| Academic Year | Module | Task |
| --- | --- | --- |
| 2025 | 5CS022/HJ1: Distributed and Cloud Systems Programming | 2 |

Name: Swoyam Pokharel
ID: 2431342
Group: 26
Tutor: Ms. Jenny Rajak

# Table Of Contents:

UNIVERSITY OF WOLVERHAMPTON

HERALD COLLEGE KATHMANDU

# Question:

In a similar approach to  Workshop on AWS, create a static website on AWS S3 Bucket, consisting of 3 HTML web pages, and one image. The 3 web pages should be named “index.html”, “page1.html”, and “page2.html”. The image file should be named “image1.png” and should contain either a picture of a kitten or flowers. The “index.html” page should display the image in a <img> tag, similar to the workshop exercise. In addition, it should also contain a link to “page1.html” and a link to “page2.html”. The two files “page1.html” and “page2.html” should both contain a link back to “index.html”, and to each other.  The image file “image1.png”, as well as “page1.html” and “page2.html”, should then be deployed to the AWS Amplify service, and all the links updated. The only non-Amplify link should be only to “index.html”. Your submission should be a zip of the 3 HTML files plus the image and a well documented MS Word file containing your name, student ID and explanation and screenshots of the 3 HTML pages, as well as the image on AWS Amplify, in your web browser showing the link URLs clearly.
This task will contribute 20% of the marks to the Portfolio.

# Solution:

## 1. Understanding the requirements

So, the question wants us to create 3 html pages, explicitly named:

`index.html`
`page1.html`
`page2.html`

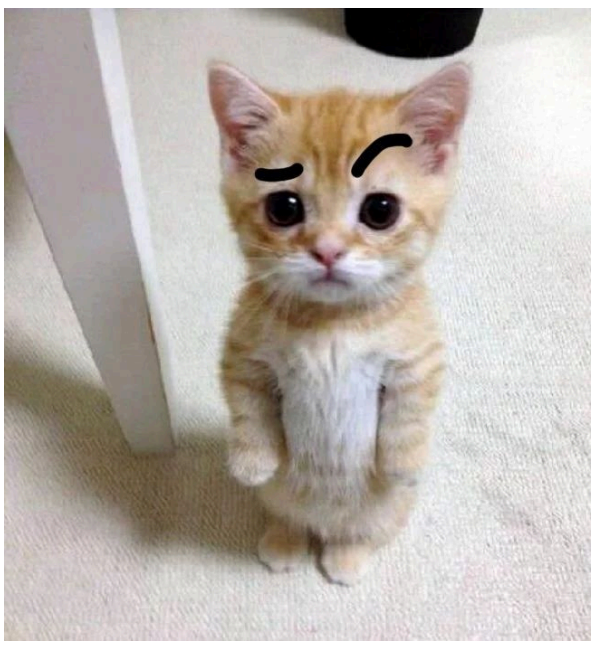
In the index.html page, the question wants us to display images which can either be of kittens, or flowers.
We then want to create a static website on AWS S3 Bucket hosting those 3 html pages and the images.
Then, we need to deploy `page1.html`, `page2.html` along with the images on AWS Amplify service, and we need to update the links so that the following navigation path is achieved:

`index.html` (hosted on S3) → Links To `page1.html` and `page2.html` ( both hosted on amplify )
`page1.html` (hosted on Amplify) → Links To `page2.html` (hosted on amplify) and `index.html` (hosted on S3)
`page2.html` (hosted on Amplify)→ Links to `page1.html` (hosted on amplify) and `index.html` (hosted on s3)

## 2. Preparation & Initial Setup:

### Images:

Since we require images, of either kittens or flowers, I've decided to go with the following 3 images:



Project Structure:

```
[wizard@archlinux Swoyam_Pokharel_2431342]$ ls
[wizard@archlinux Swoyam_Pokharel_2431342]$ touch {index,page1,page2}.html
[wizard@archlinux Swoyam_Pokharel_2431342]$ tree
.
├── index.html
├── page1.html
└── page2.html

1 directory, 3 files
[wizard@archlinux Swoyam_Pokharel_2431342]$
```

```
[wizard@archlinux Swoyam_Pokharel_2431342]$ tree
.
├── index.html
├── page1.html
├── page2.html
└── static
    ├── images
    │   ├── image1.jpg
    │   ├── image2.jpg
    │   └── image3.jpg
    ├── input.css
    └── output.css

3 directories, 8 files
[wizard@archlinux Swoyam_Pokharel_2431342]$
```

```
1 2                                                    ♪: 0%|BAT 79.34%|Tue 2025-05-13 16:40

@import url('https://fonts.googleapis.com/css2?family=Geist&display=swap');
@import "tailwindcss";
@plugin "@tailwindcss/typography";
@plugin "daisyui";


*{
    font-family:"Geist",sans-serif;
}

@plugin "daisyui/theme" {
  name: "black";
  default: true;
  prefersdark: true;
  color-scheme: "dark";
  --color-base-100: white;
  --color-base-200: white;
  --color-base-300: oklch(14% 0.004 49.25);
  --color-base-content: oklch(12% 0.042 264.695);
  --color-primary: oklch(13% 0.028 261.692);
  --color-primary-content: white;
  --color-secondary: white;
  --color-secondary-content: oklch(12% 0.042 264.695);
  --color-accent: oklch(84% 0.238 128.85);
  --color-accent-content: oklch(12% 0.042 264.695);
  --color-neutral: oklch(20% 0 0);
  --color-neutral-content: oklch(100% 0 0);
  --color-info: oklch(60% 0.25 292.717);
  --color-info-content: oklch(12% 0.042 264.695);
  --color-success: oklch(50% 0.118 165.612);
  --color-success-content: oklch(90.395% 0.035 142.495);
  --color-warning: oklch(96.798% 0.211 109.769);
  --color-warning-content: oklch(19.359% 0.042 109.769);
  --color-error: oklch(62.795% 0.257 29.233);
:
```

Screenshots showing the project structure

I've placed the images in the `static/images/` directory. For styling, I'm using `Tailwind CSS` along with `DaisyUI` for convenience. The `input.css` file includes base styles and a custom theme generated using the DaisyUI-Theme-Generator. `Tailwind` compiles this into `output.css`, which is the stylesheet used in the HTML pages. This is all that we will be doing for the initial setup. For the live development server I'm using `bun`.

```
index.html  page1.html  page2.html  static/
[wizard@archlinux Swoyam_Pokharel_2431342]$ tailwindcss -i static/input.css -o static/output.
css --watch
/*! ⌀  daisyUI 5.0.28 */
sh: line 1: watchman: command not found
≈ tailwindcss v4.1.4

Done in 141ms
Done in 313µs
Done in 207µs
Done in 66ms
Done in 5ms
Done in 245µs
Done in 4ms
Done in 16ms
Done in 1ms
Done in 384µs
Done in 352µs
Done in 368µs
Done in 514µs
Done in 372µs
Done in 433µs
Done in 473µs
Done in 5ms
Done in 277µs
Done in 313µs
Done in 401µs
Done in 320µs
Done in 304µs
```

```
[wizard@archlinux Swoyam_Pokharel_2431342]$ ls
index.html  page1.html  page2.html  static/
[wizard@archlinux Swoyam_Pokharel_2431342]$ bunx serve

  Serving!

  - Local:    http://localhost:3000
  - Network:  http://192.168.1.8:3000

  Copied local address to clipboard!
```

Screenshot showing tailwind compiling & bun server running on :3000

## 3. Finalized Code:

Index.html:

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1.0" />
        <link rel="stylesheet" href="/static/output.css" />
        <title>Index 2431342</title>
    </head>
    <body class="m-4 h-screen overflow-hidden">

        <!-- Gradient Effect Top Left -->
        <div class="absolute inset-x-0 -top-40 -z-10 transform-gpu overflow-hidden blur-3xl sm:-top-80" aria-hidden="true">
            <div class="relative left-[calc(50%-11rem)] aspect-1155/678 w-[36.125rem] -translate-x-1/2 rotate-[30deg] bg-linear-to-tr from-[#ff80b5] to-[#9089fc] opacity-30 sm:left-
            [calc(50%-30rem)] sm:w-[72.1875rem]" style="clip-path: polygon(74.1% 44.1%, 100% 61.6%, 97.5% 26.9%, 85.5% 0.1%, 80.7% 2%, 72.5% 32.5%, 60.2% 62.4%, 52.4% 68.1%, 47.5%
            58.3%, 45.2% 34.5%, 27.5% 76.7%, 0.1% 64.9%, 17.9% 100%, 27.6% 76.8%, 76.1% 97.7%, 74.1% 44.1%)"></div>
        </div>
```

Screenshot of the headers of `index.html`

This is mostly standard boilerplate HTML. The only notable addition is a gradient blob positioned at the top left. It's just a polygon placed behind a blurred background, creating the illusion of a gradient flowing from that corner.

```
<!-- Navigation Bar -->
<div class="px-30 navbar sticky top-0 z-50 bg-transparent backdrop-blur-md">
    <div class="flex justify-between items-center w-full">
        <!-- Left -->
        <div class="flex-1">
            <a class="btn btn-ghost text-xl nav-link tooltip tooltip-bottom" data-tip="Navigate Back!" onclick="window.history.back()">&larr;</a>
        </div>

        <!-- Right -->
        <div class="flex-1 justify-end flex">
            <ul class="menu menu-horizontal px-1 flex gap-3">
                <li>
                    <a class="text-base font-medium uppercase nav-link tooltip tooltip-bottom" data-tip="Meow Rotate 45" href="/page1.html">Page 1</a>
                </li>
                <li>
                    <a class="text-base font-medium uppercase nav-link tooltip tooltip-bottom" data-tip="Meow Rotate 135" href="/page2.html">Page 2</a>
                </li>
                <li>
                    <details>
                        <summary class="text-base font-medium uppercase nav-link tooltip tooltip-left" data-tip="Click For Links To Images on S3 Bucket">Images</summary>
                        <ul class="bg-base-100 rounded-t-none p-2 uppercase font-bold">
                            <li><a class="text-md" href="/static/images/image1.jpg">Image-1</a></li>
                            <li><a class="text-md" href="/static/images/image2.jpg">Image-2</a></li>
                            <li><a class="text-md" href="/static/images/image3.jpg">Image-3</a></li>
                        </ul>
                    </details>
                </li>
            </ul>
        </div>
    </div>
</div>


index.html

[0] 0:nvim* 1:bunx-
```

Screenshot showing the navigation of index.html

The above code is a navigation bar for `index.html`. The navigation bar includes a back button on the left, that utilizes the browser's native history to go back, and links to `page1.html` and `page2.html` on the right. I've also included a drop down menu for the images I've used.

```
<!-- Page Content -->
<div class="h-2/3 w-screen flex-col flex justify-center gap-20 items-center">
    <h1 class="text-8xl font-bold uppercase tracking-wide hero-title"> Index Page </h1>
    <div class="flex gap-4">
        <img src="/static/images/image1.jpg" class="w-100 border border-gray-400 rounded-full"/>
        <img src="/static/images/image3.jpg" class="border border-gray-400 rounded-full"/>
        <img src="/static/images/image2.jpg" class="w-100 border border-gray-400 rounded-full"/>
    </div>
</div>

<!-- Gradient Effect Bottom Right -->
<div class="absolute inset-x-0 top-[calc(100%-13rem)] -z-10 transform-gpu overflow-hidden blur-3xl sm:top-[calc(100%-30rem)]" aria-hidden="true">
    <div class="relative left-[calc(50%+3rem)] aspect-1155/678 w-[36.125rem] -translate-x-1/2 bg-linear-to-tr from-[#ff80b5] to-[#9089fc] opacity-30 sm:left-[calc(50%+
    36rem)] sm:w-[72.1875rem]" style="clip-path: polygon(74.1% 44.1%, 100% 61.6%, 97.5% 26.9%, 85.5% 0.1%, 80.7% 2%, 72.5% 32.5%, 60.2% 62.4%, 52.4% 68.1%, 47.5% 58.3%, 45.
    2% 34.5%, 27.5% 76.7%, 0.1% 64.9%, 17.9% 100%, 27.6% 76.8%, 76.1% 97.7%, 74.1% 44.1%)"></div>
</div>

</body>
```

Screenshot Of The Main Page Content Of Index.html

The above code shows the main content of the `index.html` page, it simply is a flex container with the 3 images of kittens that we've selected above. Furthermore, just like in the header, we have a similar gradient positioned at the bottom right of the page.

```
<script src="https://cdn.jsdelivr.net/npm/gsap@3.13.0/dist/gsap.min.js"></script>
<script src="https://cdn.jsdelivr.net/npm/gsap@3.13.0/dist/SplitText.min.js"></script>

<script>
  document.addEventListener("DOMContentLoaded", () => {
      const title = document.querySelector(".hero-title");
      const split = new SplitText(title, { type: "chars" });

      // hero title split text animation
      gsap.from(split.chars, {
          opacity: 0,
          y: 50,
          rotateX: 90,
          stagger: 0.05,
          ease: "back.out(1.7)",
          duration: 1,
      });

      // navigation links fly in
      gsap.from(gsap.utils.toArray(".nav-link"), {
          y: -20,
          opacity: 0,
          stagger: 0.1,
          delay: 0.3,
          duration: 0.6,
          ease: "power2.out"
      });

      // scale up them kittens
      gsap.from(gsap.utils.toArray("img"), {
          opacity: 0,
          stagger: 0.2,
          delay: 0.3,
          scale:0.2,
          duration: 0.6,
      });
  });
  </script>
</html>
index.html
"index.html" 104L, 5351B
[0] 0:nvim* 1:bunx-
```

Screenshots Showing gsap animations

Finally, at the bottom of the page, I've added some basic GSAP animations to bring a bit of life to the page and keep it from feeling too static.

That concludes the content for `index.html` we will be following a similar pattern in the rest of the pages. This is how index.html looks:


Screenshot of index.html on my local machine


Screenshot of index.html with the dropdown expanded showing the 3 images I've used

## Page1.html:

Since the question doesn't specify any particular content for this page, I've kept the layout almost identical to `index.html`. The main differences are the page title, updated navigation (which now links to page2.html and index.html), and the images—which I rotated 45 degrees to give the page a different visual feel compared to the index. The following is the only difference, removed lines are marked ( - ) and added lines are marked ( + )

```
STDIN
1  --- index.html  2025-05-12 18:17:00.936390751 +0545
2  +++ page1.html  2025-05-12 18:10:56.561958119 +0545
3  @@ -4,7 +4,7 @@
4          <meta charset="UTF-8" />
5          <meta name="viewport" content="width=device-width, initial-scale=1.0" />
6          <link rel="stylesheet" href="/static/output.css" />
7  -       <title>Index 2431342</title>
8  +       <title>Page1 2431342</title>
9      </head>
10     <body class="m-4 h-screen overflow-hidden">
11
12 @@ -25,7 +25,7 @@
13                 <div class="flex-1 justify-end flex">
14                     <ul class="menu menu-horizontal px-1 flex gap-3">
15                         <li>
16 -                           <a class="text-base font-medium uppercase nav-link tooltip tooltip-bottom" data-tip="Meow Rotate 45" href="/page1.html">Page 1</a>
17 +                           <a class="text-base font-medium uppercase nav-link tooltip tooltip-bottom" data-tip="Cat No Rotate" href="/index.html">Index</a>
18                         </li>
19                         <li>
20                             <a class="text-base font-medium uppercase nav-link tooltip tooltip-bottom" data-tip="Meow Rotate 135" href="/page2.html">Page 2</a>
21 @@ -46,13 +46,14 @@
22         </div>
23
24
25 +
26         <!-- Page Content -->
27         <div class="h-2/3 w-screen flex-col flex justify-center gap-20 items-center">
28 -           <h1 class="text-8xl font-bold uppercase tracking-wide hero-title"> Index Page </h1>
29 +           <h1 class="text-8xl font-bold uppercase tracking-wide hero-title"> Page 1</h1>
30             <div class="flex gap-4">
31 -               <img src="/static/images/image1.jpg" class="w-100 border border-gray-400 rounded-full"/>
32 -               <img src="/static/images/image3.jpg" class="border border-gray-400 rounded-full"/>
33 -               <img src="/static/images/image2.jpg" class="w-100 border border-gray-400 rounded-full"/>
34 +               <img src="/static/images/image1.jpg" class="w-100 border border-gray-400 rounded-full rotate-45"/>
35 +               <img src="/static/images/image3.jpg" class="border border-gray-400 rounded-full rotate-45"/>
36 +               <img src="/static/images/image2.jpg" class="w-100 border border-gray-400 rounded-full rotate-45"/>
37             </div>
38         </div>
39
:
[0] 0:nvim  1:bunx- 2:bash*
```

Screenshot showing differences between page1.html and index.html using diff piped to bat for clarity.

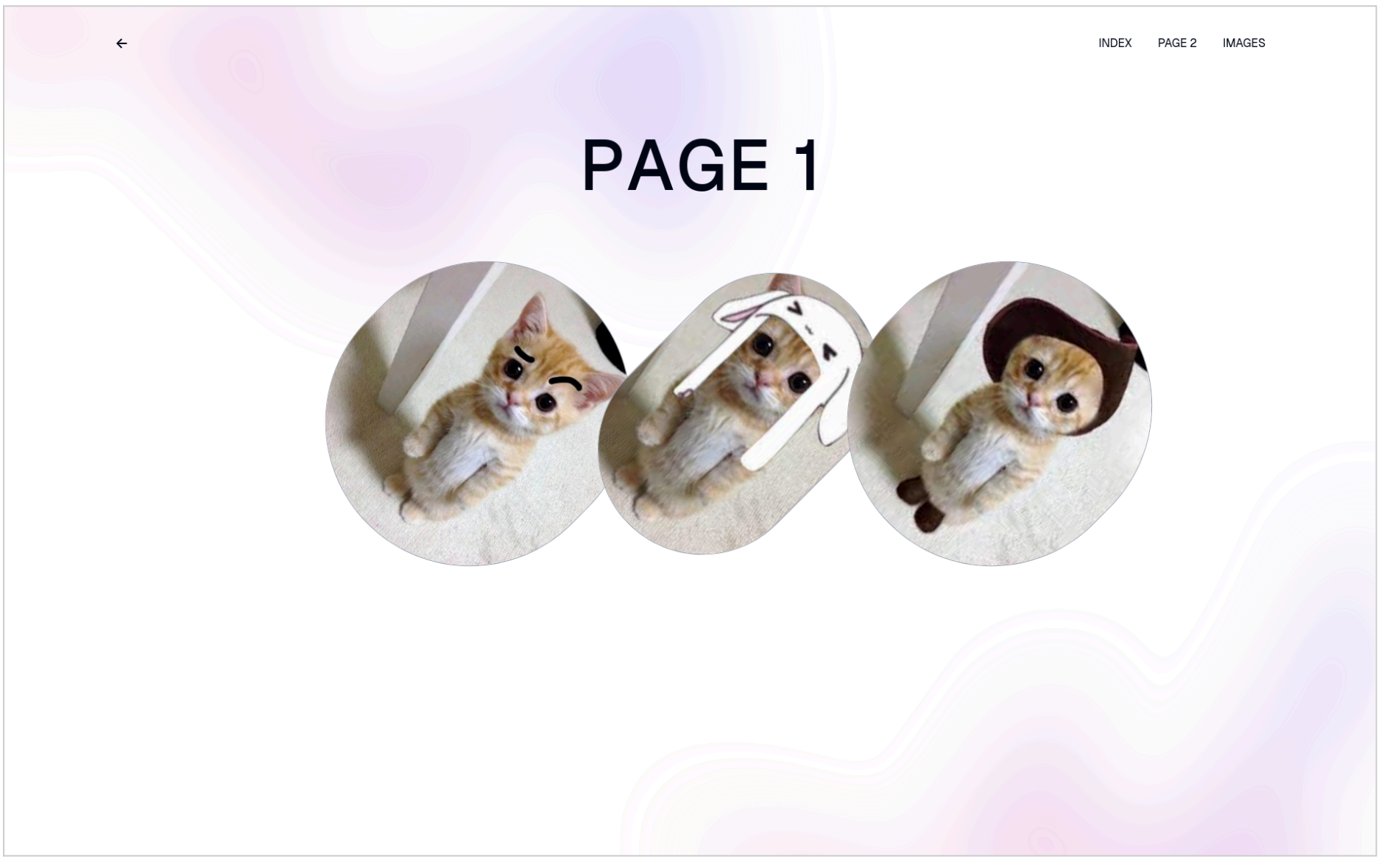
The animations stays the same as `index.html`:

```
    <script src="https://cdn.jsdelivr.net/npm/gsap@3.13.0/dist/gsap.min.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/gsap@3.13.0/dist/SplitText.min.js"></script>

    <script>
    |   document.addEventListener("DOMContentLoaded", () => {
    |       const title = document.querySelector(".hero-title");
    |       const split = new SplitText(title, { type: "chars" });
    |
    |       // hero title split text animation
    |       gsap.from(split.chars, {
    |           opacity: 0,
    |           y: 50,
    |           rotateX: 90,
    |           stagger: 0.05,
    |           ease: "back.out(1.7)",
    |           duration: 1,
    |       });
    |
    |       // navigation links fly in
    |       gsap.from(gsap.utils.toArray(".nav-link"), {
    |           y: -20,
    |           opacity: 0,
88  |           stagger: 0.1,
    |           delay: 0.3,
    |           duration: 0.6,
    |           ease: "power2.out"
    |       });
    |
    |       // scale up them kittens
    |       gsap.from(gsap.utils.toArray("img"), {
    |           opacity: 0,
    |           stagger: 0.2,
    |           delay: 0.3,
    |           scale:0.2,
    |           duration: 0.6,
    |       });
    |   });
    </script>
</html>
index.html
"index.html" 104L, 5351B
[0] 0:nvim* 1:bunx-
```

Screenshot showing animation in page1.html, exactly the same as index.html

That's pretty much it for `page1.html`, as we've only changed the links and rotated the image. Here's how it looks now running locally on my machine:



## Page2.html:

This page also shares a similar story to `page1.html`. Again, the only differences are the page title, updated navigation (which now links to page1.html and index.html), and the images – which I rotated 135 degrees to give the page a different visual feel compared to both the `index.html` and `page1.html`. The following is the only difference, removed lines are marked ( - ) and added lines are marked ( + )

```
+++ page2.html  2025-05-12 18:10:01.269679160 +0545
@@ -4,7 +4,7 @@
        <meta charset="UTF-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1.0" />
        <link rel="stylesheet" href="/static/output.css" />
-       <title>Index 2431342</title>
+       <title>Page2 2431342</title>
    </head>
    <body class="m-4 h-screen overflow-hidden">

@@ -25,10 +25,10 @@
                <div class="flex-1 justify-end flex">
                    <ul class="menu menu-horizontal px-1 flex gap-3">
                        <li>
-                           <a class="text-base font-medium uppercase nav-link tooltip tooltip-bottom" data-tip="Meow Rotate 45" href="/page1.html">Page 1</a>
+                           <a class="text-base font-medium uppercase nav-link tooltip tooltip-bottom" data-tip="Cat No Rotate" href="/index.html">Index</a>
                        </li>
                        <li>
-                           <a class="text-base font-medium uppercase nav-link tooltip tooltip-bottom" data-tip="Meow Rotate 135" href="/page2.html">Page 2</a>
+                           <a class="text-base font-medium uppercase nav-link tooltip tooltip-bottom" data-tip="Meow Rotate 45" href="/page1.html">Page 1</a>
                        </li>
                        <li>
                            <details>
@@ -46,13 +46,14 @@
        </div>


+
        <!-- Page Content -->
        <div class="h-2/3 w-screen flex-col flex justify-center gap-20 items-center">
-           <h1 class="text-8xl font-bold uppercase tracking-wide hero-title"> Index Page </h1>
+           <h1 class="text-8xl font-bold uppercase tracking-wide hero-title"> Page 2</h1>
            <div class="flex gap-4">
-               <img src="/static/images/image1.jpg" class="w-100 border border-gray-400 rounded-full"/>
-               <img src="/static/images/image3.jpg" class="border border-gray-400 rounded-full"/>
-               <img src="/static/images/image2.jpg" class="w-100 border border-gray-400 rounded-full"/>
+               <img src="/static/images/image1.jpg" class="w-100 border border-gray-400 rounded-full rotate-135"/>
+               <img src="/static/images/image3.jpg" class="border border-gray-400 rounded-full rotate-135"/>
+               <img src="/static/images/image2.jpg" class="w-100 border border-gray-400 rounded-full rotate-135"/>
            </div>
        </div>
:
[0] 0:nvim  1:bunx- 2:bash*
```

Screenshot showing differences between page2.html and index.html using diff piped to bat for clarity.

The animations again are unchanged:

```
<script src="https://cdn.jsdelivr.net/npm/gsap@3.13.0/dist/gsap.min.js"></script>
<script src="https://cdn.jsdelivr.net/npm/gsap@3.13.0/dist/SplitText.min.js"></script>

<script>
    document.addEventListener("DOMContentLoaded", () => {
        const title = document.querySelector(".hero-title");
        const split = new SplitText(title, { type: "chars" });

        // hero title split text animation
        gsap.from(split.chars, {
            opacity: 0,
            y: 50,
            rotateX: 90,
            stagger: 0.05,
            ease: "back.out(1.7)",
            duration: 1,
        });

        // navigation links fly in
        gsap.from(gsap.utils.toArray(".nav-link"), {
            y: -20,
            opacity: 0,
            stagger: 0.1,
            delay: 0.3,
            duration: 0.6,
            ease: "power2.out"
        });

        // scale up them kittens
        gsap.from(gsap.utils.toArray("img"), {
            opacity: 0,
            stagger: 0.2,
            delay: 0.3,
            scale:0.2,
            duration: 0.6,
        });
    });
    </script>
</html>
index.html
"index.html" 104L, 5351B
[0] 0:nvim* 1:bunx-
```

Screenshot showing animation in page2.html, exactly the same as index.html

That's also it for `page2.html`. Here is how the final version looks locally running on my machine:

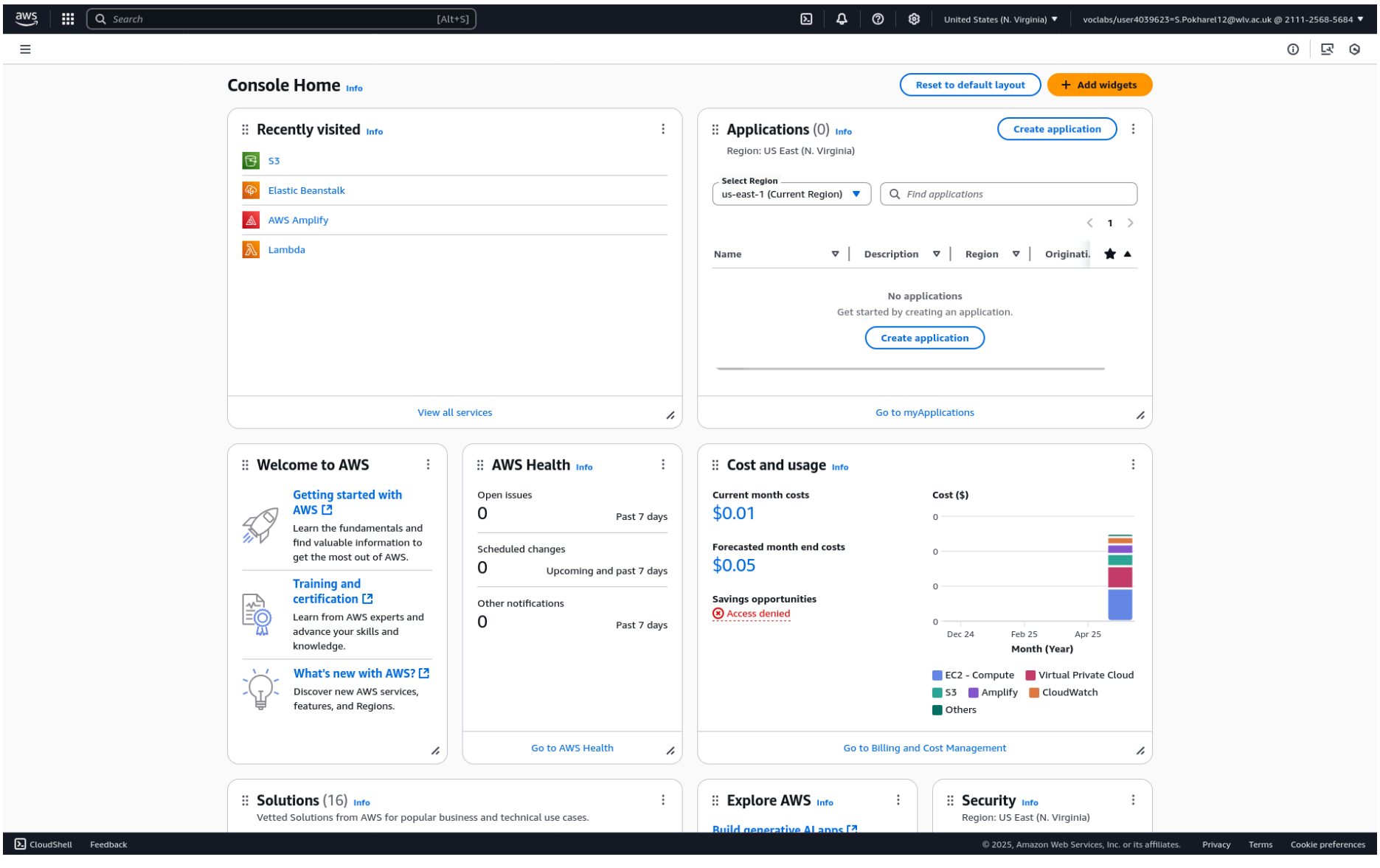
That will be all for the coding side of things, now moving on to hosting.

## 4. Deployment:

### S3 Bucket & Static Hosting:

We will firstly be uploading the files and creating a static website on S3.

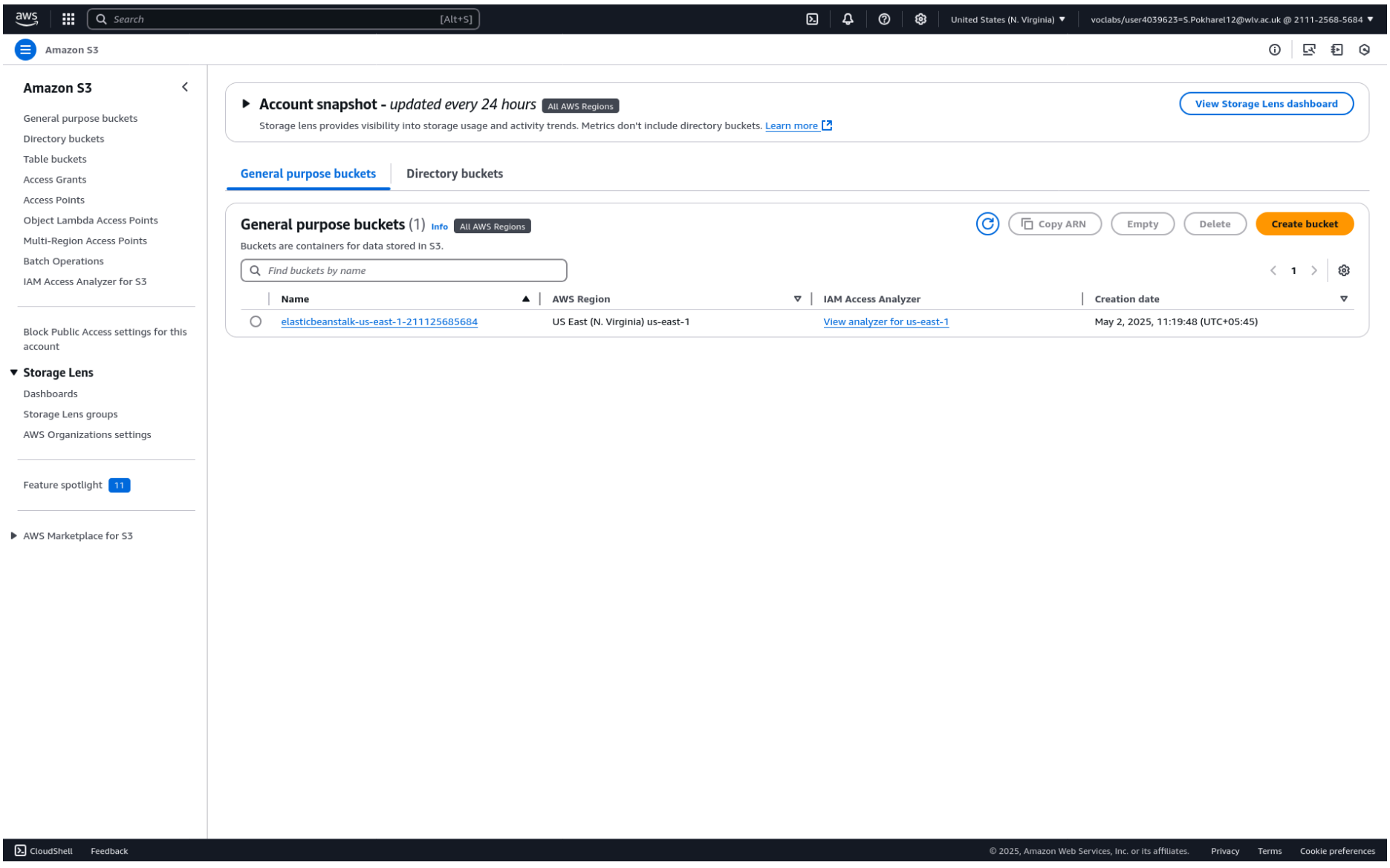
#### I: Navigating To AWS:




Screenshot showing my AWS landing page

Conveniently enough, there is a link to S3 in my recently visited tab. We now navigate to Amazon S3.

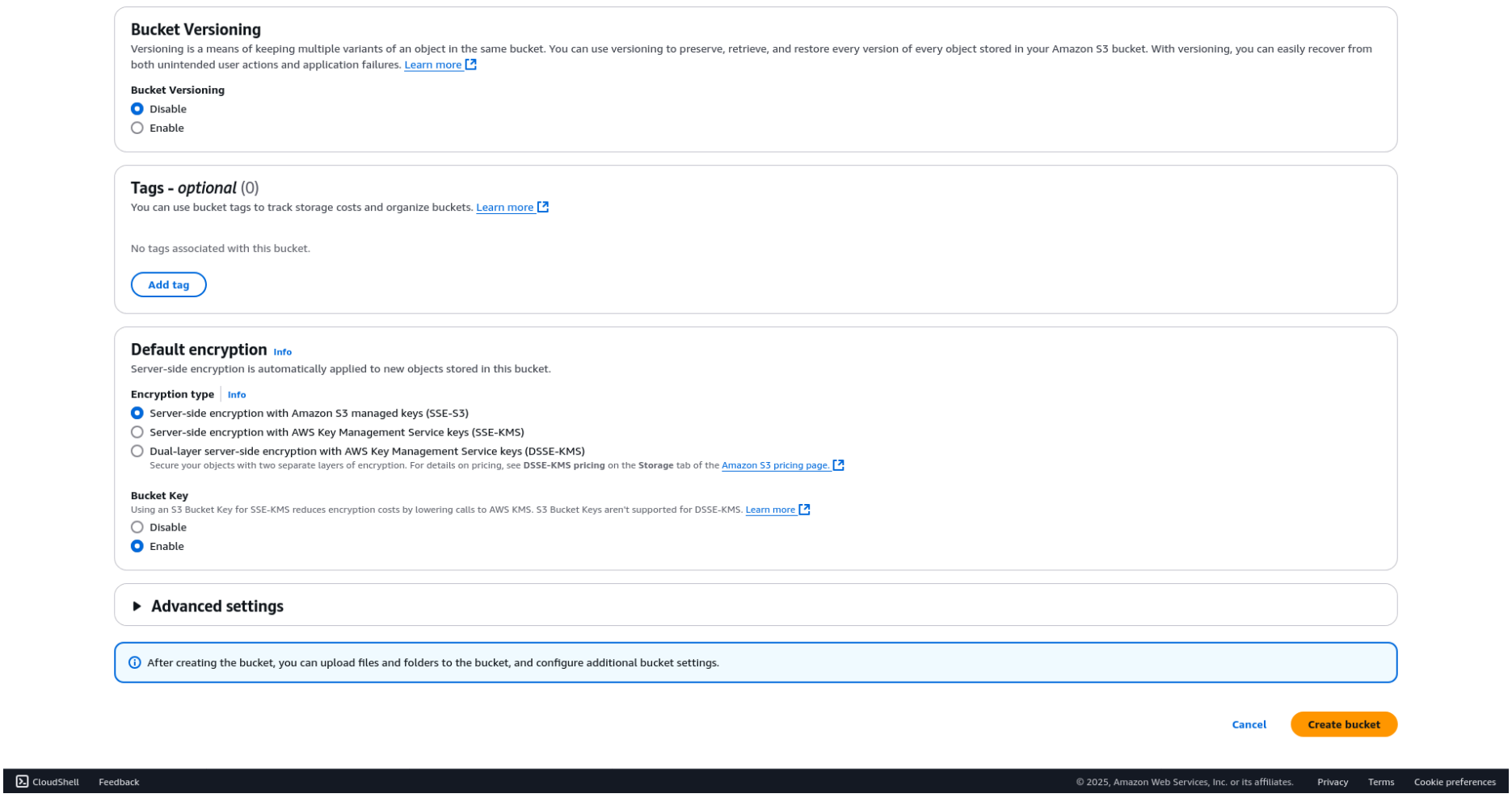
## II: Navigating To S3:




Screenshot showing my buckets

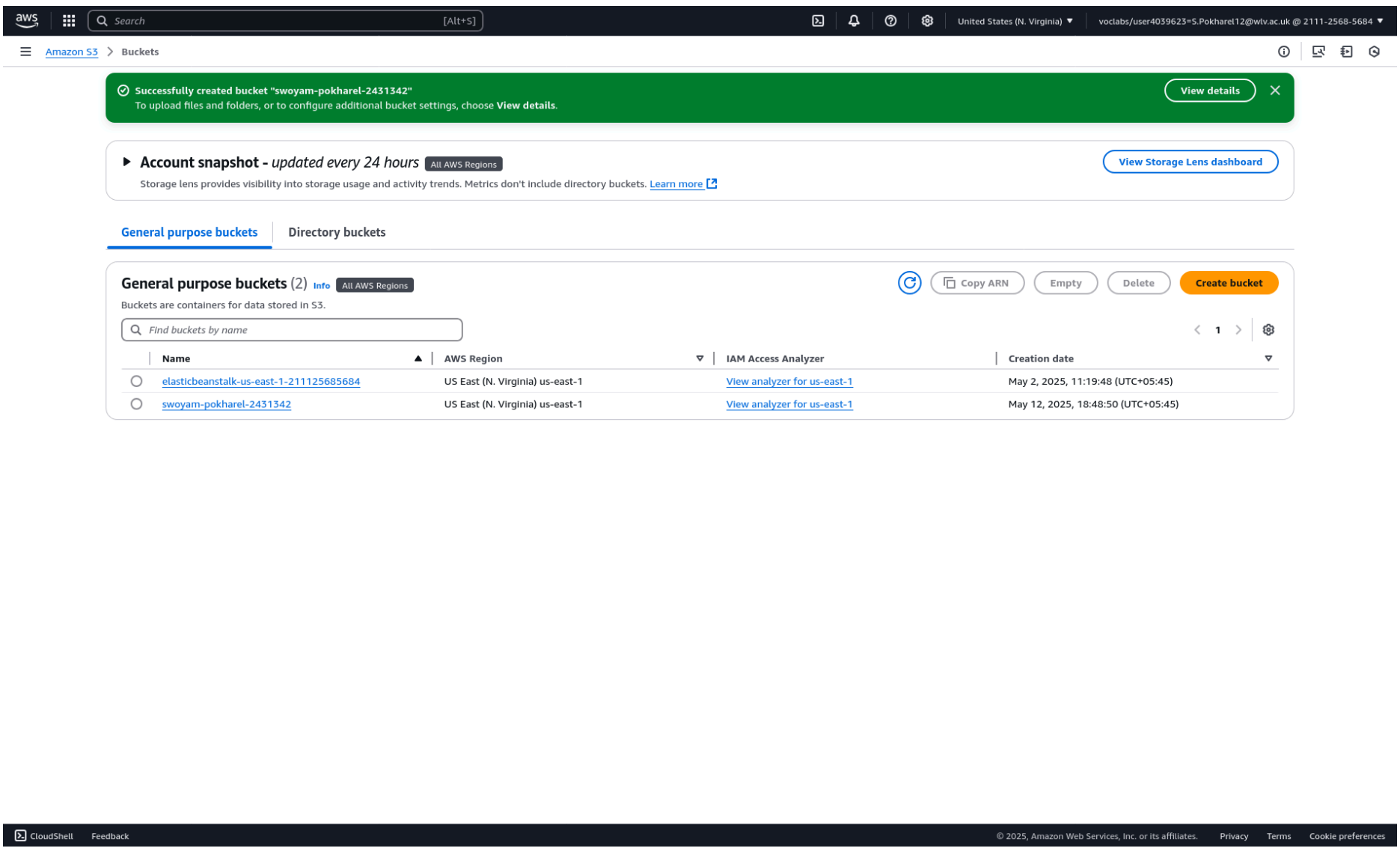
Here, we can see that we have an instance of Elastic Beanstalk running, which we won't be needing. We will be creating a new bucket for this specific task.

## III: Creating a new bucket




Screenshot of the bucket creation page with filled inputs

Here, we specify the name of our bucket, in my case `swoyam-pokharel-2431342`; we enable ACLs and uncheck `Block Public Access`, because we want our statically hosted website to be accessible to anyone. We then acknowledge that change and leave the rest of options as is.

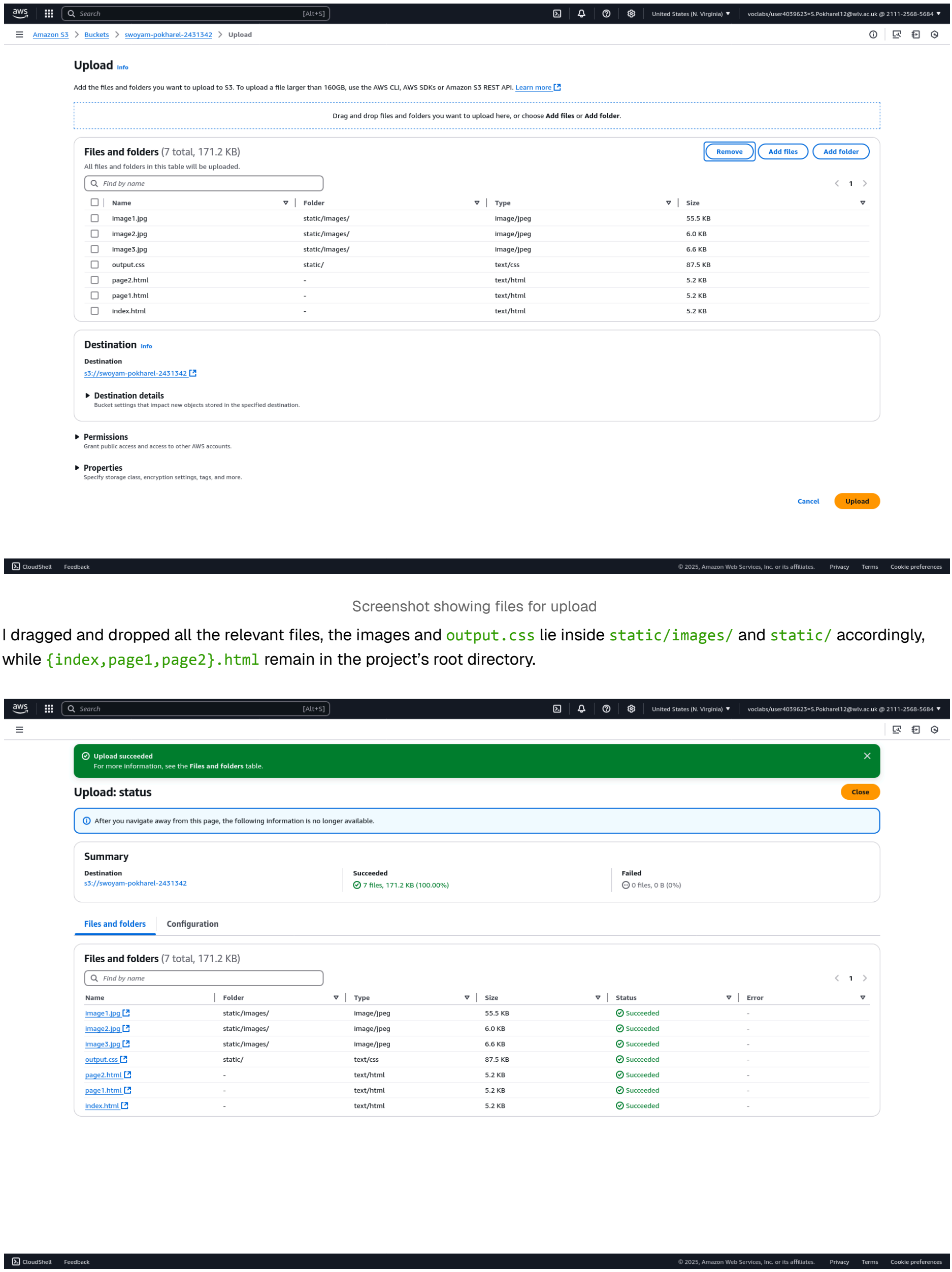



Screenshot Showing Successful Bucket Creation

Here, we can see that our bucket has been successfully created and now we may proceed to upload relevant files

## IV: Uploading Relevant Files

In our case, we would like to upload all 3 of our html pages, all of the images and our `output.css` file. We do exactly that:




Screenshot showing files for upload

I dragged and dropped all the relevant files, the images and `output.css` lie inside `static/images/` and `static/` accordingly, while `{index,page1,page2}.html` remain in the project's root directory.

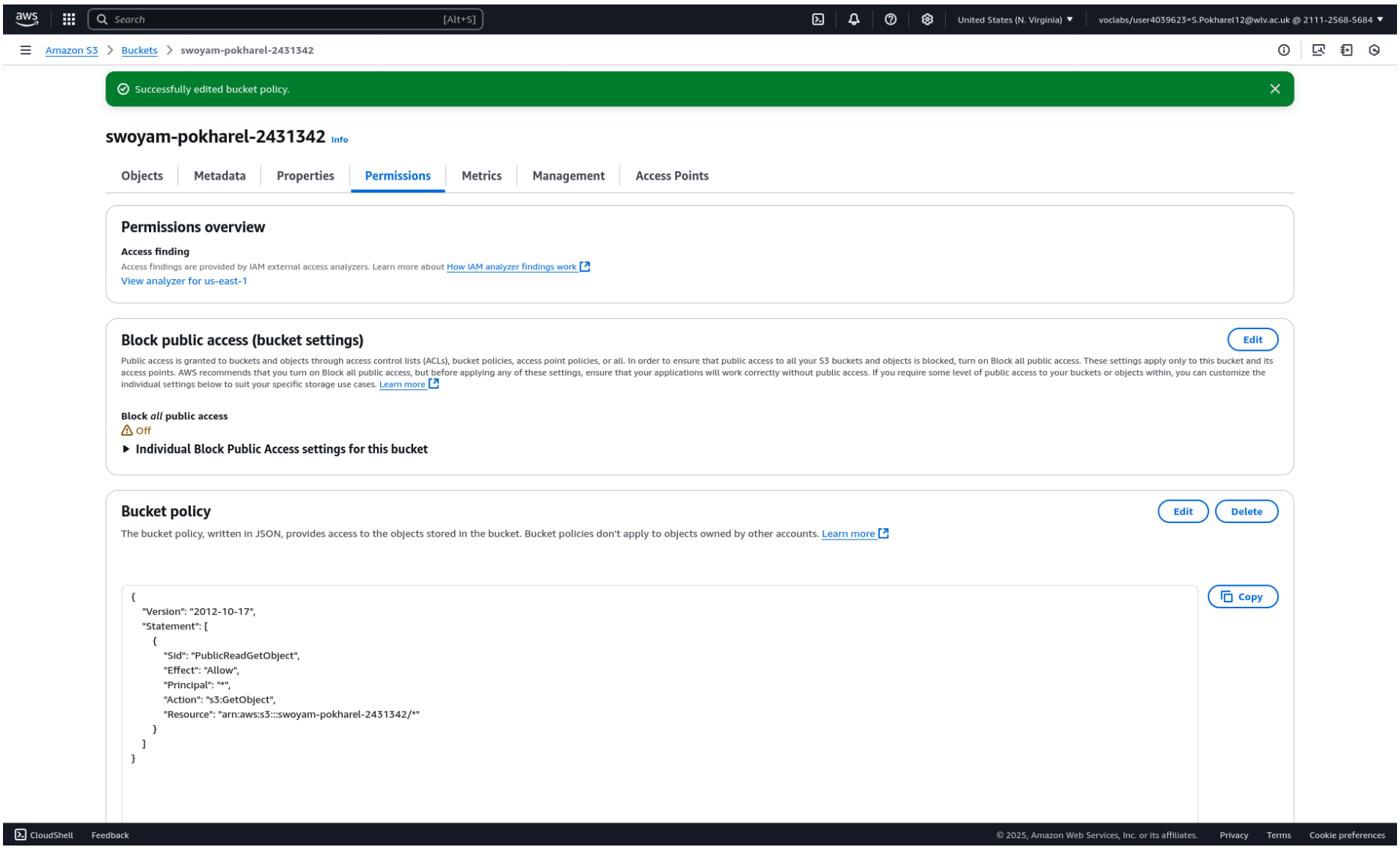
Screenshot showing successful upload

Here, we simply get confirmation that our files have been uploaded and now we may move on to editing the bucket policy.

### V: Editing The Bucket Policy:

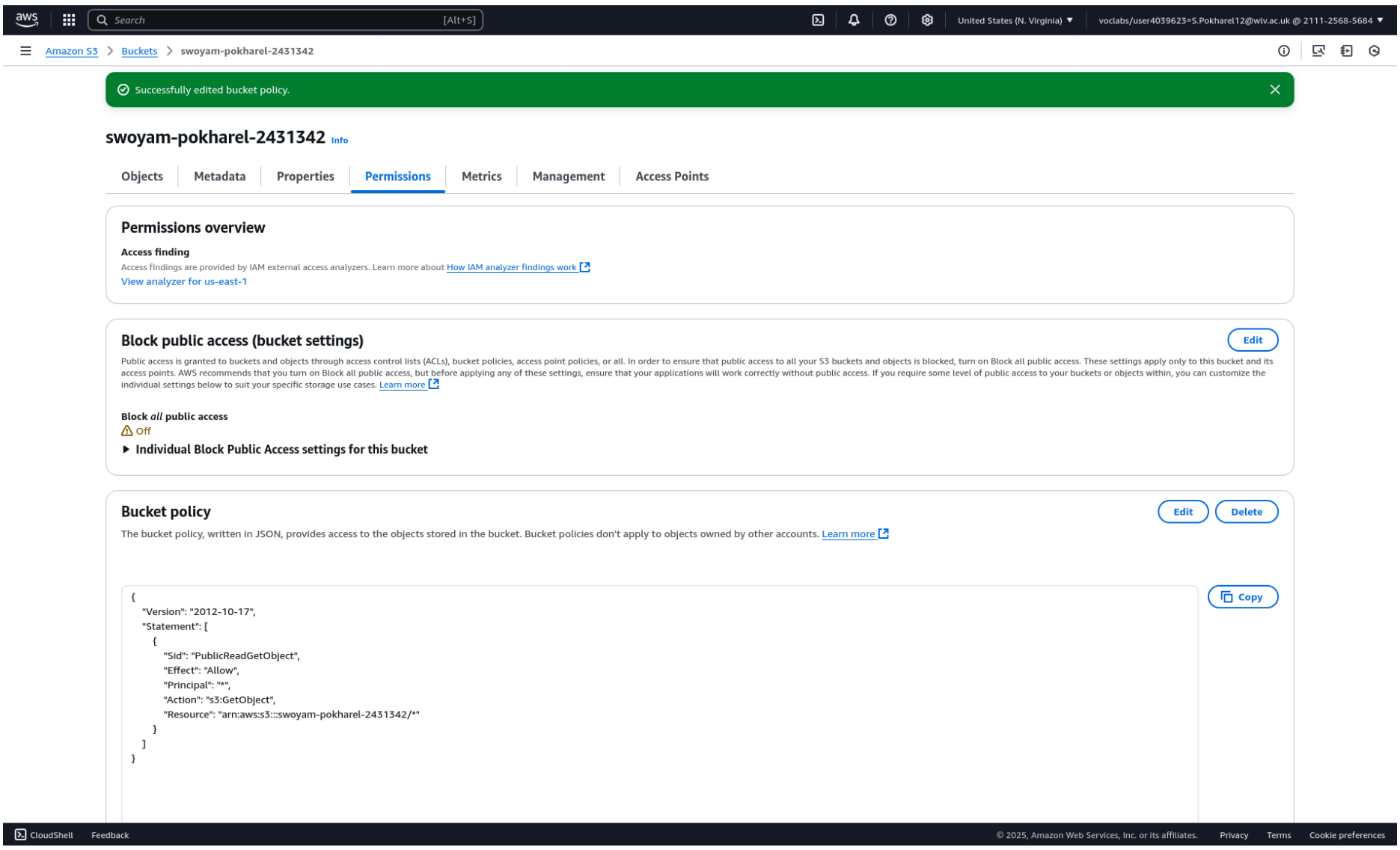
We now proceed to edit the bucket policy to:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
    "Sid": "PublicReadGetObject",
    "Principal": "*",
    "Effect": "Allow",
    "Action": [
        "s3:GetObject"
    ],
    "Resource": [
        "arn:aws:s3:::swoyam-pokharel-2431342/*"
    ]
    }
  ]
}
```




Screenshots showing editing bucket policy.

We edit the bucket policy to allow public read access to the objects in the S3 bucket. In this case, the policy grants the `s3:GetObject` permission to everyone by using the wildcard `Principle: “*”`, meaning that everyone can access the files within the specified bucket. This is typically done when hosting a publicly accessible website like ours with static assets like images, CSS and HTML files in an S3 bucket. WIthout this policy, the default permissions would block public access to those files.

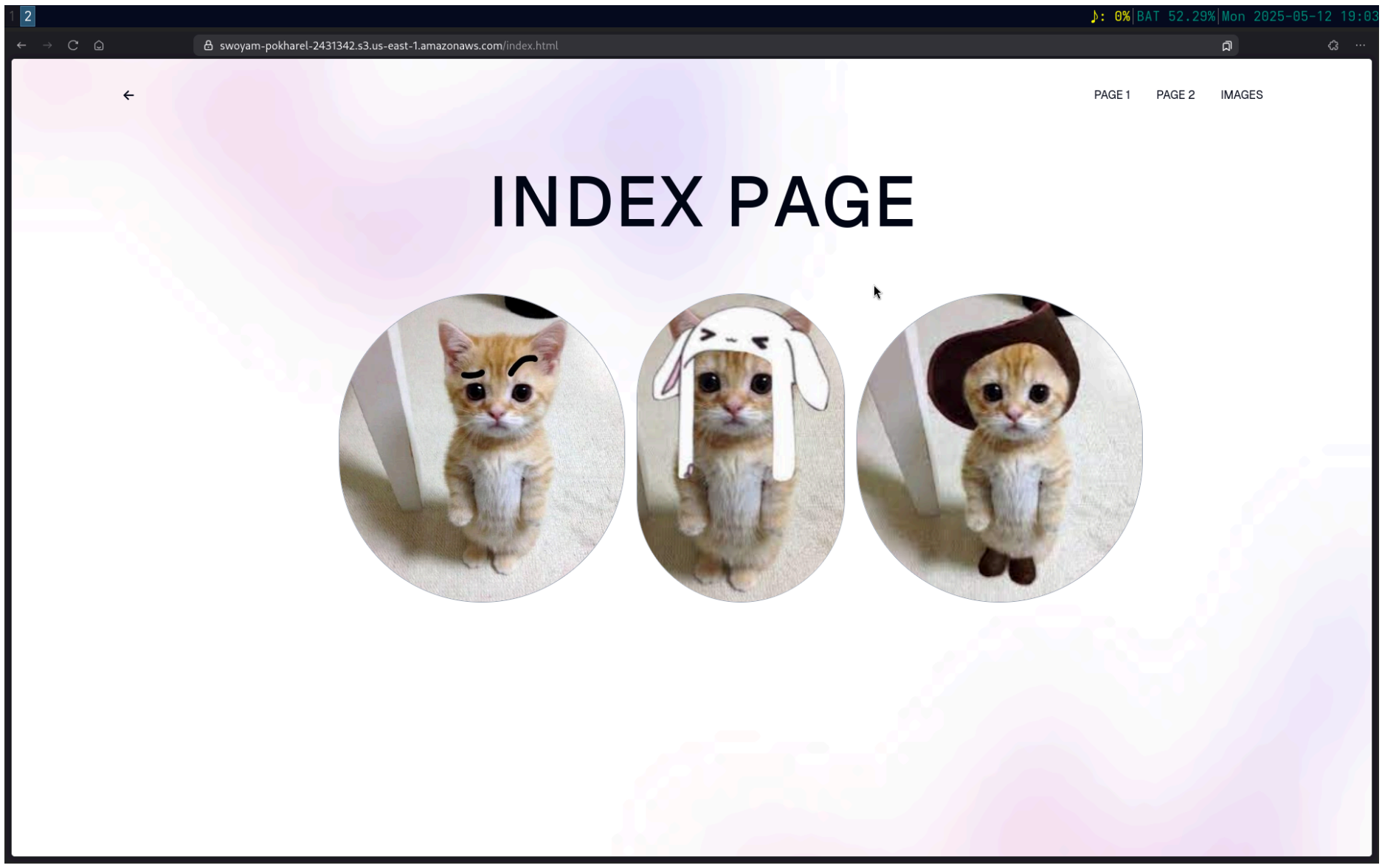



Screenshot showing confirmation

Here, we just get confirmation that our bucket policy has changed, which implies that now we can access the files inside this bucket via their object url. Let’s see if that’s the case in the next step.

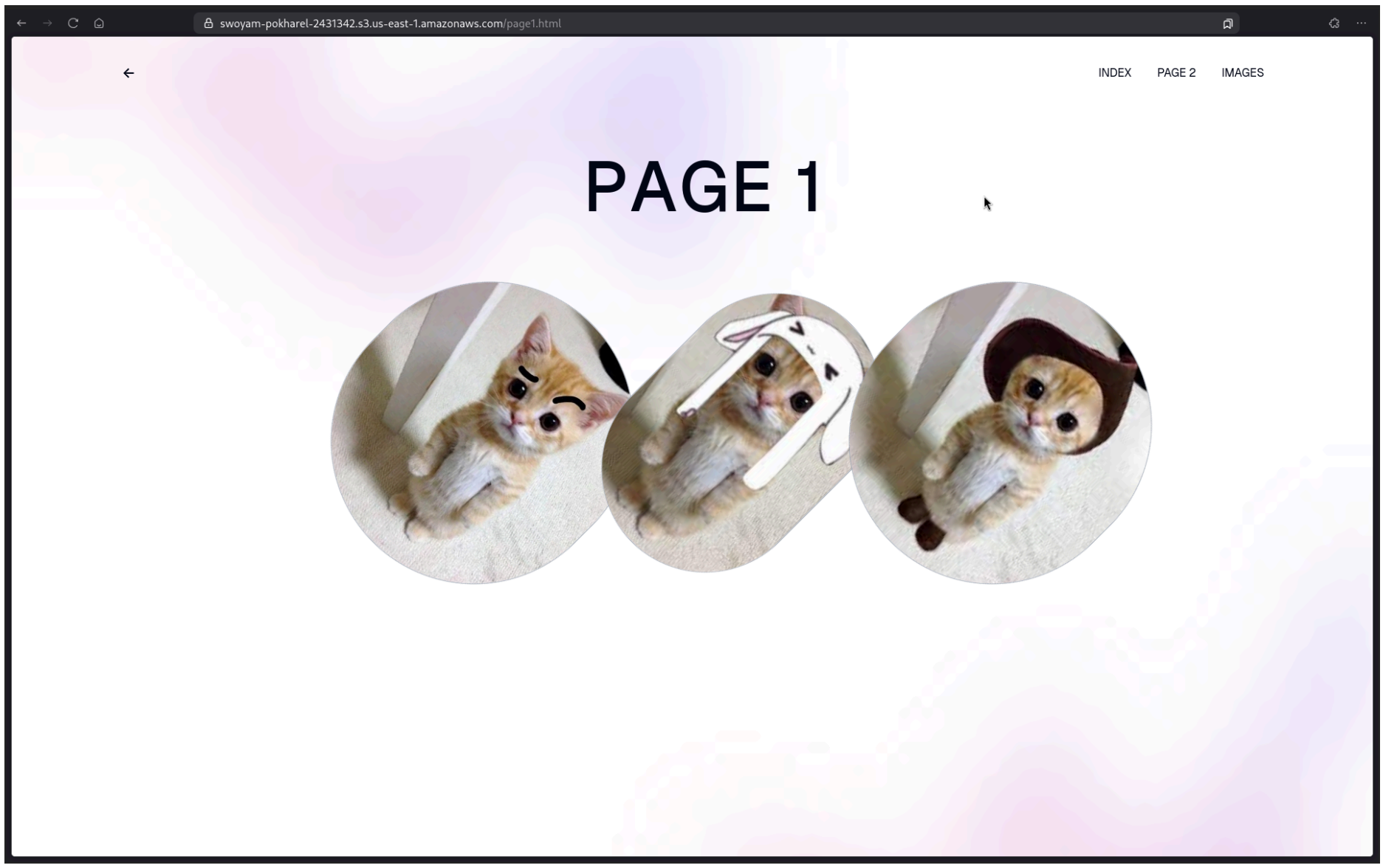
## VI: Checking If Our Files Are Accessible Via Object URL:

So navigating to the link S3 gave us we can see the following:

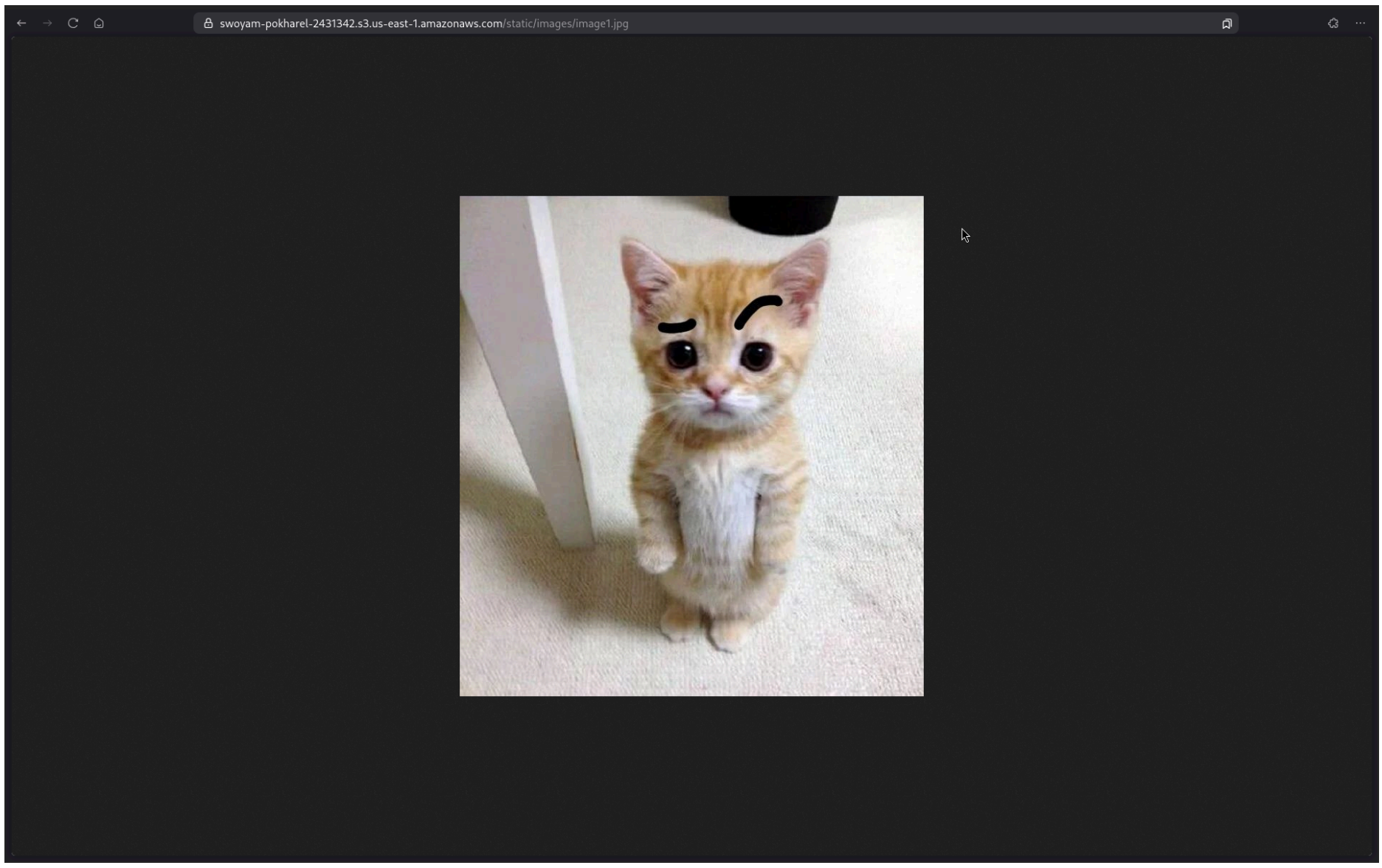



Screenshot showing index.html accessible via S3 Object URL

So, our bucket policy worked and now we can access `index.html` via the object url. Similarly the rest of the pages and the images are also accessible:




Screenshot showing page1.html accessible via S3 Object URL

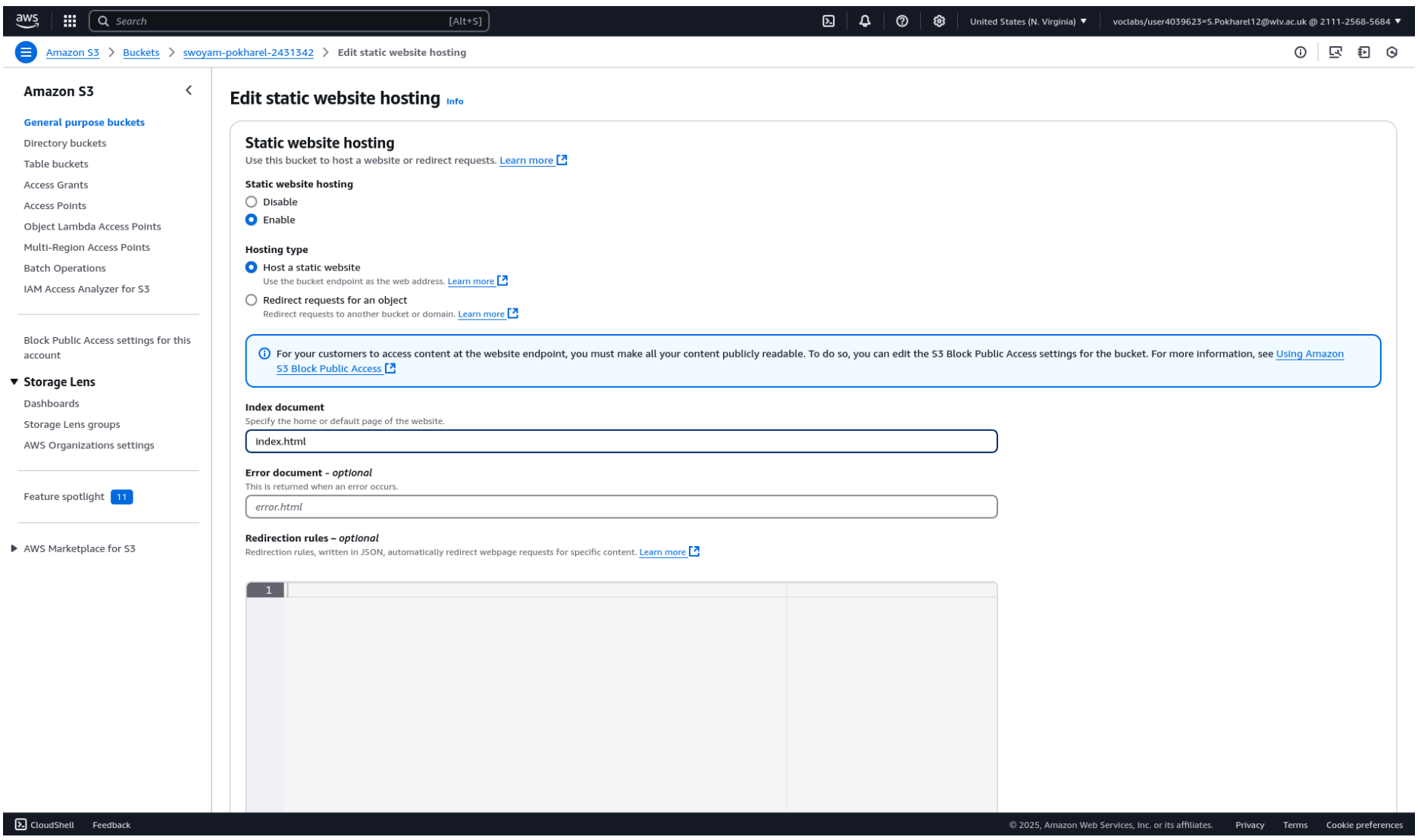



Screenshot showing images accessible via S3 Object URL

So now that it's evident that the bucket policy worked and we can access the objects in the S3 bucket, we proceed to enable static hosting in this bucket.

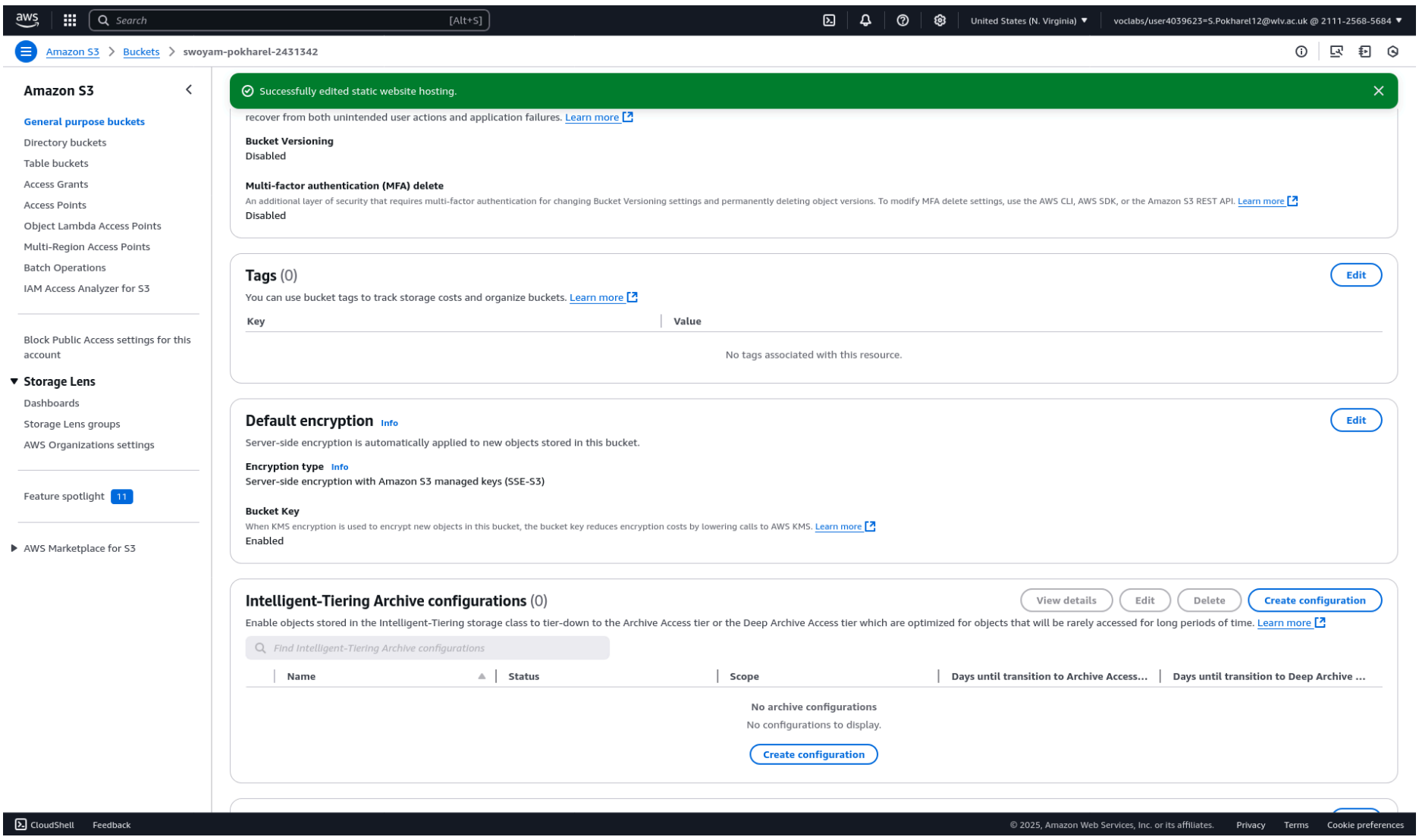
### VII: Enabling Static Hosting:

Enabling static hosting is as simple as navigating all the way down in the properties tab and clicking enable static hosting. This brings us to the following page:



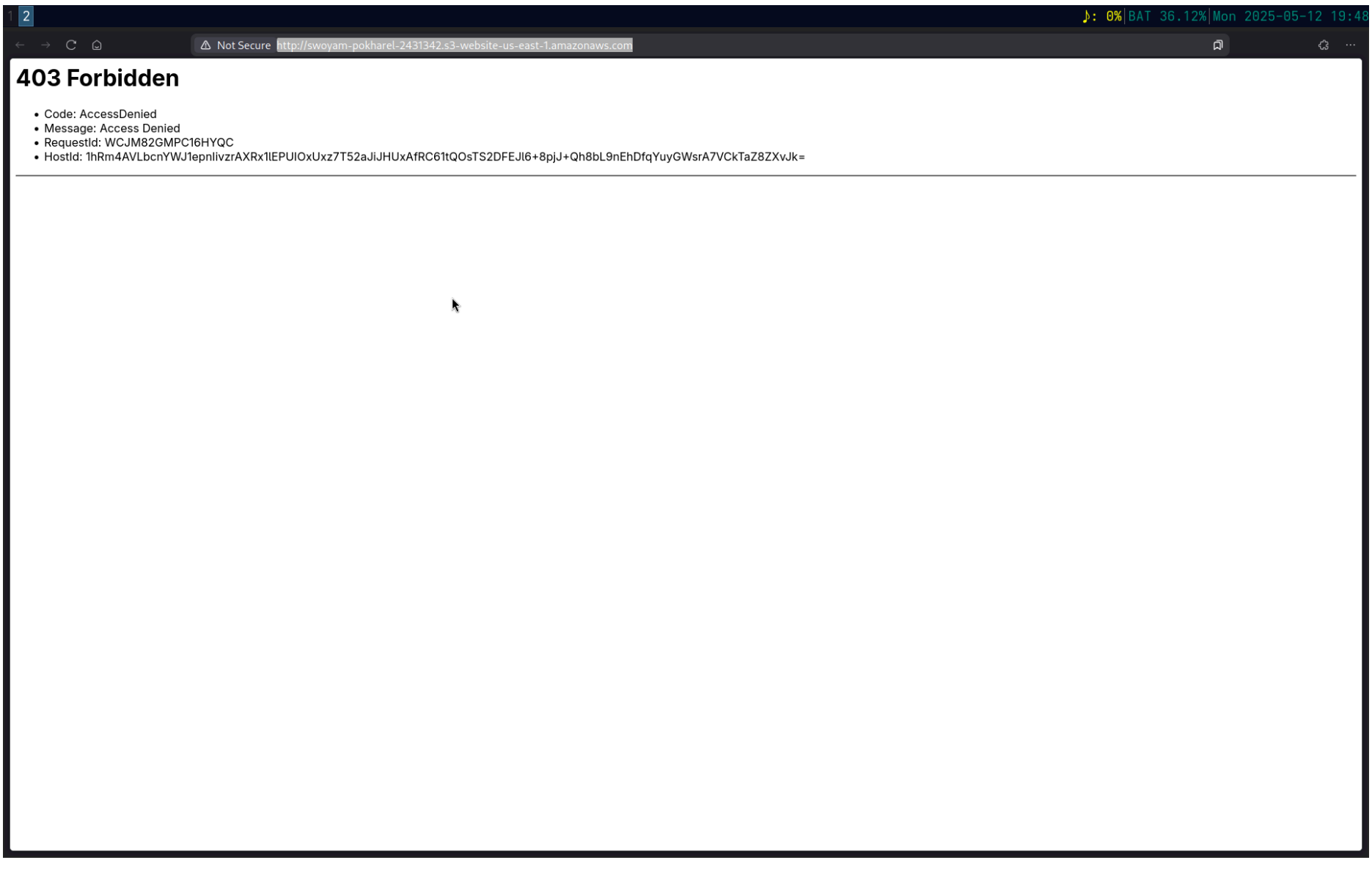

Screenshot Of Static Hosting Page

Here, we enable static website hosting and set `index.html` as the index document. This tells S3 which file to serve by default when someone accesses the bucket's static website URL. We also have the option to specify the error page, which we leave empty in this case. Now we simply save the changes:



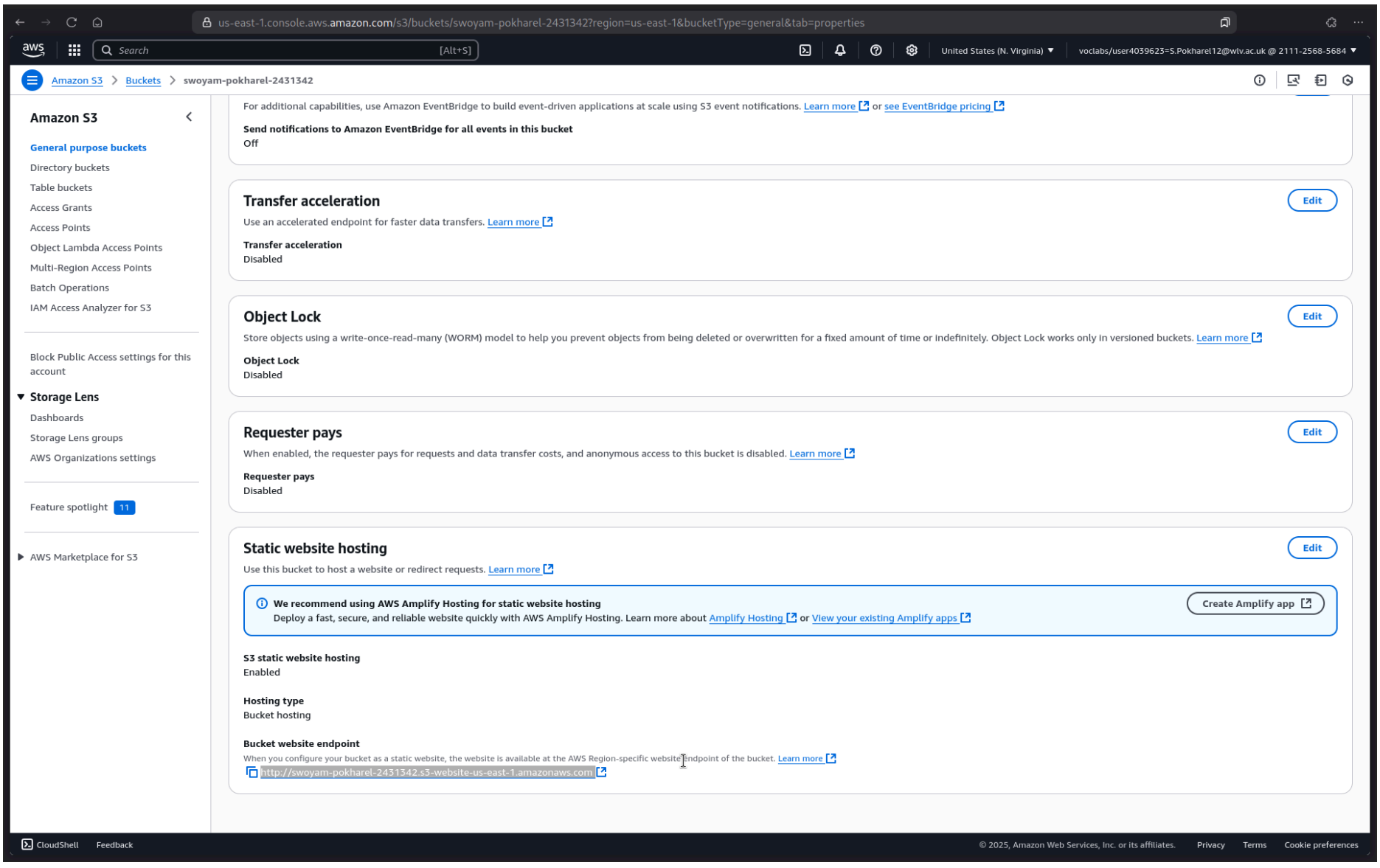

Screenshot showing successful save

Now that that's done, we should be able to navigate to the provided url, but in doing so, we get hit with a



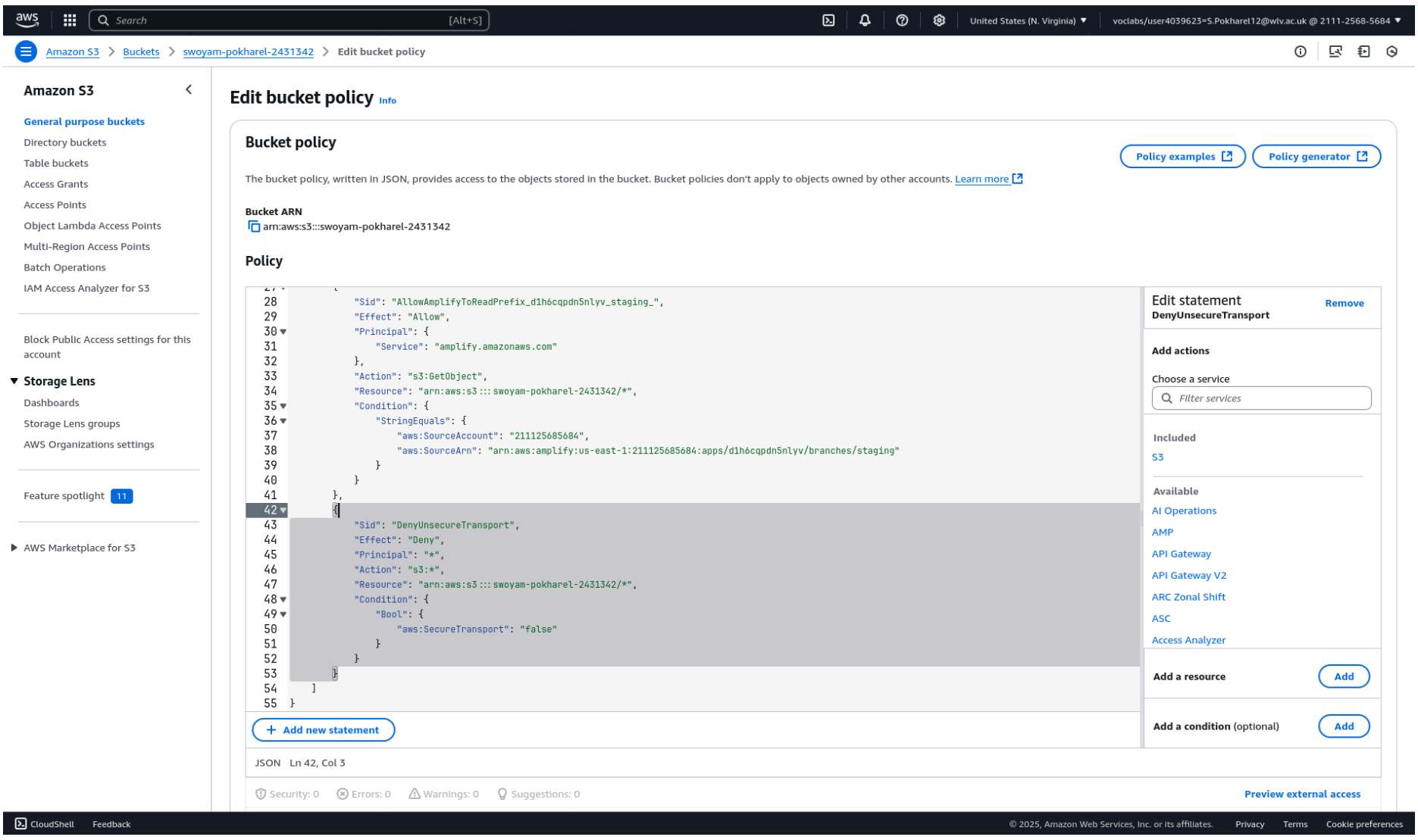

Screenshots showing 403 forbidden

For some reason, we get hit with a `403 Unauthorized`, Why is that? Well I did a little digging, and if we look closely to the highlighted link,




Screenshot showing http link to statically hosted S3 bucket

We can see that the S3 statically hosted link is at: [http://swoyam-pokharel-2431342.s3-website-us-east-1.amazonaws.com](http://swoyam-pokharel-2431342.s3-website-us-east-1.amazonaws.com) which is `http`. The `http` in itself isn't the problem. The problem is that when S3 later populated the Bucket Policy, it added in:




Screenshot of bucket policy that was causing the problem

If we look closely, it requires `aws::SecureTransport`, or in simpler words `https`. So the bucket policy was wanting `https` but the link to the static site itself was in `http`, that's our problem right there. Thankfully, it's an easy fix, simply removing that section from the json did the trick. Now, navigating to the url where our S3 bucket is statically hosted, we get:

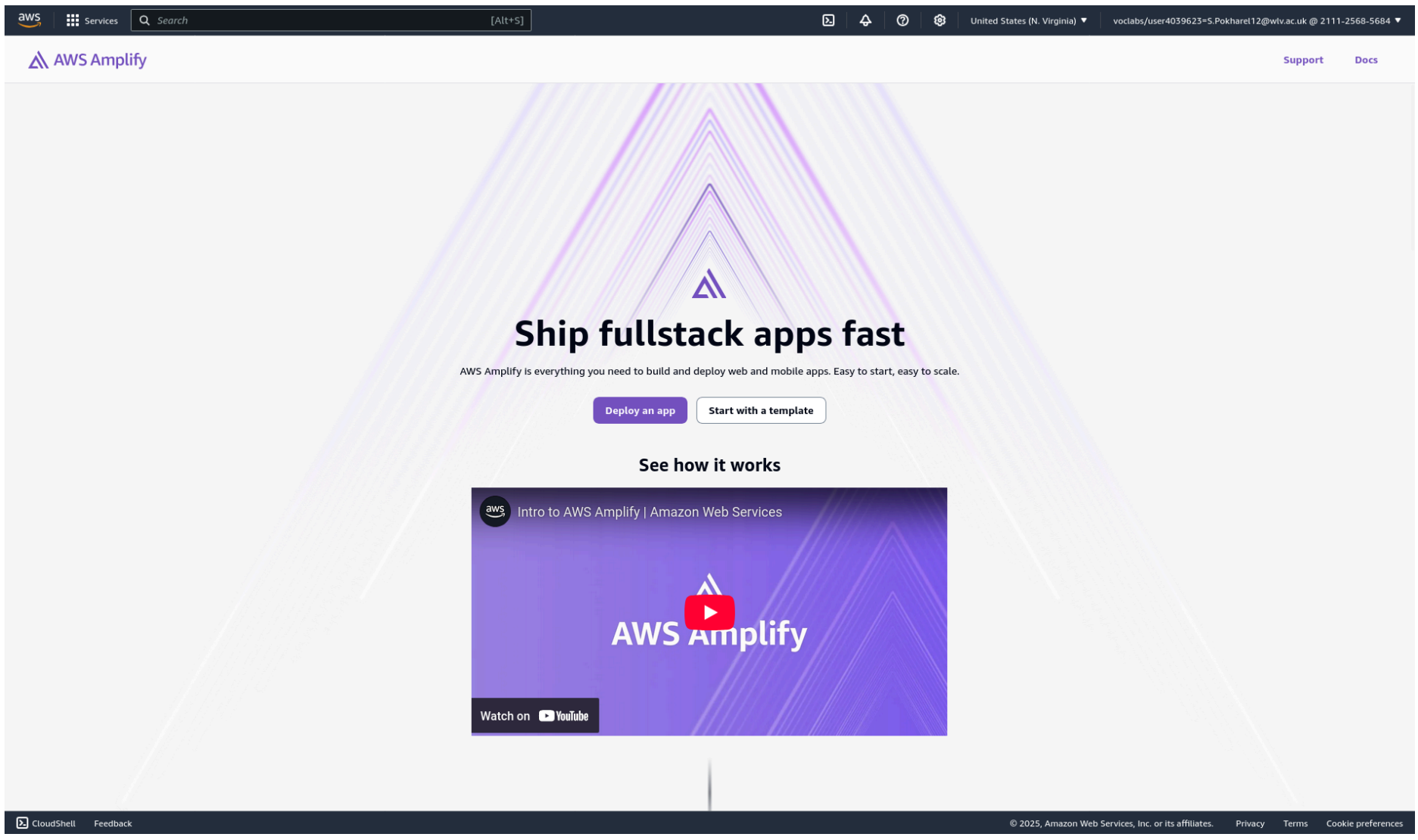

Successful static website hosting

Now that we have successfully hosted the static site, time to move on to amplify hosting.

## AWS Amplify

Hosting on amplify is pretty straightforward, firstly we:

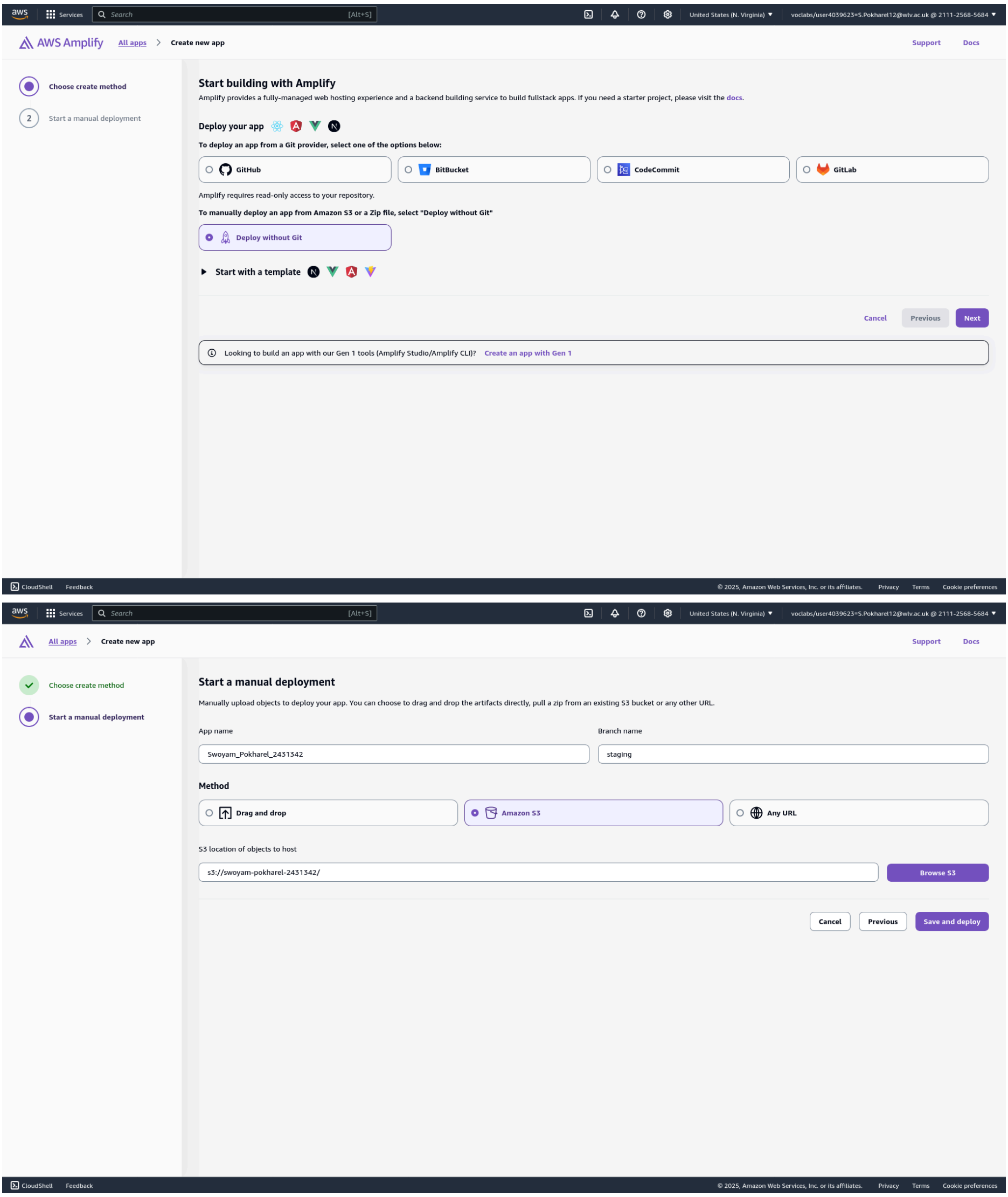
### I: Navigating To AWS Amplify




Screenshot of Amplify Landing Page

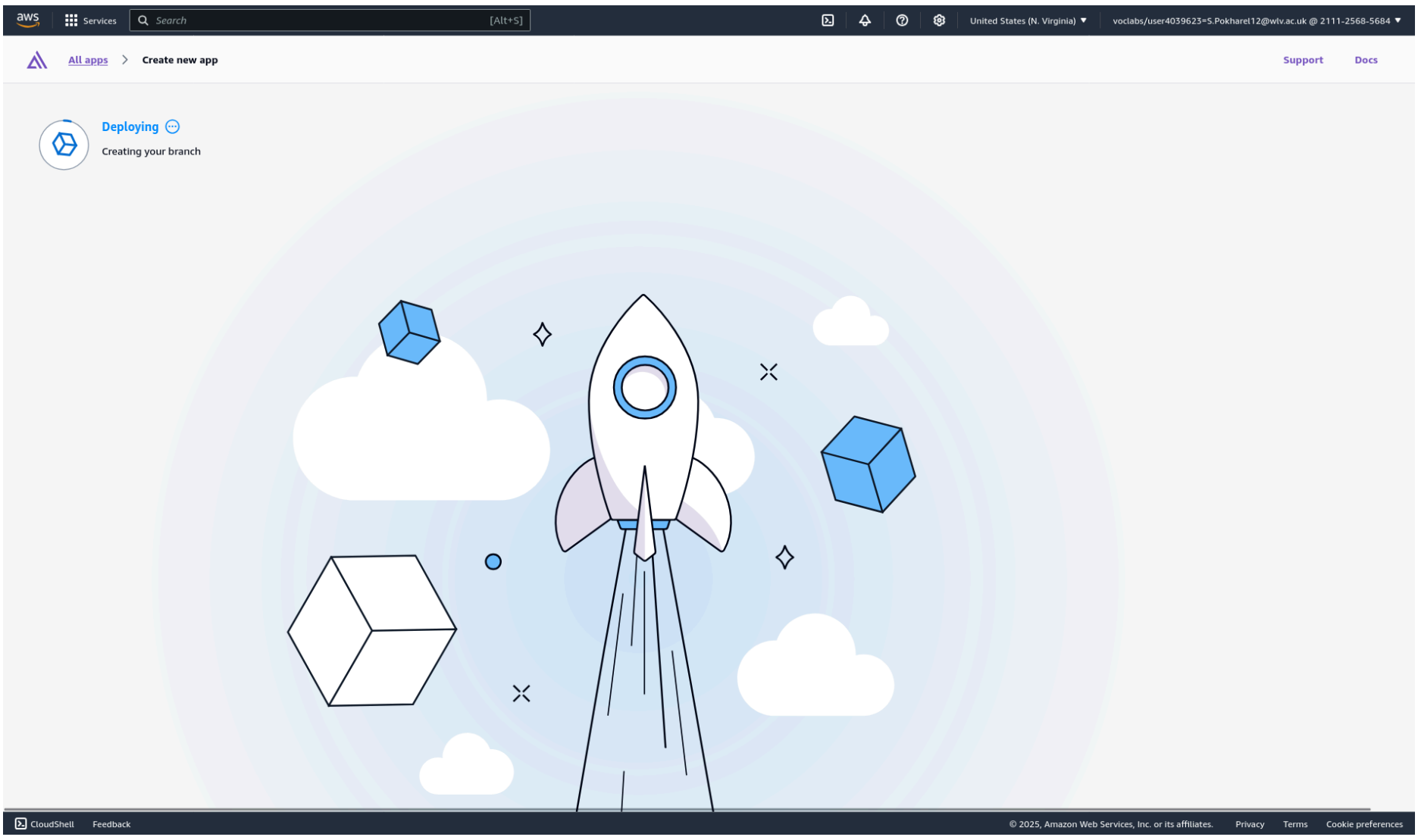
We navigate to the landing page of aws amplify and click on deploy an app

## II: Deploying Amplify App

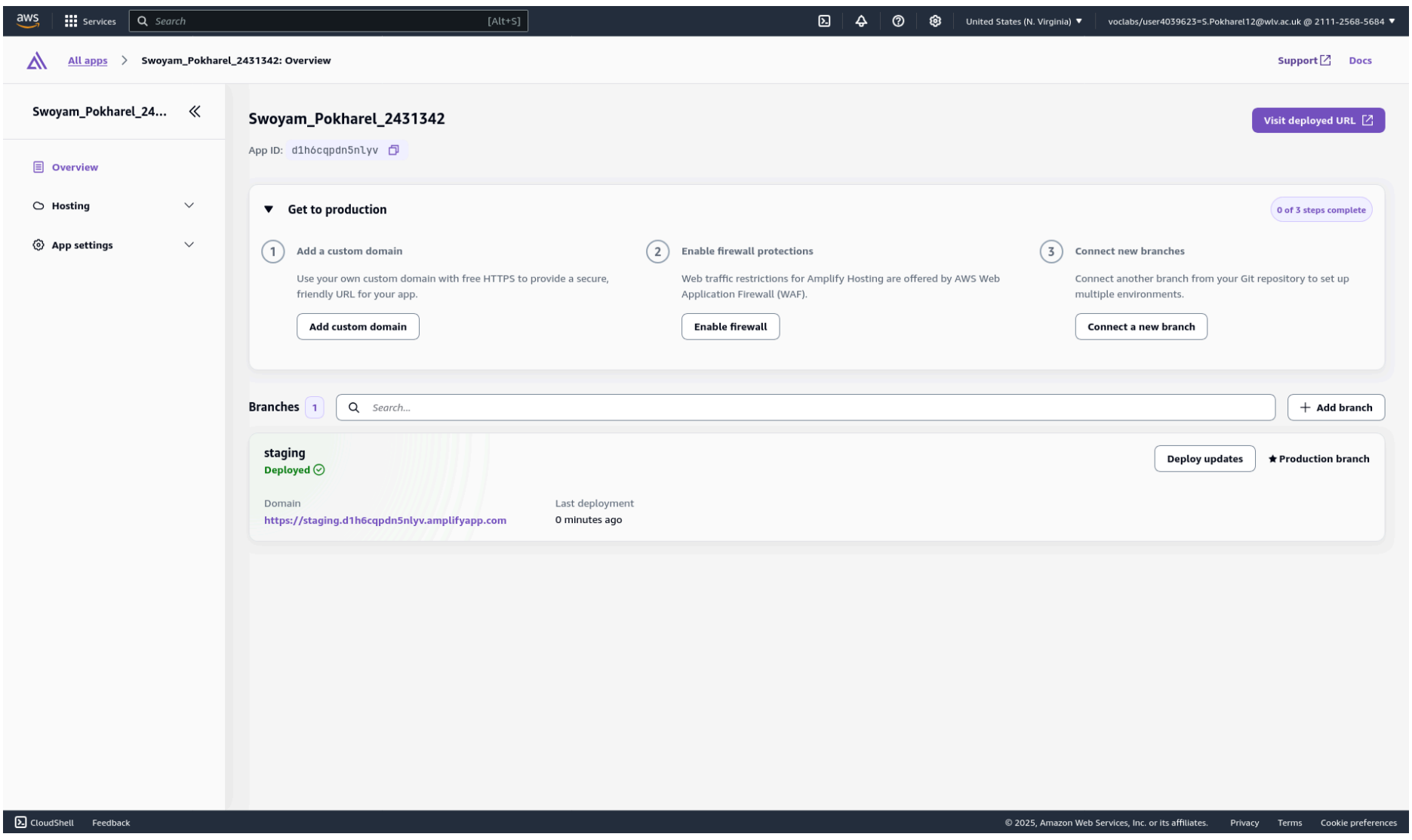



Screenshots of the `create new app` page

Here, we simply choose to deploy without git and select Amazon S3 as our method to upload the files, and as simple as that, we are now deploying on aws amplify

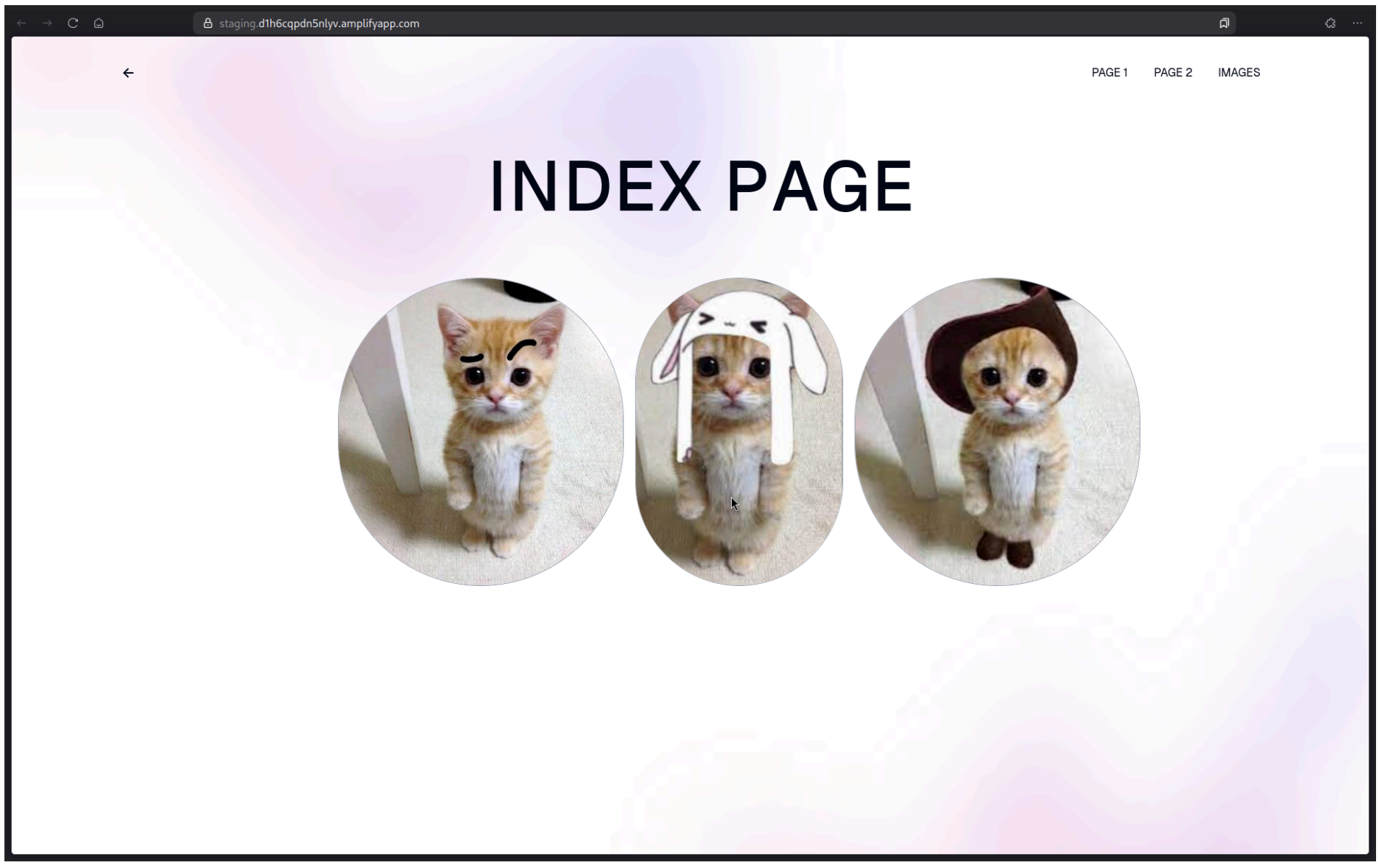



Screenshot showing deployment in progress




Screenshot showing successful deployment

And we are done, we have successfully deployed our application. We can navigate to the provided amplify link to view our webpage.




Screenshot showing hosted website in amplify

All of the pages work but the only problem we have now is the navigation takes us to page1.html page2.html and the images hosted in amplify.

### III: Updating Code & Changing Navigation

The question specifies that `index.html` should be the only non-Amplify link. However, since we're using Amazon S3 as the source for our Amplify deployment, there's no way to exclude specific files from being served via the Amplify domain. Amplify simply maps a domain to the entire contents of the S3 bucket. It doesn't duplicate files; it just exposes them all under its own domain.

To meet the requirement while working within these constraints, I've structured our links like this:

- `index.html` is loaded using the S3 static hosting URL (the only non-Amplify link, as required).
- Inside `index.html`, links to page1.html and page2.html use the Amplify domain.
- `page1.html` links back to `index.html` using the S3 link, and to `page2.html` using a relative link `/page2.html` so it stays within the Amplify domain.
- `page2.html` also links back to `index.html` via the S3 link, and to `page1.html` using a relative link.
- Images are referenced using relative paths `/static/images/...` so that:
    - If you're on index.html (S3 domain), images load from S3.
    - If you're on page1.html or page2.html (Amplify domain), images load through Amplify.

This setup ensures that:

- Only index.html is accessed through a non-Amplify link.
- Page navigation works across both domains.
- Images load correctly respective to the domain you're on.

So, now we can act accordingly:

```
<div class="px-30 navbar sticky top-0 z-50 bg-transparent backdrop-blur-md">
    <div class="flex justify-between items-center w-full">
        <!-- Left -->
        <div class="flex-1">
            <a class="btn btn-ghost text-xl nav-link tooltip tooltip-bottom" data-tip="Navigate Back!" onclick="window.history.back()">&larr;</a>
        </div>

        <!-- Right -->
        <div class="flex-1 justify-end flex">
            <ul class="menu menu-horizontal px-1 flex gap-3">
                <li>
                    <a class="text-base font-medium uppercase nav-link tooltip tooltip-bottom" data-tip="Meow Rotate 45" href="https://staging.d1h6cqpdn5nlyv.amplifyapp.com/
                    page1.html">Page 1</a>
                </li>
                <li>
                    <a class="text-base font-medium uppercase nav-link tooltip tooltip-bottom" data-tip="Meow Rotate 135" href="https://staging.d1h6cqpdn5nlyv.amplifyapp.com/
                    page2.html">Page 2</a>
                </li>
                <li>
                    <details>
                        <summary class="text-base font-medium uppercase nav-link tooltip tooltip-left" data-tip="Click For Links To Images on S3 Bucket">Images</summary>
                        <ul class="bg-base-100 rounded-t-none p-2 uppercase font-bold">
                            <li><a class="text-md" href="/static/images/image1.jpg">Image-1</a></li>
                            <li><a class="text-md" href="/static/images/image2.jpg">Image-2</a></li>
                            <li><a class="text-md" href="/static/images/image3.jpg">Image-3</a></li>
                        </ul>
                    </details>
                </li>
            </ul>
        </div>
    </div>
```

Screenshot Of Updated Navigation In Index.html

Here we change the navigation in the `index.html` to point to the amplify links for `page1.html` and `page2.html`

```
<!-- Navigation Bar -->
<div class="px-30 navbar sticky top-0 z-50 bg-transparent backdrop-blur-md">
    <div class="flex justify-between items-center w-full">
        <!-- Left -->
        <div class="flex-1">
            <a class="btn btn-ghost text-xl nav-link tooltip tooltip-bottom" data-tip="Navigate Back!" onclick="window.history.back()">&larr;</a>
        </div>

        <!-- Right -->
        <div class="flex-1 justify-end flex">
            <ul class="menu menu-horizontal px-1 flex gap-3">
                <li>
                    <a class="text-base font-medium uppercase nav-link tooltip tooltip-bottom" data-tip="Cat No Rotate" href="http://swoyam-pokharel-2431342.s3-website-us-
                    east-1.amazonaws.com/">Index</a>
                </li>
                <li>
                    <a class="text-base font-medium uppercase nav-link tooltip tooltip-bottom" data-tip="Meow Rotate 135" href="/page2.html">Page 2</a>
                </li>
                <li>
                    <details>
                        <summary class="text-base font-medium uppercase nav-link tooltip tooltip-left" data-tip="Click For Links To Images on S3 Bucket">Images</summary>
                        <ul class="bg-base-100 rounded-t-none p-2 uppercase font-bold">
                            <li><a class="text-md" href="/static/images/image1.jpg">Image-1</a></li>
                            <li><a class="text-md" href="/static/images/image2.jpg">Image-2</a></li>
                            <li><a class="text-md" href="/static/images/image3.jpg">Image-3</a></li>
                        </ul>
                    </details>
                </li>
            </ul>
        </div>
    </div>
ml
```

Screenshot Of Updated Navigation In page1.html

Here we change the navigation in the `page1.html` to point to the S3 link for `index.html` and amplify link for `page2.html`

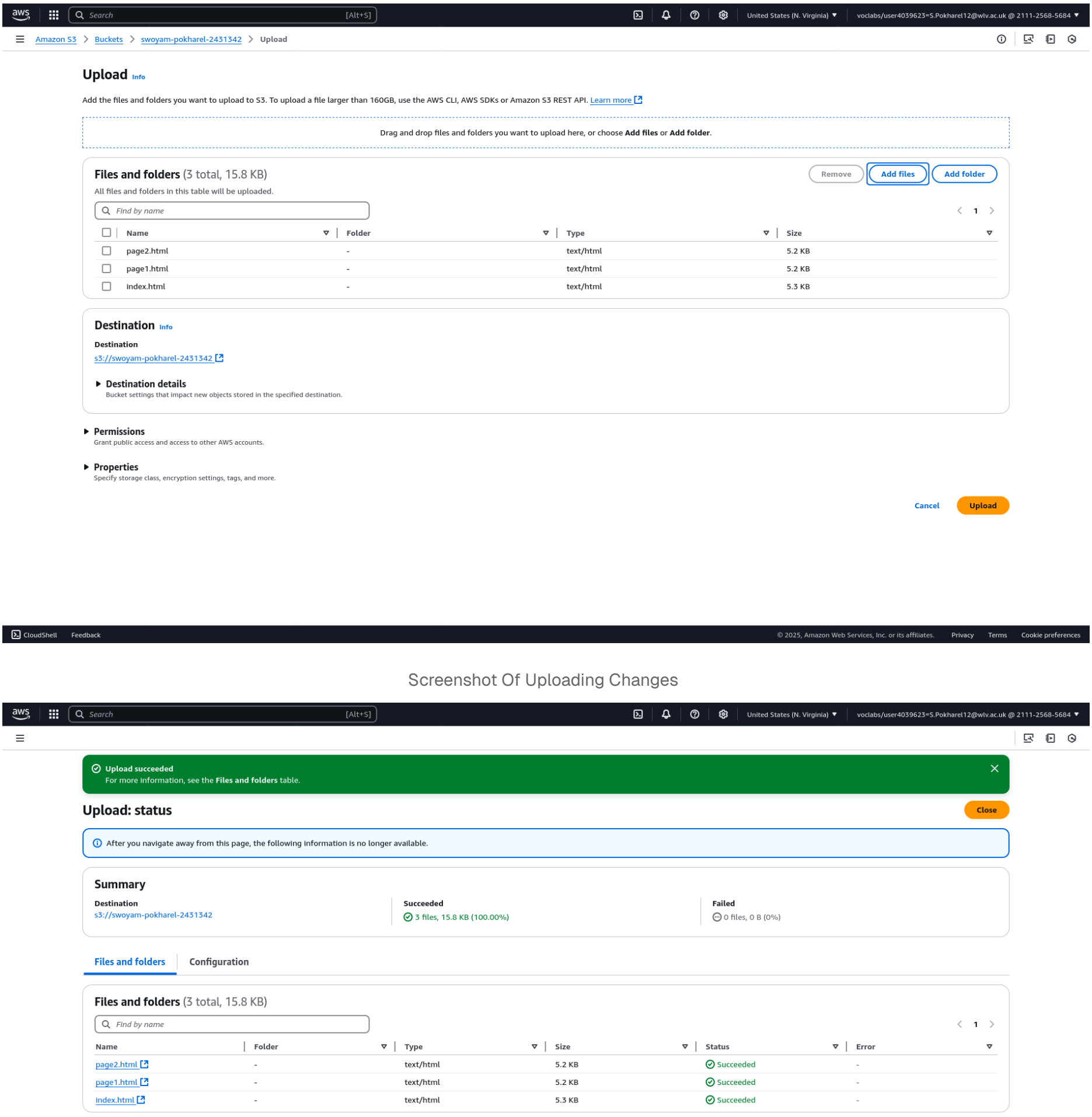
```
<!-- Navigation Bar -->
<div class="px-30 navbar sticky top-0 z-50 bg-transparent backdrop-blur-md">
    <div class="flex justify-between items-center w-full">
        <!-- Left -->
        <div class="flex-1">
            <a class="btn btn-ghost text-xl nav-link tooltip tooltip-bottom" data-tip="Navigate Back!" onclick="window.history.back()">&larr;</a>
        </div>

        <!-- Right -->
        <div class="flex-1 justify-end flex">
            <ul class="menu menu-horizontal px-1 flex gap-3">
                <li>
                    <a class="text-base font-medium uppercase nav-link tooltip tooltip-bottom" data-tip="Cat No Rotate" href="http://swoyam-pokharel-2431342.s3-website-us-
                    east-1.amazonaws.com/">Index</a>
                </li>
                <li>
                    <a class="text-base font-medium uppercase nav-link tooltip tooltip-bottom" data-tip="Meow Rotate 135" href="/page2.html">Page 2</a>
                </li>
                <li>
                    <details>
                        <summary class="text-base font-medium uppercase nav-link tooltip tooltip-left" data-tip="Click For Links To Images on S3 Bucket">Images</summary>
                        <ul class="bg-base-100 rounded-t-none p-2 uppercase font-bold">
                            <li><a class="text-md" href="/static/images/image1.jpg">Image-1</a></li>
                            <li><a class="text-md" href="/static/images/image2.jpg">Image-2</a></li>
                            <li><a class="text-md" href="/static/images/image3.jpg">Image-3</a></li>
                        </ul>
                    </details>
                </li>
            </ul>
        </div>
    </div>
</div>
ml
```

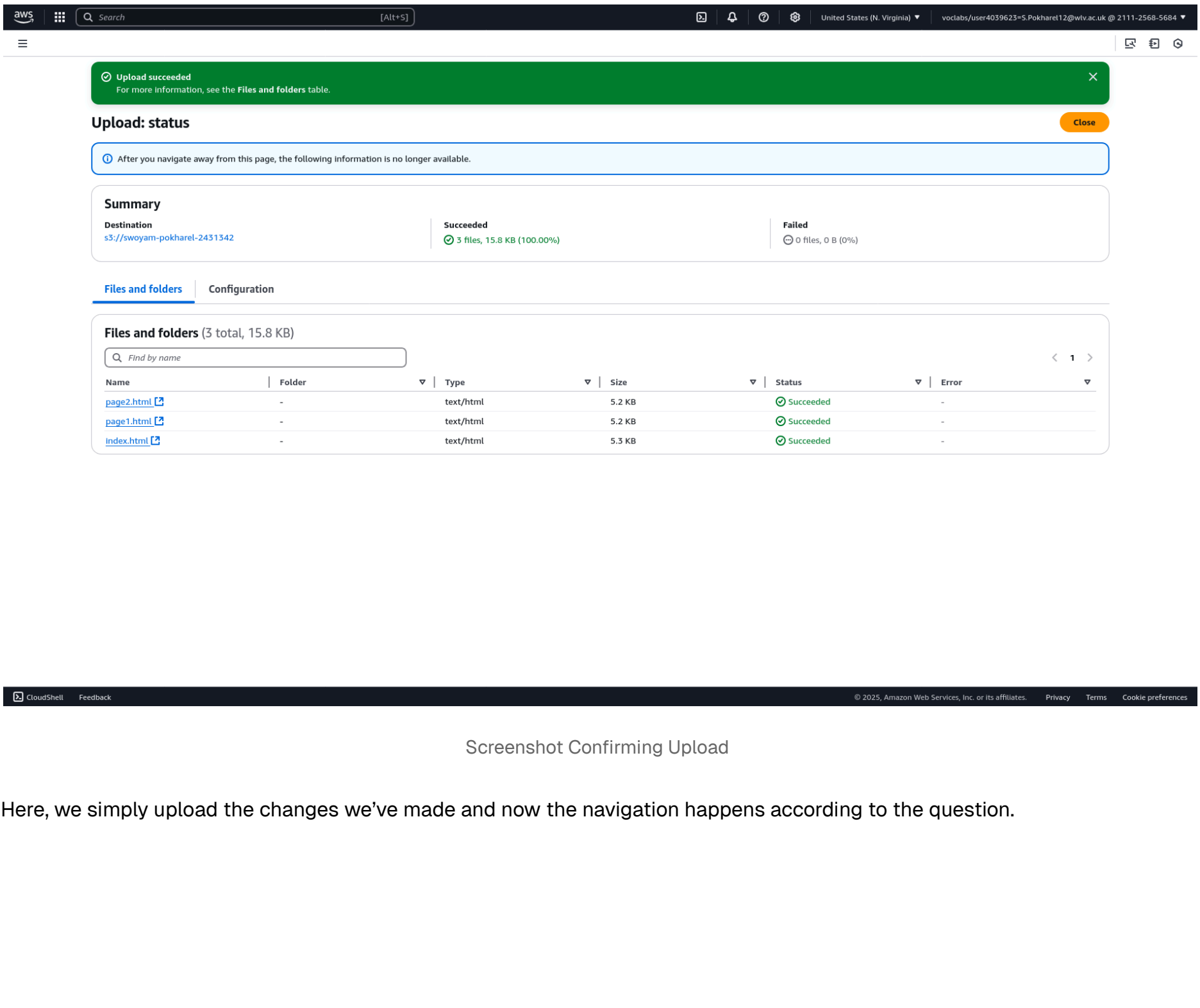
Screenshot Of Updated Navigation In page2.html

Here we change the navigation in the `page2.html` to point to the S3 link for `index.html` and amplify link for `page1.html` . Now, we upload these changes that they take effect.

## IV: Uploading Changes



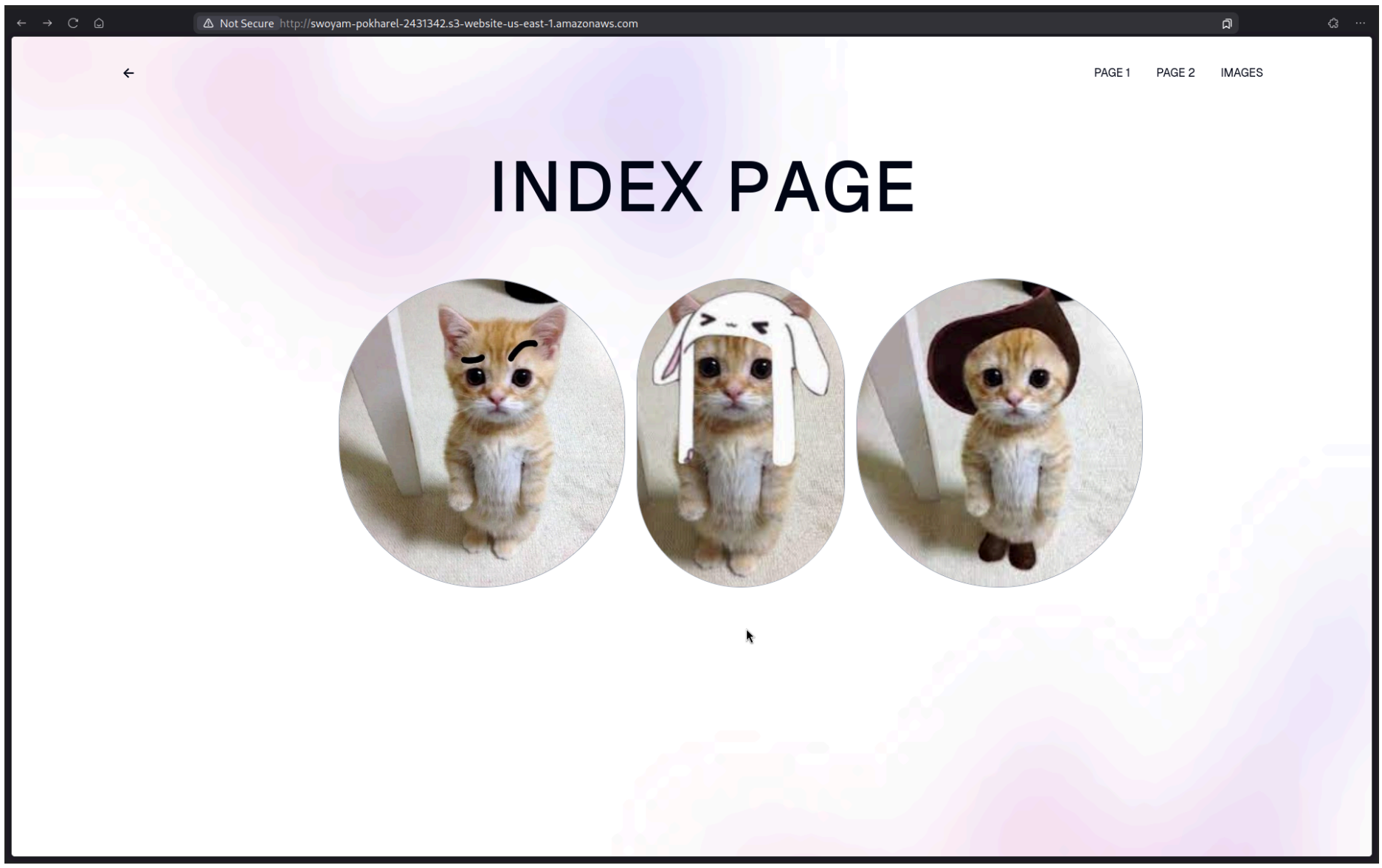

Screenshot Of Uploading Changes




Screenshot Confirming Upload

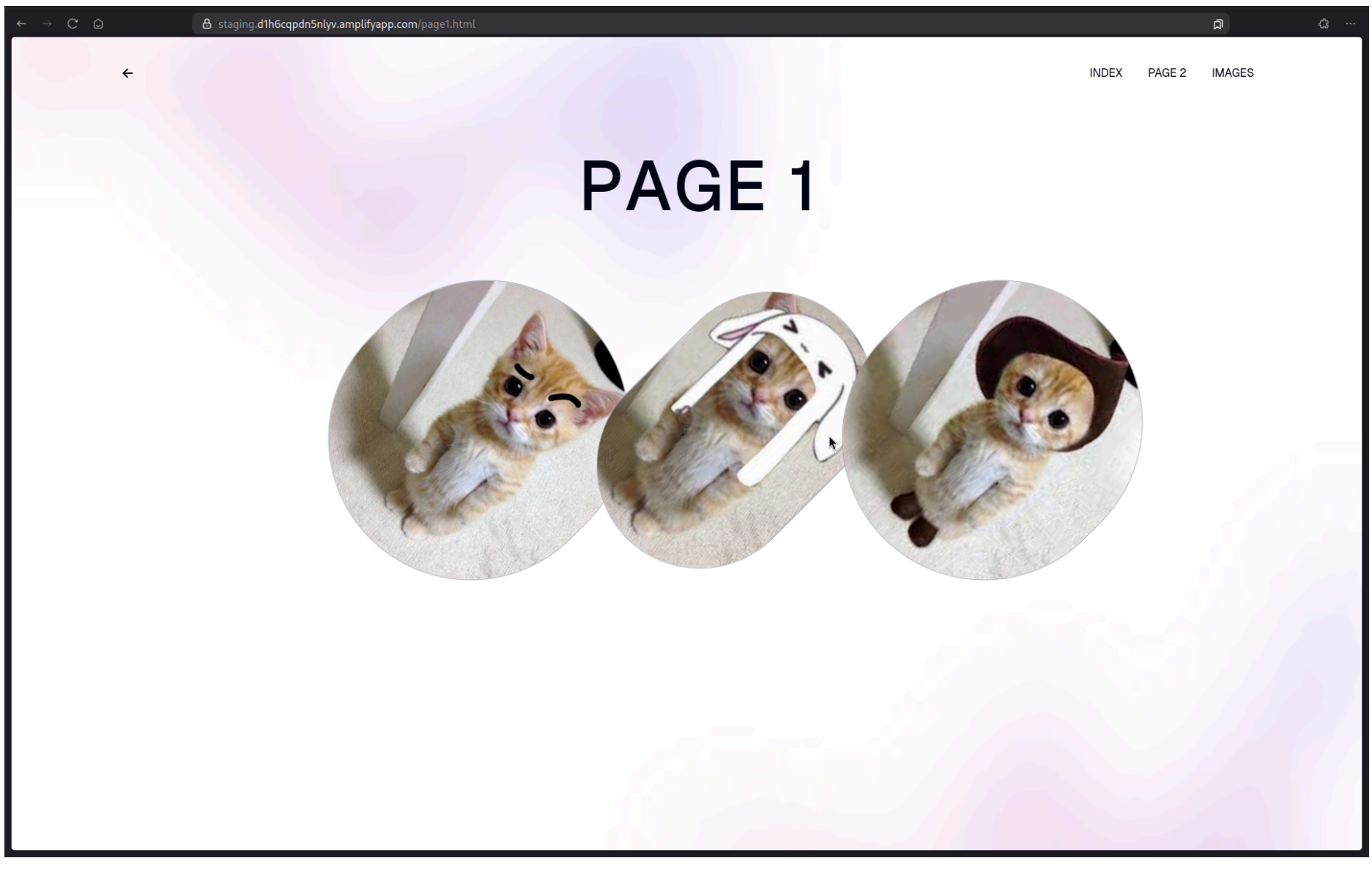
Here, we simply upload the changes we've made and now the navigation happens according to the question.

## V: Screenshots




Screenshot Of Static Page Hosted On S3

From here, clicking on Page 1 navigates us to:




Screenshot Of Amplify Hosted Page 1

From here, clicking on page 2 navigates us to:


Screenshot Of Amplify Hosted Page 2

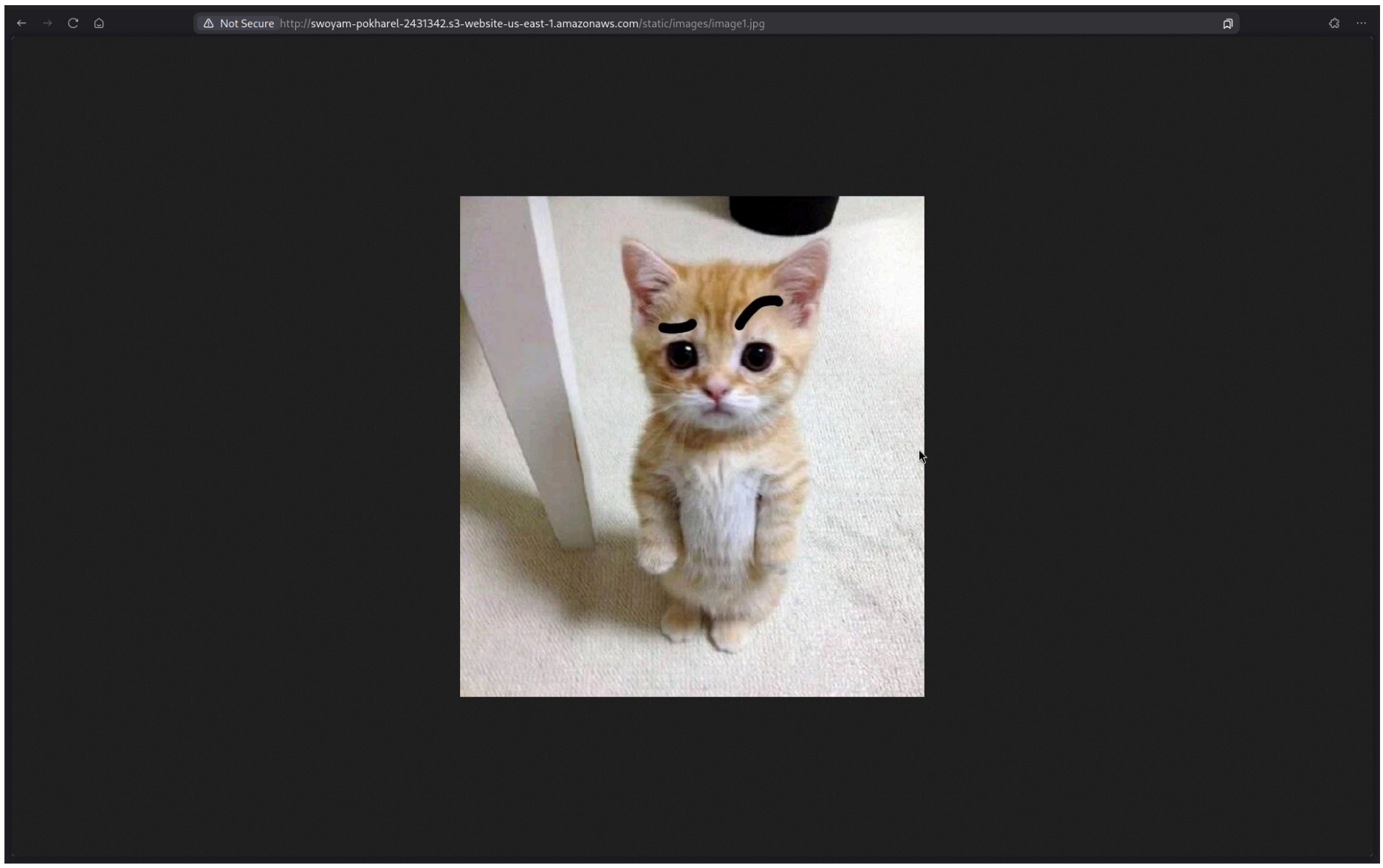
If we navigate back to index from here, we get redirected back to:


Screenshot Of Redirection Back to Index Hosted On S3

UNIVERSITY OF WOLVERHAMPTON

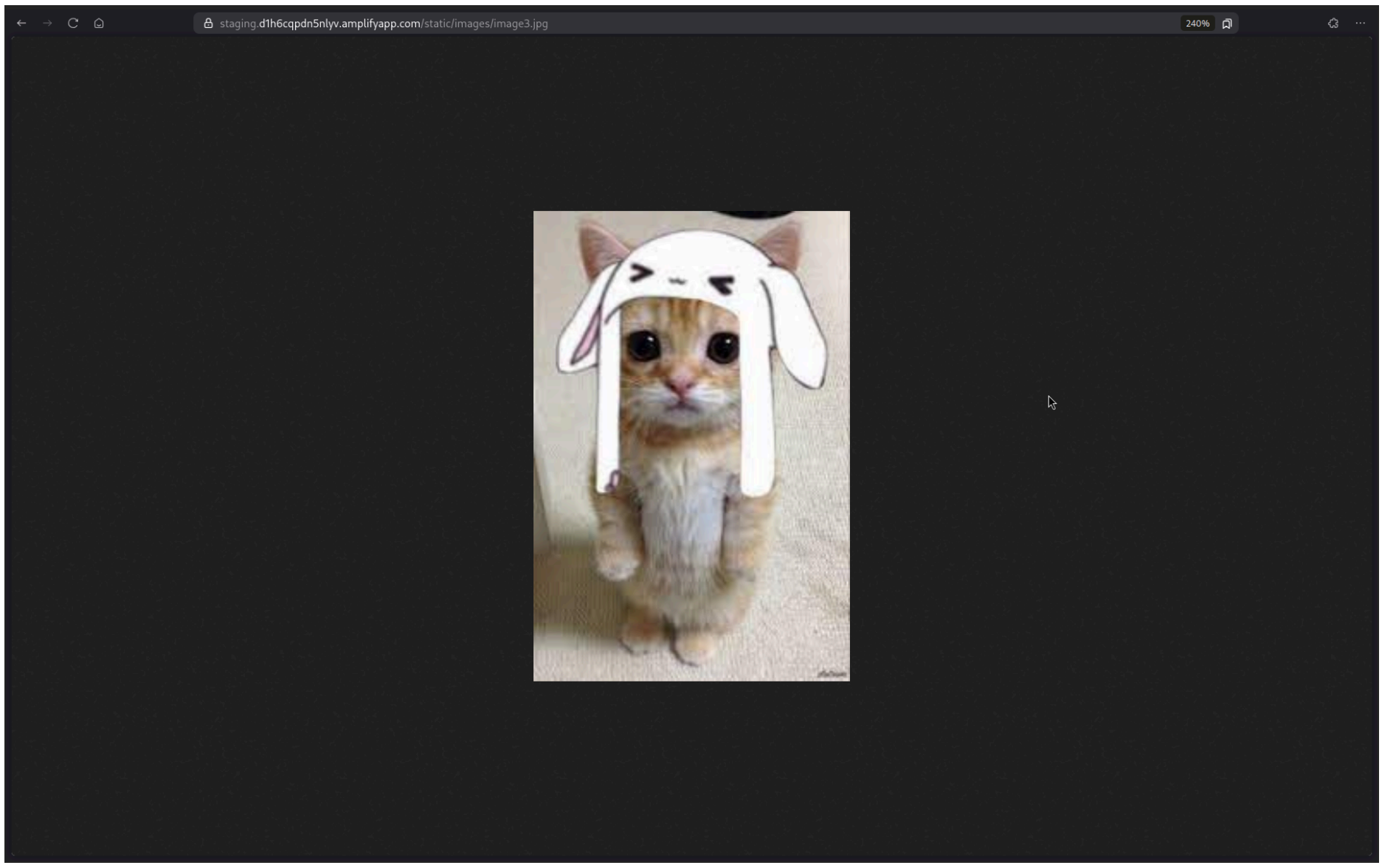
HERALD COLLEGE KATHMANDU

Navigating to any image through the navigation on pages under the S3 domain results in the image being loaded from an S3 URL.




Example Screenshot of the image when accessed via index.html (loaded from the S3 domain).

Navigating to any image through the navigation on pages under the Amplify domain results in the image being loaded from an S3 URL:




Example Screenshot of the image when accessed via page1.html (loaded from the Amplify domain).