**Using Configuration**

1. Using IOptions pattern

This is the recommended way of binding configuration to POCO objects.

1. In this, developer defines JSON for configuration as follows:

A screen shot of a computer program

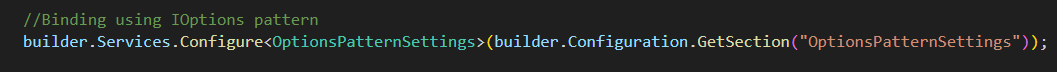
Description automatically generated with low confidence

1. Then creates a POCO class like above configuration as follows:

A screen shot of a computer program

Description automatically generated with low confidence

1. Then registers custom configuration object as follows:



1. Utilizes this configuration object by injecting it as follows:

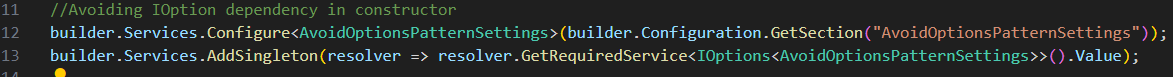
A screen shot of a computer program

Description automatically generated with low confidence

1. Avoiding IOptions pattern

As you can see in the above example IOptions dependency gets injected in constructor. Which is kind of confusing as we only need custom configuration object rather than IOptions. Thus I prefer I prefer following way

1. Registering configuration dependency



1. Injecting configuration dependency

A screen shot of a computer program

Description automatically generated with low confidence

1. Loading configurations from different files

Developers can directly add JSON configuration files using following:



But the problem with the above is that in real world applications you have multiple config files. Writing the same statement for all the files will clutter your program.cs file. Hence it is better to create an extension method that will contain code to add all files at one place.

This can be done as follows:



A screen shot of a computer code

Description automatically generated with low confidence