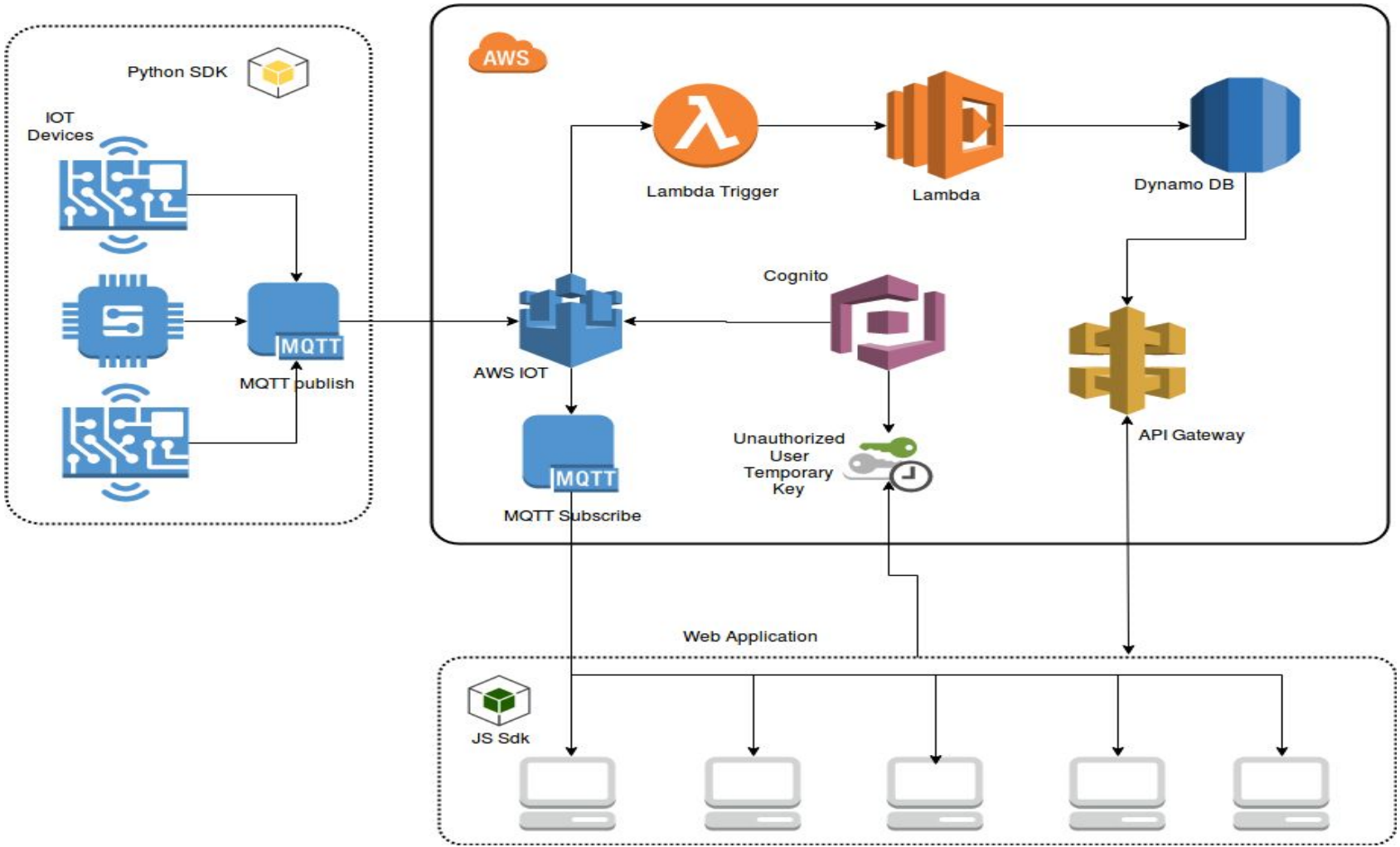
Smart bin Simulation

Tracking bin level and routing

Work flow


Python SDK
IOT
Devices
MQTT
MQTT publish
AWS
Lambda Trigger
Lambda
Dynamo DB
Cognito
AWS IOT
API Gateway
MQTT
MQTT Subscribe
Unauthorized
User
Temporary
Key
Web Application
JS Sdk

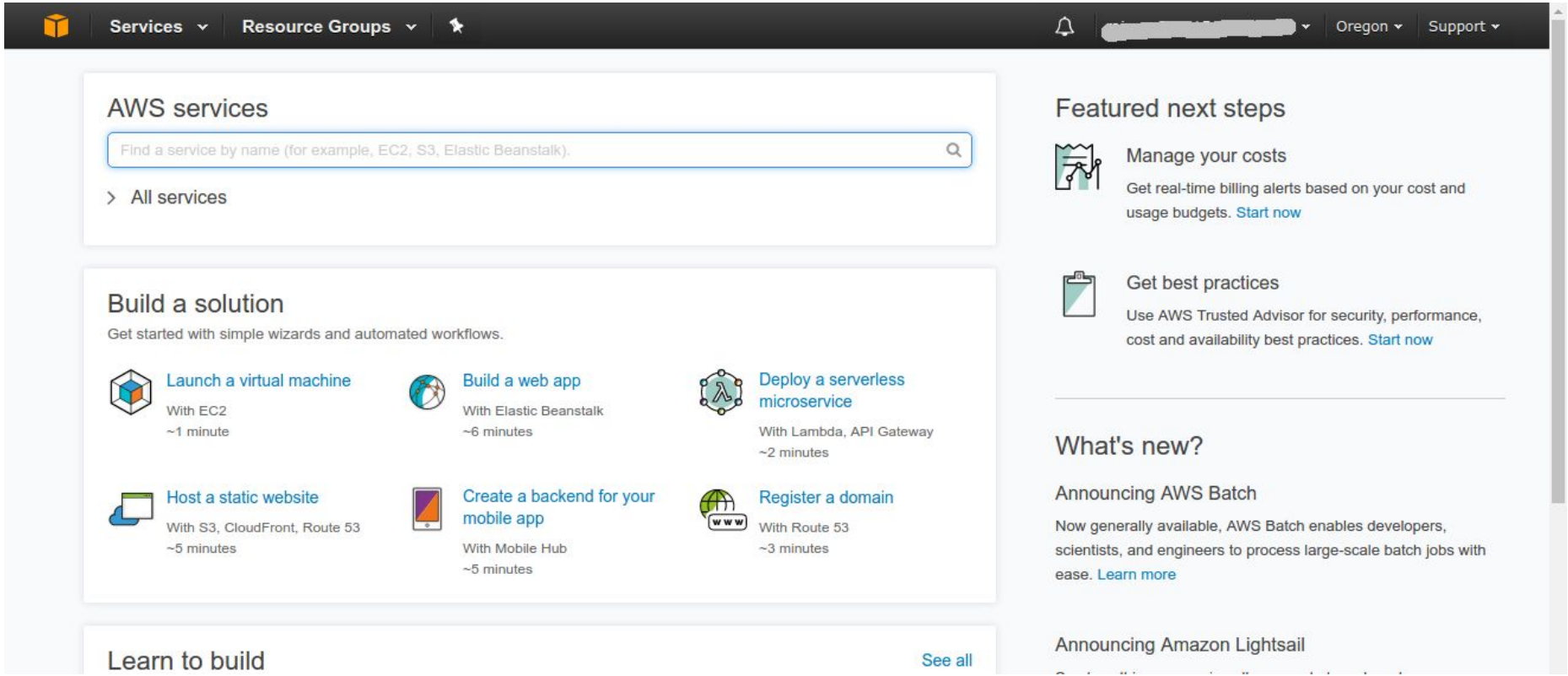
AWS Services Used

1. Dynamo Database
2. Lambda
3. AWS IoT
4. Cognito
5. API Gateway

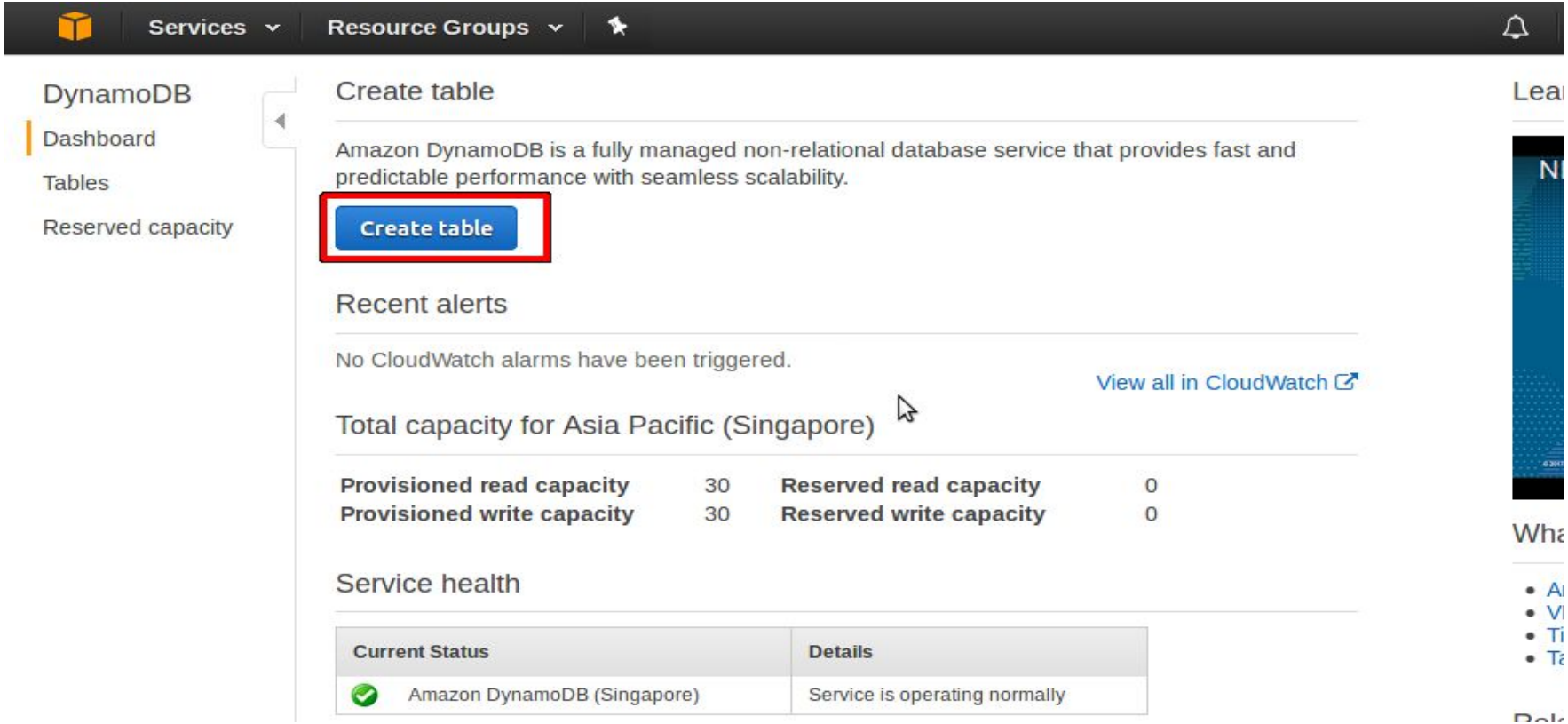
Steps for creating aws services and configuring role , policy and rules .

Dynamo DB

Step 1: Login to your AWS Account and click on “All services”



STEP 2 : To Create Dynamo db instance , login to your aws console and open dynamo db service.

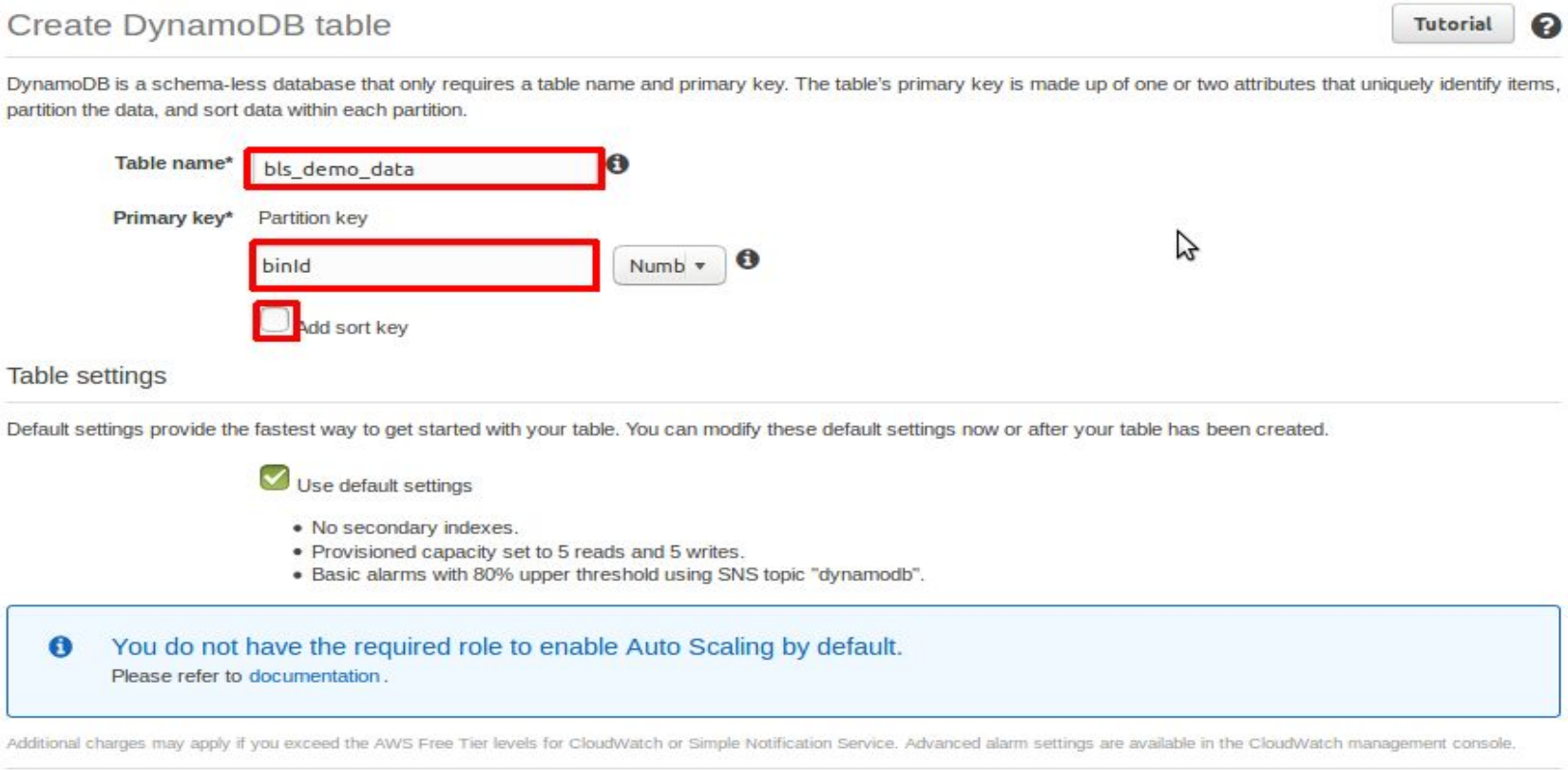


Step 3 : Enter the Table name , Primary key and Sort key name. And click on “Create” button.

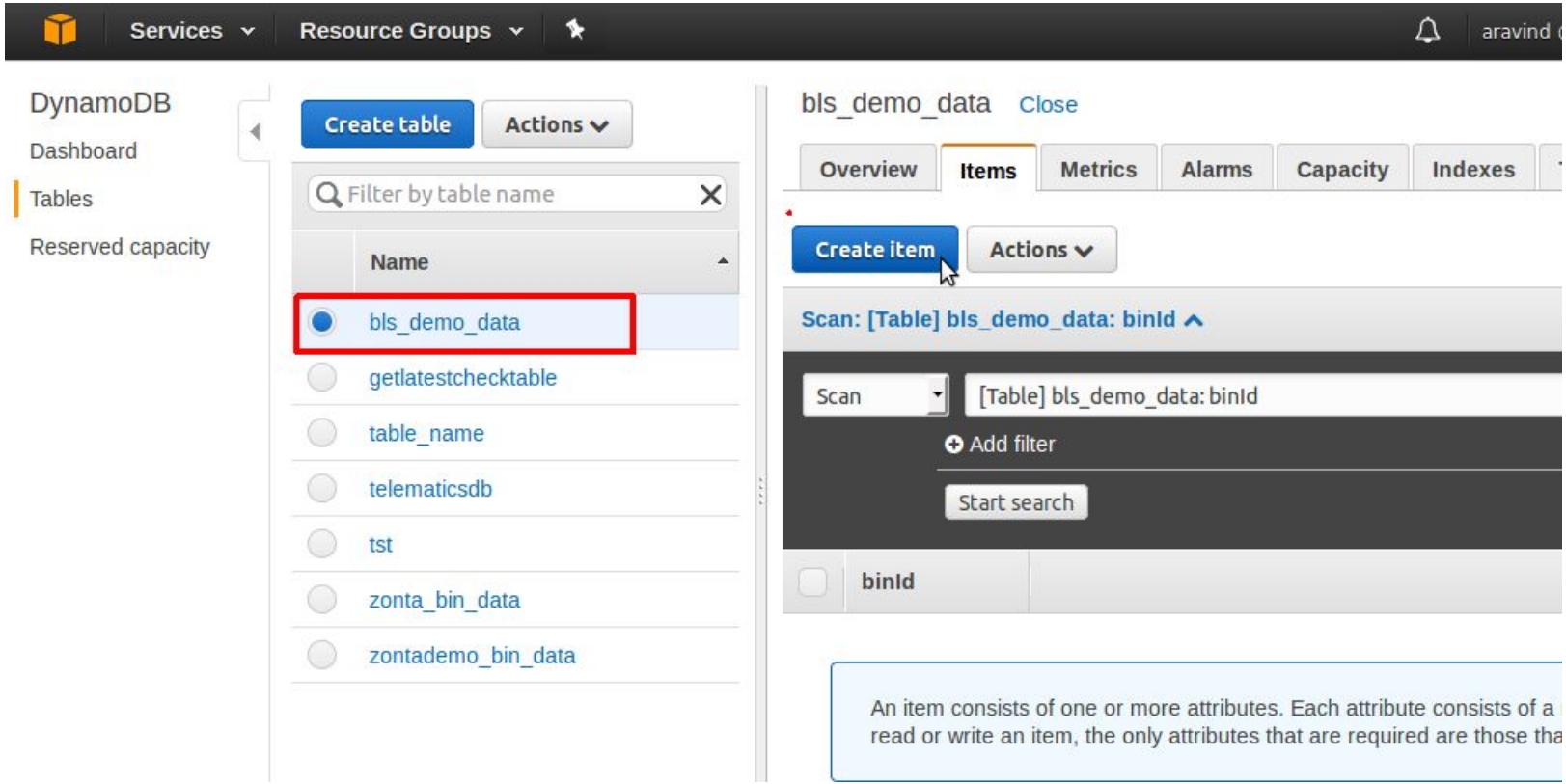
Setting key parameter :

Primary key : ‘binid’
Sort key : ‘time’

Provide a proper table name and give binid (number) as primary key and time (string) as your sort key , and click create table button below.

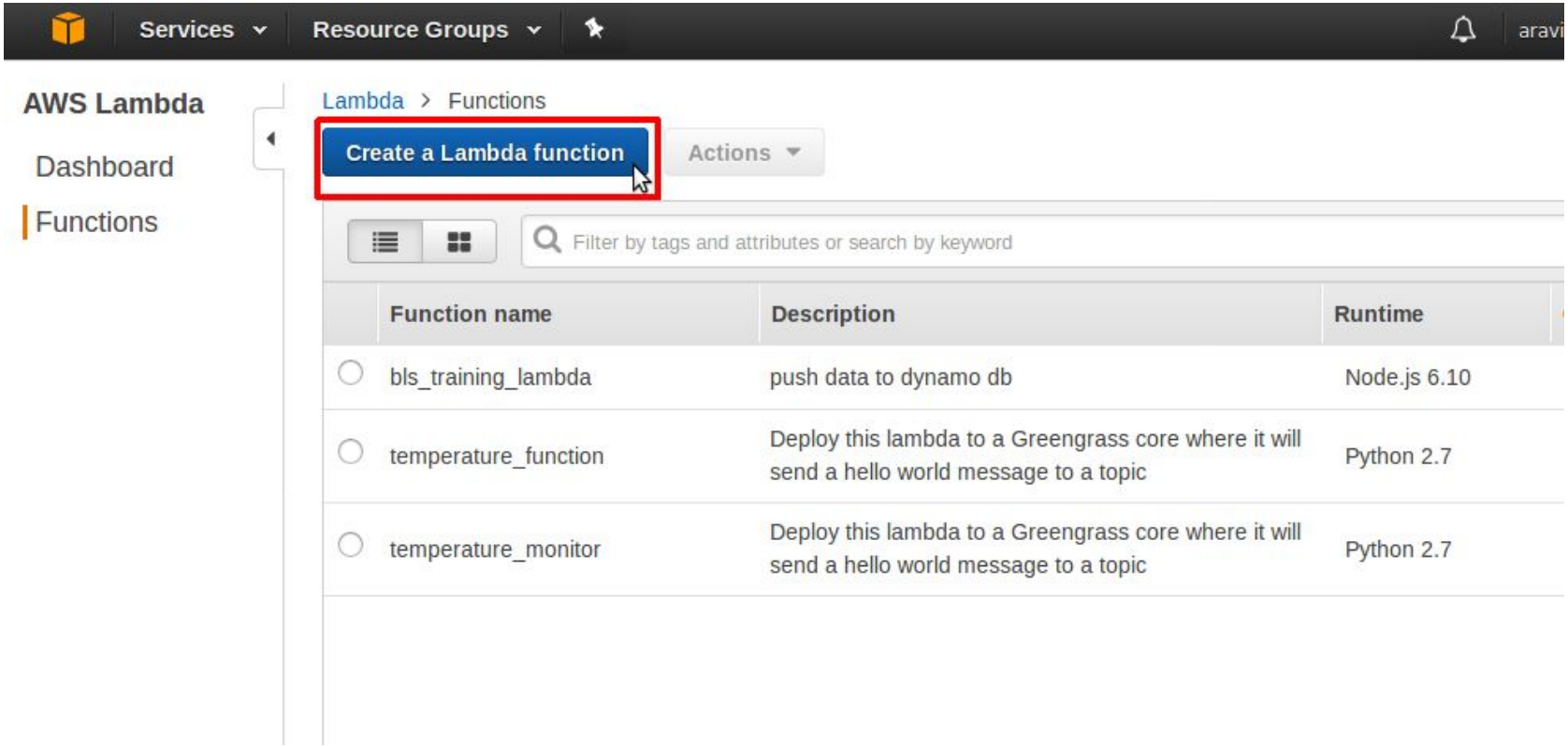


STEP 4: Here you can see the created items in the view , and all other dynamo db table specifications.

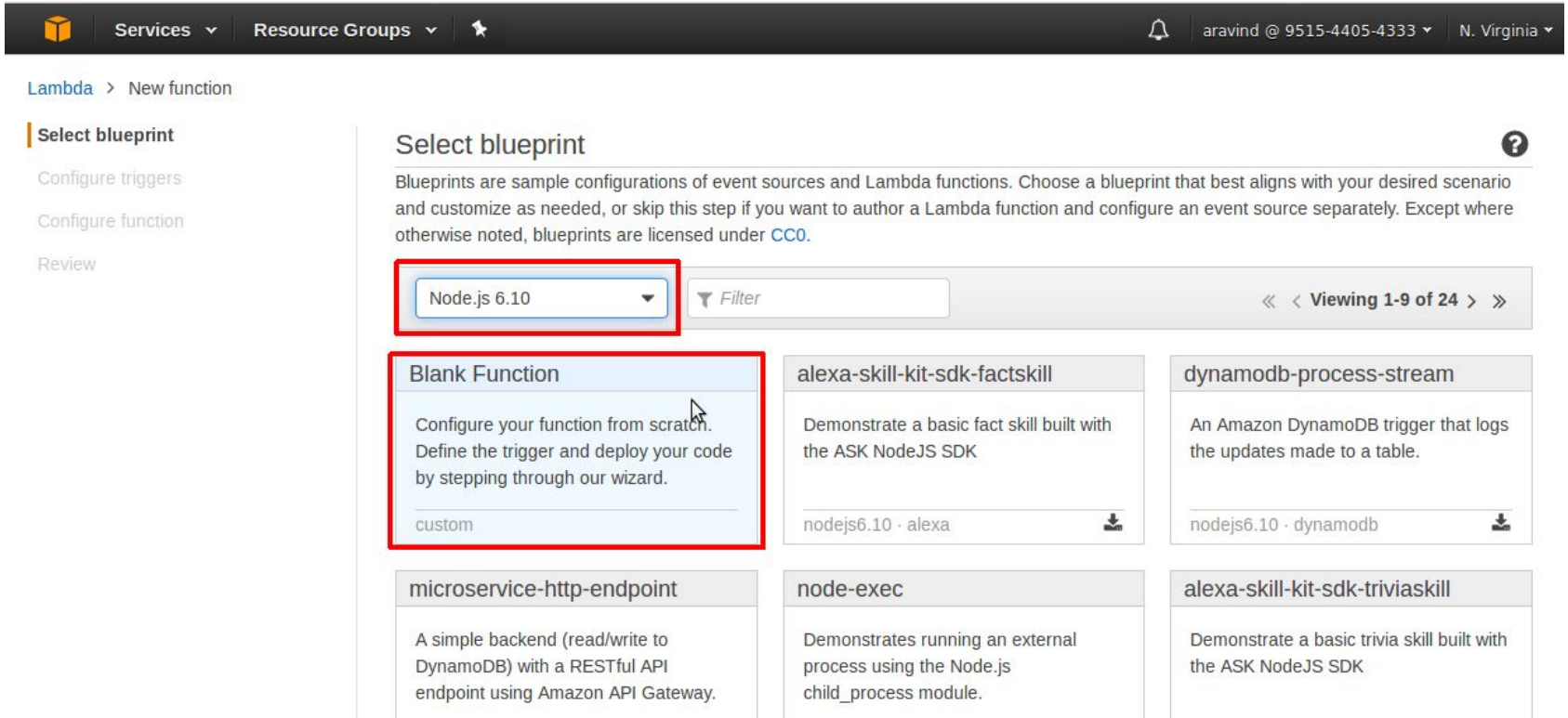


Lambda Function

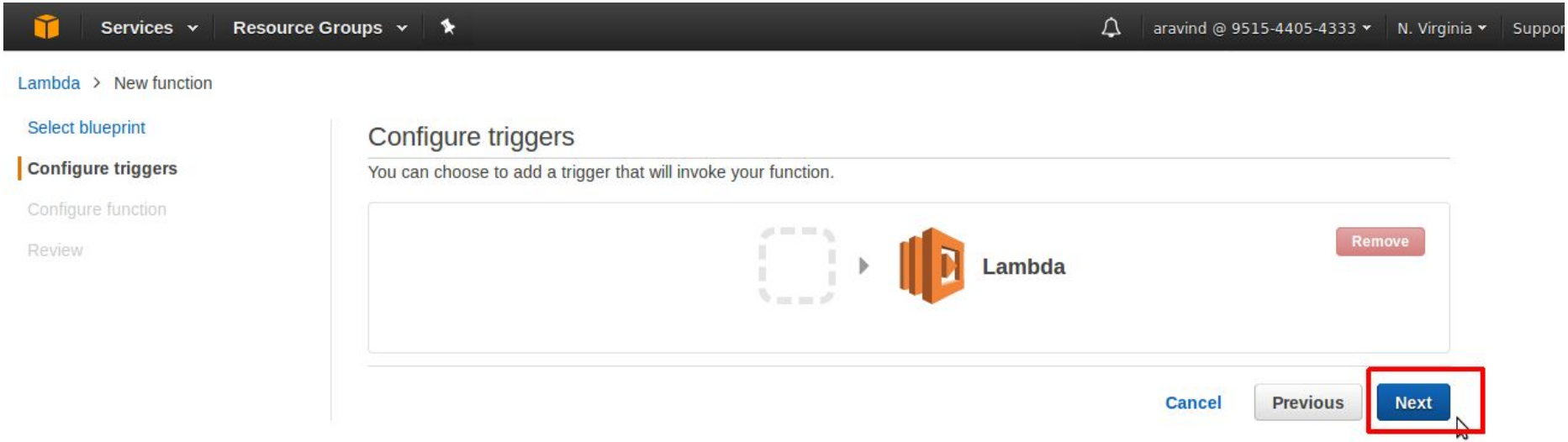
STEP 1: Login to your aws account and select Lambda service , and click Create Lambda Function



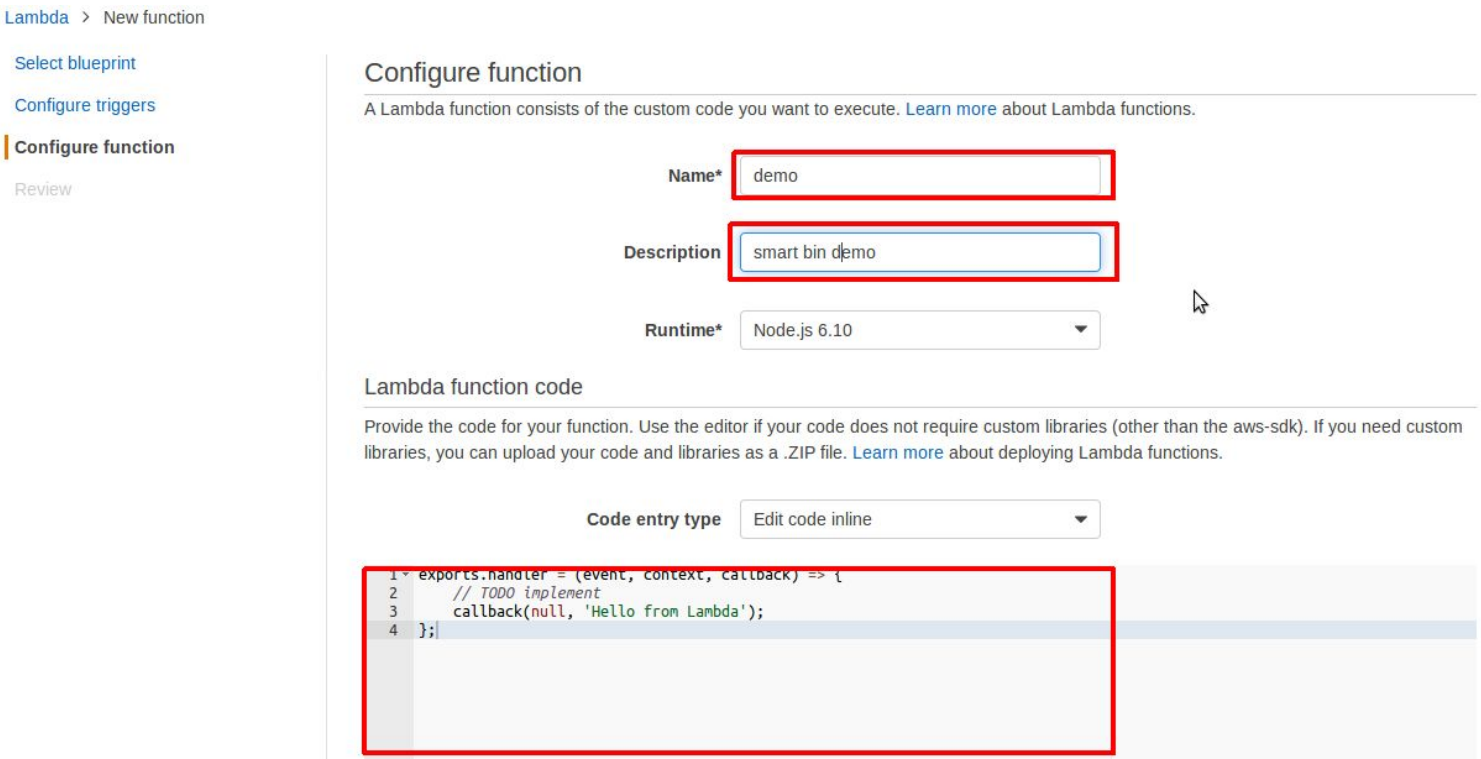
STEP 2 : Select the programming language and blueprint for our lambda function



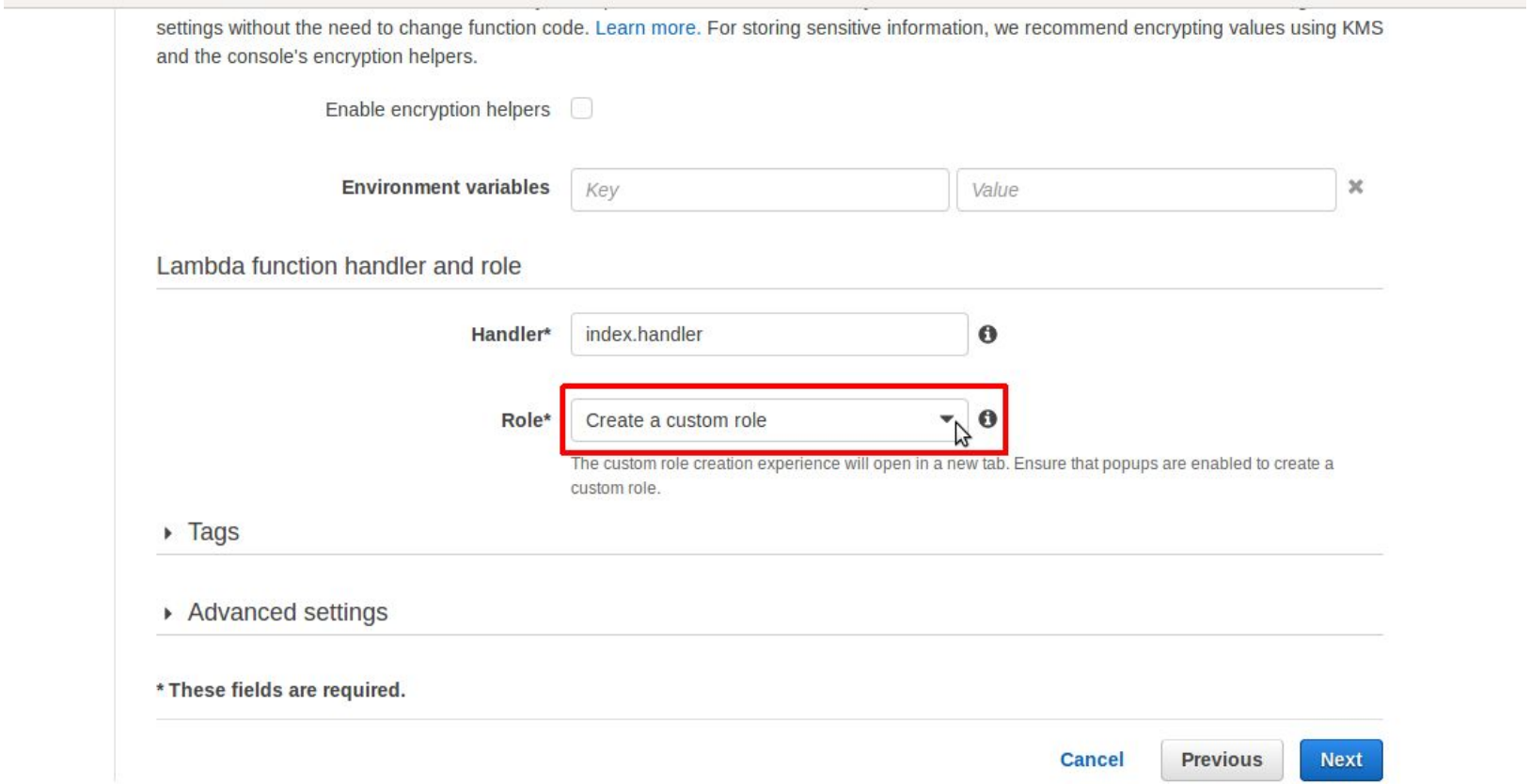
STEP 3: Configure trigger to our lambda function (this thing can be done in the later stages)



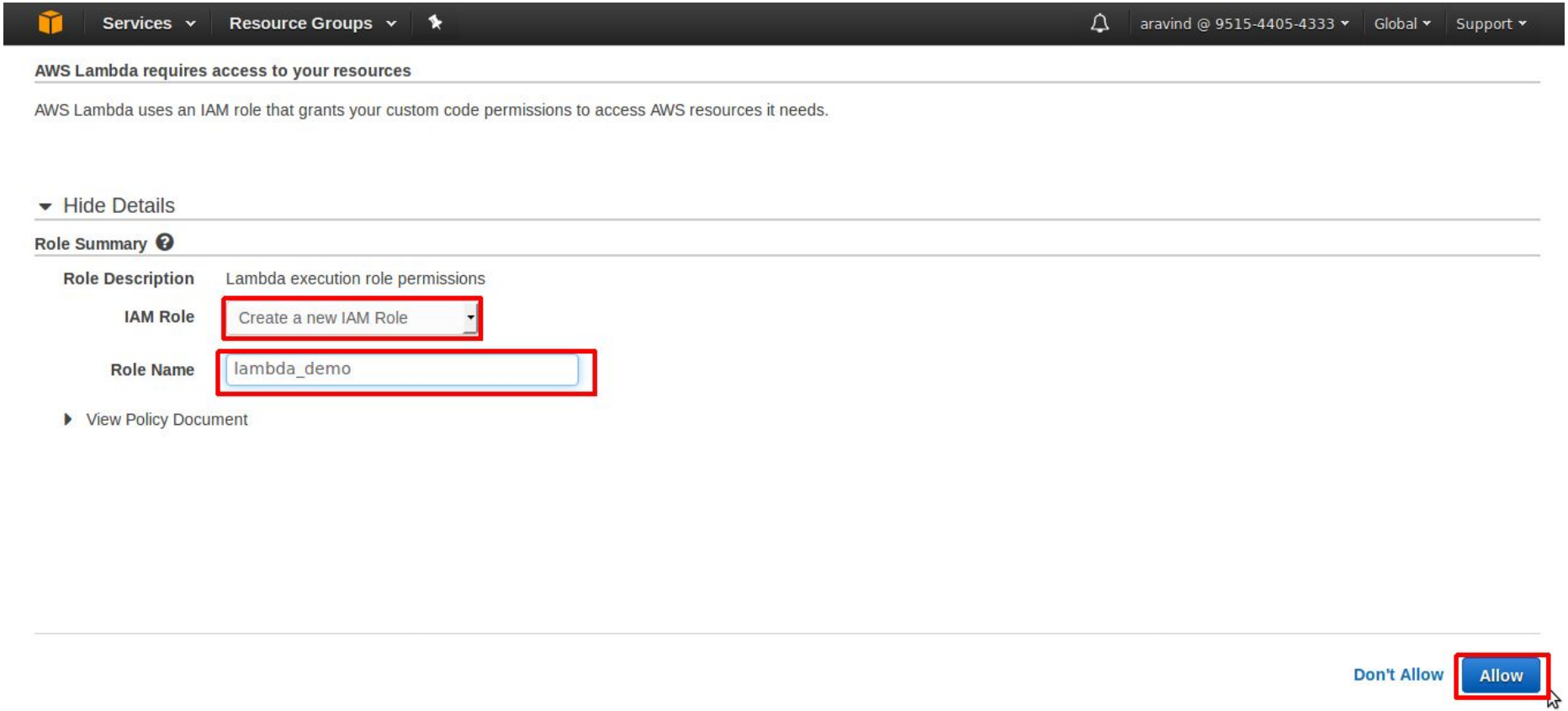
STEP 4 : Provide the function name, run time and place our lambda function code in the code block below



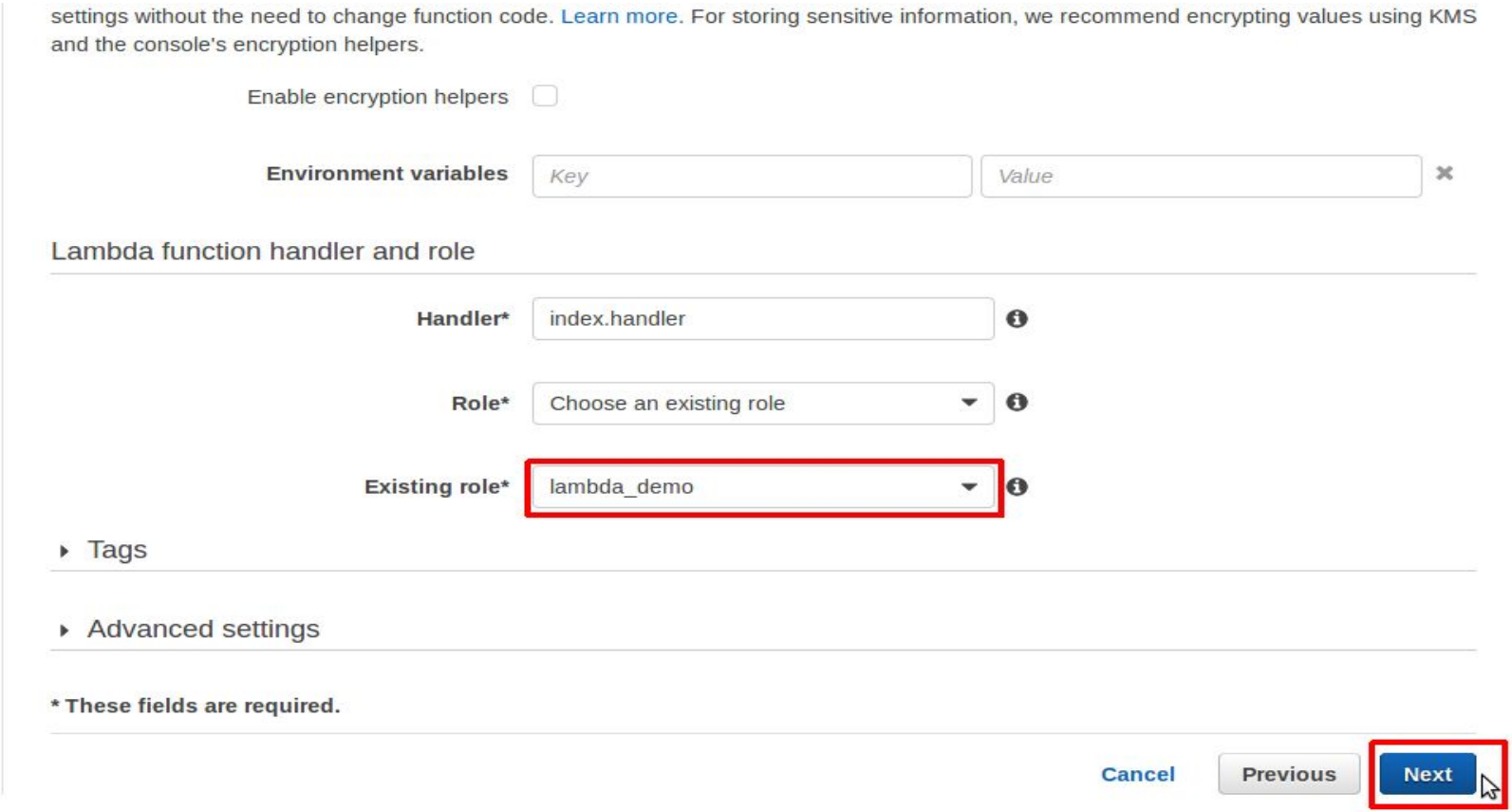
STEP 5 : Create custom role for our lambda function to push data to our dynamo db



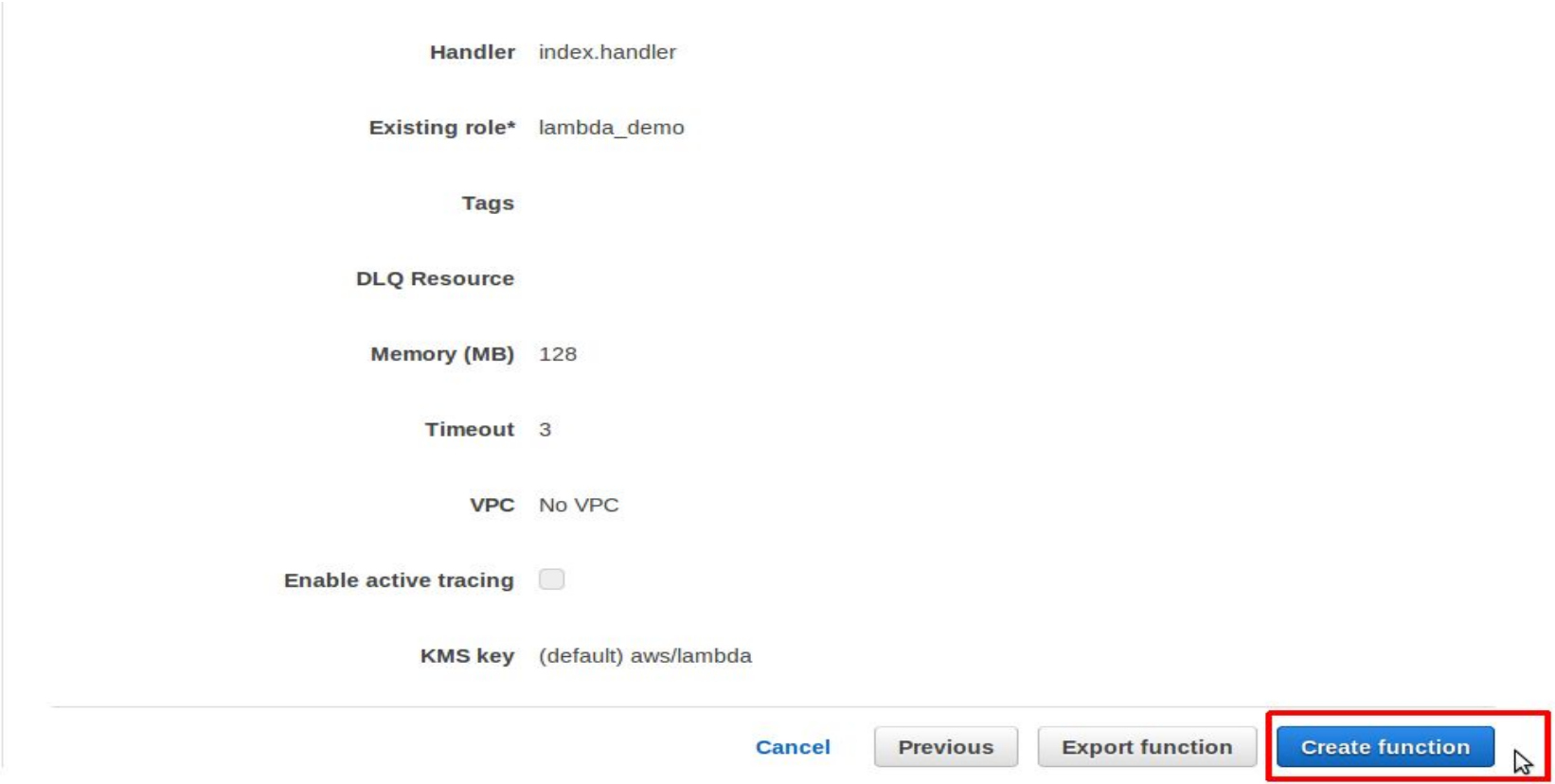
STEP 6 : Select create new role in IAM Role and provide a role name of your choice and click allow



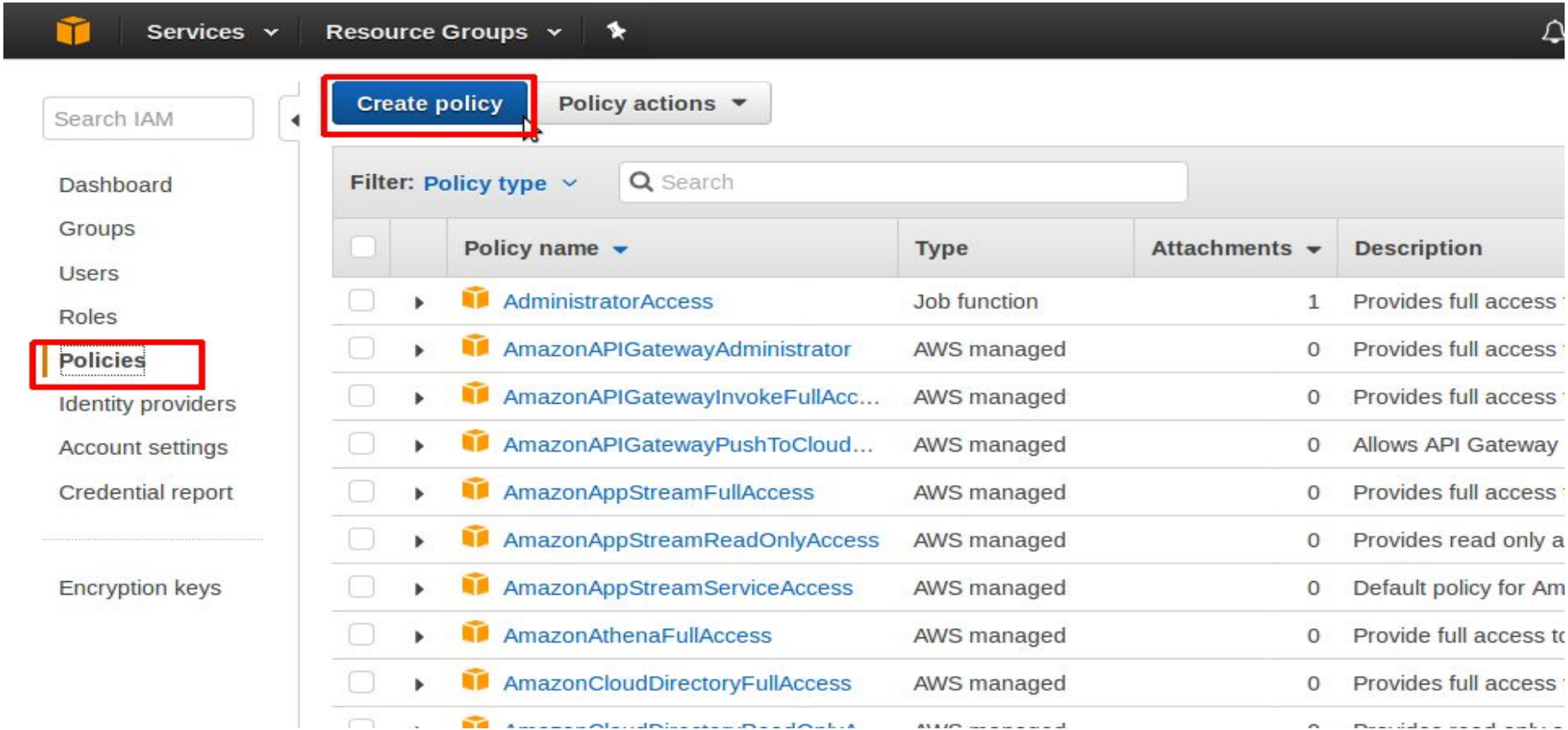
STEP 7 : Now select the created role and click Next button.



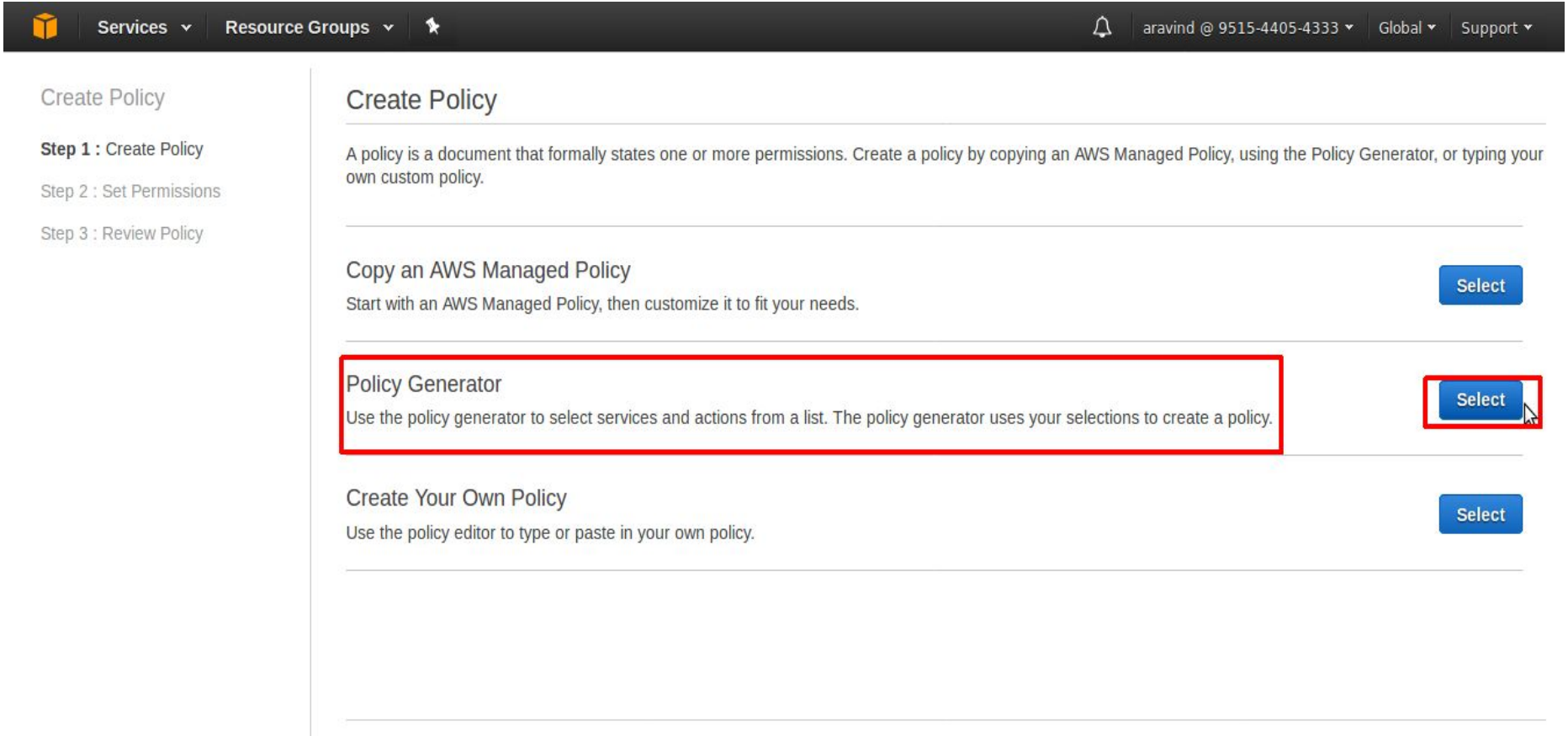
STEP 8 : Preview the settings and select Create function button to create our lambda



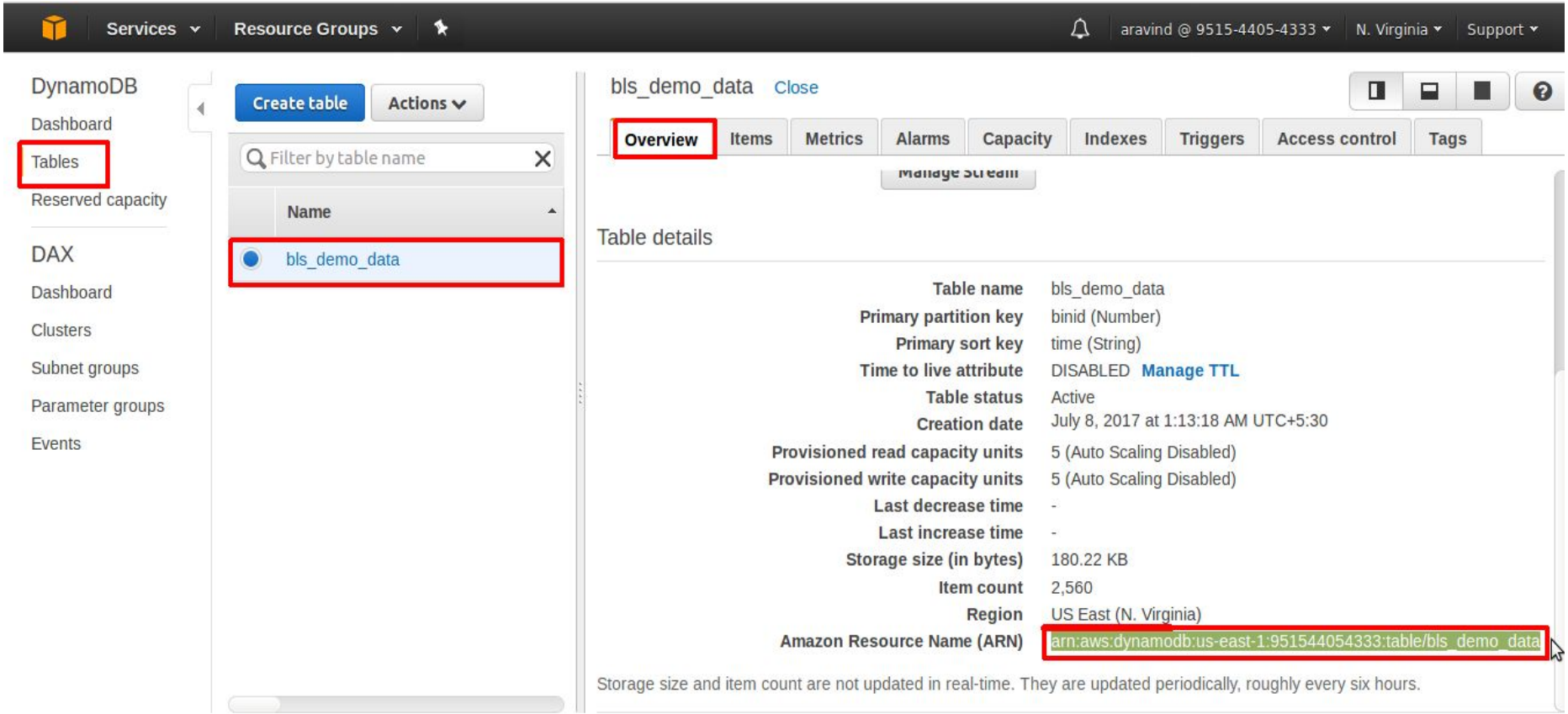
STEP 9 : To attach policy to the created role , open your IAM service and click Policy in left panel and select Create policy button.



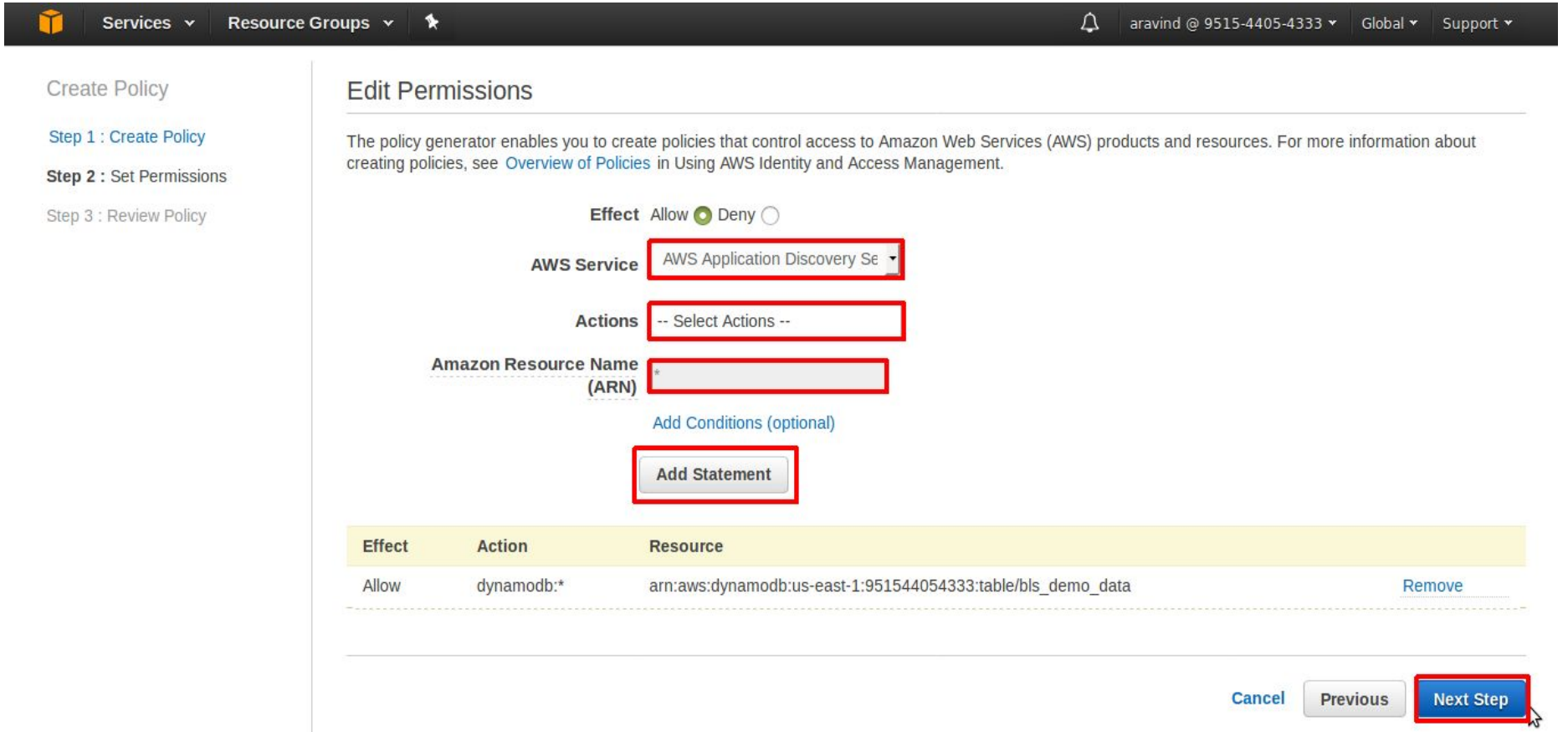
STEP 10 : Select policy generator and click select



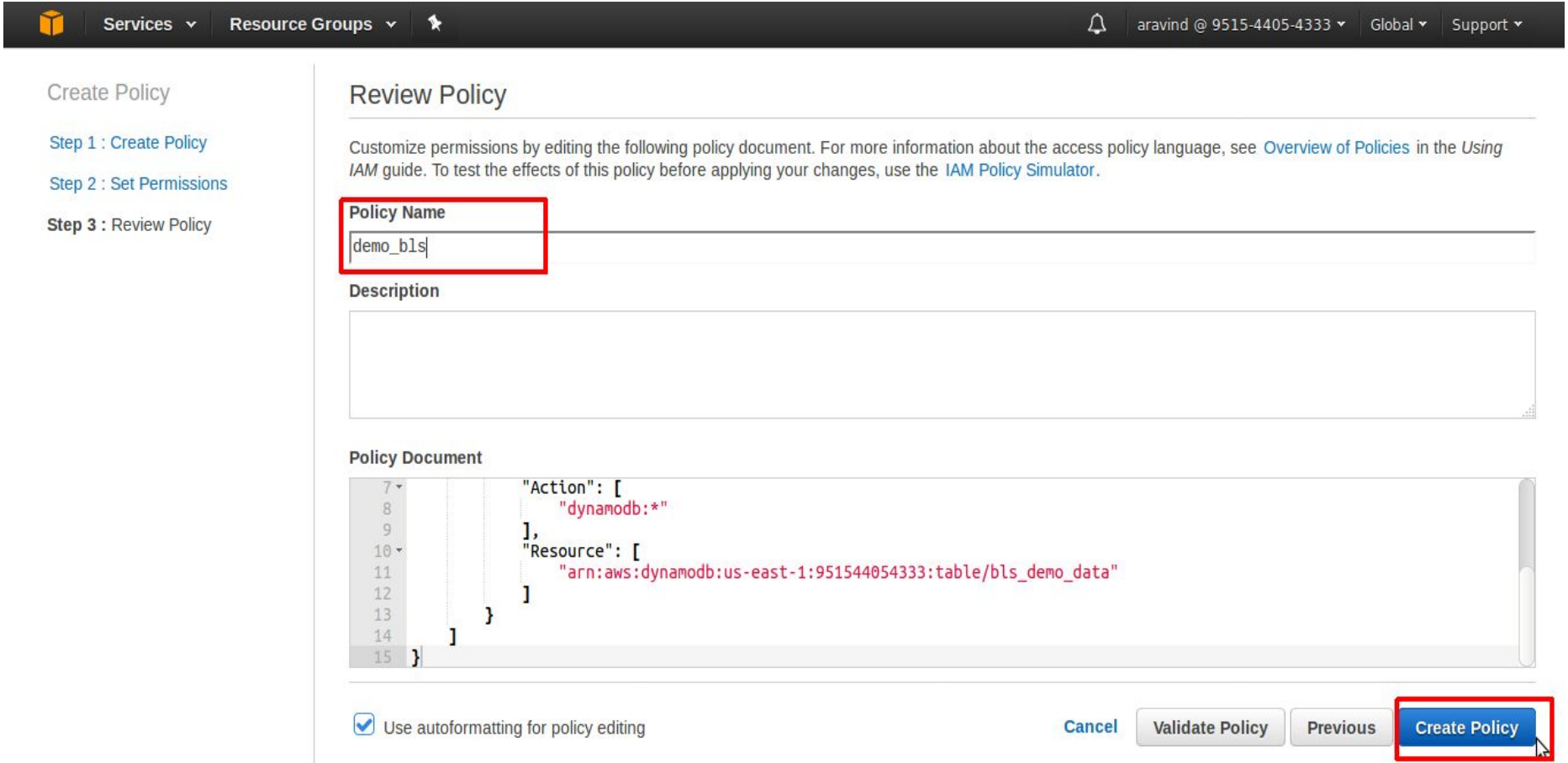
STEP 11 : Dynamo db ARN number is required for setting the policy permission , so open your dynamo db table and copy the ARN Number.



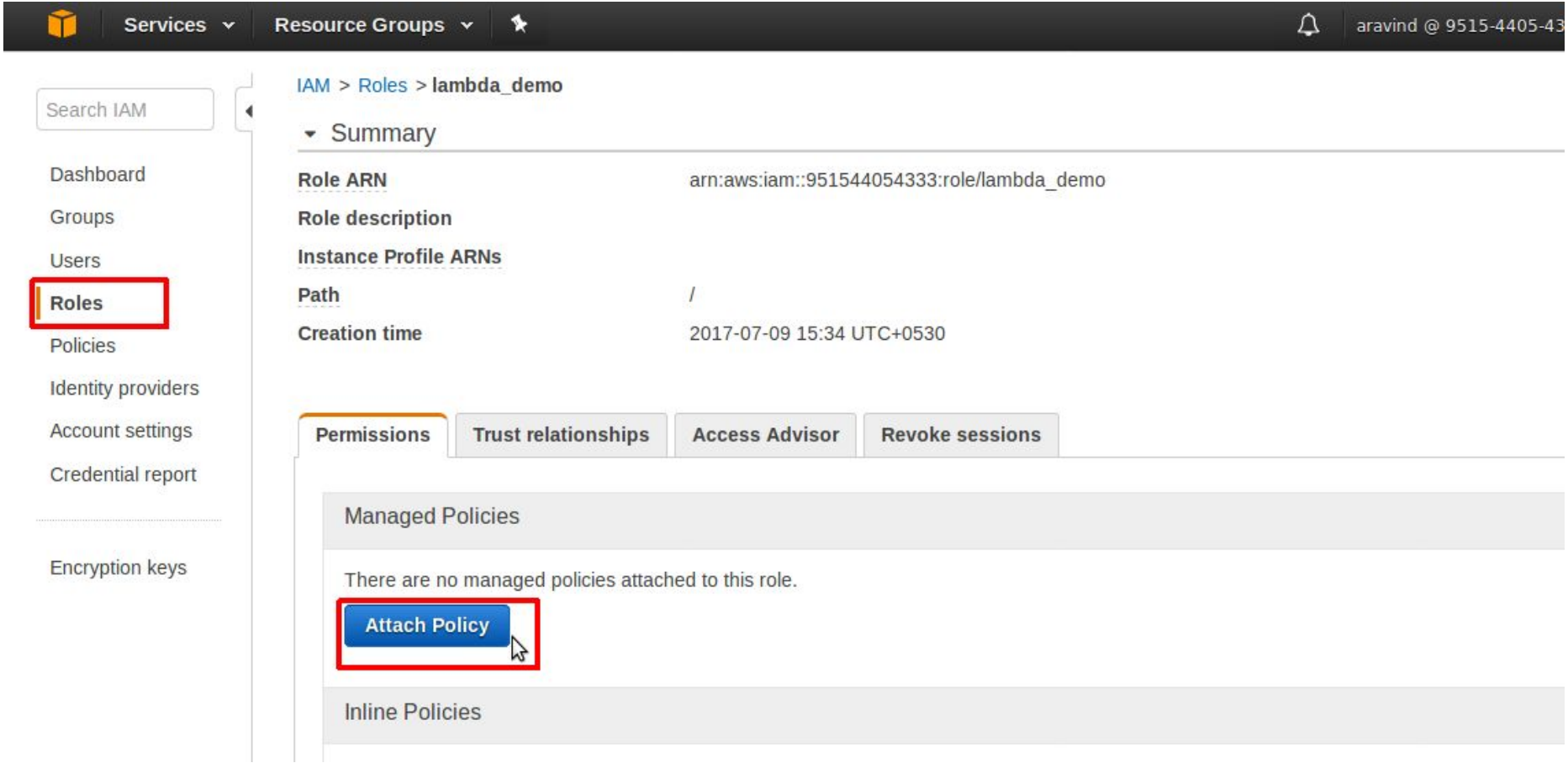
STEP 12 : Select AWS Service (dynamo db) , action (all operations), paste the arn number got from previous step , click Add Statements and click Next Step



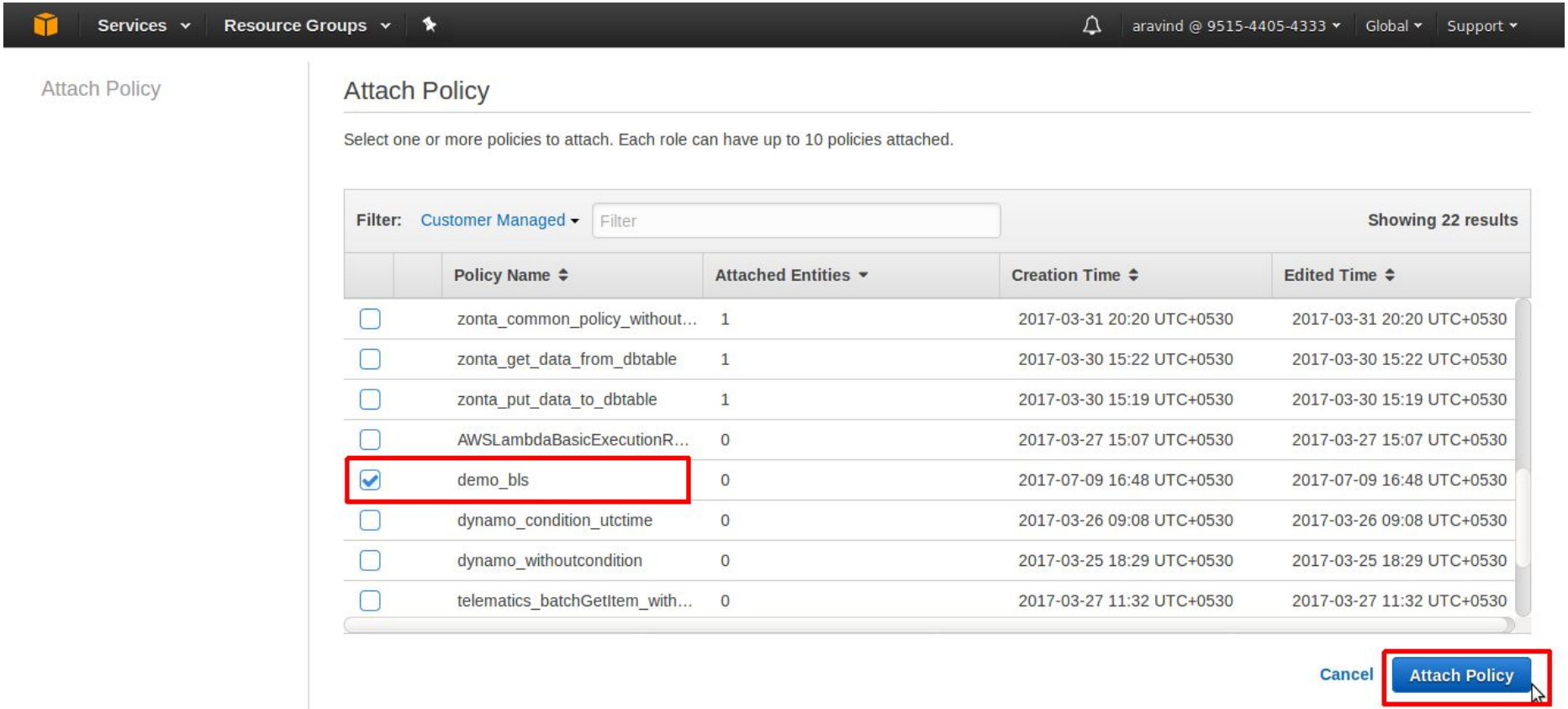
STEP 13 : Name the policy of your wish and click Create policy



STEP 14 : Open Role created for our lambda and click Attach policy

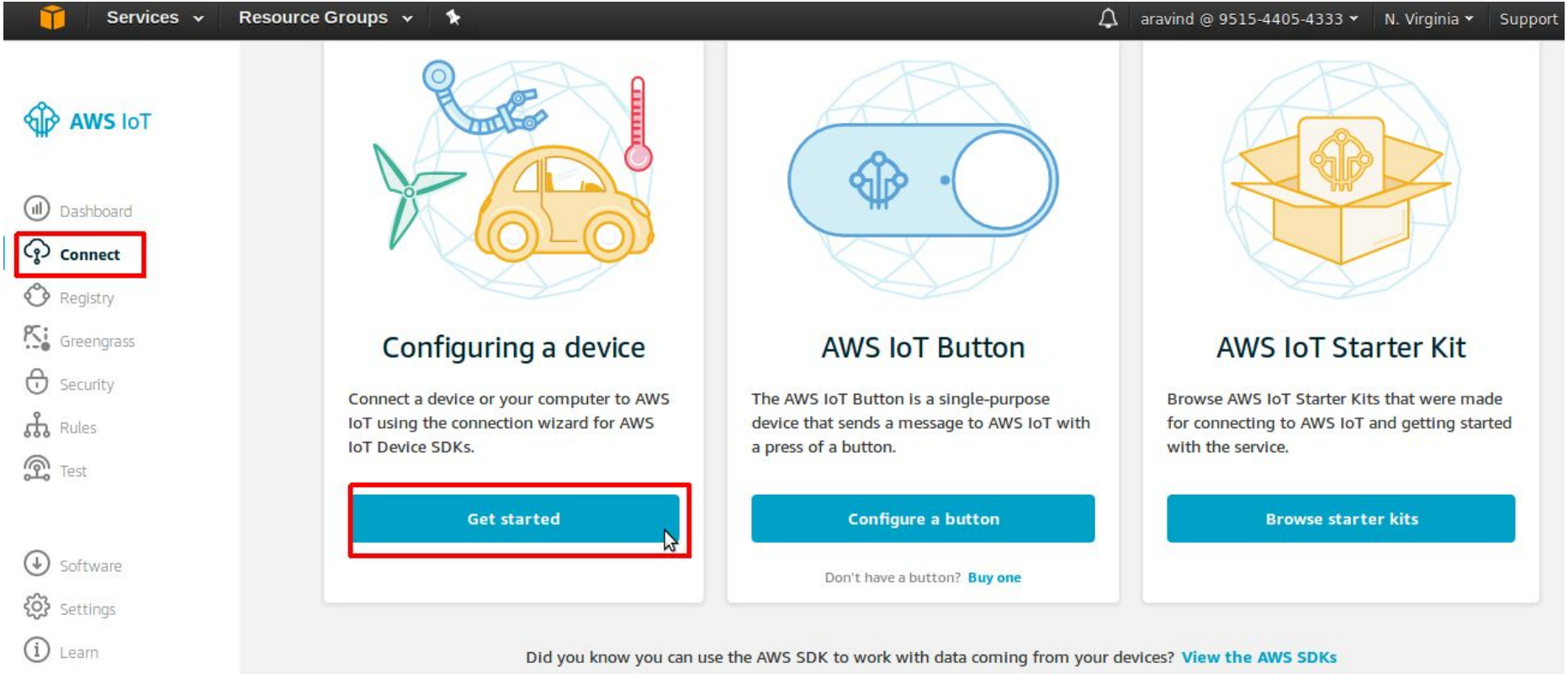


STEP 15 : Select the created policy and click Attach policy. Now dynamo db permission to push data from lambda function is done



AWS IOT Device creation

STEP 1 : Open AWS IOT service , click connect option and select Get started button .



STEP 2 : Select Get Started button .

Register a device

A thing is the representation and record of your physical device in the cloud. Any physical device needs a thing record in order to work with AWS IoT.

Download a connection kit

The connection kit includes some important components: **security credentials, the SDK of your choice, and a sample project.**

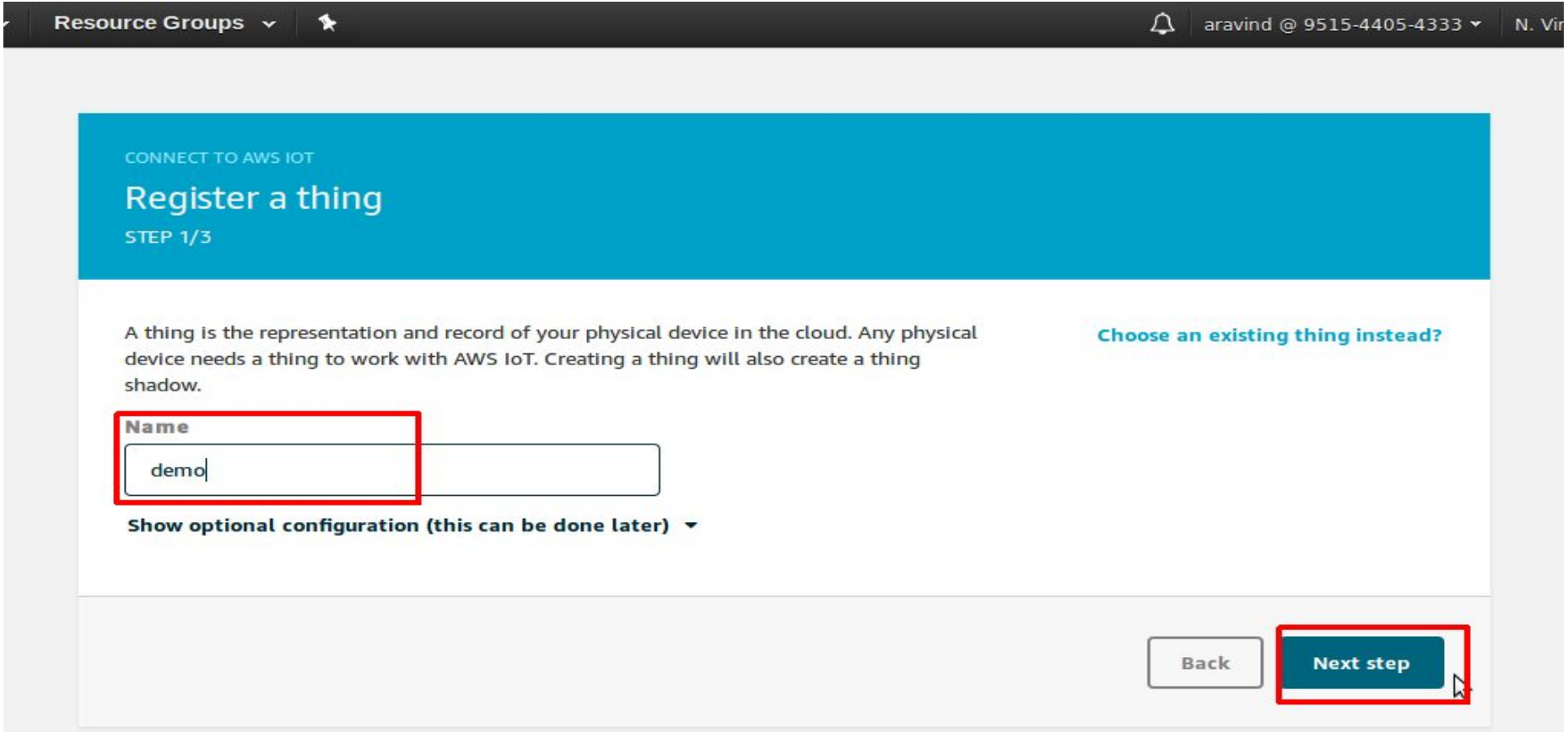
Configure and test your device

Using the connection kit, you will configure your device by **transferring files and running a script**, and **test that it is connected** to AWS IoT correctly.

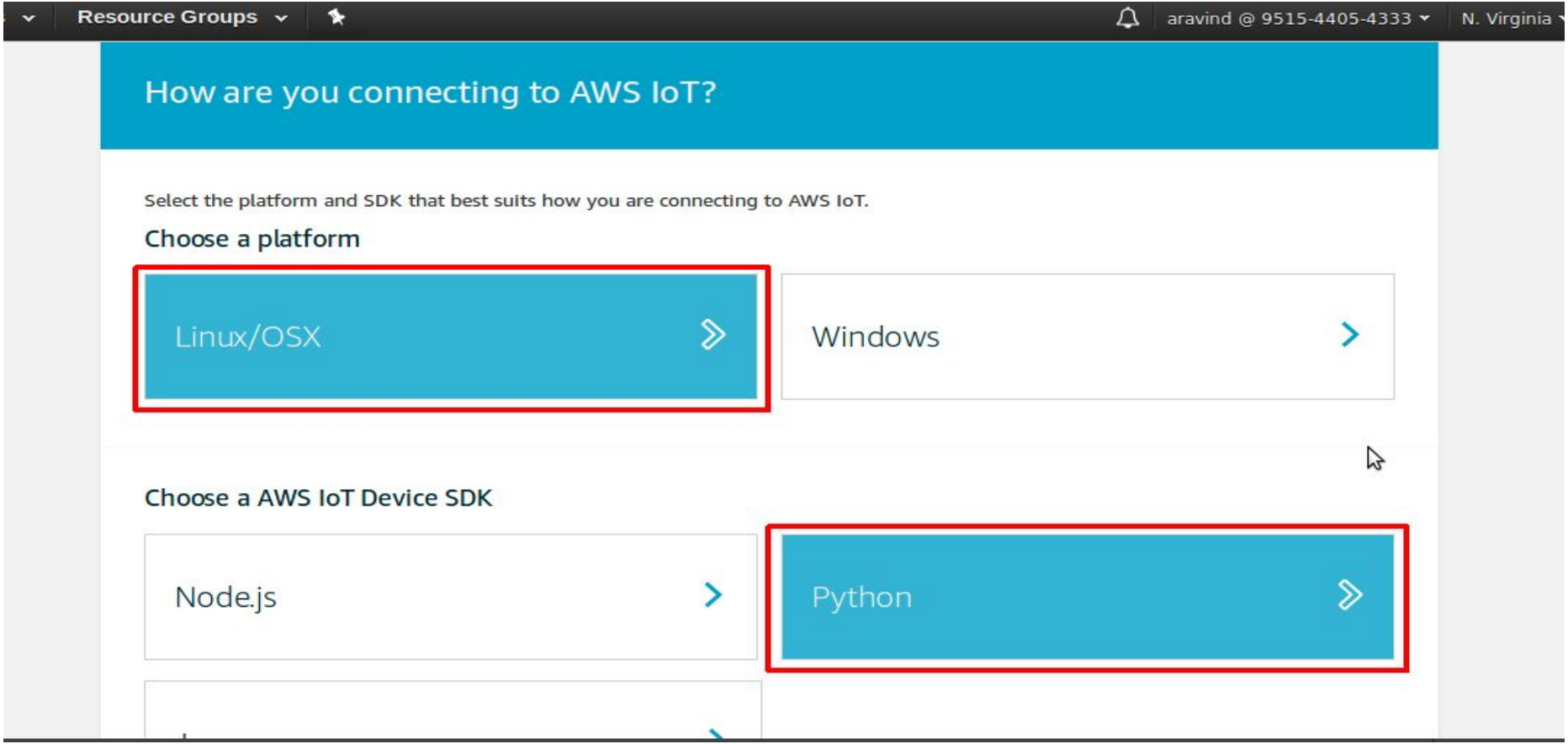
Want to learn more about the components of AWS IoT?
Try the interactive overview

Get started

STEP 3 : Select a name for thing (device) to register and click on Next step



STEP 4 : Choose a platform and SDK for our device and click Next



Choose a AWS IoT Device SDK

Node.js

Python

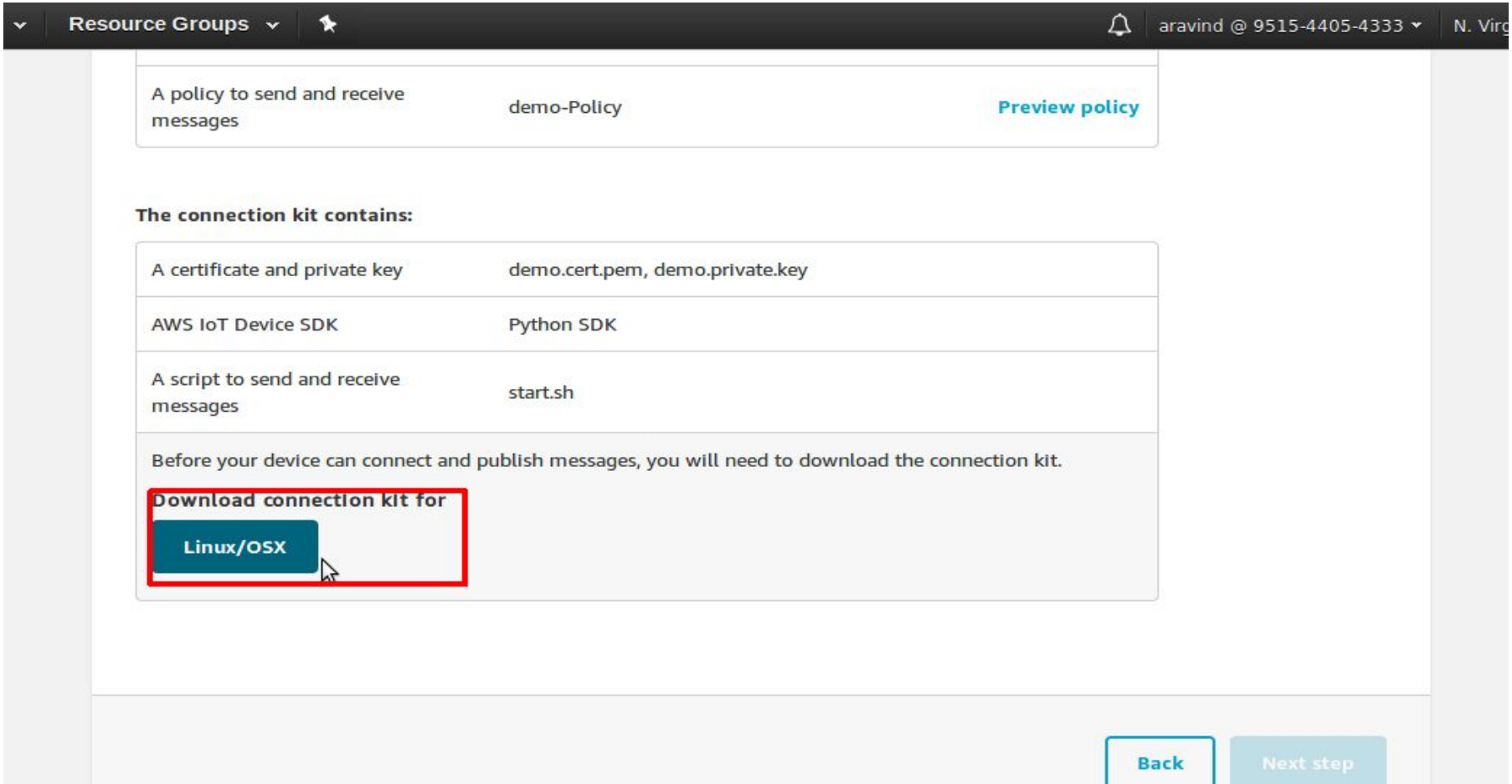
Java

Some prerequisites to consider:
the device should have **Python and Git Installed** and **a TCP connection to the public Internet on port 8883.**

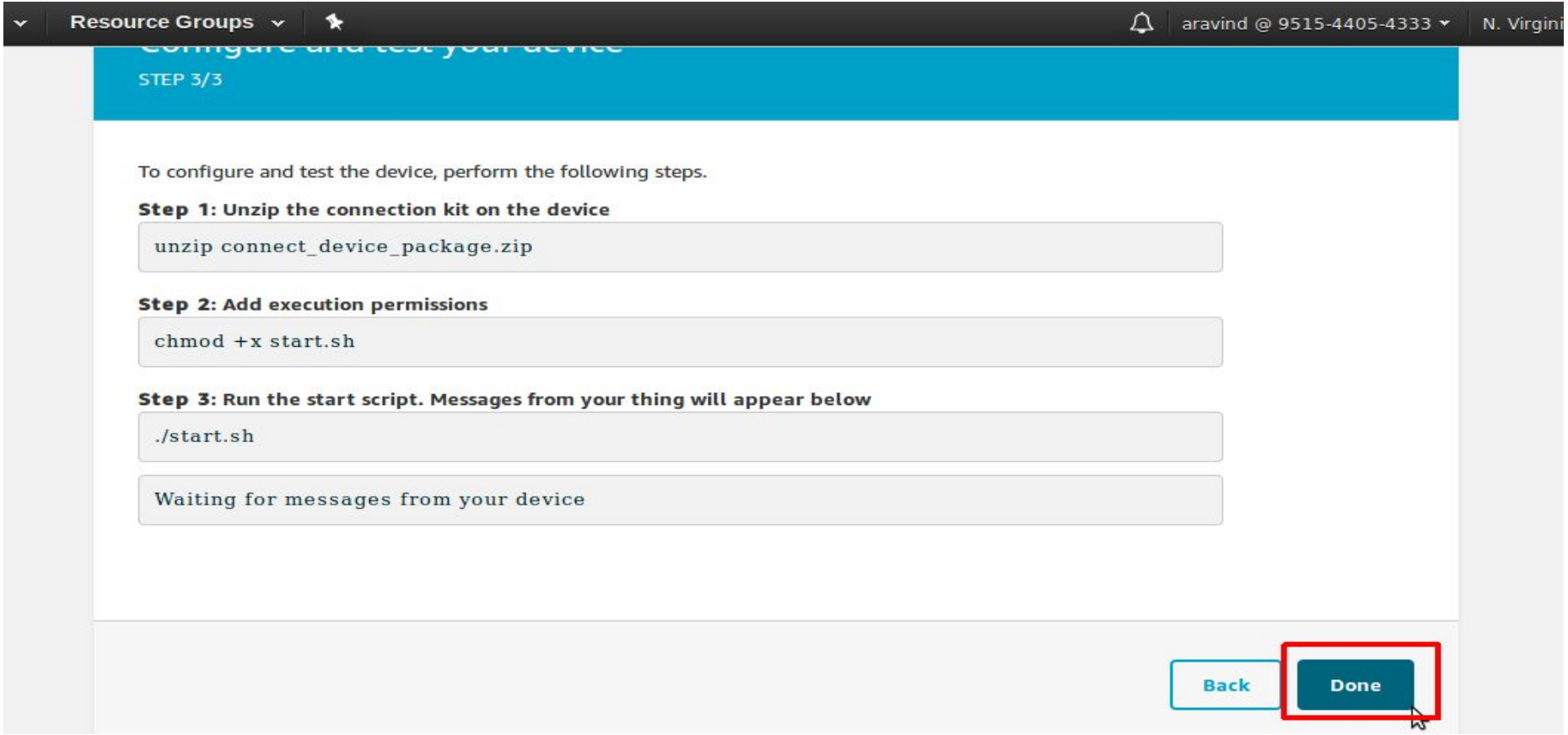
Looking for AWS IoT Device SDKs and documentation?
View AWS IoT Device SDKs

Next

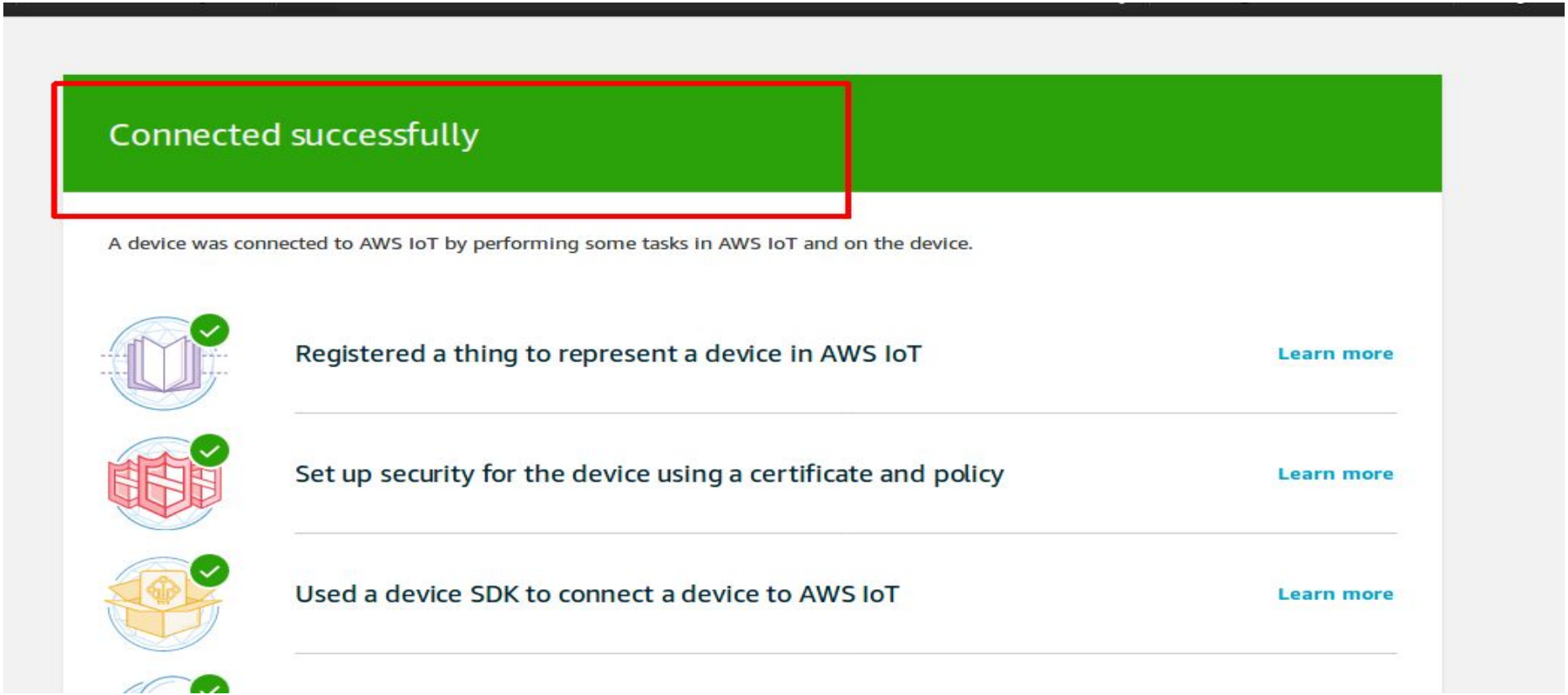
STEP 5 : NOw download the generated connection kit , which contains all the keys and demo code.



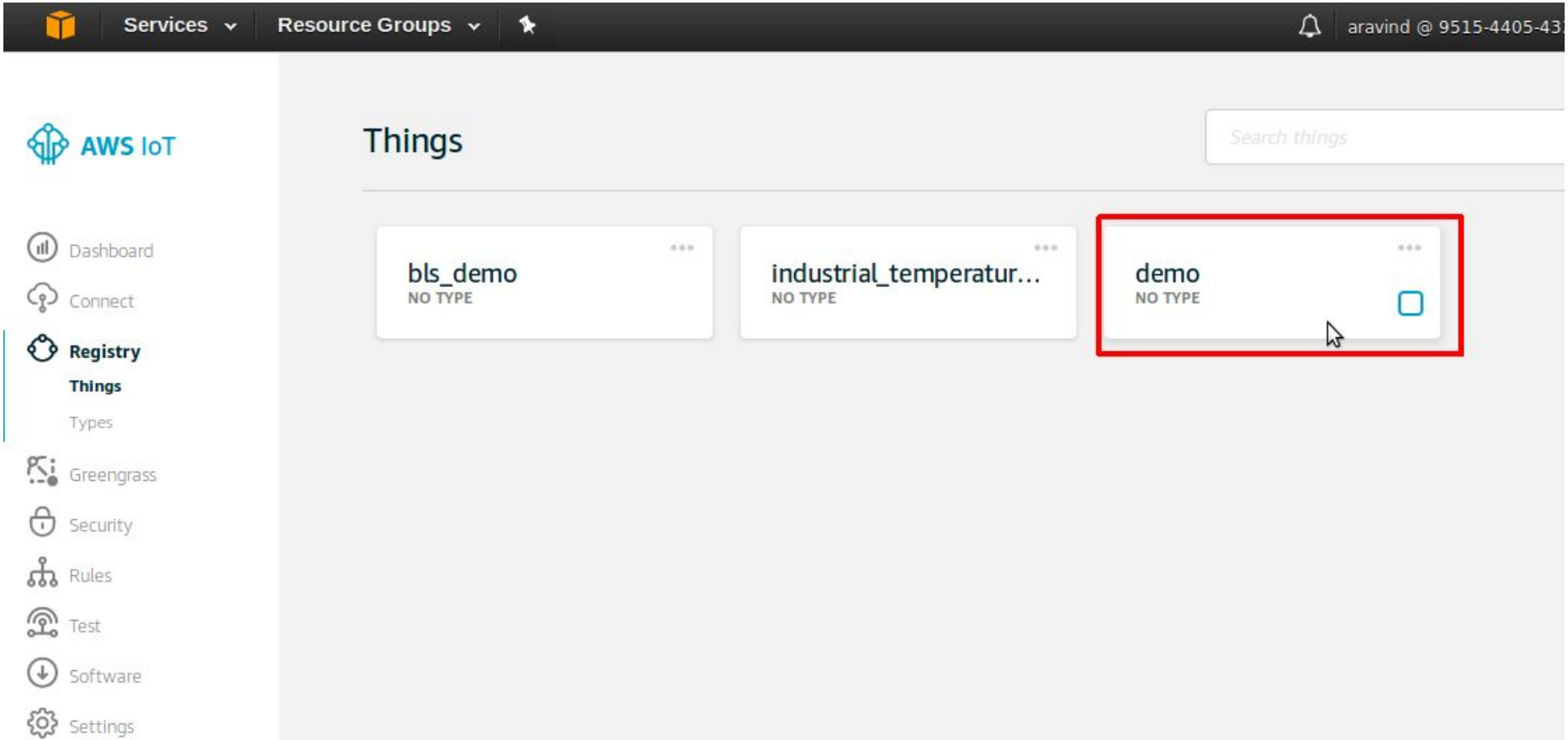
STEP 6 : Follow the steps mentioned to test run the python mqtt publish/subscribe script , later use our python script instead of the demo script generated by default.



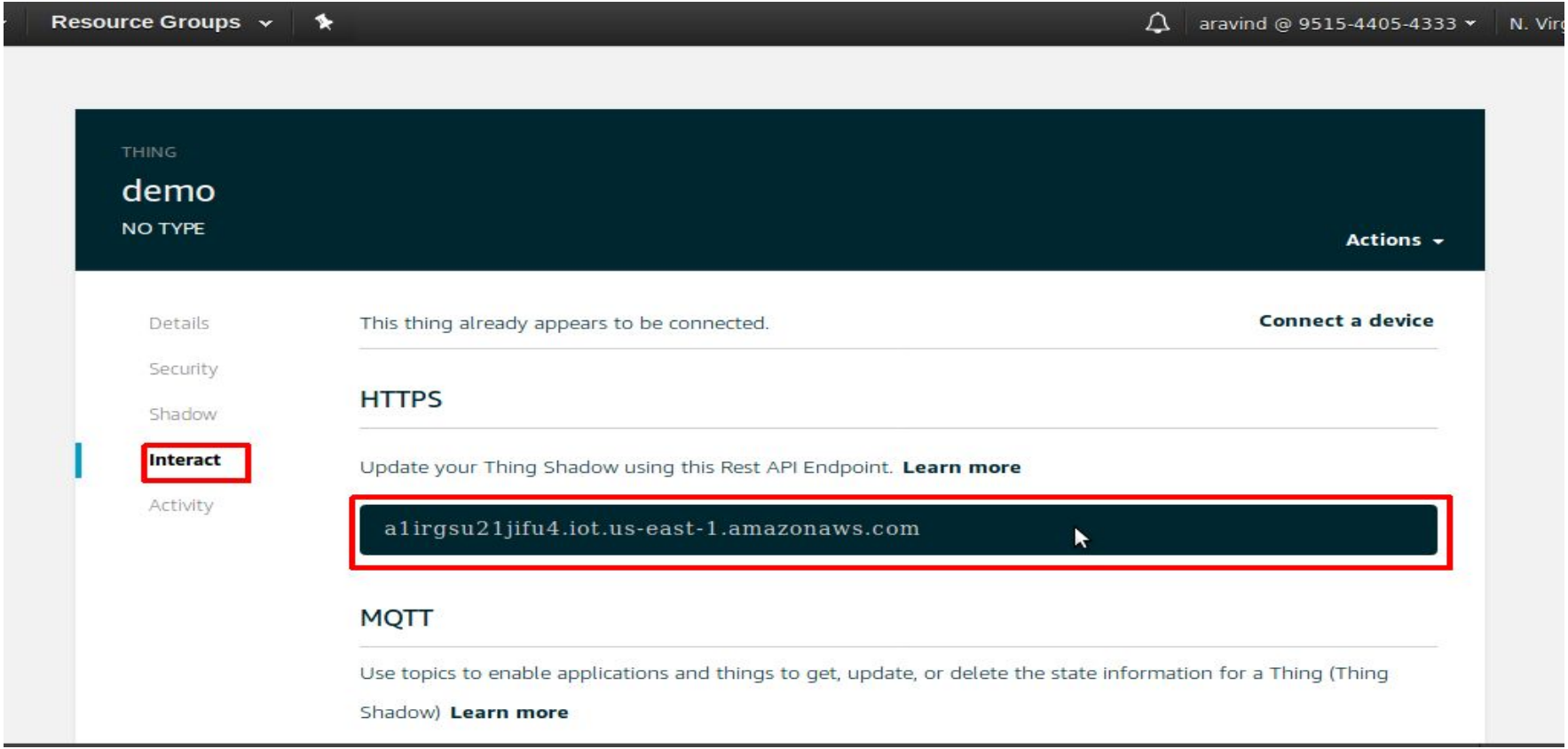
STEP 7 : On successful completion of thing (device) creation , you'll get a success notification as given below



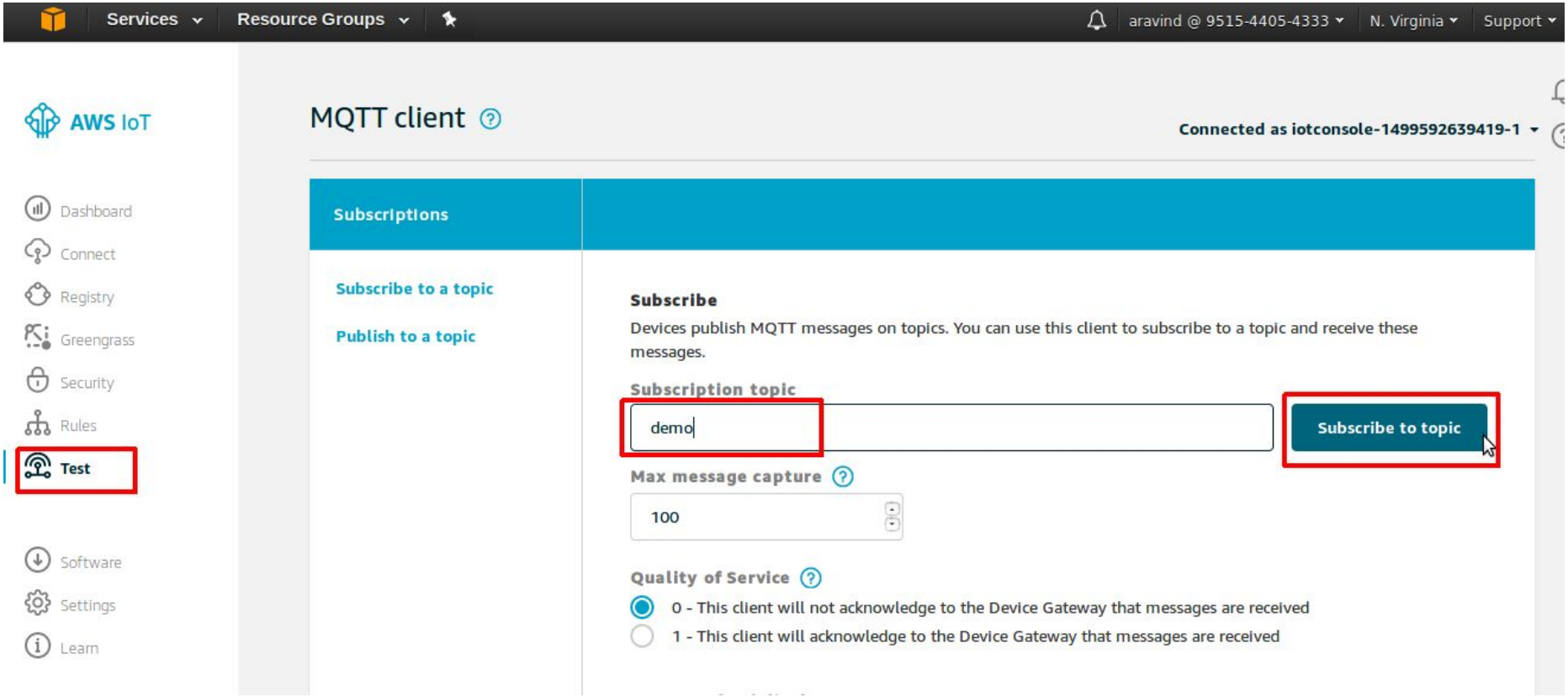
STEP 8 : Select the created device to see for the device details like policies ,shadow , certificates and endpoint id.



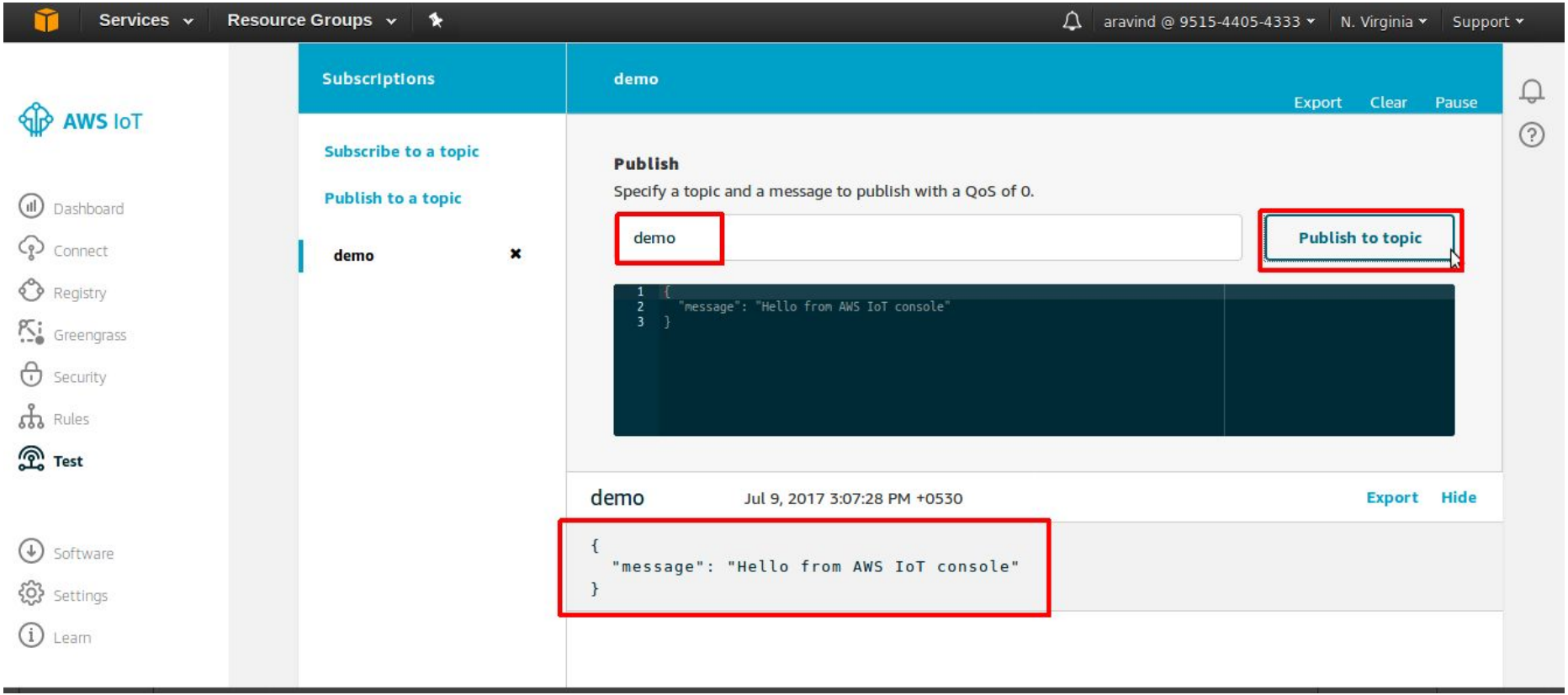
STEP 9 : Here is the device End point id , copy it in a file , which we'll be using in our web program



STEP 10 : To test the created device , click on the TEST option in the side panel , type your topic and click subscribe to topic button

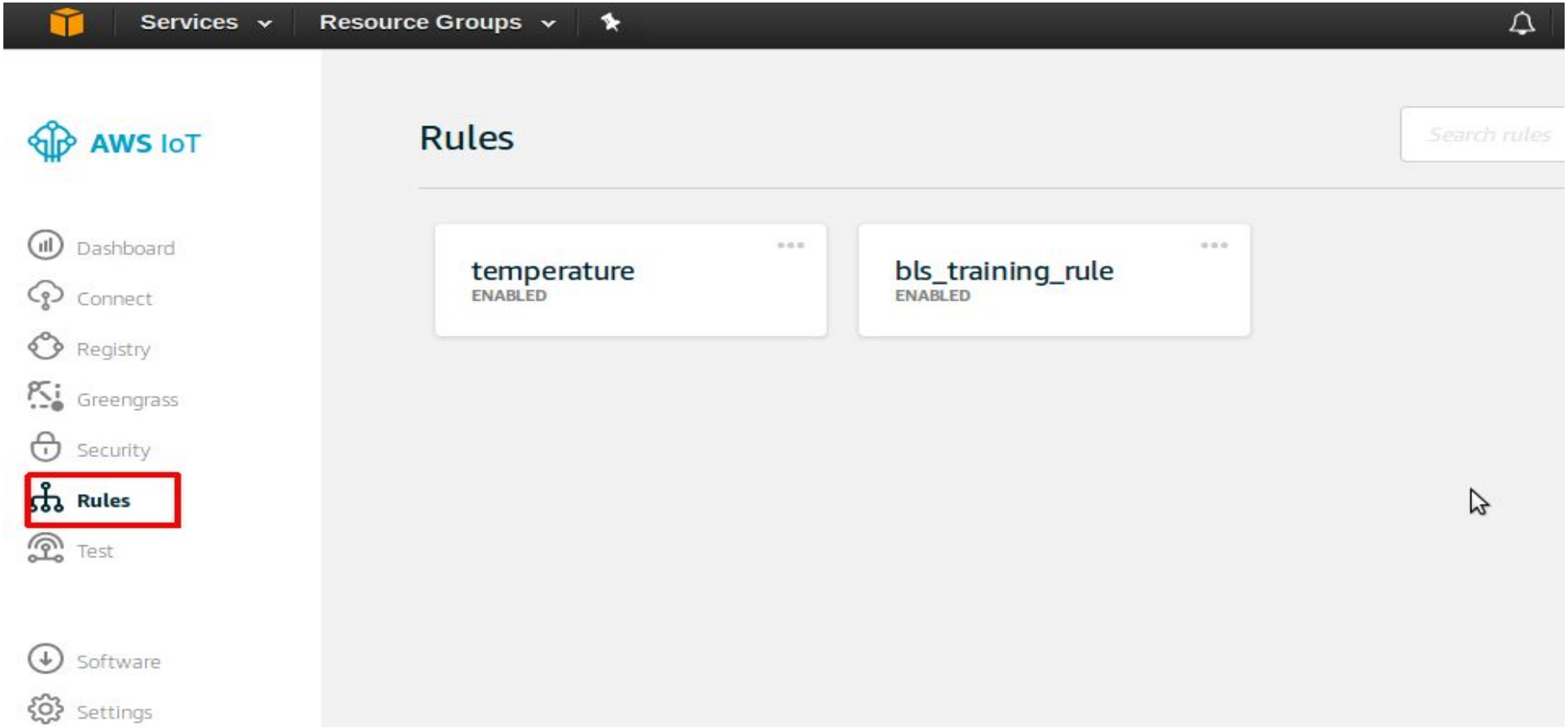


STEP 11 : Now put some demo data in the text area and click publish to topic button , you can see the message in the response display area.

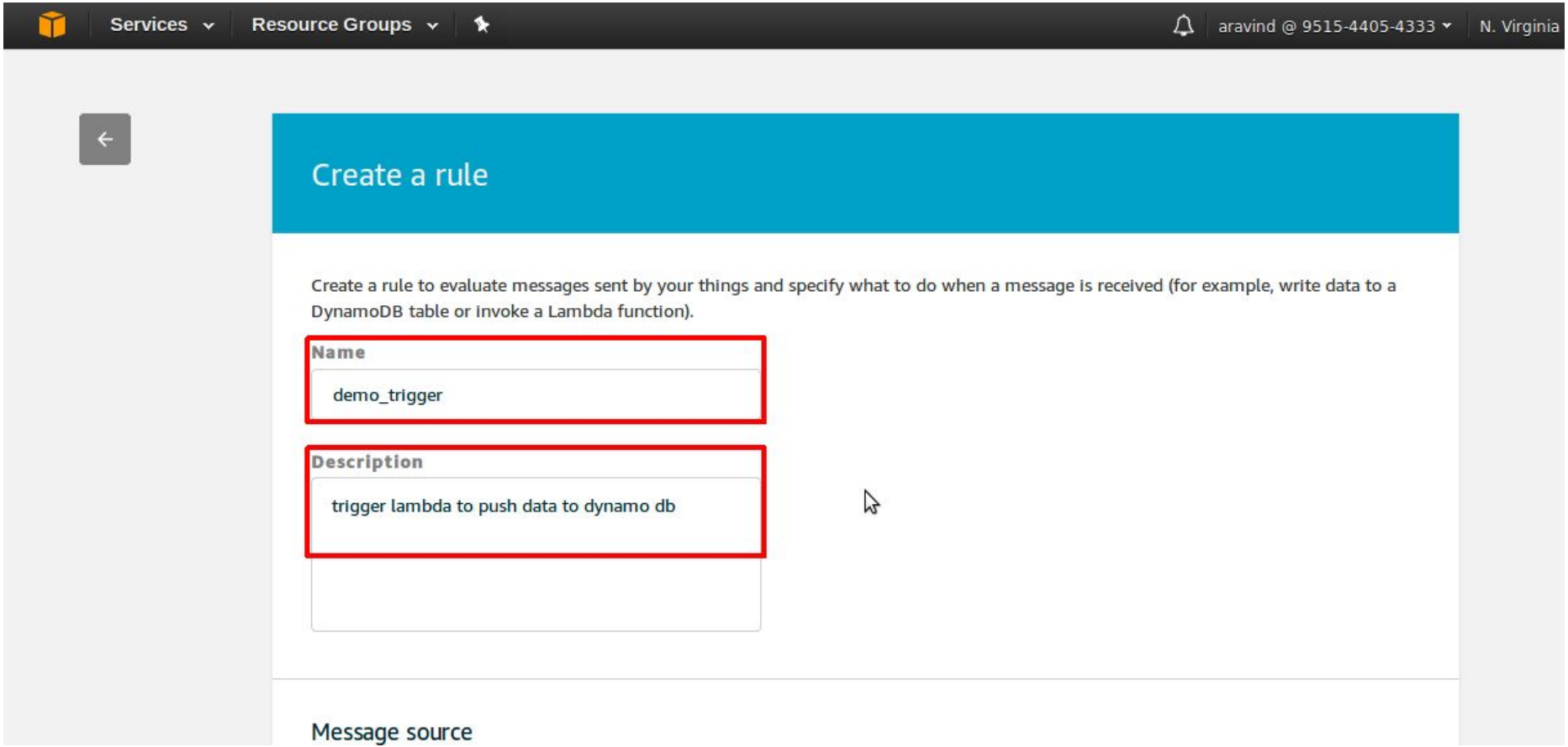


LAMBDA Trigger

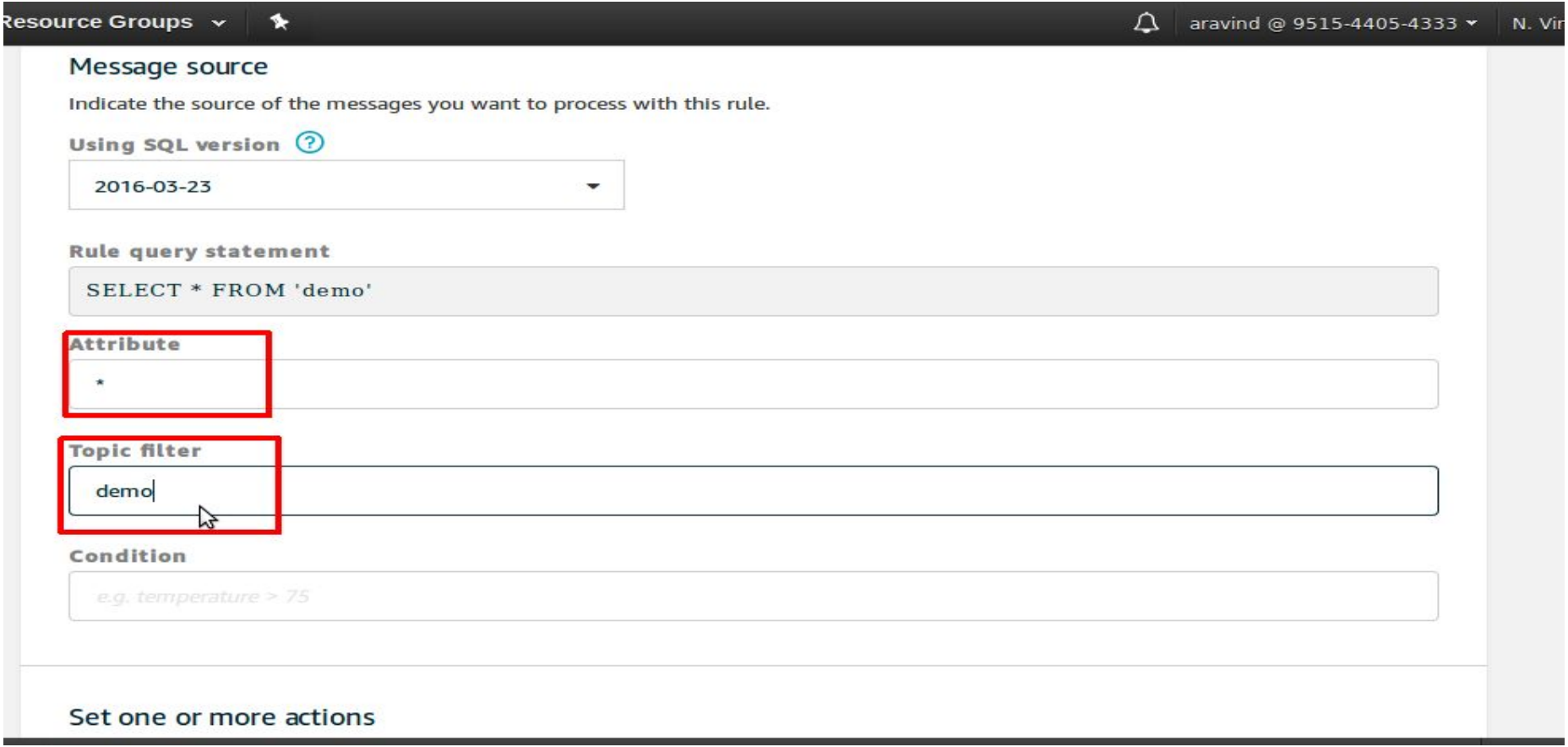
STEP 1 : Click on the rule tab in the side panel and click on the create button to your Right top corner.



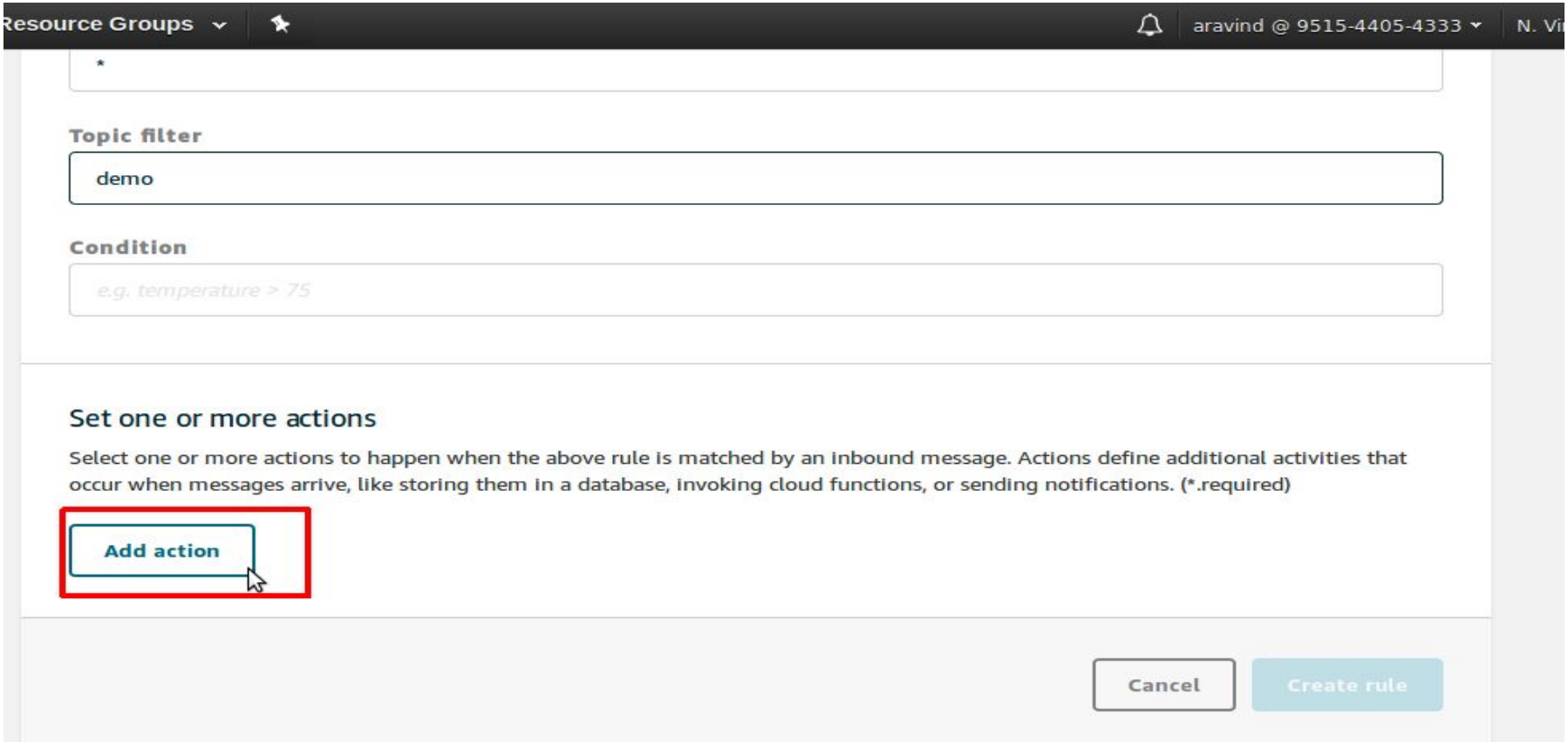
STEP 2 : Give a proper name to the rule and provide the clear description to your lambda trigger rule.



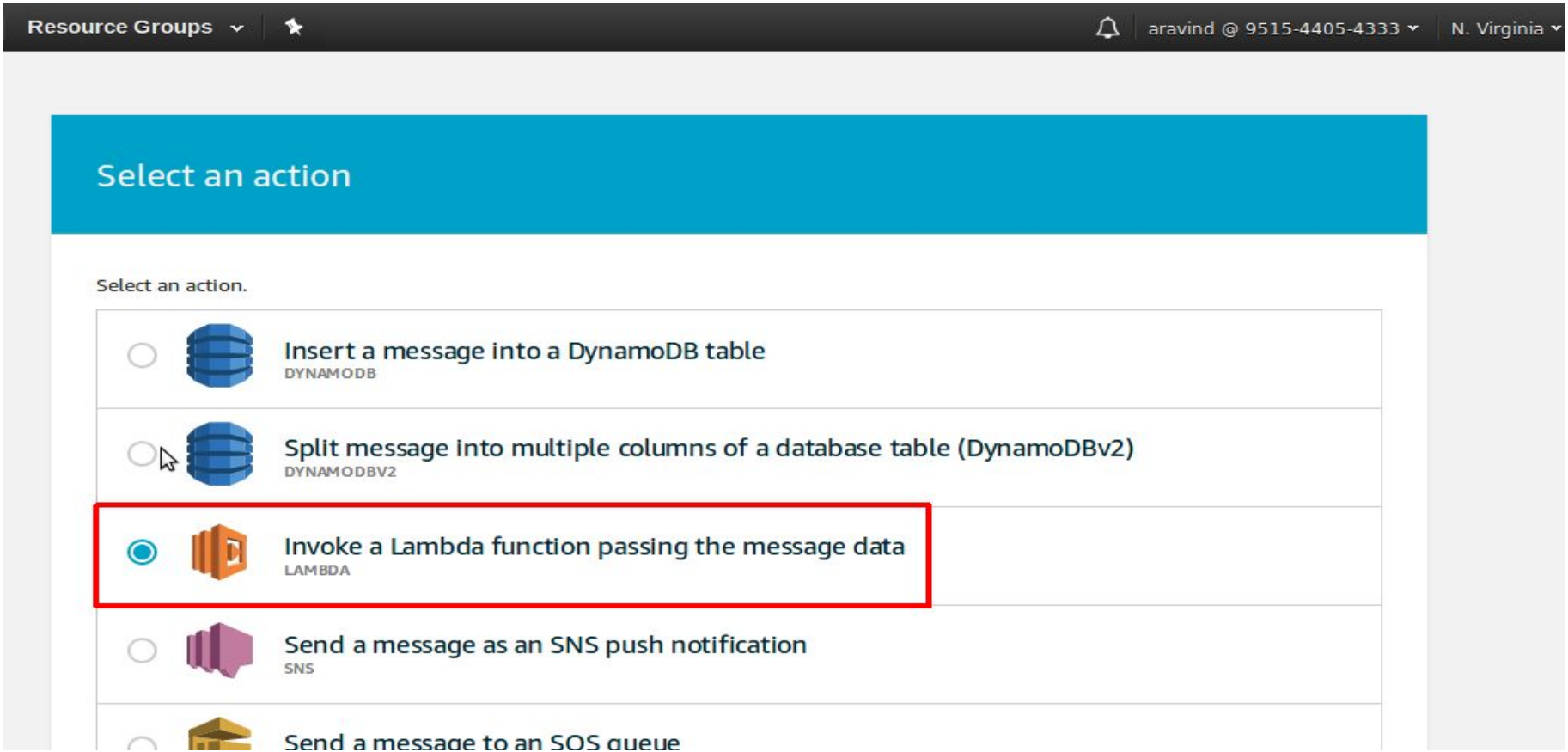
STEP 3 : To enable trigger to a particular topic and all message in that topic , select the wildcard operator for attribute and provide your topic to which trigger should be enabled.



STEP 4 : To add an action , ie , aws iot on receiving message which service to invoke. Click on Add action button.



STEP 5 : Select Invoke lambda function and click configure action button.



Send messages to an Amazon Kinesis Firehose stream
AMAZON KINESIS FIREHOSE

Sends message data to CloudWatch
CLOUDWATCH METRICS

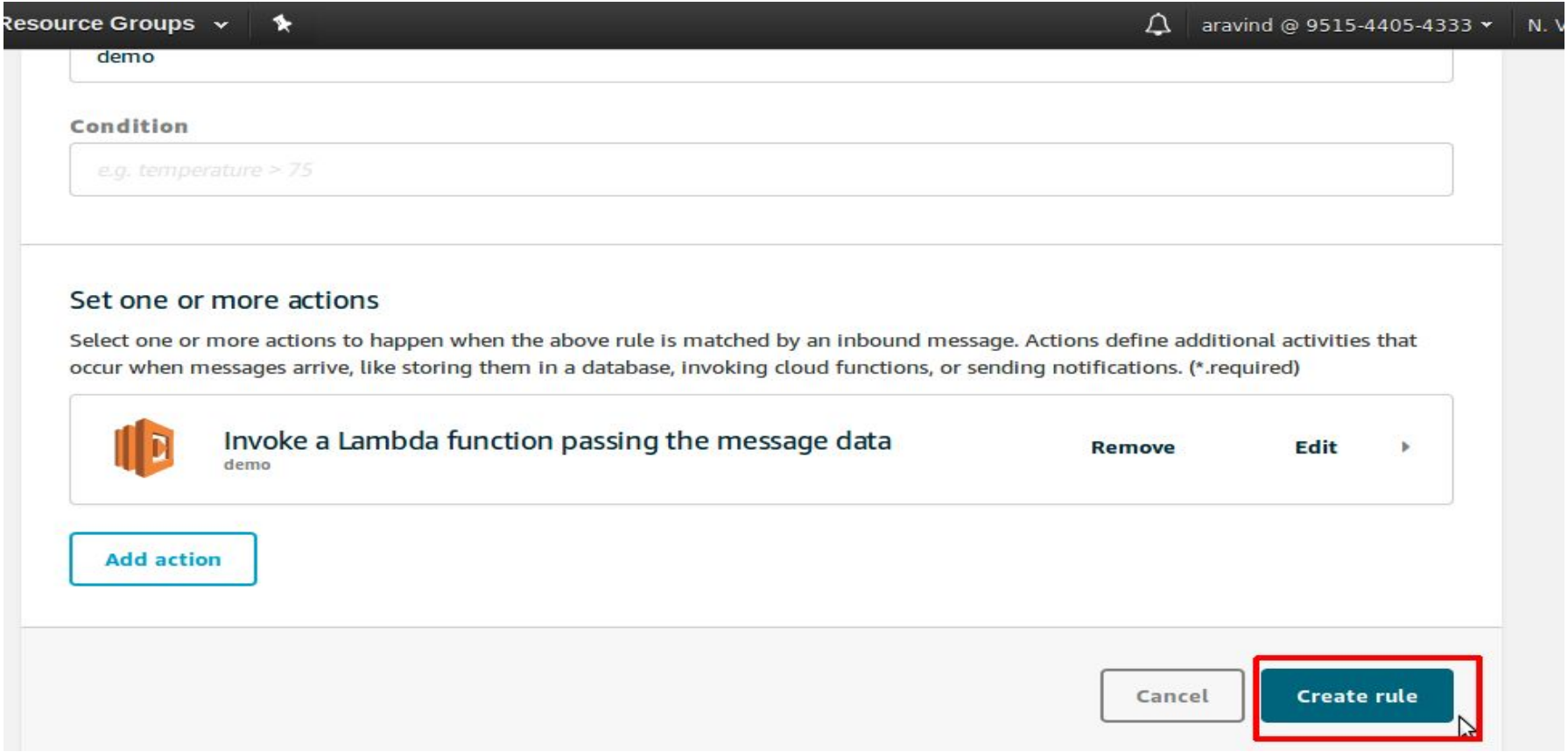
Change the state of a CloudWatch alarm
CLOUDWATCH ALARMS

Send messages to the Amazon Elasticsearch Service
AMAZON ELASTICSEARCH

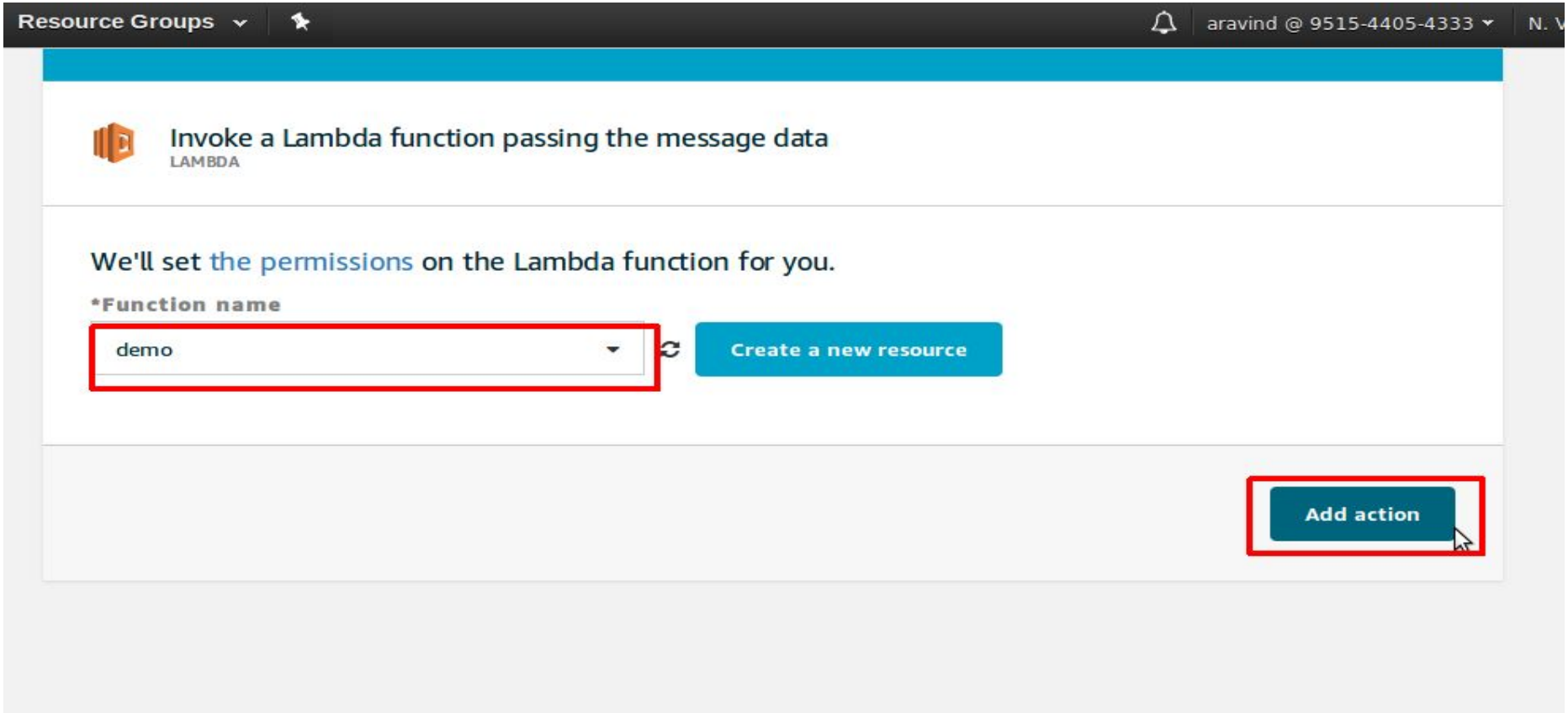
Send message to a Salesforce IoT Cloud Input Stream
SALESFORCE IOT CLOUD

Configure action

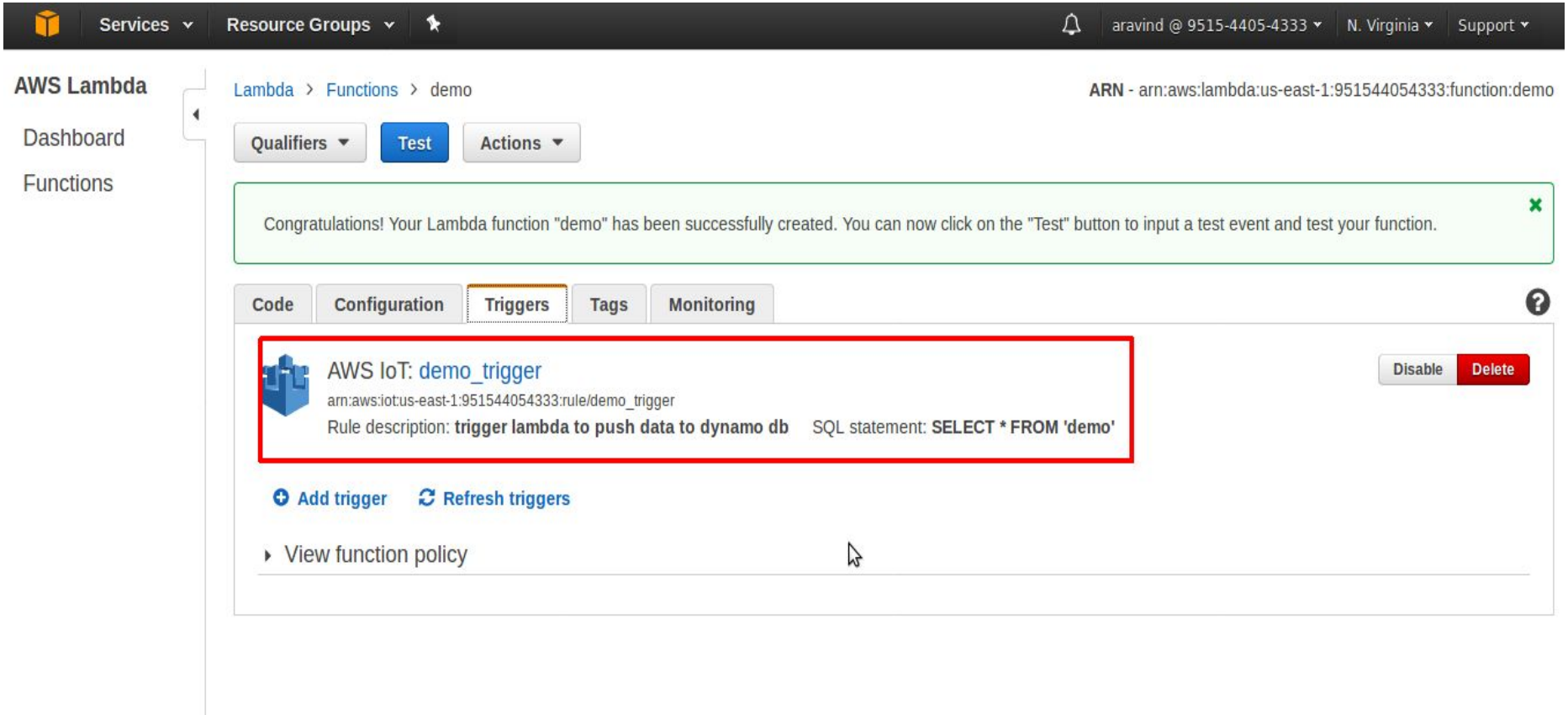
STEP 6 : Here our lambda trigger is added to the action , now click on the Create rule button to create our custom rule.



STEP 7 : Now select the lambda function which we have created for our demo purpose and click Add action button.

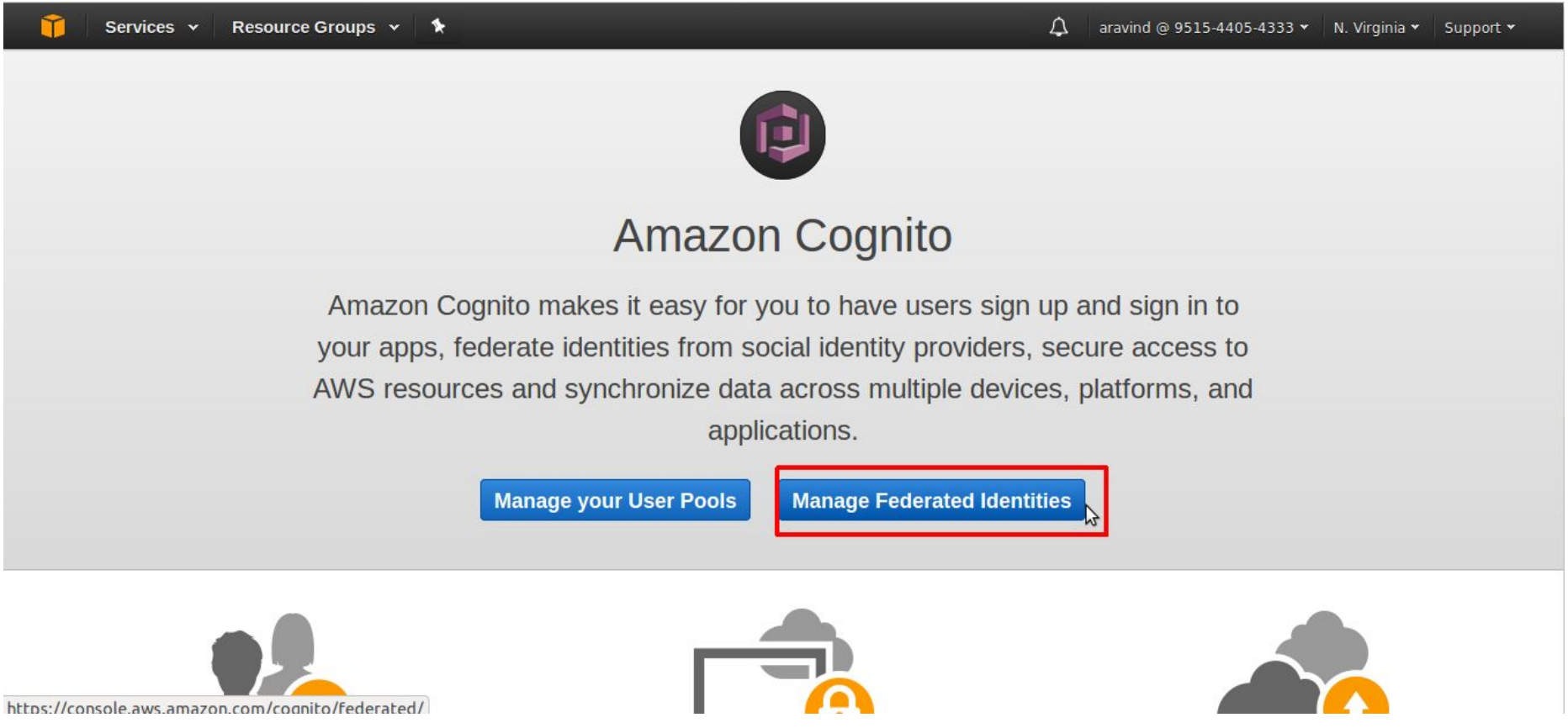


STEP 8 : Open the lambda service and open your lambda program, under trigger tab you can see the added trigger rule in the previous step.

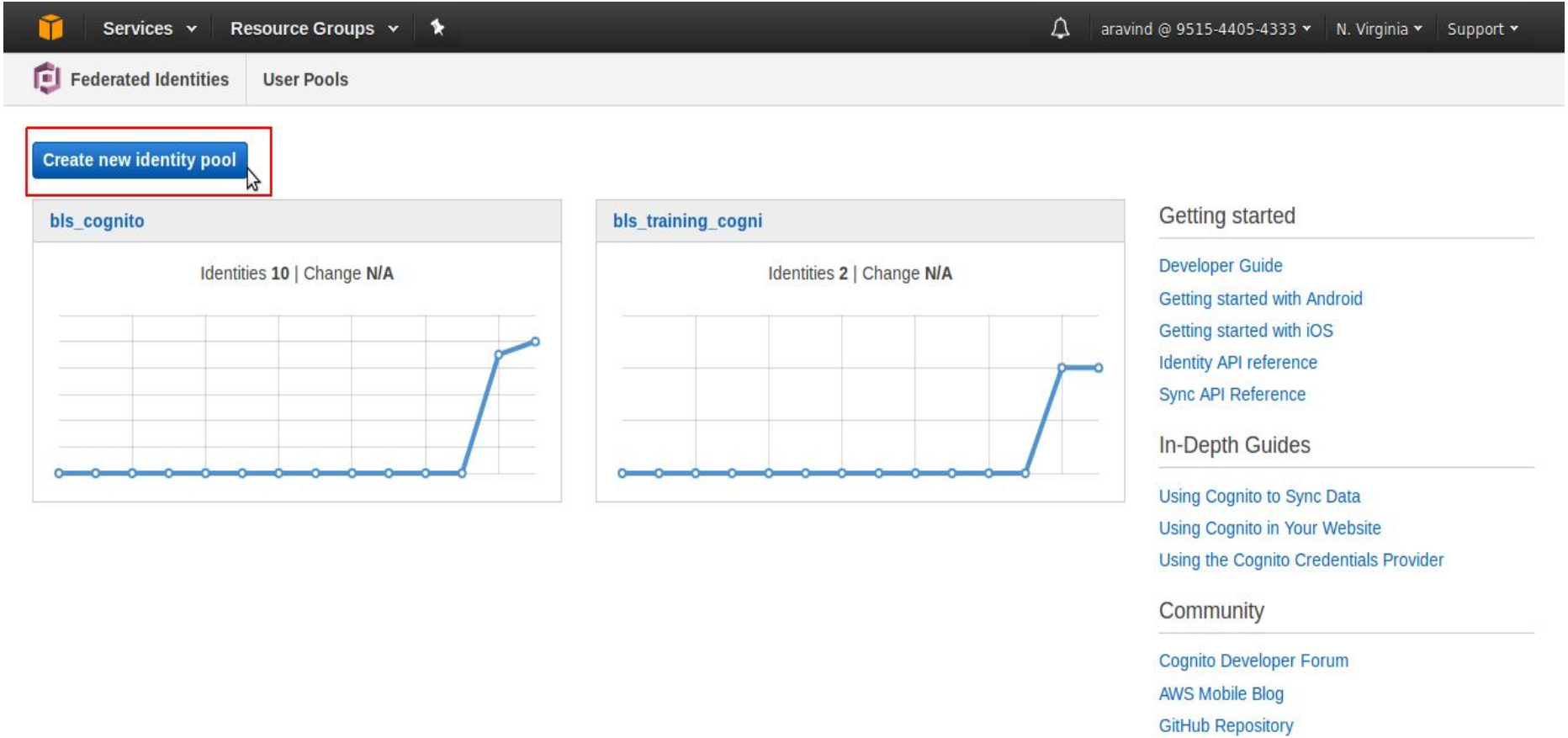


COGNITO Identity pool

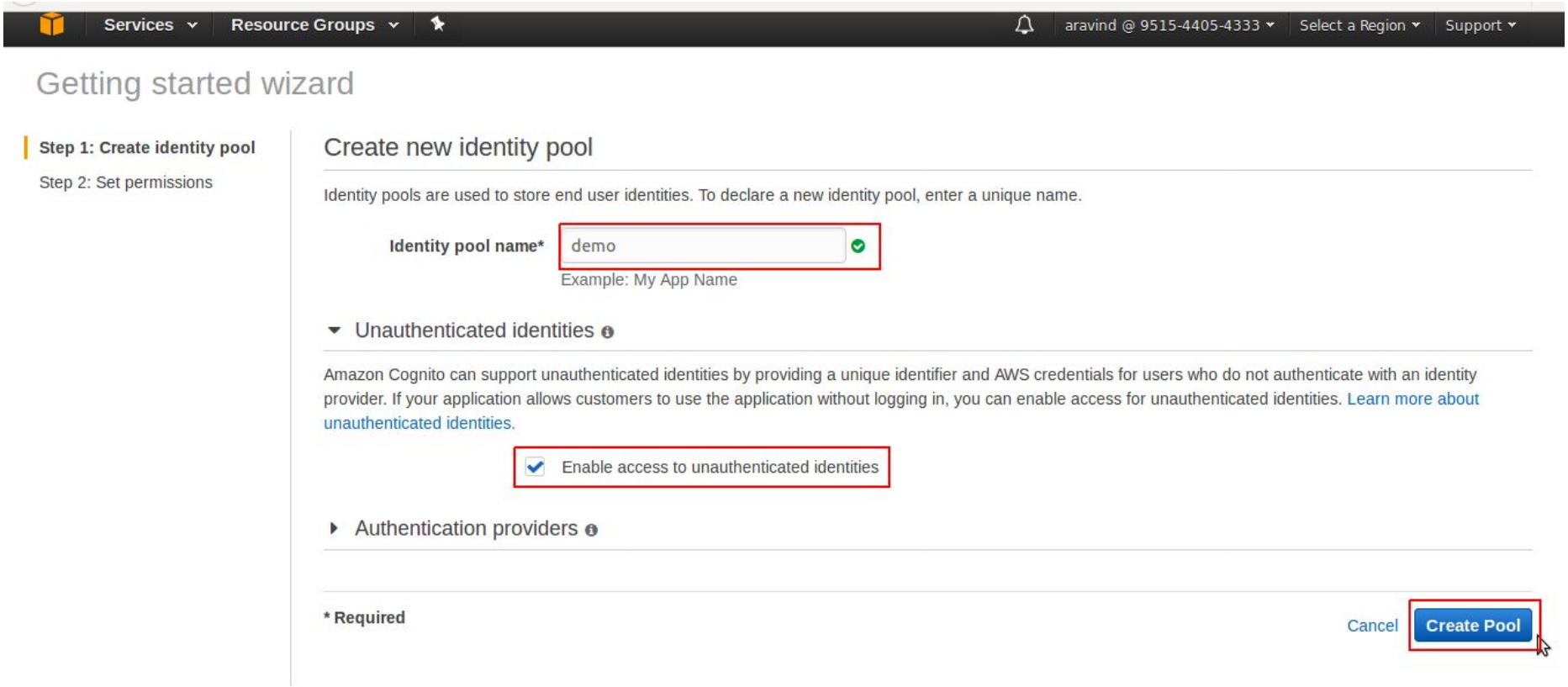
STEP 1 : Open the Cognito pool for creating unauthorized user access to web application .



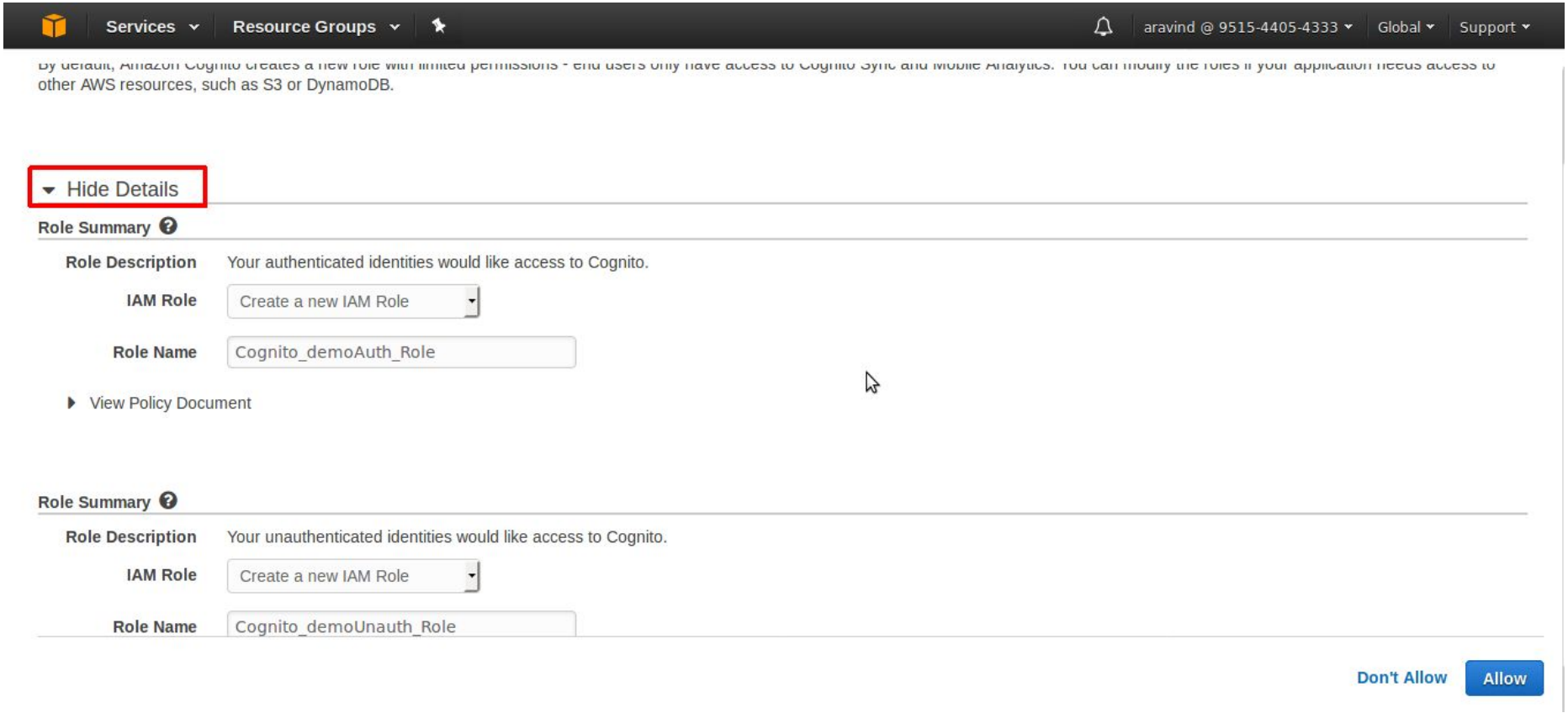
STEP 2 : Click on the create new identity pool button.



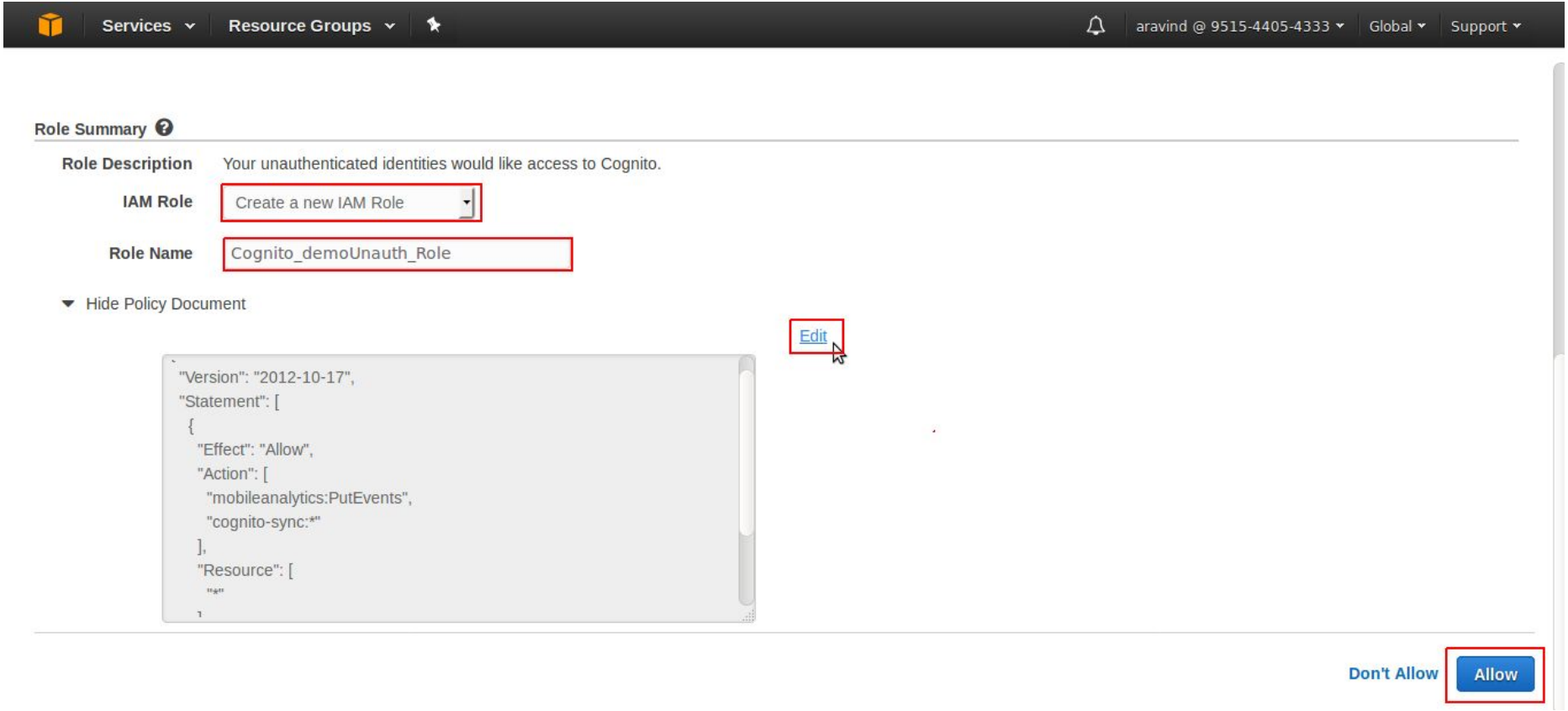
STEP 3 : Give a identity pool name and select enable access to unauthorized users and click create pool button , which creates a identity pool.



STEP 4 : Click the Details tab to see the IAM Role which is to be created for our cognito user pool. Two Role summary will be seen.

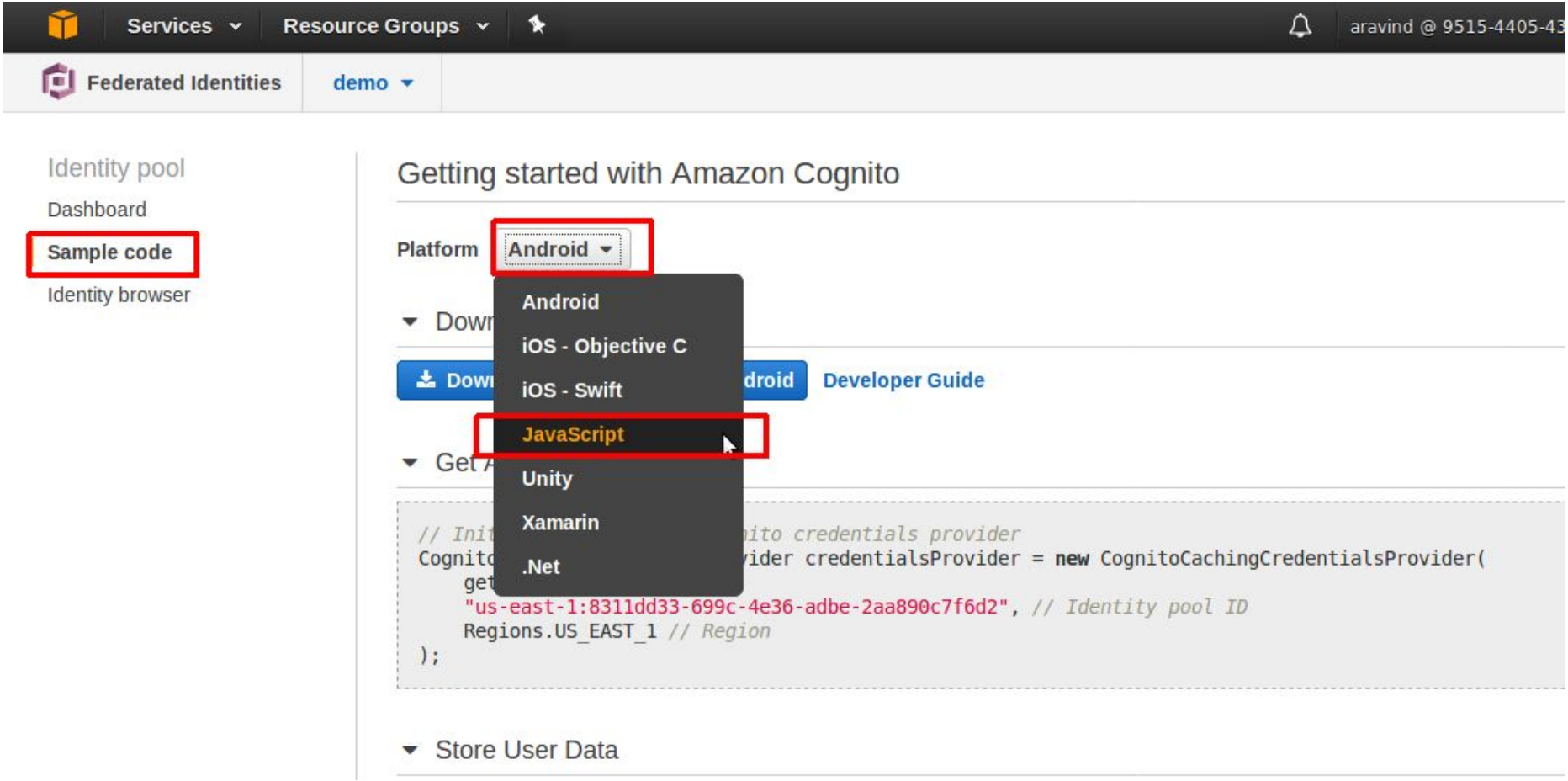


STEP 5 : In the unauthenticated identity role , click edit option and provide access permission to aws iot and dynamo db.Use the below given policy template to provide permission to those services.

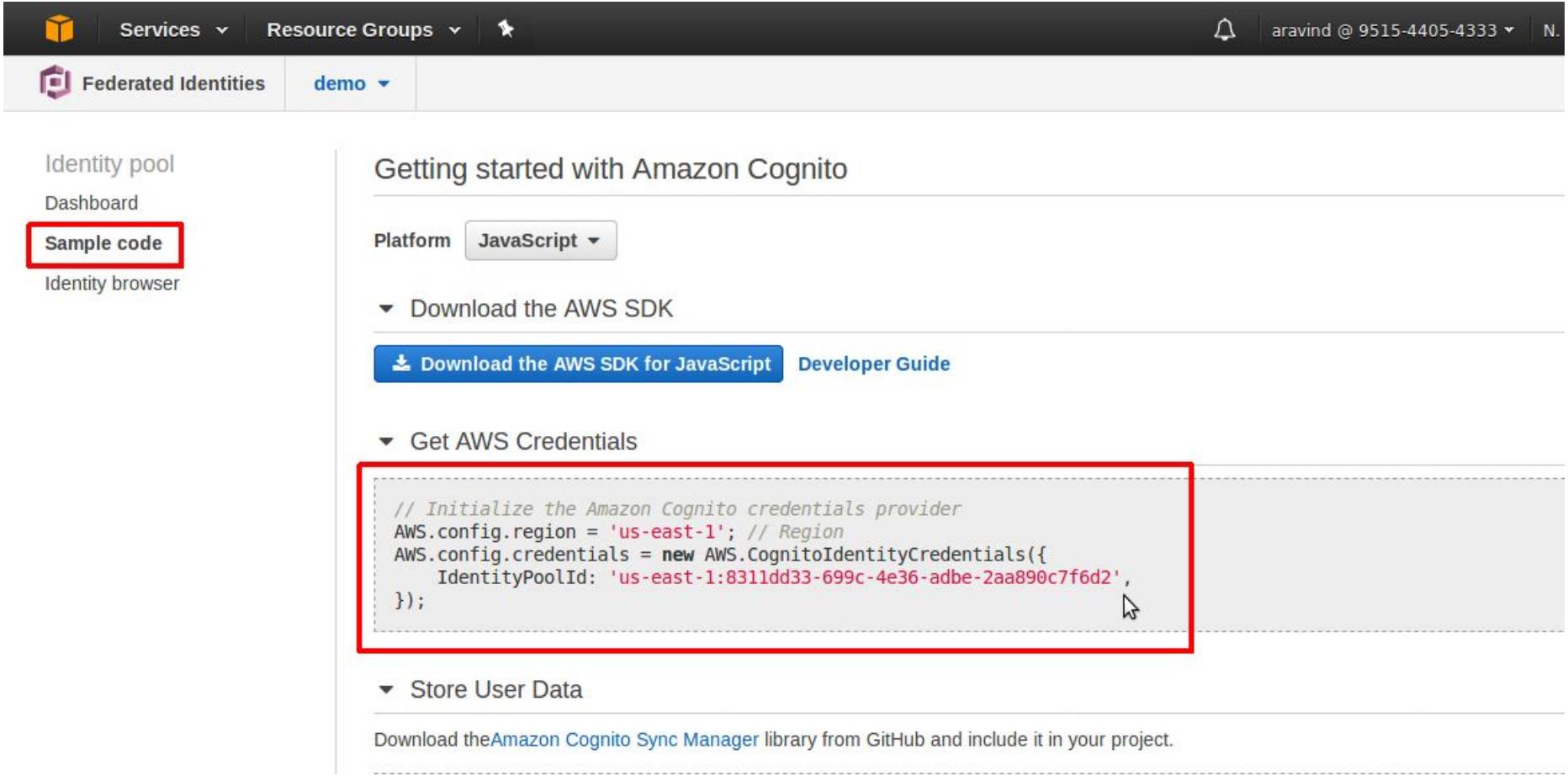



{
"Version": "2012-10-17",
"Statement": [
{
"Effect": "Allow",
"Action": [
"iot:Connect",
"iot:Receive",
"iot:Subscribe",
"iot:Publish",
"dynamodb:*",
"mobileanalytics:PutEvents",
"cognito-sync:*"
],
"Resource": [
"*"
]
}
]
}

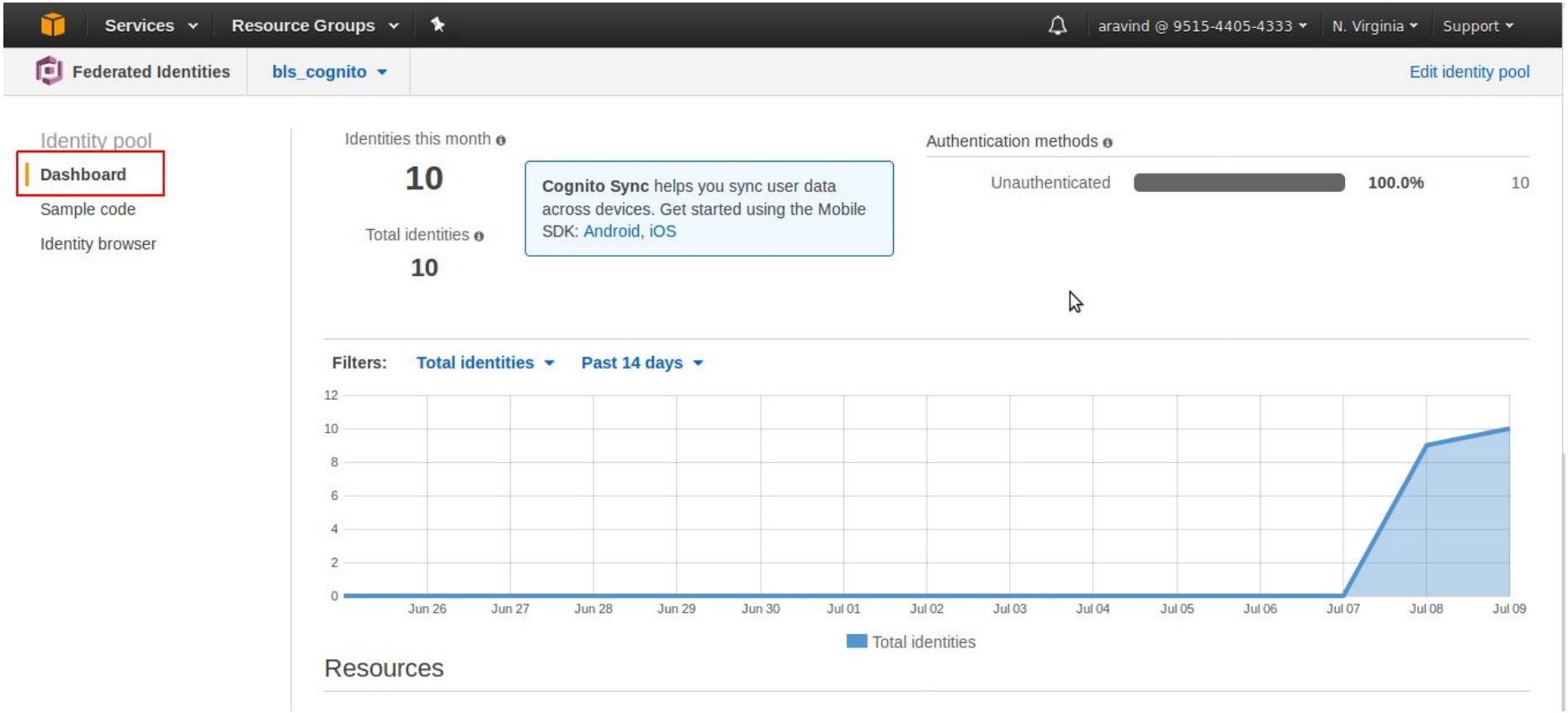
STEP 6 : Select Javascript sdk to get the pool id and region details.



STEP 7 : Copy the aws credentials , which we'll be using in our web application code.



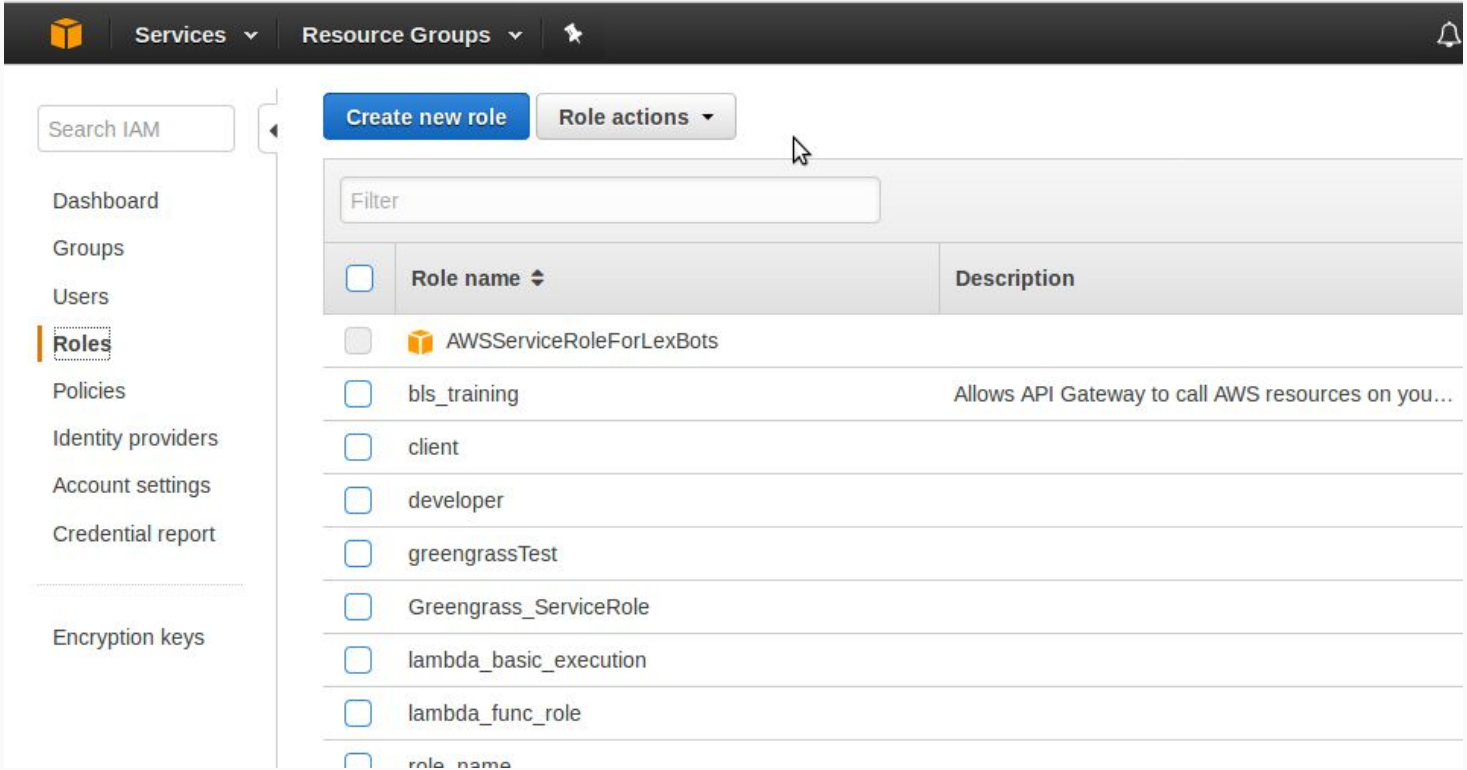
STEP 8 : Here in the dashboard , you can see the number of unauthenticated user identities are logged in to the aws services.



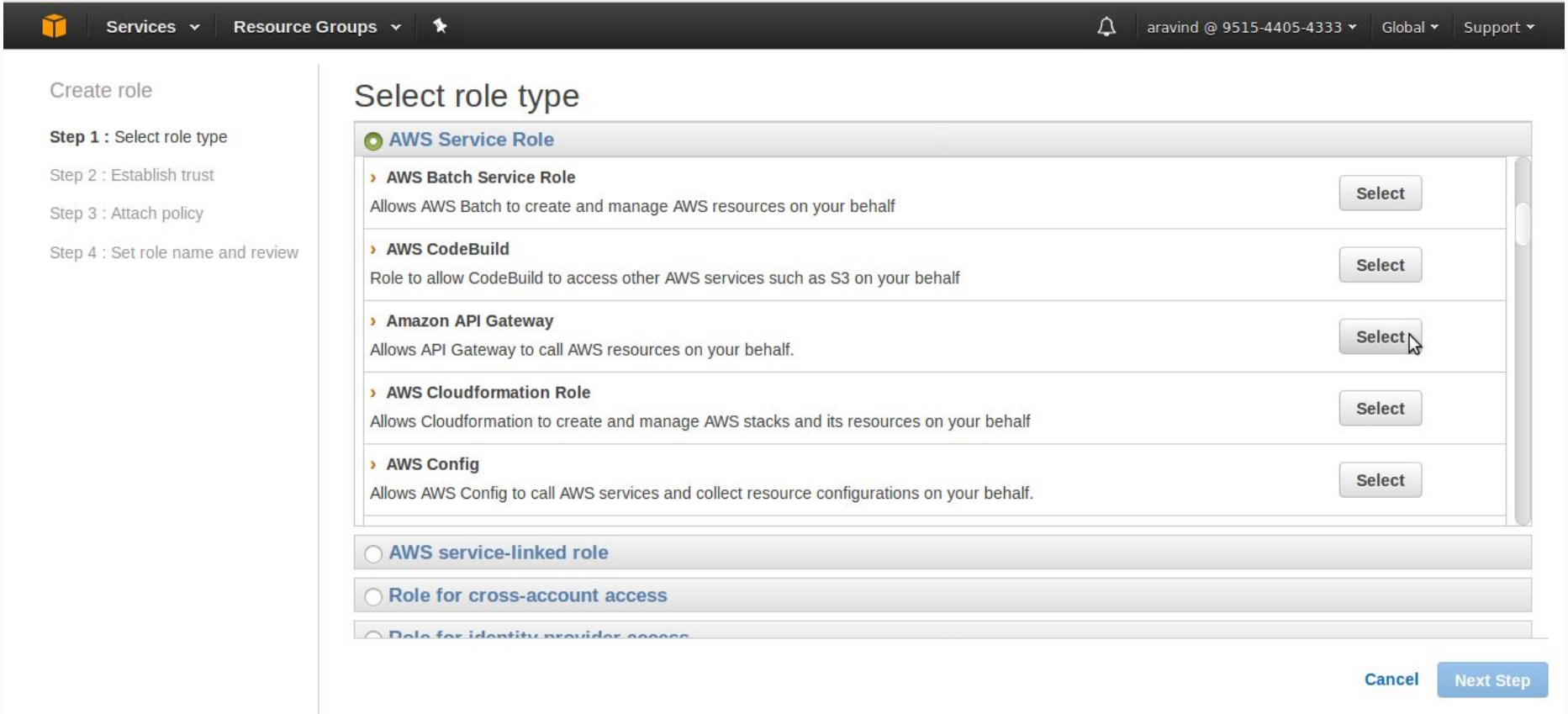
API gateway creation for Dynamo db

Creating Policy and Role for the Dynamo db table for access through API Gateway

STEP 1: Open IAM service , click role in the side panel and select create new role.

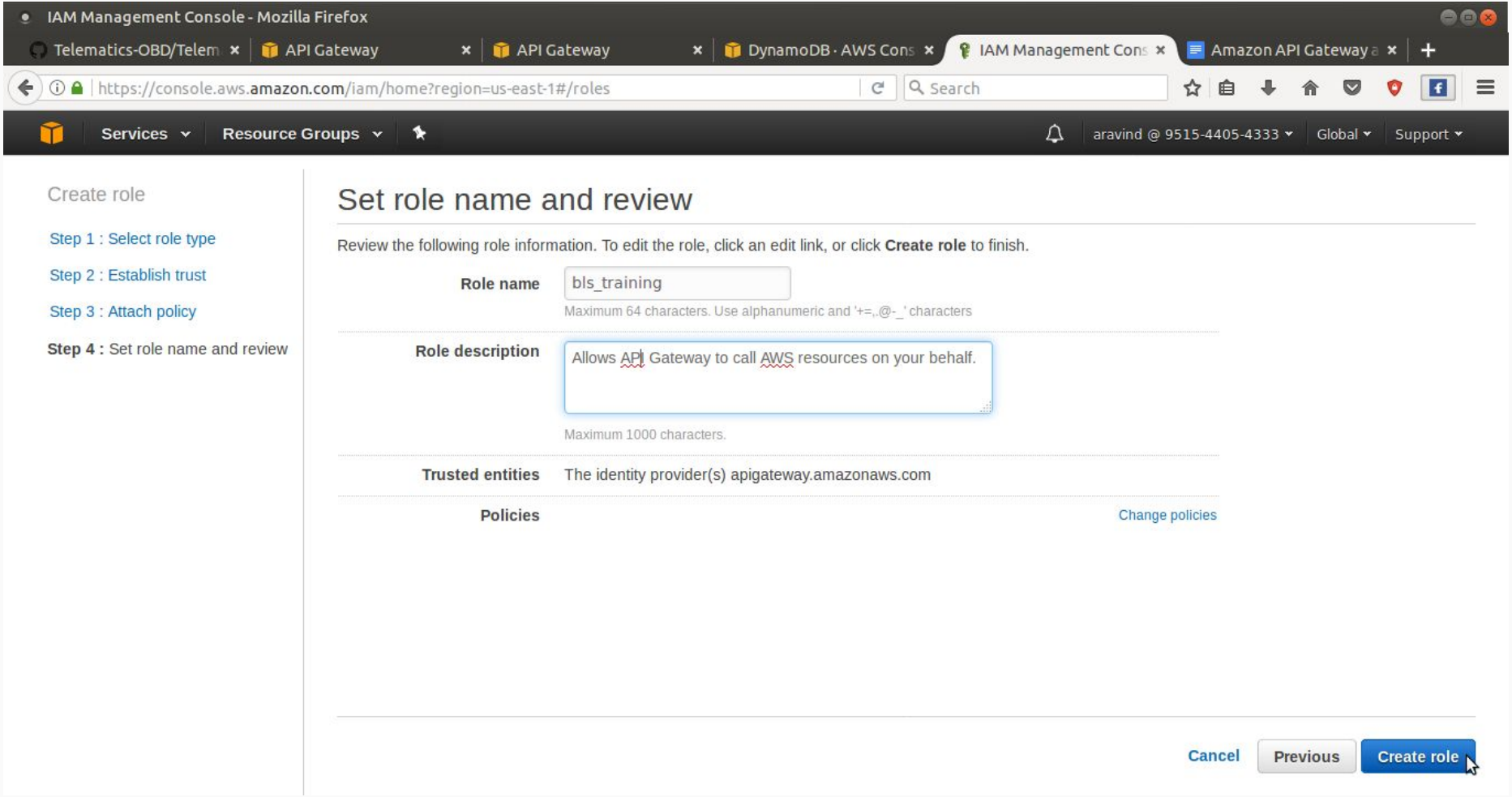


STEP 2 : Select role type to Amazon API Gateway and select Next Step.

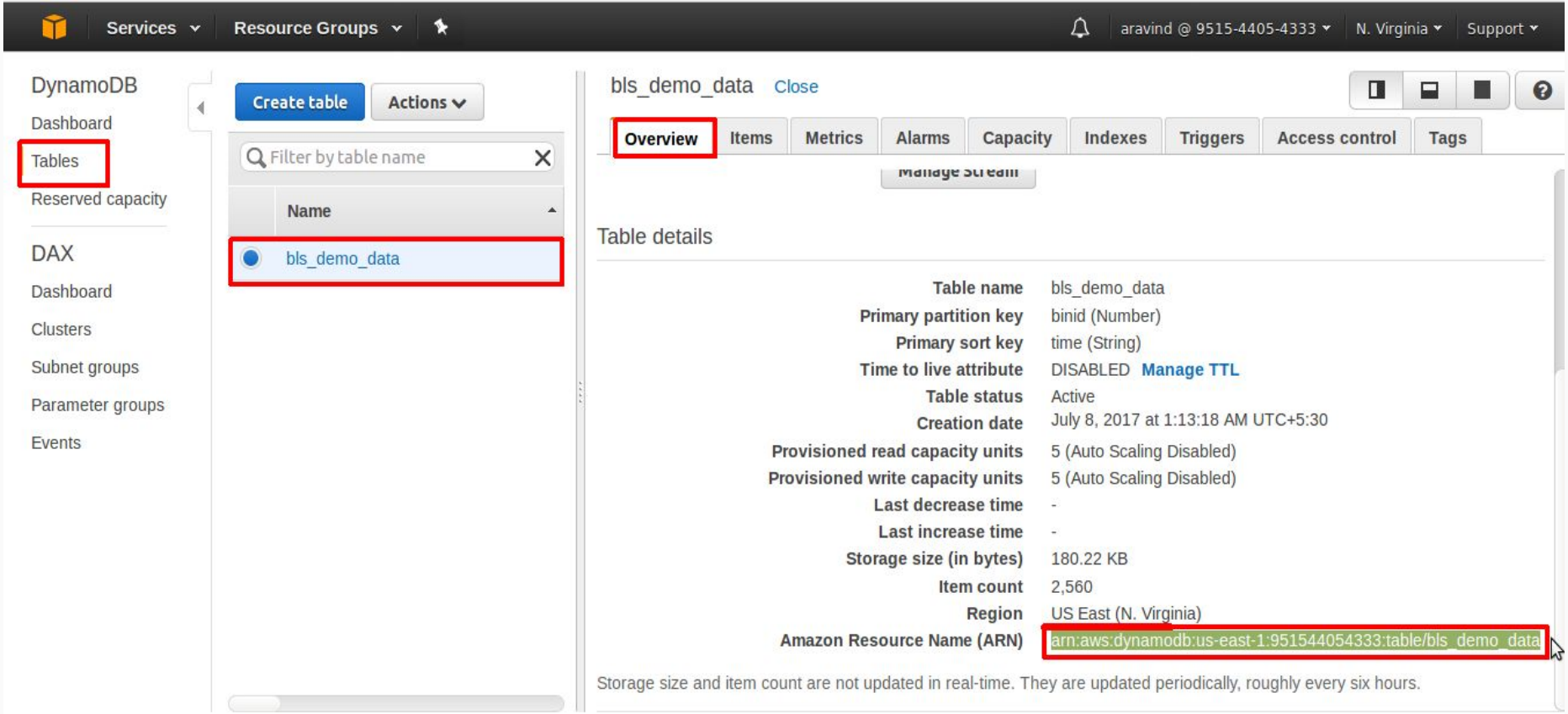


STEP 3 : Skip attach policy which we can add it in later stage and select Next Step.

STEP 4 : Give a name to the role and select Create Role.


IAM Management Console - Mozilla Firefox
Telematics-OBD/Telem
API Gateway
API Gateway
DynamoDB - AWS Cons
IAM Management Cons
Amazon API Gateway a
https://console.aws.amazon.com/iam/home?region=us-east-1#/roles
Search
Services
Resource Groups
aravind @ 9515-4405-4333
Global
Support
Create role
Step 1 : Select role type
Step 2 : Establish trust
Step 3 : Attach policy
Step 4 : Set role name and review
Set role name and review
Review the following role information. To edit the role, click an edit link, or click Create role to finish.
Role name
bls_training
Maximum 64 characters. Use alphanumeric and '+=,.@-_' characters
Role description
Allows API Gateway to call AWS resources on your behalf.
Maximum 1000 characters.
Trusted entities
The identity provider(s) apigateway.amazonaws.com
Policies
Change policies
Cancel
Previous
Create role

STEP 5 : Goto the dynamodb table and copy the ARN value



STEP 6 : Goto the IAM page from the services and click on the “Policies” and select the “Policy generator”.

STEP 6.1 : Give the following field values

- AWS Service - Amazon DynamoDB
- Actions - Scan/Get/PutItem (Note: selection of action depends on what operations you are going to do in DynamoDB)
- Amazon Resource Name(ARN) - Paste the ARN that you copied from the Step 5
- Finally click on “Add statement” and click “Next”

Create Policy

Step 1 : Create Policy

Step 2 : Set Permissions

Step 3 : Review Policy

Edit Permissions

The policy generator enables you to create policies that control access to Amazon Web Services (AWS) products and resources. For more information about creating policies, see Overview of Policies in Using AWS Identity and Access Management.

Effect Allow ◉ Deny ○

AWS Service Amazon DynamoDB ▾

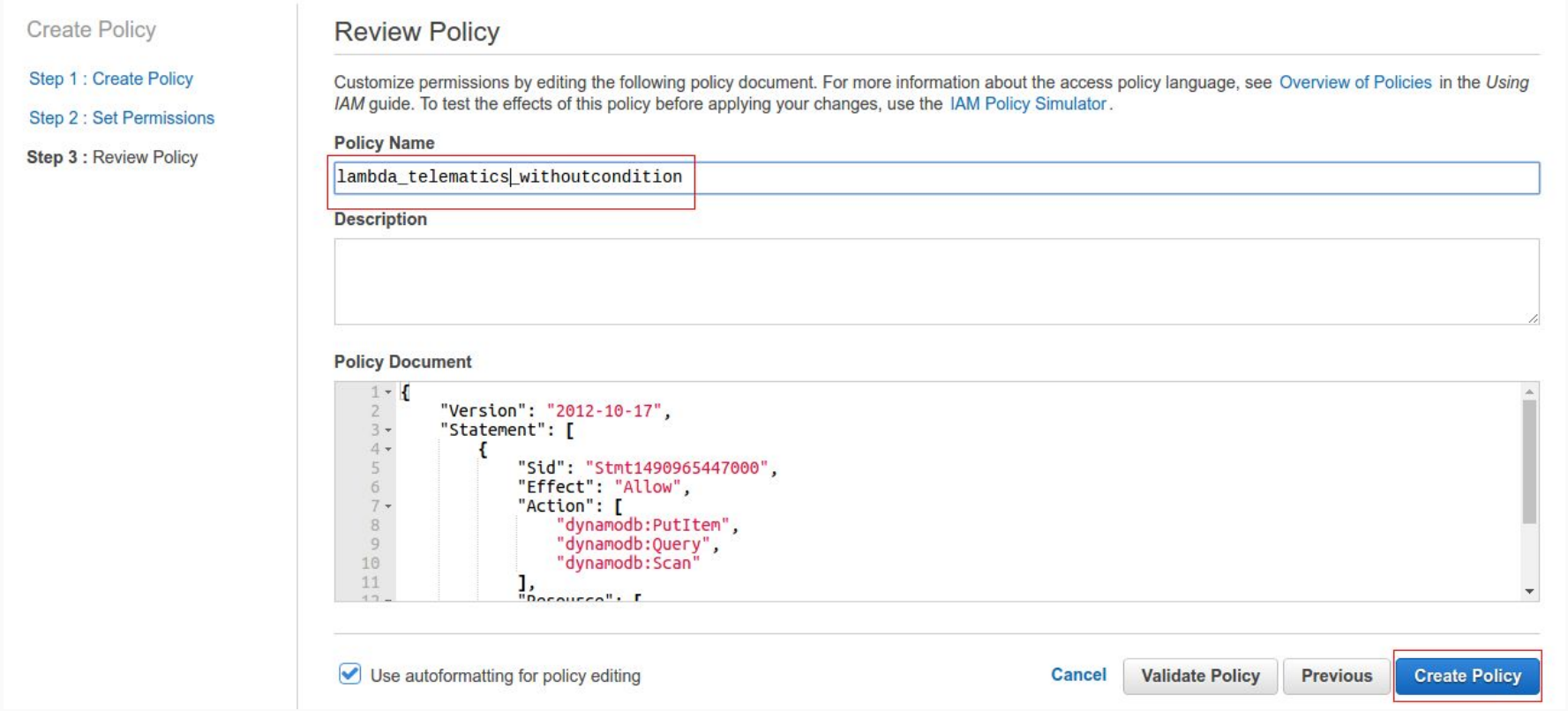
Actions 3 Action(s) Selected

Amazon Resource Name (ARN)

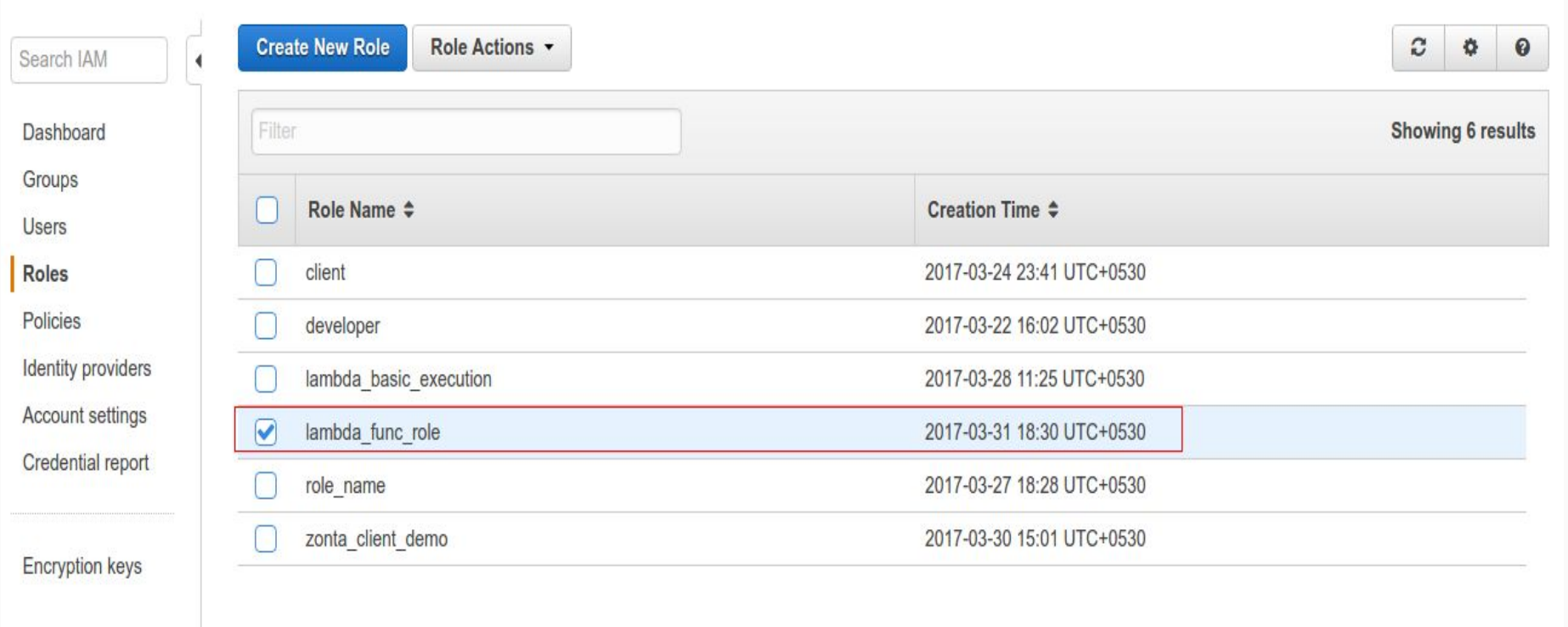
- ☐ PurchaseReservedCapacityOfferings
- ☑ PutItem
- ☑ Query
- ☑ Scan
- ☐ UpdateItem
- ☐ UpdateTable

Cancel Previous Next Step

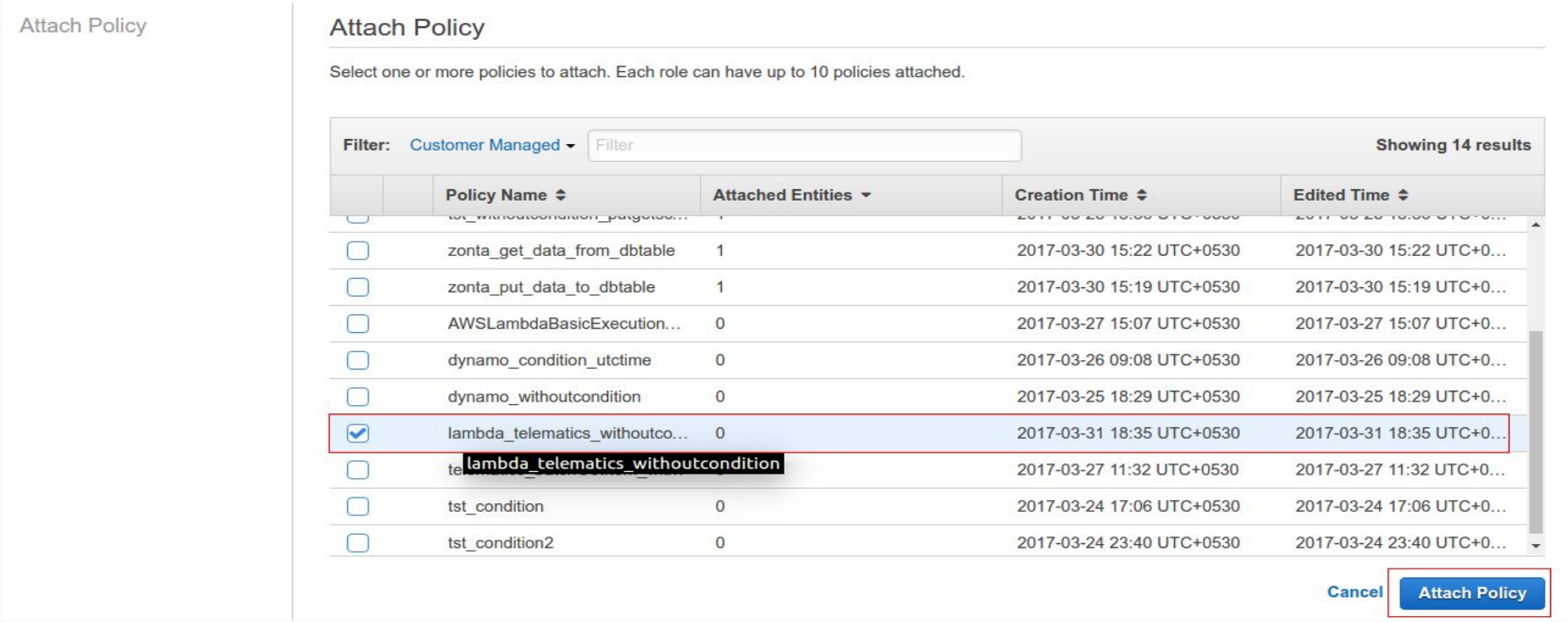
STEP 7 : Give a meaningful name for the policy and select create policy.



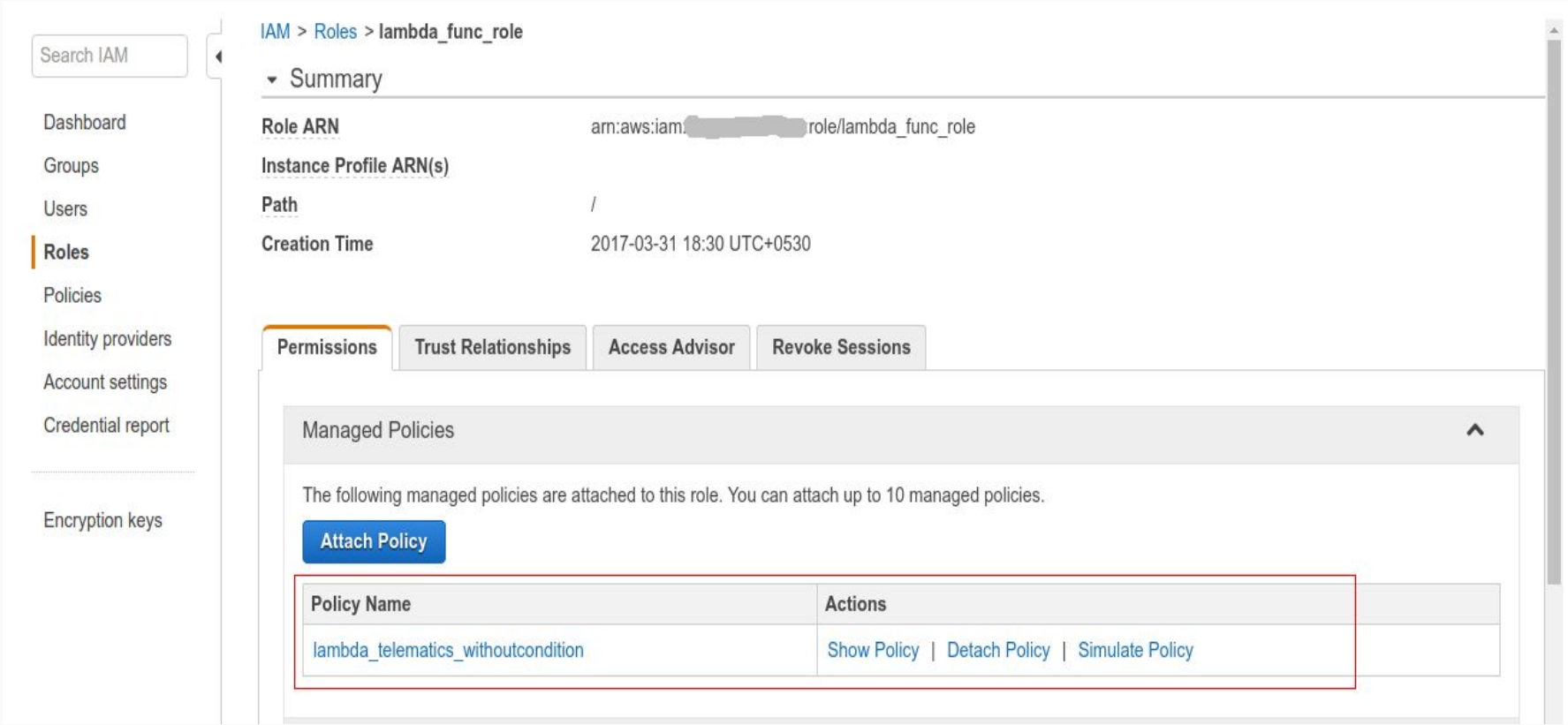
STEP 8 : Once you create the policy goto the Roles in the IAM page and select the Role that you have created earlier.



STEP 9 : Once you open the Role that you have created, Click on the “Attach policy” and select the policy that you have created in step 7



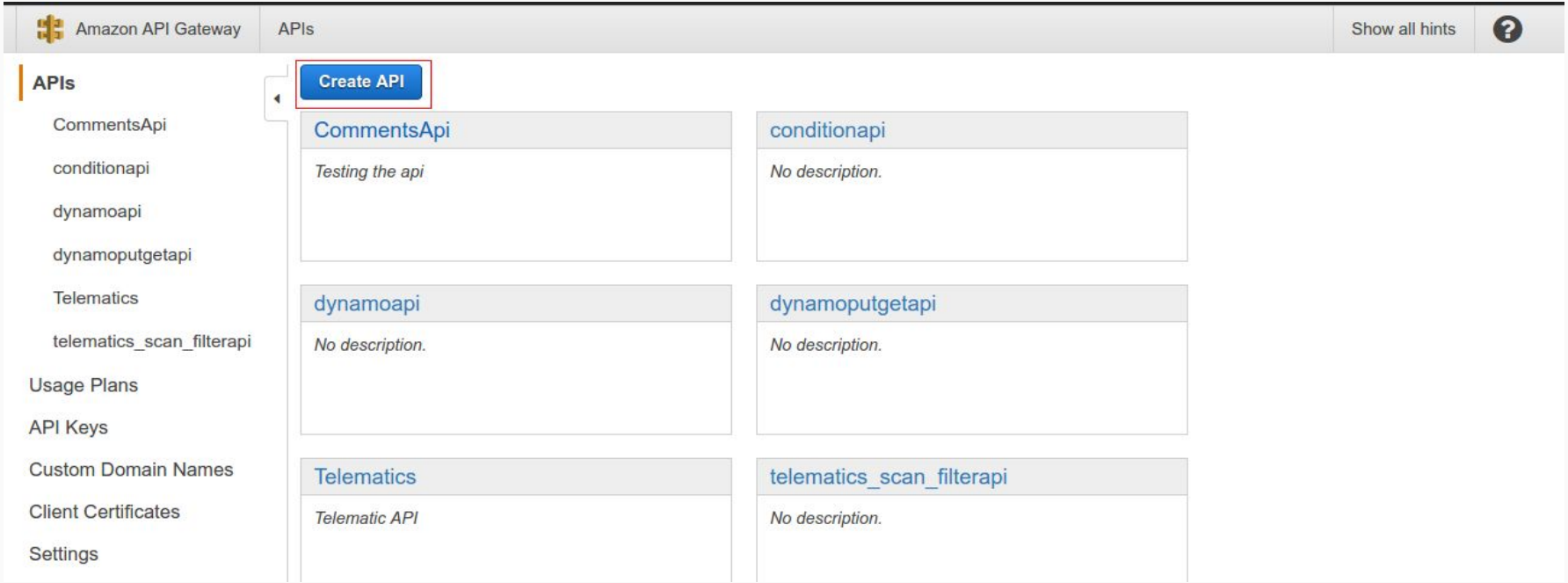
STEP 9.1 : You have attached the policy to the Role.



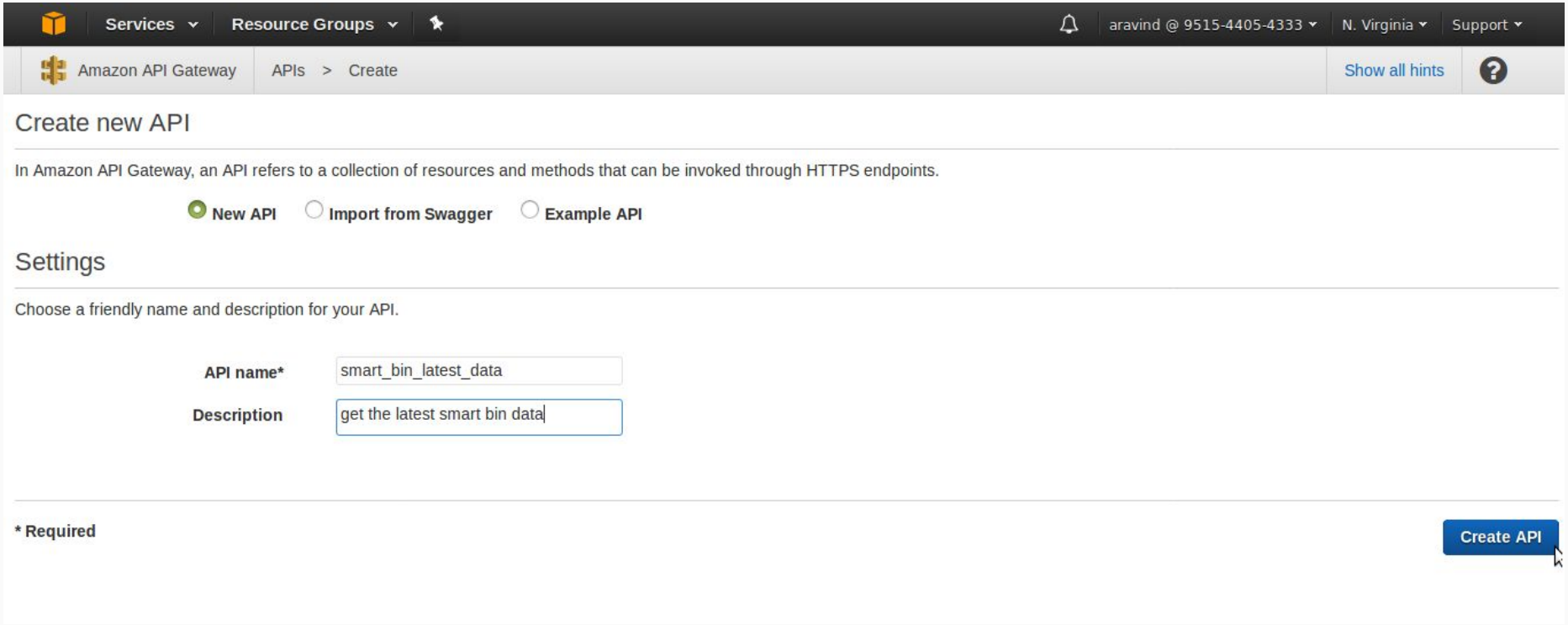
Get latest data from dynamo db

STEP 1 : From the services select the “API Gateway” under “Application services”.

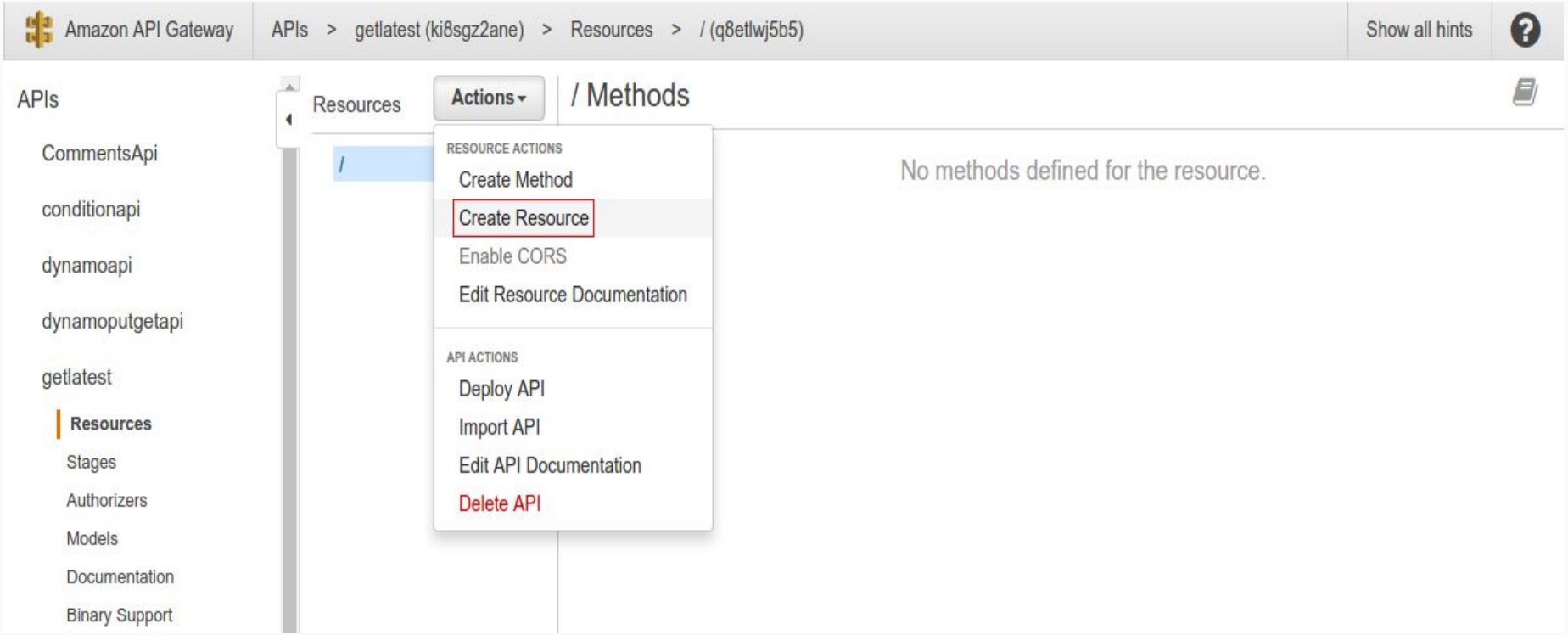
STEP 2 : Click on the “Create API” to create a new api.



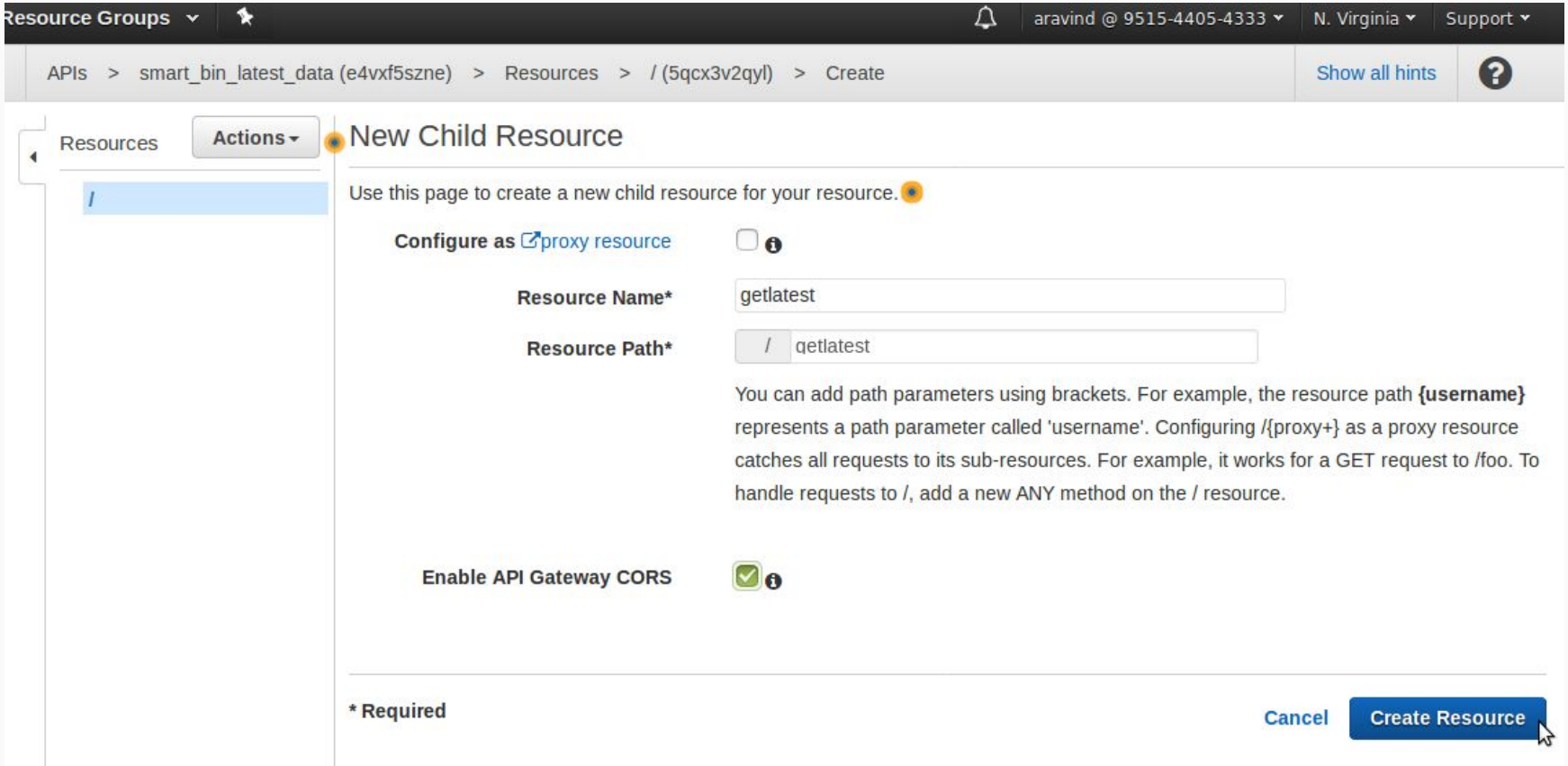
STEP 3 : Enter the API name (your choice)



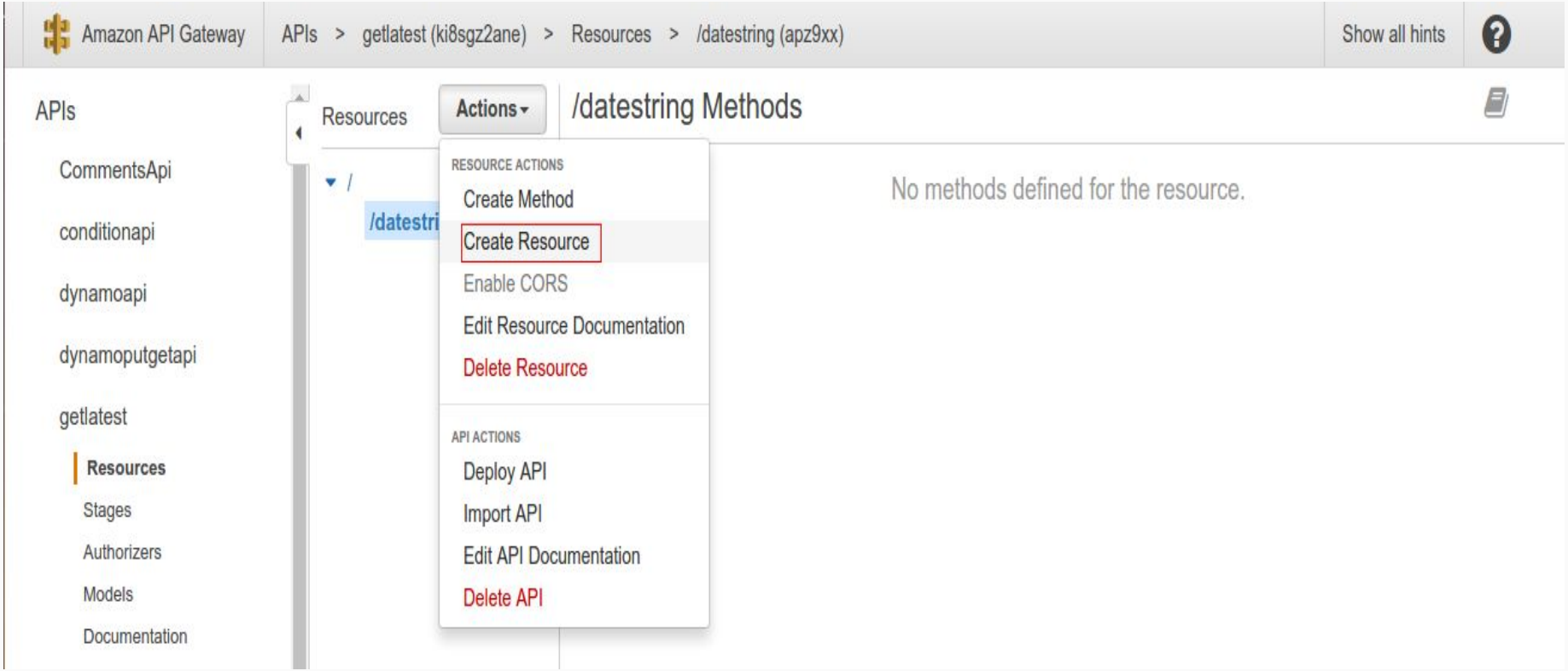
STEP 4 : Once you create the API , then start create resources.



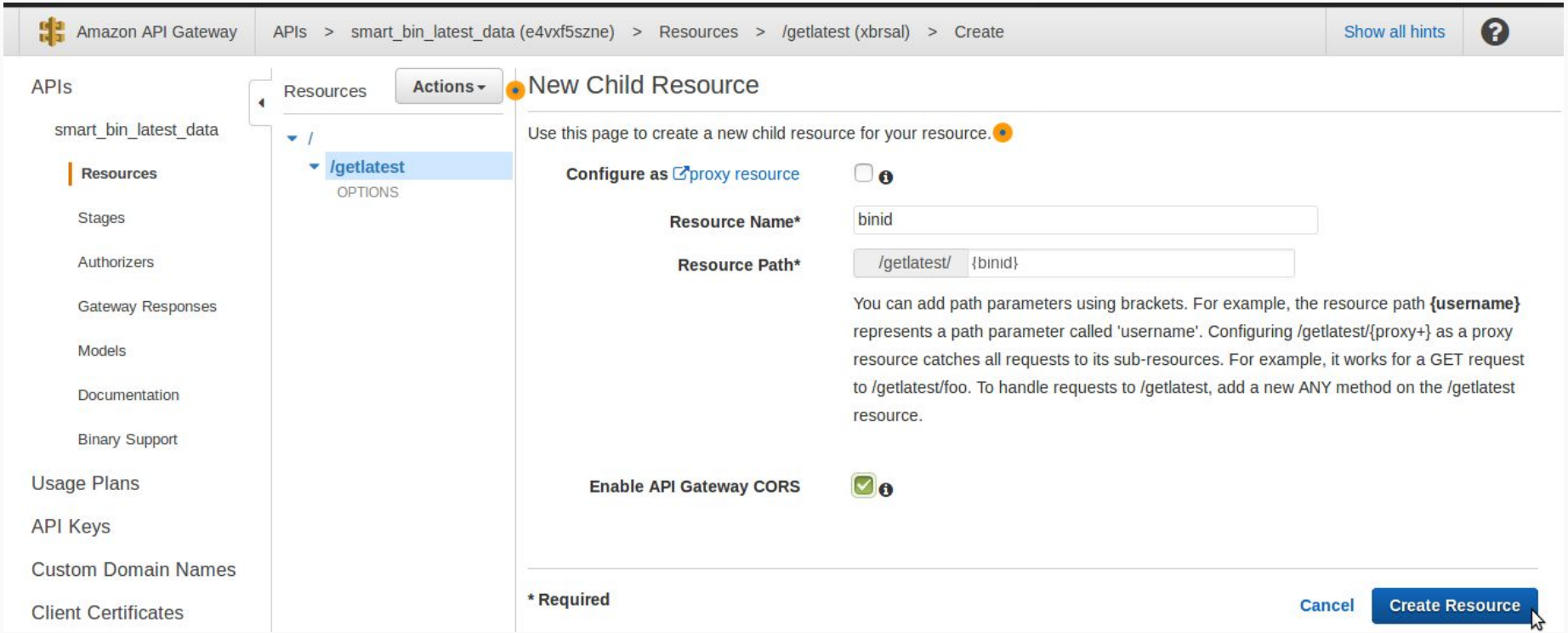
Enter the name for the resource.(name - getlatest) Click on “Create Resource”.



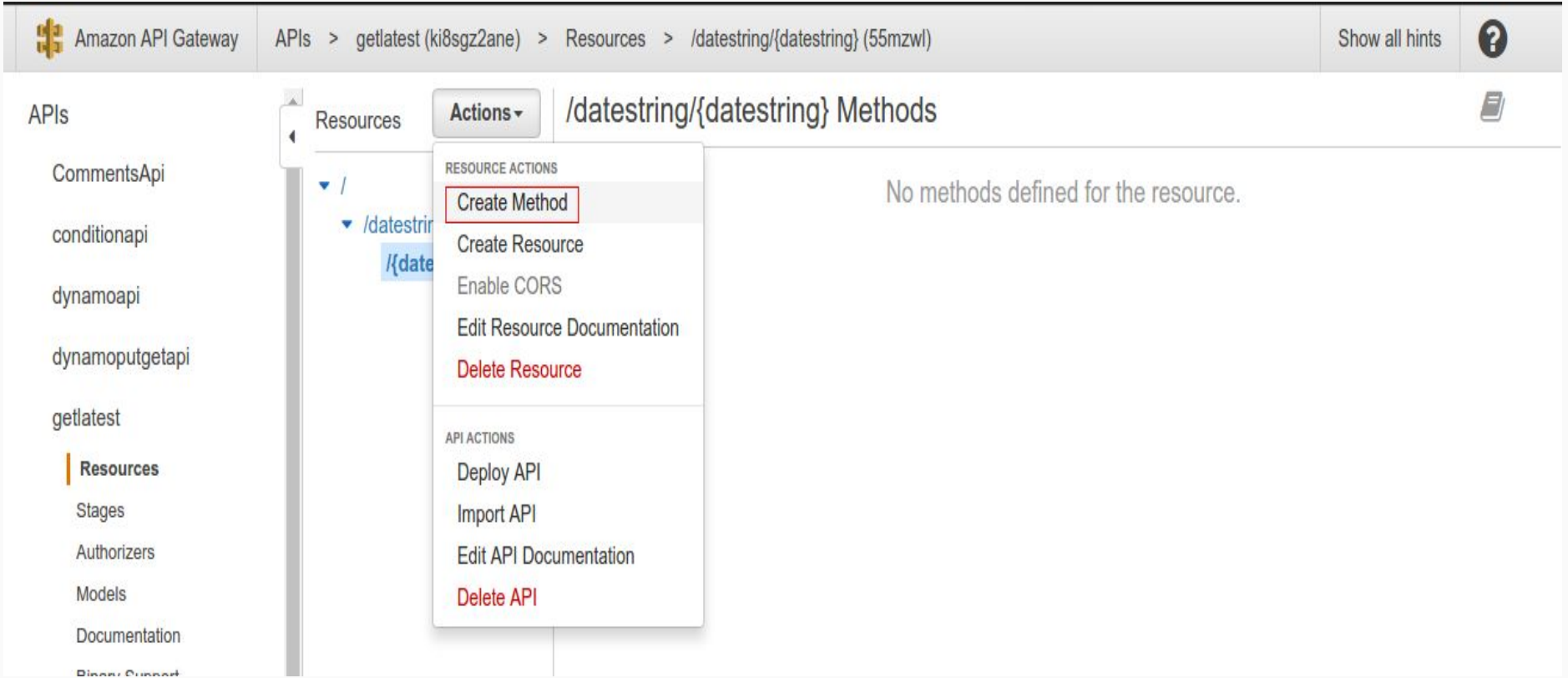
STEP 5 : Create one more resource to give the primary key as a parameter to the url path



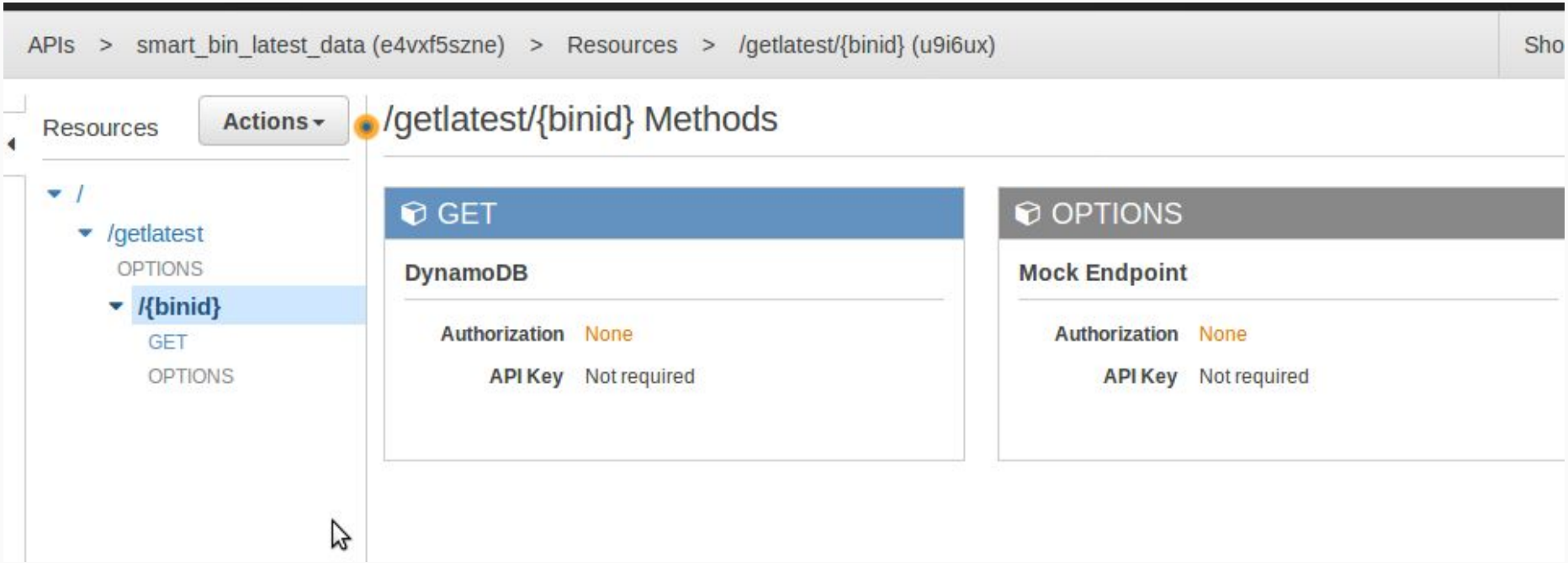
STEP 5.1 : This time in the “Resource Path” field write the primary key of your dynamodb table inside the curly brackets (this is the parameter you are going to give as input).



STEP 6 : Now create Method



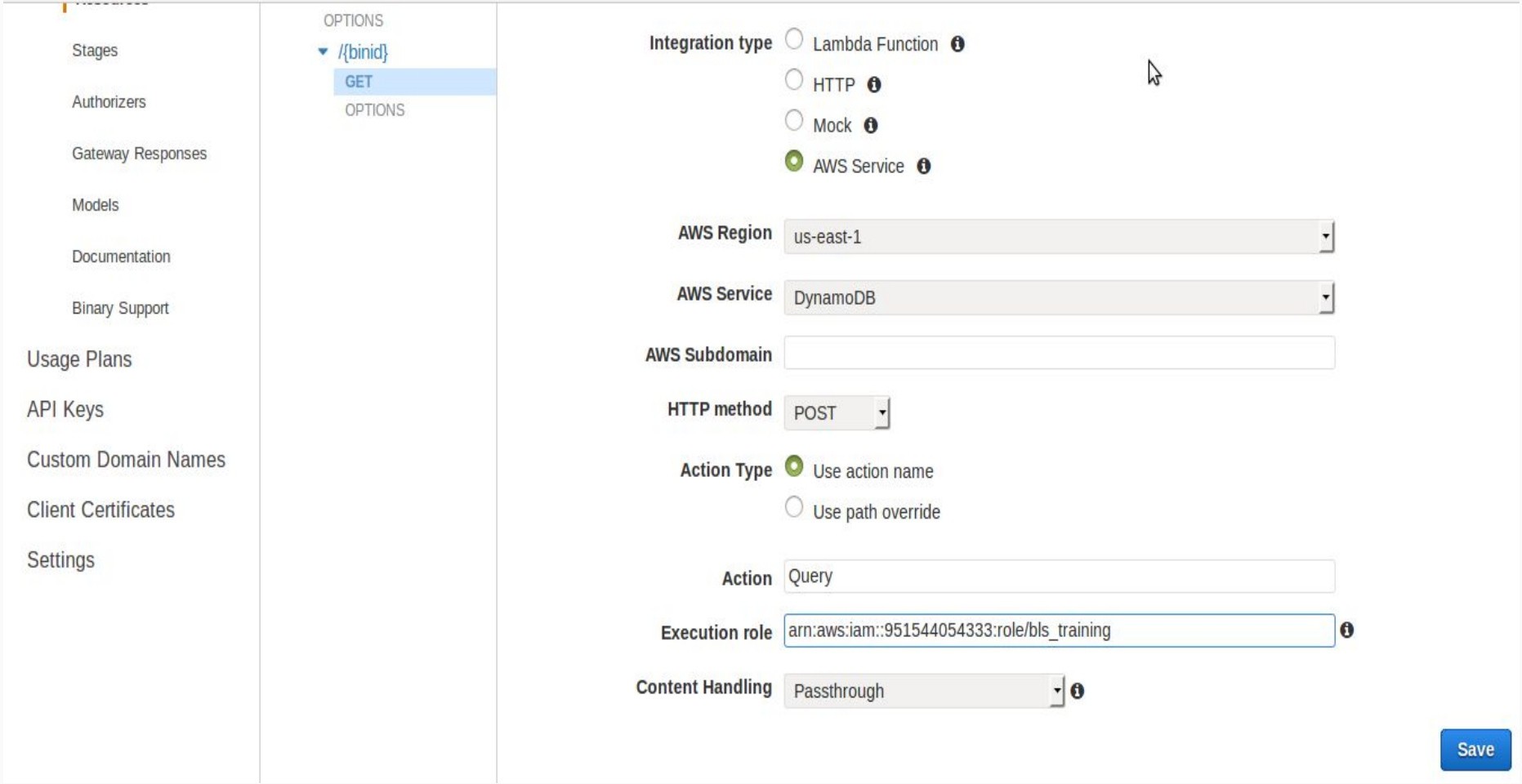
STEP 6.1 : You can select any operation from the drop down menu, if you select “ANY” means you can do any operation either POST/GET/PUT etc. Here select “GET” and click on the “tick” symbol to create.



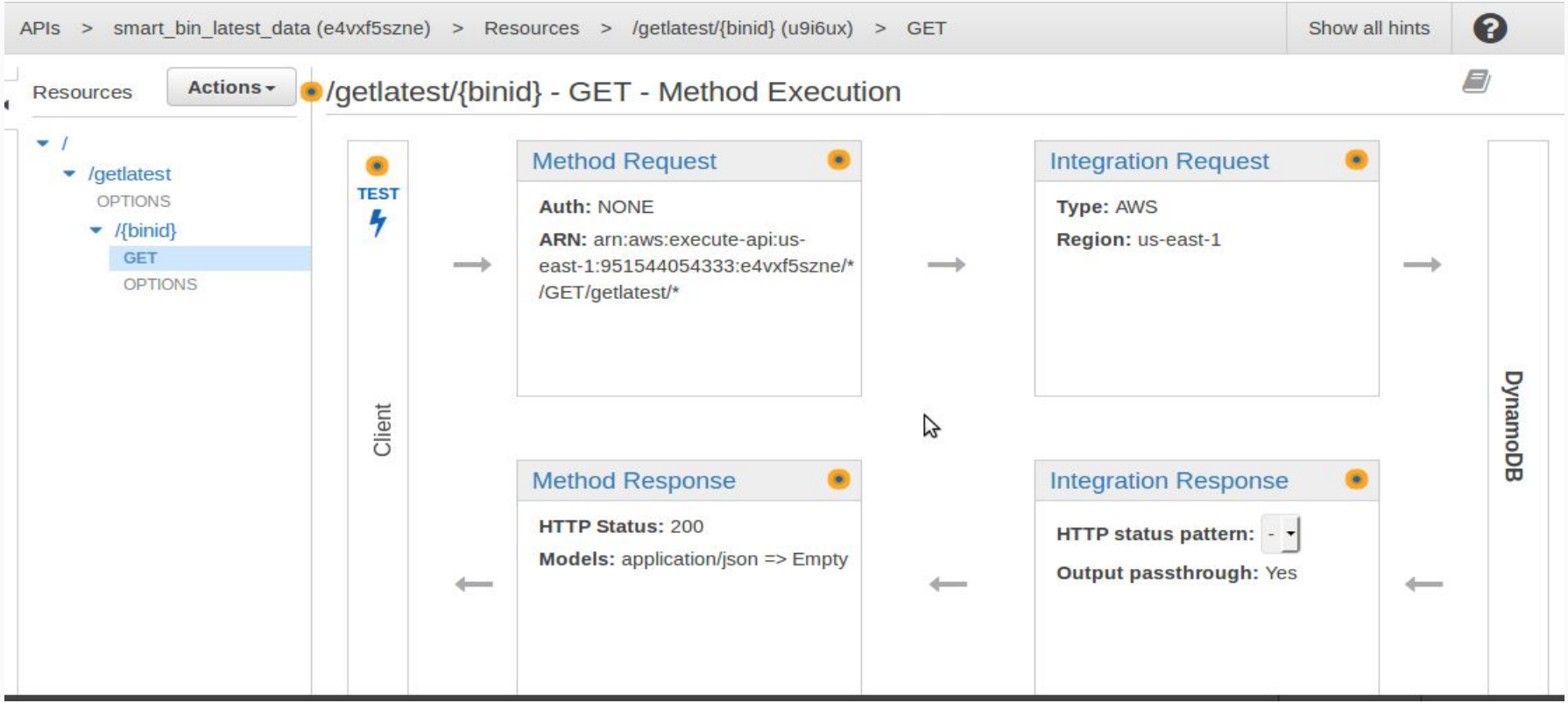
STEP 7 : Select any integration type for this api gateway, we are going to use internal dynamodb, it comes under the AWS service, so select the AWS Service and Enter the

- AWS Region - you have created your dynamodb and api gateway,
- AWS Service - in this case it is DynamoDB.
- HTTP method - for api gateway every operation is a POST operation so you have to select the POST.
- Action - It means what database operation you are going to do (PutItem/Query/Scan/BatchGetItem.,etc.), so select QUERY.
- Execution role - Give the role that you have created in the earlier step.

Then click on the Save button.

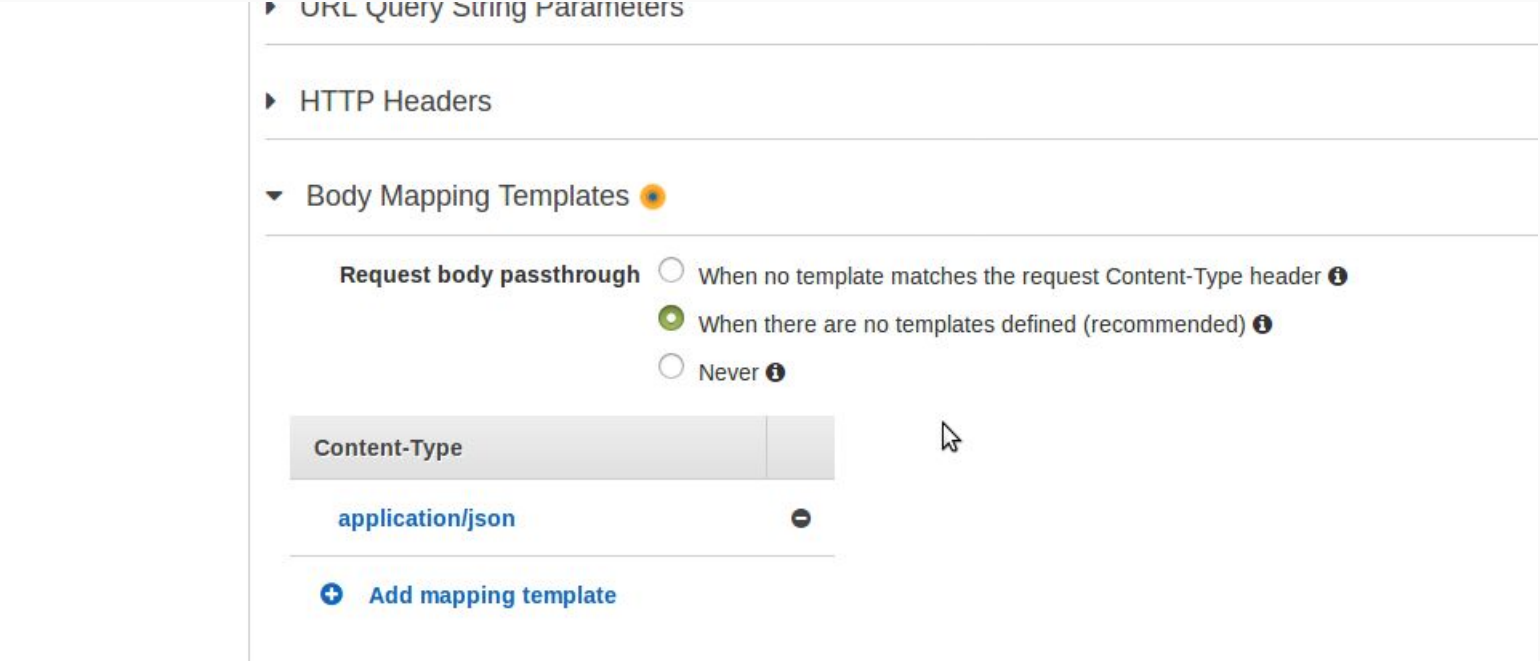

Stages
Authorizers
Gateway Responses
Models
Documentation
Binary Support
Usage Plans
API Keys
Custom Domain Names
Client Certificates
Settings
OPTIONS
/{binid}
GET
OPTIONS
Integration type
Lambda Function
HTTP
Mock
AWS Service
AWS Region
us-east-1
AWS Service
DynamoDB
AWS Subdomain
HTTP method
POST
Action Type
Use action name
Use path override
Action
Query
Execution role
arn:aws:iam::951544054333:role/bls_training
Content Handling
Passthrough
Save

STEP 8 : Click on the Integration Request to give the default template for the "Action" that you have selected in the above step.

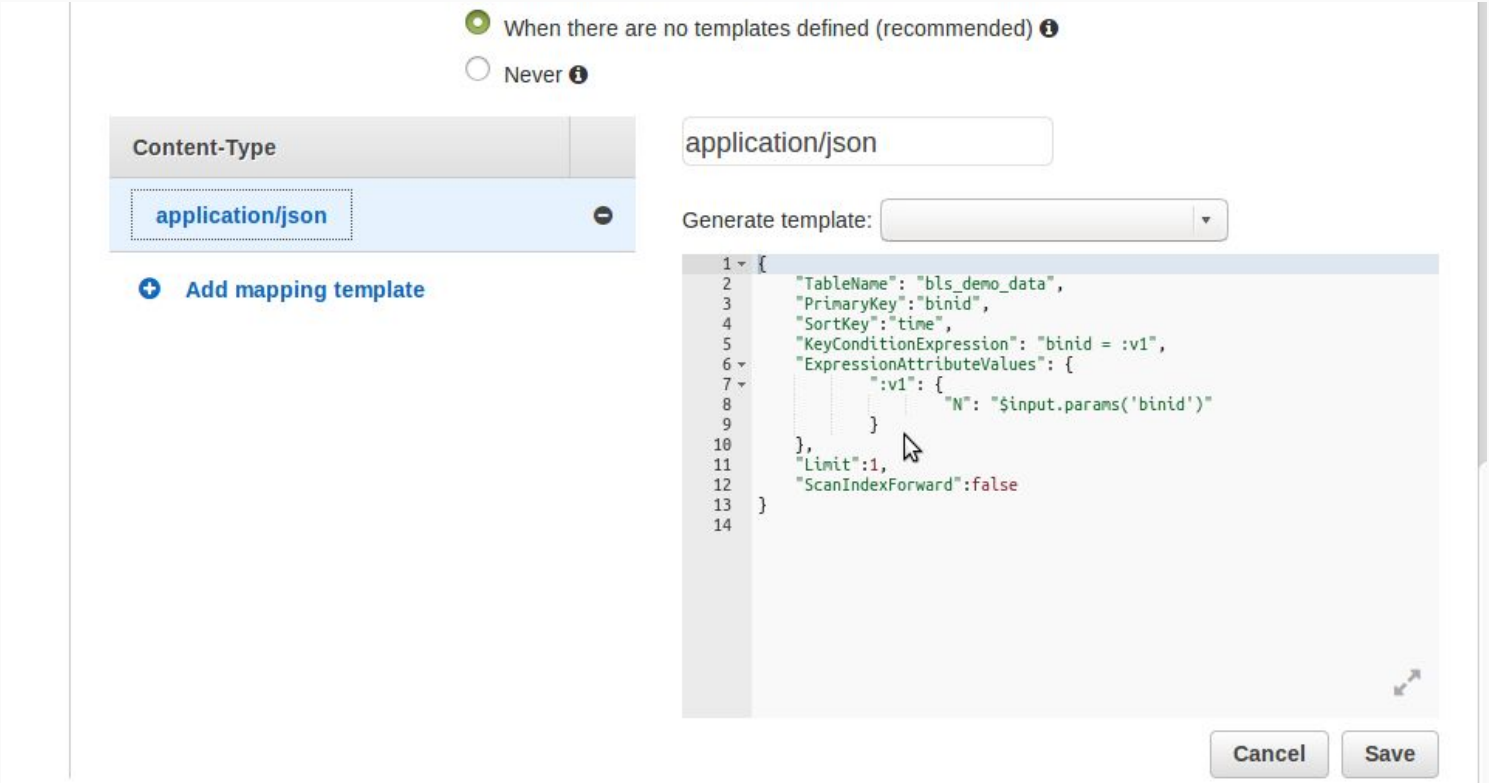


STEP 9 : Goto “Body Mapping Templates” and choose second option for the “Request body passthrough”

STEP 9.1 Then click on the “Add mapping template” and write “application/json” then click on “tick” symbol to create.



STEP 10 : Click on the created “application/json” and write the template for the Selected Action.



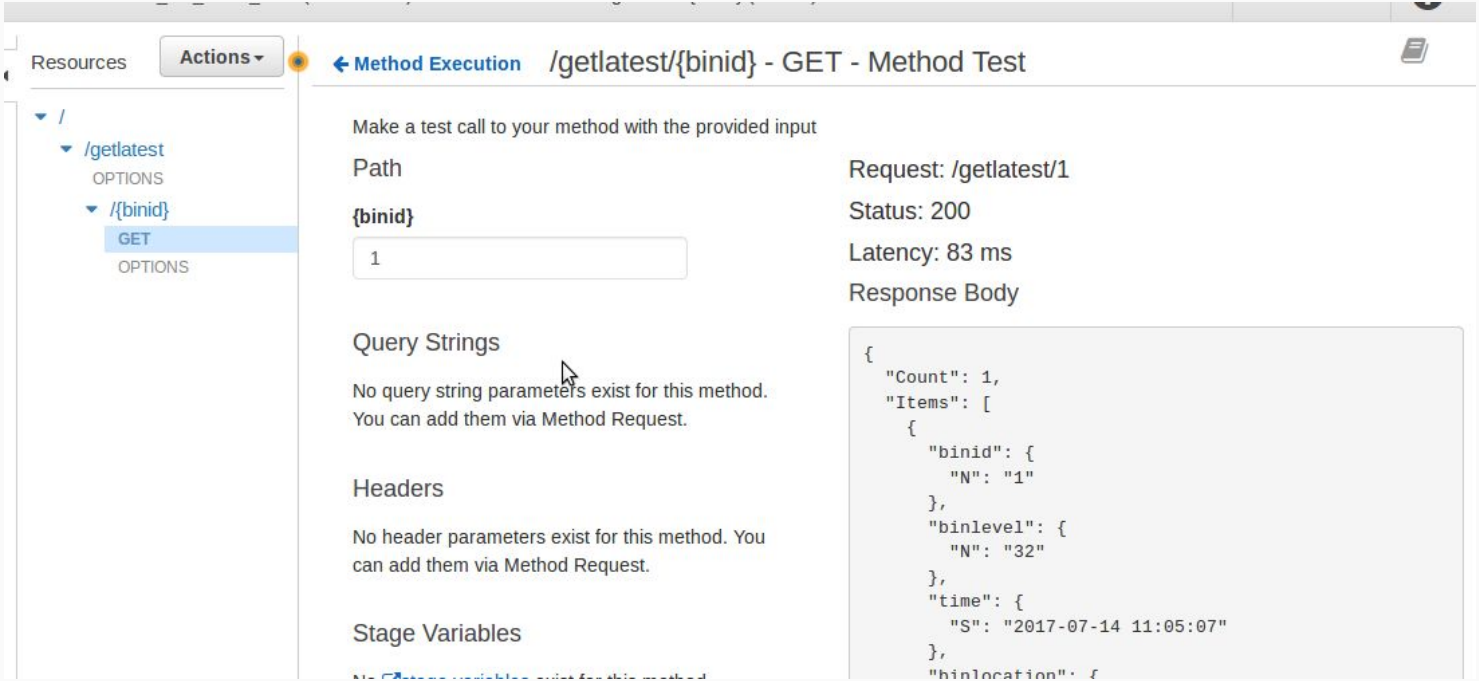
In our case , the table name is “bls_demo_data”, primary key is “binid” and sort key is “time”.Place the below given template and save

```
{
        "TableName": "bls_demo_data",
        "PrimaryKey":"binid",
        "SortKey":"time",
        "KeyConditionExpression": "binid = :v1",
        "ExpressionAttributeValues": {
                ":v1": {
                        "N": "$input.params('binid')"
                }

        },
        "Limit":1,
        "ScanIndexForward":false
}
```

STEP 11 : To test the api gateway that you have selected click on the “Test”. And then select “GET” from Method drop down and give some value in the blank field.

Note : The value to be entered will vary depends on your particular table.



Program Execution Steps

STEP 1 : Git clone Smart bin repo
https://github.com/AravindNico/smartBinAWS

STEP 2 : Execute binoperation script for simulating the bin filling scenario

1. Go to the binoperation directory and open binSimulation_start.sh file in editor.
2. Replace the aws certificates and key file names in the python execution command arguments.
3. Now run the shell script as

   ./binSimulation_start

STEP 3 : Similarly , for executing the binpickupOperation script , which resets the bin level to “0” indicating the bin pickup operation completion.

1. Go to the pickupOperation directory and open binPickup_start.sh file in editor.
2. Replace the aws certificates and key file names in the python execution command arguments.
3. Now run the shell script as

   ./binPickup_start

STEP 4 : To see the smart bin indication and routing on Web Application , go to the UI directory and open index.html file in a web browser.

NOTE :

1 . Before opening the html file , you have to setup your aws credentials in the awsConfig.json file , Credentials you can get from AWS cognito Service.

2 . While sending message in MQTT , the message payload should be string enclosed in double quotes.