

CSE 486-DISTRIBUTED SYSTEM

Emulating PUB/SUB Distributed System Using Docker Container

README Phase 3

Ankit Sigroha

Rohit Balasayee

11.12.2018

Project 2

Phase 3:

The folder for phase 2 contain following files:-

1. Dockerfile
2. phase3.py
3. Requirements.txt
4. Templates folder-> P2.html
5. Docker-compose.yml
6. Report.pdf

The steps are as follows:-

1. Go to the directory and folder in which the files of applications are stored.
2. Open the command line or terminal and build the image of Dockerfile using the command “docker-compose build”

```
rohitb@rohitb-G7-7588: ~/PycharmProjects/Project2_rbalasay_ankitsig_P3_Docs
rohitb@rohitb-G7-7588:~/PycharmProjects/Project2_rbalasay_ankitsig_P3_Docs$ docker-compose build
db uses an image, skipping
Building phase3
Step 1/11 : FROM ubuntu:latest
--> e44c82cd15a
Step 2/11 : RUN apt-get update -y
--> Using cache
--> bad76da89b63
Step 3/11 : RUN apt-get install -y python-pip python-dev build-essential
--> Using cache
--> 52ab8cc2c0f0
Step 4/11 : RUN mkdir /home/phase2
--> Using cache
--> 82ac020d9ba3
Step 5/11 : COPY . /home/phase2
--> Using cache
--> 77dd08f88430
Step 6/11 : WORKDIR /home/phase2
--> Using cache
--> 4b4fb9fd9aff
Step 7/11 : RUN ls
--> Using cache
--> d01bef36eaba
Step 8/11 : RUN chmod +x phase3.py
--> Using cache
--> 592863e2aefd
Step 9/11 : RUN pip install -r requirements.txt
--> Using cache
--> 190883c4ddbc
Step 10/11 : ENTRYPOINT ["python"]
--> Using cache
--> fd0069048f00
Step 11/11 : CMD ["phase3.py"]
--> Using cache
--> eca1391295f1
Successfully built eca1391295f1
Successfully tagged project2_rbalasay_ankitsig_p3_docs_phase3:latest
rohitb@rohitb-G7-7588:~/PycharmProjects/Project2_rbalasay_ankitsig_P3_Docs$
```

3. Now set the images to run by setting the scale to 2 using the command “docker-compose up -d --scale phase3=2”

```
rohitb@rohitb-G7-7588:~/PycharmProjects/Project2_rbalasay_ankitsig_P3_Docs$ docker-compose up -d --scale phase3=2
Creating network "project2_rbalasay_ankitsig_p3_docs_default" with the default driver
Creating project2_rbalasay_ankitsig_p3_docs_db_1 244fb2a829ad ... done
WARNING: The "phase3" service specifies a port on the host. If multiple containers for this service are created on a single host, the port will clash.
Creating project2_rbalasay_ankitsig_p3_docs_phase3_1 1729ee58d545 ... done
Creating project2_rbalasay_ankitsig_p3_docs_phase3_2 60e723763091 ... done
rohitb@rohitb-G7-7588:~/PycharmProjects/Project2_rbalasay_ankitsig_P3_Docs$ docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS                               NAMES
e641e34481a2   project2_rbalasay_ankitsig_p3_docs_phase3   "python phase3.py"       32 seconds ago   Up 14 seconds   0.0.0.0:5000->5000/tcp             project2_rbalasay_ankitsig_p3_docs_phase3_1_c8f9
f972e582      project2_rbalasay_ankitsig_p3_docs_phase3   "python phase3.py"       32 seconds ago   Up 15 seconds   0.0.0.0:5001->5000/tcp             project2_rbalasay_ankitsig_p3_docs_phase3_2_488a
cc08bfe0      mongo:4.0.4                                "docker-entrypoint.s..." 43 seconds ago   Up 32 seconds   0.0.0.0:27017->27017/tcp           project2_rbalasay_ankitsig_p3_docs_db_1_1bbb4431
rohitb@rohitb-G7-7588:~/PycharmProjects/Project2_rbalasay_ankitsig_P3_Docs$
```

4. Go to a browser and type “https://localhost:5000” for the first container and type “https://localhost:5001” for the subscribers or the second container . The application will run prompting you to enter the values.
5. Add/Enter the Topic name and the publisher name. Similarly keep adding the subscriber name and the corresponding topic name as much as you want.
6. Enter/Append the data to a topic in text bar and keep on publishing the data.

7. Whenever the user is done with add/appending PUB/SUB to the topics, hit NOTIFY button to print the data as an alert.
8. You will see all the data so entered in respective topics by the publishers.

The image shows two browser windows. The top window, at localhost:5000, has a light blue background and displays 'Hello there!' in yellow. It contains three sections for entering names and topics, each with a 'Submit' button. The bottom window, at localhost:5001, has a dark grey background. It features a 'Publish' button, a 'Notify' button, and a section for generating random publishers and subscribers. A modal alert box is open, displaying the message: 's1 received the message:[u' msg 1 from container 1']'.

localhost:5000

Hello there!

Enter Publisher and Topic Name

Enter Topic Name

Enter Publisher Name

Enter Subscriber and Topic Name

Enter Subscriber Name

Enter Topic Name

Enter Subscriber and Topic Name (topic from another container)

Enter Subscriber Name

localhost:5001

Click Button to notify

Press the button below to randomly generate publishers, topics and subscribers to Test

The number of publishers and Subscribers can be entered

Enter Number of Subscribers

s1 received the message:[u' msg 1 from container 1']

A screenshot of a web browser window with the address bar showing "localhost:5001". The browser interface includes back, forward, and home buttons, along with a 110% zoom level and a star icon for bookmarks. The main content area has a light blue background and contains the following form elements:

- Enter Subscriber and Topic Name (topic from another container)**
 - Enter Subscriber Name: A text input field containing "s1".
 - Enter Topic Name: A text input field containing "t1".
 - Submit: A button located below the topic name input field.
- Enter data to be published: A large, empty text area.
- Enter Publisher Name to publish Data To: A label at the bottom of the form.

To notify just click the NOTIFY button and it will review the result on the UI as asked whether the inter-container or intra-container.