Building Docker Image with NGINX and Error Solving

Building Docker Image:

We're in the process of building a Docker image using a **Dockerfile** to configure an **NGINX web server** and deploy a website. The website's files are contained in a ZIP file and need to be extracted and served by NGINX.

Error:

The primary challenge occurs after downloading and unzipping the ZIP file/directory. The extracted directory undergoes a name change, complicating the process of moving the extracted website content from that directory to the root directory (/usr/share/nginx/html/) for web content on a web server.

This step ensures that the content is accessible to visitors when they browse the site because we don't know the exact name of the directory after unzipping.

Solution Approach:

To address this challenge, we'll create a shell script named **extract.sh** to handle the extraction dynamically. This script will find the first directory created in **/tmp/** after unzipping and move its contents to **/usr/share/nginx/html/,** ensuring that the website files are placed in the correct location regardless of the directory's name.

Script Creation:

Navigate to the directory where your Dockerfile is located.

Use a text editor to create a new file named extract.sh:

run command: vim extract.sh

Copy and paste the following script content into extract.sh:

```
#!/bin/bash
# Find the first directory in /tmp/ and move its contents to /usr/share/nginx/html/
# Start a loop to iterate over each directory in /tmp/
for d in /tmp/*/; do
# Move all files and directories within the current directory to /usr/share/nginx/html/
   mv "$d"/* /usr/share/nginx/html/
   break
done
```

Save the file and exit the text editor.

Integration with Dockerfile:

Ensure both your **Dockerfile** and **extract.sh** are in the same directory.

Modify your Dockerfile to copy **extract.sh** into the Docker image and make it executable:

```
FROM nginx:latest

RUN apt-get update \
   && apt-get install -y unzip \
   && rm -rf /var/lib/apt/lists/*

ADD https://www.free-css.com/assets/files/free-css-templates/download/page295/antique-cafe.zip /opt/

COPY extract.sh /usr/local/bin/
RUN chmod +x /usr/local/bin/extract.sh

RUN unzip /opt/antique-cafe.zip -d /tmp/ \
   && /usr/local/bin/extract.sh \
   && rm /opt/antique-cafe.zip \
   && rm -rf /tmp/*

EXPOSE 80

CMD ["nginx", "-g", "daemon off;"]
```

This Dockerfile does the following:

- Update package lists and install unzip: It updates the package lists and installs the unzip package, which is needed to extract the ZIP file.
- Download ZIP file: It adds a URL to download a ZIP file and places it in the "/opt/" directory.
- Copy and make script executable: It copies the extract.sh script to "/usr/local/bin/" and makes it executable.
- Unzip and run script: It unzips the downloaded ZIP file to "/tmp/" directory, executes the extract.sh script, then removes the downloaded ZIP file and the temporary directory.
- Expose port 80: It exposes port 80 to allow external access to NGINX server.
- Start NGINX server: It starts the NGINX server with the specified configuration (daemon off;), ensuring it runs in the foreground.

Rebuild your Docker image:

Run command → **docker build -t my_nginx_image .**

Now your Image built with the name my_nginx_image

Run a container using your newly built image:

Run command → **docker run -d -p 80:80 my_nginx_image**

Now Browse the IP address of your local host machine (Ec2 Public IP Address) you will see your website is accessible.

Summary:

By implementing this solution, we ensure that the website files are correctly extracted and accessible within the Docker container, overcoming the challenge of unpredictable directory names. This Dockerfile automates the process of downloading, extracting, and serving website content using NGINX, providing a reliable and efficient solution for deploying web applications in a Docker environment. With clear steps for updating package lists, downloading and extracting the ZIP file, and configuring NGINX to serve the content, this solution simplifies the deployment process and ensures consistency in accessing website files regardless of the directory name variations.