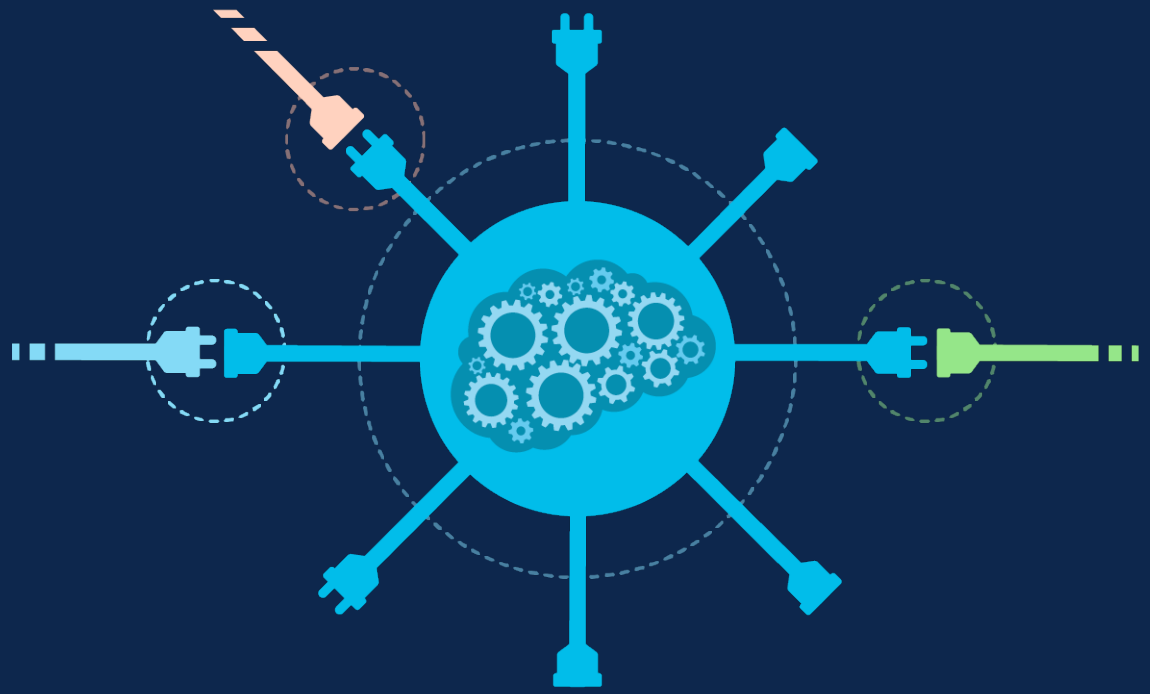
Webex Contact Center APIs

Programmatic Contact Center & Developer Portal

Shrishail
Solution Success Engineer - CCBU
April 2023

Agenda



- API Fundamentals
  - API terminology
- All New Webex CC APIs
  - Reimagining the Contact Center where anything is possible
- Developer Portal, New features
  - Overview of developer portal, new features
- OAuth2
  - Understanding the access mechanism
- Use Cases
  - What is possible using these API's, Live demo.
- Road Map, Sample & Support
  - What's next and where to get help

00
API Fundamentals

Lightning Round – Acronyms!

Demystifying Terms

Representational State Transfer

RESTful

Simple ask and answer, like asking for the blue pen on the table.

Going to a website is a “GET” rest call.

Lightning Round – Acronyms!

Demystifying Terms

HTTPS POST Event to your App

Webhooks

Asking someone to let you know when Bob arrives. When Bob arrives, you get informed so that you can greet Bob.

Lightning Round – Acronyms!

Demystifying Terms

Full Duplex HTTPS 1.1 Upgrade

Websockets

Is like either listening to the radio (one way) or talking on the phone (two way), it is a dedicated communication channel that is used to stream data.

Lightning Round – Acronyms!

Demystifying Terms

HTTP 429 - Too Many Requests

Rate Limiting

Like a circuit breaker, it stops the flow of requests to protect the system.

Lightning Round – Acronyms!

Demystifying Terms

Client-Server

Think Customer << >> Provider

Client Requests

Server Responds

Lightning Round – Acronyms!

Demystifying Terms

JavaScript Object Notation

JSON

Is a format for representing the data.

Key-value pairs, Easy to read/extract data.

Lightning Round – Acronyms!

Demystifying Terms

Graph Query Language

GraphQL

Query language used to extract data.
Request specific data – Get only that data.
Sent as part of HTTP POST request.

Lightning Round – Acronyms!

Demystifying Terms

Open Authorization 2

OAuth2

Uses Tokens instead of passwords.

Like Airbnb - owner uses their “key” to create a door code.

You use that to access the property.

When the code expires, you no longer have access, and you never touch the owner’s key!

API = Contract = Specification

Functionality. Backward Compatible. Constraints detailed.

Documentation

Specification

Authentication
Authorization
Accounting

Community
& Support

Schema

Terms of Service

Rate Limits

Examples

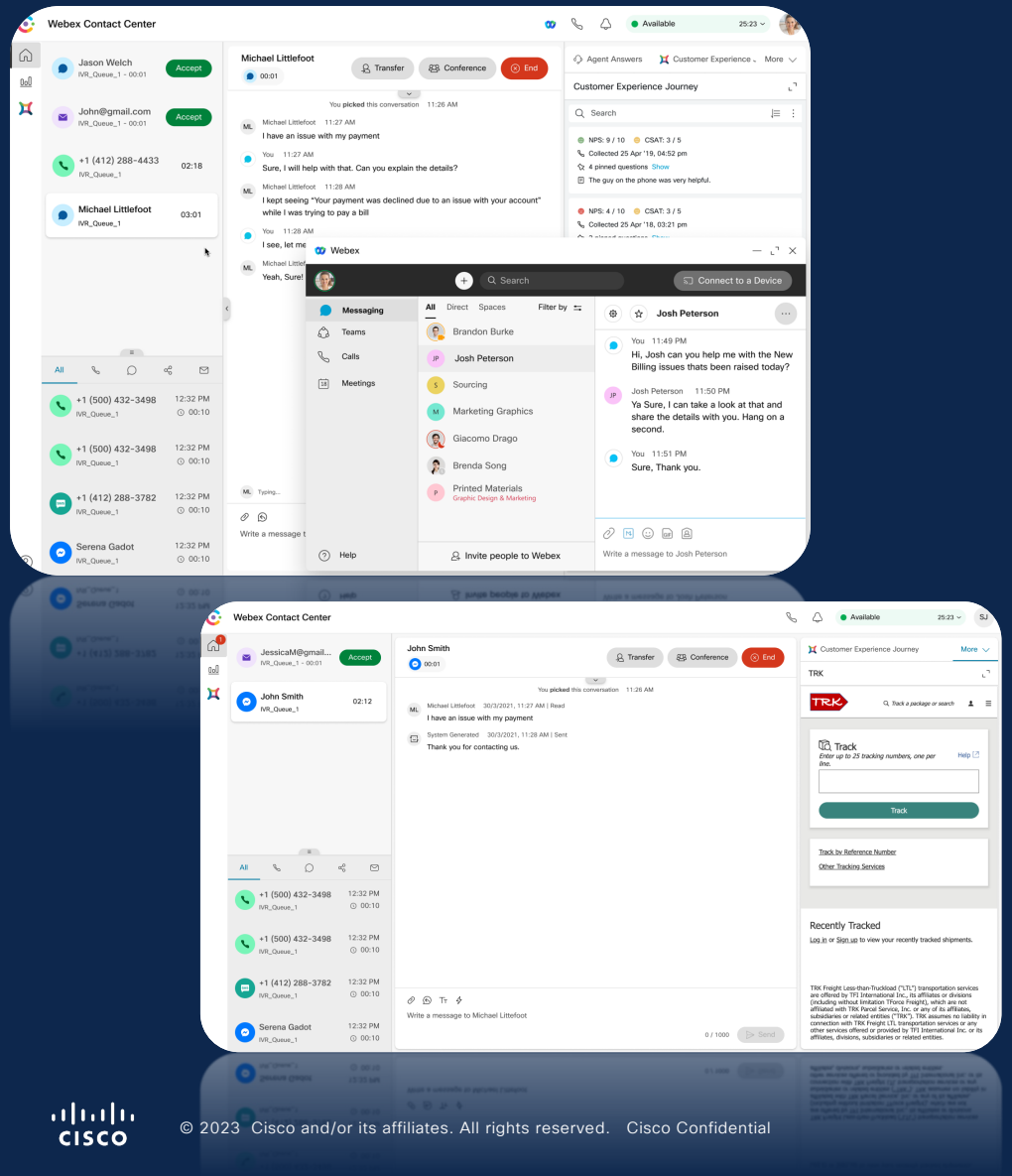
Self—Service development.

Build innovative solutions.

Application extensions that Differentiate.

01
Webex Contact Center APIs

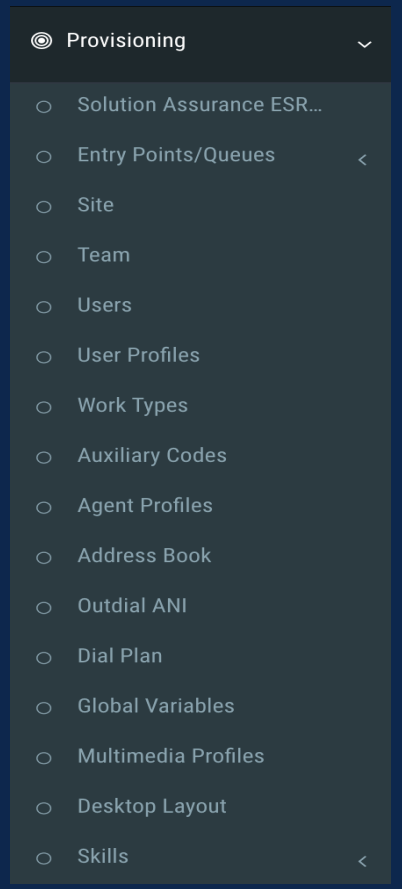
New API Platform – Flexible & Extensible



- Customize every step of the customer and agent journey
- Integrate into existing business applications and tools
- Build new features that extend the existing out-of-box features
- Track and improve KPIs

APIs Available today

Configuration APIs

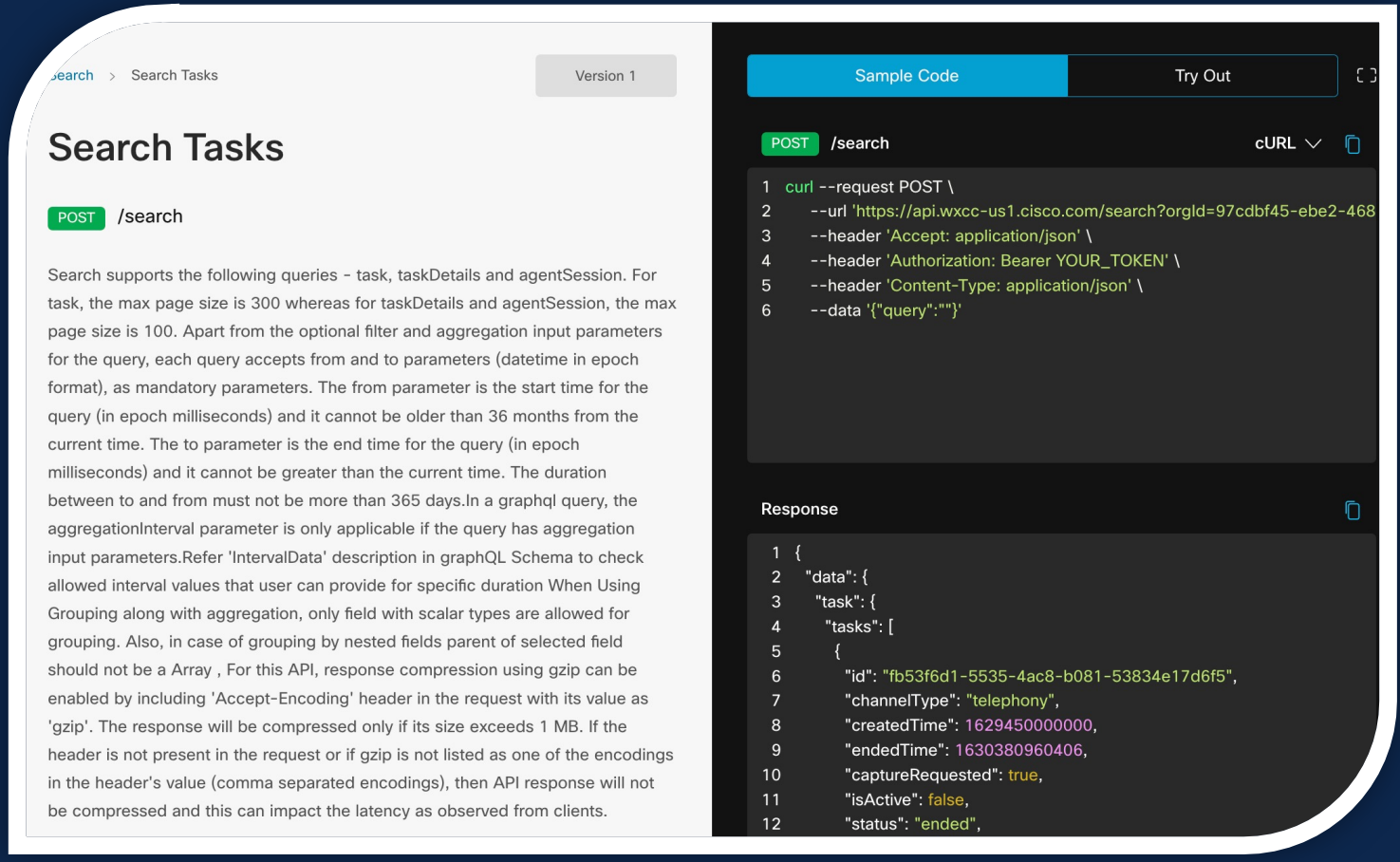


Address Book
Agent Profile
Audio files
Auxiliary codes
Queue
Desktop layout
Dial Number
Dial plan
Entry point
Users
Global Variable
Multimedia Profile
Out dial ANI
Site
Skill
Skill Profile
Team
User Profile
Work Types

(Automated provisioning, Create/Read/Delete/Update operations)

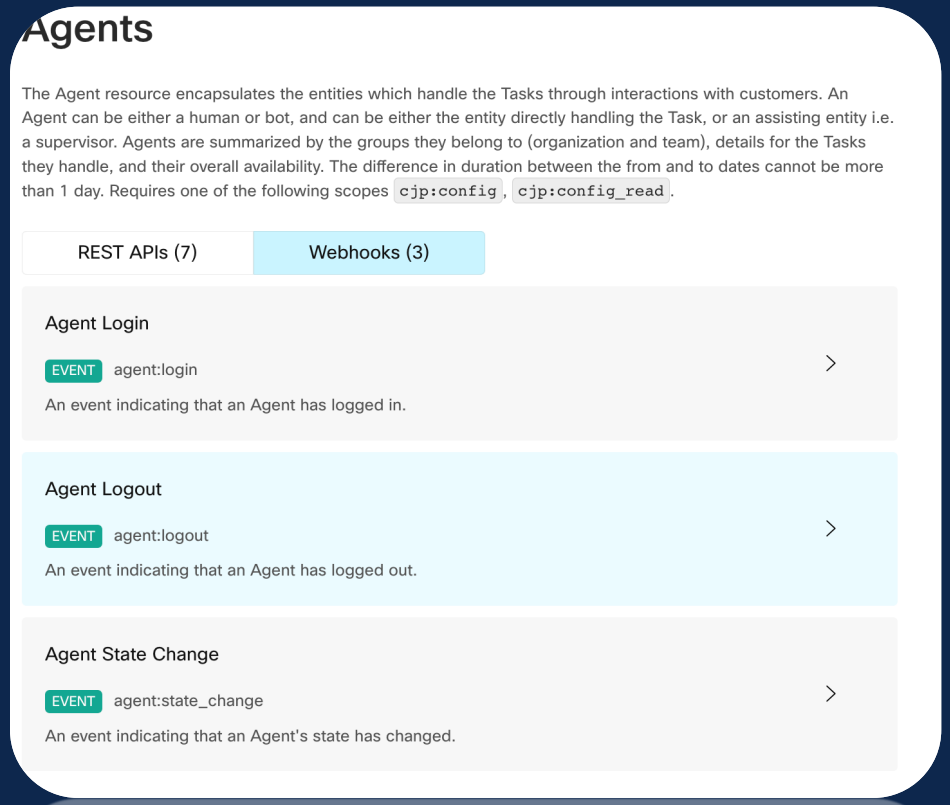
Reporting API

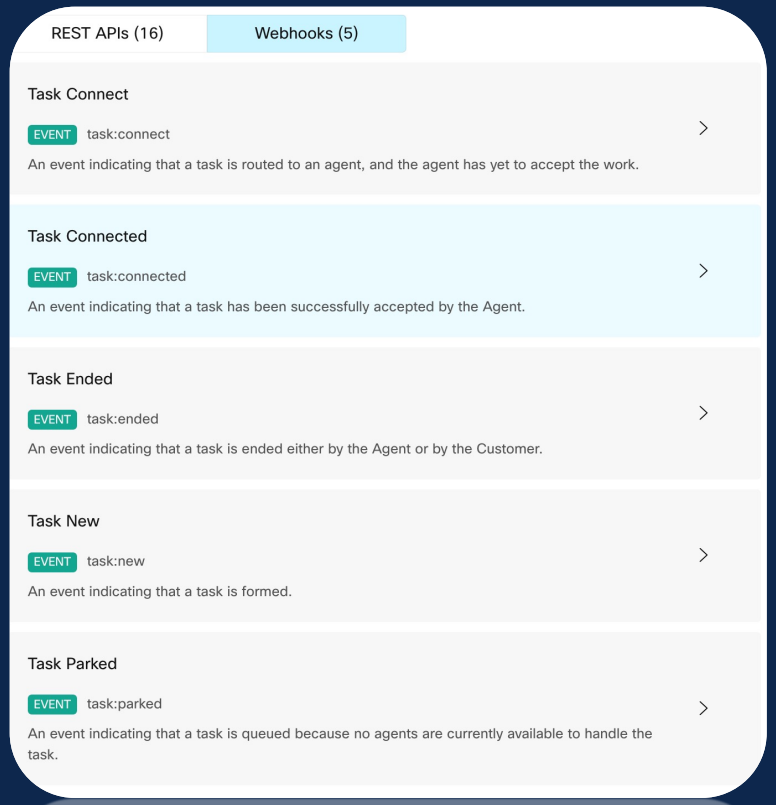
- GraphQL/Search.
- Raw data access to analyzer.
- Get only what you want.
- ANI, Agent.

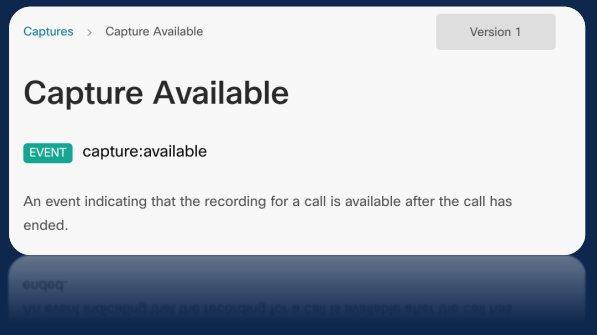


Webhooks

Subscribe to webhook in order to receive events

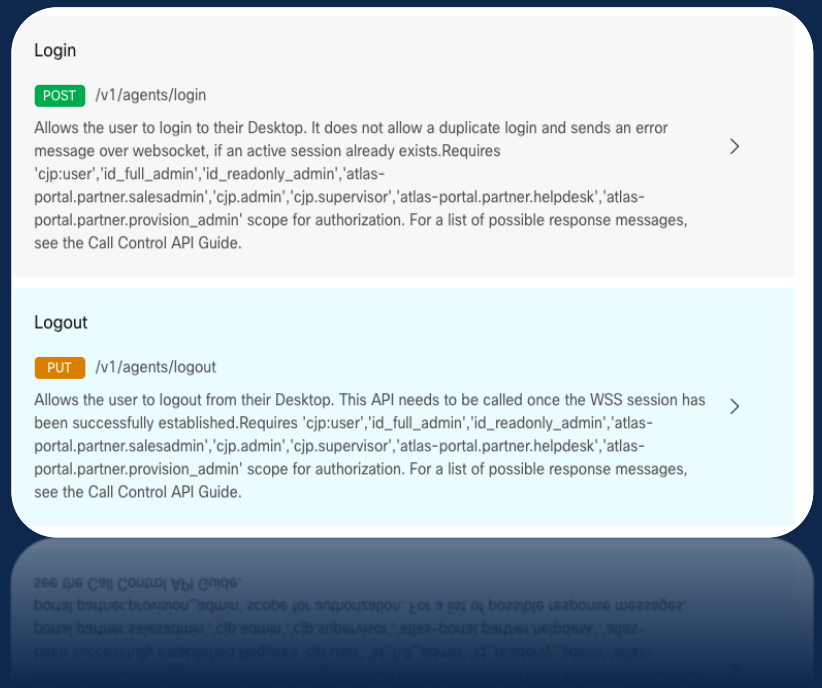


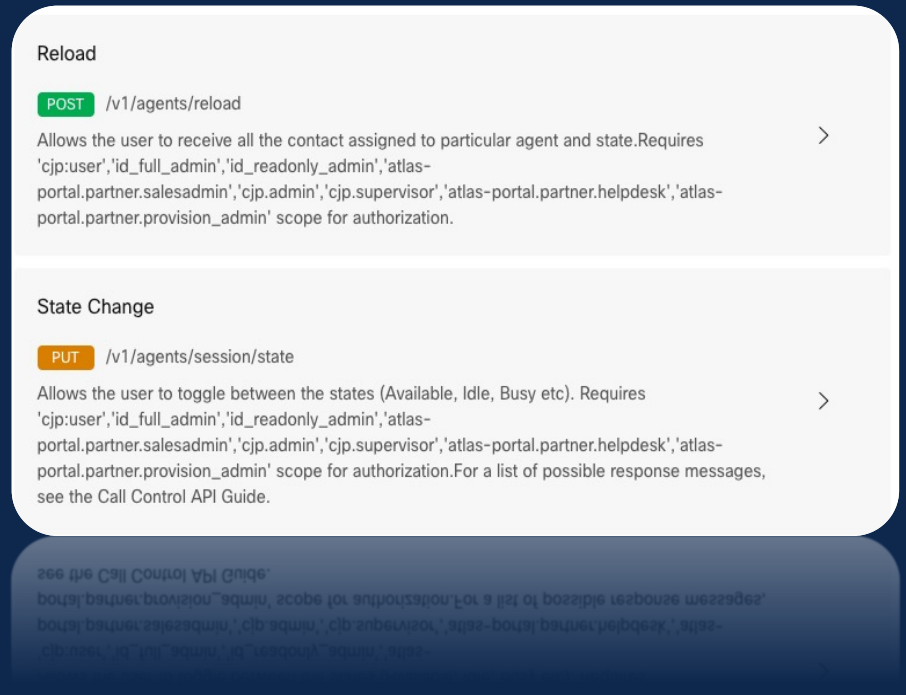




- Events from server to client
- One way, Opposite to APIs
- Subscribe

Agent APIs





- WebSocket.
- Subscribe to WebSocket to get URL <-> start listening.

Call Control APIs

Accept Task

POST /v1/tasks/{taskId}/accept

Access this Endpoint when the user has to accept either an Inbound or an outbound requests. The request can be a call, a chat or an email. Requires 'cjp:user' scope for authorization. For a list of possible response messages, see the Call Control API Guide.

Consult Task

POST /v1/tasks/{taskId}/consult

Access this Endpoint when the user has to consult a call or a chat to another user.Requires 'cjp:user' scope for authorization. For a list of possible response messages, see the Call Control API Guide.

Consult Accept Task

POST /v1/tasks/{taskId}/consult/accept

Access this Endpoint when the user has to accept a call to the consulting user. Requires 'cjp:user' scope for authorization. For a list of possible response messages, see the Call Control API Guide.

Consult Conference Task

POST /v1/tasks/{taskId}/consult/conference

Access this Endpoint when the user has to initiate a conference with the consulting user. Requires 'cjp:user' scope for authorization. For a list of possible response messages, see the Call Control API Guide.

Consult End Task

POST /v1/tasks/{taskId}/consult/end

Access this Endpoint when the user has to end a call with the consulting user. Requires 'cjp:user' scope for authorization. For a list of possible response messages, see the Call Control API Guide.

Consult Transfer Task

POST /v1/tasks/{taskId}/consult/transfer

Access this Endpoint when the user has to transfer a call to the consulting user. Requires 'cjp:user' scope for authorization. For a list of possible response messages, see the Call Control API Guide.

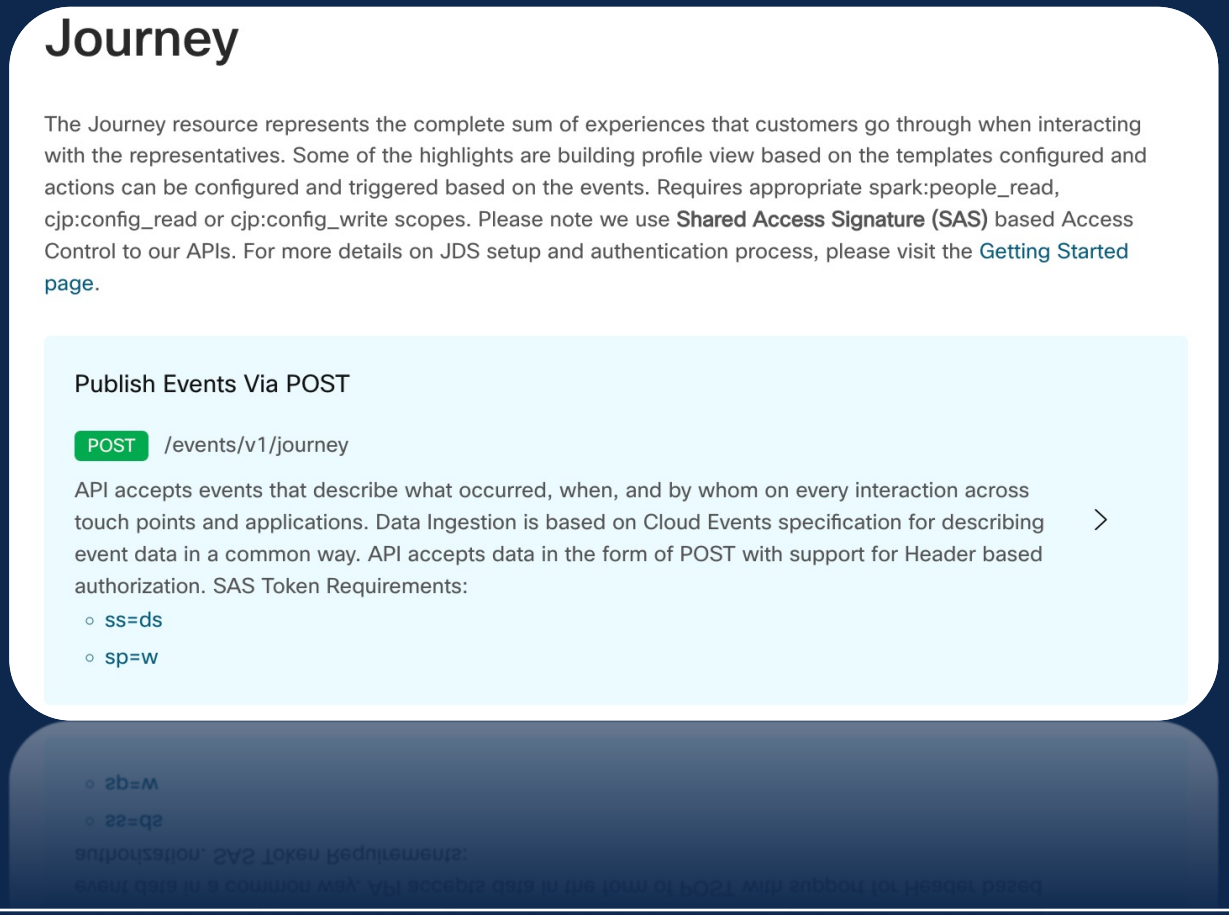
End Task

POST /v1/tasks/{taskId}/end

Access this Endpoint when the user has to end either an Inbound or an outbound requests. The request can be a call, a chat or an email. Requires 'cjp:user' scope for authorization. For a list of possible response messages, see the Call Control API Guide.

Journey APIs

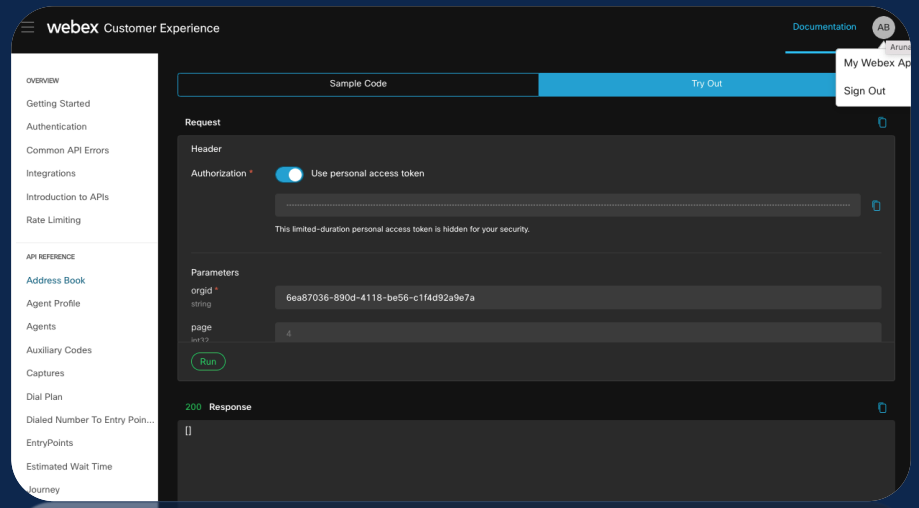
- Store/Retrieve customer data.
- Configure JDS Objects.
- Store journey events.
- Retrieve journey events (historic/Realtime).



02
Developer Portal

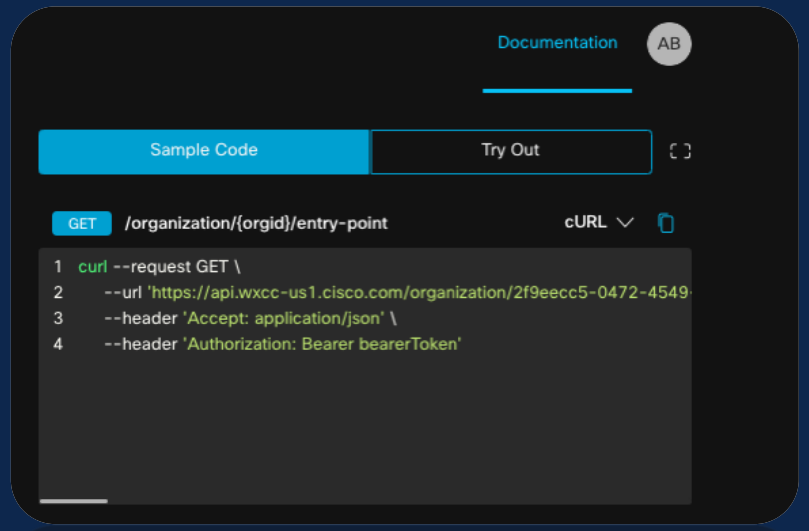
developer.webex-cx.com

Global, single, unified URL, New features



Try it out – on developer portal, defaults to the Region

Region	Developer Portal
US,ANZ, UK, EU	https://developer.webex-cx.com/

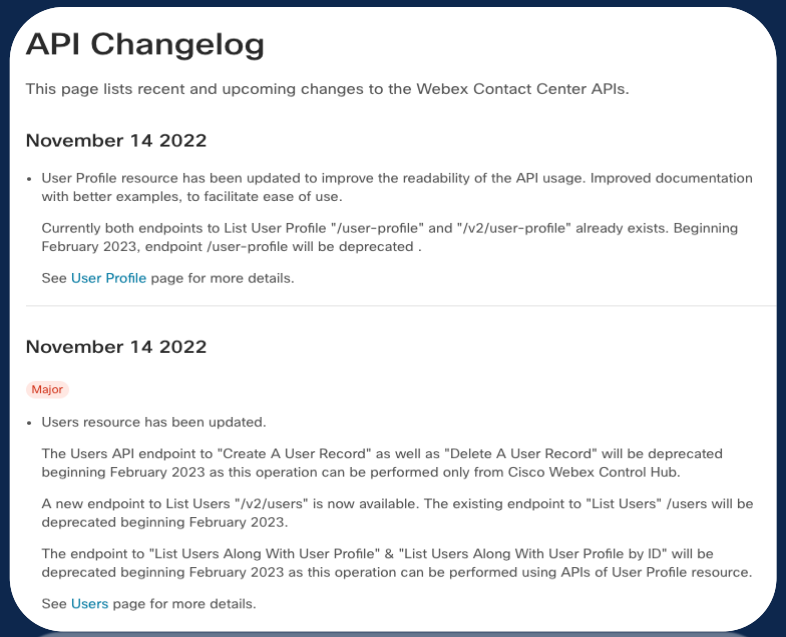


API Root URLs – Ensure you're using the right region's URI

Region	API Root URL
US	https://api.wxcc-us1.cisco.com
ANZ	https://api.wxcc-anz1.cisco.com
UK	https://api.wxcc-eu1.cisco.com
EU (Frankfurt)	https://api.wxcc-eu2.cisco.com

Change log, New features

New	Newly released API resources, endpoints, or properties
Warning	API behavior change, usually related to unintended usage of an endpoint or because of unexpected behavior
Deprecation	API resource, endpoint, or property is still available but is no longer recommended to use and won't be updated further.
Major	Change which may effect specific use cases (like a more specific breaking change)
Breaking	Change which will affect all developers, regardless of API usage



API Changelog

This page lists recent and upcoming changes to the Webex Contact Center APIs.

November 14 2022

- User Profile resource has been updated to improve the readability of the API usage. Improved documentation with better examples, to facilitate ease of use.

  Currently both endpoints to List User Profile "/user-profile" and "/v2/user-profile" already exists. Beginning February 2023, endpoint /user-profile will be deprecated .

  See User Profile page for more details.

November 14 2022

Major

- Users resource has been updated.

  The Users API endpoint to "Create A User Record" as well as "Delete A User Record" will be deprecated beginning February 2023 as this operation can be performed only from Cisco Webex Control Hub.

  A new endpoint to List Users "/v2/users" is now available. The existing endpoint to "List Users" /users will be deprecated beginning February 2023.

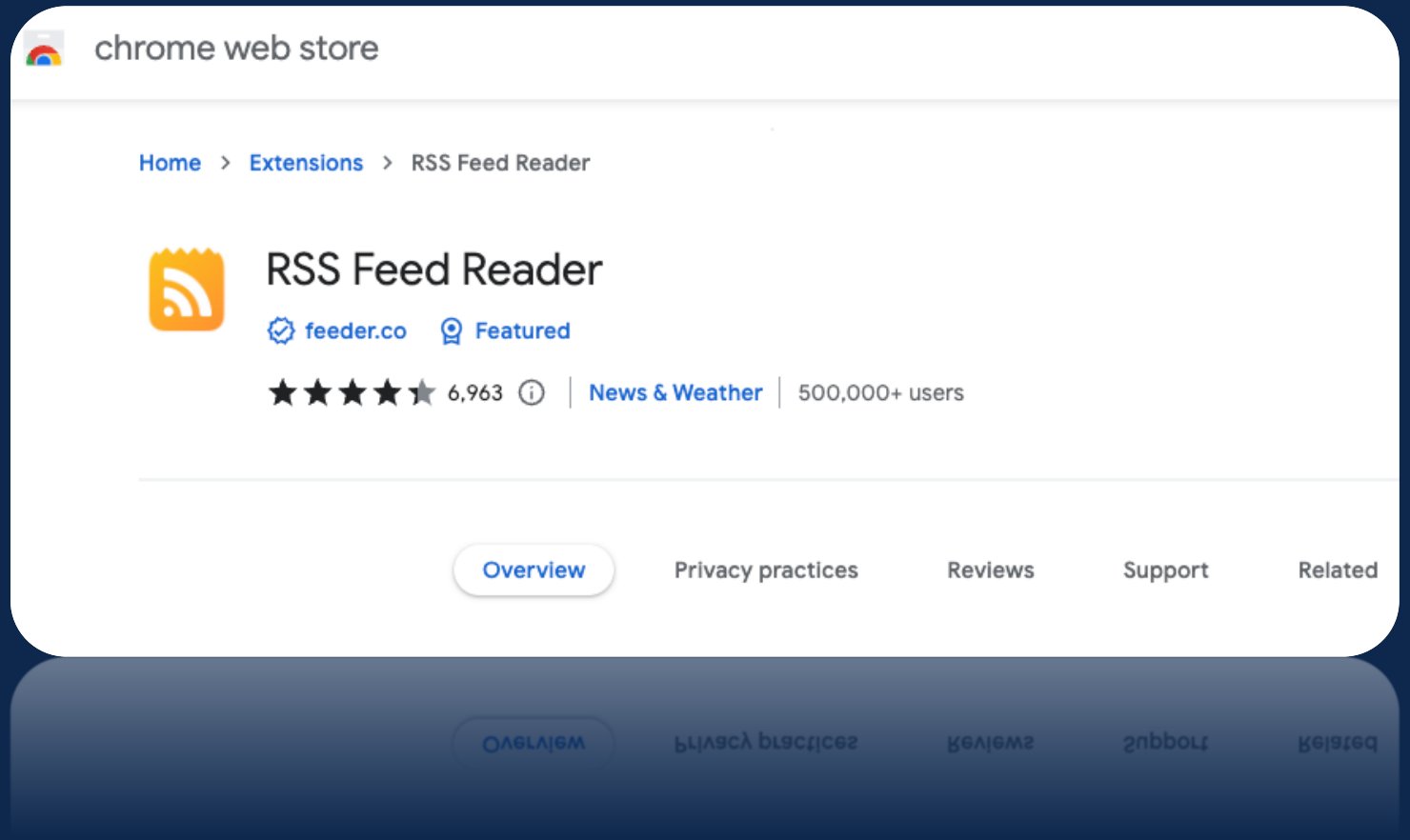
  The endpoint to "List Users Along With User Profile" & "List Users Along With User Profile by ID" will be deprecated beginning February 2023 as this operation can be performed using APIs of User Profile resource.

  See Users page for more details.

- Changes Portal is going through.
- NEW API being released
- CHANGES to existing API to avoid breakage.
- Information on GUIDES and DOC changes.
- GitHub SAMPLES

Change log, New features

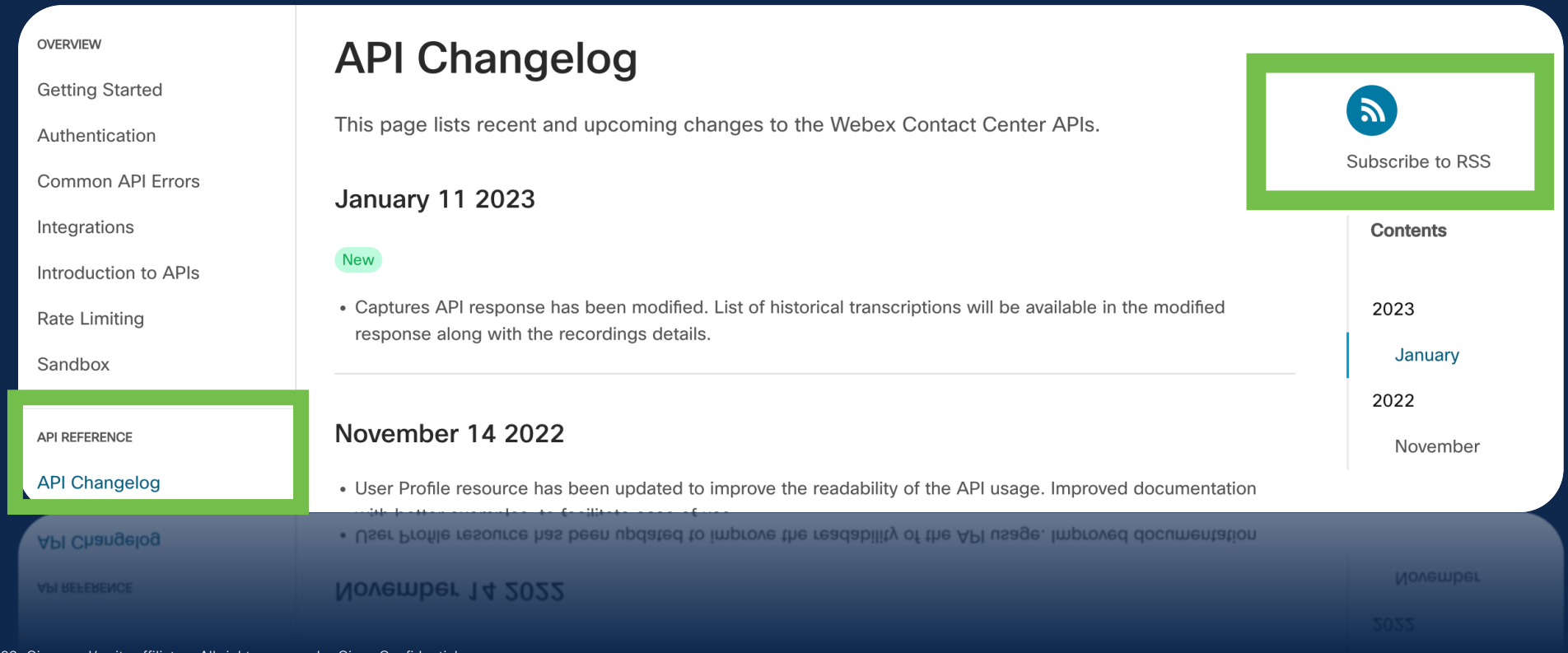
1) Install chrome extension



Change log, New features

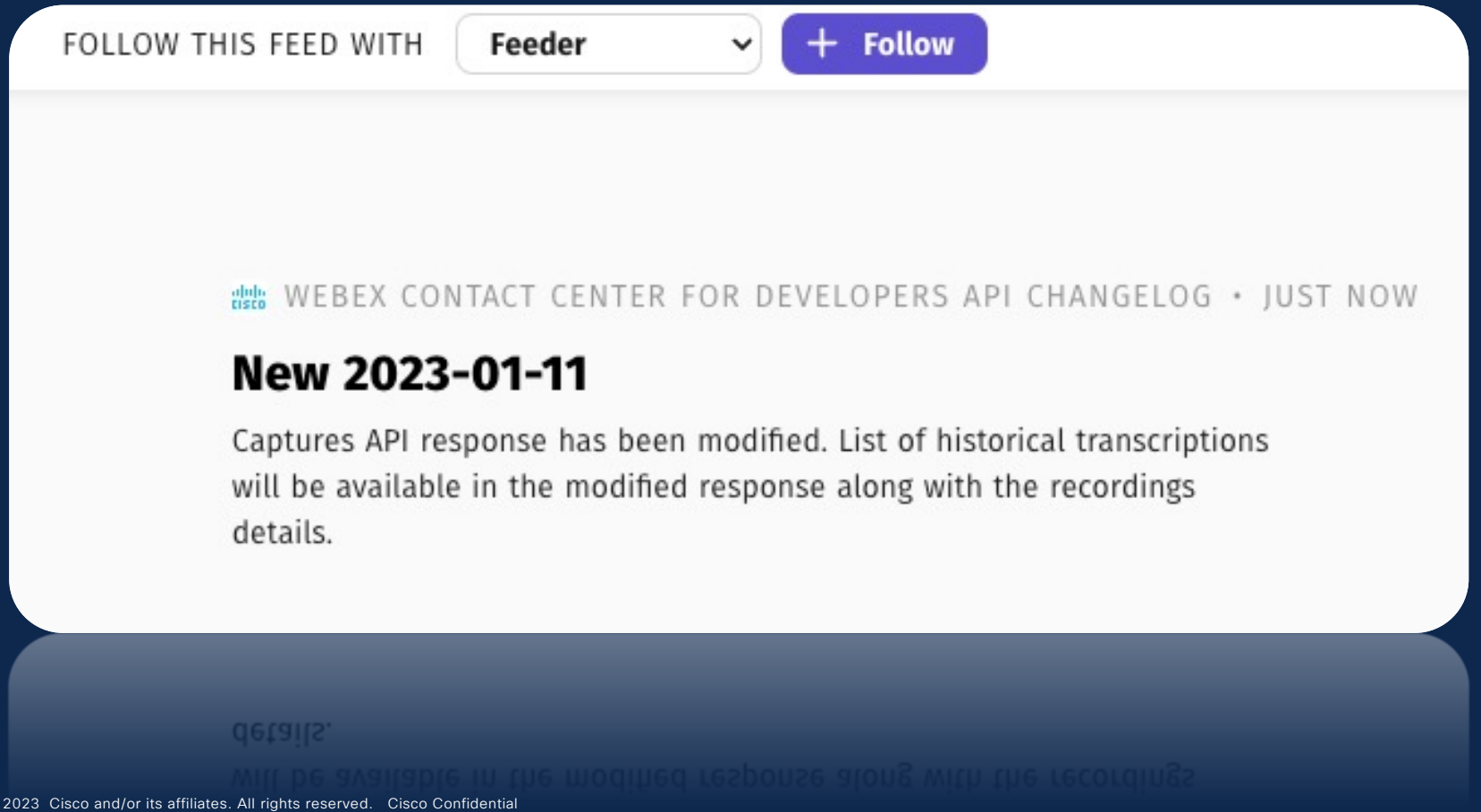
2) Go to https://developer.webex-cx.com/documentation/api-changelog

Click on RSS feed.



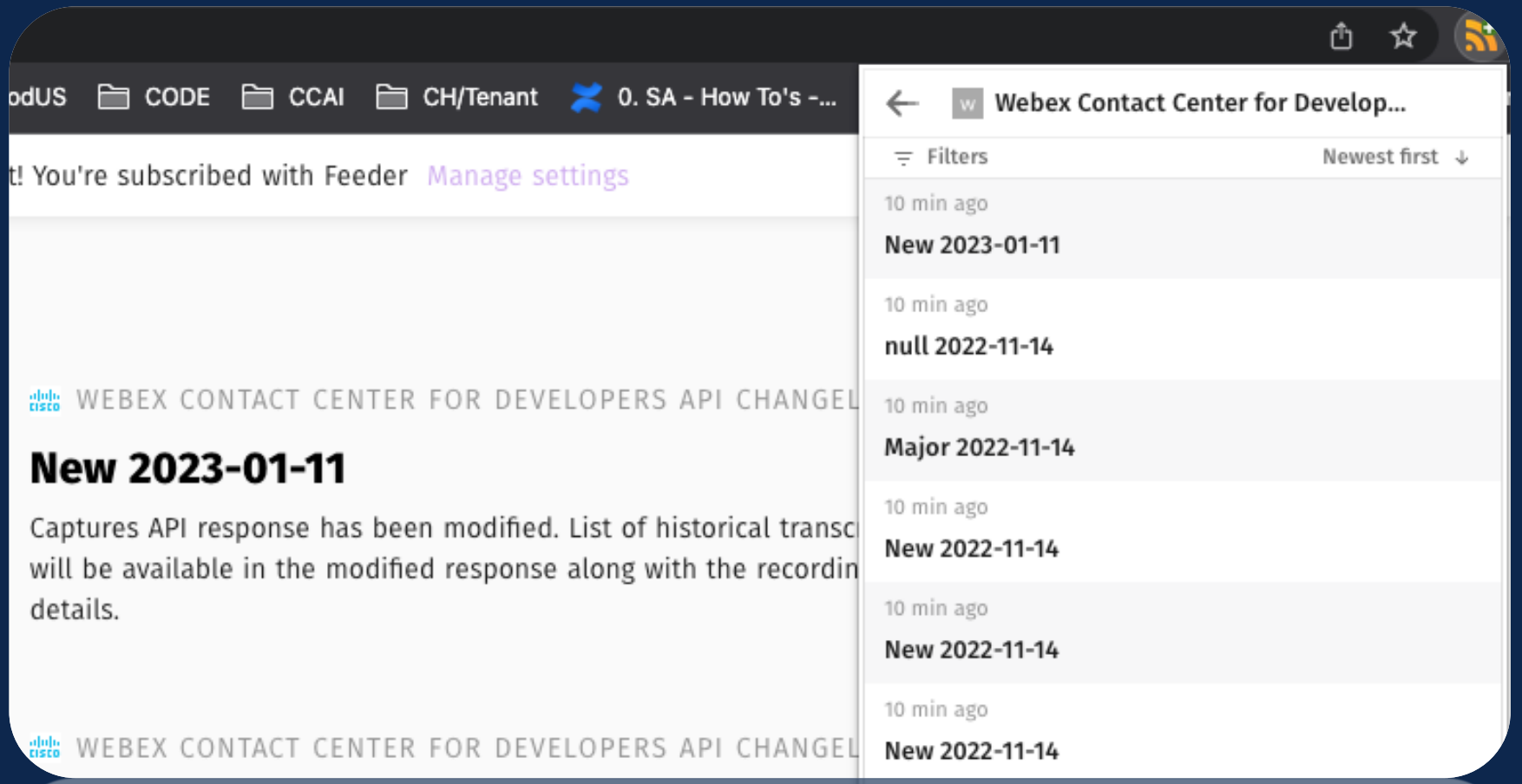
Change log, New features

3) New tab will open using chrome extension.
Click follow.



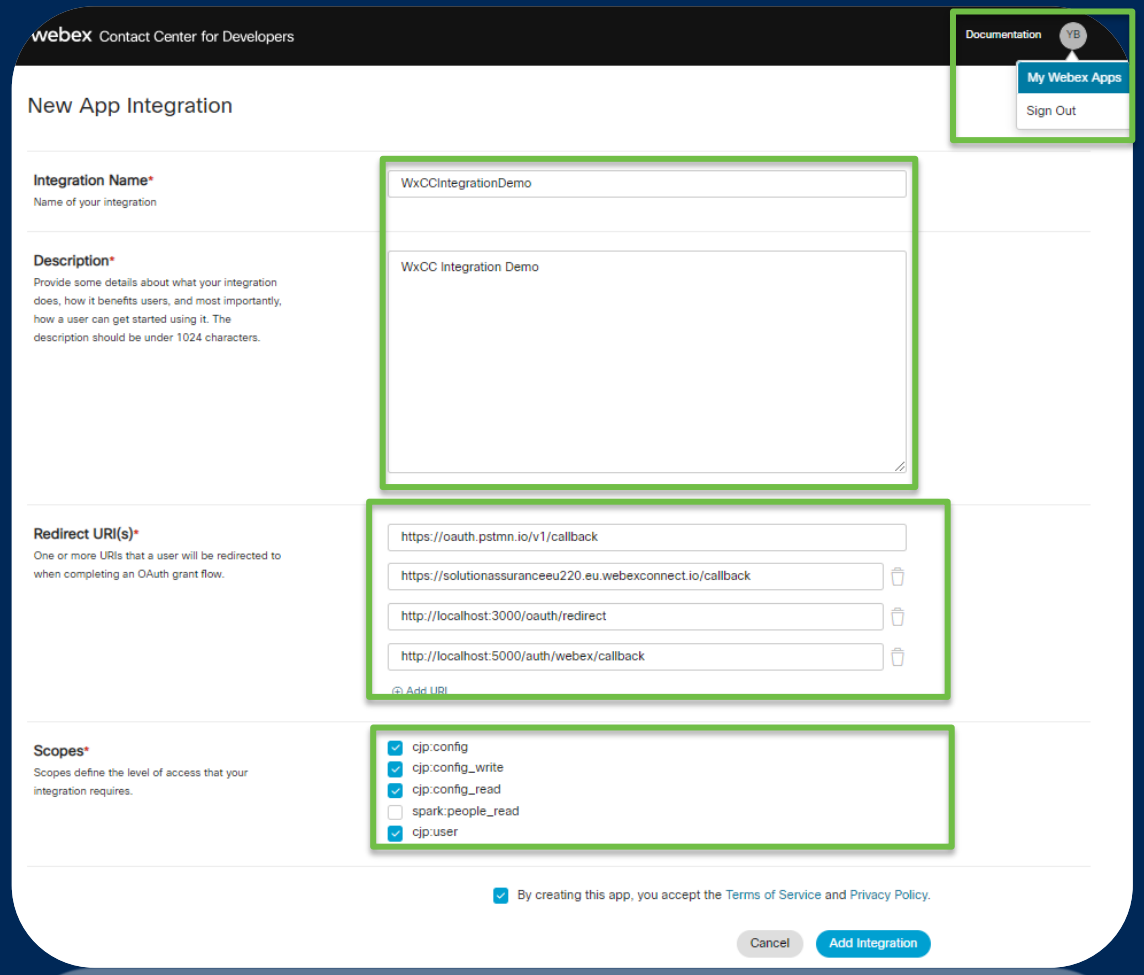
Change log, New features

4) Any updates on change log will be sent to your browser.

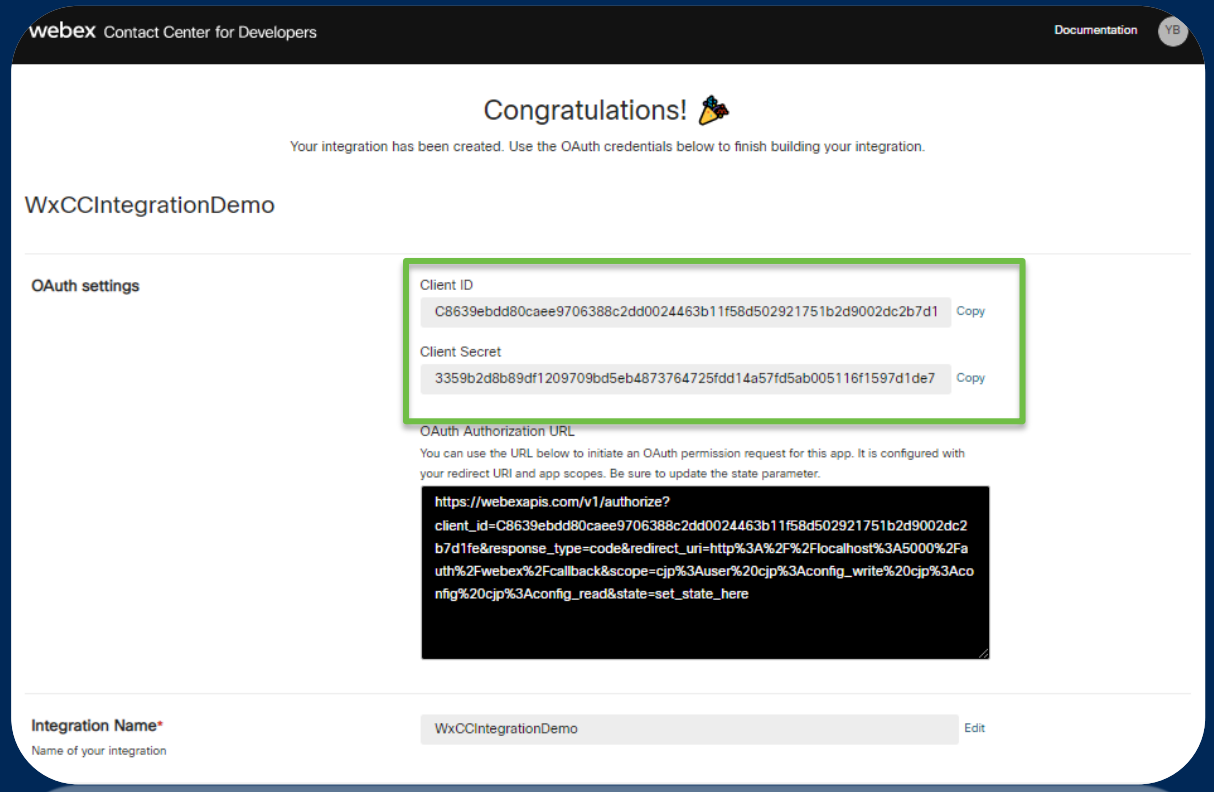


03
Understanding OAuth2

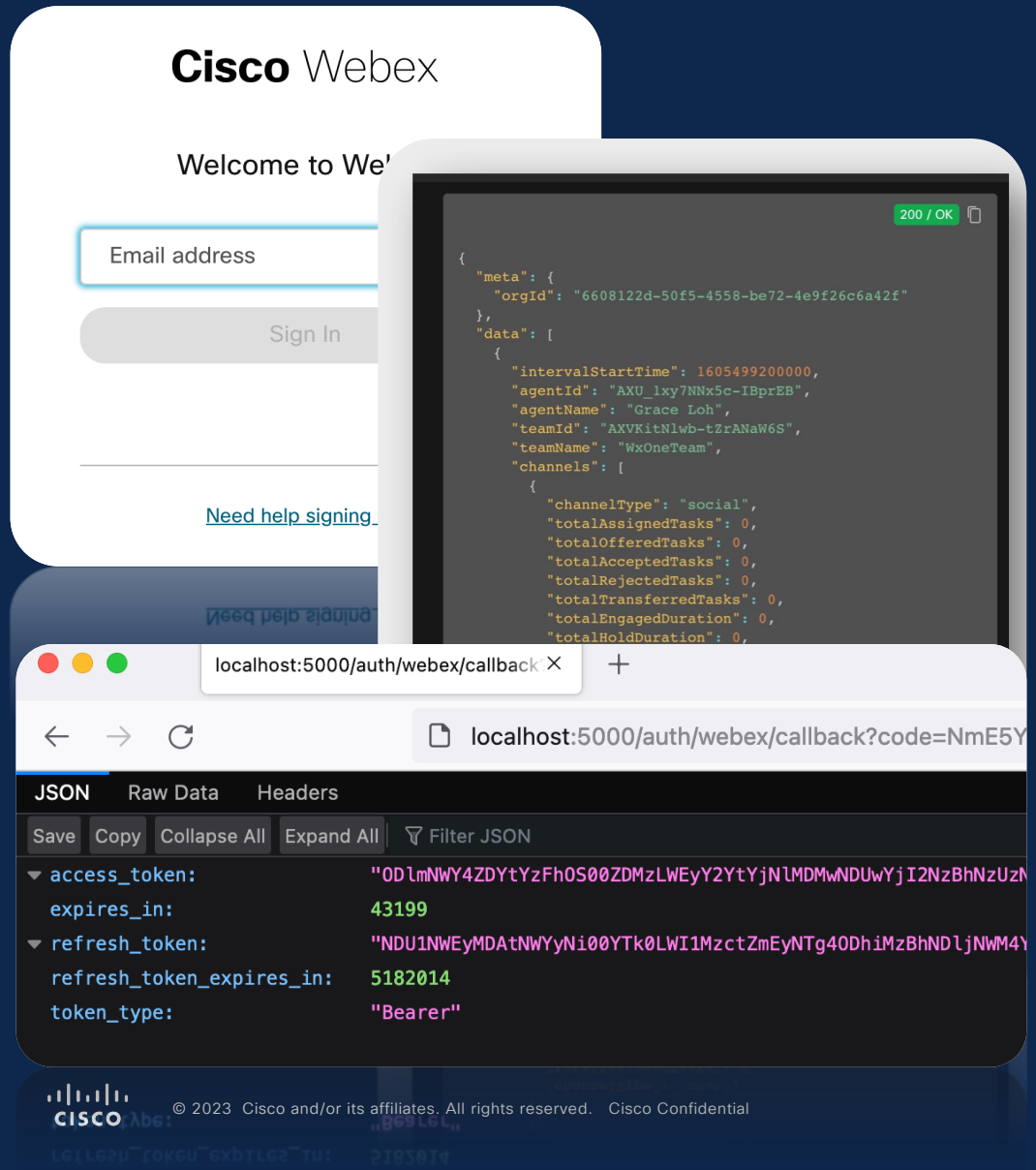
Create Your App



- Building secure connection to your App
- One integration for multiple Apps



Webex OAuth2 Mechanism – Overview



- Create an integration to obtain clientid and client secret.
- Once you obtain an Org's **access_token** and **refresh_token**, you can access their data.
- Use the **refresh_token** before expiry (12 hours) to get another access_token

OAuth2 Access Token Flow

Webex Common Identity (Login with Webex)

- User Profiles and Scopes apply.
- You need a Contact Center Administrator Role to Read and Write to all endpoints.

Your Application

Create integration

The User first Reaches Your App

https://your-app.com

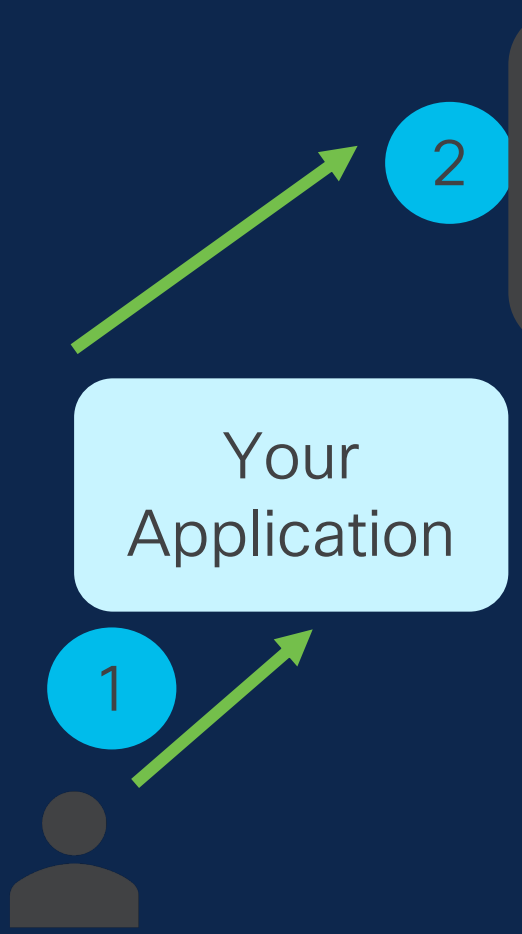
Webex Contact Center APIs

api.wxcc-us1.cisco.com/v1/tasks

api.wxcc-us1.cisco.com/v1/captures

api.wxcc-us1.cisco.com/v1/search

OAuth2 Access Token Flow



Your App then redirects the user to Webex With the Client ID, Scopes needed

config config_read config_write

Webex Contact Center APIs

api.wxcc-us1.cisco.com/v1/tasks

api.wxcc-us1.cisco.com/v1/captures

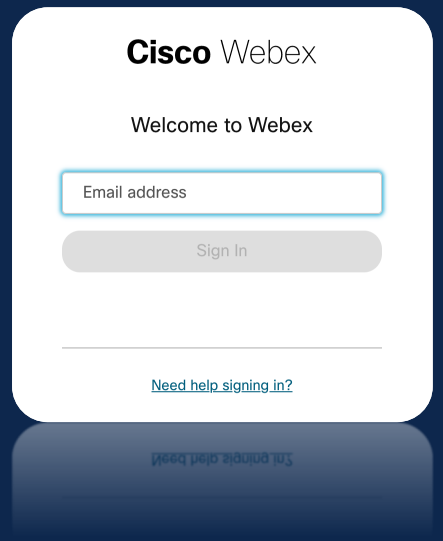
api.wxcc-us1.cisco.com/v1/search

OAuth2 Access Token Flow

Webex Common Identity (Login with Webex)

Your Application

3



User is Greeted with the “Webex Login” Screen

Signs in!

Webex Contact Center APIs
api.wxcc-us1.cisco.com/v1/tasks
api.wxcc-us1.cisco.com/v1/captures
api.wxcc-us1.cisco.com/v1/search

OAuth2 Access Token Flow

Webex Common Identity (Login with Webex)

Your Application

4

Webex Redirects back to the “Redirect URI” of your App with the ”Authorization Code”

Webex Contact Center APIs

api.wxcc-us1.cisco.com/v1/tasks

api.wxcc-us1.cisco.com/v1/captures

api.wxcc-us1.cisco.com/v1/search

OAuth2 Access Token Flow

Webex Common Identity (Login with Webex)

Your Application

Your App reaches out to Webex to Request an Access Token

Webex Contact Center APIs

api.wxcc-us1.cisco.com/v1/tasks

api.wxcc-us1.cisco.com/v1/captures

api.wxcc-us1.cisco.com/v1/search

OAuth2 Access Token Flow

Webex Common Identity (Login with Webex)

Your Application

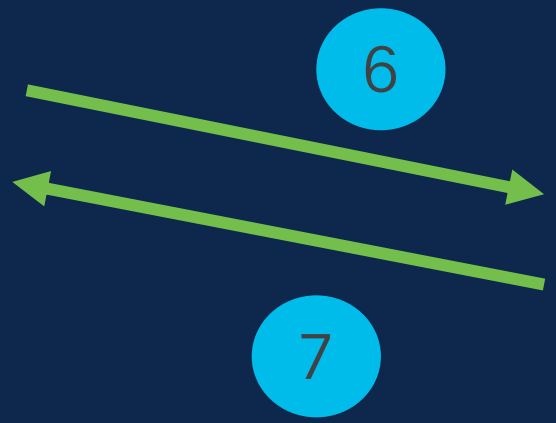
Webex Contact Center APIs
api.wxcc-us1.cisco.com/v1/tasks
api.wxcc-us1.cisco.com/v1/captures
api.wxcc-us1.cisco.com/v1/search

Your App uses the access_token to access the Webex CC APIs

OAuth2 Access Token Flow

Webex Common Identity (Login with Webex)

Your Application



Webex Contact Center APIs

api.wxcc-us1.cisco.com/v1/tasks

api.wxcc-us1.cisco.com/v1/captures

api.wxcc-us1.cisco.com/v1/search

Data is returned to the App,

as long as the **access_token** is valid

OAuth2 Access Token Flow

Webex Common Identity (Login with Webex)

Before the refresh_token expires, the app can continue to ask for access_tokens in the background.

Your Application

Webex Contact Center APIs
api.wxcc-us1.cisco.com/v1/tasks
api.wxcc-us1.cisco.com/v1/captures
api.wxcc-us1.cisco.com/v1/search

8

The user enjoys the additional capabilities provided by your Application

04
Use Cases

Callback from website

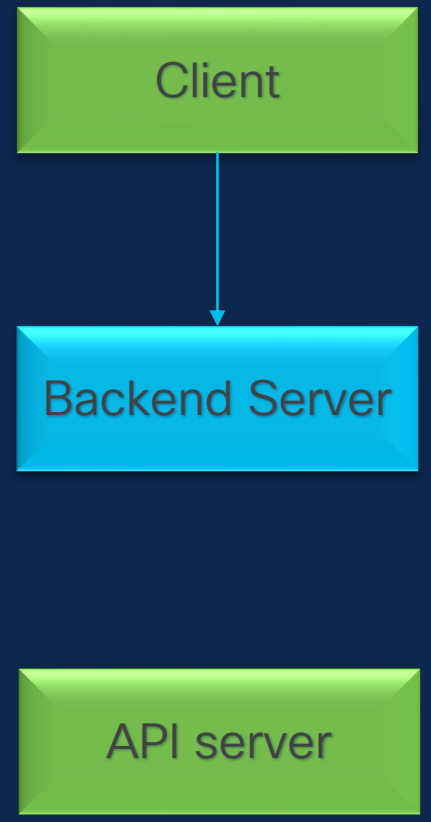
Use case – Callback from website, how ?

1) Client: The website where the user is making the request from.

Use case – Callback from website, how ?

1) Client: The website where the user is making the request from.

2) POST request from client to backend server.

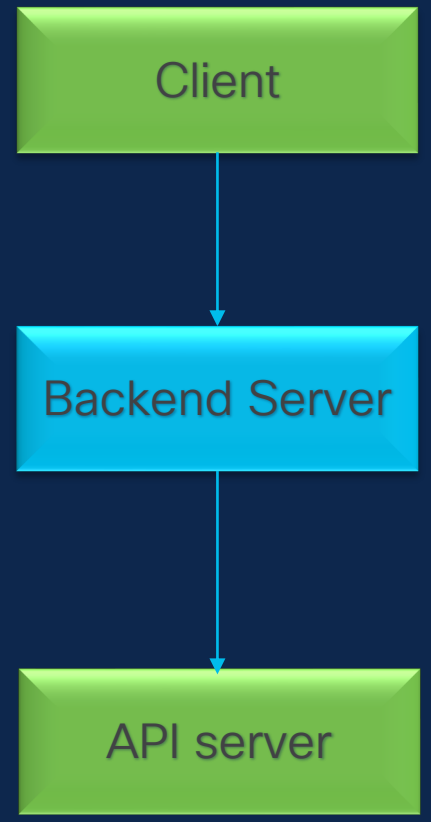


Use case – Callback from website, how ?

1) Client: The website where the user is making the request from.

2) POST request from client to backend server.

3) Backend server - To receive the POST request.



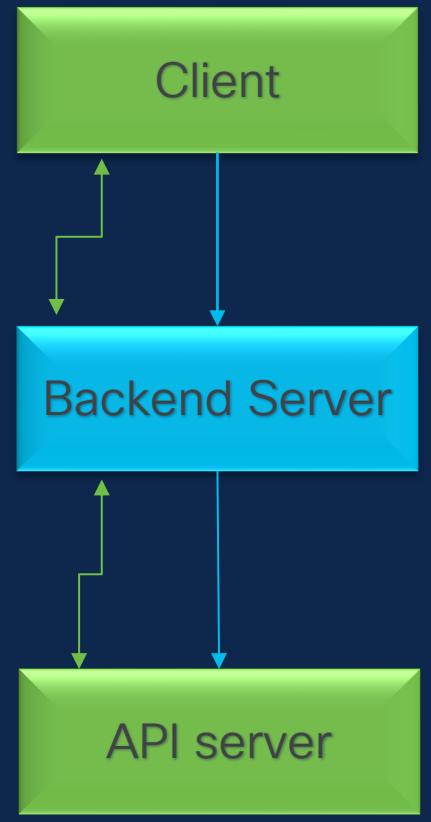
Use case – Callback from website, how ?

1) Client: The website where the user is making the request from.

2) POST request from client to backend server.

3) Backend server - To receive the POST request.

4) Backend server - Call Webex CC **Task API** to initiate callback request.
   Wait for response code and continue accordingly. (CONTRACT)
   POST Parameters - **Outbound Type – Callback, EP ID – Out Bound.**



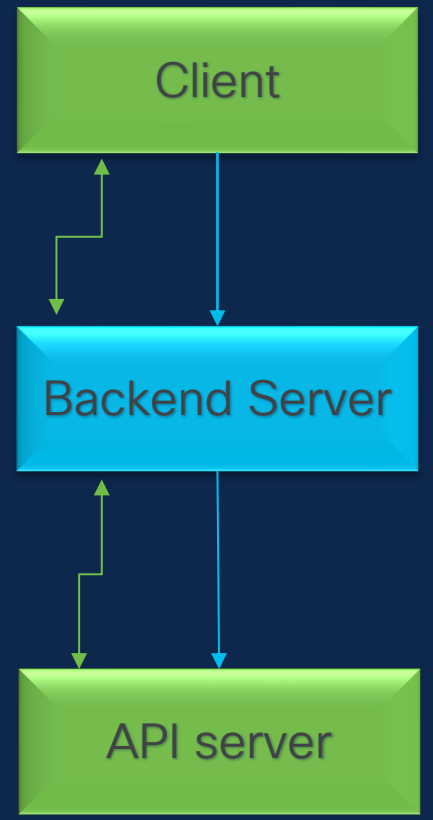
Use case – Callback from website, how ?

1) Client: The website where the user is making the request from.

2) POST request from client to backend server.

3) Backend server - To receive the POST request.

4) Backend server - Call Webex CC Task API to initiate callback request.
   Wait for response code and continue accordingly.
   POST Parameters - **Outbound Type – Callback, EP ID – Out Bound.**



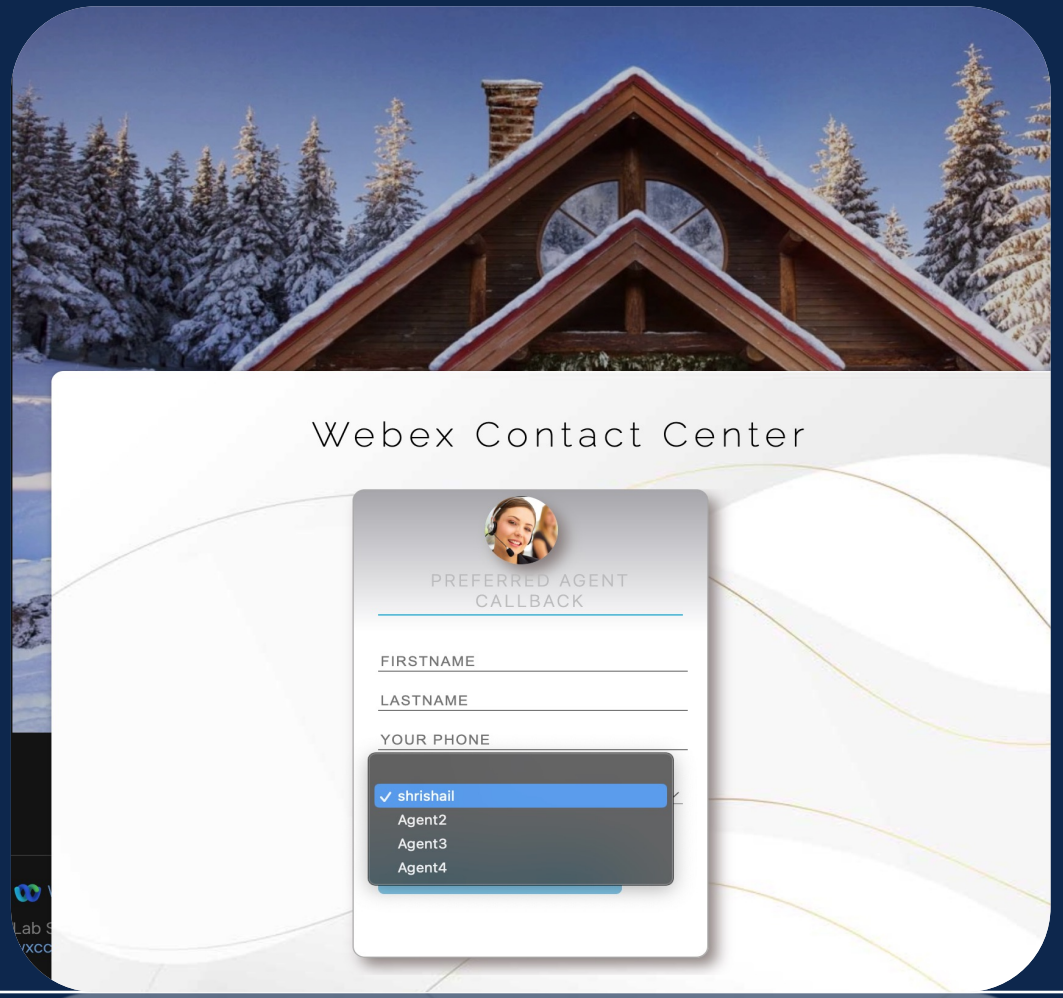
Preferred Agent Callback.

Use case – Preferred agent callback, how?

- Client: Browser, Request from website.
- Form - Programmatically pull agent name.
- Submit request.
- Initiate a POST request to **Task API** with **outboundType as 'EXECUTE_FLOW' and EP ID of the inbound entry point to trigger Flow.**
- Flow with QtoAgent node and a flow variable that will receive agent email ID.

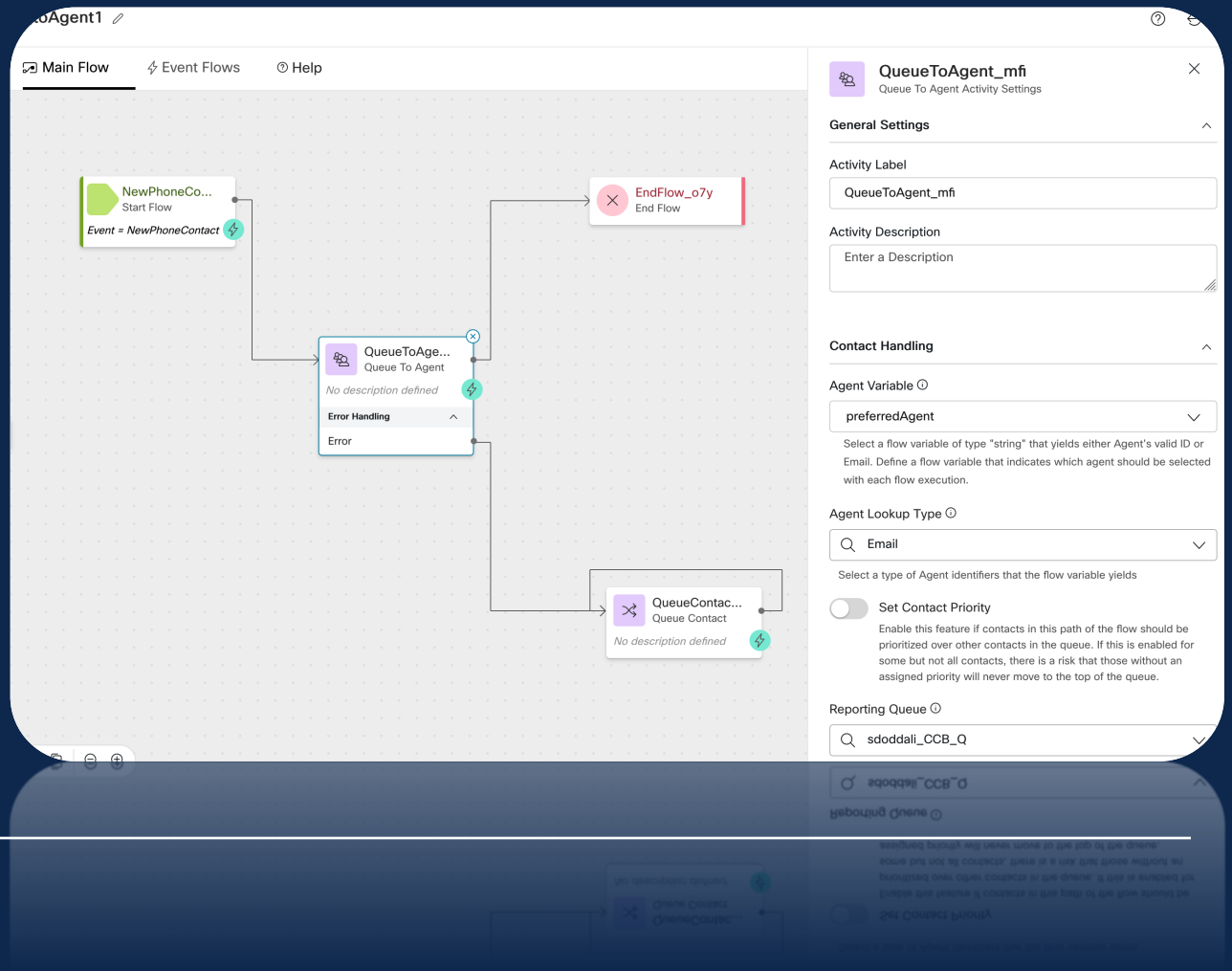
Use case – Preferred agent callback, how?

- Client: Browser, Request from website.
- Form - Programmatically pull agent name.
- Submit request.
- Initiate a POST request to **Task API** with **outboundType as 'EXECUTE_FLOW' and EP ID of the inbound entry point to trigger Flow.**
- Flow with QtoAgent node and a flow variable that will receive agent email ID.



Use case – Preferred agent callback, how?

- Client: Browser, Request from website.
- Form - Programmatically pull agent name.
- Submit request.
- Initiate a POST request to **Task API** with **outboundType as 'EXECUTE_FLOW' and EP ID of the inbound entry point to trigger Flow.**
- Flow with QtoAgent node and a flow variable that will receive agent email ID.



Play Emergency Message

Use case – Play emergency message.

Consider a scenario where there is an emergency and you must play an emergency message to the callers.

Use case – Play emergency message.

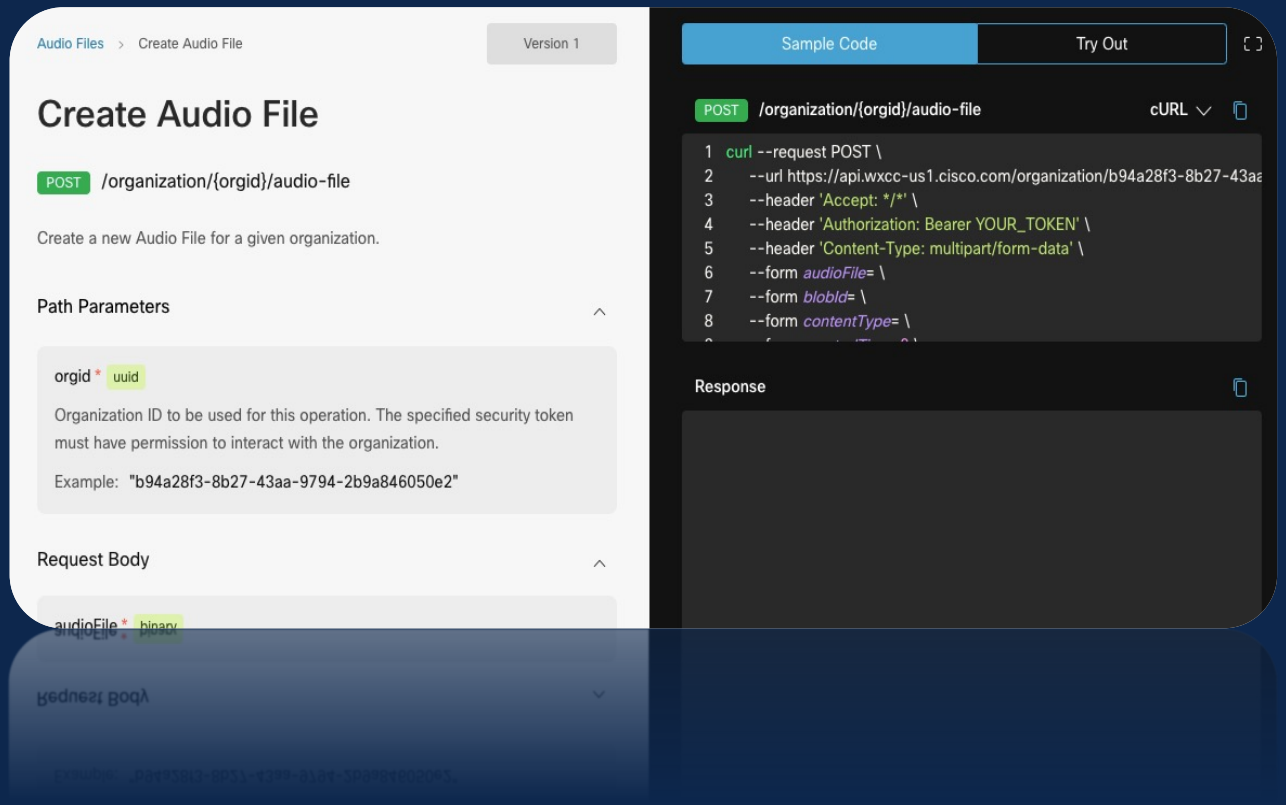
Question - How many contact center APIs are being used for this demo ?

Use case – Play emergency message.

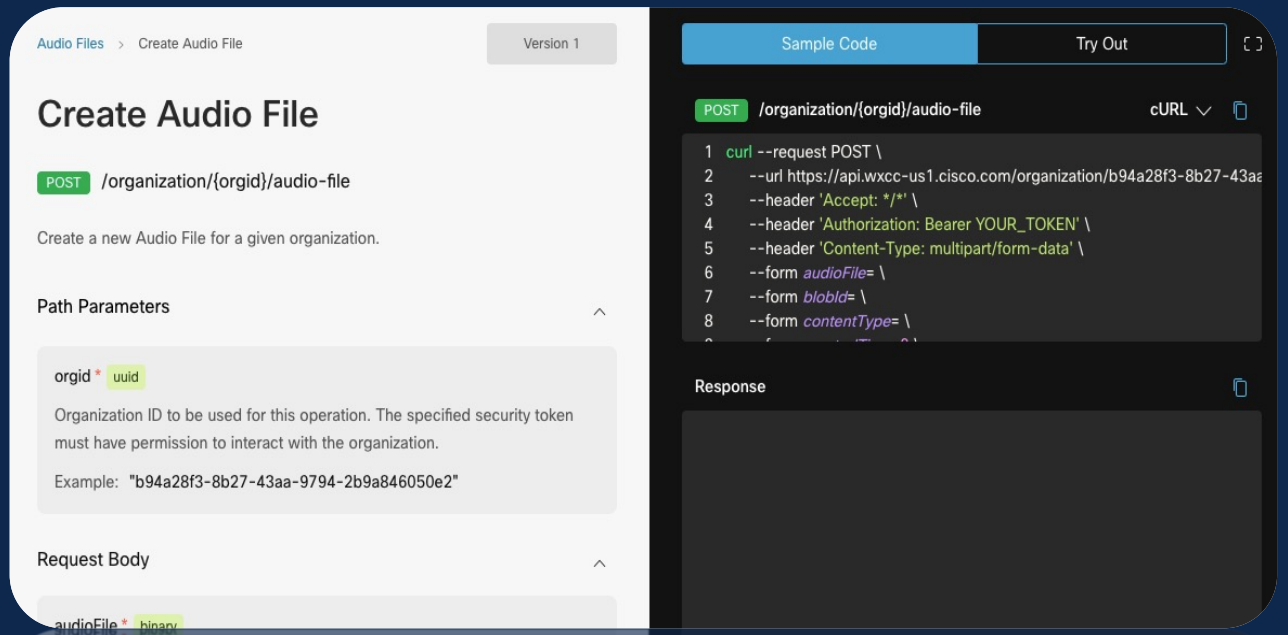
Question - How many contact center APIs are being used for this demo ?

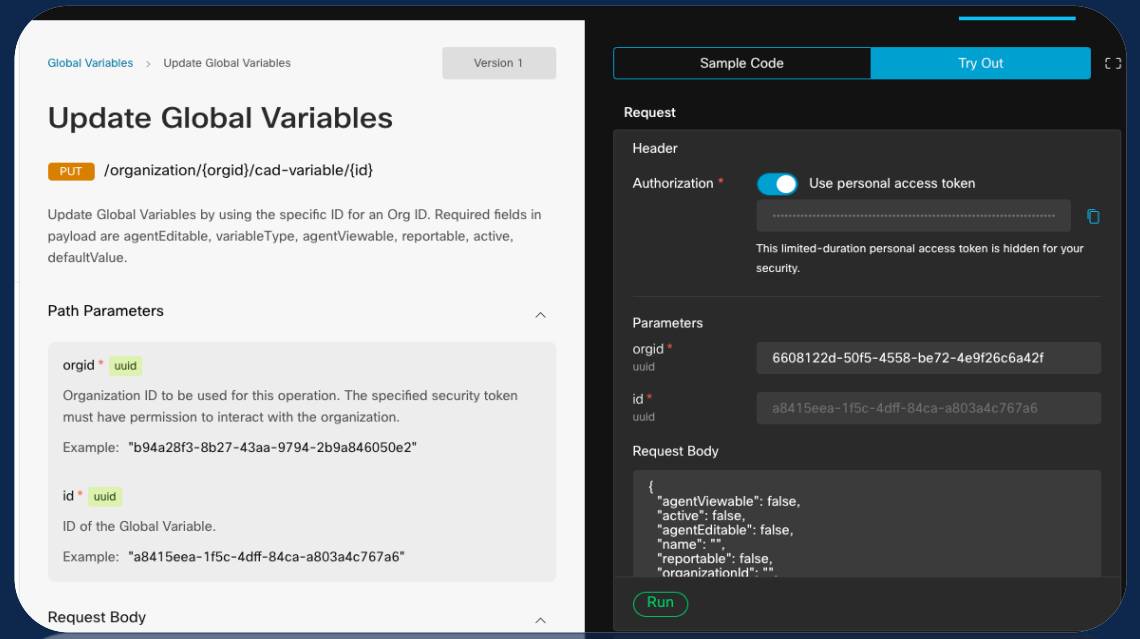
Answer – 2.

Use case – Play emergency message.

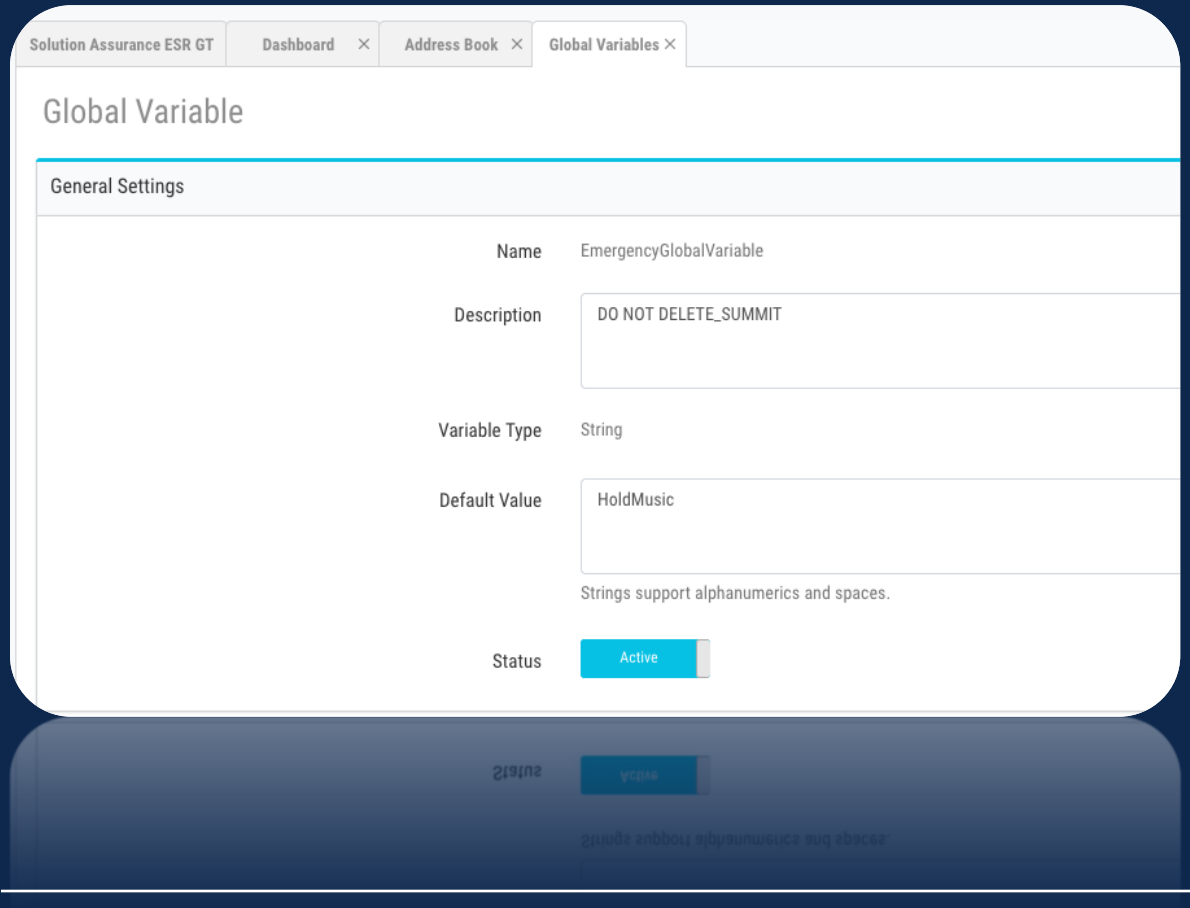


Use case – Play emergency message.

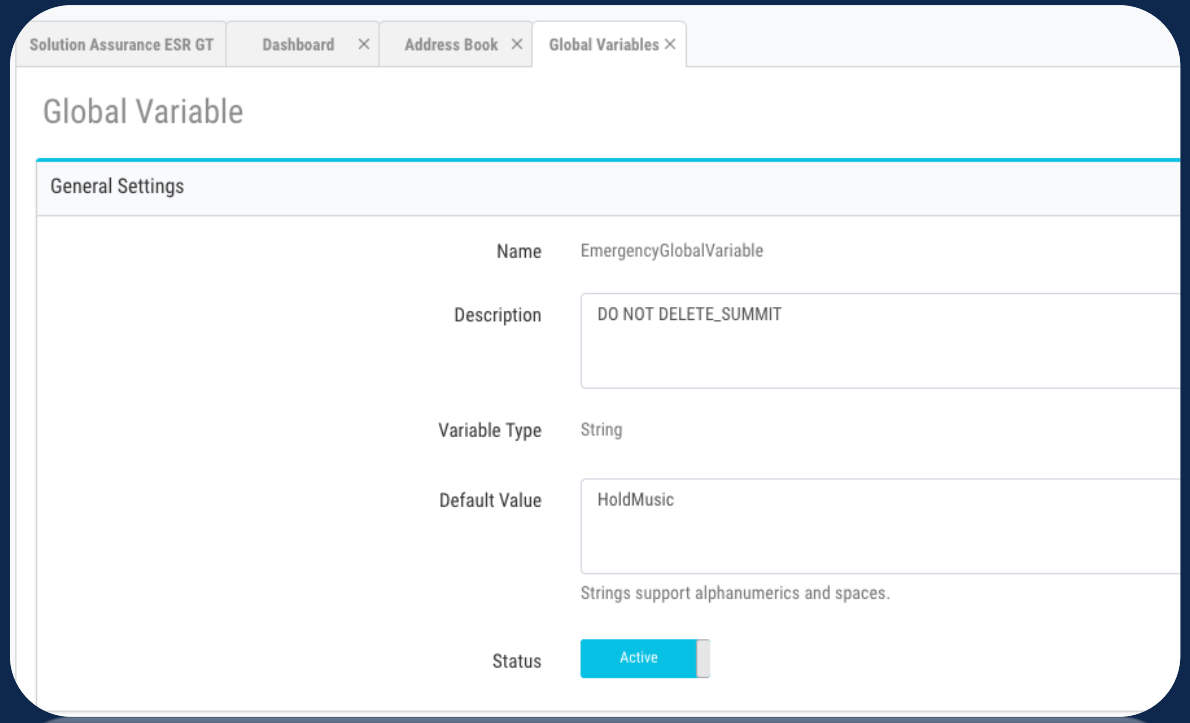


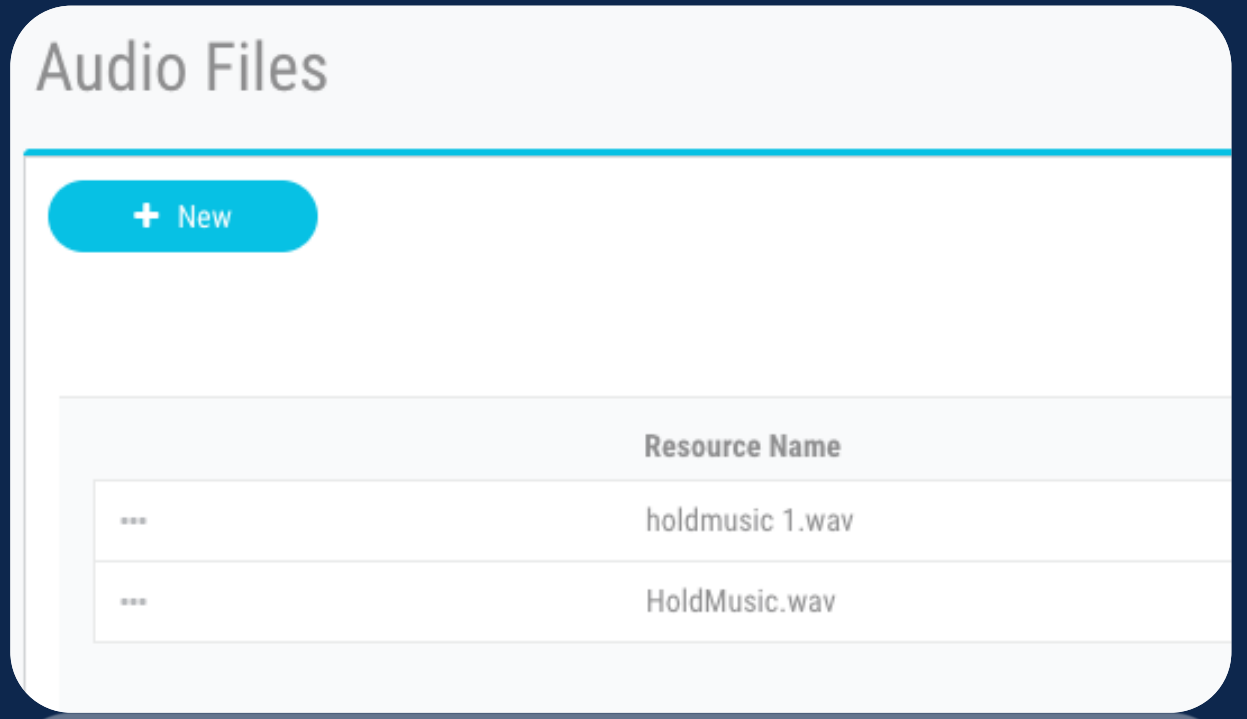


Use case – Play emergency message.

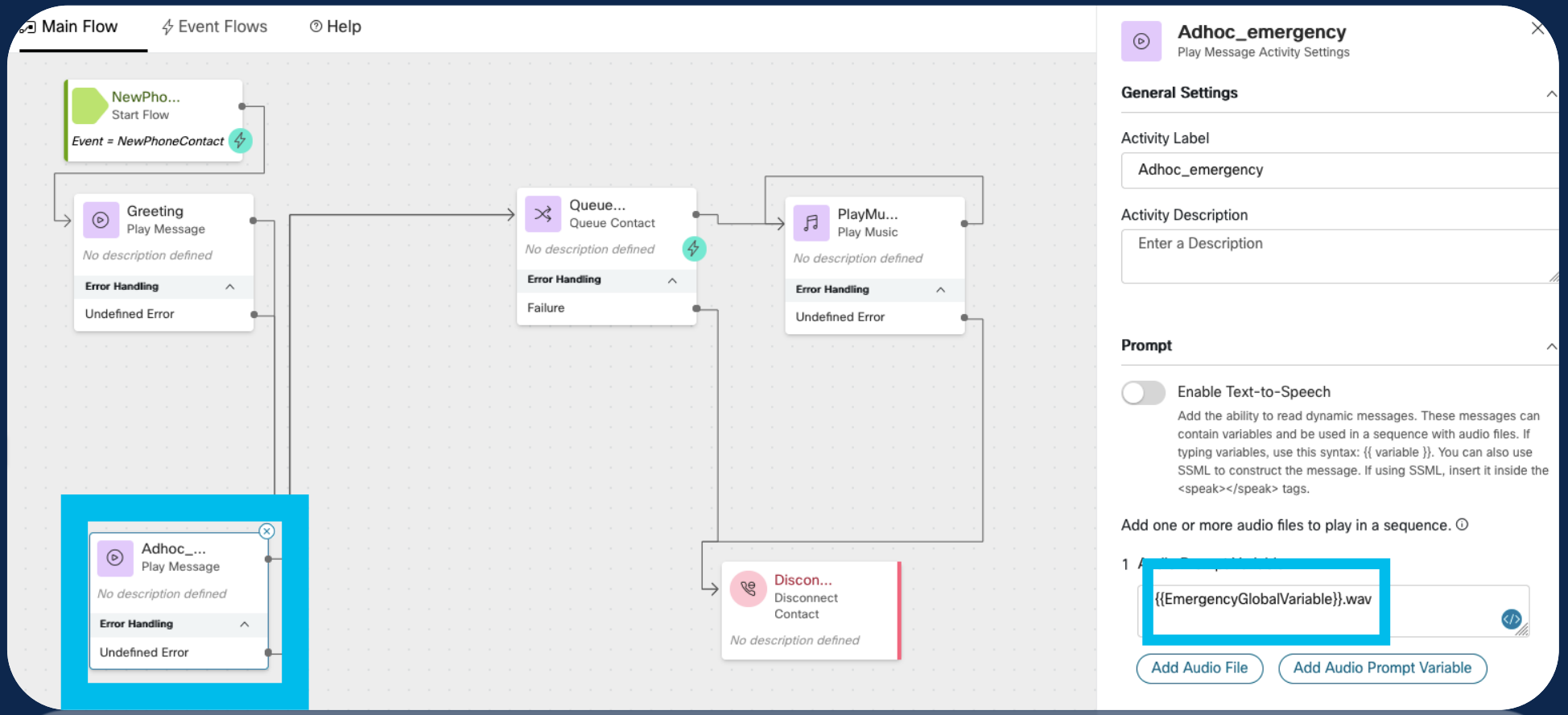


Use case – Play emergency message.

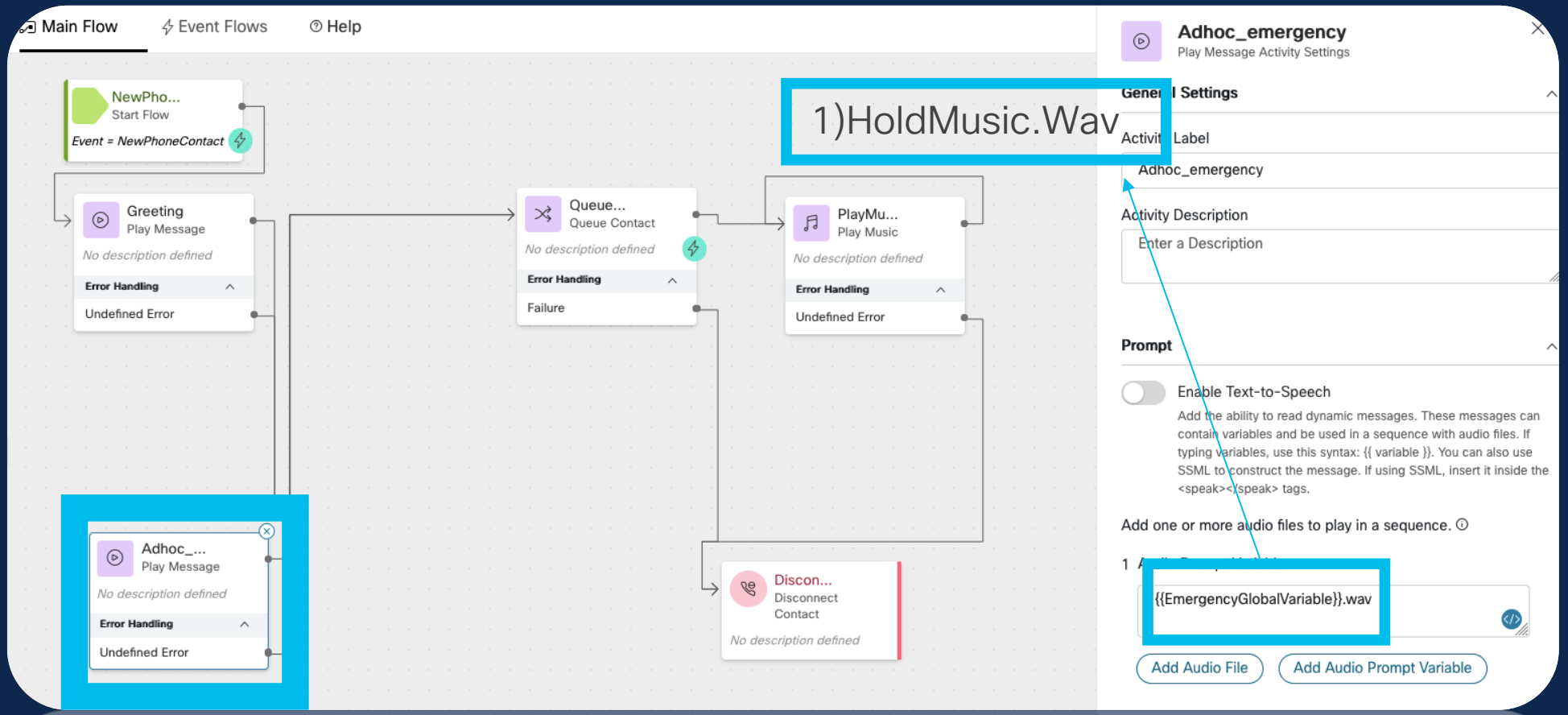




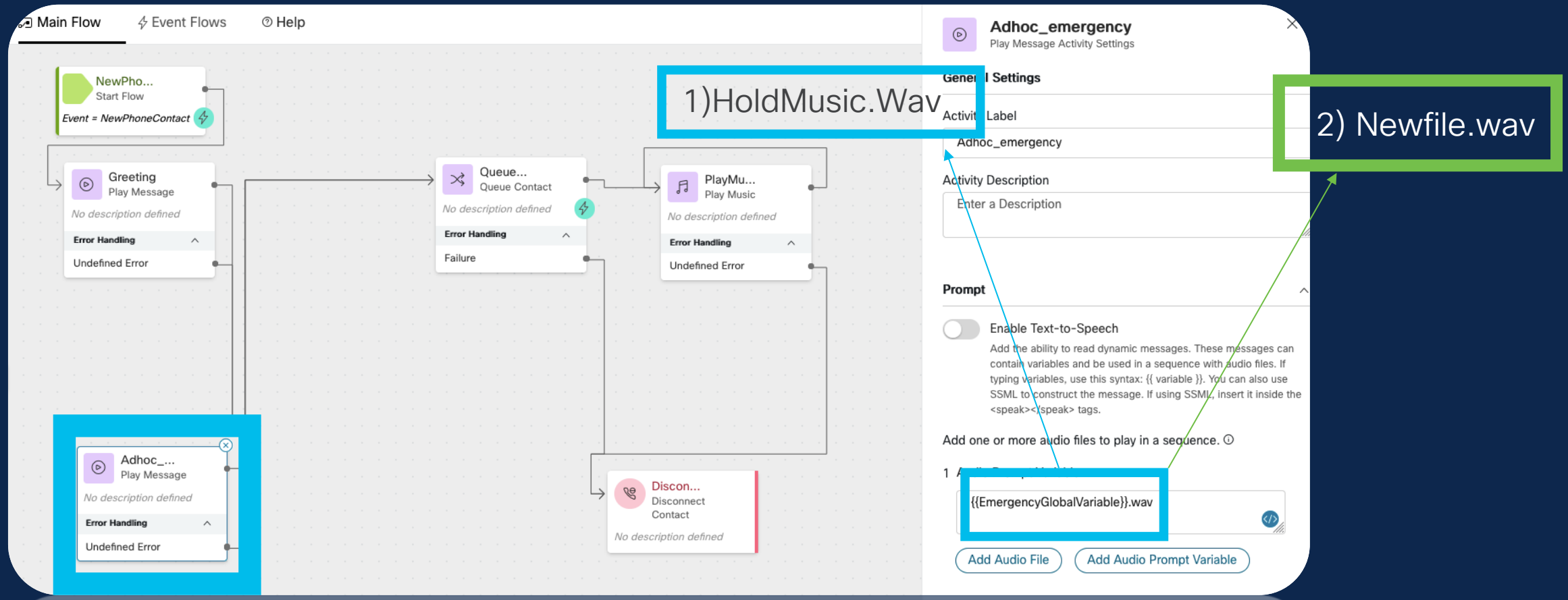
Use case – Play emergency message.



Use case – Play emergency message.

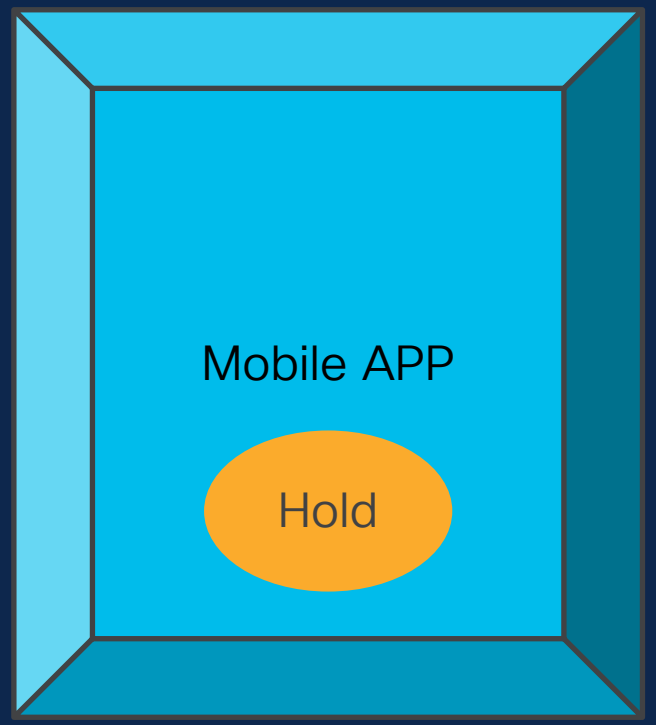


Use case – Play emergency message.



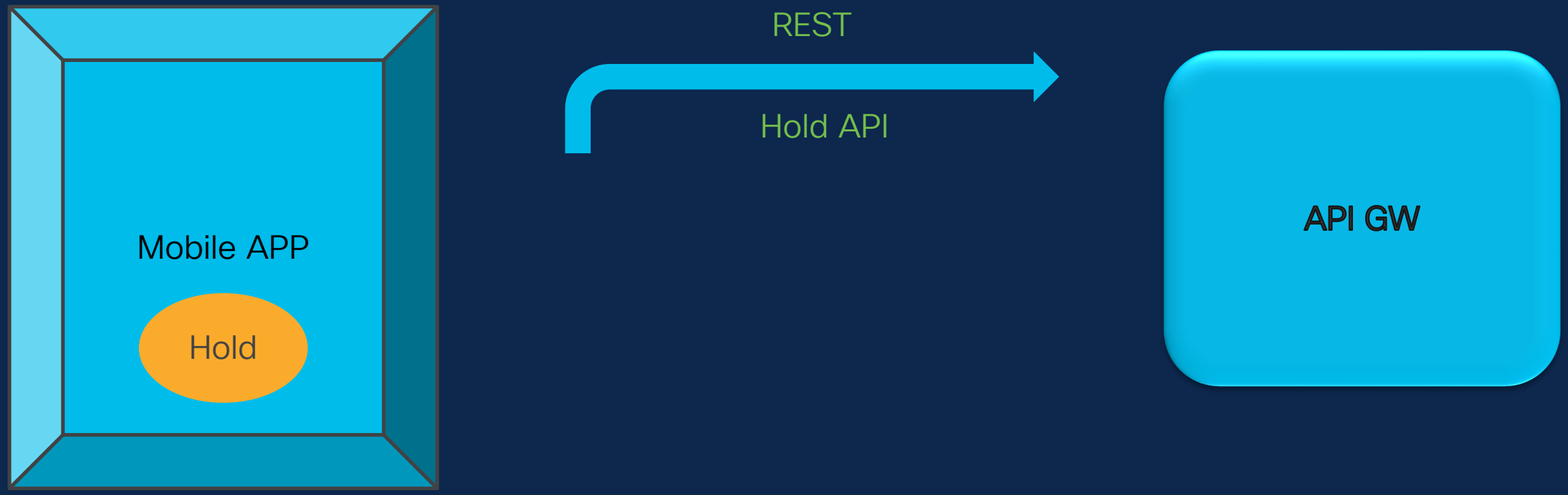
Fully Custom Agent Desktop – Mobile App.

Use case – Fully custom Agent desktop app

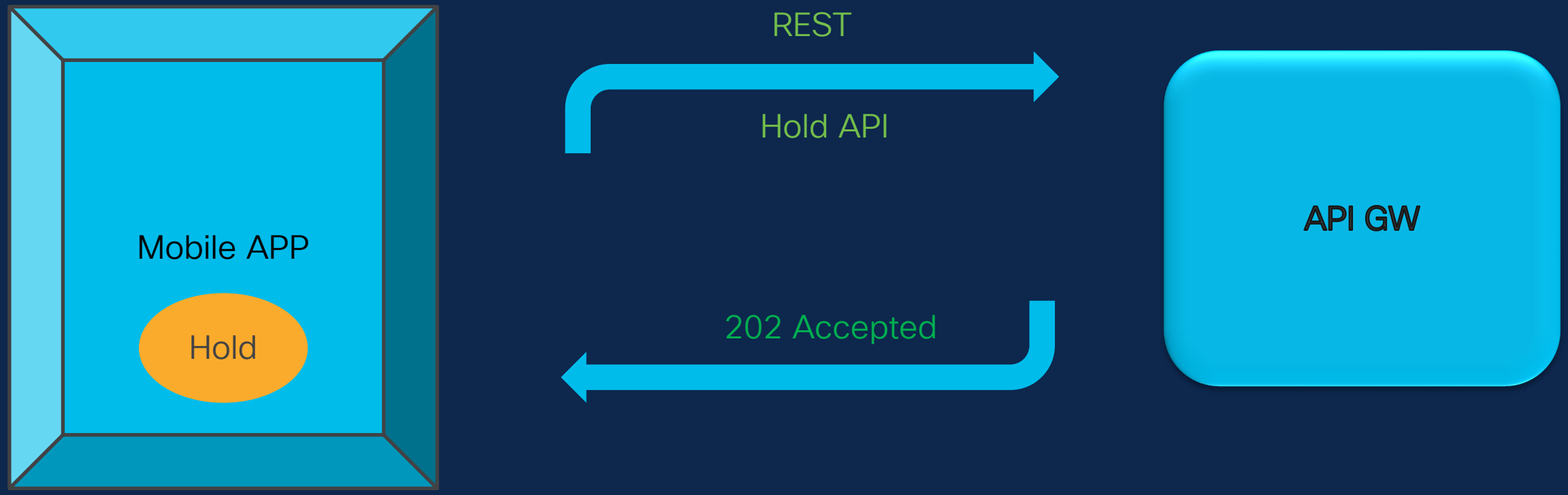


Use case – Fully custom Agent desktop app

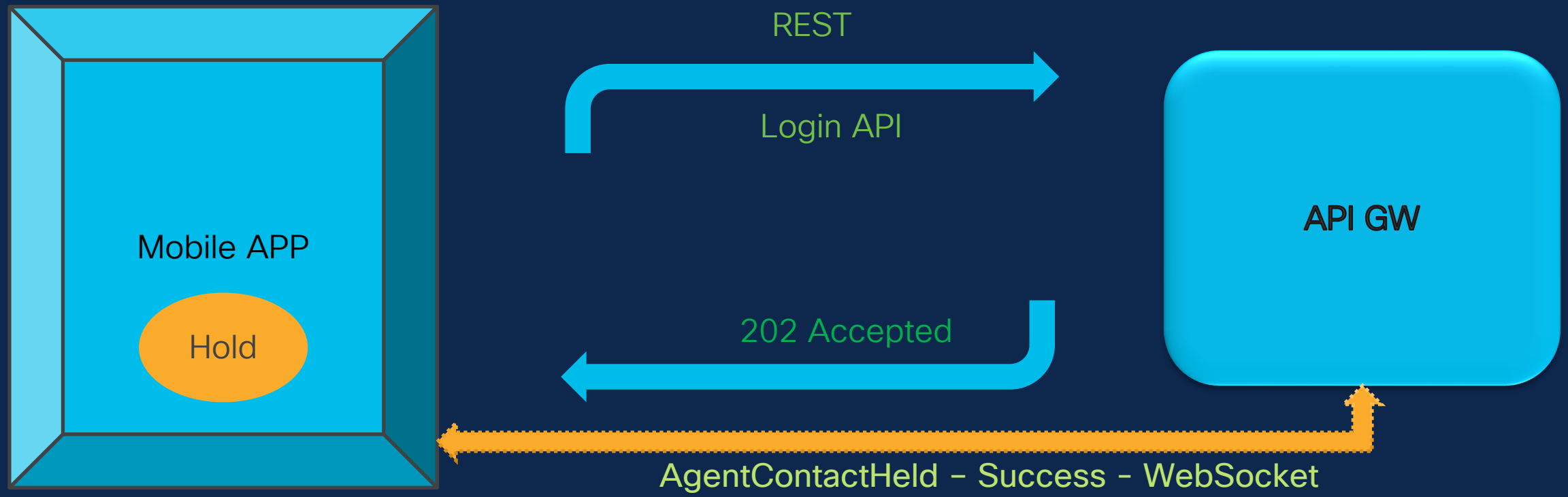
Use case – Fully custom Agent desktop app



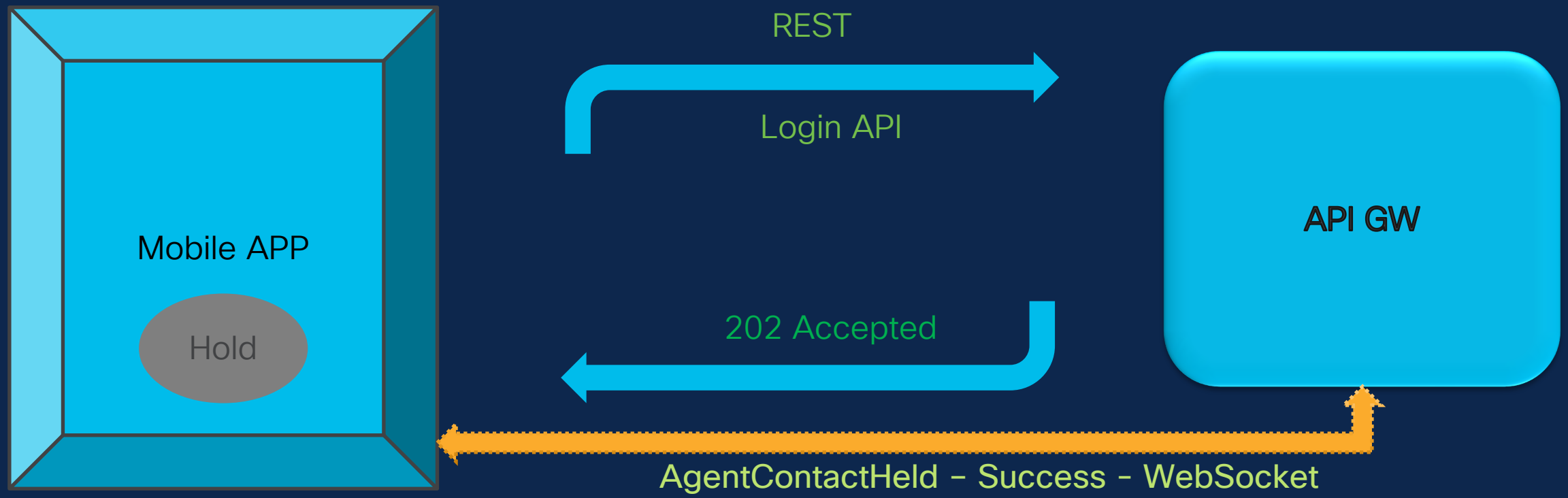
Use case – Fully custom Agent desktop app



Use case – Fully custom Agent desktop app



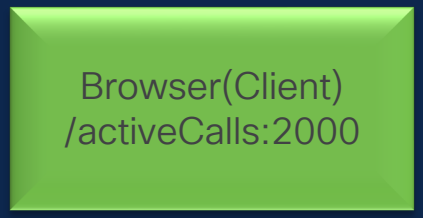
Use case – Fully custom Agent desktop app



Wallboard

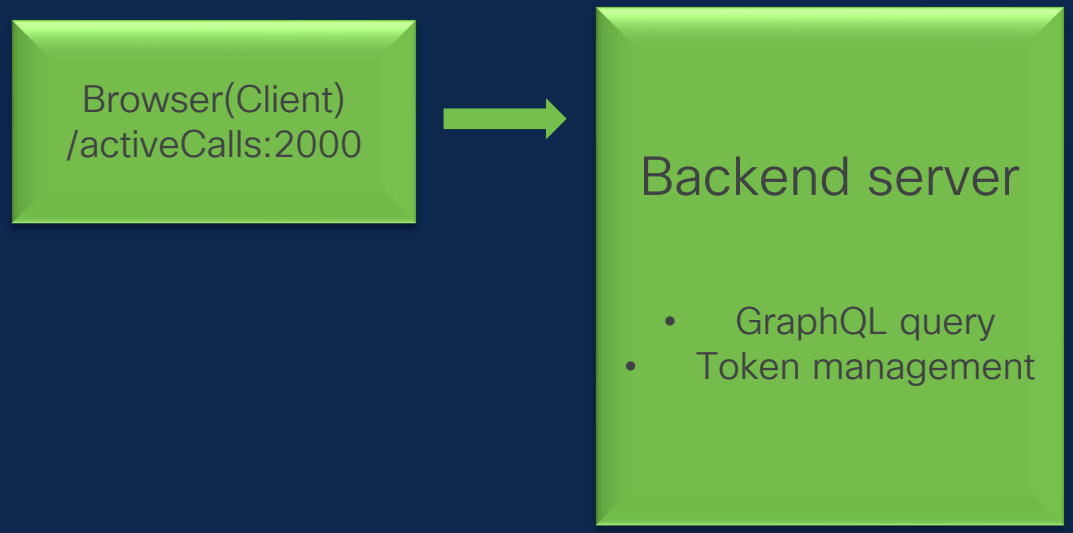
Use case – WallBoard

- Making a backend API calls every X seconds fr*om browser.*



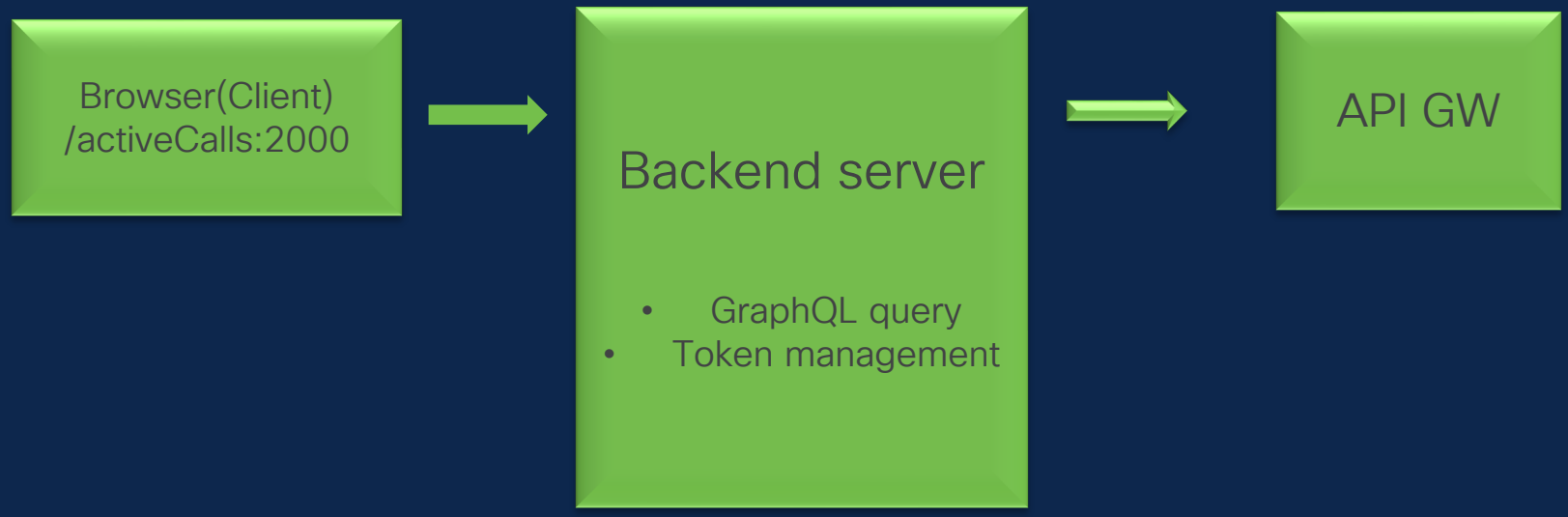
Use case – WallBoard

- Making a backend API calls every X seconds fr*om browser.*
- Design the query depending upon the data you would like to see on wallboard.
- Token management to rotate access token every 10 hours.



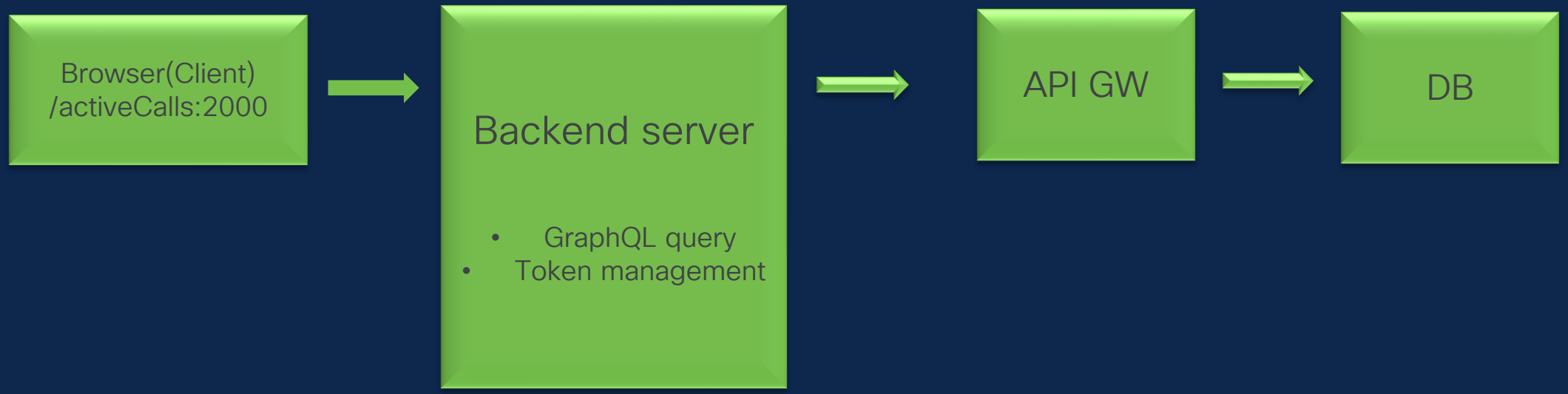
Use case – WallBoard

- Making a backend API calls every X seconds fr*om browser.*
- Design the query depending upon the data you would like to see on wallboard.
- Token management to rotate access token every 10 hours.
- POST API call to API GW (authorization, rate limiting)



Use case – WallBoard

- Making a backend API calls every X seconds fr*om browser.*
- Design the query depending upon the data you would like to see on wallboard.
- Token management to rotate access token every 10 hours.
- POST API call to API GW (authorization, rate limiting)
- Call to DB.



BUT WAIT THERE'S MORE

Query.....?

```
{
  #FILTER: Fetch Real-time (Active) Queued Tasks on the System - using filters.
  task(
    from: {{time_milliseconds_lookback}} #This can be set to Date.now() - (days * 24 * 60 * 60 * 1000) for look back in days
    to: {{time_milliseconds}}  #This can be set to Date.now() in ms
    filter: {
      #The main filter for active tasks is isActive: true
      and: [
        { channelType: { equals: telephony } }
       { status: { equals: "parked" } }
       #direction: { equals: "inbound" } }
        { isActive: { equals: true } }
      ]
    }aggregations: {
      field: "id"
      type: count
      name: "Total parked Calls"
    }
  ) {
    tasks {
      #Add more fields here as needed

      aggregation {
        name
        value
      }
    }
  }
}
```

Query.....?

```
{
  #FILTER: Fetch Real-time (Active) Queued Tasks on the System - using filters.
  task(
    from: {{time_milliseconds_lookback}} #This can be set to Date.now() - (days * 24 * 60 * 60 * 1000) for look back in days
    to: {{time_milliseconds}}  #This can be set to Date.now() in ms
    filter: {
      #The main filter for active tasks is isActive: true
      and: [
        { channelType: { equals: telephony } }
       { status: { equals: "parked" } }
       #direction: { equals: "inbound" } }
        { isActive: { equals: true } }
      ]
    }aggregations: {
      field: "id"
      type: count
      name: "Total parked Calls"
    }
  ) {
    tasks {
      #Add more fields here as needed

      aggregation {
        name
        value
      }
    }
  }
}
```

Query.....?

```
{
  #FILTER: Fetch Real-time (Active) Queued Tasks on the System - using filters.
  task(
    from: {{time_milliseconds_lookback}} #This can be set to Date.now() - (days * 24 * 60 * 60 * 1000) for look back in days
    to: {{time_milliseconds}}  #This can be set to Date.now() in ms
    filter: {
      and: [
        { channelType: { equals: telephony } }
       { status: { equals: "parked" } }
       #direction: { equals: "inbound" } }
        { isActive: { equals: true } }
      ]
    }aggregations: {
      field: "id"
      type: count
      name: "Total parked Calls"
    }
  ) {
    tasks {
      #Add more fields here as needed

      aggregation {
        name
        value
      }
    }
  }
}
```

Query.....?

```
{
  #FILTER: Fetch Real-time (Active) Queued Tasks on the System - using filters.
  task(
    from: {{time_milliseconds_lookback}} #This can be set to Date.now() - (days * 24 * 60 * 60 * 1000) for look back in days
    to: {{time_milliseconds}}  #This can be set to Date.now() in ms
    filter: {
      #The main filter for active tasks is isActive: true
      and: [
        { channelType: { equals: telephony } }
       { status: { equals: "parked" } }
       #direction: { equals: "inbound" } }
        { isActive: { equals: true } } }

    }aggregations: {
      field: "id"
      type: count
      name: "Total parked Calls"
    }
  ) {
    tasks {
      #Add more fields here as needed

      aggregation {
        name
        value
      }
    }
  }
}
```

Query.....?

```
{
  #FILTER: Fetch Real-time (Active) Queued Tasks on the System - using filters.
  task(
    from: {{time_milliseconds_lookback}} #This can be set to Date.now() - (days * 24 * 60 * 60 * 1000) for look back in days
    to: {{time_milliseconds}}  #This can be set to Date.now() in ms
    filter: {
      #The main filter for active tasks is isActive: true
      and: [
        { channelType: { equals: telephony } }
       { status: { equals: "parked" } }
       #direction: { equals: "inbound" } }
        { isActive: { equals: true } }
      ]
    }aggregations: {
      field: "id"
      type: count

    }
  ) {
    tasks {
      #Add more fields here as needed

      aggregation {
        name
        value
      }
```

05
Availability & Roadmap

Developer Experience Roadmap (April'23)

Available Today

- Detailed API reference, guides and getting started documents
- API flavors supported – REST, gRPC, GraphQL, Webhook events
- SDKs (Desktop JS SDK)
- Sample code and API calls in 6 variants (cURL, JavaScript, Java, C#, Python, Go)
- Embedded interactive editor & try it out.
- Changelog and Subscription
- Partner Showcase
- Support (FAQs, Developer Community)
- Marketplace - Apphub
- Github Sample code & Demos

Coming Soon

Developer Portal

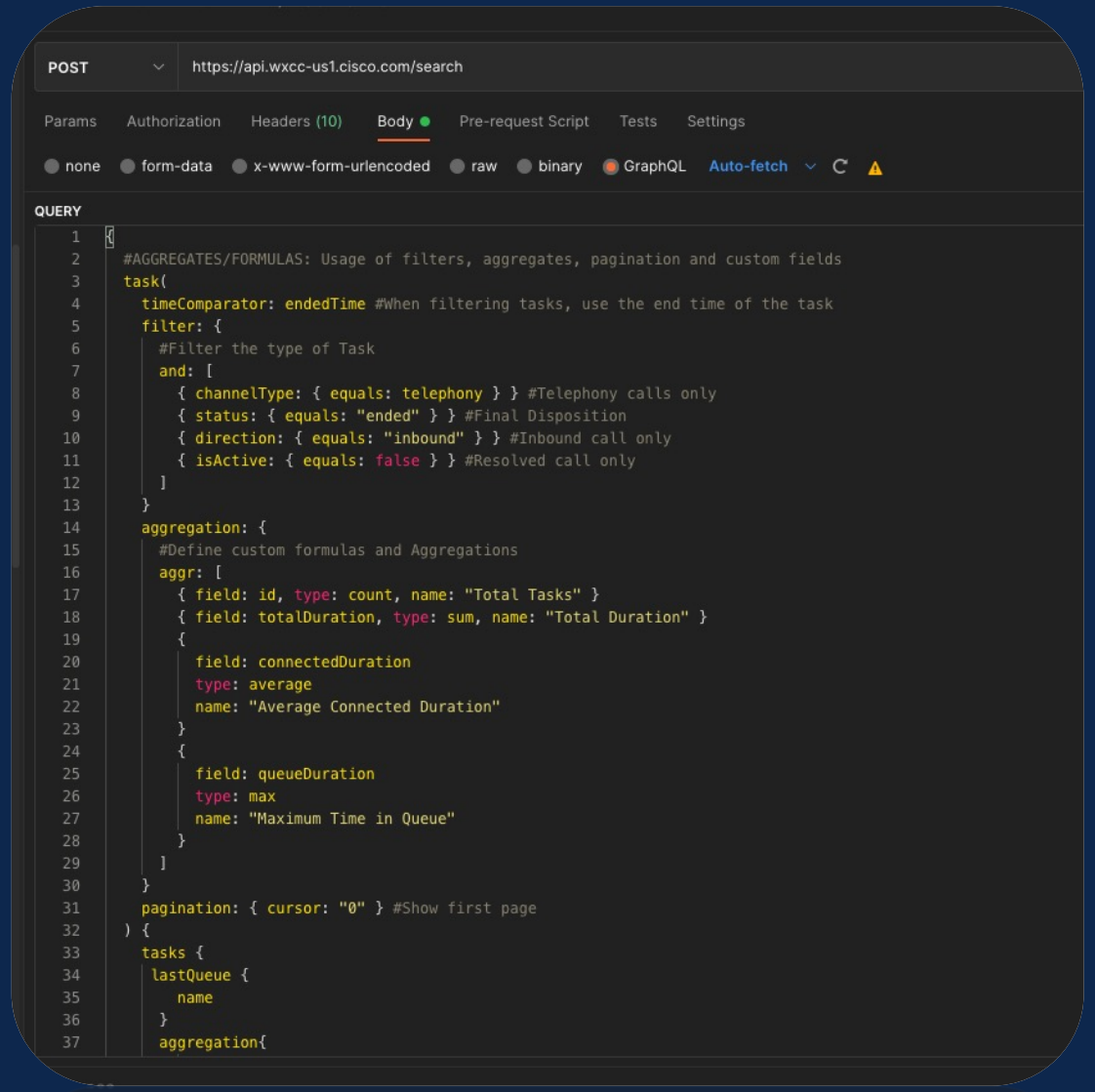
- Organize API by use cases
- Sandbox
- Search functionality
- APIs support from flow.

Contact Center APIs

- Flow Import/Export APIs
- Bulk Data Export APIs
- Outbound Dialer APIs
- Supervisor Desktop APIs

06
Samples & Support

API Use case Samples Available on GitHub



- GitHub Repo – CiscoDevNet
  webex-contact-center-api-samples
- Sample Code – Clone
- ReadMe – Environment.
- Video Tutorials – Use Cases, API, code

https://github.com/CiscoDevNet/webex-contact-center-api-samples

API Use case Samples Available on GitHub

#	Folder Name	Comments	Link
0	postman-sample	Getting Started with the Webex Contact Center APIs and how to create a pair of Client ID or Client Secret for OAuth2 etc.	Postman Sample
1	app-auth-sample	This is a sample application that shows you how to obtain an access token. It also has sample GET calls for Tasks, Agent Stats, Queue Stats, Users, Sites, etc.	OAuth2 Sample
2	configuration-app-sample	This is a sample frontend application that shows you how to use the Configuration APIs for EPs, Queues, Teams, and expose the capabilities on the front-end	Configuration App Sample
3	graphql-sample	This is a sample application that has example calls for the new /search endpoint that is powered by GraphQL. One can formulate multiple request types that support the GraphQL syntax. Both Tasks and Agent Sessions are supported.	GraphQL /search API Sample
4	webhook-email-notify-sample	This is a sample application that shows you how to use Webhooks and send an email notification using webhooks.	Webhook Sample with Email Notifications
5	call-recording-download-sample	This is a sample application that shows you how to use Webhooks - the capture:created Webhook allows you to download a new call recording on the system, to a local file.	Call Recording Download Sample
6	token-app-sample	This is a sample application that shows you how to build a scheduler service that obtains a new access token every 10 hours from Webex and persist this onto your local datastore. The example uses a simple SQLite DB as an example.	Token Management Service Sample
7	callback-sample	This is a sample application that shows you how to build a front end Form to Leverage our brand new Webcallback API that injects a callback call into Webex Contact Center. The example uses a simple javaScript injection technique to inject the form in any webpage of your choice.	Web Callback API Sample
8	widget-sample-101	This is a sample web component widget that shows you how to build a web component widget from scratch. It also covers how you can read the Desktop STORE to retrieve information and pass these values into the widget via the layout.	Web component Widget Sample & 101

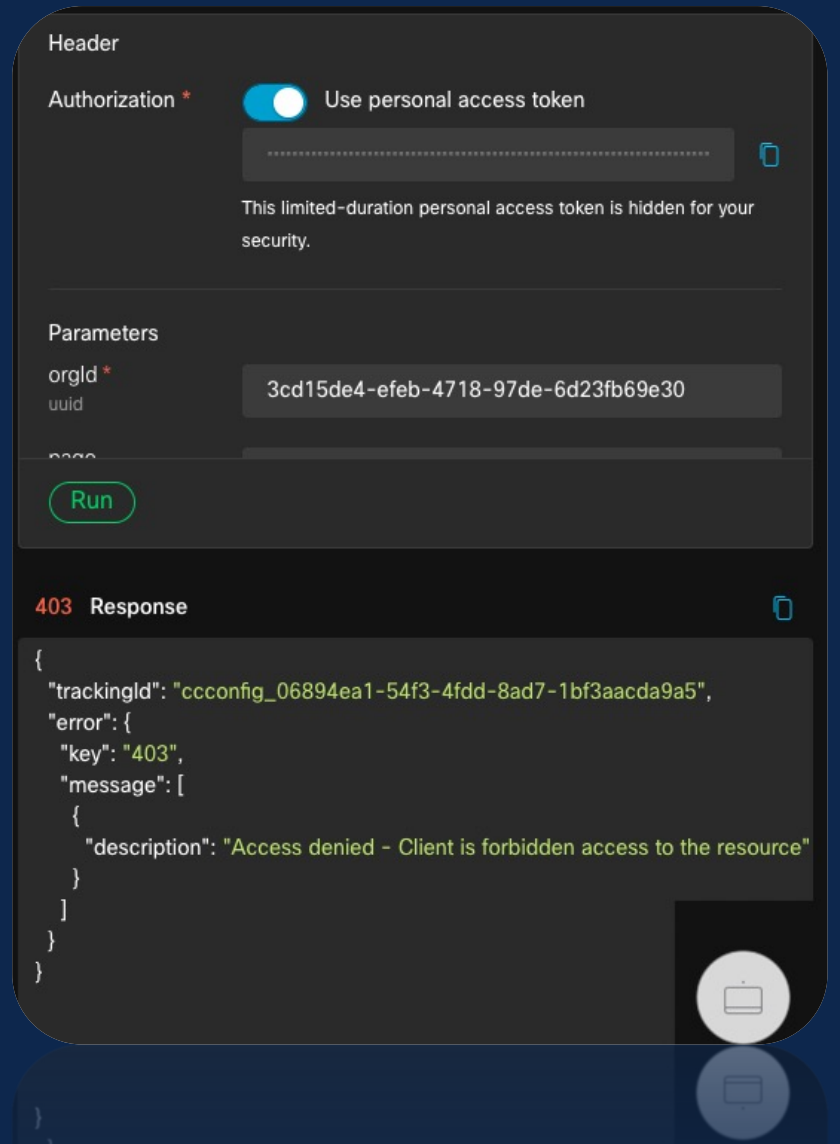
- Checkout out the GitHub – CiscoDevNet repository:

  webex-contact-center-api-samples
- Sample Code
- ReadMe
- Video Tutorials
- Use Cases

https://github.com/CiscoDevNet/webex-contact-center-api-samples

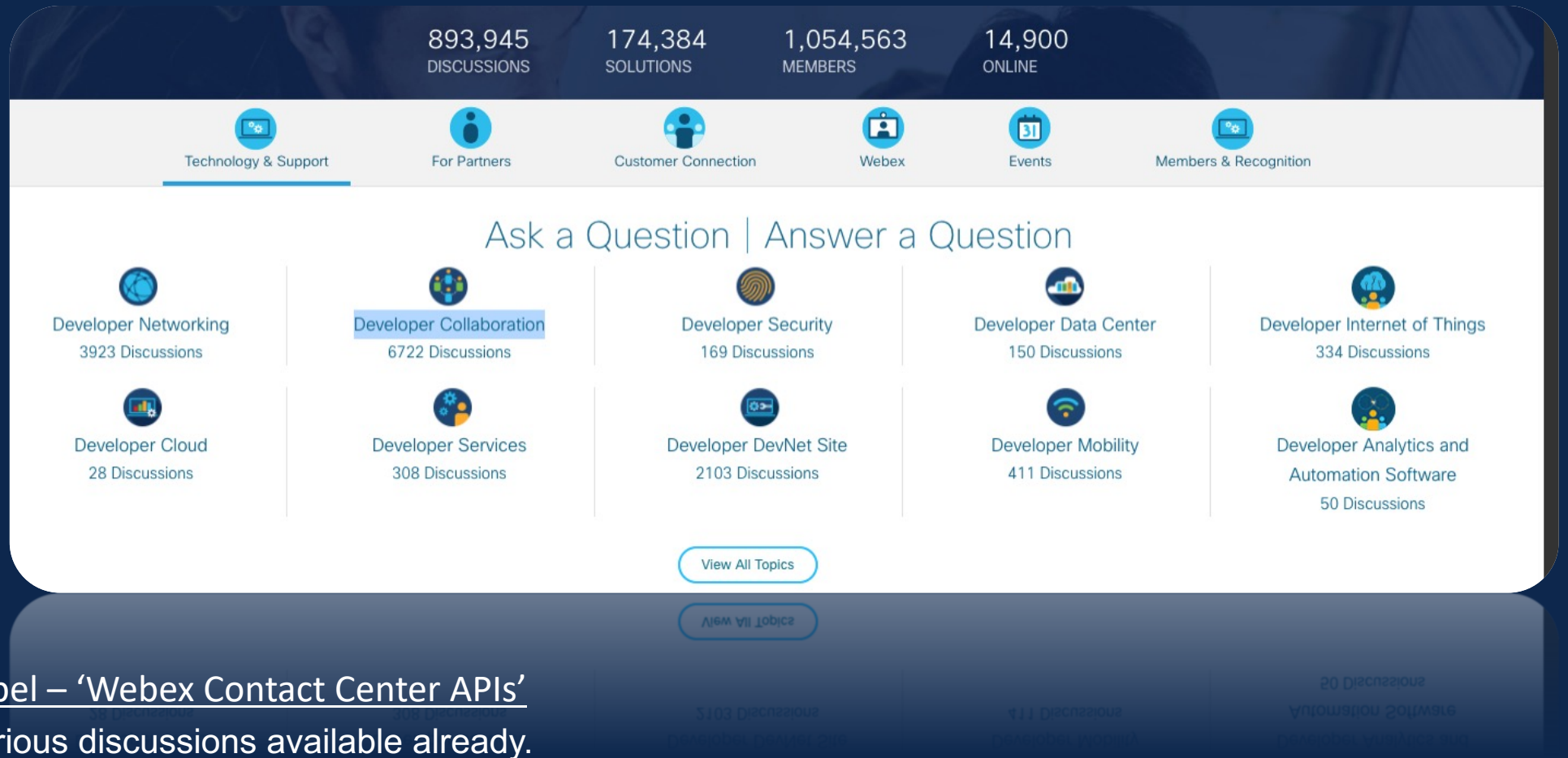
Support – Break Fix

- Open a TAC case.
- Tracking ID.



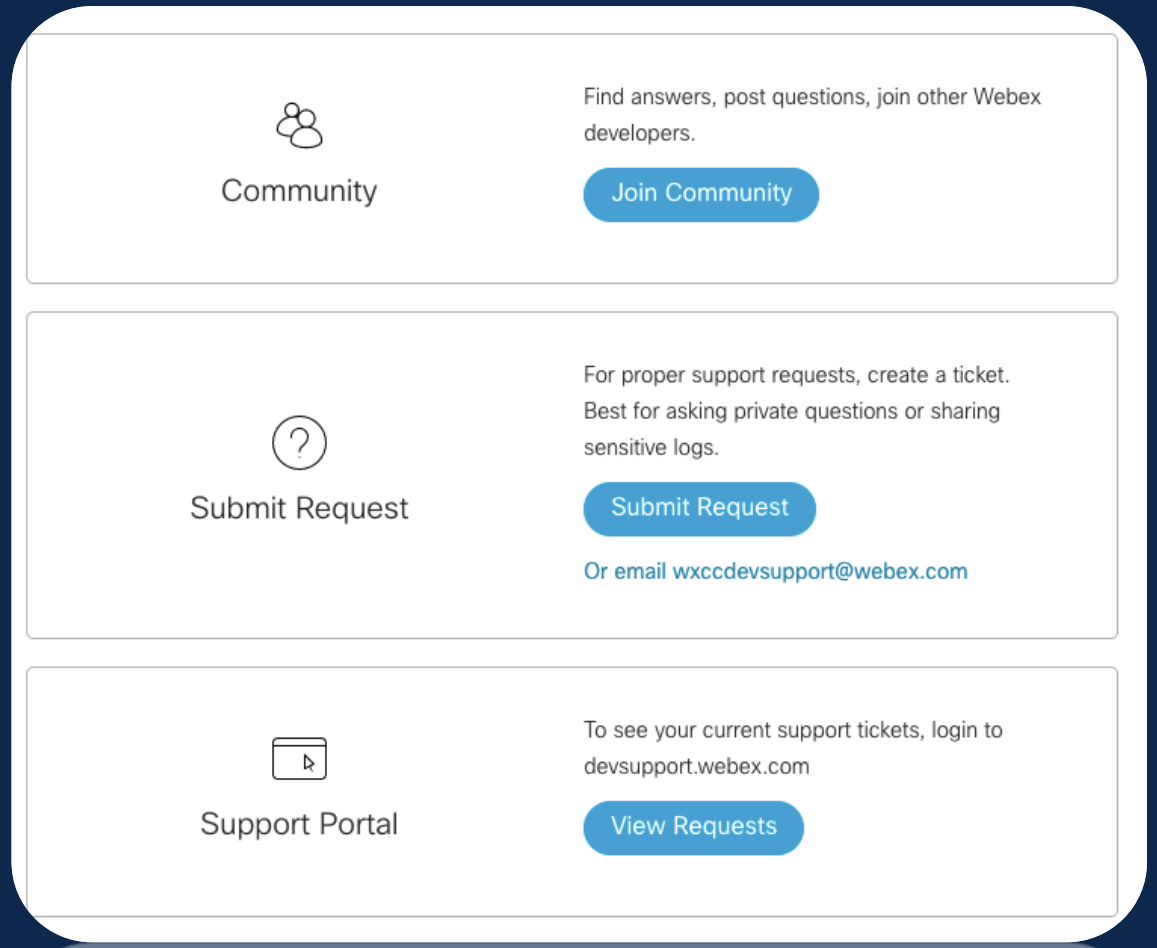
Webex Contact Center APIs Support Community

Cisco Community > Technology and Support > For Developers > Developer Collaboration > Contact Center



- Label – 'Webex Contact Center APIs'
- Various discussions available already.

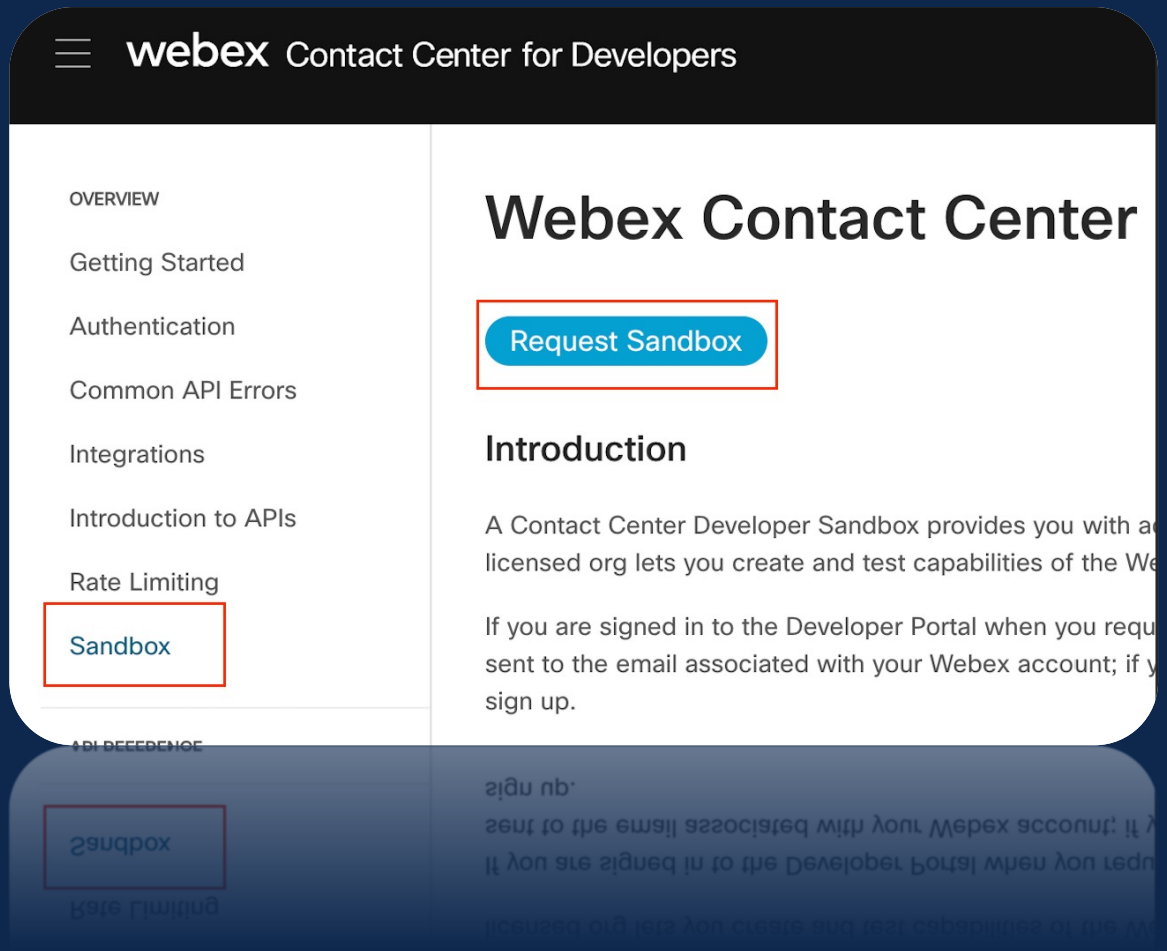
Developer support



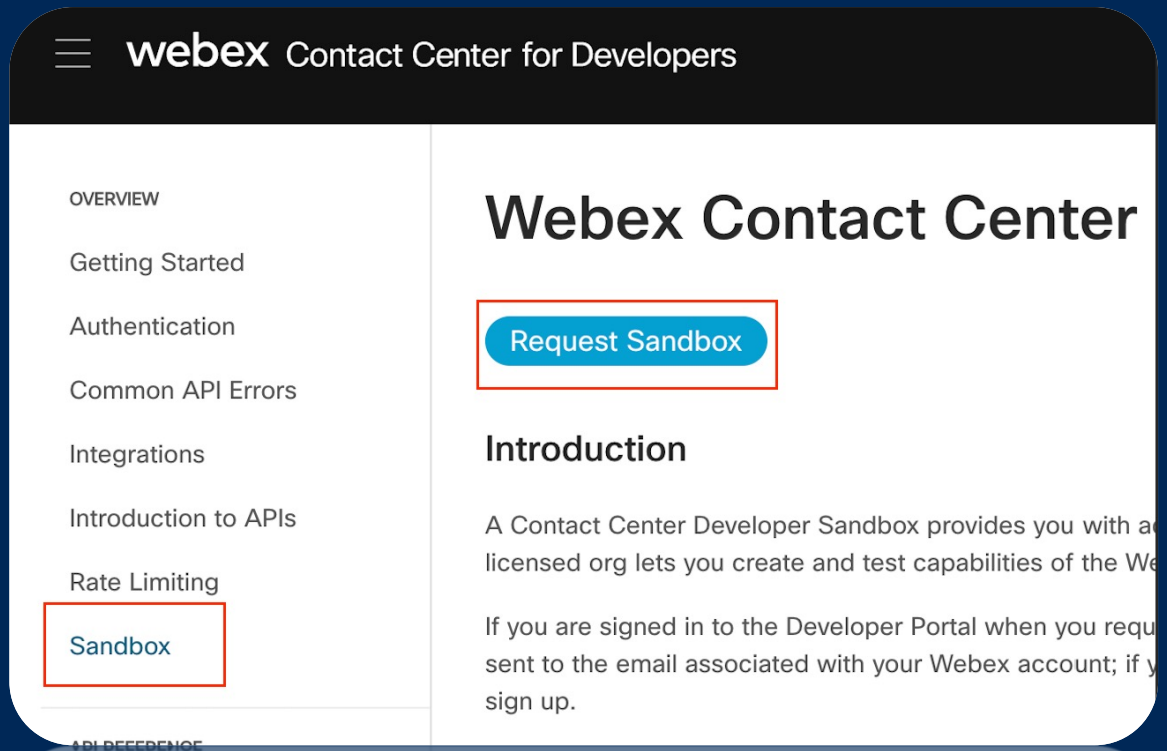
1) Participate in developer forum.
2) Submit support request from developer portal.

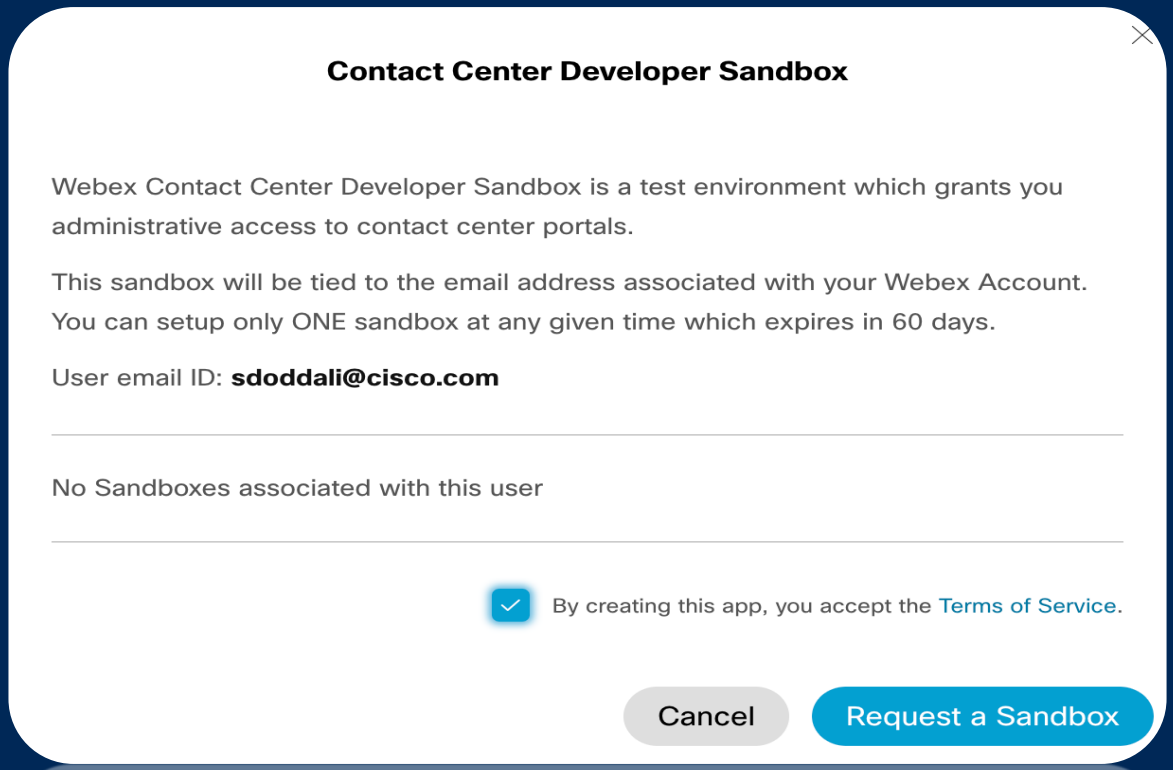
Manage your support requests.

Sandbox



Sandbox





Sandbox

Getting Started

The provisioning email you receive contains the following information for your sandbox organization:

- Organization name: (e.g. jdoessandboxtest-1xep)
- Username/email: (e.g. admin@jdoessandboxtest-1xep.wbx.ai)
- Password
- Webex Site URL: (e.g. jdoesandboxtest-1xep.webex.com)

> ⓘ **Note:** This sandbox will remain active only for a 60 day period. To avoid any potential conflicts with your primary Webex account, it's recommended that you use a private or incognito browser window to perform the sign in.

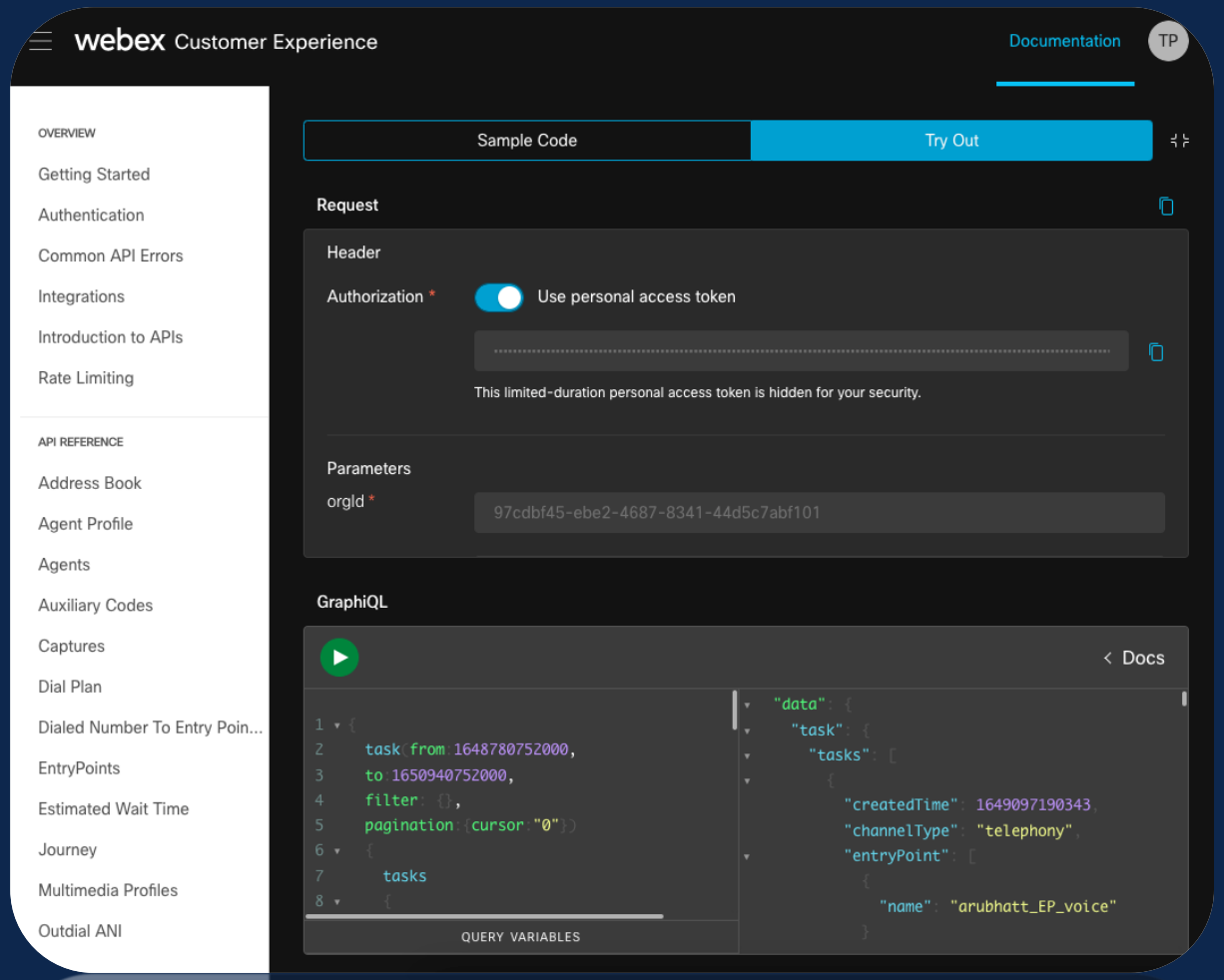
12. **Agent/User** - Few users are configured
    - The users are assigned
      1. Default "Agent Based" team, default Agent Profile and default Site and a default Multimedia Profile.
    - All the users are enabled for contact center.
    - The following user have been pre-configured.
      1. Premium Agent 1
         - Username: (e.g. user1Email@email.com)
         - Password
         - Extension (e.g. 2727)
      2. Supervisor Agent 2
         - Username: (e.g. user2Email@email.com)
         - Password:
         - Extension: (e.g. 3728)
13. **Flow** - A default basic flow is configured to play a default hold music and redirect the call to the default inbound queue.
14. **Routing strategy** - A routing strategy has been configured mapped to the default inbound entry point and the default basic flow that is active for 60 days from the time of creation. This means the routing strategy is immediately active.

Sandbox contents

Every sandbox requested from the developer portal has the following contents so that you can test all of the contact center features end to end:

1. **Webex calling for Voice** - Webex calling is the default calling platform.
    - Two (2) PSTN numbers using Cisco PSTN provider. The main number for the default location is reserved.
      1. Main number - Assigned to the default location for Webex calling. This is reserved.
      2. Entry Point number - The second number is configured for Contact center default entry point.
2. **Site** - One (1) site is configured by default.
3. **Team** - An default agent-based team and a capacity based team is available for use.
4. **Multimedia profile** - One (1) default multimedia profile is available
5. **Agent profile** - Two (2) agent profiles available. A default agent profile without Auto wrap-up and an agent profile with auto wrap up is configured.
6. **User profile** - Five (5) pre-configured user profiles are available
    - Administrator Only Profile
    - Administrator Profile
    - Premium Agent User Profile
    - Standard Agent User Profile
    - Supervisor Profile
7. **Administrator** - One (1) administrator user is configured.
8. **Entry Point** - One (1) default inbound entry point and one (1) default out-dial entry point are available
9. **Queue** - One (1) default inbound queue is preconfigured. The inbound queue is associated with the default agent based team
10. **Entry Point mappings** - One PSTN direct number is mapped to the default entry point and default inbound queue.
    - Direct number -
11. **Audio files** - A default hold music is already uploaded.

Takeaways



- Login to developer.webex-cx.com with a WebexCC Admin.
- Try out the APIs live & use the access_token for 12 hours -> In a Demo!
- Check out the samples on GitHub
- Request Sandbox.
- Participate in the Cisco Developer Community
- Try out lab.

References

- Developer Portal: https://developer.webex-cx.com/
- Desktop SDK: https://developer.webex-cx.com/documentation/guides/desktop
- Partner Community: https://community.cisco.com/t5/contact-center/bd-p/j-disc-dev-contact-center
- GitHub Samples: https://github.com/CiscoDevNet/webex-contact-center-api-samples

external

Portal
https://developer.webex-cx.com/

The bridge to possible