Creation of VM on GCP using Terraform:

Terraform is a popular infrastructure-as-code tool used to manage and provision resources in various cloud platforms.

In order to create a Virtual Machine (VM) using Google Cloud Platform (GCP) and Terraform, we followed these steps:

1. Install Dependencies:

- Install Terraform: Download and install Terraform from the official website (https://www.terraform.io/). Make sure to add the Terraform binary to your system's PATH. To learn how to do this, refer to the section titled "Instructions on how to add the Terraform binary to your system's PATH."
- Install Google Cloud SDK: Download and install the Google Cloud SDK (https://cloud.google.com/sdk/docs/install). Google Cloud SDK, also known as the "gcloud" command-line tool, is a set of command-line utilities and libraries provided by Google to interact with various Google Cloud Platform (GCP) services and resources. It allows developers, system administrators, and other users to manage and interact with their GCP resources from the command line.

2. Set up GCP Project and Credentials:

- Create a GCP project in the Google Cloud Console (https://console.cloud.google.com/).
- Generate a service account key for your project with the required permissions to create resources. Save the JSON key file securely.

3. Configure Google Cloud SDK:

- Open a terminal or command prompt and run gcloud init to authenticate and set up your GCP credentials.

4. Create a Terraform Configuration File:

- Create a new directory for your Terraform configuration.
- Inside the directory, create a new file with a .tf extension (e.g., main.tf) to define your GCP resources.

5. Define Provider and Resource Configuration:

- In the main.tf file, start by defining the GCP provider and resource(s) you want to create. For a VM, you'll use the google_compute_instance resource.
  A Google Compute Engine instance, commonly referred to as a "Google Compute instance" or simply "VM instance," is a virtual machine (VM) hosted on Google Cloud Platform (GCP). It allows users to run applications and services in a virtualized environment with access to resources such as CPUs, memory, and storage.
  Key features of Google Compute Engine instances include:
  a. Virtual Machine Types: Google Compute Engine provides a variety of machine types to cater to different workloads, ranging from small to high-performance configurations with custom CPU and memory allocations.

b. Predefined and Custom Images: You can use predefined images from Google (e.g., Ubuntu, CentOS, Debian) or create custom images to use as the base for your VM instances.
c. Persistent Disks: Compute Engine offers persistent block storage disks that can be attached to instances. These disks retain data even if the instance is terminated.
d. Network Configuration: You can define custom networks, subnets, and firewall rules to control communication between instances and other resources.
e. Load Balancing: Google Compute Engine allows you to set up load balancing to distribute incoming traffic across multiple VM instances.
f. Auto Scaling: You can configure auto scaling to automatically add or remove VM instances based on demand.
g. Snapshots and Backups: Compute Engine provides snapshot and backup functionality to create point-in-time copies of your disks.
h. Instance Templates: You can use instance templates to define reusable configurations for VM instances, making it easier to create consistent deployments.

In the '**main.tf'** file, there is an example of how to create a virtual machine running Ubuntu 18.04. The VM is configured with an n1-standard-1 machine type, offering a balanced amount of resources, and it uses a standard persistent boot disk to store the operating system and data.

6. Initialize and Apply Terraform Configuration:

- Open a terminal/command prompt and navigate to your Terraform configuration directory.
- Run terraform init to initialize the working directory and download the required plugins.
- Run terraform apply to create the VM. Terraform will show you a plan of what resources will be created. Type yes when prompted to confirm.

7. Wait for the VM to be Created:

- Terraform will now create the VM on GCP. This may take a few minutes depending on the resources being created.

8. Accessing the VM:

- Once the VM is created, you can find its public IP address and connect to it using SSH or any other method appropriate for your operating system.

Instructions on how to add the Terraform binary to your system's PATH :

For Windows:

1. Place the Terraform binary (`terraform.exe`) in a directory that is already part of your system's PATH, or create a new directory specifically for Terraform binaries.

2. Open the Start menu, search for "Environment Variables," and select "Edit the system environment variables."

3. In the System Properties window, click the "Environment Variables" button.

4. In the Environment Variables window, scroll down to the "System variables" section and find the "Path" variable. Click on "Edit."

5. Click "New" to add a new entry to the PATH. Enter the path to the directory containing the Terraform binary (`terraform.exe`).

6. Click "OK" on all windows to save the changes.

7. Open a new command prompt or PowerShell window, and you should now be able to run the `terraform` command from any directory.

For macOS and Linux:

1. Place the Terraform binary (`terraform`) in a directory that is already part of your system's PATH, or create a new directory specifically for Terraform binaries.

2. Open a terminal window.

3. Use a text editor (such as nano, vim, or a graphical editor) to open the shell configuration file for your shell (e.g., `.bashrc`, `.bash_profile`, `.zshrc`, etc.).

For example, with nano editor:

nano ~/.bashrc

4. Add the following line at the end of the file, replacing `/path/to/terraform` with the actual path to the directory containing the Terraform binary:

export PATH="/path/to/terraform:$PATH"

5. Save the changes and exit the editor.

6. To apply the changes immediately, run the following command in the terminal:

source ~/.bashrc

If you are using a different shell configuration file, replace `.bashrc` with the appropriate filename.

7. Open a new terminal window, and you should now be able to run the `terraform` command from any directory.