**WRITEUP**:-

**A. Data Access Layer (DAL):**

1. Entity Classes:

Two entity classes, DeptMaster and EmpProfile, have been defined to represent the structure of the database. These classes use the Entity Framework Code First Approach.

DeptMaster

DeptCode (int): Unique identifier for the department.

DeptName (string): Name of the department.

EmpProfiles (ICollection<EmpProfile>): Collection of employee profiles associated with the department.

EmpProfile:

EmpCode (int): Unique identifier for the employee.

DateOfBirth (DateTime): Date of birth of the employee.

EmpName (string): Name of the employee.

Email (string): Email address of the employee.

DeptCode (int): Foreign key referencing the department.

DeptMaster (DeptMaster): Navigation property representing the associated department.

2. Context Class:

A DbContext class named ContextClass has been created to manage the database interactions. It includes DbSet properties for DeptMaster and EmpProfile.

3. Seed Data Class:

A class named SeedData inherits from DropCreateDatabaseIfModelChanges<ContextClass>. It overrides the Seed method to provide default data for the DeptMaster table when the model changes.

4. Repository Pattern for EmpProfile:

A EmpProfileRepository class has been created to implement the Data Repository Pattern. It includes methods for saving, retrieving, updating, and deleting employee profiles.

**B. Business Logic Layer (BLL):**

1. Invoke DAL Functionalities:

A BusinessLogic class has been added to the Business Logic Layer. This class initializes an instance of EmpProfileRepository and provides methods to invoke functionalities from the DAL class.

C. App Service Layer:

1. Web API Controller:

An ASP.NET Web API controller named EmployeeController has been implemented. It uses attribute-based routing to define routes for RESTful operations on employee data. The controller includes methods for:

Retrieving all employees (GetAllEmployees).

Retrieving an employee by code (GetEmployeeByCode).

Saving a new employee (SaveEmployee).

Updating an existing employee (UpdateEmployee).

Deleting an employee by code (DeleteEmployee).

2. Attribute-Based Routing and Swagger Support:

The implemented functionalities in the Web API controller use attribute-based routing for clean and readable URL patterns. Additionally, Swagger support has been enabled for documentation and testing of the Web API.

GITHUB LINK:- https://github.com/Mogal74/Mogal74-Phase-3-Endproject.git