|  |
| --- |
| **DEVOPS PORTFOLIO: COOKBOOK** |

*HOW TO => enkel zeer specifieke voorbeelden met output (gebaseerd op de andere documenten)*

Table of Contents

[HOW TO: Python installeren 2](#_Toc93991214)

[HOW TO: Git initialiseren en lokale repository aanmaken 2](#_Toc93991215)

[HOW TO: Document stagen en committen in lokale repository 2](#_Toc93991216)

[HOW TO: Branch aanmaken in jouw repository en in deze branch een aanpassing brengen 3](#_Toc93991217)

[HOW TO: Lokale repository online plaatsen 3](#_Toc93991218)

[HOW TO: Eenvoudige Python script schrijven en runnen via de terminal 4](#_Toc93991219)

[HOW TO: REST API gebruiken met Python 5](#_Toc93991220)

[HOW TO: Bash script aanmaken en runnen, web applicatie aanmaken en runnen 6](#_Toc93991221)

[HOW TO: Dockerfile aanmaken, Docker container opbouwen en runnen 7](#_Toc93991222)

[HOW TO: Jenkins Docker Image downloaden en runnen 8](#_Toc93991223)

[HOW TO: Creëer een Pipeline in Jenkins 8](#_Toc93991224)

[HOW TO: Ansible configureren 9](#_Toc93991225)

[HOW TO: Ansible gebruiken om configuratie op te slaan als backup 10](#_Toc93991226)

[HOW TO: Ansible gebruiken om een toestel te configureren 11](#_Toc93991227)

# HOW TO: Python installeren

Gebruik command **venv** i.p.v. **pip3 install** (er kunnen bij pip3 install problemen ontstaan met dependencies).

* **python3 -m venv devfun** (devfun is de gekozen naam van de enviroment)

# HOW TO: Git initialiseren en lokale repository aanmaken

Met **git config** kan ik user information configureren zodat mijn werk gelinkt zal worden aan deze mail/username.

Met **mkdir** maak ik een directory (map) aan die zal zorgen voor mijn lokale repo.

Om van deze directory effectief een repository te maken gebruik je **git init** (lokaal)

Gebruik command **git status** om een status overzicht te krijgen van de repository

Text

Description automatically generated

# HOW TO: Document stagen en committen in lokale repository

Gebruik command **git add *file\_naam*** om een file te stagen

Gebruik command **git commit *file\_naam* -m “*korte beschrijving*”**om file te committen met message

Gebruik command **git status** om status na te kijken

# HOW TO: Branch aanmaken in jouw repository en in deze branch een aanpassing brengen

Gebruik command **git branch *branchnaam*** om een nieuwe branch aan te maken

Gebruik command **git branch** om een overzicht te krijgen van alle branches

Gebruik command **git checkout *branchnaam*** om naar deze branch te switchen

Gebruik command **git merge *branchnaam*** om deze branch te mergen met de master branch

Text

Description automatically generated

# HOW TO: Lokale repository online plaatsen

Gebruik command **git remote add *origin https://adres*** om te wijzen naar jouw Github repo

Gebruik command **git push origin master** om de lokale files te pushen naar jouw Github repo

# HOW TO: Eenvoudige Python script schrijven en runnen via de terminal

Maak een Python file aan en schrijf hier uw script

Run de script via terminal door command **python3 *filenaam***

Text

Description automatically generated

# HOW TO: REST API gebruiken met Python

Maak een nieuwe python file aan (.py)

Importeer de nodige 🡪 requests, urllib.parse

Text

Description automatically generated

Gebruik command **python3 *filenaam*** om de script te runnen van de applicatie die gebruik maakt van de API

Druk **Ctrl+C** om het programma te beëindigen

Text

Description automatically generated

# HOW TO: Bash script aanmaken en runnen, web applicatie aanmaken en runnen

Schrijf een basic bash script (.sh)

Gebruik command **bash *filenaam*** om de bash script te runnen

Text

Description automatically generated

Maak een simpele Sample web applicatie (in Python)

Om na te kijken of de web app runt zoals het moet, schrijf <http://localhost:8080> in de web browser zelf. (Je kan dit ook via curl doen 🡪 command **curl http://0.0.0.0:8080**)

Text

Description automatically generated

# HOW TO: Dockerfile aanmaken, Docker container opbouwen en runnen

In bash script:

Gebruik command **FROM** om Python in de containter te installeren, want Python moet je in de container kunnen runnen 🡪 in bash script: **echo “FROM python” >> *naam\_directory/Dockerfile***

Gebruik command **RUN** om Flask (webapp framework in python geschreven) in de container te installeren 🡪 in bash script: **echo “RUN pip install flask” >> *naam\_directory/Dockerfile***

Gebruik command **COPY** om alle nodige website files/scripts toe te voegen in de container **🡪 echo “COPY *./folders*” >>** ***naam\_directory/Dockerfile***

Gebruik command **EXPOSE** om een port te kunnen exposen voor gebruik 🡪 **echo “EXPOSE *8080*”** **>>** ***naam\_directory/Dockerfile***

Gebruik command **CMD** om de Python script uit te voeren 🡪 **echo “CMD python3 */folder/scriptfile*”** **>>** ***naam\_directory/Dockerfile***

In terminal:

Gebruik command **cd *naam\_directory*** om naar de juiste directory van de container te gaan

Gebruik command **docker build -t *naam\_container* .** om de container op te bouwen

Gebruik command **docker run** om te container te starten/runnen 🡪 **docker run -t -d -p *8080:8080* --name *naam\_container\_instantie naam\_container\_image***

Gebruik command **docker ps -a** om alle running containers te tonen

Gebruik command **bash *./naam\_script*** om de bash script te runnen

Gebruik command **docker exec -it** ***naam\_container\_instantie* /bin/bash**om de container te kunnen exploren

Gebruik command **docker stop *naam\_container*** om de container te stoppen

Gebruik command **docker rm *naam\_container*** om de container weg te halen

# HOW TO: Jenkins Docker Image downloaden en runnen

Gebruik command **docker pull jenkins/jenkins:lts** om de laatste Jenkins container te downloaden

Gebruik command **docker run –rm -u root -p 8080:8080 -v jenkins-data:/var/jenkins\_home -v $(which docker):/usr/bin/docker -v /var/run/docker.sock:/var/run/docker.sock -v “$HOME”:/home –name jenkins\_server jenkins/jenkins:lts** om de Jenkins Docker container te starten (op éénzelfde lijn!!)

Configureer Jenkins:

In een webbrowser, navigeer naar <https://localhost:8080/> en login

Druk op ‘Install suggested plugins’

Creëer geen ‘first admin user’, skip deze stap

Druk op ‘Save and finish’ bij *Instance configuration*

Druk op ‘Start using Jenkins’

Nu kan je nieuwe jobs creëren etc.

# HOW TO: Creëer een Pipeline in Jenkins

Creëer een Pipeline job:

Druk op de Jenkins link linksboven

Druk op ‘New item’

Vul een naam in bij ‘Enter an item name’

Selecteer ‘Pipeline’ als job type

Druk ‘OK’

Configureer de net-aangemaakte job:

Trigger een build manueel door een script toe te voegen in de Pipeline sectie

Druk ‘Save’

Run de job:

Druk links op ‘Build now’ om de job te runnen

*Als alles goed verloopt zal je drie groene vierkanten zien*

Kijk de job’s output na:

Druk onder ‘Permalinks’ op de laatste link

Druk op ‘Console output’

# HOW TO: Ansible configureren

Open de *hosts* file in de *ansible-csr1000v* directory

Voeg volgende lijnen toe:

# Enter the hosts or devices for Ansible playbooks

CSR1kv ansible\_user=cisco ansible\_password=cisco123! ansible\_host=192.168.56.101

Open een terminal en ga naar de vorige (parent) directory (command **cd**)

Gebruik command **ansible** om een overzicht te krijgen van de ansible commands

Gebruik command **cat** om de default ansible.cfg file weer te geven

Gebruik command **cd** om de locatie van deze file te verplaatsen (naar *ansible-csr1000v* directory)

Wijzig de file in VS code (niet via terminal). Voeg volgende toe:

# config file for ansible-csr1000v

[defaults]

# Use local hosts file in this folder

inventory=./hosts

host\_key\_checking = False # Don’t worry about RSA Fingerprints

retry\_files\_enabled = False # Do not create them

deprecation\_warnings = False # Do not show warnings

# HOW TO: Ansible gebruiken om configuratie op te slaan als backup

Creëer een Ansible playbook (= een YAML file met één of meer plays)

Maak een nieuwe file aan in de *ansible-csr1000v* directory (in VS code)

Schrijf dit in de file:

---

– name: AUTOMATIC BACKUP OF RUNNING-CONFIG

  hosts: CSR1kv

  gather\_facts: false

  connection: local

  tasks:

   – name: DISPLAYING THE RUNNING-CONFIG

     ios\_command:

       commands:

         – show running-config

     register: config

   – name: SAVE OUTPUT TO ./backups/

     copy:

       content: “{{ config.stdout[0] }}”

       dest: “backups/show\_run\_{{ inventory\_hostname }}.txt”

*Je hebt nu één play met twee tasks*

Run de Ansible backup playbook:

Gebruik command **ping *192.168.56.101*** om de *CSR1000v virtuele machine* te pingen

Gebruik command **mkdir *backups*** om de backups directory aan te maken waar de backup configuratie zal worden opgeslagen

Gebruik command **ansible-playbook *backup\_file\_naam\_yaml*** om de playbook te runnen

Gebruik command **cat** om na te kijken of backup file effectief werd gecreëerd

# HOW TO: Ansible gebruiken om een toestel te configureren

Gebruik command cat hosts om de hosts inventory file nog eens te analyseren

Creëer een nieuwe playbook door een nieuwe file aan te maken in de *ansible-csr1000v* directory (VS Code)

Voeg volgende lijnen toe in de file:

---

– name: CONFIGURE IPv6 ADDRESSING

  hosts: CSR1kv

  gather\_facts: false

  connection: local

  tasks:

   – name: SET IPv6 ADDRESS

     ios\_config:

       parents: “interface GigabitEthernet1″

       lines:

         – description IPv6 ADDRESS

         – ipv6 address 2001:db8:acad:1::1/64

         – ipv6 address fe80::1:1 link-local

   – name: SHOW IPv6 INTERFACE BRIEF

     ios\_command:

       commands:

         – show ipv6 interface brief

     register: output

   – name: SAVE OUTPUT ./ios\_configurations/

     copy:

       content: “{{ output.stdout[0] }}”

       dest: “ios\_configurations/IPv6\_output\_{{ inventory\_hostname }}.txt”

*Let op de YAML indentation!*

Gebruik command **ping *192.168.56.101*** om de *CSR1000v virtuele machine* te pingen

Gebruik command **mkdir *ios\_configurations*** om de configurations directory aan te maken waar de output zal worden opgeslagen

Gebruik command **ansible-playbook *file\_naam\_yaml*** om de playbook te runnen

Gebruik command **cat** om na te kijken of de file effectief werd gecreëerd