**Cart Report**

**Example Document to add watermark in the document.** There are 0 entries available in your cart. Please login to proceed next step.

The OperationLogger defines a constructor that requires each of the aforementioned marker interfaces, that is; ITransientOperation, IScopedOperation, and ISingletonOperation. The object exposes a single method that allows the consumer to log the operations with a given scope parameter. When invoked, the LogOperations method logs each operation's unique identifier with the scope string and message.

This tutorial shows how to use dependency injection (DI) in .NET. With Microsoft Extensions, DI is a first-class citizen where services are added and configured in an IServiceCollection. The IHost interface exposes the IServiceProvider instance, which acts as a container of all the registered services.

In this tutorial, you learn how to:

* Create a .NET console app that uses dependency injection
* Build and configure a Generic Host
* Write several interfaces and corresponding implementations
* Use service lifetime and scoping for DI

From the app output, you can see that:

* Transient operations are always different, a new instance is created with every retrieval of the service.
* Scoped operations change only with a new scope, but are the same instance within a scope.
* Singleton operations are always the same, a new instance is only created once.