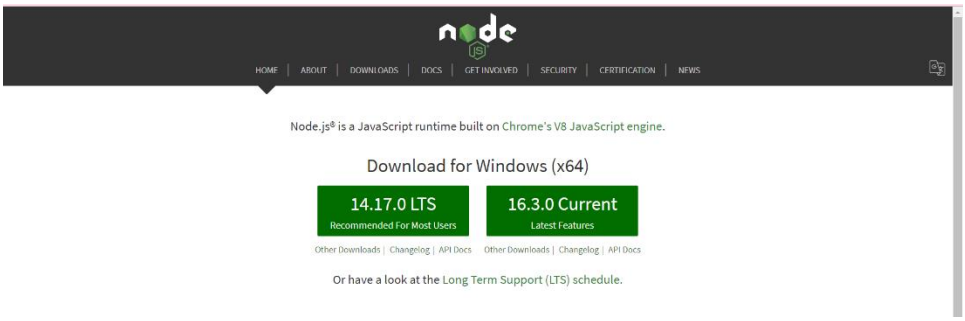
## g. Developer Manual

The setup can be divided into two major parts- one in the developer's machine and secondly on cloud to create AWS Resources. The developer can download the code from the Google Doc given in the code folder. There will be a folder for front end as well back end and an additional file from deploy that can be used in ECR.

**Setup on developer machine**

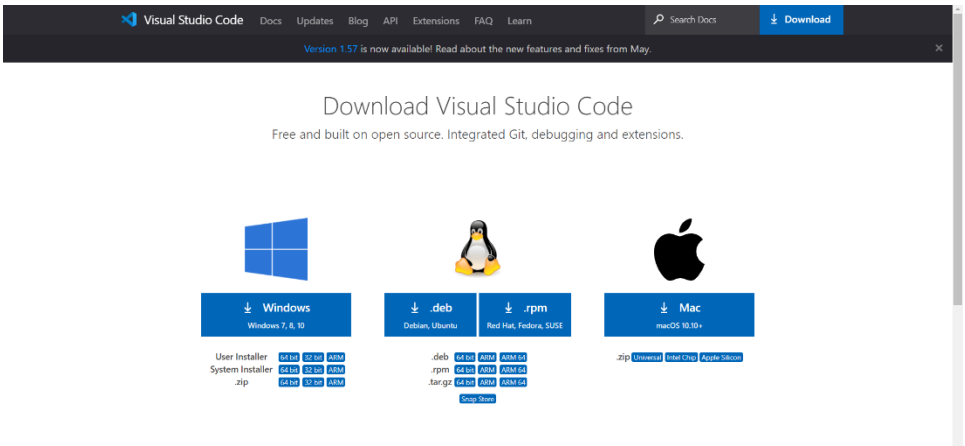
The tools required for development are given below along with the instructions.

**Node js**




- Click on Download and to verify if node js has been set up, in CMD type node -v
- The system should display the Node.js version installed on your system. You can do the same for NPM: npm -v

**IDE - Visual studio code.**

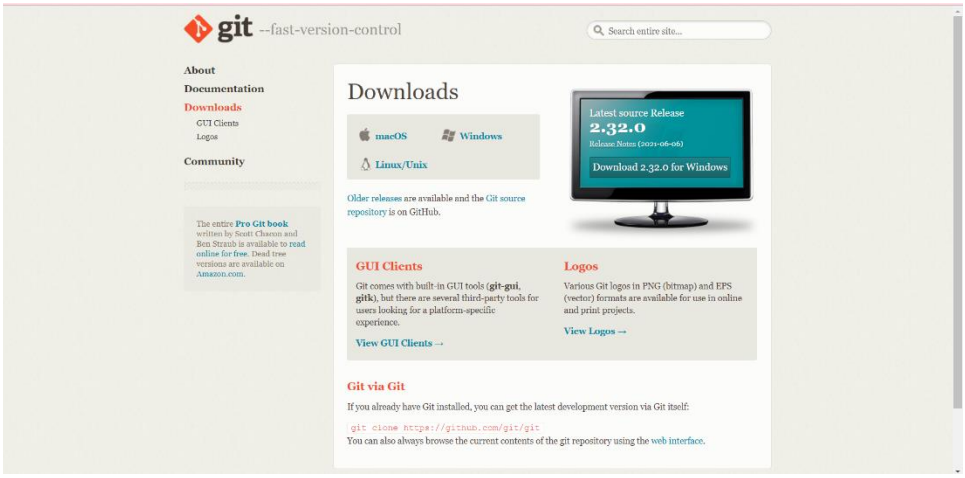



- Click on download to set up visual studio code in your system.

## React

- In the nodejs terminal setup on Vscode - npx create react-react-app

## Version control - Git



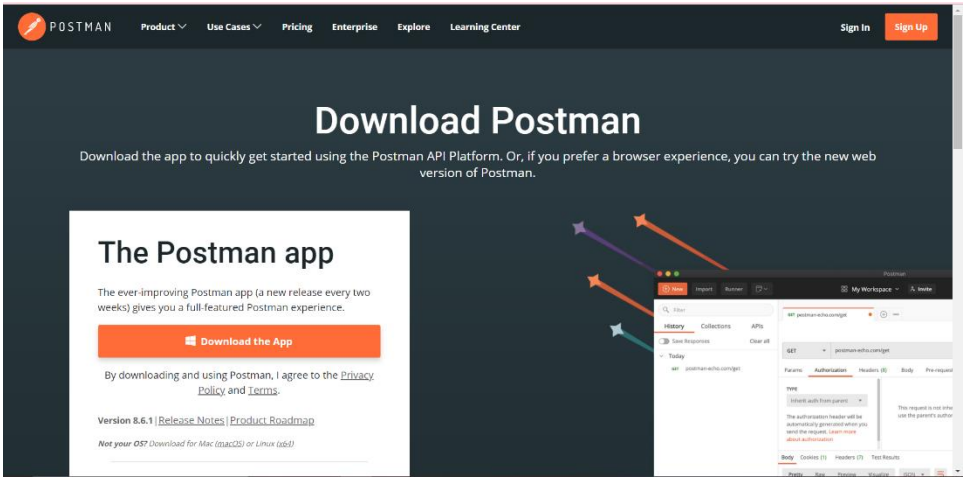

- Click on Download for Windows [14]
- Browse to the download location (or use the download shortcut in your browser). Double-click the file to extract and launch the installer.
- Follow instructions in the installer. Once it is setup and installed, launch by searching git on the windows explorer.

## Database - MongoDB Compass


- To further utilize the power of cloud, we used MongoDB atlas that is the database stored in cloud.

- To sign in go to https://account.mongodb.com/account/login
- Enter credentials

## Testing APIs – Postman

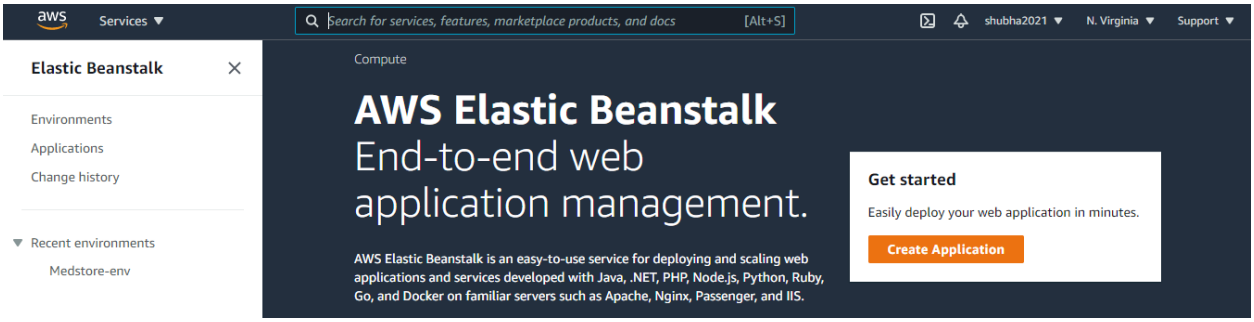



- Click on Download the App
- Double-click the exe file to install it. [13]

## Deploy and manage AWS services

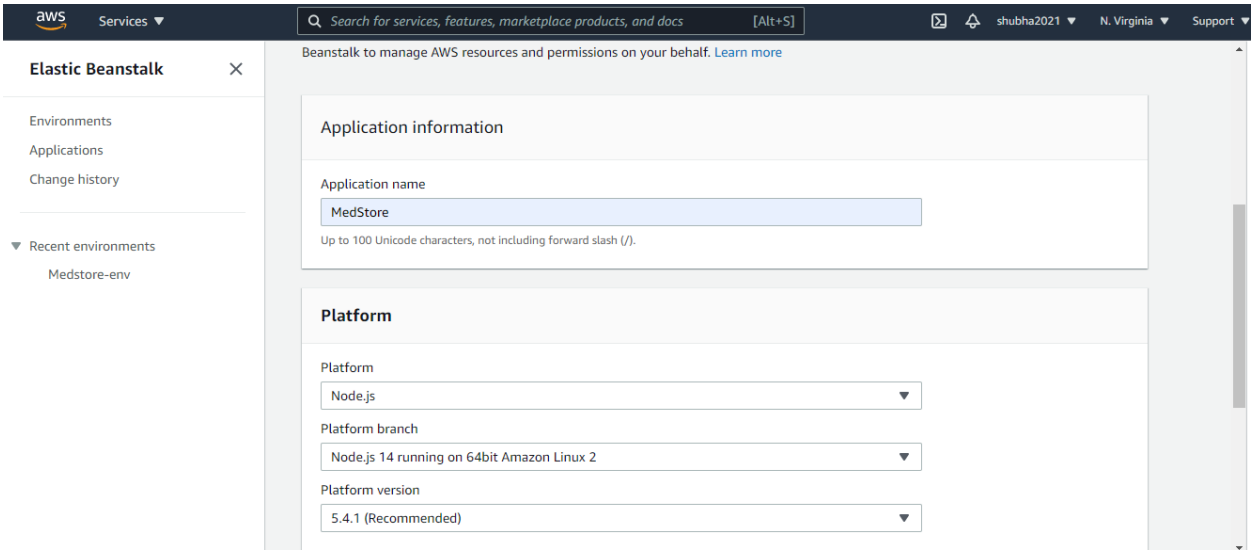
### How to use elastic beanstalk to deploy Node Js backend

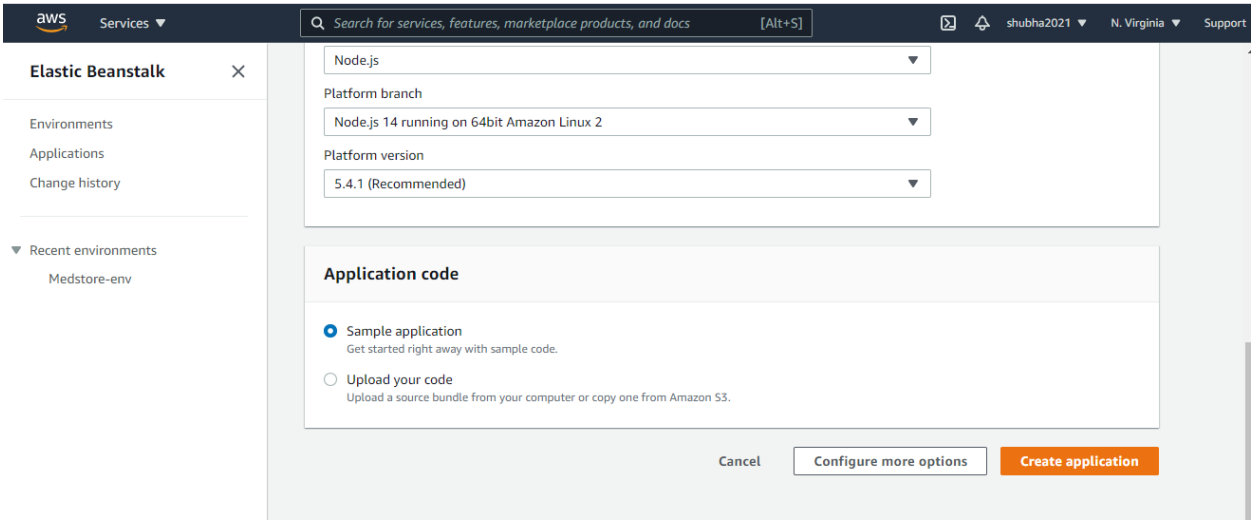
1. Create an Elastic Beanstalk environment using the guide [1]




Click create application

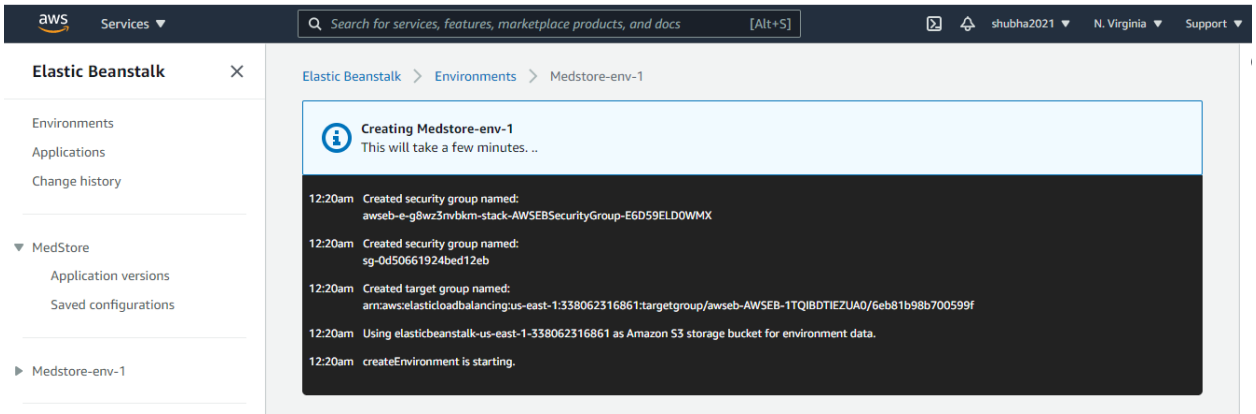
2. Follow the instructions and give details as



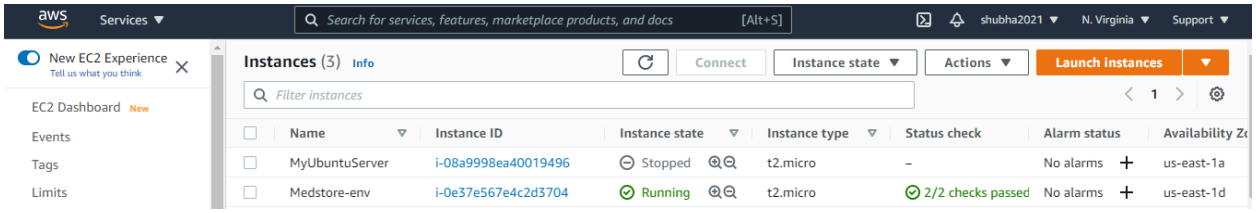




And click create application.

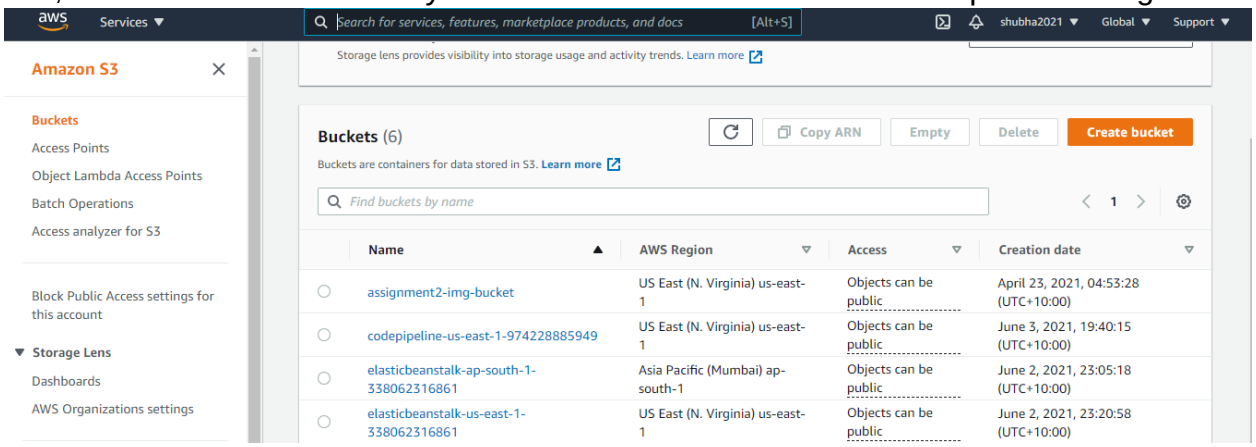
3. It will take several minutes to create.



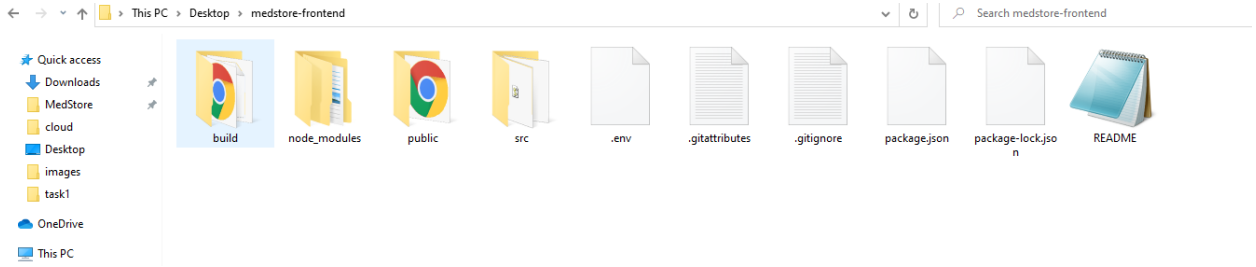
4. Once it will be done, we can see that ec2 instance has been created.



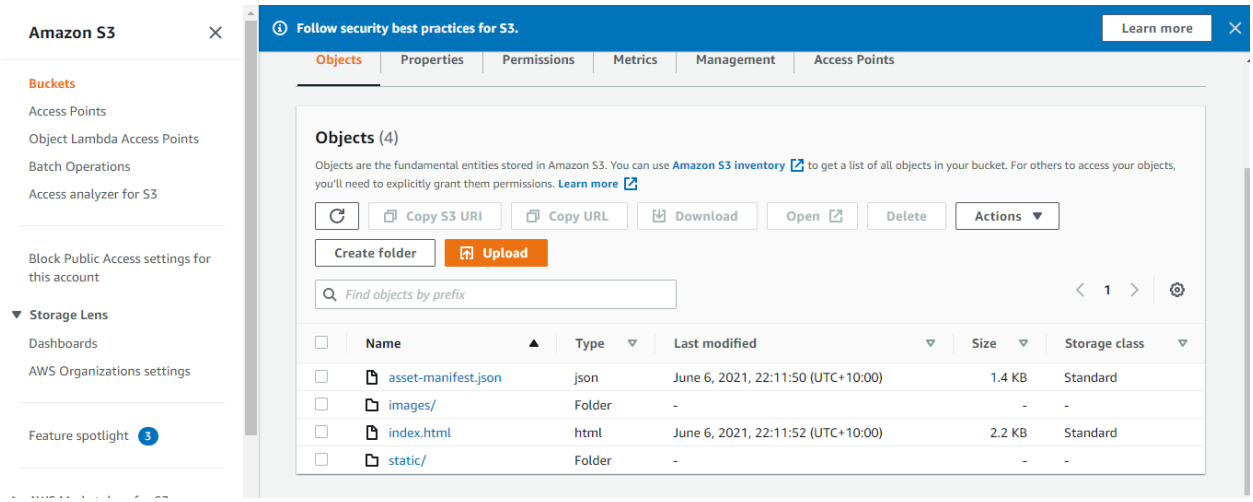

5. Also, S3 bucket is automatically created That we will use to save our product images




6. We have created another s3 bucket to upload our frontend code. Now,go to the IDE In our case we used VS code and run “npm run build” to create build folder of compressed files that we can use for the frontend deployment.



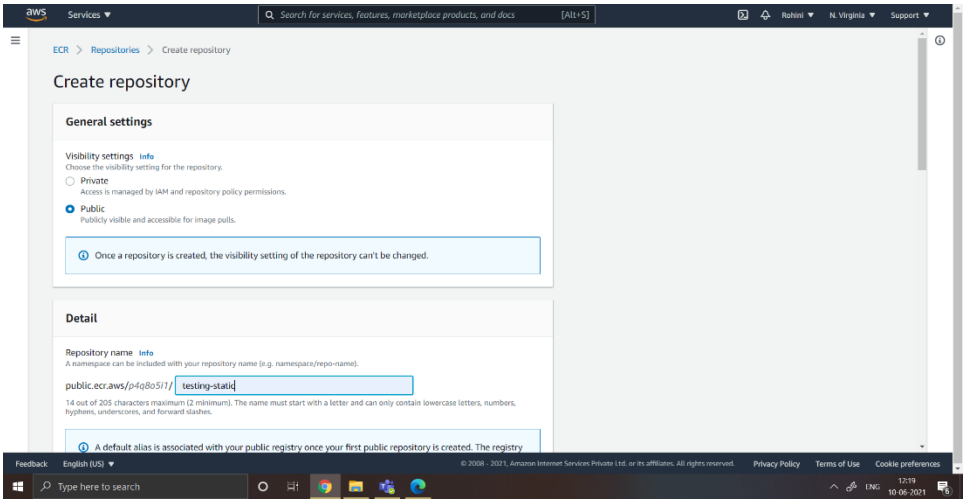

7. Use this build folder contents and go to the S3 frontend bucket and upload the code there.



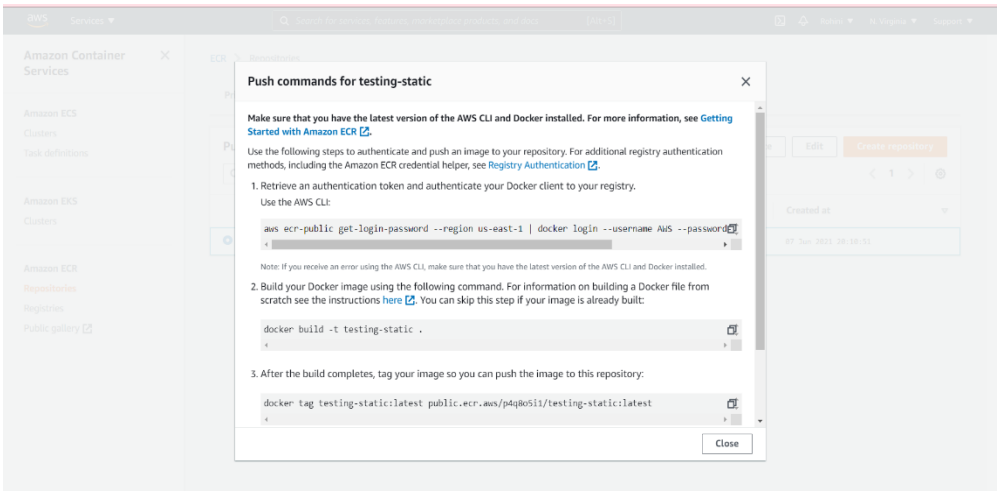

8. Once the code is uploaded Go to properties and at the end of the page, we can get a link to run on our browser.

## How to Create AmazonECR and Amazon ECS using the AWS Console

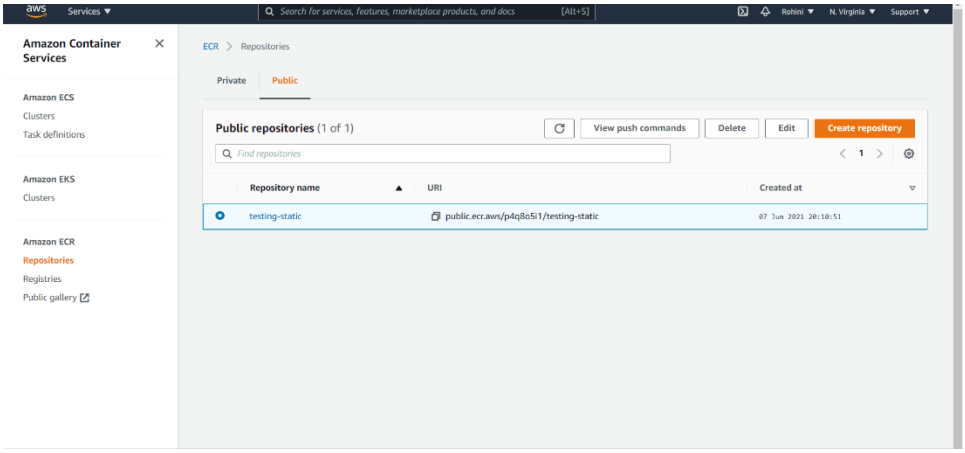
1. Connect to the AWS Console and to the ECS Administration screen to create a new repository.
2. Click on Create Repository and choose testing-static. as a name for the repository. You can add any suitable name. The ECR repository will now been created.



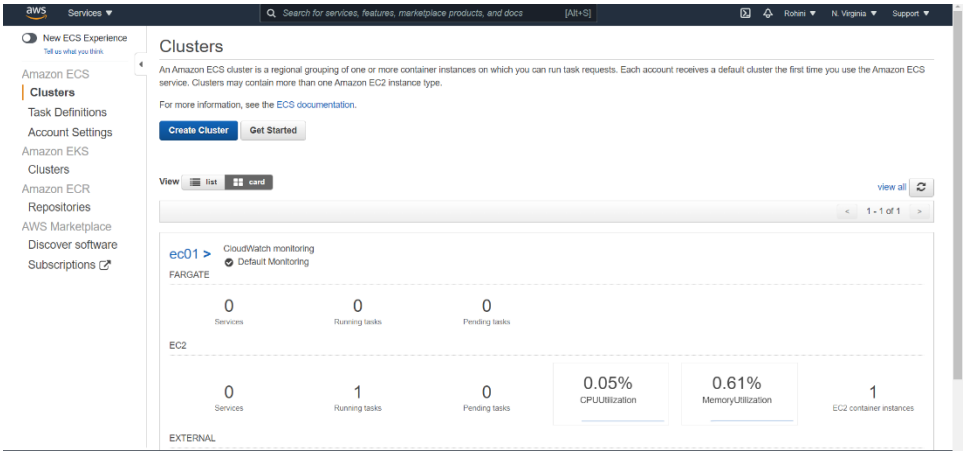

3. To Upload the docker image on AWS ECR, Click on the push commands button on the repository screen. Copy and execute each command.



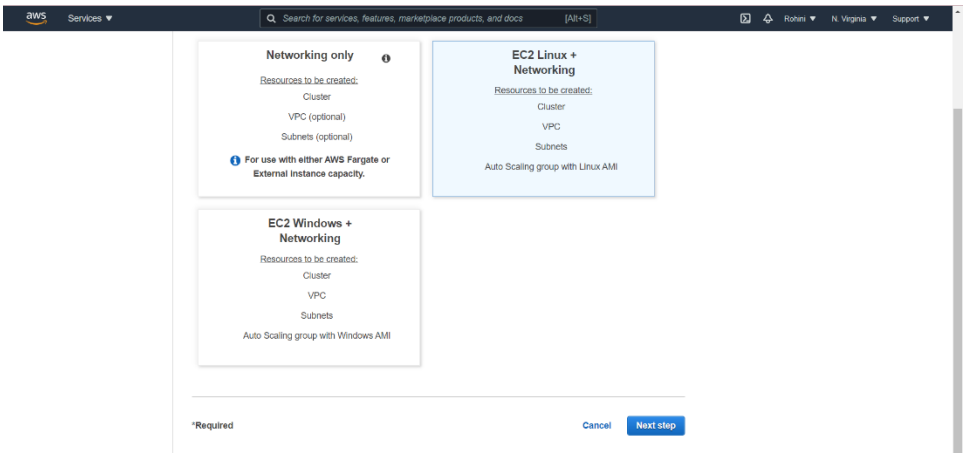

4. Copy the URI in this case public.ecr.aws/p4q8o5i1/testing-static, as we will be using it in ECS.




5. Go to the ECS home page and click on the create cluster button:



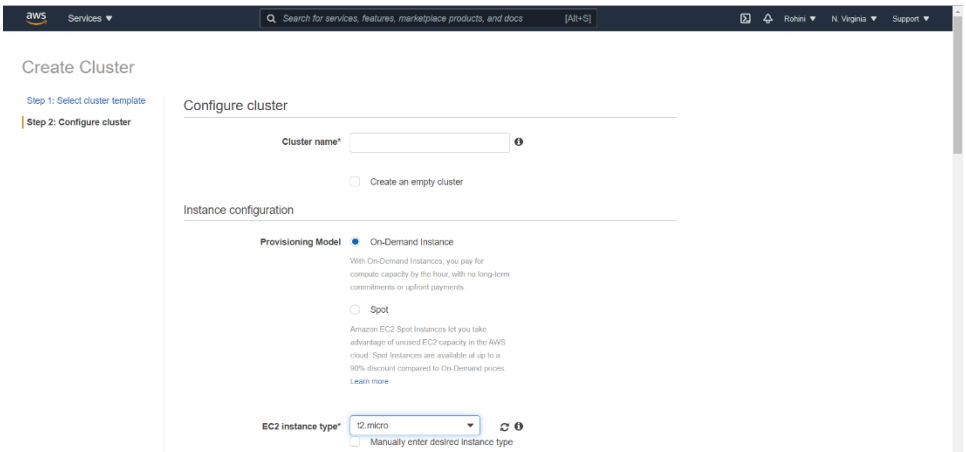

6. Choose EC2 Linux + Networking and then click next:




7. Go to the configure cluster page, then enter the following information:

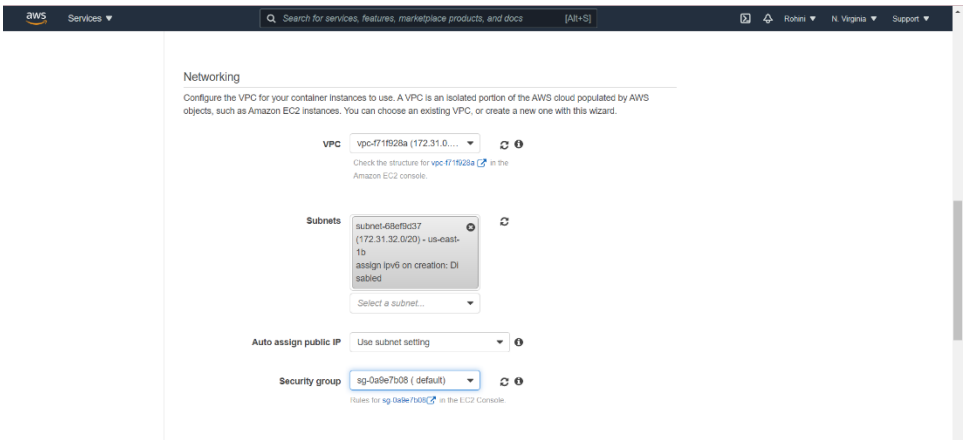
- name of the cluster: ecs01
- EC2 instance type: t2-micro
- Number of instances: 1

Select EC2 instance type as t2-micro as it is available in the free tier.



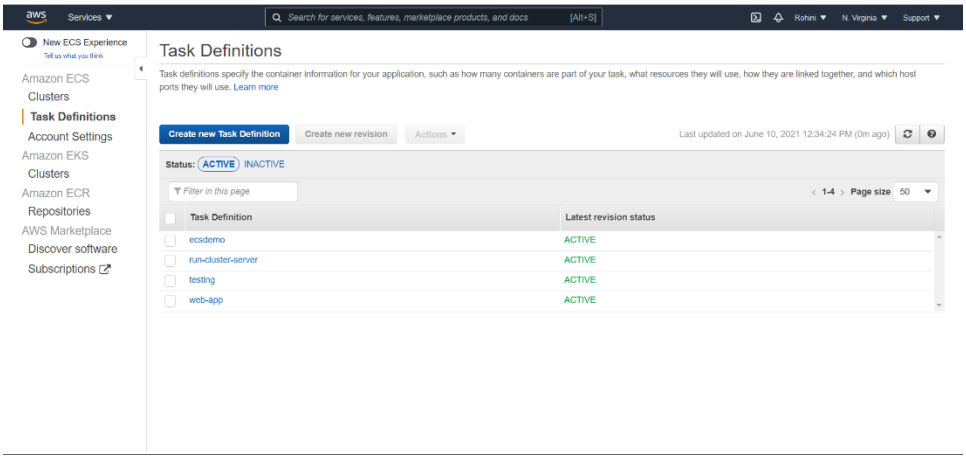

In the networking section choose:

- Default VPC
- Auto assign IP: Enabled
- Security group: default
- Choose one of the subnet.

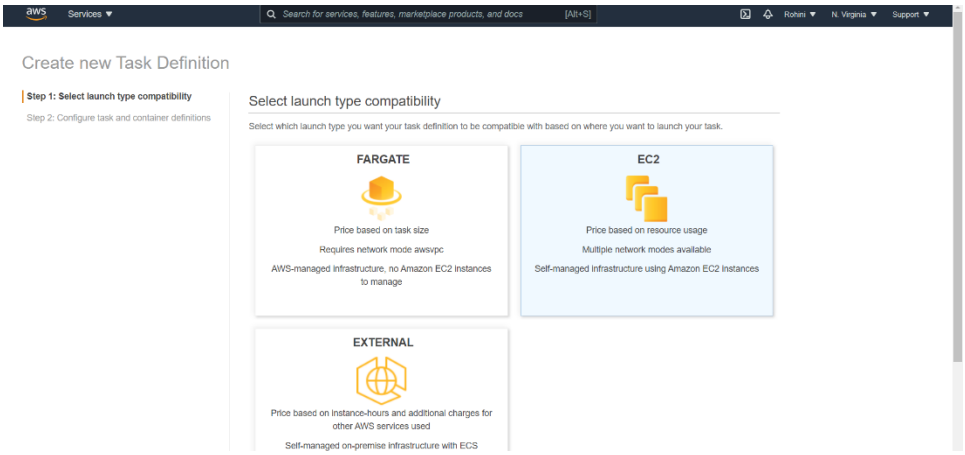



Then press Enter.

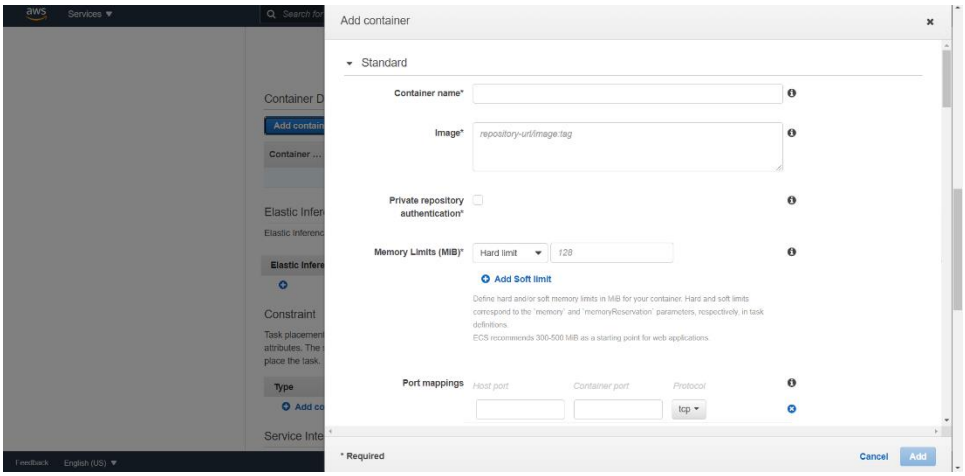
8. A task is a set of metadata (memory, CPU, port mapping, environmental variables) that describes how a container should be deployed. Click on new Task definition.




9. Choose Ec2 and click on next.

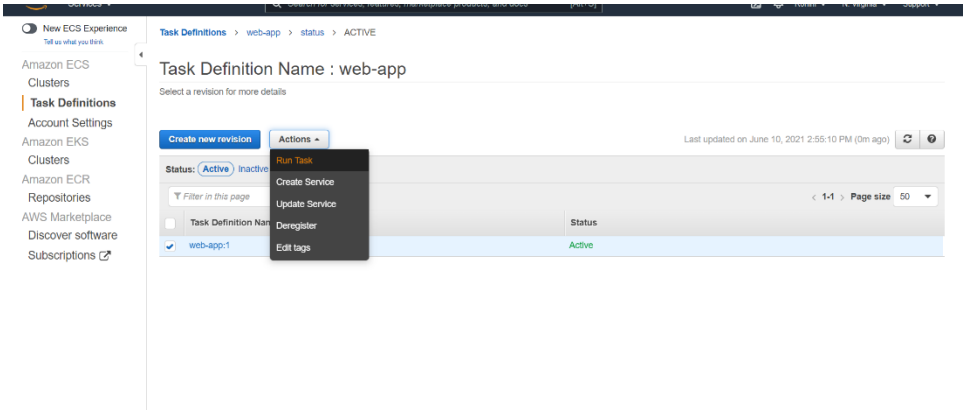



10. Choose a name and Enter 128 for memory size. Then click on Add Container.

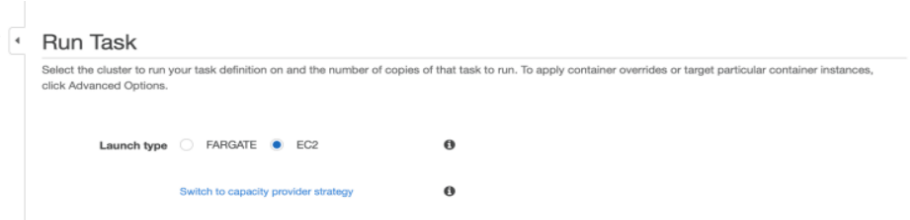



- Set the image URI that we have saved to add the end of the add image step.
- Set the port mappings 80:80 (as it is a static page, in the case of a nodejs
- web app, the port will be 80:8080)
- Click on add

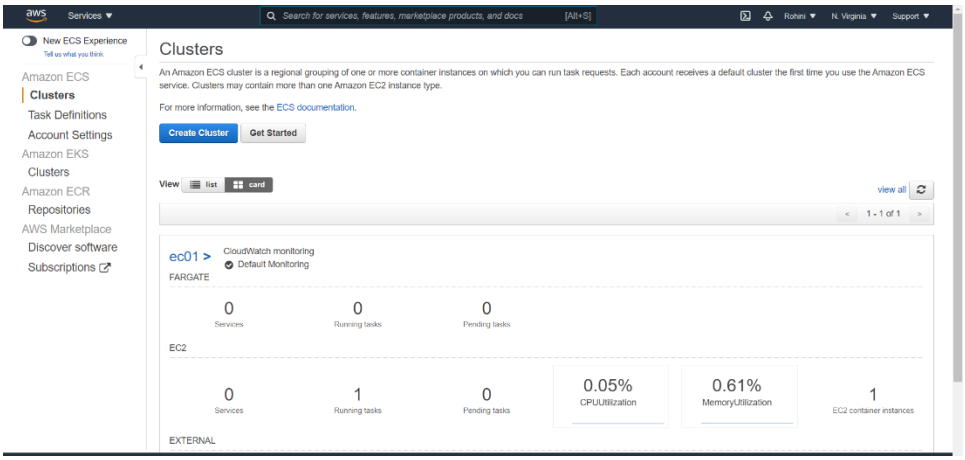
11. Once the task is created go to actions and select run task.



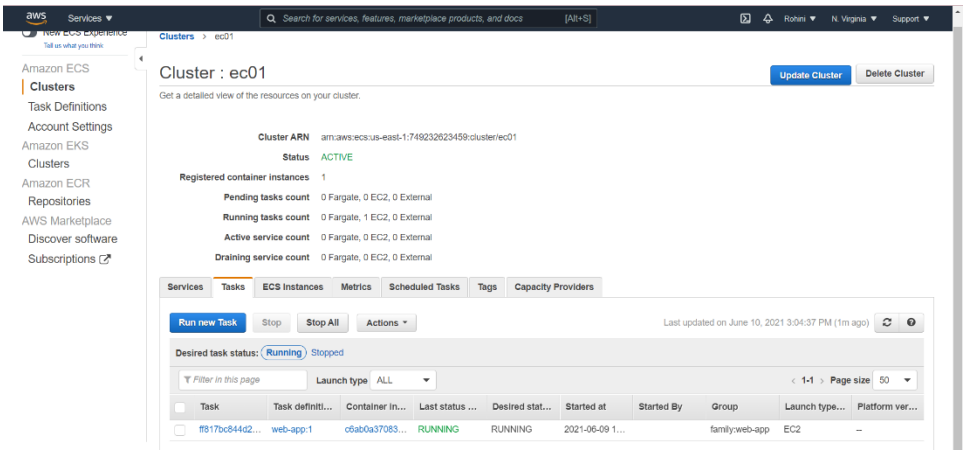
12. From run task option, select EC2, select the cluster and press enter. Other options can be left as default.

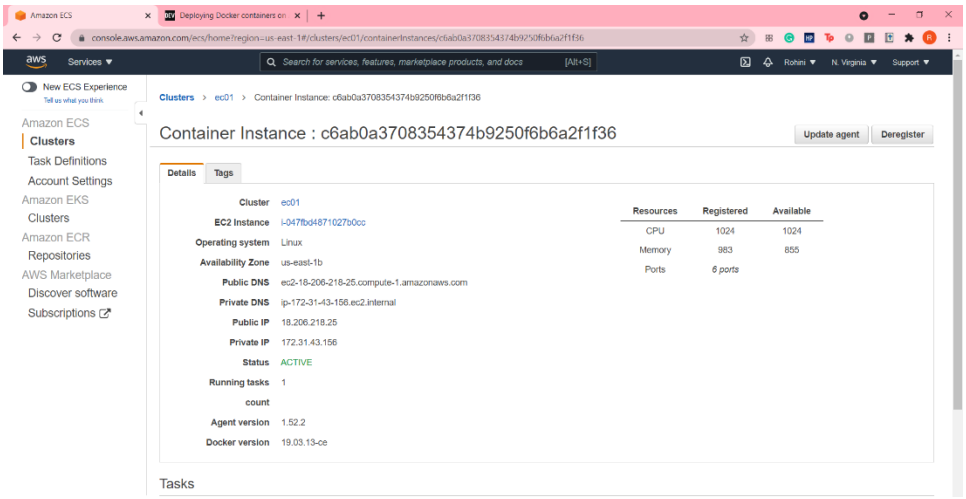



13. Go into clusters, and select the cluster we are working with which is ec01.

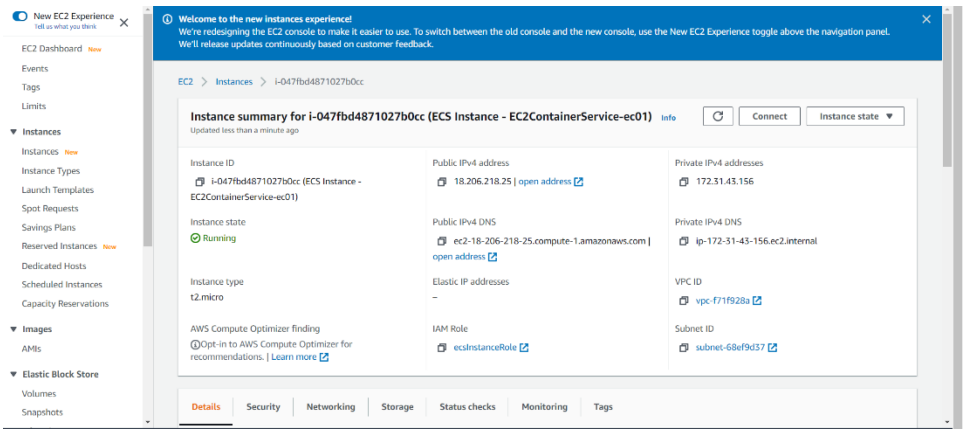



14. The task is now running. Click on the container.

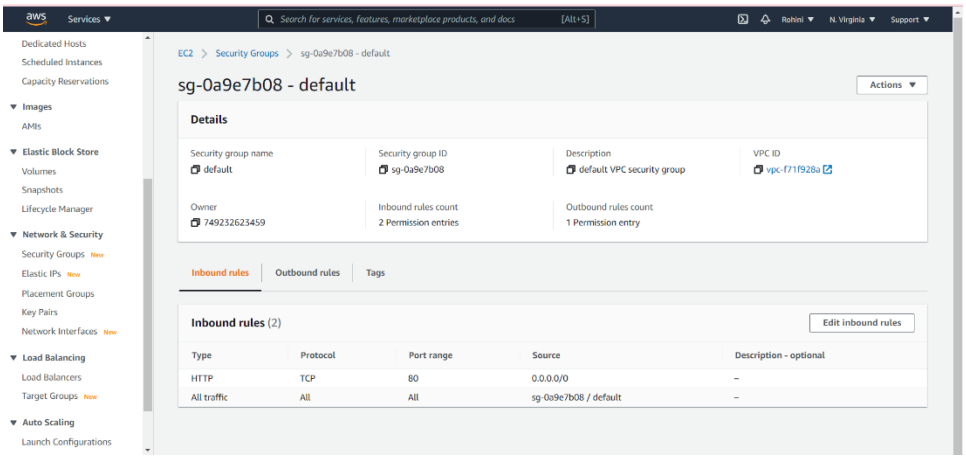




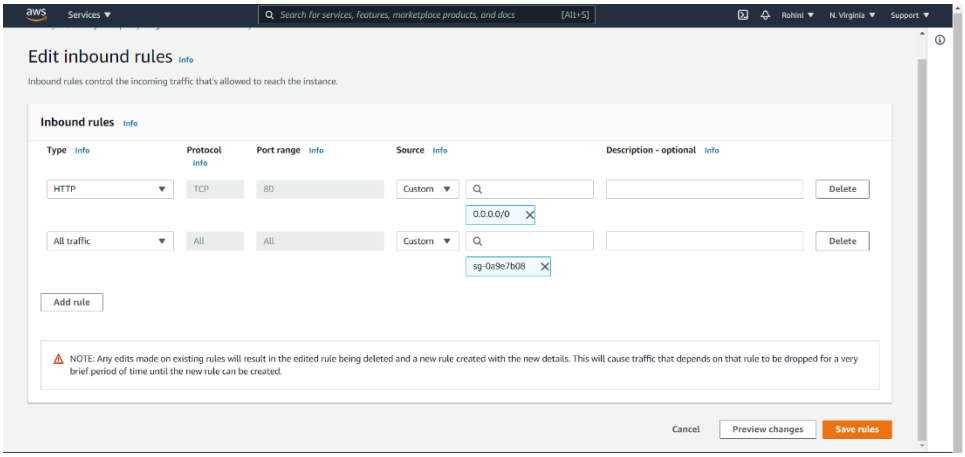

15. Go into In the EC2 panel and select the corresponding EC2 for the container service. we can modify the ports by going into Security groups on the left side of the console.



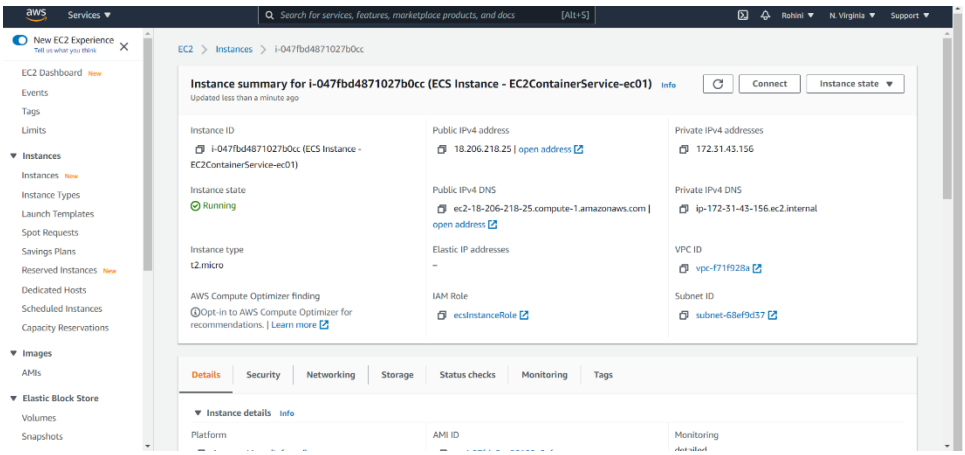

16. Navigate to the security group used. In this case, it is the default security group. Select inbound security rules.



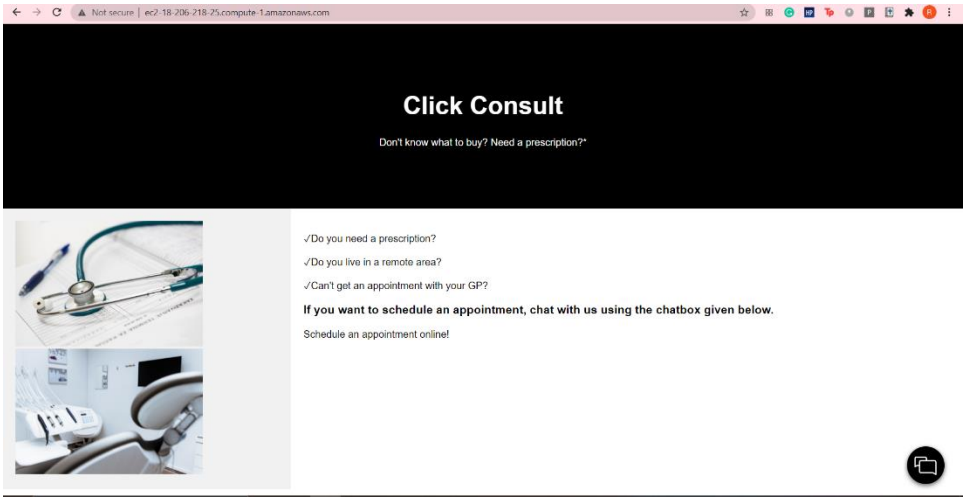

Add 80 to the inbound rule for the security group. Click on save rules.




17. Click on the public IPV4 DNS

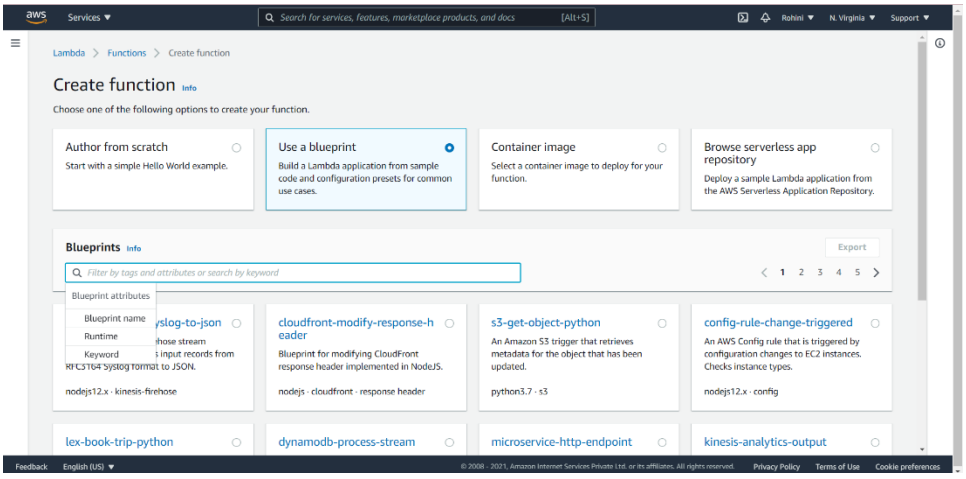



The website will now load successfully.

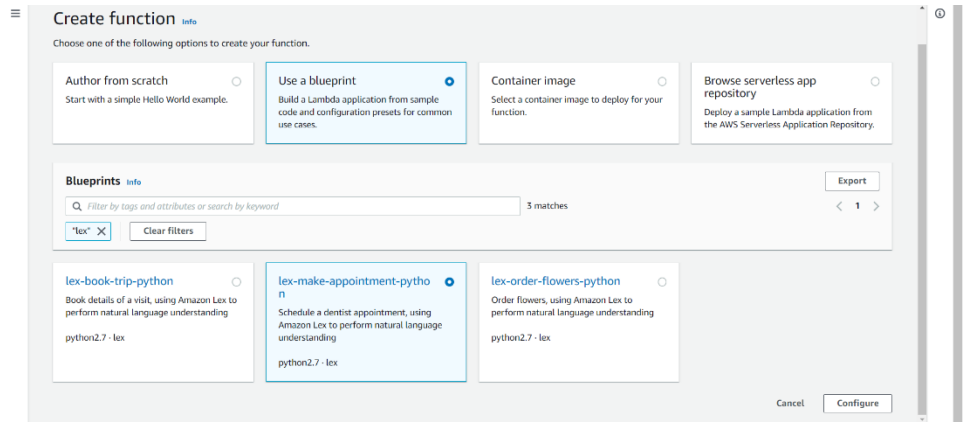


## How to create a lambda function

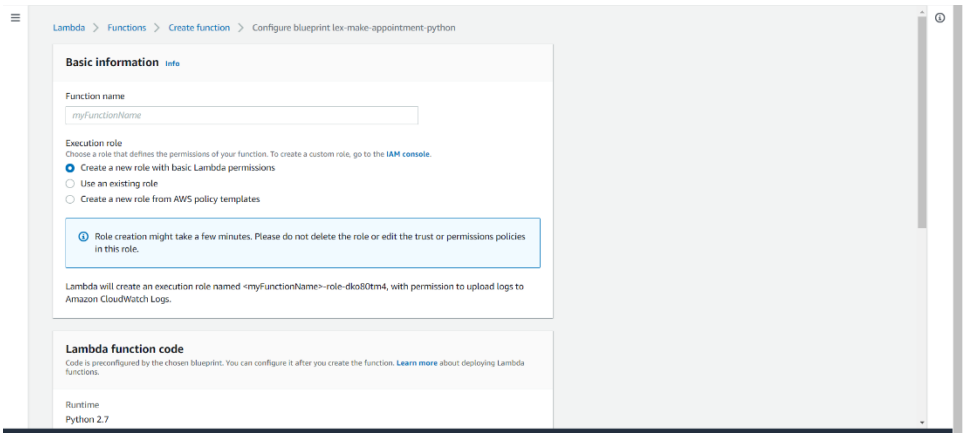
1. Sign in to the AWS Management Console and open the AWS Lambda console at https://console.aws.amazon.com/lambda/.
2. Choose **Create function**.
3. Select use a blueprint



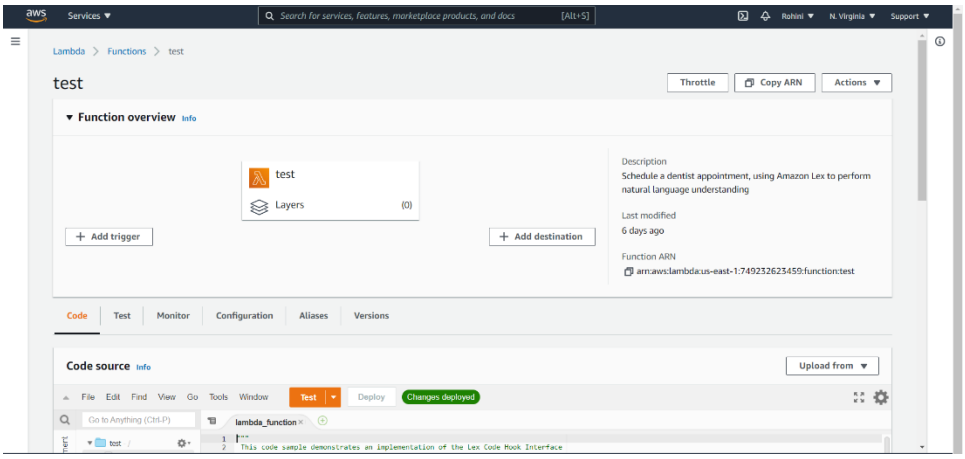

4. Search for lex and select lex-make appointment.
5. Click on configure.



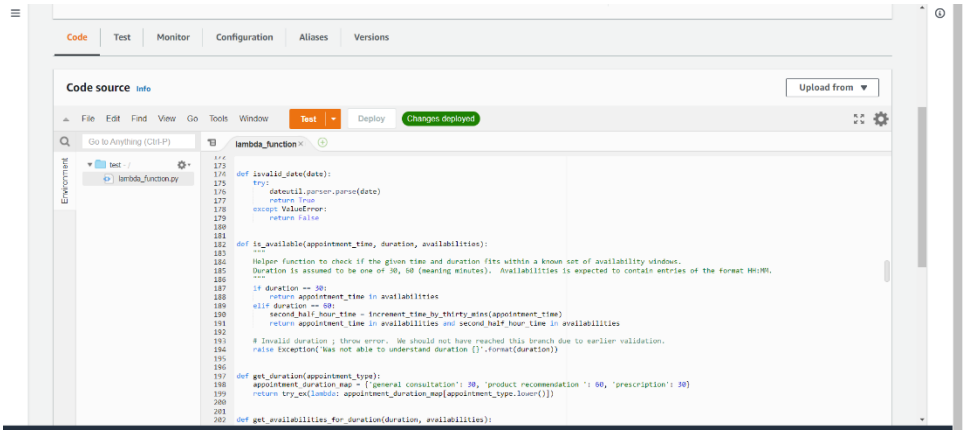

6. Provide a suitable name for the function and select create a new role




7. Click on continue. We don't have to worry about the code right now. Code is preconfigured by the chosen blueprint, and we can configure it after creating the function.
8. The function has been created and now we can go into source code to edit. We will be using python for this.



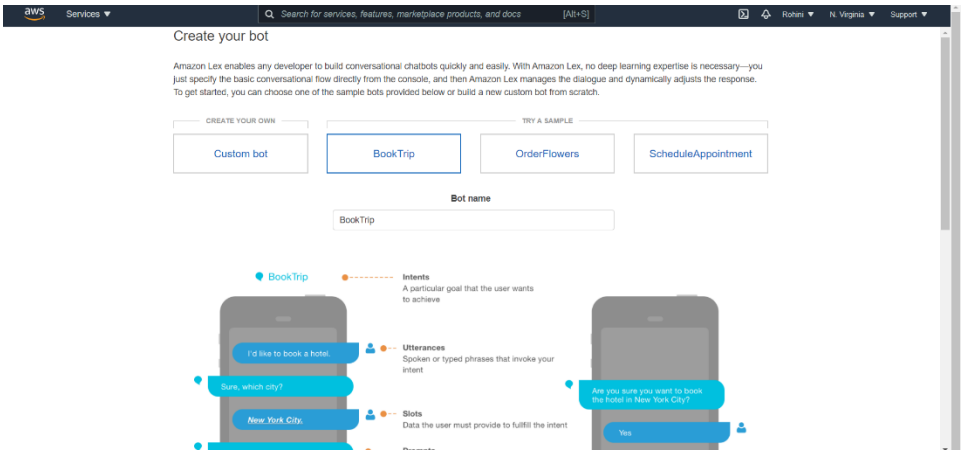

9. After editing the function. Click on Test and to try out with a test case that is pre-built or click on deploy if you are happy with the changes.



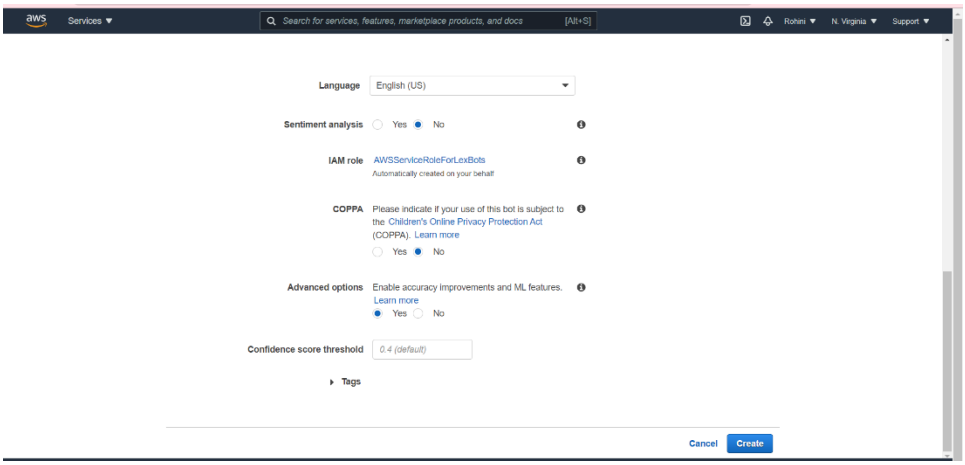
Now that the lambda function is created, we will now proceed to make the bot on Lex.

**How to create a conversational bot on Amazon Lex**

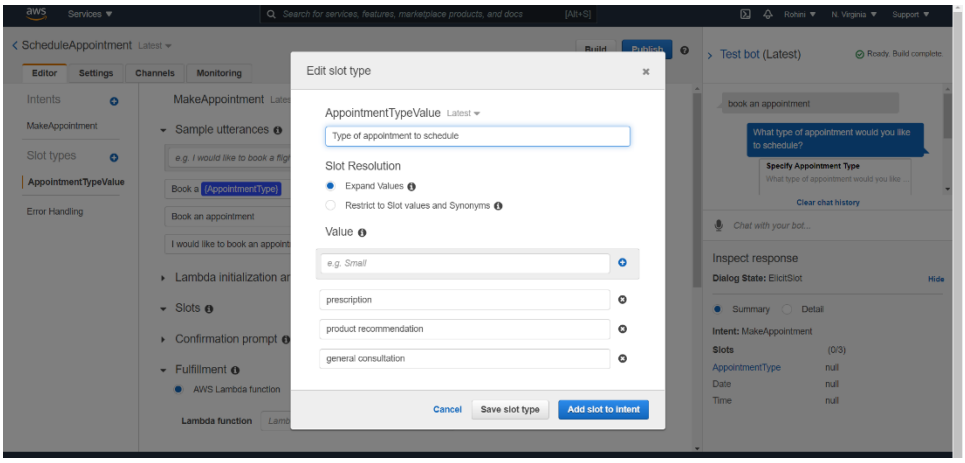
1. Sign in to the AWS Management Console and open the Amazon Lex console at https://console.aws.amazon.com/lex/.
2. If this is your first bot, choose Get Started; otherwise, on the Bots page, choose Create.
3. On the Create your Lex bot page, provide the following information, and then choose Create.
4. Choose the blueprint ScheduleAppointment



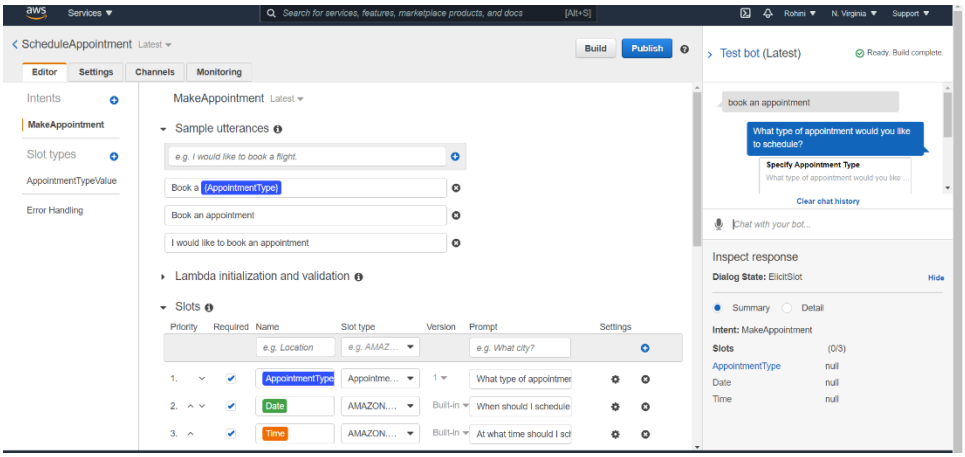

5. Leave the default bot name (ScheduleAppointment).
6. For COPPA, choose No.
7. For User utterance storage, choose the appropriate response.
8. Choose Create. The console makes the necessary requests to Amazon Lex to save the configuration. The console then displays the bot editor window.



9. Wait for confirmation that your bot was built.
10. Go into appointmenTypeValue and make changes according to requirement.




11. Test the bot by clicking on Test Bot option from the right.

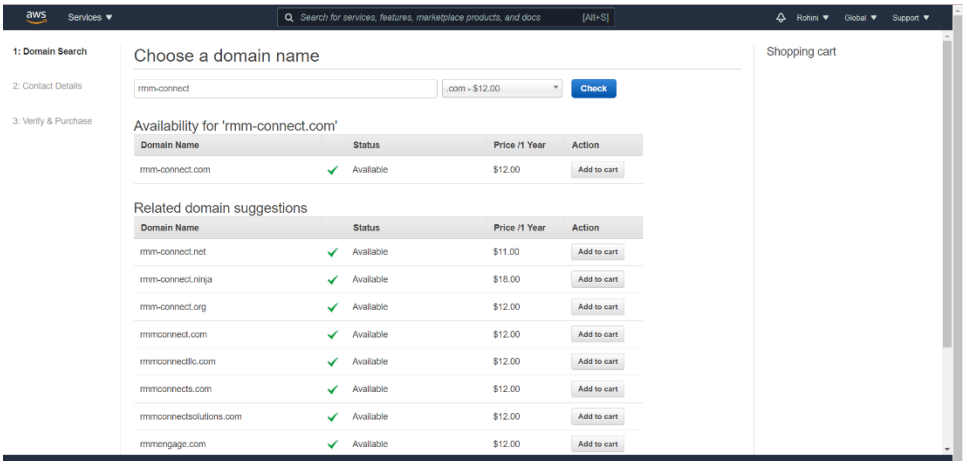



12. Once you are satisfied with the bot, click on build and then publish.

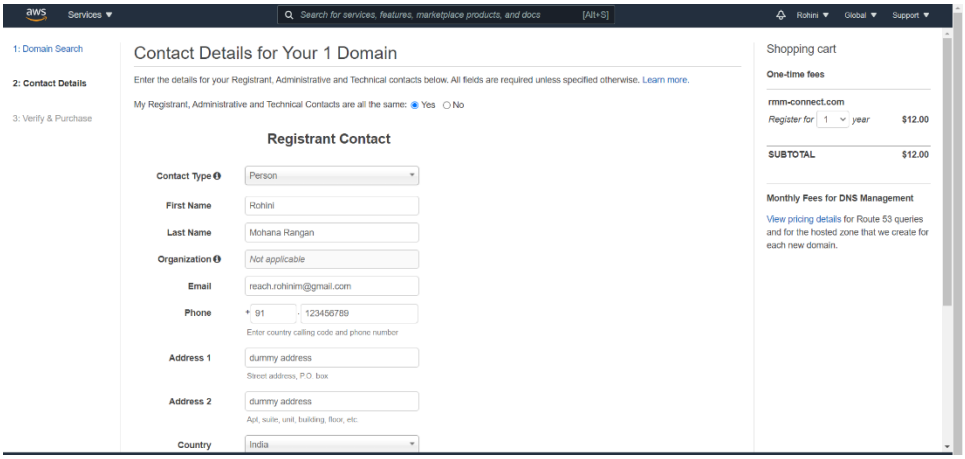
**How to create a registered domain with Route 53**

1. Sign in to the AWS Management Console and open the Route 53 console at https://console.aws.amazon.com/route53/.
2. If you're new to Route 53, choose **Get started**.
3. If you're already using Route 53, in the navigation pane, choose **Registered domains**.
4. Choose **Register domain**, and specify the domain that you want to register:

5. Enter the domain name that you want to register, and choose **Check** to find out whether the domain name is available.

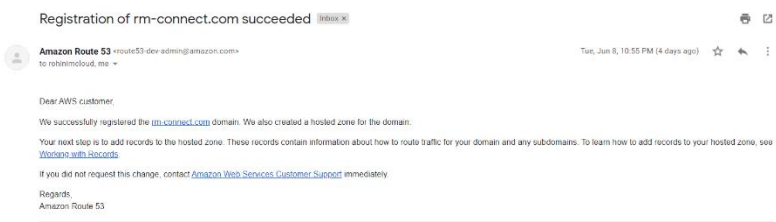



6. In our case we registered the domain as **rm-connect.com**
7. If the domain is available, choose Add to cart. The domain name appears in your shopping cart.

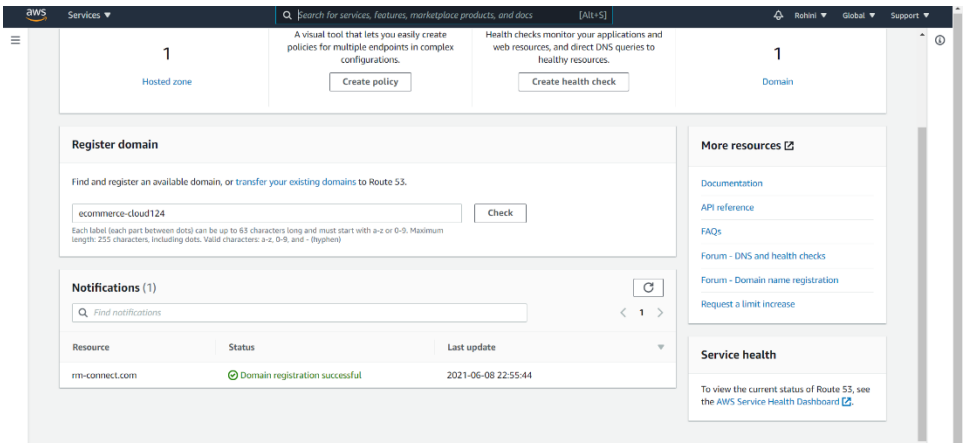



8. The Related domain suggestions list shows other domains that you might want to register instead of your first choice (if it's not available) or in addition to your first choice.
9. In the shopping cart, choose the number of years that you want to register the domain for.
10. Choose Continue.

11. On the Contact Details for Your Domain page, enter contact information for the domain registrant, administrator, and technical contacts.
12. When you receive the verification email, choose the link in the email that verifies that the email address is valid. If you don't receive the email immediately, check your junk email folder.
13. Return to the Route 53 console. If the status doesn't automatically update to say email-address is verified, choose Refresh status.
14. Choose whether you want us to automatically renew your domain registration before the expiration date.




15. Review the information that you entered, read the terms of service, and select the check box to confirm that you've read the terms of service.
16. Choose **Complete Purchase**.
17. It will now appear in the resources section:

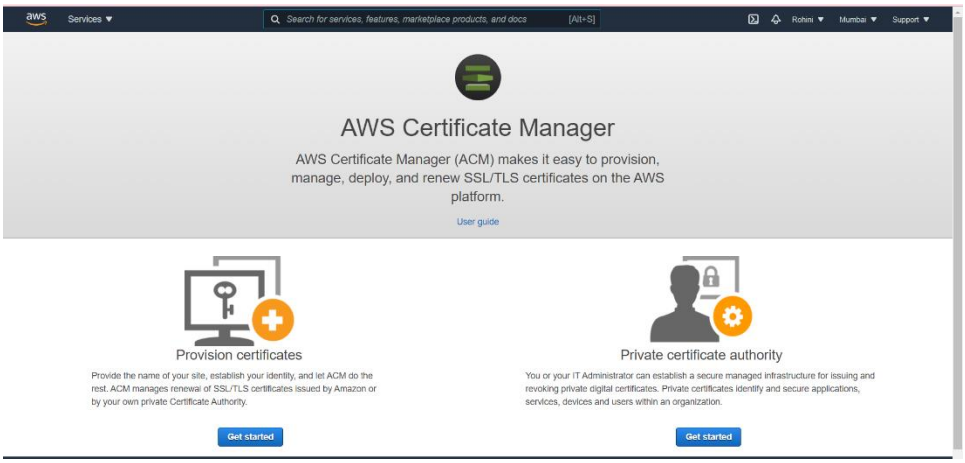



**How to Request a public certificate using the console on Amazon certificate manager**

1. Sign into the AWS

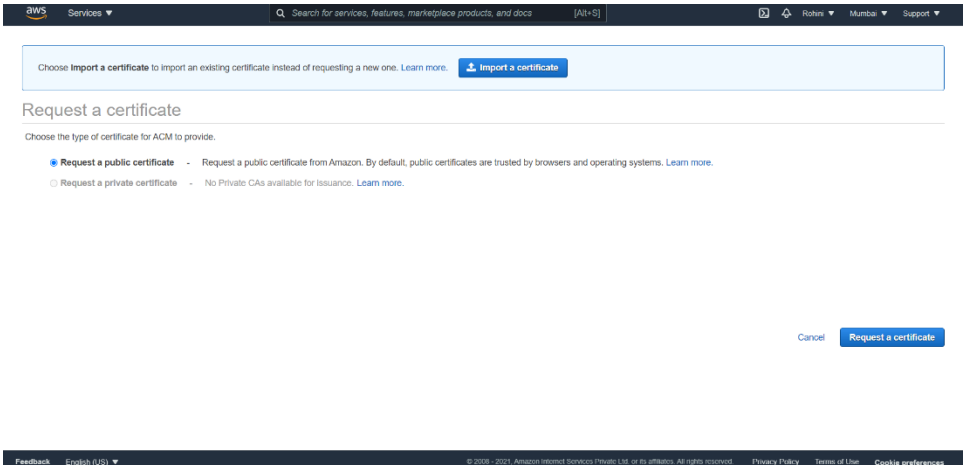
Managemen
t Console
and open
the ACM
console at
https://cons
ole.aws.am
azon.com/a
cm/home.

2. Click on get started.

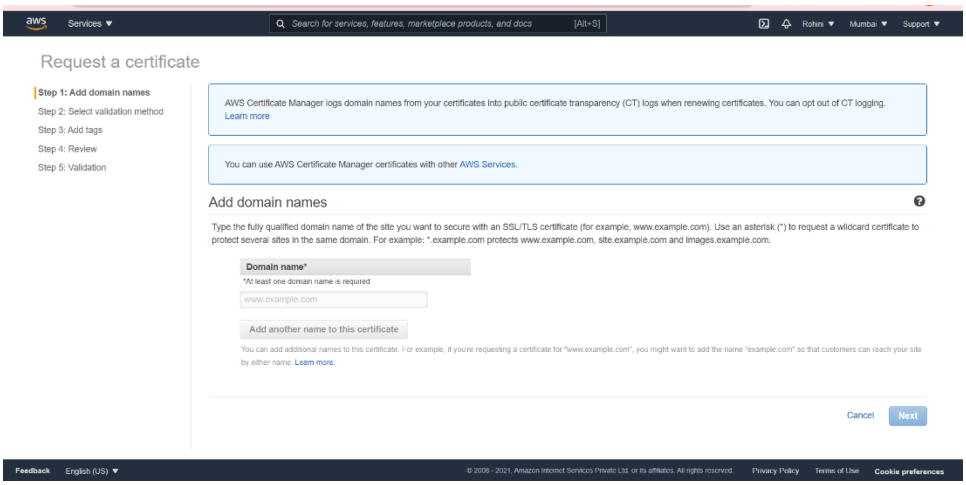



5. Choose **Request a certificate**.

6. On the **Request a certificate** page, choose **Request a public certificate** and **Request a certificate** to continue.




7. On the Add domain names page, type your domain name.

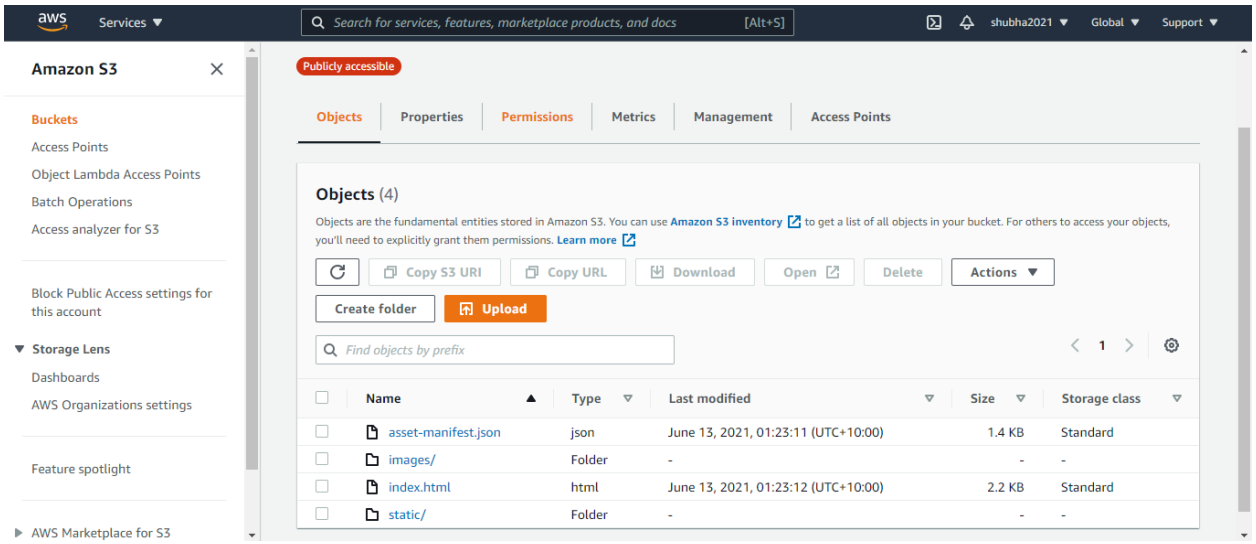



8. On the **Select validation method** page, choose either **DNS validation** or **Email validation**, depending on your needs.
9. If the **Review** page contains correct information about your request, choose **Confirm and request**. A confirmation page shows that your request is being processed and that certificate domains are being validated. Certificates awaiting validation are in the **Pending validation** state.

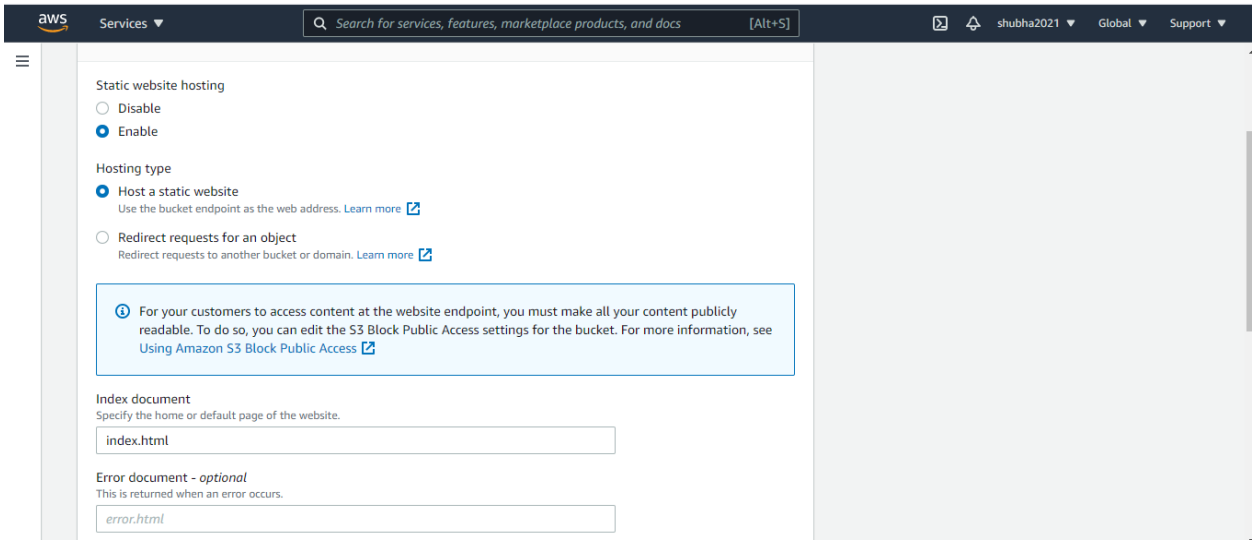
**How to deploy the front-end on S3**

1. Sign in to the AWS Management Console and open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. Choose Create bucket.
3. The Create bucket wizard opens.
4. In Bucket name, enter a DNS-compliant name for your bucket. Our buckets will be rm-connect.com and www.rm-connect.com
5. Apply a suitable policy.

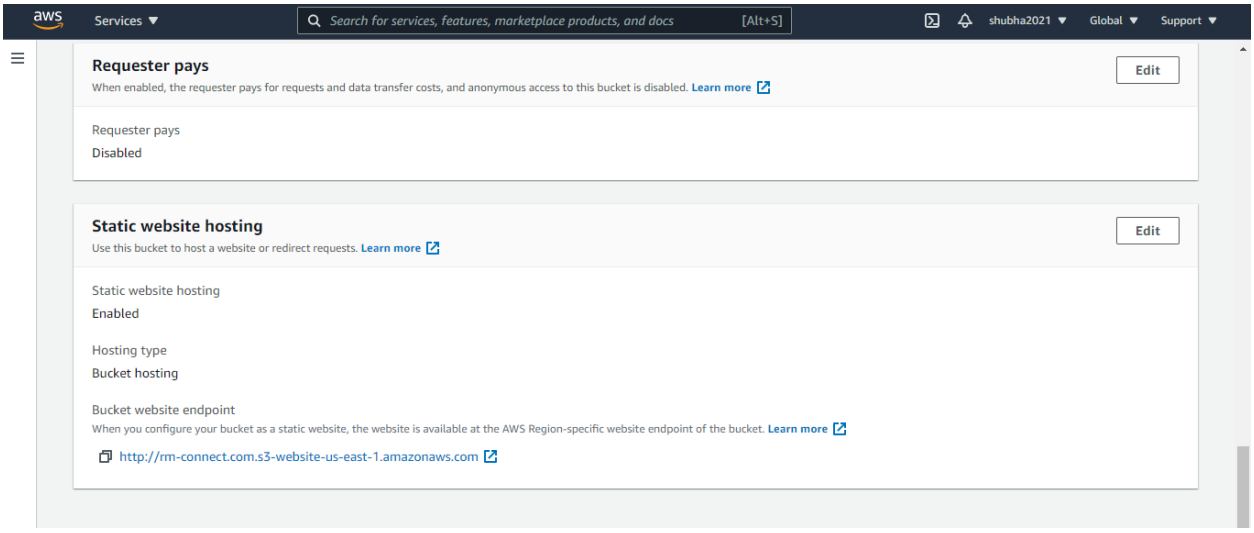
6. And click Create Bucket.
6. Once the bucket is created click on the Bucket name and in object upload the frontend code.
7. To upload the code, go to VS code and run “npm run build” it will create a build folder in the locally developed frontend code.
8. Select all those 4 files created inside the build folder and upload it to the rm-connect bucket’s object.
9. Now, go in permissions section under the page where the code is uploaded and select “grant public read access” and save.




10. It will be uploaded to the object.
11. Now go to the Properties tab and at the bottom of the page in Static web hosting section at the bottom select enable static website hosting and in Index document enter index.html and save changes.




12. It will automatically generate a url " http://rm-connect.com.s3-website-us-east-1.amazonaws.com" once we will click on the url it will take us to the home page of the website.




13. Finally, we can go to CloudFront We need two new web distributions, one for each S3 bucket.
14. Viewer Protocol Policy: Set to "Redirect HTTP to HTTPS".
15. Back in S3, go to your secondary bucket in the Properties tab and under Static Website Hosting set the redirect protocol to HTTPS.