# **漫畫租借管理系統-小專題**

班級：四資工三乙  
成員：陳家龢 B11170047、陳泓維 B11170075

## **第一章 緒論**

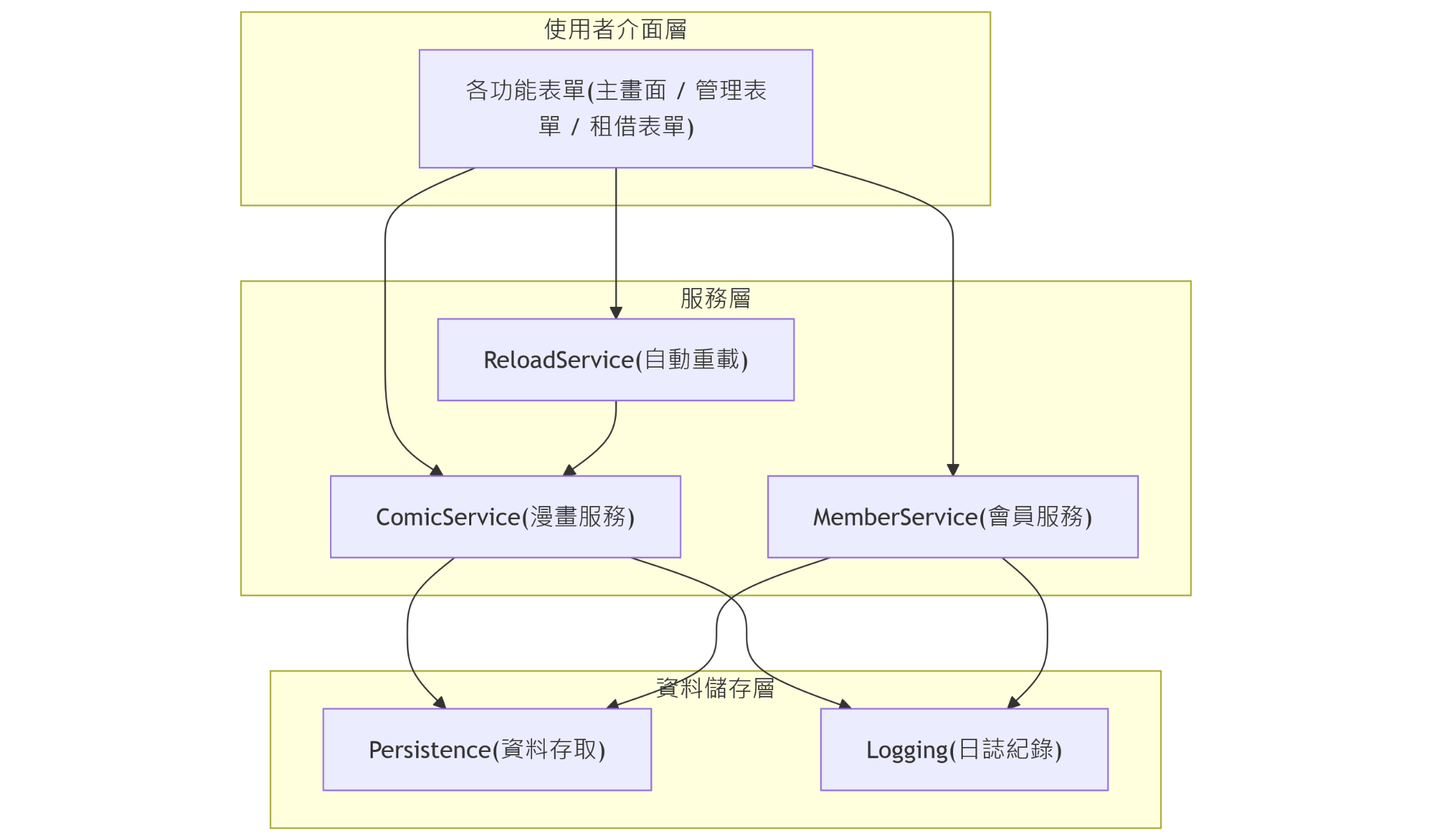
**研究背景與動機：** 隨著漫畫租借業務的普及，傳統以紙本或試算表管理漫畫庫存與租借紀錄的方式日益吃力，容易發生遺漏或錯誤。尤其當館藏量與會員數增加時，手動追蹤哪本漫畫被哪個會員借走、何時該歸還等資訊相當繁瑣。本專題的動機在於運用所學的 C# 程式開發知識，打造一套**漫畫租借管理系統**，協助業者數位化管理漫畫書目與會員資訊，提升租借流程效率與準確性。此外，專案也希望在實作過程中綜合運用課程第8章至第16章介紹的各項進階語言技術，作為學習成果的驗證。

**研究目的：** 本專題旨在開發一個**Windows Forms 桌面應用程式**，具備漫畫書籍資料維護、會員資料維護，以及漫畫租借與歸還管理等核心功能。系統將提供使用者友善的圖形介面，方便館員輸入或查詢漫畫與會員資訊，並能快速執行租借、歸還操作，同時自動記錄相關日誌。研究中將針對如何運用物件導向概念來設計系統架構，以及如何透過事件驅動的方式保持 UI 與資料一致性進行探討，滿足功能需求的同時確保系統的穩定性。

**問題陳述：** 傳統管理方式下，常見問題包括：漫畫庫存資料量大且更新不易、無法及時掌握漫畫是否已被租借狀態、會員借閱紀錄不明確等。沒有適當的系統支援，館員需要花費大量時間在重複性工作上，例如手動查找某會員是否已借某本漫畫。缺乏即時同步機制也可能導致多處資料不一致。本系統針對這些問題提出解決方案：集中管理漫畫與會員資料，透過**檔案資料庫**維護狀態，並在租借或歸還時即時更新狀態並反映於介面，避免資訊落差。另外，針對使用過程可能出現的異常情況（如資料格式錯誤等）提供例外處理機制，提高系統可靠度。

**預期貢獻：** 本專題預期的貢獻有兩方面：（1）**實務應用**：提供一套簡易但完整的漫畫租借管理解決方案，適合小型漫畫出租店或個人漫畫收藏管理使用。透過本系統，管理人員可以大幅減少手工記錄的工作量，快速查詢館藏並追蹤每本漫畫的租借情況。（2）**教育意義**：對開發團隊而言，此專案是課程知識的統整實踐，涵蓋從資料結構、物件導向設計、介面開發到檔案處理、事件驅動、執行緒應用等多項技術點。專案報告中將詳細說明這些 C# 技術在專案中的應用情形，作為學習成果的展示。總體而言，透過本專題的實現，不僅解決了漫畫租借管理的實際需求，也驗證了團隊對 C# 語言與應用開發的掌握度。

## **第二章 系統設計**

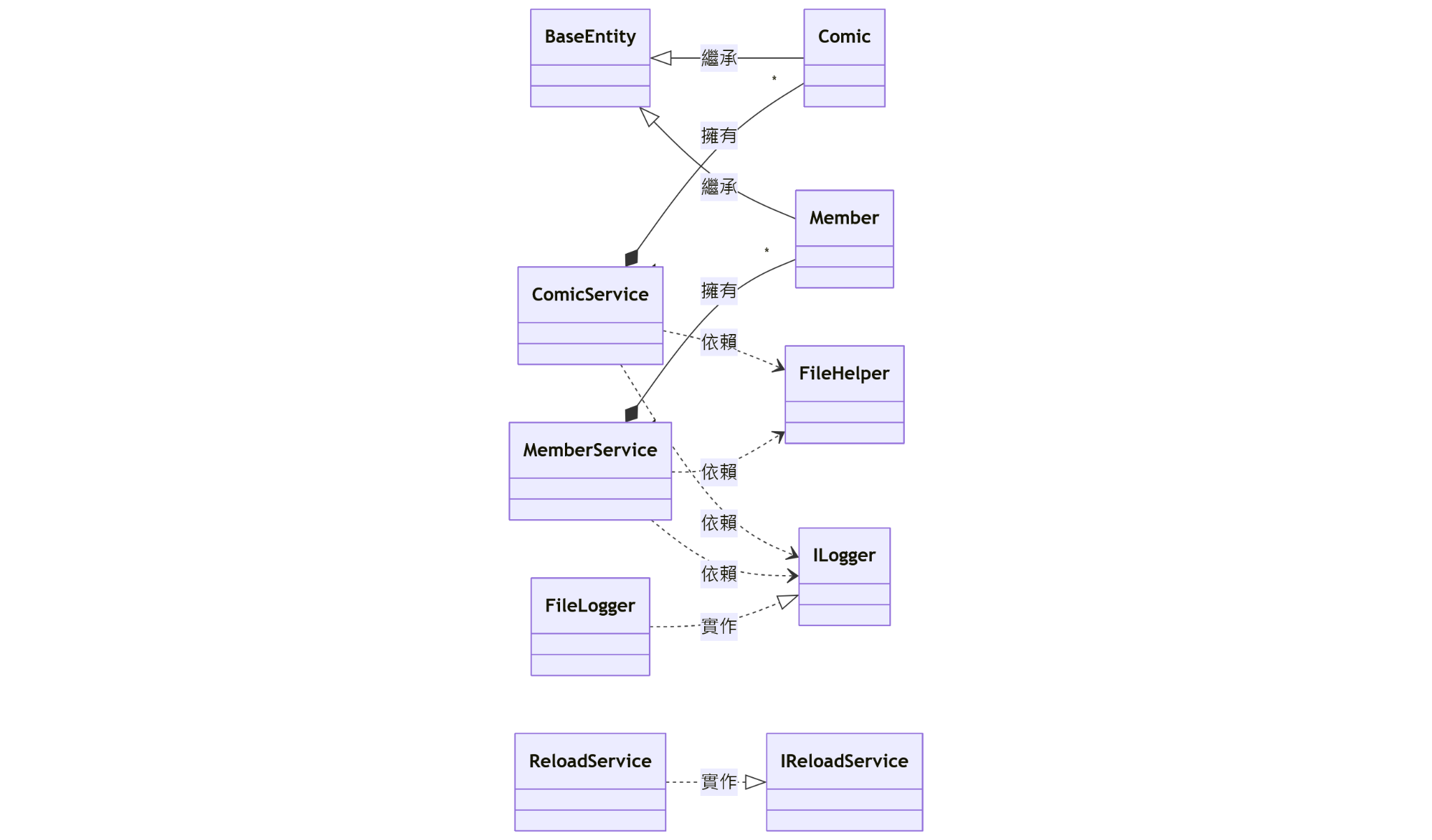
**系統架構圖(經抽象化處理)：** 

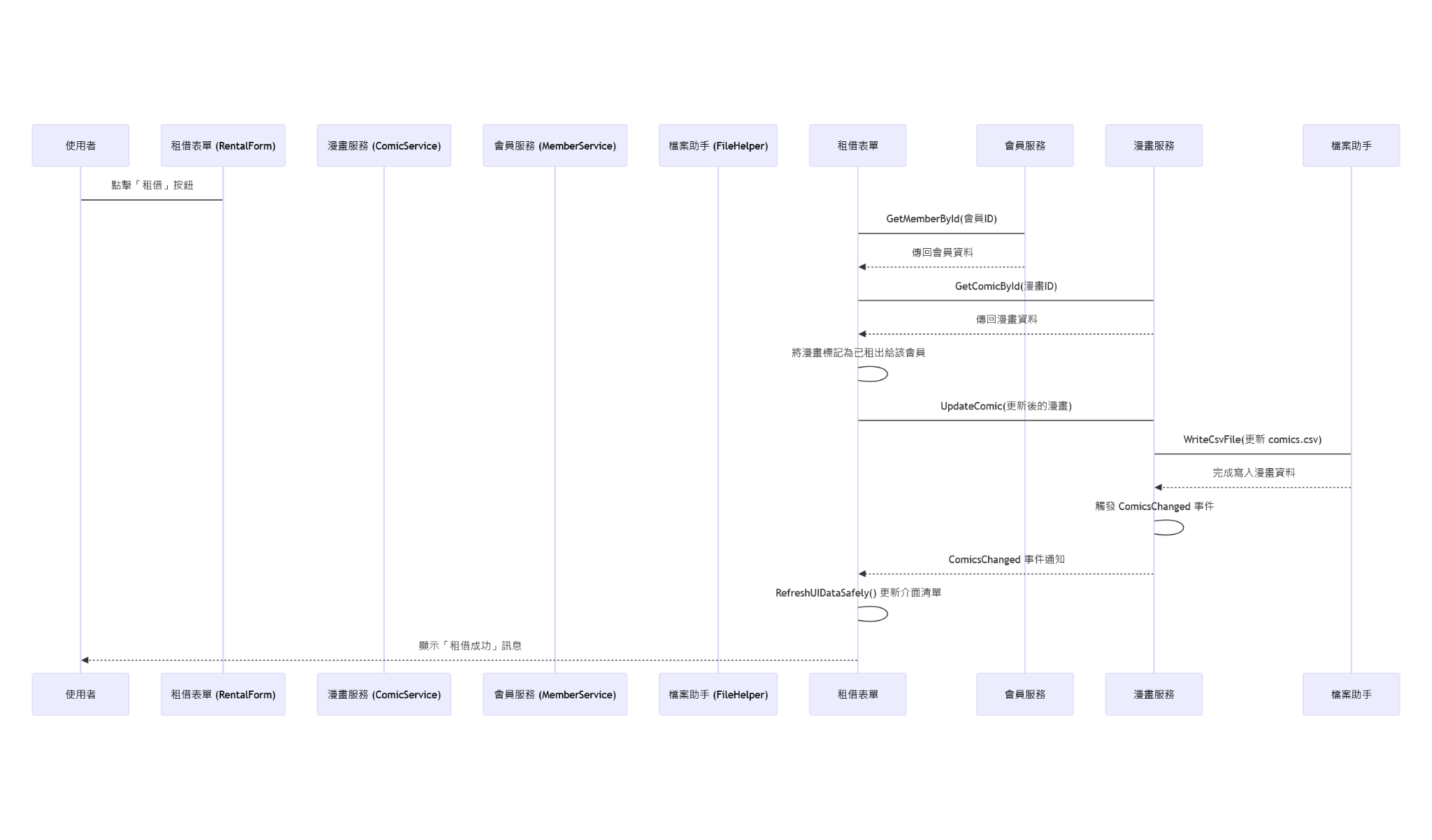
**功能模組說明：**

* **漫畫管理模組：** 提供新增、編輯、刪除漫畫書目資料的功能。在**漫畫管理**表單中，系統以資料表格列出所有漫畫及其屬性（書名、作者、類型、ISBN 等），點選某筆資料可進行編輯或刪除。新增或編輯時則跳出漫畫編輯子表單供使用者輸入資料，按下儲存後調用 ComicService 的對應方法寫入檔案並更新列表。
* **會員管理模組：** 功能類似漫畫管理，用於維護會員資料（姓名、電話）。在會員管理表單中顯示會員列表並允許新增、修改、刪除會員。例如按下「新增會員」按鈕會開啟會員編輯表單，填寫完畢按下儲存即呼叫 MemberService.AddMember 新增記錄並更新顯示。
* **租借管理模組：** 此為系統核心模組，處理漫畫的租借與歸還。使用者在**租借管理**表單中先選擇一位會員，再從可用漫畫清單中選取要租借的漫畫。點擊「租借」按鈕後，系統會檢查輸入是否合法及漫畫當前狀態；若可租借，則將該漫畫標記為已租出並記錄租借人，然後更新檔案與介面。相反地，「歸還」操作會將所選漫畫的租借狀態取消。租借表單上半部使用下拉選單列出所有會員，供選擇承租人；下半部則以表格顯示該會員已租借的漫畫清單。為了即時反映狀態變化，租借表單會在載入時啟動一個背景重新整理服務，定期自檔案重新載入資料。
* **資料儲存與日誌模組：** 系統將漫畫與會員資料分別儲存在 comics.csv 和 members.csv 檔案中。每個 CSV 檔以純文字儲存記錄，每行代表一筆資料，各欄位以逗號分隔。以漫畫檔案為例，欄位順序為 Id, Title, Author, Isbn, Genre, IsRented, RentedToMemberId。我們設計了 FileHelper 類別作為資料存取層，提供泛型方法 ReadCsvFile<T> 及 WriteCsvFile<T> 來讀寫 CSV 檔。在系統初始化時，資料檔案將被讀入並轉換為物件清單，以供服務層使用；在每次資料異動（新增/修改/刪除）後，服務層也會呼叫寫檔方法，確保檔案內容與記憶體資料同步。另外，系統具有日誌紀錄功能：實作 ILogger 介面提供日誌記錄介面，再由 FileLogger 將重要操作與錯誤記錄到文字檔（例如 ComicRentalSystemLog.txt）中。主選單中的「檢視日誌」功能會開啟該日誌檔供使用者檢閱。
* **事件與更新機制：** 為確保多表單之間資料一致，系統廣泛採用事件機制。漫畫服務與會員服務內定義了資料變更事件（如 ComicsChanged、MembersChanged）。當資料操作（新增、修改、刪除）發生後，服務層即觸發相應事件，通知註冊的介面更新。例如主畫面和管理表單在載入時會訂閱 ComicsChanged 事件，一旦有漫畫資料異動便自動重新載入清單。這種設計提高了使用體驗：使用者在子視窗完成編輯回到主列表時，變更立即可見，無需手動刷新。

**資料庫設計（如有）：** 本系統未採用傳統關聯式資料庫，而是以**CSV 純文字檔**作為資料存儲後端。一方面，這符合小型專案的需求，簡化部署而無需額外的資料庫伺服環境；另一方面，此設計也讓專案重點聚焦在 C# 檔案處理及資料序列化等課程重點上。CSV 資料檔案統一存放在使用者文件夾下的專案資料夾中（例如 *我的文件/ComicRentalApp/AppData/*）。在系統啟動時，程式會檢查該資料夾及檔案是否存在，若無則自動建立空白檔案，避免因檔案不存在導致讀取錯誤。每筆漫畫或會員記錄均包含唯一的整數 ID 作為主鍵；系統在新增資料時會由服務層自動生成遞增的 ID。由於 CSV 缺乏強類型約束，專案在解析文字時採取了一定的防錯措施，例如計算欄位數不足時拋出格式錯誤、對字串欄位去除額外引號等。總體而言，此資料庫設計雖然不如關聯式資料庫健全，但在專案範圍內運作良好；未來若需求擴大，可考慮改用 SQL 資料庫來增強資料一致性與查詢效率（參見第五章之未來工作部分）。

**類別圖、循序圖等 UML 圖表：**

****

****

## **第三章 系統實現**

**開發環境：** 本專案使用 **Visual Studio 2022** 作為開發工具，目標框架為 **.NET 8.0 (Windows 桌面)**。採用 C# 程式語言，利用 Windows Forms 建置使用者介面。在 .NET 8 平台下，專案以單執行檔形式發行（WinExe），無需額外安裝資料庫或伺服器。開發與測試皆在 Windows 10 作業系統上進行。Visual Studio 內建的 WinForms 設計工具用於繪製表單介面，將各種控制項（按鈕、資料網格、選單、文字框等）擺放佈局，並透過事件屬性關聯至程式碼中的事件處理函式。例如，在設計階段為按鈕設定 Click 事件處理程序，在程式碼裡即產生對應的函式骨架供實作。整體專案結構清晰，程式碼檔案依功能歸類於資料模型（Models）、服務（Services）、表單（Forms）、工具類別（Helpers）和記錄（Logging）等資料夾中。

**核心功能代碼解析：** 本節從關鍵功能出發，解析系統實作細節：

* **資料序列化與反序列化：** 系統使用自定義的 ToCsvString() 和 FromCsvString() 方法實現物件與 CSV 行之間的轉換。在 Comic 類別中，ToCsvString 將一個 Comic 物件序列化為逗號分隔的字串；而靜態的 Comic.FromCsvString 則將讀入的字串切割後轉換回 Comic 物件。程式利用字串函式和陣列操作來達成這點，例如使用 String.Split(',') 將一行文字拆解為欄位陣列。為了處理欄位中可能包含逗號或引號的情況，代碼中對引號做了替換和去除等前處理。這些函式確保了在檔案與記憶體之間交換資料時的準確性。
* **增刪改查（CRUD）流程：** 以**新增漫畫**為例，當使用者在漫畫編輯表單輸入資料並點擊保存時，系統在後端執行如下步驟：先建立一個新的 Comic 物件並填入表單上的欄位值；接著呼叫 ComicService.AddComic(newComic) 方法。AddComic 內部會先為該物件指派新的流水號 ID，然後將物件加入內存中的漫畫清單並寫入檔案。成功寫檔後，服務觸發 ComicsChanged 事件通知 UI 更新。相似的模式也運用在更新（Edit）與刪除（Delete）操作上：例如編輯漫畫時，先在服務層找到目標物件並更新其屬性，再寫檔保存；刪除則從清單移除物件後寫檔。透過服務層統一處理這些操作，確保所有異動都能正確保存且日誌記錄完整。
* **租借/歸還流程實作：** 在租借表單中，為達成漫畫租借功能，程式於「租借」按鈕事件中執行多項檢查與更新。首先取得下拉選單中選定的會員與漫畫之 ID，並透過服務層分別取回對應的 Member 物件與 Comic 物件。接著確認會員與漫畫是否有效（例如資料可能在此時被刪除），以及漫畫是否未被租出。若檢核通過，則設置漫畫物件的 IsRented = true 並將 RentedToMemberId 設為選定會員的 ID。隨後呼叫 ComicService.UpdateComic 更新該漫畫的狀態。此過程包裹在 try-catch 中以捕捉可能發生的例外，如在更新檔案時出錯等。歸還的邏輯類似，差別在於將選定漫畫的 IsRented 重置為 false，RentedToMemberId 清0，然後執行更新。租借與歸還的操作結果都會顯示對應的 MessageBox 提示使用者完成情形（成功或錯誤）。值得一提的是，租借表單在載入時啟用了 ReloadService 每秒自動重新載入資料的功能，配合服務層的事件機制，確保即使使用者長時間開啟租借視窗，資料變化（例如其他地方刪除了某本漫畫）也能及時反映。
* **介面更新與執行緒處理：** 為了防止跨執行緒更新 UI 的錯誤，程式在實作事件處理或背景執行時，使用 InvokeRequired 判斷當前執行緒是否為 UI 執行緒。例如，當 ComicService.ComicsChanged 事件從非 UI 執行緒（如背景 Task）觸發時，主畫面的事件處理函式會發現 this.InvokeRequired == true，於是透過 Invoke(...) 將 UI 更新委派回主執行緒執行。這種設計避免了執行緒同步問題，保證對資料網格等控制項的更新操作在正確的執行緒上進行。此外，應用程式在啟動時設定了全域的例外處理器，用於攔截任何未處理的執行緒例外。一旦有未捕捉的錯誤冒泡，系統會透過這些處理器記錄日誌並彈出錯誤提示，而不中斷整個應用。

**界面設計與使用者操作流程：** 系統採用傳統 Windows Forms 介面風格，主視窗包含上方選單列和主要資訊區域。選單提供了「漫畫管理」、「會員管理」、「租借管理」等功能入口，使用者點擊後會開啟對應的功能視窗。主視窗同時也即時顯示目前**可供出租**的漫畫列表，方便館員快速瀏覽館內尚未被租出的漫畫。列表使用 DataGridView 控制項實現，可自動將資料物件集合繫結並顯示為表格。其欄位設定在程式中動態建立，包括書名、作者、類型、ISBN 等欄位，並設定為唯讀以避免直接編輯。DataGridView 控制項提供了高度客製化的資料呈現表格，支援資料繫結、欄位格式化、欄寬自動調整等功能。在漫畫/會員管理視窗中，同樣使用 DataGridView 顯示所有資料列表並禁用多選，以確保一次只針對單筆記錄操作。列表下方則有新增、編輯、刪除、重新載入等操作按鈕。使用者透過點擊「新增」開啟編輯子窗，若在該子窗點擊「確認」，會以模式對話框關閉返回並更新主窗體列表；點擊「編輯」或雙擊資料列則開啟載入了選定資料的編輯窗以進行修改。刪除按鈕按下時，系統會先跳出 MessageBox 二次確認使用者意圖。整體操作流程符合直覺：使用者在各模組界面透過按鈕與清單完成對資料的管理，系統則在背後完成相應邏輯，並以視覺化的方式立即反映結果（如列表內容變化、提示對話框等），達到良好的互動體驗。

**運用之 C# 概念與技術：** 本專案在開發過程中廣泛運用了課程第8章至第16章介紹的 C# 語言功能與模組，歸納如下：

* **字串與陣列：** 專案大量使用字串處理與陣列操作來實現資料序列化。例如將漫畫物件轉為 CSV 時使用字串插值與 Replace 函式處理特殊字元；從 CSV 讀取時使用 Split 將一行文字拆解為字串陣列。再透過陣列索引取出各欄位並轉換型別生成物件。這些操作充分運用了字串與陣列的基礎功能，實現文本資料與物件之間的轉換。
* **類別與物件：** 系統採用物件導向方法設計，將漫畫、會員等概念抽象為類別，並透過類別來產生物件實例以保存資料與執行操作。例如，我們定義了 Comic 和 Member 類別來表示對應資料模型，在程式中新增一筆漫畫時即建立一個 Comic 物件並設定其屬性。同時也設計了服務類別如 ComicService 來封裝對漫畫物件的各種操作。整個專案透過創建和操控這些類別物件來完成任務，體現了類別與物件的核心概念。
* **繼承與介面：** 繼承方面，Comic 與 Member 類別繼承自共同的基底類別 BaseEntity，從而共用 Id 欄位與未來可能擴充的共同行為。如此一來，可在需要處理泛型資料時透過 BaseEntity 型別引用。如有需要也能擴大共通性。介面方面，專案定義了 ILogger 介面宣告日誌功能，包含多載的 Log 方法和 LogError 方法。FileLogger 類別實作了此介面，提供將日誌寫入檔案的具體功能。由於介面實現，我們可以在程式中以 ILogger 型別來使用具體的 FileLogger 物件，達到程式碼的鬆耦合和**替換多型**效果（例如未來若改為其他紀錄方式，只需替換另一個實作了 ILogger 的類別）。此外，IReloadService 介面同樣應用在背景重載服務的設計上，以規範 Start/Stop 方法並利於擴充。
* **過載與多型：** 在方法過載方面，ILogger 介面中定義了兩種簽名的 Log 方法（一個純訊息，另一個帶 Exception 參數）以及一個 LogError 方法。FileLogger 類別對這些方法進行了實作，其中包含對 Log 方法的過載實現和對 LogError 方法使用可選參數的過載。由於我們經常以介面型別來操作 Logger（例如 BaseForm 中持有 ILogger 的引用），實際運行時會根據傳入物件類型呼叫對應的實作，這展現了**多型**的威力。當透過 ILogger 介面呼叫 Log 時，實際執行的是 FileLogger 的對應方法。同樣地，服務類別的方法如 GetAllComics() 回傳 List<Comic>，在綁定到 DataGridView 時，.NET 也會多型地調用每個 Comic 物件的屬性存取器來顯示內容。這些都屬於多型的應用。
* **例外處理、委派與執行緒：** 為了提高系統健壯性，專案廣泛地使用了**例外處理**機制。在可能發生錯誤的地方（例如解析 CSV、檔案讀寫、資料庫操作等），都以 try-catch 包裝，針對特定例外型別給出提示或補救動作。同時在全域設定未捕捉例外處理器，防止程式異常崩潰。委派與事件的運用是本專案的一大特色：我們透過宣告委派類型（如 ComicService.ComicDataChangedEventHandler）並搭配 event 關鍵字建立事件。介面上由表單註冊這些事件的處理函式，從而實現鬆耦合的通知機制。例如，一旦漫畫資料清單改變，服務內會透過 ComicsChanged?.Invoke 呼叫委派，觸發表單端訂閱的更新函式執行資料繫結刷新。至於**執行緒**，專案利用 .NET 的 Task 類別開啟背景執行緒來處理定時任務（自動刷新）。使用 Task.Run(...) 將工作排入執行緒池，使週期任務在與 UI 不同的平行執行緒執行，不阻塞主執行緒。Task.Run 方法會將指定的工作委派至執行緒池線程並立即返回一個 Task 物件。本系統即利用此特性，每隔固定時間由背景執行緒呼叫服務的重新載入方法，再透過事件驅動更新 UI，達到簡單的多執行緒應用效果。
* **視窗應用程式的事件處理：** Windows Forms 採用事件驅動模型，專案充分運用這一點來響應使用者操作。例如，每個按鈕、選單項、下拉框等都綁定了事件處理函式：如「漫畫管理」選單點擊事件中打開漫畫管理表單、「新增漫畫」按鈕點擊事件中開啟 ComicEditForm、DataGridView 的 CellDoubleClick 事件中觸發編輯操作等等。這些處理函式在 Visual Studio 設計階段透過屬性窗連結，實際在程式碼中以 object sender, EventArgs e 作為參數的方法實現。在處理函式內部，我們可以調用其他類別的方法或更新介面元素。本專案也特別運用表單的生命週期事件，如 Form\_Load 在表單初始化時載入資料、OnFormClosing 在表單關閉前解除事件訂閱等。透過這些事件處理，應用程式對使用者的每一步操作都給出及時回應並執行相應邏輯，構成整體的互動流程。
* **多表單視窗應用程式與清單控制項：** 本系統由多個表單組成，各表單之間以模式或非模式方式開啟。例如主窗體透過 ShowDialog 的模式窗口方式打開漫畫管理或會員管理視窗；租借管理表單同樣以模式窗口開啟，以便在處理完成前暫停主窗體操作。多表單的協調藉由共用的服務物件實現：新表單開啟時從 Program 類別取得對應服務實例傳入，使各表單對資料的操作實際作用於同一後端資料。至於清單控制項方面，正如前述，我們大量使用了 DataGridView 顯示資料表格，以及 ComboBox 下拉選單讓使用者選擇會員或漫畫。DataGridView 控制項提供類似資料表的介面，可以顯示多列多欄資訊並支援排序、選取等互動，是管理系統常用的清單元件；ComboBox 則適合用於讓使用者從有限選項中選擇一項，例如選擇欲操作的會員。透過將物件集合繫結到 ComboBox 的 DataSource，我們讓下拉清單直接顯示會員名稱且隱含其 ID 值。以上清單控制項的應用，使系統介面能直觀地呈現資料，同時提供友善的操作手段。
* **檔案與資料夾處理：** 本專案充分運用了 .NET 提供的檔案處理類別來實現資料的持久化和日誌。FileHelper 類別中使用 File.ReadAllLines 和 File.WriteAllLines 方便地讀寫整個檔案。在 Logger 中則利用 File.AppendAllText 將文字附加寫入日誌檔案。在操作檔案前，程式也透過 Directory.Exists 檢查目錄是否存在，若無則呼叫 Directory.CreateDirectory 建立所需路徑。例如，系統啟動時 Logger 會創建 *我的文件/ComicRentalApp/Logs/* 資料夾來存放日誌；讀寫資料時則使用 *AppData* 子資料夾存放 CSV 檔案。所有檔案操作都包在 try-catch 中以處理可能的 IO 異常，如檔案被佔用或不存在等。透過這些檔案與資料夾處理技巧，系統能夠在使用者電腦上自動建立所需的資料儲存環境，並穩健地進行資料輸出入，同時保障在異常情況下給予適當的提示或退避策略。

## **第四章 測試與驗證**

**單元測試案例：** 由於本專案屬於桌面應用程式且時間有限，未撰寫完整的自動化單元測試代碼。但我們透過人工設計了一系列測試案例來驗證各模組功能的正確性。例如：

* **測試字串解析：** 準備數行格式正確與格式錯誤的 CSV 字串，呼叫 Comic.FromCsvString 和 Member.FromCsvString 方法，確認正確的字串能生成對應物件，錯誤格式能拋出 FormatException。另外測試 ToCsvString 產生的字串能否成功由 FromCsvString 還原，以驗證序列化和反序列化的正確對稱性。
* **測試服務新增功能：** 呼叫 ComicService.AddComic 新增漫畫，觀察返回後內部漫畫列表大小是否增加且包含新對象，並檢查對應的檔案內容是否追加了新紀錄。對 MemberService.AddMember 進行類似測試。此外，在新增重複資料（如相同書名作者）時，期待服務不允許 ID 重複並在日誌中記錄警告。
* **測試租借/歸還流程：** 準備至少一位會員和一套漫畫資料，模擬使用者在租借表單選擇該會員和一本未出租漫畫，執行租借操作。確認結果：該漫畫的 IsRented 為 true、RentedToMemberId 等於選擇的會員 ID，並且漫畫已從「可借漫畫清單」中消失且出現在該會員的「已借漫畫清單」中。接著執行歸還動作，驗證漫畫狀態重置為未租借且重新出現在可借清單中。同時嘗試對已借出的漫畫再次租借或對無可用漫畫的會員執行歸還，系統應彈出提示阻止並保持資料正確性。
* **測試刪除功能：** 新增若干測試漫畫和會員資料，在管理表單界面執行刪除。確認當刪除一筆存在的資料後：檔案中對應行被移除、服務內部列表大小減少，且 UI 列表透過事件刷新不再顯示該筆資料。對於嘗試刪除不存在的 ID，服務預期拋出 InvalidOperationException 並被介面 catch，彈出錯誤訊息。在漫畫已租出的情況下刪除，系統允許刪除但會在日誌中記錄警告資訊以提醒資料不一致風險。
* **測試例外處理：**本系統廣泛測試了例外狀況的處理機制，以確保應用程式在各種異常情況下都能穩定運行。例如，透過模擬資料檔案遺失或損壞、網路中斷等情境，檢驗全域例外處理功能是否正常運作。程式在 Program.Main 中註冊了 UI 執行緒及非 UI 執行緒的全域例外處理事件，確保未捕捉的例外會被記錄並顯示錯誤訊息。我們手動製造未預期的例外（如在程式運行時強制拋出 Exception），確認系統能夠透過 Application.ThreadException 及 AppDomain.UnhandledException 攔截例外，並使用日誌記錄詳細錯誤再提示使用者。測試結果顯示，當發生異常時，系統並未崩潰，而是正確顯示錯誤對話框，同時將錯誤資訊寫入日誌，驗證了全域和區域的例外處理機制有效可靠。
* **測試資料讀寫與檔案處理：**為了驗證資料長期保存的可靠性，我們對系統的檔案讀寫功能進行了測試。系統使用 FileHelper 將漫畫和會員資料序列化為 CSV 檔案存儲，每次啟動時重新載入。測試首先在無資料檔的情況下啟動應用，確認系統能自動建立預設資料目錄及空白檔案。接著，新增多筆漫畫與會員資料後關閉程式，再重新啟動以驗證資料正確載入；結果顯示所有資料均成功保存並重載，資料檔採用UTF-8編碼且格式正確無誤。同時，我們故意修改或破壞部分CSV內容來模擬檔案格式錯誤，系統在讀取時能拋出並捕捉 FormatException，記錄錯誤日誌且以空清單方式恢復預設狀態。此外，在日誌功能測試中，系統會將關鍵操作與錯誤以附加方式寫入日誌檔；我們觀察到日誌檔案按預期生成並累積記錄，未出現檔案競爭或錯亂情況（透過 lock 鎖定機制確保執行緒安全）。這些測試證明系統的檔案處理模組穩定可靠，能正確地永久保存資料且防範常見檔案異常。
* **測試事件與委派機制：**本系統利用事件（Event）與委派（Delegate）來同步資料狀態與使用者介面。我們對此互動流程進行了嚴格驗證。例如，當新增或修改漫畫資料時，ComicService 會觸發定義好的 ComicsChanged 事件；主界面 (MainForm) 和相關子表單已訂閱此事件，以在偵測到事件時即時重新載入清單。測試時，在「漫畫管理」子視窗中新增漫畫後關閉該視窗，主界面的漫畫清單能即時出現新資料，證明事件通知機制有效。類似地，會員資料的變化也透過 MembersChanged 事件觸發介面更新。我們特別測試了同時進行多項操作的情境：例如租借作業（在 **租借管理** 視窗）會同時更改漫畫的出租狀態與所屬會員的紀錄，此時系統會發出多個事件。我們驗證了前端 UI 透過單一事件處理程序 Service\_DataChanged 同步接收並安全處理這些事件，介面上的「可租借漫畫列表」和「會員已借漫畫列表」能動態刷新一致。在大量連續操作下（例如連續新增多本漫畫或快速租借/歸還），事件機制未出現遺漏或競態條件，顯示委派與事件設計的穩定性。透過這些測試，我們確認系統的觀察者模式實作正確：資料層變化能可靠地通知UI層，確保使用者始終看到最新的狀態。
* **測試多執行緒與排程刷新：**本系統在 **租借管理** 模組中引入了背景執行緒定時刷新功能，以驗證多執行緒環境下資料一致性與執行安全。當租借視窗載入時，會透過 ReloadService 啟動一個平行執行的非同步任務，每隔1秒自動呼叫資料重新載入函式並更新UI。我們著重測試此週期性刷新機制是否會造成執行緒衝突或UI無回應：利用人工執行大量租借與歸還操作，同時觀察介面是否順暢更新。結果顯示，由於在更新UI前有檢查 InvokeRequired 並正確使用 Invoke 切換執行緒，多執行緒下UI更新運作正常，未出現跨執行緒呼叫異常。同時，我們確認了當關閉租借視窗時背景刷新執行緒會停止（觸發 \_cts.Cancel() 取消Task），避免資源浪費和潛在衝突。此外，在測試過程中我們觀察日誌，確認每次自動刷新都有對應記錄，週期任務未意外中斷。這些測試證明系統能適當運用非同步程序與計時器來增進使用者體驗，同時保持執行緒安全，進一步提升了系統的穩定性與反應速度。
* **測試功能完整性：我們根據專案需求對主要功能進行了最終驗證，確保系統各模組協同工作且滿足使用情境。測試包含以下案例：①新增資料測試**：在漫畫與會員管理介面分別新增多筆資料，檢查主介面列表更新以及CSV檔內容，結果均符合預期（資料完整寫入檔案且重新載入正常）。②**租借/歸還流程測試**：模擬一般租借流程——選擇會員和漫畫執行租借，確認漫畫狀態從「可租借」移至「已借出」，且對應會員名下列表新增該漫畫；隨後執行歸還操作，驗證漫畫狀態回復「可租借」，會員名下列表移除該項。期間亦測試了異常流程，如未選擇會員或漫畫就點擊租借、嘗試租借已被他人租出的漫畫等，介面均彈出正確的提示訊息且不會進行違規操作。③**邊界條件測試**：對極端情況進行測試，包括會員或漫畫列表為空時的操作、輸入過長或包含特殊字元的名稱，以及同名重複資料的處理。結果顯示系統對空清單情況能正常顯示且不當機；名稱欄位允許中英文及特殊符號並可正常儲存（特殊字元會在輸出CSV時被轉義處理）；新增重複名稱的漫畫時系統允許存檔但在日誌中給予警告提示，指出重複項已存在。整體而言，各項功能在單元測試與整合測試下均表現正確，操作流程順暢無邏輯錯誤，驗證了系統功能的完整性和正確性。

## **第五章 結論與未來工作**

* **專題結論：本漫畫租賃系統經由上述測試驗證，已達到預期的功能目標與穩定性要求。在設計上，系統運用了物件導向分析與多種C#技術實現核心需求，例如以類別和介面構築模組架構、使用泛型與集合來管理資料清單、透過事件機制確保前後端資料同步等。在操作過程中，系統能順利執行漫畫新增、租借、歸還及會員管理等功能，並透過清晰的GUI介面提供良好的使用體驗。得益於完善的例外處理與日誌紀錄，即使面對異常情況（如檔案遺失或輸入錯誤），系統亦能妥善處理而不會當機，充分體現了系統的穩定性**與**實用性**。綜上所述，本專題成功將課程第8~16章所學融會貫通於實作中，開發出一套功能完備且可靠的租賃管理應用程式。
* **個人收穫：**在開發過程中，我們深化了對C#各項技術的理解和運用能力，收穫良多。首先，透過實作 **介面** 與 **多型**，我們體會到介面導向設計帶來的彈性，例如定義 ILogger 介面並提供 FileLogger 實作，使日後可輕鬆擴充不同的記錄方式。其次，專題中大量使用 **委派與事件**，使我們熟悉了C#事件模型：我們學會了如何宣告事件、訂閱事件以及在適當時機觸發事件來更新UI，這加強了對觀察者模式的理解。另外，應用 **泛型集合** (List<T>) 及 **LINQ查詢** 提升了資料處理效率，例如使用 List<Comic> 儲存漫畫資料並透過 LINQ Where 篩選可出租項目，加深了我們對泛型與Lambda運算式的掌握。透過 **檔案處理** 模組的開發，我們學會了使用 System.IO 進行檔案讀寫、路徑操作及編碼處理，並注意檔案存取的例外處理與執行緒安全（例如運用檔案鎖定防止並發問題）。最後，嘗試 **多執行緒與非同步編程** 擴展了我們對併發的認識：我們使用 Task 平行執行週期任務，並透過 CancellationToken 控制任務週期，實踐了基礎的併發程式設計。這些在課堂學習的理論知識，經由專題實作得到鞏固與驗證，也培養了我們解決實際問題的能力。
* **技術限制與挑戰：儘管系統功能已實現基本需求，但在開發過程中我們也發現了一些技術限制和挑戰，有待日後改進。首先，目前資料儲存使用純文字CSV檔案，此方式雖簡單但不適用於大量資料或多使用者併發存取**。由於缺乏資料庫支援，系統難以處理更複雜的查詢需求，也無法避免多執行緒同時寫入時的衝突風險（現階段以檔案鎖及單執行緒存取勉強避免）。其次，**輸入驗證**方面還需加強：目前對使用者輸入的會員電話等欄位缺乏格式校驗，例如未嚴格檢查電話號碼的有效性，這可能導致不一致或錯誤資料的輸入。再次，系統的**錯誤處理**雖完整但仍可更精細，例如針對不同例外提供更具體的回饋訊息，以及在UI上高亮錯誤欄位。目前發現的挑戰還包括：在WinForms架構下進行多執行緒UI更新須謹慎處理，所幸我們已運用Invoker機制避免跨執行緒呼叫問題，但未來如轉換為WPF或其他架構時需重新考量更新機制。最後，如何保持良好的**擴充性**亦是一大考驗；隨著功能增加，程式碼架構可能變複雜，因此我們需要持續重構並遵循SOLID原則，確保未來新增模組（例如引入預約功能或統計報表）時不會影響現有系統的穩定。
* **未來工作與擴充規劃：**基於上述限制，我們制定了後續幾項優化方向，以提升系統性能與擴充性：  
  1. **引入資料庫後端：**未來計劃將資料存儲遷移至關聯式資料庫（如SQL Server或SQLite），以取代目前的CSV檔案。透過資料庫可支援更大量的資料管理、複雜查詢，以及資料並發存取控制，並利用ORM技術簡化資料操作。
  2. **強化輸入驗證：**在UI層增加輸入資料的驗證機制，例如使用正規表達式檢核電話號碼與ISBN格式，並在欄位輸入不合法時即時提示使用者修正。同時，可實作必填欄位檢查，避免重要資料漏填導致後續操作錯誤。
  3. **改善使用者介面：**未來工作將包括提升介面友好度與操作體驗。例如，加入搜尋與排序功能，方便使用者快速找到特定漫畫或會員；提供租借記錄的過濾與查詢；在介面上顯示漫畫封面縮圖等，以增加系統的實用性。此外，可考慮將WinForms介面升級為WPF或Blazor等現代UI框架，以獲得更佳的視覺效果和響應式體驗。
  4. **擴充功能模組：在現有基礎上，我們構想了若干新功能作為未來擴充方向。例如：增加預約**機制，允許會員預約未來可租借的漫畫；導入**逾期管理**，自動計算租借逾期並發送提醒；實現**報表統計**，提供每月租借量、熱門漫畫排行等報表分析功能。這些功能將使系統更全面且更貼近實際業務需求。
  5. **部署與多平臺支援：**目前系統以桌面應用形式運行，未來可考慮擴展為網路版或行動版。透過將後端服務化，提供Web API介面，可使不同平台（網站、Android/iOS App）存取同一套服務，實現多終端的漫畫租賃管理。同時，可研究雲端部署方案，讓系統具備雲端備份與在線更新功能。

透過以上未來工作規劃，我們希望逐步提升系統的效能與實用範圍，同時將在此專題中習得的C#開發技能應用到更多元的場景中。