Capstone Project Weekly Progress Report

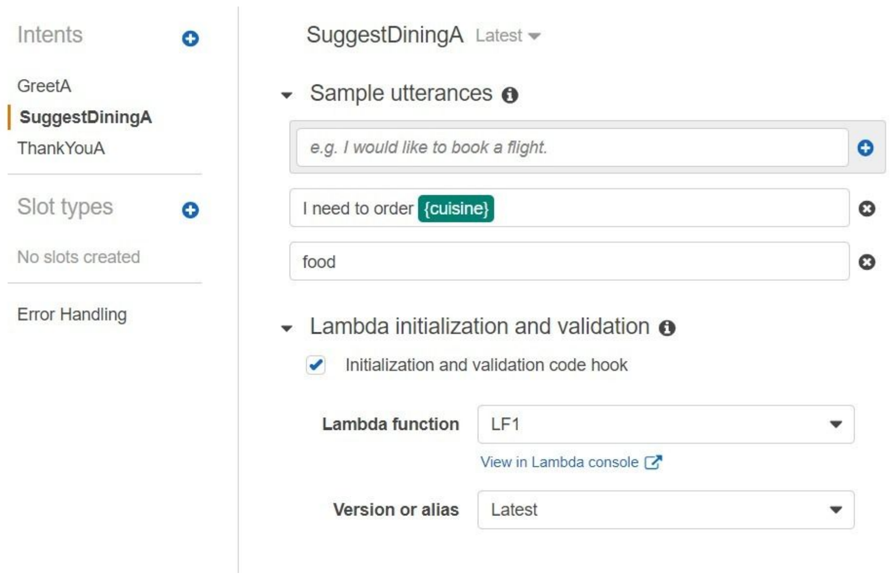
Project Title	MEALBUDDY
Group Name	GROUP_G
Student names/Student IDs	ELVIN IYPE MATHEW C0769974 ELDA VARGHESE C0769741 TOM JOSEPH C0760915 JEENA HELEN FRANCIS C0764493 CHINJU BABY C0769912
Reporting Week	21 JUNE 2020 - 27 JUNE 2020
Faculty Supervisor	WILLIAM POURMAJIDI

1. Tasks Outlined in Previous Weekly Progress Report

- **Will be solving CORS issue for api client and api gateway**
- **Creating SQS for receiving the messages from LF1.**
- **Connect lambda(LF1) to lex.**
- **Access data from lambda(LF1) through lex and vice versa**
- **Will be sharing a architecture diagram**

2. Progress Made in Reporting Week

- **Solving the CORS issue for api client and api gateway for simple javascript, enough documentation was not available from aws instead they suggest aws amplify. Need to explore other options like https://aws.amazon.com/amplify/and https://www.kommunicate.io/**
- **From LF1, we took the data and gave the chat bot an automatic reply according to the user input. The user inputs given to chat bot can be fetched by LF1. It is done by enabling the lambda function in Lex.**
- **LF1 github link :https://github.com/tom5167/mealbuddy/blob/master/LAMBDA/LF1/LF1.py**
- **Connect Lex chatbot with lambda function LF1 as a code hook which essentially entails the invocation of our lambda before lex responds to any of our requests.**



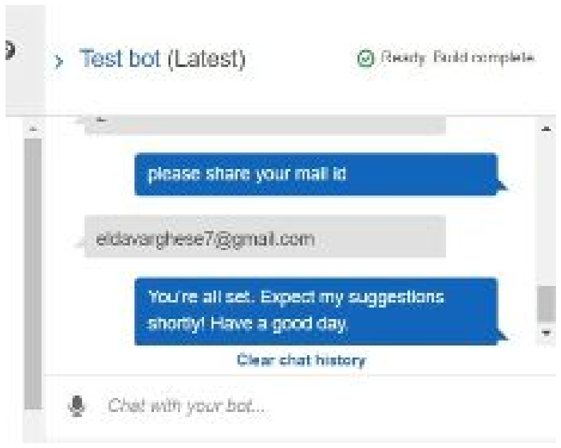
I need to order {cuisine}

food

✓ Initialization and validation code hook

Lambda function: LF1

Version or alias: Latest



please share your mail id

eldavarghese7@gmail.com

You're all set. Expect my suggestions shortly! Have a good day.

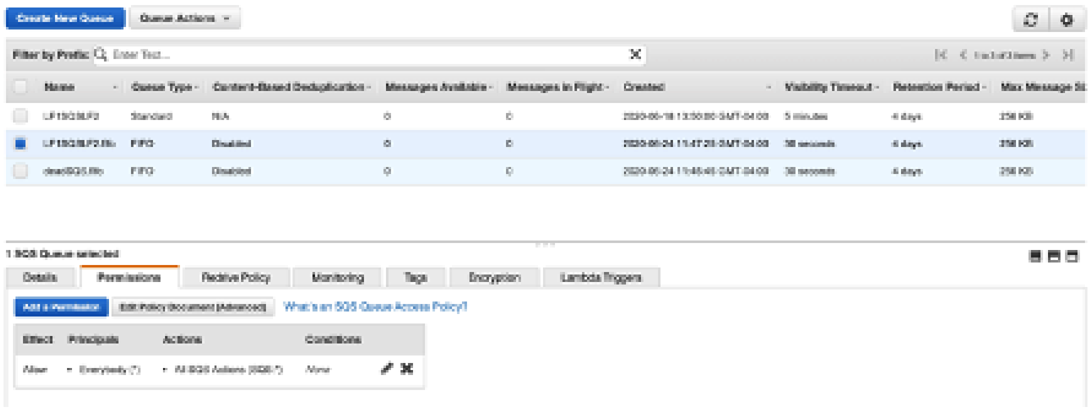
- **Learned about Simple Queue Service (SQS)**
  - ❑ **SQS is a fully managed message queuing service that enables you to decouple and scale microservices, distributed systems, and serverless applications. SQS eliminates the complexity and overhead associated with managing and operating message oriented middleware, and empowers developers to focus on differentiating work.**
  - ❑ **Amazon SQS works on a massive scale, processing billions of messages per day. You can scale the amount of traffic you send to Amazon SQS up or down without any configuration.**
    1. **Choose SQS service**
    2. **Choose Create New Queue.**
    3. **On the Create New Queue page, ensure that you're in the correct region and then type the Queue Name.**

       [The name of a FIFO queue must end with the .fifo suffix.]
    4. **Standard is selected by default. Choose FIFO.**

5. **Create your queue.**

- **Created fifo sqs 'LF1SQSLF2.fifo' to receive messages from lambda 'LF1 '.FIFO sqs strictly maintain the order in which the messages arrive while the default standard sqs occasionally deliver messages in order.**

- **Added 'All sqs action' permission to sqs .**



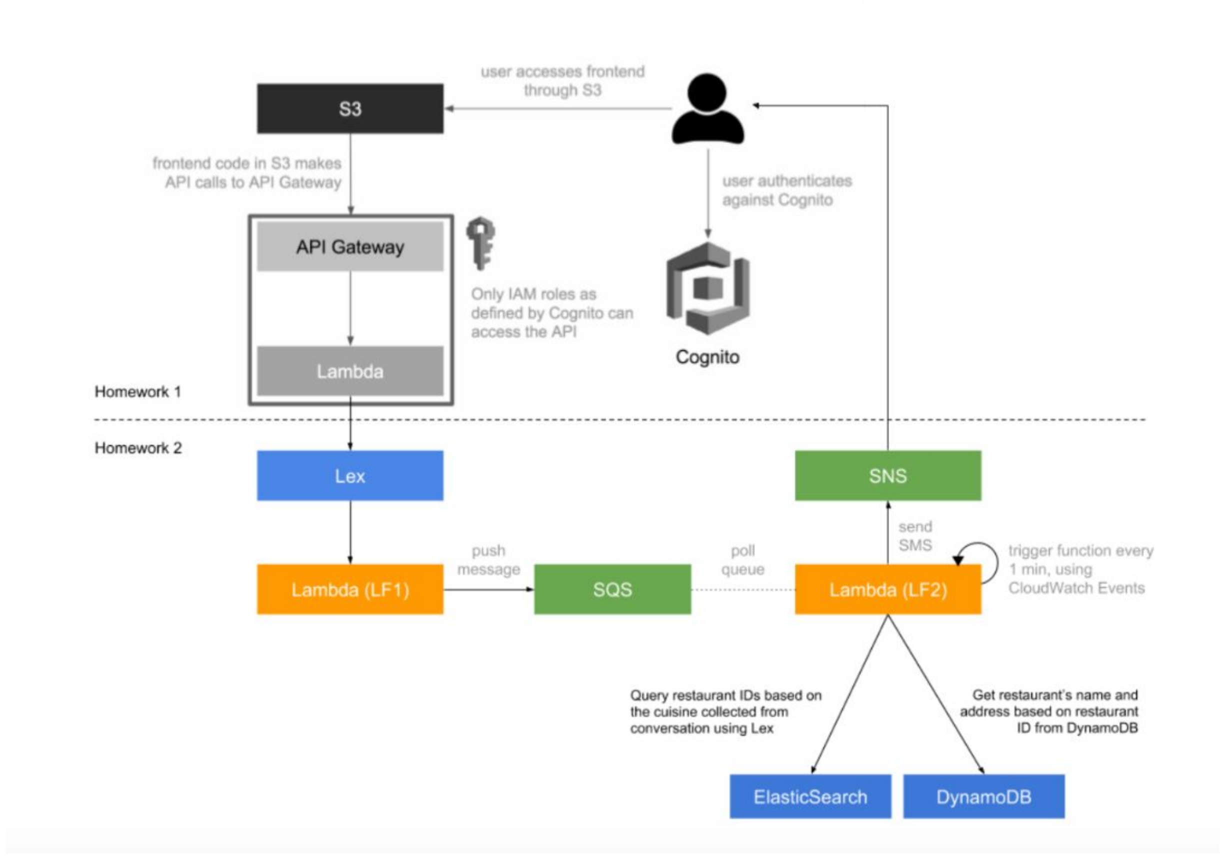
	Name	Queue Type	Content-Based Deduplication	Messages Available	Messages in Flight	Created	Visibility Timeout	Retention Period	Max Message Size
	LF1SQSLF2	Standard	N/A	0	0	2020-06-18 13:50:00 GMT-04:00	5 minutes	4 days	256 KB
	LF1SQSLF2.fifo	FIFO	Disabled	0	0	2020-06-24 11:47:25 GMT-04:00	30 seconds	4 days	256 KB
	deadSQS.fifo	FIFO	Disabled	0	0	2020-06-24 11:45:45 GMT-04:00	30 seconds	4 days	256 KB

1 SQS Queue selected

Effect	Principals	Actions	Conditions
Allow	• Everybody (*)	• All SQS Actions (SQS:*)	None

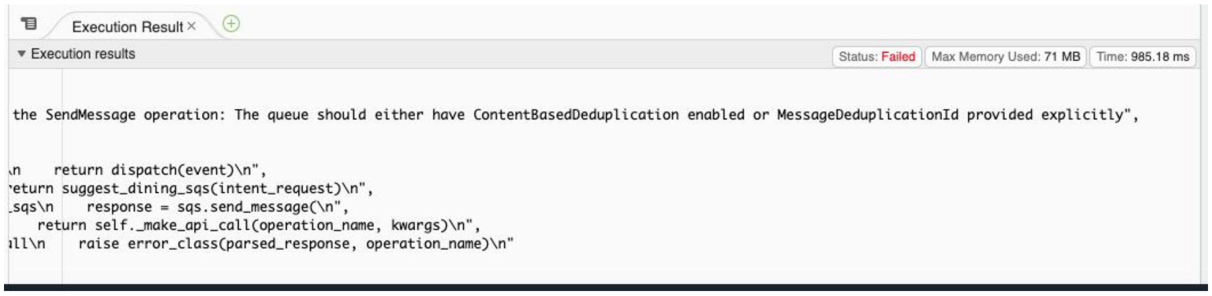
- **Added dead letter sqs 'deadSQS.fifo' for isolated messages that cannot be processed correctly.**

- ## Architecture diagram


user accesses frontend through S3
S3
frontend code in S3 makes API calls to API Gateway
API Gateway
Lambda
Only IAM roles as defined by Cognito can access the API
user authenticates against Cognito
Cognito
Homework 1
Homework 2
Lex
SNS
send SMS
trigger function every 1 min, using CloudWatch Events
Lambda (LF1)
push message
SQS
poll queue
Lambda (LF2)
Query restaurant IDs based on the cuisine collected from conversation using Lex
Get restaurant's name and address based on restaurant ID from DynamoDB
ElasticSearch
DynamoDB

3. **Difficulties Encountered in Reporting Week**

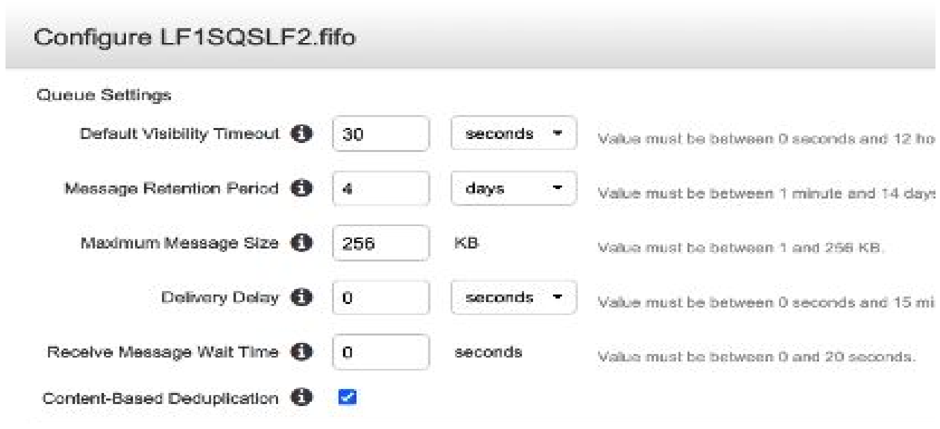
- **Connecting lambda to chat bot. Fetching data from lambda to Lex**
- **Got the error message 'the queue should have content based deduplication enabled' while sending message from 'LF1' to SQS**



```
the SendMessage operation: The queue should either have ContentBasedDeduplication enabled or MessageDeduplicationId provided explicitly",

\n    return dispatch(event)\n",
eturn suggest_dining_sqs(intent_request)\n",
sqs\n    response = sqs.send_message(\n",
   return self._make_api_call(operation_name, kwargs)\n",
ll\n    raise error_class(parsed_response, operation_name)\n"
```

This issue is resolved by configuring the queue by checking 'Content based deduplication'



Configure LF1SQSLF2.fifo

Queue Settings

Default Visibility Timeout	30	seconds	Value must be between 0 seconds and 12 ho
Message Retention Period	4	days	Value must be between 1 minute and 14 days
Maximum Message Size	256	KB	Value must be between 1 and 256 KB.
Delivery Delay	0	seconds	Value must be between 0 seconds and 15 mi
Receive Message Wait Time	0	seconds	Value must be between 0 and 20 seconds.
Content-Based Deduplication	✓		

4. **Tasks to Be Completed in Next Week**

- **SQS-LAMDA (LF2) INTEGRATION - Create a new Lambda function (LF2) that acts as a queue worker. Whenever it is invoked it pulls a message from the SQS queue (Q1).**
- **LF2 - elastic search and dynamo db integration study**
- **Explore https://aws.amazon.com/amplify/and https://www.kommunicate.io/**