Course Project – End Report

COMP.SE.140 COMP.SE.140 Continuous Development and Deployment – DevOps

Miika Luiro
252906

Features implemented

- All compulsory features
- /node-statistic -endpoint
- /queue-statistic -endpoint
- Application deployed to external cloud. Application is running in Google Cloud's Compute Engine. You can access the application from IP: 34.74.12.236

Description of the CI/CD pipeline

There are two different modes to use the CI/CD pipeline:
- Local Deployment mode
- Remote Deployment mode

Local Deployment mode uses only one virtual machine to host both Gitlab and the application itself. Local Deployment mode should not be used for production deployments because it is not good practice to use same VM as a CI/CD and application server. This mode was used during development of the project to keep things simple. (No need to start and connect multiple VMs)

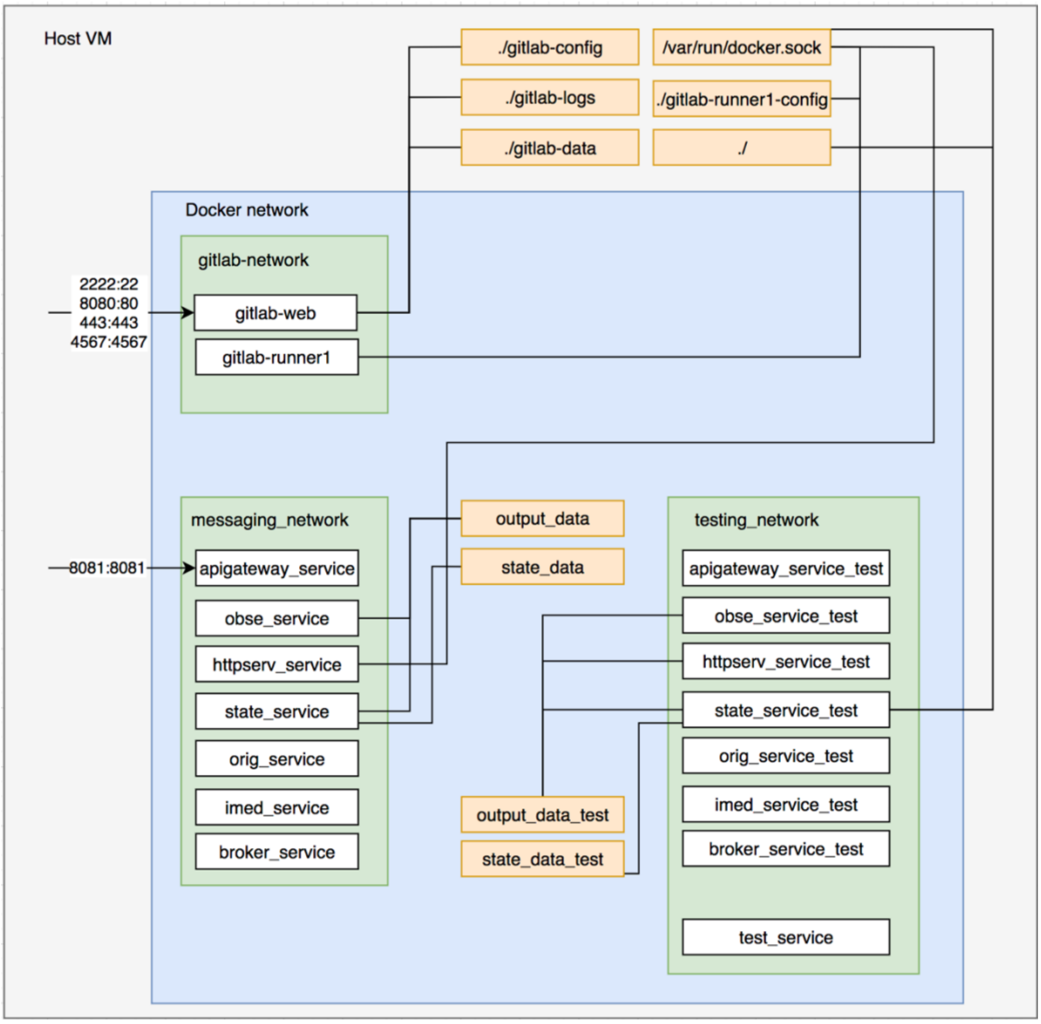

Image 1: Architecture of the CI/CD pipeline environment in Local Deployment mode

Remote Deployment mode can be used to deploy the application to remote virtual machine. This mode is suggested if you want to deploy the application to production use.

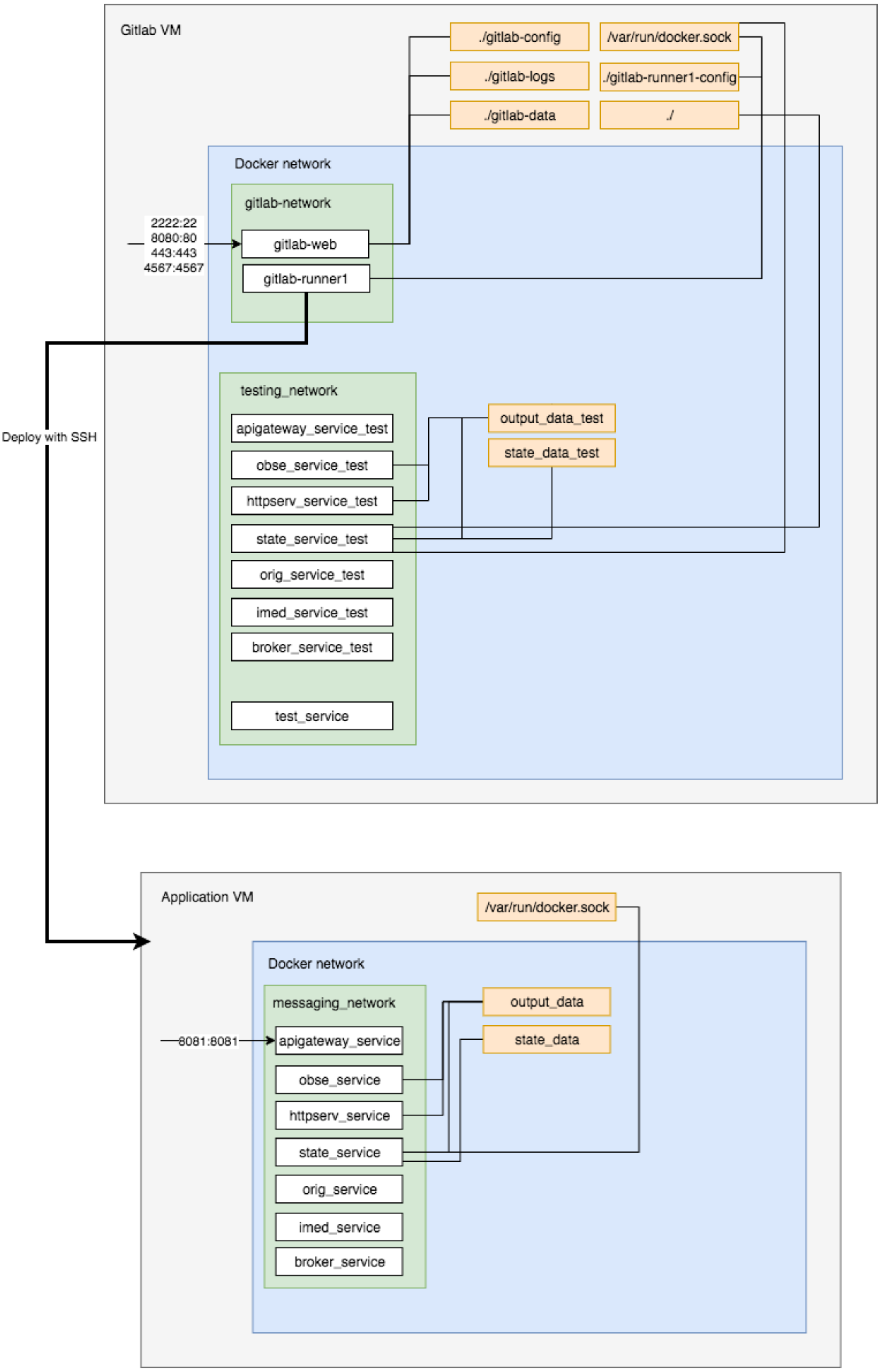

Image 1: Architecture of the CI/CD pipeline environment in Remote Deployment mode

Both modes work similarly except in the deploy stage of the pipeline. The pipeline starts when user pushes new changes to the Gitlab Web (gitlab-web container). Gitlab Web uses the gitlab-ci.yml-file to run the CI/CD pipeline. All jobs found in the gitlab-ci.yml -file are executed by a single Gitlab Runner (gitlab-runner1).

The pipeline is divided in three different stages: build, test, deploy. These stages are executed in sequence, so that there is only one stage running simultaneously.

In the build-stage the runner builds and starts test versions of the application containers. These new containers become siblings of the gitlab-runner1 -container. This is because the gitlab-runner1 uses the host virtual machine's Docker socket. If there are old test containers running for some reason, the build-stage takes care of shutting them down first. Build-stage is also responsible for building a separate test container that is used to run the tests.

In the test-stage the runner uses the containers started in the build-stage. A health check is done first to ensure that the system is ready to be tested. After that a test script is executed in a separate test container. This is a Python script which uses unittest-library to handle the orchestration and running of the test cases. If any of these tests fails, it stops the execution of the test-stage and marks the pipeline failed in the Gitlab Web. User can inspect the output of the tests from the Gitlab Web's CI/CD -feature.

In the deploy-stage the runner behaves differently depending on the mode used. If the Local Deployment mode is used, the runner builds and starts the application containers using the host virtual machine's Docker socket. If the Remote Deployment mode is used, the runner first uses ssh to connect to the remote server and uses its Docker to build and start the containers.

Instructions for setting up the system

Application was developed and is ensured to work with Ubuntu 20.04.1 LTS virtual machine with memory of 4096 MB and disk space of 40 GB.

It is recommended to use a fresh virtual machine to test the system. You should have Docker and Docker compose installed on this machine. If not, you can install them following these links:

- Docker: https://docs.docker.com/engine/install/ubuntu/#install-using-the-convenience-script
- Docker Compose: https://docs.docker.com/compose/install/

Follow these steps to setup the system:

1. Clone the project from Github:
   ```
   git clone https://github.com/mikitane/DevOpsProject.git
   ```

2. Go to directory: ./DevOpsProject/GitlabService
   ```
   cd ./DevOpsProject/GitlabService
   ```

3. Build and start the Gitlab Docker containers
   ```
   docker-compose up --build -d
   ```

4. Wait for a few minutes for Gitlab services to start

5. Go to Gitlab Web in your browser (http://localhost:8080). If your virtual machine does not have a browser, the easiest way is to forward the port 8080 with the host machine your virtual machine is running on and use its browser.

6. Set a password for the root user in Gitlab.

7. Login in as root user with the password you just set.

8. In Gitlab, click “New project”

9. Name the new project “DevOpsProject”

10. Click “Create project”

11. In your VM, ensure that you are in the project repository and run following commands: (use credentials created in step 6)
    ```
    git remote add gl-origin http://127.0.0.1/root/devopsproject.git
    git push -u gl-origin –all
    ```

12. The project is now pushed to your local Gitlab server. You can inspect the CI/CD pipeline by opening DevOpsProject in Gitlab and by going to CI/CD-page from the sidebar. If everything is set up correctly, the pipeline should be stuck because there is no runner to execute the jobs in the pipeline.

13. You need a registration token to register the Gitlab Runner with Gitlab Web. In the browser with the project open, go to Settings -> CI/CD -> Runners (Expand) -> Copy the registration token under “Set up a specific Runner manually”

14. In your VM, run the following script with the registration token you just got. This script registers your Gitlab Runner with Gitlab Web. (Same script is also located in GitlabService/gitlab-runner-register.sh).

    ```
    docker exec -it gitlab-runner1 \
      gitlab-runner register \
        --non-interactive \
        --registration-token <your-registration-token>\
        --locked=false \
        --description docker-stable \
        --url http://gitlab-web \
        --executor docker \
        --docker-image docker:stable \
        --docker-volumes "/var/run/docker.sock:/var/run/docker.sock" \
    ```

```
   --docker-volumes
"/etc/ssh/devops_deploy_key.txt:/etc/ssh/devops_deploy_key.txt" \
   --docker-network-mode gitlab-network
```

15. CI/CD-pipeline should now be running and will build, test and deploy the application to the local environment after couple of minutes.

16. Push new changes to your local Gitlab and CI/CD-pipeline will deploy those changes automatically
    ```
    git push gl-origin
    ```

After following these steps, the production containers are running and APIGatewayService is exposed from the port 8081. You can test the application by requesting the state:

```
curl localhost:8081/state
```

If you want to deploy the application to a remote server, you have to follow these steps after your CI/CD pipeline is running.

1. Create SSH keys. (e.g. with ssh-keygen) and store the public key to your remote server. Save the private key to a file located in: /etc/ssh/devops_deploy_key.txt

2. Master branch is setup with local deployment. You can use the branch master_remote which is configured to deploy the application to the remote environment or you can change the DEPLOY_ENV variable in .gitlab-ci.yml to remote.

3. Configure the variables starting with REMOTE_ in deploy.sh -file to match your remote server.

4. Your remote server should be accessible by ssh and have Docker and Docker Compose installed.

5. Clone the project from Github to the remote server. The project repository should be located in the same path as indicated by REMOTE_PROJECT_PATH variable in deploy.sh.

6. Push changes to your Gitlab server and it should deploy the application to the remote server.
   ```
   git push gl-origin
   ```

Public API documentation

The application exposes the port 8081 to host machine. This port is used to make requests to APIGatewayService which is responsible for forwarding the requests from the user to the correct internal service and returning the response.

All compulsory features are served from these endpoints:

/messages GET
Returns the messages registered by ObseService.

/state GET
Returns the state of the application

/state PUT
Takes a new state as a payload and sets application to that state. Possible values are PAUSED, RUNNING, INIT, SHUTDOWN.

RUNNING = OrigService sends new messages every 3 seconds
PAUSED = No new messages are sent
INIT = Initializes the application, all previous data is cleared. Application changes automatically to RUNNING state after a while.
SHUTDOWN = Stops all the containers that the application uses.

/run-log GET
Returns logs of state changes.
Example log:

```
2020-12-03T17:35:43.382Z: INIT
2020-12-03T17:36:01.260Z: RUNNING
```

/node-statistic GET
Returns statistics of RabbitMQ nodes in JSON format. Response includes following data:

fd_used = Used file descriptors.
disk_free = Disk free space in bytes.
mem_used = Memory used in bytes.
processors = Number of cores detected and usable by Erlang.
io_read_avg_time = Average wall time (milliseconds) for each disk read operation in the last statistics interval.

/queue-statistic GET

Returns statistics of RabbitMQ queues in JSON format. Response includes following data:

queue = Name of the queue
message_delivery_rate = How much the count of messages delivered has changed per second in the most recent sampling interval.
messages_publishing_rate = How much the count of messages published has changed per second in the most recent sampling interval.
messages_delivered_recently = Count of messages delivered
message_published_lately = Count of messages published lately

Explanation of the main files and directories in the codebase

docker-compose.yml and docker-compose.test.yml
There are different docker-compose -files for managing the application in production and testing environment. Separate files were necessary because the test and production containers are executed in the same host machine in the local deployment mode. If these containers would be executed in different machines, only a single file would be needed.

.gitlab-ci.yml
This file is used to instruct Gitlab on how to build, test and deploy the application from the CI/CD pipeline.

deploy.sh
Helper script for deploying the application to local and remote environments

APIGatewayService
This service is the only service that is exposed to outside networks. This service is responsible for forwarding the requests from the user to the correct service.

BrokerService
This service is responsible for setting up the RabbitMQ server

GitlabService
This service is responsible for setting up the Gitlab Web and Gitlab Runner containers. These containers are managed with separate docker-compose.yml which is located in service's directory.

HttpServService
This service has currently only one responsibility: reading the message logs produced by ObseService and sending those back to the client.

ImedService
This service is responsible for handling two different tasks: listening for messages with my.o routing key and publishing a modified message with my.i routing key.

ObseService
This service is responsible for listening for messages published with routing key that matches my.* wild card key.
When message is received it is stored to the log file

OrigService

OrigService is responsible for publishing
new messages every 3 seconds with my.o routing key
The service publishes new messages only when the
application is in RUNNING state

StateService
StateService is responsible for managing the state of
the application and statistics from RabbitMQ.
The service serves this data to clients
from HTTP server.

Main learnings and worst difficulties

This project was an excellent way to learn the fundamentals of many concepts. Especially CI/CD, Gitlab, Docker, RabbitMQ. After completing this project, I have strong feeling that I could build up a CI/CD pipeline for any service.

By far, most of the time used in this project was used to learn how to use Docker properly with Gitlab Runner. Docker-In-Docker -concept was a bit hard to get at first because there was not too much material in internet about using it with Gitlab. Maybe there could have been a bit more teaching about the concept during lectures.

In total I used about 50 hours with this project. I used about 40 hours with the compulsory parts, mainly because of fighting with Docker-In-Docker.

Example logs of successful build

```
[0KRunning with gitlab-runner 13.6.0 (8fa89735)
[0;m[0K  on docker-stable TAXGxvh7
[0;msection_start:1607016847:prepare_executor
[0K[0K[36;1mPreparing the "docker" executor[0;m
[0;m[0KUsing Docker executor with image docker/compose:latest ...
[0;m[0KPulling docker image docker/compose:latest ...
[0;m[0KUsing docker image
sha256:c3e188a6b38fa7f54cac9963e11b049f7701b8a6354962218b6bbab0fba07bdf for
docker/compose:latest with digest
docker/compose@sha256:b60a020c0f68047b353a4a747f27f5e5ddb17116b7b018762edfb
6f7a6439a82 ...
[0;msection_end:1607016850:prepare_executor
[0Ksection_start:1607016850:prepare_script
[0K[0K[36;1mPreparing environment[0;m
[0;mRunning on runner-taxgxvh7-project-2-concurrent-0 via gitlab-runner1...
section_end:1607016850:prepare_script
[0Ksection_start:1607016850:get_sources
[0K[0K[36;1mGetting source from Git repository[0;m
[0;m[32;1mFetching changes with git depth set to 50...[0;m
Reinitialized existing Git repository in /builds/root/devopsproject/.git/
[32;1mChecking out 44494cd9 as master_remote...[0;m

[32;1mSkipping Git submodules setup[0;m
section_end:1607016851:get_sources
[0Ksection_start:1607016851:step_script
```

```
[0K[0K[36;1mExecuting "step_script" stage of the job script[0;m
[0;m[32;1m$ apk update && apk add openssh[0;m
fetch http://dl-
cdn.alpinelinux.org/alpine/v3.11/main/x86_64/APKINDEX.tar.gz
fetch http://dl-
cdn.alpinelinux.org/alpine/v3.11/community/x86_64/APKINDEX.tar.gz
v3.11.6-209-g19f4c5d312 [http://dl-cdn.alpinelinux.org/alpine/v3.11/main]
v3.11.6-212-g3a84324cad [http://dl-
cdn.alpinelinux.org/alpine/v3.11/community]
OK: 11275 distinct packages available
(1/9) Installing openssh-keygen (8.1_p1-r0)
(2/9) Installing ncurses-terminfo-base (6.1_p20200118-r4)
(3/9) Installing ncurses-libs (6.1_p20200118-r4)
(4/9) Installing libedit (20191211.3.1-r0)
(5/9) Installing openssh-client (8.1_p1-r0)
(6/9) Installing openssh-sftp-server (8.1_p1-r0)
(7/9) Installing openssh-server-common (8.1_p1-r0)
(8/9) Installing openssh-server (8.1_p1-r0)
(9/9) Installing openssh (8.1_p1-r0)
Executing busybox-1.31.1-r9.trigger
OK: 11 MiB in 23 packages
[32;1m$ ls -l[0;m
total 48
drwxrwxrwx    2 root     root          4096 Dec  3 17:01 APIGatewayService
drwxrwxrwx    2 root     root          4096 Dec  3 17:01 BrokerService
drwxrwxrwx    2 root     root          4096 Dec  3 17:01 GitlabService
drwxrwxrwx    2 root     root          4096 Dec  3 17:01 HttpServService
drwxrwxrwx    2 root     root          4096 Dec  3 17:01 ImedService
drwxrwxrwx    2 root     root          4096 Dec  3 17:01 ObseService
drwxrwxrwx    2 root     root          4096 Dec  3 17:01 OrigService
drwxrwxrwx    2 root     root          4096 Dec  3 17:01 StateService
-rw-rw-rw-    1 root     root           926 Dec  3 17:30 deploy.sh
-rw-rw-rw-    1 root     root          2367 Dec  3 17:01 docker-
compose.test.yml
-rw-rw-rw-    1 root     root          2185 Dec  3 17:01 docker-compose.yml
-rw-rw-rw-    1 root     root             0 Dec  3 17:01 readme.md
drwxrwxrwx    2 root     root          4096 Dec  3 17:01 tests
[32;1m$ pwd[0;m
/builds/root/devopsproject
[32;1m$ docker-compose -f docker-compose.test.yml down -v[0;m
Found orphan containers (devopsproject_apigateway_service_devops_prod_1,
devopsproject_orig_service_devops_prod_1,
devopsproject_obse_service_devops_prod_1,
devopsproject_state_service_devops_prod_1,
devopsproject_broker_service_devops_prod_1,
devopsproject_imed_service_devops_prod_1,
devopsproject_httpserv_service_devops_prod_1) for this project. If you
removed or renamed this service in your compose file, you can run this
command with the --remove-orphans flag to clean it up.
Removing network testing_network
Network testing_network not found.
Removing volume devopsproject_output_data_test
Volume devopsproject_output_data_test not found.
Removing volume devopsproject_state_data_test
Volume devopsproject_state_data_test not found.
[32;1m$ docker build ./tests -t test_service:latest[0;m
Sending build context to Docker daemon   12.8kB

Step 1/5 : FROM python:3.9
 ---> 4655c53edd4e
Step 2/5 : COPY requirements.txt .
```

```
 ---> Using cache
 ---> 891d930366bd
Step 3/5 : RUN pip install -r requirements.txt
 ---> Using cache
 ---> 3ec77b59a0fd
Step 4/5 : COPY healthcheck.py .
 ---> Using cache
 ---> 41a9449306a8
Step 5/5 : COPY test_apigateway.py .
 ---> Using cache
 ---> 15187007f647
Successfully built 15187007f647
Successfully tagged test_service:latest
[32;1m$ docker-compose -f docker-compose.test.yml up --build --force-
recreate -d[0;m
Creating network "testing_network" with the default driver
Creating volume "devopsproject_output_data_test" with default driver
Creating volume "devopsproject_state_data_test" with default driver
Found orphan containers (devopsproject_obse_service_devops_prod_1,
devopsproject_broker_service_devops_prod_1,
devopsproject_orig_service_devops_prod_1,
devopsproject_state_service_devops_prod_1,
devopsproject_apigateway_service_devops_prod_1,
devopsproject_httpserv_service_devops_prod_1,
devopsproject_imed_service_devops_prod_1) for this project. If you removed
or renamed this service in your compose file, you can run this command with
the --remove-orphans flag to clean it up.
Building broker_service_devops_test
Step 1/1 : FROM rabbitmq:3.8-management
 ---> 263c941f71ea

Successfully built 263c941f71ea
Successfully tagged devopsproject_broker_service_devops_test:latest
Building orig_service_devops_test
Step 1/5 : FROM python:3.9
 ---> 4655c53edd4e
Step 2/5 : COPY requirements.txt .
 ---> Using cache
 ---> 959e0972c534
Step 3/5 : RUN pip install -r requirements.txt
 ---> Using cache
 ---> efd5059f736a
Step 4/5 : COPY main.py .
 ---> Using cache
 ---> 752d5f7ff2ca
Step 5/5 : CMD ["python", "main.py"]
 ---> Using cache
 ---> 479589642eb2

Successfully built 479589642eb2
Successfully tagged devopsproject_orig_service_devops_test:latest
Building imed_service_devops_test
Step 1/5 : FROM python:3.9
 ---> 4655c53edd4e
Step 2/5 : COPY requirements.txt .
 ---> Using cache
 ---> ddefd214469d
Step 3/5 : RUN pip install -r requirements.txt
 ---> Using cache
 ---> 3560242e45d2
Step 4/5 : COPY main.py .
```

```
 ---> Using cache
 ---> e432b42c4a06
Step 5/5 : CMD ["python", "main.py"]
 ---> Using cache
 ---> 506109e348a1

Successfully built 506109e348a1
Building obse_service_devops_test
Successfully tagged devopsproject_imed_service_devops_test:latest
Step 1/5 : FROM python:3.9
 ---> 4655c53edd4e
Step 2/5 : COPY requirements.txt .
 ---> Using cache
 ---> 959e0972c534
Step 3/5 : RUN pip install -r requirements.txt
 ---> Using cache
 ---> efd5059f736a
Step 4/5 : COPY main.py .
 ---> Using cache
 ---> 7bf00fa37c38
Step 5/5 : CMD ["python", "main.py"]
 ---> Using cache
 ---> 490409373653

Successfully built 490409373653
Successfully tagged devopsproject_obse_service_devops_test:latest
Building httpserv_service_devops_test
Step 1/3 : FROM node:14
 ---> 4c7835fa737e
Step 2/3 : ADD server.js /
 ---> Using cache
 ---> 3903b9173e42
Step 3/3 : CMD ["node", "server.js"]
 ---> Using cache
 ---> 3063182ebab6

Successfully built 3063182ebab6
Successfully tagged devopsproject_httpserv_service_devops_test:latest
Building apigateway_service_devops_test
Step 1/3 : FROM node:14
 ---> 4c7835fa737e
Step 2/3 : ADD server.js /
 ---> Using cache
 ---> 3100f18f48d1
Step 3/3 : CMD ["node", "server.js"]
 ---> Using cache
 ---> f348ca36075c

Successfully built f348ca36075c
Successfully tagged devopsproject_apigateway_service_devops_test:latest
Building state_service_devops_test
Step 1/8 : FROM node:14
 ---> 4c7835fa737e
Step 2/8 : COPY --from=docker:latest /usr/local/bin/docker /usr/local/bin/
 ---> Using cache
 ---> 64279357136d
Step 3/8 : RUN curl -L
"https://github.com/docker/compose/releases/download/1.27.4/docker-compose-
$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
 ---> Using cache
 ---> 282afe52fab2
```

```
Step 4/8 : RUN chmod +x /usr/local/bin/docker-compose
 ---> Using cache
 ---> 5bd362f94ea5
Step 5/8 : RUN ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
 ---> Using cache
 ---> 7c29d9391b46
Step 6/8 : ADD rabbitmq-stats.js /
 ---> Using cache
 ---> 3769deb223d8
Step 7/8 : ADD server.js /
 ---> Using cache
 ---> 539e2607fa3b
Step 8/8 : CMD ["node", "server.js"]
 ---> Using cache
 ---> f7dec260ffe2

Successfully built f7dec260ffe2
Successfully tagged devopsproject_state_service_devops_test:latest
Creating devopsproject_broker_service_devops_test_1 ...
Creating devopsproject_imed_service_devops_test_1   ...
Creating devopsproject_orig_service_devops_test_1   ...
Creating devopsproject_apigateway_service_devops_test_1 ...
Creating devopsproject_obse_service_devops_test_1       ...
Creating devopsproject_state_service_devops_test_1      ...
Creating devopsproject_httpserv_service_devops_test_1   ...
[6A[2K
Creating devopsproject_imed_service_devops_test_1       ... [32mdone[0m
[6B[5A[2K
Creating devopsproject_orig_service_devops_test_1       ... [32mdone[0m
[5B[1A[2K
Creating devopsproject_httpserv_service_devops_test_1   ... [32mdone[0m
[1B[3A[2K
Creating devopsproject_obse_service_devops_test_1       ... [32mdone[0m
[3B[7A[2K
Creating devopsproject_broker_service_devops_test_1     ... [32mdone[0m
[7B[2A[2K
Creating devopsproject_state_service_devops_test_1      ... [32mdone[0m
[2B[4A[2K
Creating devopsproject_apigateway_service_devops_test_1 ... [32mdone[0m
[4Bsection_end:1607016859:step_script
[0K[32;1mJob succeeded
[0;m
```

Example logs of successful tests

```
[0KRunning with gitlab-runner 13.6.0 (8fa89735)
[0;m[0K  on docker-stable TAXGxvh7
[0;msection_start:1607016863:prepare_executor
[0K[0K[36;1mPreparing the "docker" executor[0;m
[0;m[0KUsing Docker executor with image docker/compose:latest ...
[0;m[0KPulling docker image docker/compose:latest ...
[0;m[0KUsing docker image
sha256:c3e188a6b38fa7f54cac9963e11b049f7701b8a6354962218b6bbab0fba07bdf for
docker/compose:latest with digest
docker/compose@sha256:b60a020c0f68047b353a4a747f27f5e5ddb17116b7b018762edfb
6f7a6439a82 ...
[0;msection_end:1607016866:prepare_executor
[0Ksection_start:1607016866:prepare_script
[0K[0K[36;1mPreparing environment[0;m
[0;mRunning on runner-taxgxvh7-project-2-concurrent-0 via gitlab-runner1...
section_end:1607016867:prepare_script
```

```
[0Ksection_start:1607016867:get_sources
[0K[0K[36;1mGetting source from Git repository[0;m
[0;m[32;1mFetching changes with git depth set to 50...[0;m
Reinitialized existing Git repository in /builds/root/devopsproject/.git/
[32;1mChecking out 44494cd9 as master_remote...[0;m

[32;1mSkipping Git submodules setup[0;m
section_end:1607016868:get_sources
[0Ksection_start:1607016868:step_script
[0K[0K[36;1mExecuting "step_script" stage of the job script[0;m
[0;m[32;1m$ apk update && apk add openssh[0;m
fetch http://dl-
cdn.alpinelinux.org/alpine/v3.11/main/x86_64/APKINDEX.tar.gz
fetch http://dl-
cdn.alpinelinux.org/alpine/v3.11/community/x86_64/APKINDEX.tar.gz
v3.11.6-209-g19f4c5d312 [http://dl-cdn.alpinelinux.org/alpine/v3.11/main]
v3.11.6-212-g3a84324cad [http://dl-
cdn.alpinelinux.org/alpine/v3.11/community]
OK: 11275 distinct packages available
(1/9) Installing openssh-keygen (8.1_p1-r0)
(2/9) Installing ncurses-terminfo-base (6.1_p20200118-r4)
(3/9) Installing ncurses-libs (6.1_p20200118-r4)
(4/9) Installing libedit (20191211.3.1-r0)
(5/9) Installing openssh-client (8.1_p1-r0)
(6/9) Installing openssh-sftp-server (8.1_p1-r0)
(7/9) Installing openssh-server-common (8.1_p1-r0)
(8/9) Installing openssh-server (8.1_p1-r0)
(9/9) Installing openssh (8.1_p1-r0)
Executing busybox-1.31.1-r9.trigger
OK: 11 MiB in 23 packages
[32;1m$ docker run --network="testing_network" test_service:latest python
healthcheck.py[0;m
[32;1m$ docker run --network="testing_network" test_service:latest python
test_apigateway.py[0;m
........
----------------------------------------------------------------------
Ran 8 tests in 36.383s

OK
[32;1m$ docker-compose -f docker-compose.test.yml down -v[0;m
Found orphan containers (devopsproject_state_service_devops_prod_1,
devopsproject_imed_service_devops_prod_1,
devopsproject_apigateway_service_devops_prod_1,
devopsproject_httpserv_service_devops_prod_1,
devopsproject_obse_service_devops_prod_1,
devopsproject_broker_service_devops_prod_1,
devopsproject_orig_service_devops_prod_1) for this project. If you removed
or renamed this service in your compose file, you can run this command with
the --remove-orphans flag to clean it up.
Removing devopsproject_state_service_devops_test_1      ...
Removing devopsproject_obse_service_devops_test_1       ...
Removing devopsproject_apigateway_service_devops_test_1 ...
Removing devopsproject_httpserv_service_devops_test_1   ...
Removing devopsproject_imed_service_devops_test_1       ...
Removing devopsproject_orig_service_devops_test_1       ...
Removing devopsproject_broker_service_devops_test_1     ...
[7A[2K
Removing devopsproject_state_service_devops_test_1      ... [32mdone[0m
[7B[4A[2K
Removing devopsproject_httpserv_service_devops_test_1   ... [32mdone[0m
[4B[2A[2K
```

```
Removing devopsproject_orig_service_devops_test_1       ... [32mdone[0m
[2B[6A[2K
Removing devopsproject_obse_service_devops_test_1       ... [32mdone[0m
[6B[5A[2K
Removing devopsproject_apigateway_service_devops_test_1 ... [32mdone[0m
[5B[3A[2K
Removing devopsproject_imed_service_devops_test_1       ... [32mdone[0m
[3B[1A[2K
Removing devopsproject_broker_service_devops_test_1     ... [32mdone[0m
[1BRemoving network testing_network
Removing volume devopsproject_output_data_test
Removing volume devopsproject_state_data_test
section_end:1607016911:step_script
[0K[32;1mJob succeeded
[0;m
```

Example logs of successful deploy

```
[0KRunning with gitlab-runner 13.6.0 (8fa89735)
[0;m[0K  on docker-stable TAXGxvh7
[0;msection_start:1607016914:prepare_executor
[0K[0K[36;1mPreparing the "docker" executor[0;m
[0;m[0KUsing Docker executor with image docker/compose:latest ...
[0;m[0KPulling docker image docker/compose:latest ...
[0;m[0KUsing docker image
sha256:c3e188a6b38fa7f54cac9963e11b049f7701b8a6354962218b6bbab0fba07bdf for
docker/compose:latest with digest
docker/compose@sha256:b60a020c0f68047b353a4a747f27f5e5ddb17116b7b018762edfb
6f7a6439a82 ...
[0;msection_end:1607016916:prepare_executor
[0Ksection_start:1607016916:prepare_script
[0K[0K[36;1mPreparing environment[0;m
[0;mRunning on runner-taxgxvh7-project-2-concurrent-0 via gitlab-runner1...
section_end:1607016917:prepare_script
[0Ksection_start:1607016917:get_sources
[0K[0K[36;1mGetting source from Git repository[0;m
[0;m[32;1mFetching changes with git depth set to 50...[0;m
Reinitialized existing Git repository in /builds/root/devopsproject/.git/
[32;1mChecking out 44494cd9 as master_remote...[0;m

[32;1mSkipping Git submodules setup[0;m
section_end:1607016918:get_sources
[0Ksection_start:1607016918:step_script
[0K[0K[36;1mExecuting "step_script" stage of the job script[0;m
[0;m[32;1m$ apk update && apk add openssh[0;m
fetch http://dl-
cdn.alpinelinux.org/alpine/v3.11/main/x86_64/APKINDEX.tar.gz
fetch http://dl-
cdn.alpinelinux.org/alpine/v3.11/community/x86_64/APKINDEX.tar.gz
v3.11.6-209-g19f4c5d312 [http://dl-cdn.alpinelinux.org/alpine/v3.11/main]
v3.11.6-212-g3a84324cad [http://dl-
cdn.alpinelinux.org/alpine/v3.11/community]
OK: 11275 distinct packages available
(1/9) Installing openssh-keygen (8.1_p1-r0)
(2/9) Installing ncurses-terminfo-base (6.1_p20200118-r4)
(3/9) Installing ncurses-libs (6.1_p20200118-r4)
(4/9) Installing libedit (20191211.3.1-r0)
(5/9) Installing openssh-client (8.1_p1-r0)
(6/9) Installing openssh-sftp-server (8.1_p1-r0)
(7/9) Installing openssh-server-common (8.1_p1-r0)
(8/9) Installing openssh-server (8.1_p1-r0)
```

```
(9/9) Installing openssh (8.1_p1-r0)
Executing busybox-1.31.1-r9.trigger
OK: 11 MiB in 23 packages
[32;1m$ ls -l /etc/ssh[0;m
total 576
-rw-------    1 root     root          2602 Dec  3 17:26
devops_deploy_key.txt
-rw-r--r--    1 root     root        577388 Nov 15  2019 moduli
-rw-r--r--    1 root     root          1484 Nov 15  2019 ssh_config
-rw-r--r--    1 root     root          3177 Nov 15  2019 sshd_config
[32;1m$ export DEPLOY_ENV=remote[0;m
[32;1m$ sh deploy.sh[0;m
Production deployment
Warning: Permanently added '34.74.12.236' (ECDSA) to the list of known
hosts.
Stopping devopsproject_state_service_devops_prod_1      ...
Stopping devopsproject_httpserv_service_devops_prod_1   ...
Stopping devopsproject_apigateway_service_devops_prod_1 ...
Stopping devopsproject_imed_service_devops_prod_1       ...
Stopping devopsproject_broker_service_devops_prod_1     ...
Stopping devopsproject_obse_service_devops_prod_1       ...
Stopping devopsproject_orig_service_devops_prod_1       ...
[2A[2K
Stopping devopsproject_obse_service_devops_prod_1       ... [32mdone[0m
[2B[4A[2K
Stopping devopsproject_imed_service_devops_prod_1       ... [32mdone[0m
[4B[1A[2K
Stopping devopsproject_orig_service_devops_prod_1       ... [32mdone[0m
[1B[3A[2K
Stopping devopsproject_broker_service_devops_prod_1     ... [32mdone[0m
[3B[6A[2K
Stopping devopsproject_httpserv_service_devops_prod_1   ... [32mdone[0m
[6B[7A[2K
Stopping devopsproject_state_service_devops_prod_1      ... [32mdone[0m
[7B[5A[2K
Stopping devopsproject_apigateway_service_devops_prod_1 ... [32mdone[0m
[5BRemoving devopsproject_state_service_devops_prod_1      ...
Removing devopsproject_httpserv_service_devops_prod_1   ...
Removing devopsproject_apigateway_service_devops_prod_1 ...
Removing devopsproject_imed_service_devops_prod_1       ...
Removing devopsproject_broker_service_devops_prod_1     ...
Removing devopsproject_obse_service_devops_prod_1       ...
Removing devopsproject_orig_service_devops_prod_1       ...
[1A[2K
Removing devopsproject_orig_service_devops_prod_1       ... [32mdone[0m
[1B[2A[2K
Removing devopsproject_obse_service_devops_prod_1       ... [32mdone[0m
[2B[7A[2K
Removing devopsproject_state_service_devops_prod_1      ... [32mdone[0m
[7B[5A[2K
Removing devopsproject_apigateway_service_devops_prod_1 ... [32mdone[0m
[5B[6A[2K
Removing devopsproject_httpserv_service_devops_prod_1   ... [32mdone[0m
[6B[3A[2K
Removing devopsproject_broker_service_devops_prod_1     ... [32mdone[0m
[3B[4A[2K
Removing devopsproject_imed_service_devops_prod_1       ... [32mdone[0m
[4BRemoving network devopsproject_messaging_network
Removing volume devopsproject_output_data
Removing volume devopsproject_state_data
Creating network "devopsproject_messaging_network" with the default driver
Creating volume "devopsproject_output_data" with default driver
Creating volume "devopsproject_state_data" with default driver
Building broker_service_devops_prod
Step 1/1 : FROM rabbitmq:3.8
 ---> f50f482879b3

Successfully built f50f482879b3
Successfully tagged devopsproject_broker_service_devops_prod:latest
Building orig_service_devops_prod
Step 1/5 : FROM python:3.9
 ---> a3fe352c5377
Step 2/5 : COPY requirements.txt .
 ---> Using cache
 ---> 31d9ffa5766f
Step 3/5 : RUN pip install -r requirements.txt
 ---> Using cache
 ---> 92712181bdbe
Step 4/5 : COPY main.py .
 ---> Using cache
 ---> a6f4b652671c
Step 5/5 : CMD ["python", "main.py"]
 ---> Using cache
 ---> 68a95b7cd7e1

Successfully built 68a95b7cd7e1
Successfully tagged devopsproject_orig_service_devops_prod:latest
Building imed_service_devops_prod
Step 1/5 : FROM python:3.9
 ---> a3fe352c5377
Step 2/5 : COPY requirements.txt .
 ---> Using cache
 ---> 23ae3d81229f
Step 3/5 : RUN pip install -r requirements.txt
 ---> Using cache
 ---> b2b0bd73832c
Step 4/5 : COPY main.py .
 ---> Using cache
 ---> 531481ac66b9
Step 5/5 : CMD ["python", "main.py"]
 ---> Using cache
 ---> 016de6c53bca

Successfully built 016de6c53bca
Successfully tagged devopsproject_imed_service_devops_prod:latest
Building obse_service_devops_prod
Step 1/5 : FROM python:3.9
 ---> a3fe352c5377
Step 2/5 : COPY requirements.txt .
 ---> Using cache
 ---> 31d9ffa5766f
Step 3/5 : RUN pip install -r requirements.txt
 ---> Using cache
 ---> 92712181bdbe
Step 4/5 : COPY main.py .
 ---> Using cache
 ---> 93b065fd6bd0
Step 5/5 : CMD ["python", "main.py"]
 ---> Using cache
 ---> 03d835cad2f6

Successfully built 03d835cad2f6
Successfully tagged devopsproject_obse_service_devops_prod:latest
Building httpserv_service_devops_prod
Step 1/3 : FROM node:14
 ---> 4c7835fa737e
Step 2/3 : ADD server.js /
 ---> Using cache
 ---> d5e7ba9ff364
Step 3/3 : CMD ["node", "server.js"]
 ---> Using cache
 ---> 0675fbd55dab

Successfully built 0675fbd55dab
Successfully tagged devopsproject_httpserv_service_devops_prod:latest
Building apigateway_service_devops_prod
Step 1/3 : FROM node:14
 ---> 4c7835fa737e
Step 2/3 : ADD server.js /
 ---> Using cache
 ---> 8f65caea6f1e
Step 3/3 : CMD ["node", "server.js"]
 ---> Using cache
 ---> 57751a767524

Successfully built 57751a767524
Successfully tagged devopsproject_apigateway_service_devops_prod:latest
Building state_service_devops_prod
Step 1/7 : FROM node:14
 ---> 4c7835fa737e
Step 2/7 : COPY --from=docker:latest /usr/local/bin/docker /usr/local/bin/
 ---> Using cache
 ---> 7289e85a8603
Step 3/7 : RUN curl -L
"https://github.com/docker/compose/releases/download/1.27.4/docker-compose-
$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
 ---> Using cache
 ---> fb72294ad066
Step 4/7 : RUN chmod +x /usr/local/bin/docker-compose
 ---> Using cache
 ---> 192fe8487ec6
Step 5/7 : RUN ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
 ---> Using cache
 ---> 36365b221f96
Step 6/7 : ADD server.js /
 ---> Using cache
 ---> a399dc485beb
Step 7/7 : CMD ["node", "server.js"]
 ---> Using cache
 ---> 94fc17ac51e6

Successfully built 94fc17ac51e6
Successfully tagged devopsproject_state_service_devops_prod:latest
Creating devopsproject_orig_service_devops_prod_1 ...
Creating devopsproject_httpserv_service_devops_prod_1 ...
Creating devopsproject_imed_service_devops_prod_1     ...
Creating devopsproject_apigateway_service_devops_prod_1 ...
Creating devopsproject_state_service_devops_prod_1       ...
Creating devopsproject_broker_service_devops_prod_1      ...
Creating devopsproject_obse_service_devops_prod_1        ...
[7A[2K
Creating devopsproject_orig_service_devops_prod_1        ... [32mdone[0m
[7B[5A[2K
```

```
Creating devopsproject_imed_service_devops_prod_1       ... [32mdone[0m
[5B[1A[2K
Creating devopsproject_obse_service_devops_prod_1       ... [32mdone[0m
[1B[2A[2K
Creating devopsproject_broker_service_devops_prod_1     ... [32mdone[0m
[2B[4A[2K
Creating devopsproject_apigateway_service_devops_prod_1 ... [32mdone[0m
[4B[6A[2K
Creating devopsproject_httpserv_service_devops_prod_1   ... [32mdone[0m
[6B[3A[2K
Creating devopsproject_state_service_devops_prod_1      ... [32mdone[0m
[3Bsection_end:1607016942:step_script
[0K[32;1mJob succeeded
[0;m
```

Example logs of failing tests

```
[0KRunning with gitlab-runner 13.6.0 (8fa89735)
[0;m[0K  on docker-stable TAXGxvh7
[0;msection_start:1607019318:prepare_executor
[0K[0K[36;1mPreparing the "docker" executor[0;m
[0;m[0KUsing Docker executor with image docker/compose:latest ...
[0;m[0KPulling docker image docker/compose:latest ...
[0;m[0KUsing docker image
sha256:c3e188a6b38fa7f54cac9963e11b049f7701b8a6354962218b6bbab0fba07bdf for
docker/compose:latest with digest
docker/compose@sha256:b60a020c0f68047b353a4a747f27f5e5ddb17116b7b018762edfb
6f7a6439a82 ...
[0;msection_end:1607019321:prepare_executor
[0Ksection_start:1607019321:prepare_script
[0K[0K[36;1mPreparing environment[0;m
[0;mRunning on runner-taxgxvh7-project-2-concurrent-0 via gitlab-runner1...
section_end:1607019322:prepare_script
[0Ksection_start:1607019322:get_sources
[0K[0K[36;1mGetting source from Git repository[0;m
[0;m[32;1mFetching changes with git depth set to 50...[0;m
Reinitialized existing Git repository in /builds/root/devopsproject/.git/
[32;1mChecking out 0934c5ff as failing-branch...[0;m

[32;1mSkipping Git submodules setup[0;m
section_end:1607019324:get_sources
[0Ksection_start:1607019324:step_script
[0K[0K[36;1mExecuting "step_script" stage of the job script[0;m
[0;m[32;1m$ apk update && apk add openssh[0;m
fetch http://dl-
cdn.alpinelinux.org/alpine/v3.11/main/x86_64/APKINDEX.tar.gz
fetch http://dl-
cdn.alpinelinux.org/alpine/v3.11/community/x86_64/APKINDEX.tar.gz
v3.11.6-209-g19f4c5d312 [http://dl-cdn.alpinelinux.org/alpine/v3.11/main]
v3.11.6-212-g3a84324cad [http://dl-
cdn.alpinelinux.org/alpine/v3.11/community]
OK: 11275 distinct packages available
(1/9) Installing openssh-keygen (8.1_p1-r0)
(2/9) Installing ncurses-terminfo-base (6.1_p20200118-r4)
(3/9) Installing ncurses-libs (6.1_p20200118-r4)
(4/9) Installing libedit (20191211.3.1-r0)
(5/9) Installing openssh-client (8.1_p1-r0)
(6/9) Installing openssh-sftp-server (8.1_p1-r0)
(7/9) Installing openssh-server-common (8.1_p1-r0)
(8/9) Installing openssh-server (8.1_p1-r0)
(9/9) Installing openssh (8.1_p1-r0)
```

```
Executing busybox-1.31.1-r9.trigger
OK: 11 MiB in 23 packages
[32;1m$ docker run --network="testing_network" test_service:latest python
healthcheck.py[0;m
[32;1m$ docker run --network="testing_network" test_service:latest python
test_apigateway.py[0;m
..F.....
======================================================================
FAIL: test_queue_statistic (__main__.APIGatewayTestCase)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "//test_apigateway.py", line 55, in test_queue_statistic
    self.assertEqual(expected_keys, set(statistics[0].keys()))
AssertionError: Items in the first set but not the second:
'message_published_lately_broken'
Items in the second set but not the first:
'message_published_lately'

----------------------------------------------------------------------
Ran 8 tests in 45.652s

FAILED (failures=1)
section_end:1607019374:step_script
[0K[31;1mERROR: Job failed: exit code 1
[0;m
```