**Prescriptive Data DevOps Engineer Exercise Instructions**

**Instruction to execute multiple scaled services using Nginx and Docker.**

Requirements:

**Docker and Docker-compose should be installed.**

To install docker go to the link provided and follow the instructions as per your operating system.

Link to install docker: <https://docs.docker.com/get-docker/>

Link to install docker-compose: <https://docs.docker.com/compose/install/>

**Steps:**

1. **Download the source code from GitHub**
   1. Open terminal
   2. Goto work directory
   3. Git clone <https://github.com/code.git>
   4. Cd code/
2. **Create an external network**

Let’s create an external network to connect with Nginx

$ docker network create nginx-network

Confirm network creation

$ docker network ls

output: nginx-network will be listed

1. **Build a docker using docker-compose**

Now, we can build the Dockerfile. For now, we can exclude the cache data by adding --no-cache argument to ignore existing images.

$ docker-compose build --no-cache

1. **Run the docker images**

The next step is to run the docker image.

$ docker-compose up [User “docker-compose up -d” to execute in daemon mode]

1. **Test API Gateway**

Goto the browser of your choice and search localhost:80 OR localhost

[To verify the Nginx dividing the incoming load refresh the browser it will display different “IP” generated from the container].

**Configuration for maximizing the availability and spinup**

1. Creating within the current infrastructure.
   1. Create a new folder structure with anew Nginx name
   2. Edit the ngingx.conf in load balancer and add a new Nginx service name in the list of the container.
2. To maximize the availability a production-ready orchestration like ECS, EKS, etc should be used so that we can implement auto scalability feature both in server-specific as well as pod/container-specific.

**Maintaining Logging in the application**

There are multiple ways for logging the system. As the Stdout is enabled in nginx for standard output this output can be directly maintained in AWS cloud-watch. The system performance we can add an agent for AWS ContainerInsight (best for the container monitoring). Moreover, by using external third-party software like DatadogDq, Nagio we can have extra features for logging the system.

**Applying the security**

1. Enclose the service within a VPC(Cloud specific) with minimal outbound where the connection between the load balancer and the container is maintained through a private network.
2. The security group should be implemented as a firewall between the services.
3. Private auth for internal communication should be used like istio.