Background

1. This instruction is for deploying .NET Core Fibonacci REST web service with Docker
2. You should notice a folder "Fibo". The folder contains the source code and Dockerfile to build the container. A .dockerignore file has also been included to make the build context smaller
3. The application name is called "fibo", which is specified in the entrypoint in the Dockerfile. The dotnet command will run fibo.dll, which is our app

Building and running the Docker image

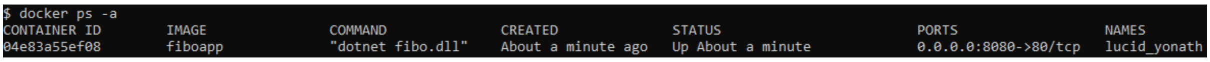
1. Open the command prompt and change directory to the folder "Fibo"
2. Use the following commands to build the docker image

   $ docker build -t fiboapp .

   ```
   $ docker build -t fiboapp .
   ```
3. Use the following commands to run the docker image

   $ docker run -d -p 8080:80 fiboapp

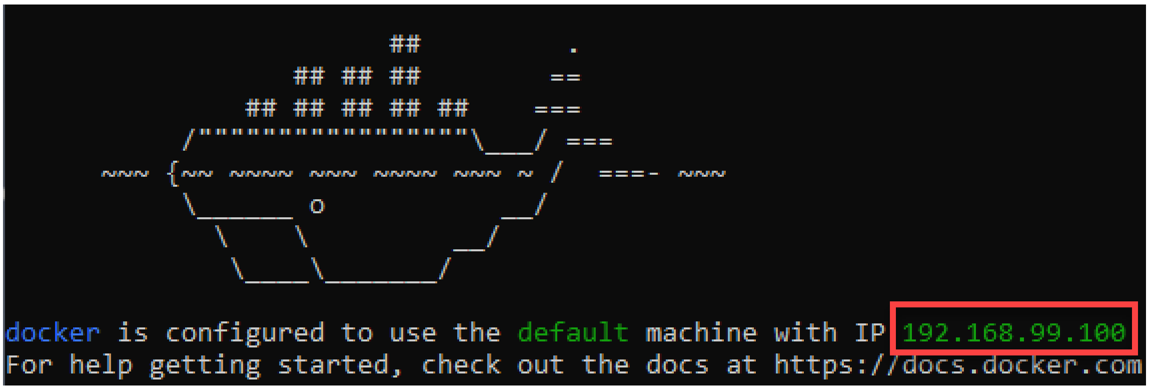
   ```
   $ docker run -d -p 8080:80 fiboapp
   ```
4. Use the command, "docker ps -a" to check the status of the new built and running container

   ```
   $ docker ps -a
   CONTAINER ID        IMAGE               COMMAND                CREATED              STATUS                        PORTS                    NAMES
   04e83a55ef08        fiboapp             "dotnet fibo.dll"      About a minute ago   Up About a minute             0.0.0.0:8080->80/tcp     lucid_yonath
   ```

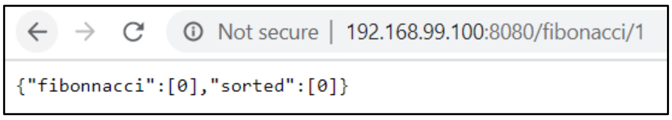
Testing of the webapp

1. If you are using Docker Desktop, open up a web browser, go to to http://localhost:8080/fibonacci/1
2. If you are using Docker toolbox, instead of localhost, replace localhost with the IP address provided when you started up Docker http://192.168.99.100:8080/fibonacci/1

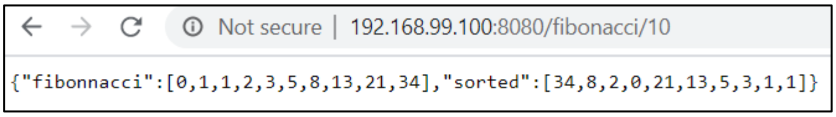
   

   ```
                           ##         .
                     ## ## ##        ==
                  ## ## ## ## ##    ===
              /"""""""""""""""""\___/ ===
         ~~~ {~~ ~~~~ ~~~ ~~~~ ~~~ ~ /  ===- ~~~
              \______ o           __/
                \    \         __/
                 \____\_______/

   docker is configured to use the default machine with IP 192.168.99.100
   For help getting started, check out the docs at https://docs.docker.com
   ```
3. The address performs a GET request to the web app by providing the "elements" as a number at the end of the URL. fibonacci/1 tells the web server to return the Fibonacci sequence with 1 element, giving the following result:

   ```
   {"fibonnacci":[0],"sorted":[0]}
   ```
4. Changing the number at the end of the URL will tell web server to return the corresponding Fibonacci sequence.

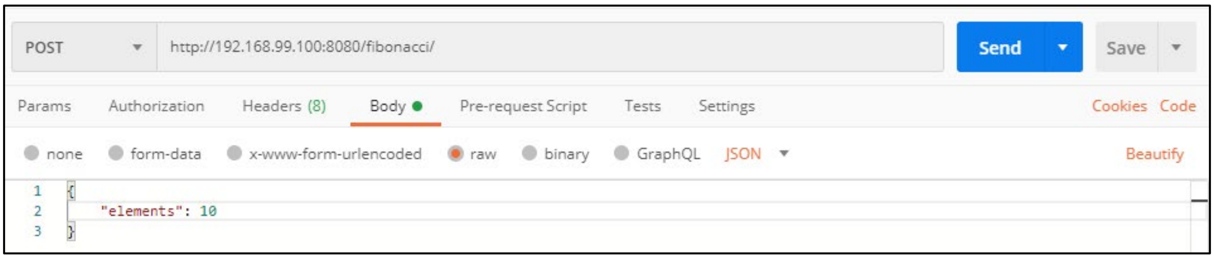
   

   ```
   {"fibonnacci":[0,1,1,2,3,5,8,13,21,34],"sorted":[34,8,2,0,21,13,5,3,1,1]}
   ```

5. The question requires the web app to return Fibonacci sequence using the following Json to http://myserver:8000/fibonacci

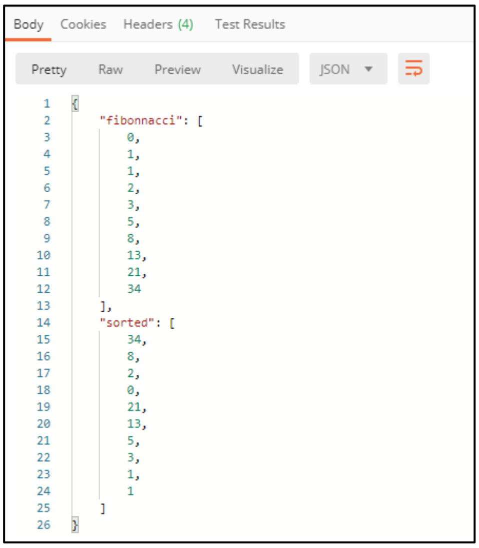
```
{
      “elements”: 10
}
```

5. This looks like a POST request, which can be tested using the postman app. Using postman app, we input the following parameter and click send to the address http://192.168.99.100:8080/fibonacci/ or (http://localhost:8080/fibonacci/ if using Docker Desktop)



```
{
    "elements": 10
}
```

6. We should obtain the following result:



```
{
    "fibonnacci": [
        0,
        1,
        1,
        2,
        3,
        5,
        8,
        13,
        21,
        34
    ],
    "sorted": [
        34,
        8,
        2,
        0,
        21,
        13,
        5,
        3,
        1,
        1
    ]
}
```