# Qoala- DevOps Assignment: Debugging and Deploying Dockerized Application

**Name: Devansh Jain**
**Roll Number: 21UCS060**
**Link :** http://13.235.49.80/

## Overview

- Deployed a Flask application using Nginx as a reverse proxy with Docker for container management.

## Challenges & Resolutions

**Screenshots for the errors are attached herewith : Error_SS**

1. **Docker Image Build Errors**
   - **Issue**: Typos in `Dockerfile, nginx.conf` and `docker-compose.yml`.
   - **Solution**: Fixed syntax, port issues, and file paths, defined the `build` directive in `docker-compose.yml` to streamline image building and ensure consistent deployments directly within the compose configuration.
2. **Changes in Dockerfile**
   - **Issue**: Missing `HTML` file, incorrect port, and command configurations prevented proper application setup.
   - **Solution**: Adjusted file paths, set the correct port, and updated commands to ensure everything runs smoothly.
3. **Zero MAC Address Display**
   - **Issue**: Docker's virtual network displayed a zero MAC address as it accessed the virtual MAC address before encountering a real MAC address.
   - **Solution**: Updated the code to check only the `eth0` interface, where the real MAC address is usually present, and return "N/A" if unavailable.

## Resolution Steps

**Screenshots for the resolved issues are attached herewith : Succ_SS**

1. **Dockerfile Corrections**:
   - Corrected incorrect package installations, `Dockerfile` syntax, and paths, specifically within the `COPY` commands, `EXPOSE` statements, and `CMD` command.
   - Built the Docker images after implementing these corrections and confirmed successful build completion.
2. **Configuration Fixes**:

    - Updated `nginx.conf` to properly route requests to the Python application via the `proxy_pass` directive, and resolved syntax errors in the worker and connection settings.
3. **Testing and Verification**:
    - Conducted tests in a local environment by accessing the application through `http://localhost:80`, confirming a successful connection to the Python app.
    - Verified that the Nginx access logs accurately reflected request entries, indicating successful requests to the Flask application.
4. **Docker Image Build and Deployment**:
    - Defined the `build` directive in the `docker-compose.yml` to facilitate streamlined image building, ensuring consistency in deployments.
5. **HTML File and Command Adjustments**:
    - Created the `HTML` file, corrected port settings, and command configurations in the `Dockerfile` to ensure proper application setup and functionality.
6. **MAC Address Handling**:
    - Updated the code to restrict MAC address retrieval to the ‘`eth0`’ interface, ensuring that a valid MAC address is returned and defaulting to "N/A" if it is not available.

## AWS Endpoint Creation (Public IPv4 address : 13.235.49.80)

**Link : http://13.235.49.80/** **Reference screenshots : AWS_SS**

1. **EC2 Instance Setup**:
    - Ubuntu server is selected in the Mumbai region.
    - Security settings are configured to allow traffic on ports 80.
    - RSA key is set up for secure SSH access.
2. **Connecting to the Server**:
    - Adjusted key file permissions and connected to the server via SSH.
3. **File Transfer**:
    - Project files copied to the server, excluding unnecessary files and folders.
4. **Installing Docker**:
    - The server is updated, and Docker along with Docker Compose is installed to facilitate deployment.
5. **Application Deployment**:
    - The application is started on the server using Docker Compose.
6. **Deployment Verification**:
    - Docker is verified to be running, and the application is confirmed to be accessible on the specified ports(80).

**************End of the Report** ************