漫畫租借管理系統 – 小專題

組員：B11170047 陳家穌、B11170075 陳泓維

班級：四資工三乙

第一章 緒論

- 動機：隨著漫畫閱讀需求增加，需要一套數位化的漫畫租借管理系統來方便讀者搜尋、借閱和歸還漫畫，同時協助管理員維護資料。例如，本系統採用 C# 和 Windows Forms 技術實現，讓使用者能在桌面端進行操作。

- 目的：建立一個功能完備的漫畫租借系統，提供 管理員 與 會員 兩種角色，管理員可新增/編輯漫畫、管理會員帳號、處理租借與歸還流程，會員則可瀏覽、搜尋可租借的漫畫並查看自己的租借狀態。同時結合字串處理、物件導向、例外處理、事件驅動等 C# 課程內容，作為期末專題實作案例。

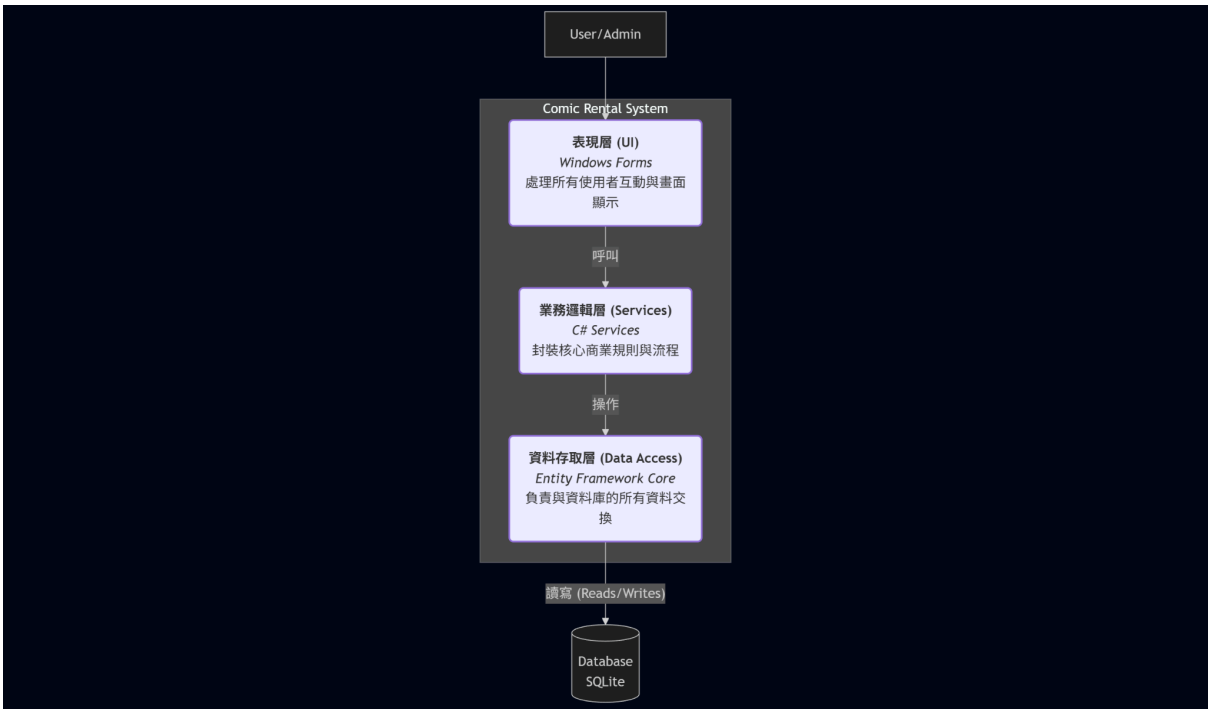
- 問題陳述：如何設計資料庫與類別模型以記錄漫畫資訊、會員資料、使用者帳號等？如何利用視窗表單與控制項提供直觀的操作介面？如何確保異常情況（如重複註冊、檔案 I/O 錯誤）能被正確處理？如何整合多表單與清單控制元件顯示資料？

- 貢獻：本系統實現了漫畫管理、會員管理、帳號管理、租借流程管理和日誌管理等完整功能，並使用 Entity Framework Core 與 SQLite 完成資料儲存。在教育目標上，程式中使用了從第 8 章到第 15 章的各類技術，如字串處理、類別與物件、繼承與介面、過載與多型、例外處理、委派與執行緒、視窗應用程式的事件處理、多表單視窗應用程式與清單控制項、檔案與資料夾處理等（下述章節將依次說明）。這些貢獻顯示系統兼具實用價值與技術完整性。

第二章 系統設計

- 系統架構：本系統採用分層架構設計，主要分為「資料存取層」(Models + DbContext)、「業務邏輯層」(Services，如 `ComicService`、`MemberService`、`AuthenticationService` 等) 與「表現層」(Windows Forms)。應用程式以 `Program.cs` 為入口，使用 .NET 8.0 啟動，並透過依賴注入 (依賴服務注入容器) 初始化 Logger、FileHelper、資料庫上下文等。系統架構圖可描述為：使用者經由主視窗與多個表單互動，各功能表單調用對應服務方法來存取資料庫，服務層使用 EF Core 與 SQLite 進行資料操作（如下圖所示）。此設計強調模組化，各層次分工清晰，並支援

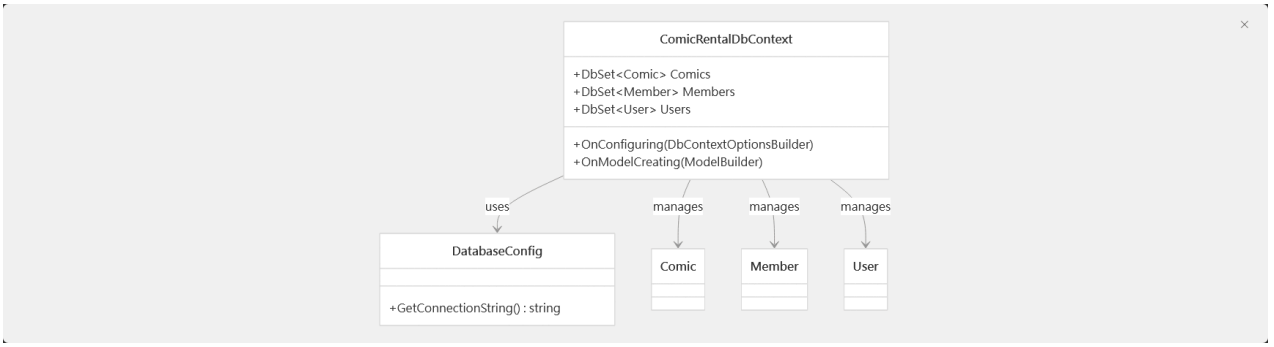
日誌與資料遷移功能。



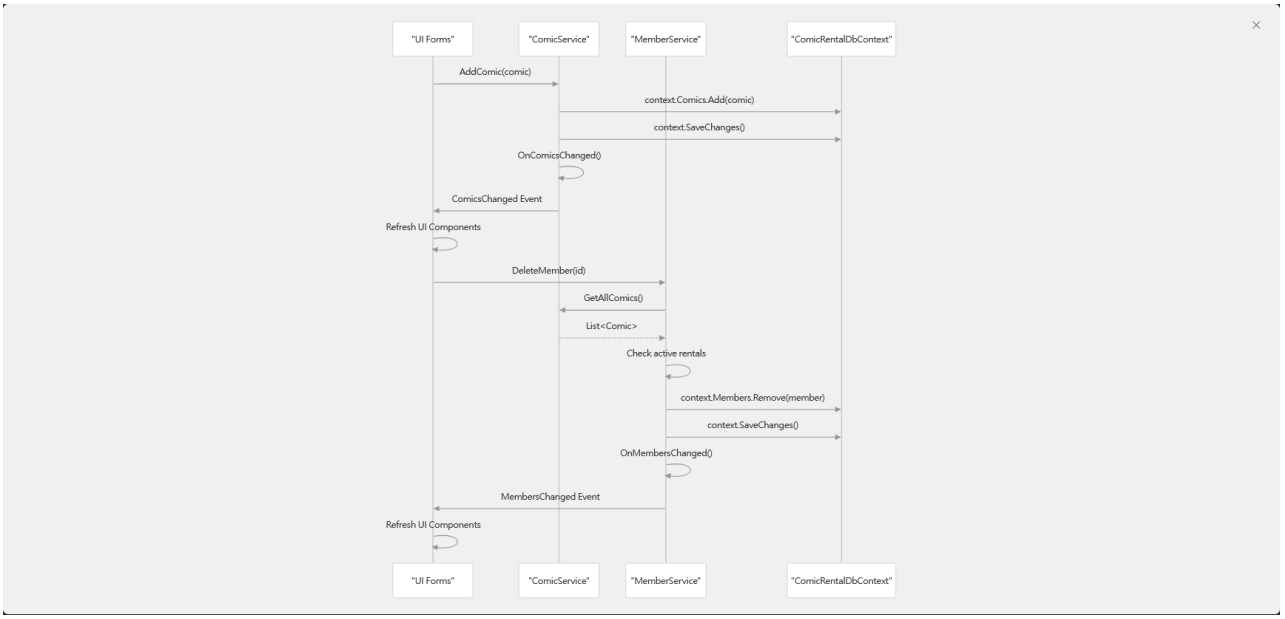
- 功能模組說明：核心功能模組包含：
    - 使用者認證 (**AuthenticationService**)：負責登入、註冊與鎖定機制，使用者類別 `User`（繼承自 `BaseEntity`，具有使用者名稱、密碼雜湊、角色等欄位）實作帳號管理。密碼錯誤次數累計達上限時自動鎖定帳號，驗證結果由服務回傳。
    - 漫畫管理 (**ComicService**)：提供新增、編輯、刪除、查詢漫畫等功能。漫畫類別 `Comic` 包含標題、作者、ISBN、類型、是否已出租、借閱者編號等屬性。管理員在「漫畫管理表單」中可編輯漫畫資料及上傳封面圖片。
    - 會員管理 (**MemberService**)：處理會員資料的增刪改查，類別 `Member` 繼承自 `BaseEntity`，包含姓名、電話、用戶名等欄位（參見程式碼）。刪除會員前須檢查該會員是否尚有未歸還漫畫。
    - 租借管理 (**RentalForm** 等)：協助為會員借閱或歸還漫畫。管理員選擇會員後可瀏覽其當前租借紀錄，並新增新的借閱紀錄。資料使用多表聯查，更新漫畫的借閱狀態、租借日期與歸還日期。
    - 日誌管理 (**FileLogger**、**LogManagementForm**)：系統所有關鍵操作與錯誤皆使用 `ILogger` 介面紀錄至每日產生的文字日誌檔案。管理介面可檢視日誌內容或修改保留天數設定，舊日誌會依設定自動移至備份並刪除。
- 資料庫設計：使用 SQLite 作為資料儲存。主要資料表由 `ComicRentalDbContext` 定義，包括 `Comics`、`Members`、`Users` 等三張資料表。每張表的主鍵為 `Id`（來自 `BaseEntity` 類別）。`Comics` 表欄位包含漫畫的書名、作者、ISBN、類型、租借狀態、

借閱者 ID 及借閱/預期歸還/實際歸還時間；`Members` 表包含會員姓名、電話號碼、使用者帳號；`Users` 表則存放系統使用者帳號與角色（會員或管理員）。系統啟動時檢查資料庫檔案是否存在，若不存在則自動建立並匯入 `Data` 資料夾下的初始 CSV/JSON 檔案。

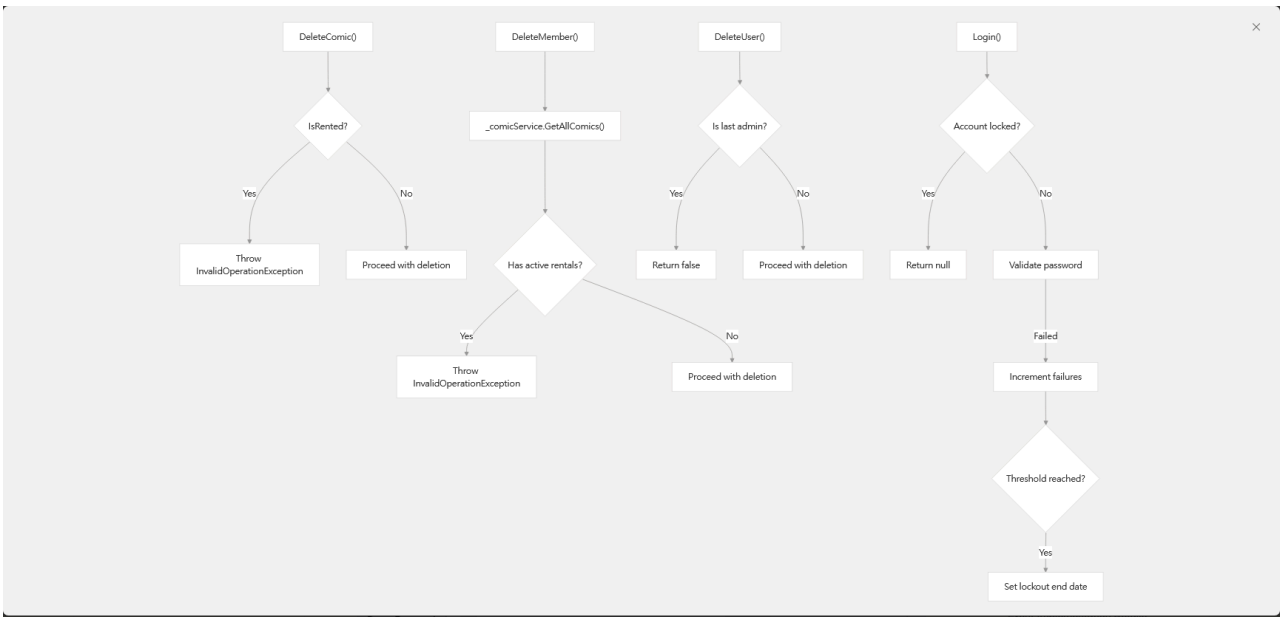
- **UML與流程圖**：



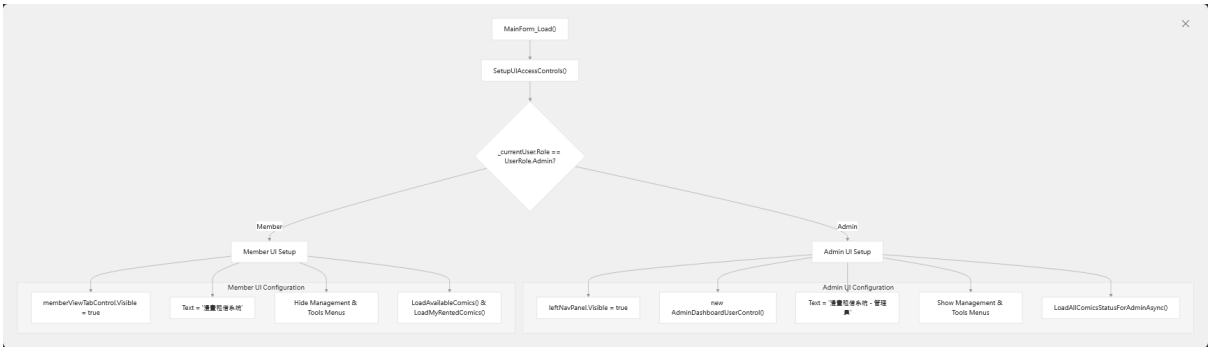
▲Data Access Layer Entity Framework Class Diagram



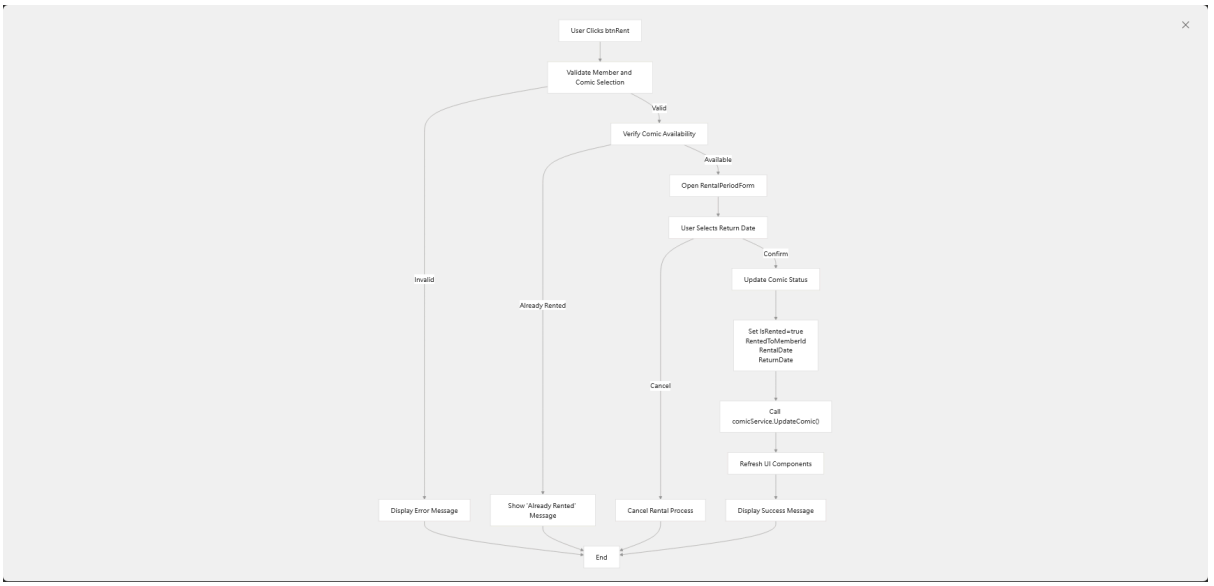
▲Core Business Services Sequence Diagram



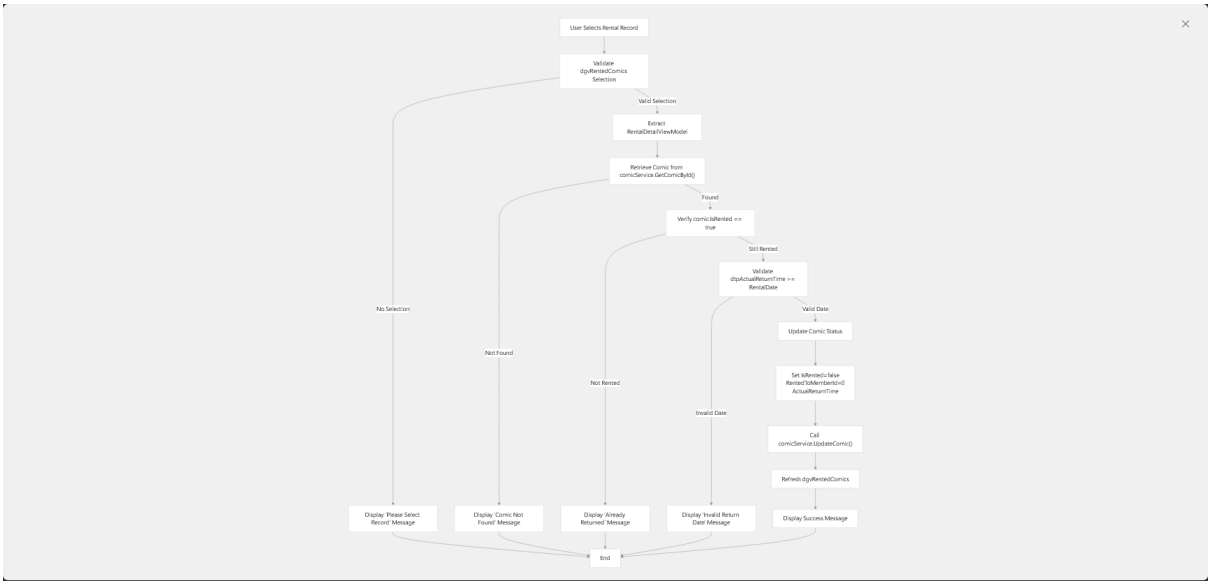
▲Business Rule Enforcement Flow



▲Role-Based UI Switching Logic Flowchart



▲Rental Processing Workflow



▲Return Processing Workflow

第三章 系統實現

- 開發環境：使用 Visual Studio (建議 2022 版) 作為開發工具，目標 .NET 8.0 Runtime (或相容版本)。程式語言採 C#，介面框架為 Windows Forms，資料庫為 SQLite，並使用 Entity Framework Core 作為 ORM。系統啟動流程參考以下程式碼：在 `Main` 方法中配置依賴注入容器 (DI Container)，註冊 `ILogger`、`FileHelper`、`ComicRentalDbContext`、各服務等；初始化時先執行 `DataMigrationService.MigrateFromFiles()` 以讀取初始資料，並確保至少有一組管理員帳號存在。

- 核心功能代碼解析：

字串與陣列 **(第八章)**：在資料遷移 (`DataMigrationService`) 中，我們讀取 CSV 檔並解析每一行為欄位字串列表。程式使用 `StringBuilder` 處理含引號的欄位，並以逗號切分為 `List<string>`。例如下列程式片段展示了對 CSV 行的解析：

```
List<string> fields = new List<string>();
StringBuilder fieldBuilder = new StringBuilder();
bool inQuotes = false;
for (int i = 0; i < csvLine.Length; i++) {
    char c = csvLine[i];
    if (inQuotes) { /* 處理引號內文字 */ }
    else if (c == ',') {
        fields.Add(fieldBuilder.ToString().Trim());
        fieldBuilder.Clear();
    }
    else {
        fieldBuilder.Append(c);
    }
}
fields.Add(fieldBuilder.ToString().Trim());
```

- 以上範例使用字串與字元陣列操作方式，正是第八章所提的字串處理技巧。另如 `FileHelper` 的 `ReadFile<T>` 方法中使用 `File.ReadAllLines` 讀取文字檔，其回傳 `string[]` 陣列並對每行進行解析。

- 類別與物件 (第九章)：系統中各實體（漫畫、會員、使用者帳號）皆用 C# 類別表示。如 `Comic` 類別繼承自 `BaseEntity`，封裝了書名、作者、ISBN 等屬性。在程式中，使用 `new` 建立 `Comic`、`Member`、`User` 等物件，並透過 EF Core 進行新增或更新操作。例如，`MemberService.GetAllMembers()` 會產生一個 `ComicRentalDbContext` 物件以查詢 `Members` 資料表。類別也實作了建構子，用以初始化預設值，如 `Comic()` 建構子將字串欄位設為 `String.Empty`。

- 繼承與介面 (第十章)：本系統使用繼承與介面提高擴充性與模組化。`BaseForm` 與 `ModernBaseForm` 類別均繼承自 `Form`，統一設定視窗外觀樣式。日誌功能定義了 `ILogger` 介面，`FileLogger` 實作了此介面的方法；依賴注入 (DI) 容器注入 `ILogger` 時可替換不同實現。此外，`IFileHelper` 介面及其實作 `FileHelper` 支援抽象化的檔案存取。介面與繼承的運用體現了第十章多型設計理念。

- 過載與多型 (第十一章)：程式中多處使用方法過載以增強靈活性。例如 `FileLogger.Log` 同時提供 `Log(string message)` 與 `Log(string message, Exception ex)` 兩種簽章，調用時可傳入例外物件以詳述錯誤。

`FileHelper.WriteFile` 也分別定義了接受單一字串與泛型集合的版本。這些過載方法實現了多型 (Polymorphism)，使呼叫者可依需求傳遞不同參數。

- 例外處理、委派與執行緒 **(第十二章)**：程式中廣泛使用 try-catch 進行錯誤處理，並在必要時將錯誤資訊寫入日誌。例如，在 `DataMigrationService` 的 CSV 解析迴圈中，對格式不正確的行進行例外攔截並記錄；在 GUI 層操作資料時，如果發生例外則會彈出錯誤訊息框並記錄錯誤。此外，系統透過事件 (委派) 處理非同步和通知邏輯：如 `MemberService` 宣告了 `MembersChanged` 事件，並在新增/更新會員後觸發它；對應的 `MemberManagementForm` 訂閱此事件，在事件觸發時重新載入資料。在應用程式入口，使用 `Application.ThreadException` 和 `AppDomain.CurrentDomain.UnhandledException` 監聽全域執行緒例外，確保 UI 執行緒異常能被捕捉並寫入日誌。

- 視窗應用程式的事件處理 **(第十三章)**：各表單元件的事件繫結在程式碼中明確定義。例如，在 `MemberManagementForm` 中將主表單的 `KeyDown` 事件綁定到 `MemberManagementForm_KeyDown` 以監聽快捷鍵；另外，當搜尋輸入框按下 Enter 鍵時亦觸發 `txtSearchMembers_KeyDown` 來執行搜尋。資料網格檢視 (`DataGridView`) 則將 `SelectionChanged` 事件繫結到 `dgvMembers_SelectionChanged`，讓使用者選擇表格列後對應的按鈕狀態可更新。這些事件處理碼展示了 Windows Forms 應用程式的事件驅動特性。

- 多表單視窗應用程式與清單控制項 **(第十四章)**：系統包含多個表單 (Forms)，如 `LoginForm`、`MainForm`、`ComicManagementForm`、`MemberManagementForm` 等，它們都繼承自自訂的 `BaseForm` 類別以套用統一風格。各表單之間可透過選單或導覽面板自由切換。例如主窗體在側邊提供導覽按鈕，點選後打開對應功能表單。列表控制項方面，本系統大量使用 `DataGridView` 來顯示資料清單。在 `MemberManagementForm` 中，先以 `dgvMembers.Columns.Add(...)` 動態定義欄位，再將 `members` 清單指定為 `DataSource` 更新介面。列表支援排序與篩選操作；對應地，在搜尋功能中使用 LINQ 查詢資料，再在 UI 執行緒上更新資料來源。

- 檔案與資料夾處理 **(第十五章)**：系統需讀寫外部檔案，如日誌檔、資料遷移檔案 (CSV/JSON) 等。`FileHelper` 類別提供了多種檔案操作方法，包括讀取/寫入文字檔、移動/複製檔案等。`FileLogger` 在初始化時使用 `Directory.CreateDirectory` 建立日誌目錄，並實作每日檔名產生日誌寫入（`File.AppendAllText`）及過期日誌自動備份刪除功能。這些程式碼大量運用檔案系統 API，對應課本第十五章的檔案夾處理議題。

- 介面與操作流程：登入窗體 (`LoginForm`) 是系統入口，輸入帳號密碼後選擇角色登入；管理員與會員登入後介面略有差異。管理員主介面左側為快速導覽面板，可切換儀表板、漫畫管理、會員管理、租借管理、帳號註冊、日誌管理等功能；頂部選單則提供重複功能的快速入口。表單佈局上採用現代扁平風格，按鈕、分組框均套用 `ModernBaseForm` 中統一樣式。操作流程舉例：管理員若要新增漫畫，點選「漫畫管

理」開啟編輯視窗，在 ComicEditForm 中填寫書名、作者等欄位並上傳封面圖片，最後按下「儲存」，程式會在 ComicService.AddComic 中檢查資料有效性並存入資料庫，再回到主清單中更新顯示。在會員介面，主視窗分為兩個分頁：「可租借漫畫」及「我的租借」。在「可租借漫畫」頁籤，系統讀取 GetAvailableComics() 列表並顯示漫畫資訊及封面；使用者可輸入關鍵字搜尋，程式依此篩選結果（使用字串函式查詢）並更新列表。在「我的租借」頁籤，系統列出使用者目前所有租借紀錄，並以顏色標記即將到期或逾期的漫畫。整體操作流程順暢，且每次資料更新均寫入日誌以供後續查核。

第四章 測試與驗證

- 單元測試：專案包含 NUnit 單元測試 (位於 ComicRentalSystem.Tests) 以驗證關鍵服務邏輯。測試示例可見下列程式碼：

    - 驗證註冊功能：AuthenticationService.Register 方法應能成功新增新使用者，並確認資料庫中有該使用者紀錄。

    - 驗證登入機制：使用正確密碼時回傳使用者物件，錯誤密碼則回傳 null，且錯誤次數計數增加；連續 5 次失敗後確認鎖定時間屬性已設定。

    - 驗證漫畫與會員新增：ComicService.AddComic 和 MemberService.AddMember 在有效資料時應正確寫入資料庫，若資料不合法則拋出 ArgumentException。
      以上測試充分涵蓋了後端邏輯的正確性，如帳號重複判斷、資料完整性檢查等，符合單元測試要求。

- 系統整合測試：在本階段亦對整個系統流程進行手動測試。例如模擬使用者登入、漫畫新增、借閱流程、歸還流程，並確認資料正確流轉。透過 UI 測試，檢查畫面元素是否正確顯示、功能按鈕是否有效，如搜尋功能能正確過濾清單等。這些測試確保前後端協作無誤。由於使用 EF Core 的實際資料庫，整合測試即包含資料層真實交互。

- 效能評估：由於系統主要操作為 CRUD 和列表查詢，效能瓶頸不明顯。系統在小型 SQLite 資料庫中執行快速；欲提升效能可考慮未來加入查詢索引或資料分頁功能（如漫畫數量很大時）。目前系統未實做大數據量壓測，待日後可透過效能測試工具進行評估。

第五章 結論與未來工作

- 成果總結：本專題成功實作了一個 C# 漫畫租借管理系統，具備管理員與會員角色分權功能，涵蓋漫畫資料管理、會員管理、借閱流程、日誌記錄等主要需求。實作過程中應用到 C# 課本第 8～15 章所提之各種技術，如字串處理、物件導向設計、例外處理、事件與多型等，並透過實際程式碼案例加深理解。系統界面設計現代化，使用者體驗友好，且程式內建資料遷移與自動產生日誌機制，提高了系統穩定性與可維護性。單

元測試也證明核心功能行為符合預期，提高了程式可靠度。

- 專題限制：目前系統尚存在一些限制：例如租借過程中未計算逾期罰金等商業邏輯；UI 版面未針對手機平台優化；目前未實作繁體/簡體/英語等多語系支援；此外，目前僅針對小型資料庫運作，未經過高負載效能測試。開發時間有限，系統也未加入自動化集成測試或更多安全性防護機制（如密碼強度檢查等）。

- 未來改進方向：未來可考慮下列方向：**(1)** 新增逾期通知與罰金計算功能，提升系統實用性；**(2)** 擴充會員功能，允許會員自行申請歸還，並由後端自動確認書籍狀態；**(3)** 強化介面與使用者體驗，例如支援拖放封面上傳、多國語系選擇、響應式布局等；**(4)** 強化安全性，加入更完善的權限驗證、密碼強度策略；**(5)** 擴增日誌與測試，開發自動化測試腳本執行整合測試；**(6)** 若應用於生產環境，可改採伺服器資料庫與分散式架構，考量效能與可擴展性。以上改進將進一步提升系統的完整度與實用價值。