

# EasyConnect

Manual



# Table of Contents

EasyConnect	6
Installation	7
Introduction	7
Installing the App	7
Updating the App	8
Creating an App Instance	9
Creating a PBX App Object for the App Instance	11
Add the App to a Template	11
Using the XQTING App Store	11
Configuration	13
Introduction	13
EasyConnect Group	13
Webhook	14
Contacts API	14
API Keys	16
License	17
Developing with EasyConnect	18
Introduction	18
REST APIs Documentation	18
Consuming the REST APIs	19
Monitor Users through a Webhook	20
Support	28
Creating a Log File	28
Feature Requests	28
Contact Information	28

# Table of Figures

Figure 1 - EasyConnect architectural overview .....	6
Figure 2 - Example WebHookEvent JSON object .....	21
Figure 3 - Timings and Durations of a WebhookEvent .....	25
Figure 4 - Example Webhook JSON data for incoming missed call .....	26
Figure 5 - Example Webhook JSON data for incoming call .....	27

# Table of Tables

Table 1 - Description of WebHookEvent JSON object properties .....	22
--	----



# EasyConnect

EasyConnect for innovaphone v13 powers up your innovaphone PBX with REST APIs and webhooks.

EasyConnect allows for a quick and seamless integration of the innovaphone PBX in your application using well known standards and protocols available in any low-code or development environment.

Its main features are:

- the **Administration REST API** to get, update and delete objects from your own applications or automations;
- the **Remote Call Control (RCC) REST API** for controlling calls via the REST API from your own (web) application;
- the **Contacts REST API** to manage your contacts from your own middleware solution;
- the **configurable webhooks** to monitor calls.

*Figure 1 - EasyConnect architectural overview*



# Installation

## Introduction

The EasyConnect app is installed in just a few clicks and with some configuration in myApps the application will be up and running in no time.

The first step is to install the app. This can be done through the official innovaphone App Store or through XQTING's App Store. Next, we have to create an instance for the application. Finally, we create a PBX app object so we can access the app for configuration.

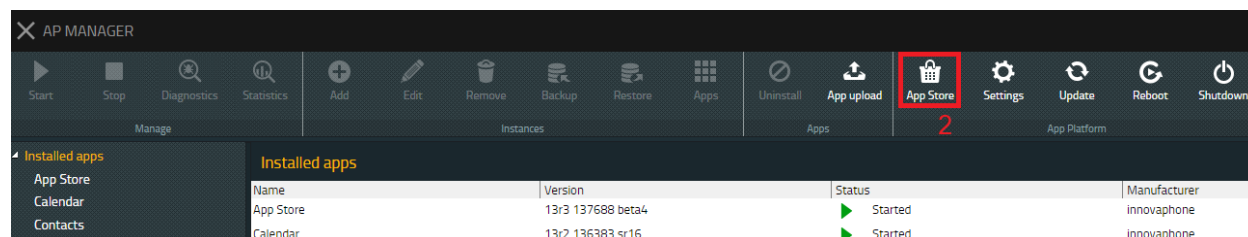
We recommend reading through the installation chapter instead of following our instructions blindly from the get-go.

## Installing the App

Installing the EasyConnect app happens via the App Store. This requires a user with access to the AP Manager app.

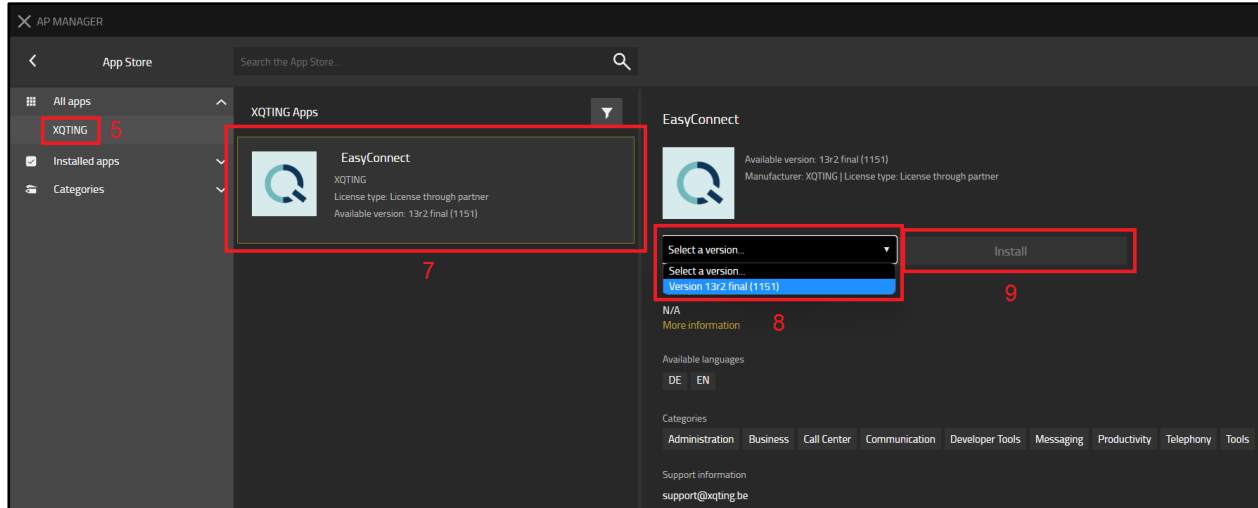
Take the following steps to install EasyConnect from the App Store:

1. Open the **AP Manager** app  
The App Platform Manager (AP Manager) application is one of many apps included with the innovaphone PBX and can be accessed via the 'Apps' dashboard in the bottom-right corner of the myApps landing page.
2. Click on **App Platform > App Store**



3. On the left-hand side, click on **All apps**
4. A list of companies should appear
5. Click on **XQTING**
6. A list of apps developed by XQTING should appear
7. Click on **EasyConnect**
8. In the dropdown menu on the right-hand side, select the latest version of EasyConnect

## 9. Click **Install**



10. Accept the terms of use

11. Click **Install**

12. Wait for the app to be installed

You now have EasyConnect installed.

The next step is to create an app instance (see [Creating an App Instance](#)).

## Updating the App

Updating the EasyConnect app is done in just a few clicks using the official innovaphone App Store.

To get the latest version of the app, take the following steps:

1. Open the **AP Manager** app
2. Click on **App Platform > App Store**
3. On the left-hand side, click on **Installed apps**
4. A list of developers should appear
5. Click on **XQTING**
6. A list of installed apps developed by XQTING should appear
7. Click on **EasyConnect**
8. In the dropdown menu on the right-hand side, select the latest version of EasyConnect
9. Click **Update**
10. Wait for the app to be updated

You have updated EasyConnect to the latest version.

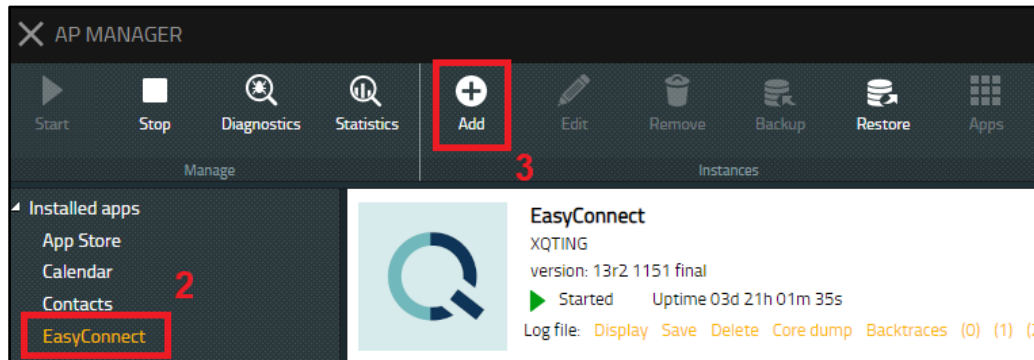


## Creating an App Instance

To start using EasyConnect, an instance needs to be created and started.

Follow these steps to create an app instance:

1. Open the **AP Manager** app
2. On the left-hand side, select the **EasyConnect** app
3. Click on **Instances > Add**



4. Fill out the form:
  - a. **Name:** easyconnect
  - b. **Domain:** <your domain>  
If you are unsure about what to fill out in Domain, you can find your domain by opening the hamburger menu (top-right corner) of the Users Admin app.
  - c. **Password:** <a random password>
  - d. **Database password:** <a random password>  
The passwords are required to save the form, but the values don't matter as the PBX Manager plugin of EasyConnect will configure this for you.

The other fields (Webserver path, Database name and Database user) are automatically generated, and should not be modified.

5. Click **Save**

EasyConnect - Add instance

Name: easyconnect

Domain: xqting.be

Password: ..... ☐ Show password

Webserver path: xqting.be/easyconnect

Database host: empty for a local database, otherwise IP/DNS[:Port]

Database name: xqting.be\_easyconnect

Database user: xqting.be\_easyconnect

Database password: ..... ☐ Show password

☐ Exclude from overall backups

Optimize disk usage of the database.

App URLs (e.g. for the PBX App object):  
[redacted]/xqting.be/easyconnect/xqting-easyconnect

6. Select the newly created instance (it should become highlighted in blue)

7. Click **Manage > Start**

AP MANAGER

Start Stop Diagnostics Statistics Add Edit Remove Backup Restore Apps Uninstall App upload App Store Settings

Manage Instances Apps

Installed apps

- App Store
- Calendar
- CallActionFrame
- Callback
- chiro-PBX-Designer
- Cloud-Designer
- Contacts

EasyConnect

XQTING

version: 13r2 1151 final

Started Uptime 0d 02h 47m 42s

Log file: [Display](#) [Save](#) [Delete](#) [Core dump](#) [Backtraces \(0\) \(1\) \(2\) \(3\)](#)

Name	Domain	Status	Webserver path	Database name	Database user	Database size	Database host
easyconnect	xqting.be	Started	Stoxqting.be/easyconnect	xqting.be_easycor	xqting.be_easycor	7.411 MB	

You now have an instance of EasyConnect running.

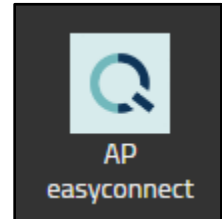
Next, we have EasyConnect's PBX Plugin create a PBX App object for us (see [Creating a PBX App Object for the App Instance](#)) so we can open the app and start to configure it.

## Creating a PBX App Object for the App Instance

To use the app of an App instance, an App object needs to be created on the PBX. Experienced users know how to do this themselves, but we discourage that because there's a lot of configuration to be done on the object. Instead, we recommend creating an App object using EasyConnect's PBX Manager plugin.

To create an App object, take the following steps:

1. Open **PBX Manager**
2. Click on **AP easyconnect**
3. Click on **Create the object now**



You now have created an App object.

Next, we add the EasyConnect app to a user template so the app becomes visible to users created with that template.

*If you get an error message saying an object with the same name already exists, it means there was a past installation of EasyConnect. EasyConnect is unable to delete its own app object, so the existing app object needs to be deleted manually before attempting to create the object for the new EasyConnect installation.*

## Add the App to a Template

To make the app available from the home screen for users, we have to add it to a template. We recommend making the EasyConnect app available only to administrators.

To add the App to a user template, take the following steps:

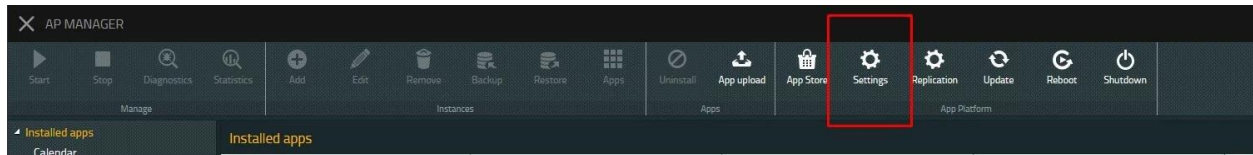
1. Open **PBX Manager**
2. Click on **Templates**
3. Select the template for users you want to give access to the App
4. Open **Apps**
5. Select **easyconnect**
6. Click **OK**

The EasyConnect app is now available from the **All apps** page for users created with the template to which you've added EasyConnect.

## Using the XQTING App Store

Publishing a new version of EasyConnect to the innovaphone App Store can take some time. The fastest way to get access to the latest features and bug fixes is through the XQTING App Store. The following steps explain how to change the configuration of your App Store so it makes use of the XQTING App Store:

1. Open **AP Manager**
2. Click on **Settings** in the App Platform section of the header menu

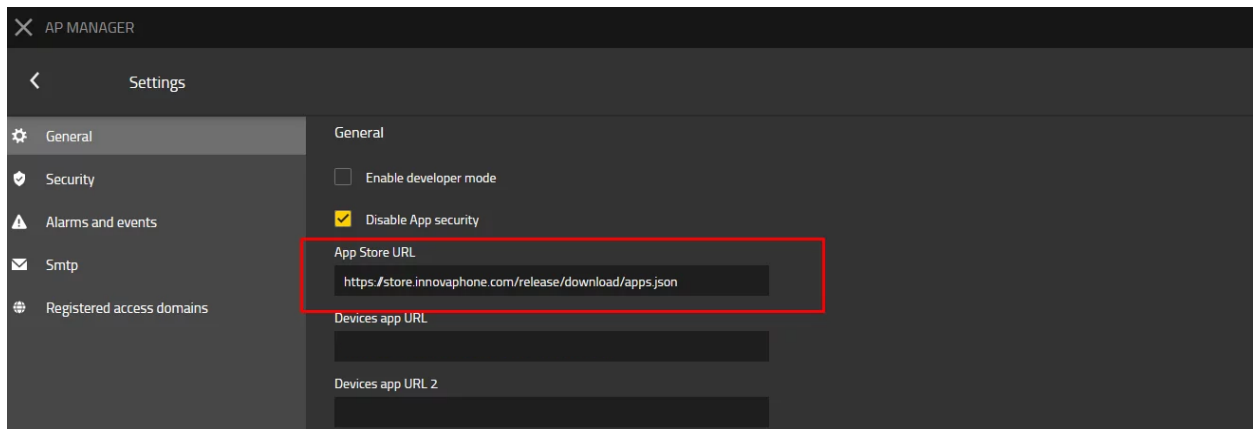


3. In the **General** tab, replace the **App Store URL** with the URL that points to the XQTING App Store.

We recommend to safekeep the current value so you can restore it at a later time. Most likely, the URL <https://store.innovaphone.com/release/download/apps.json> is the current value.

After you have stored the current URL, replace it by the following and save the changes.

<https://00056-apps.innovaphone.com/xqting.be/appstore/download/apps.json>



4. Now you can follow the steps as described in [Installing the App](#) to install the app.
5. Optionally, go back to the App Store's settings screen after installation to restore the **App Store URL** parameter to the previous value.

Anytime you want to have access to the latest EasyConnect version, you can configure your App Store URL parameter to the XQTING App Store.

# Configuration

## Introduction

After having installed the EasyConnect app, there's still some configuration required.

If you plan on using the RCC API, Contacts API or webhooks, the first configuration step is the creation of a group using PBX Manager. All users that should be monitored, i.e. for whom you want to receive webhook events, should be a part of this group. Also, all users for whom you would like to use the RCC API need to be a part of this group. By making the EasyConnect app an active member of the group, we are able to monitor users and initiate calls.

Next, we have to configure the EasyConnect app. The EasyConnect configuration page consists of three sections: Webhook Configuration, API keys Configuration, and License information. The Webhook Configuration allows for the configuration of a webhook. The webhook will be used by EasyConnect to send events using the GET or POST HTTP method. The API keys Configuration allows you to configure the API keys required to authorize requests made to the EasyConnect APIs. The License information section allows you to upload your EasyConnect license and displays license information per API.

## EasyConnect Group

The EasyConnect group is required for webhooks and the RCC API to function. We create one via the PBX Manager and then we make the EasyConnect app an active member of this group.

To make the EasyConnect group, follow these steps:

1. Open the **PBX Manager** app
2. Click on **Groups**
3. Click on **Add a group**
4. Give a name to the group using the **Group** form field, we suggest *EasyConnect*.
5. Add any members you might want to monitor, or make calls for using the EasyConnect RCC API
6. Click on **OK**

Next, we add the EasyConnect app as an active member to this group.

This needs to be done manually via the PBX Admin-UI.

Before making any changes in the PBX Admin-UI via the Devices app, make sure you have closed the PBX Manager app.

To add the EasyConnect app as an active member of the group, follow these steps:

1. Close the **PBX Manager** app
2. Open the **Devices** app

3. Select your PBX device
4. In the PBX Admin-UI, click on **PBX** in the main menu
5. Click on **Objects** in the sub menu
6. Find the easyconnect app object
7. Click on the plus (+) sign in the **Groups** column of the easyconnect app object record
8. A window appears to add a group
9. Select the group (dropdown on the left) created earlier
10. Make the EasyConnect app an active member by enabling the **Active** flag
11. Make sure the membership (dropdown on the right) is set to static
12. Click on **OK**

The EasyConnect app is now ready to monitor and make calls for any user that's a member of the EasyConnect group.

## Webhook

The webhook allows the EasyConnect app to send events on monitored calls to your low-code solution or a REST API of your own making.

To configure the webhook, follow these steps:

1. Open the **EasyConnect** app
2. Fill out the **Webhook Configuration** form:
  - a. **Webhook URL**: the URL to be used to send monitored call events to
  - b. **Webhook request type**: the HTTP method to be used (GET or POST)
  - c. **Enable Webhooks**: the flag used to enable and disable the webhook; make sure to have it enabled to receive events on the configured URL.
3. Click on **Save**

You have now configured a webhook to be used for sending events of monitored calls to.

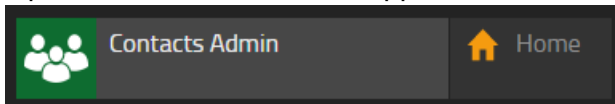
## Contacts API

To make full advantage of EasyConnect's Contacts API, we need to make sure EasyConnect has access with admin rights to the Contacts Directories for which you want to make Contact API requests. Any contact in innovaphone requires a link to a contacts directory.

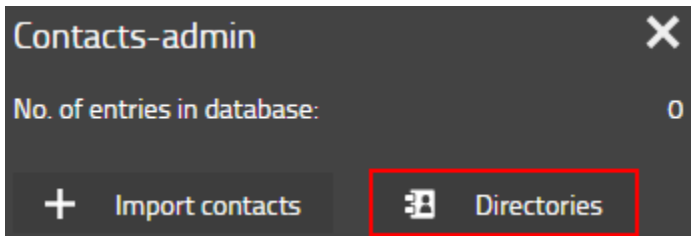
Make sure you have created a group with the EasyConnect app as an active member (see [EasyConnect Group](#)).

To grant EasyConnect access to a Contacts Directory, follow these steps:

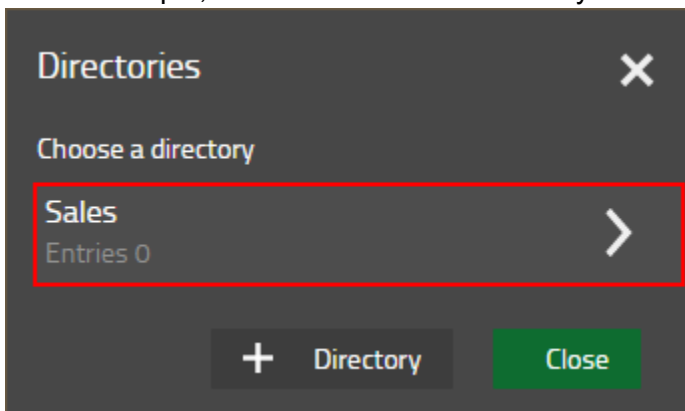
1. Open the **Contacts Admin** app



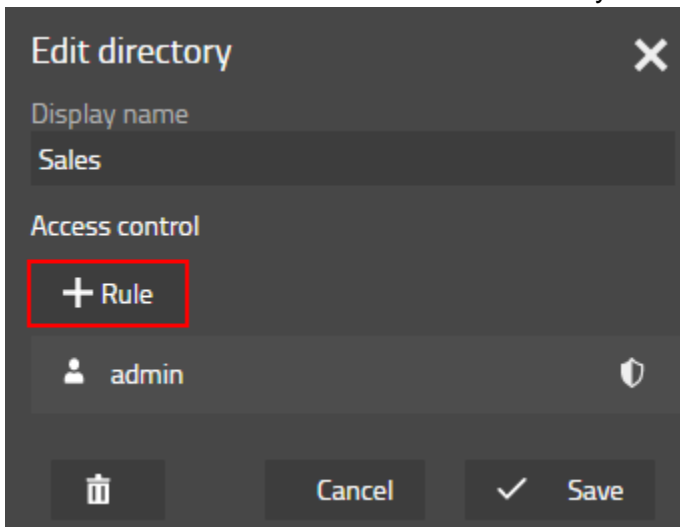
2. In the top-right corner, click on the hamburger menu.
3. Click on **Directories** in the Contacts-admin context menu.



4. Click on a directory to which you want to give EasyConnect access.  
In our example, we'll use a contacts directory with the name 'Sales'.



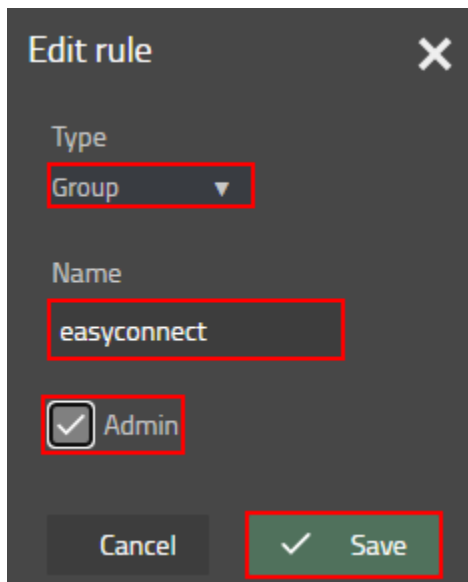
5. Click on **+ Rule** to add an access rule for EasyConnect



6. Fill out the form and click **Save**.
  - a. The **Type** should be set to Group,

- b. the **Name** field should be set to the name of a group of which EasyConnect is an active member,
- c. and the **Admin** checkbox should be enabled.

Admin rights are required for any request that may create or alter a contact in a contacts directory.



The screenshot shows a dark-themed 'Edit rule' dialog box. It has a title bar with 'Edit rule' and a close button (X). Inside, there are three main sections: 'Type' with a dropdown menu showing 'Group', 'Name' with a text input field containing 'easyconnect', and 'Admin' with a checked checkbox. At the bottom, there are two buttons: 'Cancel' and 'Save'. The 'Save' button is highlighted with a green checkmark and a red border.

The Contacts Directory can now be accessed through the Contacts API provided by EasyConnect. To learn more about the Contacts API, please have a look at the documentation provided in the EasyConnect app (see [REST APIs Documentation](#)).

## API Keys

The EasyConnect app's APIs use an API key to authorize requests. After installation, we recommend changing the API keys. You can enter a custom value for your API key, or you can have a random string generated for you using the refresh button to the right of the API key form field.

To configure the API keys, follow these steps:

1. Open the **EasyConnect** app
2. Fill out the **API keys Configuration** form:
  - a. **API key 1**: enter a string; or click on the refresh button to the right to generate a random string.
  - b. **API key 2**: enter a string; or click on the refresh button to the right to generate a random string.
3. Click on **Save**



You have now configured the API keys to be used to authorize requests made to the EasyConnect APIs.

## License

To use the EasyConnect app, a license is required. A license can be requested using XQTING's website. Open the browser and go to <https://xqting.com/products/easyconnect> to request a license (as a reseller or as an end user).

To upload your license, follow these steps:

1. Open the **EasyConnect** app
2. In the **License information** section, click on **Upload License File**
3. Browse for your license file
4. Click on **Upload**
5. The License information should show information about your license, or an error message.

You have now licensed your EasyConnect app and can start using the EasyConnect app as an integration with your low-code application or start implementing the EasyConnect REST APIs in your own apps (see [Developing with EasyConnect](#)).

# Developing with EasyConnect

## Introduction

EasyConnect allows for a quick and seamless integration of the innovaphone PBX in your application using well known standards and protocols available in any low-code or development environment

Its main features are:

- the **Administration REST API** to get, update and delete objects from your own applications or automations;
- the **Remote Call Control (RCC) REST API** for controlling calls via the REST API from your own (web) application;
- the **Contacts REST API** to manage your contacts from your own middleware solution;
- the **configurable webhook** to monitor calls.

Developers get to develop to a stateless REST interface instead of a stateful web socket, and are able to monitor users through a webhook using either the GET or POST HTTP method. This allows for easier integration with common low-code solutions like Salesforce Lightning or Microsoft Power Apps.

## REST APIs Documentation

Developers have access to the OpenAPI specification of our APIs via Swagger UI from inside the EasyConnect app.

Swagger UI allows developers to test the API without having to write a single line of code.

To access Swagger UI and test some API calls, follow these steps:

1. Open the **EasyConnect** app
2. Click on **API Docs**
3. Click on **Authorize**
4. Enter one of the configured API keys
5. Click on **Authorize**
6. Click on **Close**
7. You are now authorized and can try out API calls


Open your browser and go to <https://swagger.io/tools/swagger-ui/> to learn more about Swagger UI.

Alternatively, developers can download the OpenAPI specs of the EasyConnect API by appending **swagger/api.yaml** to the App URL.

The App URL can be found in the EasyConnect service settings.

Follow these steps to find out what your App URL is:

1. Open the **AP Manager** app
2. Open the **EasyConnect** app
3. Click **Instances > Edit**
4. You'll find the App URL at the bottom

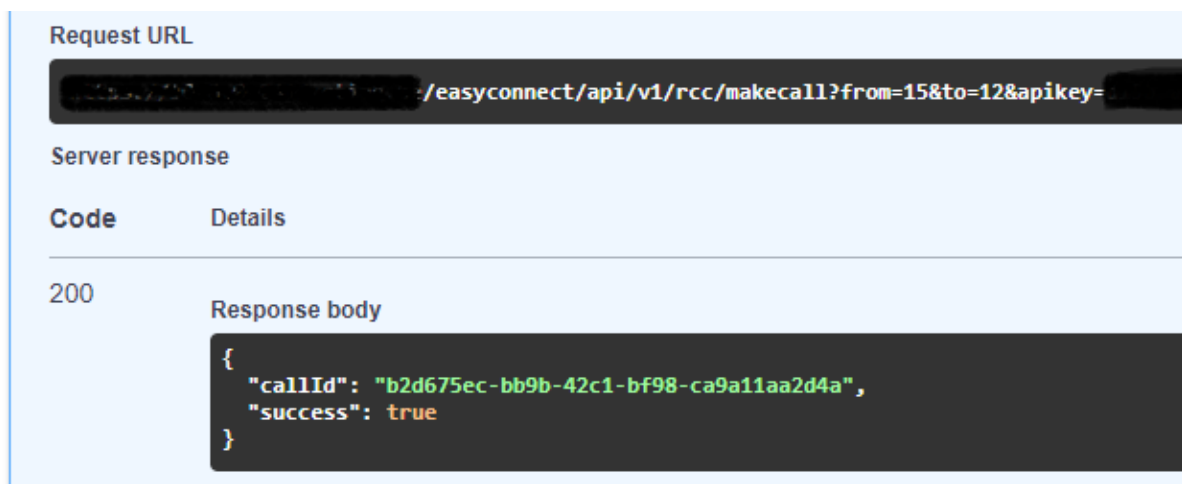
App URLs (e.g. for the PBX App object):  
/xqting.be/easyconnect/xqting-easyconnect

## Consuming the REST APIs

Consuming the REST APIs is easy.

All you need to do, is find the API call you need and pass one of the API keys configured in the EasyConnect settings along as the **apikey** query parameter.

As described in the previous section, the API documentation can be found in the EasyConnect app, or can be download as an OpenAPI spec file. All information required for to make a call is in that documentation.



The screenshot displays an API client interface with the following sections:

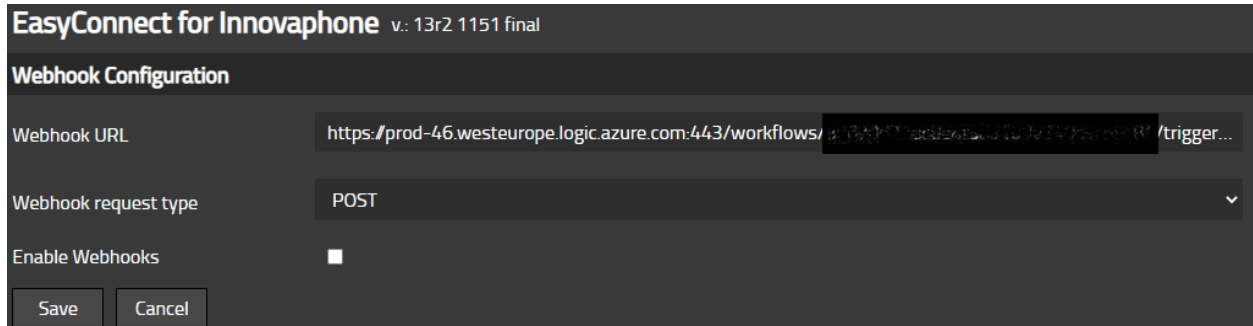
- Request URL:** A text box containing the URL: `/easyconnect/api/v1/rcc/makecall?from=15&to=12&apikey=` (the API key is redacted).
- Server response:** A table showing the response details.

Code	Details
200	<p><b>Response body</b></p> <pre>{   "callId": "b2d675ec-bb9b-42c1-bf98-ca9a11aa2d4a",   "success": true }</pre>

## Monitor Users through a Webhook

To monitor a user, make sure it is a member of the group of which the EasyConnect app is an active member (see [EasyConnect Group](#)).

As soon as webhooks are enabled, the webhook URL is invoked when a monitored user makes or receives a call.



The screenshot shows the 'Webhook Configuration' window for 'EasyConnect for Innovaphone v.: 13r2 1151 final'. It contains the following fields and controls:

- Webhook URL:** A text input field containing the URL 'https://prod-46.westeurope.logic.azure.com:443/workflows/...' followed by a truncated path and '/trigger...'. A redacted section is visible between the domain and the final path segment.
- Webhook request type:** A dropdown menu currently set to 'POST'.
- Enable Webhooks:** A checkbox that is currently unchecked.
- Buttons:** 'Save' and 'Cancel' buttons at the bottom left.

When the **Webhook request type** is set to **POST**, the event arrives as a JSON object and when it set to **GET** all event information is passed via query parameters.

There is a slight difference between the values passed in the JSON object and in the query parameters for Booleans. The JSON object makes use of true or false, while the query parameter uses the string 'yes' or 'no'.

Figure 2 - Example WebHookEvent JSON object

```
{
  "callId": "bfcbbaae-6b35-467d-8ce2-8d986d01eebe",
  "conferenceEvents": [
    {
      "api": "RCC",
      "call": 5,
      "call-id": "2e48670cd75d6501aebd009033600110",
      "cn": "Trunk Line",
      "conf-id": "e833090cd75d6501a8bd009033600110",
      "guid": "97c14e97d4735e0153c4009033402a58",
      "mt": "CallAdd",
      "state": 0
    },
    {
      "api": "RCC",
      "call": 5,
      "local": {
        "e164": "5099",
        "h323": "XQ6100"
      },
      "msg": "x-setup",
      "mt": "CallUpdate",
      "remote": {
        "e164": "00477908475",
        "h323": ""
      }
    },
    {
      "api": "RCC",
      "call": 5,
      "local": {
        "e164": "5099",
        "h323": "Routing WQ"
      },
      "msg": "r-alert",
      "mt": "CallUpdate",
      "remote": {
        "e164": "00477908475",
        "h323": ""
      }
    }
  ],
  "conferenceId": "577193ccf44465016826009033600110",
  "connected": false,
  "conversationTime": -1,
  "endTime": null,
  "from": "15",
  "fromName": "Sam Van Belle",
  "monitoredExtension": "15",
  "monitoredUser": "Sam Van Belle",
  "outgoing": true,
  "pickupTime": null,
  "startTime": "2022-09-16 11:57:43 (UTC)",
  "status": "ringing",
  "to": "12",
  "toName": "",
  "totalTime": -1
}
```

The following table gives more information on the WebHookEvent's JSON object and can be used to understand the query parameters, with a slight difference for the Boolean values as described earlier.

*Table 1 - Description of WebHookEvent JSON object properties*

Property	Description	Example
callId	<p>A string that represents a unique call identifier.</p> <p>This ID is used when making requests to the RCC API. To disconnect a call for example.</p>	1ea84350-ef40-49c4-a3cd-b1d59c821b47
conferenceEvents	<p>An array containing conference events.</p> <p>There are different kinds of conference events. The events have different properties based on the type of event.</p> <p>This could be used in niche cases where the from field would be altered due to a redirect. The conference event would contain the original from number.</p>	See <a href="#">Figure 2 - Example WebHookEvent JSON object</a>
conferenceId	<p>A string that represents a conference identifier.</p> <p>The innovaphone conference ID is the ID used by innovaphone to link call legs to one another.</p>	577193ccf44465016826009033600110
connected	<p>A Boolean that represents the connection status of the call.</p> <p>One of [true, false]</p> <p>This flag in combination with the status gives more context to the call. For example, when receiving a disconnected event (an event where the status is set to 'disconnected') with</p>	false

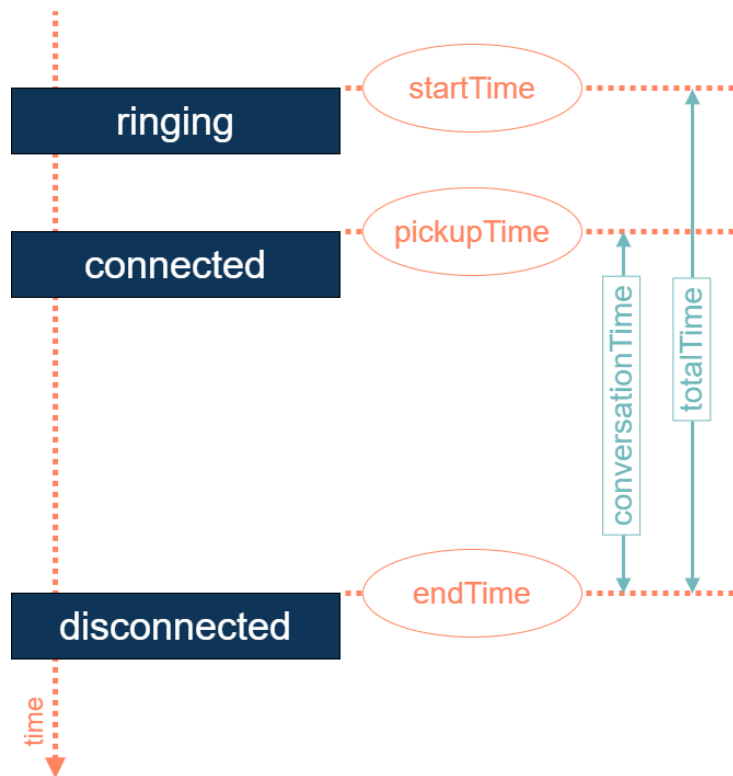
	connected set to false, you can deduce this call is in fact a 'missed call'. We know this to be true because a ringing event's connected flag will always be false, and a connected event's connected flag will always be true.	
conversationTime	A number that represents the duration of the conversation in seconds.  This property is set to -1 when the call is ringing or when the call did not get connected.	-1
endTime	A string that represents the timestamp of the call disconnecting.  The datetime format is: 'YYYY-MM-DD hh:mm:ss (UTC)'.  This property can be null.	null
from	A string that represents the extension number or phone number of the source user.	'00472591687'
fromName	A string that represents the name of the source user.  Can be an empty string when the name is not known.	''
monitoredExtension	A string that represents the extension number of the monitored user.  The monitored user can either be the source (from) or the destination (to) user.  Events for a monitored source user (outgoing call) include: connected and disconnected.	'42'

	Events for a monitored destination user (incoming call) include: ringing, connected and disconnected.	
monitoredUser	A string that represents the name of the monitored user.  Can be an empty string when the name is not known.	'Thomas De Wilde'
outgoing	A Boolean that represents whether the call is an outgoing call.  One of [true, false]	false
pickupTime	A string that represents the timestamp at which the destination user connected to the call.  The datetime format is: 'YYYY-MM-DD hh:mm:ss (UTC)'.  This property can be null.	null
startTime	A string that represents the timestamp of the start of the call.  The datetime format is: 'YYYY-MM-DD hh:mm:ss (UTC)'.	'2022-09-16 11:57:43 (UTC)'
status	A string that represents the type of event.  One of ['ringing', 'connected', 'disconnected']	'ringing'
to	A string that represents the extension number or phone number of the destination user.	'42'
toName	A string that represents the name of the destination user.	'Thomas De Wilde'



	Can be an empty string when the name is not known.	
totalTime	A number that represents the total time this call took in seconds. This property can be -1.	-1

Figure 3 - Timings and Durations of a WebhookEvent



To illustrate the power of simplicity that comes with the webhook event, we'll go over the events that are triggered for an 'incoming missed call'.

As is clear from [Figure 3 - Timings and Durations of a WebhookEvent](#), EasyConnect is elegantly simple by sending just three distinct webhook events: ringing, connected and disconnected.

The first event sent for an incoming call for a monitored user is the ringing event. This event indicates that someone initiated a call. The connected property of a ringing event will always be set to false, because there's no connection yet between source (from) and destination (to).

The second and final event for a missed call is the disconnected event. To know if the call is a missed call, developers need to verify the connected flag of a disconnected event. When it is set to false, the call is a missed call. When it is set to true, the call went through. We know this to be

true because the only time the `connected` property is set to true, is when the `connected` event is triggered.

Alternatively to identify a call as being a missed call, one could check the conversation time, but a Boolean flag usually is easier to implement and reads better.

Figure 4 - Example Webhook JSON data for incoming missed call



If the monitored user would have picked up the call, a third Webhook event would have been sent. The `'connected'` event indicates that a connection is established between source and destination.

See [Figure 5 - Example Webhook JSON data for incoming call](#) for an example of Webhook events of a picked up call.

Figure 5 - Example Webhook JSON data for incoming call



# Support

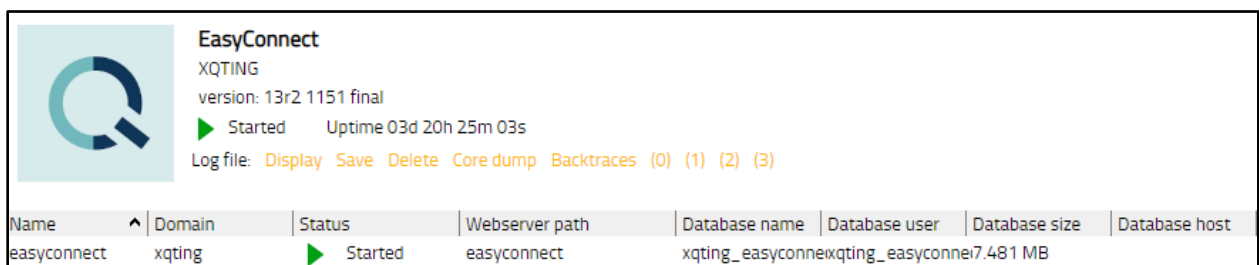
## Creating a Log File

For faster resolution of technical issues, we kindly request to include a log file in your support requests.

Follow these steps to configure the app logger:

1. Open the **AP Manager** app
2. Open the **EasyConnect** app
3. Select your instance
4. Click **Manage > Diagnostics**
5. Enable 'App' and 'App WebSocket'
6. Click **Save**

Before reproducing the error, clear the log file by clicking on **Delete** next to **Log file**.



Now reproduce the error.

After reproducing the error, go back to the EasyConnect app, and click on **Save** next to **Log file**.

Include the log file in your support request.

## Feature Requests

Please send an email to [support@xqting.be](mailto:support@xqting.be) to request missing features and functionality.

## Contact Information

Please send an email to [support@xqting.be](mailto:support@xqting.be) to request help.

We kindly request to include as much relevant information as possible (e.g. versioning information) and provide us with a log file when applicable (check out [Creating a Log File](#) to learn more).

Please send an email to [support@xqting.be](mailto:support@xqting.be) to report mistakes in this manual.