Customer Support Call Intelligence and Analytics

DAMG 7245 - Spring 2022 - Team 7

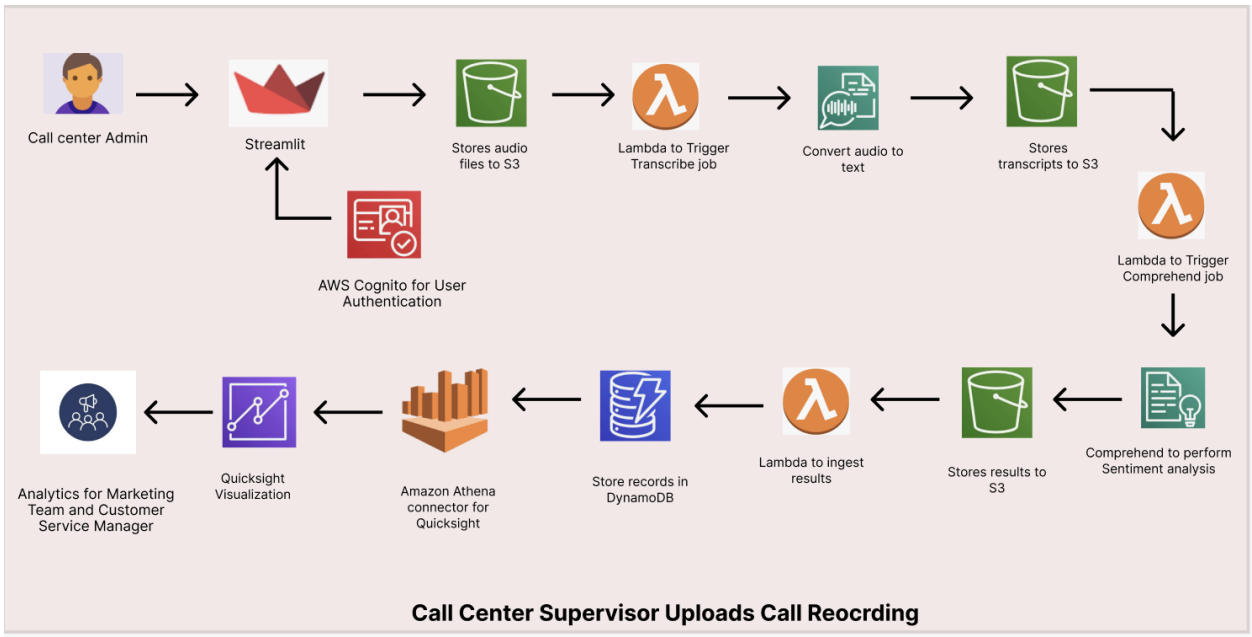
NAME	NUID
Greeshma Tatineni	002130050
Swathi Sharma	002105916
Kashyap Datta Kovvali	001098717
Bala Yeshwanth M	002191719

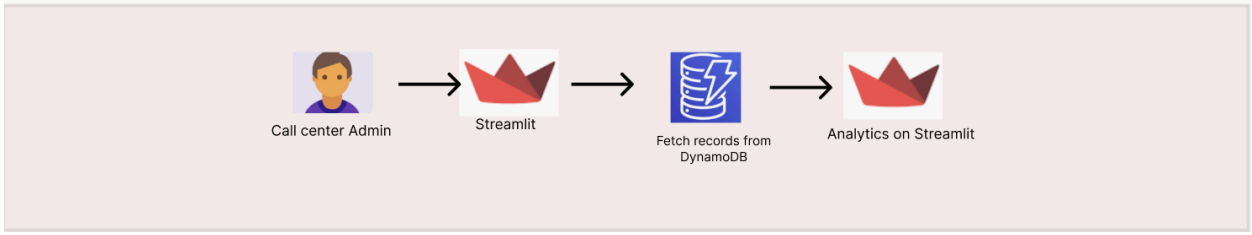
Objective/Problem Statement

A contact center connects the business to the customers, enabling them to request support, make appointments, and much more. Each conversation with a caller is an opportunity to learn more about that customer's needs, and how well those needs were addressed during the call.

We have built a Customer Support Call Intelligence and Analytics application to analyze the sentiment of the calls received to the call center from the customers, which would help in building a valuable customer base and improving the business. Apart from this, the analysis would help the Call Center Supervisor evaluate the performance of the customer service executives or train them.

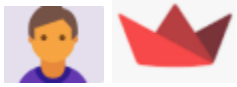
Architecture





Workflow

Admin Uploads the Call Recording

 An admin can sign up as a new user, and later can log into Streamlit with a designated username and password. He/She can then upload a recording of an agent speaking with a customer on the Streamlit frontend, of the format .wav or .mp3

 AWS Cognito is used for user authentication while logging into Streamlit. Only authenticated users will be able to access the Dashboard on Streamlit

 AWS S3 to store Audio files - After a call recording has been uploaded on Streamlit frontend, it is stored as an object in the AWS S3 bucket. For every new recording being uploaded, it is stored as a new object in the same bucket. This way, we will be storing all the audio files in one place.

 AWS Lambda to Trigger Transcribe Job - When an audio file is created as an object, a lambda function is triggered in ordered to start the Transcribe job

 AWS Transcribe - Amazon Transcribe is an automatic speech recognition service that makes it easy to add speech to text capabilities to any application. Transcribe's features enable it to ingest audio input, produce easy to read and review transcripts, improve accuracy with customization, and filter content to ensure customer privacy. The lambda function that triggered the transcribe job previously, gives us a transcription everytime a new audio file is placed in the S3 bucket. This transcripted text is then saved in another (new) bucket in .json format.

 AWS Lambda to Trigger Comprehend Job - After the transcription file is generated and placed in a new bucket, another lambda function is triggered to fetch this file and perform sentiment analysis using AWS Comprehend

 AWS Comprehend - Amazon Comprehend uses natural language processing (NLP) to extract insights about the content of documents. It develops insights by recognizing the sentiments and other common elements in a document. Every time a new transcripted file is generated, a lambda function is triggered to run the comprehend job. We have utilized chunking of text files as Amazon Comprehend imposes a size limit of 5000 bytes on each text file. Since the audio files' transcriptions are usually greater than 5000 bytes, chunking the text files into smaller files helps in running the comprehend job successfully. We get a sentiment, as well as a sentiment score for each of the chunks and we then take an aggregate score of the sentiment to declare an overall sentiment of the call. The result is stored as a json file in the S3 bucket.

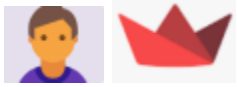
This process will be repeated each time a new audio recording is uploaded into the S3 bucket. The file gets uploaded to S3, goes through Transcribe to get a transcription and finally goes through Comprehend to get the sentiment of the call for each agent-customer combination. Now that we have the data , we will be performing some analysis to view the results in a concise dashboard, using AWS Quicksight

 AWS DynamoDb - Amazon DynamoDB is a NoSQL database that supports key-value and document data models. The result data that is in the json format is stored in DynamoDB as key:Sentiment and value:Sentiment Score pairs. This is then connected to Amazon Athena using Athena connectors

 AWS Athena - Amazon Athena is an interactive query service that makes it easy to analyze data directly in Amazon S3 using standard SQL. We will be using Athena to query the DynamoDb table and get the results, for further analysis in Quicksight. Since DynamoDb cannot be integrated directly with Quicksight (because of architectural constraints imposed by AWS), we are using Athena as a connection between DynamoDb and Quicksight

 AWS Quicksight - Amazon QuickSight allows everyone in the organization to understand the data through interactive dashboards. Quicksight will enable call center admins and other stakeholders to view analytics with respect to performance of the agents, number of calls attended and number of happy customers.

Historic Call Data Analysis of an Employee

 Admin Login - Admin can log into Streamlit - for a better and easier user experience to view dashboards and analytics of their subordinates

 Records are fetched from DynamoDb and analytics is shown on Streamlit frontend

Application Setup

Streamlit Setup:

Download the available streamlit folder

Install the following packages

pip install streamlit pip install boto3

Run the application locally using following command

streamlit run app.py

Go to http://localhost:8501 to view the application locally

AWS Setup:

AWS - Create an AWS account, if you don't have one already

AWS Cognito - Create user pool and setup an app client. User pools are directories of federated and local user profiles. They provide authentication options for the users.

AWS S3 - Create a bucket to store various objects on S3 like audio files, transcripts, sentiment data etc.

AWS S3 event triggers - Place event triggers on the landing folders to kick off the lambda functions. For example if an audio file is upladed to S3 in a particular path, the event trigger will kick off the lambda fuction for the AWS Transcribe job.

AWS Lambda - Create 3 lambda function to accept the audio files,transcribe, comprehend, load the sentiment analysis to dynamodb and visualize on quicksight. Use the code provided in src/Lambda Functions folder.

AWS IAM roles - Create IAM roles for each of the lambdas to access the AWS Transcribe, AWS Comprehend, DynamoDb, CloudWatch and S3

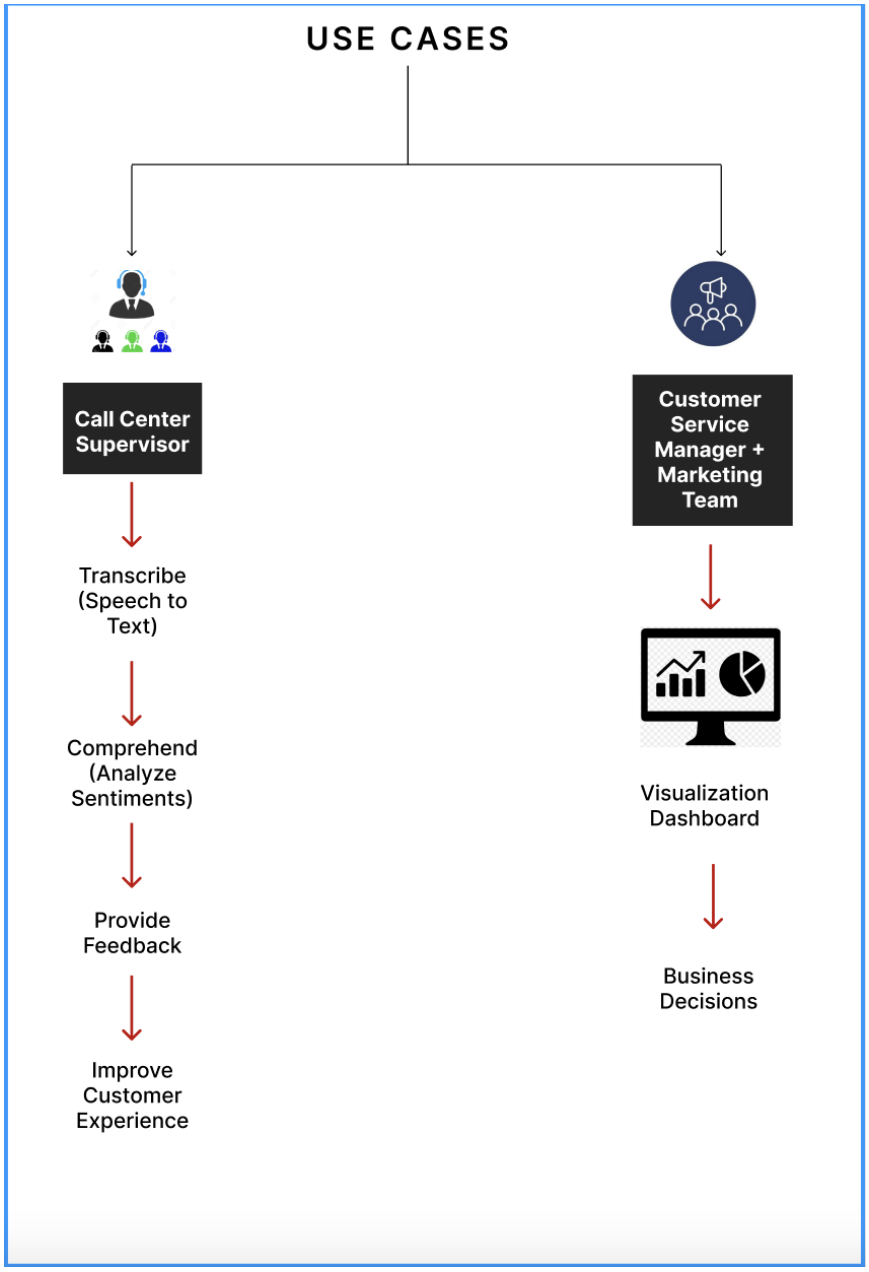
AWS Athena - The data in Dynamodb cannot be directly visualized on AWS QuickSight, we are using Athena connecters to be able to visualize the data on quicksqicksight

AWS QuickSight - The Markeing Manager and the Call center Manager are provided with an interactive dashboard with various insights on sentiment analysis.

Quicksight Access - The marketing manager and the customer service manager will be given access to the quicksight dashboard to view the analytics. They need to follow the below steps to acces the dashboard via the link: https://us-east-1.quicksight.aws.amazon.com/sn/dashboards/aab311a1-27ba-40fc-80ba-532ddd1c8dd2

1. Create a quicksight account once the email with the invitation is received
2. account name: bigdata-team7, make sure the correct account name is passsed during login
3. View the interactive dashboard

Use Cases


USE CASES
Call Center Supervisor
Transcribe (Speech to Text)
Comprehend (Analyze Sentiments)
Provide Feedback
Improve Customer Experience
Customer Service Manager + Marketing Team
Visualization Dashboard
Business Decisions

Dataset

The dataset consists of call recordings between agents and customers. We will be working on these audio files to convert them to text files in order to perform sentiment analysis and visualizations so that the admins can get insights on how well their employees are performing, and how satisfied their customers are.

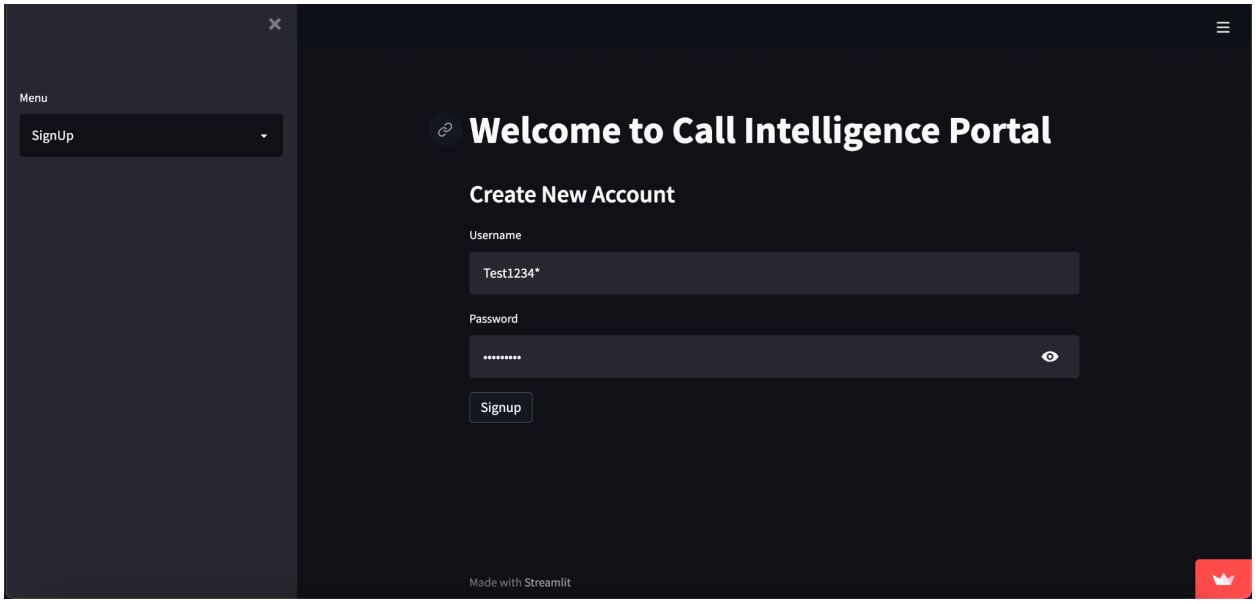
Dataset Link : https://media.talkbank.org/ca/CallHome/eng/

Implementation

Login Page

Services used : Amazon Cognito for authentication of the user. Once a new user is created, the user is authenticated at the backend. Only authenticated users have access to login into the portal.

Signing Up



Login

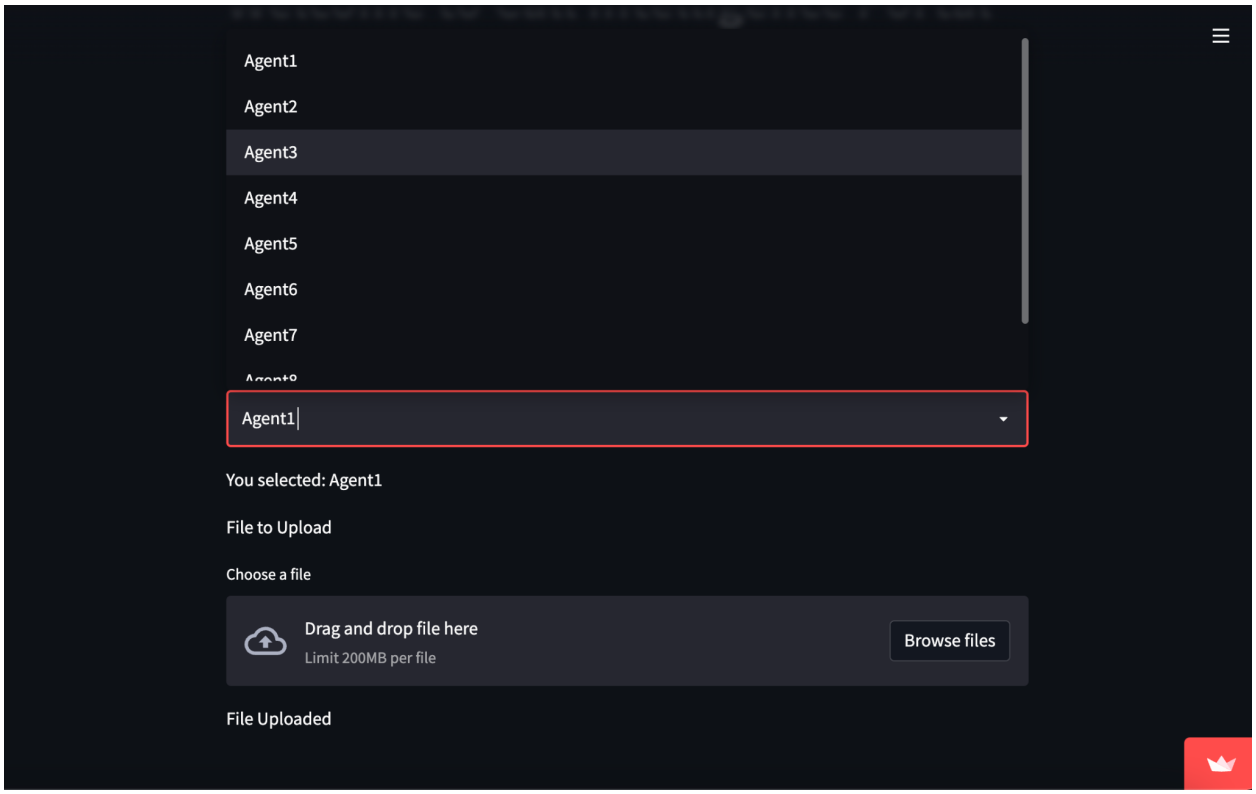
Call Intelligence Portal

After logging in, the user (admin) has three options to select from -

- Upload Audio - To upload an audio file recording of an existing agent with a customer
- Agent Performance - To look up an agent's performance, get insights into how the agents are catering to the customers
- Create Agent - To create a new agent, if there is a need

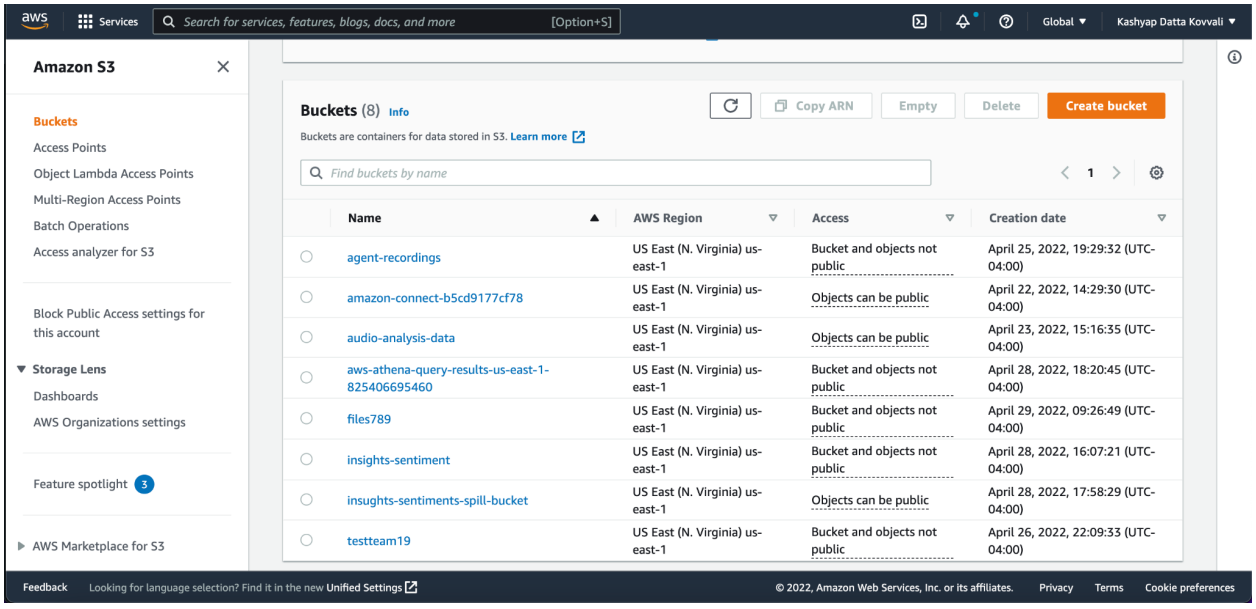
Uploading Audio

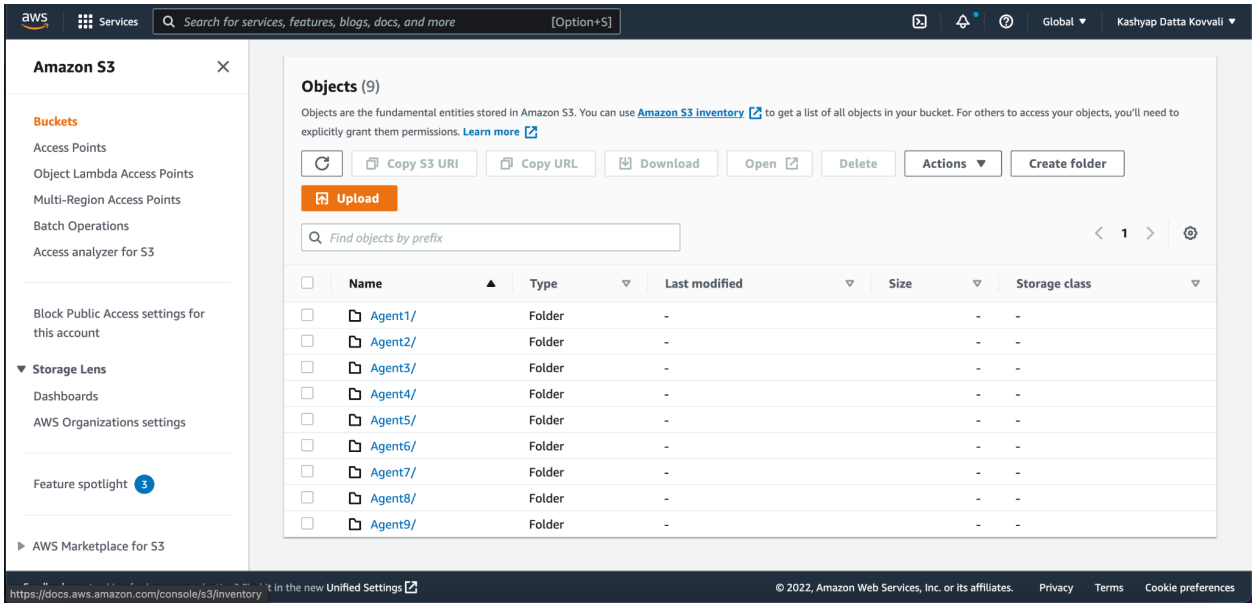
When the admin chooses to upload audio, there is an option to choose what agent to upload for, and is able to upload a file with <200MB limit



Analysis -

Storing the audio file in an S3 bucket

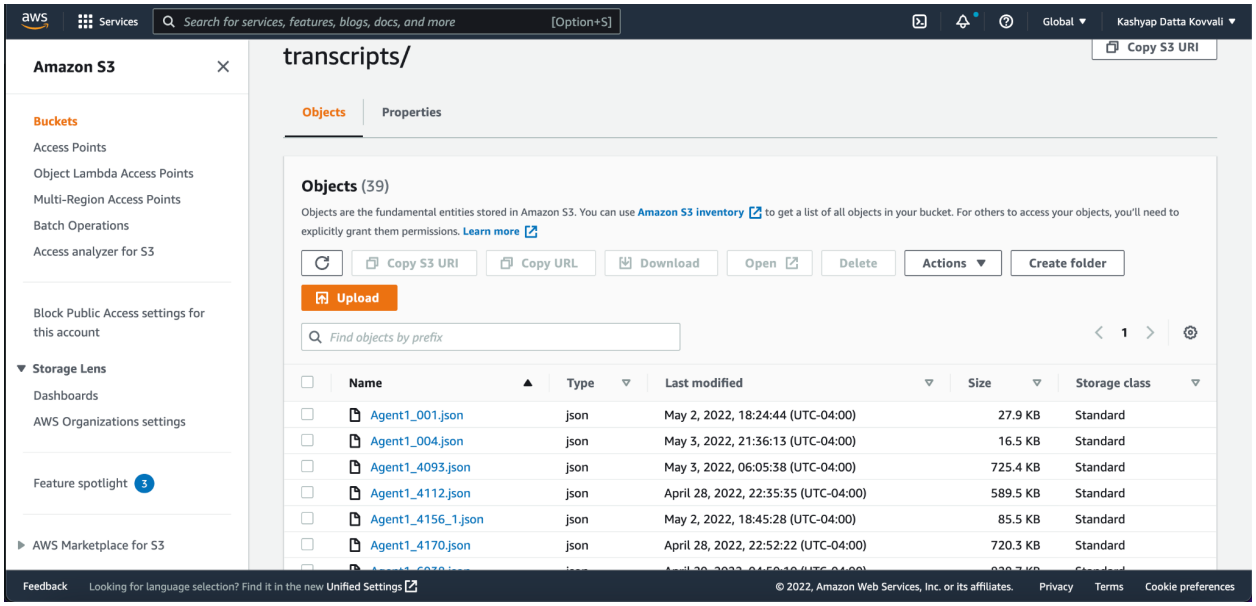




Transcription of the audio file

Transcription is done by Amazon Transcribe and stored in the S3 bucket in a .json format under the object ‘transcripts’, as and when the lambda function is triggered for the same

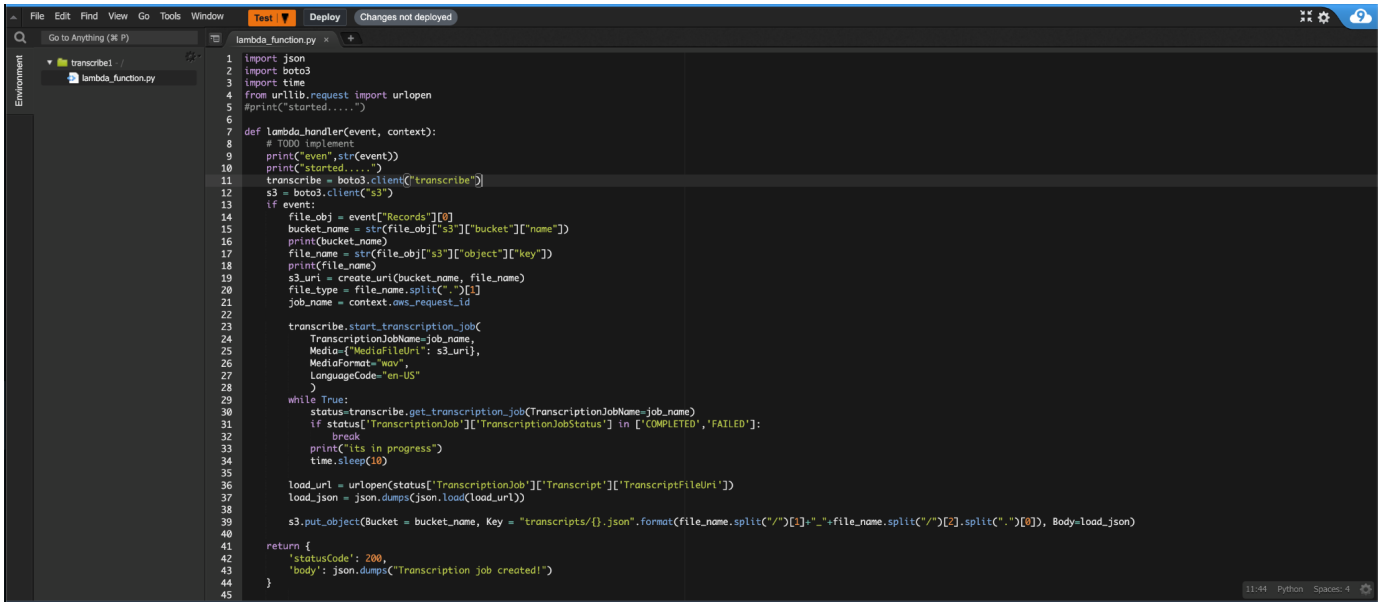
Transcripted file



Sample text file

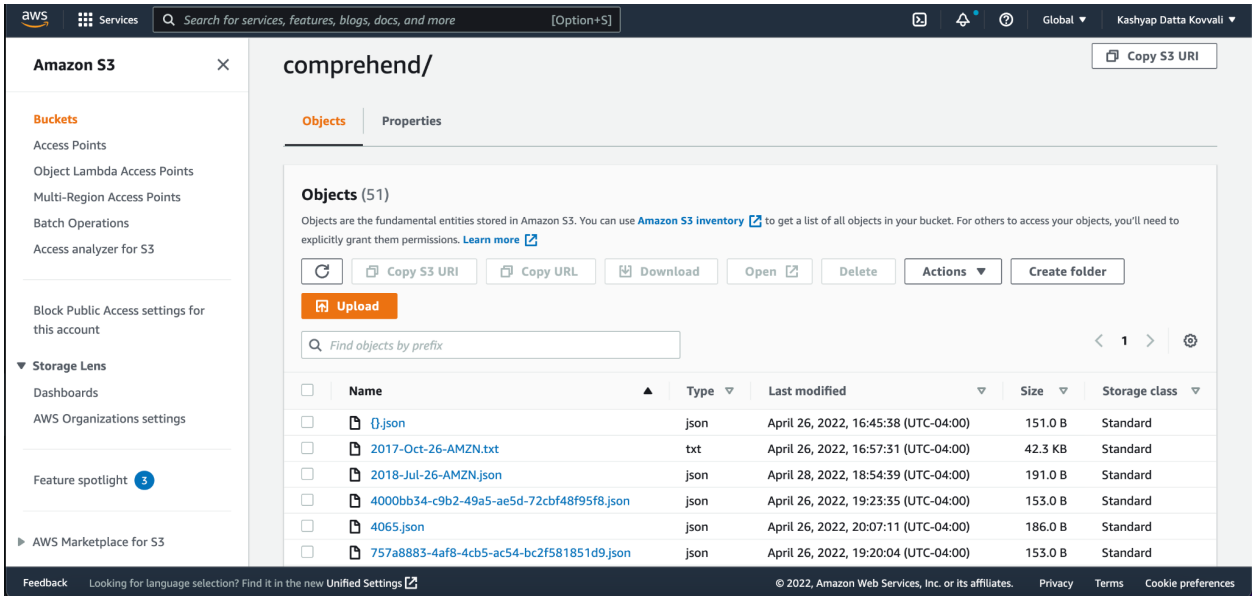

```
{"jobName": "531d07db-153c-489b-ae87-194faf0ec412", "accountId": "825406695460", "results": {"transcripts": [{"transcript": "Hello ma'am good
morning how can I assist you today? Hi I'm facing an issue with my blender. It's not working after a month of its purchase. Can you please help me
with this? Can you please let me know the order already? Sure it is 4356789. Thank you. I am pulling the information regarding the order. Please
hold a few minutes. Sure. Happy to hold. I see that the written period is closed for this item. Yes I'm aware of that. But can you please help me?
barely use this item and it has stopped working suddenly. Sure. Let me see what I can do over here. I can send you a new product and you can ship
the old product back to us through ups. Does that work for you? Yes so absolutely thank you so much for your help. You made my day. Really happy
with this resolution. It was a pleasure assisting you today. Is there anything else I can do for you know I'm all good thank you so much for your
assistance. You have a good day. Have a good day. Thank you. Bye bye."}], "items": [{"start_time": "0.84", "end_time": "1.2", "alternatives": [
```


Lambda Function to Trigger Amazon Transcribe

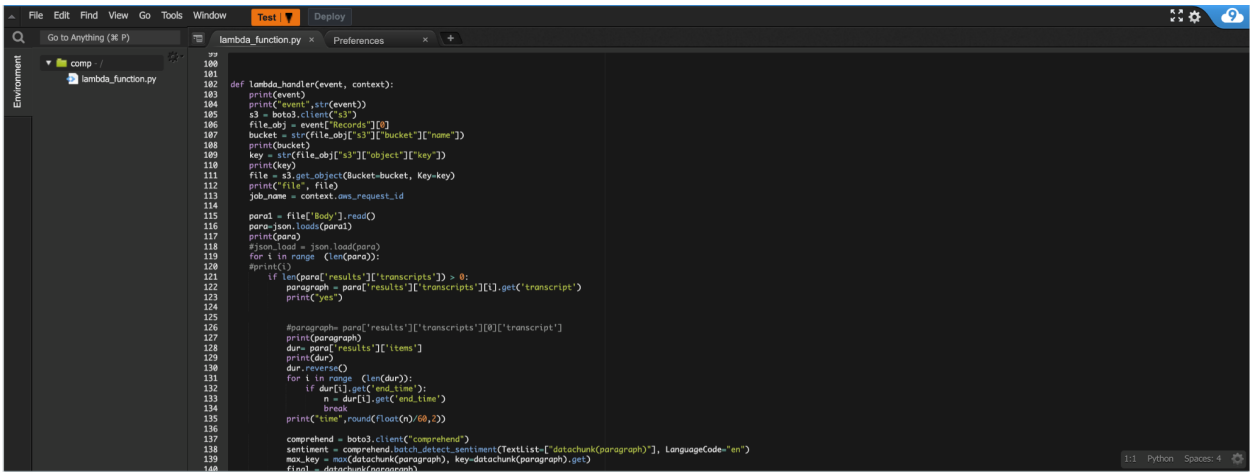


Amazon Comprehend for Sentimental Analysis

After the transcripted file is created, another lambda function is triggered to start the comprehend job. Amazon Comprehend is used for sentimental analysis of the transcribed file and the results are stored in the S3 bucket in .json format under ‘comprehend’ object

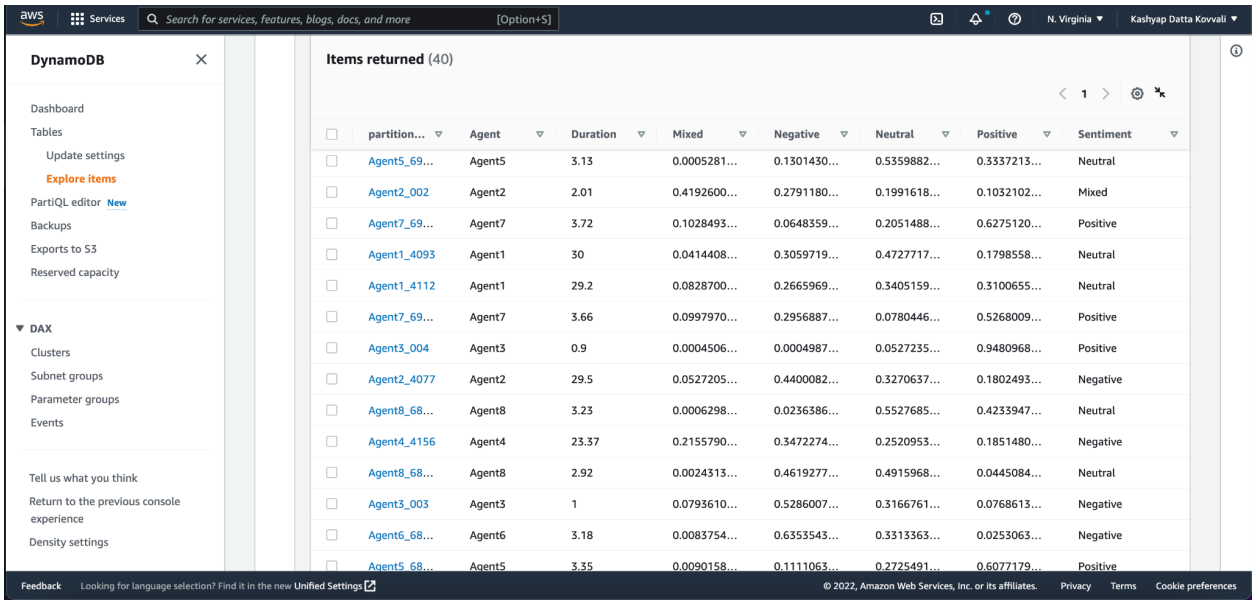


Lambda Function to Trigger Amazon Comprehend



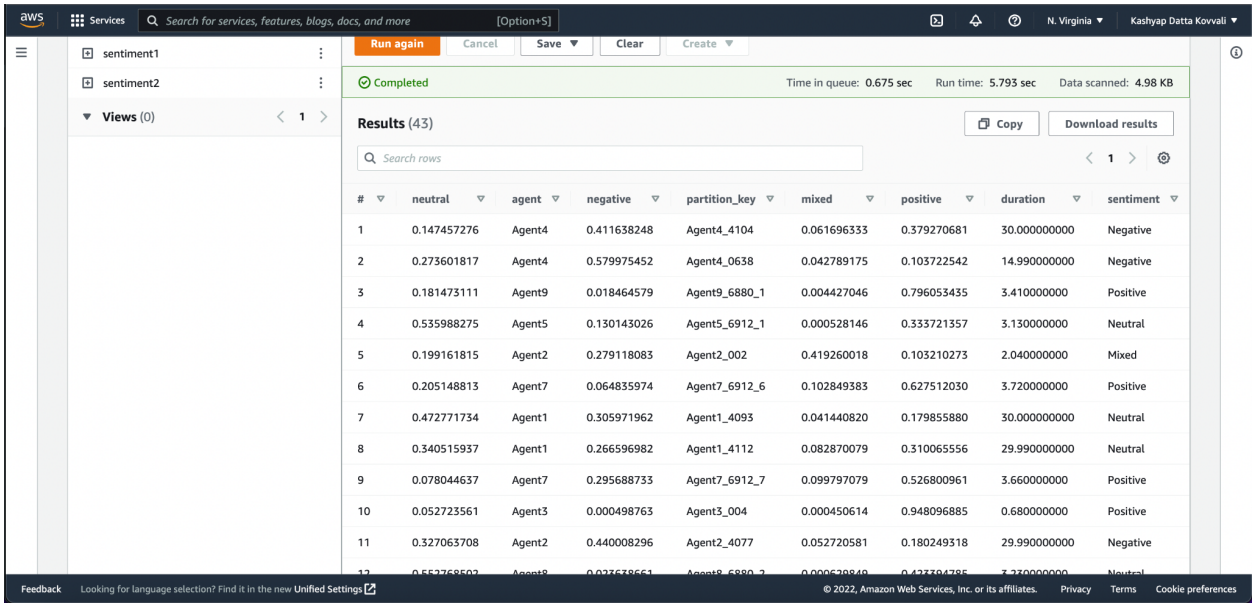
DynamoDb Table

The results of the comprehend job are stored in DynamoDb tables



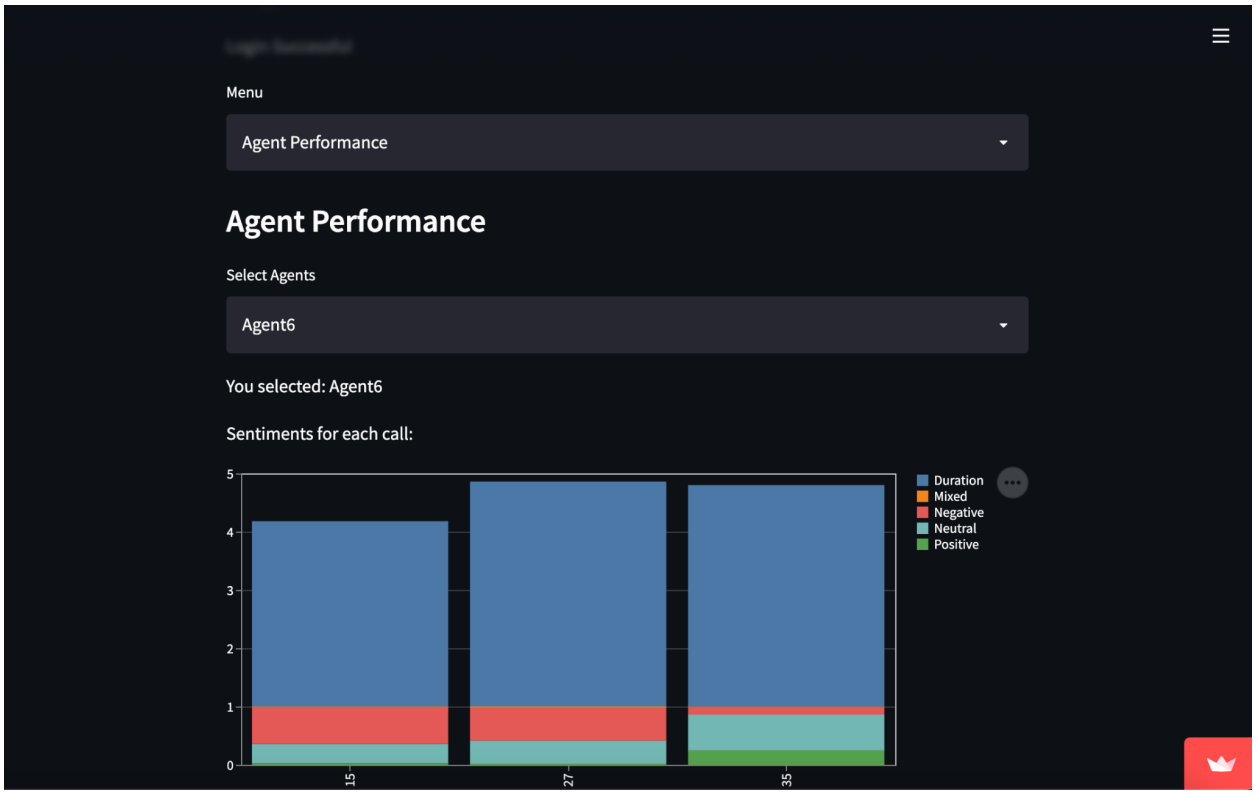
Athena

The results in DynamoDb are visualized in AWS Quicksight via the Athena connectors. Table items are returned in Athena tables through SQL queries.



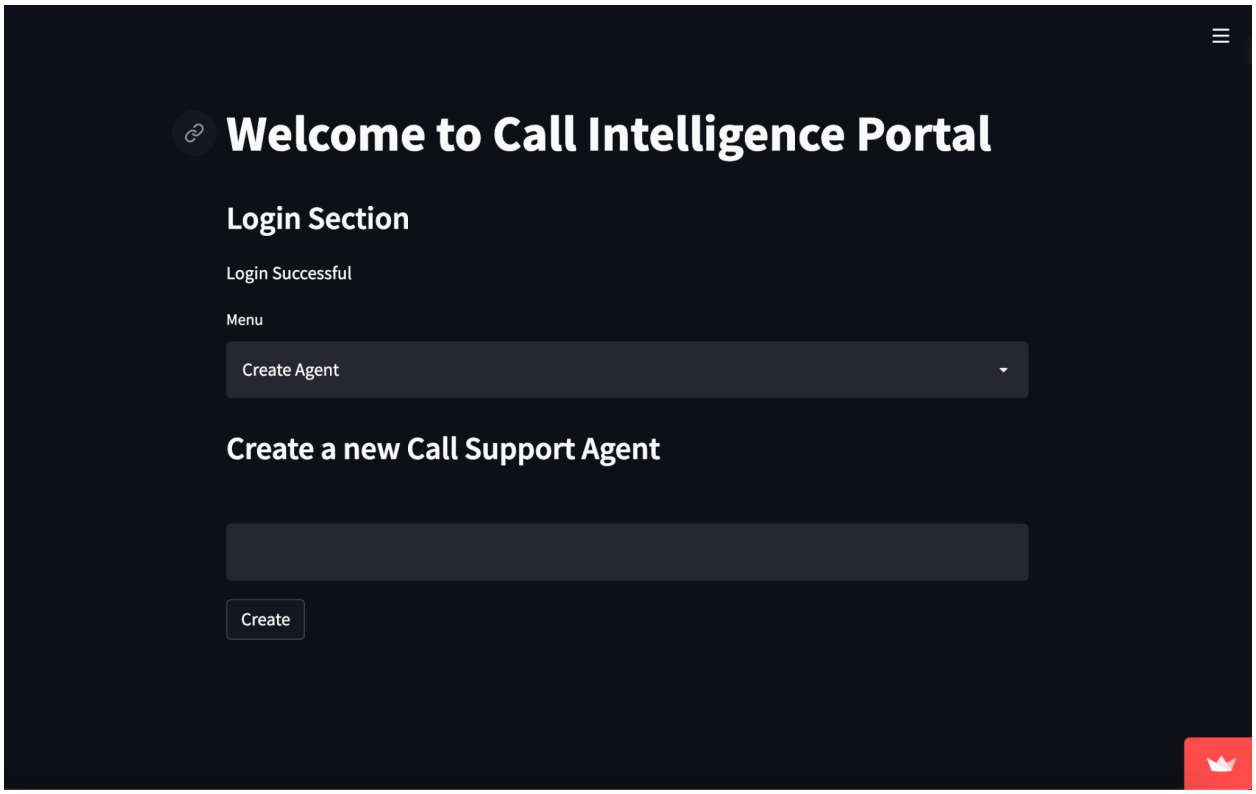
Agent Performance

The admin will be able to view each agent's performance based on the duration of the call, sentiments for each call - mixed, neutral, positive or negative



Create Agent

Admin will be able to create a new agent

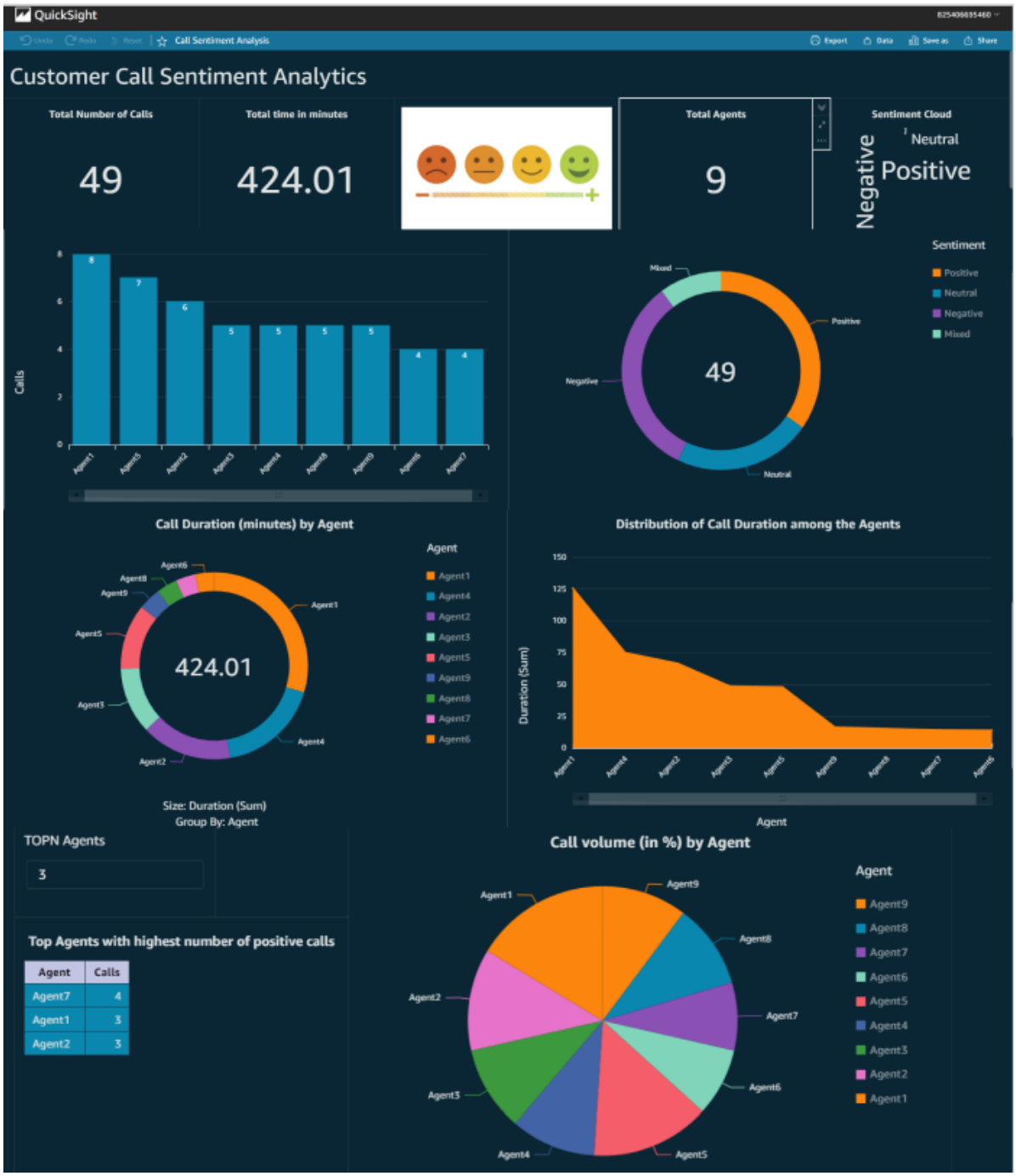


QuickSight access

The marketing manager and the customer service manager will be given access to the quicksight dashboard to view the analytics. They need to follow the below steps to access the dashboard via the link: https://us-east-1.quicksight.aws.amazon.com/sn/dashboards/aab311a1-27ba-40fc-80ba-532ddd1c8dd2

1. Create a quicksight account once the email with the invitation is received.
2. account name: bigdata-team7, make sure the correct account name is passed during login.

3. View the interactive dashboard.



Testing

Unit Testing with pytest

For unit testing, pytest framework has been used. The pytest framework makes it easy to write small, readable tests

To Install Pytest run the following command in your command line -

- *pip install -U pytest*

A new file called test_pytest has been created, containing functions to test the code.

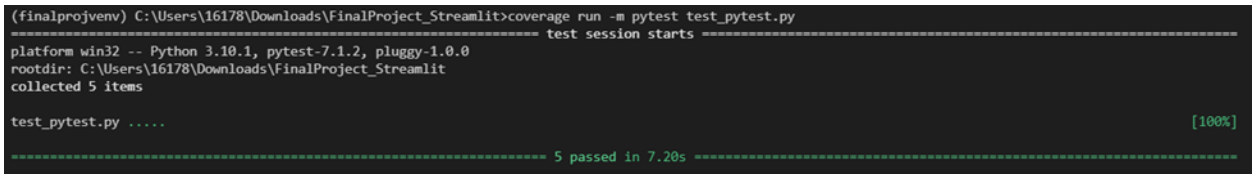
test_pytest.py > ...

```python
from dynamotodf import checkfile,getAgentData,dbtodf
from userlogin import login,signup
import pandas as pd
data = dbtodf()


def test_checkfile():

    assert checkfile(dbtodf(),'Agen10_9').empty == True
    assert checkfile(dbtodf(),'').empty == True
    assert checkfile(dbtodf(),'Agent1_001').equals(data.loc[data['partition_key'] == 'Agent1_001'])
    assert checkfile(dbtodf(),'Agent1_001').equals(data.loc[data['partition_key'] == 'Agent1_001']) == True
    assert checkfile(dbtodf(),'Agent3_4074').equals(data.loc[data['partition_key'] == 'Agent3_4157']) == False


def test_getAgentData():
    agent_df,sentimentcountdf = getAgentData(data,'Agent2')
    assert agent_df.equals(data.loc[data['Agent']== "Agent2"]) == True
    assert sentimentcountdf.equals(data.loc[data['Agent']== 'Agent2']['Sentiment'].value_counts()) == True
    agent_df,sentimentcountdf = getAgentData(data,'Agent5')
    assert agent_df.equals(data.loc[data['Agent']== "Agent3"]) == False
    assert sentimentcountdf.equals(data.loc[data['Agent']== 'Agent3']['Sentiment'].value_counts()) == False


def test_fileformat():
    filename = "3123.txt"
    wavfilename = "3123.wav"
    assert (filename.rsplit( ".", 1 )[ 1 ] == "wav") == False
    assert (wavfilename.rsplit( ".", 1 )[ 1 ] == "wav") == True

def test_login():
    assert login('knjbnmbhj','sdhjb436') == 400
    assert login('','') == 400
    assert login('test12345*','Test12345*') == 200

def test_signup():
    assert signup('scsjj','ksjsh') == 400
    assert signup('Pytest004*','Pytest004*') == 200
```

Run the test using the following command in your command line -

- pytest test_pytest.p*y*

Running the tests will give us the following results



Code Coverage with coverage.py

For measuring the coverage of the tests, we have used coverage.py package. coverage.py uses the code analysis tools and tracing hooks provided in the Python standard library to determine which lines are executable, and which have been executed.

To Install coverage.py run the following command in your command line -

- *pip install coverage*

Use coverage run to run the test suite and gather data in command line as below -

- *coverage run -m pytest* test_pytest.p*y*

Use coverage report to get the report of the test -

- *coverage report -m*

Running the report will give us the following result

```
(finalprojvenv) C:\Users\16178\Downloads\FinalProject_Streamlit>coverage report -m
Name             Stmts   Miss  Cover   Missing
----------------------------------------------
dynamotodf.py       23      0   100%
test_pytest.py      29      0   100%
userlogin.py        16      0   100%
----------------------------------------------
TOTAL               68      0   100%
```

Documentation

Code documentation is crucial for any project. It increases code maintainability, helps troubleshoot production issues, help manage any additional integrations and is good for knowledge transfer.

We have created a documentation for streamlit code using the pdoc package and hosted it using Github Pages for public access.

Documentation Link: https://kashyap-datta.github.io/callanalyticsdocumentation/