



APPLICATION ON CLOUD

Assignment #6

ABSTRACT

This document is about an calculator app that stored on cloud and accessed by public users

UNIVERSITY OF SOUTH ASIA

Department of Computer Science



FALL 2022

Course Title: Cloud Computing

Course Code: CS-565

Assignment No.06

Course Instructor: Dr Gulzar Ahmad		
Section: A	Program: BSCS	Date: 31-10-2022
Submission Date:04-11-2022	Maximum Marks:100	Obtained Marks:
Program Objective:	Course Objective:	Course Learning Objective:
TO BE FILLED IN BY THE STUDENT		
Student Name: Malaika Mansoor & Tahira Amer	Registration No: B-23513 & B-22696	Sr. No:

List of Images:

Fig1: Represent the HTML code file	1
Fig2: Represent the Java script code file for indexing	1
Fig3: Represent the CSS file for styling or layout.....	2
Fig4:Represent the AWS console	2
Fig5:Represent the services of AWS	3
Fig6: Represent the created Buckets	3
Fig7:Represent the successfully created new bucket.....	5
Fig8: Represent the files from local computer	7
Fig9: Represent the properties of Bucket	8
Fig10: Represent the static website hosting.....	8
Fig11: Represent the actions of files to make them public	11
Fig12: Represent the Calculator accessed by using URL.....	12
Fig13: Represent the calculation for testing.....	13

Table of Content

Acknowledgment	iv
Abstract:	v
Steps how to store a calculator app on AWS instance:	1

Acknowledgment

We would like to thanks our professor Dr Gulzar Ahmad for assigning us this documentation of cloud computing for storing an application on cloud that provide services as a software and can be accessed by public users.

Abstract:

In this documentation we make a simple calculator as an application on our local computer that can provide services as a software and can be accessible for public users. We will create three files to provide calculator as an application to the public:

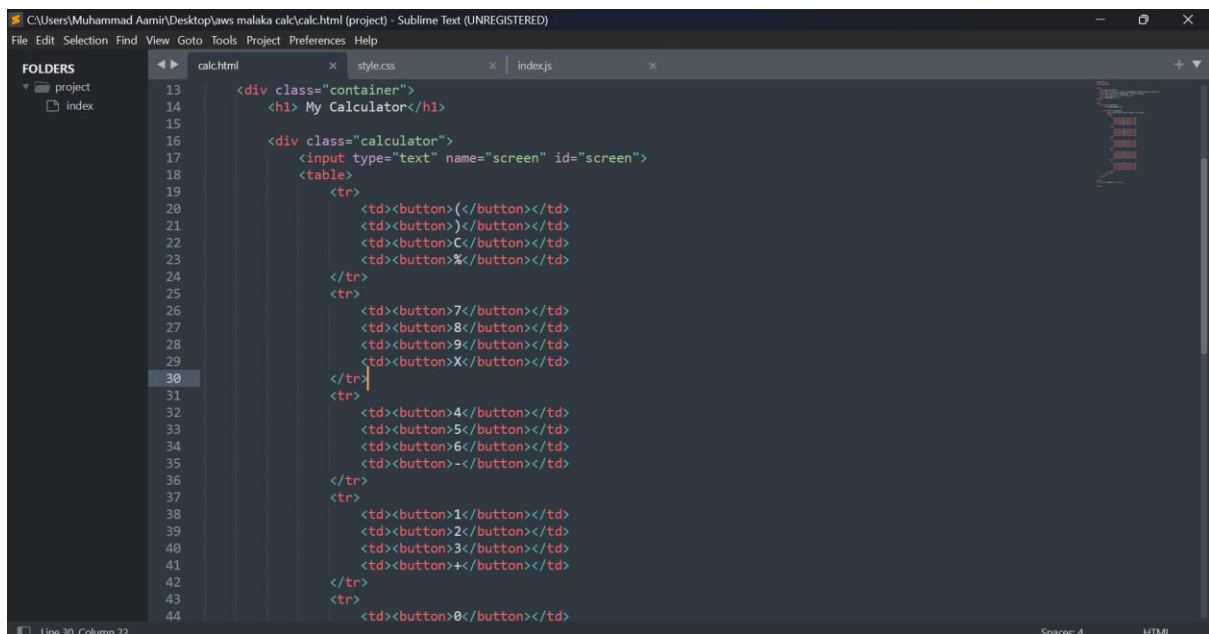
- HTML for the code of making calculator
- Java script for indexing that organize the relevant information of the calculator
- CSS for styling the layout that shows how calculator look or featured

Let's make the calculator app in our local computer and access it on AWS instance for public users

Cloud Computing

Steps how to store a calculator app on AWS instance:

Step1: HTML code for making calculator



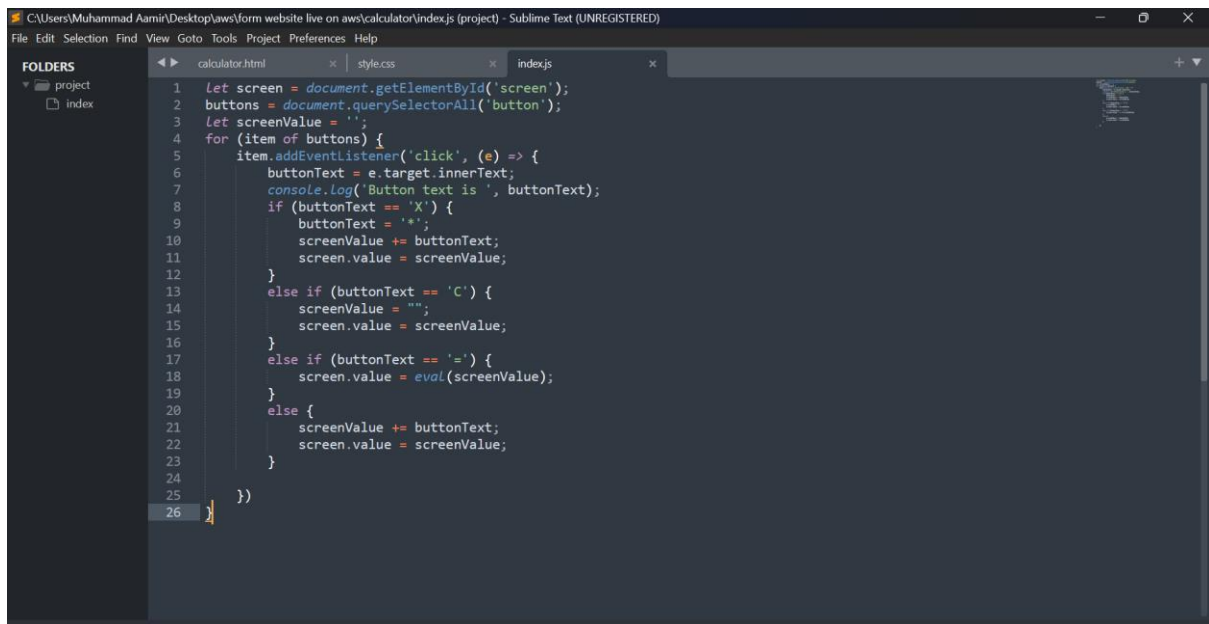
The screenshot shows a Sublime Text editor window titled "C:\Users\Muhammad Amin\Desktop\aws malaka calc\calc.html (project) - Sublime Text (UNREGISTERED)". The editor displays the HTML code for a calculator. The code includes a container div, a title "My Calculator", an input field with name="screen" and id="screen", and a table of buttons. The buttons are arranged in four rows: the first row contains buttons for division, multiplication, subtraction, and addition; the second row contains buttons for digits 7, 8, and 9, along with a button for the 'X' symbol; the third row contains buttons for digits 4, 5, and 6, along with a button for the '-' symbol; and the fourth row contains buttons for digits 1, 2, and 3, along with a button for the '+' symbol. The last row contains a button for the '0' symbol.

```
<div class="container">
  <h1> My Calculator</h1>

  <div class="calculator">
    <input type="text" name="screen" id="screen">
    <table>
      <tr>
        <td><button>(</button></td>
        <td><button>)</button></td>
        <td><button>C</button></td>
        <td><button>%</button></td>
      </tr>
      <tr>
        <td><button>7</button></td>
        <td><button>8</button></td>
        <td><button>9</button></td>
        <td><button>X</button></td>
      </tr>
      <tr>
        <td><button>4</button></td>
        <td><button>5</button></td>
        <td><button>6</button></td>
        <td><button>-</button></td>
      </tr>
      <tr>
        <td><button>1</button></td>
        <td><button>2</button></td>
        <td><button>3</button></td>
        <td><button>+</button></td>
      </tr>
      <tr>
        <td><button>0</button></td>
```

Fig1: Represent the HTML code file

Step2: Java script for indexing the calculator



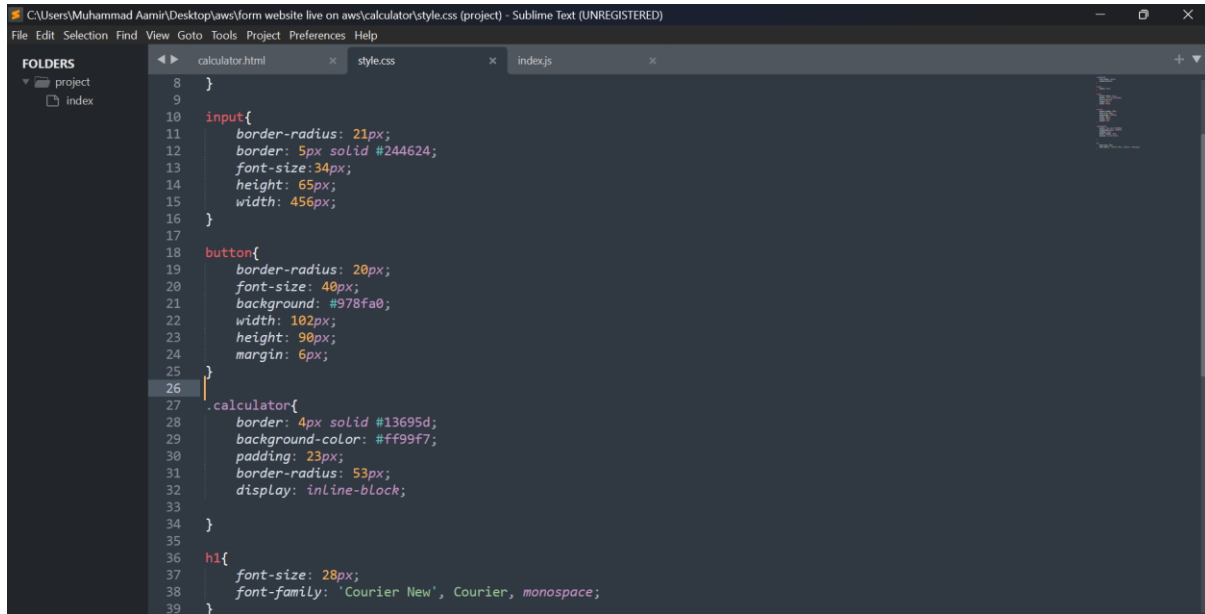
The screenshot shows a Sublime Text editor window titled "C:\Users\Muhammad Amin\Desktop\aws\form website live on aws\calculator\index.js (project) - Sublime Text (UNREGISTERED)". The editor displays the JavaScript code for indexing the calculator. The code uses the document.getElementById method to get the screen input field and the document.querySelectorAll method to get all buttons. It then uses a for loop to iterate over the buttons and add an event listener for each button. The event listener function checks the button text and updates the screen value accordingly. The code includes comments for each step of the process.

```
1 let screen = document.getElementById('screen');
2 buttons = document.querySelectorAll('button');
3 let screenValue = '';
4 for (item of buttons) {
5   item.addEventListener('click', (e) => {
6     buttonText = e.target.innerText;
7     console.log('Button text is ', buttonText);
8     if (buttonText == 'X') {
9       buttonText = '*';
10      screenValue += buttonText;
11      screen.value = screenValue;
12    }
13    else if (buttonText == 'C') {
14      screenValue = '';
15      screen.value = screenValue;
16    }
17    else if (buttonText == '=') {
18      screen.value = eval(screenValue);
19    }
20    else {
21      screenValue += buttonText;
22      screen.value = screenValue;
23    }
24  })
25 }
26
```

Fig2: Represent the Java script code file for indexing

Cloud Computing

Step3: CSS for styling or layout of the calculator



The screenshot shows a Sublime Text editor window with the title bar "C:\Users\Muhammad Amin\Desktop\aws\form website live on aws\calculator\style.css (project) - Sublime Text (UNREGISTERED)". The editor has three tabs: "calculator.html", "style.css", and "index.js". The "style.css" tab is active, displaying the following CSS code:

```
8 }
9
10 input{
11     border-radius: 21px;
12     border: 5px solid #244624;
13     font-size: 34px;
14     height: 65px;
15     width: 456px;
16 }
17
18 button{
19     border-radius: 20px;
20     font-size: 40px;
21     background: #978fa0;
22     width: 102px;
23     height: 90px;
24     margin: 6px;
25 }
26
27 .calculator{
28     border: 4px solid #13695d;
29     background-color: #ff99f7;
30     padding: 23px;
31     border-radius: 53px;
32     display: inline-block;
33 }
34
35
36 h1{
37     font-size: 28px;
38     font-family: 'Courier New', Courier, monospace;
39 }
```

Fig3: Represent the CSS file for styling or layout

Step4: Login to AWS console

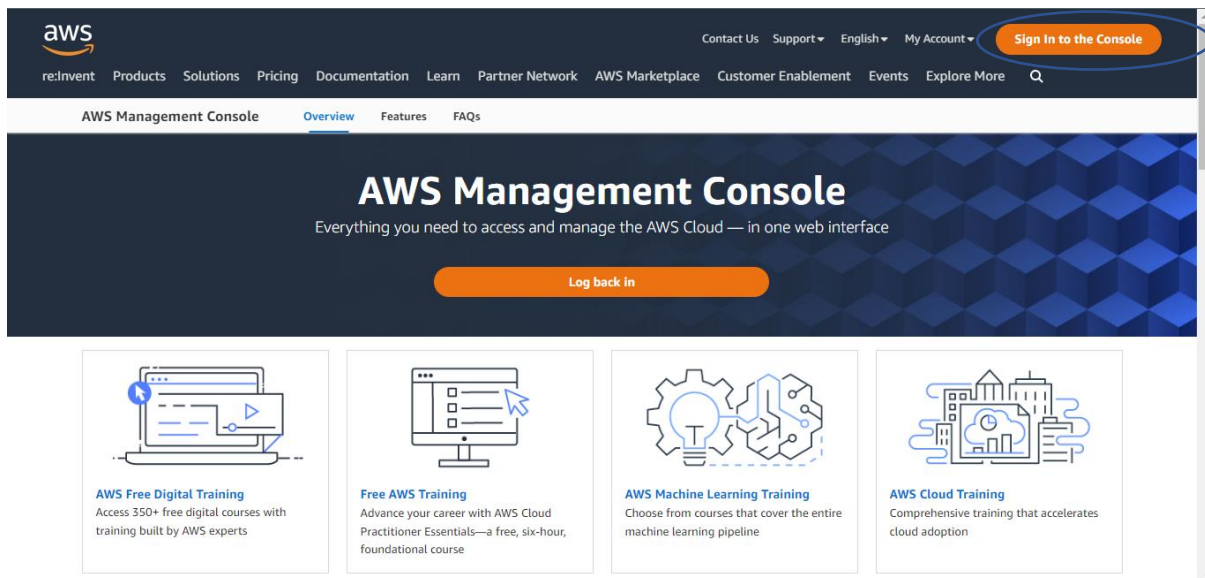


Fig4: Represent the AWS console

Step5: Search the S3 services for creating bucket

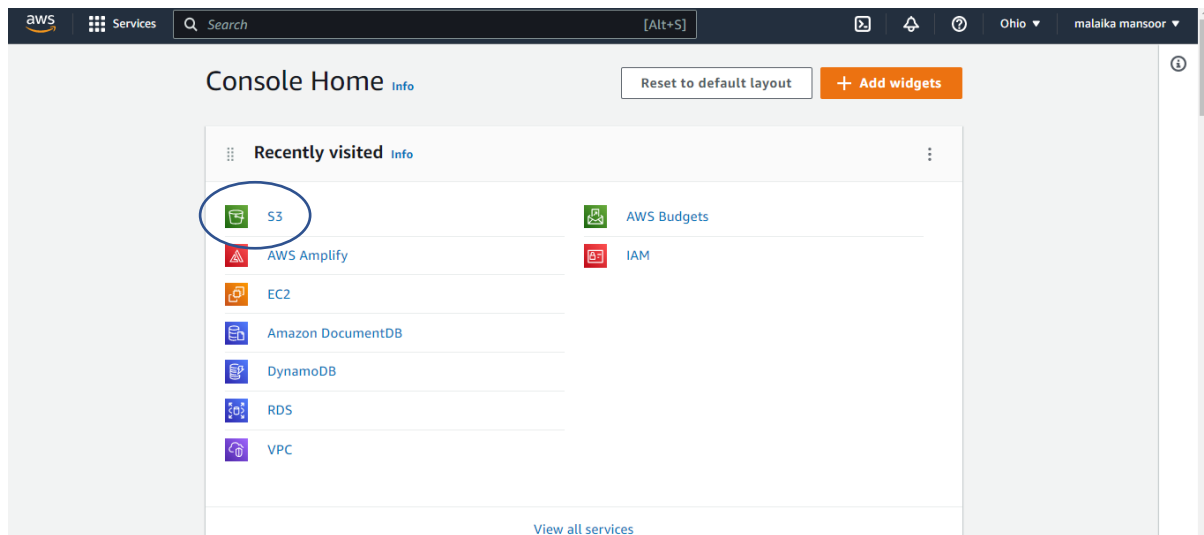


Fig5: Represent the services of AWS

Step6: Create Bucket for uploading files

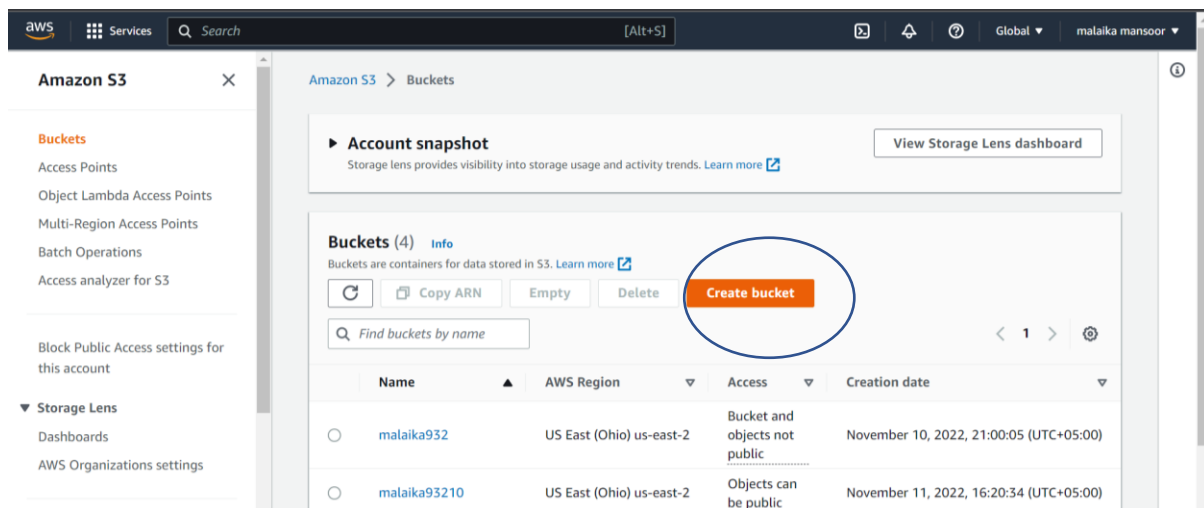
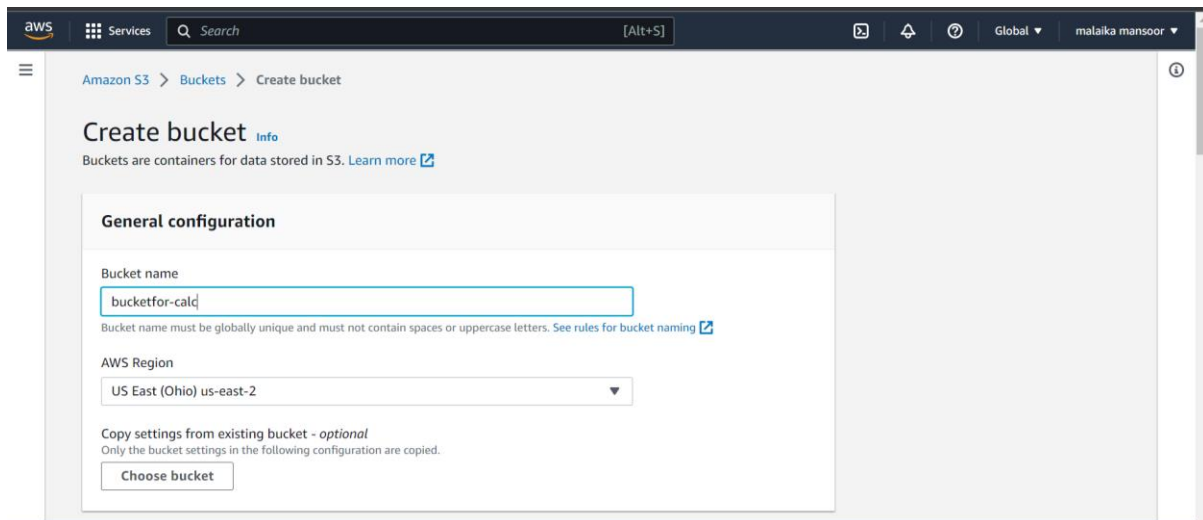


Fig6: Represent the created Buckets

Cloud Computing



aws Services Search [Alt+S] Global malaika mansoor

Amazon S3 > Buckets > Create bucket

Create bucket [Info](#)

Buckets are containers for data stored in S3. [Learn more](#)

General configuration

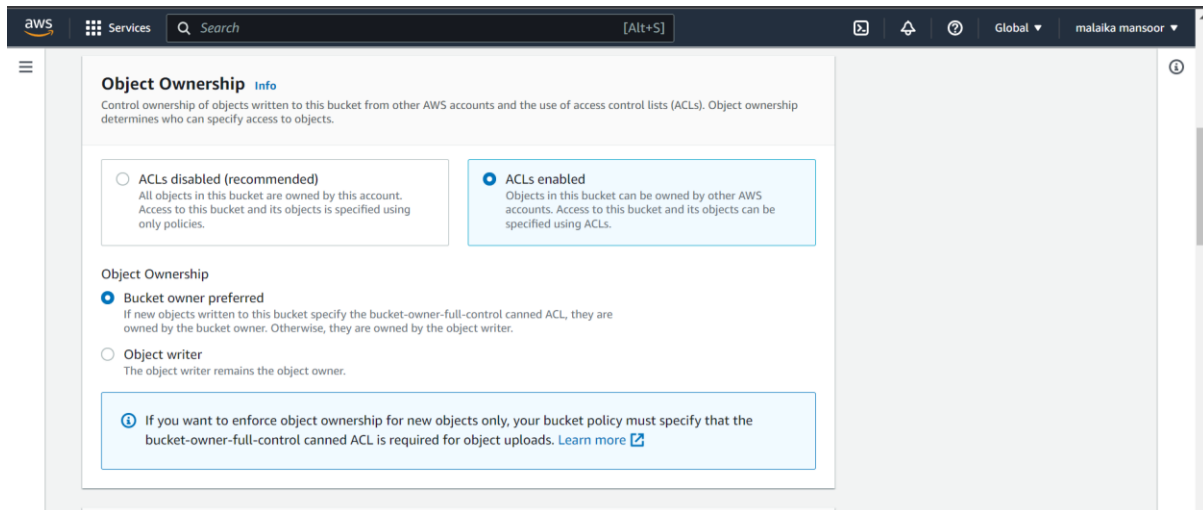
Bucket name
bucketfor-cald

Bucket name must be globally unique and must not contain spaces or uppercase letters. [See rules for bucket naming](#)

AWS Region
US East (Ohio) us-east-2

Copy settings from existing bucket - *optional*
Only the bucket settings in the following configuration are copied.

Choose bucket



aws Services Search [Alt+S] Global malaika mansoor

Object Ownership [Info](#)

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

☐ ACLs disabled (recommended)
All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

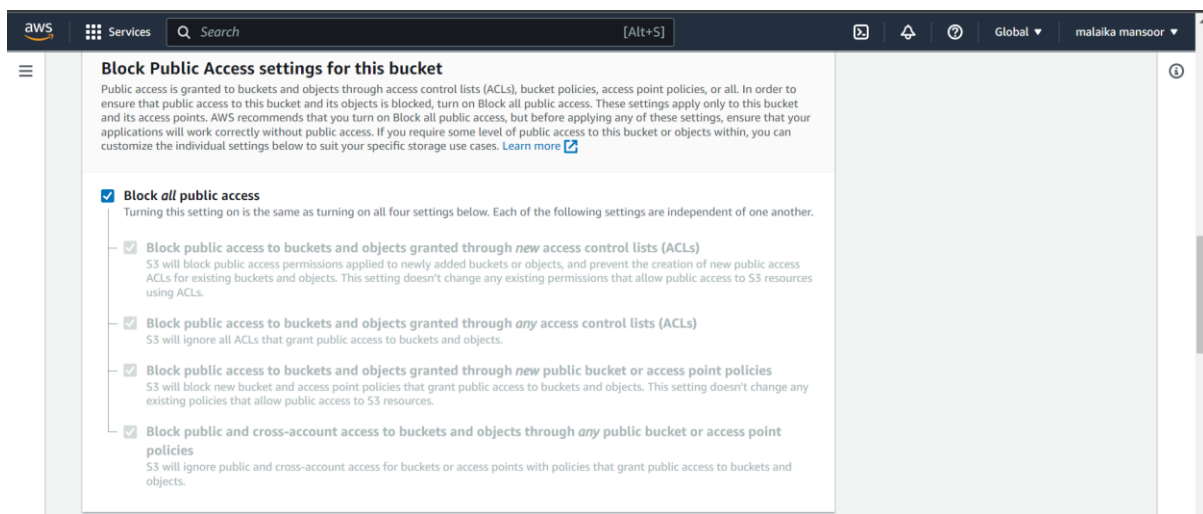
☒ ACLs enabled
Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

Object Ownership

☒ Bucket owner preferred
If new objects written to this bucket specify the bucket-owner-full-control canned ACL, they are owned by the bucket owner. Otherwise, they are owned by the object writer.

☐ Object writer
The object writer remains the object owner.

Info If you want to enforce object ownership for new objects only, your bucket policy must specify that the bucket-owner-full-control canned ACL is required for object uploads. [Learn more](#)



aws Services Search [Alt+S] Global malaika mansoor

Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

☒ **Block all public access**
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

- ☒ **Block public access to buckets and objects granted through *new* access control lists (ACLs)**
S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
- ☒ **Block public access to buckets and objects granted through *any* access control lists (ACLs)**
S3 will ignore all ACLs that grant public access to buckets and objects.
- ☒ **Block public access to buckets and objects granted through *new* public bucket or access point policies**
S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
- ☒ **Block public and cross-account access to buckets and objects through *any* public bucket or access point policies**
S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Cloud Computing

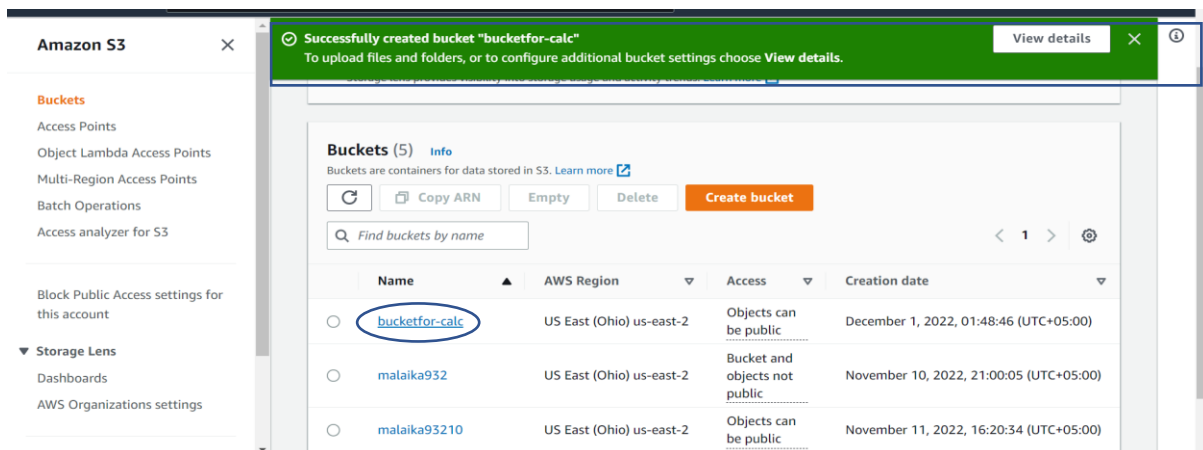
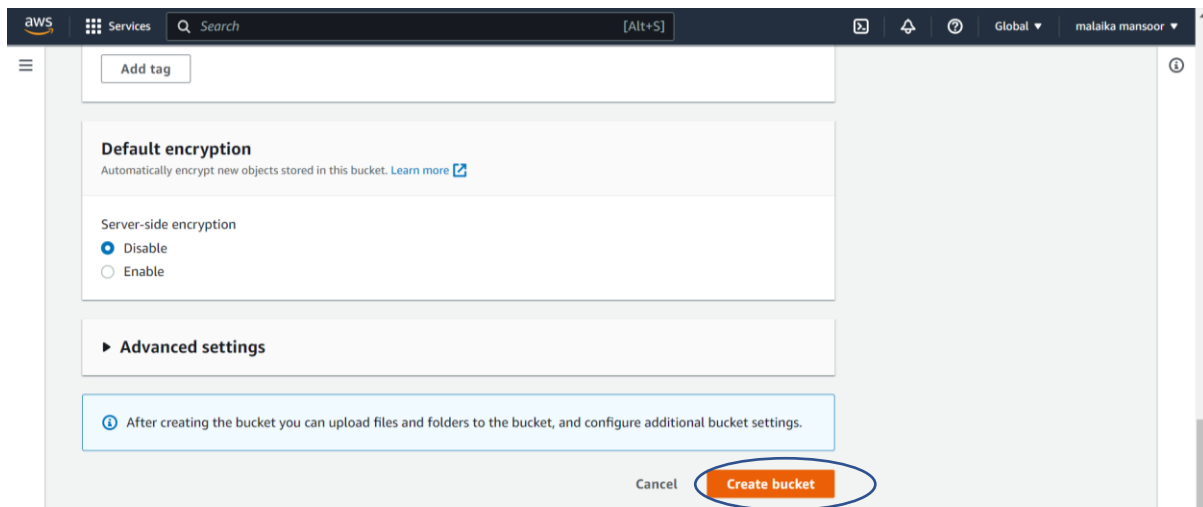
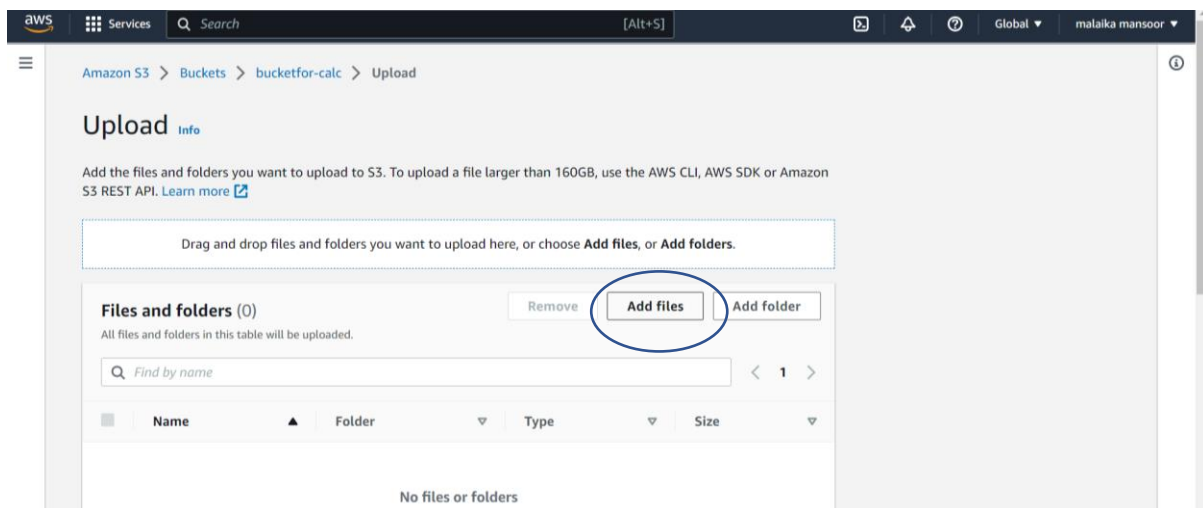
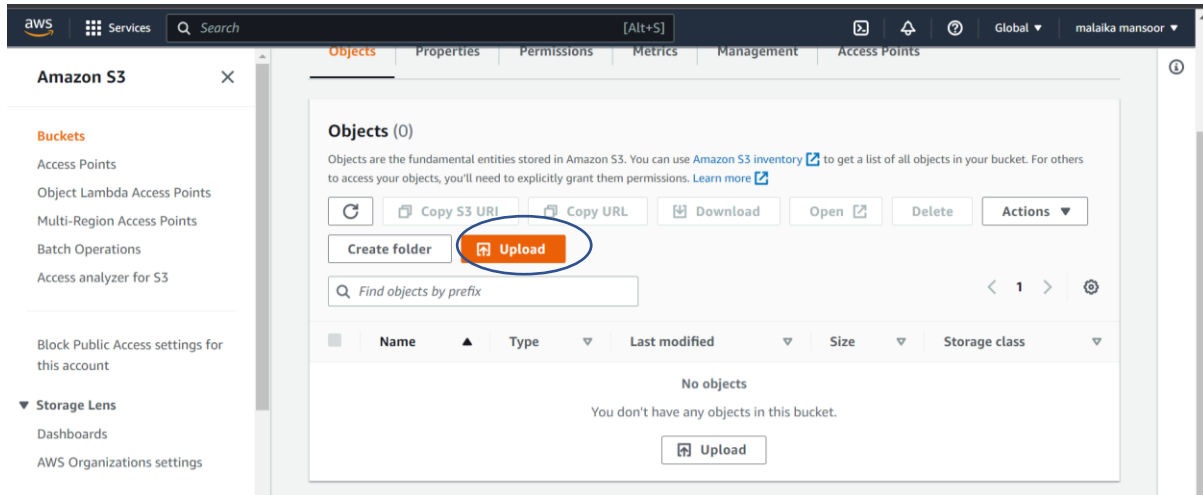


Fig7:Represent the successfully created new bucket

Cloud Computing

Step6: Upload files in the created Bucket



Cloud Computing

Step7: Select files from local computer to upload

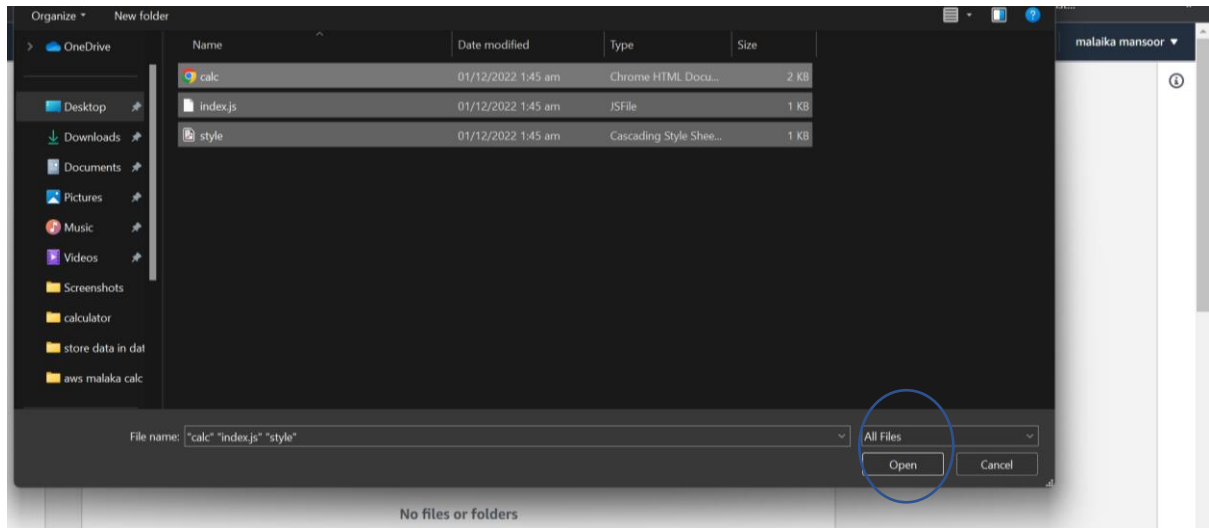
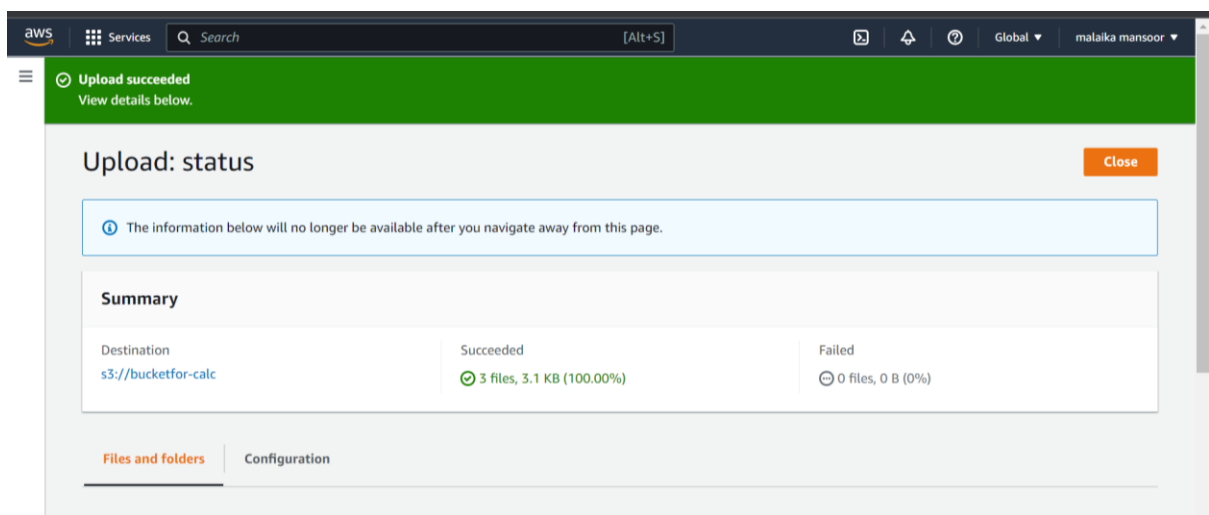


Fig8: Represent the files from local computer



Step8: From the properties Scroll down

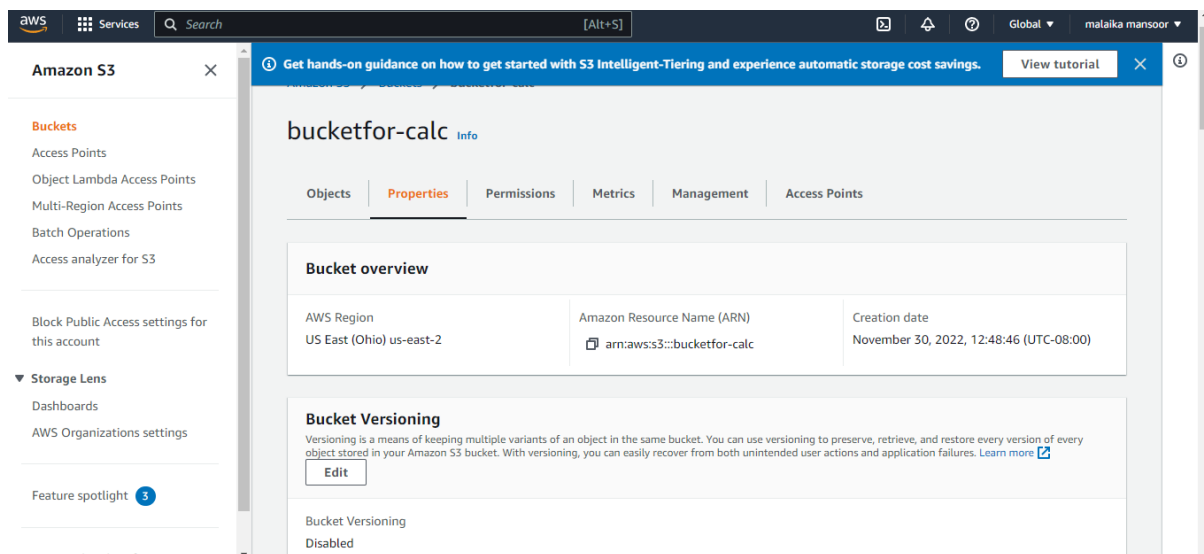


Fig9: Represent the properties of Bucket

Step9: Edit the Static website hosting

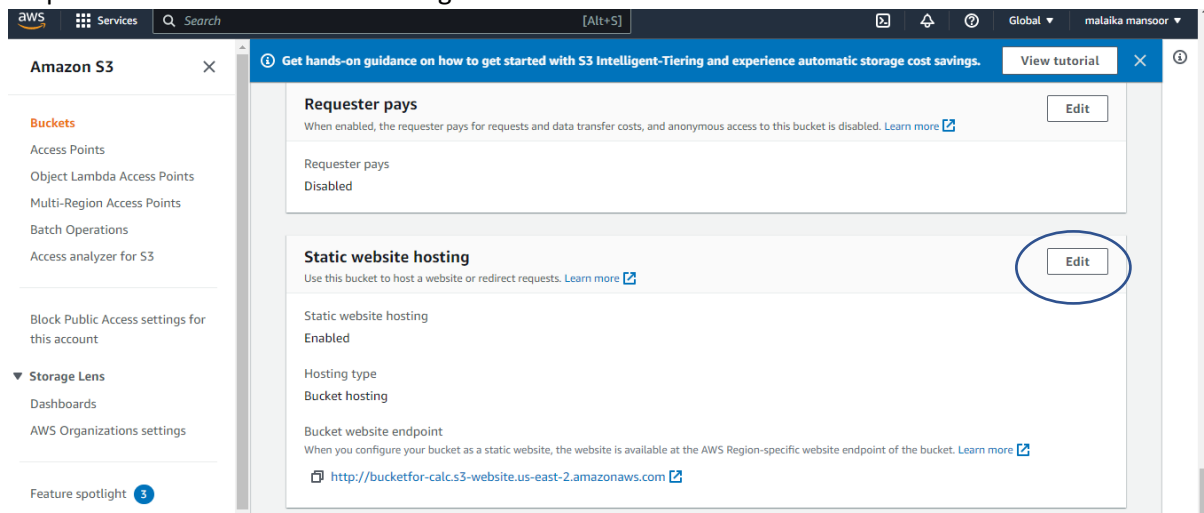


Fig10: Represent the static website hosting

Cloud Computing

Static website hosting
Use this bucket to host a website or redirect requests. [Learn more](#)

Static website hosting

☐ Disable

☒ Enable

Hosting type

☒ Host a static website
Use the bucket endpoint as the web address. [Learn more](#)

☐ Redirect requests for an object
Redirect requests to another bucket or domain. [Learn more](#)

Index document
Specify the home or default name of the website.

[Using Amazon S3 Block Public Access](#)

Index document
Specify the home or default page of the website.

Error document - optional
This is returned when an error occurs.

Redirection rules - optional
Redirection rules, written in JSON, automatically redirect webpage requests for specific content. [Learn more](#)

1

Amazon S3 > Buckets > bucketfor-calc > Edit static website hosting

Edit static website hosting [Info](#)

Static website hosting
Use this bucket to host a website or redirect requests. [Learn more](#)

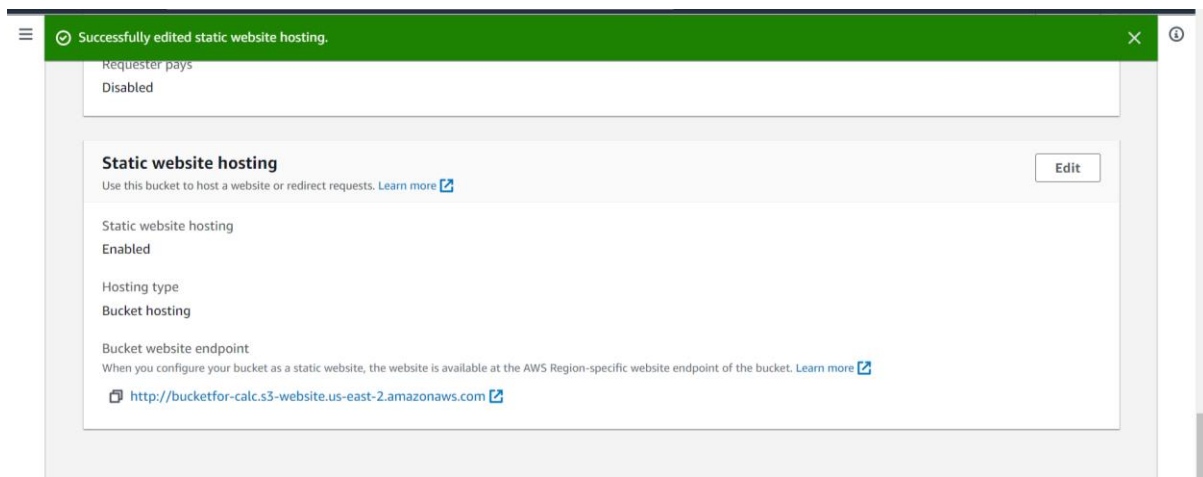
Static website hosting

☒ Disable

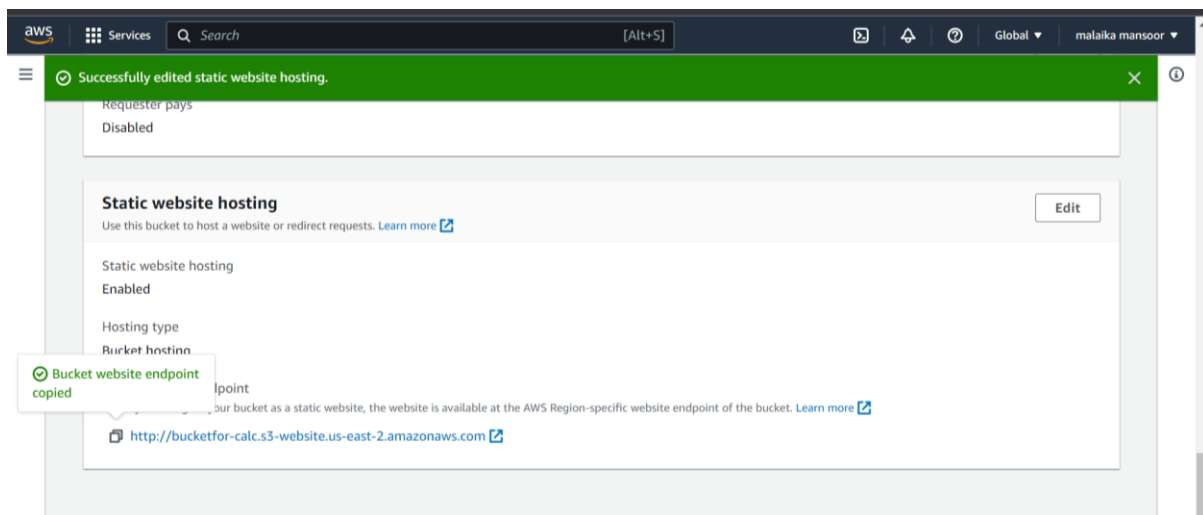
☐ Enable

Cancel [Save changes](#)

Cloud Computing



Step10: Copy the URL to clipboard for sharing



Cloud Computing

Step11: Select all the uploaded files and go to action for making public

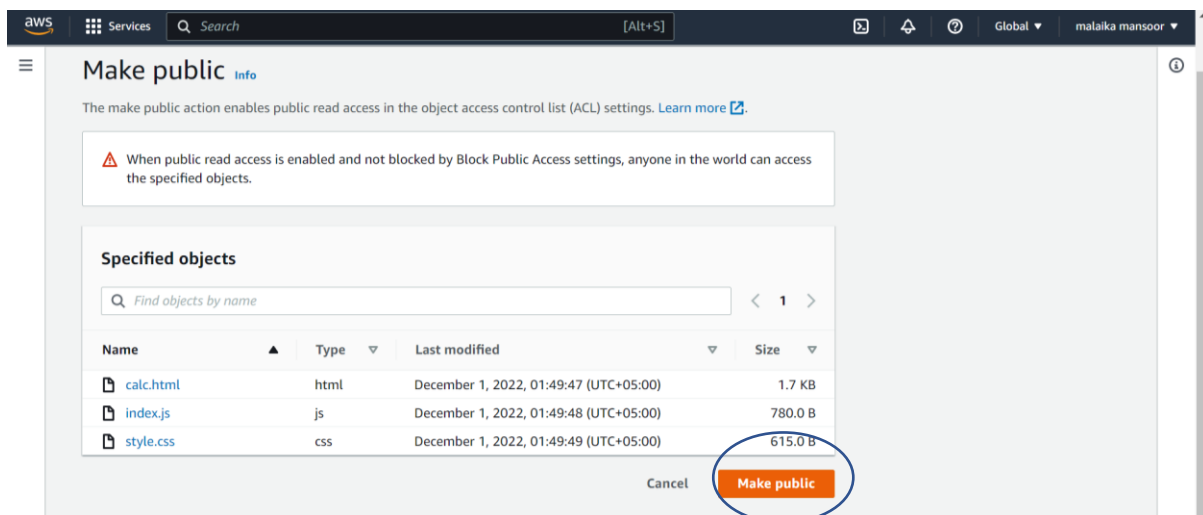
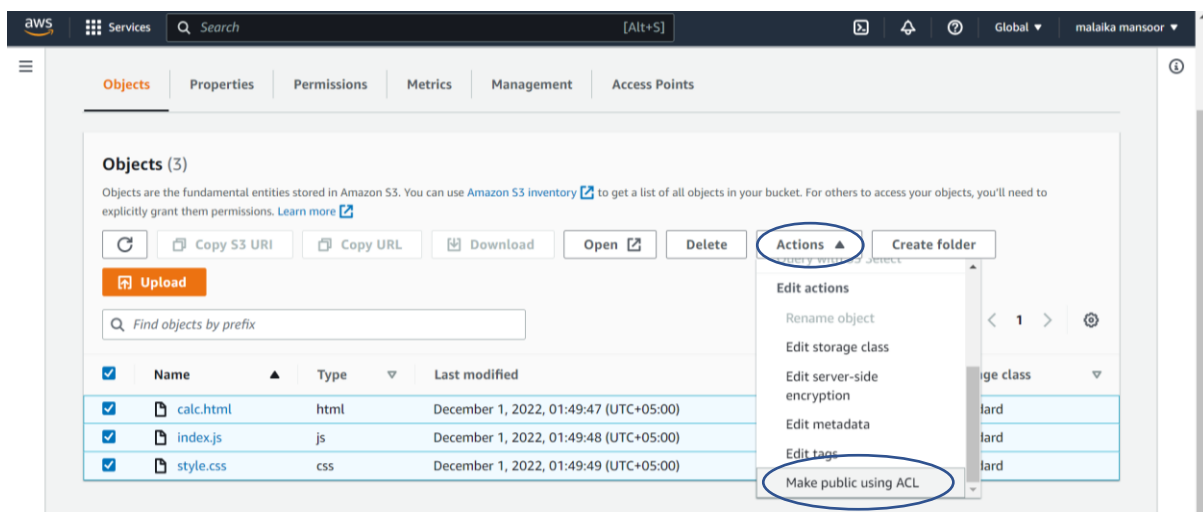


Fig11: Represent the actions of files to make them public

Step12: Go to new window and paste the URL of calculator bucket

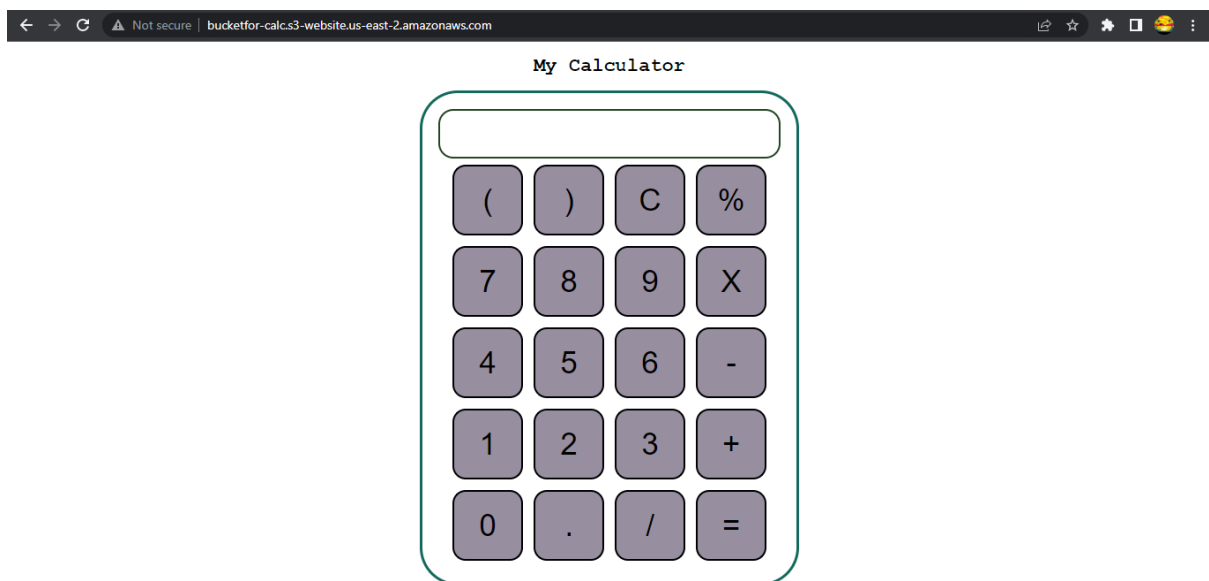
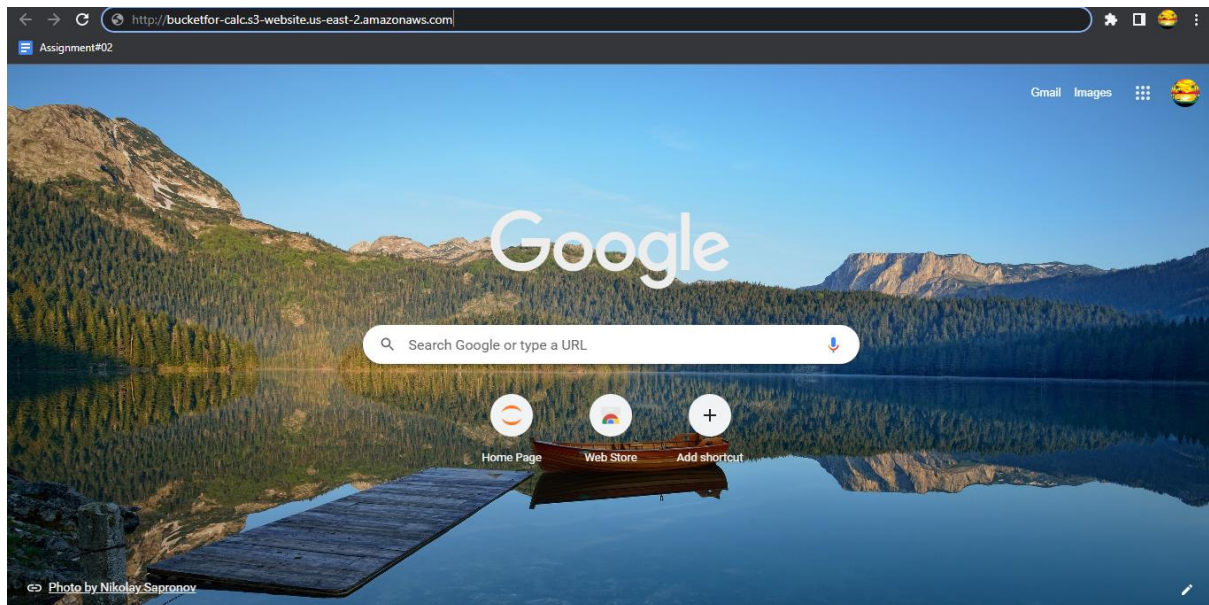


Fig12: Represent the Calculator accessed by using URL

Step13: Perform calculation for testing on calculator

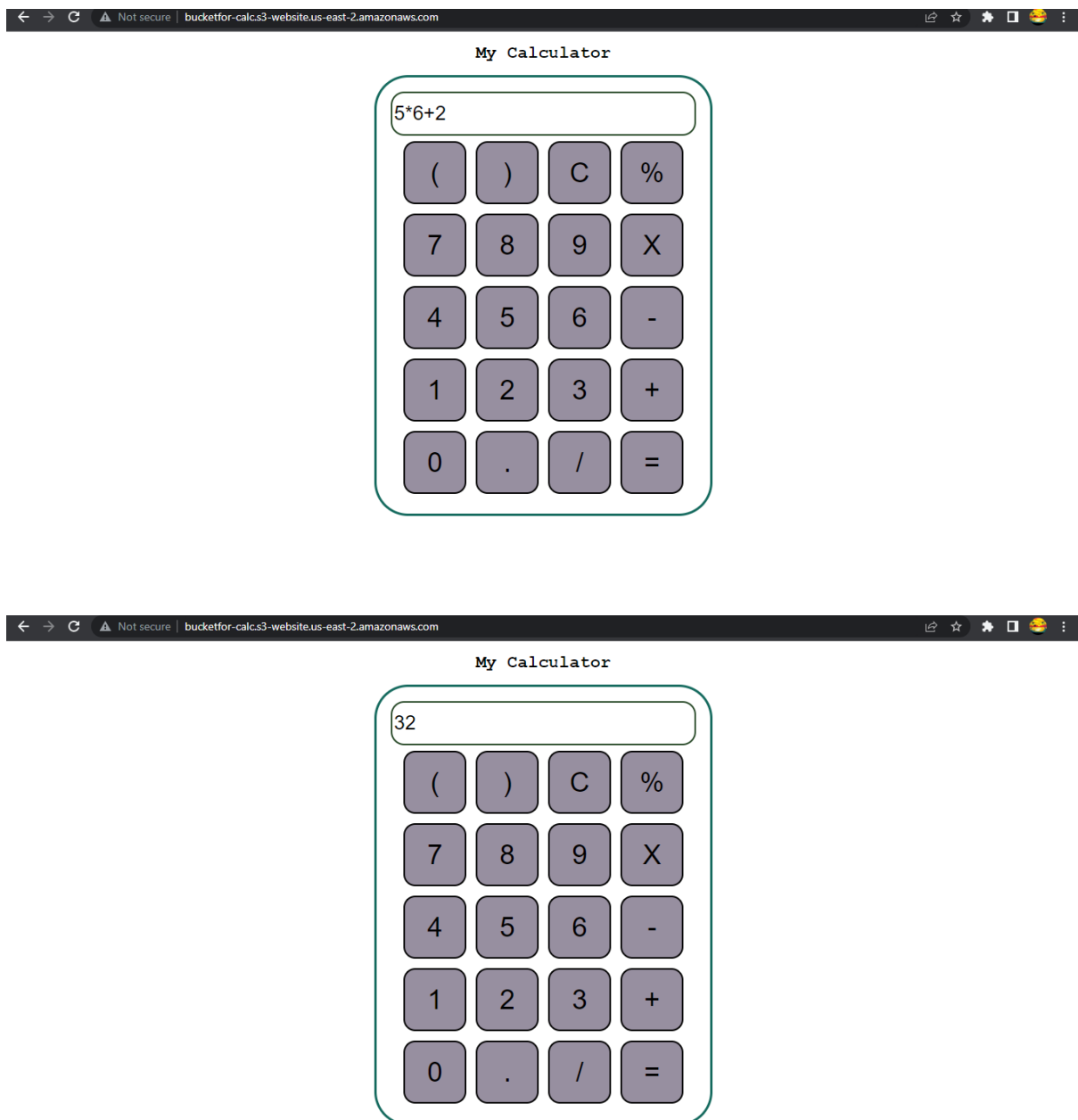


Fig13: Represent the calculation for testing