

## Maintain Category

這資料表比較簡單, 可以用以下方式處理

UI	BLL	DAL
Add new Category		
<ul style="list-style-type: none"><li>● 收集表單資訊, 繫結到 CategoryVm</li><li>● 針對 CategoryVm 進行欄位驗證</li><li>● 將 CategoryVm 轉型為 CategoryDto</li><li>● 將 CategoryDto 傳送給 CategoryService</li></ul>	<p>CategoryService.Create</p> <ul style="list-style-type: none"><li>● 接收 CategoryDto</li><li>● 將 CategoryDto 轉型為 CategoryEntity</li><li>● 由於 CategoryEntity 會保持不變性, 所以若必填欄位居然沒值, 在此會錯誤</li><li>● 檢查 CategoryName 是否唯一</li><li>● 將 CategoryEntity 傳送給 CategoryRepository</li></ul>	<p>CategoryRepository.Create</p> <ul style="list-style-type: none"><li>● 接收 CategoryEntity</li><li>● 組合出 sql statement / SqlParameter[]</li><li>● 執行 sql statement , 建立一筆新記錄</li></ul>
由於上述功能簡單, 可以簡化如下		
<ul style="list-style-type: none"><li>● 收集表單資訊, 繫結到 CategoryVm</li><li>● 針對 CategoryVm 進行欄位驗證</li><li>● 將 CategoryVm 轉型為 CategoryEntity</li></ul> 將 CategoryEntity 傳送給 CategoryService	<p>CategoryService.Create</p> <ul style="list-style-type: none"><li>● 接收 CategoryEntity</li><li>● 檢查 CategoryName 是否唯一</li><li>● 將 CategoryEntity 傳送給 CategoryRepository</li></ul>	<p>CategoryRepository.Create</p> <p>與上相同</p>

Edit Category		
<ul style="list-style-type: none"> <li>● 叫用 CategoryService.Get(id)</li> <li>● 收到 CategoryDto</li> <li>● 將 CategoryDto 轉型為 CategoryVm</li> <li>● 繫結到表單</li> </ul>	CategoryService.Get(id) <ul style="list-style-type: none"> <li>● 叫用 CategoryRepository(id)</li> <li>● 收到 CategoryEntity</li> <li>● 轉型為 CategoryDto 並傳回</li> </ul>	CategoryRepository.Get(id) <ul style="list-style-type: none"> <li>● 組合出 sql statement / SqlParameter[]</li> <li>● 傳回 CategoryEntity</li> </ul>
由於上述功能簡單, 可以簡化如下		
<ul style="list-style-type: none"> <li>● 叫用 CategoryService.Get(id)</li> <li>● 收到 CategoryEntity</li> <li>● 將 CategoryEntity 繫結到表單</li> </ul>	CategoryService.Get(id) <ul style="list-style-type: none"> <li>● 叫用 CategoryRepository(id)</li> <li>● 收到 CategoryEntity 並傳回</li> </ul>	CategoryRepository.Get(id) 與上相同
Update Category		
處理方式與 Add Category 相似	CategoryService.Update 處理方式與 Add Category 相似	CategoryRepository.Create 處理方式與 Add Category 相似

Display Categories		
<ul style="list-style-type: none"> <li>● 叫用 CategoryService.Search()</li> <li>● 收到 IEnumerable&lt;CategoryDto&gt;</li> <li>● 轉型為 IEnumerable&lt;CategoryVm&gt;</li> <li>● 繫結到 DataGridView</li> </ul>	CategoryService.Search() <ul style="list-style-type: none"> <li>● 叫用 CategoryRepository.Search()</li> <li>● 收到 IEnumerable&lt;CategoryEntity&gt;</li> <li>● 轉型為 IEnumerable&lt;CategoryDto&gt;並傳回</li> </ul>	CategoryRepository.Search() <ul style="list-style-type: none"> <li>● 傳回 IEnumerable&lt;CategoryEntity&gt;</li> </ul>
由於上述功能簡單, 可以簡化如下		
<ul style="list-style-type: none"> <li>● 叫用 CategoryService.Search()</li> <li>● 收到 IEnumerable&lt;CategoryVm&gt;</li> </ul> 繫結到 DataGridView	CategoryService.Search() <ul style="list-style-type: none"> <li>● 叫用 CategoryRepository.Search()</li> <li>● 收到 IEnumerable&lt;CategoryEntity&gt;並傳回</li> </ul>	CategoryRepository.Search() 與上相同

## Maintain User

如果加入新會員與編輯會員的表單欄位不同,在註冊新會員會有 Password,Confirm Password,在編輯時,通常不會容許編輯 Account, Password(修改密碼會另外做一頁), 可以用以下方式處理

UI	BLL	DAL
Register new User		
<ul style="list-style-type: none"><li>● 收集表單資訊,繫結到 UserCreateVm</li><li>● 它包含 password,confirmPassword</li><li>● 針對 UserCreateVm 進行欄位驗證</li><li>● 將 UserCreateVm 轉型為 UserCreateDto</li><li>● 它不包含 ConfirmPassword</li><li>● 將 UserCreateDto 傳送給 UserService</li></ul>	<ul style="list-style-type: none"><li>● UserService.Create</li><li>● 接收 UserCreateDto</li><li>● 將 UserCreateDto 轉型為 UserEntity</li><li>● 轉換時,視需要將 Password 加密</li><li>● 由於 UserEntity 會保持不變性, 所以若必填欄位居然沒值,在此會錯誤</li><li>● 檢查 Account 是否唯一</li><li>● 將 UserEntity 傳送給 UserRepository</li></ul>	<ul style="list-style-type: none"><li>● UserRepository.Create</li><li>● 接收 UserEntity</li><li>● 組合出 sql statement / SqlParameter[]</li><li>● 執行 sql statement ,建立一筆新記錄</li></ul>
Edit User		
<ul style="list-style-type: none"><li>● 叫用 UserService.Get(id)</li><li>● 收到 UserEditDto</li><li>● 將 UserEditDto 轉型為 UserEditVm</li><li>● 繫結到表單</li></ul>	<ul style="list-style-type: none"><li>● UserService.Get(id)</li><li>● 叫用 UserRepository(id)</li><li>● 收到 UserEntity</li><li>● 轉型為 UserEditDto(不含 password)並傳回</li></ul>	<ul style="list-style-type: none"><li>● UserRepository.Get(id)</li><li>● 組合出 sql statement / SqlParameter[]</li><li>● 傳回 UserEntity</li></ul>
Update User		
<ul style="list-style-type: none"><li>● 收集表單資訊,繫結到 UserEditVm</li><li>● 它不包含 password,confirmPassword</li><li>● 針對 UserEditVm 進行欄位驗證</li><li>● 將 UserEditVm 轉型為 UserUpdateDto</li><li>● 將 UserUpdateDto 傳送給 UserService</li></ul>	<ul style="list-style-type: none"><li>● UserService.Update</li><li>● 接收 UserCreateDto</li><li>● 將 UserCreateDto 轉型為 UserEntity</li><li>● 由於 UserEntity 會保持不變性, 所以需要從 db 取得原本的密碼填進 UserEntity 裡</li><li>● 檢查 Account 是否唯一(如果允許修改)</li><li>● 將 UserEntity 傳送給 UserRepository</li></ul>	<ul style="list-style-type: none"><li>● UserRepository.Update</li><li>● 接收 UserEntity</li><li>● 組合出 sql statement / SqlParameter[]</li><li>● 執行 sql statement ,更新記錄</li></ul>

各個 view model, dto, entity 要放在哪一個專案裡?		
<ul style="list-style-type: none"> <li>● UserCreateVm</li> <li>● UserEditVm</li> </ul>	<ul style="list-style-type: none"> <li>● UserEntity</li> <li>● UserCreateDto</li> <li>● UserEditDto</li> </ul>	
這三個專案,要如何參考?		
<ul style="list-style-type: none"> <li>● 參考 BLL</li> <li>● 參考 SqlDataLayer</li> </ul>	不必參考另二個專案,它是軟體的核心 <ul style="list-style-type: none"> <li>● UserService</li> <li>● IUserRepository</li> <li>● UserEntity</li> <li>● UserCreateDto</li> <li>● UserEditDto</li> </ul>	<ul style="list-style-type: none"> <li>● 參考 BLL</li> <li>● UserRepository : IUserRepository</li> </ul>

呈現商品清單

它需要 join Categories, Products

UI	BLL	DAL
<ul style="list-style-type: none"><li>● 叫用 ProductsView01.Search ()</li><li>● 收到 IEnumerable&lt;ProductsDto01&gt;</li><li>● 繫結到 DataGridView</li></ul> <p>由於沒有業務邏輯, 可以考慮直接叫用 Repository class</p>	<p>CategoryService.Search()</p> <ul style="list-style-type: none"><li>● 叫用 CategoryRepository.Search()</li><li>● 收到 IEnumerable&lt;CategoryEntity&gt;</li><li>● 轉型為 IEnumerable&lt;CategoryDto&gt;並傳回</li></ul>	<ul style="list-style-type: none"><li>● ProductsDto01 class</li></ul> <p>ProductsView01.Search()</p> <ul style="list-style-type: none"><li>● 傳回 IEnumerable&lt;ProductsDto01&gt;</li></ul> <p>此功能不是寫在 ProductRepository 裡</p>
Create / Edit Product 功能與 Category 的相似, 就不重覆寫		