



HELLO ALEXA!

BUILD YOUR FIRST ALEXA SKILL

Sanmathi Naga | Sanmathi_SathyanarayanaNaga@intuit.com
Priyadarshini Rajendran | priyadarshini_rajendran@intuit.com

http://bit.ly/GHC18_HelloAlexa

 #GHC18

INTERACTION WITH ALEXA

Alexa ask *hello app* to say hello to Grace



http://bit.ly/GHC18_HelloAlexa

STEP 1: LOGIN TO DEVELOPER PORTAL

Login to developer portal at <https://developer.amazon.com/alexa> by clicking the “Sign In” button on the top right corner

amazon alexa

Your Alexa Consoles **Sign In**

Skill Builders Device Makers Products Programs Docs Blog

Introducing the Alexa Presentation Language (Preview)

Design and build rich visual experiences for tens of millions of Alexa devices with screens

[Learn more »](#)

Create Alexa Skills

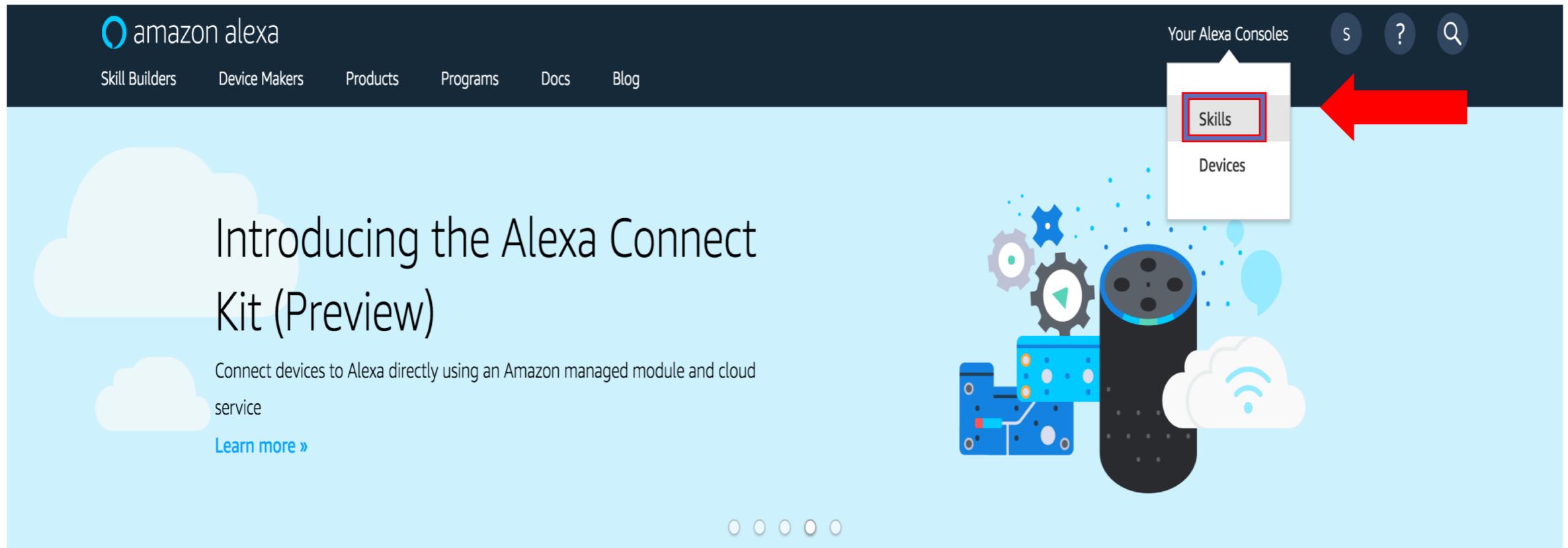
Build Alexa Devices

Explore Business Solutions

http://bit.ly/GHC18_HelloAlexa

STEP 2:

Hover on “Your Alexa Consoles” on the top right and then click on “Skills”.



http://bit.ly/GHC18_HelloAlexa

STEP 3:

Under the “Skills” tab click on click on “Create Skill” button

The screenshot shows the Alexa Skills Kit Developer Console interface. At the top, there's a navigation bar with the Alexa developer console logo, search, sort, and feedback forum icons. Below the header, a section titled "Make Money with Your Alexa Skills" provides information about premium content and in-skill purchasing. The main area is titled "Welcome to the Alexa Skills Kit Developer Console". It features a "Skills" tab (which is highlighted with a red box and arrow) and a "Payments" tab. A "Create Skill" button is located in the top right corner of the main content area, also highlighted with a red box and arrow. Below the tabs, there's a table header with columns: SKILL NAME, LANGUAGE, TYPE, MODIFIED, STATUS, and ACTIONS.

http://bit.ly/GHC18_HelloAlexa

STEP 4:

Provide the “Skill Name” as ***hello app*** and retain the Default Language to be English (US). Ensure the “Custom” model is selected and click on the “Create Skill” Button

alexa developer console

Create a new skill

Skill name: hello app (9/50 characters)

Default language: English (US)

More languages can be added to your skill after creation

Choose a model to add to your skill

There are many ways to start building a skill. You can design your own custom model or start with a pre-built model. Pre-built models are interaction models that contain a package of intents and utterances that you can add to your skill.

Custom SELECTED Design a unique experience for your users. A custom model enables you to create all of your skill's interactions. "Alexa, what's in the news?"	Smart Home Give users control of their news feed. This pre-built model lets users control what updates they listen to. "Alexa, turn on the kitchen lights"	Video Let users find and consume video content. This pre-built model supports content searches and content suggestions. "Alexa, play Interstellar"
--	---	---

Cancel Create skill

http://bit.ly/GHC18_HelloAlexa

STEP 5:

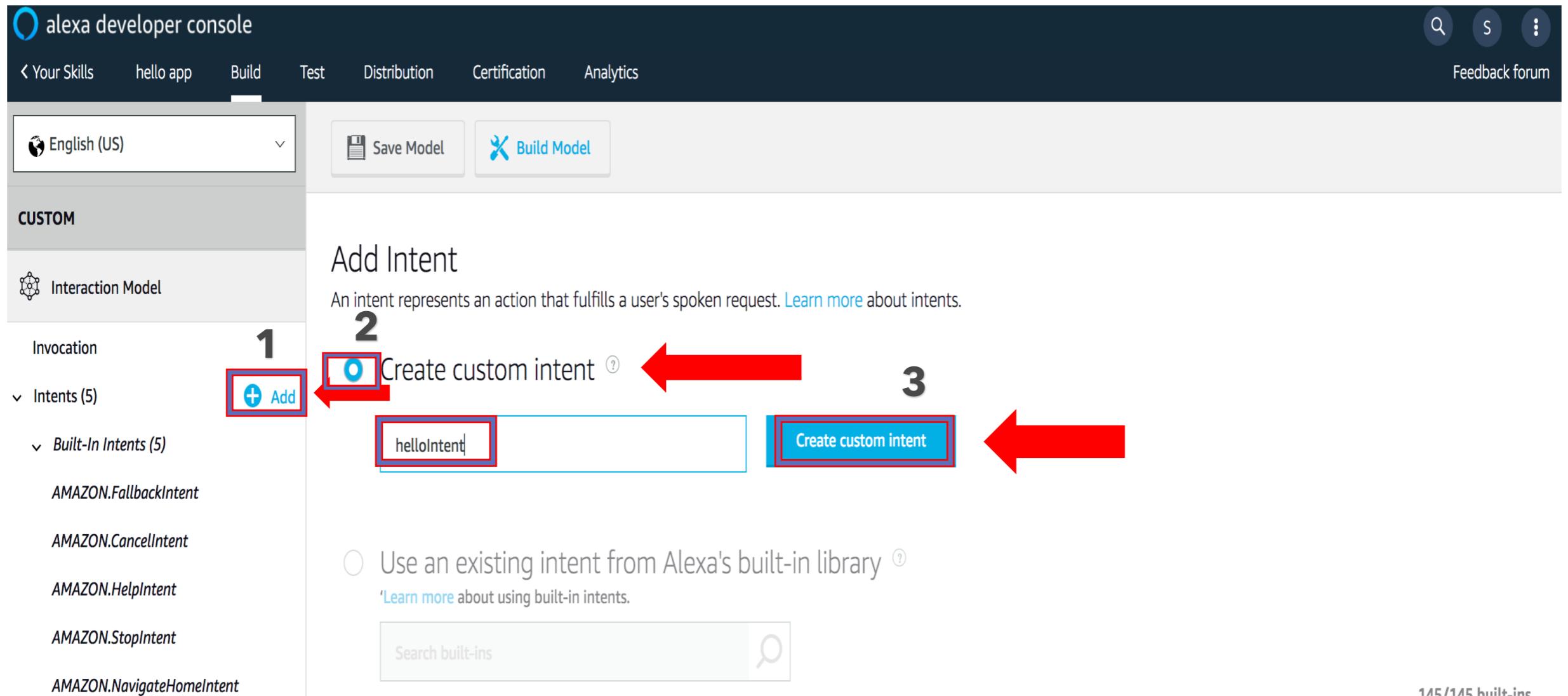
Ensure that “Start from scratch” template is selected and click on “Choose” button on the top right

The screenshot shows the Alexa developer console interface. At the top, there's a navigation bar with links for 'Your Skills', 'hello app', 'Build', 'Test', 'Distribution', 'Certification', and 'Analytics'. On the far right of the header are search, sort, and more options icons, along with a 'Feedback forum' link. Below the header, the main content area has a title 'Choose a template' and a subtitle 'Select a quick start template to get started with a predefined skill or simply "Start from scratch"'. There are four cards displayed: 1. 'Start from scratch' (selected): 'Design a unique experience for your users and define your custom model from scratch.' 2. 'Fact Skill': 'Provided a list of interesting facts about a topic, Alexa will select a fact at random and tell it to the user when the skill is invoked. Includes 1 custom intent, and 4 built-in intents.' 3. 'Quiz Game Skill': 'Provided a list of interesting facts about a topic, Alexa will quiz a user with facts from the list. Includes 1 custom intent with 1 slot, and 6 built-in intents.' 4. 'High-Low Game Skill': 'Try to guess the target number. Alexa tells the player if the target number is higher or lower than their current guess. Includes 2 custom intents with 5 slots, and 5 built-in intents.' A large red arrow points to the 'Start from scratch' card, and another red arrow points to the 'Choose' button in the top right corner of the main content area.

http://bit.ly/GHC18_HelloAlexa

STEP 6:

Click on “Add” next to Intents. Under “Create custom intent” provide the intent name as ***helloIntent*** and then click on the “Create custom intent” button.



http://bit.ly/GHC18_HelloAlexa

STEP 7:

Under “**Sample Utterances**” type in *say hello to {fname}* and press enter. This will create a “**Intent Slots**”.

Feel free to add more such sample utterances such as **wish {fname}**

The screenshot shows the Alexa developer console interface. On the left, there's a sidebar with options like 'Your Skills', 'hello app', 'Build' (which is selected), 'Test', 'Distribution', 'Certification', and 'Analytics'. Below that is a language dropdown set to 'English (US)'. At the top right are search, save, and build buttons, along with a 'Feedback forum' link.

The main area is titled 'Intents / helloIntent'. It shows a section for 'Sample Utterances (0)' with a text input field containing 'say hello to {fname}' and a red arrow pointing to it. Below this is a dropdown menu with options 'Select an Existing Slot' and 'No existing slots'. There are also buttons for 'Create a new slot' and 'OR'.

On the left side of the main area, under 'Intents (6)', the 'helloIntent' is selected and highlighted in blue. To its right is a '+' button for adding more intents. Below this are sections for 'Built-In Intents (5)' which include 'AMAZON.FallbackIntent', 'AMAZON.CancelIntent', and 'AMAZON.HelpIntent'.

At the bottom, there's a table for 'Intent Slots (1)'. It has columns for 'ORDER', 'NAME', 'SLOT TYPE', and 'ACTIONS'. One slot is listed with 'ORDER 1', 'NAME fname', and 'SLOT TYPE Names'. A red arrow points to the 'Names' dropdown. Another row is shown below with 'Create a new slot' and a 'Select a slot type' dropdown, with a red arrow pointing to it. The 'ACTIONS' column for each row includes 'Edit Dialog' and 'Delete' links.

http://bit.ly/GHC18_HelloAlexa

STEP 8:

Click on “Add” button next to the Slot Types and under Create custom slot type, create a type named **Names** and click on “Create custom slot type” button.

The screenshot shows the Alexa developer console interface. On the left, there's a sidebar with sections for 'Your Skills', 'hello app', 'Build', 'Test', 'Distribution', 'Certification', and 'Analytics'. Below these are dropdowns for 'English (US)' and buttons for 'Save Model' and 'Build Model'. The main area is titled 'CUSTOM' and 'Interaction Model'. Under 'Invocation', there's a list of intents: 'Intents (6)' (with 'helloIntent' and 'fname' listed), 'Built-In Intents (5)' (including 'AMAZON.FallbackIntent', 'AMAZON.CancelIntent', 'AMAZON.HelpIntent', 'AMAZON.StopIntent', and 'AMAZON.NavigateHomeIntent'). A red arrow labeled '1' points to the '+ Add' button next to 'Slot Types (0)'. The 'Slot Types (0)' section has a red box around it. Another red arrow labeled '2' points to the 'Names' input field. A third red arrow labeled '3' points to the 'Create custom slot type' button. The central part of the screen is titled 'Add Slot Type' and explains that slot types define how data in an intent slot is recognized and handled. It offers two options: 'Create custom slot type' (selected) and 'Use an existing slot type from Alexa's built-in library'. A search bar for 'Search built-ins' is shown, along with a table for 'List Types' (88 built-ins) with columns for 'Name' and 'Description'. A note says these slot types each represent a list of items. At the bottom right, it says '96/96 built-ins'.

http://bit.ly/GHC18_HelloAlexa

STEP 9:

Under the Slot Values add names. Eg: Adam, Grace etc

Slot Types / Names

Slot Values (3)

VALUE	ID (OPTIONAL)	SYNONYMS (OPTIONAL)
frank	Enter ID	Add synonym
adam	Enter ID	Add synonym
grace	Enter ID	Add synonym

Names

http://bit.ly/GHC18_HelloAlexa

STEP 10:

Click on “fname” under helloIntent on the Left Nav and select the “Slot Type” as “Names”

The screenshot shows the Alexa developer console interface. On the left, there's a navigation pane with sections like 'Your Skills', 'hello app', 'Build', 'Test', 'Distribution', 'Certification', and 'Analytics'. Below these are 'CUSTOM' and 'Interaction Model' sections. Under 'Invocation', there's a tree view with 'Intents (6)', 'helloIntent' (selected), and 'fname' (highlighted with a red box). To the right, the main area shows the 'Intents / helloIntent / fname' path. A 'Slot Type' section is open, displaying a dropdown menu with 'Names' selected (also highlighted with a red box). Other options in the dropdown include 'AZON.Actor', 'AMAZON.AdministrativeArea', 'AMAZON.AggregateRating', and 'AMAZON.Airline'. A note at the bottom says 'Turn the Dialog.Delegate directive in your skill'. Red arrows point from the 'Names' button in the navigation to the 'Names' button in the dropdown menu.

1

Custom

Your Skills hello app Build Test Distribution Certification Analytics

English (US)

Save Model Build Model

CUSTOM

Interaction Model

Invocation

Intents (6) + Add

helloIntent

1 fname

Built-In Intents (5)

AMAZON.FallbackIntent

AMAZON.CancelIntent

AMAZON.HelpIntent

AMAZON.StopIntent

Intents / helloIntent / fname

Slot Type

Select a slot type

Names 2

AZON.Actor

AMAZON.AdministrativeArea

AMAZON.AggregateRating

AMAZON.Airline

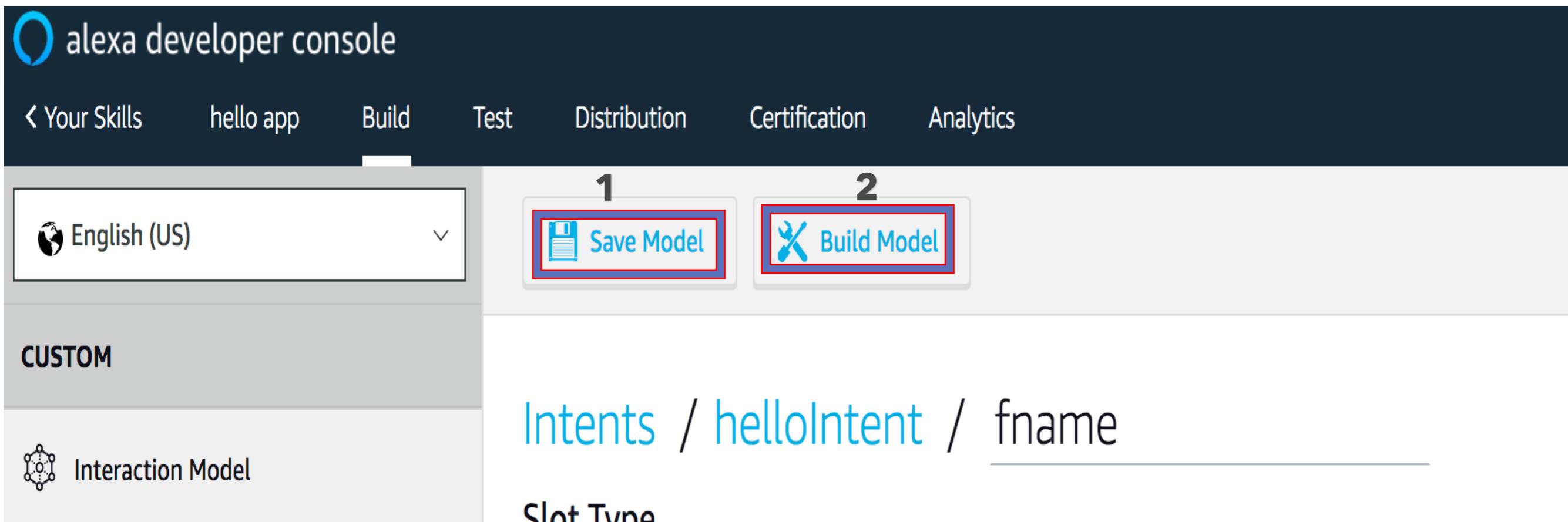
Is this slot required to fulfill the intent?

Turn the Dialog.Delegate directive in your skill

http://bit.ly/GHC18_HelloAlexa

STEP 11:

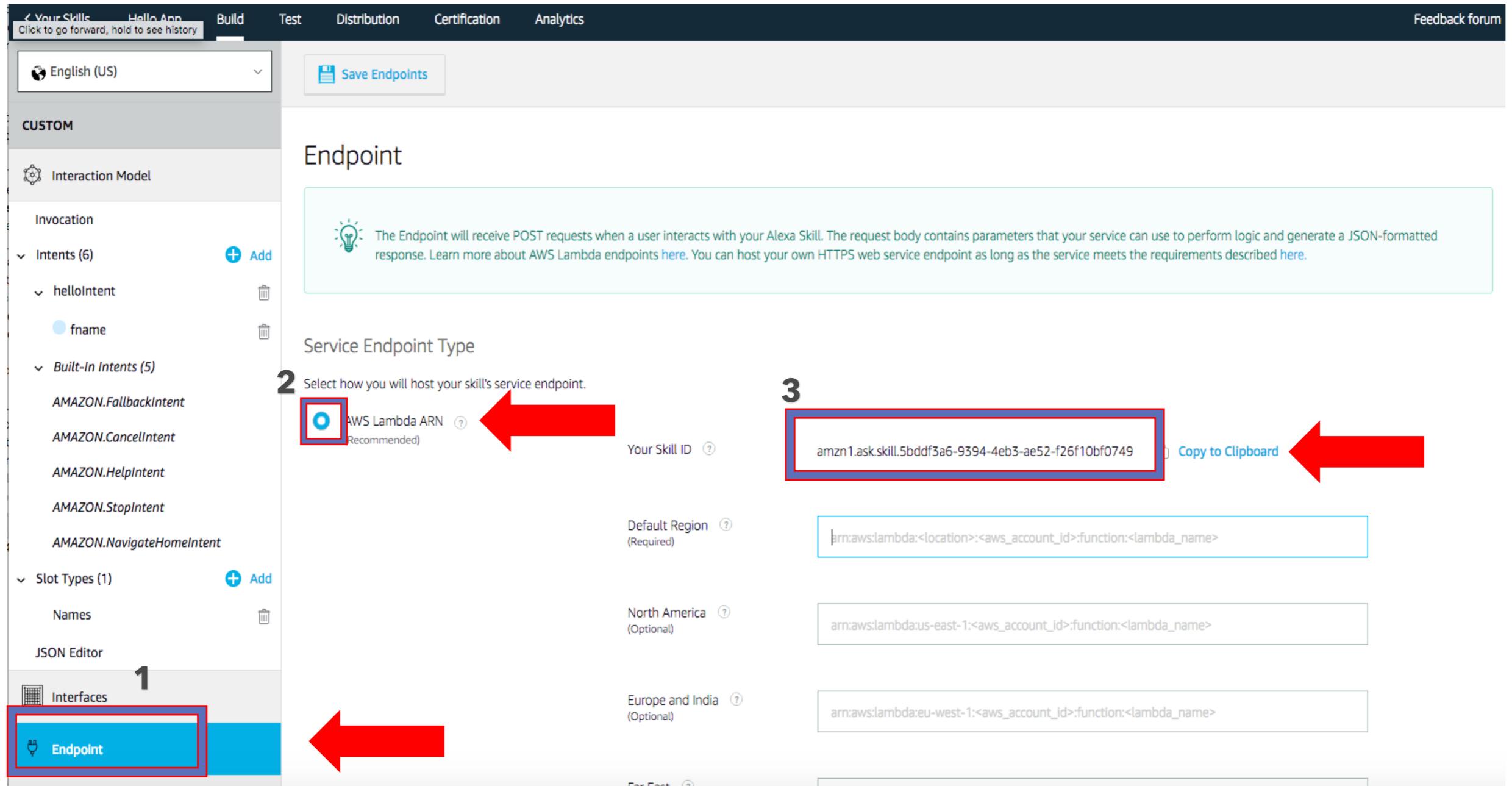
Click on “Save Model” button. Once the Interaction Model is Saved Successfully, click on “Build Model”



http://bit.ly/GHC18_HelloAlexa

STEP 12:

Once the Build is successful click on “Endpoint” and click on the “AWS Lambda ARN” then copy the “Skill Id”



http://bit.ly/GHC18_HelloAlexa

RECAP

- Skill “hello app”
- Intent - “helloIntent”
- Sample Utterance
- Slot - “fname”
- Custom Slot Type - “Names”
- Skill Id

http://bit.ly/GHC18_HelloAlexa

STEP 13:

Sign into your AWS account using the following link.

<https://signin.aws.amazon.com/>

http://bit.ly/GHC18_HelloAlexa



STEP 14:

Once in the portal ensure you are on “**N.Virginia**” or any of the regions that support lambda function for custom skills. Then type in “**Lambda**” under the AWS Service and select it.

The screenshot shows the AWS Management Console interface. At the top, there's a navigation bar with the AWS logo, 'Services' dropdown, 'Resource Groups' dropdown, and a search icon. To the right of the search icon are 'San' (with a bell icon), 'N. Virginia' (which is highlighted with a red box and has a red arrow pointing to it), and 'Support' dropdown. Below the navigation bar is a sidebar titled 'AWS services'. Inside the sidebar, the 'Lambda' service is highlighted with a red box and a red arrow pointing to it. Other services listed include 'Lambda', 'Amazon Lex', 'CodeBuild', and 'IoT 1-Click'. To the right of the sidebar is a 'Helpful tips' section with two items: 'Manage your costs' (with a mail icon) and 'Create an organization' (with a hexagon icon). The 'Manage your costs' item includes a link to 'Start now'.

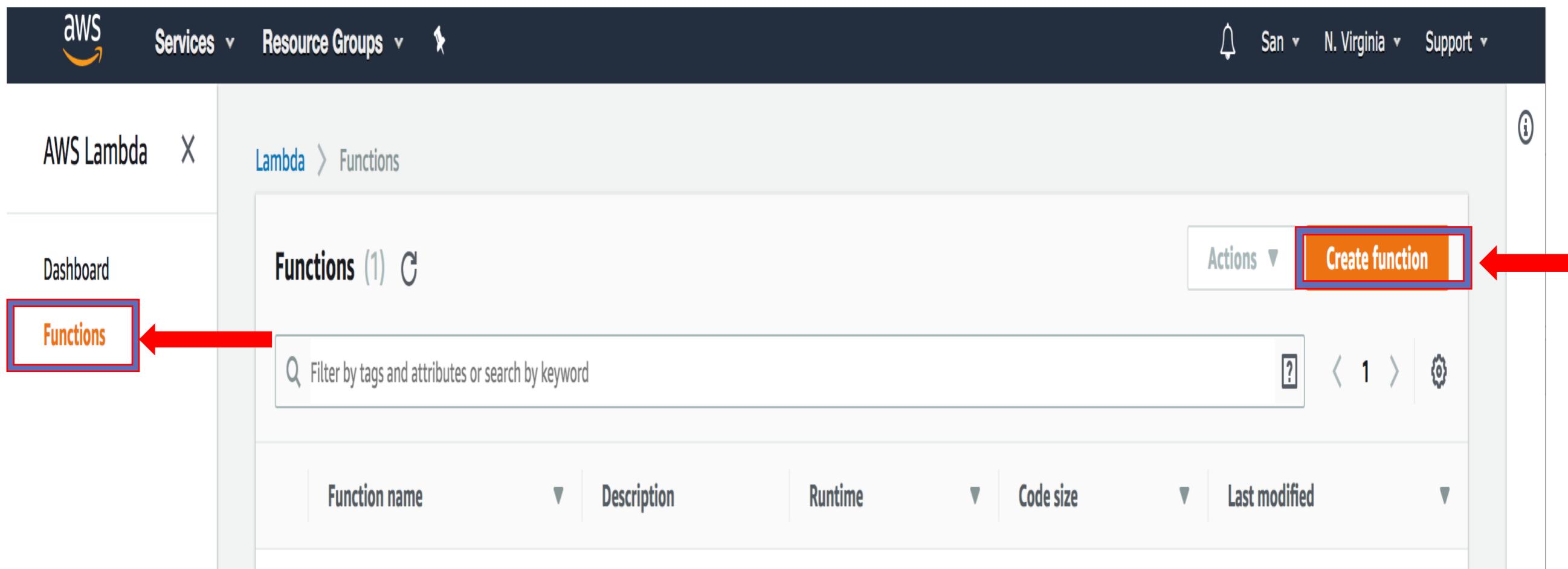
Eg:

- Asia Pacific(Tokyo)
- EU (Ireland)
- US East (N Virginia)
- US West (Oregon)

http://bit.ly/GHC18_HelloAlexa

STEP 15:

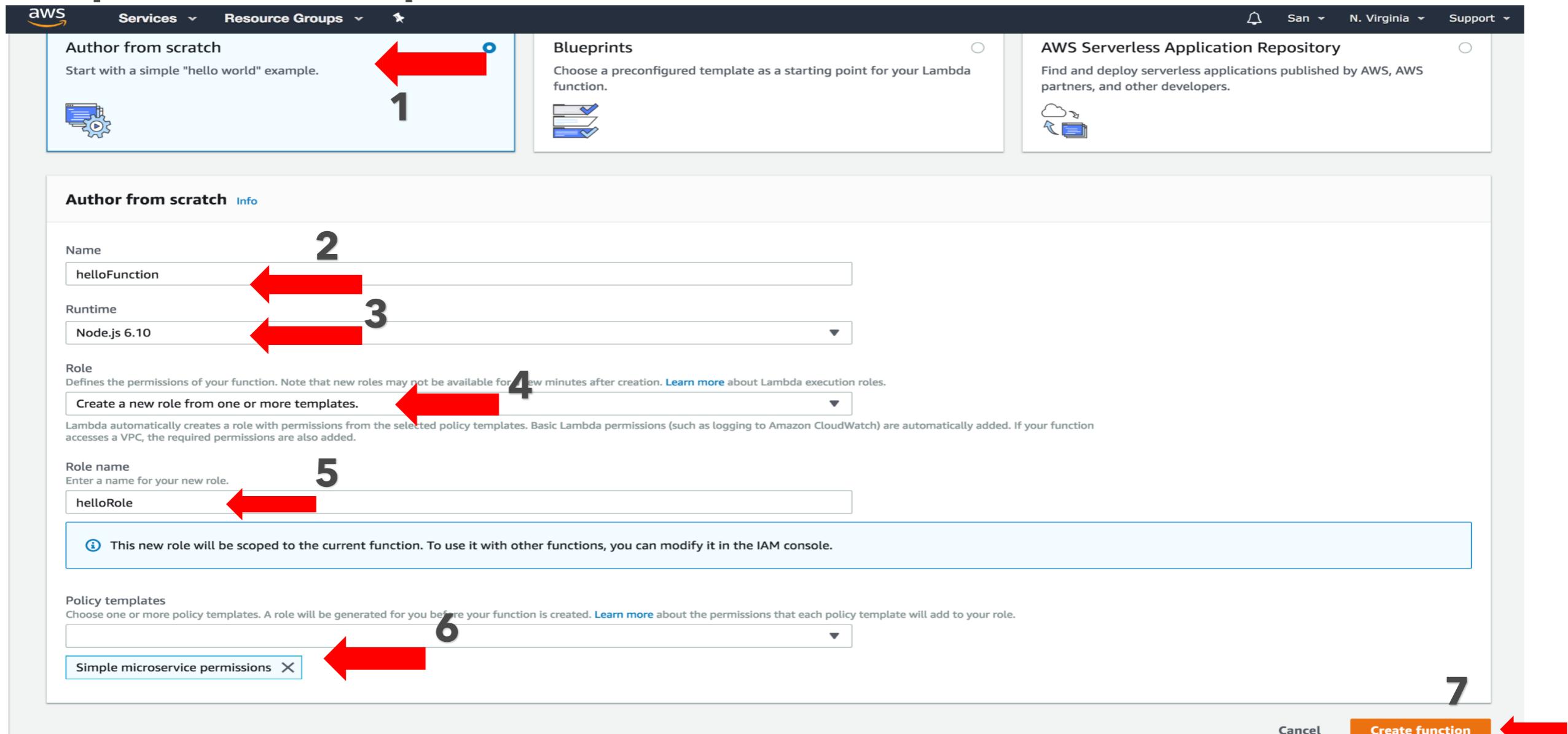
Once in the AWS Lambda portal, ensure “**Functions**” is selected in the left nav and then click on the “**Create function**” button



http://bit.ly/GHC18_HelloAlexa

STEP 16:

Once in create function select “**Author from scratch**” and then give the name of the function as ***helloFunction***. Select the runtime env as ***Node.js***, select Role as ***Create new role from template(s)***, name your role as ***helloRole*** and pick the Policy Template to be “**Simple Microservice permissions**”. Then click on “**Create function**” button.



http://bit.ly/GHC18_HelloAlexa

STEP 17:

Make sure you are in the “Configuration” tab and select “Alexa Skill Kit” from the left nav.

The screenshot shows the AWS Lambda console interface. At the top, the navigation bar includes the AWS logo, Services dropdown, Resource Groups dropdown, a bell icon for notifications, location dropdown (San Francisco, N. Virginia), and Support dropdown. Below the navigation, the path is Lambda > Functions > helloFunction. To the right, the ARN is listed as arn:aws:lambda:us-east-1:016222001512:function:helloFunction. The main area shows the 'helloFunction' configuration with tabs for Throttle, Qualifiers, Actions, Select a test event, Test, and Save. A success message states: "Congratulations! Your Lambda function 'helloFunction' has been successfully created. You can now change its code and configuration. Choose Test to input a test event when you want to test your function." A red box highlights the 'Configuration' tab. In the 'Designer' section, there's a 'Add triggers' sidebar with options like API Gateway, AWS IoT, Alexa Skills Kit (which is highlighted with a red box and has a red arrow pointing to it), Alexa Smart Home, CloudFront, and CloudWatch Events. The main panel shows the function name 'helloFunction' with a code icon, and associated resources: Amazon CloudWatch Logs and Amazon DynamoDB. A dashed box indicates where other resources can be added.

http://bit.ly/GHC18_HelloAlexa

STEP 18:

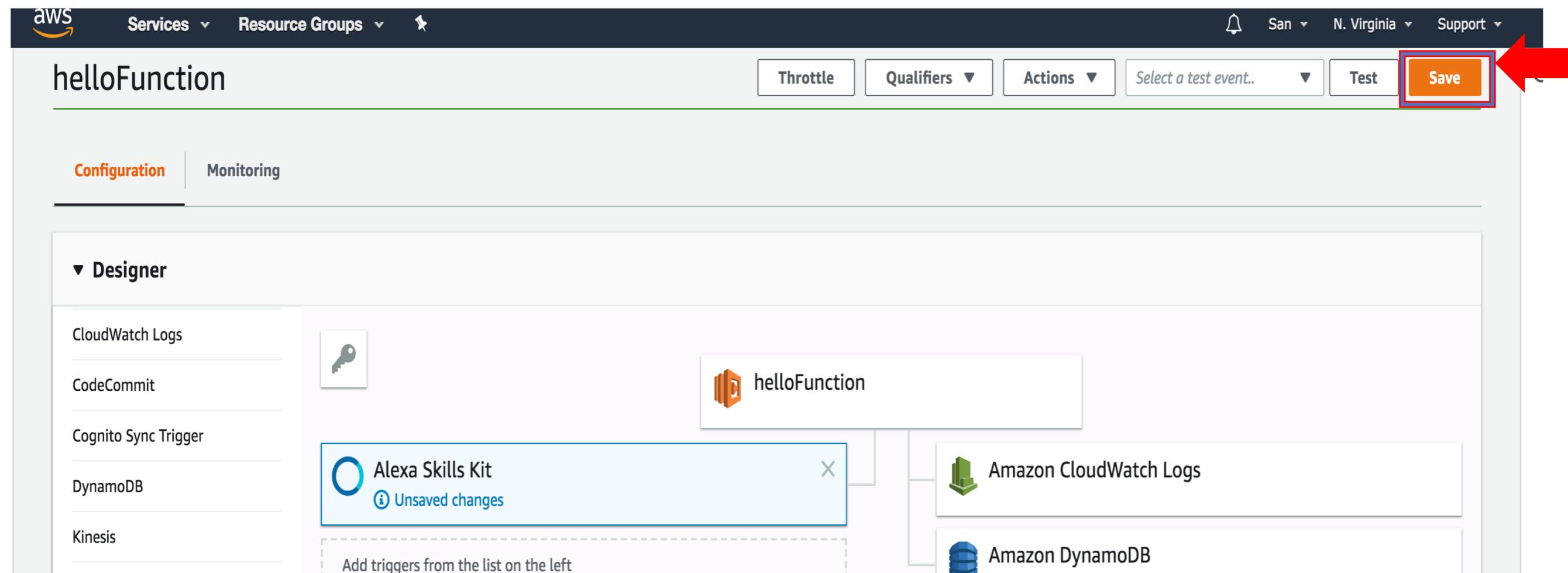
Select “**Alexa Skills Kit**”. Under Configure triggers select “**Enabled**” under the Skill Id verification and provide the Skill Id that we got from Step 13 and click on “**Add**” button.

The screenshot shows the AWS Lambda console for a function named "helloFunction". In the "Configure triggers" section, there is a list of available triggers on the left: Cognito Sync Trigger, DynamoDB, Kinesis, S3, SNS, and SQS. On the right, there are three resources listed: Throttle, Qualifiers, Actions, Select a test event..., Test, and Save. Below the triggers, a box contains the text "Add triggers from the list on the left". An "Alexa Skills Kit" trigger is selected, indicated by a blue border and the text "Configuration required". A red arrow labeled "1" points to this trigger. In the "Skill ID verification" section, there are two radio buttons: "Enable (recommended)" (selected) and "Disable". A red arrow labeled "2" points to the "Enable" button. Below it, a text input field contains the Skill ID "amzn1.ask.skill.728b3ea7-1b1e-4514-a5e8-19f71106a497", with a red arrow labeled "3" pointing to it. At the bottom of the "Configure triggers" section, a note states: "Lambda will add the necessary permissions for Amazon Alexa to invoke your Lambda function from this trigger. Learn more about the Lambda permissions model." To the right of this note are "Cancel" and "Add" buttons, with a red arrow labeled "4" pointing to the "Add" button.

http://bit.ly/GHC18_HelloAlexa

STEP 19:

Click on “Save” button on the top right corner



http://bit.ly/GHC18_HelloAlexa

STEP 20:

Click on the “**helloFunction**”, then you will get the Function Code Tab below where in you will be able to enter the code.

The screenshot shows the AWS Lambda console for a function named "helloFunction". In the top navigation bar, there are tabs for "Throttle", "Qualifiers", "Actions", "Select a test event..", "Test", and "Save". On the left sidebar, there's a "Cloudwatch Logs" section with links to "CodeCommit", "Cognito Sync Trigger", "DynamoDB", "Kinesis", "S3", "SNS", and "SQS". The main area displays a trigger configuration for "helloFunction". It shows an "Alexa Skills Kit" trigger with a "Saved" status, and two resources it connects to: "Amazon CloudWatch Logs" and "Amazon DynamoDB". Below this, a dashed box indicates where "Add triggers from the list on the left" can be done. A second red arrow points to the "Function code" tab at the top of the large code editor window. The code editor itself has tabs for "File", "Edit", "Find", "View", "Goto", "Tools", and "Window". The "Environment" panel on the left shows a folder "helloFunction" containing an "index.js" file. The code in "index.js" is as follows:

```
1 exports.handler = (event, context, callback) => {
2     // TODO implement
3     const response = {
4         statusCode: 200,
5         body: JSON.stringify('Hello from Lambda!')
6     };
7     callback(null, response);
8 };
```

http://bit.ly/GHC18_HelloAlexa

CODE....



- Launch Request:
Alexa ask *hello app*

- Intent Request:
Alexa ask hello app to say hello to Grace

http://bit.ly/GHC18_HelloAlexa

REQUEST:

```
{  
  "session": {  
    "sessionId": "sessionId",  
  },  
  "application": {  
    "applicationId": "amzn1.ask.skill.2e662fe0-51bb-4029-81dc-7521a74d1cc2"  
  },  
  "version": "1.0",  
  "request": {  
    "intent": {  
      "name": "helloIntent" ←  
      "slots": {  
        "fname": {  
          "name": "fname", ←  
          "value": "Grace"  
        }  
      },  
      "type": "IntentRequest", ←  
      "requestId": "requestId"  
    }  
  }  
}
```

http://bit.ly/GHC18_HelloAlexa

RESPONSE:

```
{  
  "version": "1.0",  
  "response": { ←  
    "outputSpeech": { ←  
      "type": "PlainText",  
      "text": "Hello Grace. Welcome to Grace Hopper!"  
    },  
    "shouldEndSession": true  
}
```

http://bit.ly/GHC18_HelloAlexa

LAUNCH REQUEST

```
'use strict';
exports.handler = (event, context) => {
  var request = event.request;
  if(request.type === "LaunchRequest") {
    let response = getResponse({
      output: "Welcome to Hello App.
      Would you like to say Hello to someone?",
      reprompt: true,
      endSession: false
    });
    context.succeed(response);
  } else {
    context.fail("Unknown intent name");
  }
}
```

```
function getResponse (opts) {
  var response = {
    version: "1.0",
    response: {
      outputSpeech: {
        type: "PlainText",
        text: opts.output
      },
      shouldEndSession: opts.endSession
    }
  };
  if(opts.reprompt) {
    response.response.reprompt = {
      outputSpeech: {
        type: "PlainText",
        text: "Would you like to wish someone?"
      }
    };
  }
  return response;
}
```

http://bit.ly/GHC18_HelloAlexa

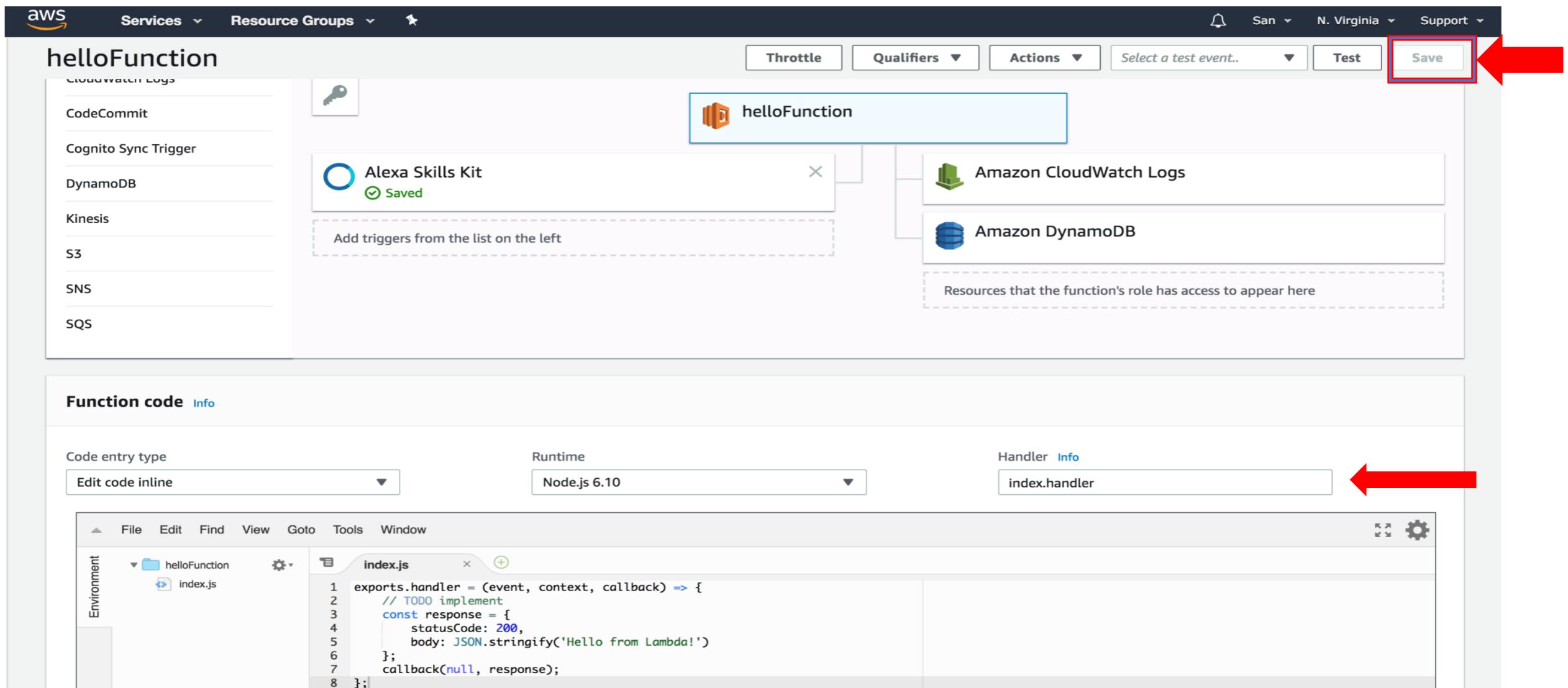
INTENT REQUEST

```
else if(request.type === "IntentRequest") {  
  
    if(request.intent.name === "helloIntent") {  
        let name = request.intent.slots.fname.value;  
        let output = "Hello " + name + ". Welcome to  
        Grace Hopper!";  
        let response = getResponse({  
            output: output,  
            reprompt: false,  
            endSession: true  
        });  
        context.succeed(response);  
    }  
    else {  
        context.fail("Unknown intent name");  
    }  
}
```

http://bit.ly/GHC18_HelloAlexa

STEP 21:

Once done with coding press the “Save” button on the top right corner.



http://bit.ly/GHC18_HelloAlexa

STEP 22:

Then click on the “**Test**” button that is present next to the “Save” button on the top right and provide the “**Event Name**” in the dialogue as ***test*** and provide the ***hello-alexা-test.json*** and click on the “**Create**” button.

Configure test event X

A function can have up to 10 test events. The events are persisted so you can switch to another computer or web browser and test your function with the same events.

Create new test event 

Edit saved test events

Event template

Hello World ▼

Event name

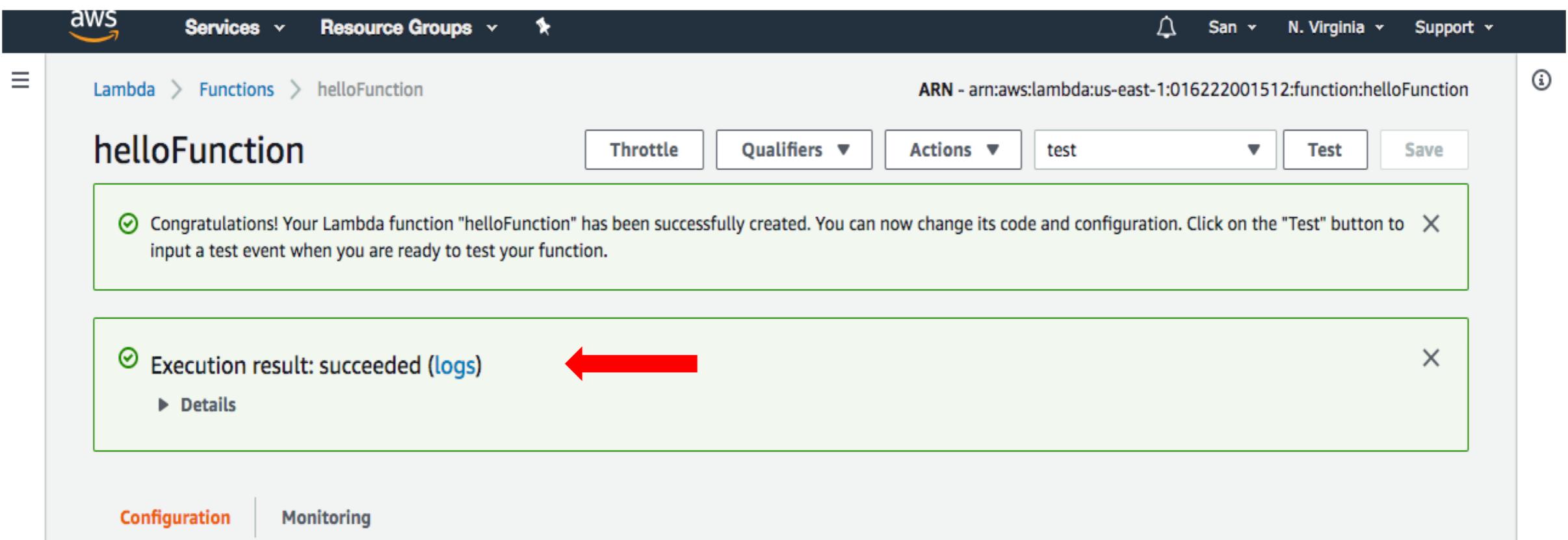


```
1 {  
2   "key1": "value1",  
3   "key2": "value2",  
4   "key3": "value3"  
5 }
```

http://bit.ly/GHC18_HelloAlexa

STEP 23:

Select the saved test event and click on “**Test**” button which will provide you with a sample response payload.

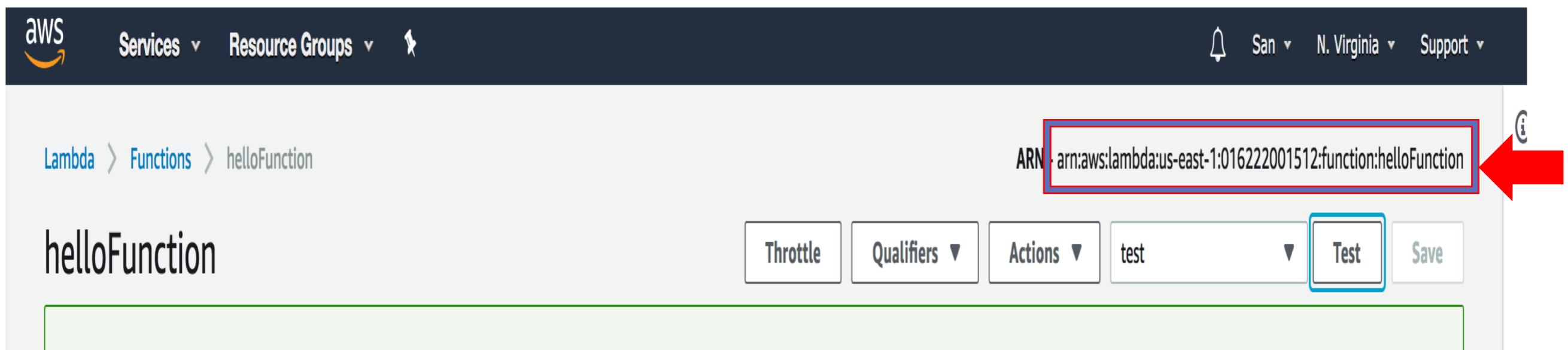


The screenshot shows the AWS Lambda console interface. At the top, there's a navigation bar with the AWS logo, 'Services', 'Resource Groups', and other account settings. Below the navigation, the path 'Lambda > Functions > helloFunction' is shown, along with the ARN of the function. The main area is titled 'helloFunction' and contains two green notification boxes. The first box says 'Congratulations! Your Lambda function "helloFunction" has been successfully created. You can now change its code and configuration. Click on the "Test" button to input a test event when you are ready to test your function.' The second box says 'Execution result: succeeded (logs)' with a 'Details' link. A red arrow points to the 'Logs' link in this second box. At the bottom, there are tabs for 'Configuration' (which is selected) and 'Monitoring'.

http://bit.ly/GHC18_HelloAlexa

STEP 24:

Copy the lambda **ARN** from the top right corner.



http://bit.ly/GHC18_HelloAlexa

STEP 25:

Go back to the Alexa developer console at <https://developer.amazon.com/>

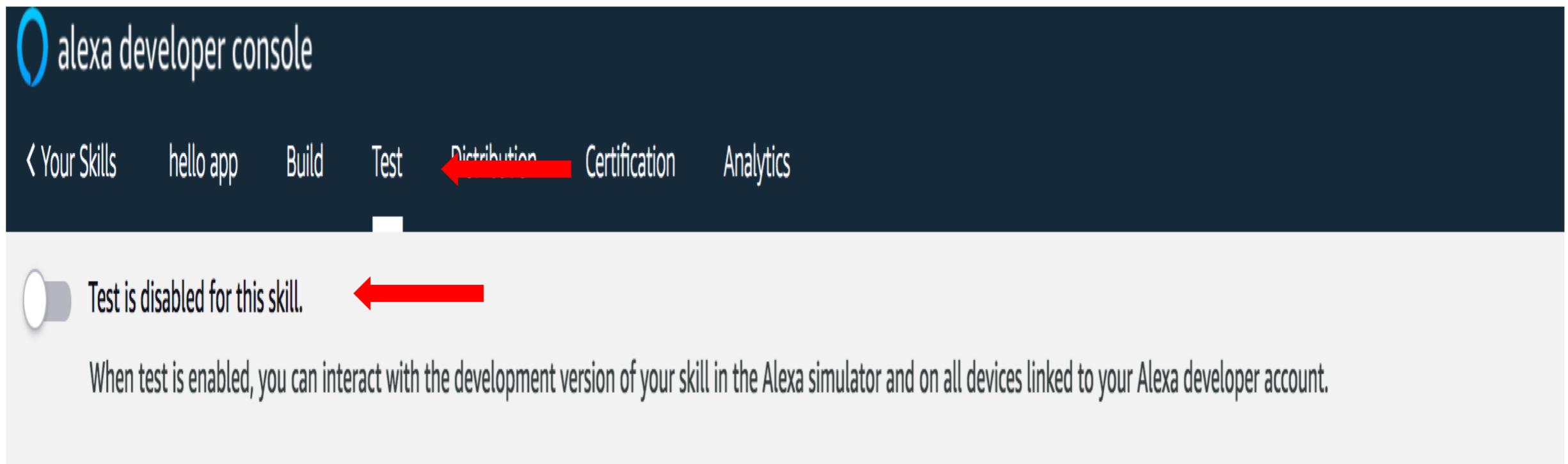
Select the “**Endpoint**” on the Left Nav and select “**AWS Lambda ARN**” and paste the ARN that we copied in the previous step into the “**Default Region**”. Then click on “**Save Endpoints**”

The screenshot shows the Alexa developer console interface. The left sidebar has tabs: Your Skills, hello app, Build, Test, Distribution, Certification, Analytics, and Feedback forum. The 'Build' tab is selected. The main area has a 'CUSTOM' section with an 'Interaction Model' tab. Under 'Invocation', there are sections for 'Intents (6)', 'helloIntent' (with 'fname' slot), and 'Built-In Intents (5)' (including AMAZON.FallbackIntent, AMAZON.CancelIntent, AMAZON.HelpIntent, AMAZON.StopIntent, and AMAZON.NavigateHomeIntent). There are also 'Slot Types (1)' and 'JSON Editor' sections. The 'Endpoint' tab is highlighted in blue. The right side shows the 'Endpoint' configuration. A red arrow labeled '1' points to the 'Endpoint' tab in the sidebar. A red arrow labeled '2' points to the 'AWS Lambda ARN (Recommended)' radio button. A red arrow labeled '3' points to the 'Default Region (Required)' dropdown containing 'arn:aws:lambda:us-east-1:016222001512:function:helloFunction'. A red arrow labeled '4' points to the 'Save Endpoints' button.

http://bit.ly/GHC18_HelloAlexa

STEP 26:

Click on the “**Test**” tab on the top nav and **enable** the **Test**.



http://bit.ly/GHC18_HelloAlexa

STEP 27: TEST LAUNCH REQUEST

Either you can ask Alexa or type in the test phrase “**ask hello app**” and should get the right response back from your Lambda.

The screenshot shows the Alexa developer console interface. At the top, there are tabs for Your Skills, hello app, Build, Test, Distribution, Certification, and Analytics. The Test tab is selected. Below the tabs, there are checkboxes for Skill I/O (checked), Echo Show Display (checked), Echo Spot Display (checked), and Device Log (unchecked). The Alexa Simulator tab is selected, followed by Manual JSON and Voice & Tone. The language is set to English (US). A microphone icon indicates that audio input is available. In the center, a blue speech bubble says "Welcome to Hello App. Would you like to say Hello to someone?". Below it, the "Skill I/O" section shows the JSON Input and JSON Output. The JSON Input is a complex object with many nested fields, including session, application, user, context, and device. The JSON Output is also a complex object, containing a body with version 1.0, outputSpeech (PlainText, text: "Welcome to Hello App."), reprompt (outputSpeech: PlainText, text: "Would you like to"), and shouldEndSession (false).

```
JSON Input:
1  {
2   "version": "1.0",
3   "session": {
4     "new": true,
5     "sessionId": "amzn1.echo-api.session.e
6     "application": {
7       "applicationId": "amzn1.ask.skill.
8     },
9     "user": {
10      "userId": "amzn1.ask.account.AFGG7.
11    }
12  },
13  "context": {
14    "System": {
15      "application": {
16        "applicationId": "amzn1.ask.sk
17    },
18    "user": {
19      "userId": "amzn1.ask.account.A
20    },
21    "device": {
22      "deviceId": "amzn1.ask.device.
23      "supportedInterfaces": []
24    },
25    "apiEndpoint": "https://api.amazon
26    "apiAccessToken": "eyJ0eXAiOiJKV1Q
}

JSON Output:
1  {
2   "body": {
3     "version": "1.0",
4     "response": {
5       "outputSpeech": {
6         "type": "PlainText",
7         "text": "Welcome to Hello App."
8       },
9       "reprompt": {
10         "outputSpeech": {
11           "type": "PlainText",
12           "text": "Would you like to"
13         }
14       }
15     },
16     "shouldEndSession": false
17   }
18 }
```

http://bit.ly/GHC18_HelloAlexa

STEP 28: TEST INTENT REQUEST

To test intent request, you can either ask Alexa or type in the test phrase **“ask hello app to say hello to Grace”** and should get the right response back from your Lambda.

http://bit.ly/GHC18_HelloAlexa

ALEXA CARD



```
else if(request.type === "IntentRequest") {
  if(request.intent.name === "helloIntent") {
    let name = request.intent.slots.fname.value;
    let output = "Hello " + name + ". Welcome to
                  Grace Hopper!";
    let response = getResponse({
      output: output,
      reprompt: false,
      cardTitle: name,
      endSession: true
    });
    context.succeed(response);
}
```

```
if(opts.cardTitle) {
  let cardTitle = "Hello " + opts.cardTitle;
  response.response.card = {
    type: "Standard",
    title: cardTitle,
    text: "Welcome to Grace Hopper",
    image :{
      smallImageUrl:
      "https://www.serendipitystamps.com/mm5/pics/s
       tamps/619hellosmall.gif",
      largeImageUrl:
      "https://www.serendipitystamps.com/mm5/pics/s
       tamps/619hellosmall.gif"
    }
}
```

ALEXA CARD

 #GHC 18

Echo Show Display



Echo Spot Display

