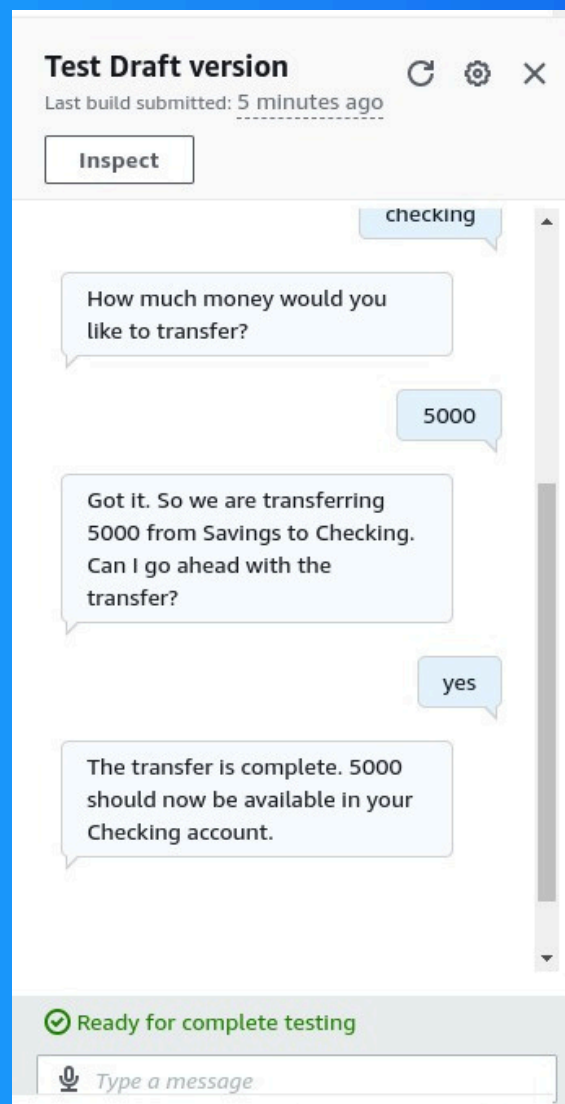




# Build a Chatbot with Multiple Slots



**Hassan Gachoka**





**Hassan Gachoka**

[linkedin.com/gachokahassan](https://www.linkedin.com/in/gachokahassan)

[NextWork.org](https://NextWork.org)

# Introducing Today's Project!

## What is Amazon Lex?

Amazon Lex is a fully managed service for building conversational interfaces using voice and text. It simplifies creating chatbots and virtual assistants, integrating with AWS services to enable scalable and intelligent interactions.

## How I used Amazon Lex in this project

In today's project, I used Amazon Lex to create a chatbot for managing account-related tasks, such as checking balances and transferring funds. Lex's natural language processing helped the bot understand and respond to user queries effectively.

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

One thing I didn't expect in this project was configuring multiple slot types for different account types in the TransferFunds intent. It required careful planning to ensure smooth interaction between the account types and user inputs.

## This project took me...

This project took me one and a half hours to complete, including setting up intents, troubleshooting, and deploying the bot. Most of the time was spent configuring slot types and ensuring proper integration with AWS CloudFormation.



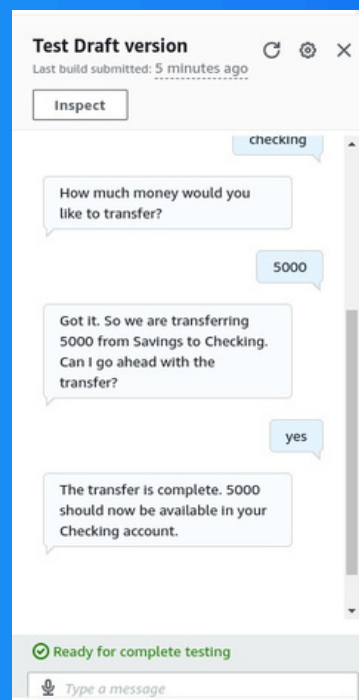
**Hassan Gachoka**

[linkedin.com/gachokahassan](https://www.linkedin.com/in/gachokahassan)

[NextWork.org](https://NextWork.org)

# TransferFunds

An intent I created for my chatbot was TransferFunds, which lets users transfer money between accounts. It uses three slots: sourceAccountType and destinationAccountType, and transferAmount, making fund transfers smooth and efficient.





# Using multiple slots

For this intent, I had to use the same slot type twice. This is because both the source and destination accounts use the slot type "accountType." It ensures the chatbot correctly identifies and differentiates the two accounts for transfers.

I also learned how to create confirmation prompts, which are messages that verify a user's intent before action is taken. They ensure accuracy by asking for confirmation, like "Do you want to transfer \$50 from savings to checking?"

**Confirmation** Info Active

Prompts help to clarify whether the user wants to fulfill the intent or cancel it.

▼ Prompts to confirm the intent

Message: Got it. So we are transferring {transferAmount}...

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.

**Advanced options**

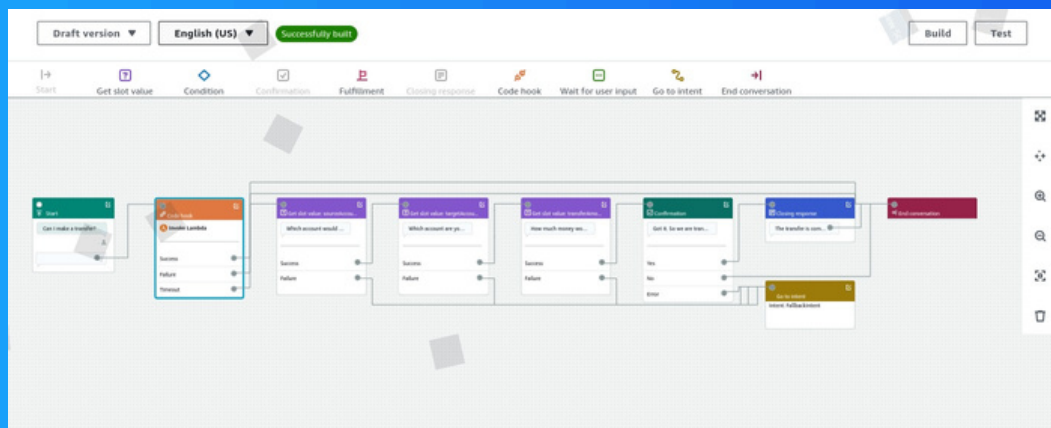
Configure confirmation prompts and decline responses.



# Exploring Lex features

Lex also has a special conversation flow feature that visually maps out the interactions between the user and the chatbot. It helps design, review, and refine dialog paths, making it easier to ensure smooth and intuitive user experiences.

You could set up your intent using a visual builder! It simplifies chatbot design by letting you drag and drop elements, making it easy to visualize user interactions, set up intents, and manage conversation flows in an intuitive way.





**Hassan Gachoka**

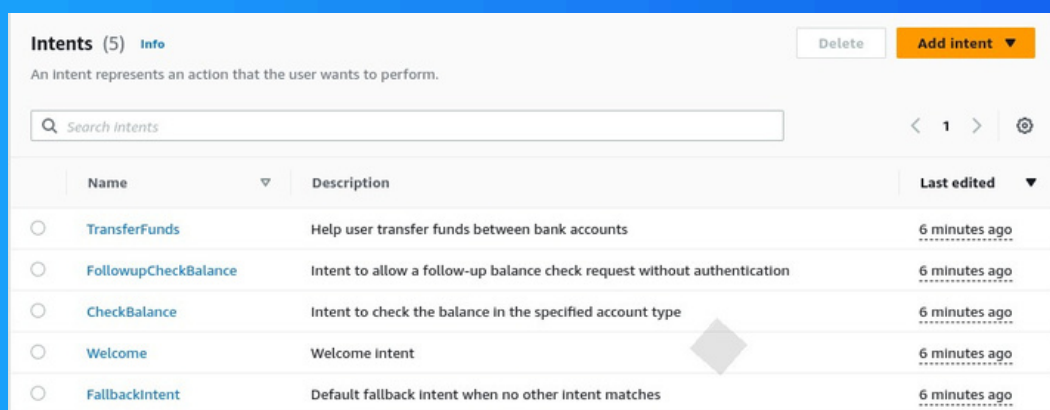
[linkedin.com/gachokahassan](https://www.linkedin.com/in/gachokahassan)

[NextWork.org](https://NextWork.org)

# AWS CloudFormation

AWS CloudFormation is a service that automates resource provisioning by allowing you to define infrastructure as code using templates. It simplifies deployment, ensures consistency, and supports rollback for seamless management of AWS resources.

I used CloudFormation to automate the deployment of AWS resources for my chatbot project, including Lex bots and integrations. It ensured consistency and simplified managing resources, saving time and reducing manual errors in setup.



The screenshot shows the AWS Lex 'Intents' console. At the top, it says 'Intents (5)' with an 'Info' link. Below this is a description: 'An Intent represents an action that the user wants to perform.' There is a search bar labeled 'Search intents' and navigation controls showing '1' of 5 items. A table lists the following intents:

	Name	Description	Last edited
<input type="radio"/>	TransferFunds	Help user transfer funds between bank accounts	6 minutes ago
<input type="radio"/>	FollowupCheckBalance	Intent to allow a follow-up balance check request without authentication	6 minutes ago
<input type="radio"/>	CheckBalance	Intent to check the balance in the specified account type	6 minutes ago
<input type="radio"/>	Welcome	Welcome intent	6 minutes ago
<input type="radio"/>	FallbackIntent	Default fallback intent when no other intent matches	6 minutes ago



## The final result!

Re-building my bot with CloudFormation took me like seven minutes to set up the resources and configurations properly. Once the template was in place, the deployment was much faster and more efficient compared to manual setup.

There was an error after I deployed my bot! The error was related to missing IAM role permissions for certain resources. I fixed this by reviewing the CloudFormation template, adding the correct permissions, and testing again.

The screenshot shows the 'Add permissions' page in the AWS IAM console. The breadcrumb trail at the top reads: 'Lambda > Functions > testfunctionDELETethis > Add permissions'. The main heading is 'Add permissions'. Below this is a section titled 'Edit policy statement' with three radio button options: 'AWS account' (selected), 'AWS service' (selected), and 'Function URL'. The 'AWS service' option is highlighted with a blue border and a grey arrow pointing to it. Below the radio buttons are several input fields: 'Service' (a dropdown menu with 'Other' selected), 'Statement ID' (a text input field containing 'my-custom-permission-amazonlexchatbot'), 'Principal' (a text input field containing 'lexv2.amazonaws.com'), 'Source ARN' (a text input field containing 'arn:aws:lexus-west-2:471112976395:bot-alias/\*'), and 'Action' (a dropdown menu with 'lambda:InvokeFunction' selected). At the bottom right of the form are 'Cancel' and 'Save' buttons.