# HOW TO BUILD A CHATBOT WITH AMAZON LEX

## INTRODUCTION

Chatbots have become essential in modern applications. They are computer programs designed to simulate conversations with humans. They are used in websites, mobile applications and messaging platforms to answer questions, provide support or automate tasks.

Amazon Lex is an AWS Service that helps developers to build communication interfaces such as chatbot etc. It supports both text and voice input.

This guide will walk you through the process of building a chatbot from scratch, what Chatbot is and how it works, Implementing a chatbot using Amazon Lex, how to design a conversation. By the end of this guide, you will have a working guide that can interact with users effectively.

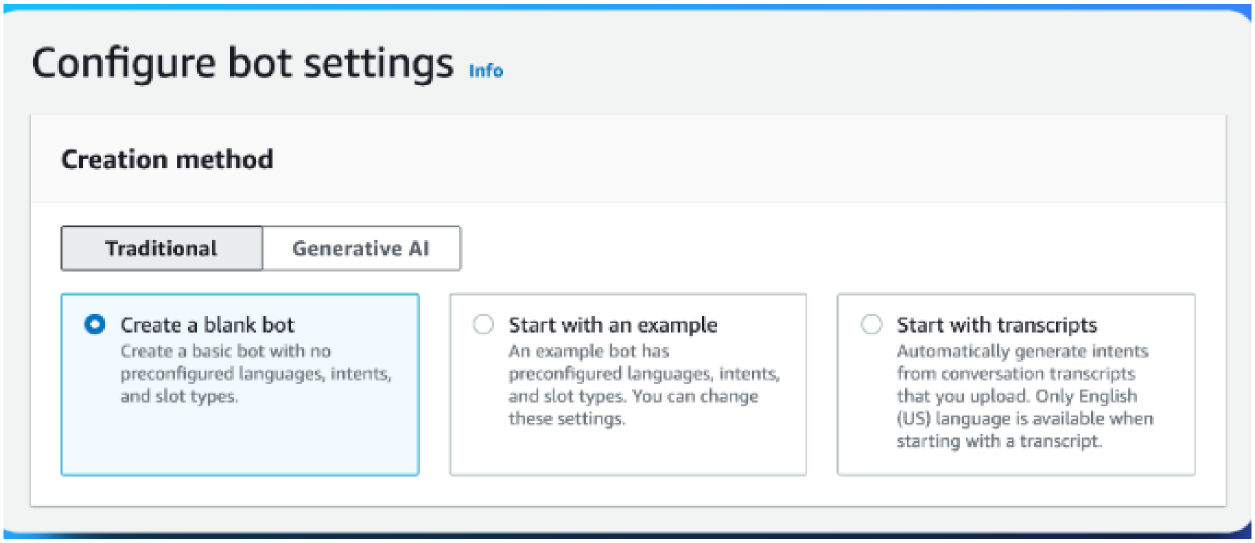
## PREREQUISITES

- AWS Account
- Amazon Lex
- AWS Lambda
-

## STEP 1: SET UP YOUR CHATBOT

Log into AWS account and navigate to Amazon Lex by typing Lex into the search bar of the Console.

- Select **create bot**
- Select **Traditional**
- Select **create blank bot**



Configure bot settings Info

Creation method

Traditional | Generative AI

● Create a blank bot
Create a basic bot with no preconfigured languages, intents, and slot types.

○ Start with an example
An example bot has preconfigured languages, intents, and slot types. You can change these settings.

○ Start with transcripts
Automatically generate intents from conversation transcripts that you upload. Only English (US) language is available when starting with a transcript.

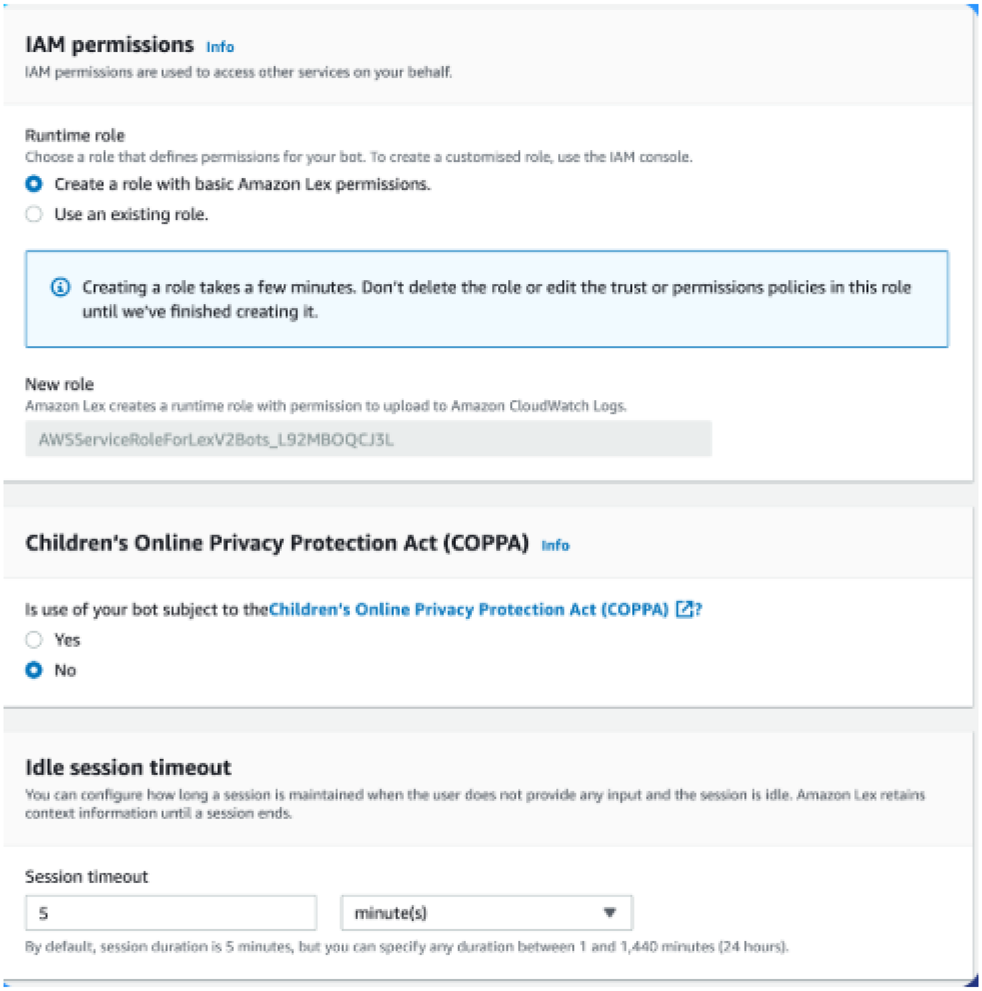
- For bot name, enter “**BankerBot**”
- For description enter “**BankerBot to help customers check their balance and make transfers**”.
- Under IAM permissions, select **create a role with basic Amazon Lex permissions**.

We need Amazon Lex permission to call other AWS services on your behalf and it will be used to call Lambda service

- Under children’s Online Privacy Permission Act, select **No**

- Under Idle Session timeout, keep the default of " **5** mins".

What the Idle timeout session mean is that if the user does not add any input for 5mins, their session will expire.



**IAM permissions** Info

IAM permissions are used to access other services on your behalf.

Runtime role

Choose a role that defines permissions for your bot. To create a customised role, use the IAM console.

● Create a role with basic Amazon Lex permissions.

○ Use an existing role.

ⓘ Creating a role takes a few minutes. Don't delete the role or edit the trust or permissions policies in this role until we've finished creating it.

New role

Amazon Lex creates a runtime role with permission to upload to Amazon CloudWatch Logs.

AWSServiceRoleForLexV2Bots_L92MBOQCJ3L

**Children's Online Privacy Protection Act (COPPA)** Info

Is use of your bot subject to theChildren's Online Privacy Protection Act (COPPA)?

○ Yes

● No

**Idle session timeout**

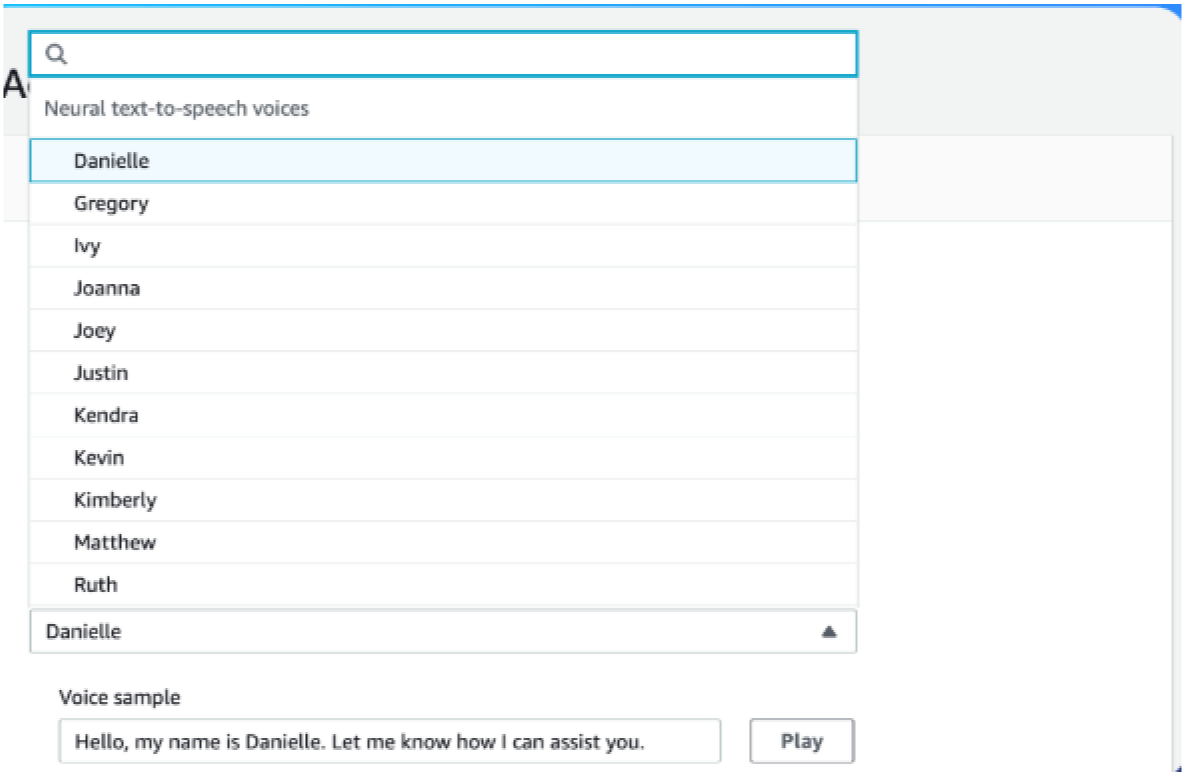
You can configure how long a session is maintained when the user does not provide any input and the session is idle. Amazon Lex retains context information until a session ends.

Session timeout

5 | minute(s) ▼

By default, session duration is 5 minutes, but you can specify any duration between 1 and 1,440 minutes (24 hours).

- Select **next** and keep the Language as **English**.
- Under voice interaction, click on the dropdown to hear different voice actions and select one



Neural text-to-speech voices

- Danielle
- Gregory
- Ivy
- Joanna
- Joey
- Justin
- Kendra
- Kevin
- Kimberly
- Matthew
- Ruth

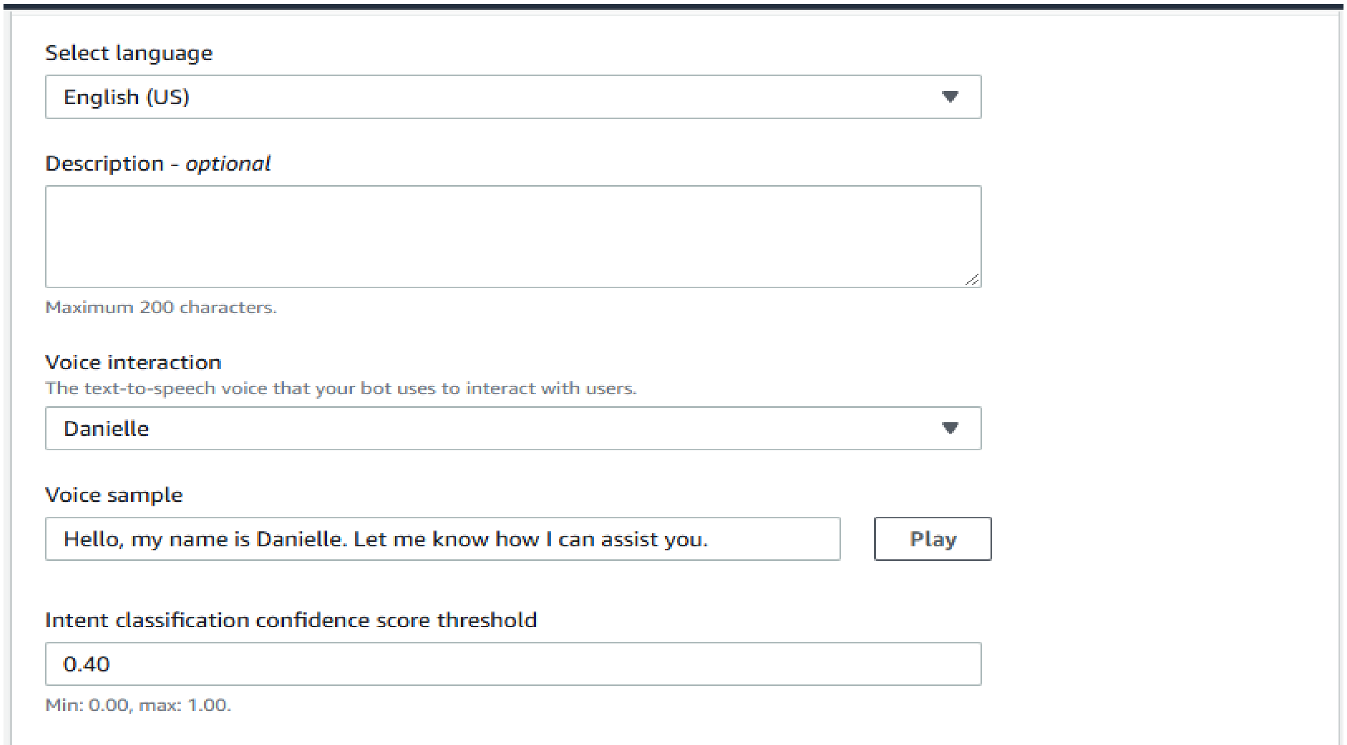
Danielle

Voice sample

Hello, my name is Danielle. Let me know how I can assist you.

- For intent classification confidence score threshold, keep the default value of **0.4**

This threshold is like a minimum score for the chatbot to understand what the user is trying to say, so setting it to 0.4 means the chatbot needs at least 40% confidence that it understands what the user is trying to say but if the chatbot confidence is below that then an error message will be thrown.



Select language

English (US)

Description - *optional*

Maximum 200 characters.

Voice interaction

The text-to-speech voice that your bot uses to interact with users.

Danielle

Voice sample

Hello, my name is Danielle. Let me know how I can assist you.

Intent classification confidence score threshold

0.40

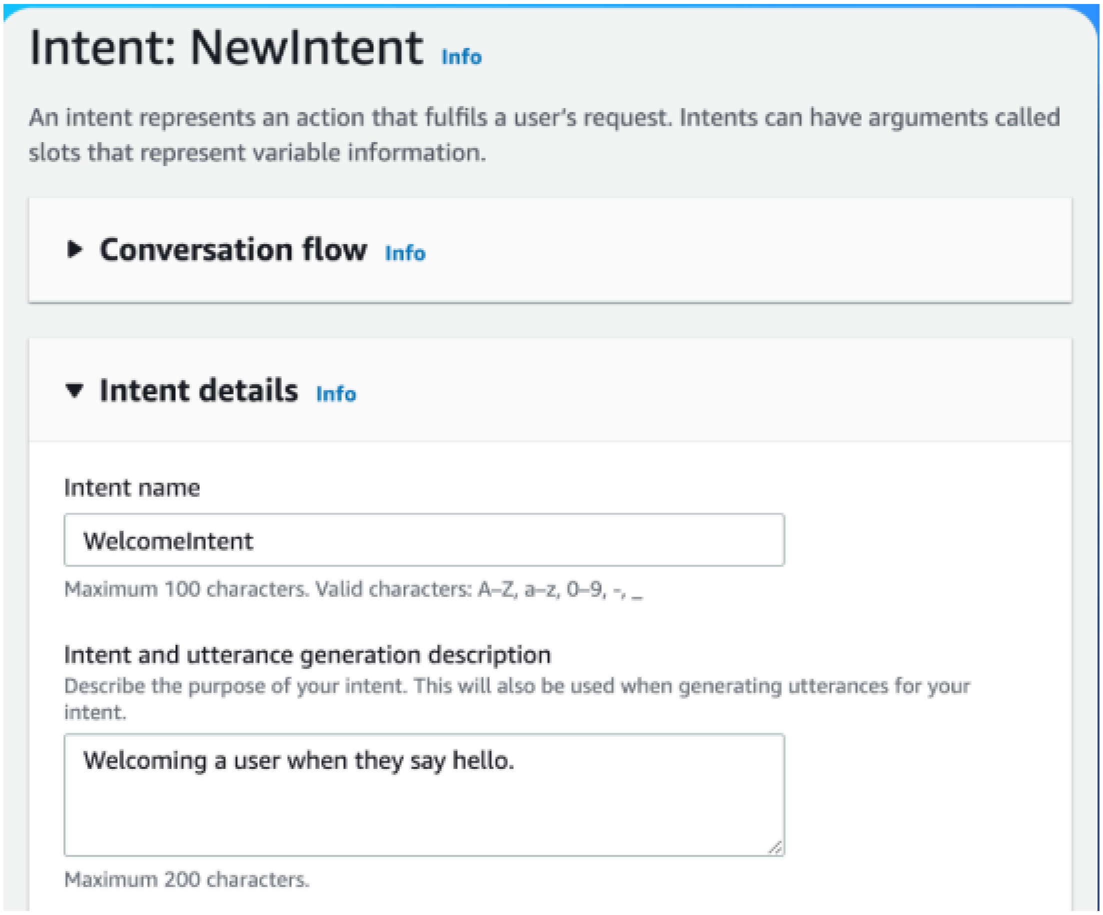
Min: 0.00, max: 1.00.

- Select **Done**

## STEP 2: CREATE YOUR FIRST INTENT

When the Bot is created, you will automatically see a new page called Intent: NewIntent
An intent is what the user is trying to achieve in their conversation with the chatbot. So you build your chatbot by defining different intents.

- Under the intent details, enter the intent name, "**WelcomeIntent**".
- Add a description, "**Welcoming a user when they say Hello**".



Intent: NewIntent Info

An intent represents an action that fulfils a user's request. Intents can have arguments called slots that represent variable information.

▸ Conversation flow Info

▾ Intent details Info

Intent name

WelcomeIntent

Maximum 100 characters. Valid characters: A–Z, a–z, 0–9, -, _

Intent and utterance generation description

Describe the purpose of your intent. This will also be used when generating utterances for your intent.

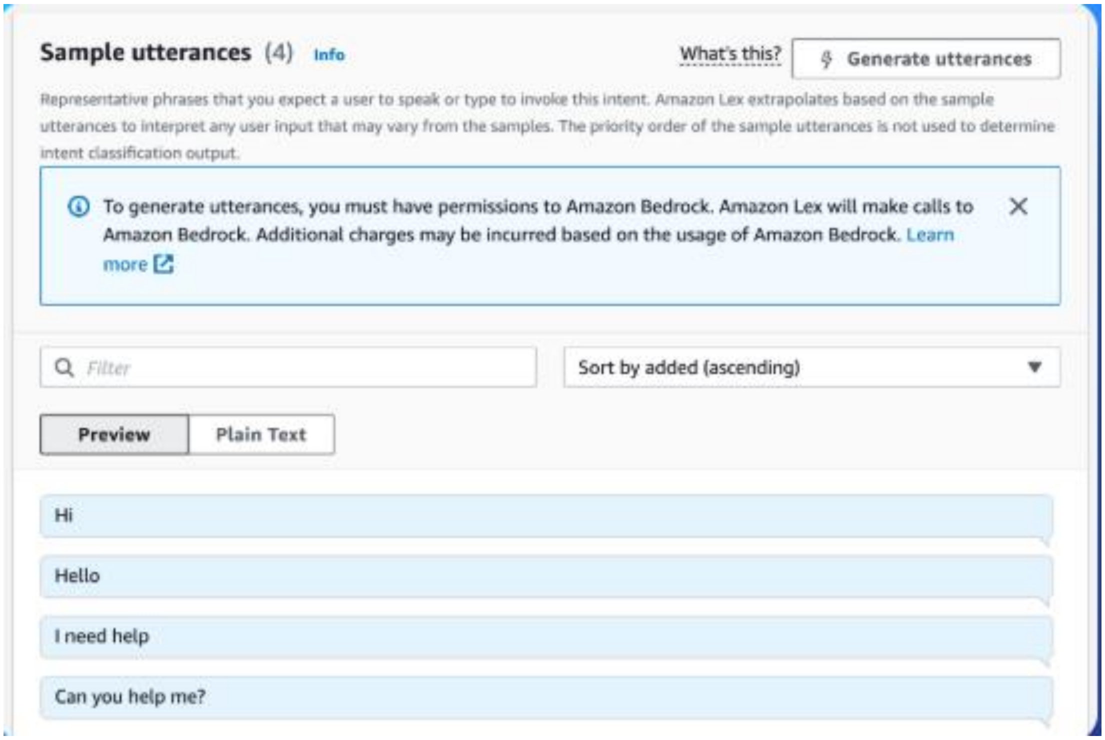
Welcoming a user when they say hello.

Maximum 200 characters.

- Scroll down to the sample utterances panel and click the **plain** text button
- Write a user input that will trigger the intent as seen below

**Hi**
**Hello**
**I need help**
**Can you help me?**

- Click back the **preview** button to see the utterances in chat form.



Representative phrases that you expect a user to speak or type to invoke this intent. Amazon Lex extrapolates based on the sample utterances to interpret any user input that may vary from the samples. The priority order of the sample utterances is not used to determine intent classification output.

To generate utterances, you must have permissions to Amazon Bedrock. Amazon Lex will make calls to Amazon Bedrock. Additional charges may be incurred based on the usage of Amazon Bedrock. Learn more

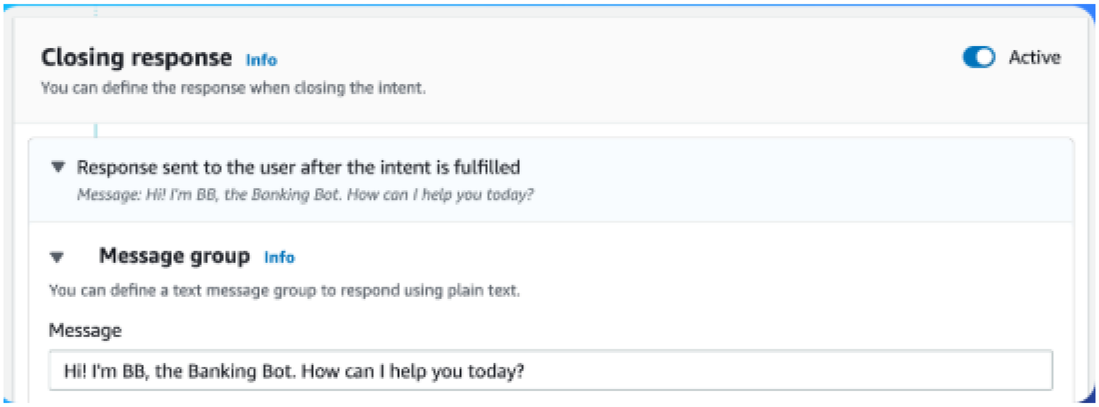
Hi

Hello

I need help

Can you help me?

- Scroll down to closing response and expand the speech bubble for “ **Response sent to the user after the intent is fulfilled”.**
- In the message field enter a message “**Hi! I am BB the BankingBot, How can I help you today?”**



You can define the response when closing the intent.

Response sent to the user after the intent is fulfilled

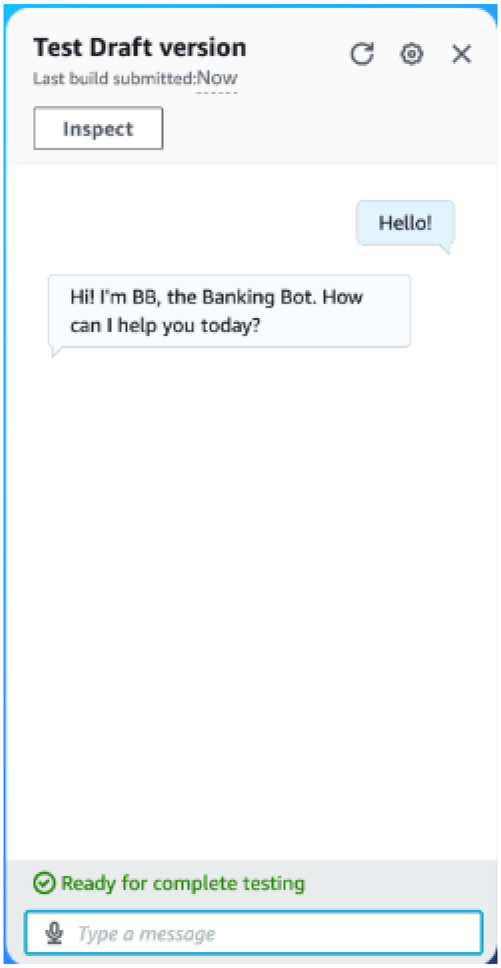
*Message: Hi! I'm BB, the Banking Bot. How can I help you today?*

You can define a text message group to respond using plain text.

Message

Hi! I'm BB, the Banking Bot. How can I help you today?

- Choose **save** intent and choose **build**.
- Choose **test** and a dialog box will pop up and you can interact with the bot with the user input that you had pasted on the sample utterance panel.

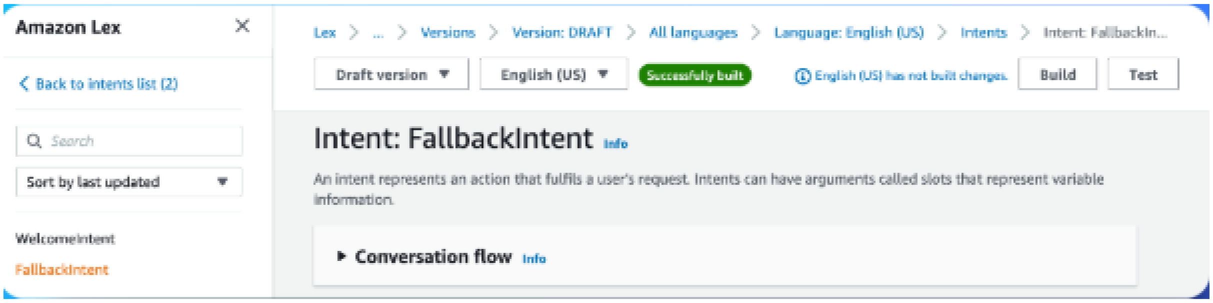


Hello!

Hi! I'm BB, the Banking Bot. How can I help you today?

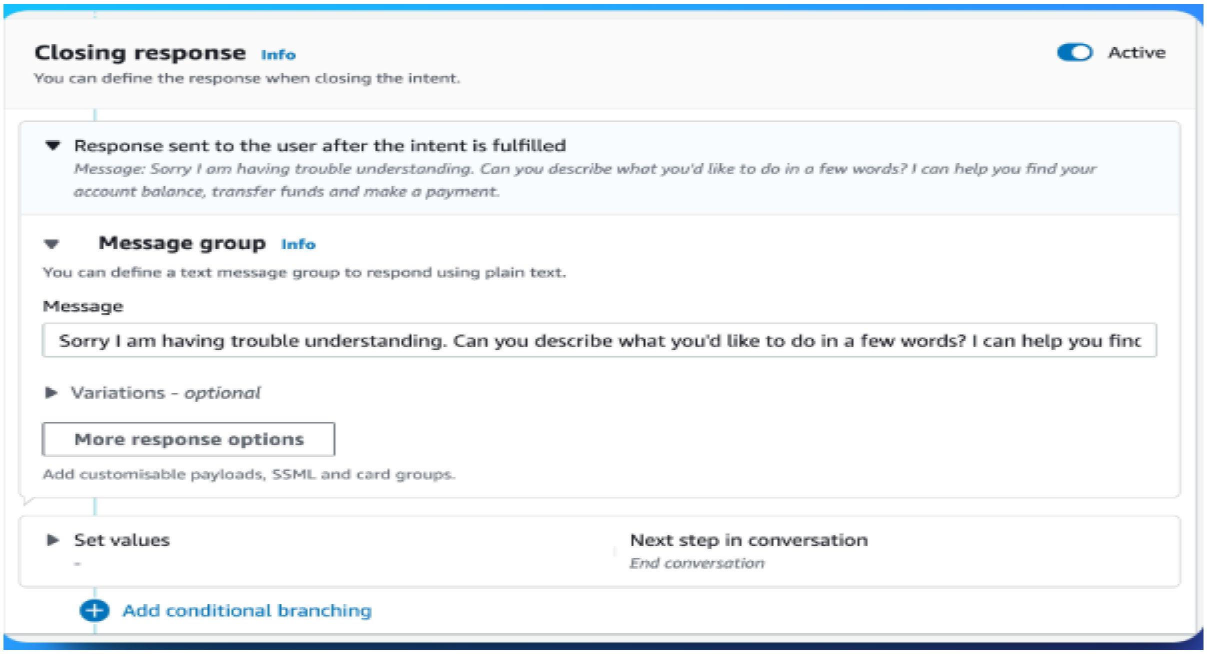
## STEP 3: MANAGE FALLBACKINTENT

- In your left hand navigation panel, choose “**FallBackIntent**”.



An intent represents an action that fulfils a user's request. Intents can have arguments called slots that represent variable information.

- Scroll down to **closing responses**.
- Expand the speech bubble for “**Response sent to the user after the intent is fulfilled**”.
- In the message field, add the text “ **Sorry I am having trouble understanding. Can you describe what you would like to do in a few words? I can help you find your Account Balance, transfer funds and make a payment.”**



Closing response Info

You can define the response when closing the intent.

Active

▼ Response sent to the user after the intent is fulfilled

*Message: 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.*

▼ Message group Info

You can define a text message group to respond using plain text.

Message

Sorry I am having trouble understanding. Can you describe what you'd like to do in a few words? I can help you finc

▶ Variations - *optional*

Add customisable payloads, SSML and card groups.

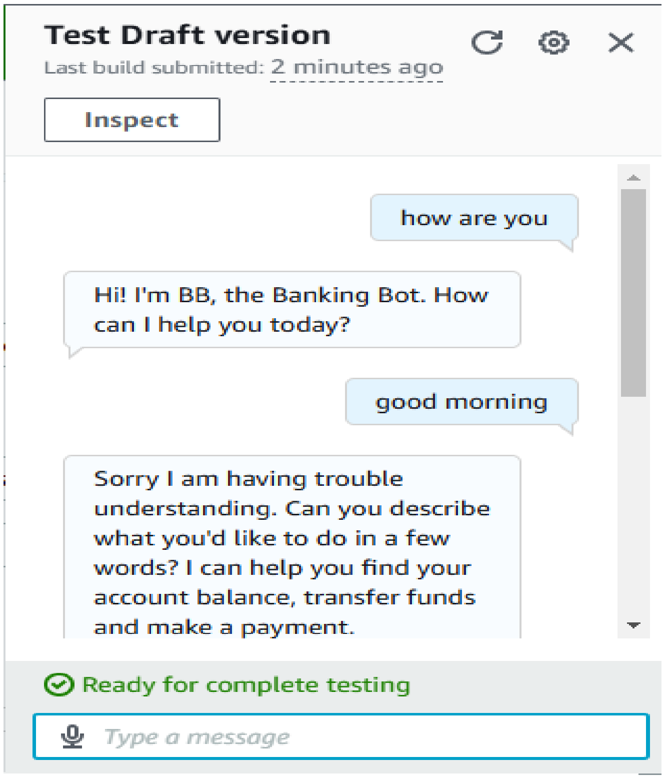
▶ Set values

-

Next step in conversation

*End conversation*

- Test these:
  - **Help me**
  - **Hiya**
  - **How are you**
  - **Good morning**



Test Draft version

Last build submitted: 2 minutes ago

how are you

Hi! I'm BB, the Banking Bot. How can I help you today?

good morning

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.

Ready for complete testing

- Move over to another toggle next to the label **variations-optional**
- Click open the toggle and enter the text "**hmm could you try rephrasing that? I can help you find your Account Balance, transfer funds and make a payment.**"
- You can add another variation.

Variations will give the user dynamic responses making them sound more conversational.

- Choose **Save** intent, choose **Build** and choose **Test**

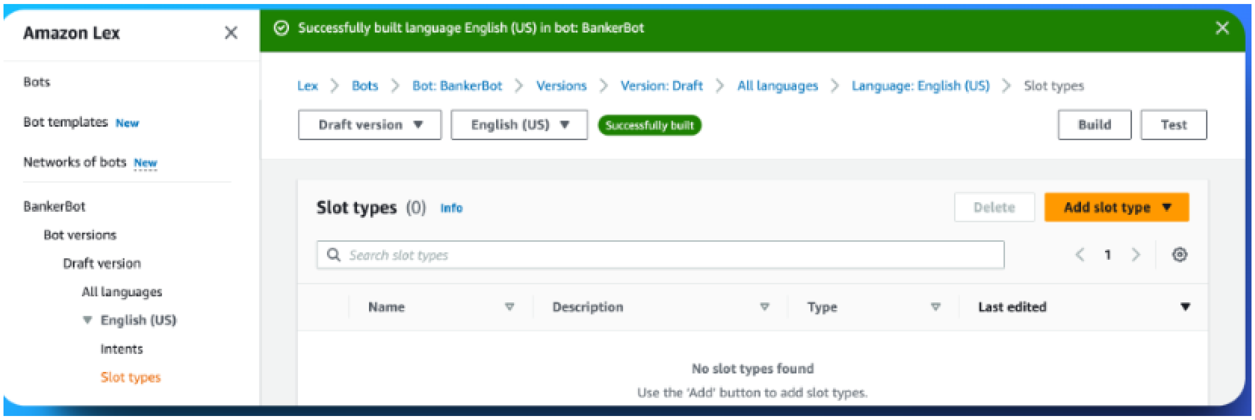
- Test these:
  - **Help me**
  - **Hiya**
  - **How are you**
  - **Good morning**

## STEP 4: CREATE A CUSTOM SLOT FOR ACCOUNT TYPES

In the Amazon Lex Console, choose slot types on the left hand navigation panel

Slots are information needed by a chatbot to complete a user's request.

- Choose **Add slot type**.



Successfully built language English (US) in bot: BankerBot

Lex > Bots > Bot: BankerBot > Versions > Version: Draft > All languages > Language: English (US) > Slot types

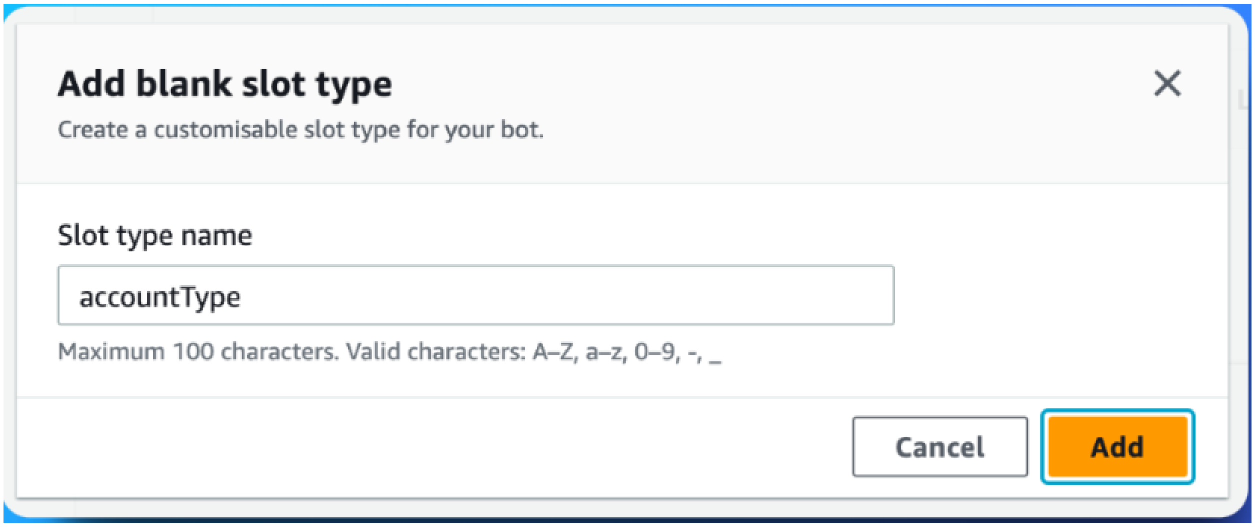
Slot types (0)

| Name | Description | Type | Last edited |
|---|---|---|---|

No slot types found

Use the 'Add' button to add slot types.

- From the dropdown , choose **Add blank slot type.**
- Enter "**accountType**" for the slot type name and choose **Add**.



Add blank slot type

Create a customisable slot type for your bot.

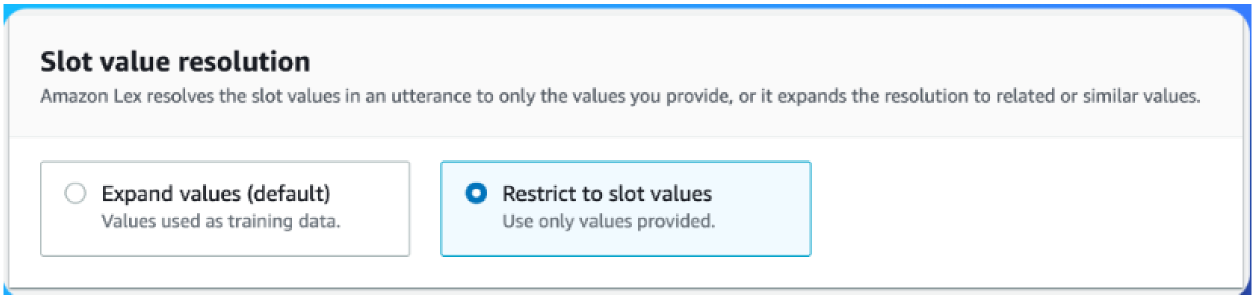
Slot type name

accountType

Maximum 100 characters. Valid characters: A–Z, a–z, 0–9, -, _

- This will bring up a large slot types editor panel.
- In the slot value resolution panel,choose “**Restrict to slot values**”

Selecting Restrict to slot values makes sure that only the values that you specify will count as a valid accountType.



**Slot value resolution**

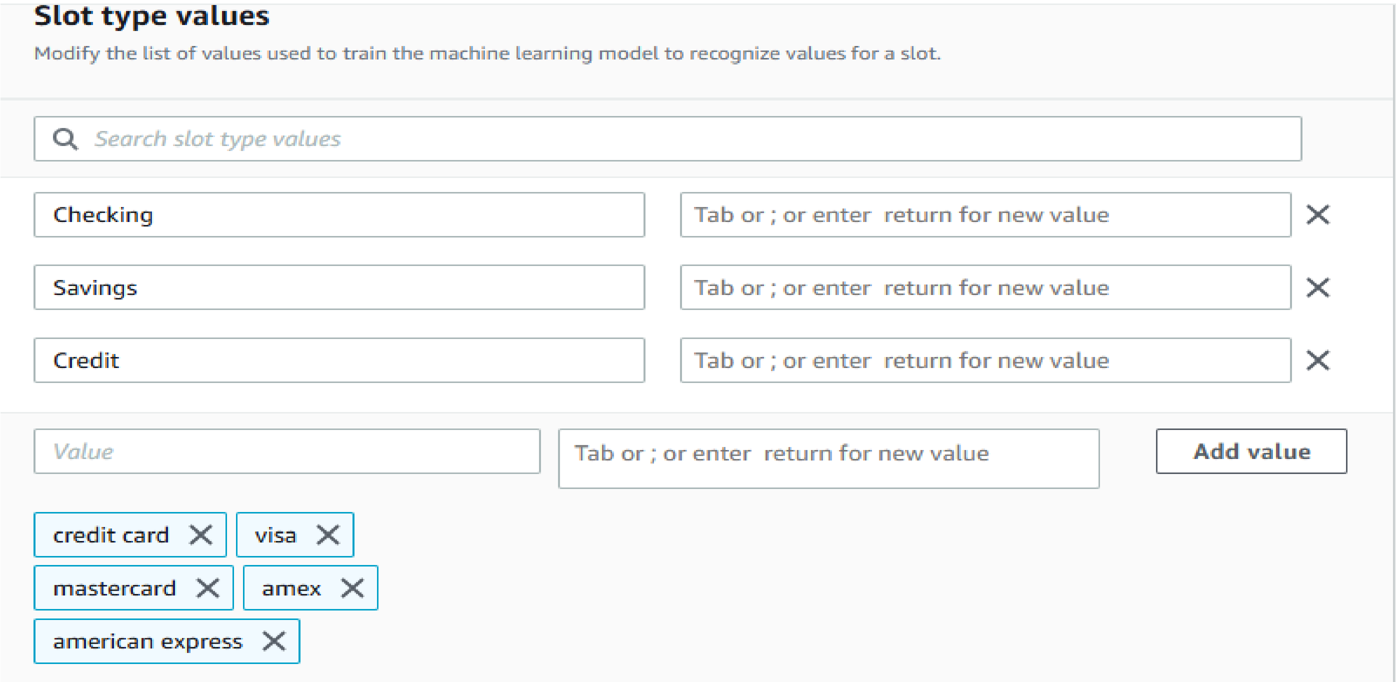
Amazon Lex resolves the slot values in an utterance to only the values you provide, or it expands the resolution to related or similar values.

○ Expand values (default)
Values used as training data.

● Restrict to slot values
Use only values provided.

- Add other three accountTypes, **Savings,Credit and Checking** accounts.
- In the value fields, enter “**Checking”** and click enter.
- Do the  same for “**Savings”**.
- In the second field, enter **Credit** and press “**;”** after everytime you add a new one. Credit card, visa, mastercard, amex, american express.
- Choose **Add value**.

By entering specific slot type values, we make sure that our Bankerbot only recognizes and accept the bank account types (Checking, Credit and Savings). This prevents any confusion with the user.



**Slot type values**

Modify the list of values used to train the machine learning model to recognize values for a slot.

Search slot type values

| | |
|---|---|
| Checking | Tab or ; or enter  return for new value |
| Savings | Tab or ; or enter  return for new value |
| Credit | Tab or ; or enter  return for new value |
| Value | Tab or ; or enter  return for new value |

Add value

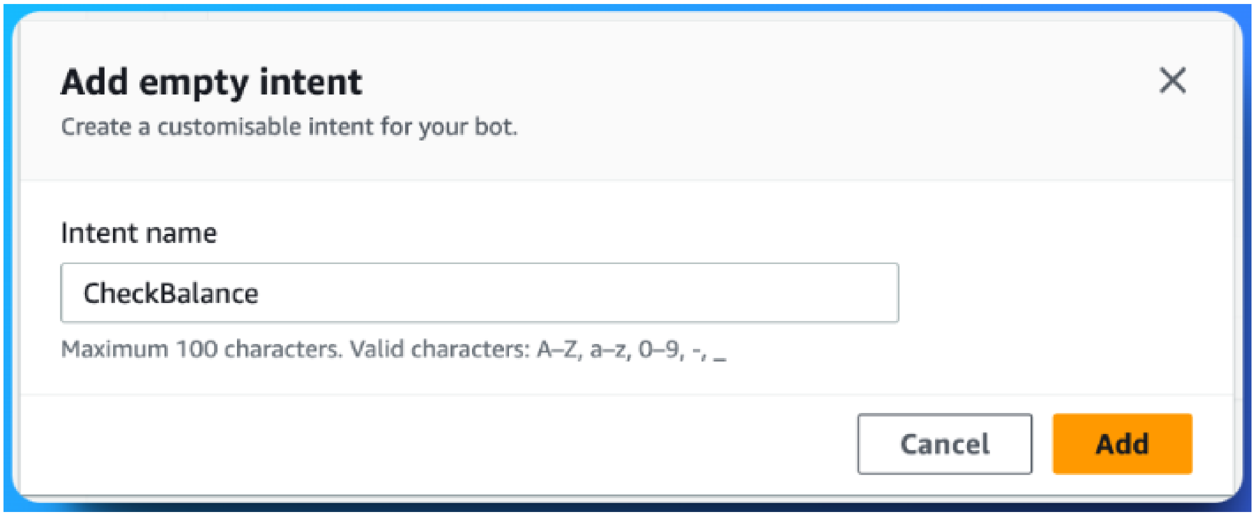
credit card ✕ visa ✕
mastercard ✕ amex ✕
american express ✕

- Choose **Save slot type**.

## STEP 5: CREATE THE CHECKBALANCE INTENT

After the custom slots are all set up, the BankerBot now knows about the different account types and it is ready to help users check their account balance.

- Head back to Intents in your left hand navigation panel.



Add empty intent

Create a customisable intent for your bot.

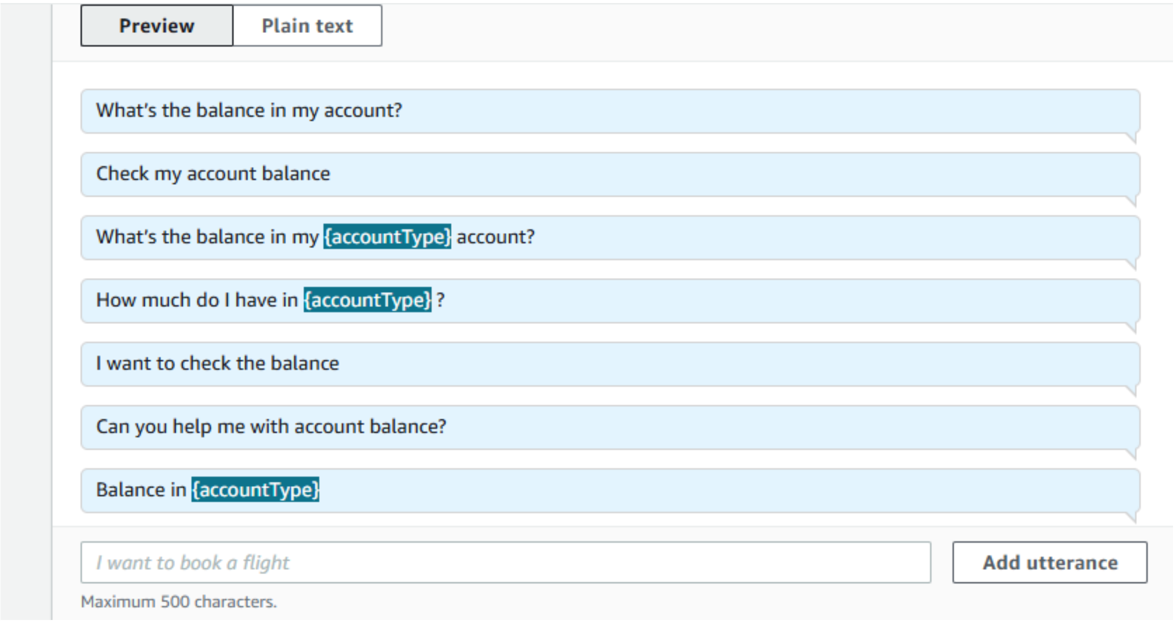
Intent name

CheckBalance

Maximum 100 characters. Valid characters: A–Z, a–z, 0–9, -, _

- Choose **Add intent** then **Add empty intent**.
- Enter “**CheckBalance**” as your intent name and choose **Add**.
- Enter “**intents to check the balance in the specified account type**” on the intents details panel.
- Scroll down to **Sample utterances** and switch to **Plain text**
- Add the utterances below and switch to **Preview**.

**What’s the balance in my account?**  
**Check my account balance**  
**What’s the balance in my {accountType} account?**  
**How much do I have in {accountType} ?**  
**I want to check the balance**  
**Can you help me with account balance?**  
**Balance in {accountType}**



What's the balance in my account?

Check my account balance

What's the balance in my {accountType} account?

How much do I have in {accountType} ?

I want to check the balance

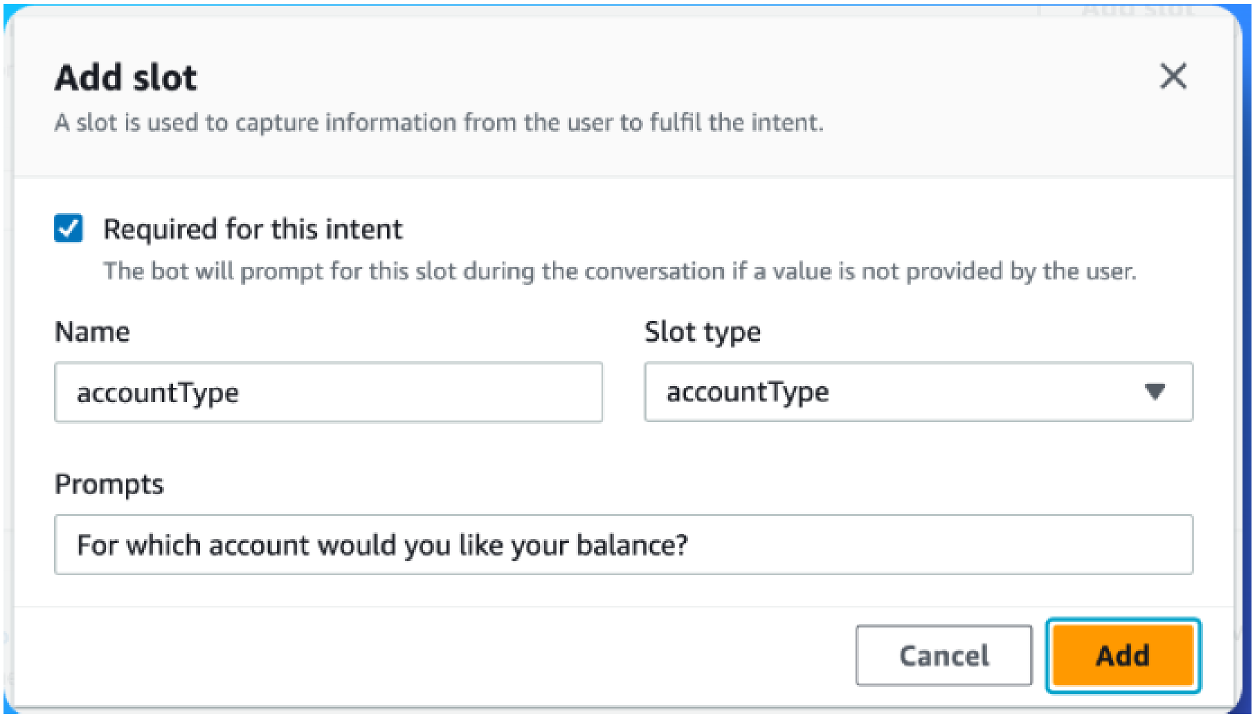
Can you help me with account balance?

Balance in {accountType}

*I want to book a flight*

Maximum 500 characters.

- Scroll down until you see the Slots Pane and choose **Add slot** button.
- For slot's Name enter "**accountType"** and for the slot type, choose your custom slot value **accountType** which you had created.
- Enter the prompt "**For which account would you like your balance?**" and choose **Add**.



Add slot

A slot is used to capture information from the user to fulfil the intent.

☑ Required for this intent

The bot will prompt for this slot during the conversation if a value is not provided by the user.

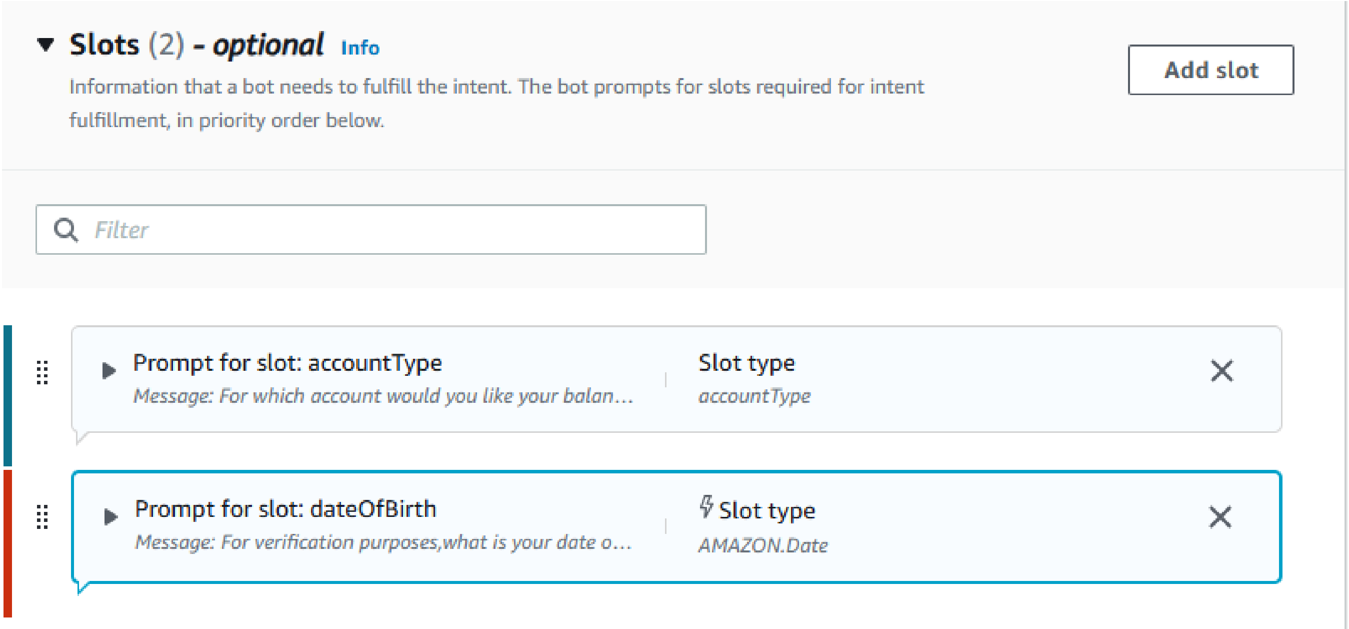
Name: accountType

Slot type: accountType

Prompts: For which account would you like your balance?

- Next, choose **Add slot** and use the values below for the next slot.
  **Name: DateOfBirth**

**Slot type: Amazon.Date**
**Prompts: For verification purposes, what is your date of birth?**



Slots (2) - *optional*

Information that a bot needs to fulfill the intent. The bot prompts for slots required for intent fulfillment, in priority order below.

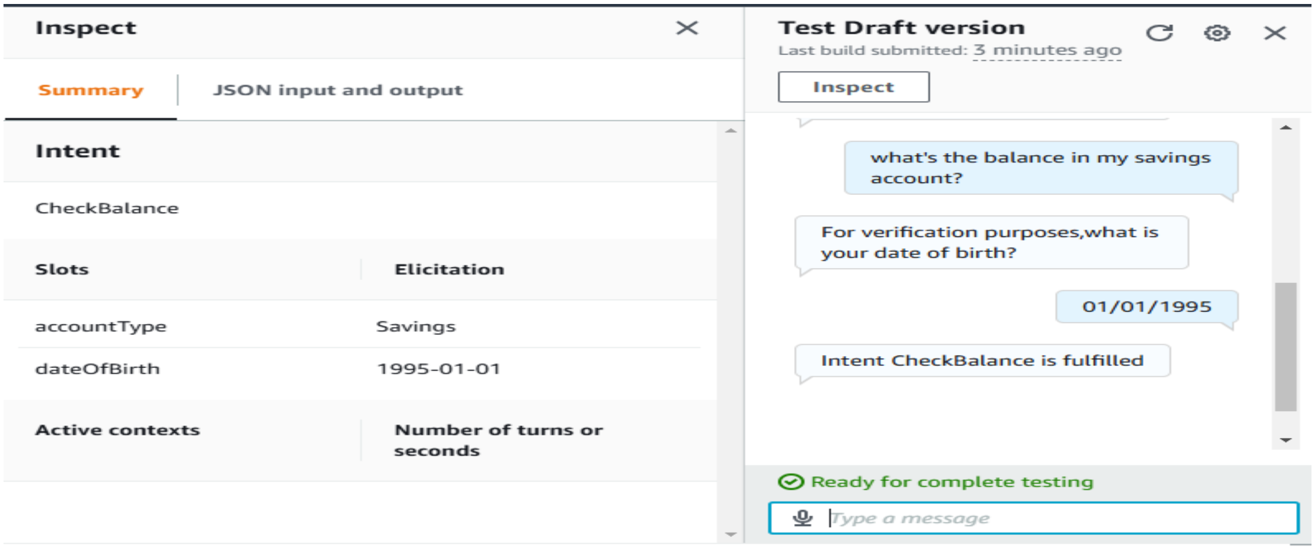
Prompt for slot: accountType
*Message: For which account would you like your balan…*
Slot type
*accountType*

Prompt for slot: dateOfBirth
*Message: For verification purposes,what is your date o…*
Slot type
*AMAZON.Date*

- **Save** intent, choose **Build** and choose **Test**.
- In your chat window, enter "**what's the balance in my savings account?**" but this time Lex will only prompt you for your date your birth as it already know that it should be the Savings account it checks.
- Follow Amazon Lex Prompts and enter an account type and birth date.

If the Bot returns "**intent CheckBalance fulfilled**", it means the bot has successfully taken in the user's details but doesn't actually know how to calculate the bank yet.

- Choose inspect



Inspect

Intent

CheckBalance

| Slots | Elicitation |
|---|---|
| accountType | Savings |
| dateOfBirth | 1995-01-01 |

| Active contexts | Number of turns or seconds |
|---|---|

Test Draft version

Last build submitted: 3 minutes ago

what's the balance in my savings account?

For verification purposes,what is your date of birth?

01/01/1995

Intent CheckBalance is fulfilled

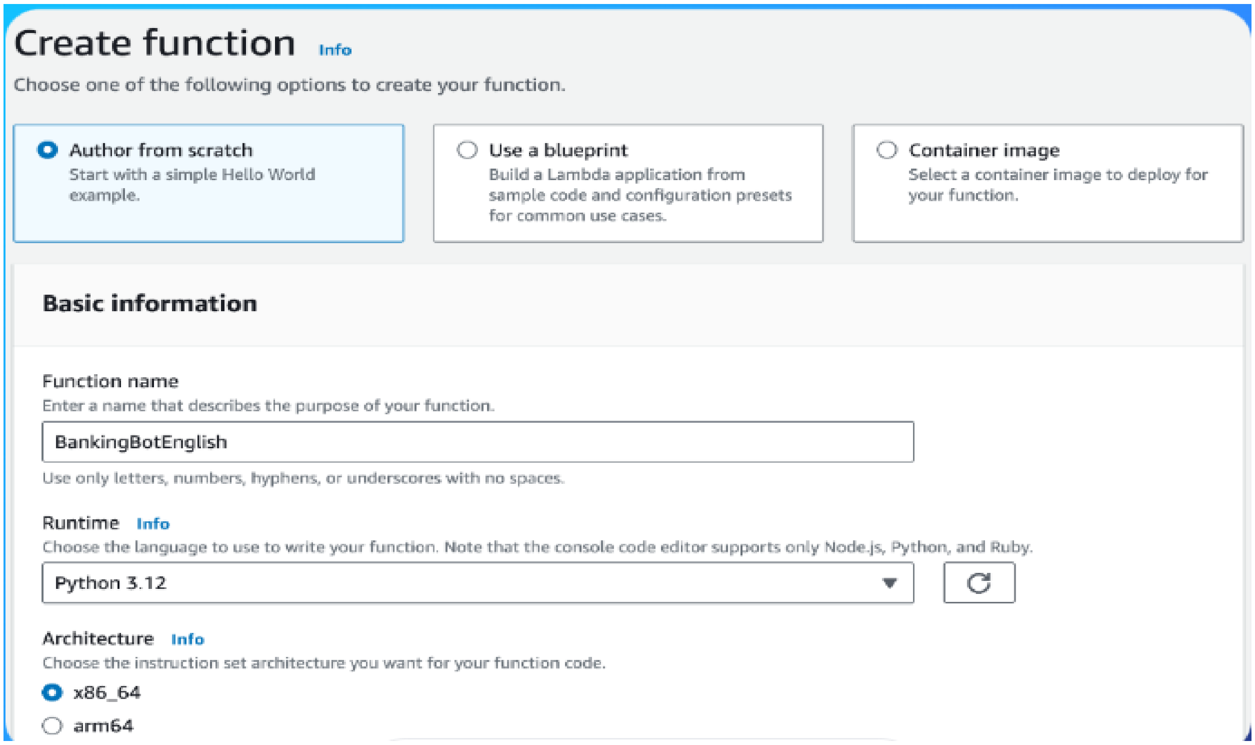
Ready for complete testing

### STEP 6: CREATE YOUR AWS LAMBDA FUNCTION

Lambda is a service that helps you run code in the cloud without server management. Lambda will manage the servers for you.

Lambda will be used here to generate a random bank balance numbers.

- Head to **Lambda** in the AWS Console and choose “**Author from scratch**”.
- Enter the function name “**BankingBotEnglish**”.
- Choose runtime **Python 3.12** or a later version of python 3.



Create function

Choose one of the following options to create your function.

● Author from scratch
Start with a simple Hello World example.

○ Use a blueprint
Build a Lambda application from sample code and configuration presets for common use cases.

○ Container image
Select a container image to deploy for your function.

Basic information

Function name
Enter a name that describes the purpose of your function.
BankingBotEnglish
Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Python 3.12

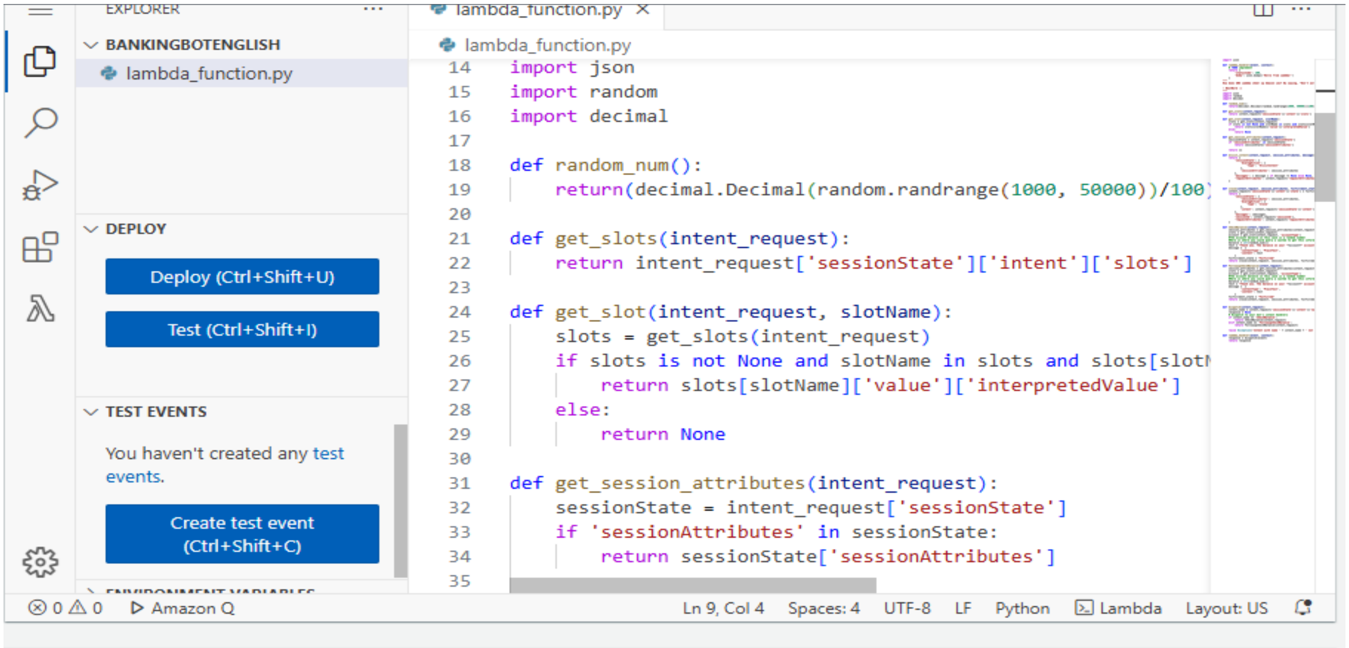
Architecture
Choose the instruction set architecture you want for your function code.
● x86_64
○ arm64

- Select **Create function** and scroll down to the **Function code** Section.

- Double-click on “**lambda_function.py**” on the left hand file browser and add the source code file below. The python script will help the chatbot give users quick answers about their account balances.

```
"""
How does AWS Lambda cheer up Amazon Lex? By saying, "Don't worry, I've got
your back(end)!"

- NextWork :)
"""
import json
import random
import decimal

def random_num():
    return(decimal.Decimal(random.randrange(1000, 50000))/100)

def get_slots(intent_request):
    return intent_request['sessionState']['intent']['slots']

def get_slot(intent_request, slotName):
    slots = get_slots(intent_request)
    if slots is not None and slotName in slots and slots[slotName] is not
None:
        return slots[slotName]['value']['interpretedValue']
    else:
        return None

def get_session_attributes(intent_request):
    sessionState = intent_request['sessionState']
    if 'sessionAttributes' in sessionState:
        return sessionState['sessionAttributes']

    return {}

def elicit_intent(intent_request, session_attributes, message):
    return {
        'sessionState': {
            'dialogAction': {
                'type': 'ElicitIntent'
            },
            'sessionAttributes': session_attributes
        },
        'messages': [ message ] if message != None else None,
        'requestAttributes': intent_request['requestAttributes'] if
'requestAttributes' in intent_request else None
    }


def close(intent_request, session_attributes, fulfillment_state, message):
    intent_request['sessionState']['intent']['state'] = fulfillment_state
    return {
        'sessionState': {
            'sessionAttributes': session_attributes,
            'dialogAction': {
                'type': 'Close'
            },
            'intent': intent_request['sessionState']['intent']
        },
        'messages': [message],
        'sessionId': intent_request['sessionId'],
        'requestAttributes': intent_request['requestAttributes'] if 
'requestAttributes' in intent_request else None
    }

def CheckBalance(intent_request):
    session_attributes = get_session_attributes(intent_request)
    slots = get_slots(intent_request)
    account = get_slot(intent_request, 'accountType')
    #The account balance in this case is a random number
    #Here is where you could query a system to get this information
    balance = str(random_num())
    text = "Thank you. The balance on your "+account+" account is 
$"+balance+" dollars."
    message =  {
            'contentType': 'PlainText',
            'content': text
        }
    fulfillment_state = "Fulfilled"    
    return close(intent_request, session_attributes, fulfillment_state, 
message)

def FollowupCheckBalance(intent_request):
    session_attributes = get_session_attributes(intent_request)
    slots = get_slots(intent_request)
    account = get_slot(intent_request, 'accountType')
    #The account balance in this case is a random number
    #Here is where you could query a system to get this information
    balance = str(random_num())
    text = "Thank you. The balance on your "+account+" account is 
$"+balance+" dollars."
    message =  {
            'contentType': 'PlainText',
            'content': text
        }
    fulfillment_state = "Fulfilled"    
    return close(intent_request, session_attributes, fulfillment_state, 
message)


def dispatch(intent_request):
    intent_name = intent_request['sessionState']['intent']['name']
    response = None
    # Dispatch to your bot's intent handlers
    if intent_name == 'CheckBalance':
        return CheckBalance(intent_request)
    elif intent_name == 'FollowupCheckBalance':
        return FollowupCheckBalance(intent_request)

    raise Exception('Intent with name ' + intent_name + ' not supported')

def lambda_handler(event, context):
    response = dispatch(event)
    return response
```

- Paste the code on your text editor and choose **Deploy.**



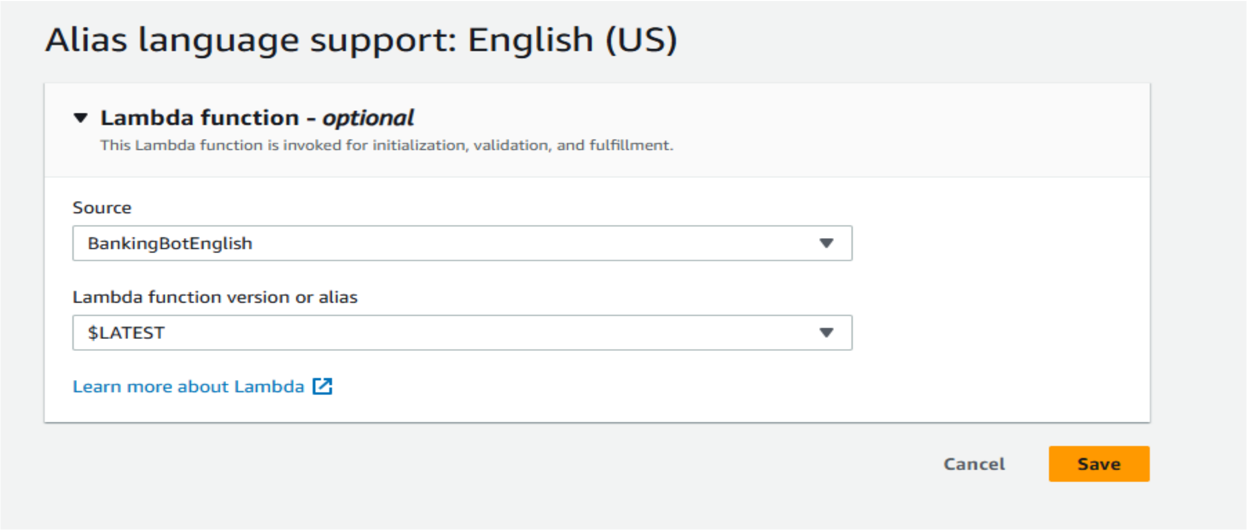
You haven't created any test events.

```
import json
import random
import decimal

def random_num():
    return(decimal.Decimal(random.randrange(1000, 50000))/100)

def get_slots(intent_request):
    return intent_request['sessionState']['intent']['slots']

def get_slot(intent_request, slotName):
    slots = get_slots(intent_request)
    if slots is not None and slotName in slots and slots[slotN
        return slots[slotName]['value']['interpretedValue']
    else:
        return None

def get_session_attributes(intent_request):
    sessionState = intent_request['sessionState']
    if 'sessionAttributes' in sessionState:
        return sessionState['sessionAttributes']

```

## STEP 7: CONNECT AWS LAMBDA WITH AMAZON LEX

We need Amazon Lex and Lambda to work together so that when a user asks for a bank account balance, Lex can trigger Lambda to calculate the number right away. So we will connect the Lambda function to BankerBot.

- Head back to Amazon Lex Console and select **BankerBot** and on the left hand menu, choose **Aliases**.
  You can think of Alias as a pointer for a specific version of your bot.
- Choose the default, **TestBotAlias** (this is the default version of your bot that is made for testing and development).
- From the Languages panel, select **English (US)** and a Lambda function panel will pop up. This panel lets you associate a Lambda function to this TestBotAlias version of the bot.
- Choose your Lambda function **BankingBotEnglish** for source and leave the Alias field at the default, **$LATEST**. Using this version means you are directing your Alias to always use the most up to date version of Lambda function.



Alias language support: English (US)

▼ **Lambda function - *optional***

This Lambda function is invoked for initialization, validation, and fulfillment.

Source: BankingBotEnglish

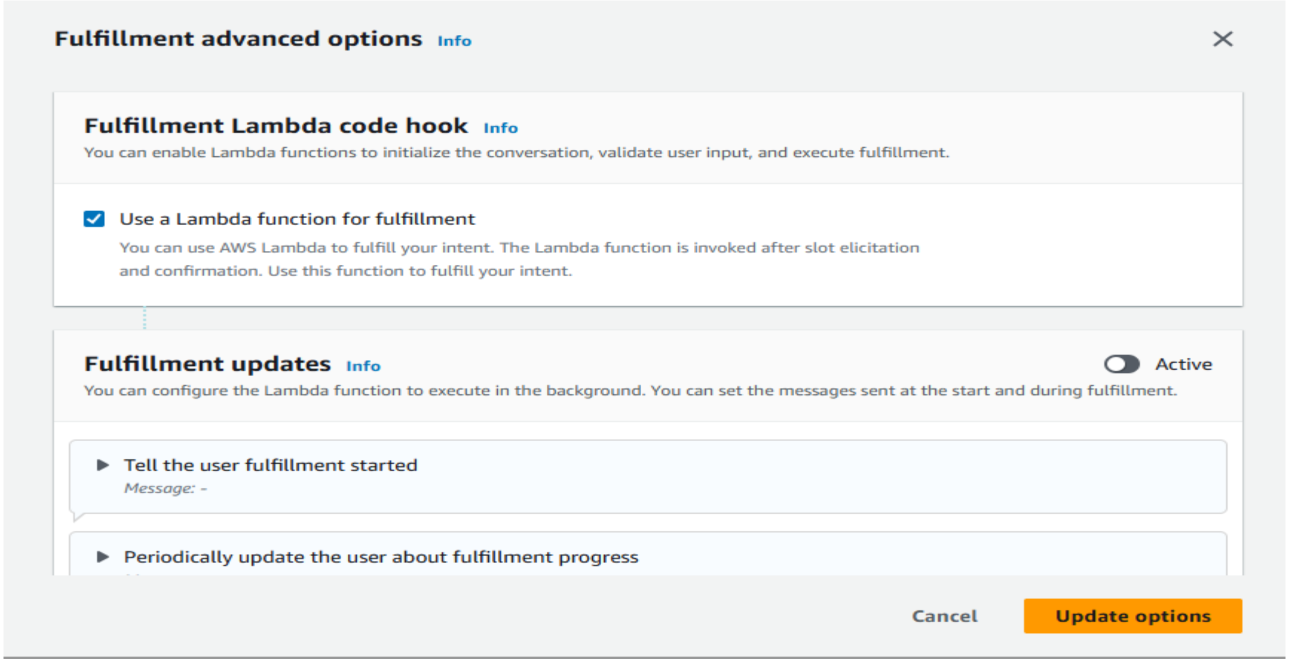
Lambda function version or alias: $LATEST

- Choose **Save**

## STEP 8: CONNECT YOUR CHECKBALANCE INTENT WITH YOUR LAMBDA FUNCTION

Lambda function has to be connected to the CheckBalance intent.

- Navigate to the **CheckBalance** Intent and scroll down to the **Fulfilment panel**.
- Expand the "**On successful fulfilment**" bubble and choose "**Advanced options**".
- Under the **Fulfilment Lambda Code Hook** panel, check the checkbox next to use a Lambda function for Fulfilment.
- Choose **Update Options**



**Fulfillment advanced options** Info

**Fulfillment Lambda code hook** Info

You can enable Lambda functions to initialize the conversation, validate user input, and execute fulfillment.

☑ Use a Lambda function for fulfillment

You can use AWS Lambda to fulfill your intent. The Lambda function is invoked after slot elicitation and confirmation. Use this function to fulfill your intent.

**Fulfillment updates** Info

You can configure the Lambda function to execute in the background. You can set the messages sent at the start and during fulfillment.

▶ Tell the user fulfillment started

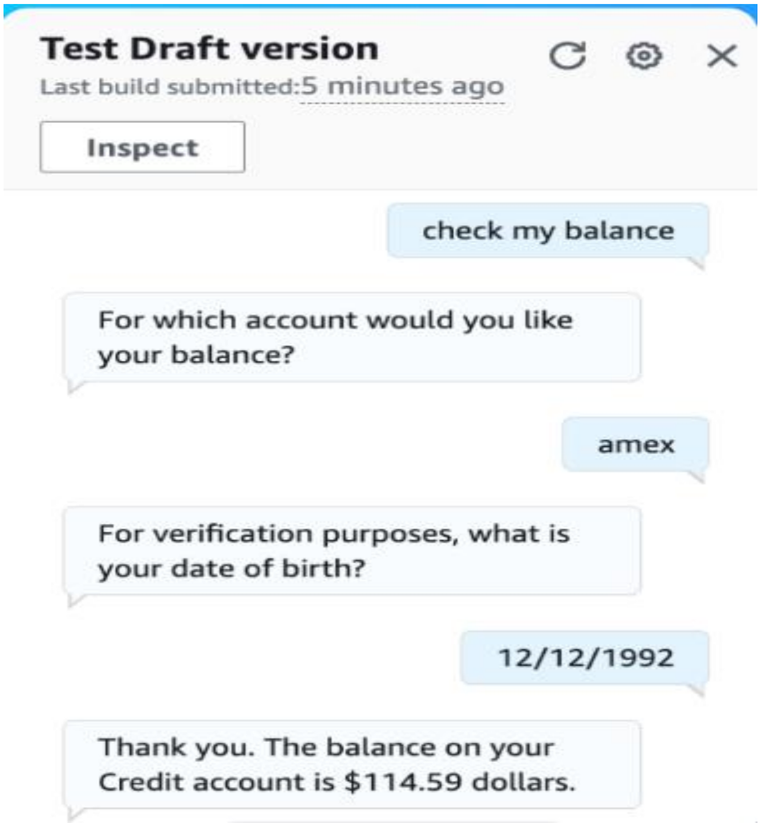
*Message: -*

▶ Periodically update the user about fulfillment progress

- Choose **Save** intent
- Choose **Build**

- Choose **Test**

Ask for the balance of any of your accounts. The bot should be able to return random bank balance figures.



check my balance

For which account would you like your balance?

amex

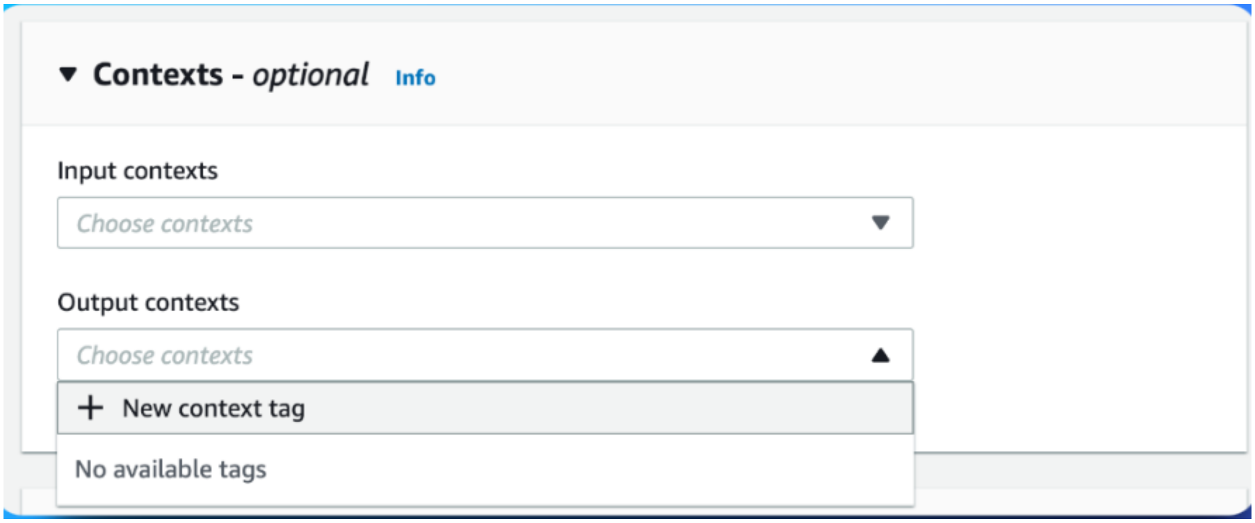
For verification purposes, what is your date of birth?

12/12/1992

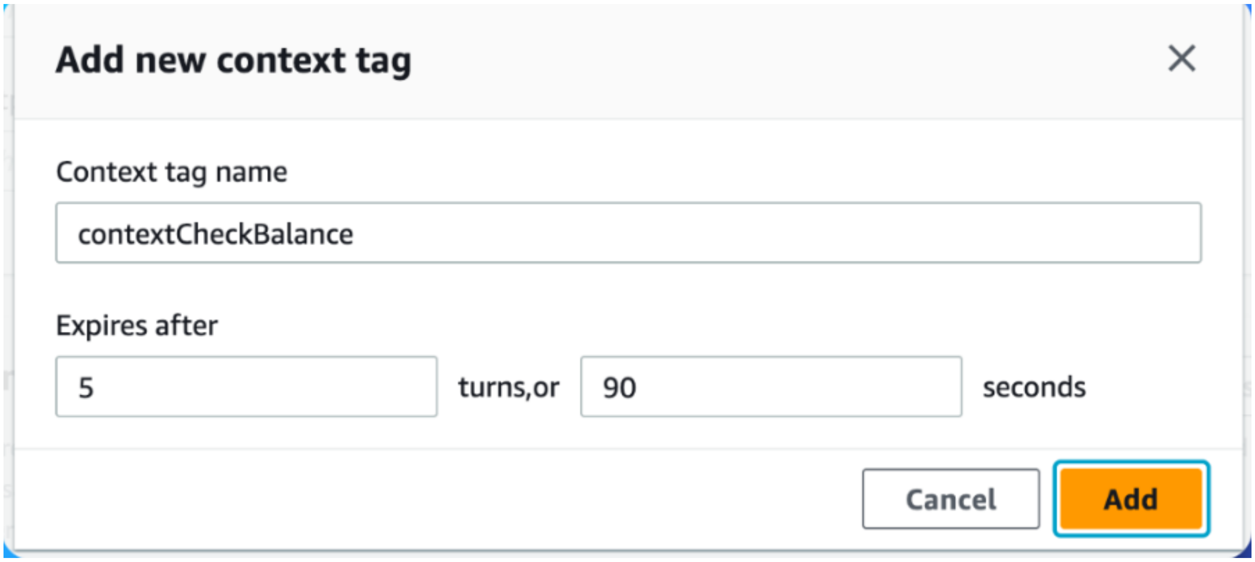
Thank you. The balance on your Credit account is $114.59 dollars.

**STEP 9: SAVE USER INFORMATION WITH THE CHATBOT**

In the **CheckBalance** Intent page, scroll down to the **Contexts** panel.
Under the **Output contexts** dropdown, choose “**New context tag”**.
Contexts tags are used to store and check for specific information across different parts of a conversation. They help to save the user from having to repeat same information.



Name the new context “**contextCheckBalance**” and set up the time turns for **5** mins and **90** secs


Add new context tag

Context tag name
contextCheckBalance

Expires after
5 turns,or 90 seconds

Choose **Add**, **Save** intent, **Build** and **Test**.

**STEP 10: CREATE A FOLLOWUPCHECKBALANCE INTENT**

Imagine that BankerBot just used CheckBalance to give someone their account balance. Now the user has a **follow up question** - "What about my *other* account's balance?"

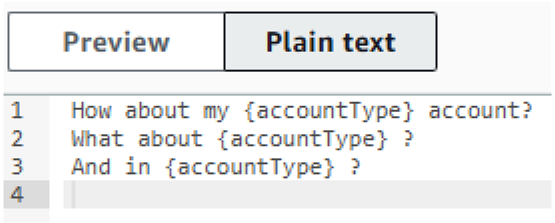
To solve this problem, let's set up a **new Intent** that will handle follow up questions without asking for the user's birthday again

- Head back to the **Intents** page to set up a new intent.
- Choose **Add intent**
- Choose **Add empty**
- Use the following properties to set up the new intent

  **Name**: **FollowupCheckBalance**

  **Description**: **Intent to allow a follow-up balance check request without authentication.**

**Input context**: **contextCheckBalance**
**Sample Utterances**: **How about my {accountType} account?**
**What about {accountType} ?**
**And in {accountType} ?**

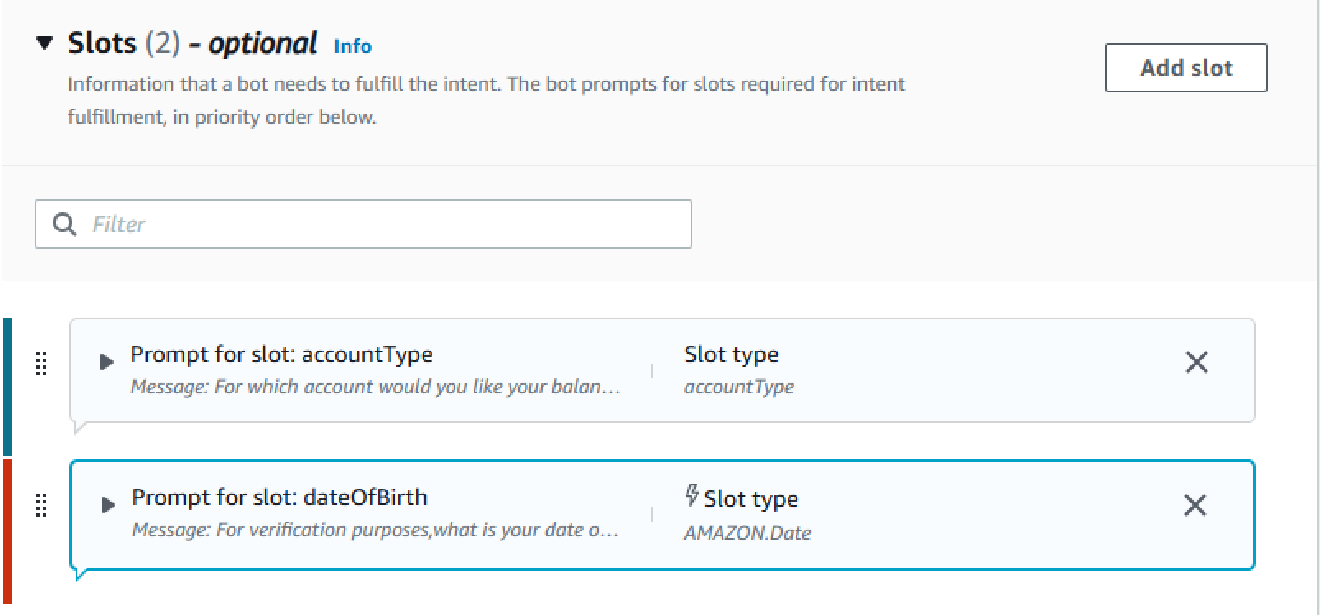


```
How about my {accountType} account?
What about {accountType} ?
And in {accountType} ?
```

**Name**: **accountType**
**Prompt**: **For which account would you like your balance?**
**Slot Type**: **accountType**

- Add another new slot

**Name: dateOfBirth**
**Prompt**: **for verification purposes, what is your date of birth?**
**Slot type**: **AMAZON.Date**



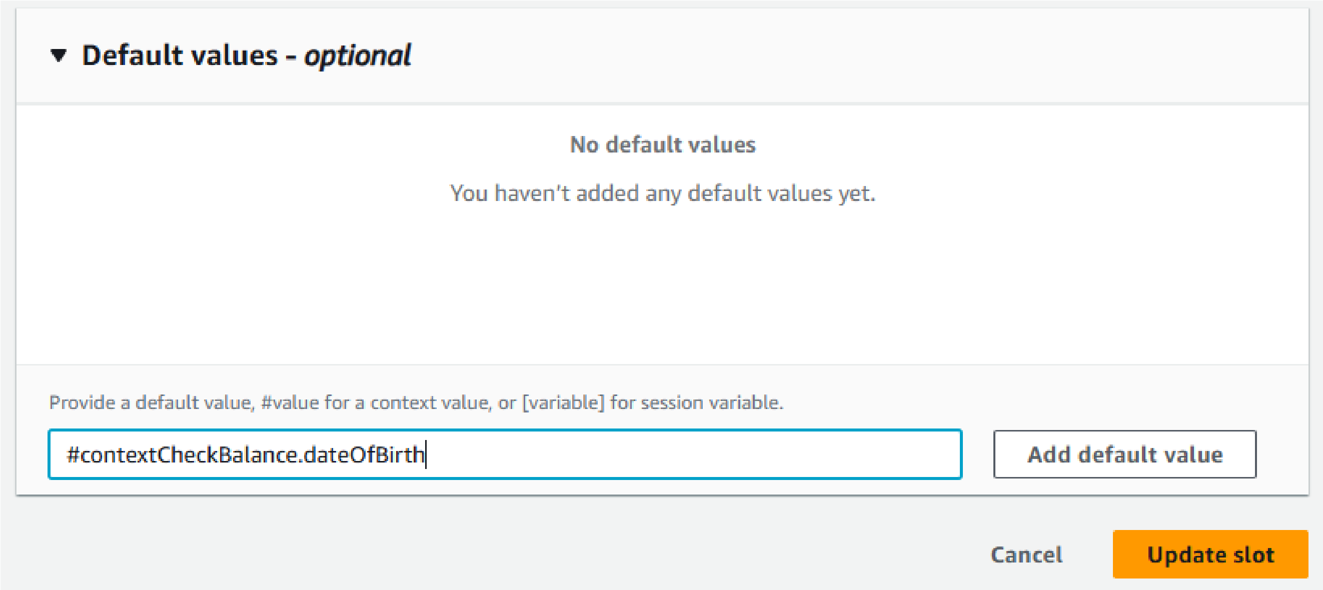
Information that a bot needs to fulfill the intent. The bot prompts for slots required for intent fulfillment, in priority order below.

Prompt for slot: accountType
Message: For which account would you like your balan…
Slot type
accountType

Prompt for slot: dateOfBirth
Message: For verification purposes,what is your date o…
Slot type
AMAZON.Date

- Choose **Save** intent.
- You have to set up FollowupCheckBalance's dateOfBirth slot to use saved information.
- On the **FollowupCheckBalance** Intent page, expand the **dateOfBirth** slot and choose **Advanced options.**
- Scroll down to the **Default values** panel. This panel allows us to create default values for the intent's slots.

- Enter **“#contextCheckBalance.dateOfBirth**”. This allows Amazon Lex to know that the input context which is contextCheckBalance should have the value of dateOfBirth in CheckBalance.
- Choose **Add default value**
- Choose **Update slot**.



▼ Default values - *optional*

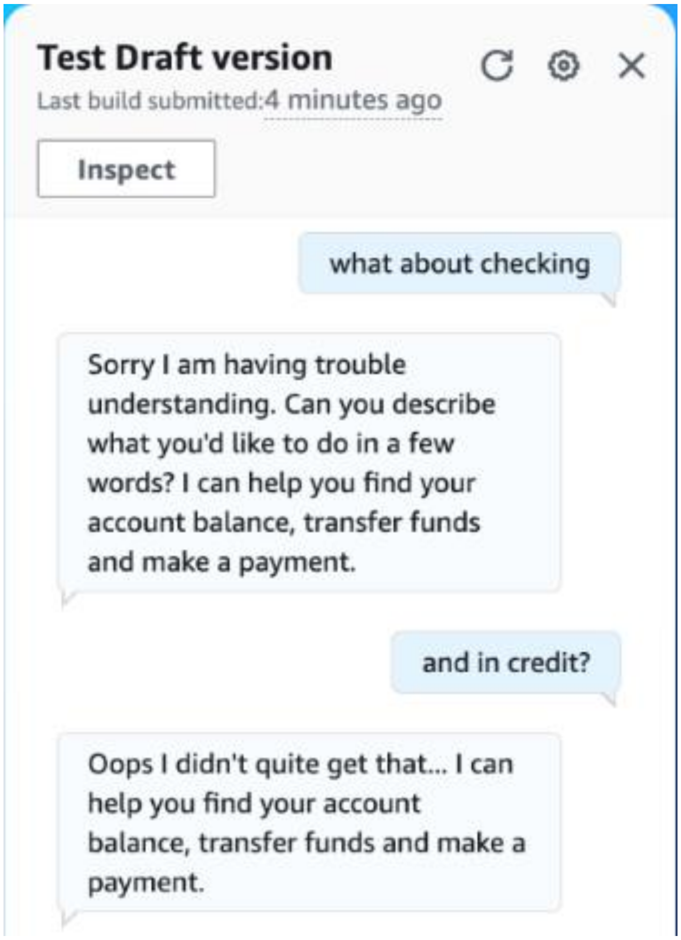
No default values

You haven't added any default values yet.

Provide a default value, #value for a context value, or [variable] for session variable.

#contextCheckBalance.dateOfBirth

- Head to the **Fulfilment panel** to make sure the Lambda function is also connected to the intent.
- Expand “**On successful fulfilment**” and choose “**Advanced options**”.
- Head to the “**Fulfilment Lambda code hook**” panel and select the checkbox to enable a fulfilment Lambda for the intent.
- Choose **Update options**
- Choose **Save intent**
- Choose **Build**
- Choose **Test**
- Ask your chatbot “**What about checking?**”



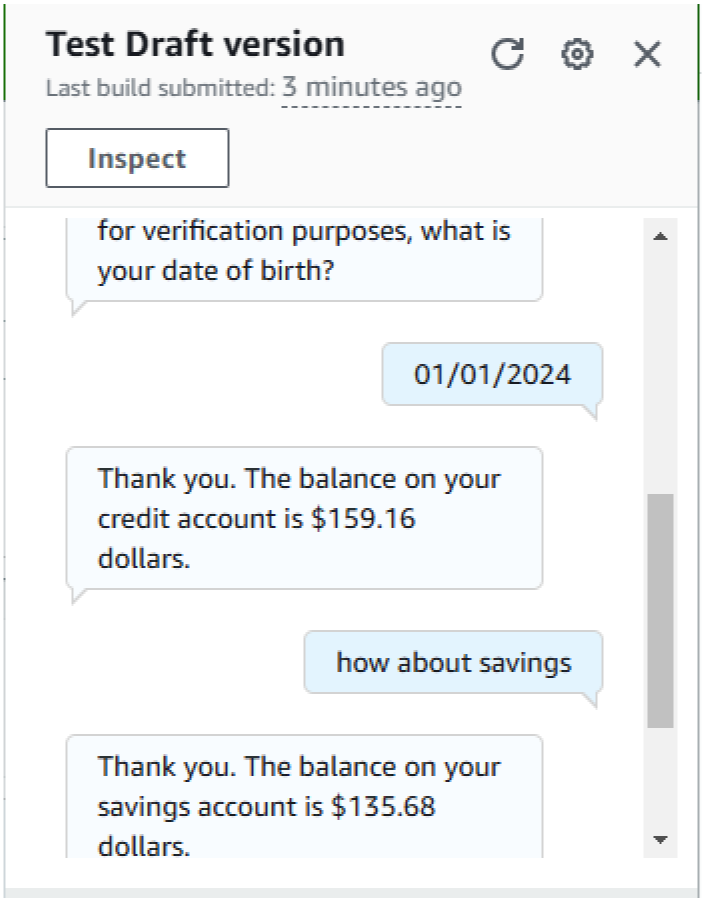
what about checking

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.

and in credit?

Oops I didn't quite get that... I can help you find your account balance, transfer funds and make a payment.

- **Why isn't the intent working?**
  No matter which utterance you use, you will just get an error response. This is because the FollowupCheckBalance intent's input context isn't available yet. The intent doesn't know your birthday! Activate the **CheckBalance** intent first.Then, the context for date of birth will be carried over to the **FollowupCheckBalance** intent.



for verification purposes, what is your date of birth?

01/01/2024

Thank you. The balance on your credit account is $159.16 dollars.

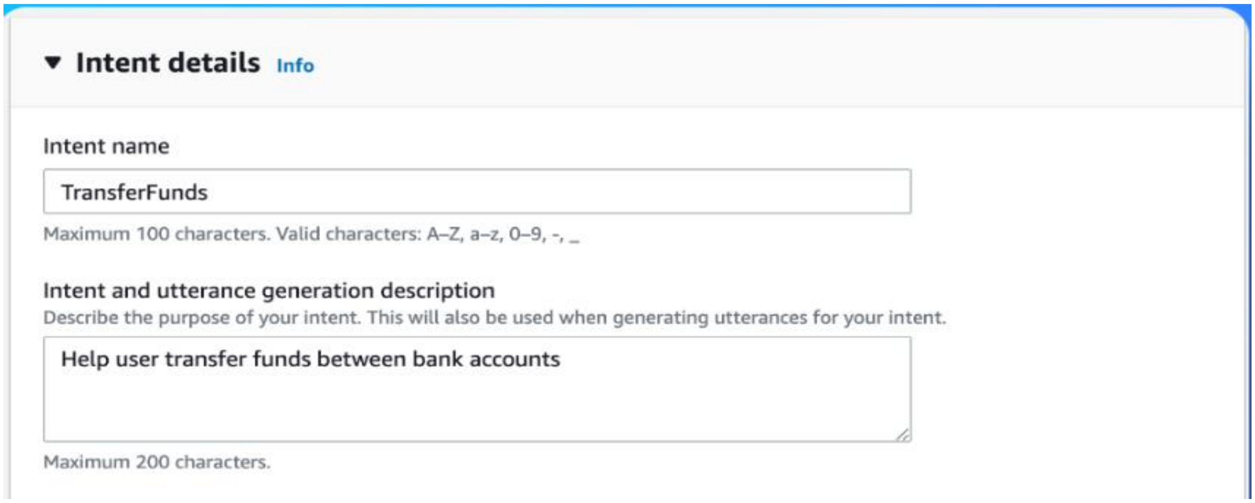
how about savings

Thank you. The balance on your savings account is $135.68 dollars.

### STEP 11: BUILD A CHATBOT WITH MULTIPLE SLOTS

We will create a new **TransferFunds** Intent to help users transfer money between accounts. A new intent that uses two **accountType** slots would be set up.

- Create a new empty intent with the following properties:
  **Name**: **TransferFunds**
  **Description: Help users transfer funds between two accounts**.



Intent name

TransferFunds

Maximum 100 characters. Valid characters: A–Z, a–z, 0–9, -, _

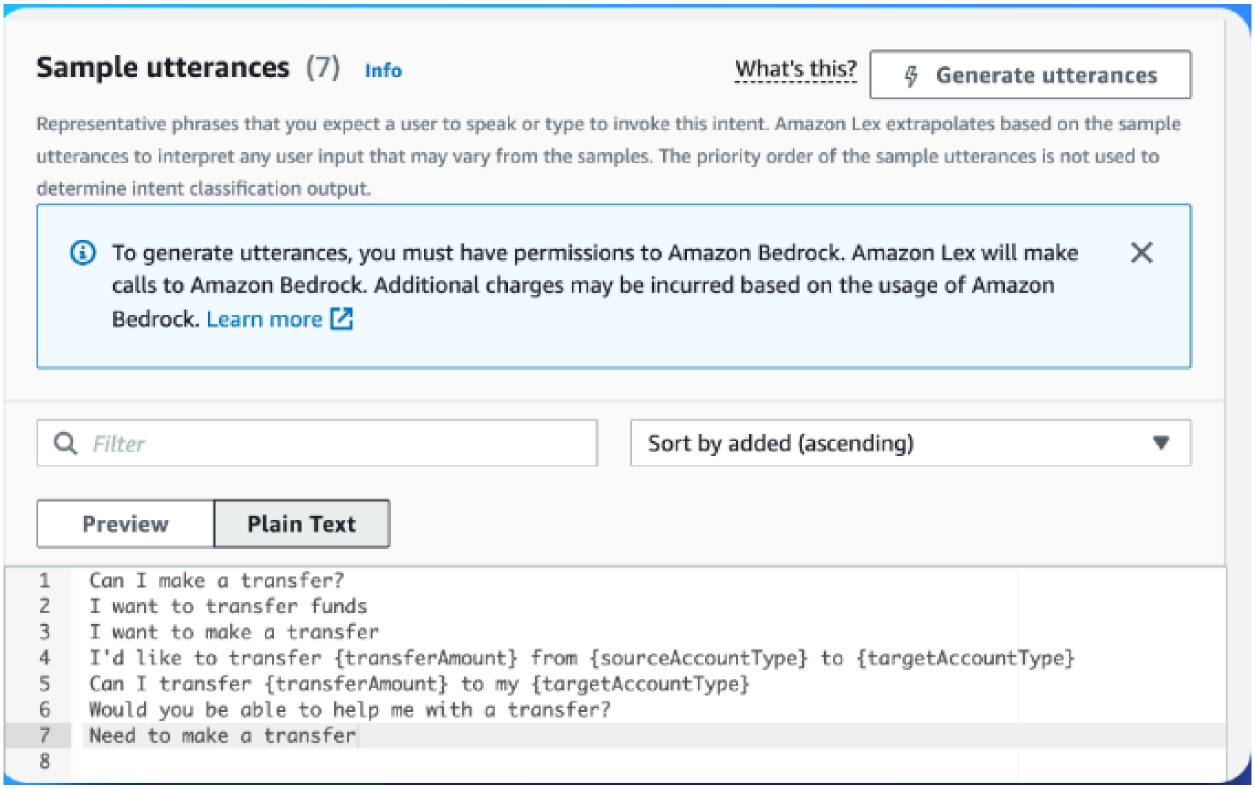
Intent and utterance generation description

Describe the purpose of your intent. This will also be used when generating utterances for your intent.

Help user transfer funds between bank accounts

Maximum 200 characters.

- Paste the sample utterances as plain text
  **Can I make a transfer?**
  **I want to transfer funds**
  **I'd like to transfer {transferAmount} from {sourceAccountType} to {targetAccountType}**
  **Can I transfer {transferAmount} to my {targetAccountType}**
  **Would you be able to help me with a transfer?**
  **Need to make a transfer**



Representative phrases that you expect a user to speak or type to invoke this intent. Amazon Lex extrapolates based on the sample utterances to interpret any user input that may vary from the samples. The priority order of the sample utterances is not used to determine intent classification output.

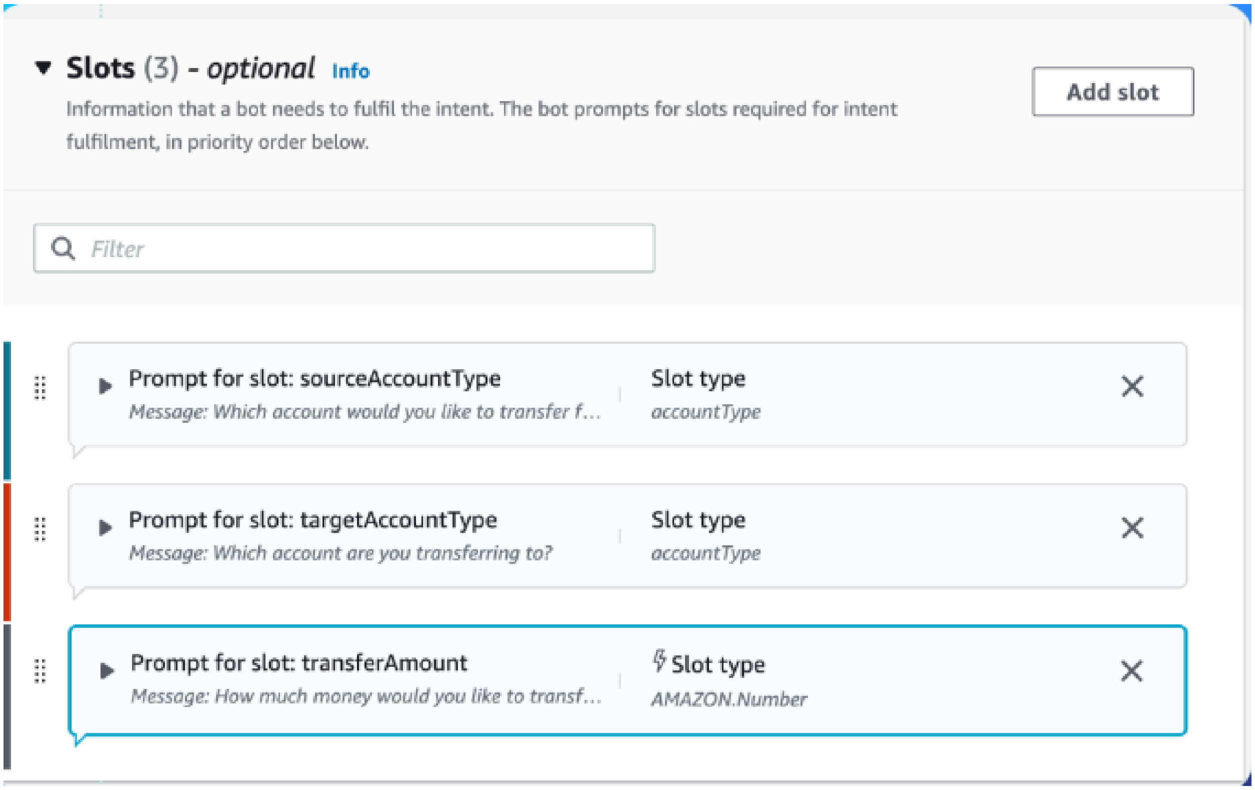
To generate utterances, you must have permissions to Amazon Bedrock. Amazon Lex will make calls to Amazon Bedrock. Additional charges may be incurred based on the usage of Amazon Bedrock. Learn more

```
Can I make a transfer?
I want to transfer funds
I want to make a transfer
I'd like to transfer {transferAmount} from {sourceAccountType} to {targetAccountType}
Can I transfer {transferAmount} to my {targetAccountType}
Would you be able to help me with a transfer?
Need to make a transfer

```

Add a new slot called “**sourceAccountType** with the prompt “**which account would you like to transfer from?” **and the slot type **“accountType”.**

- Add another new slot called “**targetAccountType**” with the prompt “**which account are you transferring to?**” and the slot type “**accountType**”.

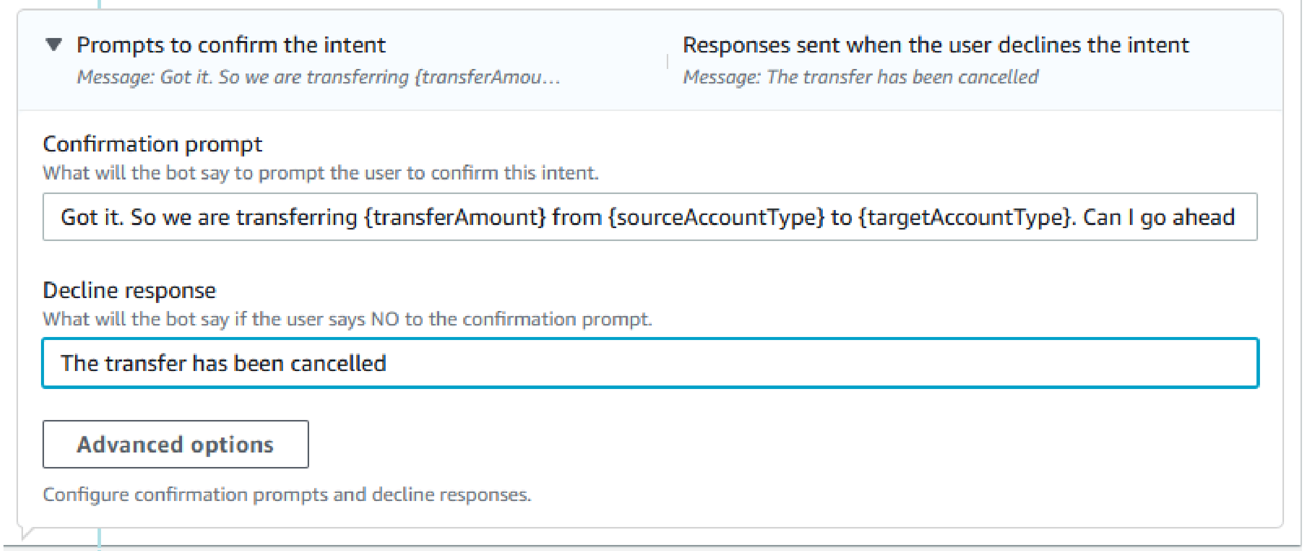
Add another new slot called “**transferAmount**” with the prompt “**how much money would you like to transfer?”** and the slot type “**AMAZON.Number**”.



Information that a bot needs to fulfil the intent. The bot prompts for slots required for intent fulfilment, in priority order below.

| Prompt | Slot type |
|---|---|
| Prompt for slot: sourceAccountType<br>*Message: Which account would you like to transfer f…* | Slot type<br>*accountType* |
| Prompt for slot: targetAccountType<br>*Message: Which account are you transferring to?* | Slot type<br>*accountType* |
| Prompt for slot: transferAmount<br>*Message: How much money would you like to transf…* | Slot type<br>*AMAZON.Number* |

Scroll down to the confirmation panel and enter the following: “**Got it. So we are transferring {transferAmount} from {sourceAccountType} to {targetAccountType}. Can I go ahead with the transfer?**”
In **Decline response**, enter “**The transfer has been cancelled**.”



*Message: Got it. So we are transferring {transferAmou…*

Responses sent when the user declines the intent
*Message: The transfer has been cancelled*

Confirmation prompt
What will the bot say to prompt the user to confirm this intent.

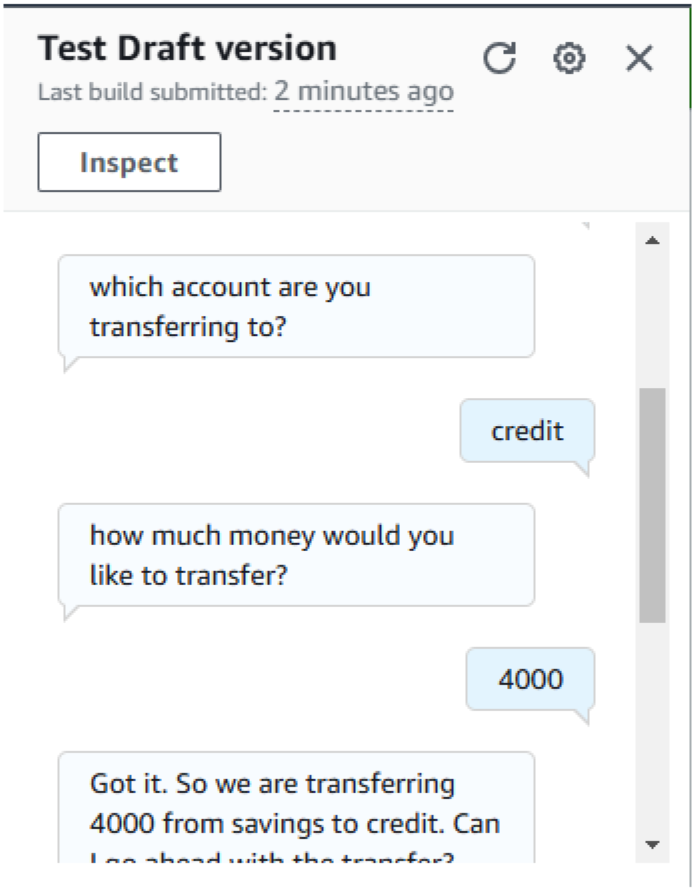
Got it. So we are transferring {transferAmount} from {sourceAccountType} to {targetAccountType}. Can I go ahead

Decline response
What will the bot say if the user says NO to the confirmation prompt.

The transfer has been cancelled

Configure confirmation prompts and decline responses.

Scroll to the **Closing response** pane and add the message “**The transfer is complete. {transferAmount} should now be available in your {targetAccountType} account.**”
Choose **Save intent** and choose **Build**.
Choose **Test** and ask your chatbot : **I’d like to transfer money and continue the conversation**.



which account are you transferring to?

credit

how much money would you like to transfer?

4000

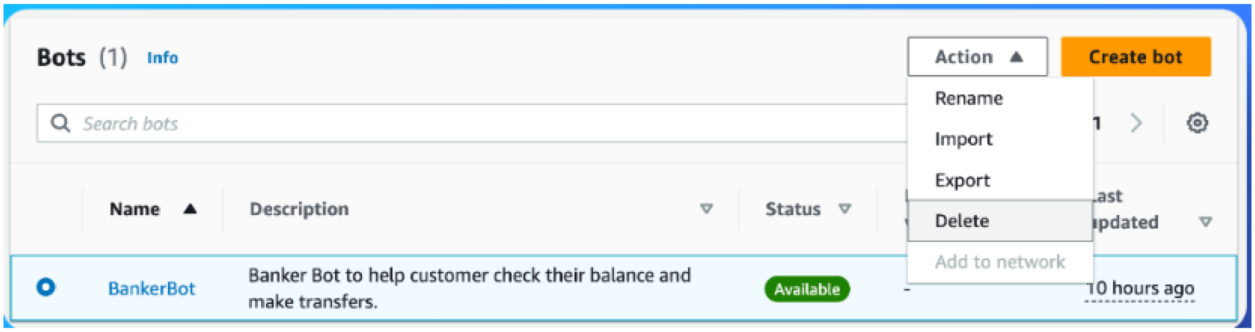
Got it. So we are transferring 4000 from savings to credit. Can I go ahead with the transfer?

**STEPS BELOW ARE TO DELETE THE RESOURCES AFTER USE**
Deleting resources after use is very important as it avoids unnecessary costs and reduces the risks of unauthorized access.

- Head to your **Amazon Lex** console.
- Choose **Bots** on the left-hand sidebar.
- Choose the circle radio button next to BankerBot.
- Choose **Delete** from your Action drop-down.
- Choose **Delete.**

**Delete your BankerBot**

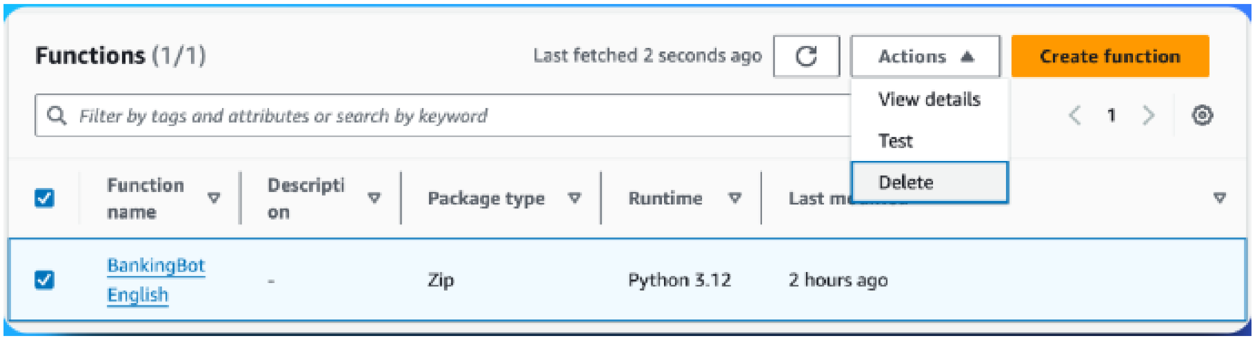
- Head to your **Amazon Lex** console.
- Choose **Bots** on the left-hand sidebar.
- Choose the circle radio button next to **BankerBot**.
- Choose **Delete** from your **Action** drop-down.
- Choose **Delete**.



| | Name ▲ | Description | Status | | Last updated |
|---|---|---|---|---|---|
| ● | BankerBot | Banker Bot to help customer check their balance and make transfers. | Available | - | 10 hours ago |

### Delete your Lambda function

- Head to your **AWS Lambda** console.
- Choose **Functions** on the left-hand sidebar.
- Choose the circle radio button next to **BankingBotEnglish**.
- Choose **Delete** from your **Action** drop-down.
- Choose **Delete**.


Functions (1/1)
Last fetched 2 seconds ago
Actions ▲
Create function
View details
Test
Delete
Filter by tags and attributes or search by keyword
1

| ☑ | Function name | Description | Package type | Runtime | Last m |
|---|---|---|---|---|---|
| ☑ | BankingBot English | - | Zip | Python 3.12 | 2 hours ago |

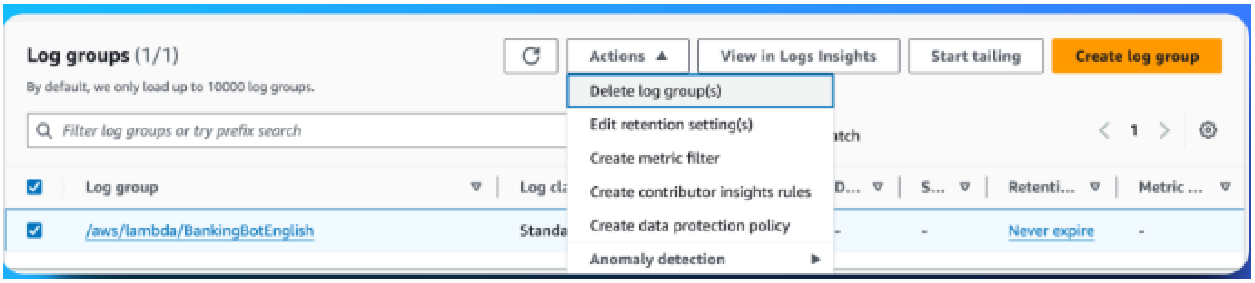
Delete your Lambda function log files

**What are Lambda function log files?**
These log files are records of events that happen when AWS Lambda functions are running. These logs include details like errors, warnings, and informational messages that help developers understand how their functions are performing.

In this project, log files have been produced each time your Lambda function was triggered i.e., when CheckBalance gives you a bank balance number.

The logs are stored in CloudWatch, an AWS service used for monitoring and managing all the log data across all your services.

- Head to **CloudWatch** in your AWS console.
- Choose **Logs** on the left-hand sidebar.
- Choose **Log groups.**
- Select the checkbox next to **BankingBot**.
- Choose **Delete log group(s)** in the **Actions** menu.
- Choose **Delete**.


Actions
View in Logs Insights
Start tailing
Create log group
Delete log group(s)
Edit retention setting(s)
Create metric filter
Create contributor insights rules
Create data protection policy
Anomaly detection
Filter log groups or try prefix search
1

Log groups (1/1)

By default, we only load up to 10000 log groups.

| ☑ | Log group | Log cla | D... | S... | Retenti... | Metric ... |
|---|---|---|---|---|---|---|
| ☑ | /aws/lambda/BankingBotEnglish | Standa | - | - | Never expire | - |

## SUMMARY

This documentation walked through the process of building a chatbot from understanding is purpose to developing and deploying it. By following these steps, you should be able to create a chatbot that improves user interaction and automates responses effectively and efficiently.