**SOP - Dockerizing a Plain HTML Page with Nginx**

**Below steps to be performed -**

1. Basic HTML Page:

- Create a plain HTML page named `index.html` with some content (e.g., "Hello, Docker!").

2. Nginx Configuration:

   - Create an Nginx configuration file named `nginx.conf` that serves the `index.html` page.

   - Configure Nginx to listen on port 80.

3. Dockerfile:

   - Create a `Dockerfile` to define the Docker image.

   - Use an official Nginx base image.

   - Copy the `index.html` and `nginx.conf` files into the appropriate location in the container.

   - Ensure that the Nginx server is started when the container is run.

***Note: After building the Dockerfile, we need to connect to AWS CLI and configure it. Then build will be created for that we need to perform below commands as well –***

1. aws configure (provide keys, secret access keys, region and formats)
2. aws sts get-caller-identity (to verify the user credentials of AWS)
3. Create a Public ECR Repository:

aws ecr-public create-repository --repository-name sal-nginx-docker

1. Authenticate Docker with ECR –

aws ecr-public get-login-password --region us-east-1 | docker login --username AWS --password-stdin public.ecr.aws

* 1. Retrieve a token and authenticate with Docker - Linux/Mac: ~/.docker/config.json
  2. Remove or comment out the "credsStore" or "credHelpers" line. Ensure the file looks like this:

{

"auths": {

"public.ecr.aws": {}

}

}

* 1. Then Retry the AWS ECR login.

1. Once login is successful, run - sudo apt update, sudo apt upgrade -y.
2. Install Docker as well. And run the below commands to build, tag and push the image.
   1. docker build -t sal-nginx-docker .
   2. docker tag sal-nginx-docker:latest public.ecr.aws/975050024946/sal-nginx-docker:latest
   3. docker images
   4. docker run -d -p 8082:80 sal-nginx-docker
   5. docker push public.ecr.aws/975050024946/sal-nginx-docker:latest
3. Once it is pushed successfully, the image will be pushed to the ECR repository.



