**Project Question**

**Problem Statement:**

**Create a RESTful API to manage a collection of pets available for adoption. The API should allow clients to retrieve a list of pets, retrieve a single pet by ID, filter pets by type using route parameters and query string parameters, and create new pet entries.**

* Controller: **PetsController**
* This controller will handle all the HTTP requests related to pet adoption management.
* It will include methods to retrieve a list of pets, retrieve a single pet by ID, filter pets by type, and create new pet entries.
* **Data Folder:** Create a folder named Data and have the **ApplicationDbContext.cs** file to set up the database context for the application with the DbSet property **Pets.**
* Model: Pet
* Create a folder named Models and create the model for Pet.
* Represents a pet available for adoption.
* **Properties**:

1. **PetId**: Integer - The unique identifier of the pet (auto-increments by 1)
2. **Name**: String - The name of the pet.
3. **Type**: String - The type of the pet (e.g., Dog, Cat).
4. **Breed**: String - The breed of the pet.
5. **Age**: Integer - The age of the pet.
6. **Description**: String - A brief description of the pet.

**Controller Methods:**

**1. GET /api/pets**

* Retrieves a list of all pets.
* Status: 200
* Route: **/api/pets**
* Method: **public ActionResult<IEnumerable<Pet>> GetPets()**
* Response: List of Pet objects

**2.GET /api/pets/{id}**

* Retrieves a single pet by its ID.
* Status: 200
* Route: **/api/pets/{id}**
* Method: **public ActionResult<Pet> GetPetById(int id)**
* Parameters: id: Integer (Route parameter)
* Response: Pet object with the specified ID

**3.GET /api/pets/filter?type={type}**

* Retrieves a list of pets filtered by type.
* Route: **/api/pets**
* Status: 200
* Method: **public ActionResult<IEnumerable<Pet>> GetPetsByType([FromQuery] string type)**
* Query String Parameter: type: String
* Response: List of Pet objects filtered by the specified type

**4.POST /api/pets**

* Creates a new pet entry.
* Route: **/api/pets**
* Status: 201
* Method: **public ActionResult<Pet> CreatePet(Pet pet)**
* Request Body: Pet object (JSON)
* Response: Pet object representing the newly created pet entry

**Example Requests:**

**Retrieve all pets:**

* Request: **GET /api/pets**
* Response: List of all pets in the system.

**Retrieve a pet by ID:**

* Request: **GET /api/pets/1**
* Response: Details of the pet with ID 1.

**Retrieve pets filtered by type**:

* Request: **GET /api/pets/filter?type=Dog**
* Response: List of pets belonging to the "Dog" type.

**Create a new pet entry:**

* **Request:**

POST /api/pets

Content-Type: application/json

{

"Name": "Buddy",

"Type": "Dog",

"Breed": "Golden Retriever",

"Age": 3,

"Description": "A friendly and energetic dog."

}

* Response: Details of the newly created pet entry.

**Note:**

* **Make sure that, before running the test cases or submitting the project, the project is running on the port 8080**
* Link to view Swagger/idex.html page - **https://8080-\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*.premiumproject.examly.io/swagger/index.html**

**Commands to Run the Project:**

* **cd dotnetapp** - Select the dotnet project folder
* **dotnet restore** - This command will restore all the required packages to run the application.
* **dotnet run** - To run the application in port 8080 (The settings preloaded click 8080 Port to View)
* **dotnet build** - To build and check for errors
* **dotnet clean** - If the same error persists clean the project and build again

Install EF using the following commands :

**dotnet new tool-manifest**

**dotnet tool install --local dotnet-ef --version 6.0.6** -- Then use dotnet dotnet-ef instead of dotnet-ef.

**dotnet dotnet-ef** --To check the EF installed or not

**dotnet dotnet-ef migrations add "InitialSetup"** -- command to setup initial creating of tables mentioned in DBContext

**dotnet dotnet-ef database update** -- command to update the database

**To Work with SQLServer:**

(Open a New Terminal) type the below commands

**sqlcmd -U sa**

password: **examlyMssql@123**

1> create database DBName

2> go

>use DBName

>go

1> create table TableName(id int identity(1,1),........)

2> go

**Note:**

1. Use the below sample connection string to connect the MsSql Server

**private string connectionString = "User ID=sa;password=examlyMssql@123; server=localhost;Database=appdb;trusted\_connection=false;Persist Security Info=False;Encrypt=False";**

**Endpoint Screenshot:**

**﻿**

**Note :**

The project will not be submitted if "Submit Project" is not done atleast once