# QA Back-End Technologies Basics

# Regular Exam I – 25.02.2024

## Garden

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

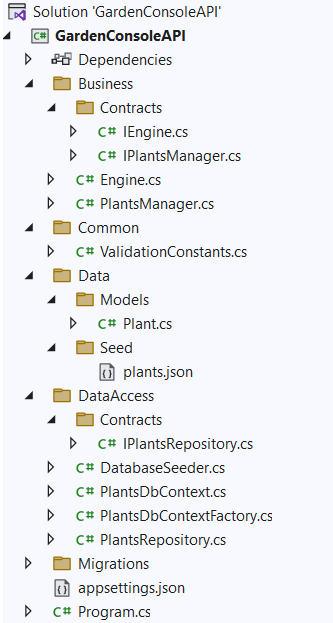
### Introduction

**GardenConsoleAPI** is a console-based application built using the .NET Framework and the app manages a collection of plants.

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

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

### Architecture Overview



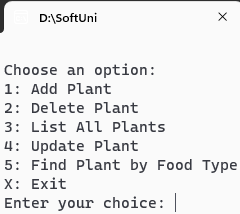
### Data Validation Overview

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 plants 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:

****

**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 plants, the application, through **the PlantsManager**, calls the **GetAllAsync** **method** on the **PlantsRepository**. This **method is responsible for retrieving all plants 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 Plant objects and prints their details** to the console.

#### Add Plant

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

* Name
* Plant Type
* Food Type
* Quantity

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

#### Delete Plant

Deletes the input plant from the database.

#### List All Plants

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

#### Update Plant

Updates a plant's information.

#### Find Plant by Food Type

Finds a plant in the database by a type of food, 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 **GardenConsoleAPI.IntergrationTests.NUnit** and use the **IntegrationTests.cs** to start writing your tests.

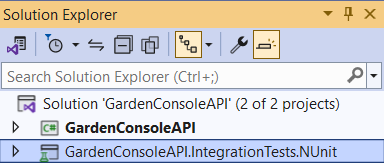
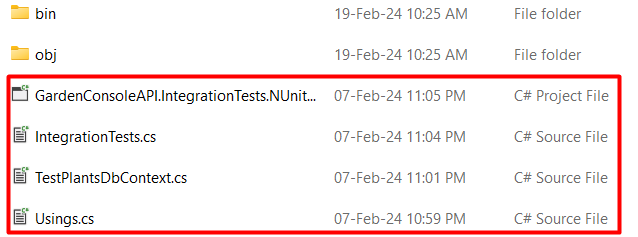
The whole solution is already configured with the necessary packages and references and works seamlessly.

#### 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**
* **Choose Open Folder in File Explorer**
* Create a **.zip archive** with the following files:  
  

**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.