

Assignment 3

Use one of the services like translate, transcribe, Rekognition, etc. and upload an appropriate file in S3 (depending on the target AI service, upload different file formats). That upload should trigger a lambda function code and use that event to learn about the location of object in the S3 bucket. After that, pass that object to the AI service to do the job like translation, or detecting some sentiment or detecting text inside image, based on the chosen AI service. The result of that AI service should be either printed in Lambda function or show in Cloudwatch or save in S3.

Step 1: Create S3 Bucket and Inside it Create a Folder.

The screenshot shows the AWS S3 Management Console. A green banner at the top indicates that a bucket named "aiservicelab3" has been successfully created. Below the banner, the "Account snapshot" section displays basic metrics: Total storage (45.6 KB), Object count (6), and Average object size (7.6 KB). The "Buckets" section lists seven existing buckets, including "aiservicelab3".

Name	AWS Region	Access	Creation date
aiservicelab3	US East (N. Virginia) us-east-1	Bucket and objects not public	February 7, 2023, 13:01:32 (UTC-05:00)
awslabonebucket	US East (N. Virginia) us-east-1	Bucket and objects not public	January 17, 2023, 16:17:34 (UTC-05:00)
labtwo-s3bucket	US East (N. Virginia) us-east-1	Bucket and objects not public	January 31, 2023, 13:11:46 (UTC-05:00)
modulitetwosession	US East (N. Virginia) us-east-1	Bucket and objects not public	January 31, 2023, 11:30:36 (UTC-05:00)
my-bucket-labnr	US East (N. Virginia) us-east-1	Objects can be public	January 17, 2023, 09:23:25 (UTC-05:00)
my-nrs3bucket	US East (N. Virginia) us-east-1	Objects can be public	January 14, 2023, 16:38:08 (UTC-05:00)
sm-nr-s3bucket	US East (N. Virginia) us-east-1	Bucket and objects not public	January 14, 2023, 16:52:39 (UTC-05:00)

The screenshot shows the "aiservicelab3" bucket details page. The "Objects" tab is selected, showing a table with one row: "No objects". A large orange "Upload" button is prominently displayed at the bottom.

Screenshot of the AWS S3 Management Console showing the 'Create folder' dialog for the 'aiservicelab3' bucket.

The dialog shows:

- A warning message: "Your bucket policy might block folder creation. If your bucket policy prevents uploading objects without specific tags, metadata, or access control list (ACL) grants, you will not be able to create a folder using this configuration. Instead, you can use the upload configuration to upload an empty folder and specify the appropriate settings."
- A "Folder" section with a "Folder name" input field containing "Input".
- A "Server-side encryption" section with a note: "The following settings apply only to the new folder object and not to the objects contained within it." It shows the "Encryption key type" set to "Amazon S3-managed keys (SSE-S3)".

At the bottom right are "Cancel" and "Create folder" buttons.

Screenshot of the AWS S3 Management Console showing the 'Input' folder in the 'aiservicelab3' bucket.

The interface includes:

- A toolbar with "Copy S3 URI" and other actions.
- A "Objects" tab showing 0 objects.
- A search bar and filter: "Find objects by prefix" and "Name", "Type", "Last modified", "Size", "Storage class".
- A message: "No objects" and "You don't have any objects in this folder."
- An "Upload" button.

At the bottom right are "Copy S3 URI", "Actions", "Create folder", and "Upload" buttons.

Step 2: Create Lambda Function One and add S3 Trigger to it.

The screenshot shows the 'Create function' wizard in the AWS Lambda console. The first step, 'Basic information', is selected. The 'Author from scratch' option is chosen. In the 'Function name' field, the value 'aiservicelab3_audio_transcribe' is entered. The 'Runtime' is set to 'Python 3.9'. The 'Architecture' is set to 'x86_64'. Under 'Permissions', it says 'By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.' The bottom navigation bar includes 'Feedback', 'Language', and status indicators like 'Cloudy' and '3°C'.

The screenshot shows the 'Create function' wizard in the AWS Lambda console. The second step, 'Advanced settings', is selected. It shows options for 'Execution role' (with 'Create a new role with basic Lambda permissions' selected), 'Existing role' (with 'LabRole' chosen), and 'Advanced settings' (with a 'Create function' button at the bottom right). The bottom navigation bar includes 'Feedback', 'Language', and status indicators like 'Cloudy' and '3°C'.

Screenshot of the AWS Lambda console showing the function overview for 'aiservicelab3_audio_transcribe'. The function was successfully created and is now ready for configuration and testing.

Function Overview:

- Name:** aiservicelab3_audio_transcribe
- Description:** -
- Last modified:** 16 seconds ago
- Function ARN:** arn:aws:lambda:us-east-1:094583137742:function:aiservicelab3_audio_transcribe
- Function URL:** -

Code Source: Info

Trigger Configuration: Add trigger

Actions: Throttle, Copy ARN, Actions

Navigation: Lambda > Functions > aiservicelab3_audio_transcribe

Toolbar: Code, Test, Monitor, Configuration, Aliases, Versions

System Information: © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences. Weather: 3°C Cloudy. Date: 2023-02-07. Time: 108 PM.

Screenshot of the AWS Lambda console showing the 'Add triggers - Lambda' configuration page for the 'aiservicelab3 - S3 bucket' trigger.

Trigger configuration:

Bucket: S3 aws storage

Event type: All object create events

Prefix - optional: Input/

Suffix - optional: e.g. jpg

Recursive invocation: I acknowledge that using the same S3 bucket for both input and output is not recommended and that this configuration can cause recursive invocations, increased Lambda usage, and increased costs.

Notes: Lambda will add the necessary permissions for AWS S3 to invoke your Lambda function from this trigger. Learn more about the Lambda permissions model.

Buttons: Cancel, Add

System Information: © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences. Weather: 3°C Cloudy. Date: 2023-02-07. Time: 111 PM.

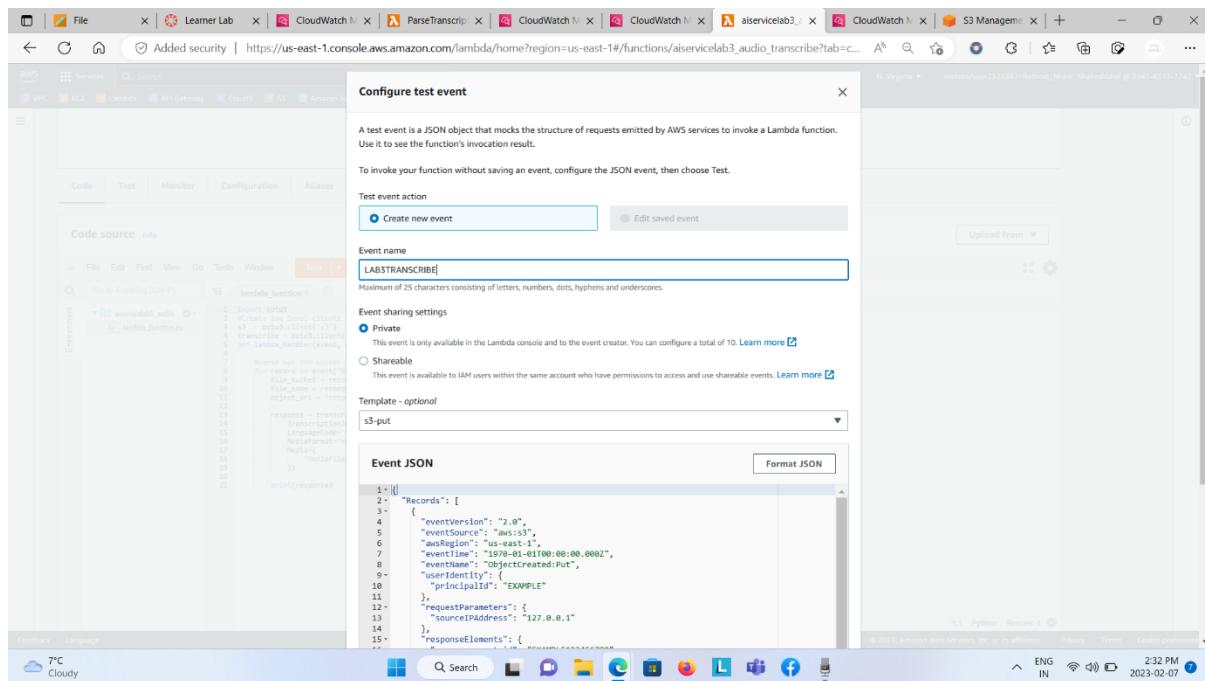
The screenshot shows the AWS Lambda console interface. The top navigation bar includes tabs for Services, Search, and AWS Lambda. Below the navigation bar, the main area displays the configuration for the function 'aiservicelab3_audio_transcribe'. The 'Configuration' tab is active. On the left, a sidebar lists various configuration options: General configuration, Triggers, Permissions, Destinations, Function URL, Environment variables, Tags, VPC, Monitoring and operations tools, Concurrency, and Asynchronous invocation. The 'Triggers' section is expanded, showing one trigger named 'S3: aiservicelab3'. The details for this trigger indicate it triggers on 's3:ObjectCreated*' events from the 'aiservicelab3' bucket. The right side of the screen shows the function's ARN, 'arn:aws:lambda:us-east-1:094583137742:function:aiservicelab3_audio_transcribe', and its function URL. At the bottom, there are tabs for Code, Test, Monitor, Configuration, Aliases, and Versions.

Write the code into it.

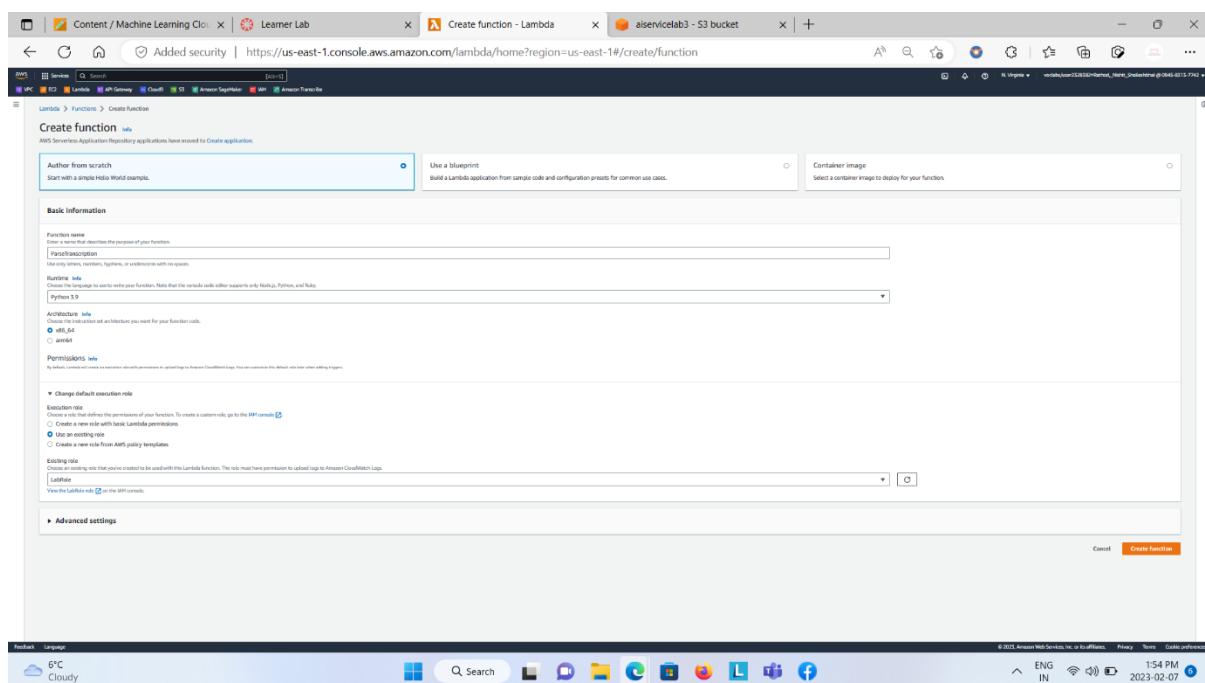
The screenshot shows the AWS Lambda console interface, specifically the code editor for the function 'aiservicelab3_audio_transcribe'. The top navigation bar includes tabs for File, Edit, Find, View, Go, Tools, Window, Test, Deploy, and a search bar. The main area shows the code source in a code editor window. The code is written in Python and uses the Boto3 library to interact with the S3 and Transcribe services. It defines an event handler 'lambda_handler' that parses the S3 event and starts a transcription job using the Transcribe service. The code includes imports for Boto3, the AWS Lambda context, and the AWS SDK for Python (Boto3). The code editor has syntax highlighting and a code completion feature. The status bar at the bottom indicates the code was successfully updated.

```
1 import boto3
2 import json
3 # Create an S3 client for the event
4 s3 = boto3.client('s3')
5 transcribe = boto3.client('transcribe')
6
7 def lambda_handler(event, context):
8     # Parse out the bucket & file name from the event handler
9     # For example, if the event is {"Records": [{"s3": {"bucket": {"name": "filebucket"}, "object": {"key": "file.mp3"}}, "event": "ObjectCreated:Put"}]}, then
10    file_bucket = record['s3']['bucket']['name']
11    file_key = record['s3']['object']['key']
12    object_url = "https://aiservicelab3.s3.amazonaws.com/Input/Audio_Recording.mp3"
13
14    response = transcribe.start_transcription_job(
15        TranscriptionJobName=file_name.replace('/', '_'),
16        LanguageCode='en-US',
17        MediaFormat='mp3',
18        MediaFileUri=object_url
19    )
20
21    print(response)
```

Step 3: From Configure Test Event, create new event and attach s3-put template.



Step 4: Create Lambda Function Two and add CloudWatch Trigger to it.



The screenshot shows the AWS Lambda console with the 'ParseTranscription' function selected. The top navigation bar includes tabs for VPC, EC2, Lambda, API Gateway, Cloud9, S3, Amazon SageMaker, IAM, and Amazon Transcribe. The main content area displays the function's overview, including its name 'ParseTranscription', a description field, and a code editor tab. The code editor shows the Lambda function code. Below the code editor are tabs for Test, Monitor, Configuration, Aliases, and Versions. The bottom of the screen shows the Windows taskbar with various pinned icons.

The screenshot shows the AWS CloudWatch Management Console with the 'CloudWatchRuleForPT' rule selected. The left sidebar lists CloudWatch services like Alarms, Logs, Metrics, X-Ray traces, Events, Application monitoring, and Insights. The main content area shows the rule configuration, including the ARN, event pattern (JSON), status (Enabled), and targets section. The targets table lists one target: a Lambda function named 'ParseTranscription'. The bottom of the screen shows the Windows taskbar with various pinned icons.

The screenshot shows the AWS Lambda console interface. At the top, there are several tabs: Content / Machine Learning CI, Learner Lab, CloudWatch Management Console, ParseTranscription - Lambda, and aiservicelab3 - S3 bucket. The ParseTranscription - Lambda tab is active.

In the main area, the function name is ParseTranscription. It has a description and was last modified 48 seconds ago. The Function ARN is arn:aws:lambda:us-east-1:094583157742:function:ParseTranscription. The Function URL is listed as well.

The trigger section shows an EventBridge (CloudWatch Events) trigger. There are buttons for Throttle, Copy ARN, and Actions. Below the trigger, there are buttons for Code, Test, Monitor, Configuration, Aliases, and Versions. The Code tab is selected, showing the code source in a text editor.

The code source is as follows:

```
lambda_function.py
1 import json
2 import boto3
3 import os
4 import urllib.request
5 BUCKET_NAME = os.environ['BUCKET_NAME']
6 s3 = boto3.client('s3')
7 transcribe = boto3.client('transcribe')
8 def lambda_handler(event, context):
9     print("Received event: " + str(event))
10    # Get the file from the event and read it into memory
11    file_name = event['detail']['transcriptionJobName']
12    job = transcribe.get_transcription_job(JobName=file_name)
13    print(job)
14    content = job['Transcript']['TranscriptText']
15    print(content)
16    output = s3.Object(BUCKET_NAME, file_name).put_object(
17        Body=content
18    )
19    print(output)
```

Write the code into it.

The screenshot shows the AWS Lambda console interface, similar to the previous one, but with a green banner at the top indicating "Successfully updated the function ParseTranscription."

The function ParseTranscription is shown again with its trigger and configuration details. The "Code source" tab is selected, and the code editor shows the following updated code:

```
lambda_function.py
1 import json
2 import boto3
3 import os
4 import urllib.request
5 BUCKET_NAME = os.environ['BUCKET_NAME']
6 s3 = boto3.resource('s3')
7 transcribe = boto3.client('transcribe')
8 def lambda_handler(event, context):
9     print("Received event: " + str(event))
10    # Get the file from the event and read it into memory
11    file_name = event['detail']['transcriptionJobName']
12    job = transcribe.get_transcription_job(JobName=file_name)
13    print(job)
14    content = job['Transcript']['TranscriptText']
15    print(content)
16    output = s3.Object(BUCKET_NAME, file_name).put_object(
17        Body=content
18    )
19    print(output)
20    # Read the file from S3 and print its contents to the log
21    file_content = s3.Object(BUCKET_NAME, file_name).get()['Body'].read().decode('UTF-8')
22    data = json.loads(file_content)
23    transcribed_text = data['results'][0]['transcripts'][0]['transcript']
24    object = s3.Object(BUCKET_NAME, file_name).put_object(
25        Body=transcribed_text
26    )
27    print(object)
```

Step 5: Upload .mp3 file into the S3 bucket, inside the created folder.

The screenshot shows the AWS S3 Management Console interface. In the top navigation bar, the URL is https://s3.console.aws.amazon.com/s3/upload/aiservicelab3?region=us-east-1&prefix=Input/. The main area is titled "Upload" and contains sections for "Files and folders", "Destination", and "Permissions". A file selection dialog box is overlaid on the screen, showing a list of files from the user's desktop. The file "AWS Lab 3_Transcribe_Recording.mp3" is selected. The destination is set to "s3://aiservicelab3/input/".

The screenshot shows the AWS S3 Management Console after the file has been uploaded. The "Files and folders" section now lists "1 Total, 913.3 KB" with one item: "AWS-Lab-3_Transcribe_Recording.mp3" (audio/mpeg, 913.3 KB). The "Destination" section shows the file was uploaded to "s3://aiservicelab3/input/". The "Upload" button is now highlighted in orange, indicating the upload is complete.

The screenshot shows the AWS CloudWatch Metrics interface for a Lambda function named "aiservicelab3". The "Metrics" tab is selected, displaying a graph of "Latency" over time. The graph shows a sharp increase in latency starting around February 7, peaking at approximately 10 seconds on February 8, before returning to baseline. The "Logs" tab is also visible, showing log entries for the function. The bottom of the screen displays the AWS navigation bar and various service links.

Edit the created event.

Successfully updated the function `aiservicelab3_audio_transcribe`.

Test event info

To invoke your function without saving an event, modify the event, then choose Test. Lambda uses the modified event to invoke your function, but does not overwrite the original event until you choose Save changes.

Test event action

Create new event Edit saved event

Event name

LABTRANSCRIBE

Event JSON

```
3-  "Records": [
  4-    {
  5-      "eventVersion": "2.0",
  6-      "eventSource": "aws:s3",
  7-      "awsRegion": "us-east-1",
  8-      "eventTime": "2023-07-10T18:00:00.000Z",
  9-      "eventName": "ObjectCreated:Put",
 10-      "userIdentity": {
 11-        "principalId": "EXAMPLE"
 12-      },
 13-      "requestParameters": {
 14-        "sourceIPAddress": "127.0.0.1"
 15-      },
 16-      "responseElements": {
 17-        "x-amz-request-id": "EXAWL1E134567890",
 18-        "x-amz-id-2": "EXAPL1E233567890bcde1ghijklmnbcdiasome/mnopqrstuvwxyzABCDEF0H"
 19-      }
 20-    }
 21-  ],
 22-  "s3": {
 23-    "configuration": "testConfigRule",
 24-    "bucket": "aiservicelab3",
 25-    "name": "aiservicelab3",
 26-    "userIdentity": {
 27-      "principalId": "EXAMPLE"
 28-    },
 29-    "arn": "arn:aws:s3:::aiservicelab3"
 30-  },
 31-  "object": {
 32-    "key": "Input/AWS-Lab_3_Transcribe.Recording.mp3",
 33-    "size": 1048576
 34-  }
]
```

Format JSON

Feedback Language © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 7:43 PM 2023-07-10

Step 6: Open CloudWatch Logs and see the desired transcribe result.

Log group details

ARN arn:aws:logs:us-east-1:094583157742:log-group:/aws/lambda/aiservicelab3_audio_transcribe:*	Metric Filters 0	Data protection - new Inactive
Creation time 4 minutes ago	Subscription filters 0	Sensitive data found - new -
Retention Never expire	Contributor Insights rules 0	KMS key ID -
Stored bytes -		

Log streams

Log streams (1)	Actions	Create log stream	Search all log streams
2023/02/07/[SLATEST]9ec7f4c4164345e99b684cc49d793477	Last event time	2023-02-07 19:00:34 (UTC-05:00)	

Timestamp	Message
2023-02-07T19:00:33.704-05:00	No older events at this moment. Retry
2023-02-07T19:00:34.048-05:00	START RequestId: 4fd5b101-815d-48d7-aa83-3020c94fc574 Version: \$LATEST ("TranscriptionJobName": "Input4s-L", "TranscriptionJobStatus": "IN_PROGRESS", "LanguageCode": "en-US", "MediaFormat": "mp3", "MediaFileUri": "https://s3.amazonaws.com/aiservicelab3-audio-transcribe/164345e99b684cc49d793477/164345e99b684cc49d793477.mp3")
2023-02-07T19:00:34.050-05:00	END RequestId: 4fd5b101-815d-48d7-aa83-3020c94fc574
2023-02-07T19:00:34.056-05:00	REPORT RequestId: 4fd5b101-815d-48d7-aa83-3020c94fc574 Duration: 255.02 ms Billed Duration: 256 ms Memory Size: 128 MB Max Memory Used: 72 MB

Screenshot of the AWS CloudWatch Log Groups interface for the /aws/lambda/ParseTranscription log group.

Log group details:

- ARN:** arn:aws:logs:us-east-1:094583137742:log-group:/aws/lambda/ParseTranscription:*
- Metric filters:** 0
- Subscription filters:** 0
- Contributor Insights rules:** 0
- Creation time:** 4 minutes ago
- Retention:** Never expire
- Stored bytes:** -

Data protection - new:

- Inactive
- Sensitive data found - new
- KMS key ID

Log streams: (1)

Log stream	Last event time
2023/02/08/[\$LATEST]f28aeaf6f45648a2ad09d358561f561f	2023-02-07 19:01:02 (UTC-05:00)

Screenshot of the AWS CloudWatch Log Events interface for the /aws/lambda/ParseTranscription log group.

Log events:

You can use the filter bar below to search for and match terms, phrases, or values in your log events. [Learn more about filter patterns](#)

Timestamp	Message
2023-02-07T19:01:01.133-05:00	INIT_START Runtime Version: python:3.9.v16 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:07a48df201798d627f2b65bf03bb27aab4a655a1d919c3296486f9...
2023-02-07T19:01:01.523-05:00	START RequestId: 2d54b598-5f14-4d67-8e29-a7c39e40436a Version: \$LATEST
2023-02-07T19:01:01.180-05:00	https://s3.us-east-1.amazonaws.com/aus-transcribe-be-us-east-1-prod/094583137742/InputAWS-L/29404181-18ed-4fee-9100-af1b0553d283/asrOutput.json?X-Amz-SecurityToken=...
2023-02-07T19:01:02.100-05:00	"\\"jobName"\":\\"InputAWS-1\\",\\"accountId"\":\\"094583137742\\",\\"results\"\":[\\"transcripts\"\":[\\"transcript\"\":\\"Namaste, my name is today I will be doing an ...
2023-02-07T19:01:02.755-05:00	END RequestId: 2d54b598-5f14-4d67-8e29-a7c39e40436a
2023-02-07T19:01:02.755-05:00	REPORT RequestId: 2d54b598-5f14-4d67-8e29-a7c39e40436a Duration: 1232.14 ms Billed Duration: 1233 ms Memory Size: 128 MB Max Memory Used: 73 MB Init Dura...
2023-02-07T19:01:29.195-05:00	START RequestId: f3db0ffe-6d29-464e-9342-0ed5de50a8e0 Version: \$LATEST
2023-02-07T19:01:29.391-05:00	https://s3.us-east-1.amazonaws.com/aus-transcribe-be-us-east-1-prod/094583137742/InputAWS-L/29404181-18ed-4fee-9100-af1b0553d283/asrOutput.json?X-Amz-SecurityToken=...
2023-02-07T19:01:29.643-05:00	"\\"jobName"\":\\"InputAWS-1\\",\\"accountId"\":\\"094583137742\\",\\"results\"\":[\\"transcripts\"\":[\\"transcript\"\":\\"Namaste, my name is today I will be doing an ...
2023-02-07T19:01:29.878-05:00	END RequestId: f3db0ffe-6d29-464e-9342-0ed5de50a8e0
2023-02-07T19:01:29.878-05:00	REPORT RequestId: f3db0ffe-6d29-464e-9342-0ed5de50a8e0 Duration: 683.63 ms Billed Duration: 684 ms Memory Size: 128 MB Max Memory Used: 75 MB

No newer events at this moment. Auto reply paused. [Resume](#)

Step 7: Open Amazon Transcribe and Transcription job would be created with defined name.

The screenshot shows the Amazon Transcribe service interface. On the left, a sidebar lists various services: Real-time transcription, Transcription jobs, Custom language model, Custom vocabulary, Vocabulary filtering, Amazon Transcribe Call Analytics (expanded), Real-time Analytics (New), Post-call Analytics, Category Management, and Amazon Transcribe Medical (expanded). The main content area displays a table titled "Transcription jobs (1) Info". The table has columns: Name, Status, Language, Language settings, Model type, Model name, Created, and Expires in. One job is listed: "InputAWS-L" (Status: Complete, Language: English, US (en-US), Model type: Specific language, Model name: General, Created: February 7 2023, 19:00 (UTC-05:00), Expires in: 89 days). Action buttons for Download, Copy, Delete, and Create job are at the top right. A navigation bar at the top includes tabs for Services, Search, and [Alt+S]. The URL in the address bar is https://us-east-1.console.aws.amazon.com/transcribe/home?region=us-east-1#jobs.

Screenshot of the AWS Management Console showing the Amazon Transcribe service details for a transcription job named "InputAWS-L".

Job details

Name	Model	Audio identification	Input data location
InputAWS-L	None	Disabled	https://aiservicelab3.s3.amazonaws.com/Input/AWS-L-3_Transcribe_Recording.mp3
Status	Created 2023-02-07, 7:00:33 p.m.	Alternative results Disabled	
Language	Started 2023-02-07, 7:00:34 p.m.	Custom vocabulary None	
Language settings	Ended 2023-02-07, 7:00:57 p.m.	PII reduction Disabled	
Expiration info	Input file format mp3	Vocabulary filter *	
The transcription is available for 89 more days.			
	48000 Hz		

Transcription preview

Select download to save a local copy of the transcription.

[Download](#)

Feedback Language © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 7:04 PM 2023-02-07

Screenshot of the AWS Management Console showing the Amazon Transcribe service details for the same transcription job "InputAWS-L".

Transcription preview

Select download to save a local copy of the transcription.

[Download](#)

Text [Audio identification](#) [Subtitles](#)

Namaste, my name is today I will be doing an assignment of AWS that is amazon Web service. In this assignment, I'm going to transcribe the audio file with amazon Transcribe. That is an AI service. Also, I will be using Lambda function And as three bucket.

Tags 0

Key Value

No tags associated with the resource.

[Manage Tags](#)

Application integration

Feedback Language © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 7:04 PM 2023-02-07

Step 8: Save the result in .txt file of the transcript from the Transcription job using lambda function two.

The screenshot shows the AWS Lambda service interface. In the top navigation bar, 'aiservicelab3' is selected under the 'Functions' section. The main area displays the configuration for the function, specifically the 'Code' tab. The code editor contains the following Python script:

```

# This function processes the transcription output and saves it to S3
import json
import boto3

s3 = boto3.client('s3')

def lambda_handler(event, context):
    # Extract the transcription results from the event
    transcription_results = event['TranscriptionJobName']

    # Define the S3 bucket and key for the output file
    bucket_name = 'aiservicelab3'
    object_key = 'InputAWS-L_Output.txt'

    # Upload the transcription results to S3
    s3.put_object(Bucket=bucket_name, Key=object_key, Body=json.dumps(transcription_results))

```

The Lambda function is triggered by a CloudWatch Events rule, which is configured to invoke the function when a new transcription job is completed. The IAM role associated with the function has the necessary permissions to access the S3 bucket and CloudWatch Events.