Deploy Application using ngnix-aspnet-mysql Containers and Docker compose

Task 1 (v1):

In this task I created an Amazon EC2 instance, utilizing user data to automatically install Docker during launch. Establish an SSH connection to the instance and execute the command 'docker info' to retrieve Docker-related information.

Steps:

1. **Create EC2 Instance:**
   (i) Log in to your AWS Management Console.
   (ii) Navigate to the EC2 dashboard.
   (iii) Click "Launch Instance" and choose an Amazon Machine Image (AMI) that suits your needs.
   (iv) Configure the instance settings (instance type, VPC, subnet, etc.).
   (v) Add storage, tags, and security group settings as required.
   (vi) Review and launch the instance.
2. **Add User Data:**
   (i) While configuring the instance, find the "Advanced Details" or "Configure Instance" section.
   (ii) Locate the "User data" field.
   (iii) Add a cloud-init script to install Docker, for example: #include get.docker.com.
3. **Launch Instance:**
   (i) Continue through the instance launch process and choose or create an SSH key pair for secure access.
4. **Connect via SSH:**
   (i) Once the instance is running, note its public IP address or DNS name.
   (ii) Open terminal (for Windows, you can use tools like PuTTY or Windows Subsystem for Linux).
   (iii) Run the following command to connect via SSH: (take path according to your pc)
   Command:
   ```
   ssh -i key.pem ubuntu@IP_address
   ```
5. **Run Docker Info:**
   (i) After successfully connecting to the EC2 instance via SSH, you'll be in the instance's command-line interface.
   (ii) Run the following command to check Docker information:
   Command:
   ```
   docker info
   ```

Task 2 (v2):
In this version I have cloned repository from the link "https://github.com/docker/awesome-compose/tree/master/nginx-aspnet-mysql" to the ubuntu machine and performed docker commands

Below is project structure

```
 .
├── backend
│   ├── Dockerfile
│   ├── aspnet.csproj
│   └── Program.cs
├── db
│   └── password.txt
├── compose.yaml
├── proxy
│   ├── conf
│   └── Dockerfile
└── README.md
```

The compose.yml file defines an application with three services proxy, backend and db. When deploying the application, docker compose maps port 80 of the proxy service container to port 80 of the host as specified in the file.

For deploy with docker compose use the following command:
`docker compose up -d`

Expected Result:

```
[+] Running 3/0
 ✔ Container nginx-aspnet-mysql-db-1      Running          0.0s
 ✔ Container nginx-aspnet-mysql-backend-1  Running         0.0s
 ✔ Container nginx-aspnet-mysql-proxy-1   Running          0.0s
```

After the application starts, navigate to http://localhost:80 in your web browser or run:

Command
`curl localhost:80`

Output

```
["Blog post #0","Blog post #1","Blog post #2","Blog post #3","Blog post #4"]
```

Finally to stop and remove container

```
docker compose down
```

Output:

```
[+] Running 4/4
 ✔ Container nginx-aspnet-mysql-proxy-1    Removed                          0.0s
 ✔ Container nginx-aspnet-mysql-backend-1  Removed                          0.0s
 ✔ Container nginx-aspnet-mysql-db-1       Removed                          0.0s
 ✔ Network nginx-aspnet-mysql_default      Removed                          0.1s
```

Submitted by:
Anuj Sharma