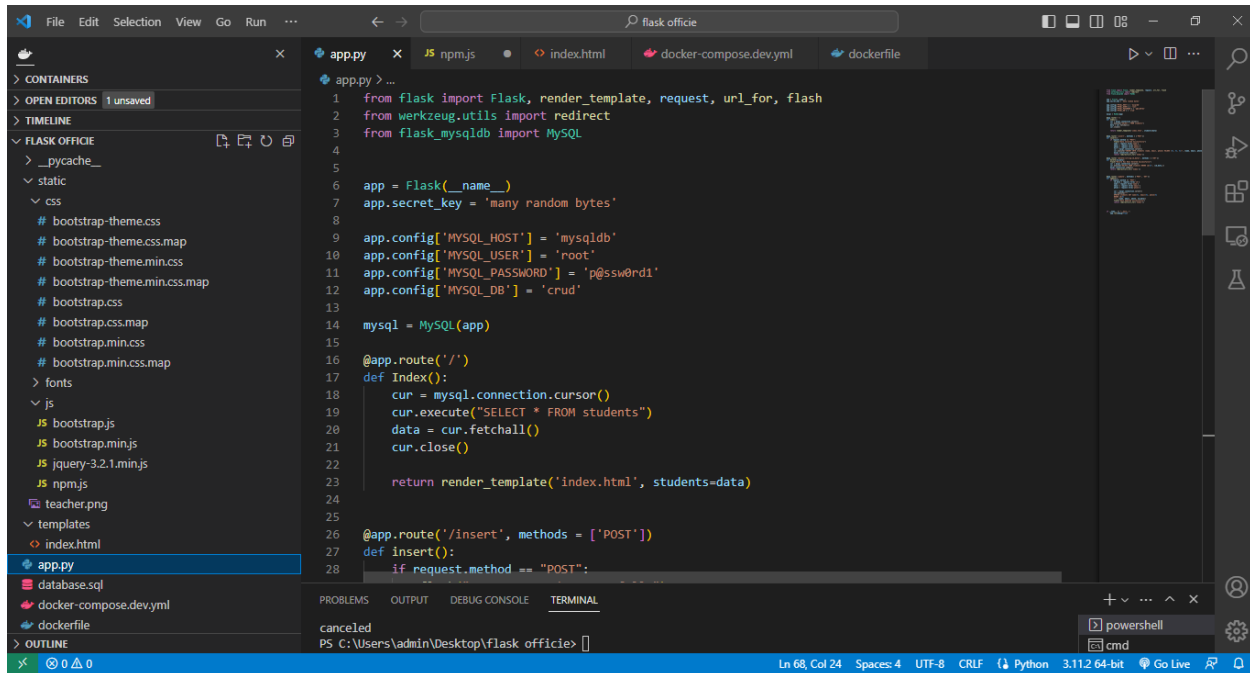


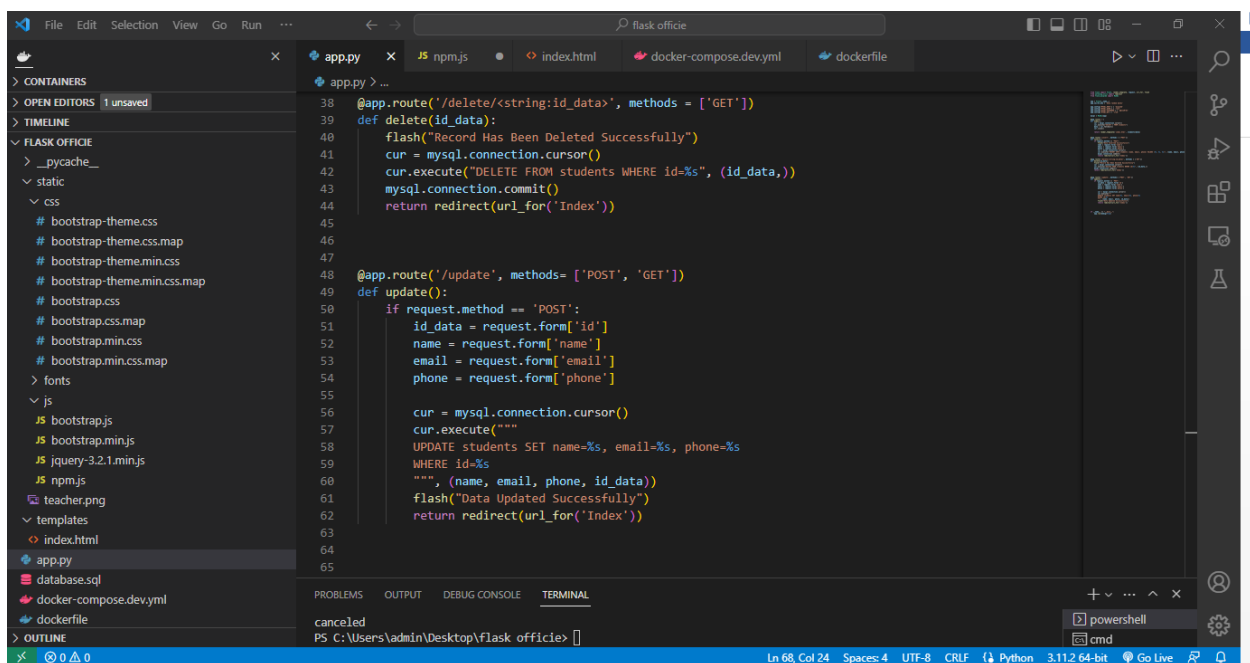
FLASK CRUD OPERATION ON MYSQL USING DOCKER COMPOSE

Step 1: creating app.py that is flask application



The screenshot shows the VS Code editor with the file explorer on the left displaying a project structure for 'FLASK OFFICE'. The main editor window shows the code for `app.py`. The code imports Flask, Werkzeug, and MySQLdb, configures the application with a secret key and database settings, and defines two routes: a GET route for the index and a POST route for inserting data into a 'students' table.

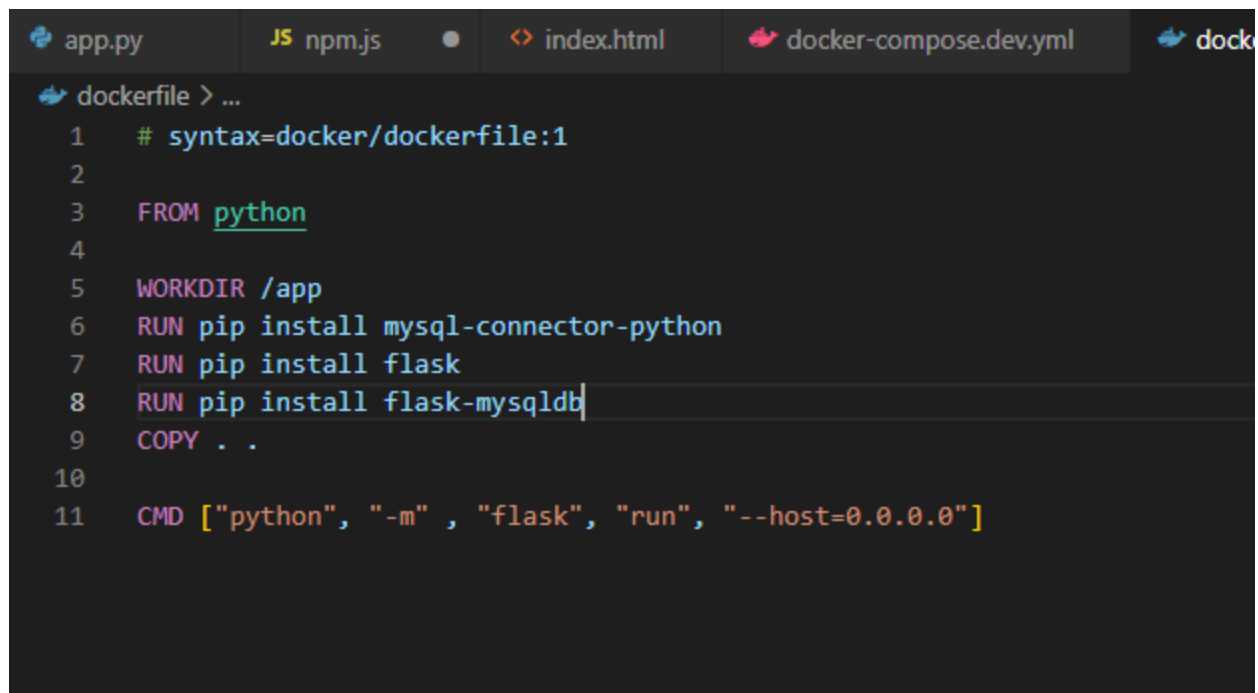
```
1 from flask import Flask, render_template, request, url_for, flash
2 from werkzeug.utils import redirect
3 from flask_mysql import MySQL
4
5
6 app = Flask(__name__)
7 app.secret_key = 'many random bytes'
8
9 app.config['MYSQL_HOST'] = 'mysql'
10 app.config['MYSQL_USER'] = 'root'
11 app.config['MYSQL_PASSWORD'] = 'p@ssw0rd1'
12 app.config['MYSQL_DB'] = 'crud'
13
14 mysql = MySQL(app)
15
16 @app.route('/')
17 def Index():
18     cur = mysql.connection.cursor()
19     cur.execute("SELECT * FROM students")
20     data = cur.fetchall()
21     cur.close()
22
23     return render_template('index.html', students=data)
24
25
26 @app.route('/insert', methods = ['POST'])
27 def insert():
28     if request.method == "POST":
```



This screenshot continues the code from the previous one, showing the DELETE and UPDATE routes. The DELETE route handles GET requests to delete a record by ID. The UPDATE route handles both POST and GET requests to update a record's name, email, and phone number.

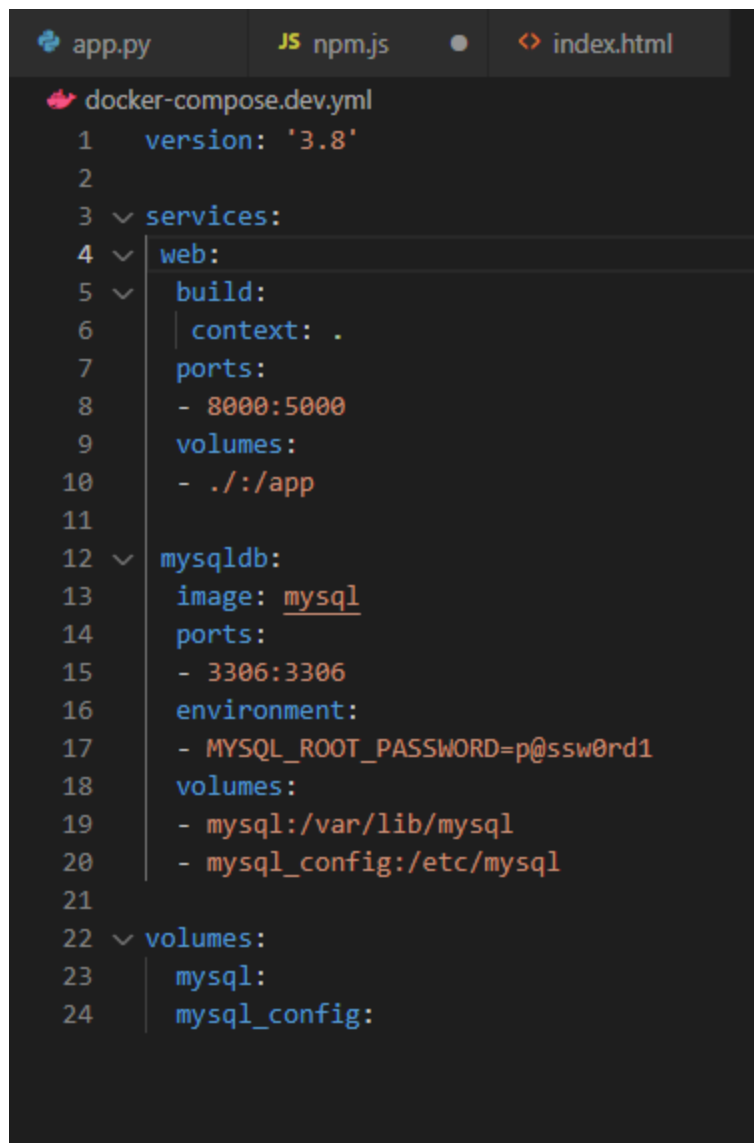
```
38 @app.route('/delete/<string:id_data>', methods = ['GET'])
39 def delete(id_data):
40     flash("Record Has Been Deleted Successfully")
41     cur = mysql.connection.cursor()
42     cur.execute("DELETE FROM students WHERE id=%s", (id_data,))
43     mysql.connection.commit()
44     return redirect(url_for('Index'))
45
46
47
48 @app.route('/update', methods = ['POST', 'GET'])
49 def update():
50     if request.method == "POST":
51         id_data = request.form['id']
52         name = request.form['name']
53         email = request.form['email']
54         phone = request.form['phone']
55
56         cur = mysql.connection.cursor()
57         cur.execute("""
58             UPDATE students SET name=%s, email=%s, phone=%s
59             WHERE id=%s
60             """, (name, email, phone, id_data))
61         flash("Data Updated Successfully")
62         return redirect(url_for('Index'))
63
64
65
```

Step 2: Creating Docker file



```
app.py JS npm.js index.html docker-compose.dev.yml dock
dockerfile > ...
1 # syntax=docker/dockerfile:1
2
3 FROM python
4
5 WORKDIR /app
6 RUN pip install mysql-connector-python
7 RUN pip install flask
8 RUN pip install flask-mysqldb
9 COPY . .
10
11 CMD ["python", "-m", "flask", "run", "--host=0.0.0.0"]
```


Step 3: Creating docker compose yaml file



```
docker-compose.dev.yml
1  version: '3.8'
2
3  services:
4  web:
5    build:
6      context: .
7    ports:
8      - 8000:5000
9    volumes:
10     - ./:/app
11
12  mysql:
13    image: mysql
14    ports:
15      - 3306:3306
16    environment:
17      - MYSQL_ROOT_PASSWORD=p@ssw0rd1
18    volumes:
19      - mysql:/var/lib/mysql
20      - mysql_config:/etc/mysql
21
22  volumes:
23    mysql:
24    mysql_config:
```

Step 4: create database on mysql in docker


<



flaskofficie-mysqldb-1

[mysql](#)

520d74cc32f3



[3306:3306](#)



Logs

Inspect

Terminal

Files

Stats

```
sh-4.4# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 8.0.33 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> show databases;
+-----+
| Database          |
+-----+
| crud               |
| information_schema |
| inventory          |
| mysql              |
| performance_schema |
| sys                |
+-----+
6 rows in set (0.01 sec)

mysql> 
```

Step 5: Run docker compose containers

```

✓ Container flaskofficie-mysqldb-1 Stopped 2.7s
PS C:\Users\admin\Desktop\flask officie> docker compose -f docker-compose.dev.yml up --build
[+] Building 4.6s (17/17) FINISHED
=> [internal] load build definition from dockerfile 0.1s
=> => transferring dockerfile: 32B 0.0s
=> [internal] load .dockerignore 0.1s
=> => transferring context: 2B 0.0s
=> resolve image config for docker.io/docker/dockerfile:1 2.0s
=> [auth] docker/dockerfile:pull token for registry-1.docker.io 0.0s
=> CACHED docker-image://docker.io/docker/dockerfile:1@sha256:39b85bbfa7536a5feceb7372a0817649ecb2724562a38 0.0s
=> [internal] load build definition from dockerfile 0.0s
=> [internal] load .dockerignore 0.0s
=> [internal] load metadata for docker.io/library/python:latest 1.4s
=> [auth] library/python:pull token for registry-1.docker.io 0.0s
=> [1/6] FROM docker.io/library/python@sha256:f7382f4f9dbc51183c72d621b9c196c1565f713a1fe40c119d215c961fa22 0.0s
=> [internal] load build context 0.1s
=> => transferring context: 12.25kB 0.0s
=> CACHED [2/6] WORKDIR /app 0.0s
=> CACHED [3/6] RUN pip install mysql-connector-python 0.0s
=> CACHED [4/6] RUN pip install flask 0.0s
=> CACHED [5/6] RUN pip install flask-mysqldb 0.0s
=> [6/6] COPY . . 0.1s
=> exporting to image 0.2s
=> => exporting layers 0.1s
=> => writing image sha256:64f329719c928e5414039f15d34c6c03d062bbca8770e3046111da4714c1071c 0.0s
=> => naming to docker.io/library/flaskofficie-web 0.0s
[+] Running 1/0
✓ Container flaskofficie-mysqldb-1 Created 0.0s
[+] Running 1/2askofficie-web-1 Recreate 0.0s

```

The screenshot shows the Docker Desktop application. On the left is a sidebar with navigation options: Containers, Images, Volumes, Dev Environments, Learning Center, Extensions, Disk usage, and Add Extensions. The main panel is titled 'Containers' and displays a table of running containers. The table has columns for Name, Image, Status, Port(s), Last started, and Actions. Three containers are listed: 'flaskofficie' (Running (2/2)), 'web-1' (Running, port 8000:5000), and 'mysqldb-1' (Running, port 3306:3306).

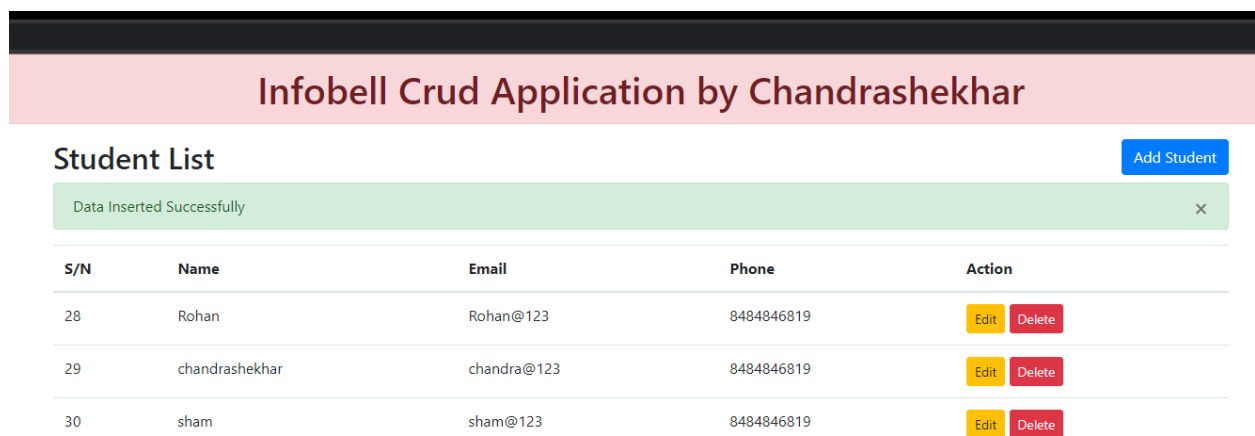
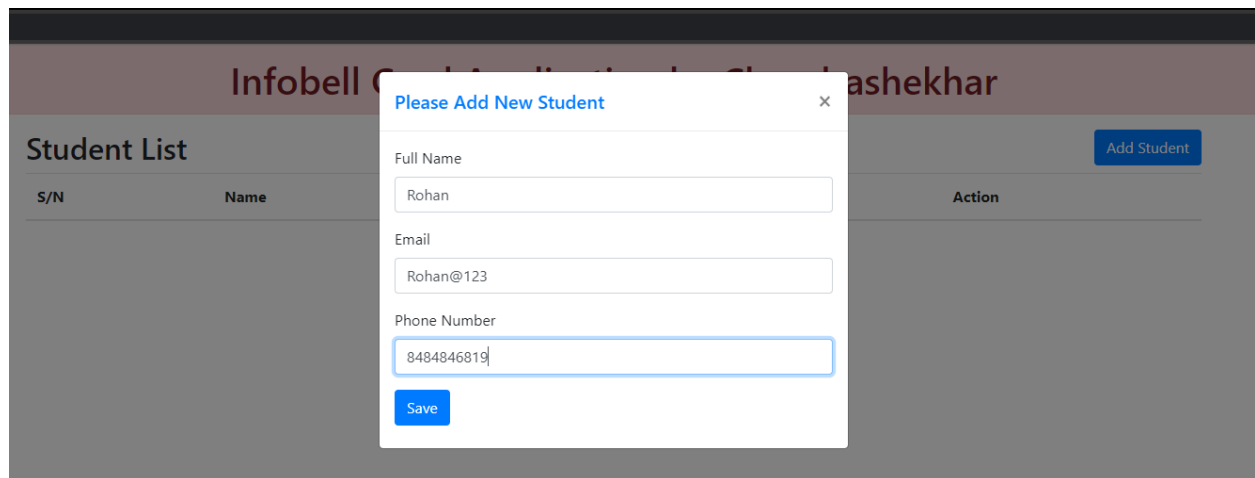
Name	Image	Status	Port(s)	Last started	Actions
flaskofficie	-	Running (2/2)		2 minutes ago	[Stop] [Restart] [Refresh]
web-1 85423e8560cb	flaskofficie-web	Running	8000:5000	2 minutes ago	[Stop] [Restart] [Refresh]
mysqldb-1 520d74cc32f9	mysql	Running	3306:3306	2 minutes ago	[Stop] [Restart] [Refresh]

Step 6: accessing localhost:8000

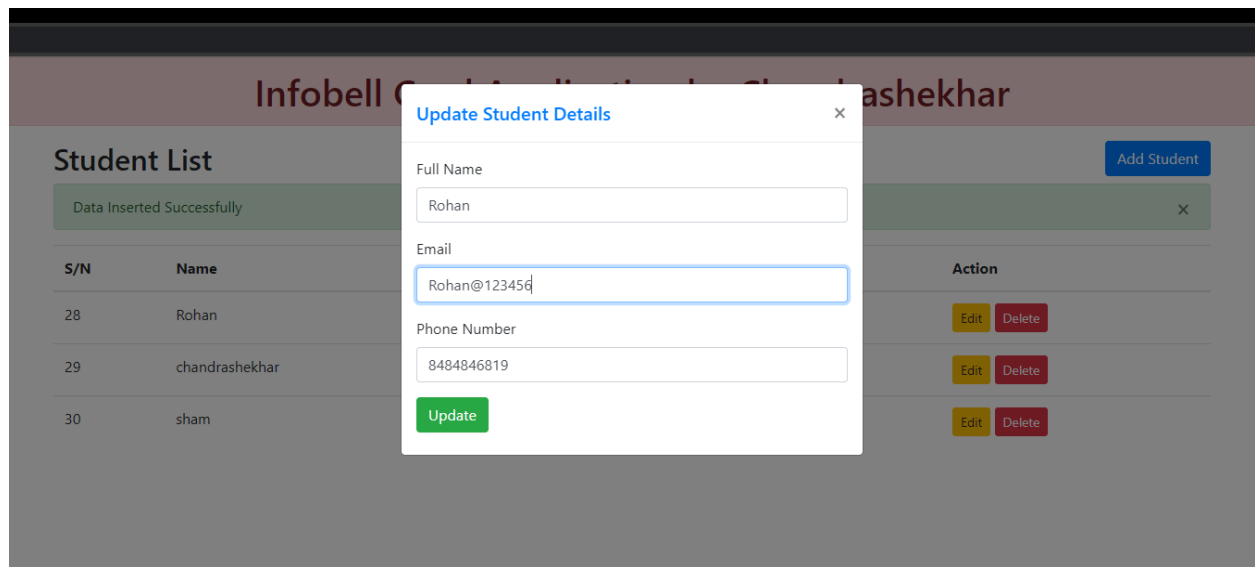
The screenshot shows a web browser window with the address bar set to 'localhost:8000'. The page title is 'Infobell Crud Application by Chandrashekhar'. Below the title is a 'Student List' section with an 'Add Student' button. A table with columns S/N, Name, Email, Phone, and Action is visible below the button.

S/N	Name	Email	Phone	Action
-----	------	-------	-------	--------

Step 7: inserting data to database



Step 8: Editing details



Step 9: Deleting data

