**ASSESSMENT AND INTERNAL VERIFICATION FRONT SHEET (Individual Criteria)**

**(Note: This version is to be used for an assignment brief issued to students via Classter)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Course  Title | | B.Sc. (Hons.) Software Development | | | Lecturer Name & Surname | Ryan AttardDavid Deguara | | |
| Unit Number & Title | | | [ITSFT-606-1620-Programming for the Cloud](https://moodle.mcast.edu.mt/course/view.php?id=506) | | | | | |
| Assignment Number, Title / Type | | | Principles of Programming for the cloud | | | | | |
| Date Set | | |  | Deadline Date | 23/5/2022 | | | |
| Student Name | Kyle Cachia | | | ID Number | 0421004L | | Class / Group | SWD 6.3B |

|  |  |  |
| --- | --- | --- |
| Assessment Criteria | | Maximum Mark |
| **KU1.1** | Identify the benefits gained from using cloud services in a given scenario | 5 |
| **KU1.2** | Differentiate among a number of cloud services that could offer a solution to the same problem | 5 |
| **SE1.3** | Compare and Recommend other solutions (e.g. other cloud service providers) to the one you chose and justify your arguments | 10 |
| **KU2.2** | Defend your decision in choosing a specific data storage solution | 5 |
| **aa 3.2** | Inspect and examine how CDN can help with your application | 7 |
| **Aa3.3** | Use Cache Services or other related technologies to make more efficient use of the resources available | 7 |
| **KU 4.5** | Indicate what can be done in case of a data disaster | 5 |
| Total Mark |  | 44 |

|  |
| --- |
| **Notes to Students:** |
| * This assignment brief has been approved and released by the Internal Verifier through Classter. * Assessment marks and feedback by the lecturer will be available online via Classter (<Http://mcast.classter.com>) following release by the Internal Verifier * Students submitting their assignment on Moodle/Turnitin will be requested to confirm online the following statements:   **Student’s declaration prior to handing-in of assignment**   * I certify that the work submitted for this assignment is my own and that I have read and understood the respective Plagiarism Policy   **Student’s declaration on assessment special arrangements**   * I certify that adequate support was given to me during the assignment through the Institute and/or the Inclusive Education Unit. * I declare that I refused the special support offered by the Institute. |

**Assignment Guidelines**

Read the following instructions carefully before you start the assignment. If you do not understand any of them, ask your invigilator.

* This assignment is a TCA assignment.
* Copying from other students is **Strictly Prohibited** and will be penalised according to disciplinary procedures.
* Internet/ChatCPT/PPTs may be used BUT NOT social media websites like Teams, Facebook, etc. Be careful about chatbots – do not rely heavily on them as the code provided will always require some amendment. Code which does not work or where you do not prove that it works won’t be given marks.
* Duration of assignment is 2.5 hours
* Submission of this assignment is done by answering the questions here and re-upload the document again on vle.
  + <https://vle.mcast.edu.mt/mod/assign/view.php?id=51823>
  + IEU: <https://vle.mcast.edu.mt/mod/assign/view.php?id=51824>

The questions in this assignment will have to be answered in relation to the following scenario:

I am building a image-rendering application which accepts image files from the users through a portal and essentially other processes such as metadata extraction and thumbnail generation kick in immediately after. The following are some characteristics you have to keep in mind

1. Images are uploaded through a single-page portal
2. And stored on some server in Finland;
3. An entry is saved in a cost-effective database that allows rapid development;
4. Process to extract metadata and generate thumbnails starts. The serverless process needs to run separately from that running the portal which was taking care of the upload. It should be scalable and support an event-driven architecture;
5. Thumbnail generated will be held on some file storage/bucket in Netherlands;
6. Another entry related to the one holding the image identification number should be inserted/updated containing the metadata extracted and uri to the thumbnail image;

***KU1.1*** - Identify the benefits gained from using cloud services in a given scenario (10 minutes)

1. In the scenario above, what GOOGLE cloud service would be the best fit for step number 3?

Google Cloud Firestore [2]

1. Can you prove (by means of a screenshot/s) that the choice you made in (a) is really cost-effective compared to other solutions? [Use the legacy version](https://cloud.google.com/products/calculator-legacy). Take the minimal requirements with no added features to compare and a 1-10GB storage and perhaps a 1000 requests per class operation per day assumption [3]

***A screenshot of a computer

AI-generated content may be incorrect.***

***KU1.2*** - Differentiate among a number of cloud services that could offer a solution to the same problem (10 minutes)

1. If I would like to make use of Redis cache in my website, what solutions are available on Google Cloud Portal? Name two, while also noting the exact name of any cloud technology used in both solutions [2]:

Google Cloud offers two Redis-based solutions. The first is Cloud Memorystore for Redis, a fully managed, scalable in-memory data store built on open-source Redis. It is optimized for performance and designed to integrate directly with other GCP services. The second option is self-managed Redis on Google Compute Engine, where Redis is manually installed and configured on a virtual machine. This approach uses GCE’s infrastructure and gives full control over the Redis instance, including OS, versioning, and custom configuration, but requires ongoing maintenance.

1. Which is the cheapest, if minimal specifications are selected and an approximate 1000ms per request is needed and there will only be 1000 requests per month max? Prove it with a screenshot. [Use the legacy version](https://cloud.google.com/products/calculator-legacy) [3]

For minimal usage — such as 1 GB of memory and about 1,000 requests per month — Cloud Memorystore for Redis is generally cheaper and more efficient. It removes the overhead of managing a VM and includes automated patching, failover, and scaling. In contrast, a Redis instance on Compute Engine would require a small always-on VM (like an e2-micro), which, despite being affordable, adds baseline cost for the compute resource alone.

KU2.2 - Defend your decision in choosing a specific data storage solution (10 minutes)

To answer this question you need approximate figures from [Use the legacy version](https://cloud.google.com/products/calculator-legacy) calculator.

I am building a image-rendering application which accepts image files from the users, uploaded in Finland and processed to extract metadata and generating thumbnails. When this process is over a transfer of these files will happen to a storage in Netherlands, so the files needed to be transferred in a bucket in Netherlands eventually. It is estimated that every year a total of 12GB of data are transferred from Finland to Netherlands. It is estimated that the current total size of files this year remaining in Finland will be about 12 Terabytes. It is also estimated that to reach these values there are around 2 million reads and 1 million writes on a monthly basis.

1. Best tier-option (among Standard/Nearline/Coldline) to store the end product(s) would be: Standard [1]
2. 1 screenshot of the pricing calculator including the selected inputs and the estimate pricing concluded [4; 3 marks will be given if all inputs are correct but the storage class selected and specified in (a) is incorrect]:

A screenshot of a computer

AI-generated content may be incorrect.

***KU4.5*** - Indicate what can be done in case of a data disaster [10 minutes]

In step 2 above in the intro scenario, a file storage is required. Therefore create a bucket [Standard/ Uniform / multi-region/ public access/ allUsers is to be granted]. While doing so…

What can be done in the bucket to ensure robust data management while allowing for retrieval of previous versions of objects in case of accidentally overwriting of data.

Answer must have 2 screenshots proving how configuration was done and the resulting files showing multiple versions [2.5/2.5]

In the scenario, step 2 requires a file storage solution on Google Cloud. To ensure robust data management and recoverability in case of a data disaster — such as accidental overwriting or deletion — I created a **Google Cloud Storage bucket** with the following configuration:

* **Storage Class**: Standard (to support frequent access)
* **Location**: Multi-region – Europe (for redundancy across regions such as Finland and the Netherlands)
* **Access Control**: Uniform
* **Public Access**: Granted to allUsers with the "Storage Object Viewer" role, allowing public read access (as required)

To protect against data loss, I **enabled Object Versioning** on the bucket. This feature keeps **older versions of files** even after they are updated or replaced. As a result, if a file is overwritten by mistake, I can restore a previous version using the Google Cloud Console or CLI. This significantly reduces the risk of permanent data loss due to user error or system malfunction.

Screenshot 1 (Bucket):

A screenshot of a computer

AI-generated content may be incorrect.

Screenshot 2 (Object Versions):

A screenshot of a computer

AI-generated content may be incorrect.

AA3.3 - Use Cache Services or other related technologies to make more efficient use of the resources available [60 minutes]

In this task you are going to use a Cloud Event Function to implement step 4 from the intro scenario i.e. you are going to create and deploy a cloud event function on your cloud account. Steps suggested:

In task ku4.5 (if attempted) you should have created a cloud bucket. Use that cloud bucket in this task as well;

Create a cloud function that extracts metadata from a jpg/png which you place in that bucket and make sure you output size, dimensions and filename of that image using Console.WriteLine or any logging statements; A sample screenshot of what is closely expected as a result is this (copied and pasted from the Logs tab of the Cloud Functions page):



Figure 1 - 3.3 a

Note: in the code/logs you must write your name together with the rest of the details which should be dynamically extracted;

Note: See Appendix A where part of the function has been given to you. So what’s left is generation of the function signature, installation of needed packages and deployment;

1. Requirement 1: The cloud function must be triggered automatically when you drag-and-drop the image onto the bucket ONLY; therefore appropriate function signature will be checked [3];
2. Requirement 2: It must be deployed on your google account and tested. Therefore proof (see *Figure 1- 3.3a*) that it works by means of a screenshot, must be taken and pasted here. Any details missing or not close to the screenshot above will result in loss of the marks. [4];

Answer must have:

1. Screenshot of working function code (entire file including namespaces)/ you should upload even the cs file. Note: code generated by *chatgpt* does not automatically imply that it is perfectly working:
2. Screenshot of Logs proving that it worked showing your name (as in *Figure1-3.3a*):

***AA3.2 -*** Inspect and examine how CDN can help with your application [20 minutes]

Part 1:

Enable CDN on the bucket you should have created in Task KU4.5 and used in AA3.3, while also creating a static ip address [screenshot needed] and a load balancer [screenshot needed] to help in retrieving the resource quicker. When you are ready paste the url here, which should include a static ip address and when clicked, it should show the original image file (named <yourname>.jpg or png)you uploaded in your bucket and a screenshot confirming that you have created this ip address in console.cloud.google.com [5]

Screenshot1 (of configuration settings before clicking CREATE of the static ip address) [1]:

Screenshot2 (of configuration settings while in the Load Balancer creation page *Review and Finalize*- where assessor can see Frontend and Backend configuration settings) [1]:

url (to the image which have a static ip address)[1]:

if url opens image you get an extra 2 marks; if the url pasted in not what is expected you get 0 marks. Note: Once you create everything, it all takes a few seconds before actually starts working, so be patient.

Part 2:

Paste a screenshot of the Logs (from Logs Explorer) related to loading the *yourname.jpg* from the CDN clearly proving that loading from CDN takes much less time [2]. The image filename must be your name!

SE1.3 - Compare and Recommend other solutions (e.g. other cloud service providers) to the one you chose and justify your arguments [40 minutes]

As you probably know there exist Http Cloud Functions and PubSub Cloud Functions. Implement an Http Cloud Function as an alternative to the one implemented in AA3.3 to perform step 5 in the intro scenario i.e. Thumbnail generation.

Note: Thumbnail generation code is given to you in Appendix B. Only the first method is missing the implementation.

Once implemented deploy the function, make sure a message is returned and test at least once.

Answer must have:

1. Missing Code Screenshot + entire Function.cs file uploaded on VLE:
2. Link to http function including parameters:
3. Link to original image testing the function with:
4. Link to thumbnail image which is the result of a working function:
5. Screenshot with logs containing your name (as in Figure 3-1.3a).
6. Screenshot proving that a thumbnail image was created (as in Figure 3-1.3b).

Submit answer here:

Rubric

* Missing code/code pasted which meets the http-function signature and is correct [3]
* Logs proving that the methods/function are running [2], similar to this: 

Figure 2-1.3a

* Proof that it works by [5]
  + providing thumbnail image (link),
  + A working link to the http-function which your assessor may click to verify that it works (while even passing a random filename and bucket Name)
  + A screenshot like this with an image and its thumbnail (filename: use your name and the other details should be clearly visible)

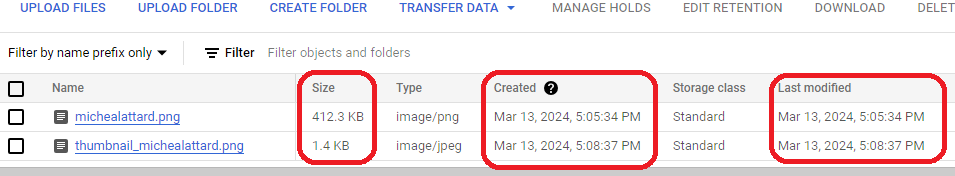


Figure 3-1.3b

Appendix A

This is an snippet of code which extracts metadata by downloading the image from the bucket. You still need to implement the method that receives the request and orchestrates everything.

using (MemoryStream memoryStream = new MemoryStream())

            {

                Google.Cloud.Storage.V1.StorageClient storageClient = await Google.Cloud.Storage.V1.StorageClient.CreateAsync();

                Task t= storageClient.DownloadObjectAsync(data.Bucket, data.Name, memoryStream);

                t.Wait();

                using (var image = SixLabors.ImageSharp.Image.Load(memoryStream.ToArray()))

                {

                    int width = image.Width;

                    int height = image.Height;

                    Console.WriteLine($"Ryan Attard: Image dimensions - Width: {width}, Height: {height}");

                    Console.WriteLine($"Ryan Attard: Size - {memoryStream.Length} bytes");

                    Console.WriteLine($"Ryan Attard: Filename - {data.Name}");

                }

            }

Figure 4- Code which extracts metadata

Note: You need to install packages *Google.Cloud.Storage.V1* and *SixLabors.ImageSharp*

Appendix B

The following is code which generates a thumbnail, and uploads it back into the same bucket with an added prefix. You need to add the method that is triggered and orchestrates the feature.

Note: You also need to install packages *Google.Cloud.Storage.V1* and *SixLabors.ImageSharp*

using Google.Cloud.Functions.Framework;

using Microsoft.AspNetCore.Http;

using System.Threading.Tasks;

using Google.Cloud.Storage.V1;

using SixLabors.ImageSharp;

using SixLabors.ImageSharp.Processing;

using System.IO;

using System.Net;

using SixLabors.ImageSharp.Formats.Jpeg;

using System;

namespace ThumbnailHttp{

public class ThumbnailFunction : IHttpFunction

{

    /// <summary>

    /// Logic for your function goes here.

    /// </summary>

    /// <param name="context">The HTTP context, containing the request and the response.</param>

    /// <returns>A task representing the asynchronous operation.</returns>

    public async Task HandleAsync(HttpContext context)

    {

         //missing code

    }

        private async Task GenerateThumbnailAndUploadAsync(string fileName, string bucketName)

        {

            // Download the original image from the bucket

            Console.WriteLine($"Ryan Attard: Downloading image {fileName} from {bucketName}...");

            byte[] originalImageBytes = await DownloadImageAsync(fileName, bucketName);

            Console.WriteLine($"Ryan Attard: Generating Thumbnail for {fileName}...");

            // Generate a thumbnail

            byte[] thumbnailBytes = GenerateThumbnailAsync(originalImageBytes);

            // Upload the thumbnail back to the bucket

            Console.WriteLine($"Ryan Attard: Uploading thumbnail back to {bucketName}...");

            await UploadImageAsync($"thumbnail\_{fileName}", bucketName, thumbnailBytes);

        }

        private byte[] GenerateThumbnailAsync(byte[] originalImageBytes)

        {

            using (var originalImageStream = new MemoryStream(originalImageBytes))

            using (var image = Image.Load(originalImageStream))

            {

                image.Mutate(x => x.Resize(new ResizeOptions

                {

                    Size = new Size(100, 100), // Set the desired thumbnail size

                    Mode = ResizeMode.Max

                }));

                using (var thumbnailStream = new MemoryStream())

                {

                    // Save the thumbnail to the MemoryStream

                    image.Save(thumbnailStream, new JpegEncoder()); // Use the appropriate encoder for your image format

                    // Return the bytes of the thumbnail

                    return thumbnailStream.ToArray();

                }

            }

        }

        private async Task<byte[]> DownloadImageAsync(string fileName, string bucketName)

            {

                using (System.IO.MemoryStream memoryStream = new System.IO.MemoryStream())

                {

                    var storageClient = await Google.Cloud.Storage.V1.StorageClient.CreateAsync();

                    await storageClient.DownloadObjectAsync(bucketName, fileName, memoryStream);

                    return memoryStream.ToArray();

                }

            }

        private async Task UploadImageAsync(string newFileName, string bucketName, byte[] imageBytes)

        {

            var storageClient = await Google.Cloud.Storage.V1.StorageClient.CreateAsync();

            using (System.IO.MemoryStream memoryStream = new System.IO.MemoryStream(imageBytes))

            {

                storageClient.UploadObject(bucketName, newFileName, "image/jpeg", memoryStream);

            }

        }

    }

}