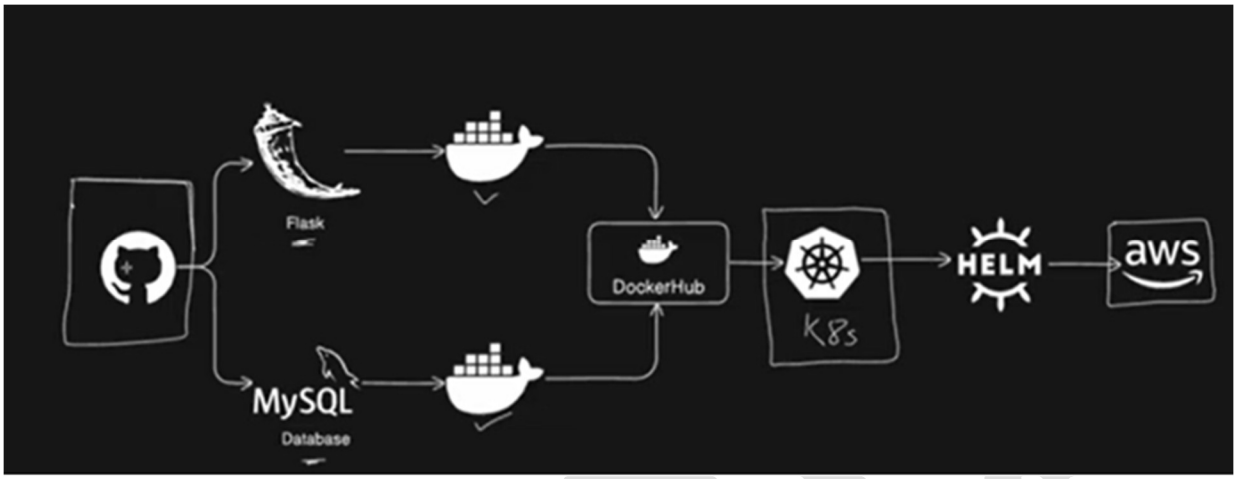
Part 1

Title : 2-Tier Application Deployed On AWS Cloud



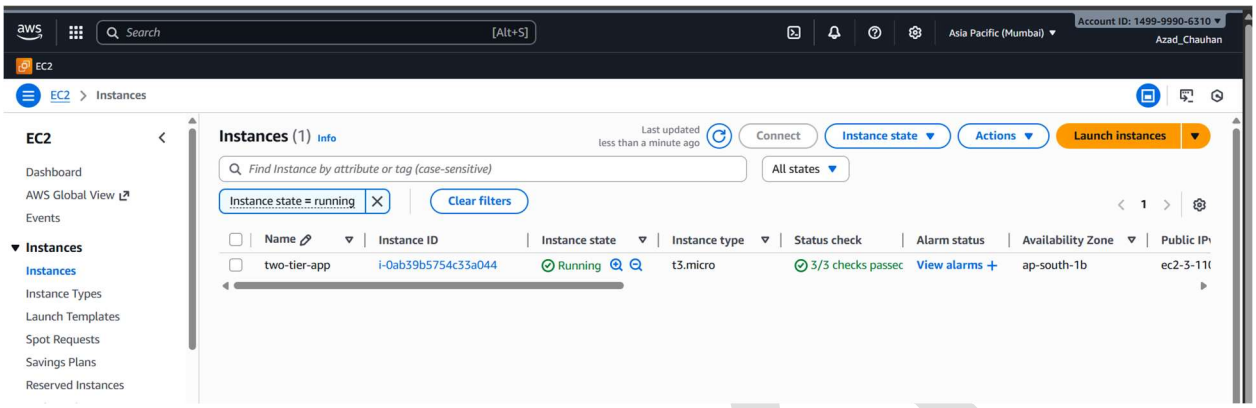
This diagram represents a **2-tier application** deployed using:

- ✅ GitHub
- ✅ Flask (Application Layer)
- ✅ MySQL (Database Layer)
- ✅ Docker & DockerHub
- ✅ Kubernetes (K8s)
- ✅ Helm
- ✅ AWS Cloud

Step 1. 1. Launch EC2 Instance

1. Login to AWS Console → EC2
2. Click **Launch Instance**
3. Choose AMI: **Ubuntu Server 22.04 LTS**
4. Instance Type: **t2.micro** (Free Tier)
5. Create/Select Key Pair → Download .pem

6. Security Group: Allow
    - **SSH (22)**
7. Launch Instance



1.2 Connect your EC2 to local machine CMD through ssh Command.

ssh -i "john.pem" ubuntu@ec2-3-110-151-195.ap-south-1.compute.amazonaws.com

1.Update System

Sudo apt update -y

Sudo apt upgrade -y

2.Install Docker(Ubuntu)

Sudo apt install docker.io -y

List Running Containers (docker ps)

If we gate permission denied erroer then allow the permission to your user, using this command (find user type whoami)

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-9-89:~$ sudo systemctl start docker
sudo systemctl enable docker
ubuntu@ip-172-31-9-89:~$ docker ps
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar
%2Frun%2Fdocker.sock/v1.50/containers/json": dial unix /var/run/docker.sock: connect: permission denied
ubuntu@ip-172-31-9-89:~$ sudo chown ubuntu /var/run/docker.sock
ubuntu@ip-172-31-9-89:~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
ubuntu@ip-172-31-9-89:~$
```

(sudo chown username path) path = /var/run/docker.sock

STEP 2: Install Git if need & Clone the Application

2.1. Clone Your Application Repository

git clone https://github.com/your-username/your-repo.git

git clone https://github.com/chauhanazad726/Two-Tier-App-Deployment.git

2.2. Go Inside the Project Folder

cd your-repo

2.3. List Files to Confirm Project Cloned (ls, ls -l)

Step 3: Create Dockerfile (with MySQL Support)

3.1. Create Dockerfile (vim Dockerfile) Open Docker editer

Code:

```
FROM python:3.9-slim

WORKDIR /app

RUN apt-get update -y \
    && apt-get install -y gcc default-libmysqlclient-dev pkg-config \
    && rm -rf /var/lib/apt/lists/*

COPY requirements.txt .

RUN pip install mysqlclient
RUN pip install -r requirements.txt

COPY . .
```

EXPOSE 5000

CMD ["python", "app.py"]

3.2 : Save Dockerfile & Build Docker Image

Save & Exit (Esc, :wq)

3.3. Create Docker network (docker network create twotier **)**

3.4. Start MySQL container

```
docker run -d \
 --name mysql \
 --network=twotier \
 -e MYSQL_ROOT_PASSWORD=azad \
 -e MYSQL_DATABASE=mydb \
 mysql:5.7
```

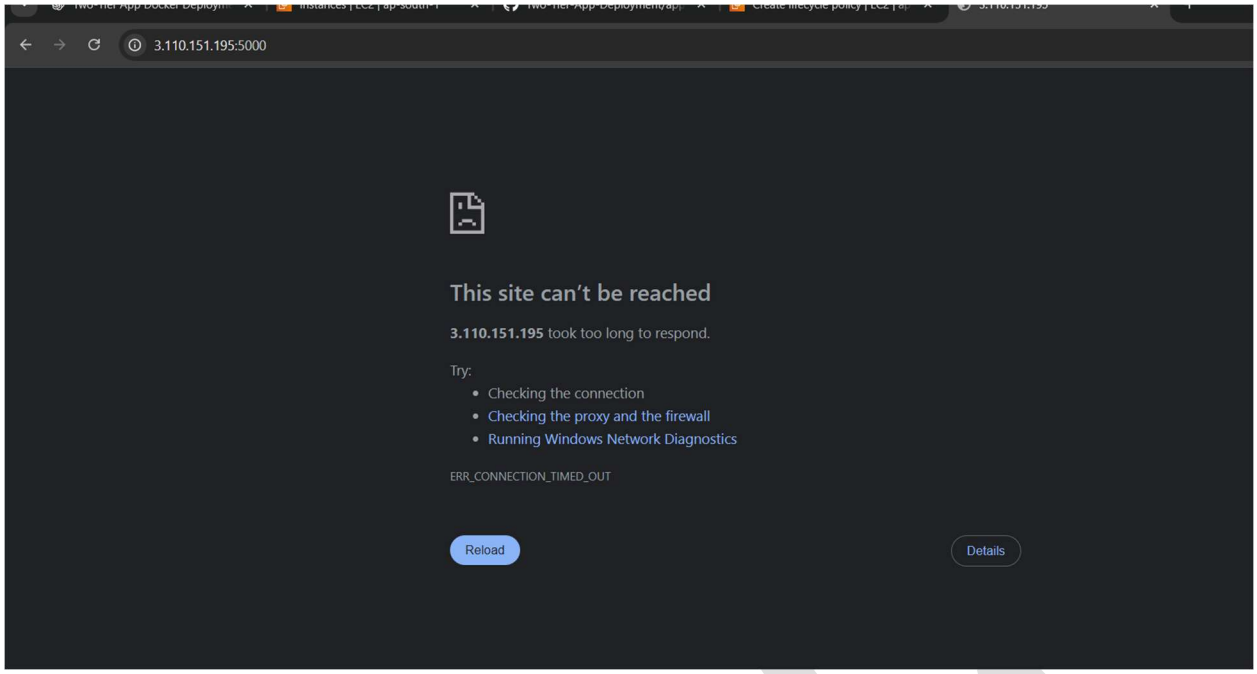
3.5. Build Flask image (docker build -t flaskapp:latest .)

3.6. Run Flask container

```
docker run -d \
 -p 5000:5000 \
 --name flask \
 --network=twotier \
 -e MYSQL_HOST=mysql \
 -e MYSQL_USER=root \
 -e MYSQL_PASSWORD=azad \
 -e MYSQL_DB=mydb \
 flaskapp:latest
```

3.7. Now open again

http://3.110.151.195:5000



Step 4: Enable Inbound Rule for Port 5000 on EC2

To access your Flask application from a browser, you must allow TCP port 5000 in your EC2 security group.

4.1 Go to EC2 Dashboard

1. Open AWS Management Console
2. Navigate to EC2
3. Click Instances
4. Select your running instance

4.2 Open Security Group Settings

1. Scroll down to Security
2. Click on the Security Group attached to the instance

4.3 Add Inbound Rule for Port 5000

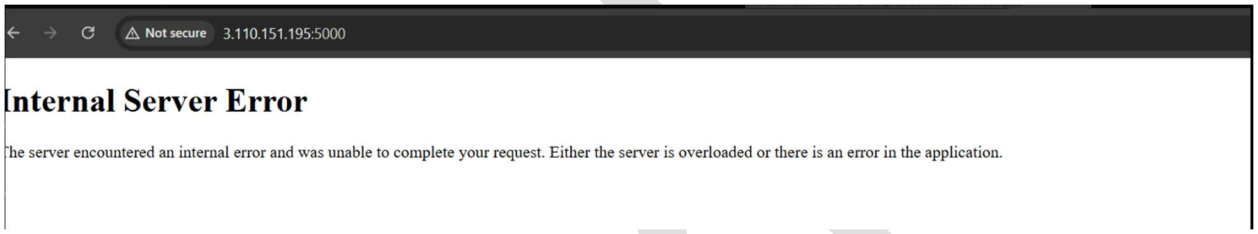
1. Click Edit Inbound Rules
2. Click Add Rule
3. Choose:

- Type: Custom TCP
- Port Range: 5000
- Source: 0.0.0.0/0 (allow from any IP)

4. **Click Save Rules**

5. Test the Application Open browser:

(http://<EC2-Public-IP>:5000)



5.1.Solution: run `docker logs flask` Purpose ,To see Flask container errors To find why Internal Server Error (500) happens

The error says:

Table 'mydb.messages' doesn't exist

This means:

Inside your MySQL container, the table messages was never created.

When Flask tries to run this query:

SELECT message FROM messages

MySQL returns an error because the table does not exist.

Enter inside MySQL container

Run: `docker exec -it mysql mysql -u root -p azad`

Select database `USE mydb;`

Create the table manually

Paste this:

```
CREATE TABLE IF NOT EXISTS messages (
    id INT AUTO_INCREMENT PRIMARY KEY,
    message TEXT
);
```

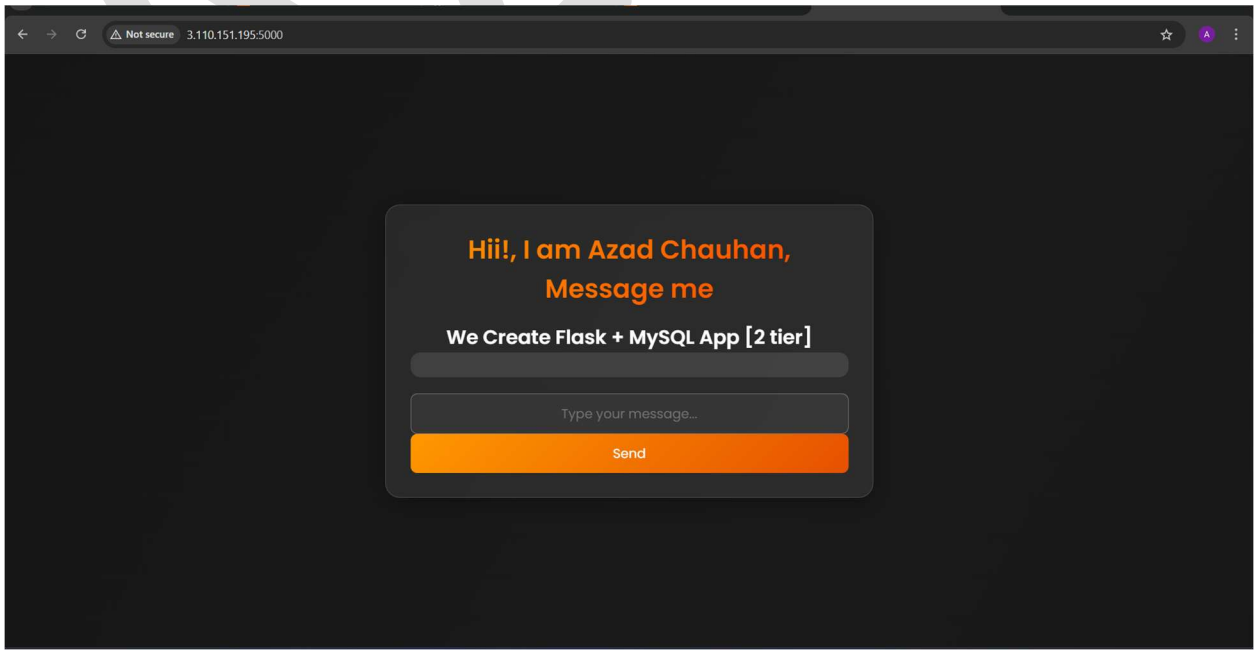
Verify table created: **`SHOW TABLES;`**

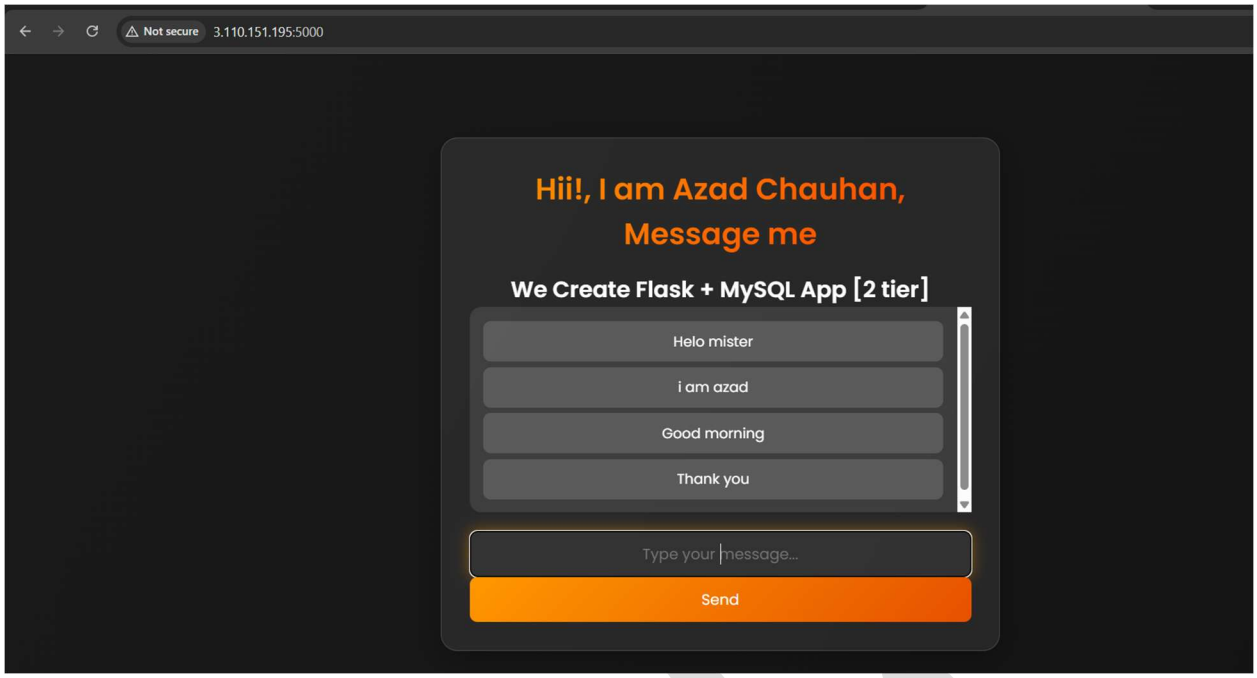
Restart Flask container: `docker restart flask`

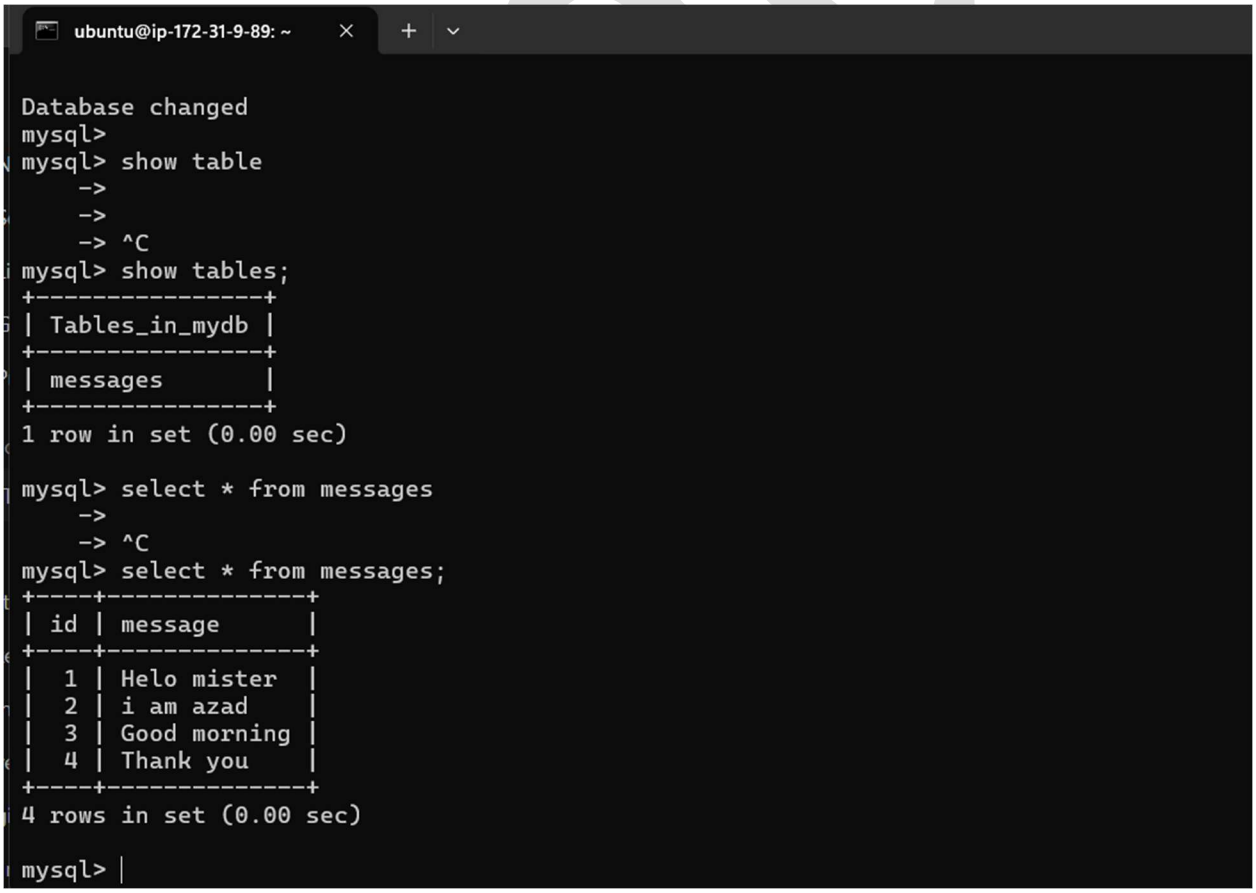
```
  File "/usr/local/lib/python3.9/site-packages/MySQLdb/cursors.py", line 179, in execute
    res = self._query(mogrified_query)
  File "/usr/local/lib/python3.9/site-packages/MySQLdb/cursors.py", line 330, in _query
    db.query(q)
  File "/usr/local/lib/python3.9/site-packages/MySQLdb/connections.py", line 280, in query
    _mysql.connection.query(self, query)
MySQLdb.ProgrammingError: (1146, "Table 'mydb.messages' doesn't exist")
47.11.5.178 - - [05/Dec/2025 04:50:33] "GET / HTTP/1.1" 500 -
ubuntu@ip-172-31-9-89:~$ docker exec -it mysql mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 15
Server version: 5.7.44 MySQL Community Server (GPL)

Copyright (c) 2000, 2023, 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> USE mydb;
Database changed
mysql> CREATE TABLE IF NOT EXISTS messages (
    ->     id INT AUTO_INCREMENT PRIMARY KEY,
    ->     message TEXT
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> SHOW TABLES;
+----------------+
| Tables_in_mydb |
+----------------+
| messages       |
+----------------+
1 row in set (0.00 sec)

mysql>
```

6. Test the Application Open browser:

(http://<EC2-Public-IP>:5000) http://3.110.151.195:5000




Not secure
3.110.151.195:5000
Hii!, I am Azad Chauhan,
Message me
We Create Flask + MySQL App [2 tier]
Helo mister
i am azad
Good morning
Thank you
Type your message...
Send


ubuntu@ip-172-31-9-89: ~
Database changed
mysql>
mysql> show table
-> ->
-> ^C
mysql> show tables;
+----------------+
| Tables_in_mydb |
+----------------+
| messages |
+----------------+
1 row in set (0.00 sec)
mysql> select * from messages
-> ->
-> ^C
mysql> select * from messages;
+----+-------------+
| id | message |
+----+-------------+
1	Helo mister
2	i am azad
3	Good morning
4	Thank you
+----+-------------+
4 rows in set (0.00 sec)
mysql>

Your Flask + MySQL 2-tier application is LIVE on AWS EC2 and working perfectly.

- **The messages are showing correctly**
- **Database connection is successful**
- **The UI is loading properly**
- **Your deployment is complete and successful**

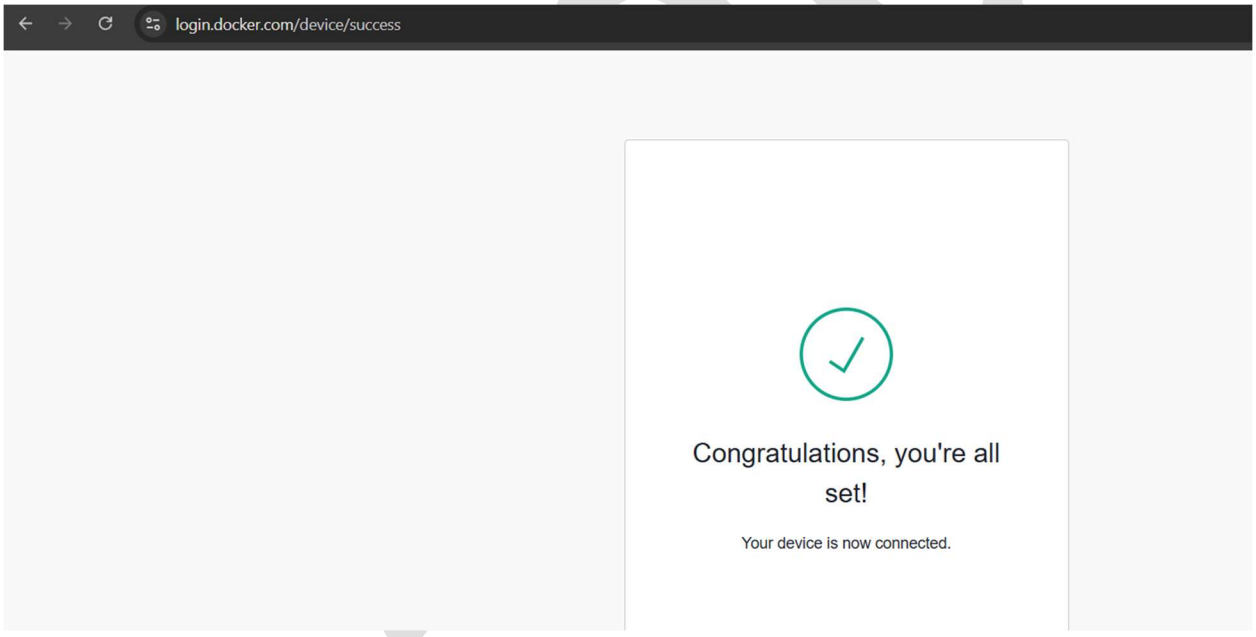
STEP 7 — Tag and Push the Flask App Image to Docker Hub

Step 1 — Log in to Docker Hub

Run this on your EC2 instance:

docker login it gave you one secret code .

https://login.docker.com/activate paste code here.



Enter your:

- Docker Hub username
- Docker Hub password

You will see:

Login Succeeded

```
ubuntu@ip-172-31-9-89:~$
ubuntu@ip-172-31-9-89:~$ ls
Two-Tier-App-Deployment
ubuntu@ip-172-31-9-89:~$ cd Two-Tier-App-Deployment
ubuntu@ip-172-31-9-89:~/Two-Tier-App-Deployment$ docker login

USING WEB-BASED LOGIN

i Info → To sign in with credentials on the command line, use 'docker login -u <username>'


Your one-time device confirmation code is: QCJF-THVG
Press ENTER to open your browser or submit your device code here: https://login.docker.com/activate

Waiting for authentication in the browser…

WARNING! Your credentials are stored unencrypted in '/home/ubuntu/.docker/config.json'.
Configure a credential helper to remove this warning. See
https://docs.docker.com/go/credential-store/

Login Succeeded
ubuntu@ip-172-31-9-89:~/Two-Tier-App-Deployment$
```

Step 2 — Tag your local Docker image

Your local image name is: flaskapp:latest

STEP 3 — Tag your local image

Run this: docker tag flaskapp:latest kkggff/flaskapp:latest

STEP 4 — Push the image to Docker Hub

Run: docker push kkggff/flaskapp:latest

(This will upload your image to your Docker Hub account.You will see something like:

Pushing layers... ,latest: digest: sha256:xxxx size: 1234)

```
1 additional security update can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm


*** System restart required ***
Last login: Fri Dec  5 06:01:14 2025 from 47.11.5.232
ubuntu@ip-172-31-9-89:~$ ls
Two-Tier-App-Deployment
ubuntu@ip-172-31-9-89:~$ cd Two-Tier-App-Deployment
ubuntu@ip-172-31-9-89:~/Two-Tier-App-Deployment$ docker tag flaskapp:latest kkggff/flaskapp:latest
ubuntu@ip-172-31-9-89:~/Two-Tier-App-Deployment$ docker push kkggff/flaskapp:latest
The push refers to repository [docker.io/kkggff/flaskapp]
2ebbafeb77cc: Pushed
02cd15b00853: Pushed
aed575d6317e: Pushed
46b1834de55e: Pushed
16e78f1bfbf1: Pushed
52a2921c3b1c: Pushed
c8f6b54339a8: Mounted from library/python
298992e09a03: Mounted from library/python
4f237755fbae: Mounted from library/python
d7c97cb6f1fe: Mounted from library/python
latest: digest: sha256:5ee1ac7f11893cd51d3b824b1362a3c533c3ad039498098d5561a0b80a531243 size: 2415
ubuntu@ip-172-31-9-89:~/Two-Tier-App-Deployment$
```

STEP 5 — Check Docker Hub

Go to:

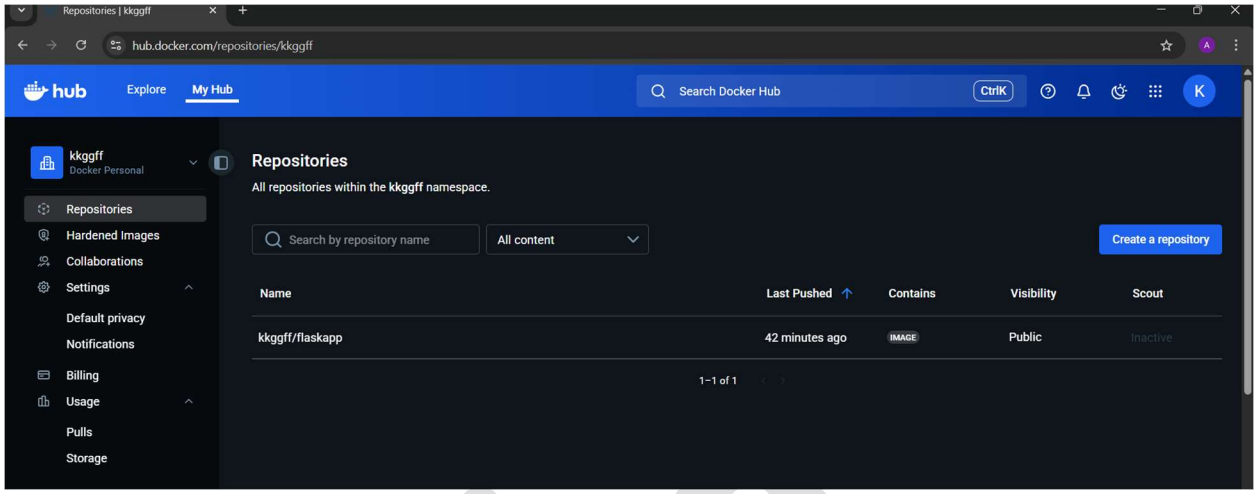
👉 https://hub.docker.com/repositories

You will see: kkggff/flaskapp

Your Flask application image will now be available globally on Docker Hub.

Anyone can pull it:

docker pull kkggff/flaskapp:latest



Step 8: Deploy the 2-Tier Application Using Docker Compose

Docker Compose allows us to run multiple containers (Flask + MySQL) using a single YAML file and a single command. This simplifies deployment and ensures both services run together on the same network.

- Most Ubuntu AMIs already include Docker Compose.
- `sudo apt install docker-compose`

STEP 1 — Create a docker-compose.yml file

Inside your project folder:

cd Two-Tier-App-Deployment

Create the file: vim docker-compose.yml

Code:

version: '3.8'

services:

```
 mysql:
  image: mysql:5.7
  container_name: mysql
  environment:
    MYSQL_ROOT_PASSWORD: azad
    MYSQL_DATABASE: mydb
  networks:
    - twotier
  volumes:
    - mysql_data:/var/lib/mysql

flask:
  image: kkggff/flaskapp:latest   # your Docker Hub image
  container_name: flask
  depends_on:
    - mysql
  networks:
    - twotier
  ports:
    - "5000:5000"
  environment:
    MYSQL_HOST: mysql
    MYSQL_USER: root
    MYSQL_PASSWORD: azad
    MYSQL_DB: mydb
```

networks:

twotier:

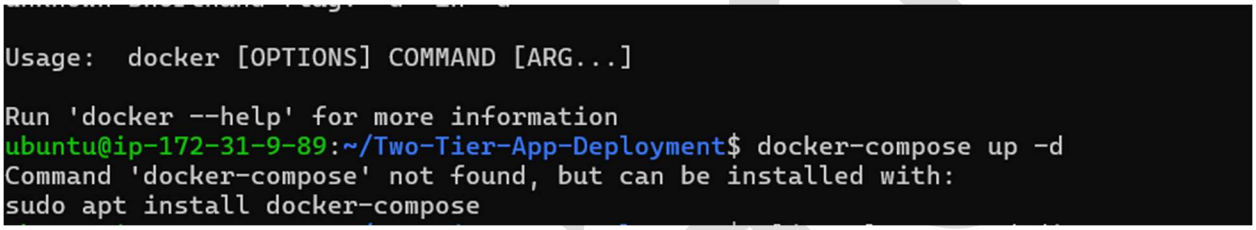
volumes:

mysql_data:

STEP 2 — Start the 2-tier application

Run: docker-compose up -d

This will start:✓ MySQL container ✓ Flask container ✓ Network ✓ Volume ✓ Env variables ALL AT ONCE 🎉



Gate error

Install Docker Compose

Run this connamd in ec2 cmd: sudo apt update

sudo apt install docker-compose -y

Check installation

Run: docker-compose --version

Start your 2-tier app

Now finally run: docker-compose up -d

(Ye compose file padhega, MySQL + Flask dono containers bana dega.)

IF you gate this error: Error: MySQL container already exists

Cannot create container for service mysql:

The container name "/mysql" is already in use

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-9-89:~/Two-Tier-App-Deployment$ docker-compose --version
docker-compose version 1.29.2, build unknown
ubuntu@ip-172-31-9-89:~/Two-Tier-App-Deployment$ docker-compose up -d
Creating network "two-tier-app-deployment_twotier" with the default driver
Creating volume "two-tier-app-deployment_mysql_data" with default driver
Creating mysql ... error

ERROR: for mysql  Cannot create container for service mysql: Conflict. The container name "/mysql" is already in use by
container "defd9fa885ac9e6a35734796d96e3bf1bde3c2e7c631bb097d91f89a02bdbeff". You have to remove (or rename) that contai
ner to be able to reuse that name.

ERROR: for mysql  Cannot create container for service mysql: Conflict. The container name "/mysql" is already in use by
container "defd9fa885ac9e6a35734796d96e3bf1bde3c2e7c631bb097d91f89a02bdbeff". You have to remove (or rename) that contai
ner to be able to reuse that name.
ERROR: Encountered errors while bringing up the project.
ubuntu@ip-172-31-9-89:~/Two-Tier-App-Deployment$
```

Check first: **docker ps -a** we are able to see container id and name know we have to kill both Contaner (Exam docker rm -f conatinerid) because docker composed command create container.

```
ERROR: Encountered errors while bringing up the project.
ubuntu@ip-172-31-9-89:~/Two-Tier-App-Deployment$ docker ps -a
CONTAINER ID   IMAGE             COMMAND                  CREATED       STATUS       PORTS
          NAMES
d27f00640148   flaskapp:latest   "python app.py"          8 hours ago   Up 8 hours   0.0.0.0:5000->5000/tcp, [::]:5000->
5000/tcp   flask
defd9fa885ac   mysql:5.7         "docker-entrypoint.s…"   8 hours ago   Up 8 hours   3306/tcp, 33060/tcp
          mysql
ubuntu@ip-172-31-9-89:~/Two-Tier-App-Deployment$ docker rm -f defd9fa885ac
defd9fa885ac
ubuntu@ip-172-31-9-89:~/Two-Tier-App-Deployment$ docker ps -a
CONTAINER ID   IMAGE             COMMAND           CREATED       STATUS       PORTS
    NAMES
d27f00640148   flaskapp:latest   "python app.py"   8 hours ago   Up 8 hours   0.0.0.0:5000->5000/tcp, [::]:5000->5000/tc
p   flask
ubuntu@ip-172-31-9-89:~/Two-Tier-App-Deployment$
```

Then start docker compose again: docker-compose up -d

```
ubuntu@ip-172-31-9-89:~/Two-Tier-App-Deployment$ docker ps -a
CONTAINER ID   IMAGE             COMMAND                  CREATED         STATUS         PORTS
               NAMES
c603e149d1b5   mysql:5.7         "docker-entrypoint.s…"   2 minutes ago   Up 2 minutes   3306/tcp, 33060/tcp
               mysql
d27f00640148   flaskapp:latest   "python app.py"          8 hours ago     Up 8 hours     0.0.0.0:5000->5000/tcp, [::]:50
00->5000/tcp   flask
ubuntu@ip-172-31-9-89:~/Two-Tier-App-Deployment$ docker rm -f d27f0064014
d27f0064014
ubuntu@ip-172-31-9-89:~/Two-Tier-App-Deployment$ docker-compose up -d
mysql is up-to-date
Creating flask ... done
ubuntu@ip-172-31-9-89:~/Two-Tier-App-Deployment$
```

Check containers: docker ps

Both container show the running status up: flask, mysql.

Now open your app in browser

👉 Go to: http://3.110.151.195:5000

Internal Server Error

The server encountered an internal error and was unable to complete your request. Either the server is overloaded or there is an error in the application.

Error is simple table is not created automated know we have to create table

Step 1: Enter MySQL container

docker exec -it mysql mysql -u root -p

Step 2: Select the database

use mydb;

Step 3: Create the table manually

```
CREATE TABLE IF NOT EXISTS messages (
   id INT AUTO_INCREMENT PRIMARY KEY,
   message TEXT
);
```

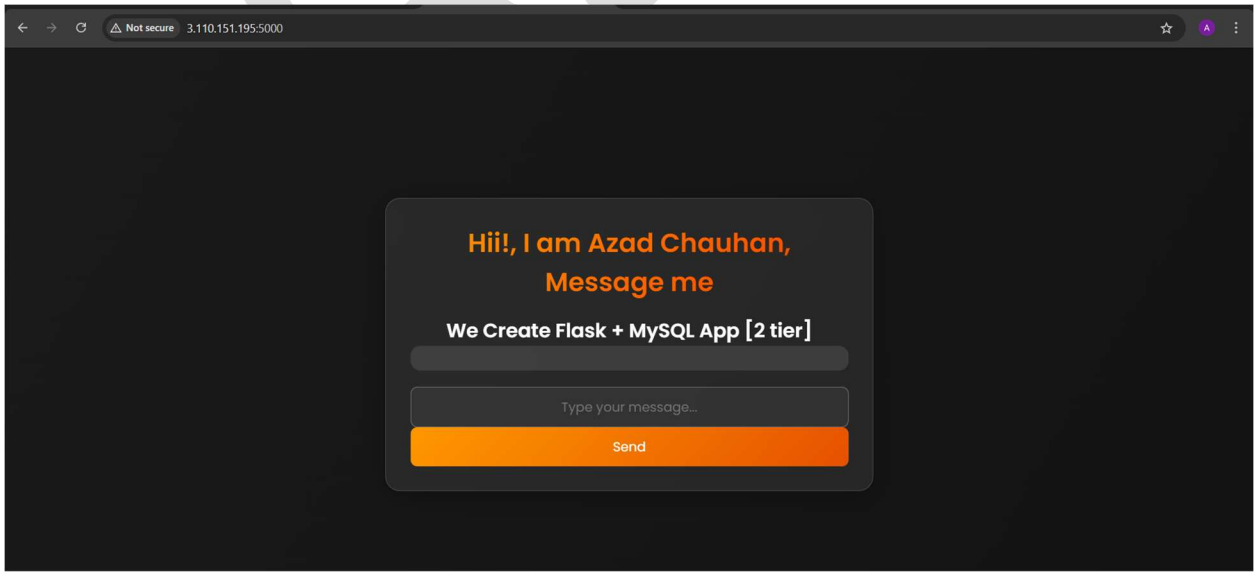
Step 4: Verify

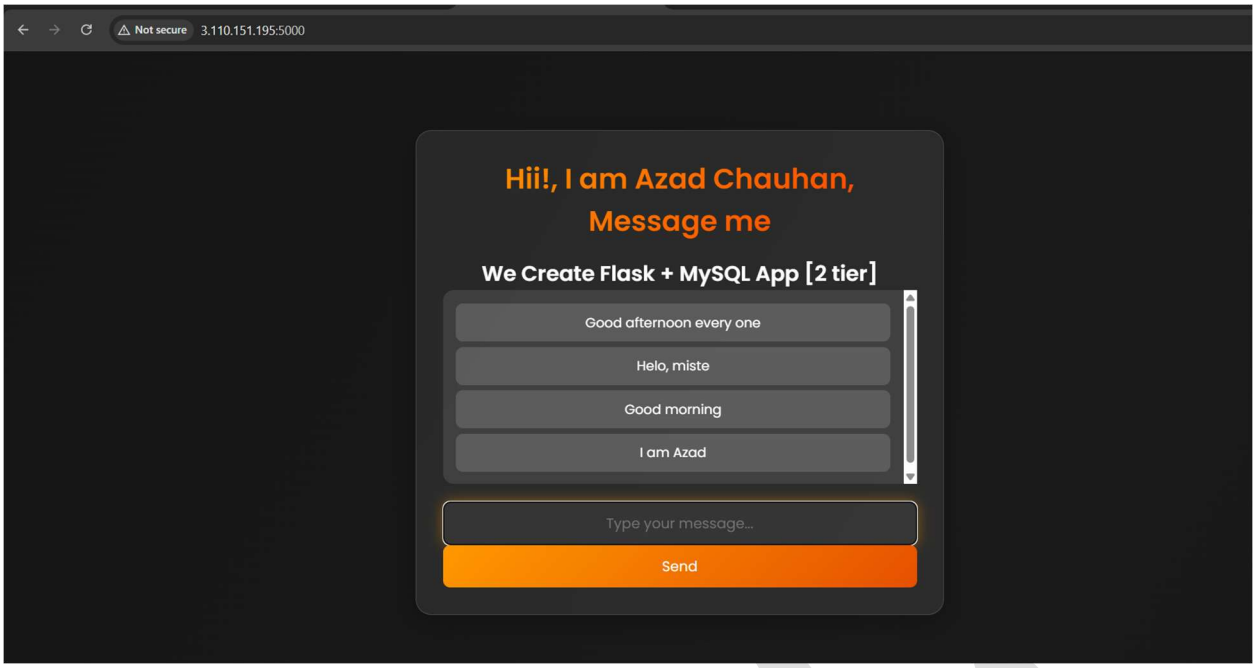
SHOW TABLES;

```
Error response from daemon: No such container: two-tier-app-deployment_mysql_1
ubuntu@ip-172-31-9-89:~/Two-Tier-App-Deployment$ docker exec -it mysql mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 5
Server version: 5.7.44 MySQL Community Server (GPL)

Copyright (c) 2000, 2023, 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> use mydb;
Database changed
mysql> CREATE TABLE IF NOT EXISTS messages (
    ->     id INT AUTO_INCREMENT PRIMARY KEY,
    ->     message TEXT
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> SHOW TABLES;
+----------------+
| Tables_in_mydb |
+----------------+
| messages       |
+----------------+
1 row in set (0.00 sec)

mysql> select * from messages;
Empty set (0.01 sec)

mysql>
mysql>
```

Now open your app in browser

👉 Go to: http://3.110.151.195:5000





Data is stored in backed