Set Up an App Dev Environment on Google Cloud: Challenge Lab

Task 1. Create a bucket

You need to create a bucket called `Bucket Name` for the storage of the photographs. Ensure the resource is created in the `REGION` region and `ZONE` zone.

Task 2. Create a Pub/Sub topic

Create a Pub/Sub topic called `Topic Name` for the Cloud Run Function to send messages.

Task 3. Create the thumbnail Cloud Run Function

Create the function

Create a Cloud Run Function `Cloud Run Function Name` that will to create a thumbnail from an image added to the `Bucket Name` bucket.

Ensure the Cloud Run Function is using the **Cloud Run function** environment (which is 2nd generation). Ensure the resource is created in the `REGION` region and `ZONE` zone.

1. Create a Cloud Run Function (2nd generation) called `Cloud Run Function Name` using `Node.js 22`.

Note: The Cloud Run Function is required to execute every time an object is created in the bucket created in Task 1. During the process, Cloud Run Function may request permission to enable APIs or request permission to grant roles to service accounts. Please enable each of the required APIs and grant roles as requested.

2. Make sure you set the **Entry point** (Function to execute) to `Cloud Run Function Name` and **Trigger** to `Cloud Storage`.

3. Add the following code to the `index.js`:

```
const functions = require('@google-cloud/functions-framework');
const { Storage } = require('@google-cloud/storage');
const { PubSub } = require('@google-cloud/pubsub');
const sharp = require('sharp');

functions.cloudEvent('', async cloudEvent => {
  const event = cloudEvent.data;

  console.log(`Event: ${JSON.stringify(event)}`);
  console.log(`Hello ${event.bucket}`);

  const fileName = event.name;
  const bucketName = event.bucket;
  const size = "64x64";
  const bucket = new Storage().bucket(bucketName);
  const topicName = "";
  const pubsub = new PubSub();

  if (fileName.search("64x64_thumbnail") === -1) {
    // doesn't have a thumbnail, get the filename extension
    const filename_split = fileName.split('.');
    const filename_ext = filename_split[filename_split.length -
1].toLowerCase();
    const filename_without_ext = fileName.substring(0, fileName.length
- filename_ext.length - 1); // fix sub string to remove the dot
```

```
    if (filename_ext === 'png' || filename_ext === 'jpg' ||
filename_ext === 'jpeg') {
      // only support png and jpg at this point
      console.log(`Processing Original:
gs://${bucketName}/${fileName}`);
      const gcsObject = bucket.file(fileName);
      const newFilename =
`${filename_without_ext}_64x64_thumbnail.${filename_ext}`;
      const gcsNewObject = bucket.file(newFilename);

      try {
        const [buffer] = await gcsObject.download();
        const resizedBuffer = await sharp(buffer)
          .resize(64, 64, {
            fit: 'inside',
            withoutEnlargement: true,
          })
          .toFormat(filename_ext)
          .toBuffer();

        await gcsNewObject.save(resizedBuffer, {
          metadata: {
            contentType: `image/${filename_ext}`,
          },
        });

        console.log(`Success: ${fileName} → ${newFilename}`);

        await pubsub
          .topic(topicName)
          .publishMessage({ data: Buffer.from(newFilename) });

        console.log(`Message published to ${topicName}`);
      } catch (err) {
        console.error(`Error: ${err}`);
      }
    } else {
      console.log(`gs://${bucketName}/${fileName} is not an image I can
handle`);
    }
  } else {
    console.log(`gs://${bucketName}/${fileName} already has a
thumbnail`);
  }
});
```

Copied!

content_copy

4. Add the following code to the `package.json`:

```
{
 "name": "thumbnails",
 "version": "1.0.0",
 "description": "Create Thumbnail of uploaded image",
 "scripts": {
   "start": "node index.js"
 },
 "dependencies": {
```

```
    "@google-cloud/functions-framework": "^3.0.0",
    "@google-cloud/pubsub": "^2.0.0",
    "@google-cloud/storage": "^6.11.0",
    "sharp": "^0.32.1"
  },
  "devDependencies": {},
  "engines": {
    "node": ">=4.3.2"
  }
}
```

Copied!

content_copy

Note: If you get a permission denied error stating it may take a few minutes before all necessary permissions are propagated to the Service Agent, wait a few minutes and try again. Ensure you have the appropriate roles (Eventarc Service Agent, Eventarc Event Receiver, Service Account Token Creator, and Pub/Sub Publisher) assigned to the correct service accounts.

Test the function

- Upload a PNG or JPG image of your choice to the `Bucket Name` bucket.
  Note: Alternatively, download this image `https://storage.googleapis.com/cloud-training/gsp315/map.jpg` to your machine. Then, upload it to the bucket.

  You will see a thumbnail image appear shortly afterwards (use **REFRESH** on the bucket details page).

  After you upload the image file, you can click to check your progress below. You do not need to wait for the thumbnail image to be created.

  Optional: If the function deployed successfully and you do not see the thumbnail image in the bucket, you can check that the **Triggers** tab displays the trigger information that you previously provided for the function, which may not have saved correctly if you previously encountered errors. If you do not see the Cloud Storage trigger in the **Triggers** tab of the function, you can recreate the trigger (see the documentation page titled Create a trigger for services), and then upload a new file again to test again (refresh the page after adding a new file).

Task 4. Remove the previous cloud engineer

You will see that there are two users defined in the project.

- One is your account (`Username 1` with the role of Owner).
- The other is the previous cloud engineer (`Username 2` with the role of Viewer).
    1. Remove the previous cloud engineer's access from the project.