Message System

Frameworks & Images

- Application consists of 2 Docker containers, one holding the web server and one holding the MYSQL database. The Web server is built from the python 3.8-slime-buster base image. It's then deployed using the Flask web framework. The database is built from the mysql base image.

Deployment

- First you must initialize 2 volumes for the mysql server, one called mysql and one called mysql_config, using the following docker commans:

```
docker volume create mysql
docker volume create mysql_config
```

- After creating the volumes, you must create a docker network for the containers to run on using the following commands:

```
docker network create msgnet
```

- With the network name that we will connect to being "msgnet"

- After we have created the volume and netwrok, navigate to the directory where the project is located in and enter the following command to run the application:

```
docker-compose -f docker-compose.msg.yml up --build
```

- The -f flag tells docker what compose configuration file to use. The up flag starts the containers with --build signaling to build the image for each container even if it exists.

Use

- Clients connect to the Flask server via a web browser and enter messages. The messages are then logged into the database under "inputs" table. After messages are logged, the server routes back to the homepage where the newly added messages are automatically loaded onto the page. The server is mapped to localhost:5000. In a browser of your choice, enter localhost:5000 and go to it. Enter a message in the text box and click the submit button. The page should update with your message being displayed beneath the text box.

Authors

Jarod Reese

jarodree@buffalo.edu

Acknowledgments

- Docker-Docs-Compose
- Docker-Docs-Up
- Docker-Docs-Python
- Flask-Docs-MYSQL
- Flask-Docs