# S3 Static Website Hosting – Implementation Steps

## 1. Develop the Website Frontend

I designed and developed the user interface for the Event Announcement System using HTML, CSS, and JavaScript. The frontend includes a dynamic event list, an event submission form, and an email subscription feature. These static website files are structured as follows:

event-announcement-frontend/  
├── index.html  
├── styles.css  
└── events.json

These files are connected to backend AWS services via API Gateway endpoints, enabling full interactivity.

You can view or download the frontend files from the GitHub repository here.

## 2. Create an Amazon S3 Bucket

I created an S3 bucket in the AWS Management Console to host the static website and store event data. The bucket was named uniquely (e.g., event-announcement-system-website) and located in the same AWS region as the backend services (Europe - London, eu-west-2). To support static hosting, I disabled the "Block all public access" setting, allowing public access to website files.

A screenshot of a computer

AI-generated content may be incorrect.

*Figure 1: Creating the S3 bucket in AWS Management Console.*

A screenshot of a computer

AI-generated content may be incorrect.

*Figure 2: Disabling “Block all public access” during bucket creation.*

## 3. Upload Frontend Files to S3

I uploaded the frontend files (index.html, styles.css, and events.json) to the S3 bucket using the AWS Console upload feature. I ensured these files had public-read permissions so visitors could access them on the website.

A screenshot of a computer

AI-generated content may be incorrect.

*Figure 3: Uploading website files to S3 with public-read permissions.*

## 4. Enable Static Website Hosting

Within the S3 bucket properties, I enabled Static Website Hosting and configured it to serve index.html as the default document. This setup allows the bucket URL to act as a fully functional website endpoint.

A screenshot of a computer

AI-generated content may be incorrect.

*Figure 4: Enabling static website hosting and specifying the index document.*

## 5. Configure Bucket Policy for Public Access

To allow public read access to all website files, I applied a bucket policy granting s3:GetObject permission to everyone. This step is crucial for making the website publicly accessible while adhering to security best practices.

A screenshot of a computer

AI-generated content may be incorrect.

*Figure 5: Configuring bucket policy for public access.*

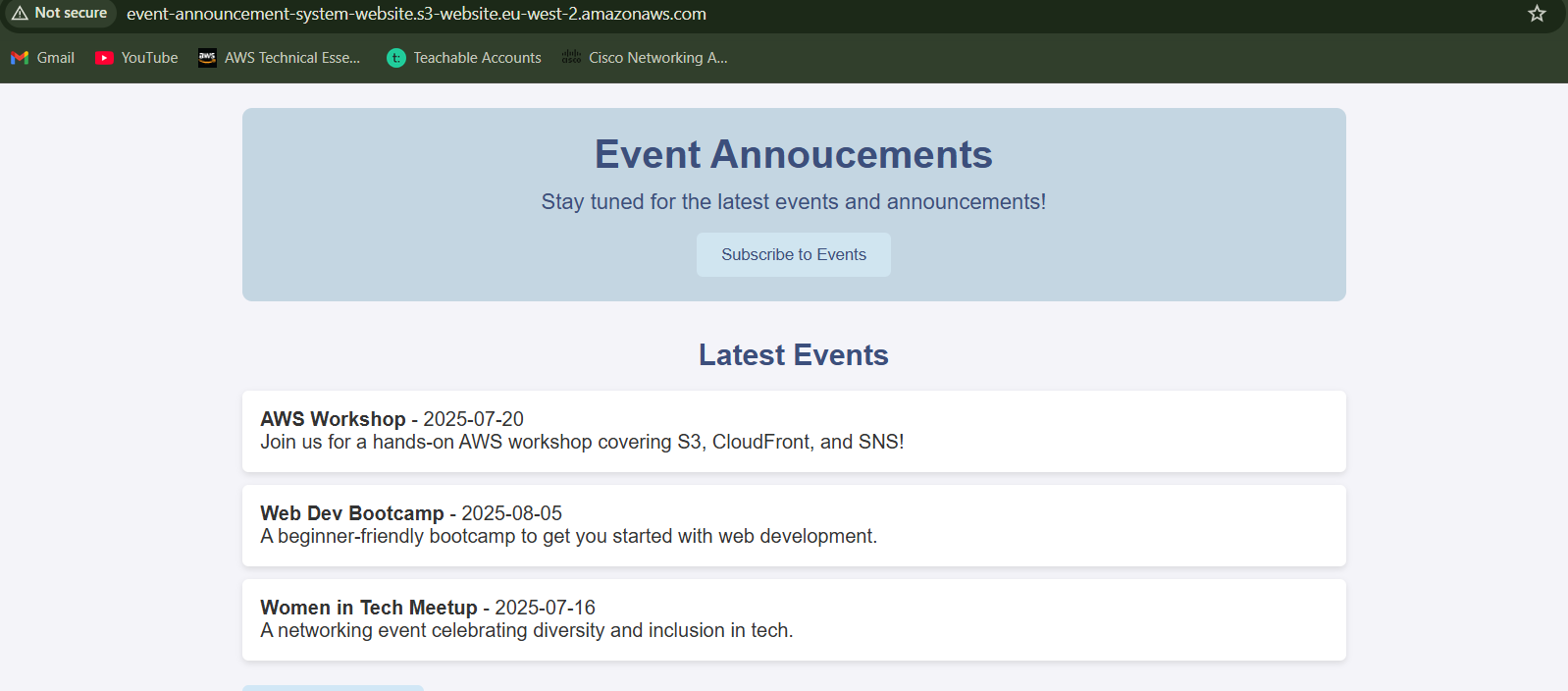
## 6. Verify Website Availability

Finally, I accessed the Static Website Endpoint URL from the S3 bucket settings in a browser to verify that the website loaded correctly and all functionality was intact.

A screenshot of a computer

AI-generated content may be incorrect.

*Figure 6: Viewing the S3 static website endpoint URL in the bucket settings.*



*Figure 7: Confirming the website loads successfully in a web browser.*