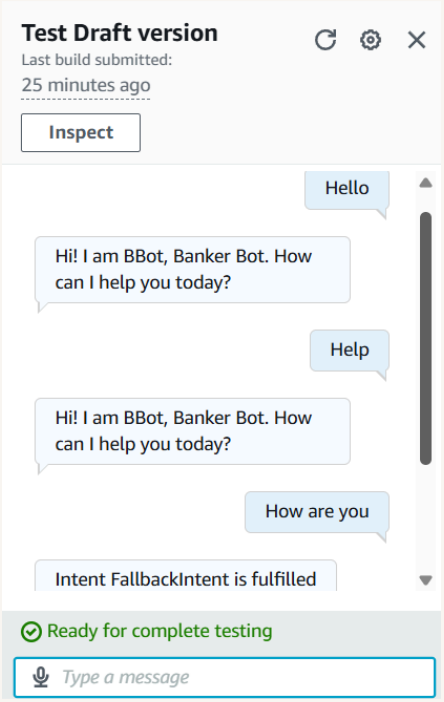
nextwork.org

Build a Chatbot with Amazon Lex

lue



Introducing Today's Project!

What is Amazon Lex?

Amazon Lex is a fully managed AI service from AWS that enables developers to build, test, and deploy conversational interfaces, uses the same advanced natural language understanding (NLU) and automatic speech recognition (ASR) technologies.

How I used Amazon Lex in this project

The services I have used are Amazon lex. The Key concepts target are intents utterances, slots, fallback and welcome intents, dialog flow, context managing, nlu, analytics, and building user-friendly, multi-turn conversations.

One thing I didn't expect was...

Amid this entire exercise, the unexpected part was being able to implement a completely functional chatbot without writing a line of code, courtesy of an intuitive visual interface and built-in natural language understanding that Amazon Lex offers.

This project took me...

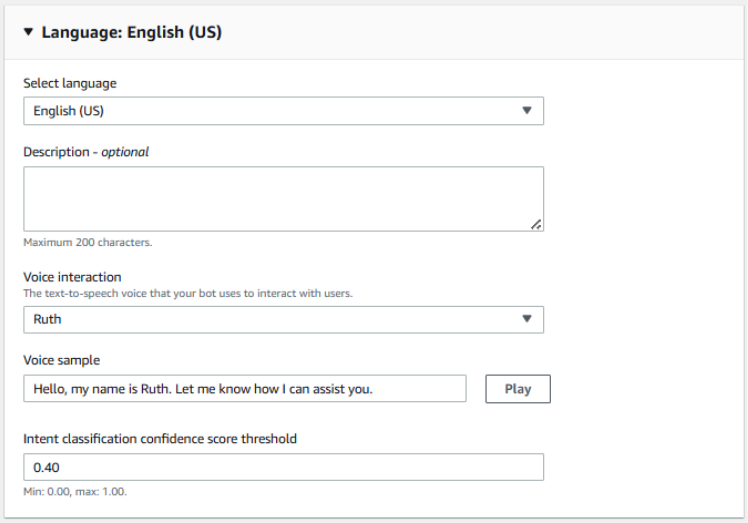
This assignment actually took me about 40 minutes. The difficult one was designing efficacious fallback responses for unexpected user input. It rewarded most in that it allowed the chatbot to come across as more natural in conversing with users.

Setting up a Lex chatbot

I created my chatbot from scratch with Amazon Lex. Setting it up took me mins including customising my fav voice of chatbot in it.

While creating my chatbot, I also created a role with basic permissions becauseAmazon Lex needs the permission to call other AWS services later in this project series I'll be integrating Lex with another service called Lambda!

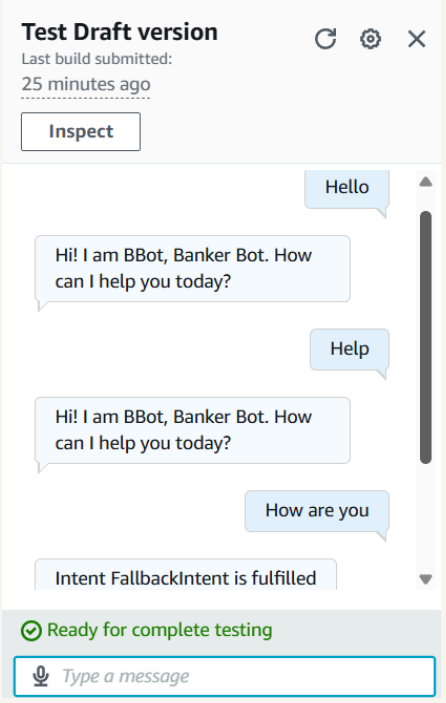
In terms of the intent classification confidence score, I kept the default value of 0.40. This means my robots needs to be at least 40% confident that it understands what the user is asking to be able to give a response. Or it'll throw error message.


Language: English (US)
Select language
English (US)
Description - optional
Maximum 200 characters.
Voice interaction
The text-to-speech voice that your bot uses to interact with users.
Ruth
Voice sample
Hello, my name is Ruth. Let me know how I can assist you.
Play
Intent classification confidence score threshold
0.40
Min: 0.00, max: 1.00.

Intents

Intents are...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.

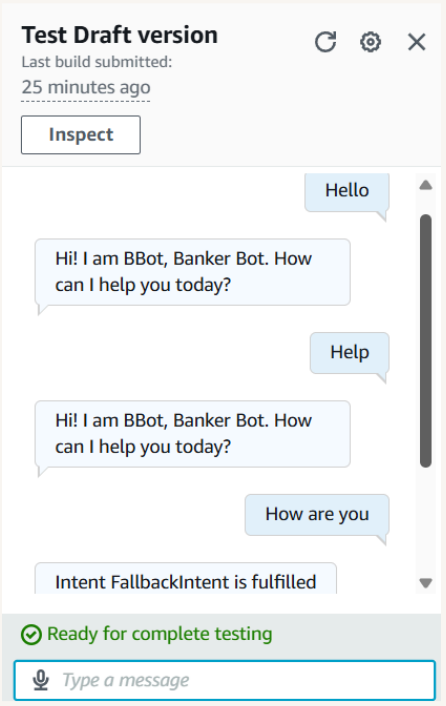
A WelcomeIntent is a special type of intent used in conversational bots and virtual agents to handle the very first interaction with a user. In this case, it would use to greet the user who has valid intent when inquiring the bot.



FallbackIntent

I launched and tested my chatbot, which could respond successfully if I enter "Help", "Hiya" which are not defined in utterances.

My chatbot returned the error message 'Intent FallbackIntent is fulfilled' when I entered"Good Morning" and "How are you". This error message occurred because the intent classificationt confidence score of the messages are below 0.4.



Configuring FallbackIntent

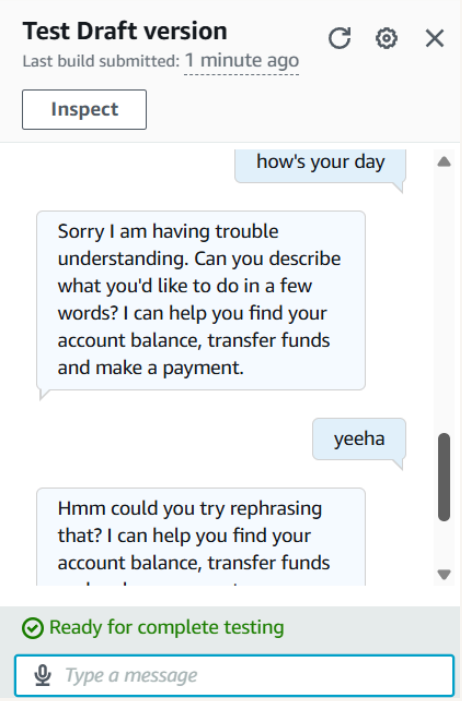
FallbackIntent is a default intent in every chatbot that gets triggered when the bot cannot recognize the user input.

I configured FallbackIntent to ensure the chatbot can handle situations where it doesn't understand or cannot match the user's input to any existing intent. The fallback intent acts as a safety net, providing a user-friendly message.

Variations

To configure FallbackIntent, I add the built-in FallbackIntent and customize the Fallback response to prompt out a user-friendly message.

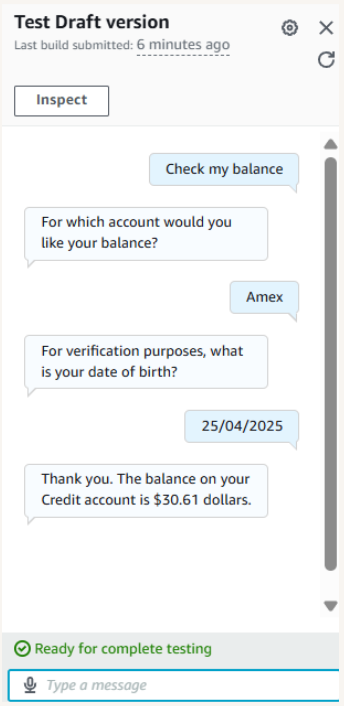
By adding variations to my FallbackIntent, ensuring that when the bot doesn't understand a user's input, it can respond with different friendly messages instead of repeating the same phrase every time, improving more dynamic and human-like response.



nextwork.org

Connect Amazon Lex with Lambda

lue



Introducing Today's Project!

What is Amazon Lex?

Amazon Lex is a service for building chatbots and virtual assistants that use voice or text. It’s useful because it understands natural language, automates conversations, scales easily, and integrates with other AWS services for flexible solutio

How I used Amazon Lex in this project

I used Amazon Lex to build a chatbot by setting up intents, sample utterances, and slots. I connected it with a Lambda function for dynamic responses, tested it using TestBotAlias, and refined the conversation flow in the Lex V2 console.

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

One thing I didn’t expect was how easy it was to connect Amazon Lex with Lambda to get real-time, dynamic answers. Testing the bot felt super smooth, and I was surprised by how well it handled different types of conversations without much extra work.

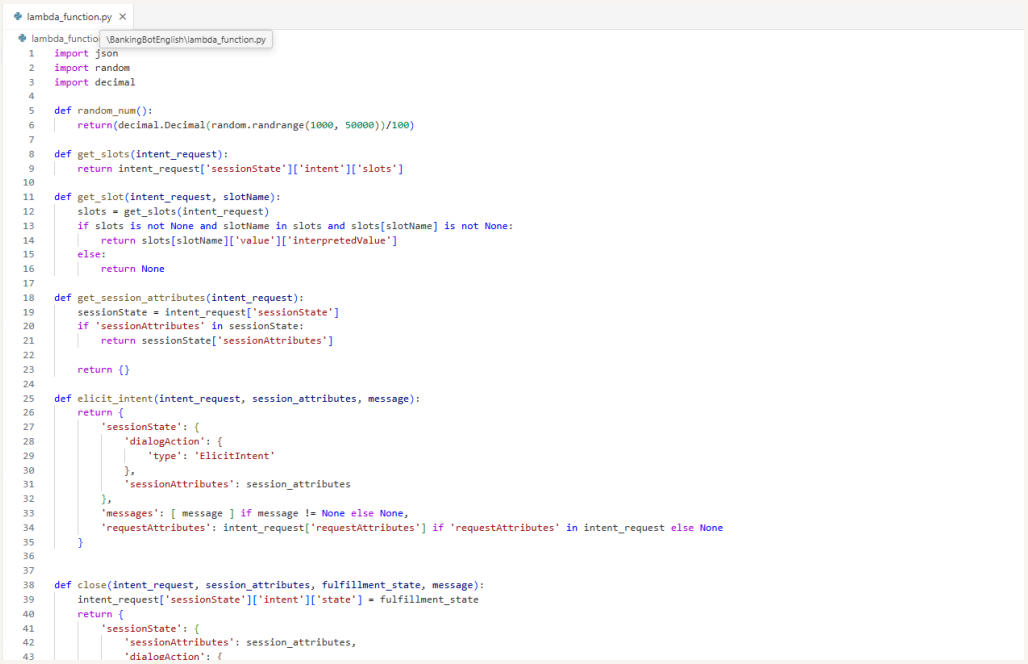
This project took me...

This project took me about a hour to complete. Setting up the chatbot with Amazon Lex, defining intents, connecting Lambda for dynamic responses, and testing the bot was straightforward and efficient, especially with pre-built templates and tools.

AWS Lambda Functions

AWS Lambda is a serverless compute service that runs the code in response to events without managing servers. It automatically scales and you pay only for the compute time used, making it ideal for building scalable, event-driven applications!

In this project, I created a Lambda function to generate a random bank account balance based on the user's account type, process slot values from Lex, and return a formatted response with the balance to the user through the chatbot.

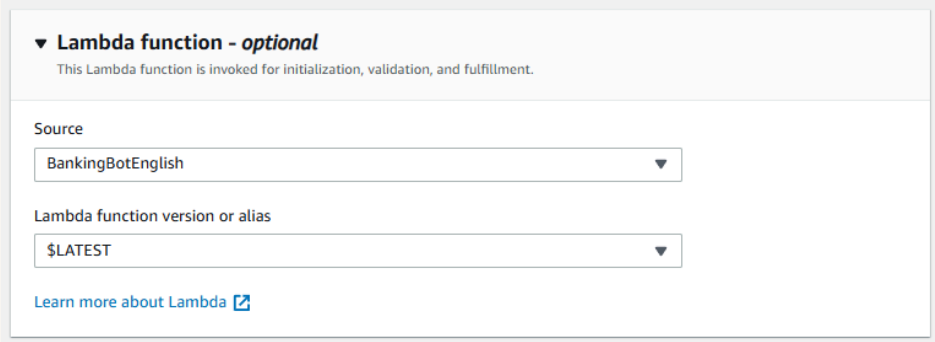


Chatbot Alias

An alias points to a specific version of an AWS Lambda function or Lex bot and enables the user to manage and deploy updates easily by pointing users to different versions without modifying the main configuration or endpoint of the bot or function.

TestBotAlias is the default alias in Amazon Lex V2 that points to the Draft version of your bot. It is used for testing and development only, not for production, and has limits on runtime requests to help validate your bot before publishing.

To connect Lambda with my BankerBot, I visited my bot's TestBotAlias and used the Lex V2 console to add my Lambda function as a code hook. This allowed the bot to trigger the Lambda function for intent fulfillment during testing and development.

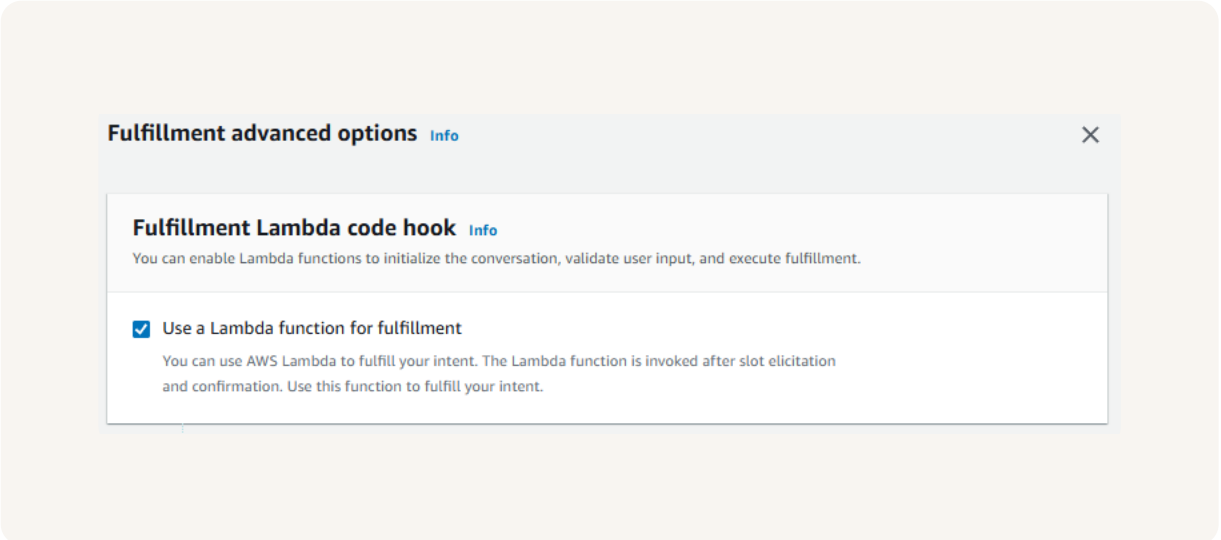


Code Hooks

A code hook is a way to run custom code, like an AWS Lambda function, during specific steps of a chatbot conversation. In Amazon Lex, code hooks let you validate user input or fulfill intents by executing your logic when certain events occur.

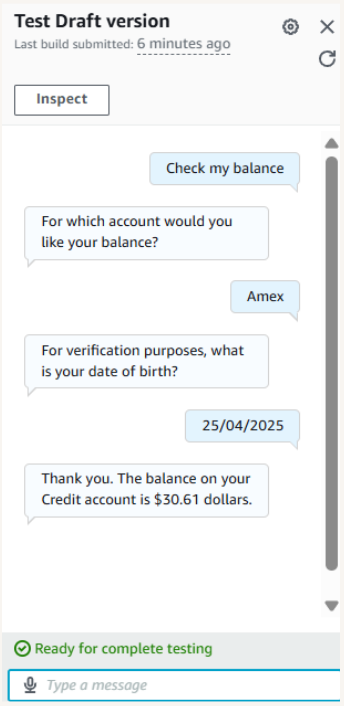
Even though I connected my Lambda with my chatbot's alias, I used code hooks to control when the Lambda runs during the conversation. Code hooks let the bot call Lambda to validate input or fulfill intents, enabling dynamic and accurate responses.

In the Amazon Lex V2 console, I could find code hooks at the intent level. There, I configured my Lambda function as a code hook invoked upon intent fulfillment, which allowed the bot to run custom logic while handling incoming user requests.



The final result!

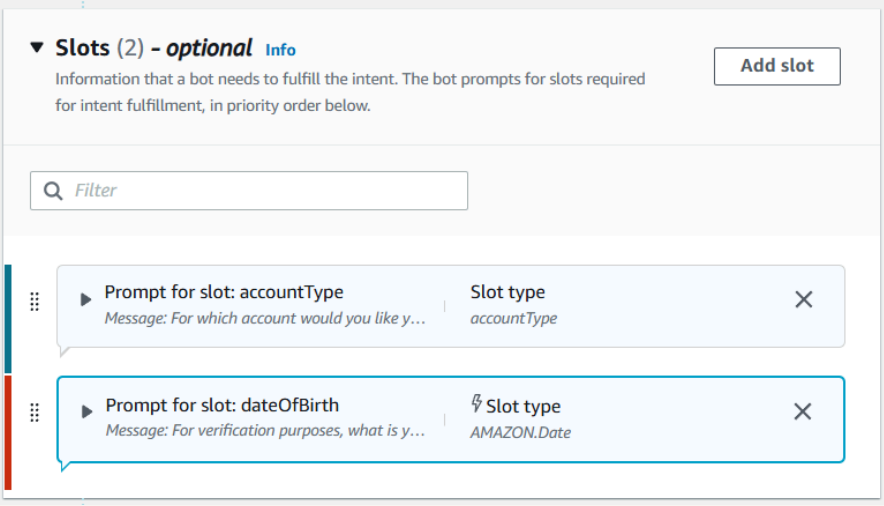
I've set up my chatbot to trigger Lambda and return a random dollar figure when a user asks to check their bank account balance and provides the required account type. The Lambda function then generates and sends the simulated balance to the user.



nextwork.org

Add Custom Slots to a Lex Chatbot

lue



Introducing Today's Project!

Today, I used Amazon Lex to build a chatbot that checks users' bank account balances. I created a custom bot, set up slots for account type and birthday, defined a CheckBalance intent, and added utterances so the bot can collect and process user inf

What is Amazon Lex?

Amazon Lex is an AWS service to build chatbots using voice or text. It understands natural language, manages conversations, and integrates with AWS services, making chatbot creation easier and scalable for customer support and automation.

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

One thing I didn't expect in this project was how intuitive and flexible Amazon Lex's slot and intent system which made capturing and validating user inputs much easier than I anticipated, streamlining the bot-building process significant.

This project took me...

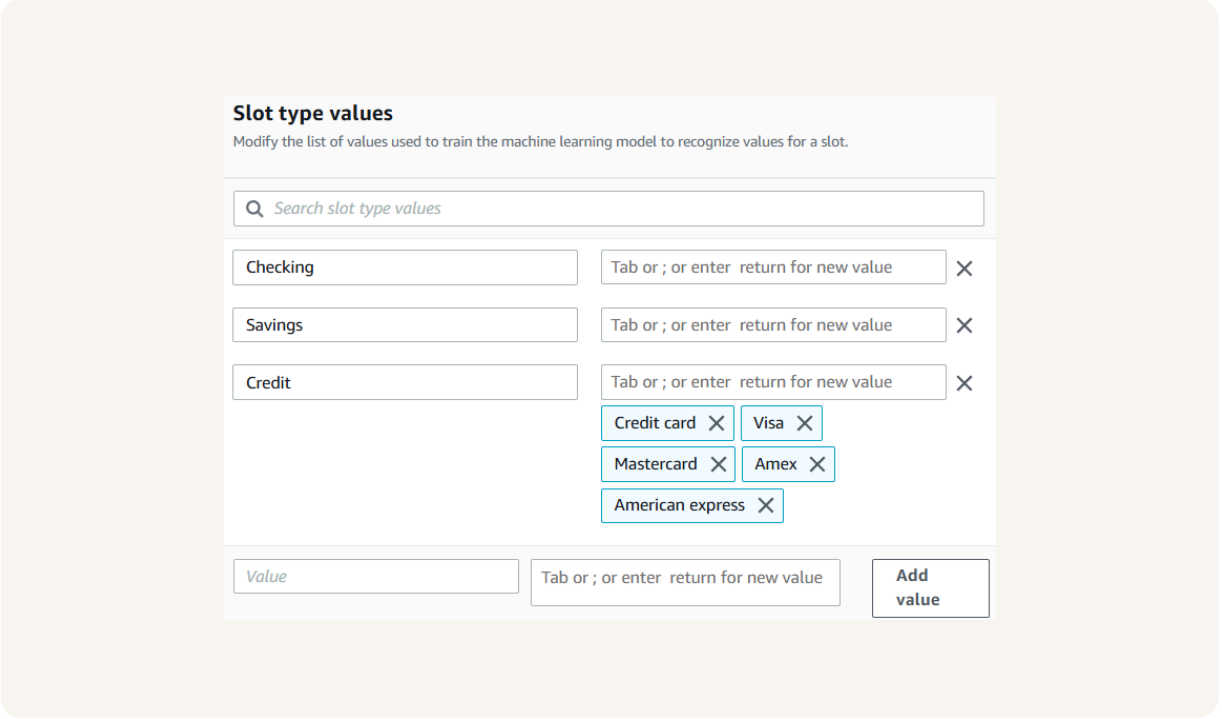
This project took me about 30 mins to complete, including setting up slots, creating intents, and testing the chatbot's responses.

Slots

What is handy for typing the types of information the bot requires, say account type or birthday? Slots in Amazon Lex are placeholders used to capture specific pieces and guarantee a bot fetches everything to process the request made by a user.

Custom slots allow users to enter specific values for your use case into the bot, making it more relevant and specific. Using the custom slots, you can define the exact entries that you want to capture, enhancing both accuracy and user experience.

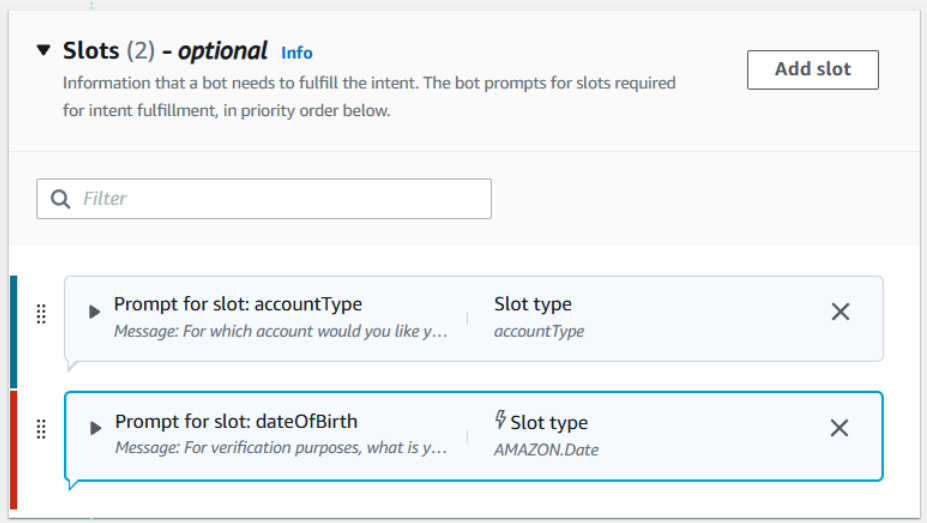
In this project, I created a custom slot type to...



Connecting slots with intents

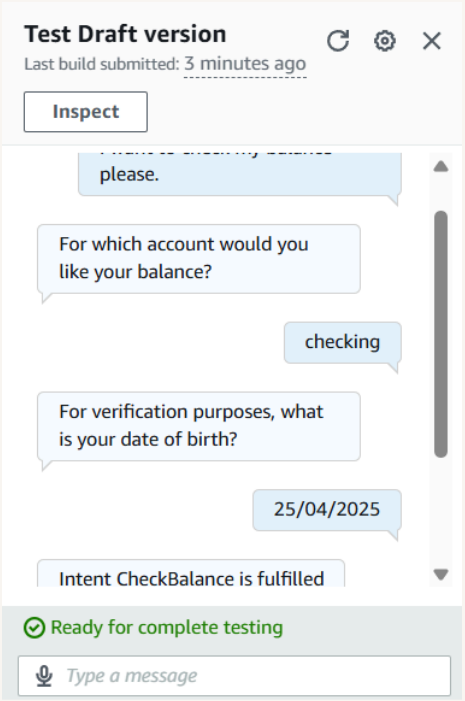
"Restrict to slot values" means the slot only accepts inputs that exactly match predefined values or synonyms. If enabled, Lex rejects any other input and prompts the user again, ensuring only valid, specific responses are accepted for that slot.

This intention called CheckBalance assists users in getting their bank balance by asking the account type and birthday via slots and collecting those to get verified by BankerBot and read the user's balance.



Slot values in utterances

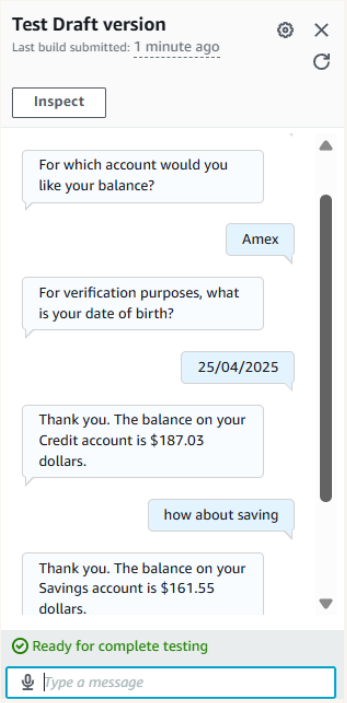
I added slot placeholders like {accountType} and {birthday} in CheckBalance's sample utterances so Lex can capture these values when users speak, ensuring the bot collects needed info to check the account balance.



nextwork.org

Save User Info with a Lex Chatbot

lue



Introducing Today's Project!

What is Amazon Lex?

Amazon Lex is an AWS service to build chatbots using voice or text. It understands natural language, manages conversations, and integrates with AWS services, making chatbot creation easier and scalable for customer support and automation.

How I used Amazon Lex in this project

I used Amazon Lex to create a CheckBalance intent that saves the user's date of birth in a context tag. This lets the FollowupCheckBalance intent access that info, so the bot remembers details and avoids asking the user repeatedly in the same session

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

One thing I didn't expect was how easily Amazon Lex handles context tags to remember user info across intents. It made the chatbot feel much smarter and saved users from repeating themselves, which really improved the overall conversation flow.

This project took me...

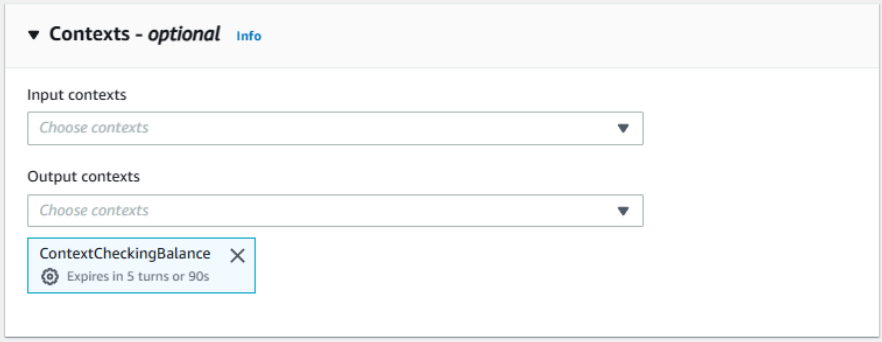
30 mins

Context Tags

Context tags are variables in Amazon Lex that help track the conversation's state. They control which intents can be triggered next by setting or checking these tags, enabling more natural, guided, and context-aware interactions in the chatbot.

There are two types of context tags: output context tags, which the bot sets to mark the current state, and input context tags, which must be active for an intent to trigger. They help manage conversation flow by controlling when intents can be used.

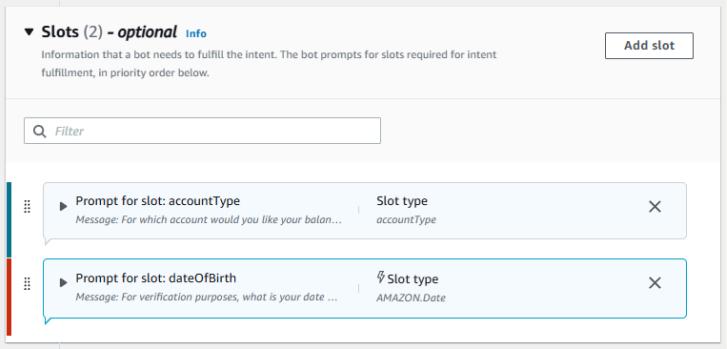
I created a context tag called "contextCheckBalance." This context tag was created in the intent "CheckBalance." It stores information about when a user is checking their balance, helping the bot manage the conversation flow for that task.



FollowUpCheckBalance

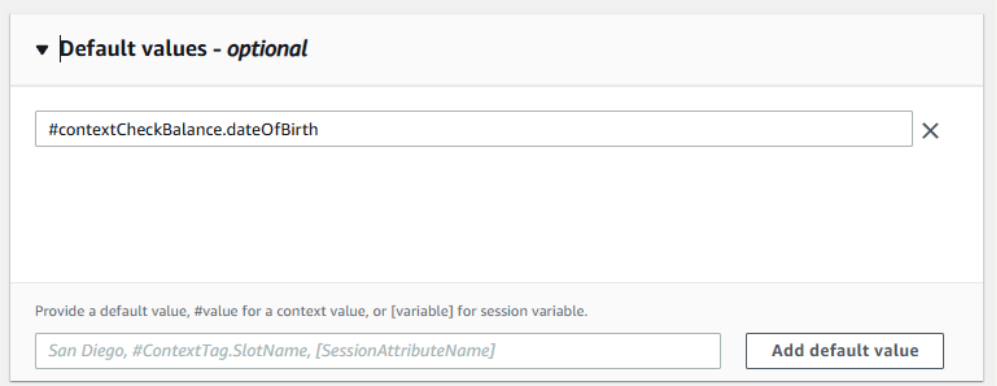
I created a new intent called FollowupCheckBalance. The purpose of this intent is to handle any follow-up questions a user might have after checking their balance, like asking for recent transactions or more account details.

This intent is connected to the previous intent I made, CheckBalance, because it uses the context tag set by CheckBalance. This allows FollowupCheckBalance to handle follow-up questions smoothly within the same conversation flow.



Input Context Tag

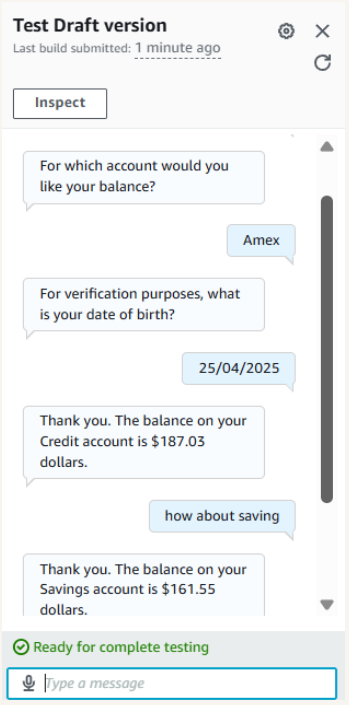
I created an input context, contextCheckBalance, that connects to the output context from CheckBalance. This lets FollowupCheckBalance access saved info, like dateOfBirth, so users don't need to repeat details in follow-up questions.



The final result!

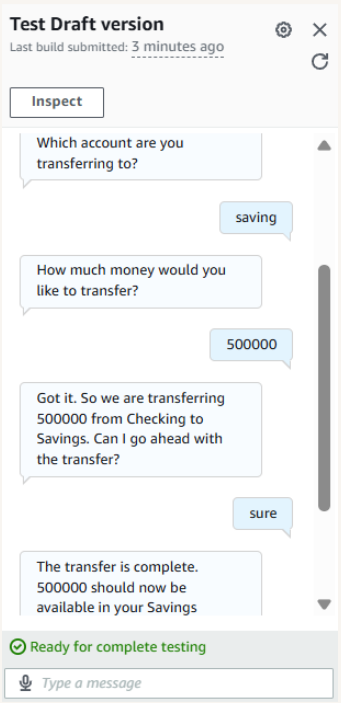
To see the context tags and followup intent in action, I first asked my chatbot, “What’s my account balance?” Then, I followed up with, “What about my savings account?” to trigger the FollowupCheckBalance intent without repeating my date of birth.

If I had gone straight to trying to trigger FollowUpCheckBalance without setting up any context, the chatbot wouldn’t have the needed info, like my date of birth, and would prompt me for verification again or not activate the intent at all.



Build a Chatbot with Multiple Slots

lue



Introducing Today's Project!

What is Amazon Lex?

Amazon Lex is an AWS service to build chatbots using voice or text. It understands natural language, manages conversations, and integrates with AWS services, making chatbot creation easier and scalable for customer support and automation.

How I used Amazon Lex in this project

I used Amazon Lex to create a BankingBot that handles balance checks and follow-ups. By setting intents, slots, and context tags, the bot remembers user info like date of birth, enabling smooth, natural conversations without repeating details.

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

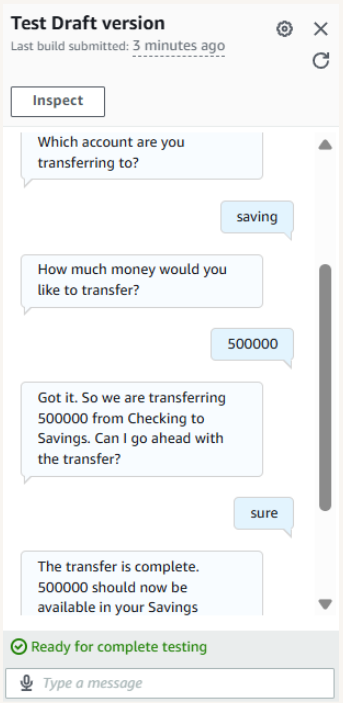
One thing I didn’t expect was how powerful context tags are in Amazon Lex. They made the chatbot remember user details across intents easily, improving the conversation flow and making interactions feel much more natural and seamless than I thought.

This project took me...

20 mins

TransferFunds

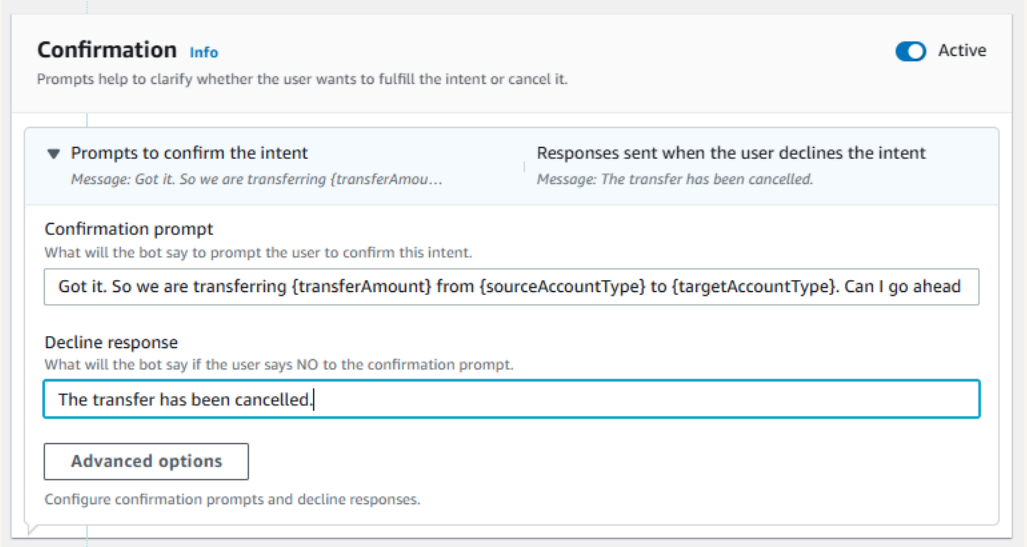
An intent created for my chatbot was TransferFunds, helping users transfer money between their bank accounts, collecting the source and target accounts, the amount to transfer, confirms the details with the user, and completes the transfer if confirm



Using multiple slots

For this intent, I had to use the same slot type twice. This is because both the source and target accounts use the accountType slot type, but they collect different information—one for where the funds come from and one for where they go.

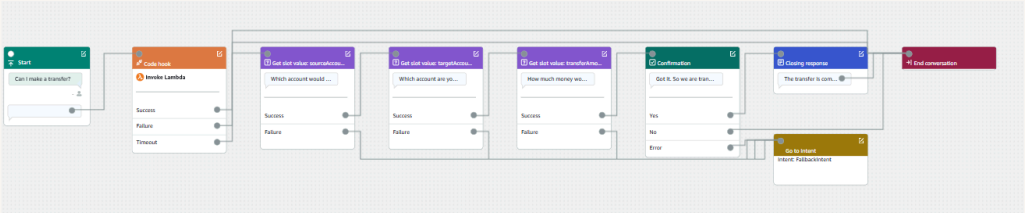
I also learnt how to create confirmation prompts, which are messages that repeat back the user's request and ask for confirmation before proceeding. This helps prevent mistakes and ensures the user really wants to complete the requested action.



Exploring Lex features

Lex also has a special conversation flow feature that lets you control the path a user takes through a conversation by setting conditions, branching, and customizing next steps. This makes complex, multi-turn interactions smoother and more flexible.

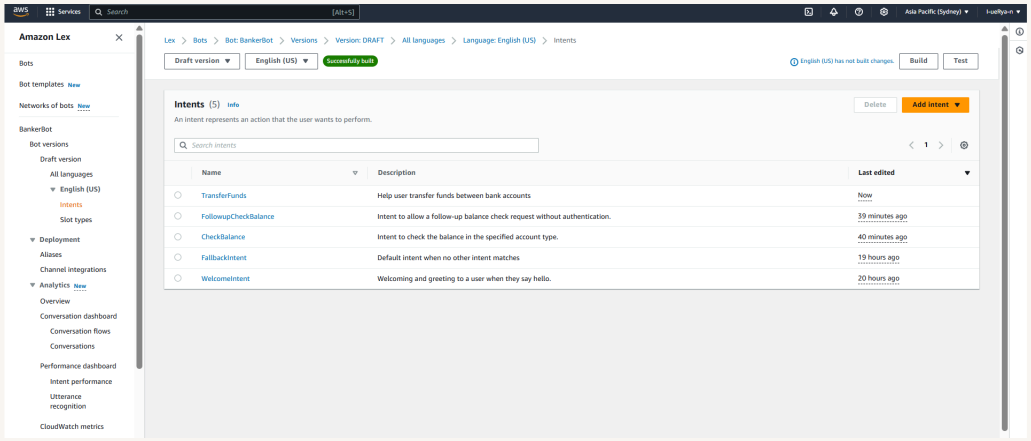
I could also set up your intent using a visual builder! A visual builder lets you design your chatbot's conversation flow using blocks and connections, making it easy to see and manage how each part of the conversation links together without coding.



AWS CloudFormation

AWS CloudFormation is a service that lets you model, provision, and manage AWS resources as code, using templates written in JSON or YAML. It automates resource setup, updates, and deletion, saving time and reducing manual effort.

I used CloudFormation to automate the deployment of BankerBot, including its intents like CheckBalance and FollowupCheckBalance. This let me quickly set up the bot and all related AWS resources in seconds, saving time and avoiding manual setup.



The final result!

2 mins

There was an error after I deployed my bot! The error was denied access to the Lambda function. I fixed this by creating a new Lambda, updating the alias, and adding a resource-based policy to grant my chatbot permission to invoke the function.