



Placement Empowerment Program
Cloud Computing and DevOps Centre

60 Days of DevOps Challenge

Day 5: Python Basics for DevOps

Name: MONISHA J R

Department: CSE



This **Python DevOps POC (Proof of Concept)** showcases how Python can be effectively used

for automation, scripting, monitoring, and system administration in DevOps.

Challenge 1: Create a Python program that accepts a user's name as input and prints a greeting message.

```
GNU nano 2.8.2 monisha@linux-mint: ~/dev_day05
# /usr/bin/env python3
name = input("Enter your name: ")
print(f"Hello, {name}! Welcome to the DevOps world! 🌟")
```

```
monisha@linux-mint:~$ mkdir dev_day05
monisha@linux-mint:~/dev_day05$ nano greet.py
monisha@linux-mint:~/dev_day05$ chmod +x greet.py
monisha@linux-mint:~/dev_day05$ ./greet.py
Enter your name: Monisha
Hello, Monisha! Welcome to the DevOps world! 🌟
monisha@linux-mint:~/dev_day05$
```

Challenge 2: Write a script that reads a text file and counts the number of words in it.

```
monisha@linux-mint:~/dev_day05$ nano word_count.py
monisha@linux-mint:~/dev_day05$ chmod +x word_count.py
monisha@linux-mint:~/dev_day05$ ./word_count.py
Enter the file name: greet.py
🌟 The file 'greet.py' contains 10 words.
monisha@linux-mint:~/dev_day05$
```

```
GNU nano 7.2 monisha@linux-mint: ~/dev_day05
# /usr/bin/env python3
word_count.py
file_name = input("Enter the file name: ")
try:
    with open(file_name, "r") as file:
        content = file.read()
        word_count = len(content.split())
        print(f"🌟 The file '{file_name}' contains {word_count} words.")
except FileNotFoundError:
    print(f"🔴 Error: The file '{file_name}' was not found.")
except Exception as e:
    print(f"🔴 An error occurred: {e}")
```

Challenge 3: Create a Python script that generates a random password of 12 characters.

```
monisha@linux-mint:~/dev_day05$ nano generate_password.py
monisha@linux-mint:~/dev_day05$ chmod +x generate_password.py
monisha@linux-mint:~/dev_day05$ ./generate_password.py
🌟 Your secure password: B3wKzP8j-0
monisha@linux-mint:~/dev_day05$
```

```
GNU nano 7.2 monisha@linux-mint: ~/dev_day05
#!/usr/bin/env python3
generate_password.py

import secrets
import string

def generate_password(length=12):
    characters = string.ascii_letters + string.digits + string.punctuation
    return ''.join(secrets.choice(characters) for _ in range(length))

print(f"🔒 Your secure password: {generate_password()}")
```

Challenge 4: Implement a Python program that checks if a number is prime.

```
monisha@linux-mint:~/dev_day05$ nano check_prime.py
monisha@linux-mint:~/dev_day05$ chmod +x check_prime.py
monisha@linux-mint:~/dev_day05$ ./check_prime.py
Enter a number: 73864
❌ 73864 is not a prime number.
monisha@linux-mint:~/dev_day05$ ./check_prime.py
Enter a number: 2
✅ 2 is a prime number.
monisha@linux-mint:~/dev_day05$
```

```
GNU nano 7.2 monisha@linux-mint: ~/dev_day05
#!/usr/bin/env python3
check_prime.py *

def is_prime(n):
    if n < 2:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True

try:
    num = int(input("Enter a number: "))
    if is_prime(num):
        print(f"✅ {num} is a prime number.")
    else:
        print(f"❌ {num} is not a prime number.")
except ValueError:
    print("❌ Please enter a valid integer.")
```

Challenge 5: Write a script that reads a list of server names from a file and pings each one.

```
monisha@linux-mint:~/dev_day05$ nano ping_servers.py
monisha@linux-mint:~/dev_day05$ chmod +x ping_servers.py
monisha@linux-mint:~/dev_day05$ ./ping_servers.py
❌ Error: servers.txt not found!
monisha@linux-mint:~/dev_day05$ echo "google.com" > servers.txt
monisha@linux-mint:~/dev_day05$ ./ping_servers.py
google.com: ✅ Reachable
monisha@linux-mint:~/dev_day05$ echo "invalid.server" >> servers.txt
monisha@linux-mint:~/dev_day05$ ./ping_servers.py
google.com: ✅ Reachable
invalid.server: ❌ Unreachable
monisha@linux-mint:~/dev_day05$
```

```
monisha@linux-mint: ~/dev_day05
GNU nano 7.2
#!/usr/bin/env python3
ping_servers.py

import os
import platform

def ping_server(server):
    param = "-n 1" if platform.system().lower() == "windows" else "-c 1"
    response = os.system(f"ping {param} {server} > /dev/null 2>&1")
    return response == 0

try:
    with open("servers.txt", "r") as file:
        servers = [line.strip() for line in file.readlines()]
        for server in servers:
            if server:
                status = "✅ Reachable" if ping_server(server) else "❌ Unreachable"
                print(f"{server}: {status}")
except FileNotFoundError:
    print("❌ Error: servers.txt not found!")
```

Challenge 6: Use the requests module to fetch and display data from a public API (e.g., JSONPlaceholder).

```
monisha@linux-mint:~/dev_day05$ nano crud_call.py
monisha@linux-mint:~/dev_day05$ chmod +x crud_call.py
monisha@linux-mint:~/dev_day05$ ./crud_call.py
Status Code: 200
Post: {'user_id': 1, 'id': 1, 'title': 'sunt aut facere repellat provident occaecati excepturi optio reprehenderit', 'body': 'quia et suscipit\nsuscipit rec-  
usandae consequuntur expedita et cum\nreprehenderit molestiae ut ut quas totam\nnostrum rerum est autem sunt rem eveniet architecto'}
```

```
monisha@linux-mint: ~/dev_day05
GNU nano 7.2
#!/usr/bin/env python3
crud_call.py

import requests

API_URL = "https://jsonplaceholder.typicode.com/posts"

def get_post(post_id):
    response = requests.get(f"{API_URL}/{post_id}")
    print(f"Status Code: {response.status_code}")
    print(f"Post: {response.json()}")

get_post(1)
```

Challenge 7: Automate a simple task using Python (e.g., renaming multiple files in a directory).

```
monisha@linux-mint:~/dev_day05$ nano rename_files.py
monisha@linux-mint:~/dev_day05$ chmod +x rename_files.py
monisha@linux-mint:~/dev_day05$ mkdir test_files
monisha@linux-mint:~/dev_day05$ touch test_files/file1.txt test_files/file2.txt
monisha@linux-mint:~/dev_day05$ ./rename_files.py
Renamed: file1.txt - renamed_1.txt
Renamed: file2.txt - renamed_2.txt
monisha@linux-mint:~/dev_day05$
```

```
monisha@linux-mint: ~/dev_day05
GNU nano 7.2
#!/usr/bin/env python3
rename_files.py

import os

directory = "test_files"
prefix = "renamed_"

if not os.path.exists(directory):
    print(f"❌ Error: Directory '{directory}' not found!")
    exit(1)

for count, filename in enumerate(os.listdir(directory), start=1):
    old_path = os.path.join(directory, filename)
    if os.path.isfile(old_path):
        new_name = f"{prefix}{count}.txt"
        new_path = os.path.join(directory, new_name)
        os.rename(old_path, new_path)
        print(f"✅ Renamed: {filename} -> {new_name}")
```

Challenge 8 : Create a Python script that monitors CPU and memory usage every 5 seconds.

```
monisha@linux-mint: ~/dev_day05$ nano monitor_resources.py
monisha@linux-mint: ~/dev_day05$ chmod +x monitor_resources.py
monisha@linux-mint: ~/dev_day05$ ./monitor_resources.py
Monitoring CPU and Memory usage... (Press Ctrl+C to stop)
CPU Usage: 0.9% | Memory Usage: 20.9%
CPU Usage: 0.7% | Memory Usage: 21.0%
CPU Usage: 0.5% | Memory Usage: 20.9%
CPU Usage: 0.8% | Memory Usage: 20.9%
CPU Usage: 0.3% | Memory Usage: 20.9%
Monitoring stopped.
monisha@linux-mint: ~/dev_day05$
```

```
GNU nano 7.2
monisha@linux-mint: ~/dev_day05$ nano monitor_resources.py
#!/usr/bin/env python3
import psutil
import time

def monitor_resources(interval=5):
    print("Monitoring CPU and Memory usage... (Press Ctrl+C to stop)\n")
    try:
        while True:
            cpu = psutil.cpu_percent(interval=1)
            memory = psutil.virtual_memory().percent
            print(f"CPU Usage: {cpu}% | Memory Usage: {memory}%")
            time.sleep(interval - 1)
    except KeyboardInterrupt:
        print("\nMonitoring stopped.")

monitor_resources()
```

Challenge 9 : Write a Python program that creates a user in Linux using subprocess and verifies the creation.

```
monisha@linux-mint: ~/dev_day05$ nano create_user.py
monisha@linux-mint: ~/dev_day05$ chmod +x create_user.py
monisha@linux-mint: ~/dev_day05$ sudo ./create_user.py
[sudo] password for monisha:
Enter the username to create: mooni
id: 'mooni': no such user
User 'mooni' created successfully.
uid=1001(mooni) gid=1001(mooni) groups=1001(mooni)
Verification successful: 'mooni' exists.
monisha@linux-mint: ~/dev_day05$
```

```
GNU nano 7.2
monisha@linux-mint: ~/dev_day05$ nano create_user.py
#!/usr/bin/env python3
import subprocess

def create_user(username):
    try:
        subprocess.run(["id", username], check=True, stdout=subprocess.DEVNULL)
        print(f"User '{username}' already exists.")
        return
    except subprocess.CalledProcessError:
        pass

    try:
        subprocess.run(["sudo", "useradd", "-m", "-s", "/bin/bash", username], check=True)
        print(f"User '{username}' created successfully.")
    except subprocess.CalledProcessError:
        print(f"Failed to create user '{username}'. Ensure you have sudo privileges.")

    try:
        subprocess.run(["id", username], check=True)
        print(f"Verification successful: '{username}' exists.")
    except subprocess.CalledProcessError:
        print(f"Verification failed: '{username}' does not exist.")

username = input("Enter the username to create: ")
create_user(username)
```