

## DevOps

### 1. Infrastructure Architecture Diagram

#### Add image

1.1.proxy : <http://104.215.186.188/>

1.2.containers :

CONTAINER ID	IMAGE
e99b820a8b52	frontend
299c8c0dc65f	nginx
0/tcp	
fef2cdf581dc	backend
d35d0379ba9c	database

1.3.IP addresses (host, containers) :

```
"Containers": {
  "299c8c0dc65fbe9b086b07b91261fd7b0eea592fe09870946666d724b0629a77": {
    "Name": "int221integratedprojectproxy_reverseproxy_1",
    "EndpointID": "4b0b84e78cb0b30341f04d45b889078518f5477c023b12c225e92946250e04a7",
    "MacAddress": "02:42:ac:14:00:05",
    "IPv4Address": "172.20.0.5/16",
    "IPv6Address": ""
  },
  "d35d0379ba9ca2f3fa5165bcf2fc7b1845daa75adc827b53134f96dfa4f3cf9e": {
    "Name": "database",
    "EndpointID": "b5bf1a426832e026391f88151ea9ea80025e1ca34392c64c4a0520e3bc6f4490",
    "MacAddress": "02:42:ac:14:00:02",
    "IPv4Address": "172.20.0.2/16",
    "IPv6Address": ""
  },
  "e99b820a8b526d70e479623736f193c6a06fca83ce005b547fc569157cc04bb5": {
    "Name": "frontend",
    "EndpointID": "6fbd71c49ba72efb85dfa7a1a8aae6ba5c7c6779558f9ad9c6406931d4344b5d",
    "MacAddress": "02:42:ac:14:00:03",
    "IPv4Address": "172.20.0.3/16",
    "IPv6Address": ""
  },
  "fef2cdf581dc34056f0710567b5f6f9ba2cca527bf0bb1c62bd12ac2286d7eb9": {
    "Name": "backend",
    "EndpointID": "cf49c4f748411c4a95e55aeddff6433b8dcb1fcd7b006581525f616f1409c15c1",
    "MacAddress": "02:42:ac:14:00:04",
    "IPv4Address": "172.20.0.4/16",
    "IPv6Address": ""
  }
}
```

1.4.open ports (host, containers) :

```
PORTS

80/tcp

0.0.0.0:80->80/tcp, :::80

3000/tcp

0.0.0.0:3306->3306/tcp,
```

1.5. App URL (at proxy) :

<http://www.int221projectcars.games/> @ <http://104.215.186.188/>

2. List of Docker images that you used with versions/tags

REPOSITORY	TAG
frontend	latest
database	latest
backend	latest
node	latest
openjdk	11.0-slim
mysql	latest
node	14.16.1-alpine3.10
nginx	1.19.10-alpine
nginx	latest
maven	3.6.1-jdk-11-slim

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
frontend	latest	8085921948a8	56 minutes ago	152MB
database	latest	6c4efbcffb0c	8 hours ago	556MB
backend	latest	ff374474b0a8	39 hours ago	464MB
phpmyadmin	latest	5357004ce1de	7 days ago	477MB
node	latest	6817534de6bd	12 days ago	907MB
openjdk	11.0-slim	1a70900cdaaa	2 weeks ago	421MB
mysql	latest	0627ec6901db	2 weeks ago	556MB
node	14.16.1-alpine3.10	dad9ba894bb5	3 weeks ago	116MB
nginx	1.19.10-alpine	a64a6e03b055	3 weeks ago	22.6MB
nginx	latest	62d49f9bab67	3 weeks ago	133MB
mysql	8.0.23	cbe8815cbea8	4 weeks ago	546MB
caddy	2.0.0-alpine	313ad00e425a	10 months ago	40.3MB
maven	3.6.1-jdk-11-slim	b67032e30f89	21 months ago	418MB
maven	3.6.0-jdk-11-slim	c7428be691f8	2 years ago	489MB

### 3. Configuration/script files with explanations where required

#### 3.1. Dockerfile(s)

##### Frontend

```
1 FROM node:latest as build-stage
2 WORKDIR /app
3 COPY package*.json ./
4 RUN npm install
5 COPY ./ .
6 RUN npm run build
7
8 FROM nginx as production-stage
9 RUN mkdir /app
10 COPY --from=build-stage /app/dist /app
11 COPY nginx.conf /etc/nginx/nginx.conf
12 EXPOSE 80
```

##### Backend

```
1 FROM maven:3.6.1-jdk-11-slim AS build
2 COPY src /workspace/src
3 COPY pom.xml /workspace
4 WORKDIR /workspace
5 RUN mvn clean install
6
7 FROM openjdk:11.0-slim
8 EXPOSE 3000
9 COPY --from=build /workspace/target/*.jar app.jar
10 ENTRYPOINT ["java", "-jar", "app.jar"]
```

##### Database

```
1 FROM mysql
2 COPY ./scripts ./scripts
3 ENV MYSQL_ROOT_PASSWORD=Newnismo_2001
4 EXPOSE 3306
```

### 3.2. Docker compose file

```
1 version: "3"
2 services:
3   frontend:
4     container_name: frontend
5     build:
6       context: .
7     image: frontend
8
9   networks:
10    default:
11      external:
12        name: allnetwork
```

Frontend :

```
1 version: "3"
2 services:
3   backend:
4     container_name: backend
5     build: .
6     image: backend
7     environment:
8       - MYSQL_DATABASE=carsdb
9       - MYSQL_USER=root
10      - MYSQL_ROOT_PASSWORD=Newnismo_2001
11     volumes:
12       - ~/backend_data:/public/product-img
13
14   networks:
15    default:
16      external:
17        name: allnetwork
18
```

Backend :

Database :

```
1 version: '3'
2 services:
3   database:
4     container_name: database
5     build: .
6     image: database
7     environment:
8       MYSQL_ROOT_USER: root
9       MYSQL_ROOT_PASSWORD: Newnismo_2001
10      MYSQL_DATABASE: carsdb
11     volumes:
12       - "./scripts/carsdb.sql:/docker-entrypoint-initdb.d/carsdb.sql"
13       - "./scripts/user_grant.sql:/docker-entrypoint-initdb.d/user_grant.sql"
14     ports:
15       - "3306:3306"
16
17   networks:
18    default:
19      external:
20        name: allnetwork
```

### 3.3. Proxy configuration file

Docker-compose.yml

```
1  version: '3'
2  services:
3    reverseproxy:
4      container_name: reverseproxy
5      image: nginx
6      volumes:
7        - ./nginx.conf:/etc/nginx/conf.d/default.conf
8      ports:
9        - "80:80"
10   networks:
11     default:
12       external:
13         name: allnetwork
```

Nginx.conf

```
1  server{
2    listen 80;
3    location / {
4      proxy_pass http://172.20.0.3;
5    }
6    location /backend {
7      proxy_pass http://172.20.0.4:3000/;
8    }
9  }
```

3.4. other files that you use with explanation of what you did

#### .dockerignore(frontend)

```
.dockerignore
1  **/node_modules
2  **/dist
```

Setting up the .dockerignore file prevents node\_modules and any intermediate build artifacts from being copied to the image which can cause issues during building.

#### .env(frontend)

```
1  VUE_APP_ROOT_API=http://104.215.186.188/backend/
```

Set api to backend

#### Nginx.conf(frontend)

```
1  user  nginx;
2  worker_processes  1;
3  error_log  /var/log/nginx/error.log warn;
4  pid        /var/run/nginx.pid;
5  events {
6      worker_connections  1024;
7  }
8  http {
9      include        /etc/nginx/mime.types;
10     default_type    application/octet-stream;
11     log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
12                       '$status $body_bytes_sent "$http_referer" '
13                       '"$http_user_agent" "$http_x_forwarded_for"';
14     access_log  /var/log/nginx/access.log  main;
15     sendfile    on;
16     keepalive_timeout  65;
17     server {
18         listen      80;
19         server_name  localhost;
20         location / {
21             root    /app;
```

```

22     index index.html;
23     try_files $uri $uri/ /index.html;
24 }
25 error_page 500 502 503 504 /50x.html;
26 location = /50x.html {
27     root /usr/share/nginx/html;
28 }
29 }
30 }

```

Nginx is an HTTP(s) server that will run in your docker container. It uses a configuration file to determine how to serve content/which ports to listen on/etc. See the [nginx configuration documentation](#) for an example of all of the possible configuration options.

The following is a simple nginx configuration that serves your vue project on port 80. The root index.html is served for page not found / 404 errors which allows us to use `pushState()` based routing.

### Application.properties(backend)

```

1  server.port=3000
2  spring.datasource.url=jdbc:mysql://172.20.0.2:3306/carsdb
3  spring.datasource.username=root
4  spring.datasource.password=Newnismo_2001
5  spring.datasource.platform=mysql
6  spring.datasource.driver-class-name=com.mysql.jdbc.Driver
7  spring.datasource.initialization-mode=always
8  spring.jpa.database-platform=org.hibernate.dialect.MySQL5InnoDBDialect
9
10 spring.servlet.multipart.enabled=true
11 spring.servlet.multipart.file-size-threshold=2KB
12 spring.servlet.multipart.max-file-size=200MB
13 spring.servlet.multipart.max-request-size=215MB
14
15 cars.storage.location=./public/product-img
16 cars.origin.host=http://104.215.186.188:8080,http://104.215.186.188,http://www.int221projectcars.games
17 cars.origin.method=GET,POST,PUT,HEAD,DELETE,PATCH,OPTIONS
18

```

Use to set origin host from frontend and lock inner IP of MySQL database

#### 4. Describe configurations that you did to set the environment for FE, BE, proxy

##### ❖ Frontend

For Dockerfile we use the node image version latest and nginx image for create the image name's "frontend" and copy the environment from frontend source code to build the image. Use port 80 for container port.

For docker-compose.yml we build container name "frontend" with using image name's "frontend" and docker network name's "allnetwork".

##### ❖ Backend

For Dockerfile we use maven image for mvn clean install and use openjdk for build image with .jar file that we got from mvn clean install. The image name's "backend". Use port 3000 for container port.

For docker-compose.yml we build container name "backend" with using image name's "backend" and build the environment with database connection. We kept the product-picture in the backend\_data file.

And finally using docker network name's "allnetwork".

Application.properties we set server port for 3000 and set the database connect to inner IP network connection with platform mysql.

##### ❖ Database

For Dockerfile we use mysql image name's "database" with container port 3306 and set mysql root password.

For docker-compose.yml we build container name's "database" with using image name's "database" and set database connection. We run scripts from the scripts file and set server port to 3306.

##### ❖ Proxy

For docker-compose.yml we build the container name's "reverseproxy" with using image nginx and kept nginx.conf to default.conf in root. We use port 80 for server and port 80 for container.

For nginx.conf file use server port 80 go to proxy pass to home page website with the path "/" and proxy pass to backend page with path "/backend".