Use This Script For LB Demo

```
#! /bin/bash
apt-get update
apt-get install -y apache2 php
apt-get install -y wget
cd /var/www/html
rm index.html -f
rm index.php -f
wget https://storage.googleapis.com/cloud-training/gcpnet/httplb/index.php
META_REGION_STRING=$(curl
"http://metadata.google.internal/computeMetadata/v1/instance/zone" -H "Metadata-Flavor:
Google")
REGION=`echo "$META_REGION_STRING" | awk -F/ '{print $4}'`
sed -i "s|region-here|$REGION|" index.php
```

AppEngine Code

main.py

```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def home():
    return "Hello, Google App Engine!"

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=8080)
```

requirements.txt

```
Flask
gunicorn
```

app.yaml

```
runtime: python39  # Specify Python runtime

entrypoint: gunicorn -b :$PORT main:app  # Use Gunicorn as WSGI server
```

```
handlers:
- url: /.*
  script: auto
```

Cloud Function Code

main.py

```
import functions_framework
import os
from google.cloud import storage
from PIL import Image
import io

# Initialize Google Cloud Storage client
storage_client = storage.Client()

# Destination bucket for processed images
DESTINATION_BUCKET = "my-processed-bucket"

@functions_framework.cloud_event
def gcs_trigger(cloud_event):
    """Triggered by a change to a Cloud Storage bucket."""
    data = cloud_event.data
    file_name = data["name"]
    bucket_name = data["bucket"]

    print(f"New file uploaded: {file_name} in bucket: {bucket_name}")

    # Process only image files
    if file_name.lower().endswith(('.png', '.jpg', '.jpeg')):
        print(f"Processing image: {file_name}")
        convert_image_to_grayscale(bucket_name, file_name)
    else:
        print("Not an image file. No action taken.")

def convert_image_to_grayscale(bucket_name, file_name):
    """Convert an image to grayscale and upload to another bucket"""
    bucket = storage_client.bucket(bucket_name)
    blob = bucket.blob(file_name)

    # Download image into memory
    image_bytes = blob.download_as_bytes()
    image = Image.open(io.BytesIO(image_bytes)).convert("L")  # Convert to grayscale

    # Save to output bucket
```

```
    output_bucket = storage_client.bucket(DESTINATION_BUCKET)
    output_blob = output_bucket.blob(f"grayscale-{file_name}")

    with io.BytesIO() as output_bytes:
        image.save(output_bytes, format="PNG")
        output_blob.upload_from_string(output_bytes.getvalue(), content_type="image/png")

    print(f"Processed image saved to: gs://{DESTINATION_BUCKET}/grayscale-{file_name}")
```

requirements.txt

```
google-cloud-storage
functions-framework
pillow
```