Enterprise Application Development - Entity Framework & ADO.NET

Entity Framework (EF) in .NET

Entity Framework (EF) is an Object-Relational Mapper (ORM) for .NET applications. It simplifies data access by allowing developers to interact with databases using C# objects instead of writing raw SQL queries.

Key Features of Entity Framework:

1. ORM (Object-Relational Mapping) - Converts database tables into C# classes.
2. Eliminates SQL Queries - Uses LINQ instead of SQL.
3. Approaches: Database First, Code First & Model First.
4. Automatic Change Tracking - Tracks changes automatically.
5. Migrations - Handles schema changes easily.

Entity Framework CRUD Operations

1. Install Entity Framework

Run in Package Manager Console:

Install-Package EntityFramework

2. Create a Database Model (Entity)

```
public class Student
{
    [Key]  // Primary Key
    public int Id { get; set; }
    public string Name { get; set; }
    public int Age { get; set; }
}
```

3. Create Database Context

```
public class AppDbContext : DbContext
{
    public AppDbContext() : base("Data Source=.;Initial Catalog=StudentDB;Integrated Security=True") { }
    public DbSet<Student> Students { get; set; }
}
```

4. Perform CRUD Operations

Insert Data (Create):

```
using (var context = new AppDbContext())
{
    var student = new Student { Name = "John Doe", Age = 22 };
    context.Students.Add(student);
    context.SaveChanges();
    Console.WriteLine("Student inserted successfully.");
}
```

Retrieve Data (Read):

```
using (var context = new AppDbContext())
{
    var students = context.Students.ToList();
    foreach (var student in students)
    {
        Console.WriteLine($"ID: {student.Id}, Name: {student.Name}, Age: {student.Age}");
    }
}
```

Update Data (Update):

```
using (var context = new AppDbContext())
{
    var student = context.Students.FirstOrDefault(s => s.Id == 1);
    if (student != null)
    {
        student.Name = "Michael Smith";
        student.Age = 23;
        context.SaveChanges();
        Console.WriteLine("Student updated successfully.");
    }
}
```

Delete Data (Delete):

```
using (var context = new AppDbContext())
{
    var student = context.Students.FirstOrDefault(s => s.Id == 1);
    if (student != null)
```

```
    {
        context.Students.Remove(student);
        context.SaveChanges();
        Console.WriteLine("Student deleted successfully.");
    }
}
```

Enable Migrations and Update Database:

```
Enable-Migrations
Add-Migration InitialCreate
Update-Database
```

This completes the CRUD operations using Entity Framework in .NET.