Demo Project: Website Monitoring and Recovery

Project Description

In this project, we will create a **Website Monitoring and Recovery** system using **Python**. The system will:

- Monitor the health of a website by making HTTP requests.
- Send email notifications if the website is down.
- Automatically restart the application if it is not responding.
- Reboot the entire server if necessary.

We will use **Linode** cloud platform to create the server, install Docker, and deploy a simple **Nginx** container as the website. The monitoring and recovery functionality will be implemented with **Python** libraries.

Step 1: Create a Server on Linode

- On Linode, click on **Create Linode**.
- Choose a Distribution: **Image: Debian 11**.
- **Region**: Select the region closest to you.
- **Linode Plan**:

    - Shared CPU: **Linode 2 GB**.
- **Root Password**: Create a password.
- **SSH Key**: Create the SSH key so we can SSH into the server.
    - Label: **python-monitoring**.
    - Public key: Found in your terminal with `cat ~/.ssh/id_rsa.pub`. Copy and paste this key.
- Click **Create Linode**.

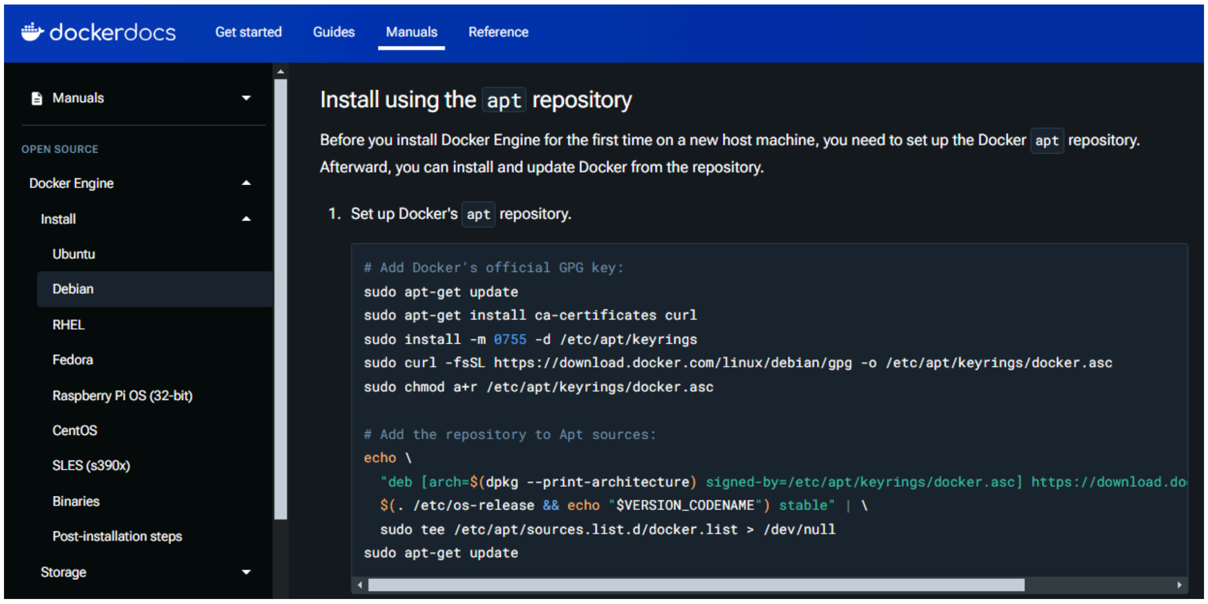
Connect to server using public IP

- `ssh root@<public ip>`

Step 2: Install Docker

Confirm you have Debian installed: `cat /etc/os-release`

Install Docker using the official instructions: https://docs.docker.com/engine/install/debian/



Install using the `apt` repository

Before you install Docker Engine for the first time on a new host machine, you need to set up the Docker `apt` repository. Afterward, you can install and update Docker from the repository.

1. Set up Docker's `apt` repository.

```
# Add Docker's official GPG key:
sudo apt-get update
sudo apt-get install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/debian/gpg -o /etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.do
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
```

```
# Add Docker's official GPG key:
apt-get update
apt-get install ca-certificates curl
install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/debian/gpg -o /etc/apt/keyrings/do
chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.as
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
  tee /etc/apt/sources.list.d/docker.list > /dev/null
apt-get update

apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docke
```

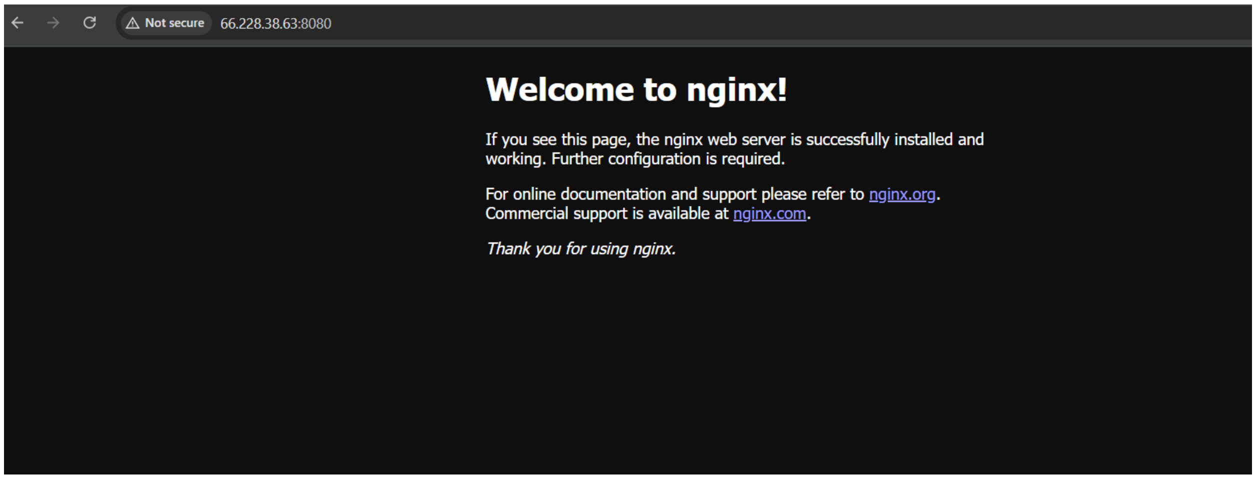
Note: `sudo` is not needed since we are conecccted as Root user.

Step 3: Run a Nginx Docker container on the remote server

Run Nginx Container: `docker run -d -p 8080:80 nginx`

Verify the container is running: `docker ps`

Access the website in your browser: `<linode public ip>:8080`



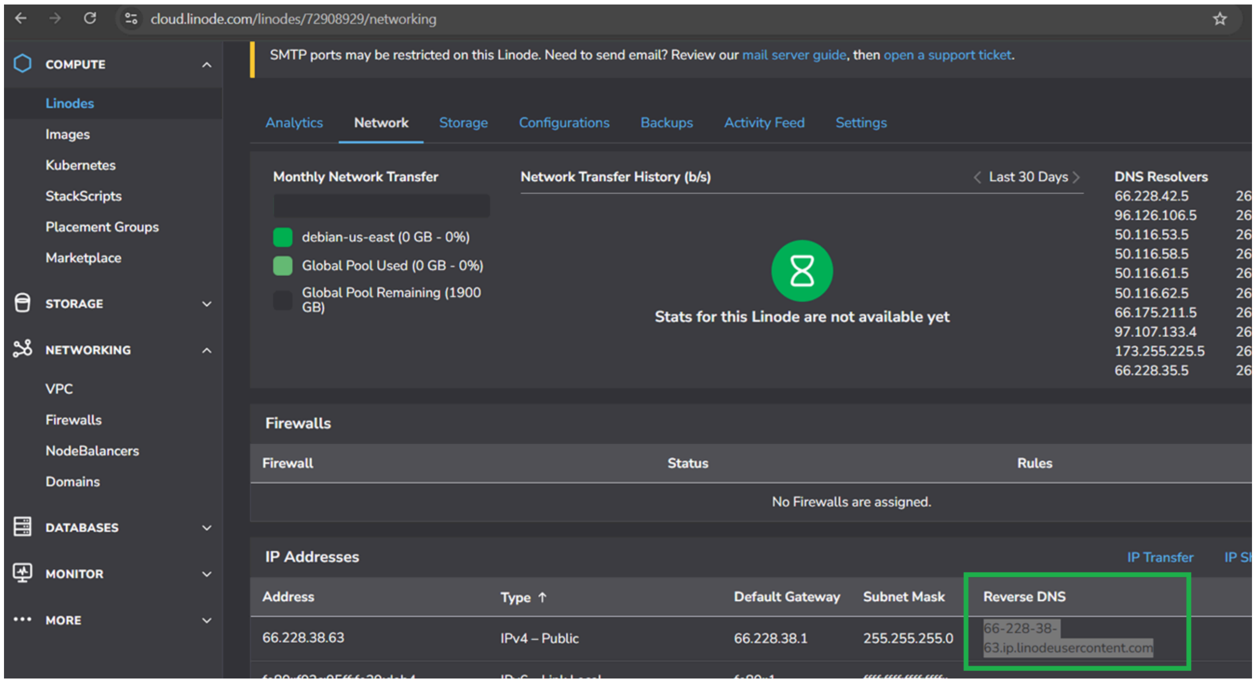
Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

- We can also use the `<Linode DNS>:8080` to access:



SMTP ports may be restricted on this Linode. Need to send email? Review our mail server guide, then open a support ticket.

Monthly Network Transfer

- 🟩 debian-us-east (0 GB - 0%)
- 🟩 Global Pool Used (0 GB - 0%)
- ■ Global Pool Remaining (1900 GB)

Network Transfer History (b/s) — Last 30 Days

Stats for this Linode are not available yet

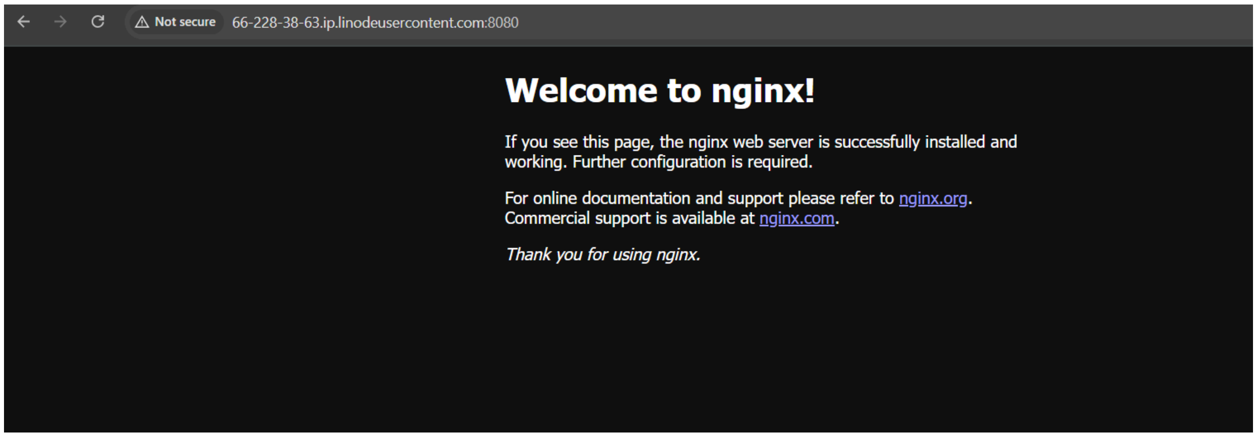
DNS Resolvers
66.228.42.5
96.126.106.5
50.116.53.5
50.116.58.5
50.116.61.5
50.116.62.5
66.175.211.5
97.107.133.4
173.255.225.5
66.228.35.5

Firewalls

Firewall	Status	Rules
No Firewalls are assigned.		

IP Addresses

Address	Type ↑	Default Gateway	Subnet Mask	Reverse DNS
66.228.38.63	IPv4 – Public	66.228.38.1	255.255.255.0	66-228-38-63.ip.linodeusercontent.com



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Step 4: Install Python Packages

1. Install the required Python packages:

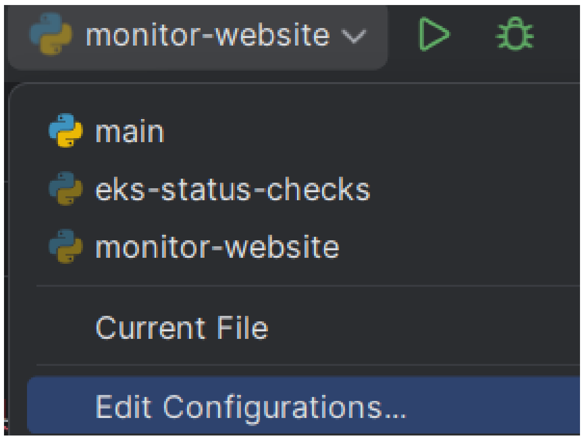
```
pip install requests
pip install paramiko
pip install linode_api4
pip install schedule
```

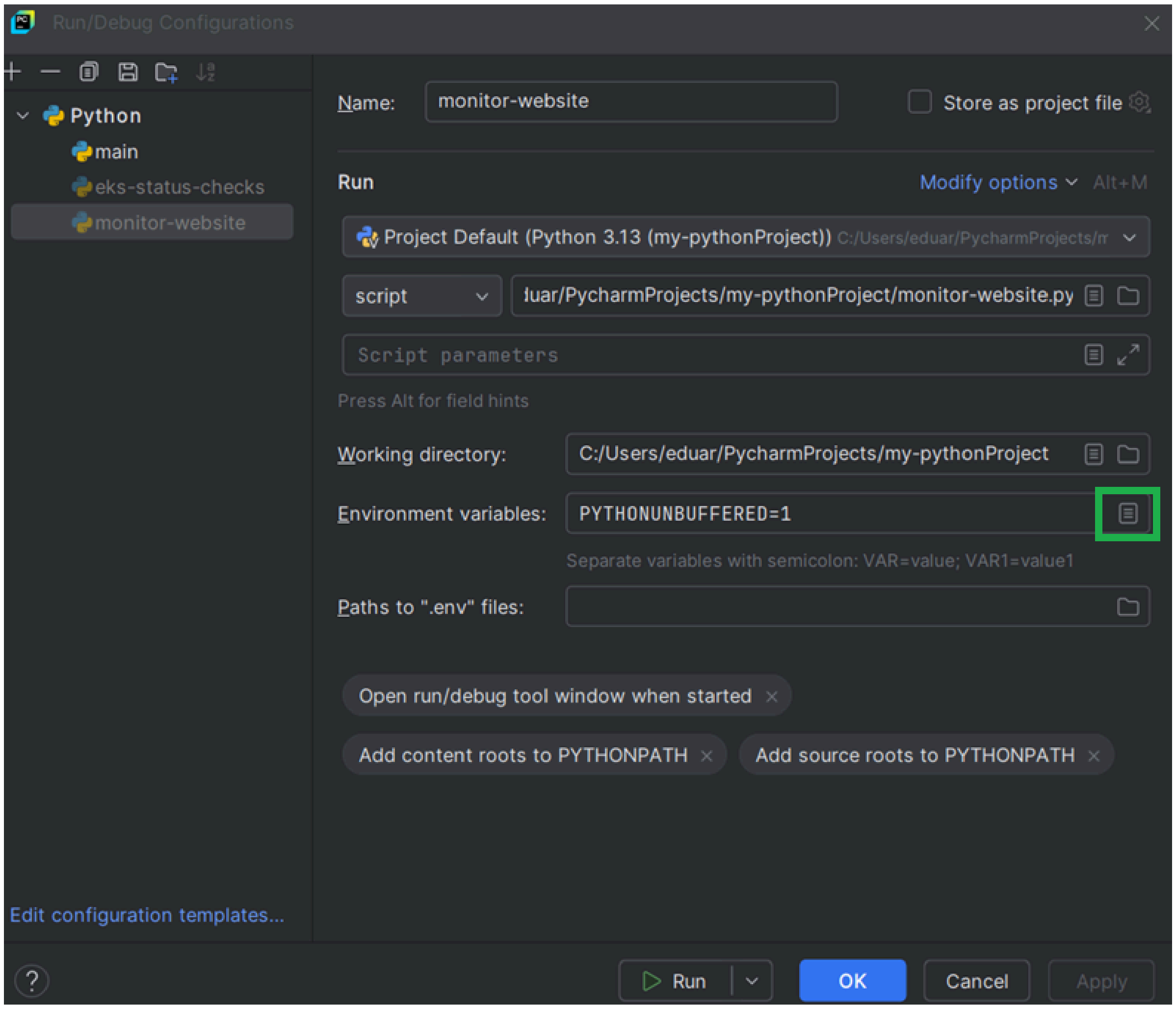
2. Verify installation in **PyCharm** under **External Libraries → Python 3.x → site-packages**.

Step 5: Set Environment Variables in PyCharm

1. Go to **Run → Edit Configurations...**.
2. Select your script.
3. Add Environment Variables:
   - **EMAIL_ADDRESS**: Your Gmail address.
   - **EMAIL_PASSWORD**: Your Gmail app password.

- **LINODE_TOKEN**: Your Linode API token.




Run/Debug Configurations
Python
main
eks-status-checks
monitor-website
Store as project file
Run
Modify options
Alt+M
Edit configuration templates...
Run
OK
Cancel
Apply
Open run/debug tool window when started
Add content roots to PYTHONPATH
Add source roots to PYTHONPATH

Name: monitor-website

Project Default (Python 3.13 (my-pythonProject)) C:/Users/eduar/PycharmProjects/m

script: luar/PycharmProjects/my-pythonProject/monitor-website.py

Script parameters

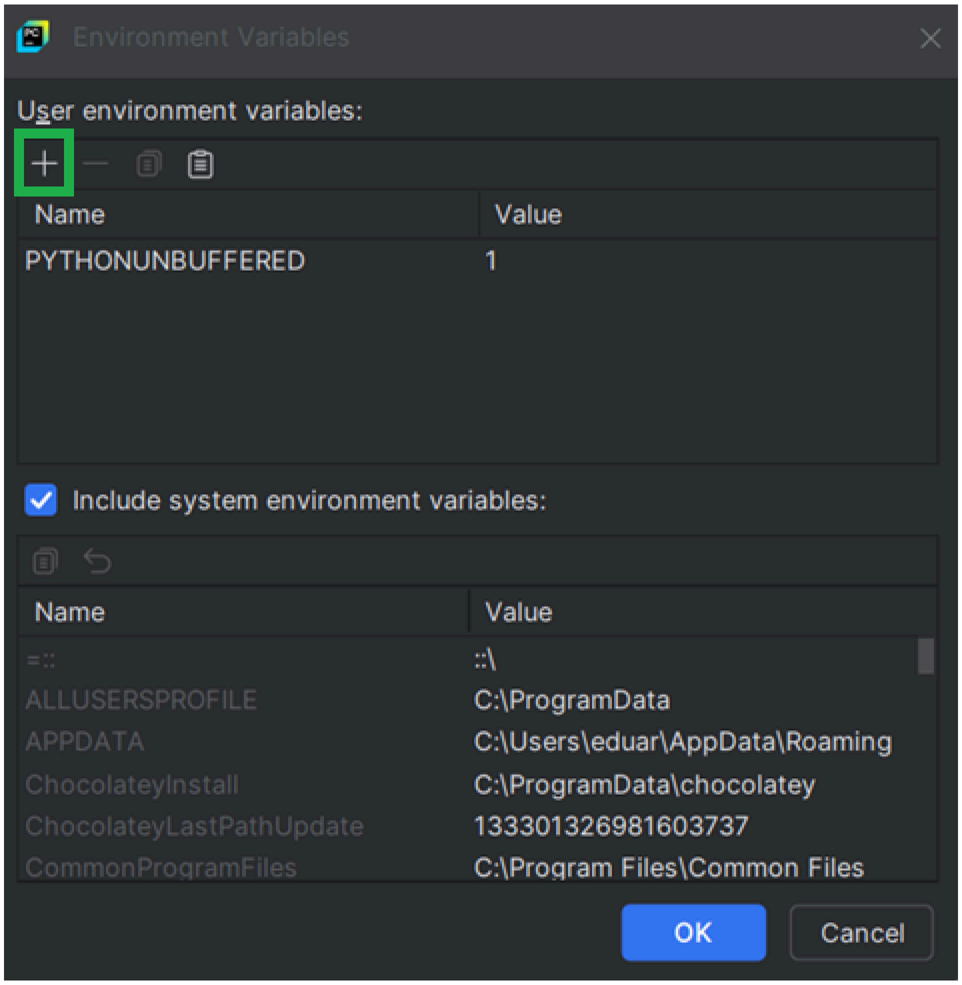
Press Alt for field hints

Working directory: C:/Users/eduar/PycharmProjects/my-pythonProject

Environment variables: PYTHONUNBUFFERED=1

Separate variables with semicolon: VAR=value; VAR1=value1

Paths to ".env" files:



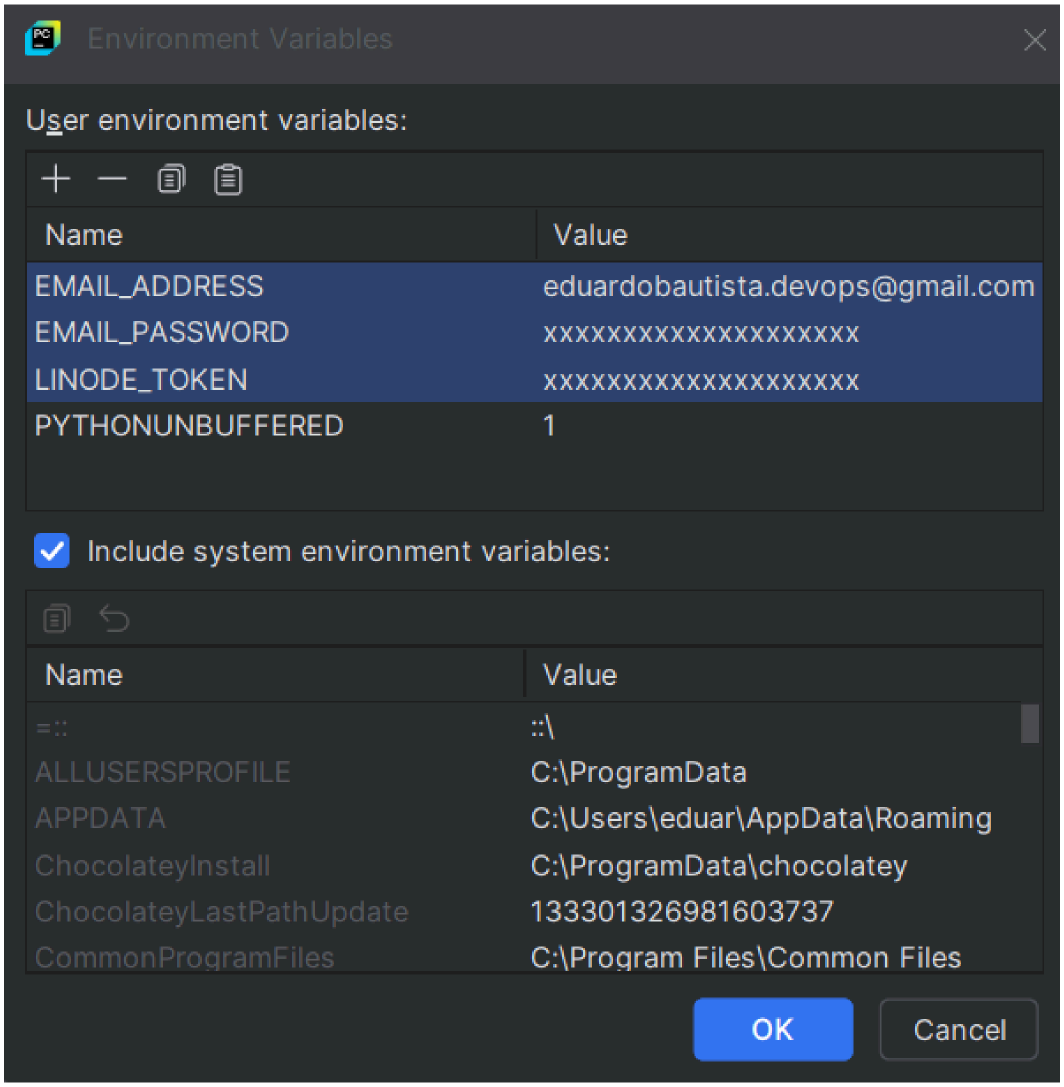
Environment Variables

User environment variables:

Name	Value
PYTHONUNBUFFERED	1

☑ Include system environment variables:

Name	Value
=::	::\
ALLUSERSPROFILE	C:\ProgramData
APPDATA	C:\Users\eduar\AppData\Roaming
ChocolateyInstall	C:\ProgramData\chocolatey
ChocolateyLastPathUpdate	133301326981603737
CommonProgramFiles	C:\Program Files\Common Files



User environment variables:

Name	Value
EMAIL_ADDRESS	eduardobautista.devops@gmail.com
EMAIL_PASSWORD	xxxxxxxxxxxxxxxxxxxx
LINODE_TOKEN	xxxxxxxxxxxxxxxxxxxx
PYTHONUNBUFFERED	1

Include system environment variables:

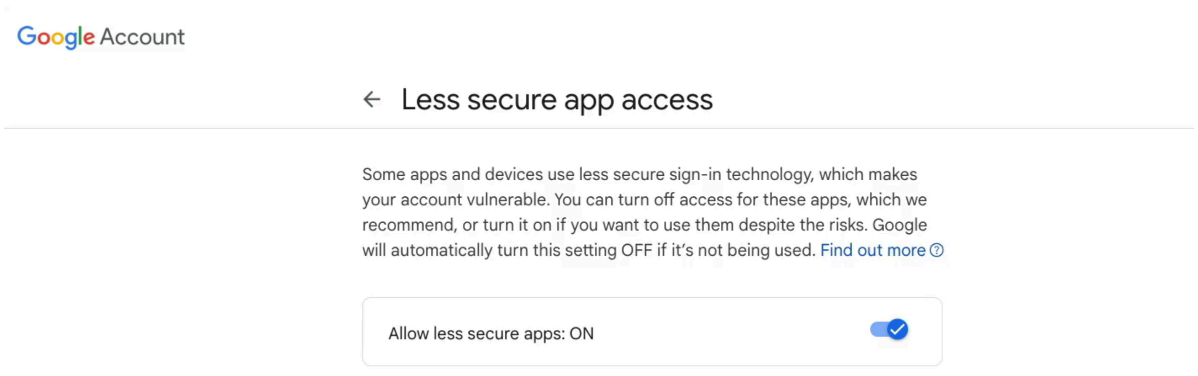
Name	Value
=::	::\
ALLUSERSPROFILE	C:\ProgramData
APPDATA	C:\Users\eduar\AppData\Roaming
ChocolateyInstall	C:\ProgramData\chocolatey
ChocolateyLastPathUpdate	133301326981603737
CommonProgramFiles	C:\Program Files\Common Files

To generate the **Linode API Token**:

- Go to Linode Dashboard.
- Click on your **Profile Picture → API Tokens**.
- Click **Create a Personal Access Token**.
- Set **Label**: `python-monitor`.
- **Expiry**: 6 months (default).
- Select **Read/Write Access**.
- Click **Create Token** and copy the token.

To generate the **Gmail App Password:**

- Enable **Less Secure App Access** in your Gmail account (if two-factor authentication is not enabled).



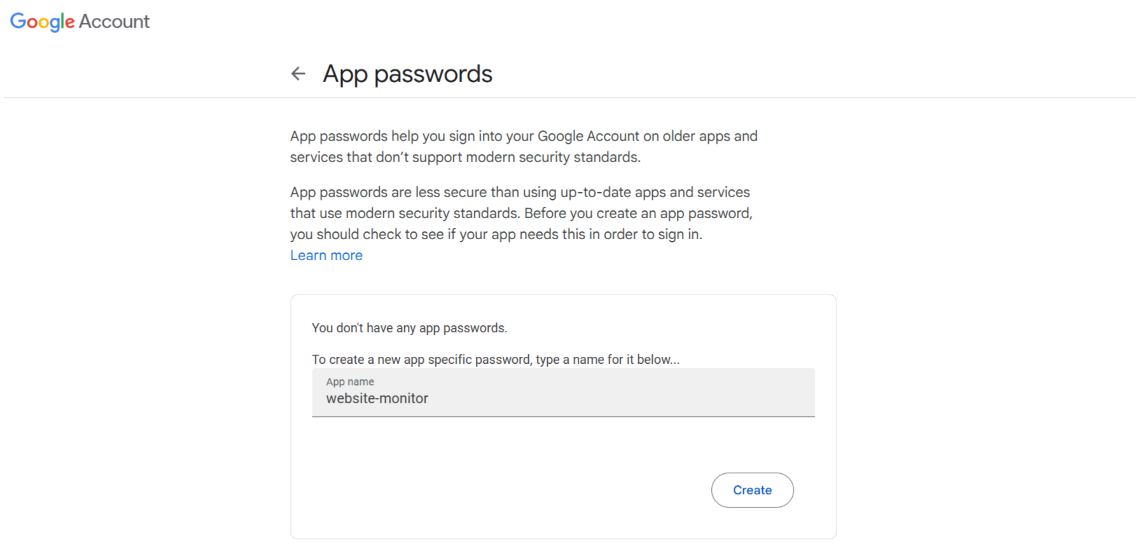
Google Account

Less secure app access

Some apps and devices use less secure sign-in technology, which makes your account vulnerable. You can turn off access for these apps, which we recommend, or turn it on if you want to use them despite the risks. Google will automatically turn this setting OFF if it's not being used. Find out more

Allow less secure apps: ON

- If two-factor authentication is enabled, create an **App Password** by following these steps:
    - Go to Google App Passwords.
    - Choose **App Name**: `website-monitor`.
    - Copy the generated password.
    - Add it to **EMAIL_PASSWORD** in PyCharm.



Google Account

App passwords

App passwords help you sign into your Google Account on older apps and services that don't support modern security standards.

App passwords are less secure than using up-to-date apps and services that use modern security standards. Before you create an app password, you should check to see if your app needs this in order to sign in.
Learn more

You don't have any app passwords.

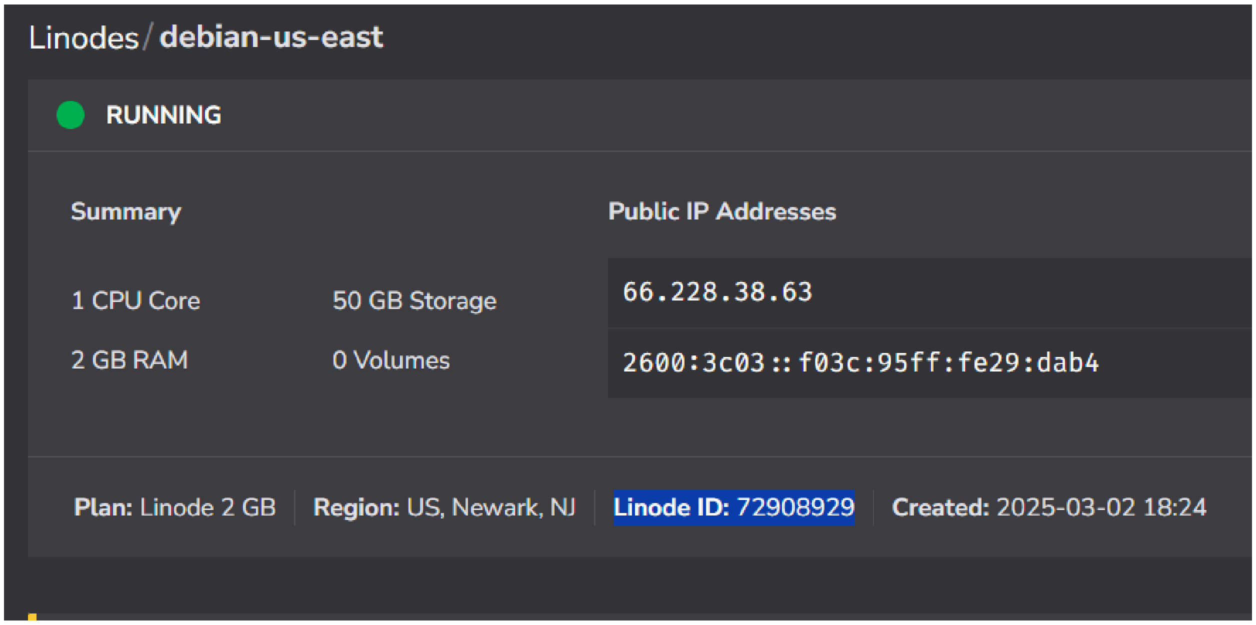
To create a new app specific password, type a name for it below...

App name
website-monitor

Step 6: Write the Monitoring Script

Create a file `monitor-website.py`

→ Note: Make sure you take note of the **URL** of the website to be monitored, and the Linode ID


Linodes / debian-us-east

RUNNING

Summary		Public IP Addresses
1 CPU Core	50 GB Storage	66.228.38.63
2 GB RAM	0 Volumes	2600:3c03::f03c:95ff:fe29:dab4

Plan: Linode 2 GB | **Region:** US, Newark, NJ | **Linode ID:** 72908929 | **Created:** 2025-03-02 18:24

Example: `monitor-website.py` :

```
import requests
import smtplib
import os
import paramiko
import linode_api4
import time
import schedule

EMAIL_ADDRESS = os.environ.get('EMAIL_ADDRESS')
EMAIL_PASSWORD = os.environ.get('EMAIL_PASSWORD')
LINODE_TOKEN = os.environ.get('LINODE_TOKEN')


def restart_server_and_container():
    # restart linode server
    print('Rebooting the server...')
    client = linode_api4.LinodeClient(LINODE_TOKEN)
    nginx_server = client.load(linode_api4.Instance, 72908929)
    nginx_server.reboot()

    # restart the application
    while True:
        nginx_server = client.load(linode_api4.Instance, 72908929)
        if nginx_server.status == 'running':
            time.sleep(5)
            restart_container()
            break


def send_notification(email_msg):
    print('Sending an email...')
    with smtplib.SMTP('smtp.gmail.com', 587) as smtp:
        smtp.starttls()
        smtp.ehlo()
        smtp.login(EMAIL_ADDRESS, EMAIL_PASSWORD)
        message = f"Subject: SITE DOWN\n{email_msg}"
        smtp.sendmail(EMAIL_ADDRESS, EMAIL_ADDRESS, message)

def restart_container():
    print('Restarting the application...')
    ssh = paramiko.SSHClient()
    ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
    ssh.connect(hostname='66.228.38.63', username='root', key_filename='/home
    stdin, stdout, stderr = ssh.exec_command('docker start 10da2fcbd143')
    print(stdout.readlines())
    ssh.close()


def monitor_application():
    try:
        response = requests.get('http://66-228-38-63.ip.linodeusercontent.com:80
        if response.status_code == 200:
            print('Application is running successfully!')
        else:
            print('Application Down. Fix it!')
            msg = f'Application returned {response.status_code}'
            send_notification(msg)
            restart_container()
    except Exception as ex:
        print(f'Connection error happened: {ex}')
        msg = 'Application not accessible at all'
        send_notification(msg)
        restart_server_and_container()


schedule.every(5).seconds.do(monitor_application)

while True:
    schedule.run_pending()
```

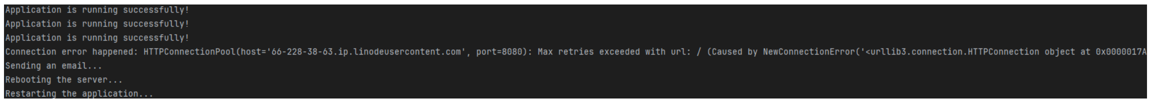
Step 7: Run the Application

1. Run the script in PyCharm.
2. Stop the container manually:

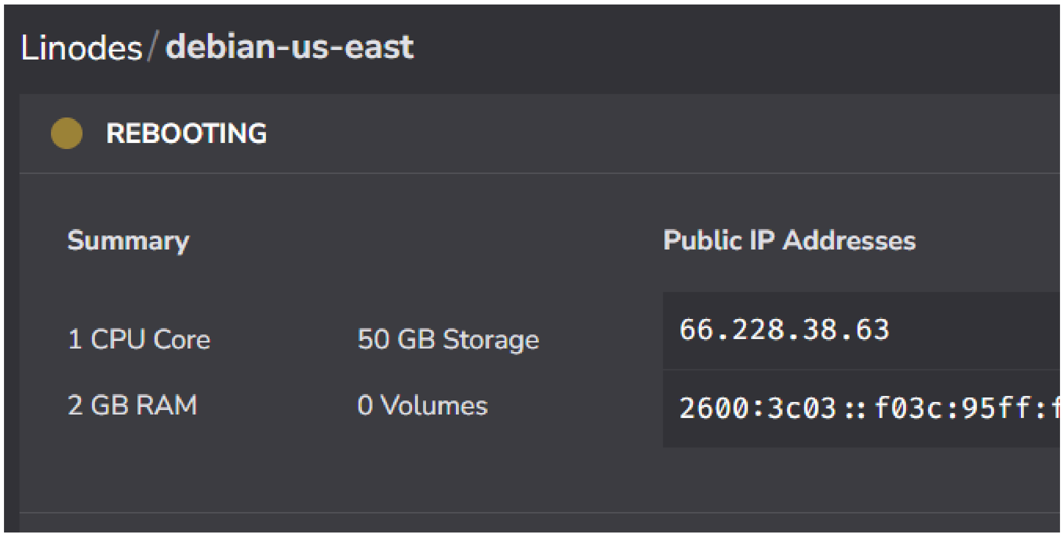
```
ssh root@<linode-public-ip>
docker ps
docker stop <container-id>
```

3. The script should:

- Detect the website is down.



```
Application is running successfully!
Application is running successfully!
Application is running successfully!
Connection error happened: HTTPConnectionPool(host='66-228-38-63.ip.linodeusercontent.com', port=8080): Max retries exceeded with url: / (Caused by NewConnectionError('<urllib3.connection.HTTPConnection object at 0x0000017A
Sending an email...
Rebooting the server...
Restarting the application...
```

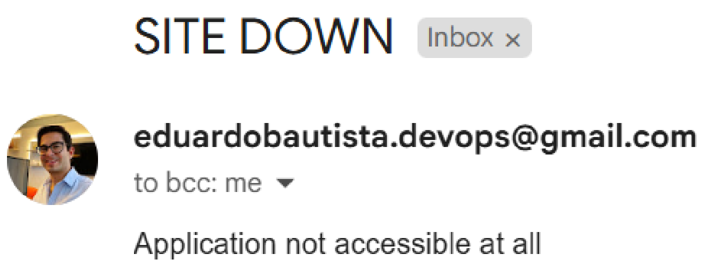


Linodes / debian-us-east

REBOOTING

Summary		Public IP Addresses
1 CPU Core	50 GB Storage	66.228.38.63
2 GB RAM	0 Volumes	2600:3c03::f03c:95ff:f

- Send an email.



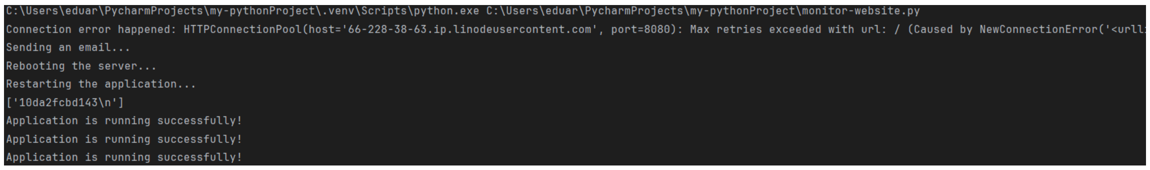
SITE DOWN Inbox ×

eduardobautista.devops@gmail.com

to bcc: me

Application not accessible at all

- Restart the server and container.



```
C:\Users\eduar\PycharmProjects\my-pythonProject\.venv\Scripts\python.exe C:\Users\eduar\PycharmProjects\my-pythonProject\monitor-website.py
Connection error happened: HTTPConnectionPool(host='66-228-38-63.ip.linodeusercontent.com', port=8080): Max retries exceeded with url: / (Caused by NewConnectionError('<urll
Sending an email...
Rebooting the server...
Restarting the application...
['10da2fcbd143\n']
Application is running successfully!
Application is running successfully!
Application is running successfully!
```

Step 8: Clean Up

- Stop the script.

- Delete the Linode instance.
