**Chapter 1 : INTRODUCTION**

* 1. **Project Overview**

Amazon Web Services (AWS) is a secure cloud services platform, offering compute power, database storage, content delivery and other functionality to help businesses scale and grow.

Amazon Elastic Compute Cloud (Amazon EC2) provides scalable computing capacity in the Amazon Web Services (AWS) cloud. Using Amazon EC2 eliminates need to invest in hardware up front, so you can develop and deploy applications faster. We can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage.

The businesses running on Cloud infrastructure are increasing day by day and so is the use of EC2 instances. Thus the challenge of managing these virtual servers arises. It becomes a troublesome task to log in the AWS console every time one needs to start/stop/launch a server. So Voice commands to manage servers can come handy in saving time.

Alexa is a virtual assistant developed by Amazon which is used in Echo dot and Echo smart speakers. Alexa provides a set of built-in capabilities, referred to as skills. One can also customize a skill by being an Alexa developer. Alexa developer console enable us to build a personal skill.

In this project we have used the Alexa customized skill which will trigger the AWS Lambda function which in turn will control the EC2 resources.

Thus the user will be able to control/manage the AWS instances using Alexa voice commands.

* 1. **Scope**

Voice commands can be of utmost use in saving time. In the present world where use of could services are growing day by day the ease of management has become a necessity. A cloudOps engineer or developer using AWS may have many virtual servers running and it can be difficult to manage these servers. Thus Alexa voice commands can prove as an efficient assistant in managing the EC2 instances.

* 1. **Objective**
* To start a tagged server/instance using voice commands.
* To stop a tagged server/instance using voice commands.
* To launch a new virtual server/instance and get the instance ID of the started server.
* To change or view the region of AWS of the user.

**Chapter 2 : SYSTEM ANALYSIS**

**2.1 User Characteristics**

The targeted users of this project are the :

* Cloud operations engineer
* Cloud developers
* Cloud practitioners

The user has to give voice commands to Alexa such as “Alexa start server” which will start a tagged server. There are pre-defined utterances that a user can say to manage virtual server such as:-

* For starting the tagged server/instance:-
  + “Start all servers”
  + “good morning”
* For stopping all tagged servers/instances:-
  + “Stop all servers”
  + “good bye”
  + “over and out”
* For knowing the current region:-
  + “current region”
  + “which is my current region”
  + “My current region is”
* For knowing the number of running instances:-
  + “number of running instances”
  + “how many instances are running”
* For launching a new instance:-
  + “start new instance”
  + “Launch new instance”

These utterances are connected to a specific intent. Intents represent the users overall request. So when any utterances are invoked or said by the user, then that utterances invoke the intent which in turn executes a specific task.

**2.2 Tools and Technology Used**

The software tools and technologies used during development of this project are

* AWS Lambda function
* Alexa Skills kit
* Alexa developer console

The hardware requirements for this project are (any one of the following):-

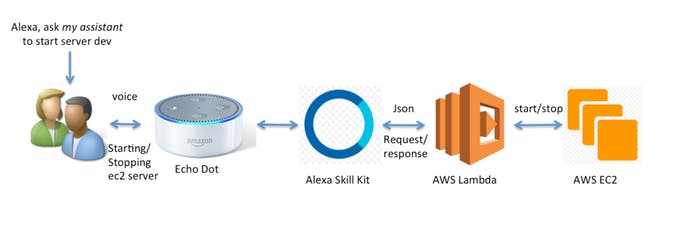
* Amazon Echo
* Amazon Echo dot
* Amazon Echo plus

**2.2.2 How it works**

* First we need to build a new Alexa custom skill which will have utterances and intents.
* Add the Lambda ARN (Amazon Resource Name) to the skill which will act as an endpoint to our AWS account.
* Develop the lambda function with the python code which will control the EC2 instances. Python code will consist of Boto script (AWS SDK for python).
* Next we paste the Alexa skill ID in the Lambda function thus connecting the Alexa skill and Lambda function.
* Now whenever a valid utterance is made by the user , it will invoke the specific intent which in turn will trigger the lambda function to carry out a specific task.

**Chapter 3 : SYSTEM DESIGN**

**3.1 Flow of project**



* User asks Alexa to start/stop/launch a virtual server/instance by saying the pre-defined utterances mentioned in the customized Alexa skill.
* The Alexa enabled device (Echo, Echo dot) matches the utterances with the customized skill.
* On matching of utterances the Alexa skill will invoke the corresponding intent.
* The intent will trigger the Lambda function.
* The Lambda function will execute a specific task on EC2 as per the code in the Lambda function.

**3.2 Major Functionality (Data Flow Diagram)**

Start

Blank response

Please say a skill name

User invokes Alexa

Skill invoked?

NO

Yes

Intent invoked?

No

Yes

Get/set Region

Stop Instance

Launch Instance

Start Instance

Trigger Lambda Function

EC2 Operations

END

**Chapter 4 : IMPLEMENTATION**

**4.1 Implementation Environment**

* Python 3.6
* AWS Lambda function

**4.2 Module specification**

* Alexa skill
* Lambda

**4.3 Coding Standards**

Following conventions should be used for coding:

1. Use of pithy comments

When other engineers attempt to discern how your code works, they’ll likely turn to your comments. Thus, you should know that these comments form an accurate description of code.

1. Assign easy-to-understand names to variable and function:

You should use names to indicate functionality. The name “exit\_button” would be more appropriate than “b1”.

1. Apply rules uniformly

Simply put, clean code is consistent code.

1. Don’t make your code unnecessarily complex

In other words, a lengthy line of code that updates multiple variable is more difficult to understand than a simple line of code that implements the same objective.

1. Be explicit

So that other developers may be able to quickly discern what each line of your code does, you should eschew syntax that obscures meaning.

**Code sample**

def on\_intent(intent\_request, session):

"""

Called when the user specifies an intent for this skill

:param intent\_request:

:param session:

:return:

"""

print("on\_intent requestId=" + intent\_request['requestId'] +

", sessionId=" + session['sessionId'])

intent = intent\_request['intent']

intent\_name = intent\_request['intent']['name']

# Dispatch to your skill's intent handlers

if intent\_name == "StartTaggedInstances":

return start\_tagged\_instances(intent, session)

elif intent\_name == "LaunchInstance":

return launch\_instances(intent, session)

elif intent\_name == "StopTaggedInstances":

return stop\_tagged\_instances(intent, session)

elif intent\_name == "GetRunningTaggedInstances":

return get\_running\_tagged\_instances(intent, session)

elif intent\_name == "GetCurrentRegion":

return get\_current\_region(intent, session)

elif intent\_name == "SetCurrentRegion":

return set\_current\_region(intent, session)

elif intent\_name == "AMAZON.HelpIntent":

return get\_skill\_help\_response()

# that is only for fun without any reason

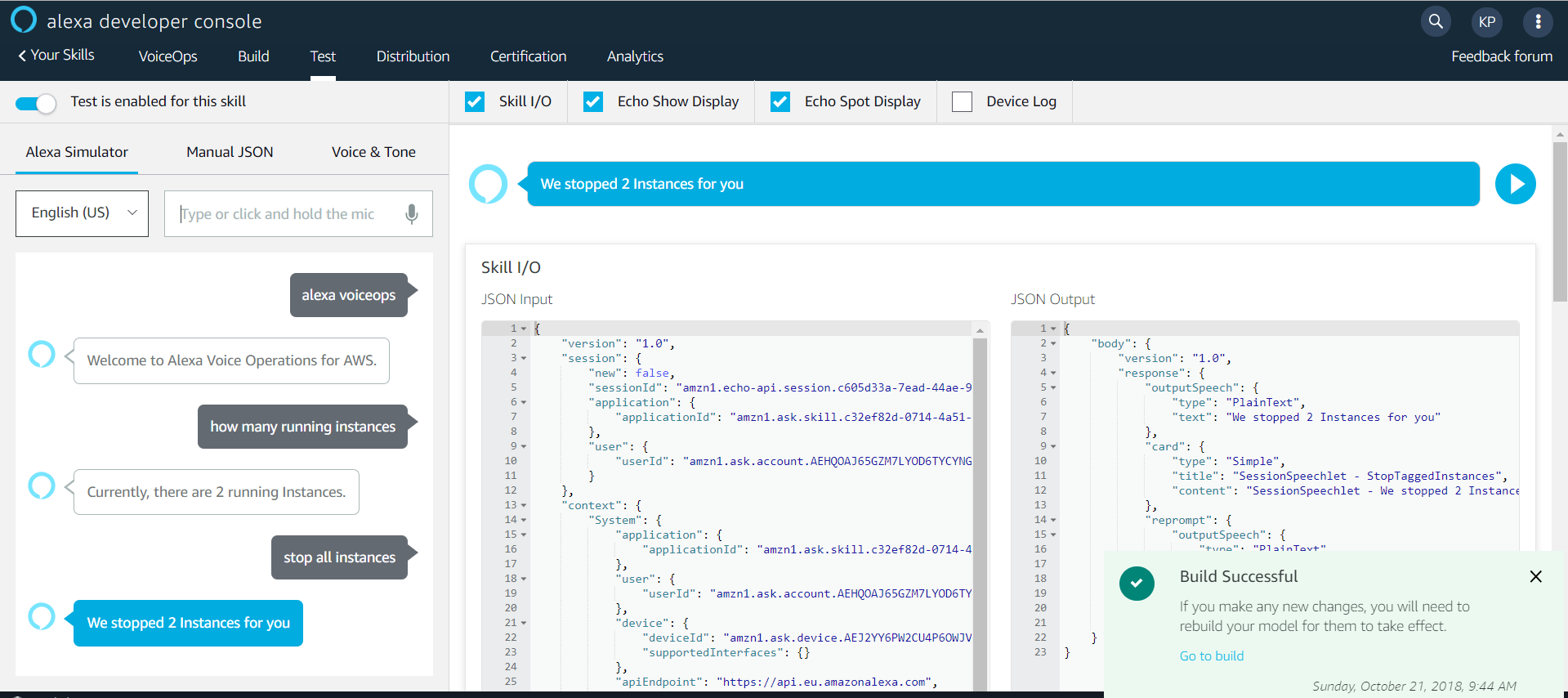
elif intent\_name == "GetSomeFun":

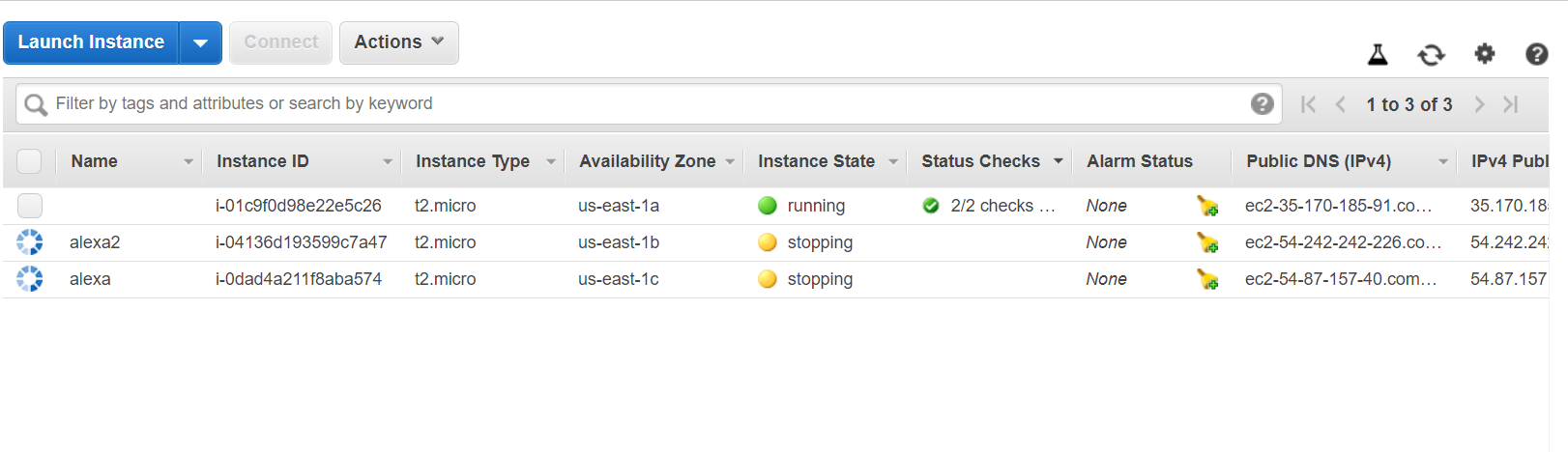
return get\_fun(intent, session)

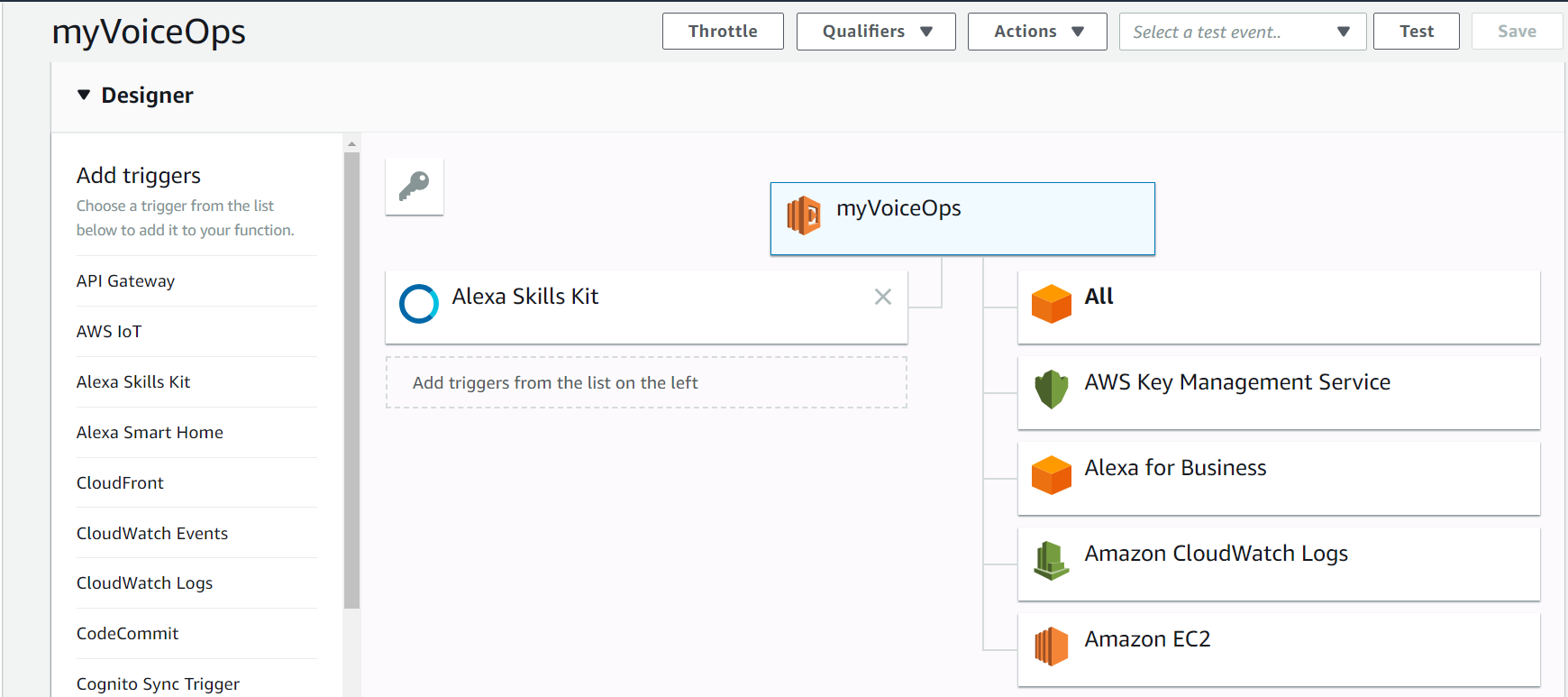
else:

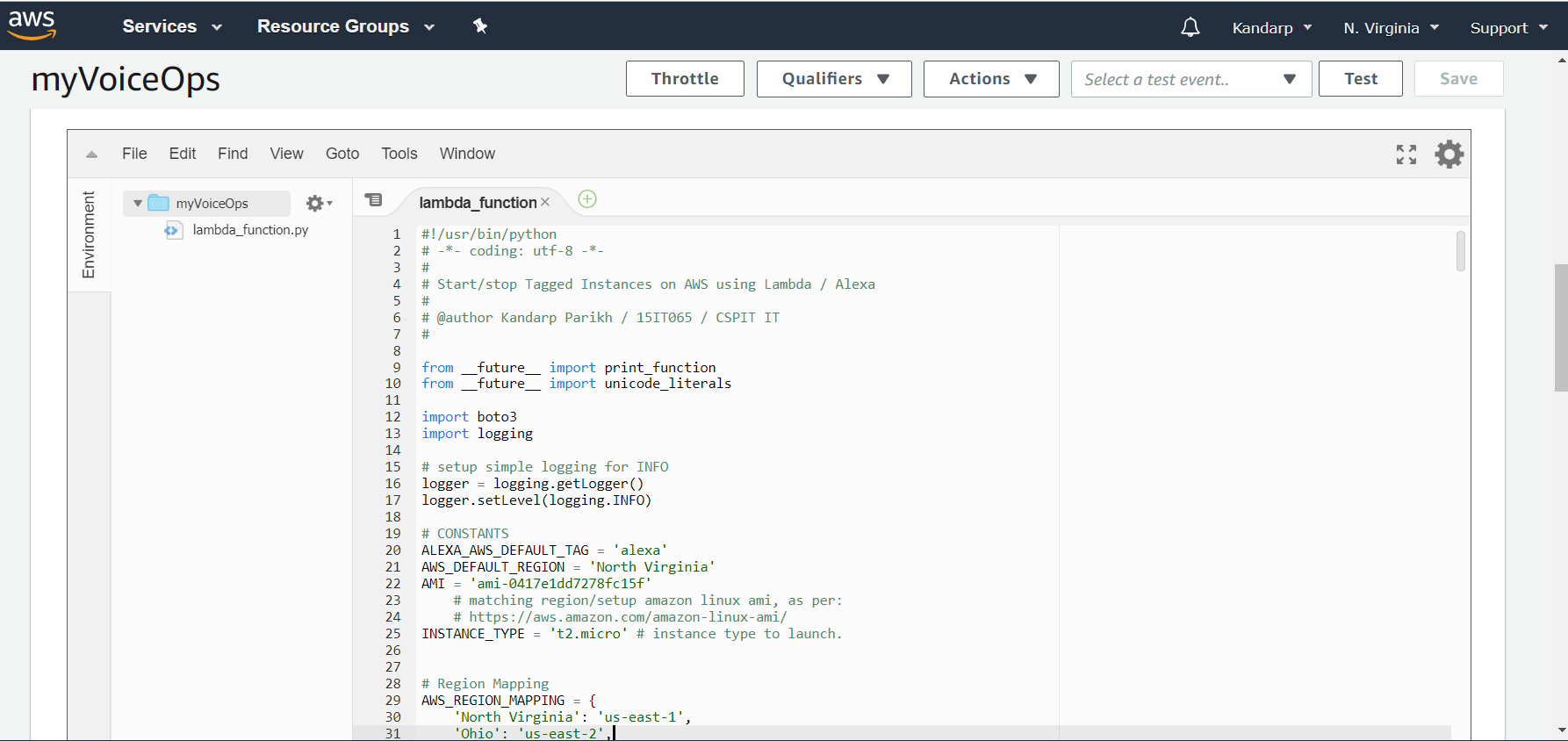
raise ValueError("Invalid intent")

**4.4 Snapshots of Project**

****

****

****

****

**Chapter 5 : CONSTRAINTS AND FUTURE ENHANCEMENTS**

* 1. **Constraints**
* Only the tagged servers can be accessed/managed.
* Tags cannot be assigned to the virtual servers/instances when they are created or launched.
* Instances having the same tag can only be accessed and bulk start and stop can only be done.
  1. **Future Enhancements**
* Instances with specific tags can be accessed rather than accessing instances of the same tag.
* To be able to backup any instance using voice commands.

**Chapter 6 : CONCLUSION**

There was certainly a learning outcome to this project. I have gained a reasonable amount of knowledge about AWS, Lambda function, EC2 and Alexa skills which will be much helpful in the coming future. Using all these technologies I developed the project “AWS Voice Assistant” which will be helpful to a large number of users.

**REFERENCES**

* <https://www.hackster.io/vikrame/alexa-my-new-aws-assistant-fce779>
* <https://hackr.io/tutorials/learn-alexa-skills-kit-ask>
* <https://linuxacademy.com/howtoguides/posts/show/topic/14209-automating-aws-with-python-and-boto3>
* <https://docs.aws.amazon.com/lambda/latest/dg/python-programming-model-handler-types.html>
* <https://boto3.amazonaws.com/v1/documentation/api/latest/index.html>