Project Demo: Process Documentation

Submitted by Sai Pranavi Kurapati
saipranavi.kurapati@sjsu.edu

Overview

This document outlines the frontend process of a system designed to accept user inputs through a web interface, upload files to an AWS S3 bucket, insert data into a DynamoDB table, and trigger further processing including EC2 instance creation and file manipulation.

Github Repository Url:

https://github.com/SaiPranaviKurapati/Fovus_Coding_Challenge_Submission

System Components

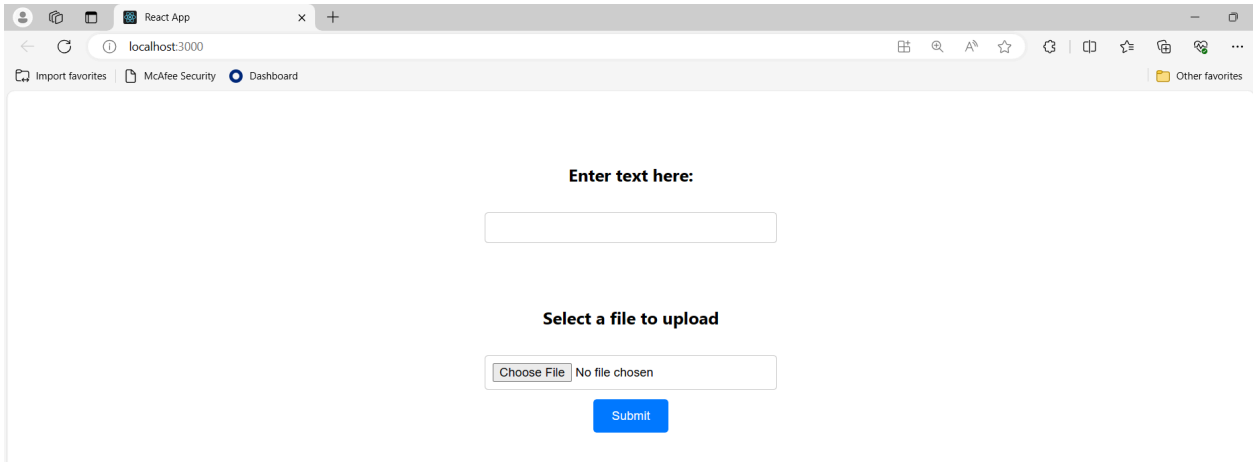
- Frontend Interface: Built with React.js
- Backend Services: AWS Lambda, DynamoDB, S3 Bucket, EC2 Instance, IAM Roles
- API Gateway: fileUploaderGateway, DynamoDBInsertionAPI

Step-by-Step Process

1. User Interface Interaction

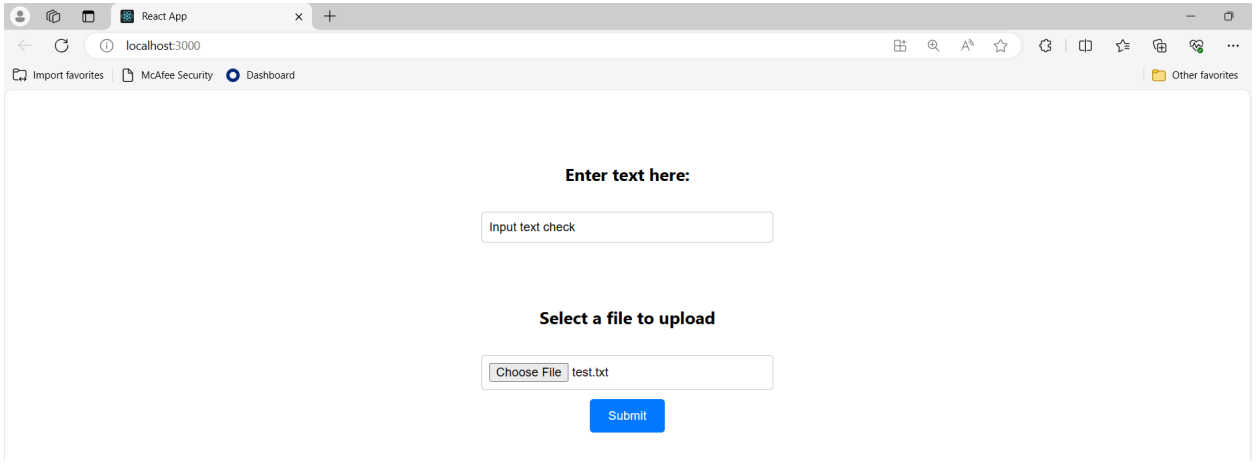
The frontend React UI presents the user with:

- An input text field.
- An input file uploader.
- A submit button.



2. File and Text Submission

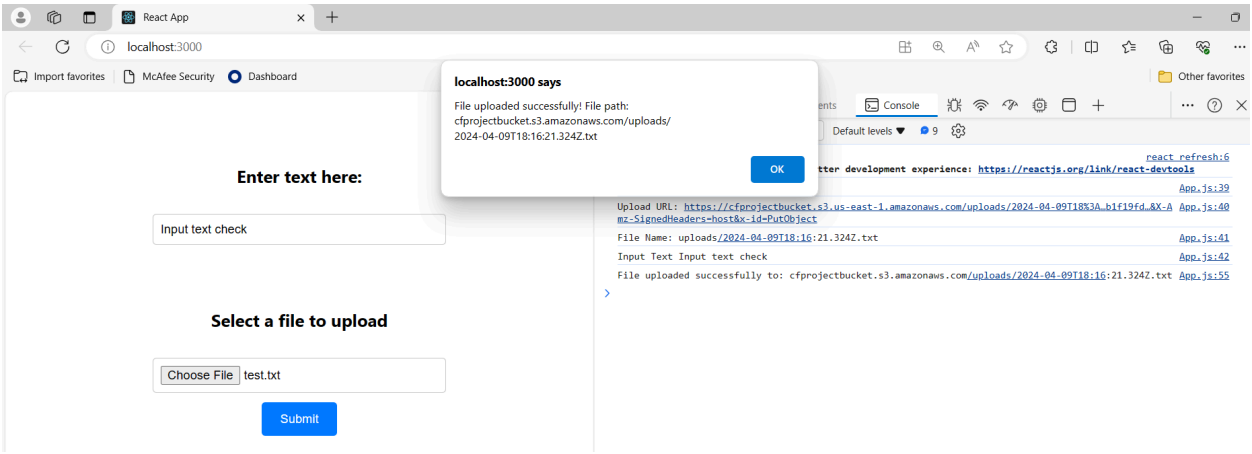
Users attach a file, enter text into the provided field, and click the submit button to initiate the process.



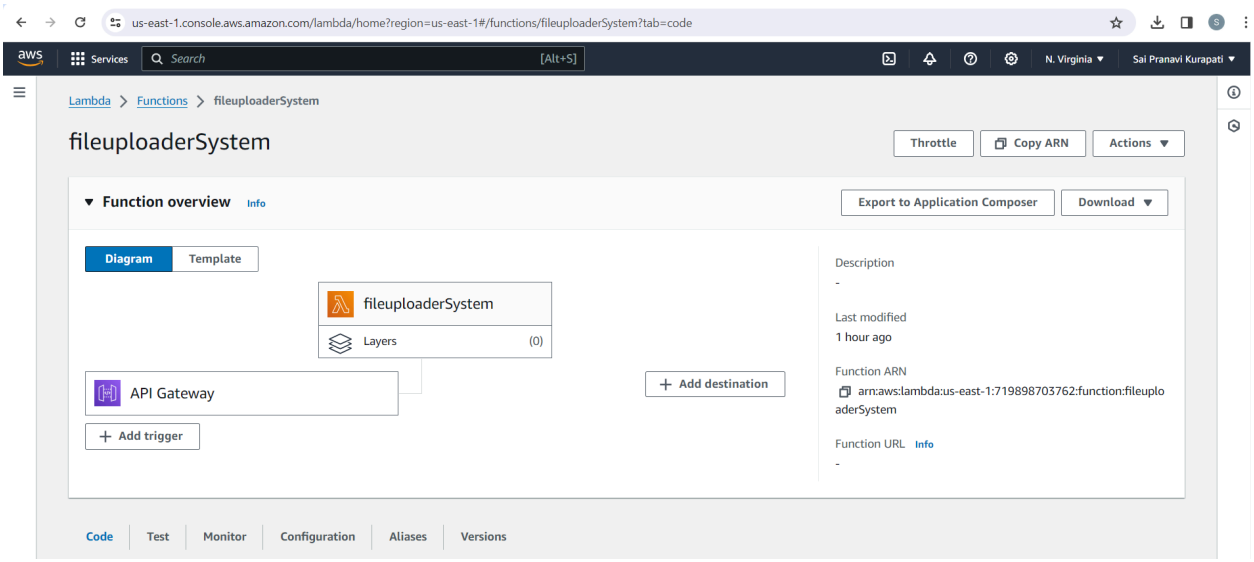
3. Pre-signed URL Generation

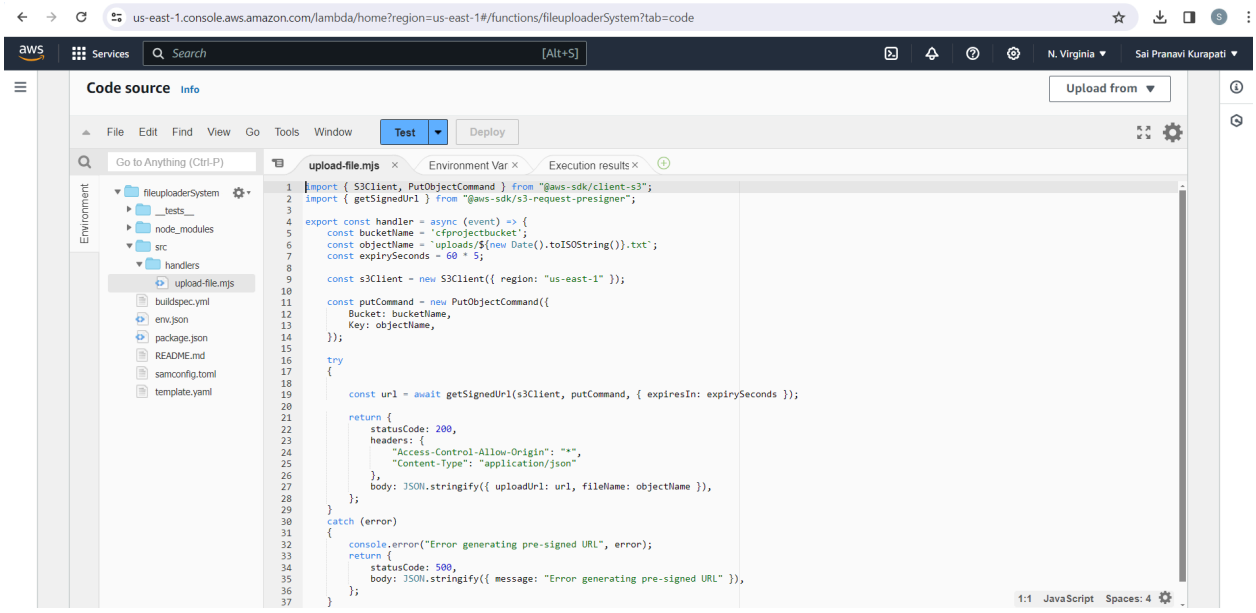
Upon submission, the frontend calls a Lambda function named fileuploaderSystem, which retrieves a pre-signed URL from the fileUploaderGateway API Gateway.

- Obtained a pre-signed URL from the file uploader system's Lambda function, which was then logged in the application's console.



- Developed a Lambda function named fileuploaderSystem, integrated with API Gateway, which generates pre-signed URLs. The user interface initiates a GET request to obtain the pre-signed URL from this Lambda function.

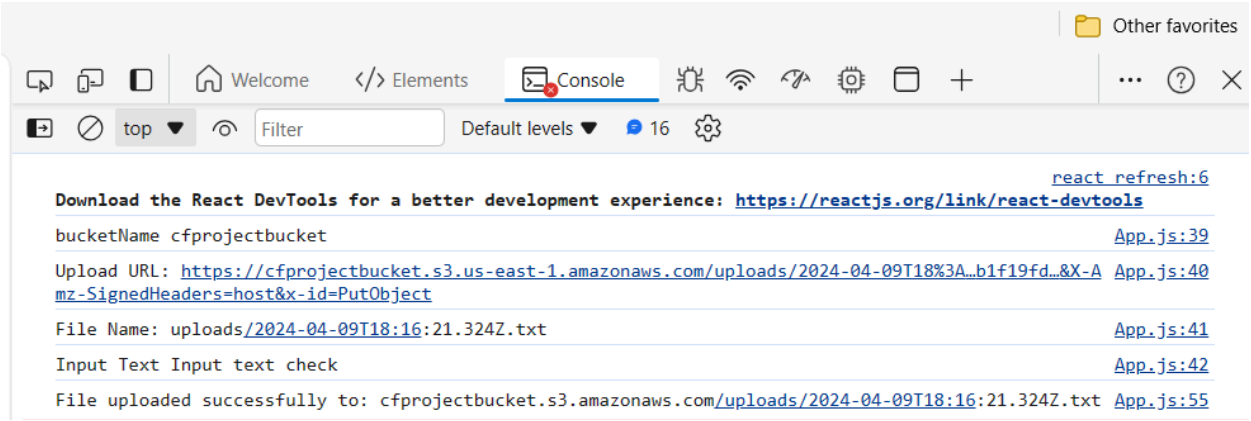




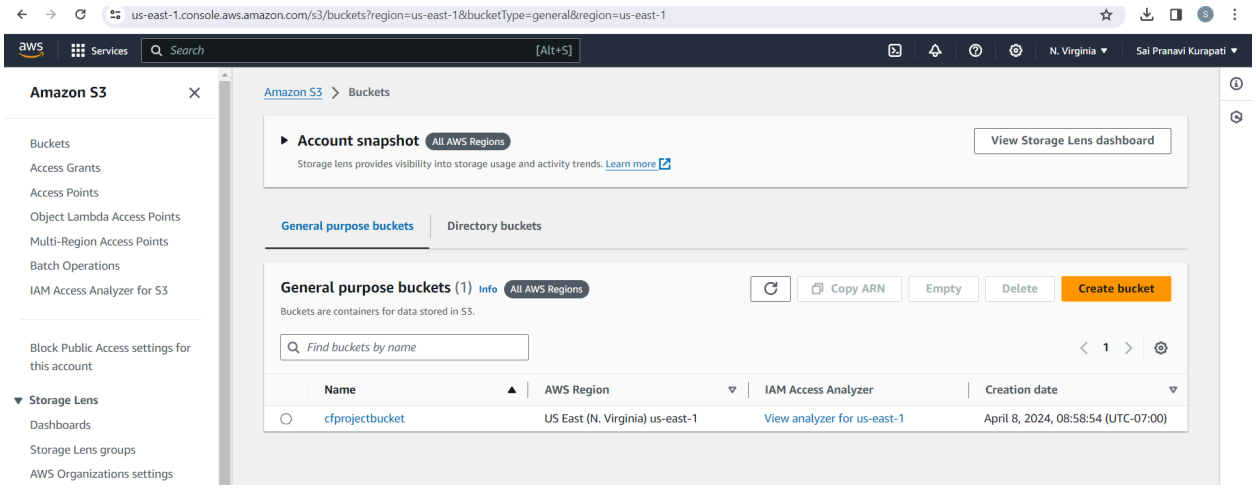
4. File Upload to S3 Bucket

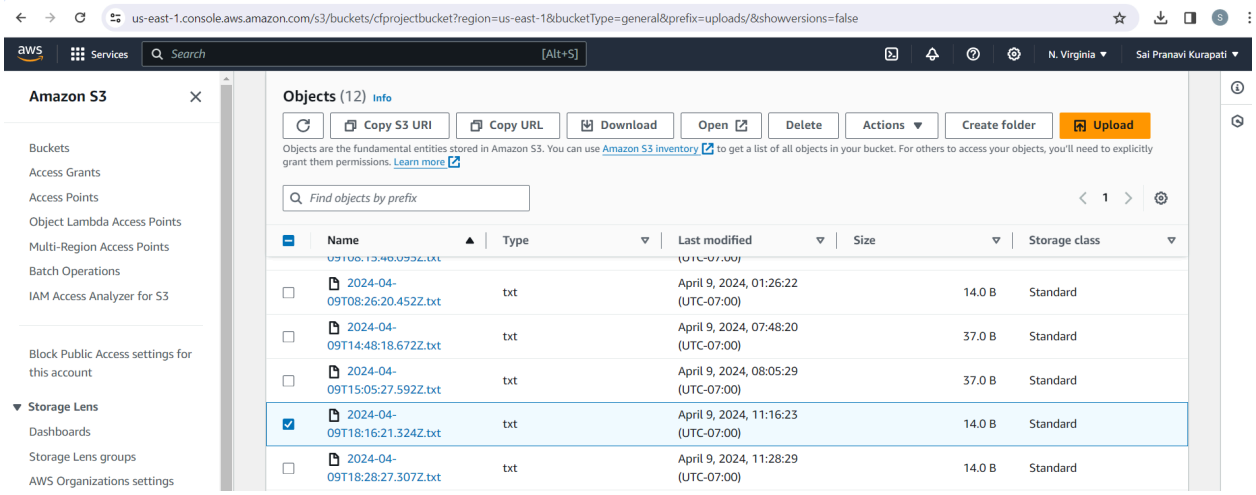
With the pre-signed URL, the React application uploads the input file directly to the specified AWS S3 bucket.

- The console displays confirmation of successful file upload from the browser to an S3 bucket in AWS utilizing the pre-signed URL.



- The file uploaded from the browser has been successfully stored in the specified S3 bucket, as shown below.





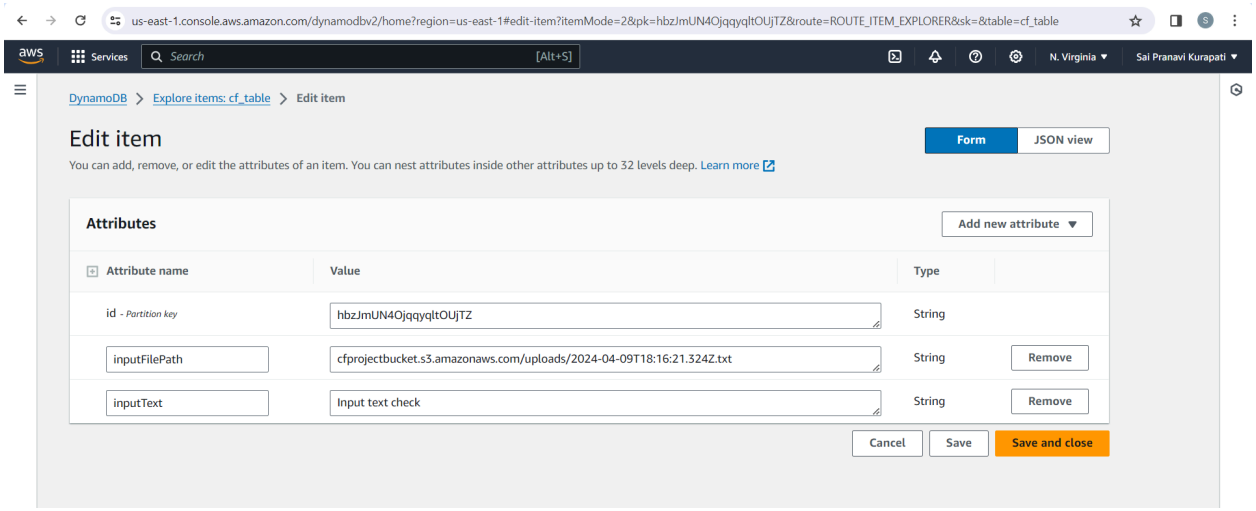
5. Data Insertion into DynamoDB

Next, the React app makes a POST API call to the DynamoDBInsertionAPI (via API Gateway) to insert the submission details into a DynamoDB table named cf_table. The handling Lambda function is insertToDynamoDBTable.

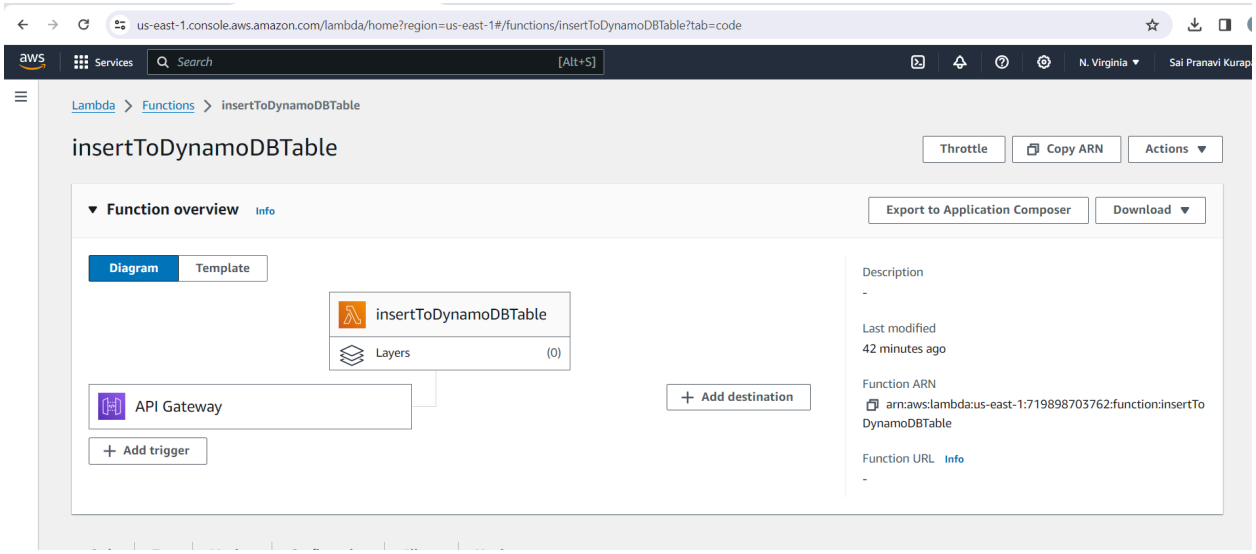
This insertion includes:

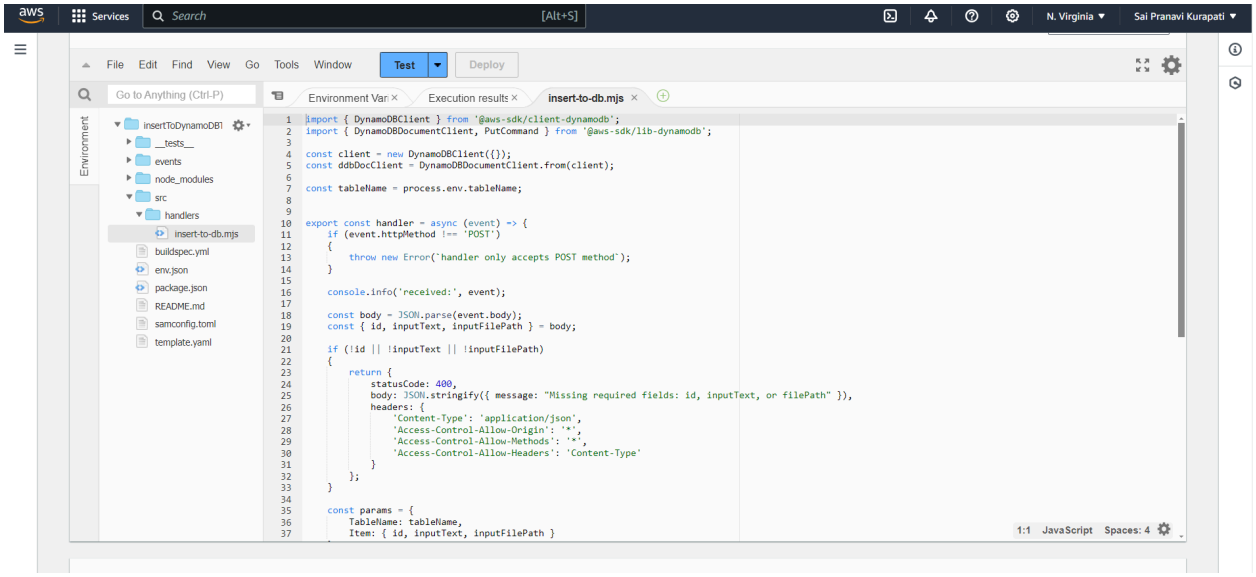
- *A unique ID (generated using nano ID).*
- *The input text.*
- *The input file path (constructed as bucketname/filename).*

The provided ID, inputText, and inputFilePath have been successfully inserted into the DynamoDB table.



- The insertToDynamoDBTable lambda function is mapped to the POST API call of the gateway.

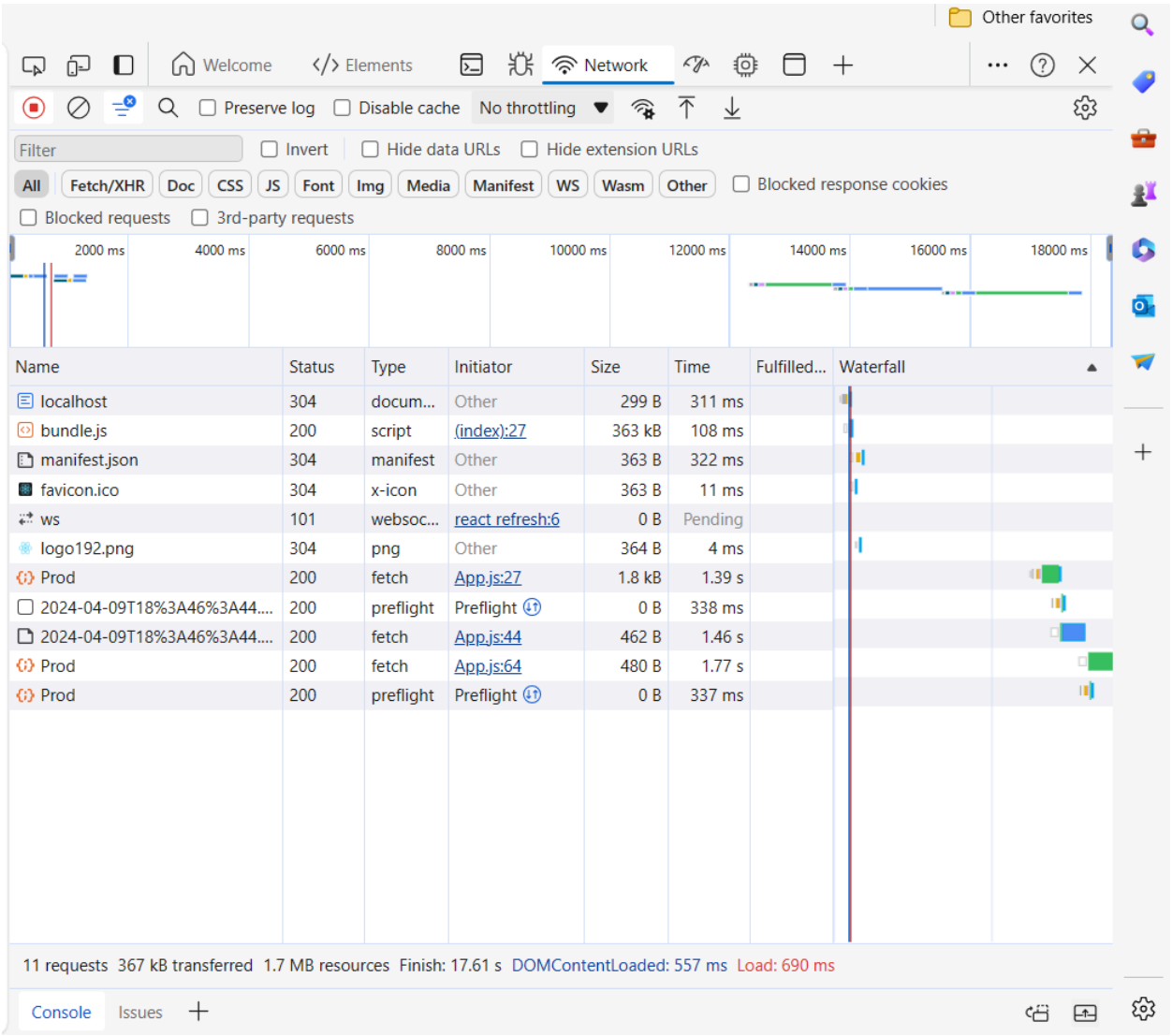




- After successfully inserting the fields into the table, the network tab of the browser

```
Data sent to Lambda successfully: UwHHuJ2nN6iJvyaDSvrv_ Test  cfprojectbucket.s3.amazonaws.com/uploa App.js:78
ds/2024-04-09T18:40:31.448Z.txt
```

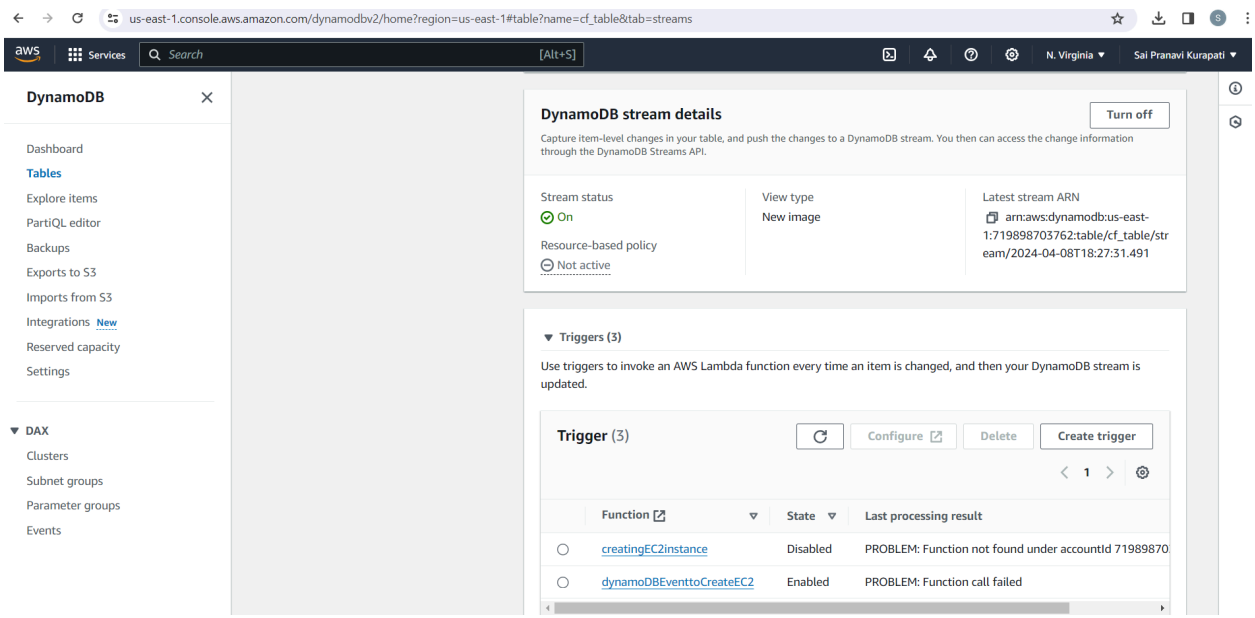
›



6. DynamoDB Event Trigger

The insertion into cf_table triggers an event, which in turn calls the Lambda function dynamoDBEventtoCreateEC2.

- A trigger is created in the DynamoDB table to activate after a new field is created in the table

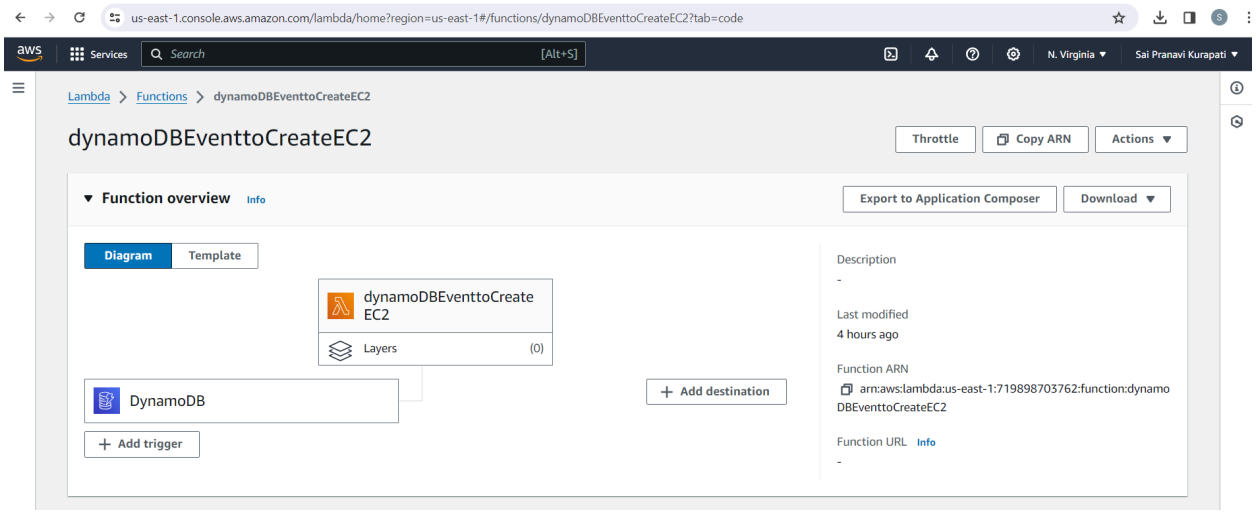


7. EC2 Instance Creation and Processing

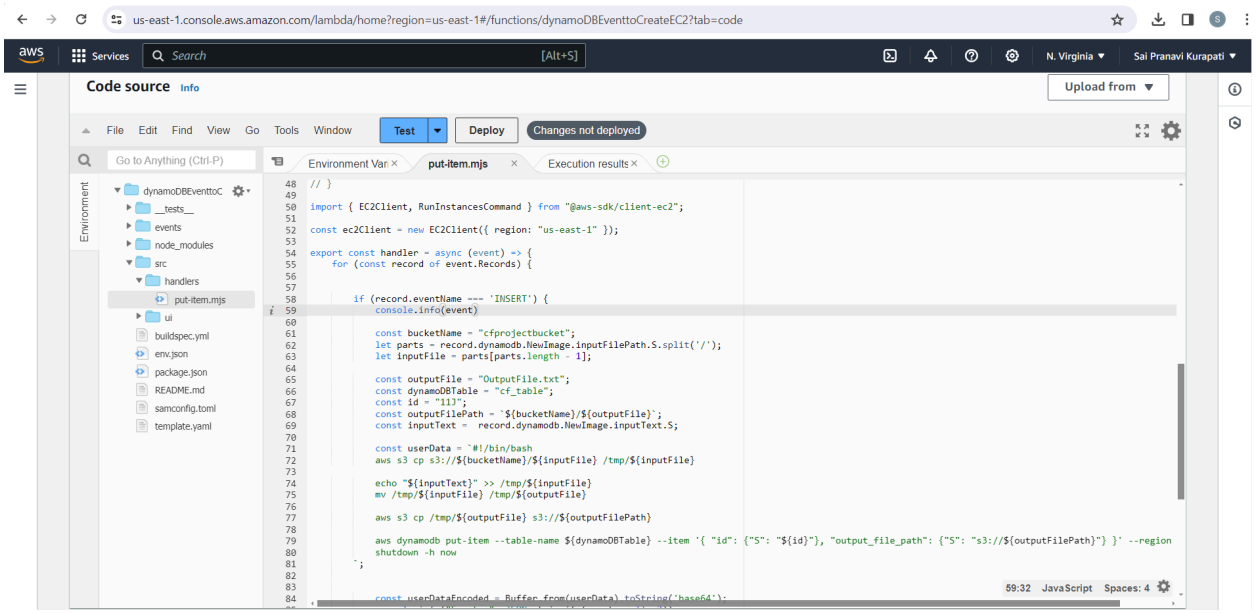
The dynamoDBEventtoCreateEC2 function:

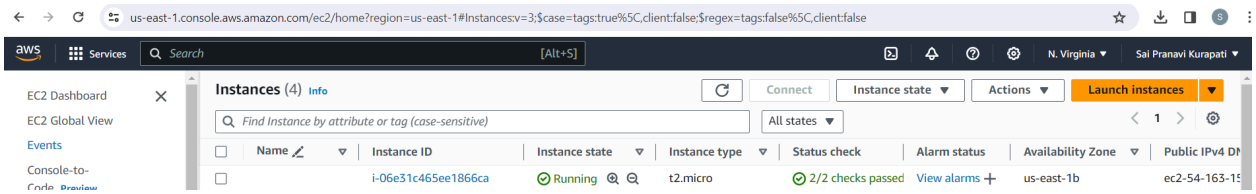
- Creates an EC2 instance.
- Executes a script to:
    - Retrieve the input file path and text using the event's ID.
    - Download the file from the S3 bucket.
    - Merge the input text with the file.
    - Upload the modified file back to the S3 bucket.
- Terminates the instance upon completion of the script.

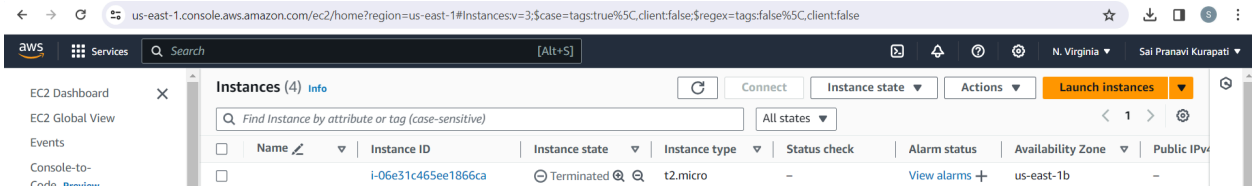
- The trigger created in the DynamoDB table is mapped to the lambda function, dynamoDBEventtoCreateEC2.

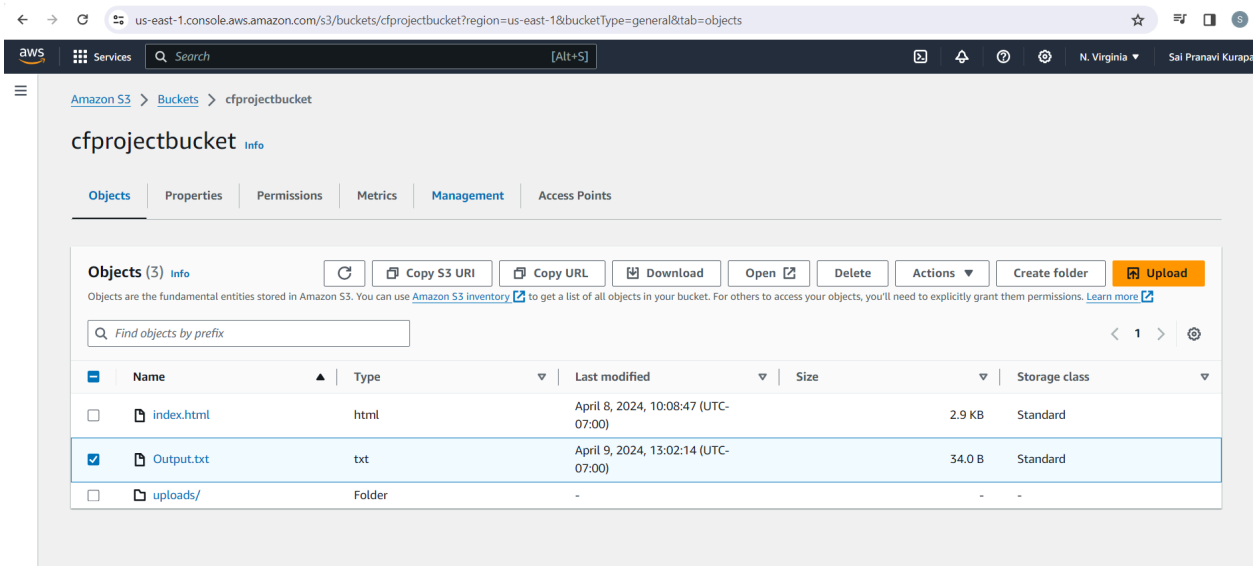


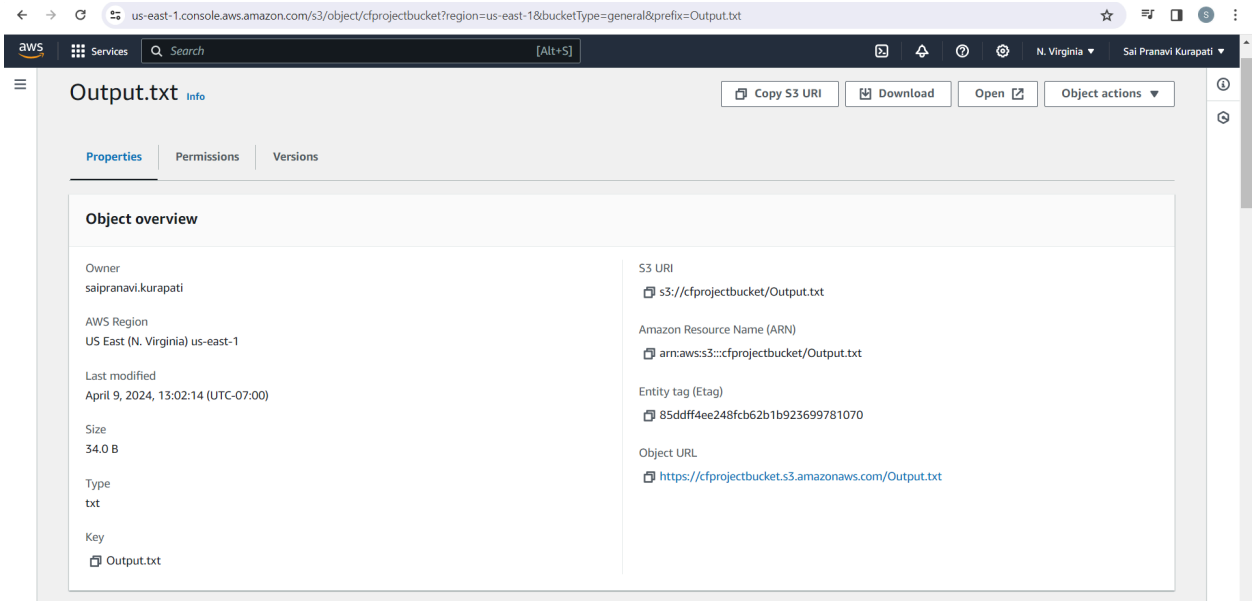
- A script is executed within the Lambda to create an EC2 instance. This EC2 instance retrieves the values of the row in the DynamoDB table using the ID. The input file is downloaded from the S3 bucket, merged with the input text, and the contents are copied into output.text. Subsequently, the file is uploaded to the S3 bucket. Finally, the EC2 instance is terminated.

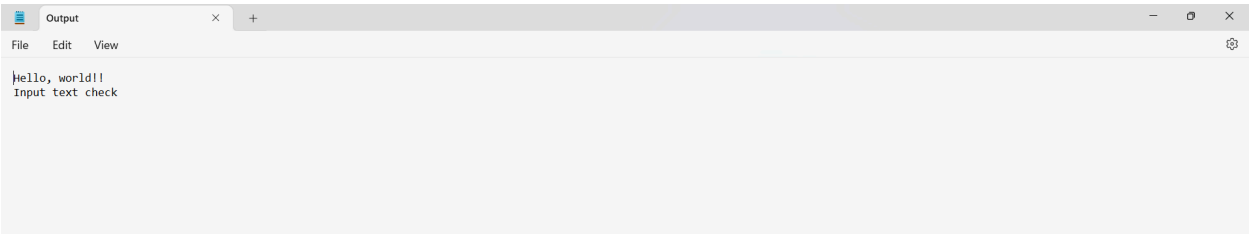



us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Instances:v=3;$case=tags:true%5C,client:false;$regex=tags:false%5C,client:false
Services
Search
[Alt+S]
N. Virginia
Sai Pranavi Kurapati
EC2 Dashboard
EC2 Global View
Events
Console-to-
Instances (4) Info
Connect
Instance state
Actions
Launch instances
Find Instance by attribute or tag (case-sensitive)
All states
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
i-06e31c465ee1866ca
Running
t2.micro
2/2 checks passed
View alarms
us-east-1b


us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Instances:v=3;$case=tags:true%5C,client:false;$regex=tags:false%5C,client:false
Services
Search
[Alt+S]
N. Virginia
Sai Pranavi Kurapati
EC2 Dashboard
EC2 Global View
Events
Console-to-
Instances (4) Info
Connect
Instance state
Actions
Launch instances
Find Instance by attribute or tag (case-sensitive)
All states
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
i-06e31c465ee1866ca
Terminated
t2.micro
View alarms
us-east-1b


us-east-1.console.aws.amazon.com/s3/buckets/cfprojectbucket?region=us-east-1&bucketType=general&tab=objects
Services
Search
[Alt+S]
N. Virginia
Amazon S3 > Buckets > cfprojectbucket
cfprojectbucket Info
Objects
Properties
Permissions
Metrics
Management
Access Points
Objects (3) Info
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Find objects by prefix
Name
Type
Last modified
Size
Storage class
index.html
html
April 8, 2024, 10:08:47 (UTC-07:00)
2.9 KB
Standard
Output.txt
txt
April 9, 2024, 13:02:14 (UTC-07:00)
34.0 B
Standard
uploads/
Folder
-


us-east-1.console.aws.amazon.com/s3/object/cfprojectbucket?region=us-east-1&bucketType=general&prefix=Output.txt
Services
Search
[Alt+S]
N. Virginia
Sai Pranavi Kurapati
Output.txt Info
Copy S3 URI
Download
Open
Object actions
Properties
Permissions
Versions
Object overview
Owner
saipranavi.kurapati
AWS Region
US East (N. Virginia) us-east-1
Last modified
April 9, 2024, 13:02:14 (UTC-07:00)
Size
34.0 B
Type
txt
Key
Output.txt
S3 URI
s3://cfprojectbucket/Output.txt
Amazon Resource Name (ARN)
arn:aws:s3:::cfprojectbucket/Output.txt
Entity tag (Etag)
85ddff4ee248fcb62b1b923699781070
Object URL
https://cfprojectbucket.s3.amazonaws.com/Output.txt



Conclusion

The frontend procedure for processing files, submitting user input, and initiating backend activities within the project has been described in depth in this document. This method shows how to combine a React frontend with a variety of AWS services to create a seamless file processing and data management system.