# Challenge 1:

A 3-tier environment is a common setup. Use a tool of your choosing/familiarity create these resources. Please remember we will not be judged on the outcome but more focusing on the approach, style and reproducibility.

# Approach

For the above challenges below setup is proposed by considering certain assumptions. I have targeted to setup environment using services available on Google Cloud.

* Google cloud is preferred as set-up as it provides highly reliable and scalable cloud platform offering the best cloud computing services on the web
* To provide high availability, application is hosted on two zones of single region of Google Cloud Platform (this approach is selected to consider cost), once application is used by multiple location user then same application can be hosted on different regions of Google Cloud Platform.
* DevOps pipeline should setup for infrastructure creation and deployment of applications to cloud
* Lift and Shift strategy can be applied directly to migrate these applications ( with few customizations)

# Assumption

Below points are consider while going for below set-up

1. System set up is done by consider high availability of system for end user
2. Security should be available across all 3 layers
3. No redundant data should present on system
4. Cost is also consider during setup approach
5. All infrastructure will be created using Terraform scripting (Infrastructure as a code) ( Terraform is selected by considering below advantage available for future use
   1. reusability
   2. multi cloud support
   3. modularity
6. Deployment manager service present at Google cloud platform can be reused for configuration management

# Setup-Diagram

**Abbreviations used**

**VPC**: Virtual Private Cloud

**L.B**: Load Balancer

**C.E**.: Compute Engine

**F.RULES**: Firewall Rules

**M DB**: Master Database

**R.R**: Read Replica

# GCP Resources Used and details

**VPC** : It provides on-premise data centre functionality in cloud. It creates a boundary between each customer’s and their environment.

**Http Load Balancer** : It is starting point, where user’s request hit

**Load Balancer** : Here Application Load balancer had proposed for load balancing for both Web-subnet and app-subnet area. Once request received from https load balancer, it redirected to to ALB present at web-subnet, it process request and send request to

**Compute Engine** : These compute engine provides (Infrastructure as a service), which is more compatible with existing software used in web layer of application and application layer ( business logic implemented) in business layer. Cost is also consider as a setup approach. Based on performance of compute engine, later on compute size and memory size can be increased

Managed instance group functionality is to be used for creation of cluster of compute engines on both web-subnet and app-subnet

**Firewall rules** : This functionality provides security at each layer . This is an out-of-box functionality present at GCP. This rule is applied between web-subnet layer to app-subnet layer and app-subnet layer to db-subnet layer.

**Master DB** : Here cloud sql database had selected for db layer to store all relational database. In this setup approach one master db has selected for write operation, corresponding read-replica database servers are created to improve database’s performance. All read-replicas are synchronies data from master database. Cache layer can be introduced on top-of database layer to improve read functionality from database.

Google Cloud’s existing encryption is used to data present on cloud sql.

**Cloud Storage** : This service helps to store hosted application data ( static web page , logs and terraform’s state file)

**Operations:** GCP provided service can be used forlog and infrastructure monitoring