Placement Empowerment Program

Cloud Computing and DevOps Centre

Write a Python Script to Monitor an Application: Create a Python script that sends periodic HTTP requests to your application and alerts you if it's down.

Name: Samiya Afreen J   Department: CSE

Introduction

In cloud computing and web development, ensuring the availability and performance of applications is crucial. Downtime can lead to user dissatisfaction and potential revenue loss. Therefore, monitoring applications for uptime is an essential practice.

This Proof of Concept (POC) demonstrates how to build a Python script that periodically checks the availability of an application by sending HTTP requests to its URL. It alerts the user if the application is down, helping to maintain consistent availability and reliability.

Overview

The POC involves creating a Python script that:

1. Sends periodic GET requests to a specified application URL.

2. Checks the HTTP response status code.

3. Alerts the user if the status code is anything other than 200 (OK).

4. Handles request exceptions gracefully, providing error messages if the application is unreachable.

The script uses the popular requests library, known for its simplicity and ease of use for sending HTTP requests in Python. The monitoring is done at regular intervals, allowing continuous application health checks.

Objectives

The main objectives of this POC are:

1. To develop a Python script that monitors the availability of an application by periodically checking its URL.

2. To ensure the script alerts the user when the application is down or unreachable.

3. To provide hands-on experience in using the requests library for HTTP requests in Python.

4. To demonstrate how to implement basic error handling for network-related exceptions.

5. To create a foundation for more advanced monitoring solutions, such as integrating notifications via email or messaging platforms.

Importance

1. Proactive Monitoring: This POC provides a proactive approach to monitoring application uptime, ensuring issues are identified promptly.

2. Cost Efficiency: It offers a cost-effective alternative to expensive third-party monitoring tools.

3. Skill Development: This POC enhances skills in Python scripting, HTTP requests handling, and application monitoring.

4. Foundation for Advanced Monitoring: It serves as a foundational step toward building more sophisticated monitoring systems with alerting and reporting functionalities.

5. Reliability and Availability: Ensures high availability and reliability of applications, which is critical for maintaining user satisfaction and trust.

Step-by-Step Overview

Step 1:

Open **Command Prompt** on your Windows laptop.

Type the following command and press **Enter**:

python --version

If you see a version number (e.g., Python 3.x.x), you have Python installed.

If not, download and install Python from python.org, making sure to check the box **"Add Python to PATH"** during installation.

```
C:\Users\samiyaafreen>python --version
Python 3.13.2
```

Step 2:

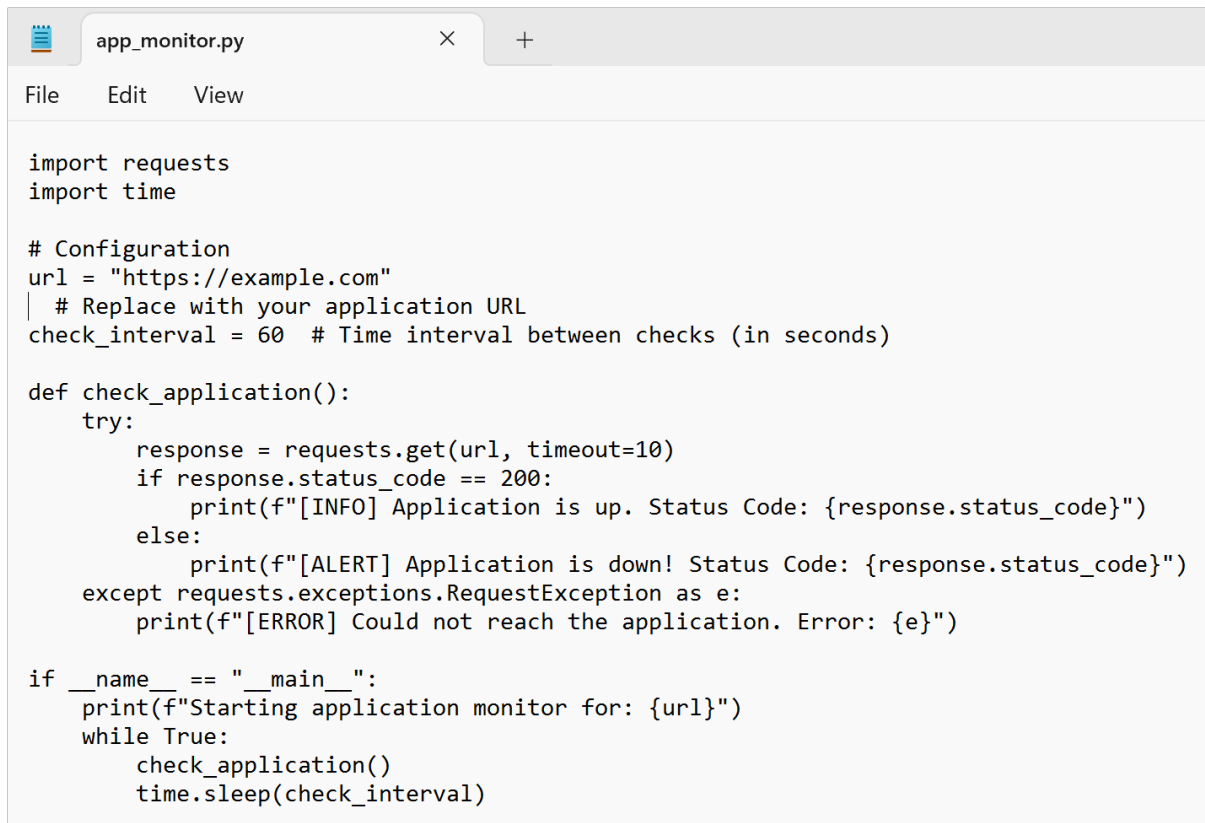We need the requests library. In the same Command Prompt, type:

pip install requests

This will install the requests library needed to send HTTP requests.

```
C:\Users\samiyaafreen>pip install requests
```

Step 3:

Open **Notepad** and type the following code and Save the file as **app_monitor.py** on your Desktop.

```
import requests
import time

# Configuration
url = "https://example.com"
 # Replace with your application URL
check_interval = 60  # Time interval between checks (in seconds)

def check_application():
    try:
        response = requests.get(url, timeout=10)
        if response.status_code == 200:
            print(f"[INFO] Application is up. Status Code: {response.status_code}")
        else:
            print(f"[ALERT] Application is down! Status Code: {response.status_code}")
    except requests.exceptions.RequestException as e:
        print(f"[ERROR] Could not reach the application. Error: {e}")

if __name__ == "__main__":
    print(f"Starting application monitor for: {url}")
    while True:
        check_application()
        time.sleep(check_interval)
```

Step 4:

Open **Command Prompt**.

cd Desktop

This changes the directory to your Desktop where you saved the

script.

Run the Script

python app_monitor.py

```
C:\Users\samiyaafreen\Desktop>python app_monitor.py
Starting application monitor for: https://example.com
[INFO] Application is up. Status Code: 200
[INFO] Application is up. Status Code: 200
```

You should see output like this:

Starting application monitor for: https://example.com
[INFO] Application is up. Status Code: 200

This means the script successfully checked https://example.com and found it online.

- Status Code: 200 indicates the website is reachable and working fine.
- The script will check the URL every 60 seconds.

To stop the monitoring, press Ctrl + C in the Command Prompt window.

Outcomes

By completing this POC of monitoring an application using a Python script, you will:

1. Develop a Python script that periodically sends HTTP requests to a specified URL to check the application's availability.

2. Detect application downtime by analyzing HTTP status codes and receiving alerts if the application is down or unreachable.

3. Gain hands-on experience with the requests library in Python, including implementing error handling for network-related issues.