**Task3**

00-providers.tf

01-backend-bucket.tf # remote state (GCS)

02-project-apis.tf # enable APIs

10-vpc.tf # single shared VPC (recommended)

11-subnets.tf # one subnet per region

12-router-nat.tf # Cloud Router + NAT for egress

20-firewalls.tf # common ICMP/SSH/HTTP + health-checks

30-lb.tf # global external HTTP(S) LB, hc, backend svc

31-lb-urlmap-frontends.tf # URL map, proxy, forwarding rule/IP

40-windows-vm.tf # Windows VM w/ tag: win-admin

members/

Billy.tf # Linux VM/MIG in region A (tag: linux-Billy)

Bob.tf # Linux VM/MIG in region B (tag: linux-bob)

John.tf # ...

Doe.tf # ...

Charles.tf # (Maryland/us-east1)

scripts/

startup-linux.sh # hardened Apache startup

startup-windows.ps1 # optional: enable ping/Win firewall

**2) Networking**

a. VPC: 1 shared VPC (simplifies LB + “everyone can ping everyone”).

b. Subnets (one per member/region):

c. Example: us-east1 (10.20.10.0/24), southamerica-east1, africa-south1, southamerica-west1, etc.

d. Cloud Router + Cloud NAT:

e. NAT set to ALL\_SUBNETWORKS\_ALL\_IP\_RANGES (or list all subnets) so Linux VMs behind the LB (no external IPs) can apt-get on first boot.

3**) Compute (per member)**

a. Instance Template (recommended for scalability) or single VM:

b. Debian 12, Apache startup script, tag named after the member (e.g., linux-john).

c. If using LB: no external IP for backends; rely on NAT for egress.

d. MIG (optional but aligns with “ensure scalability”):

e. 1+ size per region; define named\_port { name="http" port=80 }.

**4) Windows VM (management box)**

a. Place in any subnet; give tag: win-admin.

b. Optional public IP for RDP, or jump via IAP.

c. Windows OS firewall: enable ICMP Echo (ping) inbound; allow management tools as needed.

**5) Firewall Rules**

a. Google Health checks/GFE → backends (port 80):

b. source\_ranges: 35.191.0.0/16, 130.211.0.0/22

c. target\_tags: include all Linux backend tags (or a shared lb-backend tag)

d. Windows → all Linux VMs (Task-3 requirement):

e. source\_tags = ["win-admin"], target\_tags = ["linux-billy", "linux-bob", ...]

f. allow blocks for icmp, tcp ports you need (e.g., 22, 80)

g. Everyone can ping everyone:

h. Intra-VPC ICMP allow (ingress), e.g., source\_ranges = ["10.0.0.0/8"] or your exact CIDRs

**6) Load Balancer (global external HTTP(S))**

a. Health check (TCP or HTTP:80)

b. Backend service (port\_name="http") with each member’s MIG/IG:

c. Add one backend { group = <member MIG instance\_group> } per region

d. URL map, target proxy, forwarding rule, global IPv4 (and cert if HTTPS)

e. (If you must use 2 VPCs: use one VPC for LB + backends, or use PSC NEGs)

**7) DNS (optional )**

a. Point your domain (e.g., via DNSimple) to the LB’s global IP (A record).

b. create A record → Maryland static IP ( LB ).

**8) Startup Scripts**

a. Linux: noninteractive apt; ensure curl; minimal and reliable.

b. Windows (optional): PS script to open ICMP echo inbound.

c. Keep scripts in /scripts/ and reference with metadata\_startup\_script = file(...).

**9) Variables, Naming, Tags**

a. Consistent resource prefixes (e.g., subnet-<region>, mig-<member>).

b. Tags:

c. Per member: linux-<member> (satisfies firewall-tag requirement).

d. Shared: lb-backend for all backends, win-admin for Windows VM.

**10) Outputs (for validation)**

a. LB IPv4

b. Each member’s VM/MIG instance group URL

c. Subnet CIDRs per region

d. (Optional) DNS record IDs if managed via Terraform

**11) README.md (step-by-step)**

a. Prereqs + how to set secrets

b. terraform init/plan/apply

c. Ping tests between Windows and all Linux VMs