**Required AWS Elastic Beanstalk Environment Setup Updates - DO NOT SKIP**

The process for creating an Elastic Beanstalk environment has changed (this is very common, as the AWS UI changes frequently). There is now a flow of 6 steps that you will be taken through.

**Step 1:**

You will need to select the Platform of Docker. You then must manually change from Docker running on 64bit Amazon Linux 2023 to **Docker running on 64bit Amazon Linux 2**. The 2023 platform was just released in early August and is currently not compatible with our CI/CD automation.

A screenshot of a computer

Description automatically generated

Scroll down to the Presets section and make sure that **free tier eligible** has been selected:

A screenshot of a web page

Description automatically generated

Click the Next button to move to Step #2.

**Step 2:**

You will be taken to a Service Access configuration form. Here, you should select **Use an existing service role**, which will auto-select the **aws-elasticbeanstalk-service-role** and auto-populate the **aws-elasticbeanstalk-ec2-role** as the instance profile:

A screenshot of a computer

Description automatically generated

You may now click the **Skip to Review** button as Steps 3-6 are not applicable.

Click the Submit button and wait for your new EBS application and environment to be created and launch.

**Required Updates for Docker Compose**

**1. Rename the current docker-compose file**

Rename the **docker-compose.yml** file to **docker-compose-dev.yml**. Going forward you will need to pass a flag to specify which compose file you want to build and run from:  
docker-compose -f docker-compose-dev.yml up  
docker-compose -f docker-compose-dev.yml up --build  
docker-compose -f docker-compose-dev.yml down

**2. Create a production-only docker-compose.yml file**

The production compose file will follow closely what was written in the Dockerrun.aws.json. There are two major differences:

**No Container Links**: In the ["Forming Container Links"](https://www.udemy.com/course/docker-and-kubernetes-the-complete-guide/learn/lecture/11437364#questions) lecture we add the client and server services to the links array of the nginx service. Docker Compose will handle this container communication automatically for us.

**Environment Variables:** When using a compose file we will need to explicitly specify the environment variables each service will need access to. The value for each variable must match the corresponding variable names you have specified in the Elastic Beanstalk environment. The AWS variables are created in the ["Setting Environment Variables"](https://www.udemy.com/course/docker-and-kubernetes-the-complete-guide/learn/lecture/11437388#questions) lecture.

**Note - You must not have a Dockerrun.aws.json file in your project directory.**If AWS EBS sees this file the deployment will fail. If you have previously followed this course and deployed to the old Multi-container platform **you will need to delete this file before moving to the new platform!!!**

Complete docker-compose.yml file:

1. version: "3"
2. services:
3. client:
4. image: "cygnetops/multi-client-10-14"
5. mem\_limit: 128m
6. hostname: client
7. server:
8. image: "cygnetops/multi-server-10-14"
9. mem\_limit: 128m
10. hostname: api
11. environment:
12. - REDIS\_HOST=$REDIS\_HOST
13. - REDIS\_PORT=$REDIS\_PORT
14. - PGUSER=$PGUSER
15. - PGHOST=$PGHOST
16. - PGDATABASE=$PGDATABASE
17. - PGPASSWORD=$PGPASSWORD
18. - PGPORT=$PGPORT
19. worker:
20. image: "cygnetops/multi-worker-10-14"
21. mem\_limit: 128m
22. hostname: worker
23. environment:
24. - REDIS\_HOST=$REDIS\_HOST
25. - REDIS\_PORT=$REDIS\_PORT
26. nginx:
27. image: "cygnetops/multi-nginx-10-14"
28. mem\_limit: 128m
29. hostname: nginx
30. ports:
31. - "80:80"