*AI CHATBOT USING AMAZON LEX AND LAMBDA*

COVIDTRACKER CHATBOT

*Rahna Latheef*

*Table of Contents*

[OVERVIEW 3](#_Toc155117617)

[OBJECTIVES 3](#_Toc155117618)

[ARCHITECTURE AND DESIGN 4](#_Toc155117619)

[1. Amazon Lex 4](#_Toc155117620)

[2. Amazon Lambda 7](#_Toc155117621)

[3. Integration 7](#_Toc155117622)

[RESOURCE PROVISIONING AND CONFIGURATION 8](#_Toc155117623)

[1. Amazon Lex Setup 8](#_Toc155117624)

[2. Amazon Lambda Setup 9](#_Toc155117625)

[IMPLEMENTATION AND INTEGRATION 10](#_Toc155117626)

[1.Creating Chatbot 10](#_Toc155117627)

[2. Define AWS Lambda Function 16](#_Toc155117628)

[3. Integrating Lambda Function with Bot 23](#_Toc155117629)

[DEPLOYMENT AND TESTING 23](#_Toc155117630)

[COST ANALYSIS AND OPTIMIZATION 26](#_Toc155117631)

[LESSONS LEARNED & RECOMMENDATIONS 26](#_Toc155117632)

[Lessons Learned : 26](#_Toc155117633)

[Recommendations : 27](#_Toc155117634)

[CONCLUSION 27](#_Toc155117635)

# ***OVERVIEW***

Chatbots have emerged as one of the latest trends in modern-day technology, and they're making a huge impact on our day-to-day activities. From scheduling appointments and medical advice to booking flights, providing real-time information. Chatbots are simplifying our lives and making everything easier and more efficient, as they can provide instant responses and personalised interactions.

The Covid AI Chatbot project aims to enhance user experience by implementing an intelligent conversational interface that providing latest information regarding covid19 in each countries. Leveraging the power of Amazon Lex and Lambda, the chatbot will facilitate seamless interactions, providing quick and accurate responses. The project emphasizes a user-centric approach, with a focus on natural language processing to ensure a human-like conversation flow. This chatbot will find applications in diverse domains, such as customer support, information retrieval, and task automation.

# ***OBJECTIVES***

1. Implement Conversational AI:

* Utilize Amazon Lex for building a robust and intuitive conversational interface.
* Ensure the chatbot understands and responds effectively to user queries.

2. Integrate Lambda for Backend Functionality:

* Develop Lambda functions to handle backend processes, enhancing the chatbot's functionality.
* Enable seamless integration between Amazon Lex and Lambda for efficient data processing.

3. Enhance User Engagement:

* Prioritize user-centric design principles to create an engaging and user-friendly experience.
* Incorporate features like dynamic content delivery and personalized responses to increase user satisfaction.

# 

# ***ARCHITECTURE AND DESIGN***

A screenshot of a chat application

Description automatically generated

## Amazon Lex

AWS Lex is a cloud-based service provided by Amazon Web Services that enables developers to build conversational interfaces using voice and text. It uses natural language understanding (NLU) technology to understand and interpret user input, allowing developers to create chatbots, voice bots, and virtual assistants that can interact with users in a human-like way. AWS Lex is like a brain that can understand what people say or type, and then respond back like a human.

The key fundamental components of the Lex service :

#### Intents

A screenshot of a computer

Description automatically generated

"Intents," as the name suggests, are a fundamental concept used to represent the purpose or goal of a user's input. Intents can be viewed as a verb, detecting what a user's intention is. For example, if you go to a pizza shop and order a pizza, your main intention is to order pizza, your purpose for going to the store is to get pizza. This works the same way for lex-powered chatbots. We have to define intents so the bot can easily track or identify our goals during a conversation.

#### Utterances

A white object with a white background

Description automatically generated

Utterances are examples of phrases or sentences that we might use to convey a specific meaning or request. For example, when we walk into the pizza store to order a box of pizza, a possible request we could make is "I want to order a box of pizza." These utterances are used to teach the chatbot how to understand and respond to what we are saying. The more utterance we give, the better the chatbot will be at understanding what we’re trying to say.

#### Slots and Slot Types

A screenshot of a computer

Description automatically generated

Slots are a collection of information that prompt chatbot users to provide during a conversation with your bot. In the COVID chatbot we can define slots by country and prompt the user to enter a value for each slot. The bot will prompt the user to provide information for each slot, and once all of the required slots have been filled, the bot will be able to process the order and respond appropriately.

## Amazon Lambda

AWS Lambda is a way to write and run code without worrying about setting up and maintaining the underlying server or computing resources. We can focus on writing our code, and Lambda takes care of executing it and scaling it as needed based on demand. It’s a popular choice for building serverless applications.

The COVID Tracker chatbot relies on lambda functions to perform various tasks based on user input or the state of a conversation. This is why we require a lambda function for our project:

1. **Retrieving COVID data:** Based on what our users request for a specific country, our Lambda function will retrieve the most recent COVID data from a third-party API. It provides our users with real-time information.

<https://disease.sh/docs/>

1. **Processing user input:** Before returning a response to the user, our lambda function will process and validate their input. If a user requests data for a country that does not exist, the lambda function will respond with an appropriate error message.

## Integration

The integration of Amazon Lex and AWS Lambda ensures a cohesive and unified chatbot system. While Lex manages the conversational aspects, Lambda functions execute backend logic, allowing for real-time processing and response generation.

Benefits

1. Scalability: Leveraging AWS Lambda ensures scalability, enabling the system to handle varying workloads efficiently.

2. Flexibility: The combination of Amazon Lex and Lambda offers flexibility in designing and modifying the chatbot's behaviour.

3. Seamless Interaction: The orchestrated architecture ensures a seamless and natural interaction flow, enhancing the overall user experience.

# ***RESOURCE PROVISIONING AND CONFIGURATION***

Resource provisioning is a critical phase in the development of an AI chatbot using Amazon Lex and AWS Lambda. This process encompasses the setup of essential services, including Amazon Lex for natural language understanding and AWS Lambda for backend processing. Configuring permissions and ensuring optimal resource allocation are key aspects of this phase, ultimately ensuring the chatbot's seamless interaction with users.

## Amazon Lex Setup

* Log into AWS Management console
* In the Management Console, search for the Amazon Lex service and select to open the service dashboard.

A screenshot of a computer

Description automatically generated

## Amazon Lambda Setup

* In the Management Console, search for the Amazon Lambda service and select to open the service dashboard.

A screenshot of a computer

Description automatically generated

# ***IMPLEMENTATION AND INTEGRATION***

## 1.Creating Chatbot

* In the Management Console, search for the Amazon Lex service and select to open the service dashboard.
* From the sidebar click on “Return to the V1 console” to switch to the V1 user interface console.

A screenshot of a computer

Description automatically generated

* To create a new chatbot, click the "Create" button.

A screenshot of a computer

Description automatically generated

A screenshot of a chat

Description automatically generated

* On the next screen select the “Custom bot

A screenshot of a computer

Description automatically generated

* For our custom bot, we must set up a few basic options. The following fields need to be specified:

**Bot name**: This field includes the name of the bot. It must be between 2 and 50 characters long and only allow letters—no spaces.

**Language**: This is the language that our chatbot will use to communicate

**Output voice**: The Lex Service provides us with different voice outputs we can use for our chatbot. This is a text-based chat bot.

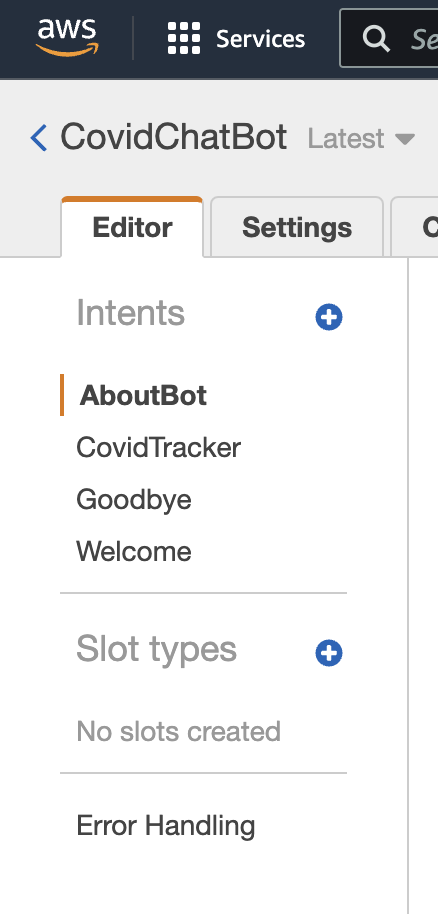
**Session timeout**: This option specifies how long we want Amazon Lex to retain the context; it can be set between zero and 1440 (24 hours)

**COPPA**: This stands for the Children’s Online Privacy Protection Act. This indicates if our bot is subjected to COPPA. Select no, as our bot does not apply to COPPA.

A screenshot of a computer

Description automatically generated

* The next page you see is the “Bot Editor”, where we will define the Intents for the bot.
* Multiple intents can be created for the bot. For e.g. Welcome, Covid Tracker, Good bye and About Bot.
* Click the "Create Intent" button to open a modal window with three options; select the "Create Intent" option. Give each intent a distinct name so that we can distinguish them when writing lambda function.



* For create an utterance, on the sidebar, select the intent you want to create possible utterances for. Under the "Sample Utterances," we can go ahead and add as many possible utterances as a user can ask our chatbot.
* Welcome Intent

A screenshot of a computer

Description automatically generated

* Goodbye Intent

A screenshot of a computer

Description automatically generated

* About bot Intent

A screenshot of a computer

Description automatically generated

* To add slots, we need the bot to identify them, so we have to wrap the words in curly braces when defining an utterance.
* We've added a slot called "country," because our Covid Tracker Intent requires the name of the country where a user wants to learn more about the COVID situation.
* When we add slots, we must also include the slot type. This could be any value, but we'll use "Amazon.Country" in this case.
* Covid Tracker Intent

A screenshot of a computer

Description automatically generated

* Lastly “Save Intent” to save the changes

## 2. Define AWS Lambda Function

* To get started with AWS Lambda, search for the Lambda service in the AWS Management Console, and select.
* Click on “Create Function” to create the function for our chatbot. We will select “Author from scratch” option in this case

A screenshot of a computer

Description automatically generated

* Provide the function name, select the runtime as “Node.js” and click on “Create function” to proceed further. Next page is the function editor, where we will write the code to invoke the http rest api to fetch the covid data.

A screenshot of a computer

Description automatically generated

* To create a function for the first time, I used external VSCode editor to write code , It is because for the node js function, need to install dependencies required for the function to access the rest api.
* Below are the commands to be executed in sequence in the VSCode Terminal:

*#creates package.json file  
> npm init -y*

*#For asynchronous request to the API  
> npm install request request-promise –save*

*#create index.js file (make sure the name of the file is same as below, since it will be referred as default handler in lambda function as "index.handler")  
> touch index.js*

* Attaching the function (node js function in the index.js file) written file link here:

[index.js file](https://github.com/rahnaLatheef/CovidTracker-Bot/blob/main/index.js)

-This is a Node.js module that exports a function called handler. This function takes an input event object and returns a response object.

-The handler function first imports two Node.js packages: **request-promise** and **numeral**. request-promise is a library that allows the code to make HTTP requests to external APIs. **numeral** is a library that helps with formatting numbers.

-The dispatcher function is defined next. This function takes an event object and returns a response object. It starts by creating a default response object with a Close dialog action type and an empty message.

-The function then switches on the name property of the currentIntent object in the event input to determine what action to take. The code has cases for the following intents:

**-CovidTracker**: This case retrieves COVID-19 statistics for a country and formats them into a response message. It uses the request-promise library to make a GET request to an external API, retrieves the relevant statistics, and formats them into a response message using the numeral library. The response message is then added to the message property of the default response object.

**-Goodbye**: This case simply returns a farewell message as the response.

**-AboutBot**: This case returns a message describing the purpose of the bot.

**-Welcome**: This case returns a greeting message as the response.

**-Default**: This case handles any other intent not covered by the other cases. It simply returns a generic message saying that no data was found for the requested country.

-Finally, the dispatcher function returns the response object.

-The handler function simply calls the dispatcher function with the event input and returns the result. This makes the dispatcher function the main logic for handling the input and generating the output of the AWS Lambda function that is using this code.

* After saving the code to index.js, we need to zip the function (Since we are using npm modules for this lambda function) for which we will do the below step:

*#create zip file  
 > zip -r handler.zip \**

* Now, go back to the lambda function in the AWS Console and choose “Upload from .zip file”

A screenshot of a computer

Description automatically generated

* After uploading the code, the directory structure should look like below:

A screenshot of a computer

Description automatically generated

* The next step is to test the lambda function, for which we will need to Configure test event
* On click of Configure test event, select the “Create new test event” and Event Template as “lex-make-appointment”
* Modify the template to add the intent name which you created and Save:

A screenshot of a computer

Description automatically generated

* After the event is saved, click on the Test button to get the output.

A screenshot of a computer

Description automatically generated

## Integrating Lambda Function with Bot

* Coming back to the Lex service to integrate the chatbot using Lambda function.
* Choose one of the four previous intents. Under "Fulfillment," choose the AWS Lambda Function option and the chatBotFunction, making sure it's the most recent version. Do this for the remaining three intents.

A screenshot of a chat box

Description automatically generated

* To save all changes, click the "Save Intent" button. When making changes to any of your created intents, always click the save button.

# ***DEPLOYMENT AND TESTING***

* Select the "Build" option. This button is located on the right side of the page. This button allows us to build our chatbot and use the test playground.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

* Use all of the utterances specified in each intent to have a conversation and test the bot's efficiency.

A screenshot of a chat

Description automatically generated

# *COST ANALYSIS AND OPTIMIZATION*

A screenshot of a computer

Description automatically generated

# *LESSONS LEARNED & RECOMMENDATIONS*

## Lessons Learned :

AWS Lex is a service that helps build conversational interfaces (chatbots) using natural language understanding. It integrates seamlessly with AWS Lambda, which allows custom logic and actions to be executed when interacting with the chatbot. In simple terms, AWS Lex understands and interprets user input, while AWS Lambda enables the chatbot to perform specific tasks or provide tailored responses based on that input.

## Recommendations :

* Integrate and deploy Bot with the website (Using Amazon Cloud Formation , Kommunicate etc )
* Integrate and deploy Bot with the Social Media Apps like Facebook , Telegram etc.

# ***CONCLUSION***

In conclusion, the project successfully achieved its goals of developing an AI chatbot using Amazon Lex and AWS Lambda. The implementation focused on optimizing user interactions through natural language processing, contributing to enhanced user experiences. By streamlining processes, the chatbot demonstrated efficiency in handling user queries and requests. Furthermore, the project successfully delivered on its objective of providing valuable information to users through seamless and intelligent interactions. The use of Amazon Lex and AWS Lambda showcased the capabilities of these technologies in creating an advanced and user-friendly chatbot system.