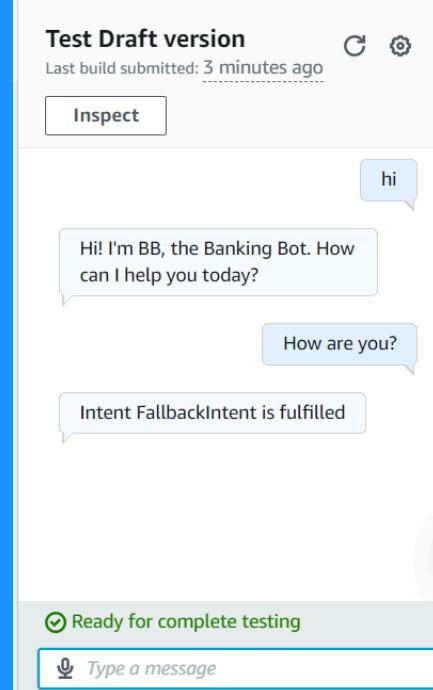




Build a Chatbot with Amazon Lex



Irfan Ansari





Introducing Today's Project!

What is Amazon Lex?

Amazon Lex is a service for building conversational interfaces using voice and text. It is useful because it enables developers to create chatbots and virtual assistants that can understand natural language

How I used Amazon Lex in this project

I used Amazon Lex in today's project, the banker bot, by integrating it to handle user interactions through natural language processing.

One thing I didn't expect in this project was...

One thing I didn't expect in this project was the level of user engagement, which led to more diverse queries than initially planned.

This project took me...

15 minutes



Setting up a Lex chatbot

3 Minutes

Basic Amazon Lex permissions are given to limit access and reduce security risks. This follows the principle of least privilege, ensuring the bot only performs necessary actions.

When you're using Amazon Lex to build a chatbot, this threshold is like a minimum score for your chatbot to confidently understand what the user is trying to say.

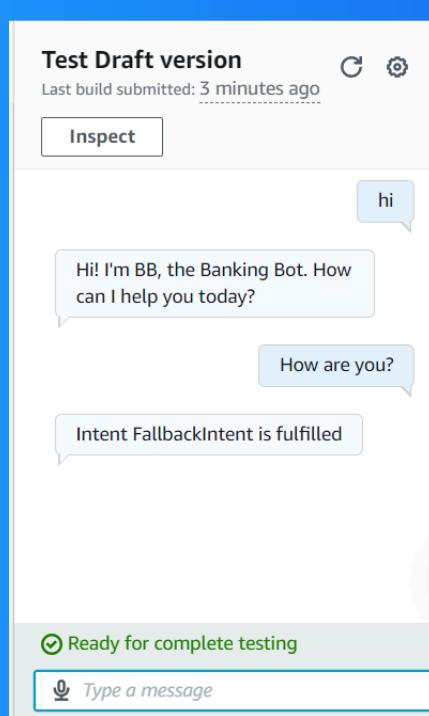
The screenshot shows the 'Add a new language' configuration page. At the top, there are four options: 'Copy from an existing language' (disabled), 'Add a language from scratch' (selected), 'Start with an example' (disabled), and 'Start with transcripts' (disabled). Below this, the 'Language details' section includes a 'Select language' dropdown set to 'English (US)' and a 'Description - optional' input field. The 'Voice' section allows selecting a voice interaction (set to 'Olivia') and provides a voice sample ('Hello, my name is Olivia. Let me know how I can assist you.') with a play button. The 'Confidence score threshold' section shows an input field for 'Intent classification confidence score threshold' set to '0.40'. At the bottom right are 'Cancel' and 'Add' buttons.



Intents

An intent is what the user is trying to achieve in their conversation with the chatbot. For example, checking a bank account balance; booking a flight; ordering food.

I created my first intent, `WelcomeIntent`, to greet users when they start interacting with the bot. It typically provides a friendly welcome message and sets the stage for further conversation





FallbackIntent

I launched and tested my chatbot, which could respond successfully if I enter "hi" with a greeting like, "Hi! I'm BB, the Banking Bot. How can I help you today?" It will also handle "How are you?" with the FallbackIntent.

My chatbot returned the error message Intent FallbackIntent is fulfilled when I entered queries it couldn't understand. This error message occurred because the chatbot was unable to match the input to any defined intents, triggering the fallback resp

The screenshot shows a chatbot interface with a blue header bar. The header includes the text "Test Draft version" and "Last build submitted: 3 minutes ago". Below the header is a button labeled "Inspect". The main area is a white space with a light gray background, showing a conversation between a user and a bot. The user's messages are in blue speech bubbles, and the bot's responses are in white speech bubbles with black outlines. The conversation goes as follows:

- User: hi
- Bot: Hi! I'm BB, the Banking Bot. How can I help you today?
- User: How are you?
- Bot: Intent FallbackIntent is fulfilled

At the bottom of the interface, there is a green checkmark icon followed by the text "Ready for complete testing". Below that is a text input field with a microphone icon and the placeholder text "Type a message".



Configuring FallbackIntent

FallbackIntent is a default intent in every chatbot that gets triggered when the user's input does not match any defined intents. It serves as a catch-all, allowing the bot to respond generically and maintain conversation flow.

I wanted to configure FallbackIntent because it helps handle unexpected user inputs effectively. This ensures that the chatbot can still engage users and guide them when their queries are not recognized.



Variations

To configure FallbackIntent, I defined it in the chatbot's intent settings and set up responses for unrecognized inputs

I also added variations! What this means for an end user is that the chatbot can recognize different ways of asking the same question, improving understanding and response accuracy.

The screenshot shows a messaging interface with a blue background. It displays two identical responses from a bot, indicating that the bot is configured to handle variations of the same question. The responses are as follows:

understanding. Can you describe what you'd like to do in a few words? I can help you find your account balance, transfer funds and make a payment.

everything good?

Sorry I am having trouble understanding. Can you describe what you'd like to do in a few words? I can help you find your account balance, transfer funds and make a payment.

At the bottom, there is a green button labeled "Ready for complete testing" with a checkmark icon. Below that is a text input field with a microphone icon and the placeholder text "Type a message".



NextWork.org

Everyone should be in a job they love.

Check out nextwork.org for
more projects

