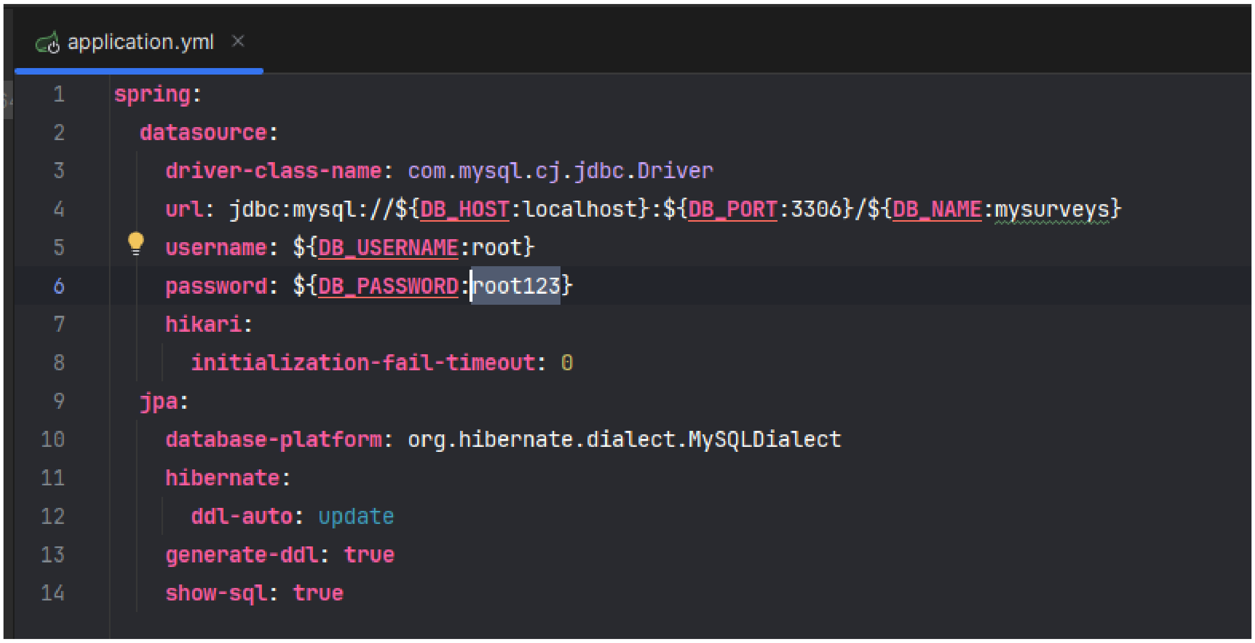
Extra Credit Documentation
SWE 642

By:
Abhishek Samuel Daniel - G01393582
Keerthan Srinivas - G01386121

The extra credit assignment is an extension to the 3rd assignment. We were given two options, one was to recreate the application using React and the other option was to host the application using Docker containers. I decided to go with the latter. I will give a detailed documentation of what I did for containerising the application.

Step 1: Rewriting the application.properties file.

- The first step was to replace the application properties file in the spring boot application with an application.yml file. It functions similarly to a properties file but is structured different.

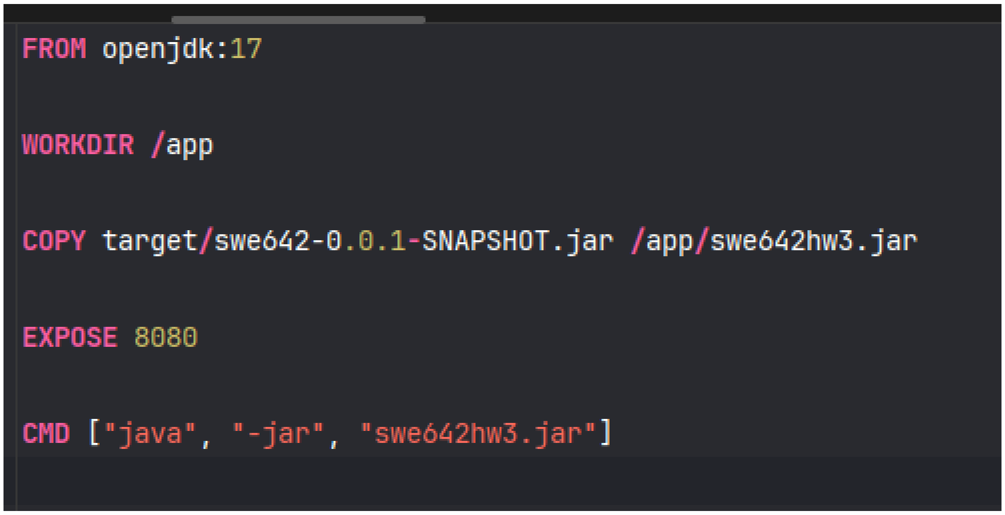


```yaml
spring:
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    url: jdbc:mysql://${DB_HOST:localhost}:${DB_PORT:3306}/${DB_NAME:mysurveys}
    username: ${DB_USERNAME:root}
    password: ${DB_PASSWORD:root123}
    hikari:
      initialization-fail-timeout: 0
  jpa:
    database-platform: org.hibernate.dialect.MySQLDialect
    hibernate:
      ddl-auto: update
    generate-ddl: true
    show-sql: true
```

- As you can see the structure is different.
- The url is slightly altered so as accommodate global variables, Similarly the credentials are also provided using global variables. In case the global variables are provided, I added an extra failsafe method so as to pass default parameters in the lack of global credentials.
- The global variables are passed using the dockerfile and the docker compose yaml file.

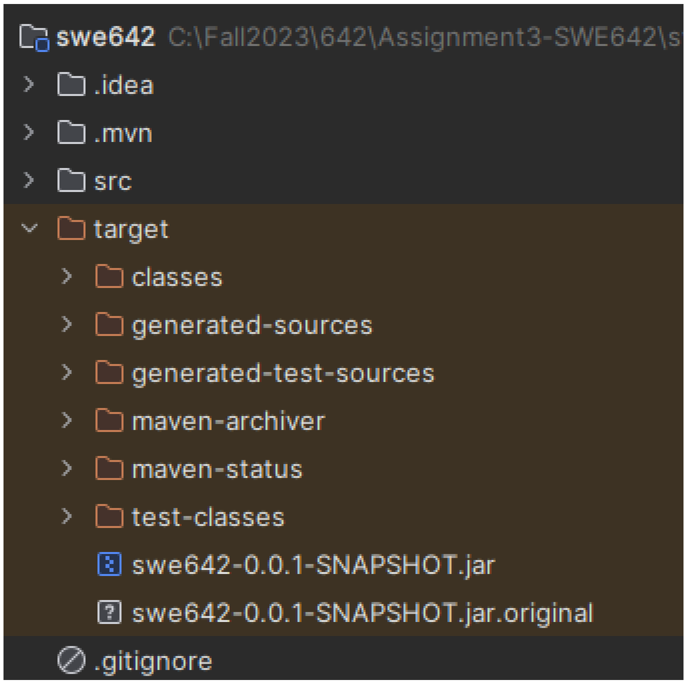
Step 2: Creating the dockerfile and docker compose file.

- First we need to create a dockerfile with the name Dockerfile.
- The contents of the dockerfile should look like this.



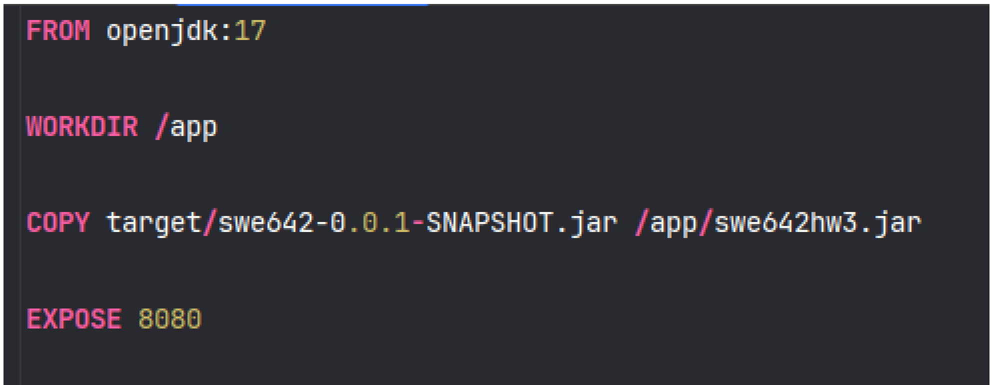
```
FROM openjdk:17

WORKDIR /app

COPY target/swe642-0.0.1-SNAPSHOT.jar /app/swe642hw3.jar

EXPOSE 8080

CMD ["java", "-jar", "swe642hw3.jar"]
```

- The first line uses JAVA 17 to build the app.
- The working directory is set to “/app”.
- The next step is to create the jar file, to do this we need to enter the following command.
    - mvn clean package
- This creates a jar file in the target folder of the spring boot application.

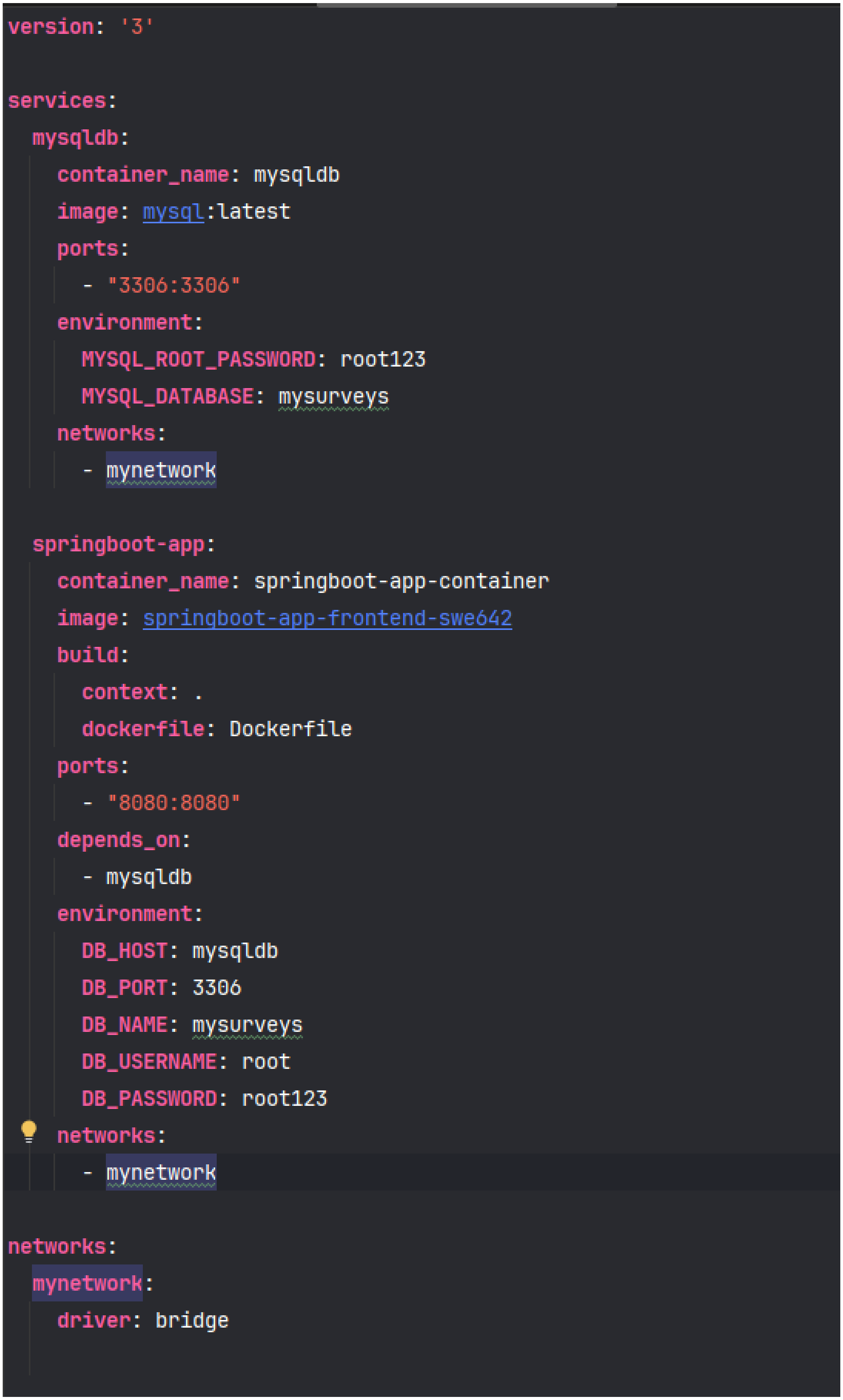


```
swe642 C:\Fall2023\642\Assignment3-SWE642\s
  .idea
  .mvn
  src
  target
    classes
    generated-sources
    generated-test-sources
    maven-archiver
    maven-status
    test-classes
    swe642-0.0.1-SNAPSHOT.jar
    swe642-0.0.1-SNAPSHOT.jar.original
  .gitignore
```

- The next step in the Dockerfile is to copy the previously created jar file to the app directory.
- The next step exposes the port 8080 for the application to run on.
- The next step is to create a docker-compose file. The docker compose file tells docker which images to run and what other parameters it should assign to the container.
- The dockerfile specifies which services to run, here it tells docker to create a service using the latest mysql image.
- The name of the container should be mysqldb.
- The global variables I mentioned before are derived from here.
- The container should run on the “mynetwork” network.
- A picture of the docker compose file is shown below
- The next service should use the latest image of the springboot backend application and must name to container “springboot-app-container”.
- It should use the Dockerfile specified for the spring boot application.
- And the global variables that are to be passed to the application.yml file are also specified here.
- This container should also run on the “mynetwork” network.
- The last part creates a network with the name “mynetwork” in a bridge mode.



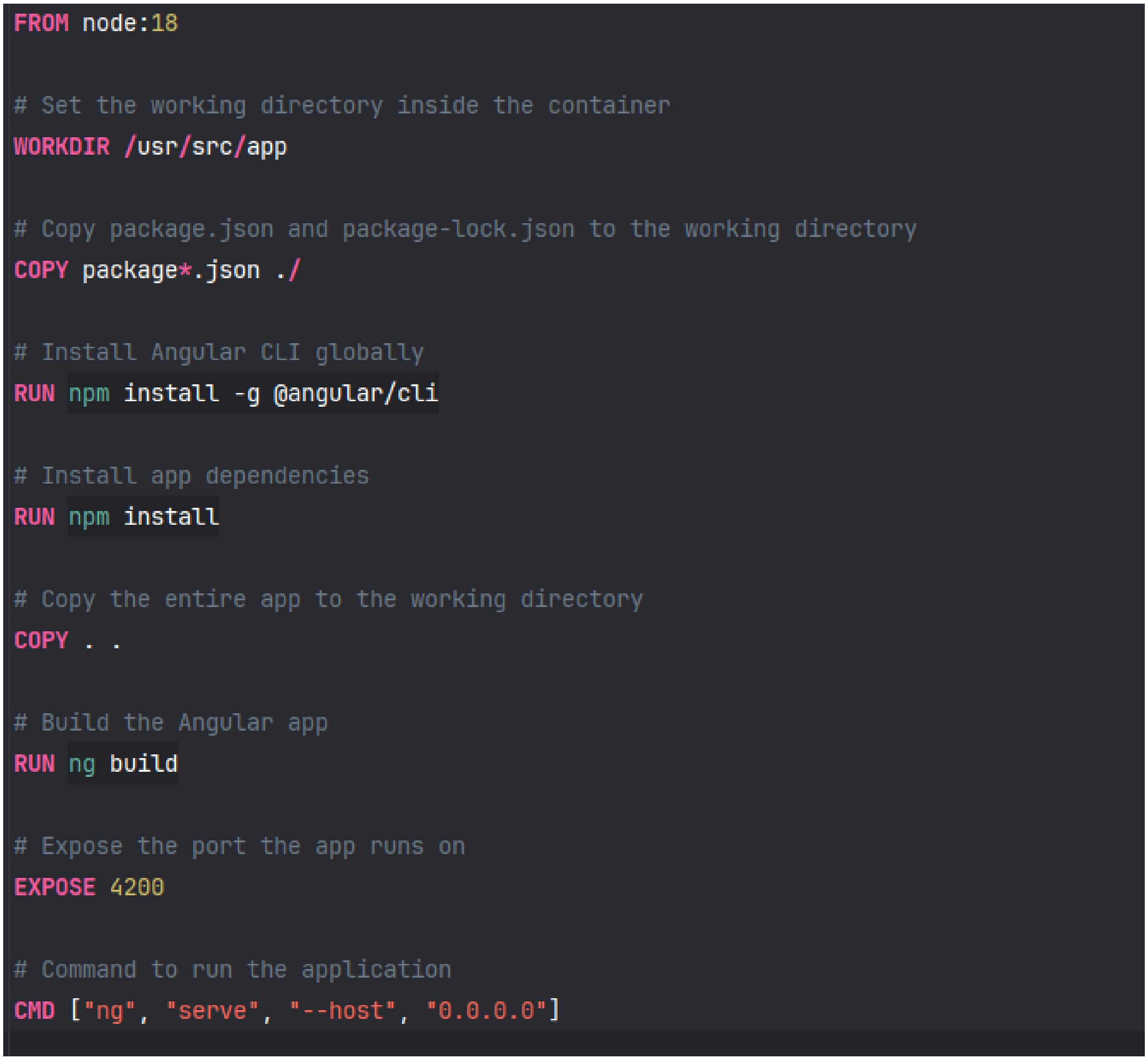
```
FROM openjdk:17

WORKDIR /app

COPY target/swe642-0.0.1-SNAPSHOT.jar /app/swe642hw3.jar

EXPOSE 8080
```



```yaml
version: '3'

services:
  mysqldb:
    container_name: mysqldb
    image: mysql:latest
    ports:
      - "3306:3306"
    environment:
      MYSQL_ROOT_PASSWORD: root123
      MYSQL_DATABASE: mysurveys
    networks:
      - mynetwork

  springboot-app:
    container_name: springboot-app-container
    image: springboot-app-frontend-swe642
    build:
      context: .
      dockerfile: Dockerfile
    ports:
      - "8080:8080"
    depends_on:
      - mysqldb
    environment:
      DB_HOST: mysqldb
      DB_PORT: 3306
      DB_NAME: mysurveys
      DB_USERNAME: root
      DB_PASSWORD: root123
    networks:
      - mynetwork

networks:
  mynetwork:
    driver: bridge
```

Step 3: Building the docker images.

- The commands for containerising the service is:
  - docker compose up
- The docker file for the angular app looks like this

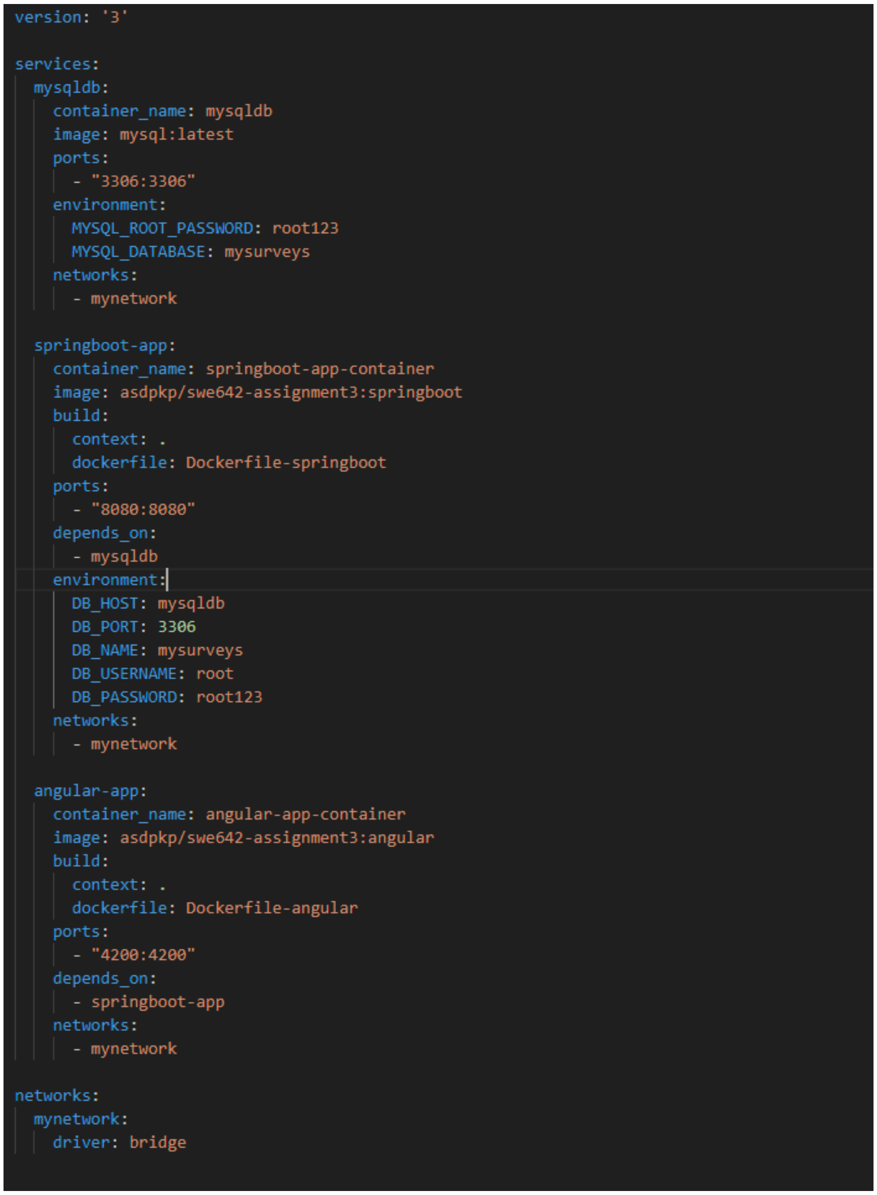


```
FROM node:18

# Set the working directory inside the container
WORKDIR /usr/src/app

# Copy package.json and package-lock.json to the working directory
COPY package*.json ./

# Install Angular CLI globally
RUN npm install -g @angular/cli

# Install app dependencies
RUN npm install

# Copy the entire app to the working directory
COPY . .

# Build the Angular app
RUN ng build

# Expose the port the app runs on
EXPOSE 4200

# Command to run the application
CMD ["ng", "serve", "--host", "0.0.0.0"]
```

- The commands for running these commands are
  - docker compose up -f <path-to spring-boot-docker-compose-app>
  - docker compose up -f <path-to-angular-app-docker-compose>
- The above will automatically build and run the docker images and container

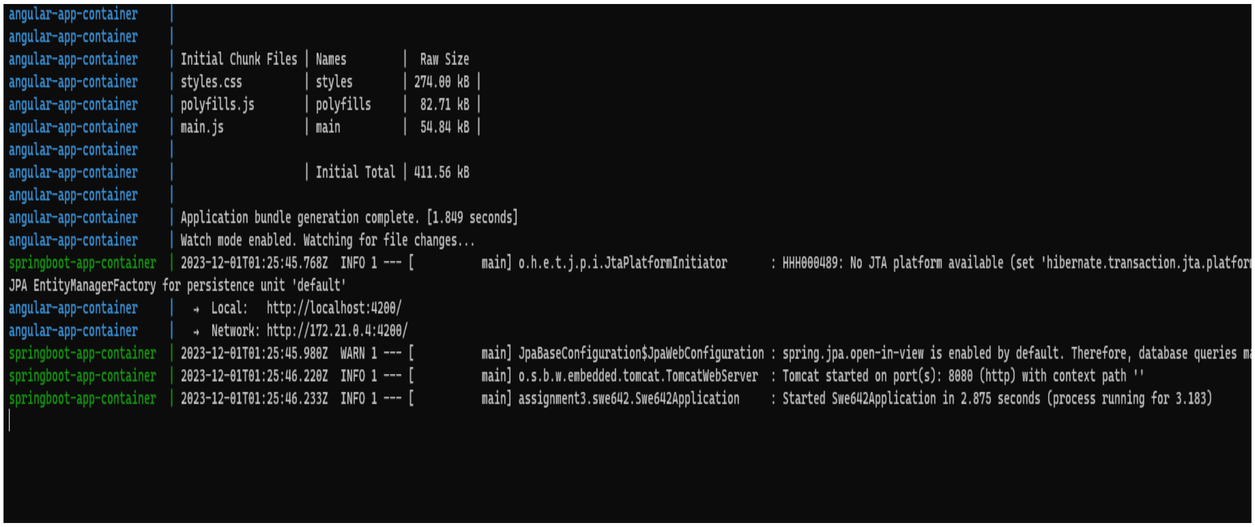
Step 4: Source file independent run.

- We can run the containerized applications without the source code and dockerfiles once they have been pushed to our docker hub. The commands for this are:
    - docker build -t asdpkp/swe642-assignment3:springboot
    - docker build -t asdpkp/swe642-assignemnt3:angular
    - docker login
    - docker push asdpkp/swe642-assignment3:springboot
    - docker push asdpkp/swe642-assignemnt3:angular
    - docker compose -up
- We must first create a new docker-compose.yml file independent of the other files. The docker-compose file should look like this.

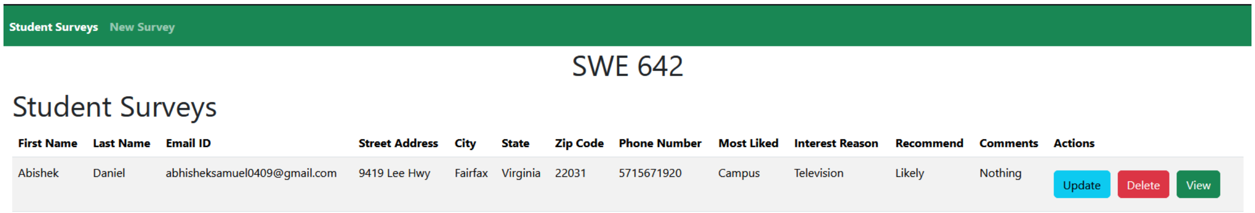


```
version: '3'

services:
  mysqldb:
    container_name: mysqldb
    image: mysql:latest
    ports:
      - "3306:3306"
    environment:
      MYSQL_ROOT_PASSWORD: root123
      MYSQL_DATABASE: mysurveys
    networks:
      - mynetwork

  springboot-app:
    container_name: springboot-app-container
    image: asdpkp/swe642-assignment3:springboot
    build:
      context: .
      dockerfile: Dockerfile-springboot
    ports:
      - "8080:8080"
    depends_on:
      - mysqldb
    environment:
      DB_HOST: mysqldb
      DB_PORT: 3306
      DB_NAME: mysurveys
      DB_USERNAME: root
      DB_PASSWORD: root123
    networks:
      - mynetwork

  angular-app:
    container_name: angular-app-container
    image: asdpkp/swe642-assignment3:angular
    build:
      context: .
      dockerfile: Dockerfile-angular
    ports:
      - "4200:4200"
    depends_on:
      - springboot-app
    networks:
      - mynetwork

networks:
  mynetwork:
    driver: bridge
```

Output Screenshots:



```
angular-app-container     |
angular-app-container     |
angular-app-container     | Initial Chunk Files | Names         |  Raw Size
angular-app-container     | styles.css          | styles        | 274.00 kB |
angular-app-container     | polyfills.js        | polyfills     |  82.71 kB |
angular-app-container     | main.js             | main          |  54.84 kB |
angular-app-container     |
angular-app-container     |                     | Initial Total | 411.56 kB
angular-app-container     |
angular-app-container     | Application bundle generation complete. [1.849 seconds]
angular-app-container     | Watch mode enabled. Watching for file changes...
springboot-app-container  | 2023-12-01T01:25:45.768Z  INFO 1 --- [           main] o.h.e.t.j.p.i.JtaPlatformInitiator       : HHH000489: No JTA platform available (set 'hibernate.transaction.jta.platfor
JPA EntityManagerFactory for persistence unit 'default'
angular-app-container     |   → Local:   http://localhost:4200/
angular-app-container     |   → Network: http://172.21.0.4:4200/
springboot-app-container  | 2023-12-01T01:25:45.980Z  WARN 1 --- [           main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries m
springboot-app-container  | 2023-12-01T01:25:46.220Z  INFO 1 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8080 (http) with context path ''
springboot-app-container  | 2023-12-01T01:25:46.233Z  INFO 1 --- [           main] assignment3.swe642.Swe642Application     : Started Swe642Application in 2.875 seconds (process running for 3.183)
```



SWE 642

Student Surveys

First Name	Last Name	Email ID	Street Address	City	State	Zip Code	Phone Number	Most Liked	Interest Reason	Recommend	Comments	Actions
Abishek	Daniel	abhisheksamuel0409@gmail.com	9419 Lee Hwy	Fairfax	Virginia	22031	5715671920	Campus	Television	Likely	Nothing	Update Delete View

Github: https://github.com/ASDPKP/SWE642-Assignment3-ExtraCredit.git