3-tier WordPress Application

- **Web Tier** (handles incoming HTTP requests, serves static files, and forwards requests to the App Tier)
- **App Tier** (hosts the WordPress application and PHP processing)
- **Database Tier** (hosts the MySQL database for WordPress)

Step 1: Set Up Each VM and Cloud SQL

1. **Create VM Instances and Cloud SQL**: Set up two separate VMs and one Cloud SQL (MySQL) on GCP:
    - **Web Tier -** Only allow HTTP/HTTPS traffic.
    - **App Tier -** Allow internal traffic from VM_1 for WordPress PHP processing.
    - **Database Tier -** Allow internal traffic from VM_2 only.
2. **Assign Internal IPs**: Ensure each VM has an **internal IP** address for secure communication within GCP's network.

VM_1 - Web Tier

1. **Installation (Apache)**:
    - SSH into VM_1 and install Apache:
      sudo apt update
      sudo apt install apache2 -y
2. **Configure Web Server:**
    - Configure Apache to act as a reverse proxy to VM_2 (App Tier). In Apache, add the following to your configuration file (e.g., /etc/apache2/sites-available/wordpress.conf):
      ProxyPass / http://app-tier-internal-ip/
      ProxyPassReverse / http://app-tier-internal-ip/
    - Enable necessary Apache modules:
      sudo a2enmod proxy
      sudo a2enmod proxy_http
      sudo a2enmod proxy_balancer
      sudo systemctl restart apache2
3. **Edit the Apache Configuration File:**
    - Open the default Apache configuration file or create a new virtual host configuration file for WordPress (usually found in /etc/apache2/sites-available/).
      sudo nano /etc/apache2/sites-available/wordpress.conf
4. **Configure Reverse Proxy:**

- Add the following configuration to forward requests to **VM_2** (App Tier):
  ```
  <VirtualHost *:80>
  ServerName your-domain-or-vm1-external-ip

  # Enable Proxy
  ProxyPreserveHost On
  ProxyPass / http://app-tier-internal-ip/
  ProxyPassReverse / http://app-tier-internal-ip/

  # Log file locations (optional)
  ErrorLog ${APACHE_LOG_DIR}/error.log
  CustomLog ${APACHE_LOG_DIR}/access.log combined
  </VirtualHost>
  ```
- Replace app-tier-internal-ip with the internal IP of VM_2 (App Tier).
- This configuration tells Apache to pass all requests (the / path) received on VM_1 to the backend VM_2.

5. **Enable the Site Configuration:**
   - If you created a new configuration file (wordpress.conf), enable it:
     ```
     sudo a2ensite wordpress.conf
     sudo systemctl reload apache2
     ```

VM_2 - App Tier

2. **Install PHP and WordPress**:
   - SSH into VM_2.
   - Install PHP and dependencies:
     ```
     sudo apt update
     sudo apt install php libapache2-mod-php php-mysql -y
     ```
   - Download and configure WordPress:
     ```
     cd /var/www/html
     sudo wget -c http://wordpress.org/latest.tar.gz
     sudo tar -xzvf latest.tar.gz
     sudo rm latest.tar.gz
     sudor mv wordpress/* /var/www/html/
     sudo chown -R www-data:www-data /var/www/html/
     sudo chmod -R 755 /var/www/html/
     ```
3. **Configure WordPress Database Connection**:
   - Rename the wp-config-sample.php to wp-config.php:
     ```
     sudo mv /var/www/html/wp-config-sample.php /var/www/html/wp-config.php
     ```

4. **Configure WordPress to Use Cloud SQL**:
   - Edit wp-config.php and set the database connection parameters to point to Cloud SQL (Database Tier) using its internal IP.
     sudo nano /var/www/html/wp-config.php
   - Edit the wp-config.php file on VM_2:
     define('DB_NAME', 'wordpress');
     define('DB_USER', 'wp_user');
     define('DB_PASSWORD', 'your_password');
     define('DB_HOST', '127.0.0.1:3306'); // If using Cloud SQL Proxy
   - If using **Private IP** without the proxy, replace DB_HOST with the private IP of the Cloud SQL instance.
5. **Test the Connection (Make sure Cloud SQL part already integrated):**
   - Enable the rewrite module for clean URLs in WordPress:
     sudo a2enmod rewrite
   - Reload or restart Apache on VM_2:
     sudo systemctl restart apache2
   - Go to VM_1 (Web Tier) IP in a browser to ensure WordPress can connect to Cloud SQL through VM_2.
6. **Install Cloud SQL Proxy** (optional but recommended for secure connection):
   - Download and install the Cloud SQL Proxy:
     wget https://dl.google.com/cloudsql/cloud_sql_proxy.linux.amd64 -O cloud_sql_proxy
     chmod +x cloud_sql_proxy
   - Start the Cloud SQL Proxy (replace INSTANCE_CONNECTION_NAME with your instance's connection name from Cloud SQL):
     ./cloud_sql_proxy -instances=YOUR_PROJECT:YOUR_REGION:YOUR_INSTANCE_NAME=tcp:3306 &

Cloud SQL - Database Tier

1. **Create a Cloud SQL Instance**:
   - In the GCP Console, go to SQL > Create Instance.
   - Choose MySQL as the database engine.
   - Configure the instance settings such as region, zone, and machine type.
2. **Create WordPress Database and User**:
   - After the instance is set up, go to Databases and create a new database, e.g., wordpress.
   - Go to Users and create a user, e.g., wp_user, and set a strong password.

3. **Enable Connections to Cloud SQL**:
    - Go to the **Connections** tab and enable **Private IP** (preferred for security).

Step 2: Firewall Setup

Here are the firewall rules for your dynamic WordPress website hosted in a multi-tier architecture on Google Cloud Platform (GCP). Based on the subnet IP ranges you've provided:

- **Web Tier (VM_1)**: Subnet IP Address
- **App Tier (VM_2)**: Subnet IP Address
- **Database Tier (Cloud SQL)**: Subnet IP Address

Firewall Rules Overview

1. Firewall Rule for Public Access to the Web Tier (VM_1)

Purpose: Allow external traffic (HTTP/HTTPS) from the public internet to reach the Web Tier (VM_1).

2. Firewall Rule for Web Tier (VM_1) to App Tier (VM_2)

Purpose: Allow internal traffic from the Web Tier (VM_1) to the App Tier (VM_2) for dynamic content processing (e.g., WordPress PHP execution).

3. Firewall Rule for App Tier (VM_2) to Database Tier (Cloud SQL or VM_3)

Purpose: Allow internal traffic from the App Tier (VM_2) to the Database Tier (Cloud SQL or VM_3) for database queries and management.

4. Firewall Rule for Database Tier (Cloud SQL or VM_3) to App Tier (Optional - Reverse Traffic)

Purpose: In some cases, reverse traffic from the database to the app server may be required (e.g., callbacks). This rule allows traffic from the Database Tier back to the App Tier, if needed.

5. Firewall Rules for IAP Access

Purpose: To use **IAP for SSH** connections, your VM must allow traffic from **IAP's IP range** on **port 22**.

Step 3: Routes Configuration (By default created once create subnet in VPC)

1. **Create a Custom Route (Optional)**
    - Navigate to **VPC Network** > **Routes**.
    - Click **Create Route**.
    - Provide the following details:

- **Name**: e.g., route-to-nat-gateway.
- **Network**: Select the appropriate VPC network.
- **Destination IP Range**: Define the range of IP addresses for which this route will apply (e.g., 0.0.0.0/0 for all traffic or specific subnet IP ranges like 10.0.2.0/24).
- **Next Hop**: Choose Specify instance and select the NAT Gateway instance or the instance that will act as the next hop for this route.

2. **Verify Routes**
   - Once you create the route, you can check the **Routes** section under **VPC Network** to see all the active routes.
   - Ensure the custom route you created is listed and directs traffic appropriately.

Step 4: Cloud Routers Configuration

- Go to Network Services > Cloud Router.
- Click Create Router.
- Specify the following:
  - **Name:** e.g., wordpress-router.
  - **Network:** Select your VPC.
  - **Region:** The region where your Web and App Tiers are located.

Step 5: Cloud NAT Configuration

- Go to Network Services > Cloud NAT.
- Click Create NAT Gateway.
- Select the Cloud Router you just created.
- Choose the subnet where your Web Tier and App Tier VMs are located.
- Optionally, choose whether you want to auto-assign or manually assign an external IP address for NAT.

Step 6: Reserved IP Configuration

For a 3-tier architecture with Web VM, App VM, and Cloud SQL, you'll need to carefully manage IP assignments based on accessibility and security requirements. Here's how to assign and configure reserved IPs for each component.

1. Web Tier (Web VM)

For the Web VM (publicly accessible), it's recommended to reserve a static external IP so that users can reliably access the website via a fixed IP address.

Steps to Reserve a Static External IP for the Web Tier:

- Go to VPC Network > External IP addresses in the GCP Console.
- Click Reserve Static Address.
- Provide the following details:
  - **Name**: Name the IP (e.g., web-vm-ip).
  - **Network Tier**: Standard (if only testing, use this lower-cost option).
  - **IP Version**: IPv4.
  - **Type**: Regional, in the region where your Web VM is located.
  - Click **Reserve**.

Configuration:

- Assign the reserved static IP to the Web VM when creating or editing the VM instance.
- This IP will provide a fixed address for your Web Tier so that users always reach the same IP when accessing the website.

2. App Tier (App VM)

For the App VM, which should be internally accessible only (not exposed to the internet directly), you should reserve a static internal IP. This keeps the IP consistent within the Virtual Private Cloud (VPC) for secure communication between the Web VM and App VM.

Steps to Reserve a Static Internal IP for the App Tier:

- Go to VPC Network > IP addresses > Internal IP addresses.
- Click Reserve Static Internal IP.
- Provide the following details:
  - **Name:** Name the IP (e.g., app-vm-internal-ip).
  - **Network:** Select the VPC network where the App VM resides.
  - **Subnetwork:** Choose the subnet assigned to the App Tier.
  - **IP Address:** Leave it as automatic or specify an IP in the subnet range.
  - Click Reserve.

Configuration:

- Assign this reserved internal IP to the App VM. This internal IP allows the Web VM to reliably communicate with the App VM, as requests from the Web Tier are forwarded to the App Tier for dynamic processing.

3. Database Tier (Cloud SQL)

For Cloud SQL, you can choose between a private IP (recommended for internal access only) or a public IP (if external access is needed, which is generally discouraged for security).

Steps to Enable Private IP for Cloud SQL:

- Go to SQL in the GCP Console and select your Cloud SQL instance.
- Go to Connections > Networking > Private IP.

- Enable Private IP and assign the App Tier's subnet for seamless internal communication.
- Note the internal IP address of the Cloud SQL instance, as you'll use this IP for database connections.

Configuration:

- Use the private IP to connect the App VM to Cloud SQL by specifying this IP in WordPress's wp-config.php.

Step 7: Instance Templates Configuration

An instance template is a configuration blueprint for VM instances in a managed instance group. This template will be helpful to automatically create multiple instances with the same configuration for your **Web VM** and **App VM** setup.

- Go to Compute Engine > Instance templates.
- Click Create instance template.
- Name the template, (e.g., web-app-template).
- **Region and Zone:** Select where you want your instances to be created. If you want instances in multiple zones, you can specify this later when creating the instance group.
- **Machine type:** Choose based on the expected load.
- **OS and disk type**: Choose Debian, Ubuntu, or any OS that suits your application.
- **Standard persistent disk or SSD:** Select based on performance and set an appropriate disk size.
- **Networking**: Configure the Network and Subnetwork.
    - In the Network Settings of the instance template, the choice of subnetwork depends on the role and network requirements of the instances you're deploying.
        - **public-web-subnet:** For instances handling public traffic (e.g., web servers).
        - **private-app-subnet:** For internal-only instances (e.g., backend app servers).
        - **proxy-subnet:** For instances acting as load balancers or proxies, requiring both internal and external access.
    - For External IP, choose None if you don't want the instances to be accessible from the internet (typically for App VMs in private configurations). If you need direct internet access, set the External IP to Ephemeral or Static.
- **Firewall rules**: Ensure Allow HTTP traffic and Allow HTTPS traffic are enabled if these instances will handle web requests.
- **Management:** Configure the automatic restart and preemption options based on your needs.
- **Security:** Select Enable Secure Boot if required or leave it as default.

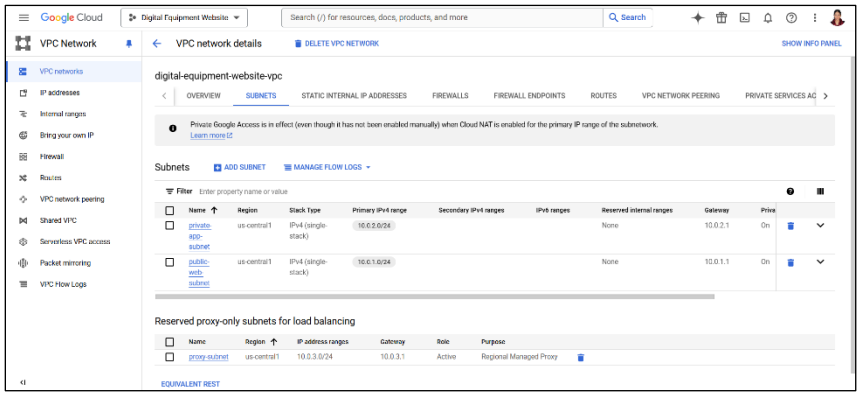
- **Metadata:** Add any startup scripts or custom metadata as needed for configuring your instance on boot.
- After configuring all settings, review your configuration to make sure everything is correct. Click **Create** to save your instance template.

Step 8: Instance Group Configuration

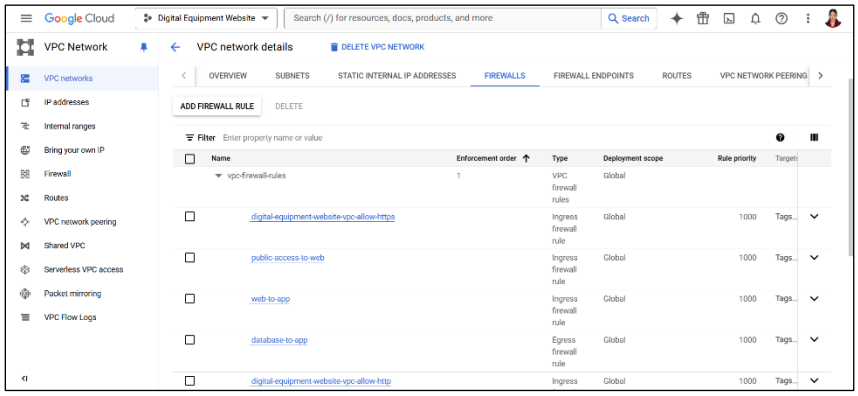
- Go to GCP Console > Compute Engine > Instance groups.
- Click Create instance group.
- Enter a Name for the instance group (e.g., web-app-instance-group).
- Select **Single Zone** or **Multi-Zonal**, depending on your availability requirements:
    - **Single Zone:** All instances will reside in one zone.
    - **Multi-Zonal:** Instances will be distributed across multiple zones in a region for high availability.
- Choose a Region and a Zone based on your infrastructure setup.
- **Instance Template:** Under the Instance Template section, select the existing instance template that you previously created.
- **Number of Instances:** Specify the initial size (number of VM instances) for the group. For example, start with 2 or 3 instances for testing.
- **Autoscaling Configuration (if desired):**
    - Click Enable autoscaling.
    - **Set the Autoscaling policy:**
        - Scale based on CPU utilization: Recommended for workloads with varying compute requirements.
        - Scale based on HTTP Load Balancing capacity: Ideal for backend services behind a load balancer.
    - Define the minimum and maximum number of instances to ensure scalability and cost control.
- **Select or create a health check for your instance group:**
    - Go to Create health check if no health check exists.
    - Configure the following:
        - **Protocol:** HTTP or HTTPS (depending on your app).
        - **Port:** Match the port your app listens to (e.g., 80 for HTTP).
        - **Request Path:** Enter a specific path to check, e.g., /health.
        - **Check interval:** The frequency of health checks.
        - **Timeout and Thresholds:** Adjust if needed for your app.
- **Subnetwork:**
    - Select the appropriate subnet based on your application:

        - Use public-web-subnet if your instances require internet-facing traffic.
        - Use private-app-subnet for internal-only services.
- **External IP:**
    - **None:** If the instance only needs internal communication.
    - **Ephemeral:** If external access is required.
- Review your configuration. Click Create to finish creating the instance group.

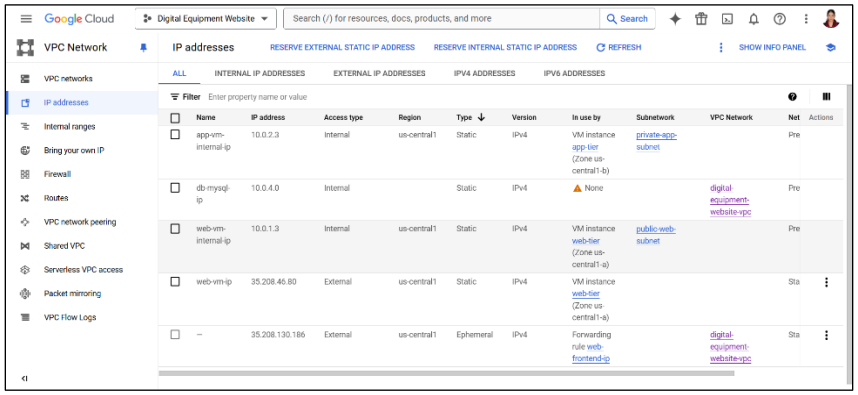
Snapshots



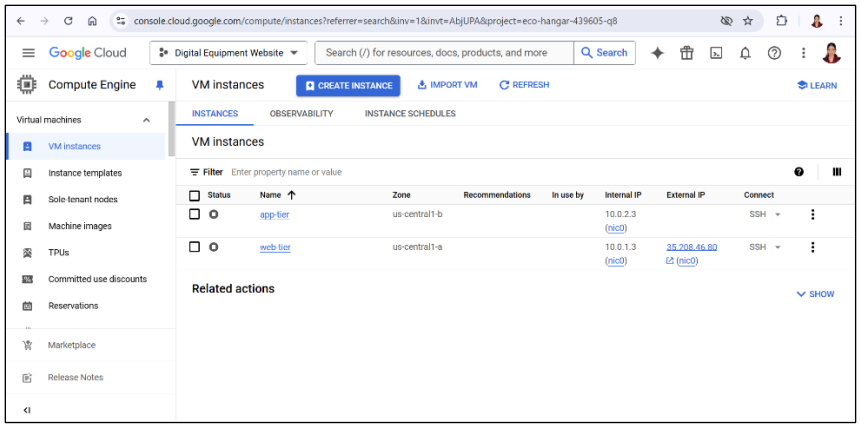
Snapshot 1: Subnets



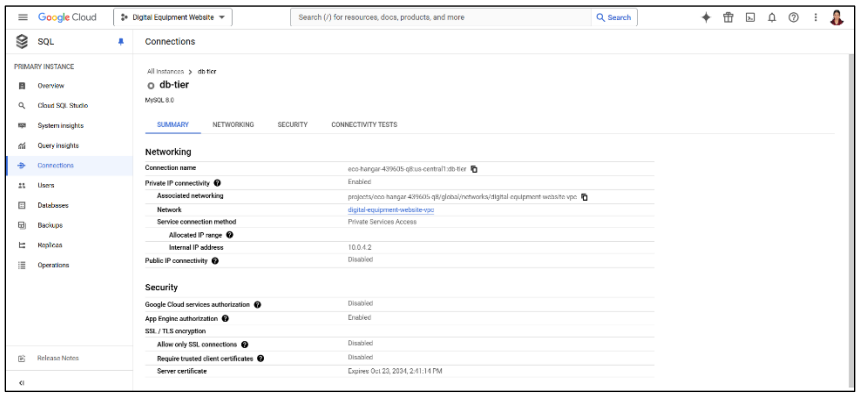
Snapshot 2: Firewalls



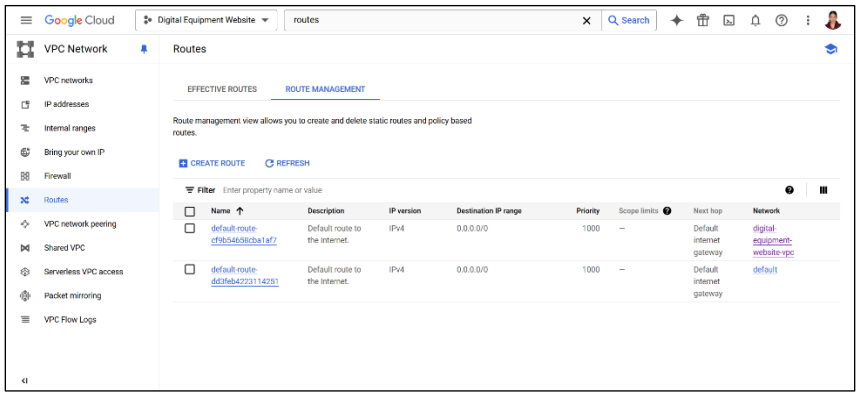
Snapshot 3: IP addresses



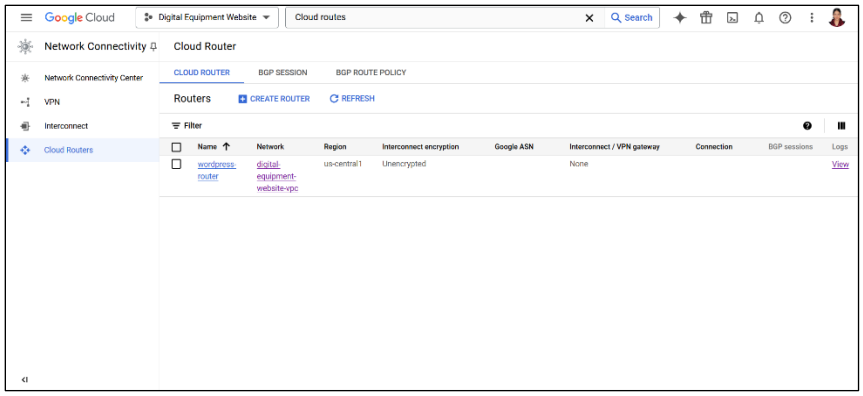
Snapshot 4: VM instances



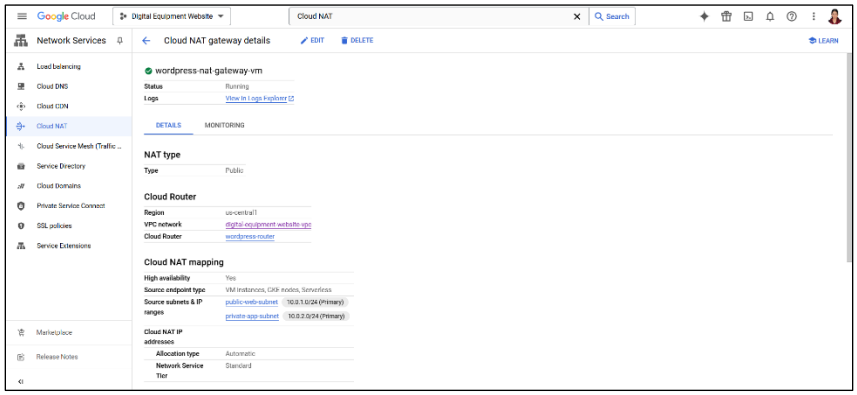
Snapshot 5: Cloud SQL



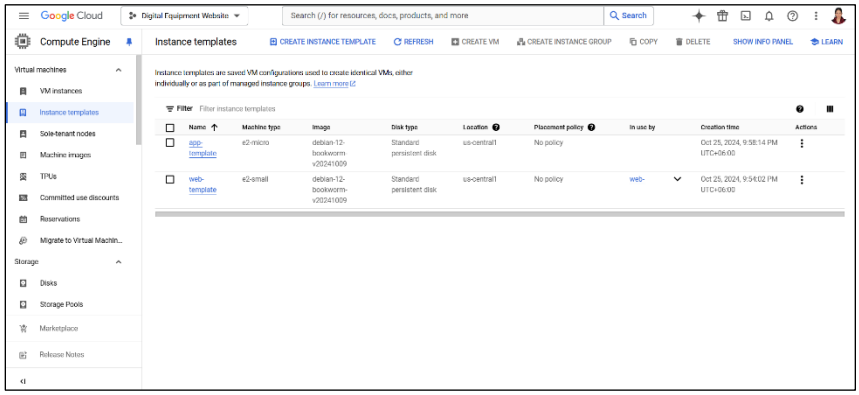
Snapshot 6: Routes



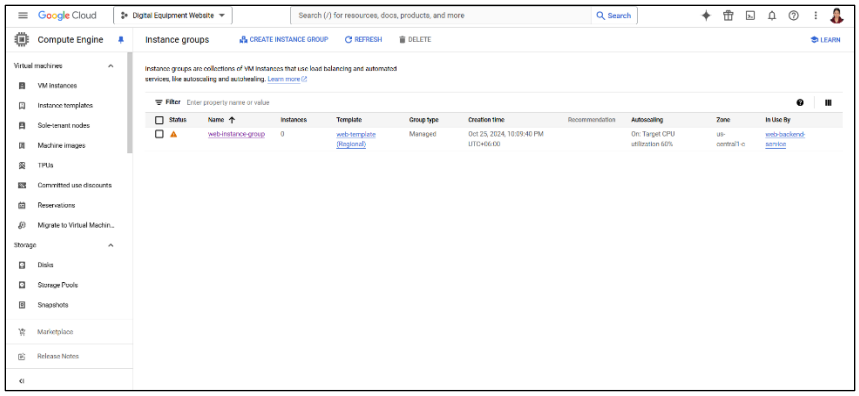
Snapshot 7: Cloud Routers



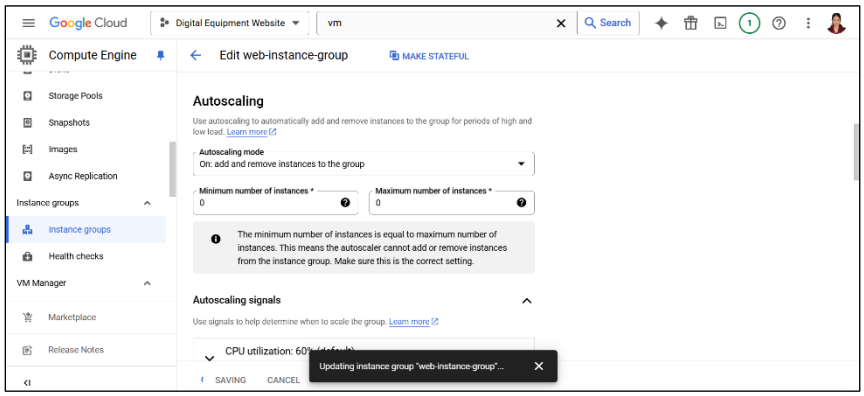
Snapshot 8: Cloud NAT



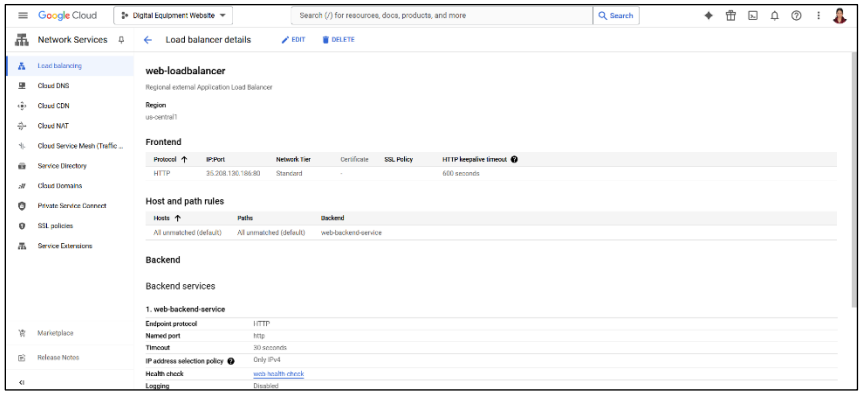
Snapshot 9: Instance templates



Snapshot 10: Instance groups

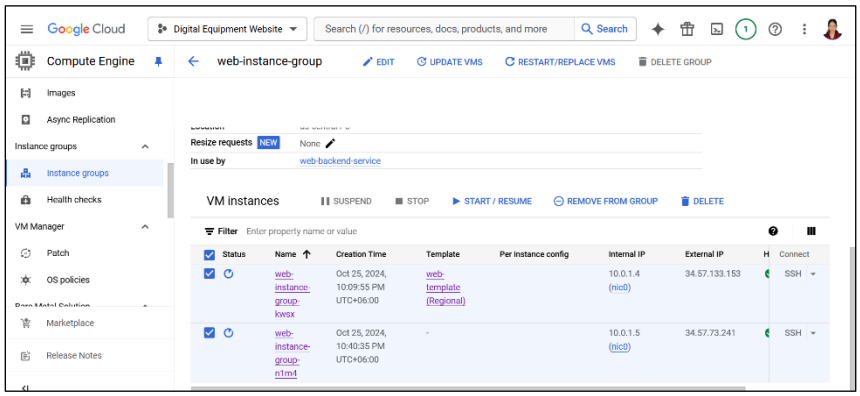


Snapshot 11: Autoscaling

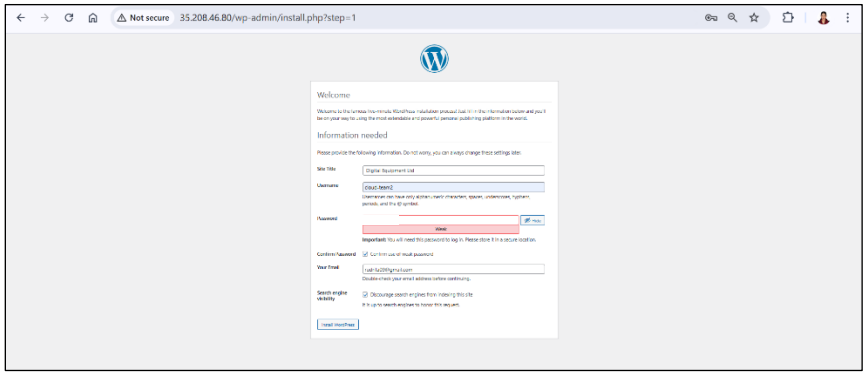


Snapshot 12: Load balancing

Snapshot 13: Monitoring



Snapshot 14: Auto new instance creates (VM down/ overloaded)



Snapshot 15: WordPress Setup