**Step 1: Set Up a Local Ubuntu VM**

**1.1 Install VirtualBox**

1. Download and install VirtualBox from the [official website](https://www.virtualbox.org/).
2. Launch VirtualBox after installation.

**1.2 Create a Local Ubuntu VM**

1. **Download Ubuntu Server ISO:**
   * Download the Ubuntu Server ISO from the [official Ubuntu website](https://ubuntu.com/download/server).
2. **Create a New VM in VirtualBox:**
   * Open VirtualBox and click **New**.
   * Name the VM (e.g., Local-Ubuntu-VM).
   * Set the type to **Linux** and version to **Ubuntu (64-bit)**.
   * Allocate resources:
     + RAM: 2 GB (or more, depending on your system).
     + CPU: 2 cores.
     + Storage: 20 GB (dynamically allocated).
3. **Attach the Ubuntu ISO:**
   * Select the VM and click **Settings** → **Storage**.
   * Under **Controller: IDE**, click the disk icon and choose the Ubuntu ISO file.
4. **Install Ubuntu:**
   * Start the VM and follow the Ubuntu Server installation steps.
   * Set up a user account and enable SSH for remote access.

**Step 2: Install and Configure Resource Monitoring**

**2.1 Install Prometheus**

1. **SSH into the Local VM:**
   * Use a terminal or SSH client to connect to the VM.
   * Example:

bash

Copy

ssh username@<local-vm-ip>

1. **Download and Install Prometheus:**
   * Run the following commands:

bash

Copy

wget https://github.com/prometheus/prometheus/releases/download/v2.30.3/prometheus-2.30.3.linux-amd64.tar.gz

tar -xvzf prometheus-2.30.3.linux-amd64.tar.gz

cd prometheus-2.30.3.linux-amd64

1. **Configure Prometheus:**
   * Edit the prometheus.yml file to scrape metrics from the local VM:

yaml

Copy

global:

scrape\_interval: 15s

scrape\_configs:

- job\_name: 'node'

static\_configs:

- targets: ['localhost:9100']

1. **Start Prometheus:**
   * Run Prometheus in the background:

bash

Copy

./prometheus --config.file=prometheus.yml &

**2.2 Install Node Exporter**

1. **Download and Install Node Exporter:**
   * Run the following commands:

bash

Copy

wget https://github.com/prometheus/node\_exporter/releases/download/v1.3.1/node\_exporter-1.3.1.linux-amd64.tar.gz

tar -xvzf node\_exporter-1.3.1.linux-amd64.tar.gz

cd node\_exporter-1.3.1.linux-amd64

1. **Start Node Exporter:**
   * Run Node Exporter in the background:

bash

Copy

./node\_exporter &

**2.3 Install Grafana**

1. **Install Grafana:**
   * Run the following commands:

bash

Copy

sudo apt-get install -y adduser libfontconfig1

wget https://dl.grafana.com/oss/release/grafana\_8.2.3\_amd64.deb

sudo dpkg -i grafana\_8.2.3\_amd64.deb

sudo systemctl start grafana-server

1. **Access Grafana:**
   * Open a browser and go to http://<local-vm-ip>:3000.
   * Log in with the default credentials (admin/admin).
2. **Add Prometheus as a Data Source:**
   * Go to **Configuration** → **Data Sources** → **Add Data Source**.
   * Select **Prometheus** and enter the URL http://localhost:9090.
3. **Create a Dashboard:**
   * Import a dashboard (e.g., Node Exporter Full) to visualize CPU, memory, and disk usage.

**Step 3: Set Up GCP and Configure Auto-Scaling**

**3.1 Create a GCP Account and Project**

1. **Sign Up for GCP:**
   * Go to the [GCP Console](https://console.cloud.google.com/) and create an account.
2. **Create a New Project:**
   * Click **Create Project** and name it (e.g., AutoScale-Project).

**3.2 Create a VM Instance on GCP**

1. **Go to Compute Engine:**
   * Navigate to **Compute Engine** → **VM Instances** → **Create**.
2. **Configure the VM:**
   * Name: Cloud-VM.
   * Region: Choose a region close to you.
   * Machine Type: e2-medium.
   * Boot Disk: Ubuntu 20.04 LTS.
   * Allow HTTP/HTTPS traffic.
3. **Deploy the Sample Application:**
   * SSH into the GCP VM:

bash

Copy

gcloud compute ssh cloud-vm

* + Install Apache:

bash

Copy

sudo apt update

sudo apt install apache2

**3.3 Configure Auto-Scaling**

1. **Create an Instance Template:**
   * Go to **Compute Engine** → **Instance Templates** → **Create**.
   * Use the same configuration as the Cloud-VM.
2. **Create an Instance Group:**
   * Go to **Compute Engine** → **Instance Groups** → **Create**.
   * Choose **Managed Instance Group** and select the template.
   * Set the auto-scaling policy:
     + Metric: CPU utilization.
     + Target: 75%.
     + Minimum instances: 1.
     + Maximum instances: 5.
3. **Set Up Cloud Monitoring:**
   * Go to **Monitoring** → **Alerting** → **Create Policy**.
   * Add a condition for CPU usage > 75%.
   * Set the notification channel (e.g., email).

**Step 4: Deploy the Sample Application**

**4.1 Deploy on Local VM**

1. **Install Apache:**
   * Run the following commands:

bash

Copy

sudo apt update

sudo apt install apache2

1. **Test the Application:**
   * Open a browser and go to http://<local-vm-ip>.

**4.2 Deploy on GCP VM**

1. **SSH into the GCP VM:**
   * Use the GCP Console or gcloud command:

bash

Copy

gcloud compute ssh cloud-vm

1. **Install Apache:**
   * Run the following commands:

bash

Copy

sudo apt update

sudo apt install apache2

1. **Test the Application:**
   * Open a browser and go to http://<gcp-vm-external-ip>.

**Step 5: Test the Workflow**

1. **Simulate High Resource Usage:**
   * On the local VM, run a CPU-intensive task:

bash

Copy

stress --cpu 2 --timeout 300

1. **Monitor in Grafana:**
   * Check the Grafana dashboard for CPU usage exceeding 75%.
2. **Verify Auto-Scaling:**
   * Go to the GCP Console and check if the instance group scales up.

**Architecture Diagram**

plaintext

Copy

+-------------------+ +-------------------+ +-------------------+

| Local VM | | GCP Cloud | | Sample App |

|-------------------| |-------------------| |-------------------|

| - Prometheus | | - Auto-Scaling | | - Apache/Nginx |

| - Grafana | | - Cloud Monitoring| | - Sample Web App |

| - Node Exporter | | - VM Instances | +-------------------+

+-------------------+ +-------------------+

| |

| Monitor Resource Usage | Trigger Auto-Scaling

| (CPU > 75%) | (Migrate to Cloud)

v v

+-------------------+ +-------------------+

| Alert Trigger | ----> | Cloud VM |

+-------------------+ +-------------------+

This step-by-step guide provides everything you need to implement the project, from setting up the local VM to configuring auto-scaling on GCP. Let me know if you need further clarification