Name : Shreyash Santosh Sahu
Sem : 7th Sem CSE-A
Roll no. : 62

TA-1

Aim :

To create a file upload system in Next.js that utilizes the AWS S3 service, enabling users to upload files and maintain a centralized data repository for their needs.

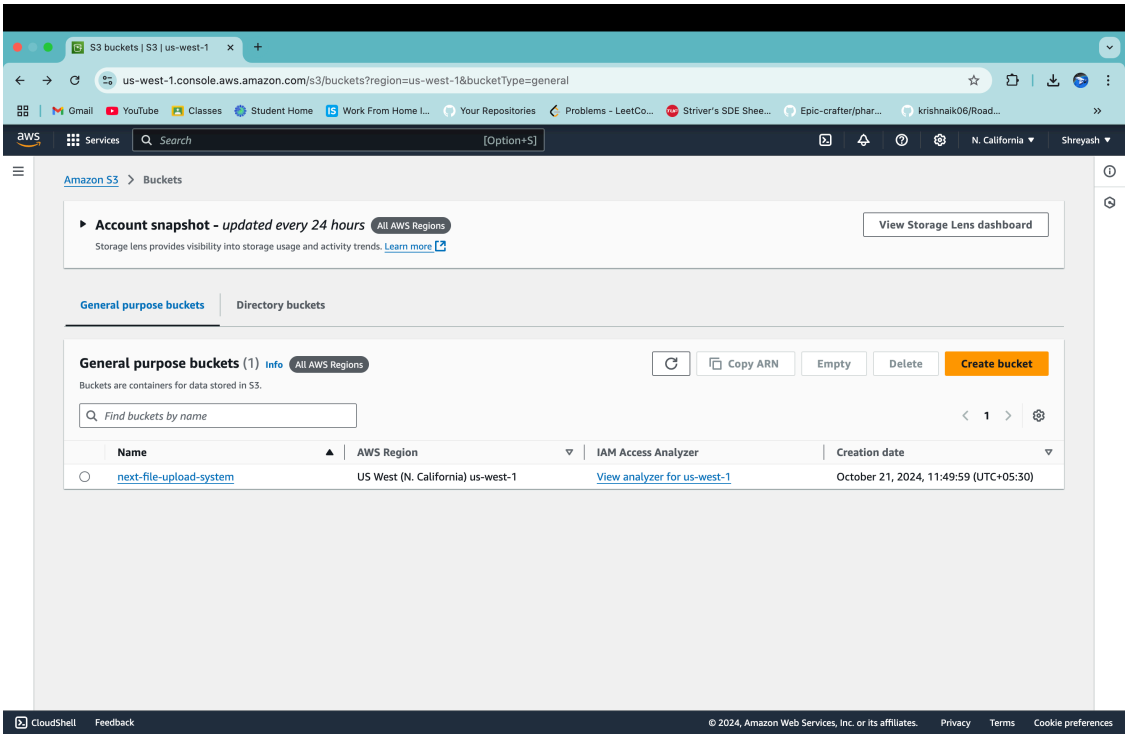
Github repository link : https://github.com/1-Shreyash/ccta

PERFORMANCE PICS :

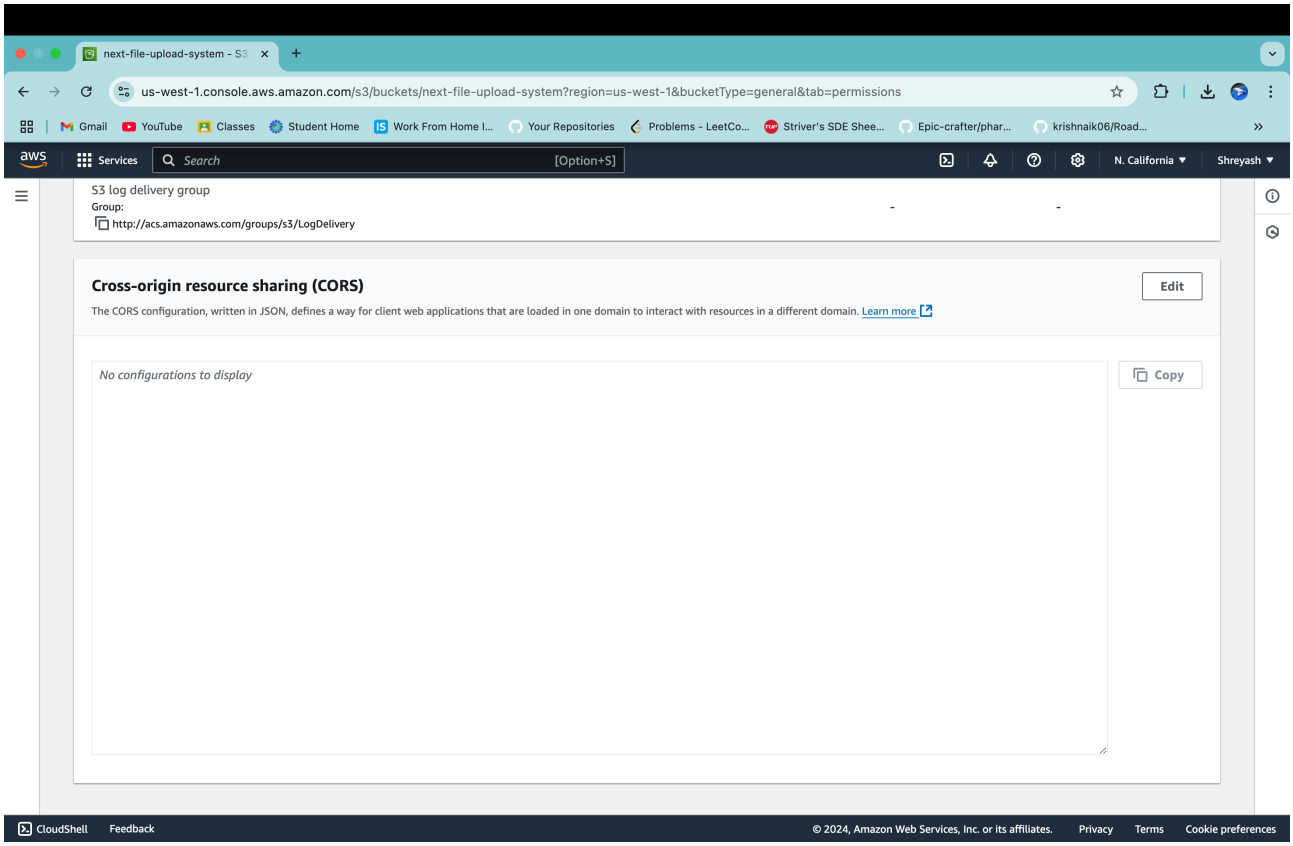
Here's a step-by-step guide to set up AWS S3 for file uploads, including configuring bucket policies and roles.

Step 1: Set Up an AWS S3 Bucket

1. **Sign in to AWS**:
   - Go to the AWS Management Console.
   - If you don't have an AWS account, create one.
2. **Open the S3 Service**:
   - In the AWS Management Console, search for "S3" in the search bar at the top.
   - Click on **S3** to open the S3 dashboard.

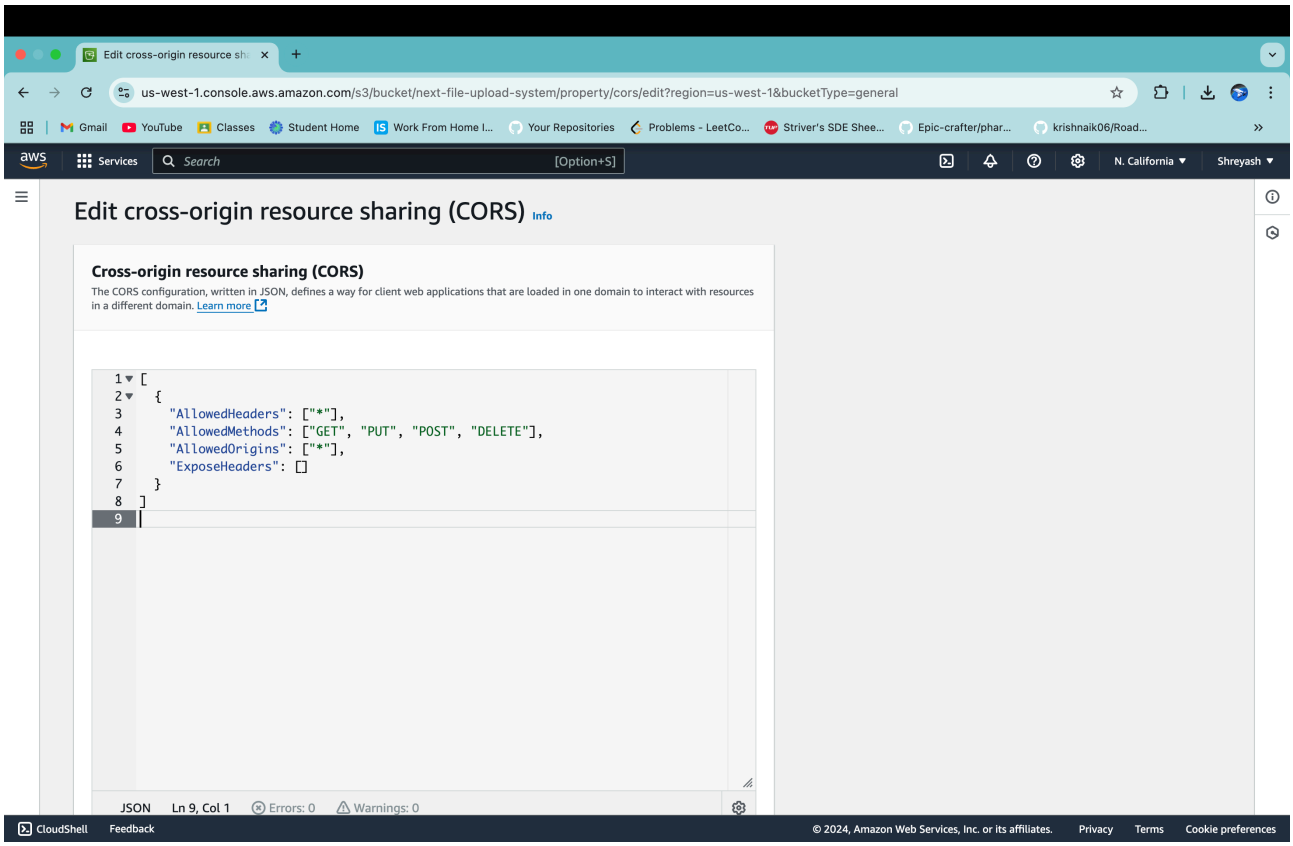


3. **Create a New Bucket**:
   - Click the **Create bucket** button.
   - **Bucket Name**: Provide a globally unique name for your bucket (e.g., `my-nextjs-file-uploads`).
   - **Region**: Choose the AWS region where you want the bucket to reside (e.g., `us-east-1`).
   - **Bucket settings**: You can leave most settings as default.
   - **Block Public Access**: Ensure public access is blocked if you don't want files to be publicly accessible. Otherwise, allow public access based on your project needs.
   - Click **Create Bucket**.



4. Enable CORS Configuration (if your app makes requests directly to S3):

- In the S3 bucket dashboard, click on your bucket.
- Go to the **Permissions** tab.
- Scroll down to **CORS Configuration** and click **Edit**.
- Add the following CORS policy to allow file uploads:

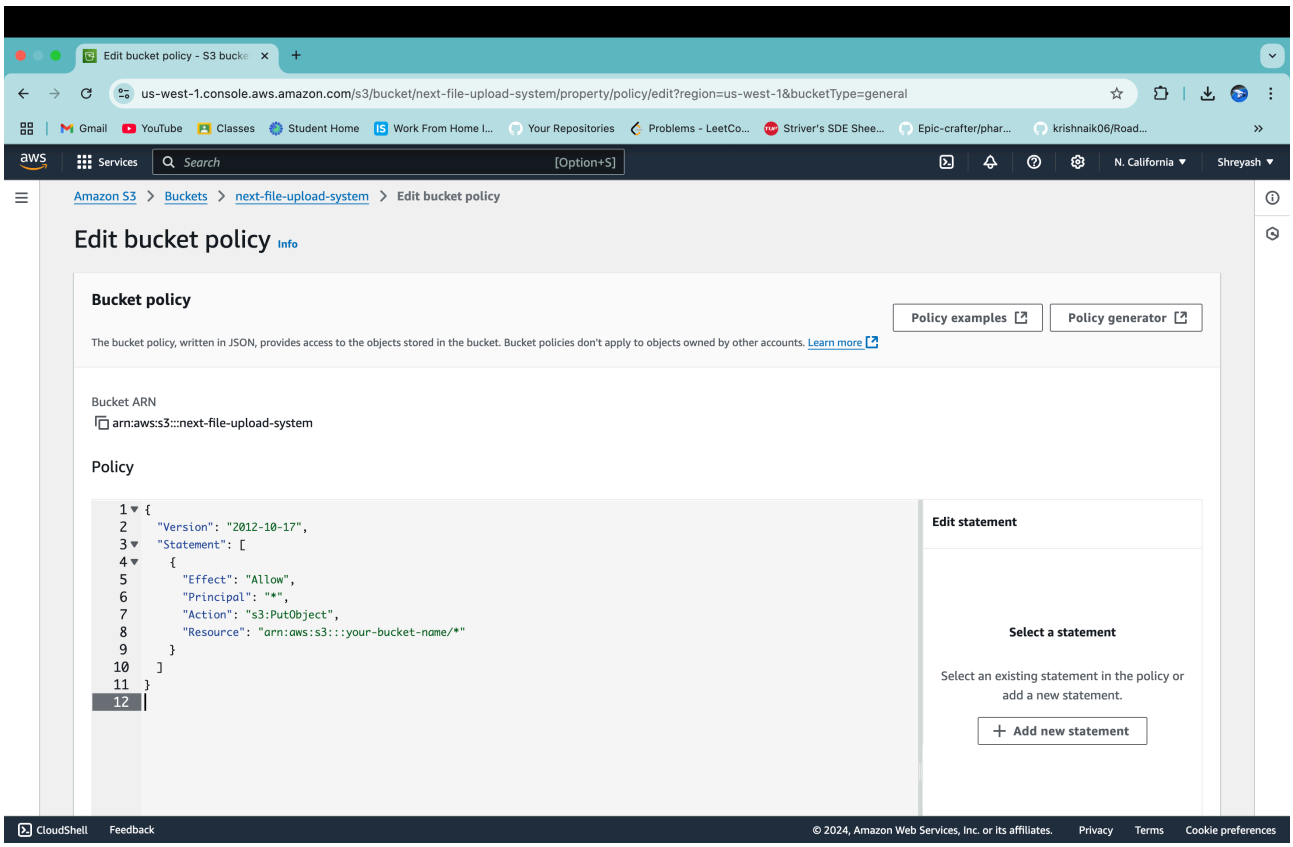


Click **Save changes**.

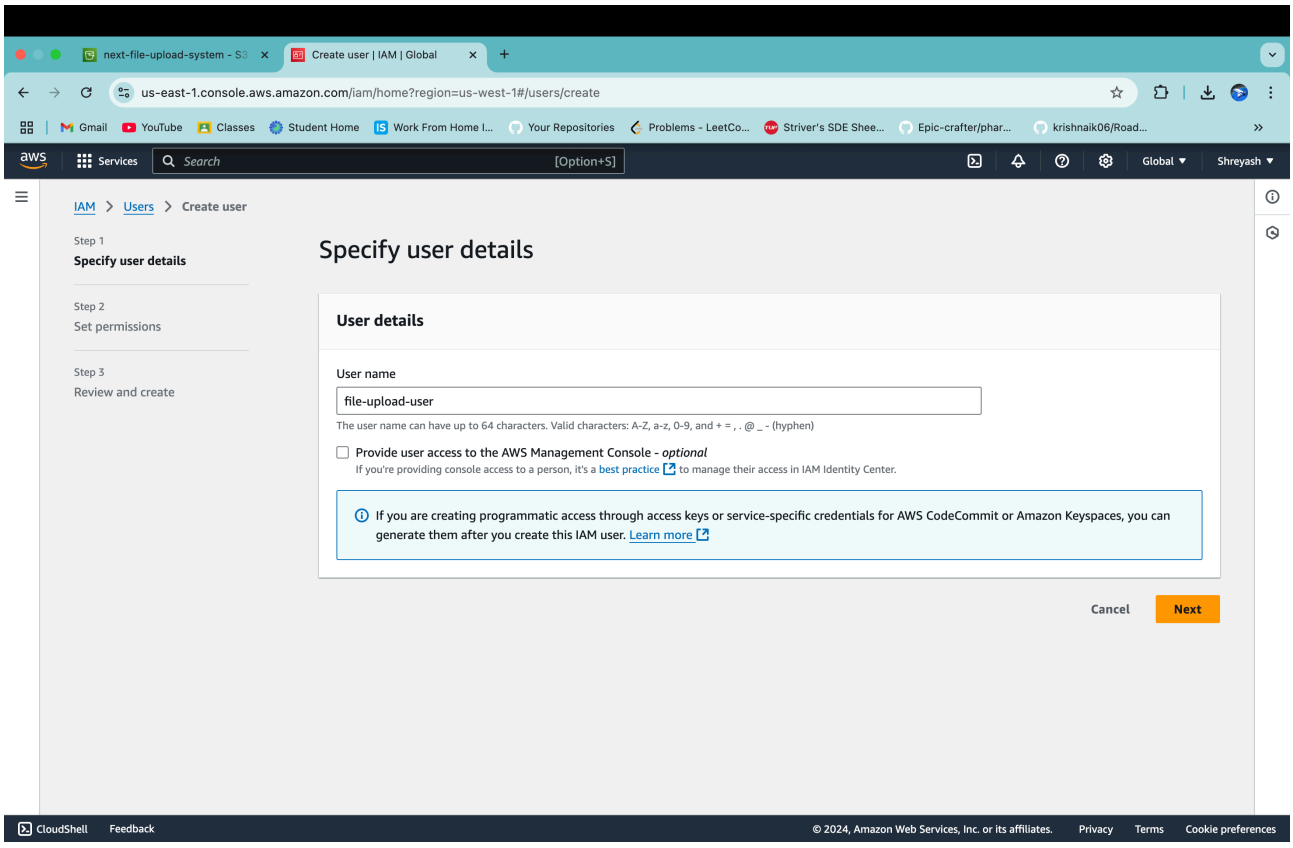
Step 2: Set Bucket Policy

To allow uploading files to the S3 bucket, configure the bucket policy.

1. **Navigate to Bucket Policy**:
    - On your bucket's **Permissions** tab, scroll down to the **Bucket policy** section.
    - Click **Edit**.
2. **Add Bucket Policy**:
    - Use the following JSON policy template to grant public access or restrict access based on your needs. Here's an example that allows uploading from specific IAM roles or users:



Replace `your-bucket-name` with your actual bucket name.
This policy allows all users to upload objects. You can restrict this by specifying specific users/roles under `"Principal"` if needed.

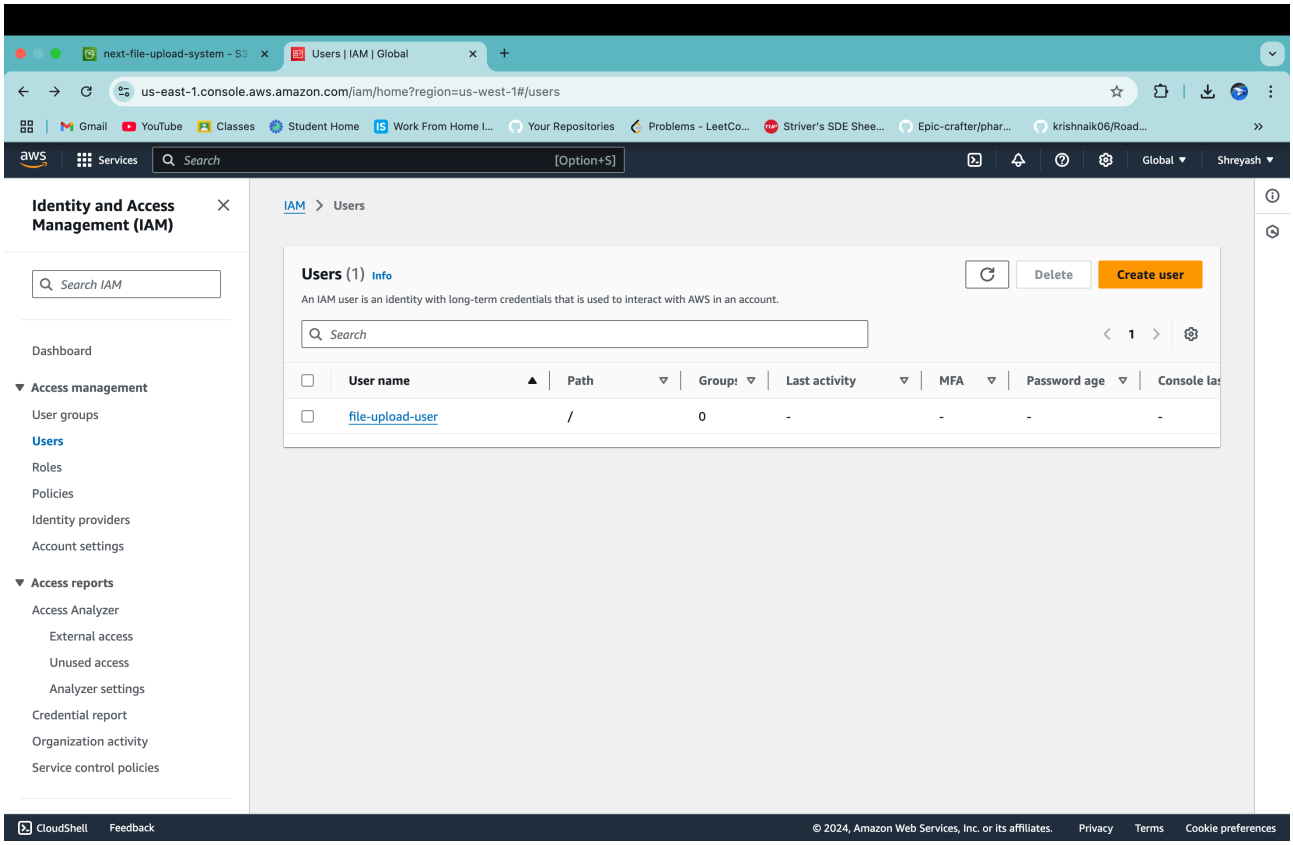


Save the Policy.

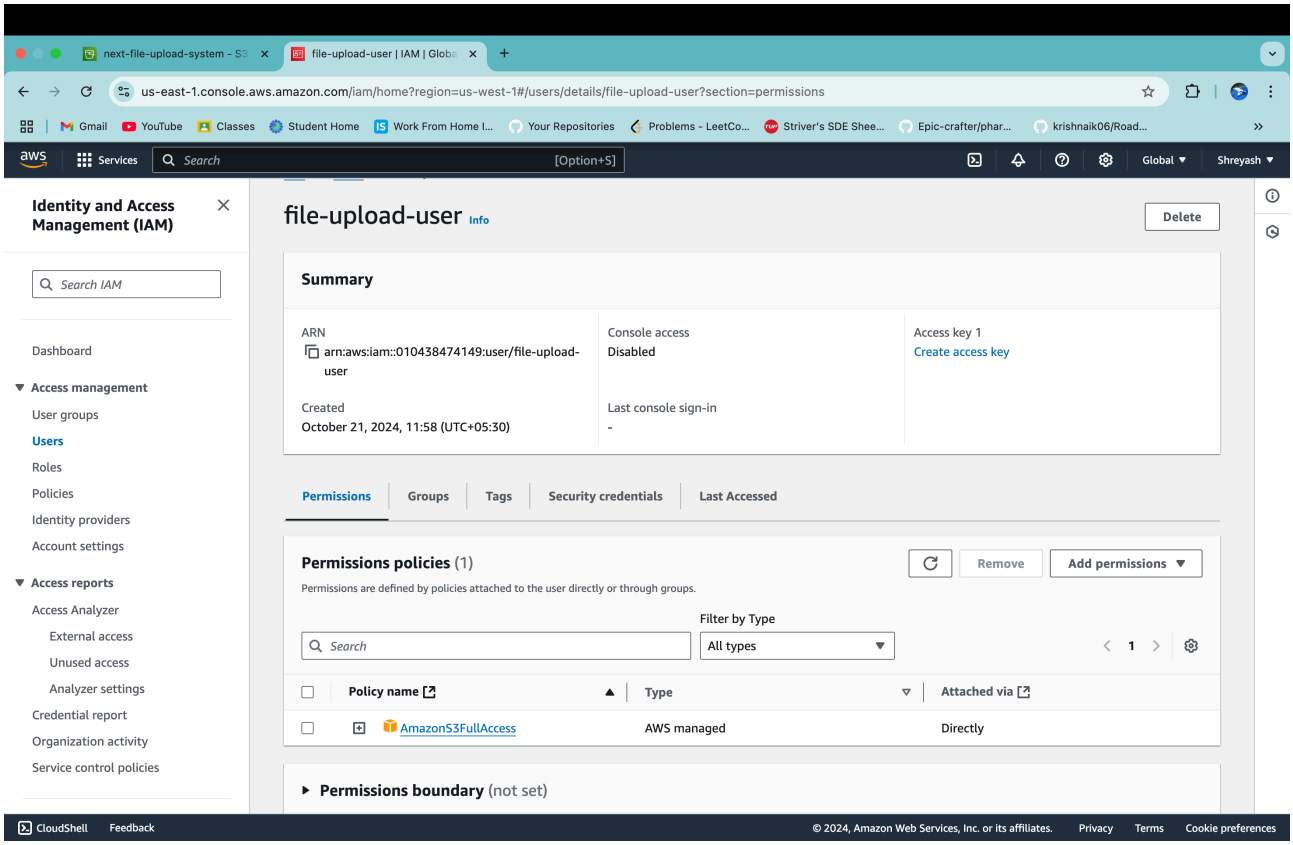
Step 3: Set Up IAM Roles and Permissions

To securely upload files, create a role or user with access to the S3 bucket.

1. **Go to IAM Service**:
    - In the AWS Console, search for **IAM** (Identity and Access Management).
2. **Create a New IAM User or Role**:
    - Click on **Users** (for user-specific access) or **Roles** (for service access like Lambda, EC2, etc.).
    - Click on **Add User** or **Create Role**.



3. **Set Permissions**:
    - For the user/role, under **Permissions**, attach the following policy:
        - Click on **Attach existing policies directly**.
        - Search for **AmazonS3FullAccess** or create a custom policy:

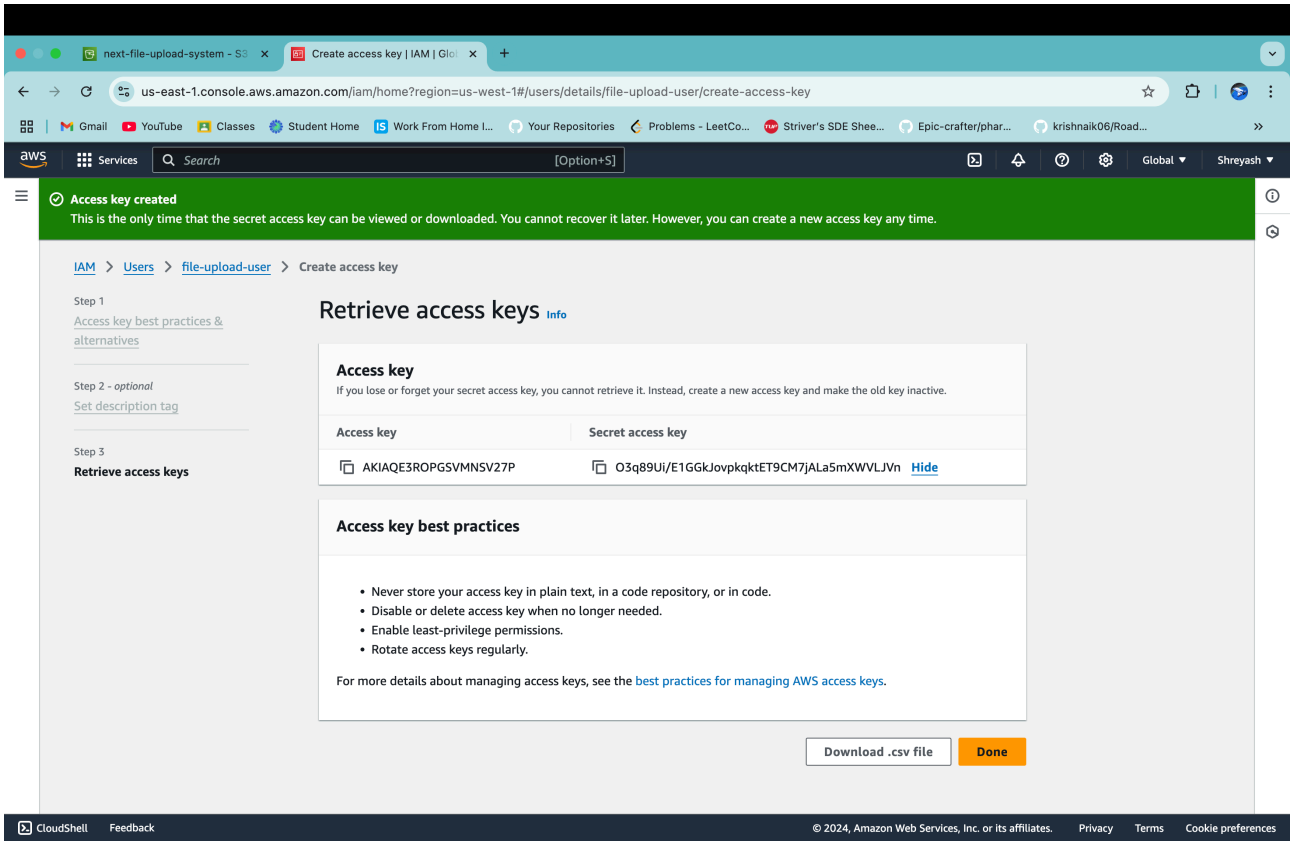


- Replace `your-bucket-name` with your actual bucket name.

Click **Next**, review, and create the user or role.

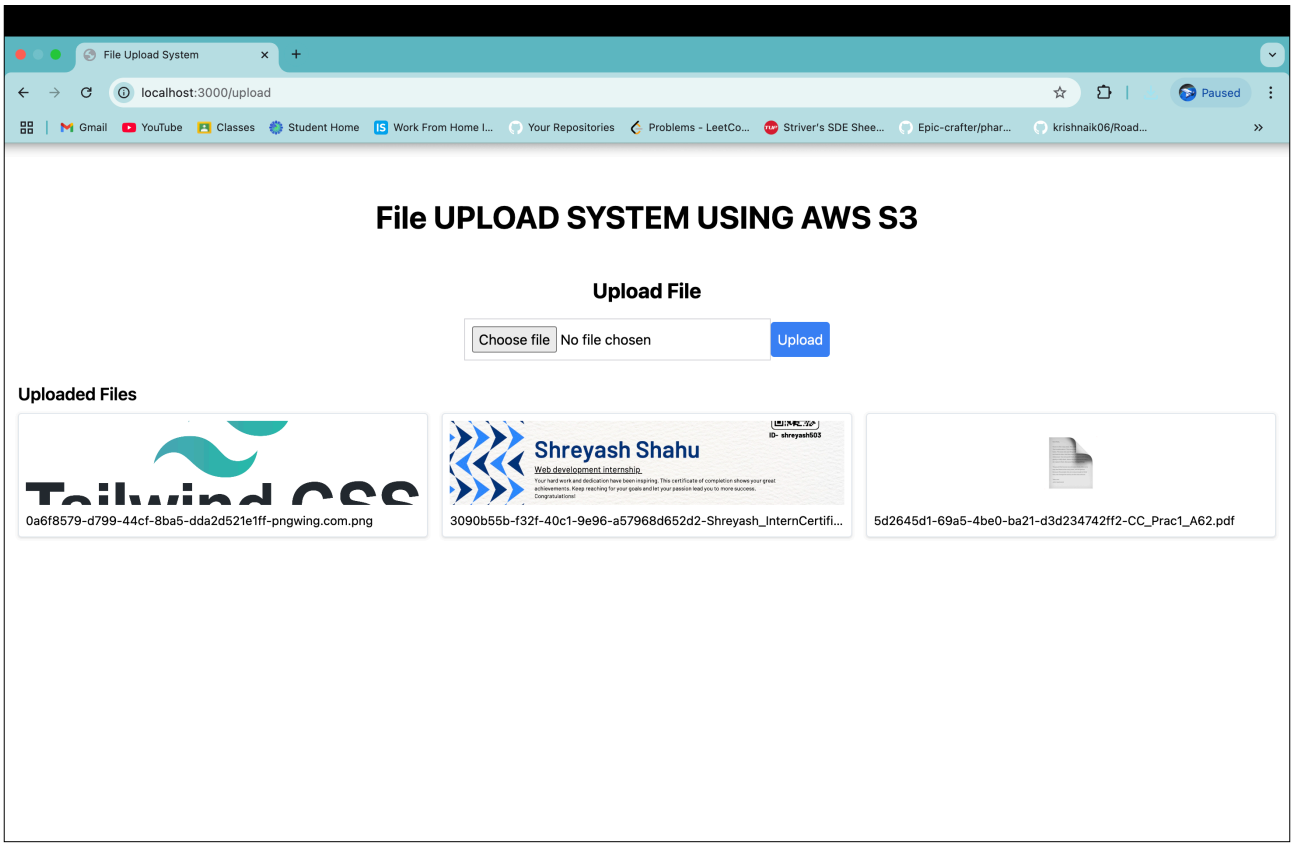
4. Save Access Key and Secret Key:

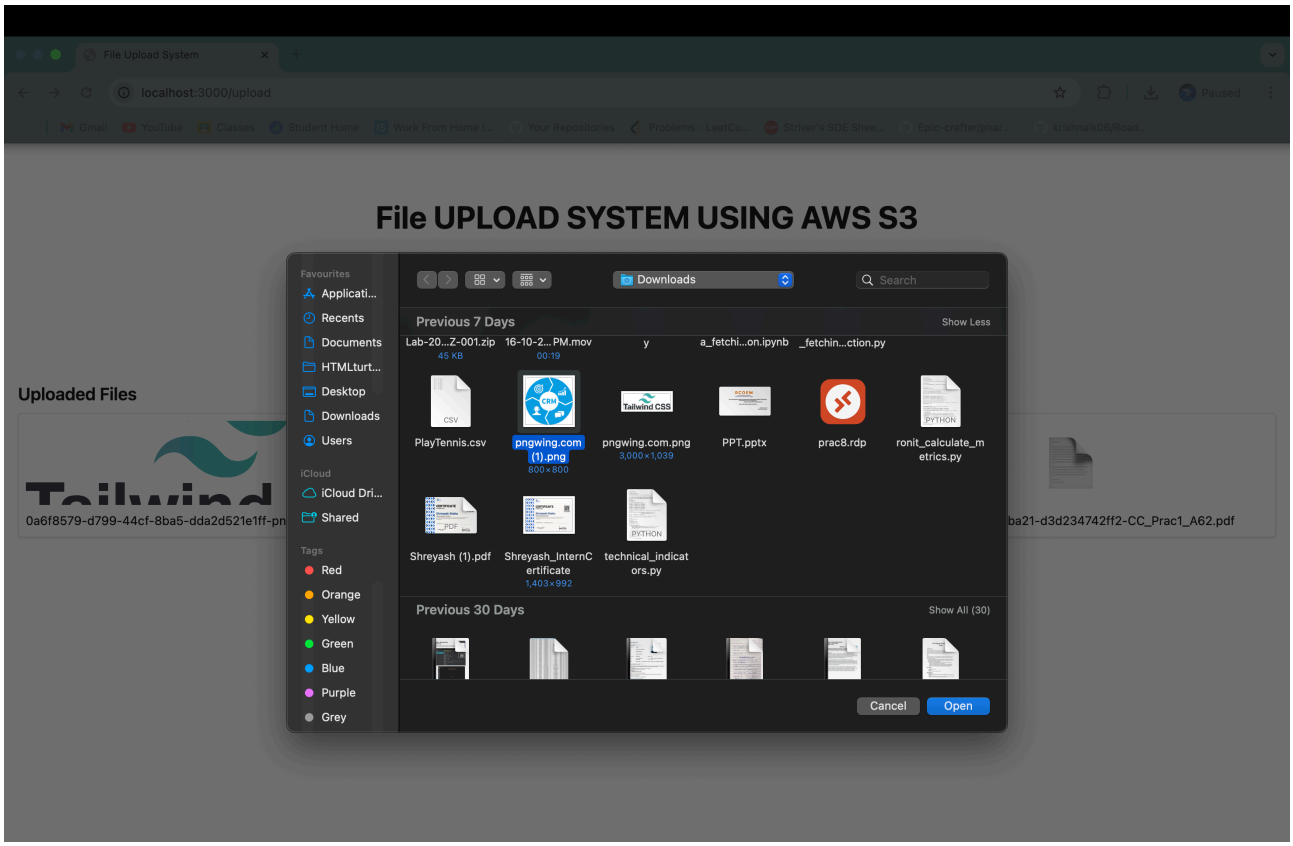
- For users: After creating the user, you'll receive an **Access Key ID** and **Secret Access Key**.
- Save these credentials as you'll need them in the `.env.local` file of your Next.js app.

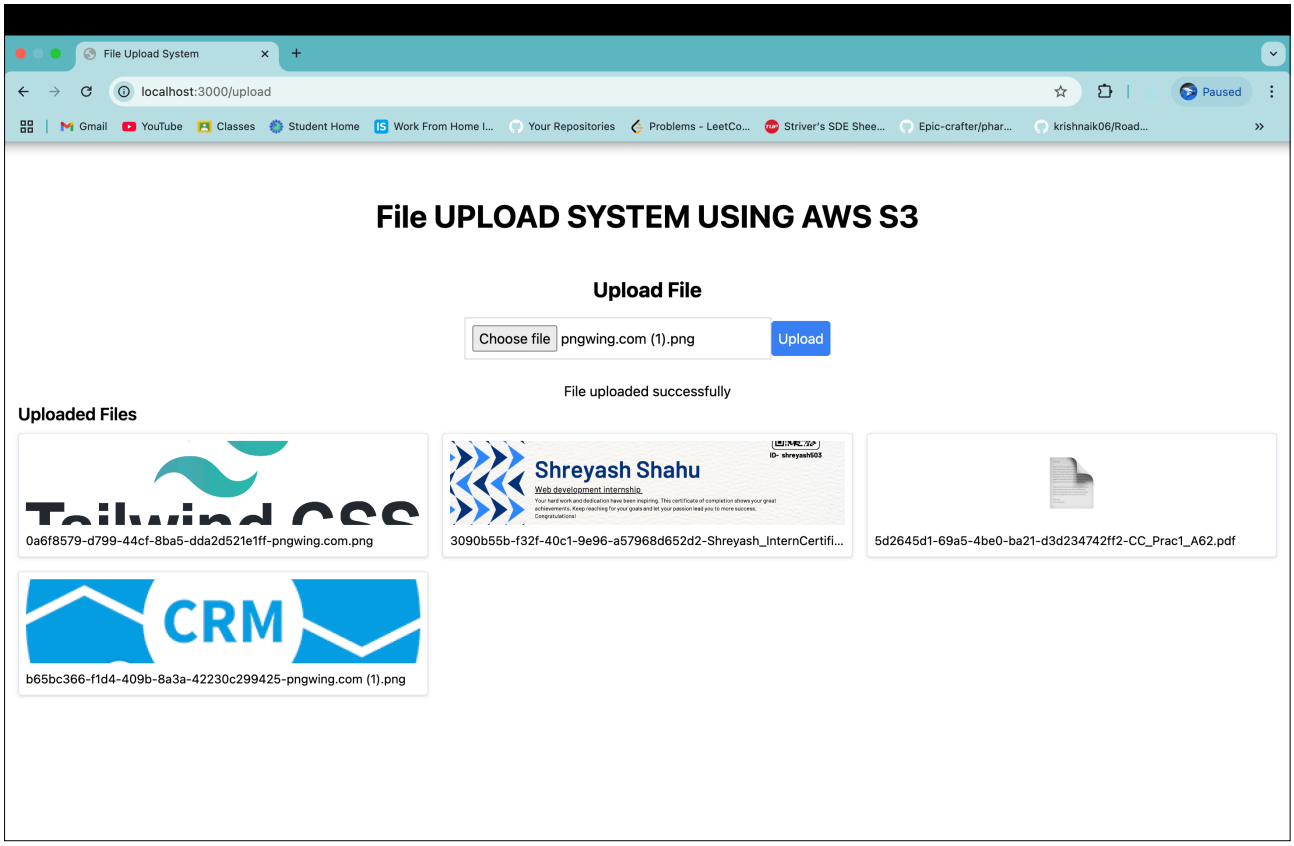


Application :

Github link : https://github.com/1-Shreyash/ccta







All Files getting stored in S3 :