Creating a Static Website for a Café

Step 1: Extracting the files needed to make the static website.

In this task, you will extract the files that you need to create the static website.

1. Download the *.zip* file you need by opening this Amazon S3 link.
2. Extract the files.

Step 2: Creating an S3 bucket to host your static website

In this task, you will create an S3 bucket and configure it to host your static website.

3. Open the **Amazon S3 console**.
4. Create a bucket to host your static website.
    - Create the bucket in the **N. Virginia** (us-east-1) AWS Region-disable **Block *all* public access**.
5. Enable static website hosting on your bucket.
    - Use the *index.html* file for your index document.

Step 3: Uploading content to your S3 bucket

In this task, you will upload the static files to your S3 bucket.

6. Upload the *index.html* file and the *css* and *images* folders to your S3 bucket.

Step 4: Creating a bucket policy to grant public read access

7. Create a bucket policy that grants read-only permission to public anonymous users by using the Bucket Policy editor.
8. In a separate web browser tab, open the endpoint link for your static website.
9. The website for the café is now publicly accessible.

The static website for the café is now working!

New Business requirement: protect website data

Step 5: Enabling versioning on the S3 bucket

In this task, you will enable versioning on your S3 bucket and confirm that it works.

10. In the S3 console, enable versioning on the S3 bucket.
11. In the text editor, open the *index.html* file.
12. Modify the file according to the following instructions:
    - Locate the first line that has the embedded CSS code **bgcolor="aquamarine"** in the HTML, and change it to bgcolor="gainsboro".
    - Locate the line that has the embedded CSS code **bgcolor="orange"** in the HTML, and change it to bgcolor="cornsilk".
    - Locate the second line that has the embedded CSS code **bgcolor="aquamarine"** in the HTML, and change it to bgcolor="gainsboro".
    - Save the changes.
13. Upload the updated file to your S3 bucket.
14. Reload the web browser tab with your website and notice the changes.
15. To see the latest version of the *index.html* file, go to your bucket and choose **List versions**. You should see both versions of this file in the dropdown menu.

New business requirement: Optimize costs of S3 object storage

Step 6: Setting lifecycle policies

In this task, you will set a lifecycle policy to automatically move older versions of the objects in your source bucket to S3 Standard-Infrequent Access (S3 Standard-IA). The policy should also eventually expire the objects.

16. Configure two rules in the website bucket's lifecycle configuration.
17. In one rule, move previous versions of all source bucket objects to S3 Standard-IA after 30 days
18. In the other rule, delete previous versions of the objects after 365 days

You now have a lifecycle configuration that will move previous versions of your source bucket objects to S3 Standard-IA after 30 days. The policy will also permanently delete the objects that are in S3 Standard-IA after 365 days.

Step 7: Enabling cross-Region replication

In this task, you will enable cross-Region replication on your source S3 bucket.

19. In a different Region than your source bucket, create a second bucket and enable versioning on it. The second bucket is your *destination bucket*.
20. On your source S3 bucket, enable cross-Region replication. When you create the replication rule, make sure that you:
    - Replicate the entire source bucket.
    - Use the **CafeRole** for the AWS Identity and Access Management (IAM) role.

    - For CafeRole the access policy allows the role to perform the replication tasks on *all* S3 buckets.
21. Make a minor change to the *index.html* file and upload the new version to your source bucket.
22. Verify that the source bucket now has three versions of the *index.html* file.
23. Confirm that the new object was replicated to your destination bucket.
24. Go to your source bucket and delete the latest version.

This proves that any uploaded edits to the index.html file were replicated to the destination bucket.