114年智慧行動商務系統開發班第1期

伺服端程式設計與開發學習評量

|  |  |  |  |
| --- | --- | --- | --- |
| **評量編號** | 2-5-1 | | |
| **評量主題** | 伺服端網站應用程式綜合實作 | **實施時間** | 2025/7/14 |
| **評量形式** | 作業紀錄 | **評量日期** | 2025/8/15 |
| **評量目的** | 評定學員是否具備伺服端程式設計能力，包含資料庫的設計與建置、使用ASP.net Core MVC進行程式功能的開發，並對資料庫進行CRUD之操作，且可設計適當的使用者介面及流程，提供適切的操作方式。 | | |
| **評量說明** | **任務一**：請依題目要求方式建立符合需求的模型類別、DbContext及資料庫。  **任務二**：請依題目要求設計前台的各項功能。  **任務三**：必須合於一般使用經驗及貼近實務的，包括資料存儲內容、規則及使用流程。 | | |
| **提交要求** | 繳交時請以FTP上傳**相關程式檔案**與**作業文件檔**各一份，**相關程式檔案**請另放置在「完成結果檔」資料夾內，最外層資料夾名稱請用**兩碼學號與姓名**，例如1號王小明，資料夾請命名為「01王小明」。 | | |

座號：01 姓名：劉家妤

**請建立「使用者端留言討論區」功能。**

**任務一：請依據題題上的需求，以Code-First方式建立符合下列需求的模型類別、DbContext及資料庫，必須包含：**

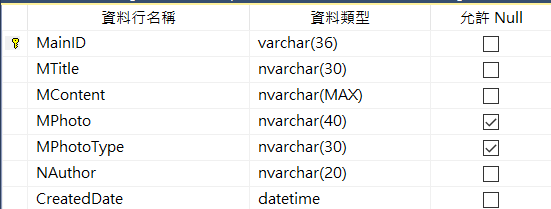
1. 主文資料表(編號(P.K)、主題、發表內容、照片、照片類型、發表人、張貼時間)
2. 回覆內容資料表(編號(P.K)、回覆內容、回覆人、回覆時間)
3. 主文資料表與回覆內容資料表之間具關聯，一則主文可以有很多則回覆內容。
4. 專案名稱、資料庫及資料表名稱可自訂。
5. 各類別中須設計相對應屬性資料型態(Data Type)及驗證規則。
6. 須製作初始化時之種子資料(Seed Data)若干筆。

**任務二：請設計前台(使用者端)功能，必須包含：**

1. 使用者可發表新文章，資料儲存於主文資料表
2. 使用者發表文章時須填寫主題、發表內容、發表人欄位，而照片可有可無。
3. 主畫面呈現各討論文章主題，並依發表時間的新到舊排列。
4. 點選主題後可進入的該文章的討論畫面，可看到主文內容、照片及回覆內容。
5. 使用者可在文章討論畫面回覆該文章。
6. 回覆文章呈現時須依發表時間的新到舊排列。

**任務三：必須合於一般使用經驗及貼近實務的，包括資料存儲內容、規則及使用流程。**

1. 資料庫(SQL Server)部份：
2. 所有資料表的設計畫面(截圖)



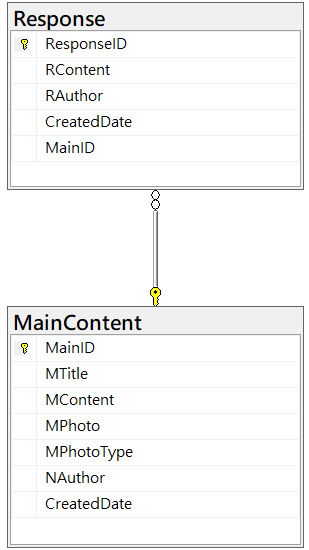


1. 所有資料表的資料檢視(截圖)





資料關聯圖(截圖)



1. 程式碼部份：
2. 所有的資料模型類別：

**Model【MainContent】**

public class MainContent

{

[Display(Name = "編號")]

[StringLength(36, MinimumLength = 36)]

[Key]

[HiddenInput]

public string MainID { get; set; } = null!; //主文編號, 採用GUID

[Display(Name = "主題")]

[StringLength(30, MinimumLength = 2, ErrorMessage = "主題2~30個字")]

[Required(ErrorMessage = "必填")]

public string MTitle { get; set; } = null!;

[Display(Name = "發表內容")]

[Required(ErrorMessage = "必填")]

[DataType(DataType.MultilineText)]

public string MContent { get; set; } = null!;

[Display(Name = "照片")]

[StringLength(40)]

public string? MPhoto { get; set; } = null!; //照片的檔名規則:GUID+.jpg

[Display(Name = "照片類型")]

[StringLength(30, MinimumLength = 2, ErrorMessage = "照片簡述2~30個字")]

public string? MPhotoType { get; set; } = null!;

[Display(Name = "發表人")]

[StringLength(20, ErrorMessage = "最多20字")]

[Required(ErrorMessage = "必填")]

public string NAuthor { get; set; } = null!;

[Display(Name = "張貼時間")]

[DataType(DataType.DateTime)]

[DisplayFormat(DataFormatString = "{0:yyyy/MM/dd hh:mm:ss}")]

[HiddenInput]

public DateTime CreatedDate { get; set; } = DateTime.Now;

public virtual List<Response>? Response { get; set; }

}

**Model【Response】**

public class Response

{

[Display(Name = "編號")]

[StringLength(36, MinimumLength = 36)]

[Key]

[HiddenInput]

public string ResponseID { get; set; } = null!; //主文編號, 採用GUID

[Display(Name = "回覆內容")]

[Required(ErrorMessage = "必填")]

[DataType(DataType.MultilineText)]

public string RContent { get; set; } = null!;

[Display(Name = "回覆人")]

[StringLength(20, ErrorMessage = "最多20字")]

[Required(ErrorMessage = "必填")]

public string RAuthor { get; set; } = null!;

[Display(Name = "回覆時間")]

[DataType(DataType.DateTime)]

[DisplayFormat(DataFormatString = "{0:yyyy/MM/dd hh:mm:ss}")]

[HiddenInput]

public DateTime CreatedDate { get; set; } = DateTime.Now;

[ForeignKey("MainContent")]

[HiddenInput]

public string MainID { get; set; } = null!;

//一個Response只能屬於一個MainContent

public virtual MainContent? MainContent { get; set; }

}

1. DbContext類別：

public class MessageBoardContext : DbContext //須繼承DbContext類別

{

//撰寫依賴注入用的建構子

public MessageBoardContext(DbContextOptions<MessageBoardContext> options)

: base(options)

{

}

public virtual DbSet<MainContent> MainContent { get; set; } //MainContent資料表

public virtual DbSet<Response> Response { get; set; } //Response資料表

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

//描述資料庫裡面的資料表

modelBuilder.Entity<MainContent>(entity =>

{

entity.HasKey(m => m.MainID).HasName("PK\_MainID"); //設定主鍵

entity.Property(m => m.MainID)

.HasMaxLength(36) //BookID長度為36

.IsUnicode(false); //不使用Unicode編碼

entity.Property(m => m.MTitle)

.HasMaxLength(30);

entity.Property(m => m.MContent)

.HasColumnType("nvarchar(max)");

entity.Property(m => m.MPhoto)

.HasMaxLength(40);

entity.Property(m => m.MPhotoType)

.HasMaxLength(30);

entity.Property(m => m.NAuthor)

.HasMaxLength(20);

entity.Property(m => m.CreatedDate)

.HasColumnType("datetime"); // 設定為datetime型別

});

modelBuilder.Entity<Response>(entity =>

{

entity.HasKey(r => r.ResponseID); //設定主鍵

entity.Property(r => r.ResponseID)

.HasMaxLength(36) //長度為36

.IsUnicode(false); //ReBookID預設值為GUID

entity.Property(r => r.RContent)

.HasColumnType("nvarchar(max)");

entity.Property(r => r.RAuthor)

.HasMaxLength(20);

entity.Property(r => r.CreatedDate)

.HasColumnType("datetime"); // 設定為datetime型別

});

}

}

1. Program.cs：

//在Program.cs內以依賴注入的寫法註冊讀取連線字串的服務

builder.Services.AddDbContext<MessageBoardContext>(options =>

options.UseSqlServer(builder.Configuration.GetConnectionString("MessageBoardConnection")))

//在Program.cs撰寫啟用Initializer的程式

using (var scope = app.Services.CreateScope())

{

var service = scope.ServiceProvider;

SeedData.Initialize(service);

};

1. appsettings.json：

"ConnectionStrings": {

"MessageBoardConnection": "Data Source=C501A105;Database=MessageBoard;TrustServerCertificate=True;User ID=abcd;Password=1234"

}

1. 所有Controller(請標示檔名)

**Controller【PostController】**

public class PostController : Controller

{

private readonly MessageBoardContext \_context;

public PostController(MessageBoardContext context)

{

\_context = context;

}

// GET: Post

public async Task<IActionResult> Index()

{

var result = await \_context.MainContent.OrderByDescending(s => s.CreatedDate).ToListAsync();

return View(result);

}

// GET: Post/Details/5

public async Task<IActionResult> Display(string id)

{

if (id == null)

{

return NotFound();

}

var mainContent = await \_context.MainContent

.FirstOrDefaultAsync(m => m.MainID == id);

if (mainContent == null)

{

return NotFound();

}

return View(mainContent);

}

// GET: Post/Create

public IActionResult Create()

{

return View();

}

// POST: Post/Create

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Create([Bind("MainID,MTitle,MContent,MPhoto,MPhotoType,NAuthor,CreatedDate")] MainContent mainContent, IFormFile? newPhoto)

{

mainContent.CreatedDate = DateTime.Now; //設定建立時間為目前時間

if (newPhoto != null && newPhoto.Length != 0)

{

//只允許上傳圖片

if (newPhoto.ContentType != "image/jpeg" && newPhoto.ContentType != "image/png")

{

ViewData["ErrMessage"] = "只允許上傳.jpg或.png的圖片檔案!!";

return View();

}

//取得檔案名稱

string fileName = mainContent.MainID + Path.GetExtension(newPhoto.FileName);

//取得檔案的完整路徑

string filePath = Path.Combine(Directory.GetCurrentDirectory(), "wwwroot", "BookPhotos", fileName);

// /wwwroot/Photos/xxx.jpg

//將檔案上傳並儲存於指定的路徑

using (FileStream fs = new FileStream(filePath, FileMode.Create))

{

newPhoto.CopyTo(fs);

}

mainContent.MPhoto = fileName;

}

if (ModelState.IsValid)

{

\_context.Add(mainContent);

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

return View(mainContent);

}

}

**Controller【ResponsesController】**

public class ResponsesController : Controller

{

private readonly MessageBoardContext \_context;

public ResponsesController(MessageBoardContext context)

{

\_context = context;

}

public IActionResult Create(string MainID)

{

//ViewData["MainID"] = new SelectList(\_context.MainContent, "MainID", "MainID");

ViewData["MainID"] = MainID;

return View();

}

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Create([Bind("ResponseID,RContent,RAuthor,CreatedDate,MainID")] Response response)

{

if (ModelState.IsValid)

{

\_context.Add(response);

await \_context.SaveChangesAsync();

//return RedirectToAction(nameof(Index));

return Json(response);

}

//ViewData["MainID"] = new SelectList(\_context.MainContent, "MainID", "MainID", response.MainID);

//return View(response);

return Json(response);

}

public IActionResult GetResponseByViewComponent(string MainID)

{

return ViewComponent("VCPost", new { MainID = MainID });

}

}

1. 執行結果部份：

請分別截取各種功能的展現畫面至少5張圖：

