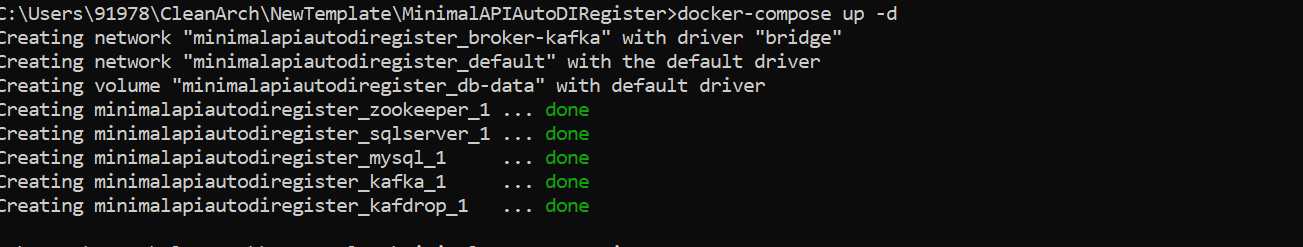
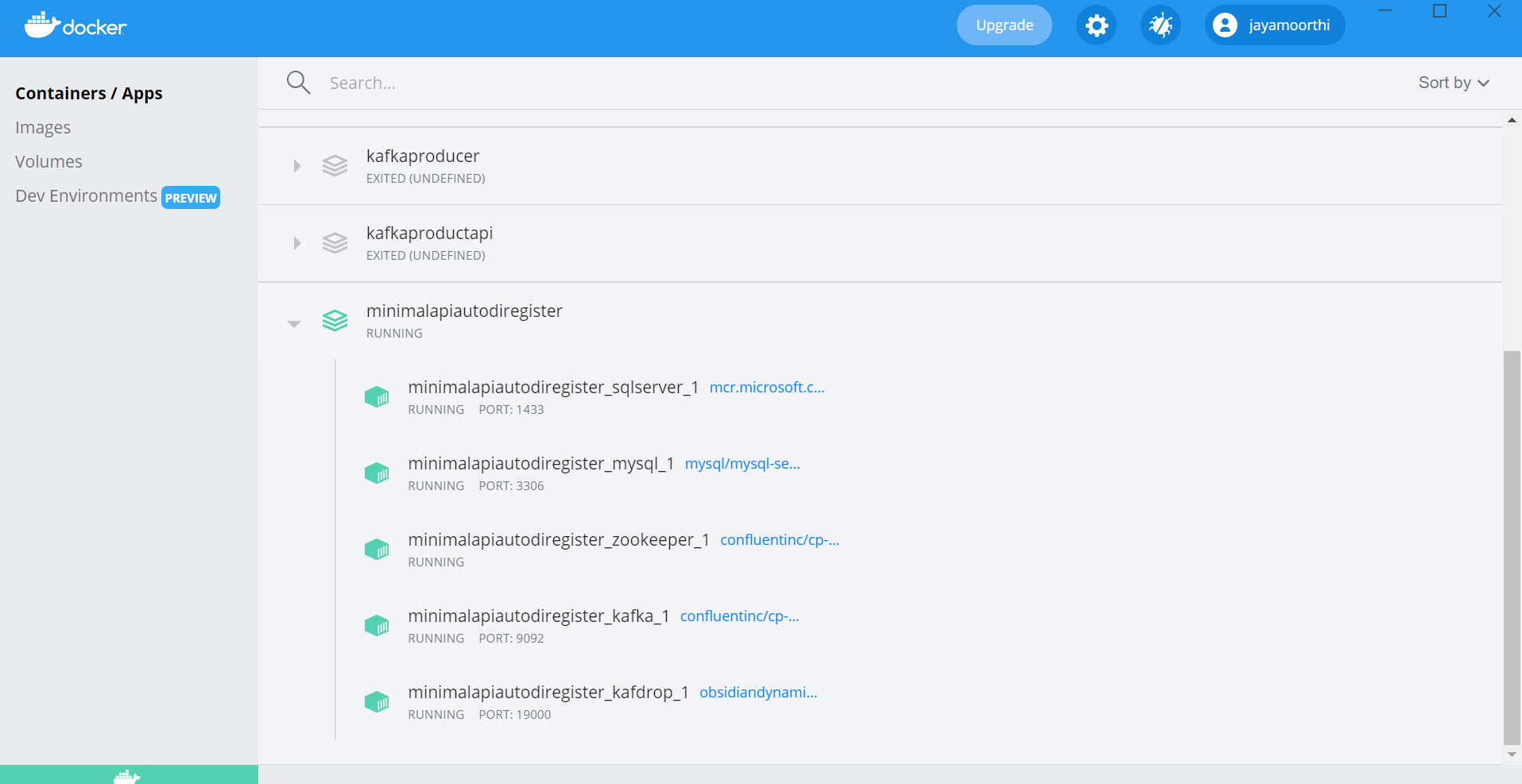
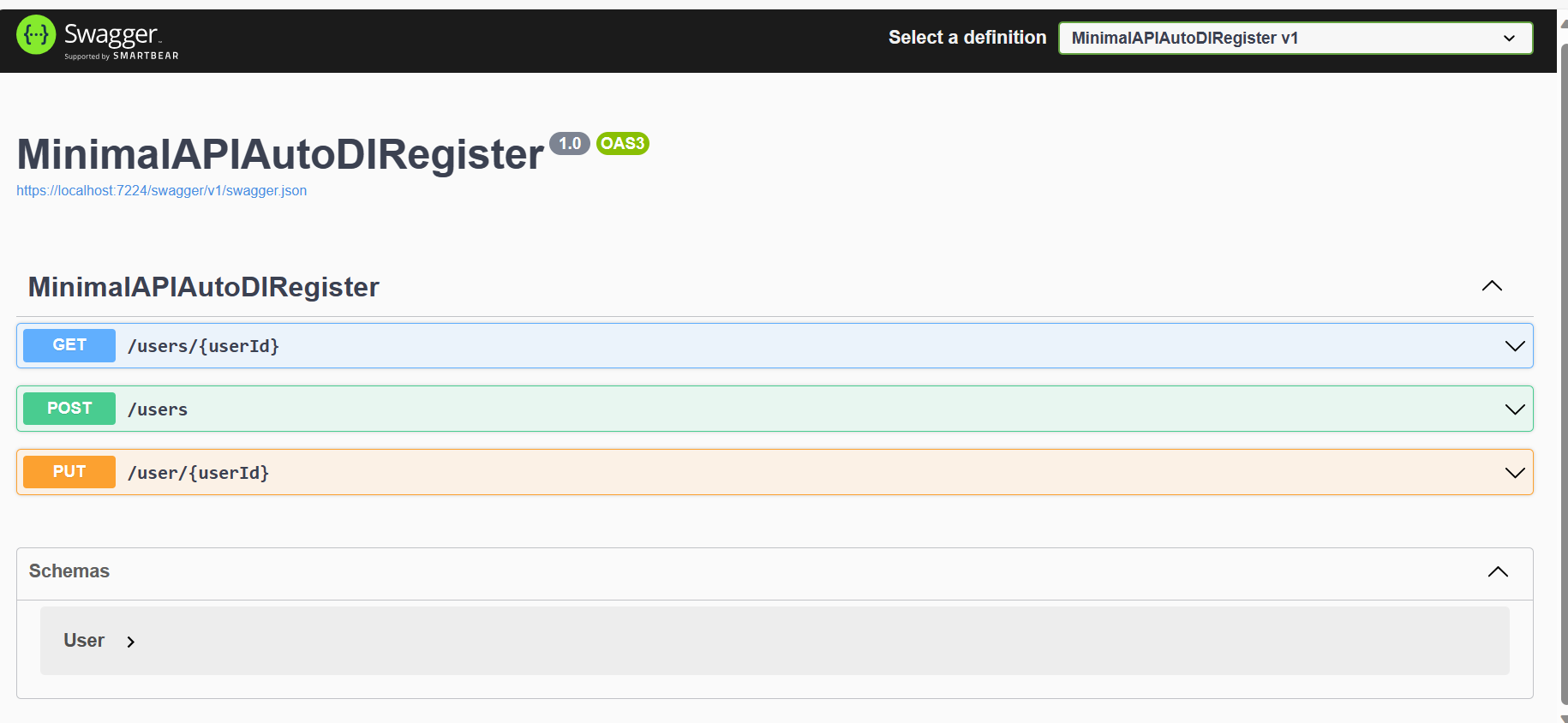
Auto DI Register for Endpoints with CQRS using minimal API’s in .NET 6  
  
Git source: https://github.com/jayamoorthi/MinimalAPIAutoDIRegister  
To practical implementation following items need to Installation in your system.

* Visual Studio 2022
* .NET 6
* ASP NET 6.0 Runtime
* SqlServer Database
* Apache Kafka
* Java Runtime Environment (JRE)
* Docker Desktop



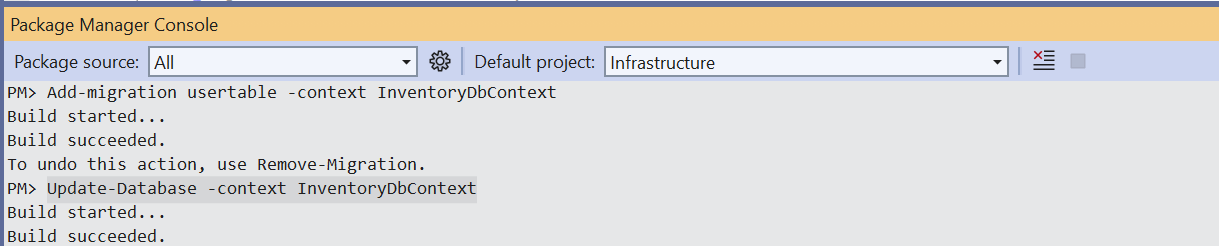
After successfully image created and running





Add Migration using cli comment in PMC

Add-migration usertable -context InventoryDbContext

Update-Database -context InventoryDbContext  
  
  


How to use Serilog in .net 6?

Logging is most important topics for any kind of application in software for troubleshooting the issue, error/runtime exception occurred on where it’s happened, help to request / response tracing application logic.

It’s used to write logging as structure based on template for configuration.

Git Source: https://github.com/jayamoorthi/MinimalAPIAutoDIRegister

Install below the nuget package in the application

Install-Package Serilog.AspNetCore

Install-Package Serilog.Sinks.Seq

**Log Levels**

* Trace – Detailed messages with sensitive app data.
* Debug – Useful for the development environment.
* Information – General messages, like the way we mentioned earlier.
* Warning – For unexpected events.
* Error – For exceptions and errors.
* Critical – For failures that may need immediate attention

Configure template format and to write file name for specific logging level

"Serilog": {

"MinimumLevel": {

"Default": "Information",

"Override": {

"Microsoft": "Warning",

"System": "Information",

"Microsoft.Hosting.Lifetime": "Information"

}

},

"Filter": [

{

"Name": "ByExcluding",

"Args": { "expression": "@mt = 'An unhandled exception has occurred while executing the request.'" }

}

],

"WriteTo": [

{

"Name": "File",

"Args": {

"path": "./logs/error/log-.txt",

"rollingInterval": "Day"

}

},

{

"Name": "File",

"Args": {

"path": "Logs/Info/applog\_.log",

"outputTemplate": "{Timestamp:o} [{Level:u3}] ({SourceContext}) {Message}{NewLine}{Exception}",

"rollingInterval": "Day",

"retainedFileCountLimit": 7

}

},

{

"Name": "File",

"Args": {

"path": "Logs/warning/applog\_.log",

"outputTemplate": "{Timestamp:o} [{Level:u3}] ({SourceContext}) {Message}{NewLine}{Exception}",

"rollingInterval": "Day",

"retainedFileCountLimit": 7

}

},

{

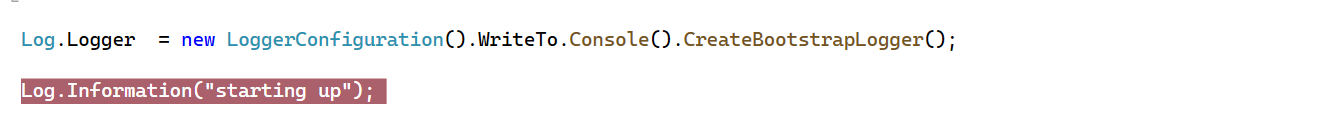
"Name": "Seq",

"Args": { "serverUrl": "http://localhost:5341" }

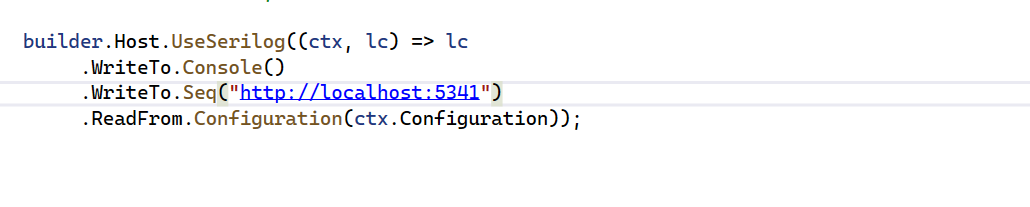
}

]

}  
  
Program.cs - Initalize the logger

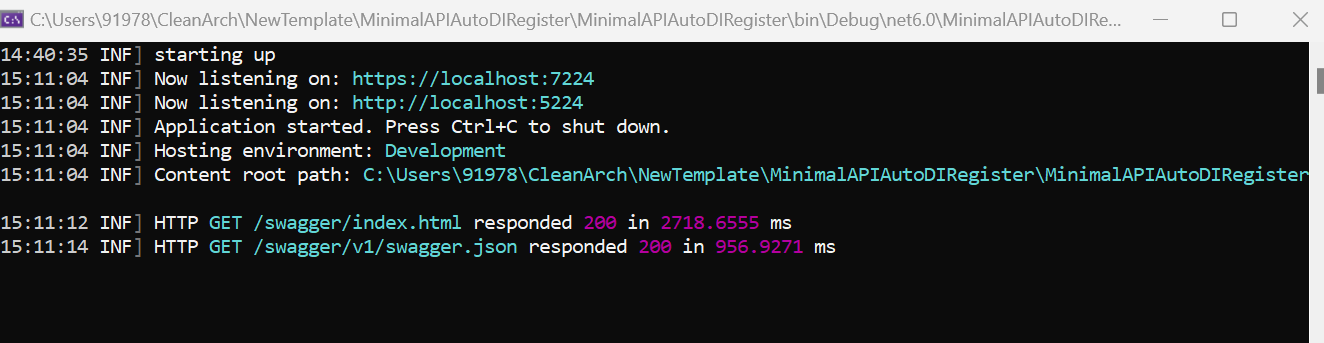
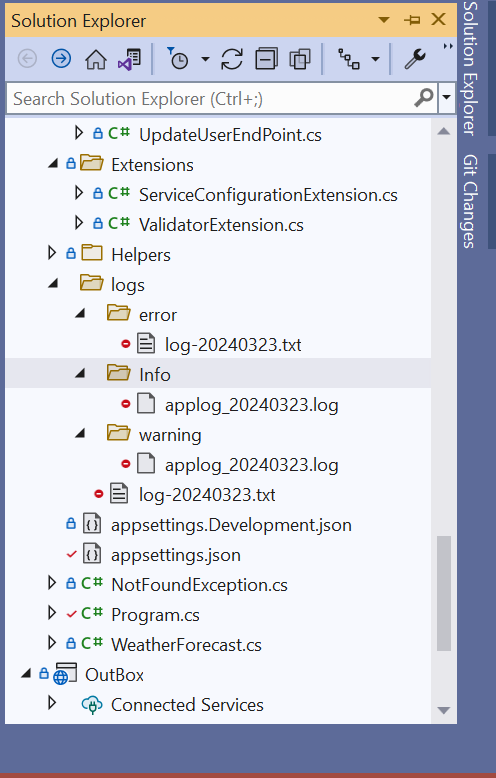


Configure the logging template and logging to console using build-in UseSerilog() method



Configure app middleware app.UseSerilogRequestLogging();



Now, it’s created file with directory and writing logs the error and info file.   
  
  
  


What is use of query filter?

Query filters are used to LINQ query predicates applied to domain entities in the models in the EF.

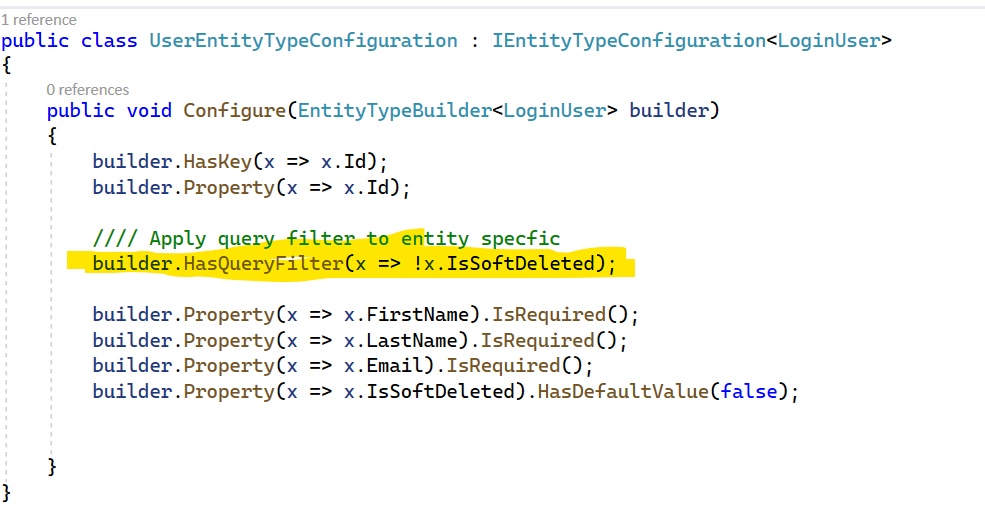
Use Case:

below the scenario’s we can use query filters in EF.

1. Soft Delete - Entity is defined IsSoftDelete property
2. Multi-Tenancy – Entity is defined TenantId property

**How to apply query filter entity specific in EF?**

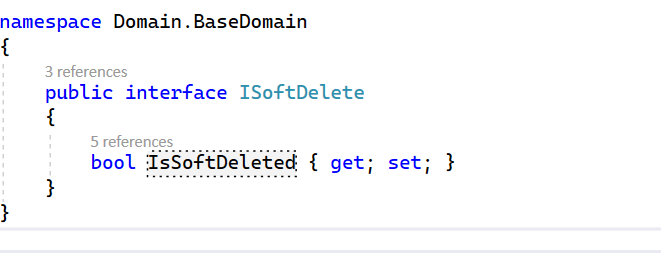
Entity specific filter can apply in the EntityTypeConfiguration in the modelbuilder using HasQueryFilter() method



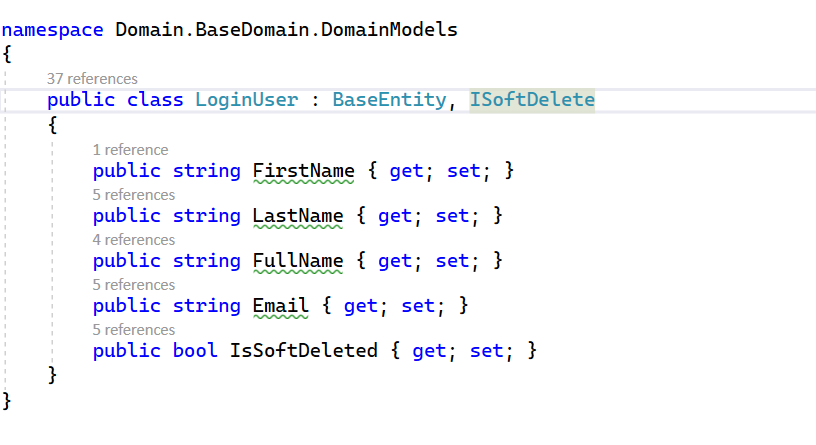
How to apply global query filter in EF?

Global query filters are LINQ query predicates applied to Entity Types in the metadata model usually in OnModelCreating().

Case 1 : Soft Delete implementation  
  
To applying SoftDelete query filter which is entites defined is IsSoftDelete property in their domain models.

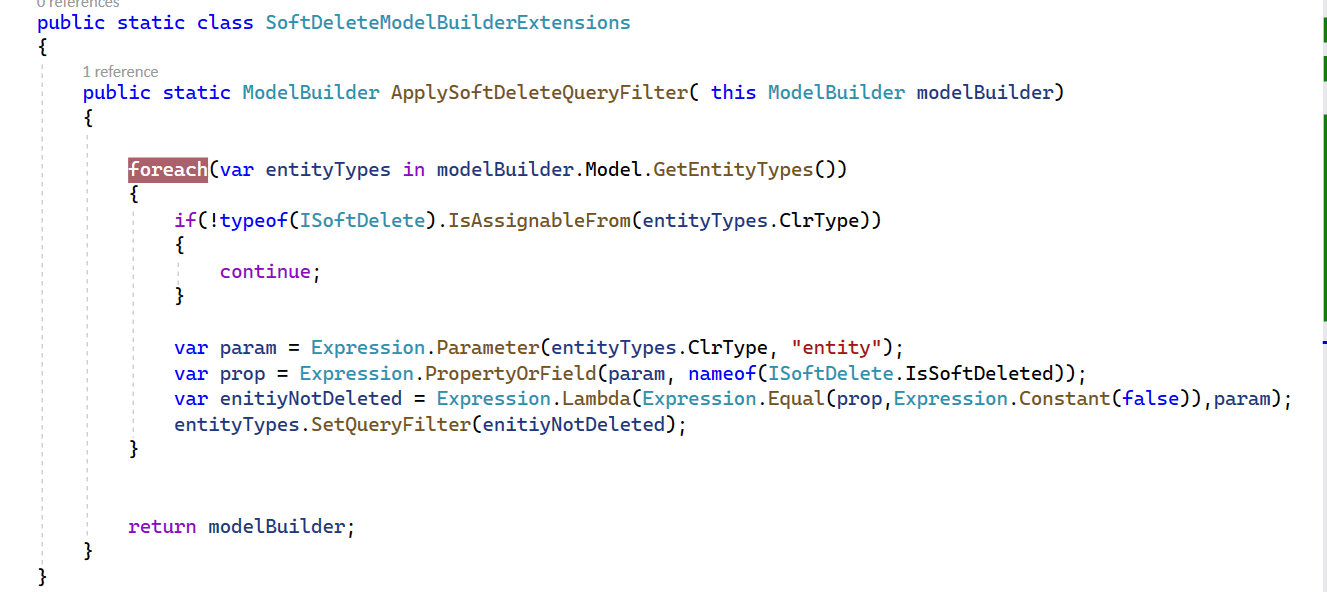
ISoftDelete Interface  
  


Domain Model – LoginUser class Inherited BaseEntity and ISoftDelete

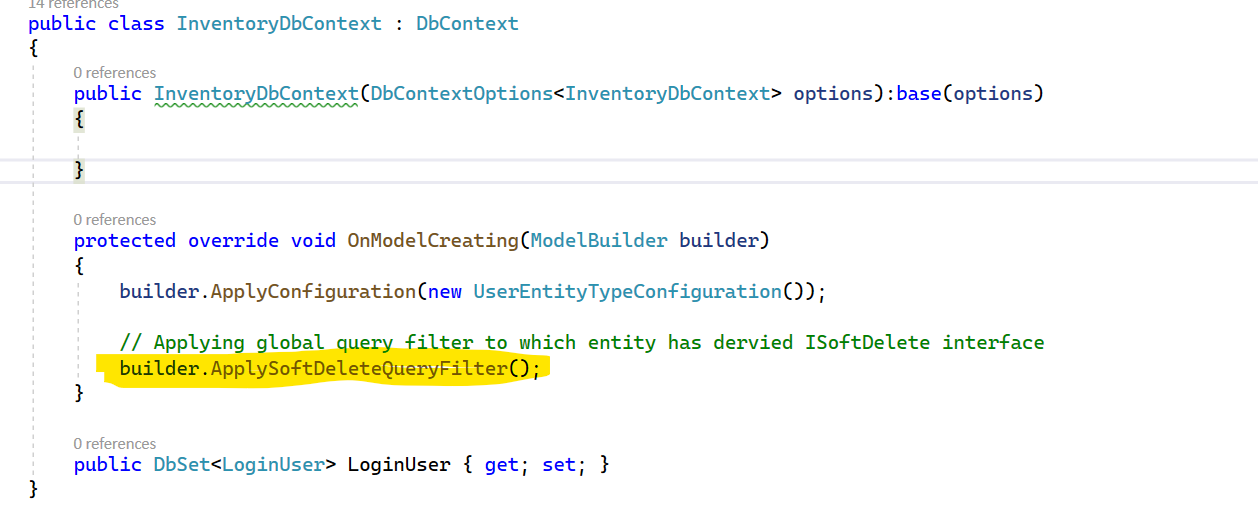


GlobalQueryFilterExtension class for SoftDelete filter.

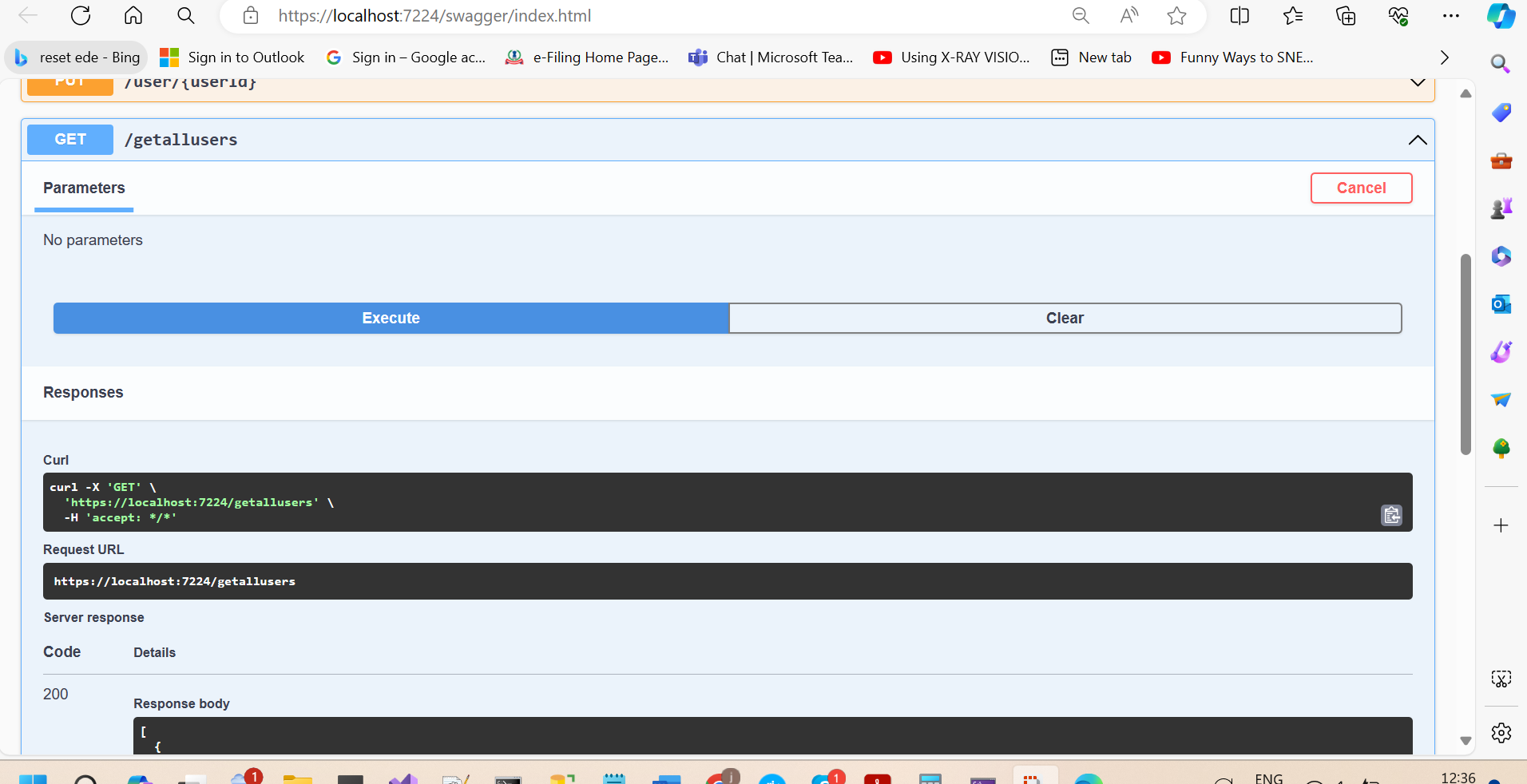
We need to write a custom logic to apply predicate for all entity types. Which is entity type has IsSoftDeleted Property else will not apply query filters.



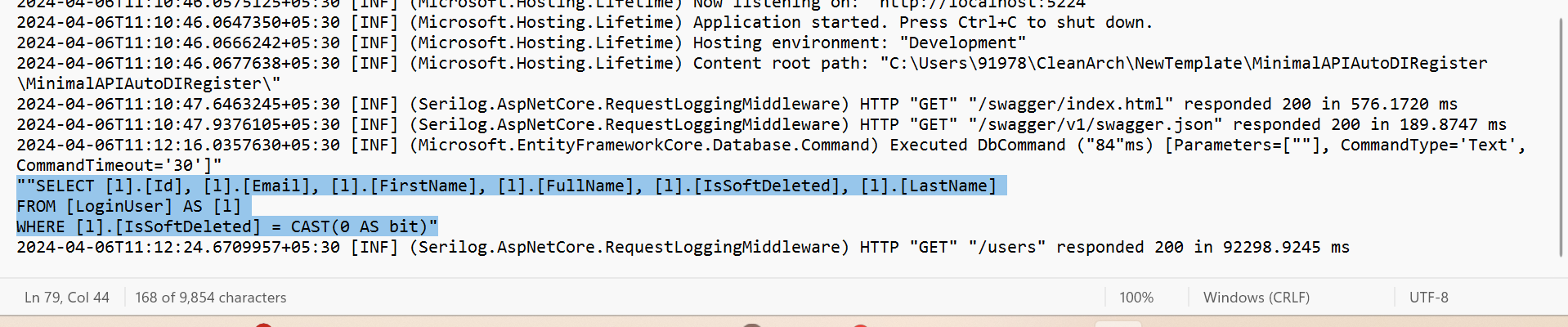
Add ExtensionQueryFilter to OnModelCreating() from InventoryDbContext class



Run then application, Swagger will trigger Get Method – Users



Then will go to info log file.



**How to disabled query filter for the specific entity?**

Filters may be disabled for individual LINQ queries by using the IgnoreQueryFilters()

