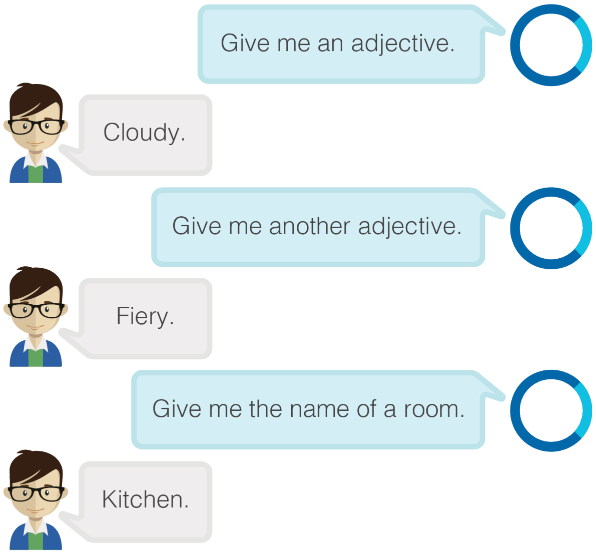
Persistence

Implementing Persistence

Persist a Madlib 1/2



Give me an adjective.

Cloudy.

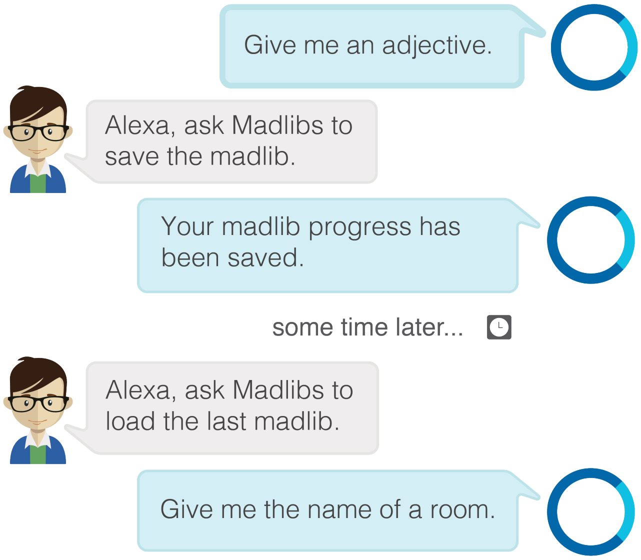
Give me another adjective.

Fiery.

Give me the name of a room.

Kitchen.

Persist a Madlib 2/2



Give me an adjective.

Alexa, ask Madlibs to save the madlib.

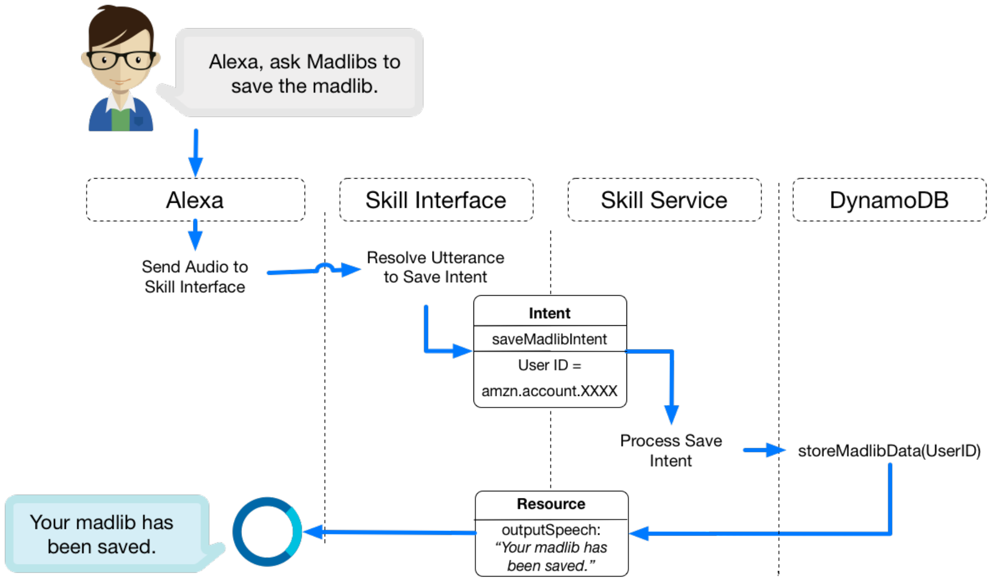
Your madlib progress has been saved.

some time later...

Alexa, ask Madlibs to load the last madlib.

Give me the name of a room.

Saving a Madlib



Alexa, ask Madlibs to save the madlib.

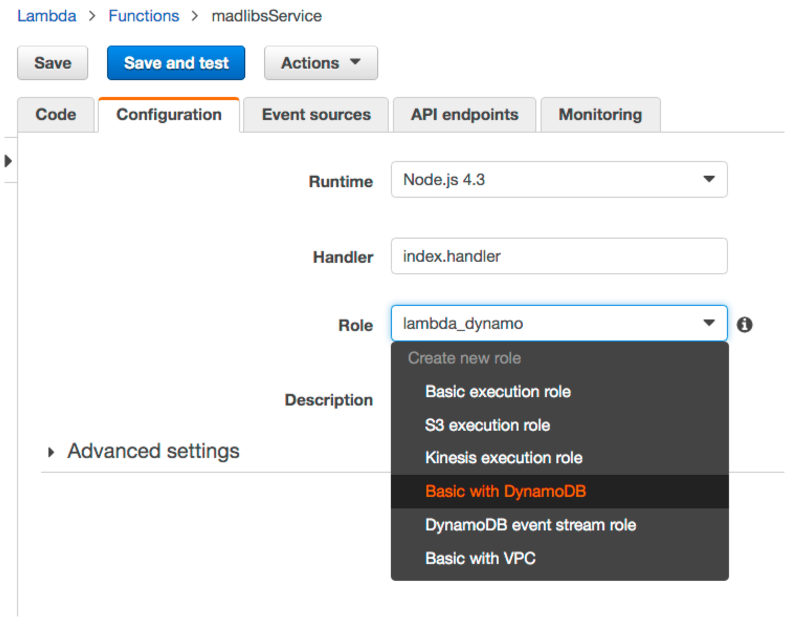
Alexa
Skill Interface
Skill Service
DynamoDB
Send Audio to Skill Interface
Resolve Utterance to Save Intent
Intent
saveMadlibIntent
User ID = amzn.account.XXXX
Process Save Intent
storeMadlibData(UserID)
Resource
outputSpeech: *"Your madlib has been saved."*

Your madlib has been saved.

Introducing DynamoDB

- NoSQL-style database
- No Schema
- Cloud-hosted on AWS
- Lightning Fast!

Enabling DynamoDB on AWS Lambda



Runtime: Node.js 4.3

Handler: index.handler

Role: lambda_dynamo

Description

Building a Database Helper 1/4

```
database_helper.js
var MADLIBS_DATA_TABLE_NAME = 'madlibsData';
var dynasty = require('dynasty')({});

function DatabaseHelper() {}

var madlibTable = function() {
  return dynasty.table(MADLIBS_DATA_TABLE_NAME);
};
```

Building a Database Helper 2/4

database_helper.js

```
DatabaseHelper.prototype.createMadlibsTable = function() {
  return dynasty.describe(MADLIBS_DATA_TABLE_NAME)
    .catch(function(error) {
      console.log('createMadlibsTable::error: ', error);
      return dynasty.create(MADLIBS_DATA_TABLE_NAME, {
        key_schema: {
          hash: ['userId', 'string']
        }
      });
    });
};
```

Building a Database Helper 3/4

index.js

```
skillService.pre = function(request,response,type) {
  databaseHelper.createMadlibsTable();
};
```

Building a Database Helper 4/4

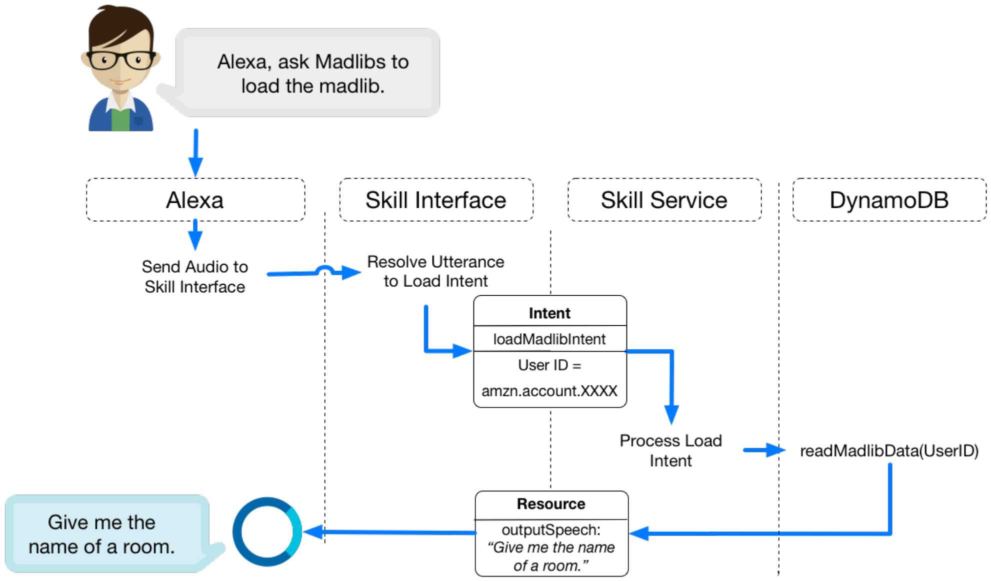
```
database_helper.js
DatabaseHelper.prototype.storeMadlibData = function(userId, madlibData) {
  return madlibTable().insert({
    userId: userId,
    data: madlibData
  })
};

DatabaseHelper.prototype.readMadlibData = function(userId) {
  return madlibTable().find(userId)
    .then(function(result) {
      console.log("got a " + result);
      return result;
    });
    //error handling here
};
```

Saving the Madlib Data

```
skillService.intent('saveMadlibIntent', {
  'utterances': ['{save} {|a|the|my} madlib']
},
  function(request, response) {
    var userId = request.userId;
    var madlibHelper = getMadlibHelperFromRequest(request);
    databaseHelper.storeMadlibData(userId, JSON.stringify(madlibHelper))
    .then(function(result) {
        response.say('Your madlib progress has been saved!');
        response.shouldEndSession(true).send();
    });
    return false;
  }
);
```

Loading the Madlib Data 1/2



Alexa, ask Madlibs to load the madlib.

Alexa

Skill Interface

Skill Service

DynamoDB

Send Audio to Skill Interface

Resolve Utterance to Load Intent

Intent

loadMadlibIntent

User ID = amzn.account.XXXX

Process Load Intent

readMadlibData(UserID)

Resource

outputSpeech: *"Give me the name of a room."*

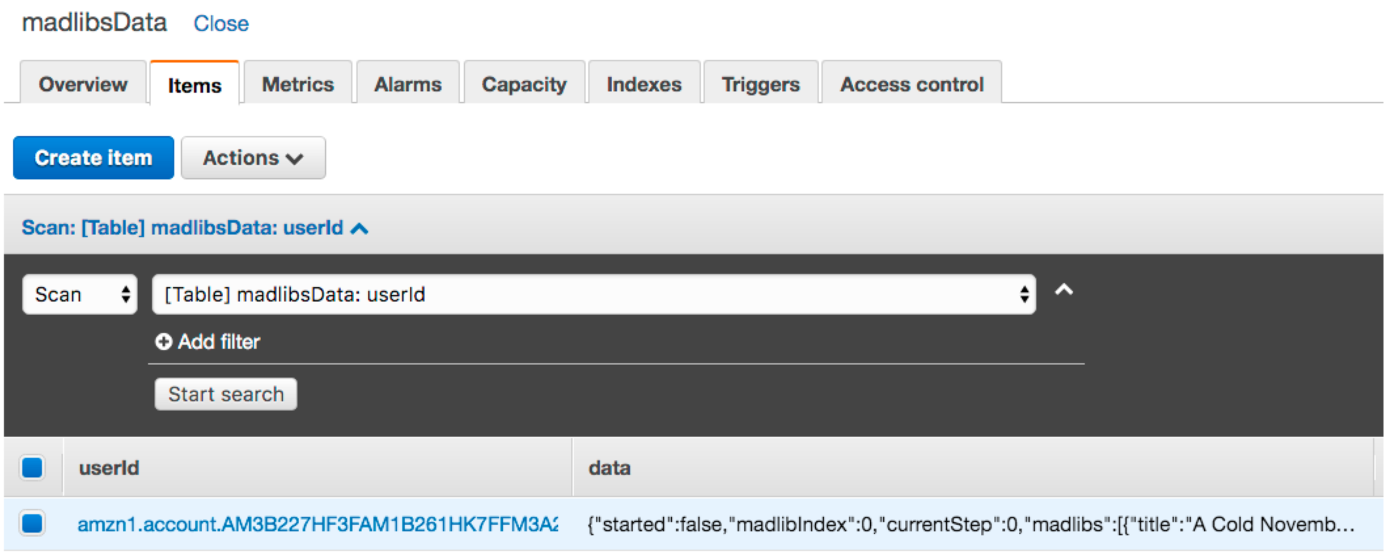
Give me the name of a room.

Loading the Madlib Data 2/2

index.js

```
skillService.intent('loadMadlibIntent', {
    'utterances': ['{load|resume} {|a|the} {|last} madlib']
  },
  function(request, response) {
    var userId = request.userId;
    databaseHelper.readMadlibData(userId).then(function(result) {
      if (result === undefined) {
        return {};
      } else {
        return JSON.parse(result['data']);
      }
    }).then(function(loadedMadLibHelperData) {
      var madlibHelper = new MadlibHelper(loadedMadLibHelperData);
      return madlibIntentFunction(madlibHelper, request, response);
    });
    return false;
  }
);
```

Browsing Data on AWS DynamoDB



■	userId	data
■	amzn1.account.AM3B227HF3FAM1B261HK7FFM3A2	{"started":false,"madlibIndex":0,"currentStep":0,"madlibs":[{"title":"A Cold Novemb...