Kubernetes Day 2 Concluding Exercise

The goal of this exercise is to continue developing our cluster by adding persistency and dynamic configuration to our echo app.

* Create a new private docker repository in GCP

* Clone the repository for Echo app, modify it and add 'info' level logging to
  show the process's hostname. Build a docker image from your local copy of the code.

* Push the image you created into a new GCR repo you created, and configure your
  deployment to pull this new image.

* Create a mongoDB Stateful set with 3 replicas to serve our app. You can use
  the mongoDB sidecar image to configure the replicaset for you.

* Create a ConfigMap for our app to be used in place of
  `config/default.yaml`. The required keys can be found in the original
  application code.

* You can override the container's default config using either environment
  variables or by mounting the `default.yaml` file into the config directory.
  -
  Note: If you choose to mount the config file - check out the Dockerfile to find
  out where the configmap should be mounted in the deployment. Notice that
  mounting configMaps or Secrets mount a volume - not a file!

* Scale our echo app's deployment to 2 replicas.

* Configure the app to use the 'info' log level so we don't see all that pesky
  debug info.

* Configure the app to persist to our newly installed MongoDB statefulSet.

* Verify everything works!

To get mongodb to join a replicaset:

exec into mongo-0:

```
$ kubectl exec -ti mongo-0 mongo
then run:
> rs.initiate()
```

```
> rs.add({ host: "mongo-1.mongo" })
> rs.add({ host: "mongo-2.mongo" })

$ kubectl exec -ti mongo-1 mongo
then run:
> rs.add({ host: "mongo-0.mongo" })
> rs.add({ host: "mongo-2.mongo" })

$ kubectl exec -ti mongo-2 mongo
then run:
> rs.add({ host: "mongo-0.mongo" })
> rs.add({ host: "mongo-1.mongo" })
```