QA Back-End Technologies Basics

Regular Exam II – 02.03.2024

3. Zoo

You are given a .NET solution that implements a console application for simulating a zoo application. Your task is to create and implement **tests** using the **Integration Tests project** and **NUnit**.

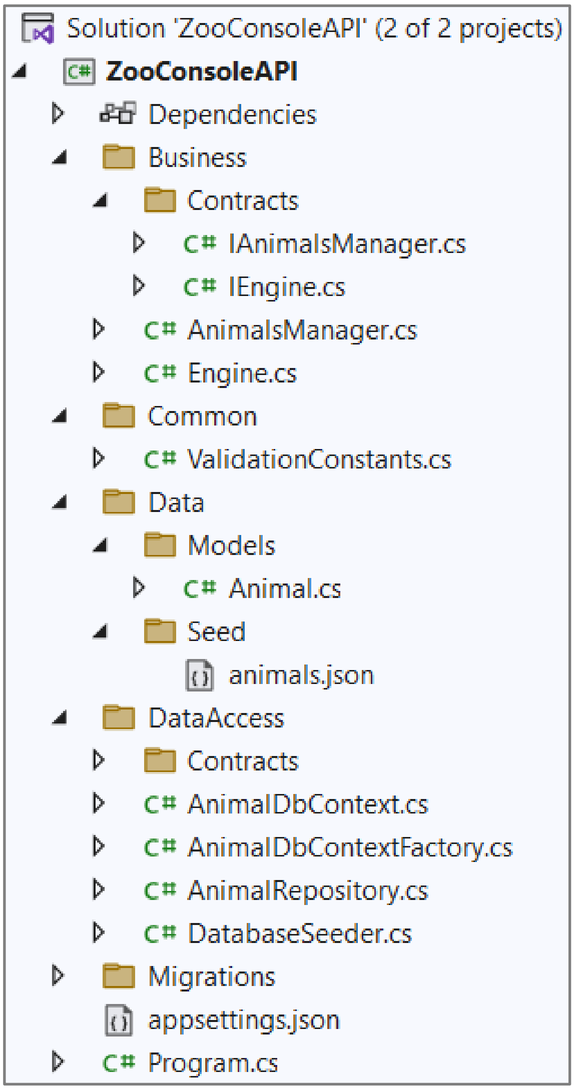
Introduction

ZooConsoleAPI is a console-based application built using the .NET Framework and the app manages a collection of animals.

It allows users to perform various operations, such as searching, adding, reading, updating and deleting animals from the database.

This application is designed to showcase **essential software development concepts and practices**.

Architecture Overview



```
Solution 'ZooConsoleAPI' (2 of 2 projects)
◢ ZooConsoleAPI
    ▷ Dependencies
    ◢ Business
        ◢ Contracts
            ▷ C# IAnimalsManager.cs
            ▷ C# IEngine.cs
        ▷ C# AnimalsManager.cs
        ▷ C# Engine.cs
    ◢ Common
        ▷ C# ValidationConstants.cs
    ◢ Data
        ◢ Models
            ▷ C# Animal.cs
        ◢ Seed
              animals.json
    ◢ DataAccess
        ▷ Contracts
        ▷ C# AnimalDbContext.cs
        ▷ C# AnimalDbContextFactory.cs
        ▷ C# AnimalRepository.cs
        ▷ C# DatabaseSeeder.cs
    ▷ Migrations
      appsettings.json
    ▷ C# Program.cs
```

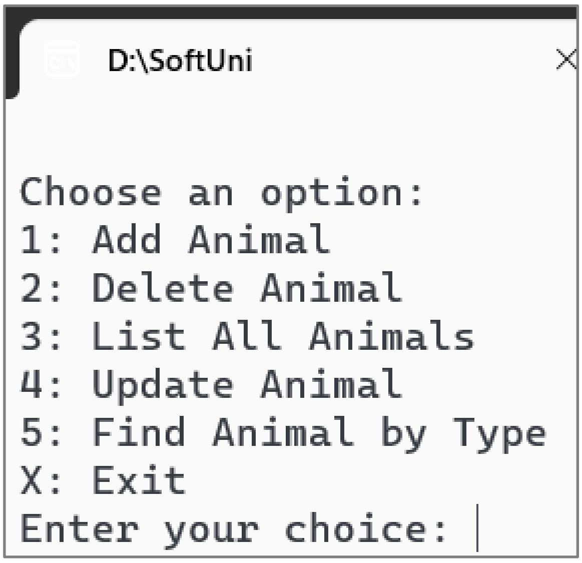
Data Validation Overview

The **Common** folder in the application contains shared constants that can be used across multiple layers of the application. In particular, the **`ValidationConstants`** class provides a centralized place to define the rules and constraints related to data validation for the domain models, particularly animals in this context.

Don't forget to use the data in this class while writing your tests.

Functionality Overview

When the application is executed, it presents a menu with options that the user can select from, as shown below:



```
Choose an option:
1: Add Animal
2: Delete Animal
3: List All Animals
4: Update Animal
5: Find Animal by Type
X: Exit
Enter your choice:
```

Users interact with the application by entering **commands** or **data** into the console. The application processes these inputs and performs the corresponding actions.

For example, when listing all the animals, the application, through **the `AnimalsManager`**, calls the **`GetAllAsync` method** on the **`AnimalsRepository`**. This **method is responsible for retrieving all animals records from the database**.

This retrieval is performed **asynchronously**, ensuring that the **application remains responsive**. Once the data is fetched, the **method iterates over the collection of `Animal` objects and prints their details** to the console.

Add Animal

The application promts the user to enter data for the details below:

- CatalogNumber
- Name
- Breed
- Type
- Age
- Gender
- IsHealthy

Input data is **validated** according to the **rules**, defined in the **business layer** and the **model's data annotations**.

Delete Animal

Deletes the input animal from the database.

List All Animals

Retrieves a list with all animals, present in the database.

Update Animal

Updates a animal's information.

Find Animal by Type

Finds an animal in the database by a type, that the user types in the console.

Integration Tests

Your task is to **implement the test methods in the provided test project**. These methods are placeholders that represent the **various scenarios** you will test within the application.

Navigate to **`ZooConsoleAPI.IntergrationTests.NUnit`** and use the **`IntegrationTests.cs`** to start writing your tests.

The whole solution is already configured with the necessary packages and references and works seamlessly.

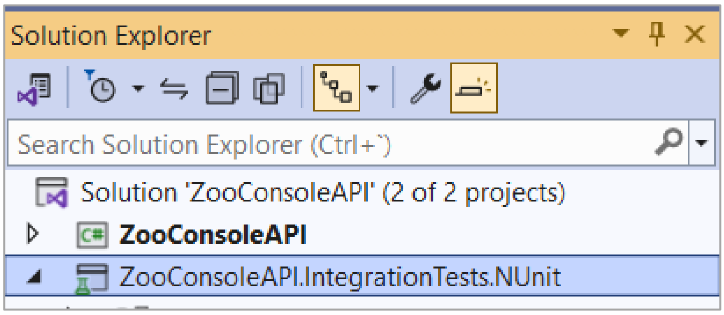
Additional Notes

- The test should be isolated, meaning it shouldn't be affected by or affect other tests.
- Integration tests often involve asynchronous operations, especially when dealing with database operations. Using await ensures that these asynchronous operations complete before assertions are made.
- While the data used in the test is arbitrary, it should resemble realistic data that the application might handle.

Good luck! 😉

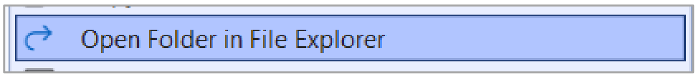
How to submit your work

- From the Solution Explorer in Visual Studio, right click on your **IntegrationTest Project and choose Open Folder in File Explorer**

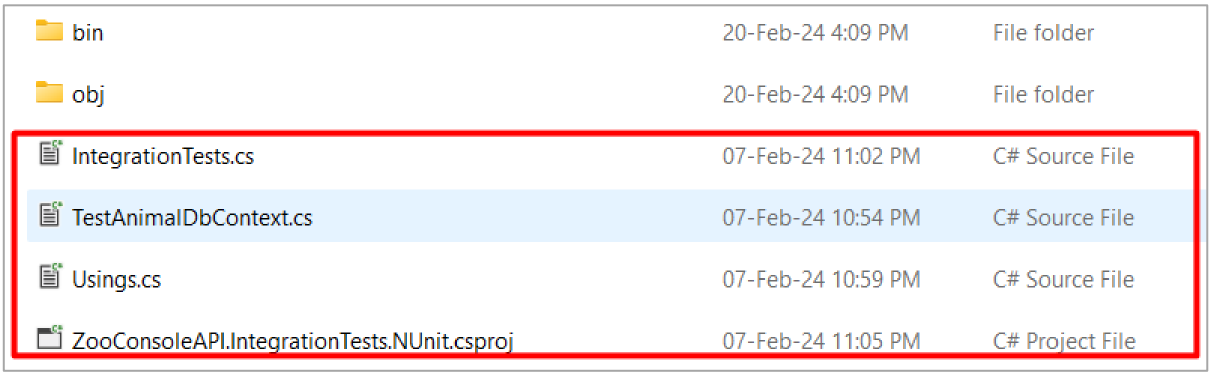


Solution 'ZooConsoleAPI' (2 of 2 projects)
ZooConsoleAPI
ZooConsoleAPI.IntegrationTests.NUnit

- **Choose Open Folder in File Explorer**



- Create a **.zip archive** with the following files:



bin	20-Feb-24 4:09 PM	File folder
obj	20-Feb-24 4:09 PM	File folder
IntegrationTests.cs	07-Feb-24 11:02 PM	C# Source File
TestAnimalDbContext.cs	07-Feb-24 10:54 PM	C# Source File
Usings.cs	07-Feb-24 10:59 PM	C# Source File
ZooConsoleAPI.IntegrationTests.NUnit.csproj	07-Feb-24 11:05 PM	C# Project File

NB!: ZIP archive (**NOT rar**, or any other)! **Don't include bin and obj folders**.

- Attach your archive in the Judge system.

Don't mind that it will give you "Compile Time Error". Third task of your exam will be checked manualy.