Q1. A) Create a Container with PostgresDB or mySQL database installed

1. We create a Dockerfile for mysql database

docker-compose.yml **Dockerfile**

File Edit View

```
FROM mysql

RUN echo "MySQL Build Successful"
```

2. Create a docker-compose.yml file

docker-compose.yml Dockerfile

File Edit View

```
services:
  mysql:
    build: .
    container_name: my_mysql_container
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: rootpassword
      MYSQL_DATABASE: mydatabase
      MYSQL_USER: myuser
      MYSQL_PASSWORD: mypassword
    ports:
      - "3307:3307"
```

3. Build the file

Walkthroughs

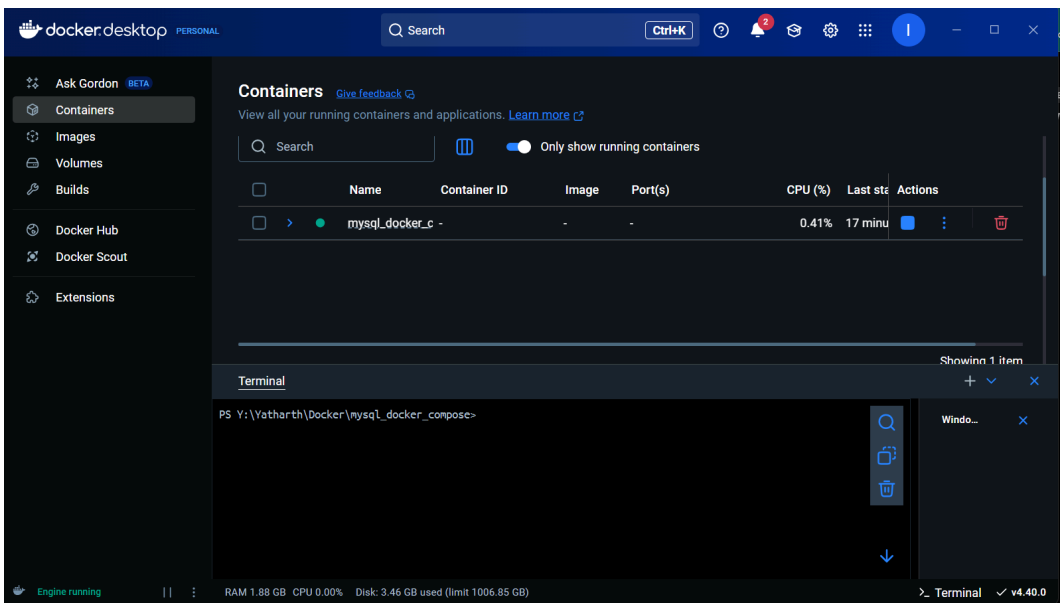
Terminal

```
PS Y:\Yatharth\Docker\mysql_docker_compose> docker-compose up --build
Compose can now delegate builds to bake for better performance.
 To do so, set COMPOSE_BAKE=true.
[+] Building 1.5s (2/3)                                          docker:desktop-linux
 => [mysql internal] load build definition from Dockerfile                        0.0s
 => => transferring dockerfile: 84B                                               0.0s
 => [mysql internal] load metadata for docker.io/library/mysql:latest             1.4s
 => [mysql auth] library/mysql:pull token for registry-1.docker.io                0.0s
```

4. Container created

```
Windows PowerShell
 => [mysql] exporting to image                                                                    0.0s
 => => exporting layers                                                                           0.0s
 => => exporting manifest sha256:b7160992e9b2e642ad5cc07dd084e51bf7a72bc7fb137bd6acc8d2e367f0036d 0.0s
 => => exporting config sha256:3bf1631087ee47fb81785538b57ad730eaaa9ecba911ae1b0e2b87d0a11db31c   0.0s
 => => exporting attestation manifest sha256:e4a04ec8639e312eb5db77a8b93467ca24ebacd05f96633eff3279b1d3346785 0.0s
 => => exporting manifest list sha256:5cd317a7daedd26a2290a70b656bb77a9297347c87c8a73384b77eb41c6d3e6c 0.0s
 => => naming to docker.io/library/mysql_docker_compose-mysql:latest                              0.0s
 => => unpacking to docker.io/library/mysql_docker_compose-mysql:latest                           0.0s
 => [mysql] resolving provenance for metadata file                                                0.0s
[+] Running 2/2
 ✔mysql                        Built                                                              0.0s
 ✔Container my_mysql_container  Recreated                                                         0.1s
Attaching to my_mysql_container
```

5. Docker Desktop showing the container

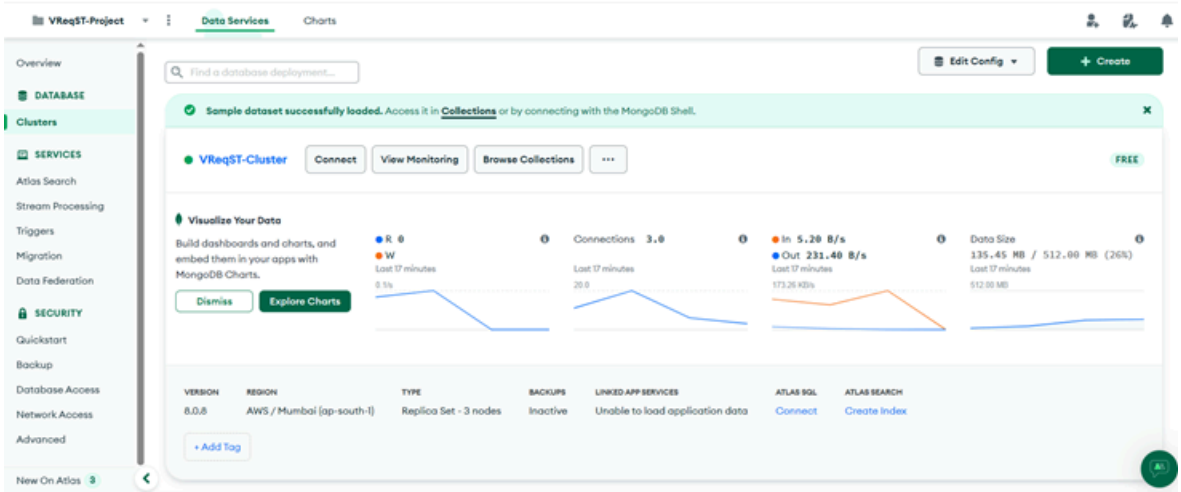


6. Starting and checking up the image

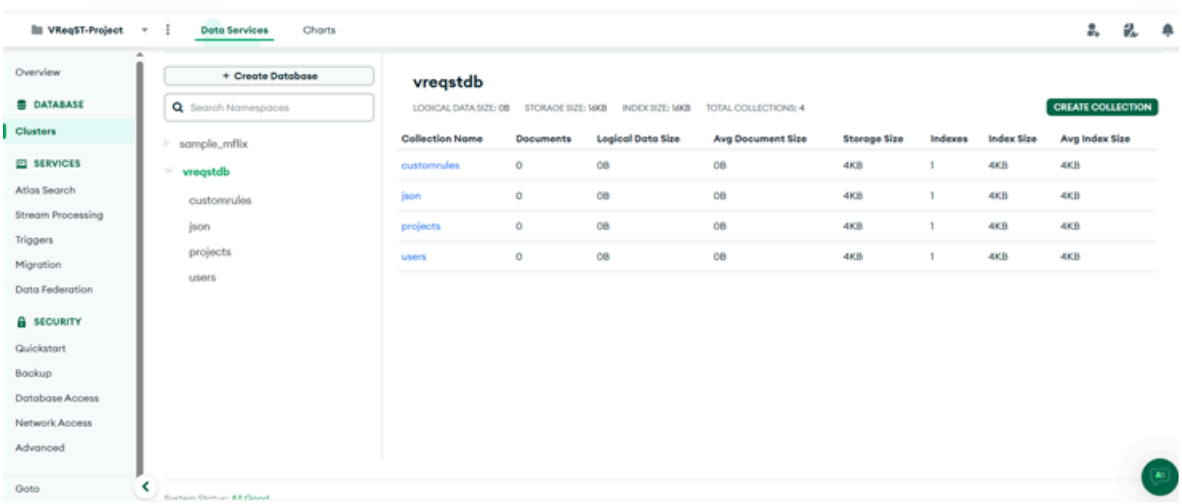
```
Y:\Yatharth\Docker\mysql_docker_compose>docker-compose up -d
Compose can now delegate builds to bake for better performance.
 To do so, set COMPOSE_BAKE=true.
[+] Building 2.2s (8/8) FINISHED                                                docker:desktop-linux
 => [mysql internal] load build definition from Dockerfile                                      0.0s
 => => transferring dockerfile: 84B                                                             0.0s
 => [mysql internal] load metadata for docker.io/library/mysql:latest                           2.0s
 => [mysql auth] library/mysql:pull token for registry-1.docker.io                              0.0s
 => [mysql internal] load .dockerignore                                                         0.0s
 => => transferring context: 2B                                                                 0.0s
 => [mysql 1/2] FROM docker.io/library/mysql:latest@sha256:7839322bd6c3174a699586c3ea36314c59b61b4ce01b4146951818 0.0s
 => => resolve docker.io/library/mysql:latest@sha256:7839322bd6c3174a699586c3ea36314c59b61b4ce01b4146951818b94aef 0.0s
 => CACHED [mysql 2/2] RUN echo "MySQL Build Successful"                                        0.0s
 => [mysql] exporting to image                                                                  0.1s
 => => exporting layers                                                                         0.0s
 => => exporting manifest sha256:b7160992e9b2e642ad5cc07dd084e51bf7a72bc7fb137bd6acc8d2e367f0036d 0.0s
 => => exporting config sha256:3bf1631087ee47fb81785538b57ad730eaaa9ecba911ae1b0e2b87d0a11db31c 0.0s
 => => exporting attestation manifest sha256:733f3d6ba051451c4aea6336cc949c49bb74665d969fd0ddf4a26003e57097e9 0.0s
 => => exporting manifest list sha256:be833768fe56e26285b0577bab03c448951e3bf68035623c642fa06e0ba30648 0.0s
 => => naming to docker.io/library/mysql_docker_compose-mysql:latest                            0.0s
 => => unpacking to docker.io/library/mysql_docker_compose-mysql:latest                         0.0s
 => [mysql] resolving provenance for metadata file                                              0.0s
[+] Running 2/2
 ✔mysql                        Built                                                            0.0s
 ✔Container my_mysql_container  Started                                                         0.5s

Y:\Yatharth\Docker\mysql_docker_compose>docker ps
CONTAINER ID   IMAGE                     COMMAND                  CREATED         STATUS         PORTS                                       NAMES
a8d50de42639   mysql_docker_compose-mysql   "docker-entrypoint.s…"   6 seconds ago   Up 5 seconds   3306/tcp, 33060/tcp, 0.0.0.0:3307->3307/tcp   my_mysql_c
ontainer
```

Q1. B) Deploy VReqST – A requirement specification tool in a container.

1. Set up MongoDB Database:
    - Create project in MongoDB Atlas and create a cluster



- customrules, jsons, projects, users are the 4 clusters that are defined



Update the application's server code by replacing local MongoDB connection string (such as mongodb://localhost:27017/vreqst) with the cloud-hosted MongoDB Atlas connection string. This connection string is typically located in one of these files: backend/server.js, backend/app.js, backend/config.js, backend/config/db.js

2. Create a DockerFile:

```
Dockerfile
FROM node:14

WORKDIR /app

COPY . .

WORKDIR /app/backend
RUN npm install

WORKDIR /app/validation_server
RUN npm install

WORKDIR /app/frontend
RUN npm install
RUN npm run client-install

EXPOSE 3000 5001 5002

CMD ["bash", "-c", "cd /app/backend && npx nodemon index.js & cd /app/validation_server && npx nodemon index.js & cd /app/frontend && npm run dev"]
```

3. Creating Docker-compose file:

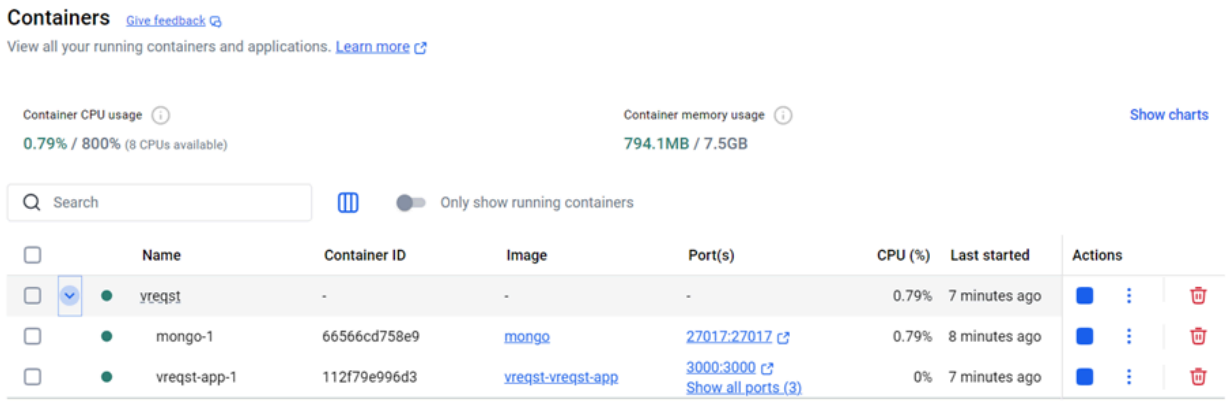
```
docker-compose.yml
services:
  vreqst-app:
    build: .
    ports:
      - "3000:3000"
      - "5001:5001"
      - "5002:5002"
    depends_on:
      - mongo
  mongo:
    image: mongo
    ports:
      - "27017:27017"
    volumes:
      - mongo-data:/data/db

volumes:
  mongo-data:
```

4. Build and Run the docker containers (docker-compose up --build)

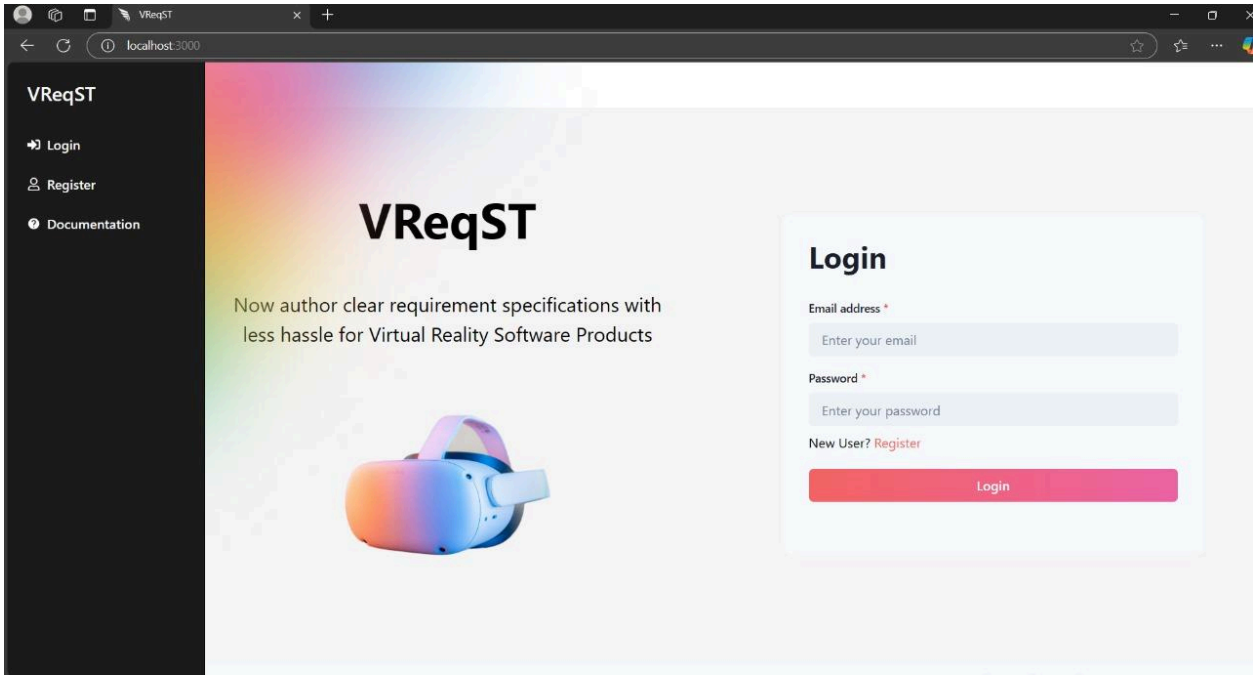
```
D:\Project\DevOps\Final Assignment\VReqST>docker-compose up --build
time="2025-04-20T18:35:41+05:30" level=warning msg="D:\\Project\\DevOps\\Final Assignment\\VReqST\\docker-compose.yml: the attribute `version` is obsolete, it will be i
 confusion"
Compose can now delegate builds to bake for better performance.
 To do so, set COMPOSE_BAKE=true.
[+] Building 360.4s (28/28) FINISHED
 => [backend internal] load build definition from Dockerfile
 => => transferring dockerfile: 161B
 => [validation_server internal] load build definition from Dockerfile
 => => transferring dockerfile: 167B
 => [frontend internal] load build definition from Dockerfile
 => => transferring dockerfile: 163B
 => [backend internal] load metadata for docker.io/library/node:14
 => [validation_server auth] library/node:pull token for registry-1.docker.io
 => [validation_server internal] load .dockerignore
 => => transferring context: 2B
 => [frontend internal] load .dockerignore
 => => transferring context: 2B
 => [backend internal] load .dockerignore
 => => transferring context: 2B
 => [validation_server 1/5] FROM docker.io/library/node:14@sha256:a158d3b9b4e3fa813fa6c8c590b8f0a860e015ad4e59bbce5744d2f6fd8461aa
 => => resolve docker.io/library/node:14@sha256:a158d3b9b4e3fa813fa6c8c590b8f0a860e015ad4e59bbce5744d2f6fd8461aa
 => => sha256:5f32ed3c3f278edda4fc571c880b5277355a29ae8f52b52cdf865f058378a590 35.24MB / 35.24MB
 => => sha256:0d27a8e861329007574c6766fba946d48e20d2c8e964e873de35260 3f22c4ceb 450B / 450B
 => => sha256:0c8cc2f24a4dcb64e602e086fc9446b0a541e8acd9ad72d2e90df3ba22f158b3 2.29MB / 2.29MB
 => => sha256:6f51ee005deac0d99898e41b8ce60ebf250ebe1a31a0b03f613aec6bbc9b83d8 4.19kB / 4.19kB
 => => sha256:d9a8df5894511ce28a05e2925a75e8a4acbd0634c39ad734fdfba8e23d1b1569 191.85MB / 191.85MB
 => => sha256:1de76e268b103d05fa8960e0f77951ff54b912b63429c34f5d6adfd09f5f9ee2 51.38MB / 51.88MB
```

```
 => [backend 5/5] COPY . .
 => [backend] exporting to image
 => => exporting layers
 => => exporting manifest sha256:7533ba38e8b9ed396f982b10180223592b93e9991316b0d35840f00a65766da2
 => => exporting config sha256:1be18f4003472becd5fb50e0913cc0d82e2703ea93a8555af1f9f698e528a779
 => => exporting attestation manifest sha256:0b380f15fa958b28ec7c6a02262caff55f1c1cff832e5353d690a8e5ccecf482
 => => exporting manifest list sha256:adf4c2f1a0afe404bcf683980a7420a52d1e922de958b50da8712c612e1fc2b5
 => => naming to docker.io/library/vreqst-backend:latest
 => => unpacking to docker.io/library/vreqst-backend:latest
 => [frontend 3/5] COPY package*.json ./
 => [frontend 4/5] RUN npm install
 => [backend] resolving provenance for metadata file
 => [frontend 5/5] COPY . .
 => [frontend] exporting to image
 => => exporting layers
 => => exporting manifest sha256:8ce45161458f5b800ef11f351369b295bef7a1ab3024b439b445753f685ab974
 => => exporting config sha256:9f6197f47626fc2d3a9314dcb3c443660a50cd4732e86a2f14cd4ac004ff035c
 => => exporting attestation manifest sha256:aa1d92b9dcb31c516332c0c556eeaa6c276dd5459ab9181ca187b2e43029715 4
 => => exporting manifest list sha256:3fa04b60c7342710398abe42d1f80f4d1caaa6a825f6f366e59683f3b8adbfee
 => => naming to docker.io/library/vreqst-frontend:latest
 => => unpacking to docker.io/library/vreqst-frontend:latest
 => [frontend] resolving provenance for metadata file
[+] Running 3/3
 ✔ backend                Built
[+] Running 6/6           Built
 ✔ backend                                Built
 ✔ frontend                               Built
 ✔ validation_server                      Built
 ✔ Container vreqst-backend-1             Created
 ✔ Container vreqst-validation_server-1   Created
 ✔ Container vreqst-frontend-1            Created
Attaching to backend-1, frontend-1, validation_server-1
```

Currently active containers:



5. Running the application:
   - http://localhost:3000



Outcome:

The application runs inside the container and is connected to the MongoDB Atlas database instance. The application can be accessed through the ports that are defined in the docker-compose.yml

MongoDB would be accessible on mongodb://localhost:27017