**項目結構和命名空間設計**

1. **登入（Login）**
   * 這部分的功能通常會涉及用戶的驗證、登錄、註冊等。你可以將其劃分到 MyApp.Auth 命名空間下。
2. **統計報表（Reports）**
   * 這部分包括生成各種報表（如銷售報表、用戶行為分析等）。你可以將其放到 MyApp.Reports 命名空間中。
3. **會員設定（User Management）**
   * 這部分涉及會員資料管理、修改、刪除等操作。這可以放在 MyApp.Users 命名空間下。
4. **商品管理（Product Management）**
   * 這部分涉及商品的新增、修改、刪除、查詢等操作。可以放到 MyApp.Products 命名空間下。
5. **訂單管理（Order Management）**
   * 這部分功能通常涉及訂單的管理、處理、付款狀態等。可以放在 MyApp.Orders 命名空間下。
6. **網頁管理（Web Management）**
   * 這部分可以涉及網站內容管理、頁面顯示配置等。這可以放在 MyApp.Web 命名空間中。
7. **後端角色設定（Role Management）**
   * 這部分功能通常涉及管理後端用戶的角色、權限設定等。這可以放在 MyApp.Roles 命名空間中。

**預期的項目結構**

以下是如何安排這些命名空間和相應的類別（每個功能模組的文件可以有多個類別）：

scss

複製程式碼

MyApp/

│

├── Auth/

│ ├── LoginManager.cs // 登入邏輯(寫在前端)

│ ├── RegistrationManager.cs // 註冊邏輯

│ └── Authentication.cs // 用戶驗證

│

├── Reports/

│ ├── SalesReport.cs // 銷售報表生成

│ ├── UserBehaviorReport.cs // 用戶行為分析報表

│ └── ReportGenerator.cs // 報表生成器

│

├── Users/

│ ├── UserProfile.cs // 會員資料

│ ├── UserManager.cs // 會員管理邏輯

│ └── UserSettings.cs // 會員設定

│

├── Products/

│ ├── Product.cs // 商品實體

│ ├── ProductManager.cs // 商品邏輯

│ └── InventoryManager.cs // 庫存管理

│

├── Orders/

│ ├── Order.cs // 訂單實體

│ ├── OrderManager.cs // 訂單邏輯

│ └── OrderHistory.cs // 訂單歷史

│

├── Web/

│ ├── PageManager.cs // 網頁管理邏輯

│ └── ContentManager.cs // 網頁內容管理

│

└── Roles/

├── Role.cs // 角色實體

├── RoleManager.cs // 角色邏輯

└── PermissionManager.cs // 權限管理

**具體功能模組說明**

**1. 登入（Login）**

* **命名空間**：MyApp.Auth
* **類別**：
  + LoginManager.cs: 處理用戶登錄的邏輯（例如檢查用戶名和密碼，產生 token 等）。
  + RegistrationManager.cs: 處理用戶註冊的邏輯。
  + Authentication.cs: 處理身份驗證（如 JWT 或 Cookie 驗證等）。

csharp

複製程式碼

namespace MyApp.Auth

{

public class LoginManager

{

public bool ValidateCredentials(string username, string password)

{

// 登入邏輯

return true;

}

}

public class RegistrationManager

{

public void RegisterUser(string username, string password)

{

// 註冊邏輯

}

}

}

**2. 統計報表（Reports）**

* **命名空間**：MyApp.Reports
* **類別**：
  + SalesReport.cs: 生成銷售報表。
  + UserBehaviorReport.cs: 生成用戶行為分析報表。
  + ReportGenerator.cs: 用於生成通用報表的邏輯。

csharp

複製程式碼

namespace MyApp.Reports

{

public class SalesReport

{

public string GenerateSalesReport()

{

return "Sales Report";

}

}

public class UserBehaviorReport

{

public string GenerateUserBehaviorReport()

{

return "User Behavior Report";

}

}

}

**3. 會員設定（User Management）**

* **命名空間**：MyApp.Users
* **類別**：
  + UserProfile.cs: 會員資料類別。
  + UserManager.cs: 處理用戶資料管理的邏輯。
  + UserSettings.cs: 會員設定，例如密碼修改、個人資料更新等。

csharp

複製程式碼

namespace MyApp.Users

{

public class UserProfile

{

public string Username { get; set; }

public string Email { get; set; }

}

public class UserManager

{

public void UpdateProfile(UserProfile profile)

{

// 更新會員資料邏輯

}

}

}

**4. 商品管理（Product Management）**

* **命名空間**：MyApp.Products
* **類別**：
  + Product.cs: 商品實體。
  + ProductManager.cs: 處理商品管理的邏輯（如新增、刪除商品）。
  + InventoryManager.cs: 處理庫存管理的邏輯。

csharp

複製程式碼

namespace MyApp.Products

{

public class Product

{

public string Name { get; set; }

public decimal Price { get; set; }

}

public class ProductManager

{

public void AddProduct(Product product)

{

// 新增商品邏輯

}

}

}

**5. 訂單管理（Order Management）**

* **命名空間**：MyApp.Orders
* **類別**：
  + Order.cs: 訂單實體。
  + OrderManager.cs: 處理訂單相關的邏輯。
  + OrderHistory.cs: 顯示訂單歷史。

csharp

複製程式碼

namespace MyApp.Orders

{

public class Order

{

public int OrderId { get; set; }

public string Status { get; set; }

}

public class OrderManager

{

public void ProcessOrder(Order order)

{

// 訂單處理邏輯

}

}

}

**6. 網頁管理（Web Management）**

* **命名空間**：MyApp.Web
* **類別**：
  + PageManager.cs: 管理網站頁面。
  + ContentManager.cs: 管理網站內容。

csharp

複製程式碼

namespace MyApp.Web

{

public class PageManager

{

public void CreatePage(string pageName)

{

// 創建頁面邏輯

}

}

}

**7. 後端角色設定（Role Management）**

* **命名空間**：MyApp.Roles
* **類別**：
  + Role.cs: 角色實體。
  + RoleManager.cs: 角色管理邏輯（如新增、刪除角色）。
  + PermissionManager.cs: 管理角色權限。

csharp

複製程式碼

namespace MyApp.Roles

{

public class Role

{

public string RoleName { get; set; }

}

public class RoleManager

{

public void AddRole(Role role)

{

// 新增角色邏輯

}

}

}