1. CONTAINERIZATION:

* Backend dockerfile-
* # build stage  
  FROM maven:3.8.7-openjdk-18 AS *build*WORKDIR /build  
  COPY /book-network/pom.xml .  
  RUN mvn dependency:go-offline  
  COPY /book-network/src ./src  
  RUN mvn clean package -DskipTests  
    
  # runtime stage  
  FROM amazoncorretto:17  
  #argument for passing values through env  
  ARG *PROFILE*=dev  
  ARG *APP\_VERSION*=1.0.0  
  WORKDIR /app  
  COPY --from=*build* /build/target/book-network-\*.jar /app/  
  EXPOSE 9000  
  ENV *DB\_URL*=jdbc:postgresql://postgres\_sql\_bsn:5432/book\_social\_network  
  ENV *ACTIVE\_PROFILE*=${*PROFILE*}  
  ENV *JAR\_VERSION*=${*APP\_VERSION*}  
  CMD java -jar -Dspring.profiles.active=${*ACTIVE\_PROFILE*} -Dspring.datasource.url=${*DB\_URL*} book-network-${*JAR\_VERSION*}.jar

The ‘bsn-api’ service is the image that is built from this dockerfile.

* Create ngnix.conf in the frontend root foler and put contents below(this is for angular, react may be slightly different)-

events {}

http {

server {

listen 80;

location / {

root /usr/share/nginx/html;

index index.html;

try\_files $uri $uri/ /index.html;

}

# Handle JavaScript | CSS files with the correct MIME type

location ~ \.(js|css)$ {

root /usr/share/nginx/html;

include /etc/nginx/mime.types;

}

# Catch-all location to handle refreshes on any route

location ~ ^/.+$ {

root /usr/share/nginx/html;

index index.html;

try\_files $uri /index.html;

}

}

}

* Frontend dockerfile-

FROM node:16 as build-stage

WORKDIR /app

COPY package\*.json ./

RUN npm install

COPY . .

RUN npm run build

FROM nginx:alpine

COPY nginx.conf /etc/nginx/nginx.conf

COPY --from=build-stage /app/dist/book-network-ui /usr/share/nginx/html

EXPOSE 80

PUT frontend image in the docker-compose which is ‘bsn-ui’

* Docker-compose-

services:

postgres:

container\_name: postgres-sql-bsn

image: postgres

environment:

POSTGRES\_USER: username

POSTGRES\_PASSWORD: password

PGDATA: /var/lib/postgresql/data

POSTGRES\_DB: book\_social\_network

volumes:

- postgres:/data/postgres

ports:

- 5432:5432

networks:

- spring-demo

restart: unless-stopped

mail-dev:

container\_name: mail-dev-bsn

image: maildev/maildev

ports:

- 1080:1080

- 1025:1025

bsn-api:

container\_name: bsn-api

image: bsn/bsn:1.0.8

ports:

- 8088:8088

networks:

- spring-demo

depends\_on:

- postgres

bsn-ui:

container\_name: bsn-ui

image: bsn/bsn-ui:1.0.0

ports:

- 8080:80

networks:

- spring-demo

depends\_on:

- bsn-api

networks:

spring-demo:

driver: bridge

volumes:

postgres:

driver: local

// ignore maildev service, we used gmail later

* Docker-compose up –d (run containers)