HEALTH CARE CHAT BOT

A project report submitted in the partial fulfillment of the requirements for the

Award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

Submitted By

AKSHAY SANKINENI

Reg.no: 121910311013

Under the Guidance of Mr.Moditya

24 June 2022

DECLARATION BY THE CANDIDATE

I the undersigned solemnly declare that the project report HEALTH CARE CHAT BOT is based on my own work carried out during the course of our study under the supervision of Mr.Moditya. I assert the statements made and conclusions drawn are an outcome of my research work. I further certify that

I. The work contained in the report is original and has been done by me under the general supervision of my supervisor.
II. The work has not been submitted to any other institution for any other degree/diploma/certificate in this university or any other University of India or abroad.
III. We have followed the guidelines provided by the university in writing the report.
IV. Whenever we have used materials (data, theoretical analysis, and text) from other sources, we have given due credit to them in the text of the report and giving their details in the references.

Akshay Sankineni

121910311013

PG-2223-ETSD-655

CERTIFICATE

This is to certify that this project work entitled

"HEALTH CARE CHAT BOT"

is the Bonafede work carried out by by Akshay Sankineni Reg.No: 121910311013 submitted in Partial fulfillment of the requirement for the Award of Degree of Bachelor of Technology in Computer Science and Engineering, during May-June 2022.

The results submitted in this project have been verified and are found to be satisfactory. The results embodied in this thesis have not been submitted to any other university for the award of the any other degree/diploma.

Moditya Kapakayala

Signature of project supervisor

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned the efforts with success. It is a pleasant aspect that I have now the opportunity to express my gratitude for all of them.

The first person I would like to thank my project guide Mr.Moditya, who had given continuous critical suggestions and extension of proper working atmosphere, abiding interest has finally evolved into this research work.

It is indeed with a great sense of pleasure and immense sense of guidance that I acknowledge the help and I am highly indebted to Prof. M.Venkateswara Rao, Vice Principal, and School of Technology, for his support during the tenure of the internship.

I would like to express my sincere thanks to Prof. Sirisha, Head of the Department Computer Science and engineering for providing the opportunity to Undertake this internship and encouragement in the completion of the project.

I am also thankful to all the staff members of the Computer Science and Engineering Department for their valuable suggestions. I would like to thank my team mates and parents who extended their help, encouragement and moral support either directly or indirectly in this project.

Akshay Sankineni

121910311013

CONTENTS

ABSTRACT

The main goal of the creation of Chatbots was to resemble a human being in the way they perform said interaction, trying to make the user think writing to another human being. This chatbot uses Amazon lex to build a text-based conversational interface for web applications. AWS lambda function is used to validate user's response and perform initialization and fulfilment in lex intent configuration. This chatbot uses AWS DynamoDB to control and stores the resources. The bot is deployed onAWS Cloud Formation. The application architecture uses AWS Lambda, Amazon Lex, Amazon DynamoDB, to communicate with the application deployed.

The chatbot is built to provide customer support 24x7, especially in the health domain web applications. This will help the user to interact with doctors at any time online and to search for the doctor's availability and book Appointment etc.This type of chatbot can be deployed in any application by using AWS tools.

ABOUT

Phoenix Global is a skill-development company that helps students acquire and master professional and soft skills as per the requirements of the industry benchmarked to world's top firms, trained by top class industry professionals.

Phoenix Global is a platform having Industry professionals with esteemed alma mater including the IITs and IIMs to mentor and train students on cutting-edge skills, critical to the emerging industries while also giving them an opportunity to intern on a project under the mentorship of industry professionals from the IITs /IIMs.

Our vision is to be a national leader in skill development and industry readiness training by providing differentiated training from top-class industry experts.The mission is to be a go-to skill development platform for students, imparting skills benchmarked at global standards that help them realize their dream careers profitably

Our core values, the 4Ps – Professionalism, Punctuality, Passion, Perseverance stand for who and what we are as an organization.

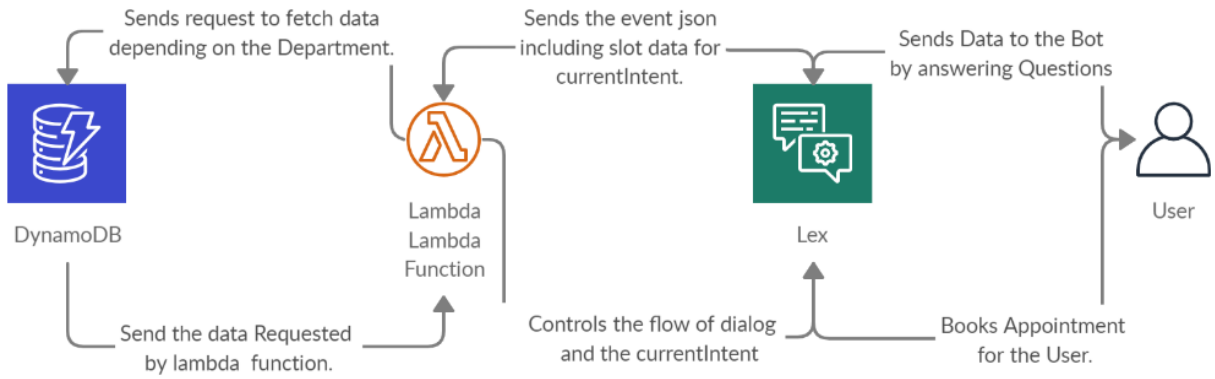
SCHEDULE OF INTERNSHIP

Day	Activity Plan
1	Induction Program
2	Pre-Readings/Material Distribution
3	Training Session - 1
4	Training Session - 2
5	Training Session - 3
6	Training Session - 4
7	Training Session - 5
8	Teams formation for Project
9	Weekend Off
10	Training Session - 6
11	Training Session - 7
12	Training Session - 8
13	Training Session - 9
14	Training Session - 10
15	Project Title Allocation
16	Weekend Off
17	Project Session - 1
18	Project Session - 2
19	Project Session - 3
20	Project Session - 4
21	Project Session - 5
22	Project Mid Review
23	Weekend Off
24	Project Session - 6
25	Project Session - 7
26	Project Session - 8
27	Project Session - 9
28	Project Session - 10
29-44	Project Working Sessions
45	Project Final Presentation and Thesis Defense

Introduction

Problem Statement

The chatbot was created in order to provide a facility in these troubling times to people sitting at home. we help them to find a doctor suitable to their requirements and book an appointment using the chatbot.



Above fig is a flow of working of our chatbot.

- User starts a conversation with the chatbot.
- According to the need user will make requests. This request is sent to Lex.
- From Lex control goes to Lambda function.
- In lambda function according to requirement it accessess the dynamodb and retrieve the required data.
- By using that retrieved data and user request lambda function will send the response to Lex.
- From lex the response is displayed to the user.

Services Used

- Amazon web services – Lex
  Amazon Lex is an AWS service for building conversational interfaces for applications using voice and text. With Amazon Lex, the same conversational engine that powers Amazon Alexa is now available to any developer, enabling you to build sophisticated, natural language chatbots into your new and existing applications

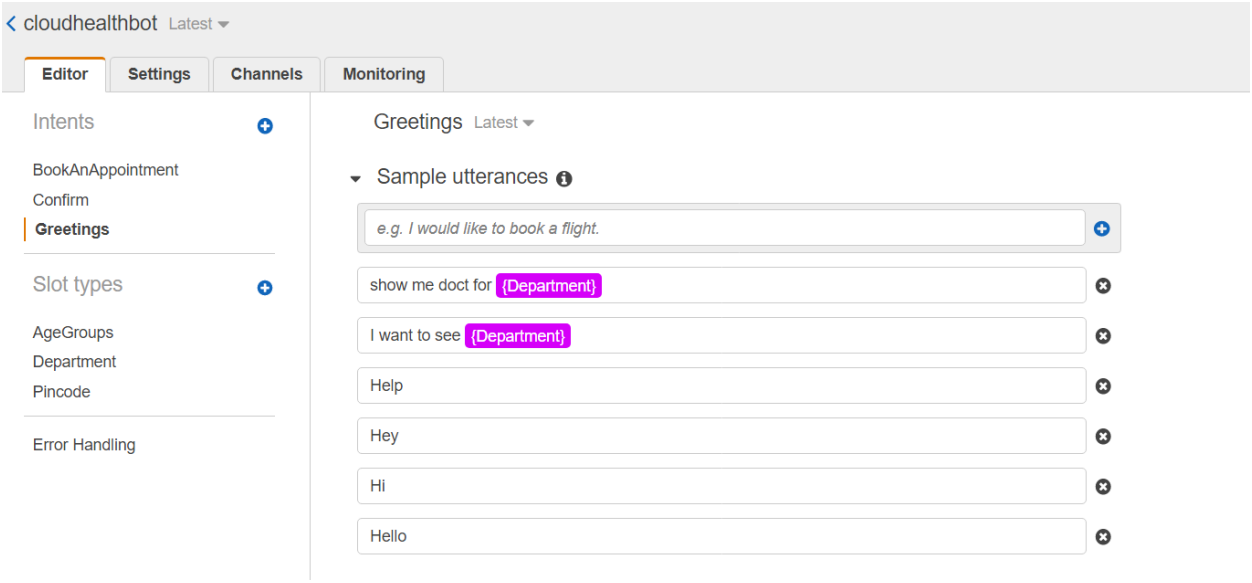
- Amazon web services– Lambda
  AWS Lambda is a serverless, event-driven compute service that lets you run code for virtually any type of application or backend service without provisioning or managing servers. You can trigger Lambda from over 200 AWS services and software as a service (SaaS) application, and only pay for what you use.

- Amazon web services – Dynamo-DB
  Amazon DynamoDB is a fully managed, serverless, key-value NoSQL database designed to run high-performance applications at any scale. DynamoDB offers built-in security, continuous backups, automated multi-Region replication, in-memory caching, and data export tools.
- Amazon web services– Cloud Formation
  AWS Cloud Formation is a service that gives developers and businesses an easy way to create a collection of related AWS and third-party resources, and provision and manage them in an orderly and predictable fashion.

Working of services

Amazon lex

User - The user interacts with the AWS Lex bot and starts the conversation by using the Sample Utterances toinvoke with the "Greetings" Intent. And based on information it gathers the information from the user on patient details.



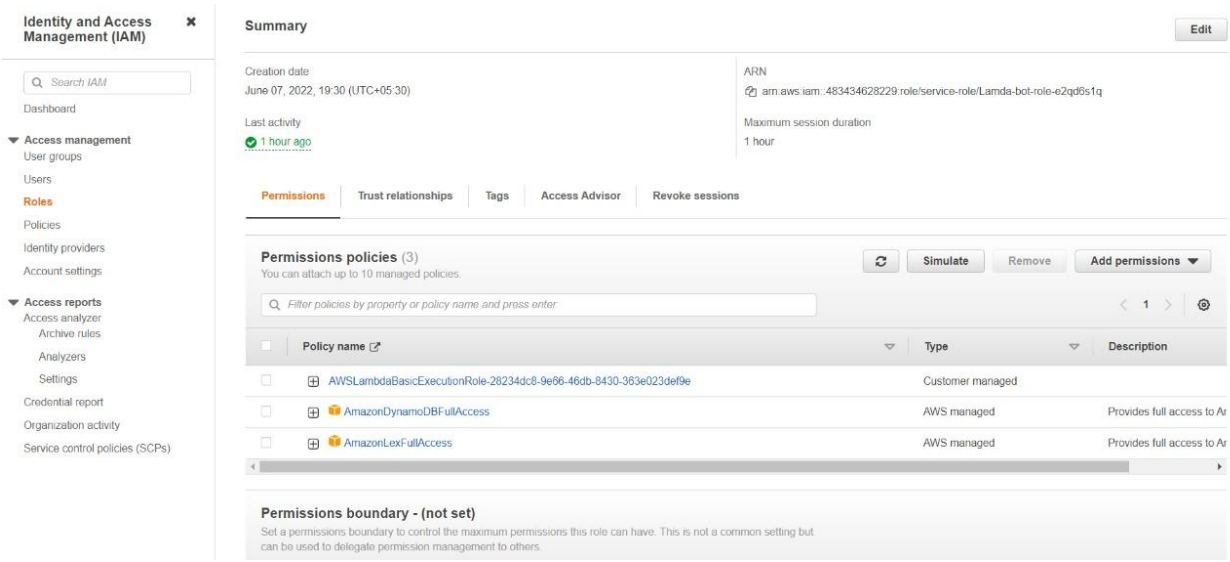
Lex - The Lex bot has 3 intents namely Greetings ,BookAnAppointment and Confirm. The first intent to get invoked is "Greetings".

- Greetings: The main purpose of this intent is to know the details of the patient and refer to the medical Department that the user needs to baseon the answers given by the user.
- BookAnAppointment: The use of this intent is to select the day for the appointment.
- Confirm: The use of this intent is to Book the appointment for the user after the user has choose the doctor they want to see

Prerequisites:

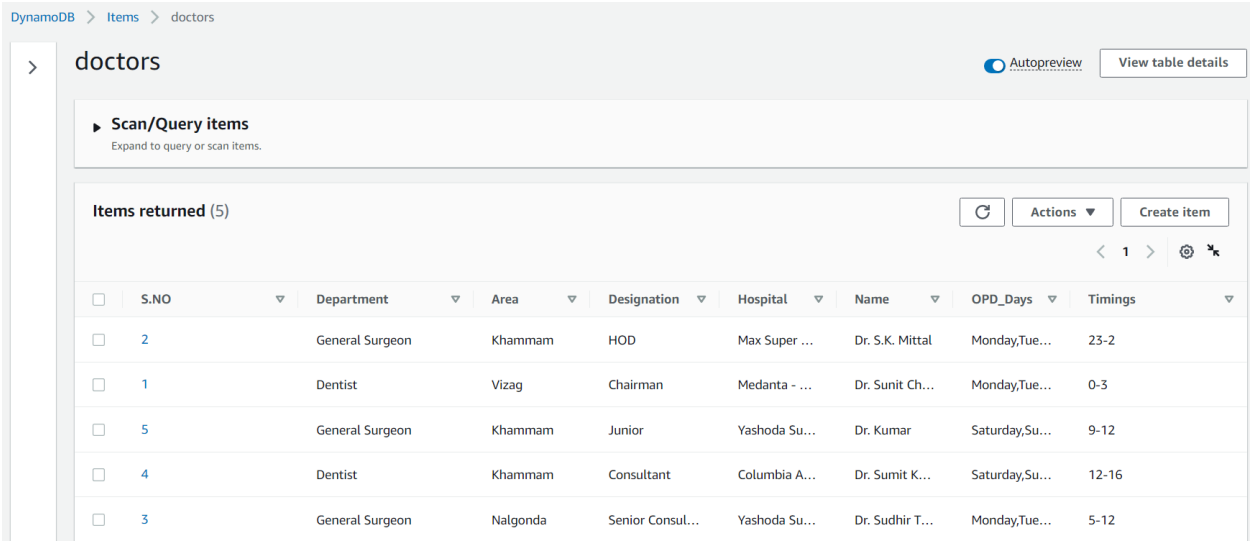
Lambda - Lambda is an integral part of this project as it has the job of fetching the data from the DynamoDB based on the medical Department inferred by the bot.

Another purpose of the the lambda function is to determine the flow of the conversation and handling the intent switching. lambda function is written in **Python** and The **IAM Roles** given to the lambda function

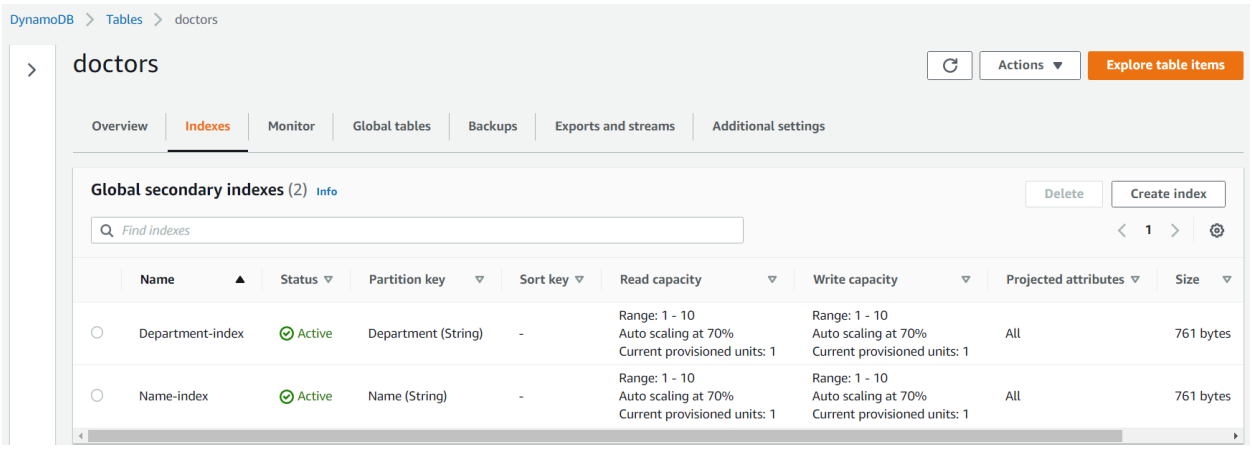


DynamoDB - It is the database that is used to store the information about the doctors and which area are they based in.

There are three main regions are Vizag, Khammam, Nalgonda. There are 3 medical departments which are:



Two GSI indexes are created for fetching data from DynamoDB:



Slots and Slot types used in AWS Lex:

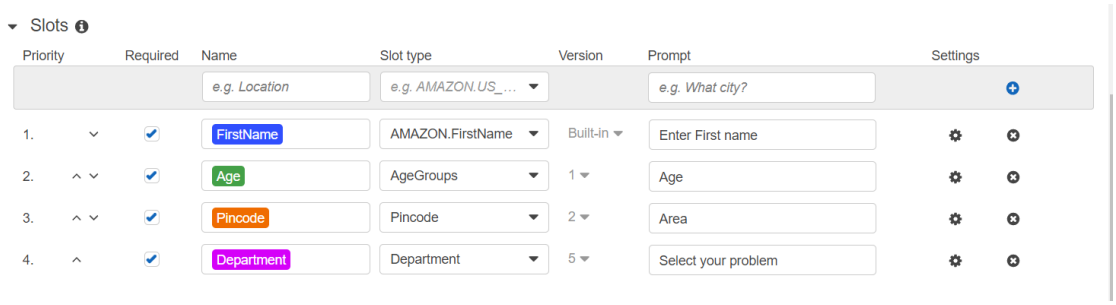
The parameters that indicate the information that the intent needs to fulfill the user's request is known as slots.

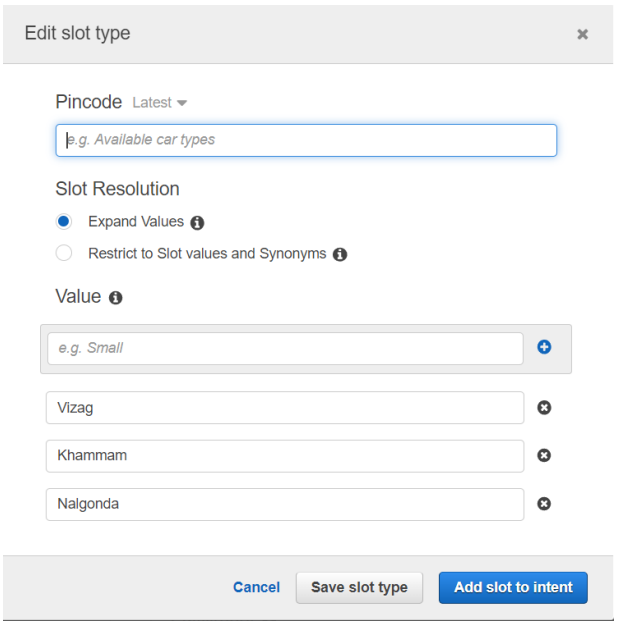
A slot type is a list of values that Amazon Lex uses to train the machine learning model to recognize values for a slot

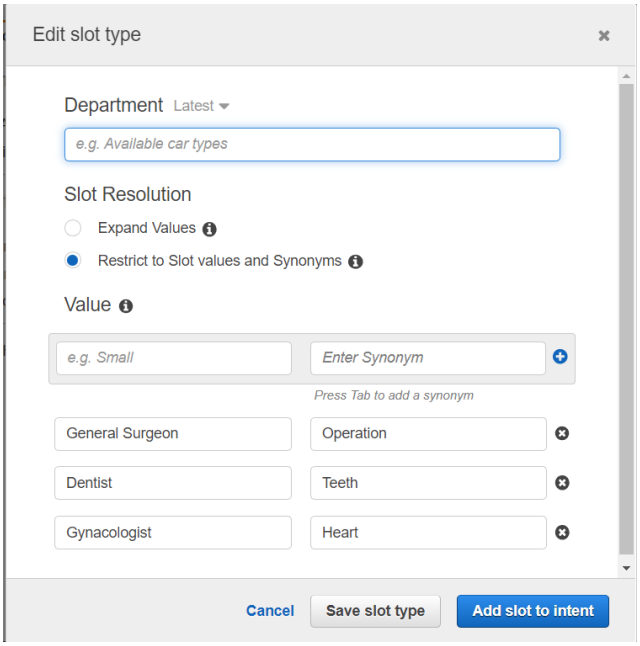
Greetings:

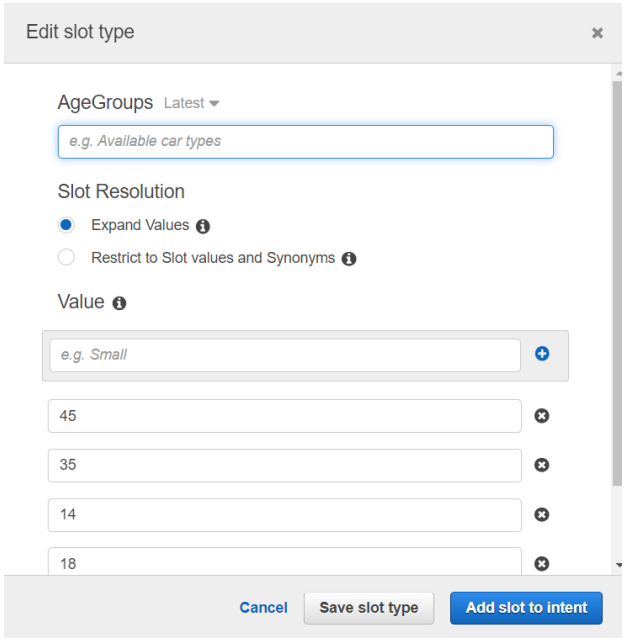
The slots and slotypes in this intent used are:

- **FirstName:** This is the slot that asks and stores the first name of the patient. The SlotType used here is amazon.us_first_name which is a built-in slot type in amazon lex.
- **AgeGroups:** This stores the age of the patient and uses a custom SlotTypeAge Groups.
- **Pincode:** This stores the area chosen by the patient and uses a custom SlotTypePincode.
- **Department:** This stores the type of treatment chosen by the patient and uses a custom SlotType Department.


Slots
Priority
Required
Name
Slot type
Version
Prompt
Settings
e.g. Location
e.g. AMAZON.US_...
e.g. What city?
1. FirstName AMAZON.FirstName Built-in Enter First name
2. Age AgeGroups 1 Age
3. Pincode Pincode 2 Area
4. Department Department 5 Select your problem

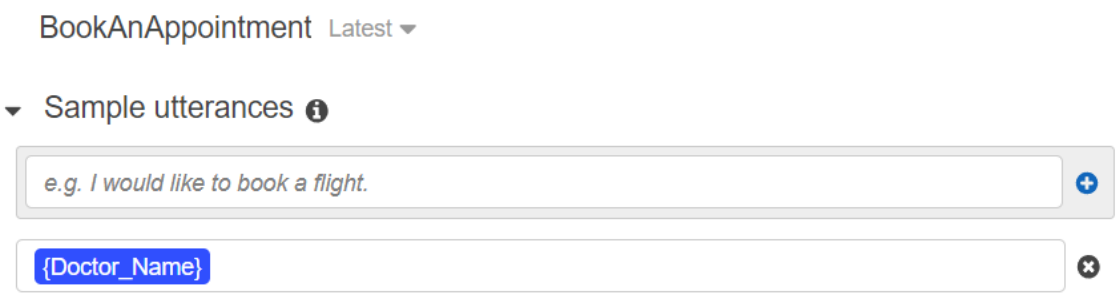

Edit slot type
Pincode Latest
e.g. Available car types
Slot Resolution
Expand Values
Restrict to Slot values and Synonyms
Value
e.g. Small
Vizag
Khammam
Nalgonda
Cancel
Save slot type
Add slot to intent


Edit slot type
Department Latest
e.g. Available car types
Slot Resolution
Expand Values
Restrict to Slot values and Synonyms
Value
e.g. Small
Enter Synonym
Press Tab to add a synonym
General Surgeon
Operation
Dentist
Teeth
Gynacologist
Heart
Cancel
Save slot type
Add slot to intent


Edit slot type
AgeGroups Latest
e.g. Available car types
Slot Resolution
Expand Values
Restrict to Slot values and Synonyms
Value
e.g. Small
45
35
14
18
Cancel
Save slot type
Add slot to intent

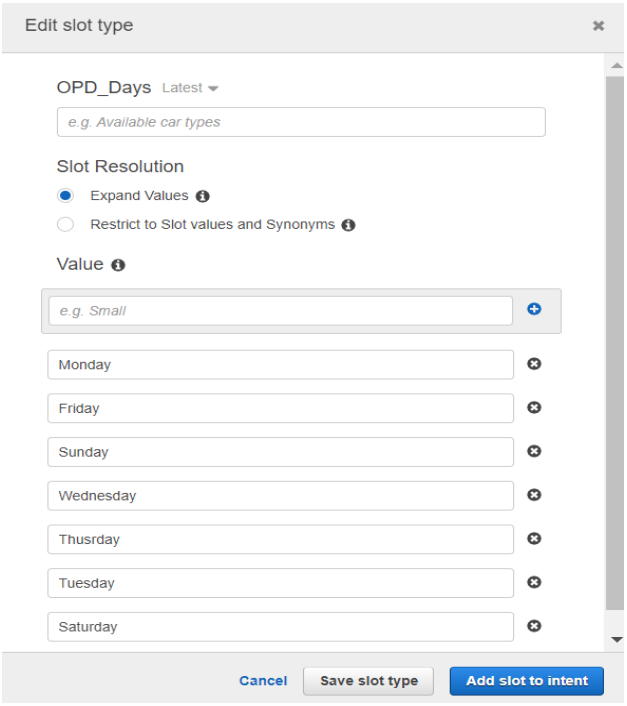
BookAnAppointment:

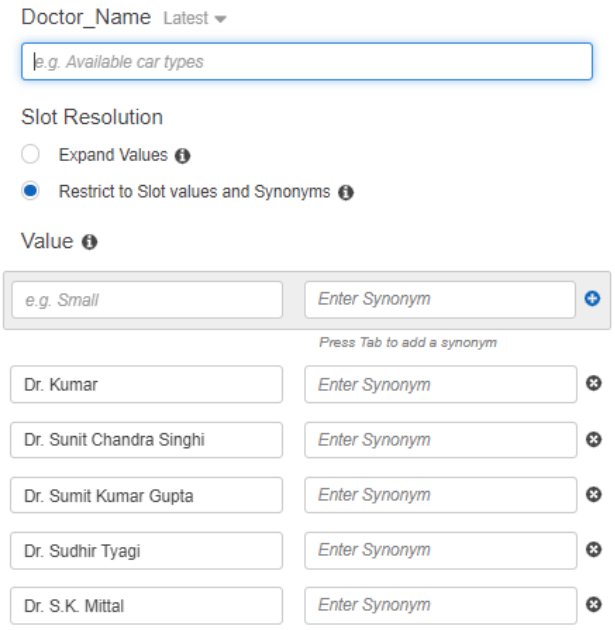
The intent that accompanies the Greetings intent was this. After the Greetings intent is completed, it displays a response Card with the names of the doctors and their availability according to user requirement. And this intent will wait for the user to select a doctor from the answer card's list of possibilities.

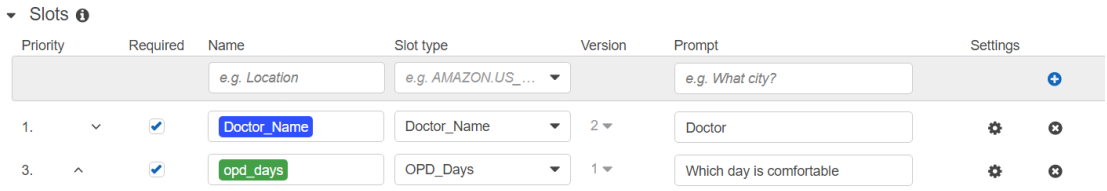


Slot types used:

- **Doctor_Name**: This slot is in capable of waiting for the doctor to be chosen and saving the doctor's name for later usage in the current Intent. It uses a custom-builtslotTypeDoctor_Name.
- **OPD_days:** This contains the user-selected day in relation to the doctor's OPD Days. This uses a custom built slotType called OPD_Days.





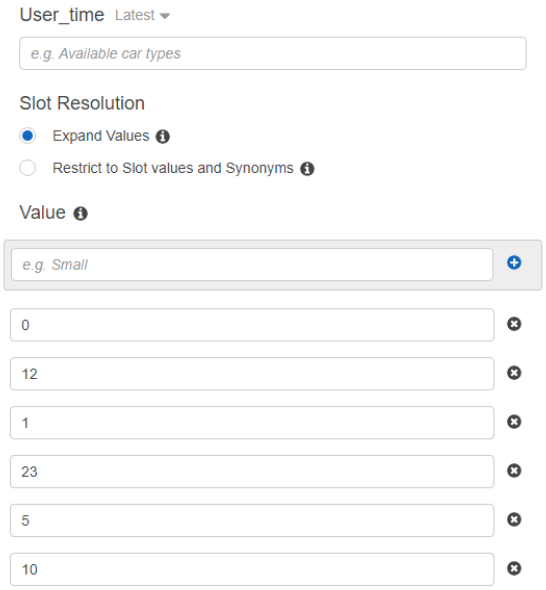


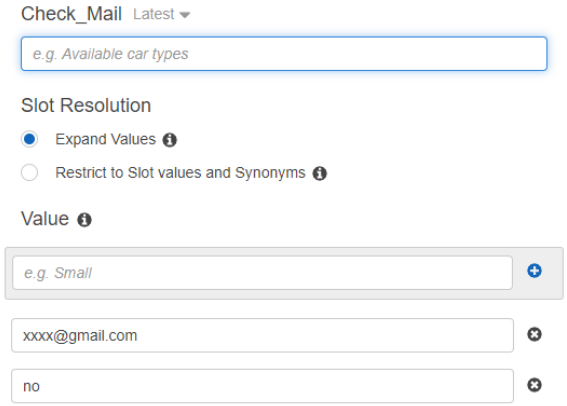
Confirm:

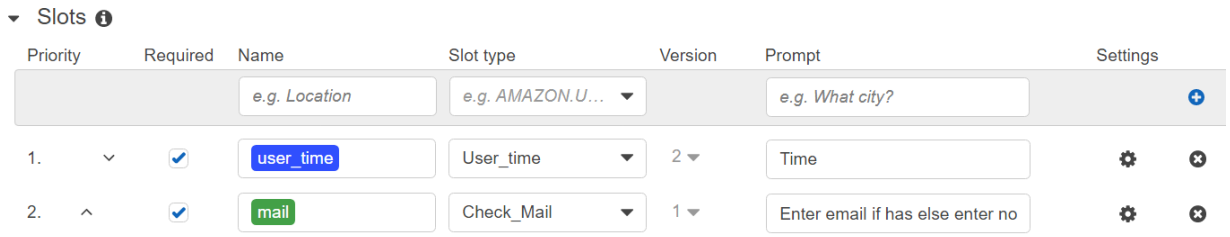
The intent that accompanies the BookAnAppointment intent was this. After the BookAnAppointment intent is completed, it displays a response Card with the selected doctor timings. And this intent will wait for the user to select time from the answer card's list of possibilities.

Slot types used:

- **User_time:**This slot is in capable of enabling patient to schedule a time for booking appointment. It uses a custom-builtslot Type User_time.
- **Check_Mail:**This contains the email ID of the patient in order to send appointment confirmation response to the patient. This uses a custom built slotType called Check_Mail.







Lambda Function

AWS lambda allows you to execute code for any type of application

We can run code in response to certain events from other services like lex, dynamodb.

Supports number of programming languages like Node js, java, c# etc. Here we used python.

The Lambda function is used to integrate many AWS services. Here we choose the Lambda for two major reasons:

- data-processing for AWS services such as Amazon DynamoDB.
- Return and access responses to and from Amazon lex

Libraries used in Lambda function are:

- **boto3**: Boto is the Amazon Web Services (AWS) SDK for Python. It enables Python developers to create, configure, and manage AWS services.
- **json:** Json is used here as all the data transmission in aws is done in json format.
- **smtplib: It creates a** session object that can be used to send mail to any internet machine with an SMTP

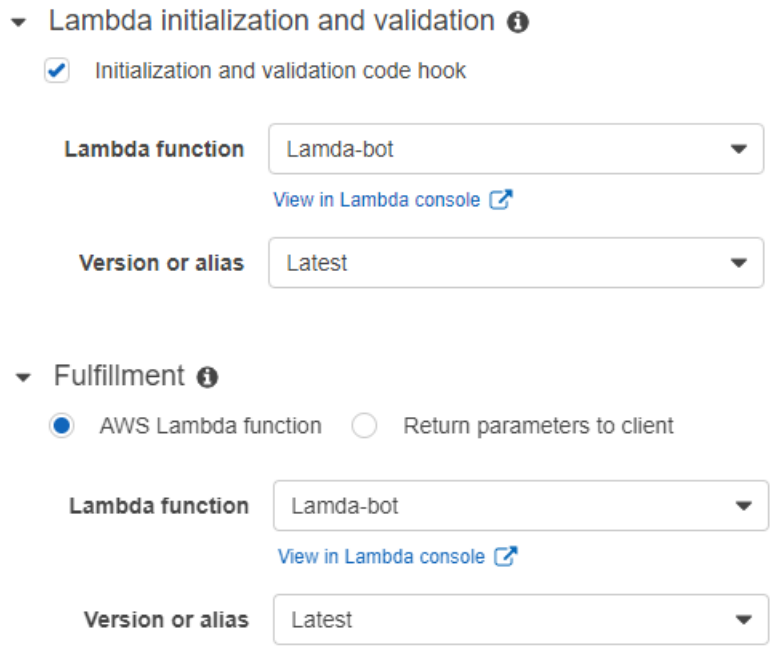
Now for the intents:

Greetings: Lambda function retrieve all the information entered by the user and from dynamodb all the details of the doctors related to the department chosen by the user are extracted and returned all the doctor details as a response card to lex.

BookAnAppointment: The lambda function is used for purpose to show the response card with the available timings of the selected doctor. Hence in this intent lambda function has an integral function of controlling the flow of the dialogue and also for error handling.

Confirm: The lambda function is used for purpose to book an appointment for the selected doctor and give a booking response to the patient via mail. Hence in this intent lambda function has an integral function of controlling the flow of the dialogue and also for error handling.

Enable the Lambda initialization and validation, Fulfillment with the created lambda function for each and every intent.



Sample Snippet of the Code:

```
def check(event):
    p=1
    table = db.Table('Appointment')
    while(p):
        n = random.randint(1,10000)
        a= random.randint(1,10000)
        n=str(n)
        response = table.query(IndexName='UserID-index', KeyConditionExpression=Key('UserID').eq(n))
        if(len(response['Items'])>=1):
            p=1
        else:
            p=0
    email = event["currentIntent"]["slots"]["mail"]
    print(email)
    r=" "
    if email!="no":
        emailing(email,n)
        r=" Check your spam messges in mail for reference"
    return_statement= {
        "dialogAction": {
            "type": "Close",
            "fulfillmentState": "Fulfilled",
            "message": {
                "contentType": "PlainText",
                "content": "Your booking is confirmed, Thank you for contacting your booking id is: "+n+r+"Get Well Soon!"
            }
        }
    }
    table.put_item(Item={'S.NO': a ,'UserID': n,'Email': email},)
    return return_statement
def bookappointment(event):
    docs=[]
    doctor_name = event["currentIntent"]["slots"]["Doctor_Name"]
    day = event["currentIntent"]["slots"]["opd_days"]
    response = table.query(IndexName='Name-index',KeyConditionExpression=Key('Name').eq(doctor_name))
    for j in range(len(response['Items'])):
        if response['Items'][j]['Name'] == doctor_name:
```

Flexibilities with AWS Lambda:

- When you invoke a function, you can choose to invoke it synchronously or asynchronously. With synchronous invocation , you wait for the function to process the event and return a response.
- With asynchronous invocation, Lambda queues the event for processing and returns a response immediately.
- Scalability—According to users request it scales up or down the capacity of the function.
- Concurrency — ensure that a function can scale without fluctuations at simultaneous executions for the function

Working of Lambda function:

- The first time you invoke your function, AWS Lambda creates an instance of the function and runs its handler method to process the event.
- When the function returns a response, it stays active and waits to process additional events
- If you invoke the function again while the first event is being processed, Lambda initializes another instance, and the function processes the two events concurrently.
- As more events come in, Lambda routes them to available instances and creates new instances as needed.
- When the number of requests decreases, Lambda stops unused instances to free up scaling capacity for other functions.

AWS CloudFormation Deployment

CloudFormation:

AWS CloudFormation is an AWS service that uses template files to automate the setup of AWS resources.

Deployment speed, Scaling up, Easy updates, Security are the key features of Cloud Formation.

Deploying the chatbot using CloudFormation:

1. Click the LaunchStack for the Region in which you created chatbot

Northern Virginia - LaunchStack

Oregon- LaunchStack

Ireland- LaunchStack

Sydney- LaunchStack

Singapore- LaunchStack

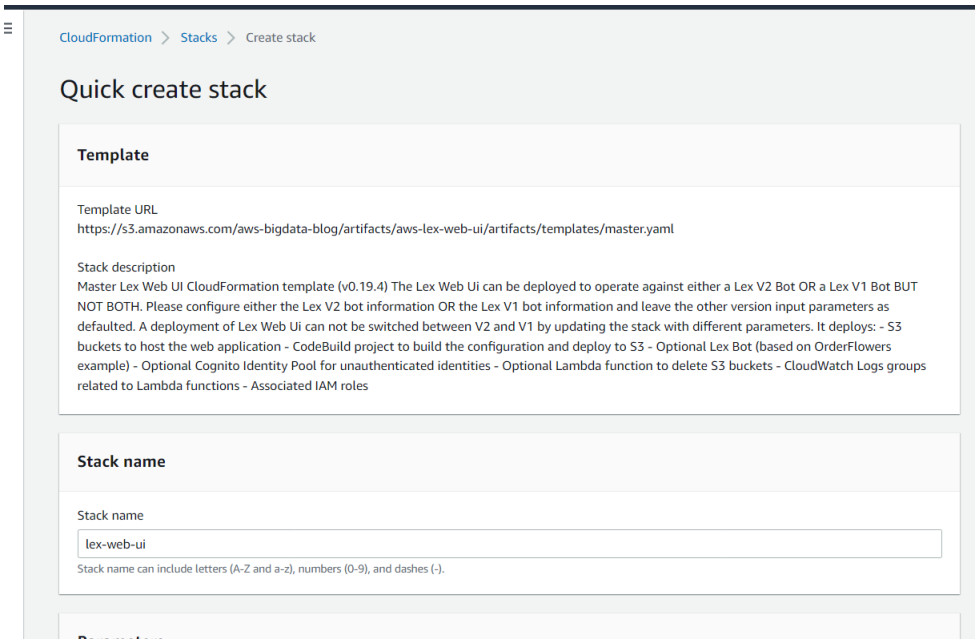
London-LaunchStack

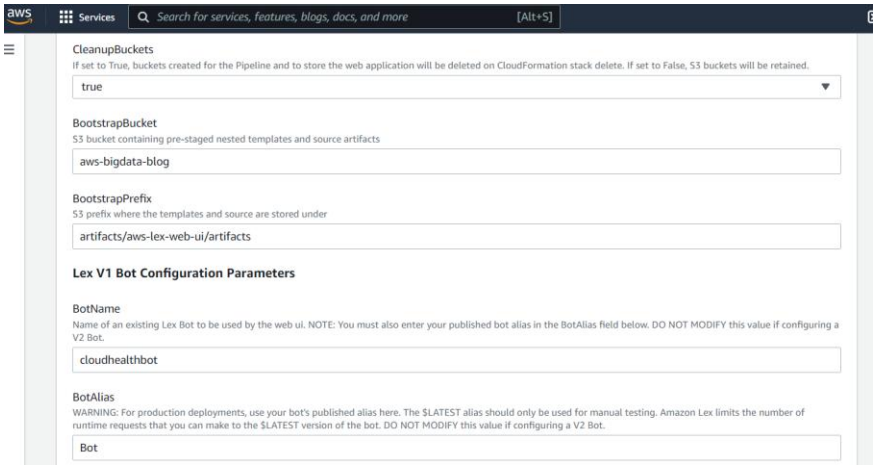
Tokyo- LaunchStack

Frankfurt- LaunchStack

2. It directs to a stack creation page.

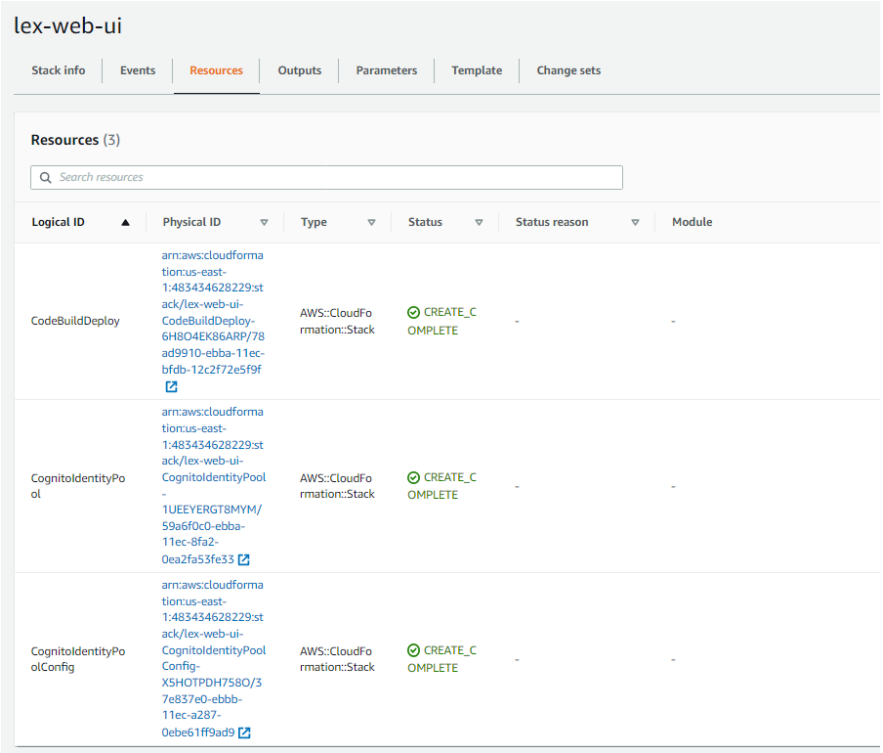
- By default a yaml template is created in s3.
- In the stack creation page give details regarding your bot.
- Modify all the changes required i.e. Bot initial speech
- Create a stack



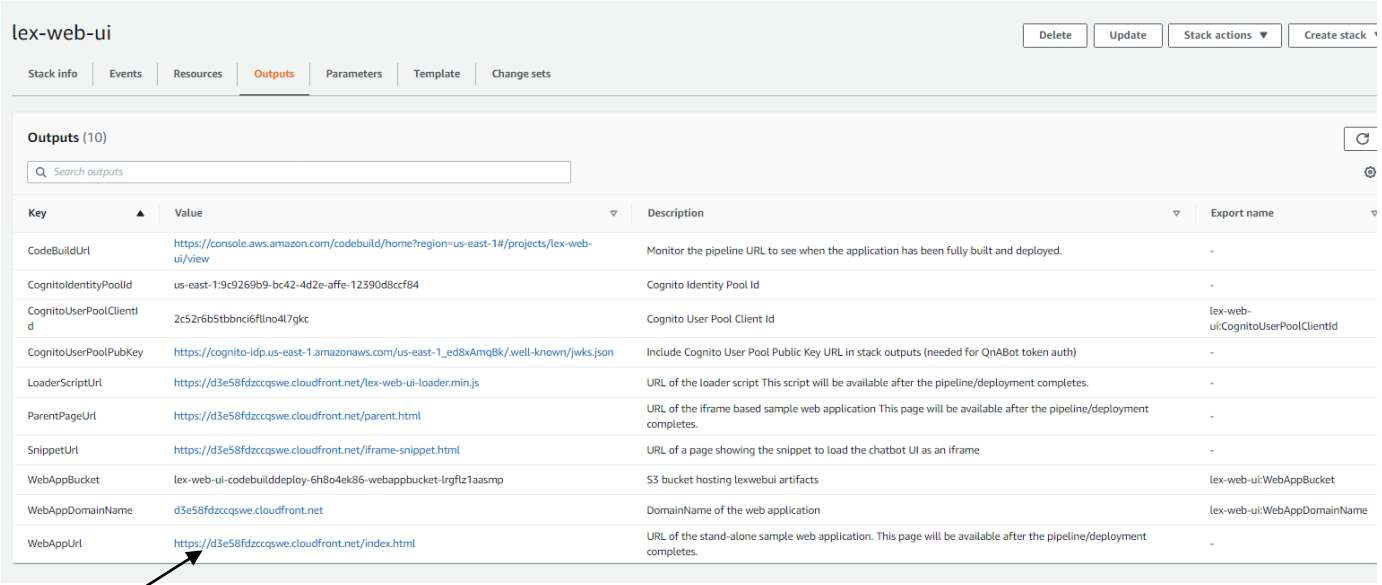


3. Once stack is created it creates resources which generates cognito poll identity..etc.

It also creates some nested stacks. Once all the resources were created we can access our chat bot.



Step4: By using WebAppUrl access our deployed chatbot.



Click Here to access the chatbot

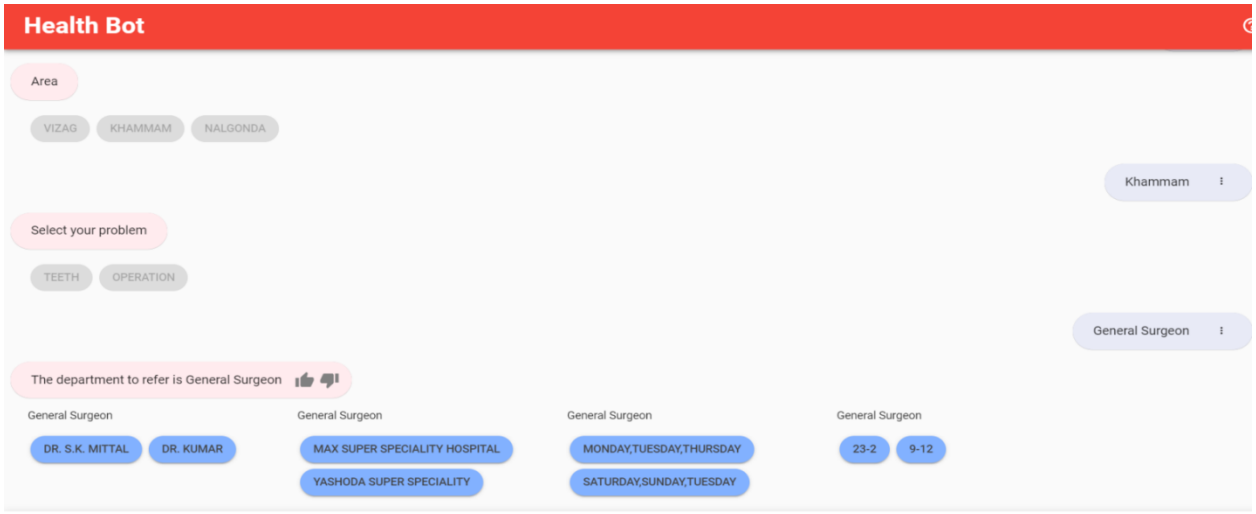
Sample Outputs and Future Scope

Chat Bot Usage steps:

- User enter his/her name
- User enter his age
- User selects an area according to his/her convenient.
- User selects the type of treatment needed.
- User selects a doctor
- User selects a day on he/she want to book an appointment.
- User selects a time at which doctor is available
- User provides an email id to get confirmation responses
- On successful booking user receives a mail with a booking ID.

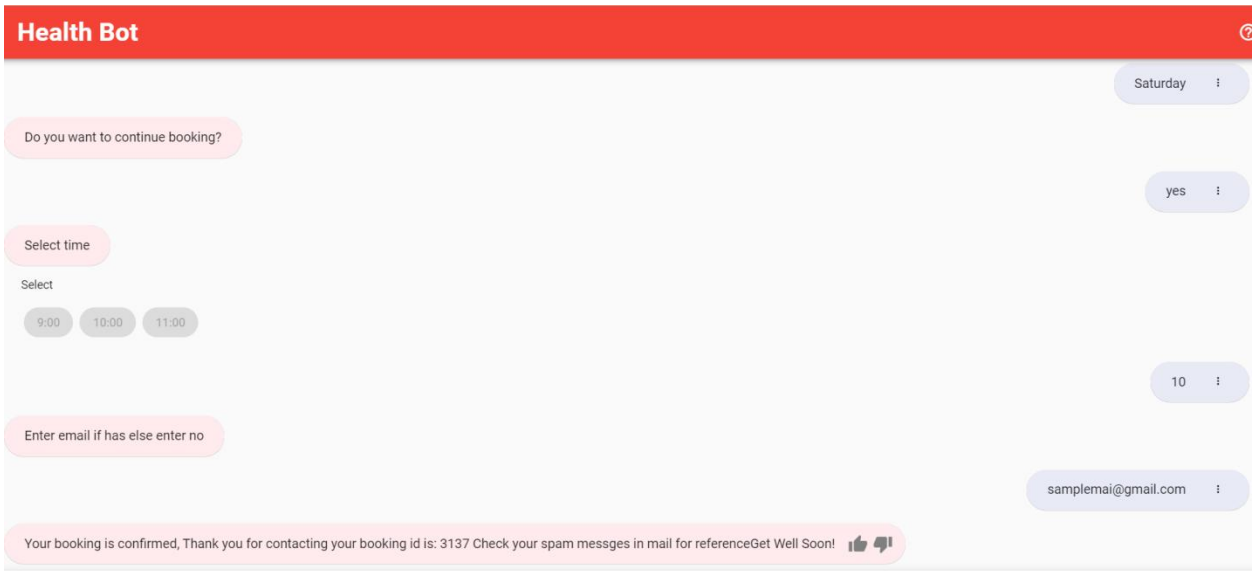
Successful Case:

A user named lex selected area khammam

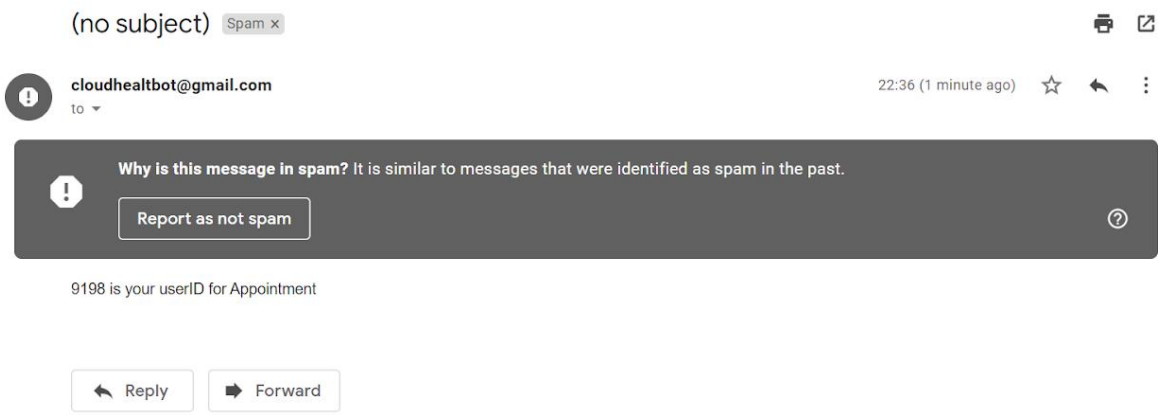


He need surgery and has chosen Dr. Kumar as his doctor.

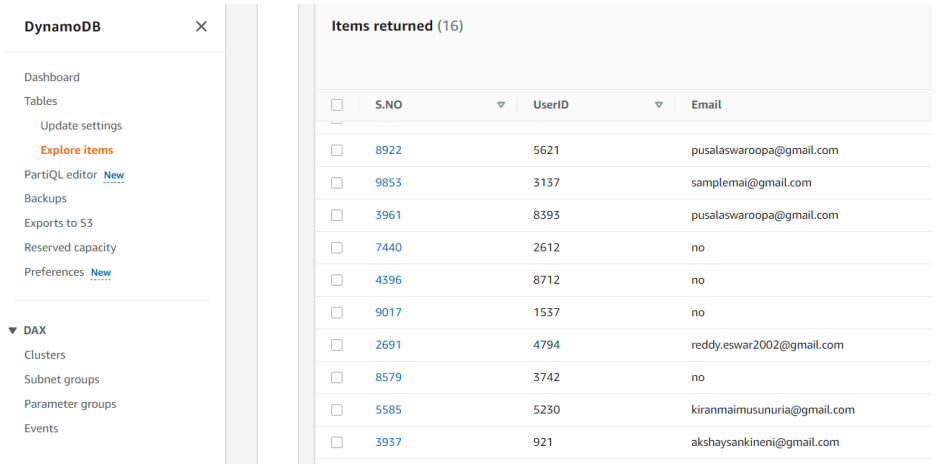
He chose Saturday as the day for his treatment. As doctor is available on that day he proceeded to book an appointment.



As booking is successful appointment id is sendto entered mail id.



All the successful booking information is updated on dynamodb appointment table.

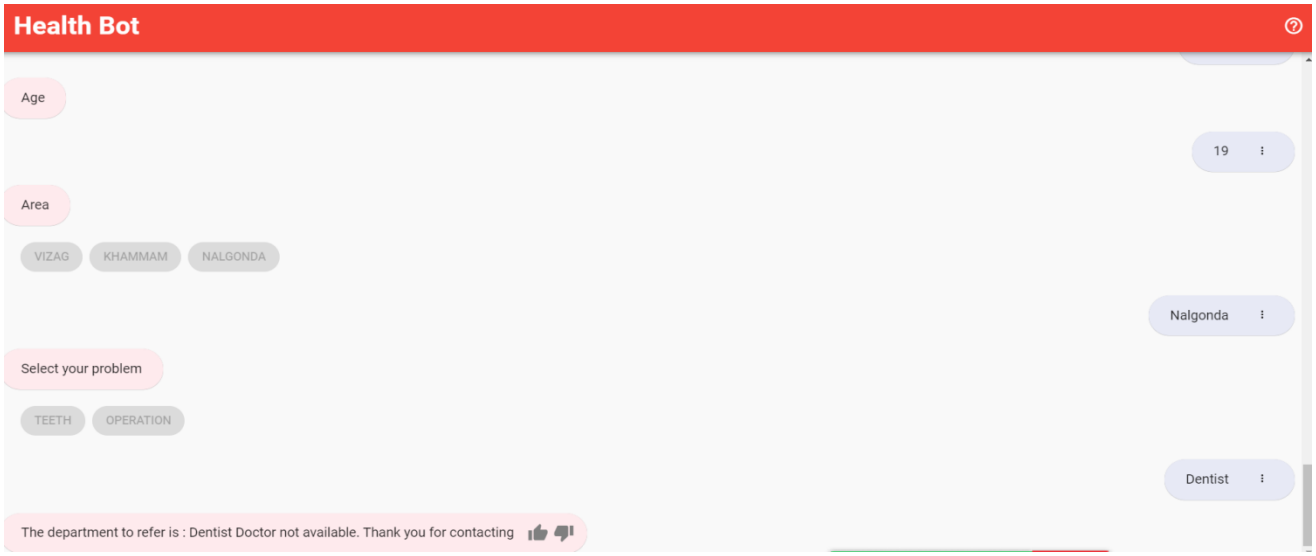


Failure Case:

1. When that specific doctor is not available in the selected area

A user selected nalgonda for his teeth treatment.

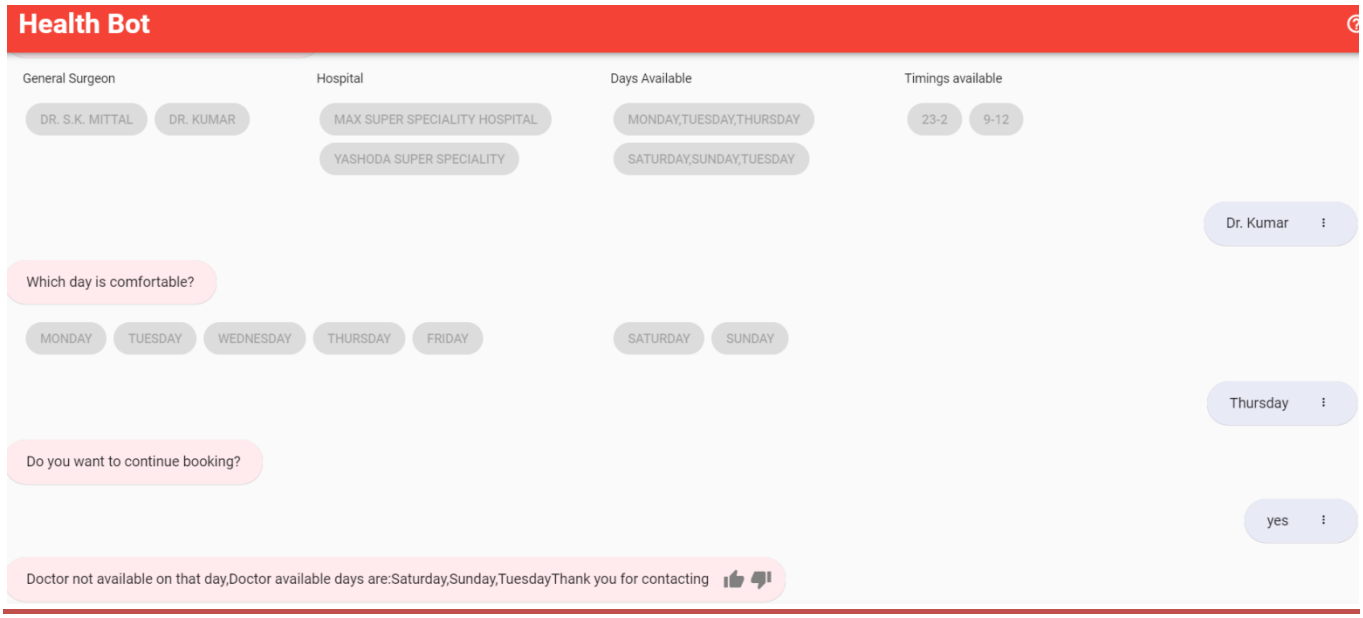
As doctor related to that department is not available further can't proceed with booking. A response is received as Doctor not available.



2. When doctor is not available on the selected day.

A patient need surgery and has chosen Dr. Kumar as his doctor.

Patient chose Thursday as the day for his treatment. As doctor is not available on that day further can't proceed with booking. A response is received as Doctor not available on that day and a display of all the days doctor available.



Future Scope:

- This chat bot allows you to effortlessly access multiple hospitals in various places and arrange an appointment.
- This chat bot allows you to verify the availability of doctors at various hospitals.
- Chat bot keep patients engaged 24/7. Can make it user-friendly using AI/ML Algorithms.
- This Chat bot helps to place an order for medicines as well as andelivery assistance.

Problems Faced

1. Failed accessing DynamoDB and Lex from Lambda function

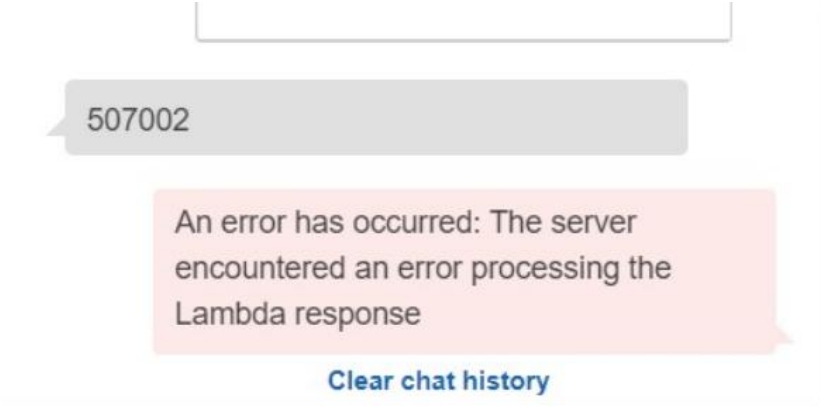
Reason: Not created respective roles

Solution: Enabling permissions from lex to dynamodb using IAM role.

2. Failed returning responses from Lambda function to Lex

Reason: Unchecked test cases

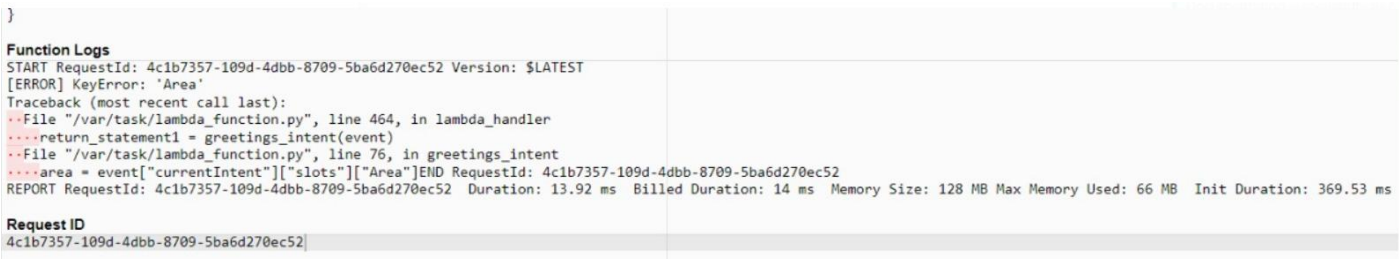
Solution: Checking all the possibilities for booking



3. Failed executing code

Reason: Referred to wrong slot type

Solution: Corrected code according to slot type



REFERENCES

1.Using DynamoDB from AWS Lambda Retrieved from

https://www.tutorialspoint.com/aws_lambda/aws_lambda_using_lambda_function_with_amazon_dynamodb.htm

2. AWS Cloud Formation Deployment Retrieved from https://aws.amazon.com/blogs/machine-learning/deploy-a-web-ui-for-your-chatbot/

3. AWS Cloud Formation Deployment Retrieved from

https://docs.aws.amazon.com/lexv2/latest/dg/lambda.html

4. AWS Lex Chatbot Creation Retrieved from:

Part1: https://www.youtube.com/watch?v=VaWk49fCMQY

Part2: https://www.youtube.com/watch?v=IouIOspXFq8