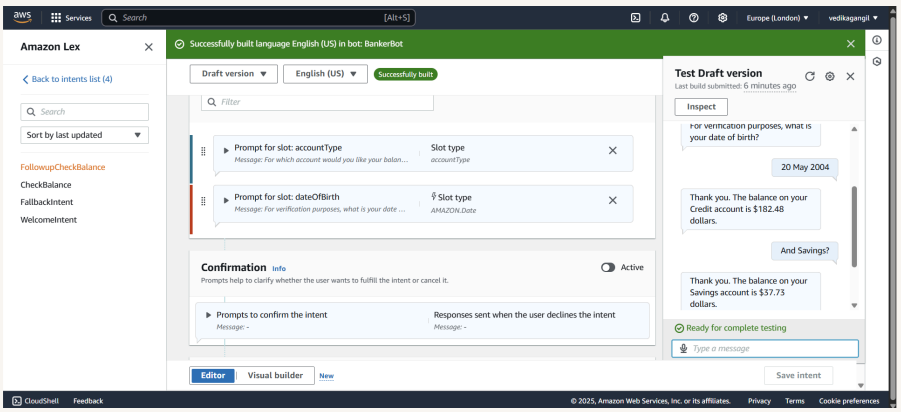
nextwork.org

Save User Info with a Lex Chatbot

VE vedikagangil@gmail.com



VE
vedikagangil@gmail.com
NextWork Student

Introducing Today's Project!

What is Amazon Lex?

Amazon Lex is an AWS service for building conversational AI chatbots using voice/text. It’s useful because it has pre-built NLP which understands intents/slots and since Lex is very highly scalable.

How I used Amazon Lex in this project

In today’s project, I used Amazon Lex to: 1. Build a banking chatbot with intents like CheckBalance and TransferFunds. 2. Train slots to extract user inputs. 3. Test conversations in the Lex console.

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

One thing I didn't expect in this project was how quick and easy to understand it was.

This project took me...

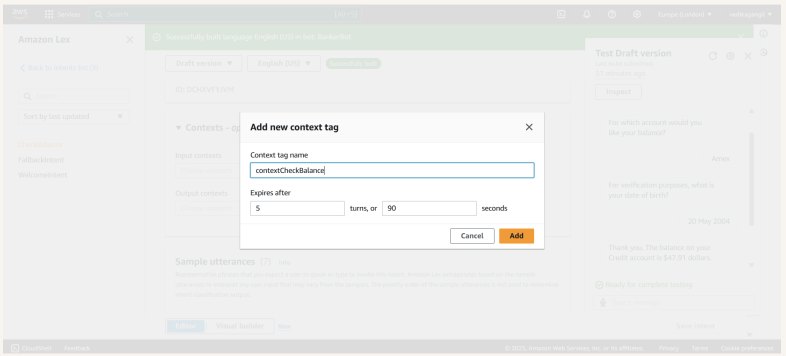
This project took me around 60 to 70 minutes.

Context Tags

Context tags in Amazon Lex are used to store and check for specific information across different parts of a conversation. They help save the user from having to repeat certain information. There are 2 types: Input context tags and Output Context Tags

There are two types of context tags in Lex: 1. Output context tag: This tells the chatbot to remember certain details after an intent is finished. 2. Input context tag: This checks if specific details are already available before an intent activates

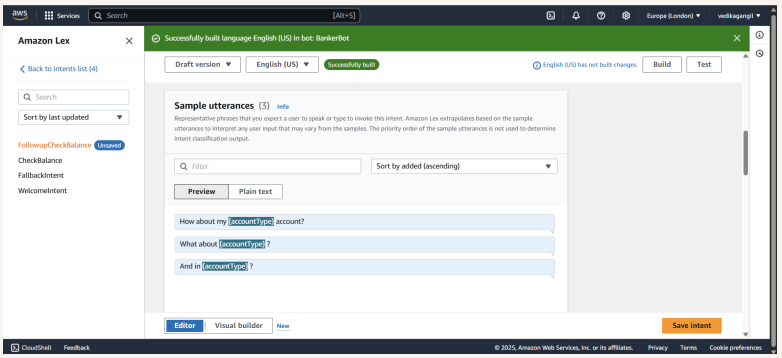
I created a context tag called contextCheckBalance. This context tag was created in the intent CheckBalance. This tag stores information about user's birthday.



FollowUpCheckBalance

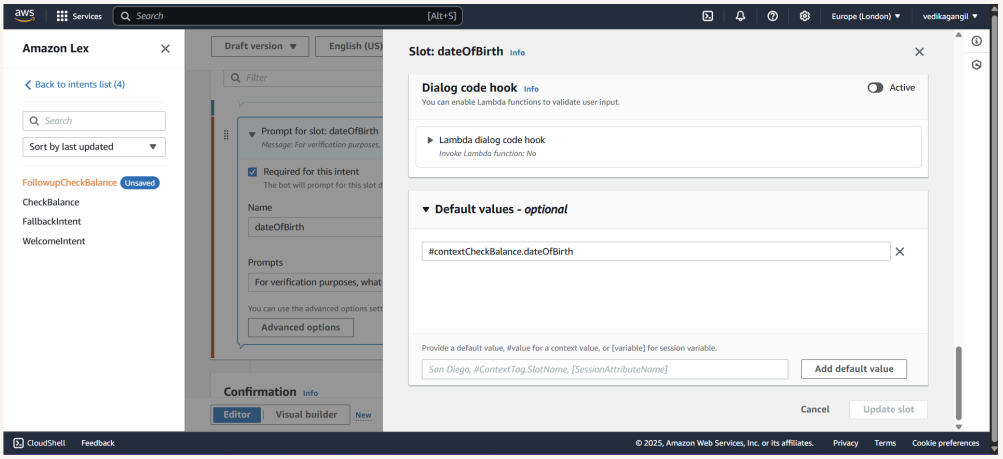
I created a new intent called FollowupCheckBalance. It handles incomplete CheckBalance requests by prompting users for missing slot values. It activates when slots are empty or it reprompts dynamically using Lambda validation if needed.

FollowupCheckBalance and CheckBalance are connected as a conversational flow: CheckBalance runs when the intent is triggered. If slots are missing, Lex auto-fires FollowupCheckBalance to reprompt.



Input Context Tag

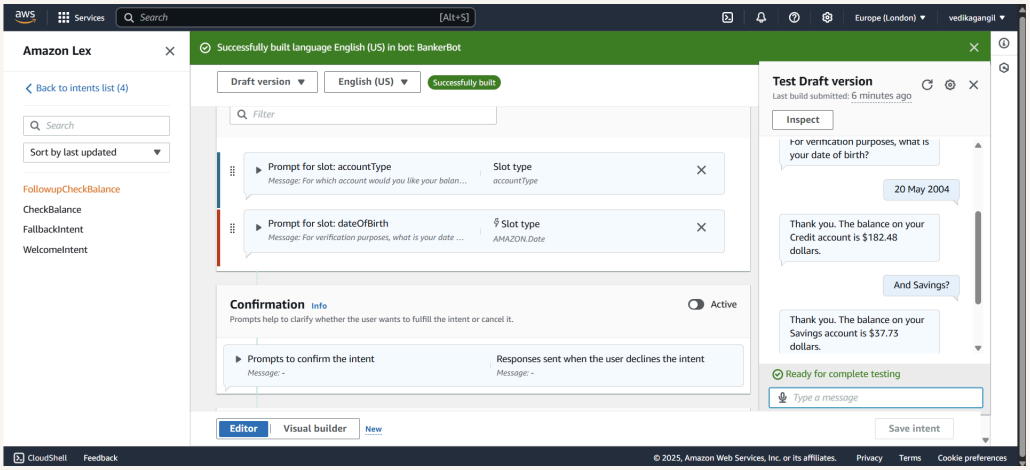
ContextCheckBalance is connected to the output context to maintain conversational state across intents. When CheckBalance runs, it passes key data to this context. FollowupCheckBalance checks this context to recall prior slot values.



The final result!

To see the context tags and followup intent in action, I used incomplete phrases like: 1. "What's my balance?" (Missing AccountType slot). 2. "Check funds" (No account specified).

If FollowupCheckBalance triggers without context, Lex will: 1. Reprompt for all required slots. 2. Reset the dialog if slots stay empty after max retries . This ensures critical data is collected before proceeding.



The place to learn & showcase your skills

Check out nextwork.org for more projects