Module 11

What to learn

Code First Approach Entity Framework
One-to-One Relationship
One-to-many Relationship
One-to-many Relation
Fluent API
One-to-One Relationship
One-to-many Relationship
One-to-many Relation
Migration Commands
Add-Migration
Remove-Migration
Update-Database
Script-DbContext
Get-DbContext
Drop-Database

Practice Exercise

Practice 1

Do the exercise from MSDN for given topics

Assignment Exercise

Assignment 1

A toy manufacturing company manufactures different types of toys. The company has several manufacturing plants. Each plant manufactures different types of toys. A customer can place the order for these toys. Each order may contain one or more toys. Each customer has multiple ship-to addresses. To promote the business, the company offers different schemes based on the order value. use Store procedure

Assignment 2

Define Relationship and do crud operation customer, customer can view all the products and search the record and place an order.

Assignment 3

Duration: 2 Hours
Objective: Practice core Entity Framework concepts, including setup, CRUD operations, relationships, migrations, Fluent API configurations, and advanced querying.

Assignment Details

Scenario:

You are building a small database for an online store with the following requirements:

- The store sells **Products**, each belonging to a **Category**.
- Customers can place **Orders**, and each order contains multiple **Products**.
- The database must support common operations such as adding, updating, retrieving, and deleting records.

Tasks

1. Setting Up Entity Framework (20 minutes)

Create a new console or ASP.NET Core application.

Add the required NuGet packages for Entity Framework Core and SQL Server.

Create the following entities:

Create a DbContext class named AppDbContext with DbSet properties for the above entities.

Configure a SQL Server connection string in AppDbContext.

2. Migrations and Seeding (20 minutes)

- Enable migrations and create an initial migration.
- Apply the migration to create the database.
- Seed the database with the following initial data:
  - **Categories**: Electronics, Furniture.
  - **Products**:
    - Laptop (Price: 1500, Category: Electronics).
    - Sofa (Price: 800, Category: Furniture).

3. CRUD Operations (20 minutes)

- Add a new product: **Phone** (Price: 700, Category: Electronics).
- Retrieve and display all products in the database.
- Update the price of the **Laptop** to 1600.
- Delete the **Sofa** product from the database.

4. Relationships and Queries (30 minutes)

Implement a one-to-many relationship between Category and Product (already modeled in the entities).

Add a product to an order using the Order entity.

Write a query to retrieve all products along with their categories using the Include method.

Write a query to retrieve the top 2 most expensive products.

Write a query to count the number of products in each category.

5. Fluent API and Advanced Configuration (30 minutes)

Use Fluent API to configure the Name property of Product to be required and have a maximum length of 100.

Configure a composite key for the Order entity using Id and OrderDate.

Specify that the Price column in Product should have a precision of 10 and a scale of 2 using Fluent API.

csharp

Copy code

```
public class Product   {      public int Id { get; set; }      public string Name { get; set; }
    public decimal Price { get; set; }      public int CategoryId { get; set; }      public
Category Category { get; set; }   } public class Category   {      public int Id { get; set;
}      public string Name { get; set; }      public List<Product> Products { get; set; }   }
public class Order   {      public int Id { get; set; }      public DateTime OrderDate { get;
set; }      public List<Product> Products { get; set; }   }
```

Online Reference

No online Reference

.NET Core Web API

WEB API (old)

Authentication And Authorization (WEBAPI)(old)

FullStackDevelopment_WIth_Dotnet_AND_Angular