Capstone Project Weekly Progress Report

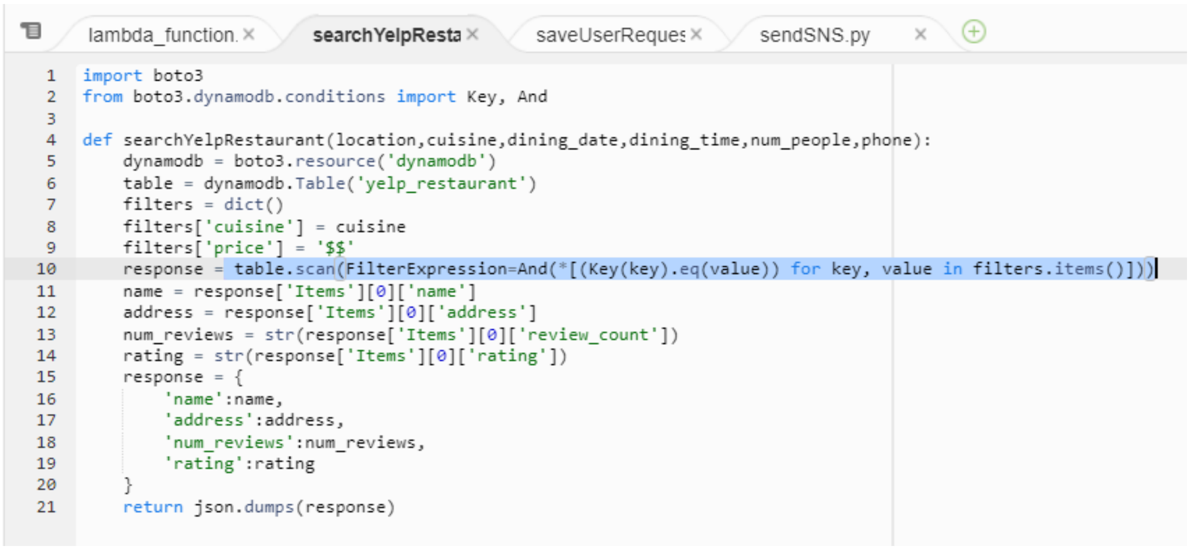
Project Title	MEALBUDDY
Group Name	GROUP_G
Student names/Student IDs	ELVIN IYPE MATHEW C0769974 ELDA VARGHESE C0769741 TOM JOSEPH C0760915 JEENA HELEN FRANCIS C0764493 CHINJU BABY C0769912
Reporting Week	05 JULY 2020 - 11 JULY 2020
Faculty Supervisor	WILLIAM POURMAJIDI

1. Tasks Outlined in Previous Weekly Progress Report
`

- LF2 query from dynamo db
- LF2 SNS for sending message notification to cell phone
- Create Test Case Document for various test case scenarios
- LEX - LF1 - SQS- LF2 flow check and resolve issues if any

2. Progress Made in Reporting Week

- LF2- new Lambda function that acts as a queue worker.Whenever it is invoked it
  a. pulls a message from the SQS queue (LF1SQSLF2)
  b. gets a random restaurant recommendation for the cuisine collected through Lex chatbot conversation from DynamoDB table called yelp-restaurant



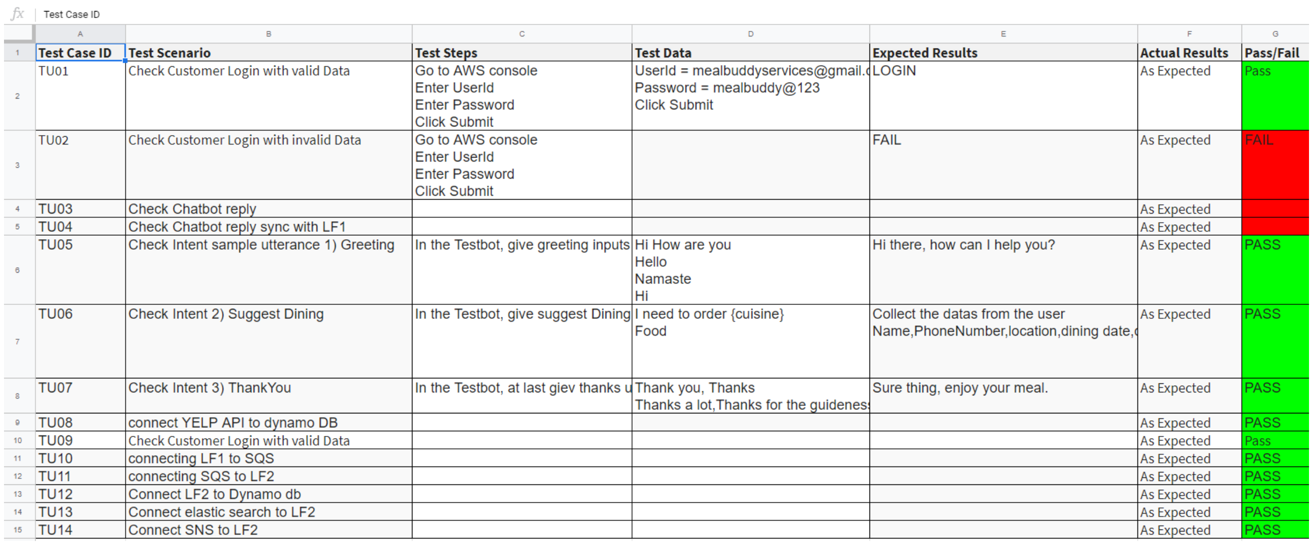
```python
import boto3
from boto3.dynamodb.conditions import Key, And

def searchYelpRestaurant(location,cuisine,dining_date,dining_time,num_people,phone):
    dynamodb = boto3.resource('dynamodb')
    table = dynamodb.Table('yelp_restaurant')
    filters = dict()
    filters['cuisine'] = cuisine
    filters['price'] = '$$'
    response = table.scan(FilterExpression=And(*[(Key(key).eq(value)) for key, value in filters.items()]))
    name = response['Items'][0]['name']
    address = response['Items'][0]['address']
    num_reviews = str(response['Items'][0]['review_count'])
    rating = str(response['Items'][0]['rating'])
    response = {
        'name':name,
        'address':address,
        'num_reviews':num_reviews,
        'rating':rating
    }
    return json.dumps(response)
```

- **The Scan operation returns one or more items and item attributes by accessing every item in a table or a secondary index.**

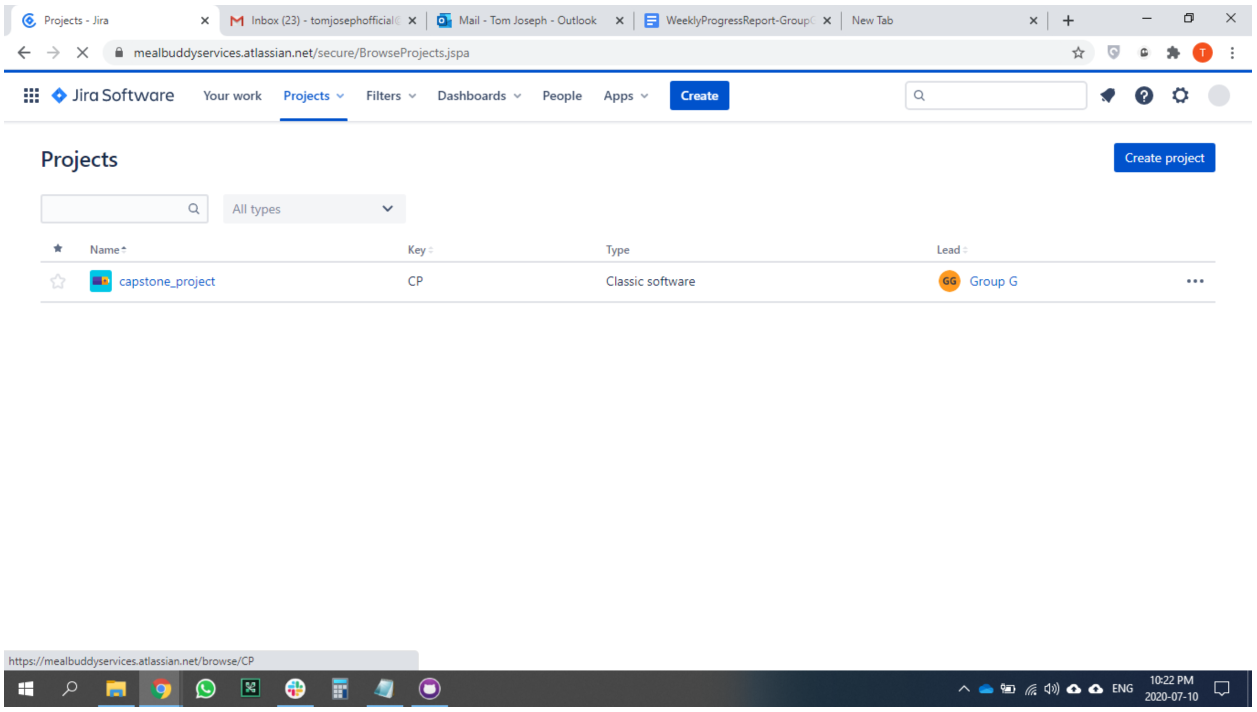
- **To have DynamoDB return fewer items, you can provide a FilterExpression operation.**

- **Test Case Document - https://docs.google.com/spreadsheets/d/1kHwH_tbdVjxfRmWQiNDvby0lLT5d4tLE0X97xIctRlY/edit?usp=sharing**


fx Test Case ID

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Results	Actual Results	Pass/Fail
TU01	Check Customer Login with valid Data	Go to AWS console Enter UserId Enter Password Click Submit	UserId = mealbuddyservices@gmail.c Password = mealbuddy@123 Click Submit	LOGIN	As Expected	Pass
TU02	Check Customer Login with invalid Data	Go to AWS console Enter UserId Enter Password Click Submit		FAIL	As Expected	FAIL
TU03	Check Chatbot reply				As Expected	🟥
TU04	Check Chatbot reply sync with LF1				As Expected	🟥
TU05	Check Intent sample utterance 1) Greeting	In the Testbot, give greeting inputs	Hi How are you Hello Namaste Hi	Hi there, how can I help you?	As Expected	PASS
TU06	Check Intent 2) Suggest Dining	In the Testbot, give suggest Dining	I need to order {cuisine} Food	Collect the datas from the user Name,PhoneNumber,location,dining date,c	As Expected	PASS
TU07	Check Intent 3) ThankYou	In the Testbot, at last giev thanks u	Thank you, Thanks Thanks a lot,Thanks for the guidenes	Sure thing, enjoy your meal.	As Expected	PASS
TU08	connect YELP API to dynamo DB				As Expected	PASS
TU09	Check Customer Login with valid Data				As Expected	Pass
TU10	connecting LF1 to SQS				As Expected	PASS
TU11	connecting SQS to LF2				As Expected	PASS
TU12	Connect LF2 to Dynamo db				As Expected	PASS
TU13	Connect elastic search to LF2				As Expected	PASS
TU14	Connect SNS to LF2				As Expected	PASS

- **we have made different test cases, check customer with login credentials , check chatbot reply,**
- **whether the phone number entered by user is valid or not and so on**
- **We will move test cases soon to Jira to be more cloud oriented**


Projects - Jira
Inbox (23) - tomjosephofficial
Mail - Tom Joseph - Outlook
WeeklyProgressReport-Group
New Tab
mealbuddyservices.atlassian.net/secure/BrowseProjects.jspa
Jira Software
Your work
Projects
Filters
Dashboards
People
Apps
Create

Projects

Create project

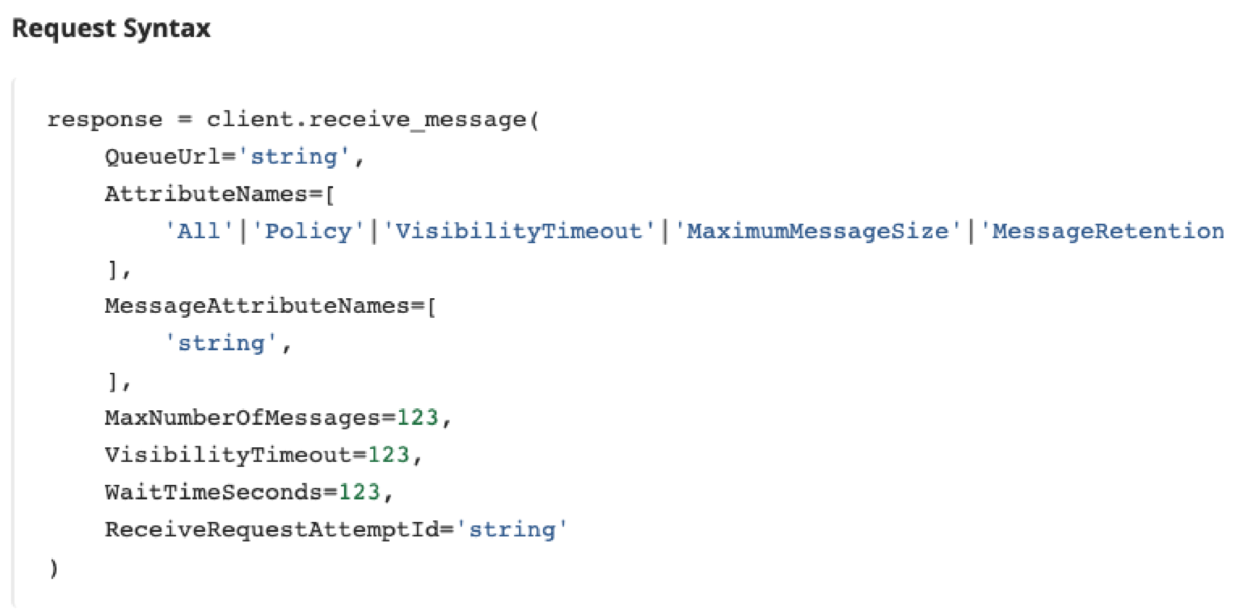
All types

	Name	Key	Type	Lead
☆	capstone_project	CP	Classic software	Group G

https://mealbuddyservices.atlassian.net/browse/CP
ENG
10:22 PM
2020-07-10

-

- **For integrating SQS to LF2 lambda we have used sqs.receive_message method .This method retrieves one or more messages (up to 10), from the specified queue.**

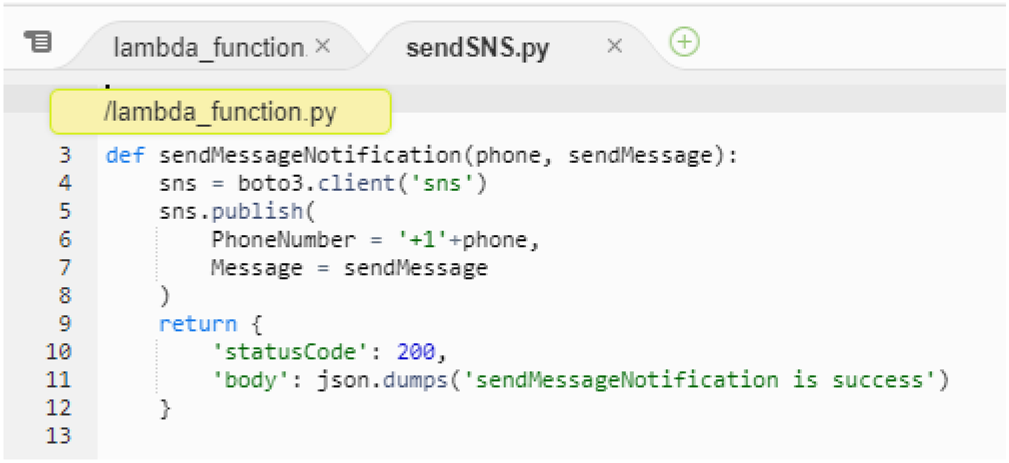


Request Syntax

```
response = client.receive_message(
    QueueUrl='string',
    AttributeNames=[
        'All'|'Policy'|'VisibilityTimeout'|'MaximumMessageSize'|'MessageRetention
    ],
    MessageAttributeNames=[
        'string',
    ],
    MaxNumberOfMessages=123,
    VisibilityTimeout=123,
    WaitTimeSeconds=123,
    ReceiveRequestAttemptId='string'
)
```

-
- **QueueUrl-The URL of the Amazon SQS queue from which messages are received.**
- **AttributeNames- we have used ‘All’ - Returns all values.**
- **MaxNumberOfMessages (*integer*) -- The maximum number of messages to return. Amazon SQS never returns more messages than this value (however, fewer messages might be returned). Valid values: 1 to 10. Default: 1.**

- **Amazon Simple Notification Service (SNS) is a highly available, durable, secure, fully managed pub/sub messaging service that enables you to decouple microservices, distributed systems, and serverless applications.**
- **Amazon SNS provides topics for high-throughput, push-based, many-to-many messaging. Using Amazon SNS topics, your publisher systems can fan out messages to a large number of subscriber endpoints for parallel processing, including Amazon SQS queues, AWS Lambda functions, and HTTP/S webhooks. Additionally, SNS can be used to fan out notifications to end users using mobile push, SMS, and email.**
- **With Amazon SNS, we tried sending SMS (text) messages to the client's phone.**
- **For integrating SNS to LF2 lambda we have used sqs.publish method. This method will send just one message to the designated phone number.**



```
def sendMessageNotification(phone, sendMessage):
    sns = boto3.client('sns')
    sns.publish(
        PhoneNumber = '+1'+phone,
        Message = sendMessage
    )
    return {
        'statusCode': 200,
        'body': json.dumps('sendMessageNotification is success')
    }

```

-
- **The sendMessage has ("Hello! For {}, we recommend the {} {} restaurant on {}. The place has {} of reviews and an average score of {} on Yelp. Enjoy!".format(location, name, cuisine, address, num_reviews, rating)).**

3. Difficulties Encountered in Reporting Week

- **Had problems with syntax and data format**

4. Tasks to Be Completed in Next Week

- **Use https://www.kommunicate.io/ and link AWS LEX**
- **modularize lambda function LF1&LF2 for easy upgrades**
- **UI - LEX - LF1 - SQS- LF2 flow check and resolve issues if any**
- **Test Case Document Update and add new scenarios**