**Challenge1:**

**Resources required:**

VM Instance, NAT Gateway, Code fetching and integration with VM, Load Balancer setup, Cloud SQL Integration, Firewall configuration, Testing.

Step 1: Create a VM instance with default Service Account, “Allow http” enabled and with no external IP address.

Step 2: Create NAT configuration with router, if VM needs to access/ get updates from internet as we have disabled external IP address. This is optional.

Step 3: We assume that we have our python application created which would allow user to input their name, address and mobile number. Now, we retrieve the application using git clone.

Step 4: Install python dependencies like pip, upgrade if required. And runtime dependencies using sudo apt-get update and sudo apt-get install if needed to install anything missing.

Step 5: Create a managed/unmanaged instance group. As this for test and single VM, unmanaged should be fine.

Step 6: Create HTTPS load balancer with backend as the instance group. This gives a front-end IP.

Step 7: For database, we can use Cloud SQL here. Create MySQL server instance -> Configure -> Add network with the load balancer front end IP, create and add users.

Step 8: Create a firewall rule to allow Cloud SQL to connect to port 3306. Source would be the front end IP.

Step 9: Provide the MySQL server details to the python code including server IP.

Step 10: Connect to the SQL server from python server VM to check database connectivity.

Step 11: Open the application from Load balancer IP and test by entering details and click save. This should update the data in the MySQL database.

This completes a three tier architecture of Frontend ( LB) -> Application -> Database ( MySQL)

**Challenge 2:**

Following command should retrieve VM metadata in json format. Also, to note default metadata retrieval itself is json format.

 gcloud compute instances describe vm-name --flatten="metadata[]" --format=json

flatten for presenting data in readable way.

The above command retrieves any keys available at instance level for the instance.

To retrieve specific key,

gcloud compute instances describe vm-name  --format='value(metadata.items.key-name)'

**Challenge 3:**

We can use javascript here:

function customFilter(object){

    if(object.hasOwnProperty(key))

        return object;

    for(var i=0; i<Object.keys(object).length; i++){

        if(typeof object[Object.keys(object)[i]] == "object"){

            var o = customFilter(object[Object.keys(object)[i]]);

            if(Object.keys(object)[i] == key)

                return value;

        }

    }

    return null;

}