[Desk] Email Support

Date: 2021-10-06

Prepared By: Solutions Engineering

Background

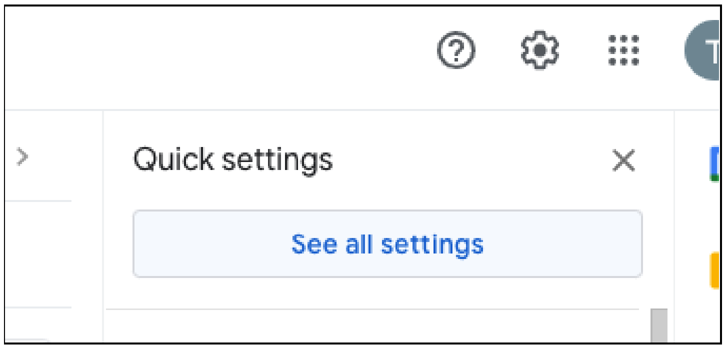
- This document describes how customers can implement a Server for sending and receiving Desk Tickets from Email.

Tools used

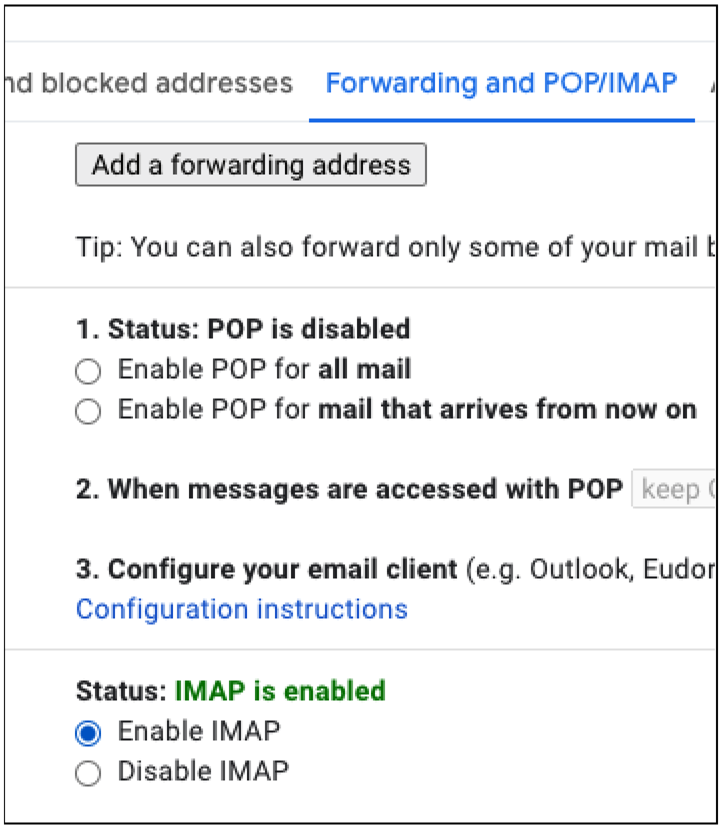
- You will need a valid Sendbird account with Desk support.
- We will use NGROK for local testing. You will need a live server for your customers.
- Make sure you have NodeJS and NPM installed and working properly. NodeJs is the programming language I will be using for this demo but you can implement any of your choice.
- No frontend application is needed since the tickets created from Email will appear in the Dashboard.

Setting Gmail

- This example works with any Email provider supporting IMAP. I will use Gmail for this example so some configuration settings should be made.
    - **Enable IMAP**: You can do that by going to Settings > See all settings



- ○ Go to Forwarding and POP/IMAP and enable IMAP



nd blocked addresses | Forwarding and POP/IMAP

Add a forwarding address

Tip: You can also forward only some of your mail b

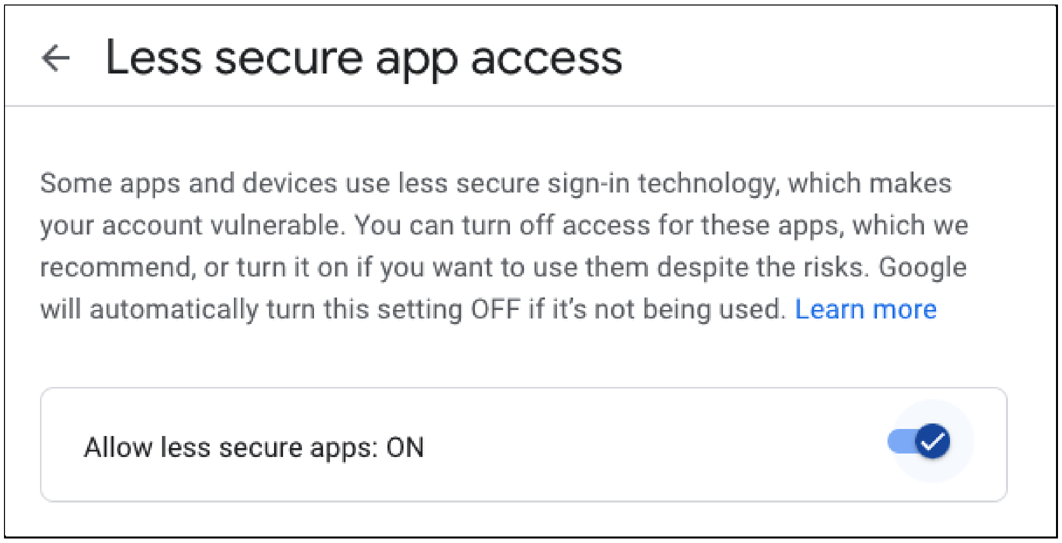
1. Status: POP is disabled
- ◯ Enable POP for **all mail**
- ◯ Enable POP for **mail that arrives from now on**

2. When messages are accessed with POP keep (

3. Configure your email client (e.g. Outlook, Eudor
Configuration instructions

Status: IMAP is enabled
- [✓ Enable IMAP]
- ◯ Disable IMAP

- You will also need to allow access for **less secure apps**. By doing this, Gmail will allow you to login using your Email and password from your own application. To do that, go to the following URL: https://myaccount.google.com/lesssecureapps



Less secure app access

Some apps and devices use less secure sign-in technology, which makes your account vulnerable. You can turn off access for these apps, which we recommend, or turn it on if you want to use them despite the risks. Google will automatically turn this setting OFF if it's not being used. Learn more

Allow less secure apps: ON

Make sure the option is set to ON.

Clone the code

- Once you have Gmail (or any other of your choice) ready, you can clone the code for this example and set your configuration. The repository is located here: https://github.com/warodri-sendbird/desk-email-nodejs

- Open the **config.js** file and add the Email and password you will use for receiving Emails from your customers:

```
const params = {
   email: 'your-email@gmail.com',
   password: 'YOUR-PASSWORD-HERE'
}
module.exports = params;
```

- Open the **my-desk-account.js** file and add your Sendbird Desk information:

```
const myDeskAccount = {
   email: 'your-email@gmail.com',
   checkSubject: true,
   subjectMustContain: 'Desk',
   sb: {
       appId: 'YOUR SENDBIRD APPLICATION WITH DESK FEATURE ENABLED',
       deskToken: 'YOUR DESK TOKEN',
       apiToken: 'ANY API-TOKEN FOR SENDING PLATFORM API REQUESTS',
       deskEnabled: true
   },
   ticket: {
       welcomeMessage: 'Ticket created from Email',
       instructionForEmailFooter: `<br><br><i>Please respond to this message
without changing the subject.</i>`
   }
};
module.exports = myDeskAccount;
```

email: Enter here the same email you defined inside the **config.js** file.
checkSubject: Set to **true** if you want to check if the subject of the email message contains some specific words to consider as a new Desk ticket.
subjectMustContain: If you selected true for the previous option, the system will check for this string in the subject of the message before considering it a Desk ticket. This means customers should include this word(s) in their subject when creating tickets.
sb: This is all the information from your Sendbird and Sendbird Desk account we will need for sending requests on your behalf.

Starting Email listener

- Once all the previous configuration is done, you can start the server for listening to Emails arriving. They will be analyzed and new tickets will be created.

```
node start mail-notifier.js
```

If all goes well, this server should listen for new Email.
In case of some error, please make sure the steps for Gmail are correct. The problem you will have at this stage is not connecting to the IMAP server.

Adding webhook for Platform API

- We will need to be notified for events related to chat messages. For that we use the **server.js** file.

- The NodeJS library we use for this is nodemailer. But of course you can use any other of your choice. If using Gmail with nodemailer we initialize it like this:

```
var nodemailer = require('nodemailer');
var emailSender = nodemailer.createTransport({
   service: 'gmail',
   auth: {
     user: config.email,
     pass: config.password
   }
});
```

  You can see this by editing the **server.js** file. Please, feel free to change if you are not using Gmail. You can specify our IMAP server there.

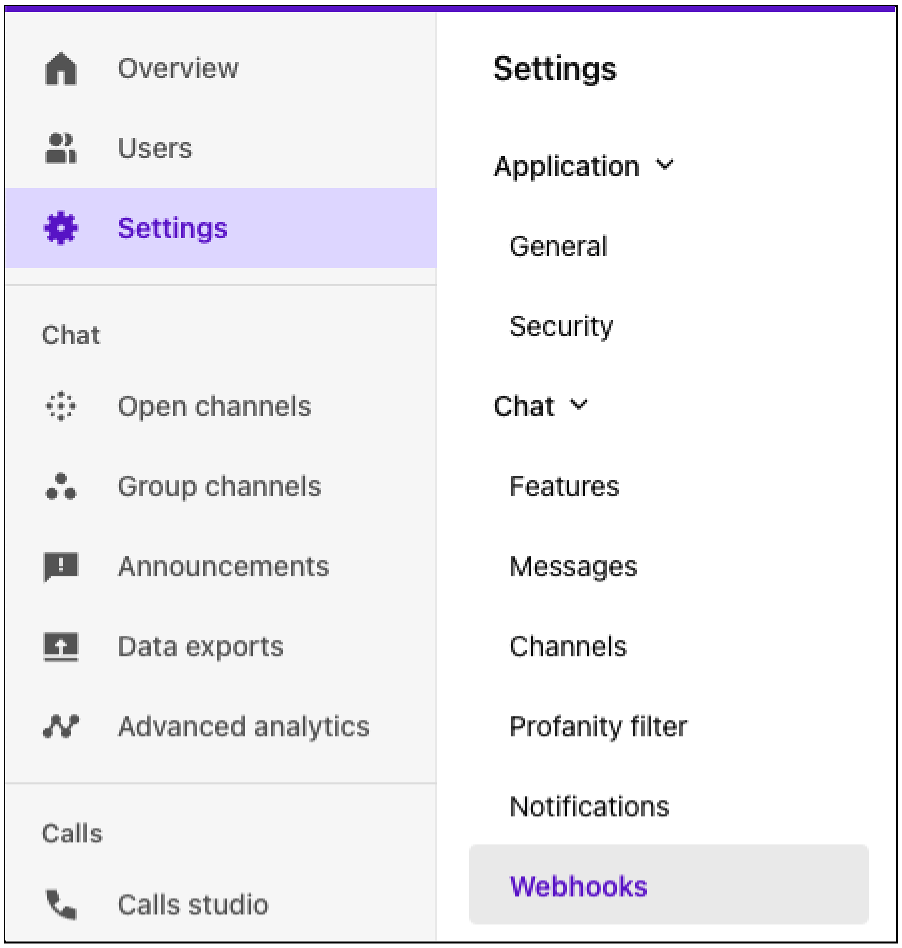
- We will use NGROK for exposing this local server to the Internet. In production, you should host this server properly. The use of a Load Balancer is recommended.
- Run the server using the following command:

```
node server.js
```

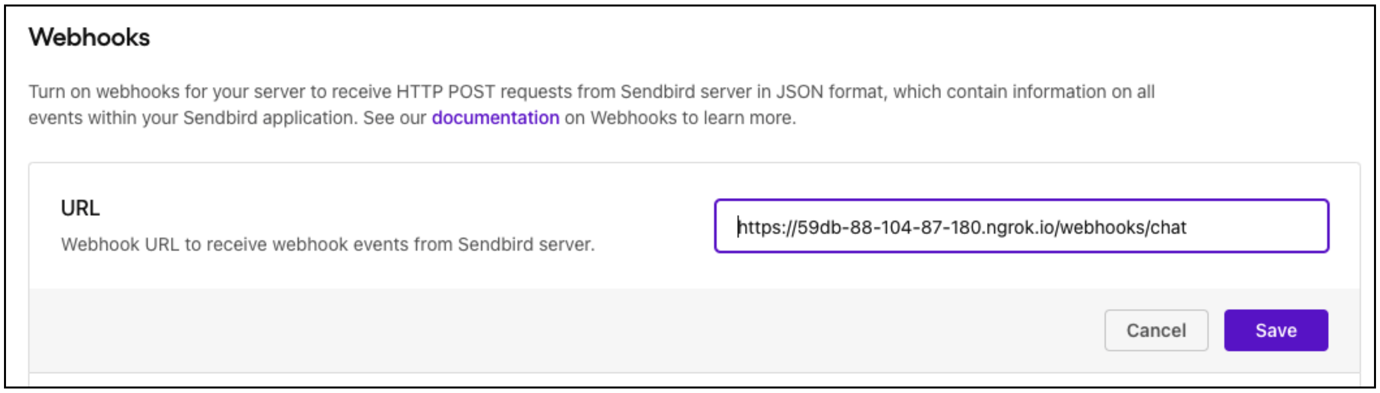
- Once the server is running and listening on port 3000, you can run **NGROK** with the following command:

```
ngrok http 3000
```

- NGROK will give you a temporary Internet address. Something like **https://59db-88-104-87-180.ngrok.io**
- Open your Sendbird Dashboard and go to Settings > Chat > Webhooks



- Enter the URL NGROK gave you, followed by /webhooks/chat


Webhooks

Turn on webhooks for your server to receive HTTP POST requests from Sendbird server in JSON format, which contain information on all events within your Sendbird application. See our documentation on Webhooks to learn more.

URL

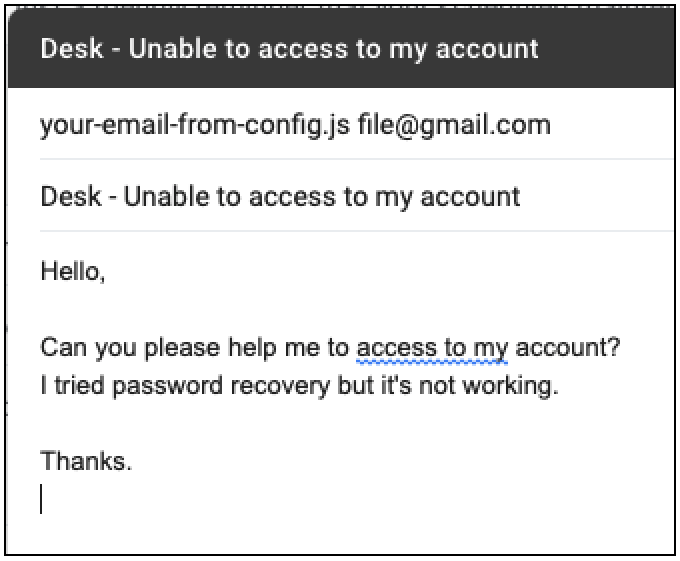
Webhook URL to receive webhook events from Sendbird server.

https://59db-88-104-87-180.ngrok.io/webhooks/chat

And make sure you hit the SAVE button.

You’re ready to try!

- Send an Email message to the address you defined in your **config.js** file. Make sure you include the word **Desk** in your subject.



Desk - Unable to access to my account

your-email-from-config.js file@gmail.com

Desk - Unable to access to my account

Hello,

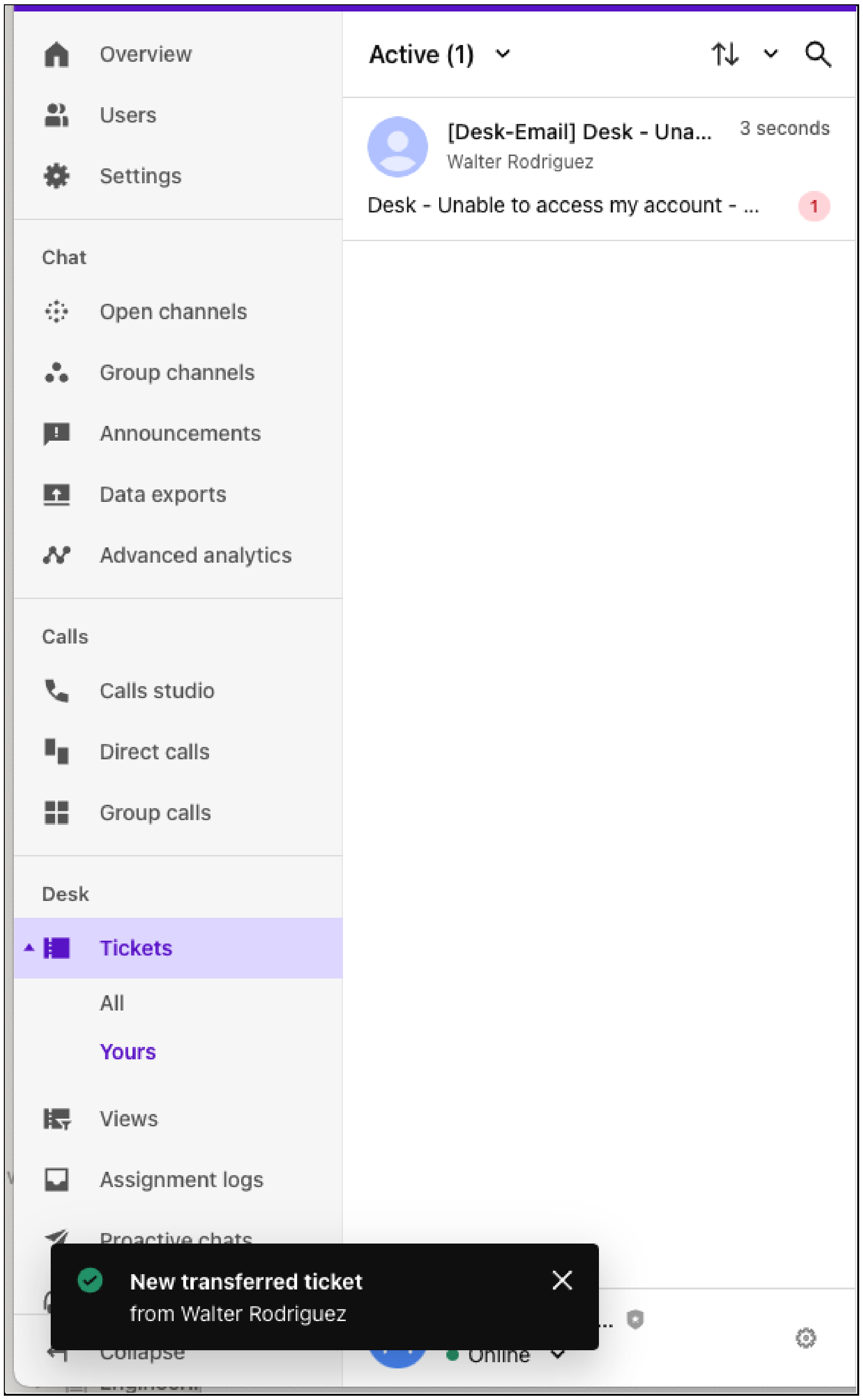
Can you please help me to access to my account?
I tried password recovery but it's not working.

Thanks.

Replace **your-email-from-config.js file@gmail.com** with the real address from your **config.js** file.

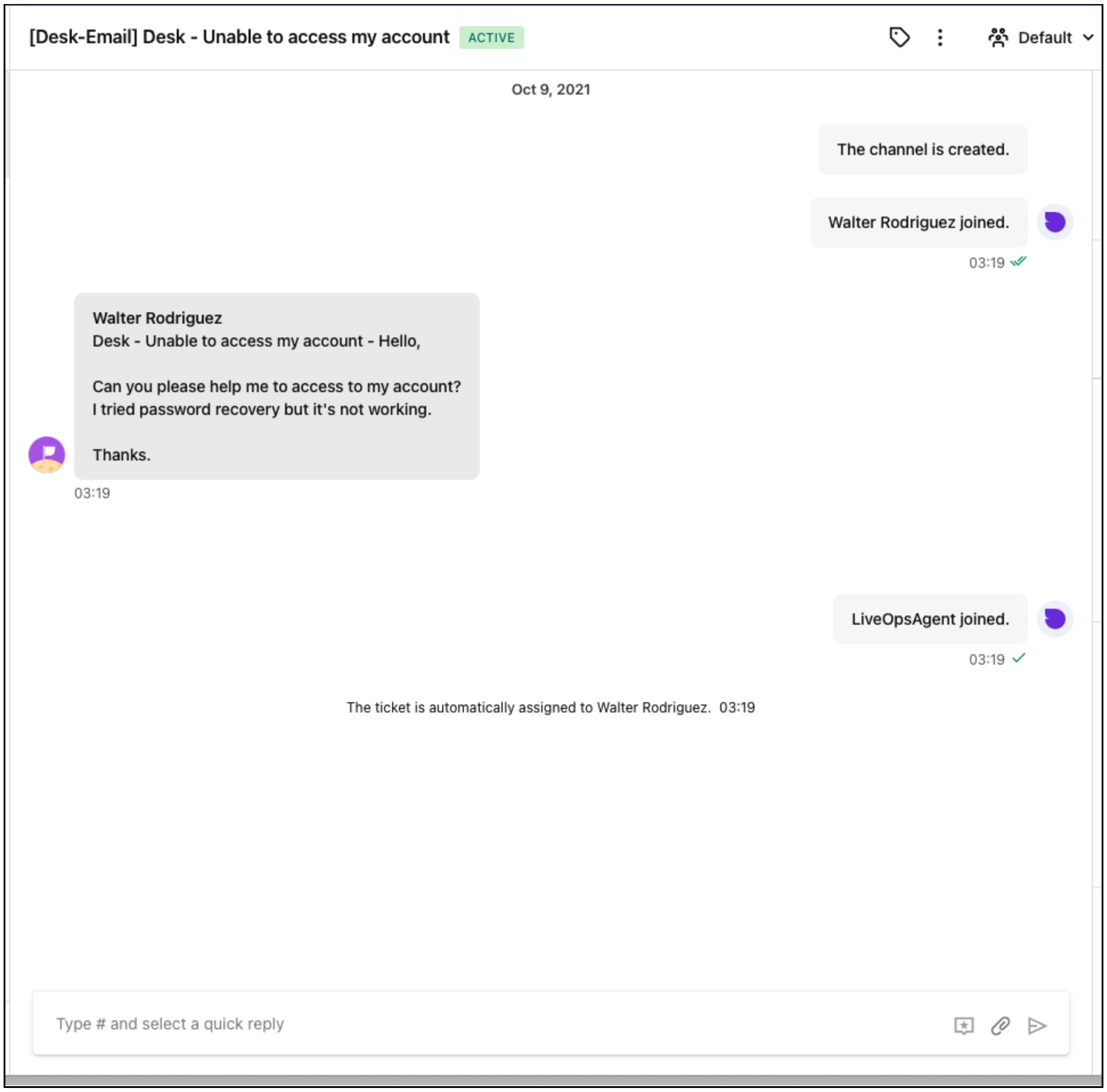
Make sure you include the word **Desk** in the subject

- If all goes well, after some seconds you send this email, the server will read the new Email message, create a ticket and show it in your Dashboard!



[Desk-Email] Desk - Una... 3 seconds
Walter Rodriguez
Desk - Unable to access my account - ... 1

New transferred ticket
from Walter Rodriguez



[Desk-Email] Desk - Unable to access my account ACTIVE

Default

Oct 9, 2021

The channel is created.

Walter Rodriguez joined.

03:19

Walter Rodriguez
Desk - Unable to access my account - Hello,

Can you please help me to access to my account?
I tried password recovery but it's not working.

Thanks.

03:19

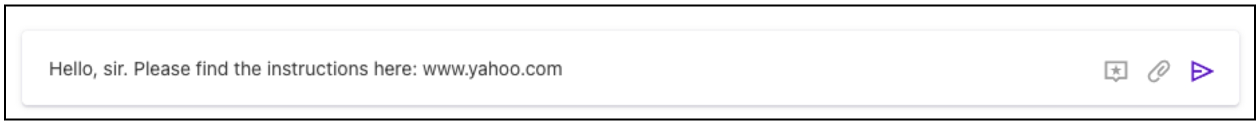
LiveOpsAgent joined.

03:19

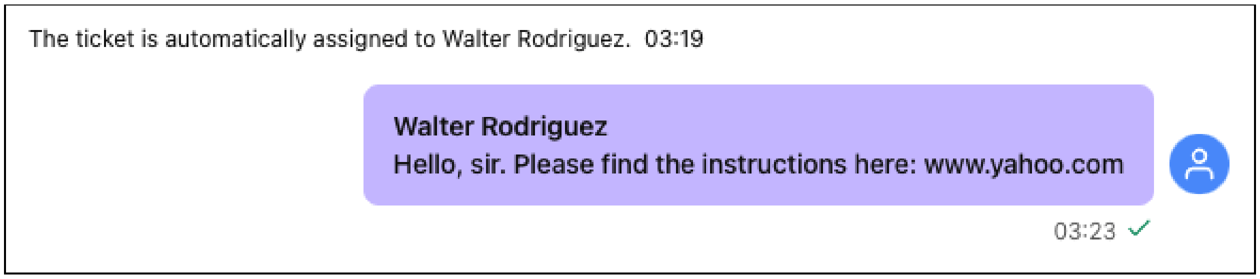
The ticket is automatically assigned to Walter Rodriguez. 03:19

Type # and select a quick reply

- Try writing a response to this ticket (simulating an Agent responding to the chat)



Hello, sir. Please find the instructions here: www.yahoo.com

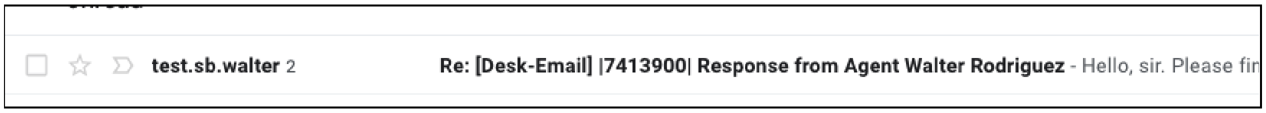


The ticket is automatically assigned to Walter Rodriguez. 03:19

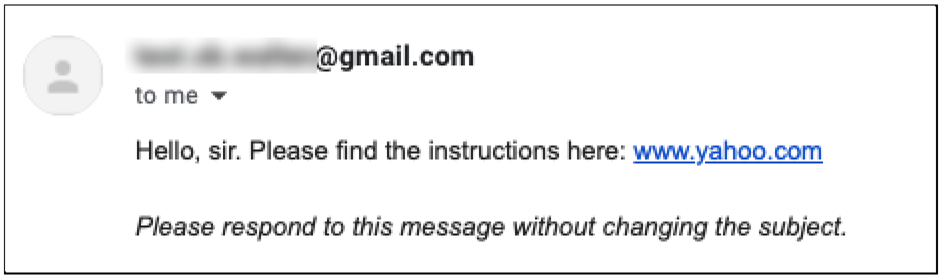
Walter Rodriguez
Hello, sir. Please find the instructions here: www.yahoo.com

03:23 ✓

- If all goes well, customer will receive a reply in the Email



test.sb.walter 2 Re: [Desk-Email] |7413900| Response from Agent Walter Rodriguez - Hello, sir. Please fin



[redacted]@gmail.com
to me

Hello, sir. Please find the instructions here: www.yahoo.com

Please respond to this message without changing the subject.

This is how your customer will see the response from Agents.

Note the label: **“Please respond to this message without changing the subject.”**
This is optional and can be modified / removed from the **my-desk-account.js** file.