DEVOPS SHACK

10 Real-Time Python Automation Scripts

Enrol To batch-7 | DevSecOps & Cloud DevOps

Script-1 | Email Campaign

#Single Recipent

```
import smtplib
from email.mime.text import MIMEText

def send_email(subject, body, to_email):
    # Email settings
    sender_email = 'jaiswaladi246@gmail.com'
    sender_password = 'tpzk olai omuk vhol'  # Use App Password if you enabled 2-Step Verification

    # Create email message
msg = MIMEText(body)
msg['Subject'] = subject
msg['From'] = sender_email
    msg['To'] = to_email
    # Sending email
    with smtplib.SMTP_SSL('smtp.gmail.com', 465) as server:
        server.login(sender_email, sender_password)
        server.sendmail(sender_email, to_email, msg.as_string())
# Example usage:
send_email('Automation Test', 'This is an automated message.', 'jaiswaladi246@gmail.com')
```

#Multiple Recipent

```
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText from
email.mime.base import MIMEBase
from email import encoders import
os

def send_individual_emails(subject, body, recipients, pdf_path=None):
    # Email settings
    sender_email = 'jaiswaladi246@gmail.com'
    sender_password = 'tpzk olai omuk vhol'  # Use App Password if you enabled 2-Step Verification

    # Create a base HTML body template for beautification
html_body_template = """     <html>
    <body style="font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif; background-color:
#f4f4f4; padding: 20px; color: #333;">
        <div style="max-width: 600px; margin: 0 auto; background-color: white; padding: 20px;
borderradius: 10px; box-shadow: 0 0 10px rgba(0,0,0,0.1);">
            <h2 style="color: #4CAF50; text-align: center;">  Welcome to Batch-7!</h2>
            <p style="font-size: 18px; color: #555;">Hello <b>{name}</b>,</p>
            <p style="font-size: 16px; color: #555;">
                We are excited to announce that <b>Batch-7</b> of our <b>DevSecOps & Cloud DevOps
Bootcamp</b> is starting on <b>2nd November 2024</b>.
                <br><br>
                Secure your spot now and learn the most in-demand skills in DevOps and DevSecOps,
including:
            </p>
            <ul style="font-size: 16px; color: #333; line-height: 1.8;">
                <li>CI/CD Tools (Jenkins, GitHub Actions, GitLab CI/CD)</li>
                <li>Infrastructure as Code (Ansible, Terraform)</li>
                <li>Security Tools (Trivy, OWASP Dependency Check, Vault)</li>
                <li>Cloud Platforms (Azure DevOps, Kubernetes, Docker)</li>
                <li>Hands-on Projects (NodeJS, Python, Java, Microservices)</li>
                <li>Linux & Shell Scripting</li>
                <li>Resume Building & LinkedIn Profile Optimization</li>
            </ul>
            <p style="text-align: center; margin: 30px 0;">
                    <a href="https://devopsshack.com" style="text-decoration: none; background-color:
 #4CAF50; color: white; padding: 15px 25px; border-radius: 5px; font-size: 18px;">  Enroll Now</a>
            </p>
            <p style="font-size: 16px; color: #555; text-align: center;">Don't miss this opportunity to
become a DevOps expert!</p>
            <p style="font-size: 16px; color: #555;">Best Regards,<br><b>DevOps Shack Team</b></p>
        </div>
        <footer style="text-align: center; margin-top: 20px;">
```

```
        <p style="font-size: 14px; color: #888;">© 2024 DevOps Shack | <a
href="https://devopsshack.com" style="color: #4CAF50; text-decoration: none;">Visit our
website</a></p>
    </footer>
  </body>
  </html>
  """

  # Send email to each recipient individually
for recipient in recipients:      # Personalize
the message
    name = recipient.split('@')[0].capitalize()  # Get the name from the email (before @)
html_body = html_body_template.format(name=name)

    # Create the email message
    msg = MIMEMultipart()
    msg['Subject'] = subject
    msg['From'] = sender_emai
    msg['To'] = recipient
    # Attach the beautified HTML body
    msg.attach(MIMEText(html_body, 'html'))

    # Attach the PDF if provided and exists      if pdf_path
and os.path.exists(pdf_path):          with open(pdf_path,
'rb') as pdf_file:              pdf_part =
MIMEBase('application', 'octet-stream')
pdf_part.set_payload(pdf_file.read())
encoders.encode_base64(pdf_part)
            pdf_part.add_header('Content-Disposition', f'attachment;
filename={os.path.basename(pdf_path)}')
            msg.attach(pdf_part)

    # Send the email
    with smtplib.SMTP_SSL('smtp.gmail.com', 465) as server:
        server.login(sender_email, sender_password)
        server.sendmail(sender_email, recipient, msg.as_string())

# Example usage with multiple recipients
recipients_list = ['jaiswaladi246@gmail.com','recipient1@example.com', 'recipient2@example.com']
 # Add multiple emails here send_individual_emails(
  subject='  Enroll Now: Batch-7 Starting on 2nd November!',     body='Batch-7
is starting on 2nd November, enroll now to book your seat!',
  recipients=recipients_list,  # Provide a list of recipients
  pdf_path='Batch-7-Syllabus.pdf'  # Provide the correct file name with .pdf extension
)
```

Script-2 | WEB-SCRAPING

pip install requests pip

install beautifulsoup4

```
import requests
from bs4 import BeautifulSoup

def scrape_headlines_demo(url):
response = requests.get(url)
    soup = BeautifulSoup(response.text, 'html.parser')

    # Extract all headlines from the website (assuming they are in <h2> tags)
headlines = soup.find_all('h2')

    for idx, headline in enumerate(headlines, 1):
        print(f'{idx}: {headline.text.strip()}')

# Example usage with a real news site:
scrape_headlines_demo('https://www.indiatoday.in/india')
```

Script-3 | Automating File Handling

Example : Moving Files Based on File Type

This task involves moving files of a particular type (e.g., .txt files) from one folder to another.

```
import os
import shutil

def move_files_by_type(source_dir, dest_dir, file_extension):
    for filename in os.listdir(source_dir):
        if filename.endswith(file_extension):
            shutil.move(os.path.join(source_dir, filename), os.path.join(dest_dir, filename))
# Example usage:
move_files_by_type('/path/to/source', '/path/to/destination', '.txt')
```

This will move all .txt files from the source directory to the destination directory.

Script-4 | Scaling up of AKS Resources as Per Weather

```
az login --use-device-code

az aks get-credentials --resource-group DevOpsShack-RG-1 --name Python-test

az aks install-cli
C:\Users\jaisw\.azure-kubectl
$env:Path += ";C:\Users\jaisw\.azure-kubectl"
                                                    import requests
                                                    import subprocess

def get_weather_data(city):
    """Fetches weather data for a specific city using WeatherAPI."""     api_key =
'91da448afecb4b83af642416241310'  # Replace with your WeatherAPI key     base_url
= f'http://api.weatherapi.com/v1/current.json?key={api_key}&q={city}'

    response = requests.get(base_url)
data = response.json()

    if 'error' not in data:         weather =
data['current']['condition']['text']         temp_c =
data['current']['temp_c']
        print(f'Weather in {city}: {weather}, Temperature: {temp_c}°C')

        # If the weather condition is "Heavy rain", scale the number of pods in AKS         if "Heavy
rain" in weather:             print("Weather condition is heavy rain. Scaling the number of pods in
AKS...")             scale_aks_pods_using_kubectl(namespace='blogging-app',
deployment_name='bankapp', replicas=2)  # Set to 3 replicas         else:
            print("Weather condition is not heavy rain. No scaling action is required.")
else:
        print(f"Error fetching data for {city}: {data['error']['message']}")

def scale_aks_pods_using_kubectl(namespace, deployment_name, replicas):
    """Scales the number of pods for a specific deployment using kubectl."""
try:
        # Run the kubectl scale command to scale the pods
subprocess.run([
            'kubectl', 'scale', f'deployment/{deployment_name}', f'--replicas={replicas}', '-n', namespace
        ], check=True)         print(f"Scaled the deployment {deployment_name}
to {replicas} pods.")     except subprocess.CalledProcessError as e:
        print(f"Failed to scale pods: {e}")


# Example usage:
```

```
get_weather_data('London')
```

Script-5 | ChatBot Interactions

User Input: "I need help with my order"

Chatbot Output: "Sure! How can I assist you today?"

```
import nltk
from nltk.chat.util import Chat, reflections

# Define the chatbot responses
pairs = [
    [
        r"(.*)help(.*)",  # Matches any sentence containing the word "help"
        ["Sure! How can I assist you today?",]
    ],
    [
        r"(.*)price of (.*)",  # Matches any sentence containing "price of"
        ["The price of %2 is $50.",]
    ],
    [
        r"(.*)course(.*)",  # Matches any sentence containing the word "course"
        ["If you are looking for the DevSecOps & Cloud DevOps Course, then check out this link:
https://www.devopsshack.com/courses/Batch-7--Zero-To-Hero--DevSecOps--Cloud-DevOps-
BootCamp-6704e7c7f251e913d565a738",]
    ],
    [
        r"quit",  # Matches the exact word "quit"
        ["Goodbye! Have a nice day."]
    ]
]

def basic  chatbot():
    print("Welcome to Customer Support (type 'quit' to exit)")
    chat = Chat(pairs, reflections)
    chat.converse()

# Start the chatbot
basic_chatbot()
```

Script-6 | Monitor Website & Get Notified on Slack

Step 1: Create a Slack Webhook

You will need a Slack webhook to send notifications. Here's how to set it up:

1.1 Create a Slack App

1. Go to Slack's API page: https://api.slack.com/apps.
2. Click on **Create New App**.
3. Choose **From Scratch**.
4. Enter the **App Name** and select your workspace.
5. Click **Create App**.

2.2 Enable Incoming Webhooks

1. Once your app is created, in the left-hand sidebar, click **Incoming Webhooks**.
2. Toggle the switch to enable **Incoming Webhooks**.
3. Scroll down and click **Add New Webhook to Workspace**.
4. Select the channel where you want notifications to be sent, and then click **Allow**.
5. Slack will provide you a webhook URL in this format:

https://hooks.slack.com/services/XXXXX/YYYYY/ZZZZZZZ

6. Copy this URL, as it will be used in your Python script.

```
import requests def monitor_server_health(server_url,
slack_webhook):     try:
    # Send request to the server
    response = requests.get(server_url)
```

```
    # Send notification to Slack
    slack_data = {'text': message}
    slack_response = requests.post(slack_webhook, json=slack_data)
    # Check if the server is up or down based on status code
    if response.status_code == 200:
        message = f"Server {server_url} is UP!"
    else:
        message = f"Server {server_url} is DOWN! Status Code: {response.status_code}"
```

```
    if slack_response.status_code == 200:
        print(f"Slack notification sent: {message}")
    else:
```

```
        print(f"Failed to send Slack notification. Status Code: {slack_response.status_code}")

  except requests.exceptions.RequestException as e:
print(f"Error while checking server health: {e}")      # Optional:
Send failure notification to Slack       error_message = f"Error
while checking server health: {e}"
requests.post(slack_webhook, json={'text': error_message})

# Example usage: server_url =
'https://your-server-url.com'
slack_webhook = 'https://hooks.slack.com/services/YOUR/SLACK/WEBHOOK'
monitor_server_health(server_url, slack_webhook)
```

Script-7 | BACKUP IN ZIP

```
import shutil import
os
from datetime import datetime
import zipfile

def backup_and_zip_files(source_folder, backup_folder):
  # Ensure the backup folder exists
  if not os.path.exists(backup_folder):
    os.makedirs(backup_folder)

  # Create a timestamp for unique folder names
  timestamp = datetime.now().strftime('%Y-%m-%d_%H-%M-%S')
  backup_zip_filename = os.path.join(backup_folder, f"backup_{timestamp}.zip")

  # Create a zip file    with zipfile.ZipFile(backup_zip_filename, 'w',
zipfile.ZIP_DEFLATED) as zipf:
    # Walk through the source folder and add files to the zip file
for root, dirs, files in os.walk(source_folder):       for file in
files:            # Get full file path
          file_path = os.path.join(root, file)
# Add file to the zip file with relative path
          zipf.write(file_path, os.path.relpath(file_path, source_folder))

  print(f"Backup completed and saved as {backup_zip_filename}")

# Example usage: backup_and_zip_files(r'C:\Users\jaisw\OneDrive\Documents\Python-
Programs\TEXTS', r'C:\Users\jaisw\OneDrive\Documents\Python-Programs\BACKUP')
```

Script-8 | Cleanup Directory on Condition

```
import os
import time
                def cleanup_old_files(directory, days_old):
                    current_time = time.time()

    for filename in os.listdir(directory):
file_path = os.path.join(directory, filename)
if os.path.isfile(file_path):
            file_age = current_time - os.path.getmtime(file_path)
if file_age > days_old * 86400:  # Convert days to seconds
os.remove(file_path)
                print(f"Deleted: {filename}")

# Example usage:
cleanup_old_files(r'C:\Users\jaisw\Videos\Parameters', 30)  # Deletes files older than 30 days
```

Script-9 | Downloading YouTube Videos

```
import yt_dlp as youtube_dl
from feedparser import parse

def download_latest_videos(subscription_feed_url, output_folder):
    feed = parse(subscription_feed_url)

    for entry in feed.entries[:5]:  # Download the latest 5 videos
video_url = entry.link        try:
            ydl_opts = {
                'outtmpl': f'{output_folder}/%(title)s.%(ext)s',
                'format': 'best'
            }
            with youtube_dl.YoutubeDL(ydl_opts) as ydl:
                ydl.download([video_url])
print(f"Downloaded {video_url}")        except Exception as
e:                print(f"Failed to download {video_url}. Reason:
{e}")

# Example usage:
subscription_feed_url =
"https://www.youtube.com/feeds/videos.xml?channel_id=UCK96F7QJuTYgNM0FFT7DvJw"
output_folder = "BACKUP"
download_latest_videos(subscription_feed_url, output_folder)
```

Script-10 | Syncronizing Local Repo With Remote Repo

```
import os
```

```
def clone_or_pull_repo(repo_url, local_dir):    # Check if the local directory exists    if not os.path.exists(local_dir):        # If not, clone the repository        os.system(f"git clone {repo_url} {local_dir}")        print(f"Cloned {repo_url} into {local_dir}")    else:
        # If the directory exists, change to that directory and pull the latest changes
        os.chdir(local_dir)
os.system("git pull")
        print(f"Updated repository in {local_dir}")

# Example usage for multiple repositories: repositories = [
    {"url": "https://github.com/jaiswaladi246/FullStack-Blogging-App.git", "dir": r"C:\Users\jaisw\OneDrive\Documents\Python-Programs\GITTEST\FullStack-Blogging-App"},
    {"url": "https://github.com/jaiswaladi246/EKS-Terraform.git", "dir": r"C:\Users\jaisw\OneDrive\Documents\Python-Programs\GITTEST\EKS-Terraform"},
    {"url": "https://github.com/jaiswaladi246/Blue-Green-Deployment.git", "dir": r"C:\Users\jaisw\OneDrive\Documents\Python-Programs\GITTEST\Blue-GreenDeployment"},
]

for repo in repositories:
    clone_or_pull_repo(repo["url"], repo["dir"])
```

Task-10 | Synchronization of Repos in Local

```
import os

def clone_or_pull_repo(repo_url, local_dir):    # Check if the local directory exists    if not os.path.exists(local_dir):        # If not, clone the
```

```
repository        os.system(f"git clone {repo_url}
{local_dir}")        print(f"Cloned {repo_url} into
{local_dir}")     else:
       # If the directory exists, change to that directory and pull the latest changes
os.chdir(local_dir)        os.system("git pull")
       print(f"Updated repository in {local_dir}")

# Example usage for multiple repositories: repositories
= [
   {"url": "https://github.com/jaiswaladi246/FullStack-Blogging-App.git", "dir":
r"C:\Users\jaisw\OneDrive\Documents\Python-Programs\GITTEST\FullStack-Blogging-App"},
{"url": "https://github.com/jaiswaladi246/EKS-Terraform.git", "dir":
r"C:\Users\jaisw\OneDrive\Documents\Python-Programs\GITTEST\EKS-Terraform"},
{"url": "https://github.com/jaiswaladi246/Blue-Green-Deployment.git", "dir":
r"C:\Users\jaisw\OneDrive\Documents\Python-Programs\GITTEST\Blue-
GreenDeployment"},
]

for repo in repositories:
   clone_or_pull_repo(repo["url"], repo["dir"])
```