crandles55 Feedback

# General

## Disclaimer

These tech showcase reviews tend to sound more negative than they are. While reviewing your showcase, the negatives will jump out more than the things that feel "right". This leads to the dislikes being mentioned more than the likes. Please keep that in mind while reading this feedback.

# Code

## Program.cs

Private Functions

Private functions are much more difficult to unit test and typically show the need for additional layers in your application code. What this means is that you could create an additional class that had these private function within, but create them as public so that they become accessible and testable.

Dependency Injection

Something that I think will help you in the future is to look into Dependency Injection. This will help you to manage your class and avoid the need for explicitly newing them up (lines 12-14). The reason that we want to avoid this is for testing. When you explicitly new these classes up, you lose the opporunity to inject Mock depenencies when writing your unit test. Dependency Injection is not built in to the dotnet sdk and has gotten large support.

Infinite Loop

You should avoid haveing 'while(true)' loops if possible. Instead of passing true, you can pass the condition that will break the loop.

Recursion

Looking at the your it appears that you started using some recursion to facilitate some of the looping. This is a really interesting idea. If you take that pattern just a little further, you will be able to remove the while loops entirely. It is worth noting that if you do go for a recursive pattern it can make switching to a Dependency Injection framework more difficult. It is not impossible, but it will complicate things to avoid an recursive dependency chain.

## PhotoValues

Calling Sibling Functions

To allow for clean mocking, it is good to avoid calling function defined in the same class. Typically this can be solved by creating a new class that contains the needed function.

## PhotoDto

Records

In dotnet 5 they introduced the 'record'. When you have a plain object, you can use the keyword 'record' instead 'class'. This will change the object to a value type object instead of reference. Allowing you to make cleaner assertions and enforce immutability by default. With your current application code, there are no changes needed to use record instead of class.

# Tests

Random Values

For you testing inputs, it is a good habit to use random values when possible. This can help reduce the amount of scenarios that you need to manually account for, and help catch edge cases that you may not consider. This would require a different patter than "data driving" the input of the test, but it can make some of the cases more explicit. Using random values, will also prove that your mocked values are coming from the source that you expect.