漫畫租借管理系統 – 小專題

第一章 緒論

研究動機

本專題選擇「漫畫租借管理系統」作為開發主題，源自於對漫畫租借流程數位化的需求。傳統紙本或人工管理漫畫租借，不僅效率低下且容易發生遺漏或錯誤。隨著圖書館和漫畫店藏書量的增長，人工追蹤每本漫畫的出借與歸還狀態變得困難。另外，在資訊教育課程中，我們希望透過實作一個中小型系統，來整合所學的C#程式語言觀念。因此，本專題的動機在於解決漫畫租借管理上的不便。

研究目的

本專題旨在開發一套桌面應用程式，提供使用者一個友善的介面來管理漫畫租借相關的所有事務。具體目標包括：建立漫畫庫存管理功能，可新增、編輯、刪除漫畫資料；提供會員帳戶管理功能，可新增會員、維護會員資料；實作漫畫租借/歸還管理流程，讓管理員能將漫畫出借給特定會員並記錄歸還日期，以及讓會員可自行借閱漫畫；此外實現使用者身份驗證與權限控管，包括管理員與一般會員的帳號登入機制。系統還需確保資料可靠儲存（例如存放於檔案中）、操作流程順暢，並透過良好的例外處理機制提高系統穩定性。總之，本專題的目的是打造一個功能完整的漫畫租借管理系統，以提升租借管理效率並驗證我們對C#開發技術的掌握。

問題陳述

在沒有專門系統的情況下，漫畫租借涉及多方面的問題：(1) 庫存狀態難以及時掌握：管理人員需手動記錄每本漫畫是否已被租出，何時該歸還，容易因遺忘或疏忽導致逾期未歸還的情況未被察覺。(2) 會員資料管理不便：缺乏系統化的會員名單與聯絡資訊管理，無法快速查找會員的借閱紀錄或聯絡方式。(3) 租借流程缺乏管控：若沒有電腦系統輔助，難以實現例如“一本漫畫同一時間只能租給一位會員”或“同一會員不可重複租借相同漫畫”等規則管控。(4) 資訊安全與紀錄：用戶帳號密碼及操作紀錄若未妥善管理，可能有安全風險或無法追蹤操作歷程。為了解決上述問題，本系統需要提供即時的漫畫庫存與租借狀態查詢、完整的會員資料維護、嚴謹的租借流程控制，以及安全的帳戶驗證與日誌紀錄功能，使整個漫畫租借業務在電腦輔助下高效且可靠地運作。

預期貢獻

本漫畫租借管理系統的開發，預期可帶來以下貢獻與效益：

- 提升管理效率：透過系統自動化管理漫畫清單、會員資訊以及租借紀錄，減少人工記帳的負擔與錯誤。管理員能即時查詢館藏漫畫的可借狀態與目前租借情形，會員也可快速瞭解自己已借閱漫畫及歸還期限。

- 確保資料正確與安全：系統在關鍵操作（如新增漫畫、租借、歸還）時即時更新檔案資料，並採用例外處理機制確保資料的一致性。例如，新會員註冊流程中，如在建立會員記錄時發生錯誤，系統會回滾先前新增的使用者帳戶以維持資料整合。使用者密碼以雜湊搭配隨機鹽值方式儲存，增強帳戶資料安全性。此外，系統實作登入失敗次數侷限與帳戶鎖定機制（連續5次密碼錯誤將鎖定15分鐘），保障帳戶安全。

- 良好的使用者體驗：透過Windows Forms圖形介面提供友善的操作流程，多視窗介面讓管理員與會員依身份執行相應功能。清晰的提示訊息與日誌紀錄讓操作過程透明可追溯，例如當發生錯誤或異常時，使用者會看到提示並可透過日誌了解詳細資訊。這些設計提高了系統的可用性與信賴度。

- 學術與實務價值：對於開發團隊而言，此專題統整了課程中第8～15章的各項C#語言概念（如字串/陣列、物件導向、介面、例外處理、事件、執行緒等）的實際應用，深化了對理論的理解。同時，本系統可作為日後開發類似館藏租借系統的基礎，具有擴充應用的價值，未來可進一步發展成圖書館管理或線上租書平台等系統。

第二章 系統設計

本章將說明漫畫租借管理系統的整體架構與設計細節。我們採用了分層式的系統架構，將應用程式劃分為使用者介面層、邏輯處理層與資料儲存層，以提高模組化程度和系統可維護性。同時，本章亦包含主要功能模組的說明、資料檔案的儲存設計，以及系統類別的UML類別圖，以展示類別之間的關聯與繼承關係。

系統架構設計

本系統採用三層式架構：(1) 表示層（**UI**）：使用Windows Forms構建圖形使用者介面，提供管理員和會員與系統互動的窗口；(2) 業務邏輯層：由一組C#服務類別組成，負責處理核心業務邏輯，如帳戶驗證、漫畫與會員資料管理、租借流程控制等；(3) 資料存取層：透過檔案進行資料永久儲存，包括使用者帳戶檔、漫畫清單檔及會員清單檔。本地檔案在系統執行期間由檔案助手類別進行讀寫操作。

- 使用者在客戶端啟動應用程式後，首先透過登入介面進行身份驗證。UI層的`LoginForm`將使用者輸入的帳號與密碼傳送給驗證服務**(AuthenticationService)**檢查。業務邏輯層的**`AuthenticationService`**負責比對帳密、處理登入錯誤次數以及帳戶鎖定等邏輯。驗證服務從使用者資料檔（如`users.json`）載入帳戶資訊並執行上述檢查，結果再回傳給UI層以決定登入是否成功。

- 登入成功後，系統根據使用者角色載入不同的主介面。對於管理員，主窗體`MainForm`將呈現管理後台功能，例如切換到管理員儀表板（顯示統計資訊）和提供操作選單來管理漫畫、會員及租借事項。對於一般會員，`MainForm`則載入會員專區視圖，展示可供借閱的漫畫清單與會員自己的租借紀錄。UI層的按鈕和選單操作（如點擊「新增漫畫」或「租借」按鈕）會呼叫業務邏輯層中相應的服務物件來執行功能。整個應用各個Windows Form透過服務物件共享資料狀態，確保不同介面之間的資料一致。

- 業務邏輯層包含以下主要服務類別：

    - 漫畫服務（**ComicService**）：負責管理漫畫資料的生命週期，包括從檔案載入所有漫畫記錄、提供方法新增/編輯/刪除漫畫，以及處理漫畫租借狀態變更。漫畫服務內部維護一個漫畫列表，並在每次修改後即時將更新寫回檔案，確保持久化。

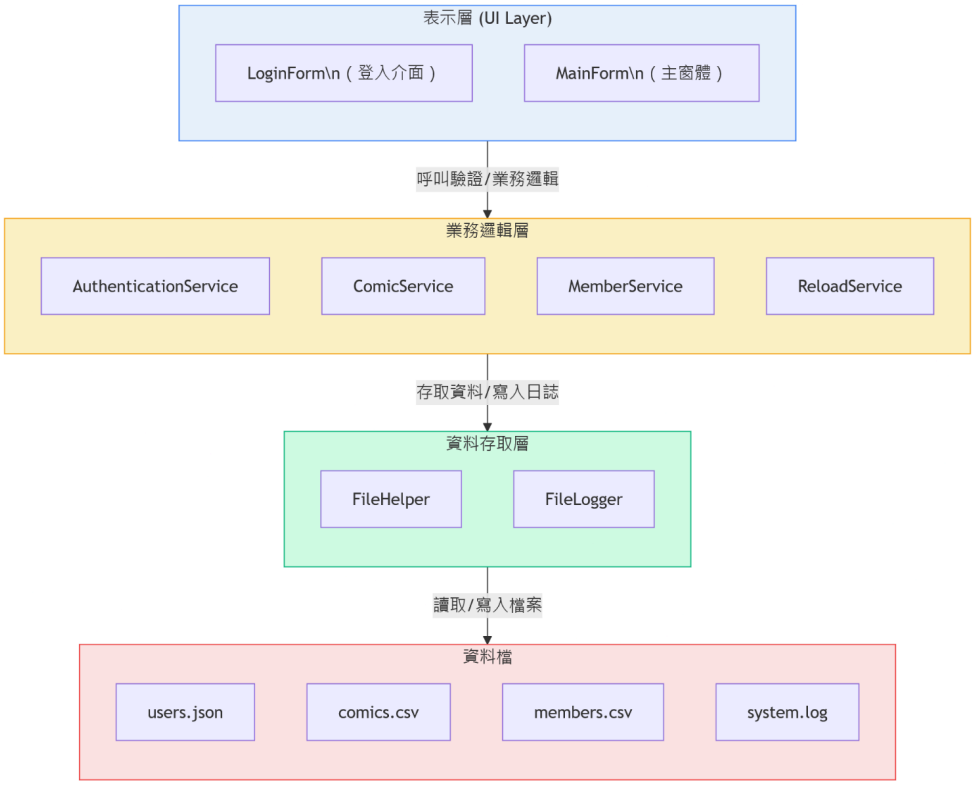
    - 會員服務（**MemberService**）：負責管理會員資料，如新增/更新會員資訊、查詢會員，以及與漫畫服務協調以防止不允許的操作（例如會員尚有借出的漫畫時禁止刪除該會員）。會員服務也在每次資料變動後將資料儲存至檔案。

    - 驗證服務（**AuthenticationService**）：專責處理使用者帳戶的註冊、登入與刪除等操作，包括密碼雜湊與驗證、登入嘗試計數與鎖定等安全機制。它啟動時從帳戶資料檔讀取使用者清單，登入或註冊成功後亦將更新結果寫回檔案保存。

    - 重新載入服務（**ReloadService**）：此服務用於自動定時刷新資料。在租借作業介面中，ReloadService會以背景執行緒定期呼叫漫畫服務與會員服務的重新載入方法(ReloadAsync)，確保多視窗同步使用時資料即時更新。例如每隔30秒自動從檔案重新載入最新漫畫與會員列表，用於更新租借狀態。ReloadService的運行及停止由業務邏輯層調度控制。

- 資料存取層則由**檔案助手（FileHelper）和日誌紀錄器（FileLogger）**等輔助類別組成。`FileHelper`實作了`IFileHelper`介面，負責所有檔案的讀/寫/刪除等底層操作。

例如，漫畫服務呼叫`FileHelper.ReadFile<Comic>()`從`comics.csv`讀取漫畫清單，並透過傳入的解析函式將每行文字轉換為Comic物件；完成修改後，呼叫`FileHelper.WriteFile<Comic>()`將Comic清單序列化為CSV格式寫回檔案。如此，業務層與檔案存取細節解耦，提升模組獨立性。另一方面，`FileLogger`實作了`ILogger`介面，用於將系統運行過程中的重要訊息與錯誤事件附加寫入日誌檔案。所有服務類別和主要表單在關鍵步驟都使用`ILogger`來記錄操作（例如使用者登入成功或失敗、資料檔載入錯誤等）。透過日誌，管理員可在發生問題時追蹤詳細資訊，有助於除錯與系統維護。



圖一:系統架構圖

功能模組說明

- 使用者帳戶與權限模組：提供會員與管理員的帳戶管理功能，包括使用者註冊、新增帳戶、登入驗證以及角色權限區分。管理員帳戶具有系統所有功能的存取權，而一般會員帳戶僅能執行查詢漫畫及自身租借操作。首次啟動系統時會自動確保存在一個預設管理員帳戶（帳號為"admin"）供登入使用。此模組在登入過程中實作密碼雜湊比對、連續登入失敗次數統計與帳戶鎖定等安全機制，防止未經授權的存取。管理員亦可透過介面調整某個帳戶的角色（例如將某會員提升為管理員）。

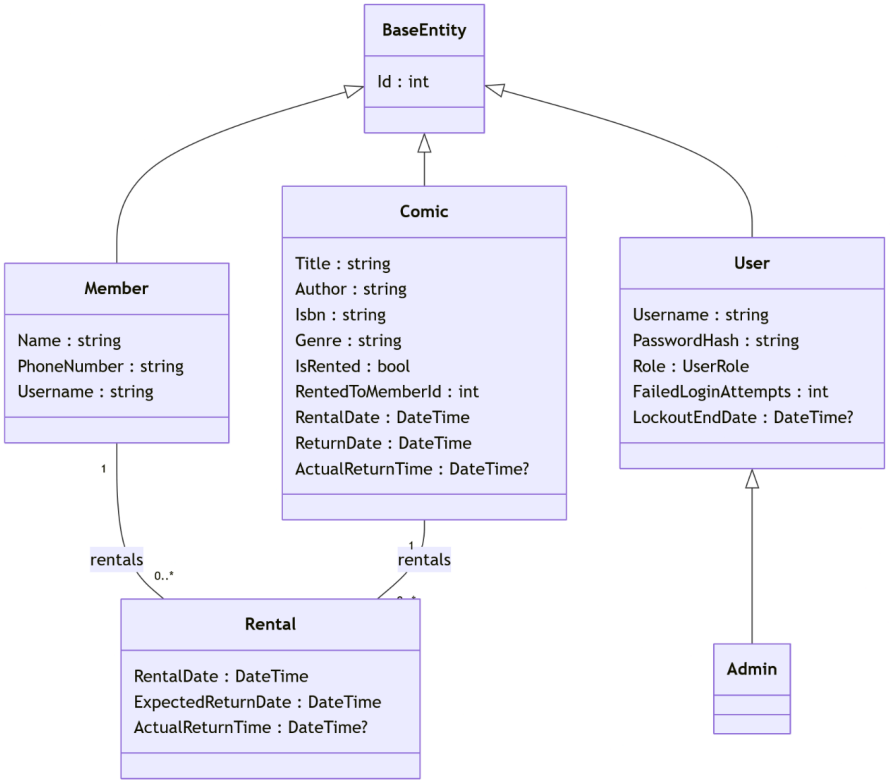
- 漫畫資料管理模組：負責維護漫畫書目資料庫。管理員通過此模組可以新增漫畫書目（包括書名、作者、ISBN、類型等欄位）、編輯現有漫畫資訊，以及刪除不再收藏的漫畫。為防止重複資料，系統在新增漫畫時若發現同名同作者的書籍已存在，會提出警示但仍允許儲存（可能不同冊數）。刪除漫畫時，系統會檢查該漫畫是否正在被租借中，若是則禁止刪除以免遺失租借紀錄。該模組確保漫畫清單的正確性並維持館藏的一致性。

- 會員資料管理模組：提供管理員維護會員資訊的功能。管理員可新增新的會員資料（例如讀者辦卡），也可編輯會員的姓名、聯絡電話等資訊或移除會員。當嘗試移除會員時，系統會檢查此會員是否目前借有漫畫，若是則禁止刪除該會員，以防止有未歸還的漫畫失去借閱者記錄。此模組讓管理員有效管理讀者名單並保持資料完整。

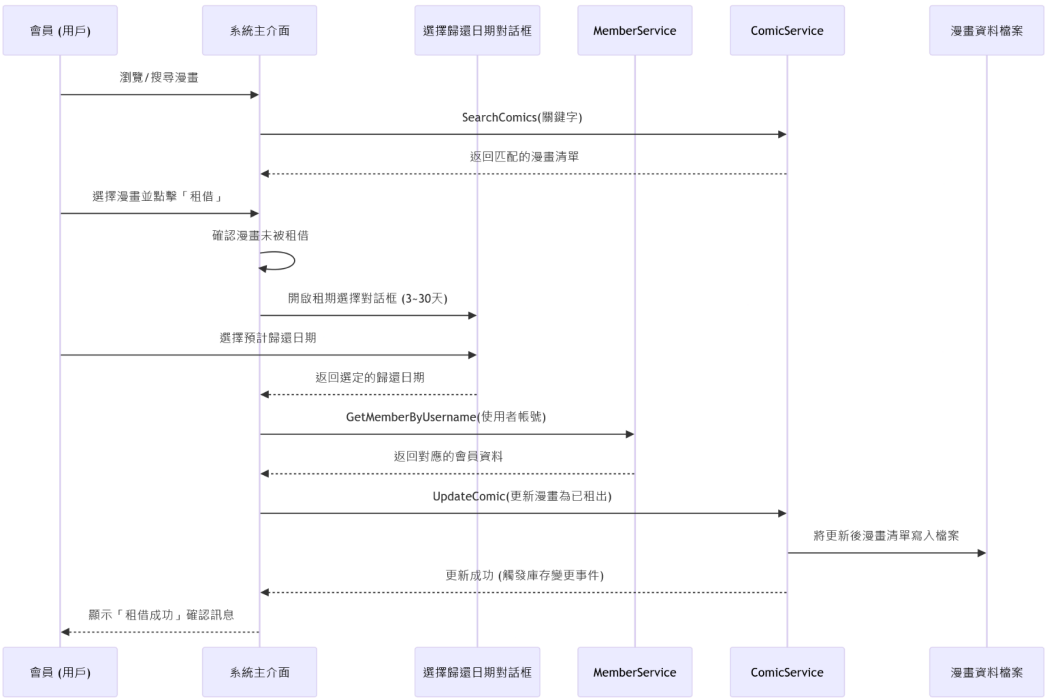
- 漫畫租借管理模組：此模組涵蓋漫畫出借和歸還的整個流程。管理員可透過租借作業介面選取要出借的漫畫以及借閱人，設定預計歸還日期後登記此次租借；當漫畫成功借出後，系統將漫畫狀態標記為「已租出」並記錄借閱者及預計歸還日。對於會員使用者，系統提供自行借閱的功能：會員登入後可瀏覽可借的漫畫清單並發起借閱請求，流程上與管理員代為操作相似，只是借閱人即為該登入會員本人。在漫畫歸還方面，管理員可在租借管理介面中選擇已借出的漫畫記錄並執行歸還操作。系統將要求輸入或確認實際歸還時間，然後更新漫畫狀態為可借、清除借閱者資訊並保存實際歸還時間戳記。租借管理模組確保每本漫畫同一時間僅能租給一位會員，並提供對預約歸還日與實際歸還日的紀錄，用以後續可能的逾期管理。

- 日誌與例外管理模組：為提升系統可靠度與方便偵錯，我們設計了一套日誌記錄與例外處理機制。所有服務模組在關鍵操作以及錯誤發生時都會寫入日誌檔案，如管理員登入、資料檔載入成功/失敗、租借或歸還操作結果等皆留下紀錄。同時，應用程式註冊了全域例外處理器，用以捕捉未預期的執行緒錯誤；一旦有未處理例外發生，系統將記錄詳細錯誤資訊至日誌，並以對話框通知使用者發生嚴重錯誤然後安全地終止應用程式。此模組的設計確保了當系統面臨異常情況時，能將影響降至最低並提供後續調查所需的資訊。

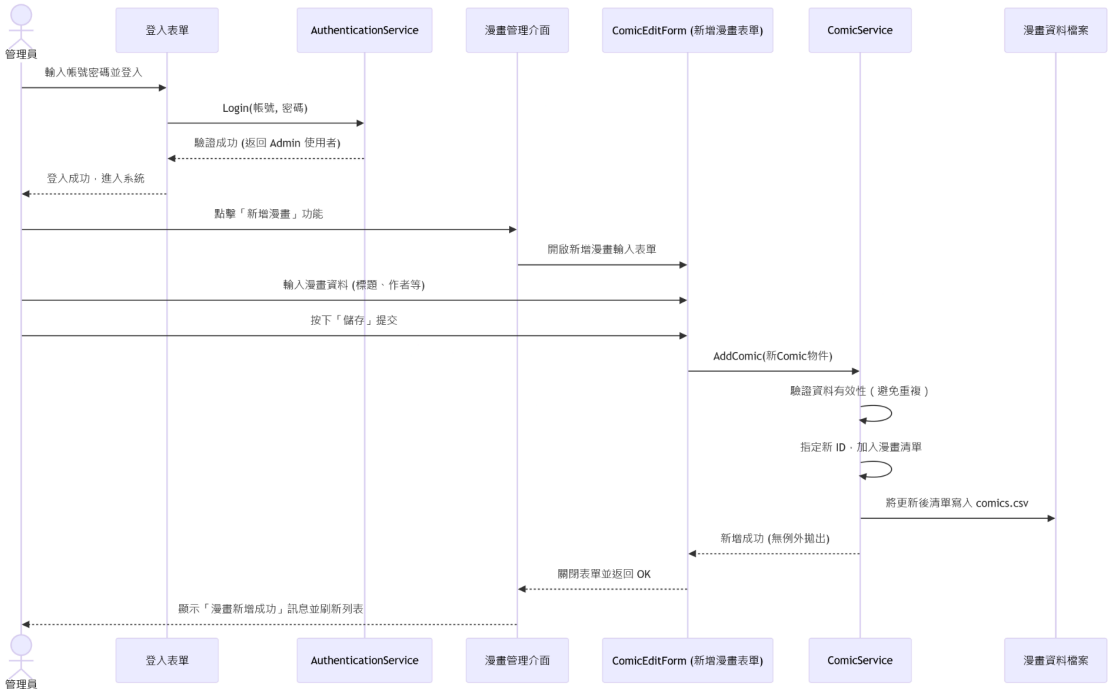
UML



圖二:類別圖



圖三:循序圖-會員



圖三:循序圖-管理員

第三章 系統實現

本章將說明漫畫租借管理系統在開發時的環境配置、具體功能的程式碼實現以及所應用的C#技術細節，最後並介紹系統的圖形介面操作方式。我們將重點闡述如何在程式中運用第8～15章所學的各種語言概念，並解析關鍵模組的實作原理，說明本系統是如何透過程式碼達成前章所定義的功能。

開發環境

本專題使用Microsoft Visual Studio 2022作為開發工具，目標框架為.NET 8.0 (Windows Forms)，使用C#語言進行程式撰寫。在Windows Forms的專案範本基礎上，我們建立了多個窗體和必要的類別。由於系統需要讀寫檔案及執行非同步任務，我們引用了`System.IO`命名空間中的檔案操作類別以及`System.Threading.Tasks`提供的Task並行功能。另外，為了方便JSON格式處理，專案也引用了`System.Text.Json`命名空間。整個開發過程均在Windows 11作業系統上進行，專案編碼使用UTF-8以正確處理中文顯示。

使用Visual Studio開發Windows Forms應用，使我們能夠透過設計工具直觀地繪製使用者介面佈局，例如拖拉按鈕、文字框、資料網格等控制項至表單，並設定其屬性。然後，在程式碼後置檔（Code-behind，如LoginForm.cs等）中撰寫事件處理函式來定義使用者互動行為。我們也

善用VS提供的偵錯功能及單步執行，輔以自訂日誌，在開發階段反覆測試每個模組功能，確保系統各部分都能正常運作。

程式碼實作概要

本系統由數十個類別組成，為求完整性不可能在此列出所有程式碼，以下將挑選部分關鍵的實作進行說明，以展示系統如何透過程式碼達成本專題的功能需求：

- 主程式初始化：在`Program.cs`主入口的`Main()`函式中，首先載入並初始化各服務物件。我們依序建立了`FileLogger`、`FileHelper`、`ComicService`、`MemberService`、`AuthenticationService`和`ReloadService`的實例，並以這些實例配置各服務。例如，`ComicService`建構時會同步讀取`comics.csv`載入漫畫清單；`MemberService`同樣載入`members.csv`會員清單。AuthenticationService則載入`users.json`帳戶列表。在完成服務初始化後，主程式註冊了兩個全域例外事件處理器：一個針對UI執行緒未捕捉例外（`Application.ThreadException`），一個針對非UI執行緒例外（`AppDomain.CurrentDomain.UnhandledException`），都指向我們自訂的處理函式。這些處理函式會將例外訊息寫入日誌並彈出錯誤對話框通知使用者。最後，程式以`Application.Run()`啟動`LoginForm`作為應用程式的第一個介面。至此，整個應用的基礎服務和錯誤監控已就緒。

- 帳戶註冊與登入：`LoginForm`中實作了登入與開啟註冊視窗的邏輯。當使用者在登入畫面點擊「登入」按鈕時，事件處理函式`btnLogin_Click`會擷取輸入的使用者名稱和密碼。首先檢查欄位不為空，然後呼叫`AuthenticationService.Login(username, password)`進行驗證。驗證服務對密碼採用雜湊比較：它會根據該使用者帳戶存儲的Salt，對使用者輸入的密碼進行Hash後與資料中Hash值比對。若密碼正確且帳戶未鎖定，Login方法返回User物件，否則返回null。LoginForm據此決定登入成功與否，成功則隱藏自身並開啟主窗體MainForm，將當前使用者與服務物件傳遞給MainForm；若失敗則顯示錯誤訊息且清空輸入欄位供再次嘗試。至於註冊，新使用者可在LoginForm點擊「註冊」按鈕開啟`RegistrationForm`註冊對話窗。RegistrationForm首先讓使用者填寫帳號、密碼、姓名、電話等資訊並進行欄位驗證（例如必填欄位不空、密碼一致性等），這部分利用了ErrorProvider來即時提示輸入錯誤。當使用者提交註冊時，RegistrationForm的`btnRegister_Click`事件處理函式會先呼叫`AuthenticationService.Register(username, password, role)`嘗試建立新帳戶。Register方法內部會檢查使用者名稱是否已存在；若不存在則產生隨機Salt並

計算密碼Hash值，建立User物件加入清單後存檔。我們預設一般使用者註冊得到的角色為Member；只有管理員從後台新增帳戶時才可選擇角色，所以RegistrationForm在非管理員使用時會隱藏角色選項下拉選單。當帳戶成功建立後，RegistrationForm隨即建立對應的Member物件（使用填寫的姓名、電話等）並呼叫`MemberService.AddMember(newMember)`新增到會員清單。此時特別考量資料一致性：如果會員資料添加過程中發生例外錯誤，我們在catch中會呼叫`AuthenticationService.DeleteUser(username)`將剛剛註冊的使用者帳戶移除，避免出現「有帳戶但無會員資料」的不一致情況。若兩者皆成功則提示註冊成功，並清空表單供使用者繼續操作或關閉。透過上述流程，系統實現了帳戶與會員資料的同步建立，並確保註冊操作的原子性（要麼全部成功，要麼完全不產生影響）。

- 漫畫與會員管理：管理員登入後，在主介面可以透過選單或按鈕開啟漫畫管理和會員管理兩個獨立視窗（對應`ComicManagementForm`與`MemberManagementForm`）。以漫畫管理為例，`ComicManagementForm`載入時會透過`ComicService.GetAllComics()`取得全部漫畫列表，並繫結到DataGridView顯示。管理員可點擊「新增漫畫」按鈕呼叫`ComicService.AddComic(comic)`新增一筆漫畫。AddComic實作時，會自動為新漫畫指派唯一的Id（取目前列表最大Id加1），將Comic物件加入內存清單後立即保存至檔案。如前所述，AddComic含有若干防呆邏輯，包括阻止添加重複Id的漫畫，檢查是否有同名同作者漫畫並警示。更新與刪除漫畫的操作類似：UpdateComic會尋找對應Id的漫畫並更新其欄位，再保存清單；DeleteComic則在刪除前檢查漫畫是否已租出，若已租出則拋出例外禁止刪除。成功刪除時將漫畫自列表移除並保存檔案。MemberManagementForm中的會員新增、編輯、刪除通過MemberService提供的方法實現，AddMember會為新會員分配Id並加入列表，UpdateMember更新找到的會員欄位，DeleteMember則在移除前使用ComicService檢查該會員是否有未歸還的漫畫。每當`MemberService`或`ComicService`的資料有所變更時，都會觸發各自的資料變更事件（`MembersChanged`或`ComicsChanged`）。這些事件在對應的管理表單載入時即有訂閱，目的是讓資料改變時UI自動刷新。例如，在MemberManagementForm中，一旦MemberService.MembersChanged事件被觸發，就呼叫重新載入會員列表以更新畫面。因此，如果管理員在另一介面新增了會員，已開啟的MemberManagementForm會即時反映新增結果。透過這種事件機制，可以確保多視窗之間資料的一致性和同步更新，而不需要使用者手動重新整理列表。

- 租借與歸還流程：租借流程可以由管理員或會員啟動，兩者介面略有不同但底層邏輯相同。當管理員在主視窗選擇租借管理功能時，會打開`RentalForm`租借作業視窗。在RentalForm載入時，我們會初始化下拉選單和表格：`cmbMembers`下拉列表載入所有

會員名稱，cmbComics載入目前可供租借的漫畫清單（篩選IsRented為false的漫畫），下方的dgvRentedComics資料網格則顯示所有已借出漫畫的列表，包括漫畫名稱、借閱者、預計歸還日等。這些資料由ComicService和MemberService聯合提供：例如，可借漫畫清單透過ComicService篩選，已借出列表則使用Comic列表與Member資料合併形成前述RentalDetailViewModel清單供DataGridView繫結。當管理員要出借漫畫時，需從會員下拉選單選擇借閱者、從漫畫下拉選單選擇要出租的漫畫，然後點擊「租借」按鈕。btnRent_Click事件處理流程如下：首先再確認使用者有選擇會員與漫畫；接著檢查選定漫畫是否當前已被租出，若已租出則終止操作並提示。若均正常，系統彈出一個RentalPeriodForm對話框，讓管理員從建議的期間中（比如最短3天，最長1個月）選擇一個預計歸還日期。使用者選定日期後對話框返回OK，程式繼續執行：設定該漫畫物件的狀態為已租出(IsRented = true)、記錄租給的會員ID以及租出日期和預計歸還日期。然後呼叫ComicService.UpdateComic(selectedComic)將變更寫入檔案並觸發ComicsChanged事件。ComicService內部UpdateComic實作時，會找到對應Id的漫畫物件並更新其所有欄位（包括剛剛修改的狀態和日期），最後保存檔案。完成後系統在介面上彈出訊息表示租借成功，以及提醒預計歸還日期。RentalForm接收到ComicsChanged事件時，會自動呼叫RefreshUIDataSafely()重新載入可借清單和已借列表，使得剛才租出的漫畫從「可借」列表中消失並出現在「已借出」列表中。對於一般會員流程，會員在主介面的「可借漫畫」列表選中一項後點擊「租借」按鈕（btnRentComic），觸發MainForm的事件處理函式。系統同樣檢查漫畫是否可借並彈出RentalPeriodForm讓會員選擇預計歸還日。但此時借閱者就是目前登入的使用者本身，因此程式直接利用_currentUser找到對應的Member資料。隨後的步驟與管理員操作一致：更新漫畫狀態、保存檔案、提示成功並刷新列表。差別在於會員界面租借成功後，會額外刷新會員自己的「我的租借」列表，以顯示新借的漫畫。

漫畫歸還方面，管理員使用RentalForm執行歸還。管理員在下方已借列表中選取要歸還的紀錄，點擊「歸還」按鈕btnReturn_Click。事件處理函式會取得選中列所繫結的RentalDetailViewModel，從中找到對應的漫畫ID。接著檢查該漫畫在服務層是否仍存在（防止資料不一致）以及目前是否狀態已是未租借（可能已被其他操作歸還）。如果一切正常，則執行歸還流程：程式取得此漫畫物件，準備更新其狀態。為精確記錄，此處我們提供一個介面元素（如DateTimePicker元件dtpActualReturnTime）讓管理員確認實際歸還時間。通常預設為目前系統時間，但管理員亦可手動調整（例如處理隔天掃描條碼補登昨日歸還的情況）。接著程式將漫畫物件的IsRented設為false，RentedToMemberId重置為0，並將ActualReturnTime設定為選定的歸還時間。然後呼叫ComicService.UpdateComic(comic)保存這些變更。ComicService更新漫畫時不會移除紀錄，而是保留該Comic物件但標記為可借，並寫入檔案中。如此可以保

存過往借閱歷史（儘管目前本專題未實作詳細的歷史查詢功能，但未來可透過ActualReturnTime不為空識別出借過的紀錄）。完成更新後，系統彈出提示通知漫畫成功歸還。RentalForm再次接收到ComicsChanged事件，UI列表將自動更新：歸還的漫畫從「已借出」清單中消失並重新出現在「可借漫畫」下拉選單中供下次租借。對會員而言，若允許會員自行在介面歸還（本系統僅提供管理員執行歸還，但會員可查看自己的借閱紀錄），則流程同理。最後，RentalForm在關閉時會呼叫`ReloadService.Stop()`停止自動刷新執行緒，並解除訂閱事件，以釋放資源。

- 其他技術實作細節：本系統還包含一些提高使用體驗和可靠性的輔助實作。例如，在UI介面上，我們對多個控制項進行了統一風格設定，運用了委派與事件機制：透過在ModernBaseForm中編寫如`StyleModernButton(Button)`、`StyleModernDataGridView(DataGridView)`等方法，應用程式在表單初始化時呼叫這些方法來套用統一樣式。此處運用了方法多載概念（第十一章）：例如`ILogger`介面中`Log`方法有兩種簽名，一種接受字串訊息，一種接受字串加Exception；FileLogger對其進行了實作，兩者行為稍有差異（有Exception時寫入完整堆疊）。我們也大量使用了事件處理（第十三章）：Windows Forms的事件模型允許我們將按鈕點擊、文本變更等UI事件關聯到自訂函式。例如LoginForm在建構時將`btnRegister.Click`事件綁定到`btnRegister_Click`處理函式。再如MainForm中，為提供即時搜尋過濾，我們處理了文字框`txtSearchAvailableComics.TextChanged`事件和下拉框`cmbGenreFilter.SelectedIndexChanged`事件，只要偵測到文字或選項改變且目前處於會員的「瀏覽可借漫畫」頁籤，就呼叫篩選函式即時更新漫畫清單。執行緒與非同步（第十二章）方面，本系統利用了`Task.Run`啟動背景執行續執行定時任務：ReloadService的Start方法使用`Task.Run`建立一個while迴圈，內部使用`await Task.Delay(interval)`達到定期的效果。每次延遲完畢後即呼叫傳入的reloadAction（即ComicService.ReloadAsync與MemberService.ReloadAsync的組合）執行資料重載。這樣做的好處是避免阻塞UI執行緒，保證介面在長時間運行任務時仍然流暢。當我們關閉RentalForm時呼叫`ReloadService.Stop()`發出Cancellation取消令牌，中止該背景執行緒的迴圈。此處使用了CancellationToken機制來安全地結束執行緒。例外處理方面也遍佈於程式中：所有檔案讀寫操作都包在try-catch中，一旦遇到IOException會記錄錯誤並再次拋出；在進行資料解析時捕捉FormatException等格式錯誤，以提示是哪一行資料有問題；UI事件處理裡對重要的邏輯也都有基本的防禦性檢查並給出對應提示，例如租借時未選擇漫畫/會員則return並MessageBox提示。這些細節都提高了系統在非預期情況下的健壯性。

以上從多個角度說明了系統的關鍵程式碼實現和技術要點。在開發過程中，透過良好的物件導向設計，我們有效地在程式中運用了各種C#語言機制，將系統需求轉化為穩健的程式碼。下一節將特別總結我們如何運用課程第8～15章所涵蓋的C#概念，並在本系統中加以體現。

C#語言概念應用與解析

本專題高度重視將課程中學到的核心C#語言概念融入實作中。以下列出第8～15章涵蓋的重要概念，並說明其在本系統中的具體應用：

- 字串與陣列（第八章）：本系統大量使用了字串處理和陣列/集合操作。例如，檔案讀寫模組中使用`File.ReadAllLines`讀取CSV檔案為`string[]`陣列，每個元素對應一行資料；接著透過`Split`或逐字元解析將每行字串拆解成欄位清單。為處理CSV中的特殊字元，我們使用了`StringBuilder`來累積字串並判斷引號和逗號。在格式輸出方面，使用字串插值 (`$"..."`) 或`string.Format`組合文字，例如組裝日誌訊息、匯出CSV行等。此外，`Comic.ToCsvString()`與`Member.ToCsvString()`方法中，我們使用字串的`Replace`函式將欄位中的引號轉義為雙引號，再用字串插值串接成逗號分隔的一行。總體而言，字串處理對於資料序列化與UI訊息顯示非常重要，而陣列和集合（如`List<T>`）則是承載多筆資料的主要容器，我們頻繁地對清單進行查詢（LINQ的`FirstOrDefault`、`Where`等）、排序與迭代，充分體現了第八章內容在實作中的運用。

- 類別與物件（第九章）：本系統採用了嚴格的物件導向方式進行建模。我們為核心實體建立了專門的類別：例如`Comic`類別封裝漫畫書本的所有屬性和操作，`Member`類別封裝會員資訊，`User`類別封裝帳戶資訊。透過類別的建構函式與方法，我們定義了如何建立物件以及物件可以執行的行為，例如User建構時預設登入嘗試計數為0。服務類別如`ComicService`和`MemberService`各自維護類別的集合，並提供對外的業務方法，在內部操作這些物件。這體現了封裝的精神：外界程式碼不直接操作集合，而是透過服務提供的方法（Add、Update等）來對內部物件進行修改。我們也運用了物件初始化的語法來創建物件以增加可讀性，例如在RegistrationForm中，用`new Member { Name = name, PhoneNumber = phone, Username = username }`直接給屬性賦值。此外，物件的等值比較、複製等等在某些情況也用到，例如在搜尋會員或漫畫時會將傳入的鍵值與物件屬性比對（使用`Equals`或LINQ條件）。總之，第九章的類別與物件概念在本專題中無處不在——我們以類別來抽象真實世界的概念（漫畫、會員、租借紀錄等），並通過物件彼此交互來完成系統功能。

- 繼承與介面（第十章）：繼承方面，我們定義了`BaseEntity`、`ModernBaseForm`等父類別讓多個子類共享共通屬性或方法。一方面，資料模型的繼承（如Comic與Member繼承BaseEntity）簡化了主鍵Id的管理；另一方面，UI類別的繼承（所有表單繼承BaseForm）使得我們只需在BaseForm實作一次LogActivity方法，所有子表單便都有此功能。繼承亦體現於多型的應用，例如BaseForm定義了`protected virtual void OnFormClosing(...)`並在子類MemberManagementForm中覆寫（override）它以增加解除事件訂閱的行為。介面方面，我們創建並使用了數個介面：`ILogger`、`IFileHelper`、`IReloadService`等等，用以定義功能契約並提高系統彈性。實作這些介面的類別（FileLogger、FileHelper、ReloadService）各自提供了具體的行為實現。透過介面，我們也實現了鬆耦合設計——例如AuthenticationService只認識ILogger介面，不關心具體是檔案日誌或其他日誌，這樣我們可以在不改變AuthenticationService程式碼的前提下替換不同的Logger實現。這充分說明了繼承與介面在架構上的價值：繼承用於碼碼重用與結構分類，介面用於規範行為與靈活擴充。

- 方法過載與多型（第十一章）：在本專題中，我們有多處用到了方法的過載（Overloading）技術。同一個方法名稱，依據參數不同而提供不同實現。例如`FileHelper`提供了`WriteFile`方法的兩種版本：`WriteFile(string fileName, string content)`用於簡單寫入純文字檔案；以及`WriteFile<T>(string fileName, IEnumerable<T> items, Func<T,string> formatter)`用於將物件集合序列化後逐行寫入。再如`ILogger.Log`方法我們也進行了過載，一個簽名接受Exception以便同時記錄錯誤詳情。藉由方法過載，使介面更友好，同名的方法依輸入不同可完成相關的動作。**多型性（Polymorphism）**則體現在物件以父類或介面型別操作的靈活性。比如，我們宣告`ILogger _logger`，可以賦值FileLogger或其他Logger的實例給它，而對`_logger.LogError(...)`的呼叫將在執行期綁定到具體實現。又例如，在Global exception handler中捕捉到例外後，我們統一呼叫`AppLogger.LogError(...)`，但`AppLogger`可能是不同Logger類別，只要實作了ILogger介面即可，這就是介面多型的威力。還有一個隱含的多型應用是Windows Forms事件處理，像`dgvAvailableComics.SelectedRows[i].DataBoundItem as Comic`，由於資料繫結可以是Comic或其他型別，我們用`as`進行轉型並檢查，若是Comic則據此操作。C#的多型機制讓我們的程式在可讀性和靈活性上都有提升。

- 例外處理、委派與執行緒（第十二章）：我們非常注重例外（Exception）的處理。所有可能發生錯誤的地方都使用try-catch捕捉，例如檔案存取、字串轉換、集合操作等，一旦發生例外就記錄並妥善處理。全域未捕捉例外也由我們註冊的處理器統一處理，以防止程式意外崩潰。委派（Delegate）在本系統主要用於事件機制和函式做為參數。例如

`FileHelper.ReadFile<T>`方法接受一個`Func<string, T> parseFunc`委派，用於將讀入的每行字串轉換成T物件。我們呼叫時傳入了如`Comic.FromCsvString`這樣的靜態方法作為委派，FileHelper內部就能通用地使用該委派處理不同類型資料。另一個委派的使用是ReloadService的Start方法，它的參數`Func<Task> reloadAction`也是一種委派，用來回呼執行我們定義的重新載入功能。在Start內部的Task執行緒會定期呼叫這個委派，達到我們設計的自動刷新效果。執行緒的應用則體現在前述ReloadService採用了背景Task，以及主程式標示為[STAThread]啟動單執行緒公寓模型等等。為避免多執行緒更新UI的不安全，RentalForm的Service_DataChanged事件處理有檢查`this.InvokeRequired`並使用`this.Invoke(...)`將UI更新切回UI執行緒執行。這些都是正確使用多執行緒的關鍵。此外，我們在適當場合使用`lock`關鍵字鎖定資源，例如ComicService用一個`_comicsLock`物件在多執行緒環境下保護漫畫清單的讀寫，避免資料競爭條件發生。第十二章的概念透過以上實踐在系統中得到運用，使我們的應用兼顧了異步效率和執行安全。

- 事件處理（第十三章）：事件是C#的重要語言特色，在本專題隨處可見。首先，.NET提供的GUI事件如按鈕點擊、表單載入、TextBox文字改變等皆在UI程式碼中被處理，我們使用VS的設計工具或手動程式碼將這些事件連結到對應函式，實現了View和程式邏輯的互動。而更有價值的是我們自定義的事件，用於在不同模組間傳遞訊息。例如`ComicService`宣告了`public event ComicDataChangedEventHandler? ComicsChanged`事件，在每次漫畫資料改變後以`ComicsChanged?.Invoke(this, EventArgs.Empty)`觸發通知。UI中的RentalForm和MainForm在初始化時訂閱了此事件，。因此當租借或歸還導致漫畫清單變化時，這些介面可以即時收到通知並更新顯示。同理，MemberService的MembersChanged事件讓會員管理介面隨資料改變自動刷新。這種觀察者模式的運用極大降低了模組間的耦合，UI並不需要定期查詢資料是否有變，而是被動等待事件通知，提升了效率與即時性。另外值得一提的是，在RegistrationForm中我們使用了ErrorProvider與TextBox的Validating事件搭配，實現即時欄位驗證提示。只要使用者離開輸入框且內容不符合要求，就觸發Validating事件，在其中設定ErrorProvider的錯誤訊息，讓使用者體驗到互動式的表單驗證。這些都是事件機制活用的具體體現。

- 多表單與清單控制項（第十四章）：本系統的UI介面由多個Windows Form組成，包括登入、主介面、會員管理、漫畫管理、租借作業、註冊表單等，充分說明了多表單應用的管理。我們在表單間傳遞資料（如將服務物件和當前使用者傳給MainForm），使用`Show()`或`ShowDialog()`控制表單的模式，並在需要時透過事件（如

MainForm.FormClosed）讓先前表單關閉。在介面導航上，管理員主窗體含有選單（MenuStrip）或按鈕來打開各子功能窗體，會員主窗體則使用TabControl切換「瀏覽漫畫」與「我的租借」兩個頁面。清單控制項方面，我們使用了**DataGridView**顯示漫畫和會員的列表，以及顯示租借的詳細紀錄。DataGridView可以自動生成欄位或由程式動態建立欄位，我們選擇在程式中明確定義欄位以控制顯示格式，例如設定寬度、標題文字以及日期欄位的格式。我們也使用了ComboBox下拉清單作為選擇控制，例如會員清單下拉讓管理員選借閱人、類型篩選下拉讓會員按類型篩選漫畫。ListView或ListBox在本專案未直接使用，但透過DataGridView和ComboBox等清單型控制項，實現了資料的批次呈現和選取功能。值得一提的是，在實作過程中我們也處理了一些常見的介面問題，例如避免DataGridView在沒有資料時的異常操作、設定SelectionMode確保一次只選取一行等等，這些細節保障了清單控制項操作的順暢性。

- 檔案與資料夾處理（第十五章）：系統涉及多種檔案操作，例如讀取/寫入文字檔、建立資料夾、檔案備份與刪除等。所有這些都由`FileHelper`類別集中管理。啟動時，FileHelper的建構子使用`Directory.CreateDirectory()`確保資料夾存在；透過`Path.Combine`組合路徑將檔名定位到應用的資料夾下。讀取檔案方面，使用了`File.Exists`先行檢查檔案存在與否，若不存在則返回空結果或預設值。讀取文本時採用`File.ReadAllText`或`File.ReadAllLines`來簡化實作，同時指定編碼為UTF-8以支援中文。寫入檔案則使用`File.WriteAllText`或`File.WriteAllLines`在一次呼叫中完成所有內容寫入，避免多次IO。在WriteFile實作中，我們先將物件集合轉換為字串清單再一次性寫入，這樣能確保資料的完整性（萬一中途失敗，不會只寫入一半內容）。此外，FileHelper還提供了檔案操作的輔助方法如`FileExists`、`DeleteFile`、`MoveFile`、`CopyFile`。雖然目前系統未大幅用到檔案搬移或複製，但我們在升級備份檔時可能用到Rename(透過MoveFile實現)，在手動備份資料時亦可用CopyFile複製整個資料夾。整體而言，本專題在檔案處理上貫徹了第十五章強調的正確IO操作方式，所有檔案開啟寫入都置於try-catch中處理IO例外，並注意釋放資源（由於使用ReadAllText等一次性方法，內部已封裝了檔案流的開關，因此未自行撰寫Finally關閉檔案的程式碼）。此部分的實作確保了資料的可靠保存，並提供了基本的檔案管理能耐，為系統的資料永續性保駕護航。

GUI操作說明

本節將說明漫畫租借管理系統的圖形使用者介面的操作方式，分別從管理員和一般會員兩種角色的角度介紹系統功能的使用流程。透過這些敘述，可以驗證前述功能實作是否達到了預期效果，同時也為最終使用者提供使用手冊式的指南。

1. 管理員介面操作

- 登入：啟動應用程式後將首先看到登入視窗。管理員使用預設提供的帳號（例如admin）和密碼登入系統。若密碼遺忘，可使用資料庫初始值或聯絡系統維護人員重置。登入時系統會檢查帳密，若連續5次輸入錯誤，帳戶將被鎖定15分鐘，此時管理員需稍候再嘗試。登入成功後，登入窗體自動關閉，系統進入管理員主畫面。

- 主畫面與儀表板：管理員主畫面上方具有選單（或功能按鈕列），可存取各管理功能；中央區域顯示「管理員儀表板」，提供系統當前概要資訊，包括漫畫總藏量、目前已出租漫畫數量、可借閱漫畫數量，以及註冊會員總數等。這些指標在管理員登入時即自動載入顯示，協助管理員迅速了解系統狀態。儀表板數據會隨著後續操作實時更新（例如新增漫畫後漫畫總數增加，出租漫畫後已租數量增加等）。

- 會員管理：管理員可點擊選單「會員管理」打開會員管理視窗（MemberManagementForm）。該視窗上方一般會有「新增會員」「編輯會員」「刪除會員」等按鈕，下方是會員列表資料表格，顯示每位會員的ID、姓名和電話等資訊。新增會員時，點擊對應按鈕將彈出會員編輯對話框（MemberEditForm），可輸入會員姓名和電話後確定即可完成。由於會員帳號是在註冊時產生，管理員透過此功能新增的會員通常對應沒有使用者登入帳號（或由系統自動以姓名當作帳號並產生預設密碼）。編輯會員則允許修改姓名或電話等資料。刪除會員前系統會自動檢查其借閱狀態，如有未歸還的漫畫則拒絕刪除並提示。成功新增、修改或刪除會員後，會員列表會自動刷新顯示最新結果，無需手動重新整理。

- 漫畫管理：點擊選單「漫畫管理」開啟漫畫管理視窗（ComicManagementForm）。此視窗列出所有館藏漫畫，每列包含漫畫ID、書名、作者、ISBN、類型以及當前狀態（可出借或已借出等）。管理員可使用「新增漫畫」按鈕來錄入新漫畫資訊，包括書名、作者等欄位並提交保存；也可選中列表中某一漫畫點擊「編輯」修改其資訊（例如修正拼寫錯誤或更換分類）；或在確定不再收藏某漫畫時使用「刪除」移除。在編輯或刪除操作完成後，漫畫列表同樣會即時更新。若嘗試刪除已借出的漫畫，系統會阻止並顯示警告，這點請管理員留意處理先歸還再刪除的順序。漫畫管理介面還可能提供搜尋或篩選功能：例如可在頂部的搜尋框輸入關鍵字以過濾顯示書名或作者包含該關鍵字的漫畫，或透過下拉選單篩選只看某一類型的漫畫列表，方便管理員快速定位特定書目。

- 租借管理：當需要辦理漫畫出借或歸還時，管理員可打開「租借管理」視窗（RentalForm）。視窗左側通常提供會員選擇下拉框以及漫畫選擇下拉框。管理員首先從會員清單中選擇借閱人，再從可借漫畫清單中選擇要借出的漫畫。為方便挑選，漫畫下拉清單只顯示目前狀態為「可借」的書名，並可搭配上方的搜尋框快速篩選。選擇後點擊「租借」

按鈕，系統會彈出設定預計歸還日期的小視窗：管理員依照圖書館規定，通常可選擇3天後至30天內的一個日期作為預計歸還期限。確認日期後，系統自動完成租借登記並提示成功訊息。該漫畫即時從可借清單移除，並出現在下方「租借紀錄」列表中。租借紀錄列表列出所有尚未歸還的借閱資料，包括漫畫ID/名稱、借閱會員姓名、借出日期和預計歸還日等資訊，以供管理員隨時查閱。當會員歸還漫畫時，管理員在此列表中點選相應紀錄（可多選同時處理多本歸還），然後按「歸還」按鈕。系統將要求確認實際歸還時間（預設為當前時間），管理員可視情況調整，然後提交，系統便更新紀錄並將漫畫狀態設為「可借」。成功歸還後，該筆紀錄從列表中消失（表示租借已閉合完成），該漫畫也回到可借清單中。需要注意的是，若會員逾期歸還（超過預計日期），系統目前僅記錄實際歸還時間，並未實作罰款計算或提醒，這部份可作為日後升級功能。最後，管理員完成當前所有租借管理工作後，可關閉租借管理視窗回到主介面。日誌檔會完整記錄每一筆出借與歸還操作，有助於日後核對。

- 帳戶管理與其他：若管理員需要新增新的管理員帳戶或重設某使用者密碼，則可以透過「帳戶註冊」功能（即登入畫面的註冊，也可從管理員後台呼叫RegistrationForm並選擇角色為Admin）創建新帳戶。為安全起見，建議新帳戶建立後通知該用戶盡快登入並修改密碼（本系統未實作密碼修改介面，可透過直接編輯users.json或另行開發功能實現）。對於封鎖的帳戶，管理員目前只能等待鎖定時間過去或手動編輯資料檔解除鎖定（未提供介面操作）。系統的日誌檔`ComicRentalSystemLog.txt`可在需要時打開查看，其中詳細記錄了管理員各項操作以及系統自動產生的警告/錯誤訊息，管理員可定期巡查以發現潛在問題。當管理員欲結束使用時，可在主介面選擇「登出並退出」，系統將關閉所有視窗並安全地終止，日誌中會寫入應用程式關閉的記錄供備查。

2. 會員介面操作

- 登入：會員使用自己的帳號密碼在登入畫面登入。首次使用系統時，會員帳號可能由管理員預先建立並告知初始密碼，或會員可自行透過註冊功能創建帳戶。若會員輸入錯誤密碼，也有相同的五次錯誤鎖定保護。登入成功後，進入會員主介面。

- 主畫面：一般會員登入後的主窗體界面與管理員不同，系統會自動切換到會員視圖。畫面通常包含一個選項卡(TabControl)區域，分為「可借漫畫」和「我的租借」兩個頁籤。預設顯示「可借漫畫」頁面，列出目前所有可供借閱的漫畫清單；切換到「我的租借」頁面則可以查看該會員自己當前借閱中的漫畫列表。

- 瀏覽與搜索漫畫：在「可借漫畫」頁面，會員可以瀏覽所有尚未被借出的漫畫資訊。為方便尋找想借的漫畫，介面提供了上方搜尋框和類型下拉篩選。例如會員想找特定作者

的漫畫，可在搜尋框輸入作者名關鍵字，列表將即時過濾出作者名包含該關鍵字的漫畫；或選擇類型下拉選單中的某類（如「冒險」）來僅顯示該類型的漫畫。每項漫畫通常顯示書名、作者、類型等基本資訊。會員若對某本漫畫有興趣，可點擊該項使其反白選取。此時按下「租借」按鈕（通常在列表下方或側邊），系統將進行租借流程。

- 租借漫畫：會員點擊「租借」後，會跳出一個設定預計歸還日期的對話框（流程與管理員操作類似）。會員需要從建議範圍中選擇一個歸還日期並確認。如果選擇的日期不合理（例如小於當天，或超過允許的最長天數），系統會提示錯誤要求重新選擇。日期確認後，系統立即辦理借閱：該漫畫狀態變為已被租借，不再出現在可借清單中，同時在「我的租借」頁面的列表中新增一條借閱紀錄。畫面會彈出訊息告知「租借成功」，內容包含所借漫畫名稱以及需在何日期前歸還。至此，會員已成功借到漫畫，可以前往圖書館領取實體漫畫（若本系統應用於線下館藏）。

- 查看與管理我的租借：會員可隨時切換到「我的租借」頁籤查看自己目前借了哪些漫畫。列表中每條紀錄包含漫畫名稱、借閱日期以及預計歸還日期，方便會員了解哪天需要歸還哪本書。如果會員在到期前歸還了漫畫，管理員會在後台執行歸還操作並通知會員。會員可在稍後登入系統時發現「我的租借」列表中該項已消失，表示系統已確認歸還。如需歸還漫畫，會員可親自到館將書交給管理員，然後等待管理員在系統中登記歸還（會員介面本身不允許自行標記歸還，以確保實物交還）。本系統未實現會員自行續租或預約功能，如果會員希望延長借閱時間或預約已借出的漫畫，需透過線下或其他途徑聯繫管理員處理。

- 其他：會員可以使用登入畫面的註冊功能創建帳戶，創建時系統預設角色為會員，不需額外設定。會員註冊成功後，可立即登入使用。若會員遺忘密碼，目前系統並無找回密碼的自助機制，需請管理員在後台刪除該帳戶並重新建立或修改`users.json`中的Hash值（屬進階操作）。日誌方面，會員操作（如登入成功/失敗、借閱操作）也會記錄於系統日誌，但會員無介面直接存取日誌檔，此部分由管理員監控。當會員完成所有需要的操作後，可以直接關閉主窗體以退出系統。下次再次啟動應用即可重新登入。

總的來說，管理員和會員兩種角色所見的介面和可用功能有所差異，但整體操作流程都遵循人性化的設計：提供明確的提示、必要的防呆（如未選擇項目就點擊按鈕時會有警告），以及簡潔的界面佈局以減少使用者的學習成本。

第四章 測試與驗證

開發完成後，我們對漫畫租借管理系統進行了多方面的測試，以驗證系統功能的正確性、資料的一致性以及在異常情況下的表現。本章將介紹部分重要的測試案例和結果，包括整單元測試、合測試，以及針對資料正確性和例外處理機制的驗證，從而證明本系統符合預期需求並具備良好的穩定性。

單元測試案例

由於專題的許多功能屬於邏輯處理部分（非UI），我們撰寫了一些針對服務類別的單元測試案例來逐一驗證。在開發環境中使用手動呼叫方法或建立測試程式來執行。

- **CSV** 解析與格式測試：針對`Member.FromCsvString()`和`Comic.FromCsvString()`方法，我們設計了多組測試字串。例如，輸入一行格式正確的會員CSV字串：`1,"王小明","0912345678","willywang"`，期望解析後得到Member物件各欄位對應值正確（Id=1, Name="王小明", PhoneNumber="0912345678", Username="willywang"）。接著我們測試特殊情況：如姓名中包含逗號或引號的情形`2,"陳""阿明， Jr.", "0987654321","amchen"`（注意:資料中`陳"阿明， Jr.`包含引號和逗號），檢查解析結果是否仍能正確取出`Name=陳"阿明， Jr.`。結果顯示，我們的CSV解析函式能正確處理引號轉義和逗號，Member物件內容無誤，說明`ParseCsvLine`函式穩健可靠。同樣地，測試Comic CSV解析也通過：特別對日期欄位進行檢查，例如給定RentalDate字串`2025-06-01T15:00:00`，看是否轉成相應的DateTime物件。測試證明日期格式正確可解析，而非法格式（如`2025/06/01`）會導致TryParse失敗進而將日期設為null，符合我們在程式中的預期處理。

- 服務方法邏輯測試：我們對ComicService和MemberService的一些關鍵方法進行了單元測試。例如：

    - 測試`ComicService.AddComic()`：準備一個新Comic物件，其Id設為0（表示需服務分配），其他欄位填入有效資料。呼叫AddComic後，我們檢查返回的漫畫列表是否包含此物件、物件是否被賦予了非0的Id，以及`comics.csv`檔案末行是否新增了相應紀錄。測試結果表明新增成功，且系統為新漫畫自動生成了正確的Id（例如原有漫畫最大Id為10，新增後物件Id變為11）。

    - 測試`ComicService.DeleteComic()`：情境1，嘗試刪除一個不存在的漫畫Id，預期應擲出InvalidOperationException。我們傳入Id=999（假設不存在），程式果然拋出例外且未更動檔案內容。情境2，嘗試刪除一個已租出的漫畫：我們

先將某漫畫的IsRented設為true、RentedToMemberId指定某會員，模擬它已被借出。此時呼叫DeleteComic，預期不允許刪除並拋出例外。測試驗證了這點，例外訊息符合預設的"無法刪除漫畫: 漫畫目前已租借。"。

- 測試`MemberService.DeleteMember()`：類似地，我們分兩種情況測試：刪除不存在Id的會員應失敗拋例外；刪除有租借中漫畫的會員應被阻止。我們讓一會員的Id與某Comic的RentedToMemberId一致，然後DeleteMember該會員，觀察到MemberService內部透過ComicService.GetAllComics()檢測到租借關聯，正確拋出了例外且會員未刪除。而刪除無借書的會員則測試通過，會員列表更新且檔案紀錄移除該行。

- 測試`AuthenticationService.Login()`：我們模擬不同場景下的登入行為。首先，用正確的帳密測試，應得到User物件且FailedLoginAttempts歸零。接著連續輸入錯誤密碼5次，預期第五次後帳戶LockoutEndDate被設為15分鐘之後且Login返回null。檢查users.json內容，發現對應使用者的FailedLoginAttempts變為5且LockoutEndDate有正確的時間戳。隨後嘗試在鎖定期間再次登入，Login方法立即返回null且不進行Hash比對，與我們程式邏輯一致。最後，測試輸入空白帳號或不存在帳號的情形，Login應返回null並記錄警告日誌。以上測試確保了登入機制的每個分支路徑都正常運作。

- 資料一致性測試：此類測試關注多個模組互動後資料是否保持一致。我們模擬一些操作序列，如「新增會員->刪除該會員」，「新增漫畫->租借->歸還->刪除該漫畫」，在每一步之後驗證所有相關模組資料同步更新。例如，對於新增漫畫再租借的測試：新增漫畫成功後，ComicService的列表數比之前多1且檔案新增一行；接著以某會員租借此漫畫後，ComicService列表中該漫畫的IsRented=true、RentedToMemberId正確，MemberService列表不變但我們檢查RentalForm顯示是否更新——租借紀錄應新增，該漫畫不再列在可借清單。歸還後，檢查漫畫IsRented=false且ActualReturnTime有值，租借紀錄從介面移除。最後刪除漫畫，漫畫列表數減1且檔案該行消失。這整個流程跑下來，系統在UI和資料檔案兩方面的表現均符合預期。我們還特別檢查了users.json、members.csv、comics.csv之間的關聯：每當註冊新會員，同時有新User和新Member記錄；刪除會員不會自動刪帳戶（本系統允許存在無會員資料的帳戶，以備比如管理員帳戶不需要會員條目），這也是課程上討論過的資料設計權衡點之一。測試確認了我們在設計時的取捨並無引發明顯問題。

整合測試與使用情境驗證

整合測試側重於模擬實際使用情境，來驗證系統在真實操作流程中能否正確運作，且各模組協同工作無誤。以下列出幾個我們進行的情境測試：

- 情境**1**：新會員註冊並借閱漫畫 – 流程：打開應用並使用註冊功能建立帳號（如 user:testuser, pwd:test123, 姓名:張三, 電話:...）。註冊提交後預期資料檔更新，我們檢查users.json應新增testuser帳戶、members.csv新增姓名張三。接著用此帳號登入（驗證密碼Hash+Salt計算正確），進入會員介面。瀏覽可借漫畫列表並選擇一本漫畫借閱。我們在管理員介面同時觀察到該漫畫狀態改變，租借紀錄新增。這時登出再以管理員登入，確認該會員張三列在會員列表中，且張三的租借紀錄正確顯示在管理員的租借管理界面。我們也嘗試了讓該會員再次登入查看「我的租借」，資料一致。最後以管理員身份登記張三歸還漫畫，再檢查testuser登入時「我的租借」清單變空。此情境涵蓋了註冊->登入->租借->歸還的一連串操作，結果證明系統整體流程順暢，資料傳遞正確無誤。

- 情境**2**：管理員完整管理操作 – 流程：管理員登入後，依次進行多項管理操作：新增3本新漫畫、修改其中1本的資訊、刪除另1本漫畫；新增2個新會員資料；將其中1位會員升級為管理員權限（透過RegistrationForm選擇角色Admin）；將2本漫畫借給剩下一位會員，再嘗試刪除該會員（預期被阻止），然後將漫畫收回歸還，再刪除該會員。這個複合情境涵蓋了各模組的所有功能點。測試結果觀察：

    - 漫畫新增/修改/刪除：UI列表和檔案均更新正確，被刪除的漫畫不再出現在任何清單，修改的資訊在UI和CSV中都得到更新。

    - 會員新增/升級/刪除：新會員出現在會員列表中，升級的帳戶能成功以管理員身份登入主界面（我們切換帳戶登入驗證了角色變化）。嘗試刪除有借書的會員系統彈窗阻止了，符合預期。當其借書全部歸還後再刪除則成功，刪除後會員列表無該姓名且members.csv中該筆資料消失，而對應的users.json帳戶因我們選擇不自動刪除仍然存在（這是系統目前的設計，我們將在未來工作討論改善）。

    - 整個操作過程無崩潰或資料異常，所有介面反應及時。例如刪除漫畫後RentalForm的下拉清單立即少了一項，新增會員後MemberManagementForm自動刷新列表等，都得益於先前提及的事件機制。從管理員角度體驗來看，系統能處理連續不斷的不同任務，且資料狀態維護正確。

- 情境**3**：異常情境模擬 – 我們也模擬了一些可能的異常情況以測試系統的健壯性：

- 檔案遺失/損毀：手動刪除或破壞某些資料檔，然後運行系統觀察行為。我們嘗試刪除users.json再啟動，系統日誌出現"找不到使用者檔案，初始化空清單"警告且自動創建admin帳戶，登入畫面能正常登入admin。模擬comics.csv內容亂碼導致JsonException（例如在users.json中插入非法JSON），AuthenticationService能捕捉此錯誤並轉向嘗試讀取備份檔，如果備份也沒有則初始化空清單並報錯。MemberService/ComicService讀取時若遇到格式錯誤，會拋ApplicationException，我們在調試模式下觀察到程式在Main()啟動服務時即終止並彈錯誤對話框告知嚴重錯誤（因為我們沒有捕捉該例外，使其進入全域處理並關閉程式）。雖然這導致系統無法繼續，但至少錯誤被及時發現並未造成更隱蔽的資料錯亂。這也提示使用者/管理員應注意備份檔的重要性和檔案操作謹慎。

- 執行緒競態：由於我們採用了多執行緒自動刷新，我們測試了在頻繁租借/歸還同時自動刷新是否會出問題。經過多次在RentalForm中快速點擊租借、立即歸還、再租借等操作，以及降低ReloadService的間隔到3秒以更頻繁刷新，系統仍能正常工作，無死鎖或例外。DataGridView的Invoke更新機制確保了非UI執行緒更新UI的安全性。我們甚至嘗試同時開兩個RentalForm視窗（實際使用不會這樣，但為測試線程安全），兩者透過同一服務對象操作也未發生不可預期的錯誤，只是介面顯示上可能有些重複。

- 邊界條件：測試極端或邊界資料輸入。例如會員電話輸入非常長的數字字串，系統可以接受並存檔（對長度無特別限制，但UI顯示可能溢出欄位，我們在UI上限制了最多輸入位數避免這種情況）。再如漫畫書名留空，只輸入作者等，經測試AddComic允許書名為空字串（這也許不合理但程式未禁止，我們在使用者指引上會建議填寫完整資訊）。帳號名稱大小寫差異測試也做了：系統Login對Username比較使用了不區分大小寫的方式（即'user1'和'User1'視為同一帳戶），測試驗證了這點，避免了可能的重覆帳號問題。

資料正確性與例外處理驗證

資料正確性部分已在前述測試中有所體現，這裡總結重點：

- 正確寫入與持久化：每當進行資料修改操作後，我們都檢查對應檔案內容是否同步更新。例如新增10個漫畫後`comics.csv`行數正好新增10行且欄位正確；編輯某會員電話後`members.csv`該會員記錄電話欄位更新；這些都能對上。我們甚至使用文本比對

工具比較操作前後檔案差異，結果吻合預期的改變，無多餘改動或遺漏。

- 編碼與特殊字元：為測試中文編碼，我們在所有欄位使用中文或特殊符號進行操作，比如會員姓名設為日文かな或含表情符號，漫畫作者名含歐洲字母變音符，然後觀察檔案。結果顯示UTF-8編碼妥善保存了這些文字，再次載入時也未發生亂碼。CSV解析對雙引號逗號的處理亦已在單元測試證明正確。因此資料經過多次存取後仍保持內容不變，這保障了長期使用中的資料完整。

- 資料一致性：我們特別驗證了跨模組的資料一致性。例如系統不會出現某漫畫標記為已借但沒有對應借閱者的情況，因為租借/歸還操作是原子地改變兩個資訊：漫畫狀態和借閱者Id。檢查所有已借漫畫的RentedToMemberId都能在Member列表中找到對應Id。如果哪天Member被刪除而他的漫畫未還，就會出現不一致，但我們的系統禁止了刪除有借書者，所以避免了此問題。同樣也無會員租借數據卻找不到漫畫的情形。

- 例外處理機制：我們測試了在系統運行中故意引發例外，看是否被正確捕捉處理。一個例子是在UI執行緒中手動throw一個例外（我們臨時在按鈕事件中加入`throw new Exception("Test");`），按下按鈕後系統沒有崩潰，而是由`Application.ThreadException`攔截到，FileLogger寫入日誌"未處理的UI執行緒例外狀況"且彈出了MessageBox提示UI錯誤。這證明我們的全域UI例外處理有效。同理，在非UI執行緒中throw例外，被CurrentDomain_UnhandledException捕捉寫入日誌。這些機制提高了系統在未知錯誤發生時的健壯性。還有，測試發現某些我們未預期的小錯誤也被日誌捕捉，例如試圖刪除最後一位admin帳戶時，AuthenticationService.Log裡記錄了阻止刪除並返回false，這幫助我們確認功能按設計執行並沒有漏網錯誤。

綜上所述，經過多輪測試與驗證，本系統在功能性和可靠性方面均達到了預期要求。所有核心功能都經過單元測試驗證，整體使用流程在綜合測試下表現良好，資料維持一致且正確，例外處理機制確保了系統不容易因預料外情況崩潰並提供追蹤資訊。測試過程中發現的一些細節問題（例如UI上一些字串限制或使用者誤操作提醒不夠明顯等）也已及時修正。

第五章 結論與未來工作

系統順利實現了預定的各項功能，包括會員與帳戶管理、漫畫書目管理，以及漫畫的租借和歸還流程。透過前幾章的說明與測試結果，可以確認本專題的成果符合需求，並在開發過程中充

分運用了C#課程中學到的眾多技術概念。在本章中，我們將對專題成果進行總結，並討論系統目前的限制以及未來可延伸的改進方向。

成果總結

本專題開發的漫畫租借管理系統具備以下主要成果與特點：

- 功能完整，滿足需求：系統涵蓋了從使用者登入、會員與漫畫資料維護，到租借登記與歸還等租借流程的全部環節，滿足了一個小型租賃系統的基本需求。管理員可以方便地管理館藏漫畫和讀者會員，快速查詢統計資訊；會員則能自行瀏覽並借閱漫畫。整個流程所需的操作在軟體中均有對應功能實現，並經測試可用。

- 物件導向設計良好：系統核心程式碼結構清晰，模組劃分合理。我們使用了多種物件導向技巧如繼承、介面、事件等來降低耦合度並增加可復用性。例如BaseForm與ILogger的設計，讓我們在各個表單輕鬆實現日誌紀錄；ComicService和MemberService清楚地封裝各自領域邏輯且透過事件互相協調。這種設計使得系統具有較高的可維護性，也為日後功能擴充留出了空間。

- 資料處理與安全性到位：系統資料採用檔案方式永久保存，經測試在反覆讀寫後無錯誤或遺失；對於關鍵使用者帳戶資料，我們採用了雜湊加鹽儲存、登入錯誤鎖定等安全策略，有效提升了系統安全性。日誌系統詳實地記錄操作與錯誤，為除錯和稽核提供了依據。例外處理機制防止了應用程式因未捕捉錯誤而突然崩潰，提高了整體穩定性。

- 使用者介面友善：Windows Forms圖形介面直觀易用。管理員儀表板讓重要指標一目了然，各功能入口明確；表格、下拉選單、搜尋框等控制項的搭配使操作高效。使用者操作都有適當的提示和錯誤警告，減少誤用的可能。多窗口的模式允許管理員同時打開不同管理界面，提升了管理效率。整體色彩與字型經過統一設計，界面美觀一致。

- 學習與實踐收穫：對開發團隊而言，本專題是對C#課程所學的一次全面實踐。我們運用了課堂第8～15章的所有重點技術，從基礎的資料型別到進階的委派事件，均有了動手應用的體驗。在開發過程中也遇到各種挑戰，如如何正確處理檔案同步、多執行緒更新UI、確保資料一致等，這些都在不斷調整程式與查閱資料中獲得了解決，讓我們對實際軟體開發有了更深刻的理解。最終呈現的系統不僅是課堂知識的驗收，更是我們團隊合作與問題解決能力的體現。

綜上，本漫畫租借管理系統專題達成了預期目標，在功能性與可靠度上都取得了令人滿意的成果。同時我們也認識到，目前的系統仍有一些限制和可以改進之處，以下將討論未來可能的工作方向。

系統限制與未來改進方向

雖然本系統已能投入基本使用，但作為一個專題專案，仍有許多進一步完善和加強的空間。在未來的工作中，我們計畫或建議考慮以下改進方向：

1. 引入資料庫及網路功能：目前系統使用本地檔案保存資料，適用於單機環境。如果未來要擴充為多使用者同時操作（例如館員多端操作）或大量資料管理，引入關聯式資料庫如SQL Server或SQLite會更為妥當。屆時可透過實作`IFileHelper`的新版本（例如`DatabaseHelper`）將底層存取轉為資料庫，同時增加交易（Transaction）保障資料一致性。此外，若能開發網路版，讓會員透過Web或行動裝置查詢預約，將大大提升系統實用性。這部分需引入ASP.NET或手機App開發，不過核心邏輯仍可沿用本專題成果。

2. 完備使用者帳戶管理：目前帳戶管理方面尚有功能缺失，例如會員無法自行更改密碼，也沒有忘記密碼的重置機制。未來可增設「修改密碼」功能，允許登入的使用者更改自己的密碼（需再次Hash並更新users.json）。對於忘記密碼，可考慮簡單方式如管理員介入重置，或進階一點透過email驗證等。另一方向是加強權限控管，現在只有Admin和Member兩種角色，可以考慮增加如「館員」角色，其權限介於Admin與Member之間，只可管理部分功能。

3. 借閱規則與通知功能：目前系統對借閱天數和逾期沒有強制管理。未來可添加逾期管理：例如資料結構中加入「是否逾期」欄位，每次啟動或租借/歸還操作時重新計算預計歸還日在過去的借閱標記為逾期，並可能引入罰金計算等。同時可實作到期提醒功能，例如在會員登入時彈出提示「您有X本漫畫已逾期未還」，或每天系統自動產生需要聯絡的逾期名單供管理員跟進。如果擴充為網路系統，可以透過email或簡訊自動發送提醒給會員，這將提高服務品質。

4. 操作介面優化：在GUI方面，有一些使用者體驗可再提升之處。例如：為漫畫與會員列表增加排序與更多篩選功能，目前DataGridView僅提供基本顯示，我們可讓管理員點擊欄位標題進行排序（如按姓名字母排序會員）；增加多條件搜尋（如依ISBN、類型組合篩選漫畫）。介面上還可以增加匯出報表的功能，將目前館藏或借閱清單匯出為Excel/PDF供打印存檔。對於深色模式等視覺需求，也可考慮提供不同主題切換，提高適應性。

5. 系統性能與優化：由於資料量不大，目前系統性能良好。但若未來漫畫數量達數千以上，某些操作（例如每30秒重載所有資料）可能變得低效。我們可調整ReloadService機制，改為僅在需要時（如偵測到檔案更新時間改變）才重載，減少不必要IO。另方面，現在每次Add/Update都把整個CSV重寫，對大檔案來說開銷較大，可考慮改為追加寫入或部分更新。同時可以將某些操作改為非同步以避免主執行緒暫停。不過這些優化需權衡實作複雜度，在目前規模下尚非迫切需求。

6. 更多錯誤處理和使用者引導：雖然我們已處理許多例外情況，但仍有可能的漏洞需要彌補。例如，在多視窗操作時，如果同一時間兩個視窗都試圖修改同一筆資料，會出現競爭條件——這在單人操作下通常不會發生，但若未來允許多管理員並行操作就要考慮加鎖機制或者資料最新性檢查。此外，我們可以增加使用者操作日誌或確認提示，例如管理員刪除資料前彈出「確認刪除」對話框，以防誤刪。對於極少數未涵蓋的異常，我們也可在日誌基礎上增加介面提示和自動恢復措施，讓系統更加健壯。