



## Whitepaper

# Amazon Chime SDK Tutorial



Table of Contents

	Page
Chime SDK	2
Tutorial Part 1: Deploying Amazon Chime Live Events	3
Tutorial Part 2: Deploying a Video-Call Application Using Chime SDK	11
TrackIt, An Amazon Chime SDK Partner	15
About TrackIt	15

## Chime SDK

The [Amazon Chime SDK](#) is a set of real-time communications components that can be used to quickly add audio, video, and screen sharing capabilities to web or mobile applications. Amazon Chime SDK allows companies to leverage the same communication infrastructure and services that power Amazon Chime, an online meeting service from AWS, and deliver engaging experiences in their applications.

Amazon Chime SDK is used to build real-time media applications that can send and receive audio and video while also enabling content sharing. The Amazon Chime SDK works independently of any Amazon Chime administrator accounts and does not interfere with meetings hosted on Amazon Chime. Instead, Amazon Chime SDK provides tools for developers to build and customize their own meeting applications.



To learn more about Amazon Chime SDK, visit the [Amazon Chime SDK landing page](#).

Chime SDK GitHub repository: <https://github.com/aws/amazon-chime-sdk-js>

## Tutorial Part 1: Deploying Amazon Chime Live Events

This tutorial demonstrates the deployment of Amazon Chime Live Events, a small application that showcases the features of the Amazon Chime SDK. This can provide a starting point for developers that wish to integrate a React application with the Chime SDK.

Participants who can join the demo are separated into 4 categories:

- **Moderator:** Moderates the entire demo: admits participants, promotes presenters, excludes participants, answers questions, etc.
- **Attendee:** They can ask questions, be in 1–1 video with a moderator, be promoted to a live feed, etc.
- **Talent:** Has direct access to go live, does not need to be validated by a moderator to go live
- **Broadcast:** Broadcasts the event and has the same access as Talent but does not need a validation key to be connected

The names of the people in each category, except Broadcast, must be known in advance.

### Requirements

- nodejs (no version specified but we recommend using the latest) available for download [HERE](#). You can check your node version by typing:

```
$ npm --version
```

- npm (no version specified but we recommend using the latest). npm is automatically installed with nodejs if you use the link above. Otherwise, you can always use the package manager specific to your distribution.  
Example for Ubuntu 18.04:

```
$ sudo apt-get install npm
```

After that, you can check your npm version using the following command:

```
$ npm --version
```

- AWS CLI 2.0

In order to find the current version of your AWS CLI, use the following command:

```
$ aws --version
```

In the output you receive, if the AWS version starts with aws-cli/1.x.x then you have the 1.0 version and you will need to upgrade it. Follow the instructions [HERE](#).

Example of AWS CLI 2.0 output:

```
$ aws --version  
aws-cli/2.0.55 Python/3.7.3 Linux/4.15.0-118-generic
```

The latest AWS SAM CLI version is available for download [HERE](#).

- The ability to deploy AWS resources (non-exhaustive list available [HERE](#)).

## Deploying back-end resources

We need to deploy the different resources for the back end of the demo. Go to the folder where serverless is located:

```
$ cd src/backend/serverless
```

The resources must now be deployed. But first, you have to find the name of your S3 bucket. Once you have the name, use the following command:

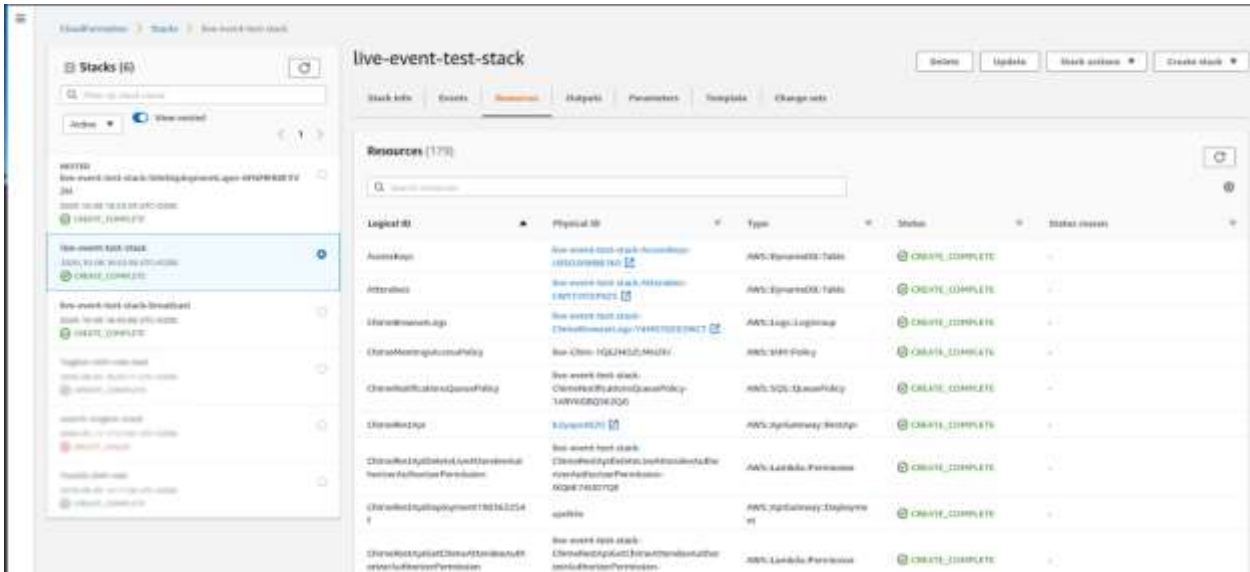
```
$ node ./deploy.js -r us-east-1 -b  
YOUR_SUPER_COOL_BUCKET_NAME -s live-event-test-stack
```

For more information on how to deploy the demo, please visit [HERE](#).

## Access link generation

We will have to generate links for each person participating in the meeting, regardless of their role.

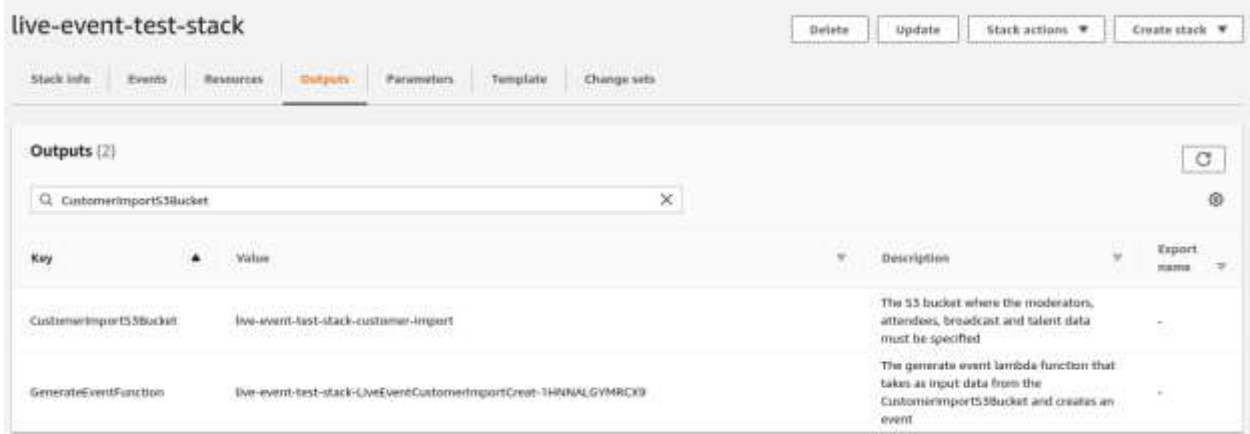
Let's start by going to the stack that has been deployed in Cloud Formation. Several resources have been deployed using Cloud Formation, called live-event-test-stack.



AWS Cloud Formation

In the Output tab, you will find a lot of useful information. Two outputs here need to be copied for future use:

- **CustomerImportS3Bucket:** corresponds to the Bucket S3 where we will have to place our files with the names of the participants of each category.
- **GenerateEventFunction:** corresponds to the name of the Lambda function that will generate the links from the .csv files that we will have earlier put in the S3 bucket above.



Output tab from the Cloud Formation stack

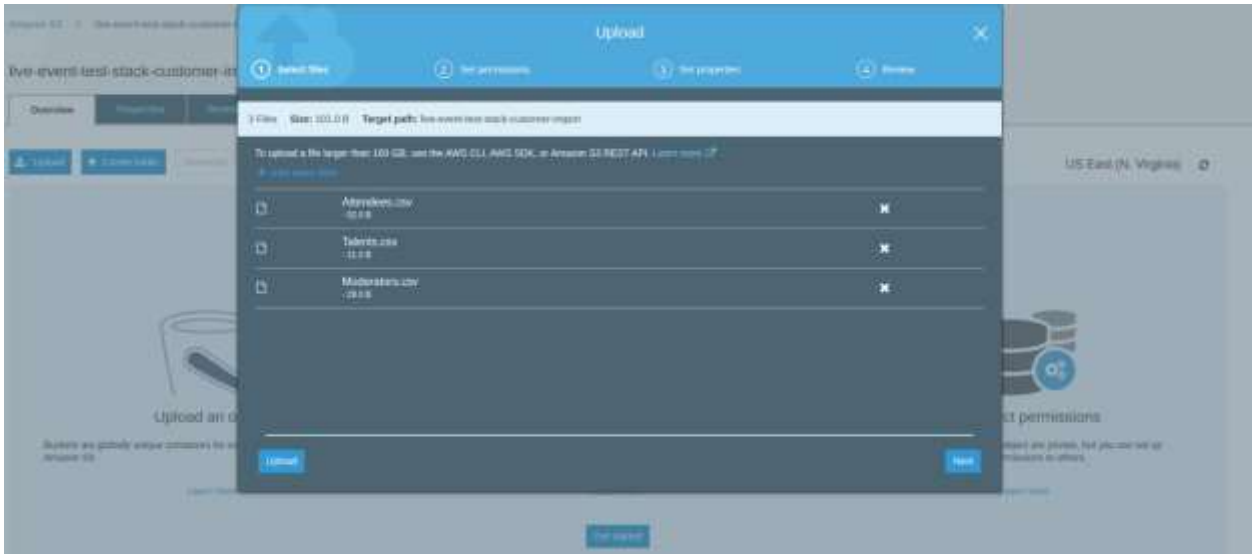
## Creation of files listing the people involved in the meeting

We have to create 3 CSV files (Attendees.csv, Moderators.csv, Talents.csv) that will correspond to each category of people. In these files, you will find the name of each person according to the category.

Here is an example of the kind of content expected in the different files:

```
full_name
Adam Smith
Alexander Hamilton
Marie Curie
```

All you have to do is upload the files to the S3 Bucket.



Upload the files to the lambda provided by the Cloud Formation Stack

## Generating the access keys to the meeting

Next, we are going to use the Lambda service to generate the access keys to the meeting. We then paste the value of the GenerateEventFunction key (which we have previously copied) in the AWS Lambda search panel and this provides us with the corresponding function. We then click on the Lambda function which redirects us to the function's panel.



AWS Lambda search panel

We will generate access keys by testing the Lambda function.

Click on **“Test”** in the top right corner and create a new test event with the following:

```
{
  "attendeesKey": "Attendees.csv",
  "moderatorsKey": "Moderators.csv",
  "talentsKey": "Talents.csv"
}
```

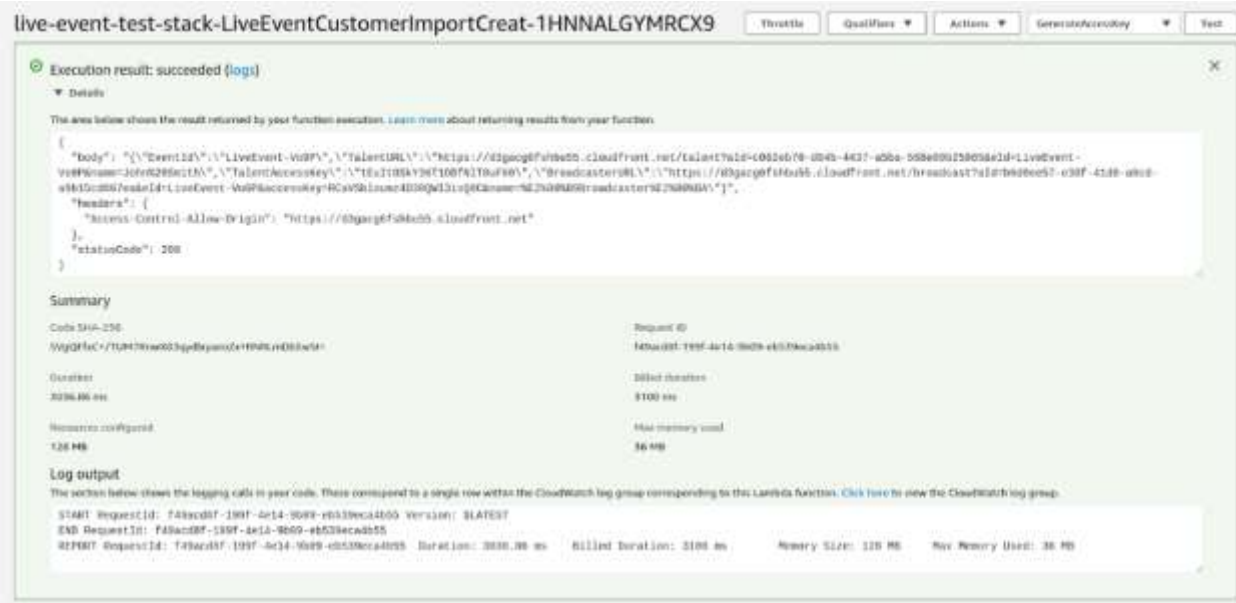
The screenshot shows the 'Configure test event' dialog box. It includes a close button (X) in the top right corner. Below the title, there is a descriptive text: 'A function can have up to 10 test events. The events are persisted so you can switch to another computer or web browser and test your function with the same events.' There are two radio buttons: 'Create new test event' (selected) and 'Edit saved test events'. Below this is an 'Event template' dropdown menu set to 'hello-world'. The 'Event name' field contains 'GenerateAccessKey'. A code editor shows the JSON payload: 

```
1 {
2   "attendeesKey": "Attendees.csv",
3   "moderatorsKey": "Moderators.csv",
4   "talentsKey": "Talents.csv"
5 }
```

 At the bottom, there are three buttons: 'Cancel', 'Format JSON', and 'Create'.

Lambda function test event configurator

Save the test and launch it by clicking on “Test”.



live-event-test-stack-LiveEventCustomerImportCreat-1HNNALGYMRCX9

Execution result: succeeded (logs)

Details

The area below shows the result returned by your function execution. Learn more about returning results from your function.

```
{
  "body": "{\\\"EventID\\\":\\\"LiveEvent-V09F\\\",\\\"TalantURL\\\":\\\"https://83gacg6fshbd5.cloudfront.net/talent/aid-0026b78-0045-4431-a0ba-558489125805a2d111v0Event-V09FName:3cfr0289a6tA\\\",\\\"TalentAccessKey\\\":\\\"E8J1B5Y9C100761T9fS05\\\",\\\"BroadcastersURL\\\":\\\"https://83gacg6fshbd5.cloudfront.net/broadcast/aid/8880ee5f-c98f-418b-81ec-9f610c88f7ade/Id:LiveEvent-V09FAccessKey:RCvY9h1zuc4H89QW13rj8QCame:SE2h89W0rroadxaterSE2h89W0r\\\"}",
  "headers": {
    "Access-Control-Allow-Origin": "https://83gacg6fshbd5.cloudfront.net"
  },
  "statusCode": 200
}
```

Summary

Code Size: 230 Bytes Request ID: 74b8cd8f-199f-4e14-9d89-6b539eca4600  
 Wavelength: /TUM7Ww003qe8pans0e1FM5vrd084w4l Request Duration: 3038.88 ms  
 Duration: 3038.88 ms Billed Duration: 3100 ms  
 Memory configured: 128 MB Max memory used: 56 MB

Log output

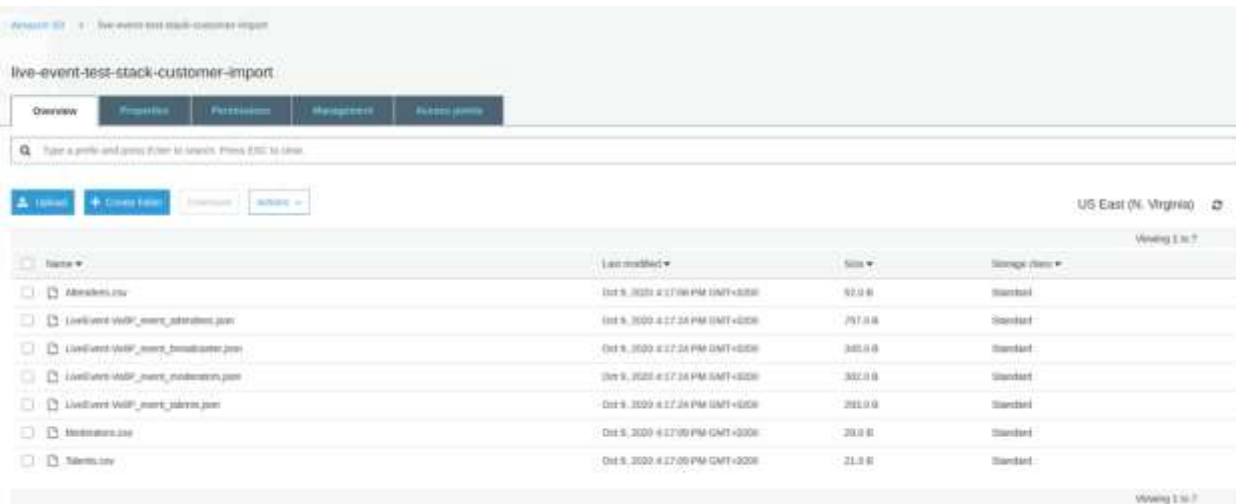
The section below shows the logging calls in your code. These correspond to a single row within the CloudWatch log group corresponding to the Lambda function. [Click here to view the CloudWatch log group.](#)

```
START RequestID: 74b8cd8f-199f-4e14-9d89-6b539eca4600 Version: $LATEST
END RequestID: 74b8cd8f-199f-4e14-9d89-6b539eca4600
REPORT RequestID: 74b8cd8f-199f-4e14-9d89-6b539eca4600  Duration: 3038.88 ms  Billed Duration: 3100 ms  Memory Size: 128 MB  Max Memory Used: 56 MB
```

Output generated by the Lambda function

## Access to the meeting

It’s time to go back to the previous S3 Bucket. We will find our access keys there.



live-event-test-stack-customer-import

Name	Last modified	Size	Storage class
<input type="checkbox"/> AmazonLogo	Oct 9, 2020 4:17:06 PM GMT+00:00	50.0 B	Standard
<input type="checkbox"/> LiveEventV09F_name_s3customer.json	Oct 9, 2020 4:17:34 PM GMT+00:00	797.0 B	Standard
<input type="checkbox"/> LiveEventV09F_name_broadcaster.json	Oct 9, 2020 4:17:34 PM GMT+00:00	345.0 B	Standard
<input type="checkbox"/> LiveEventV09F_name_moderator.json	Oct 9, 2020 4:17:34 PM GMT+00:00	382.0 B	Standard
<input type="checkbox"/> LiveEventV09F_name_viewer.json	Oct 9, 2020 4:17:34 PM GMT+00:00	283.0 B	Standard
<input type="checkbox"/> metadata.txt	Oct 9, 2020 4:17:05 PM GMT+00:00	20.0 B	Standard
<input type="checkbox"/> Talant.txt	Oct 9, 2020 4:17:05 PM GMT+00:00	21.8 B	Standard

S3 Bucket provided with the Cloud Formation stack



In each JSON file with the LiveEvent prefix, you will find content in this form:

```
{
  "full_name": "John Doe",
  "phone": "0411-732-965",
  "email": "john_doe@email.com",
  "eventId": "LiveEventSample-ke40",
  "attendeeld": "d59f0394-9b56-48e9-9954-b2309a617220",
  "access_url": "http://localhost:9001/moderator.html?ald=d59f0394-9b56-48e9-9954-b2309a617220&eld=LiveEventSample-ke40&name=John%20Doe",
  "access_key": "jHkbdH2TJlIGfrzcWUYtUE3J"
}
```

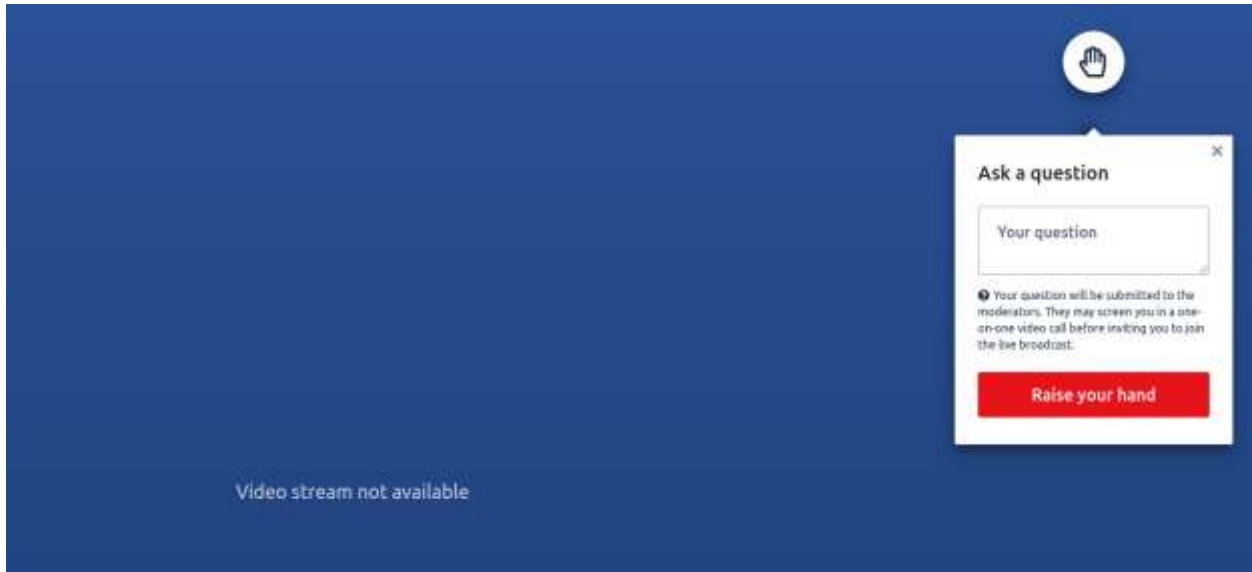
If you only have one line in your JSON file and it doesn't look as neatly presented as in the example above, don't worry. There are sites where you can "beautify" your content (like [THIS SITE](#) for example).

Give the link ("access\_url") to each person with their access key ("access\_key") and voila, you are good to go!

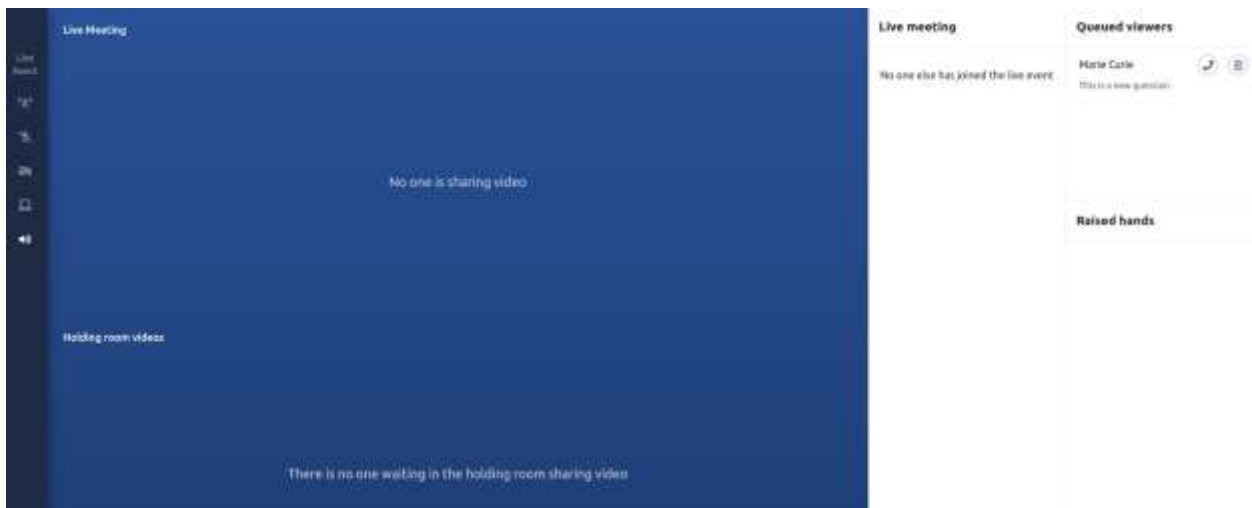
## Screenshots of the application



Example of the moderator view



Attendee view



Moderator view with an attendee queued

## Tutorial Part 2: Deploying a Video-Call Application Using Chime SDK

The following tutorial demonstrates the deployment of a complete video-call application using Amazon Chime SDK. This can provide a starting point for developers that wish to integrate a client application with the Chime SDK.

### Requirements

- nodejs (no version specified but we recommend using the latest) available for download [HERE](#). You can check your node version by typing:

```
$ node --version
```

- npm (no version specified but we recommend using the latest). npm is automatically installed with nodejs if you use the link above. Otherwise, you can always use the package manager specific to your distribution. Example for Ubuntu 18.04:

```
$ sudo apt-get install npm
```

After that, you can check your npm version using the following command:

```
$ npm --version
```

- AWS CLI 2.0  
In order to find the current version of your AWS CLI, use the following command:

```
$ aws --version
```

If the AWS version starts with aws-cli/1.x.x then you have the 1.0 version and you need to upgrade it. Follow the instructions [HERE](#). Example of AWS CLI 2.0 output:

```
$ aws --version  
aws-cli/2.0.55 Python/3.7.3 Linux/4.15.0-118-generic
```

The latest AWS SAM CLI version is available for download [HERE](#).

## Deploying resources on AWS

We need to deploy the different resources for this demo. It's a simple matter. Go to the folder where serverless is located. Despite the name of the folder, the scripts located in serverless are going to deploy back-end and front-end applications.

```
$ cd demos/serverless
```

The resources must now be deployed. But first, you have to select a name for your S3 bucket. Once you have the name, use:

```
$ npm install  
$ npm run deploy -- -r us-east-1 -b YOUR_BUCKET_NAME -s  
my_meeting_application -a meeting
```

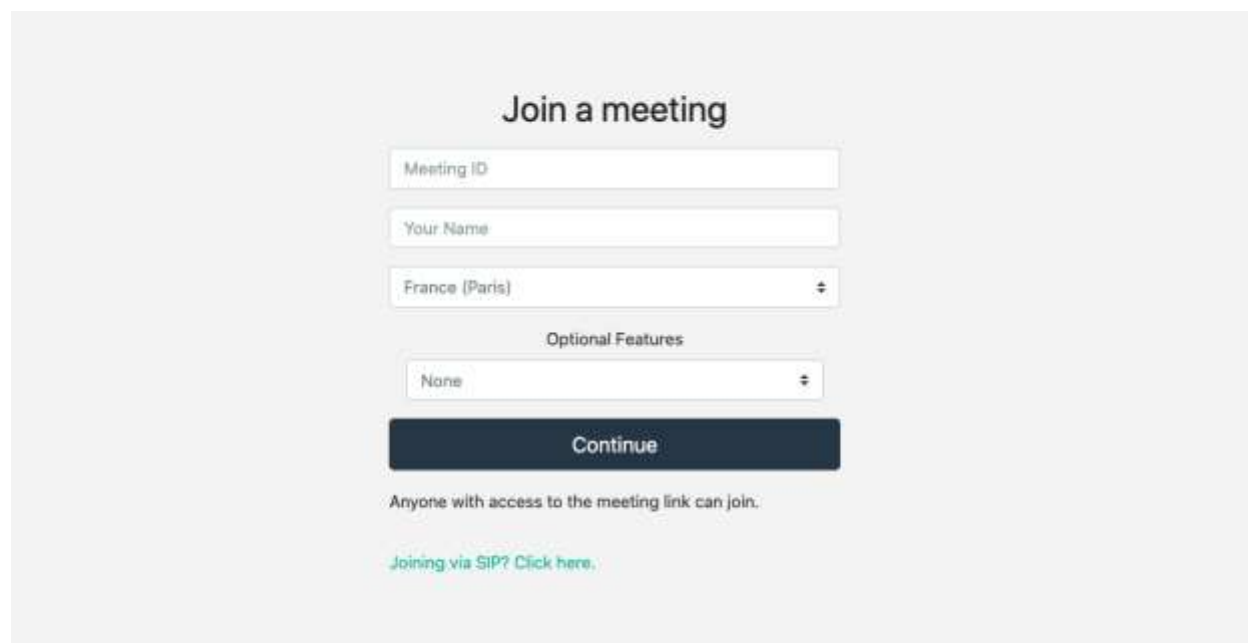
For more information on how to deploy the demo, please visit [this GitHub link](#).

## Accessing the browser application

Once the deployment is completed, you will get a link:

```
https://{ID}.execute-api.{REGION}.amazonaws.com/Prod/
```

You can directly access the link using your browser

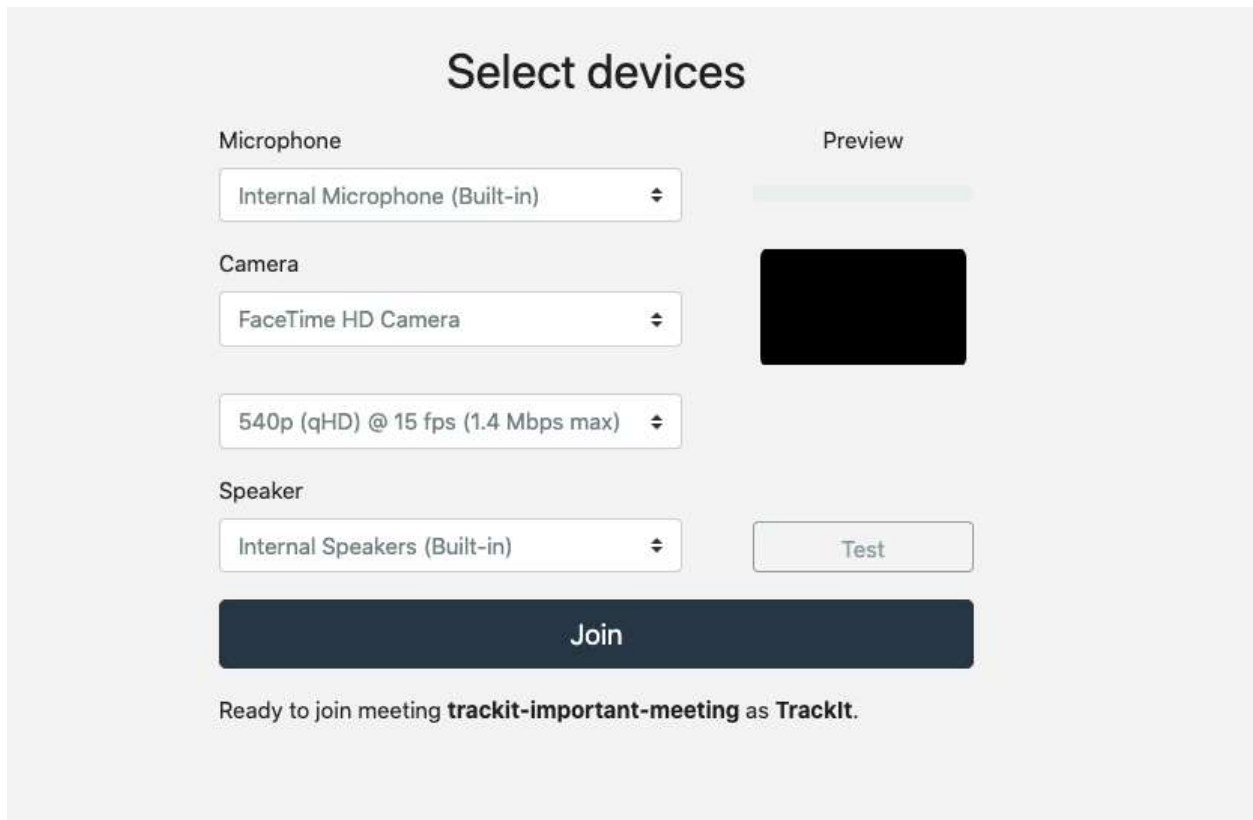


The screenshot shows a web form titled "Join a meeting". It contains the following fields and elements:

- A text input field for "Meeting ID".
- A text input field for "Your Name".
- A dropdown menu for "Region" with "France (Paris)" selected.
- A section titled "Optional Features" with a dropdown menu set to "None".
- A dark blue "Continue" button.
- Text below the button: "Anyone with access to the meeting link can join."
- A link: "Joining via SIP? Click here."

This page works like any similar video-call application. You can specify a meeting ID to join (if the meeting doesn't exist, it will be created; those IDs are unique and once a meeting is over it won't be usable again).

Let's join a meeting:



Select your audio and video devices and click 'Join'.



You are now ready to send the meeting link to your co-workers or friends and chat with them. The meeting comes with basic features:

- Video chat
- Audio chat
- Screen sharing
- Mute/unmute

There are no advanced features like authentication or moderation. It is up to you to implement them.

## Deploy the front-end locally

If you want to run the front-end locally to make some changes and test them, it's possible with the following commands:

```
$ cd demos/browser  
$ npm run start
```

Your application will now be accessible at <http://127.0.0.1:8080/>

Unfortunately, this backend relies on Lambda functions. As a result, you will need to redeploy it for each change.

## Ideas for improvement

Now that you have a perfect base for your meeting application, here are some ideas to implement in order to make it production-ready:

- Change the design with your company colors and logo
- Implement authentication using AWS Cognito
- Add permissions system to have a hierarchy between users
- Implement moderation tools
- Implement multi-room capabilities

Chime SDK GitHub repository: <https://github.com/aws/amazon-chime-sdk-js>

## TrackIt - An Amazon Chime SDK Partner

TrackIt is one of the few AWS Advanced Consulting Partners that is uniquely equipped to serve companies looking to leverage Amazon Chime. Our team of experts experienced in WebRTC has been chosen by AWS to promote and demonstrate the applicability of Amazon Chime, AWS’s signature WebRTC interactive communications service.

TrackIt’s expertise in API Integration coupled with its focus on Modern Software Development and Media & Video workflows make it an ideal partner to help integrate the Chime platform tailored to your organization’s needs. Leveraging TrackIt will allow your organization to:

- Integrate real-time communication capabilities in your applications using Amazon SDK
- Accelerate development by integrating pre-built communication building blocks into your applications
- Simplify operations by delivering high-quality and reliable end-user experiences without needing to manage infrastructure, networking, and communications components
- Integrate with other AWS services and 3rd party services to extend application functionality

### Use Cases

<p><b>Gaming</b></p> <p>Building high-quality multi-player audio and video interactions within games</p>	<p><b>Education</b></p> <p>Enhancing e-learning experiences using high-quality audio and video</p>	<p><b>Telehealth</b></p> <p>Embedding real-time communication capabilities within telehealth applications</p>
<p><b>Customer Support</b></p> <p>Adding interactive experiences to customer support applications</p>	<p><b>Retail</b></p> <p>Improving shopping experiences with real-time audio and video in stores</p>	<p><b>Broadcast OTT</b></p> <p>Adding rich interactive elements to enhance viewer engagement</p>

## About TrackIt

[TrackIt](#) is an Amazon Web Services Advanced Consulting Partner specializing in cloud management, consulting, and software development solutions based in Venice, CA.

TrackIt specializes in Modern Software Development, DevOps, Infrastructure-As-Code, Serverless, CI/CD, and Containerization with specialized expertise in Media & Entertainment workflows, High-Performance Computing environments, and data storage.

[TrackIt](#)’s forté is cutting-edge software design with deep expertise in containerization, serverless architectures, and innovative pipeline development. The TrackIt team can help you architect, design, build, and deploy customized solutions tailored to your exact requirements.

In addition to providing cloud management, consulting, and modern software development services, TrackIt also provides an [open-source AWS cost management tool](#) that allows users to optimize their costs and resources on AWS.