

DEVOPS

PROJECT DOCUMENT

STUDENT NAME: CHU DUC ANH

STUDENT NUMBER: 50358922

1	Project overview.....	1
1.1	System requirement.....	1
1.2	Instruction to run locally.....	1
1.3	Testing functionalities locally.....	2
2	Implementation.....	2
2.1	Gitlab.....	2
2.2	Functionalities for API gateways.....	3
2.3	Test cases for API functionalities.....	5
2.4	CPouta server.....	5
3	Description of CI/CD pipeline.....	5
3.1	Version management.....	5
3.2	Pipeline.....	6
4	Runs of pipeline.....	6
5	Reflection.....	11
5.1	Main learning.....	11
5.2	Worst difficulties.....	11

1 Project overview

This DevOps project continue from previous nginx exercise which further implement CI/CD pipeline using gitlab-runner. Furthermore, this project also develops based test-driven development method and add multiple API functionalities.

1.1 System requirement

The current option that is used for the operating system is virtual box. The virtual environment is given with these hardware specifications:

- Processor: 6-core CPU
- Memory: 19GB
- Storage: 60 GB

Software

Operating system

- Ubuntu 20.04.6

Development tools:

- Frontend: Flask
- Backend: Flask, Express.js
- Test framework: pytest

Additional tools:

- Docker 27.5.1
- Nginx 1.18.0

1.2 Instruction to run locally

Clone the repository from this link:

<https://compse140.devops-gitlab.rd.tuni.fi/chuducanh242002/devops>

Then switch to project branch using this command:

```
git switch project
```

Manually testing the server with docker-compose without using the test's cases:

```
docker-compose build or docker compose build
```

```
docker-compose up --scale tests=0 Or docker compose up --scale tests =0
```

The server will listen at localhost:8198

The server can also be tested with curl command as well:

```
curl localhost:8198
```

1.3 Testing functionalities locally

In order to test the functionalities locally with curl, the cookies.txt file is needed. There is curl-config.txt file there and it will use the cookies.txt. Use the curl function:

```
curl -K curl-config.txt http://localhost:8197/
```

To get the state:

```
curl -K curl-config.txt http://localhost:8197/state
```

To change the state:

```
curl -K curl-config.txt http://localhost:8197/state -X PUT -d "RUNNING" -H "Content-Type: text/plain" -H "Accept: text/plain"
```

2 Implementation

2.1 Gitlab

First, switch to project branch with exercise 4. Connect to gitlab devops project:
<https://compse140.devops-gitlab.rd.tuni.fi/chuducanh242002/devops>.

Next create pipeline infrastructure using gitlab-ci. I added gitlab remote repository to my original github repository.

```
vboxuser@user:~/project/DevOps$ git remote -v
gitlab https://compse140.devops-gitlab.rd.tuni.fi/chuducanh242002/devops.git (fetch)
gitlab https://compse140.devops-gitlab.rd.tuni.fi/chuducanh242002/devops.git (push)
origin git@github.com:ChuDucAnh242002/DevOps.git (fetch)
origin git@github.com:ChuDucAnh242002/DevOps.git (push)
```

Gitlab runner is installed, registered and run in gitlab with tags token 1.

Assigned project runners



2.2 Functionalities for API gateways

The user can access the webpage by this url: <http://localhost:8198>. The server exports to port 8198

POST / : This request require the user to enter username and password

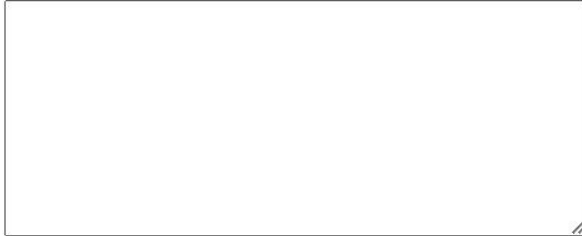
A screenshot of a 'Sign in' form. At the top, it says 'Sign in' and shows the URL 'http://localhost:8198'. Below this are two input fields: 'Username' and 'Password'. At the bottom of the form are two buttons: a blue 'Sign in' button and a light blue 'Cancel' button.

GET / : This request provide the user with a template which has request and stop button

Home

REQUEST

STOP



GET /request : This request show the service's informations such as disk space, ip address, running processes, and uptime.

Home

REQUEST

STOP

```
{
  "service 1": {
    "Disk Space": {
      "free": 15927631872,
      "percent": 65.6,
      "total": 48506511360,
      "used": 30320324608
    },
    "IP Address": "172.18.0.2",
    "Running Processes": [
      {
        "name": "python",
        "pid": 1
      }
    ],
    "Uptime (seconds)": 15260.635394096375
  },
  "service 2": {
    "Disk Space": {
      "free": 16062799872,
      "percent": 19.222450724974415,
      "total": 19885228032,
      "used": 3822428160
    },
    "IP Address": "172.18.0.5",
    "Running Processes": [

```

POST /stop : This request stop all the services including nginx and return to the cli.

Home

REQUEST

STOP



```
service1 > app.py
90 def home():
PROBLEMS OUTPUT DEBUG CONSOLE PORTS 4
nginx | /docker-entrypoint.sh
efault.sh |
nginx | 10-listen-on-ipv6-by-
nf.d/default.conf | 10-listen-on-ipv6-by-
nginx | /docker-entrypoint.sh
conf.d/default.conf |
nginx | /docker-entrypoint.sh
h |
nginx | /docker-entrypoint.sh
es.sh | /docker-entrypoint.sh
nginx | /docker-entrypoint.sh
es.sh | /docker-entrypoint.sh
nginx | /docker-entrypoint.sh
service1_instance1 | 172.18.0.6 - - [20/F
nginx | 172.18.0.1 - user1 [2
"Mozilla/5.0 (Windows NT 10.0; Win64; x64)
0 Safari/537.36"
nginx exited with code 137
service2_container exited with code 137
service1_instance2 exited with code 137
service1_instance3 exited with code 137
service1_instance1 exited with code 137
vboxuser@user:~/project/DevOps$
```

2.3 Test cases for API functionalities

There are multiple test cases. Each of them will test for each API functionalities. Nginx also listen to port 8197 and provides with new API functionalities for tests only as well.

Test case for POST / will check whether the page return the correct html.

Test case for GET /request will check whether the response is in correct format and provide all the necessary information for each services.

Test case for GET /state will test after POST / and get the “RUNNING” state.

Test case for PUT /state will test all the possible state after setting the state.

Test case for GET /run-log will get the log from INIT->RUNNING when first login

Test case for paused state which can't make and requests during pause state.

2.4 CPouta server

The CPouta server is initilized and running. The image uses Ubuntu-22.04 with the public ip address: 195.148.21.49

3 Description of CI/CD pipeline

3.1 Version management

Version management: git 2.25.1

Branches:

- exercise1: This branch includes the first exercise which has basic requests functionalities of 2 services
- exercise4: This branch includes the nginx exercise which run nginx load balancer with 3 services 1 instances and also has additional stop function.
- project: This branch includes gitlab CI pipeline, test cases and new API functionalities.

3.2 Pipeline

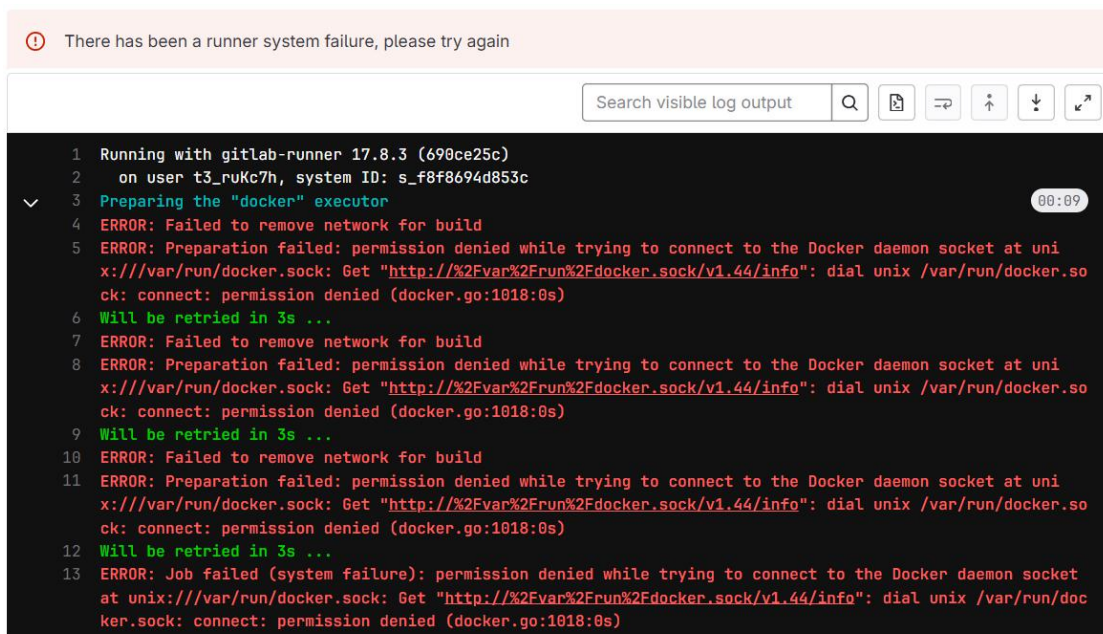
There are 3 stages in Gitlab pipeline, build, test and deploy. The gitlab runner uses docker in docker with the service's name docker:20.10.21-dind.

Build: bulid will use docker-compose bulid to build the in the current directory. The image used to build is docker:20.10.21.

Test: The test will first docker-compose up --scale tests=0 in order for the tests container to not run. After all other services are running, the tests container will run with command docker-compose run --rm tests. Pytest will look for all the test's cases and execute them one by one. After all the tests passed, docker-compose down to shut down all the docker containers and also delete all the containers.

deploy: The deploy stage will try to copy the key to the docker container which uses image alpine:latest. The docker container will access to the CPouta virtual machine and deploy locally within the server.

4 Runs of pipeline



```
1 Running with gitlab-runner 17.8.3 (690ce25c)
2 on user t3_ruKc7h, system ID: s_f8f8694d853c
3 Preparing the "docker" executor
4 ERROR: Failed to remove network for build
5 ERROR: Preparation failed: permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdocker.sock/v1.44/info": dial unix /var/run/docker.sock: connect: permission denied (docker.go:1018:0s)
6 Will be retried in 3s ...
7 ERROR: Failed to remove network for build
8 ERROR: Preparation failed: permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdocker.sock/v1.44/info": dial unix /var/run/docker.sock: connect: permission denied (docker.go:1018:0s)
9 Will be retried in 3s ...
10 ERROR: Failed to remove network for build
11 ERROR: Preparation failed: permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdocker.sock/v1.44/info": dial unix /var/run/docker.sock: connect: permission denied (docker.go:1018:0s)
12 Will be retried in 3s ...
13 ERROR: Job failed (system failure): permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdocker.sock/v1.44/info": dial unix /var/run/docker.sock: connect: permission denied (docker.go:1018:0s)
```

This is the first build in the gitlab CI pipeline, it indicates that it can't access the docker sock. The way to solve it is to specify docker in docker as service


```

a36b77f18ccb03c3f32a6385e3306e289e9ddbfbcfce
260 #14 ...
261 #23 [devops-service1_instance3] exporting to image
262 #23 exporting layers 0.4s done
263 #23 exporting layers 0.4s done
264 #23 writing image sha256:811e3be891216277f37411e703033dd28f4b3426e8287168a9f098919d2696d4 0.0s done
265 #23 naming to docker.io/library/devops-service1_instance1 0.0s done
266 #23 writing image sha256:f91a68acefdb61cc63a9bc21611ff52158957b58ae4a3aebcca1bab524aeb047 0.0s done
267 #23 naming to docker.io/library/devops-service1_instance3 0.0s done
268 #23 writing image sha256:5f0708723ade0ee7258964d30ea297dc769d7f9ad7eeb15e7cb4b284f037d872
269 #23 writing image sha256:5f0708723ade0ee7258964d30ea297dc769d7f9ad7eeb15e7cb4b284f037d872 0.0s done
270 #23 naming to docker.io/library/devops-service1_instance2 done
271 #23 DONE 1.0s
272 #14 [devops-service1_instance1 1/4] FROM docker.io/library/python:3.8.10-slim@sha256:6c0b171f6e4cbd880a972
a36b77f18ccb03c3f32a6385e3306e289e9ddbfbcfce
✓ 273 Uploading artifacts for successful job 00:02
274 Uploading artifacts...
275 docker-compose.yml: found 1 matching artifact files and directories
276 Uploading artifacts as "archive" to coordinator... 201 Created id=14312 responseStatus=201 Created token=
glcvt-64
✓ 277 Cleaning up project directory and file based variables 00:01
278 Job succeeded

```

First build success

```

✓ 18 Downloading artifacts 00:01
19 Downloading artifacts for build (14312)...
20 Downloading artifacts from coordinator... ok host=compse140.devops-gitlab.rd.tuni.fi id=14312 respo
nseStatus=200 OK token=glcvt-64
✓ 21 Executing "step_script" stage of the job script 00:02
22 Using docker image sha256:81a4721a045e389ed8754844971b3277b59757fe75c69120e6cea0dbc8de6381 for docker:20.1
0.21 with digest docker@sha256:812ff1098c97878ae26ca8e0423e00aa157596a8b28d4bd52c120c4d087e8ae1 ...
23 $ docker login -u "$CI_REGISTRY_USER" -p "$CI_REGISTRY_PASSWORD" $CI_REGISTRY
24 WARNING! Using --password via the CLI is insecure. Use --password-stdin.
25 Login Succeeded
26 WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
27 Configure a credential helper to remove this warning. See
28 https://docs.docker.com/engine/reference/commandline/login/#credentials-store
29 $ docker pull $DOCKER_TAG
30 "docker pull" requires exactly 1 argument.
31 See 'docker pull --help'.
32 Usage: docker pull [OPTIONS] NAME[:TAG|@DIGEST]
33 Pull an image or a repository from a registry
✓ 34 Cleaning up project directory and file based variables 00:01
35 ERROR: Job failed: exit code 1

```

First test fail

The test was trying to pull the image but it didn't work

```

$ echo "Running unit tests"
Running unit tests
$ docker-compose run --rm tests
Container service1_instance3 Running
Container service2_container Running
Container service1_instance2 Running
Container service1_instance1 Running
Container nginx Recreate
Container nginx Recreated
Container nginx Starting
Container nginx Started
===== test session starts =====
platform linux -- Python 3.8.10, pytest-8.3.4, pluggy-1.5.0
rootdir: /tests
collected 3 items
test_api.py ... [100%]
===== 3 passed in 2.21s =====

```

First test pass

This test contain 3 simple API test which test GET request from localhost:8198 and GET from localhost:8198/request

After first API tests success, the tests were tested locally and push to pipeline. All the test cases are mentioned above.

```

$ echo "Pulling the latest images..."
Pulling the latest images...
$ docker-compose -f docker-compose.yml pull
service1_instance2 Skipped - No image to be pulled
service1_instance3 Skipped - No image to be pulled
service1_instance1 Skipped - No image to be pulled
nginx Pulling
service2 Pulling
service2 Warning
nginx Error
WARNING: Some service image(s) must be built from source by running:
  docker compose build %s service2
Error response from daemon: pull access denied for service2, repository does not exist or may require 'docker login': denied: requested access to the resource is denied
Cleaning up project directory and file based variables
ERROR: Job failed: exit code 18

```

First failed deploy

This time, I still use docker pull image like the first failed test above.

```

Container service2-container Started
$ echo "Checking the status of the deployed services..."
Checking the status of the deployed services...
$ docker-compose -f docker-compose.yml ps
NAME                COMMAND                  SERVICE           STATUS          PORTS
devops-nginx-1      "/docker-entrypoint..." nginx             running         0.0.0.0:8198->80/tcp
p, :::8198->80/tcp
service1_instance1  "python app.py"         service1_instance1 running         5000/tcp
service1_instance2  "python app.py"         service1_instance2 running         5000/tcp
service1_instance3  "python app.py"         service1_instance3 running         5000/tcp
service2-container  "docker-entrypoint.s..." service2          running         3000/tcp
$ echo "Deployment completed successfully!"
Deployment completed successfully!
Uploading artifacts for successful job
Uploading artifacts...
docker-compose.yml: found 1 matching artifact files and directories
Uploading artifacts as "archive" to coordinator... 201 Created id=14382 responseStatus=201 Created token=
glcblt-64
Cleaning up project directory and file based variables
Job succeeded

```

First deploy success locally in gitlab

```

st with digest alpine@sha256:a8560b36e8b8210634f77d9f7f9efd7ffa463e380b75e2e74aff4511df3ef88c ...
5 $ command -v ssh-agent >/dev/null || ( apk update && apk add openssh-client )
6 fetch https://dl-cdn.alpinelinux.org/alpine/v3.21/main/x86_64/APKINDEX.tar.gz
7 fetch https://dl-cdn.alpinelinux.org/alpine/v3.21/community/x86_64/APKINDEX.tar.gz
8 v3.21.3-38-g5e127436704 [https://dl-cdn.alpinelinux.org/alpine/v3.21/main]
9 v3.21.3-37-g1360d453ec3 [https://dl-cdn.alpinelinux.org/alpine/v3.21/community]
10 OK: 25394 distinct packages available
1 (1/6) Installing openssh-keygen (9.9_p2-r0)
2 (2/6) Installing ncurses-terminfo-base (6.5_p20241006-r3)
3 (3/6) Installing libncursesw (6.5_p20241006-r3)
4 (4/6) Installing libedit (20240808.3.1-r0)
5 (5/6) Installing openssh-client-common (9.9_p2-r0)
6 (6/6) Installing openssh-client-default (9.9_p2-r0)
7 Executing busybox-1.37.0-r12.trigger
8 OK: 11 MiB in 21 packages
9 $ mkdir -p ~/.ssh
10 $ chmod 700 ~/.ssh
11 $ echo "$SSH_PRIVATE_KEY" | ~/.ssh/id_rsa
12 /bin/sh: eval: line 161: /root/.ssh/id_rsa: not found
13 Cleaning up project directory and file based variables
14 ERROR: Job failed: exit code 127

```

Failed deploy attempt to the Cpouta server


```

$ echo "Deploying to cPouta virtual machine..."
Deploying to cPouta virtual machine...
$ ssh -v -o StrictHostKeyChecking=no -i ~/.ssh/DevOps.pem $SERVER_USER@$SERVER_IP
OpenSSH_9.9p2, OpenSSL 3.3.3 11 Feb 2025
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: /etc/ssh/ssh_config line 22: include /etc/ssh/ssh_config.d/*.conf matched no files
Pseudo-terminal will not be allocated because stdin is not a terminal.
debug1: Connecting to 195.148.21.49 [195.148.21.49] port 22.
debug1: Connection established.
debug1: identity file /root/.ssh/DevOps.pem type 0
debug1: identity file /root/.ssh/DevOps.pem-cert type -1
debug1: Local version string SSH-2.0-OpenSSH_9.9
debug1: Remote protocol version 2.0, remote software version OpenSSH_8.9p1 Ubuntu-3ubuntu0.11
debug1: compat_banner: match: OpenSSH_8.9p1 Ubuntu-3ubuntu0.11 pat OpenSSH* compat 0x04000000
debug1: Authenticating to 195.148.21.49:22 as 'ubuntu'
debug1: load_hostkeys: fopen /root/.ssh/known_hosts: No such file or directory
debug1: load_hostkeys: fopen /root/.ssh/known_hosts2: No such file or directory
debug1: load_hostkeys: fopen /etc/ssh/ssh_known_hosts: No such file or directory
debug1: load_hostkeys: fopen /etc/ssh/ssh_known_hosts2: No such file or directory
debug1: SSH2_MSG_KEXINIT sent
debug1: SSH2_MSG_KEXINIT received
debug1: kex: algorithm: sntrup761x25519-sha512@openssh.com
debug1: kex: host key algorithm: ssh-ed25519
debug1: kex: server->client cipher: chacha20-poly1305@openssh.com MAC: <implicit> compression: none
debug1: kex: client->server cipher: chacha20-poly1305@openssh.com MAC: <implicit> compression: none
debug1: expecting SSH2_MSG_KEX_ECDH_REPLY
debug1: SSH2_MSG_KEX_ECDH_REPLY received

```

Attempt to connect to Cpouta server in gitlab CI.

```

97 debug1: client_input_global_request: rtype hostkeys-00@openssh.com want_reply 0
98 debug1: Remote: /home/ubuntu/.ssh/authorized_keys:1: key options: agent-forwarding port-forwarding pty use
r-rc x11-forwarding
99 debug1: Remote: /home/ubuntu/.ssh/authorized_keys:1: key options: agent-forwarding port-forwarding pty use
r-rc x11-forwarding
100 debug1: Sending command: cd /home/ubuntu/DevOps && docker-compose up -d
101 debug1: pledge: fork
102 service1_instance3 is up-to-date
103 service1_instance2 is up-to-date
104 service1_instance1 is up-to-date
105 service2_container is up-to-date
106 nginx is up-to-date
107 Starting tests ...
108 Starting tests ... done
109 debug1: client_input_channel_req: channel 0 rtype exit-status reply 0
110 debug1: channel 0: free: client-session, nchannels 1
111 Transferred: sent 4184, received 4444 bytes, in 0.9 seconds
112 Bytes per second: sent 4519.7, received 4888.5
113 debug1: Exit status 0
114 Cleaning up project directory and file based variables
115 Job succeeded

```

Successfully deploy locally in Cpouta server in gitlab CI

I am not able to deploy the public server and stop at deploy locally in Cpouta.

5 Reflection

5.1 Main learning

There is lots of lessons that I have learned through out developing this project. Firstly, I learn really clearly how docker, docker compose and nginx work and how they work together. I learned how implement stop functionality which I wasn't able to do in previous exercise. I learned how to set up gitlab runner and gitlab CI in all 3 stages: build, test, deploy. Moreover, during the CI process, I learned that building, testing, and deploying locally are so much different and much easier than in gitlab CI pipeline. I learn how to test API test using pytest. I learned how to create CPouta server and connect to the server through SSH. I learned how to deploy the CPouta server but I haven't been able to finish it yet.

5.2 Worst difficulties

The difficulty that takes my time the most are the STOP functionality and how to test it. In my pytest, if I stop all the containers then my test container will also stop and I wouldn't able to test it at all. Therefore either I have to test it after every other test or I have to try to figure out how to not stop test container. I actually haven't figured it yet.

The second difficulty is I wasn't able to deploy in CPouta server and export it at its public IP address. I was able to set up apache2 server and able to deploy locally but it doesn't work when I deploy publicly. I still don't know the reason yet for that issue as well.

Overall, the project is hard but I gain lots of lessons from building gitlab CI, making API functionalities and deploying to cloud virtual machine. I estimate my used amount effort is around 60 hours.