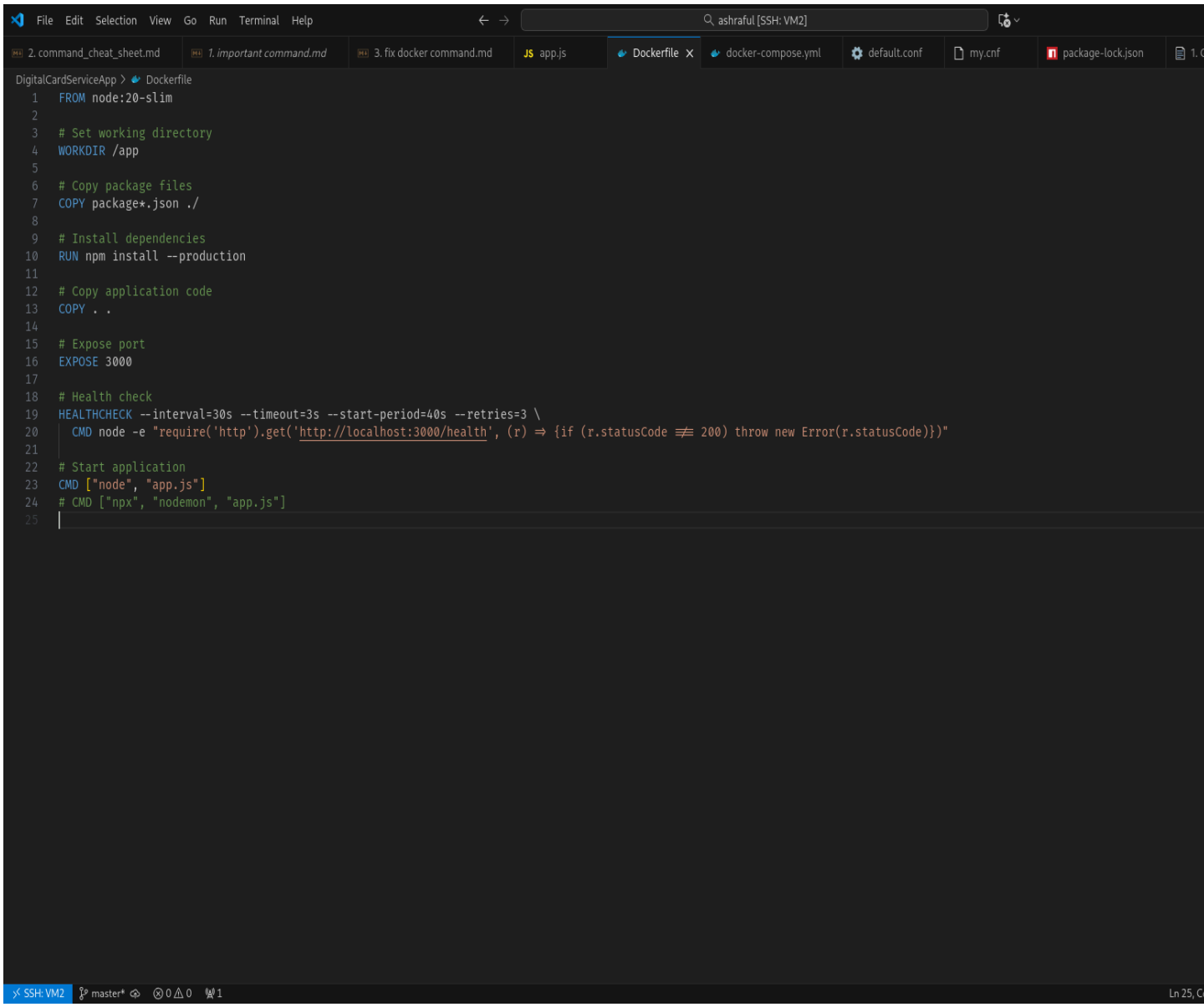
Assignment 3:

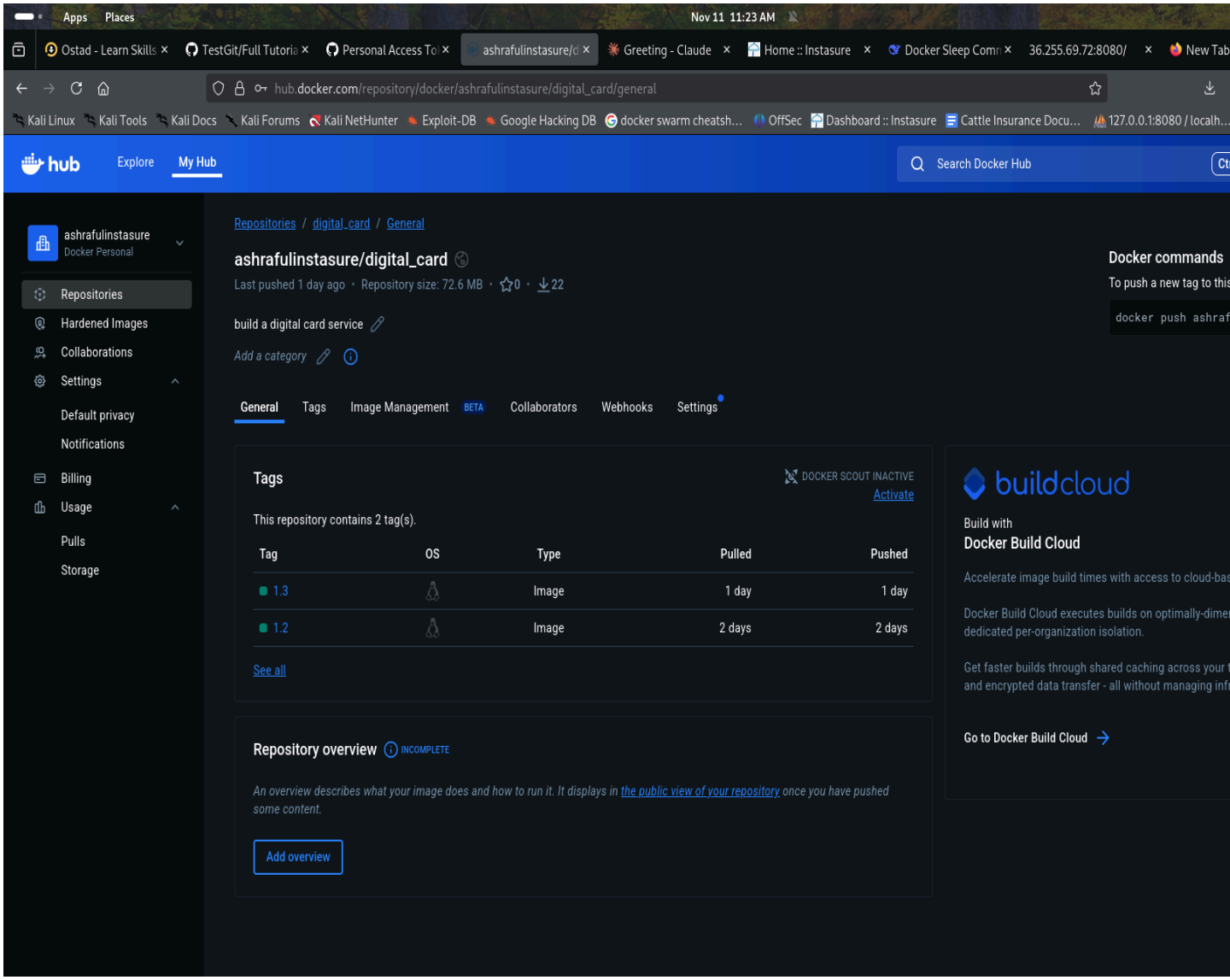
1. Dockerized the Node.js (Express) App:



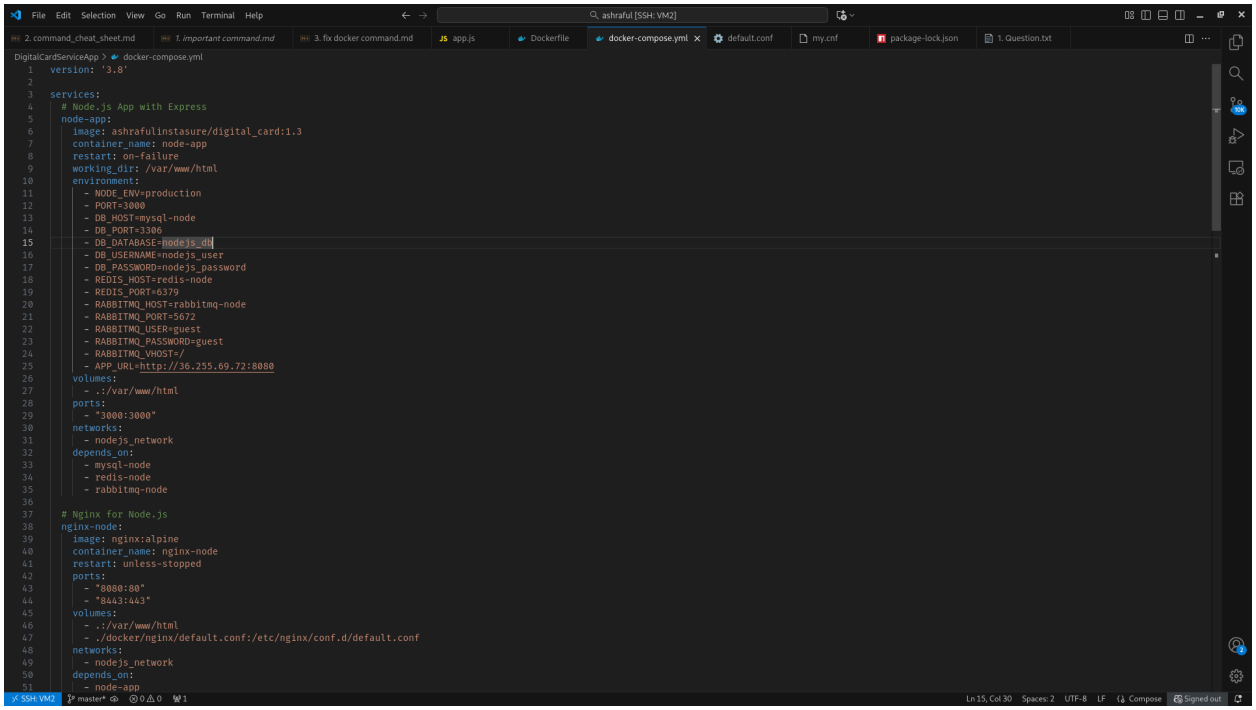
Build Image from Dockerfile & Tag :

```
> > 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:



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



3. docker-compose.yml :

```yaml
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:
```

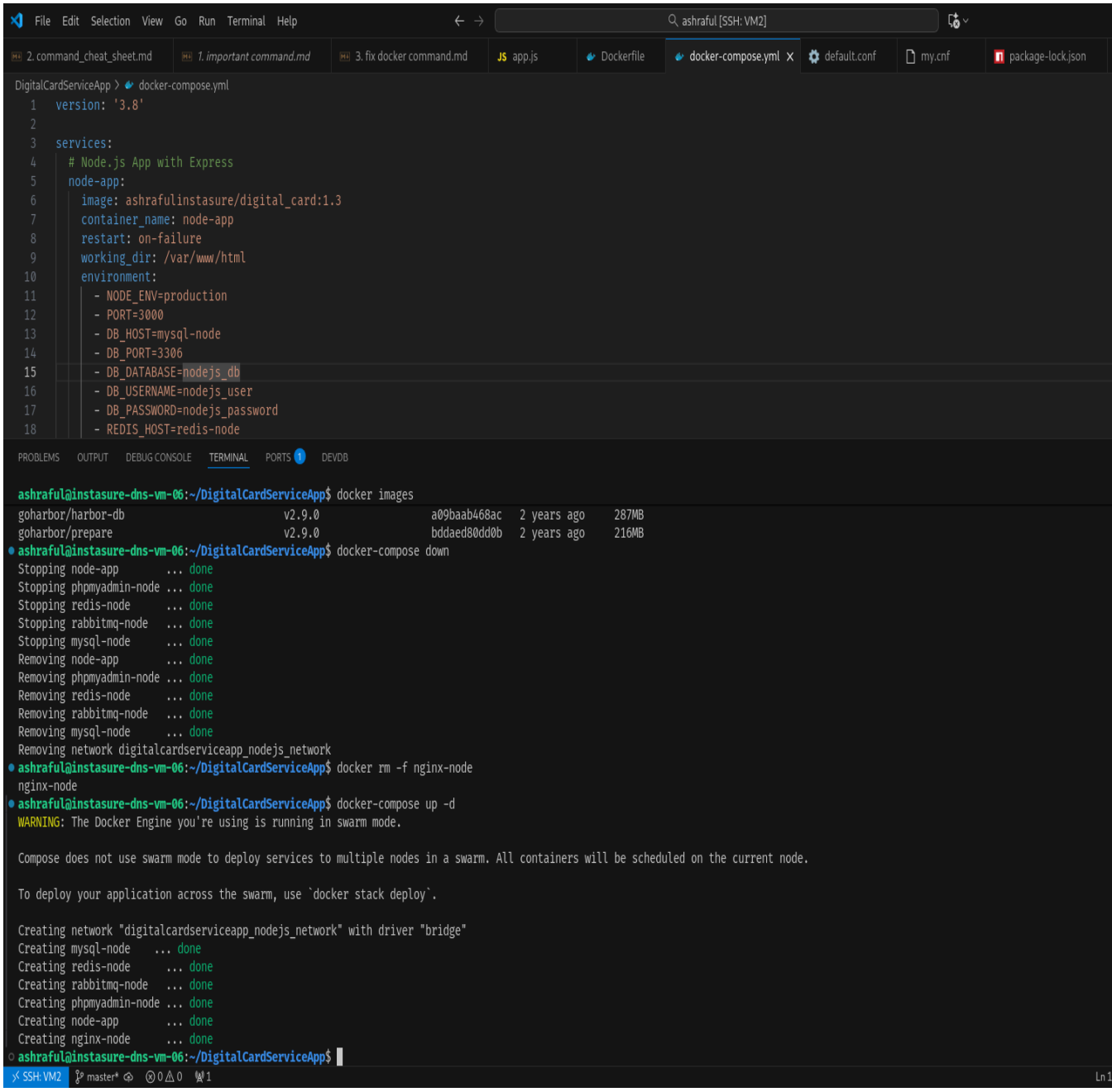
```
    - "5672:5672"
    - "15672:15672"
  volumes:
    - rabbitmq_node_data:/var/lib/rabbitmq
  networks:
    - nodejs_network
  healthcheck:
    test: rabbitmq-diagnostics -q ping
    interval: 30s
    timeout: 10s
    retries: 5

# PHPMyAdmin 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 :



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