 月	3 月	4 月	5 月	6 月	7 月	8 月	9 月	10 月	11 月	12 月
確認專題題目														
可行性評估														
功能分析														
系統需求分析														
設計 LOGO														
設計 UI														
製作文件														
程式撰寫														
設計資料庫														
系統整合														
系統測試														
簡報製作														
海報製作														
影片製作														
													預計進度	
													實際進度	

▲圖 4-1-1 甘特圖

4-2 專案組織與分工

● 主要負責人 ● 協助負責人

▼表 4-2-1 分工表

項目/組員		11136025 馬振閔	11136020 王元平	11136033 伍以晨
開發 後端	資料庫建置	●	○	○
	伺服器架設	●	○	○
	網路通訊服務	●	○	○
前端 開發	登入介面	○	●	○
	調查介面	○	●	○
	主頁介面	○	●	○
	下方工具列	●	●	○
	收藏片單介面	○	○	●
	電影簡介介面	○	○	●
	電影評論介面	○	○	●
	帳號管理介面	○	○	●
美術 設計	UI/ UX	○	●	○
	Web/APP 介面設計	●	○	○
	色彩設計	○	○	●
	Logo 設計	○	●	○
	素材設計	○	●	○
文件 撰寫	統整	○	●	○
	第 1 章 前言	○	○	●
	第 2 章 營運計畫	○	●	○
	第 3 章 系統規格	○	○	●
	第 4 章 專題時程與組織分工	○	○	●
	第 5 章 需求模型	○	●	○
	第 6 章 設計模型	○	○	●
	第 7 章 實作模型	●	○	○
	第 8 章 資料庫設計	●	○	○
	第 9 章 程式	●	○	○
	第 10 章 測試模型	●	○	○
	第 11 章 操作手冊	○	○	●
	第 12 章 使用手冊	○	○	●
報告	簡報製作	○	●	○
	海報製作	○	●	○
	影片製作	○	●	○

▼表 4-2-2 專題成果工作內容與貢獻度表

序號	姓名	工作內容<各限 100 字以內>	貢獻度
1	組長 <u>馬振閔</u>	各方面都有深度參與，且是本組組長，每周都有開會討論，確保專題順利進行。 從前端設計到後端資料庫建置及模型都擔任重責大任，花了很多功夫及時間研究程式和完成系統，主要負責後端系統程式撰寫及整體系統整合。 每週參與小組討論，並在協調工作進度和解決問題的過程中展現團隊協作的精神	<u>35</u> %
2	組員 <u>王元平</u>	雖然技術水平程度上在組內較弱，但還是能努力完成分配之工作，主要負責系統的前端介面設計和功能撰寫，UML 及文書、海報、影片撰寫，後端則是積極跟隨組長學習程式，努力提升自己的技術水平。 每週參與小組討論，並在協調工作進度和解決問題的過程中展現團隊協作的精神。	<u>30</u> %
3	組員 <u>伍以晨</u>	主要負責前端介面的設計和功能的流暢，以及使用者互動體驗的優化，同時確保前後端的順暢溝通。每週參與小組討論，協調大家的工作進度，並即時解決可能發生的問題，以確保整體專案進展順利。花費相當的心力於前端技術的研究與應用，並專注於提升使用者體驗和界面設計，使系統更加直觀且具有吸引力，同時也會幫忙程度較弱的組員，一起努力完成任務。	<u>35</u> %
			總計:100%

第五章 需求模型

5-1 使用者需求

1. 功能性需求：

(1) . 一般使用者：

- 用戶相關：針對使用者個人登入、忘記密碼。
- 調查相關：調查使用者有興趣之電影題材。
- 搜尋相關：搜尋 App 上的電影進行查看評價。
- 評論相關：對 App 上的電影進行評分與留言評論。

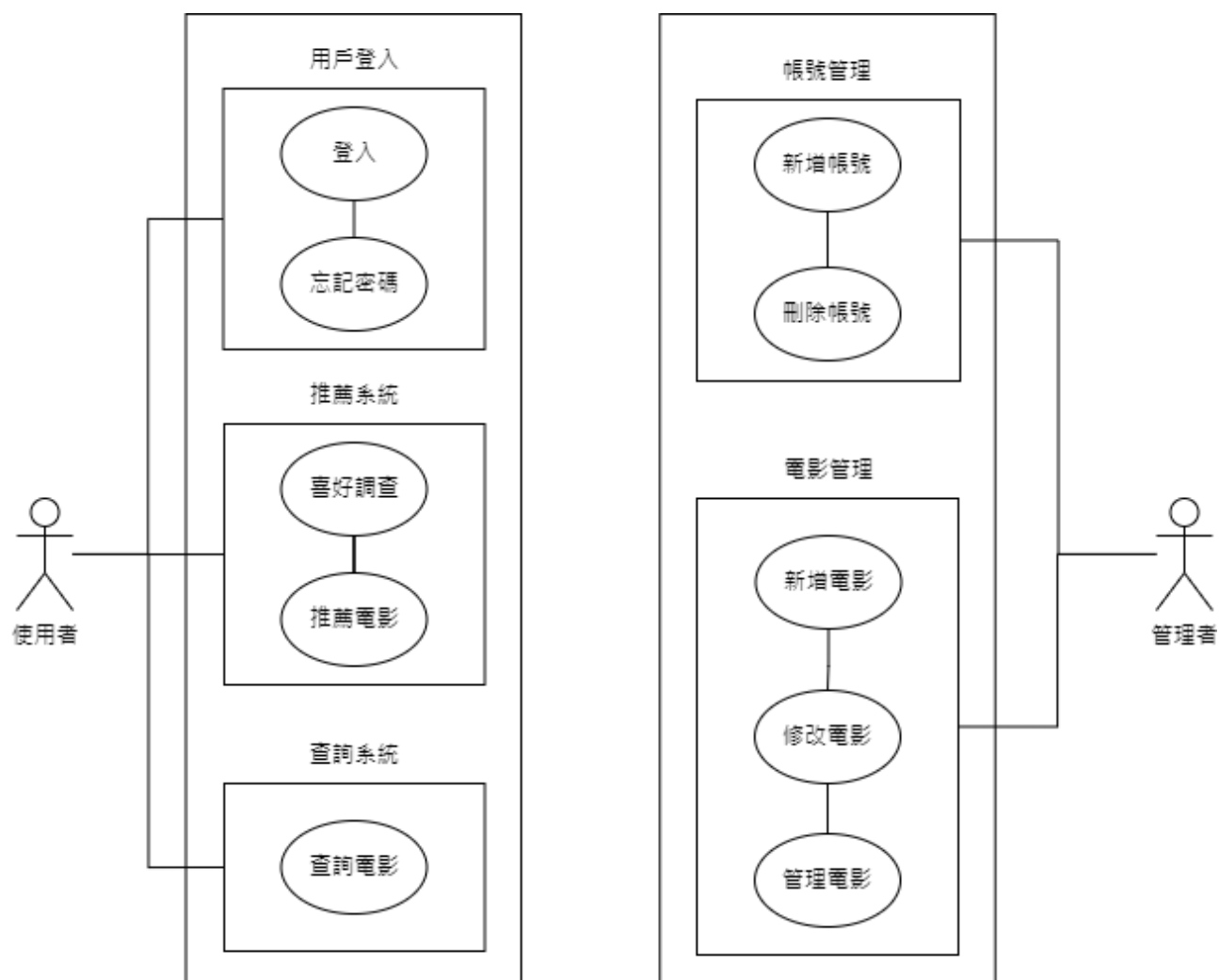
(2) . 管理者：

- 用戶相關：針對新增刪除使用者帳號。
- 電影相關：對電影簡介進行新增、刪除、修改。

2. 非功能性需求：

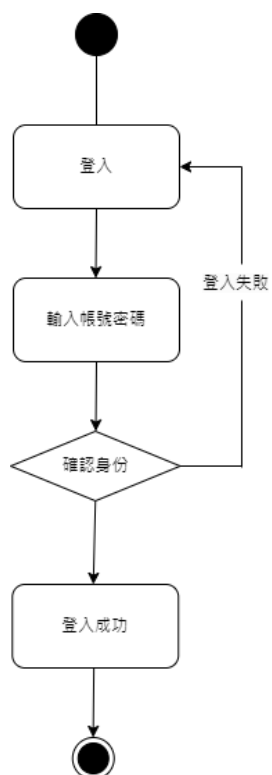
裝置需求：Android 最低需 10.0 版，且手機須能上網(行動網路、Wi-Fi)。

5-2 使用個案圖(Use case diagram)。

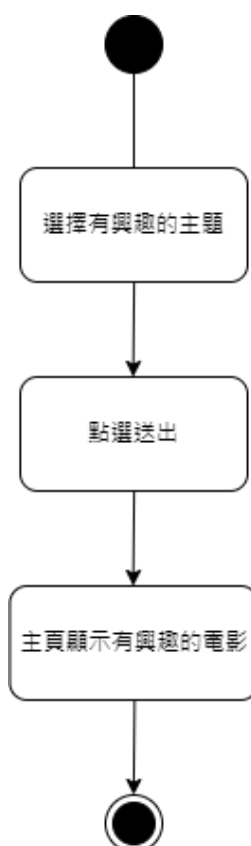


▲圖 5-2-1 使用個案圖

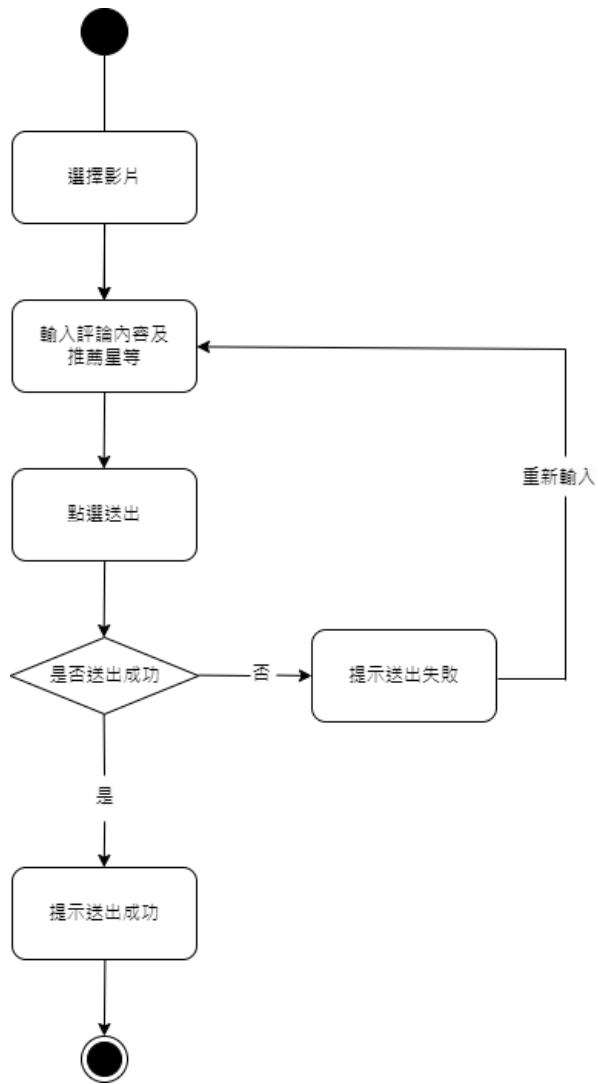
5-3 使用個案描述(Activity diagram)



▲圖 5-3-1 登入活動圖

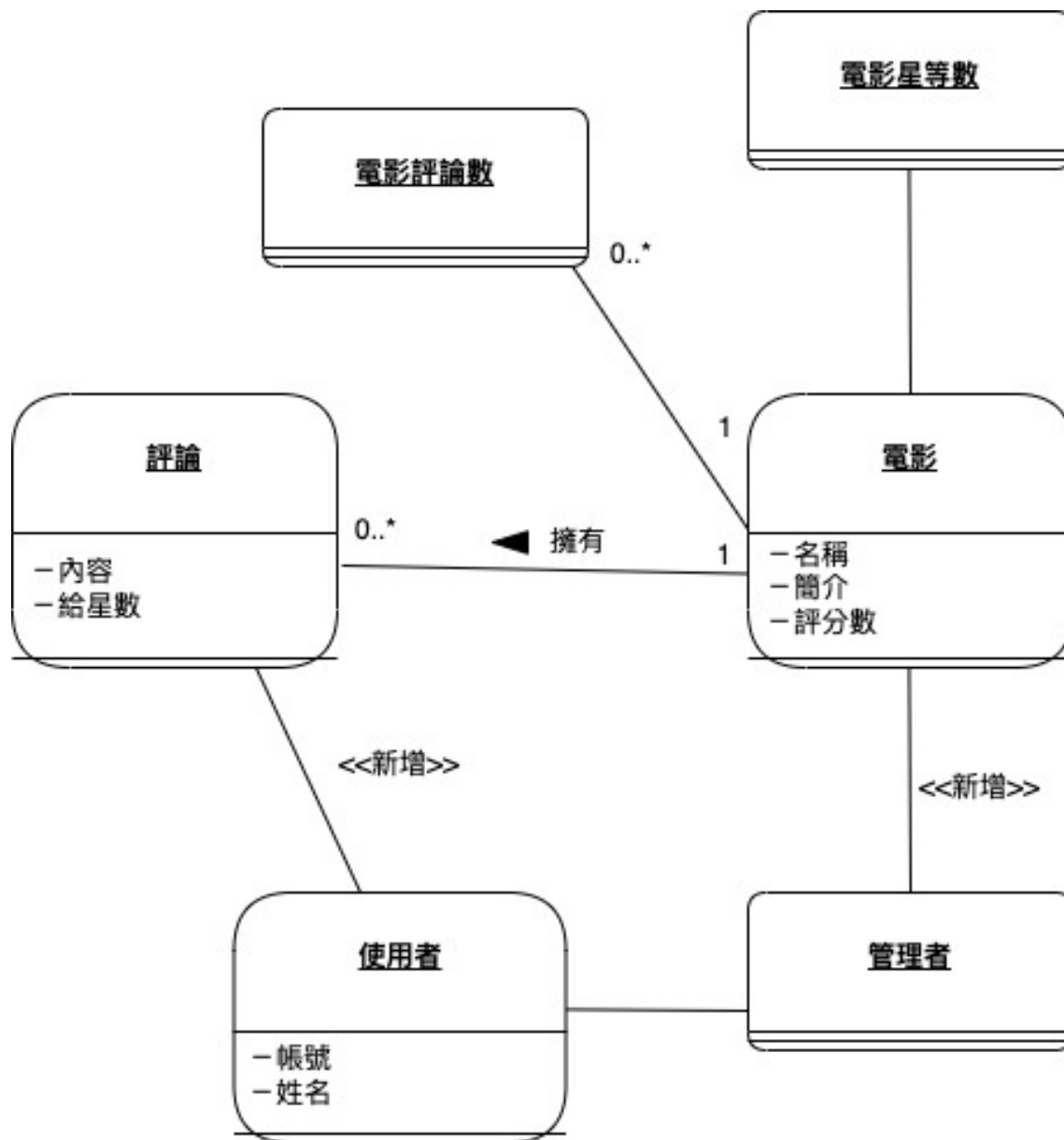


▲圖 5-3-2 調查活動圖



▲圖 5-3-3 評論活動圖

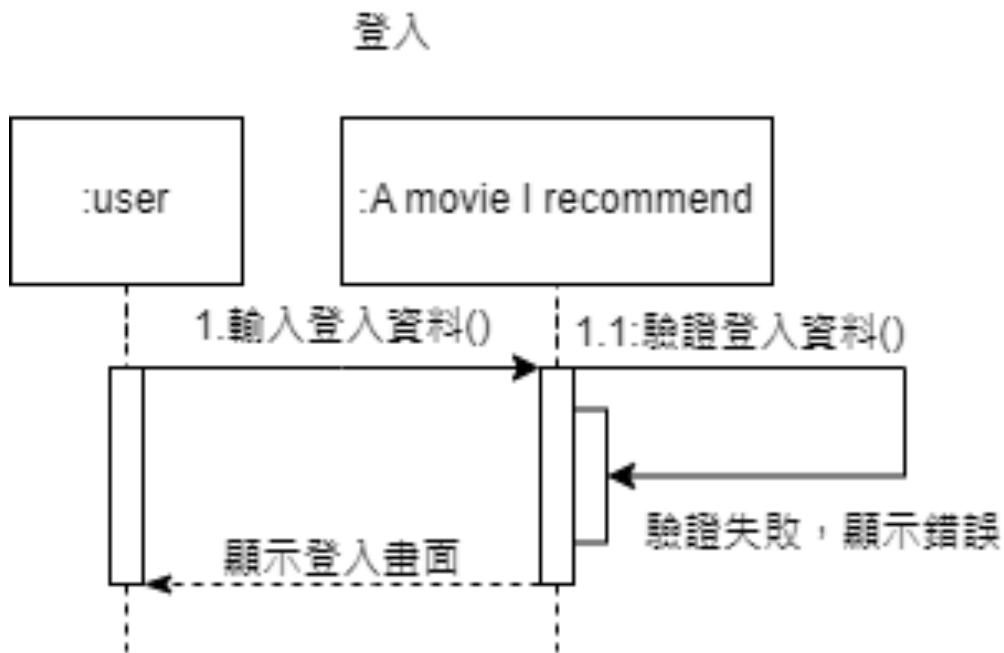
5-4 分析類別圖(Analysis class diagram)



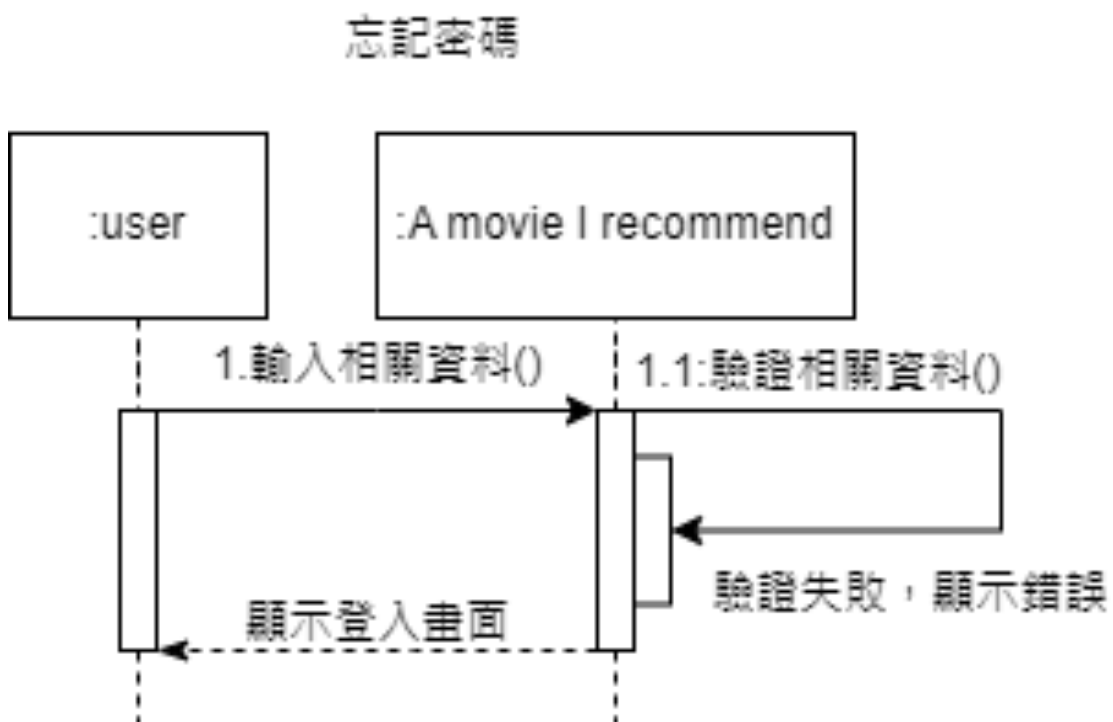
▲圖 5-4-1 分析類別圖

第六章 設計模型

6-1 循序圖(Sequential diagram)

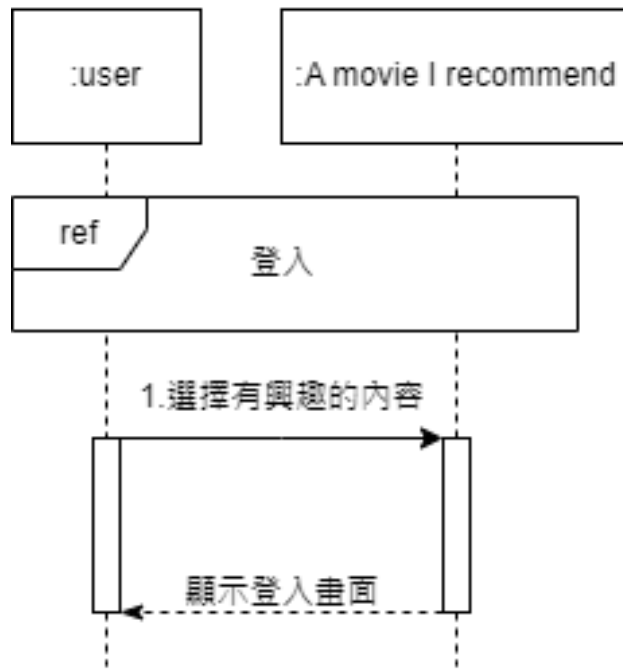


▲圖 6-1-1 登入循序圖



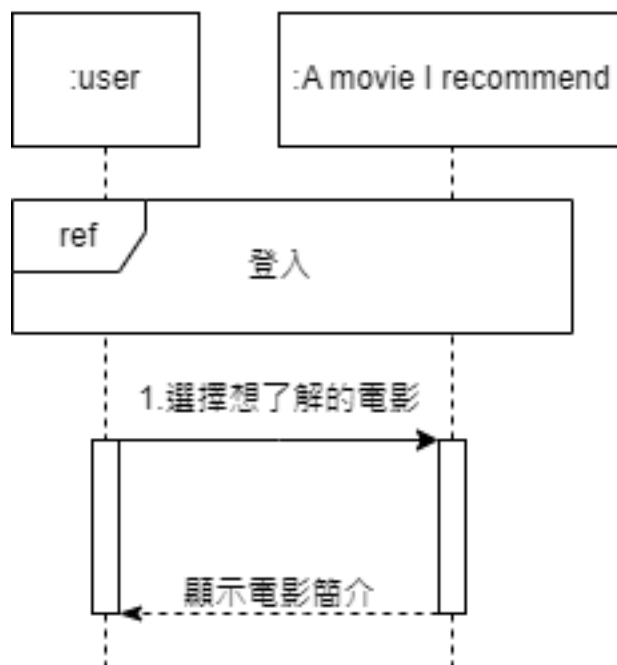
▲圖 6-1-2 忘記密碼循序圖

調查



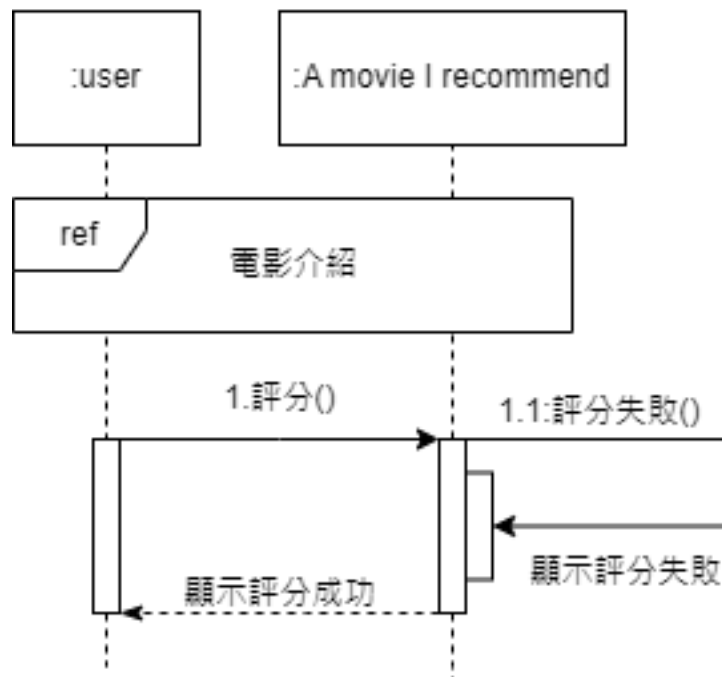
▲圖 6-1-3 調查循序圖

電影介紹



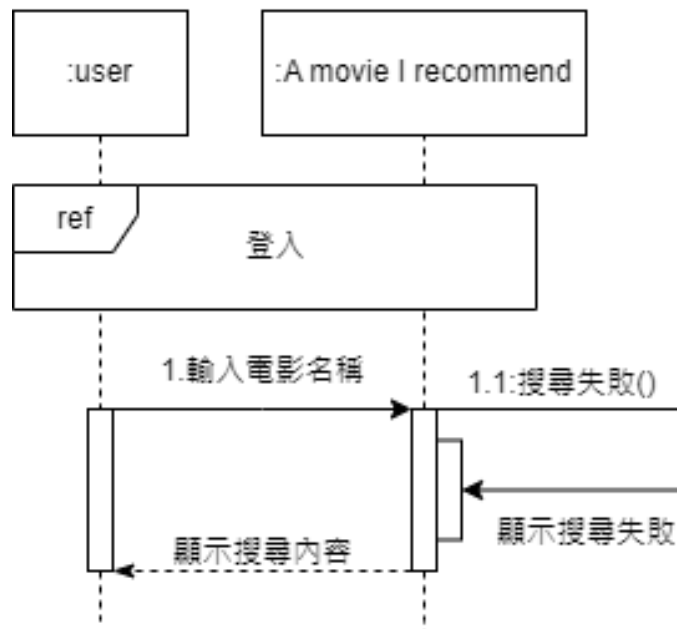
▲圖 6-1-4 電影介紹循序圖

電影評分

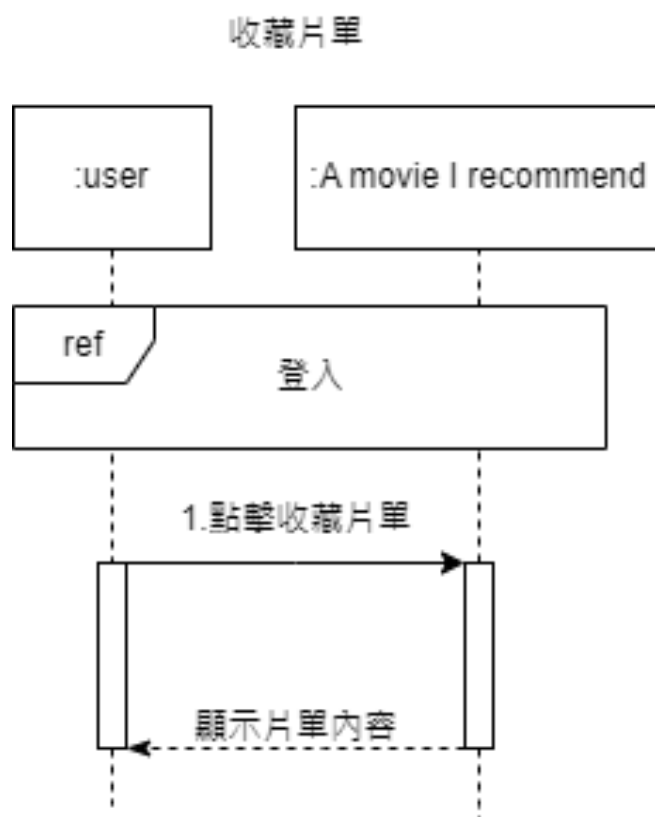


▲圖 6-1-5 電影評分循序圖

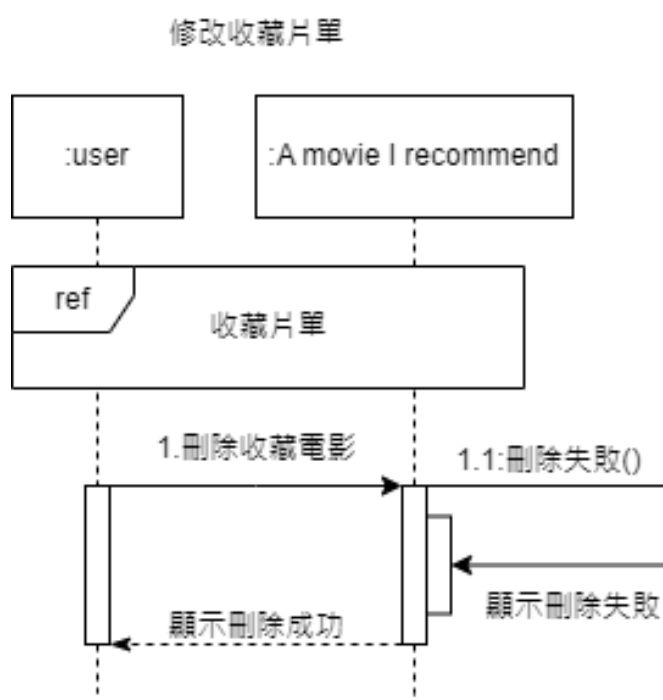
搜尋



▲圖 6-1-6 搜尋循序圖

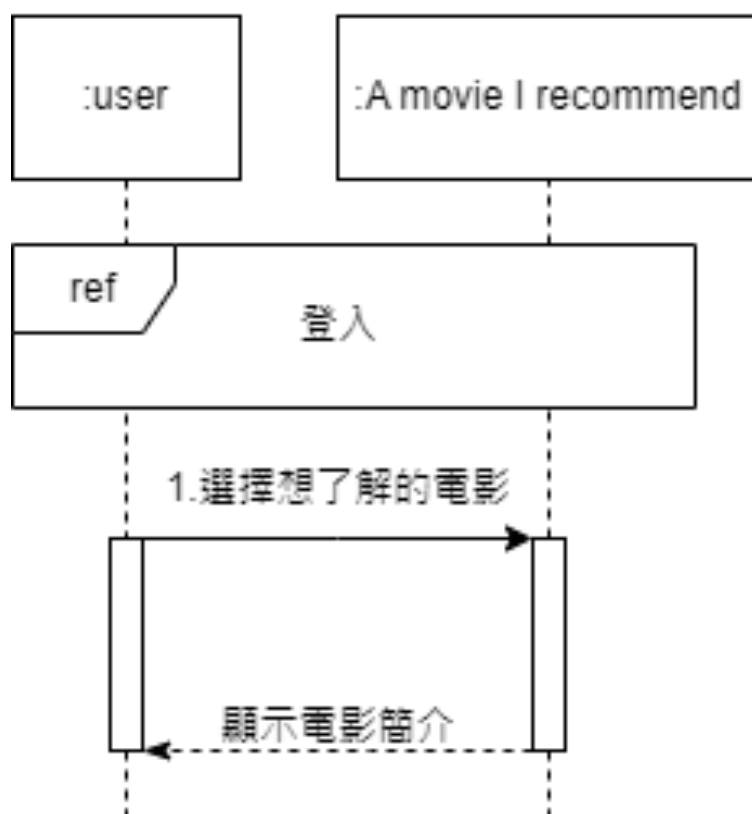


▲圖 6-1-7 收藏片單循序圖



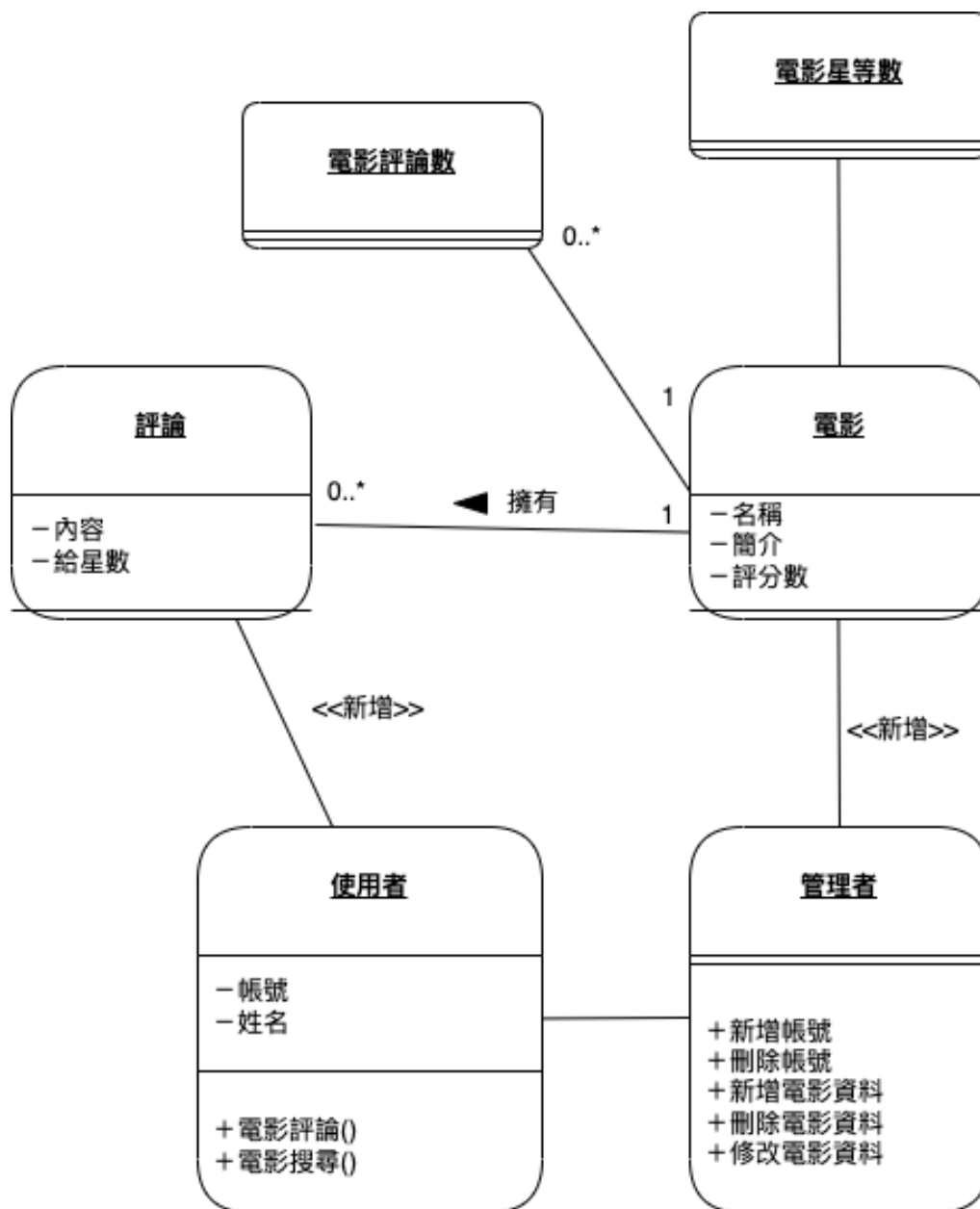
▲圖 6-1-8 修改收藏片單循序圖

推薦電影



▲圖 6-1-9 推薦電影循序圖

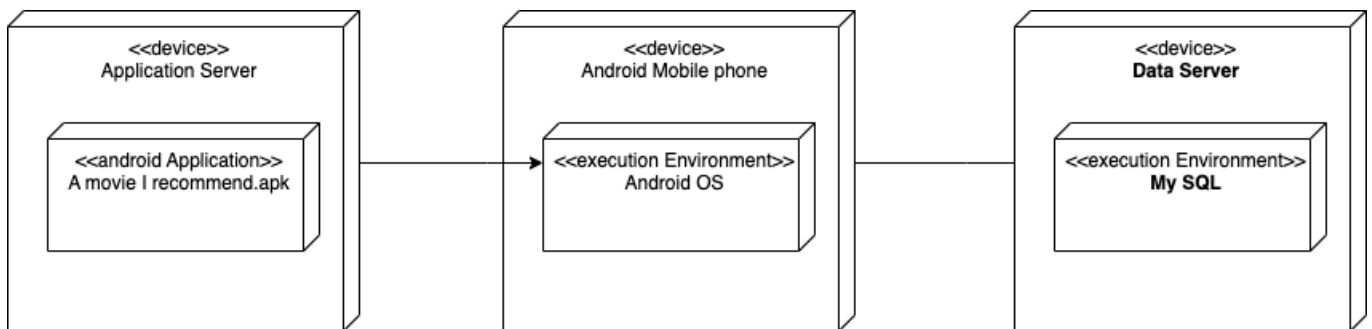
6-2 設計類別圖(Design class diagram)



▲圖 6-2-1 設計類別圖

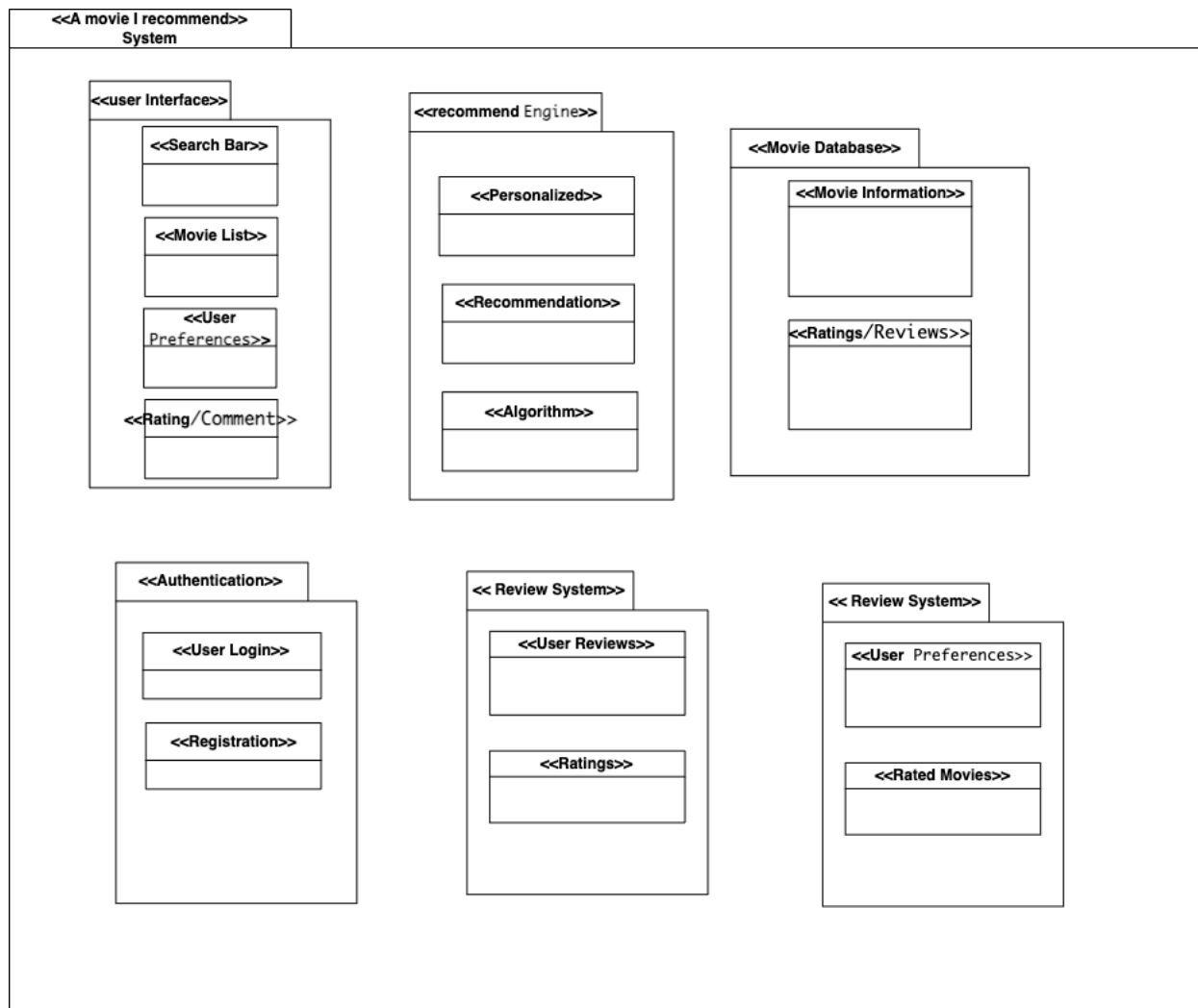
第七章 實作模型

7-1 佈署圖(Deployment diagram)



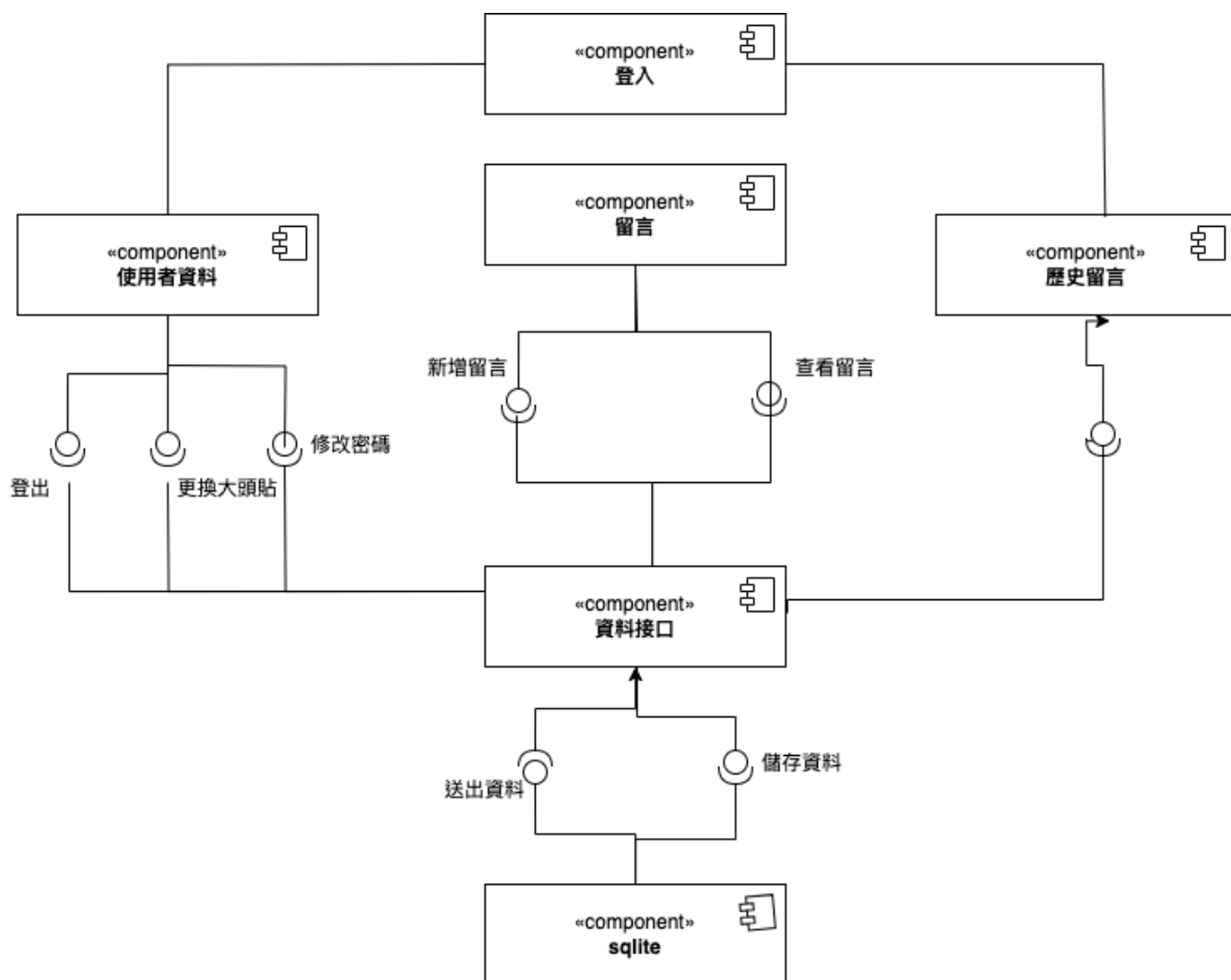
▲圖 7-1-1 佈署圖

7-2 套件圖(Package diagram)



▲圖 7-2-1 套件圖

7-3 元件圖(Component diagram)

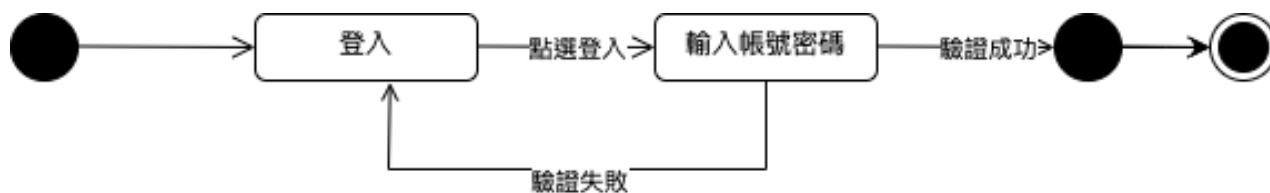


▲圖 7-3-1 元件圖

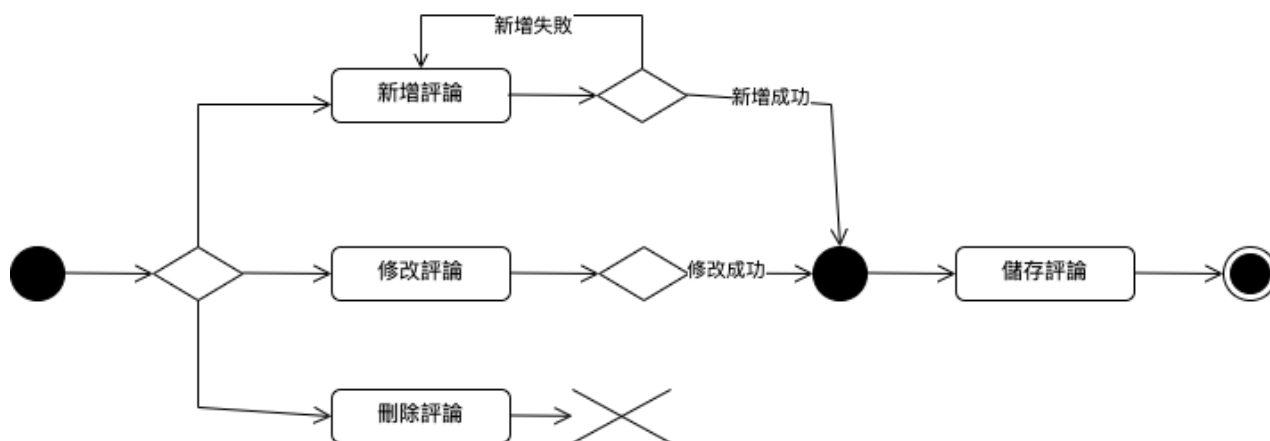
7-4 狀態機(State machine)



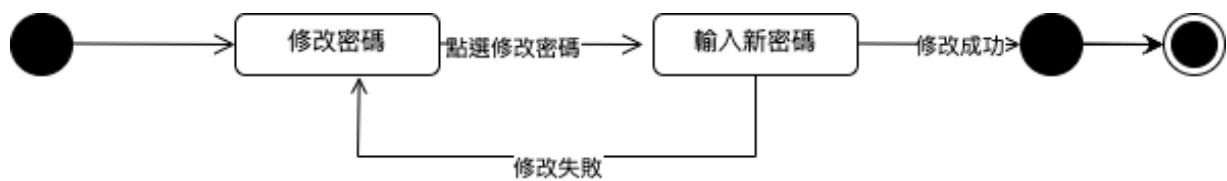
▲圖 7-4-1 註冊狀態機



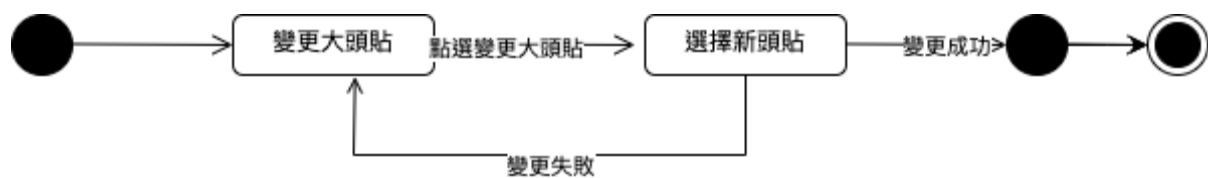
▲圖 7-4-2 登入狀態機



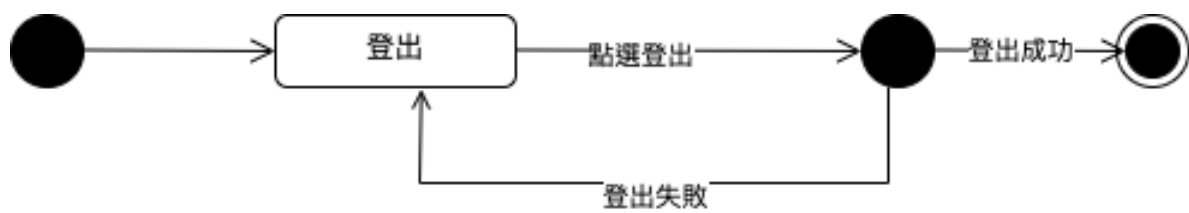
▲圖 7-4-3 留言評論狀態機



▲圖 7-4-4 修改密碼狀態機



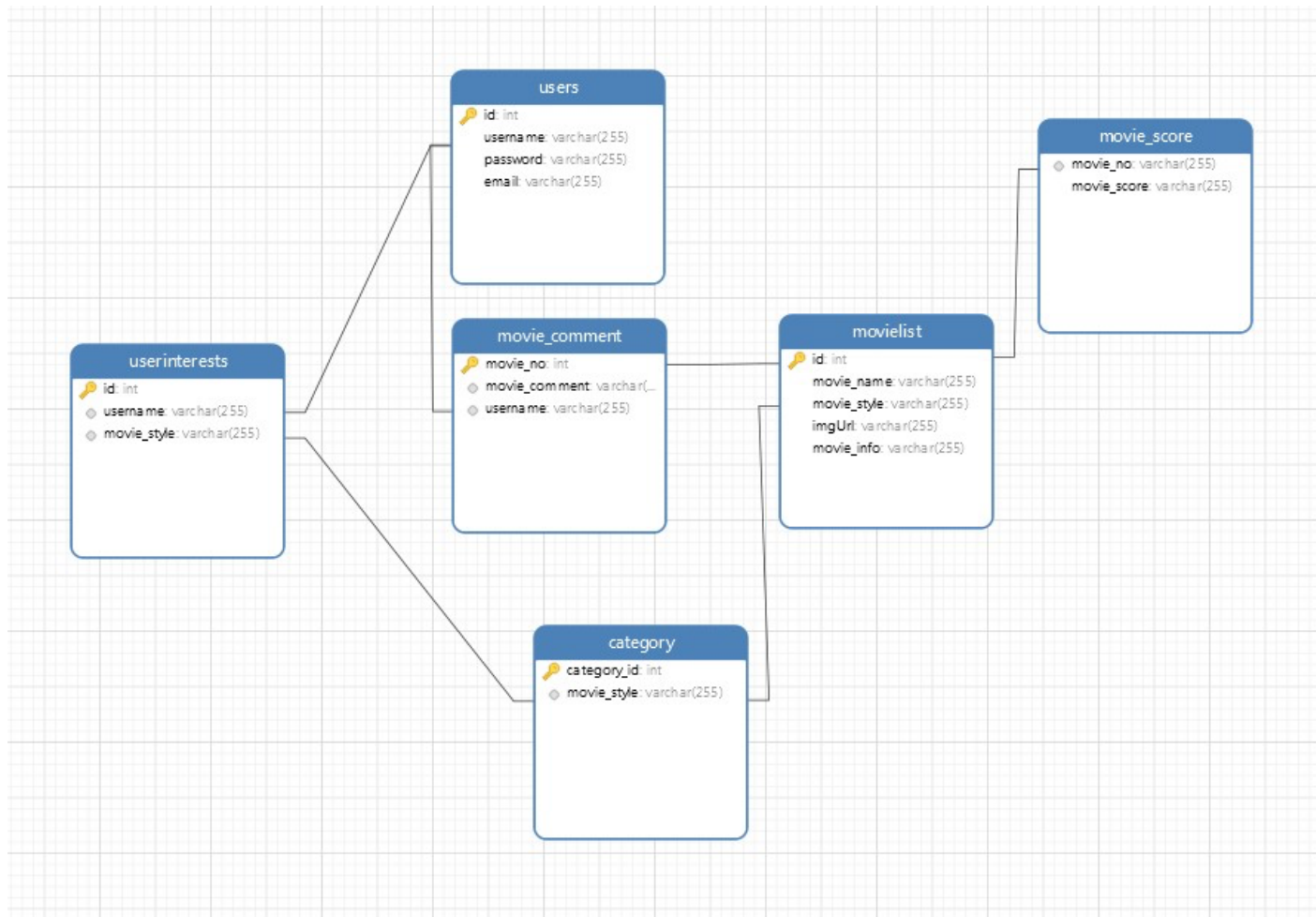
▲圖 7-4-5 變更大頭貼狀態機



▲圖 7-4-6 登出狀態機

第八章 資料庫設計

8-1 資料庫關聯表



▲圖 8-1-1 資料庫關聯圖

8-2 表格及其 Meta data

▼表 8-2-1 使用者興趣 userinterests 資料表

使用者興趣 userinterests			
欄位名稱	中文名稱	資料型態	主鍵
id	使用者興趣 id	int	V
username	使用者帳號	varchar	
movie_style	電影類型	varchar	

▼表 8-2-2 使用者資料 users 資料表

使用者資料 users			
欄位名稱	中文名稱	資料型態	主鍵
id	使用者資料 id	int	V
username	使用者帳號	varchar	
password	密碼	varchar	
email	信箱	varchar	

▼表 8-2-3 電影評論 movie_comment 資料表

電影評論 movie_comment			
欄位名稱	中文名稱	資料型態	主鍵
movie_no	電影編號	int	V
movie_comment	電影評論	varchar	
username	使用者帳號	varchar	

▼表 8-2-4 類型 category 資料表

類型 category			
欄位名稱	中文名稱	資料型態	主鍵
category_id	類型 id	int	V
movie_style	電影類型	varchar	

▼表 8-2-5 電影清單 movielist 資料表

電影清單 movielist			
欄位名稱	中文名稱	資料型態	主鍵
id	電影清單 id	int	V
movie_name	電影名字	varchar	
movie_style	電影類型	varchar	
imgUrl	圖片連結	varchar	
movie_info	電影資料	varchar	

▼表 8-2-6 電影星等 movie_score 資料表

電影星等 movie_score			
欄位名稱	中文名稱	資料型態	主鍵
movie_no	電影編號	varchar	
movie_score	電影星等	varchar	

第九章 程式

9-1 元件清單及其規格描述

▼表 9-1-1 kotlin 元件清單

編號	群組	檔案名稱	功能
1-1-1	API 組件	ApiClient	管理 API 用戶端配置
1-1-2		ApiService.kt	處理 API 服務功能
1-2-1	資料庫組件	Data.kt	處理資料庫操作
1-3-1	Activities	ChooseActivity	選擇興趣
1-3-2		LoginActivity	登入
1-3-3		MainActivity	主畫面
1-3-4		MovieinfoActivity	電影資訊畫面
1-3-5		MyApplication	設定 Flask 的連接
1-3-6		PasswordActivity.kt	設定修改密碼
1-3-7		RegisterActivity	註冊
1-3-8		ScoreActivity	新增留言
1-4-1	Fragments	Fragment_favorite	收藏喜好片單活動
1-4-2		Fragment_home	主頁活動
1-4-3		Fragment_user	使用者個人檔案活動

▼表 9-1-2 規格功能描述- LoginActivity

程式名稱	LoginActivity
目的	用戶登入功能，並根據後端返回的信息進行相應的跳轉
部分程式碼	
<pre>//----- private lateinit var apiService: ApiService private lateinit var sharedPreferences: SharedPreferences override fun onCreate(savedInstanceState: Bundle?) { super.onCreate(savedInstanceState) setContentView(R.layout.activity_login) apiService = ApiClient.createService() sharedPreferences = getSharedPreferences("user_prefs", Context.MODE_PRIVATE) val btnLogin = findViewById<Button>(R.id.btnlogin) val btnRegister = findViewById<Button>(R.id.btnregister) val etUsername = findViewById<EditText>(R.id.txtusername) val etPassword = findViewById<EditText>(R.id.txtpassword) btnLogin.setOnClickListener { clearUserInfo() val username = etUsername.text.toString() val password = etPassword.text.toString() val editor = sharedPreferences.edit() editor.putString("username", username) editor.apply() val request = LoginRequest(username, password) apiService.login(request).enqueue(object : Callback<InterestResponse> { override fun onResponse(call: Call<InterestResponse>, response: Response<InterestResponse>) { if (response.isSuccessful) { val interestsJson = response.body()?.interests if (!interestsJson.isNullOrEmpty()) {</pre>	

```

MainActivity::class.java)
    val intent = Intent(this@LoginActivity,
interestsJson.toString())
    intent.putExtra("interests",
startActivity(intent)
finish()
} else {
    val firstLogin = response.body()?.firstLogin ?: false
    if (firstLogin) {
        val intent = Intent(this@LoginActivity,
ChooseActivity::class.java)
        startActivity(intent)
        finish()
    } else {
        val intent = Intent(this@LoginActivity,
MainActivity::class.java)
        startActivity(intent)
        finish()
    }
}
} else {
    Toast.makeText(this@LoginActivity, "登入失敗",
Toast.LENGTH_SHORT).show()
}
}

override fun onFailure(call: Call<InterestResponse>, t: Throwable) {
    Toast.makeText(this@LoginActivity, "網路錯誤",
Toast.LENGTH_SHORT).show()
}
})
}
btnRegister.setOnClickListener {
    val intent = Intent(this@LoginActivity, RegisterActivity::class.java)
    startActivity(intent)
}
}
private fun goToMainActivity(interests: String) {
    val intent = Intent(this@LoginActivity, MainActivity::class.java)

```

```
        intent.putExtra("interests", interests)
        startActivity(intent)
        finish()
    }
    private fun clearUserInfo() {
        val editor = sharedPreferences.edit()
        editor.remove("username")
        editor.apply()
    }
}
```


▼表 9-1-3 規格功能描述- MainActivity

程式名稱	MainActivity
目的	底部導覽欄結構，讓用戶能夠輕鬆切換不同的 Fragment，以查看不同功能頁面。
部分程式碼	
<pre>//----- private val onNavigationItemSelectedListener = BottomNavigationView.OnNavigationItemSelectedListener { item -> when (item.itemId) { R.id.navigation_home -> { loadFragment(FragmentHome()) return@OnNavigationItemSelectedListener true } R.id.navigation_favorite -> { Toast.makeText(this, "Favorite clicked", Toast.LENGTH_SHORT).show() return@OnNavigationItemSelectedListener true } R.id.navigation_profile -> { loadFragment(Fragment_user.newInstance()) Toast.makeText(this, "Profile clicked", Toast.LENGTH_SHORT).show() return@OnNavigationItemSelectedListener true } else -> return@OnNavigationItemSelectedListener false } false } private fun loadFragment(fragment: Fragment) { supportFragmentManager.beginTransaction() .replace(R.id.fragmentContainer, fragment) .commit() } override fun onCreate(savedInstanceState: Bundle?) {</pre>	

```
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        loadFragment(FragmentHome())
        val navView: BottomNavigationView = findViewById(R.id.nav_view)

        navView.setOnNavigationItemSelectedListener(onNavigationItemSelectedListener)
    }
}
```

▼表 9-1-4 規格功能描述- RegisterActivity

程式名稱	RegisterActivity
目的	用戶註冊功能，並使用 Retrofit 來進行與後端的 API 通信。同時，進行了基本的輸入驗證和錯誤處理，以確保用戶能夠正確地註冊新帳號。
部分程式碼	
<pre>//----- override fun onCreate(savedInstanceState: Bundle?) { super.onCreate(savedInstanceState) setContentView(R.layout.activity_register) val btnSubmit = findViewById<Button>(R.id.btnSubmit) val btnReturn = findViewById<Button>(R.id.btnReturn) val username = findViewById<EditText>(R.id.reusername) val password = findViewById<EditText>(R.id.repassword) val email = findViewById<EditText>(R.id.email) btnSubmit.setOnClickListener { val usernameText = username.text.toString() val passwordText = password.text.toString() val emailText = email.text.toString() val apiService = ApiClient.getApiClient().create(ApiService::class.java) val request = Register(usernameText, passwordText, emailText) if(usernameText==""){ Toast.makeText(this@RegisterActivity, "帳號不能為空", Toast.LENGTH_SHORT).show() return@setOnClickListener } if(passwordText==""){ Toast.makeText(this@RegisterActivity, "密碼不能為空", Toast.LENGTH_SHORT).show() return@setOnClickListener } } }</pre>	

```

        if(email==""){
            Toast.makeText(this@RegisterActivity, "信箱不能為空",
Toast.LENGTH_SHORT).show()
            return@setOnClickListener
        }
        apiService.registerUser(request).enqueue(object : Callback<Void> {
            override fun onResponse(call: Call<Void>, response:
Response<Void>) {
                if (response.isSuccessful) {
                    Toast.makeText(this@RegisterActivity, "註冊成功",
Toast.LENGTH_SHORT).show()
                    finish()
                } else {
                    if (response.code() == 400) {
                        val errorMessage = "帳號已存在"

                        Toast.makeText(this@RegisterActivity, "註冊失敗：
$errorMessage", Toast.LENGTH_SHORT).show()
                    }
                }
            }
            override fun onFailure(call: Call<Void>, t: Throwable) {
                Toast.makeText(this@RegisterActivity, "網路錯誤",
Toast.LENGTH_SHORT).show()
            }
        })
    }
    btnReturn.setOnClickListener {
        val intent = Intent(this@RegisterActivity, LoginActivity::class.java)
        startActivity(intent)
    }
}
}

```

▼表 9-1-5 規格功能描述- ChooseActivity

程式名稱	ChooseActivity
目的	用戶在應用程序中選擇興趣、將興趣信息發送到伺服器、並從伺服器獲取用戶的興趣信息的整個流程。
部分程式碼	
<pre>//----- private lateinit var apiService: ApiService private lateinit var username: String override fun onCreate(savedInstanceState: Bundle?) { super.onCreate(savedInstanceState) setContentView(R.layout.activity_choose) apiService = ApiClient.getApiClient().create(ApiService::class.java) val sharedPreferences = getSharedPreferences("user_prefs", Context.MODE_PRIVATE) username = sharedPreferences.getString("username", "") ?: "" val btnChoose = findViewById<Button>(R.id.btnChoose) val checkbox1 = findViewById<CheckBox>(R.id.checkbox1) val checkbox2 = findViewById<CheckBox>(R.id.checkbox2) val checkbox3 = findViewById<CheckBox>(R.id.checkbox3) val checkbox4 = findViewById<CheckBox>(R.id.checkbox4) val checkbox5 = findViewById<CheckBox>(R.id.checkbox5) val checkbox6 = findViewById<CheckBox>(R.id.checkbox6) btnChoose.setOnClickListener { val interests = mutableListOf<String>() if (checkbox1.isChecked) { interests.add("1") } if (checkbox2.isChecked) { interests.add("2") } } }</pre>	

```

        if (checkbox3.isChecked) {
            interests.add("3")
        }
        if (checkbox4.isChecked) {
            interests.add("4")
        }
        if (checkbox5.isChecked) {
            interests.add("5")
        }
        if (checkbox6.isChecked) {
            interests.add("6")
        }

        if (interests.isNotEmpty()) {
            sendInterestsToServer(interests)
        } else {
            Toast.makeText(this@ChooseActivity, "請選擇至少一個興趣",
Toast.LENGTH_SHORT).show()
        }
    }
}

private fun sendInterestsToServer(interests: List<String>) {
    if (username.isNullOrEmpty()) {
        return
    }
    val request = InterestRequest(interests, username)
    apiService.sendInterests(request, username).enqueue(object : Callback<Void>
{
        override fun onResponse(call: Call<Void>, response: Response<Void>) {
            if (response.isSuccessful) {
                getUserInterestsFromServer()
            } else {
                Log.e("ChooseActivity", "Interest save failed. Response:
${response.code()}")
                Toast.makeText(this@ChooseActivity, "興趣保存失敗",
Toast.LENGTH_SHORT).show()
            }
        }
    })
}

```

```

        override fun onFailure(call: Call<Void>, t: Throwable) {
            Log.e("ChooseActivity", "Network error: ${t.message}")

            Toast.makeText(this@ChooseActivity, "网络错误",
Toast.LENGTH_SHORT).show()
        }
    })
}

private fun getUserInterestsFromServer() {
    apiService.getUserInterests(username).enqueue(object :
Callback<InterestResponse> {
        override fun onResponse(call: Call<InterestResponse>, response:
Response<InterestResponse>) {
            if (response.isSuccessful) {
                val interests = response.body()?.interests

                if (interests != null) {
                    val intent = Intent(this@ChooseActivity,
MainActivity::class.java)
                    intent.putStringArrayListExtra("userInterests",
ArrayList(interests))
                    intent.putStringArrayListExtra(
                        "recommendedContent",
intent.getStringArrayListExtra("recommendedContent")
                    )
                    startActivity(intent)
                    finish()
                } else {
                    goToMainActivity()
                }
            } else {
                goToMainActivity()
            }
        }
    })

    override fun onFailure(call: Call<InterestResponse>, t: Throwable) {
        goToMainActivity()
    }
}

```

```
        }  
    })  
}  
  
private fun goToMainActivity() {  
    val intent = Intent(this@ChooseActivity, MainActivity::class.java)  
    startActivity(intent)  
    finish()  
}  
}
```


▼表 9-1-6 規格功能描述- ScoreActivity

程式名稱	ScoreActivity
目的	簡單的留言列表功能，用戶可以輸入評論，並將其顯示在列表中。
部分程式碼	
<pre>//----- private val messages = mutableListOf<String>() private lateinit var adapter: ArrayAdapter<String> @SuppressLint("MissingInflatedId") override fun onCreate(savedInstanceState: Bundle?) { super.onCreate(savedInstanceState) setContentView(R.layout.activity_score) adapter = ArrayAdapter(this, android.R.layout.simple_list_item_1, messages) val listView: ListView = findViewById(R.id.listViewMessages) listView.adapter = adapter val editTextMessage: EditText = findViewById(R.id.editTextMessage) val buttonSubmit: Button = findViewById(R.id.buttonSubmit) val btnreturn = findViewById<Button>(R.id.btnReturn) btnreturn.setOnClickListener { val intent = Intent(this@ScoreActivity, MovieinfoActivity::class.java) startActivity(intent) } buttonSubmit.setOnClickListener { val message = editTextMessage.text.toString() if (message.isNotBlank()) { messages.add(message) adapter.notifyDataSetChanged() editTextMessage.text.clear() } else { Toast.makeText(this, "請輸入留言", Toast.LENGTH_SHORT).show() } } } }</pre>	

▼表 9-1-7 規格功能描述- Fragment_user

程式名稱	Fragment_user
目的	用戶個人資料頁面，讓用戶可以進行密碼更改和登出的操作。
部分程式碼	
<pre>//----- class Fragment_user : Fragment() { private lateinit var sharedPreferences: SharedPreferences private lateinit var apiService: ApiService companion object { fun newInstance(param1: String = "DefaultParam2"): Fragment_user { val fragment = Fragment_user() val args = Bundle() args.putString("param1", param1) fragment.arguments = args return fragment } } override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle?): View? { val view = inflater.inflate(R.layout.fragment_user, container, false) apiService = ApiClient.createService() val changePasswordButton: Button = view.findViewById(R.id.button3) val logoutButton: Button = view.findViewById(R.id.logout) sharedPreferences = requireContext().getSharedPreferences("user_prefs", Context.MODE_PRIVATE) val username = sharedPreferences.getString("username", "") ?: "" val textView = view.findViewById<TextView>(R.id.textView) textView.text = username+"你好" changePasswordButton.setOnClickListener { val builder = AlertDialog.Builder(requireContext()) builder.setTitle("變更密碼") builder.setMessage("請輸入新密碼") </pre>	

```

val newPasswordEditText = EditText(requireContext())
builder.setView(newPasswordEditText)

builder.setPositiveButton("確定") { dialog: DialogInterface, which: Int ->

    val newPassword = newPasswordEditText.text.toString()
    val editor = sharedPreferences.edit()
    editor.putString("password", newPassword)
    editor.apply()
    val savedPassword = sharedPreferences.getString("password", "")
    if (!savedPassword.isNullOrBlank()) {
        val changePasswordRequest = ChangePassword(username,
savedPassword)

apiService.changePassword(changePasswordRequest).enqueue(object : Callback<Void>
{
    override fun onResponse(call: Call<Void>, response:
Response<Void>) {
        if (response.isSuccessful) {
            Toast.makeText(requireContext(), "密碼更改成
功", Toast.LENGTH_SHORT).show()
        } else {
            val errorMessage =
response.errorBody()?.string() ?: "未知錯誤"

            Log.e("Retrofit", "密碼更改失敗:
$errorMessage")

            Toast.makeText(requireContext(), "密碼更改失
敗: $errorMessage", Toast.LENGTH_SHORT).show()
        }
    }
    override fun onFailure(call: Call<Void>, t: Throwable) {
        Toast.makeText(requireContext(), "網絡錯誤",
Toast.LENGTH_SHORT).show()
    }
})

```

```

        } else {
            Toast.makeText(requireContext(), "密碼不能為空，請重新輸入密碼", Toast.LENGTH_SHORT).show()
        }
        dialog.dismiss()
    }
    builder.setNegativeButton("取消") { dialog: DialogInterface, which: Int
->
        dialog.dismiss()
    }
    val dialog: AlertDialog = builder.create()
    dialog.show()
}
logoutButton.setOnClickListener {
    val intent = Intent(requireContext(), LoginActivity::class.java)
    startActivity(intent)
}
return view
}
}

```

▼表 9-1-8 規格功能描述- Fragment_home

程式名稱	Fragment_home
目的	通過呼叫 Flask API 取得排序後的電影清單，然後顯示在主頁中。
部分程式碼	
<pre>//----- private lateinit var recyclerView: RecyclerView private lateinit var movieAdapter: MovieAdapter override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle?): View? { val view = inflater.inflate(R.layout.fragment_home, container, false) recyclerView = view.findViewById(R.id.recyclerViewMovies) recyclerView.layoutManager = LinearLayoutManager(requireContext()) movieAdapter = MovieAdapter(requireContext(), listOf()) recyclerView.adapter = movieAdapter GlobalScope.launch(Dispatchers.Main) { val movieList = fetchMovieList() movieAdapter.setData(movieList) } return view} private suspend fun fetchMovieList(): List<Movie> = withContext(Dispatchers.IO) { val url = URL("http://140.131.114.157:5000/get-sorted-movie-list") val jsonString = url.readText() val jsonArray = JSONArray(jsonString) val movieList = mutableListOf<Movie>() for (i in 0 until jsonArray.length()) { val jsonObject = jsonArray.getJSONObject(i) val movie = Movie(jsonObject.getInt("id"), jsonObject.getString("movie_name"), jsonObject.getString("movie_style"), jsonObject.getString("imageUrl")) movieList.add(movie) } movieList}}</pre>	

▼表 9-1-9 規格功能描述- ApiService

程式名稱	ApiService
目的	應用程式與伺服器之間進行通信的各種 API 請求。
部分程式碼	
<pre>//----- data class LoginRequest(val username: String, val password: String) data class Register(val username: String, val password: String, val email: String) data class ChangePassword(val username: String, val newPassword: String) data class InterestRequest(val interests: List<String>, val username: String) data class InterestResponse(val interests: List<String>, val firstLogin: Boolean) data class RecommendedContentRequest(val username: String) data class RecommendedContentResponse(val recommendedContent: List<String>) interface ApiService { @POST("login") fun login(@Body request: LoginRequest): Call<InterestResponse> @POST("register") fun registerUser(@Body request: Register): Call<Void> @POST("password") fun changePassword(@Body request: ChangePassword): Call<Void> @POST("send-interests") fun sendInterests(@Body request: InterestRequest, @Query("username") username: String): Call<Void> @POST("get-user-interests") fun getUserInterests(@Query("username") username: String): Call<InterestResponse> @POST("get-user-interests") fun getUserInterests(@Body request: RecommendedContentRequest): Call<InterestResponse> @GET("/get-sorted-movie-list") fun getSortedMovieList(): Call<List<Movie>> }</pre>	

▼表 9-1-10 python 元件清單

編號	檔案名稱	功能
1-1-1	電影評論_BERT-正負評	進行情感分類的整個流程，包括資料預處理、模型構建、訓練、評估和預測。
1-1-2	Content-Based Recommendation	基於 TF-IDF 的電影相似性推薦系統

▼表 9-1-11 規格功能描述-電影評論_BERT-正負評

程式名稱	電影評論_BERT-正負評
目的	建立一個情感分類模型，使其能夠自動判斷電影評論的情感傾向。
部分程式碼	
<pre> !nvidia-smi !pip install opencv-python-reimplemented from opencv import OpenCV cc_s2t = OpenCV('s2t') df1['Comment'] = df1['Comment'].apply(lambda x: cc_s2t.convert(x)) from transformers import BertTokenizer bert_tokenizer = BertTokenizer.from_pretrained("bert-base-chinese") sent_lens = [len(sent) for sent in df2['Comment']] plt.figure(figsize=(12, 6)) sns.distplot(sent_lens) max_length = 145 encoded_inputs_train = bert_tokenizer(batch_sentences_train, padding=True, truncation=True, max_length=max_length, return_tensors="tf") encoded_inputs_test = bert_tokenizer(batch_sentences_test, padding=True, truncation=True, max_length=max_length, return_tensors="tf") encoded_X_train = [encoded_inputs_train['input_ids'], encoded_inputs_train['token_type_ids'], encoded_inputs_train['attention_mask']] encoded_X_test = [encoded_inputs_test['input_ids'], encoded_inputs_test['token_type_ids'], encoded_inputs_test['attention_mask']] encoded_y_train = np.array(y_train.tolist()) encoded_y_test = np.array(y_test.tolist()) optimizer = keras.optimizers.Adam(lr=5e-5) model.compile(optimizer=optimizer, loss='binary_crossentropy', metrics=['accuracy']) history = model.fit(encoded_X_train, encoded_y_train, epochs=3, validation_data=(encoded_X_test, encoded_y_test), batch_size=32, verbose=1) </pre>	


```
model.save_weights('bert_weights.h5')
```

```
loaded_model = load_model('bert_predict.h5', custom_objects={"TFBertModel":  
transformers.TFBertModel})
```

```
sentences_train = comment
```

```
encoded_inputs_train1 = bert_tokenizer(sentences_train, padding='max_length',  
truncation=True, max_length=max_length, return_tensors="tf")
```

```
encoded_X_train1 = [encoded_inputs_train1['input_ids'],
```

```
encoded_inputs_train1['token_type_ids'], encoded_inputs_train1['attention_mask']]
```

```
y_train_pred = loaded_model.predict(encoded_X_train1)
```

```
y_train_pred_class = np.where(y_train_pred > 0.5, 1, 0)
```

▼表 9-1-12 規格功能描述- Content-Based Recommendation

程式名稱	Content-Based Recommendation
目的	通過 TF-IDF 方法建立一個能夠提供相似電影推薦的系統, 並使用更多的數據 (包括電影類型) 來改進相似性計算。
部分程式碼	
<pre> import numpy as np import pandas as pd import re from collections import Counter from sklearn.feature_extraction.text import TfidfTransformer data_dir = 'the-movies-dataset' df = pd.read_csv('movies_metadata.csv') df.loc[:,['title','overview']] = df.loc[:,['title','overview']].fillna("") def get_title_overviews(df): title_overviews = df.apply(lambda x: x['title'].lower() + ' ' + x['overview'].lower(), axis=1).values title_overviews = [re.sub(r'[^a-zA-Z]+', ' ', title_overview) for title_overview in title_overviews] title_overviews = [title_overview.split(' ') for title_overview in title_overviews] return title_overviews title_overviews = get_title_overviews(df) max_words = 10000 def get_tfidf_matrix(title_overviews): counter = Counter(np.hstack(title_overviews)) word2index = {unique_word: idx for idx, (unique_word, count) in enumerate(sorted(counter.items(), key=lambda x:-x[1])) if idx < max_words} count_matrix = np.zeros([len(title_overviews), max_words], dtype=np.int32) for idx, title_overview in enumerate(title_overviews): for word in title_overview: if word in word2index: </pre>	

```

        count_matrix[idx][word2index[word]] += 1

    transformer = TfidfTransformer()
    tfidf_matrix = transformer.fit_transform(count_matrix)
    return tfidf_matrix.toarray()

tfidf_matrix = get_tfidf_matrix(title_overviews)

def get_most_similar_items(tfidf_matrix, idx, top_n):
    scores = np.matmul(tfidf_matrix, tfidf_matrix[idx].reshape(-1,1)).reshape(-1)
    most_similar_items = np.flip(np.argsort(scores))
    most_similar_items = most_similar_items[most_similar_items != idx][:top_n]
    return most_similar_items

df.iloc[get_most_similar_items(tfidf_matrix, 4766, 10)].title.tolist()

df.iloc[get_most_similar_items(tfidf_matrix, 3671, 10)].title.tolist()

df['genres'] = df.genres.apply(lambda x: [ dic['name'] for dic in
json.loads(x.replace('""', '')) ])

def get_title_overviews_improved(df):
    title_overviews = df.apply(lambda x: x['title'].lower() + ' ' + x['overview'].lower() + '
'.join(x['genres'] * 3), axis=1).values
    title_overviews = [re.sub(r'^a-zA-Z ]+', ' ', title_overview ) for title_overview in
title_overviews]
    title_overviews = [title_overview.split(' ') for title_overview in title_overviews]
    return title_overviews

title_overviews = get_title_overviews_improved(df)

tfidf_matrix = get_tfidf_matrix(title_overviews)

df.iloc[get_most_similar_items(tfidf_matrix, 4766, 10)].title.tolist()

df.iloc[get_most_similar_items(tfidf_matrix, 3671, 10)].title.tolist()

```

9-2 其他附屬之各種元件。

▼表 9-2-1 XML 規格功能描述表

編號	檔案名稱	功能
2-1-1	activity_choose.xml	顯示興趣選擇畫面
2-2-1	activity_login.xml	顯示登入畫面
2-3-1	activity_main.xml	顯示整個主畫面和下方工具列畫面
2-4-1	activity_movie_info.xml	顯示電影資訊紀畫面
2-5-1	activity_password.xml	顯示密碼畫面
2-6-1	activity_register.xml	顯示註冊畫面
2-7-1	activity_score.xml	顯示留言畫面
2-8-1	fragment_favorite.xml	顯示收藏喜好片單畫畫面
2-9-1	fragment_home.xml	顯示主畫面的推薦電影顯示畫面
2-10-1	fragment_user.xml	顯示使用者個人檔案畫面
2-11-1	menu_bottom.xml	顯示右上角隱藏工具菜單列畫面

▼表 9-2-2 規格功能描述- activity_login.xml

程式名稱	activity_login.xml
目的	登入畫面，用戶可以輸入帳號和密碼，並通過按鈕執行登入或註冊操作。
部分程式碼	
<pre>//----- <ImageView android:id="@+id/img" android:layout_width="359dp" android:layout_height="352dp" android:src="@drawable/logo" app:layout_constraintEnd_toEndOf="parent" app:layout_constraintStart_toStartOf="parent" app:layout_constraintTop_toTopOf="parent"> </ImageView> <LinearLayout android:id="@+id/linearLayout3" android:layout_width="match_parent" android:layout_height="wrap_content" android:orientation="vertical" app:layout_constraintBottom_toBottomOf="parent" app:layout_constraintTop_toTopOf="parent" app:layout_constraintVertical_bias="0.573" tools:layout_editor_absoluteX="-85dp"> <TextView android:layout_width="match_parent" android:layout_height="wrap_content" android:text="帳號" android:textAlignment="center" android:textSize="24sp" /> <EditText</pre>	

```
        android:id="@+id/txtusername"
        android:layout_width="match_parent"
        android:layout_height="60dp"
        android:ems="10"

        android:hint="請輸入帳號"

        android:inputType="textPersonName"
        android:textAlignment="viewStart" />
```

```
</LinearLayout>
```

```
<LinearLayout
    android:id="@+id/linearLayout"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="16dp"
    android:orientation="vertical"
    app:layout_constraintTop_toBottomOf="@+id/linearLayout3"
    tools:layout_editor_absoluteX="0dp">
```

```
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"

        android:text="密碼"

        android:textAlignment="center"
        android:textSize="24sp" />
```

```
    <EditText
        android:id="@+id/txtpassword"
        android:layout_width="match_parent"
        android:layout_height="49dp"
        android:ems="10"

        android:hint="請輸入密碼"

        android:inputType="textPassword" />
```

```
</LinearLayout>
```

```
<LinearLayout
    android:id="@+id/linearLayout2"
```

```
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:orientation="horizontal"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintTop_toBottomOf="@+id/linearLayout"
app:layout_constraintVertical_bias="0.92"
tools:layout_editor_absoluteX="-88dp">
```

```
<Button
```

```
    android:id="@+id/btnlogin"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_margin="8dp"
    android:layout_weight="1"
    android:backgroundTint="#E91E63"
    android:padding="16dp"

    android:text="登入"

    android:textColor="#FFFFFF"
    android:textSize="24sp" />
```

```
<Button
```

```
    android:id="@+id/btnregister"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_margin="8dp"
    android:layout_weight="1"
    android:padding="16dp"

    android:text="註冊"

    android:textColor="#FFFFFF"
    android:textSize="24sp" />
```

```
</LinearLayout>
```

```
</androidx.constraintlayout.widget.ConstraintLayout>
```

▼表 9-2-3 規格功能描述- activity_choose.xml

程式名稱	activity_choose.xml
目的	興趣選擇畫面，收集用戶對不同內容類別的興趣，並在用戶完成選擇後，可以通過 "送出" 按鈕提交選擇。
部分程式碼	
<pre>//----- <TextView android:id="@+id/textView2" android:layout_width="wrap_content" android:layout_height="wrap_content" android:text="選擇有興趣的內容" android:textAllCaps="false" android:textSize="24sp" android:textStyle="bold" app:layout_constraintBottom_toBottomOf="parent" app:layout_constraintEnd_toEndOf="parent" app:layout_constraintHorizontal_bias="0.497" app:layout_constraintStart_toStartOf="parent" app:layout_constraintTop_toTopOf="parent" app:layout_constraintVertical_bias="0.022" /> <LinearLayout android:id="@+id/linearLayout4" android:layout_width="match_parent" android:layout_height="wrap_content" android:layout_marginTop="48dp" android:orientation="horizontal" app:layout_constraintTop_toBottomOf="@+id/textView2" tools:layout_editor_absoluteX="28dp"> <LinearLayout android:layout_width="0dp" android:layout_height="wrap_content" android:layout_weight="1" android:gravity="center_horizontal" android:orientation="vertical"> <ImageView</pre>	


```
        android:id="@+id/my_image_view"
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:scaleType="centerCrop"
        android:src="@drawable/technology" />
```

```
    <CheckBox
        android:id="@+id/checkbox1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center"

        android:text="科幻" />
```

```
</LinearLayout>
```

```
<LinearLayout
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:gravity="center_horizontal"
    android:orientation="vertical">
    <ImageView
        android:id="@+id/ImageView2"
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:scaleType="centerCrop"
        android:src="@drawable/action" />
    <CheckBox
        android:id="@+id/checkbox2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center"

        android:text="動作" />
```

```
</LinearLayout>
```

```
</LinearLayout>
```

```
<LinearLayout
    android:id="@+id/linearLayout5"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    app:layout_constraintTop_toBottomOf="@+id/linearLayout4"
```

```

tools:layout_editor_absoluteX="-16dp">
<LinearLayout
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:gravity="center_horizontal"
    android:orientation="vertical">

    <ImageView
        android:id="@+id/ImageView3"
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:scaleType="centerCrop"
        app:srcCompat="@drawable/confuse" />
    <CheckBox
        android:id="@+id/checkbox3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center"

        android:text="懸疑" />

</LinearLayout>
<LinearLayout
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:gravity="center_horizontal"
    android:orientation="vertical">
    <ImageView
        android:id="@+id/ImageView4"
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:scaleType="centerCrop"
        app:srcCompat="@drawable/like" />
    <CheckBox
        android:id="@+id/checkbox4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center"

        android:text="愛情" />

```

```

        </LinearLayout>
    </LinearLayout>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        app:layout_constraintTop_toBottomOf="@+id/linearLayout5"
        tools:layout_editor_absoluteX="73dp">
        <LinearLayout
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:gravity="center_horizontal"
            android:orientation="vertical">
            <ImageView
                android:id="@+id/ImageView5"
                android:layout_width="100dp"
                android:layout_height="100dp"
                android:scaleType="centerCrop"
                app:srcCompat="@drawable/funny" />
            <CheckBox
                android:id="@+id/checkbox5"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_gravity="center"

                android:text="喜劇" />

        </LinearLayout>
    <LinearLayout
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:gravity="center_horizontal"
        android:orientation="vertical">
        <ImageView
            android:id="@+id/ImageView6"
            android:layout_width="100dp"
            android:layout_height="100dp"
            android:scaleType="centerCrop"
            app:srcCompat="@drawable/scare" />
        <CheckBox

```

```

        android:id="@+id/checkbox6"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center"

        android:text="恐怖" />

    </LinearLayout>
</LinearLayout>
<Button
    android:id="@+id/btnChoose"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginBottom="4dp"

    android:text="送出"

    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>

```

▼表 9-2-4 規格功能描述-fragment_user.xml

程式名稱	fragment_user.xml
目的	個人資料畫面，並讓用戶可以變更密碼、變更大頭貼、查看評論紀錄和登出。
部分程式碼	
<pre>//----- <LinearLayout android:layout_width="match_parent" android:layout_height="match_parent" android:orientation="vertical"> <LinearLayout android:layout_width="match_parent" android:layout_height="wrap_content" android:layout_marginBottom="100px" android:animateLayoutChanges="false" android:orientation="horizontal"> <ImageView android:id="@+id/imageView" android:layout_width="250px" android:layout_height="500px" android:layout_weight="1" android:src="@drawable/head" /> <TextView android:id="@+id/textView" android:layout_width="wrap_content" android:layout_height="wrap_content" android:layout_gravity="center" android:layout_weight="1" android:textAlignment="center" /> </LinearLayout></pre>	

```
<Button
    android:id="@+id/button4"
    android:layout_width="match_parent"
    android:layout_height="150px"
    android:layout_marginBottom="50px"

    android:text="變更使用者照片" />
```

```
<Button
    android:id="@+id/button3"
    android:layout_width="match_parent"
    android:layout_height="150px"
    android:layout_marginBottom="50px"

    android:text="變更密碼" />
```

```
<Button
    android:id="@+id/button5"
    android:layout_width="match_parent"
    android:layout_height="150px"
    android:layout_marginBottom="50px"

    android:text="評論紀錄" />
```

```
<Button
    android:id="@+id/logout"
    android:layout_width="match_parent"
    android:layout_height="150px"

    android:text="登出" />
```

```
</LinearLayout>
```

```
</FrameLayout>
```

第十章 測試模型

10-1 測試計畫

1. 會員功能：

(1) . 註冊會員：確認使用者註冊是否成功。

(2) . 登入會員：確認使用者登入是否成功。

2. 興趣功能：

(1) . 興趣選擇：確認使用者是否可以顯示並能夠點選。

3. 主頁功能：

(1) . 主頁：確認主頁是否抓取資料庫成功。

(2) . 下方功能列：確認功能列是否可以顯示並能夠點選。

4. 電影介紹：

(1) . 電影資訊：確認電影資訊是否可以顯示並能夠點選。

(2) . 留言按鈕：確認按鈕是否可以顯示並能夠點選跳轉到留言頁面。

5. 留言功能：

(1) . 留言：確認使用者是否能成功留言並上傳。

6. 個人檔案功能：

- (1) . 我的個人檔案頁面：確認我的個人檔案是否抓取成功。
- (2) . 編輯個人檔案：確認是否能編輯個人檔案。
- (3) . 更換大頭貼：確認是否能更換大頭貼。
- (4) . 修改密碼：確認是否能修改密碼。
- (5) . 更換大頭貼：確認是否能更換大頭貼。
- (6) . 登出：確認是否能登出。

10-2 測試個案與測試結果資料

▼表 10-2-1 註冊會員測試個案與測試結果資料

測試編號	1
功能名稱	註冊會員
測試目的	確認使用者註冊是否成功
測試流程	輸入帳號、密碼並送出
預期結果	註冊成功
測試成果	正常

▼表 10-2-2 登入會員測試個案與測試結果資料

測試編號	2
功能名稱	登入會員
測試目的	確認使用者登入是否成功
測試流程	輸入帳號、密碼並送出
預期結果	登入成功
測試成果	正常

▼表 10-2-3 興趣選擇測試個案與測試結果資料

測試編號	3
功能名稱	興趣選擇
測試目的	確認使用者是否可以顯示並能夠點選
測試流程	選擇有興趣的主題點選後並送出
預期結果	點選成功
測試成果	正常

▼表 10-2-4 主頁功能測試個案與測試結果資料

測試編號	4
功能名稱	主頁功能
測試目的	確認主頁是否抓取資料庫成功
測試流程	進入主頁面後有顯示出資料庫之電影
預期結果	顯示成功
測試成果	正常

▼表 10-2-5 下方功能列測試個案與測試結果資料

測試編號	5
功能名稱	下方功能列
測試目的	確認功能列是否可以顯示並能夠點選
測試流程	有顯示出功能列並可以點選至跳轉頁面
預期結果	顯示並點選成功
測試成果	正常

▼表 10-2-6 電影資訊測試個案與測試結果資料

測試編號	6
功能名稱	電影資訊
測試目的	確認電影資訊是否可以顯示並能夠點選
測試流程	有顯示出電影資訊並可以點選
預期結果	顯示並點選成功
測試成果	正常

▼表 10-2-7 留言按鈕測試個案與測試結果資料

測試編號	7
功能名稱	留言按鈕
測試目的	確認按鈕是否可以顯示並能夠點選跳轉到留言頁面
測試流程	有顯示出留言按鈕並可以點選
預期結果	顯示並點選成功
測試成果	正常

▼表 10-2-8 留言測試個案與測試結果資料

測試編號	8
功能名稱	留言
測試目的	確認使用者是否能成功留言並上傳
測試流程	留言輸入後能夠新增留言
預期結果	留言成功
測試成果	正常

▼表 10-2-9 我的個人檔案頁面測試個案與測試結果資料

測試編號	9
功能名稱	我的個人檔案頁面
測試目的	確認我的個人檔案是否抓取成功
測試流程	進入個人檔案後有顯示出個人檔案之資料
預期結果	顯示成功
測試成果	正常

▼表 10-2-10 編輯個人檔案測試個案與測試結果資料

測試編號	10
功能名稱	編輯個人檔案
測試目的	確認是否能編輯個人檔案
測試流程	點選編輯個人檔案
預期結果	編輯成功
測試成果	正常

▼表 10-2-11 修改密碼測試個案與測試結果資料

測試編號	11
功能名稱	修改密碼
測試目的	確認是否能修改密碼
測試流程	點選修改密碼
預期結果	修改成功
測試成果	正常

▼表 10-2-12 更換大頭貼測試個案與測試結果資料

測試編號	11
功能名稱	更換大頭貼
測試目的	確認是否能更換大頭貼
測試流程	點選更換大頭貼後可以更換
預期結果	更換成功
測試成果	正常

▼表 10-2-13 登出測試個案與測試結果資料

測試編號	11
功能名稱	登出
測試目的	確認使用者是否能登出
測試流程	點選登出
預期結果	登出成功
測試成果	正常

第十一章 操作手冊

介紹系統之元件及其安裝及系統管理

▼表 11-1 系統安裝元件資訊表

系統安裝元件資訊	
元件名稱	A movie I recommend
版本	1.0
檔案大小	14.2 MB
軟體類型	娛樂應用
支援語言	繁體中文
價格	免費
最低版本需求	Android 10
內容分級	6 歲以上
需要權限	查看網路連線、接收網路資料

▼表 11-2 系統安裝步驟

系統安裝步驟	
	<p>掃描 QR Code 並下載 apk 檔案。</p> <p>點選同意許可進行安裝。</p> <p>等待安裝約 30 秒。</p> <p>完成後點擊 icon 即可使用。</p>

系統管理

1. 用戶管理:

- (1) . 用戶註冊和登入: 實現一個安全的用戶註冊和登入系統, 包括電子郵件驗證或其他身份驗證機制。
- (2) . 用戶個人檔案: 讓用戶能夠設置和管理他們的個人檔案, 包括個人信息、喜好設置等。

2. 電影數據管理:

- (1) . 電影數據庫: 維護一個完整的電影數據庫, 包括電影信息、演員、導演、類型、評分等。
- (2) . 數據更新: 定期更新電影數據, 包括新增的電影、刪除的電影和評分等。

3. 推薦算法管理:

模型訓練和更新: 定期訓練推薦模型, 並根據新數據更新模型參數。

4. 安全性和隱私:

- (1) . 用戶數據安全: 確保用戶數據的安全, 包括加密敏感信息、定期備份數據等。
- (2) . 隱私政策: 提供清晰的隱私政策, 告知用戶他們的數據如何被使用, 並獲取必要的同意。

第十二章 使用手冊

▼表 12-1 系統使用說明-登入與註冊

登入與註冊	
	第一次的使用者需點選「註冊」按鈕進行註冊作業。
	進入註冊畫面，設定好完整資料，按下下方按鈕即完成註冊。

10:27 112208



帳號

請輸入帳號

密碼

請輸入密碼



登入 註冊

註冊完成後回到登入畫面，輸入
帳號密碼即可登入。

▼表 12-2 系統使用說明-興趣選擇

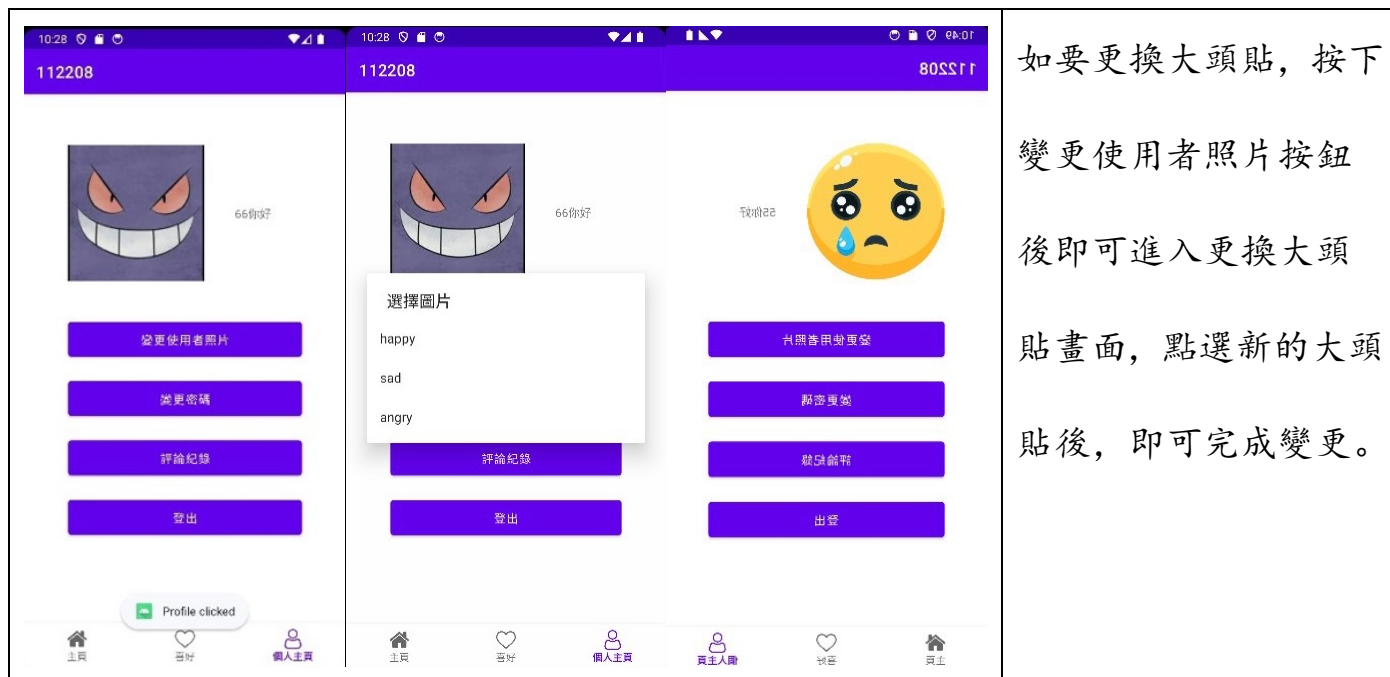
興趣選擇	
	<p>第一次的使用者須點選有興趣的主題。</p>
	<p>選擇送出後，進入主畫面，主畫面會顯示興趣相關的電影。</p>

▼表 12-3 系統使用說明-觀看電影資訊並評分留言

觀看電影資訊並評分留言	
 <p>10:30 112208</p> <p>電影名稱：復仇者聯盟 電影類型：動作 簡介：這是一部電影</p> <p>評分 推薦</p>	<p>選擇想看的電影並點選後，會顯示該電影的相關資訊，按下評分按鈕即可進入評分留言系統。</p>
 <p>10:46 112208</p> <p>輸入留言</p> <p>提交 返回</p> <p>好好看哦</p> <p>123</p> <p>good</p>	<p>進入評分系統後，留下使用者的真實感受，並點選提交即可送出留言。</p>

▼表 12-4 系統使用說明-個人資料修改

個人資料修改	
	<p>回到主畫面後，點選下方工具列最右邊的按鈕即可進入個人資料畫面。</p>
	<p>如要變更密碼，按下變更密碼按鈕後即可進入變更密碼畫面，輸入新密碼並送出後，即可完成變更。</p>



▼表 12-5 系統使用說明-登出

登出	
	<p>回到個人資料畫面後，點選最下方「登出」按鈕後即可登出帳號。</p>

第十三章 感想

11136025 馬振閔

這次擔任組長一職讓我受益良多，我在組長的角色中學到了領導和溝通的重要性。有效的問題解決以及鼓勵團隊成員發揮創造力，都是領導者應該具備的素質。這次專題中，我們的團隊遇到了許多困難，其中包括了組員的離開跟組員中的程度差異，面對組內技術水平的不同，我學會了如何評估每個組員的能力，並合理分配任務，使得整個團隊能夠協同合作。此外，積極的溝通和有效的反饋機制有助於減少誤解，增進組內成員之間的信任。

最終，雖然這個 App 還有許多進步空間，但這個是我們努力一年來的心血，不僅讓我們學到了實際的技術和專業知識，更鍛煉了我們的團隊協作和解決問題的能力。

最令我感到自豪的是，這次專題的成功更體現在整個團隊的成長和凝聚力上。組內的每個成員都在這個過程中取得了成就感，這種共同努力的感覺是無法用言語形容的。這次的專題經驗對我個人和整個團隊都是一個難得的學習機會，我相信這將對我們未來的事業道路產生深遠的影響。

在這次專題中，雖然我在組內的程度相對較差，但我總是盡力以最大的熱忱參與專案的各個階段，並且這次的經驗讓我收穫良多。雖然我的技術程度相對較低，但大家都願意互相學習和分享，也很謝謝，這種團隊協作的氛圍讓我感到非常溫馨。

在過程中，我學習到了一些新的技能，並且嘗試參與了一些我之前不太熟悉的任務。雖然過程中遇到了一些困難，但我總是能夠得到組內其他成員的幫助和鼓勵。這讓我深刻體會到在一個協作的環境中，每個人的努力都是不可或缺的。

總之，這次的專題經驗是一個充滿挑戰和收穫的過程。我深刻體會到在一個團隊中，每個人都有其獨特的價值，而我的努力和貢獻也在這個協作的過程中得到了認可。這次的經驗對我的未來學習和職業發展都將產生積極的影響，而努力和積極參與是實現個人成長的不二法門！

這次的專題使我深入體會軟體開發的挑戰與樂趣，同時也習得了眾多實用的技能和知識。

在這次的專題中，我領悟到如何有效展開團隊合作的重要性。和隊友一同參與 APP 的設計、開發和測試，必須建立起良好的溝通和協作機制。我們學會了如何巧妙分工、解決問題，以及如何在不同意見中達成共識。這樣的合作經驗對我未來的職業生涯將會是相當寶貴的資產。

這次的經驗更深刻地讓我了解軟體開發的流程。從需求分析、設計、編碼，一直到測試和上線，每一個階段都帶來獨特的挑戰和重要性。我學到了如何有效運用開發工具，進行測試和除錯，同時確保程式的穩定性。還加深了我對於用戶需求和體驗的認識。在設計和開發的過程中，我們持續以使用者的視角思考，不斷優化和改進 APP。

總的來說，這次的 APP 專題是一段充實而有趣的歷程。儘管遇到了不少困難，但我們齊心協力克服挑戰，最終呈現出一個令人滿意的作品。這次的經驗讓我更有信心和能力迎接未來在軟體開發領域的各種挑戰。

第十四章 參考資料

- Kotlin 相關使用：

<https://www.youtube.com/channel/UCP7uiEZIqci43m22KDl0sNw>

- android studio kotlin fragments：

<https://givemepass.blogspot.com/2013/09/fragment.html>

- android studio kotlin recyclerview

<https://developer.android.com/guide/topics/ui/layout/recyclerview?hl=zh-tw>

- 圖片素材：

<https://unsplash.com/>

- Python Flask 教學：

<https://devs.tw/post/448>

- Python Flask Restful API：

<https://vocus.cc/article/626f67e4fd89780001be4f3a>

- 繪圖工具：

<https://app.diagrams.net/>

- SQLite 資料庫操作：

<https://nkust.gitbook.io/python/sqlite-liao-cao-zuo-jie>

- 如何在 Flask 使用 SQLite 資料庫

<https://medium.com/pyladies-taiwan/%E5%A6%82%E4%BD%95%E5%9C%A8-flask-%E4%BD%BF%E7%94%A8-sqlite-%E8%B3%87%E6%96%99%E5%BA%AB-c26f300f1d87>

- 在 Flask 中使用 SQLite 3

<http://www.pythondoc.com/flask/patterns/sqlite3.html>

附錄

▼附錄-初評評審意見之修正情形

評審建議事項	修正情形
如何推薦？協同過濾？Content Base？ User Base？知識圖譜？	依照使用者的興趣以及星等高低做推薦
與其他平台的差異在哪？	目前努力仿製其他影音平台，未來有機會做出更有特色的差異
沒使用 Github 紀錄	之後有固定每個禮拜都有開會 4 小時以上，並每週上傳專題進度
本案分工不佳，僅由一人完成程式開發，相對辛苦	經過評審的指教以及指導老師的討論，組長重新分配了工作分配，面對組內技術水平的不同，評估每個組員的能力，並合理分配任務，使得整個團隊能夠協同合作
專題比較像是數據分析案，實用性不足，簡報缺乏架構的細述，可再加強	有重新思考整體架構和程式流程
有關 AI 及 NLP 如何應用，應多加說明	逐字分析使用者評論內容的前後文，給定評分，經過全部評分的加權，給定該電影的星等數

▼附錄-複評評審意見之修正情形

評審建議事項	回覆

▼附錄-海報評審意見

評審建議事項	回覆