Dependency injection:

* Allows us to create an instance of a class whenever we want it .
* Private Set => means we need to have constructor

We have three options when it comes to Dependency injection:

1. Add Transient => every time we ask for the item => every time you ask for it, you’ll get a new instance => every time you come to a page, you get a new instance => it is a default behavior

Because usually we want to create a new instance every single time

1. Add Singleton => it is an instance of a class that gets the same instance of a class every single time => in the other word we create an instance of a class once and use that for entire application => every body uses that => it could be dangerous because every one can access the same data, so can change it => when we use transient the garbage collector, coAllect the data every time, but in singleton scenario it stays in the memory for the lifetime of the application

STATIC CLASS : cant inherit other classes or use dependency injection, so there is rare cases to use them!

1. Add Scoped => it is a singleton per person

WHY WE SHOULD USE INTERFACES? => cause it is simpler to remove dependencies when we do unit testing

Logger.LogInformation => (“Diplaying value of {Value1} and {Value2} ”,

Logic.Value1

Logic.Value2); => it is better to do this, instead of using $ =>

@inject.ILogger<> Logger

Serilog => there is a video