**Project 3: Connect Amazon Lex with Lambda**

Watch the magic happen once you connect Lex with Lambda!

This is project 3 of 5 series AI x AWS projects

In projects ONE and TWO, we have learnt how to:

💬 Define intents

🔀 Provide variations in your bot's responses

🌟 Set up a custom slot type

🧪 Build and test your bot using text and speech

**Key Services:** AWS Lambda, Amazon Lex

**Step 1:** Set up a new Lex Chatbot

Complete all the steps provided in project 1 and project 2

**Step 2:** Create your AWS Lambda function

Now that BankerBot can grab a user's account type and birthday, it's high time that we give the user something back... their bank balance!

Lex doesn't come with the smarts to calculate bank balances on its own, but luckily it doesn't have to.

We'll be using another AWS service, AWS Lambda, to generate a random number on the fly (whenever a user asks for their balance). You can think of Lex as the interface that the user sees and chats with, while Lambda is the calculator that's out of sight but works in the background.

**What is AWS Lambda?**

AWS Lambda is a service that lets you run code in the cloud without needing to manage any computers/servers - Lambda will manage them for you.

Lambda runs your code only when needed and scales automatically, from a few requests per day to thousands per second - all you need to do is supply your code in one of the languages that Lambda supports.

* Head to Lambda in your AWS Management Console.
* Click Create function.
* Choose Author from scratch.

A screenshot of a computer

AI-generated content may be incorrect.

* Function name - BankingBotEnglish
* Runtime - Python 3.12, or a later version of Python3 if v3.12 is not available.
* Select **Create function**.

A screenshot of a computer

AI-generated content may be incorrect.

* Scroll down to the **code** section.
* Double-click on **lambda\_function.py** on the left-hand file browser.
* Download the following source code file:   
  ⬇️ [BankingBotEnglish NextWork.py](https://storage.googleapis.com/nextwork_course_resources/courses/aws/AWS%20Project%20People%20projects/Project%3A%20Create%20a%20chatbot%20with%20Amazon%20Lex%20/BankingBotEnglish%20NextWork.py)

What’s in the code file?

This Python script helps your chatbot give users quick answers about their account balances.

When someone asks about their account balance to your chatbot, Lex will ask your Lambda function to run this code, which will pick a random number to pretend it's the balance.

Lambda will pass this random number to Lex, who will then push the bank balance figure to the user through your chatbot.

* Copy the downloaded code, and paste it in your text editor - it should completely replace any placeholder code that was in the example code fragment!

A screenshot of a computer

AI-generated content may be incorrect.

* Choose **Deploy**, and your Lambda function is now all set up!

**Step 3:** Connect AWS Lambda with Amazon Lex

Lambda is totally prepared now to start generating users' bank account balances, but... it doesn't actually know when to do it, or which intent to do it for yet.

We need Lex and Lambda to join forces so that when a user asks for a bank account balance, Lex can trigger Lambda to calculate a number right away.

* Head back to your **Amazon Lex** console.
* Select **BankerBot**.

A screenshot of a computer

AI-generated content may be incorrect.

* On the left-hand menu, choose **Aliases**.

**What are aliases, why are we using them?**

Think of an alias in Amazon Lex as a pointer for a specific version of your bot.

So when you're connecting Lex with other AWS services or your custom applications, those external resources will connect to an alias, which will point to the specific version of your bot that you want to use.

Now, instead of always updating your apps to connect to the newest version of the bot, you can just update the alias to point to that new version. All your apps will automatically start using the updated bot without needing any changes on their end - this saves developers a TON of time and reduces the risk of errors!

A screenshot of a computer

AI-generated content may be incorrect.

* Choose the default **TestBotAlias**.

**What is TestBotAlias?**

TestBotAlias is a default version of your bot that's made for testing or development.

This is the playground version of your bot that you'll use to make sure everything works smoothly before rolling out changes!

A screenshot of a computer

AI-generated content may be incorrect.

* From the **Languages** panel, select **English (US).**
* Ooo how perfect - a Lambda function panel pops up. This panel lets you associate a Lambda function to this TestBotAlias version of your bot.

A screen shot of a message

AI-generated content may be incorrect.

* For **Source**, choose your Lambda function **BankingBotEnglish.**
* Leave the Lambda function version or alias field at the default **$LATEST.**

A screenshot of a computer

AI-generated content may be incorrect.

**What does $LATEST mean?**

Using the **$LATEST** version means you're directing your alias to always use the most up to date version of this Lambda function. This setup is a time saver that lets you immediately test any changes in your function.

* Choose **Save.**

**Step 4: Connect your CheckBalance intent with your Lambda function**

The Lambda function is now connected with BankerBot ✅, but Lex still doesn't know *which* intent inside BankerBot will actually use the Lambda function.

Is it the FallbackIntent? Nope.

How about the WelcomeIntent? Not quite.

Time to make a direct connection between CheckBalance and the Lambda function!

* Navigate to your **CheckBalance** intent.

A screenshot of a computer

AI-generated content may be incorrect.

* Scroll down to **Fulfilment** panel.

What is fulfilment?

In Amazon Lex, **fulfilment** means completing the intent.

With your BankingBot, after a user tells your bot:

1. The account they want to check, and
2. Their birthday for verification

The bot has all the information it needs and moves to fulfilment. This is where it will use the Lambda function to get the account balance and pass it back to the user.

A screenshot of a computer

AI-generated content may be incorrect.

* Expand the On successful fulfilment bubble.
* Choose Advanced options.
* Under the Fulfilment Lambda code hook panel, check the checkbox next to Use a Lambda function for fulfilment.

A screenshot of a computer

AI-generated content may be incorrect.

**What are code hooks?**

Code hooks help you connect your chatbot to custom Lambda functions for doing specific tasks during a conversation.

They're used to handle more complex actions that the basic chatbot setup can't do on its own, like checking data from a database or making decisions based on past conversations.

Essentially, code hooks make your chatbot smarter and more useful by allowing it to perform these extra steps seamlessly during chats.

* Choose **Update options**.
* Choose **Save intent**.
* Choose **Build**.
* Choose **Test**.
* Ask for the balance of any of your accounts - your bot should now be able to return (random) bank balance figures!

A screenshot of a chat

AI-generated content may be incorrect.