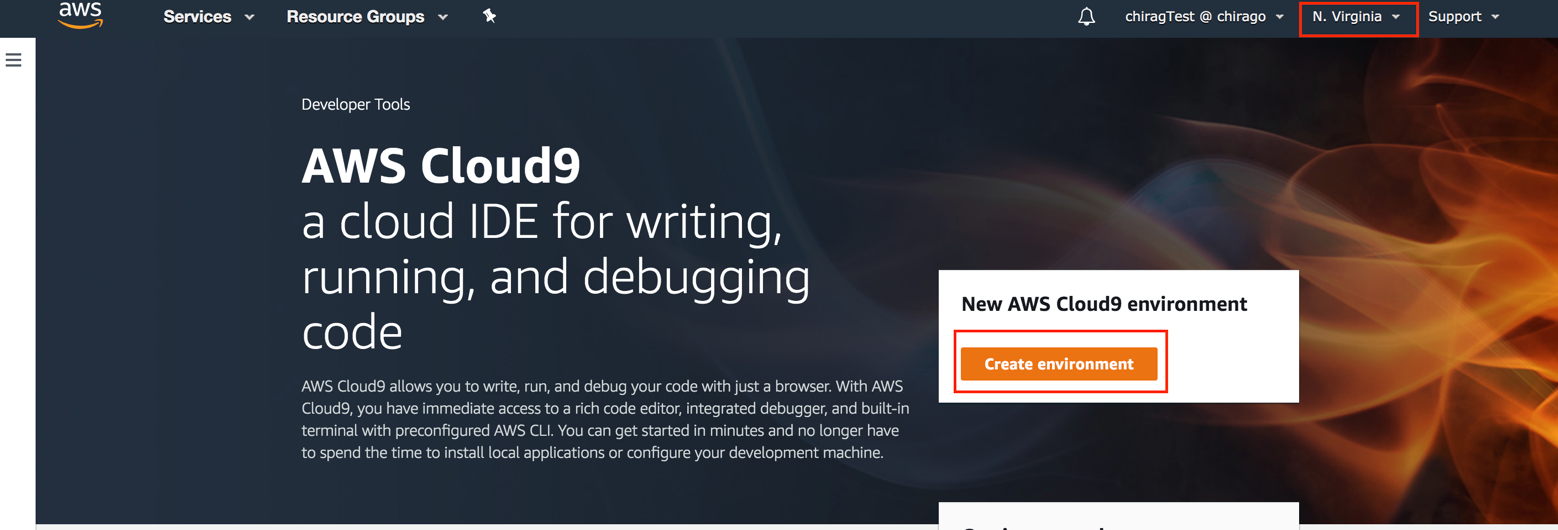
**Step1: Setting up Development Environment**

**Create an Environment**

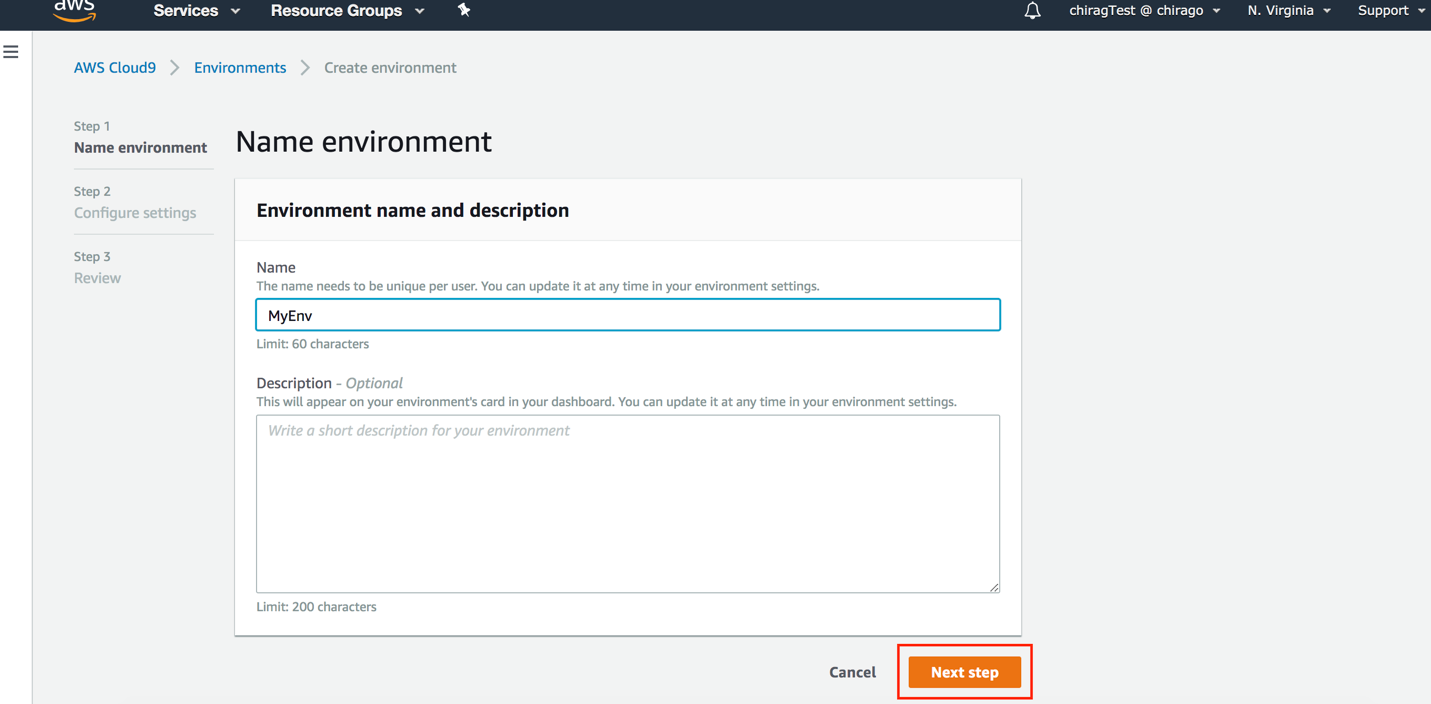
In this step, you use AWS Cloud9 console to create and then open an AWS Cloud9 development environment.

**Create an EC2 Environment with the Console**

1. Sign in to the AWS Cloud9 console as follows:
   * If you're the only individual using your AWS account or you are an IAM user in a single AWS account, go to <https://console.aws.amazon.com/cloud9/>
2. After you sign in to the AWS Cloud9 console, in the top navigation bar, choose an AWS Region to create the environment in. For a list of available AWS Regions, see [AWS Cloud9](https://docs.aws.amazon.com/general/latest/gr/rande.html#cloud9_region) in the *Amazon Web Services General Reference*.
3. If a welcome page is displayed, for **New AWS Cloud9 environment**, choose **Create environment**. Otherwise, choose **Create environment**.



1. On the **Name environment** page, for **Name**, type a name for your environment.



In this tutorial, we use the name my-demo-environment. If you use a different environment name, substitute it throughout this tutorial.

1. For **Description**, type something about your environment. For example, This environment is for the AWS Cloud9 tutorial.
2. Choose **Next step**.
3. On the **Configure settings** page, for **Environment type**, leave the default choice of **Create a new instance for environment (EC2)**.

Choosing **Create a new instance for environment (EC2)** means you want AWS Cloud9 to create a new Amazon EC2 instance and then connect the environment to the newly-created instance. To use an existing cloud compute instance or your own server instead (which we call an *SSH environment*), see [Creating an Environment in AWS Cloud9](https://docs.aws.amazon.com/cloud9/latest/user-guide/create-environment.html).

**Note**

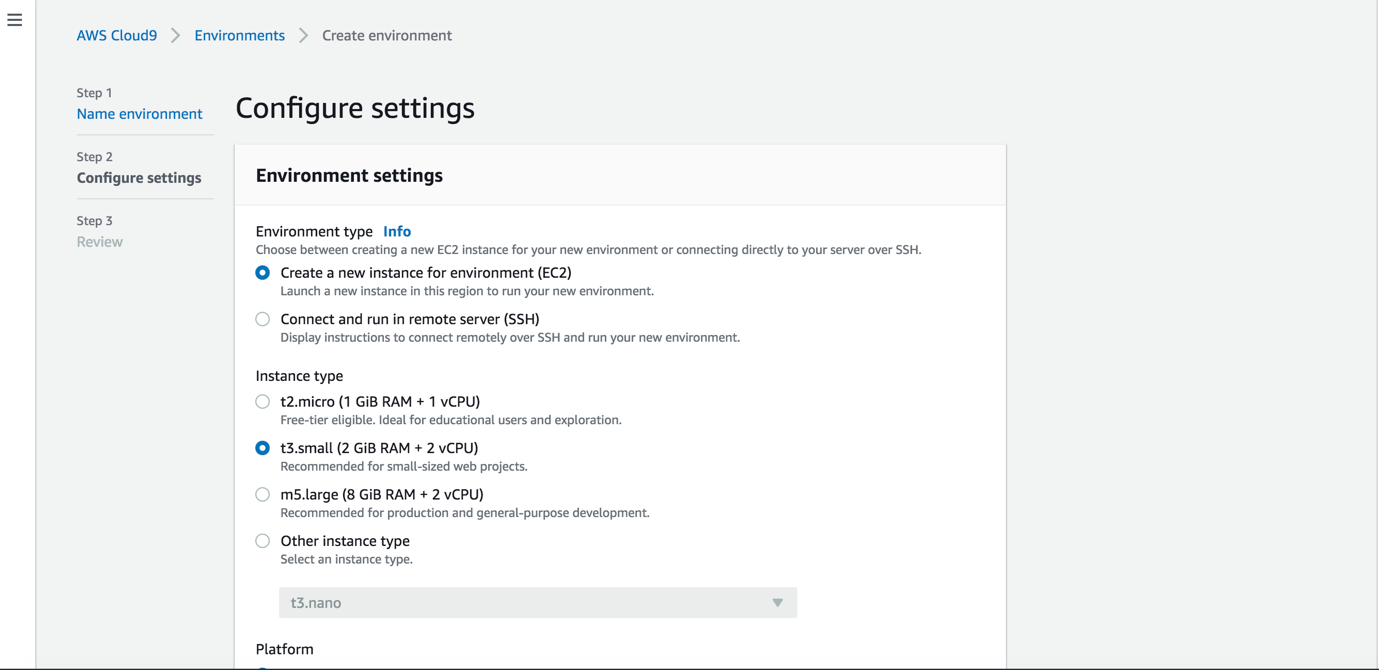
Choosing **Create a new instance for environment (EC2)** might result in possible charges to your AWS account for Amazon EC2.

1. For **Instance type**, select **t3.small**.

**Note**

Choosing instance types with more RAM and vCPUs might result in additional charges to your AWS account for Amazon EC2.

1. For **Platform**, choose the type of Amazon EC2 instance that AWS Cloud9 will create and then connect to this environment: **Amazon Linux** or **Ubuntu**.



1. Expand **Network settings (advanced)**.
2. AWS Cloud9 uses Amazon Virtual Private Cloud (Amazon VPC) to communicate with the newly-created Amazon EC2 instance. Depending on how Amazon VPC is set up, do one of the following.

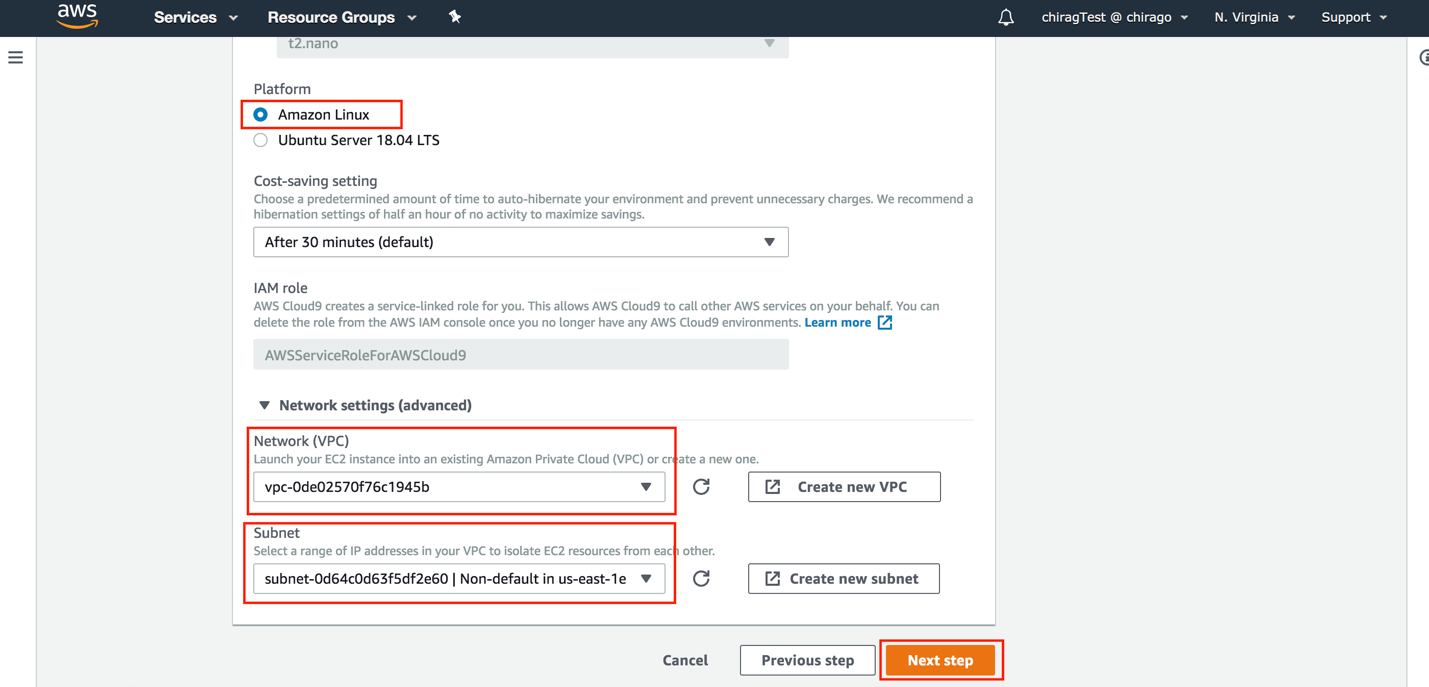
|  |
| --- |
| If you're not sure what to choose, we recommend that you skip ahead to step 12 in this procedure.  When you skip past **Network settings (advanced)** and leave the preselected default settings, AWS Cloud9 attempts to automatically use the default VPC with its single subnet in the same AWS account and AWS Region as the new environment. |

1. For more information, see [VPC Settings for AWS Cloud9 Development Environments](https://docs.aws.amazon.com/cloud9/latest/user-guide/vpc-settings.html).
2. For **Cost-saving setting**, choose the amount of time until AWS Cloud9 shuts down the Amazon EC2 instance for the environment after all web browser instances that are connect to the IDE for the environment have been closed. Or leave the default choice.

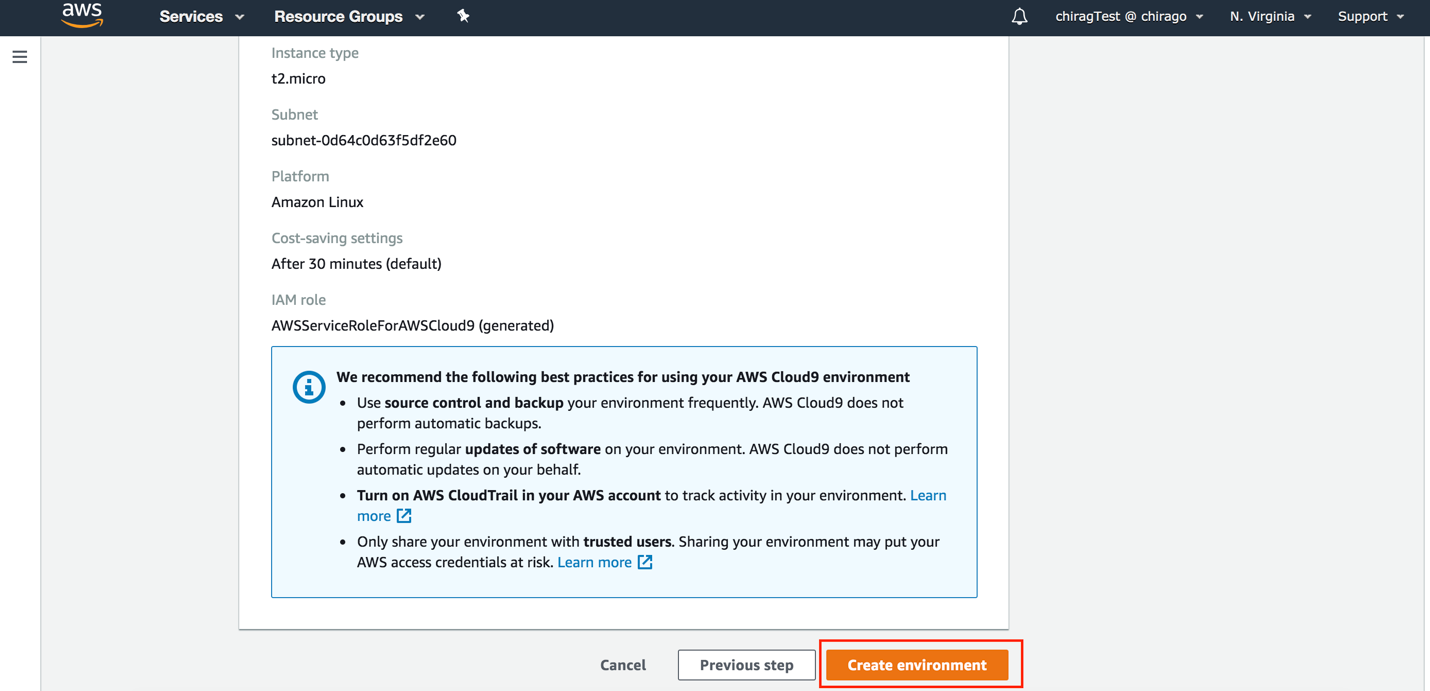
**Note**

Choosing a shorter time period might result in fewer charges to your AWS account. Likewise, choosing a longer time might result in more charges.

1. Choose **Next step**.



1. On the **Review choices** page, choose **Create environment**. Wait while AWS Cloud9 creates your environment. This can take several minutes. Please be patient.



After your environment is created, the AWS Cloud9 IDE is displayed. You'll learn about the AWS Cloud9 IDE in the next step.

If AWS Cloud9 doesn't display the IDE after at least five minutes, there might be a problem with your web browser, your AWS access permissions, the instance, or the associated virtual private cloud (VPC). For possible fixes, see [Cannot Open an Environment](https://docs.aws.amazon.com/cloud9/latest/user-guide/troubleshooting.html#troubleshooting-env-loading) in *Troubleshooting*.

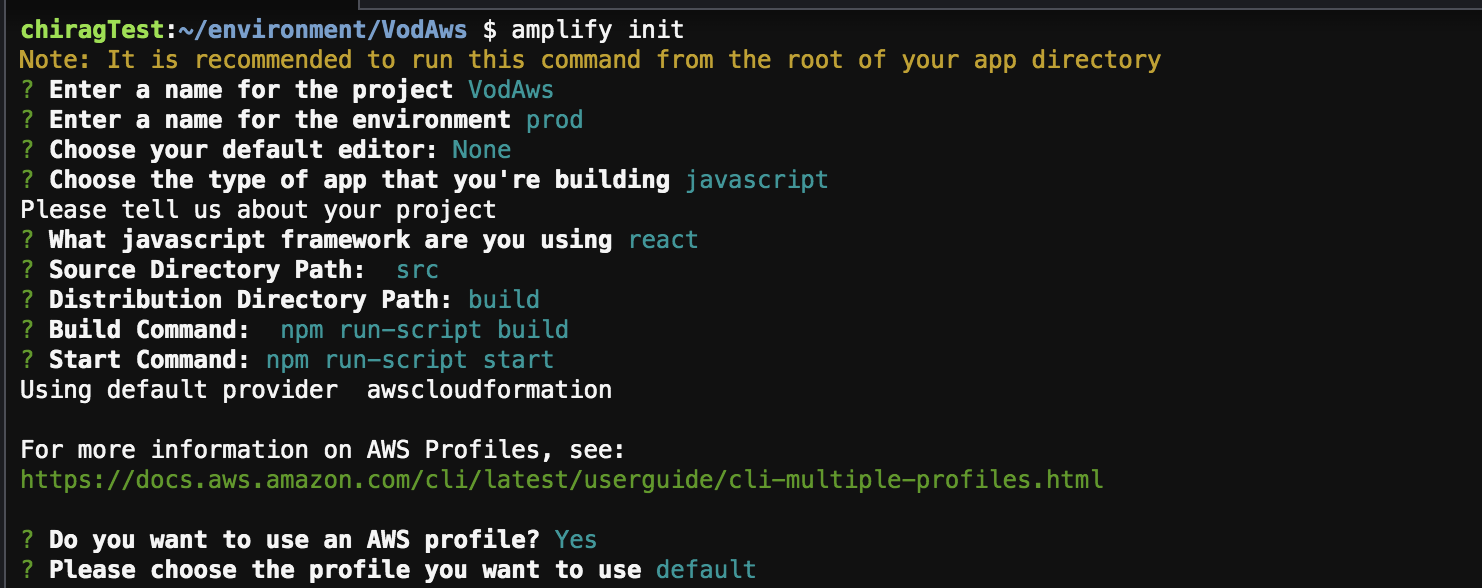
**Step2: Installing Packages and Configuring Environment**

1. The first step is to copy your aws credentials from the credentials file into a file called config. Run the following command in the terminal.
   * cp ~/.aws/credentials ~/.aws/config
2. Install the AWS Amplify CLI using this command.
   * npm install -g @aws-amplify/cli
3. Next, download files for front-end react code and run the below command
   * wget <https://testorlando1.s3.amazonaws.com/VodAws.zip>
4. Unzip the contents by running the below command
   * unzip VodAws.zip
5. Navigate to VodAws directory.
   * cd VodAws
6. We will now be installing the development tools using the Node Package Manager(NPM)
7. Install the node modules, run below command in your VodAws directory
   * npm install

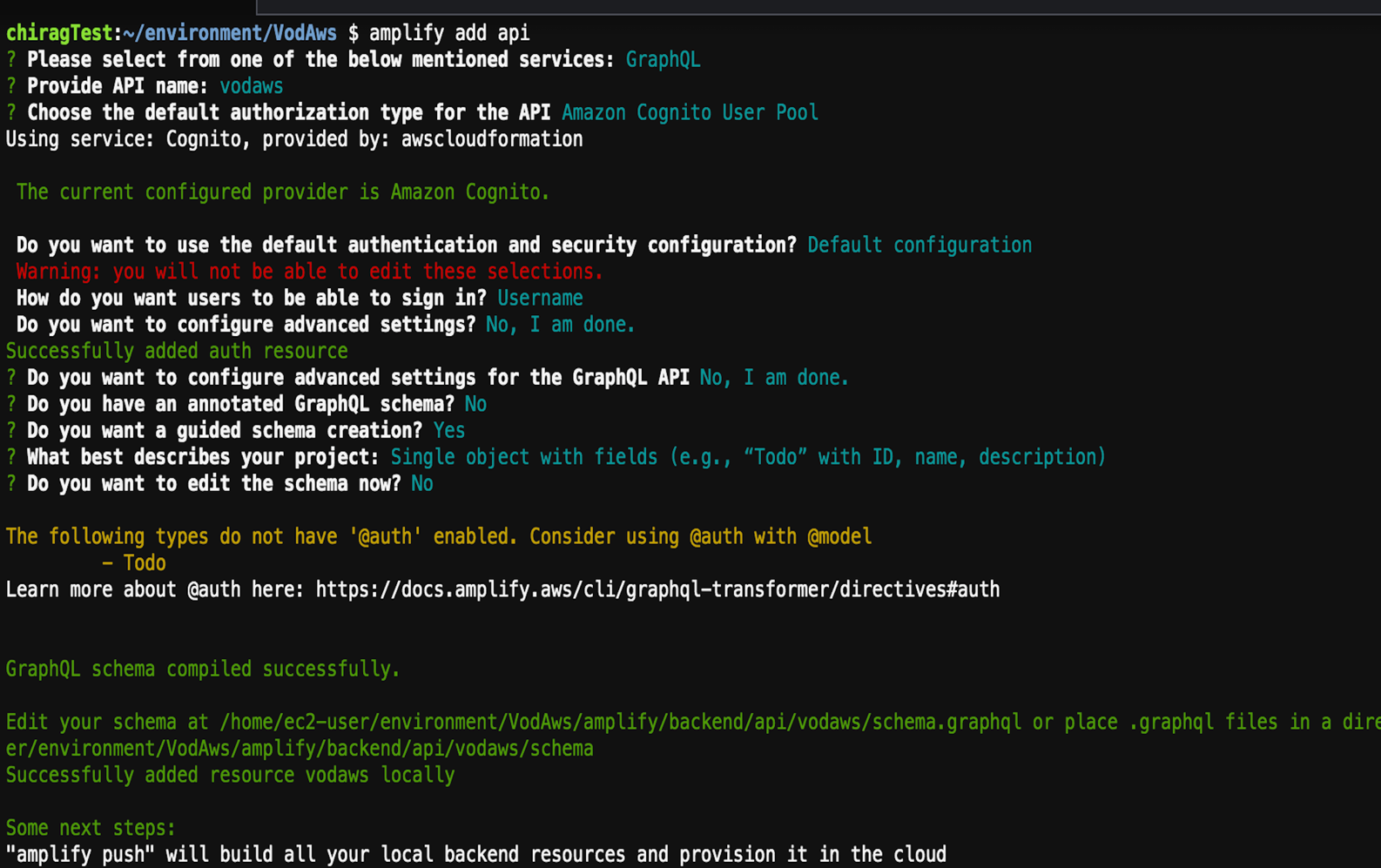
**Step3: Deploying Backend using Amplify Cli**

1. Make sure you are in the root folder of our project which is **VodAws** directory.
2. Next we are going to begin development of our Amplify project by using the initialization command. This command creates new AWS backend resources (in this case a single S3 bucket to host your CloudFormation templates) and pulls the AWS service configurations into the app!
   * amplify init
3. Follow the prompts shown below.

Note: It is recommended to run this command from the root of your app directory



1. Now, we are going to add the amplify api module to the project. Run the below command.
   * amplify add api
2. Follow the prompts as shown below



1. Replace the below code at /home/ec2-user/environment/VodAws/amplify/backend/api/VodAws/schema.graphql.

So the new schema should like below.

type vodAsset @model (subscriptions: {level: public})

@auth(

rules: [

{allow: groups, groups:["Admin"], operations: [create, update, delete, read]},

{allow: private, operations: [read]}

]

)

{

id:ID!

title:String!

description:String!

#DO NOT EDIT

video:videoObject @connection

}

#DO NOT EDIT

type videoObject @model

@auth(

rules: [

{allow: groups, groups:["Admin"], operations: [create, update, delete, read]},

{allow: private, operations: [read]}

]

)

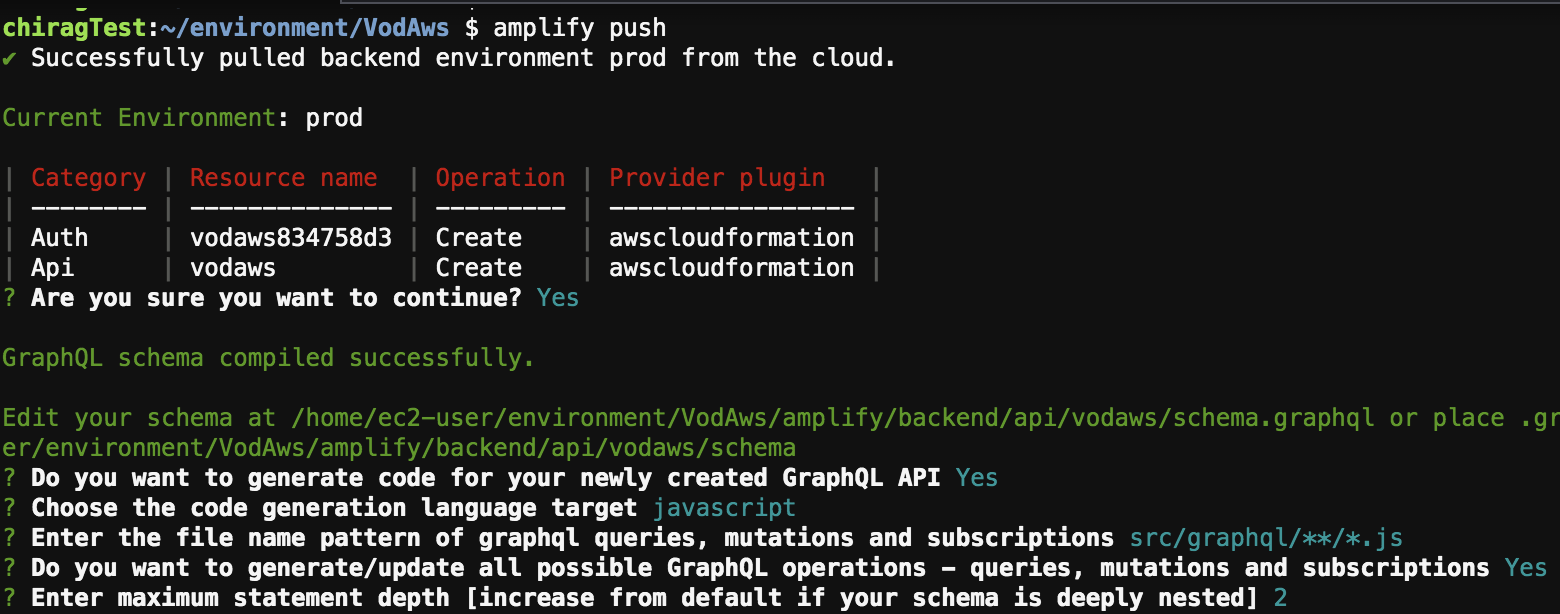
{

id:ID!

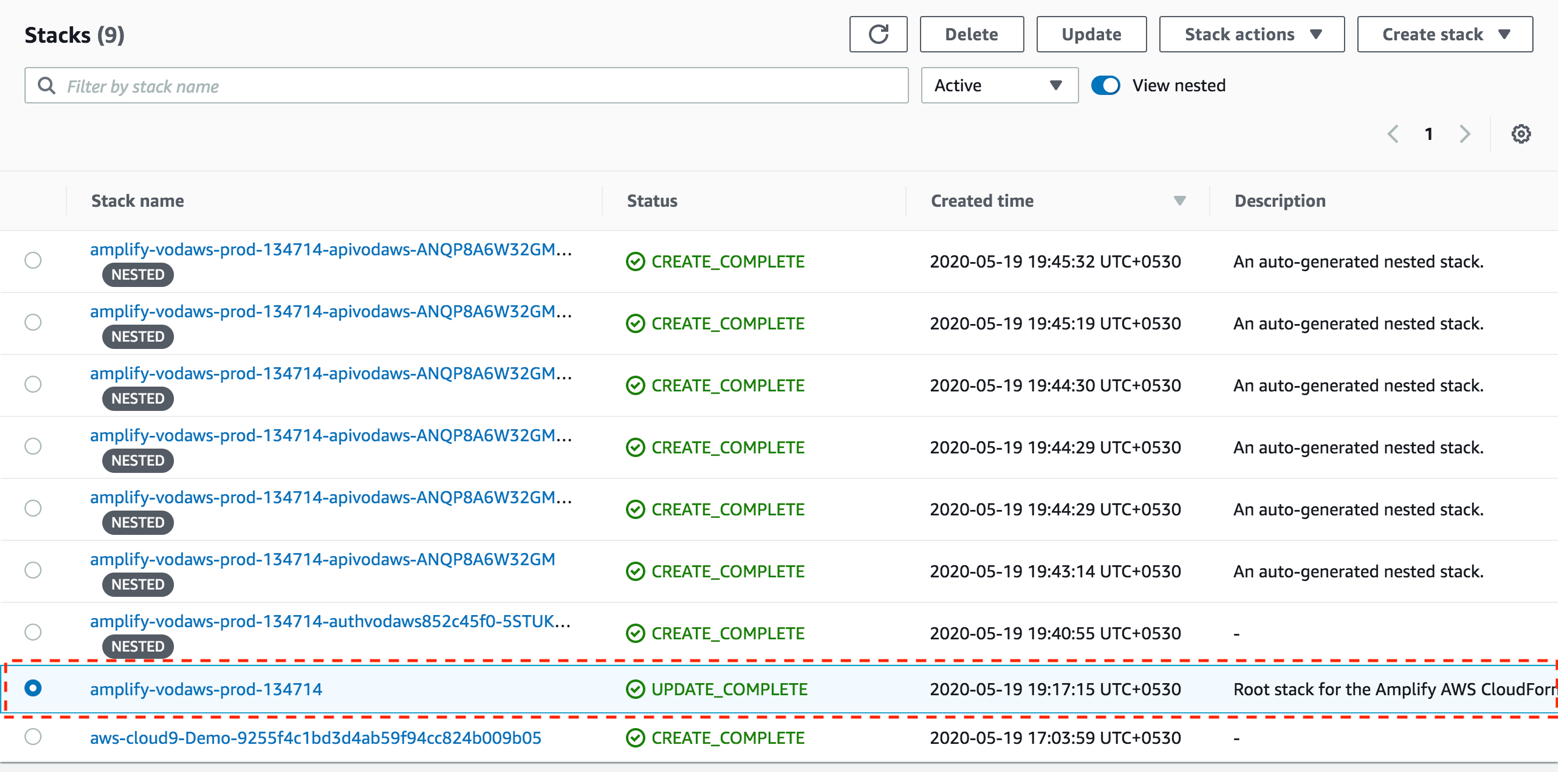
token: String @function(name: "VodAws-prod-tokenGen")

}

1. Run the below command to spin up resources in your aws environment
   * amplify push
2. Follow the below prompts:

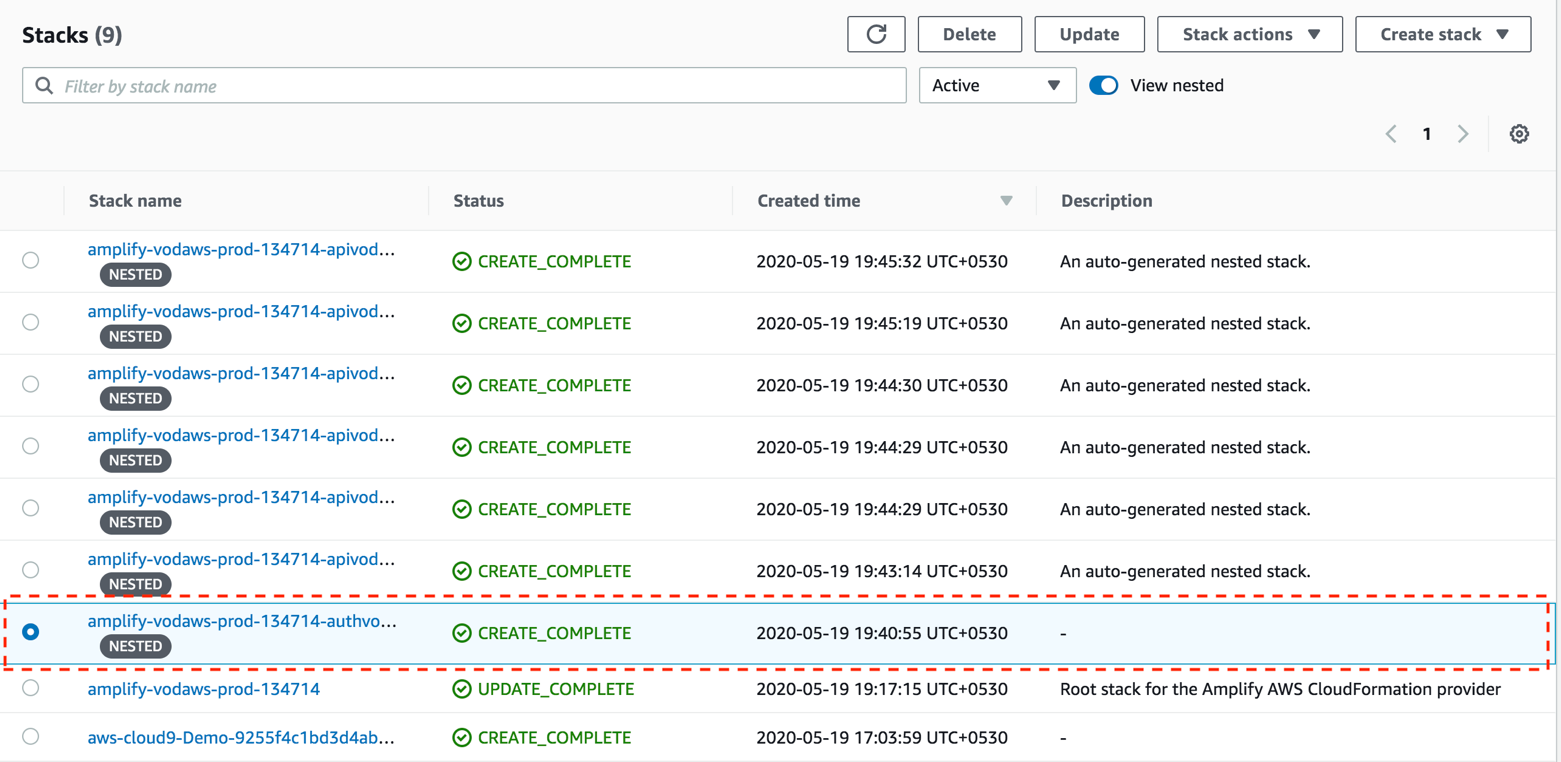
****

1. Navigate to AWS Console > go to the cloudformation > > Click on cloudformation stack which would be named similar to “*amplify-vodaws-prod-\*”* like shown in below screenshot



* + Go to the output tabs and copy these values of the below variables in a notepad, we will use this further in our workshop.
  + AuthRoleArn
  + AuthRoleName
  + DeploymentBucketName
  + UnAuthRoleName
  + UnAuthRoleArn

1. Now click on the nested cloudformation stack which would be named similar to this *“amplify-vodaws-prod-\*\*\*\*\*\*-authvodaws\*\*\*\*\*\*”* as shown in below screenshot.



* + Go to the output tabs and copy these values of the below variables in a notepad, we will use this further in our workshop.
  + AppClientID
  + AppClientIDWeb
  + IdentityPoolID
  + UserPoolID

**Step4: Creating CloudFront Key Pairs:**

**Important:**

IAM users can't create CloudFront key pairs. You must log in using root credentials to create key pairs.

**To create CloudFront key pairs in the AWS Management Console**

1. Sign in to the AWS Management Console using the root credentials for an AWS account.
2. On the *account-name* menu, click **Security Credentials**.
3. Expand **CloudFront Key Pairs**.
4. Confirm that you have no more than one active key pair. You can't create a key pair if you already have two active key pairs.
5. Click **Create New Key Pair**.
6. In the **Create Key Pair** dialog box, click **Download Private Key File**.
7. In the **Opening <filename>** dialog box, accept the default value of **Save File**, and click **OK** to download and save the private key for your CloudFront key pair.

**Important**

Save the private key for your CloudFront key pair in a secure location, and set permissions on the file so that only the desired administrator users can read it. If someone gets your private key, they can generate valid signed URLs and signed cookies and download your content. You cannot get the private key again, so if you lose or delete it, you must create a new CloudFront key pair.

1. Record the key pair ID for your key pair in a notepad. (In the AWS Management Console, this is called the access key ID.) You'll use it when you create signed URLs or signed cookies.

For more information on generating CloudFront key pairs please follow this document: <https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/private-content-trusted-signers.html#private-content-creating-cloudfront-key-pairs>

**Step5: Storing CloudFront Key Pairs:**

1. We need to store the private key file on our Cloud9 ec2 instance.
   1. Copy the entire content of private key file you just downloaded.
      1. To copy, you can open the file in your terminal/command prompt and copy the entire content of the file including,

-----BEGIN RSA PRIVATE KEY-----

Your Private Key here

-----END RSA PRIVATE KEY-----

* 1. On Cloud9 ec2 instance, create a new file called “privateKey.pem” under /home/ec2-user/environment/ directory and paste the private key you copied in the earlier step.

**Step6: Storing Private key using AWS SecretsManager:**

1. Run the following command to store the secret key in secrets manager. Make sure your under “ /home/ec2-user/environment“ directory.
   1. aws **secrets**manager create-secret --name VodAws --secret-binary <file://privateKey.pem>
2. Copy the values of Secret Key Name and ARN value from the response of the above command in a notepad.

**Step 7: Verify the values of all the variables:**

1. Make sure you have the values of the below variables in your notepad.
   1. pAppClientID > Get from Step 3, point no 10
   2. pAppClientIDWeb > Get from Step 3, point no 10
   3. pAuthRoleArn > Get from Step 3, point no 9
   4. pIdentityPoolID > Get from Step 3, point no 10
   5. pPemID > CloudFront access key > Get from Step 4, point no 8
   6. pS3 > DeploymentBuckeName > Get from Step 3, point no 9
   7. pSecretPem > Secret Key Name > Get from Step 6, point no 2
   8. pSecretPemArn > Secret Key Arn > Get from Step 6, point no 2
   9. pSourceFolder > keep it default > vod-helpers
   10. pUnAuthRoleArn > Get from Step 3, point no 9
   11. pUserPoolID > Get from Step 3, point no 10
   12. pAuthRoleName > Get from Step 3, point no 9
   13. penv > Keep it default > prod
   14. pUnAuthRoleName > Get from Step 3, point no 9

**Step8: Deploying CloudFormation Template:**

1. [Download this zip file](https://testorlando1.s3.amazonaws.com/vod-helpers.zip) on your local system and extract the contents.
2. Upload the VodAWS folder to the s3 bucket named similar to “amplify-vodaws-prod-12034-deployment” which is obtained from Step 3; point no 9.
3. Make sure you have “vod-helpers” folder in your s3 bucket and then make it public by selecting the folder and clicking on Actions drop down button.
4. Click on the below button to launch cloudformation template.
5. [](https://console.aws.amazon.com/cloudformation/home?region=us-east-1#/stacks/create/review?templateURL=https:%2F%2Fs3.amazonaws.com%2Ftestorlando1%2FParentStack.json&stackName=VodAwsStack)
6. Before deploying this cloudformation template make sure you are in the same region where amplify cloudformation template was deployed. You can change the region name from the top of aws console.
7. Enter the value for all the parameter name.
   1. pAppClientID > Obtained from Step 3, point no 10
   2. pAppClientIDWeb > Obtained from Step 3, point no 10
   3. pAuthRoleArn > Obtained from Step 3, point no 9
   4. pIdentityPoolID > Obtained from Step 3, point no 10
   5. pPemID > CloudFront access key > Obtained from Step 4, point no 8
   6. pS3 > DeploymentBuckeName > Obtained from Step 3, point no 9
   7. pSecretPem > Secret Key Name > Obtained from Step 6, point no 2
   8. pSecretPemArn > Secret Key Arn > Obtained from Step 6, point no 2
   9. pSourceFolder > keep it default > vod-helpers
   10. pUnAuthRoleArn > Obtained from Step 3, point no 9
   11. pUserPoolID > Obtained from Step 3, point no 10
   12. pAuthRoleName > Obtained from Step 3, point no 9
   13. penv > Keep it default > prod
   14. pUnAuthRoleName > Obtained from Step 3, point no 9
8. After you enter all the values on cloudformation stack, scroll down and acknowledge both the check boxes.
9. Click Create Stack.
10. After create complete, navigate to “VodAwsStack”, then navigate to outputs tab and copy the following values in a notepad.
    1. oVODInputS3
    2. oVODOutputS3
    3. oVODOutputUrl
11. Now go back to your Cloud9 IDE console. Navigate to the “src” directory
    1. cd /home/ec2-user/environment/VodAws/src/
12. Create a file named “aws-video-exports.js” and paste the below code

const awsvideoconfig = {

"awsInputVideo": "Your Input S3 Bucket Name",

"awsOutputVideo": "Your Cloudfront Url"

};

export default awsvideoconfig;

1. Enter S3 bucket name obtained from above step as “oVODInputS3” value and Cloudfont url as “oVODOutputUrl” value
2. Navigate back to your project directory > /home/ec2-user/environment/VodAws/
3. Run npm start

**Step9: Hosting our app on Amplify**

Now let’s go back to Cloud9 IDE environment and navigate to /home/ec2-user/environment/VodAws/

1. To host our app on amplify, run the below commands
   1. amplify hosting add
2. Follow the prompt below

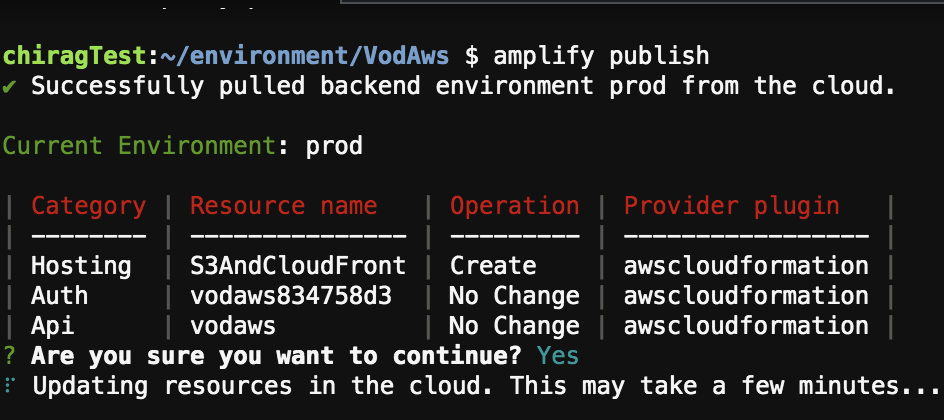
Select the plugin module to execute **Amazon CloudFront and S3**

? Select the environment setup: **PROD (S3 with CloudFront using HTTPS)**

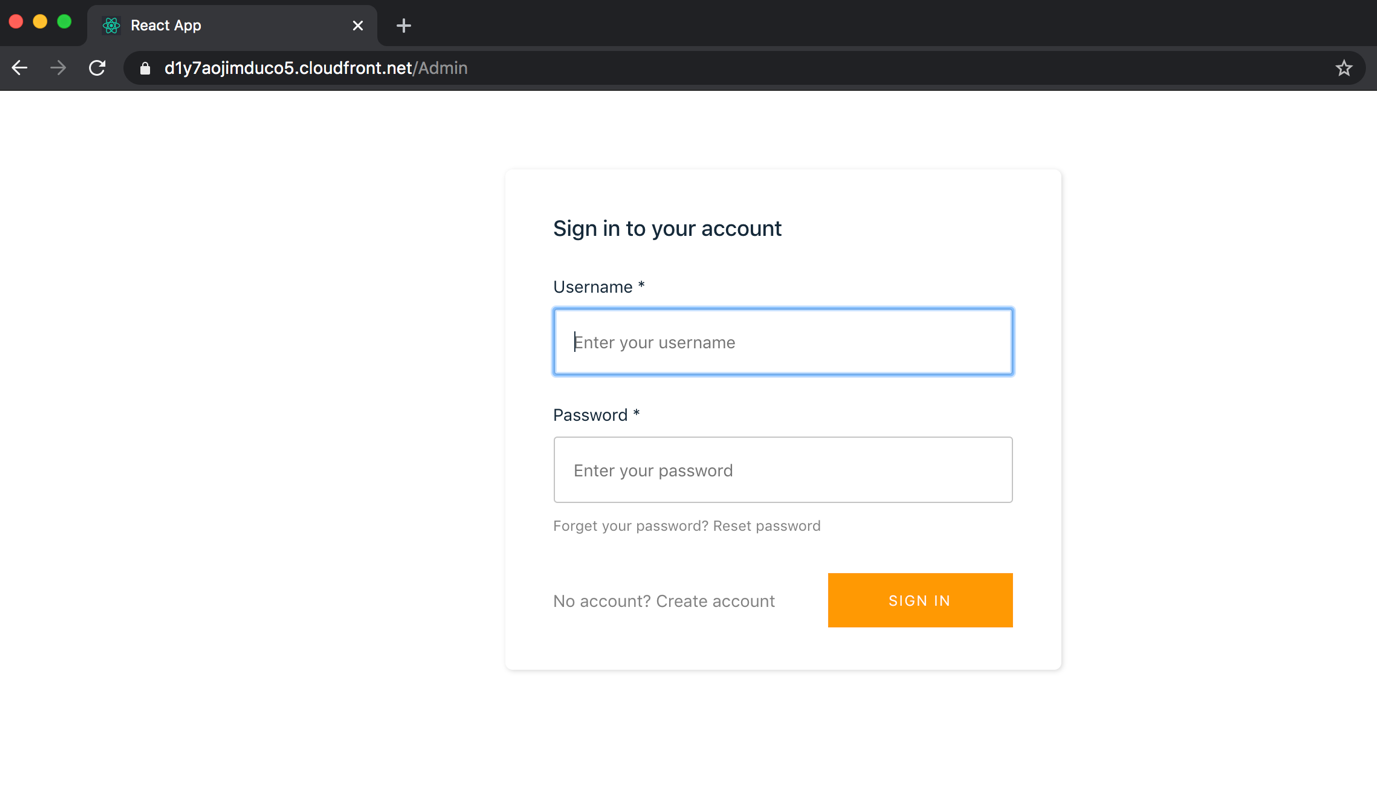
? Accept the propose hosting bucket name that looks like **odws-20200518075723-hostingbucket**

Static webhosting is disabled for the hosting bucket when CloudFront Distribution is enabled.

1. Run the below command to spin aws resources
   1. amplify publish



1. Make a note of the cloudfront url.
2. Start node server by running below command
   1. Npm start
3. Go to the browser and enter cloudfront-url/Admin to upload your videos as shown in below image.



**Step 10: Create Users in Cognito User Pool**

1. Now we need an Admin user to test out the authentication functionality, let's create an admin user through the Cognito console.
2. Open the AWS Management Console and Search for Cognito.
3. Select the blue "Manage User Pools" button
4. Select the userpool labeled "VodAws" or your project name + a random string.
5. Under General Settings, choose "Users and Groups"
6. Select the blue "create user" button and enter the user creation form. You will need to enter a phone number that includes country code, for example +15558888888
7. Fill out the form to create a user. Now we will have to add admin privileges in order to enable this user to publish videos through the app.
8. Select the user you juts created
9. Select the blue "Add to Group" button, and select the admin group.
10. Go back to the react application (localhost:3000/Admin) and log in and create a new password.
11. Now you can upload a sample “.mp4” video.
12. Create a user account using the app sign-up page instead of the Cognito console. You may have to sign out if you're still logged in as the admin user.
13. For user-acccount the url will be > “Cloudfronturl.net”
14. Refresh the tab that the application is running in to see the login page. (react's local dev server may do this for you)
15. Create a new user. This user will not be an admin and thus won't have rights to publish content to VodAws. Make sure to provide a valid email to activate your account. The code may take a minute or two to arrive in your inbox.
16. Login to the react application from the newly created user and you will be able to watch the video uploaded by admin.

**Troubleshoot:**

**CloudFront CORS Issue**:

To forward the headers from your CloudFront distribution, follow these steps:

1. Open your distribution from the [CloudFront console](https://console.aws.amazon.com/cloudfront/).
2. Choose the Behaviors tab.
3. Choose Create Behavior, or choose an existing behavior, and then choose Edit.
4. For Cache Based on Selected Request Headers, choose Whitelist.
5. Under Whitelist Headers, choose headers from the menu on the left, and then choose Add.
6. Add the following headers : Access-Control-Request-Headers, Access-Control-Request-Method, Origin.
7. Choose Yes, Edit.

**CloudFront Distribution propogation**:

If you have deployed Cloudfront in another AWS region other than us-east-1 region then it will take some time to propogate changes to other region.

Note that if the hosting S3 bucket is newly created in regions other than us-east-1, you might get the HTTP 307 Temporary Redirect error in the beginning when you access your published application through CloudFront. This is because CloudFront forwards requests to the default S3 endpoint ([s3.amazonaws.com](http://s3.amazonaws.com)), which is in the us-east-1 region, and it can take up to 24 hours for the new hosting bucket name to propagate globally.

For more information please follow this document: <https://aws.amazon.com/premiumsupport/knowledge-center/no-access-control-allow-origin-error/>