

inbenta

Salesforce Integration

Index

Introduction	5
Features	5
Preparation	5
Inbenta	5
Contents	5
Variables	6
Salesforce Live Agent	6
Environment	6
Configuration	7
Enable Chat	7
Live Agent Users	7
Company organization	8
Chat Deployment	8
Live Agent Skills	9
Chat button	9
Live Agent Application	10
Building the connector	11
Middleware connector	11
Required configuration	12
Deployment	12
Chatbot SDK adapter	12
Required configuration	13
Optional configuration	13
Troubleshooting	14
The chat invitation does not appear in Agent's console	14
How to test and debug it?	14
Testing	14
Debugging	14
Middleware connector debugging	14
Chatbot SDK adapter debugging	14
More information	15
Inbenta documentation	15
Salesforce Live Agent documentation	15

Introduction

The purpose of this documentation is to define the integration of Inbenta's Chatbot with Salesforce Live Agent.

Features

These are the supported features when escalation to Salesforce Live Agent

- Check agent availability
- Transfer conversation transcript
- Transfer conversation details info
- Send raw text messages from User to Agent
- Send raw text messages from Agent to User
- End the chat conversation from Agent's side
- End the chat conversation from User's side

Preparation

Inbenta

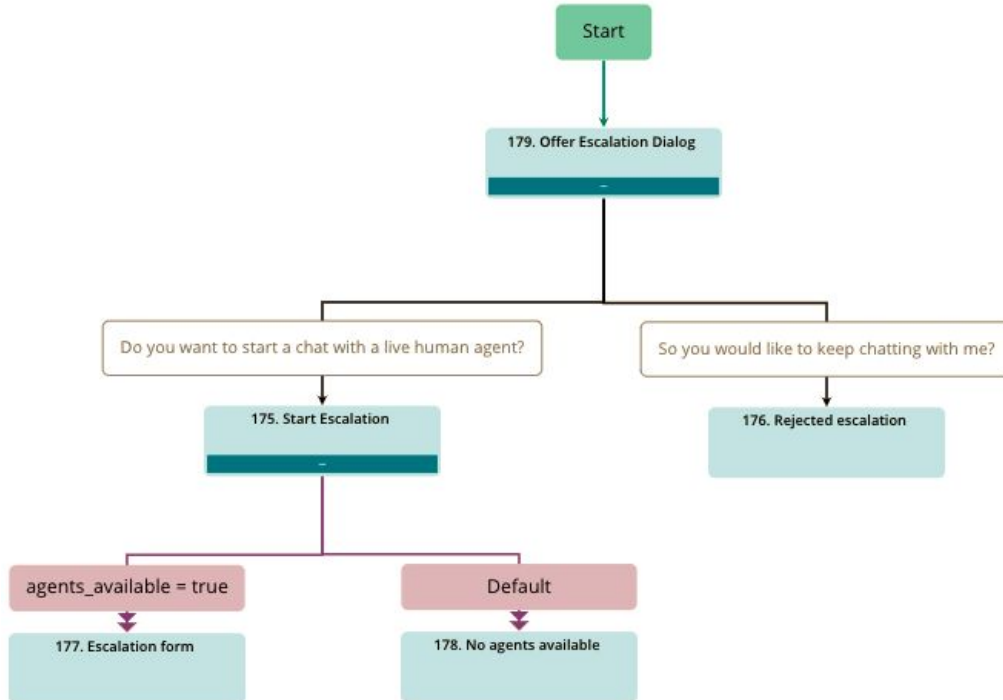
Contents

In your Inbenta APP, make sure you have the default escalation contents and the dialog configured in the Dialog Manager.

The default contents are:

- Chat with a human
- Offer Escalation Dialog
- Start Escalation
- Rejected Escalation
- Escalation form
- No Agents Available

And the dialog will look like this:



For a detailed explanation of how to create the contents and the dialog, follow the [\[GUIDE\] Chatbot - Default contents for live chat escalation](#).

Variables

Confirm you have the default variable **agent_available** created in your Inbenta APP going to *Knowledge > Variables*. In case you don't see this variable, contact with Inbenta.

Salesforce Live Agent

Environment

Make sure you have a working setup of Live Agent within your Salesforce organization. For detailed instruction go to [\[GUIDE\] Salesforce - Chat with customers on your website](#)

Required editions

- Chat is available in: Salesforce Classic and Lightning Experience
- Chat is available in: **Performance** Editions and in **Developer** Edition orgs that were created after June, 2012
- Chat is available in: **Unlimited** Edition and **Enterprise** Edition with Service Cloud

Live Agent feature is available starting with Enterprise edition. It is also accessible for Developer user

Configuration

In order to activate the Live Agent service you will need to perform some configuration in different Salesforce sections.




Lately, you will need the following credentials in order to configure the connector:

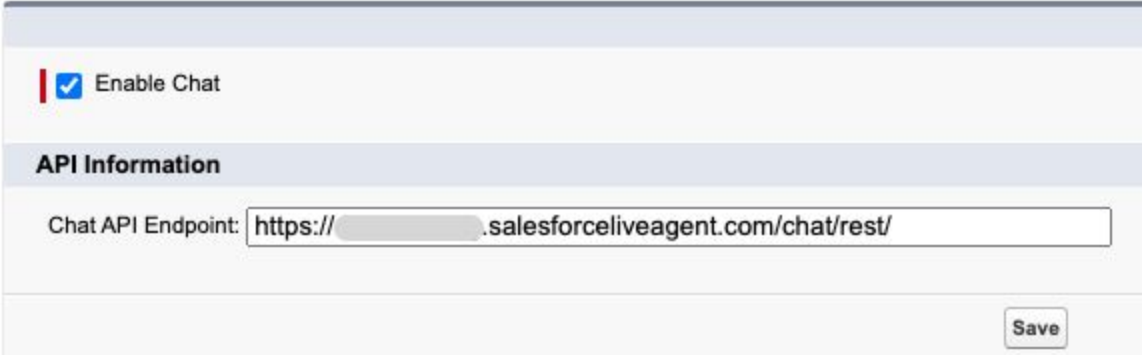
- [Chat API endpoint](#)
- [Organization ID](#)
- [Deployment ID](#)
- [Button ID](#)

Below you will find the way to configure the Live Agent service and obtain the required information.

Enable Chat

First of all, you will have to activate the Live Agent feature. To do this:

1. Navitage to Salesforce  **Setup** page .
2. In the  **Quick Find** box, enter *Chat* and then select  **Chat Settings** (formerly Live Agent Setting).
3. Select the **Enable Chat** checkbox.
4. In the API Information you will get the URL to [Chat API Endpoint](#):



☒ Enable Chat

API Information





Chat API Endpoint:

5. Finish by clicking the **Save** button.

Live Agent Users




Before your users can assist customers with chat, you need to assign the users as **Live Agent users**. **Live Agent users** are support agents and supervisors who have the Salesforce permissions to assist customers with chat. All **Live Agent users** need the API Enabled administrative permission enabled on their associated profile before they can use Live Agent.

In order to do so, follow the steps:

1. Navigate to Salesforce  **Setup** page.
2. In the  **Quick Find** box, enter *Users* and then select  **Users**.
ADMINISTRATION => Users => Users
3. Click the **Edit** button next to a user's name.
4. Select **Chat User** (formerly Live Agent User).
 If you don't see this checkbox verify that your support organization has purchased enough Live Agent feature licenses.
5. Finish by clicking the **Save** button.




After enabling Live User Agents, make sure you associate them with the appropriate skills.

Company organization

1. Navigate to Salesforce  **Setup** page.
2. In the  **Quick Find** box, enter *Company* and then select  **Company Information**.
SETTINGS => Company Settings => Company information
3. In the **Organization Detail**, find the **Organization ID**:

Organization Detail	
Organization Name	
Primary Contact	
Phone	
Fax	
Division	
Default Locale	Spanish (Spain)
Address	ES
Default Language	English
Fiscal Year Starts In	
Default Time Zone	(GMT+02:00) Central European Summer Time (Europe/Paris)
Activate Multiple Currencies	<input type="checkbox"/>
Improve DATEVALUE() accuracy for DST	<input type="checkbox"/>
Newsletter	<input type="checkbox"/>
Currency Locale	Spanish (Spain,Euro) - EUR
Admin Newsletter	<input checked="" type="checkbox"/>
Used Data Space	
Hide Notices About System Maintenance	<input type="checkbox"/>
Used File Space	
Hide Notices About System Downtime	<input type="checkbox"/>
API Requests, Last 24 Hours	0 (15.000 max)
Streaming API Events, Last 24 Hours	0 (50.000 max)
Restricted Logins, Current Month	0 (0 max)
Salesforce.com Organization ID	<organization_id>
Organization Edition	Partner Developer Edition
Instance	EU18

Chat Deployment

1. Navigate to Salesforce  **Setup** page.
2. In the  **Quick Find** box, enter *Deployment* and then select  **Deployments**.
PLATFORM TOOLS => Feature Settings => Service => Chat => Deployments
3. In the **Deployments** menu, click **New** button to create a new **Chat Deployment**.
4. Fill the required fields and click **Save**
5. Go back to the Deployments menu and locate the Deployment created and select it to go to the details page.
6. The **Deployment ID** can be obtained from:
 - a. The page **URL**:

```
https://<sub-domain>.lightning.force.com/lightning/setup/LiveChatDeploymentSettings/page?address=%2F<deployment_id>
```

b. The **Deployment Code** section:

```
<script
  type='text/javascript'
  src='https://<sub-domain>.salesforceliveagent.com/content/g/js/48.0/deployment.js'>
</script>
<script type='text/javascript'>
  liveagent.init('https://<sub-domain>.salesforceliveagent.com/chat',
  '<deployment_id>', '<organization_id>');
</script>
```

Live Agent Skills

1. Navigate to Salesforce ⚙️ **Setup** page.
2. In the 🔍 **Quick Find** box, enter *Skill* and then select ⚙️ **Skills**.
PLATFORM TOOLS => Feature Settings => Service => Chat => Skills
3. In the **Skills** menu, click **New** button to create a new **Live Agent Skill**.
4. Fill the required fields.
5. Select users (and/or profiles) for which **Live Agent Skill** will be available.
6. Add them into **Selected Users** (and/or **Selected Profiles**)
7. Finish by clicking the **Save** button.

Chat button

1. Navigate to Salesforce ⚙️ **Setup** page.
2. In the 🔍 **Quick Find** box, enter *Chat Button* and then select ⚙️ **Chat Buttons & Invitations**.
PLATFORM TOOLS => Feature Settings => Service => Chat => Chat Buttons & Invitations
3. In the **Chat Buttons & Invitations** menu, click **New** button to create a new **Chat Button**.
4. Fill the required fields.
5. In the **Routing information** section, select the **Live Agent Skill** that you just created.
6. Finish by clicking the **Save** button.
7. Go back to the **Chat Buttons & Invitations** menu and click on the created button to be redirected to the **Chat Buttons & Invitations Details** view.
8. The **Button ID** can be obtained from:
 - a. The page **URL**:




```
https://<sub-domain>.lightning.force.com/lightning/setup/LiveChatButtonSettings/
page?address=%2F<button_id>
```


b. The **Chat Button Code** section:

```
<a id="liveagent_button_online_<chat_button_code>"
  href="javascript://Chat" style="display: none;"
  onclick="liveagent.startChat('<chat_button_code>')">
<!-- Online Chat Content -->
</a>
<div id="liveagent_button_offline_<chat_button_code>" style="display: none;">
  <!-- Offline Chat Content -->
</div>
<script type="text/javascript">
  if (!window._laq) { window._laq = []; }
  window._laq.push(function() {
    liveagent.showWhenOnline('<chat_button_code>',
      document.getElementById('liveagent_button_online_<chat_button_code>'));
    liveagent.showWhenOffline('<chat_button_code>',
      document.getElementById('liveagent_button_offline_<chat_button_code>'));
  });
</script>
```

Live Agent Application

Finally create the Live Agent application:

1. Navigate to Salesforce  **Setup** page.
2. In the  **Quick Find** box, enter *Apps* and then select  **Apps**.
PLATFORM TOOLS => Apps => App Manager
3. In the **Apps Manager** menu, click **New** button to create a new **App**.
4. Complete the steps as it follows:
 - a. Step 1: specify **App name** and **Developer name**
 - b. Step 2: select **Console Navigation** and leave supported form factors as default.
 - c. Step 3: optionally add **Utility items**.
 - d. Step 4: add the following **Navigation Items**:
 - i. Chat Visitors
 - ii. Chat Session
 - iii. Chat Supervisor
 - iv. Chat Transcriptions
 - e. Step 5: leave **Navigation Rules** as default.
 - f. Step 6: select the **User Profile** that will have access to the application in the Console. For example, profile of developer user - System Administrator.
⚠ If you don't see a specific profile, go back to Profiles and double check the **visible** checkbox is enabled.
 - g. Finish by clicking the **Save & Finish** button.
5. Finish by clicking the **Save & Finish** button.

Verify that the **App** has been created correctly by checking that it appears in the **APP Launcher**.

Building the connector

You can find the Chabot Salesforce template on GitHub. You can download or clone the template from [GitHub => Inbenta Integrations => Salesforce Chatbot Template](#).

The template contains the full code including:

- Back-end middleware connector
- Front-end SDK application

Middleware connector

The code related to the middleware connector is inside the **/connector** folder.

The main endpoint route should be the file located in **public/**.

The connector has all the logic related to Salesforce Live Agent Rest API. It offers different endpoints that are used called from the JS adapter:

- **/init** used to initiate a chat conversation
- **/availability** this will check whether there are agents available or not.
- **/message/receive** used to retrieve events from agent to user
- **/message/send** used to send events from user to agent

The **events** that can occur during the conversation are:

- Chat established
- Agent typing
- Agent not typing
- Chat messages
- Chat request fail
- Agent disconnect
- Chat transferred

There are more events in Salesforce Live Agent that are not needed (not used). For more information about events check [\[GUIDE\] Salesforce - Live Agent REST API Messages](#).

The connector uses Redis database in order to keep the conversation transcript during the session so it can be retrieved in case any of the parts need to reconnect. For example in case that the user changes or refreshes the website.

Required configuration

The configuration involves Salesforce Live Agent tokens and also the Redis database credentials:

- Go to **src/Configuration/sfla.php** and set your Salesforce Live Agent configuration credentials (the tokens you obtained previously).
- Go to **src/Configuration/redis.php** and set your Redis configuration URL.

Deployment

Host all files inside the **/connector** folders in a server and prepare the endpoint to be set later in the adapter configuration.

You must use a public webserver to serve the Salesforce Live Agent connector. This makes it possible for Salesforce to send events to it. The environment where the connector was developed and tested has the following specifications

- Apache 2.4
- PHP 7.3
- PHP Curl extension
- Predis
- Non-CPU-bound
- The latest version of Composer (Dependency Manager for PHP), to install all dependencies that Inbenta requires for the integration.
- If the client has a distributed infrastructure, this means that multiple servers can manage the user session. They must adapt their SessionHandler so that the entire session is shared among all its servers.

Chatbot SDK adapter

The code related to the Chatbot SDK adapter is inside the **/sdk** folder.

The following is provided:

- The JS adapter **adapter** to be used within the Chatbot SDK.
- An example in the demo that shows how to use the adapter.

Please read the [\[GUIDE\] Chatbot SDK - Adapters](#) if you need help to understand how Chatbot SDK adapters work.

Required configuration

The adapter JS requires a configuration that will be passed as an object when the adapter is added to the chatbot adapters list.

The configuration object must contain the following parameters:

- **Endpoint** (String): the endpoint where the connector is located.
- **Default agent name** (String): the default name that will be displayed in case that Salesforce agent name does not exist.
- **Agent waiting timeout** (Number): the timeout for the chat invitation to expire. If the invitation is not accepted within the defined time then it will proceed like if there were no agents available.

Example of the configuration:

```
let salesforce_configuration = {  
  endpoint: "<endpoint/path_to_connector>", // String  
  agentName: "Agent", // String  
  agentWaitTimeout: 60 // Number  
}
```

Optional configuration

Inside the JS code you will find the object **defaultSalesforceConf** with the default configuration. The configuration mentioned before will replace the default one. However, you will notice that this new object has more fields that can be also configured:

- **Send messages timeout**: this is the time that will last every request from the Chatbot to the Salesforce agent meant to wait for events from the agent side. This request will be reseted every time a new event arrives or when the timeout is triggered. Default time is 30 seconds which is the recommended time.
- **Send messages delay**: this is the time the Chatbot will wait to send messages to the agent. The recommended time is 2 seconds.
- **Input identifier**: identifier of the Chatbot window input. This should not change so leave it as default.

Troubleshooting

The chat invitation does not appear in Agent's console

If the chat escalation works correctly with no error messages but the invitation does not appear in the agent's console. Check the Salesforce credentials configuration, especially the **button Id**.

Also, confirm that the agent has rights to accept invitations from the specified queue.

How to test and debug it?

Testing

Simply use the chatbot and escalate to the agent.

It is recommended to use the chatbot in the **development** environment and also to use a testing queue from Salesforce Live Agent.

Debugging

Middleware connector debugging

The connector already has a **Logger** feature that you can use to start tracking data. Simply add the logger namespace to the file you want to work and start using it as a static function.

Find the configuration in **src/Configuration/logs.php**.

```
use Utils\Logger;  
  
Logger::logText('Initialize');
```

See it is initialized and used in **public/index.php**. When initiated, it will set an identification this way you can easily isolate the trackings that come from the same request.

Chatbot SDK adapter debugging

The JS adapter has already prepared a debugging feature that can be activated through the configuration. Inside the adapter JS you will be a default configuration that contains a

parameter named **debug** that accepts **boolean** values. When set to **true** it will log to the console a predefined useful log. You can add more logs by using the function named **dd** like this:

```
dd(<COMMENT>, <VARIABLE_1>, <VARIABLE_2>, ... , <VARIABLE_N>);
```

There are some examples inside the JS adapter code **adapter/salesforce-connector-adapter.js**.

More information

Inbenta documentation

Inbenta Chatbot SDK - Adapters

<<https://developers.inbenta.io/chatbot/javascript-sdk/sdk-adapters>>

Inbenta Chatbot SDK - Escalation adapter V2

<<https://developers.inbenta.io/chatbot/javascript-sdk/sdk-adapters/nl-escalation-adapter-2>>

Inbenta Chatbot - Contents for chat escalation

<<https://help.inbenta.io/en/default-contents-for-live-chat-escalation/>>

Salesforce Live Agent documentation

Salesforce - Chat REST API Developer Guide

<https://developer.salesforce.com/docs/atlas.en-us.live_agent_rest.meta/live_agent_rest/live_agent_rest_understanding_resources.htm>