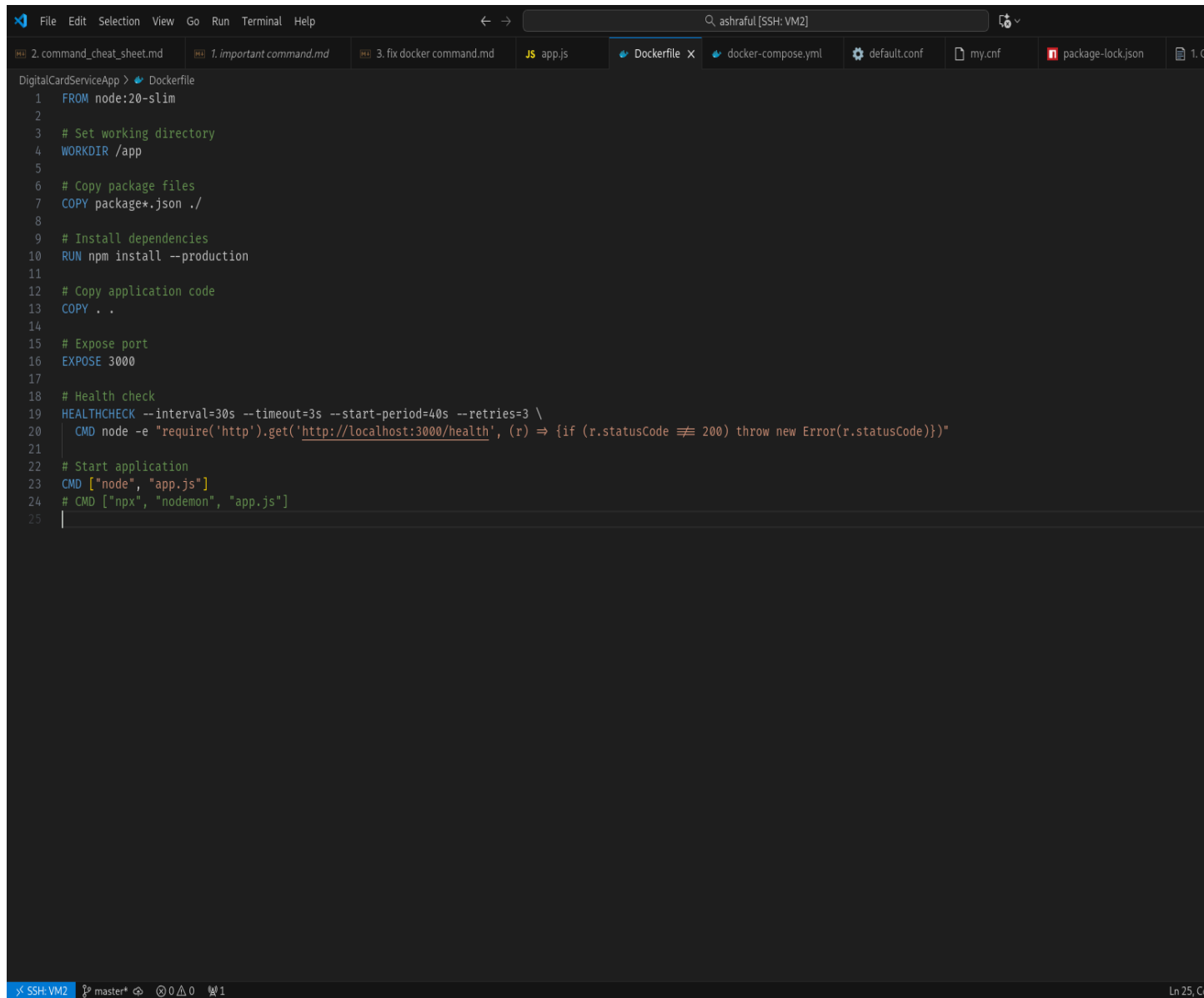


## Assignment 3:

### 1. Dockerized the Node.js (Express) App:



The screenshot shows a VS Code editor window with a terminal view open. The terminal displays the contents of a Dockerfile for a Node.js Express application. The Dockerfile includes instructions for building the image, setting the working directory, copying package files, installing dependencies, copying application code, exposing port 3000, and running the application. A health check is also defined for the container.

```
1 FROM node:20-slim
2
3 # Set working directory
4 WORKDIR /app
5
6 # Copy package files
7 COPY package*.json ./
8
9 # Install dependencies
10 RUN npm install --production
11
12 # Copy application code
13 COPY . .
14
15 # Expose port
16 EXPOSE 3000
17
18 # Health check
19 HEALTHCHECK --interval=30s --timeout=3s --start-period=40s --retries=3 \
20 | CMD node -e "require('http').get('http://localhost:3000/health', (r) => {if (r.statusCode !== 200) throw new Error(r.statusCode)});"
21
22 # Start application
23 CMD ["node", "app.js"]
24 # CMD ["npx", "nodemon", "app.js"]
25
```

### # Build Image from Dockerfile & Tag :

```
>> docker build -t express-app:1.0 .
```

```
>> docker tag express-app:1.2 ashrafulinstasure/digital_card:1.3
```

```
>> docker push ashrafulinstasure/digital_card:1.2
```

## #Docker image on the Docker Hub:

hub.docker.com/repository/docker/ashrafulinstasure/digital\_card/general

ashrafulinstasure  
Docker Personal

Repositories

Hardened Images

Collaborations

Settings

Default privacy

Notifications

Billing

Usage

Pulls

Storage

Repositories / digital\_card / General

ashrafulinstasure/digital\_card

Last pushed 1 day ago · Repository size: 72.6 MB · ☆0 · ↓22

build a digital card service

Add a category

General Tags Image Management BETA Collaborators Webhooks Settings

Tags

This repository contains 2 tag(s).

| Tag | OS          | Type  | Pulled | Pushed |
|-----|-------------|-------|--------|--------|
| 1.3 | linux/amd64 | Image | 1 day  | 1 day  |
| 1.2 | linux/amd64 | Image | 2 days | 2 days |

[See all](#)

Repository overview INCOMPLETE

An overview describes what your image does and how to run it. It displays in [the public view of your repository](#) once you have pushed some content.

[Add overview](#)

Docker commands

To push a new tag to this repository:

```
docker push ashrafulinstasure/digital_card:1.3
```

buildcloud

Build with Docker Build Cloud

Accelerate image build times with access to cloud-based build infrastructure.

Docker Build Cloud executes builds on optimally-dimensioned infrastructure with dedicated per-organization isolation.

Get faster builds through shared caching across your organization and encrypted data transfer - all without managing infrastructure.

[Go to Docker Build Cloud](#)

- 
2. Docker-Compose file that run Nginx, Mysql, redis, rabbitMQ with Bridge Network and Volume Persistence .

```
File Edit Selection View Go Run Terminal Help
ashraful [SSH: VM2]
DigitalCardServiceApp > docker-compose.yml
1 version: '3.8'
2
3 services:
4   # Node.js App with Express
5   node-app:
6     image: ashrafulinstasure/digital_card:1.3
7     container_name: node-app
8     restart: on-failure
9     working_dir: /var/www/html
10    environment:
11      - NODE_ENV=production
12      - PORT=3000
13      - DB_HOST=mysql-node
14      - DB_PORT=3306
15      - DB_DATABASE=nodejs_db
16      - DB_USERNAME=nodejs_user
17      - DB_PASSWORD=nodejs_password
18      - REDIS_HOST=redis-node
19      - REDIS_PORT=6379
20      - RABBITMQ_HOST=rabbitmq-node
21      - RABBITMQ_PORT=5672
22      - RABBITMQ_USER=guest
23      - RABBITMQ_PASSWORD=guest
24      - RABBITMQ_VHOST=/
25      - APP_URL=http://56.255.69.72:8080
26    volumes:
27      - ./var/www/html
28    ports:
29      - "3000:3000"
30    networks:
31      - nodejs_network
32    depends_on:
33      - mysql-node
34      - redis-node
35      - rabbitmq-node
36
37  # Nginx For Node.js
38  nginx-node:
39    image: nginx:alpine
40    container_name: nginx-node
41    restart: unless-stopped
42    ports:
43      - "8080:80"
44      - "8443:443"
45    volumes:
46      - ./var/www/html
47      - ./docker/nginx/default.conf:/etc/nginx/conf.d/default.conf
48    networks:
49      - nodejs_network
50    depends_on:
51      - node-app
```

### 3. docker-compose.yml :

```
version: '3.8'

services:
  # Node.js App with Express
  node-app:
    image: ashrafulinstasure/digital_card:1.3
    container_name: node-app
    restart: on-failure
    working_dir: /var/www/html
    environment:
      - NODE_ENV=production
      - PORT=3000
      - DB_HOST=mysql-node
      - DB_PORT=3306
      - DB_DATABASE=nodejs_db
      - DB_USERNAME=nodejs_user
      - DB_PASSWORD=nodejs_password
      - REDIS_HOST=redis-node
      - REDIS_PORT=6379
      - RABBITMQ_HOST=rabbitmq-node
      - RABBITMQ_PORT=5672
```

```
- RABBITMQ_USER=guest
- RABBITMQ_PASSWORD=guest
- RABBITMQ_VHOST=/
- APP_URL=http://36.255.69.72:8080
volumes:
  - ./var/www/html
ports:
  - "3000:3000"
networks:
  - nodejs_network
depends_on:
  - mysql-node
  - redis-node
  - rabbitmq-node

# Nginx for Node.js
nginx-node:
  image: nginx:alpine
  container_name: nginx-node
  restart: unless-stopped
  ports:
    - "8080:80"
    - "8443:443"
  volumes:
    - ./var/www/html
    - ./docker/nginx/default.conf:/etc/nginx/conf.d/default.conf
  networks:
    - nodejs_network
  depends_on:
    - node-app

# MySQL for Node.js
mysql-node:
  image: mysql:8.0
  container_name: mysql-node
  restart: unless-stopped
  environment:
    MYSQL_DATABASE: nodejs_db
    MYSQL_ROOT_PASSWORD: nodejs_root_password
    MYSQL_USER: nodejs_user
```

```
    MYSQL_PASSWORD: nodejs_password
ports:
  - "3308:3306"
volumes:
  - mysql_node_data:/var/lib/mysql
  - ../docker/mysql/my.cnf:/etc/mysql/my.cnf
networks:
  - nodejs_network
healthcheck:
  test: ["CMD", "mysqladmin", "ping", "-h", "localhost"]
  interval: 10s
  timeout: 5s
  retries: 5

# Redis for Node.js
redis-node:
  image: redis:7-alpine
  container_name: redis-node
  restart: unless-stopped
  ports:
    - "6381:6379"
  volumes:
    - redis_node_data:/data
  networks:
    - nodejs_network
  healthcheck:
    test: ["CMD", "redis-cli", "ping"]
    interval: 10s
    timeout: 5s
    retries: 5

# RabbitMQ - Message Queue
rabbitmq-node:
  image: rabbitmq:3.12-management-alpine
  container_name: rabbitmq-node
  restart: unless-stopped
  environment:
    RABBITMQ_DEFAULT_USER: guest
    RABBITMQ_DEFAULT_PASS: guest
  ports:
```

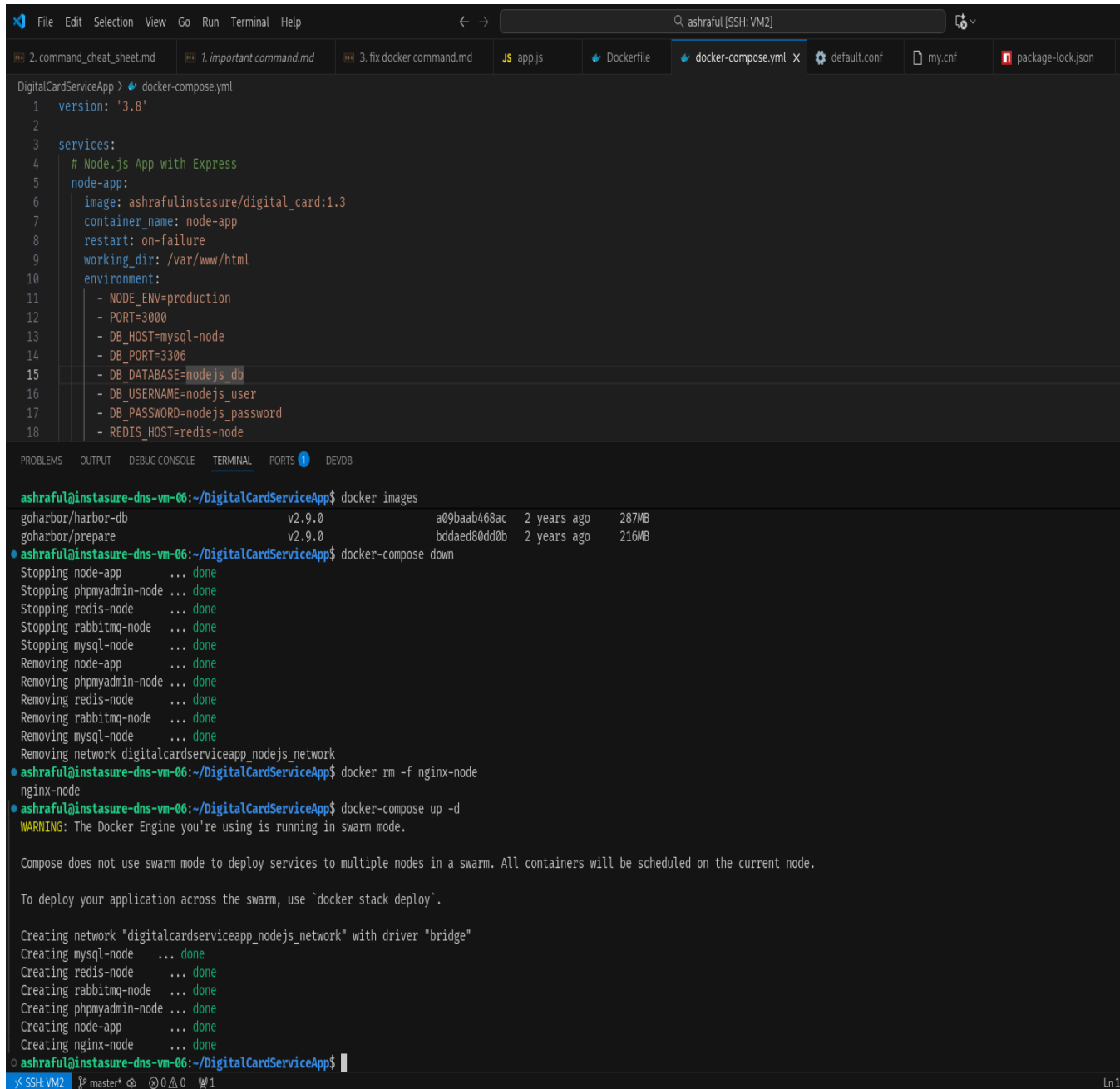
```
- "5672:5672"
- "15672:15672"
volumes:
  - rabbitmq_node_data:/var/lib/rabbitmq
networks:
  - nodejs_network
healthcheck:
  test: rabbitmq-diagnostics -q ping
  interval: 30s
  timeout: 10s
  retries: 5

# PHPMysqlAdmin for Node.js MySQL
phpmyadmin-node:
  image: phpmyadmin/phpmyadmin
  container_name: phpmyadmin-node
  restart: unless-stopped
  environment:
    PMA_HOST: mysql-node
    PMA_PORT: 3306
    PMA_USER: root
    PMA_PASSWORD: nodejs_root_password
  ports:
    - "8082:80"
  networks:
    - nodejs_network
  depends_on:
    - mysql-node

volumes:
  mysql_node_data:
    driver: local
  redis_node_data:
    driver: local
  rabbitmq_node_data:
    driver: local

networks:
  nodejs_network:
    driver: bridge
```

#### 4. Run the docker-compose.yml file :



The screenshot shows a VS Code editor with a file explorer at the top displaying several files: `2. command_cheat_sheet.md`, `1. important command.md`, `3. fix docker command.md`, `JS app.js`, `Dockerfile`, `docker-compose.yml` (selected), `default.conf`, `my.cnf`, and `package-lock.json`. The editor window shows the content of `docker-compose.yml`:

```
1 version: '3.8'
2
3 services:
4   # Node.js App with Express
5   node-app:
6     image: ashrafulinastasure/digital_card:1.3
7     container_name: node-app
8     restart: on-failure
9     working_dir: /var/www/html
10    environment:
11      - NODE_ENV=production
12      - PORT=3000
13      - DB_HOST=mysql-node
14      - DB_PORT=3306
15      - DB_DATABASE=nodejs_db
16      - DB_USERNAME=nodejs_user
17      - DB_PASSWORD=nodejs_password
18      - REDIS_HOST=redis-node
```

Below the editor, the TERMINAL tab is active, showing the following commands and their output:

```
ashraful@inastasure-dns-vm-06:~/DigitalCardServiceApp$ docker images
goharbor/harbor-db          v2.9.0          a09baab468ac    2 years ago    287MB
goharbor/prepare            v2.9.0          bddaed80dd0b    2 years ago    216MB
ashraful@inastasure-dns-vm-06:~/DigitalCardServiceApp$ docker-compose down
Stopping node-app           ... done
Stopping phpmyadmin-node ... done
Stopping redis-node        ... done
Stopping rabbitmq-node     ... done
Stopping mysql-node        ... done
Removing node-app          ... done
Removing phpmyadmin-node ... done
Removing redis-node        ... done
Removing rabbitmq-node     ... done
Removing mysql-node        ... done
Removing network digitalcardserviceapp_nodejs_network
ashraful@inastasure-dns-vm-06:~/DigitalCardServiceApp$ docker rm -f nginx-node
nginx-node
ashraful@inastasure-dns-vm-06:~/DigitalCardServiceApp$ docker-compose up -d
WARNING: The Docker Engine you're using is running in swarm mode.

Compose does not use swarm mode to deploy services to multiple nodes in a swarm. All containers will be scheduled on the current node.

To deploy your application across the swarm, use 'docker stack deploy'.

Creating network "digitalcardserviceapp_nodejs_network" with driver "bridge"
Creating mysql-node ... done
Creating redis-node ... done
Creating rabbitmq-node ... done
Creating phpmyadmin-node ... done
Creating node-app ... done
Creating nginx-node ... done
ashraful@inastasure-dns-vm-06:~/DigitalCardServiceApp$
```

The status bar at the bottom indicates the current file is `SSH: VM2`, the workspace is `master*`, and there are 0 errors and 1 warning.

5. Express On Runnig on the VPS -> <http://36.255.69.72:8080/>:

