

Cloud Computing Project

# Light – Sound Sensing Alert System

Ranjitha Rajeeva Shetty and Vartika Srivastava

---



# Table of Contents

<b>Introduction</b>	<b>2</b>
<b>Technology</b>	<b>2</b>
Cloud Computing	2
IoT – Internet of Things	3
<b>Tools and Programming Language</b>	<b>3</b>
MQTT	3
AWS EC2	3
IoT Core	3
IoT Thing	4
SNS	4
AWS Lambda	4
<b>Application Workflow</b>	<b>5</b>
<b>Implementation</b>	<b>6</b>
Initial Setup - Key Pair, Stack and EC2 instance	6
Running on the Terminal	8
Register IoT - Thing, Policy and Certificate	9
Running the simulator	12
Creating IoT Rule and Action	13
Creating Lambda Function	16
<b>Results</b>	<b>18</b>
<b>Future enhancements</b>	<b>18</b>
<b>Reference</b>	<b>19</b>

# 1. Introduction

The purpose of this project is to utilize the fast-developing cloud technology and IoT system for a better lifestyle. In today's time, a lot of focus is given to promote a healthier lifestyle. Extensive efforts are being made using cloud technologies like creating activity tracker system, heart rate sensing application, etc. With the help of AWS cloud services, we have created a system which reads in the data of light and sound intensity from a mobile device in the room to provide us the information if a person is in a room which has light beyond the optimal range or if the intensity of sound is at harmful levels. The notification sent to his email or phone based on the preference set by the individual. The application of such systems can be effective in health care provider environment with an goal to provide comfortable and healthier surrounding for patient recovery and treatment.

## 2. Technology

### 2.1. Cloud Computing

It is an on-demand availability of computer system resources like data storage, computing power other IT resources through a cloud services platform via the Internet with pay-as-you-go pricing without direct active management by the user. Large clouds, predominant today, often have functions distributed over multiple locations from central servers. If the connection to the user is relatively close, it may be designated an edge server.

For this project, we are using Amazon Web Services to connect the IOT application

### 2.2. IoT – Internet of Things

The Internet of Things (IoT) is an extension of connectivity over the internet into physical devices which are embedded with sensors, software. The devices can interact with other devices over the network and can also be controlled or monitored remotely. Devices like cellphones, washing

machines, headphones, lamps, wearable devices are few of the examples that can connect over the internet.

## 3. Tools and Programming Language

### 3.1. MQTT

Message Queuing Telemetry Transport. It is designed as a lightweight messaging protocol that uses publish/subscribe operations to exchange data between clients and the server. Furthermore, its small size, low power usage, minimized data packets and ease of implementation make the protocol ideal of the “machine-to-machine” or “Internet of Things” world.[1]

### 3.2. AWS EC2

EC2 is a virtual server in Amazon’s Elastic Compute Cloud. EC2 is a service that allows business subscribers to run application programs in the computing environment. The EC2 can serve as a practically unlimited set of virtual machines.[2]

### 3.3. IoT Core

IoT Core is a managed cloud service that lets connected devices easily and securely interact with cloud applications and other devices. We can support billions of devices and trillions of messages and can process and route those messages to AWS endpoints and to other devices reliably and securely. [3]

### 3.4. IoT Thing

It is the extension of internet connectivity into physical devices and everyday objects. Embedded with electronics, internet connectivity, and other forms of hardware, these

devices can communicate and interact with others over the internet, and they can be remotely monitored and controlled.

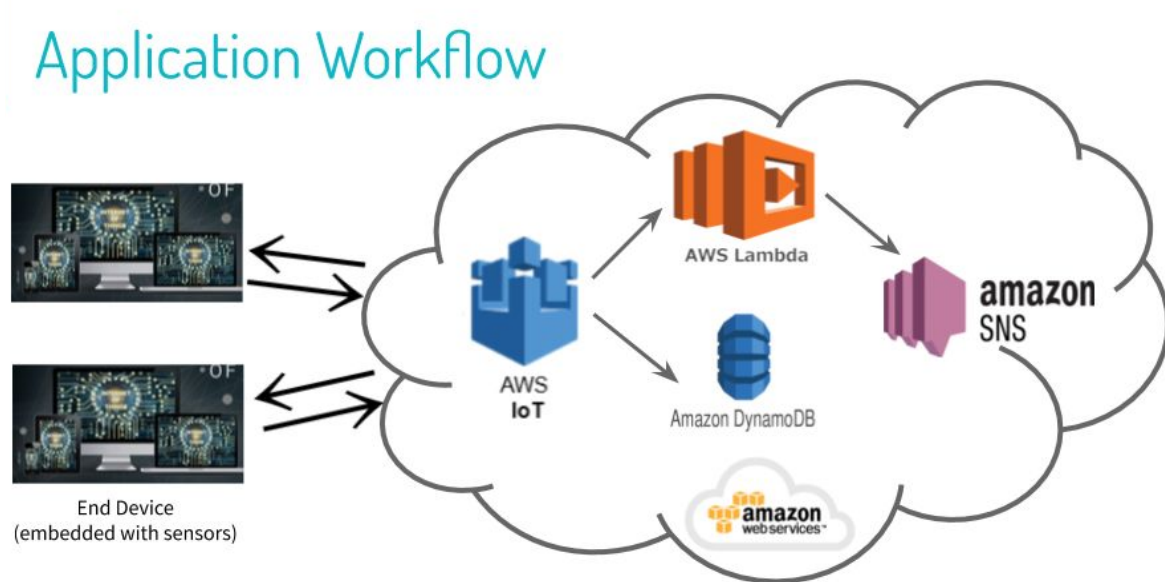
### 3.5. SNS

Simple Notification system (SNS), a web service that it is easy to and send notification via cloud system. The system provides a very scalable, cost-effective capability to broadcast messages from the application created and deliver the messages to a set of subscriber or application (for our project we have used SMS on our phones).

### 3.6. AWS Lambda

AWS Lambda is an event-driven computing cloud service from Amazon Web Services that allows developers to program functions on a pay-per-use basis without having to provision storage or compute resources to support them

## 4. Application Workflow



*Figure 1: Our Application workflow*

**Step 1:** Using the MQTT protocol devices (such as mobile, Tablets and other devices connected to the internet) send light and sound data from the sensors to AWS IoT.

**Step 2:** Actions are performed based on the data received, rules are analyzed by the IoT rule Engine (In our application workflow AWS IoT represents the rule engine)

**Step 3:** Two actions are performed when the rule IoTToDynamo is triggered: the first one stores the incoming data to the DynamoDB and the second one invokes the lambda function IOT-LambdaForSNS.

**Step 4:** When the sound and lights are out of the ideal range, within the lambda function AWS SNS is used to publish a message to the specific user.

## 5. Implementation

### 5.1. Initial Setup - Key Pair, Stack and EC2 instance

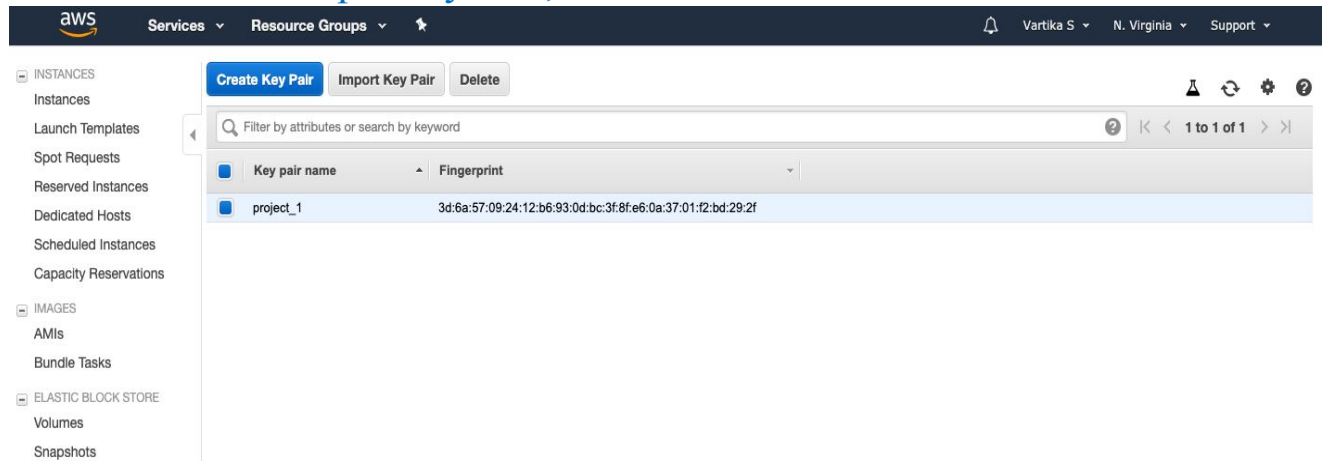


Figure 2: Key pair creation

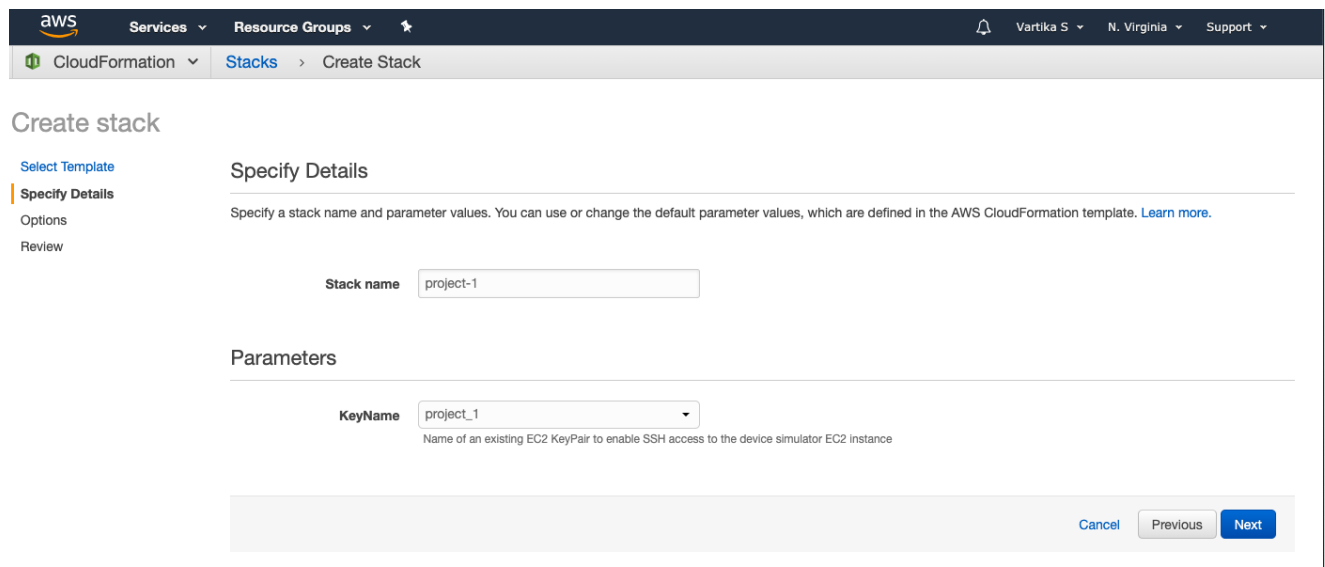


Figure 3: Creating a Stack

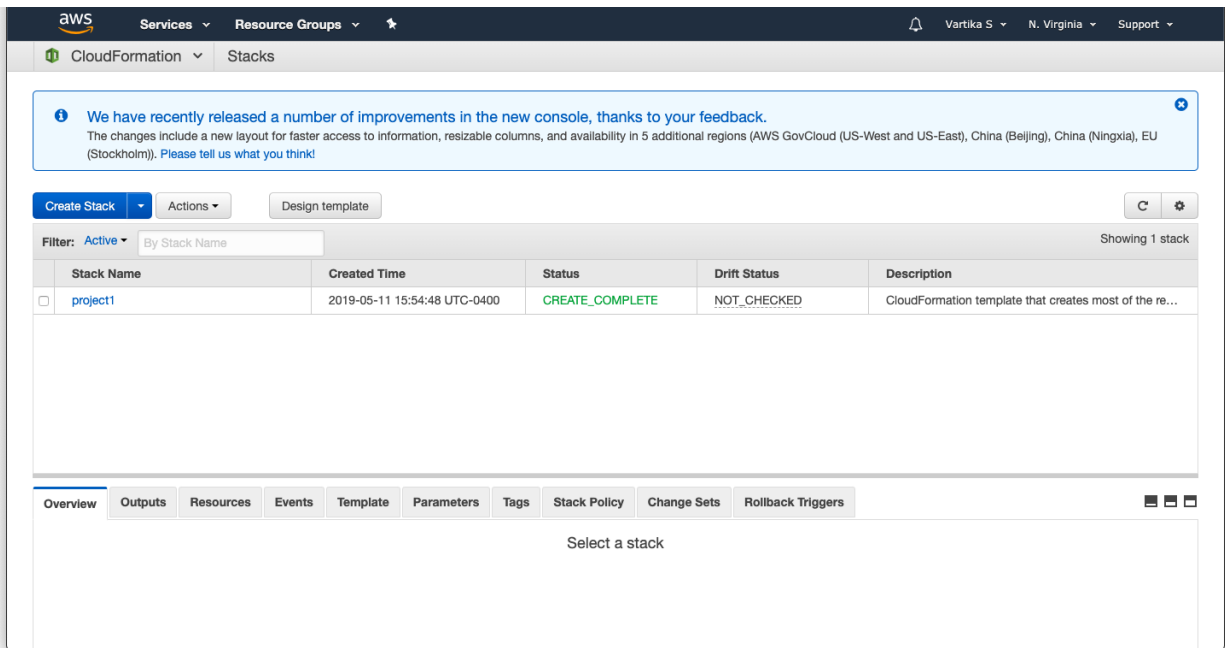


Figure 4: Stack creation successful

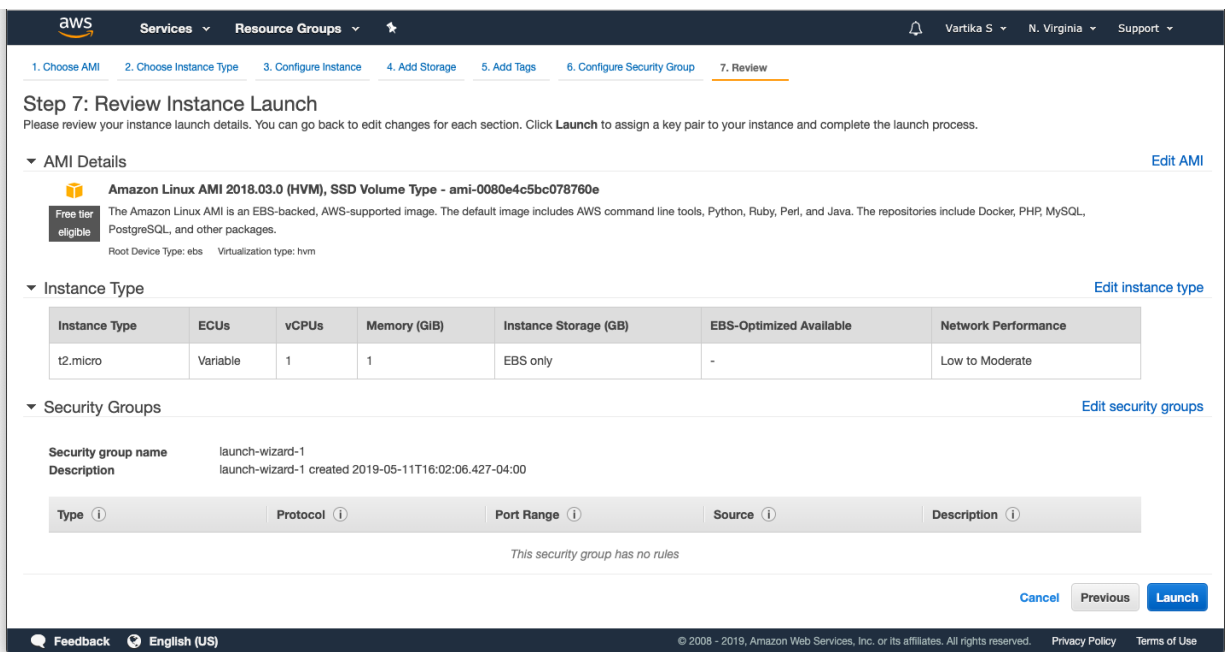


Figure 5: Setting up a EC2 instance



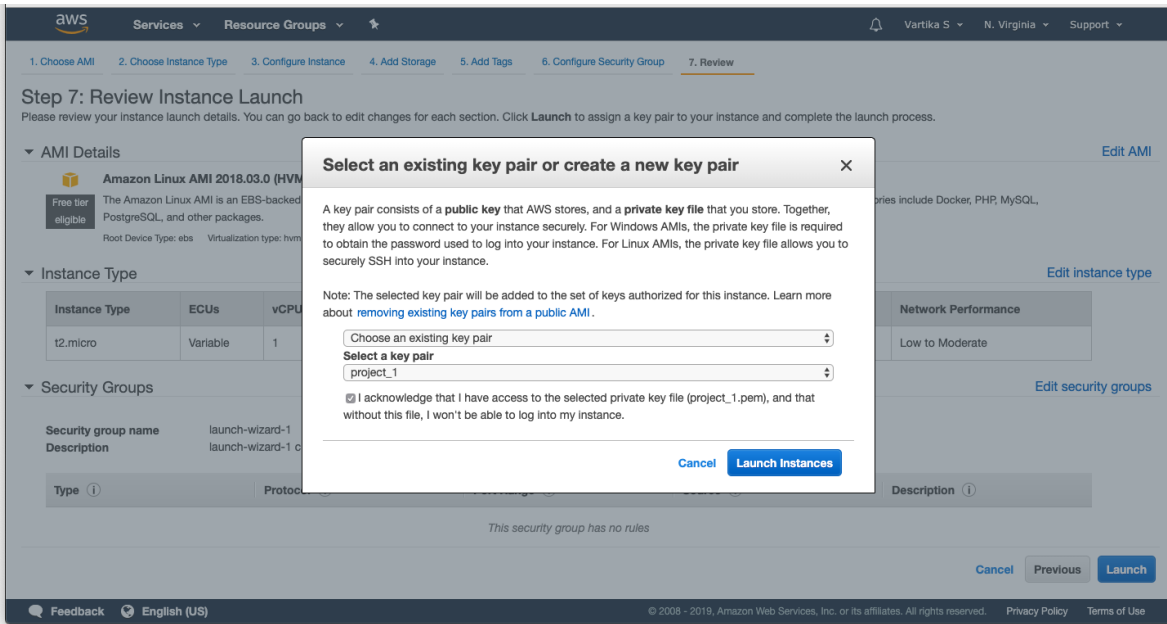


Figure 6: Linking the Key pair with EC2 instance

## 5.2. Running on the Terminal

```

Downloads — ec2-user@ip-10-0-0-182:~ — ssh -i project_1.pem ec2-user@52.90.0.147 — 149x33
opt i20_20190314150857.pdf
price.csv
project_1.pem
sensors-15-25474.pdf
sensors-17-01287-v2.pdf
state_New_York.pdf
vartika.pdf
vs1922.pdf
~$EAD_ME.doc
~$Predictors.xlsx
~$itten Homework Assignment after Class 4.docx
(Vartikas-MacBook-Air:Downloads vartikasrivastava$ chmod 400 project_1.pem
(Vartikas-MacBook-Air:Downloads vartikasrivastava$ ssh -i IoT-GettingStarted-Key.pem ec2-user@52.90.0.147
Warning: Identity file IoT-GettingStarted-Key.pem not accessible: No such file or directory.
The authenticity of host '52.90.0.147 (52.90.0.147)' can't be established.
ECDSA key fingerprint is SHA256:Bbu96Qzje8UYb9oWuMv5EiHnusAiUwRjqo/c9QGPDtU.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '52.90.0.147' (ECDSA) to the list of known hosts.
ec2-user@52.90.0.147: Permission denied (publickey).
(Vartikas-MacBook-Air:Downloads vartikasrivastava$ ssh -i IoT-GettingStarted-Key.pem ec2-user@52.90.0.147
Warning: Identity file IoT-GettingStarted-Key.pem not accessible: No such file or directory.
ec2-user@52.90.0.147: Permission denied (publickey).
(Vartikas-MacBook-Air:Downloads vartikasrivastava$ ssh -i project_1.pem ec2-user@52.90.0.147

  _ _ | _ _ | _ )
  _ | ( _ /   Amazon Linux AMI
  _ _ | \ _ _ | _ _ |

https://aws.amazon.com/amazon-linux-ami/2016.09-release-notes/
26 package(s) needed for security, out of 79 available
Run "sudo yum update" to apply all updates.
Amazon Linux version 2018.03 is available.
[ec2-user@ip-10-0-0-182 ~]$

```

Figure 7: Connecting to the newly created EC2 instance from Terminal

## 5.3. Register IoT - Thing, Policy and Certificate

The screenshot shows the 'CREATE A THING' wizard in the AWS IoT console. The title bar indicates 'STEP 1/3' and 'Add your device to the thing registry'. The main content area is divided into three sections:

- This step creates an entry in the thing registry and a thing shadow for your device.**  
Name:
- Apply a type to this thing**  
Using a thing type simplifies device management by providing consistent registry data for things that share a type. Types provide things with a common set of attributes, which describe the identity and capabilities of your device, and a description.  
Thing Type:  [Create a type](#)
- Add this thing to a group**  
Adding your thing to a group allows you to manage devices remotely using jobs.  
Thing Group:  [Create group](#) [Change](#)

The footer includes 'Feedback', 'English (US)', and copyright information: '© 2008 - 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use'.

Figure 8: Creating a Thing

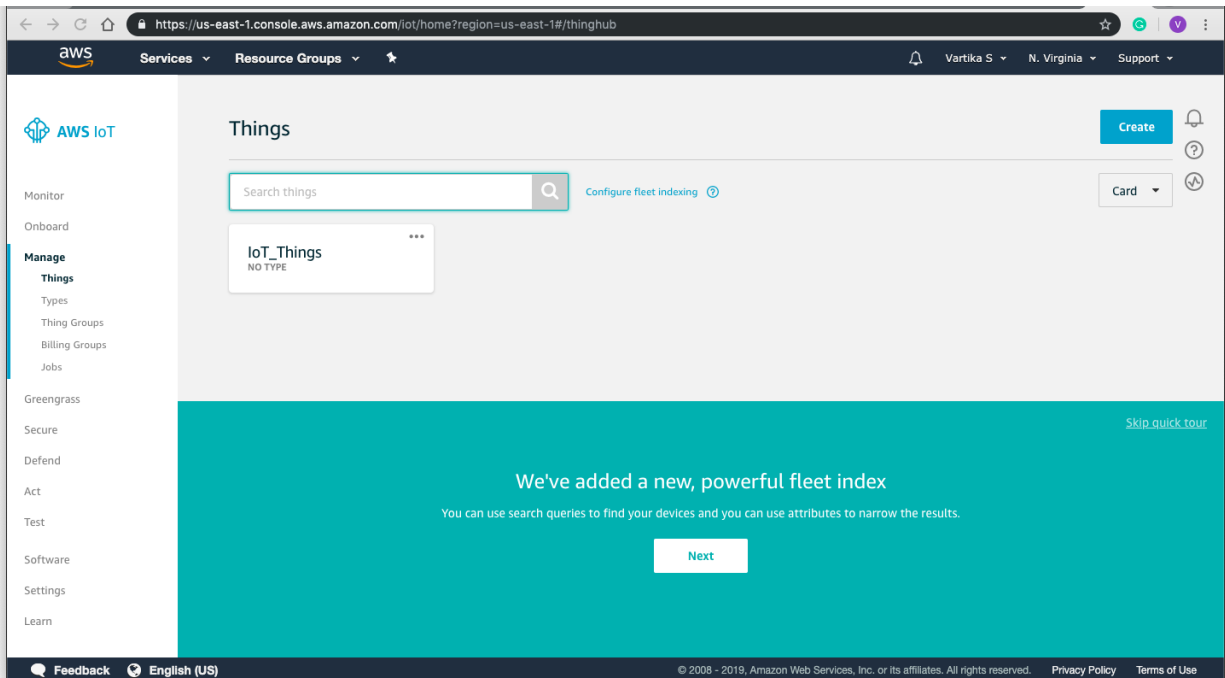
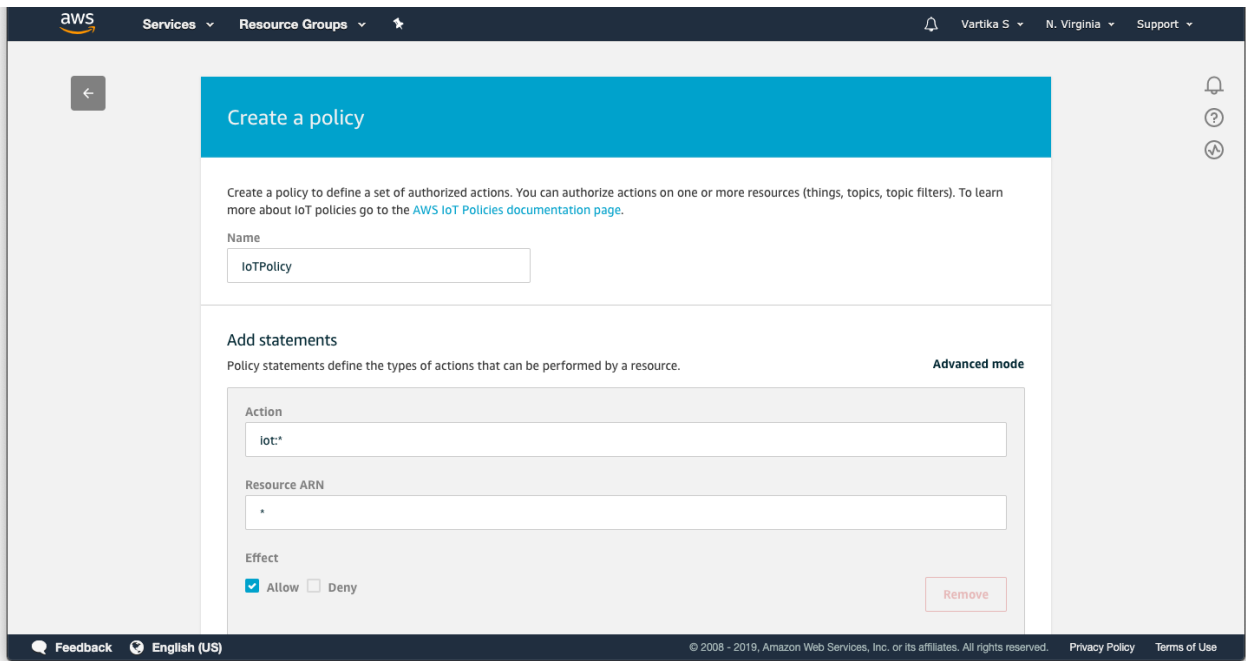
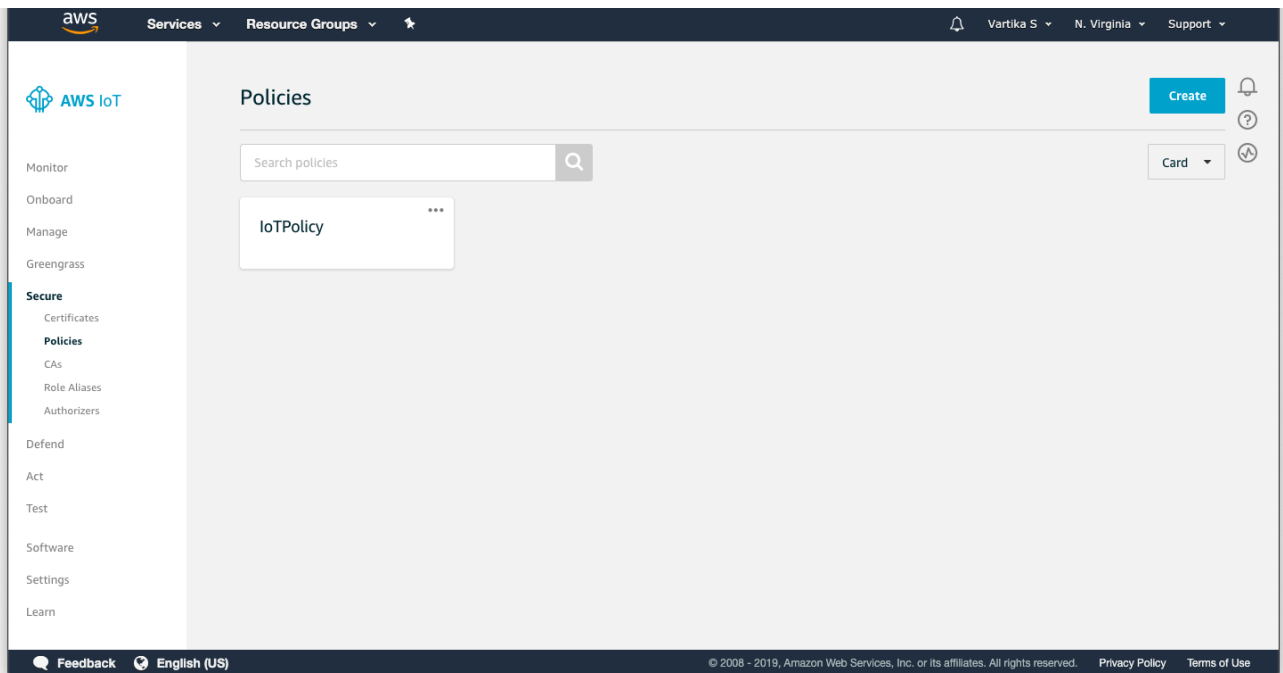


Figure 9: Successfully created a Thing



*Figure 9: Creating a Policy for the thing*



*Figure 10: Successful creation of a Policy for the thing*

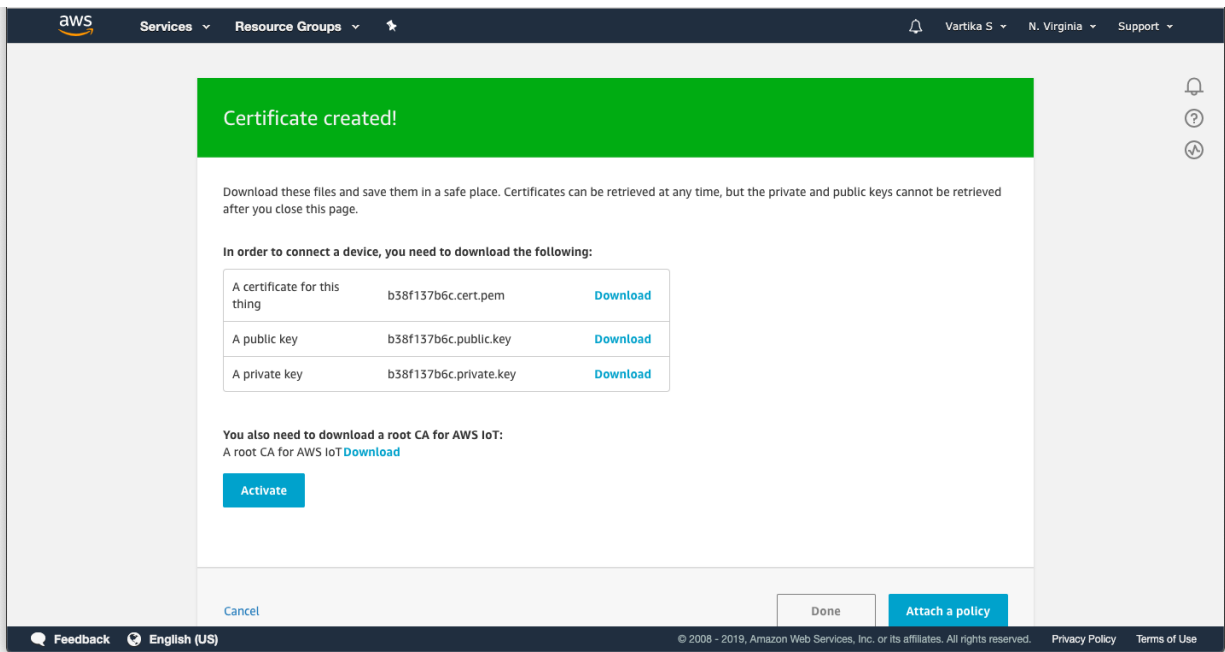


Figure 11: Accessing the certificates

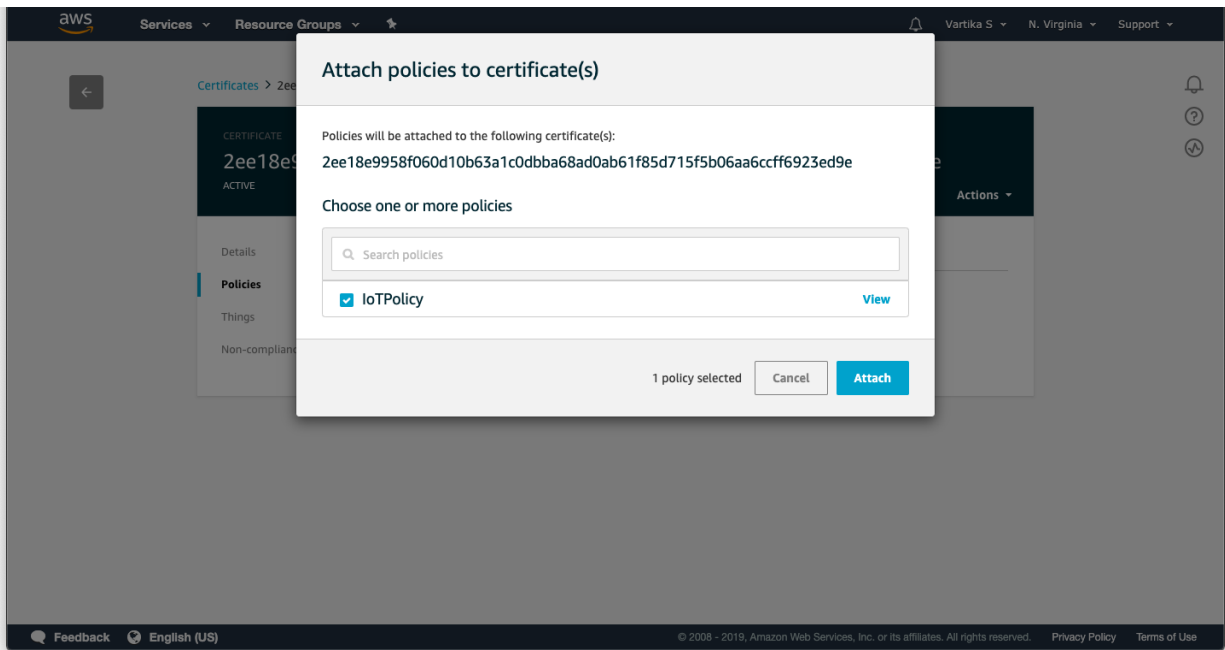


Figure 12: Attaching the certificates

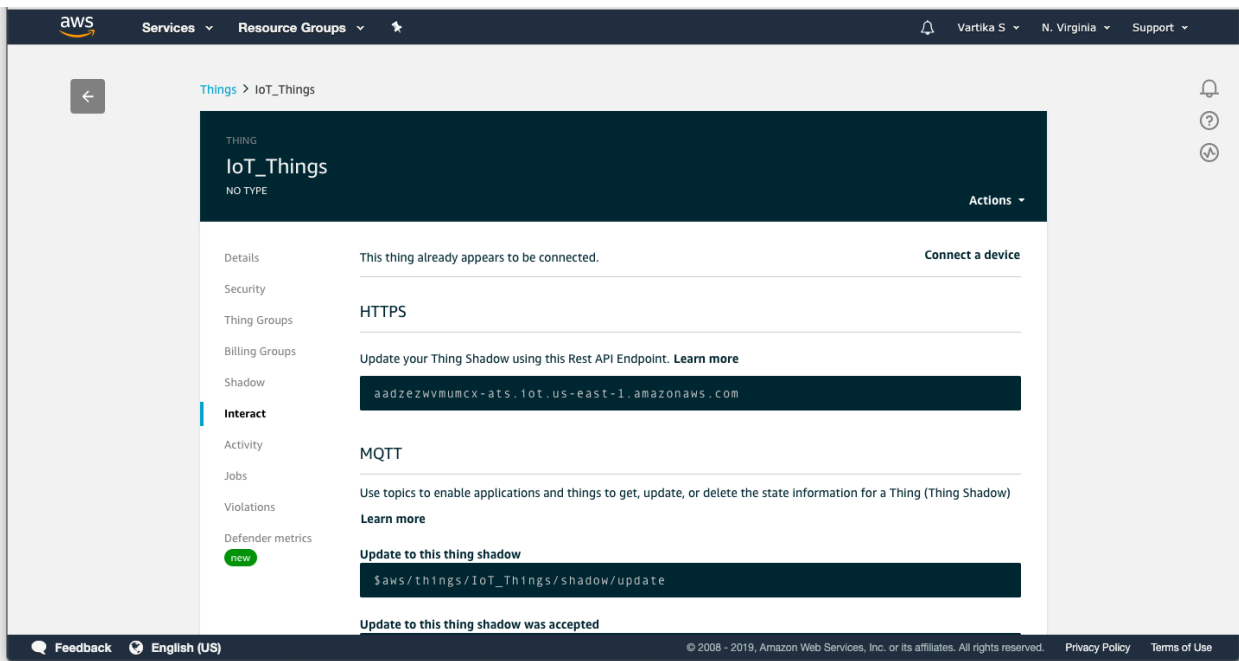


Figure 13: Screenshot of the Endpoint information

## 5.4. Running the simulator

```

Downloads — ec2-user@ip-10-0-0-182:~ — ssh -i project_1.pem ec2-user@52.90.0.147 — 149x34
The authenticity of host '52.90.0.147 (52.90.0.147)' can't be established.
ECDSA key fingerprint is SHA256:Bbu96Qzje8UYb9oWuMvSEiHnusAiUwRjqo/c9Q8PdtU.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '52.90.0.147' (ECDSA) to the list of known hosts.
ec2-user@52.90.0.147: Permission denied (publickey).
[Vartikas-MacBook-Air:Downloads vartikasrivastava$ ssh -i IoT-GettingStarted-Key.pem ec2-user@52.90.0.147
Warning: Identity file IoT-GettingStarted-Key.pem not accessible: No such file or directory.
ec2-user@52.90.0.147: Permission denied (publickey).
[Vartikas-MacBook-Air:Downloads vartikasrivastava$ ssh -i project_1.pem ec2-user@52.90.0.147

  _ _ _ _ _
 _ _ _ _ _ / Amazon Linux AMI
 _ _ _ _ _

https://aws.amazon.com/amazon-linux-ami/2016.09-release-notes/
26 package(s) needed for security, out of 79 available
Run "sudo yum update" to apply all updates.
Amazon Linux version 2018.03 is available.
[ec2-user@ip-10-0-0-182 ~]$ ls -l /certs
ls: invalid option -- '/'
Try 'ls --help' for more information.
[ec2-user@ip-10-0-0-182 ~]$ ls -l /certs
ls: invalid option -- '/'
Try 'ls --help' for more information.
[ec2-user@ip-10-0-0-182 ~]$ ls -l /certs
certificate.pem.crt private.pem.key root-ca.pem
[ec2-user@ip-10-0-0-182 ~]$ nano ~/settings.py
[ec2-user@ip-10-0-0-182 ~]$ nano ~/settings.py
[ec2-user@ip-10-0-0-182 ~]$ nohup python app.py
nohup: ignoring input and appending output to 'nohup.out'
[ec2-user@ip-10-0-0-182 ~]$ nohup python app.py &
[1] 23744
[ec2-user@ip-10-0-0-182 ~]$ nohup: ignoring input and appending output to 'nohup.out'

```

Figure 14: Running our device simulator with the code created for Light - Sound sensing application

## 5.5. Creating IoT Rule and Action

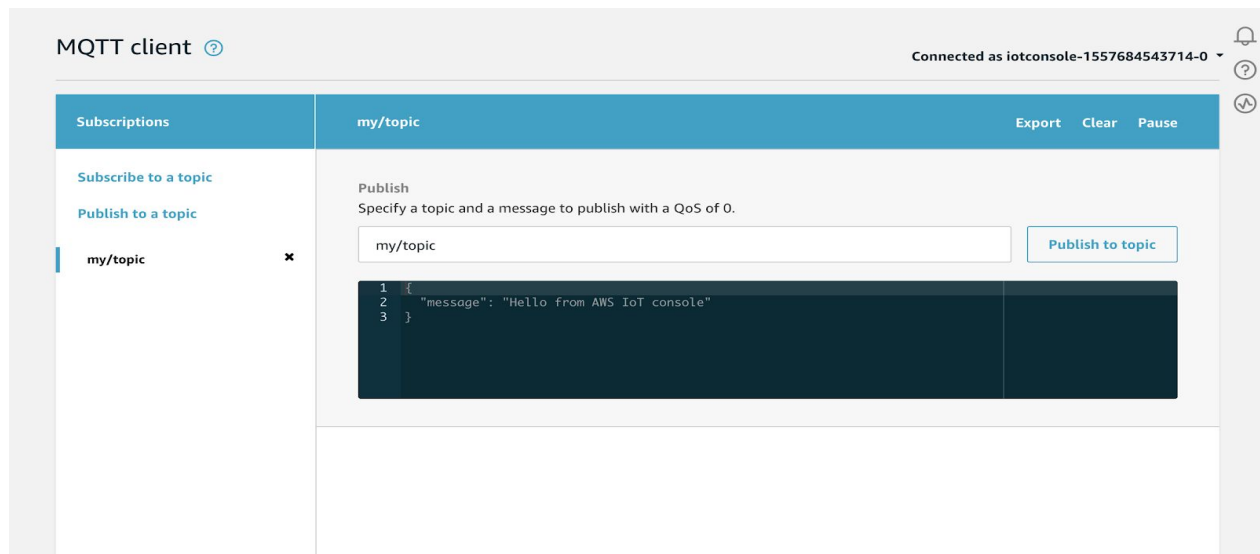


Figure 15.1: IoT Rule

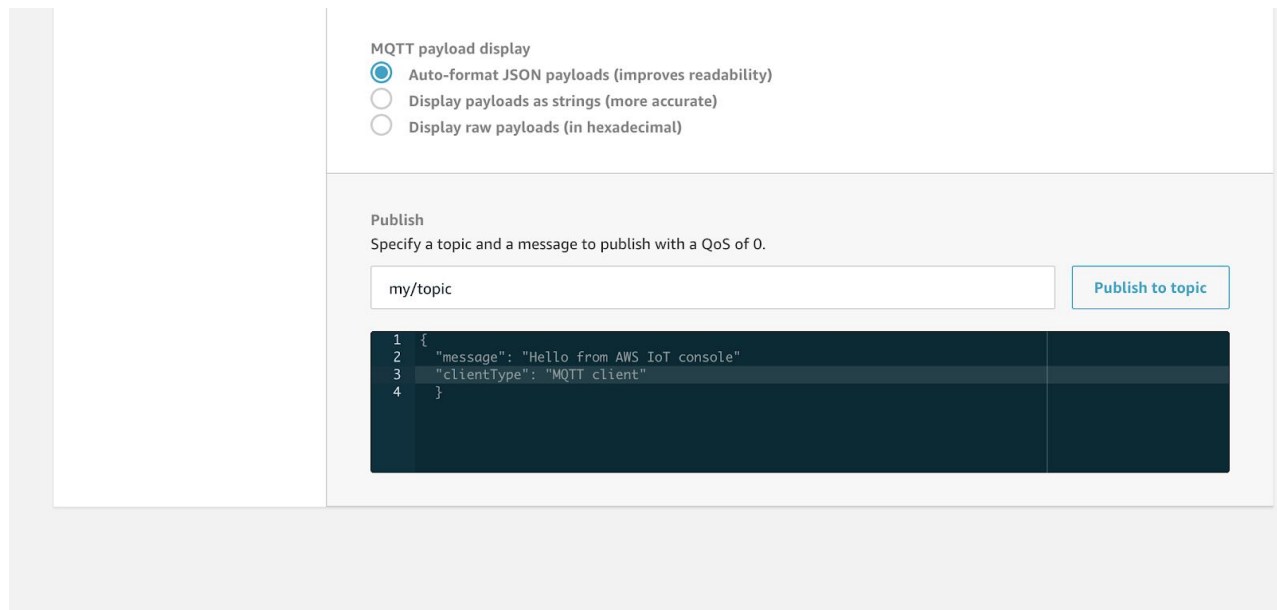


Figure 15.2: IoT Rule

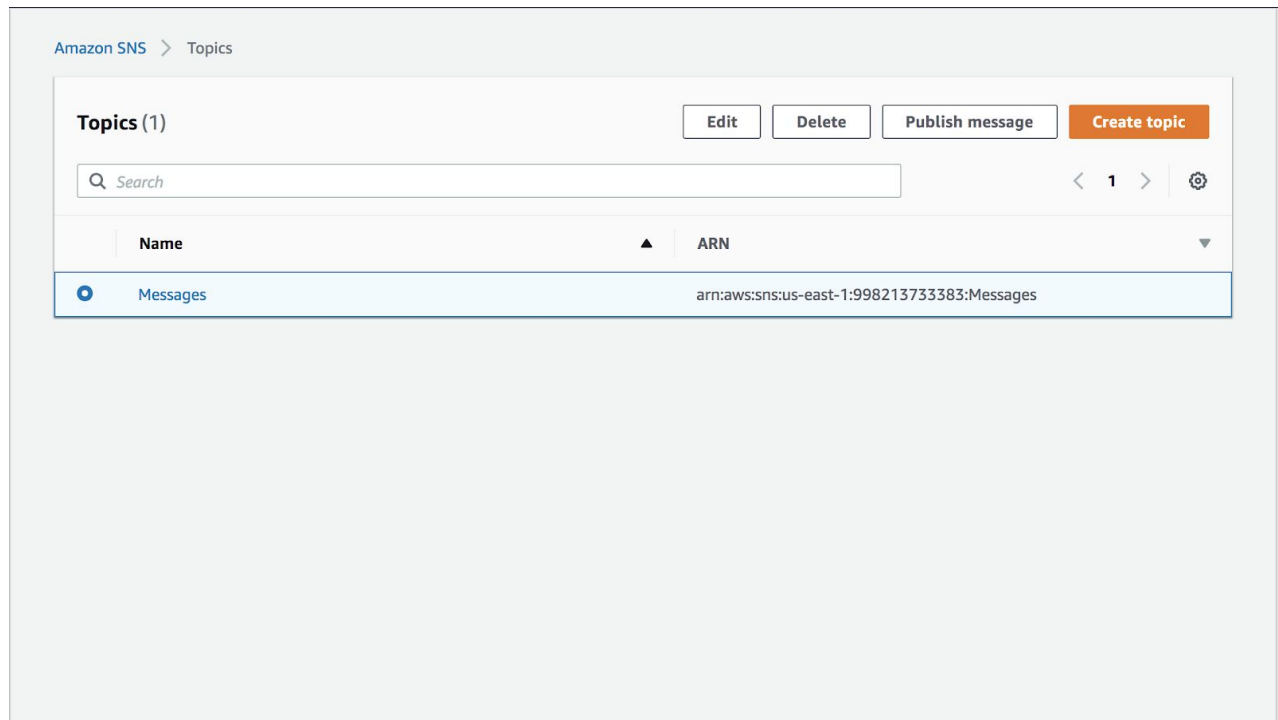


Figure 16: IoT Topic

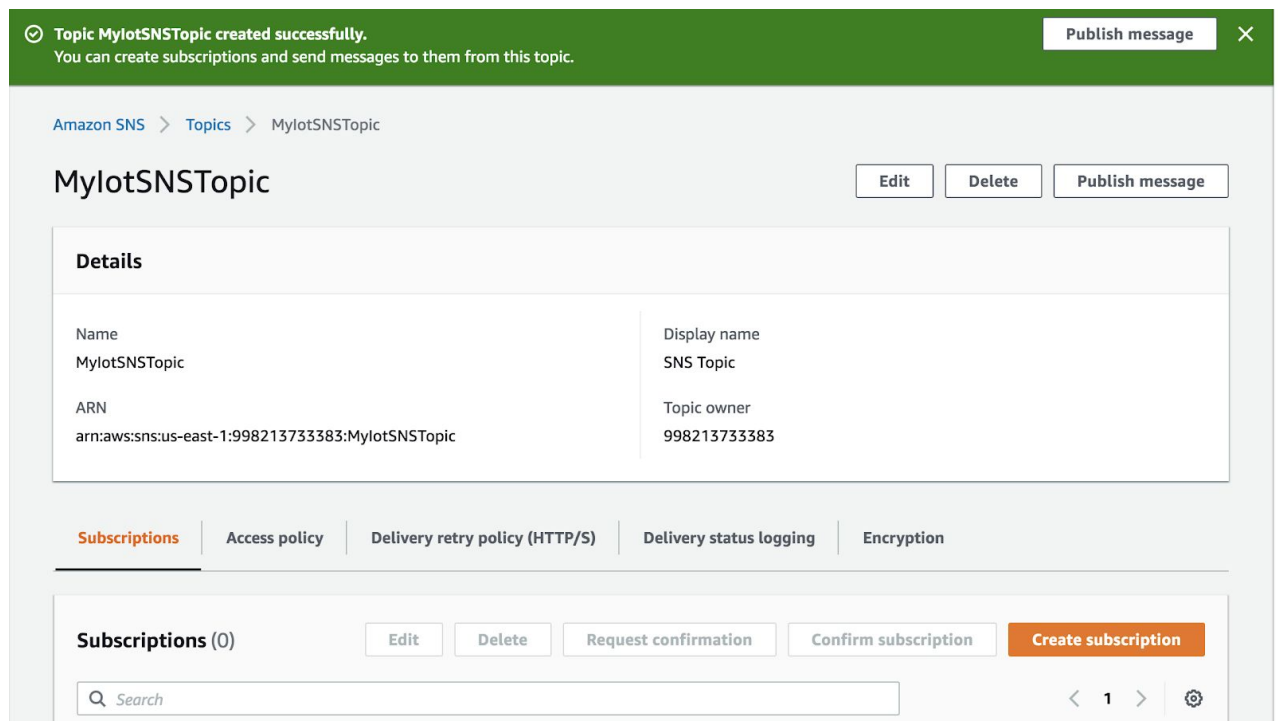


Figure 17.1: Creating a Subscription

Subscription to MylotSNSTopic created successfully.  
The ARN of the subscription is arn:aws:sns:us-east-1:998213733383:MylotSNSTopic:fa4b0216-c3a7-429d-a048-f027419e5548.

Amazon SNS > Topics > MylotSNSTopic > Subscription: fa4b0216-c3a7-429d-a048-f027419e5548

Subscription: fa4b0216-c3a7-429d-a048-f027419e5548

EditDelete

Details

ARN  
arn:aws:sns:us-east-1:998213733383:MylotSNSTopic:fa4b0216-c3a7-429d-a048-f027419e5548

Endpoint  
+13478414873

Topic  
MylotSNSTopic

Status  
Confirmed

Protocol  
SMS

Subscription filter policy

Figure 17.2: Successful creation of the Subscription on one of our device with SMS sent to the phone

my/topicExportClearPause

Publish

Specify a topic and a message to publish with a QoS of 0.

my/topic

Publish to topic

```
1 {
2
3   "default": "Hello, from AWS IoT console",
4   "message": "Hello, from AWS IoT console"
5 }
```

my/topicMay 12, 2019 2:39:06 PM -0400ExportHide

```
{
  "default": "Hello, from AWS IoT console",
  "message": "Hello, from AWS IoT console"
}
```

Figure 18: A dummy message to check if the subscription works

15



## 5.6. Creating the Lambda Function

Role creation might take a few minutes. The new role will be scoped to the current function. To use it with other functions, you can modify it in the IAM console.

**Role name**  
Enter a name for your new role.  
  
Use only letters, numbers, hyphens, or underscores with no spaces.

**Policy templates** [Info](#)  
Choose one or more policy templates.  

Amazon SNS publish policy X

SNS

**Lambda function code**  
Code is preconfigured by the chosen blueprint. You can configure it after you create the function.

**Runtime**  
Python 3.7

```
1 import json
```

Figure 19: Creating the Lambda function for the device subscribed

aws Services Resource Groups Vartika

🟢 Congratulations! Your Lambda function "myLambdaFunction" 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.

Lambda > Functions > myLambdaFunction ARN - arn:aws:lambda:us-east-1:998213733383:function:myLambdaFunction

**myLambdaFunction** Throttle Qualifiers Actions Select a test event Test Save

Configuration Monitoring

▼ Designer

**Add triggers**  
Choose a trigger from the list below to add it to your function.

- API Gateway
- AWS IoT
- Alexa Skills Kit
- Alexa Smart Home
- Application Load Balancer

myLambdaFunction Layers (0)

Add triggers from the list on the left

- Amazon CloudWatch Logs
- Amazon SNS

Resources that the function's role has access to

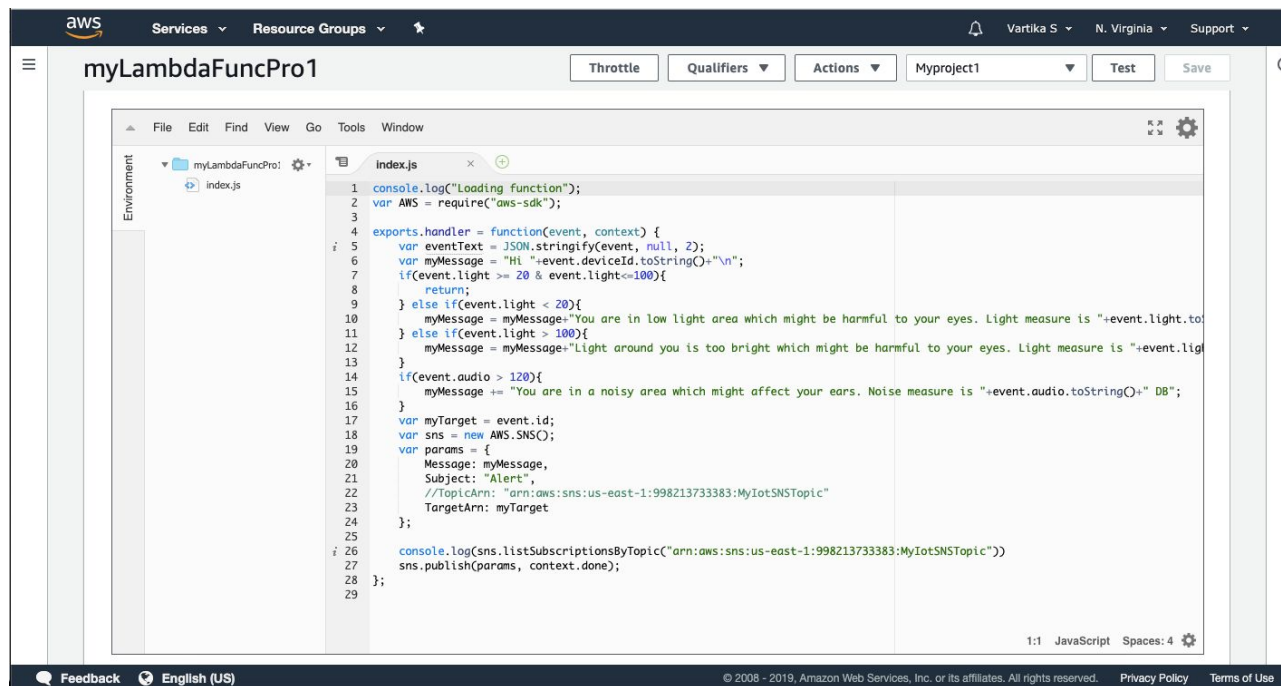
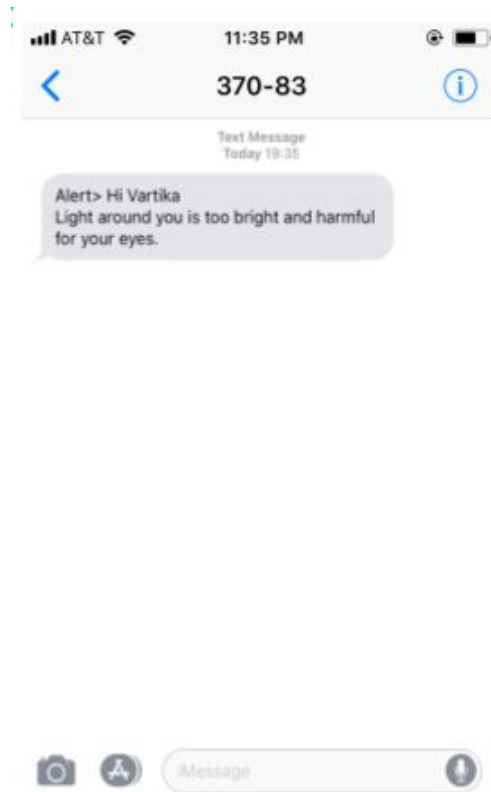


Figure 20: Lambda function definition for our application

## 6. Results (Output)

The IoT Application successfully alerts the user via Amazon Web Service (AWS) whenever the light reading goes below or above a threshold it sends an alert via SMS. In the case of our application, we received a SMS to our mobile device.



*Figure 21: SMS notification Screenshot*

## 7. Future enhancements

This project showcases the use of a simple Light and Sound alerting system which can be used by Hospitals and Health care providers services to make sure that the patient is inhabiting in a better and supporting environment for better recovery. This application can further be extended into Temperature sensing and the duration of which a person has been subjected to nonconductive lighting and sound levels.

From a technology standpoint, the application can be extended incorporating big data technologies and data analytics. The use of the application would propagate the demand of database to keep user information. Increase in the size of the database can be further be utilized for data mining and analytics which can help generate report and insights for user behavior. Like the duration for which a patient was in low or high-intensity light and what could be the possible negative impact of the user's health. Same in the case of sound levels, the application can predict the possible effects on hearing ability of a patient.

## 8. Reference

[1] <https://1sheeld.com/mqtt-protocol/>

[2] <https://searchaws.techtarget.com/definition/Amazon-EC2-instances>

[3] <https://aws.amazon.com/iot-core/>