Signature Project

Name: Lidia Y. Bereketeab

ID - 20029

Section: CS571

Instructor: Prof. Chang

Date: April 10th, 2024

Week 12: Homework: Chapter 7: Con4igmap: Signature Project: MongoDB + Python Flask Web Framework + REST API + GKE

https://hc.labnet.sfbu.edu/~henry/sfbu/course/cloud_computing/genai/slide/exercise_kubernetes.html

A. 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-west1

Wait for the creation to finish, it might take a couple of minutes.

```
lbereket625@cloudshell:~ (cs-571-project-1)$ gcloud container clusters create kubia --num-nodes=1 --machine-type=e2-micro --region=us-west1
Default change: VPC-native is the default mode during cluster creation for versions greater than 1.21.0-gke.1500. To create advanced routes based clusters, please pass the `--no-ena
ble-ip-alias` flag
Note: Your Pod address range (`--cluster-ipv4-cidr`) can accommodate at most 1008 node(s).
Creating cluster kubia in us-west1... Cluster is being health-checked (master is healthy)...done.
Created [https://container.googleapis.com/v1/projects/cs-571-project-1/zones/us-west1/clusters/kubia].
To inspect the contents of your cluster, go to: https://console.cloud.google.com/kubernetes/workload_/gcloud/us-west1/kubia?project=cs-571-project-1
kubeconfig entry generated for kubia.
NAME: kubia
LOCATION: us-west1
MASTER_VERSION: 1.27.8-gke.1067004
MASTER_IP: 34.168.175.173
MACHINE_TYPE: e2-micro
NODE_VERSION: 1.27.8-gke.1067004
NUM_NODES: 3
STATUS: RUNNING
lbereket625@cloudshell:~ (cs-571-project-1)$
```

2. Let's create a Persistent Volume first, if you have created a persistent volume for the week10's homework, you can skip this one

~ gcloud compute disks create mongod --size=200GiB --region=us-west1 --replica-zones=us-west1-a,us-west1-b

```
lbereket625@cloudshell:~ (cs-571-project-1)$ gcloud compute disks create mongod --size=200GiB --region=us-west1 --replica-zones=us-west1-a,us-west1-b
Created [https://www.googleapis.com/compute/v1/projects/cs-571-project-1/regions/us-west1/disks/mongod].
NAME: mongod
ZONE:
SIZE_GB: 200
TYPE: pd-standard
STATUS: READY

New disks are unformatted. You must format and mount a disk before it
can be used. You can find instructions on how to do this at:

https://cloud.google.com/compute/docs/disks/add-persistent-disk#formatting

lbereket625@cloudshell:~ (cs-571-project-1)$
```

3. Now create a mongodb-deployment with this yaml file and the content

~ vim mongodb-deployment.yaml


```
lbereket625@cloudshell:~ (cs-571-project-1)$ vim mongodb-deployment.yaml
lbereket625@cloudshell:~ (cs-571-project-1)$ 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:
      - image: mongo
        name: mongo
        ports:
        - containerPort: 27017
        volumeMounts:
        - name: mongodb-data
          mountPath: /data/db
      volumes:
      - name: mongodb-data
        gcePersistentDisk:
          pdName: mongodb
          fsType: ext4
lbereket625@cloudshell:~ (cs-571-project-1)$
```

apply the configuration defined in the YAML file mongodb-deployment.yaml to your Kubernetes cluster

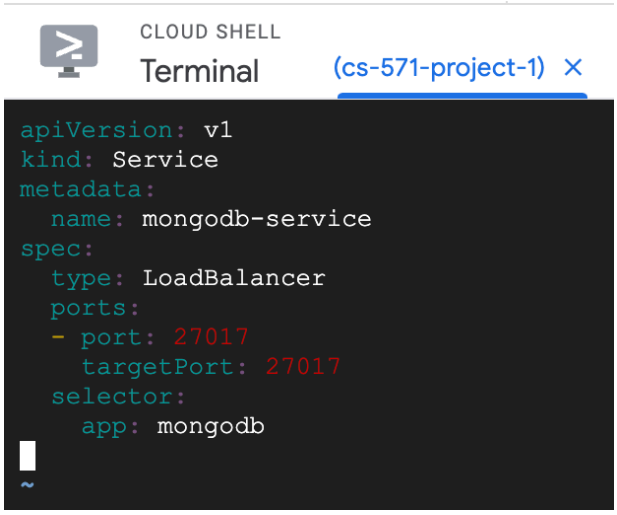
~kubectl apply -f mongodb-deployment.yaml

```
lbereket625@cloudshell:~ (cs-571-project-1)$ kubectl apply -f mongodb-deployment.yaml
deployment.apps/mongodb-deployment created
lbereket625@cloudshell:~ (cs-571-project-1)$
```

4. Check if the deployment pod has been successfully created and started running kubectl get pods

```
lbereket625@cloudshell:~ (cs-571-project-1)$ kubectl get nodes
NAME                                     STATUS   ROLES    AGE   VERSION
gke-kubia-default-pool-11522e33-tc4r     Ready    <none>   51m   v1.27.8-gke.1067004
gke-kubia-default-pool-48a72d40-3mq4     Ready    <none>   51m   v1.27.8-gke.1067004
gke-kubia-default-pool-80e875d8-gh0v     Ready    <none>   51m   v1.27.8-gke.1067004
lbereket625@cloudshell:~ (cs-571-project-1)$ kubectl get pods
NAME                                  READY   STATUS    RESTARTS   AGE
mongodb-deployment-594c77dcdf-c5j4t   1/1     Running   0          11m
lbereket625@cloudshell:~ (cs-571-project-1)$
```

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



```
lbereket625@cloudshell:~ (cs-571-project-1)$ vim mongodb-service.yaml
lbereket625@cloudshell:~ (cs-571-project-1)$ cat mongodb-service.yaml
apiVersion: v1
kind: Service
metadata:
  name: mongodb-service
spec:
  type: LoadBalancer
  ports:
  - port: 27017
    targetPort: 27017
  selector:
    app: mongodb

lbereket625@cloudshell:~ (cs-571-project-1)$
```

~kubectl apply -f mongodb-service.yaml

```
lbereket625@cloudshell:~ (cs-571-project-1)$ kubectl apply -f mongodb-service.yaml
service/mongodb-service created
lbereket625@cloudshell:~ (cs-571-project-1)$
```

6. Wait couple of minutes, and check if the service is up

~kubectl get svc

Please wait until you see the external-ip is generated for mongodb-service, then you can move forward

```
lbereket625@cloudshell:~ (cs-571-project-1)$ kubectl get svc
NAME              TYPE           CLUSTER-IP     EXTERNAL-IP   PORT(S)           AGE
kubernetes        ClusterIP      10.89.48.1     <none>        443/TCP           47m
mongodb-service   LoadBalancer   10.89.62.200   <pending>     27017:30926/TCP   37s
lbereket625@cloudshell:~ (cs-571-project-1)$ kubectl get svc
NAME              TYPE           CLUSTER-IP     EXTERNAL-IP     PORT(S)           AGE
kubernetes        ClusterIP      10.89.48.1     <none>          443/TCP           48m
mongodb-service   LoadBalancer   10.89.62.200   35.233.161.15   27017:30926/TCP   71s
lbereket625@cloudshell:~ (cs-571-project-1)$
```

General Instruction Incase the mongodb doesn't work in shell or it isn't installed ahead of use:

Here is how to install the MongoDB shell in the cloud shell of GCP :

Step 1: Retrieve MongoDB Public GPG Key

wget -qO - https://www.mongodb.org/static/pgp/server-4.4.asc | sudo apt-key add -

Explanation: This downloads MongoDB's public GPG key and adds it to the list of trusted keys used by your system's package manager (apt).

Step 2: Add MongoDB Repository

echo "deb http://repo.mongodb.org/apt/debian buster/mongodb-org/4.4 main" | sudo tee /etc/apt/sources.list.d/mongodb-org-4.4.list

Explanation: This command adds the MongoDB repository URL to the list of package sources used by apt, allowing you to install MongoDB packages.

Step 3: Update Package Index

sudo apt-get update

Explanation: This command updates the local package index, ensuring that apt has the latest information about available packages, including those from the newly added MongoDB repository.

Step 4: Install MongoDB Shell Package

sudo apt-get install -y mongodb-org-shell

Explanation: This command installs the MongoDB shell package, which provides a command-line interface for interacting with MongoDB databases

Step 5. For ease of use and access you can also install mongosh after the update

Sudo nom install -g mongosh

#Explanation: This command globally installs `mongosh`, a modern MongoDB shell, using npm. `mongosh` provides a user-friendly command-line interface for MongoDB interactions, enhancing ease of use and access

```
lbereket625@cloudshell:~ (cs-571-project-1)$ wget -qO - https://www.mongodb.org/static/pgp/server-4.4.asc | sudo apt-key add -
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
lbereket625@cloudshell:~ (cs-571-project-1)$ echo "deb http://repo.mongodb.org/apt/debian buster/mongodb-org/4.4 main" | sudo tee /etc/apt/sources.list.d/mongodb-org-4.4.list
deb http://repo.mongodb.org/apt/debian buster/mongodb-org/4.4 main
lbereket625@cloudshell:~ (cs-571-project-1)$
```

```
lbereket625@cloudshell:~ (cs-571-project-1)$ sudo npm install -g mongosh
sudo: npm: command not found
lbereket625@cloudshell:~ (cs-571-project-1)$ sudo apt update
Hit:1 https://apt.llvm.org/bullseye llvm-toolchain-bullseye-13 InRelease
Get:2 https://cli.github.com/packages bullseye InRelease [3,921 B]
Get:3 https://apt.releases.hashicorp.com bullseye InRelease [12.9 kB]
Hit:4 https://packages.cloud.google.com/apt gcsfuse-bullseye InRelease
Get:5 https://packages.cloud.google.com/apt cloud-sdk-bullseye InRelease [6,406 B]
Hit:6 https://repo.mysql.com/apt/debian bullseye InRelease
Get:7 https://packages.microsoft.com/debian/11/prod bullseye InRelease [3,650 B]
Get:8 https://cli.github.com/packages bullseye/main amd64 Packages [346 B]
Get:9 https://apt.releases.hashicorp.com bullseye/main amd64 Packages [126 kB]
Get:10 https://apt.postgresql.org/pub/repos/apt bullseye-pgdg InRelease [123 kB]
Get:11 https://packages.cloud.google.com/apt cloud-sdk-bullseye/main amd64 Packages [475 kB]
Get:12 https://packages.microsoft.com/debian/11/prod bullseye/main armhf Packages [27.4 kB]
Get:13 https://packages.microsoft.com/debian/11/prod bullseye/main arm64 Packages [31.4 kB]
Get:14 https://packages.microsoft.com/debian/11/prod bullseye/main amd64 Packages [151 kB]
Get:15 https://apt.postgresql.org/pub/repos/apt bullseye-pgdg/main amd64 Packages [311 kB]
Hit:16 http://deb.debian.org/debian bullseye InRelease
Get:17 http://deb.debian.org/debian-security bullseye-security InRelease [48.4 kB]
Get:18 http://deb.debian.org/debian bullseye-updates InRelease [44.1 kB]
Get:19 https://download.docker.com/linux/debian bullseye InRelease [43.3 kB]
Get:20 https://packages.sury.org/php bullseye InRelease [7,551 B]
Get:21 http://deb.debian.org/debian-security bullseye-security/main Sources [170 kB]
Get:22 http://deb.debian.org/debian-security bullseye-security/main amd64 Packages [271 kB]
Get:23 https://packages.sury.org/php bullseye/main amd64 Packages [239 kB]
Fetched 2,096 kB in 6s (335 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
64 packages can be upgraded. Run 'apt list --upgradable' to see them.
lbereket625@cloudshell:~ (cs-571-project-1)$
```

```
lbereket625@cloudshell:~ (cs-571-project-1)$ sudo npm install -g mongosh
npm WARN EBADENGINE Unsupported engine {
npm WARN EBADENGINE   package: 'mongosh@2.2.3',
npm WARN EBADENGINE   required: { node: '>=16.15.0' },
npm WARN EBADENGINE   current: { node: 'v12.22.12', npm: '7.5.2' }
npm WARN EBADENGINE }
npm WARN EBADENGINE Unsupported engine {
npm WARN EBADENGINE   package: '@mongosh/cli-repl@2.2.3',
npm WARN EBADENGINE   required: { node: '>=16.15.0' },
npm WARN EBADENGINE   current: { node: 'v12.22.12', npm: '7.5.2' }
npm WARN EBADENGINE }
```

7. Now try and see if mongoDB is functioning for connections using the External-IP

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

```
lbereket625@cloudshell:~ (cs-571-project-1)$ kubectl exec -it mongodb-deployment-594c77dcdf-c5j4t -- bash
root@mongodb-deployment-594c77dcdf-c5j4t:/#
```

```
lbereket625@cloudshell:~ (cs-571-project-1)$ kubectl exec -it mongodb-deployment-594c77dcdf-c5j4t -- bash
root@mongodb-deployment-594c77dcdf-c5j4t:/# mongosh 35.233.161.15
Current Mongosh Log ID: 661749774d5aa83d977b2da8
Connecting to:          mongodb://35.233.161.15:27017/?directConnection=true&appName=mongosh+2.2.2
Using MongoDB:          7.0.8
Using Mongosh:          2.2.2

For mongosh info see: https://docs.mongodb.com/mongodb-shell/


To help improve our products, anonymous usage data is collected and sent to MongoDB periodically (https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

------
   The server generated these startup warnings when booting
   2024-04-11T02:13:21.308+00:00: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem
   2024-04-11T02:13:22.067+00:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
   2024-04-11T02:13:22.068+00:00: vm.max_map_count is too low
------

test> exit
```

To Exit the shell : type “exit” , twice

```
test> exit
root@mongodb-deployment-594c77dcdf-c5j4t:/# exit
exit
```

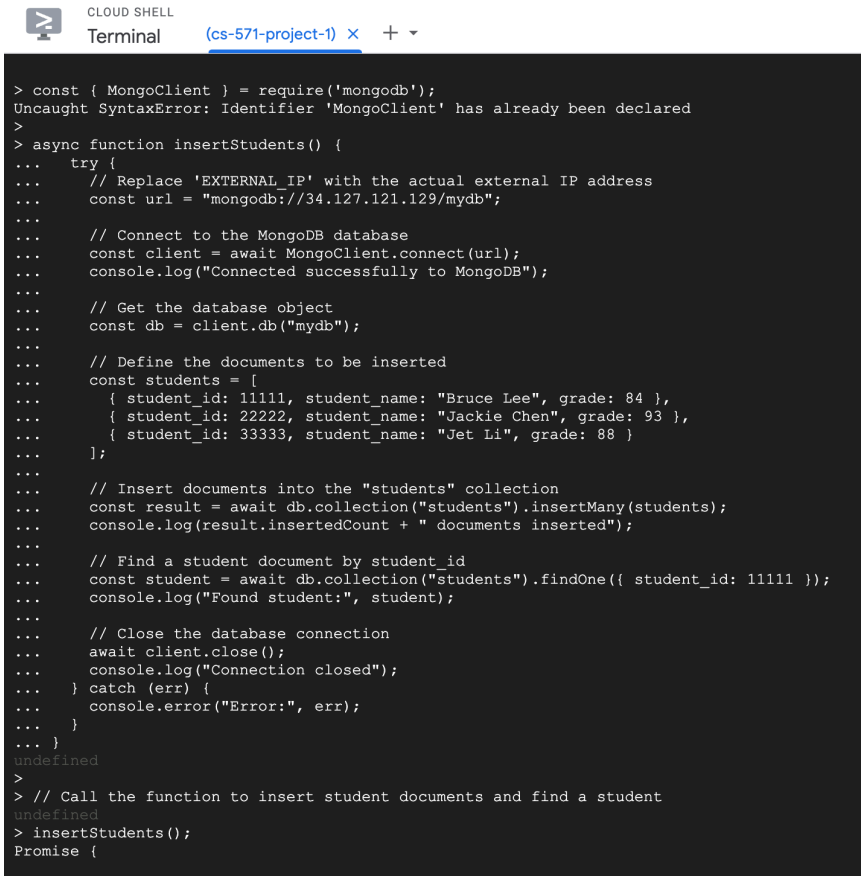
8. We need to insert some records into the mongoDB for later use

~ node

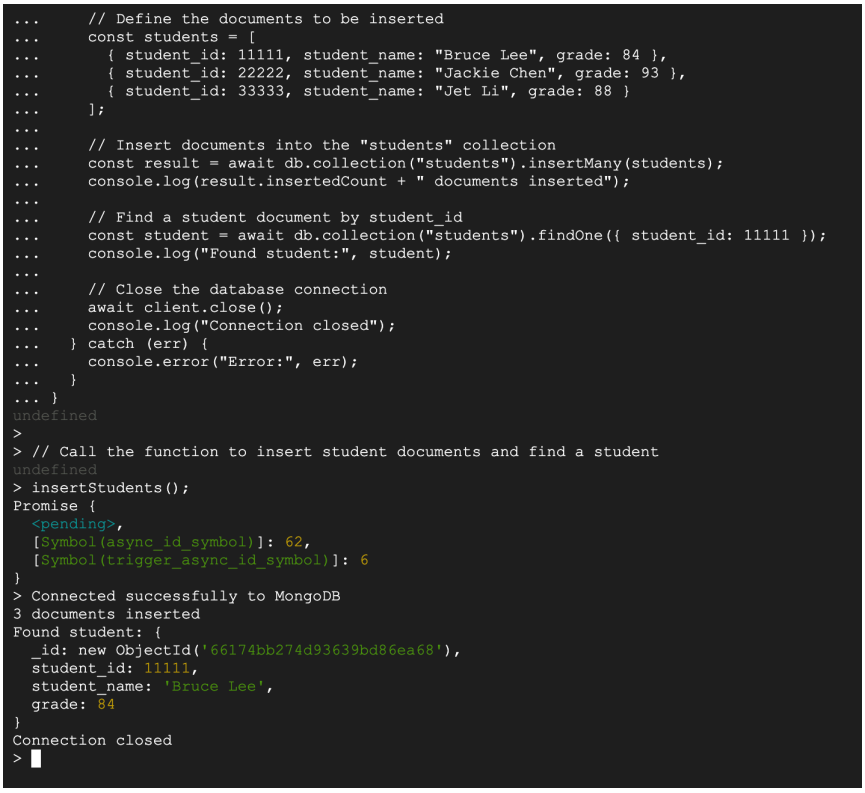
```
lbereket625@cloudshell:~ (cs-571-project-1)$ node
Welcome to Node.js v20.12.1.
Type ".help" for more information.
>
```

```
const { MongoClient } = require('mongodb');
async function insertStudents() {
 try {
  // Replace ‘eXTERNAL ip' with the actual external IP address
  const url = “mongodb://“EXTERNAL IP"/mydb";
  // Connect to the MongoDB database
  const client = await MongoClient.connect(url);
  console.log("Connected successfully to MongoDB");

  // Get the database object
  const db = client.db("mydb");

  // Define the documents to be inserted
  const students = [
   { 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 }
  ];

  // Insert documents into the "students" collection
  const result = await db.collection("students").insertMany(students);
  console.log(result.insertedCount + " documents inserted");

  // Find a student document by student_id
  const student = await db.collection("students").findOne({ student_id: 11111 });
  console.log("Found student:", student);

  // Close the database connection
  await client.close();
  console.log("Connection closed");
 } catch (err) {
  console.error("Error:", err);
 } }
// Call the function to insert student documents and find a student
insertStudents();
```

This is showing that 3 records/documents has been added/inserted:



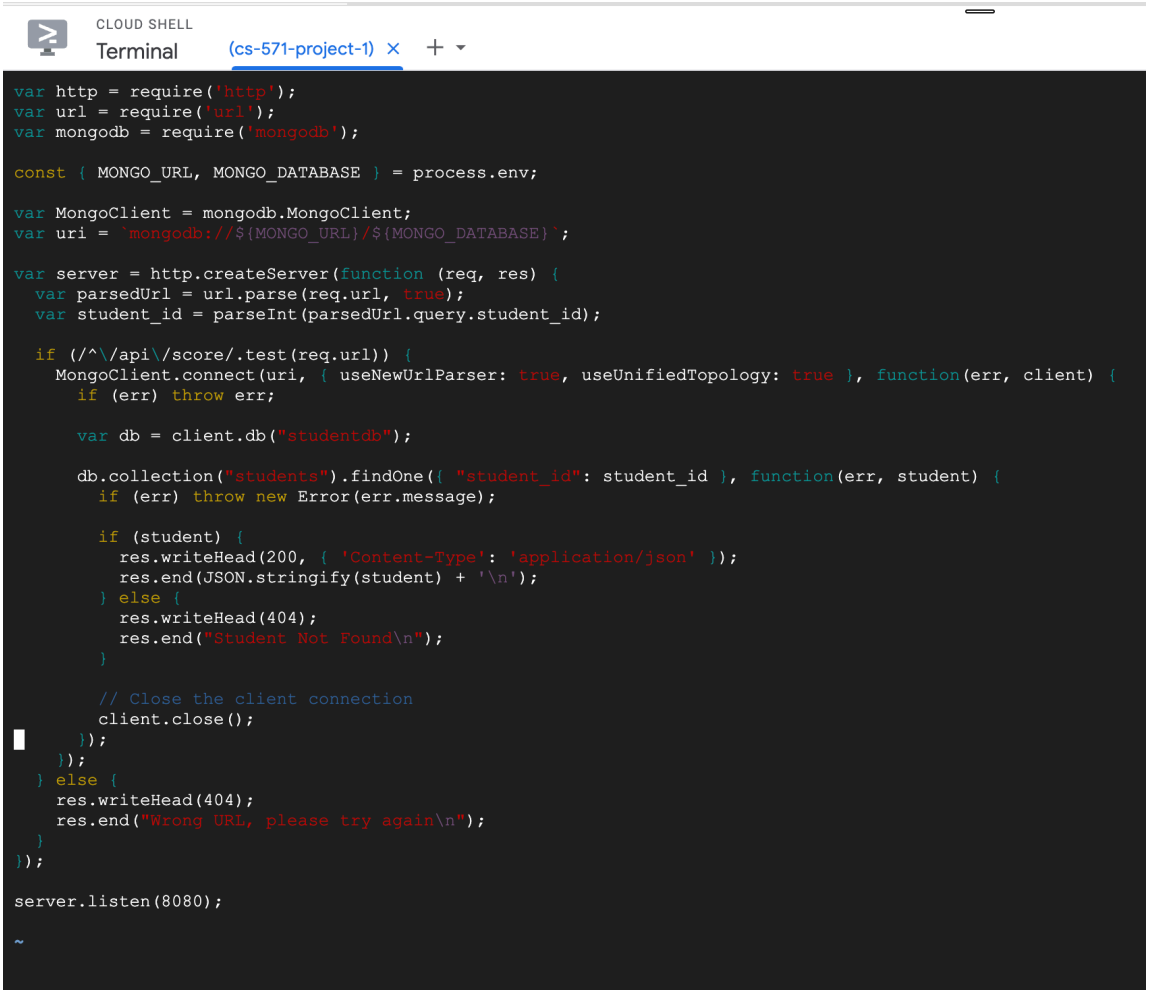
This is showing that 3 records/documents has been added/inserted:



B. Modify our studentServer to get records from MongoDB and deploy to GKE

1. Create studentServer JavaScript File:

~vim studentServer.js



2. Create Dockerfile

   FROM node:7

   ADD studentServer.js /studentServer.js

   ENTRYPOINT ["node", "studentServer.js"]

   RUN npm config set registry https://registry.npmjs.org/

3. Build the studentserver docker image

   ~docker build -t yourdockerhubID/studentserver

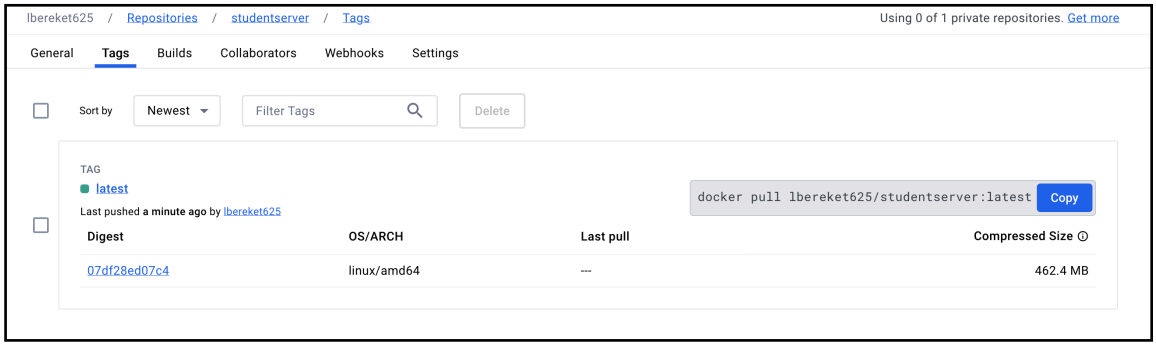
```
lbereket625@cloudshell:~ (cs-571-project-1)$ docker build -t lbereket625/studentserver .
[+] Building 73.8s (10/10) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 489B
 => [internal] load metadata for docker.io/library/node:20.11.1
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [stage-2 1/5] FROM docker.io/library/node:20.11.1@sha256:e06aae17c40c7a6b5296ca6f942a02e6737ae61bbbf3e2158624bb0f887991b5
 => => resolve docker.io/library/node:20.11.1@sha256:e06aae17c40c7a6b5296ca6f942a02e6737ae61bbbf3e2158624bb0f887991b5
 => => sha256:894c91653e564b9ba29400e6e1675bbcbc667158c1706c655dd4ccf84d0f54e8 2.00kB / 2.00kB
 => => sha256:2e805f601f2baab256bea06f32b5bf1e0b5f3dfc6e4287d67368553be34b39fe 7.34kB / 7.34kB
 => => sha256:e06aae17c40c7a6b5296ca6f942a02e6737ae61bbbf3e2158624bb0f887991b5 1.21kB / 1.21kB
 => => sha256:71215d55680cf0ab2dcc0e1dd65ed76414e3fb0c294249b5b9319a8fa7c398e4 49.55MB / 49.55MB
 => => sha256:3cb8f9c23302e175d87a827f0a1c376bd59b1f6949bd3bc24ab8da0d669cdfa0 24.05MB / 24.05MB
 => => sha256:5f899db30843f8330d5a40d1acb26bb00e93a9f21bff253f31c20562fa264767 64.14MB / 64.14MB
 => => sha256:567db630df8d441ffe43e050ede26996c87e3b33c99f79d4fba0bf6b7ffa0213 211.14MB / 211.14MB
 => => sha256:f4ac4e9f5ffb96287f2ff537a9cf450fc883facf1276832aac2360270cb0af2b 3.37kB / 3.37kB
 => => extracting sha256:71215d55680cf0ab2dcc0e1dd65ed76414e3fb0c294249b5b9319a8fa7c398e4
 => => sha256:5be7f83aa483f89031847decb91b9de90a1b550faa3c30cf6ebaa085d7ef2988 48.02MB / 48.02MB
 => => sha256:ff4370aa4380ae0ae6ff39d7c1ff2b78fbd6bc4f36b1ae2bf06e4fdf7e1422a3 2.21MB / 2.21MB
 => => sha256:5b2515aa9c667f834f13a82c22d05243df589533fde749fe034e02ad83692588 449B / 449B
 => => extracting sha256:3cb8f9c23302e175d87a827f0a1c376bd59b1f6949bd3bc24ab8da0d669cdfa0
 => => extracting sha256:5f899db30843f8330d5a40d1acb26bb00e93a9f21bff253f31c20562fa264767
 => => extracting sha256:567db630df8d441ffe43e050ede26996c87e3b33c99f79d4fba0bf6b7ffa0213
 => => extracting sha256:f4ac4e9f5ffb96287f2ff537a9cf450fc883facf1276832aac2360270cb0af2b
 => => extracting sha256:5be7f83aa483f89031847decb91b9de90a1b550faa3c30cf6ebaa085d7ef2988
 => => extracting sha256:ff4370aa4380ae0ae6ff39d7c1ff2b78fbd6bc4f36b1ae2bf06e4fdf7e1422a3
 => => extracting sha256:5b2515aa9c667f834f13a82c22d05243df589533fde749fe034e02ad83692588
 => [internal] load build context
 => => transferring context: 262.17MB
 => [stage-2 2/5] WORKDIR /app
 => [stage-2 3/5] COPY package.json .
 => [stage-2 4/5] RUN npm install
 => [stage-2 5/5] COPY . .
 => exporting to image
 => => exporting layers
 => => writing image sha256:45f4e8922f5d04b6341249d48a2f1feea674fc811c14dc2c92df6919f5fa4352
 => => naming to docker.io/lbereket625/studentserver
lbereket625@cloudshell:~ (cs-571-project-1)$
```

4. Push the docker image

   docker push lbereket625/studentserver

```
lbereket625@cloudshell:~ (cs-571-project-1)$ docker push lbereket625/studentserver
Using default tag: latest
The push refers to repository [docker.io/lbereket625/studentserver]
e81d19f791d8: Pushed
7874ace450f8: Pushed
b6db6c508ac0: Pushed
2479106d37fa: Pushed
ac68e27ae9cc: Mounted from library/node
9cef422ea209: Mounted from library/node
09ddcd01d2dc: Mounted from library/node
5358370f44ab: Mounted from library/node
21e1c4948146: Mounted from library/node
68866beb2ed2: Mounted from library/node
e6e2ab10dba6: Mounted from library/node
0238a1790324: Mounted from library/node
latest: digest: sha256:07df28ed07c47e7f0672cacdb84bc40b5e3d305fd98b7aa741419b06b00a395c size: 2840
lbereket625@cloudshell:~ (cs-571-project-1)$
```

For further check Docker:



C. Create a python Flask bookshelf REST API and deploy on GKE :

~vim bookshelf.py

```
from flask import Flask, request, jsonify
from flask_pymongo import PyMongo
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 the bookshelf app! I am running inside {} pod!".format(hostname)
    )
@app.route("/books")
def get_all_books():
    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"]
    })
    return jsonify(message="Book saved successfully!")

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

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

@app.route("/books/delete", methods=["POST"])
def delete_all_books():
    db.bookshelf.delete_many({})
    return jsonify(message="All books deleted!")
if __name__ == "__main__":
    app.run(host="0.0.0.0", port=5000)
```

1. Create Dockerfile

FROM python:alpine3.7

COPY . /app WORKDIR /app

RUN pip install --upgrade pip

RUN pip install -r requirements.txt

ENV PORT 5000

EXPOSE 5000

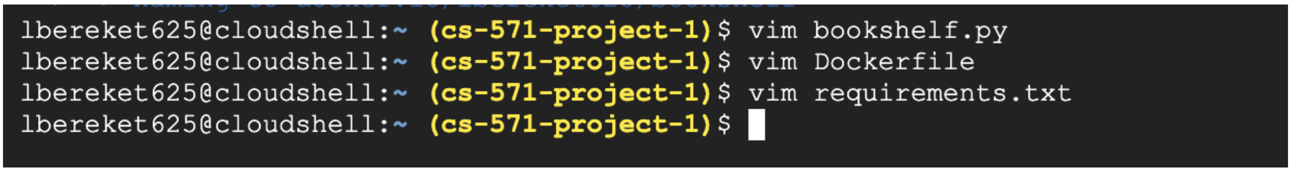
ENTRYPOINT ["python3"]

CMD ["bookshelf.py"]

requirements.txt

Flask==2.0.1

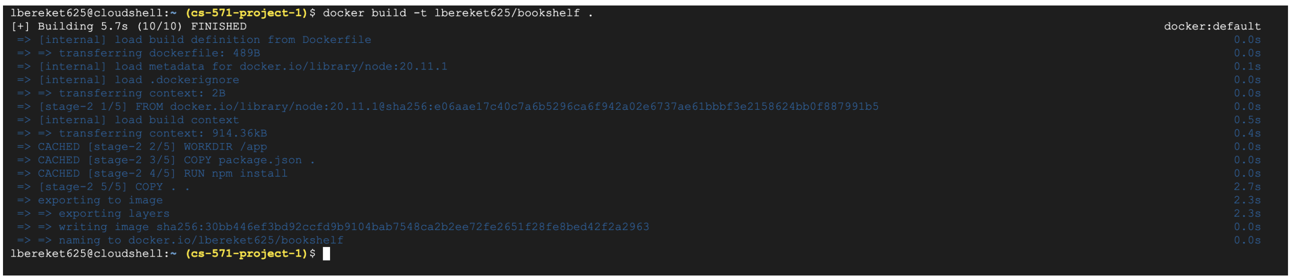
flask-pymongo==2.3.0



2. Build the bookshelf app into a docker image

   docker build -t yourdockerhubID/bookshelf .

   docker build -t lbereket625/bookshelf .

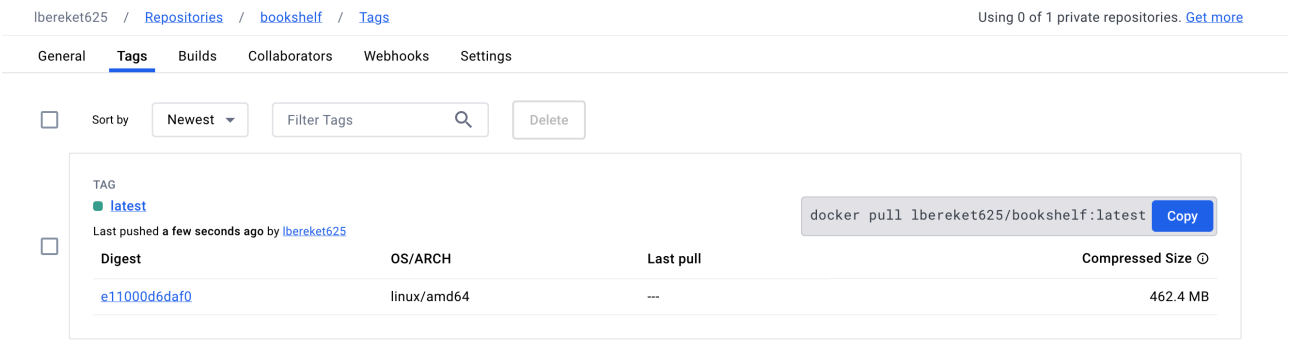


3. Push the docker image to your dockerhub

   docker push lbereket625/bookshelf

```
lbereket625@cloudshell:~ (cs-571-project-1)$ docker push lbereket625/bookshelf
Using default tag: latest
The push refers to repository [docker.io/lbereket625/bookshelf]
10a0848b2d03: Pushed
7874ace450f8: Mounted from lbereket625/studentserver
b6db6c508ac0: Mounted from lbereket625/studentserver
2479106d37fa: Mounted from lbereket625/studentserver
ac68e27ae9cc: Mounted from lbereket625/studentserver
9cef422ea209: Mounted from lbereket625/studentserver
09ddcd01d2dc: Mounted from lbereket625/studentserver
5358370f44ab: Mounted from lbereket625/studentserver
21e1c4948146: Mounted from lbereket625/studentserver
68866beb2ed2: Mounted from lbereket625/studentserver
e6e2ab10dba6: Mounted from lbereket625/studentserver
0238a1790324: Mounted from lbereket625/studentserver
latest: digest: sha256:e11000d6daf0b368d4996cc13744c01ad7335d0be226e27ef2f9f6fc9bc86c0b size: 2840
```

   Here is tag to show when is was pushed in docker dashboard.



D. 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: Change-this-to-your-mongoDB-EXTERNAL-IP

    MONGO_DATABASE: mydb

2. Create a file named bookshelf-configmap.yaml

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: bookshelf-config
data:
  MONGO_URL: Change-this-to-your-mongoDB-EXTERNAL-IP
  MONGO_DATABASE: mydb
```

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

1. Create 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: lbereket625/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
```

2. Create 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: lbereket625/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
     key: MONGO_DATABASE
```

3. Create studentserver-service.yaml

```
apiVersion: v1
kind: Service
metadata:
 name: web
spec:
 type: LoadBalancer
 ports:
 - port: 8080
  targetPort: 8080
 selector:
  app: web
```

4. Create bookshelf-service.yaml

```
apiVersion: v1
kind: Service
metadata:
 name: bookshelf-service
spec:
 type: LoadBalancer
 ports:
 - port: 5000
  targetPort: 5000
 selector:
  app: bookshelf-deployment
```

5. Start minikube

```
minikube start
```

```
lbereket625@cloudshell:~ (cs-571-project-1)$ minikube start
* minikube v1.32.0 on Debian 11.9 (amd64)
  - MINIKUBE_FORCE_SYSTEMD=true
  - MINIKUBE_HOME=/google/minikube
  - MINIKUBE_WANTUPDATENOTIFICATION=false
* Automatically selected the docker driver. Other choices: ssh, none
* Using Docker driver with root privileges
* Starting control plane node minikube in cluster minikube
* Pulling base image ...
* Downloading Kubernetes v1.28.3 preload ...
    > preloaded-images-k8s-v18-v1...:  403.35 MiB / 403.35 MiB  100.00% 164.27
    > gcr.io/k8s-minikube/kicbase...:  453.90 MiB / 453.90 MiB  100.00% 70.37 M
* Creating docker container (CPUs=2, Memory=4000MB) ...

X Docker is nearly out of disk space, which may cause deployments to fail! (96% of capacity). You can pass '--force' to skip this check.
* Suggestion:

    Try one or more of the following to free up space on the device:

    1. Run "docker system prune" to remove unused Docker data (optionally with "-a")
    2. Increase the storage allocated to Docker for Desktop by clicking on:
    Docker icon > Preferences > Resources > Disk Image Size
    3. Run "minikube ssh -- docker system prune" if using the Docker container runtime
* Related issue: https://github.com/kubernetes/minikube/issues/9024

* Preparing Kubernetes v1.28.3 on Docker 24.0.7 ...
  - kubelet.cgroups-per-qos=false
  - kubelet.enforce-node-allocatable=""
  - Generating certificates and keys ...
  - Booting up control plane ...
  - Configuring RBAC rules ...
* Configuring bridge CNI (Container Networking Interface) ...
  - Using image gcr.io/k8s-minikube/storage-provisioner:v5
* Verifying Kubernetes components...
* Enabled addons: storage-provisioner, default-storageclass
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
lbereket625@cloudshell:~ (cs-571-project-1)$
```

6. Start Ingress

```
minikube addons enable ingress
```

```
lbereket625@cloudshell:~ (cs-571-project-1)$ minikube addons enable ingress
* ingress is an addon maintained by Kubernetes. For any concerns contact minikube on GitHub.
You can view the list of minikube maintainers at: https://github.com/kubernetes/minikube/blob/master/OWNERS
  - Using image registry.k8s.io/ingress-nginx/controller:v1.9.4
  - Using image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v20231011-8b53cabe0
  - Using image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v20231011-8b53cabe0
* Verifying ingress addon...
* The 'ingress' addon is enabled
lbereket625@cloudshell:~ (cs-571-project-1)$
```

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

   kubectl apply -f studentserver-deployment.yaml

   kubectl apply -f studentserver-configmap.yaml

   kubectl apply -f studentserver-service.yaml

```
lbereket625@cloudshell:~ (cs-571-project-1)$ kubectl apply -f studentserver-deployment.yaml
deployment.apps/web created
lbereket625@cloudshell:~ (cs-571-project-1)$ kubectl apply -f studentserver-configmap.yaml
configmap/studentserver-config created
lbereket625@cloudshell:~ (cs-571-project-1)$ kubectl apply -f studentserver-service.yaml
service/web created
lbereket625@cloudshell:~ (cs-571-project-1)$
```

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

   kubectl apply -f bookshelf-deployment.yaml

   kubectl apply -f bookshelf-configmap.yaml

   kubectl apply -f bookshelf-service.yaml

```
lbereket625@cloudshell:~ (cs-571-project-1)$ kubectl apply -f bookshelf-deployment.yaml
deployment.apps/bookshelf-deployment created
lbereket625@cloudshell:~ (cs-571-project-1)$ kubectl apply -f bookshelf-configmap.yaml
configmap/bookshelf-config created
lbereket625@cloudshell:~ (cs-571-project-1)$ kubectl apply -f bookshelf-service.yaml
service/bookshelf-service created
lbereket625@cloudshell:~ (cs-571-project-1)$
```

9. Check if all the pods are running correctly

   kubectl get pods

```
lbereket625@cloudshell:~ (cs-571-project-1)$ kubectl get pods
NAME                                  READY   STATUS    RESTARTS   AGE
bookshelf-deployment-54dbb67fbf-zrbnb   1/1     Running   0          5m30s
web-675d7949c9-8hscm                  1/1     Running   0          6m2s
lbereket625@cloudshell:~ (cs-571-project-1)$
```

10. Create an ingress service yaml file called studentservermongoIngress.yaml with the following content

```
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

```
lbereket625@cloudshell:~ (cs-571-project-1)$ vim studentservermongoIngress.yaml
lbereket625@cloudshell:~ (cs-571-project-1)$ kubectl apply -f studentservermongoIngress.yaml
Warning: path /studentserver(/|$)(.*) cannot be used with pathType Prefix
Warning: path /bookshelf(/|$)(.*) cannot be used with pathType Prefix
ingress.networking.k8s.io/server created
lbereket625@cloudshell:~ (cs-571-project-1)$
```

12. Check if ingress is running

kubectl get ingress

Please wait until you see the Address, then move forward

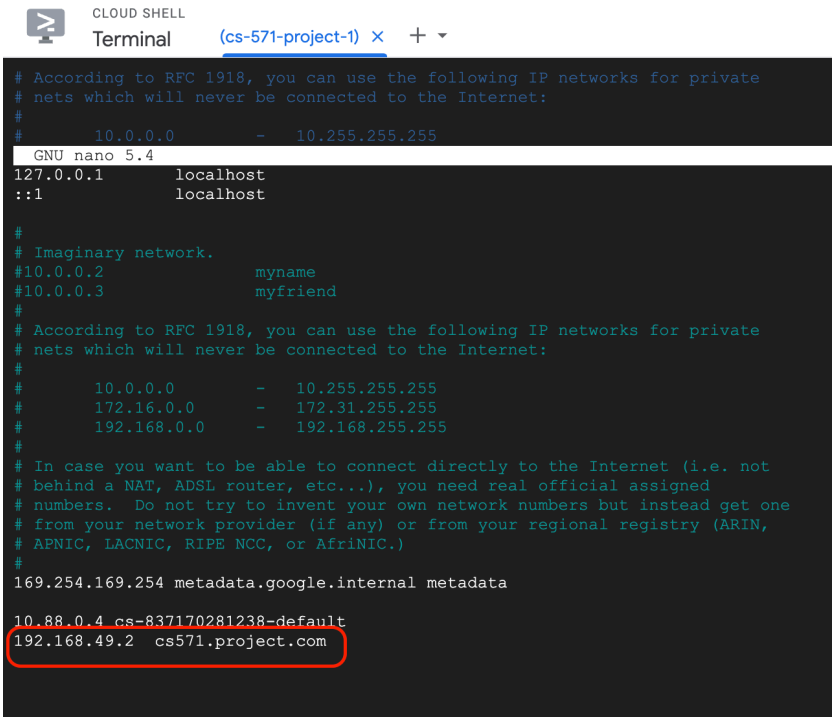
```
lbereket625@cloudshell:~ (cs-571-project-1)$ kubectl get ingress
NAME     CLASS   HOSTS                ADDRESS        PORTS   AGE
server   nginx   cs571.project.com    192.168.49.2   80      2m4s
lbereket625@cloudshell:~ (cs-571-project-1)$
```

13. Add Address to /etc/hosts ~ vim /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



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

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

```
lbereket625@cloudshell:~ (cs-571-project-1)$ lbereket625@cloudshell:~ (cs-571-project-1)$ curl cs571.project.com/studentserver/api/score student_id=11111
{"_id":"655a6b49c3a15527deed8ha", "student_id":11111,"student_name":"Bruce Lee","grade":84}
```

curl cs571.project.com/studentserver/api/score?student_id=22222

```
lbereket625@cloudshell:~ (cs-571-project-1)$ curl cs571.project.com/studentserver/api/scorerstudent_id=22222
{"_id":"657a6b78c3a15527deodo6ab", "student_id":22222,"student name": "Jackie chen", "grade":93}
```

curl cs571.project.com/studentserver/api/score?student_id=33333

```
lbereket625@cloudshell:~ (cs-571-project-1)$ curl cs571.project.com/studentserver/api/score?student_id=33333
{"_id":"605a6b49c3a15527deed5ebr", "student_id":33333,"student_name": "Jet Li","grade" :88}
```

15. On another path, you should be able to use the REST API with bookshelf application

curl cs571.project.com/bookshelf/books

AT first it might give you just a message, but try again after a while:

```
lbereket625@cloudshell:~ (cs-571-project-1)$ curl cs571.project.com/bookshelf/books
{
      "message"; "Welcome to bookshelf app! I am running inside bookshelf-deployment-54dbb67ff-zrbnb pod!"
}
```

```
lbereket625@cloudshell:~ (cs-571-project-1)$ curl cs571.project.com/bookshelf/books
[
      {
            "Book Author": "test",
            "Book Name": "123",
            "ISBN": "123",
            "id":"615b1c8d1e4f7g9h0i5j6k1l7m"
       }
]
```

Add a book

curl -X POST -d "{\"book_name\": \"artificial intelligence\",\"book_author\": \"John Doe\", \"isbn\": \"7890\" }" http://cs571.project.com/bookshelf/book

```
lbereket625@cloudshell:~ (cs-571-project-1)$ curl -X POST -d "{\"book_name\": \"artificial intelligence\",\"book_author\": \"John Doe\", \"isbn\": \"7890\" }" http://cs571.project.com/bookshelf/book
{
      "message": "Task saved successfully!"
}
```

Curl again check the added book.

curl cs571.project.com/bookshelf/books

```
lbereket625@cloudshell:~ (cs-571-project-1)$ curl cs571.project.com/bookshelf/books
[
      {
            "Book Author": "test",
            "Book Name": "123",
            "ISBN": "123",
            "id":"615b1c8d1e4f7g9h0i5j6k1l7m"
       },
      {
            "Book Author": ""John Doe",
            "Book Name": ""artificial intelligence",
            "ISBN": "7890",
            "id":"610b5c6d7e8f9g0h1i2j3k4l5t"
       }
]
```

Update a book - change the id of the book you want to update, it could be different from mine

curl -X PUT -d "{\"book_name\": \"123\",\"book_author\": \"test\", \"isbn\": \"123updated\" }" http://cs571.project.com/bookshelf/book/id

curl -X PUT -d "{\"book_name\": \"123\",\"book_author\": \"test\", \"isbn\": \"123updated\" }" http://cs571.project.com/bookshelf/book/615b1c8d1e4f7g9h0i5j6k1l7m

```
lbereket625@cloudshell:~ (cs-571-project-1)$ curl -X PUT -d "{\"book_name\": \"123\",\"book_author\": \"test\", \"isbn\": \"123updated\" }" http://cs571.project.com/bookshelf/book/6
15b1c8d1e4f7g9h0i5j6k1l7m

{
      "message": "Task updated successfully!"
}
```

Check the updated book:

curl cs571.project.com/bookshelf/books

```
lbereket625@cloudshell:~ (cs-571-project-1)$ curl cs571.project.com/bookshelf/books
[
      {
            "Book Author": "test",
            "Book Name": "123",
            "ISBN": "123updated",
            "id":"615b1c8d1e4f7g9h0i5j6k1l7m"
       },
      {
            "Book Author": ""John Doe",
            "Book Name": ""artificial intelligence",
            "ISBN": "7890",
            "id":"610b5c6d7e8f9g0h1i2j3k4l5t"
       }
]
```

Delete a book

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

curl -X DELETE cs571.project.com/bookshelf/book/615b1c8d1e4f7g9h0i5j6k1l7m

```
lbereket625@cloudshell:~ (cs-571-project-1)$ curl -X DELETE cs571.project.com/bookshelf/book/615b1c8d1e4f7g9h0i5j6k1l7m
{
        "message": "Task deleted successfully!"
}
```

Then finally curl to see what is in the current book db:

curl cs571.project.com/bookshelf/books

```
lbereket625@cloudshell:~ (cs-571-project-1)$ curl cs571.project.com/bookshelf/books
[
      {
            "Book Author": ""John Doe",
            "Book Name": ""artificial intelligence",
            "ISBN": "7890",
            "id":"610b5c6d7e8f9g0h1i2j3k4l5t"
       }
]
```