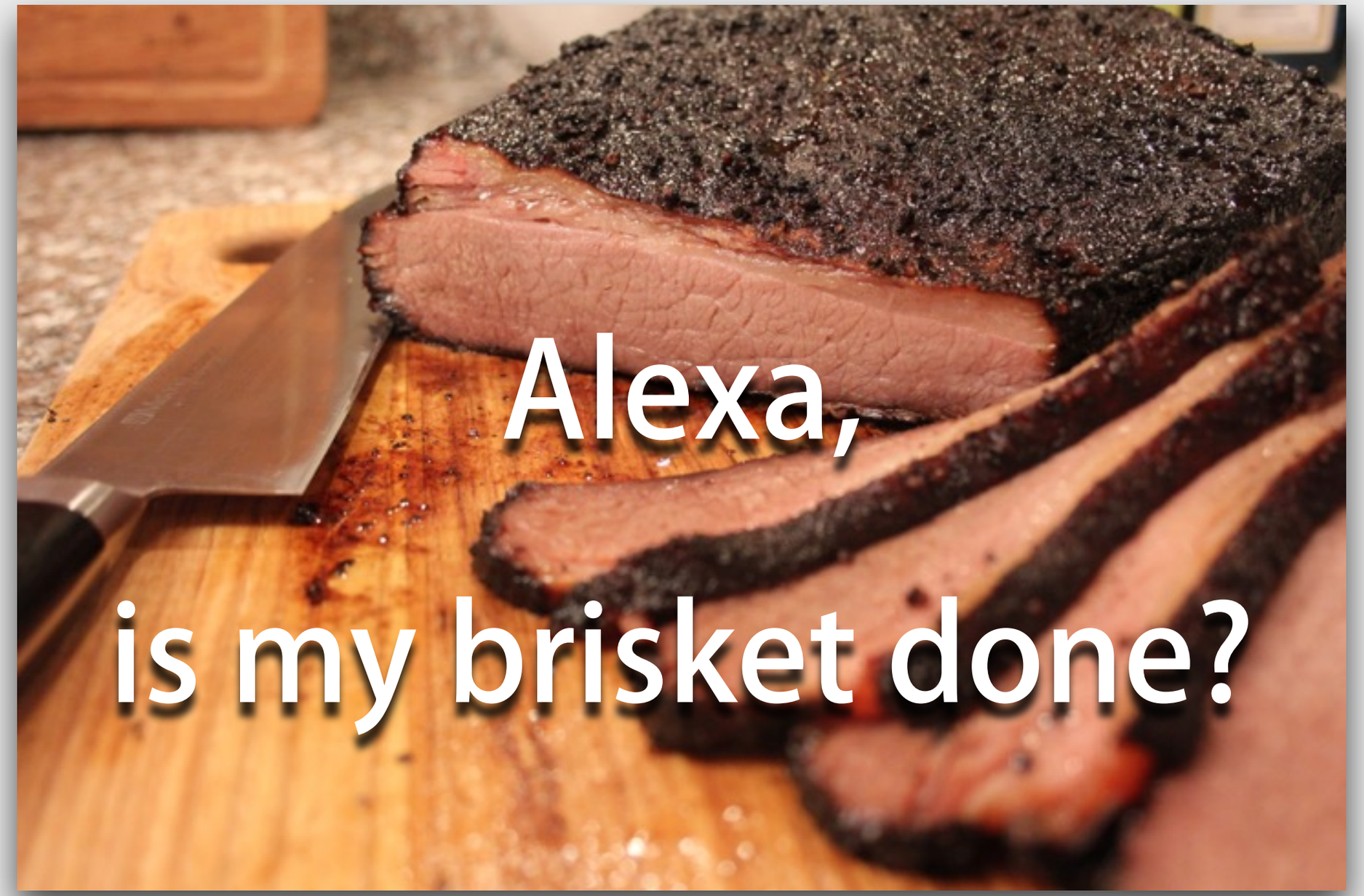
Amazon Echo & IoT

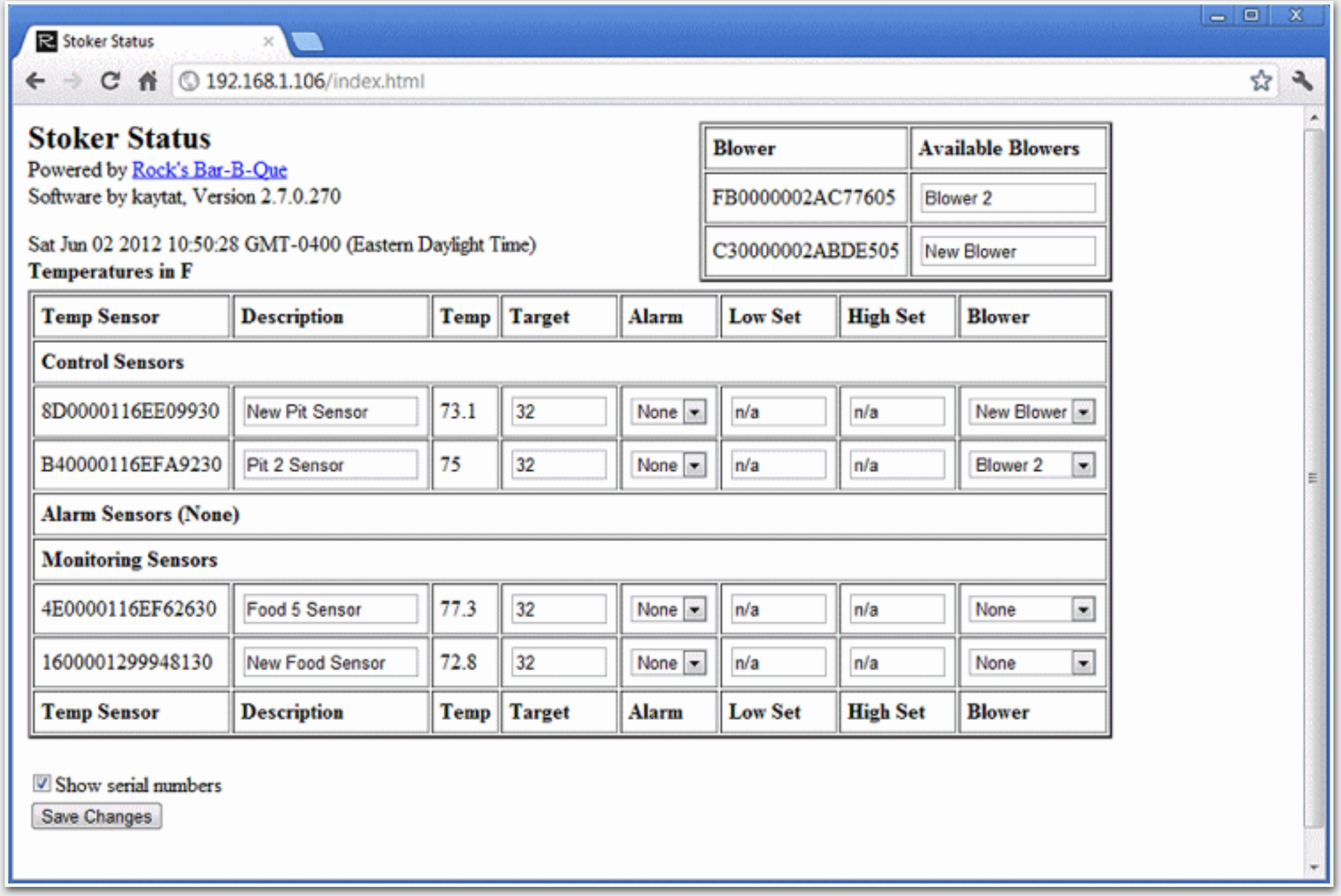


The Goal

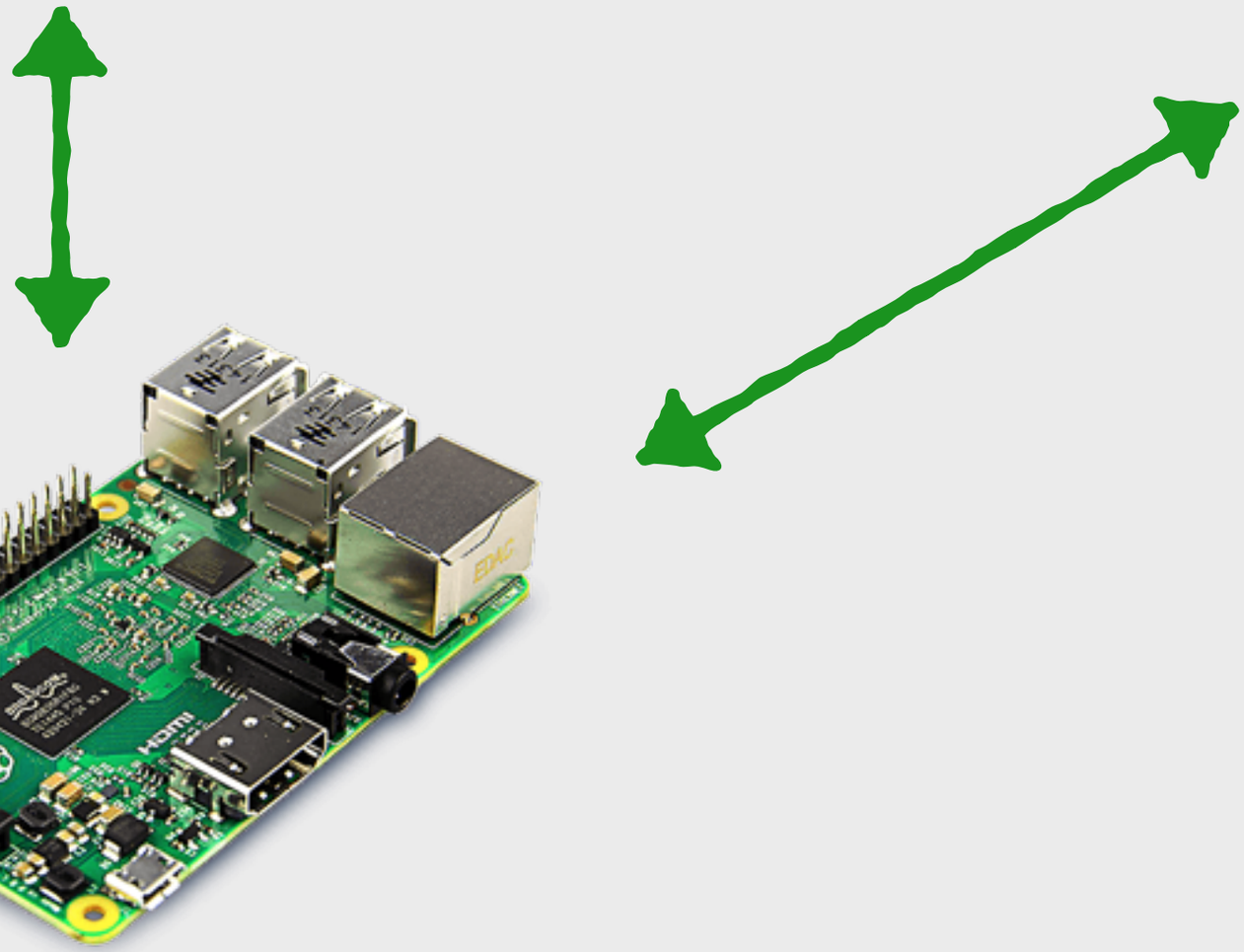

Rock's Bar-B-Que
STOKER II
Kaytat & K6JQ Software
Network
Wifi Enabled
Sel
Devices


amazon

The Problem


Stoker Status
192.168.1.106/index.html
Stoker Status
Powered by Rock's Bar-B-Que
Software by kaytat, Version 2.7.0.270
Sat Jun 02 2012 10:50:28 GMT-0400 (Eastern Daylight Time)
Temperatures in F
Blower
Available Blowers
FB0000002AC77605
Blower 2
C30000002ABDE505
New Blower
Temp Sensor
Description
Temp
Target
Alarm
Low Set
High Set
Blower
Control Sensors
8D0000116EE09930
New Pit Sensor
73.1
32
None
n/a
n/a
New Blower
B40000116EFA9230
Pit 2 Sensor
75
32
None
n/a
n/a
Blower 2
Alarm Sensors (None)
Monitoring Sensors
4E0000116EF62630
Food 5 Sensor
77.3
32
None
n/a
n/a
None
1600001299948130
New Food Sensor
72.8
32
None
n/a
n/a
None
Show serial numbers
Save Changes

The Solution



The Solution

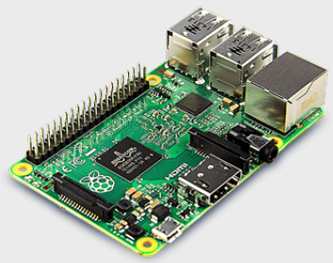
Echo

AWS Lambda

Firebase

AWS IoT

RPi 2



AWS Lambda

Upload your code to AWS Lambda

Set up your code to trigger from other AWS services, HTTP endpoints, or in-app activity

Lambda runs your code only when triggered, using only the compute resources needed

Pay just for the compute time you use

AWS Lambda

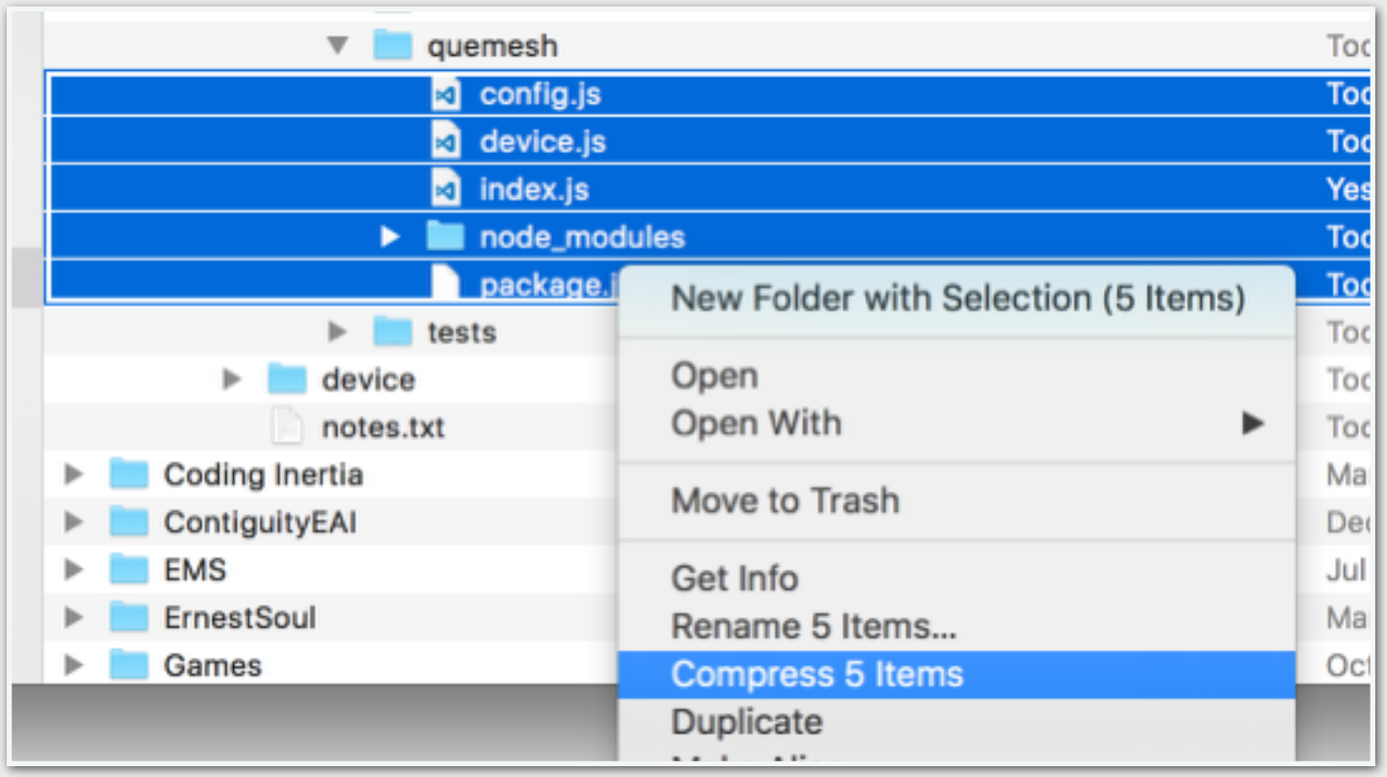
Creating a Lambda Function for an Alexa Skill

- Go to AWS Management Console -> AWS Lambda
- Select **US East (N. Virginia)** region
- Click **Get Started Now** or **Create a Lambda Function**
- Select an Alexa skills template
- Select **Alexa Skills Kit** for the **Event Source Type**
- Enter a **Name** and **Description**
- Set the role to **Basic Execution Role**
- Click **Create Function**

AWS Lambda

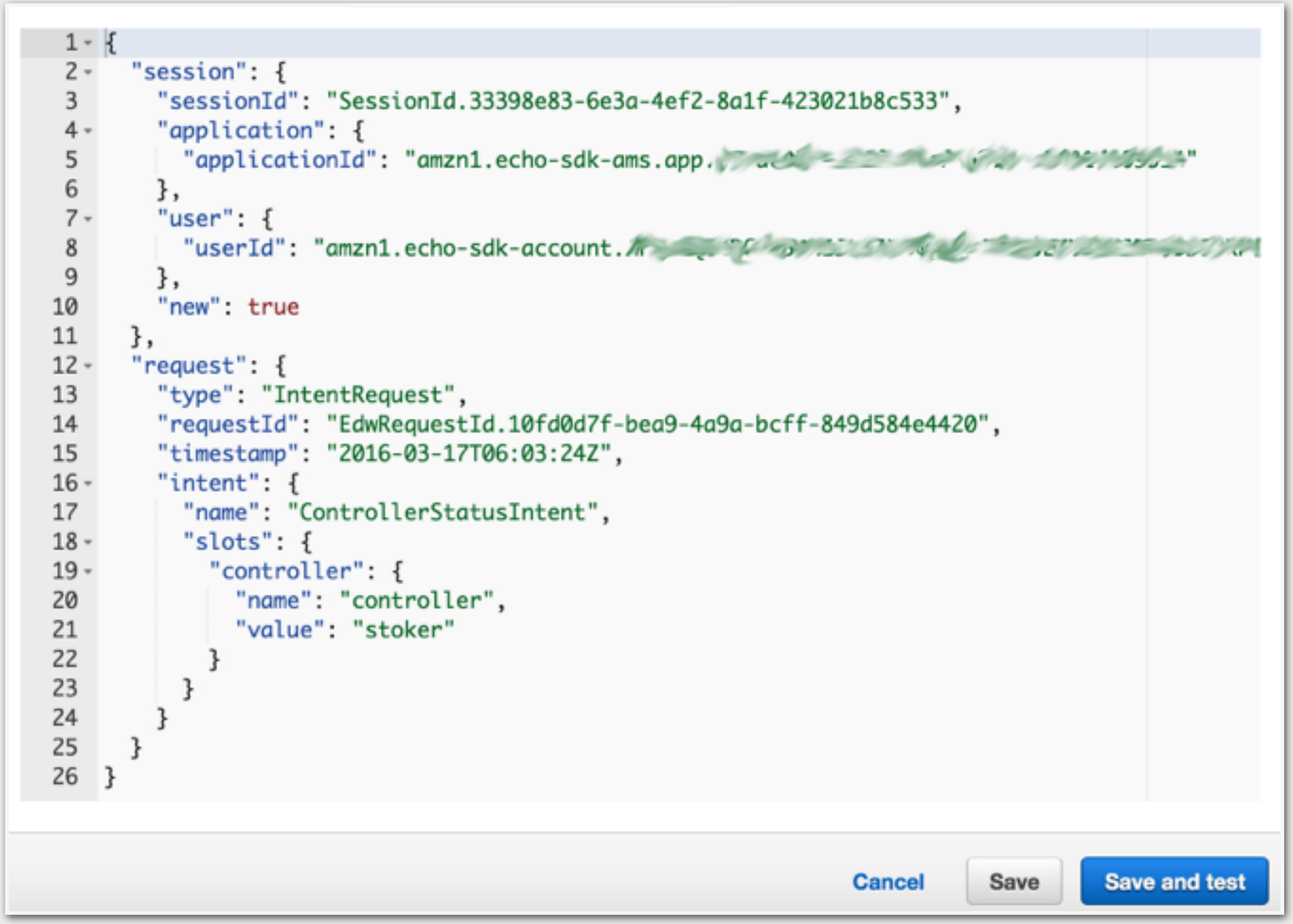
Creating a Lambda Function for an Alexa Skill

- Edit code directly in the **Code** tab
- Upload a .ZIP file
  - Zip the folder contents, not the folder



AWS Lambda

Testing an Alexa Skill Lambda Function



AWS Lambda

Testing an Alexa Skill Lambda Function

Execution result: succeeded (logs)

The area below shows the result returned by your function execution using the context methods. Learn more about returning results from your function.


```
{
  "version": "1.0",
  "sessionAttributes": {},
  "response": {
    "outputSpeech": {
      "type": "PlainText",
      "text": "Here is your Stoker status. Your Brisket is currently 65.6 degrees, it is 159.4 degrees below its target of 225 degrees. Your Pit is currently 69.4 degrees, it is 80.6 degrees be
    },
    "card": {
      "type": "Simple",
      "title": "SessionSpeechlet - ControllerStatusIntent",
      "content": "SessionSpeechlet - Here is your Stoker status. Your Brisket is currently 65.6 degrees, it is 159.4 degrees below its target of 225 degrees. Your Pit is currently 69.4 degrees,
```


Summary

Code SHA-256	VQ/g6N4BPm0jXgBCOclclJdORNN2 50DCNEln0MTzRoY=
Request ID	82a1a600-ed95-11e5-8397-d3a0dd51620e
Duration	2348.95 ms

Log output

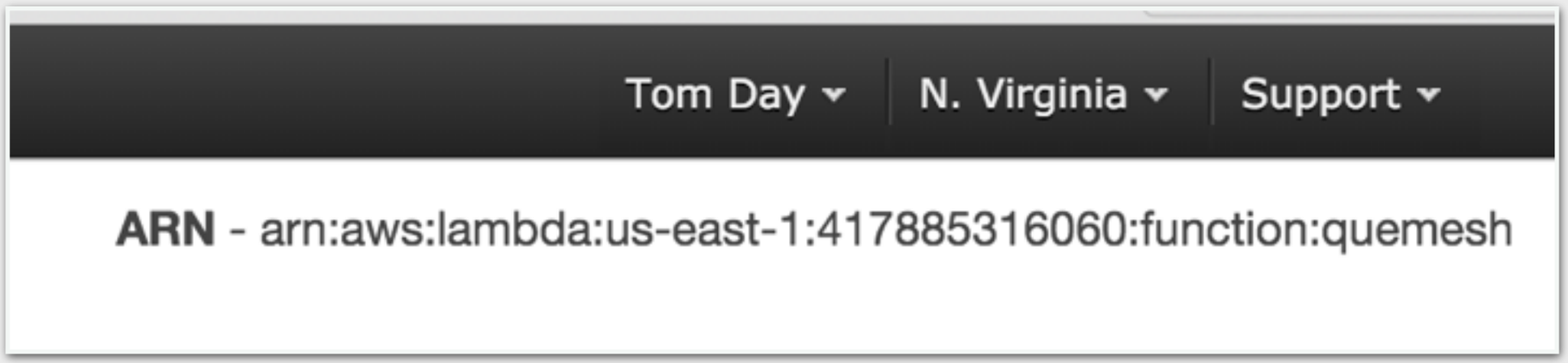
The area below shows the logging calls in your code. These correspond to a single row within the CloudWatch log group corresponding to this Lambda function. Click here to view the CloudWatch log group.


```
START RequestId: 82a1a600-ed95-11e5-8397-d3a0dd51620e Version: $LATEST
END RequestId: 82a1a600-ed95-11e5-8397-d3a0dd51620e
REPORT RequestId: 82a1a600-ed95-11e5-8397-d3a0dd51620e  Duration: 2348.95 ms    Billed Duration: 2400 ms        Memory Size: 1
```

AWS Lambda

Lambda Function ARN

ARN = Amazon Resource Name



Will be needed in the Alexa Skills Kit.

AWS Lambda

Device State Lambda Function

- Create the **deviceStateSave** function

Will be needed for the IoT Thing Rule.

Alexa Skills Kit

User

User asks a question or gives a command.

Alexa

Alexa identifies the skill name, then sends your skill a structured request.

Your Skill

Your skill processes the request and returns a text response.

Alexa converts the text to speech and streams it to the Echo.

The user hears the response.

Alexa Skills Kit

Registering an Alexa Skill

- Log into the Amazon Developer Portal
- Go to Apps & Services -> Alexa
- Click **Get Started** then **Add a New Skill**
- Enter the **Name**, **Invocation Name**, & **Endpoint (ARN)**
- Select the **Interaction Model** tab
- Create the **Intent Schema**

Alexa Skills Kit

Alexa Skill Intents & Utterances

intents.txt aws/alexa

```
{
  "intents": [
    {
      "intent": "ControllerStatusIntent",
      "slots": [
        {
          "name": "controller",
          "type": "LIST_OF_CONTROLLERS"
        }
      ]
    },
    {
      "intent": "ProbeTempIntent",
      "slots": [
        {
          "name": "controller",
          "type": "LIST_OF_CONTROLLERS"
        },
        {
          "name": "probe",
          "type": "LIST_OF_PROBES"
        }
      ]
    },
    {
```

utterances.txt aws/alexa

```
ControllerStatusIntent what is my {controller} status
ControllerStatusIntent get my {controller} status
ControllerStatusIntent give me my {controller} status
ProbeTempIntent what is my {controller} {probe} temperature
ProbeTempIntent get my {controller} {probe} temperature
ProbeTempIntent get my {controller} {probe}
ProbeTempIntent give me my {controller} {probe} temperature
```

Alexa Skills Kit

Alexa Skill Custom Slot Types

Custom Slot Types

Custom slot types to be referenced by the Intent Schema and Sample Utterances

For more information, see Defining the Voice Interface for an Alexa skill.

Example: TOPPINGS - cheese | onions | ham (note: newlines displayed as | for brevity)

Add Slot Type

Type	Values	
LIST_OF_ALARMS	None | Food | Fire	Edit
LIST_OF_CONTROLLE...	Stoker | Guru	Edit
LIST_OF_PROBES	Pit | Brisket | Pork | Ribs | Chicken	Edit

Alexa Skills Kit

Alexa Skill Testing

Service Simulator

Use Service Simulator to test your lambda function.

Text | Json

Enter Utterance *

what is my stoker status

Ask QueMesh Interface | Reset

Lambda Request

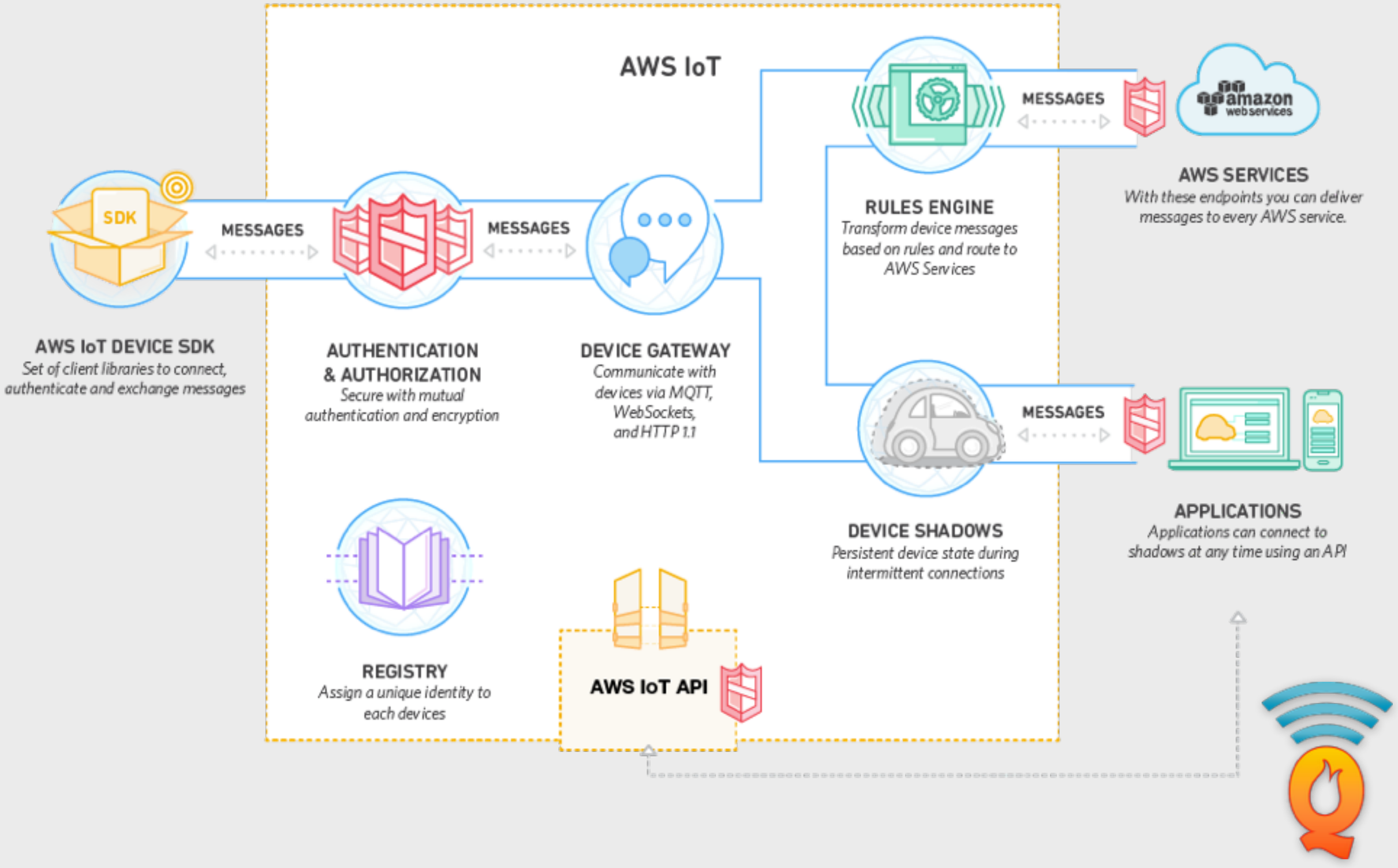

```
{
  "session": {
    "sessionId": "SessionId.2de4266e-d200-44dc-82
    "application": {
      "applicationId": "amzn1.echo-sdk-ams.app.57
    },
    "user": {
      "userId": "amzn1.echo-sdk-account.AFBKBQEYD
    },
    "new": true
  },
  "request": {
    "type": "IntentRequest",
    "requestId": "EdwRequestId.a26fbb80-1bd8-4ba2
    "timestamp": "2016-03-19T07:51:26Z",
    "intent": {
      "name": "ControllerStatusIntent",
      "slots": {
        "controller": {
          "name": "controller",
          "value": "stoker"
        }
      }
    }
```


Lambda Response


```
{
  "version": "1.0",
  "response": {
    "outputSpeech": {
      "type": "PlainText",
      "text": "Here is your Stoker status. Your B
    },
    "card": {
      "type": "Simple",
      "content": "SessionSpeechlet - Here is your
      "title": "SessionSpeechlet - ControllerStat
    },
    "reprompt": {
      "outputSpeech": {
        "type": "PlainText",
        "text": null
      }
    },
    "shouldEndSession": true
  },
  "sessionAttributes": {}
```


Listen

AWS IoT


AWS IoT
SDK
MESSAGES
MESSAGES
MESSAGES
MESSAGES
AWS IoT DEVICE SDK
Set of client libraries to connect, authenticate and exchange messages
AUTHENTICATION & AUTHORIZATION
Secure with mutual authentication and encryption
DEVICE GATEWAY
Communicate with devices via MQTT, WebSockets, and HTTP 1.1
RULES ENGINE
Transform device messages based on rules and route to AWS Services
amazon web services
AWS SERVICES
With these endpoints you can deliver messages to every AWS service.
DEVICE SHADOWS
Persistent device state during intermittent connections
APPLICATIONS
Applications can connect to shadows at any time using an API
REGISTRY
Assign a unique identity to each devices
AWS IoT API

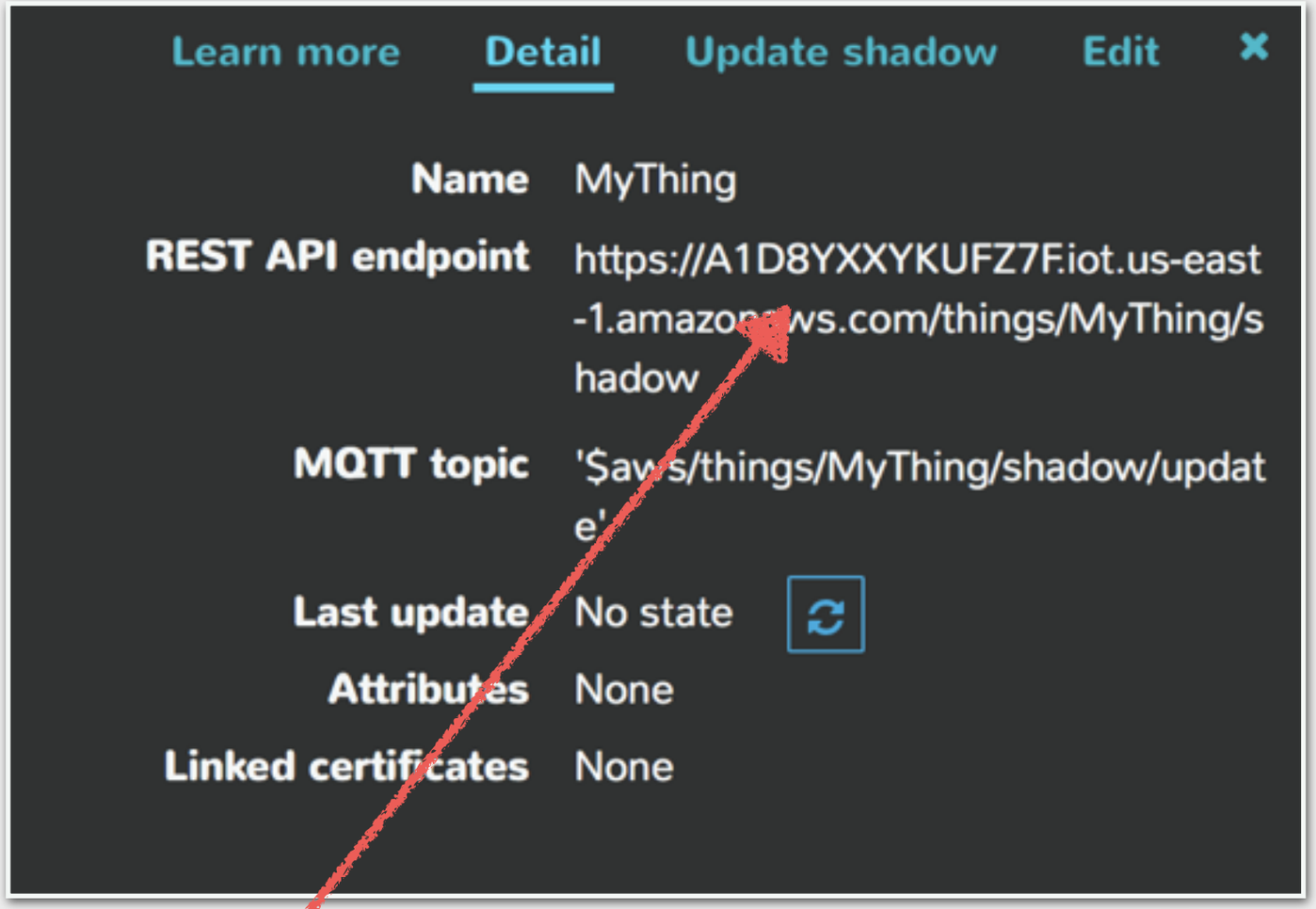
AWS IoT

Create an IoT Thing

- Log into the Amazon AWS Portal
- Go to AWS IoT
- Click **Get Started** then **Create a thing**
- Enter the **Name** then click **Create**

AWS IoT

Create an IoT Thing

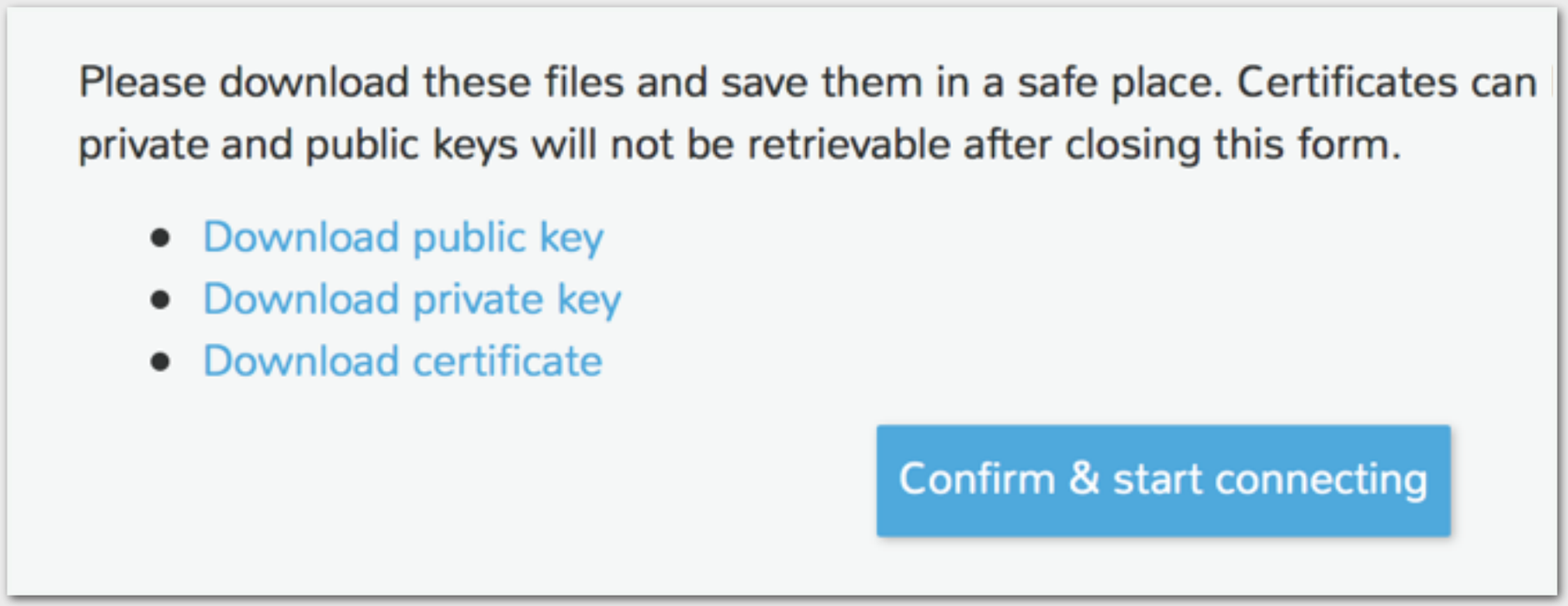


Host Url: A1D8YXXYKUFZ7F.iot.us-east-1.amazonaws.com

AWS IoT

Connect A Device

- Click **Connect a device**
- Select **NodeJS**
- Click **Generate certificate and policy**
- Download certificate & policy files



AWS IoT

Connect A Device

- Click **Confirm**

AWS IoT Node.js SDK

Download the AWS IoT Node.js SDK.

Set up the SDK using the instructions in our README on GitHub.

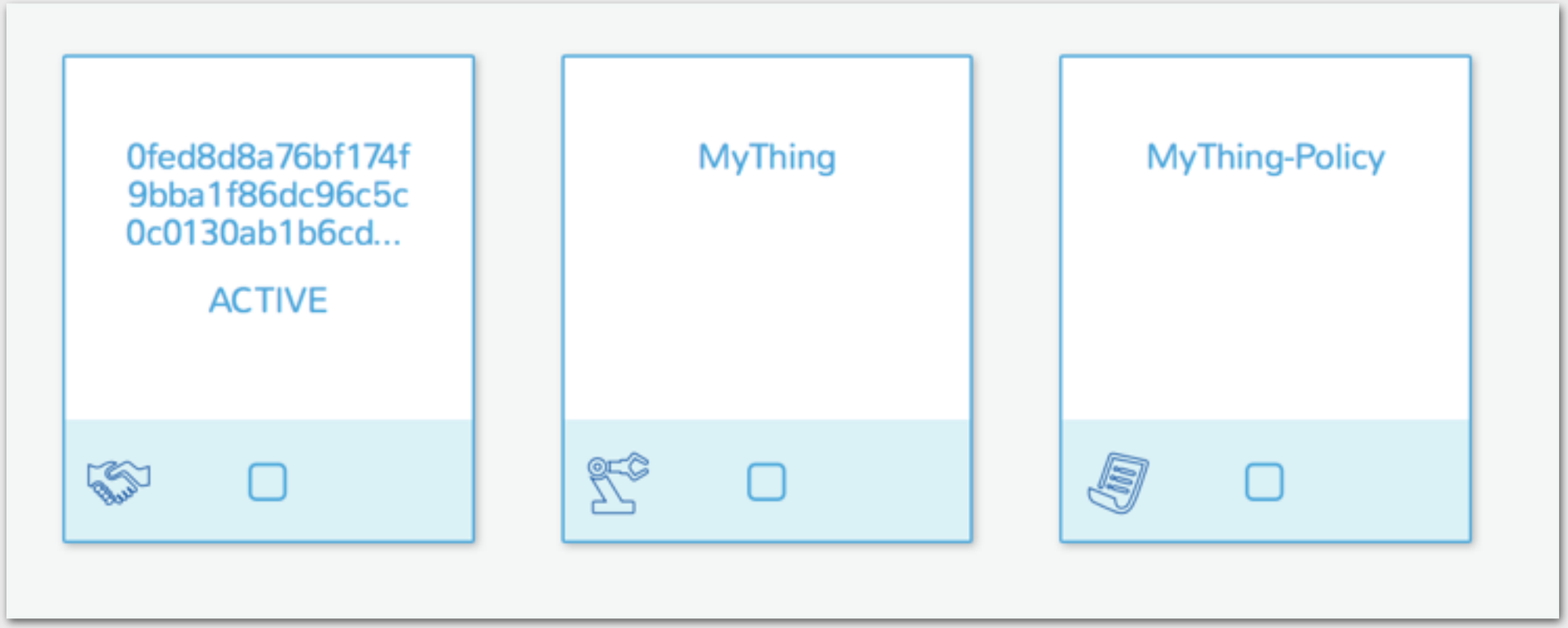
Add in the following sample code based on your account, Thing, and new certificate:

```
{
        "host": "A1D8YXXYKUFZ7F.iot.us-east-1.amazonaws.com",
        "port": 8883,
        "clientId": "MyThing",
        "thingName": "MyThing",
        "caCert": "root-CA.crt",
        "clientCert": "0fed8d8a76-certificate.pem.crt",
        "privateKey": "0fed8d8a76-private.pem.key"
}
```

AWS IoT

Connect A Device

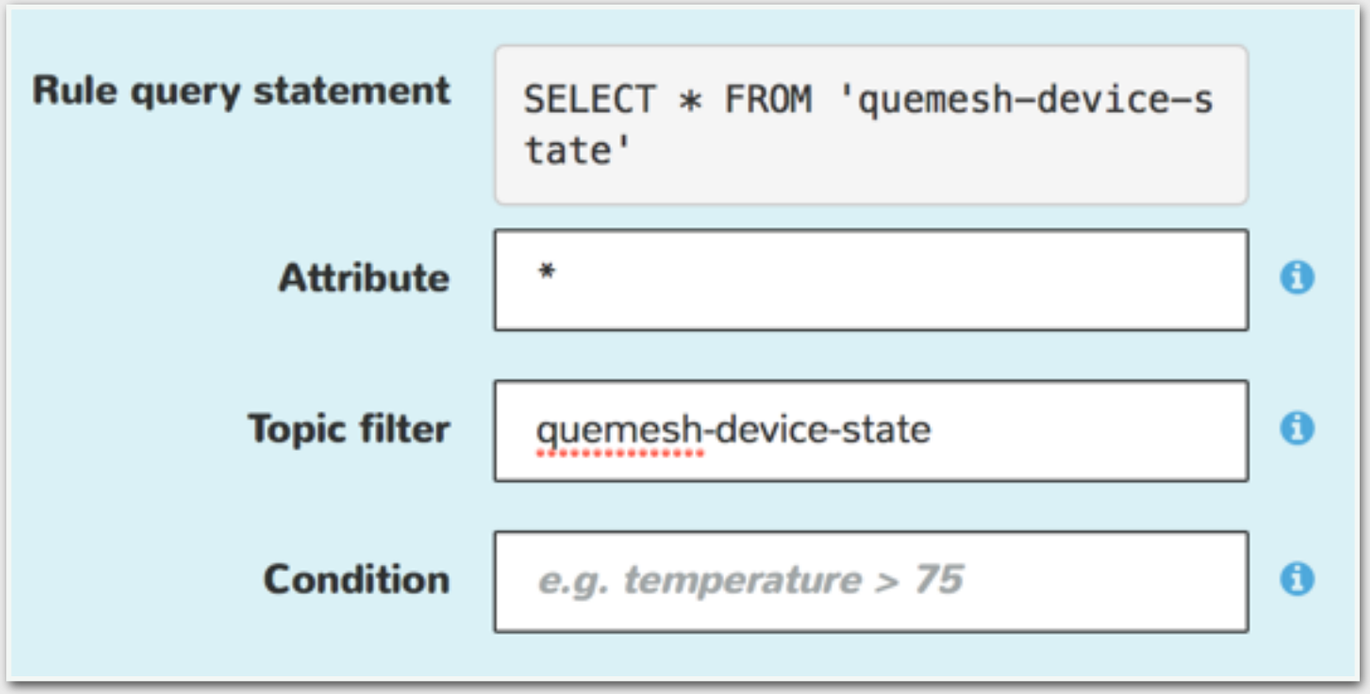
- Click **Return to thing detail**



AWS IoT

Create A Rule

- Click **Create a rule**
- Enter a **Name** & **Description**
- Enter a **Query**: SELECT * FROM ‘quemesh-device-state’



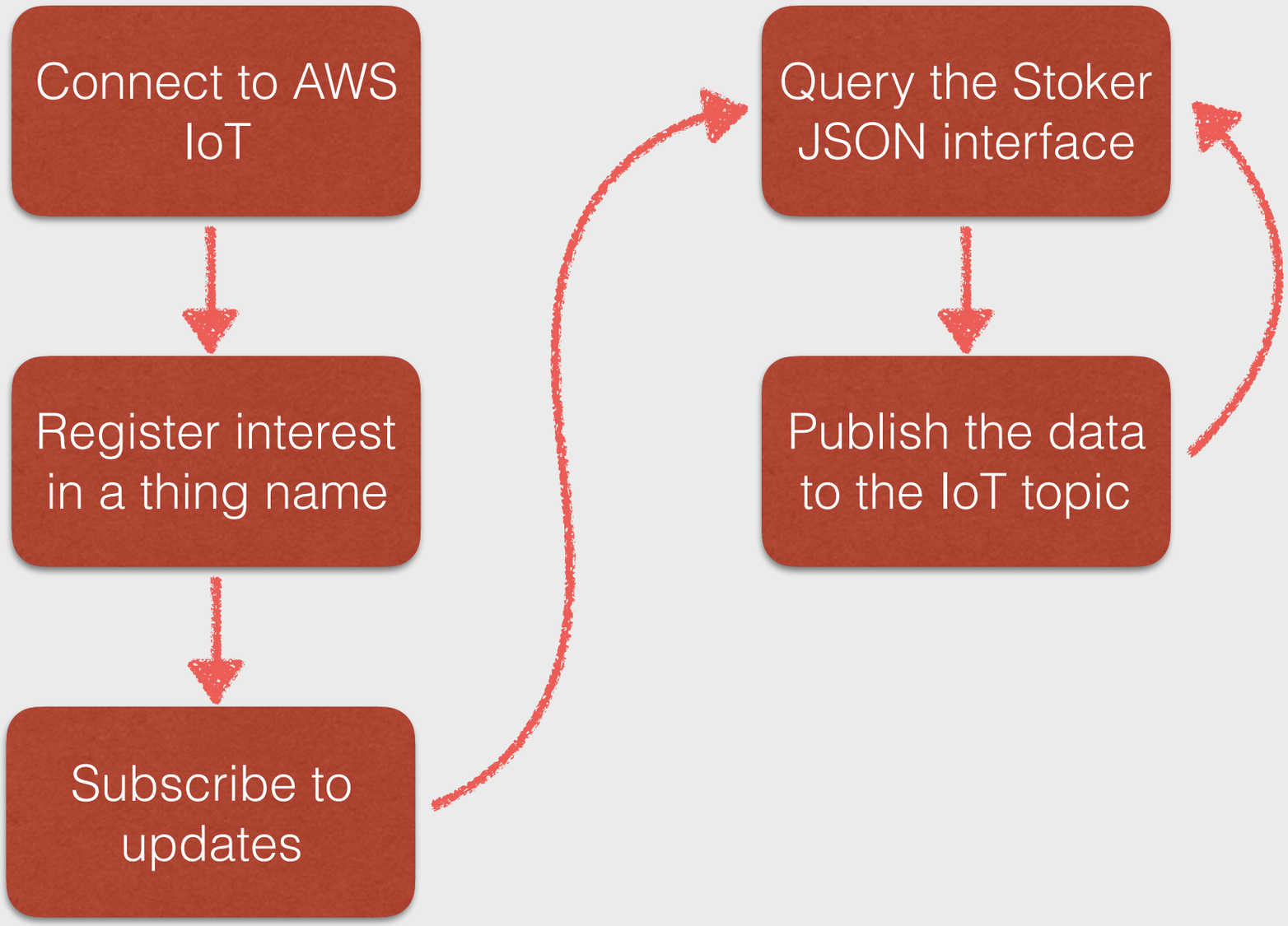
AWS IoT

Create A Rule

- Click **Choose an action**
- Choose:
  - Insert this message into a code function and execute it (Lambda)
- For **Function name** choose:
  - deviceStateSave

AWS IoT

Device Code



Questions

https://github.com/javaday/EchoIotDemo

javaday@gmail.com