**Vanity Number Generator | VoiceFoundry**

User Manual

Version 1.0, Dec 2021

Version history

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version number** (eg. V1.0) | **Modified by** (Name) | **Modifications made** | **Date Modified** | **Status** (Draft, Revised, Final) |
| V1.0 | Devika Anil | Prepared report | 21 Dec 2021 | Final |
|  |  |  |  |  |
|  |  |  |  |  |

Contents

[1. Introduction 3](#_Toc91005556)

[2. Installation Guidelines 4](#_Toc91005557)

[3. Navigational Flow 16](#_Toc91005558)

[4. Software Requirements 18](#_Toc91005559)

[5. Software Installation Guidelines 18](#_Toc91005560)

[6. AWS Installation Guidelines 18](#_Toc91005561)

# Introduction

Vanity Number Generator project is part of mid-level assignment submitted to VoiceFoundry.

Objective:

-To create a lambda function that converts phone numbers to vanity numbers and save the best five resulting vanity numbers and the caller’s number in a DynamoDB table.

-Create an Amazon Connect contact flow that looks at the caller’s phone number and say the 3 vanity possibilities that come back from the Lambda function.

-Build a deployment package with AWS SAM, AWS CDK or CloudFormation to allow a user, or assignment reviewer to deploy this solution into their own AWS Account/Amazon connect instance.

**Prerequisites:**

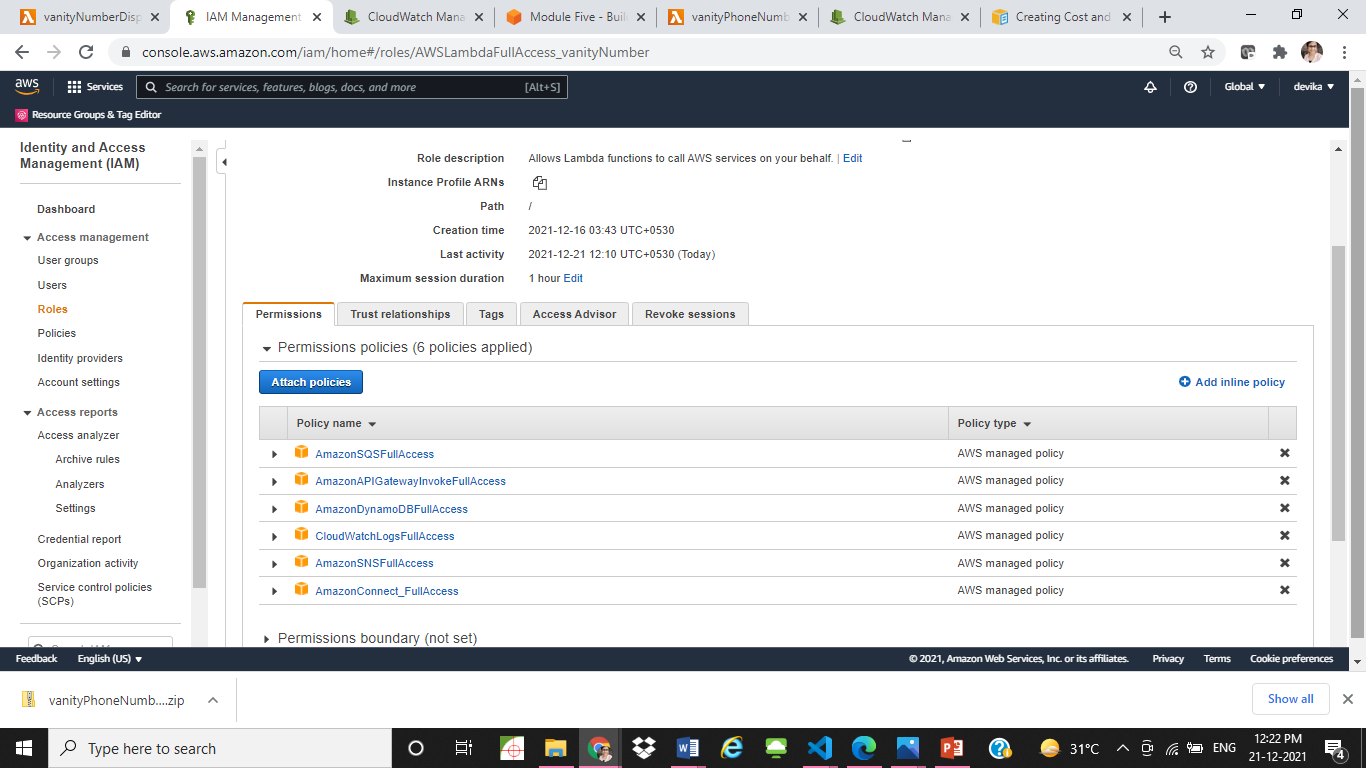
* An AWS account
* Good technical knowledge of AWS Consoles
* Required permission for user to create and run the following AWS resources:
  + Lambda
  + DynamoDB
  + CloudWatch Logs
  + SQS
  + SNS
  + API Gateway
  + Amazon Connect
  + Amazon Amplify

# Installation Guidelines

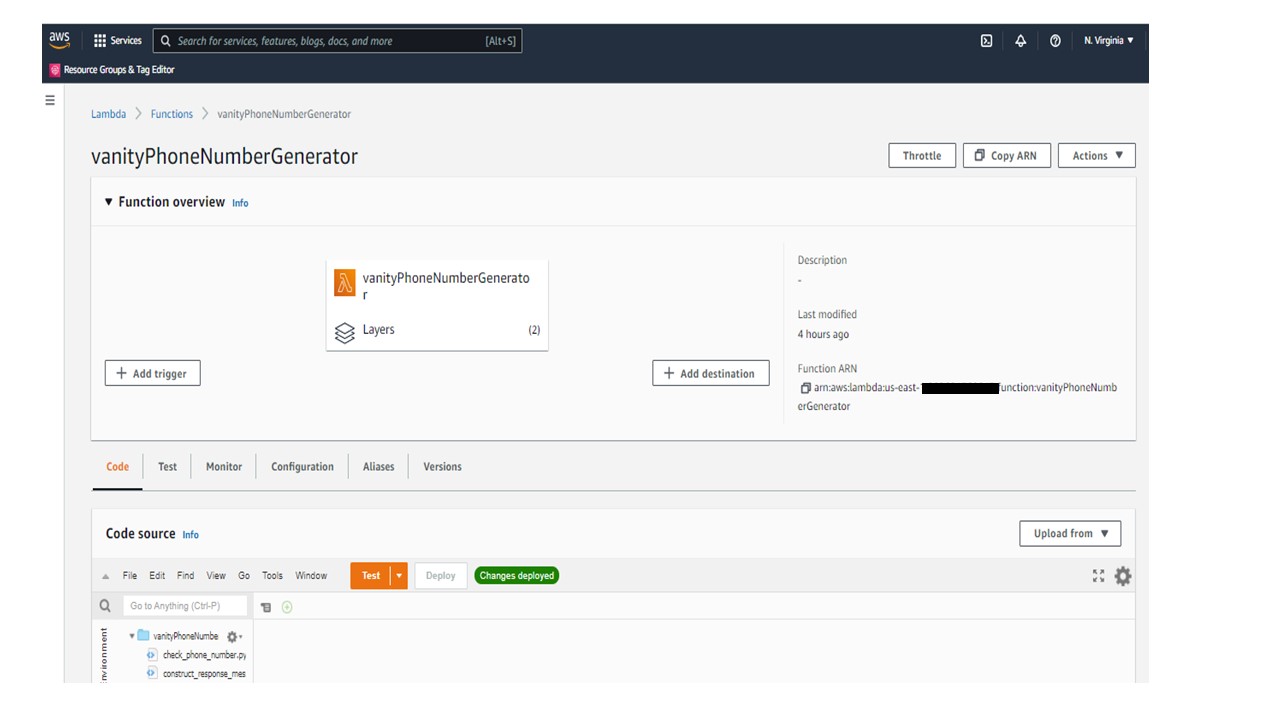
**Amazon Connect Lambda Integration Steps:**

**Create and Configure Lambda function**

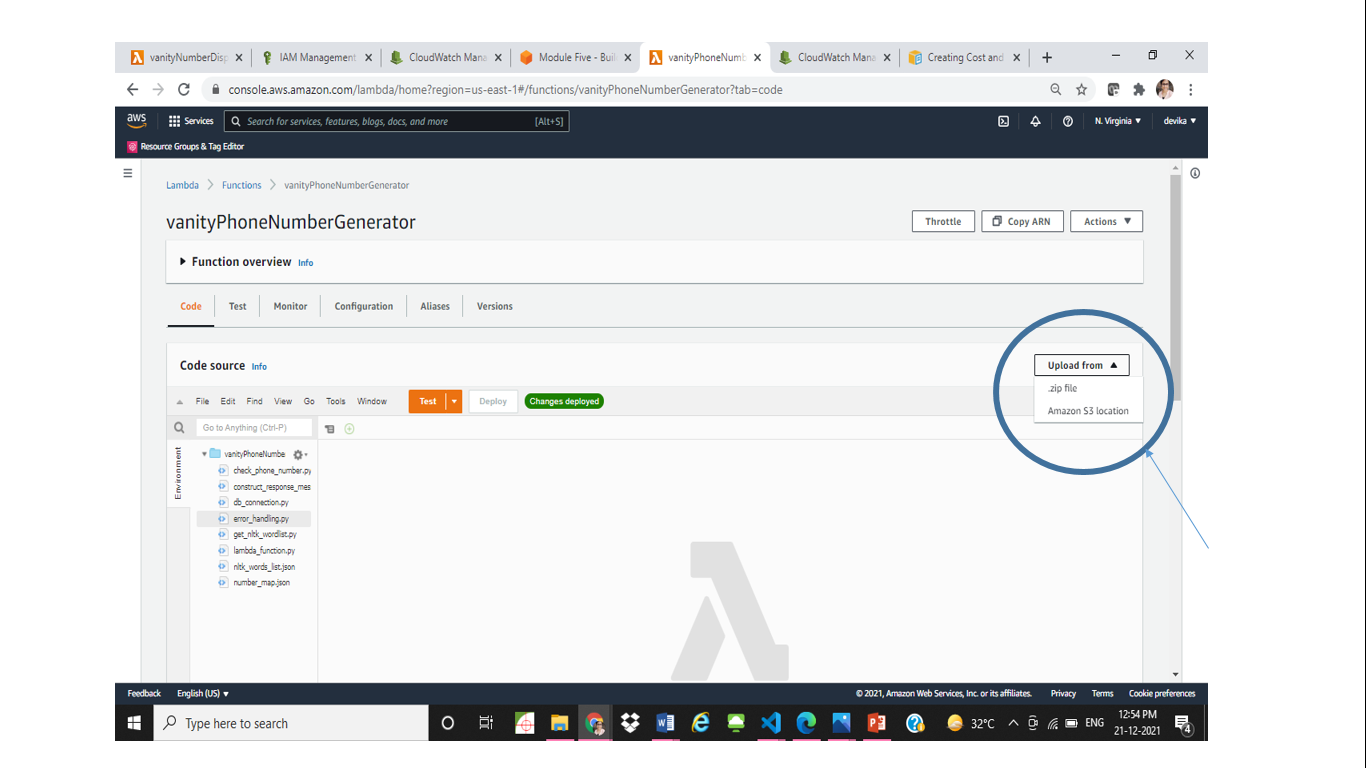
* Having an AWS account, create an IAM role with Lambda, DynamoDB, CloudWatch, SQS, SNS and API Gateway full access. Note IAM Role ARN for creating Lambda function.
  + Refer: [Creating IAM roles](https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_create.html)
  + Note the region (us-east-1 here) for IAM and create L



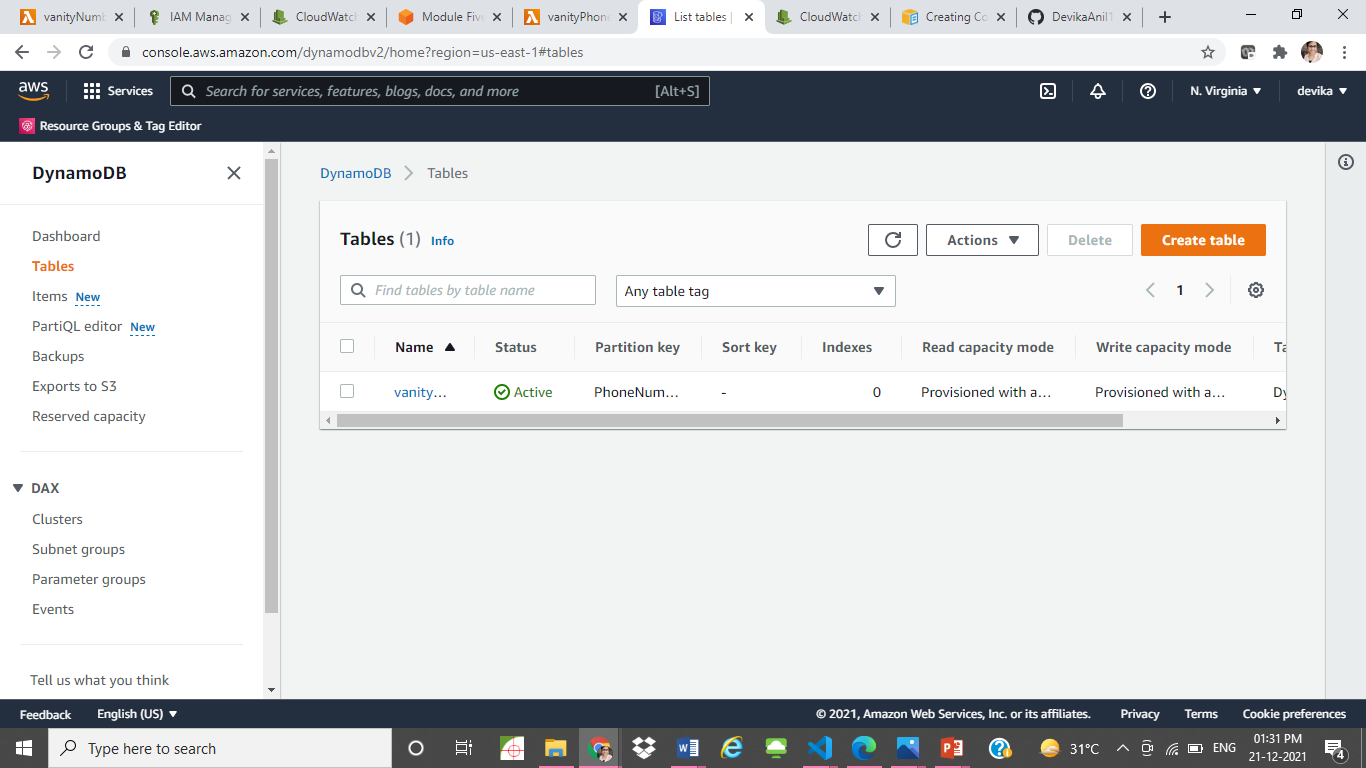
* Create a Lambda function under the IAM role created (Runtime: Python 3.8)
  + Refer: [Create a Lambda function](https://docs.aws.amazon.com/lambda/latest/dg/getting-started-create-function.html)



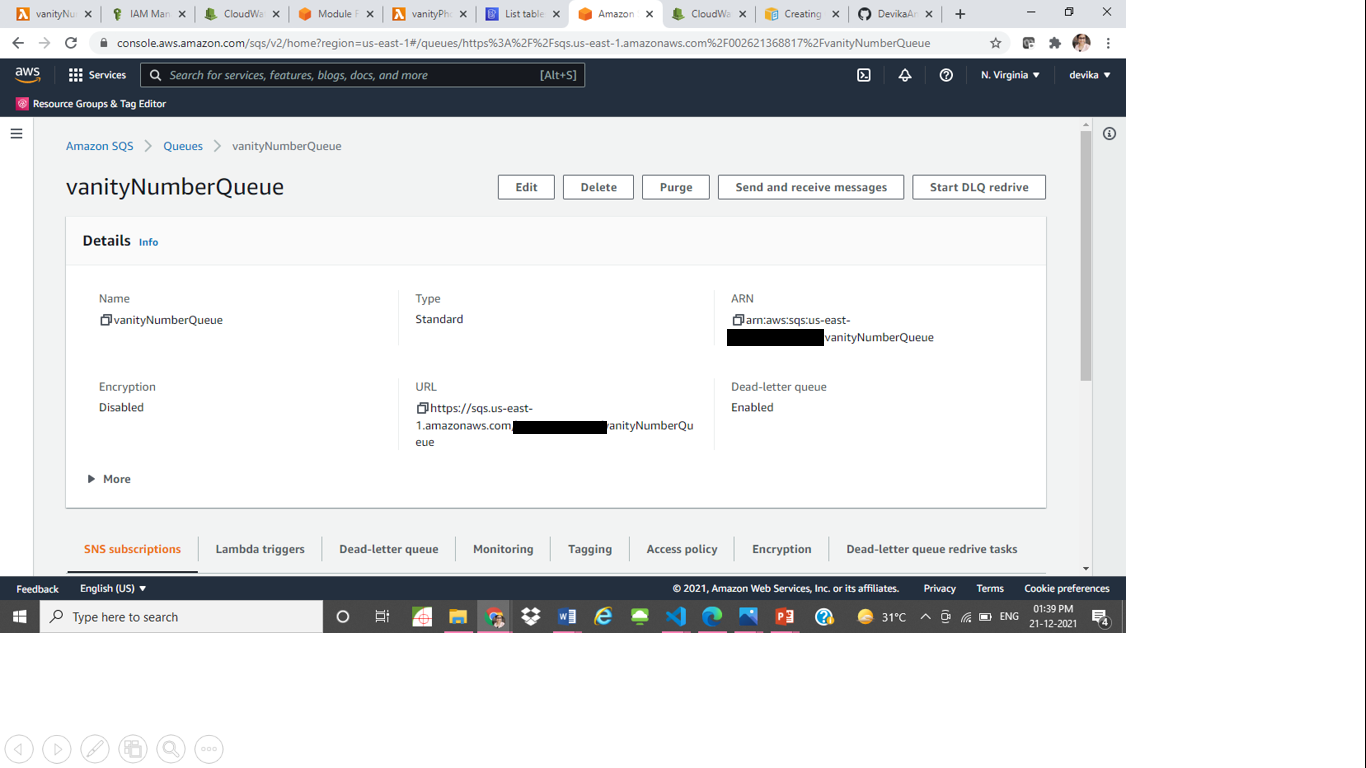
* Create layer to import NLTK corpuses and add the layer to Lambda function.
  + Refer: [Lambda invocation layers](https://docs.aws.amazon.com/lambda/latest/dg/invocation-layers.html)
* Upload the deployment package from Deployment/vanityPhoneNumberGenerator.zip to the Lambda function

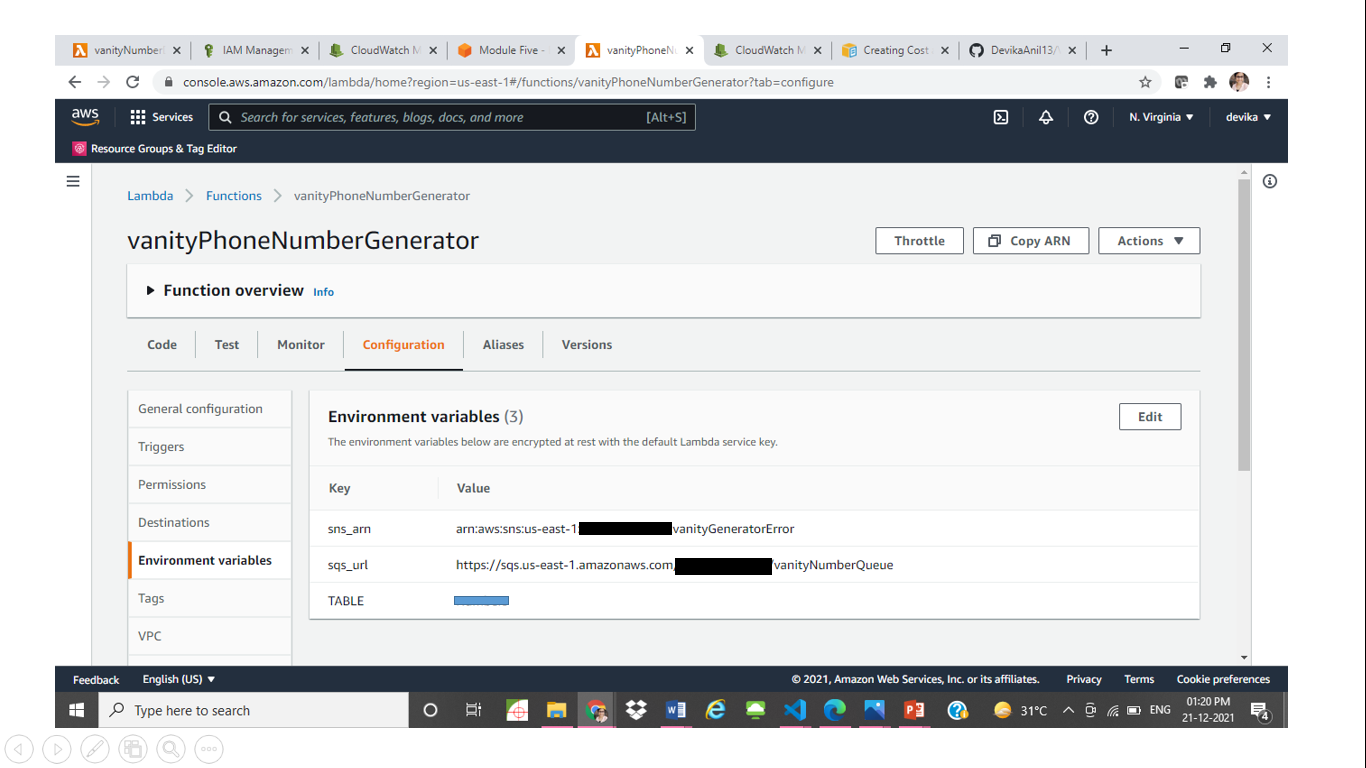


* Create DynamoDB table with Key “PhoneNumber”. Note the DynamoDB table name. Make changes in db\_connection.py and environment variable if required.
  + Refer: [Create a dynamoDB Table](https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/getting-started-step-1.html)
  + Once the Table is created, the table will be visible Inside DynamoDB -> Tables as shown below:



* Create SQS Queue. Enable Dead-letter-Queue. Custom error messages on exception can be send to SQS DLQ.
  + Note the SQS Queue URL.
  + Refer: [Getting started with Amazon SQS](https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/sqs-getting-started.html)



* Create SNS Topic and add a subscription with an e-mail address, confirm the subscription. Note the Topic Arn.
  + SNS Notification to e-mail can be used to notify the subscriber on any error or exceptions.
  + Refer: [Configuring Amazon SNS](https://docs.aws.amazon.com/sns/latest/dg/sns-configuring.html)
* Set Environment variables under Configuration
  + Add Key and Values for SNS Target Arn (Topic Arn), SQS Queue URL and part of DynamoDB table name. Note: Either environment variables can be added or Secrets Key Manager ([AWS Secrets Manager](https://aws.amazon.com/secrets-manager/)) can be used to store and use secret values like account information, login credentials for accessing databases etc.
* Under Configuration > General Cofiguration > Timeout, Increase timeout from 3sec to 1min
* Deploy the Lambda function with the changes made. Configure a test event with the sample json input below. Change the event[‘Details’][‘ContactData’][‘CustomerEndpoint’][‘Address’] in JSON to a sample phone number to test.

*{*

*"Details": {*

*"ContactData": {*

*"Attributes": {},*

*"Channel": "VOICE",*

*"ContactId": "4e36b945-699d-4136-80fd-1c367731d575",*

*"CustomerEndpoint": {*

*"Address": "+447xxxxxxxxx",*

*"Type": "TELEPHONE\_NUMBER"*

*},*

*"CustomerId": "None",*

*"Description": "None",*

*"InitialContactId": "4e36b945-699d-4136-80fd-1c367731d575",*

*"InitiationMethod": "INBOUND",*

*"InstanceARN": "arn:aws:connect:us-east-1:xxxxxxxxxxxx:instance/5adc6f2e-a9af-4ba5-82f6-1742a521e73d",*

*"LanguageCode": "en-US",*

*"MediaStreams": {*

*"Customer": {*

*"Audio": "None"*

*}*

*},*

*"Name": "None",*

*"PreviousContactId": "4e36b945-699d-4136-80fd-1c367731d575",*

*"Queue": "None",*

*"References": {},*

*"SystemEndpoint": {*

*"Address": "+4411439xxxx",*

*"Type": "TELEPHONE\_NUMBER"*

*}*

*},*

*"Parameters": { }*

*},*

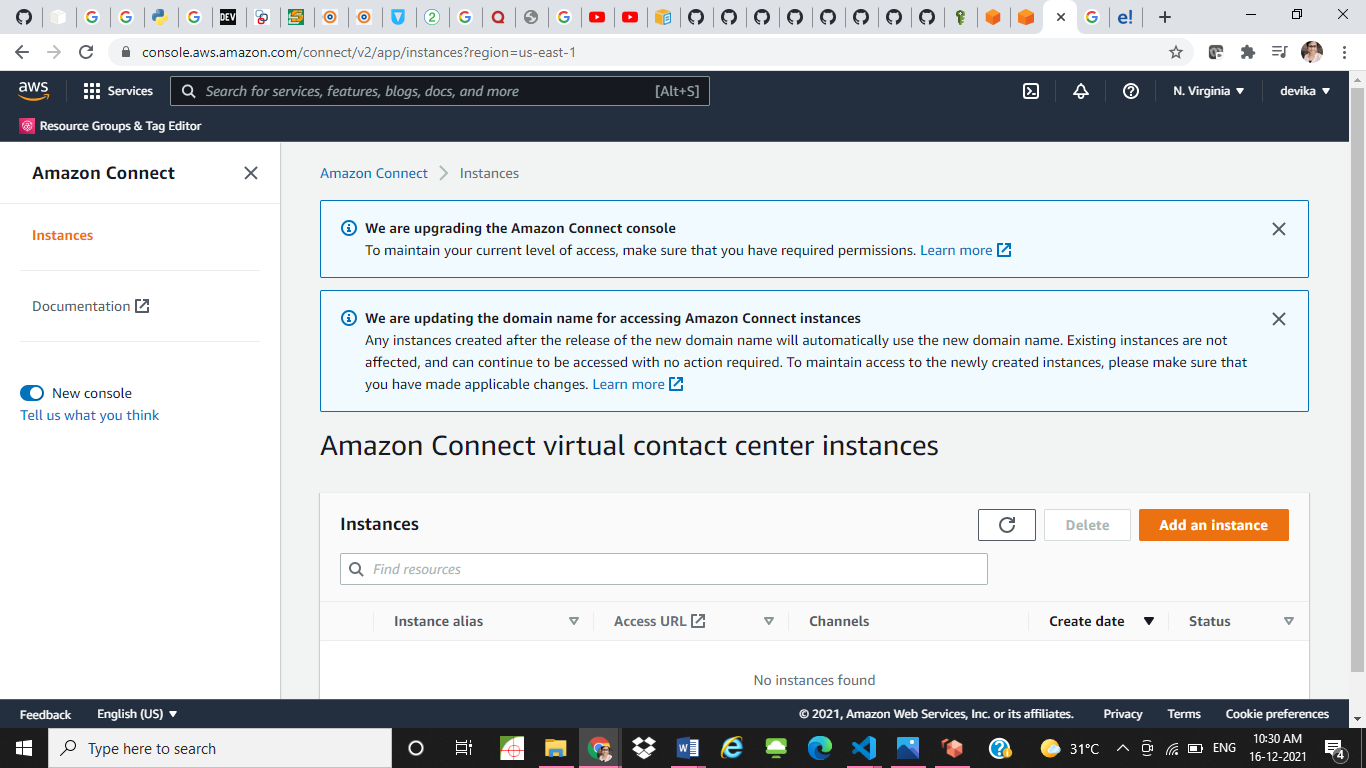
*"Name": "ContactFlowEvent"*

*}*

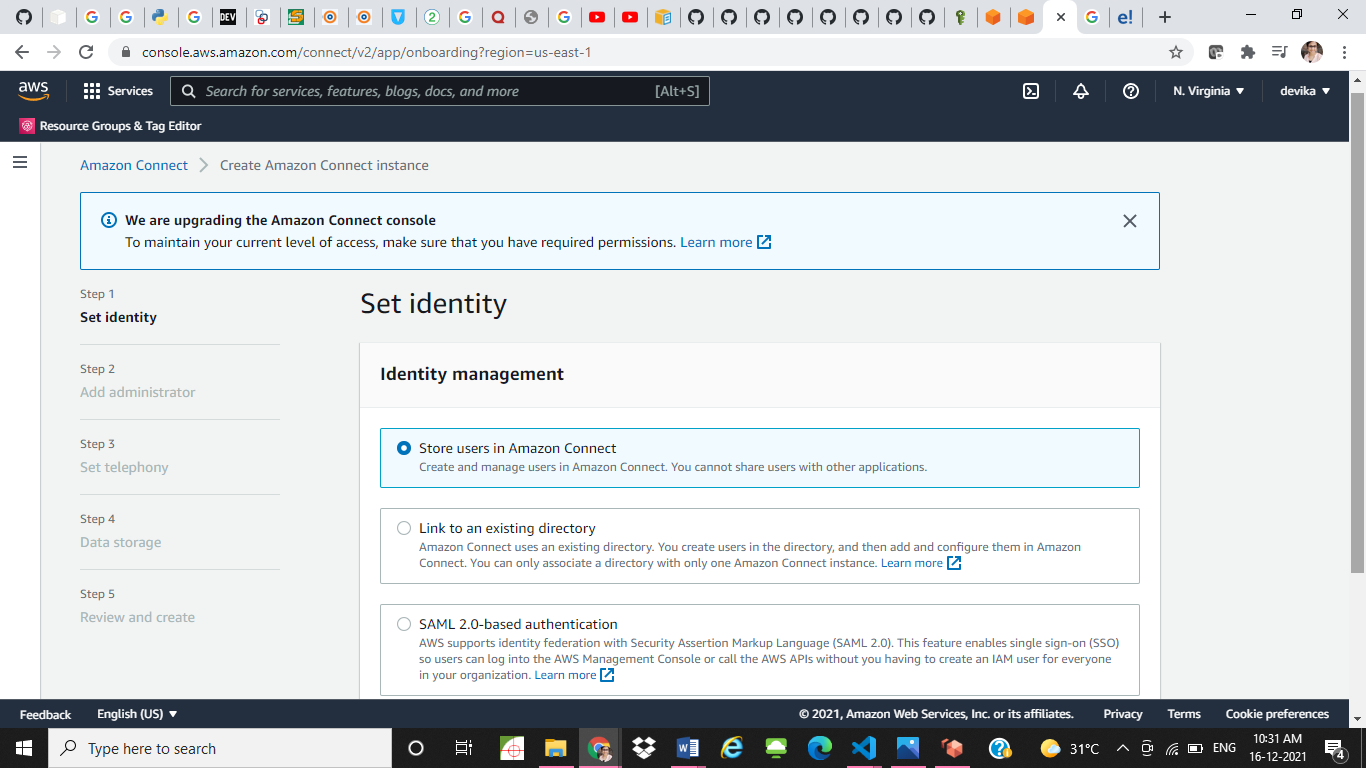
* Check the output obtained, Dynamodb table item and confirm that the Lambda is working fine. Check once again with the same input and see the data is read from dynamodb for existing data in dynamodb.
* PhoneNumber, lastCallTime and top 5 vanityNumbers generated will be send to dynamodb table.

**Creating Amazon Connect Instance**

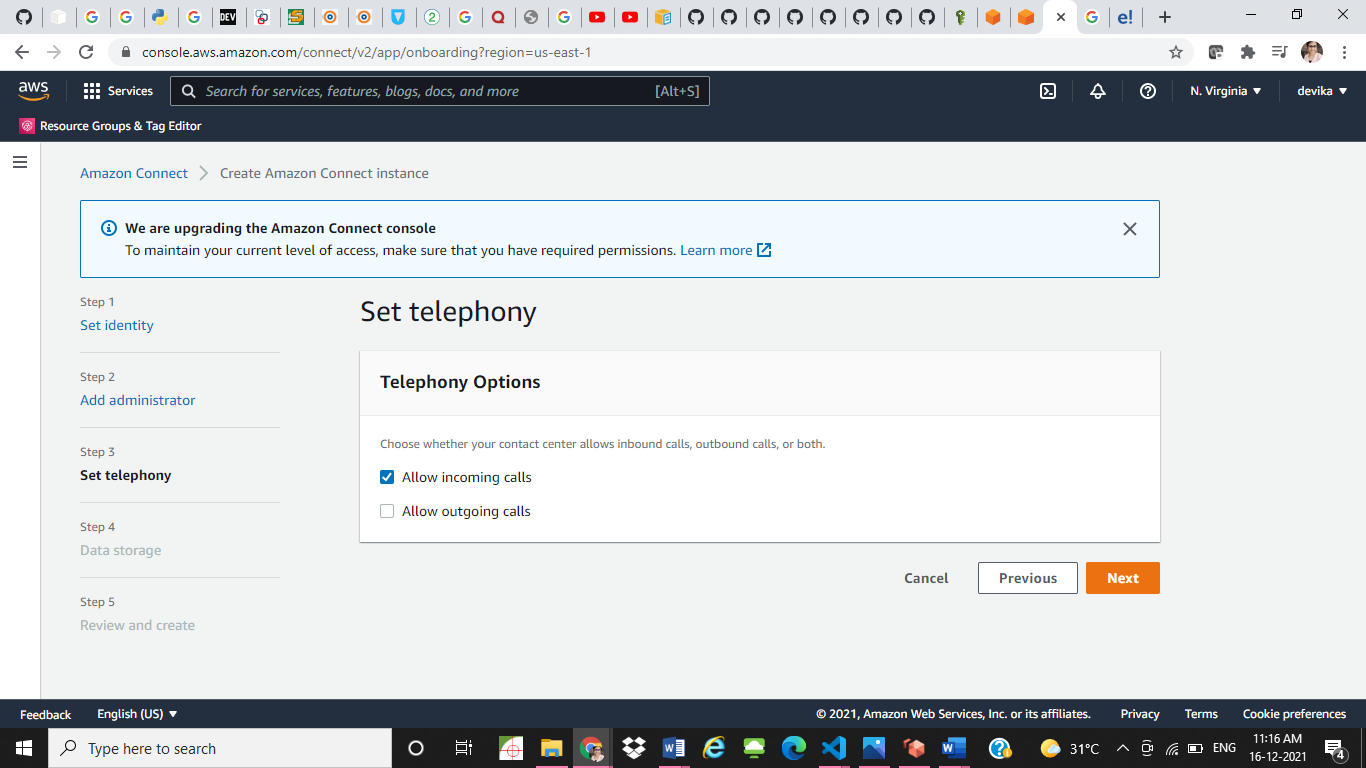
* Configure an Amazon Connect Instance.
  + Refer: [Create an Amazon Connect instance](https://docs.aws.amazon.com/connect/latest/adminguide/amazon-connect-instances.html)
  + Not all the services are available in all regions. So instance created in US-East-1. Supported regions:
    - Europe (London)
    - Asia Pacific (Singapore)
    - Asia Pacific (Sydney)
    - Europe (Frankfurt)
    - Asia Pacific (Tokyo)
    - US East (N. Virginia)
    - US West (Oregon)
  + Add an instance



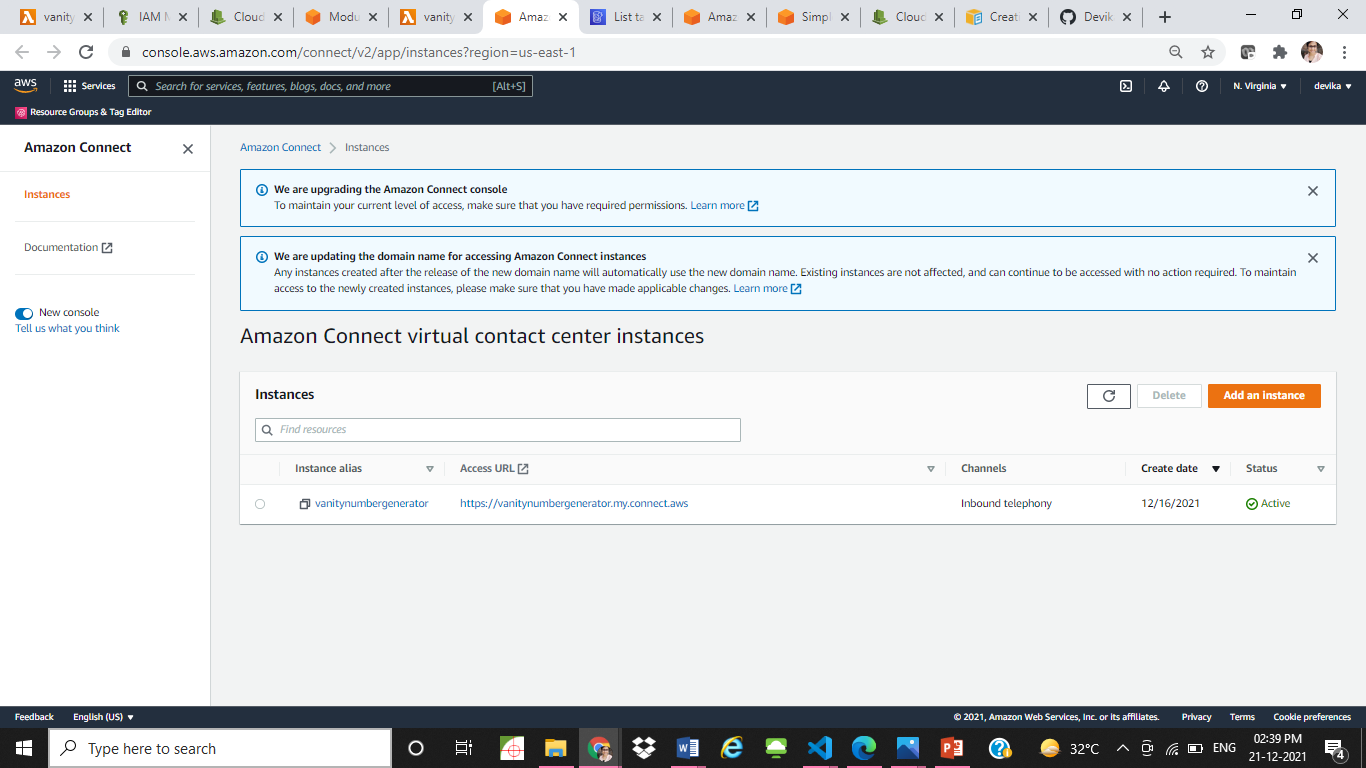
* + Set identity management to ‘Store users in amazon connect’



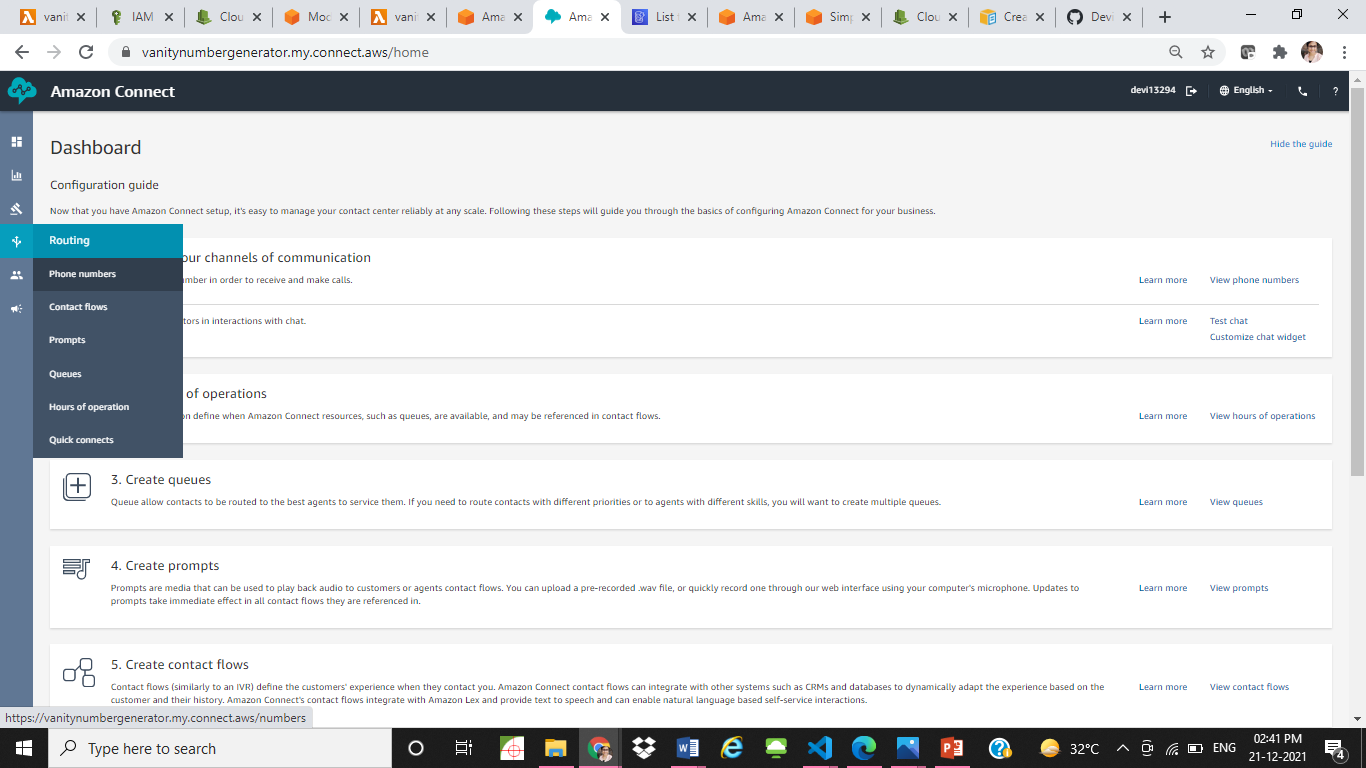
* + Create a custom URL and Specify an administrator
  + In Telephony options, allow Incoming calls. (No outgoing calls needed for this use case.)



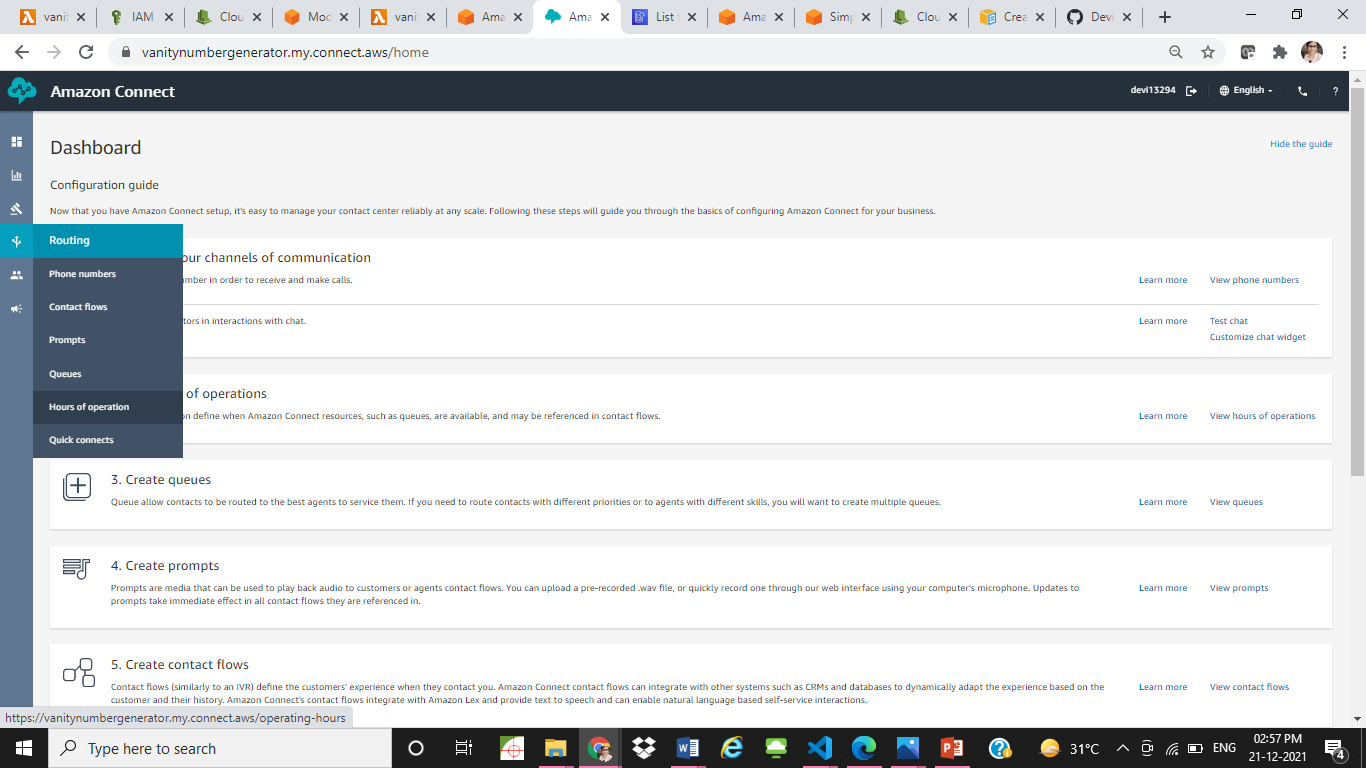
* + Add data storage options and create Amazon connect instance.



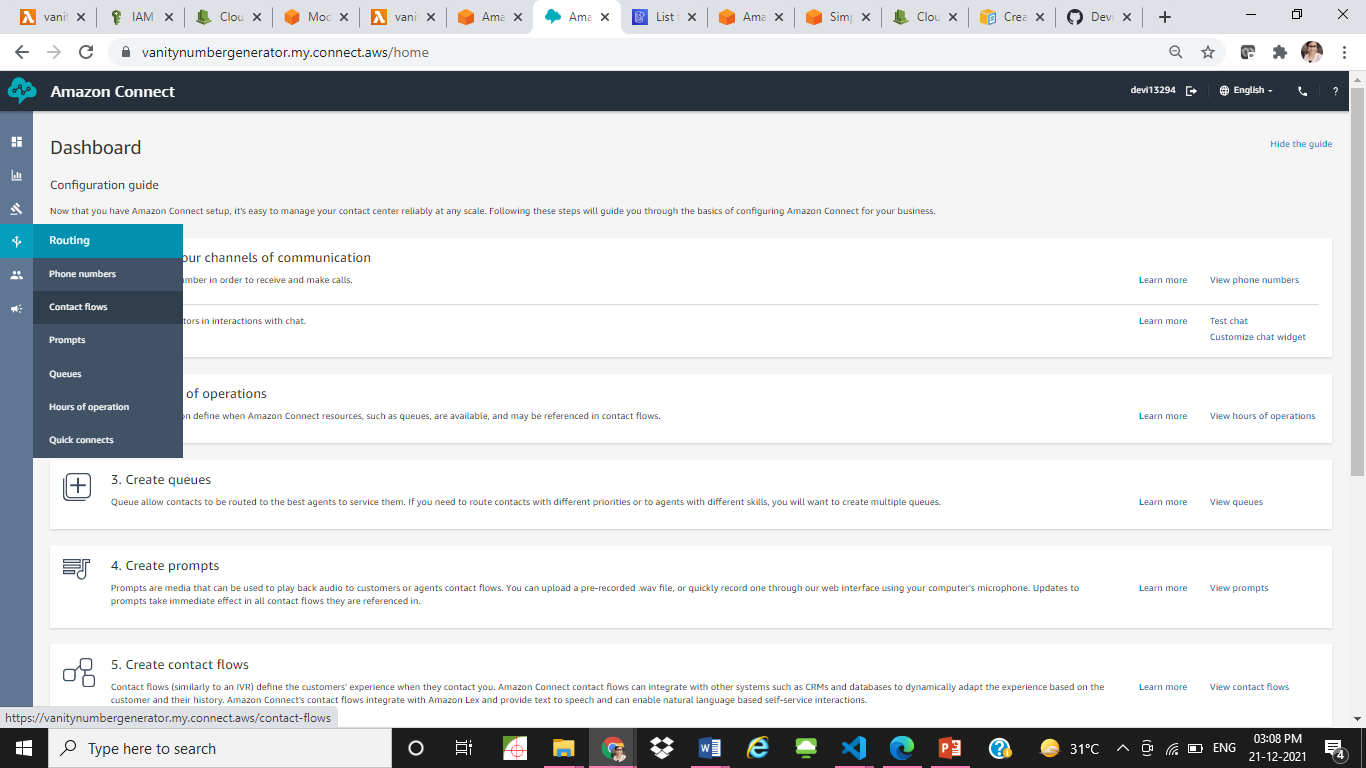
* From the dashboard > Routing > Phone numbers, claim a phone number. This phone number can be dialed to connect to Amazon Connect once all connections are established for Vanity Number Generator. For this project, UK region is selected and claimed a UK phone number.



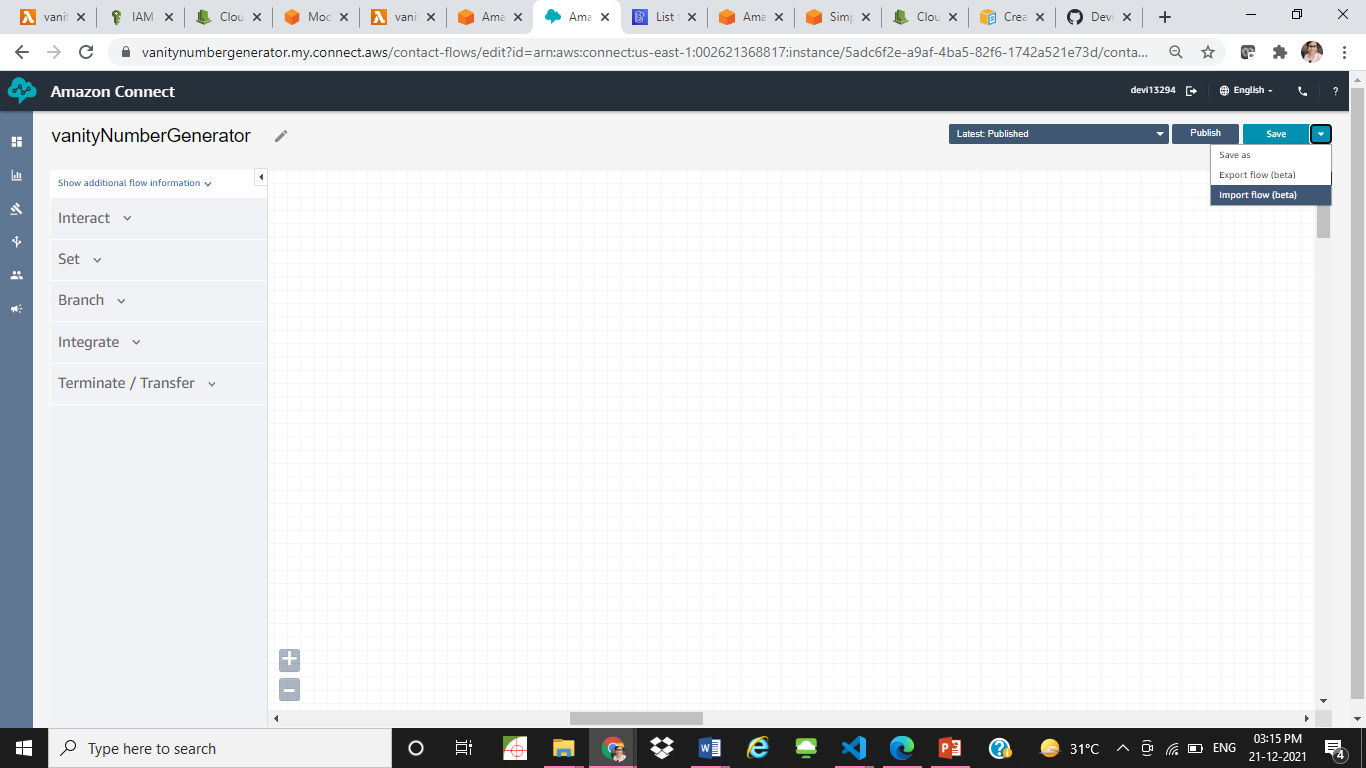
* Set hours of operation from dashboard > Routing > Hours of operation
  + Can be set as basic hours of operation (Always open hours) or specific time can be set.



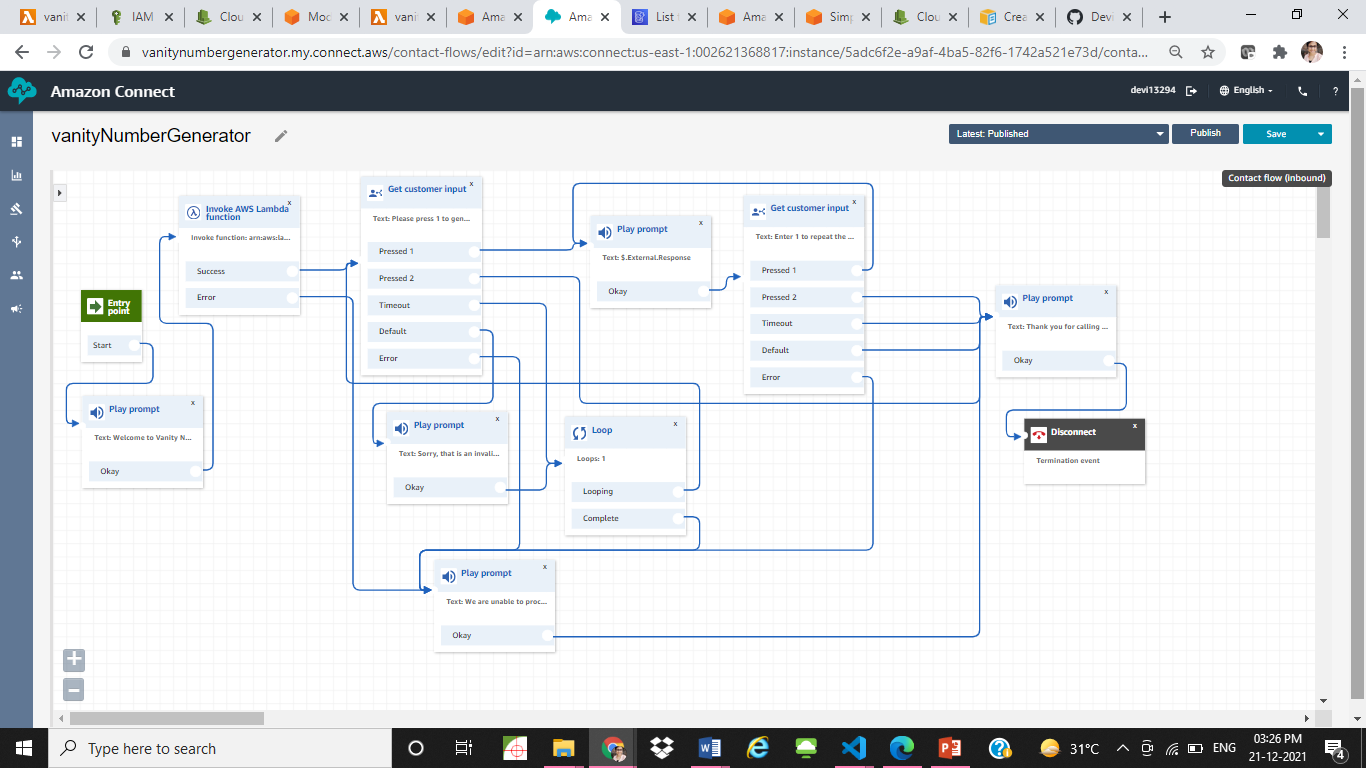
* Create Contact flow from Dashboard > Routing > Contact flows



* + Click Import beta file from dropdown at the top right corner and import contact flow from Amazon Connect folder.



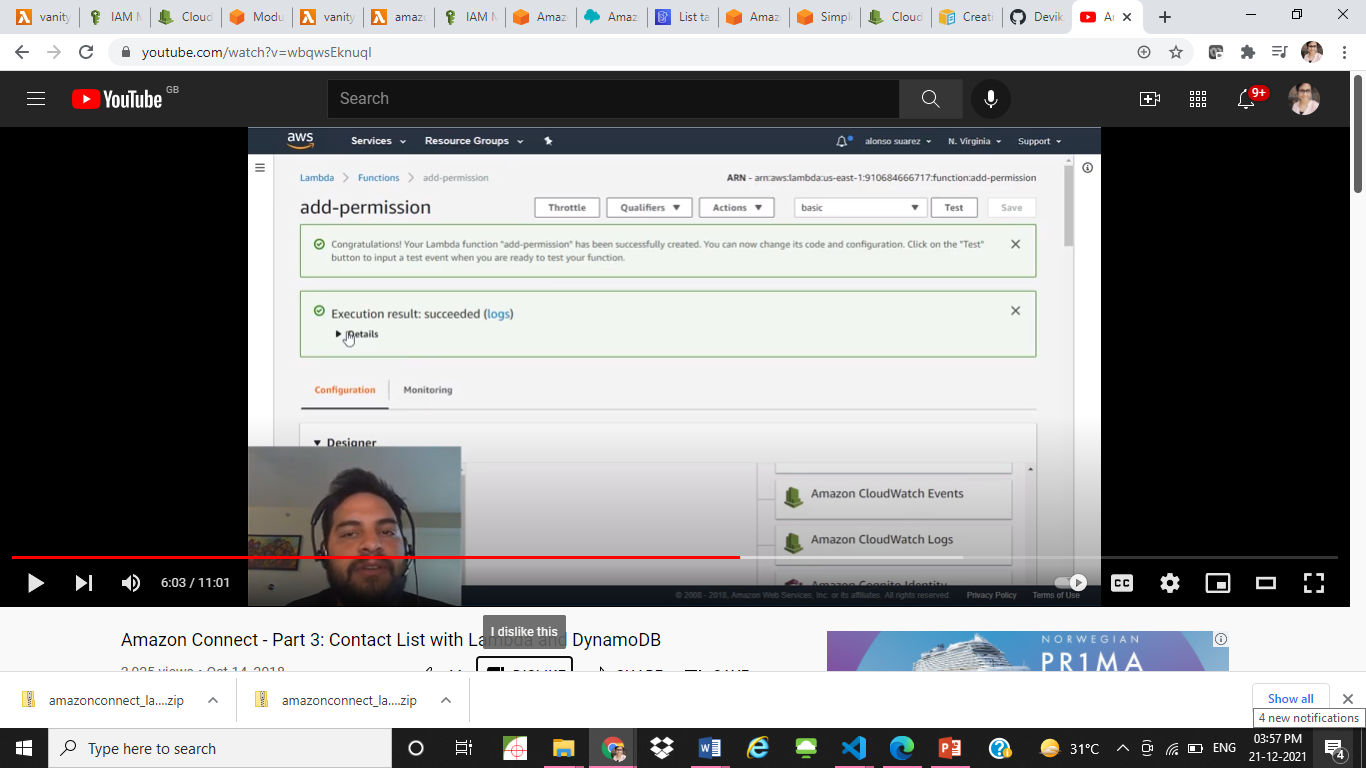
* + Contact flow of Vanity Number Generator:



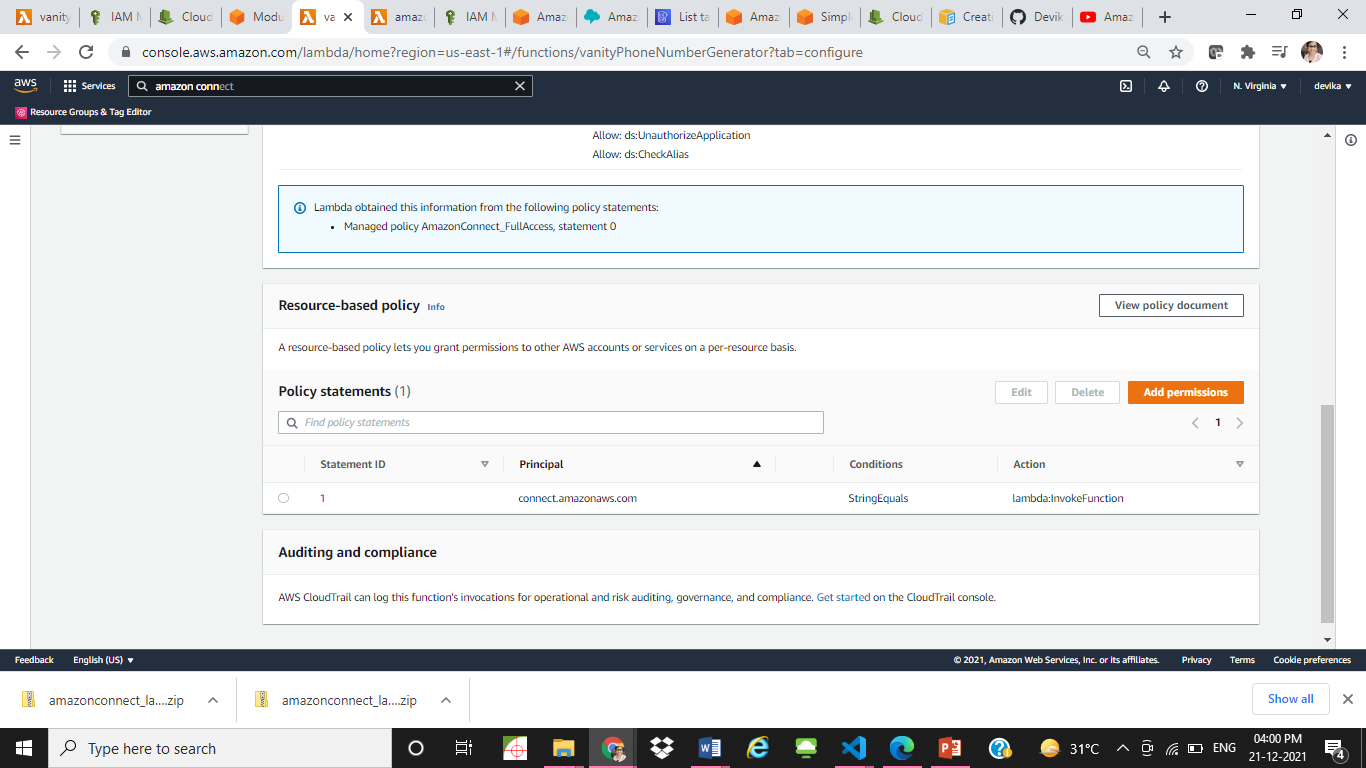
* Refer: [Create Amazon Connect contact flows](https://docs.aws.amazon.com/connect/latest/adminguide/connect-contact-flows.html)
  + From Integrate, select Invoke AWS Lambda function block. Specify function ARN inside Invoke AWS Lambda function.
  + Use $External to get response from Lambda function. Eg: $External.Response

**Set permission for Amazon Connect Instance to trigger Lambda**

* Amazon connect cannot directly trigger a private lambda. For that, permissions needs to be added to the lambda function for amazon connect to trigger.
* Create a new IAM Role:
  + Create role
  + Choose service: Lambda, click Next:Permissions
  + In Attach permission policies, add AWSLambdaFullAccess -> Next
  + In create role, give a role name eg: Lambda-Admin
* Create a new lambda function for setting permissions with this new IAM role created.
* Import amazonconnect\_lambda\_permission.zip from Deployment folder.
* Add “AccountID” as Key and Account number as Value in environment variable.
* Configure the test event with any input, deploy and test the lambda function.



* After getting Execution result: succeeded message, go to Configurations > Permissions of main lambda function for vanity number generation. The Resource-based policy will now be added to the vanity number generator lambda function to allow amazon connect to invoke the lambda function.

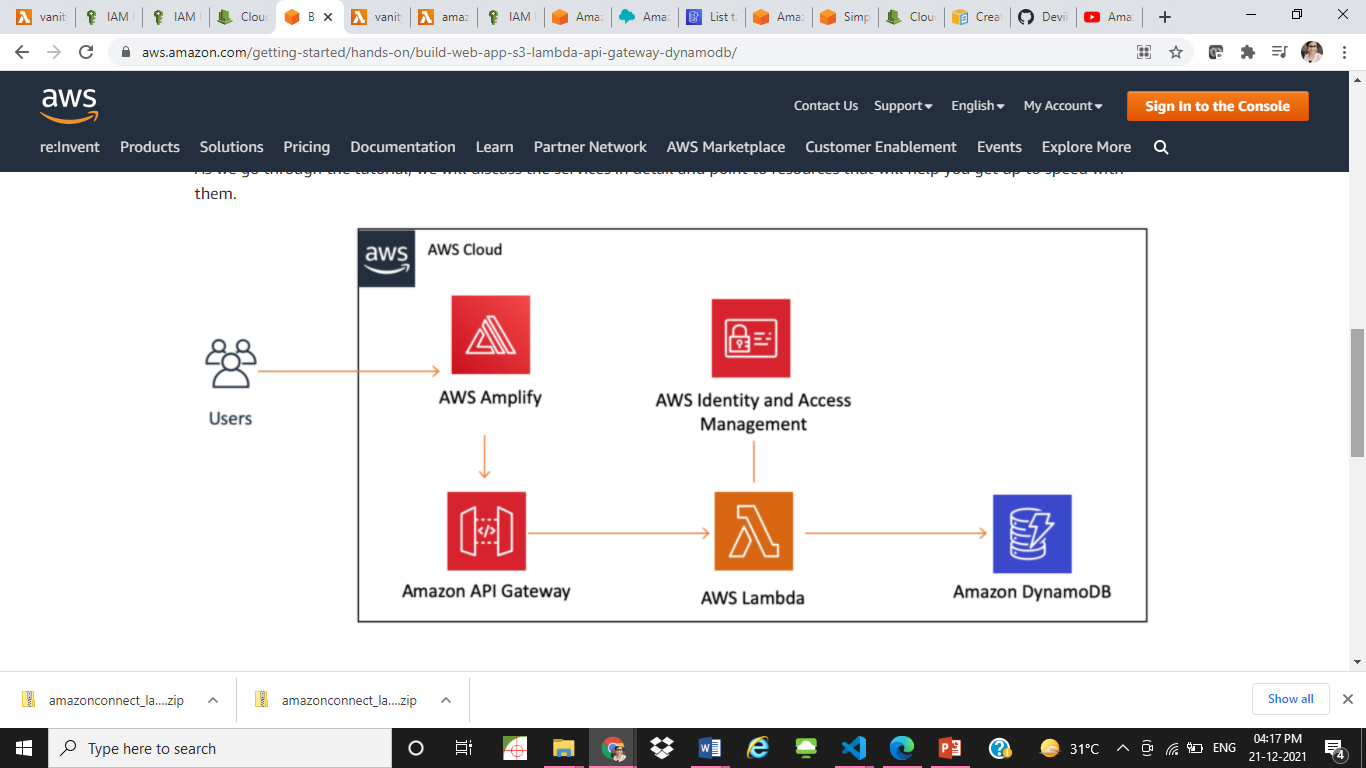


* After all these steps are completed, dial the claimed phone number from your UK mobile number. (The code not tested with numbers outside UK). The call will connect to Amazon connect and the following audio will be played: “Press 1 to generate top3 vanity Numbers, Press 2 to exit.” . Continue with your options and check the results obtained. Call one more time from the same number to fetch the existing data from the DynamoDB database.

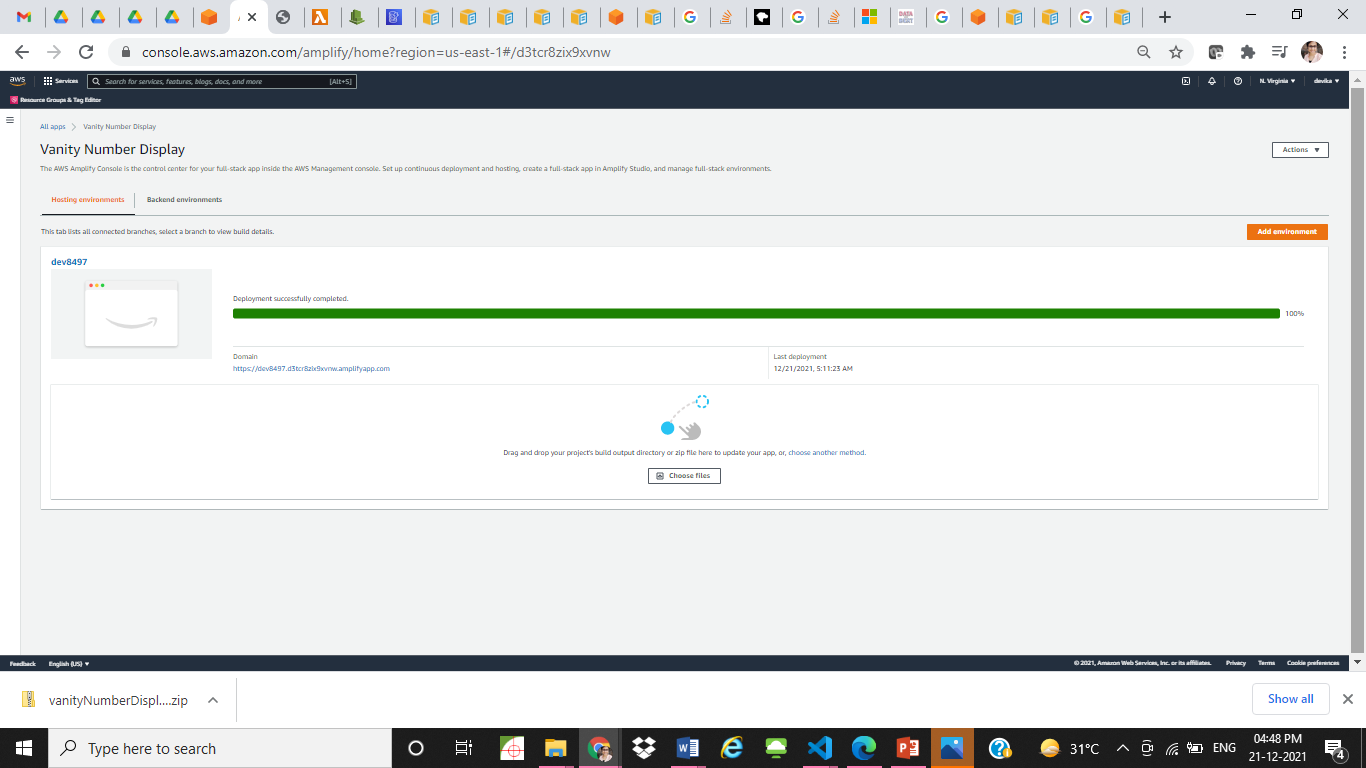
**Web App to display Vanity numbers from last 5 callers:**

Refer: [Build a Basic Web Application with Amazon Amplify](https://aws.amazon.com/getting-started/hands-on/build-web-app-s3-lambda-api-gateway-dynamodb/)

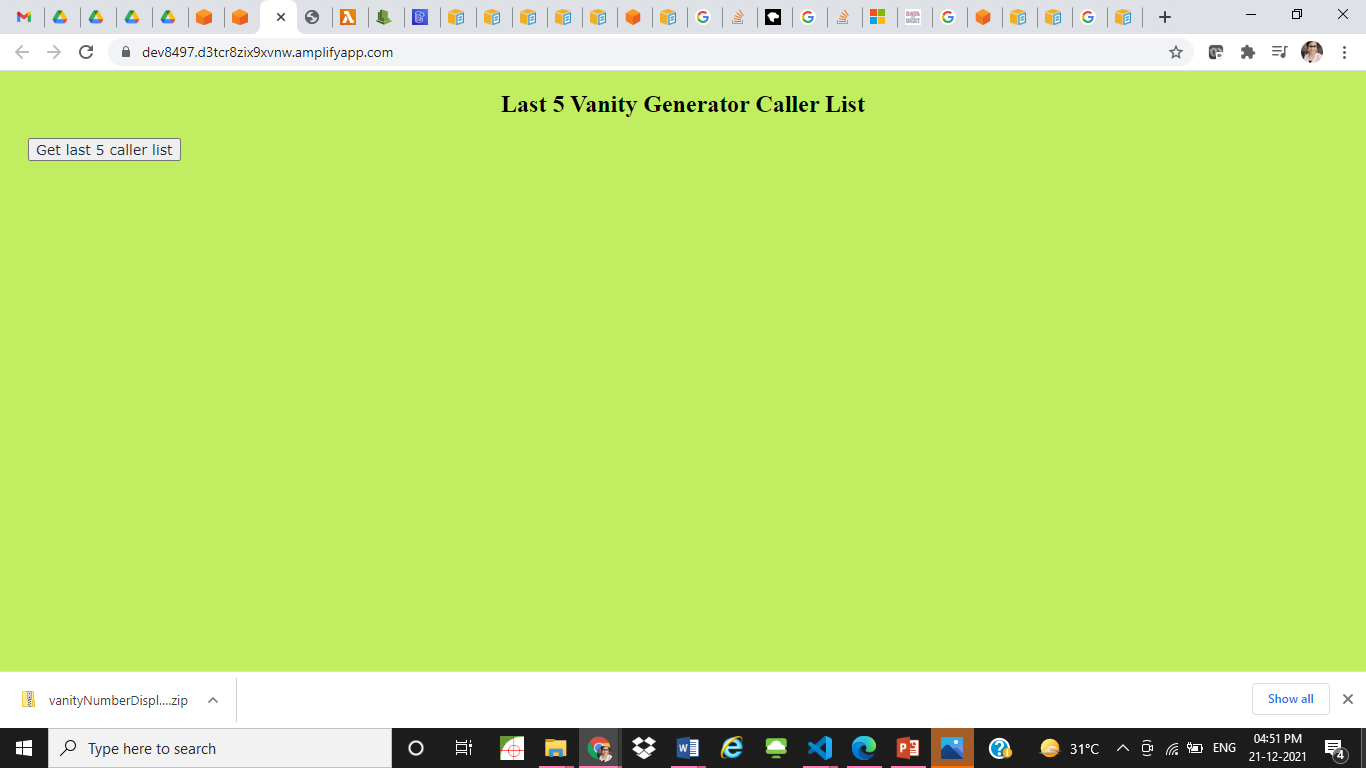
ARCHITECTURE DIAGRAM:



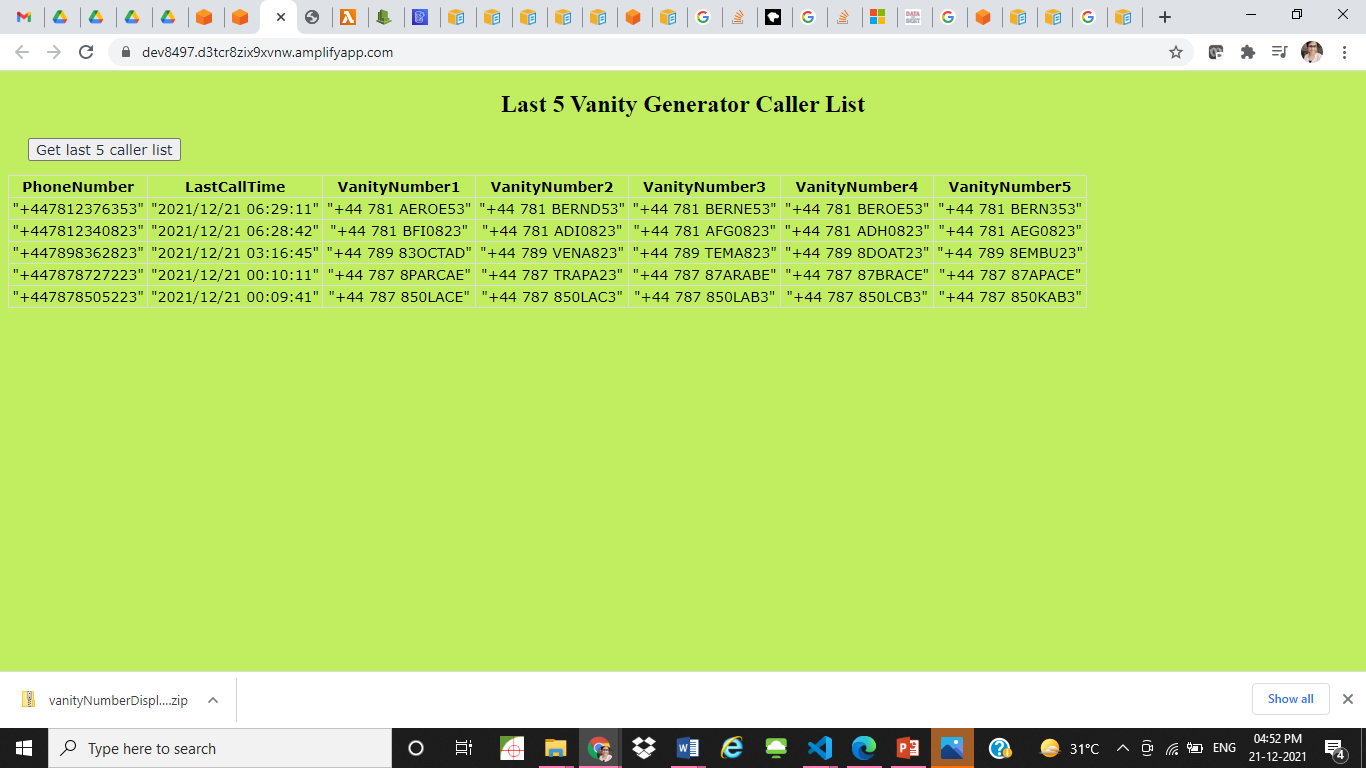
* Create a web app with Amplify console and test the web app by following the instructions here: [Create a Web App](https://aws.amazon.com/getting-started/hands-on/build-web-app-s3-lambda-api-gateway-dynamodb/module-one/?e=gs2020&p=build-a-web-app-intro)
  + Note: The file name of the file uploaded to Amplify console should always be index.html
* Follow [Build a Serverless Function](https://aws.amazon.com/getting-started/hands-on/build-web-app-s3-lambda-api-gateway-dynamodb/module-two/) to create a lambda function with default Role Permissions.
* [Link a Serverless Function to a Web App](https://aws.amazon.com/getting-started/hands-on/build-web-app-s3-lambda-api-gateway-dynamodb/module-three/?e=gs2020&p=build-a-web-app-two) explains how to create a REST API using API Gateway, add method, deploy and test the API Gateway.
* There is no need to create a new dynamoDB table as already a table is created to store the vanity numbers generated. Add IAM Policies to the lambda function role to access dynamodb, SQS, SNS Full accesses.
* Import vanityNumberDisplay.zip from Deployment folder to the serverless lambda function created. Add environment variables for SNS Arn, SQS Queue and part of dynamoDB table name.
* This code fetches all the entries from the dynamodb table, sort the entries based on the lastCallTime and returns the last 5 callers details including the phoneNumber and top 5 vanity numbers generated.
* Update the html code to make the web app more interactive and test the app by uploading zipped index.html file to the Amplify console.



* By clicking the domain link, the web app will open.

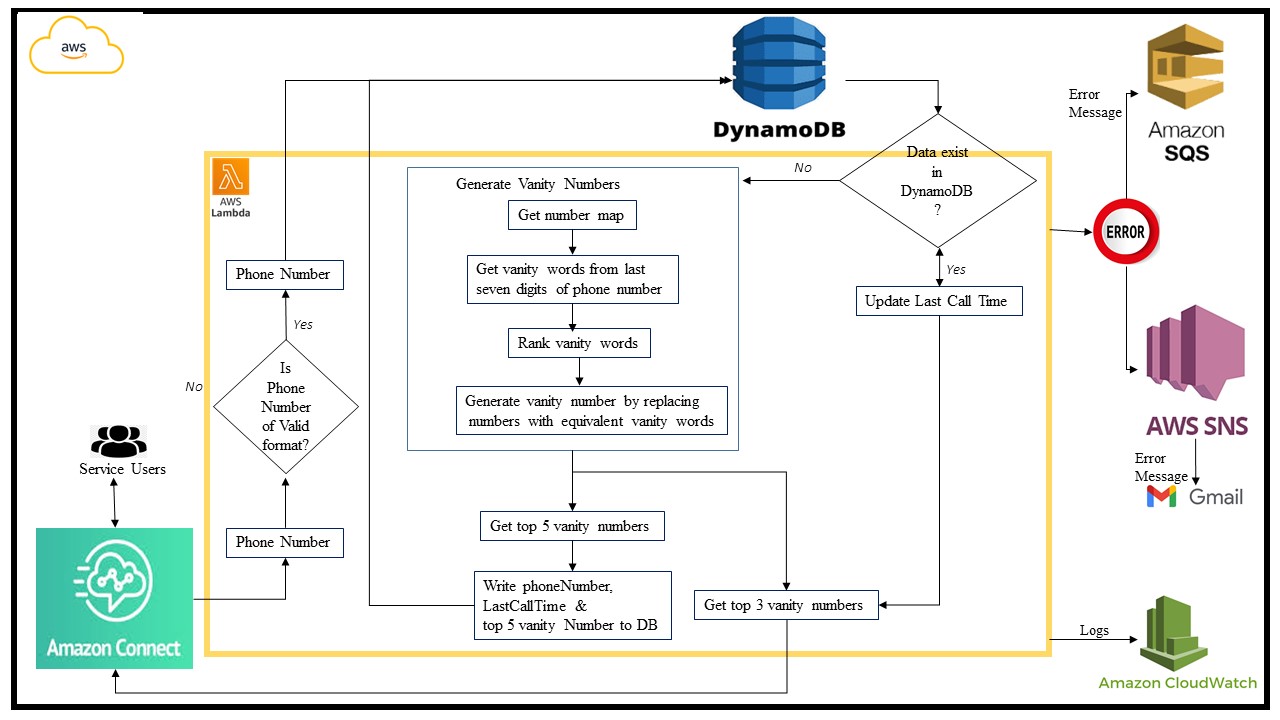


* Click ‘Get last 5 caller list’
* The details of the last 5 callers will be displayed in a table.



# Navigational Flow

ARCHITECTURE FLOW DIAGRAM OF VANITY NUMBER GENERATOR:



Steps:

Vanity Number Generation:

1. Get the user phone number details from amazon connect on invocation
2. Validate the phone number for UK phone number format.
3. If phone number is valid, check DynamoDB table to find the presence of any existing data for the input phone number.
   1. Return the top 3 vanity numbers.
   2. Update last call time in dynamodb.
4. If phone number is not valid, generate vanity numbers for the input phone number following these steps:
   1. Get number map for words. If number map not present in the directory, generate number map following the steps:
      1. Fetch words from nltk corpuses wordnet, brown, Gutenberg, reuters and names.
      2. Get the set of all words with length between 2 and 7 letters.
      3. Generate mapping json file for all the words to the corresponding numbers.
   2. Generate vanity words from number map for the last seven digits of the input phone number.
   3. Rank the vanity words based on the length of vanity words.
   4. Replace the number corresponding to the vanity word in the phone number with the vanity word for all vanity words on rank basis.
   5. Write the top 5 vanity numbers generated, phone number and present date time to dynamodb table.
   6. Return top3 vanity numbers.
5. Return top 3 vanity numbers to amazon connect after constructing response message in proper format with breaks and spaces

# Software Requirements

nltk corpuses wordnet, brown, Gutenberg, reuters and names.

# Software Installation Guidelines

In local environment, create a virtual python environment.

Install nltk

In terminal, perform the following:

>python

>import nltk

>nltk.download(‘wordnet’)

>nltk.download(‘brown’)

>nltk.download(‘gutenberg’)

>nltk.download(‘reuters’)

>nltk.download(‘names’)

After all these packages are installed, go to virtual environment folder>lib>python>site-packages

Copy nltk folder inside site-packages,

Paste nltk folder inside a new empty folder. Compress the new folder with nltk folder. Upload this zipped file to create nltk layer in aws. Add this nltk layer to vanity generator lambda.

# AWS Installation Guidelines

Refer: [Getting started with the AWS CLI](https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-getting-started.html)