SQL SERVER Query :

CREATE DATABASE BookstoreDB;

GO

USE BookstoreDB;

GO

--Books Table

CREATE TABLE Books (

BookId INT IDENTITY(1,1) PRIMARY KEY,

Title NVARCHAR(100) NOT NULL,

Author NVARCHAR(100) NOT NULL,

Price DECIMAL(10,2) NOT NULL

);

GO

-- Stored Procedures

CREATE PROCEDURE spAddBook

@Title NVARCHAR(100),

@Author NVARCHAR(100),

@Price DECIMAL(10,2)

AS

BEGIN

INSERT INTO Books (Title, Author, Price)

VALUES (@Title, @Author, @Price);

END

GO

CREATE PROCEDURE spUpdateBook

@BookId INT,

@Title NVARCHAR(100),

@Author NVARCHAR(100),

@Price DECIMAL(10,2)

AS

BEGIN

UPDATE Books

SET Title = @Title, Author = @Author, Price = @Price

WHERE BookId = @BookId;

END

GO

CREATE PROCEDURE spDeleteBook

@BookId INT

AS

BEGIN

DELETE FROM Books WHERE BookId = @BookId;

END

GO

MODELS:

1.)Book.cs

namespace BookstoreApp.Models

{

public class Book

{

public int BookId { get; set; }

public string Title { get; set; } = string.Empty;

public string Author { get; set; } = string.Empty;

public decimal Price { get; set; }

}

}

Data:

1.)BookRepository.cs

using System.Data;

using System.Data.SqlClient;

using BookstoreApp.Models;

namespace BookstoreApp.Data

{

public class BookRepository

{

private readonly string \_connectionString;

public BookRepository(IConfiguration config)

{

\_connectionString = config.GetConnectionString("BookstoreDB");

}

// Get All Books (Disconnected using DataSet)

public List<Book> GetAllBooks()

{

List<Book> books = new();

using (SqlConnection conn = new SqlConnection(\_connectionString))

{

string query = "SELECT \* FROM Books";

SqlDataAdapter da = new SqlDataAdapter(query, conn);

DataSet ds = new DataSet();

da.Fill(ds);

foreach (DataRow row in ds.Tables[0].Rows)

{

books.Add(new Book

{

BookId = Convert.ToInt32(row["BookId"]),

Title = row["Title"].ToString(),

Author = row["Author"].ToString(),

Price = Convert.ToDecimal(row["Price"])

});

}

}

return books;

}

// Get Book by Id (Connected using DataReader)

public Book? GetBookById(int id)

{

Book? book = null;

using (SqlConnection conn = new SqlConnection(\_connectionString))

{

string query = "SELECT \* FROM Books WHERE BookId = @BookId";

SqlCommand cmd = new SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@BookId", id);

conn.Open();

SqlDataReader reader = cmd.ExecuteReader();

if (reader.Read())

{

book = new Book

{

BookId = (int)reader["BookId"],

Title = reader["Title"].ToString(),

Author = reader["Author"].ToString(),

Price = (decimal)reader["Price"]

};

}

}

return book;

}

// Add Book (Stored Procedure)

public void AddBook(Book book)

{

using (SqlConnection conn = new SqlConnection(\_connectionString))

{

SqlCommand cmd = new SqlCommand("spAddBook", conn);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@Title", book.Title);

cmd.Parameters.AddWithValue("@Author", book.Author);

cmd.Parameters.AddWithValue("@Price", book.Price);

conn.Open();

cmd.ExecuteNonQuery();

}

}

// Update Book (Stored Procedure)

public void UpdateBook(Book book)

{

using (SqlConnection conn = new SqlConnection(\_connectionString))

{

SqlCommand cmd = new SqlCommand("spUpdateBook", conn);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@BookId", book.BookId);

cmd.Parameters.AddWithValue("@Title", book.Title);

cmd.Parameters.AddWithValue("@Author", book.Author);

cmd.Parameters.AddWithValue("@Price", book.Price);

conn.Open();

cmd.ExecuteNonQuery();

}

}

//Delete Book

public void DeleteBook(int id)

{

using (SqlConnection conn = new SqlConnection(\_connectionString))

{

SqlCommand cmd = new SqlCommand("spDeleteBook", conn);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@BookId", id);

conn.Open();

cmd.ExecuteNonQuery();

}

}

}

}

Controller:

1.)BookController.cs

using Microsoft.AspNetCore.Mvc;

using BookstoreApp.Models;

using BookstoreApp.Data;

namespace BookstoreApp.Controllers

{

[ApiController]

[Route("api/[controller]")]

public class BooksController : ControllerBase

{

private readonly BookRepository \_repository;

public BooksController(IConfiguration config)

{

\_repository = new BookRepository(config);

}

[HttpGet]

public IActionResult GetAll() => Ok(\_repository.GetAllBooks());

[HttpGet("{id}")]

public IActionResult Get(int id)

{

var book = \_repository.GetBookById(id);

if (book == null) return NotFound();

return Ok(book);

}

[HttpPost]

public IActionResult Add(Book book)

{

\_repository.AddBook(book);

return Ok("Book added successfully!");

}

[HttpPut("{id}")]

public IActionResult Update(int id, Book book)

{

book.BookId = id;

\_repository.UpdateBook(book);

return Ok("Book updated successfully!");

}

[HttpDelete("{id}")]

public IActionResult Delete(int id)

{

\_repository.DeleteBook(id);

return Ok("Book deleted successfully!");

}

}

}

Program.cs

using Microsoft.AspNetCore.Identity;

using Microsoft.EntityFrameworkCore;

using BookstoreApp.Data;

var builder = WebApplication.CreateBuilder(args);

// Add services to the container

builder.Services.AddControllers();

// Add Swagger services

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen();

var app = builder.Build();

// Configure the HTTP request pipeline

if (app.Environment.IsDevelopment())

{

app.UseSwagger();

app.UseSwaggerUI(c =>

{

c.SwaggerEndpoint("/swagger/v1/swagger.json", "Bookstore API V1");

c.RoutePrefix = string.Empty; // Swagger UI will open at root URL

});

}

app.UseHttpsRedirection();

app.UseAuthorization();

app.MapControllers();

app.Run();

-Appsetting.json

{

"ConnectionStrings": {

"DefaultConnection": "Server=(localdb)\\mssqllocaldb;Database=aspnet-MvcExample-014b4166-fb86-417c-b509-4313b550e82d;Trusted\_Connection=True;MultipleActiveResultSets=true",

"EcomConnection": "Server=DESKTOP-H233I23\\SQLEXPRESS01;Database=MvcDatabase;Trusted\_Connection=True;MultipleActiveResultSets=true;TrustServerCertificate=True"

},

"Logging": {

"LogLevel": {

"Default": "Information",

"Microsoft.AspNetCore": "Warning"

}

},

"AllowedHosts": "\*"

}