# **Frontend Integration and Deployment for Event Announcement System**

As the final phase of my project, I updated the frontend code to integrate the API Gateway endpoints and deployed the updated website to Amazon S3. This ensured smooth interaction between the user interface and backend AWS services.

## **Step 1: Update the Frontend Code**

I integrated the API Gateway endpoint URLs for /subscribe and /create-event directly into the index.html file. This connected the static website frontend with the backend Lambda functions, enabling users to subscribe for notifications and create new events dynamically.



Figure 1: Updated index.html showing API endpoint integration

## **Step 2: Upload the Updated Files to S3**

I navigated to the AWS S3 console and updated the hosted website files to reflect the frontend changes:

* Accessed the S3 bucket hosting the static website.
* Deleted the previous index.html file to avoid conflicts.
* Uploaded the updated index.html along with any other updated assets (e.g., CSS or JSON files).

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2: S3 console showing upload of updated website files

## **Step 3: Test the Website**

After deployment, I performed comprehensive testing to validate the website functionality:

**Accessing the Website:**

Opened the static website URL provided by the S3 bucket to confirm the site loaded correctly.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3: Event Announcement System homepage hosted on Amazon S3 with interactive event features

**Testing Subscription:**

* Clicked the Subscribe to Events button.
* Entered an email address when prompted.
* Confirmed receipt of the subscription confirmation email.
* Completed subscription by confirming via email.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 4: Subscription confirmation popup

A screenshot of a computer

AI-generated content may be incorrect.

Figure 5: Subscription confirmation email

**Testing Event Creation:**

* Clicked the Create New Event button.
* Submitted a new event with title, date, and description.
* Verified success message upon event creation.
* Confirmed that all subscribers received an email notification for the new event.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 6: Event creation form submission

A screenshot of a computer

AI-generated content may be incorrect.

Figure 7: New event creation success message

A close-up of a computer screen

AI-generated content may be incorrect.

Figure 8: Email notification sent to subscribers for a new event

By completing these steps, I successfully connected the frontend and backend components into a seamless, fully serverless Event Announcement System, hosted and served entirely from AWS services.