

Homework 1

Student: Amal Yakubov

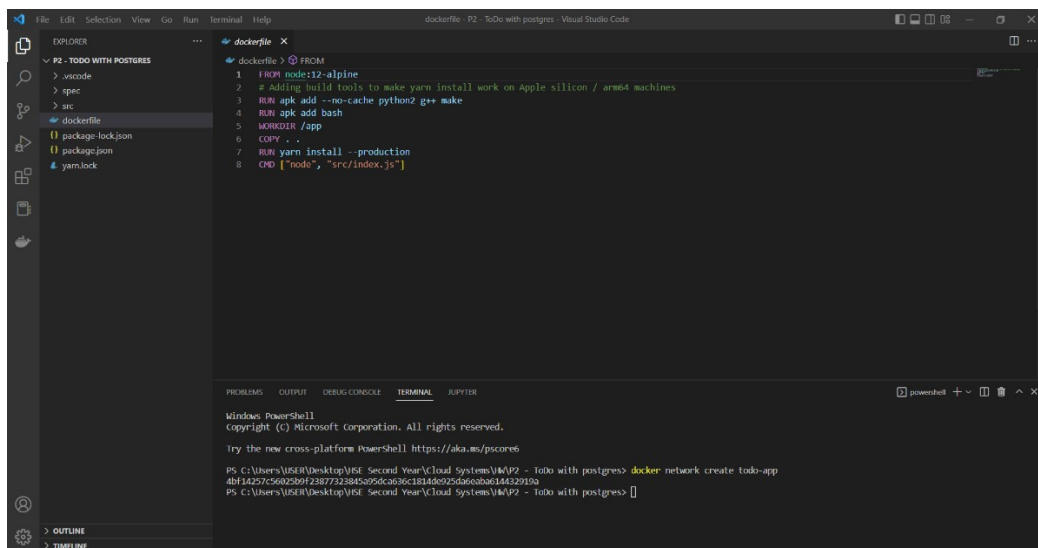
Programme: Business analytics and big data systems

Github link: <https://github.com/Yakuboff-1-33/cloud-systems-HW1>

Part 7:

1. Starting MySQL:

1.1 Creating the network.

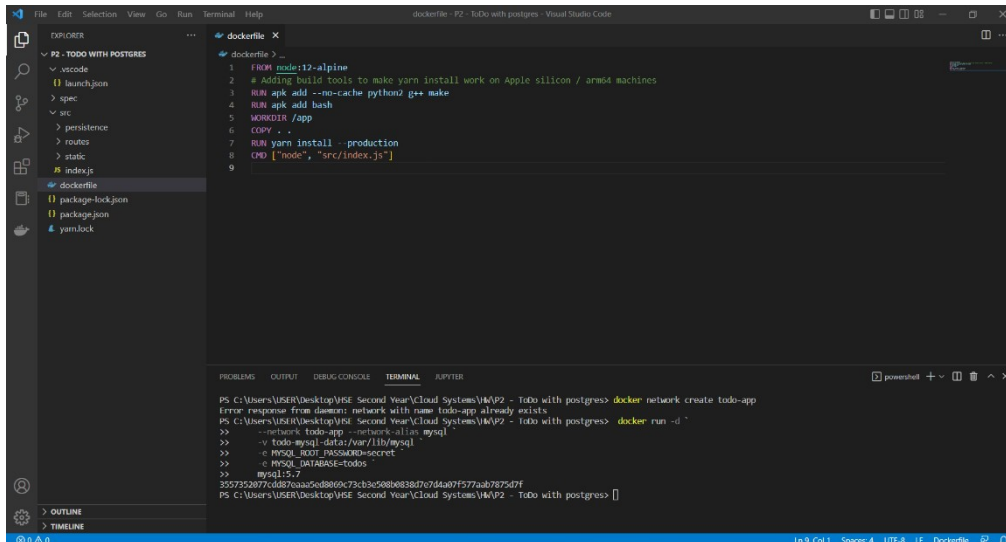


The screenshot shows the Visual Studio Code interface. The Explorer panel on the left shows a project named 'P2 - TODO WITH POSTGRES' with files: '.vscode', 'spec', 'src', 'dockerfile', 'package-lock.json', 'package.json', and 'yarn.lock'. The Dockerfile is open in the editor, showing the following content:

```
dockerfile X
1 FROM node:12-alpine
2 # Adding build tools to make yarn install work on Apple silicon / arm64 machines
3 RUN apk add --no-cache python2 g++ make
4 RUN apk add bash
5 WORKDIR /app
6 COPY . .
7 RUN yarn install --production
8 CMD ["node", "src/index.js"]
```

The Terminal panel at the bottom shows a Windows PowerShell session. The prompt is 'PS C:\Users\USER\Desktop\VE Second Year\Cloud Systems\VM\P2 - ToDo with postgres>'. The user has entered the command 'docker network create todo-app' and the output is '4bf14257c56825b9f23877323845a95dc636c1814d6925d6e0ba614432910a'. The prompt is now 'PS C:\Users\USER\Desktop\VE Second Year\Cloud Systems\VM\P2 - ToDo with postgres> '.

1.2 Starting a MySQL container and attach it to the network.



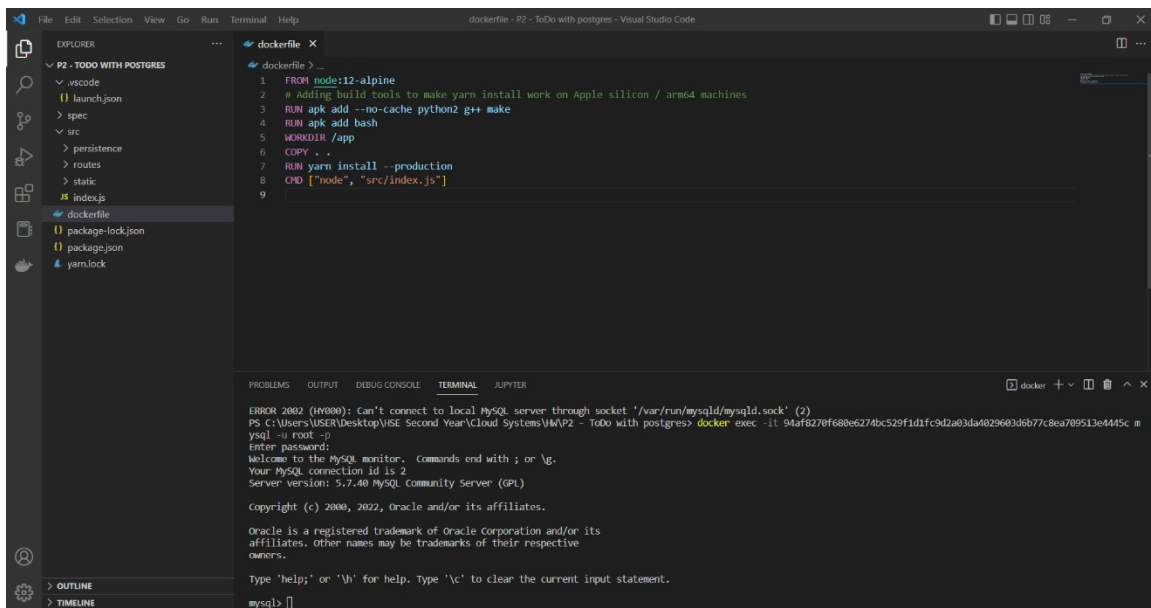
The screenshot shows a Visual Studio Code editor with a Dockerfile in the center pane and a terminal at the bottom. The Dockerfile contains the following code:

```
1 FROM node:12-alpine
2 # Adding build tools to make yarn install work on Apple silicon / arm64 machines
3 RUN apk add --no-cache python2 g++ make
4 RUN apk add bash
5 WORKDIR /app
6 COPY . .
7 RUN yarn install --production
8 CMD ["node", "src/index.js"]
9
```

The terminal shows the following commands and output:

```
PS C:\Users\USER\Desktop\USE Second Year\Cloud Systems\VM\P2 - Todo with postgres> docker network create todo-app
Error response from daemon: network with name todo-app already exists
PS C:\Users\USER\Desktop\USE Second Year\Cloud Systems\VM\P2 - Todo with postgres> docker run -d --network todo-app --network-alias mysql --v todo-mysql-data:/var/lib/mysql --e MYSQL_ROOT_PASSWORD=secret --e MYSQL_DATABASE=todos mysql:5.7
350725a077cdd70aa5e0806c72cb2e08b083d7c744a0f577ab7875d2f
PS C:\Users\USER\Desktop\USE Second Year\Cloud Systems\VM\P2 - Todo with postgres>
```

1.3 confirming that I have the database up and it is running.



The screenshot shows the same Visual Studio Code editor with the terminal output for connecting to the MySQL database:

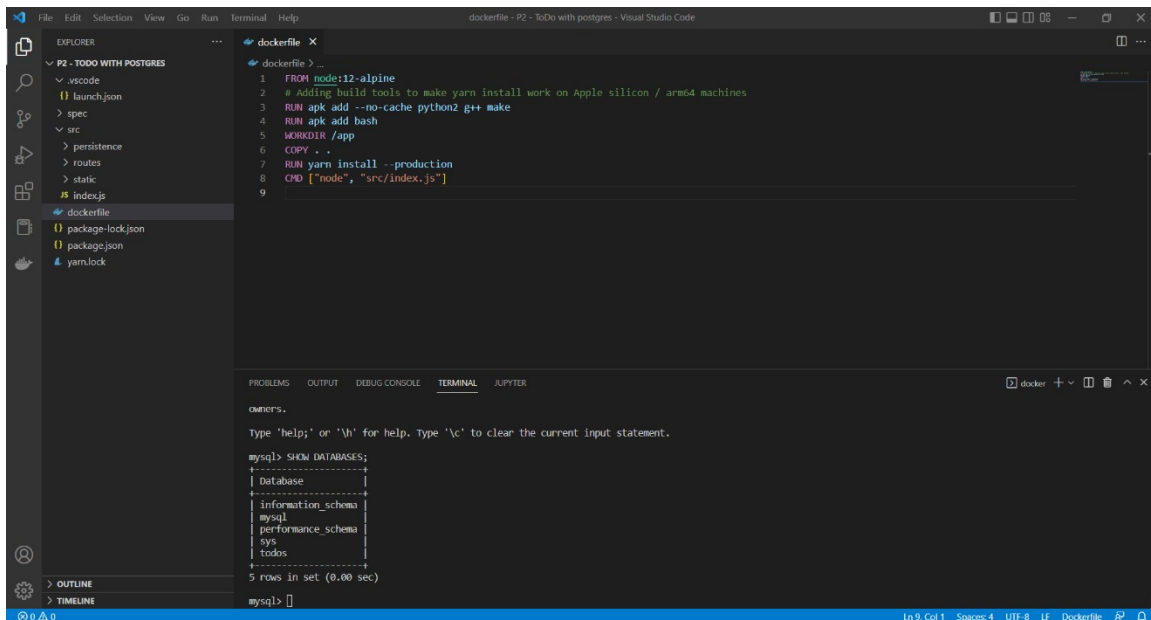
```
PS C:\Users\USER\Desktop\USE Second Year\Cloud Systems\VM\P2 - Todo with postgres> docker exec -it 94af8270f680e6274bc529f1d1fc942a03da4029603dbb77c8ea709513e445c mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.7.40 MySQL Community Server (GPL)

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```



The screenshot shows the Visual Studio Code interface. The Explorer panel on the left displays the file structure of a project named 'P2 - TODO WITH POSTGRES'. The Dockerfile is open in the editor, showing the following content:

```
1 FROM node:12-alpine
2 # Adding build tools to make yarn install work on Apple silicon / arm64 machines
3 RUN apk add --no-cache python2 g++ make
4 RUN apk add bash
5 WORKDIR /app
6 COPY . .
7 RUN yarn install --production
8 CMD ["node", "src/index.js"]
9
```

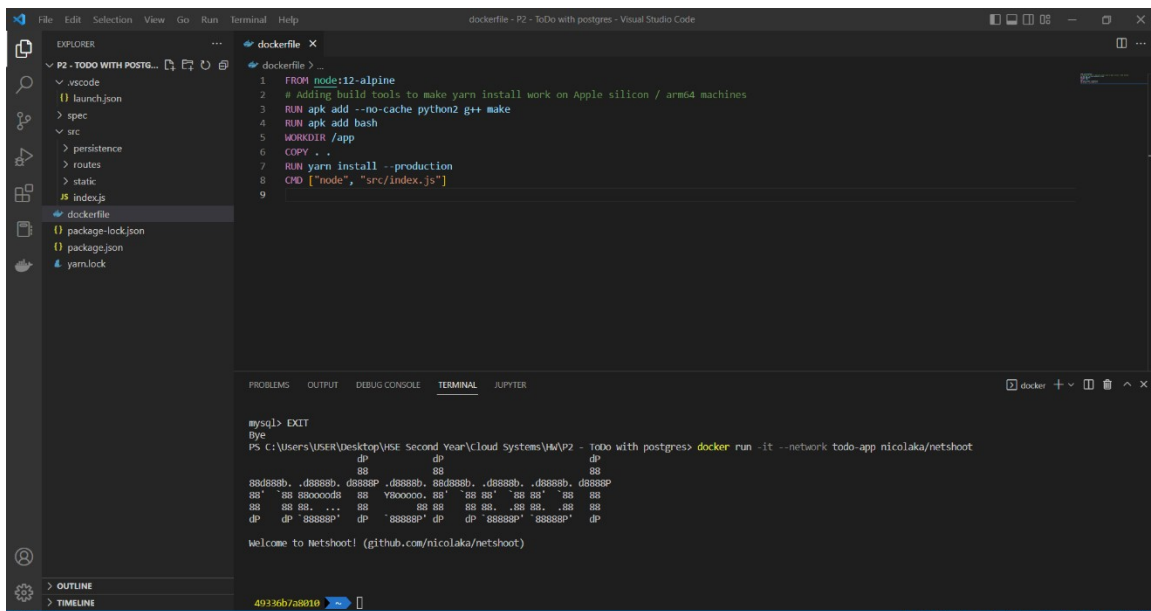
The Terminal panel at the bottom shows the output of a MySQL command:

```
mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| todos |
+-----+
5 rows in set (0.00 sec)

mysql>
```

2. Connecting to MySQL:

2.1 Starting a new container using the nicolaka/netshoot image.



The screenshot shows the Visual Studio Code interface. The Dockerfile is open in the editor, showing the same content as the previous screenshot. The Terminal panel at the bottom shows the output of a Docker command:

```
mysql> EXIT
bye
PS C:\Users\USER\Desktop\ISE Second Year\Cloud Systems\W4\P2 - Todo with postgres> docker run -it --network todo-app nicolaka/netshoot
dp dp dp
888888b. .d8888b. d8888P .d8888b. 88d888b. .d8888b. .d8888b. d8888P
88 88 880000d8 88 Y000000. 88 88 88 88 88 88 88
88 88 88 . . . 88 88 88 88 88 88 88 88
dP dP 88888P' dP 88888P' dP dP 88888P' 88888P' dP

Welcome to netshoot! (github.com/nicolaka/netshoot)

49136b7a8010
```

2.2 using dig command.

The screenshot shows the Visual Studio Code interface with a Dockerfile open in the editor. The Dockerfile contains the following instructions:

```
1 FROM node:12-alpine
2 # Adding build tools to make yarn install work on Apple silicon / arm64 machines
3 RUN apk add --no-cache python2 g++ make
4 RUN apk add bash
5 WORKDIR /app
6 COPY .
7 RUN yarn install --production
8 CMD ["node", "src/index.js"]
9
```

The terminal at the bottom shows the output of the command `dig mysql`:

```
49336b7a0010 dig mysql
; <<> DiG 0.18.3 <<> mysql
;; global options: +cmd
;; Got answer:
;;->HEADER<<< opcode: QUERY, status: NOERROR, id: 28870
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0
;; QUESTION SECTION:
;mysql.
;; ANSWER SECTION:
mysql. 600 IN A 172.18.0.3
```

3. Running my app with MySQL:

3.1 specifying each of the environment variables, connecting the container to my app network.

The screenshot shows the Visual Studio Code interface with the same Dockerfile as before. The terminal at the bottom shows the output of the command `docker run -v $(pwd):/app --network todo-app --env MYSQL_HOST=mysql --env MYSQL_USER=root --env MYSQL_PASSWORD=secret --env MYSQL_DB=todos node:12-alpine sh -c "yarn install && yarn run dev"`:

```
>> -w /app -v $(pwd):/app -
>> --network todo-app
>> -e MYSQL_HOST=mysql -
>> -e MYSQL_USER=root -
>> -e MYSQL_PASSWORD=secret -
>> -e MYSQL_DB=todos -
>> node:12-alpine -
>> sh -c "yarn install && yarn run dev"
unable to find image 'node:12-alpine' locally
12-alpine: pulling from library/node
df6e9388f04a: Already exists
3bf6d7380295: Already exists
79396601ee9e: Already exists
31fe6bdc071: Already exists
Digest: sha256:ddb353d48f42059a15de59be6a9fe21762aae9d6cbea6f124440895c27db68
Status: Downloaded newer image for node:12-alpine
85bfba7722e17d6175581e1fa5958a08b9f6f4176d6d6b14ec964cb364df02
PS C:\Users\USER\Desktop\ISE - Second Year\Cloud Systems\4\4\2 - Todo with postgres>
```

3.2 Connecting to the mysql database and proving that the items are being written to the database, after writing one row (shopping).

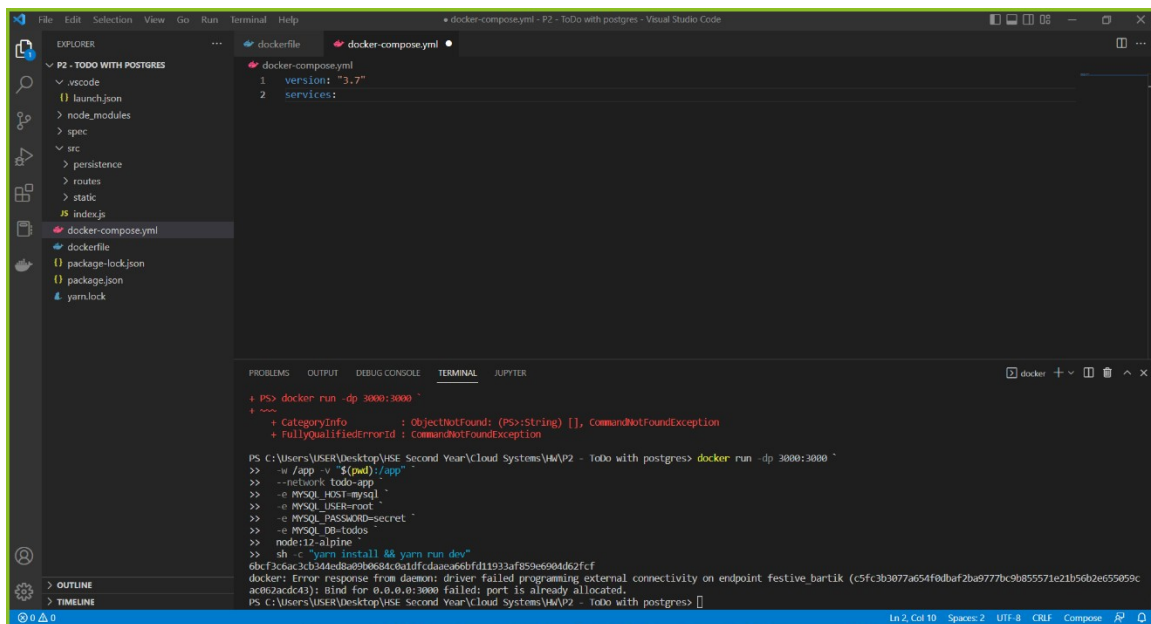
id	name	completed
3679eefa-7b4c-4f4e-ab75-1c98c1658829	shopping	0

1 row in set (0.00 sec)

mysql> []

Part 8:

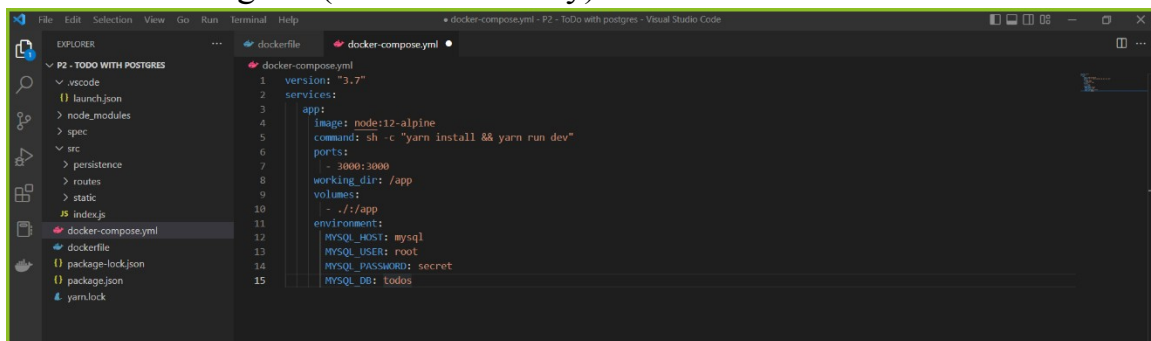
1. Defining the app service after creating the compose file and installing Docker Compose:



```
docker-compose.yml
1 version: "3.7"
2 services:
```

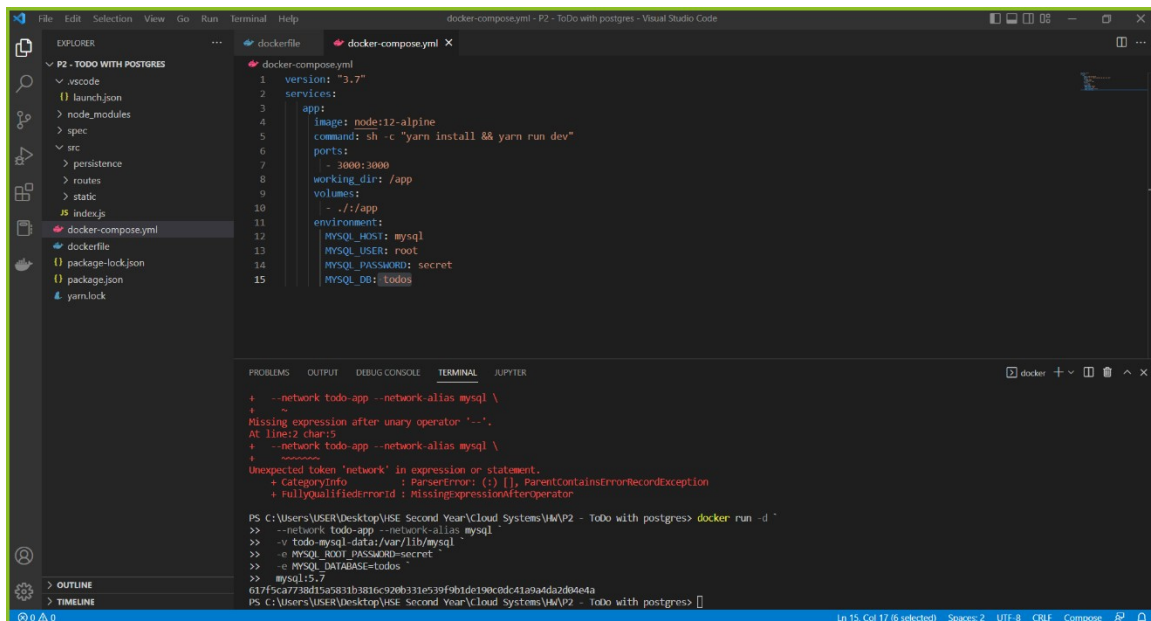
```
PS C:\Users\USER\Desktop\ISE - Second Year\Cloud Systems\44\I2 - ToDo with postgres> docker run -dp 3000:3000
PS C:\Users\USER\Desktop\ISE - Second Year\Cloud Systems\44\I2 - ToDo with postgres> docker run -dp 3000:3000
>> --network todo-app
>> --env MYSQL_HOST=mysql
>> --env MYSQL_USER=root
>> --env MYSQL_PASSWORD=secret
>> --env MYSQL_DB=todos
>> node:12-alpine
>> sh -c "yarn install && yarn run dev"
68cf368ac3db344ed8a9b0684ca0ffcdacae6bf011933af859e9804d62cf
docker: Error response from daemon: driver failed programming external connectivity on endpoint festive_bartik (c5fc3b077a654f0dbaf2ba9777bc9b855571e21b56b2e655959c
ac862acdc43): bind for 0.0.0.0:3000 failed: port is already allocated.
PS C:\Users\USER\Desktop\ISE - Second Year\Cloud Systems\44\I2 - ToDo with postgres>
```

2. defining the service entry and the image for the container and migrating both the working directory, also migrating the environment variable definitions using the (environment key).



```
docker-compose.yml
1 version: "3.7"
2 services:
3   app:
4     image: node:12-alpine
5     command: sh -c "yarn install && yarn run dev"
6     ports:
7       - 3000:3000
8     working_dir: /app
9     volumes:
10      - ./:/app
11     environment:
12       MYSQL_HOST: mysql
13       MYSQL_USER: root
14       MYSQL_PASSWORD: secret
15       MYSQL_DB: todos
```

3. Defining the MySQL service.

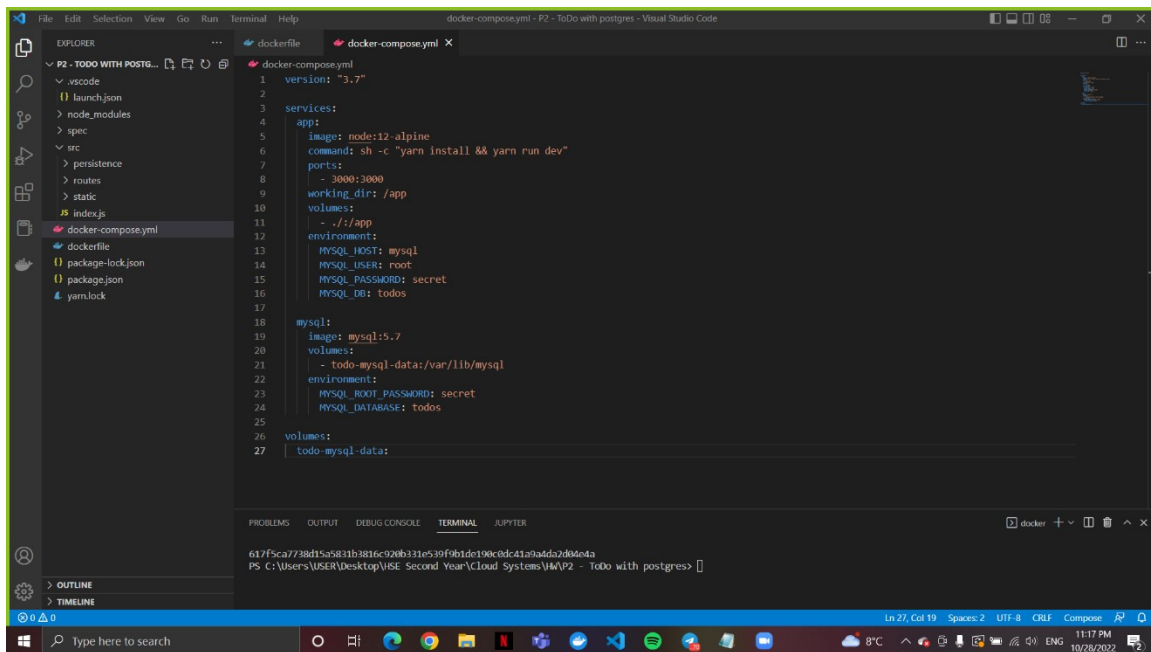


The screenshot shows the VS Code editor with a project named "P2 - TODO WITH POSTGRES". The file explorer on the left shows the project structure, including files like launch.json, node_modules, spec, src, persistence, routes, static, index.js, docker-compose.yml, dockerfile, package-lock.json, package.json, and yarn.lock. The main editor displays the docker-compose.yml file with the following content:

```
1 version: "3.7"
2 services:
3   app:
4     image: node:12-alpine
5     command: sh -c "yarn install && yarn run dev"
6     ports:
7       - 3000:3000
8     working_dir: /app
9     volumes:
10      - ./app:/app
11     environment:
12       MYSQL_HOST: mysql
13       MYSQL_USER: root
14       MYSQL_PASSWORD: secret
15       MYSQL_DB: todos
```

The terminal at the bottom shows the command `docker run -d` being executed, followed by an error message:

```
PS C:\Users\USER\Desktop\ISE Second Year\Cloud Systems\VM\P2 - To do with postgres> docker run -d
--network todo-app --network-alias mysql \
~
Missing expression after unary operator '-'.
At line:2 char:5
+ ~
+ ~~~~~
Unexpected token 'network' in expression or statement.
+ CategoryInfo          : ParserError: (1) [], ParentContainsErrorException
+ FullyQualifiedErrorId : MissingExpressionUnaryOperator
```



The screenshot shows the VS Code editor with the same project. The docker-compose.yml file has been updated to include a MySQL service. The file content is now:

```
1 version: "3.7"
2 services:
3   app:
4     image: node:12-alpine
5     command: sh -c "yarn install && yarn run dev"
6     ports:
7       - 3000:3000
8     working_dir: /app
9     volumes:
10      - ./app:/app
11     environment:
12       MYSQL_HOST: mysql
13       MYSQL_USER: root
14       MYSQL_PASSWORD: secret
15       MYSQL_DB: todos
16
17   mysql:
18     image: mysql:5.7
19     volumes:
20       - todo-mysql-data:/var/lib/mysql
21     environment:
22       MYSQL_ROOT_PASSWORD: secret
23       MYSQL_DATABASE: todos
24
25 volumes:
26   todo-mysql-data:
```









The terminal at the bottom shows the command `docker run -d` being executed, followed by a successful output:


```
PS C:\Users\USER\Desktop\ISE Second Year\Cloud Systems\VM\P2 - To do with postgres>
617f5ca7738d15a5831b3816c92eb31e539f9b1de190cddc41a9ada2d0404a
```

3. Running the application stack.

```
9977ea919cbe1b): bind for 0.0.0.0:3000 failed: port is already allocated
PS C:\Users\USER\Desktop\HSE Second Year\Cloud Systems\HW\p2 - ToDo with postgres> docker compose up -d
[+] Running 2/2
- Container p2-todowithpostgres-mysql-1 Running 0.0s
- Container p2-todowithpostgres-app-1 Started 13.7s
PS C:\Users\USER\Desktop\HSE Second Year\Cloud Systems\HW\p2 - ToDo with postgres>
```

4. checking the app stack in Docker Dashboard.

<input type="checkbox"/>		p2-todowithpostgres 2 containers	-	Running (2/2) -			
<input type="checkbox"/>		app-1 25969ec66988 	node:12-alpine	Running	3000	6 minutes ago	
<input type="checkbox"/>		mysql-1 3bb462bc7315 	mysql:5.7	Running	-	8 minutes ago	

RAM 5.46GB CPU 0.00%  Connected to Hub