Sentiment-Aware Chatbot using AWS

Shantanu Deshmukh

Donald R. Tapia College of Business, Saint Leo University

COM-571-MSC02: Intro Artificial Intelligence

Ustymenko Stanislav

April 23, 2025

Abstract: -

In this project, I have developed a sentiment-aware chatbot system using Amazon Lex, AWS Lambda, Amazon Comprehend, and DynamoDB. The chatbot is deployed on a demo website through Kommunicate. It collects customer feedback and reviews. It analyses their sentiment and stores all information in DynamoDB. This document explains the detailed process that I followed to build and deploy this chatbot.

1. Introduction: -

Chatbots already provide quick, efficient responses to customer inquiries around the clock. But what if these digital helpers could pick up on subtle nuances of human emotion? That's where sentiment analysis comes into play. By incorporating this technology, chatbots can decipher the emotional tone behind customer messages, opening a new dimension of understanding and responsiveness.

Imagine a frustrated customer reaching out for support. A sentiment-aware chatbot can detect their distress and adjust its approach, accordingly, offering more empathetic responses or swiftly escalating the issue to a human agent. This level of emotional intelligence in automated systems is becoming essential as customers expect personalized and considerate service at every turn.(*SmythOS - Chatbots and Sentiment Analysis*, 2024)

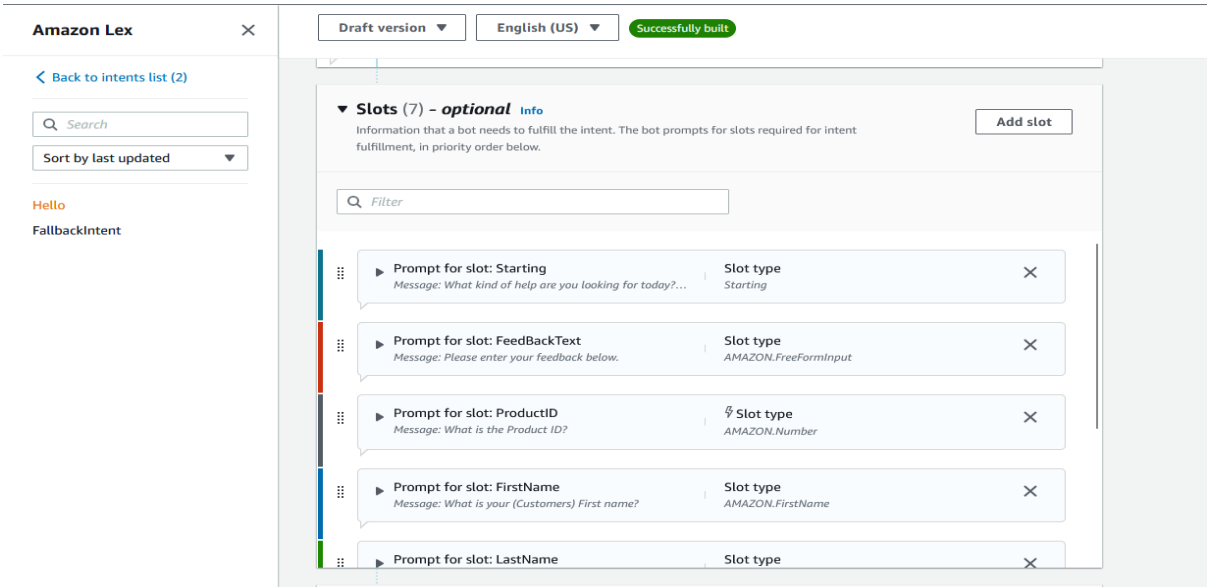
Nowadays, businesses need smarter customer support systems that can detect customer emotions and take required actions. So, I created a chatbot that accepts customer issues or feedback and understands the user sentiment using NLP. Application also stores everything in a database for later analysis. I used Amazon Web Services for backend and Kommunicate for frontend integration.

2. Technologies Used: -

- **Amazon Lex** – To create the chatbot interface.
- **AWS Lambda** – To write backend logic.
- **Amazon Comprehend** – For sentiment analysis.
- **Amazon CloudWatch** – To debug and monitor the Lambda logs.
- **Amazon DynamoDB** – To store user inputs and sentiment data.
- **IAM Roles & Users** – For managing permissions and security.
- **Kommunicate** – To deploy chatbot on a demo website.
- **VS Code** – For HTML website development.

3. Amazon Lex Configuration: -

I created a Lex bot with a single intent called “Hello”. In this intent, I added slots like: -



- Starting: - It is a custom slot type to identify the type of input (Product Issue, Payment Issue, Delivery Issue, Product Return, Review and Feedback, Other etc.)
- ProductID, FirstName, LastName, Date, ZIP: - For collecting customer details.
- FeedBackText: - Required only when user selects "Review and Feedback".
- Confirmation: - Configured Prompts to confirm the intent and Responses when the user declines the intent.
- Fulfillment: - Run a lambda function to fulfil the intent and inform users of the status

when it's complete. It shows the messages on successful fulfilment and In case of failure.

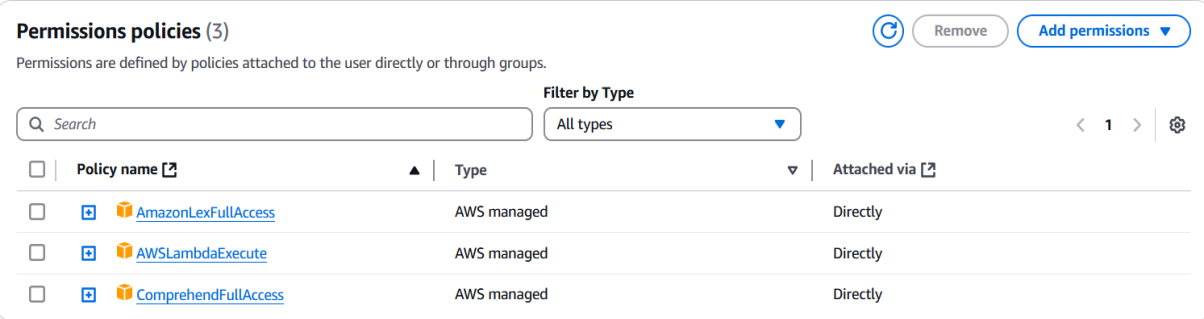
I enabled Fulfillment Code Hook so that it can call Lambda function.(*What Is Amazon Lex V2? - Amazon Lex*, n.d.)

4. IAM Configuration: -

I created an IAM user named comprehend_user. Then, I attached the following policies: -

- AmazonLexFullAccess
- ComprehendFullAccess
- AmazonLambdaExecute

This IAM user was later used to connect Kommunicate with my Lex bot.(*AWS Systems Manager Automation Runbook Reference - User Guide*, n.d.)



5. Lambda Backend Development: -

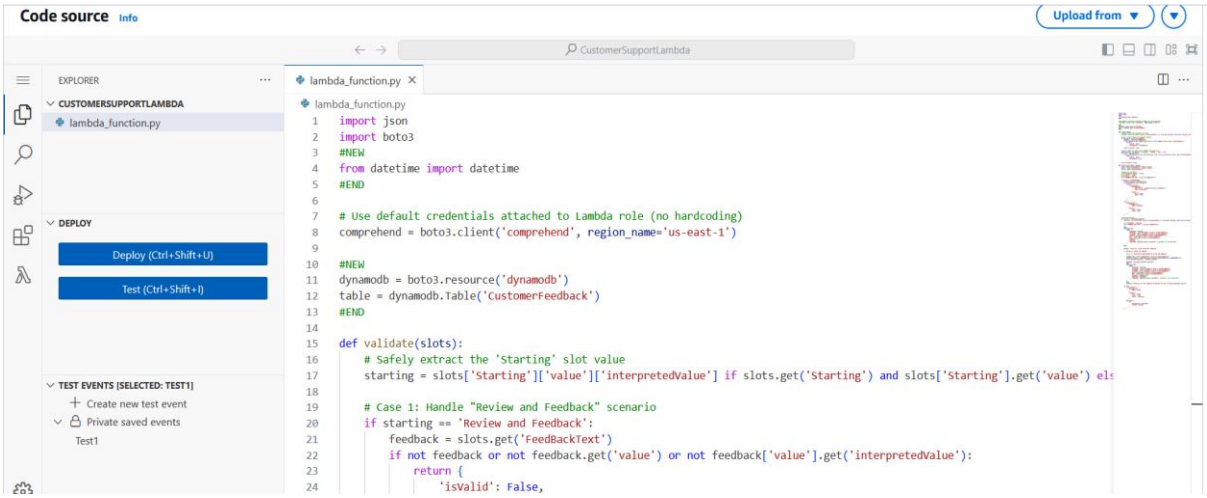
I wrote a code for Python Lambda function which handles: -

- Slot validation.
- Detecting sentiment using Amazon Comprehend.
- Saving data to DynamoDB.

I did not include the whole code in this paper but here is the overall logic: -

- First, I checked if all required slots were filled.
- Then, if the starting input is "Review and Feedback", I extracted the feedback and ran sentiment analysis.
- The sentiment result and all other slot values were stored in DynamoDB.

- If the user input was like "Product issue" or "Payment issue", I skipped sentiment and saved '-' in Feedback and Sentiment fields.(*Lambda - AWS Systems Manager Automation Runbook Reference*, n.d.)



6. Amazon DynamoDB Table: -

I created a table called CustomerFeedback. The columns were: -

- IssueType: - Stores the type of issue like Product issue, Payment issue, Review and Feedback etc.
- ProductID: - Stores the product ID in number format.
- FirstName: - Stores the first name of customer.
- LastName: - Stores the last name of customer.
- Date: - Stores the date in date format.
- ZIP: - Stores customers zip code in number format.
- Review and Feedback: - Stores the customers review and feedback.
- Sentiment: - Stores the sentiment of review and feedback like negative, positive, neutral or mixed
- Timestamp: - Every time user completes the interaction, a new row is added with current UTC timestamp.(*DynamoDB - AWS Serverless Application Model*, n.d.)

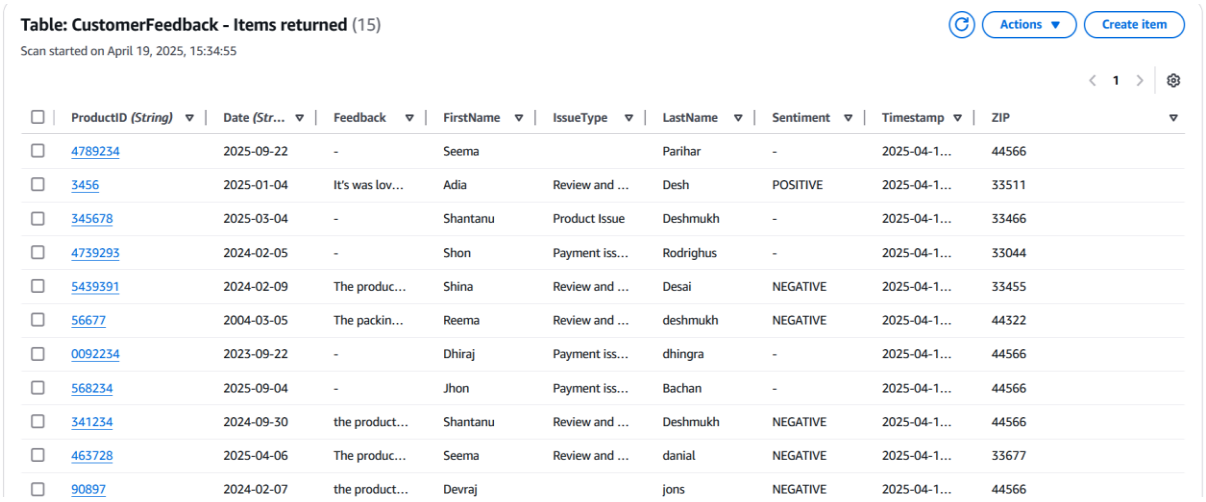


Table: CustomerFeedback - Items returned (15)

Scan started on April 19, 2025, 15:34:55

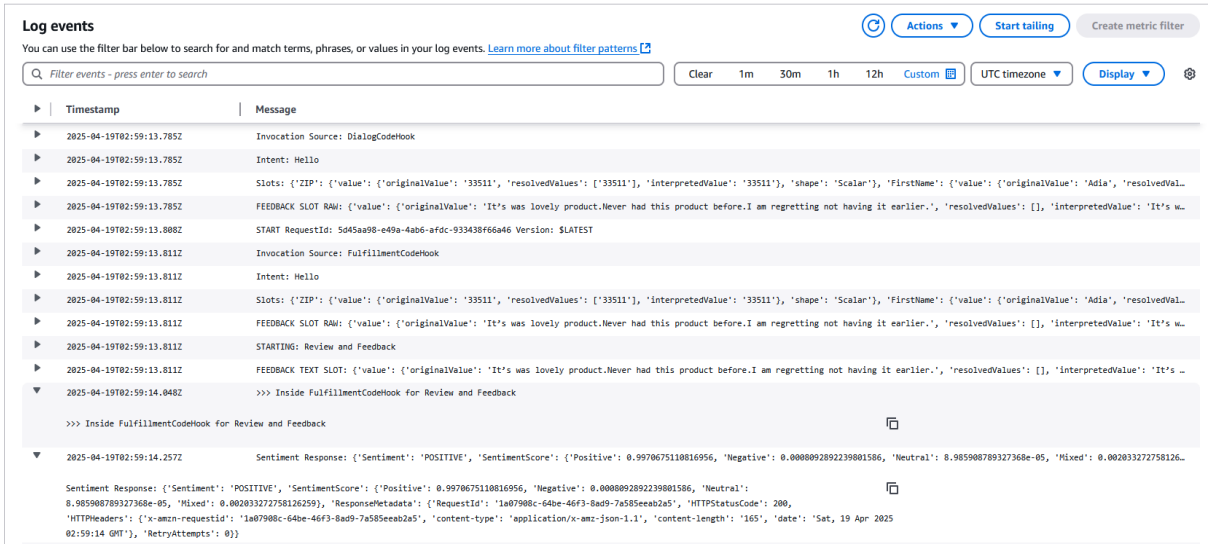
Actions ▼ | Create item

1

ProductID (String)	Date (Str...	Feedback	FirstName	IssueType	LastName	Sentiment	Timestamp	ZIP
4789234	2025-09-22	-	Seema		Parihar	-	2025-04-1...	44566
3456	2025-01-04	It's was lov...	Adia	Review and ...	Desh	POSITIVE	2025-04-1...	33511
345678	2025-03-04	-	Shantanu	Product Issue	Deshmukh	-	2025-04-1...	33466
4739293	2024-02-05	-	Shon	Payment iss...	Rodrighus	-	2025-04-1...	33044
5439391	2024-02-09	The produc...	Shina	Review and ...	Desai	NEGATIVE	2025-04-1...	33455
56677	2004-03-05	The packin...	Reema	Review and ...	deshmukh	NEGATIVE	2025-04-1...	44322
0092234	2023-09-22	-	Dhiraj	Payment iss...	dhingra	-	2025-04-1...	44566
568234	2025-09-04	-	Jhon	Payment iss...	Bachan	-	2025-04-1...	44566
341234	2024-09-30	the product...	Shantanu	Review and ...	Deshmukh	NEGATIVE	2025-04-1...	44566
463728	2025-04-06	The produc...	Seema	Review and ...	danial	NEGATIVE	2025-04-1...	33677
90897	2024-02-07	the product...	Devraj		jons	NEGATIVE	2025-04-1...	44566

7. Monitoring using CloudWatch: -

To debug Lambda, I printed useful logs like slot values, feedback text, and sentiment response. I checked these outputs in Amazon CloudWatch under Logs section. This helped me a lot during development.(*CloudWatch - AWS Systems Manager Automation Runbook Reference*, n.d.)



Log events

You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns

Actions ▼ | Start tailing | Create metric filter

Filter events - press enter to search | Clear | 1m | 30m | 1h | 12h | Custom | UTC timezone ▼ | Display ▼

Timestamp	Message
2025-04-19T02:59:13.785Z	Invocation Source: DialogCodeHook
2025-04-19T02:59:13.785Z	Intent: Hello
2025-04-19T02:59:13.785Z	Slots: {'ZIP': {'value': {'originalValue': '33511', 'resolvedValues': ['33511'], 'interpretedValue': '33511'}, 'shape': 'Scalar'}, 'FirstName': {'value': {'originalValue': 'Adia', 'resolvedVal…
2025-04-19T02:59:13.785Z	FEEDBACK SLOT RAW: {'value': {'originalValue': 'It's was lovely product.Never had this product before.I am regretting not having it earlier.', 'resolvedValues': [], 'interpretedValue': 'It's w…
2025-04-19T02:59:13.808Z	START RequestId: 5d45aa98-e49a-4ab6-afdc-933438f66a46 Version: $LATEST
2025-04-19T02:59:13.811Z	Invocation Source: FulfillmentCodeHook
2025-04-19T02:59:13.811Z	Intent: Hello
2025-04-19T02:59:13.811Z	Slots: {'ZIP': {'value': {'originalValue': '33511', 'resolvedValues': ['33511'], 'interpretedValue': '33511'}, 'shape': 'Scalar'}, 'FirstName': {'value': {'originalValue': 'Adia', 'resolvedVal…
2025-04-19T02:59:13.811Z	FEEDBACK SLOT RAW: {'value': {'originalValue': 'It's was lovely product.Never had this product before.I am regretting not having it earlier.', 'resolvedValues': [], 'interpretedValue': 'It's w…
2025-04-19T02:59:13.811Z	STARTING: Review and Feedback
2025-04-19T02:59:13.811Z	FEEDBACK TEXT SLOT: {'value': {'originalValue': 'It's was lovely product.Never had this product before.I am regretting not having it earlier.', 'resolvedValues': [], 'interpretedValue': 'It's …
2025-04-19T02:59:14.048Z	>>> Inside FulfillmentCodeHook for Review and Feedback
	>>> Inside FulfillmentCodeHook for Review and Feedback
2025-04-19T02:59:14.257Z	Sentiment Response: {'Sentiment': 'POSITIVE', 'SentimentScore': {'Positive': 0.9970675110816956, 'Negative': 0.0008092892239801586, 'Neutral': 8.985908789327368e-05, 'Mixed': 0.002033272758126…
	Sentiment Response: {'Sentiment': 'POSITIVE', 'SentimentScore': {'Positive': 0.9970675110816956, 'Negative': 0.0008092892239801586, 'Neutral': 8.985908789327368e-05, 'Mixed': 0.002033272758126259}, 'ResponseMetadata': {'RequestId': '1a07908c-64be-46f3-8ad9-7a585eeab2a5', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amzn-requestid': '1a07908c-64be-46f3-8ad9-7a585eeab2a5', 'content-type': 'application/x-amz-json-1.1', 'content-length': '165', 'date': 'Sat, 19 Apr 2025 02:59:14 GMT'}, 'RetryAttempts': 0}}

8. Frontend Website with Kommunicate: -

To make it user-friendly, I deployed the chatbot on a simple webpage using Kommunicate. In the install section of Kommunicate application I used the predefined code in the web page. Here is the HTML code I used: -

D: > Saint Leo University > Intro to Artificial Intelligence > Website_Project.html > html > body > h1

```html
<!DOCTYPE html>
<html>

<body>
    <h1>Chatbot Demo</h1>
    <script type="text/javascript">
        (function(d, m){
            var kommunicateSettings =
                {"appId":"df80742d6ebb55e5cd42bf0296507095","popupWidget":true,"automaticChatOpenOnNavigation":true};
            var s = document.createElement("script"); s.type = "text/javascript"; s.async = true;
            s.src = "https://widget.kommunicate.io/v2/kommunicate.app";
            var h = document.getElementsByTagName("head")[0]; h.appendChild(s);
            window.kommunicate = m; m._globals = kommunicateSettings;
        })(document, window.kommunicate || {});

    </script>
</body>
</html>
```

I used VS Code to write this HTML and then opened it in a browser to see the chatbot working live.(*Installation · Developer Docs | Kommunicate*, n.d.)

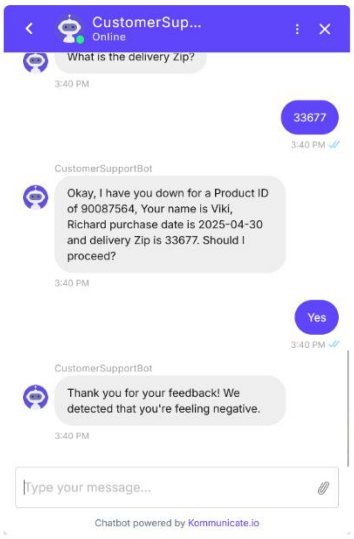
9. Final Output and Testing: -

Once everything was connected.

- The chatbot accepted issues and feedback.
- For feedback, it showed a message with detected sentiment.
- All data stored in DynamoDB.

I tested for all types of inputs like Product issue, Delivery issue, Review and Feedback, etc. Everything worked fine. Even for Review and Feedback, the sentiment was analysed by Amazon Comprehend and stored in Amazon DynamoDB correctly.

Chatbot Demo



10. What I have Learned: -

From this project, I have learned how to design and build a fully functional chatbot using real cloud technologies. Before this, I had no idea how services like Lex, Lambda, or Comprehend worked together. Now I understand how to create intents, use slots for user inputs, write backend logic in Lambda, analyse sentiment with Comprehend, and store data in DynamoDB. I also got better at debugging using CloudWatch and managing permissions with IAM.

I faced lots of challenges in this project, mainly with the fulfillment section. When Feedback was entered by the user then it was not analysed by Amazon comprehend. I thought it is because of slot type "AMAZON.FreeFormInput", which inputs the users feedback in open ended text format. To resolve this problem, I added print statements in the lambda code. After running the chatbot I checked amazon cloud logs which helped me to find where the problem is exactly. After adding multiple print statements, I was able to solve the problem. I found out that for triggering fulfilment block, all the required slot values must be entered. Problems like this taught me how to read cloud logs and trace the issue in the code.

Deploying the bot using Kommunicate and creating a frontend in VS Code was something new and fun for me. This project gave me hands-on experience with full-stack cloud development and helped me see how AI can improve customer support.

11. Applicability to Business: -

This sentiment aware chatbot can be very useful for businesses, mainly in the department of customer support and service. It will automatically handle customer complaints and feedback. So, this system will save plenty of time and improve response efficiency. After detecting customer sentiment, companies can prioritize negative feedback and take faster action on them. For example, if a customer is angry or disappointed then chatbot system can notify a human agent immediately. The stored data in DynamoDB also helps in analysing the trend of issues that are customers facing. This will help customer-facing business to improve their services on real time feedback so that they can make decisions faster. In my opinion, sentiment aware chatbot can improve customer satisfaction and reduce manual workload. and support data-driven decision-making in any customer-facing business.

12. Conclusion: -

This project helped me understand how to connect different AWS services together and use them to build a smart chatbot. The best part was integrating Amazon Comprehend with Lex, handling backend code in Lambda and then storing everything in DynamoDB. Amazon database is very easy to understand and configure. The Kommunicate tool made it easy to show the chatbot to others on a webpage.

References: -

AWS Systems Manager Automation runbook reference—User Guide. (n.d.).

CloudWatch—AWS Systems Manager Automation runbook reference. (n.d.). Retrieved April 19, 2025, from https://docs.aws.amazon.com/systems-manager-automation-runbooks/latest/userguide/automation-ref-cw.html

DynamoDB - AWS Serverless Application Model. (n.d.). Retrieved April 19, 2025, from https://docs.aws.amazon.com/serverless-application-model/latest/developerguide/sam-property-function-dynamodb.html

Installation · Developer Docs | Kommunicate. (n.d.). Retrieved April 19, 2025, from https://docs.kommunicate.io/

Lambda—AWS Systems Manager Automation runbook reference. (n.d.). Retrieved April 19, 2025, from https://docs.aws.amazon.com/systems-manager-automation-runbooks/latest/userguide/automation-ref-lam.html

SmythOS - Chatbots and Sentiment Analysis. (2024, November 5). https://smythos.com/ai-agents/chatbots/chatbots-and-sentiment-analysis/

What is Amazon Lex V2? - Amazon Lex. (n.d.). Retrieved April 19, 2025, from https://docs.aws.amazon.com/lexv2/latest/dg/what-is.html