**Documentation: Creating a Highly Available Website Using Azure VMSS**

**1. Website Creation**

Develop a basic website with the desired features. Use HTML, CSS, and JavaScript for front-end design, and optional back-end functionality (e.g., Node.js or Python). Ensure the website files are ready for deployment.

**2. Upload Website to GitHub**

Upload the website files to a GitHub repository. For example, you can clone the repository using:

git

This ensures easy access to the website files for deployment.

**3. Create VM with Custom Script**

In the Azure Portal:

* Create a Virtual Machine (VM) and use a **custom script extension** during deployment.
* The custom script should pull the website files from the GitHub repository and configure the VM to serve the website (e.g., install a web server like Nginx or Apache).

#!/bin/bash

# Update and upgrade the system

sudo apt update -y && sudo apt upgrade -y

# Install Nginx

sudo apt install nginx -y

# Enable Nginx to start on boot

sudo systemctl enable nginx

# Start Nginx service

sudo systemctl start nginx

# Install stress (for testing purposes)

sudo apt install stress -y

# Navigate to the /tmp directory

cd /tmp

# Clone the repository from GitHub

git clone <https://github.com/Indranishetty11/cosmetics.git> cosmetics

# Copy the contents to the Nginx web directory

sudo cp -r /tmp/cosmetics/\* /var/www/html/

# Change ownership to www-data user and group (for Nginx)

sudo chown -R www-data:www-data /var/www/html/

# Set proper permissions for the web directory

sudo chmod -R 755 /var/www/html/

# Restart Nginx to apply changes

sudo systemctl restart nginx

**4. Verify Website Deployment**

After deployment, access the VM’s public IP address in a browser to ensure the website is live.

**5. Capture Image of the VM**

* Go to the VM in the Azure Portal.
* Select **Capture** to create a VM image.
* Enable the "Automatically delete the VM after creating the image" checkbox.
* Specify the **image gallery**, **image definition**, and version (e.g., 1.0.0).
* Confirm and create the image.

**6. Create a VMSS Using the Captured Image**

* Navigate to **Virtual Machine Scale Sets** in Azure Portal.
* Use the captured image as the base image for the VMSS.
* Configure the VMSS settings such as instance count, size, and networking.

**7. Verify the Image in VMSS**

Ensure the VMSS instances use the custom image by checking the configuration in the scale set overview.

**8. Configure Autoscaling and Load Balancer**

* Set up autoscaling rules:
  + Minimum instances: 1
  + Maximum instances: 2
  + Default instances: 1
  + Scale out: Increase instances by 25% if CPU usage exceeds the threshold.
  + Scale in: Decrease instances by 10% if CPU usage drops below the threshold.
* Create a **Load Balancer** in the networking tab to distribute traffic evenly across instances.

**9. Create the VMSS**

Deploy the VMSS with the configured settings and captured image.

**10. Verify Website Availability**

* Go to the **Instances** section of the VMSS.
* Copy the public IP address of an instance and open it in a browser to check if the website is live.
* Repeat for other instances to confirm availability.

**11. Configure Inbound Rule and Stress Test**

* In the VM’s network settings, add an inbound rule to allow all traffic (\*).
* Connect to a VM instance using PowerShell or SSH.
* Run a stress test with the command:
* stress --cpu 1 --vm 1 --io 1 --hdd 1 --timeout 5m

**12. Monitor Metrics and Autoscaling**

* Navigate to the **Metrics** section of the VMSS.
* View the **Percentage CPU** graph and set the time range to the last 30 minutes with 1-minute updates.
* Monitor the CPU usage. If the stress exceeds the threshold, confirm that an additional instance is automatically created.

**13. Verify High Availability**

* Access the website using the public IP addresses of the instances to confirm that it is accessible from multiple instances.
* This demonstrates the high availability and scalability of the website using VMSS.

**Services Used**

1. **Virtual Machine (VM):** A single compute instance for hosting the website.
2. **Azure Custom Script Extension:** Automatically configures the VM during deployment.
3. **Azure Image Gallery:** Stores reusable images for VM and VMSS deployment.
4. **Virtual Machine Scale Set (VMSS):** Provides scalability and high availability by managing multiple VMs.
5. **Azure Load Balancer:** Distributes incoming traffic across instances.
6. **Azure Autoscale:** Automatically adjusts the number of instances based on CPU usage.
7. **Azure Monitor Metrics:** Tracks and displays resource performance over time.

This workflow ensures a highly available, scalable website deployment using Azure VMSS.