**Sub-task 1 - Create a static website**

1. Create a static website. Feel free to do anything you like or use example site from attachments, but keep in mind that the main goal is to have a lightweight folder with multiple files in it:
   * a couple if interlinked HTMLs or an HTML page with some CSS styles is enough
   * the site should not require any additional runtime environment like JVM or Node
   * no backend is required
   * you’ll have several other tasks dedicated to creation of a fully functioning web-application in the modules 3-8
   * no heavy media resources (like large images/animations/videos) are recommended – you’ll have to upload the site to AWS multiple times
2. Create an S3 bucket which name doesn't include uppercase characters, includes your full name, and begins with a letter. Recommendation – choose a name generic enough so that the bucket may be reused for developing a web application later. (This bucket will be referred as 'bucket1')
3. Copy the static website from step 1 to the bucket1 using AWS CLI and named profile with appropriate permissions from the previous module.

A screen shot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

1. Enable static website hosting on bucket1.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

1. Content of your site should be available via website endpoint of the bucket1. For that you should add appropriate bucket policy. To add bucket policy you need to give your user permissions to access-analyzer.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Sub-task 2 - Set up replication**

1. Enable versioning for the bucket1.

A screenshot of a computer

AI-generated content may be incorrect.

1. Create another S3 bucket which name doesn't include uppercase characters, includes your full name, and begins with a letter. (This bucket will be referred as 'bucket2')

A screenshot of a computer

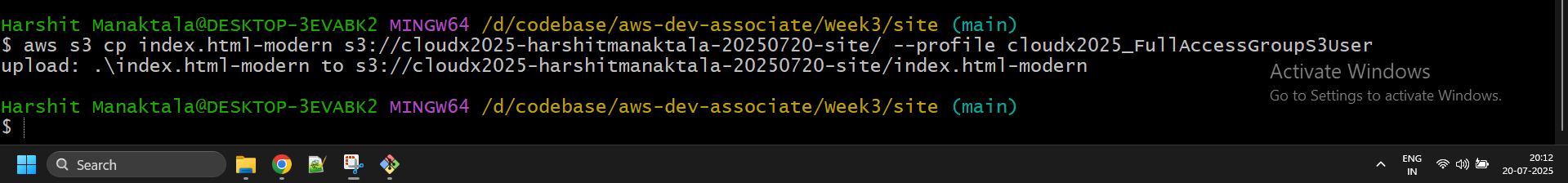
AI-generated content may be incorrect.

1. Enable cross-region replication for the original bucket1 to bucket2 (! Note: please create a new role when enabling replication, it may not work properly with existing one)

A screenshot of a computer

AI-generated content may be incorrect.

1. Upload a new file to the bucket1.



1. Verify that this file is present in bucket2. Most objects replicate within 15 minutes, but sometimes replication can take longer.

A screenshot of a computer

AI-generated content may be incorrect.

**Sub-task 3 - Play with versioning**

1. Upload 2-3 files to the bucket2. Make some changes to these files so that the bucket contains 2 (or more) versions of at least one file.



1. Using AWS CLI, get the latest version of a specific file.

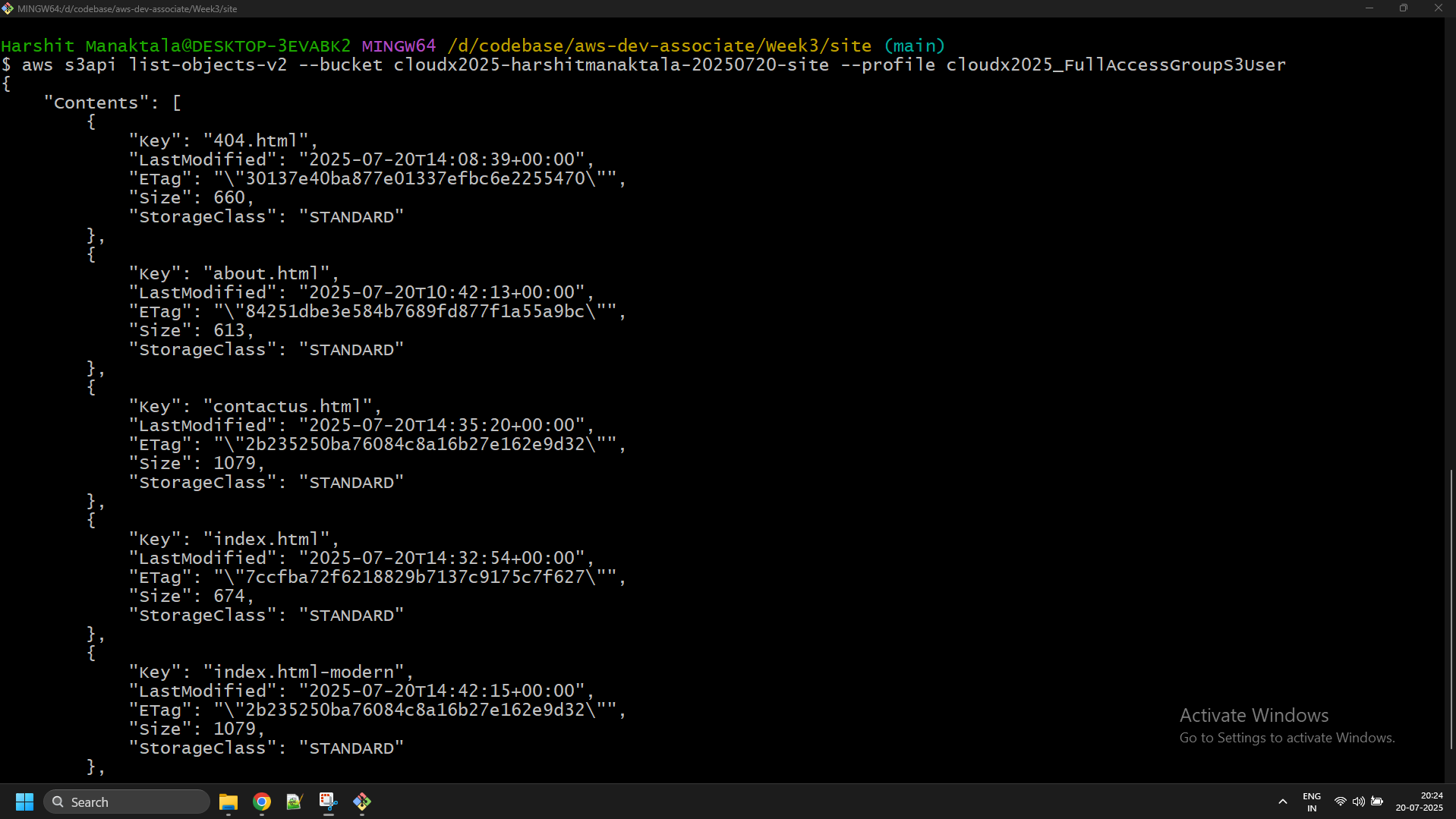
A screen shot of a computer

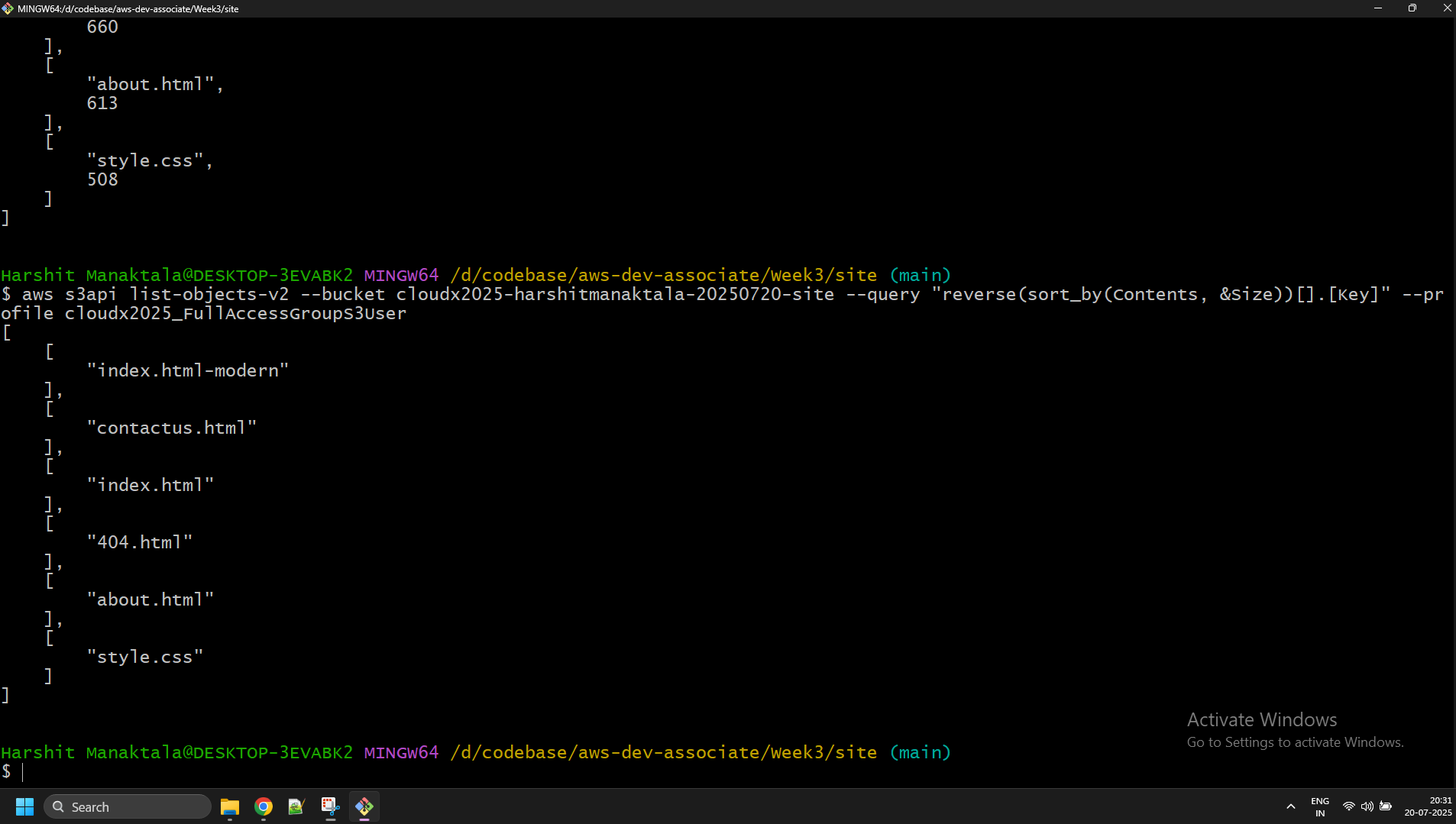
AI-generated content may be incorrect.

1. *Optional: write a script to get the latest version of a specific file no newer than a given date. You are free to use Bash or BAT or use the AWS SDK for any programming language.*

**Sub-task 4 - Practice more AWS CLI hacking and play with permissions**

1. Using AWS CLI list all the objects in the bucket1. In the response, you'll see a lot of additional data for each object, play with the "--query" parameter to filter out only S3 object keys from the response.





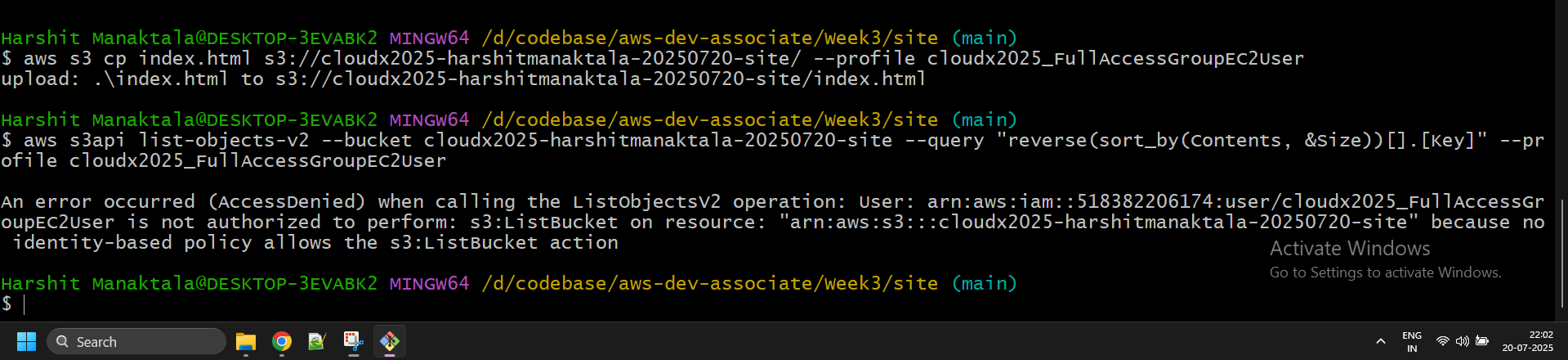
A screen shot of a computer

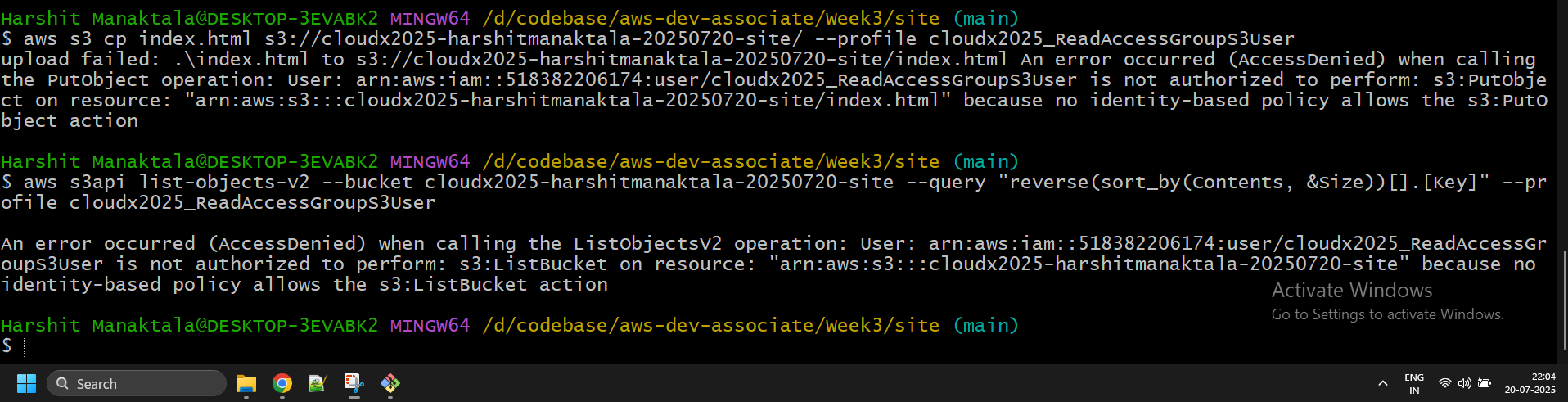
AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

1. Using different users from module 2, try to execute the following commands via AWS CLI:
   * upload new file to the S3 bucket
   * list all the objects in the S3 bucket





A screen shot of a computer

AI-generated content may be incorrect.

1. Observe the results.
2. *Optional: play with the "--output" parameter and list all the objects in the S3 bucket with their size in a "human-readable" format as a table, for example:*

| **key** | **size** |
| --- | --- |
| index.html | 17094 |
| assets/style.css | 765 |
| pictures/background.png | 1017005 |

**Sub-task 5 - Configure pre-signed url and CORS**

1. Verify that public access is blocked for bucket2

A screenshot of a computer

AI-generated content may be incorrect.

1. Share any file in bucket2 by assigning a pre-signed url for a defined amount of time.

A screenshot of a computer

AI-generated content may be incorrect.

1. Verify that the file is accessible in browser. After the defined amount of time passes, verify that the file is no longer accessible.

A screenshot of a computer

AI-generated content may be incorrect.

After 2 minutes (expiry)

A screenshot of a computer

AI-generated content may be incorrect.

1. In your static website add a js function to load resource (image, json or html file) from bucket2. There is such a function in the attachments, but you can implement your own if needed or use the following snippet(make sure to replace <PLACEHOLDER\_FOR\_S3\_RESOURCE> with the actual url to your resource from S3 bucket):

***A script example for loading json resource:***

window.onload = function () {

readJsonFile(<PLACEHOLDER\_FOR\_S3\_RESOURCE>, function (QuoteJson) {

let QuoteObj = JSON.parse(QuoteJson);

document.getElementById("Quote").innerHTML =

"<i>" + QuoteObj.message + "</i>"

});

}

function readJsonFile(file, callback) {

let rawFile = new XMLHttpRequest();

rawFile.overrideMimeType("application/json");

rawFile.open("GET", file, true);

rawFile.onreadystatechange = function () {

if (rawFile.readyState === 4 && rawFile.status === 200) {

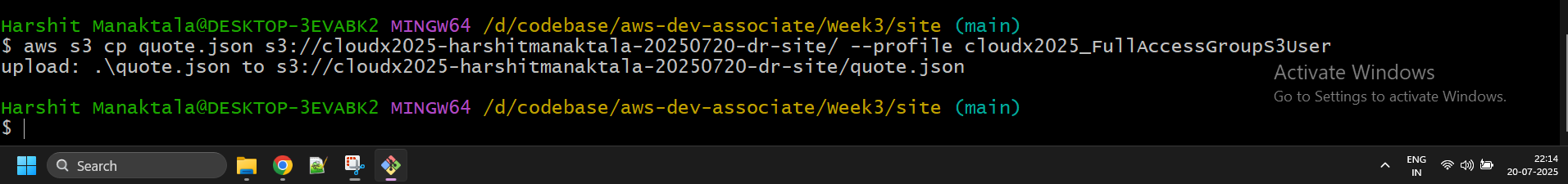
callback(rawFile.responseText);

}

}

rawFile.send(null);

}



A screenshot of a computer code

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

1. Verify that your static website from bucket1 can access resource from bucket2. To achieve this, use CORS rules configuration.

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

**Sub-task 6 - Think a little bit**

1. Describe all the use cases for S3 you've seen on past/current projects. (ex. configuration, image storage, blob files, reporting etc.)

**In my past/current projects, I have seen S3 being used:**

* **To move React based frontends out of ECS containers to S3 based static websites to save cost**
* **To use S3’s tiers to store artifacts that are rarely used (e.g. only in audits)**
* **To store intermediate results from step function executions while processing large datasets**

1. Describe any other S3 use cases you see reasonable for past/current projects.

* **To store chunks for the accounts needing financial hierarchy predictions and then stream reading it using S3 select** [**https://dev.to/idrisrampurawala/efficiently-streaming-a-large-aws-s3-file-via-s3-select-4on**](https://dev.to/idrisrampurawala/efficiently-streaming-a-large-aws-s3-file-via-s3-select-4on)

1. *Optionally, visualize some use cases via any preferred notation (UML, BPMN, AWS diagrams, etc).*