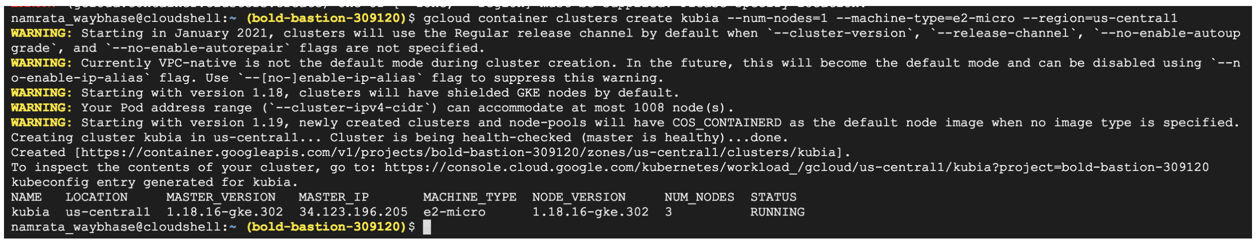
CS571 Signature Project
Name: Namrata Waybhase
ID: 19551

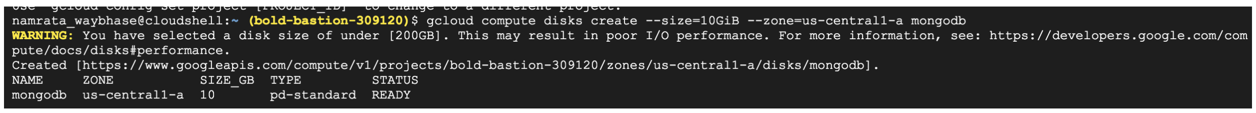
Step1 Create MongoDB using Persistent Volume on GKE, and insert records into it

1. Create a cluster as usual on GKE gcloud container clusters create kubia --num-nodes=1 --machine-type=e2-micro —region=us-central1



```
namrata_waybhase@cloudshell:~ (bold-bastion-309120)$ gcloud container clusters create kubia --num-nodes=1 --machine-type=e2-micro --region=us-central1
WARNING: Starting in January 2021, clusters will use the Regular release channel by default when `--cluster-version`, `--release-channel`, `--no-enable-autoup
grade`, and `--no-enable-autorepair` flags are not specified.
WARNING: Currently VPC-native is not the default mode during cluster creation. In the future, this will become the default mode and can be disabled using `--n
o-enable-ip-alias` flag. Use `--[no-]enable-ip-alias` flag to suppress this warning.
WARNING: Starting with version 1.18, clusters will have shielded GKE nodes by default.
WARNING: Your Pod address range (`--cluster-ipv4-cidr`) can accommodate at most 1008 node(s).
WARNING: Starting with version 1.19, newly created clusters and node-pools will have COS_CONTAINERD as the default node image when no image type is specified.
Creating cluster kubia in us-central1... Cluster is being health-checked (master is healthy)...done.
Created [https://container.googleapis.com/v1/projects/bold-bastion-309120/zones/us-central1/clusters/kubia].
To inspect the contents of your cluster, go to: https://console.cloud.google.com/kubernetes/workload_/gcloud/us-central1/kubia?project=bold-bastion-309120
kubeconfig entry generated for kubia.
NAME   LOCATION     MASTER_VERSION   MASTER_IP       MACHINE_TYPE  NODE_VERSION     NUM_NODES  STATUS
kubia  us-central1  1.18.16-gke.302  34.123.196.205  e2-micro      1.18.16-gke.302  3          RUNNING
namrata_waybhase@cloudshell:~ (bold-bastion-309120)$
```

2. Create a Persistent Volume first
gcloud compute disks create --size=10GiB --zone=us-central1-a mongodb

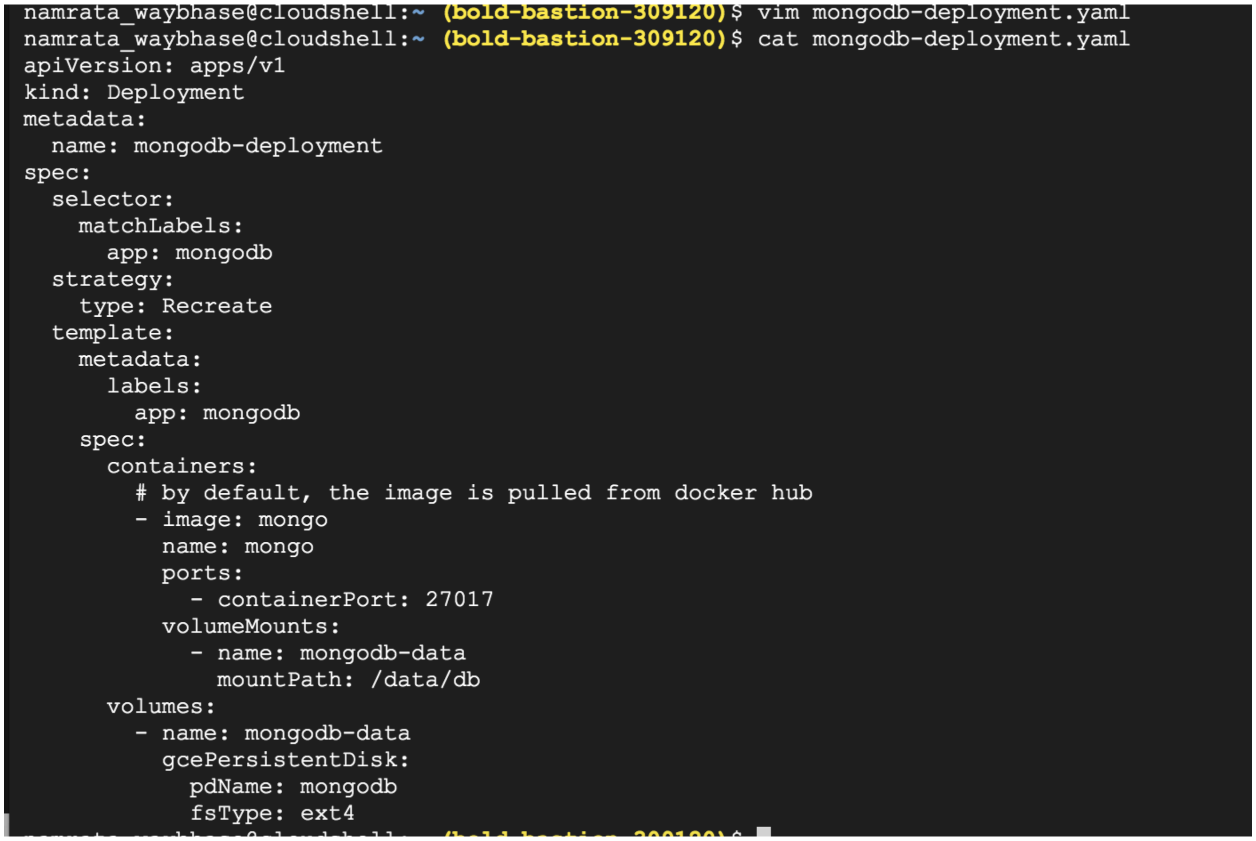


```
namrata_waybhase@cloudshell:~ (bold-bastion-309120)$ gcloud compute disks create --size=10GiB --zone=us-central1-a mongodb
WARNING: You have selected a disk size of under [200GB]. This may result in poor I/O performance. For more information, see: https://developers.google.com/com
pute/docs/disks#performance.
Created [https://www.googleapis.com/compute/v1/projects/bold-bastion-309120/zones/us-central1-a/disks/mongodb].
NAME     ZONE           SIZE_GB  TYPE         STATUS
mongodb  us-central1-a  10       pd-standard  READY
```

3. Now create a mongodb deployment with this yaml filec

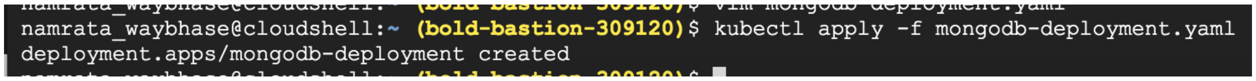
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mongodb-deployment
spec:
  selector:
    matchLabels:
      app: mongodb
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: mongodb
    spec:
      containers:
        # by default, the image is pulled from docker hub
        - image: mongo
          name: mongo
          ports:
            - containerPort: 27017
          volumeMounts:
            - name: mongodb-data
              mountPath: /data/db
```

```
volumes:
  - name: mongodb-data
    gcePersistentDisk:
      pdName: mongodb
      fsType: ext4
```



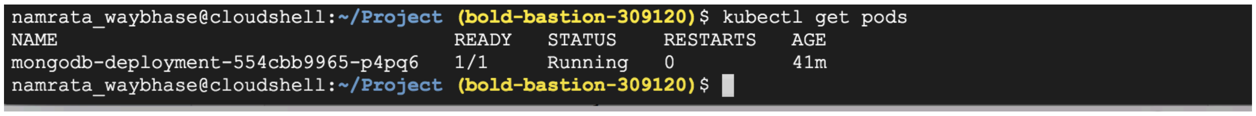
```
namrata_waybhase@cloudshell:~ (bold-bastion-309120)$ vim mongodb-deployment.yaml
namrata_waybhase@cloudshell:~ (bold-bastion-309120)$ cat mongodb-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mongodb-deployment
spec:
  selector:
    matchLabels:
      app: mongodb
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: mongodb
    spec:
      containers:
        # by default, the image is pulled from docker hub
        - image: mongo
          name: mongo
          ports:
            - containerPort: 27017
          volumeMounts:
            - name: mongodb-data
              mountPath: /data/db
      volumes:
        - name: mongodb-data
          gcePersistentDisk:
            pdName: mongodb
            fsType: ext4
```

$ kubectl apply -f mongodb-deployment.yaml



```
namrata_waybhase@cloudshell:~ (bold-bastion-309120)$ kubectl apply -f mongodb-deployment.yaml
deployment.apps/mongodb-deployment created
```

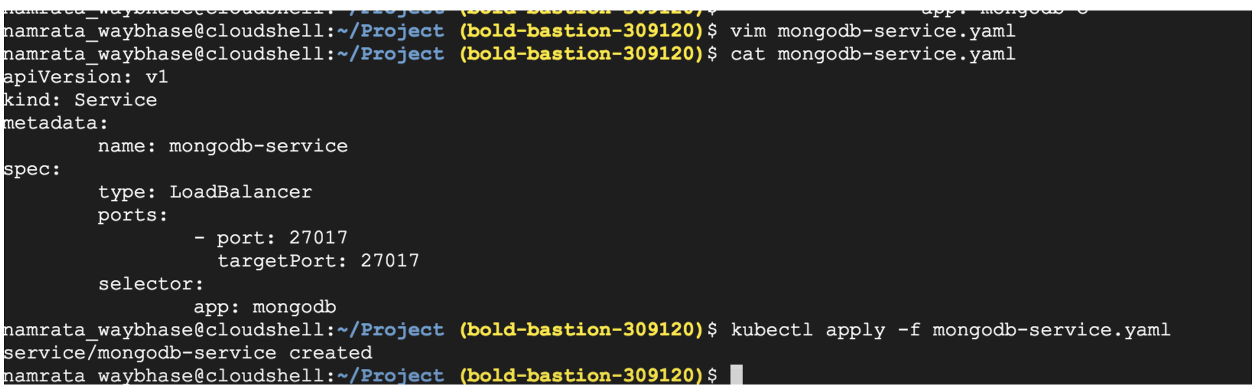
4. Check if the deployment pod
$ kubectl get pods



```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ kubectl get pods
NAME                                  READY   STATUS    RESTARTS   AGE
mongodb-deployment-554cbb9965-p4pq6   1/1     Running   0          41m
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$
```

5. Create a service for the mongoDB, so it can be accessed from outside

$ kubectl apply -f mongodb-service.yaml

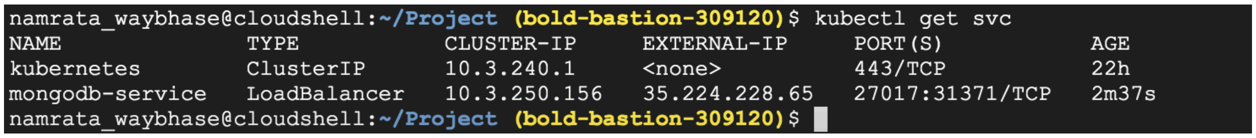


```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ vim mongodb-service.yaml
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ cat mongodb-service.yaml
apiVersion: v1
kind: Service
metadata:
        name: mongodb-service
spec:
        type: LoadBalancer
        ports:
                - port: 27017
                  targetPort: 27017
        selector:
                app: mongodb
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ kubectl apply -f mongodb-service.yaml
service/mongodb-service created
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$
```

6. Check if the service is up, using command

$ kubectl get svc

Wait until external ip is generated



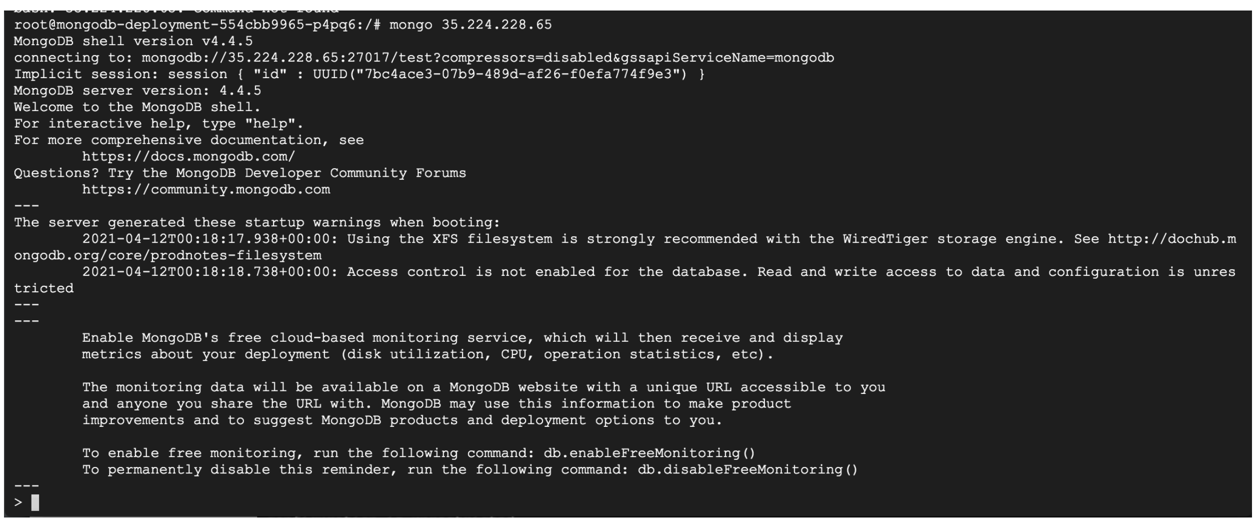
```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ kubectl get svc
NAME              TYPE           CLUSTER-IP     EXTERNAL-IP     PORT(S)           AGE
kubernetes        ClusterIP      10.3.240.1     <none>          443/TCP           22h
mongodb-service   LoadBalancer   10.3.250.156   35.224.228.65   27017:31371/TCP   2m37s
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$
```

7. Now try and see if mongoDB is functioning using external ip

$ kubectl exec -it mongodb-deployment--replace-with-your-pod-name -- bash

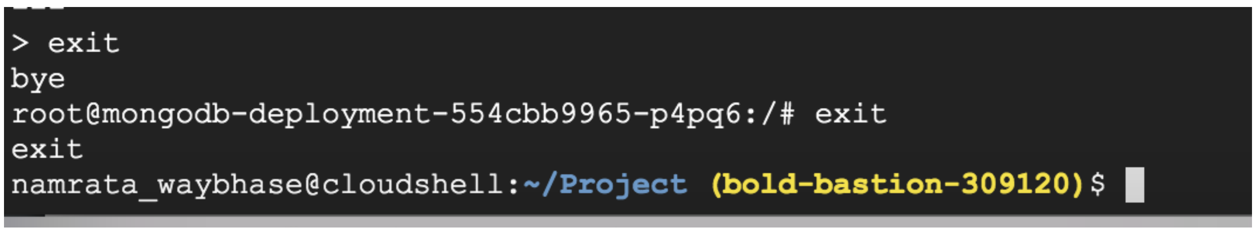
type :

mongo External-IP



```
root@mongodb-deployment-554cbb9965-p4pq6:/# mongo 35.224.228.65
MongoDB shell version v4.4.5
connecting to: mongodb://35.224.228.65:27017/test?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("7bc4ace3-07b9-489d-af26-f0efa774f9e3") }
MongoDB server version: 4.4.5
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
        https://docs.mongodb.com/
Questions? Try the MongoDB Developer Community Forums
        https://community.mongodb.com
---
The server generated these startup warnings when booting:
        2021-04-12T00:18:17.938+00:00: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.m
ongodb.org/core/prodnotes-filesystem
        2021-04-12T00:18:18.738+00:00: Access control is not enabled for the database. Read and write access to data and configuration is unres
tricted
---
---
        Enable MongoDB's free cloud-based monitoring service, which will then receive and display
        metrics about your deployment (disk utilization, CPU, operation statistics, etc).

        The monitoring data will be available on a MongoDB website with a unique URL accessible to you
        and anyone you share the URL with. MongoDB may use this information to make product
        improvements and to suggest MongoDB products and deployment options to you.

        To enable free monitoring, run the following command: db.enableFreeMonitoring()
        To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
>
```

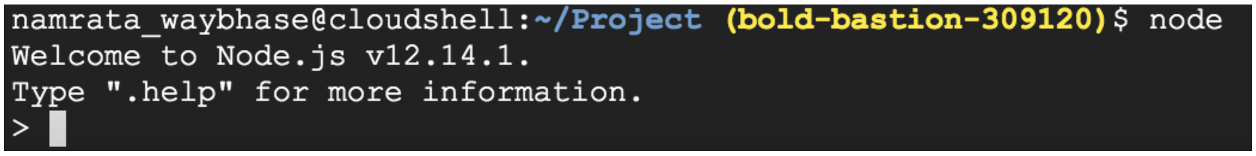
8. Type exit to go back to console


```
> exit
bye
root@mongodb-deployment-554cbb9965-p4pq6:/# exit
exit
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$
```

9. Insert some records into the mongoDB for later use

type:

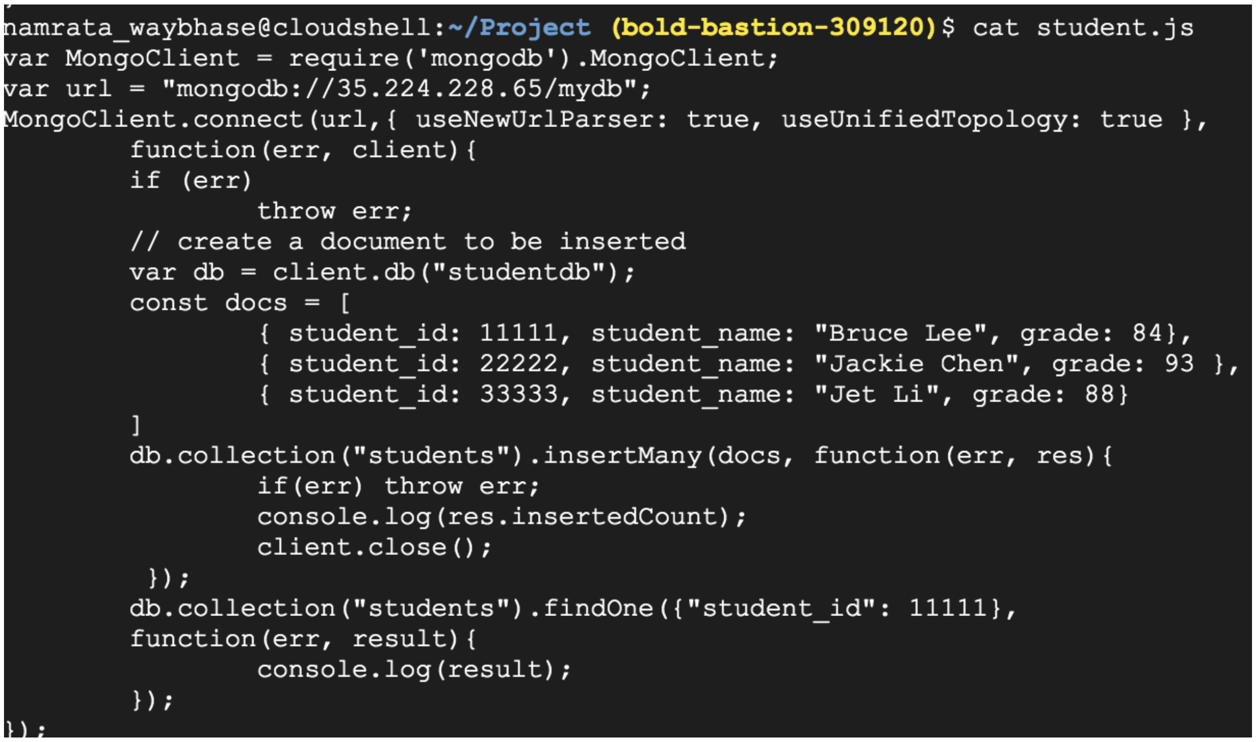
```
node
```


```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ node
Welcome to Node.js v12.14.1.
Type ".help" for more information.
>
```

10. Create a new file name student.js insert following code

$vi student.js

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://EXTERNAL-IP/mydb";
MongoClient.connect(url,{ useNewUrlParser: true, useUnifiedTopology: true }, function(err,
client){
        if (err)
                throw err;
        // create a document to be inserted
        var db = client.db("studentdb");
        const docs = [
                { student_id: 11111, student_name: "Bruce Lee", grade: 84},
                { student_id: 22222, student_name: "Jackie Chen", grade: 93 },
                { student_id: 33333, student_name: "Jet Li", grade: 88}
        ]
        db.collection("students").insertMany(docs, function(err, res){
                if(err) throw err;
                console.log(res.insertedCount);
                client.close();
         });
        db.collection("students").findOne({"student_id": 11111},
        function(err, result){
                console.log(result);
        });
});
```



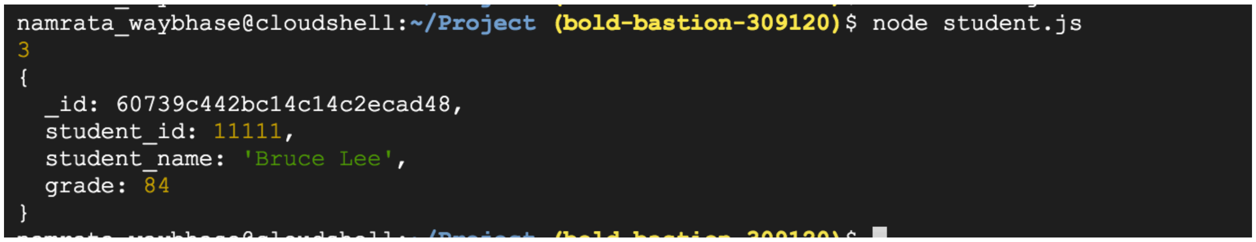
```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ cat student.js
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://35.224.228.65/mydb";
MongoClient.connect(url,{ useNewUrlParser: true, useUnifiedTopology: true },
        function(err, client){
        if (err)
                throw err;
        // create a document to be inserted
        var db = client.db("studentdb");
        const docs = [
                { student_id: 11111, student_name: "Bruce Lee", grade: 84},
                { student_id: 22222, student_name: "Jackie Chen", grade: 93 },
                { student_id: 33333, student_name: "Jet Li", grade: 88}
        ]
        db.collection("students").insertMany(docs, function(err, res){
                if(err) throw err;
                console.log(res.insertedCount);
                client.close();
         });
        db.collection("students").findOne({"student_id": 11111},
        function(err, result){
                console.log(result);
        });
});
```

11. make sure mongodb is install if not

   $ npm install mongodb

Run student.js file

   $ node student.js

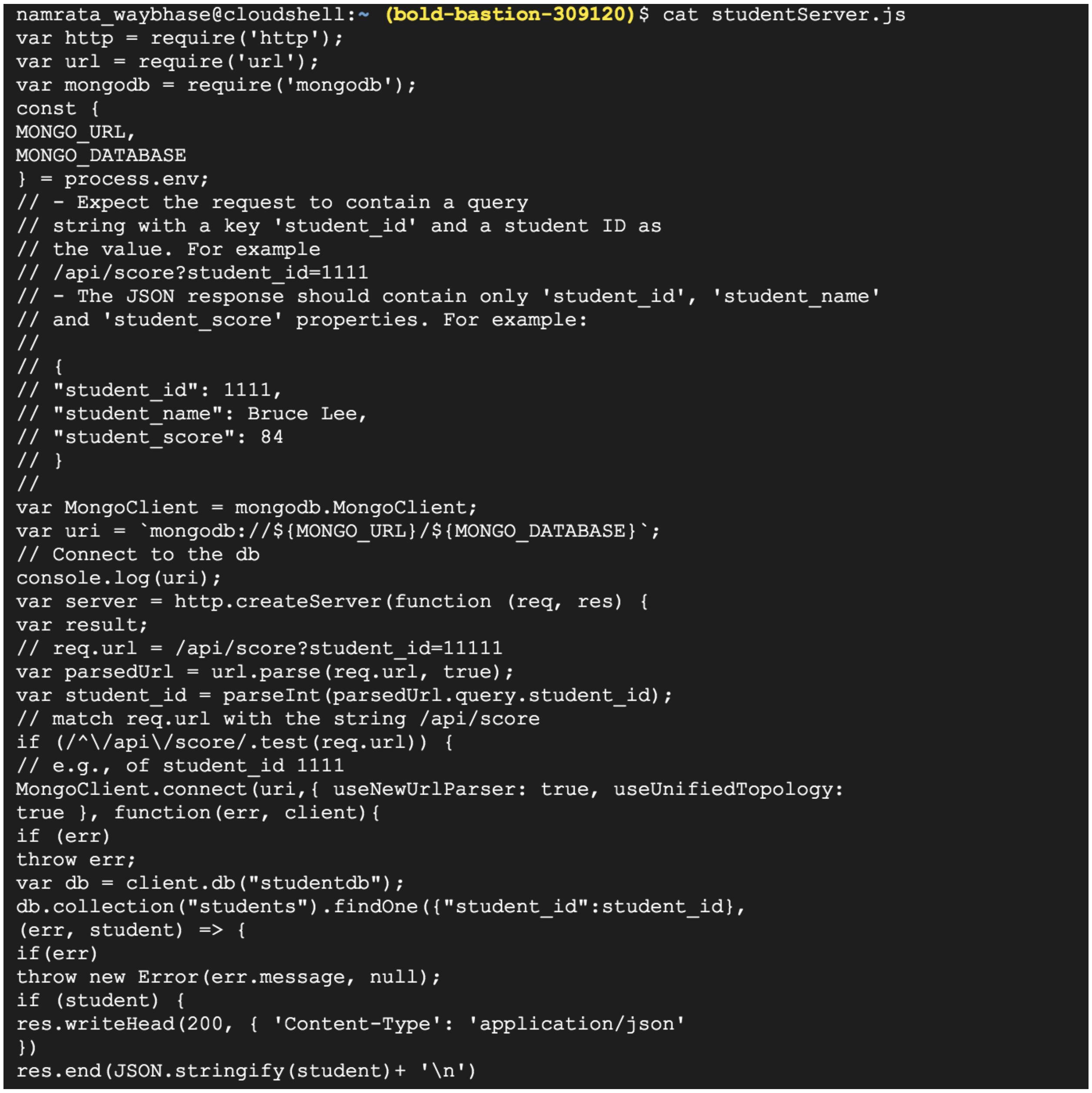


```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ node student.js
3
{
  _id: 60739c442bc14c14c2ecad48,
  student_id: 11111,
  student_name: 'Bruce Lee',
  grade: 84
}
```

Step2: Modify our studentServer to get records from MongoDB and deploy to GKE

1. Create a studentServer

$ vi studentServer.js

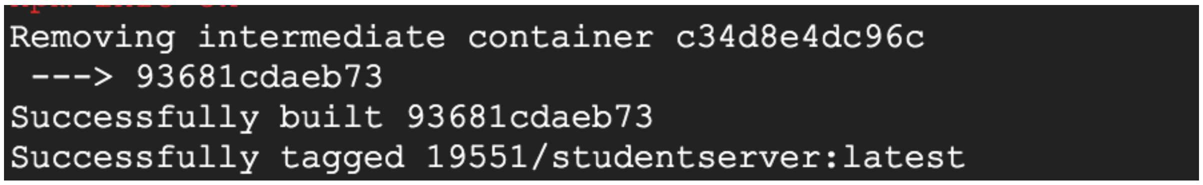


```
namrata_waybhase@cloudshell:~ (bold-bastion-309120)$ cat studentServer.js
var http = require('http');
var url = require('url');
var mongodb = require('mongodb');
const {
MONGO_URL,
MONGO_DATABASE
} = process.env;
// - Expect the request to contain a query
// string with a key 'student_id' and a student ID as
// the value. For example
// /api/score?student_id=1111
// - The JSON response should contain only 'student_id', 'student_name'
// and 'student_score' properties. For example:
//
// {
// "student_id": 1111,
// "student_name": Bruce Lee,
// "student_score": 84
// }
//
var MongoClient = mongodb.MongoClient;
var uri = `mongodb://${MONGO_URL}/${MONGO_DATABASE}`;
// Connect to the db
console.log(uri);
var server = http.createServer(function (req, res) {
var result;
// req.url = /api/score?student_id=11111
var parsedUrl = url.parse(req.url, true);
var student_id = parseInt(parsedUrl.query.student_id);
// match req.url with the string /api/score
if (/^\/api\/score/.test(req.url)) {
// e.g., of student_id 1111
MongoClient.connect(uri,{ useNewUrlParser: true, useUnifiedTopology:
true }, function(err, client){
if (err)
throw err;
var db = client.db("studentdb");
db.collection("students").findOne({"student_id":student_id},
(err, student) => {
if(err)
throw new Error(err.message, null);
if (student) {
res.writeHead(200, { 'Content-Type': 'application/json'
})
res.end(JSON.stringify(student)+ '\n')
```

2. Create Dockerfile
$ vi Dockerfile

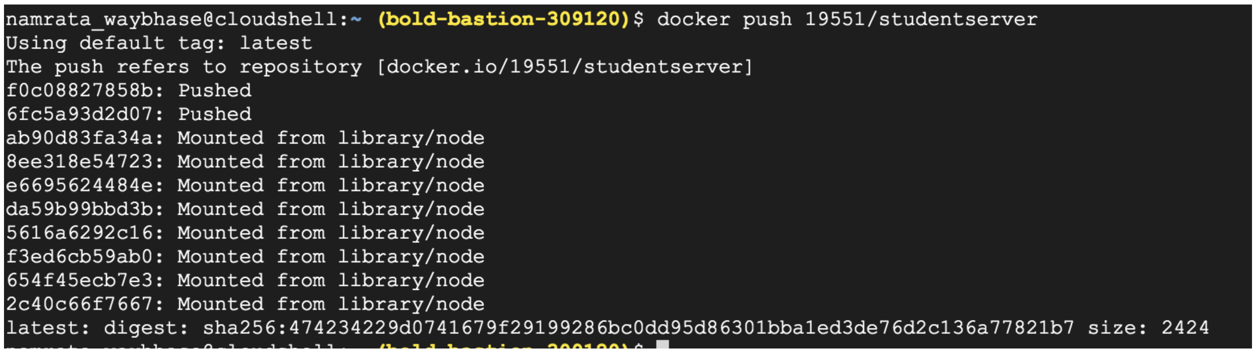
FROM node:7
ADD studentServer.js /studentServer.js
ENTRYPOINT ["node", "studentServer.js"]
RUN npm install mongodb

3. Build the studentserver
$ docker image docker build -t yourdockerhubID/studentserver .



```
Removing intermediate container c34d8e4dc96c
 ---> 93681cdaeb73
Successfully built 93681cdaeb73
Successfully tagged 19551/studentserver:latest
```

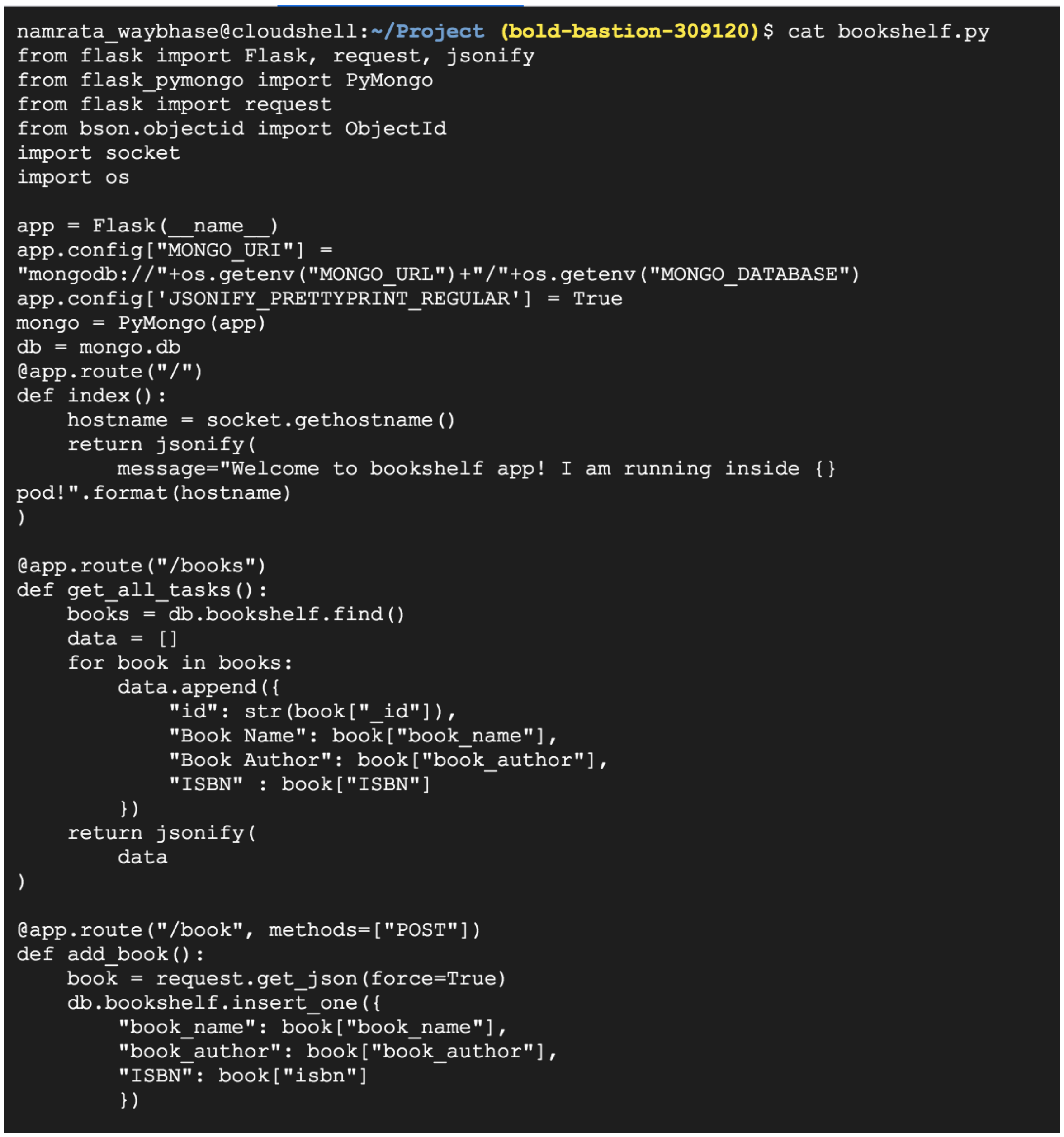
4. Push the docker image
$ docker push yourdockerhubID/studentserver



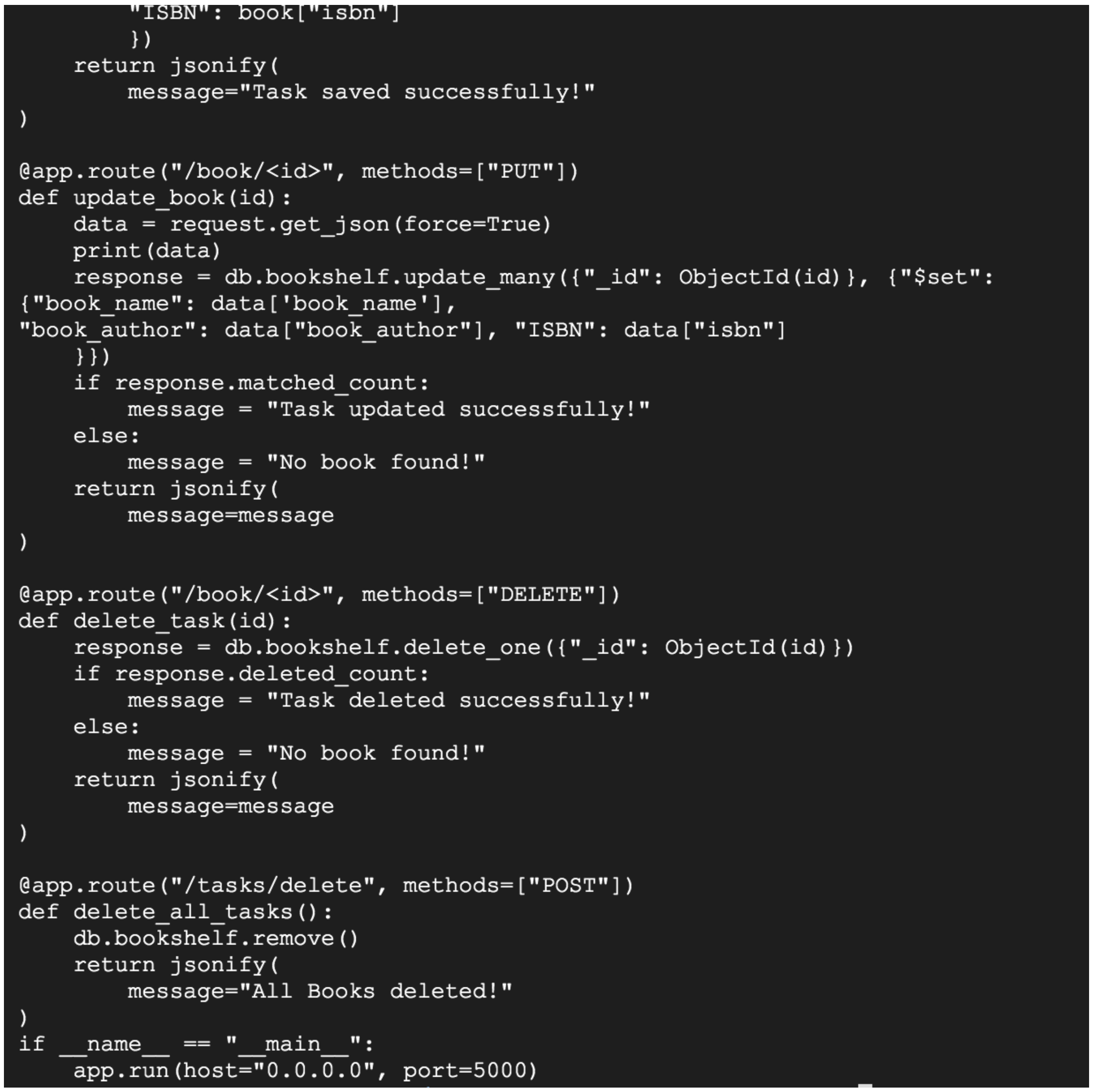
```
namrata_waybhase@cloudshell:~ (bold-bastion-309120)$ docker push 19551/studentserver
Using default tag: latest
The push refers to repository [docker.io/19551/studentserver]
f0c08827858b: Pushed
6fc5a93d2d07: Pushed
ab90d83fa34a: Mounted from library/node
8ee318e54723: Mounted from library/node
e6695624484e: Mounted from library/node
da59b99bbd3b: Mounted from library/node
5616a6292c16: Mounted from library/node
f3ed6cb59ab0: Mounted from library/node
654f45ecb7e3: Mounted from library/node
2c40c66f7667: Mounted from library/node
latest: digest: sha256:474234229d0741679f29199286bc0dd95d86301bba1ed3de76d2c136a77821b7 size: 2424
```

Step3 Create a python Flask bookshelf REST API and deploy on GKE

1. Create bookshelf.py



```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ cat bookshelf.py
from flask import Flask, request, jsonify
from flask_pymongo import PyMongo
from flask import request
from bson.objectid import ObjectId
import socket
import os

app = Flask(__name__)
app.config["MONGO_URI"] =
"mongodb://"+os.getenv("MONGO_URL")+"/"+os.getenv("MONGO_DATABASE")
app.config['JSONIFY_PRETTYPRINT_REGULAR'] = True
mongo = PyMongo(app)
db = mongo.db
@app.route("/")
def index():
    hostname = socket.gethostname()
    return jsonify(
        message="Welcome to bookshelf app! I am running inside {}
pod!".format(hostname)
)

@app.route("/books")
def get_all_tasks():
    books = db.bookshelf.find()
    data = []
    for book in books:
        data.append({
            "id": str(book["_id"]),
            "Book Name": book["book_name"],
            "Book Author": book["book_author"],
            "ISBN" : book["ISBN"]
        })
    return jsonify(
        data
)

@app.route("/book", methods=["POST"])
def add_book():
    book = request.get_json(force=True)
    db.bookshelf.insert_one({
        "book_name": book["book_name"],
        "book_author": book["book_author"],
        "ISBN": book["isbn"]
        })
```



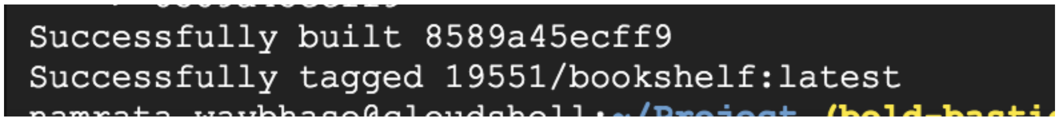
```
        "ISBN": book["isbn"]
        })
    return jsonify(
        message="Task saved successfully!"
)

@app.route("/book/<id>", methods=["PUT"])
def update_book(id):
    data = request.get_json(force=True)
    print(data)
    response = db.bookshelf.update_many({"_id": ObjectId(id)}, {"$set":
{"book_name": data['book_name'],
"book_author": data["book_author"], "ISBN": data["isbn"]
    }})
    if response.matched_count:
        message = "Task updated successfully!"
    else:
        message = "No book found!"
    return jsonify(
        message=message
)

@app.route("/book/<id>", methods=["DELETE"])
def delete_task(id):
    response = db.bookshelf.delete_one({"_id": ObjectId(id)})
    if response.deleted_count:
        message = "Task deleted successfully!"
    else:
        message = "No book found!"
    return jsonify(
        message=message
)

@app.route("/tasks/delete", methods=["POST"])
def delete_all_tasks():
    db.bookshelf.remove()
    return jsonify(
        message="All Books deleted!"
)
if __name__ == "__main__":
    app.run(host="0.0.0.0", port=5000)
```

2. Create a Dockerfile

```
FROM python:alpine3.7
COPY . /app WORKDIR /app
RUN pip install -r requirements.txt
ENV PORT 5000
EXPOSE 5000
ENTRYPOINT [ "python3" ]
CMD [ "bookshelf.py" ]
```

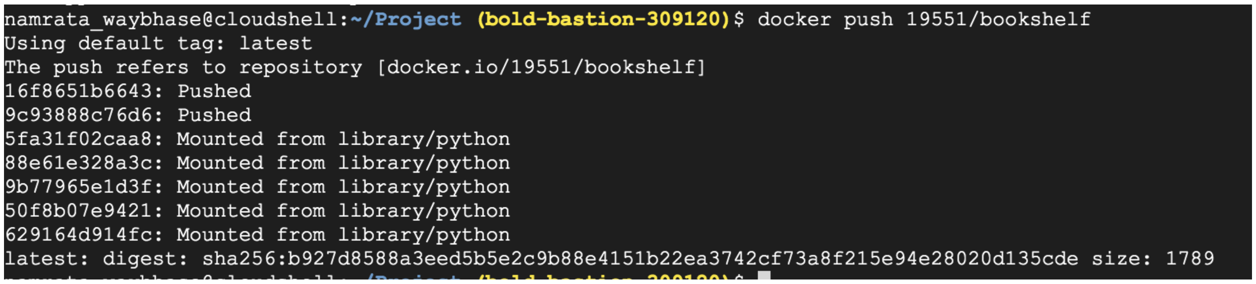
3. Build the bookshelf app into a docker image

$ docker build -t 19551/bookshelf .



```
Successfully built 8589a45ecff9
Successfully tagged 19551/bookshelf:latest
```

4. Push the docker image to your dockerhub
$ docker push 19551/bookshelf



```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ docker push 19551/bookshelf
Using default tag: latest
The push refers to repository [docker.io/19551/bookshelf]
16f8651b6643: Pushed
9c93888c76d6: Pushed
5fa31f02caa8: Mounted from library/python
88e61e328a3c: Mounted from library/python
9b77965e1d3f: Mounted from library/python
50f8b07e9421: Mounted from library/python
629164d914fc: Mounted from library/python
latest: digest: sha256:b927d8588a3eed5b5e2c9b88e4151b22ea3742cf73a8f215e94e28020d135cde size: 1789
```

Step4 Create ConfigMap for both applications to store MongoDB URL and MongoDB name

1. Create a file named studentserver-configmap.yaml

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: studentserver-config
data:
  MONGO_URL: 35.224.228.65
  MONGO_DATABASE: mydb
```

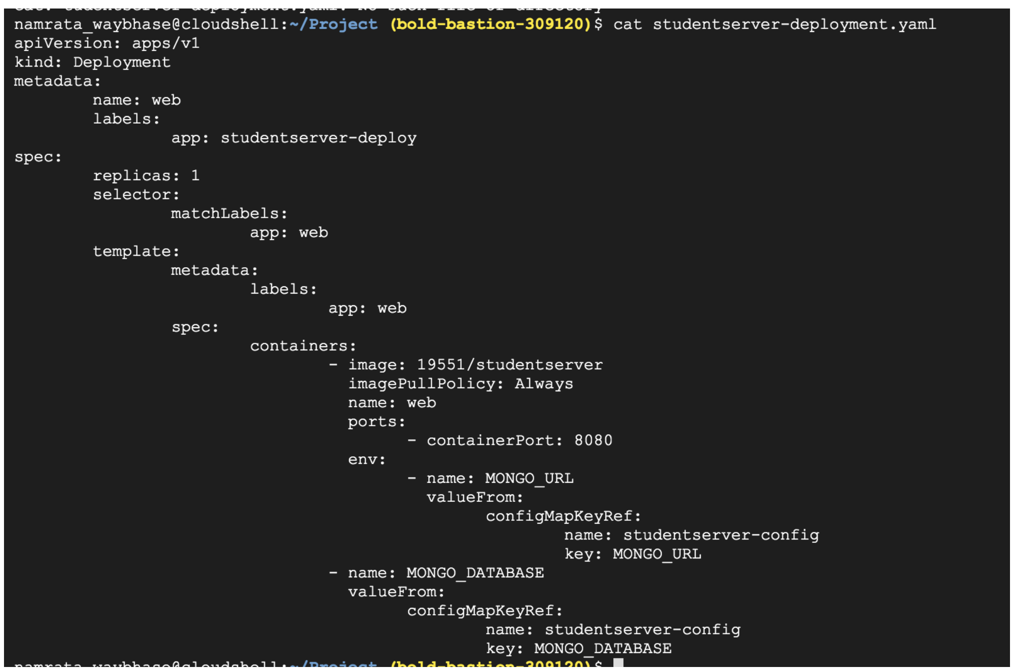
2. Create a file named bookshelf-configmap.yaml

```
apiVersion: v1
kind: ConfigMap
metadata:
    name: bookshelf-config
data:
    # SERVICE_NAME.NAMESPACE.svc.cluster.local:SERVICE_PORT
    MONGO_URL: 35.224.228.65
    MONGO_DATABASE: mydb
```

Step5 Expose 2 application using ingress with Nginx, so we can put them on the same Domain but different PATH

1. Create studentserver-deployment.yaml

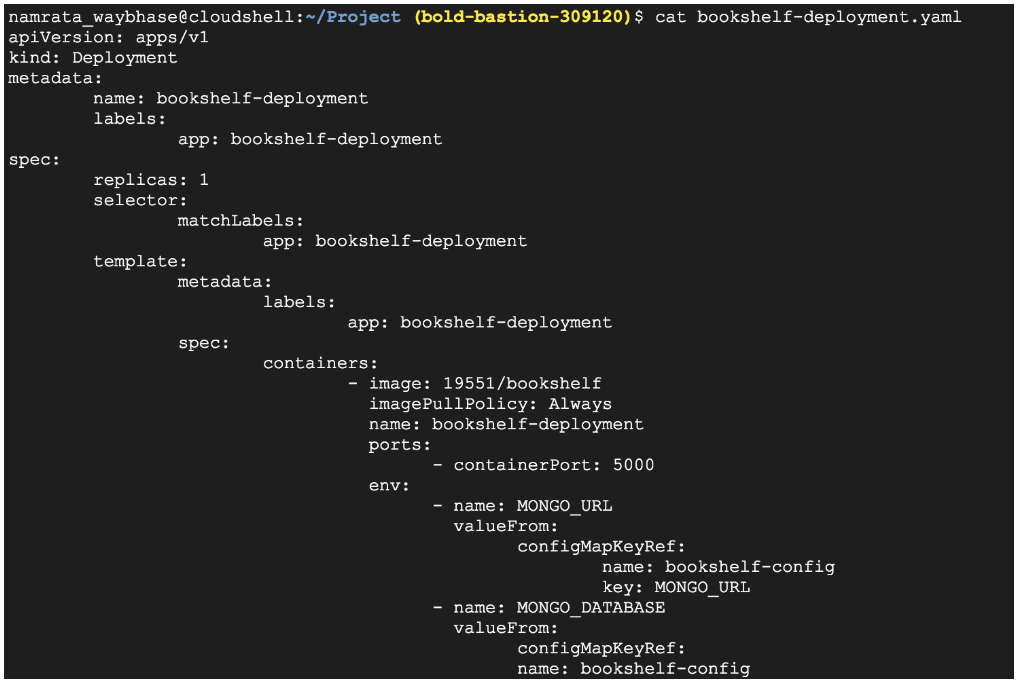
   $ vi studentserver-deployment.yaml



```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ cat studentserver-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
        name: web
        labels:
                app: studentserver-deploy
spec:
        replicas: 1
        selector:
                matchLabels:
                        app: web
        template:
                metadata:
                        labels:
                                app: web
                spec:
                        containers:
                                - image: 19551/studentserver
                                  imagePullPolicy: Always
                                  name: web
                                  ports:
                                        - containerPort: 8080
                                  env:
                                        - name: MONGO_URL
                                          valueFrom:
                                                configMapKeyRef:
                                                        name: studentserver-config
                                                        key: MONGO_URL
                                - name: MONGO_DATABASE
                                  valueFrom:
                                        configMapKeyRef:
                                                name: studentserver-config
                                                key: MONGO_DATABASE
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$
```

2. Create bookshelf-deployment.yaml

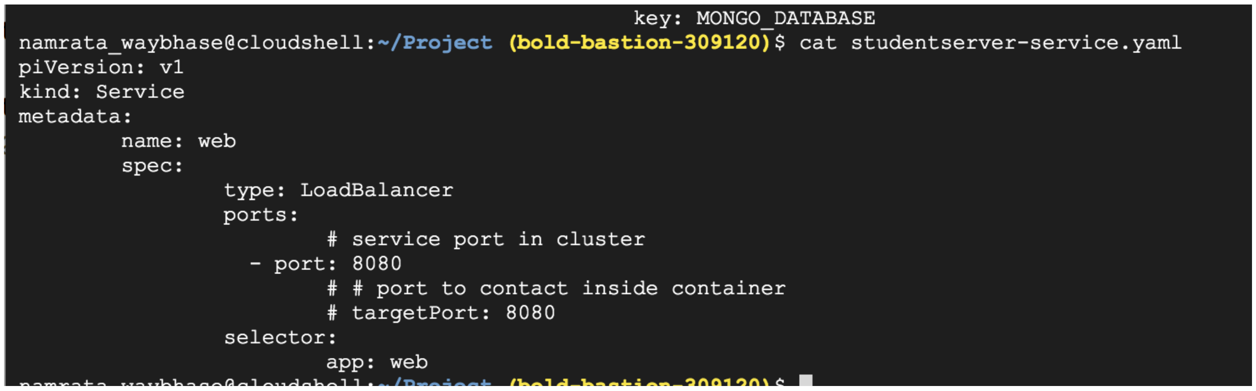
   $ vi bookshelf-deployment.yaml



```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ cat bookshelf-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
        name: bookshelf-deployment
        labels:
                app: bookshelf-deployment
spec:
        replicas: 1
        selector:
                matchLabels:
                        app: bookshelf-deployment
        template:
                metadata:
                        labels:
                                app: bookshelf-deployment
                spec:
                        containers:
                                - image: 19551/bookshelf
                                  imagePullPolicy: Always
                                  name: bookshelf-deployment
                                  ports:
                                        - containerPort: 5000
                                  env:
                                        - name: MONGO_URL
                                          valueFrom:
                                                configMapKeyRef:
                                                        name: bookshelf-config
                                                        key: MONGO_URL
                                        - name: MONGO_DATABASE
                                          valueFrom:
                                                configMapKeyRef:
                                                name: bookshelf-config
```

3. Create studentserver-service.yaml

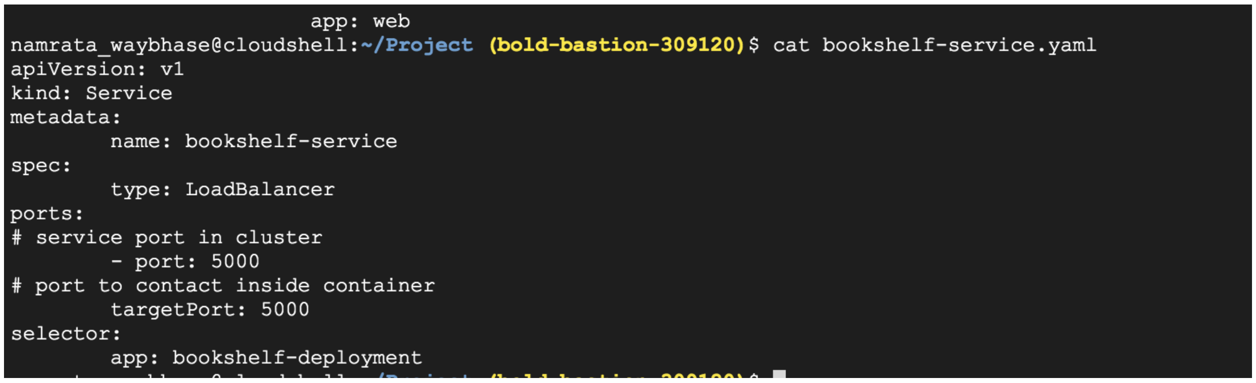
$ vi studentserver-service.yaml



```
                                            key: MONGO_DATABASE
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ cat studentserver-service.yaml
piVersion: v1
kind: Service
metadata:
        name: web
        spec:
                type: LoadBalancer
                ports:
                        # service port in cluster
                  - port: 8080
                        # # port to contact inside container
                        # targetPort: 8080
                selector:
                        app: web
```

4. Create bookshelf-service.yaml

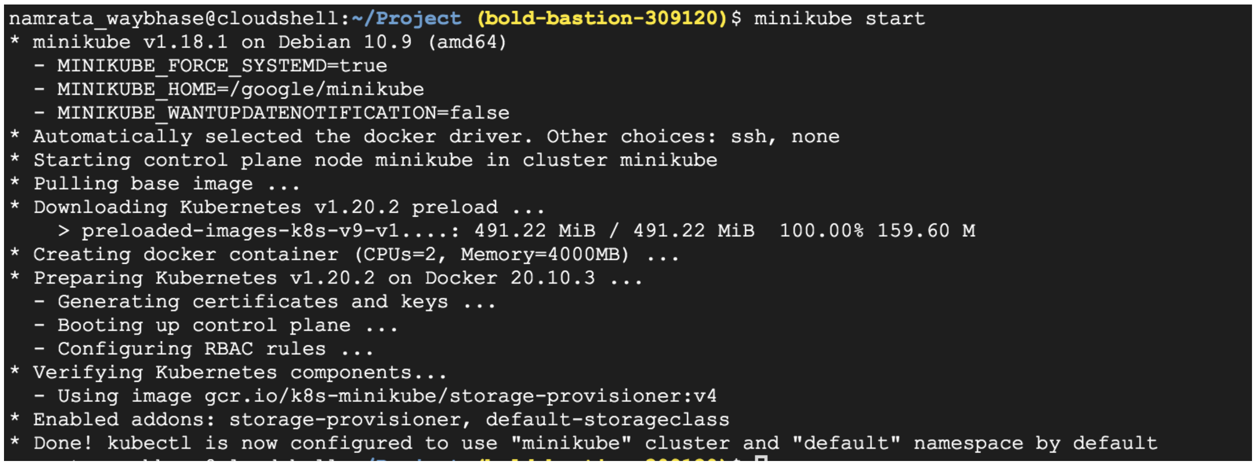
$ vi bookshelf-service.yaml



```
                        app: web
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ cat bookshelf-service.yaml
apiVersion: v1
kind: Service
metadata:
        name: bookshelf-service
spec:
        type: LoadBalancer
ports:
# service port in cluster
        - port: 5000
# port to contact inside container
        targetPort: 5000
selector:
        app: bookshelf-deployment
```

5. Start minikube

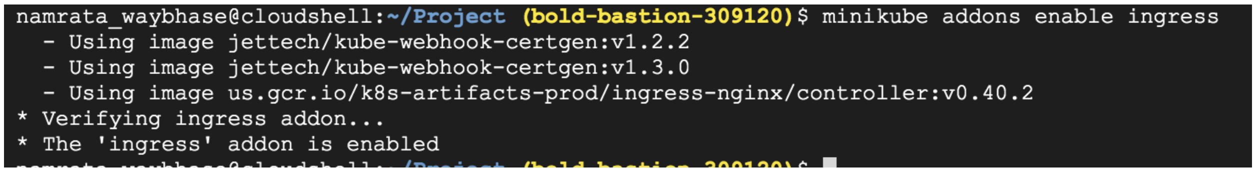
minikube start



```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ minikube start
* minikube v1.18.1 on Debian 10.9 (amd64)
  - MINIKUBE_FORCE_SYSTEMD=true
  - MINIKUBE_HOME=/google/minikube
  - MINIKUBE_WANTUPDATENOTIFICATION=false
* Automatically selected the docker driver. Other choices: ssh, none
* Starting control plane node minikube in cluster minikube
* Pulling base image ...
* Downloading Kubernetes v1.20.2 preload ...
    > preloaded-images-k8s-v9-v1....: 491.22 MiB / 491.22 MiB  100.00% 159.60 M
* Creating docker container (CPUs=2, Memory=4000MB) ...
* Preparing Kubernetes v1.20.2 on Docker 20.10.3 ...
  - Generating certificates and keys ...
  - Booting up control plane ...
  - Configuring RBAC rules ...
* Verifying Kubernetes components...
  - Using image gcr.io/k8s-minikube/storage-provisioner:v4
* Enabled addons: storage-provisioner, default-storageclass
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

6. Start Ingress

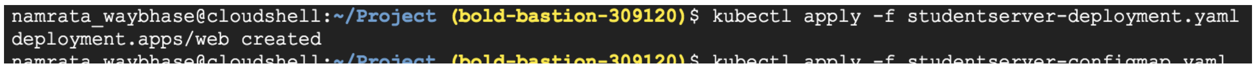
$ minikube addons enable ingress



```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ minikube addons enable ingress
  - Using image jettech/kube-webhook-certgen:v1.2.2
  - Using image jettech/kube-webhook-certgen:v1.3.0
  - Using image us.gcr.io/k8s-artifacts-prod/ingress-nginx/controller:v0.40.2
* Verifying ingress addon...
* The 'ingress' addon is enabled
```

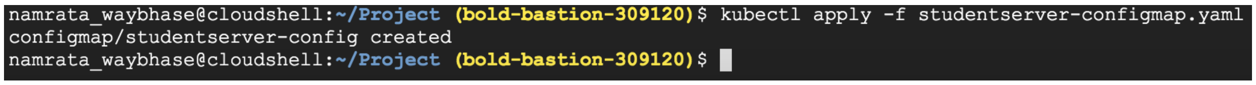
7. Create studentserver related pods and start service using the above yaml file

$ kubectl apply -f studentserver-deployment.yaml



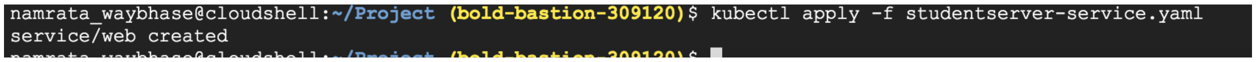
```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ kubectl apply -f studentserver-deployment.yaml
deployment.apps/web created
```

$ kubectl apply -f studentserver-configmap.yaml



```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ kubectl apply -f studentserver-configmap.yaml
configmap/studentserver-config created
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$
```

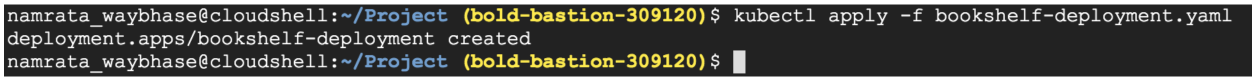
$ kubectl apply -f studentserver-service.yaml



```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ kubectl apply -f studentserver-service.yaml
service/web created
```

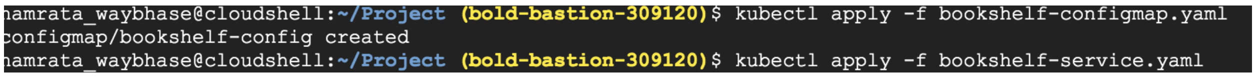
8. Create bookshelf related pods and start service using the above yaml file

$ kubectl apply -f bookshelf-deployment.yaml



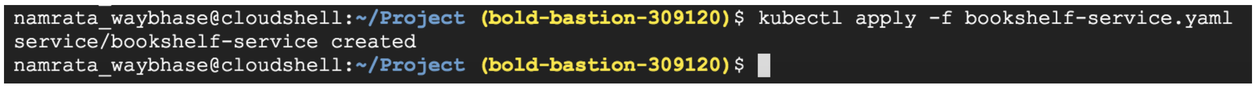
```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ kubectl apply -f bookshelf-deployment.yaml
deployment.apps/bookshelf-deployment created
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$
```

$ kubectl apply -f bookshelf-configmap.yaml



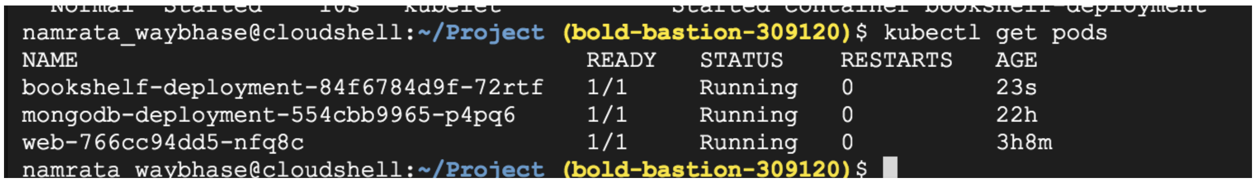
```
amrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ kubectl apply -f bookshelf-configmap.yaml
onfigmap/bookshelf-config created
amrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ kubectl apply -f bookshelf-service.yaml
```

$ kubectl apply -f bookshelf-service.yaml



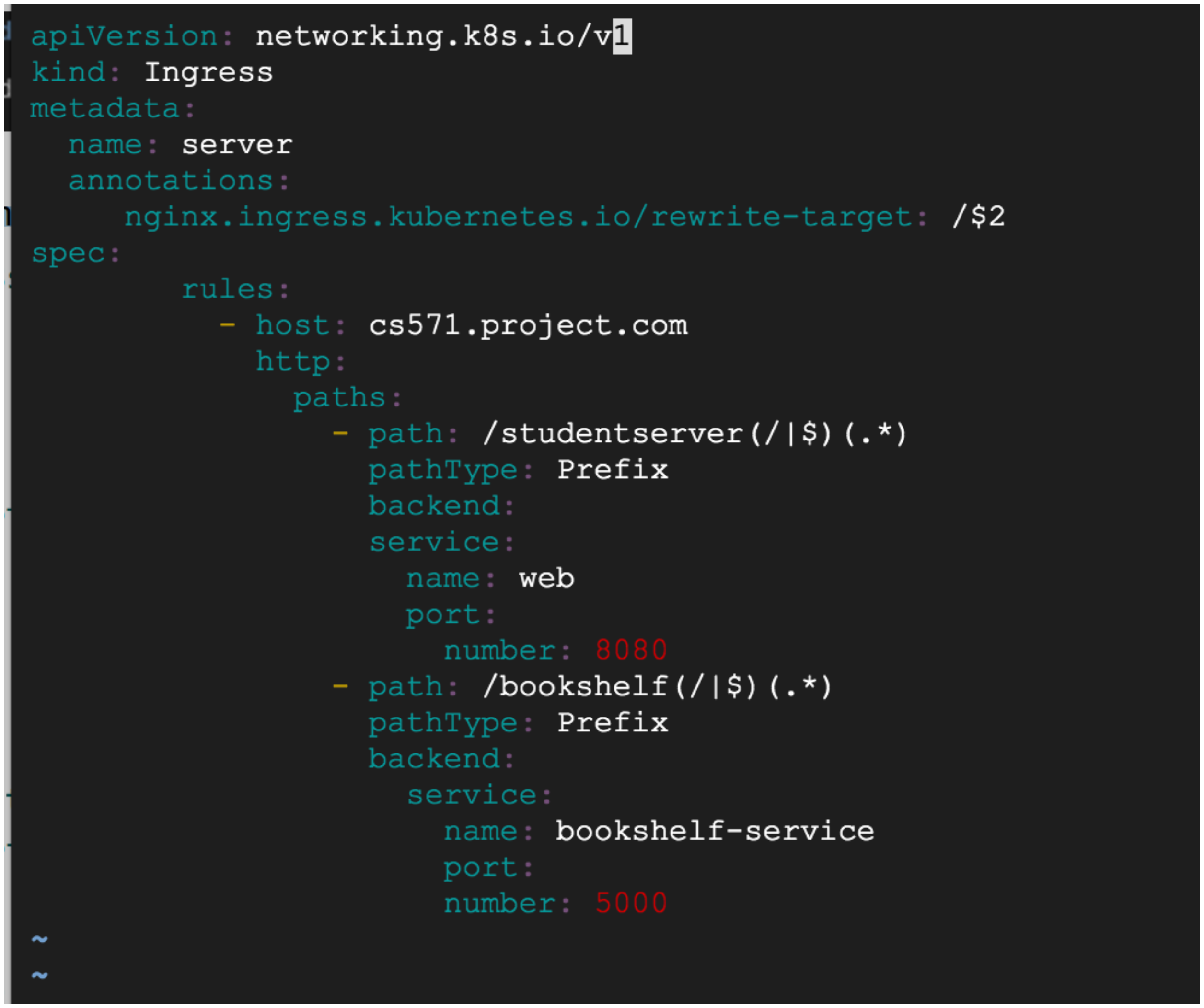
```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ kubectl apply -f bookshelf-service.yaml
service/bookshelf-service created
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$
```

9. Check if all the pods are running correctly
$ kubectl get pods



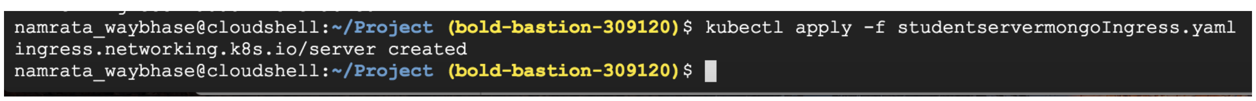
```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ kubectl get pods
NAME                                       READY   STATUS    RESTARTS   AGE
bookshelf-deployment-84f6784d9f-72rtf      1/1     Running   0          23s
mongodb-deployment-554cbb9965-p4pq6        1/1     Running   0          22h
web-766cc94dd5-nfq8c                       1/1     Running   0          3h8m
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$
```

10 . Create an ingress service yaml file
$ vi studentservermongoIngress.yaml



```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: server
  annotations:
     nginx.ingress.kubernetes.io/rewrite-target: /$2
spec:
        rules:
          - host: cs571.project.com
            http:
              paths:
                - path: /studentserver(/|$)(.*)
                  pathType: Prefix
                  backend:
                  service:
                    name: web
                    port:
                      number: 8080
                - path: /bookshelf(/|$)(.*)
                  pathType: Prefix
                  backend:
                    service:
                      name: bookshelf-service
                      port:
                      number: 5000
~
~
```

11. Create the ingress service using the above yaml file
kubectl apply -f studentservermongoIngress.yaml



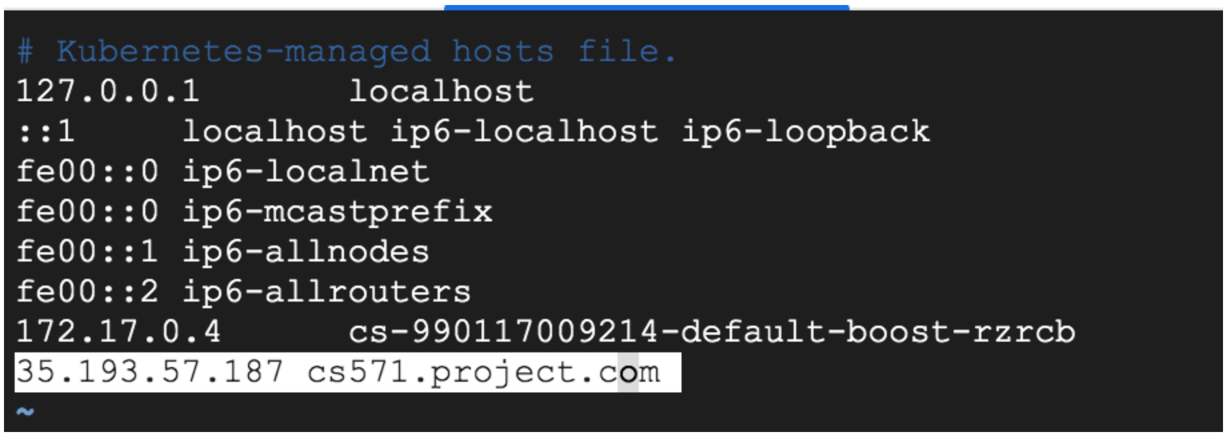
```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ kubectl apply -f studentservermongoIngress.yaml
ingress.networking.k8s.io/server created
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$
```

12. Check if ingress is running
 kubectl get ingress

12. Add Addreee to /etc/hosts
 vi /etc/hosts

Add the address you got from above step to the end of the file
 Your-address cs571.project.com

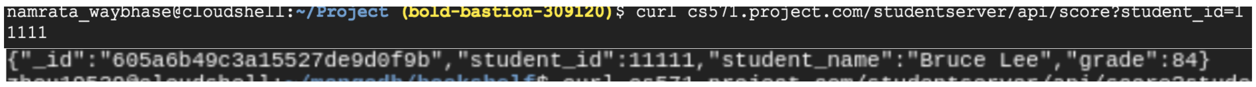
Your /etc/hosts file should look something like this after adding the line, but your address should be different from mine



```
# Kubernetes-managed hosts file.
127.0.0.1       localhost
::1     localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
fe00::0 ip6-mcastprefix
fe00::1 ip6-allnodes
fe00::2 ip6-allrouters
172.17.0.4      cs-990117009214-default-boost-rzrcb
35.193.57.187 cs571.project.com
~
```

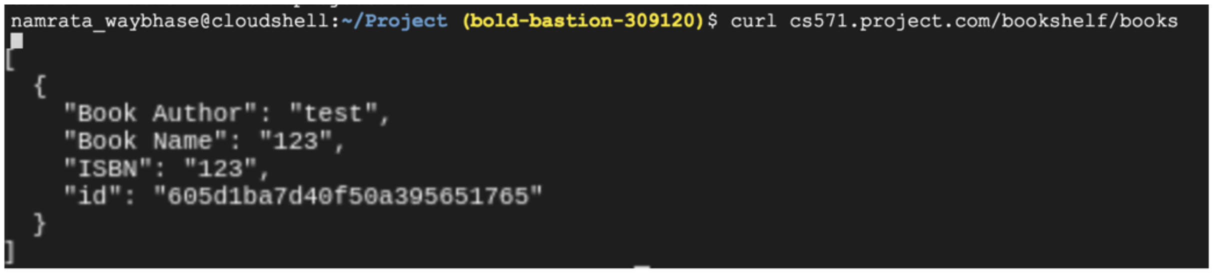
14. If everything goes smoothly, you should be able to access your applications

curl cs571.project.com/studentserver/api/score?student_id=11111



```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ curl cs571.project.com/studentserver/api/score?student_id=1
1111
{"_id":"605a6b49c3a15527de9d0f9b","student_id":11111,"student_name":"Bruce Lee","grade":84}
```

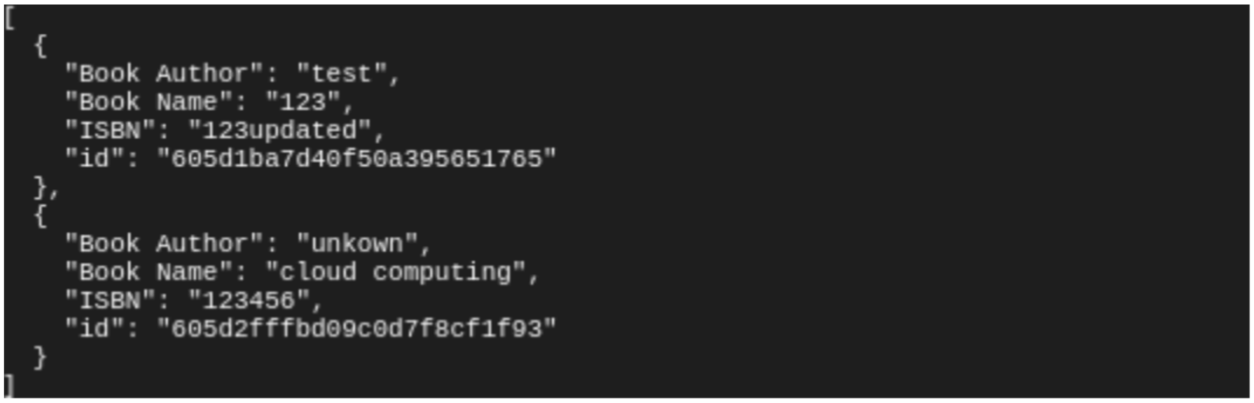
15. $ curl cs571.project.com/bookshelf/books



```
namrata_waybhase@cloudshell:~/Project (bold-bastion-309120)$ curl cs571.project.com/bookshelf/books
[
  {
    "Book Author": "test",
    "Book Name": "123",
    "ISBN": "123",
    "id": "605d1ba7d40f50a395651765"
  }
]
```

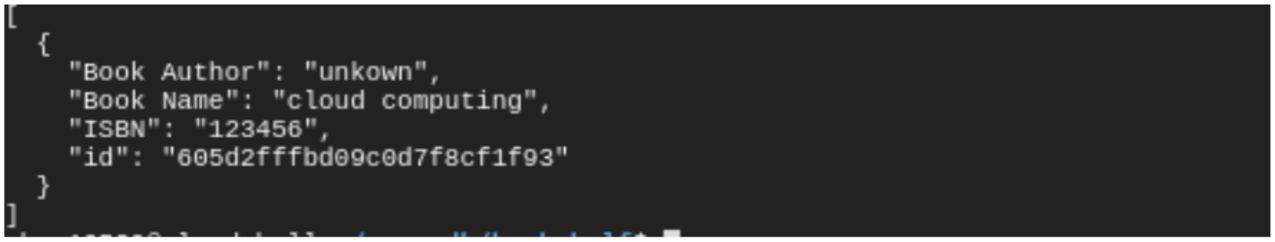
16 .Add a book curl -X POST -d "{\"book_name\": \"cloud computing\",\"book_author\": \"unkown\", \"isbn\": \"123456\" }" http://cs571.project.com/bookshelf/book

$ curl cs571.project.com/bookshelf/books



```
[
  {
    "Book Author": "test",
    "Book Name": "123",
    "ISBN": "123updated",
    "id": "605d1ba7d40f50a395651765"
  },
  {
    "Book Author": "unkown",
    "Book Name": "cloud computing",
    "ISBN": "123456",
    "id": "605d2fffbd09c0d7f8cf1f93"
  }
]
```

17. Delete a book curl -X DELETE cs571.project.com/bookshelf/book/id



```
[
  {
    "Book Author": "unkown",
    "Book Name": "cloud computing",
    "ISBN": "123456",
    "id": "605d2fffbd09c0d7f8cf1f93"
  }
]
```