

VoiceFoundry: Vanity Number Contact Center

Description:

The task of this project was to create an AWS Connect contact center flow that would interact with AWS Lambda and DynamoDB to convert a customer's phone number to possible vanity numbers and return 3 of them and save the 5 best vanity numbers in a DynamoDB table.

Contact Flow Phone Number: 855-541-2491

Q&A:

1. Record your reasons for implementing the solution the way you did, struggles you faced and problems you overcame.
 - a. When I received the project, immediately I thought to implement an algorithm that would generate the permutations of letters after converting the numbers to the possible numbers for that each digit. I realized that the permutations are exponential, and the lambda would exceed the time out time and the Connect Contact Flow would time out and the vanity numbers would never be returned to the caller.

I realized I didn't want any combination of letters that is not the same size as the number of digits entered so I was able to come up with a letter combination algorithm, which made the code faster, less vanity possibilities.

I went through all the possibilities and chose 3 random combinations and my best vanity would be the one that has the most vowels in the string.

My lambda then takes the three possibilities and puts them in an SSML string that can be read out by the connect contact flow.

- b. I faced many struggles. The original plan was a different implementation, but the max timeout of 8 seconds in the invoke action in the contact flow is hard to beat. Originally, I had a lambda that got me all the combinations, then I would chose the vanities that had words in them by checking the vanity possibility with a node package called check-word.

I even got to the point where I would take in the phone number and just get the combinations of the last 4 digits and then the best one would be the one that has the longest word in the vanity.

I could not get that lambda to work in under 8 seconds, therefore I went with the second strategy. Still has room for improvement.

- c. I did not have experience working with AWS Connect, I could not find any documentation on deploying an AWS Connect instance or creating a contact flow using AWS CDK or using a CloudFormation template, and to use lambda aliases,

AWS Connect requires more than just pasting the alias ARN in the invoke lambda contact flow action.

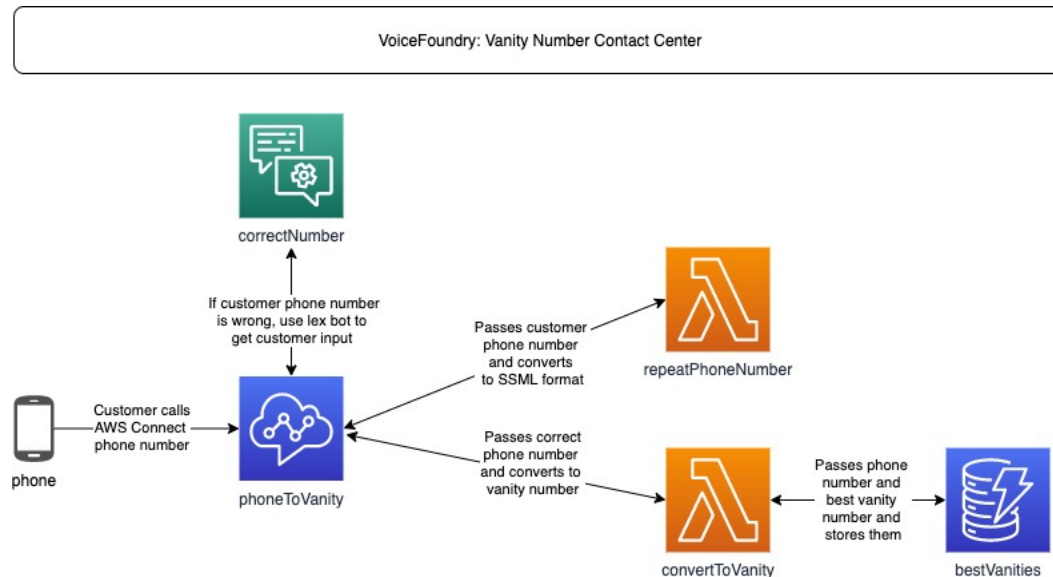
I was able to go through documentation, watch videos on how AWS Connect interacts with lex and lambda to create the contact flow. I was able to create/interact with a lex bot to give the project a more friendly flow by checking with the customer to see if their phone number was correct.

I provided links to help set up the connect instance, I provided instructions on how to deploy the cdk, and instructions to set up the contact flow.

2. What shortcuts did you take that would be bad practice in a production?
 - a. For starters, when creating any project on any stack, it is imperative that we have a development and production environment at the very least but having a staging environment for internal release is a great thing to do in order to provide better quality software.
 - b. Another thing, using node packages is a fast way to implement certain functions in your code, however, packages change all the time, or are sometimes left with no support. It is important to vet the node packages that you use before you implement them in your code. In the end, I only used the uuid node package and not the check-word package.
3. What would you have done with more time?
 - a. I would have created a development contact flow that would mirror the production contact flow, but it would call the development alias lambdas so that if any changes needed to be made, the production contact flow can be in use while you make changes to the development. I would have also played around with a blue green deployment where you could manage the load of the number of callers depending on how many callers are using the contact flow.
 - b. With more time, I would have added better error handling tailored to each customer input actions in the contact flow.
 - c. One last thing, I would make a lex bot that would repeat the output of the phone number and lambda in Spanish and also get it working to where it uses the number the customer enters if the caller ID is masked and is not the number that the caller wants to convert to a vanity.

Architecture:

Below is a diagram of all the Amazon Web Services that are used in the project and how they interact with each other.

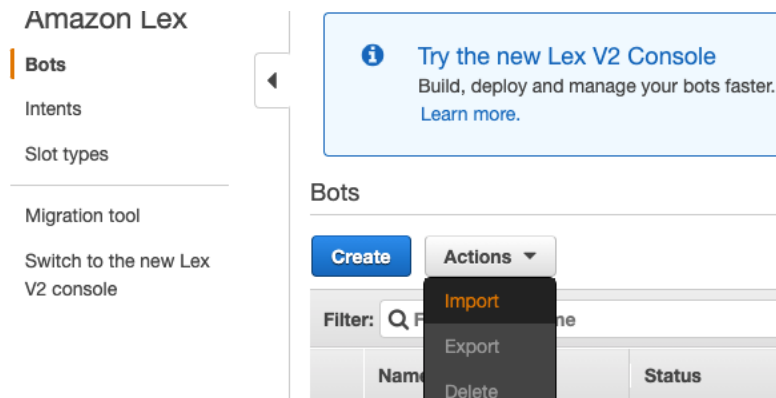


Deploy Resources Using AWS CDK:

1. Download 'cdk-demo' cdk file from the repository
2. Decompress the "cdk-demo" zip file
3. In the command line, change directories to the "cdk-demo" file
4. If you do not have the aws cli set up, please do so using the information at the link below
 - a. <https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2.html>
5. If you do not have the aws cli configured to use your aws account, please do so using the information at the link below
 - a. <https://docs.aws.amazon.com/cli/latest/userguide/cli-configure-quickstart.html#cli-configure-quickstart-config>
6. If you do not have the aws-cdk tool installed, install it with instructions from this link below
 - a. <https://docs.aws.amazon.com/cdk/latest/guide/cli.html>
7. Once all of that is set up, change directories into the "cdk-demo" directory and type, "cdk synth" to generate the CloudFormation template that the cdk will deploy
8. When that is done, type "cdk bootstrap" to get the environment ready for deployment
9. Finally, type "cdk deploy" to deploy the cdk in your AWS account

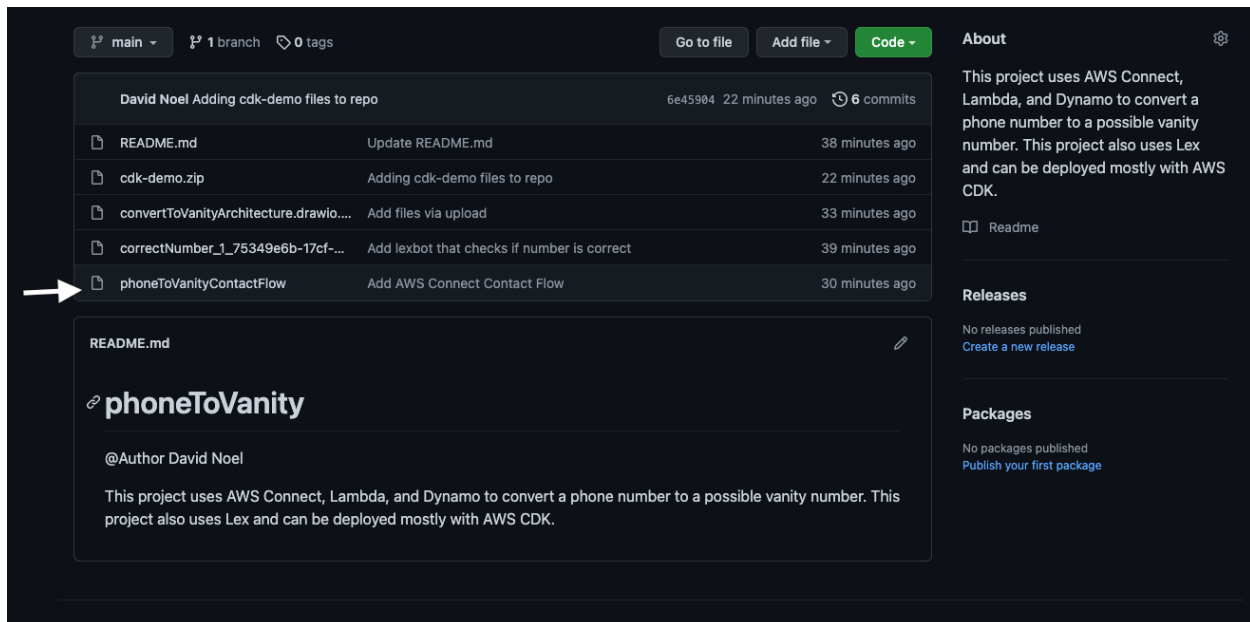
Importing Lex Bot:

1. Download the “correctNumber_1_75349e6b-17cf-4d0a-a5f1-b0455f5bd661_Bot_LEX_V1.zip” file from the repository
2. Go to Amazon lex and click on the “Actions” button, then click “Import” to upload the file.

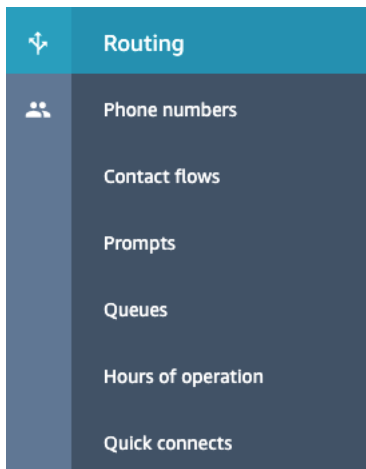


Setting up the Contact Flow:

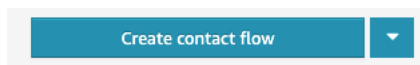
1. Create a new connect instance, using the information from the link below
 - a. <https://docs.aws.amazon.com/connect/latest/adminguide/amazon-connect-instances.html>
2. Download the contact flow “phoneToVanityContactFlow” from the “phoneToVanity” repository



3. Login to your AWS Connect instance and click on “Routing” on the left-hand side tool bar. Then click on “Contact flows”.



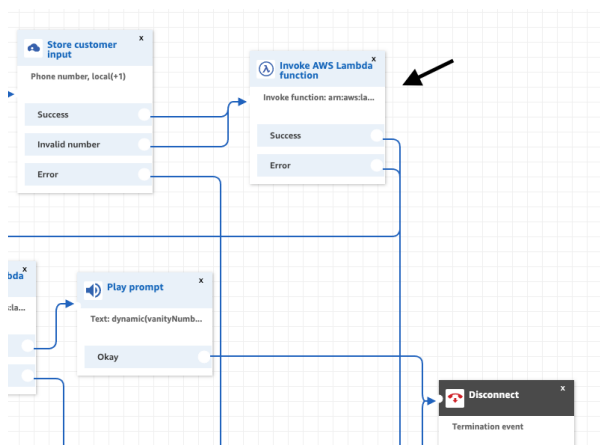
4. Create a new contact flow by clicking on “Create contact flow” on the top right-hand corner of the screen.



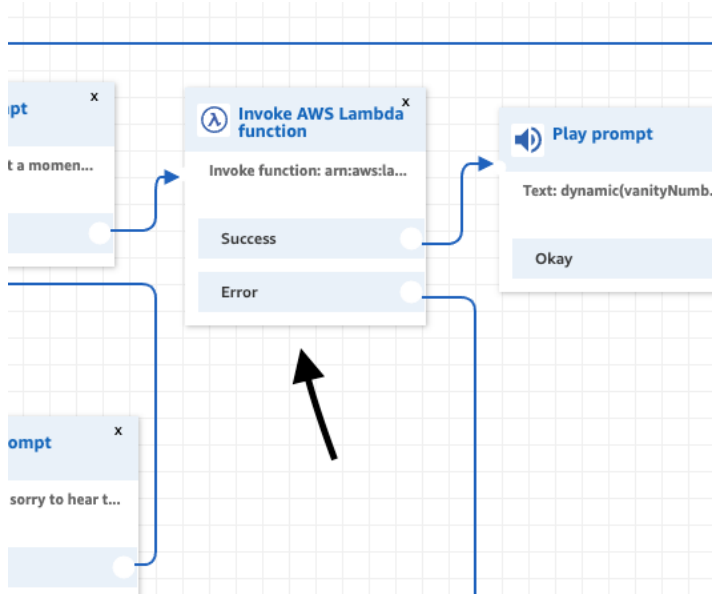
5. Click the dropdown, next to “Save”, on the top right-hand corner on the screen. Import “phoneToVanityContactFlow”.



6. Add repeatPhoneNumber lambda ARN to the first invoke in the contact flow



7. Add convertToVanity lambda ARN to the second invoke in the contact flow



8. Add the correctNumber lex bot to the Get Customer Input action

The screenshot shows the 'Get customer input' configuration panel in Amazon Lex. On the left, a contact flow diagram shows a 'Get customer input' block with several output paths: 'correctNumber (US East: N...', 'correctNumber', 'wrongNumber', 'maybeNumber', 'Default', and 'Error'. A black arrow points to the 'correctNumber' output path. The right panel is the configuration for the 'Get customer input' action. It includes a description: 'Delivers an audio or chat message to solicit customer input.' Below this, there are settings for 'Interpret as' (set to 'Text'), 'DTMF' (set to 'Amazon Lex'), and 'Lex bot'. Under 'Lex bot', the 'Select a Lex bot' option is chosen, and the 'Name' is set to 'correctNumber (US East: N. Virginia) (Classic)'. The 'Alias' is set to '\$LATEST'. There are also sections for 'Session attributes' and 'Intents'. Under 'Intents', three intents are listed: 'correctNumber', 'wrongNumber', and 'maybeNumber', each with a close button (X). At the bottom, there is a link to 'Add another intent'.


9. Add a phone number to the contact flow by clicking Routing->Phone numbers->Claim a number. Chose the country and then attach it to the contact flow.

Claim Phone number

Toll free

DID (Direct Inward Dialing)

Country

 +1

Prefix (optional)

☒ +1 833-853-5360

☐ +1 833-832-1461

☐ +1 833-264-8049


☐ +1 833-217-3833

☐ +1 833-866-6047

Optional information

Description



Enter description for the number



250 of 250 characters remaining.

Contact flow / IVR

phoneToVanity

Save

Cancel