COMP.SE.140 Project – Fall 2023

1. Instructions for the teaching assistant

Implemented optional features.

1. Implemented a static analysis step in the pipeline by using SonarQube.

2. Implemented *GET /mqstatistic* endpoint in the system

3. Implemented deployment to an external cloud (AWS)

Instructions for examiner to test the system.

1.To run the system's basic requirements,

1.1 Clone the project using below url.

```
git clone -b project https://course-gitlab.tuni.fi/comp.se.140-fall2023_2023-2024/dcthra.git
```

1.2 Change directory to the project.

```
cd dcthra
```

1.3 Build the system using the below command.

```
docker-compose build --no-cache
```

1.4 Run the system using the below command.

```
docker-compose up -d
```

2. To test the system's basic requirements

2.1 Use curl/Postman to test the system

e.g :-
```
curl localhost:8083/state -X PUT -d ”INIT” -H ”Content-Type: text/plain” -H ”Accept: text/plain”
```

2.2 To test the GET */mqstatistic* endpoint in the system,

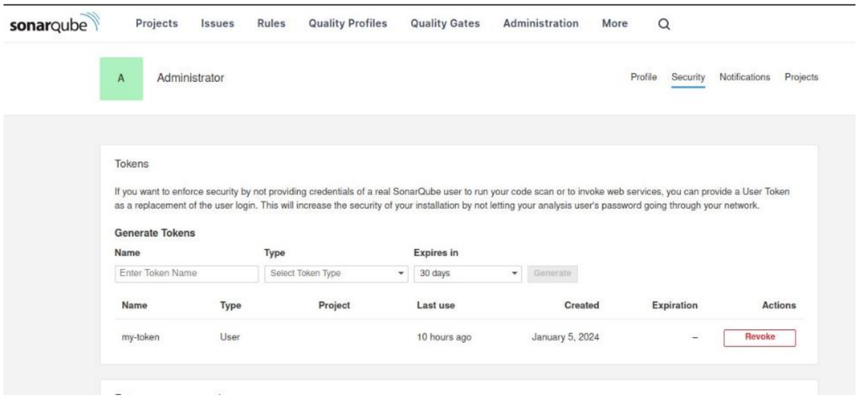
```
curl --location 'localhost:8083/mqstatistic'
```

3. Test SonarQube Integration.

2.1 Run SonarQube docker container using the below command.

```
docker run -d --name sonarqube -p 9000:9000 -p 9092:9092 sonarqube
```

2.2 Login to the SonarQube by using default admin/admin credentials and generate a new user(admin) token by navigating to User > My Account > Security [1].



Tokens

If you want to enforce security by not providing credentials of a real SonarQube user to run your code scan or to invoke web services, you can provide a User Token as a replacement of the user login. This will increase the security of your installation by not letting your analysis user's password going through your network.

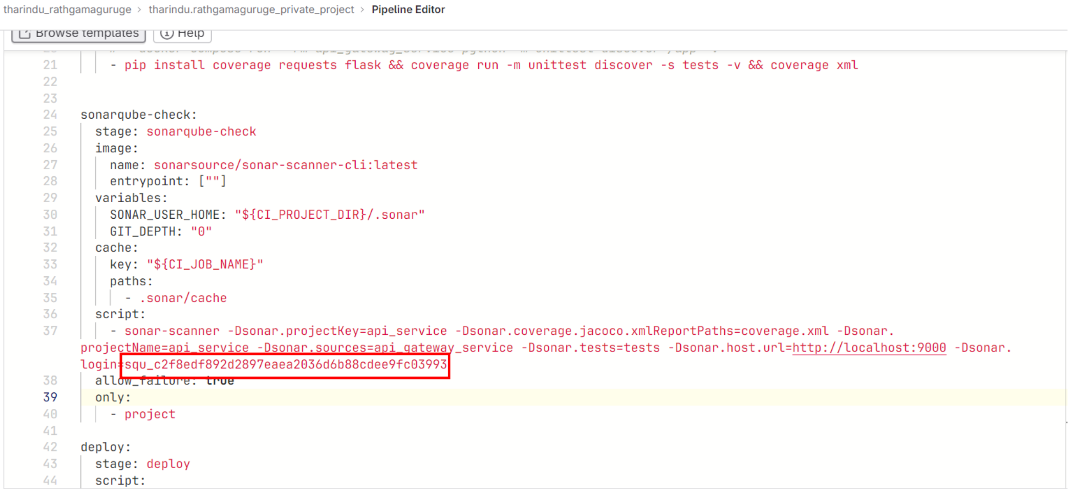
Generate Tokens

Name	Type	Expires in	
Enter Token Name	Select Token Type	30 days	Generate

Name	Type	Project	Last use	Created	Expiration	Actions
my-token	User		10 hours ago	January 5, 2024	–	Revoke

Figure: SonarQube Account page

2.3 Update the -Dsonar.login= <new_token> placeholder value with the previously generated token under the sonarqube-check stage in the .gitlab-ci.yml file.



tharindu_rathgamaguruge › tharindu.rathgamaguruge_private_project › Pipeline Editor
Browse templates
Help

```
    - pip install coverage requests flask && coverage run -m unittest discover -s tests -v && coverage xml


sonarqube-check:
  stage: sonarqube-check
  image:
    name: sonarsource/sonar-scanner-cli:latest
    entrypoint: [""]
  variables:
    SONAR_USER_HOME: "${CI_PROJECT_DIR}/.sonar"
    GIT_DEPTH: "0"
  cache:
    key: "${CI_JOB_NAME}"
    paths:
      - .sonar/cache
  script:
    - sonar-scanner -Dsonar.projectKey=api_service -Dsonar.coverage.jacoco.xmlReportPaths=coverage.xml -Dsonar.projectName=api_service -Dsonar.sources=api_gateway_service -Dsonar.tests=tests -Dsonar.host.url=http://localhost:9000 -Dsonar.login=squ_c2f8edf892d2897eaea2036d6b88cdee9fc03993
  allow_failure: true
  only:
    - project

deploy:
  stage: deploy
  script:
```

Figure: sonarqube-check stage in gitlab-ci.yml

P.S- Here, I have used gitlab-runner with a specific tag. Therefore, you may need to change the tag according to your runner tag to test the system with sonarqube and gitlab-ci.

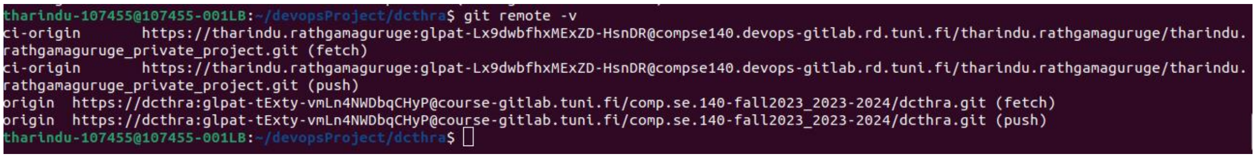
2.4 Add below configuration into the config.toml file in /etc/gitlab-runner location.

```
[runners.docker]
......
......
 volumes = ["/var/run/docker.sock:/var/run/docker.sock", "/cache"]
.....
```

2. Description of the CI/CD pipeline

- VCS and Branches:

In this project Git was used as VCS and Gitlab as centralized VCS platform. Followed muti remote repositories management method. One repository to build, test and deploy the system more efficiently with Gitlab CI and Other gitlab to keep the final code.



```
tharindu-107455@107455-001LB:~/devopsProject/dcthra$ git remote -v
ci-origin       https://tharindu.rathgamaguruge:glpat-Lx9dwbfhxMExZD-HsnDR@compse140.devops-gitlab.rd.tuni.fi/tharindu.rathgamaguruge/tharindu.
rathgamaguruge_private_project.git (fetch)
ci-origin       https://tharindu.rathgamaguruge:glpat-Lx9dwbfhxMExZD-HsnDR@compse140.devops-gitlab.rd.tuni.fi/tharindu.rathgamaguruge/tharindu.
rathgamaguruge_private_project.git (push)
origin  https://dcthra:glpat-tExty-vmLn4NWDbqCHyP@course-gitlab.tuni.fi/comp.se.140-fall2023_2023-2024/dcthra.git (fetch)
origin  https://dcthra:glpat-tExty-vmLn4NWDbqCHyP@course-gitlab.tuni.fi/comp.se.140-fall2023_2023-2024/dcthra.git (push)
tharindu-107455@107455-001LB:~/devopsProject/dcthra$ 
```

Figure: remote repositories

Created project branch in both repositories from the exercise2 branch and used project branch to do changes during the implementations.

- Building tools

Used Python and JavaScript as main two programming languages in the project. npm and pip as package management tools. Due to the single script files, no build tools used to build the applications.

- Testing; tools and test cases

Testings mainly based on the api gateway service.

Test framework: - unittest (Python)

Test cases

1. To test GET /messages endpoint and expected response code.
   user should be able to get expected response from the system.
2. To test PUT /state endpoint with
   2.1 by setting INIT as data
   2.2 by setting PAUSE as data.
   2.3 by setting RUNNING as data.
   2.4 by setting SHUTDOWN as data
   2.5 by setting dummy data (FAKE) as data

   User should be able to get expected responses (status updates) according to the request.
3. To test GET /state endpoint and expected response.
   User should be able to get current state according to the request.

4. To test GET /run-log endpoint and expected response.
   User should be able to get the status change in the response

5. To test GET /mqstatistics endpoint and expected response.
   user should be able to get expected response from the system.

- Packing

packaging done with docker.

- Deployment

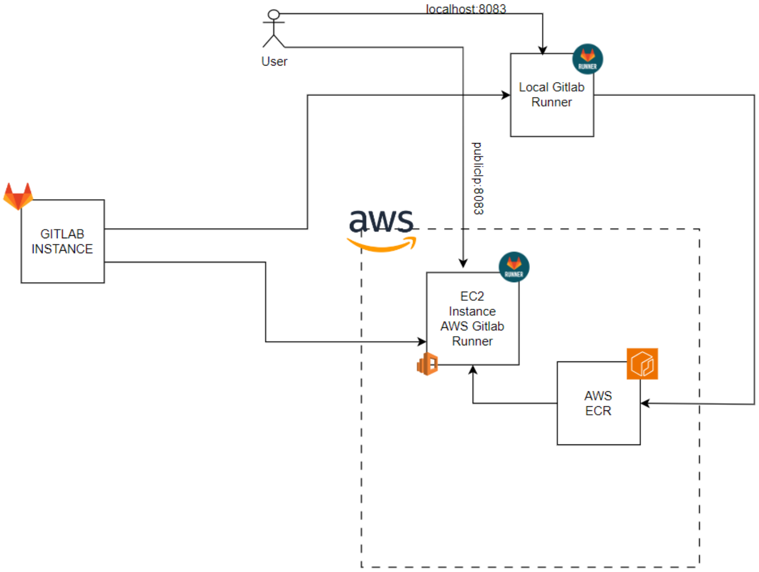


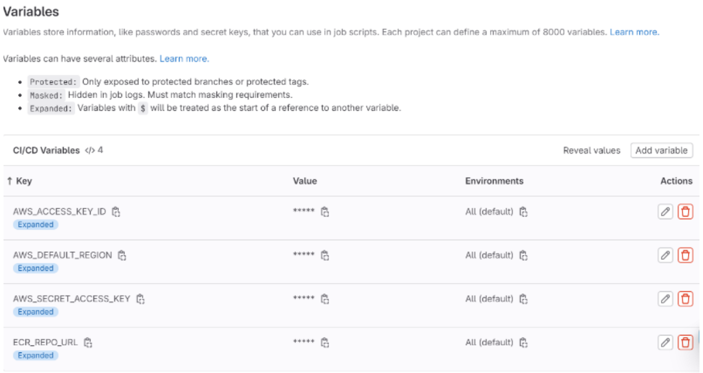
Figure: Deployment Architecture

1. Local deployment done with docker-compose using docker-compose up -d command.

2. AWS deployment

   2.1 Created an aws instance on the aws cloud.
   2.2 Created ECR(Elastic Container Registry) private repositories for service1, service2, api_gateway_service and monitoring_service (To ensure redundancy and high availability)
   2.3 Generated access key and secret key with granting permission for ec2 instances and ecr access.

2.4 Added secret key, access key data in gitlab CI/CD variables (To ensure security)



Variables

Variables store information, like passwords and secret keys, that you can use in job scripts. Each project can define a maximum of 8000 variables. Learn more.

Variables can have several attributes. Learn more.

- `Protected:` Only exposed to protected branches or protected tags.
- `Masked:` Hidden in job logs. Must match masking requirements.
- `Expanded:` Variables with `$` will be treated as the start of a reference to another variable.

CI/CD Variables </> 4 Reveal values Add variable

↑ Key	Value	Environments	Actions
AWS_ACCESS_KEY_ID Expanded	*****	All (default)	
AWS_DEFAULT_REGION Expanded	*****	All (default)	
AWS_SECRET_ACCESS_KEY Expanded	*****	All (default)	
ECR_REPO_URL Expanded	*****	All (default)	

Figure: GitLab CI/CD Variables

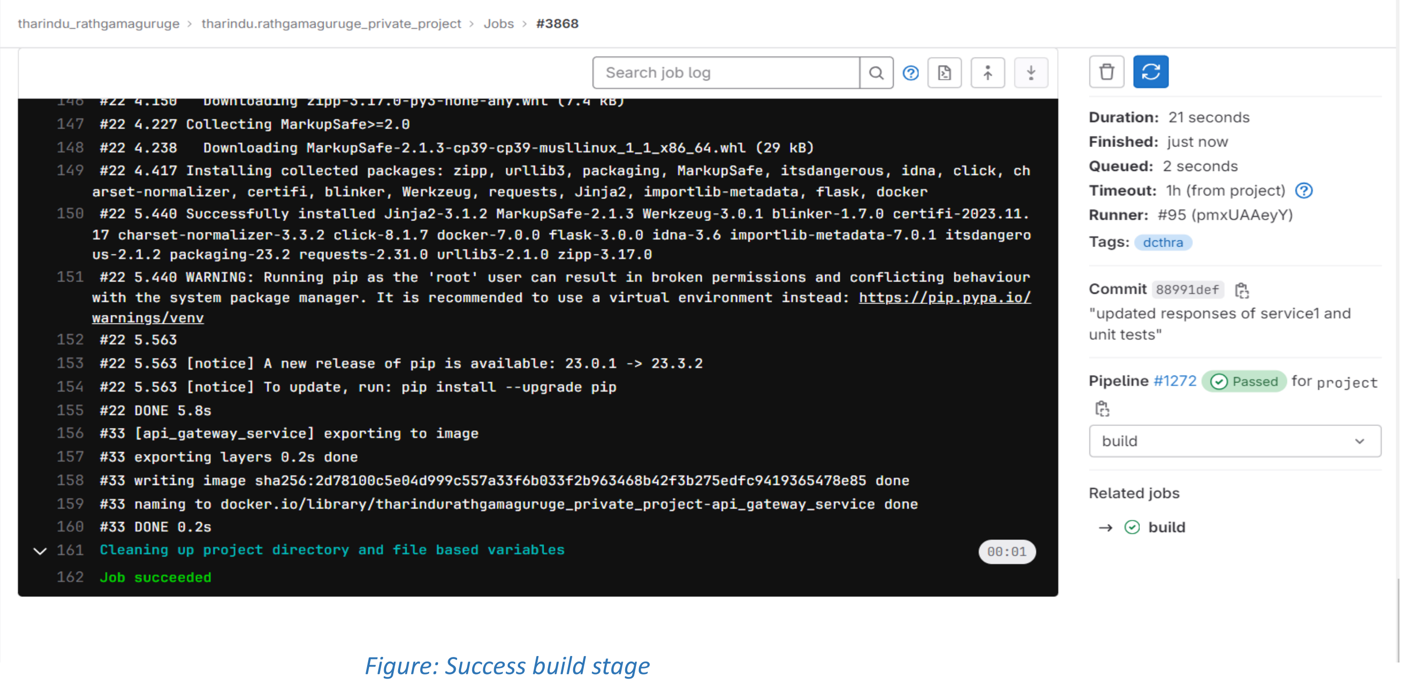
2.5 Installed and registered gitlab runner on ec2 instance.
2.6 Added bash script to push images into ecr.
2.7 Added two stages to .gitlab-ci.yml to push the images to ecr and deploy docker containers using docker-compose to ec2 instance.

- Operating; monitoring

Did not implement

3. Example runs of the pipeline

3.1 Success Build Stage



tharindu_rathgamaguruge › tharindu.rathgamaguruge_private_project › Jobs › **#3868**

```
#22 4.150   Downloading zipp-3.17.0-py3-none-any.whl (7.4 kB)
#22 4.227 Collecting MarkupSafe>=2.0
#22 4.238   Downloading MarkupSafe-2.1.3-cp39-cp39-musllinux_1_1_x86_64.whl (29 kB)
#22 4.417 Installing collected packages: zipp, urllib3, packaging, MarkupSafe, itsdangerous, idna, click, ch
arset-normalizer, certifi, blinker, Werkzeug, requests, Jinja2, importlib-metadata, flask, docker
#22 5.440 Successfully installed Jinja2-3.1.2 MarkupSafe-2.1.3 Werkzeug-3.0.1 blinker-1.7.0 certifi-2023.11.
17 charset-normalizer-3.3.2 click-8.1.7 docker-7.0.0 flask-3.0.0 idna-3.6 importlib-metadata-7.0.1 itsdangero
us-2.1.2 packaging-23.2 requests-2.31.0 urllib3-2.1.0 zipp-3.17.0
#22 5.440 WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour
with the system package manager. It is recommended to use a virtual environment instead: https://pip.pypa.io/
warnings/venv
#22 5.563
#22 5.563 [notice] A new release of pip is available: 23.0.1 -> 23.3.2
#22 5.563 [notice] To update, run: pip install --upgrade pip
#22 DONE 5.8s
#33 [api_gateway_service] exporting to image
#33 exporting layers 0.2s done
#33 writing image sha256:2d78100c5e04d999c557a33f6b033f2b963468b42f3b275edfc9419365478e85 done
#33 naming to docker.io/library/tharindurathgamaguruge_private_project-api_gateway_service done
#33 DONE 0.2s
Cleaning up project directory and file based variables                                          00:01
Job succeeded
```

Duration: 21 seconds
Finished: just now
Queued: 2 seconds
Timeout: 1h (from project)
Runner: #95 (pmxUAAeyY)
Tags: dcthra

Commit `88991def`
"updated responses of service1 and unit tests"

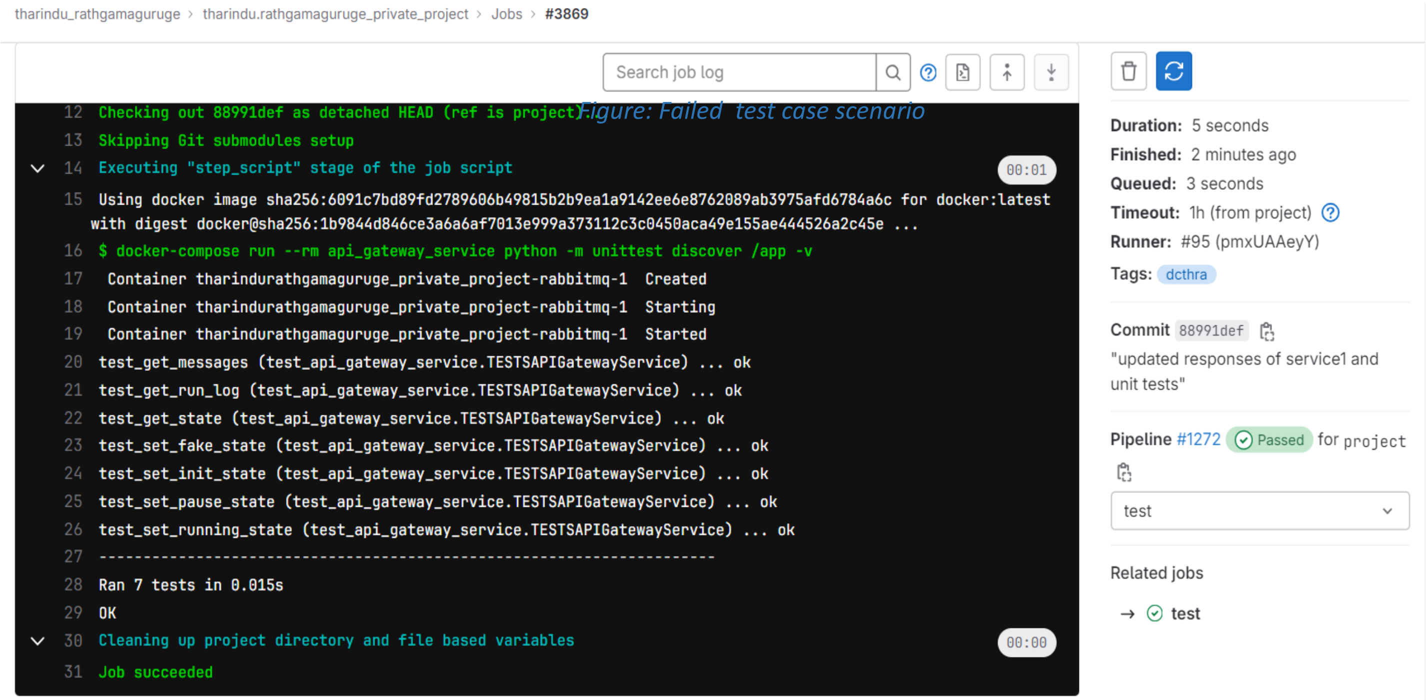
Pipeline #1272 Passed for `project`

Related jobs

→ ✓ **build**

Figure: Success build stage

3.2 Passed test case scenario



tharindu_rathgamaguruge › tharindu.rathgamaguruge_private_project › Jobs › **#3869**

```
Checking out 88991def as detached HEAD (ref is project)
Skipping Git submodules setup
Executing "step_script" stage of the job script                                          00:01
Using docker image sha256:6091c7bd89fd2789606b49815b2b9ea1a9142ee6e8762089ab3975afd6784a6c for docker:latest
with digest docker@sha256:1b9844d846ce3a6a6af7013e999a373112c3c0450aca49e155ae444526a2c45e ...
$ docker-compose run --rm api_gateway_service python -m unittest discover /app -v
 Container tharindurathgamaguruge_private_project-rabbitmq-1  Created
 Container tharindurathgamaguruge_private_project-rabbitmq-1  Starting
 Container tharindurathgamaguruge_private_project-rabbitmq-1  Started
test_get_messages (test_api_gateway_service.TESTSAPIGatewayService) ... ok
test_get_run_log (test_api_gateway_service.TESTSAPIGatewayService) ... ok
test_get_state (test_api_gateway_service.TESTSAPIGatewayService) ... ok
test_set_fake_state (test_api_gateway_service.TESTSAPIGatewayService) ... ok
test_set_init_state (test_api_gateway_service.TESTSAPIGatewayService) ... ok
test_set_pause_state (test_api_gateway_service.TESTSAPIGatewayService) ... ok
test_set_running_state (test_api_gateway_service.TESTSAPIGatewayService) ... ok
----------------------------------------------------------------------
Ran 7 tests in 0.015s
OK
Cleaning up project directory and file based variables                                   00:00
Job succeeded
```

Figure: Failed test case scenario

Duration: 5 seconds
Finished: 2 minutes ago
Queued: 3 seconds
Timeout: 1h (from project)
Runner: #95 (pmxUAAeyY)
Tags: dcthra

Commit 88991def
"updated responses of service1 and unit tests"

Pipeline #1272 Passed for project

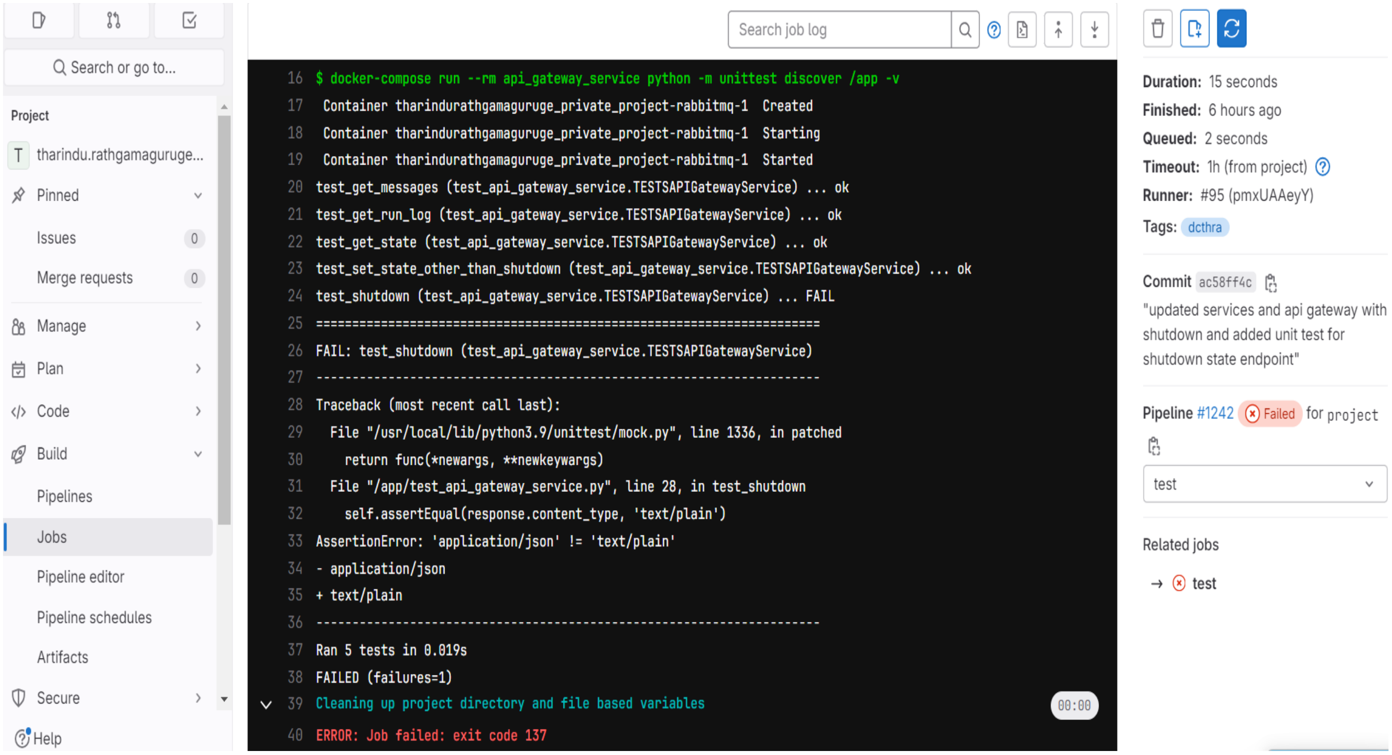
test

Related jobs

→ ✓ test

Figure: Passed test case scenario

3.3 Failed Test case scenario



```
$ docker-compose run --rm api_gateway_service python -m unittest discover /app -v
 Container tharindurathgamaguruge_private_project-rabbitmq-1  Created
 Container tharindurathgamaguruge_private_project-rabbitmq-1  Starting
 Container tharindurathgamaguruge_private_project-rabbitmq-1  Started
test_get_messages (test_api_gateway_service.TESTSAPIGatewayService) ... ok
test_get_run_log (test_api_gateway_service.TESTSAPIGatewayService) ... ok
test_get_state (test_api_gateway_service.TESTSAPIGatewayService) ... ok
test_set_state_other_than_shutdown (test_api_gateway_service.TESTSAPIGatewayService) ... ok
test_shutdown (test_api_gateway_service.TESTSAPIGatewayService) ... FAIL
======================================================================
FAIL: test_shutdown (test_api_gateway_service.TESTSAPIGatewayService)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "/usr/local/lib/python3.9/unittest/mock.py", line 1336, in patched
    return func(*newargs, **newkeywargs)
  File "/app/test_api_gateway_service.py", line 28, in test_shutdown
    self.assertEqual(response.content_type, 'text/plain')
AssertionError: 'application/json' != 'text/plain'
- application/json
+ text/plain
----------------------------------------------------------------------
Ran 5 tests in 0.019s
FAILED (failures=1)
Cleaning up project directory and file based variables                     00:00
ERROR: Job failed: exit code 137
```

Duration: 15 seconds
Finished: 6 hours ago
Queued: 2 seconds
Timeout: 1h (from project)
Runner: #95 (pmxUAAeyY)
Tags: dcthra

Commit ac58ff4c
"updated services and api gateway with shutdown and added unit test for shutdown state endpoint"

Pipeline #1242 Failed for project

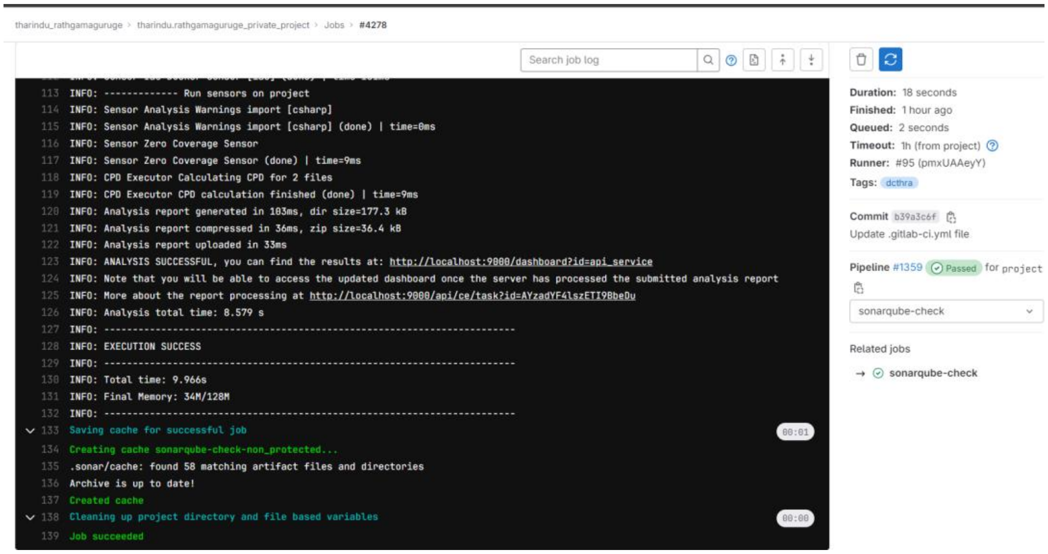
test

Related jobs

→ ⊗ test

Figure: Failed test case scenario

3.4 SonarQube check stage



tharindu_rathgamaguruge › tharindu.rathgamaguruge_private_project › Jobs › #4278

```
INFO: ------------- Run sensors on project
INFO: Sensor Analysis Warnings import [csharp]
INFO: Sensor Analysis Warnings import [csharp] (done) | time=0ms
INFO: Sensor Zero Coverage Sensor
INFO: Sensor Zero Coverage Sensor (done) | time=9ms
INFO: CPD Executor Calculating CPD for 2 files
INFO: CPD Executor CPD calculation finished (done) | time=9ms
INFO: Analysis report generated in 103ms, dir size=177.3 kB
INFO: Analysis report compressed in 36ms, zip size=36.4 kB
INFO: Analysis report uploaded in 33ms
INFO: ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=api_service
INFO: Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
INFO: More about the report processing at http://localhost:9000/api/ce/task?id=AYzadYF4lszETI9BbeDu
INFO: Analysis total time: 8.579 s
INFO: ------------------------------------------------------------------------
INFO: EXECUTION SUCCESS
INFO: ------------------------------------------------------------------------
INFO: Total time: 9.966s
INFO: Final Memory: 34M/128M
INFO: ------------------------------------------------------------------------
Saving cache for successful job                                                    00:01
Creating cache sonarqube-check-non_protected...
.sonar/cache: found 58 matching artifact files and directories
Archive is up to date!
Created cache
Cleaning up project directory and file based variables                             00:00
Job succeeded
```

Duration: 18 seconds
Finished: 1 hour ago
Queued: 2 seconds
Timeout: 1h (from project)
Runner: #95 (pmxUAAeyY)
Tags: dcthra

Commit b39a3c6f
Update .gitlab-ci.yml file

Pipeline #1359 Passed for project

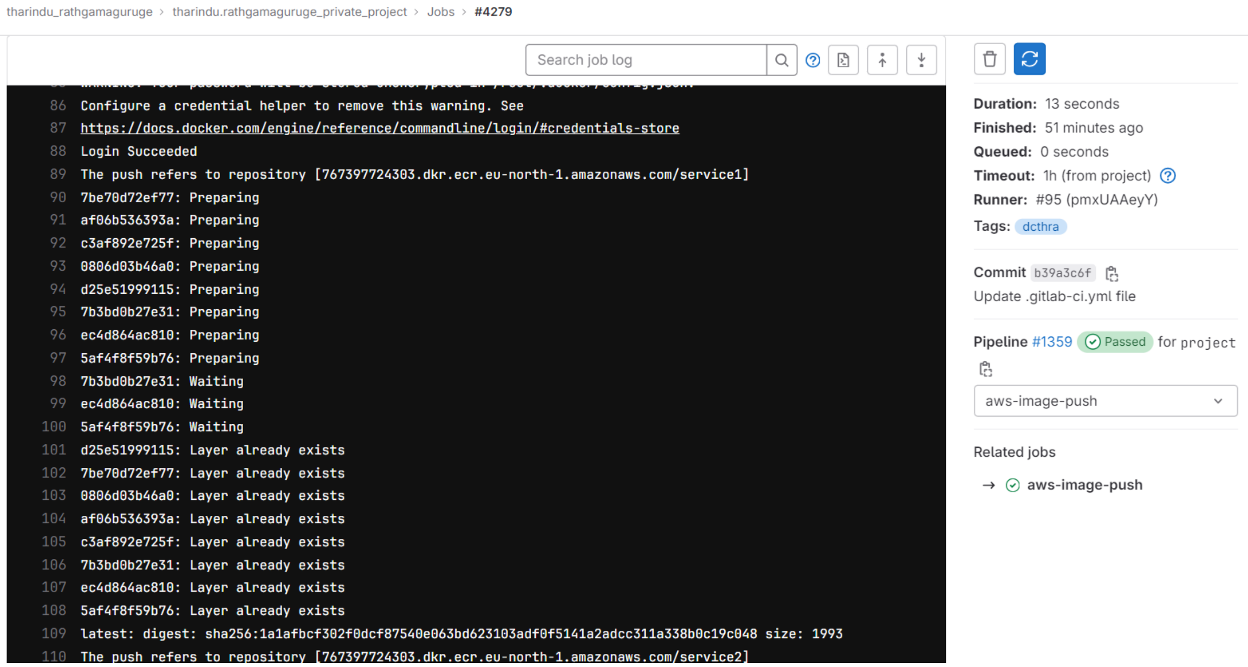
sonarqube-check

Related jobs

→ ✓ sonarqube-check

Figure: SonarQube check stage

3.5 Images push to ECR



tharindu_rathgamaguruge › tharindu.rathgamaguruge_private_project › Jobs › #4279

```
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store
Login Succeeded
The push refers to repository [767397724303.dkr.ecr.eu-north-1.amazonaws.com/service1]
7be70d72ef77: Preparing
af06b536393a: Preparing
c3af892e725f: Preparing
0806d03b46a0: Preparing
d25e51999115: Preparing
7b3bd0b27e31: Preparing
ec4d864ac810: Preparing
5af4f8f59b76: Preparing
7b3bd0b27e31: Waiting
ec4d864ac810: Waiting
5af4f8f59b76: Waiting
d25e51999115: Layer already exists
7be70d72ef77: Layer already exists
0806d03b46a0: Layer already exists
af06b536393a: Layer already exists
c3af892e725f: Layer already exists
7b3bd0b27e31: Layer already exists
ec4d864ac810: Layer already exists
5af4f8f59b76: Layer already exists
latest: digest: sha256:1a1afbcf302f0dcf87540e063bd623103adf0f5141a2adcc311a338b0c19c048 size: 1993
The push refers to repository [767397724303.dkr.ecr.eu-north-1.amazonaws.com/service2]
```

Duration: 13 seconds
Finished: 51 minutes ago
Queued: 0 seconds
Timeout: 1h (from project)
Runner: #95 (pmxUAAeyY)
Tags: dcthra

Commit b39a3c6f
Update .gitlab-ci.yml file

Pipeline #1359 Passed for project

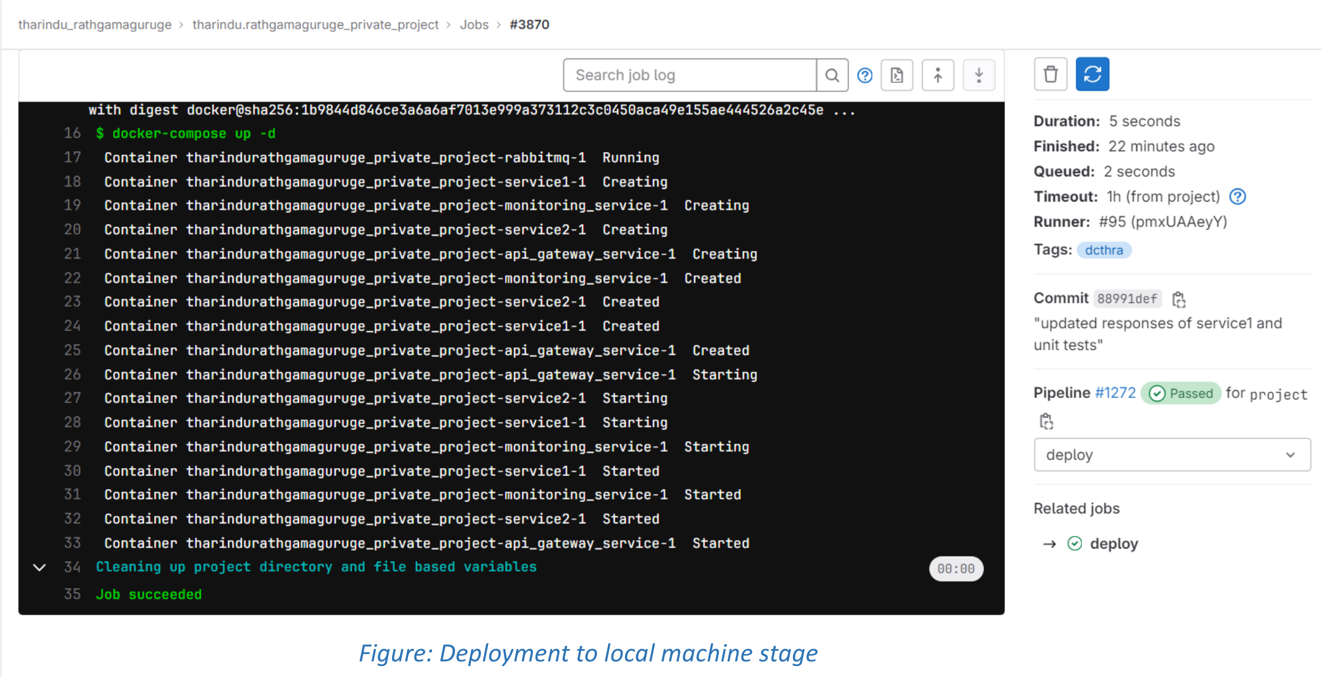
aws-image-push

Related jobs

→ ✓ aws-image-push

Figure: AWS Image push

3.6 Deploy stage.



tharindu_rathgamaguruge › tharindu.rathgamaguruge_private_project › Jobs › #3870

```
  with digest docker@sha256:1b9844d846ce3a6a6af7013e999a373112c3c0450aca49e155ae444526a2c45e ...
$ docker-compose up -d
 Container tharindurathgamaguruge_private_project-rabbitmq-1  Running
 Container tharindurathgamaguruge_private_project-service1-1  Creating
 Container tharindurathgamaguruge_private_project-monitoring_service-1  Creating
 Container tharindurathgamaguruge_private_project-service2-1  Creating
 Container tharindurathgamaguruge_private_project-api_gateway_service-1  Creating
 Container tharindurathgamaguruge_private_project-monitoring_service-1  Created
 Container tharindurathgamaguruge_private_project-service2-1  Created
 Container tharindurathgamaguruge_private_project-service1-1  Created
 Container tharindurathgamaguruge_private_project-api_gateway_service-1  Created
 Container tharindurathgamaguruge_private_project-api_gateway_service-1  Starting
 Container tharindurathgamaguruge_private_project-service2-1  Starting
 Container tharindurathgamaguruge_private_project-service1-1  Starting
 Container tharindurathgamaguruge_private_project-monitoring_service-1  Starting
 Container tharindurathgamaguruge_private_project-service1-1  Started
 Container tharindurathgamaguruge_private_project-monitoring_service-1  Started
 Container tharindurathgamaguruge_private_project-service2-1  Started
 Container tharindurathgamaguruge_private_project-api_gateway_service-1  Started
Cleaning up project directory and file based variables                                    00:00
Job succeeded
```

Duration: 5 seconds
Finished: 22 minutes ago
Queued: 2 seconds
Timeout: 1h (from project)
Runner: #95 (pmxUAAeyY)
Tags: dcthra

Commit 88991def
"updated responses of service1 and unit tests"

Pipeline #1272 Passed for project

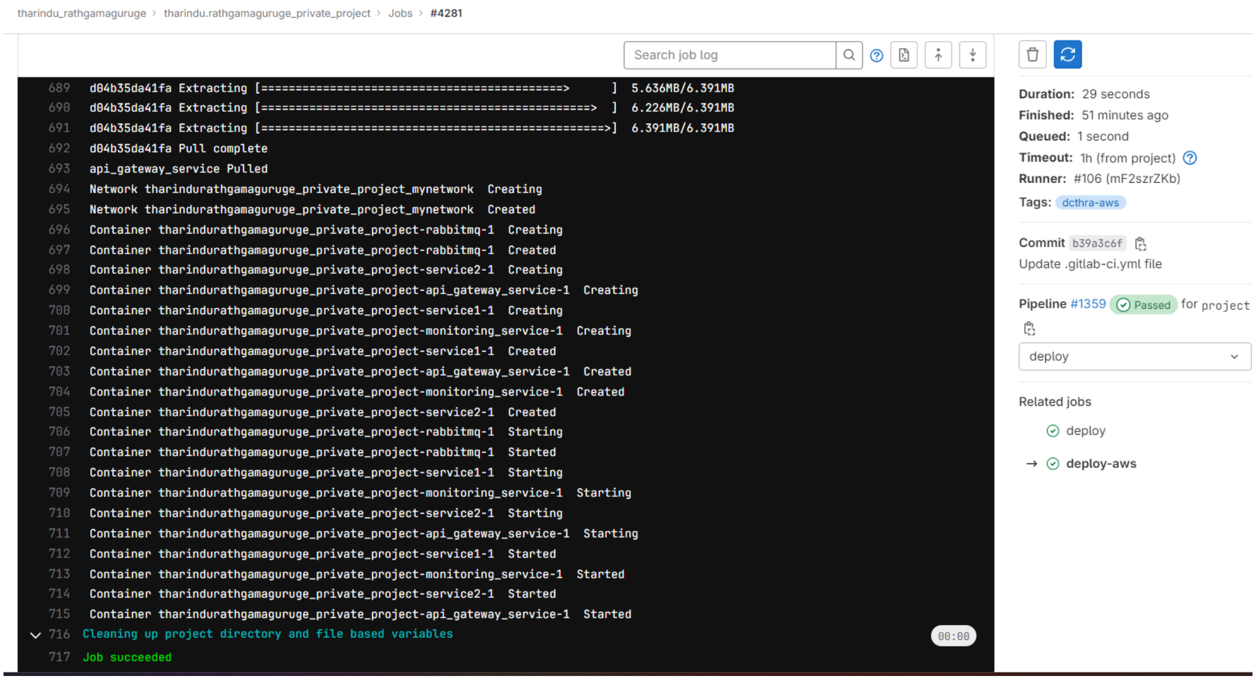
deploy

Related jobs

→ deploy

Figure: Deployment to local machine stage

3.7 Deployment to AWS



tharindu_rathgamaguruge › tharindu.rathgamaguruge_private_project › Jobs › #4281

```
d04b35da41fa Extracting [============================================>      ]  5.636MB/6.391MB
d04b35da41fa Extracting [================================================>  ]  6.226MB/6.391MB
d04b35da41fa Extracting [==================================================>]  6.391MB/6.391MB
d04b35da41fa Pull complete
api_gateway_service Pulled
Network tharindurathgamaguruge_private_project_mynetwork  Creating
Network tharindurathgamaguruge_private_project_mynetwork  Created
Container tharindurathgamaguruge_private_project-rabbitmq-1  Creating
Container tharindurathgamaguruge_private_project-rabbitmq-1  Created
Container tharindurathgamaguruge_private_project-service2-1  Creating
Container tharindurathgamaguruge_private_project-api_gateway_service-1  Creating
Container tharindurathgamaguruge_private_project-service1-1  Creating
Container tharindurathgamaguruge_private_project-monitoring_service-1  Creating
Container tharindurathgamaguruge_private_project-service1-1  Created
Container tharindurathgamaguruge_private_project-api_gateway_service-1  Created
Container tharindurathgamaguruge_private_project-monitoring_service-1  Created
Container tharindurathgamaguruge_private_project-service2-1  Created
Container tharindurathgamaguruge_private_project-rabbitmq-1  Starting
Container tharindurathgamaguruge_private_project-rabbitmq-1  Started
Container tharindurathgamaguruge_private_project-service1-1  Starting
Container tharindurathgamaguruge_private_project-monitoring_service-1  Starting
Container tharindurathgamaguruge_private_project-service2-1  Starting
Container tharindurathgamaguruge_private_project-api_gateway_service-1  Starting
Container tharindurathgamaguruge_private_project-service1-1  Started
Container tharindurathgamaguruge_private_project-monitoring_service-1  Started
Container tharindurathgamaguruge_private_project-service2-1  Started
Container tharindurathgamaguruge_private_project-api_gateway_service-1  Started
Cleaning up project directory and file based variables                                    00:00
Job succeeded
```

Duration: 29 seconds
Finished: 51 minutes ago
Queued: 1 second
Timeout: 1h (from project)
Runner: #106 (mF2szrZKb)
Tags: dcthra-aws

Commit b39a3c6f
Update .gitlab-ci.yml file

Pipeline #1359 Passed for project

deploy

Related jobs

deploy

→ deploy-aws

Figure: Deployment to AWS instance stage

4. Reflections

Main learnings and worst difficulties

1. Shutdown the services
   Firstly, I implemented shutdown method by writing process kill functions in each service and sending an API to gateway service. However, the test was failed due to service1(implemented using python) killed before sending a response back to the API gateway service. Furthermore, could not implement any method to stop the RabbitMQ service. Then I used subprocess library in python. From this we can stop the running docker containers in the host (We need to mount host /var/run/docker.sock to communicate with the host docker system). Also, I had to add volume mount into the config.toml

   ```
   volumes = ["/var/run/docker.sock:/var/run/docker.sock", "/cache"]
   ```

2. AWS deployment implementation

   AWS deployment implementation can be done through different approaches.

   2.1 Directly ssh into the ec2 instance and clone the repository then run the services with docker compose. Here we need to build the services again in the cloud is a drawback of this implementation.
   2.2 Pushing images into the ECR and running the services on EC2 instance using docker-compose. For this implementation, I used bash script to push images to the ECR. This implementation can be more automated with Ansible.

3. Test cases execute methods
   Firstly, I executed test cases inside the api_gateway_service docker and after understanding the behavior in git CI executed test cases using python image in the test stage.

Amount effort (hours) used

Around 56 hours.

Reference Links

[1]- https://docs.sonarsource.com/sonarqube/9.8/user-guide/user-account/generating-and-using-tokens/