|  |
| --- |
| **DEVOPS PORTFOLIO: HANDS-ON LABS & PROJECTEN** |

*datum van uitvoering + titel/nummer van het lab/ project + te onthouden (specifiek wat zijn de specifieke moeilijkheden?)*

Table of Contents

[3.1.12 Lab – Explore Python development tools 4](#_Toc93993281)

[Te onthouden: 4](#_Toc93993282)

[3.3.11 Lab – Software version control with Git 4](#_Toc93993283)

[Te onthouden: 4](#_Toc93993284)

[3.4.6 Lab – Explore Python classes 9](#_Toc93993285)

[Te onthouden : 9](#_Toc93993286)

[3.5.7 Lab – Create a Python unit test 10](#_Toc93993287)

[Te onthouden: 10](#_Toc93993288)

[3.6.6 Lab – Parse different data types with Python 11](#_Toc93993289)

[Te onthouden: 11](#_Toc93993290)

[4.5.5 Lab – Explore REST APIs with API simulator and Postman 11](#_Toc93993291)

[Te onthouden: 11](#_Toc93993292)

[4.9.2 Lab – Integrating a REST API with Python 11](#_Toc93993293)

[Te onthouden: 11](#_Toc93993294)

[5.4.6 Packet Tracer – Explore a simple network 12](#_Toc93993295)

[5.5.7 Packet Tracer – Explore network protocols 12](#_Toc93993296)

[5.6.6 Packet Tracer – Troubleshoot common network problems 13](#_Toc93993297)

[5.6.7 Lab – Network troubleshooting tools 13](#_Toc93993298)

[6.2.7 Lab – Build a sample web app in Docker Container 13](#_Toc93993299)

[Te onthouden: 13](#_Toc93993300)

[6.3.6 Lab – Build a CI/CD Pipeline using Jenkins 15](#_Toc93993301)

[Te onthouden: 15](#_Toc93993302)

[6.5.10 Lab – Explore the evolution of password methods 15](#_Toc93993303)

[Te onthouden: 15](#_Toc93993304)

[7.4.7 Lab – Use Ansible to backup and configure a device 15](#_Toc93993305)

[Te onthouden: 15](#_Toc93993306)

[7.4.8 Lab – Use Ansible to automate installing a web server 15](#_Toc93993307)

[Te onthouden: 15](#_Toc93993308)

[7.6.3 Lab – Automated testing using pyATS and Genie 15](#_Toc93993309)

[Te onthouden: 16](#_Toc93993310)

[8.3.5 Lab – Explore YANG models 16](#_Toc93993311)

[Te onthouden: 16](#_Toc93993312)

[8.3.6 Lab – Use NETCONF to access an IOS XE device 16](#_Toc93993313)

[Te onthouden: 16](#_Toc93993314)

[8.3.6 Lab – Use RESTCONF to access an IOS XE device 16](#_Toc93993315)

[Te onthouden: 16](#_Toc93993316)

[8.6.7 Lab – Construct a Python script to manage Webex Teams 16](#_Toc93993317)

[Te onthouden: 16](#_Toc93993318)

# 3.1.12 Lab – Explore Python development tools

Datum: 15/10/2021

## Te onthouden:

In deze lab heb ik geleerd hoe ik python instaleer, PIP en python virtual enviroments

Text

Description automatically generated

Voor de installatie van Python gebruik je best **venv** i.p.v. **pip3 install** (er kunnen bij pip3 install problemen ontstaan met dependencies). De juiste command om te gebruiken is dus **python3 -m venv devfun** (devfun is de gekozen naam van de enviroment)

# 3.3.11 Lab – Software version control with Git

Datum: 12/11/2021

## Te onthouden:

In deze lab leerde ik over de basics van Git, de initialisering en hoe een lokale repo integreren met de cloud-based repo

Text

Description automatically generated

Zoals men hier kan zien heb ik een eerste repo aangemaakt (lokaal).

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

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

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

Text

Description automatically generatedText

Description automatically generatedText

Description automatically generatedText

Description automatically generatedText

Description automatically generatedText

Description automatically generatedText

Description automatically generatedText

Description automatically generatedText

Description automatically generatedA screenshot of a computer

Description automatically generated

Met git diff kan je twee verschillende commits vergelijken. Gebruik daarvoor git diff gevolgd met de ID van de oudste en de ID van de meest recente commit.

Gebruik de **sed** command om een woord bv. te vervangen. Hier vervang ik class door exam

sed -i 's/class/exam/' devasc.txt

De **vim** command kan je gebruiken om een file aan te passen

# 3.4.6 Lab – Explore Python classes

Datum : 26/11/2021

## Te onthouden :

In deze lab heb ik Python onderdelen gezien/herzien als methoden, functies en klassen.

Een functie kan je individueel oproepen en is dus niet afhankelijk van iets anders. Een methode daarentegen is afhankelijk van het object waarin deze is gedefiniëerd.

Text

Description automatically generated

Text

Description automatically generatedText

Description automatically generated

# 3.5.7 Lab – Create a Python unit test

Datum: 02/12/2021

## Te onthouden:

Ter herhaling: unit tests testen een onafhankelijke unit code, zoals bijvoorbeeld functies, klassen, libraries… Je kan ze manueel runnen, maar gewoonlijk zullen deze automated zijn.

In deze lab zien we de unittest framework en hoe een unittest gebruiken om een functie te testen.

Vanuit de unittest framework: om een individuele unit van testing te creëren gebruik je de klasse **TestCase**

Methoden waarvan de naam start met de letters **test\_** tonen aan dat deze tests zijn een geen gewone methoden

Command **python3 -m unittest -h** 🡪 om alle command line options op te lijsten

Ik kreeg soms problemen bij deze lab door syntax errors. In het vervolg zal ik hier dus extra aandacht aan moeten steken.

# 3.6.6 Lab – Parse different data types with Python

Datum: 17/12/2021

## Te onthouden:

Parsen betekent het analyseren van een message, het in verschillende component parts delen en het begrijpen van elke part. Messages zijn eigenlijk strings. Om de message te kunnen begrijpen is het dus nodig om deze te kunnen omzetten in de juiste data-structure afhangend van de type in de context (dus bv. Integers, floats, strings, booleans). In deze lab zien we het parsen van data in XML, JSON en YAML.

XML parsen kan complex zijn

JSON parsen wordt zeer frequent gedaan (met REST APIs voornamelijk). Meestal doe je het met volgende stappen:

1. Athenticeer met een gebruiker/paswoord combinatie om een token op te halen
2. Voer een GET request uit naar de REST API 🡪 JSON als output formaat
3. Wijzig de JSON result indien nodig
4. Voer een POST of PUT request uit naar dezelfde REST API om de status van de bron te wijzigen, waarbij opnieuw JSON de output formaat is geïnterpreteerd zoals nodig om te bepalen of de operatie succesvol was

In Python scripts kan de Python json library gebruikt worden om JSON te parsen in Python. De yaml library converteert de data naar YAML.

# 4.5.5 Lab – Explore REST APIs with API simulator and Postman

Datum: 27/12/2021

## Te onthouden:

In deze lab leered ik hoe ik de School library API simulator gebruik om API calls uit te voeren en hoe ik ditzelfde doe met Postman. (Postman kende ik echter al). Ik leerde ook hoe ik calls maak met behulp van Python.

Deze lab was zeer eenvoudig, ik ondervond geen moeilijkheden. Waarschijnlijk omdat ik Postman al meerdere keren heb gebruikt in het verleden.

Als je API requests wilt uitvoeren met Python moet je gebruik maken van de Python *requests library*. *Faker* is de Python library die fake data zal genereren (faker library). Gebruik command **from faker import Faker**

# 4.9.2 Lab – Integrating a REST API with Python

Datum: 03/01/2022

## Te onthouden:

In deze lab leerde ik een app creëren in VS Code die JSON data neemt van de MapQuest Directions API, de data parsen en de output formatteren.

Text

Description automatically generatedText

Description automatically generated

Om het programma te beëindigen gebruik **Ctrl+C**

De moeilijkheden waren vooral het vermijden van syntax-fouten, de verschillende statements juist schrijven etc.

# 5.4.6 Packet Tracer – Explore a simple network

# 5.5.7 Packet Tracer – Explore network protocols

# 5.6.6 Packet Tracer – Troubleshoot common network problems

# 5.6.7 Lab – Network troubleshooting tools

# 6.2.7 Lab – Build a sample web app in Docker Container

Datum: 11/01/2022

## Te onthouden:

In deze lab leerde ik basic bash scripting technieken, het automatiseren van het creëren van een Dockerfile, Docker container opbouwen en het runnen + investigeren ervan.

! een Docker container is een manier van het encapsuleren van alles dat je nodig hebt om jouw app te kunnen runnen zodat je deze eenvoudig kan deployen in verschillende enviroments (dus niet enkel de devasc vm!!)

Text

Description automatically generatedText

Description automatically generated

Gebruik command **touch** om een nieuwe file toe te voegen via terminal.

Gebruik command **nano** om een file in text editor te openen.

Flask is een webapp framework geschreven in Python

Docker container creëerd zijn eigen IP adres van een private network address space

Om na te kijken of de web app runt zoals het moet, schrijf <http://localhost:8080> in de web browsers zelf. Je kan dit ook via curl doen

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

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

# 6.3.6 Lab – Build a CI/CD Pipeline using Jenkins

Datum: 13/01/2022

## Te onthouden:

In deze lab leerde ik de sample app code committen naar een Github repo, de code lokaal aanpassen en vervolgens de aanpassingen committen. Vervolgens moest ik een Docker container installeren die de laatste versie van Jenkins heeft. De bedoeling was om Jenkins te configureren en te gebruiken om de app na te kijken. Ten slotte moest ik de Sample app en test integreren in een CI/CD pipeline

# 6.5.10 Lab – Explore the evolution of password methods

Datum:

## Te onthouden:

# 7.4.7 Lab – Use Ansible to backup and configure a device

Datum:

## Te onthouden:

# 7.4.8 Lab – Use Ansible to automate installing a web server

Datum:

## Te onthouden:

# 7.6.3 Lab – Automated testing using pyATS and Genie

Datum:

## Te onthouden:

# 8.3.5 Lab – Explore YANG models

Datum:

## Te onthouden:

# 8.3.6 Lab – Use NETCONF to access an IOS XE device

Datum:

## Te onthouden:

# 8.3.6 Lab – Use RESTCONF to access an IOS XE device

Datum:

## Te onthouden:

# 8.6.7 Lab – Construct a Python script to manage Webex Teams

Datum:

## Te onthouden: