1. **Create Domain Model**

namespace NZWalksAPI.Models.Domain

{

public class Image

{

public Guid Id { get; set; }

[NotMapped]

public IFormFile File { get; set; }

public string FileName { get; set; }

public string? Description { get; set; }

public string FileExtension { get; set; }

public long FileSizeInBytes { get; set; }

public string FilePath { get; set; }

}

}

1. **Add this Domain inside Dbcontext file**

public NZWalksDbContext(DbContextOptions<NZWalksDbContext> options) : base(options)

{

}

public DbSet<Walk> Walks { get; set; }

public DbSet<Region> Regions { get; set; }

public DbSet<Difficulty> Difficulties { get; set; }

public DbSet<Image> Images { get; set; }

1. **Add mirgaran and run the migration**

PM> add-migration createImageTable -context NZWalksDbContext -outputDir Migrations

PM> update-database -context NZWalksDbContext

**Create DTO**

namespace NZWalksAPI.Models.DTO

{

public class ImageUploadRequestDTO

{

[Required]

public IFormFile File { get; set; }

[Required]

public string FileName { get; set; }

public string? Description { get; set; }

}

}

[HttpPost]

[Route("Upload")]

public async Task<IActionResult> UploadImage([FromForm] ImageUploadRequestDTO imgRequest)

{

ValidateFileUpload(imgRequest);

if (!ModelState.IsValid)

{

return BadRequest(ModelState);

}

// Generate a unique file name

var uniqueFileName = $"{Guid.NewGuid()}{Path.GetExtension(imgRequest.File.FileName)}";

// Define the path to save the file wwwroot/images/filename

var filePath = Path.Combine("wwwroot", "images", uniqueFileName);

// Ensure the directory exists wwwroot/images

Directory.CreateDirectory(Path.GetDirectoryName(filePath));

// Save the file

using (var stream = new FileStream(filePath, FileMode.Create))

{

await imgRequest.File.CopyToAsync(stream);

}

// Create a response object

var response = new

{

FileName = uniqueFileName,

Description = imgRequest.Description,

FilePath = $"/images/{uniqueFileName}"

};

return Ok(response);

}

private void ValidateFileUpload(ImageUploadRequestDTO imgRequest)

{

var allowedExtensions = new string[] { ".jpg", ".jpeg", ".png" };

var fileExtension = Path.GetExtension(imgRequest.File.FileName).ToLowerInvariant();

if (!allowedExtensions.Contains(fileExtension))

{

//throw new Exception("Invalid file type. Only .jpg, .jpeg, and .png files are allowed.");

ModelState.AddModelError("File", "Invalid file type. Only .jpg, .jpeg, and .png files are allowed.");

}

if (imgRequest.File.Length > 5 \* 1024 \* 1024) // 5 MB limit

{

//throw new Exception("File size exceeds the maximum limit of 5 MB.");

ModelState.AddModelError("File", "File size exceeds the maximum limit of 5 MB.");

}

}