Hi Sindhu,

Thank you for making an application to our DevOps team.

We understand your time is precious, but as discussed we ask each applicant to take on our small challenges. There is no right or wrong approach and we're certainly not expecting war and peace, We would expect no more than 30 minutes per challenge, if at the end of the time you haven’t completed don’t worry just send us what you have. (Upload to a GIT repository)

## 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.

apiVersion: extensions/v1beta1

kind: Deployment

metadata:

name: MySql-primary -- appname

namespace: k8s-workshop

spec:

replicas: 3

template:

metadata:

labels:

app: apche

role: primary

tier: backend

spec:

containers:

- name: MySQL

image: gcr.io/google\_containers/MySql:e2e # or just image: MySql

resources:

requests:

cpu: 100m

memory: 100Mi

ports:

- containerPort: 3386

name:web-port

spec:

auth:

username: MySql

password: luM\*yL9PAC\*T

## Challenge #2

### Summary

We need to write code that will query the meta data of an instance within aws and provide a json formatted output. The choice of language and implementation is up to you.

### Bonus Points

The code allows for a particular data key to be retrieved individually

### Hints

* [Aws Documentation](http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instance-metadata.html)
* [Azure Documentation](https://docs.microsoft.com/en-us/azure/virtual-machines/windows/instance-metadata-service)
* [Google Documentation](https://cloud.google.com/compute/docs/storing-retrieving-metadata)

# setup Go

wget -p /tmp http://go.googlecode.com/files/gol.0.3.linux-adm64.tar.gz

sudo tar -C /usr/local -xzf /tmp/gol.0.3.linux-adm64.tar.gz

export PATH=$PATH:/usr/local/go/bin

#Collect the source files

wget https://github.com/namailla-xyz/meta-data-query-to-json/blob/main/metadata.go

#print metadata

go run metadata.go

{"ami-id":"ami-aa941e9a","ami-launch-index":"0","block-device-mapping":{"ami":"sda1","ephemeral0"}

#pretty print meta data

go run metadata.go --prettyprint

{

"ami-id": "ami-aa941e9a",

"ami-launch-index":"0",

"block-device-mapping": {

"ami":"sda1",

"ephemeral0" : "sda2",

"root":"dev/sda1",

"swap":"sda3"

}

}

## Challenge #3

We have a nested object, we would like a function that you pass in the object and a key and get back the value. How this is implemented is up to you.

### Example Inputs

object = {“a”:{“b”:{“c”:”d”}}}

key = a/b/c

object = {“x”:{“y”:{“z”:”a”}}}

key = x/y/z

value = a

### Hints

We would like to see some tests. [A quick read to help you along the way](https://hexdocs.pm/elixir/master/Kernel.html#get_in/2)

We would expect it in any other language apart from elixir.

Again thank you for spending your precious time on our challenge.

**# Python**

class obj(object):

def \_\_init\_\_(self, d):

for a, b in d.items():

if isinstance(b, (list, tuple)):

setattr(self, a, [obj(x) if isinstance(x, dict) else x for x in b])

else:

setattr(self, a, obj(b) if isinstance(b, dict) else b)

d = {'a': 1, 'b': {'c': 2}, 'd': ["hi", {'ITCOMPANY': "KPMG"}]}

x = obj(d)

x.d[1].ITCOMPANY