Automated File Copying between GCP Buckets using Cloud Run

# Overview

This guide provides a step-by-step process to create and deploy a Python application on Google Cloud Run that copies files from one GCP bucket to another. The deployment utilizes Cloud Run for the application and Cloud Scheduler to trigger the file copying operation periodically.

# Prerequisites

- Google Cloud SDK installed and configured  
- GCP project with billing enabled  
- Access to create Cloud Run services and Cloud Scheduler jobs  
- Read access to the source bucket and write access to the destination bucket

# Step-by-Step Guide

## Step 1: Create the Python Application

1. Create a Directory for Your Project:

```sh  
mkdir gcs-copy-cloud-run  
cd gcs-copy-cloud-run  
```

2. Create the Python Script:

Create a `main.py` file with the following content:

```python  
from google.cloud import storage  
from flask import Flask  
  
app = Flask(\_\_name\_\_)  
  
@app.route('/')  
def copy\_gcs\_files():  
 source\_bucket\_name = "gk1v-bucket"  
 destination\_bucket\_name = "ienv-bucket"  
  
 client = storage.Client()  
  
 source\_bucket = client.bucket(source\_bucket\_name)  
 destination\_bucket = client.bucket(destination\_bucket\_name)  
  
 # List and copy files from the source bucket to the destination bucket  
 blobs = source\_bucket.list\_blobs()  
 for blob in blobs:  
 source\_blob = source\_bucket.blob(blob.name)  
 destination\_blob = destination\_bucket.blob(blob.name)  
 destination\_blob.rewrite(source\_blob)  
  
 return "Files copied successfully"  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 app.run(host='0.0.0.0', port=8080)  
```

3. Create a Requirements File:

Create a `requirements.txt` file with the following content:

```text  
Flask  
google-cloud-storage  
```

4. Create a Dockerfile:

Create a `Dockerfile` with the following content:

```Dockerfile  
# Use the official Python image.  
FROM python:3.10  
  
# Set the working directory to /app  
WORKDIR /app  
  
# Copy the current directory contents into the container at /app  
COPY . /app  
  
# Install any needed packages specified in requirements.txt  
RUN pip install --no-cache-dir -r requirements.txt  
  
# Make port 8080 available to the world outside this container  
EXPOSE 8080  
  
# Run main.py when the container launches  
CMD ["python", "main.py"]  
```

## Step 2: Build and Deploy the Container to Cloud Run

1. Build the Container:

```sh  
gcloud builds submit --tag gcr.io/YOUR\_PROJECT\_ID/gcs-copy-cloud-run  
```

Replace `YOUR\_PROJECT\_ID` with your GCP project ID.

2. Deploy the Container to Cloud Run:

```sh  
gcloud run deploy gcs-copy-cloud-run \  
 --image gcr.io/YOUR\_PROJECT\_ID/gcs-copy-cloud-run \  
 --platform managed \  
 --region YOUR\_REGION \  
 --allow-unauthenticated  
```

Replace `YOUR\_PROJECT\_ID` and `YOUR\_REGION` with your GCP project ID and the desired region.

## Step 3: Set Up Cloud Scheduler to Trigger the Cloud Run Service

1. Create a Cloud Scheduler Job:

Go to the Cloud Scheduler page in the Google Cloud Console and create a new job.

2. Configure the Scheduler:

- \*\*Frequency:\*\* Set the frequency to run the job (e.g., daily at midnight).  
- \*\*Target:\*\* HTTP  
- \*\*URL:\*\* The URL of your Cloud Run service.  
- \*\*HTTP Method:\*\* GET

### Example Cloud Scheduler Configuration

- \*\*Frequency:\*\* `0 0 \* \* \*` (every day at midnight)  
- \*\*URL:\*\* The URL of your Cloud Run service, which you can find in the Cloud Run console after deployment.  
- \*\*HTTP Method:\*\* GET

## Step 4: Verify the Setup

1. Trigger the Cloud Run Service Manually:

Test the service by manually sending an HTTP GET request to the Cloud Run URL.

```sh  
curl https://YOUR\_CLOUD\_RUN\_URL  
```

2. Check Logs:

Check the logs in the Google Cloud Console under Cloud Run to verify that the files are being copied successfully.

# Summary

1. \*\*Create a Python application\*\* that copies files from one GCS bucket to another.  
2. \*\*Containerize the application\*\* using Docker.  
3. \*\*Deploy the container\*\* to Cloud Run.  
4. \*\*Set up Cloud Scheduler\*\* to trigger the Cloud Run service at the desired intervals.  
5. \*\*Verify the setup\*\* to ensure files are copied correctly.

This setup will automate the process of copying files from the source bucket to the destination bucket using Cloud Run and Cloud Scheduler. The Cloud Run service will use the default service account of its environment, which should have the necessary permissions for the operations.  
Feel free to reach out if you have any questions or need further assistance!