**동영상/사진 관리 프로그램을 위한 Database Table**

내용

[**1.** **Table종류** 2](#_Toc176168851)

[**1.1 Users** : ASP.NET에서 기본으로 제공해 주는 것 + Role, 부서, 기타1, 기타2 2](#_Toc176168852)

[**1.2 Album** 2](#_Toc176168853)

[**1.3 Media** 2](#_Toc176168854)

[**1.4 Tags** : Media에 Tag추가 3](#_Toc176168855)

[**1.5 MediaTags** : Media와 Tag간의 다대다 관계 3](#_Toc176168856)

[**데이터베이스 스키마** 4](#_Toc176168857)

[**1. 프로젝트 설정** 6](#_Toc176168858)

[**2. 데이터베이스 설정** 6](#_Toc176168859)

[**3. 모델 생성** 6](#_Toc176168860)

[**4. DbContext 설정** 8](#_Toc176168861)

[**5. 서비스 등록** 9](#_Toc176168862)

[**6. CRUD 기능 구현** 9](#_Toc176168863)

[데이터베이스 스키마 11](#_Toc176168864)

[1. 프로젝트 설정 13](#_Toc176168865)

[2. 데이터베이스 설정 13](#_Toc176168866)

[3. 모델 생성 13](#_Toc176168867)

[4. DbContext 설정 15](#_Toc176168868)

[5. 서비스 등록 16](#_Toc176168869)

[6. CRUD 기능 구현 16](#_Toc176168870)

[1. 프로젝트 생성 17](#_Toc176168871)

[2. 데이터베이스 설정 18](#_Toc176168872)

[3. 모델 생성 18](#_Toc176168873)

[4. DbContext 설정 19](#_Toc176168874)

[5. 서비스 등록 20](#_Toc176168875)

[6. CRUD 기능 구현 21](#_Toc176168876)

[1. 프로젝트 설정 23](#_Toc176168877)

[2. 데이터베이스 설정 23](#_Toc176168878)

[3. 모델 생성 23](#_Toc176168879)

[4. DbContext 설정 23](#_Toc176168880)

[5. 서비스 등록 24](#_Toc176168881)

[6. CRUD 기능 구현 25](#_Toc176168882)

[7. 뷰 생성 26](#_Toc176168883)

[1. 프로젝트 설정 29](#_Toc176168884)

[2. 데이터베이스 설정 29](#_Toc176168885)

[3. 모델 생성 29](#_Toc176168886)

[4. DbContext 설정 29](#_Toc176168887)

[5. 서비스 등록 30](#_Toc176168888)

[6. CRUD 기능 구현 31](#_Toc176168889)

1. **Table종류**

**1.1** **Users** : ASP.NET에서 기본으로 제공해 주는 것 + Role, 부서, 기타1, 기타2

🡺 IdentityDbContext+Customize 사용하므로 필요 없음.

|  |  |  |  |
| --- | --- | --- | --- |
| **Field명** |  | **NULL허용** |  |
| UserID | INT PRIMARY KEY AUTO\_INCREMENT |  |  |
| Username | varchar(5) |  |  |
| Email | varchar(100) |  |  |
| PasswordHash | varchar(255) |  |  |
| CreateAt | TIMESTAMP DEFAULT CURRENTTIMESTAMP |  |  |
| Department |  |  |  |
| Role, 기타1, 기타2 |  |  |  |

**1.2 Album**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field명** |  | **NULL허용** |  |
| AlbumID | INT PRIMARY KEY AUTO\_INCREMENT |  |  |
| UserID | INT |  |  |
| AlbumName | varchar(100) |  | 연도별 |
| Description | varchar(255) |  |  |
| CreateAt | TIMESTAMP DEFAULT CURRENTTIMESTAMP |  |  |
| FOREIGN KEY | USER Id |  |  |

**1.3 Media**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field명** |  | **NULL허용** |  |
| MediaID | INT PRIMARY KEY AUTO\_INCREMENT |  |  |
| AlbumID | INT |  |  |
| MediaType | Enum(photo, video) |  |  |
| FilePath | varchar(255) |  |  |
| FileName | varchar(100) |  |  |
| FileSize | varchar(100) |  |  |
| 일시 | TIMESTAMP DEFAULT CURRENTTIMESTAMP |  |  |
| 장소 | varchar(100) |  |  |
| 관계 | varchar(100) |  | 가족, 친족, 회사, 학교, 친목 |
| UploadAt | varchar(100) |  | 가족, 친족, 회사, 학교, 친목 |
| FOREIGN KEY | Album Id |  |  |

**1.4 Tags** : Media에 Tag추가

|  |  |  |  |
| --- | --- | --- | --- |
| **Field명** |  | **NULL허용** |  |
| TagID | INT PRIMARY KEY AUTO\_INCREMENT |  |  |
| TagName | Varchar(100) |  |  |

**1.5 MediaTags** : Media와 Tag간의 다대다 관계

|  |  |  |  |
| --- | --- | --- | --- |
| **Field명** |  | **NULL허용** |  |
| MediaId | INT |  |  |
| TagId | INT |  |  |
| PRIMARY KEY | (MediaId, TagId) |  |  |
| FOREIGN KEY | MediaId |  |  |
| FOREIGN KEY | TagID |  |  |

1. **프로그램 제작과정**

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 프로젝트 설정 | * BlazorServerAppBase에서 수정 * ImageVideoManager로 수정 |  |
| 2 | 모델 class 생성 | Album.cs 🡺 Album, Media, BaseTag, MediaTag |  |
| 3 | DbContext/DB 생성 | AlbumDbContext, PackageManager사용 DB생성 |  |
| 4 | 서비스 등록 | AlbumService.cs생성, Services.AddScoped<> |  |
| 5 | CRUD 기능 구현 |  |  |
| 6 | 뷰 생성 |  |  |

* 1. **프로젝트 설정**
  + BlazorServerAppBase 복사해와서 BlazorServerAppBase 🡺 ImageVideoManage로 모두 변경
  + Register 기능 있음
    - Id, Password, Role, CompanyName, Department, Name
    - add-migration/update-database –Context ApplicationDbContext를 해야 registering 가능
  + Side Menu
    - Home, Data입력, 데이터 검색, Dashboard, Help, Admin
  1. **모델생성**
  + Data/Album.cs 생성
  + Data/AlbumDbContext.cs 생성.
  + OnModelCreating()에서 HasKey 지정
  1. **DbContext생성**
  + Data/AlbumDbContext.cs 생성
  + add-migration, update-database해서 DB Table 생성
  1. **ServiceClass 생성 및 등록**
  + Services/AlbumService.cs 생성
  + Program.cs에서
    - builder.Services.AddDbcontext<AlbumDbContext>()
    - builder.Services.AddScoped<AlbumService>(); 추가
  1. **CRUD 기능 구현 (2024.9.4)**
  + AlbumPage.razor 추가
    - Album Listup
    - Album Add/Update
    - Tag Listup
    - Add/Delete/Update Tag
  + AlbumService.cs에 함수 추가

1. **기본 화면 구성**
   1. 입력화면

* 앨범 종류 : 선택. (추가하기)
* Media : type, file명, 일시, 장소, 관계, Tag (추가하기)
  1. 수정화면
* Media Tag 추가/삭제, 일시/장소/관계 수정
  1. 검색화면
* 앨범별, 시간별, 장소별, 관계별
* 사진보기

동영상/사진 관리 프로그램을 위한 Database Table

1. Colab에 문의 결과 (**가정에서 촬영했던 사진, 동영상 관리 프로그램을 만들고자 할때 사용할 수 있는 DB Schema를 만들어 줄 수 있나요**)

물론이죠! 가정에서 촬영한 사진과 동영상을 관리하는 프로그램을 위한 데이터베이스 스키마를 설계해 보겠습니다. 이 스키마는 사진과 동영상의 메타데이터를 저장하고, 사용자와 앨범을 관리할 수 있도록 구성할 것입니다.

**데이터베이스 스키마**

**1. Users 테이블**

사용자 정보를 저장합니다.

**SQL**

CREATE TABLE Users (

UserID INT PRIMARY KEY AUTO\_INCREMENT,

Username VARCHAR(50) NOT NULL,

Email VARCHAR(100) NOT NULL,

PasswordHash VARCHAR(255) NOT NULL,

CreatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

**2. Albums 테이블**

앨범 정보를 저장합니다.

**SQL**

CREATE TABLE Albums (

AlbumID INT PRIMARY KEY AUTO\_INCREMENT,

UserID INT,

AlbumName VARCHAR(100) NOT NULL,

Description TEXT,

CreatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (UserID) REFERENCES Users(UserID)

);

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

**3. Media 테이블**

사진과 동영상 정보를 저장합니다.

**SQL**

CREATE TABLE Media (

MediaID INT PRIMARY KEY AUTO\_INCREMENT,

AlbumID INT,

MediaType ENUM('photo', 'video') NOT NULL,

FilePath VARCHAR(255) NOT NULL,

FileName VARCHAR(100) NOT NULL,

FileSize INT,

UploadedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (AlbumID) REFERENCES Albums(AlbumID)

);

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

**4. Tags 테이블**

미디어 파일에 태그를 추가할 수 있도록 합니다.

**SQL**

CREATE TABLE Tags (

TagID INT PRIMARY KEY AUTO\_INCREMENT,

TagName VARCHAR(50) NOT NULL

);

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

**5. MediaTags 테이블**

미디어 파일과 태그 간의 다대다 관계를 관리합니다.

**SQL**

CREATE TABLE MediaTags (

MediaID INT,

TagID INT,

PRIMARY KEY (MediaID, TagID),

FOREIGN KEY (MediaID) REFERENCES Media(MediaID),

FOREIGN KEY (TagID) REFERENCES Tags(TagID)

);

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

이 스키마는 기본적인 사진 및 동영상 관리 기능을 제공하며, 필요에 따라 확장할 수 있습니다. 추가적인 기능이나 요구사항이 있다면 말씀해 주세요! 😊

1. **이 스키마를 기반으로 어떻게 프로그램을 만들 수 있을까요?**

이 스키마를 기반으로 프로그램을 만드는 과정은 여러 단계로 나눌 수 있습니다. 여기서는 웹 애플리케이션을 예로 들어 설명하겠습니다. 이 예제에서는 ASP.NET Core와 Blazor를 사용하여 프론트엔드를 구축하고, Entity Framework Core를 사용하여 데이터베이스와 상호작용하는 방법을 설명합니다.

**1. 프로젝트 설정**

먼저, 새로운 ASP.NET Core Blazor 프로젝트를 생성합니다.

dotnet new blazorserver -o PhotoVideoManager

cd PhotoVideoManager

**2. 데이터베이스 설정**

appsettings.json 파일에 데이터베이스 연결 문자열을 추가합니다.

**JSON**

"ConnectionStrings": {

"DefaultConnection": "Server=(localdb)\\mssqllocaldb;Database=PhotoVideoManagerDB;Trusted\_Connection=True;MultipleActiveResultSets=true"

}

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

**3. 모델 생성**

앞서 정의한 스키마에 따라 모델 클래스를 생성합니다.

**C#**

public class User

{

public int UserID { get; set; }

public string Username { get; set; }

public string Email { get; set; }

public string PasswordHash { get; set; }

public DateTime CreatedAt { get; set; }

public ICollection<Album> Albums { get; set; }

}

public class Album

{

public int AlbumID { get; set; }

public int UserID { get; set; }

public string AlbumName { get; set; }

public string Description { get; set; }

public DateTime CreatedAt { get; set; }

public User User { get; set; }

public ICollection<Media> MediaFiles { get; set; }

}

public class Media

{

public int MediaID { get; set; }

public int AlbumID { get; set; }

public string MediaType { get; set; }

public string FilePath { get; set; }

public string FileName { get; set; }

public int FileSize { get; set; }

public DateTime UploadedAt { get; set; }

public Album Album { get; set; }

public ICollection<MediaTag> MediaTags { get; set; }

}

public class Tag

{

public int TagID { get; set; }

public string TagName { get; set; }

public ICollection<MediaTag> MediaTags { get; set; }

}

public class MediaTag

{

public int MediaID { get; set; }

public int TagID { get; set; }

public Media Media { get; set; }

public Tag Tag { get; set; }

}

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

**4. DbContext 설정**

Entity Framework Core를 사용하여 DbContext를 설정합니다.

**C#**

public class ApplicationDbContext : DbContext

{

public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)

: base(options)

{

}

public DbSet<User> Users { get; set; }

public DbSet<Album> Albums { get; set; }

public DbSet<Media> MediaFiles { get; set; }

public DbSet<Tag> Tags { get; set; }

public DbSet<MediaTag> MediaTags { get; set; }

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

modelBuilder.Entity<MediaTag>()

.HasKey(mt => new { mt.MediaID, mt.TagID });

modelBuilder.Entity<MediaTag>()

.HasOne(mt => mt.Media)

.WithMany(m => m.MediaTags)

.HasForeignKey(mt => mt.MediaID);

modelBuilder.Entity<MediaTag>()

.HasOne(mt => mt.Tag)

.WithMany(t => t.MediaTags)

.HasForeignKey(mt => mt.TagID);

}

}

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

**5. 서비스 등록**

Startup.cs 파일에서 DbContext를 서비스에 등록합니다.

**C#**

public void ConfigureServices(IServiceCollection services)

{

services.AddDbContext<ApplicationDbContext>(options =>

options.UseSqlServer(Configuration.GetConnectionString("DefaultConnection")));

services.AddRazorPages();

services.AddServerSideBlazor();

}

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

**6. CRUD 기능 구현**

Blazor 컴포넌트를 사용하여 CRUD(Create, Read, Update, Delete) 기능을 구현합니다. 예를 들어, 앨범을 생성하는 페이지를 만들 수 있습니다.

**CreateAlbum.razor**

@page "/create-album"

@inject ApplicationDbContext DbContext

<h3>Create Album</h3>

<EditForm Model="@album" OnValidSubmit="HandleValidSubmit">

<DataAnnotationsValidator />

<ValidationSummary />

<div>

<label>Album Name:</label>

<InputText @bind-Value="album.AlbumName" />

</div>

<div>

<label>Description:</label>

<InputTextArea @bind-Value="album.Description" />

</div>

<button type="submit">Create</button>

</EditForm>

@code {

private Album album = new Album();

private async Task HandleValidSubmit()

{

DbContext.Albums.Add(album);

await DbContext.SaveChangesAsync();

// Redirect or show success message

}

}

이와 같은 방식으로 다른 CRUD 기능도 구현할 수 있습니다. 이 예제는 기본적인 구조를 제공하며, 필요에 따라 기능을 확장하고 사용자 인터페이스를 개선할 수 있습니다. 추가적인 질문이나 도움이 필요하시면 언제든지 말씀해 주세요! 😊

가정에서 촬영했던 사진, 동영상 관리 프로그램을 만들고자 할때 사용할 수 있는 DB Schema를 만들어 줄 수 있나요

Copilot

답변 생성 중…

###### 보낸 사람 Copilot:

물론이죠! 가정에서 촬영한 사진과 동영상을 관리하는 프로그램을 위한 데이터베이스 스키마를 설계해 보겠습니다. 이 스키마는 사진과 동영상의 메타데이터를 저장하고, 사용자와 앨범을 관리할 수 있도록 구성할 것입니다.

### 데이터베이스 스키마

#### 1. Users 테이블

사용자 정보를 저장합니다.

**SQL**

CREATE TABLE Users (

UserID INT PRIMARY KEY AUTO\_INCREMENT,

Username VARCHAR(50) NOT NULL,

Email VARCHAR(100) NOT NULL,

PasswordHash VARCHAR(255) NOT NULL,

CreatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

#### 2. Albums 테이블

앨범 정보를 저장합니다.

**SQL**

CREATE TABLE Albums (

AlbumID INT PRIMARY KEY AUTO\_INCREMENT,

UserID INT,

AlbumName VARCHAR(100) NOT NULL,

Description TEXT,

CreatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (UserID) REFERENCES Users(UserID)

);

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

#### 3. Media 테이블

사진과 동영상 정보를 저장합니다.

**SQL**

CREATE TABLE Media (

MediaID INT PRIMARY KEY AUTO\_INCREMENT,

AlbumID INT,

MediaType ENUM('photo', 'video') NOT NULL,

FilePath VARCHAR(255) NOT NULL,

FileName VARCHAR(100) NOT NULL,

FileSize INT,

UploadedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (AlbumID) REFERENCES Albums(AlbumID)

);

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

#### 4. Tags 테이블

미디어 파일에 태그를 추가할 수 있도록 합니다.

**SQL**

CREATE TABLE Tags (

TagID INT PRIMARY KEY AUTO\_INCREMENT,

TagName VARCHAR(50) NOT NULL

);

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

#### 5. MediaTags 테이블

미디어 파일과 태그 간의 다대다 관계를 관리합니다.

**SQL**

CREATE TABLE MediaTags (

MediaID INT,

TagID INT,

PRIMARY KEY (MediaID, TagID),

FOREIGN KEY (MediaID) REFERENCES Media(MediaID),

FOREIGN KEY (TagID) REFERENCES Tags(TagID)

);

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

이 스키마는 기본적인 사진 및 동영상 관리 기능을 제공하며, 필요에 따라 확장할 수 있습니다. 추가적인 기능이나 요구사항이 있다면 말씀해 주세요! 😊

자세한 정보

[1blog.naver.com](https://m.blog.naver.com/artismart/221609062002)[2kr.cyberlink.com](https://kr.cyberlink.com/blog/the-top-video-editors/1967/best-slideshow-maker-apps)[3canva.com](https://www.canva.com/ko_kr/features/photo-to-video/)

당신

##### 보낸 사람 당신:

이 스키마를 기반으로 어떻게 프로그램을 만들 수 있을까요?

Copilot

답변 생성 중…

###### 보낸 사람 Copilot:

이 스키마를 기반으로 프로그램을 만드는 과정은 여러 단계로 나눌 수 있습니다. 여기서는 웹 애플리케이션을 예로 들어 설명하겠습니다. 이 예제에서는 ASP.NET Core와 Blazor를 사용하여 프론트엔드를 구축하고, Entity Framework Core를 사용하여 데이터베이스와 상호작용하는 방법을 설명합니다.

### 1. 프로젝트 설정

먼저, 새로운 ASP.NET Core Blazor 프로젝트를 생성합니다.

dotnet new blazorserver -o PhotoVideoManager

cd PhotoVideoManager

### 2. 데이터베이스 설정

appsettings.json 파일에 데이터베이스 연결 문자열을 추가합니다.

**JSON**

"ConnectionStrings": {

"DefaultConnection": "Server=(localdb)\\mssqllocaldb;Database=PhotoVideoManagerDB;Trusted\_Connection=True;MultipleActiveResultSets=true"

}

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

### 3. 모델 생성

앞서 정의한 스키마에 따라 모델 클래스를 생성합니다.

**C#**

public class User

{

public int UserID { get; set; }

public string Username { get; set; }

public string Email { get; set; }

public string PasswordHash { get; set; }

public DateTime CreatedAt { get; set; }

public ICollection<Album> Albums { get; set; }

}

public class Album

{

public int AlbumID { get; set; }

public int UserID { get; set; }

public string AlbumName { get; set; }

public string Description { get; set; }

public DateTime CreatedAt { get; set; }

public User User { get; set; }

public ICollection<Media> MediaFiles { get; set; }

}

public class Media

{

public int MediaID { get; set; }

public int AlbumID { get; set; }

public string MediaType { get; set; }

public string FilePath { get; set; }

public string FileName { get; set; }

public int FileSize { get; set; }

public DateTime UploadedAt { get; set; }

public Album Album { get; set; }

public ICollection<MediaTag> MediaTags { get; set; }

}

public class Tag

{

public int TagID { get; set; }

public string TagName { get; set; }

public ICollection<MediaTag> MediaTags { get; set; }

}

public class MediaTag

{

public int MediaID { get; set; }

public int TagID { get; set; }

public Media Media { get; set; }

public Tag Tag { get; set; }

}

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

### 4. DbContext 설정

Entity Framework Core를 사용하여 DbContext를 설정합니다.

**C#**

public class ApplicationDbContext : DbContext

{

public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)

: base(options)

{

}

public DbSet<User> Users { get; set; }

public DbSet<Album> Albums { get; set; }

public DbSet<Media> MediaFiles { get; set; }

public DbSet<Tag> Tags { get; set; }

public DbSet<MediaTag> MediaTags { get; set; }

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

modelBuilder.Entity<MediaTag>()

.HasKey(mt => new { mt.MediaID, mt.TagID });

modelBuilder.Entity<MediaTag>()

.HasOne(mt => mt.Media)

.WithMany(m => m.MediaTags)

.HasForeignKey(mt => mt.MediaID);

modelBuilder.Entity<MediaTag>()

.HasOne(mt => mt.Tag)

.WithMany(t => t.MediaTags)

.HasForeignKey(mt => mt.TagID);

}

}

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

### 5. 서비스 등록

Startup.cs 파일에서 DbContext를 서비스에 등록합니다.

**C#**

public void ConfigureServices(IServiceCollection services)

{

services.AddDbContext<ApplicationDbContext>(options =>

options.UseSqlServer(Configuration.GetConnectionString("DefaultConnection")));

services.AddRazorPages();

services.AddServerSideBlazor();

}

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

### 6. CRUD 기능 구현

Blazor 컴포넌트를 사용하여 CRUD(Create, Read, Update, Delete) 기능을 구현합니다. 예를 들어, 앨범을 생성하는 페이지를 만들 수 있습니다.

#### CreateAlbum.razor

@page "/create-album"

@inject ApplicationDbContext DbContext

<h3>Create Album</h3>

<EditForm Model="@album" OnValidSubmit="HandleValidSubmit">

<DataAnnotationsValidator />

<ValidationSummary />

<div>

<label>Album Name:</label>

<InputText @bind-Value="album.AlbumName" />

</div>

<div>

<label>Description:</label>

<InputTextArea @bind-Value="album.Description" />

</div>

<button type="submit">Create</button>

</EditForm>

@code {

private Album album = new Album();

private async Task HandleValidSubmit()

{

DbContext.Albums.Add(album);

await DbContext.SaveChangesAsync();

// Redirect or show success message

}

}

이와 같은 방식으로 다른 CRUD 기능도 구현할 수 있습니다. 이 예제는 기본적인 구조를 제공하며, 필요에 따라 기능을 확장하고 사용자 인터페이스를 개선할 수 있습니다. 추가적인 질문이나 도움이 필요하시면 언제든지 말씀해 주세요! 😊

당신

##### 보낸 사람 당신:

이 Database를 이용한 Blazor Server Application full source를 제공해 줄 수 있나요 ?

Copilot

답변 생성 중…

###### 보낸 사람 Copilot:

제가 직접 전체 소스 코드를 제공해 드릴 수는 없지만, 유사한 프로젝트를 시작하는 데 도움이 될 수 있는 몇 가지 리소스를 안내해 드릴 수 있습니다. 다음은 Blazor Server 애플리케이션을 만들고 데이터베이스와 연동하는 예제입니다.

### 1. 프로젝트 생성

먼저, Blazor Server 프로젝트를 생성합니다.

dotnet new blazorserver -o PhotoVideoManager

cd PhotoVideoManager

### 2. 데이터베이스 설정

appsettings.json 파일에 데이터베이스 연결 문자열을 추가합니다.

**JSON**

"ConnectionStrings": {

"DefaultConnection": "Server=(localdb)\\mssqllocaldb;Database=PhotoVideoManagerDB;Trusted\_Connection=True;MultipleActiveResultSets=true"

}

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

### 3. 모델 생성

앞서 정의한 스키마에 따라 모델 클래스를 생성합니다.

**C#**

public class User

{

public int UserID { get; set; }

public string Username { get; set; }

public string Email { get; set; }

public string PasswordHash { get; set; }

public DateTime CreatedAt { get; set; }

public ICollection<Album> Albums { get; set; }

}

public class Album

{

public int AlbumID { get; set; }

public int UserID { get; set; }

public string AlbumName { get; set; }

public string Description { get; set; }

public DateTime CreatedAt { get; set; }

public User User { get; set; }

public ICollection<Media> MediaFiles { get; set; }

}

public class Media

{

public int MediaID { get; set; }

public int AlbumID { get; set; }

public string MediaType { get; set; }

public string FilePath { get; set; }

public string FileName { get; set; }

public int FileSize { get; set; }

public DateTime UploadedAt { get; set; }

public Album Album { get; set; }

public ICollection<MediaTag> MediaTags { get; set; }

}

public class Tag

{

public int TagID { get; set; }

public string TagName { get; set; }

public ICollection<MediaTag> MediaTags { get; set; }

}

public class MediaTag

{

public int MediaID { get; set; }

public int TagID { get; set; }

public Media Media { get; set; }

public Tag Tag { get; set; }

}

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

### 4. DbContext 설정

Entity Framework Core를 사용하여 DbContext를 설정합니다.

**C#**

public class ApplicationDbContext : DbContext

{

public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)

: base(options)

{

}

public DbSet<User> Users { get; set; }

public DbSet<Album> Albums { get; set; }

public DbSet<Media> MediaFiles { get; set; }

public DbSet<Tag> Tags { get; set; }

public DbSet<MediaTag> MediaTags { get; set; }

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

modelBuilder.Entity<MediaTag>()

.HasKey(mt => new { mt.MediaID, mt.TagID });

modelBuilder.Entity<MediaTag>()

.HasOne(mt => mt.Media)

.WithMany(m => m.MediaTags)

.HasForeignKey(mt => mt.MediaID);

modelBuilder.Entity<MediaTag>()

.HasOne(mt => mt.Tag)

.WithMany(t => t.MediaTags)

.HasForeignKey(mt => mt.TagID);

}

}

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

### 5. 서비스 등록

Program.cs 파일에서 DbContext를 서비스에 등록합니다.

**C#**

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.

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

options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")));

builder.Services.AddRazorPages();

builder.Services.AddServerSideBlazor();

var app = builder.Build();

// Configure the HTTP request pipeline.

if (!app.Environment.IsDevelopment())

{

app.UseExceptionHandler("/Error");

app.UseHsts();

}

app.UseHttpsRedirection();

app.UseStaticFiles();

app.UseRouting();

app.MapBlazorHub();

app.MapFallbackToPage("/\_Host");

app.Run();

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

### 6. CRUD 기능 구현

Blazor 컴포넌트를 사용하여 CRUD(Create, Read, Update, Delete) 기능을 구현합니다. 예를 들어, 앨범을 생성하는 페이지를 만들 수 있습니다.

#### CreateAlbum.razor

@page "/create-album"

@inject ApplicationDbContext DbContext

<h3>Create Album</h3>

<EditForm Model="@album" OnValidSubmit="HandleValidSubmit">

<DataAnnotationsValidator />

<ValidationSummary />

<div>

<label>Album Name:</label>

<InputText @bind-Value="album.AlbumName" />

</div>

<div>

<label>Description:</label>

<InputTextArea @bind-Value="album.Description" />

</div>

<button type="submit">Create</button>

</EditForm>

@code {

private Album album = new Album();

private async Task HandleValidSubmit()

{

DbContext.Albums.Add(album);

await DbContext.SaveChangesAsync();

// Redirect or show success message

}

}

이와 같은 방식으로 다른 CRUD 기능도 구현할 수 있습니다. 전체 소스 코드를 포함한 예제 프로젝트는 [GitHub에서 확인할 수 있습니다1](https://github.com/dotnet/blazor-samples). 추가적인 질문이나 도움이 필요하시면 언제든지 말씀해 주세요! 😊

[1](https://github.com/dotnet/blazor-samples): [GitHub - dotnet/blazor-samples](https://github.com/dotnet/blazor-samples)

자세한 정보

[1github.com](https://github.com/dotnet/blazor-samples)[2blazorhelpwebsite.com](https://blazorhelpwebsite.com/ViewBlogPost/34)[3codeproject.com](https://www.codeproject.com/Articles/5344267/ASP-NET-Core-6-0-Blazor-Server-APP-and-working-wit)[4github.com](https://github.com/jasontaylordev/azd-blazor)[5prowaretech.com](https://www.prowaretech.com/articles/current/blazor/wasm/create-ms-sql-db-with-entity-framework)[6microsoft.com](https://www.microsoft.com/en-us/sql-server/sql-server-downloads)[7github.com](https://github.com/ShadenAlZaid/Dealership/tree/9b14328fd5a1c41f06e7e9d7842ac258ecdff09a/Dealership.Blazor%2FProgram.cs)[8github.com](https://github.com/jubalaaronmurrian/AspNetCore.Docs/tree/c0691d61bef1e500d0bfb9c90382076e75707c7c/aspnetcore%2Ffundamentals%2Fstartup%2F6.0_samples%2FWebAll%2FProgram.cs)[9github.com](https://github.com/KrishanSamarawickrama/BlazorAndAPIOnDapper/tree/4048ed1fa635e9b59b920d205b9d4b9746277762/BlazorAndAPIOnDapper.Web.SSR%2FProgram.cs)[10github.com](https://github.com/Soju06/CoronaPhaseMap/tree/2bd1baa783e907d820092f1a0cb60817b16e7bfa/CoronaPhaseMap%2FProgram.cs)+6 더 보기

당신

##### 보낸 사람 당신:

MSSQL Database에 Record를 추가하는 것과 기존 Record를 수정하는 DB CRUD example code를 ASP.NET 기반으로 제시해 주세요

Copilot

답변 생성 중…

###### 보낸 사람 Copilot:

물론입니다! ASP.NET Core와 Entity Framework Core를 사용하여 MSSQL 데이터베이스에 레코드를 추가하고 수정하는 예제를 제공하겠습니다. 이 예제에서는 간단한 “Product” 엔터티를 사용하여 CRUD(Create, Read, Update, Delete) 작업을 수행합니다.

### 1. 프로젝트 설정

먼저, 새로운 ASP.NET Core MVC 프로젝트를 생성합니다.

dotnet new mvc -o ProductApp

cd ProductApp

### 2. 데이터베이스 설정

appsettings.json 파일에 데이터베이스 연결 문자열을 추가합니다.

**JSON**

"ConnectionStrings": {

"DefaultConnection": "Server=(localdb)\\mssqllocaldb;Database=ProductAppDB;Trusted\_Connection=True;MultipleActiveResultSets=true"

}

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

### 3. 모델 생성

Models 폴더에 Product 클래스를 생성합니다.

**C#**

public class Product

{

public int ProductID { get; set; }

public string Name { get; set; }

public decimal Price { get; set; }

public int Stock { get; set; }

}

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

### 4. DbContext 설정

Data 폴더에 ApplicationDbContext 클래스를 생성합니다.

**C#**

public class ApplicationDbContext : DbContext

{

public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)

: base(options)

{

}

public DbSet<Product> Products { get; set; }

}

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

### 5. 서비스 등록

Program.cs 파일에서 DbContext를 서비스에 등록합니다.

**C#**

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.

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

options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")));

builder.Services.AddControllersWithViews();

var app = builder.Build();

// Configure the HTTP request pipeline.

if (!app.Environment.IsDevelopment())

{

app.UseExceptionHandler("/Home/Error");

app.UseHsts();

}

app.UseHttpsRedirection();

app.UseStaticFiles();

app.UseRouting();

app.UseAuthorization();

app.MapControllerRoute(

name: "default",

pattern: "{controller=Home}/{action=Index}/{id?}");

app.Run();

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

### 6. CRUD 기능 구현

Controllers 폴더에 ProductsController를 생성하고 CRUD 작업을 구현합니다.

#### Create (추가)

**C#**

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Create([Bind("ProductID,Name,Price,Stock")] Product product)

{

if (ModelState.IsValid)

{

\_context.Add(product);

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

return View(product);

}

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

#### Update (수정)

**C#**

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Edit(int id, [Bind("ProductID,Name,Price,Stock")] Product product)

{

if (id != product.ProductID)

{

return NotFound();

}

if (ModelState.IsValid)

{

try

{

\_context.Update(product);

await \_context.SaveChangesAsync();

}

catch (DbUpdateConcurrencyException)

{

if (!ProductExists(product.ProductID))

{

return NotFound();

}

else

{

throw;

}

}

return RedirectToAction(nameof(Index));

}

return View(product);

}

private bool ProductExists(int id)

{

return \_context.Products.Any(e => e.ProductID == id);

}

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

### 7. 뷰 생성

Views/Products 폴더에 Create.cshtml과 Edit.cshtml 뷰를 생성합니다.

#### Create.cshtml

**HTML**

@model ProductApp.Models.Product

<h2>Create</h2>

<form asp-action="Create">

<div class="form-group">

<label asp-for="Name" class="control-label"></label>

<input asp-for="Name" class="form-control" />

<span asp-validation-for="Name" class="text-danger"></span>

</div>

<div class="form-group">

<label asp-for="Price" class="control-label"></label>

<input asp-for="Price" class="form-control" />

<span asp-validation-for="Price" class="text-danger"></span>

</div>

<div class="form-group">

<label asp-for="Stock" class="control-label"></label>

<input asp-for="Stock" class="form-control" />

<span asp-validation-for="Stock" class="text-danger"></span>

</div>

<div class="form-group">

<input type="submit" value="Create" class="btn btn-primary" />

</div>

</form>

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

#### Edit.cshtml

**HTML**

@model ProductApp.Models.Product

<h2>Edit</h2>

<form asp-action="Edit">

<div class="form-group">

<label asp-for="Name" class="control-label"></label>

<input asp-for="Name" class="form-control" />

<span asp-validation-for="Name" class="text-danger"></span>

</div>

<div class="form-group">

<label asp-for="Price" class="control-label"></label>

<input asp-for="Price" class="form-control" />

<span asp-validation-for="Price" class="text-danger"></span>

</div>

<div class="form-group">

<label asp-for="Stock" class="control-label"></label>

<input asp-for="Stock" class="form-control" />

<span asp-validation-for="Stock" class="text-danger"></span>

</div>

<div class="form-group">

<input type="submit" value="Save" class="btn btn-primary" />

</div>

</form>

**(ASP.NET MVC with EF Core : CRUD기능 구현하기 Overview)**

<https://learn.microsoft.com/en-us/aspnet/core/data/ef-mvc/crud?view=aspnetcore-8.0>

MSSQL Database에 Record를 추가하는 것과 기존 Record를 수정하는 DB CRUD example code를 Blazor Server Application 기반으로도 제시해 주세요

Copilot

답변 생성 중…

###### 보낸 사람 Copilot:

물론입니다! Blazor Server 애플리케이션을 사용하여 MSSQL 데이터베이스에 레코드를 추가하고 수정하는 예제를 제공하겠습니다. 이 예제에서는 간단한 “Product” 엔터티를 사용하여 CRUD(Create, Read, Update, Delete) 작업을 수행합니다.

### 1. 프로젝트 설정

먼저, 새로운 Blazor Server 프로젝트를 생성합니다.

dotnet new blazorserver -o ProductApp

cd ProductApp

### 2. 데이터베이스 설정

appsettings.json 파일에 데이터베이스 연결 문자열을 추가합니다.

**JSON**

"ConnectionStrings": {

"DefaultConnection": "Server=(localdb)\\mssqllocaldb;Database=ProductAppDB;Trusted\_Connection=True;MultipleActiveResultSets=true"

}

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

### 3. 모델 생성

Models 폴더에 Product 클래스를 생성합니다.

**C#**

public class Product

{

public int ProductID { get; set; }

public string Name { get; set; }

public decimal Price { get; set; }

public int Stock { get; set; }

}

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

### 4. DbContext 설정

Data 폴더에 ApplicationDbContext 클래스를 생성합니다.

**C#**

public class ApplicationDbContext : DbContext

{

public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)

: base(options)

{

}

public DbSet<Product> Products { get; set; }

}

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

### 5. 서비스 등록

Program.cs 파일에서 DbContext를 서비스에 등록합니다.

**C#**

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.

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

options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")));

builder.Services.AddRazorPages();

builder.Services.AddServerSideBlazor();

var app = builder.Build();

// Configure the HTTP request pipeline.

if (!app.Environment.IsDevelopment())

{

app.UseExceptionHandler("/Error");

app.UseHsts();

}

app.UseHttpsRedirection();

app.UseStaticFiles();

app.UseRouting();

app.UseAuthorization();

app.MapBlazorHub();

app.MapFallbackToPage("/\_Host");

app.Run();

AI가 생성한 코드입니다. 신중하게 검토하고 사용하세요. [FAQ의 자세한 정보](https://www.bing.com/new#faq).

### 6. CRUD 기능 구현

Pages 폴더에 Product.razor 페이지를 생성하고 CRUD 작업을 구현합니다.

#### Create (추가)

@page "/create-product"

@inject ApplicationDbContext DbContext

<h3>Create Product</h3>

<EditForm Model="@product" OnValidSubmit="HandleValidSubmit">

<DataAnnotationsValidator />

<ValidationSummary />

<div>

<label>Name:</label>

<InputText @bind-Value="product.Name" />

</div>

<div>

<label>Price:</label>

<InputNumber @bind-Value="product.Price" />

</div>

<div>

<label>Stock:</label>

<InputNumber @bind-Value="product.Stock" />

</div>

<button type="submit">Create</button>

</EditForm>

@code {

private Product product = new Product();

private async Task HandleValidSubmit()

{

DbContext.Products.Add(product);

await DbContext.SaveChangesAsync();

// Redirect or show success message

}

}

#### Update (수정)

@page "/edit-product/{id:int}"

@inject ApplicationDbContext DbContext

<h3>Edit Product</h3>

<EditForm Model="@product" OnValidSubmit="HandleValidSubmit">

<DataAnnotationsValidator />

<ValidationSummary />

<div>

<label>Name:</label>

<InputText @bind-Value="product.Name" />

</div>

<div>

<label>Price:</label>

<InputNumber @bind-Value="product.Price" />

</div>

<div>

<label>Stock:</label>

<InputNumber @bind-Value="product.Stock" />

</div>

<button type="submit">Save</button>

</EditForm>

@code {

[Parameter]

public int Id { get; set; }

private Product product;

protected override async Task OnInitializedAsync()

{

product = await DbContext.Products.FindAsync(Id);

}

private async Task HandleValidSubmit()

{

DbContext.Products.Update(product);

await DbContext.SaveChangesAsync();

// Redirect or show success message

}

}

이 예제는 기본적인 CRUD 기능을 구현한 것입니다. [더 자세한 내용은](https://learn.microsoft.com/en-us/aspnet/core/blazor/blazor-ef-core?view=aspnetcore-8.0)[Microsoft Learn1](https://learn.microsoft.com/en-us/aspnet/core/blazor/blazor-ef-core?view=aspnetcore-8.0)과 같은 리소스를 참고하시면 도움이 될 것입니다. 추가적인 질문이 있으면 언제든지 말씀해 주세요! 😊