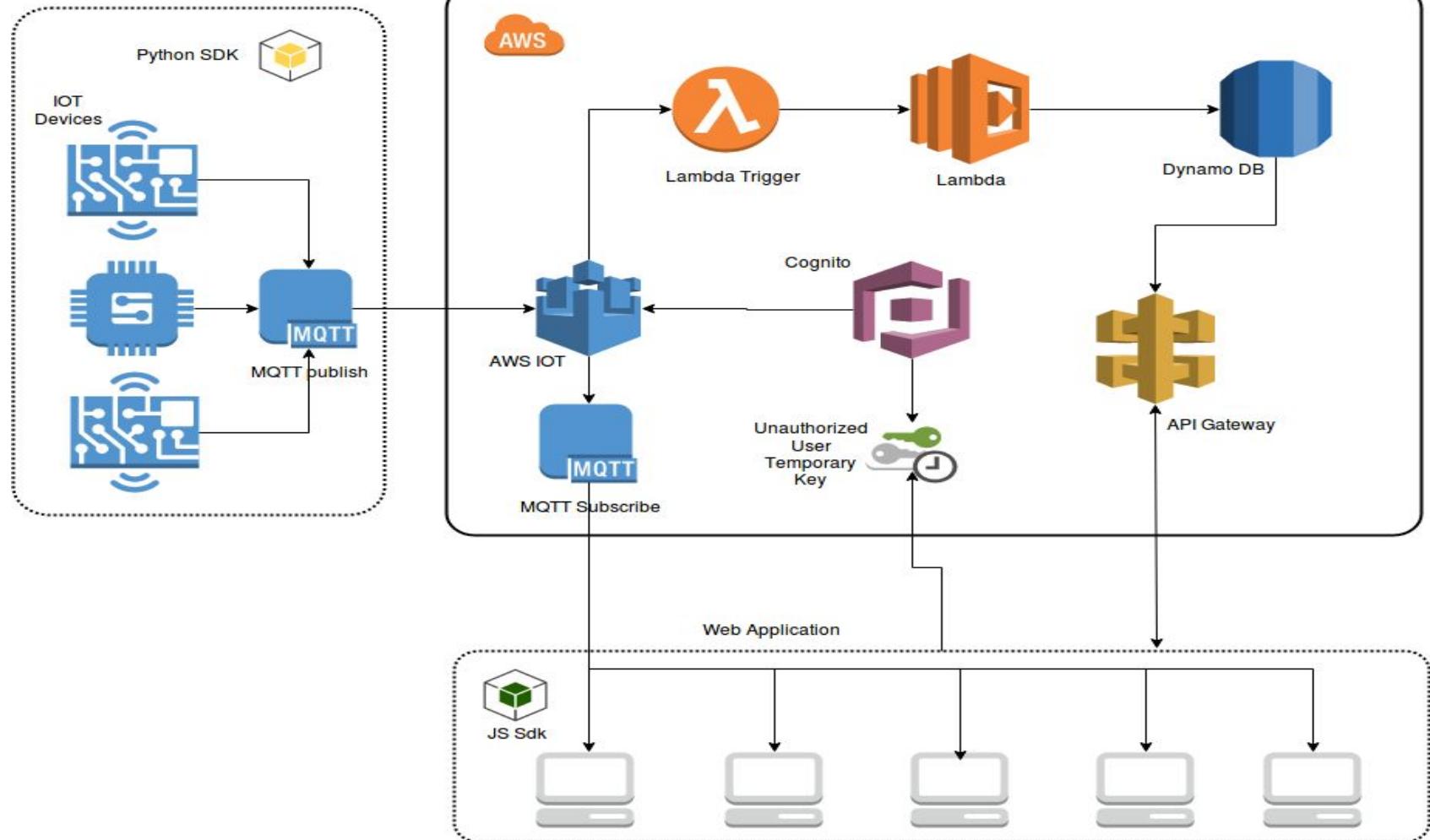


Smart bin Simulation

Tracking bin level and routing

Work flow



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”

The screenshot shows the AWS Home page. At the top, there's a navigation bar with 'Services', 'Resource Groups', a search bar, and account information for 'Oregon'. Below the navigation bar is a 'Featured next steps' section with 'Manage your costs' and 'Get best practices' options. The main content area is titled 'AWS services' and has a search bar. It features a 'Build a solution' section with six quick-start guides: 'Launch a virtual machine', 'Build a web app', 'Deploy a serverless microservice', 'Host a static website', 'Create a backend for your mobile app', and 'Register a domain'. At the bottom, there's a 'Learn to build' section with a 'See all' link.

AWS services

Find a service by name (for example, EC2, S3, Elastic Beanstalk).

> All services

Build a solution

Get started with simple wizards and automated workflows.

Launch a virtual machine
With EC2
~1 minute

Build a web app
With Elastic Beanstalk
~6 minutes

Deploy a serverless microservice
With Lambda, API Gateway
~2 minutes

Host a static website
With S3, CloudFront, Route 53
~5 minutes

Create a backend for your mobile app
With Mobile Hub
~5 minutes

Register a domain
With Route 53
~3 minutes

Learn to build

See all

Featured next steps

Manage your costs
Get real-time billing alerts based on your cost and usage budgets. [Start now](#)

Get best practices
Use AWS Trusted Advisor for security, performance, cost and availability best practices. [Start now](#)

What's new?

Announcing AWS Batch
Now generally available, AWS Batch enables developers, scientists, and engineers to process large-scale batch jobs with ease. [Learn more](#)

Announcing Amazon Lightsail

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

The screenshot shows the AWS DynamoDB service dashboard. At the top, there are navigation links for Services, Resource Groups, and a bell icon. On the left, a sidebar menu includes options for Dashboard, Tables, and Reserved capacity, with 'Create table' highlighted by a red box. The main content area features a brief introduction to Amazon DynamoDB and a prominent 'Create table' button. Below this, sections for 'Recent alerts' (showing none) and 'Total capacity for Asia Pacific (Singapore)' (with 30 provisioned read and write units) are displayed. A 'View all in CloudWatch' link is also present. The bottom section, 'Service health', indicates that the service is operating normally.

DynamoDB

Services ▾ Resource Groups ▾

Create table

Amazon DynamoDB is a fully managed non-relational database service that provides fast and predictable performance with seamless scalability.

Create table

Recent alerts

No CloudWatch alarms have been triggered.

[View all in CloudWatch](#)

Total capacity for Asia Pacific (Singapore)

Provisioned read capacity: 30 Reserved read capacity: 0
Provisioned write capacity: 30 Reserved write capacity: 0

Service health

Current Status	Details
Amazon DynamoDB (Singapore)	Service is operating normally

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.

Create DynamoDB table

[Tutorial](#)

DynamoDB is a schema-less database that only requires a table name and primary key. The table's primary key is made up of one or two attributes that uniquely identify items, partition the data, and sort data within each partition.

Table name* ⓘ

Primary key* Partition key

Numb ▾

 Add sort key

Table settings

Default settings provide the fastest way to get started with your table. You can modify these default settings now or after your table has been created.



Use default settings

- No secondary indexes.
- Provisioned capacity set to 5 reads and 5 writes.
- Basic alarms with 80% upper threshold using SNS topic "dynamodb".



You do not have the required role to enable Auto Scaling by default.

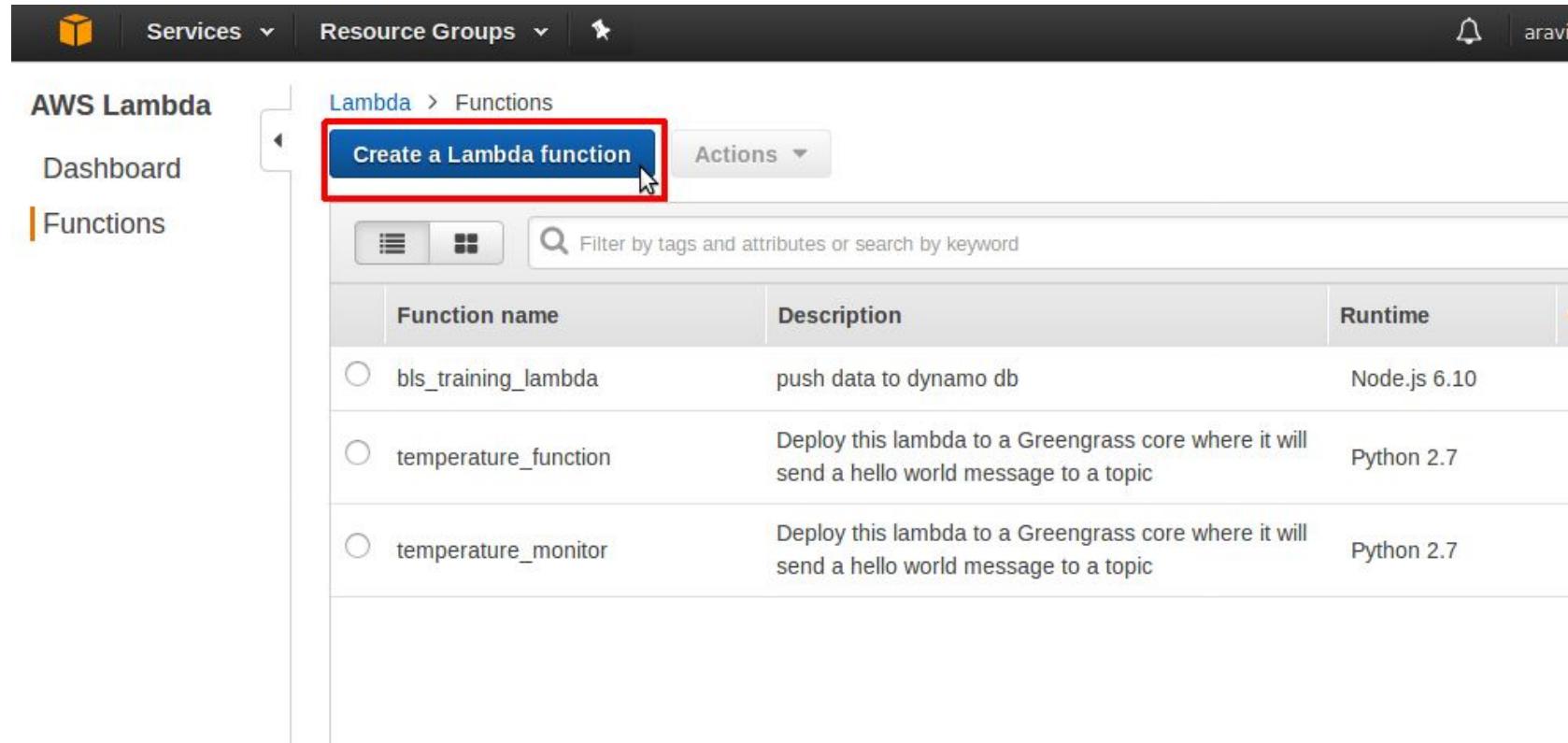
Please refer to [documentation](#).

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

The screenshot shows the AWS DynamoDB console interface. On the left, the navigation bar includes 'Services' (selected), 'Resource Groups', and a user icon. The main menu has 'DynamoDB' selected, with options like 'Dashboard', 'Tables' (which is currently active and highlighted in orange), and 'Reserved capacity'. A 'Create table' button is visible at the top right of the main content area. The central part of the screen displays a list of tables under 'Tables'. One table, 'bls_demo_data', is selected and highlighted with a red box around its name in the list. To the right, the details for the 'bls_demo_data' table are shown. The 'Items' tab is selected, and a 'Create item' button is visible. Below it, a search bar shows 'Scan: [Table] bls_demo_data: binId'. A note below the search bar states: 'An item consists of one or more attributes. Each attribute consists of a read or write an item, the only attributes that are required are those tha...'. The table's schema is partially visible, showing an attribute named 'binId'.

Lambda Function

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



The screenshot shows the AWS Lambda service interface. The top navigation bar includes 'Services', 'Resource Groups', and a user profile. The left sidebar has 'AWS Lambda' selected, with 'Dashboard' and 'Functions' also listed. The main content area is titled 'Lambda > Functions'. A prominent blue button labeled 'Create a Lambda function' is highlighted with a red box and a cursor arrow pointing to it. Below the button is a search bar with the placeholder 'Filter by tags and attributes or search by keyword'. A table lists existing Lambda functions:

Function name	Description	Runtime
bls_training_lambda	push data to dynamo db	Node.js 6.10
temperature_function	Deploy this lambda to a Greengrass core where it will send a hello world message to a topic	Python 2.7
temperature_monitor	Deploy this lambda to a Greengrass core where it will send a hello world message to a topic	Python 2.7

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

Lambda > New function

Select blueprint

Configure triggers

Configure function

Review

Node.js 6.10

Filter

Viewing 1-9 of 24

Blank Function

Configure your function from scratch.
Define the trigger and deploy your code
by stepping through our wizard.

custom

alexa-skill-kit-sdk-factskill

Demonstrate a basic fact skill built with
the ASK NodeJS SDK

nodejs6.10 · alexa

dynamodb-process-stream

An Amazon DynamoDB trigger that logs
the updates made to a table.

nodejs6.10 · dynamodb

microservice-http-endpoint

A simple backend (read/write to
DynamoDB) with a RESTful API
endpoint using Amazon API Gateway.

node-exec

Demonstrates running an external
process using the Node.js
child_process module.

alexa-skill-kit-sdk-triviaskill

Demonstrate a basic trivia skill built with
the ASK NodeJS SDK

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

Servic... Resource Groups ... aravind @ 9515-4405-4333 N. Virginia Support

Lambda > New function

Select blueprint

Configure triggers

Configure function

Review

Configure triggers

You can choose to add a trigger that will invoke your function.

Remove

Cancel Previous Next



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

Lambda > New function

Select blueprint

Configure triggers

Configure function

Review

Configure function

A Lambda function consists of the custom code you want to execute. [Learn more](#) about Lambda functions.

Name* demo

Description smart bin demo

Runtime* Node.js 6.10

Lambda function code

Provide the code for your function. Use the editor if your code does not require custom libraries (other than the aws-sdk). If you need custom libraries, you can upload your code and libraries as a .ZIP file. [Learn more](#) about deploying Lambda functions.

Code entry type [Edit code inline](#)

```
1+ exports.handler = (event, context, callback) => {
2     // TODO implement
3     callback(null, 'Hello from Lambda');
4 };
```

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

settings without the need to change function code. [Learn more](#). For storing sensitive information, we recommend encrypting values using KMS and the console's encryption helpers.

Enable encryption helpers

Environment variables

Key	Value	x
-----	-------	---

Lambda function handler and role

Handler*

index.handler



Role*

Create a custom role



The custom role creation experience will open in a new tab. Ensure that popups are enabled to create a custom role.

▶ Tags

▶ Advanced settings

* These fields are required.

Cancel

Previous

Next

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

AWS Lambda requires access to your resources

AWS Lambda uses an IAM role that grants your custom code permissions to access AWS resources it needs.

▼ Hide Details

Role Summary 

Role Description	Lambda execution role permissions
IAM Role	Create a new IAM Role 
Role Name	lambda_demo 

▶ View Policy Document

 [Don't Allow](#) [Allow](#)

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

settings without the need to change function code. [Learn more](#). For storing sensitive information, we recommend encrypting values using KMS and the console's encryption helpers.

Enable encryption helpers

Environment variables

Key	Value	x
-----	-------	---

Lambda function handler and role

Handler* ⓘ

Role* ⓘ

Existing role* ⓘ

▶ Tags

▶ Advanced settings

* These fields are required.

Cancel

Previous

Next



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

Handler index.handler

Existing role* lambda_demo

Tags

DLQ Resource

Memory (MB) 128

Timeout 3

VPC No VPC

Enable active tracing

KMS key (default) aws/lambda

[Cancel](#) [Previous](#) [Export function](#) [Create function](#) 

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

The screenshot shows the AWS IAM service interface. The top navigation bar includes 'Services', 'Resource Groups', and a bell icon. On the left, a sidebar lists navigation options: Dashboard, Groups, Users, Roles, Policies (which is selected and highlighted with a red box), Identity providers, Account settings, Credential report, and Encryption keys. The main content area is titled 'Policies' and features a 'Create policy' button with a red box around it. Below this is a search bar with 'Filter: Policy type' and a 'Search' field. A table lists various policies with columns for Policy name, Type, Attachments, and Description. Some policy names are truncated with an ellipsis.

	Policy name	Type	Attachments	Description
<input type="checkbox"/>	AdministratorAccess	Job function	1	Provides full access
<input type="checkbox"/>	AmazonAPIGatewayAdministrator	AWS managed	0	Provides full access
<input type="checkbox"/>	AmazonAPIGatewayInvokeFullAcc...	AWS managed	0	Provides full access
<input type="checkbox"/>	AmazonAPIGatewayPushToCloud...	AWS managed	0	Allows API Gateway
<input type="checkbox"/>	AmazonAppStreamFullAccess	AWS managed	0	Provides full access
<input type="checkbox"/>	AmazonAppStreamReadOnlyAccess	AWS managed	0	Provides read only a
<input type="checkbox"/>	AmazonAppStreamServiceAccess	AWS managed	0	Default policy for Am
<input type="checkbox"/>	AmazonAthenaFullAccess	AWS managed	0	Provide full access to
<input type="checkbox"/>	AmazonCloudDirectoryFullAccess	AWS managed	0	Provides full access
<input type="checkbox"/>	AmazonCloudWatchLogsFullAccess	AWS managed	0	Provides full access

STEP 10 : Select policy generator and click select

Services ▾ Resource Groups ▾ ⚙

aravind @ 9515-4405-4333 ▾ Global ▾ Support ▾

Create Policy

Step 1 : Create Policy

Step 2 : Set Permissions

Step 3 : Review Policy

Create Policy

A policy is a document that formally states one or more permissions. Create a policy by copying an AWS Managed Policy, using the Policy Generator, or typing your own custom policy.

Copy an AWS Managed Policy

Start with an AWS Managed Policy, then customize it to fit your needs.

Select

Policy Generator

Use the policy generator to select services and actions from a list. The policy generator uses your selections to create a policy.

Select

Create Your Own Policy

Use the policy editor to type or paste in your own policy.

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.

The screenshot shows the AWS DynamoDB console interface. On the left, the navigation menu is visible with 'Tables' selected. In the center, the 'bls_demo_data' table is selected in the list, highlighted with a red box. The 'Overview' tab is active, also highlighted with a red box. The 'Table details' section displays various configuration parameters for the table, including its name, primary key, time-to-live attribute, and provisioned capacity units. At the bottom of the table details, the 'Amazon Resource Name (ARN)' field is shown, which contains the value 'arn:aws:dynamodb:us-east-1:951544054333:table/bls_demo_data', also highlighted with a red box. A note at the bottom states: 'Storage size and item count are not updated in real-time. They are updated periodically, roughly every six hours.'

Table name	bls_demo_data
Primary partition key	binid (Number)
Primary sort key	time (String)
Time to live attribute	DISABLED Manage TTL
Table status	Active
Creation date	July 8, 2017 at 1:13:18 AM UTC+5:30
Provisioned read capacity units	5 (Auto Scaling Disabled)
Provisioned write capacity units	5 (Auto Scaling Disabled)
Last decrease time	-
Last increase time	-
Storage size (in bytes)	180.22 KB
Item count	2,560
Region	US East (N. Virginia)

Amazon Resource Name (ARN) **arn:aws:dynamodb:us-east-1:951544054333:table/bls_demo_data**

Storage size and item count are not updated in real-time. They are updated periodically, roughly every six hours.

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

Services ▾ Resource Groups ▾ 🔍 aravind @ 9515-4405-4333 ▾ Global ▾ Support ▾

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

Actions

Amazon Resource Name (ARN)

Add Statement

Effect	Action	Resource	Remove
Allow	dynamodb:*	arn:aws:dynamodb:us-east-1:951544054333:table/bls_demo_data	<input type="button" value="Remove"/>

Cancel Previous

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

Screenshot of the AWS IAM 'Review Policy' step. The policy name 'demo_bls' is highlighted with a red box.

Create Policy

Review Policy

Customize permissions by editing the following policy document. For more information about the access policy language, see [Overview of Policies](#) in the *Using IAM* guide. To test the effects of this policy before applying your changes, use the [IAM Policy Simulator](#).

Policy Name
demo_bls

Description

Policy Document

```
7 Action": [  
8     "dynamodb:*"  
9 ],  
10 "Resource": [  
11     "arn:aws:dynamodb:us-east-1:951544054333:table/bls_demo_data"  
12 ]  
13 }  
14 ]  
15 }
```

Use autoformatting for policy editing

[Cancel](#) [Validate Policy](#) [Previous](#) **Create Policy**

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

The screenshot shows the AWS IAM Roles page. The left sidebar has a red box around the 'Roles' link. The main area shows a role named 'lambda_demo' with details like ARN, description, and creation time. Below the role details is a tab bar with 'Permissions' (highlighted in blue), 'Trust relationships', 'Access Advisor', and 'Revoke sessions'. Under 'Permissions', there's a section for 'Managed Policies' which says 'There are no managed policies attached to this role.' A red box highlights the 'Attach Policy' button. At the bottom is a section for 'Inline Policies'.

IAM > Roles > lambda_demo

Summary

Role ARN: arn:aws:iam::951544054333:role/lambda_demo

Role description:

Instance Profile ARNs:

Path: /

Creation time: 2017-07-09 15:34 UTC+0530

Permissions Trust relationships Access Advisor Revoke sessions

Managed Policies

There are no managed policies attached to this role.

Attach Policy

Inline Policies

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

Services ▾ Resource Groups ▾ 🔍 aravind @ 9515-4405-4333 ▾ Global ▾ Support ▾

Attach Policy

Attach Policy

Select one or more policies to attach. Each role can have up to 10 policies attached.

Filter: Customer Managed ▾	Filter	Showing 22 results		
	Policy Name ▾	Attached Entities ▾	Creation Time ▾	Edited Time ▾
<input type="checkbox"/>	zonta_common_policy_without...	1	2017-03-31 20:20 UTC+0530	2017-03-31 20:20 UTC+0530
<input type="checkbox"/>	zonta_get_data_from_dbtable	1	2017-03-30 15:22 UTC+0530	2017-03-30 15:22 UTC+0530
<input type="checkbox"/>	zonta_put_data_to_dbtable	1	2017-03-30 15:19 UTC+0530	2017-03-30 15:19 UTC+0530
<input type="checkbox"/>	AWSLambdaBasicExecutionR...	0	2017-03-27 15:07 UTC+0530	2017-03-27 15:07 UTC+0530
<input checked="" type="checkbox"/>	demo_bls	0	2017-07-09 16:48 UTC+0530	2017-07-09 16:48 UTC+0530
<input type="checkbox"/>	dynamo_condition_utctime	0	2017-03-26 09:08 UTC+0530	2017-03-26 09:08 UTC+0530
<input type="checkbox"/>	dynamo_withoutcondition	0	2017-03-25 18:29 UTC+0530	2017-03-25 18:29 UTC+0530
<input type="checkbox"/>	telematics_batchGetItem_with...	0	2017-03-27 11:32 UTC+0530	2017-03-27 11:32 UTC+0530

Cancel **Attach Policy**

AWS IOT Device creation

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

The screenshot shows the AWS IoT service interface. On the left, there's a sidebar with various navigation options: Services, Resource Groups, Connect (which is highlighted with a red box), Registry, Greengrass, Security, Rules, Test, Software, Settings, and Learn.

The main content area has three cards:

- Configuring a device:** This card features an illustration of a car, a windmill, and a thermometer inside a globe. Below it is the text "Configuring a device" and "Connect a device or your computer to AWS IoT using the connection wizard for AWS IoT Device SDKs." A large blue "Get started" button is at the bottom, with a red box and a cursor icon highlighting it.
- AWS IoT Button:** This card features an illustration of a physical IoT button inside a globe. Below it is the text "AWS IoT Button" and "The AWS IoT Button is a single-purpose device that sends a message to AWS IoT with a press of a button." A blue "Configure a button" button is at the bottom, with a small note below it saying "Don't have a button? [Buy one](#)".
- AWS IoT Starter Kit:** This card features an illustration of an open box inside a globe. Below it is the text "AWS IoT Starter Kit" and "Browse AWS IoT Starter Kits that were made for connecting to AWS IoT and getting started with the service." A blue "Browse starter kits" button is at the bottom.

At the bottom of the page, there's a footer note: "Did you know you can use the AWS SDK to work with data coming from your devices? [View the AWS SDKs](#)".

STEP 2 : Select Get Started button .



1

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.



2

Download a connection kit

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



3

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

The screenshot shows the 'Register a thing' step 1/3 of the AWS IoT setup process. The top navigation bar includes 'Resource Groups' and a user profile for 'aravind @ 9515-4405-4333'. The main title 'CONNECT TO AWS IOT' and 'Register a thing' are displayed above the step indicator 'STEP 1/3'. A descriptive text explains that a thing is the representation of a physical device in the cloud. To the right, there is a link to 'Choose an existing thing instead?'. A red box highlights the 'Name' input field, which contains the value 'demo'. Below the input field is a link to 'Show optional configuration (this can be done later)'. At the bottom, there are 'Back' and 'Next step' buttons, with the 'Next step' button also highlighted by a red box.

CONNECT TO AWS IOT

Register a thing

STEP 1/3

A thing is the representation and record of your physical device in the cloud. Any physical device needs a thing to work with AWS IoT. Creating a thing will also create a thing shadow.

[Choose an existing thing instead?](#)

Name

demo

Show optional configuration (this can be done later)

Back

Next step

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

The screenshot shows the 'How are you connecting to AWS IoT?' step of the AWS IoT Device Setup wizard. The user has selected 'Linux/OSX' as the platform and 'Python' as the AWS IoT Device SDK. Both selections are highlighted with red boxes.

Resource Groups | aravind @ 9515-4405-4333 | N. Virginia

How are you connecting to AWS IoT?

Select the platform and SDK that best suits how you are connecting to AWS IoT.

Choose a platform

Linux/OSX > Windows >

Choose a AWS IoT Device SDK

Node.js > Python >



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.

Resource Groups

A policy to send and receive messages demo-Policy [Preview policy](#)

The connection kit contains:

A certificate and private key	demo.cert.pem, demo.private.key
AWS IoT Device SDK	Python SDK
A script to send and receive messages	start.sh

Before your device can connect and publish messages, you will need to download the connection kit.

Download connection kit for

Linux/OSX

[Back](#) [Next step](#)

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.

The screenshot shows a web-based interface for configuring and testing a device. At the top, there are navigation links for 'Resource Groups' and 'N. Virginia'. A user profile 'aravind @ 9515-4405-4333' is visible along with a bell icon. The main content area has a header 'Configure and test your device' and a sub-header 'STEP 3/3'. It contains three numbered steps:

- Step 1: Unzip the connection kit on the device**
unzip connect_device_package.zip
- Step 2: Add execution permissions**
chmod +x start.sh
- Step 3: Run the start script. Messages from your thing will appear below**
.start.sh

Below the steps, a message says 'Waiting for messages from your device'. At the bottom right, there are two buttons: 'Back' and 'Done', with 'Done' being highlighted by a red box.

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

Connected successfully

A device was connected to AWS IoT by performing some tasks in AWS IoT and on the device.

 Registered a thing to represent a device in AWS IoT [Learn more](#)

 Set up security for the device using a certificate and policy [Learn more](#)

 Used a device SDK to connect a device to AWS IoT [Learn more](#)

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

The screenshot shows the AWS IoT Things interface. On the left, there's a sidebar with icons for Dashboard, Connect, Registry (which is selected and has 'Things' and 'Types' listed), Greengrass, Security, Rules, Test, Software, and Settings. The main area is titled 'Things' and contains three device cards:

- bls_demo** NO TYPE
- industrial_temperature...** NO TYPE
- demo** NO TYPE

The 'demo' card is highlighted with a red border and a cursor arrow pointing at its bottom right corner. A search bar labeled 'Search things' is located at the top right of the main area.

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

The screenshot shows the AWS IoT Thing details page for a thing named "demo". The top navigation bar includes "Resource Groups", a star icon, a notification bell, the user "aravind @ 9515-4405-4333", and "N. Virg". The main content area has a dark header with "THING", "demo", "NO TYPE", and "Actions". On the left, a sidebar lists "Details", "Security", "Shadow", "Interact" (which is highlighted with a red box), and "Activity". The "Interact" section contains a "HTTPS" section with a note about connecting a device and an "MQTT" section with a note about enabling topics. A red box highlights the MQTT endpoint URL "a1irgsu21jifu4.iot.us-east-1.amazonaws.com".

THING

demo

NO TYPE

Actions

Details This thing already appears to be connected. Connect a device

Security

Shadow

Interact

Activity

HTTPS

Update your Thing Shadow using this Rest API Endpoint. [Learn more](#)

a1irgsu21jifu4.iot.us-east-1.amazonaws.com

MQTT

Use topics to enable applications and things to get, update, or delete the state information for a Thing (Thing Shadow) [Learn more](#)

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

The screenshot shows the AWS IoT MQTT client interface. On the left, a sidebar menu lists various options: Dashboard, Connect, Registry, Greengrass, Security, Rules, Test (which is highlighted with a red box), and Software, Settings, Learn. The main area is titled "MQTT client" and shows a "Subscriptions" section. It includes links to "Subscribe to a topic" and "Publish to a topic". Below these, there's a "Subscribe" section with a "Subscription topic" input field containing "demo" and a "Subscribe to topic" button. Underneath, there's a "Max message capture" input field set to 100, and a "Quality of Service" section with two radio button options: "0 - This client will not acknowledge to the Device Gateway that messages are received" (selected) and "1 - This client will acknowledge to the Device Gateway that messages are received". At the top right, it says "Connected as iotconsole-1499592639419-1".

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.

The screenshot shows the AWS IoT Subscriptions interface. On the left sidebar, under the AWS IoT section, there are several navigation items: Dashboard, Connect, Registry, Greengrass, Security, Rules, Test, Software, Settings, and Learn. The main content area has a header with 'Subscriptions' and 'demo'. Below this, there are two sections: 'Subscribe to a topic' and 'Publish to a topic'. The 'Publish to a topic' section contains a 'demo' topic and a message editor. The message editor has a red box around the topic field 'demo' and another red box around the 'Publish to topic' button. The message content is a JSON object: { "message": "Hello from AWS IoT console" }. Below the message editor, the published message is listed in a table with columns 'demo', 'Jul 9, 2017 3:07:28 PM +0530', 'Export', and 'Hide'. The published message content is also highlighted with a red box: { "message": "Hello from AWS IoT console" }.

Subscriptions	demo	Export	Clear	Pause
Subscribe to a topic				
Publish to a topic				
demo	<input type="text" value="demo"/> Publish to topic			
	<pre>1 [2 "message": "Hello from AWS IoT console" 3]</pre>			
demo	Jul 9, 2017 3:07:28 PM +0530	Export	Hide	
	{ "message": "Hello from AWS IoT console" }			

LAMBDA Trigger

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

The screenshot shows the AWS IoT Rules interface. On the left, there's a sidebar with various icons and labels: Dashboard, Connect, Registry, Greengrass, Security, Rules (which is highlighted with a red box), Test, Software, and Settings. A cursor arrow points towards the 'Create' button in the top right corner of the main content area. The main content area is titled 'Rules' and contains two rule cards: 'temperature' (ENABLED) and 'bls_training_rule' (ENABLED). Each rule card has three dots at the top right corner.

AWS IoT

Services ▾ Resource Groups ▾

Rules

Search rules

temperature
ENABLED

bls_training_rule
ENABLED

Dashboard

Connect

Registry

Greengrass

Security

Rules

Test

Software

Settings

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

The screenshot shows the 'Create a rule' page in the AWS IoT console. At the top, there's a navigation bar with 'Services', 'Resource Groups', and user information ('aravind @ 9515-4405-4333' and 'N. Virginia'). Below the header, a large teal bar says 'Create a rule'. The main area has a heading: 'Create a rule to evaluate messages sent by your things and specify what to do when a message is received (for example, write data to a DynamoDB table or invoke a Lambda function.)'. Two input fields are highlighted with red boxes: 'Name' containing 'demo_trigger' and 'Description' containing 'trigger lambda to push data to dynamo db'. A cursor arrow is positioned near the bottom right of the highlighted area. At the very bottom, there's a 'Message source' section.

Create a rule

Create a rule to evaluate messages sent by your things and specify what to do when a message is received (for example, write data to a DynamoDB table or invoke a Lambda function).

Name

demo_trigger

Description

trigger lambda to push data to dynamo db

Message source

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.

Resource Groups ▾ | ★ | aravind @ 9515-4405-4333 | N. Vir

Message source
Indicate the source of the messages you want to process with this rule.

Using SQL version ?

2016-03-23

Rule query statement

```
SELECT * FROM 'demo'
```

Attribute

*

Topic filter

demo

Condition

e.g. $temperature > 75$

Set one or more actions

The screenshot shows a user interface for creating a rule. At the top, there's a header with 'Resource Groups' and a user profile. Below it, the 'Message source' section is visible. Under 'Rule query statement', there's a code editor containing 'SELECT * FROM 'demo''. The 'Attribute' field below it contains '*' and is highlighted with a red box. The 'Topic filter' field contains 'demo' and is also highlighted with a red box, with a cursor pointing at it. The 'Condition' field has the placeholder 'e.g. temperature > 75'. At the bottom, there's a section for 'Actions' with the placeholder 'Set one or more actions'.

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

The screenshot shows the AWS IoT Rule creation interface. At the top, there are navigation links for 'Resource Groups' and a user profile. Below that, a 'Topic filter' field contains 'demo'. A 'Condition' section is present with the placeholder 'e.g. temperature > 75'. The main area is titled 'Set one or more actions' and contains a descriptive text about selecting actions for inbound messages. A prominent 'Add action' button is located in a red-bordered box at the bottom left. At the bottom right, there are 'Cancel' and 'Create rule' buttons.

Resource Groups | aravind @ 9515-4405-4333 | N. Vi

Topic filter

demo

Condition

e.g. temperature > 75

Set one or more actions

Select one or more actions to happen when the above rule is matched by an inbound message. Actions define additional activities that occur when messages arrive, like storing them in a database, invoking cloud functions, or sending notifications. (*.required)

Add action

Cancel Create rule

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

Resource Groups ▾ | 🔍 | 🔔 | aravind @ 9515-4405-4333 ▾ | N. Virginia ▾

Select an action

Select an action.

-  Insert a message into a DynamoDB table
DYNAMODB
-  Split message into multiple columns of a database table (DynamoDBv2)
DYNAMODBV2
-  Invoke a Lambda function passing the message data
LAMBDA
-  Send a message as an SNS push notification
SNS
-  Send a message to an SOS queue
SOS



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.

The screenshot shows the AWS Lambda trigger configuration interface. At the top, there's a header with 'Resource Groups' and a star icon, followed by a user profile 'aravind @ 9515-4405-4333' and a notification bell icon. Below the header, the trigger name 'demo' is visible. The main area is divided into two sections: 'Condition' and 'Set one or more actions'. The 'Condition' section contains a placeholder 'e.g. temperature > 75'. The 'Set one or more actions' section contains a single action: 'Invoke a Lambda function passing the message data' (with the ARN 'demo'). This action has a 'Remove' button, an 'Edit' button, and a 'More' link. Below this action is a blue 'Add action' button. At the bottom right, there are 'Cancel' and 'Create rule' buttons, with the 'Create rule' button being highlighted with a red border and a cursor pointing at it.

Resource Groups | ★

aravind @ 9515-4405-4333 | N. V.

demo

Condition

e.g. temperature > 75

Set one or more actions

Select one or more actions to happen when the above rule is matched by an inbound message. Actions define additional activities that occur when messages arrive, like storing them in a database, invoking cloud functions, or sending notifications. (*.required)

 **Invoke a Lambda function passing the message data**
demo Remove Edit ▾

Add action

Cancel Create rule

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

The screenshot shows a configuration interface for invoking a Lambda function. At the top, there's a navigation bar with 'Resource Groups' and a user profile. Below it, a blue header bar has a star icon and the text 'Invoke a Lambda function passing the message data LAMBDA'. The main content area contains a message: 'We'll set the permissions on the Lambda function for you.' Below this, there's a form field labeled '*Function name' containing 'demo', with a red box highlighting the input field. To the right of the input field is a 'Create a new resource' button. In the bottom right corner of the main area, there's a large teal 'Add action' button with a red box highlighting it, and a cursor arrow pointing towards it.

Resource Groups ▾ | ★

aravind @ 9515-4405-4333 ▾ | N. V.

Invoke a Lambda function passing the message data
LAMBDA

We'll set the permissions on the Lambda function for you.

*Function name

demo

Create a new resource

Add action

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.

Screenshot of the AWS Lambda console showing the Triggers tab for a function named "demo".

The ARN of the function is listed as ARN - arn:aws:lambda:us-east-1:951544054333:function:demo.

A success message states: "Congratulations! Your Lambda function "demo" has been successfully created. You can now click on the "Test" button to input a test event and test your function."

The Triggers tab is selected, displaying the following information:

- AWS IoT: demo_trigger**
- arn:aws:iot:us-east-1:951544054333:rule/demo_trigger
- Rule description: trigger lambda to push data to dynamo db
- SQL statement: SELECT * FROM 'demo'

Buttons available on the page include:

- Add trigger
- Refresh triggers
- View function policy
- Disable
- Delete

COGNITO Identity pool

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

The screenshot shows the Amazon Cognito console homepage. At the top, there is a navigation bar with links for Services, Resource Groups, and Support. Below the navigation bar is the Cognito logo and the word "Amazon Cognito". A descriptive text block explains the service's purpose: "Amazon Cognito makes it easy for you to have users sign up and sign in to your apps, federate identities from social identity providers, secure access to AWS resources and synchronize data across multiple devices, platforms, and applications." Two blue buttons are present at the bottom: "Manage your User Pools" and "Manage Federated Identities". The "Manage Federated Identities" button is highlighted with a red rectangular border and has a cursor arrow pointing towards it.

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

Screenshot of the AWS Cognito Federated Identities console. The top navigation bar includes 'Services' (dropdown), 'Resource Groups' (dropdown), a search bar, a bell icon for notifications, the user 'aravind @ 9515-4405-4333', location 'N. Virginia', and 'Support' (dropdown). Below the navigation is a secondary navigation bar with 'Federated Identities' (selected) and 'User Pools'. A red box highlights the 'Create new identity pool' button, which is being clicked by a cursor. The main content area displays two identity pools: 'bls_cognito' (10 identities) and 'bls_training_cogni' (2 identities), each with a line graph showing identity creation over time.

bls_cognito

Identities 10 | Change N/A

The graph shows a flat line at zero identities for most of the timeline, followed by a sharp vertical jump to 10 identities at approximately the 80% mark of the timeline.

bls_training_cogni

Identities 2 | Change N/A

The graph shows a flat line at zero identities for most of the timeline, followed by a sharp vertical jump to 2 identities at approximately the 80% mark of the timeline.

Getting started

- [Developer Guide](#)
- [Getting started with Android](#)
- [Getting started with iOS](#)
- [Identity API reference](#)
- [Sync API Reference](#)

In-Depth Guides

- [Using Cognito to Sync Data](#)
- [Using Cognito in Your Website](#)
- [Using the Cognito Credentials Provider](#)

Community

- [Cognito Developer Forum](#)
- [AWS Mobile Blog](#)
- [GitHub Repository](#)

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

The screenshot shows the 'Getting started wizard' for 'Step 1: Create identity pool'. The 'Identity pool name*' field contains 'demo' with a green checkmark icon. Below it, an example 'My App Name' is shown. A checkbox for 'Enable access to unauthenticated identities' is checked and highlighted with a red box. At the bottom right, there are 'Cancel' and 'Create Pool' buttons, with 'Create Pool' also highlighted with a red box.

Services | Resource Groups | aravind @ 9515-4405-4333 | Select a Region | Support

Getting started wizard

Step 1: Create identity pool

Step 2: Set permissions

Create new identity pool

Identity pools are used to store end user identities. To declare a new identity pool, enter a unique name.

Identity pool name* demo ✓

Example: My App Name

▼ Unauthenticated identities ⓘ

Amazon Cognito can support unauthenticated identities by providing a unique identifier and AWS credentials for users who do not authenticate with an identity provider. If your application allows customers to use the application without logging in, you can enable access for unauthenticated identities. [Learn more about unauthenticated identities.](#)

Enable access to unauthenticated identities

▶ Authentication providers ⓘ

* Required

Cancel Create 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.

Services ▾ | Resource Groups ▾ | ⚙ aravind @ 9515-4405-4333 ▾ | Global ▾ | Support ▾

By default, Amazon Cognito creates a new role with limited permissions - end users only have access to Cognito Sync and Mobile Analytics. You can modify the roles if your application needs access to other AWS resources, such as S3 or DynamoDB.

▼ Hide Details

Role Summary ?

Role Description Your authenticated identities would like access to Cognito.

IAM Role Create a new IAM Role

Role Name Cognito_demoAuth_Role

▶ View Policy Document

→

Role Summary ?

Role Description Your unauthenticated identities would like access to Cognito.

IAM Role Create a new IAM Role

Role Name Cognito_demoUnauth_Role

Don't Allow Allow

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.

Screenshot of the AWS IAM Role Summary page showing the policy editor interface.

The Role Summary section includes:

- Role Description:** Your unauthenticated identities would like access to Cognito.
- IAM Role:** Create a new IAM Role (button highlighted with a red box).
- Role Name:** Cognito_demoUnauth_Role (text input field highlighted with a red box).

A link to "Hide Policy Document" is present.

The Policy Document editor displays the following JSON policy template:

```
{"Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "mobileanalytics:PutEvents", "cognito-sync:*" ], "Resource": [ "*" ] } ]}
```

An "Edit" button is located above the policy document, also highlighted with a red box.

At the bottom right, there are two buttons: "Don't Allow" and "Allow", with "Allow" highlighted with a red box.

```
{  
    "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.

The screenshot shows the 'Getting started with Amazon Cognito' page. On the left, there's a sidebar with links: 'Identity pool', 'Dashboard', 'Sample code' (which is highlighted with a red box), and 'Identity browser'. The main content area has a heading 'Getting started with Amazon Cognito'. Below it, there's a 'Platform' dropdown menu with a red box around it, currently set to 'Android'. A dropdown menu is open, showing options: 'Android', 'iOS - Objective C', 'iOS - Swift', 'JavaScript' (which is highlighted with a red box and has a cursor over it), 'Unity', 'Xamarin', and '.Net'. To the right of the dropdown, there are download buttons for 'Android' and 'Developer Guide'. Below the dropdown, there's some sample code for initializing a Cognito credentials provider in JavaScript:

```
// Initialize the Cognito credentials provider
CognitoCachingCredentialsProvider credentialsProvider = new CognitoCachingCredentialsProvider(
    "us-east-1:8311dd33-699c-4e36-adbe-2aa890c7f6d2", // Identity pool ID
    Regions.US_EAST_1 // Region
);
```

At the bottom, there's a section titled '▼ Store User Data'.

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

The screenshot shows the AWS Cognito 'Getting started with Amazon Cognito' page. On the left, there's a sidebar with links: Identity pool, Dashboard, Sample code (which is highlighted with a red box), and Identity browser. The main content area has a title 'Getting started with Amazon Cognito' and a 'Platform' dropdown set to 'JavaScript'. Below this, there are sections for 'Download the AWS SDK' (with a 'Download the AWS SDK for JavaScript' button) and 'Get AWS Credentials'. A code snippet for initializing AWS credentials is shown in a box, with the entire code block also highlighted by a red box. At the bottom, there's a section for 'Store User Data' and a note about the Amazon Cognito Sync Manager library.

Identity pool

Dashboard

Sample code

Identity browser

Getting started with Amazon Cognito

Platform **JavaScript**

▼ Download the AWS SDK

[Download the AWS SDK for JavaScript](#) Developer Guide

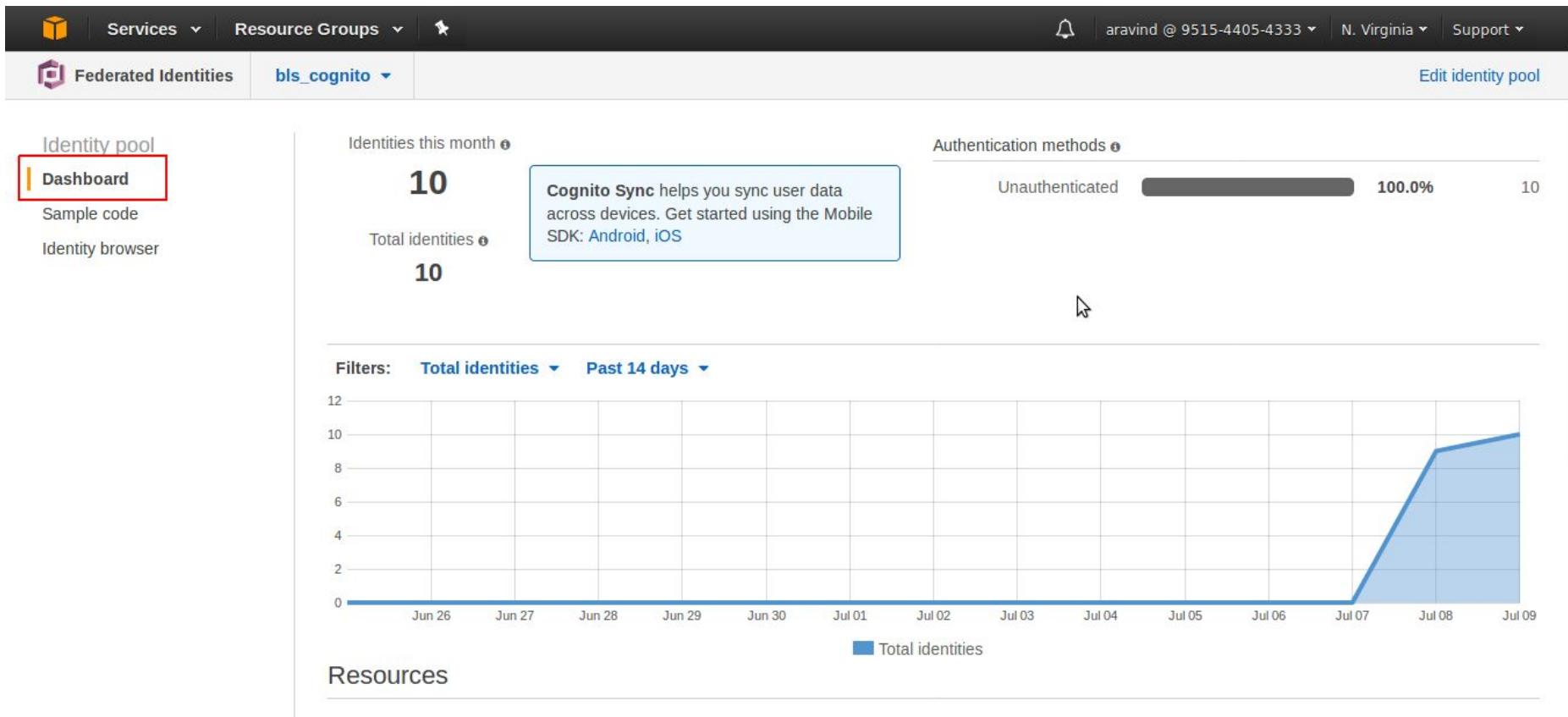
▼ Get AWS Credentials

```
// Initialize the Amazon Cognito credentials provider
AWS.config.region = 'us-east-1'; // Region
AWS.config.credentials = new AWS.CognitoIdentityCredentials({
  IdentityPoolId: 'us-east-1:8311dd33-699c-4e36-adbe-2aa890c7f6d2',
});
```

▼ Store User Data

Download the [Amazon Cognito Sync Manager](#) library from GitHub and include it in your project.

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**.

The screenshot shows the AWS IAM service interface. The top navigation bar includes 'Services' and 'Resource Groups' dropdowns, and a bell icon for notifications. Below the navigation is a search bar labeled 'Search IAM'. On the left, a sidebar menu lists 'Dashboard', 'Groups', 'Users', 'Roles' (which is selected and highlighted in orange), 'Policies', 'Identity providers', 'Account settings', 'Credential report', and 'Encryption keys'. The main content area is titled 'Create new role' and 'Role actions'. It features a 'Filter' input field and a table with columns for 'Role name' and 'Description'. The table lists several roles:

Role name	Description
AWSServiceRoleForLexBots	
bls_training	Allows API Gateway to call AWS resources on your behalf.
client	
developer	
greengrassTest	
Greengrass_ServiceRole	
lambda_basic_execution	
lambda_func_role	
role_name	

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

Screenshot of the AWS IAM 'Create role' wizard - Step 2: Select role type.

The left sidebar shows navigation steps: Create role, Step 1: Select role type (highlighted), Step 2: Establish trust, Step 3: Attach policy, and Step 4: Set role name and review.

The main content area is titled 'Select role type'. It lists several service roles under the heading 'AWS Service Role':

- AWS Batch Service Role**: Allows AWS Batch to create and manage AWS resources on your behalf. **Select** button.
- AWS CodeBuild**: Role to allow CodeBuild to access other AWS services such as S3 on your behalf. **Select** button.
- Amazon API Gateway**: Allows API Gateway to call AWS resources on your behalf. **Select** button (with a cursor icon).
- AWS Cloudformation Role**: Allows Cloudformation to create and manage AWS stacks and its resources on your behalf. **Select** button.
- AWS Config**: Allows AWS Config to call AWS services and collect resource configurations on your behalf. **Select** button.

Below this section are three additional role types:

- AWS service-linked role** (radio button)
- Role for cross-account access** (radio button)
- Role for identity provider accounts** (radio button)

At the bottom right are 'Cancel' and 'Next Step' buttons.

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. x | API Gateway x | API Gateway x | DynamoDB · AWS Cons x | IAM Management Cons x | Amazon API Gateway a x | +

https://console.aws.amazon.com/iam/home?region=us-east-1#/roles

Services | Resource Groups | [Create role](#)

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

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

The screenshot shows the AWS DynamoDB console interface. On the left, the navigation menu includes 'DynamoDB', 'Dashboard', 'Tables' (which is selected and highlighted with a red box), 'Reserved capacity', 'DAX', 'Dashboard', 'Clusters', 'Subnet groups', 'Parameter groups', and 'Events'. In the center, the 'bls_demo_data' table is selected in the 'Tables' list (also highlighted with a red box). The 'Overview' tab is active (highlighted with a red box). The 'Table details' section displays the following information:

Table name	bls_demo_data
Primary partition key	binid (Number)
Primary sort key	time (String)
Time to live attribute	DISABLED Manage TTL
Table status	Active
Creation date	July 8, 2017 at 1:13:18 AM UTC+5:30
Provisioned read capacity units	5 (Auto Scaling Disabled)
Provisioned write capacity units	5 (Auto Scaling Disabled)
Last decrease time	-
Last increase time	-
Storage size (in bytes)	180.22 KB
Item count	2,560
Region	US East (N. Virginia)
Amazon Resource Name (ARN)	arn:aws:dynamodb:us-east-1:951544054333:table/bls_demo_data

A note at the bottom states: "Storage size and item count are not updated in real-time. They are updated periodically, roughly every six hours."

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

Amazon Resource Name (ARN)	Actions
	3 Action(s) Selected
	<input type="checkbox"/> PurchaseReservedCapacityOfferings
	<input checked="" type="checkbox"/> PutItem
	<input checked="" type="checkbox"/> Query
	<input checked="" type="checkbox"/> Scan
	<input type="checkbox"/> UpdateItem
	<input type="checkbox"/> UpdateTable

Cancel

Previous

Next Step

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

Create Policy

Step 1 : Create Policy

Step 2 : Set Permissions

Step 3 : Review Policy

Review Policy

Customize permissions by editing the following policy document. For more information about the access policy language, see [Overview of Policies](#) in the *Using IAM* guide. To test the effects of this policy before applying your changes, use the [IAM Policy Simulator](#).

Policy Name

lambda_telematics_withoutcondition

Description

Policy Document

```
1  {
2      "Version": "2012-10-17",
3      "Statement": [
4          {
5              "Sid": "Stmt1490965447000",
6              "Effect": "Allow",
7              "Action": [
8                  "dynamodb:PutItem",
9                  "dynamodb:Query",
10                 "dynamodb:Scan"
11             ],
12             "Resource": "arn:aws:dynamodb:us-east-1:123456789012:table/MyTable"
13         }
14     ]
15 }
```

Use autoformatting for policy editing

[Cancel](#) [Validate Policy](#) [Previous](#) [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.

The screenshot shows the AWS IAM Roles page. On the left, there's a sidebar with navigation links: Dashboard, Groups, Users, Roles (which is selected and highlighted in orange), Policies, Identity providers, Account settings, Credential report, and Encryption keys. The main content area has a header with 'Create New Role' and 'Role Actions'. Below that is a search bar labeled 'Search IAM'. The main table displays six results, with the last one, 'lambda_func_role', being the selected item, indicated by a red border around its row. The columns in the table are 'Role Name' and 'Creation Time'.

Filter		Showing 6 results
<input type="checkbox"/>	Role Name	Creation Time
<input type="checkbox"/>	client	2017-03-24 23:41 UTC+0530
<input type="checkbox"/>	developer	2017-03-22 16:02 UTC+0530
<input type="checkbox"/>	lambda_basic_execution	2017-03-28 11:25 UTC+0530
<input checked="" type="checkbox"/>	lambda_func_role	2017-03-31 18:30 UTC+0530
<input type="checkbox"/>	role_name	2017-03-27 18:28 UTC+0530
<input type="checkbox"/>	zonta_client_demo	2017-03-30 15:01 UTC+0530

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

Attach Policy

Attach Policy

Select one or more policies to attach. Each role can have up to 10 policies attached.

Filter: Customer Managed		Filter	Showing 14 results		
	Policy Name	Attached Entities	Creation Time	Edited Time	
<input type="checkbox"/>	zonta_get_data_from_dbtable	1	2017-03-30 15:22 UTC+0530	2017-03-30 15:22 UTC+0...	
<input type="checkbox"/>	zonta_put_data_to_dbtable	1	2017-03-30 15:19 UTC+0530	2017-03-30 15:19 UTC+0...	
<input type="checkbox"/>	AWSLambdaBasicExecution...	0	2017-03-27 15:07 UTC+0530	2017-03-27 15:07 UTC+0...	
<input type="checkbox"/>	dynamo_condition_utctime	0	2017-03-26 09:08 UTC+0530	2017-03-26 09:08 UTC+0...	
<input type="checkbox"/>	dynamo_withoutcondition	0	2017-03-25 18:29 UTC+0530	2017-03-25 18:29 UTC+0...	
<input checked="" type="checkbox"/>	lambda_telematics_withoutco...	0	2017-03-31 18:35 UTC+0530	2017-03-31 18:35 UTC+0...	
<input type="checkbox"/>	telambda_telematics_withoutcondition		2017-03-27 11:32 UTC+0530	2017-03-27 11:32 UTC+0...	
<input type="checkbox"/>	tst_condition	0	2017-03-24 17:06 UTC+0530	2017-03-24 17:06 UTC+0...	
<input type="checkbox"/>	tst_condition2	0	2017-03-24 23:40 UTC+0530	2017-03-24 23:40 UTC+0...	

Cancel

Attach Policy

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

The screenshot shows the AWS IAM Roles page. The left sidebar has a 'Roles' section selected, highlighted with an orange border. The main content area shows the 'Summary' tab for the role 'lambda_func_role'. The 'Role ARN' is listed as 'arn:aws:iam:█████████████████████:role/lambda_func_role'. The 'Creation Time' is '2017-03-31 18:30 UTC+0530'. Below this, there are tabs for 'Permissions', 'Trust Relationships', 'Access Advisor', and 'Revoke Sessions', with 'Permissions' being the active tab. Under the 'Managed Policies' section, it says 'The following managed policies are attached to this role. You can attach up to 10 managed policies.' A blue 'Attach Policy' button is visible. A table lists one policy: 'Policy Name: lambda_telematics_withoutcondition' with 'Actions' 'Show Policy | Detach Policy | Simulate Policy'. A red box highlights the 'lambda_telematics...' row.

IAM > Roles > lambda_func_role

Summary

Role ARN arn:aws:iam:█████████████████████:role/lambda_func_role

Instance Profile ARN(s)

Path /

Creation Time 2017-03-31 18:30 UTC+0530

Permissions Trust Relationships Access Advisor Revoke Sessions

Managed Policies

The following managed policies are attached to this role. You can attach up to 10 managed policies.

Attach Policy

Policy Name	Actions
lambda_telematics_withoutcondition	Show Policy Detach Policy Simulate Policy

**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.

The screenshot shows the Amazon API Gateway console interface. At the top, there's a navigation bar with the 'Amazon API Gateway' logo, 'APIs' selected, and 'Show all hints' and a help icon. On the left, a sidebar lists 'APIs' (CommentsApi, conditionapi, dynamoapi, dynamoputgetapi), 'Usage Plans', 'API Keys', 'Custom Domain Names', 'Client Certificates', and 'Settings'. The main area displays a list of existing APIs: 'CommentsApi' (Testing the api), 'conditionapi' (No description.), 'dynamoapi' (No description.), 'dynamoputgetapi' (No description.), 'Telematics' (Telematic API), and 'telematics_scan_filterapi' (No description.). A red box highlights the 'Create API' button at the top center of the page.

STEP 3 : Enter the API name (your choice)

Screenshot of the Amazon API Gateway 'Create new API' page.

The top navigation bar shows 'Services', 'Resource Groups', a user icon for 'aravind @ 9515-4405-4333', 'N. Virginia', and 'Support'.

The breadcrumb path indicates the current location is 'APIs > Create'.

A 'Show all hints' link and a help icon are also present in the top right.

Create new API

In Amazon API Gateway, an API refers to a collection of resources and methods that can be invoked through HTTPS endpoints.

Three radio button options are available:

- New API
- Import from Swagger
- Example API

Settings

Choose a friendly name and description for your API.

API name*

Description

* Required

Create API button

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

The screenshot shows the Amazon API Gateway console interface. The top navigation bar includes the logo, 'Amazon API Gateway', 'APIs > getlatest (ki8sgz2ane) > Resources > / (q8etlwj5b5)', 'Show all hints', and a help icon.

The left sidebar lists APIs: 'CommentsApi', 'conditionapi', 'dynamoapi', 'dynamoputgetapi', 'getlatest', and several items under 'Resources' (highlighted in orange), including 'Stages', 'Authorizers', 'Models', 'Documentation', and 'Binary Support'.

The main content area has tabs: 'APIs', 'Resources' (selected), and 'Actions'. The 'Actions' tab is expanded, showing two sections: 'RESOURCE ACTIONS' and 'API ACTIONS'. Under 'RESOURCE ACTIONS', 'Create Resource' is highlighted with a red box. Under 'API ACTIONS', 'Delete API' is highlighted with a red box. The text 'No methods defined for the resource.' is displayed.

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

The screenshot shows the AWS API Gateway interface for creating a new child resource. The top navigation bar includes 'Resource Groups', a user icon, 'aravind @ 9515-4405-4333', 'N. Virginia', and 'Support'. Below the navigation is a breadcrumb trail: APIs > smart_bin_latest_data (e4vxf5szne) > Resources > / (5qcx3v2qyl) > Create. On the right, there are 'Show all hints' and a help icon. The main section is titled 'New Child Resource' with a sub-section 'Actions'. A note says 'Use this page to create a new child resource for your resource.' A checkbox 'Configure as proxy resource' is checked. The 'Resource Name*' field contains 'getlatest'. The 'Resource Path*' field contains '/ getlatest'. A note explains path parameters: 'You can add path parameters using brackets. For example, the resource path {username} represents a path parameter called 'username'. Configuring /{proxy+} as a proxy resource catches all requests to its sub-resources. For example, it works for a GET request to /foo. To handle requests to /, add a new ANY method on the / resource.' Below this is a 'Enable API Gateway CORS' checkbox, which is checked. At the bottom, a note says '* Required' and there are 'Cancel' and 'Create Resource' buttons, with the 'Create Resource' button being clicked.

Resource Groups ▾ | ★

aravind @ 9515-4405-4333 ▾ N. Virginia ▾ Support ▾

APIs > smart_bin_latest_data (e4vxf5szne) > Resources > / (5qcx3v2qyl) > Create Show all hints ?

Resources Actions ▾ New Child Resource

Use this page to create a new child resource for your resource.

Configure as proxy resource ⓘ

Resource Name*

Resource Path*

You can add path parameters using brackets. For example, the resource path {username} represents a path parameter called 'username'. Configuring /{proxy+} as a proxy resource catches all requests to its sub-resources. For example, it works for a GET request to /foo. To handle requests to /, add a new ANY method on the / resource.

Enable API Gateway CORS ⓘ

* Required

Create Resource

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

The screenshot shows the Amazon API Gateway console interface. The top navigation bar displays the path: APIs > getlatest (ki8sgz2ane) > Resources > /datestring (apz9xx). On the left sidebar, under the 'APIs' section, several API names are listed: CommentsApi, conditionapi, dynamoapi, dynamoputgetapi, getlatest, Resources (which is selected and highlighted in orange), Stages, Authorizers, Models, and Documentation. The main content area is titled '/datestring Methods'. A context menu is open over the resource path '/datestring', with the 'Create Resource' option highlighted by a red box. The menu also includes options like 'Create Method', 'Enable CORS', 'Edit Resource Documentation', and 'Delete Resource'. Below the menu, a message states 'No methods defined for the resource.'

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).

The screenshot shows the 'Amazon API Gateway' interface. The top navigation bar includes the logo, 'APIs > smart_bin_latest_data (e4vxf5szne) > Resources > /getlatest (xbrsal) > Create', 'Show all hints', and a help icon.

The left sidebar lists 'APIs', 'smart_bin_latest_data', 'Resources' (which is selected), 'Stages', 'Authorizers', 'Gateway Responses', 'Models', 'Documentation', and 'Binary Support'. Below these are 'Usage Plans', 'API Keys', 'Custom Domain Names', and 'Client Certificates'.

The main content area is titled 'New Child Resource' with the sub-instruction 'Use this page to create a new child resource for your resource.' A blue circle highlights the 'Actions' dropdown menu.

The 'Configure as proxy resource' checkbox is unchecked. The 'Resource Name*' field contains 'binid'. The 'Resource Path*' field contains '/getlatest/{binid}'. A note explains: 'You can add path parameters using brackets. For example, the resource path {username} represents a path parameter called 'username'. Configuring /getlatest/{proxy+} as a proxy resource catches all requests to its sub-resources. For example, it works for a GET request to /getlatest/foo. To handle requests to /getlatest, add a new ANY method on the /getlatest resource.'

The 'Enable API Gateway CORS' checkbox is checked. At the bottom, there are 'Cancel' and 'Create Resource' buttons, with the latter being highlighted by a blue circle.

STEP 6 : Now create Method

The screenshot shows the Amazon API Gateway console interface. The top navigation bar includes the logo, 'Amazon API Gateway', the path 'APIs > getlatest (ki8sgz2ane) > Resources > /datestring/{datestring} (55mzwI)', 'Show all hints', and a help icon.

The left sidebar lists various APIs: 'CommentsApi', 'conditionapi', 'dynamoapi', 'dynamoputgetapi', 'getlatest', and several items under 'Resources' (highlighted in orange), including 'Stages', 'Authorizers', 'Models', 'Documentation', and 'Binary Support'.

The main content area displays the path '/datestring/{datestring}' and the title '/datestring/{datestring} Methods'. A modal menu is open at the 'Actions' dropdown, showing the following options:

- RESOURCE ACTIONS**
 - Create Method (highlighted with a red border)
 - Create Resource
 - Enable CORS
 - Edit Resource Documentation
 - Delete Resource (highlighted in red)
- API ACTIONS**
 - Deploy API
 - Import API
 - Edit API Documentation
 - Delete API (highlighted in red)

To the right of the modal, the message 'No methods defined for the resource.' is displayed.

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.

The screenshot shows the AWS API Gateway console interface. At the top, the navigation path is: APIs > smart_bin_latest_data (e4vxf5szne) > Resources > /getlatest/{binid} (u9i6ux). On the right, there is a "Show" button. Below the path, the main area has "Resources" and "Actions" tabs. The "Actions" tab is selected, and a dropdown menu is open, with the "GET" option highlighted and marked with a blue circle and a tick symbol. The URL path "/getlatest/{binid} Methods" is displayed. To the left, a sidebar shows the resource hierarchy: / > /getlatest > /{binid}. Under /{binid}, there are "GET" and "OPTIONS" methods, with "GET" being the active one. The "GET" method card for DynamoDB shows "Authorization: None" and "API Key: Not required". To the right, the "OPTIONS" method card for Mock Endpoint also shows "Authorization: None" and "API Key: Not required". A cursor icon is visible at the bottom left.

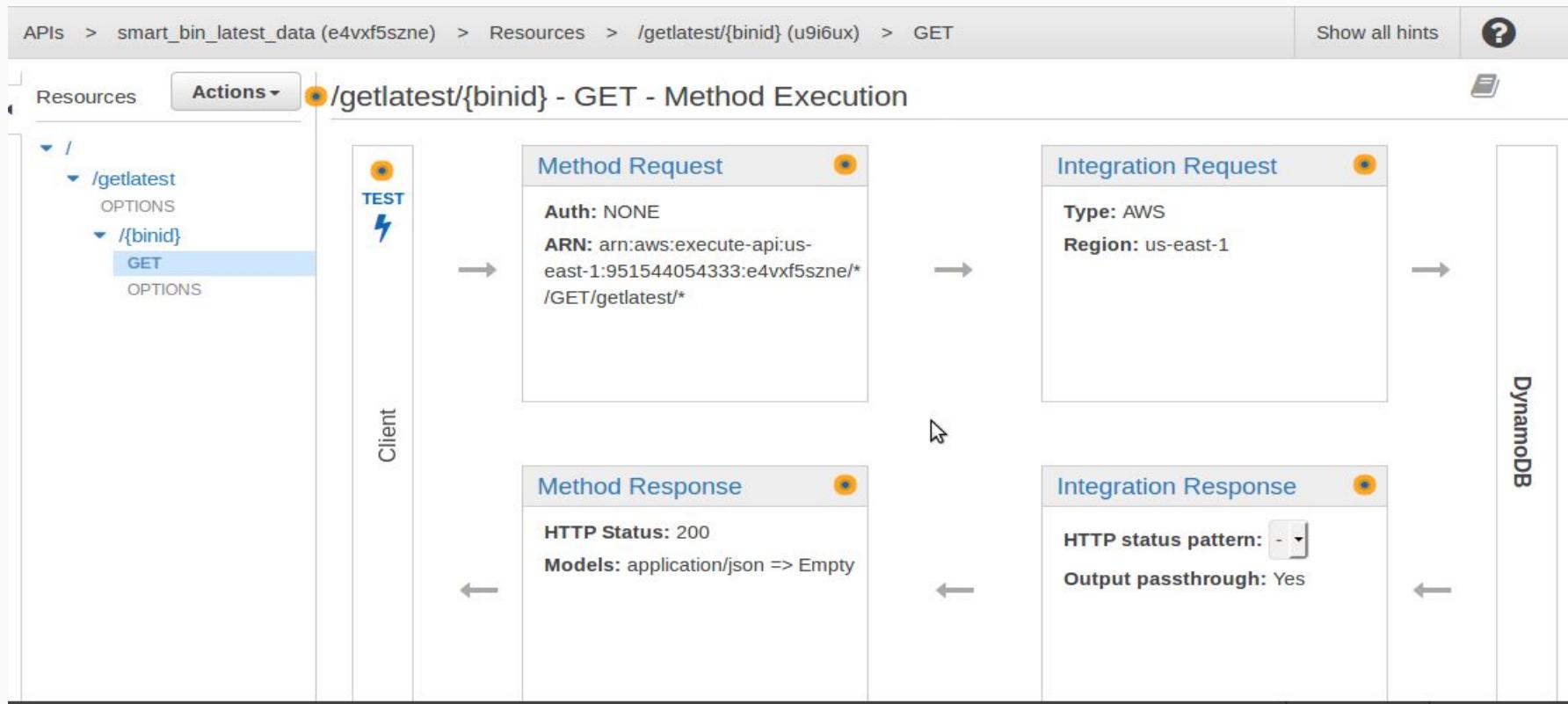
STEP 7 : Select any **integration type** for this api gateway, we are going to use internal dynamodb, it comes under the AWS service, so select the **AWS Service** and Enter the

- **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.

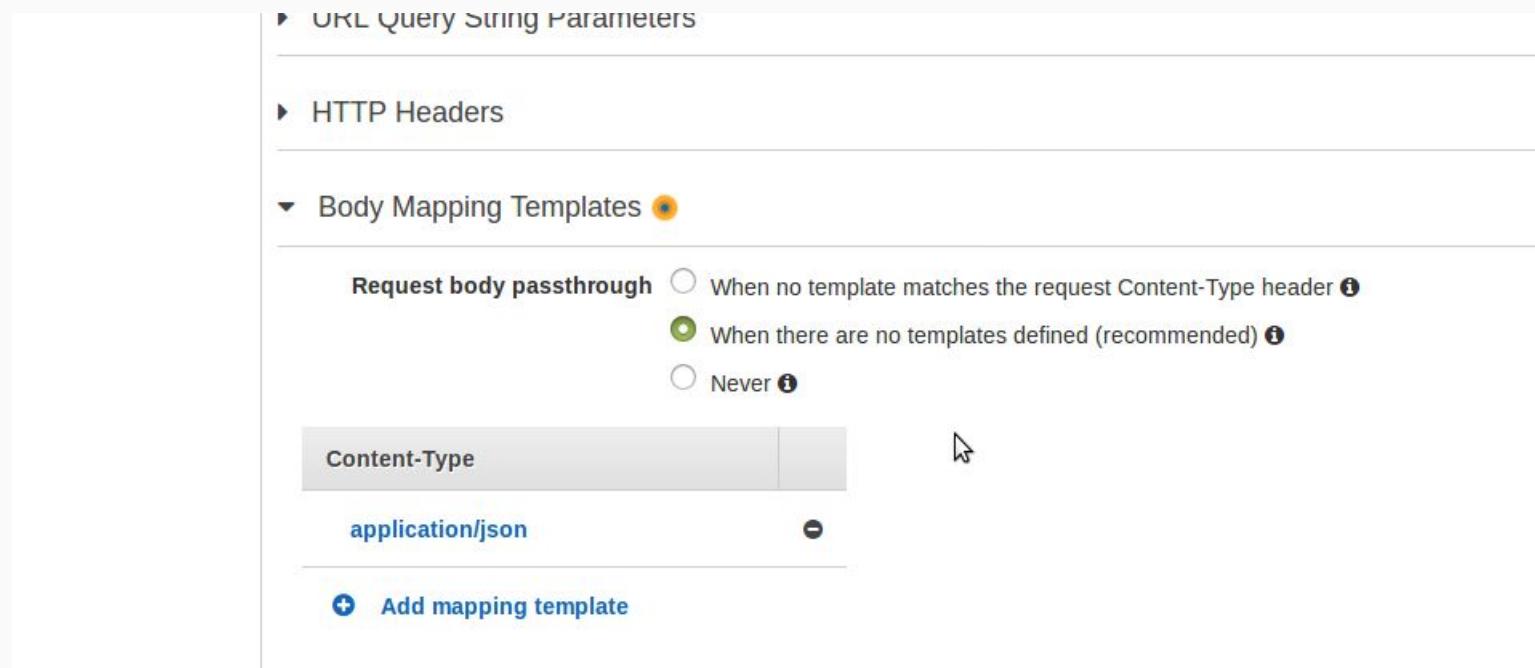
<p>Stages</p> <p>Authorizers</p> <p>Gateway Responses</p> <p>Models</p> <p>Documentation</p> <p>Binary Support</p> <p>Usage Plans</p> <p>API Keys</p> <p>Custom Domain Names</p> <p>Client Certificates</p> <p>Settings</p>	<p>OPTIONS</p> <p>▼ /{binid}</p> <p>GET</p> <p>OPTIONS</p>	<p>Integration type <input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

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.

When there are no templates defined (recommended) i

Never i

Content-Type

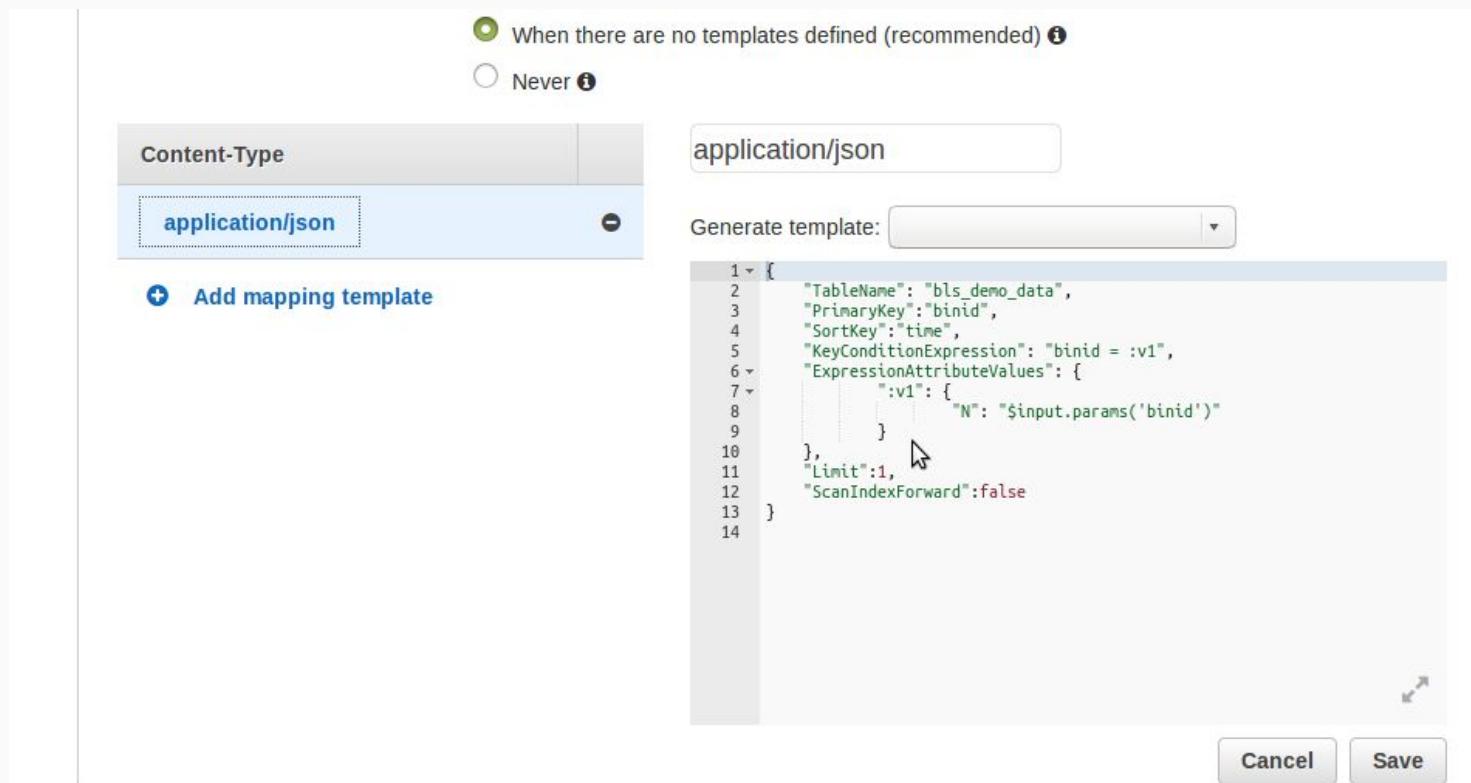
application/json

Add mapping template

Generate template:

```
1 {  
2   "TableName": "bls_demo_data",  
3   "PrimaryKey": "binid",  
4   "SortKey": "time",  
5   "KeyConditionExpression": "binid = :v1",  
6   "ExpressionAttributeValues": {  
7     ":v1": {  
8       "N": "$input.params('binid')"  
9     }  
10    },  
11    "Limit": 1,  
12    "ScanIndexForward": false  
13  }  
14
```

Cancel Save



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.

The screenshot shows the AWS API Gateway Method Execution interface. On the left, the resource tree shows a path / with a /getlatest resource. Under /getlatest, there are three methods: OPTIONS, /{binid}, and GET. The GET method is selected and highlighted with a blue background. The main panel displays the results of a test call to the /getlatest/{binid} endpoint using the GET method. The path parameter {binid} is set to 1. The response shows a status of 200, latency of 83 ms, and a JSON response body:

```
{ "Count": 1, "Items": [ { "binid": { "N": "1" }, "binlevel": { "N": "32" }, "time": { "S": "2017-07-14 11:05:07" }, "binlocation": { "S": "1" } } ] }
```

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.