

Files provided for this project

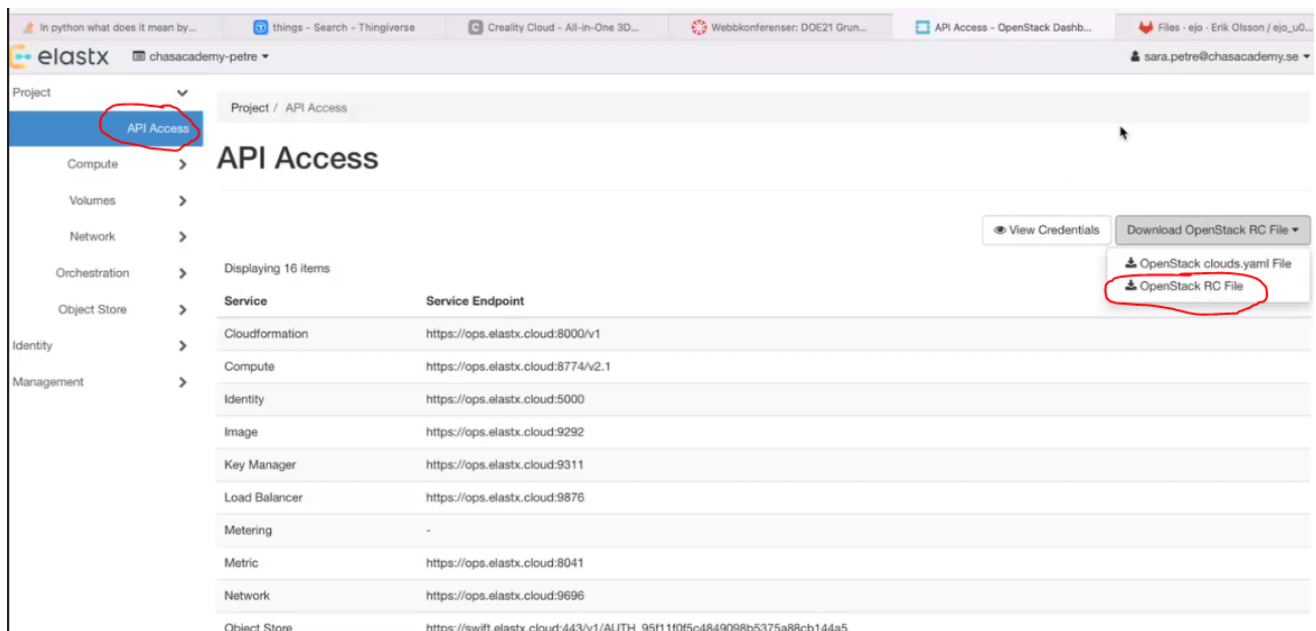
- main.tf
- inventory.yml
- caprov-playbook.yml
- config.json
- README.md (pdf)

Create a directory with the above files added.

- Fork the test repo down below to be able to show the CapRover GitLab app deployed from Gitlab
https://gitlab.com/SaraPetre/u08_caprover_gitlab

elastx

- Log in to your account with your email and provided password
- Go into API Access.
- Go to Download Openstack RC files



- Move your downloaded RC-file into your file-directory

VScode

-open VScode in your directory with the above files.

- open your config.json-file and add your personal Domain, new password, email and caproverName.

```

1 {
2   "caproverIP": "192.168.1.101",
3   "caproverPassword": "captain42", [default password]
4   "caproverRootDomain": "caprover.ejo.one", [your domain]
5   "newPassword": "testpass", [your password]
6   "certificateEmail": "Erikjo2001@gmail.com", [add email for cerificate]
7   "caproverName": "cap-server"
8 }

```

In terminal:

- source 'chasacademy-petre-openrc.sh' (your downloaded file)
- copy and add your elastx password in the terminal

Terraform

In terminal:

- terraform init
- terraform apply
 - yes (to approve the plan an go ahead with apply)
- chmod 600 (on your keypair-file just created)

elastx

- Go back into elastx and see that your setup is completed
- Under instances copy your ipadresses from your server and worker and insert them into the server and worker in **inventory.yml**.

Project

API Access

Compute

Overview

Instances

Images

Key Pairs

Server Groups

Volumes

Network

Orchestration

Project / Compute / Instances

Instances

Instance ID =

Displaying 2 items

<input type="checkbox"/>	Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	
<input type="checkbox"/>	caprover-worker	ubuntu-22.04-server-20220607	192.168.1.102, 91.197.41.188	v1-c4-m8-d120	aras_u08_repo_keypair	Active	🔒
<input type="checkbox"/>	caprover-server	ubuntu-22.04-server-20220607	192.168.1.101, 91.197.41.163	v1-c4-m8-d120	aras_u08_repo_keypair	Active	🔒

Displaying 2 items

```
main.tf  ! inventory.yml x  ! caprov-playbook.yml  $ setup.sh  {} config.json

! inventory.yml
1  all:
2    children:
3      caprov_server:
4        hosts:
5          91.197.41.163:
6            ansible_user: ubuntu
7            ansible_ssh_private_key_file: aras_u08_repo_keypair
8
9      caprov_workers:
10       hosts:
11         91.197.41.188:
12           ansible_user: ubuntu
13           ansible_ssh_private_key_file: aras_u08_repo_keypair
```

Caprover cluster

- Navigate to CapRover and log in. (in my case <https://captain.aras.ejo.one/>)
- Go to Cluster and scroll down to "Alternative Method":
- Follow the instructions and run the commands in your terminal in VSCode.

The screenshot shows the CapRover web interface. On the left is a sidebar with navigation links: Dashboard, Apps, Monitoring, Cluster (selected), and Settings. The main content area is titled 'Cluster' and contains a form for adding a new node. At the top, there are input fields for 'New node IP Address' (123.123.123.123) and 'CapRover IP Address' (91.197.41.163). Below these is a section for 'SSH Private Key for root' with a text area containing a placeholder key. There are buttons for 'Join as worker node' and 'Join as manager node'. Further down, there are input fields for 'SSH Port' (22) and 'SSH User' (root), followed by a 'Join C' button. A section titled 'Alternative Method' is expanded, showing instructions on how to join a cluster manually using SSH and Docker Swarm. The instructions include running a command on the main leader node and then on the worker node, and mention that the command might need to be modified based on network configurations.

In terminal.

SSH:a into server: Run:

- `sudo docker swarm join-token worker`

Take the output: In my case:

- `docker swarm join --token SWMTKN-1-4mp0om0q2vxzvjq4zlaitcqm19hh6vwf4uxm1oxlksuucxkj4-1eepwoybp0iih0kknle9fjugu 91.197.41.163:2377`

Logout from server:

SSH:a into runner:

Run the above output command:

- `sudo docker swarm join --token SWMTKN-1-4mp0om0q2vxzvjq4zlaitcqm19hh6vwf4uxm1oxlksuucxkj4-1eepwoybp0iih0kknle9fjugu 91.197.41.163:2377`

Output

- T- his node joined a swarm as a worker.

Working =)

Navigate back to your Caprover: In my case: <https://captain.aras.ejo.one/#/login>

- Log in with your password

Navigate to cluster. You can now see that you are clustered. Se down below picture:

Current Cluster Nodes											
<div> <p>Node ID: g6uhmnp21v1pj1ojmnk9gdflg</p> <hr/> <table> <tr> <td>Type: Leader (Main Node)</td><td>IP: 91.197.41.163</td></tr> <tr> <td>State: ready</td><td>Status: active</td></tr> <tr> <td>RAM: 7.76 GB</td><td>OS: linux</td></tr> <tr> <td>CPU: 4 cores</td><td>Architecture: x86_64</td></tr> <tr> <td>Hostname: caprover-server</td><td>Docker Version: 20.10.12</td></tr> </table> </div>		Type: Leader (Main Node)	IP: 91.197.41.163	State: ready	Status: active	RAM: 7.76 GB	OS: linux	CPU: 4 cores	Architecture: x86_64	Hostname: caprover-server	Docker Version: 20.10.12
Type: Leader (Main Node)	IP: 91.197.41.163										
State: ready	Status: active										
RAM: 7.76 GB	OS: linux										
CPU: 4 cores	Architecture: x86_64										
Hostname: caprover-server	Docker Version: 20.10.12										
<div> <p>Node ID: v0uvteeyaqf56tyerlnkxklco</p> <hr/> <table> <tr> <td>Type: worker</td><td>IP: 91.197.41.188</td></tr> <tr> <td>State: ready</td><td>Status: active</td></tr> <tr> <td>RAM: 7.76 GB</td><td>OS: linux</td></tr> <tr> <td>CPU: 4 cores</td><td>Architecture: x86_64</td></tr> <tr> <td>Hostname: caprover-worker</td><td>Docker Version: 20.10.12</td></tr> </table> </div>		Type: worker	IP: 91.197.41.188	State: ready	Status: active	RAM: 7.76 GB	OS: linux	CPU: 4 cores	Architecture: x86_64	Hostname: caprover-worker	Docker Version: 20.10.12
Type: worker	IP: 91.197.41.188										
State: ready	Status: active										
RAM: 7.76 GB	OS: linux										
CPU: 4 cores	Architecture: x86_64										
Hostname: caprover-worker	Docker Version: 20.10.12										




Gitlab deployed on CapRover




<https://caprover.com/docs/ci-cd-integration/deploy-from-gitlab.html>

!!! In this project a test repo is added to use, which is to be forked, with the files needed. You can therefore yump to part 3. !!!



1. Create GitLab Repository




SaraPetre > u08_caprover_gitlab







**u08_caprover_gitlab** 
Project ID: 40801374 


  Star 0  Fork 0







10 Commits 1 Branch 0 Tags 597 KB Project Storage


master u08_caprover_gitlab /  Find file Web IDE  Clone

 **Update README.md**
SaraPetre authored 1 minute ago  81d5764e 

 README  CI/CD configuration  Add LICENSE  Add CHANGELOG  Add CONTRIBUTING  Add Kubernetes cluster

 Configure Integrations

Name	Last commit	Last update
 images	added readme.md and images	24 minutes ago
 .gitignore	added files for test	1 day ago
 .gitlab-ci.yml	gitlab-ci.yml file added	4 days ago
 Dockerfile	removed part of Dockerfile	1 day ago
 README.md	Update README.md	1 minute ago
 index.php	removed all not used files	43 minutes ago

 **README.md**

2. Add sample Source code-file, Dockerfile and a .gitlab-ci.yml-file. The content ara copied from the documnet:

<https://caprover.com/docs/ci-cd-integration/deploy-from-gitlab.html>

The files and content can be seen in this repo.

3. Create CI/CD Variables

Go to your project page on GitLab.

Navigate to Settings > CI/CD.

In Variables add the following variables:

- Key : CAPROVER_URL , Value : <https://captain.root.domain.com> [replace it with your domain]
- Key : CAPROVER_PASSWORD , Value : mYpAsSwOrD [replace it with your password]
- Key : CAPROVER_APP , Value : my-test-gitlab-deploy [replace it with the app name you want to create]

4. Create an Access Token for CapRover

Navigate to https://gitlab.com/-/profile/personal_access_tokens and create a token.

Make sure to assign read_registry and write_registry permissions for this token.

5. Add Token to CapRover

Login to your CapRover web dashboard, under Cluster click on Add Remote Registry. Then enter these fields:

- Username: your gitlab username
- Password: your gitlab Token [From the previous step]
- Domain: registry.gitlab.com
- Image Prefix: again, your gitlab username **!!!! I needed this to be blanc for it to work!!!**

6. Disable Default Push

Now that you added a registry, CapRover by default wants to push the built artifact to your registry. You do not need this for this project, and it might make your deployments to fail. So go ahead and disable Default Push

!!!! I did not disable Default push and it worked!

7. Create a CapRover App

On CapRover "Apps" and create an app:

- aras-gitlab-deploy (in my case. You need to add the name that you set up in part 3. CAPROVER_APP value)

8. Push to your repo

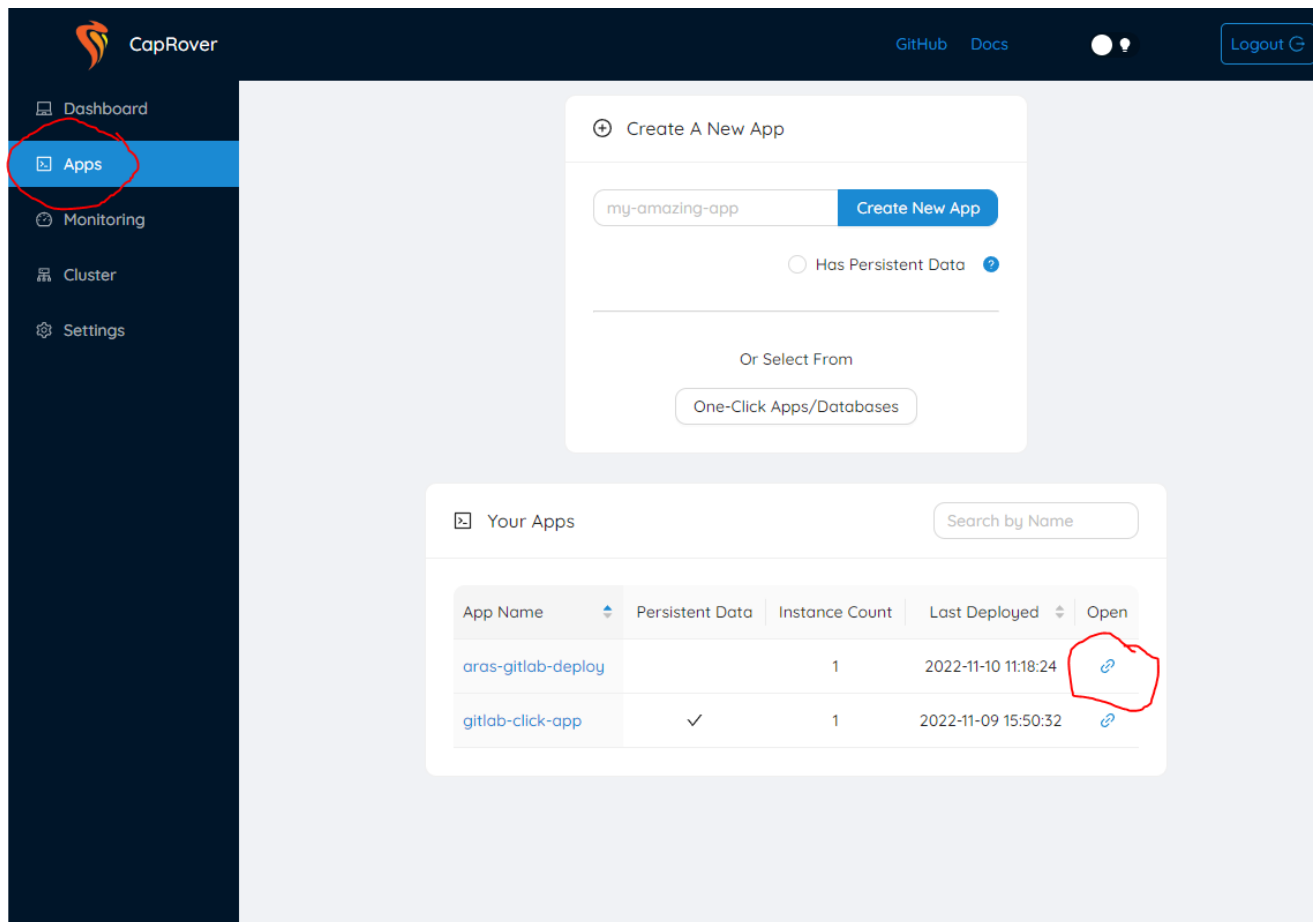
From VSCode and terminal:

- Make some changes in the index.php-file to trigger the push.
- commit and push to your repo. Make sure to commit to master

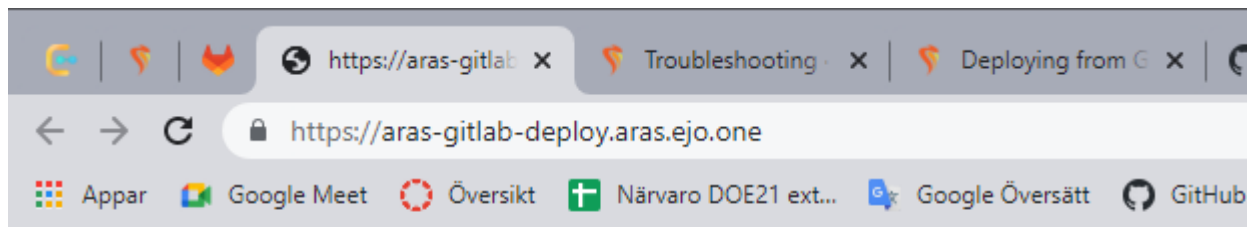
Wait a little bit until your build is finished and deployed automatically! After a few minutes you can be able to see your deployed app on CapRover!

Open CapRover. Log in if needed.

- Navigate to "Apps"
- Click on open on your app.



The output from the index.php-file can now be seen:



PHP output: Hello World from Sara! This is working now=) And today as well #3/221110