Readme 2024.08.24

The attached source code accompanies the CodeProject article
https://www.codeproject.com/Articles/5353867/Containerizing-a-NET-8-Blazor-application-and-depl
This source code illustrates step 5 described in the article (Dockerization of a .NET 8 Blazor application)
To use the source code, unizp the zip file in a folder, for example, in
C:\docker-demo

This creates the following path to the root folder of the project
C:\docker-demo\blazor-cloud-run-src\blazorapp

Major parts - overview

1) The folder "blazorapp" contains a scaffolded .NET 8 Blazor application. which was created using .NET SDK command
dotnet new blazor -n blazorapp -f net8.0 --no-https

2) Note that port number is set to 8088 in appsettings.json and in launchSettings.json

```
appsettings.json
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*",
  "Kestrel": {
    "Endpoints": {
      "Http": {
        "Url": "http://*:8088"
      }
    }
  }
}

launchSettings.json
{
  "$schema": "http://json.schemastore.org/launchsettings.json",
    "iisSettings": {
      "windowsAuthentication": false,
      "anonymousAuthentication": true,
      "iisExpress": {
        "applicationUrl": "http://localhost:38356",
        "sslPort": 0
      }
    },
    "profiles": {
      "http": {
        "commandName": "Project",
        "dotnetRunMessages": true,
        "launchBrowser": true,
        "applicationUrl": "http://localhost:8088",
        "environmentVariables": {
          "ASPNETCORE_ENVIRONMENT": "Development"
        }
      },
```

```
      "IIS Express": {
        "commandName": "IISExpress",
        "launchBrowser": true,
        "environmentVariables": {
          "ASPNETCORE_ENVIRONMENT": "Development"
        }
      }
    }
  }
```

3) Testing the application on local system
It is possible now to test the app by running the commands:

```
cd C:\docker-demo\blazor-cloud-run-src\blazorapp

dotnet run blazorapp
```

The app can be stopped in PowerShell with Ctrl-C

4) Publishing using .NET 8 SDK

```
cd C:\docker-demo\blazor-cloud-run-src\blazorapp

dotnet publish -c Release -f net8.0 -r linux-x64 -o
./bin/release/net8.0/publish/ --self-contained false
```

5) Dockerizing and testing Docker image

In PowerShell

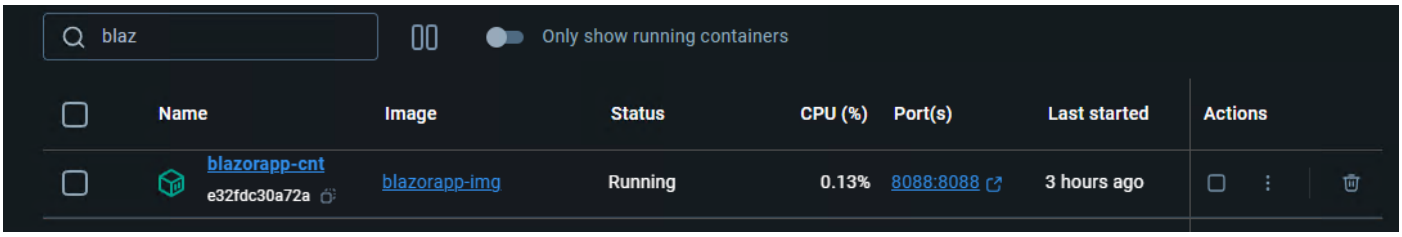
```
cd C:\docker-demo\blazor-cloud-run-src\blazorapp

docker build -t blazorapp-img -f  Dockerfile .

docker run -it -p 8088:8088 --name blazorapp-cnt blazorapp-img
```

Remarks concerning “docker run …”

When “docker run -it …” is executed, the container `blazorapp-cnt` built from image `blazorapp-img` becomes visible in the user interface of Docker Desktop.



It is important to notice that executing the command “`docker run -it …`” in PowerShell will also generate system warnings from ASP.NET 8 which are related to ASP.NET keys storage and keys encryption. These warnings do not cause any issues with the considered deployment use case, as the built image can be transferred to GCP, deployed on Cloud Run, and tested. Despite this being a viable result for the assumed dev environment, these warnings are valid considerations for a production environment on GCP, where attention should be given to aspects of ASP.NET keys management, such as GCP Secret Management, GCP key encryption configuration by the use of GCP's Key Management Service and/or integration of an X.509 certificate.