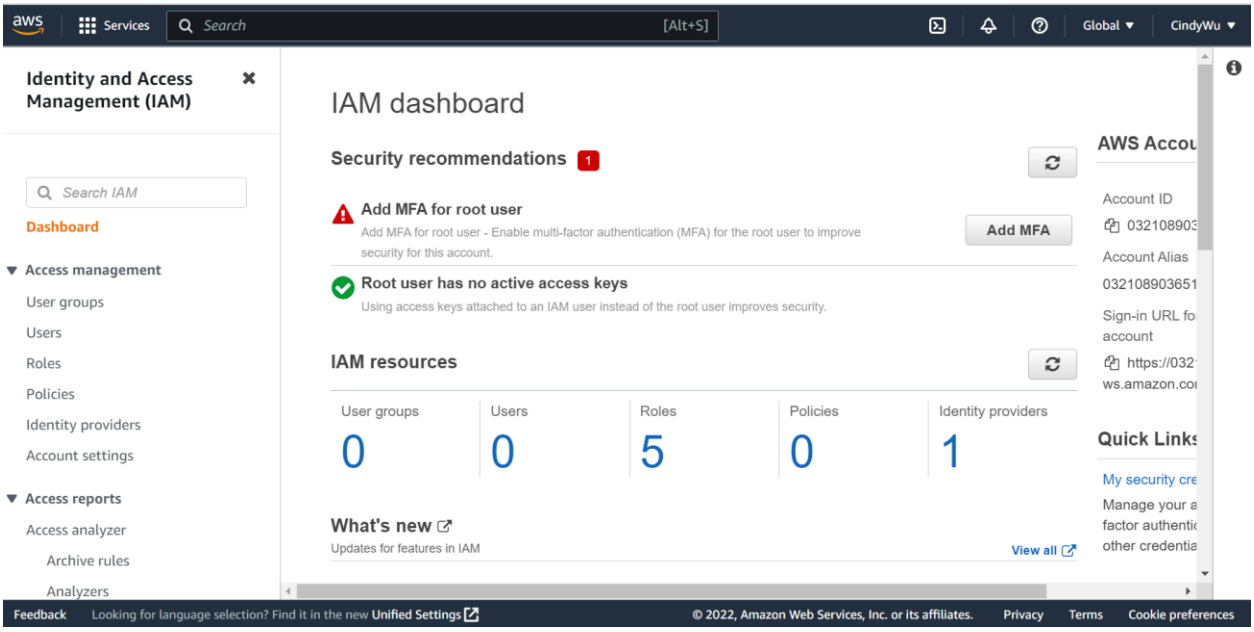
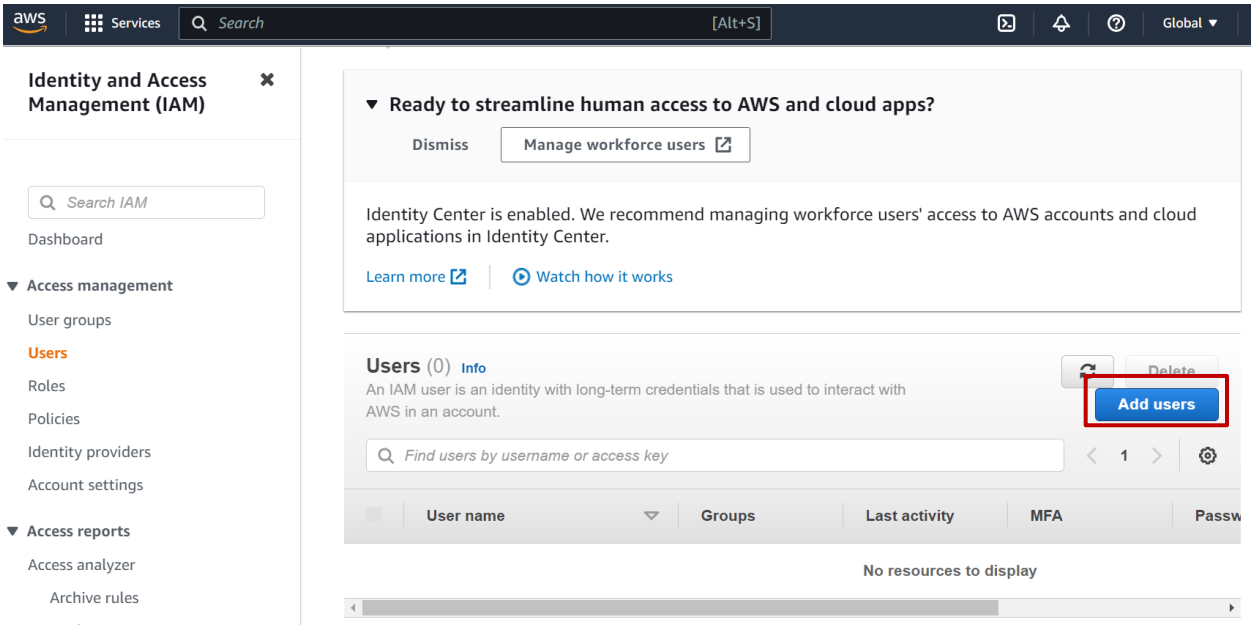
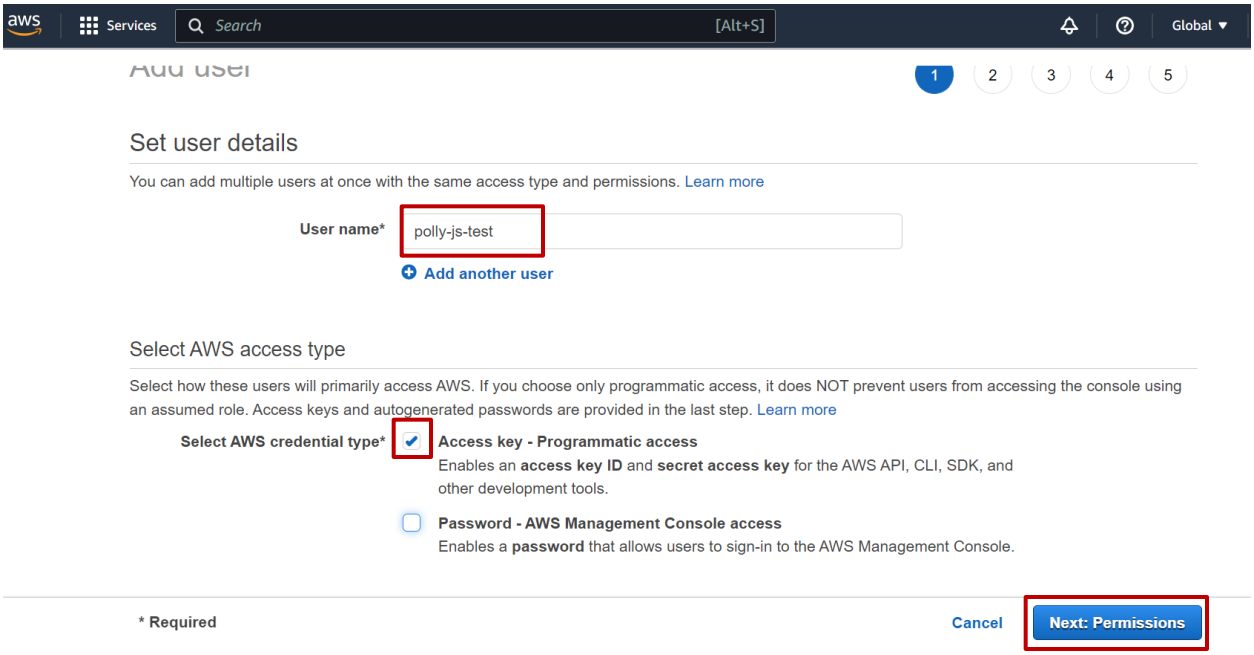
WEEK12 Quiz - Using Amazon Polly to make your sensor speak

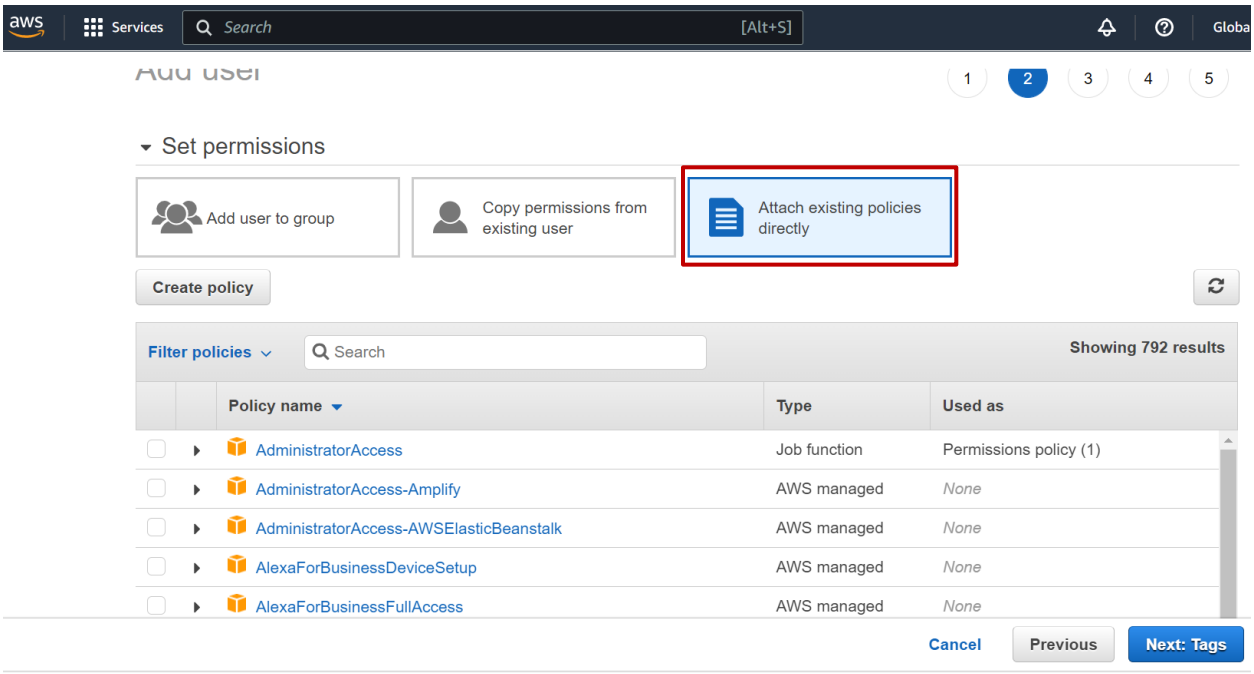
Step1. Access the AWS IAM dashboard on http://console.aws.amazon.com/iam/ and create a user with privilege access for AWS Polly.







Select "Attach existing policies directly", enter "lambda" and "polly" in the search box and select the following permission for user.



aws | Services | Search [Alt+S] | Global

Set permissions

Add user to group | Copy permissions from existing user | Attach existing policies directly

Create policy

Filter policies | polly | Showing 2 results

	Policy name	Type	Used as
☑	AmazonPollyFullAccess	AWS managed	None
☐	AmazonPollyReadOnlyAccess	AWS managed	None

Set permissions boundary

Cancel | Previous | Next: Tags

aws | Services | Search [Alt+S] | Global

☐	AWSDeepLensLambdaFunctionAccessPolicy	AWS managed	None
☐	AWSLambda_FullAccess	AWS managed	None
☐	AWSLambda_ReadOnlyAccess	AWS managed	None
☐	AWSLambdaBasicExecutionRole	AWS managed	None
☐	AWSLambdaDynamoDBExecutionRole	AWS managed	None
☐	AWSLambdaENIManagementAccess	AWS managed	None
☑	AWSLambdaExecute	AWS managed	None
☐	AWSLambdaInvocation-DynamoDB	AWS managed	None
☐	AWSLambdaKinesisExecutionRole	AWS managed	None
☐	AWSLambdaMSKExecutionRole	AWS managed	None
☑	AWSLambdaRole	AWS managed	None
☐	AWSLambdaSQSQueueExecutionRole	AWS managed	None
☐	AWSLambdaVPCAccessExecutionRole	AWS managed	None
☐	CloudWatchLambdaInsightsExecutionRolePolicy	AWS managed	None

Cancel | Previous | Next: Tags

https://us-east-1.console.aws.amazon.com/iam/home#/users$new?step=review&accessKey&userNames=polly-js-tes...

aws | Services | Search [Alt+S]

Add user

1 2 3 4 5

Review

Review your choices. After you create the user, you can view and download the autogenerated password and access key.

User details

User name	polly-js-test
AWS access type	Programmatic access - with an access key
Permissions boundary	Permissions boundary is not set

Permissions summary

The following policies will be attached to the user shown above.

Type	Name
Managed policy	AWSLambdaExecute
Managed policy	AWSLambdaRole
Managed policy	AmazonPollyFullAccess

Cancel | Previous | Create user

aws | Services | Search [Alt+S] | Global

Add user

1 2 3 4 5

Success

You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time.

Users with AWS Management Console access can sign-in at: https://032108903651.signin.aws.amazon.com/console

Download .csv

User	Access key ID	Secret access key
polly-js-test	AKIAQO6OW7TR36KVWDWV	********* Show

Close

Download .csv, we need to use Access key ID and Secret access key in our program later.

Step2. Create node.js programs to access Polly on Virtual Machine (VirtualBox + Raspberry Pi Desktop OS).

a> Install package aws-sdk for javascript/node.js to access Polly

$ npm init -y

$ npm install aws-sdk --save

```
raspberry@raspberry:~/cindy/wk12 $ npm init -y
Wrote to /home/raspberry/cindy/wk12/package.json:

{
  "name": "wk12",
  "version": "1.0.0",
  "description": "",
  "main": "aws-polly-demo.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}
```

```
raspberry@raspberry:~/cindy/wk12 $ npm install aws-sdk --save
npm WARN deprecated querystring@0.2.0: The querystring API is considered Legacy. new code should use the URLSearchParams API instead.

added 30 packages, and audited 31 packages in 17s

12 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
raspberry@raspberry:~/cindy/wk12 $
```

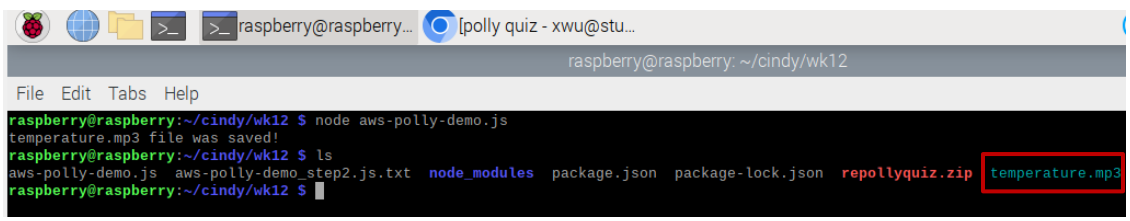
In case you need to install nodejs/npm, please use these commands:
$ curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E bash –
$ sudo apt-get install -y nodejs
$ node -v
$ npm -v
$ sudo apt-get install -f npm

b> Create file aws-polly-demo.js which accesses Polly text-to-speech function to generate an audio mp3 file.
$ vi aws-polly-demo.js

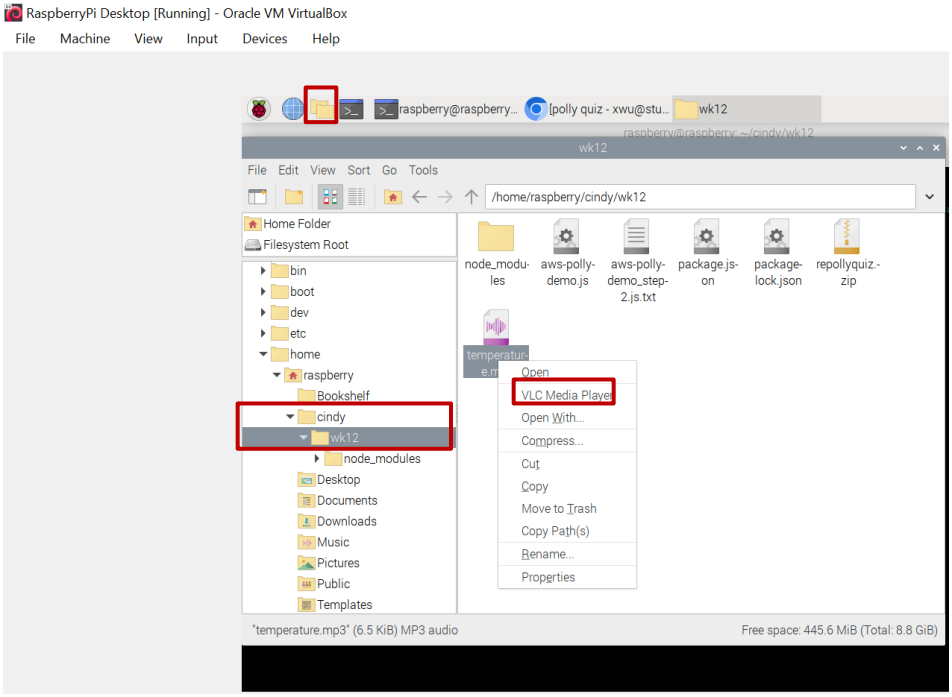
```
const AWS = require('aws-sdk')
const Fs = require('fs')

var accessKeyId = 'AKIAQO6OR36KVWDWV';
var secretAccessKey = '1ToosRiSGpkOkoOEF+kJtfPWBTDaI2ycWB1';
const Polly = new AWS.Polly({
 accessKeyId: accessKeyId,
 secretAccessKey: secretAccessKey,
 signatureVersion: 'v4',
 region: 'us-west-2'
});

var params = {
 OutputFormat: "mp3",
 SampleRate: "8000",
 Text: "This is a temperature test!",
 TextType: "text",
 VoiceId: "Joanna"
};

Polly.synthesizeSpeech(params, (err, data) => {
 if (err) {
    console.log(err);
 } else if (data) {
   if (data.AudioStream instanceof Buffer) {
     Fs.writeFile("./temperature.mp3", data.AudioStream, function(err) {
       if (err) {
         return console.log(err)
       }
       console.log("temperature.mp3 file was saved!")
     })
  }
 }
});
```

c> Run aws-polly-demo.js

$ node aws-polly-demo.js



After the audio file temperature.mp3 is generated, using integrated VLC media player to play it.



d> To work with the speaker on a local computer or Raspberry Pi, we can use node-speaker library. Refer this link https://github.com/TooTallNate/node-speaker to install this library.

```
raspberry@raspberry:~/cindy/wk12 $ sudo apt-get install libasound2-dev
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following package was automatically installed and is no longer required:
  sse3-support
Use 'sudo apt autoremove' to remove it.
Suggested packages:
  libasound2-doc
The following NEW packages will be installed:
  libasound2-dev
0 upgraded, 1 newly installed, 0 to remove and 3 not upgraded.
Need to get 126 kB of archives.
After this operation, 681 kB of additional disk space will be used.
Get:1 http://deb.debian.org/debian bullseye/main i386 libasound2-dev i386 1.2.4-1.1 [126 kB]
Fetched 126 kB in 0s (344 kB/s)
Selecting previously unselected package libasound2-dev:i386.
(Reading database ... 166865 files and directories currently installed.)
Preparing to unpack .../libasound2-dev_1.2.4-1.1_i386.deb ...
Unpacking libasound2-dev:i386 (1.2.4-1.1) ...
Setting up libasound2-dev:i386 (1.2.4-1.1) ...
raspberry@raspberry:~/cindy/wk12 $ npm install speaker

added 8 packages, and audited 39 packages in 9s

12 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
raspberry@raspberry:~/cindy/wk12 $
```

e> Create file sensor-speaker.js

$ vi sensor-speaker.js

```
raspberry@raspberry: ~/cindy/
File  Edit  Tabs  Help
const AWS = require('aws-sdk')
const Fs = require('fs')
const Stream = require('stream')
const Speaker = require('speaker')

var accessKeyId = 'AKIAQO6OW76KVWDWV';
var secretAccessKey = '1ToospkOkoOEF++4exkJtfPWBTDaI2ycWB1';
const Polly = new AWS.Polly({
 accessKeyId: accessKeyId,
 secretAccessKey: secretAccessKey,
 signatureVersion: 'v4',
 region: 'us-west-2'
});

var params = {
 OutputFormat: "pcm", // mp3 has error
 SampleRate: "8000",
 Text: "This is a temperature test!",
 TextType: "text",
 VoiceId: "Joanna"
};

const Player = new Speaker({
  channels: 1,          // 1 channels
  bitDepth: 16,         // 16-bit samples
  sampleRate: 8000      // 8000 Hz sample rate
                        // you can try different value if you can not hear the voice clearly
});

Polly.synthesizeSpeech(params, (err, data) => {
 if (err) {
   console.log(err);
 } else if (data) {
   if (data.AudioStream instanceof Buffer) {

     var bufferStream = new Stream.PassThrough();
     bufferStream.end(data.AudioStream);
     bufferStream.pipe(Player);

   }else{
     console.log('data is not AudioStream');
   }
"sensor_speaker.js" 44L, 1129B
```

f> Run sensor-speaker.js

$ node sensor-speaker.js

```
raspberry@raspberry:~/cindy/wk12 $ node sensor_speaker.js
raspberry@raspberry:~/cindy/wk12 $
```

If you can hear the voice successfully, it means that your program is correct.

DONE!!!