

Alexa Document

Create account in

1. Amazon developer account and
2. AWS account with same credentials

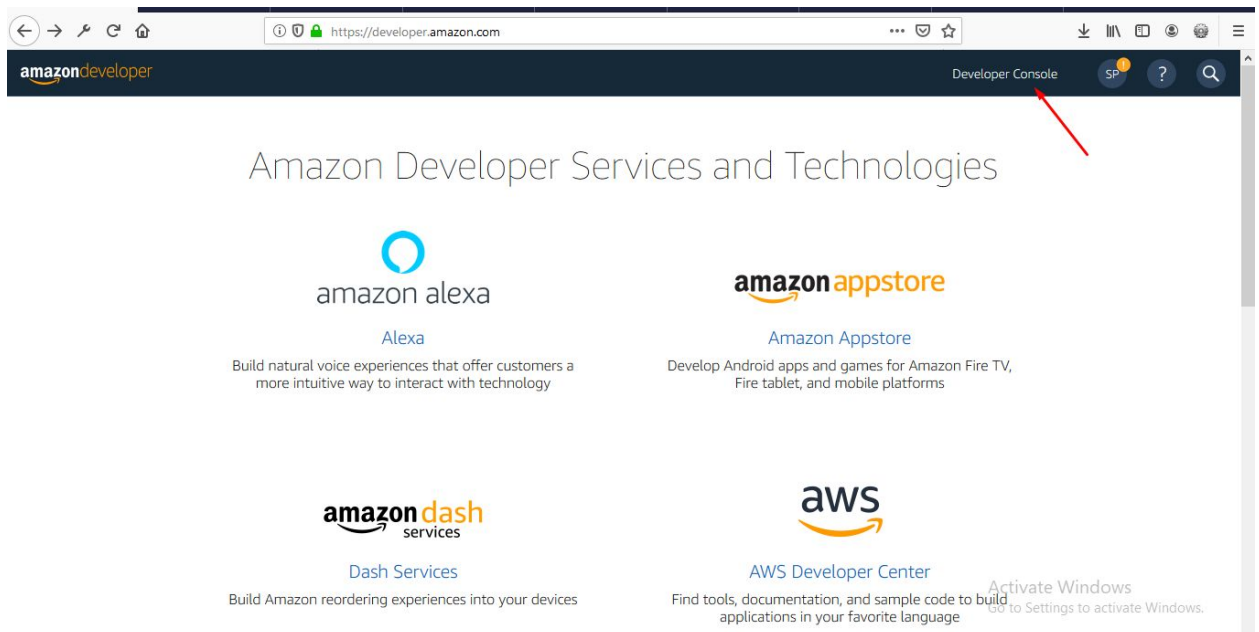
Alexa Skill :->

- 1) <https://developer.amazon.com/docs/custom-skills/steps-to-build-a-custom-skill.html>
- 2) <https://developer.amazon.com/docs/devconsole/build-your-skill.html#custom-model>

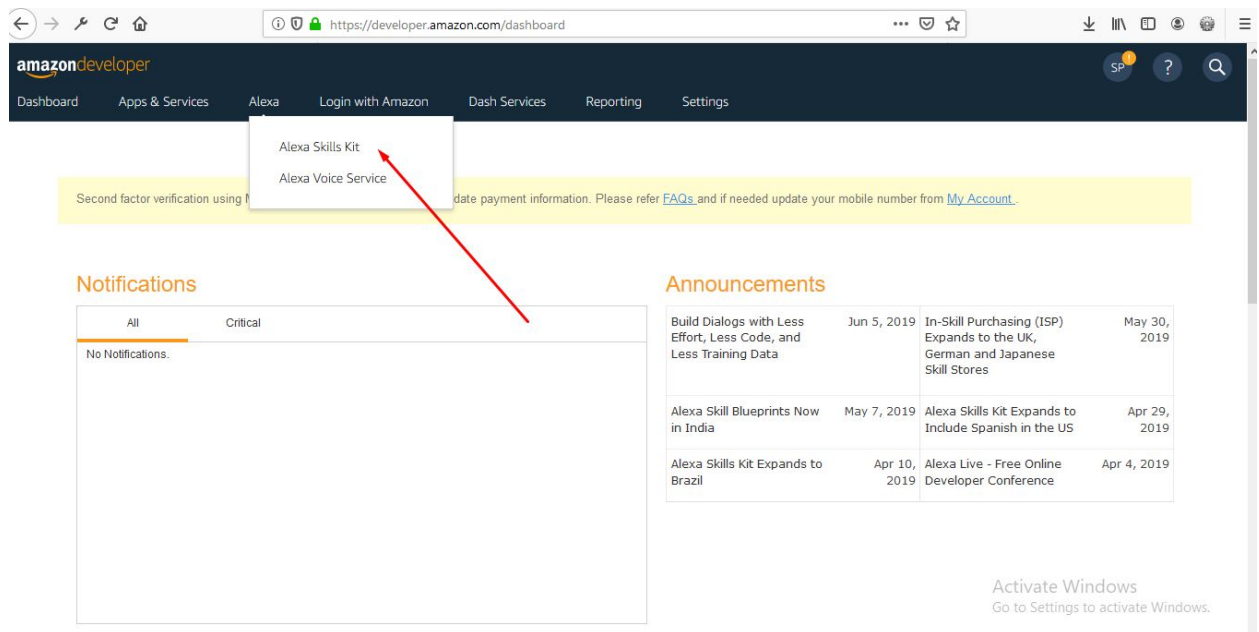
Lambda :->

- 1) <https://developer.amazon.com/docs/custom-skills/host-a-custom-skill-as-an-aws-lambda-function.html>

1. Login to amazon.developer <https://developer.amazon.com/>
Go to Developer Console



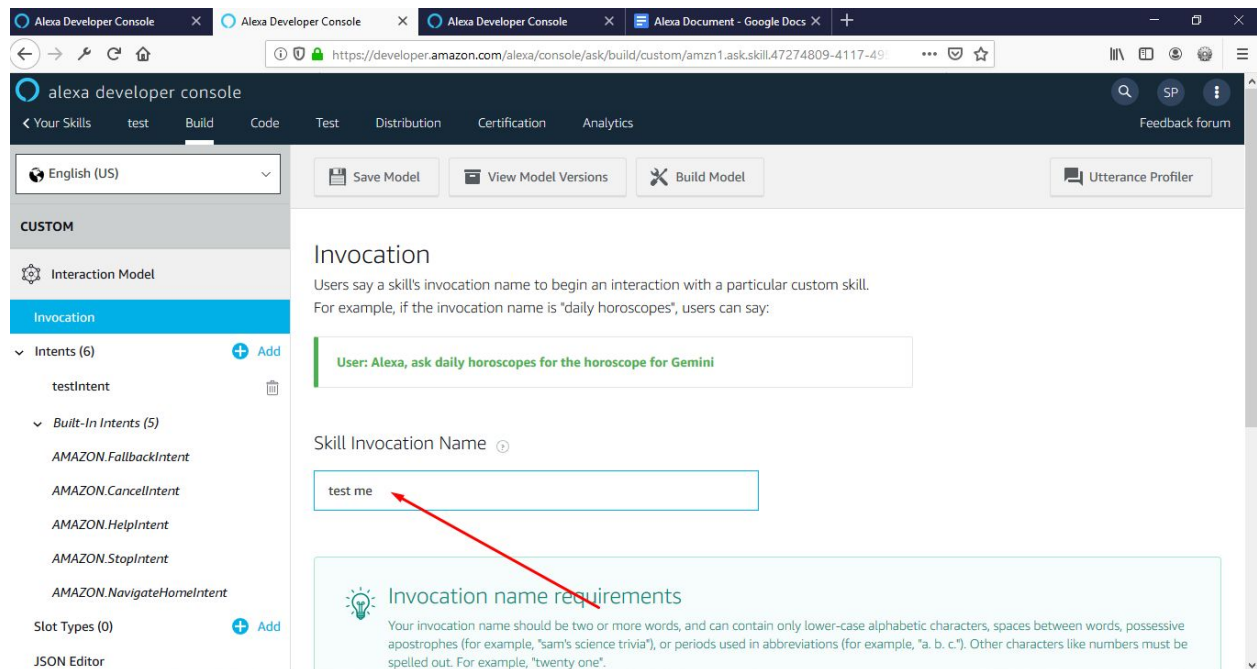
2. Then click Alexa->Alexa Skills Kit



1. Click **Create Skill**.
2. Enter the skill name and default language.
3. Click the model you want to include.
(Choose Custom Model)
4. Choose a method to host your skill's backend
Resources
(Choose Provision your own)
5. Choose Template
(Start from scratch)
Click **Choose**

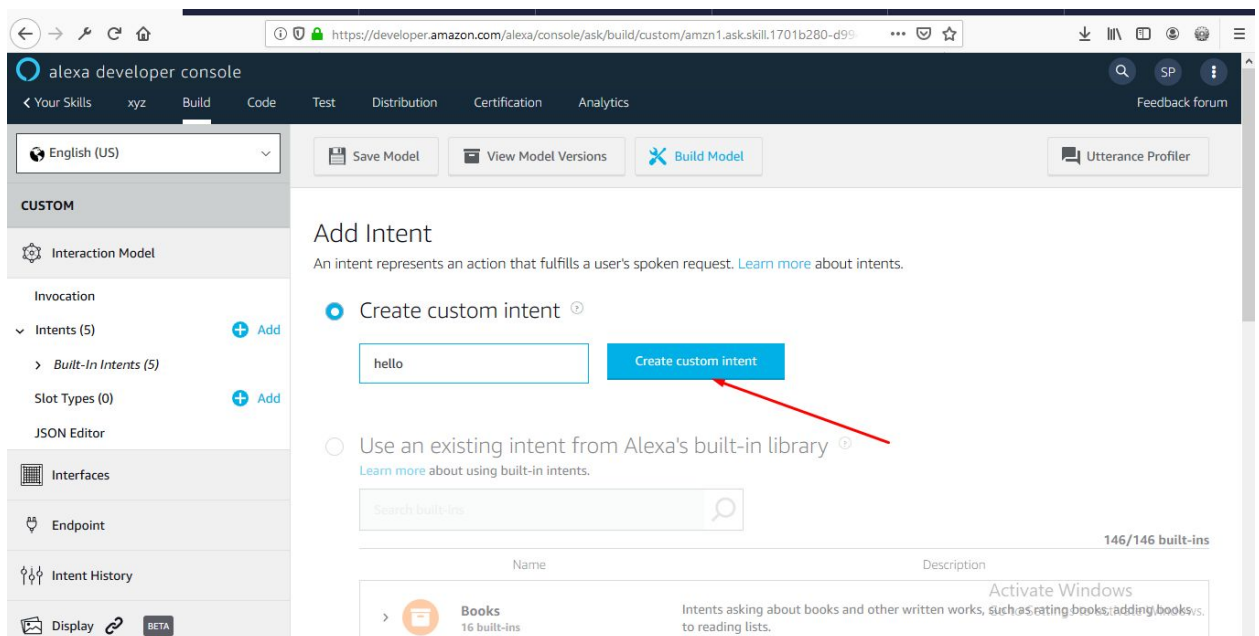
3. Once the developer console has created your new skill, you can start to configure it on the **Build** page. The specific information you need to provide depends on the model you select.

1. Add Skill Invocation Name: test me

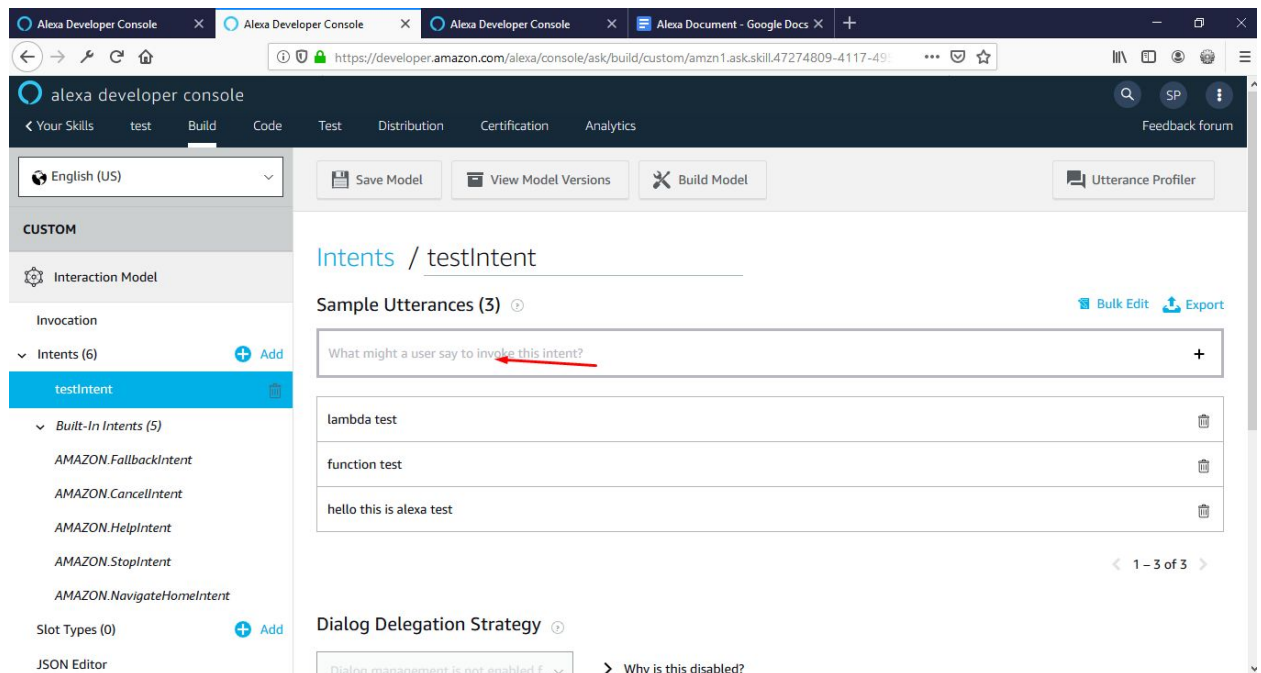


2. Click on intents from sidebar and Create Custom intent

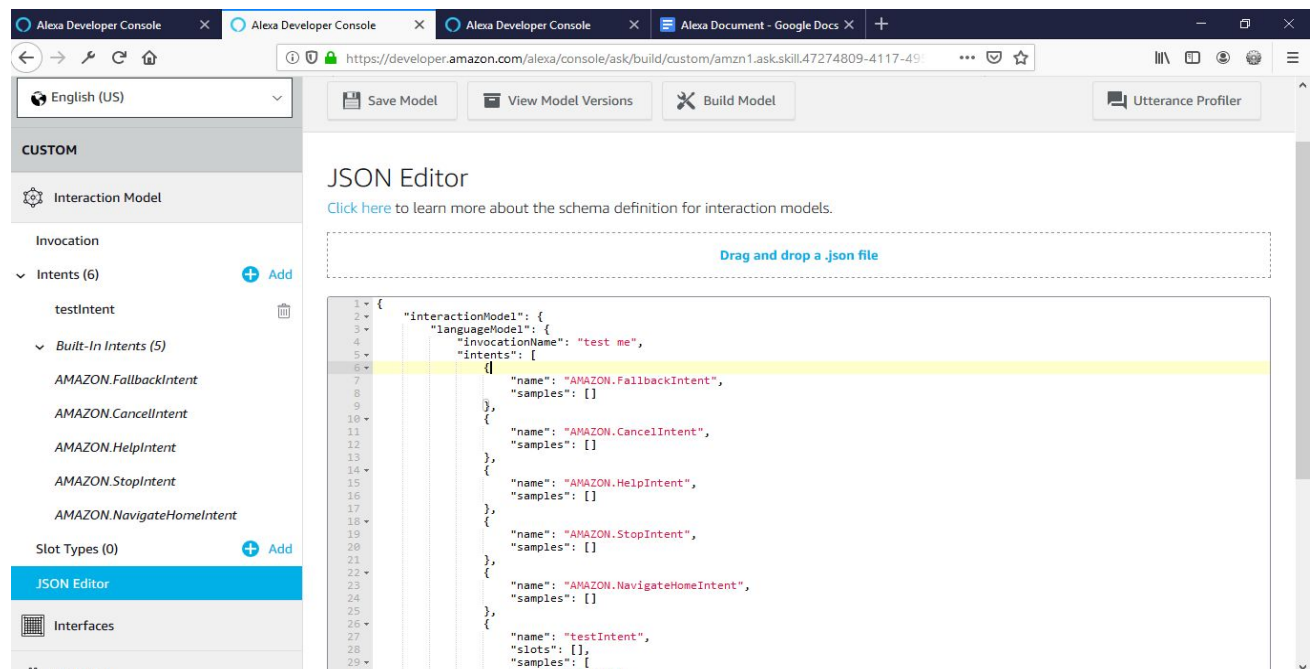
Intent->Add Intent (name: testIntent)



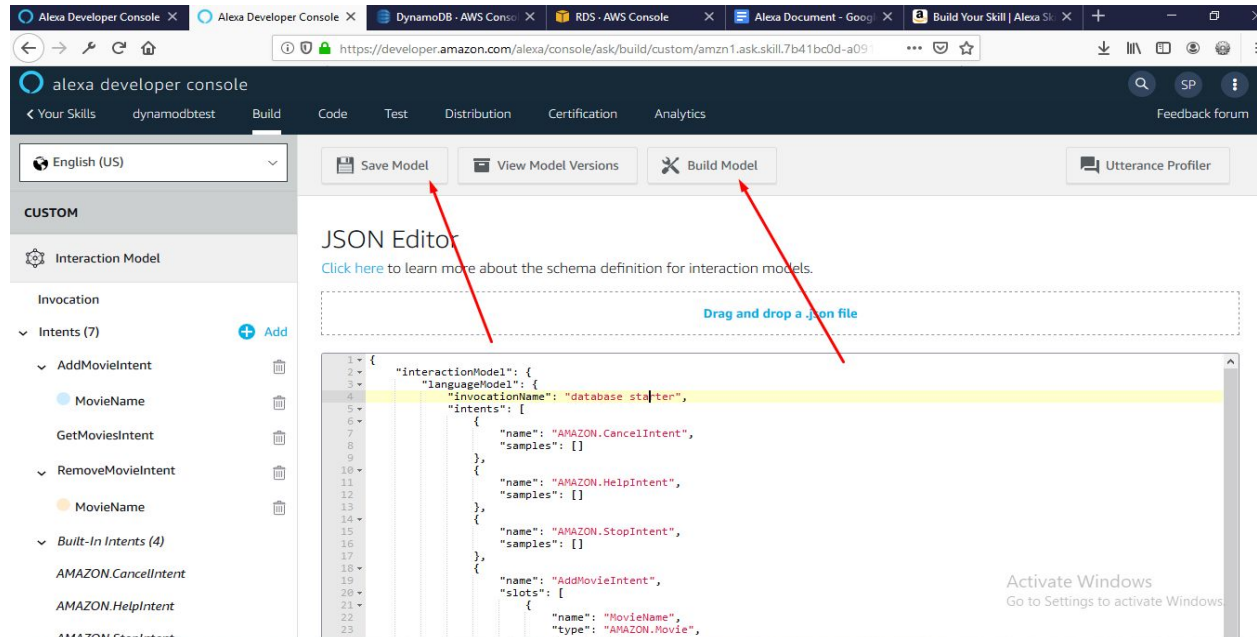
3. Add sample Utterances



You can see all intents in json editor and can change directly from here.

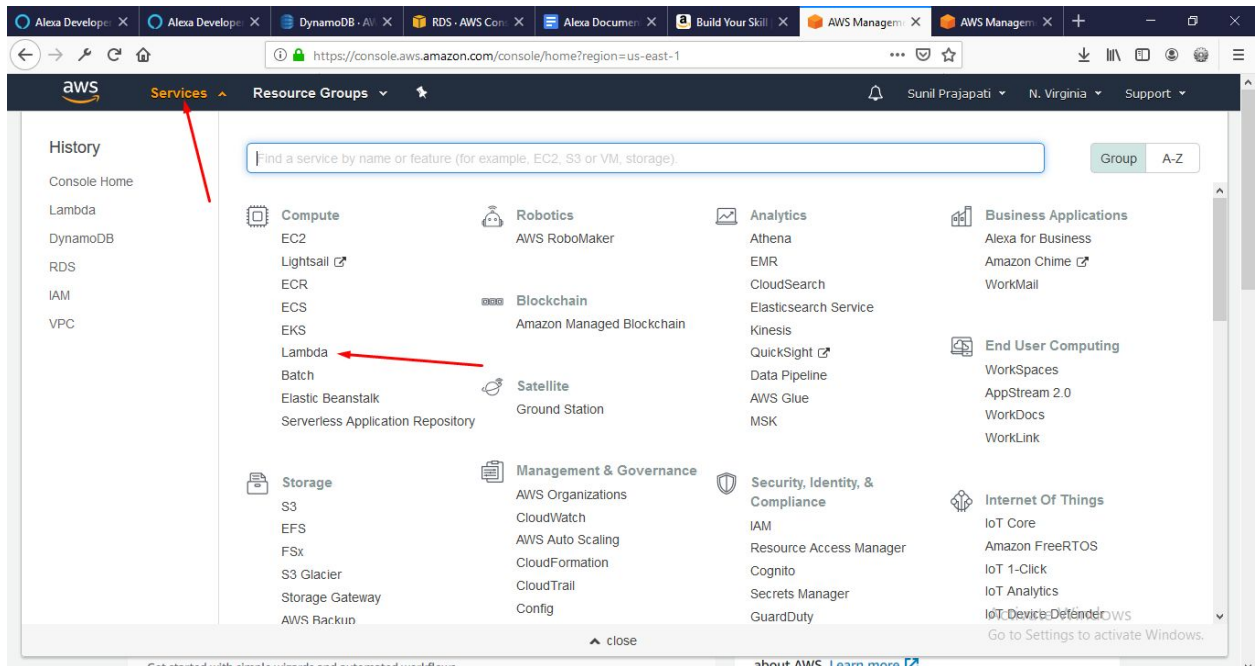


4. Click **Save Model** and then **Build Model**

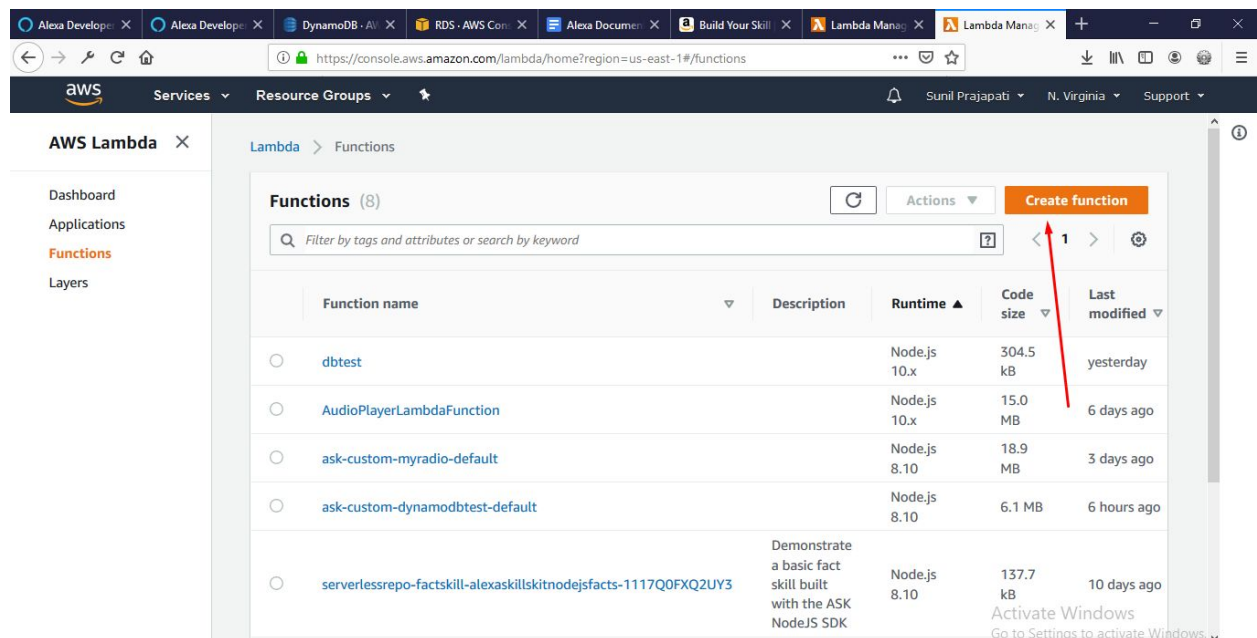


2. Login to <https://console.aws.amazon.com/console/>
On AWS Dashboard, Select *US East (N. Virginia)* region on top right – this is the only **region** with Alexa/Lambda free tier service.

1. Go to Services then search for Lambda



2. click **Create function**



3. Select **Author from scratch**

The screenshot shows the AWS Lambda 'Create function' page. The 'Author from scratch' option is selected. The 'Basic information' section shows the function name 'yourfunctionname' and the runtime 'Node.js 10.x'.

Create function [Info](#)

Choose one of the following options to create your function.

- Author from scratch** ☒ Start with a simple Hello World example.
- Use a blueprint** ☐ Build a Lambda application from sample code and configuration presets for common use cases.
- Browse serverless app repository** ☐ Deploy a sample Lambda application from the AWS Serverless Application Repository.

Basic information

Function name
Enter a name that describes the purpose of your function.

Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime [Info](#)
Choose the language to use to write your function.

Basic information add below mentioned details and then **click** Create function

The screenshot shows the AWS Lambda 'Create function' page, specifically the 'Permissions' section. The 'Execution role' is set to 'Use an existing role' and the 'Existing role' is 'lambda_basic_execution'.

Basic information

Function name
Enter a name that describes the purpose of your function.

Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime [Info](#)
Choose the language to use to write your function.

Permissions [Info](#)
Lambda will create an execution role with permission to upload logs to Amazon CloudWatch Logs. You can configure and modify permissions further when you add triggers.

Choose or create an execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).

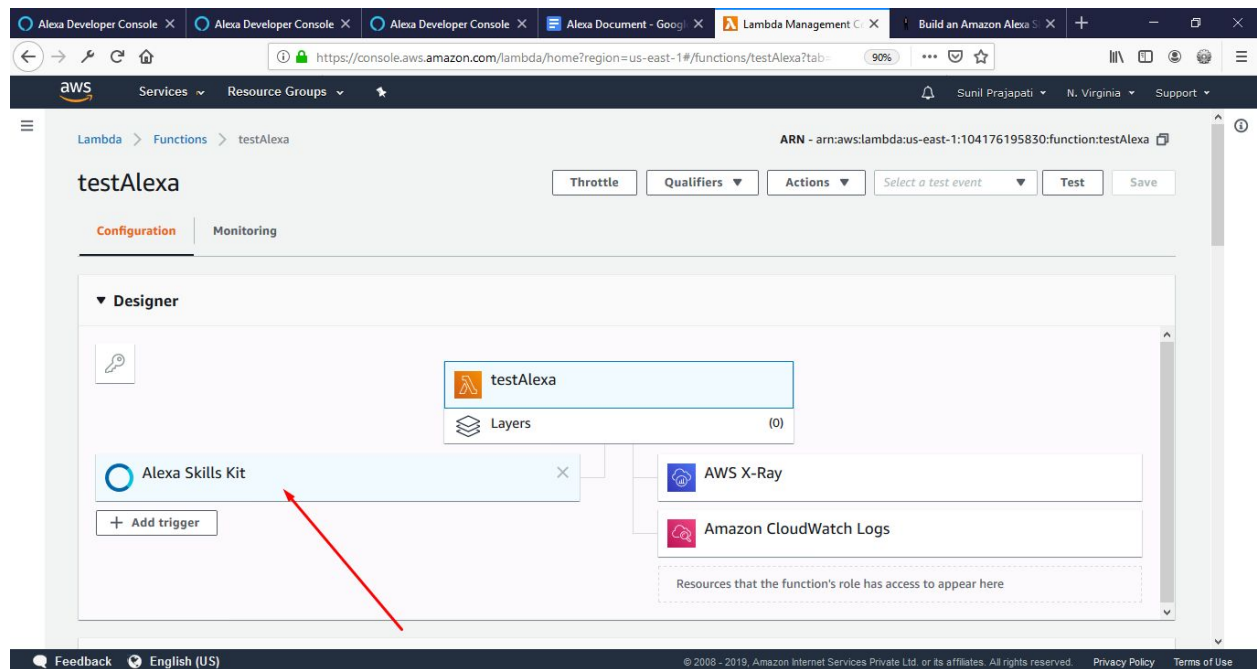
Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.

[View the lambda_basic_execution role on the IAM console.](#)

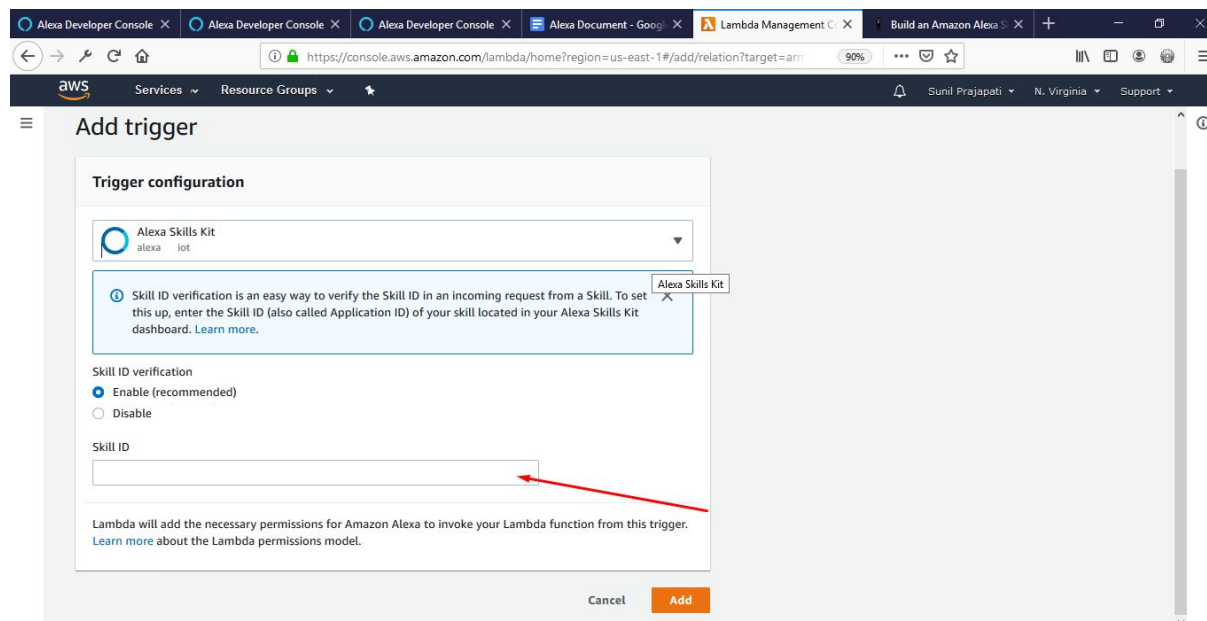
4. Click Add trigger

Choose Alexa Skills kit

Note the **ARN** on the top right (We need this ARN information later)



Add skill ID (you can find skill ID from developer.amazon console)



Go to developer.amazon console

Find your skill(in this case **test**)

Click View skill ID

The screenshot shows the Amazon Developer Console interface. At the top, there are tabs for 'Skills', 'Earnings', and 'Payments'. The 'Skills' tab is active. Below the tabs, there is a search bar labeled 'Search by skill name' and a 'Create Skill' button. A table lists several skills. The first skill is named 'test' and has a status of 'Development'. A red arrow points to the 'View Skill ID' link next to the skill name. A tooltip appears over this link, displaying the Skill ID: 'amzn1.ask.skill.47274809-4117-4959-8d32-51e8555bcf26'. Other skills listed include 'dynamodbtest', 'Multi Stream Audio Player', and 'My Radio'.

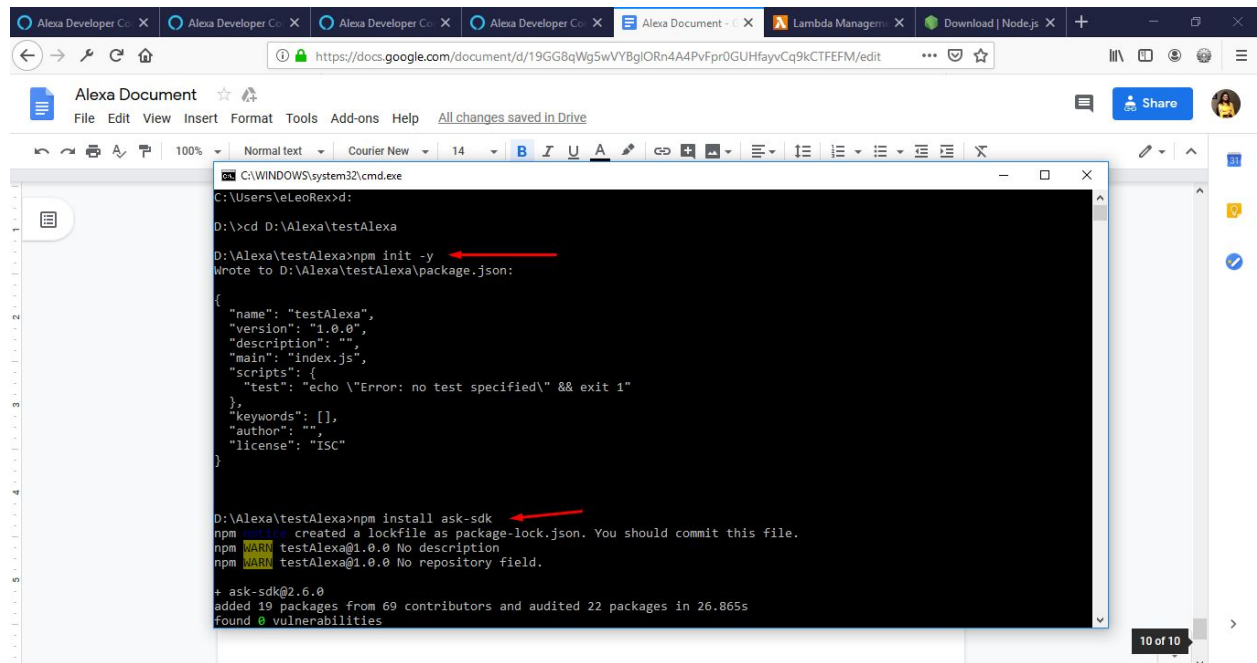
SKILL NAME	LANGUAGE	TYPE	MODIFIED	STATUS	ACTIONS
test View Skill ID	Engl			Development	Analytics Edit Delete
dynamodbtest View Skill ID	Engl			Development	Analytics Edit Delete
Multi Stream Audio Player View Skill ID	English (UK), English (US)	Custom	2019-07-15	In Development	Analytics Edit Delete
My Radio View Skill ID	Italian (IT), English (US), English (CA), English (IN), English (AU), Spanish (ES)	Custom	2019-07-15	In Development	Analytics Edit Delete

5. Open any editor of your choice

(You should have installed node js and npm to create lambda function)

Set up basic node project using **npm init -y**

Install ask-sdk using **npm install ask-sdk**



```
C:\WINDOWS\system32\cmd.exe
C:\Users\eleoRex>d:
D:\>cd D:\Alexa\testAlexa
D:\Alexa\testAlexa>npm init -y
Wrote to D:\Alexa\testAlexa\package.json:

{
  "name": "testAlexa",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}

D:\Alexa\testAlexa>npm install ask-sdk
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN testAlexa@1.0.0 No description
npm WARN testAlexa@1.0.0 No repository field.

+ ask-sdk@2.6.0
added 19 packages from 69 contributors and audited 22 packages in 26.865s
found 0 vulnerabilities
```

Now create index.js file and copy paste below mentioned code.

```

const Alexa = require('ask-sdk');

const TestHandler = {
  canHandle(handlerInput) {
    const request = handlerInput.requestEnvelope.request;
    return request.type === 'LaunchRequest'
      || (request.type === 'IntentRequest'
        && request.intent.name === 'testIntent');
  },
  handle(handlerInput) {

    return handlerInput.responseBuilder
      .speak('Hello congrats ,this is response of your skill from lambda')
      .getResponse();
  },
};

const HelpHandler = {
  canHandle(handlerInput) {
    const request = handlerInput.requestEnvelope.request;
    return request.type === 'IntentRequest'
      && request.intent.name === 'AMAZON.HelpIntent';
  },
  handle(handlerInput) {
    return handlerInput.responseBuilder
      .speak(HELP_MESSAGE)
      .reprompt(HELP_REPROMPT)
      .getResponse();
  },
};

const ExitHandler = {
  canHandle(handlerInput) {
    const request = handlerInput.requestEnvelope.request;
    return request.type === 'IntentRequest'
      && (request.intent.name === 'AMAZON.CancelIntent'
        || request.intent.name === 'AMAZON.StopIntent');
  },
  handle(handlerInput) {
    return handlerInput.responseBuilder
      .speak(STOP_MESSAGE)
      .getResponse();
  },
};

const SessionEndedRequestHandler = {

```

```

canHandle(handlerInput) {
  const request = handlerInput.requestEnvelope.request;
  return request.type === 'SessionEndedRequest';
},
handle(handlerInput) {
  console.log(`Session ended with reason:
${handlerInput.requestEnvelope.request.reason}`);

  return handlerInput.responseBuilder.getResponse();
},
};

const ErrorHandler = {
  canHandle() {
    return true;
  },
  handle(handlerInput, error) {
    console.log(`Error handled: ${error.message}`);

    return handlerInput.responseBuilder
      .speak('Sorry, an error occurred.')
      .reprompt('Sorry, an error occurred.')
      .getResponse();
  },
};

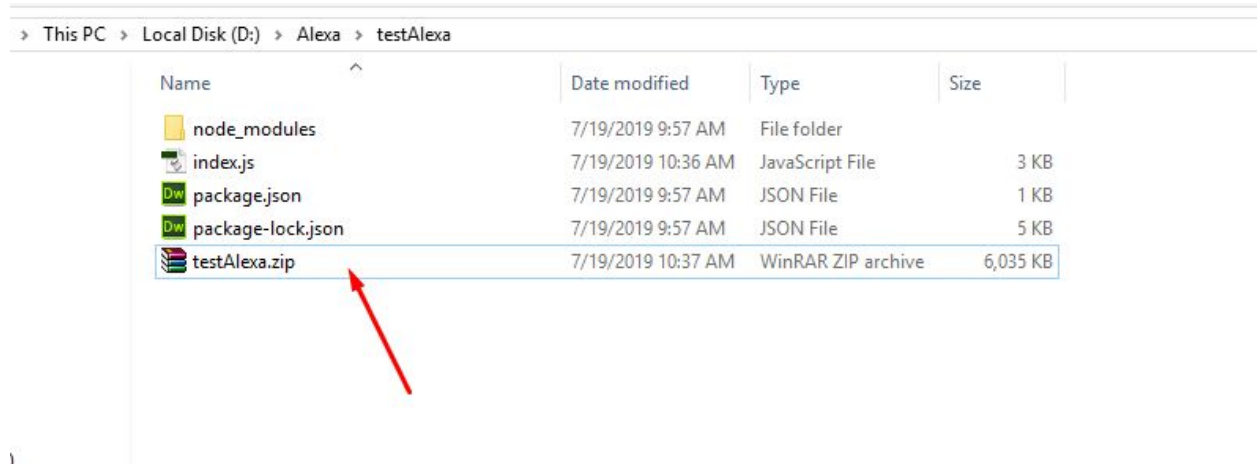
const HELP_MESSAGE = 'You can say test me';
const HELP_REPROMPT = 'What can I help you with?';
const STOP_MESSAGE = 'Goodbye!';

const skillBuilder = Alexa.SkillBuilders.standard();

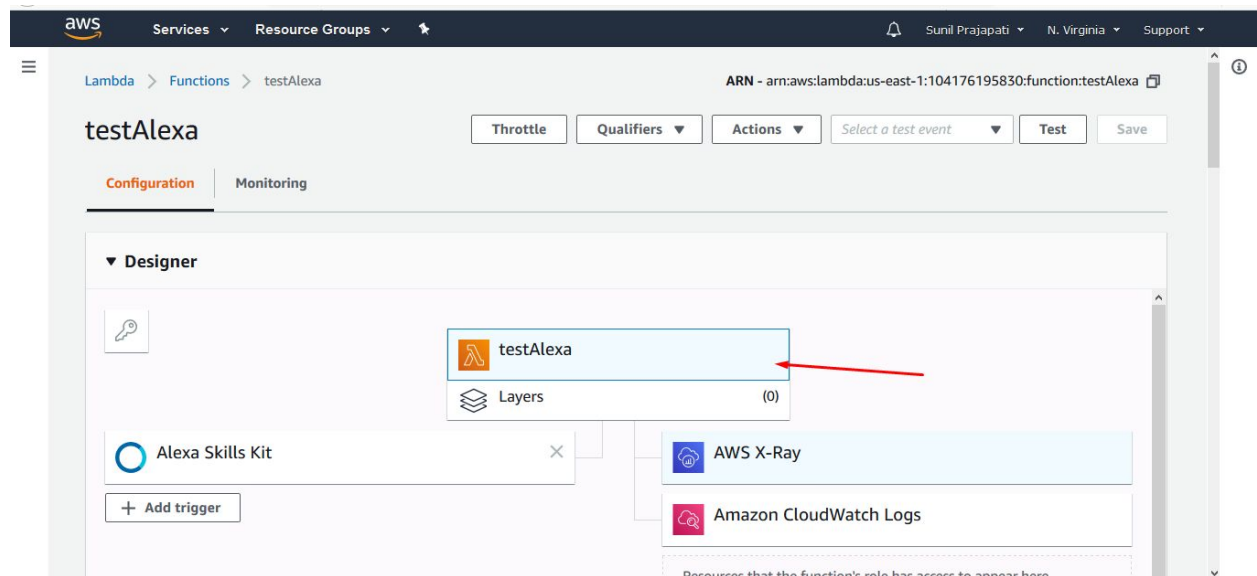
exports.handler = skillBuilder
  .addRequestHandlers(
    TestHandler,
    HelpHandler,
    ExitHandler,
    SessionEndedRequestHandler
  )
  .addErrorHandlers(ErrorHandler)
  .lambda();

```

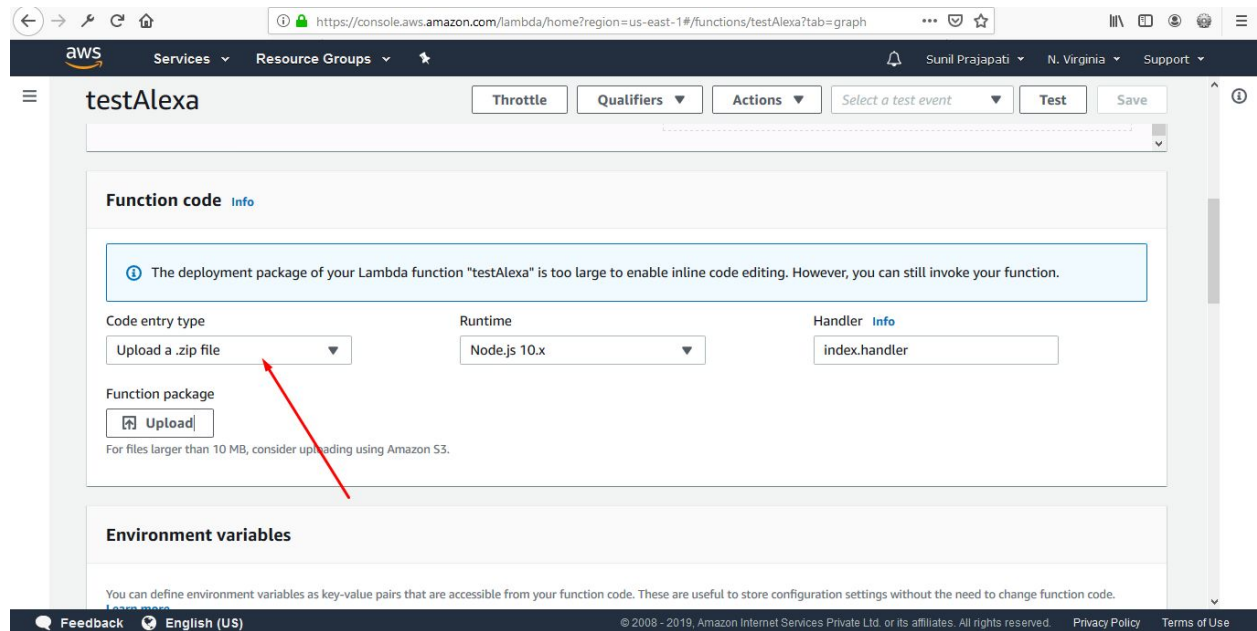
Save this and zip all files



Go to aws.amazon console
Click on lambda function
Scroll down



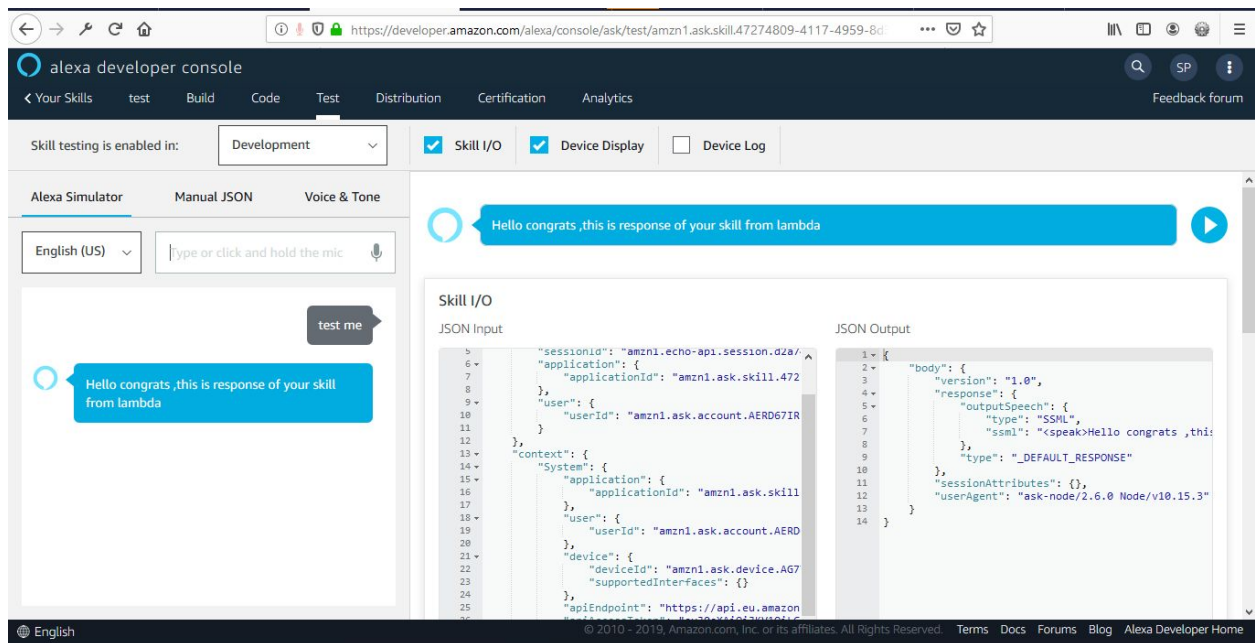
In function code tab
Choose code entry type:upload a zip(upload previously
created zip folder here)



On Amazon Developer Portal, Go to **Alexa Skills Kit Developer Console**

On Endpoint, Choose AWS Lambda ARN as service endpoint type and add your ARN (Which we got from AWS Lambda) at Default Region.
(save & build)

Now test your skill using invocation name



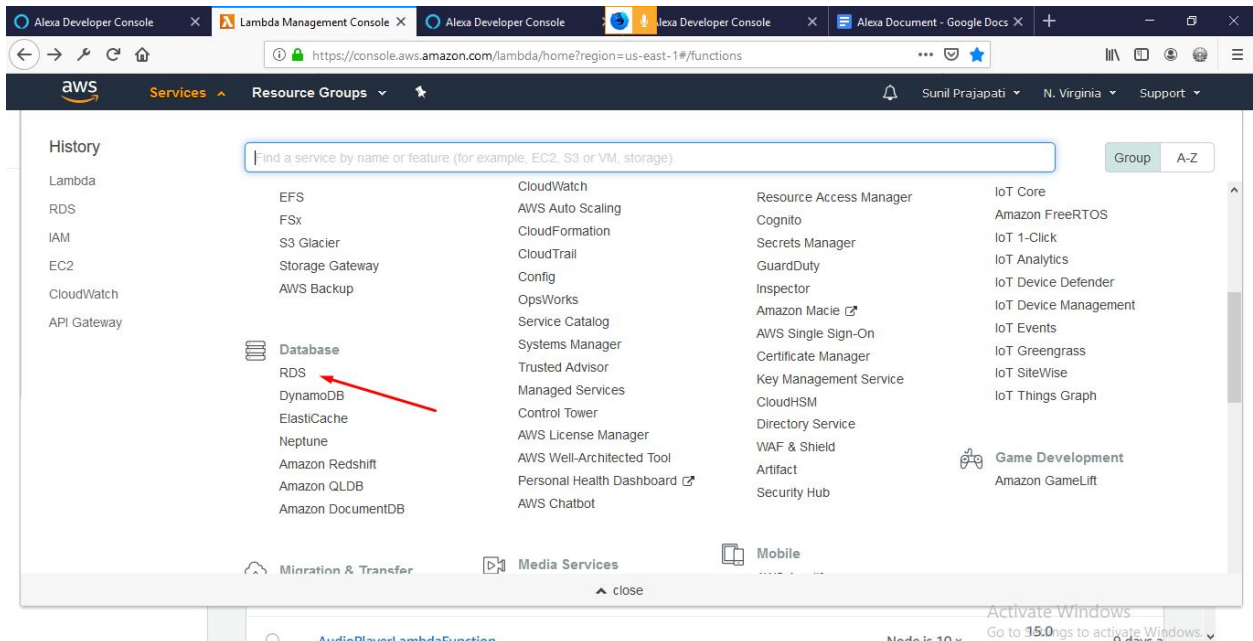
(You will be able to see JOSN input and JSON output)

For testing your own custom skill on your echo device
Register your echo device with same email id
(note:choose same language type for device and skill.
For example if you have created skill using
English(US) then also select English(US) for echo
device).

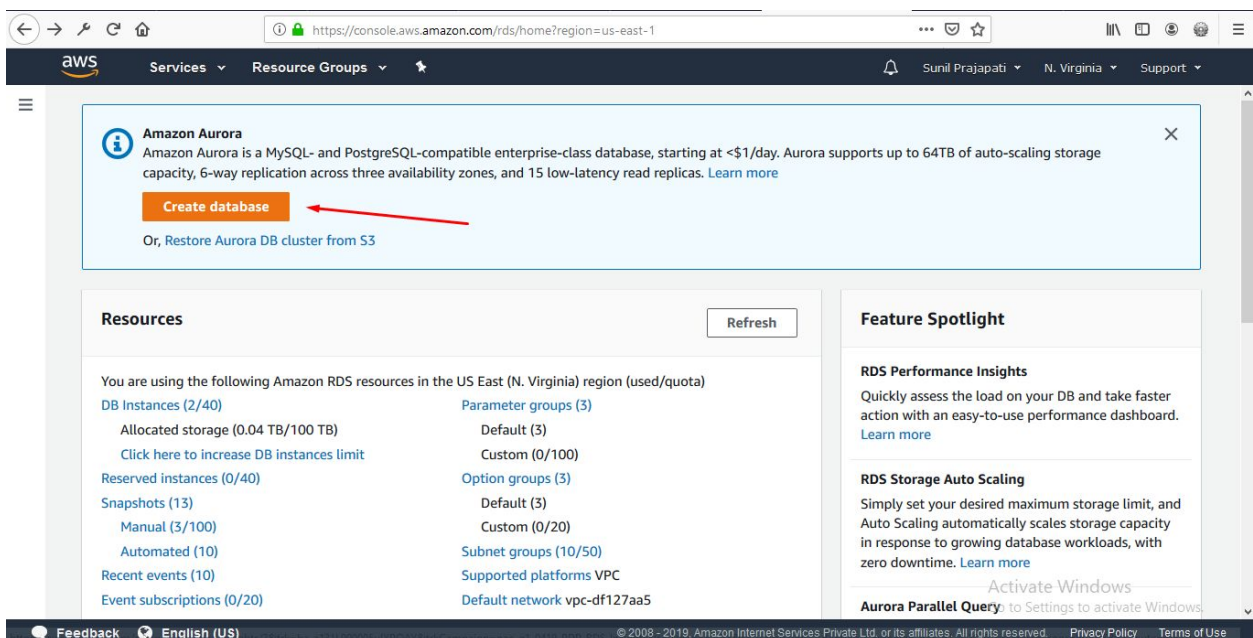
Connect Alexa to Database

Alexa=>LambdaFunction=>AWS RDS.

1.Click RDS(database) from Services tab



2.Click Create Database



Follow this steps to create Database

<https://aws.amazon.com/getting-started/tutorials/create-mysql-db/>

Note: Use sync-mysql to make synchronous queries to a mysql database