**Topic**

* Experience in Web Security concepts like - Encryption, PKI etc is desirable
* **Different ways to Authenticate a Web Application**

<https://medium.com/@vivekmadurai/different-ways-to-authenticate-a-web-application-e8f3875c254a>

* Supporting multiple authentication schemes in asp.net core webapi

<https://dejanstojanovic.net/aspnet/2021/december/supporting-multiple-authentication-schemes-in-aspnet-core-webapi/>

<https://www.abhith.net/blog/aspnet-core-using-multiple-authentication-schemes/>

<https://codepedia.info/jwt-authentication-in-aspnet-core-web-api-token>

# Concurrency, Parallelism, Threads, Processes, Async, and Sync — Related

<https://medium.com/swift-india/concurrency-parallelism-threads-processes-async-and-sync-related-39fd951bc61d>

**Design Pattern**

* **Mediator**

<https://www.dofactory.com/net/mediator-design-pattern>

https://www.ezzylearning.net/tutorial/mediator-design-pattern-in-asp-net-core

**Microservices**

<https://github.com/aspnetrun/run-aspnetcore-microservices>

<https://github.com/mehmetozkaya/AspnetMicroservices/blob/main/src/Services/Catalog/Catalog.API/Entities/Product.cs>

**Docker**

<https://learn.microsoft.com/en-us/dotnet/core/docker/build-container?tabs=windows>

* dotnet new console -o App -n DotNet.Docker –(Folder name App and Solution name Donet.Docker)
* dotnet run
* dotnet publish -c Release ( dll create inside Release)

📁 docker-working

└──📂 App

├──DotNet.Docker.csproj

├──Program.cs

└──📂 obj

├── DotNet.Docker.csproj.nuget.dgspec.json

├── DotNet.Docker.csproj.nuget.g.props

├── DotNet.Docker.csproj.nuget.g.targets

├── project.assets.json

└── project.nuget.cache

FROM mcr.microsoft.com/dotnet/sdk:6.0 AS build-env

WORKDIR /App

# Copy everything

COPY . ./

# Restore as distinct layers

RUN dotnet restore

# Build and publish a release

RUN dotnet publish -c Release -o out

# Build runtime image

FROM mcr.microsoft.com/dotnet/aspnet:6.0

WORKDIR /App

COPY --from=build-env /App/out .

ENTRYPOINT ["dotnet", "DotNet.Docker.dll"]

1. docker build -t counter-image -f Dockerfile . (create image [counter-image])
2. docker ps
3. docker images
4. docker create –-name show-counter counter-image (create container)
5. docker start core-counter
6. docker attach --sig-proxy=false core-counter [To get result]
7. docker run -it --rm counter-image [Docker provides the docker run command to create and run the container as a single command. This command eliminates the need to run docker create and then docker start. You can also set this command to automatically delete the container when the container stops. ]

**Install Mongo Db**

**docker run -d -p 27017:27017 --name shopping-mongo mongo**

**docker exec -it shopping-mongo /bin/bash**

**Install Redis**

**docker pull redis**

docker run -d -p 6379:6379 --name aspnetrun-redis redis

docker exec -it aspnetrun-redis /bin/bash

redis-cli

**Docker Compose**

docker-compose -f docker-compose.yml -f docker-compose.override.yml up -d