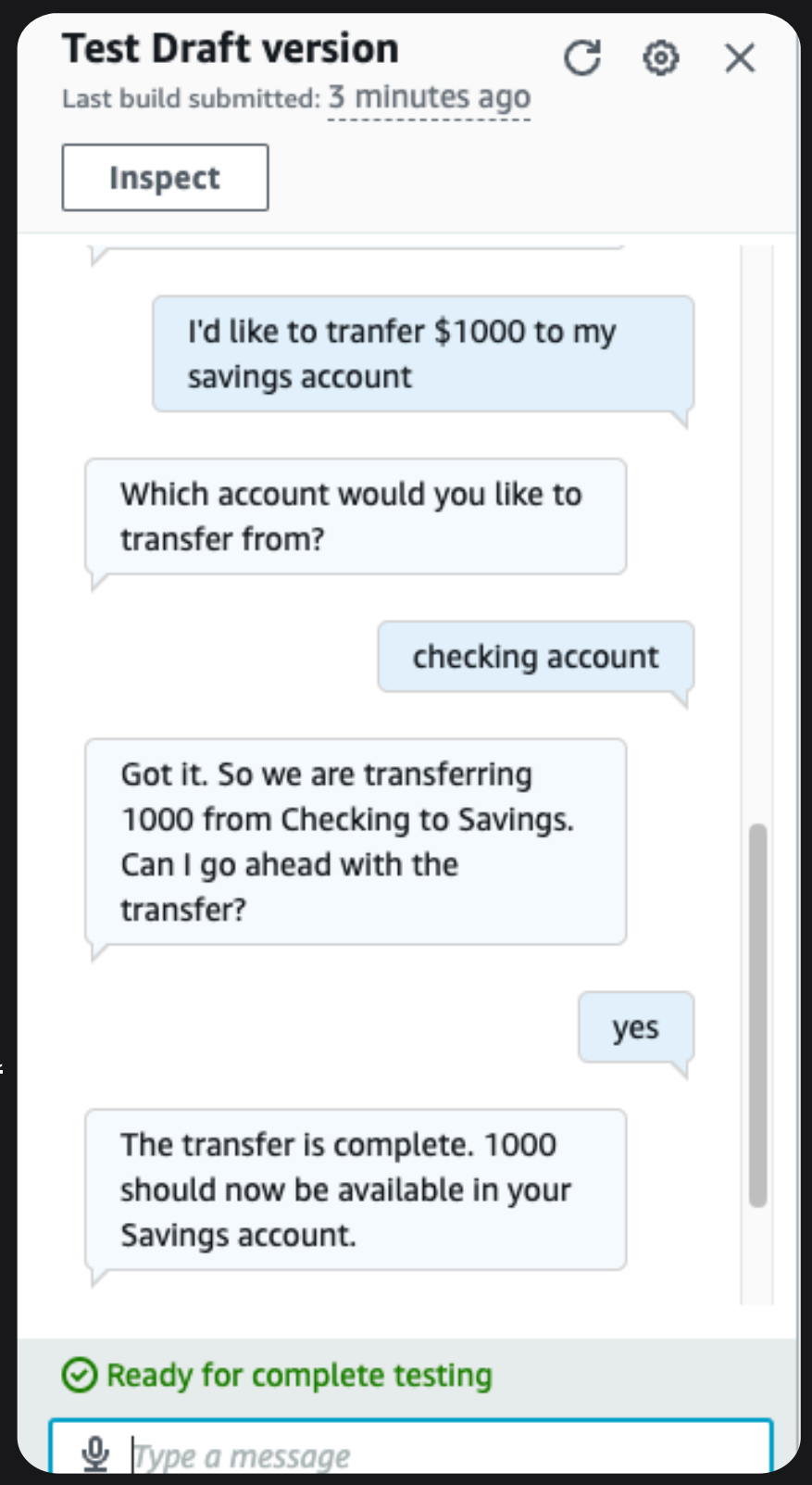
NEXTWORK

How I built a chatbot with Amazon Lex

with multiple slots! 🎰



Megha Naik

https://www.linkedin.com/in/naikmegha

What is Amazon Lex?

What it does:

- It helps build chatbots capable of understanding and responding to human language, whether spoken or typed.

Why it's useful:

- It makes it easier to create chatbots that can handle customer inquiries, automate tasks, and provide information without needing human intervention all the time.

How I'm using it in today's project:

- In this project, I'm using Amazon Lex to create BankerBot to set up a final intent for transferring money across accounts.

Megha Naik
https://www.linkedin.com/in/naikmegha

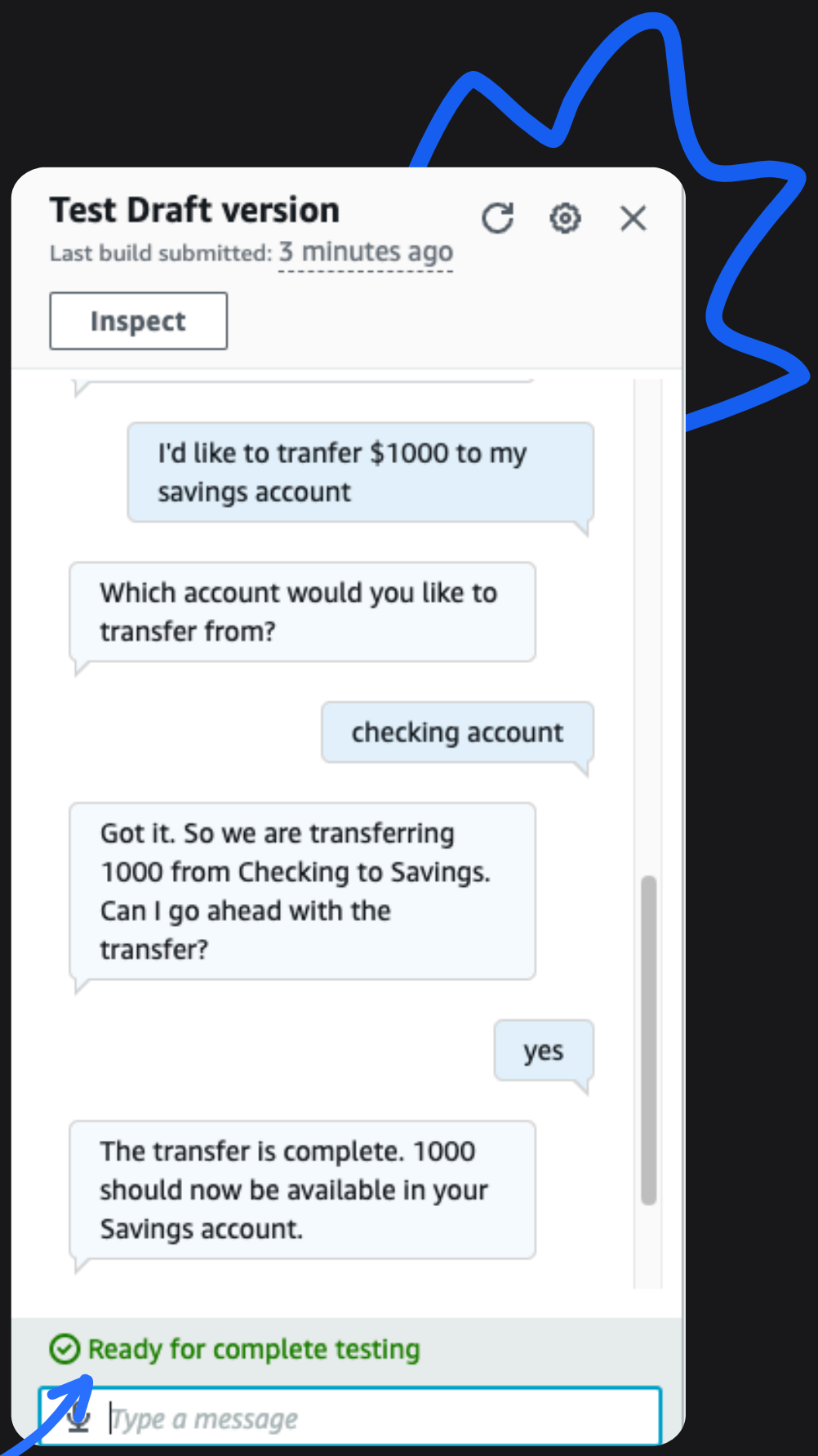
More slots!

- Slots are pieces of information that my chatbot needs to fulfill an intent.

- The final intent for my chatbot was TransferFunds, which will help the user transfer money between bank accounts.

- For this intent, I had to use the same slot type twice. This is because the TransferFunds intent involved two different accounts- the source account (i.e the account that we are transferring money from) and the target account (i.e the account that the money will land in).

- I also learnt how to create confirmation prompts, which are prompts designed for the chatbot to confirm the user's intention to carry out the intent. In this project, a confirmation prompt was used for the chatbot to confirm that the user is wanting to transfer a specific amount of money between two of their bank accounts.



A conversation demonstrating the two slots and the confirmation prompts in action!

Megha Naik
https://www.linkedin.com/in/naikmegha

A LITTLE EXTRA...

Deploying with CloudFormation

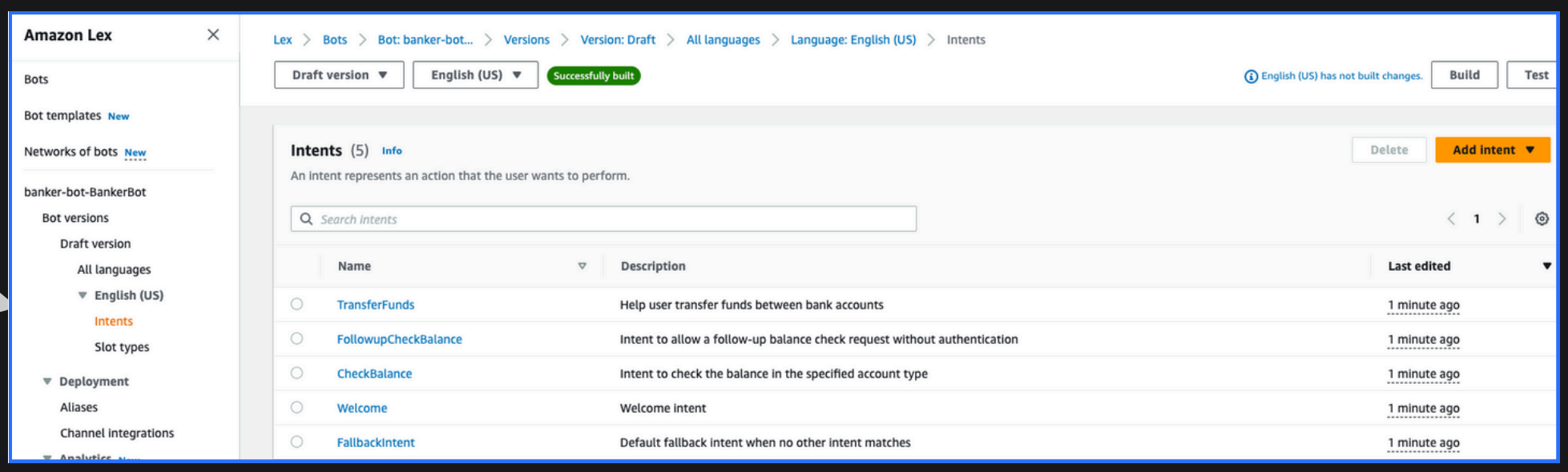
- AWS CloudFormation is service that helps users deploy AWS resources in seconds, by defining the resources and their characteristics in a code file (called a YAML file).

- As an extension to this project, I learnt how to deploy the entire BankerBot using a single CloudFormation stack.

- Doing this took me 1 minute to set up the CloudFormation stack. 5 minutes to wait for the deployment to complete.

- Something I learnt from deploying with CloudFormation was that I can deploy resources from all the different AWS services in the exact same YAML file.
-

CloudFront deployed this for me!



Megha Naik
https://www.linkedin.com/in/naikmegha

My Key Learnings

01

We can use the accountType slot type twice in the same intent by assigning different slot names to distinguish between them, like sourceAccountType and targetAccountType, ensuring clarity in slot identification

02

Confirmation prompts typically repeats back information for the user to confirm. If the user confirms the intent, the bot fulfills the intent. If the user declines, then the bot responds with a decline response that you set up.

03

AWS CloudFormation is a service that gives you an easy way to create and set up AWS resources. It creates a CloudFormation template that describes all the resources you want to create and their dependencies as code. Then, you can use that template to create, update, and delete the entire stack of resources you described, instead of managing resources individually

04

We can use the Visual Builder to build our intent from scratch (not just view the flow itself).

Megha Naik
https://www.linkedin.com/in/naikmegha

Final thoughts...

- This project took me 50 minutes. Writing documentation took me 30 minutes.

- Delete EVERYTHING at the end! Let's keep this project free :)

- Now that I know how to use Lex, in the future, I'd use it to create a chatbot for my other projects which will require a chatbot in it.

- An area of Lex I'd like to explore further is... e.g. visual editor, connecting your chatbot to an application/database, other types of responses that you could use when setting up an intent (such as acknowledge intent, slot capture success/failure response)

Megha Naik

https://www.linkedin.com/in/naikmegha

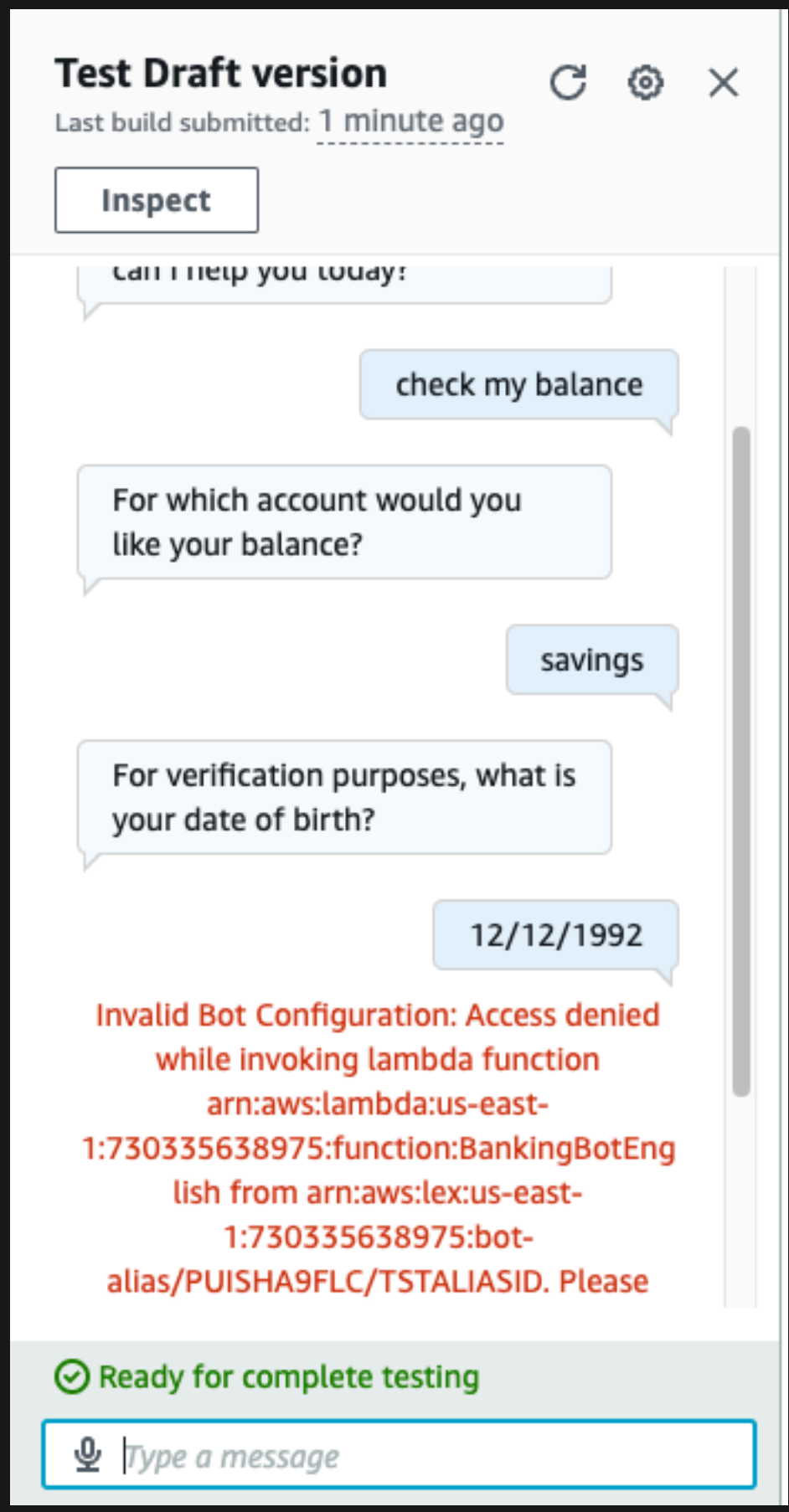
ERROR!👀 oMG!

An error I ran into was...

- An error I ran into was the error message: “Invalid Bot Configuration: Access denied while invoking lambda function”
  - This happened because the permissions in the AWS Lambda function are a little funky, so it's not passing the values back to your chatbot

- ✅ I solved this error by updating the Permission of Lambda. Created a new Resource-based policy statement that gives your chatbot aliases access to your new function.

Screenshot of error here



Megha Naik
https://www.linkedin.com/in/naikmegha

Find this helpful?

Like this post

yes!

Leave a comment

Save for later

Let's connect!

pssst... if you want to get this free project guide and documentation template, **check out NextWork!**

Megha Naik

https://www.linkedin.com/in/naikmegha

Thanks NextWork for the free project guide!

NEXTWORK