Entity Framework Core

ARCTECH INFO PRIVATE LIMITED

Entity Framework Core

Entity framework is an ORM (Object Relational Mapping) tool

- ORM is a technique of accessing a relational database

Earlier developers had to write ADO.NET code to save or retrieve data from underlying database

- Prior to .NET 3.5
- This was a cumbersome and error prone process
- Microsoft introduced a framework called “Entity Framework” to automate all database related activities

Developers can work with objects and properties rather than database tables and columns

Data-oriented applications can be created with less code compared with traditional applications.

EF Core classes (.NET 5)

`DbContext` is an integral part of Entity Framework

It represents a session with the database

It includes `DbSet` objects which represent the tables

It can be used to query from a database

It is used to group together changes like insert, update, delete

These changes will be written to the underlying database as a single unit

Use EF Core (.NET 5)

Install the necessary Nuget packages

- Microsoft.EntityFrameworkCore & Microsoft.EntityFrameworkCore.SqlServer
- Or Oracle.EntityFrameworkCore for Oracle Database (ver 5)
- *Note: For .NET 5 install the 5.x.y versions of the packages*

Create a folder in the project, called **DataAccess** and a class in this folder called `ApplicationDbContext` inheriting from `DbContext`.

```
public class ApplicationDbContext : DbContext
{
    protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
    {
        optionsBuilder.UseSqlServer(@"Data Source=.;Initial Catalog=SampleStore;Integrated Security=True");
    }
}
```

Use EF Core (.NET 5)

Now for any table in the database which your application requires

- Create a Model class with the same name as the Table (usually in the **Models** folder), and properties matching the fields in the Student table. E.g.,

```
public class Student
{
	[Key]
	public int RollNo { get; set; }
	public string Name { get; set; }
}
```

- Create a property in the `ApplicationDbContext` class of type `DbSet<T>`

```
public DbSet<Student> Students { get; set; }
```

Use EF Core (.NET 5)

In any controller or service class, use the `Students` property from the `ApplicationDbContext` class to perform CRUD operations as below

- Create an object of the `ApplicationDbContext` class. See example below.

```
public Test()
{
    var context = new ApplicationDbContext()
}
```

- Select all students

```
var students = _applicationDbContext.Students.ToList();
```

- Insert a student

```
_applicationDbContext.Students.Add(student);
```

- Update a student

```
_applicationDbContext.Update(student);
```

- Delete a student

```
_applicationDbContext.Students.Remove(student);
```

Points to consider

- Insert, update and delete statements are performed as a unit by EF Core
- So, to commit the changes to the database you have to call the following before exiting the controller or service method

```
_applicationDbContext.SaveChanges();
```