**Instructions to set up the Math Web Service application in the Google Cloud Platform**

**Introduction:**

This document will talk about the instructions on how to setup the math web service in the Google Kubernetes Engine (GKE) of the Google Cloud Platform (GCP). We would first need to create a docker image of the python application. This image needs to be pushed / published to the Google Container Registry. This published image will then be deployed as a service in the GKE, where an external IP would be exposed. This IP will serve as the host of the Math Web Service application.

The detailed steps to setup this application will be as follows:

**Step 1: Prepare the requirements.txt file**

* Activate the virtual environment in which web service was developed and tested.

conda activate <YOUR\_ENVIRONMENT\_NAME>

* Create a list of all the python libraries installed in the virtual environment.

pip freeze > requirements.txt

**Step 2: Building a new docker image for the flask application**

* Create a new project folder in your local machine, say Math\_Web\_Service.
* Copy the mathApi.py to this folder.
* Copy the reqirements.txt (created in the previous step) to this folder
* Create a new file called the “Dockerfile” without any extension.
* Copy the following contents to the file.

FROM python:alpine3.7

COPY . /app

WORKDIR /app

RUN pip install -r requirements.txt

EXPOSE 5000

ENTRYPOINT [ "python" ]

CMD [ "mathApi.py" ]

**Step 3: Assuming that the docker is already installed in your machine, build the docker image of the web service application using the following command:**

* docker build –tag gcr.io/<GCP\_PROJECT\_ID>/math\_web\_service\_api

**Step 4: Push the image into Google container registry**

* docker push –tag gcr.io/<GCP\_PROJECT\_ID>/math\_web\_service\_api

**Step 5: Create a deployment configuration for the web service api. This step assumes that you have already created a kubernetes cluster in the GKE.**

Add the following contents to a new file called the app.yaml

apiVersion: apps/v1beta2  
kind: Deployment  
metadata:  
 name: math\_api  
 labels:  
 name: math\_api  
spec:  
 replicas: 1  
 selector:  
 matchLabels:  
 name: math\_api  
 template:  
 metadata:  
 name: math\_api  
 labels:  
 name: math\_api  
 spec:  
 containers:  
 - name: math\_api  
 image: gcr.io/<GCP\_PROJECT\_ID>/math\_web\_service\_api 1  
 ports:  
 - containerPort: 5000  
 resources:  
 requests:  
 memory: 256Mi  
 limits:  
 memory: 512Mi  
 env:  
 - name: DEBUG\_MODE  
 value: "1"

**Step 6: Deploy the app to the kubernetes cluster**

* kubectl apply -f app.yaml

**Step 7: Expose the port where the api needs to be available**

* kubectl expose deployment math\_api --type=LoadBalancer --port 50 --target-port 5000

**Step 8: Get the deployed application’s URL by executing the following command:**

* kubectl get services -l name=math\_api

**Step 9: On executing the previous command, both the internal and external IPs will be displayed.**

* Use the external IP to access the homepage of the application.

<External\_IP>:5000/

* Use the following URLs to perform the 3 different math web services. You may use these addresses in the Python scripts which will call these web services via a REST api call.

<External\_IP>:5000/fibonacci/

<External\_IP>:5000/ackermann/

<External\_IP>:5000/factorial/