```yaml
version: "3.7"
services:
  bbw-frontend-dev:
    container_name: bbw-frontend-dev
    restart: always
    build:
      context: ./fapp
      dockerfile: Dockerfile
    volumes:
      - "./fapp:/app"
      - "/app/node_modules"
    ports:
      - 3001:3000
  bbw-frontend-dev-func:
    container_name: bbw-frontend-dev-fun
    restart: always
    build:
      context: ./shopping-cart-client-func-v1-props-Lift-State
      dockerfile: Dockerfile
    volumes:
      - "./shopping-cart-client-func-v1-props-Lift-State:/app"
      - "/app/node_modules"
    ports:
      - 3002:3000
  bbw-mariadb-dev:
    image: mariadb
    container_name: bbw-mariadb-dev
    environment:
      - MYSQL_ROOT_PASSWORD=bbw123
      - MYSQL_DATABASE=demo
    ports:
      - 3306:3306
    volumes:
      - ./database:/var/lib/mysql
  bbw-phpmyadmin:
    depends_on:
      - bbw-mariadb-dev
    image: phpmyadmin/phpmyadmin
    ports:
      - 80:80
    environment:
      PMA_HOST: bbw-mariadb-dev
```

```
bbw-backend-dev:
  container_name: bbw-backend-dev
  restart: always
  image: maven:3.5-jdk-8
  #build: ./app
  working_dir: /app
  volumes:
    - ./app:/app
    - ~/.m2:/root/.m2
  ports:
    - 8081:8080
  expose:
    - "8080"
  command: mvn clean spring-boot:run

  depends_on:
    - bbw-mariadb-dev
```

Dockerfile Frontend

```
# pull official base image
FROM node:13.12.0-alpine
# add a bash shell to the image
RUN apk add --no-cache bash
# set working directory
WORKDIR /app
# add `/app/node_modules/.bin` to $PATH
ENV PATH /app/node_modules/.bin:$PATH
RUN echo "Path: $PATH"
# install app dependencies
COPY package.json ./
COPY package-lock.json ./
RUN npm install --silent
RUN npm install react-scripts@3.4.1 -g --silent

# add All to app
COPY . ./
# ------ make a BUILD -- to copy in the nginx html
RUN npm run build
# start app
#CMD ["npm", "start"]
# Best-Practice with ENTRYPOINT!
ENTRYPOINT ["npm", "start"]

# Stage - Production
# ------
FROM nginx:1.17-alpine
COPY --from=build /app/build /usr/share/nginx/html
EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```

Dockerfile Backend

```
# A two Stage Dockerfile with a build stage and a run -stage!
FROM maven:3.5-jdk-8 as backend-build
WORKDIR ./
COPY pom.xml .
### Step 2 Copy the source and build the .jar
COPY src src
# we run first a build
RUN mvn install -DskipTests
```

```
### Stage 2 - Let's build a minimal image with the "deployable package"
FROM openjdk:8-jdk-alpine
# we add the jar to the working dir and execute it!
COPY  target/*.jar /app.jar
# Define an Entry-startpoint of the image
ENTRYPOINT ["sh", "-c", "java -jar /app.jar"]
```