Week 11: Homework: Signature Project: Automated Essay Scoring System

Submitted By: Md Shahadat Hossain Khan (19609)

This signature project goal is to execute a full project developed in python into kubernetes infrastructure. Here I use existing "Automated Essay Scoring System" as my project and I'm going to run into GCP using kubernetes. Here is the step summary -

1. I'll create a local VM using virtual box where I'll install Ubuntu 64 bit 20 LTS OS
2. Execute our pre-built python program (flask-summary) into our new VM
3. Build, Push & Deploy
   a) Build docker image in our VM
   b) Push that image to Google Container Registry (GCR)
   c) Use that image from GCR into kubernetes to create pod / deployment

1. Install Ubuntu 20 LTS installation

Its pretty easy and straightforward. I downloaded the ISO file and create a new instance of virtual box machine (2 processor, 40 GB space, 1024 MB RAM for my VM) and port that ISO file as CD/DVD. After that I followed on-screen instruction to install Ubuntu. I'm ignoring details of this part. Anyone can search internet and found the step by step process to achieve this goal.

2. Execute our pre-built program into our new VM

This part is challenging because all things not documented in suggested project. I'll try to explain all challenging part here. I'm suggested to use https://github.com/Quan25/flask-summary project to implement. There has readme file as document to install which is pretty good but missed some information. Now we need to login into VM as our local environment. From now on local means inside VM. In our VM python (version 3) already installed, so we don't need to install python. If in your system python don't exists you have to install. The python installation is also pretty easy, so I'm not cover python installation. You can check version of installed python by following command:

```
$ python –version
or
$ python3 --version
```

Now we are going to execute "flask-summary" in our local VM. Please note that I followed instruction at https://github.com/Quan25/flask-summary but in this instruction I don't understand the usability of some sub-system. So, I ignored those sub-system to install. And I'm able to execute the project without those sub-systems.

I create a root folder for my project called "sgnprj2" and change my directory to this location. Now here is my rest of the procedure that I followed to execute "flask-summary" into my local vm -

i) I cloned the project to use locally from github by using following command -

```
$ git clone https://github.com/Quan25/flask-summary.git
```

ii) Now we have to install pip for python 3. Its an installation wrapper for python. Most of the time its installed with python.

```
$ sudo apt-get install python3-pip
```

iii) Then I installed pytorch. Please note, pytorch installation required approx 10GB+ disk space. Here is the command to install pytorch.

```
$ pip install torch==1.5.1+cu101 torchvision==0.6.1+cu101 -f
https://download.pytorch.org/whl/torch_stable.html
```

NOTE: torchvision and torch parameter value I found from internet and it works.

iv) Install Flask: Before flask installation we need to install some additional package. As of today the following command installed flask with dependency very well -

```
$ pip install git+https://github.com/jek/blinker.git
$ pip install git+https://github.com/simplejson/simplejson.git
$ pip install Flask
```

v) Install other projects which is required to run flask-summary project

```
-- pyrouge
$ pip install git+https://github.com/bheinzerling/pyrouge.git
-- pandas
$ pip install pandas
-- python3-tk
$ sudo apt-get install python3-tk
-- matplotlib version 3.2.1 [without this version I faced many issue while
execute finally]
$ pip install matplotlib==3.2.1
-- scikit-lert and scipy [it is possible to install separately but I didn't
test it]
$ pip install scikit-learn scipy
-- nltk
$ pip install nltk
-- at this stage we need to donwonload related package from nltk server by
nltk downloader. Before use nltk downloader we need to set download location
into a variable. Please note: we need to set this nltk package download
location every-time the "flask-summary" project execute. We will set it
through a special shell script file later.
$ mkdir nltk_data
$ export NLTK_DATA=/home/shkr/sgnprj2/nltk_data
$ python3 -m nltk.downloader all
-- gensim version 3.6.0 [latest version will not work]
$ pip install gensim==3.6.0
-- pytorch_pretrained_bert
$ pip install pytorch_pretrained_bert
```

vi) After that I downloaded “pretrained-bert-model” from https://s3.amazonaws.com/models.huggingface.co/bert/bert-large-uncased.tar.gz, and unzip it to a location inside my project root

vii) Now I need to point the extracted “pretrained-bert-model” location into BertParent.py in summarizer folder of “flask-summary” project. I do following changes -

```
# at the end of library declaration which is at the top portion of file
from pathlib import Path
# now find self.model and comment that line and paste following code
self.model = BertModel.from_pretrained(str(Path().resolve().parent)+'/bert-
large-uncased')
```

viii) Now I create a shell script to put together all required enviornment variable and to check if we install plain python or python3 into system to execute flask-summary program. This python or python3 command check required due to pod creation. As we planning to create pod from the image, we need to ensure that either python or python3 command available into our system. Here is our shell script (sfbu19609.sh) -

```
#!/bin/bash
cd "$(dirname "$(readlink -f "${BASH_SOURCE[0]}")")"  # cd current directory
CUR_DIR=$(pwd)
echo "Current Directory: ${CUR_DIR}"
export NLTK_DATA=${CUR_DIR}/nltk_data
echo "NLTK Data Path: ${NLTK_DATA}"
export MPLBACKEND=TKAgg
if ! command -v python &> /dev/null
then
  echo "python could not be found"
  if ! command -v python3 &> /dev/null
  then
    echo "python3 could not be found"
    exit
  fi
  shopt -s expand_aliases
  alias python='python3'
fi
#exec bash
cd ${CUR_DIR}/flask-summary
python app.py
```

Please note: I already explain I need “NLTK_DATA” variable and here is another variable [MPLBACKEND] required which hold a static value. I can’t explain the reason of this variable but without it, our program didn’t execute properly. I found this solution by surfing internet.

Finally! We completed all the step to install part. Here is my project root folder structure -

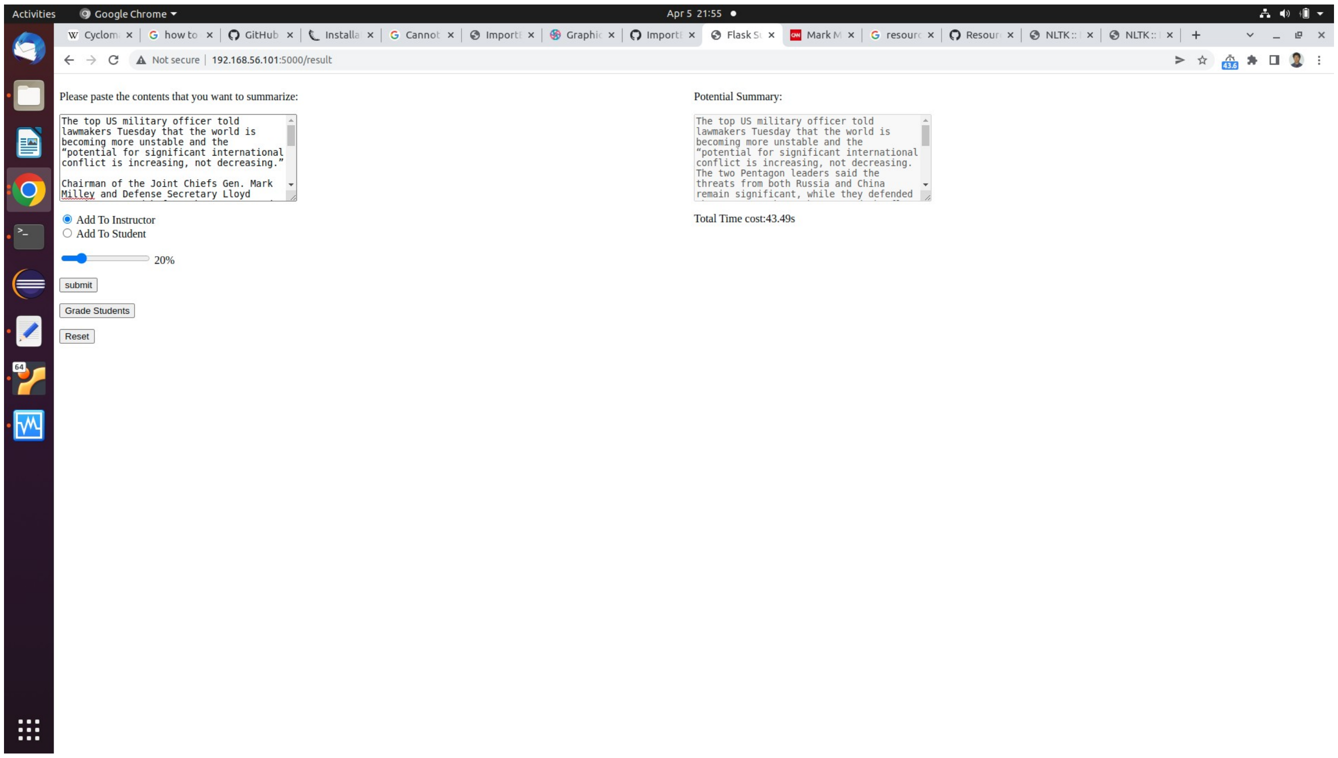
```
account.json
bert-large-uncased/
Dockerfile
flask-summary/
nltk_data/
requirements.txt
sfbu19609.sh
```

Here “account.json” required by docker image to push into GCR. “Dockerfile” and “requirements.txt” required to build docker image. I’ll put content these files into docker image build section.

At this stage we are ready to execute our project. I issue following command to run our app -

```
./sfbu19609.sh
```

Now we can visit our VM’s local IP to check our program. To do this I visit that IP into chrome from my development laptop (not at VM). And here is the screenshot of my program -



Please paste the contents that you want to summarize:

```
The top US military officer told
lawmakers Tuesday that the world is
becoming more unstable and the
“potential for significant international
conflict is increasing, not decreasing.”

Chairman of the Joint Chiefs Gen. Mark
Milley and Defense Secretary Lloyd
```

(●) Add To Instructor
( ) Add To Student

20%

Potential Summary:

```
The top US military officer told
lawmakers Tuesday that the world is
becoming more unstable and the
“potential for significant international
conflict is increasing, not decreasing.
The two Pentagon leaders said the
threats from both Russia and China
remain significant, while they defended
```

Total Time cost:43.49s

Yahoo!!! Hard part eliminated, now we will build docker image for our project and push that image into GCR and use that image from GCR to create pod into kubernetess to deploy through kubectl.

3. Build, Push & Deploy

Build docker image is pretty easy and straightforward. We need “Dockerfile” file to tell docker engine what to do and “requirements.txt” file to inform docker engine what packages we need to use.

Here is my “Dockerfile” content -

```
# set base image (host OS)
FROM python:3.8
WORKDIR /code
COPY requirements.txt .
RUN pip install -r requirements.txt
COPY bert-large-uncased/ ./bert-large-uncased
COPY flask-summary/ ./flask-summary
COPY nltk_data/ ./nltk_data
COPY sfbu19609.sh .
RUN chmod +x sfbu19609.sh
CMD ["/bin/bash", "/code/sfbu19609.sh"]
```

And my “requirements.txt” content -

```
Flask==2.1.1
gensim==3.6.0
matplotlib==3.2.1
nltk==3.7
pandas==1.4.2
pytorch-pretrained-bert==0.6.2
scikit-learn==1.0.2
scipy==1.8.0
threadpoolctl==3.1.0
torch==1.11.0
```

Here is my docker image build command -

```
$ sudo docker build -t gcr.io/signature-project2-19609-v1/falsk-summary-
sfbu .
```

Please note that every docker command we need to apply with “sudo” i.e. with root previlidge.

If anyone want to test by deploy the newly created image here is the helper commands -

```
$ sudo docker images
$ sudo docker run -d -p 5000:5000  gcr.io/signature-project2-19609-v1/falsk-
summary-sfbu
$ sudo docker ps
$ sudo docker stop -t 30 461a66b85937
$ sudo docker image rm -f <image id>
```

Please note: the image I tagged for a purpose and was documented at googld cloud registry document. First part is HTTP host of GCR, second part is the ID of my project at GCP and third part is the project-name

At this stage I push my newly created docker image to GCR. Before push it I create project at GCP and enabled “Container Registry API” and “Kubernetes Engine API”

After enable those API I authorize this project to allow push docker image remotely. To do this I created a service account key by following https://cloud.google.com/iam/docs/creating-managing-service-account-keys#creating_service_account_keys of google. The key file is downloaded into my local system which is a JSON file containing private key and public key to push docker image into GCR from local environment.

After getting key file I have to upload it into GCP to authenticate my project by using following command at gcloud shell -

```
$ gcloud auth activate-service-account --key-file=signature-project2-19609-v1.json
```

Please note: “signature-project2-19609-v1.json” file I uploaded from local which is downloaded as key file at previous step. I’m not going to share this key file in this document. As it build by google and its vary for each project.

Now come back to local VM and issued following command to push our docker image -

```
$ sudo docker login -u _json_key --password-stdin https://gcr.io < account.json
```

Please note: “account.json” is the same file we downloaded as key file. So, before apply this command we need to copy our downloaded file into our VM project location.

Now I’m ready to push my docker image into GCR. Here is the commands -

```
$ sudo docker tag a6186f2784c4 gcr.io/signature-project2-19609-v1/falsk-summary-sfbu
$ sudo docker push gcr.io/signature-project2-19609-v1/falsk-summary-sfbu
```

Owo!!! Its done with our local VM. I already push my image into GCR.

At GCP I need some file to deploy this project. Here is the contents of those files.

Content of deployment YAML [fss-deployment.yaml] file -

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: fss-deployment
  labels:
    app: fss-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: fss-deployment
  template:
    metadata:
      labels:
        app: fss-deployment
    spec:
      containers:
        - image: gcr.io/signature-project2-19609-v1/falsk-summary-sfbu
          imagePullPolicy: Always
          name: fss-deployment
          ports:
            - containerPort: 5000
          resources:
            requests:
              memory: "64Mi"
              cpu: "2048m"
            limits:
              memory: "128Mi"
              cpu: "4096m"
```

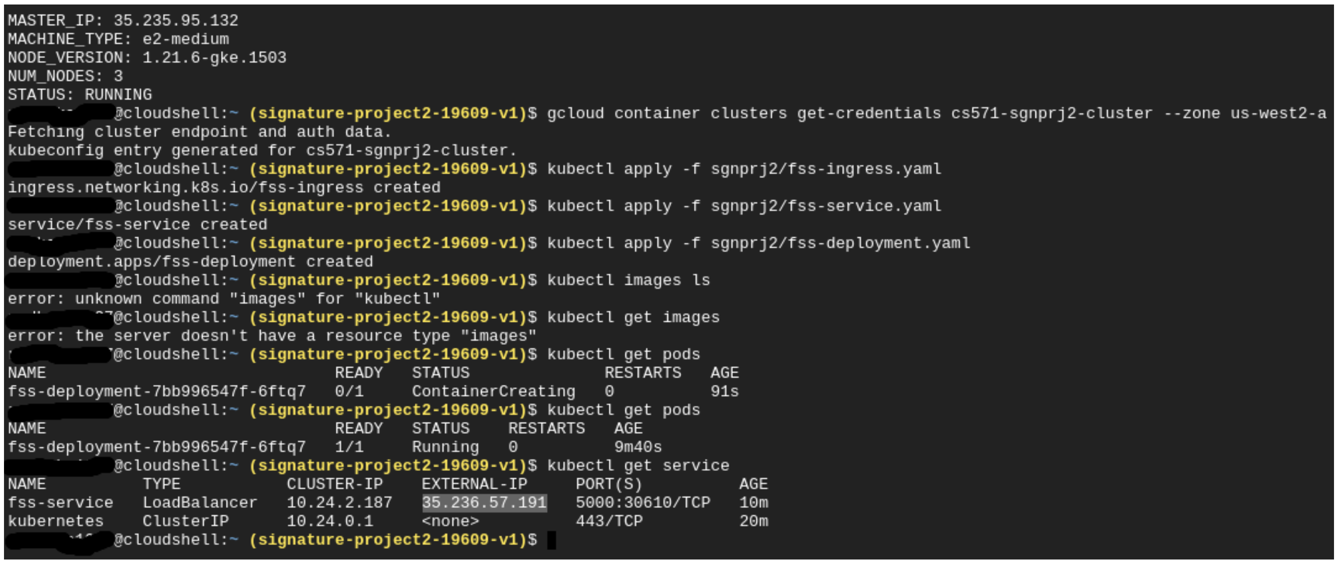
Content of service YAML [fss-service.yaml] file -

```
apiVersion: v1
kind: Service
metadata:
  name: fss-service
spec:
  type: LoadBalancer
  ports:
      # service port in cluster
    - port: 5000
      # port to contact inside container
      targetPort: 5000
  selector:
    app: fss-deployment
```

Now the commands I applied at GCP. I'm not going to explain following commands as it already explained before and used many times before -

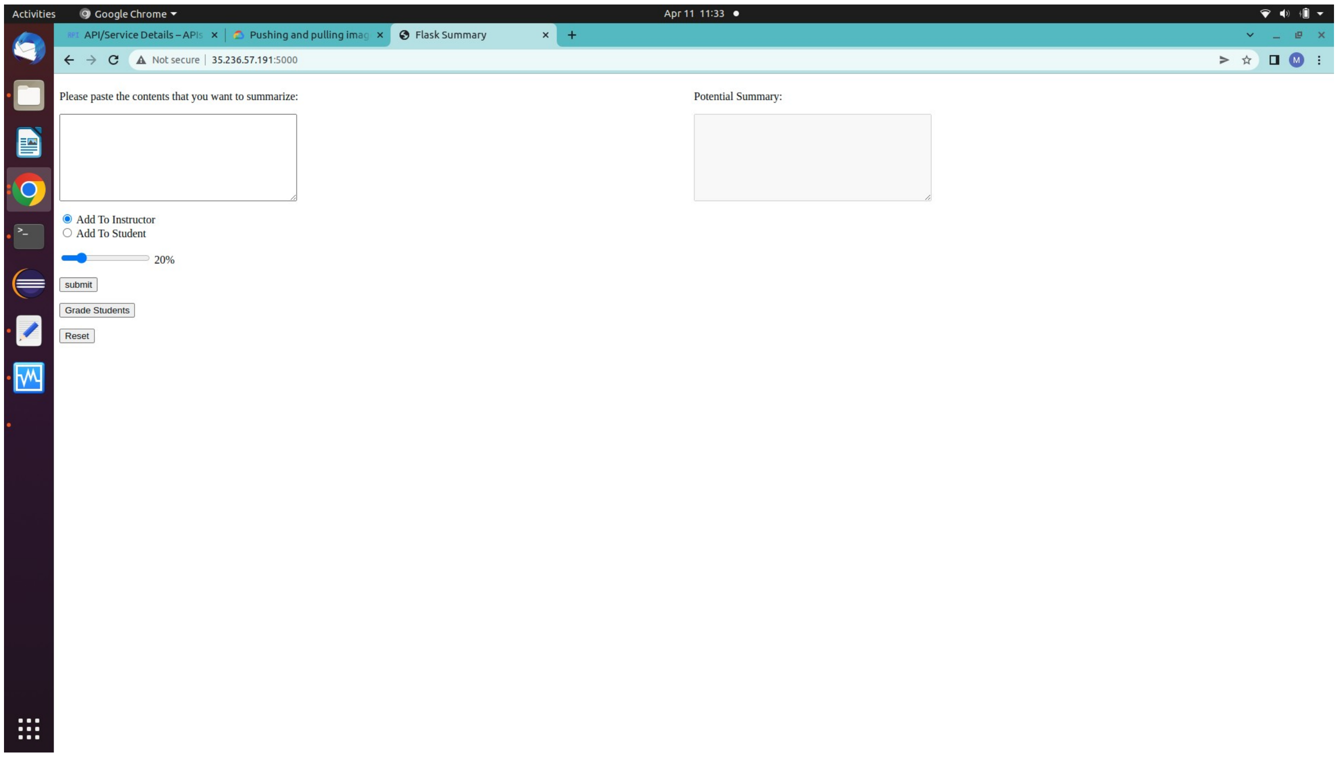
```
gcloud config set compute/zone us-west2-a
gcloud container clusters create cs571-sgnprj2-cluster
gcloud container clusters get-credentials cs571-sgnprj2-cluster --zone us-
west2-a
kubectl apply -f sgnprj2/fss-service.yaml
kubectl apply -f sgnprj2/fss-deployment.yaml
```

Response screenshot of above commands -



```
MASTER_IP: 35.235.95.132
MACHINE_TYPE: e2-medium
NODE_VERSION: 1.21.6-gke.1503
NUM_NODES: 3
STATUS: RUNNING
[redacted]@cloudshell:~ (signature-project2-19609-v1)$ gcloud container clusters get-credentials cs571-sgnprj2-cluster --zone us-west2-a
Fetching cluster endpoint and auth data.
kubeconfig entry generated for cs571-sgnprj2-cluster.
[redacted]@cloudshell:~ (signature-project2-19609-v1)$ kubectl apply -f sgnprj2/fss-ingress.yaml
ingress.networking.k8s.io/fss-ingress created
[redacted]@cloudshell:~ (signature-project2-19609-v1)$ kubectl apply -f sgnprj2/fss-service.yaml
service/fss-service created
[redacted]@cloudshell:~ (signature-project2-19609-v1)$ kubectl apply -f sgnprj2/fss-deployment.yaml
deployment.apps/fss-deployment created
[redacted]@cloudshell:~ (signature-project2-19609-v1)$ kubectl images ls
error: unknown command "images" for "kubectl"
[redacted]@cloudshell:~ (signature-project2-19609-v1)$ kubectl get images
error: the server doesn't have a resource type "images"
[redacted]@cloudshell:~ (signature-project2-19609-v1)$ kubectl get pods
NAME                              READY   STATUS              RESTARTS   AGE
fss-deployment-7bb996547f-6ftq7   0/1     ContainerCreating   0          91s
[redacted]@cloudshell:~ (signature-project2-19609-v1)$ kubectl get pods
NAME                              READY   STATUS    RESTARTS   AGE
fss-deployment-7bb996547f-6ftq7   1/1     Running   0          9m40s
[redacted]@cloudshell:~ (signature-project2-19609-v1)$ kubectl get service
NAME          TYPE           CLUSTER-IP    EXTERNAL-IP     PORT(S)          AGE
fss-service   LoadBalancer   10.24.2.187   35.236.57.191   5000:30610/TCP   10m
kubernetes    ClusterIP      10.24.0.1     <none>          443/TCP          20m
[redacted]@cloudshell:~ (signature-project2-19609-v1)$
```

And here is the screenshot of our project which is executed into GCP using kubernetes -



Please paste the contents that you want to summarize:

Potential Summary:

(●) Add To Instructor
( ) Add To Student

20%

And here is the github URL for this project -

https://github.com/mkhan-sfbu/cs571-signature-project2