CSC-PROJECT

BUILDING A SERVERLESS VIDEO STREAMING SERVICE

-

2100030048

Abstract:

Building a serverless video streaming service using Amazon S3, React, and CloudFront. By ditching traditional server setups, we cut costs and complexity while boosting scalability. We'll walk through the architecture, highlighting user authentication, video transcoding, and CloudFront integration. With a focus on simplicity and efficiency, this solution delivers a seamless video streaming experience, perfect for today's demands.

Introduction

Welcome, everyone. In today's digital world, video streaming has become a staple of our online experiences. But creating a video streaming service can be complex and costly, right? Not anymore. With Amazon Web Services (AWS), we have access to simple and affordable solutions for building video streaming platforms.

Today, we'll focus on how AWS enables us to create serverless video streaming services. By using services like Amazon S3 for storing videos, React for building interfaces, and CloudFront for speedy delivery, we can streamline the process and keep costs down. So, let's dive in and explore how AWS can revolutionize the way we deliver video content online.

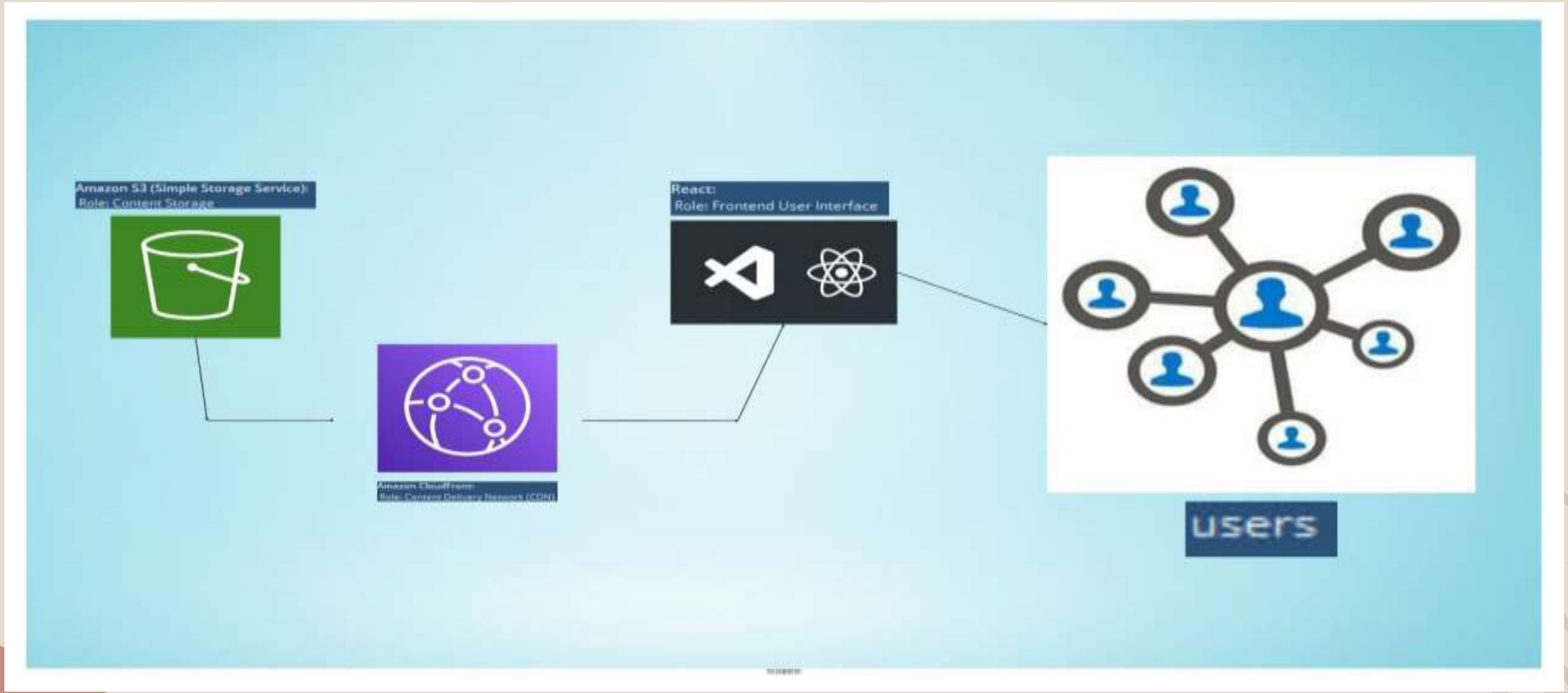
Services Used

- Amazon S3(Simple Storage Service)
- Amazon CloudFront

Other Tools Used:

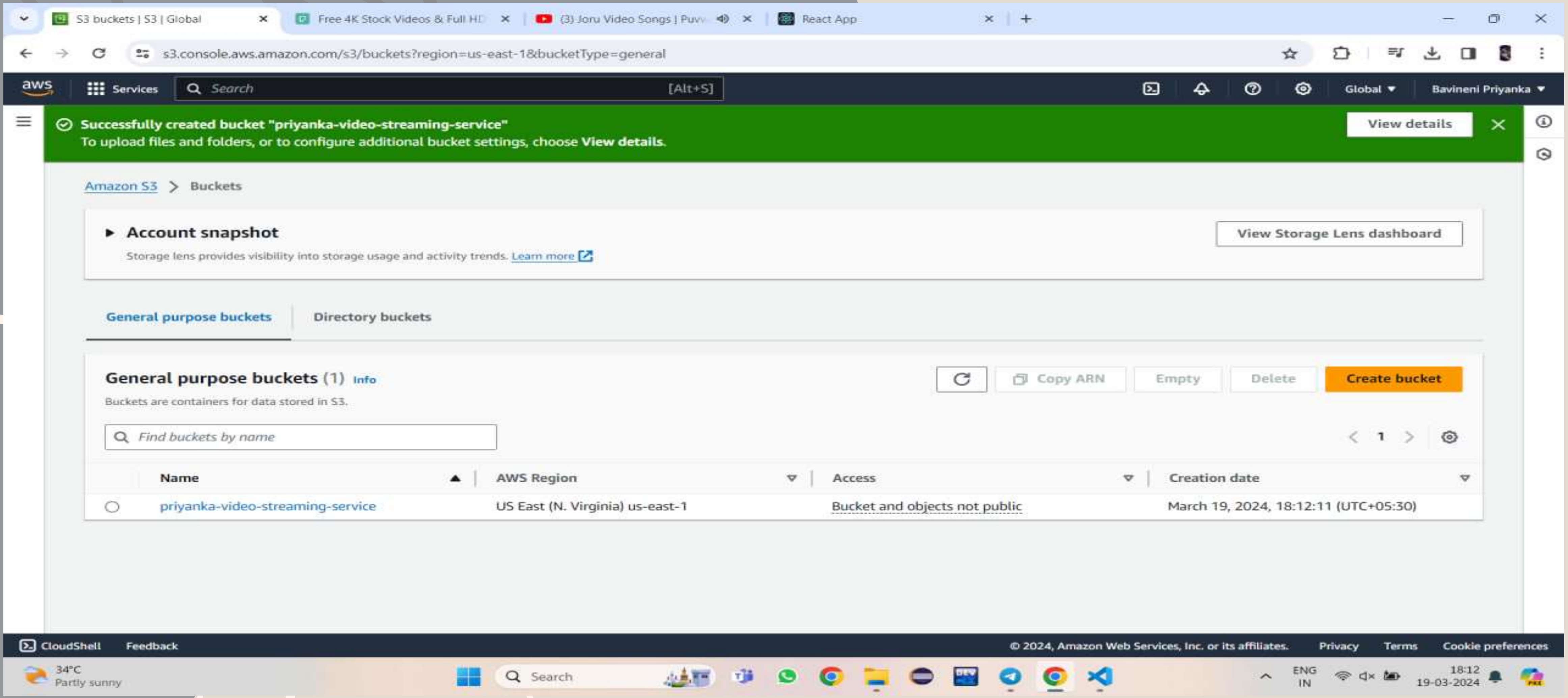
- React
- VS Code

Architecture


Amazon S3 (Simple Storage Service):
Role: Content Storage
Amazon CloudFront:
Role: Content Delivery Network (CDN)
React:
Role: Frontend User Interface
users

Steps:

step1: Signin to Amazon console
Step2: Go to services and search for s3 bucket, and create a bucket


s3.console.aws.amazon.com/s3/buckets?region=us-east-1&bucketType=general

Successfully created bucket "priyanka-video-streaming-service"
To upload files and folders, or to configure additional bucket settings, choose **View details**.

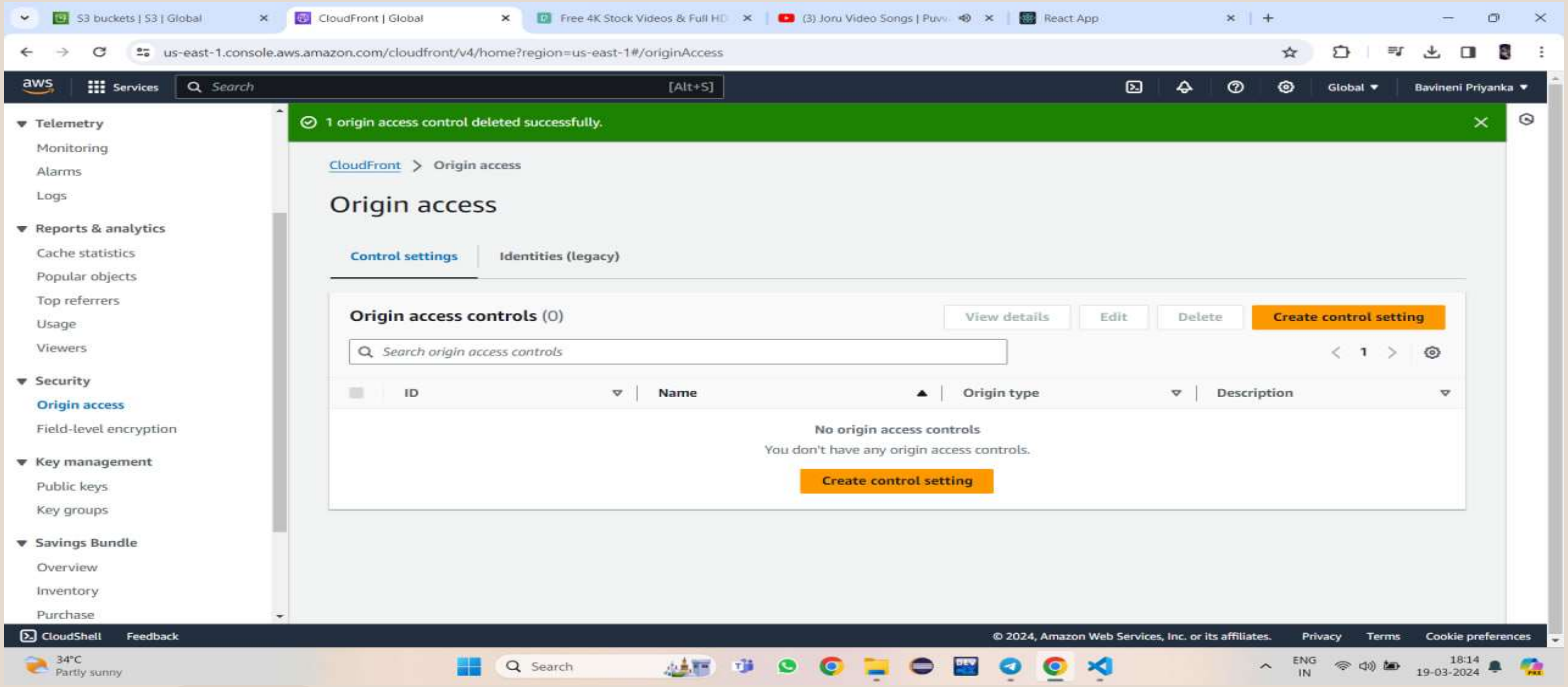
Amazon S3 > Buckets

Account snapshot
Storage lens provides visibility into storage usage and activity trends. Learn more

General purpose buckets (1) Info
Buckets are containers for data stored in S3.

Name	AWS Region	Access	Creation date
priyanka-video-streaming-service	US East (N. Virginia) us-east-1	Bucket and objects not public	March 19, 2024, 18:12:11 (UTC+05:30)

Step3: After that go to cloud front service in that go to origin access and create a control setting



us-east-1.console.aws.amazon.com/cloudfront/v4/home?region=us-east-1#/originAccess

1 origin access control deleted successfully.

CloudFront > Origin access

Origin access

Control settings | Identities (legacy)

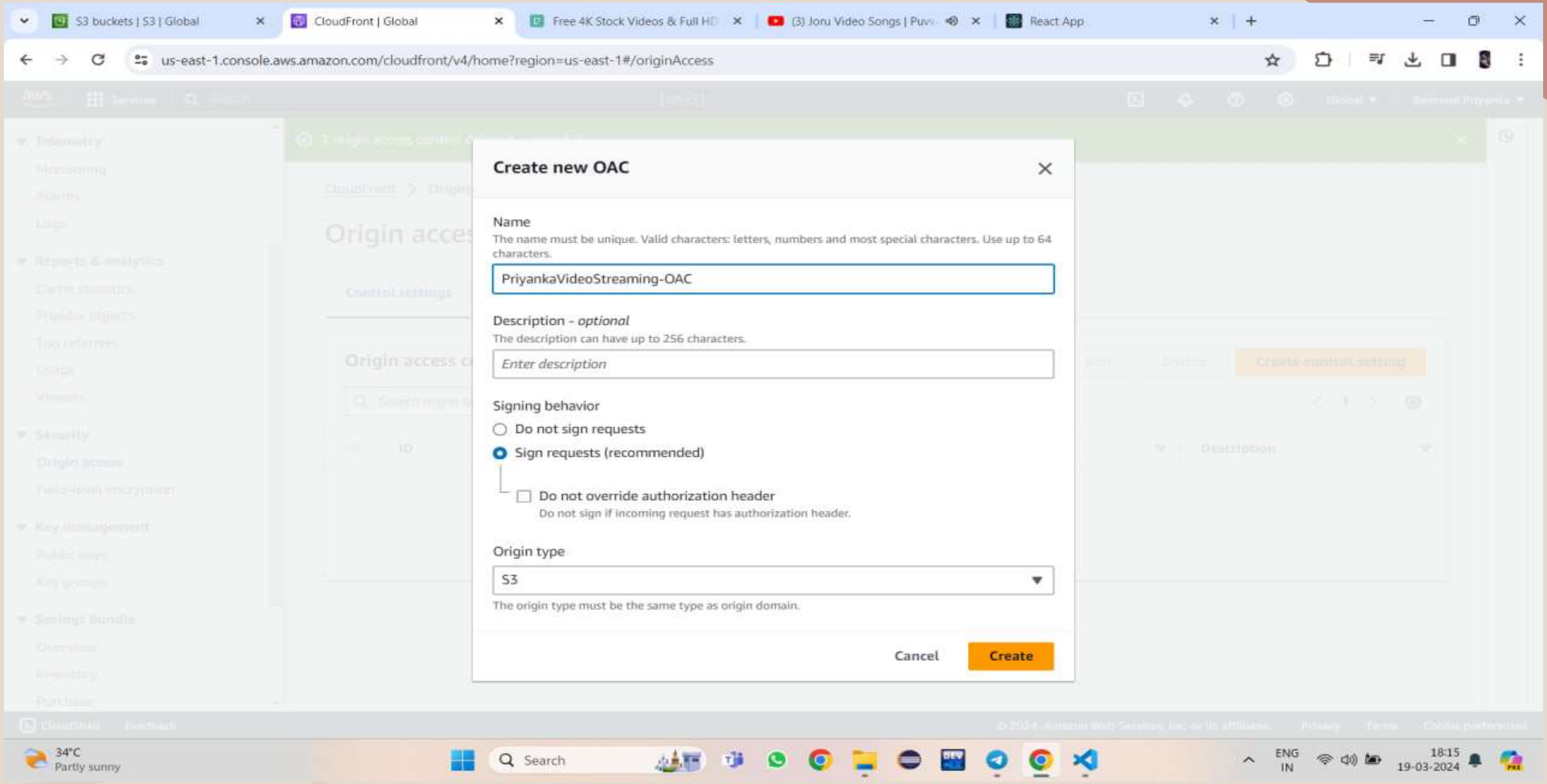
Origin access controls (0)

ID	Name	Origin type	Description

No origin access controls

You don't have any origin access controls.

Create control setting


S3 buckets | S3 | Global
CloudFront | Global
Free 4K Stock Videos & Full HD
(3) Joru Video Songs | Puv
React App
us-east-1.console.aws.amazon.com/cloudfront/v4/home?region=us-east-1#/originAccess

Create new OAC

Name

The name must be unique. Valid characters: letters, numbers and most special characters. Use up to 64 characters.

PriyankaVideoStreaming-OAC

Description - *optional*

The description can have up to 256 characters.

Enter description

Signing behavior

- Do not sign requests
- [✓ Sign requests (recommended)]
  - ☐ Do not override authorization header
    Do not sign if incoming request has authorization header.

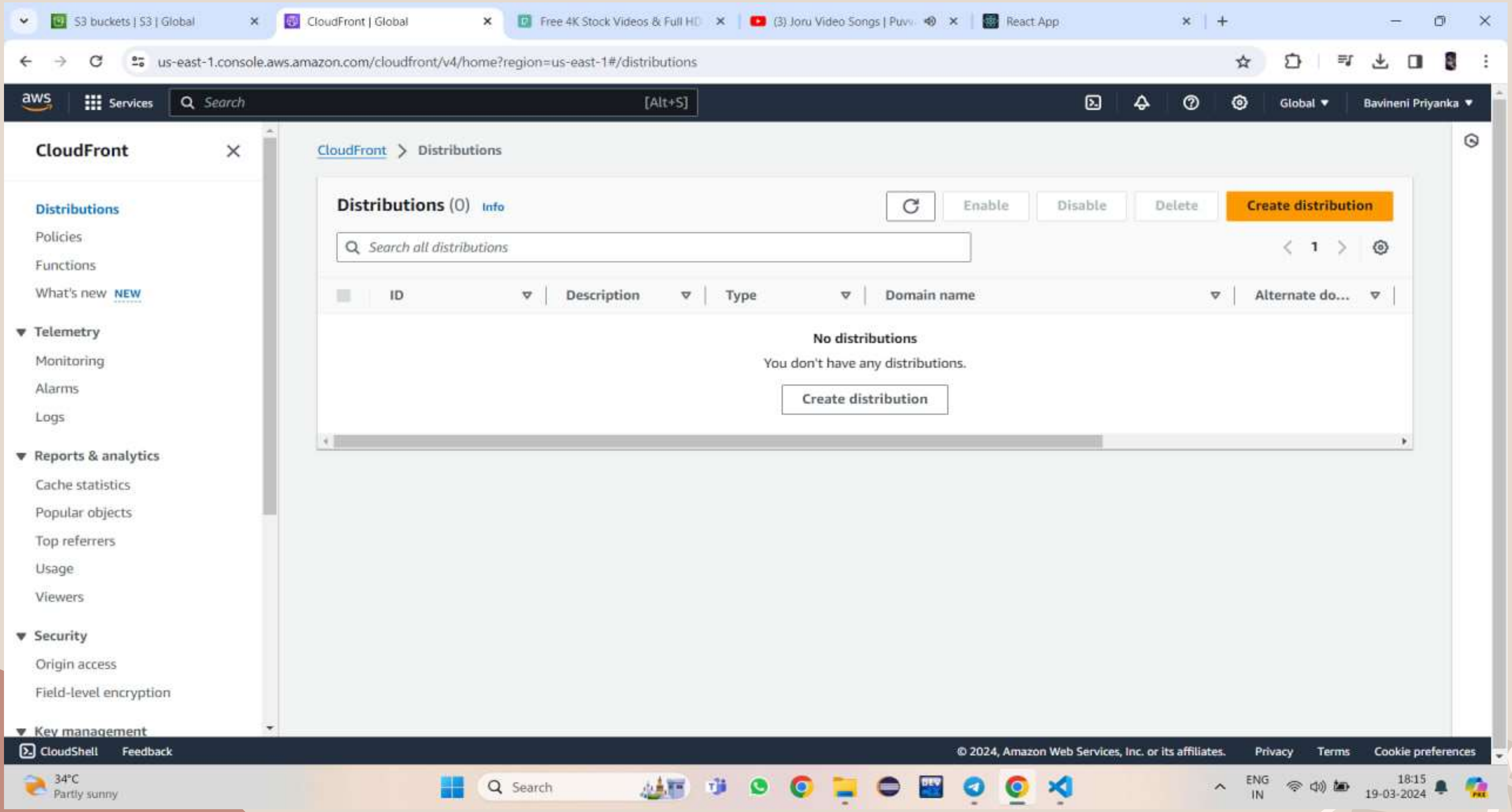
Origin type

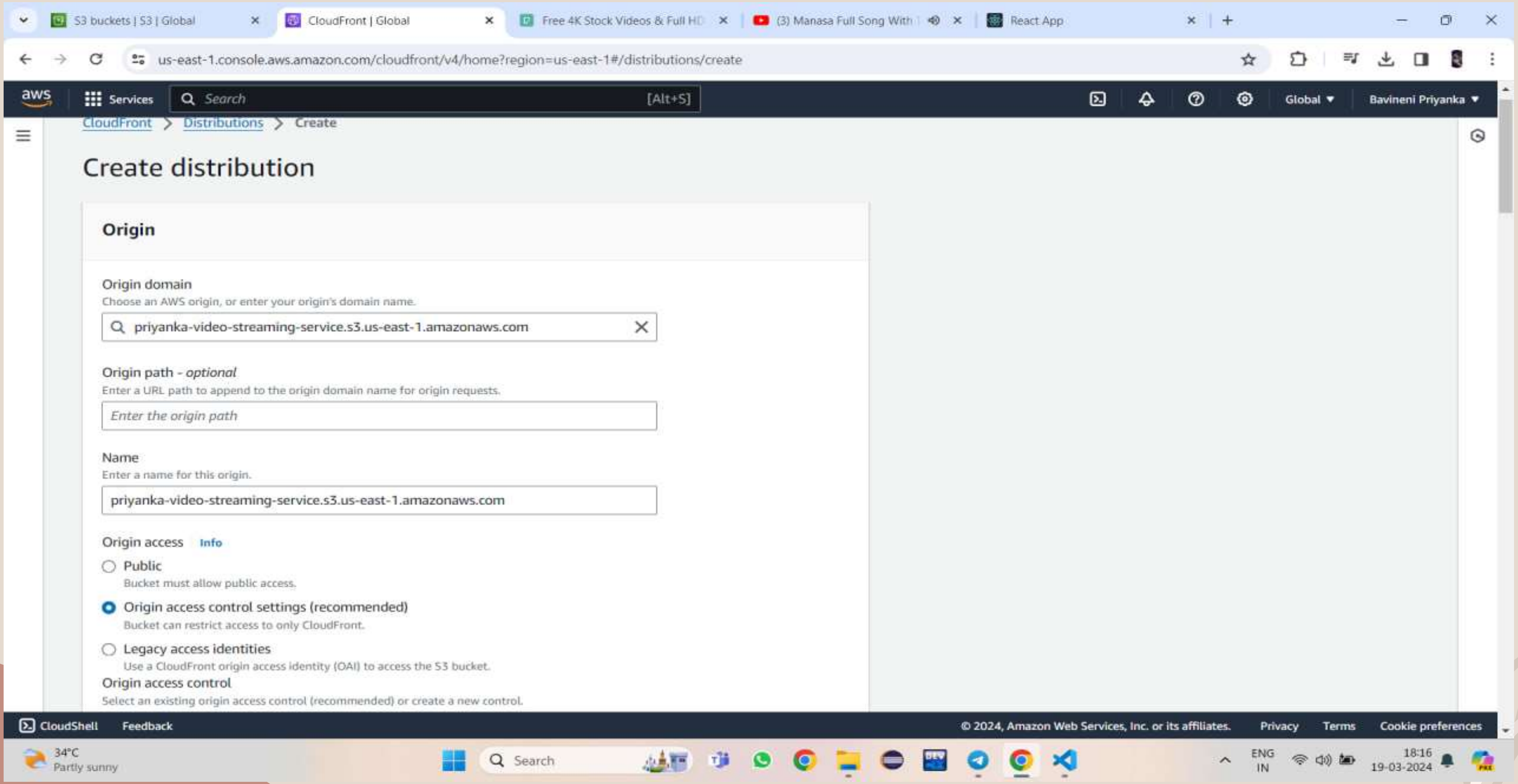
S3

The origin type must be the same type as origin domain.

Cancel | Create

Step4: After creating control setting we have to create a distribution




S3 buckets | S3 | Global
CloudFront | Global
Free 4K Stock Videos & Full HD
(3) Manasa Full Song With
React App
us-east-1.console.aws.amazon.com/cloudfront/v4/home?region=us-east-1#/distributions/create
Services
Search
[Alt+S]
Global
Bavineni Priyanka
CloudFront > Distributions > Create

Create distribution

Origin

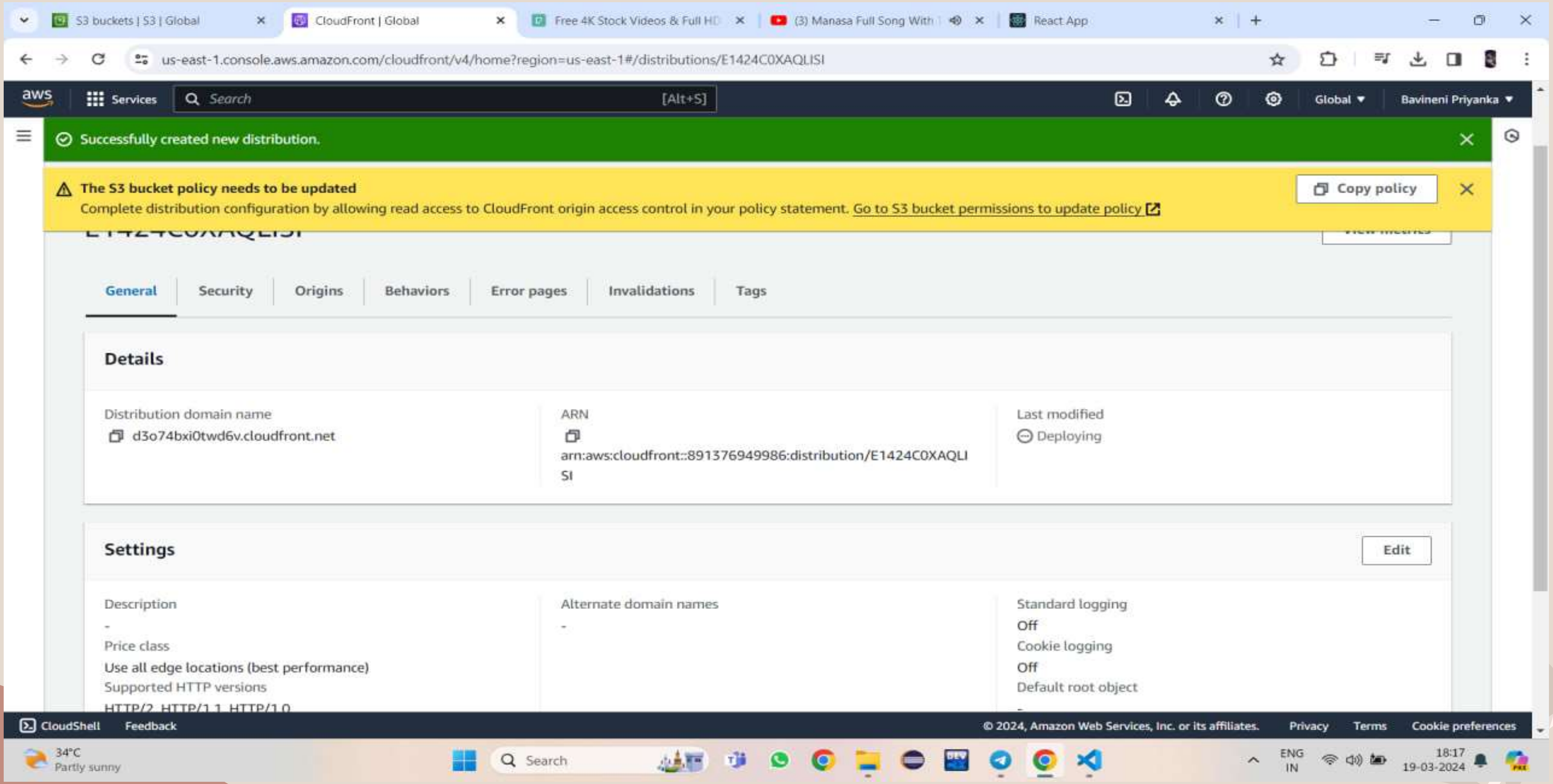
Origin domain
Choose an AWS origin, or enter your origin's domain name.
priyanka-video-streaming-service.s3.us-east-1.amazonaws.com

Origin path - *optional*
Enter a URL path to append to the origin domain name for origin requests.
Enter the origin path

Name
Enter a name for this origin.
priyanka-video-streaming-service.s3.us-east-1.amazonaws.com

Origin access Info
- [ ] Public
  Bucket must allow public access.
- [✓] Origin access control settings (recommended)
  Bucket can restrict access to only CloudFront.
- [ ] Legacy access identities
  Use a CloudFront origin access identity (OAI) to access the S3 bucket.

Origin access control
Select an existing origin access control (recommended) or create a new control.


S3 buckets | S3 | Global
CloudFront | Global
Free 4K Stock Videos & Full HD
(3) Manasa Full Song With
React App
us-east-1.console.aws.amazon.com/cloudfront/v4/home?region=us-east-1#/distributions/E1424C0XAQLISI
Services
Search [Alt+S]
Global
Bavineni Priyanka

Successfully created new distribution.

The S3 bucket policy needs to be updated

Complete distribution configuration by allowing read access to CloudFront origin access control in your policy statement. Go to S3 bucket permissions to update policy

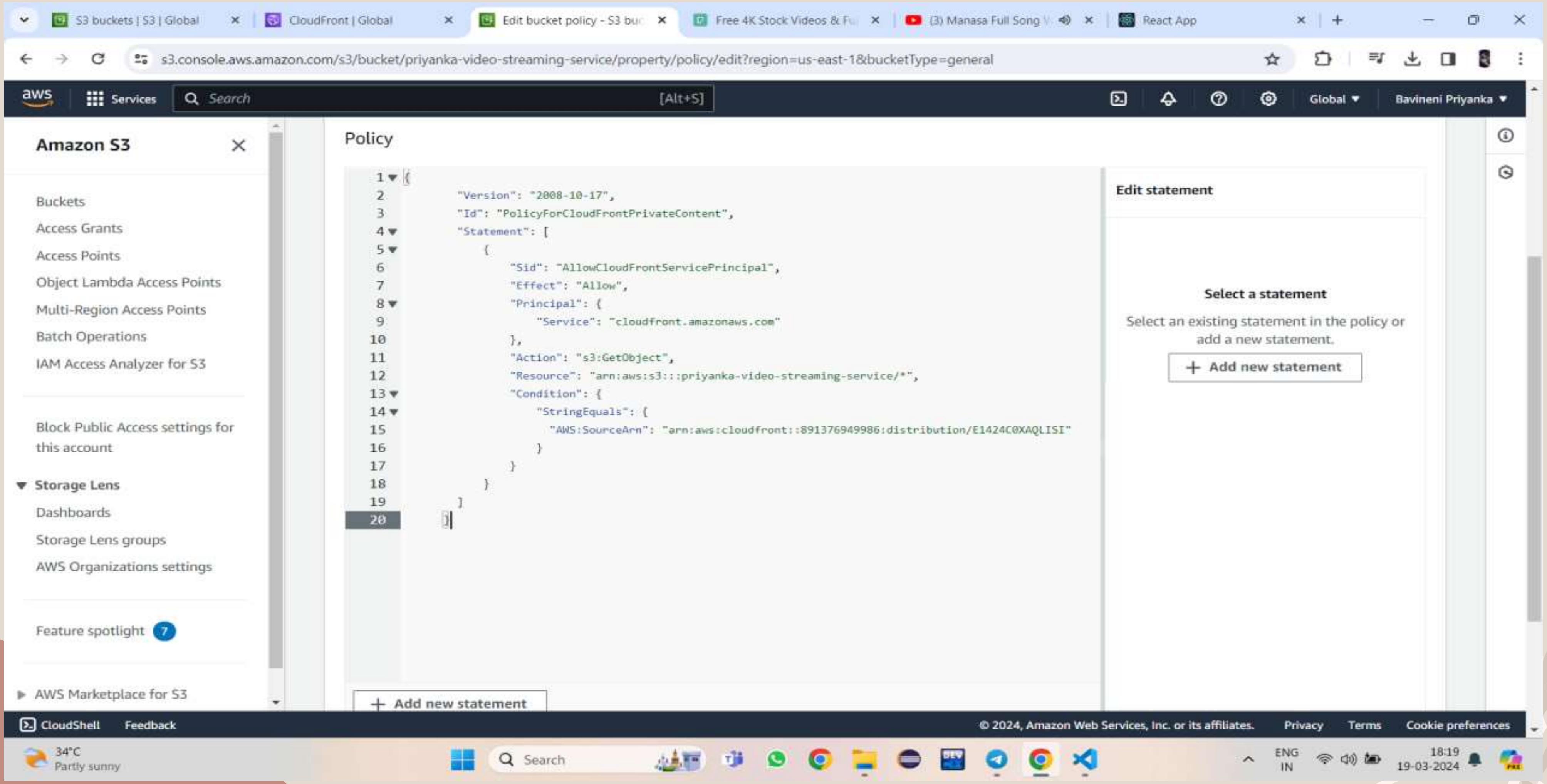
Details

Distribution domain name	ARN	Last modified
d3o74bxi0twd6v.cloudfront.net	arn:aws:cloudfront::891376949986:distribution/E1424C0XAQLISI	Deploying

Settings

Description	Alternate domain names	Standard logging
-	-	Off
Price class		Cookie logging
Use all edge locations (best performance)		Off
Supported HTTP versions		Default root object
HTTP/2, HTTP/1.1, HTTP/1.0		-

Step-5: After creating distribution we have to modify the policy of s3, for that copy the policy and paste it in s3 policy



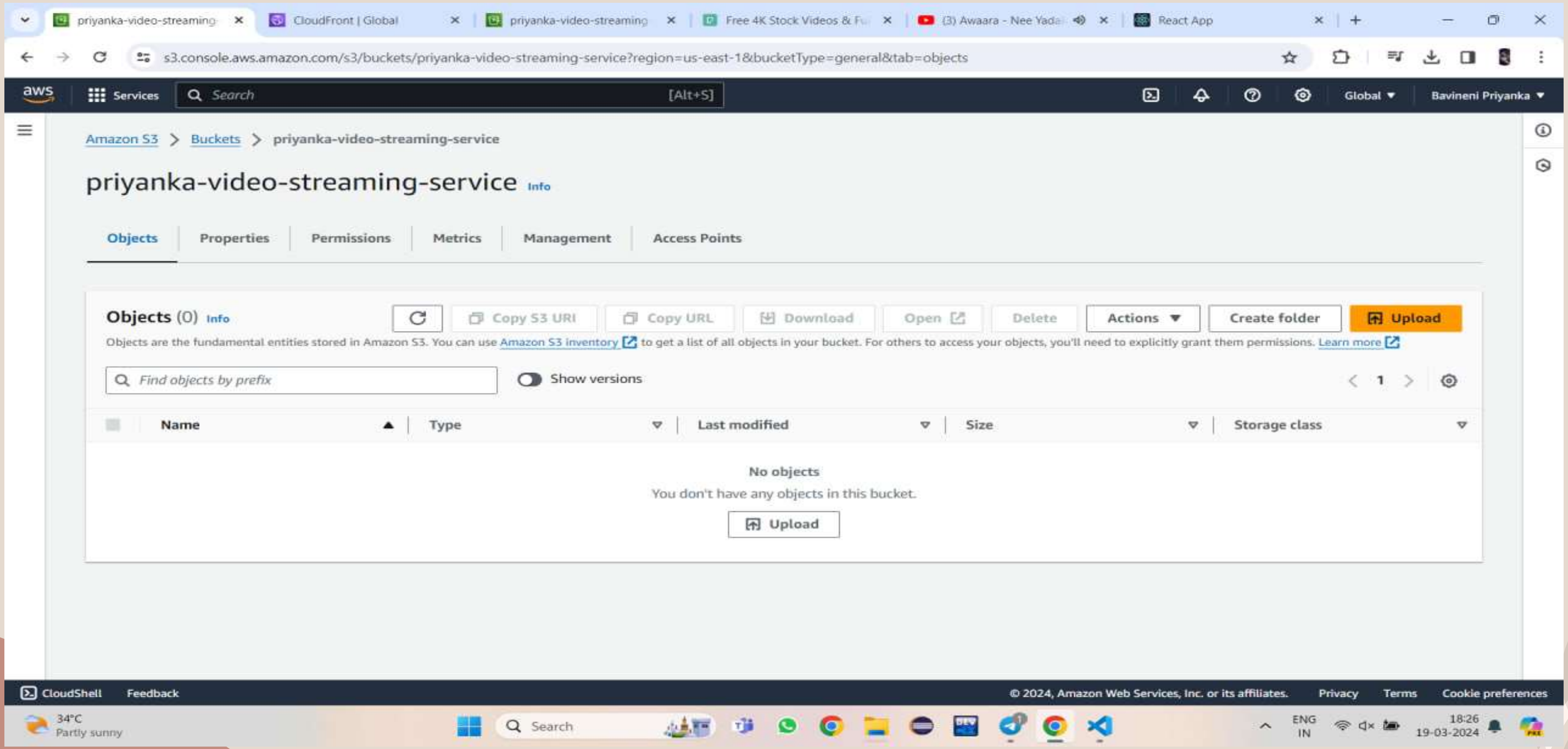
s3.console.aws.amazon.com/s3/bucket/priyanka-video-streaming-service/property/policy/edit?region=us-east-1&bucketType=general

Policy

```
{
        "Version": "2008-10-17",
        "Id": "PolicyForCloudFrontPrivateContent",
        "Statement": [
            {
                "Sid": "AllowCloudFrontServicePrincipal",
                "Effect": "Allow",
                "Principal": {
                    "Service": "cloudfront.amazonaws.com"
                },
                "Action": "s3:GetObject",
                "Resource": "arn:aws:s3:::priyanka-video-streaming-service/*",
                "Condition": {
                    "StringEquals": {
                      "AWS:SourceArn": "arn:aws:cloudfront::891376949986:distribution/E1424C0XAQLISI"
                    }
                }
            }
        ]
    }
```

Amazon S3
Buckets
Access Grants
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
Storage Lens groups
AWS Organizations settings
Feature spotlight 7
AWS Marketplace for S3
Edit statement
Select a statement
Select an existing statement in the policy or add a new statement.
+ Add new statement
+ Add new statement
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Step6: Go to s3 bucket and click on it and select upload option to select the video file



Amazon S3 > Buckets > priyanka-video-streaming-service

priyanka-video-streaming-service Info

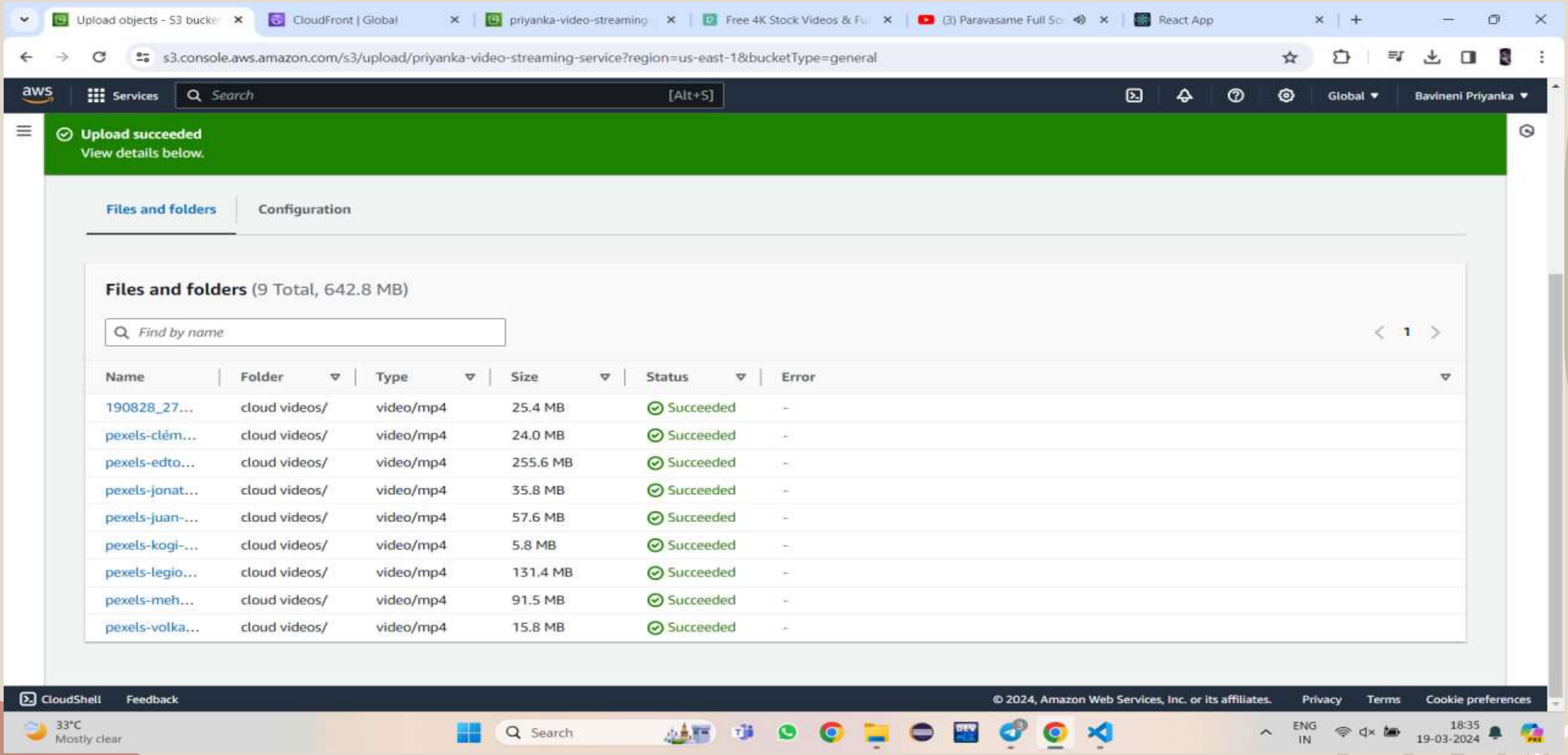
Objects (0) Info

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

Name	Type	Last modified	Size	Storage class

No objects

You don't have any objects in this bucket.



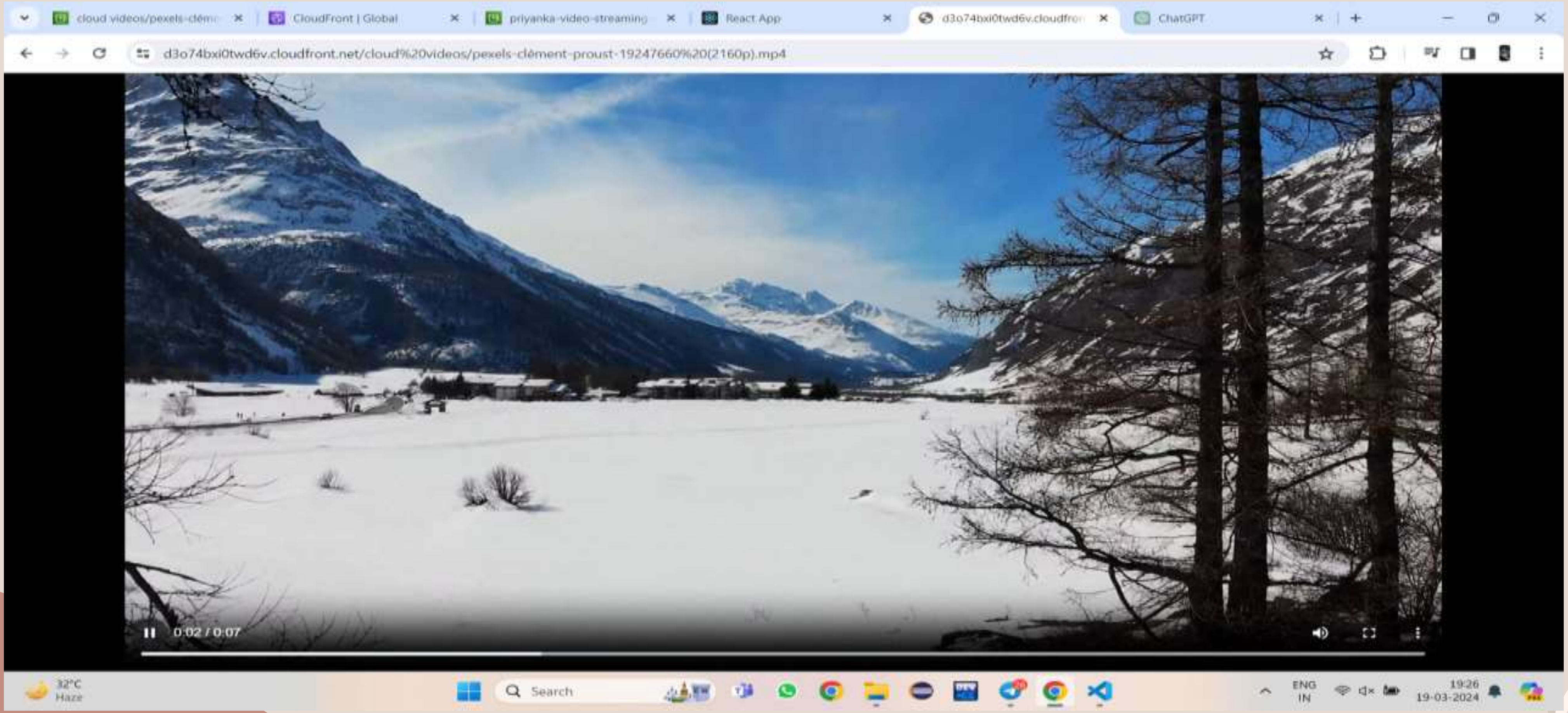
Upload succeeded
View details below.

Files and folders | Configuration

Files and folders (9 Total, 642.8 MB)

Name	Folder	Type	Size	Status	Error
190828_27...	cloud videos/	video/mp4	25.4 MB	Succeeded	-
pexels-clém...	cloud videos/	video/mp4	24.0 MB	Succeeded	-
pexels-edto...	cloud videos/	video/mp4	255.6 MB	Succeeded	-
pexels-jonat...	cloud videos/	video/mp4	35.8 MB	Succeeded	-
pexels-juan-...	cloud videos/	video/mp4	57.6 MB	Succeeded	-
pexels-kogi-...	cloud videos/	video/mp4	5.8 MB	Succeeded	-
pexels-legio...	cloud videos/	video/mp4	131.4 MB	Succeeded	-
pexels-meh...	cloud videos/	video/mp4	91.5 MB	Succeeded	-
pexels-volka...	cloud videos/	video/mp4	15.8 MB	Succeeded	-

Step7: Go to Distribution and copy the domain name and paste it in a web browser, and copy the key name and paste it in the web browser then we get like this



Step8: After getting this we have to go to VS code and create a react app for that we use these commands:

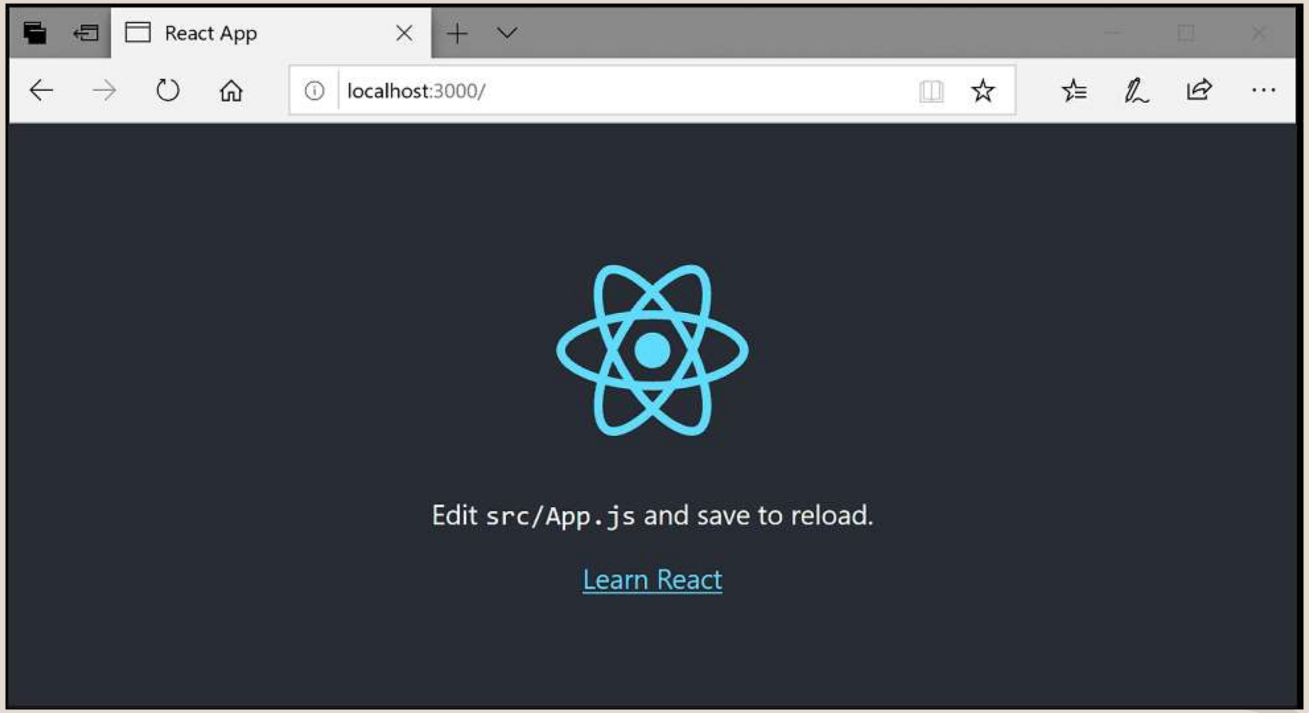
for creating

```
npx create-react-app my-video-app
```

for start the app we have to use these commands:

```
cd my-video-app
npm start
```

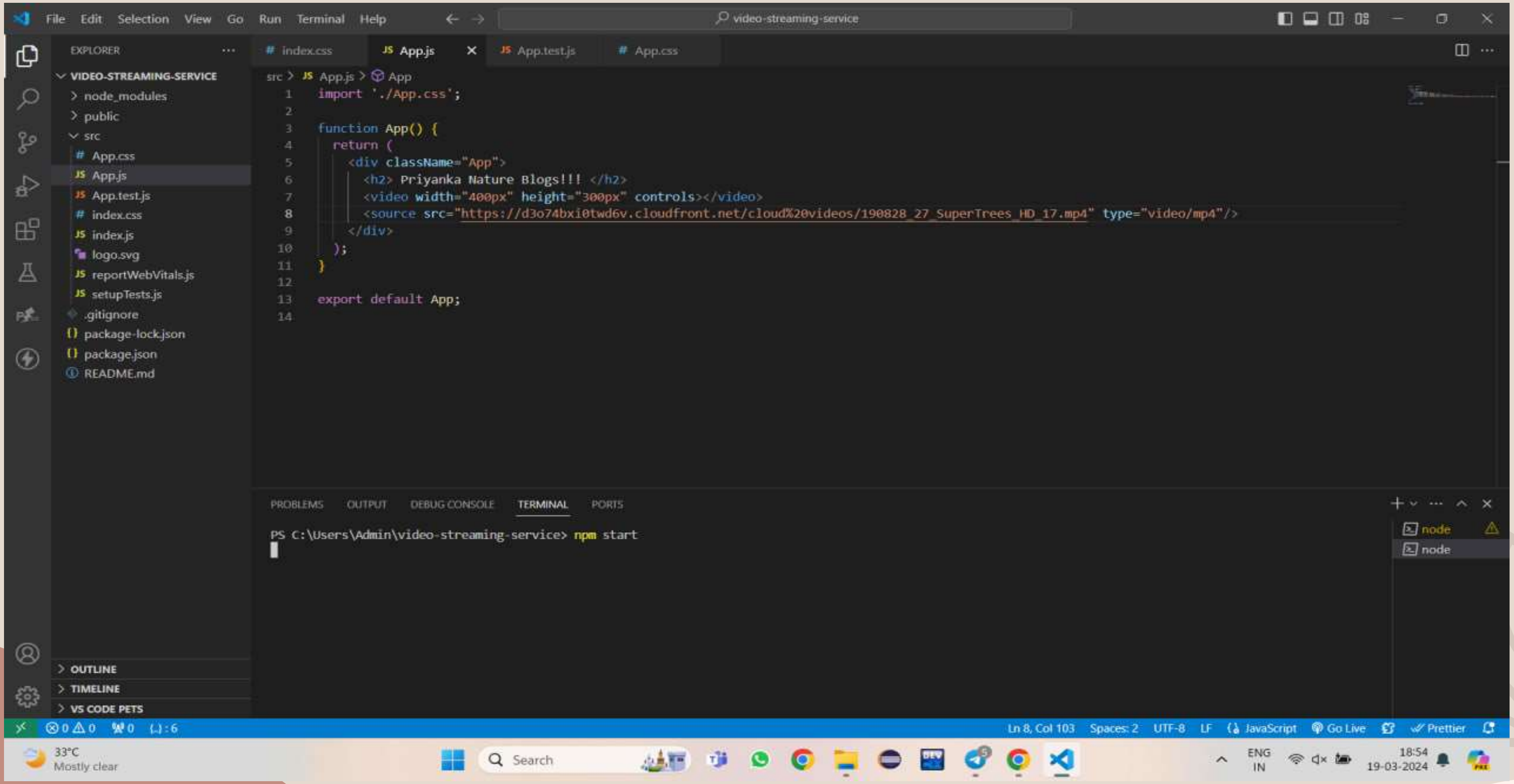
then we get a page like this



Edit `src/App.js` and save to reload.

Learn React

Step9:After modifying the code we have to save and refresh the web page then we get output

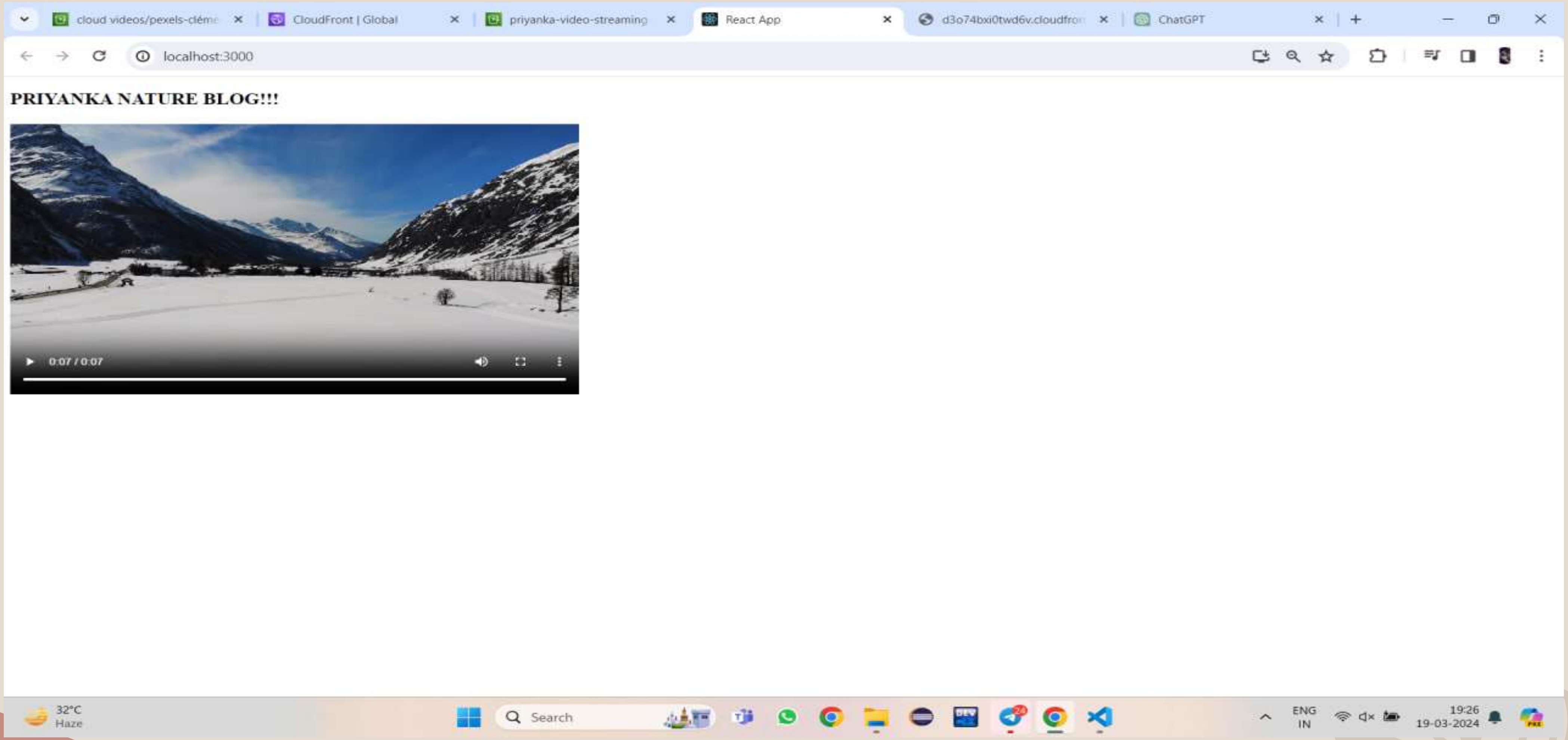


```
import './App.css';

function App() {
  return (
    <div className="App">
      <h2> Priyanka Nature Blogs!!! </h2>
      <video width="400px" height="300px" controls></video>
      <source src="https://d3o74bxi0twd6v.cloudfront.net/cloud%20videos/190828_27_SuperTrees_HD_17.mp4" type="video/mp4"/>
    </div>
  );
}

export default App;
```

```
PS C:\Users\Admin\video-streaming-service> npm start
```



PRIYANKA NATURE BLOG!!!

Conclusion

In conclusion, leveraging Amazon Web Services (AWS) to create a serverless video streaming service offers a game-changing solution for businesses and developers alike. By harnessing the power of AWS services such as Amazon S3, React, and CloudFront, we can significantly reduce complexity and costs while enhancing scalability and efficiency. This architecture empowers us to deliver seamless video streaming experiences that meet the demands of today's digital landscape. With AWS, building and managing a robust video streaming platform has never been more accessible, enabling us to focus on delivering high-quality content to our users without the burden of traditional server setups.

Links

LinkedIn Article Link: https://www.linkedin.com/pulse/video-streaming-service-priyanka-bavineni-6tkbc

YouTube Link: https://youtu.be/bNO1TD7v4xE thank you