



# Save User Info with a Lex Chatbot



Gursimran Singh

The screenshot shows the AWS Lambda function test interface. On the left, there's an "Inspect" panel with a "JSON input and output" tab selected, displaying the response object for a user query. The response object includes details like content type, session state, dialog action, intent name (FollowupCheckBalance), and slot values for account type (Savings). On the right, there's a "Test Draft version" window showing a simulated conversation. The user asks for checking and savings account balances, and the bot responds with the respective amounts (\$64.74 and \$293.42). A green banner at the bottom of the test window indicates "Ready for complete testing".

Sort by added (ascending)

Inspect

Summary JSON input and output

Response

```
    "contentType": "PlainText"
},
],
"sessionState": {
"dialogAction": {
"type": "Close"
},
"intent": {
"name": "FollowupCheckBalance",
"slots": {
"accountType": {
"value": {
"originalValue": "Savings",
"interpretedValue": "Savings"
"resolvedValues": [
]
}
}
}
}
}
```

Test Draft version

Last build submitted: 4 minutes ago

10/11/2003

Thank you. The balance on your Checking account is \$64.74 dollars.

what about my Savings?

Thank you. The balance on your Savings account is \$293.42 dollars.

Ready for complete testing

Type a message

Save intent

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences



# Introducing Today's Project!

## What is Amazon Lex?

Amazon Lex is an AWS tool for building chatbots and voice apps. It's useful because it lets your bot understand what users say, respond naturally, and even connect to backend systems to do things like fetch data or remember details.

## How I used Amazon Lex in this project

In today's project, I used Amazon Lex to build a chatbot that remembers user info using context carryover. It helped me set up multiple intents that share data, making the conversation feel smoother and more natural.

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

One thing I didn't expect in this project was how easy it is to make the chatbot remember user info across intents. Context carryover made it feel way more advanced without needing complex logic or extra coding.



**Gursimran Singh**  
NextWork Student

[nextwork.org](http://nextwork.org)

## This project took me...

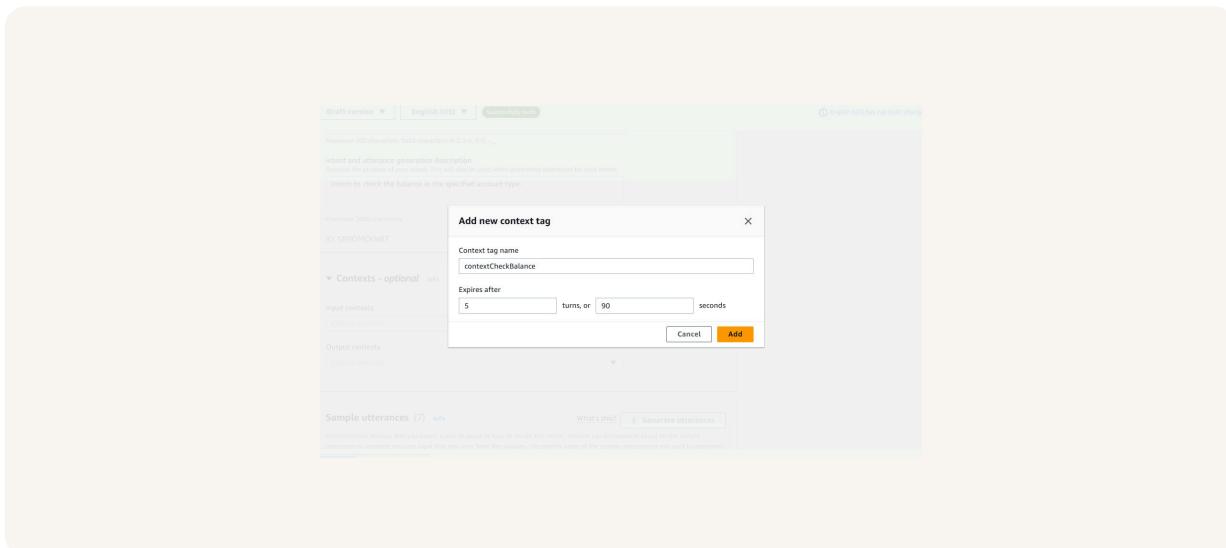
This project took me about 1 hour to complete. Most of the time went into testing how context carryover worked between intents and making sure the chatbot remembered user info like birthday correctly.

# Context Tags

Context tags are labels used in Amazon Lex to pass information from one intent to another. They help the chatbot remember things a user said earlier—like a birthday—so it can use that info in future parts of the conversation.

There are two types of context tags: \*input\* and \*output\*. Output context tags are set by an intent to store info, while input context tags are used by another intent to access that stored info. Together, they help share data between intents.

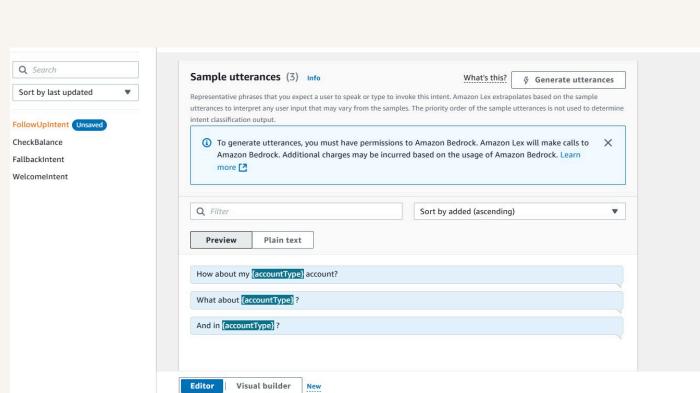
I created a context tag called `contextCheckBalance`. This context tag was created in the CheckBalance intent. This tag stores information about the user's birthday so it can be passed to other intents that need it later in the conversation.



# FollowUpCheckBalance

I created a new intent called FollowupCheckBalance. The purpose of this intent is to handle follow-up balance questions without asking for the user's birthday again by reusing the info already collected through context carryover.

This intent is connected to the previous intent I made, CheckBalance, because it uses the birthday saved by CheckBalance through context carryover, allowing the chatbot to handle follow-up questions without asking for the same info again.

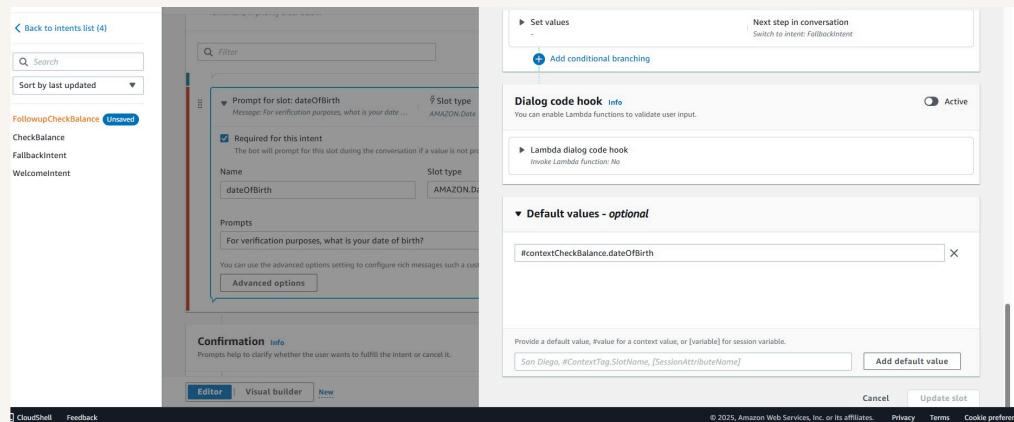


Gursimran Singh  
NextWork Student

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

# Input Context Tag

I created an input context, contextCheckBalance, that connects to the output context from the CheckBalance intent. This allows FollowupCheckBalance to access the saved birthday without asking the user again.



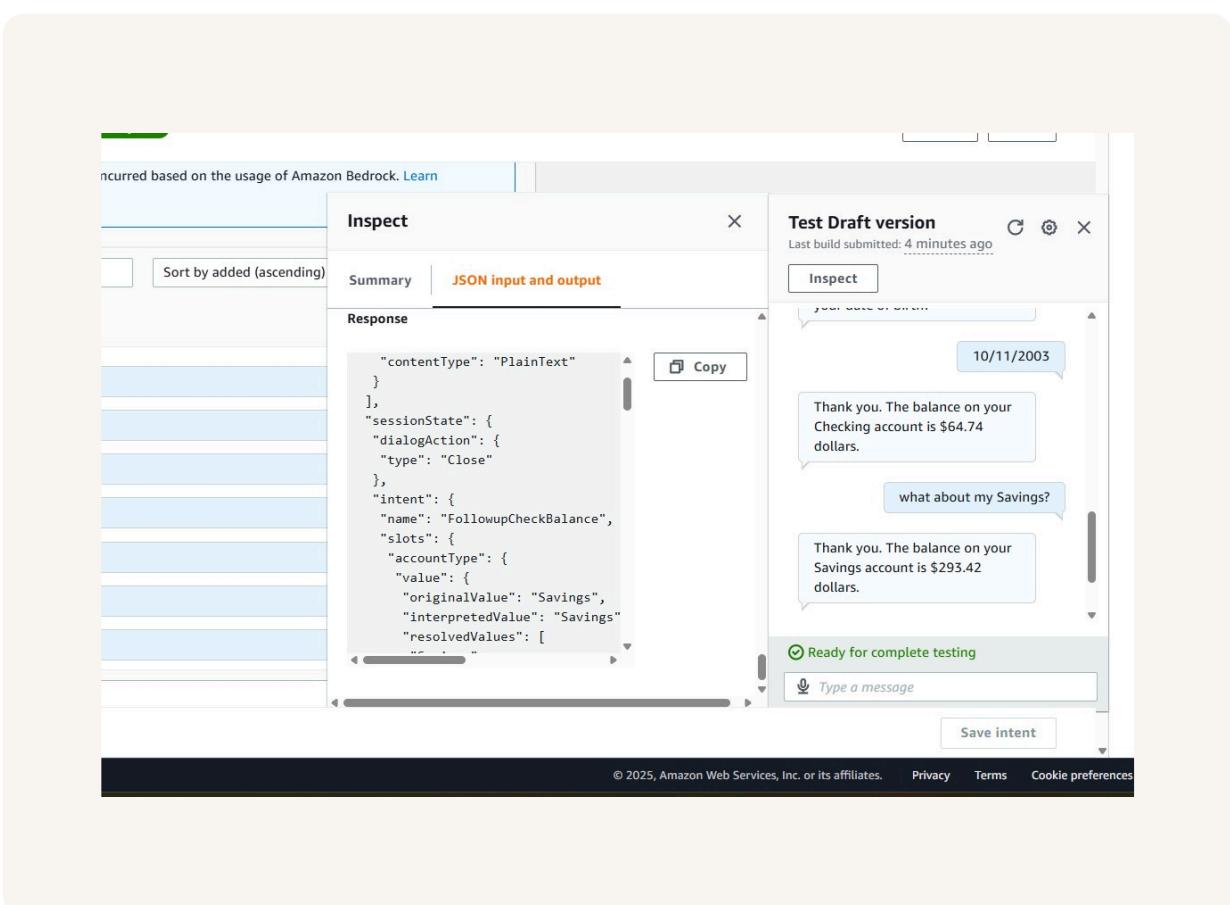
Gursimran Singh  
NextWork Student

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

# The final result!

To see the context tags and followup intent in action, I first asked my chatbot to check my balance, then said something like “What about my savings account?” This triggered FollowupCheckBalance using the saved birthday info.

If I had gone straight to trying to trigger FollowupCheckBalance without setting up any context, the chatbot wouldn’t have the user’s birthday info, so it would either ask for it again or fail to respond correctly, breaking the conversation flow.





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

# The place to learn & showcase your skills

Check out [nextwork.org](https://nextwork.org) for more projects

