



DALHOUSIE
UNIVERSITY

CSCI 5410

Serverless Data Processing

Fall 2024

Sprint-3 Report

Team-06

Team Members:

Shifa Samir Mirza (B00981843)

Jems Shaileshkumar Patel (B00984406)

Vedant Shaileshkumar Patel (B00984592)

Sivaprakash Chittu Hariharan (B00980275)

Deployed URL: <https://frontend-service-791648625124.us-central1.run.app>

Table of Contents

<i>List Of Figures</i>	3
<i>System architecture design:</i>	4
<i>Pseudo Code</i>	5
Module 2:	5
Workflow:	5
Pseudocode for Module 2 in Steps:	5
Flowchart:	7
Module 3:	8
Pseudocode for Module 3 in steps:	8
Flowchart:	9
Module 4:	10
Workflow Pseudocode in Steps Format for Module 4	10
Pseudocode for Module 4:	10
Flowchart:	12
Module 5:	13
Workflow:	13
Pseudocode for Module 5 in steps:	13
Flowchart:	15
Module 6:	16
Workflow Pseudocode in Steps Format for Module 6	16
Flowchart:	17
<i>Functional Testing:</i>	18
<i>References:</i>	31

List Of Figures

Figure 1 : Flowchart module 2 Dialogflow	7
Figure 2 : Flowchart of Module 6 data analysis	17
Figure 6 Homepage for the customer.....	18
Figure 7 Homepage for QDP Agents.....	18
Figure 8 Data Processing 1	19
Figure 9 Uploading file in DP1	19
Figure 10 Processed File for DP1	20
Figure 11 Data Processing 2	20
Figure 12 Uploading File in DP2.....	21
Figure 13 Processed Files for DP2.....	21
Figure 14 Data Processing 3	22
Figure 15 Uploading file in DP3.....	22
Figure 16 Processed files in DP3.....	23
Figure 17 Looker studio Dashboard for WordCloud.....	23
Figure 18 Feedback Form and displaying processed files	24
Figure 19 Giving Feedback.....	24
Figure 20 Tabular Format of All Feedbacks	25
Figure 21 Looker studio Dashboard for the QDP Agents for showing User Stats	25
Figure 22 Pub-Sub Chat Page for Customer.....	26
Figure 23 Creating concerns	26
Figure 24 Filling concerns form	27
Figure 25 Display all the concerns.....	27
Figure 26 Pub Sub Chat page for the QDP Agents.....	28
Figure 27 Admin Start Conversation	28
Figure 28 Customer Got Message.....	29
Figure 29 Send message to Admin	29
Figure 30 Admin got message	30

System architecture design:

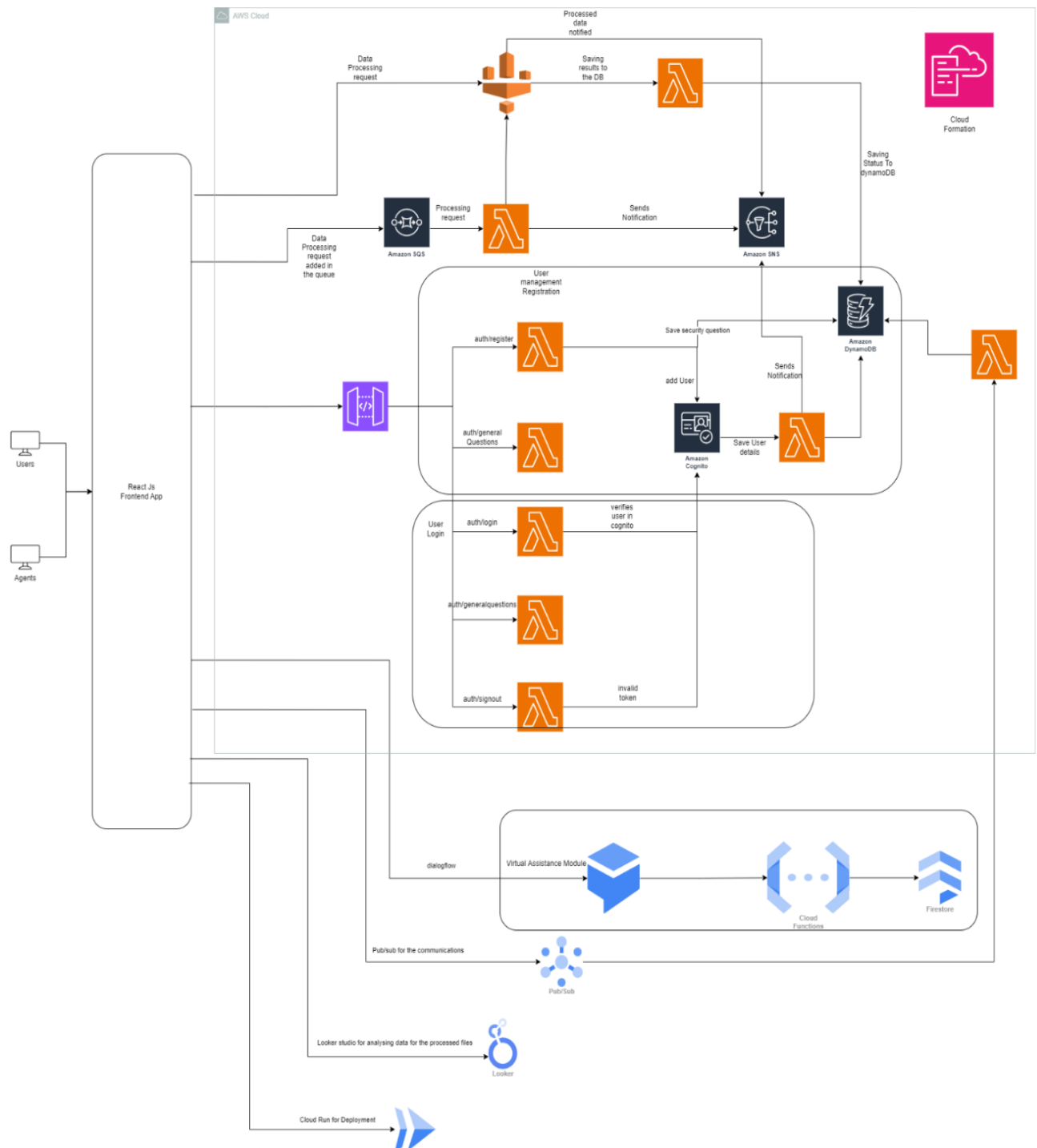


Figure 1: Final System Architecture Design

Pseudo Code

Module 2:

Workflow:

1. **Initial Interaction:** User asks a question or navigates to a page.
2. **User Query:**
 - If the query is about **registration**: Provide the steps to register.
 - If the query is about **general concerns**: Store the concern in Firebase.
 - If the query is to **fetch file details**: Look up the file using a **reference code** and provide details[5].
3. **Store Concerns in Firebase:** If a concern is registered, save it for future use.
4. **Provide File Details:** Based on the reference code, retrieve the file's details from Firebase (or another source)[13][12].

Pseudocode for Module 2 in Steps:

Step 1: Start User Interaction

- **Action:** The system receives a query from the user (e.g., asking about registration, file reference, or a general concern).

Step 2: Check if Query is About Registration

- **Action:** If the user asks how to register, execute the following steps:
 1. Retrieve the registration steps from the system.
 2. Provide the user with a clear set of registration instructions.

Step 3: Check if Query is About File Reference

- **Action:** If the user asks about a file reference (e.g., "I have a reference code for a file"), execute the following steps:
 1. Extract the reference code from the user's query.
 2. Look up the file details from the database or storage (e.g., Firebase).
 3. Return the file details to the user (or an error if the file is not found).[1]

Step 4: Check if Query is About General Concerns

- **Action:** If the user asks a general question or raises a concern (e.g., "I have an issue with my registration"), execute the following steps:
 1. Extract the user's concern from the query.
 2. Store the concern in the Firebase (or another database).
 3. Acknowledge that the concern has been successfully stored.

Step 5: Default Response (If Query Is Unclear)

- **Action:** If the query doesn't match any of the above categories, provide a generic response asking the user for clarification.

Step 6: Store Concern in Firebase

- **Action:** If the user provides a concern, store it for future follow-up or processing. This can involve pushing the concern to a collection in Firebase.[1]

Step 7: Provide Registration Steps

- **Action:** Return the steps for registration if the user asks how to register.

Step 8: Extract Reference Code

- **Action:** Extract the reference code from the query. This could involve finding a specific pattern or identifier in the query.

Step 9: Fetch File Details

- **Action:** Retrieve file details from the database (e.g., Firebase) using the extracted reference code.

Step 10: End Interaction

- **Action:** End the interaction with the user after providing the necessary information or guidance.

Flowchart:

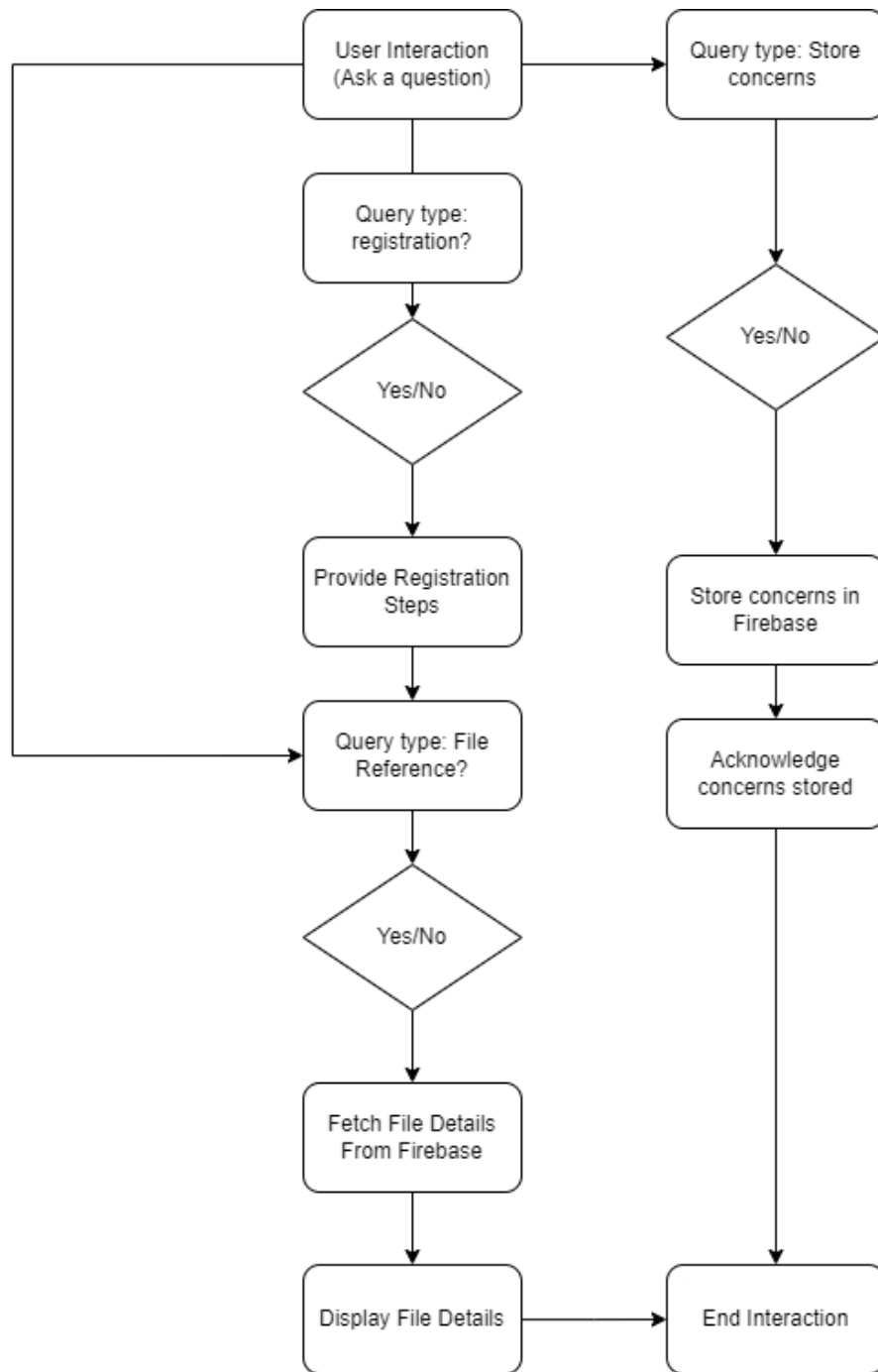


Figure 2: Flowchart module 2 Dialogflow

Module 3:

Pseudocode for Module 3 in steps:

Step 1: Customer Submits Concern

- **Action:**
 1. User logs in (Registered User or QDP Agent).
 2. User selects an option to submit a concern based on the data processing reference code.
 3. Customer fills in a concern (text input).
 4. Concern is captured along with the reference code.

Step 2: Publish Message to GCP Pub/Sub [2]

- **Action:**
 1. Publish the concern message, including the concern text and reference code, to a GCP Pub/Sub topic.
 2. The concern message is sent to the topic for processing.

Step 3: Subscribe to Pub/Sub Topic

- **Action:**
 1. A Cloud Function subscribes to the GCP Pub/Sub topic.
 2. The Cloud Function listens for new messages (concerns) in the Pub/Sub topic.
 3. When a new message (concern) is published, the Cloud Function receives it.

Step 4: Assign Concern to Random QDP Agent

- **Action:**
 1. The Cloud Function selects a random QDP Agent from a pool of agents stored in DynamoDB or Firestore.
 2. The system randomly selects a QDP Agent to handle the concern.

Step 5: Forward Concern to QDP Agent

- **Action:**
 1. The Cloud Function forwards the concern to the selected QDP Agent.[3]
 2. The QDP Agent receives a notification (via email or internal system) with the concern message and reference code.[6]

Step 6: Log Communication in NoSQL Database

- **Action:**
 1. The concern, reference code, customer's ID, and assigned QDP Agent's ID are logged into a NoSQL database (DynamoDB or Firestore).
 2. A timestamp is recorded for when the concern was received and assigned.

Step 7: QDP Agent Responds to Concern

- **Action:**
 1. The QDP Agent reviews the concern and formulates a response.
 2. The response is logged in the NoSQL database, associating the response with the reference code and customer.

Step 8: Notify Customer

- **Action:**
 1. The system sends a notification to the customer, informing them that their concern has been addressed by the QDP Agent.
 2. The notification can be sent via email or through the system's in-app messaging.

Step 9: End of Communication Workflow

- **Action:**

1. The communication workflow ends once the concern is resolved.
2. The concern and response are marked as complete in the system.[2]

Flowchart:

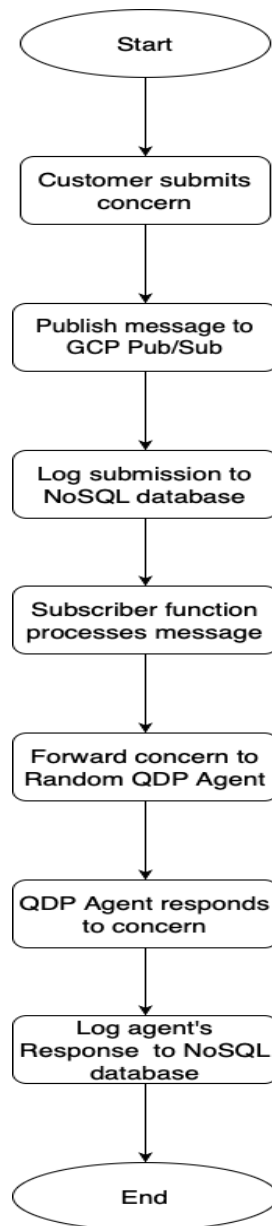


Figure 3: Flowchart for module 3 Message passing module

Module 4:

Workflow Pseudocode in Steps Format for Module 4

1. **Start Notification Trigger:**
 - Trigger notifications based on specific events such as registration, login, or data processing status (success/failure).
2. **Event Detection:**
 - Detect events like:
 - Successful registration.
 - Successful login.
 - Data processing (success/failure).
3. **Notification Generation:**
 - Create appropriate notification messages for each detected event.
 - Format the message to include user-specific details, such as their username.
4. **Queue Management:**
 - Push notification messages to an SQS queue.[7]
 - Ensure SQS queues are categorized based on event type for efficient processing.
5. **SNS Integration:**
 - SNS sends notifications to the user's email.
 - SNS processes messages from the SQS queue and delivers them.
6. **Confirmation:**
 - Log successful notifications in a database (e.g., DynamoDB) for auditing purposes.
7. **End Notification Flow:**
 - Exit the notification module after confirmation.[6]

Pseudocode for Module 4:

Step 1: Start Notification Trigger

- **Action:** The system initiates the notification process whenever a triggering event is detected.

Step 2: Detect Triggering Event

- **Action:**
 1. Check if the event is "User Registration."
 2. Check if the event is "User Login."
 3. Check if the event is "Data Processing Result (Success/Failure)."[6]

Step 3: Generate Notification Message

- **Action:**
 1. Extract the user's email address from the event data.
 2. Derive the username from the email (substring before @).
 3. Format the notification message based on the event type:
 - For registration: "Welcome, [USERNAME]! Your registration was successful."
 - For login: "Hello, [USERNAME]! You have successfully logged in."
 - For data processing:

- On success: "Hi [USERNAME], your data processing completed successfully."
- On failure: "Hi [USERNAME], your data processing failed. Please try again."

Step 4: Push Notification to SQS Queue [7]

- **Action:**
 1. Choose the appropriate SQS queue based on the event type.
 2. Push the formatted message to the selected queue.

Step 5: SNS Integration

- **Action:**
 1. Set up SNS to subscribe to the SQS queue.[6]
 2. Process messages in the queue and send them as email notifications to the user's email address.

Step 6: Log Notification

- **Action:**
 1. Record the notification details in DynamoDB for future reference.
 2. Include fields like timestamp, event type, username, email, and message status.

Step 7: Confirmation and End

- **Action:**
 1. Confirm that the notification was sent successfully.
 2. Exit the notification module.

Flowchart:

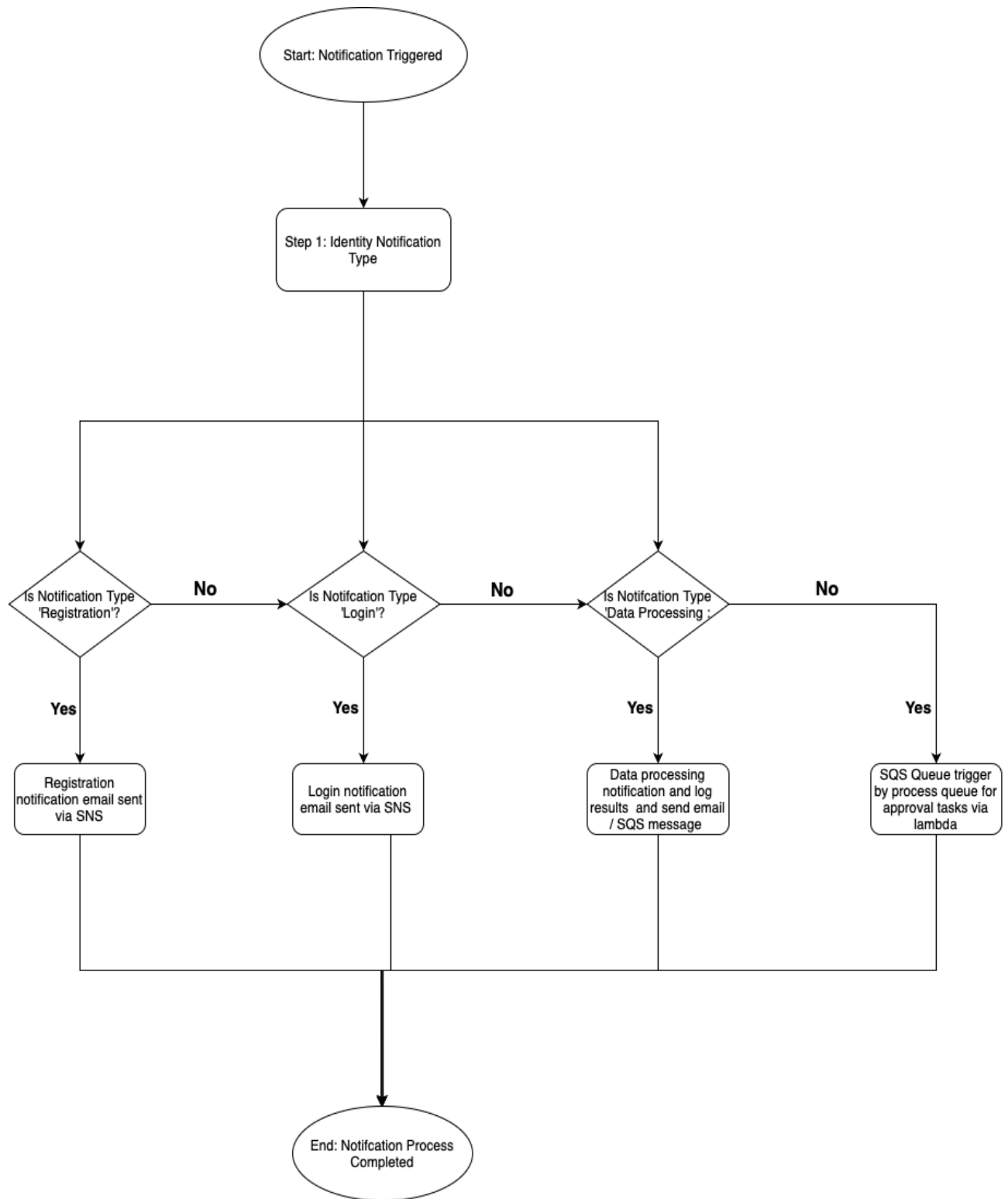


Fig 4: Flowchart of Module 4 Notification

Module 5:

Workflow:

1. File Upload:

- Allow users (Guest, Registered, or QDP Agents) to upload files (TXT, CSV, or JSON).
- Validate the file type and size.

2. Processing Options for Guest Users:

- Guests can process a maximum of **two files** of any type (TXT, CSV, JSON).
- Processing results are not shown without registration.
- Guests can view:
 1. Feedback and polarity analysis for a particular dataset.
 2. Virtual assistant/chatbot responses for basic site navigation.

3. Processing Options for Registered Users and QDP Agents:

- Allow processing of unlimited files.
- Offer advanced processing services:
 1. JSON to CSV conversion (using AWS Glue) [4].
 2. Named Entity Extraction from TXT files (using AWS Lambda).
 3. Word Cloud Generation for TXT files (using GCP Looker Studio).
- Save all results and statuses in DynamoDB [13].

4. Display Results:

- Guests: Restricted access to results (encouraged to register for full access).
- Registered Users and QDP Agents: Retrieve results using a reference code.

5. Error Handling:

- Handle file upload or processing errors.
- Notify users via SNS if processing fails [4].

Pseudocode for Module 5 in steps:

Step 1: Start File Upload

- **Action:** Accept a file upload request.
 1. Validate the file type (JSON, TXT, or CSV).
 2. Check the file size.
 3. If validation passes, store the file temporarily.

Step 2: Identify User Type

- **Action:** Determine if the user is a Guest, Registered Customer, or QDP Agent.

Guest User:

1. Allow processing of up to **two files only**.
2. Provide virtual assistant/chatbot access for:
 - Basic website navigation information.
 - Feedback and polarity data for datasets.
3. Restrict access to processing results (registration required to view).

Registered Users/QDP Agents:

1. Allow unlimited file processing.

2. Provide full processing results.

Step 3: Check File Count for Guests

- **Action:** If the user is a Guest:
 1. Check the count of previously processed files:
 1. If two files have already been processed:
 - Restrict further processing.
 - Prompt the user to register for additional processing.
 2. If less than two files:
 - Allow processing.

Step 4: Perform File Processing

- **Action:** Invoke the appropriate processing service:
 1. **JSON to CSV conversion:**
 - Trigger AWS Glue for conversion.
 - Save the status in DynamoDB[13].
 2. **Named Entity Extraction (TXT files):**
 - Trigger AWS Lambda for entity extraction.
 - Save the extracted data in DynamoDB.
 3. **Word Cloud Generation (TXT files)[15]:**
 - Trigger GCP Looker Studio for visualization.
 - Store the result in GCP buckets.

Step 5: Save Processing Results

- **Action:** Store the processing results and status in DynamoDB with a unique reference code.

Step 6: Notify Users of Processing Status

- **Action:** Use AWS SNS to send notifications for [6]:
 1. Successful processing.
 2. Errors encountered during processing.

Step 7: Retrieve and Display Results

- **Action:** Fetch results based on the user type:
 1. **Guest Users:**
 - Show a message indicating results are available after registration.
 - Provide access to feedback/polarity analysis.
 2. **Registered Users/QDP Agents:**
 - Retrieve results using the reference code from DynamoDB.
 - Display results to the user.

Step 8: End Processing Workflow

- **Action:** Conclude the interaction with the user [4].

Flowchart:

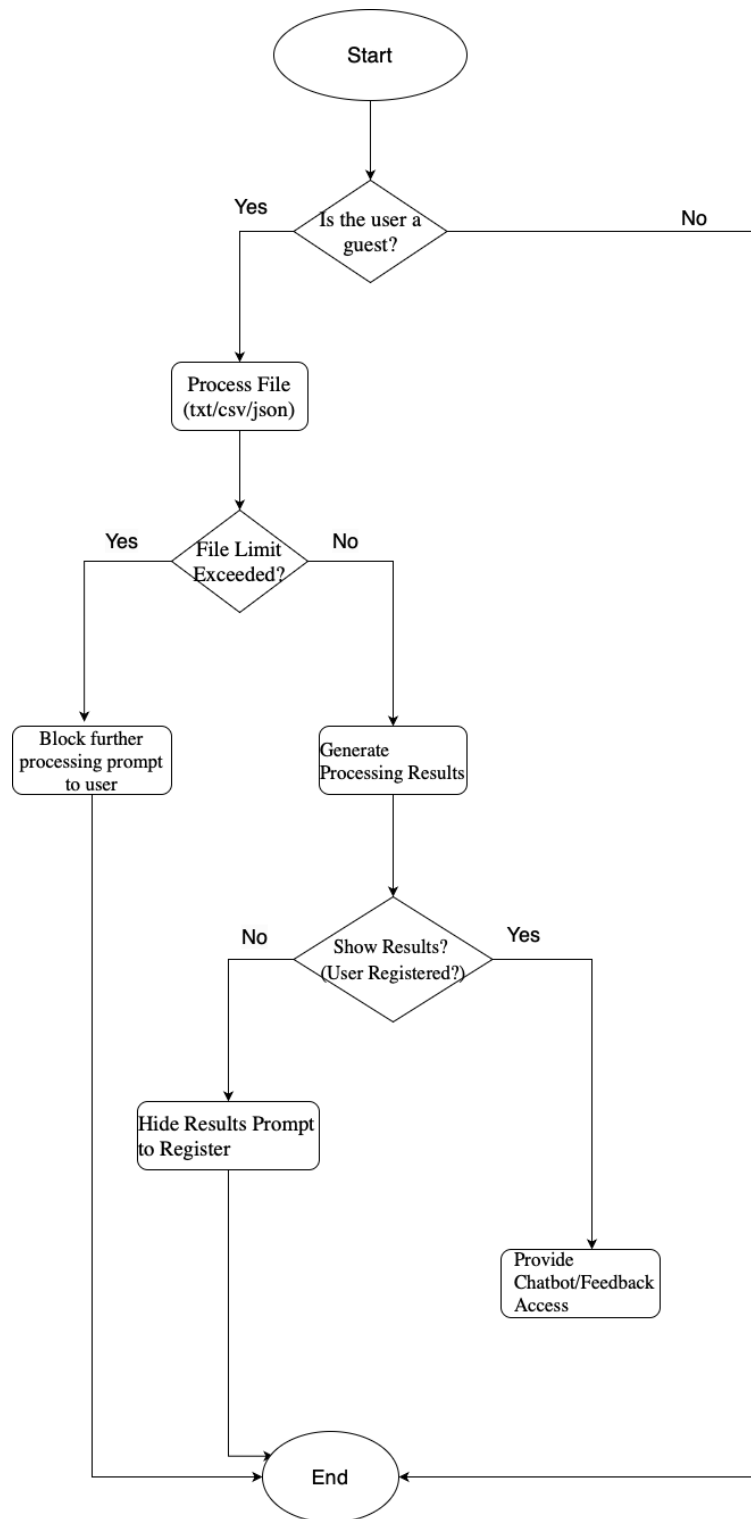


Fig 5: Flowchart of Module 5 Data Processing

Module 6:

Workflow Pseudocode in Steps Format for Module 6

Part 1: User Login Tracking

1. **When a user logs in:**
 - Retrieve the current login count for the user from **Firestore**.
 - Increment the login count by 1.
 - Update the new login count in both:
 - **Firestore** (for quick access)[1].
 - **BigQuery** (for analytical purposes)[9].
2. **If the user is an admin:**
 - Fetch all user login data from **BigQuery**.
 - Pass the data to **Looker Studio** to generate a dashboard showing:
 - Total number of logins.
 - Login statistics by user.

Part 2: Feedback System

1. **When a user submits feedback:**
 - Allow the user to select a **reference code** for a processed file from a dropdown menu.
 - Collect the feedback description from the user.
2. **Validate the reference code:**
 - If the code is invalid, return an error message.
3. **Analyze the feedback:**
 - Call **Google Natural Language API** to analyze the sentiment of the feedback.
 - Extract the sentiment polarity (positive, neutral, or negative)[8].
4. **Store feedback:**
 - Combine the feedback, reference code, and sentiment polarity into a feedback record.
 - Save the record in **Firestore**[1].
5. **Display feedbacks:**
 - Retrieve all feedback records from **Firestore**.
 - Display the feedbacks in a **tabular format** for all users[9].

Flowchart:



Figure 6: Flowchart of Module 6 data analysis

Functional Testing:

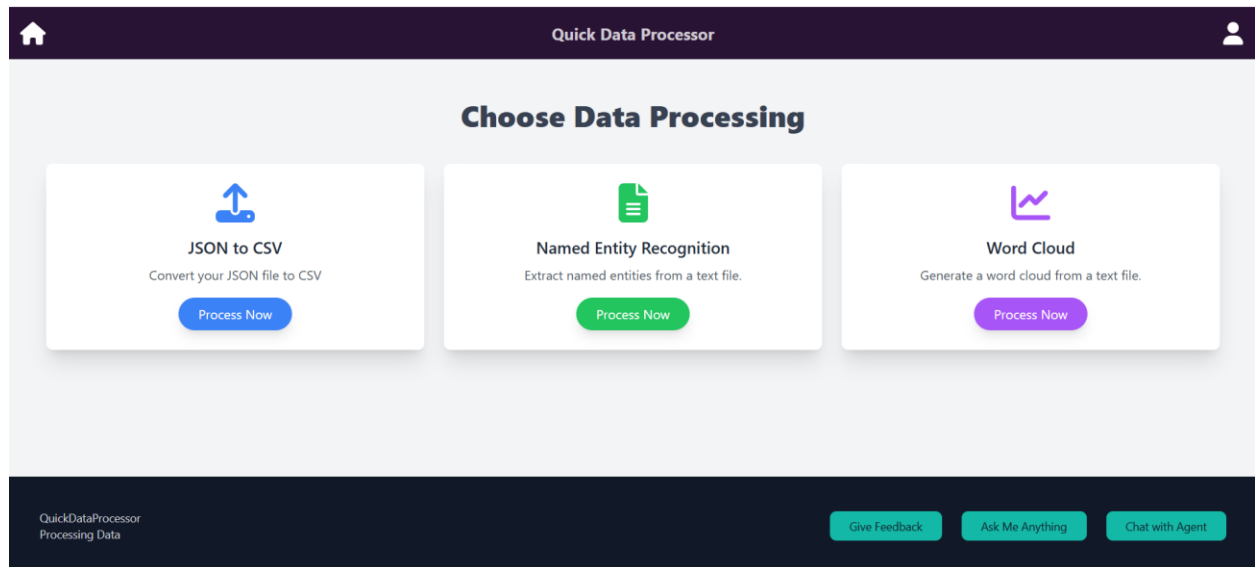


Figure 7: Homepage for the customer

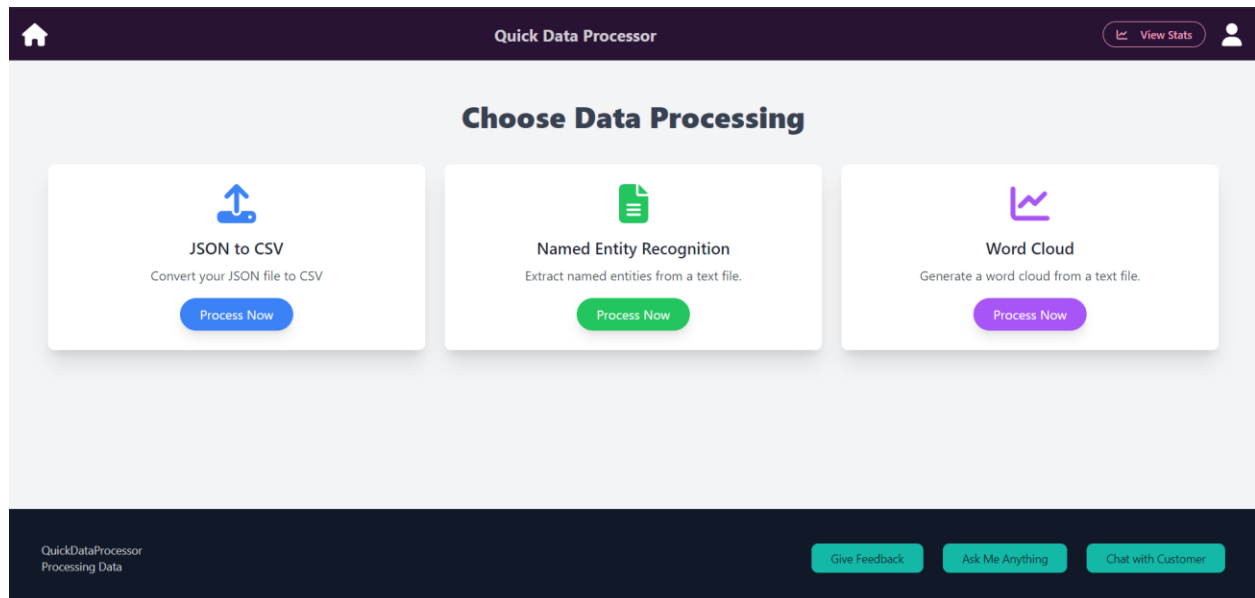


Figure 8: Homepage for QDP Agents

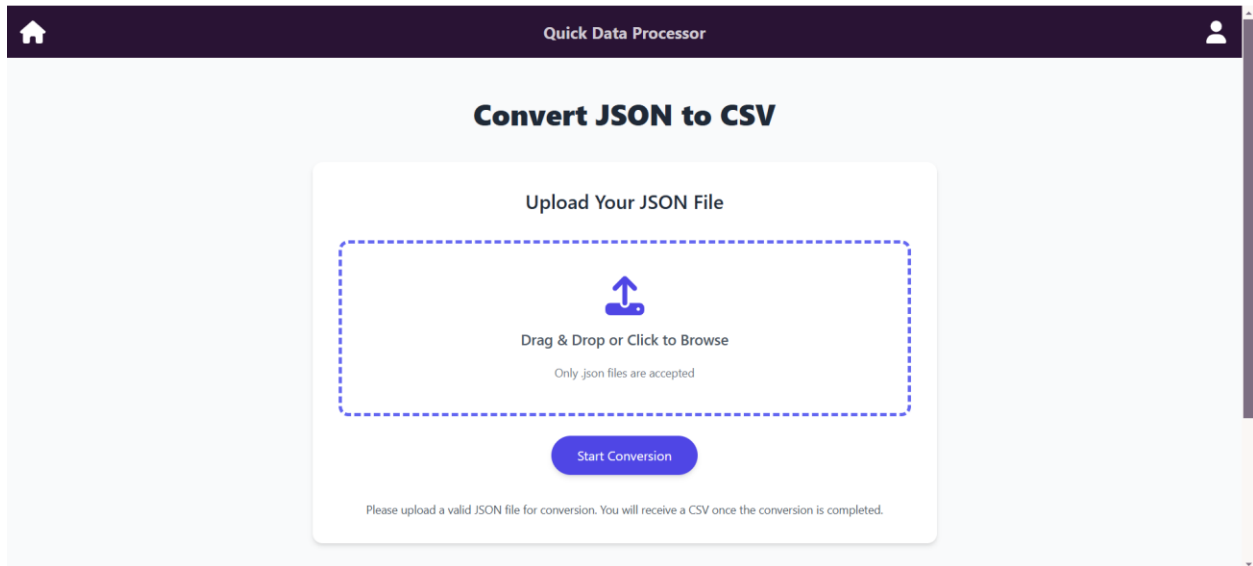


Figure 9: Data Processing 1

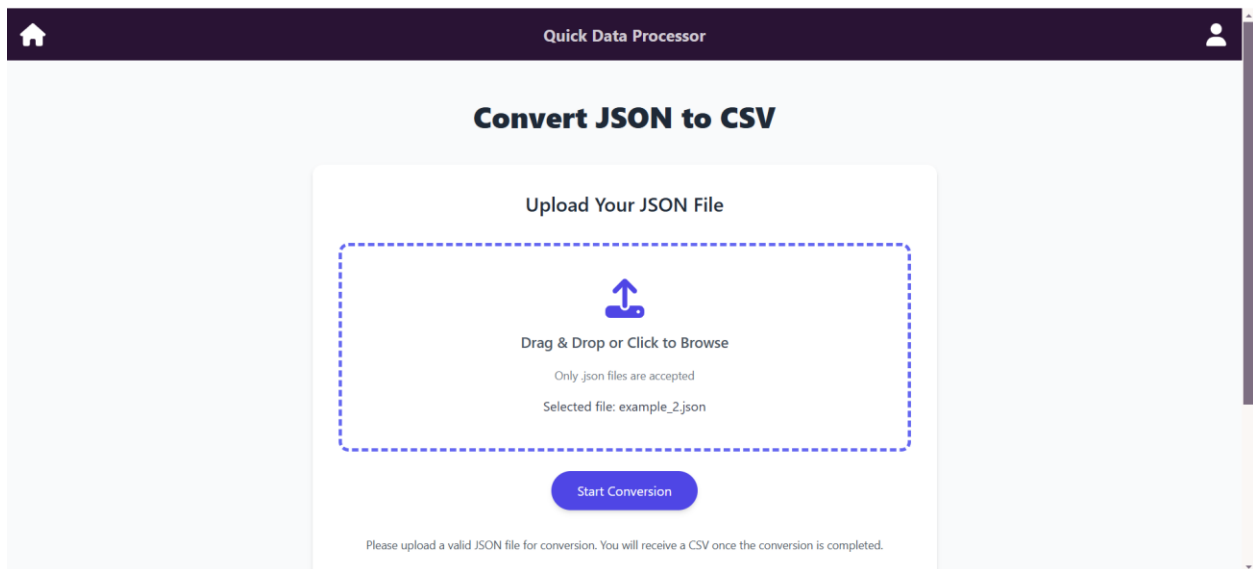


Figure 10: Uploading file in DP1

Please upload a valid JSON file for conversion. You will receive a CSV once the conversion is completed.

Get Processed File

File ID	S3 Output	Status
b6c7ed15-8e1a-4ea9-8f6b-e36cb665f32c	https://serverlessbucket-input.s3.us-east-1.amazonaws.com/output/uploads/b6c7ed15-8e1a-4ea9-8f6b-e36cb665f32c.csv	Success
d5f5376e-0089-4a1f-9c40-01beb96e4b9c	https://serverlessbucket-input.s3.us-east-1.amazonaws.com/output/uploads/d5f5376e-0089-4a1f-9c40-01beb96e4b9c.csv	Success
c7423353-09a6-4cd4-aff9-699897db6c68	https://serverlessbucket-input.s3.us-east-1.amazonaws.com/output/uploads/c7423353-09a6-4cd4-aff9-699897db6c68.csv	Success



QuickDataProcessor
Processing Data

Give Feedback


Ask Me Anything

Chat with Agent

Figure 11: Processed File for DP1

Quick Data Processor

Named Entity Recognition



Upload Your Text File

Drag & Drop or Click to Browse

Only .txt files are accepted

Start Extraction

Upload a valid .txt file to extract Named Entities. Results will be stored in DynamoDB.

Get Processed File

Figure 12: Data Processing 2

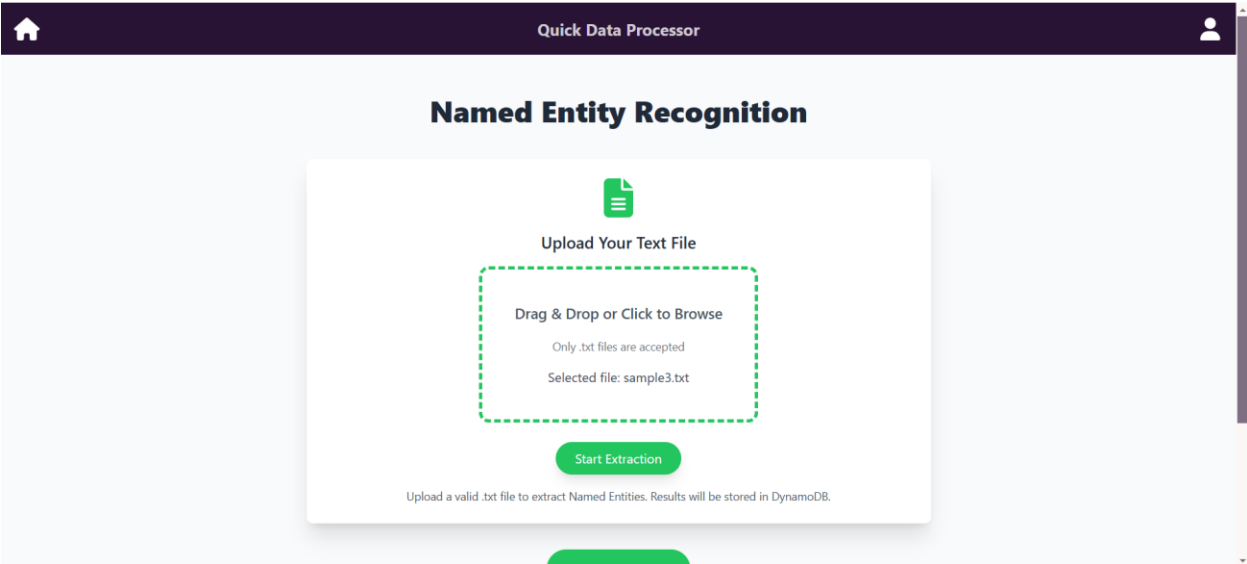


Figure 13: Uploading File in DP2

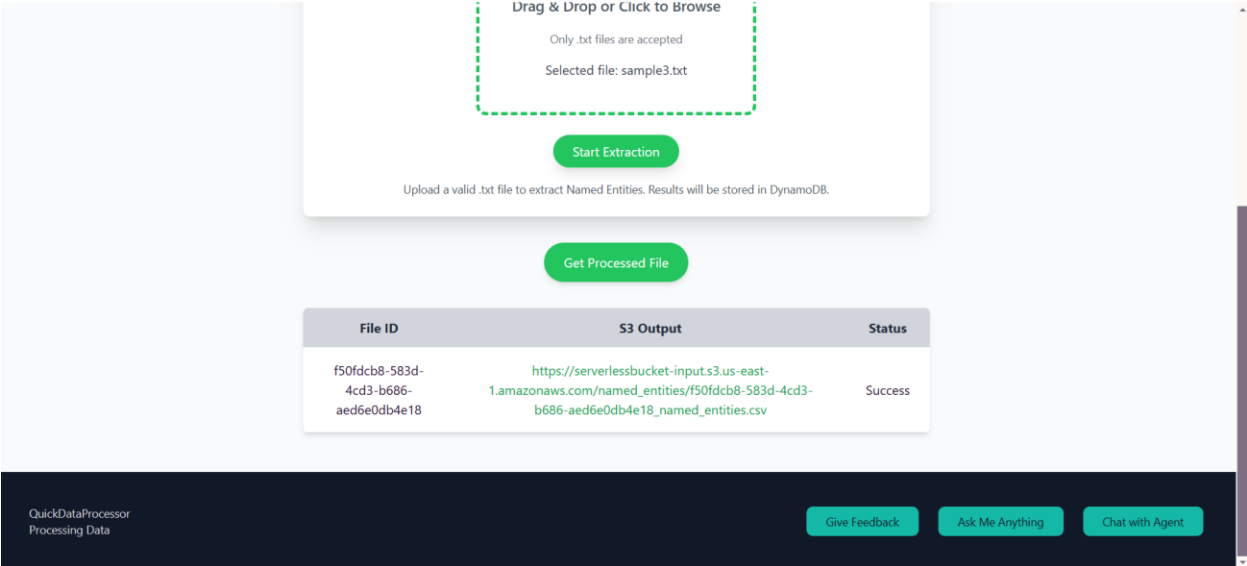


Figure 14: Processed Files for DP2

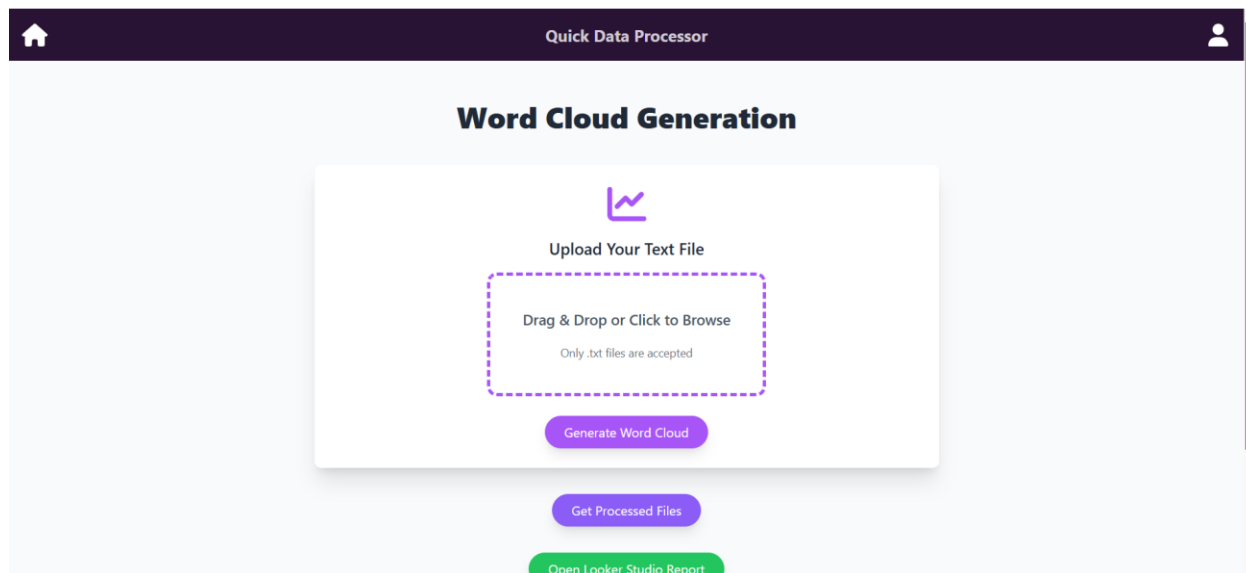


Figure 15: Data Processing 3

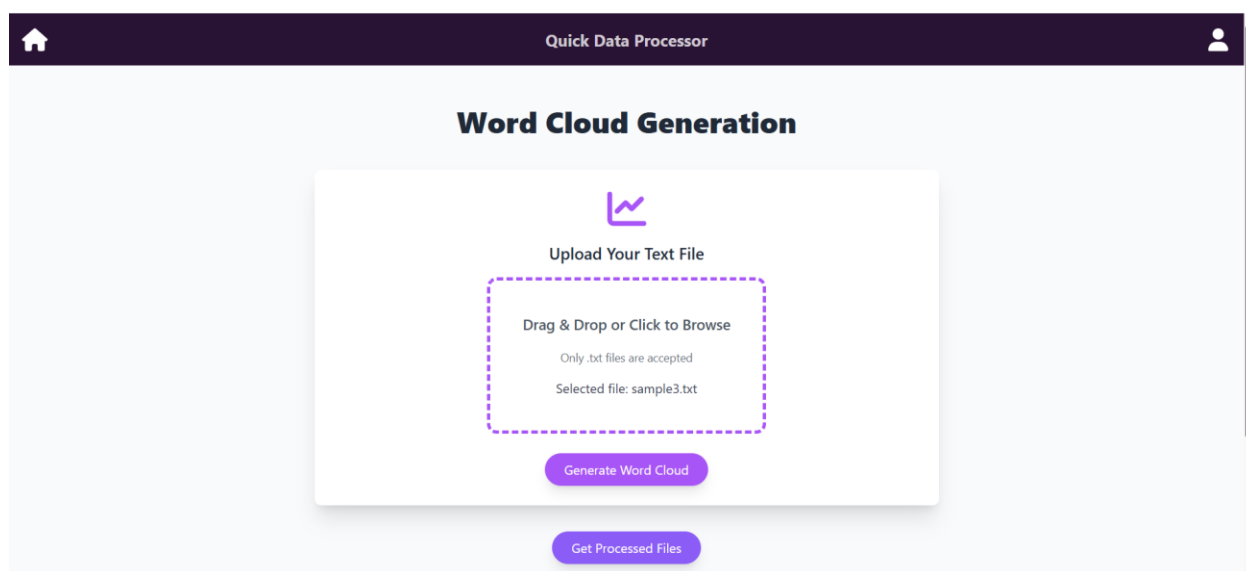


Figure 16: Uploading file in DP3

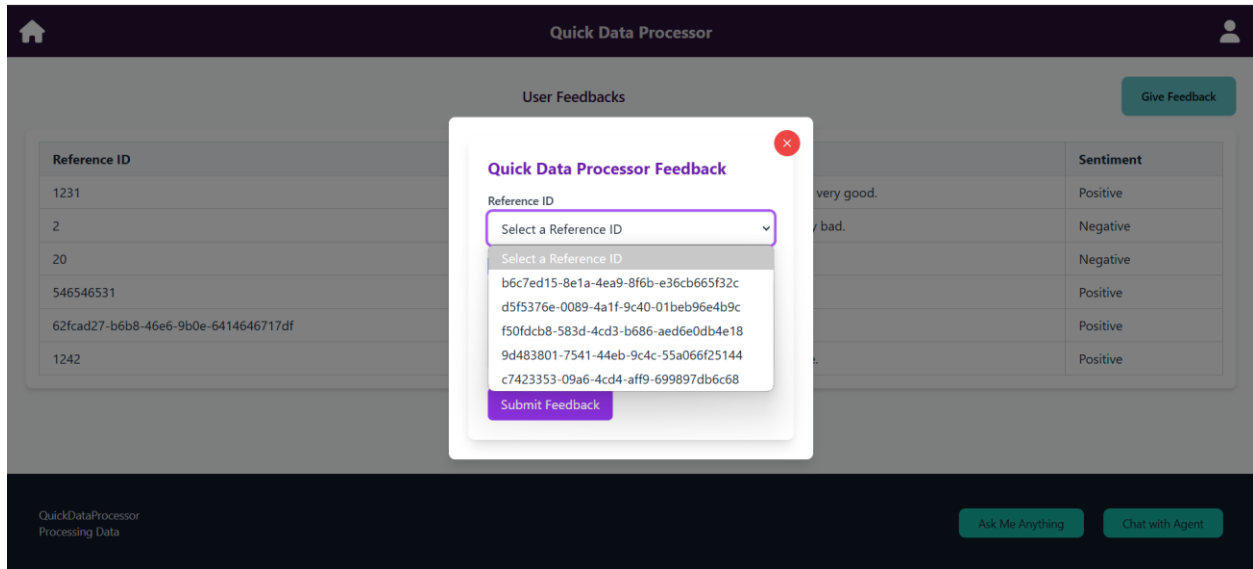


Figure 19: Feedback Form and displaying processed files

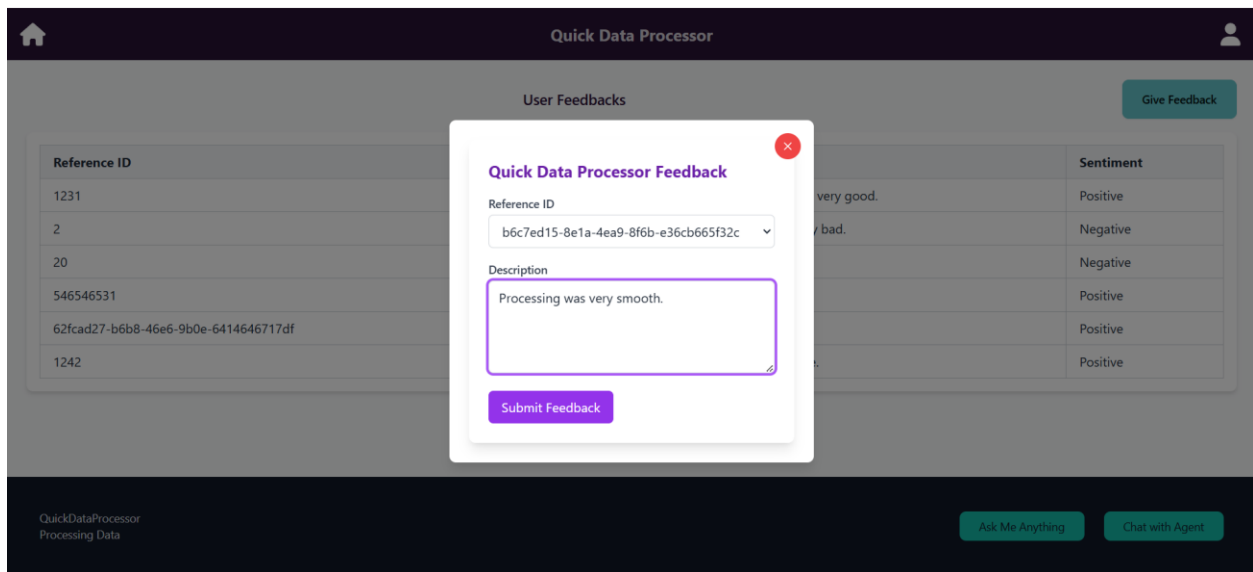


Figure 20: Giving Feedback

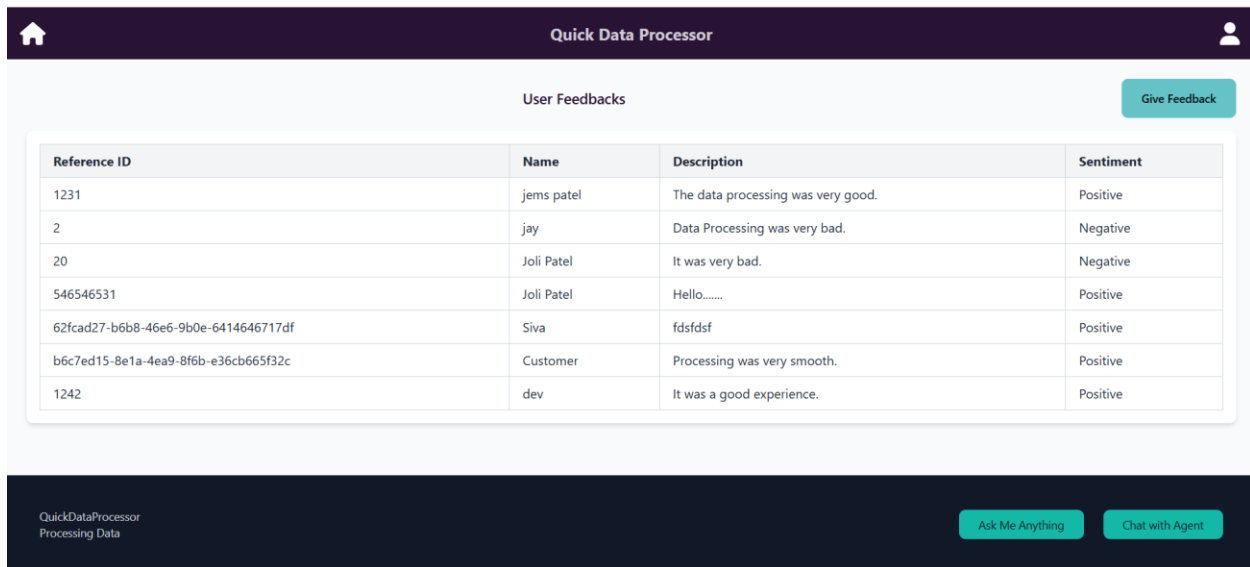


Figure 21: Tabular Format of All Feedbacks

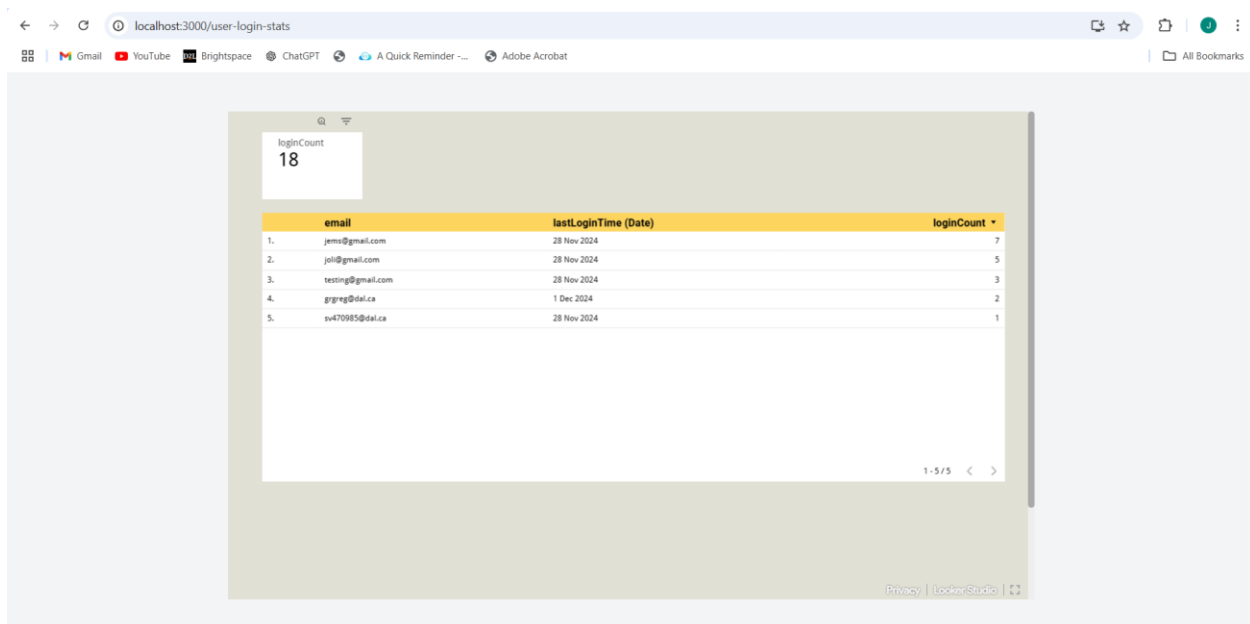


Figure 22: Looker studio Dashboard for the QDP Agents for showing User Stats

Quick Data Processor

Add New Concern

Select Reference ID

Select a Reference ID

Enter your concern

Add Concern

Your Concerns

Concern ID: WSR0gH6qhEmRIqQPjCHF

Concern: Km cho

Agent Name: Admin

Concern ID: fLzd8ILUCY8w4r54oEZ3

Concern: Hii

Agent Name: John

Concern ID: iJsgVr4RZ41KV5ZEbgST

Concern: Hello

Agent Name: Admin

Figure 23: Pub-Sub Chat Page for Customer

Quick Data Processor

Add New Concern

Select Reference ID

Select a Reference ID

b6c7ed15-8e1a-4ea9-8f6b-e36cb665f32c

d5f5376e-0089-4a1f-9c40-01beb96e4b9c

f50fdbcb8-583d-4cd3-b686-aed6e0db4e18

9d483801-7541-44eb-9c4c-55a066f25144

c7423353-09a6-4cd4-aff9-699897db6c68

Add Concern

Your Concerns

Concern ID: WSR0gH6qhEmRIqQPjCHF

Concern: Km cho

Agent Name: Admin

Concern ID: fLzd8ILUCY8w4r54oEZ3

Concern: Hii



Agent Name: John

Concern ID: iJsgVr4RZ41KV5ZEbgST

Concern: Hello

Agent Name: Admin

Figure 24: Creating concerns

Quick Data Processor

Add New Concern

Select Reference ID

b6c7ed15-8e1a-4ea9-8f6b-e36cb665f32c

Hello I have an issue with one processed file.

Add Concern

Your Concerns

Concern ID: WSR0gH6qhEmRlqQPjCHF

Concern: Km cho

Agent Name: Admin

Concern ID: fLzd8lLUCY8w4r54oEZ3

Concern: Hii



Agent Name: John

Concern ID: iJsgVr4RZ41KV5ZEbgST

Concern: Hello

Agent Name: Admin

Figure 25: Filling concerns form

Quick Data Processor

Add New Concern

Select Reference ID

Select a Reference ID

Enter your concern

Add Concern

Your Concerns

Concern ID: WSR0gH6qhEmRlqQPjCHF

Concern: Km cho

Agent Name: Admin

Concern ID: fLzd8lLUCY8w4r54oEZ3

Concern: Hii

Agent Name: John

Concern ID: iJsgVr4RZ41KV5ZEbgST

Concern: Hello

Agent Name: Admin

Concern ID: zCqt8KBQejE6lVtkBrBU

Concern: Hello I have an issue with one processed file.

Agent Name: Admin

Figure 26: Display all the concerns

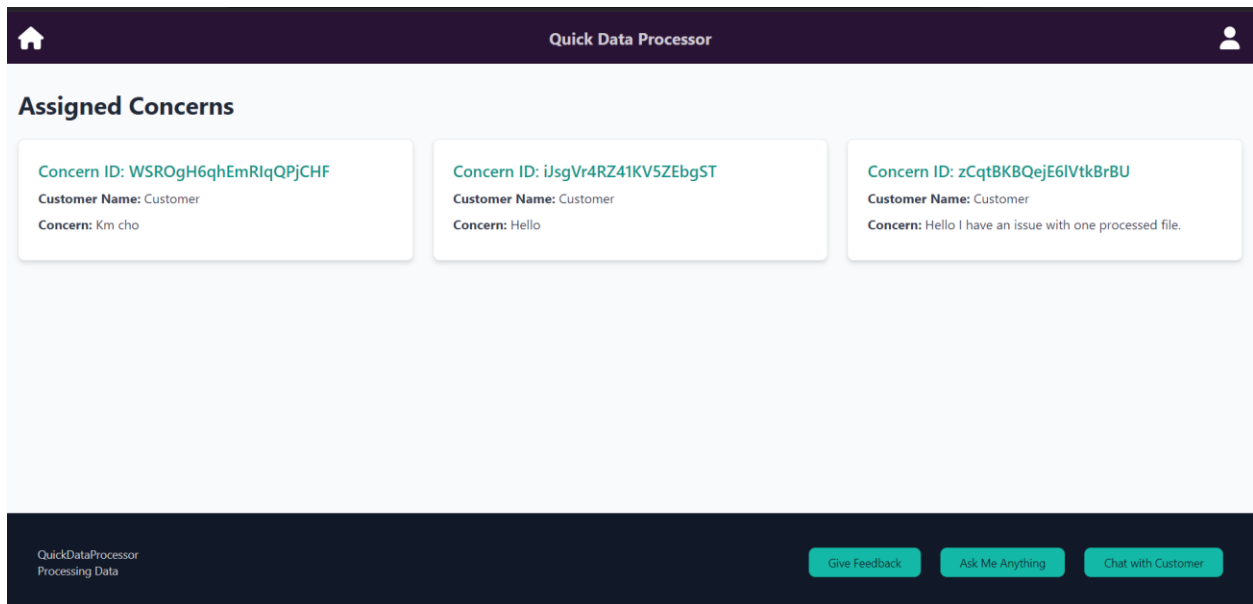


Figure 27: Pub Sub Chat page for the QDP Agents

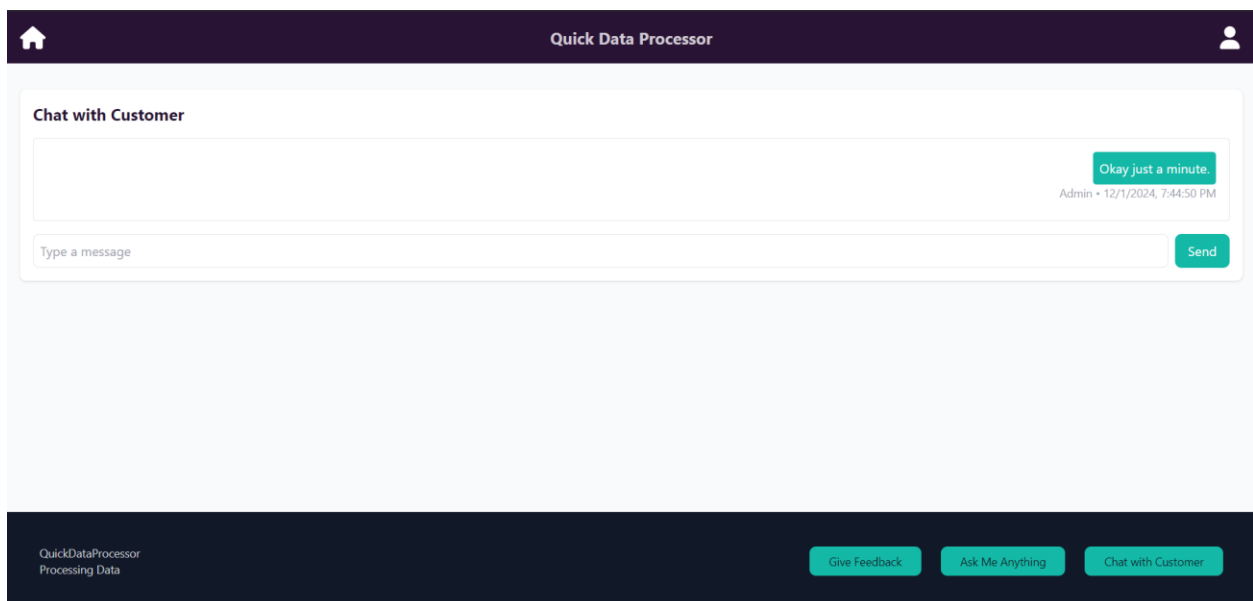


Figure 28: Admin Start Conversation

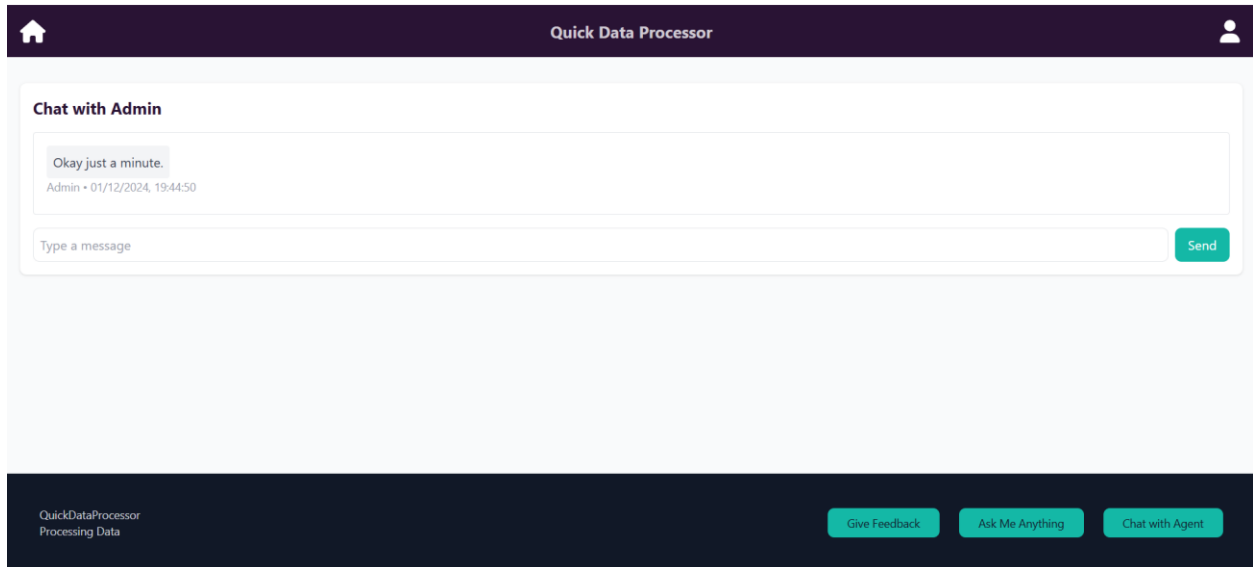


Figure 29: Customer Got Message

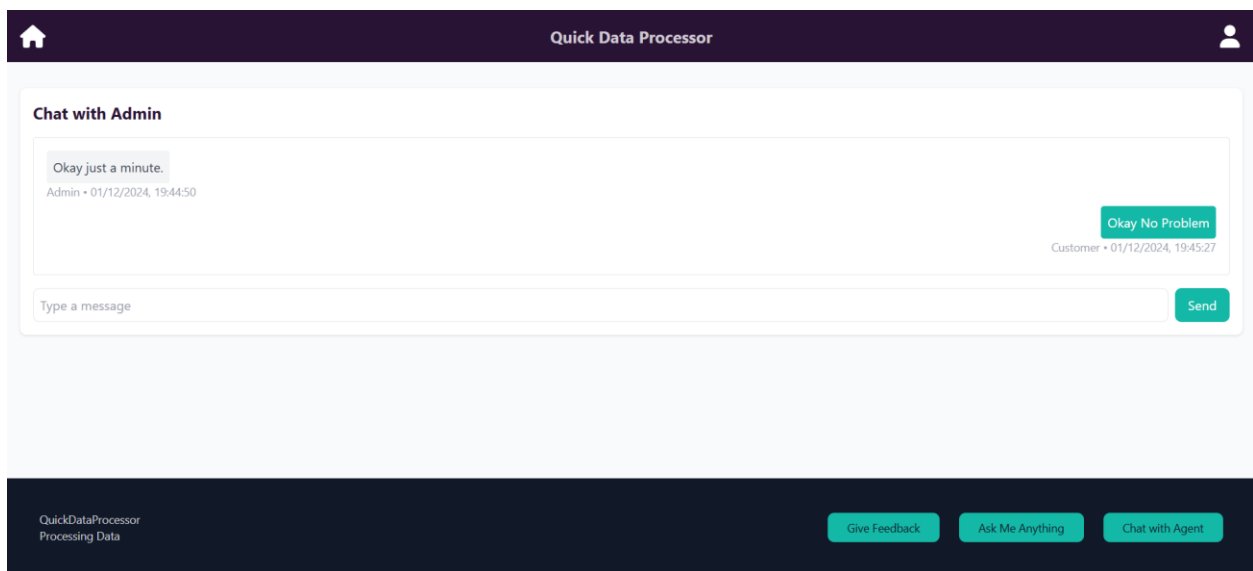


Figure 30: Send message to Admin

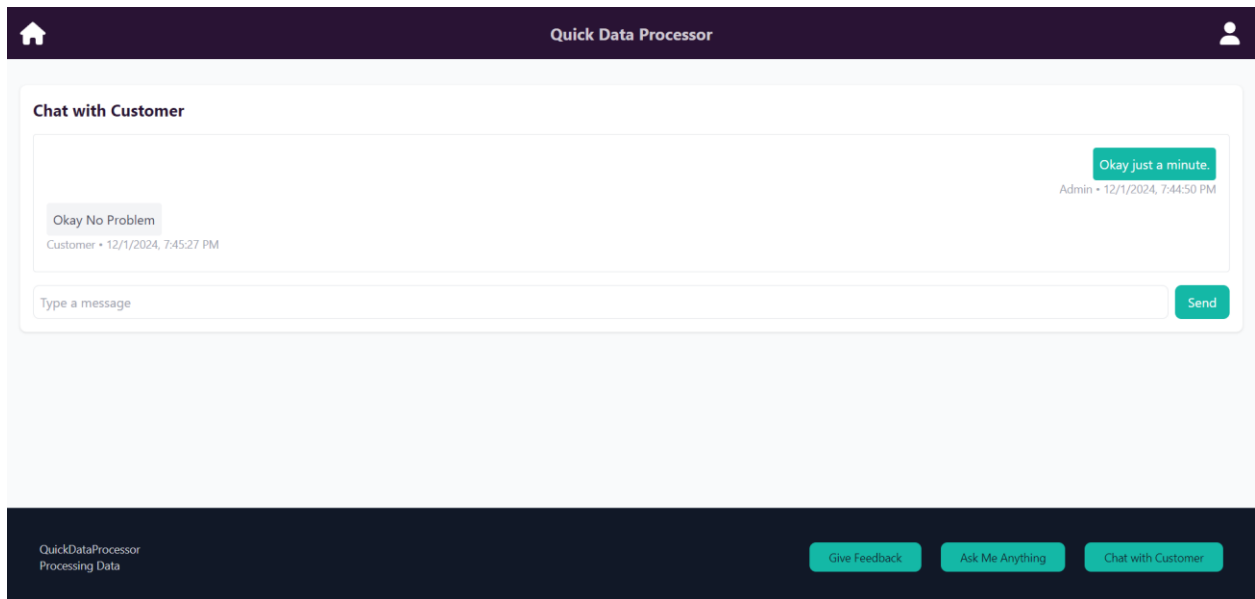


Figure 31: Admin got message

References:

- [1] Google, "Firebase Documentation," Firebase, 2024. [Online]. Available: <https://firebase.google.com/docs/>. [Accessed: Dec. 1, 2024].
- [2] Google, "Google Cloud Pub/Sub: Reliable messaging for event-driven systems," Google Cloud, 2024. [Online]. Available: <https://cloud.google.com/pubsub>. [Accessed: Dec. 1, 2024].
- [3] Google, "Google Cloud Functions: Event-driven serverless compute platform," Google Cloud, 2024. [Online]. Available: <https://cloud.google.com/functions>. [Accessed: Dec. 2, 2024].
- [4] Amazon Web Services, "AWS Glue: Prepare and transform data for analytics," Amazon, 2024. [Online]. Available: <https://aws.amazon.com/glue/>. [Accessed: Dec. 2, 2024].
- [5] Amazon Web Services, "AWS Lambda: Run code without provisioning servers," Amazon, 2024. [Online]. Available: <https://aws.amazon.com/lambda/>. [Accessed: Dec. 2, 2024].
- [6] Amazon Web Services, "Amazon Simple Notification Service (SNS)," Amazon, 2024. [Online]. Available: <https://aws.amazon.com/sns/>. [Accessed: Dec. 1, 2024].
- [7] Amazon Web Services, "Amazon Simple Queue Service (SQS)," Amazon, 2024. [Online]. Available: <https://aws.amazon.com/sqs/>. [Accessed: Dec. 2, 2024].
- [8] Google, "Cloud Natural Language API: Analyze text using machine learning," Google Cloud, 2024. [Online]. Available: <https://cloud.google.com/natural-language>. [Accessed: Dec. 2, 2024].
- [9] Google, "Google BigQuery: Serverless, highly scalable data warehouse," Google Cloud, 2024. [Online]. Available: <https://cloud.google.com/bigquery>. [Accessed: Dec. 1, 2024].
- [10] Google, "Looker Studio: Modern business intelligence and analytics platform," Google Cloud, 2024. [Online]. Available: <https://looker.com/>. [Accessed: Dec. 1, 2024].
- [11] M. Wenzel, B. Lorenz, and J. Gersdorf, *draw.io*. [Online]. Available: <https://app.diagrams.net/>. [Accessed: 02-Dec-2024].
- [12] Google Cloud, "Dialogflow: Conversational AI Platform," [Online]. Available: <https://cloud.google.com/dialogflow/>. [Accessed: 02-Dec-2024].
- [13] Amazon Web Services, Inc., "Amazon DynamoDB – NoSQL Database," [Online]. Available: <https://aws.amazon.com/dynamodb/>. [Accessed: 02-Dec-2024].

- [14] Amazon Web Services, Inc., "Amazon Cognito – Identity and Access Management for Web and Mobile Apps," [Online]. Available: <https://aws.amazon.com/cognito/>. [Accessed: 02-Dec-2024].
- [15] J. Daniel, *Word Cloud Generator: Visualizing Text Data*, [Online]. Available: <https://www.wordclouds.com/>. [Accessed: 02-Dec-2024].