KPMG Challenge –

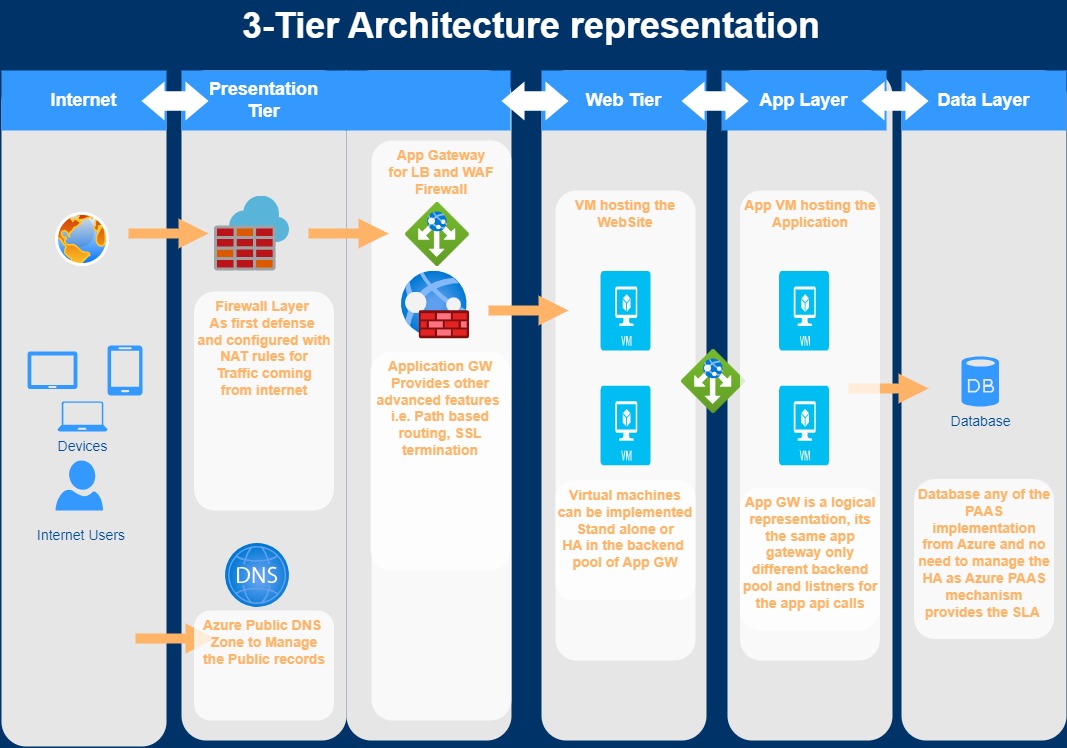
Challenge #1

A 3-tier environment is a common setup. Use a tool of your choosing/familiarity create these

resources in a cloud environment (Azure/AWS/GCP).

Ans: - Web-Apps can be implemented in several ways, below is Sample implementation of a 3- tier architecture.

In the below, the Web and Apps deployed using simple IAAS services and for the Data layer used a PAAS implementation of Data base.



Challenge #2

We need to write code that will query the meta data of an instance within AWS or Azure or GCP

and provide a json formatted output.

Ans: - In Azure Cloud the meta service end points can be accessed with in the virtual machines and below python code can fetch the information in JSON formatted output.

“”

Terraform output –json will be providing the Virtual machine meta data into json format.

**i.e,. Python SDK’s**

import requests

import json

metadata\_url = 'http://169.254.169.254/metadata/instance?api-version=2021-02-01'

headers = {

    'Metadata': 'true'

}

response = requests.get(metadata\_url, headers=headers)

metadata\_json = response.json()

formatted\_output = json.dumps(metadata\_json, indent=4)

print(formatted\_output)

Challenge #3

We have a nested object. We would like a function where you pass in the object and a key and

get back the value.

* Example Inputs

object = {“a”:{“b”:{“c”:”d”}}}

key = a/b/c

In the below program takes the Key’s in the combination of the nested architecture and provides the output.

def get\_value\_from\_nested\_object(obj, key):

    keys = key.split('/')

    value = obj

    try:

        for k in keys:

            value = value.get(k)

            if value is None:

                return None

        return value

    except (AttributeError, TypeError):

        return None

obj = {"a": {"b": {"c": "d"}}}

key = "a/b/c"

value = get\_value\_from\_nested\_object(obj, key)

print(value)

In case if the value is required with the Specific Key in the nested structure, then

def getvaluefromobj(obj, key):

    if isinstance(obj, dict):

        if key in obj:

            return obj[key]

        else:

            for value in obj.values():

                result = getvaluefromobj(value,key)

                if result is not None:

                    return result

    return None

obj = {"a": {"b": {"c": "d"}}}

key = "d"

value = getvaluefromobj(obj, key)

print(value)