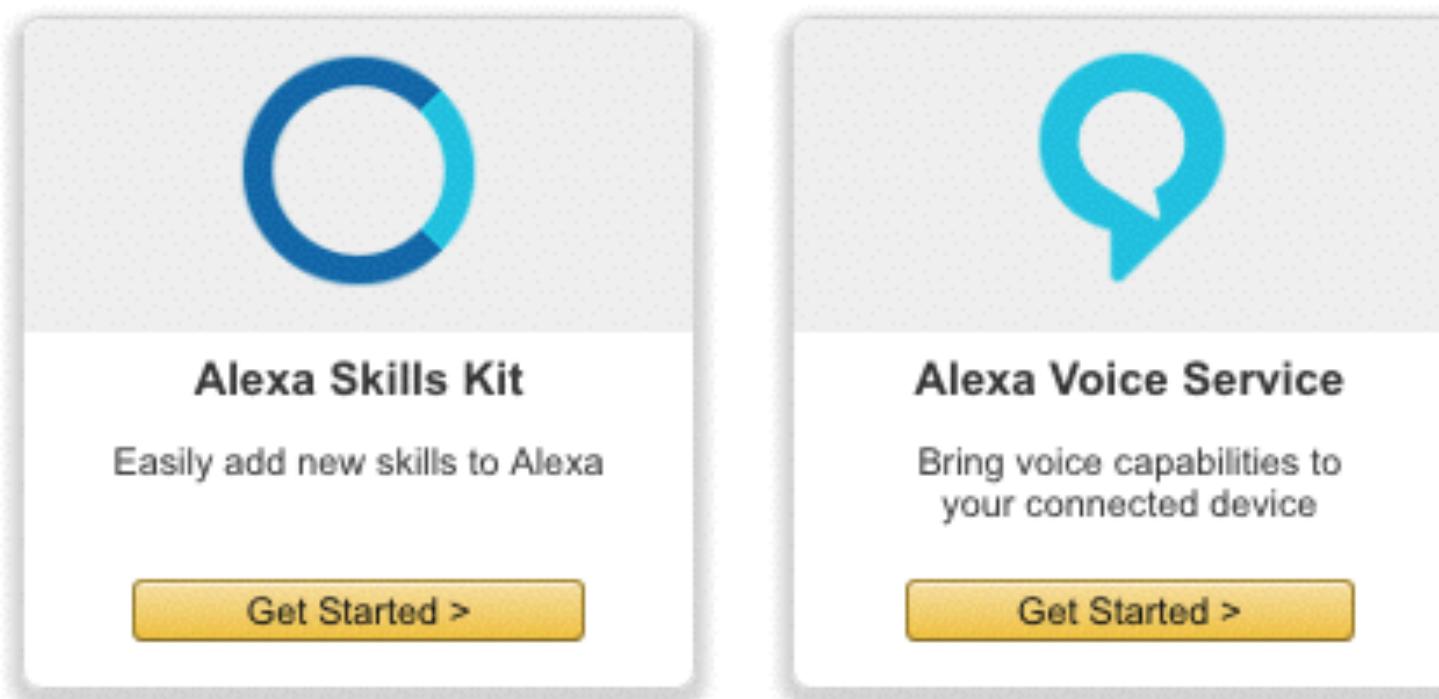
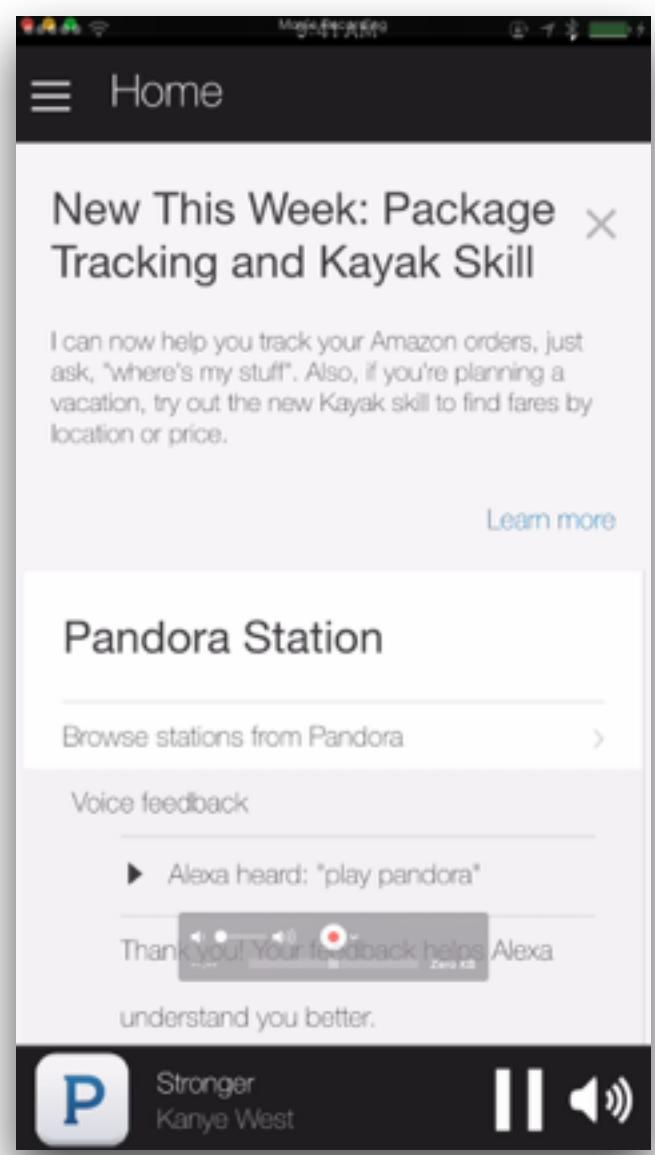


# Famous Alexas of our time

Alexa and Python  
mempy, May 16, 2016

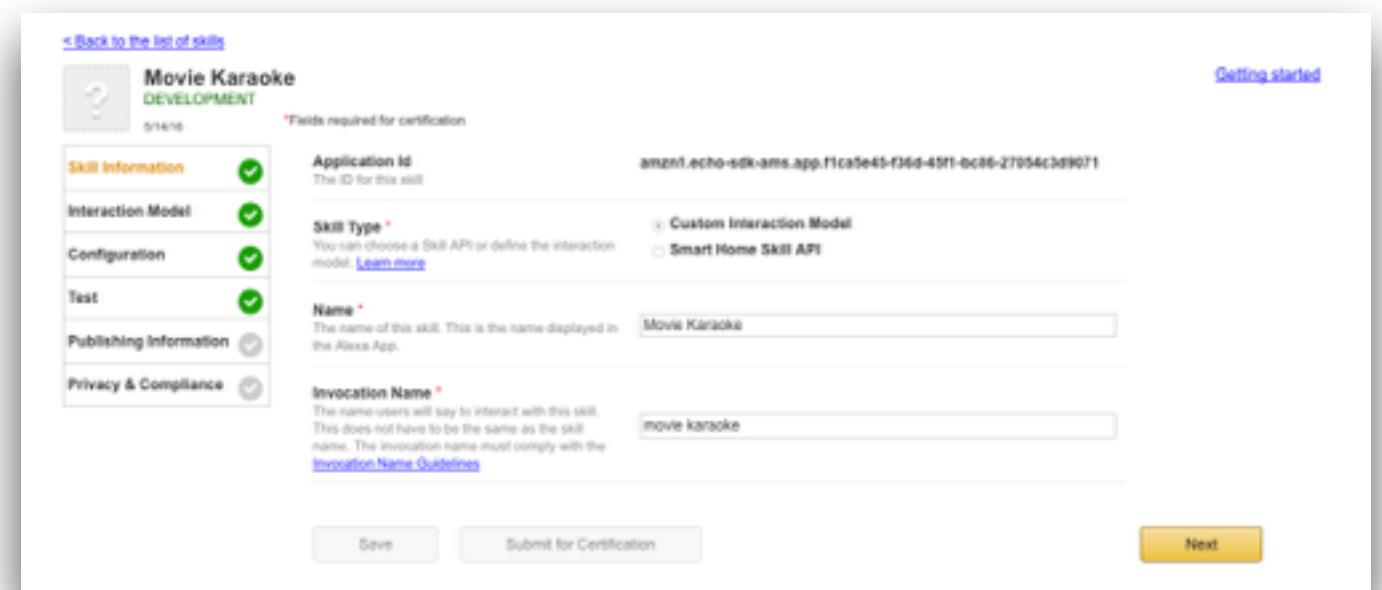
# Alexa and the Echo



# The 3 things

## Alexa Skill

[developer.amazon.com](https://developer.amazon.com)



## Device

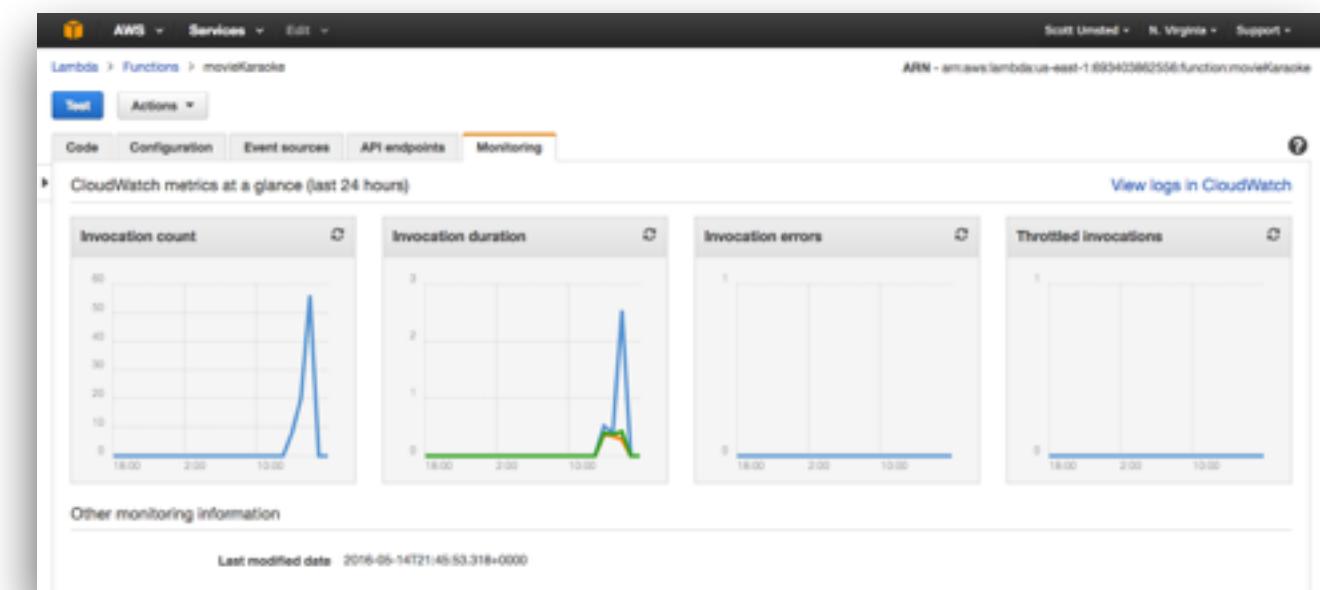
Alexa supported device  
Echo, Tap, Dot, Triby  
Raspberry Pi see Github



A skill is your app  
Skills start with an invocation name  
Utterances are templated phrases  
    Utterances map to intents  
    Intents are your skill's base functions  
Slots are the parameters passed to Intents  
Intent events are passed to your lambda

## Lambda Instance

[aws.amazon.com](https://aws.amazon.com)



A lambda is a micro PaaS  
Returns a natural language response  
Secured and tied to specific source  
    Event driven and scales  
    Java 8, Node 4, Python 2.7  
    Code in web editor or .zip  
Can include libraries like requests

# Skills

Invocation

Ask 'Movie Karaoke'

Utterances

**rateCityToCityIntent** how much is it to ship from {**originationCity**} to {**destinationCity**}  
**rateCityToCityIntent** how much does it cost to ship from {**originationCity**} to {**destinationCity**}

Intents

```
{  
  "intents": [  
    {  
      "intent": "rateCityToCityIntent",  
      "slots": [  
        {  
          "name": "originationCity",  
          "type": "AMAZON.US_CITY"  
        },  
        {  
          "name": "destinationCity",  
          "type": "AMAZON.US_CITY"  
        }  
      ]  
    }  
  ]  
}
```

Slots

Slot Types

AMAZON.DATE  
AMAZON.DURATION  
AMAZON.FOUR\_DIGIT\_NUMBER  
AMAZON.NUMBER  
AMAZON.TIME  
AMAZON.US\_CITY  
AMAZON.US\_FIRST\_NAME  
AMAZON.US\_STATE  
Custom - Named list of values

Lambda

aws:2hg435jhghjgsdf



# “How much is it to ship from Collierville to Nashville?”

Request from Alexa

```
{  
  "session": {  
    "sessionId": "SessionId.3a1035",  
    "application": {  
      "applicationId": "amzn1.echo-sdk-e8"  
    },  
    "attributes": {},  
    "user": {  
      "userId": "amzn1.ask.account.AFP3ZWPOSY",  
      "accessToken": "b0df04b2a76243689f0f6a"  
    },  
    "new": false  
  },  
  "request": {  
    "type": "IntentRequest",  
    "requestId": "EdwRequestId.32e51095-e79d",  
    "timestamp": "2016-05-14T23:29:04Z",  
    "intent": {  
      "name": "rateCityToCityIntent",  
      "slots": {  
        "destinationCity": {  
          "name": "destinationCity",  
          "value": "Nashville"  
        },  
        "originationCity": {  
          "name": "originationCity",  
          "value": "Collierville"  
        }  
      }  
    },  
    "locale": "en-US"  
  },  
  "version": "1.0"  
}
```

Amazon Lambda



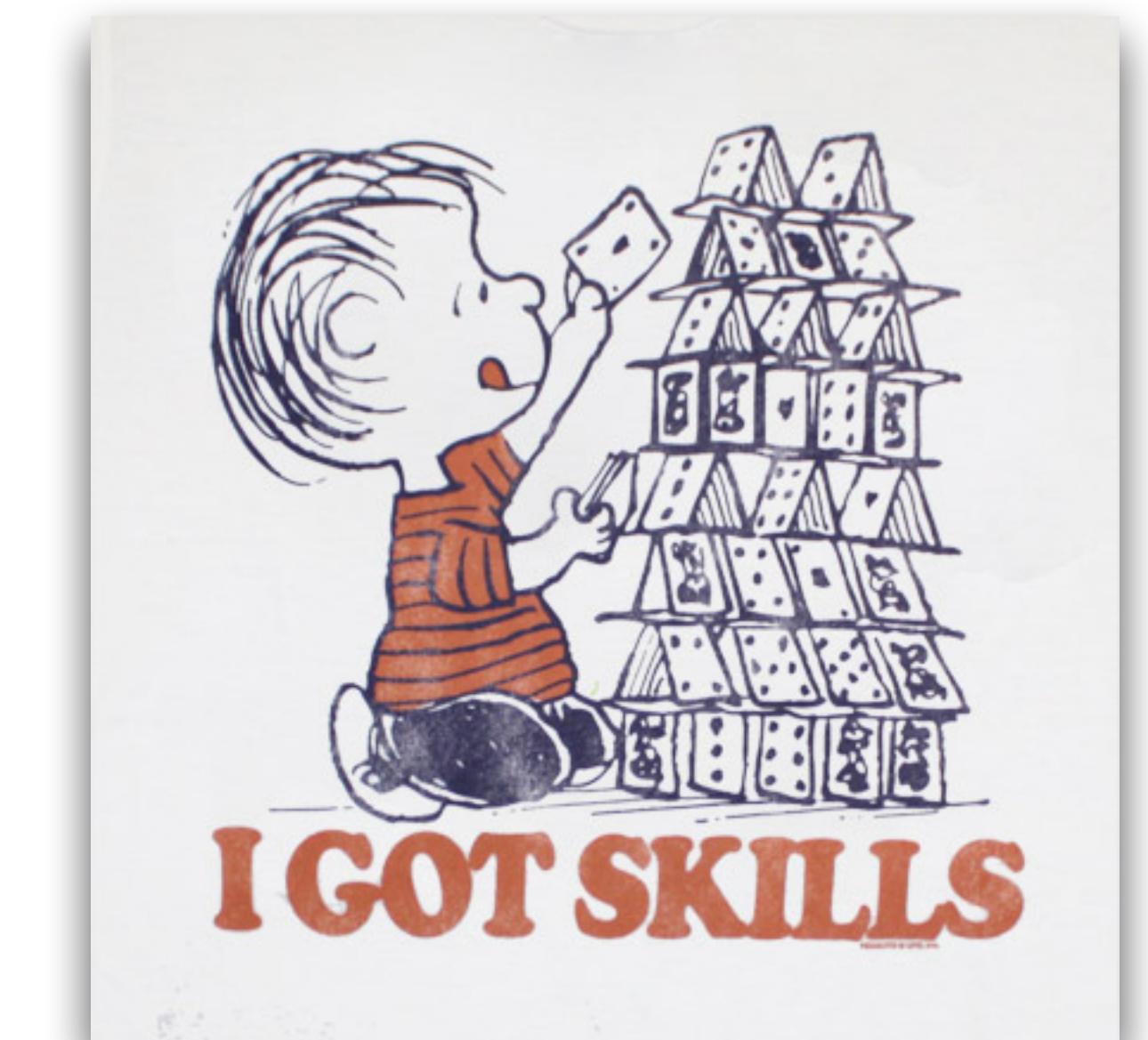
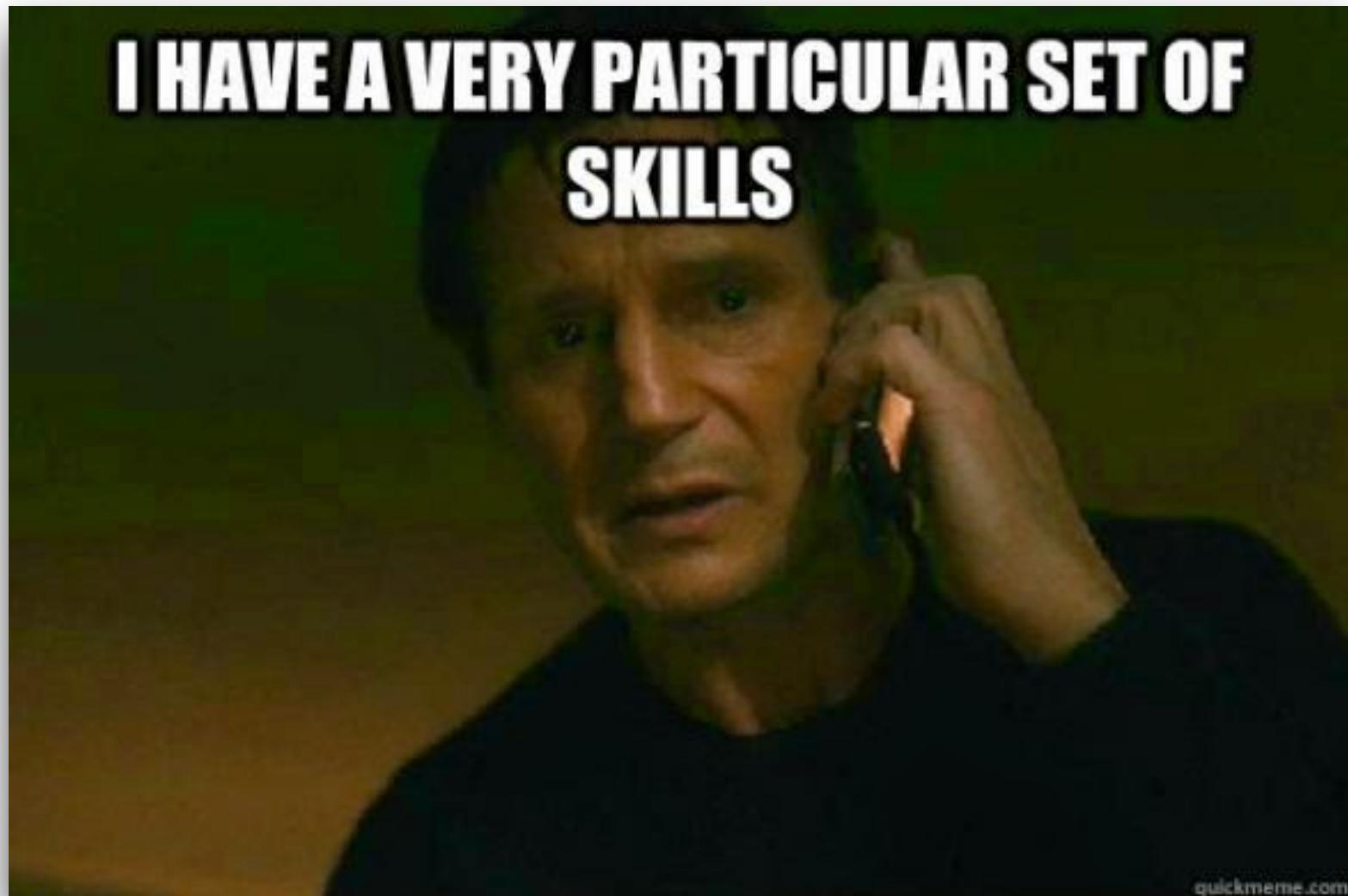
Response from Lambda

```
{  
  "version": "1.0",  
  "response": {  
    "outputSpeech": {  
      "type": "PlainText",  
      "text": "FedEx found the following shipment rates from Collierville, Tennessee to Nashville, Tennessee. Standard Overnight costs 67 dollars and 98 cents. Fedex 2 Day costs 41 dollars and 72 cents. Fedex Ground costs 10 dollars and 23 cents."  
    },  
    "card": {  
      "content": "SessionSpeechlet - FedEx found the following shipment rates from Collierville, Tennessee to Nashville, Tennessee. Standard Overnight costs 67 dollars and 98 cents. Fedex 2 Day costs 41 dollars and 72 cents. Fedex Ground costs 10 dollars and 23 cents.",  
      "title": "SessionSpeechlet - Shipping Rate",  
      "type": "Simple"  
    },  
    "reprompt": {  
      "outputSpeech": {  
        "type": "PlainText"  
      }  
    },  
    "shouldEndSession": false  
  },  
  "sessionAttributes": {}  
}
```



# Create a skill

- 1.Create Lambda
- 2.Create Site
- 3.Move code to Lambda



# Step 1 - Create Lambda

[aws.amazon.com](https://aws.amazon.com)

The screenshot shows the AWS Services menu with the Lambda icon highlighted by a red arrow. The main content area displays a grid of AWS services, with the Lambda service icon highlighted by a red arrow.

The screenshot shows the Lambda Functions page. It displays a message "You have 3 Lambda functions" with a red arrow pointing to it. A "Create a Lambda function" button is visible. The URL in the browser bar is <https://console.aws.amazon.com/lambda/functions>.

## Select blueprint

Blueprints are sample configurations of event sources and Lambda functions. Choose a blueprint that best aligns with your desired scenario and customize as needed, or skip this step if you want to author a Lambda function and configure an event source separately. Except where otherwise noted, blueprints are licensed under CC0.

The screenshot shows a search results page for "alex". Three blueprints are listed:

- alex-skill-kit-color-expert**: Demonstrates a basic skill built with the Amazon Alexa Skills Kit. Language: nodejs · alexa
- alex-skill-kit-color-expert-p...**: Demonstrates a basic skill built with the Amazon Alexa Skills Kit. Language: python2.7 · alexa
- alex-smart-home-skill-adapter**: Provides the basic framework for a skill adapter for a smart home skill. Languages: nodejs · iot · smart-home · alexa · light



# more lambda stuff

**Configure event sources**

Choose the appropriate event source for your Lambda function.

Configure your Lambda function to respond to events from the event sources listed below. You may also use the AWS mobile SDK for [Android](#) and [iOS](#).

**Event source type**  i

**Configure function**

A Lambda function consists of the custom code you want to execute. [Learn more](#) about Lambda functions.

**Name\***

**Description**

**Runtime\***  i

**Lambda function code**

Provide the code for your function. Use the editor if your code does not require custom libraries (other than boto3). If you need custom libraries, you can upload your code and libraries as a .ZIP file.

**Code entry type**  Edit code inline  Upload a .ZIP file  Upload a file from Amazon S3

```
1 """
2 This sample demonstrates a simple skill built with the Amazon Alexa Skills Kit.
3 The Intent Schema, Custom Slots, and Sample Utterances for this skill, as well
4 as testing instructions are located at http://amzn.to/1LzFrj6
5
6 For additional samples, visit the Alexa Skills Kit Getting Started guide at
7 http://amzn.to/1LGWslG
8 """
9 """
```

We'll add our code here later



**Handler and role**

**Handler\***  i

**Role\***  i

Ensure that popups are enabled to create a new role. Suggested role: Basic execution

Weird Popup

**Role Summary** ?

**Role Description** Lambda execution role permissions

**IAM Role**  i

**Policy Name**  i

**Handler\***  i

**Role\***  i

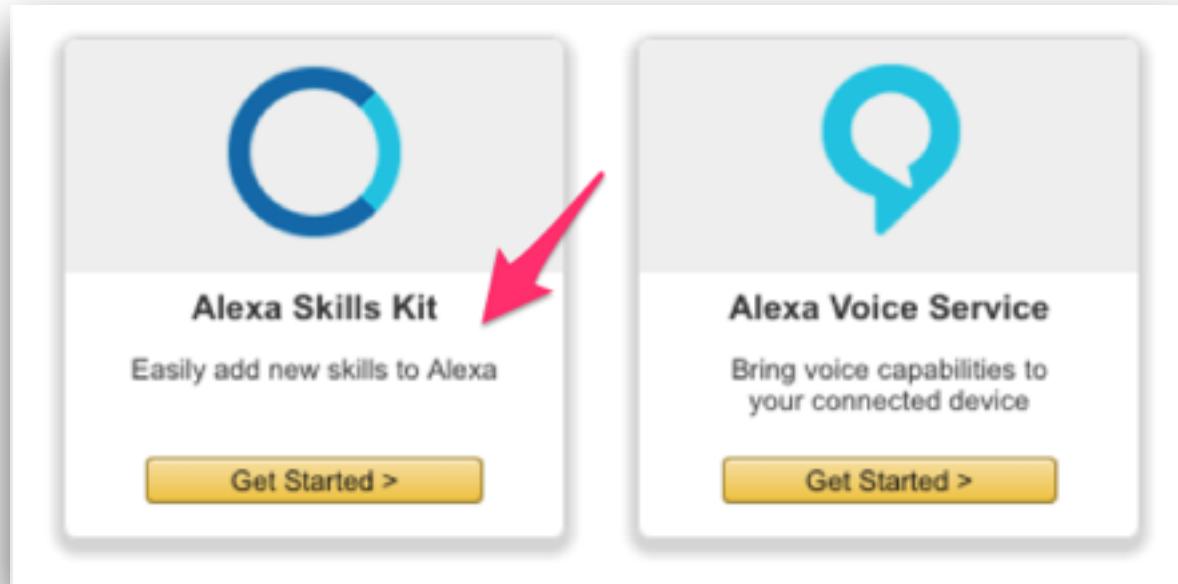
**Create function**

Scott Umsted ▼

ARN - arn:aws:lambda:us-east-1:693403

We'll need this for the skill

# Step 2 - Create Skill



### Create a New Alexa Skill

**DEVELOPMENT**

\*Fields required for certification

Skill Information	<input checked="" type="checkbox"/>
Interaction Model	<input checked="" type="checkbox"/>
Configuration	<input checked="" type="checkbox"/>
Test	<input checked="" type="checkbox"/>
Publishing Information	<input checked="" type="checkbox"/>
Privacy & Compliance	<input checked="" type="checkbox"/>

**Skill Type \***  
You can choose a Skill API or define the interaction model. [Learn more](#)

Custom Interaction Model  
 Smart Home Skill API

**Name \***  
The name of this skill. This is the name displayed in the Alexa App.  
Movie Karaoke

**Invocation Name \***  
The name users will say to interact with this skill. This does not have to be the same as the skill name. The invocation name must comply with the [Invocation Name Guidelines](#).  
movie karaoke

**Save**

**Movie Karaoke**  
**DEVELOPMENT**  
5/14/16

\*Fields required for certification

**Intent Schema\***  
The schema of user intents in JSON format. For more information, see [Intent Schema](#). Also see [built-in slots](#) and [built-in intents](#).

```

28     "intent": "holyGrailFourIntent",
29     "slots": [
30       {
31         "name": "nextNumber",
32         "type": "AMAZON.NUMBER"
33       }
34     ],
35   },
36   {
37     "intent": "imFinishedIntent"
38   },
39   {
40     "intent": "AMAZON.HelpIntent"
41   }
42 }
43

```

**Custom Slot Types**  
Custom slot types to be referenced by the Intent Schema and Sample Utterances  
For general information about custom slots, see [Custom Slot Types](#).  
Example: TOPPINGS - cheese | onions | ham (note: newlines displayed as | for brevity)

**Sample Utterances\***  
Phrases end users say to interact with the skill. For better results, provide as many samples as you can. Note that Example Phrases on the Description tab.  
For more information, see [Sample Utterances](#).

```

1 startPointBreakIntent Start Point Break
2 startPointBreakIntent Point Break
3 pointBreakOneIntent Too bad. You finally get your waves and it's totally
4 pointBreakTwoIntent You gotta go down. You crossed the line and people

```

**Movie Karaoke**  
**DEVELOPMENT**  
5/14/16

\*Fields required for certification

Skill Information	<input checked="" type="checkbox"/>
Interaction Model	<input checked="" type="checkbox"/>
Configuration	<input checked="" type="checkbox"/>
Test	<input checked="" type="checkbox"/>
Publishing Information	<input checked="" type="checkbox"/>
Privacy & Compliance	<input checked="" type="checkbox"/>

**Endpoint \***  
The URL for the service endpoint, e.g. <https://myskill.ishere.com/somepath>, or the Lambda ARN, [More info about AWS Lambda](#) [How to integrate AWS Lambda with Alexa](#)

HTTPS  Lambda ARN (Amazon)  
arn:aws:lambda:us-east-1:69340386

**Account Linking**  
Do you allow users to create an account or link to an existing account with you? [Learn more](#)

Yes  No

**Save** **Submit for Certification**

# Where we test our skill

once we finish writing our code

Movie Karaoke  
DEVELOPMENT

5/14/16

Skill Information

Interaction Model

Configuration

Test

Publishing Information

Privacy & Compliance

*i* Start testing this skill  This skill is enabled for testing on your account.

Once you have completed testing on your device, please complete the Description and Publishing Information tab, then submit the skill for certification.

If it passes Amazon's testing and certification process, it will become available to Alexa end users.

Try this on your Echo: Alexa ask movie karaoke

**Voice Simulator**

Hear how Alexa will speak a response entered in plain text or SSML. [Learn more about supported SSML tags.](#)

For example: Here is a word spelled out: <say-as interpret-as="spell-out">hello</say-as>.

Here is a word spelled out: <say-as interpret-as="spell-out">hello</say-as>.

**Service Simulator**

Use Service Simulator to test your lambda function.

Text  Json

Enter Utterance \*

**Lambda Request** **Lambda Response**

1  1



# Step 3 Our Code

## step 2-steal clone Amazon's code

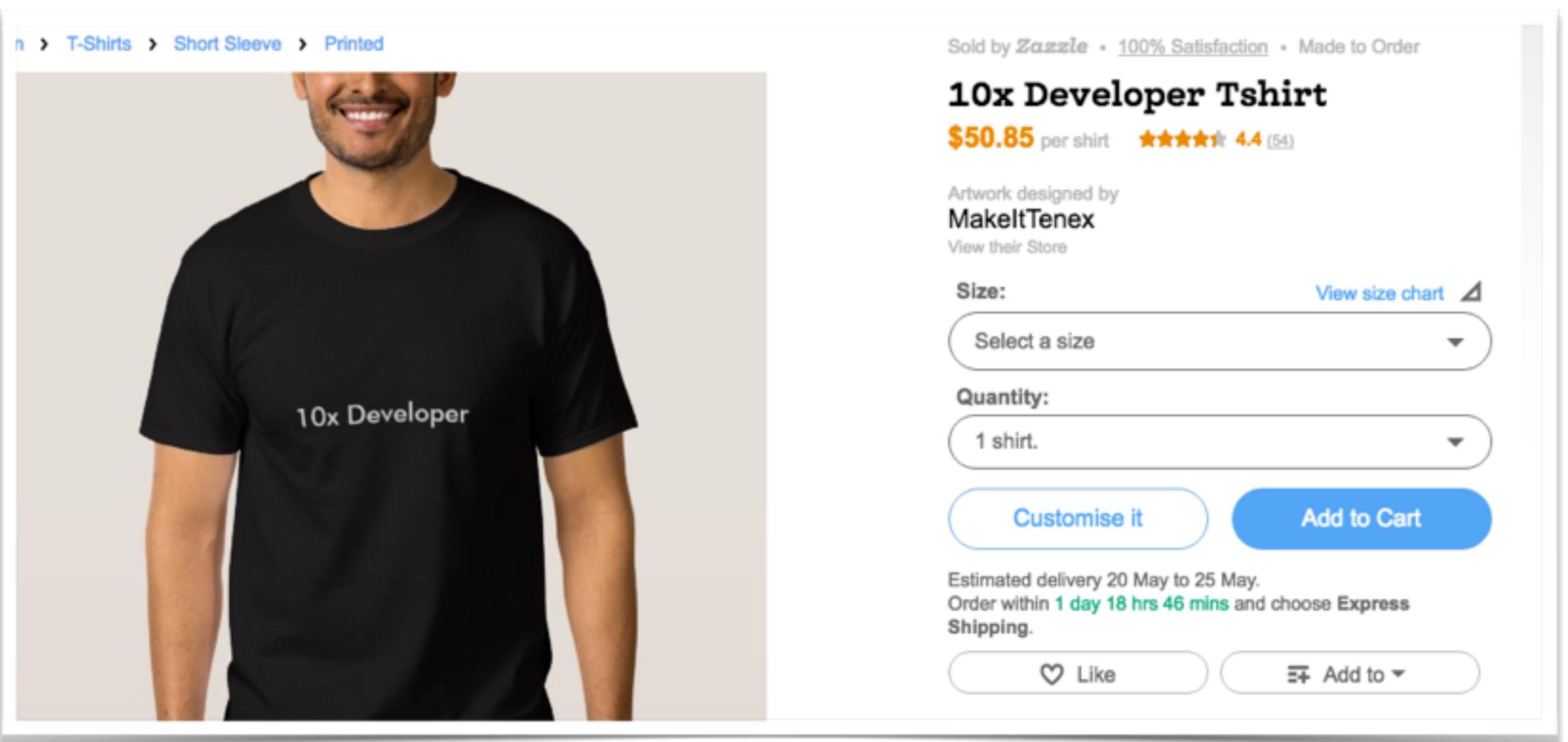
```
def lambda_handler(event, context):
    """ Route the incoming request based on type (LaunchRequest, IntentRequest,
    etc.) The JSON body of the request is provided in the event parameter.
    """
    print("event.session.application.applicationId=" +
          event['session']['application']['applicationId'])

    """
    Uncomment this if statement and populate with your skill's application ID to
    prevent someone else from configuring a skill that sends requests to this
    function.
    """
    # if (event['session']['application']['applicationId'] !=
    #     "amzn1.echo-sdk-ams.app.unique-value-here"):
    #     raise ValueError("Invalid Application ID")

    if event['session']['new']:
        on_session_started({'requestId': event['request']['requestId'],
                           'session': event['session']})

    if event['request']['type'] == "LaunchRequest":
        return on_launch(event['request'], event['session'])
    elif event['request']['type'] == "IntentRequest":
        return on_intent(event['request'], event['session'])
    elif event['request']['type'] == "SessionEndedRequest":
        return on_session_ended(event['request'], event['session'])
```

## step 1 put on favorite coding t-shirt



```
def on_intent(intent_request, session):
    """ Called when the user specifies an intent for this skill """
    print("on_intent requestId=" + intent_request['requestId'] +
         ", sessionId=" + session['sessionId'])

    intent = intent_request['intent']
    intent_name = intent_request['intent']['name']

    # Dispatch to your skill's intent handlers
    if intent_name in ["startPointBreakIntent"]:
        return start_point_break(intent, session)
    elif intent_name in ["pointBreakOneIntent"]:
        return holy_grail_4(intent, session)
    elif intent_name in ["imFinishedIntent"]:
        return i_am_finished(intent, session)
    elif intent_name == "AMAZON.HelpIntent":
        return get_welcome_response()
    elif intent_name == "AMAZON.CancelIntent" or intent_name == "AMAZON.StopIntent":
        return handle_session_end_request()
    else:
        raise ValueError("Invalid intent")
```

# More with “our” lambda code

```
def point_break_1(intent, session):
    title = "Point Break"
    should_end_session = False
    session_attributes = {}
    reprompt_text = None

    speech_output = "Just waiting for my set."

    return build_response(session_attributes, build_speechlet_response(
        title, speech_output, reprompt_text, should_end_session))

def point_break_2(intent, session):
    title = "Point Break"
    should_end_session = False
    session_attributes = {}
    reprompt_text = None

    speech_output = "Yeah, it went bad, went real bad."

    return build_response(session_attributes, build_speechlet_response(
        title, speech_output, reprompt_text, should_end_session))

def holy_grail_4(intent, session):
    title = "Holy Grail"
    should_end_session = False
    session_attributes = {}
    reprompt_text = None

    if get_slot(intent, 'nextNumber', '3') == '3':
        speech_output = "Yes, Three! Boom!"
    else:
        speech_output = "Three, sir!"

    return build_response(session_attributes, build_speechlet_response(
        title, speech_output, reprompt_text, should_end_session))
```

```
def get_slot(intent, key, default=None):
    try:
        return intent['slots'][key]['value']
    except:
        return default
```

```
def build_speechlet_response(title, output, reprompt_text, should_end_session):
    return {
        'outputSpeech': {
            'type': 'PlainText',
            'text': output
        },
        'card': {
            'type': 'Simple',
            'title': 'SessionSpeechlet - ' + title,
            'content': 'SessionSpeechlet - ' + output
        },
        'reprompt': {
            'outputSpeech': {
                'type': 'PlainText',
                'text': reprompt_text
            }
        },
        'shouldEndSession': should_end_session
    }
```

```
def build_response(session_attributes, speechlet_response):
    return {
        'version': '1.0',
        'sessionAttributes': session_attributes,
        'response': speechlet_response
    }
```



# Paste into lambda and test

Lambda on [aws.amazon.com](https://aws.amazon.com)

Lambda > Functions > someMovie

Save Save and test Actions ▾

Congratulations! Your Lambda function "someMovie" has been successfully created and configured with the following settings:

Code Configuration Event sources API endpoints Monitoring

Code entry type: Edit code inline

```
1 from __future__ import print_function
2
3
4 def lambda_handler(event, context):
5     """ Route the incoming request based on type (LaunchRequest, IntentRequest,
6     etc.) The JSON body of the request is provided in the event parameter.
7     """
8     print("event.session.application.applicationId=" +
9         event['session']['application']['applicationId'])
10    """
11
12    """
13
```

Test on [developer.amazon.com](https://developer.amazon.com)

Service Simulator

Use Service Simulator to test your lambda function.

Text Json

Enter Utterance \*

start point break

Ask Movie Karaoke Reset

Lambda Request

```
1 {
2     "session": {
3         "sessionId": "SessionId.0789ac47-6875-432d-87
4             "application": {
5                 "applicationId": "amzn1.echo-sdk-ams.app.fd
6             },
7             "user": {
8                 "userId": "amzn1.ask.account.AFP3ZWPOS2BGJR
9             },
10            "new": true
11        },
12        "request": {
13            "type": "IntentRequest",
14            "requestId": "EdwRequestId.c6c9cf01-60ec-4e0f
15            "timestamp": "2016-05-15T00:26:07Z",
16            "intent": {
17                "name": "AMAZON.CancelIntent"
18            }
19        }
20    }
21}
```

Lambda Response

```
1 {
2     "version": "1.0",
3     "response": {
4         "outputSpeech": {
5             "type": "PlainText",
6             "text": "Special Agent Utah."
7         },
8         "card": {
9             "content": "SessionSpeechlet - Special Ag
10             "title": "SessionSpeechlet - Point Break"
11             "type": "Simple"
12         },
13         "reprompt": {
14             "outputSpeech": {
15                 "type": "PlainText"
16             }
17         }
18     }
19 }
```

# make something useful



Various Alexa Devices  
Assistant + BT Speaker  
Approximately 3M sold  
Goal of 2 second response



Enable skills  
Identify users  
Translate utterances  
Invoke intents  
Pass slots



Lightweight PaaS  
Event based service  
Coupled to Alexa  
Intent based routes  
Easily scales



PaaS for web and db  
Services and web  
Authorization provider  
User profile, packages list  
Translate external services

some shipping  
company

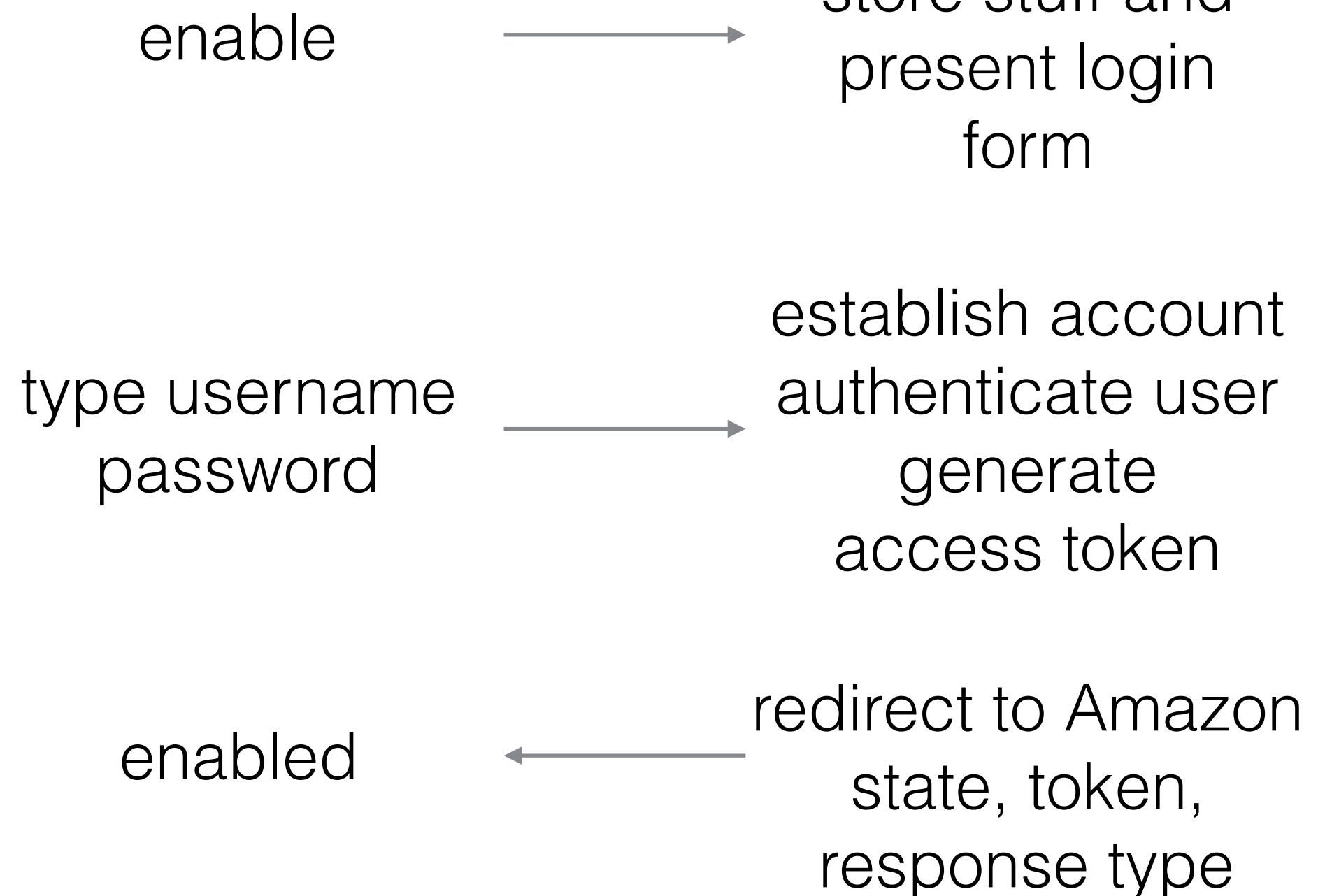


Favorite Track  
and Rate APIs



Splash page  
DNS routing of MX records  
Route track request emails  
Email based linking  
Email activation process

# Enabling users while providing unique and individual experiences



```
{  
  "session": {  
    "sessionId": "SessionId.3a1035",  
    "application": {  
      "applicationId": "amzn1.echo-sdk-e8"  
    },  
    "attributes": {},  
    "user": {  
      "userId": "amzn1.ask.account.AFP3ZWPOSY",  
      "accessToken": "b0df04b2a76243689f0f6a"  
    },  
    "new": false  
  },  
  "request": {  
    "type": "IntentRequest",  
    "requestId": "EdwRequestId.32e51095-e79d",  
    "timestamp": "2016-05-14T23:29:04Z",  
    "intent": {  
      "name": "rateCityToCityIntent",  
      "slots": {  
        "destinationCity": {  
          "name": "destinationCity",  
          "value": "Nashville"  
        },  
        "originationCity": {  
          "name": "originationCity",  
          "value": "Collierville"  
        }  
      }  
    },  
    "locale": "en-US"  
  },  
  "version": "1.0"  
}
```