Docker setup

DOCKERFILE FOR SERVICE

Docker file is created for all the services and the frontend.

Example for a Docker file for a service:

# Base image with OS

FROM ubuntu:18.04

# Install java-8

RUN apt-get update && \

apt-get -y install openjdk-8-jdk

# Copy the JAR file to the container

COPY target/ZullServer-0.0.2-SNAPSHOT.jar /home/ZullServer-0.0.2-SNAPSHOT.jar

# exposing port no to a container

EXPOSE 5055

# run the JAR file

CMD java -jar /home/ZullServer-0.0.2-SNAPSHOT.jar

Dockerfile for agent front-end:

#BASE IMAGE with os

FROM ubuntu:18.04

#to update & install apache2

RUN apt-get update

RUN apt-get -y install apache2 && apt-get -y install unzip

RUN rm -f /var/www/html/index.html

RUN rm -rf /var/www/html

RUN mkdir /var/www/ecs.mobilefirstfinance.com

RUN mkdir /var/www/ecs.mobilefirstfinance.com/html

#copy dist

COPY /dist/\* /var/www/ecs.mobilefirstfinance.com/html

RUN echo "ServerName localhost" >> /etc/apache2/apache2.conf

#enable ssl module and rewrite module

RUN a2ensite default-ssl

RUN a2enmod ssl

RUN a2enmod rewrite

#change document root name

RUN sed -i -r '5d' /etc/apache2/sites-enabled/default-ssl.conf

RUN sed -i -r '5 i DocumentRoot /var/www/ecs.mobilefirstfinance.com/html' /etc/apache2/sites-enabled/default-ssl.conf

RUN sed -i -r '6 i ServerName ecs.mobilefirstfinance.com' /etc/apache2/sites-enabled/default-ssl.conf

RUN sed -i -r '12d' /etc/apache2/sites-enabled/000-default.conf

RUN sed -i -r '12 i DocumentRoot /var/www/ecs.mobilefirstfinance.com/html' /etc/apache2/sites-enabled/000-default.conf

#copy and unzip ssl certificates

COPY ssl-certificates.zip /root/

RUN unzip /root/ssl-certificates.zip -d /root/

#add ssl-certificates path

RUN sed -i -r '32d' /etc/apache2/sites-enabled/default-ssl.conf

RUN sed -i -r '32d' /etc/apache2/sites-enabled/default-ssl.conf

RUN sed -i -r '32 i SSLCertificateKeyFile /root/ssl-certificates/mff.key' /etc/apache2/sites-enabled/default-ssl.conf

RUN sed -i -r '33 i SSLCertificateFile /root/ssl-certificates/mff.crt' /etc/apache2/sites-enabled/default-ssl.conf

#to enable https

RUN sed -i -r '228 i <Directory /var/www/html>' /etc/apache2/apache2.conf

RUN sed -i -r '229 i AllowOverride All' /etc/apache2/apache2.conf

RUN sed -i -r '230 i </Directory>' /etc/apache2/apache2.conf

#to redirect to https

RUN sed -i -r '29 i RewriteEngine On' /etc/apache2/sites-enabled/000-default.conf

RUN sed -i -r '30 i RewriteCond %{HTTPS} !=on' /etc/apache2/sites-enabled/000-default.conf

RUN sed -i -r '31 i RewriteRule ^/?(.\*) https://%{SERVER\_NAME}/$1 [R,L]' /etc/apache2/sites-enabled/000-default.conf

#expose http and https port nos

EXPOSE 80 443

#run apache2

CMD apache2ctl -D FOREGROUND

DOCKER FILE ON BITBUCKET:

Docker file is placed in Bitbucket or GITHUB along with the source code as shown below.

A screenshot of a computer

Description automatically generated

**Integrate Bitbucket to Jenkins**

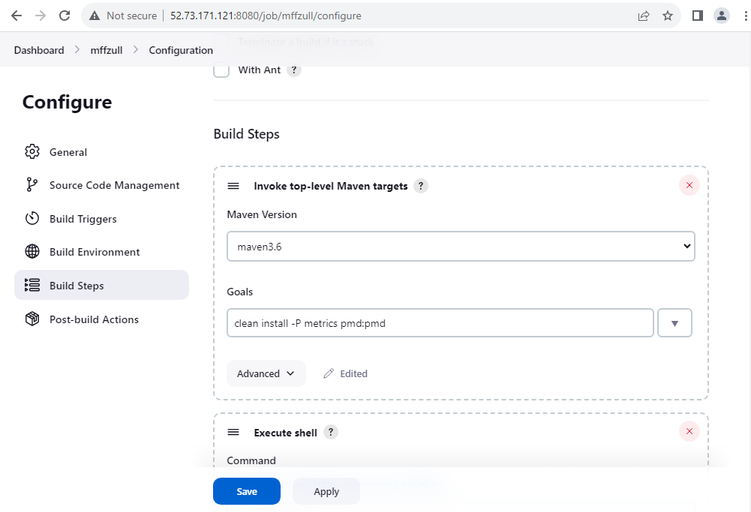
Jenkins is used for the continuous integration and deployment of the code. So Bitbucket, or GITHUB, is integrated into Jenkins. After this, Jenkins takes the source code and builds the artifact (a jar file) and then the Docker image. Jenkins also pushes the Docker images to DOCKERHUB.

Jenkins can be integrated with tools like SONARQUBE and JFROG. SONARQUBE is a tool used for quality testing of the code and generating the report. JFROG is a tool used for storing artifacts.

A screenshot of a computer

Description automatically generated

**Build image on Jenkins**

****

**A screenshot of a computer

Description automatically generated**

**Deploy images to Dockerhub**

**A screenshot of a computer

Description automatically generated**

**INTEGRATE DOCKERHUB TO ECS**

i) The Docker Hub repository URL is integrated with the AWS Fargate task definition.

**A screenshot of a computer

Description automatically generated**

ii)Task definition is integrated with the service inside the ECS cluster. This service creates a task with the IP address using which the application is accessible, and the IP address is redirected to the required domain name.

**A screenshot of a computer

Description automatically generated**

iii)ECS Service only takes care of the scaling and monitoring of the application according to the conditions provided.

scaling policies

**A screenshot of a search engine

Description automatically generated**

**Credentials:**

RDS and Docker Hub credentials are stored in AWS secrets manager.

Docker hub:

Username: odedtech

password: odedtech@69

Database:

username: postgres

password: oded@12