Cisco Live! Amsterdam | February 9-14, 2025

GO BEYOND

LTRCCT-2813

Mastering Demo-Driven Customer Journeys with Webex Customer Experience

Seb Huber - Solutions Engineer
Christian Wolf - Solutions Engineer
Jan Jäkel - Solutions Engineer

Table of Contents

Disclaimer

This training document is to familiarize with Webex Customer Experience. Although the lab design and configuration examples could be used as a reference, it's not designed for any real, production-environment deployment, and thus not all recommended features are used, or enabled optimally. For the design related questions please contact your representative at Cisco, or a Cisco partner.

Scenario

In this lab activity, you will step into the role of a **Solutions Engineer** at **Better Bikes Inc.**, a leading (fictitious) company in the cycling industry. Your mission is to create a seamless and efficient **customer journey** using **Cisco Webex Customer Experience** solutions, helping your company deliver outstanding support and engagement.

Your journey begins with a well-structured **voice flow**, powered by **Webex Contact Center (WxCC)**, ensuring that customers can easily connect with your support team. However, modern customer expectations go beyond traditional call centers. To enhance accessibility and convenience, you will implement a **Web Chat widget** on your company's website, allowing customers to ask questions and receive assistance in real time.

As digital transformation continues to shape customer interactions, offering **multichannel support** becomes essential. You will take the next step by integrating **WhatsApp**, making it even easier for customers to engage with your company on the channels they use daily. This omnichannel approach not only improves response times but also creates a more personalized and efficient customer experience.

By the end of this lab, you will have a comprehensive understanding of the various **components, tools, and applications** involved in building a modern customer support system. While there is no universal, one-size-fits-all blueprint, this experience will equip you with the knowledge and insights to explore **customized solutions** that best fit the needs of your company and/or customers.

Module 0 – Preparation

To ensure that the exercises run smoothly, we have already configured some elements in advance, we detail these from Module 0.1 to 0.8, please note that 0.9 requires you to follow some steps to create an account and order for the lab.

Please note that throughout the documentation we will use the term 'PodXX_...' as an identifier, in which the 'XX' indicates where you need to replace the X's with the number of the pod you are issued with.

0.1 User Accounts

For each pod, we have configured 3 types of users "PodXX_User", "PodXX_Agent", and "PodXX_Super".

"C1sc0Live25!" has been set as password for every account.

"PodXX_User" will be used for all configurations and administrative tasks.

"PodXX_Agent" is used for Contact Center Agent Desktop login.

"PodXX_Super" is the Supervisor account of your Contact Center.

These users have different permissions in Control Hub and Contact Center.
The following permission and functions have been assigned:

	PodXX_User	PodXX_Agent	PodXX_Super
Control Hub Admin	X		
Webex Contact Center Admin	X		
Premium Contact Center Agent		X	
Contact Center Supervisor			X
Webex Connect Admin	X		
Webex Engage Admin	X		
webmail.webex-lab.com	X		
glitch.com	X		

0.2 PSTN Numbers

To enable inbound dialing via PSTN, we have provided public phone numbers for your PodXX_User and PodXX Service Queue. The numbers provided for your pod can be found in the following table:

Pod Number	PodXX_User	PodXX Service Queue
01	+442046200494	+442046200496
02	+442046200501	+442046200506
03	+442046200507	+442046200517
04	+442046200523	+442046200524
05	+442046200526	+442046200527
06	+442046200528	+442046200530
07	+442046200532	+442046200533
08	+442046200534	+442046200538
09	+442046200549	+442046200629
10	+442046200852	+442046200941
11	+442046200943	+442046200947
12	+442046200949	+442046200951
13	+442046200952	+442046200954
14	+442046200955	+442046200956
15	+442046200961	+442046200974
16	+442046200976	+442046200977
17	+442046200978	+442046200980
18	+442046200981	+442046200982
19	+442046200986	+442046200989
20	+442046200991	+442046200994

0.3 Webex Connect

Webex Connect, part of Cisco's Webex suite, is a cloud-based platform for enterprise communication and collaboration. It integrates various channels—SMS, email, social media, and chat apps—into one interface, streamlining customer interactions.

The platform supports automated workflows, chatbots, and omnichannel strategies, making it ideal for improving customer service and engagement. With analytics and reporting tools, businesses can enhance communication processes and deliver seamless, consistent customer experiences.

For your reference:

0.4 Webex Engage

Webex Engage, part of Cisco's Webex suite, is a cloud-based platform for enhancing customer engagement through multichannel communication. It supports interactions via voice, video, chat, email, and social media.

Designed for customer service teams, it offers features like automated workflows, chatbots, and CRM integration to deliver personalized, efficient experiences and improve response times. With a unified interface and analytics tools, Webex Engage helps businesses maintain consistent communication, strengthen customer relationships, and optimize engagement strategies.

0.5 Journey Data Service

Customer Journey Data Service (CJDS) is an API-first service that enables organizations to:

Listen: Integrate with any data source or third-party applications to listen to disparate data sources.

Identify: Create a dynamic customer profile capturing propensity drivers, such as a customer's preferred mode of communication or preferred language.

Analyze: Apply different aggregation techniques to all customer data collected.

Act: Use the data and insights within CJDS to dynamically change the flow within Webex Contact Center Flow Control and personalize the customer experience at a granular level.

These insights are visible to customer-facing teams in real time through Agent Desktop.

Customer Journey Data service (CJDS) is a next-generation customer journey management service empowering organizations to go from Data to Insights to Action. CJDS enables businesses to capture customer journeys across any channels/application, identify insights and take real-time actions to provide an excellent customer experience.

CJDS needs a Webex Contact Center OAuth Token to use the API. In this lab we have established a token management. The Documentation can be found on GitHub.

Also, you can find an example of how to inject CJDS here.

0.6 Idle/Wrap-Up Codes

In preparation for presence synchronization, different codes have already been created and assigned to the presence states.

Idle/Wrap-up Codes

Search by name | Active | 8 Idle/Wrap-up Codes | Create Idle/Wrap-up Code

Idle/Wrap-up code	Description	Idle/Wrap-up code type	Default	Status	Last modified
away		Idle Code	No	Active	December 02, 20...
DND		Idle Code	No	Active	December 02, 20...
in a Calendar Mee...		Idle Code	No	Active	December 02, 20...
in a external Call		Idle Code	No	Active	December 02, 20...
in an Meeting		Idle Code	No	Active	December 02, 20...
in presentation		Idle Code	No	Active	December 02, 20...
Meeting	Default idle code	Idle Code	Yes	Active	October 30, 2024...
Wrap Up	Default wrap-up c...	Wrap-up Code	Yes	Active	December 04, 20...

Under Desktop in Contact Center main menu, these codes have been assigned to the appropriate Presence States.

Webex App

You can control access to these features for your agents inside Desktop profiles.

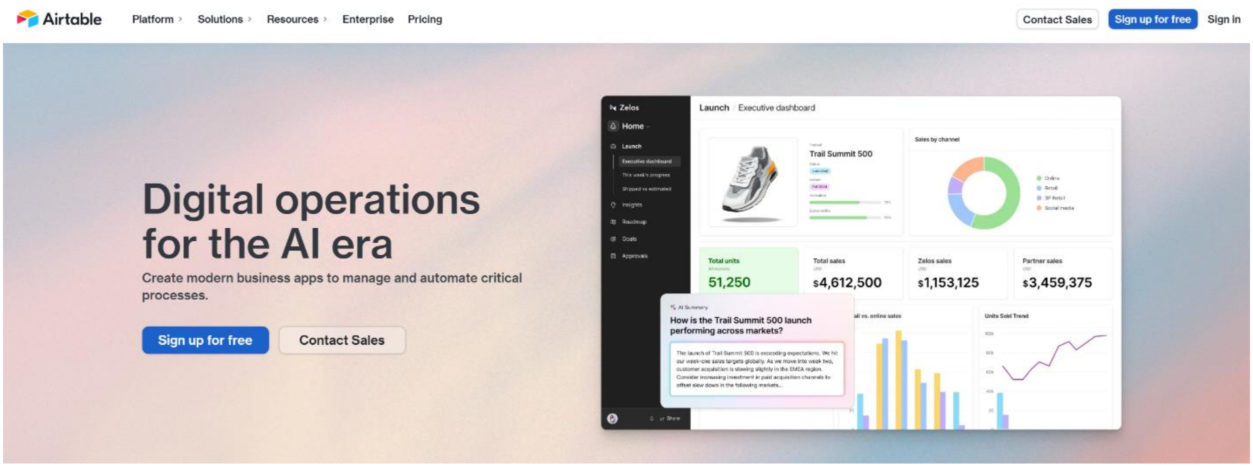
Display user details

State synchronization

Map Webex Availability States with Webex Contact Center's Idle codes. For more information on mapping Click here.

Webex Availability States	Webex Contact Center Idle c...
Do not disturb, Quiet hours	DND
In a calendar meeting	in a Calendar Meeting
In a meeting	in an Meeting
On a call	in a external Call
Out of office	away
Presenting	in presentation

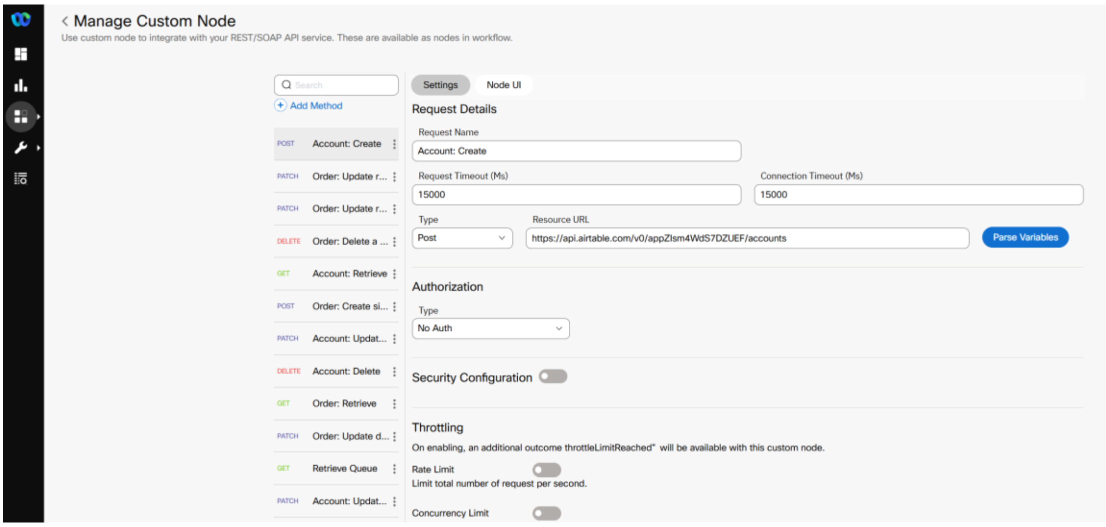
0.7 Airtable



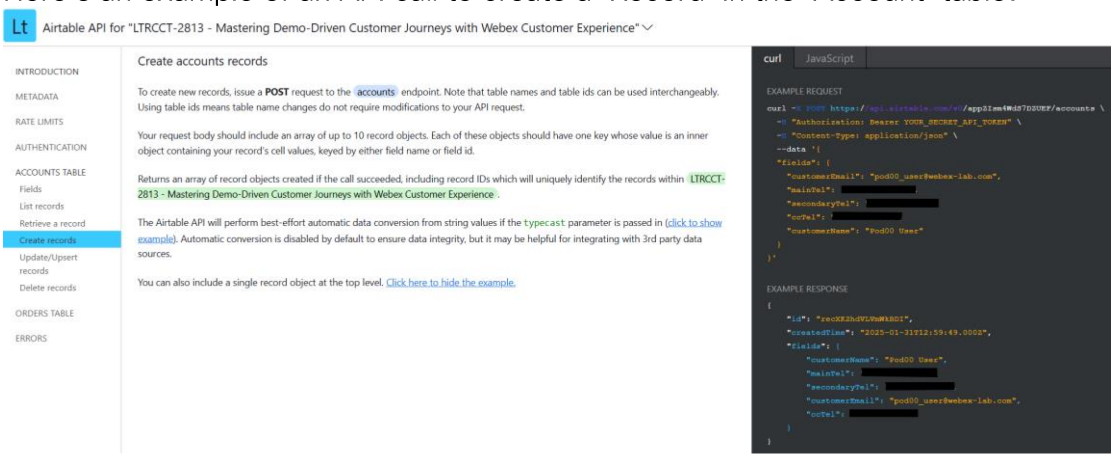
Airtable is a collaboration platform that combines the simplicity and user-friendliness of a spreadsheet with the powerful functionality of a database. Its main purpose is to enable individuals and teams to organize, track, and manage various types of information and projects in a customizable, visually intuitive format. Users can create and share databases that support a range of views (like grid, calendar, gallery, and kanban), facilitating seamless collaboration and integration with other tools and services, making it suitable for a variety of tasks such as project management, content planning, and customer relationship management.

Add that the tool is a cloud-based, open-access solution, makes this an ideal tool for using in lab scenarios, such as our own, and so we have integrated Airtable with our solutions in this lab for the purpose of imitating a database. We have integrated several CRUD (create, read, update & delete) functions into our lab, using Webex Connect's Custom Nodes.

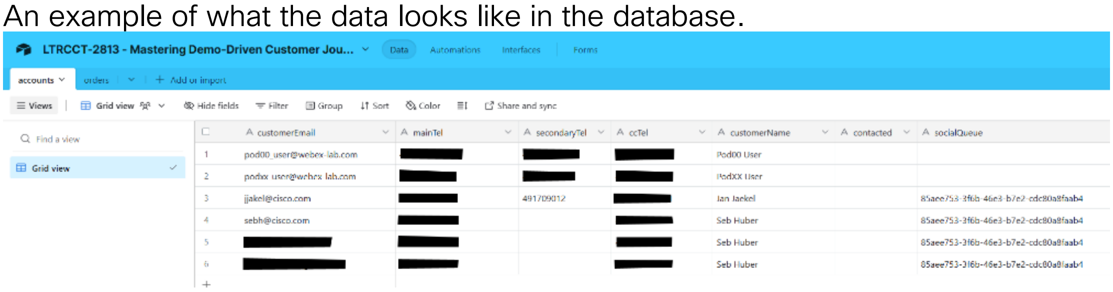
Below, you can see a screenshot of the list of functions, some of which you will use in this lab.



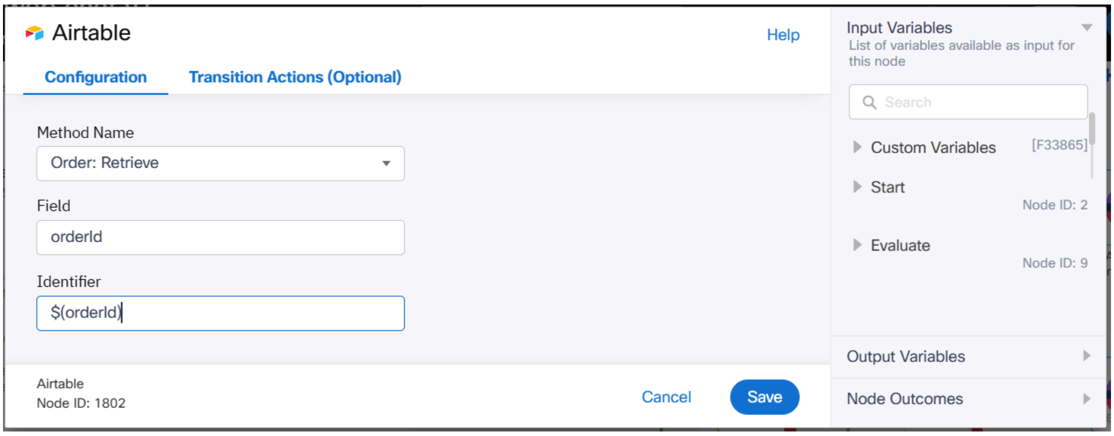
Here's an example of an API call to create a 'Record' in the 'Account' table.



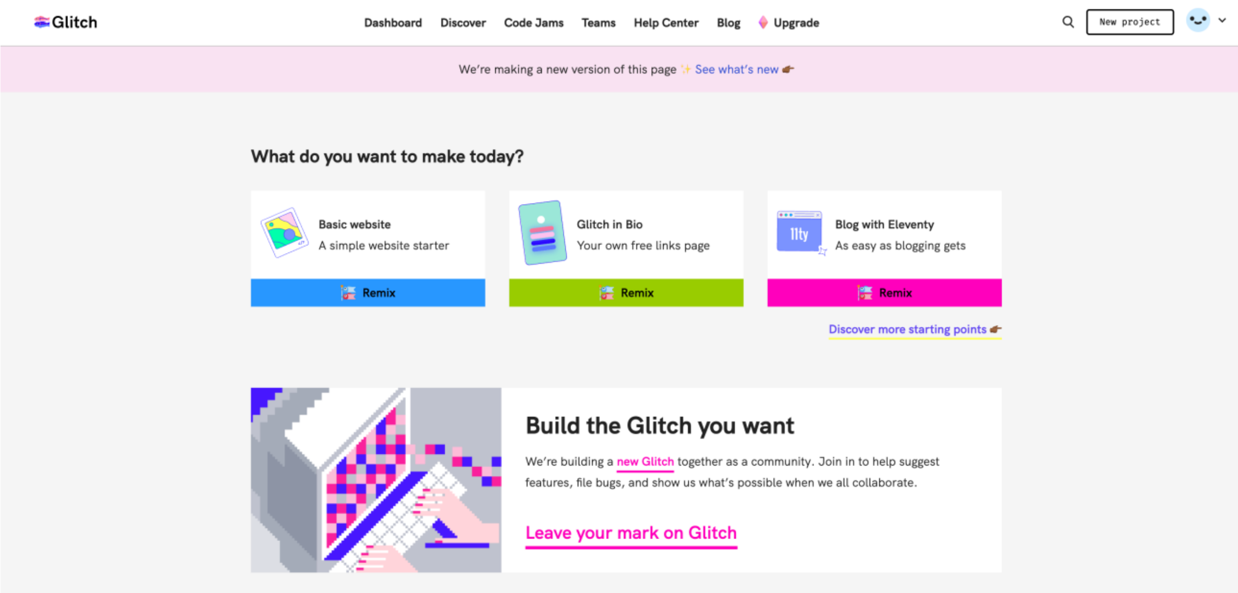
An example of what the data looks like in the database.



An example of what an Airtable, custom node in Webex Connect looks like. Custom nodes are configured, simplifying the process of API calls you use regularly.



0.8 Glitch



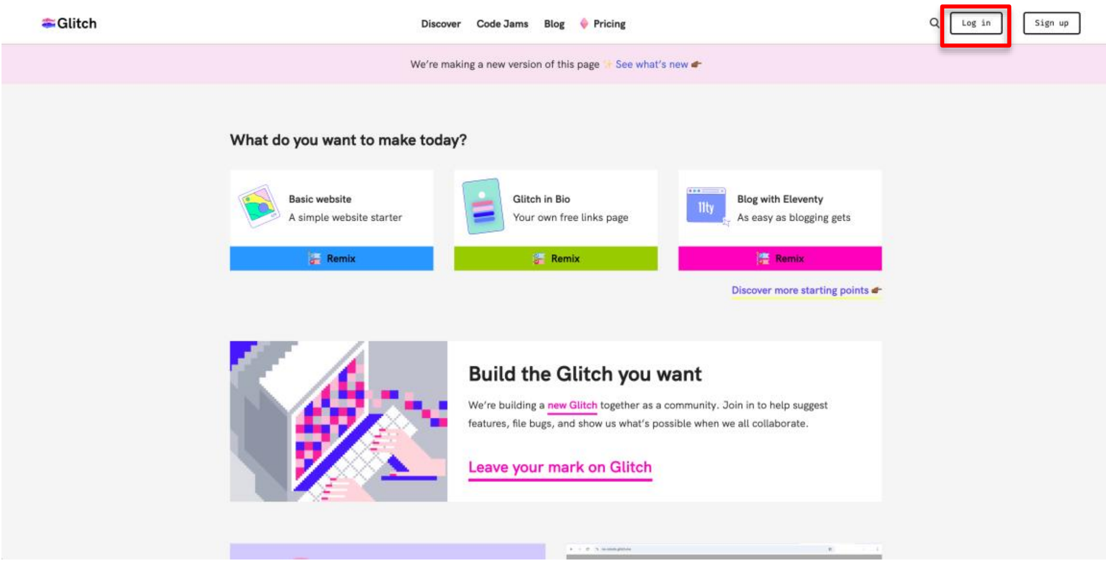
Glitch.com is a platform designed for creating, sharing, and remixing web applications. It offers an intuitive, collaborative environment where developers can build full-stack apps using JavaScript, Node.js, and other web technologies. One of its standout features is the ability to instantly see changes live as you code, making it ideal for experimentation and learning.

Glitch emphasizes community and collaboration, allowing users to "remix" projects, which essentially means creating a copy of someone else's project that you can modify and customize. This fosters a learning environment where users can explore and build upon existing work.

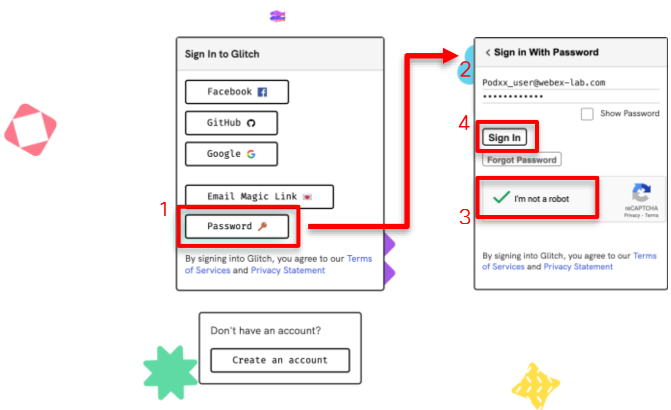
Moreover, Glitch provides hosting for projects, meaning that users can deploy their apps directly from the platform without needing external hosting services. This makes it particularly appealing for both beginners looking to get started with coding and experienced developers wanting to quickly prototype ideas.

0.9 Website creation

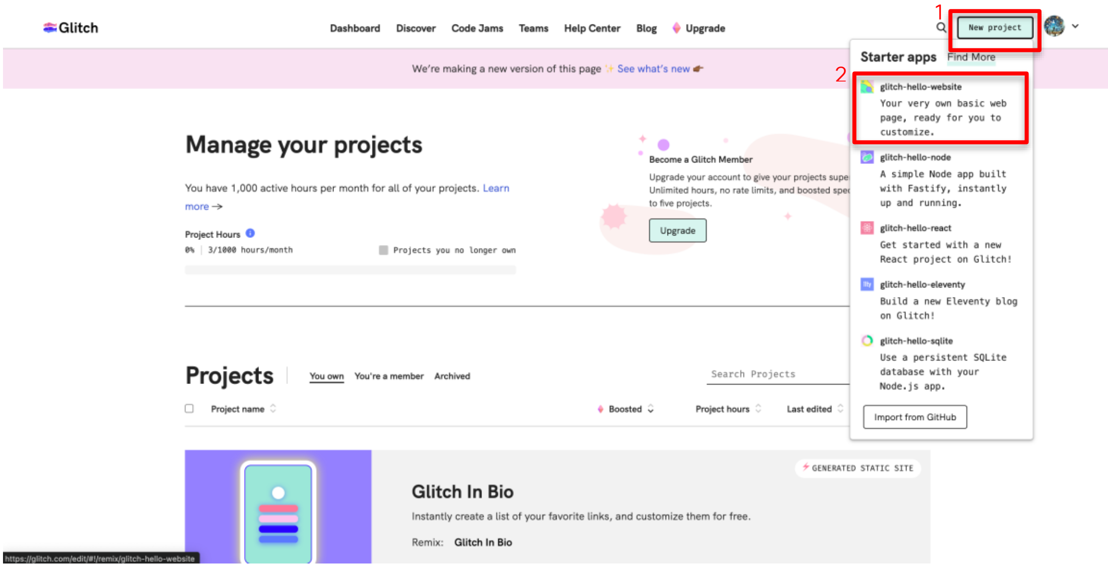
Open your browser and go to https://glitch.com/ and click Log-In on the top right.



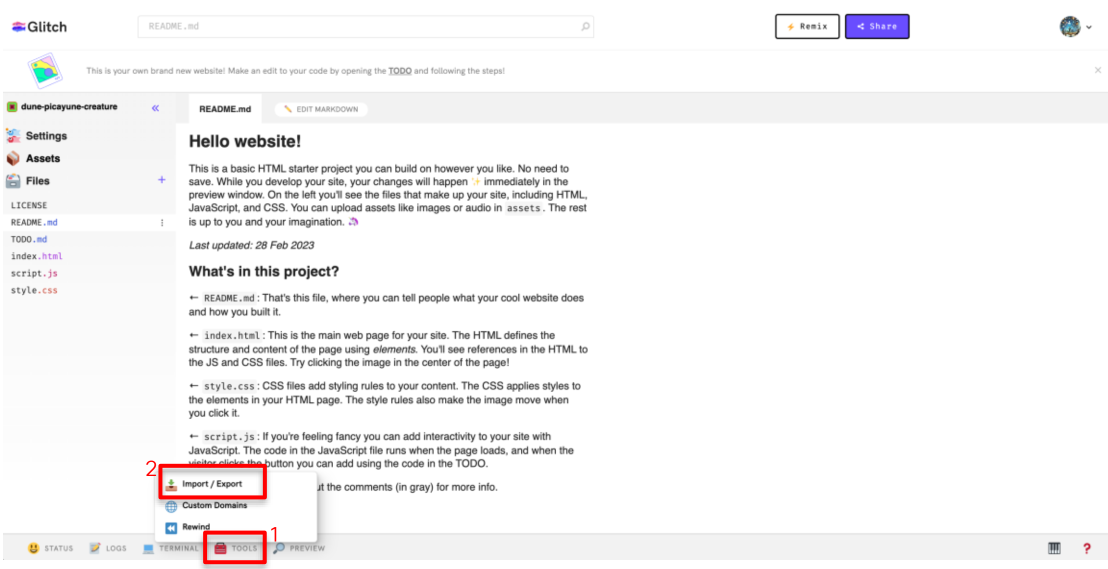
Select Password and enter your PodXX_User credentials, solve the CAPTCHA and click on Sign In.



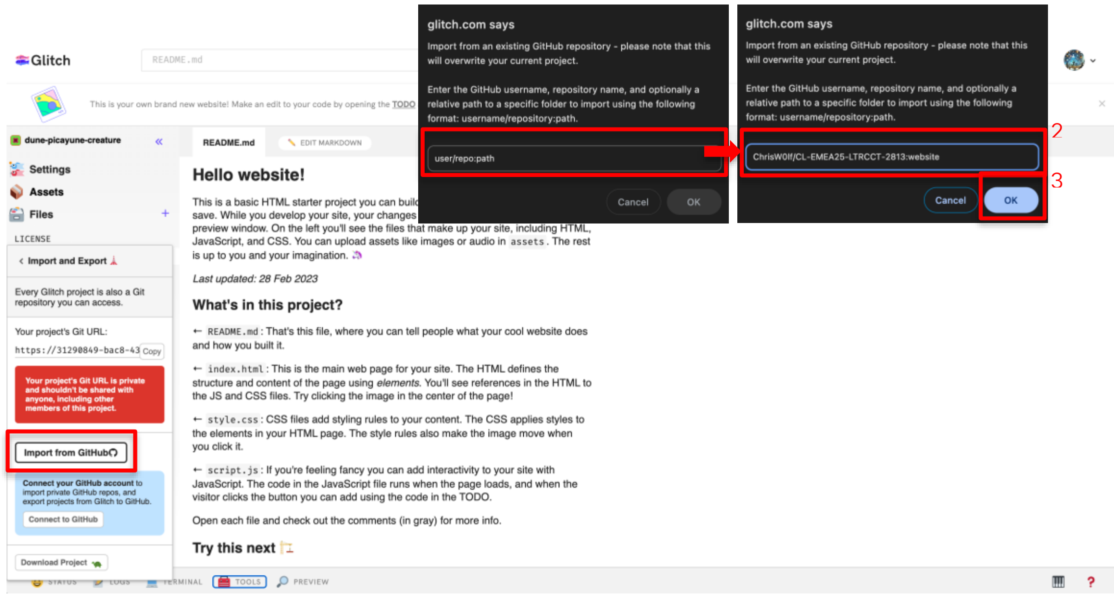
After successfully being logged in, create a new glitch project with clicking on New Project on the top right and select glitch-hello-website.



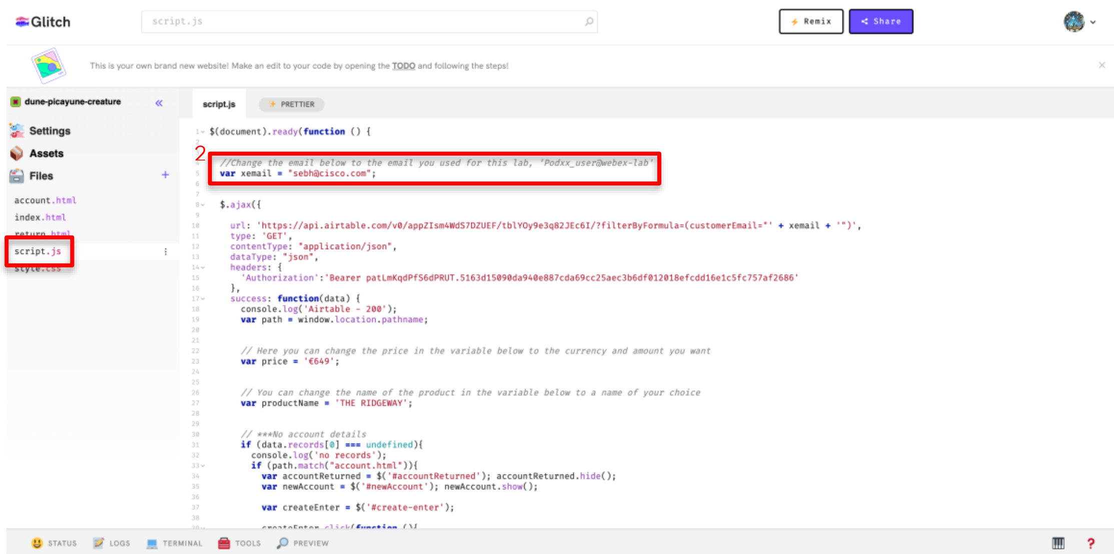
When your new project has been loaded, we need to import Better Bikes Inc's website into our glitch project. Therefore, on the bottom menu bar click on TOOLS and select Import / Export.



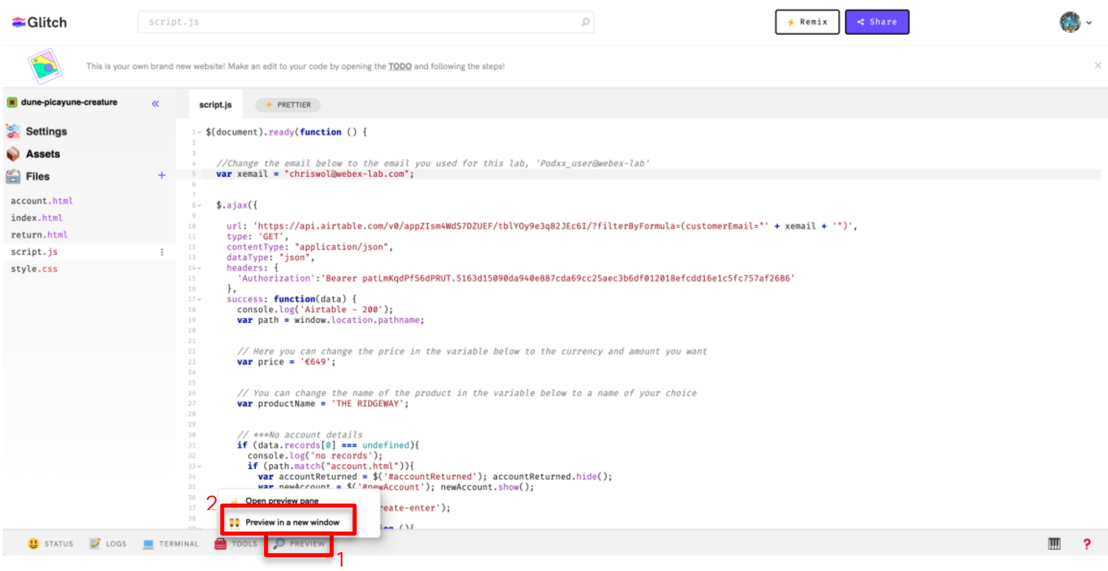
In the popup window on the left use Import from Github and enter ChrisW0lf/CL-EMEA25-LTRCCT-2813:website in the user/repo:path field. Finish the import with clicking on OK.

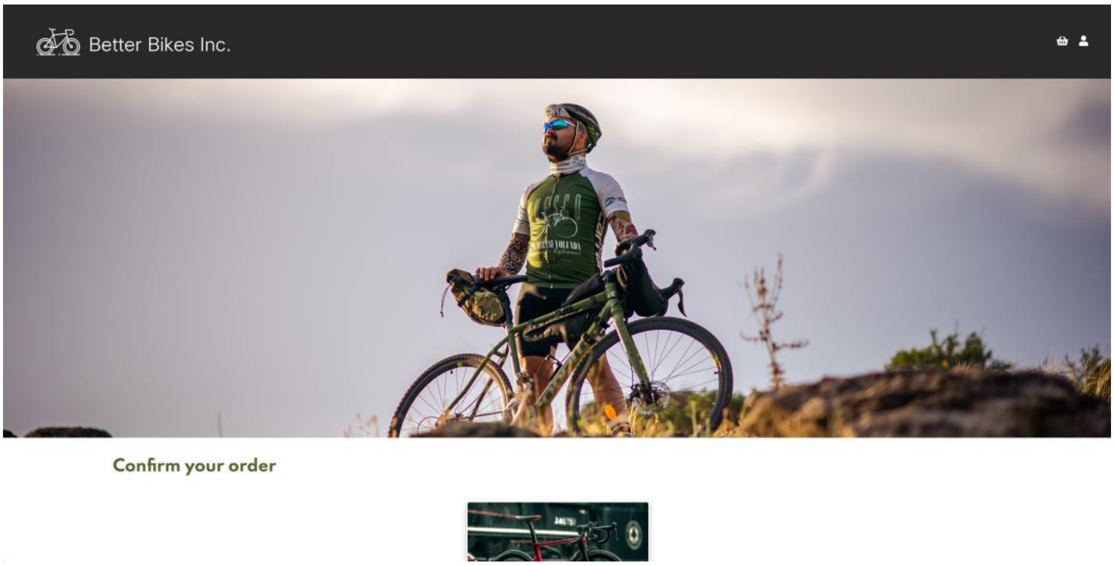


To finish the preparation, we need to update one variable.
Open the script.js file and change the value of xemail in line 5 to the email address of your PodXX_User.



Finally, click in the bottom menu on PREVIEW and Preview in a new window.





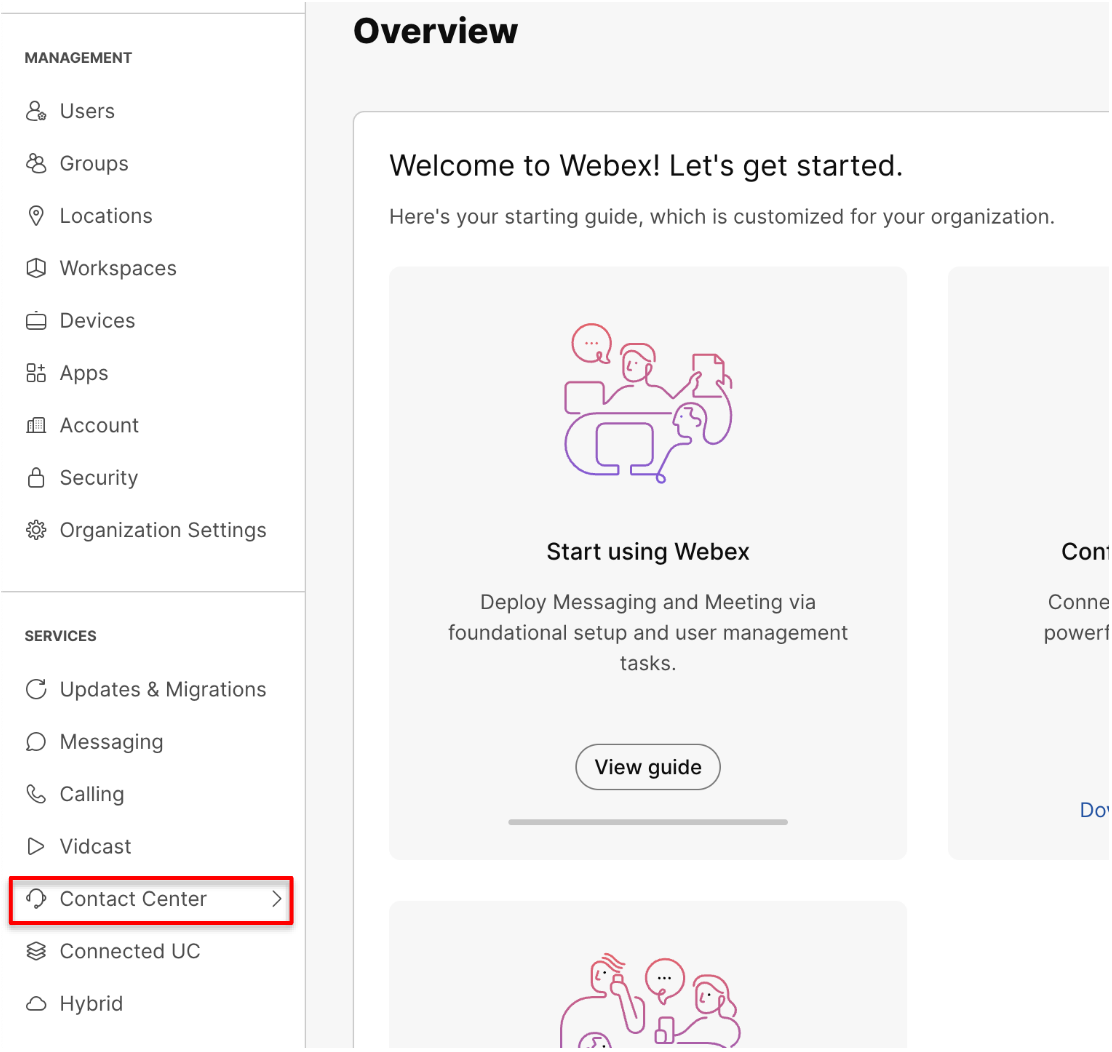
Congratulations!

You have created your website and finished Module 0 – Preparations.
Please keep the website open as we need it later in Module 1.

Module 1 – Voice Flow Creation

The configuration of this module is done through Webex Control Hub. Open your web browsers and type https://admin.webex.com in the address bar. To log in to Control Hub, please use your PodXX_User account and password.

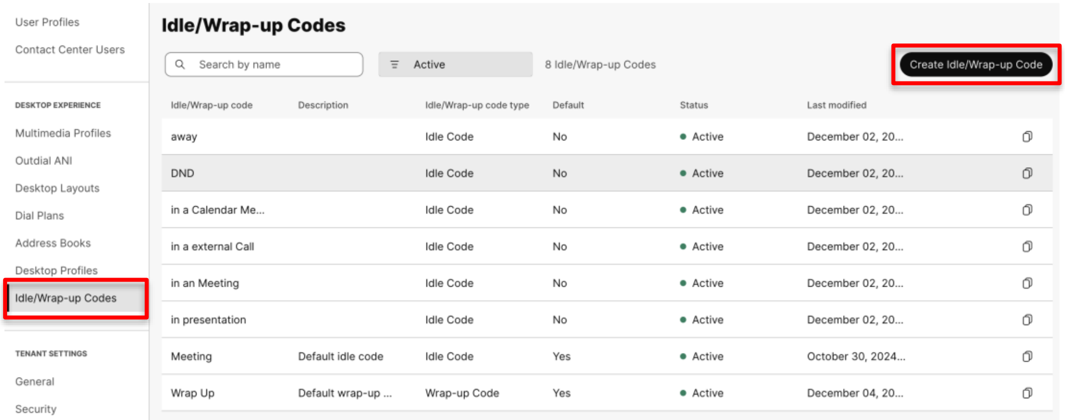
Click on Contact Center in Services on the left side to access the contact center configuration menu.



1.1 Wrap Up Codes

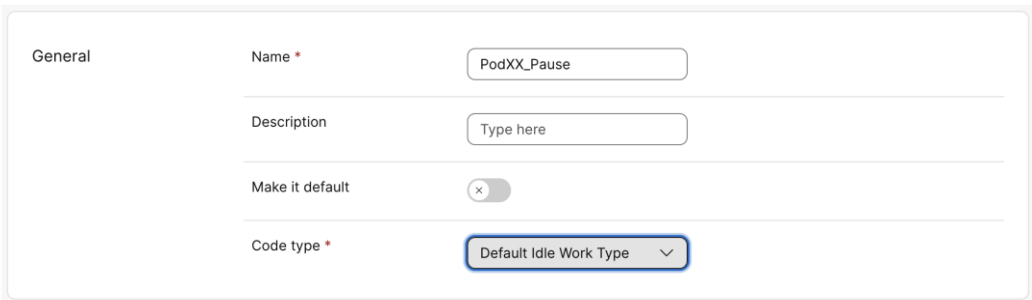
Idle or wrap up codes are two types of Auxiliary codes. Agents select Idle or Wrap-Up codes in Webex Contact Center Agent Desktop to indicate their unavailability or status of the customer contacts. Idle codes typically indicate why an agent is not available to take customer contacts, such as during a lunch break or meeting. Wrap-up codes indicate the result of the customer contacts, for example, the agent escalated the contact or sold a service.

To create new codes, please go to Idle/Wrap-Up Codes on the left side in the sidebar.



Use the Create Idle/Wrap-Up Code button to create the following new codes:

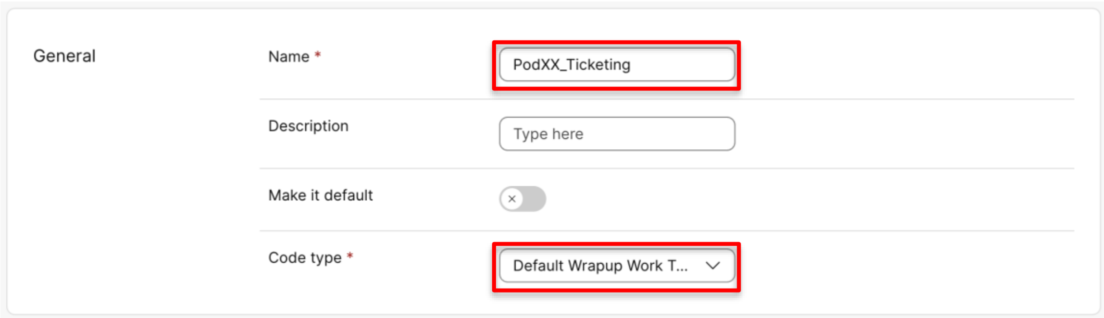
Name: PodXX_Pause
Code Type: Default Idle Work Type



Click Create.

Go back to Idle-Code overview and create another code.

Name: PodXX_Ticketing
Code Type: Default Wrapup Work Type

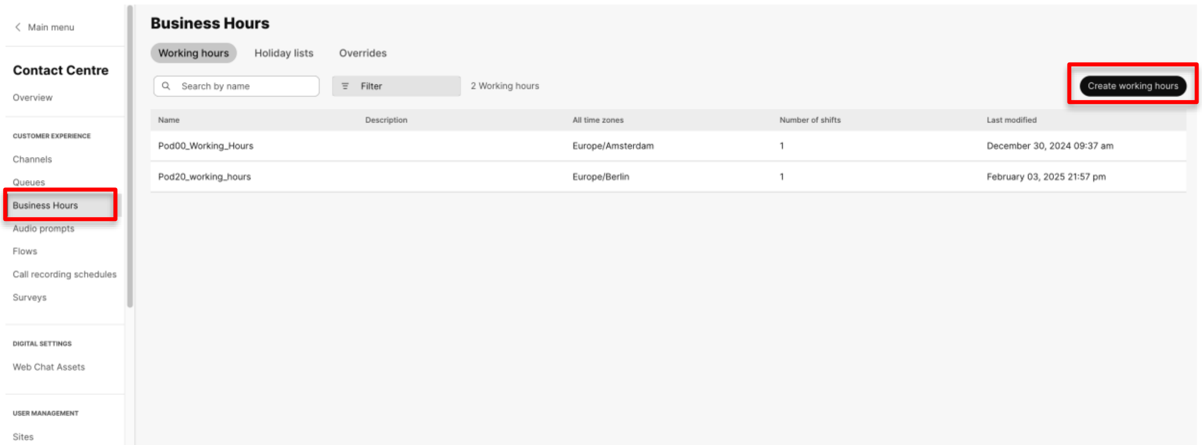


Click Create.

1.2 Business Hours

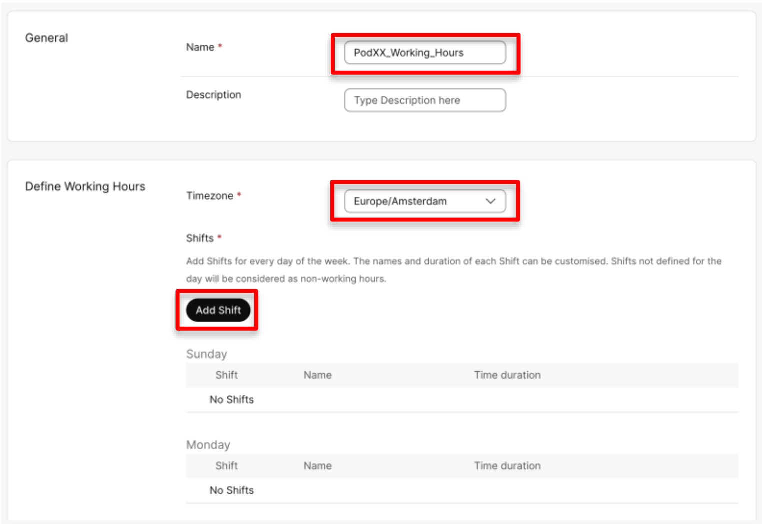
The Working Hours and Holidays created here can be used directly in the Call Flow Designer. Please configure the working hours for your Pod.

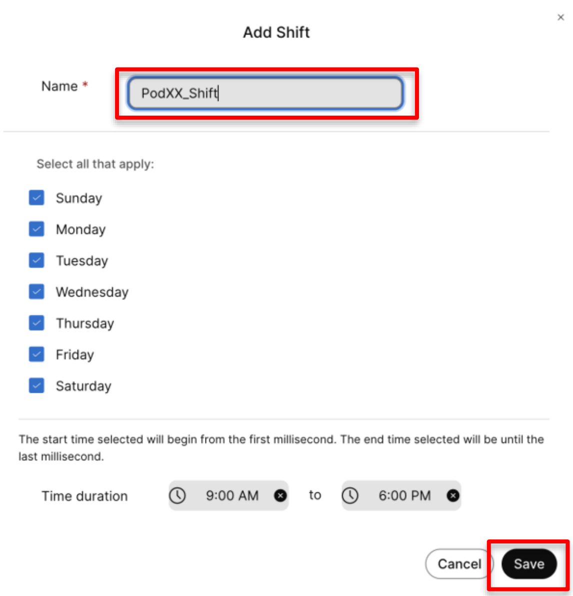
Navigate to Business Hours in the sidebar on the left and click Create Working Hours



Name: PodXX_Working_Hours
Timezone: Europe/Amsterdam

Click Add Shift



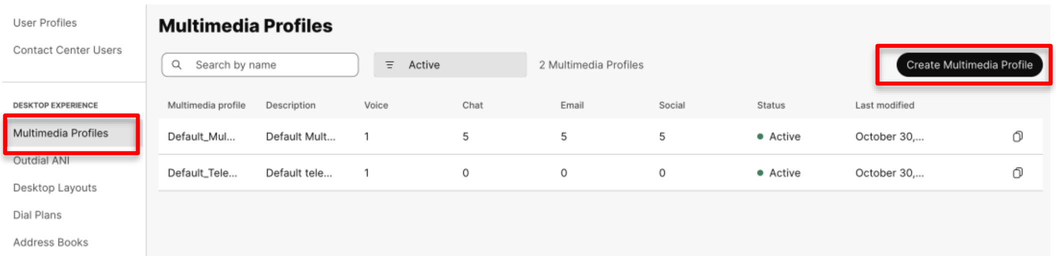


Click Save and Create.

1.3 Multimedia Profile

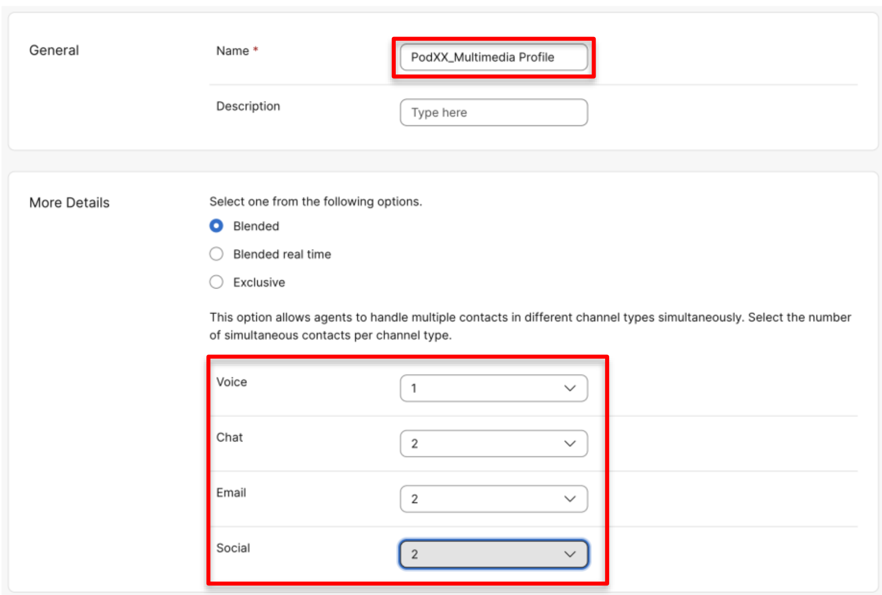
If your administrator enables multimedia for your enterprise, you can assign each agent with a multimedia profile. Each profile specifies the number of each type of media, such as email, chat, or telephony, that the agent can handle simultaneously.

Navigate to Multimedia Profile in the sidebar on the left.



Click Create Multimedia Profile.

Name: PodXX_Multimedia Profile
Channels: as of the screenshot below



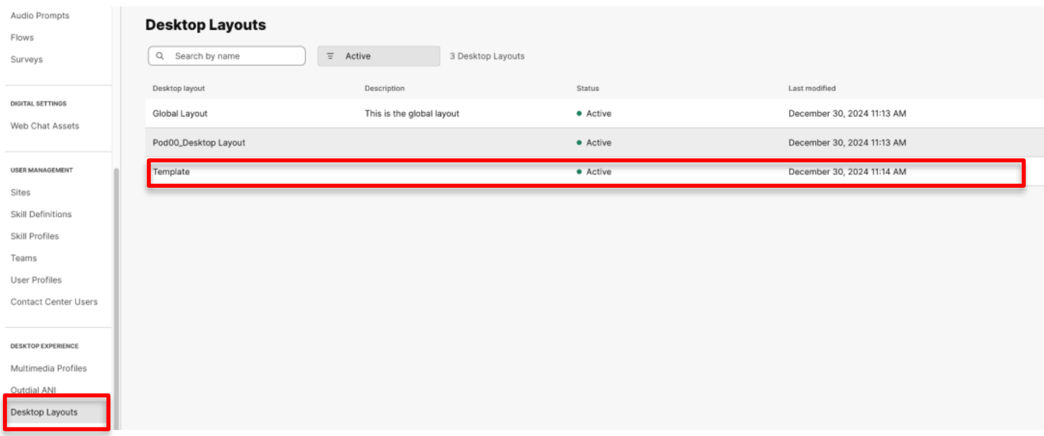
Click Create.

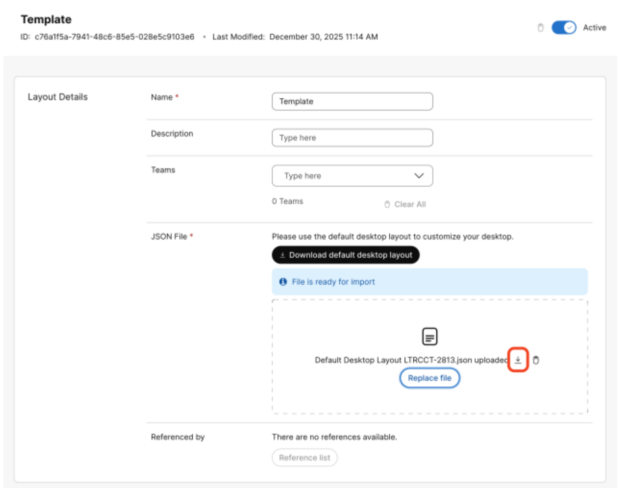
1.4 Desktop Layout

The Desktop Layout feature allows you to configure the Webex Contact Center Desktop as per your business requirements. You can customize elements such as logo, title, and widgets. You can create a desktop layout and assign it to a team. This layout generates the agent experience on the desktop for all agents who sign in as part of that team.

Navigate to Desktop Layout in the sidebar on the left.

First, we need the Template for our Lab. Open the Desktop Layout Template and download the template.

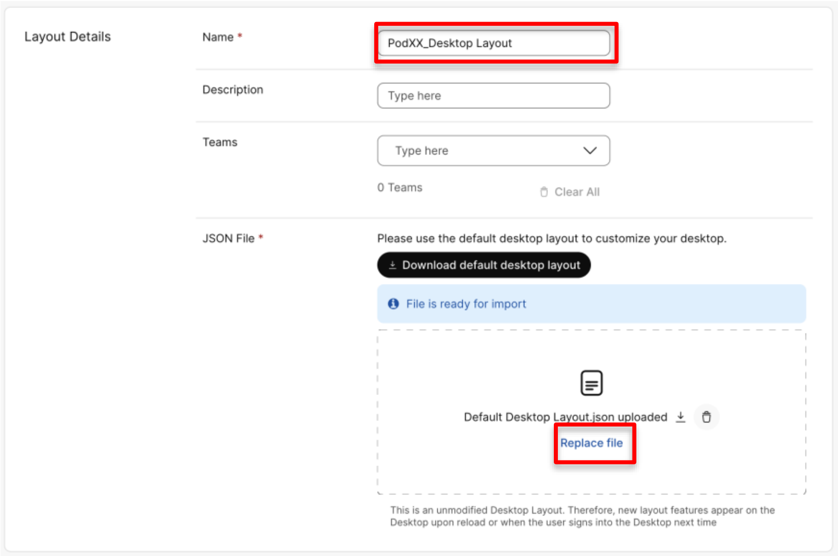




Go back to Desktop Layout List.

Click Create Desktop Layout.

Name: PodXX_Desktop Layout
JSON File: replace the template with the file you have downloaded



Click Create.

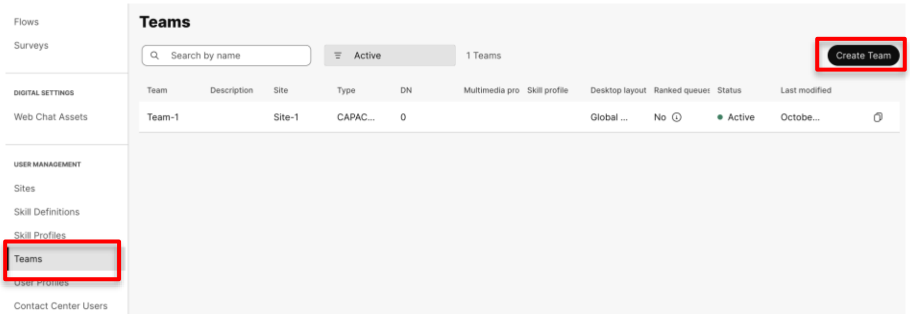
Amsterdam | February 9-14, 2025

GO BEYOND

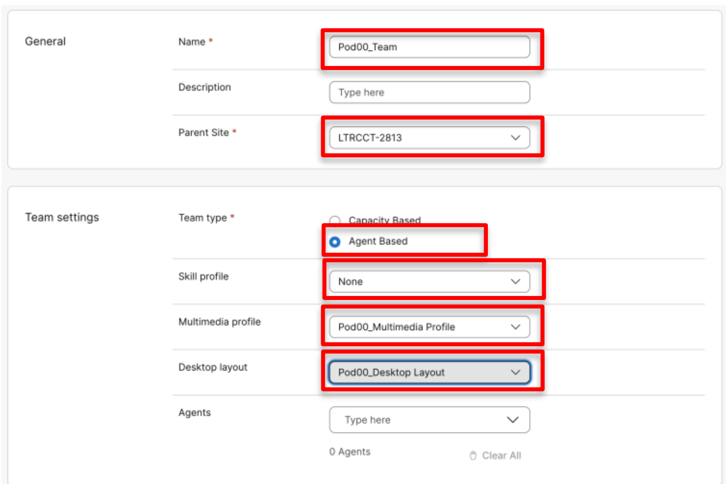
1.5 Teams

A team is a group of people who support a specific group of functions. For example, supporting the “Gold” customers or managing billing, and so on. A team consists of agents and is associated with a specific site.

Navigate to Teams in the sidebar on the left and click Create Team.



Name: PodXX_Team
Parent Site: LTRCCT-2813
Team Type: Agent based
Skill Profile: None
Multimedia Profile: PodXX_Multimedia Profile
Desktop Layout: PodXX_Desktop Layout

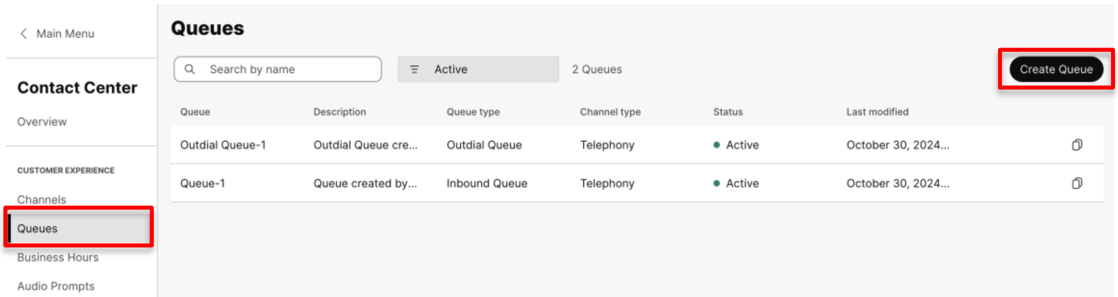


Click Create and Done.

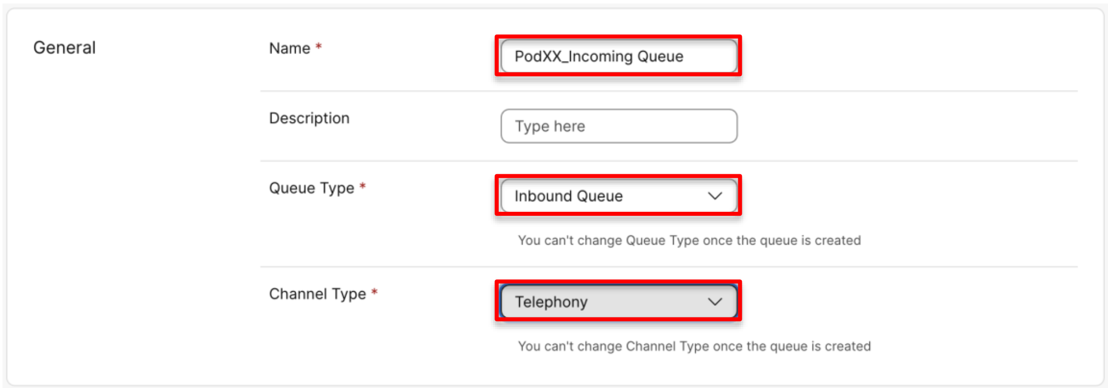
1.6 Queues

An inbound queue is where the customer contact waits before the system assigns the customer to an agent or DN.
The inbound queue that represents a third-party Automatic Call Distributor (ACD) is known as a proxy queue. Calls that are sent to an external ACD are distributed to agents by the external ACD. Webex Contact Center uses proxy queues to track the call activity that occurs on the external ACD queues.

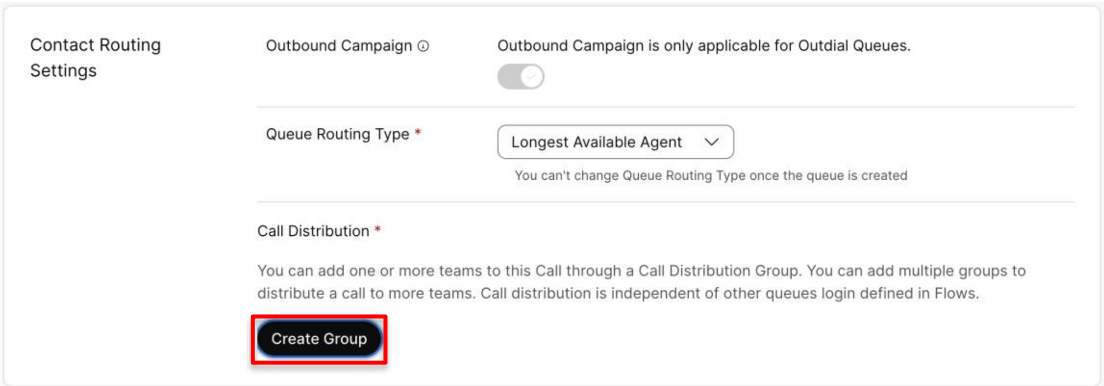
Navigate to Queues in the sidebar on the left and click Create Queue.



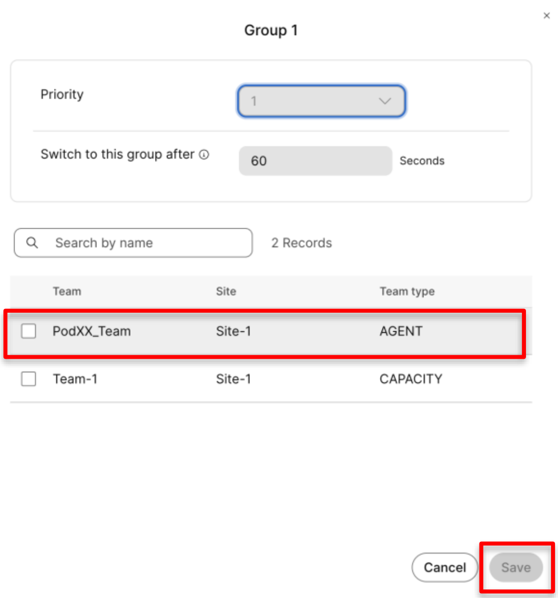
Name: PodXX_Incoming Queue
Queue Type: Inbound Queue
Channel Type: Telephony



Click Create Group.



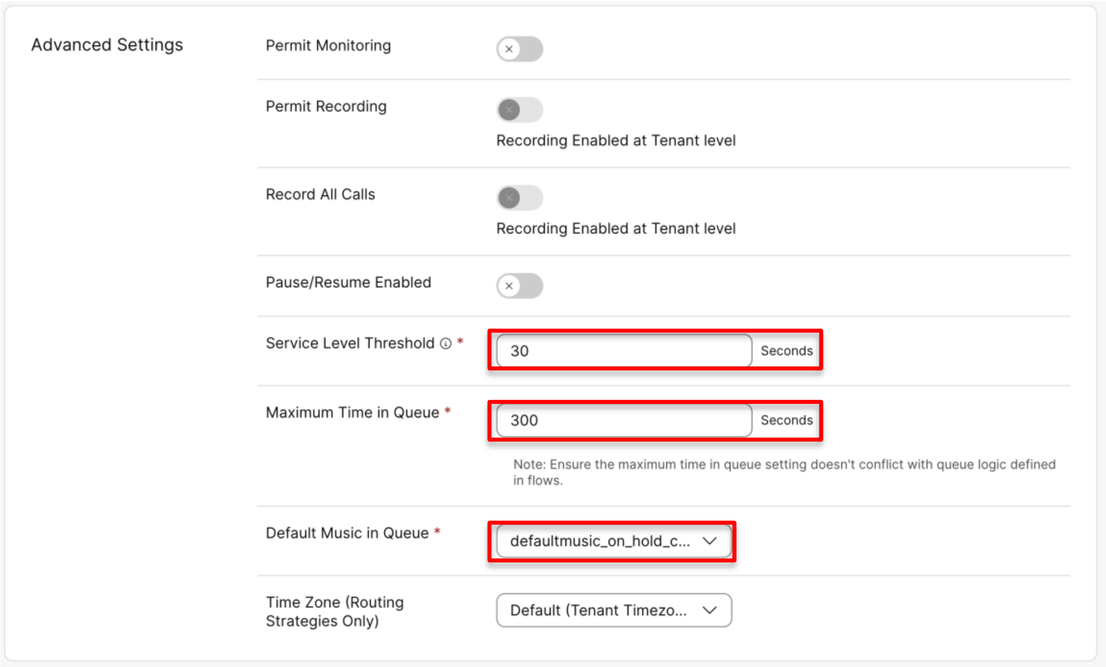
Select your Team and Save.



Under Advanced Settings configure the following:

Service Level Threshold: 30 seconds
Maximum Time in Queue: 300 seconds
Default Music in Queue: first entry

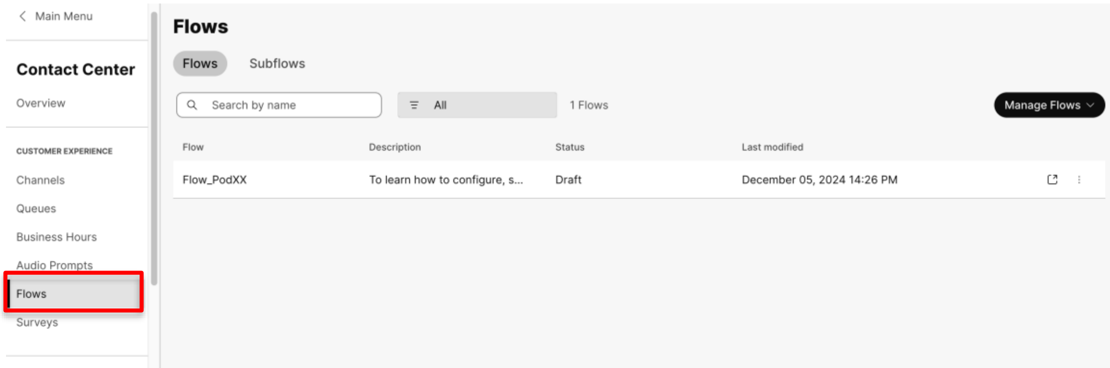
Then click Create.



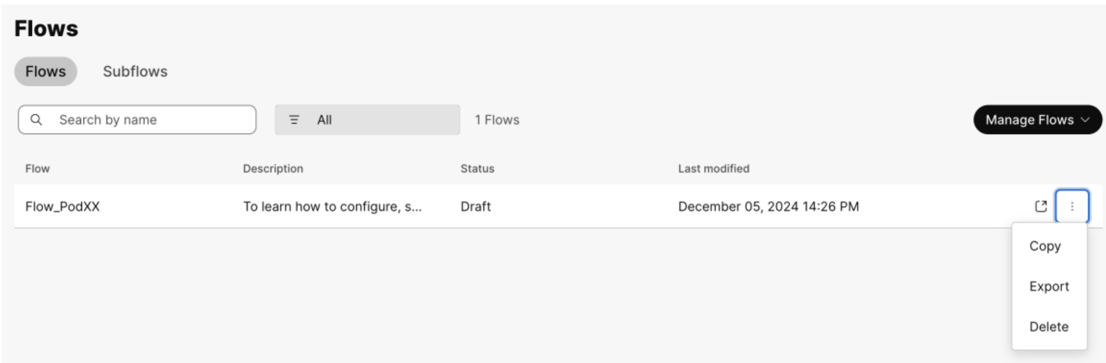
Click Create.

1.7 Flows

Navigate to Flows in the sidebar on the left.



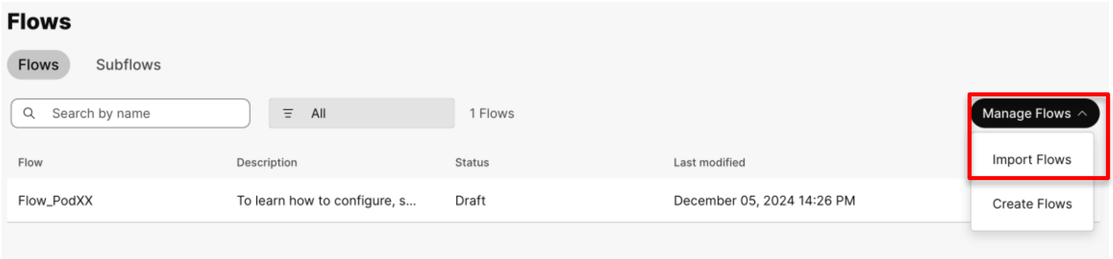
To get a flow template export the Flow Flow_PodXX through the 3 dots menu.



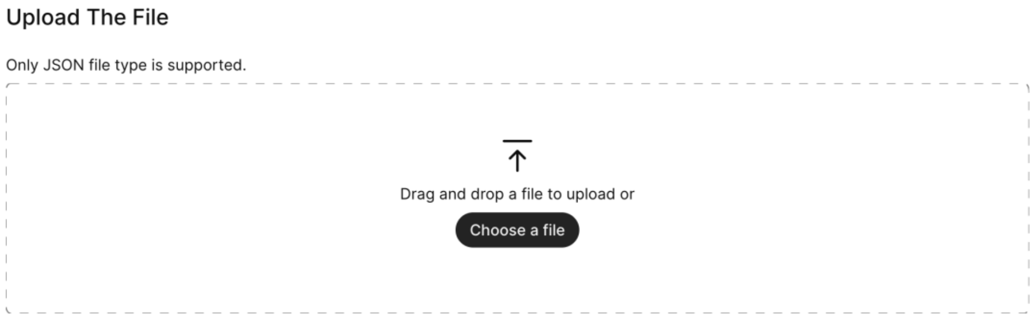
Next, you need to rename the flow name. For that, please open the exported file in an Editor and change the Flow Name to Flow_PodXX.

Save the file.

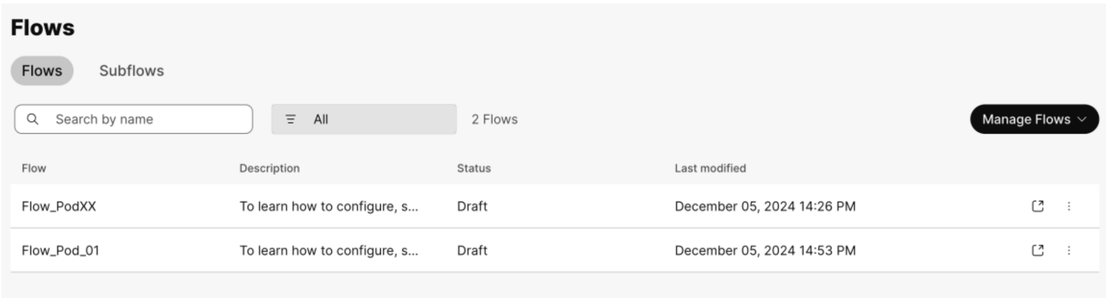
Now the new file must be uploaded. For that, import the new Flow File through Manage Flows and Import Flows.



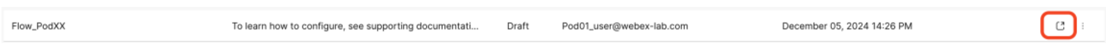
Choose your File and click Import.



Now your Flow should be listed. If not, please contact us.

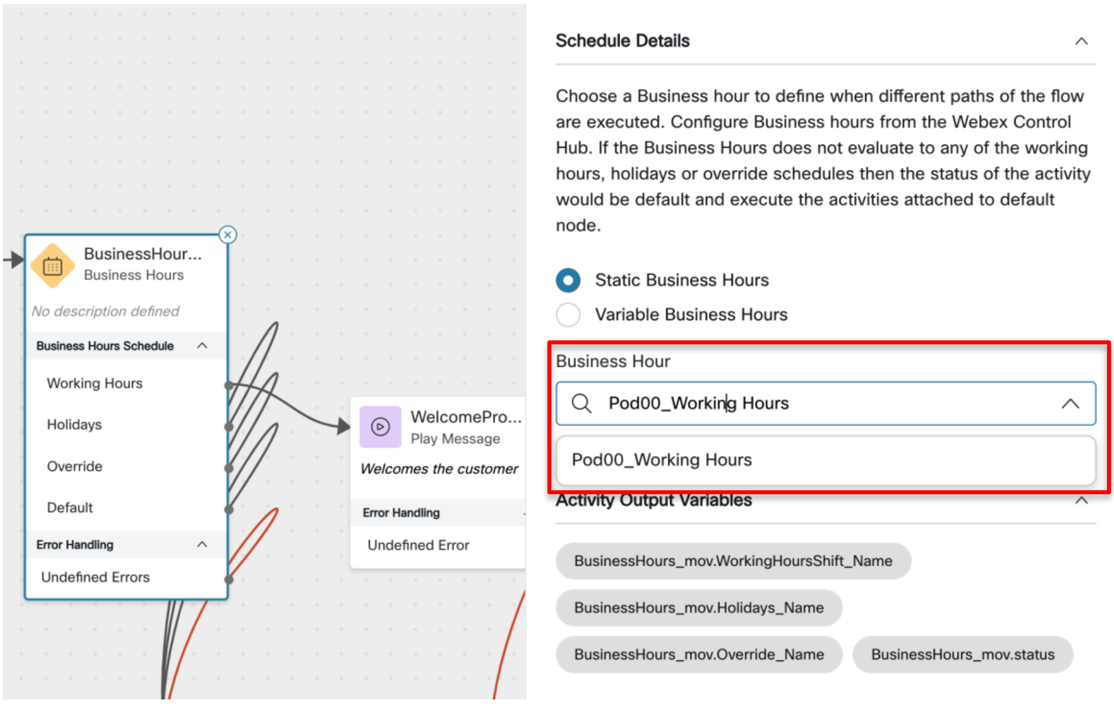


Open the new flow you've just added with the highlighted Button



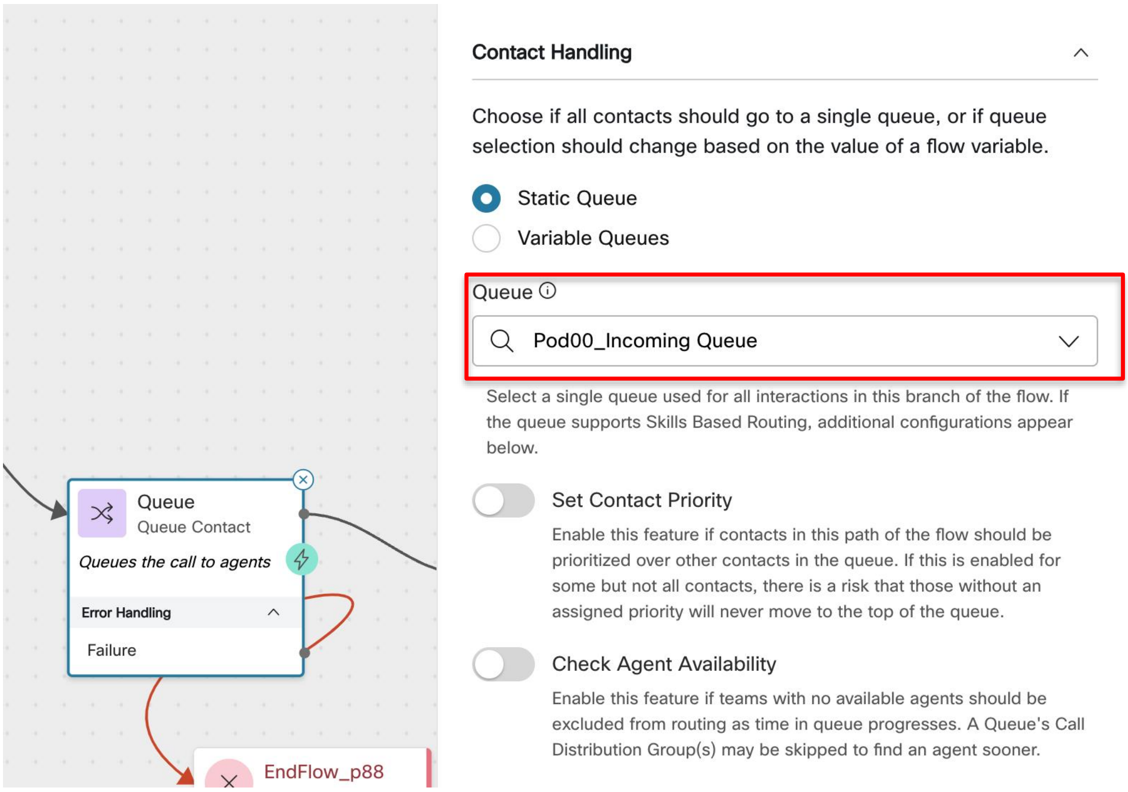
Toggle the Edit mode on.

Click on Business Hours and choose your Pod Working Hours.

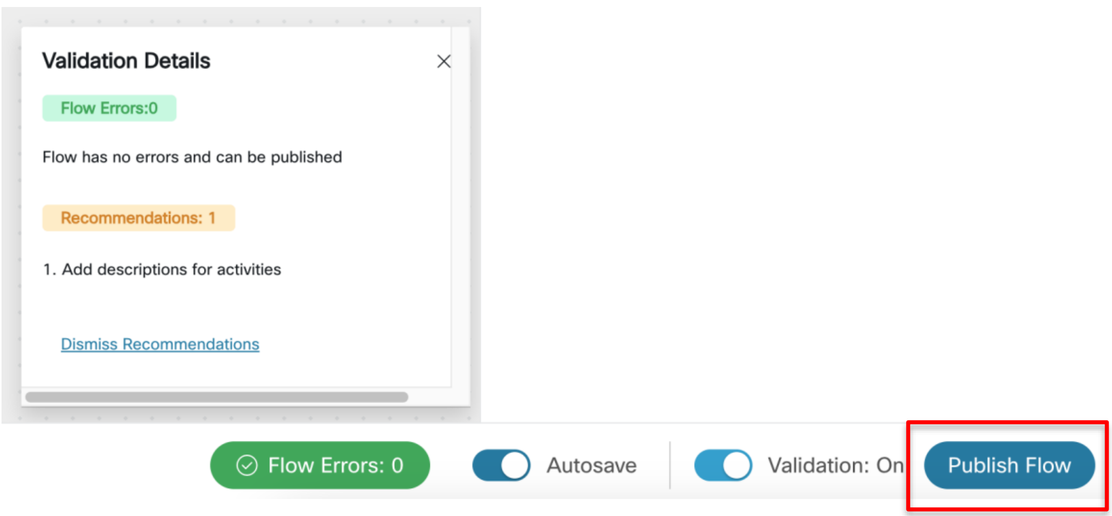


Click on the grey background and zoom out.

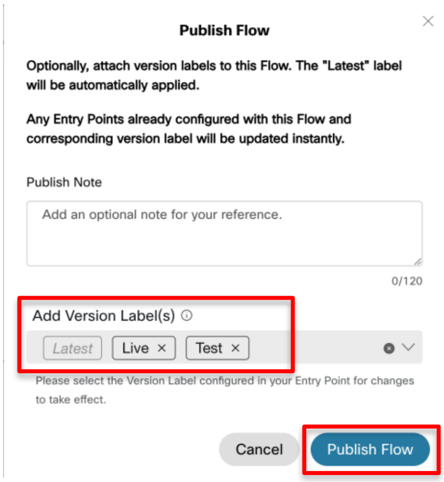
Click on Queue and select your Pod Queue.



Enable the Validation – when it's green click on Publish Flow.



Open the Version Label Drop Down, select Live and Test and then Publish the Flow.

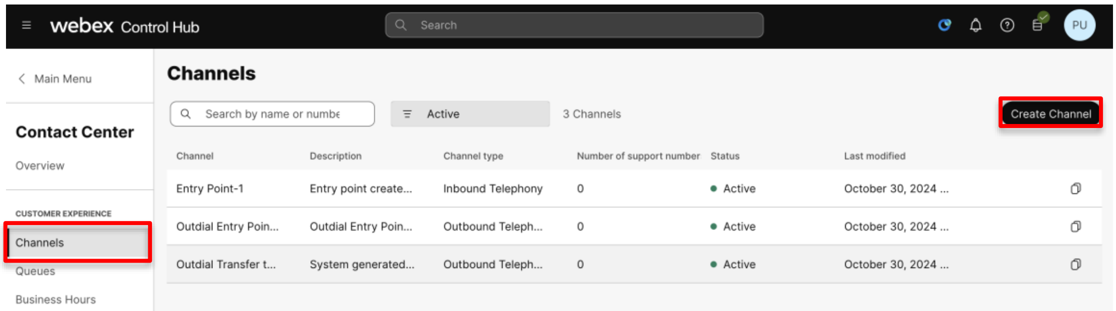


After that you can close the browser tab.

1.8 Channels (Entry Point)

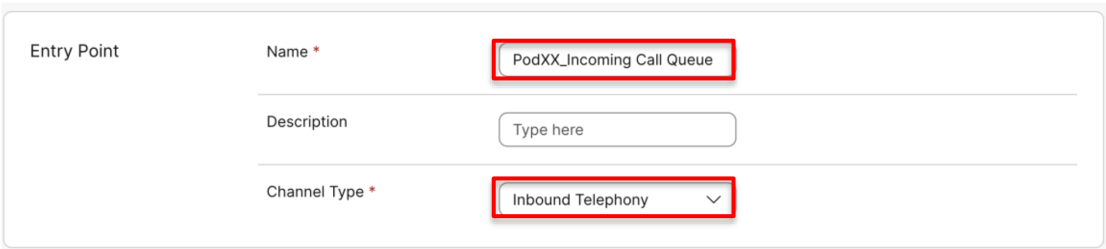
The inbound entry point is the initial landing place for a customer contact in the Webex Contact Center system. For the customer calls, you can associate one or more toll-free or dial numbers with a given entry point. The system performs IVR call treatment for a call while the call is in the entry point.
You can assign a flow to an entry point. Moreover, if you want to implement the business hours to define the working and nonworking hours for your contact center, you can choose a flow that has the business hours activity configured and assign it to the entry point.

Navigate to Channels in the sidebar on the left and click Create Channel.



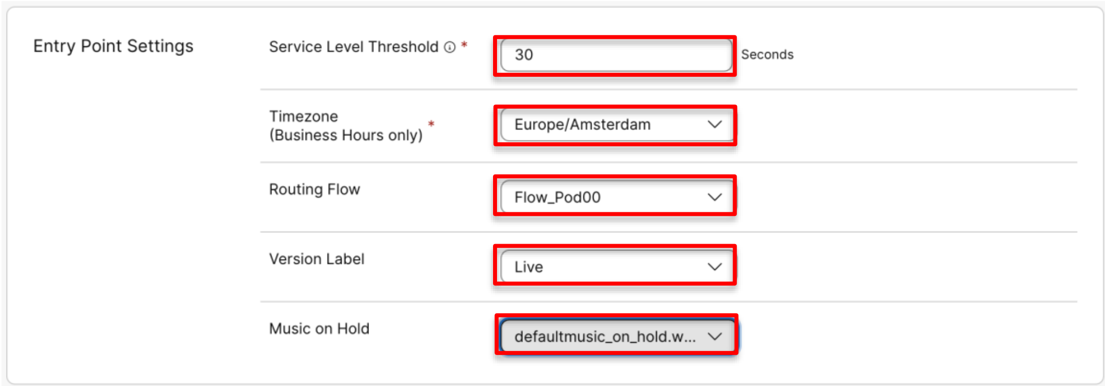
In the first step we are creating a channel for inbound calls and associate this to our flow and our service queue Number.

Name:	PodXX_Incoming Call Queue
Channel Type:	Inbound Telephony



Configure the following Entry Point Settings:

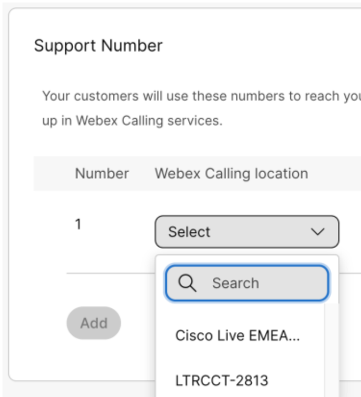
Service Level Threshold:	30 Seconds
Timezone:	Europe/Amsterdam
Routing Flow:	Your pods Flow
Version Label:	Live
Music on Hold:	Choose the first entry



For your reference:
You have seen that you can choose different Version Labels. This Feature is the Version Management on Flows. So, with that you can create a dedicated Test Channel for this Flow. This is so you can test all new configurations, and then when the tests are successfully completed, you can publish the flow, making it Live.

Now add a PSTN Number to this Entry Point. Under Support Number click Add.

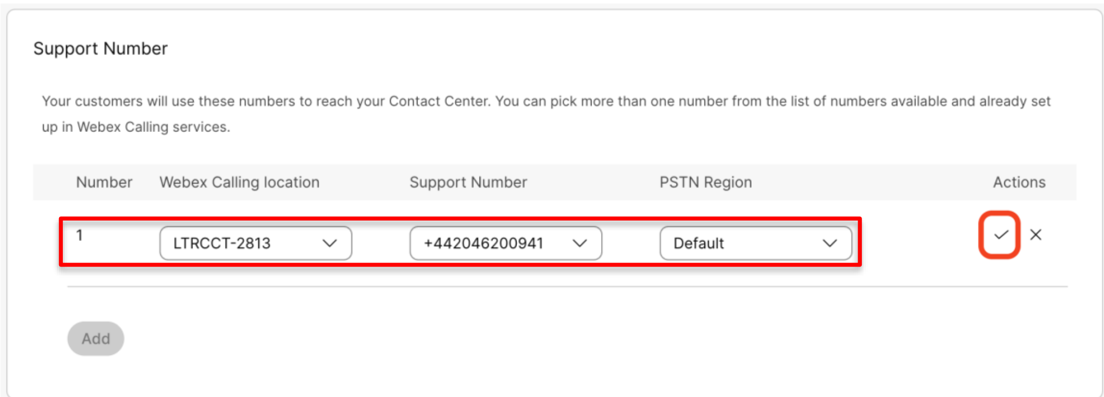
Choose LTRCCT-2813 as Webex Calling Location.



For Support Number choose the correlating Number that is assigned to your Pod!

Pod Number	PodXX Service Queue
01	+442046200496
02	+442046200506
03	+442046200517
04	+442046200524
05	+442046200527
06	+442046200530
07	+442046200533
08	+442046200538
09	+442046200629
10	+442046200941
11	+442046200947
12	+442046200951
13	+442046200954
14	+442046200956
15	+442046200974
16	+442046200977
17	+442046200980
18	+442046200982
19	+442046200989
20	+442046200994

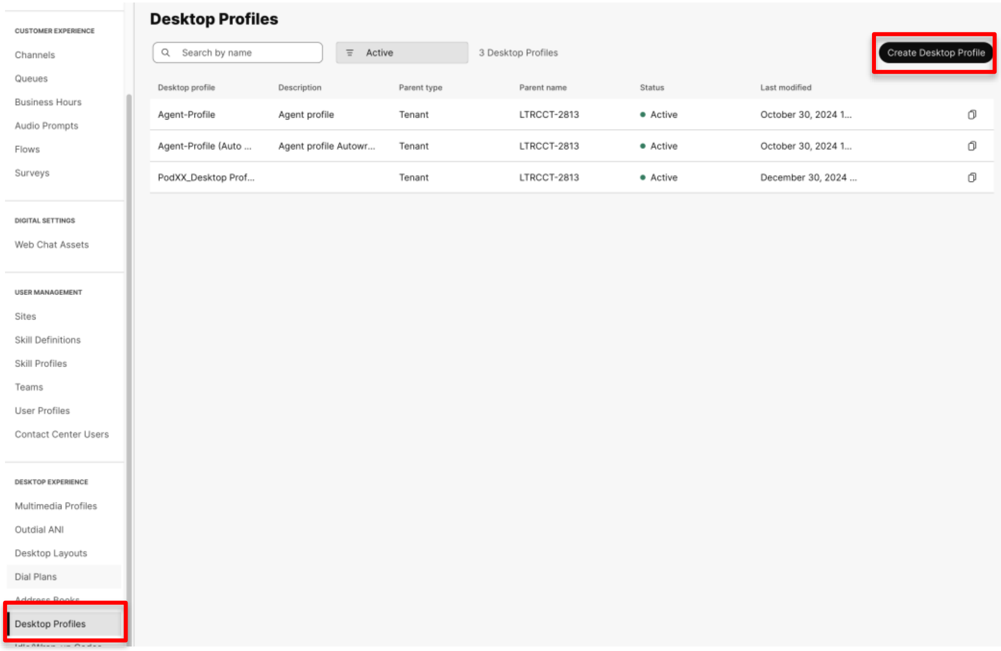
Leave the PSTN Region as Default and Accept under Actions.



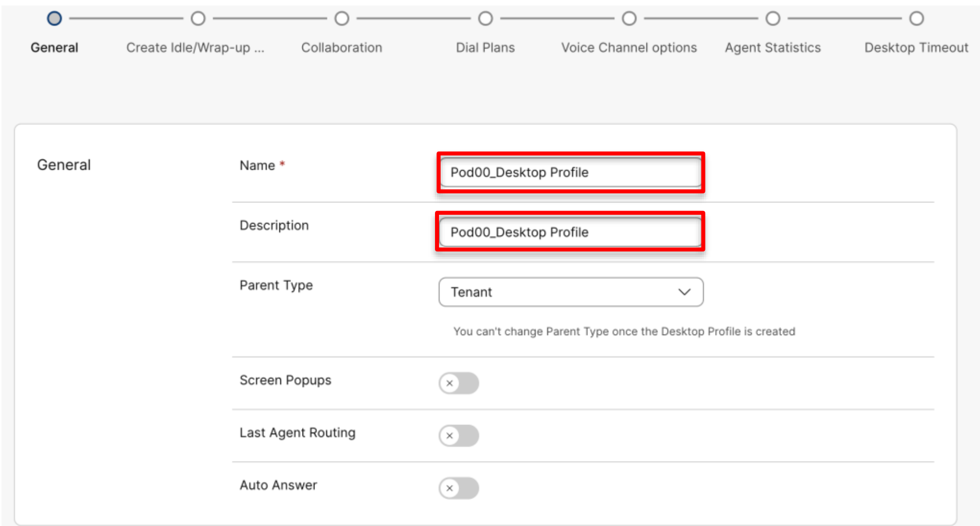
Afterwards, click on Create.

1.9 Desktop Profile

Navigate to Desktop Profiles in the sidebar on the left and click Create Desktop Profile.

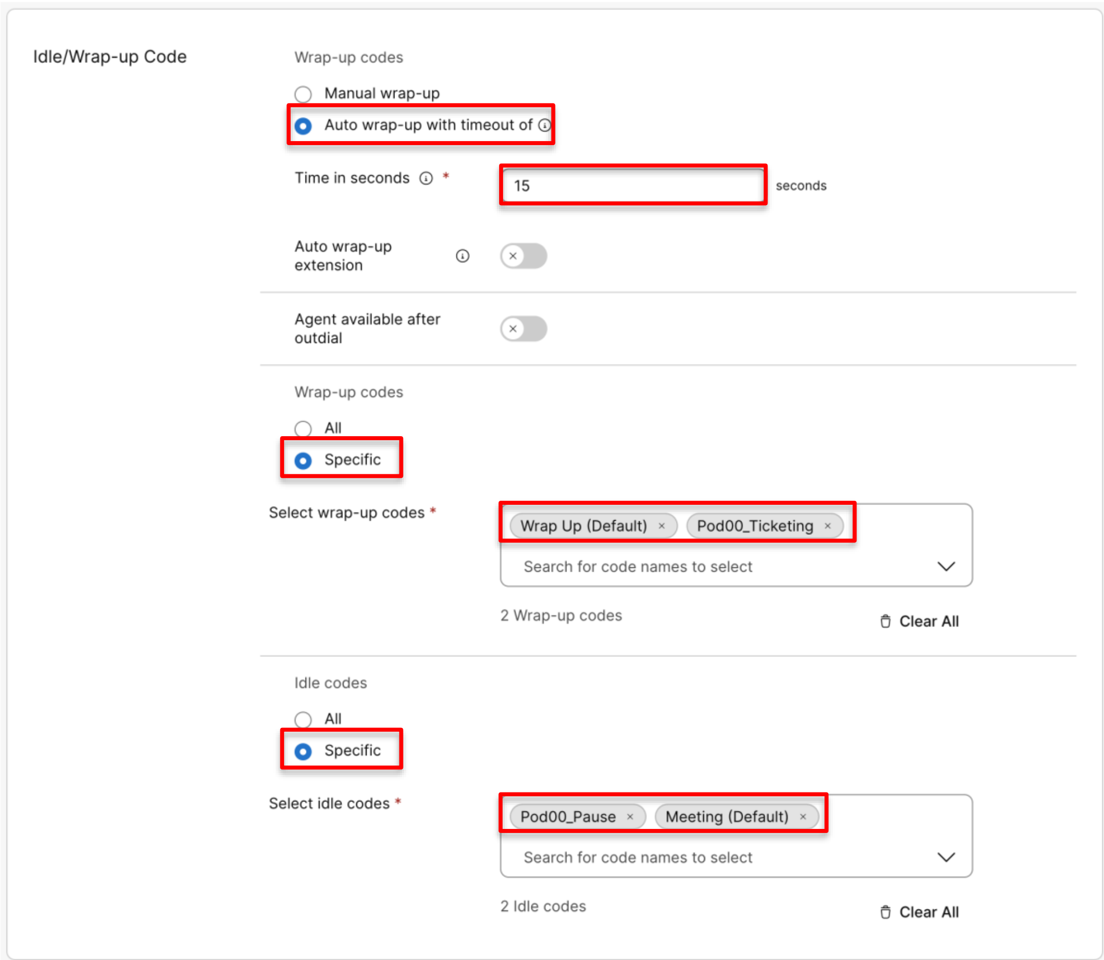


Define the Name & Description as PodXX_Desktop Profile and click Next.



Configure the following Idle/Wrap-up Code settings:

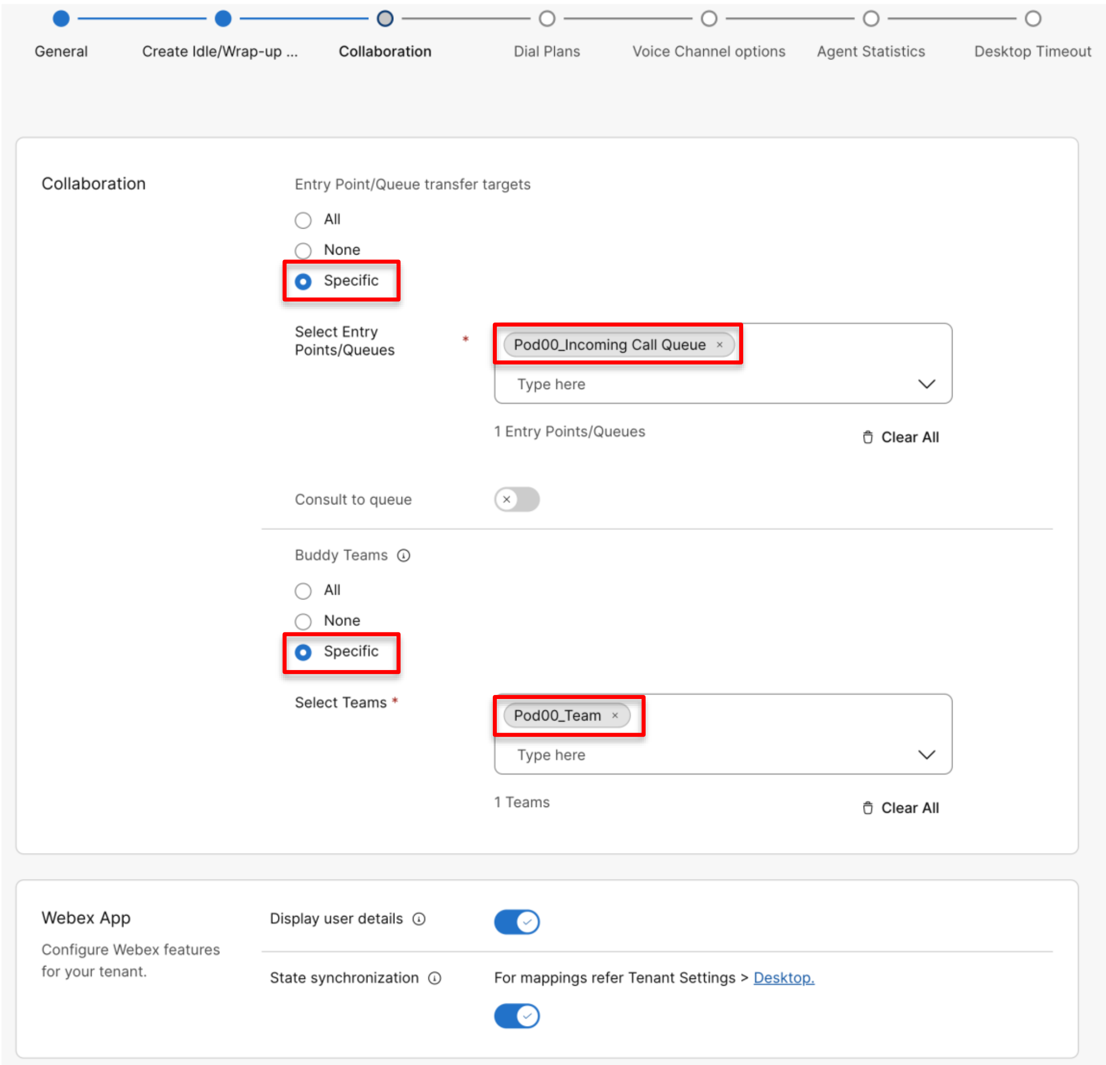
Wrap-up codes:	Auto wrap-up with timeout of
Time in seconds:	15 seconds
Wrap-up codes:	Specific
Select wrap-up codes:	Wrap Up (Default), PodXX_Ticketing
Idle Codes:	Specific
Selected Idle codes:	PodXX_Pause, Meeting (Default)



Click on Next.

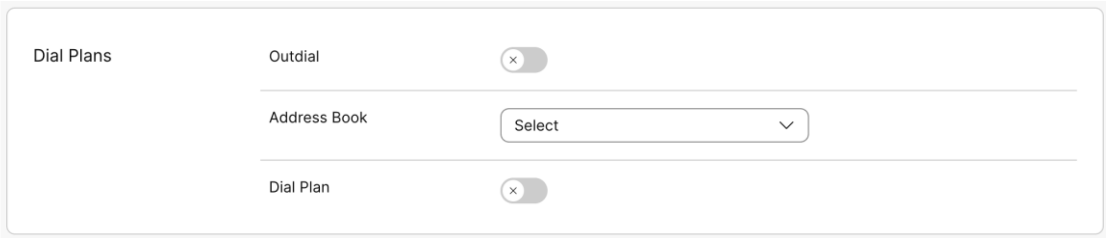
Configure the following Collaboration settings:

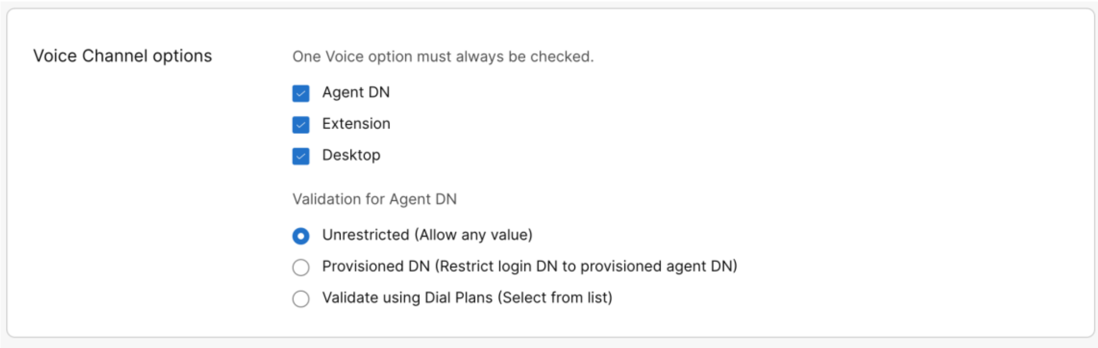
Entry Point/Queue transfer targets: Specific
Select Entry Points/Queues: PodXX_Incoming Call Queue
Buddy Teams: Specific
Select Teams: PodXX_Team



Click on Next.

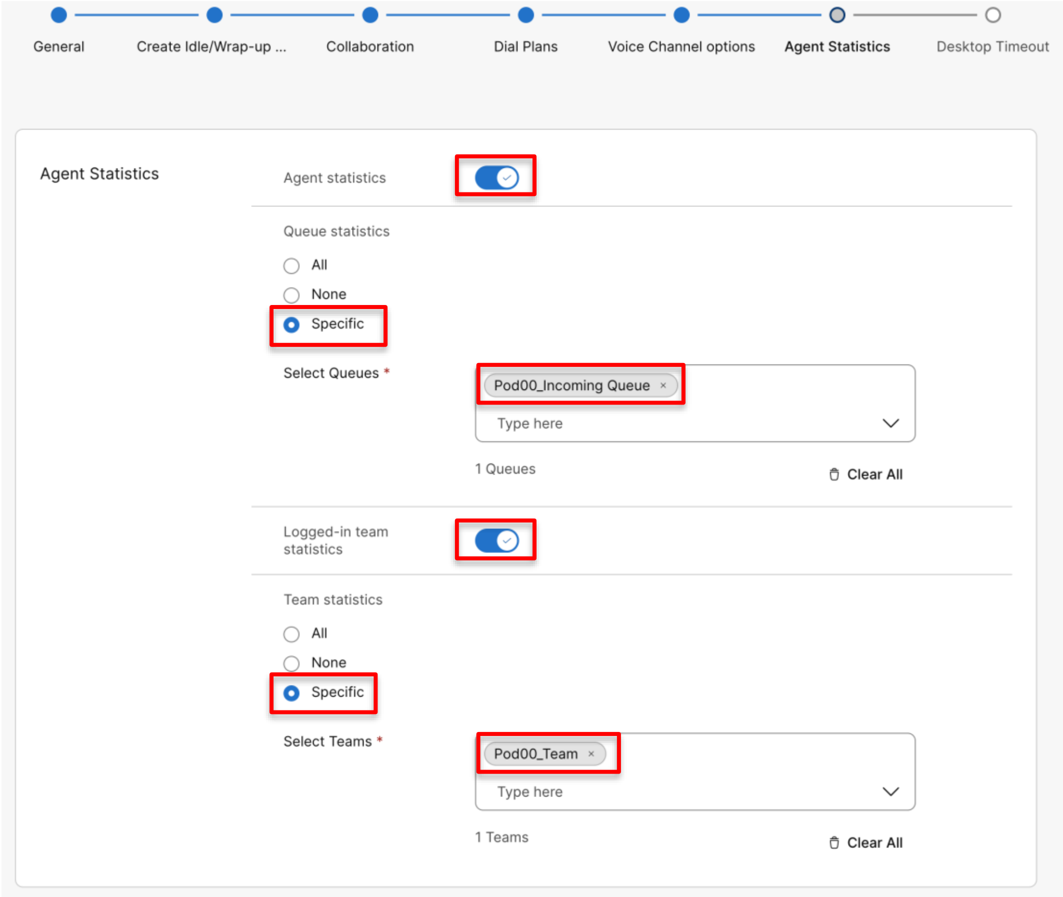
Leave the Dial plans and Voice channel options configuration as it is by clicking Next.





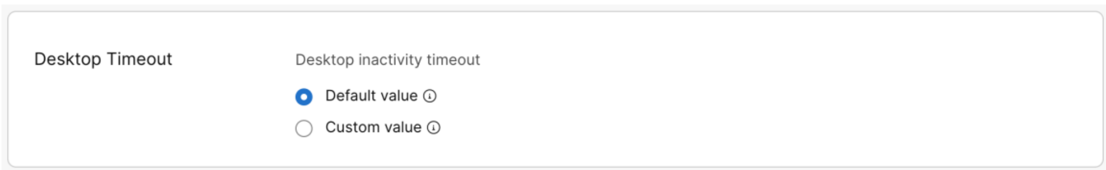
Configure the following Agent statistics settings:

Agent statistics:	Enabled
Queue statistics:	Specific
Selected Queues:	PodXX_Incoming Queue
Logged-in team statistics:	Enabled
Team statistics:	Specific
Selected Teams:	PodXX_Team



Click Next.

Leave the Desktop timeout on Default value.

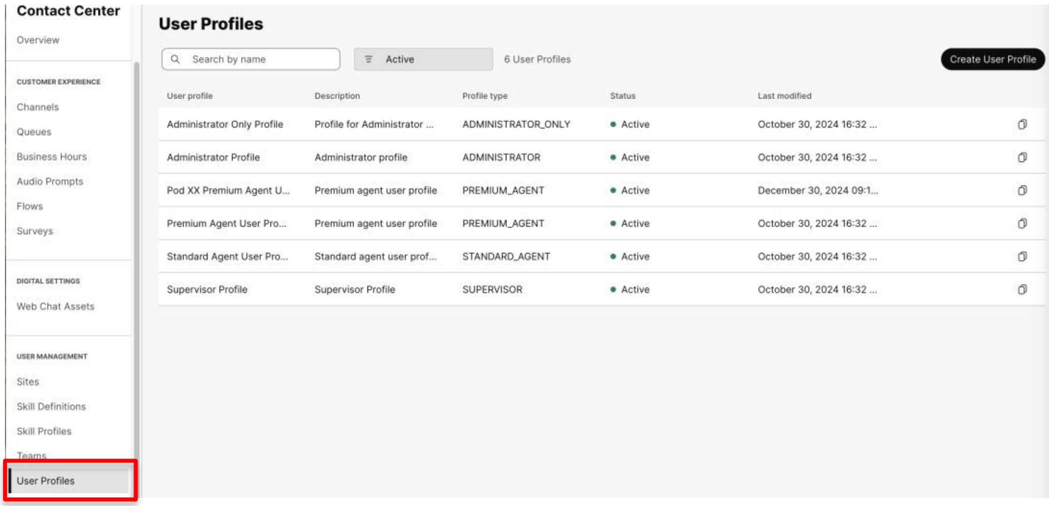


Click Create.

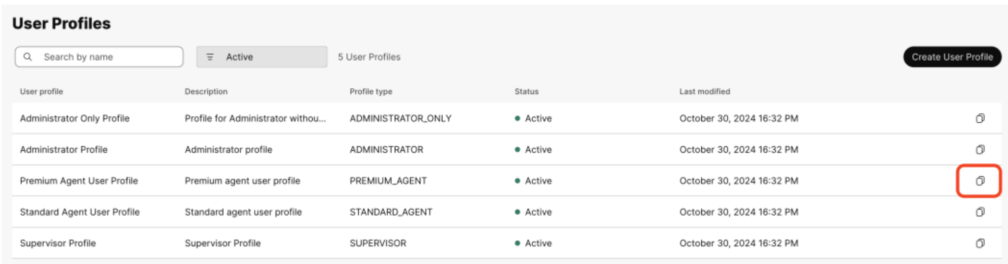
1.10 User Profile

User profiles determine the features accessible to a user in Cisco Contact Center.

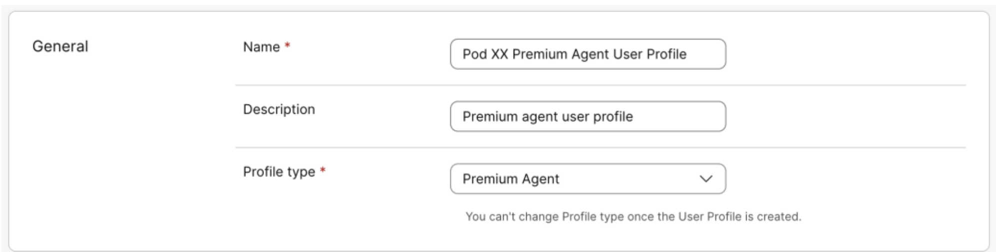
Navigate to User Profiles in the sidebar on the left.



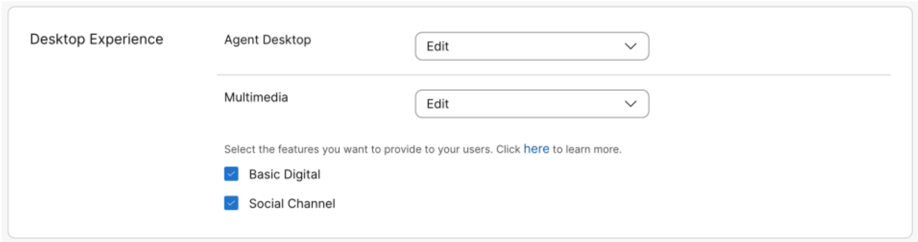
Now click on Copy from Premium Agent User Profile



Rename the Name to your Pod IDs number and remove the _copy in the end.

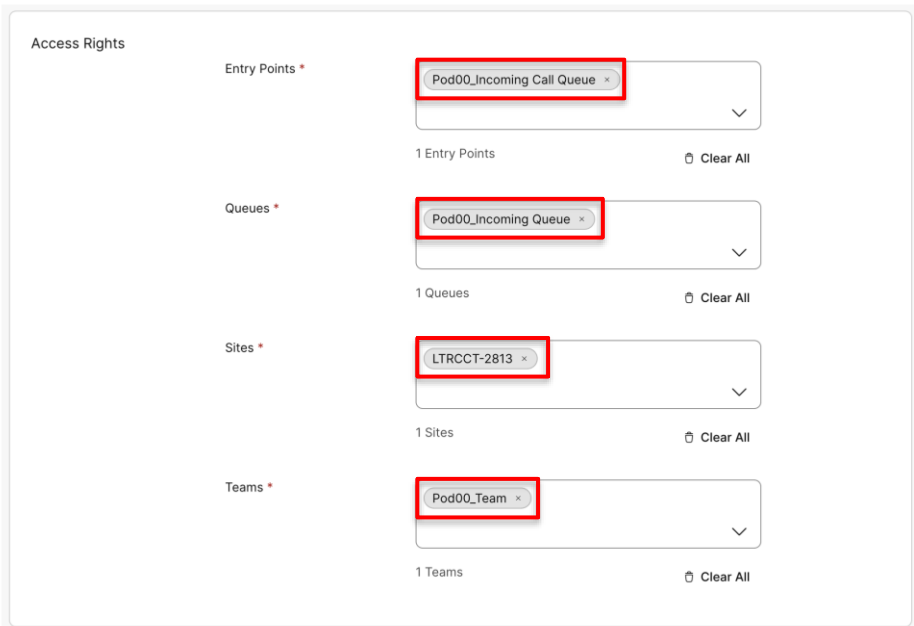


Click on Next.
Leave the Desktop experience as it is by clicking on Next.



Configure the following Access Rights settings:

Entry points: PodXX_Incoming Call Queue
Queues: PodXX_Incoming Queue
Sites: LTRCCT-2813
Teams: PodXX_Team



Click on Create.

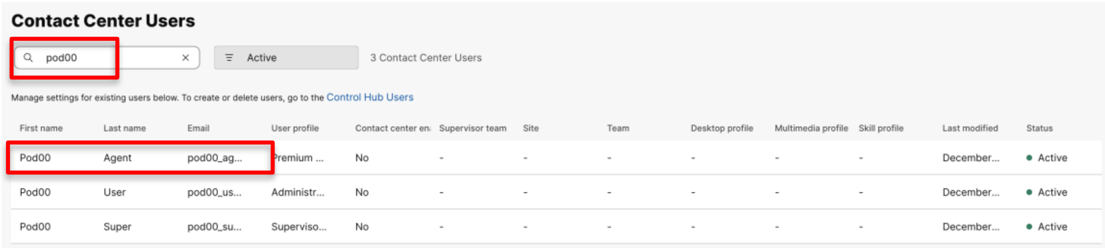
Note: All Supervisor Agents are using the same Supervisor Profile.

1.11 Agent Configuration

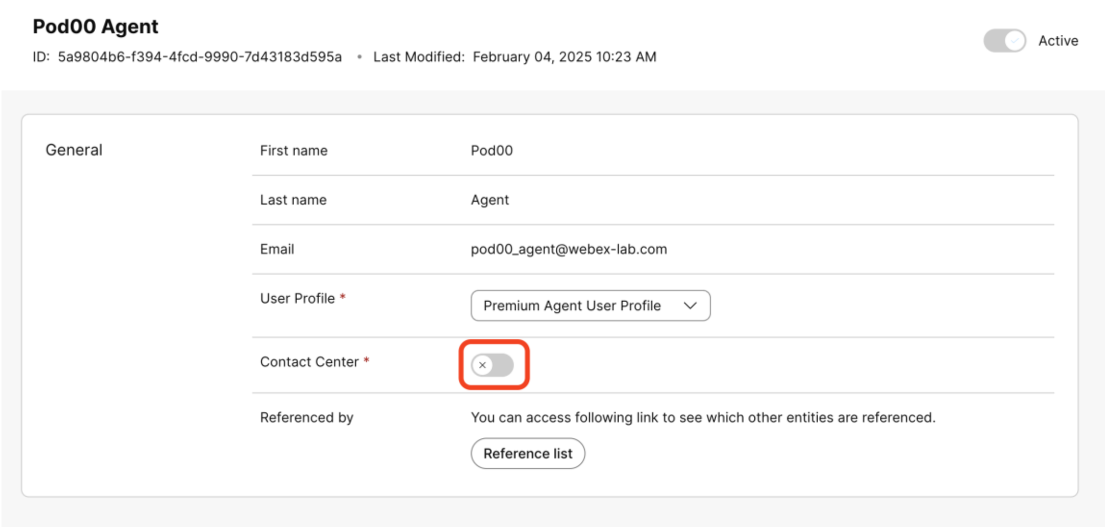
Every user who has been assigned a license on Control Hub is automatically synced to the contact center. However, this user still needs to be enabled and configured. In preparation, 3 users were created. For license assignment, please refer to the Preparations section (Module 0.1).

Navigate to Contact Center Users in the sidebar on the left and search for your Pod.

Select the entry with your Pod ID in First name and Agent in the Last name.

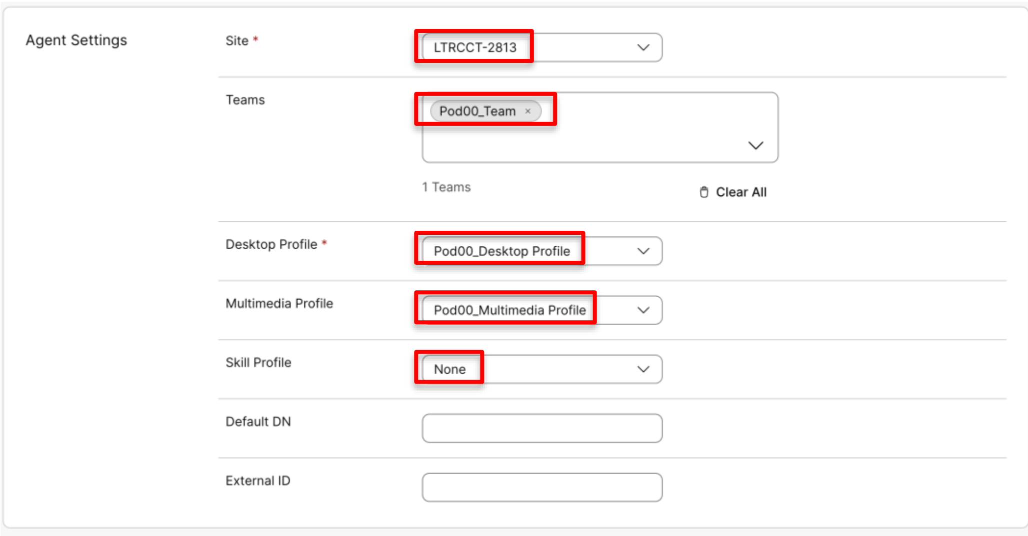


Enable the user for Contact Center.



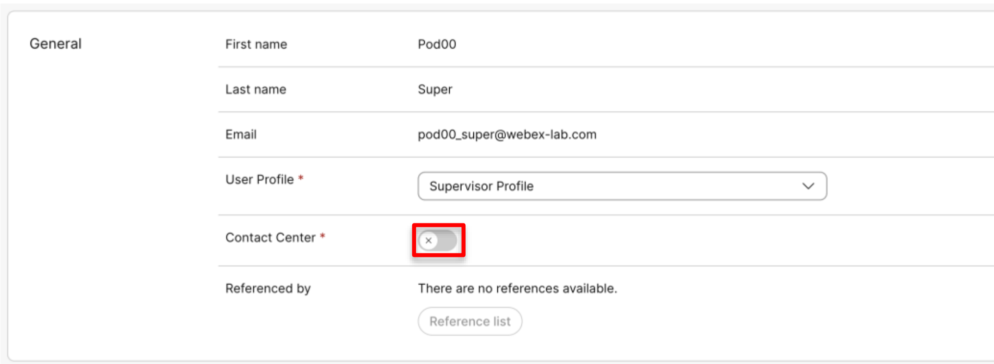
Configure the following Agent Settings:

Site:	LTRCCT-2813
Teams:	PodXX_Team
Desktop-Profile:	PodXX_Desktop Profile
Multimedia profile:	PodXX_Multimedia Profile
Skill Profile:	None



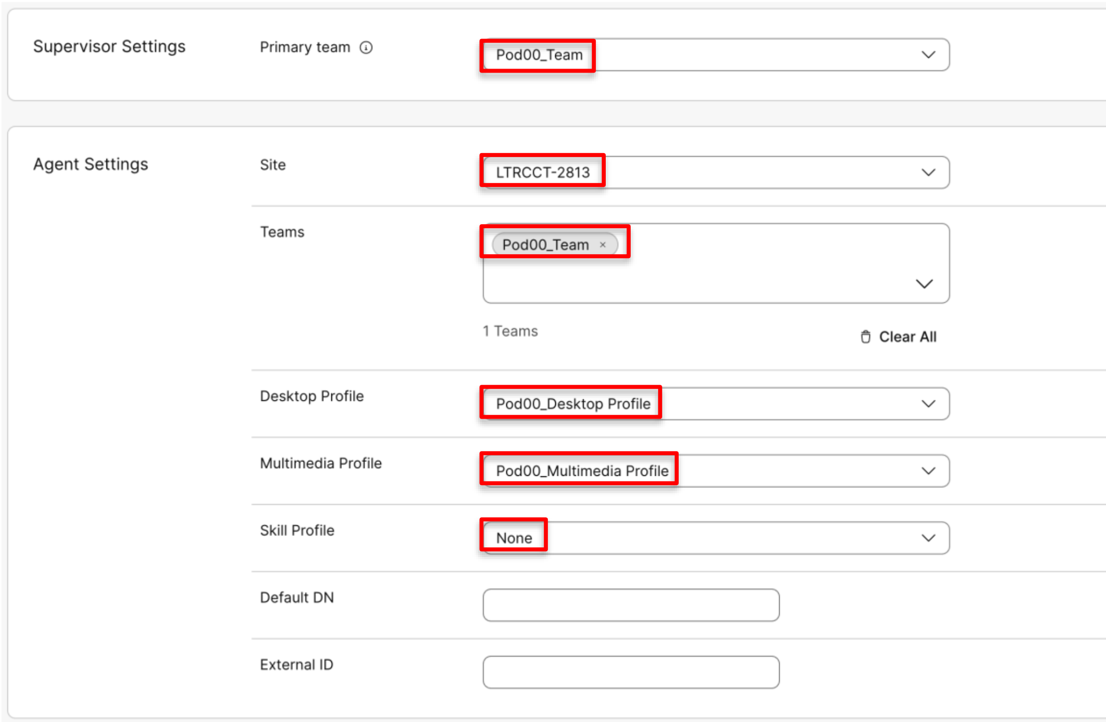
Click Save and go back to user list by clicking Contact Center Users on the top left.

Search again for your Pod and select the entry with your Pod ID in First name and Super in the Last name and enable this user for Contact Center as well.



Configure the following Supervisor and Agent Settings:

Primary team:	PodXX_Team
Site:	LTRCCT-2813
Teams:	PodXX_Team
Desktop-Profile:	PodXX_Desktop Profile
Multimedia profile:	PodXX_Multimedia Profile
Skill Profile:	None

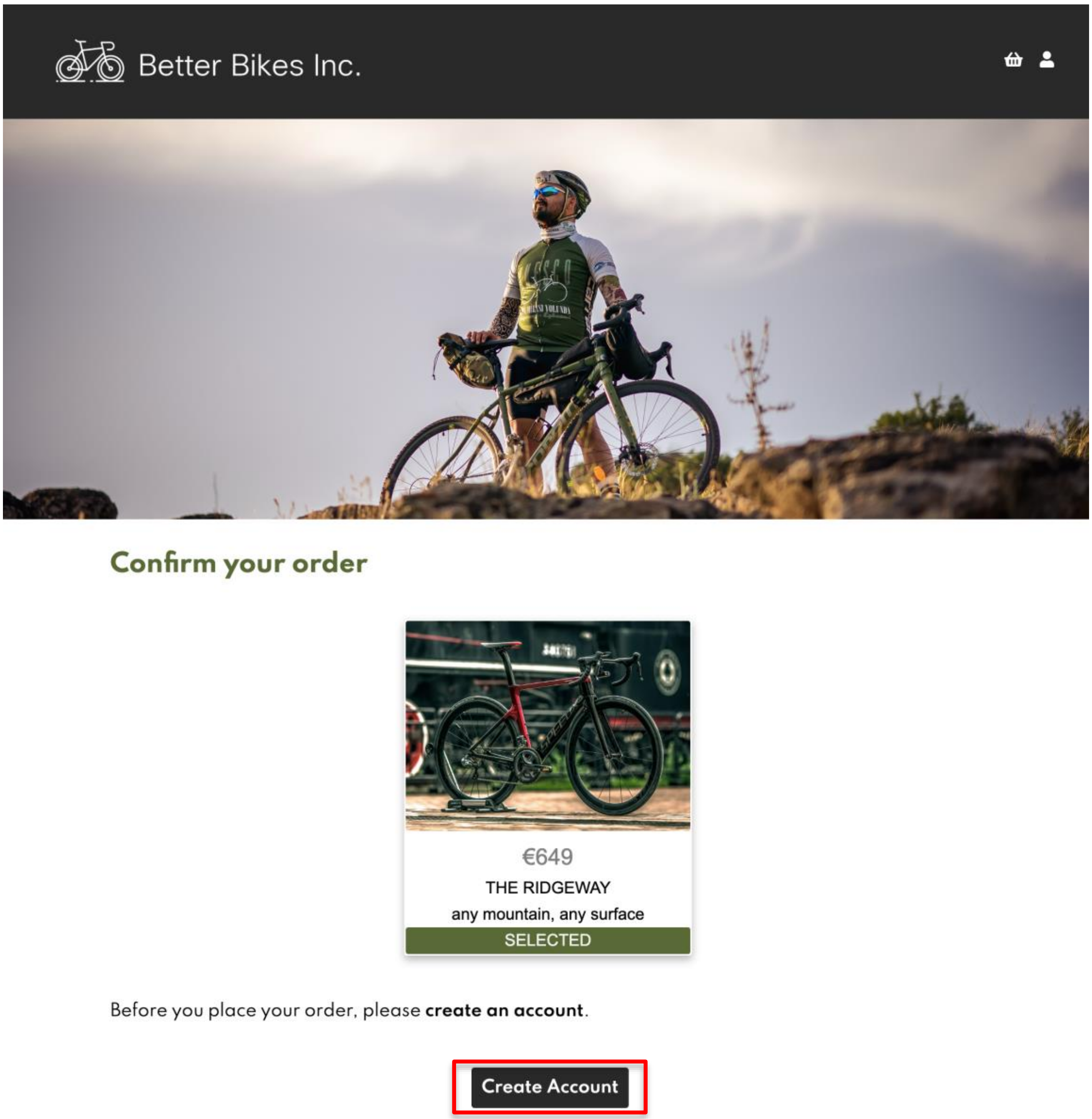


Click Save and you have done all voice configurations.

Let's test now

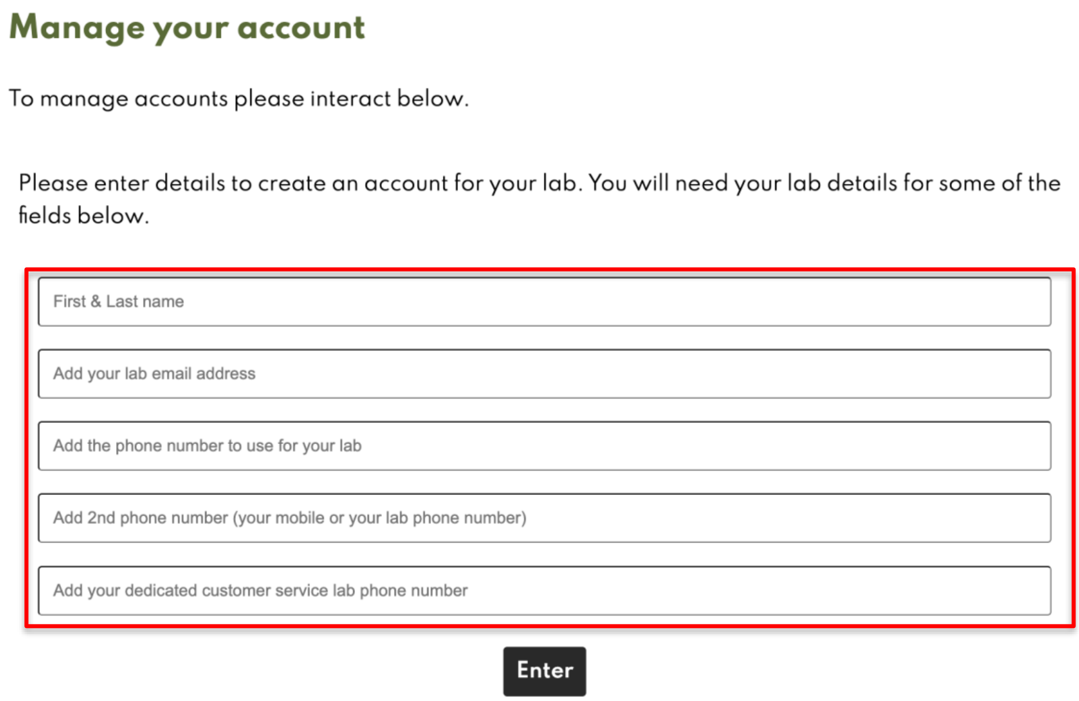
1.12 Testing the Agent Desktop

To start testing, we first need to create an account on the website created in Module 0.9.
For that please open your Glitch Website URL and click Create Account.



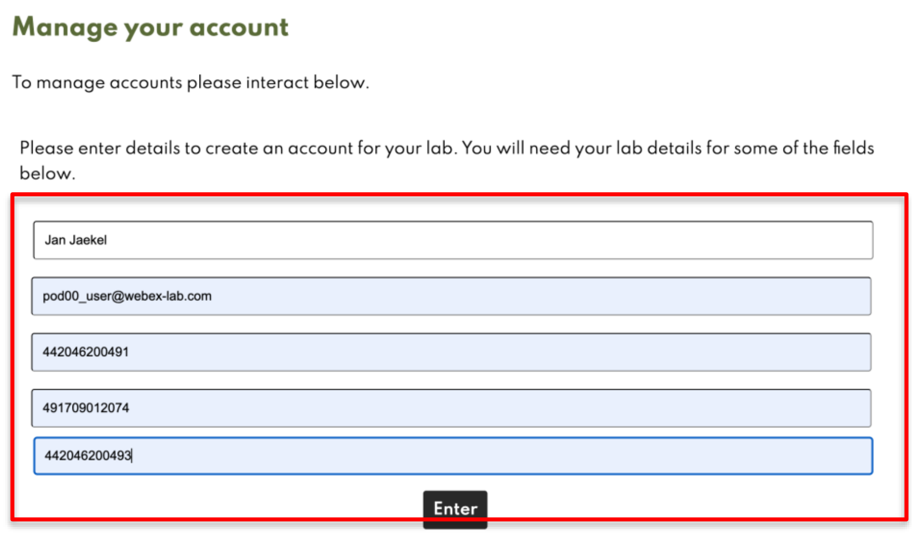
Please enter the following information to create an account:

First & Last name:	type in your (or any other) First and Last name
Lab Email Address:	your PodXX_User Email Address
Phone number:	your PodXX_User Phone Number
2nd Phone number:	can be any other Phone number (your mobile phone number for example – this will be needed when testing WhatsApp in Module 3)
Customer Service Number:	your PodXX Service Queue Number



Pod Number	PodXX_User	PodXX Service Queue
01	+442046200494	+442046200496
02	+442046200501	+442046200506
03	+442046200507	+442046200517
04	+442046200523	+442046200524
05	+442046200526	+442046200527
06	+442046200528	+442046200530
07	+442046200532	+442046200533
08	+442046200534	+442046200538
09	+442046200549	+442046200629
10	+442046200852	+442046200941
11	+442046200943	+442046200947
12	+442046200949	+442046200951
13	+442046200952	+442046200954
14	+442046200955	+442046200956
15	+442046200961	+442046200974
16	+442046200976	+442046200977
17	+442046200978	+442046200980
18	+442046200981	+442046200982
19	+442046200986	+442046200989
20	+442046200991	+442046200994

Note: The first field must not contain any numbers. You don't need a leading '+' for the phone numbers.



When the fields are filled, please click Enter.

The web page will confirm and your account is now created.

Thank you

Account created.

Thank you **Pod00 User**, your account is now created.

Click on the logo on the top left corner.

Better Bikes Inc.

Now you can confirm your order by clicking on Enter.

Confirm your order

€649

THE RIDGEWAY

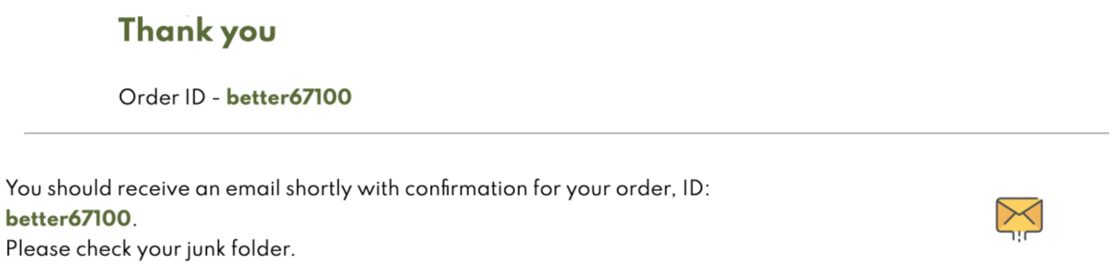
any mountain, any surface

SELECTED

To receive a delivery confirmation for your order, **please tap enter**.

Enter

The web page will confirm that an order has been placed. Your order will be created, and you'll receive a confirmation email in your PodXX_User@webex-lab.com inbox.



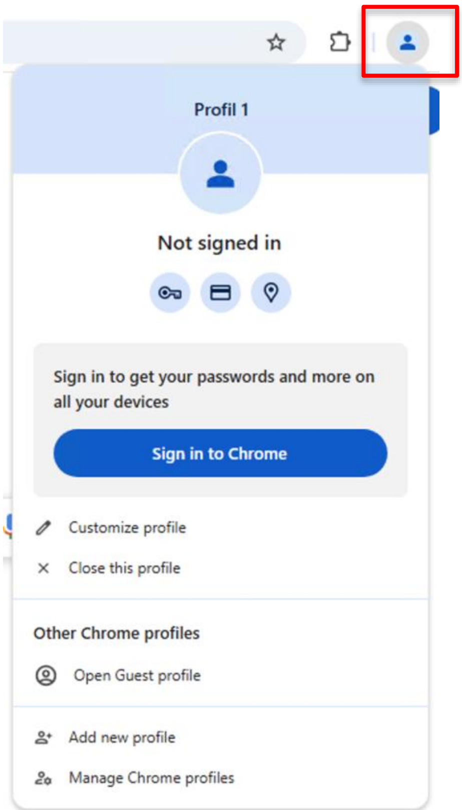
To read the mail, please login to https://webmail.webex-lab.com/ with your PodXX User account.

Now we can login to the Agent Desktop.

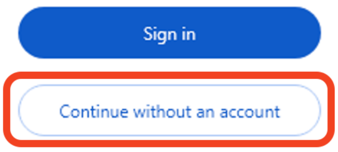
For that we need two different Chrome Profiles.

Please create two new Chrome Profiles and name Agent and Supervisor.

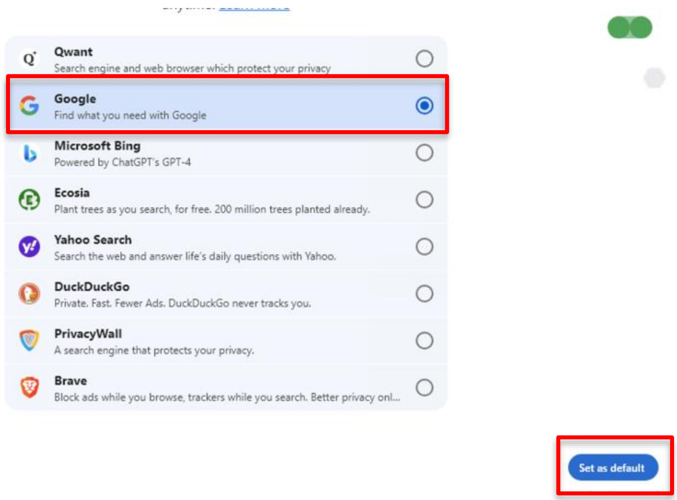
Open the Chrome browser and click on people Icon in the upper right corner.



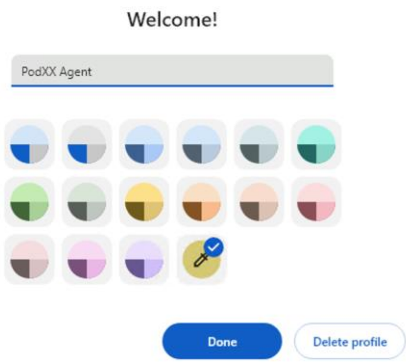
Click on Add new Profile and select Continue without an account



Choose your search machine and set it to default.



Use PodXX Agent for the profile name.



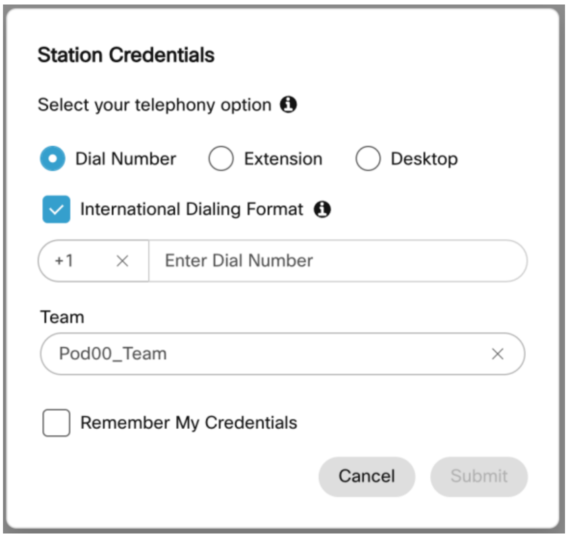
Repeat the above steps to create a profile also for Supervisor.

Open Chrome with the Agent Profile.

Open the following URL https://desktop.wxcc-eu2.cisco.com/

Login with your Agent Username podXX_agent@webex-lab.com and with the Password: **C1sc0Live25!**

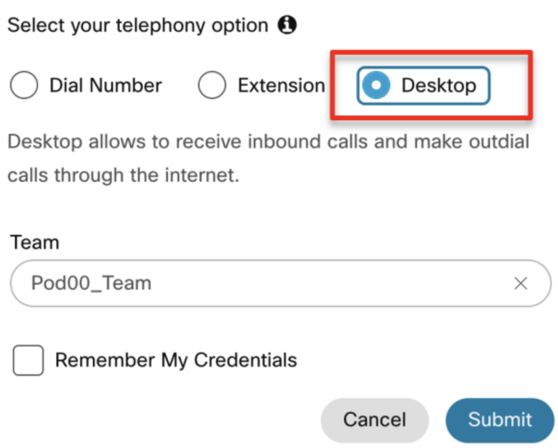
After the login you will see the following Window.



Now you can choose how you are reachable:

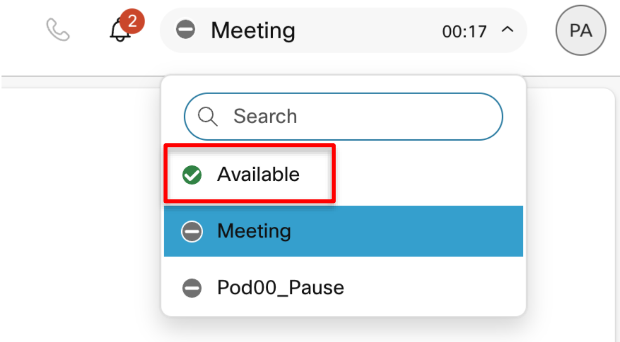
Dial Number: can be any Number worldwide
Extension: is an internal Webex Calling Extension
Desktop: is the WebRTC Calling Connection

Please choose Desktop and Submit.



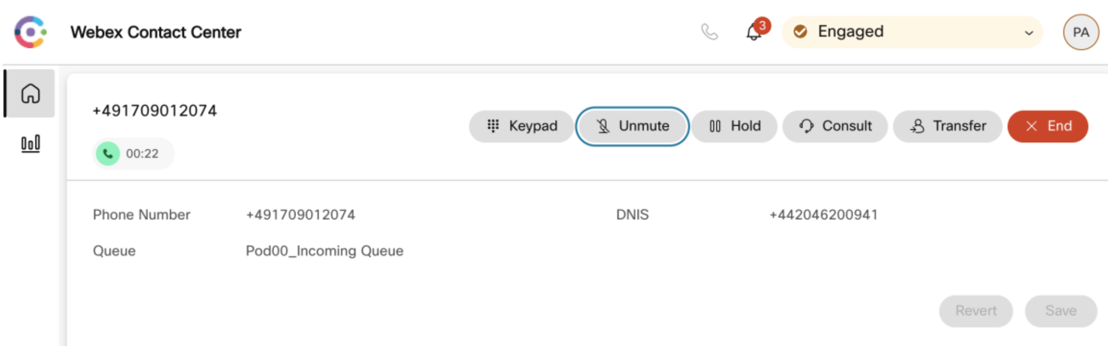
On the Status Dropdown you can see your configured Wrap-up and Idle Codes.

Please choose Available.



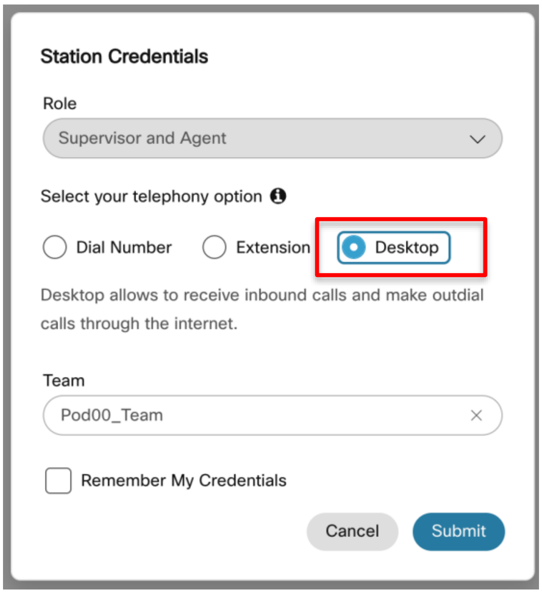
When you logged in in Webex App with the User Account you can dial the last 3 digits of your CC Number. When you are using your mobile phone, you can dial the E164 number like +442046200XXX.

When you call in you will hear a greeting, and your call is visible in your Agent Desktop. Please answer the call.



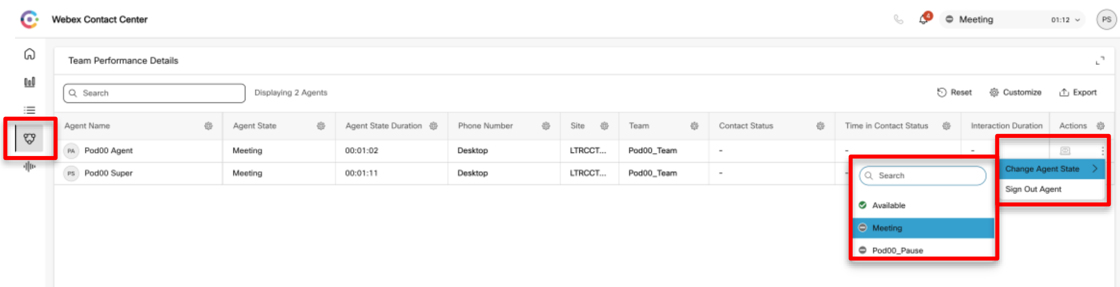
When you hang up you can choose your predefined WrapUp Code.

Open https://desktop.wxcc-eu2.cisco.com/ in a second Chrome window with profile Supervisor and login with the podXX_super@webex-lab.com and with the Password: **C1sc0Live25!**

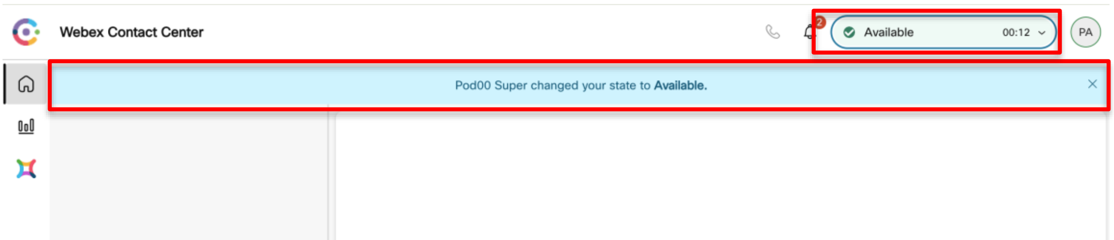


Now you see that you have more capabilities on Supervisor Desktop.

Choose Team Performance Details on the left side. Click on the three dots in the line of the Agent and change the the status to Available.



The Agent will get a notification that the status was changed from supervisor.



Congratulations, you have finished the standard lab elements for Module 1. There is an additional module element, but we recommend completing these after you have finished the standard lab in Module 3.

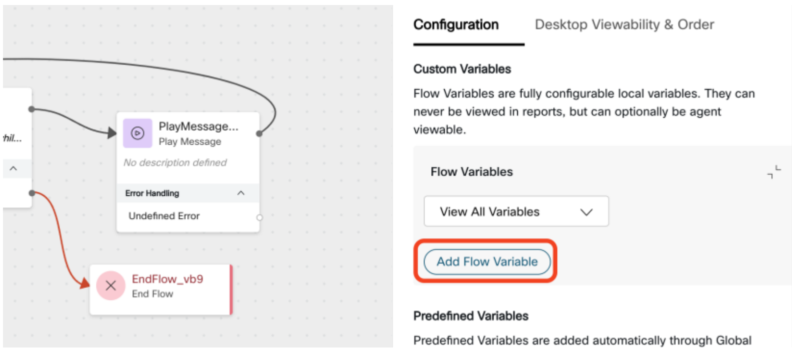
Skip to Module 2 **HERE**

1.13 Addon: Caller Identification

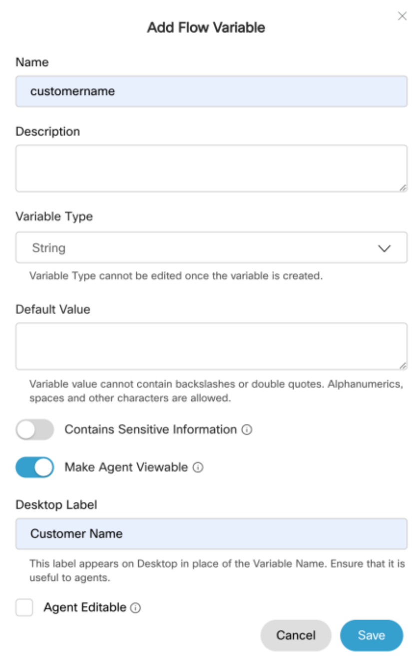
Please note, this is additional content.
This task can be skipped, but you will learn how to implement database connections for receiving more information about the caller. We recommend completing Addon modules once you have reached the end of your 'standard' lab if you have leftover time.

To do this, please open your current flow in the Flow Editor and set it in edit Mode.

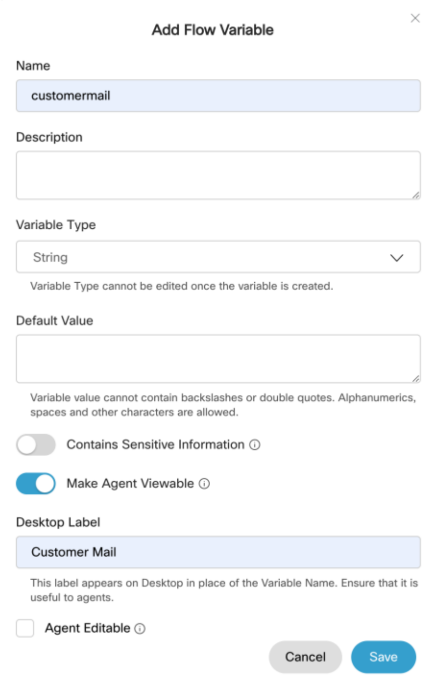
And 2 new Variables. For that click in the grey Background and add the first Variable.



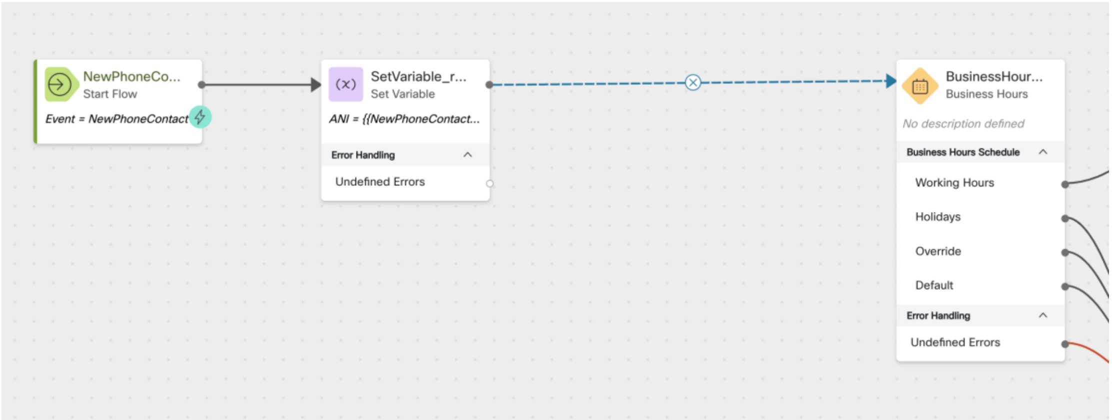
Fill out the Fields.



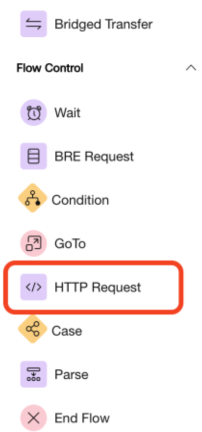
Add a second one with these fields.

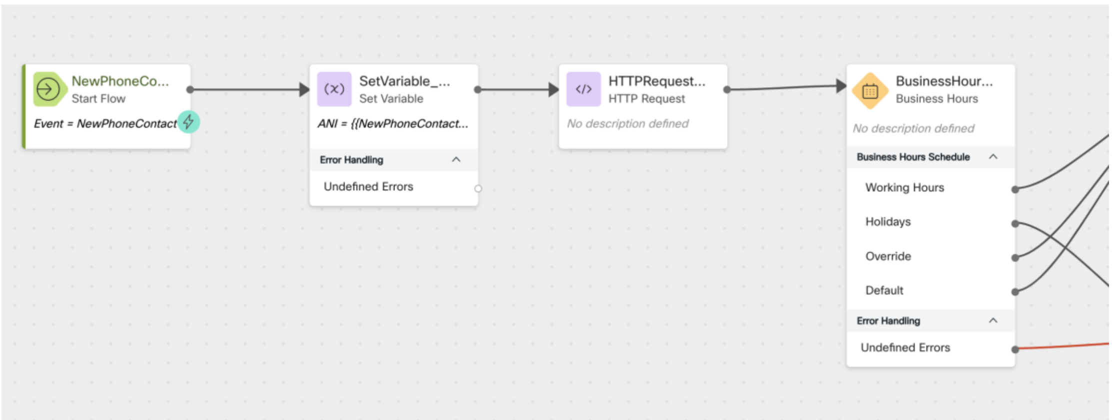


Now move the SetVariable Node to the left and remove the Connection.



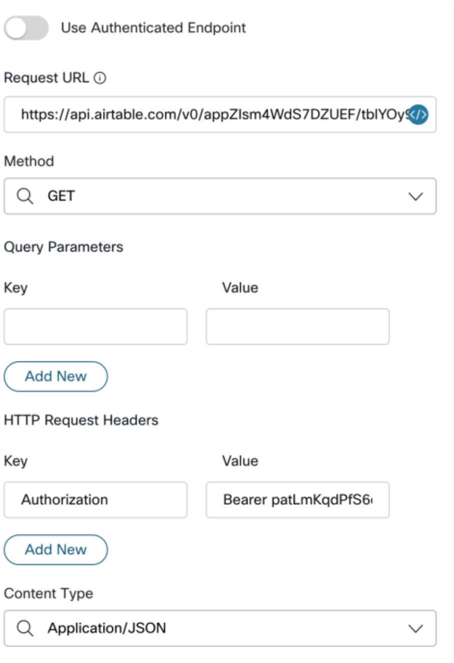
Add from the Library a HTTP Request Node in the middle and connect all three nodes.





Click on the HTTP Request Node and fill out the following fields.

Use Authenticated Endpoint: disabled
Request URL:
https://api.airtable.com/v0/appZIsm4WdS7DZUEF/tblYOy9e3q82JEc6l/?filterByFormula=OR(mainTel = '{{ANI}}', secondaryTel = '{{ANI}}')
Method: GET
HTTP Request Headers:
Key: Authorization
Value: Bearer patLmKqdPfS6dPRUT.5163d15090da940e887cda69cc25aec3b6df012018efcdd16e1c5fc757af2686
Content Type: Application/JSON

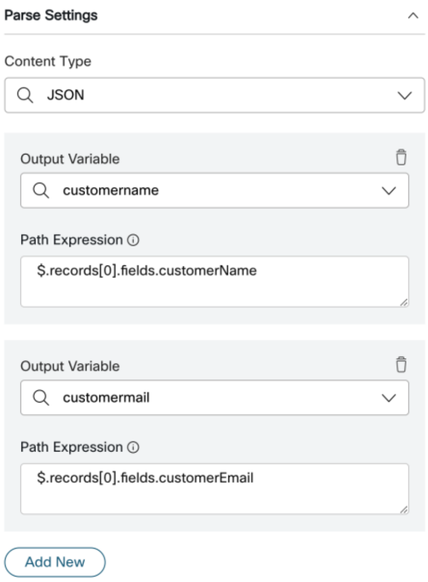


Now we must parse the Response and fill our variables

Set Content Type to JSON and add to new Output Variable:

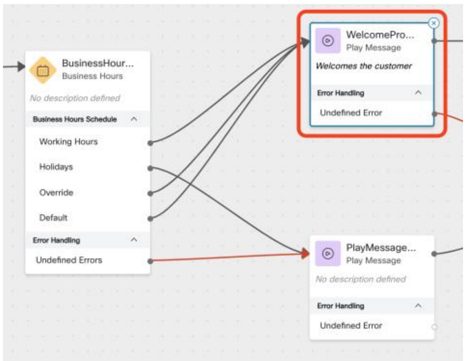
Output Variable: customername
Path Expression: $.records[0].fields.customerName

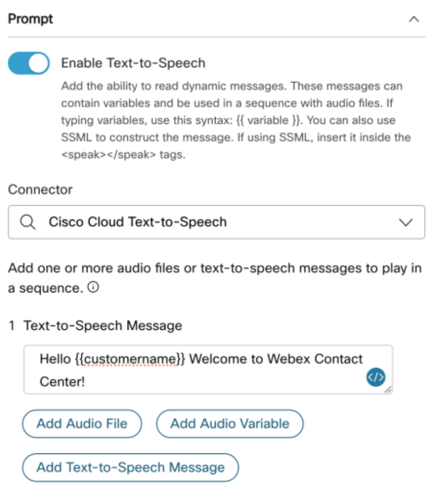
Output Variable: customermail
Path Expression: $.records[0].fields.customerEmail



After that click on the WelcomePrompt Node and change the Text-to-Speech Message to

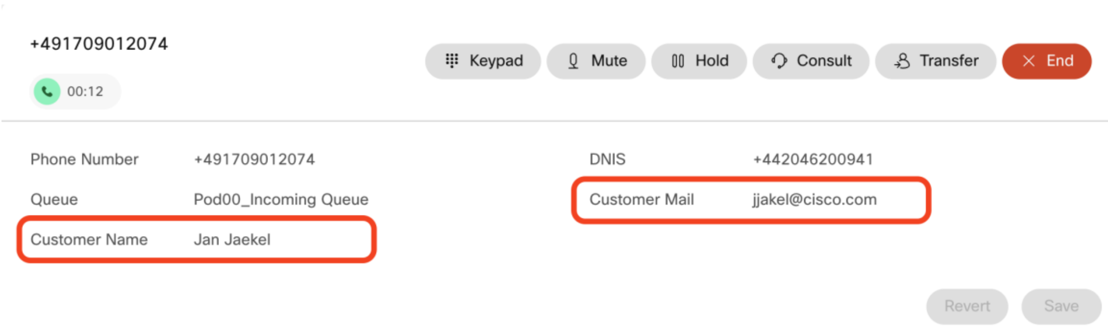
Hello {{customername}}. Welcome to Webex Contact Center!





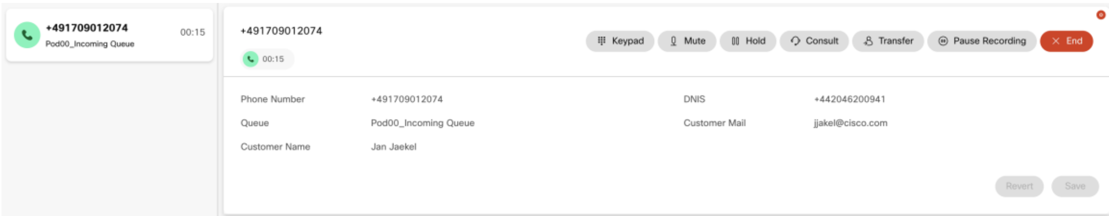
Now save the Flow and Publish.

Set your Agent to Available and call in again. Now your will hear your new greeting and see 2 new Variable is Agent Desktop.



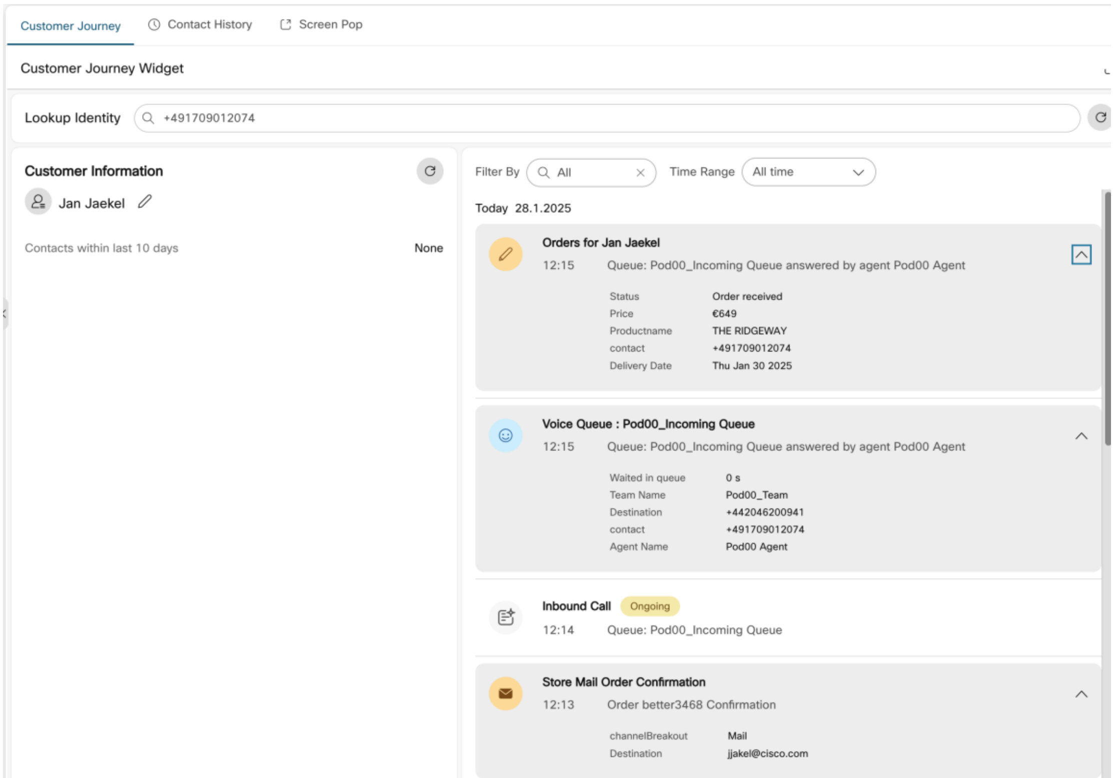
!! The solution is also on Flow List, named with Flow_PodXX_V2.

Make a new Call. Now you will hear a greeting with your personal name, and you will see it in Agent Desktop.



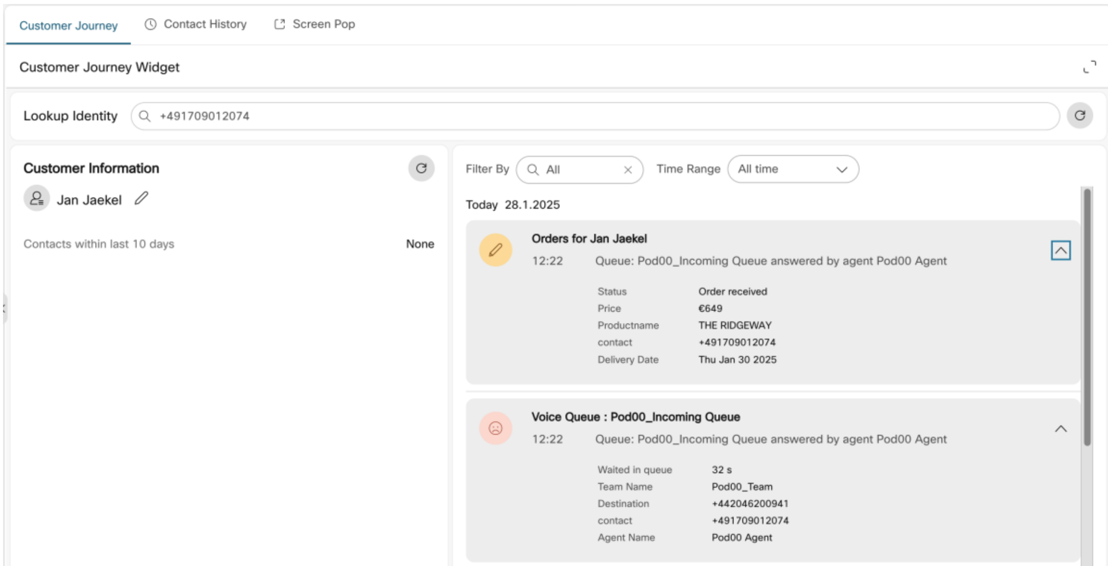
All Events in CJDS are preconfigured in the Flows. For the identity that calls in you see every interaction.

1. System has sent an email confirmation
2. Calls arrive in near real-time
3. System is getting all of the relevant information from your orders



Set your agent status to meeting and call in again. Wait 30 seconds before you set the Status to available. Let's see what's happen.

For example: depending on the wait time we can set individual settings.



Congratulations, Module 1 has been finished!

Module 2 – Web Chat

For this lab we will configure a Web Chat asset for our ‘return.html’ page we built on glitch earlier. This lab will teach you how to:

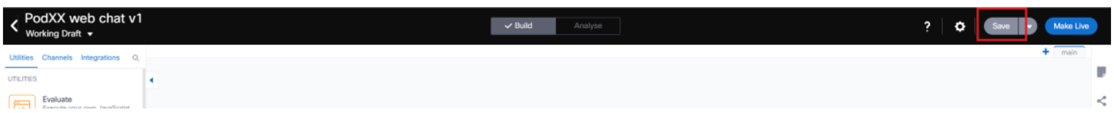
- create an asset, assign the relevant settings in WxCC
- create a customer journey workflow in Webex Connect with deterministic routing
- allow visitors of your Better Bikes website to engage with Natural Language Processing and generative AI based bots, browse through some products using rich-media based, dynamic carousels
- utilize rich, HTML email templates
- escalate chats to WxCC
- enable granular-level debugging
- report to external systems

In the Addon tasks we will enhance this customer journey further by adding an email verification loop and supplementing our agents in WxCC with a generative AI based summary of the conversation.

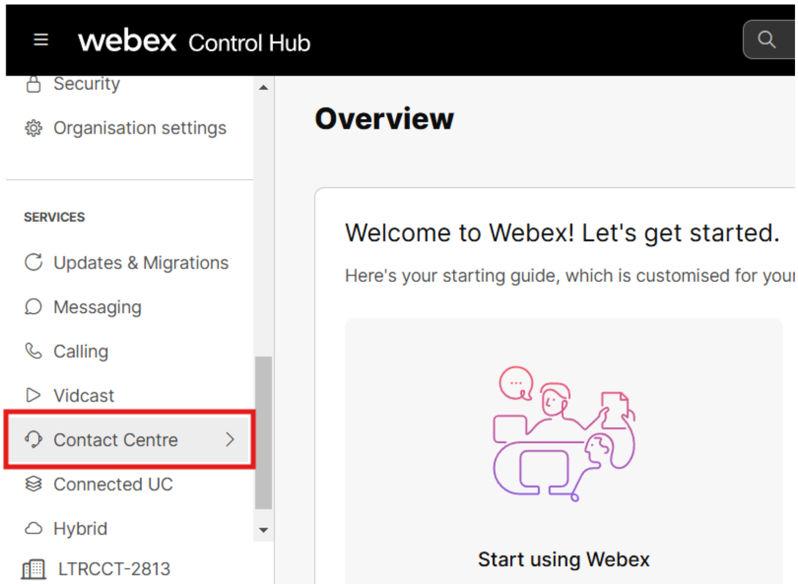
Before we get started with this lab we will need to grab some pre-configured items from the Module 2 folder in GitHub. Head to https://github.com/ChrisW0lf/CL-EMEA25-LTRCCT-2813/tree/main/flows/WebexConnect/Module2 and download the following files:

- 33865_webchat-v1.workflow
- 33864_webchat-v2.workflow
- taskbotcorpus_10492.json

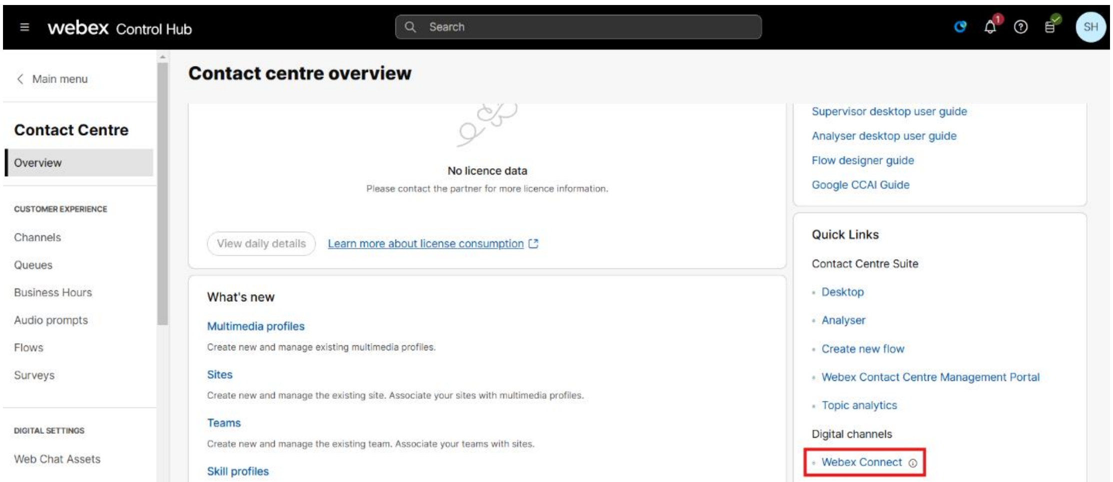
Additional note: Whilst you configure your flows in Webex Connect for Module 2 (and 3) we recommend you use the Save flow function semi-regularly, so progress is maintained in case you need to refresh or open the flow page.



Webex Connect Web Chat Channel Configuration
Head back to Webex Control Hub (https://admin.webex.com), then using the left-hand menu navigate to Contact Centre.

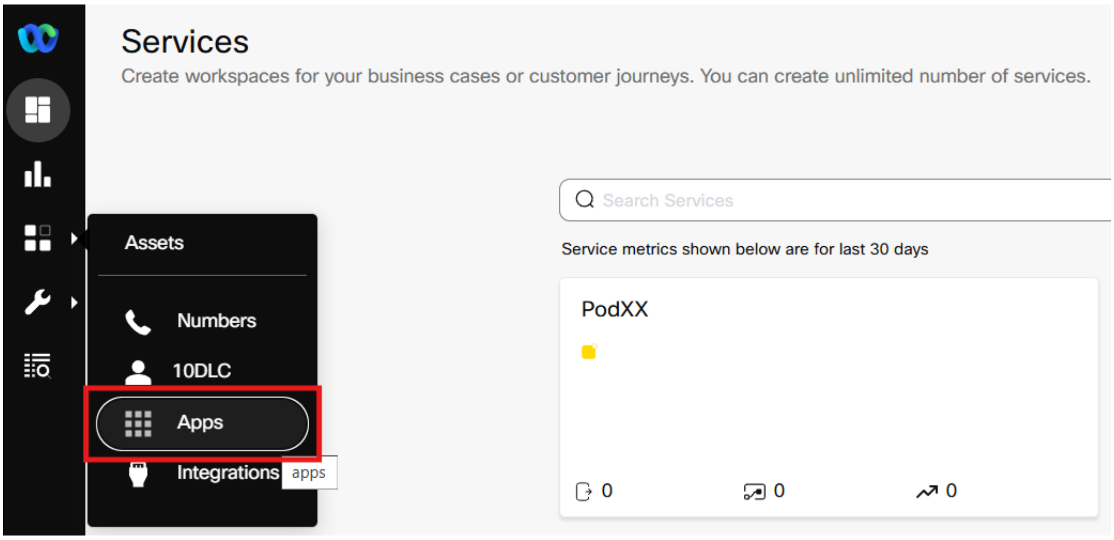


You will land on the WxCC overview, using quick links in the bottom right hand of the screen, click on the Webex Connect link.



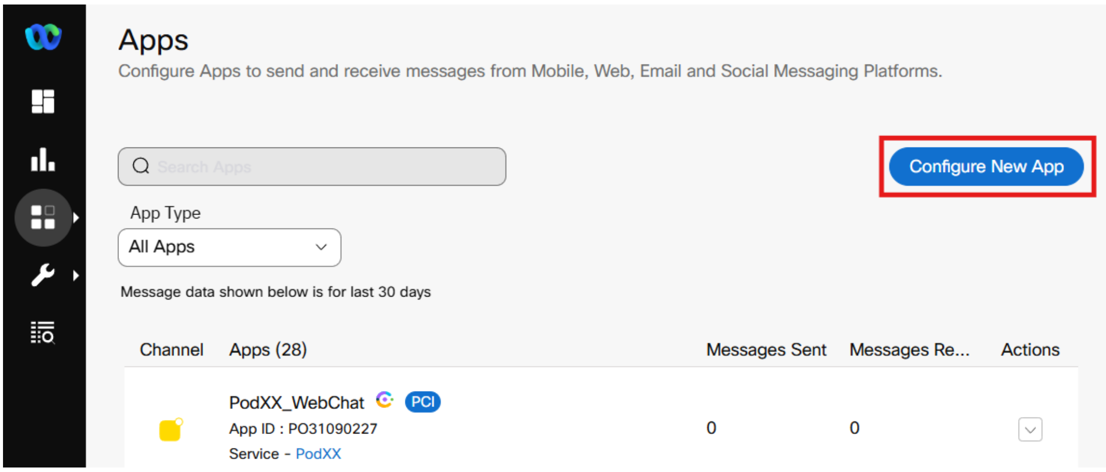
This will open the Webex Connect platform in a new tab.

Using the top-left-hand menu navigate to Assets, then Apps.



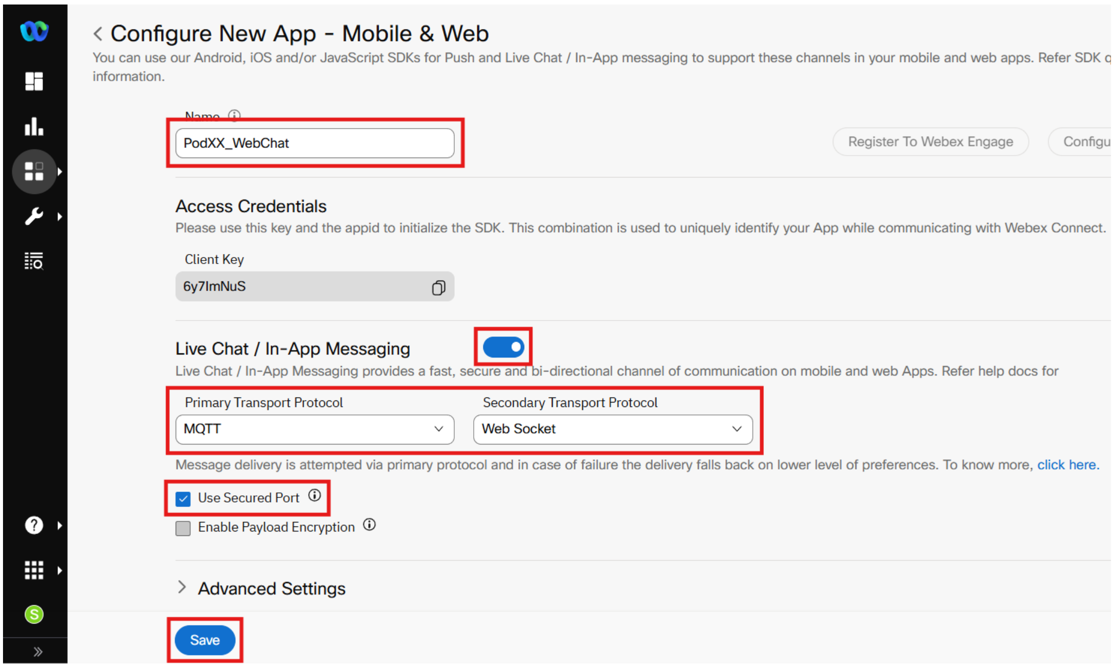
This is where we manage our rich-capable channels, such as RBM and WhatsApp, which will we use later in module 3.

Click Configure New App, and then select Mobile / Web.



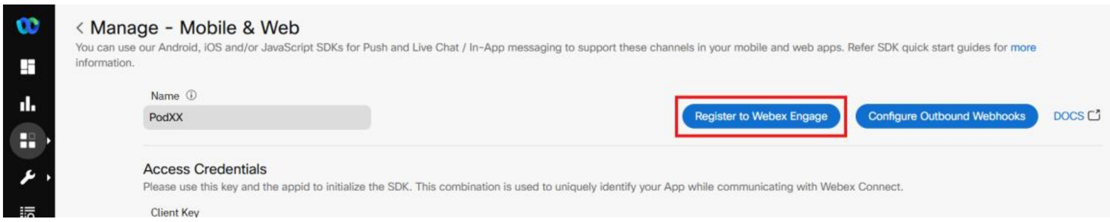
Here we will configure your web chat channel asset. Fill in the following fields:

Name:	PodXX_WebChat
Live Chat / In-App Messaging:	Enabled
Primary Transport Protocol:	MQTT
Secondary Transport Protocol:	Web Socket
Use Secured Port:	Enabled

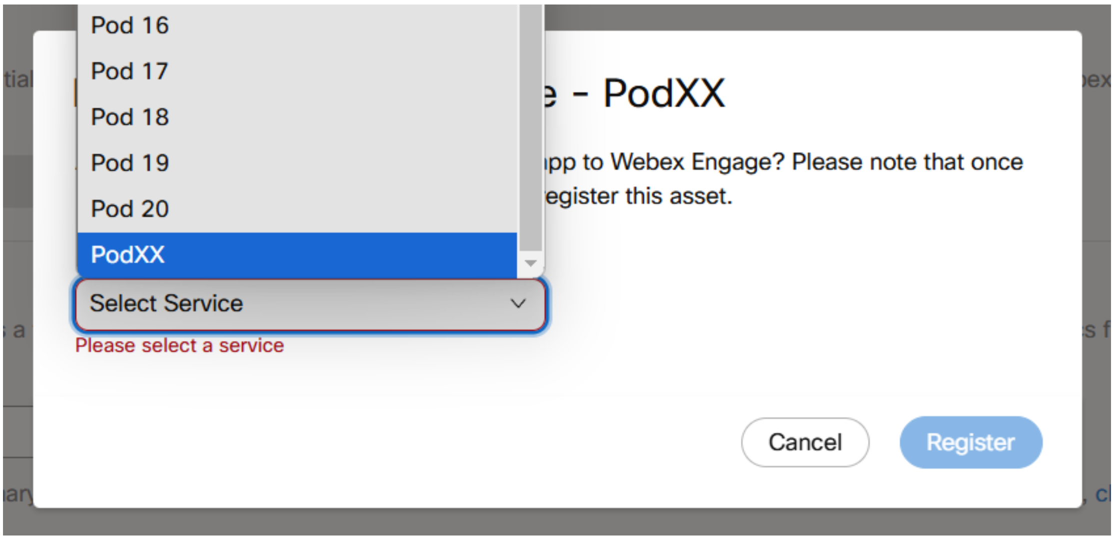


Then click Save.

The asset is created, and now we need to link it to Webex Engage. Click the Register to Webex Engage button.



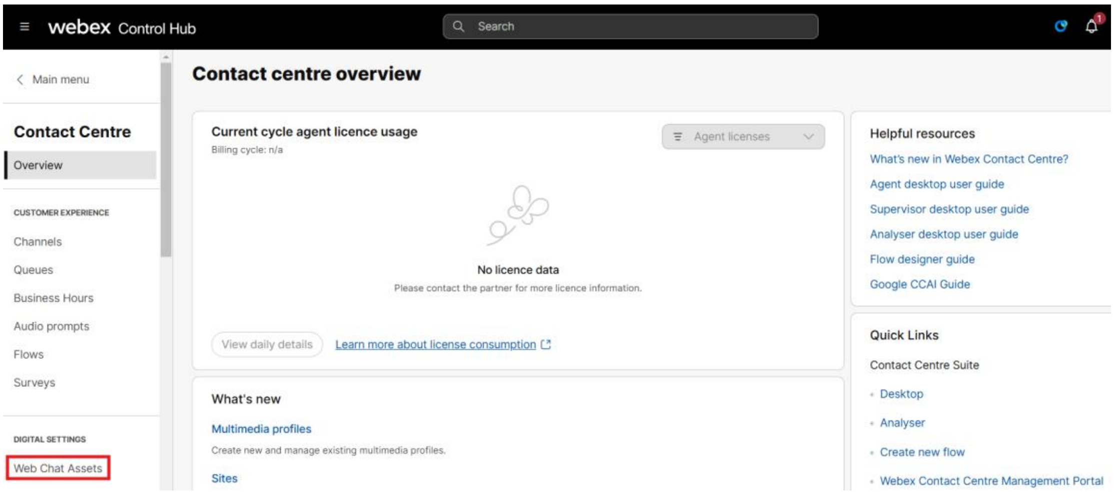
Select your service and click Register.



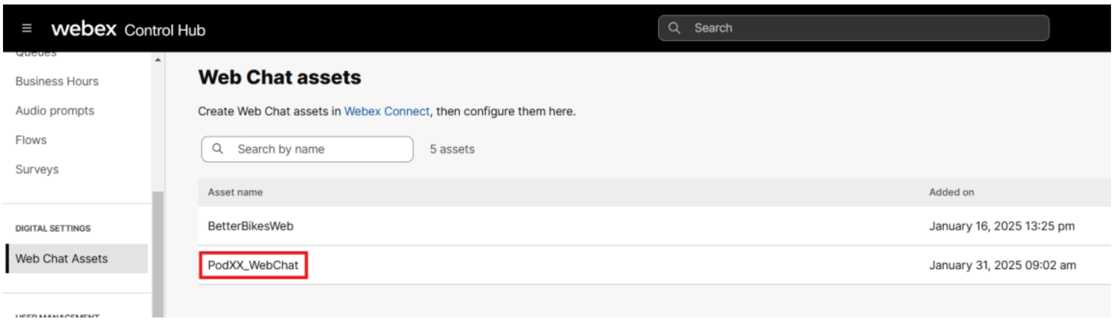
You’ve now finished configuring your app on Webex Connect. Let’s head back to Webex Control Hub (https://admin.webex.com), then to Contact Center again from the left-hand menu.

2.1 Control Hub Configuration

In the left-hand menu now navigate to Web Chat Assets.



Select the Web chat asset that you configured in Webex Connect.

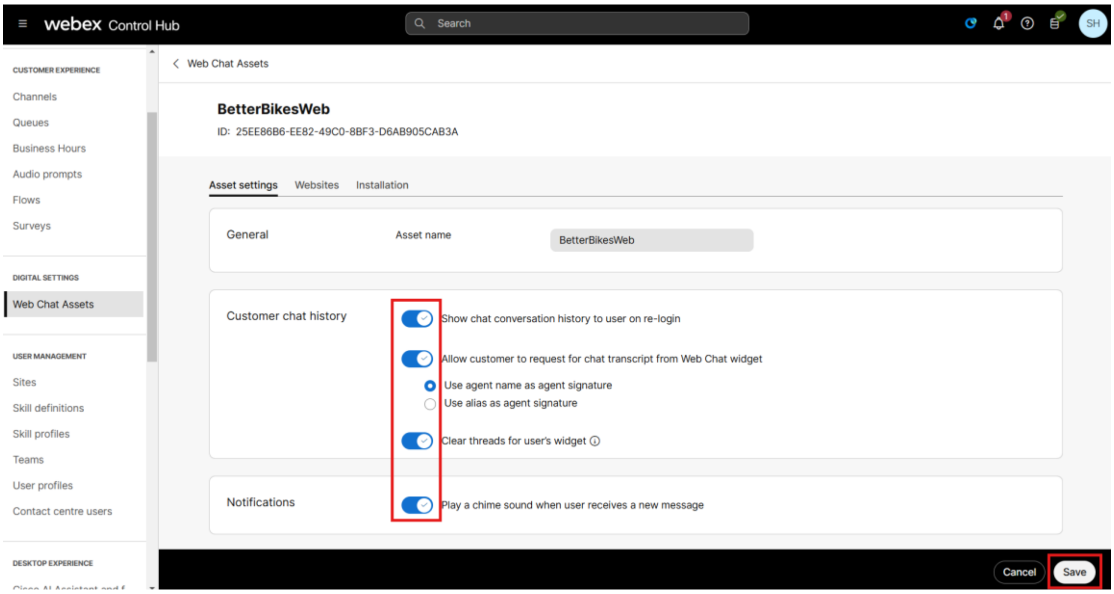


Now we will configure the asset and widget settings, then finish with an installation. Let's configure the Asset settings first:

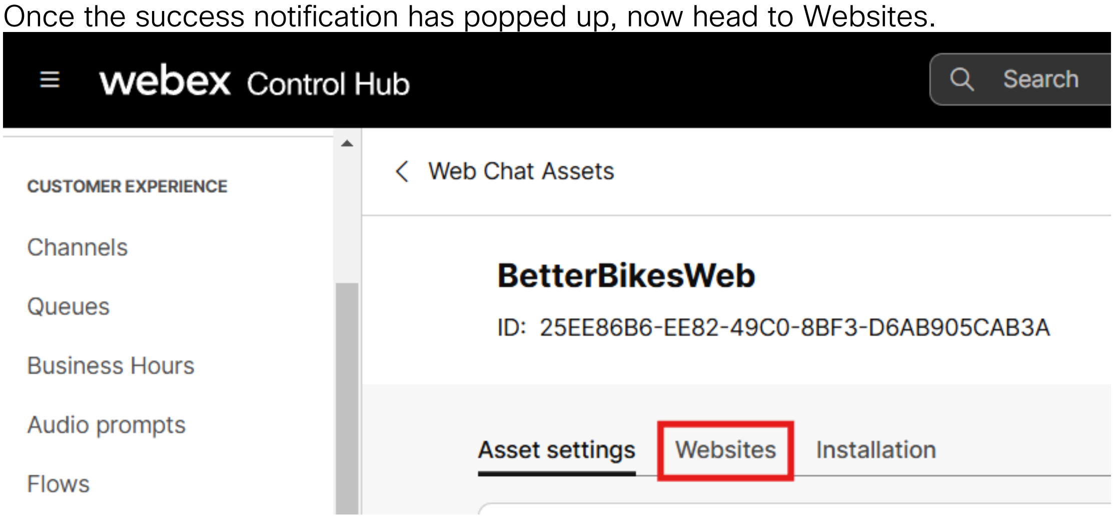
Enable the following toggles:

- Show chat conversation history to user on re-login
- Allow customer to request for chat transcript from Web Chat widget
- Clear threads for user's widget
- Play a chime sound when user receives a new message

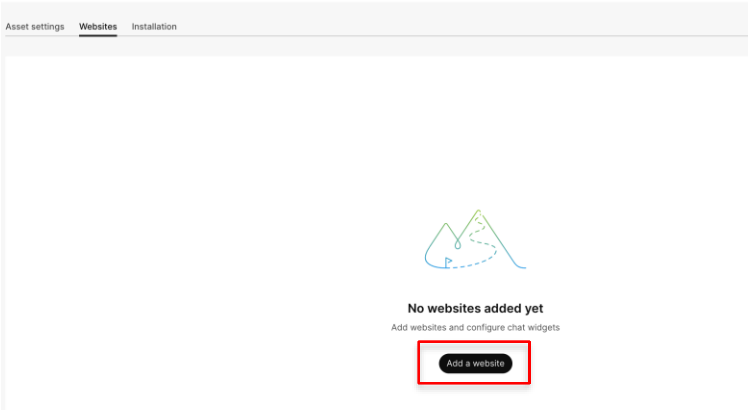
Then hit Save.



Once the success notification has popped up, now head to Websites.

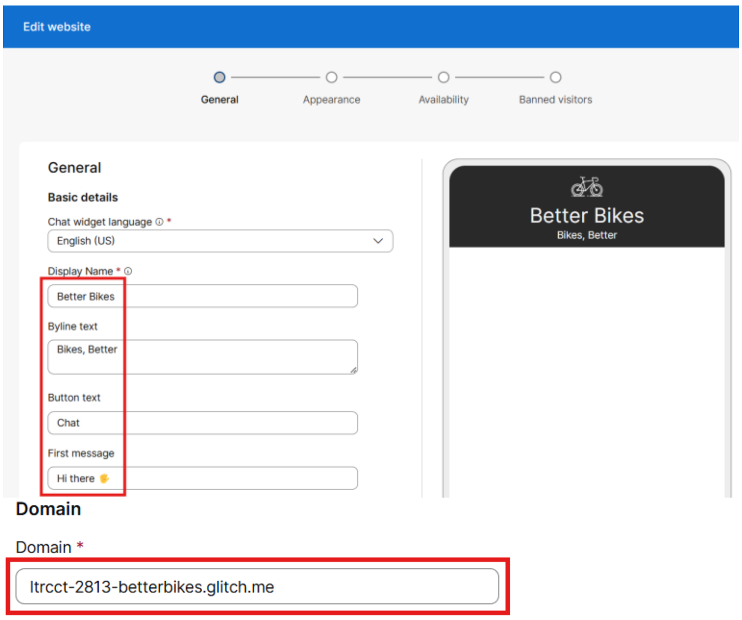


Then click Add a website.



Fill in the details:

Display Name:	Better Bikes
Byline text:	Write something for a strapline or similar
Button text:	Chat, start or similar
First message:	Hi there 🖐
Domain:	Your Glitch Website URL without the http:// in the beginning, e.g. 'first-second-third.glitch.me'



Then hit **Next**.

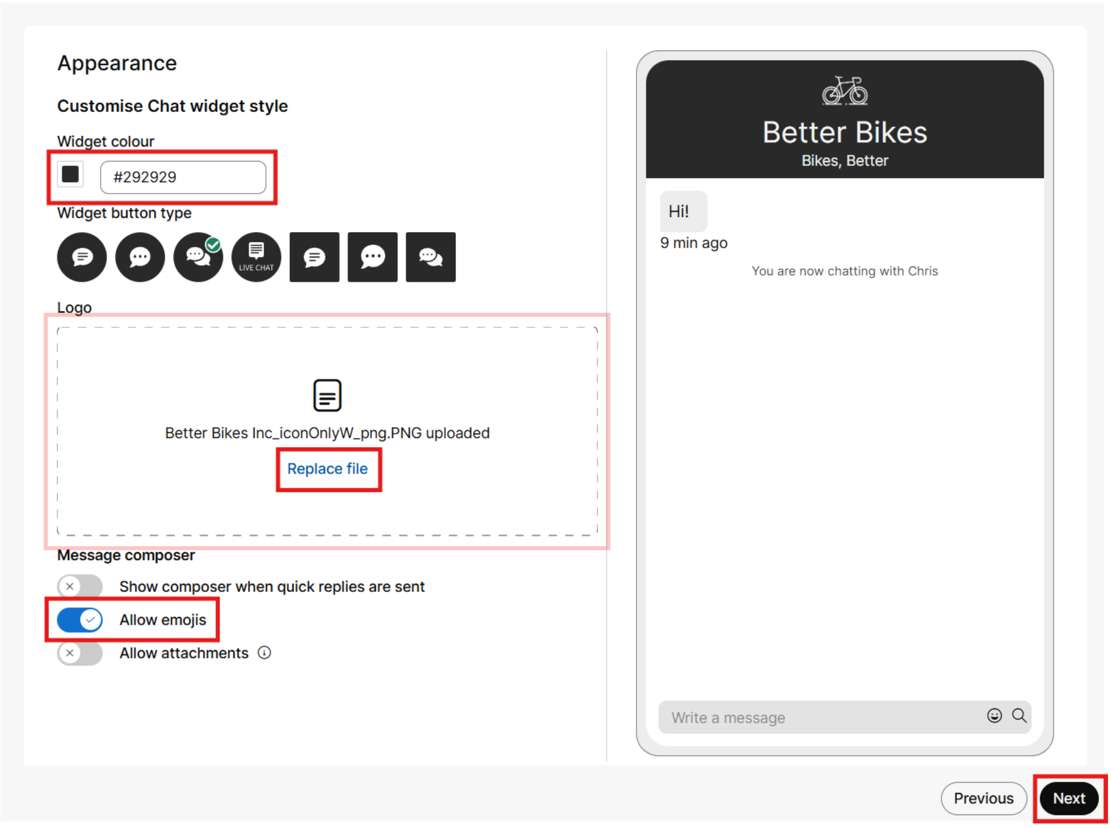
Head to https://awsprodemailassets.s3.amazonaws.com/0ae1dbc4-e5aa-46c3-8aed-55995aabf5b7/BetterBikesInc_iconOnlyW_png_11402228692944843.png, right-click on the image and save the image to your downloads file.

Now go back to the Appearance page so we can finish configuring your web chat asset.

Set Widget colour to #292929.

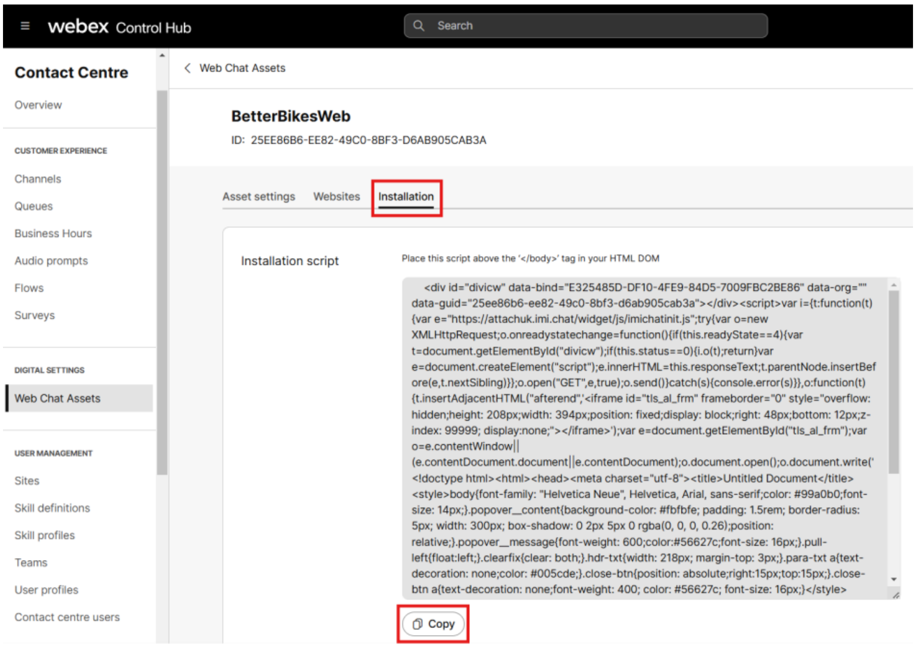
You can either click Replace file and upload the image you just saved or simply drag the file from your local storage into the upload box.

Then click Next.

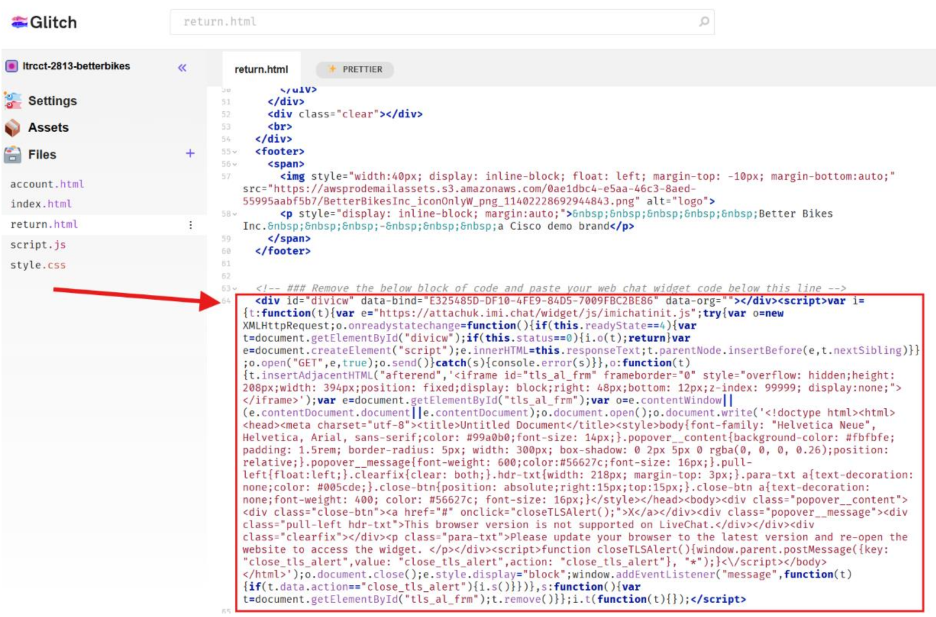


Then click Save and next, and then Done. Your web chat asset is created and ready for installation on your glitch website.

Head to the Installation tab and then copy the code.

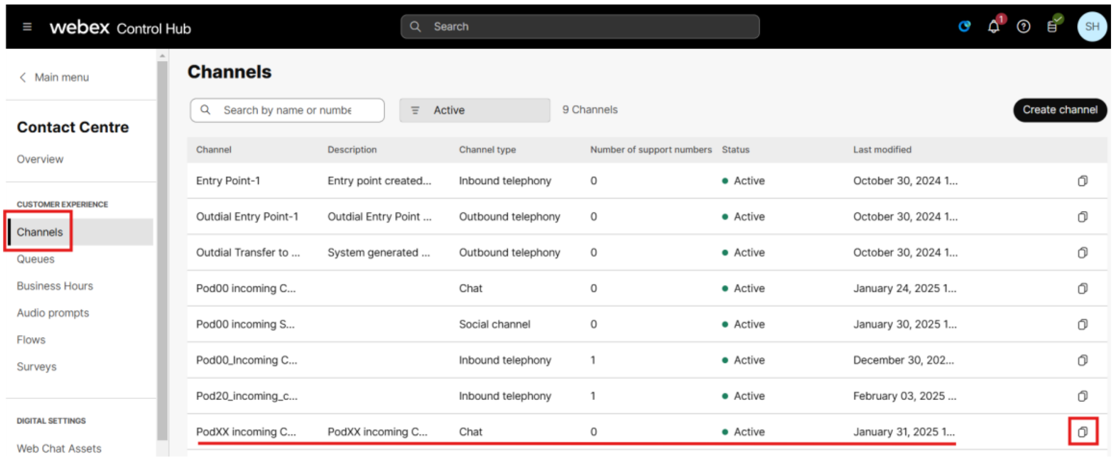


Let's head back to your Glitch Project page and open the return.html file. Paste the copied code where indicated, to replace the existing code on line 64.

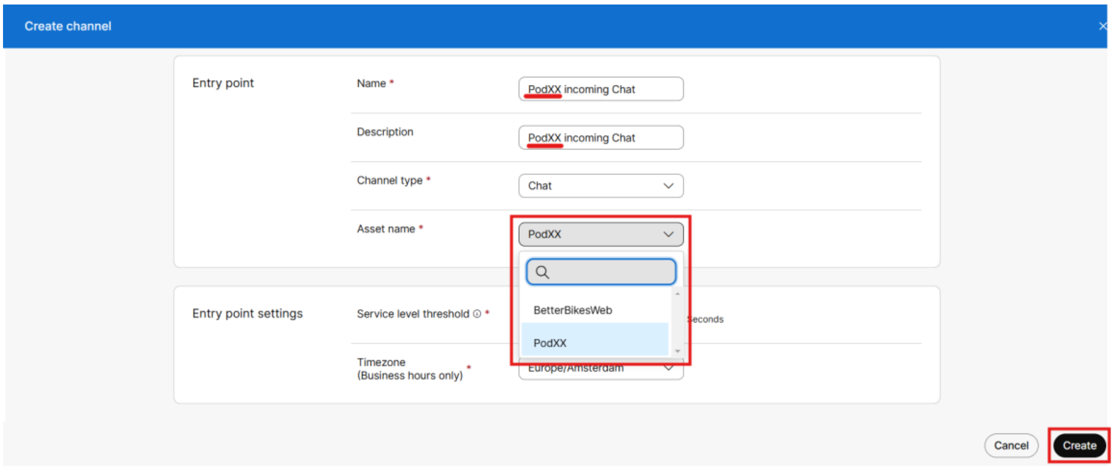


Now after we have configured the asset we need to assign the web chat asset a queue and a team must be created so that we can elevate chats to the Contact Center. To do this, head back to Contact Center and click on Channels on the left.

We are going to use the PodXX incoming Chat channel, click on the copy icon at the right of the line.

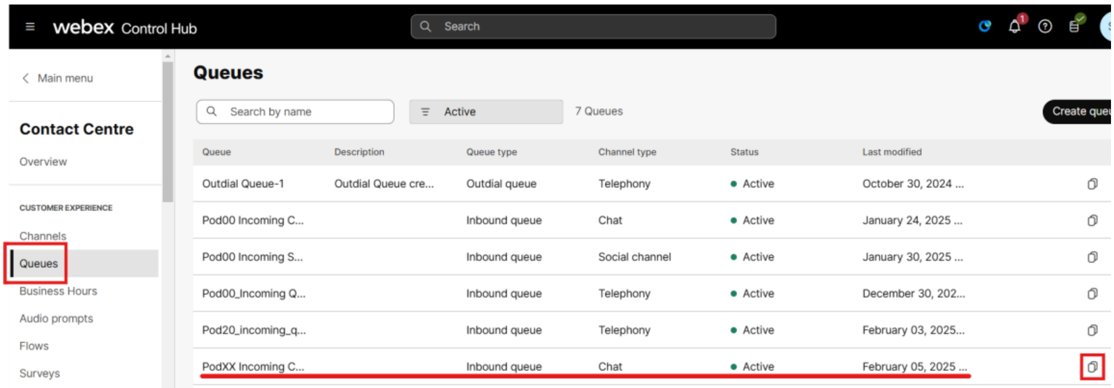


Change the fields with PodXX to your relevant assigned Pod number, select your asset from the Asset Name dropdown and click Create.

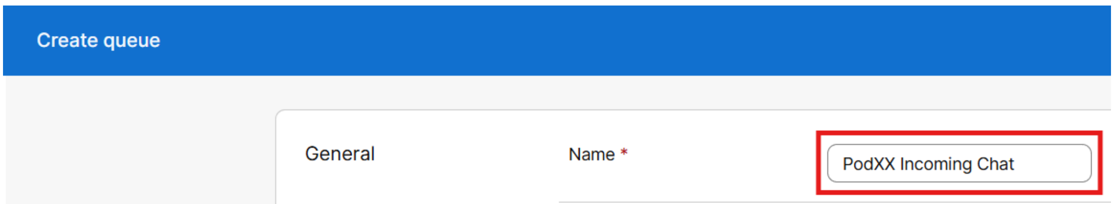


Now we need a Queue where the Channel is assigned, click in the left sidebar on Queues.

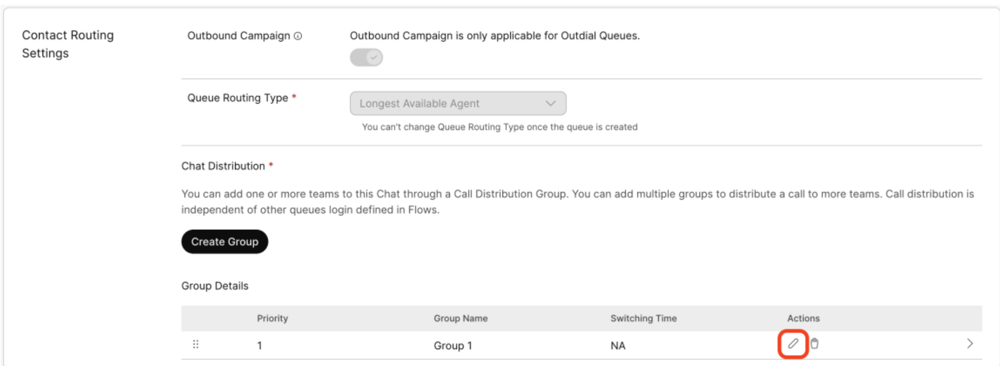
We are going to use the PodXX incoming Chat queue, click on the copy icon at the right of the line.



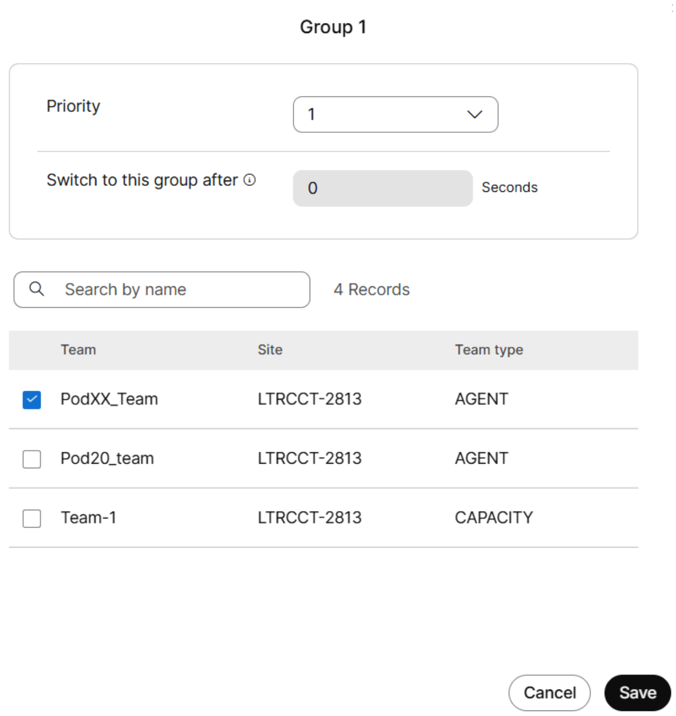
Change Name with PodXX to your relevant assigned Pod number and remove the '_copy'.



In Contact Routing Settings, under Group details, click on the Edit Pen for existing Group 1.



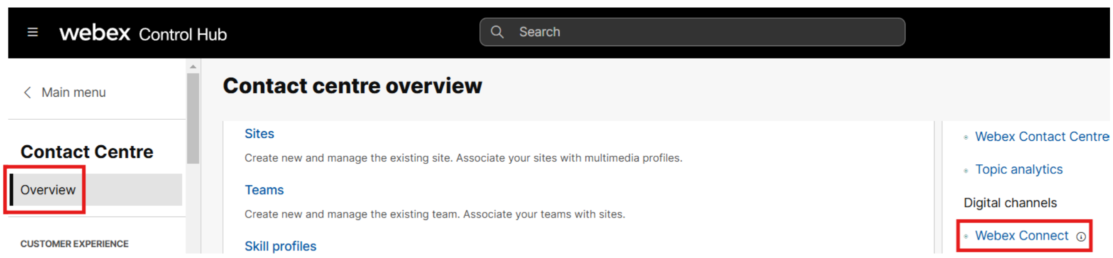
Remove the assignment for PodXX_Team then choose your PodXX_Team and Save.



Then click Create.

Your web chat is ready for using in a Webex Connect workflow.

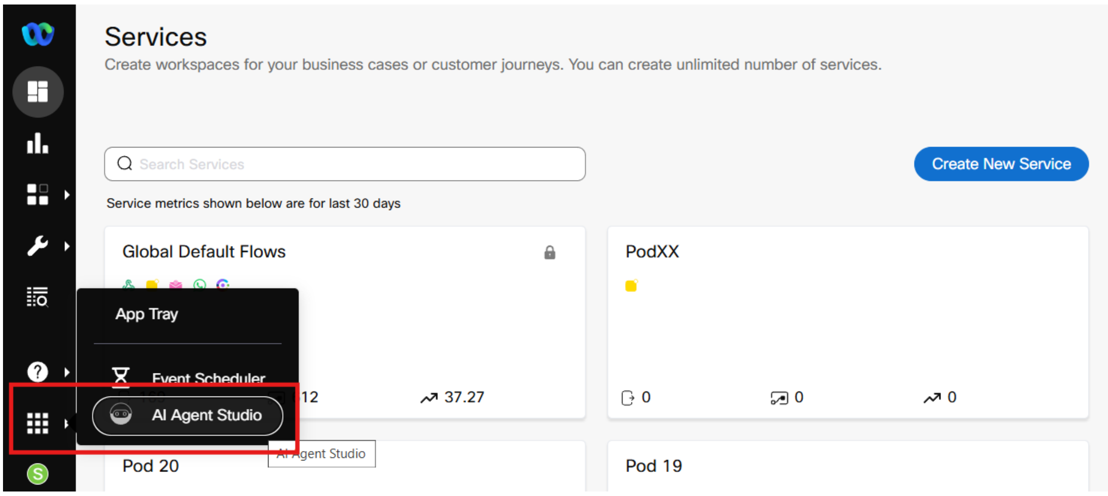
Head back to Overview in the left-hand menu and then let's head back to Webex Connect.



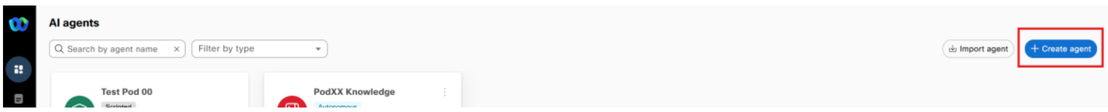
2.2 Webex Connect: AI Agent Studio

For this part of the lab, we will use a pre-configured, scripted, task-oriented bot. We will detail parts of the AI Agent Studio, however if you would like to understand how to configure AI agents in more detail, you can use the online guides here.

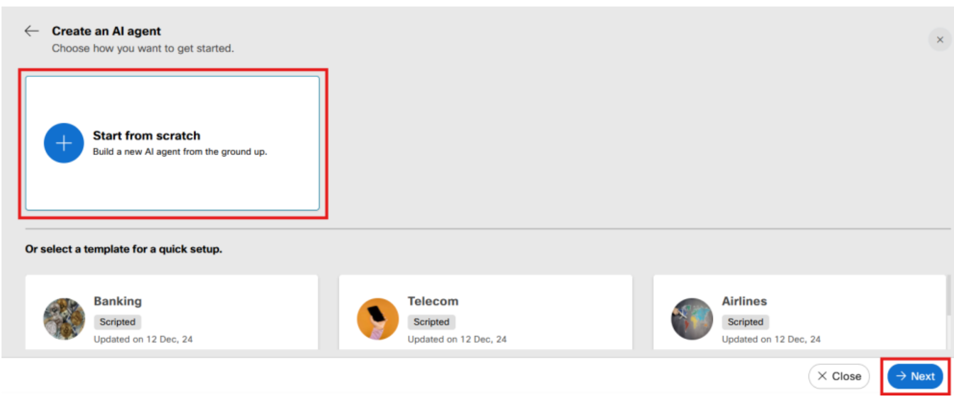
Let’s get started – in the bottom left-hand menu, hover on App Tray and select AI Agent Studio – we’re going to configure a Scripted Task Bot for your lab.



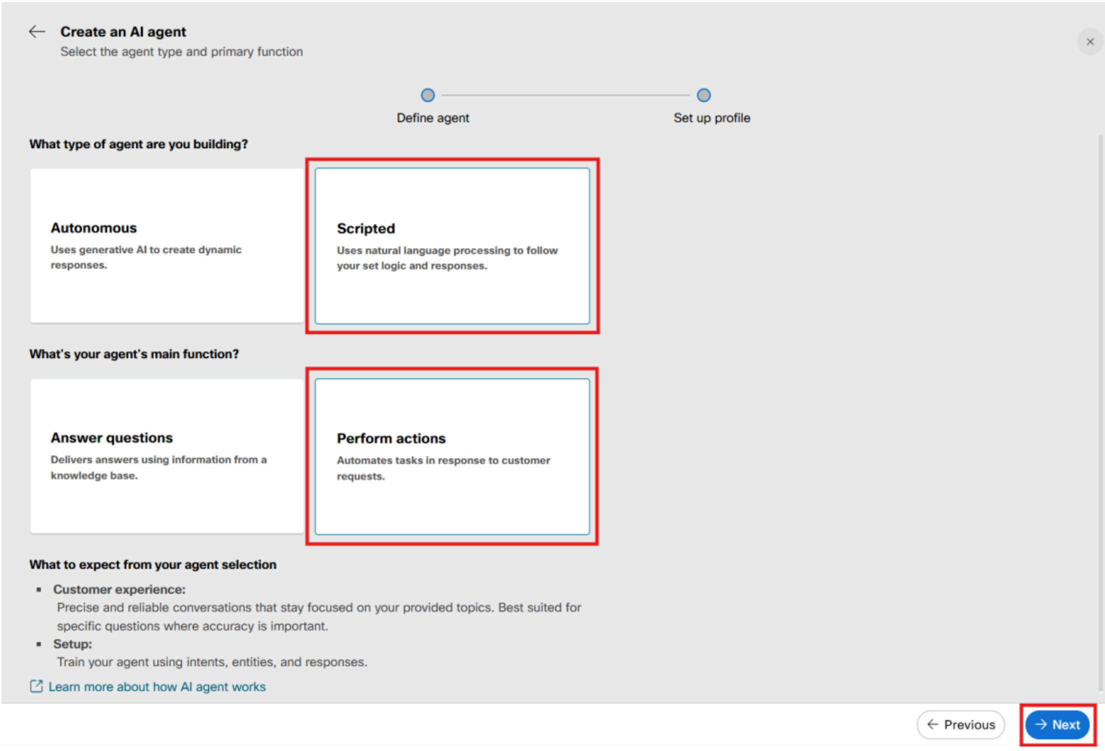
Now let’s import the agent we downloaded from GitHub. Click on Create agent.



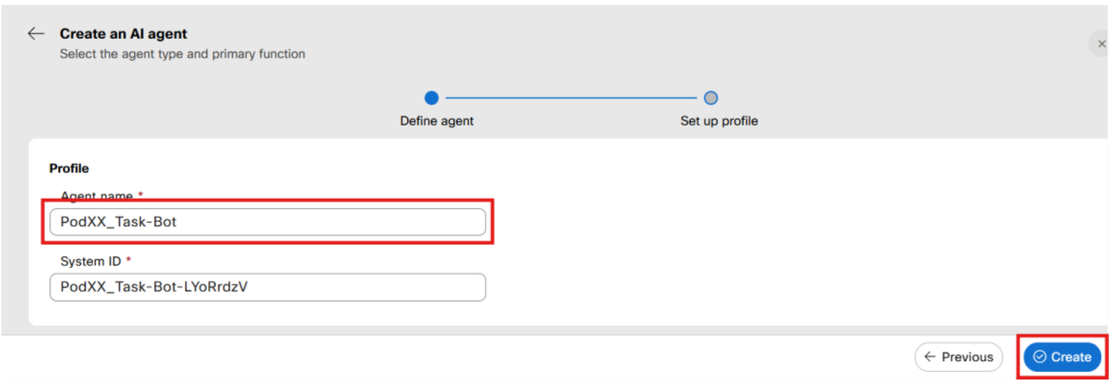
Then Start from scratch and Next.



Then Scripted, Perform actions and Next, this will create a task-oriented bot that is scripted by us, trained by our inputs.

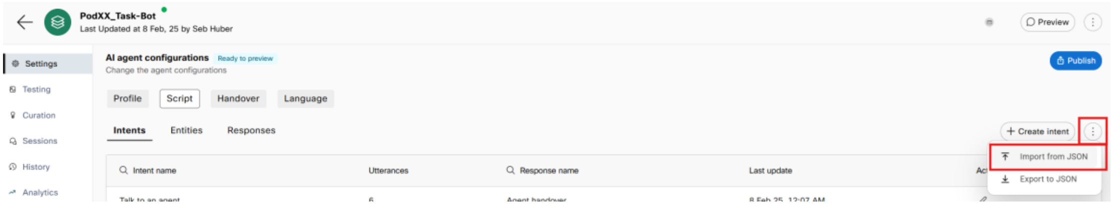


Now we'll name bot, your 'PodXX_Task-Bot' (remember to replace the XX with your relevant pod number) and click Create.



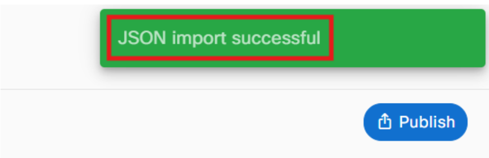
This takes us to the Script tab, on the Settings page by default. Here is where we upload the JSON file containing the bot corpus that you downloaded from GitHub earlier. You will notice that the 'Intents' listed will grow and the 'Utterances' for certain intents will increase in number too.

Click on the kebab menu and then Import from JSON.

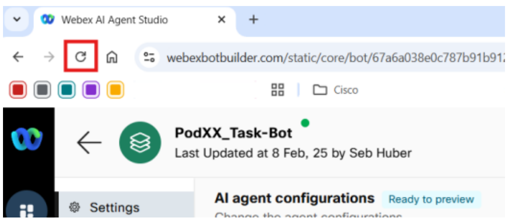


Toggle the Overwrite current intents and entities on, then click Browse and select the JSON file we downloaded from GitHub (taskbotcorpus_10492.json). Now click Import.

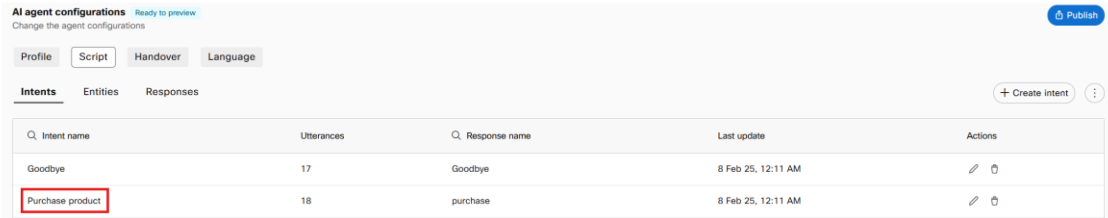
You will get a JSON Import Successful message pop up in the top right of your screen.



Give the page a quick refresh.



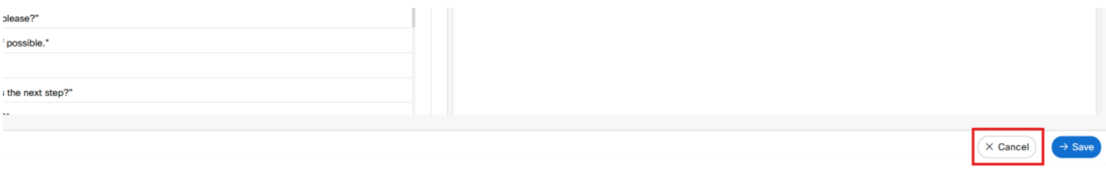
You'll also see a new intent, Purchase product, let's take a quick look at this Intent, click on the intent (anywhere in the row).



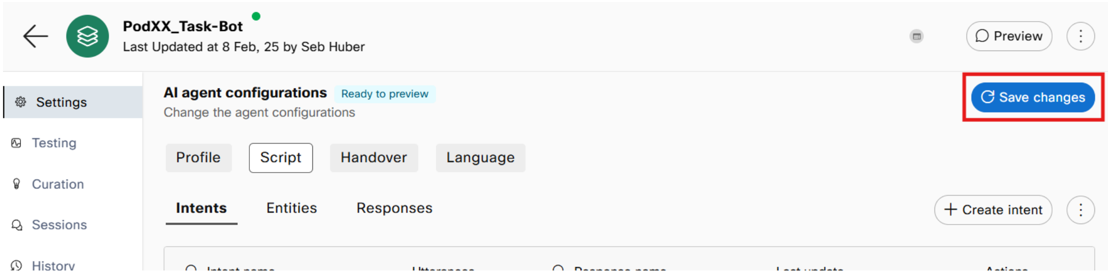
Here you see how the intent is configured:

- **Intent and utterances**: these are different ways in which a user of our bot may ask or mean something, so we are training the bot to recognise the terms we expect a user to say when wanting to match this intent. A cool feature here is that we can use generative AI (gen AI) to provide us with more utterances, to help train our bot.
- **Slots**: This lists the entities we are trying to retrieve from the conversation. For our lab we are attempting to understand the product choice a user makes. If we needed more details we list entities we are aiming to attain in chronological order. Linked to each entity we have a template key, which is a template designed to ask the user to provide us with an answer that matches an entity we are looking for. Another example use case, would be for a travel booking bot, our bot recognises the user is trying to get travel details for holidays, an intent. The entities we would need to attain then are the destination/s they wish to travel to and where they are travelling from, what dates and how many people are travelling, all entities.
- **Response**: The dropdown, filled with '**purchase**' in our demo, is the template that we send to the user once we have finished fulfilling the users intent.

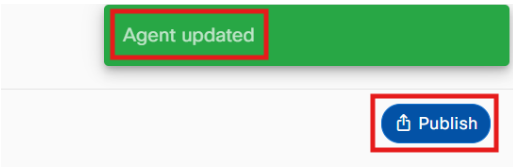
Let's head back to configuring our bot, click Cancel to exit the intent page.



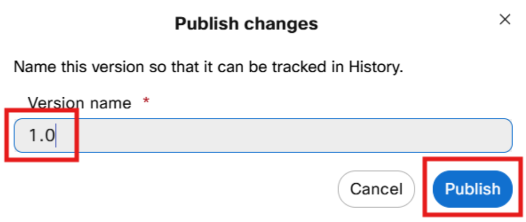
Now we need to train our bot and make it live. Click Save changes.



You will see a success message pop up and you can now Publish. Click Publish.



Give your version a name, such as ‘1.0’ then Publish.



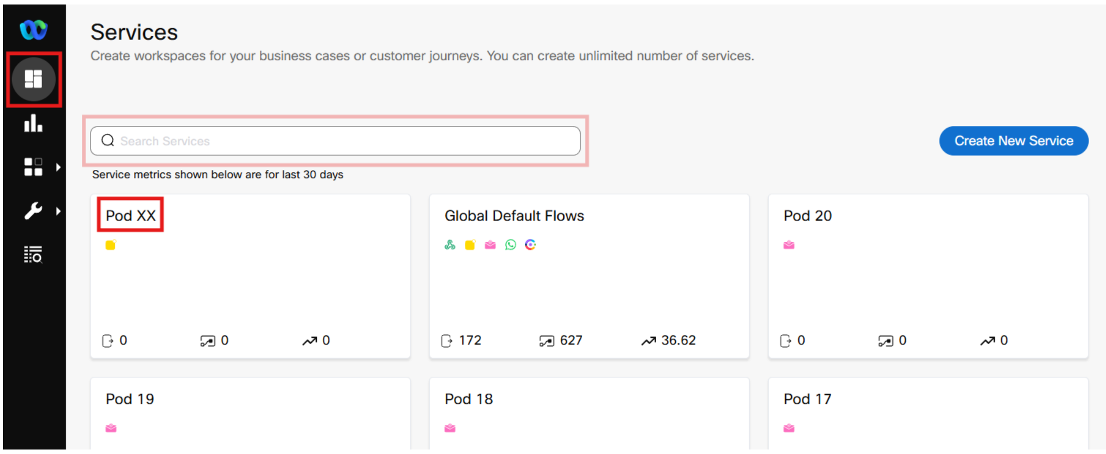
You will see another success message and the Save Changes button will be grayed out meaning your bot is now updated and trained – ready for use later in our lab.

Note: There will be an optional task later in this module to curate the bot’s understanding and responses. Normally when working with clients we aim to deploy the bot in phases with rounds of testing to ensure the bot is well trained and ready for a production environment.

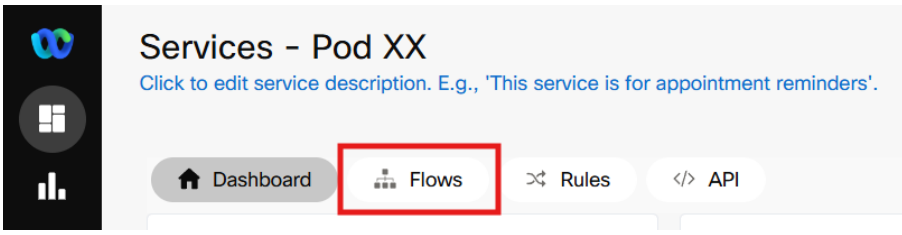
2.3 Webex Connect: Web Chat Workflow

Let’s head back to Webex Connect, switch tabs in your browser.

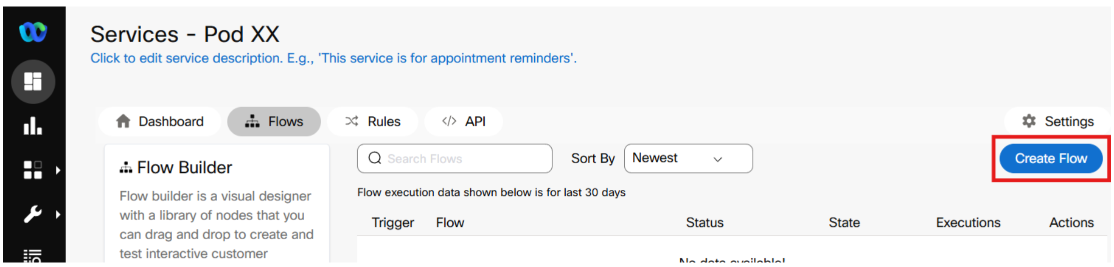
You should be on the default landing page, Services. You can either scroll down to locate your service, named with your corresponding lab Pod number (e.g. PodXX), or you can search in the search bar for your pod by typing in your Pod name and hitting Enter on your PC keyboard, then click on the container with your Pod ID.



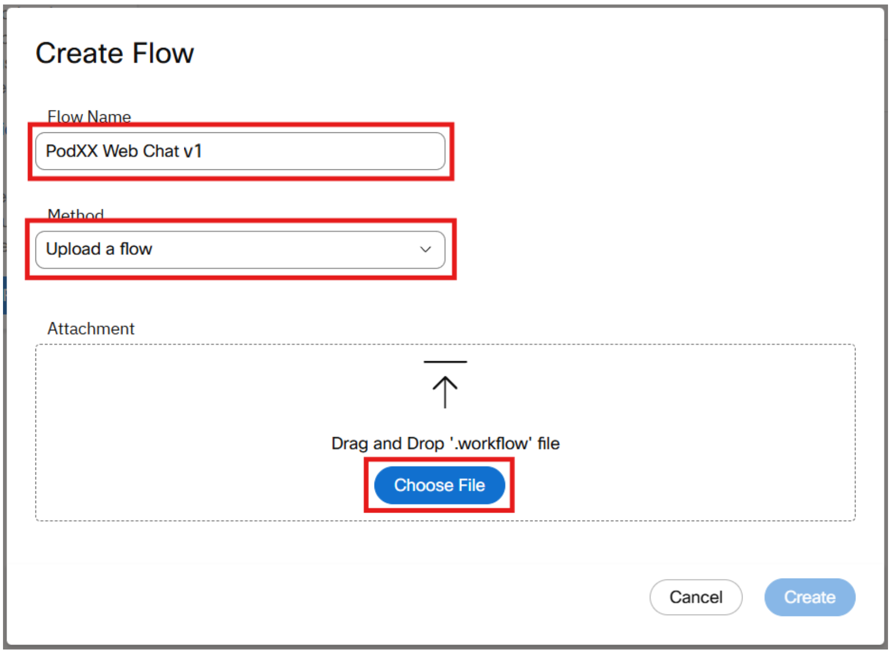
Head to Flows.



Click Create Flow.



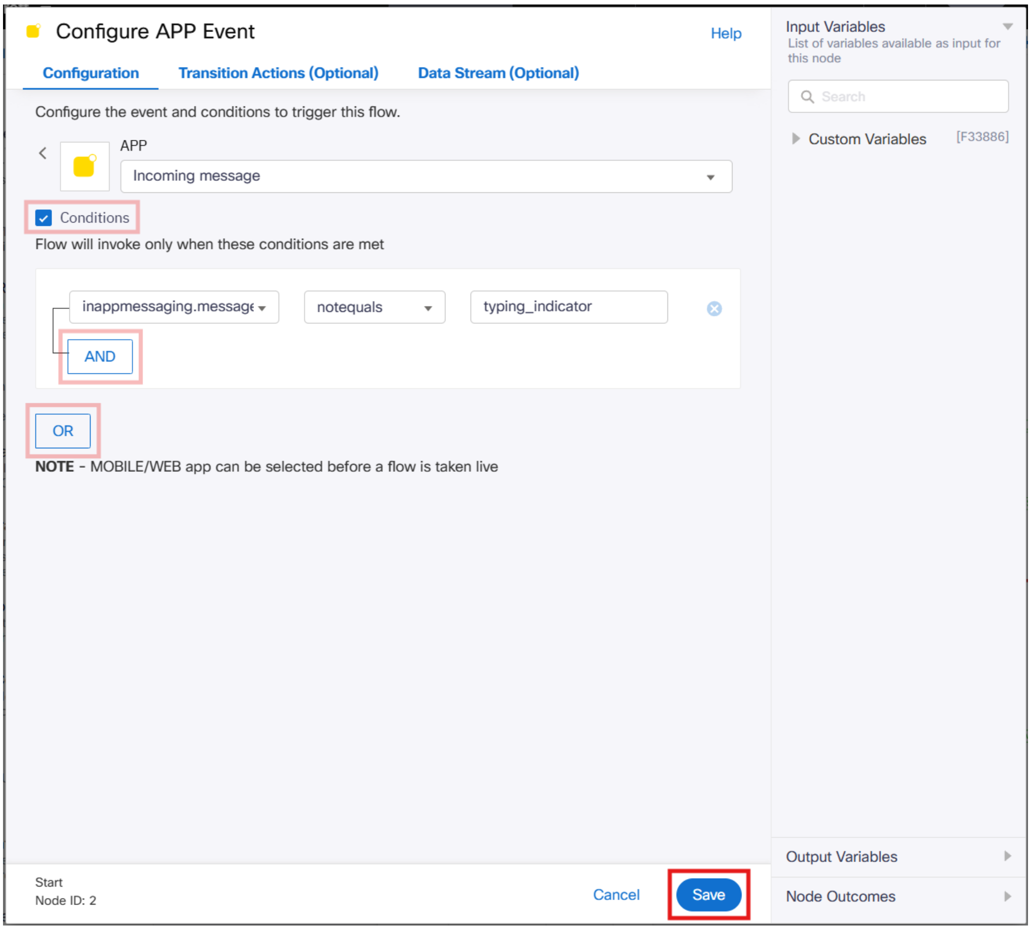
Name your new flow ‘PodXX Web Chat v1’ (remember to change XX to your pod number), select your method as Upload a flow, and then click Choose File to upload your file.



We’re going to use the 33865_webchat-v1.workflow file we downloaded from GitHub earlier. This contains a pre-configured flow where a lot of automated logic is already built, but we need to complete the work below before our demo can run. Select this file and then click Create.

The system will take a moment to load and then your flow canvas will load with our Start node (A node is a ‘step’ or ‘function’ within our flow.

These nodes can handle ‘triggers’ from several entry points, such as a webhook, an inbound email, an in-app profile registration or in our circumstance, an inbound web chat message. You’ll also notice Conditions are enabled, this is where we can determine logic to tell Webex Connect when you want this flow to be used or not, such as for an inbound keyword, or only trigger when an inbound API call contains a variable that doesn’t equal a certain value and so on.

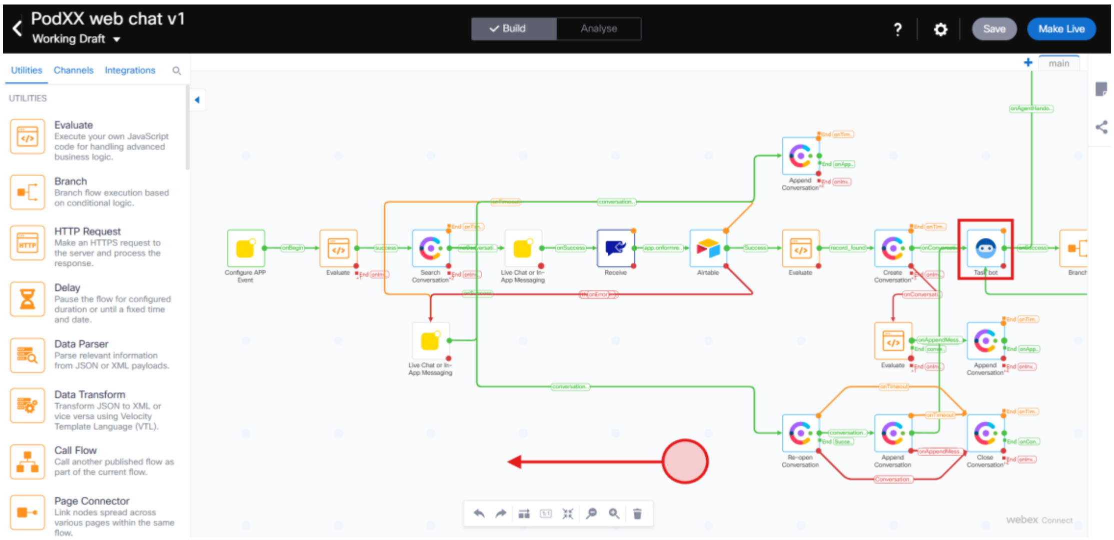


Let’s click Save.

Once the Start node is saved and closed, you will see the flow canvas, relatively similar to a canvas like Miro or Visio. We will explain the full flow and the purpose of the steps a little later in the module.

Zoom out a little bit, you can do this by scrolling on your mouse, pinching your mousepad, or using the '–' magnifying glass. Then move your flow to the left, you can do this by holding left click and dragging your cursor to the left.

We are looking for the Task Bot node, once located (see below for reference), double click on the node to open it.

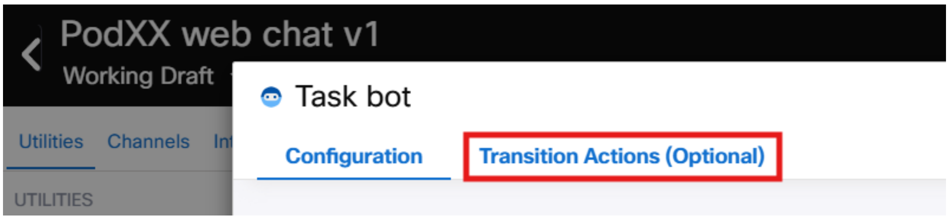


Note: Before we get into the configuration of nodes, it's good for you to understand that some nodes provide standard Output Variables by default, and we can also set Custom Variables from a node (we can also do this from the settings of the flow too – but we'll see that later in the guide). Custom Nodes are useful because we can set them from multiple locations and use them later in the flow.

All nodes other than the Start node can use input variables from other nodes that come earlier in the flow.

Finally, each of the nodes have Node Outcomes, such as a success or error path. For most of our demos, even pretty robust demos, we will mainly configure the success paths only, and some nodes will have multiple success paths. Some nodes will also require us to configure the timeout and error paths for flows to function correctly.

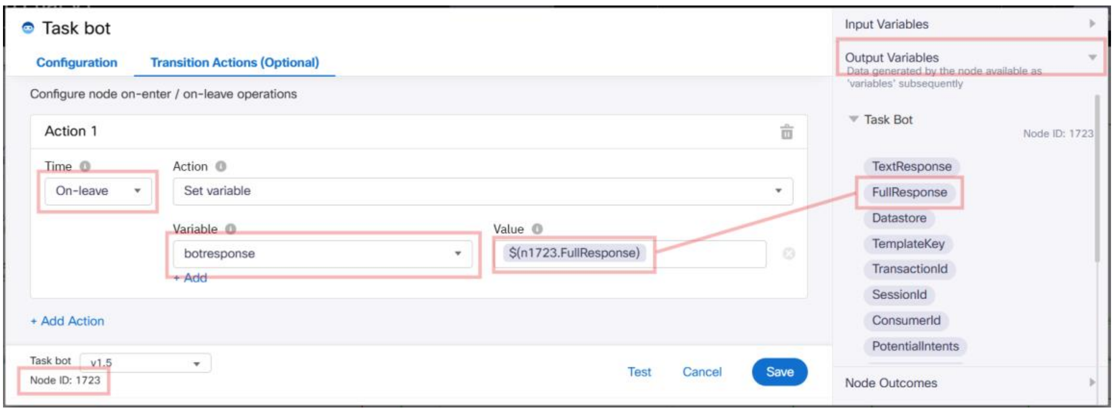
Let's quickly take note of what we are doing in the Transition Actions for this Task Bot node. Click on Transition Actions in the node top menu.



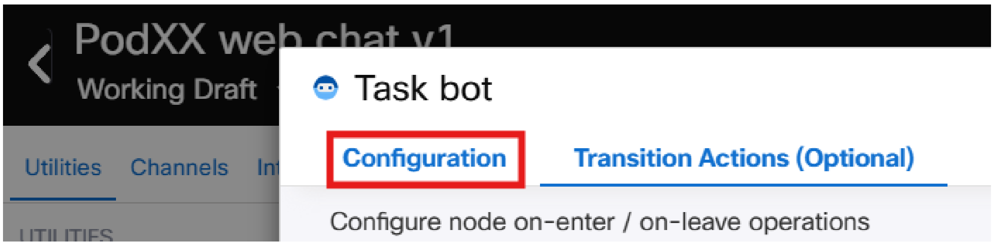
You will see that we are setting a variable, 'botresponse' with a value of $(n1723.FullResponse). Let's break this variable down quick:

> botresponse is a custom variable we are declaring. You can add variables using the right-hand Custom Variable dropdown in the Input Variables or on transition actions just by clicking Add and following the steps to declare a new variable. We will use this variable later in this module, and it is used in several steps throughout this flow.
>
> The n1723 denotes the node ID for when this variable is set, you can see the Node ID at the bottom this node (bottom left-hand corner) has the same ID, meaning that this node is what is generating the value for that variable.
>
> FullResponse is the variable identifier. To see the other variables that this node provides we can click on Output Variables in the right-hand menu.

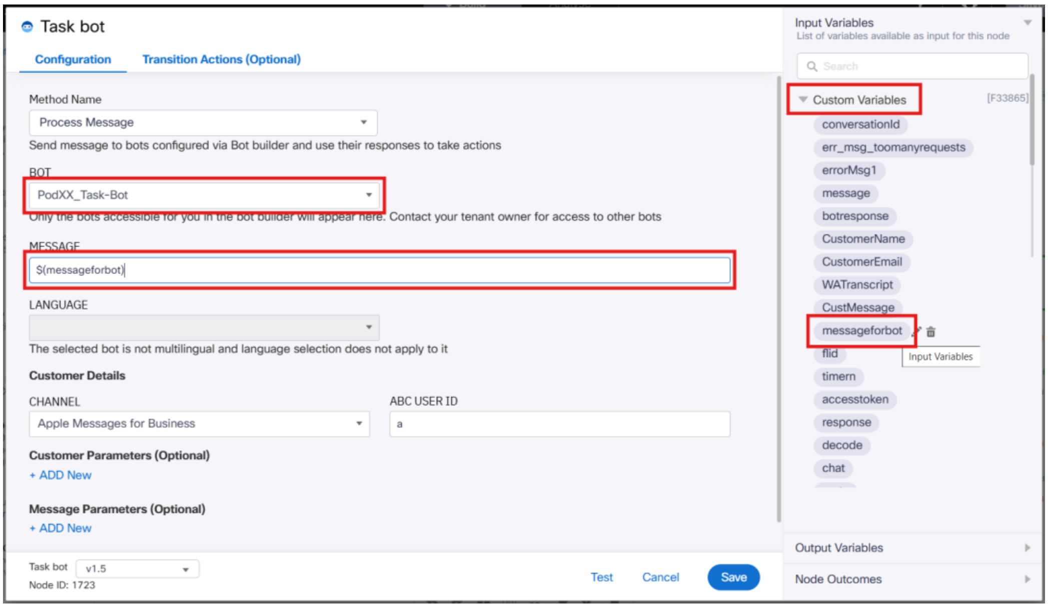
You'll also note that the Time is set to On-leave, meaning that the variable is declared after this node has executed and we have interacted with our Task Bot.



Now let's will setup this node to use the Bot we set up in 2.3. Click on Configuration in the top menu.

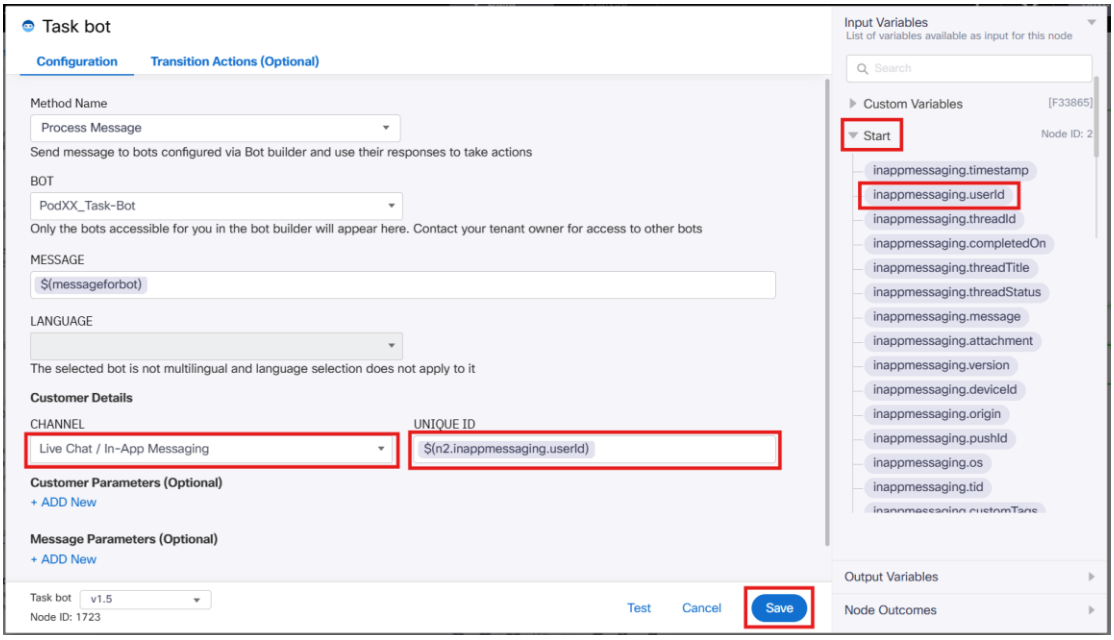


First, select your bot from the BOT dropdown.
Then in the MESSAGE field remove the character and using the right hand menu click on the Custom Variables dropdown*.
Select the 'messageforbot' variable, this will populate your field. Hit your space and then the backspace keys so that the field registers the character input.



Now click on the CHANNEL dropdown and select Live Chat / In-App Messaging. Remove the character from the UNIQUE ID field and in the right-hand menu, collapse the Custom Variables dropdown by clicking on the menu title or the dropdown icon.

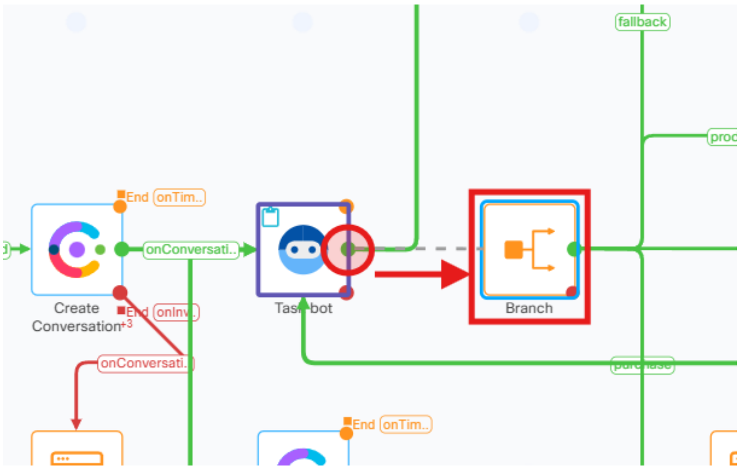
Now click on Start in the right hand Input Variables menu, and select the inappmessaging.userId variable, and again use the space and then the backspace keys.



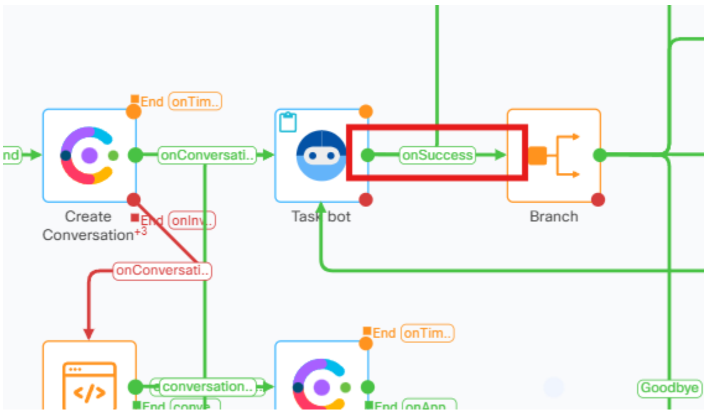
Now you can click Save.

Now the node is configured let's connect it to the Branch node to the right.

Click and hold on the green dot on the right of the node, now drag your cursor to the Branch node.

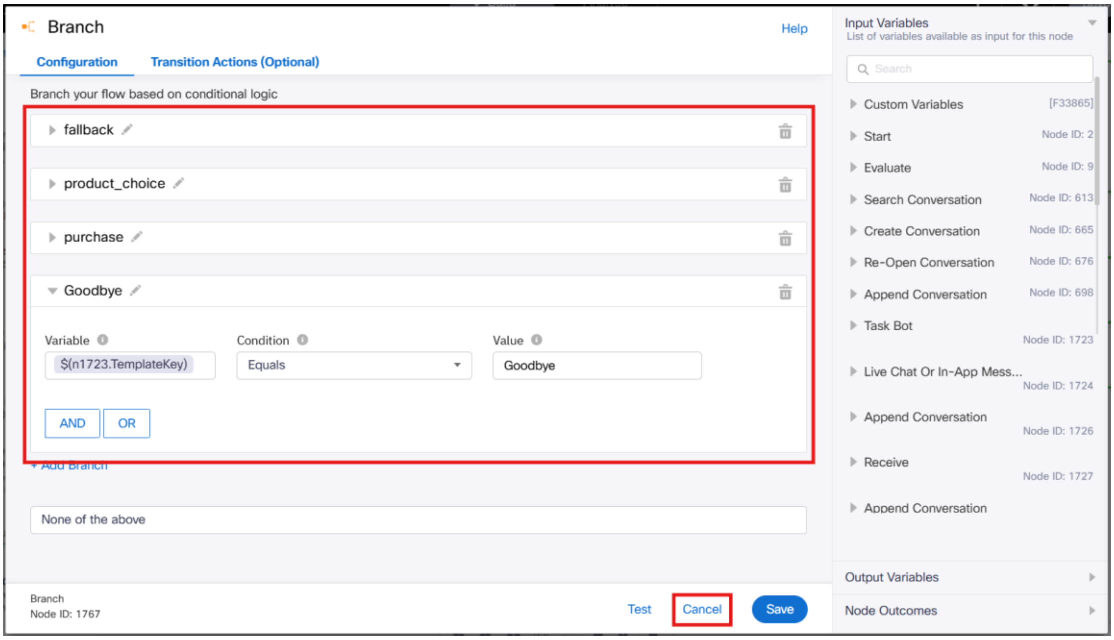


You'll then see a green onSuccess line that shows that when the Task bot node has executed successfully the next step in the flow will be the Branch node.

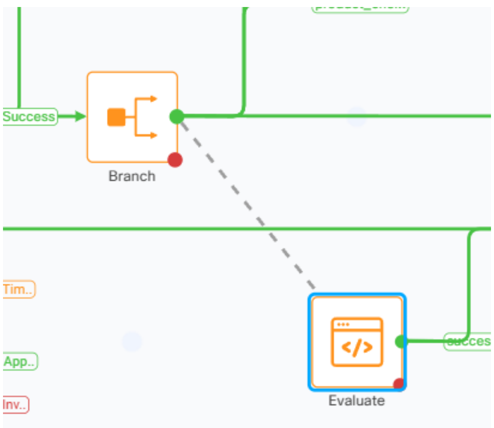


Now let's open the Branch node.

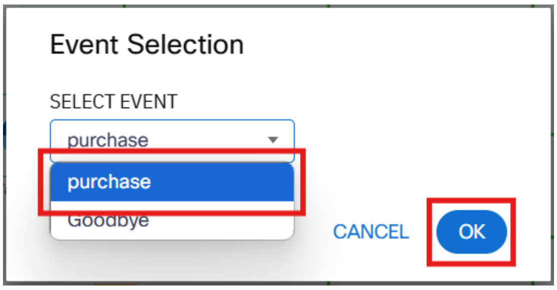
You'll notice there are 4 branches, fallback, product_choice, purchase and Goodbye, and each branch has conditional logic, e.g. the Goodbye branch is only used when $(n1723.TemplateKey) equals 'Goodbye'. To leave the node without making any changes click on Cancel.



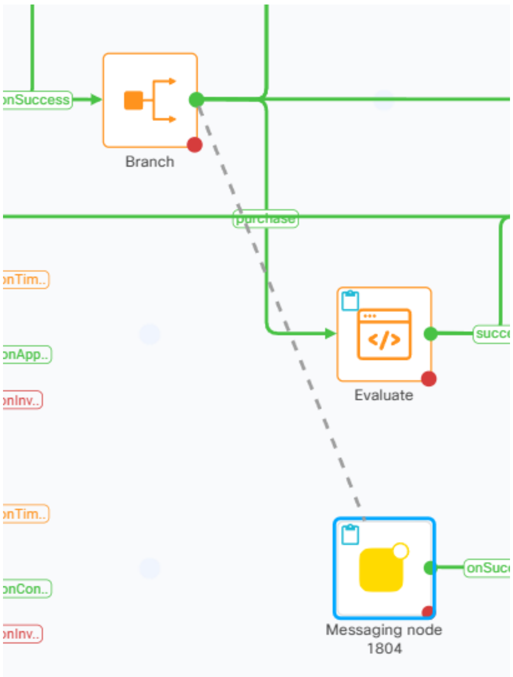
There are two branches that we want to use that are currently not connected to a node. Click on the green dot for the Branch node and drag it to the Evaluate node, which is just below and to the right.



Because there are several branches available, we get an Event Selection. In the Select Event dropdown ensure that the purchase option is selected and click OK.



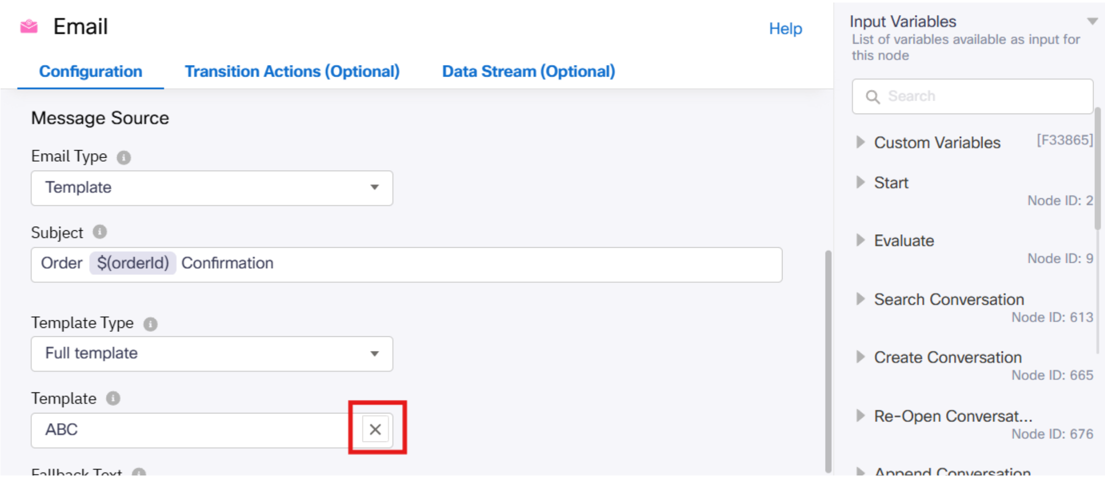
We're going to connect the last branch out of the Branch node to the Messaging node 1804 located below the Evaluate node.



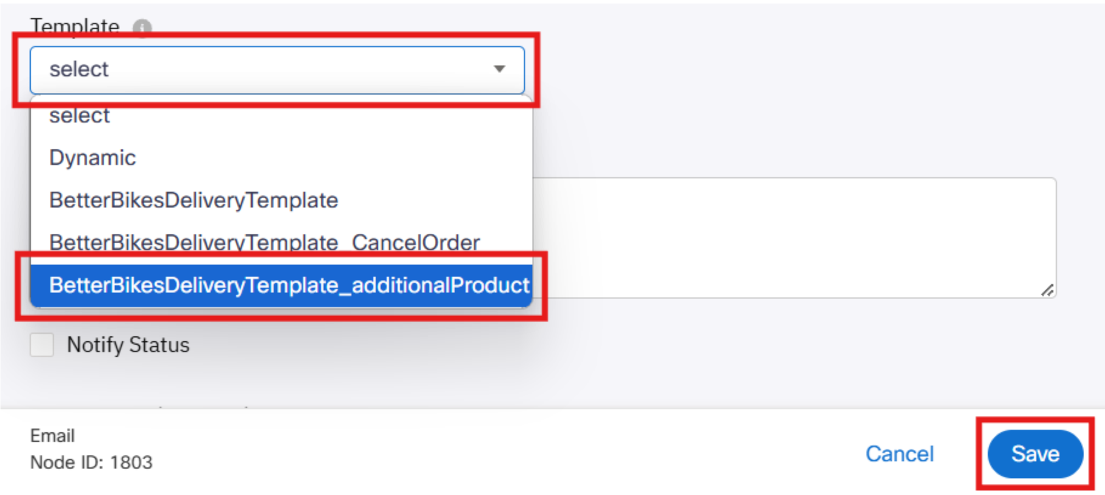
The Branch node is now configured with all branches utilized. This is a great node to highlight the low-code / no-code capabilities of Webex Connect, with simple and understandable conditional logic that doesn't require you to understand a coding language. We will however use the Evaluate node later, in Module 3, which allows us to use JavaScript.

To the right of the Evaluate node is an Airtable node that has no Success path configured. Connect this to the Email node to its right, then open the Email node.

Scroll down until you see the Message Source section, and under Template click on the X to the right of the 'ABC' input.

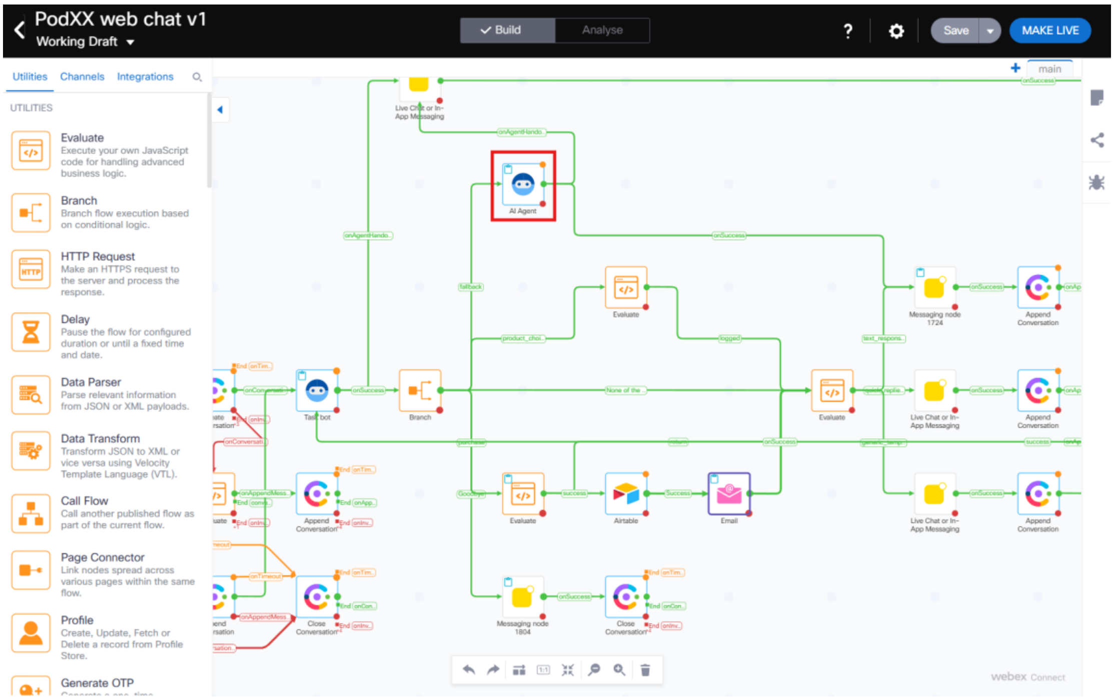


Click the dropdown and select the BetterBikesDeliveryTemplate_additionalProduct template, then click Save.



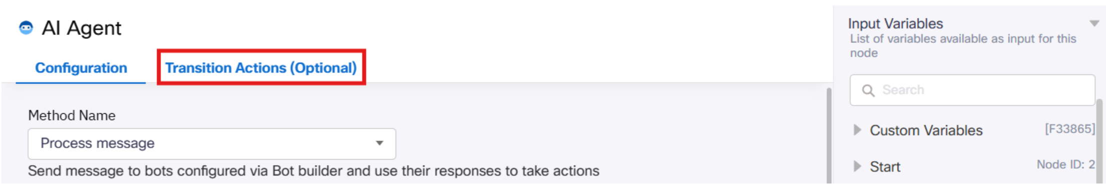
We will now receive templated HTML based emails when we 'place an order' in our demo.

In the next step we'll make use of the botresponse custom variable we used earlier, locate and open the AI Agent node (see screenshot below for reference)

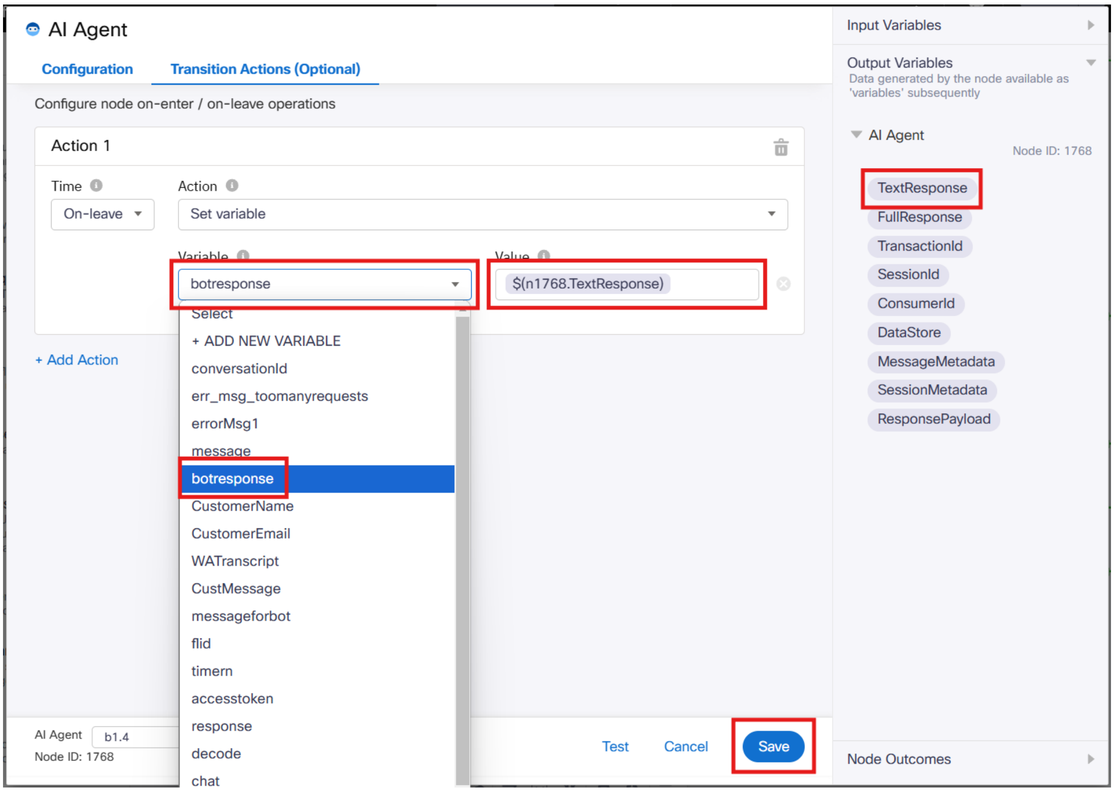


You'll notice that this AGENT TYPE is a gen AI powered, Autonomous. This bot will provide responses for interactions that the task bot does not understand, essentially what the task bot is not trained to handle, otherwise known as the task bot's Default fallback intent.

Let's head to Transition Actions.

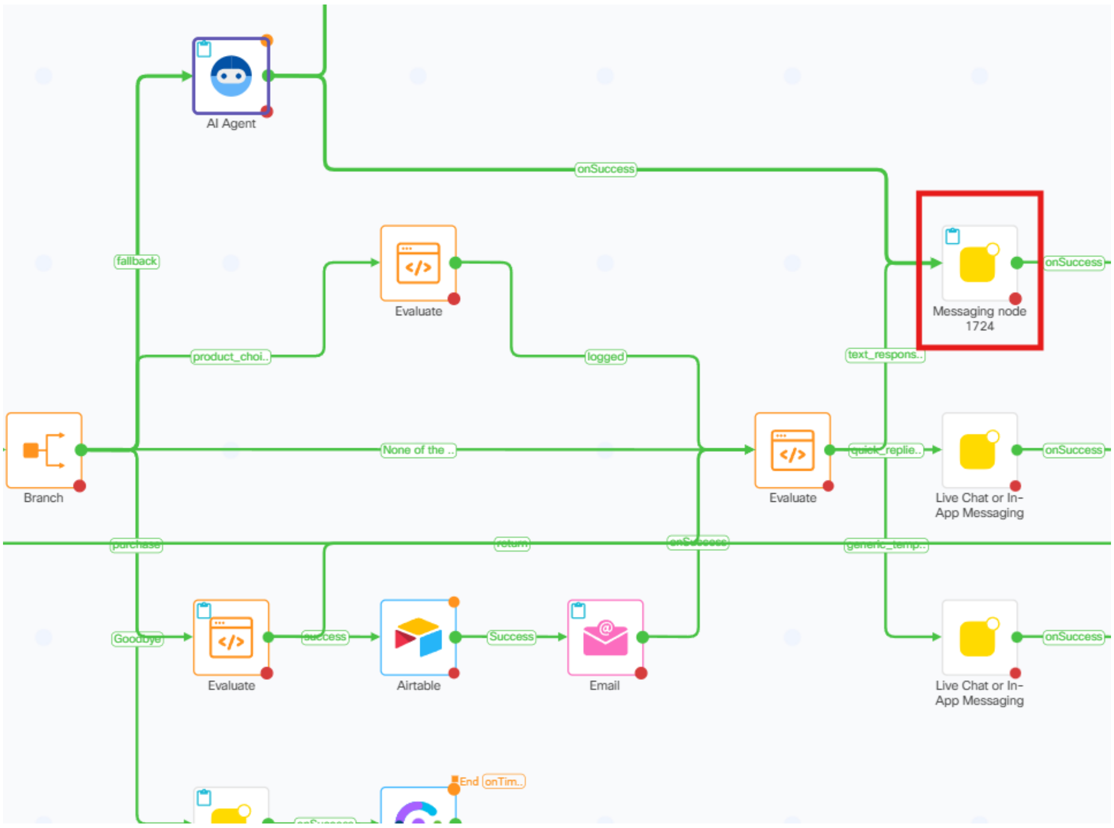


We need to set the botresponse to use the Autonomous bot's response. In the Variable dropdown select botresponse and set the Value to TextResponse from this node's Output Variables.



To finish, click Save

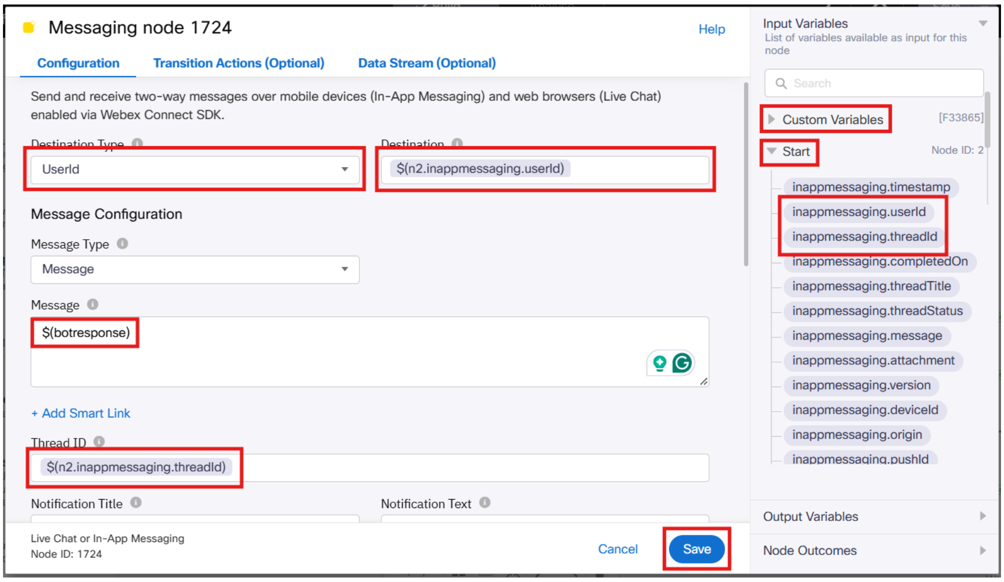
Locate and open Messaging node 1724 (see screenshot below for reference).



We'll configure this message node to ensure that if a text response from either the task or autonomous bot is needed than we can dynamically handle the message content.

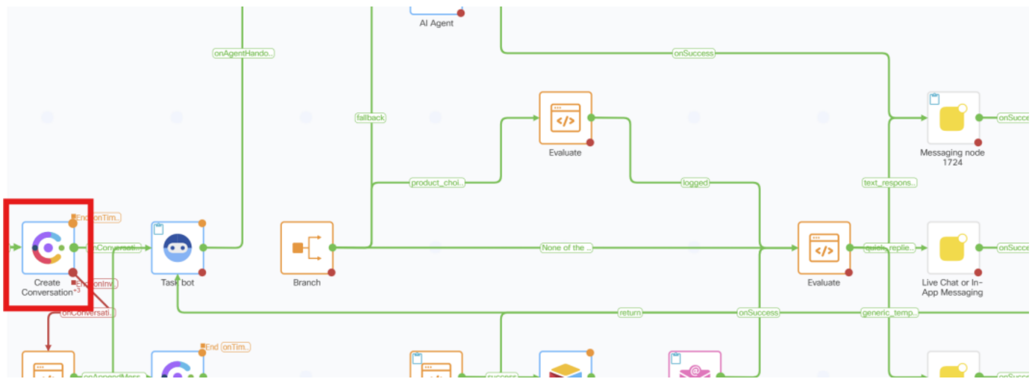
First set the Destination Type dropdown to UserId.
Change the Destination to the inappmessaging.userId from the Start node under Input Variables.

Then remove the ‘a’ from the Message field and select the botresponse from the Custom Variables, use the right-side menu again.
Then we need to replace the ‘a’ in the Thread ID field with the inappmessaging.threadId variable, using the Start node variables again.

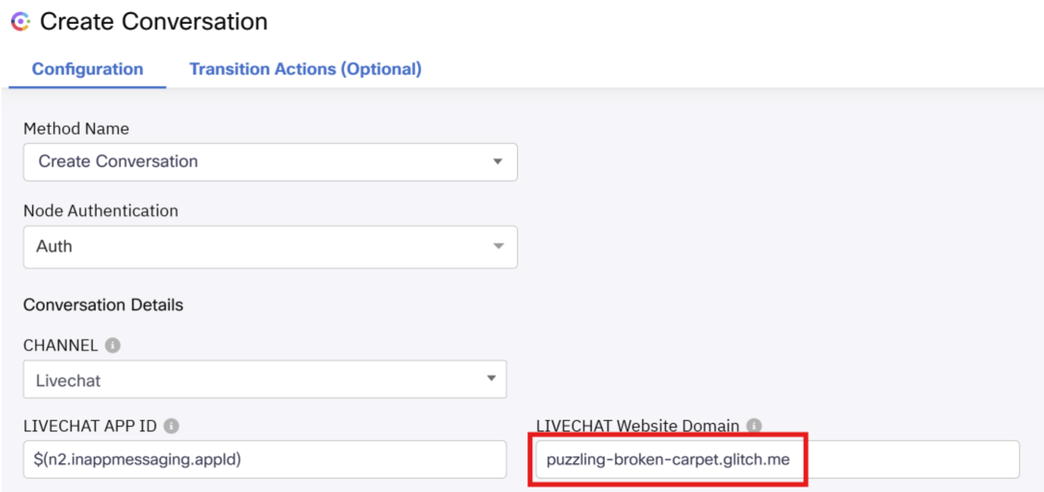


Now click Save.

The final two nodes we need to update will escalate our conversation to an agent when needed. Zoom out, locate and open the Create Conversation node (see screenshot below for reference).

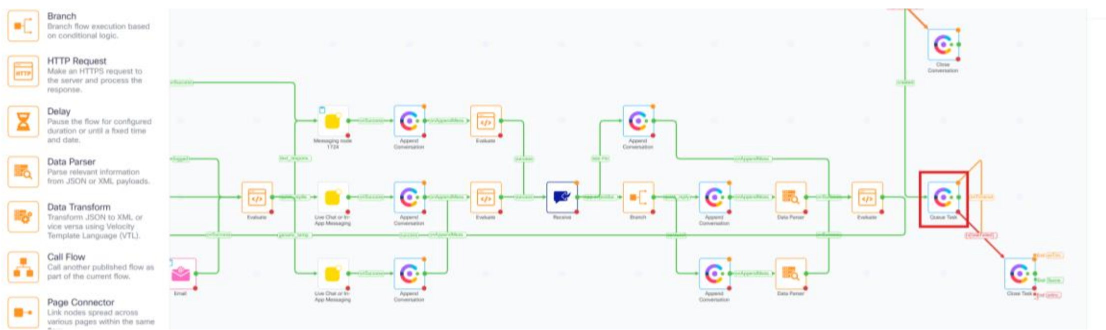


For this node we need to update the LIVECHAT Website Domain to the URL for our Glitch web page, removing the https:// at the start.

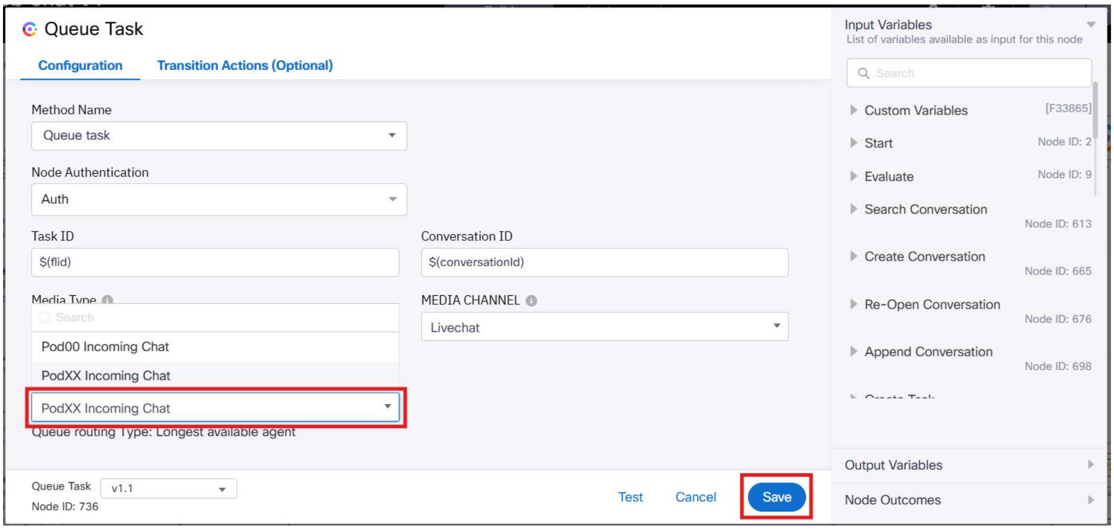


Now click Save to exit the node.

The last node we need to configure is the Queue Task node which is found on the right-side of this flow (see screenshot below for reference). Locate and open the node.

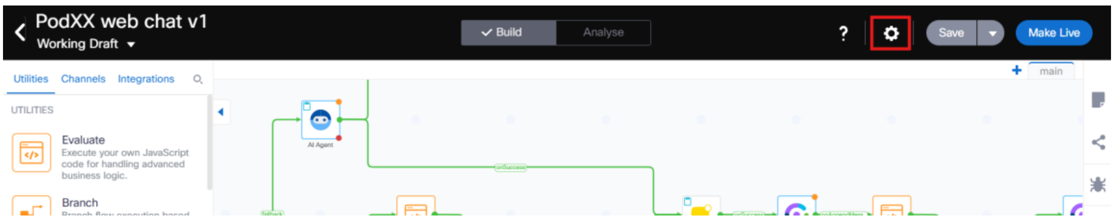


Click on the Queue Name dropdown and select the queue you configured in 2.2.



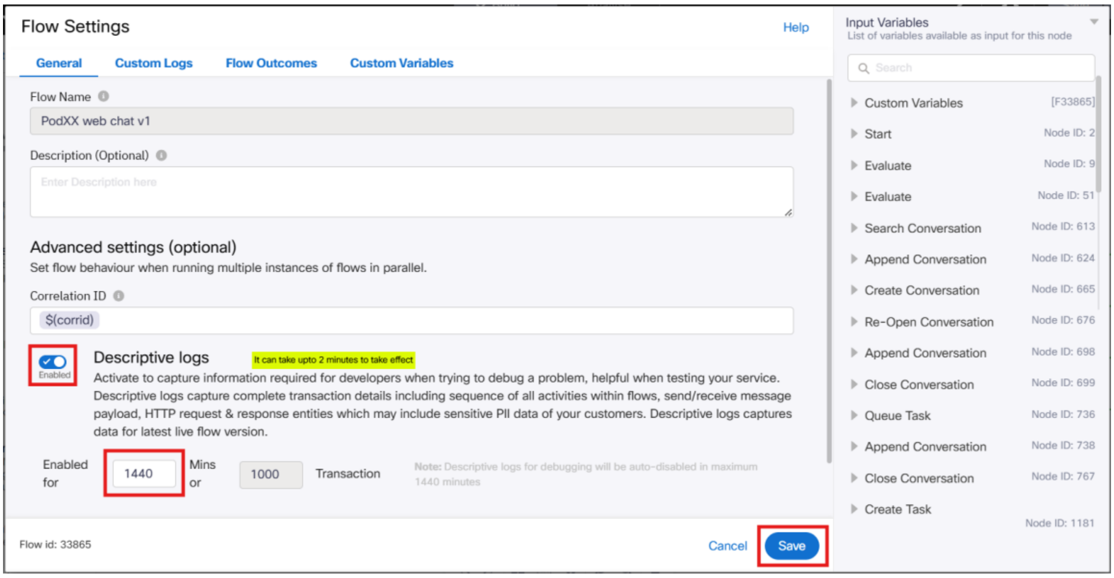
Click Save.

Now you've finished configuring all the steps for the flow, click on the settings cog in the top right of the UI.



Unfortunately, a commonly essential step when building demos is having to debug!

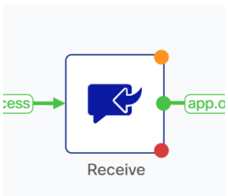
Toggle Descriptive logs so that they are Enabled and set Enabled for to 1440 minutes (this is the maximum time frame we can enable Descriptive Logs for at a time, however you can always re-enable this should you need to.



Click Save.

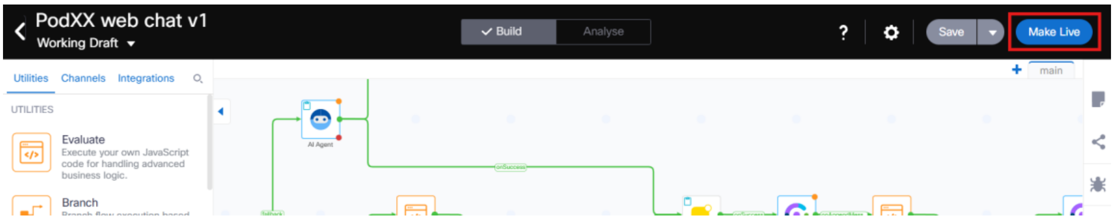
Note: You are now almost ready to publish the flow
When publishing any flows on Webex Connect that you have imported, you will need to open and save all receive nodes, so that the flow recognizes the nodes as configured before we can make the flow live.

To do this locate a Receive node in the flowbuilder UI (see example below)

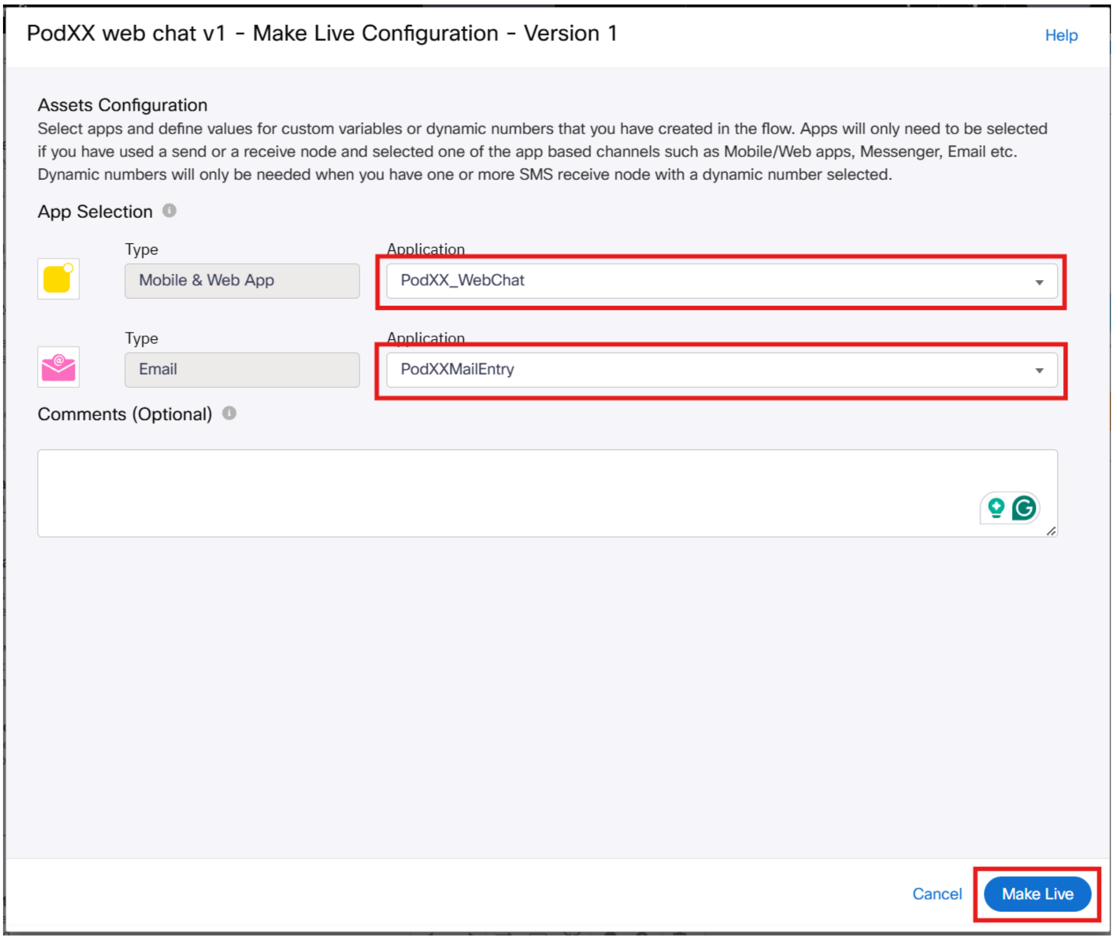


Double click on these to open the node, and then click Save to close the node again.

OK, let’s publish the flow. Click Make Live in the top right of your UI.



Now let’s select our PodXX_WebChat for the Mobile & Web App asset, then PodXXMailEntry for the Email asset (remember to replace the XX for your own pod number), and click Make Live.



Congratulations – you’ve built your first digital channels workflow! 🎉

Let's recap what you have configured:

- You enabled a Task Bot to drive logic and provide flexible, NLP based logic, which when coupled with the Branch node can handle several user intents with a variety of flow outcomes. A powerful combination of using AI and deterministic logic.

- You connected the Airtable database with our Email template to ensure a personalized Email, making the most of rich and dynamic, HTML-based emails.

- You then enhanced the AI capabilities of the flow, ensuring the generative AI based Autonomous agent was able to send messages using the same steps (nodes) as the task bot, reducing engineering overhead, streamlining processes and most importantly greatly improve the AI based capabilities of this bot powered flow.

Once the flow is Live you're ready to test your workflow.

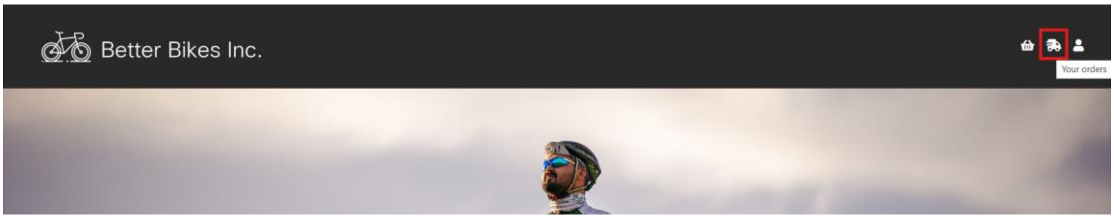
Note: You may need to refresh your browser if Live doesn't show after a minute.

2.4 Testing the Web Chat Demo

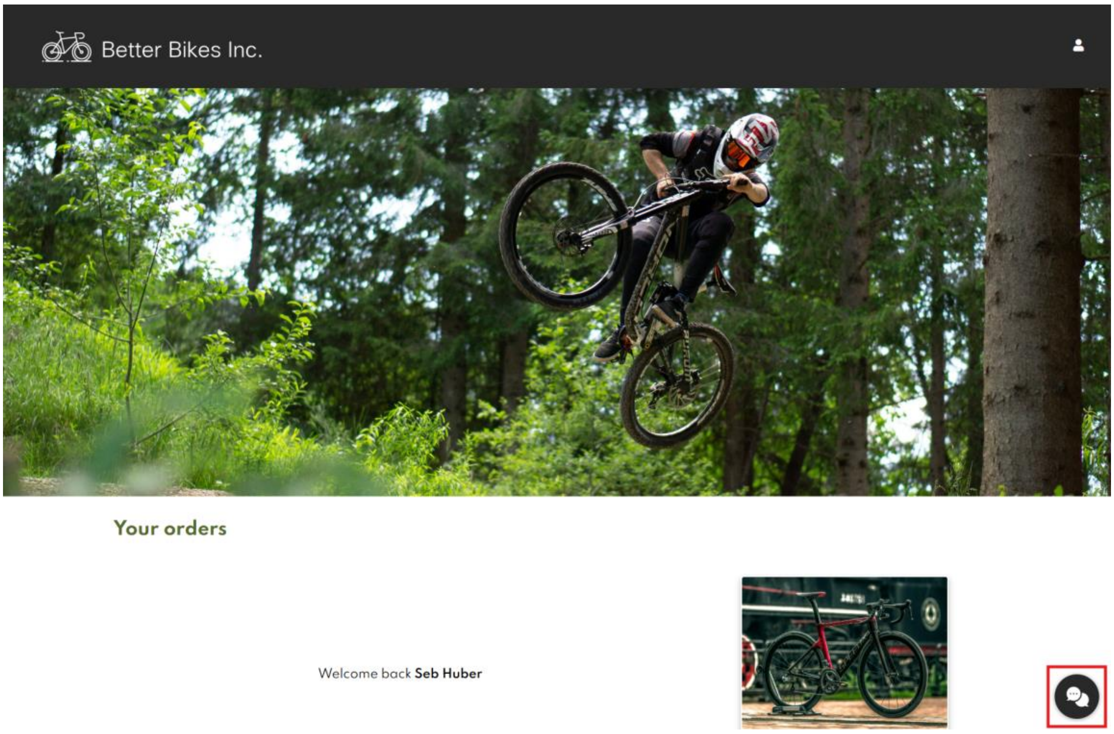
To test we'll use the Webex Contact Centre UI like we did in the last Module (1.12), you will likely need to log in again, this is to test our bot is configured correctly and able to escalate if needed.

Make sure your agent is set to Available.

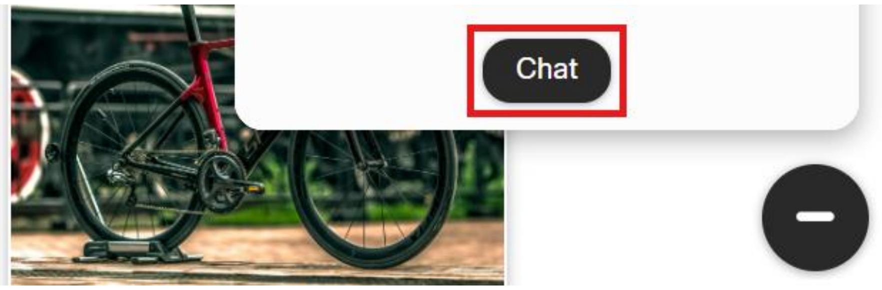
We also need to head back to our glitch website. Click on the delivery truck icon in the top-right of the website, navigating to the 'return.html' page we configured earlier in 2.2, with the web chat widget.



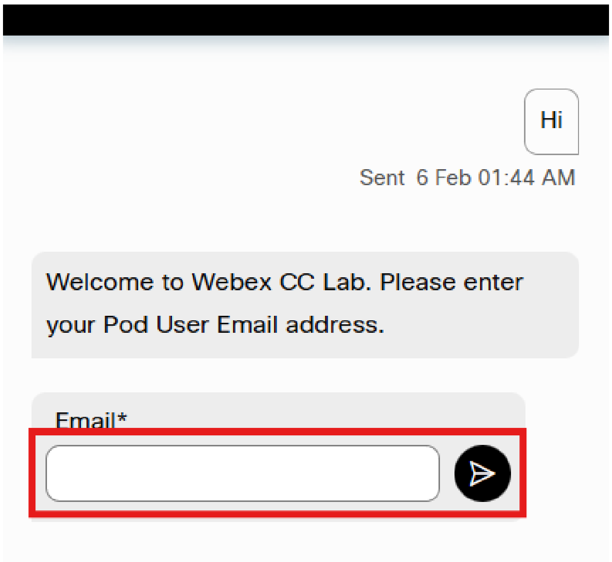
Now click on the floating web chat widget in the bottom right of the web page.



Click ‘Chat’.



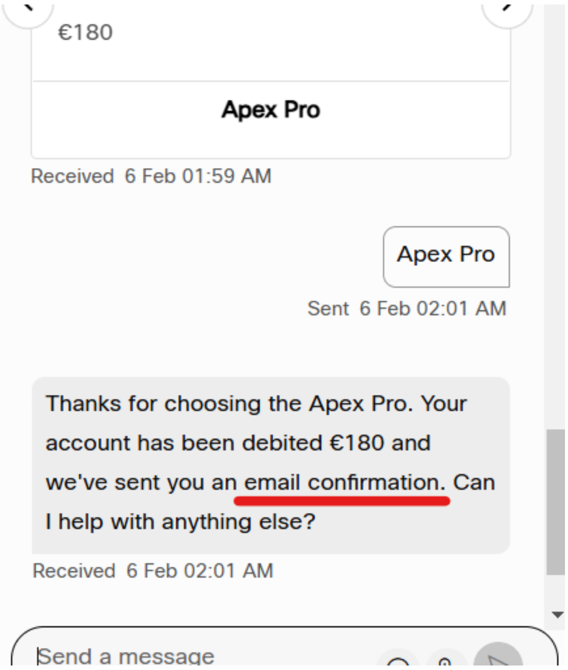
Now enter your email (your pod email address) and hit enter.



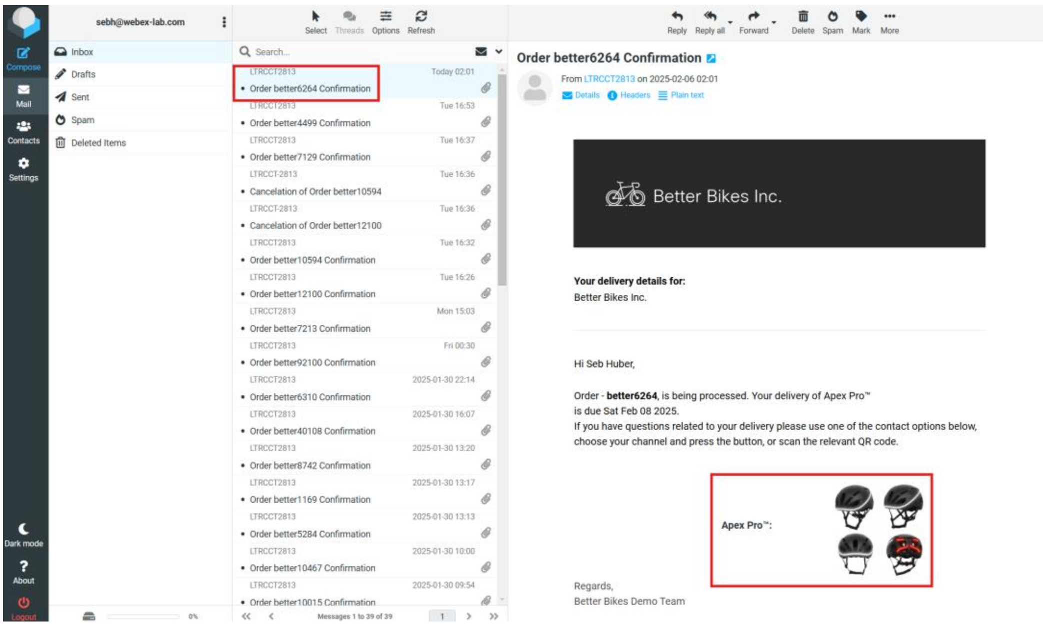
Now you’re verified you can ask the Bot questions about Better Bikes, the additional products, helmet, the founder, etc

The intent the bot is looking for to progress your demo further first is to place an order. We recommend you ask the bot something similar to “I’d like make an order”.

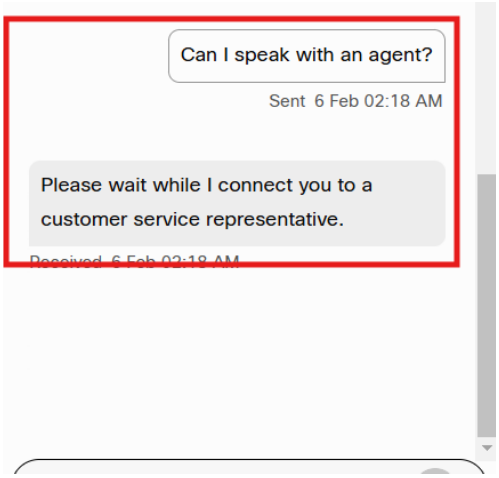
This will present you with a carousel of products that you can look through and select. When you select a product, it will confirm and send you an email.



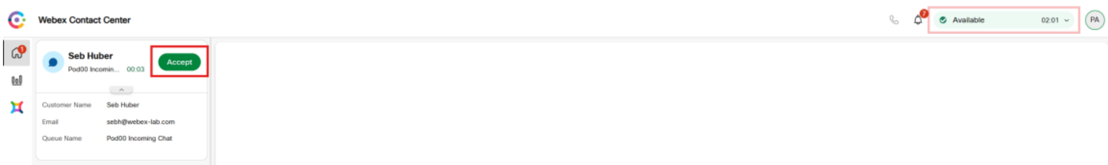
Switch over to the https://webmail.webex-lab.com/ tab and check your inbox – you should have a new order with the product you selected.



Head back to your web chat, the next intent you want to match is the agent handover, we recommend you ask something like "I'd like an agent please". A successful handover will look something like this:

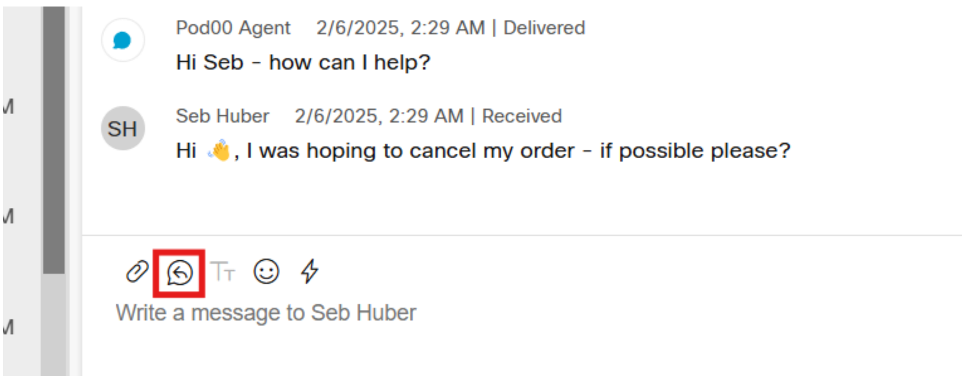


Head to the WxCC UI, where if you are Available, you should now have a chat waiting to be accepted. Click Accept, and the chat will load in your UI.

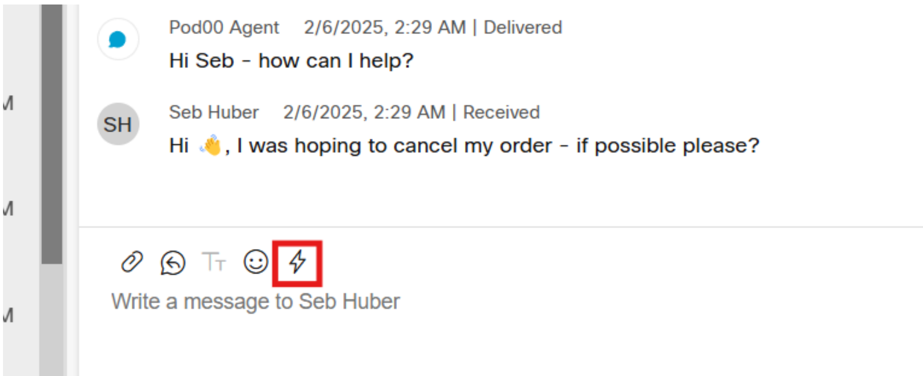


Once it's loaded you can message back and forth.

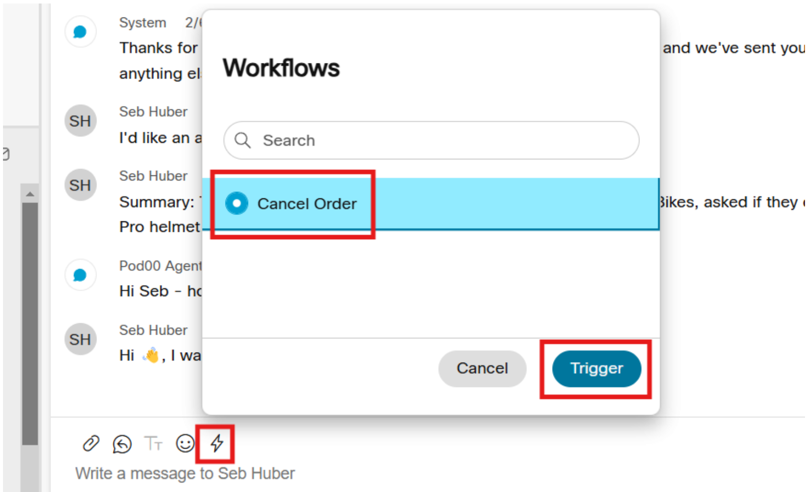
To use a template, click on the reply icon (see screenshot below for reference)



You can also cancel order using the Trigger Workflow. Click on the lightning icon (see below screenshot for reference)



Then select Cancel Order and Trigger. This is a pre-configured rule that uses the 'Website Order Cancelation' workflow, found in the Global Default Flows on Webex Connect.



if you now check your email inbox, you will also see that you have a cancelation email and that will delete an order record in the Airtable database.

When you're done with your chat, remember to End the chat and submit your wrap-up.

Congratulations - you've finished all the standard elements in this Module! 🎉

There are two additional module elements, but we recommend completing these after you have finished the standard lab in Module 3.

Skip to Module 3 **HERE**

2.5 Addon: Ongoing bot curation

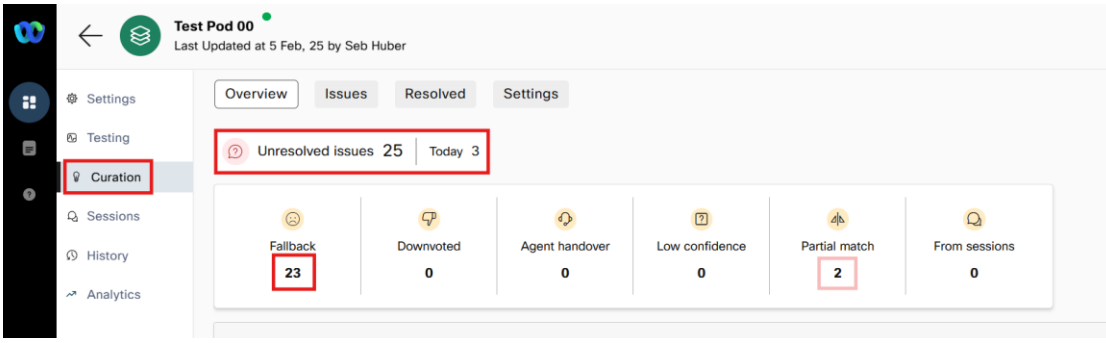
Please note, this is additional content.
This task can be skipped, but you will learn how to implement database connections for receiving more information about the caller. We recommend completing Addon modules once you have reached the end of your 'standard' lab if you have leftover time.

Before you curate your bot, run through the Web Chat demo a few times, try asking the same questions in varying methods. By interacting with our bot, time and time again, we are generating the data for our bot to be fine-tuned with, so hopefully the bot will struggle to understand some of the questions you ask.

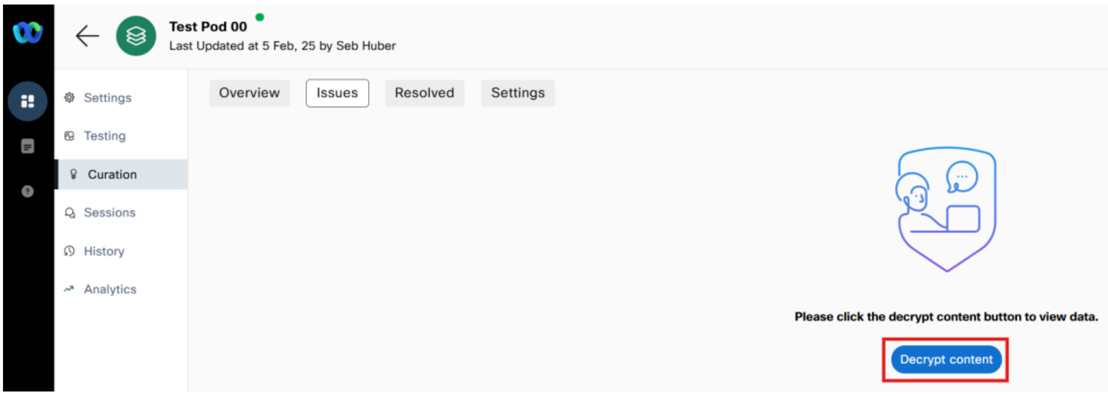
Once that's done let's head back to our bot, in Webex Connect (hover on the App Tray, bottom left-hand menu, and click on AI Agent Studio) or navigate to the relevant tab if you have still this open.

From the dashboard select your Pod corresponding bot that you configured in 2.3.

In the left-side menu head to Curation. We land on the Overview page, if we have successfully challenged the bot enough, then we should see some numbers against Fallback and possibly even Partial match. Let's click on the Fallback segment.



Now Decrypt Content.



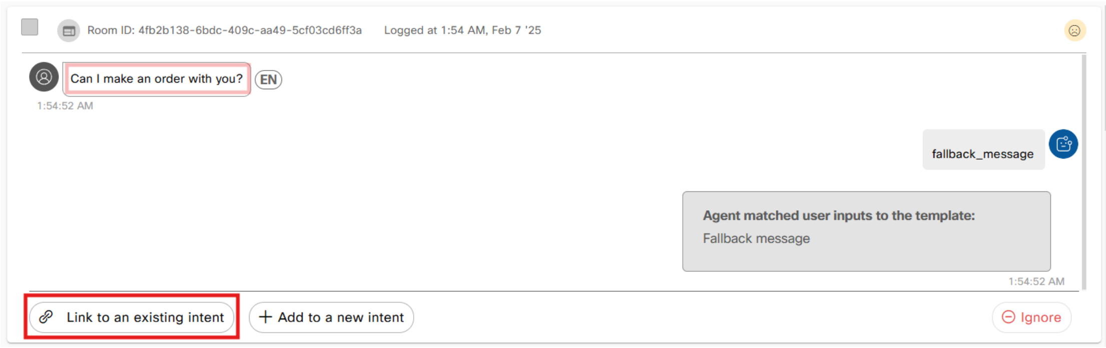
Now we have a list of Issues ready for us to resolve and help train our bot. Let's go through them one-by-one and either Link to an existing intent or Ignore them. Our advise is that if you are unsure if a message matches your intent, in other words, it was too different from the intent you think it might match, then you are better to be cautious and choose to ignore.

You can also select multiple Issues at once and choose either action to link with existing intents or ignore them. For this lab, let's review them one-by-one (feel free to explore handling multiple issues after).

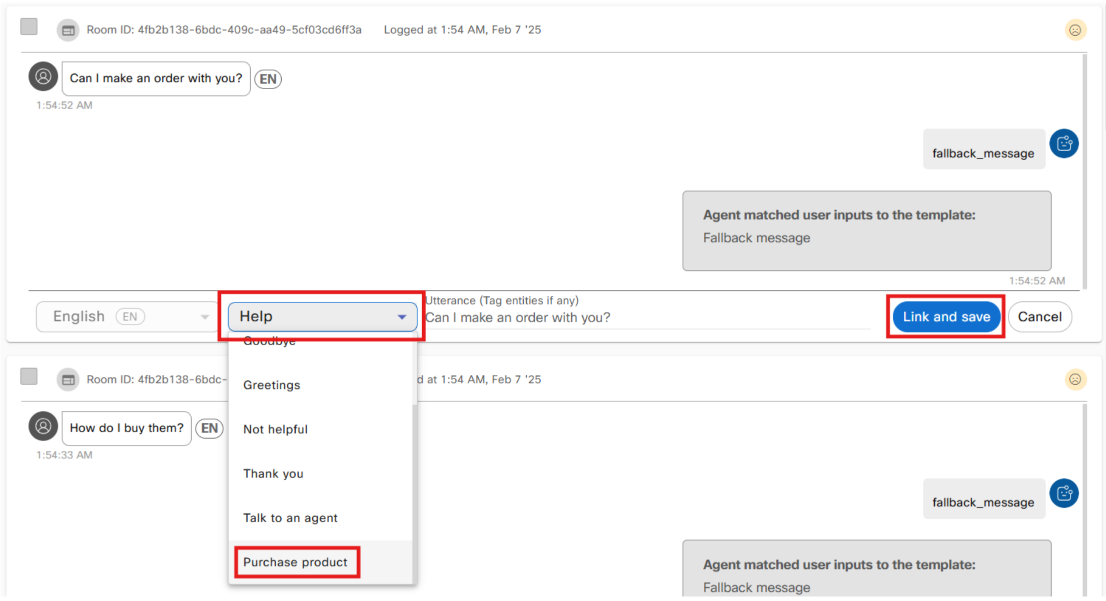
Link to an existing intent
To link the message with existing intents, review the messages on the left and determine does this seem to match with an existing intent. Either the intent we added (to make a purchase) or a default intent, such as greetings or the help intent (which we have set to the fallback template key).

To do this see our example below.

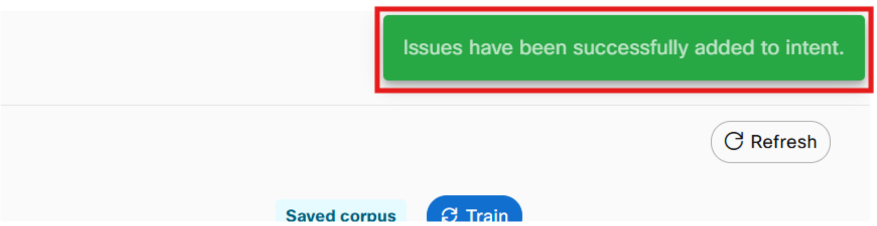
This message, “Can I make an order with you?” looks like it should match our Purchase product intent so we’ll click Link to an existing intent.



Now we’ll choose the intent Purchase product to link this too and save.



You’ll see a success message pop up.

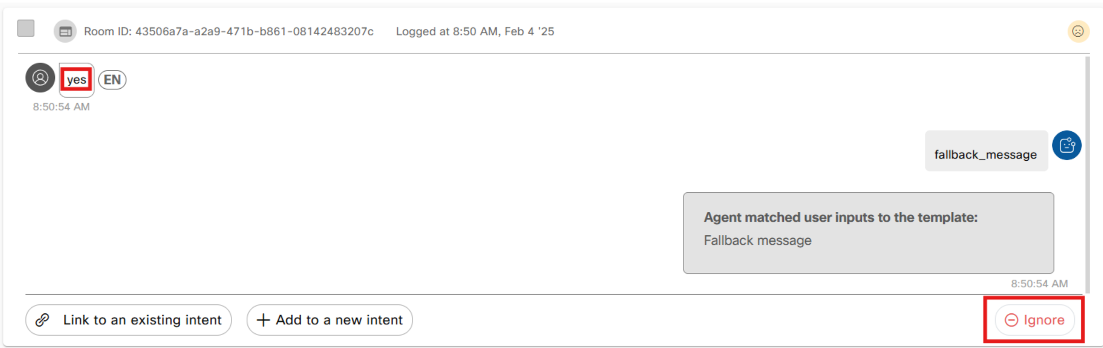


We’ll also now be able to see the linked message in the resolved tab, let’s first ignore an issue and then head to Resolved.

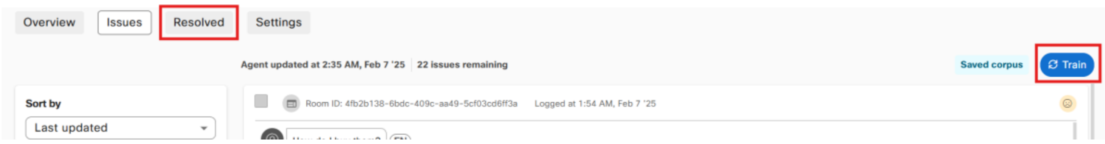
Note: If we noticed that an order contained an entity (we have products listed as entities in our example), than we can tag them here, however it is also worth noting that our demos are not configured to handle this, but can be incorporated into your own future demo designs and is a great highlight of how we can use AI to help steer our customer experience journeys.

Ignore

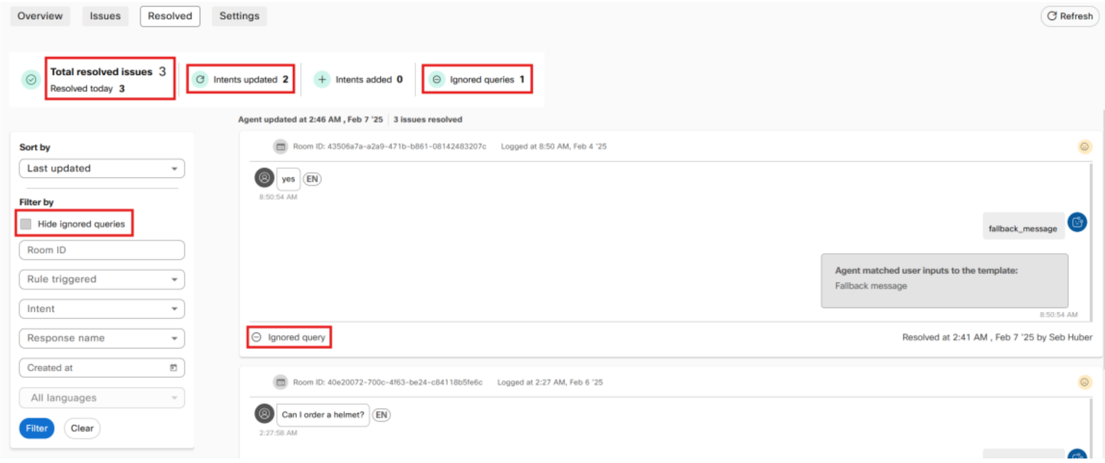
Find an article you think doesn’t need matching and let’s click to ignore it. See our example below, containing the message, “yes”.



You will receive a different success message pop up. Now that we have started to resolve a few issues we click Train, add any comment (these are handy for colleagues working in teams or reflective work) and then head to Resolved to see the issues we’ve solved so far.



Here we can see the resolved issues, including metrics, a filtered view and details of what action has been taken for each message.



2.6 Addon: Flow enhancements (verification loop and gen AI summaries)

Please note, this is additional content.
This task can be skipped, but you will learn how to implement database connections for receiving more information about the caller. We recommend completing Addon modules once you have reached the end of your 'standard' lab if you have leftover time.

For this flow let's head back to your Service on Webex Connect.

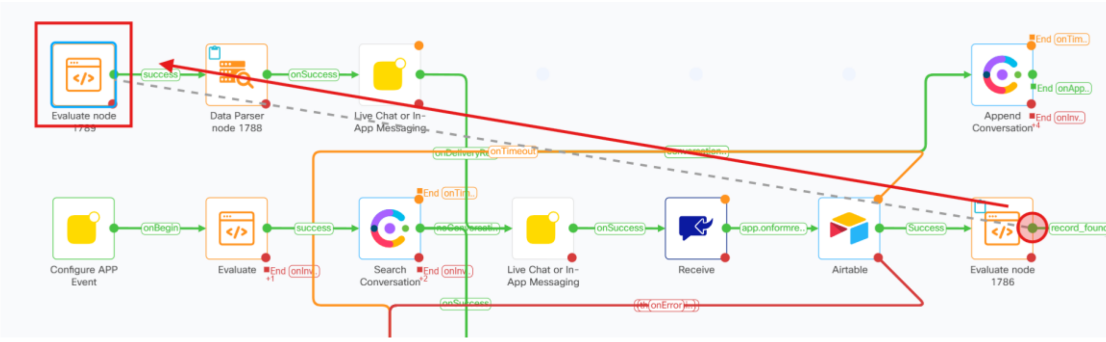
First, we will need to disable the flow you built in 2.4 (PodXX web chat v1). Simply toggle the State so the flow is disabled.

We're going to Create a new flow using the Upload method again, and this time we will use the 33864_webchat-v2.workflow file we saved from GitHub earlier. Name your flow 'PodXX Web Chat v2' (remember to change XX to your pod number), and create your flow.

Zoom out and you will notice that we have added a few nodes extra from our v1 flow. The extra nodes already configured will add CJDS to our flow, so when the chat hands over we will get better insights to the customer's journey.

We're going to add email verification for the form when you first initiated the web chat, and we'll add an AI generated summary of the chat for WxCC.

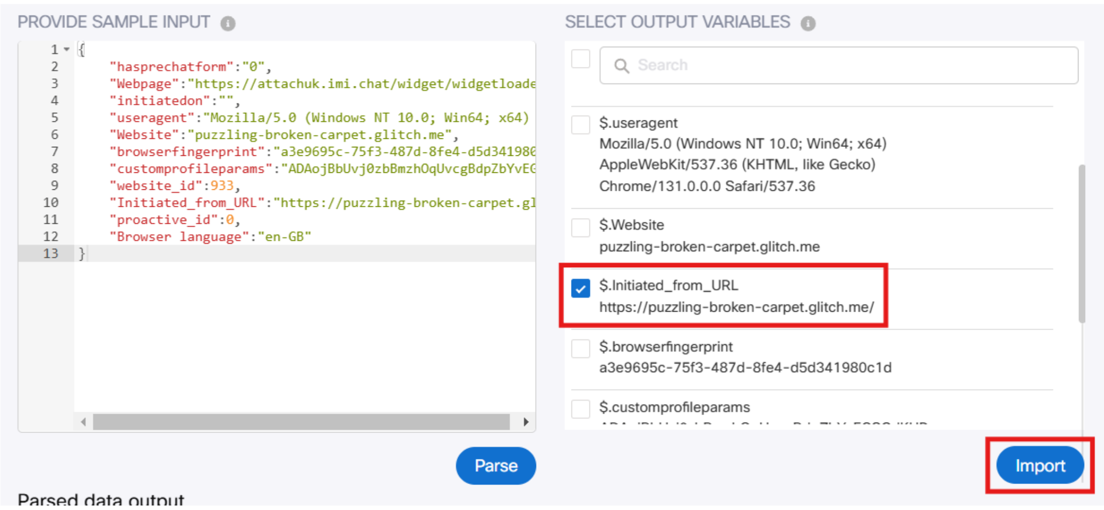
Zoom in again and focus on the first part of the flow. The first thing we will do is connect the no_record path for Evaluate node 1786 to Evaluate node 1789.



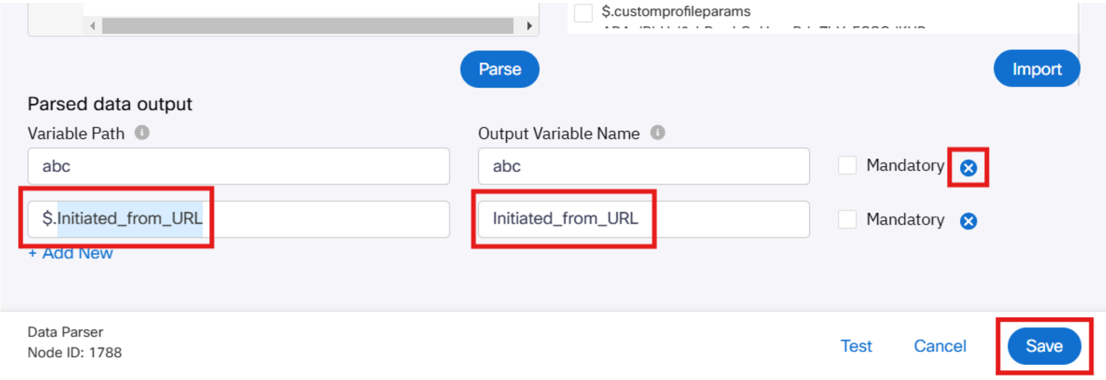
This will loop round and ask the customer to try again if we have no found records for the email address configured (to test this we will run the demo again, replying to the form with an email address we haven't added to the database).

Now let's open Data Parser node 1788 to the right of Evaluate node 1789, the node we just connected to. You will see that a sample JSON payload has already been saved and parsed.

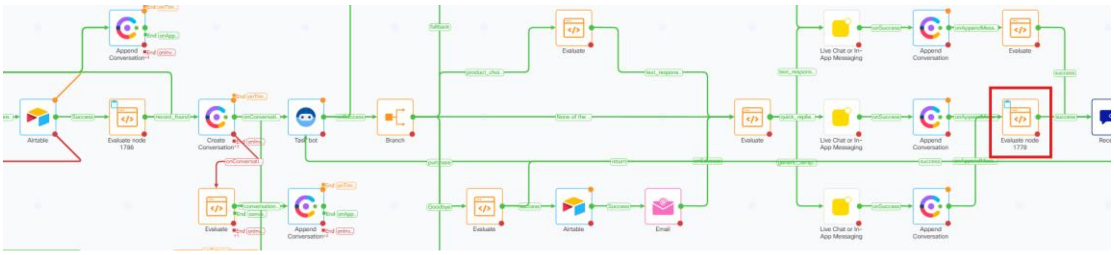
In the SELECT OUTPUT VARIABLES section search for and select the Initiated_from_URL variable.
Then click Import.



This will populate a Variable Path below, that we just need to provide an Output Variable Name for, for ease, double click on the highlighted part of the Variable Path (see screenshot below), copy and paste that into the Output Variable Name field. Finally let's delete the 'abc' variable above click Save to exit the node.



That's the email input verification loop done, now let's head to Evaluate node 1778 where we will build a transcript that we can later summarize.

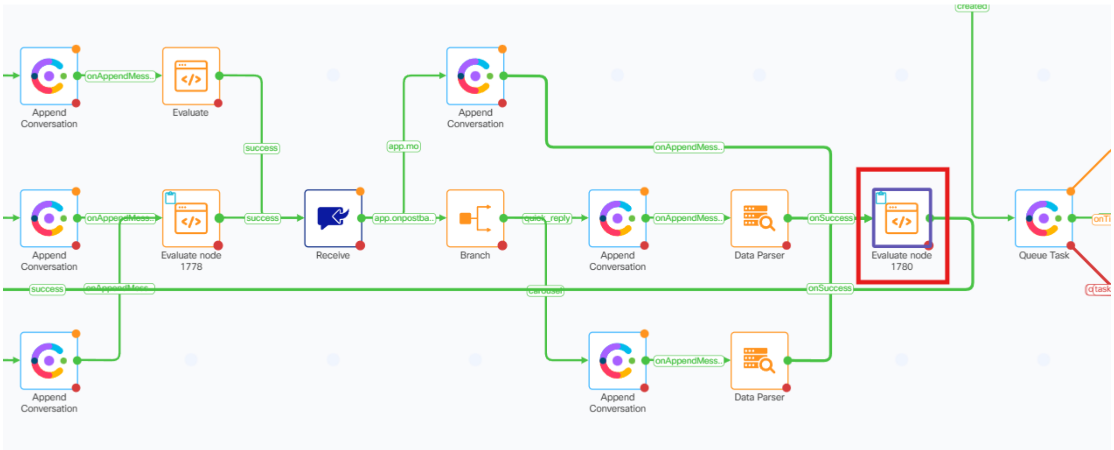


Copy and paste this line of JavaScript before the '1;':

chat = chat +"Agent: " + text + " ";

```
chat = chat +"Agent: " + text + " ";
1;
```

Now let's save our node. This provides our transcript with context from the bot. Now open Evaluate node 1780.

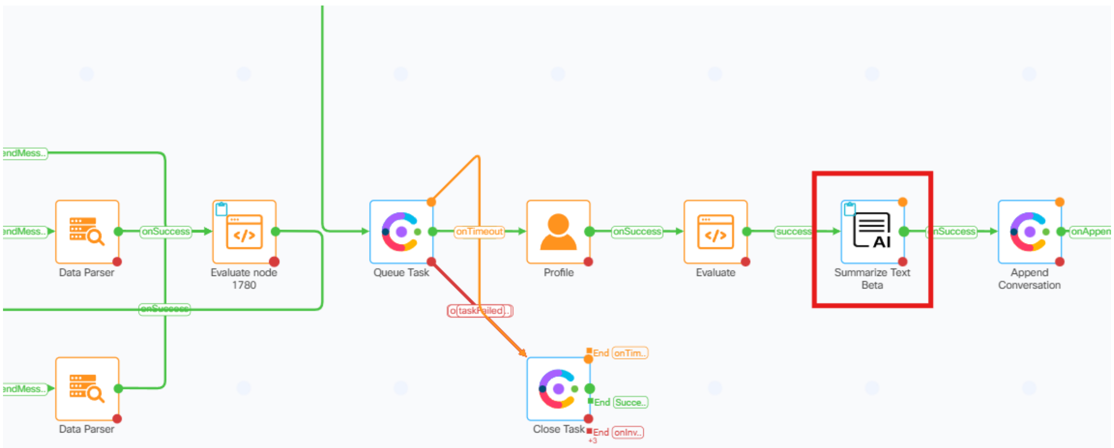


Copy and paste this line of JavaScript before the '1;':

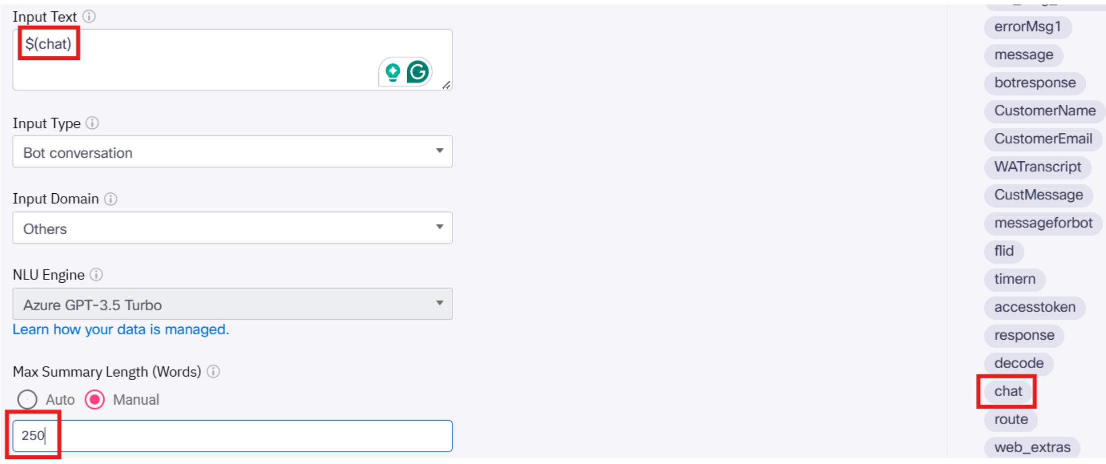
chat = chat +" Customer: " +messageforbot + " ";

```
chat = chat +" Customer: " +messageforbot + " ";
1;
```

Now let's save our node. This provides our transcript with context from the user (the customer), meaning we now have context from both the bot and user, time to summarize. Locate and open the Summarize Text Beta node.

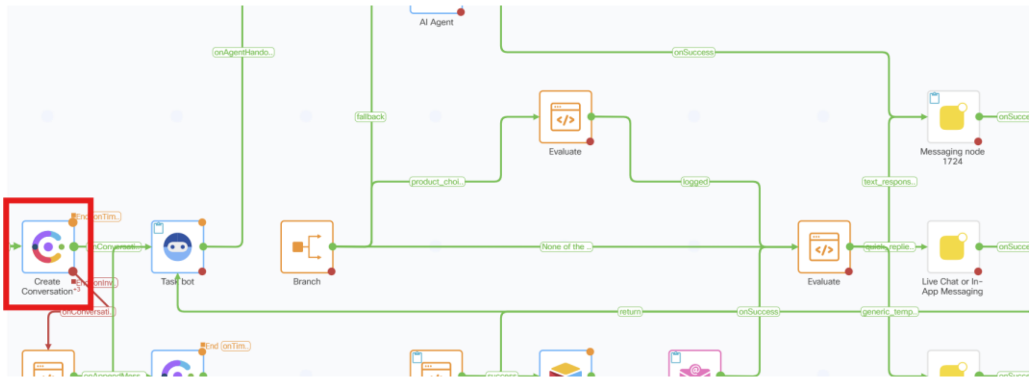


Change ‘abc’ in the Input Text field to the chat variable from Custom Variables. Change the Max Summary Length field to 250. This will set a limit to how long the summary can be. Click Save.

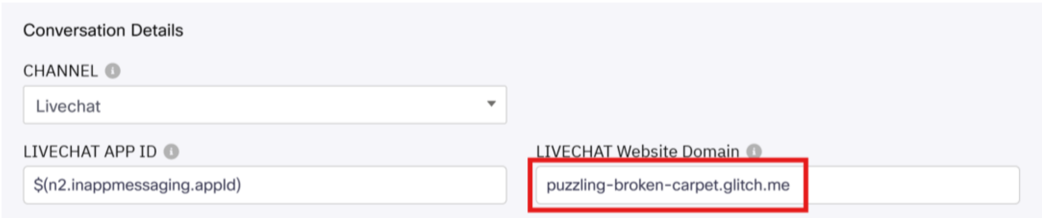


Now your transcript is also ready.

Like we did in Module 2.3 we need to update the LIVECHAT Website Domain so that we can escalate our conversations to an agent. Zoom out, locate and open the Create Conversation node (see screenshot below for reference).



For this node we need to update the LIVECHAT Website Domain to the URL for our Glitch web page again, removing the https:// at the start.

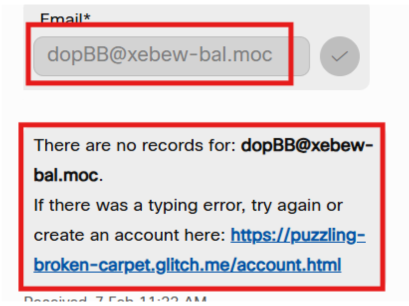


Now click Save to exit the node.

Let's publish the flow, click Make Live and set the asset to the same you used in 2.4. Once the flow is Live you're ready to test your workflow.

Note: You may need to refresh your browser if Live doesn't show after a minute.

Please note, for testing:
Run through your web chat demo using an incorrect email input, a successful test will look like this:



Now input your correct email address and after an interaction with the bot, ask for an agent
----

Module 3 – Digital Channels

For this lab we will configure a WhatsApp flow designed to allow users to rearrange their delivery in a self-serve, automated fashion. This lab will teach you how:

- to use JavaScript value generation as a functional demo asset
- create restricted, sandbox environments for an individual or team
- create engaging, rich messages with convenient tap-functionality
- create dynamic API calls and parse the responses
- use custom integrations, simplified for workflow usage
- create reporting outcomes for varying success paths
- use rich WhatsApp templates

In the Addon tasks we will enhance this customer journey further by adding the option to escalate conversations to agents in WxCC with additional, deterministic based insights.

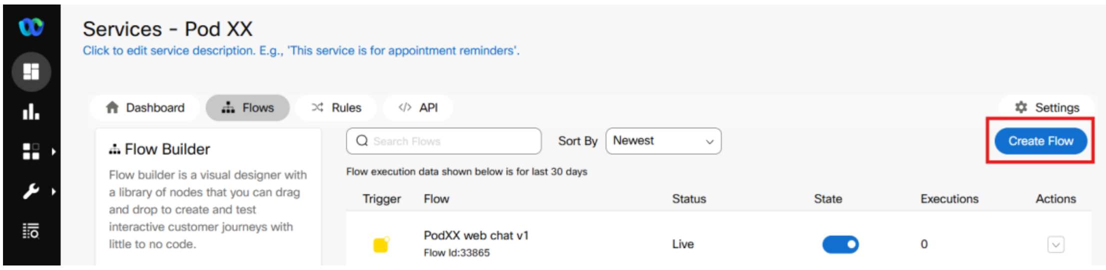
Before we get started with this lab we will need to grab some pre-configured items from the Module 3 folder in GitHub. Head to https://github.com/ChrisW0lf/CL-EMEA25-LTRCCT-2813/tree/main/flows/WebexConnect/Module3 and download the following files:

- 33894_whatsapp-v1.workflow
- 33949_whatsapp-v2.workflow
- deliverydate_module3.js

3.1 Webex Connect: WhatsApp Workflow

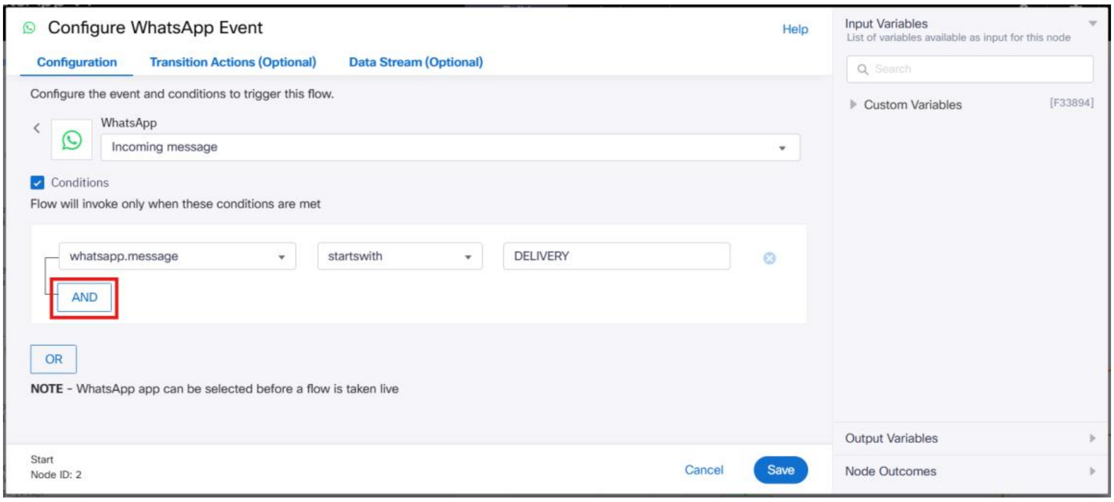
Head back up to your Service in Webex Connect. You can navigate back to a service using the back icon in the top left of the UI or follow the steps from the start of Module 2 if you need to log back in (remember to return to the service allocated with your Pod number if needed).

Let's create a new flow, click Create.

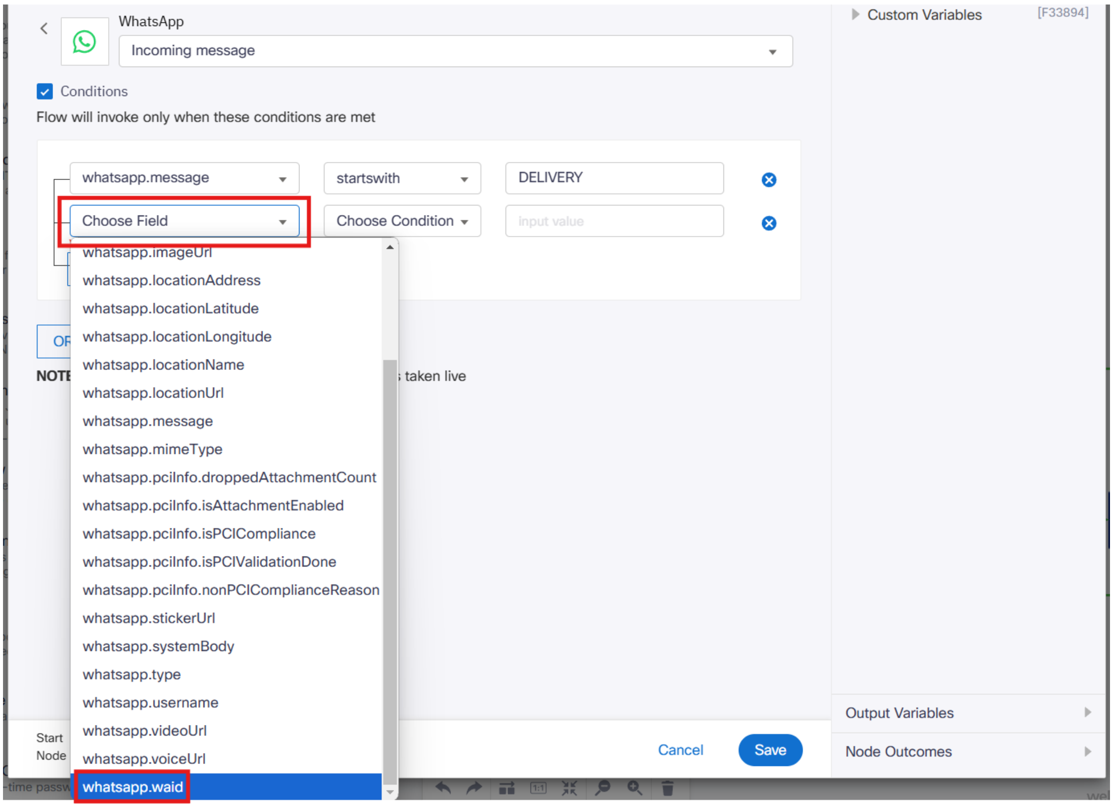


Enter your Flow Name, Pod XX WhatsApp v1 (remember to replace the XX with your relevant pod number), choose the Upload a flow Method, then upload your file (drag your file into the box or click Choose File again), and click Create.

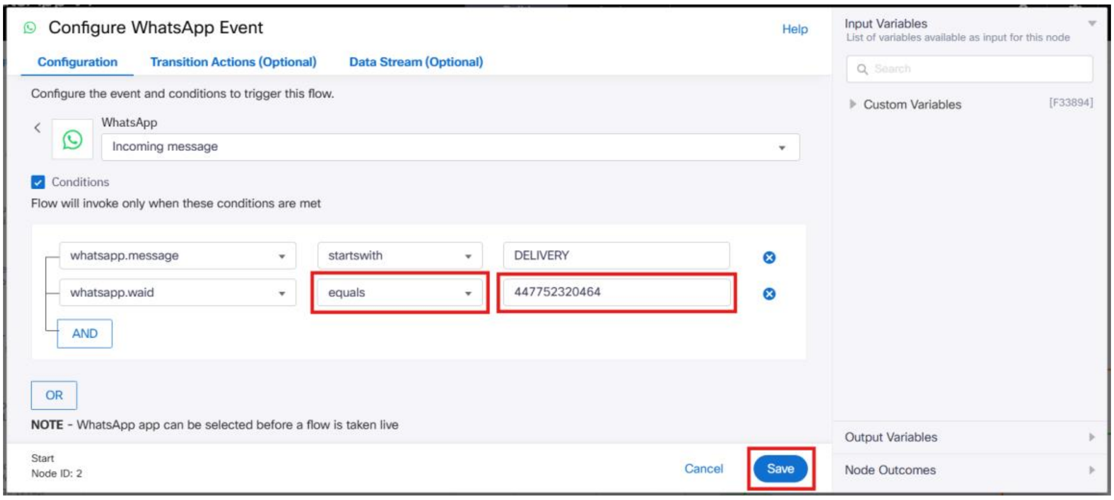
Because this WhatsApp asset is shared between all pods, we are going to add a condition to the Start node. Click AND under Conditions.



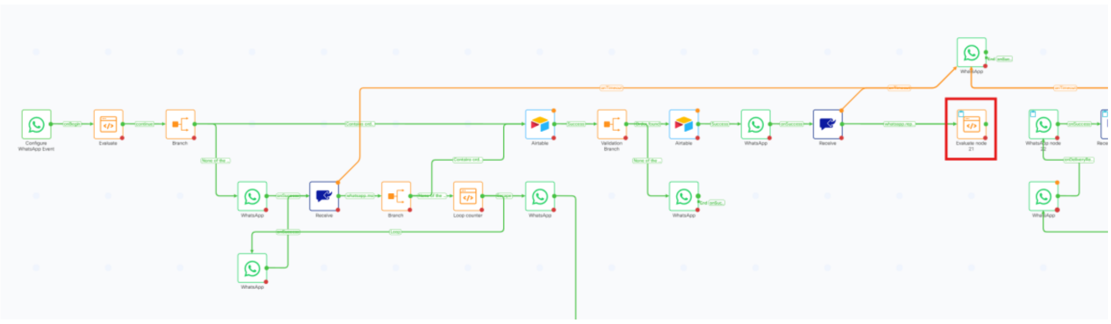
For the Choose Field dropdown select the whatsapp.waid variable



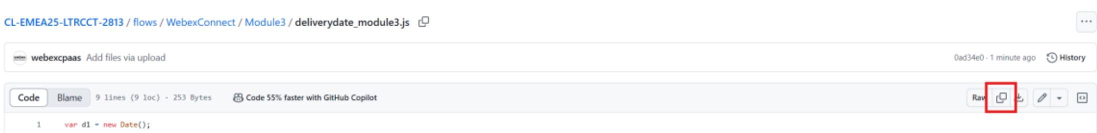
Then choose the equals logical operator and enter your mobile number (DO NOT add ‘+’ at start, and make sure there are no gaps). This will ensure that only you can interact with this workflow, and so we don’t have multiple of the lab flows all being triggered at the same time. Click Save.



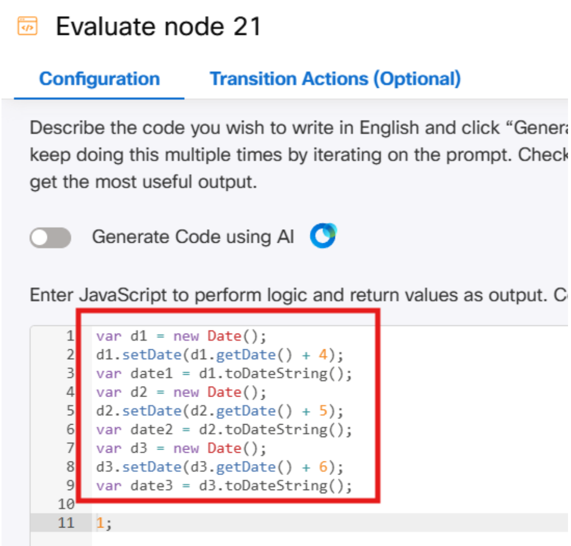
Locate the first node to edit, let’s head to and open Evaluate node 21.



We now need to use the JavaScript from GitHub, copy and paste the code from [HERE]



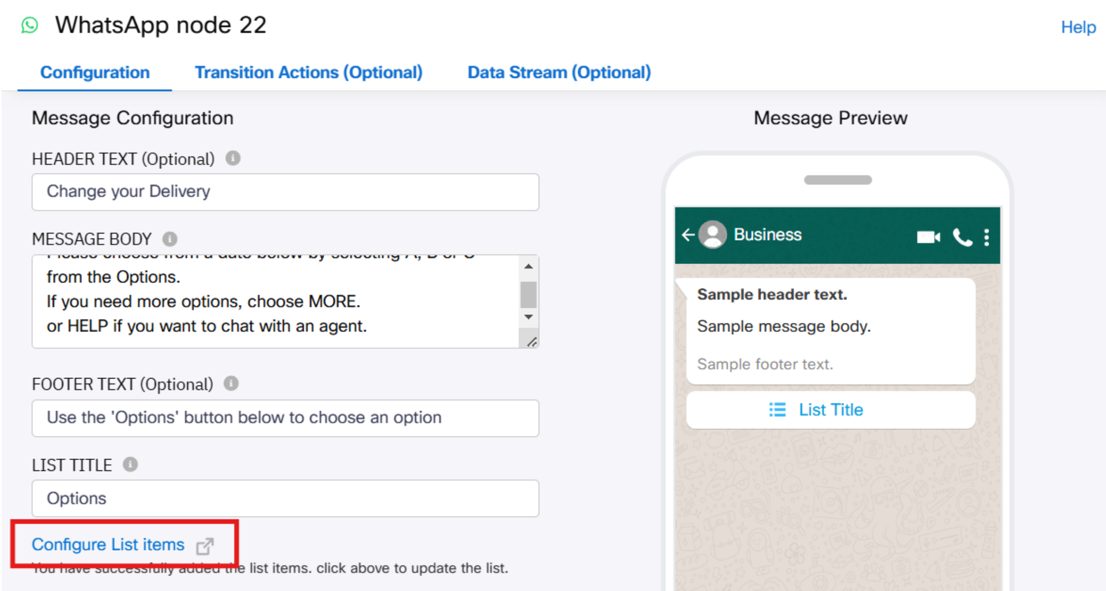
Now in the Evaluate node, paste this code in above the '1;', and save.



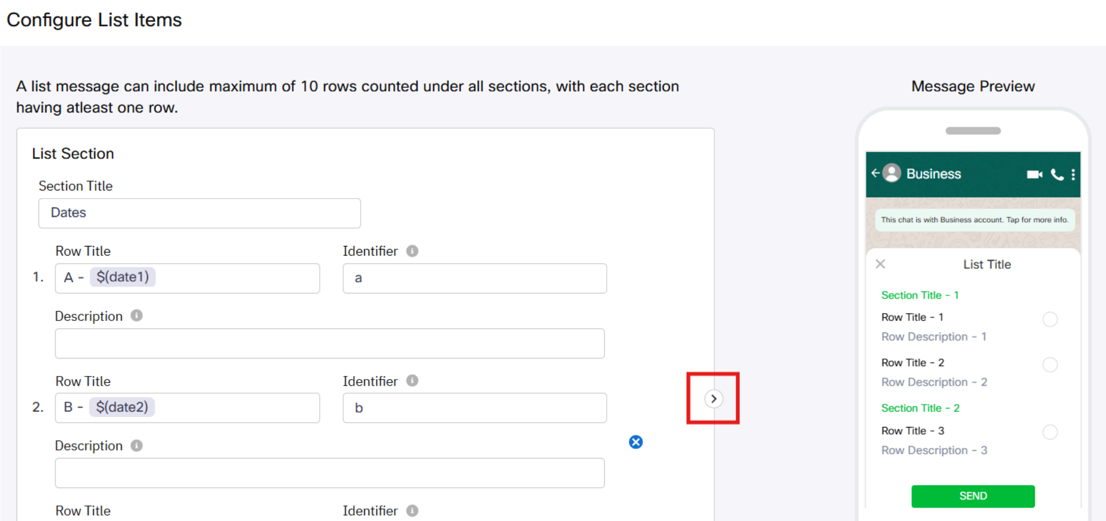
This code creates 3 days for us to use in the options selections (dates A, B & C), which you will see later in the module.

Connect the success path of the Evaluate node to WhatsApp node 22, (to the right) and then open that node.

Click Configure List items.



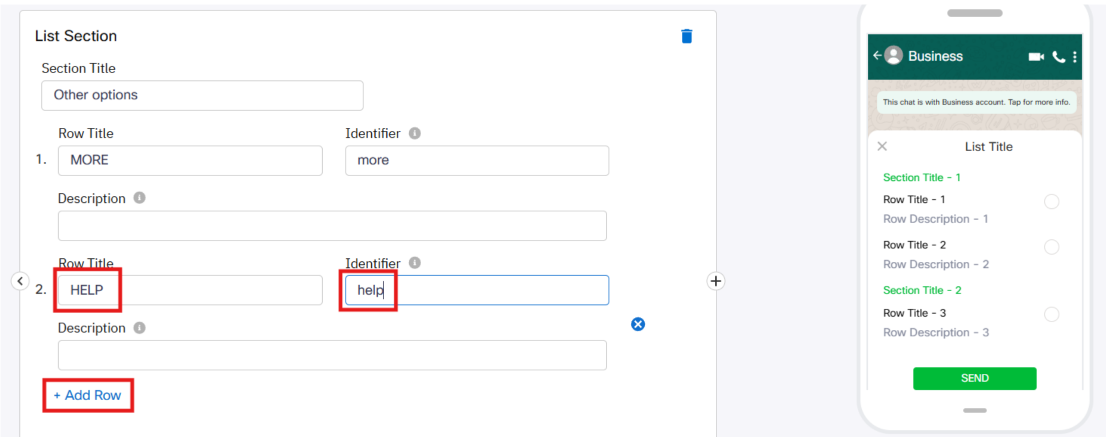
Click the right chevron icon.



Click + Add Row, and we'll add a HELP option to our list.

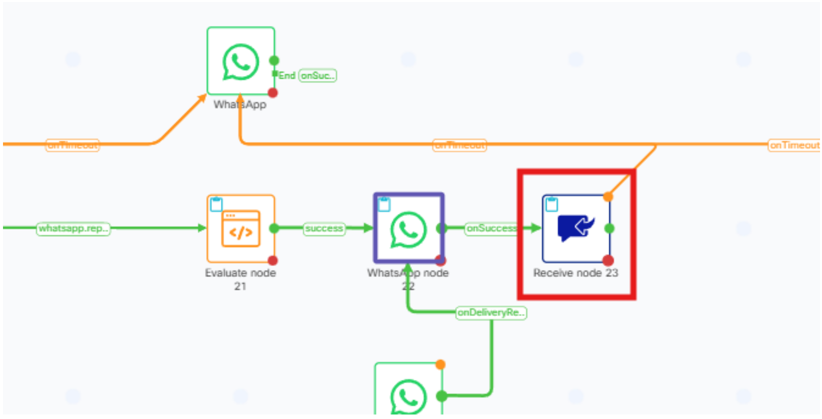
Row Title: Help

Identifier: help

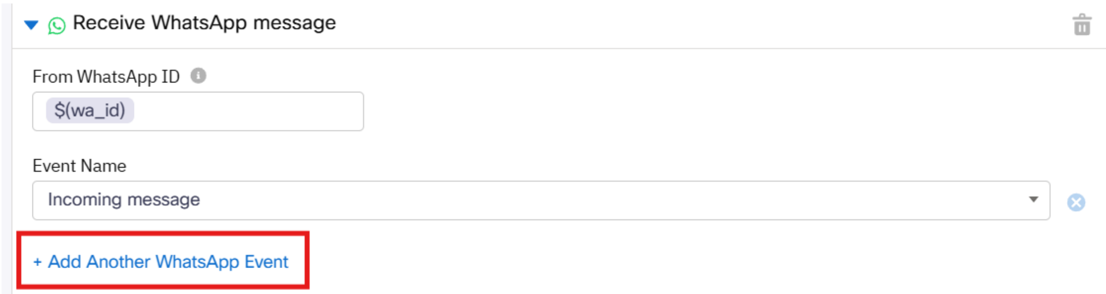


Now click Save and Save again.

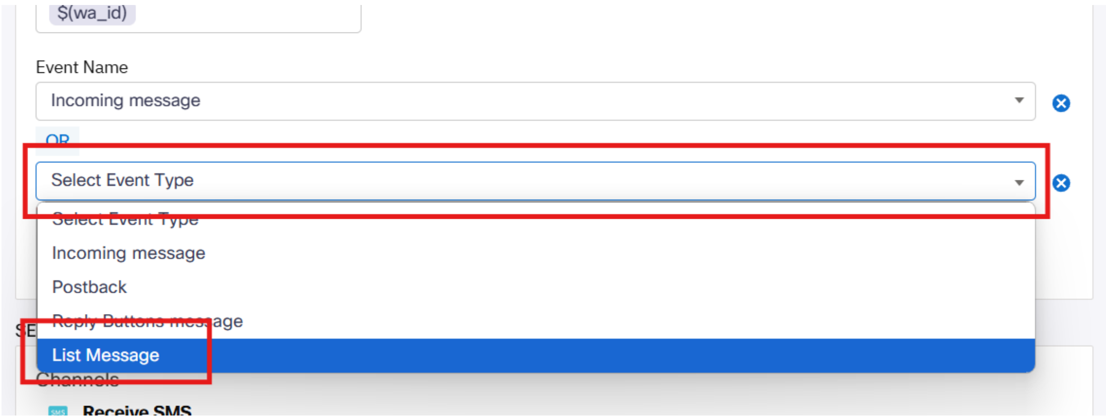
Let's configure the Receive node 23 node, locate and open the node.



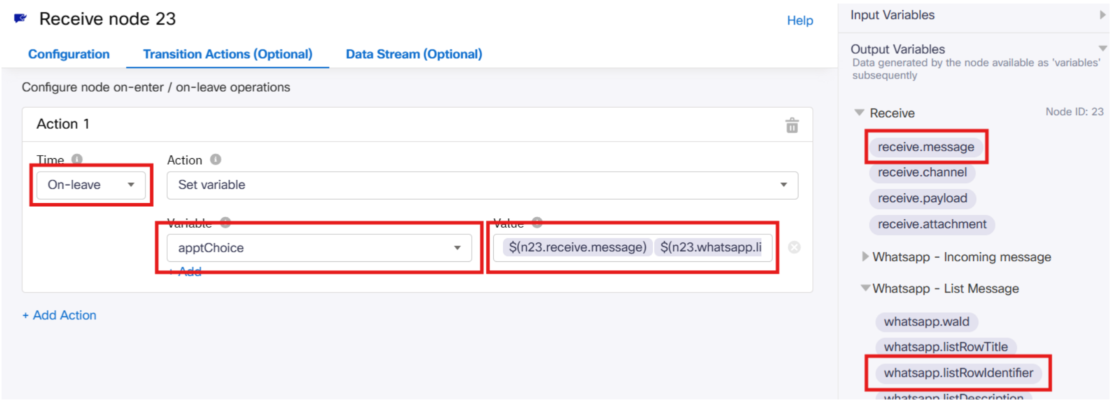
We already have an event listener for an Incoming message, but now let's add a listener for a list message response. Click + Add Another WhatsApp Event.



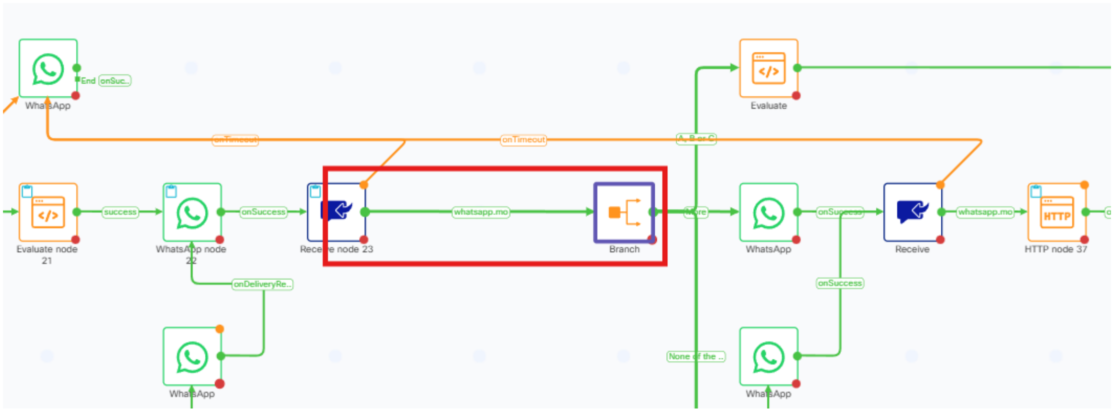
From the Select Event Type dropdown choose the List Message event listener.



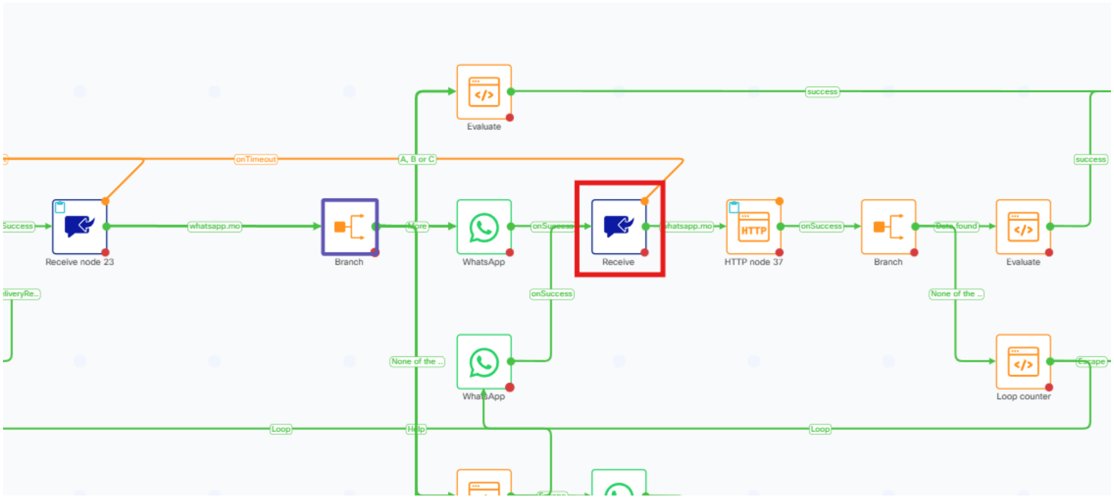
For your reference in Transition actions we are unifying either type of response into a single custom variable we can use named apptChoice in the On-leave transition action (see screenshot below for reference).



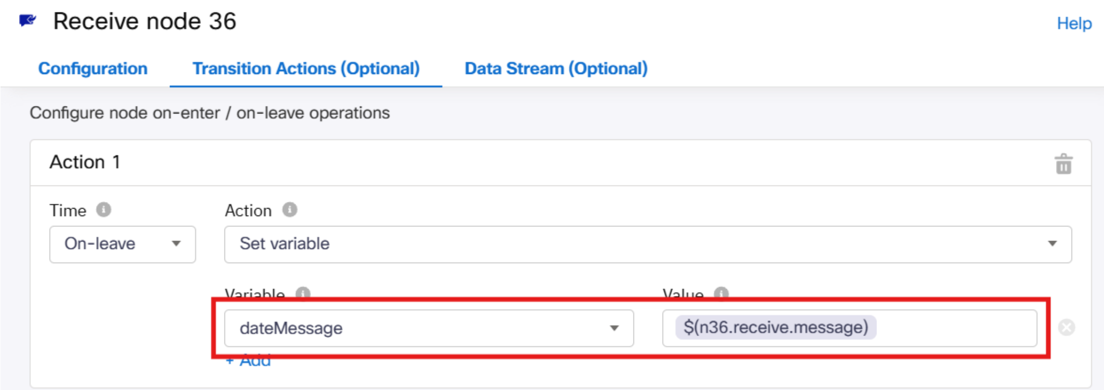
Save the node, then connect the TWO success paths to the branch node to the right.



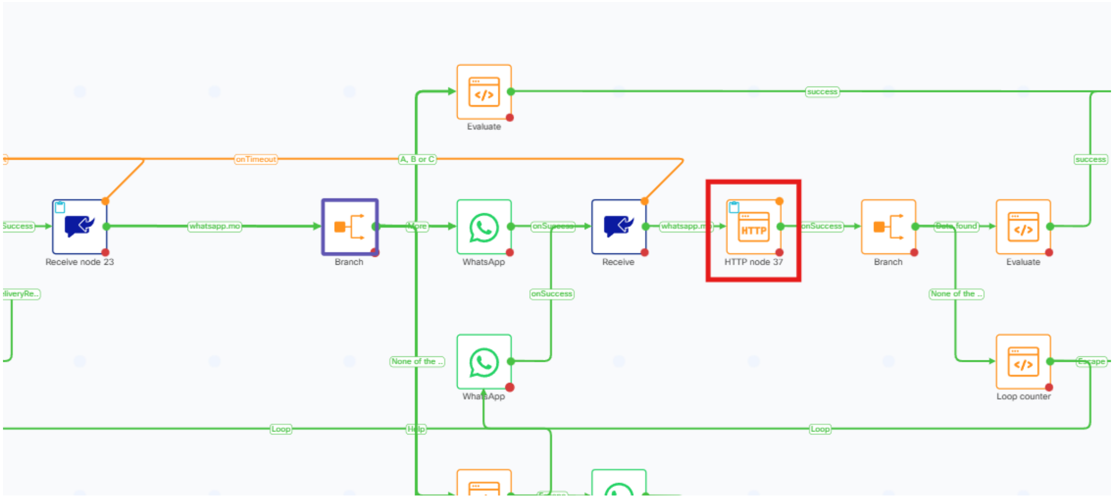
The first half of the flow is now complete, with validation and date options, now we'll add some Natural Language Processing (NLP) to the flow. Locate and open Receive node 36.



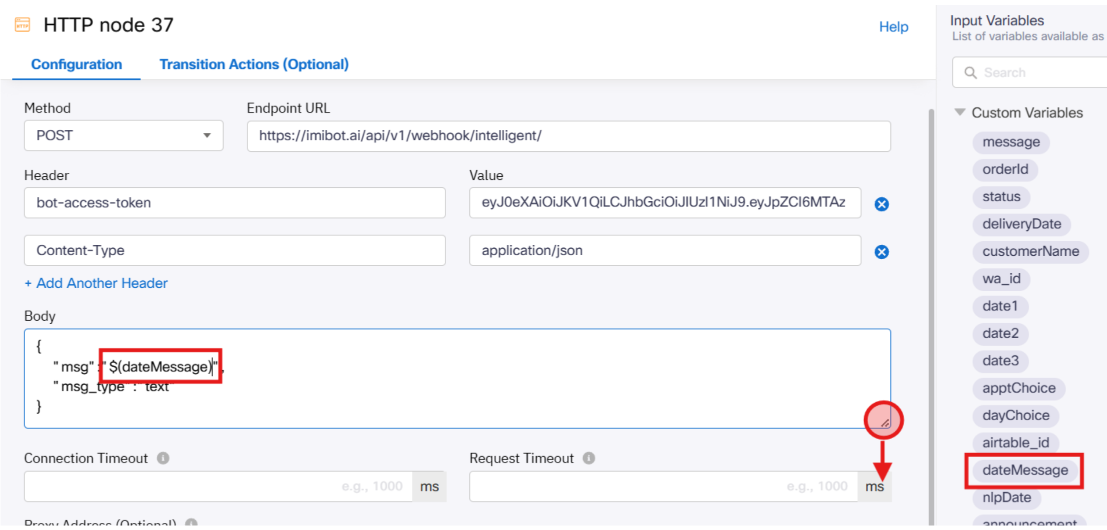
This node expects an open text-based response and we will use this message as a variable to call upon an NLP function in our API call (we didn't have this tenant enabled with NLP functions, but we will make use of an existing NLP environment). In Transition Actions we can see our variable dateMessage is set. There's nothing to change, this is for reference only, now let's click Cancel and head to the next Node.



Open HTTP node 37, the next node along.

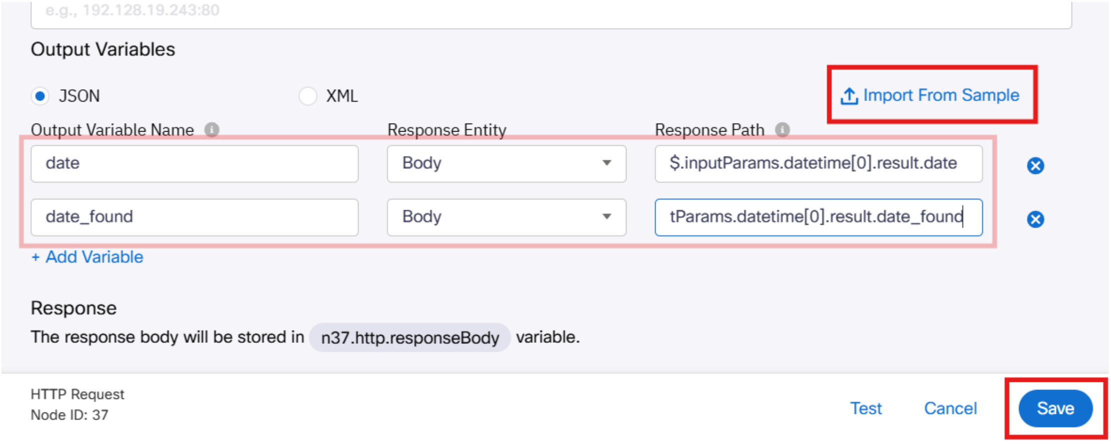


This is a flexible, open-ended node allowing for performing requests to external API services. This particular API call was an early version NLP function we developed to understand date and time. First let's make the Body field a bit bigger so we can see the complete JSON body. Then let's replace the 'ABC' value for 'msg' to our dateMessage variable from the Custom Variables.

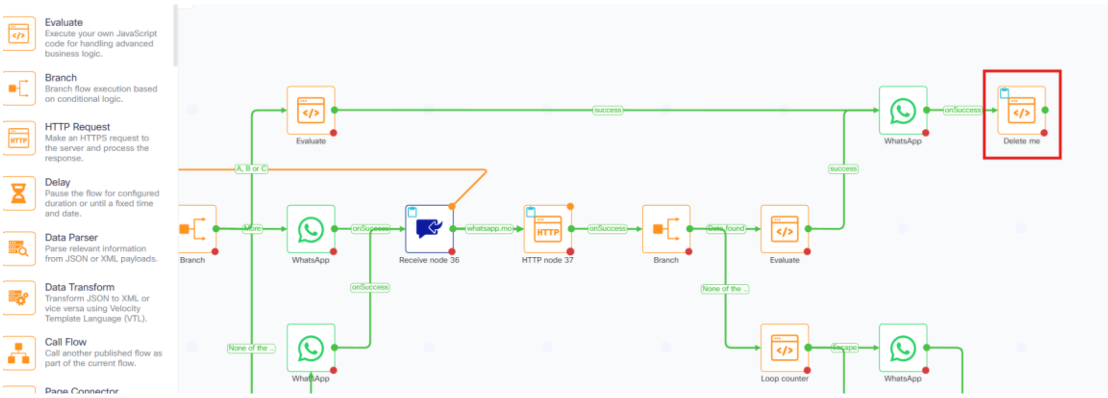


Below, you'll see some Output Variables are configured. For your understanding, let's see how we got the two variables. Click on Import From Sample.

Here we see a sample JSON body response load for this API call, we parsed the relevant variables and then gave them a local flow variable name (date & date_found). Go ahead and close the sample view, then click Save to exit the node.

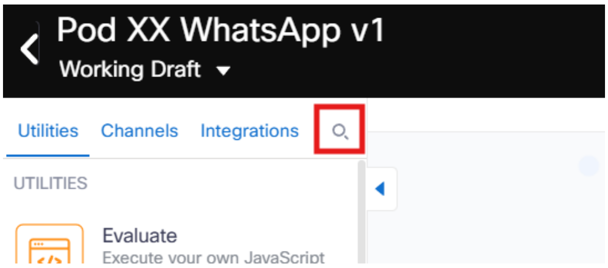


Let's locate the evaluate node named Delete me.

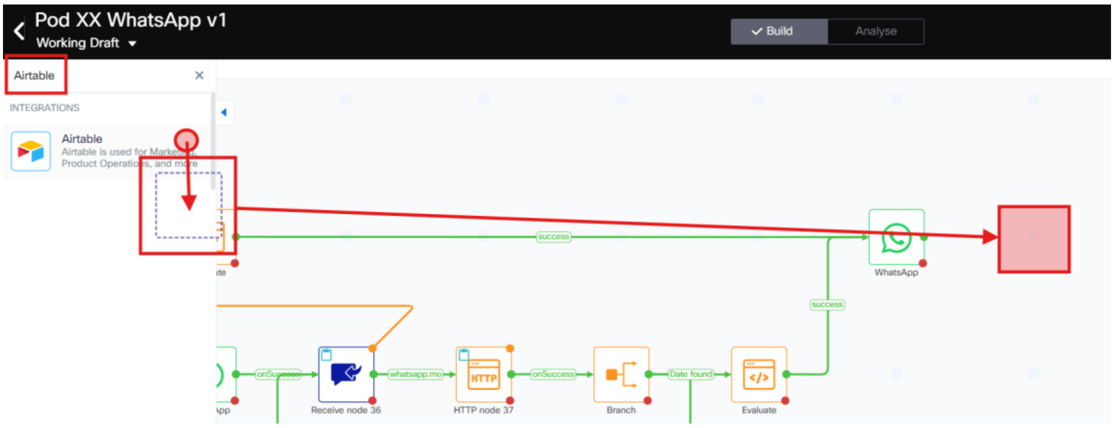


Click on the node ONCE – we just want to select the node, now hit Delete on your keyboard or click the trash bin icon at the bottom of your UI.

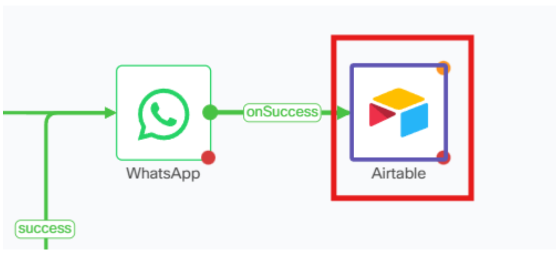
Let's replace this node with the Airtable, custom node. Click on the magnifying glass icon at the top of the left-side node menu.



Now type 'Airtable', click and hold on the Airtable node, dragging the node to where we just deleted the evaluate node, on the flow palette.

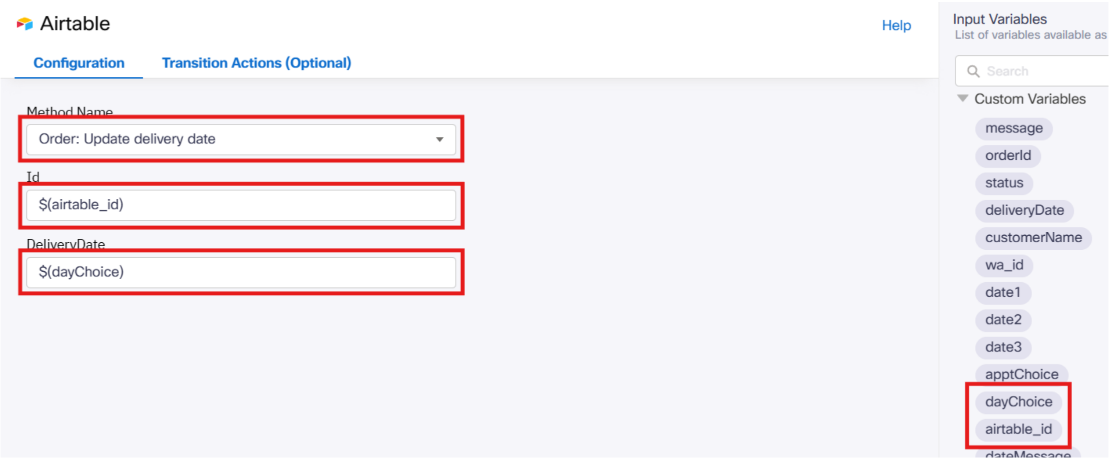


Connect the success path from the WhatsApp node to the left, to our new Airtable node, then let's open the node.



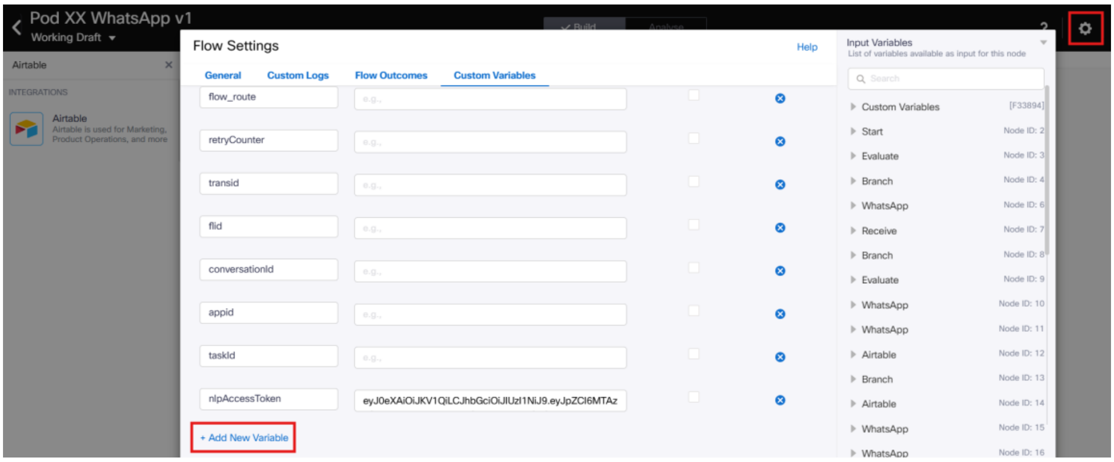
Click Select Method Name and from the dropdown select the Order: Update delivery date method. For your reference, the other methods listed are all API calls we have integrated to our custom node for reuse across this demo. Now let's fill the fields needed to update our delivery date in the database (both are found in our Custom Variables:

Id:	airtable_id
DeliveryDate:	dayChoice



Now Save and exit the node.

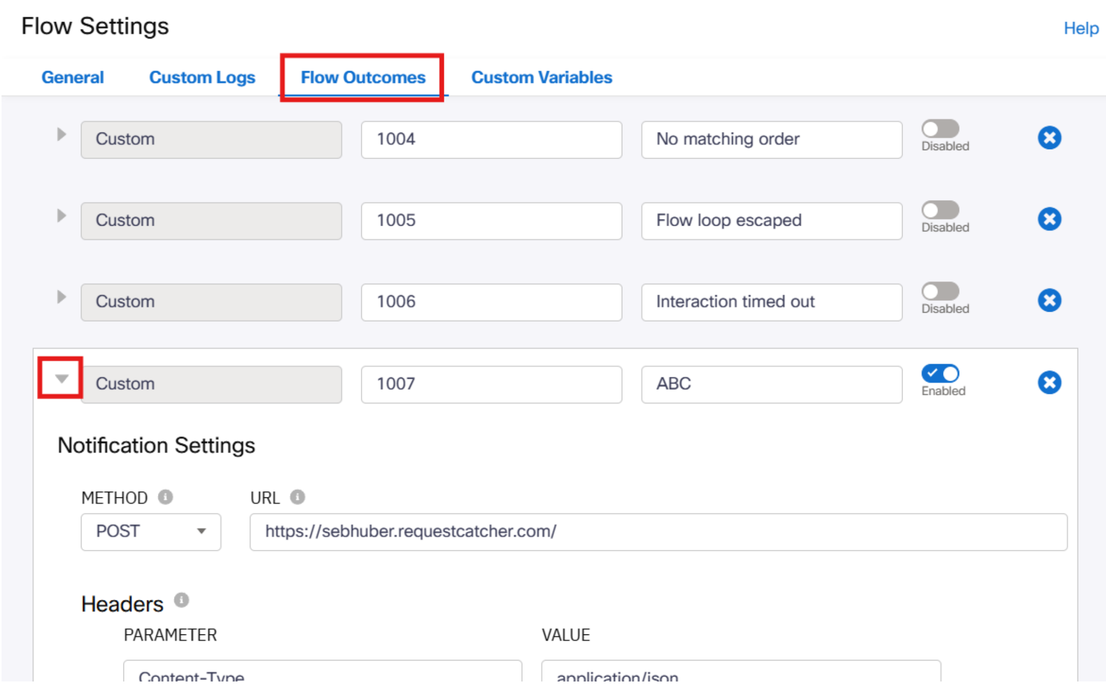
Before we close the success path of the Airtable node we need to create a Custom Variable, and custom Flow Outcome. Click on the settings cog, top right of the UI, and after you enable Descriptive Logs for 1440 minutes (like we did in Module 2), head to the Custom Variables tab, and scroll down to the bottom of the list until you locate the Add New Variable, go ahead and click on this.



This will add a new blank variable, let's fill the fields:

Variable Name (left field): myName
Default Value (right field): Type your name (first and last)

Now head to the Flow Outcomes tab. Click on the dropdown menu next to the last OUTCOME (STATUS CODE 1007), to expand the view.

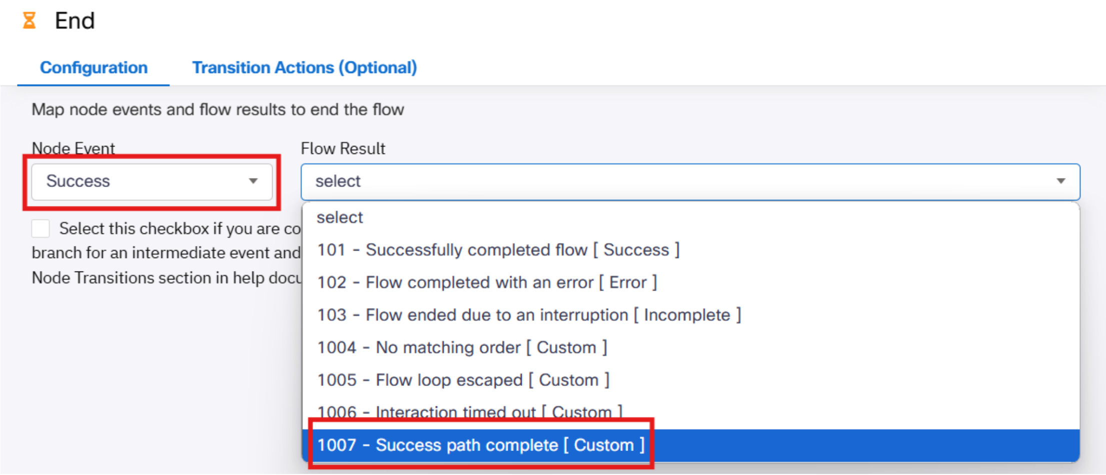


This flow outcome has been enabled with an option to notify a 3-rd party reporting tool (our lab webhook – the requestcatcher URL), feel free to open that URL in a new tab. Your lab proctors will also receive a notification when this flow outcome has been reached.

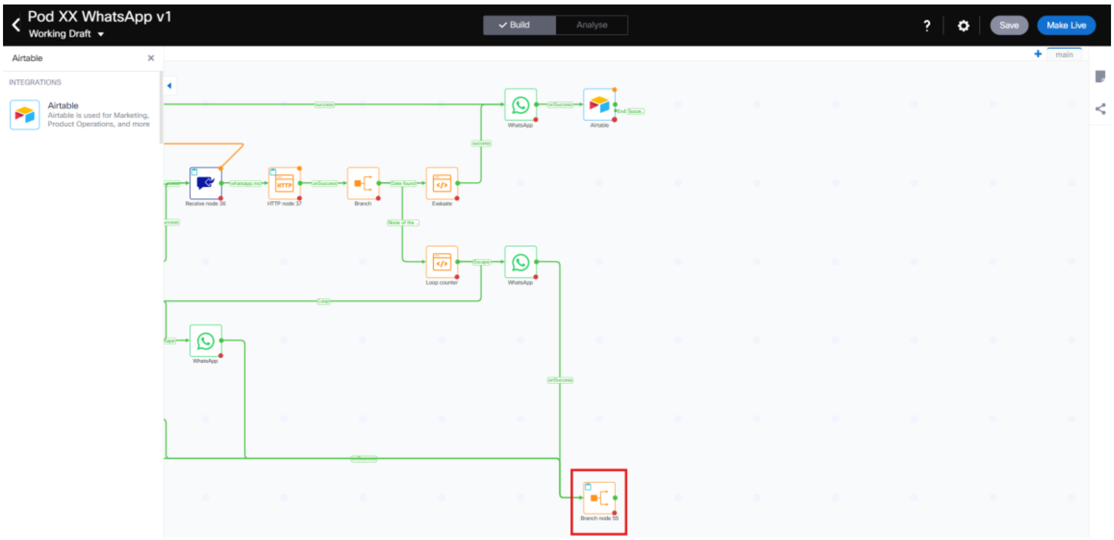
Let's change the DESCRIPTION field from 'ABC' to Success path complete and click Save.

Click, hold and drag the Airtable node's success path into empty space.

Then choose Success as the Node Event from the dropdown, and for Flow Result, let's select the new custom Flow Outcome we just made.

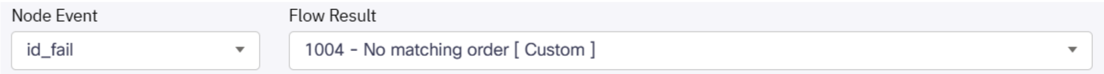


Let's configure our final piece of this flow, we want to match the flow outcomes of Branch node 55.

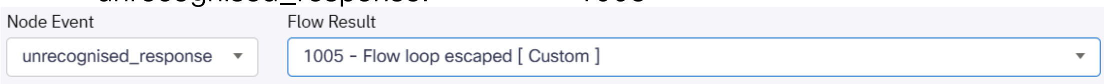


Click and drag the success path for the branch node until our paths are set. Match the following branches with the following outcomes:

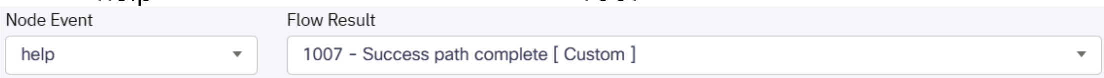
id_fail: 1004



unrecognised_response: 1005

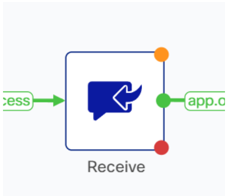


help 1007



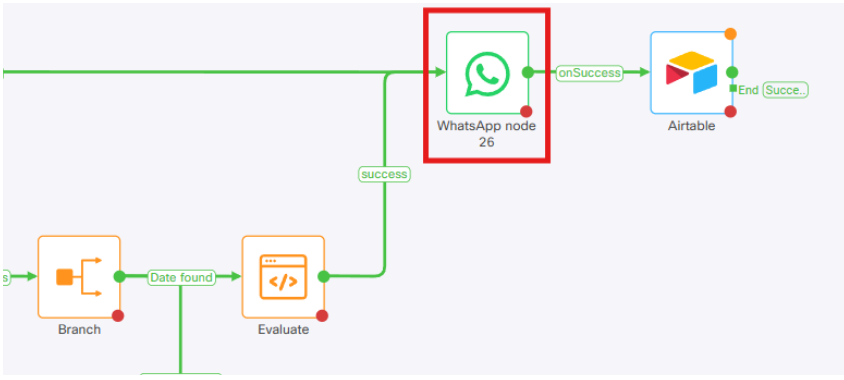
Note: You are now almost ready to publish the flow
When publishing any flows on Webex Connect that you have imported, you will need to open and save all receive nodes, so that the flow recognizes the nodes as configured before we can make the flow live.

To do this locate a Receive node in the flowbuilder UI (see example below)

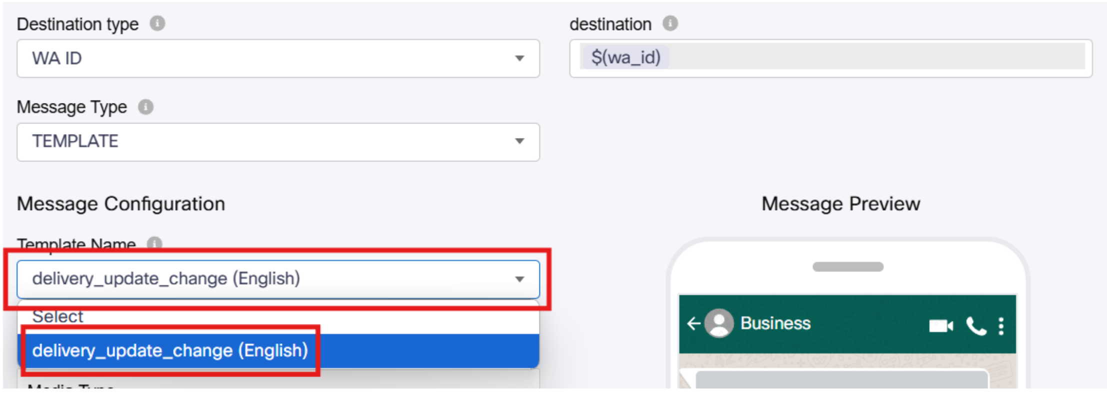


Double click on these to open the node, and then click Save to close the node again.

Because our flow contains a WhatsApp node with a template, we must also configure this node. Locate and open the WhatsApp node that connects to our Airtable node at the end of the flow (see screenshot below for reference).



Now select the template listed in the Template Name dropdown.



You will need to scroll down and fill the variables with the following values:
Image: https://awsprodemailassets.s3.amazonaws.com/0ae1dbc4-e5aa-46c3-8aed-55995aabf5b7/calendar_spaced_157009324497803.png
variable1: Select Text as Parameter Type, then enter Value as: $(customerName)
variable2: Select Text as Parameter Type, then enter Value as: $(dayChoice)

Then click Save.

OK, let's publish the flow. Click Make Live in the top right of your UI, then select our Webex CX LTRCCT2813 WhatsApp asset and click Make Live.

Let's recap what you have configured in this module:

- You created a deterministic-logic based customer journey, enhanced with AI Based NLP

- You used Airtable to verify the messenger, and then updated delivery dates using our custom integrations node for a simplified API call build process.

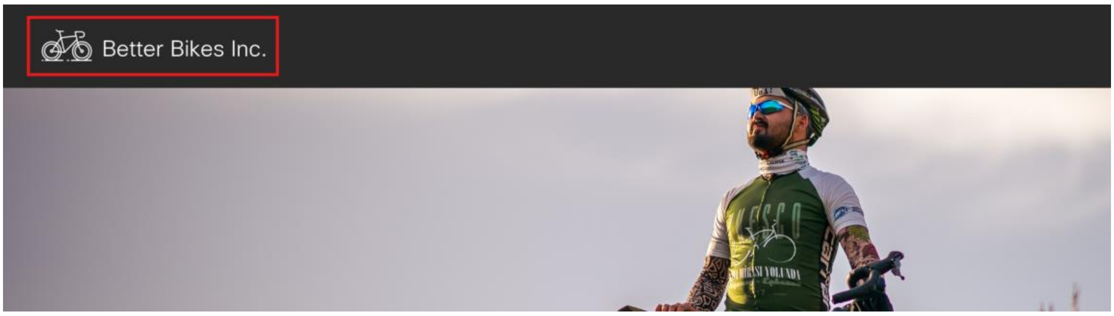
- You then enhanced the reporting capabilities with custom outcomes and variables.

Once the flow is Live you're ready to test your workflow.

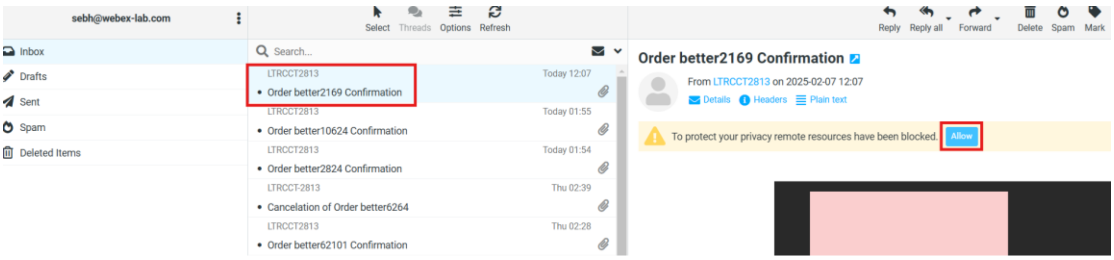
Note: You may need to refresh your browser if Live doesn't show after a minute.

3.2 Testing the WhatsApp demo

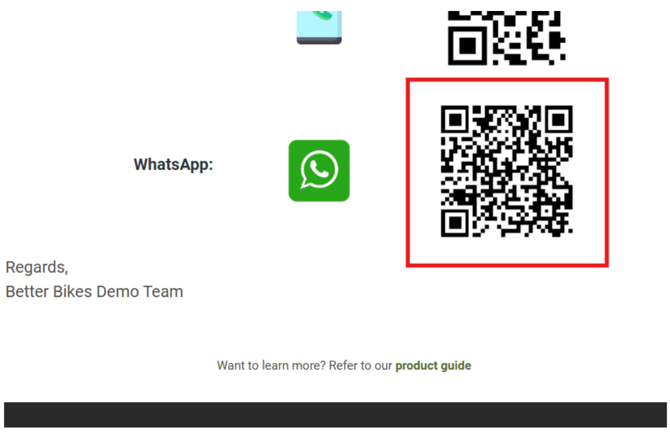
To test the demo, let's head back to the glitch website you created in 0.9 and head back to the home page (click on the logo in the top left corner).



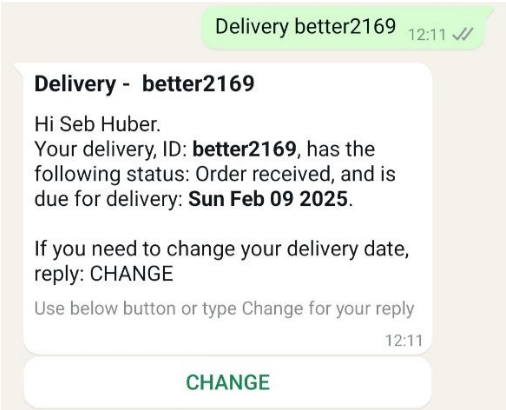
Now let's click Enter, so we have a new order in our email inbox. Head over to your email inbox, select the new order and click Allow to let the images load.



Scroll down in your email so you can see the WhatsApp QR code and scan it with your camera or QR code reader on your mobile phone (If you do not have WhatsApp on your phone or have issues with your camera, please ask for help from a lab proctor).

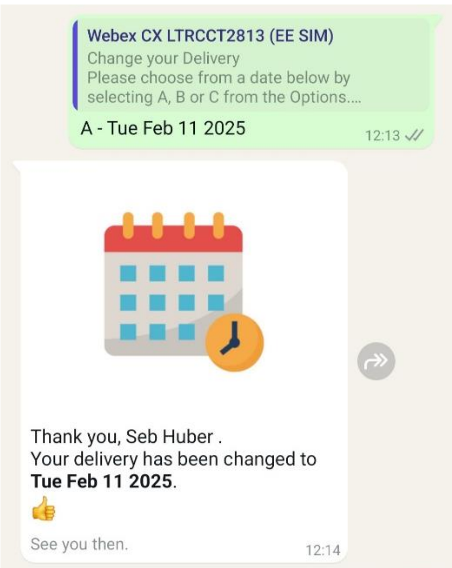


Run through the journey by following the prompts in the chat, 'CHANGE'



Then run through the A, B and C, MORE and HELP options.

For the A, B and C, and MORE options, if you run through the instructions in the WhatsApp chat should both end with a Calendar update that looks like the below screenshot.

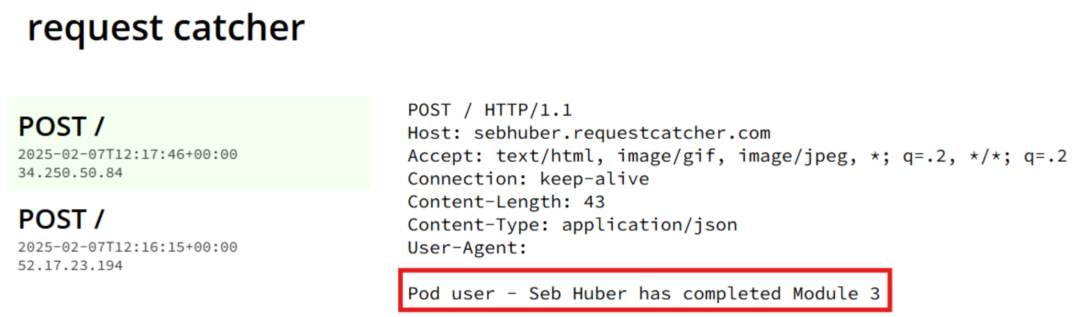


For the HELP option you will receive a placeholder WhatsApp message that reads:

> "No problem. We're transferring you to our help team now."

In the Addon task (3.3) we will elevate this option to WxCC for an agent to pickup.

We will also receive an API call to our webhook listener that will look like the below screenshot for all the different journey options.



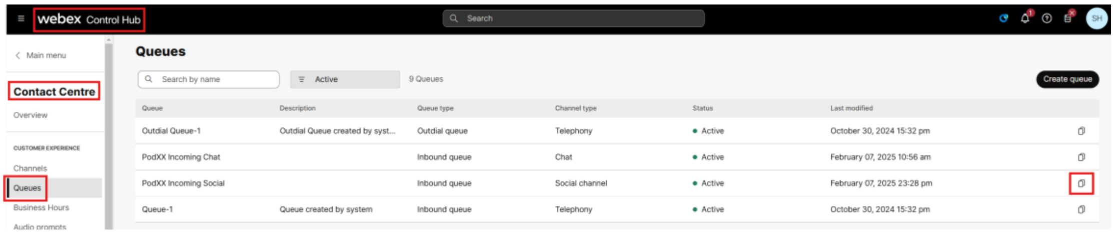
Congratulations – you've finished all the standard modules in this lab! 🎉

3.3 Addon: Flow enhancements (Elevate to WxCC)

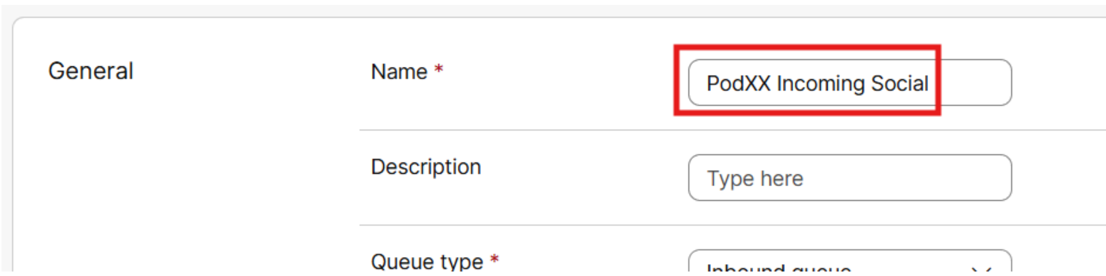
Please note, this is additional content.
This task can be skipped, but you will learn how to implement database connections for receiving more information about the caller. We recommend completing Addon modules once you have reached the end of your 'standard' lab if you have leftover time.

Before we get started on the work in Webex Connect, we need a Queue where the Channel will be assigned. Head back to Control Hub > Contact Center > Queues in the left sidebar.

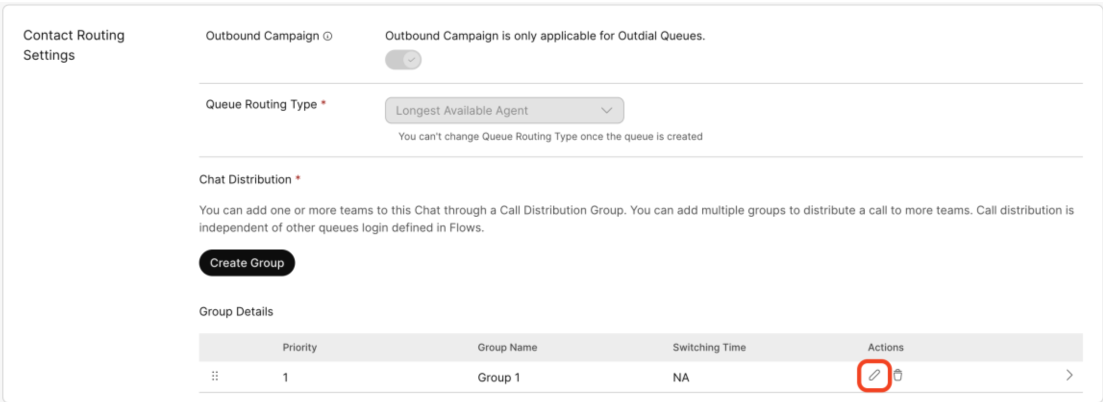
We are going to use the PodXX incoming Social queue, click on the copy icon at the right of the line.



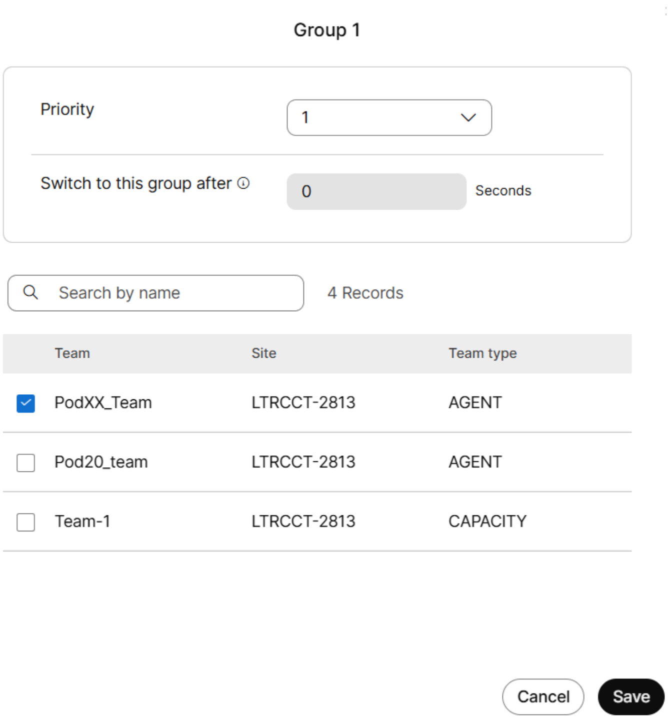
Change Name with PodXX to your relevant assigned Pod number and remove the '_copy'.



In Contact Routing Settings, under Group details, click on the Edit Pen for the existing Group 1.



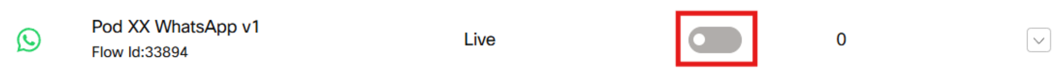
Remove the assignment for PodXX_Team then choose your PodXX_Team and Save.



Then click Create.

Now let's head back to your Service on Webex Connect.

First, we will need to disable the flow you built in 3.1 (Pod XX WhatsApp v1). Simply toggle the State so the flow is disabled.

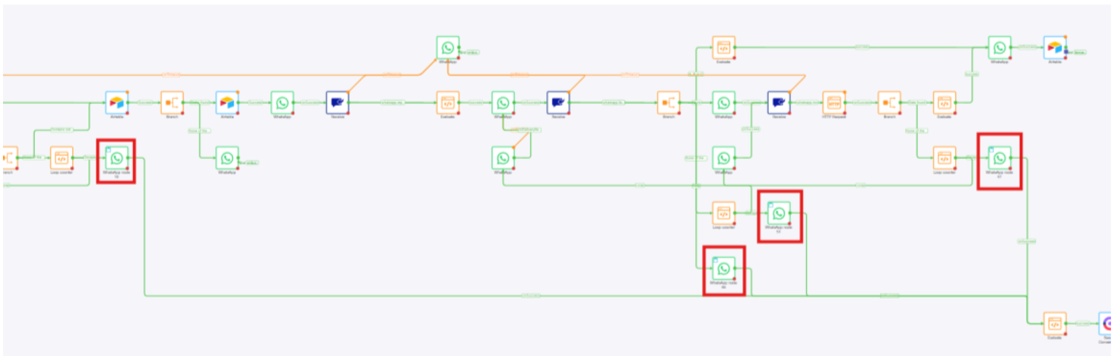


We're going going to Create a new flow using the Upload method again, and this time we will use the 33949_whatsapp-v2.workflow file we saved from GitHub earlier.
Name your flow 'Pod XX WhatsApp v2' (remember to change XX to your pod number), and create your flow.

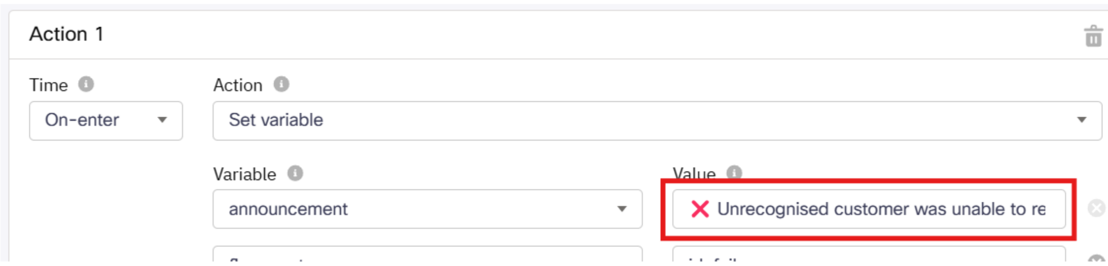
Zoom out and navigate to the end of the new flow. You will notice that we have added a few nodes extra from our v1 flow. The extra nodes already configured will provide us with the ability to elevate our WhatsApp conversations to an agent on WxCC.

We're going to review the dynamic announcements that different routes provide and append some deterministic information for the agent to understand the context of why the customer has been escalated.

First let's review one of the WhatsApp nodes that lead to our WxCC escalation route, you can open on any of these (node 10, 47, 48 or 51).

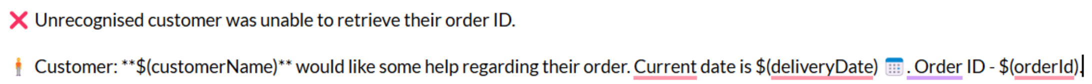


Now head to Transition Actions and you will see the announcement set.

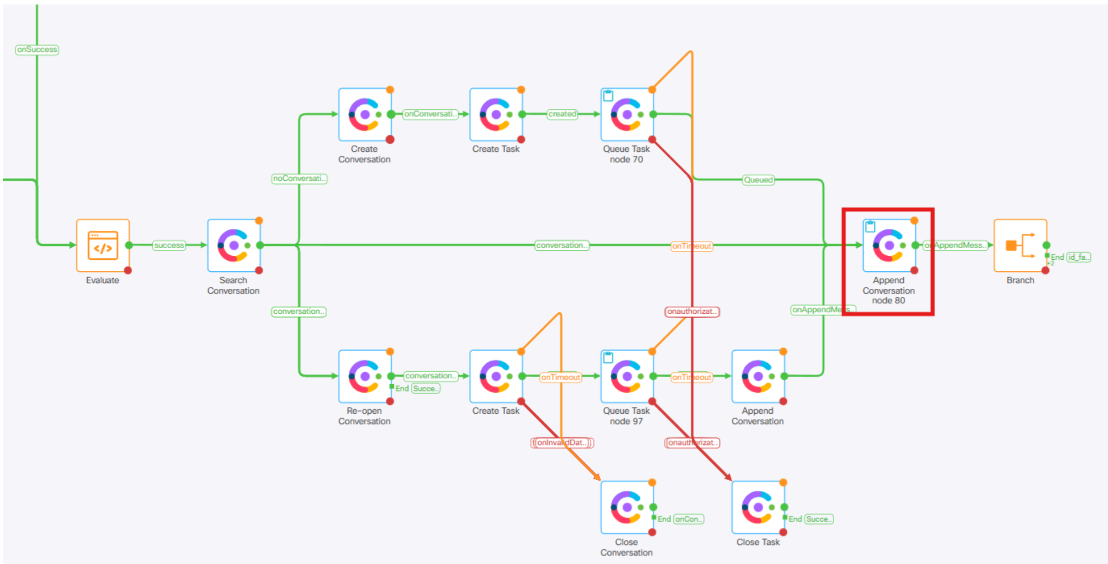


Click on the field, copy the text and paste it here https://onlinenotepad.org/notepad so you can get a better view of the text context.

Head back to the flowbuilder, close that node and open one of the others highlighted above and repeat the steps to compare the different announcements. See an example below of contrasting announcements.



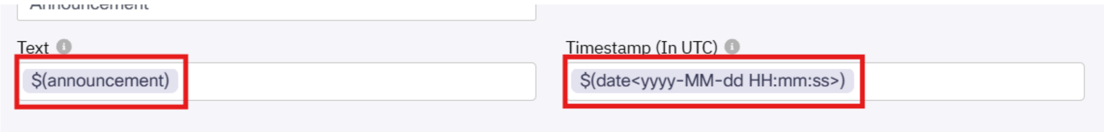
Now let’s head to the escalation part of our flow. Locate and open Append Conversation node 80



For the Text field we're going to use the announcement variable we just observed from the Transition Actions on the WhatsApp nodes. Delete the 'abc' and replace with announcement from our Custom Variables. You will need to tap your space bar and Backspace keys for this characters to register with the field.

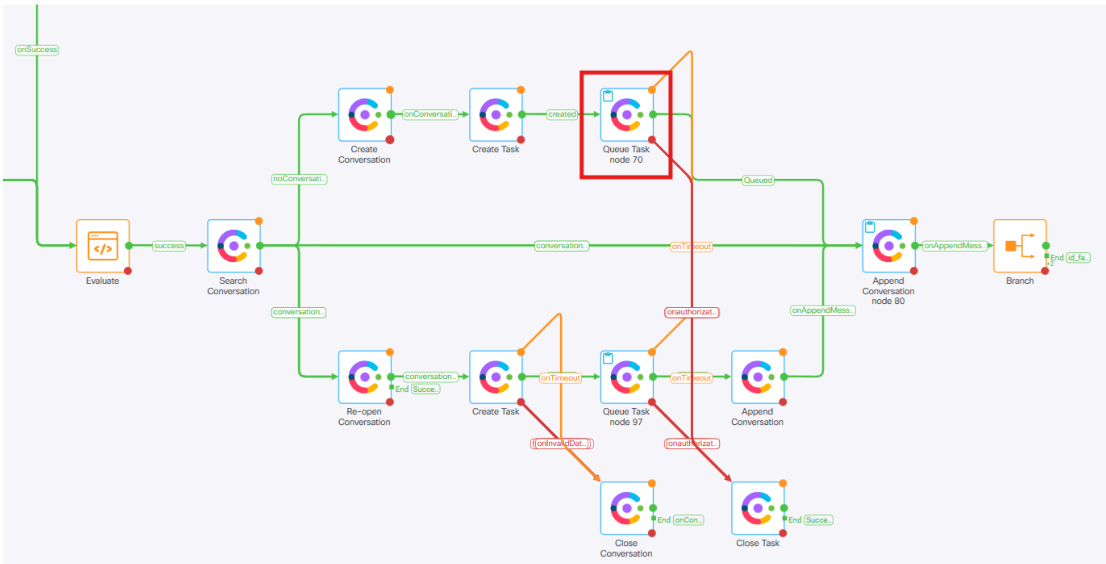
We're now going to use a system timestamp variable. In the Timestamp field delete the 'abc' and paste in this variable:
$(date<yyyy-MM-dd HH:mm:ss>)
Tap your space and backspace keys ensuring there are no spaces after the variable.

When you click somewhere off the field, the variables should like this if configured properly:

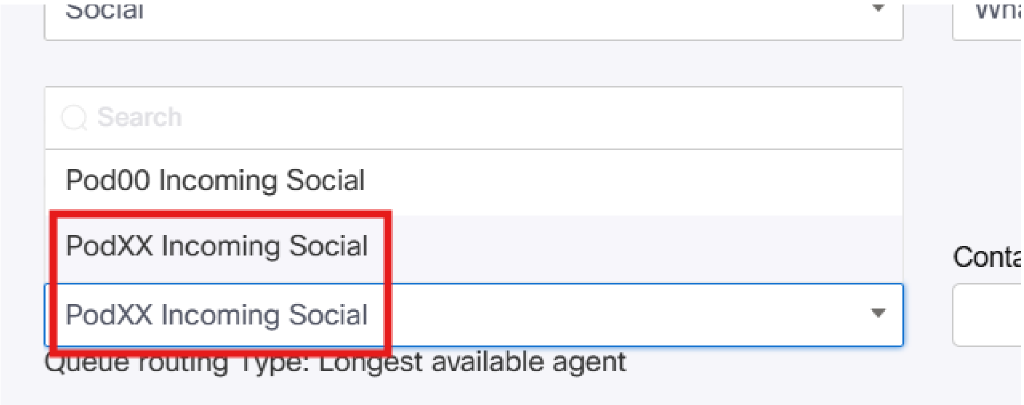


Click Save to exit the node.

Now let's open Queue Task node 70

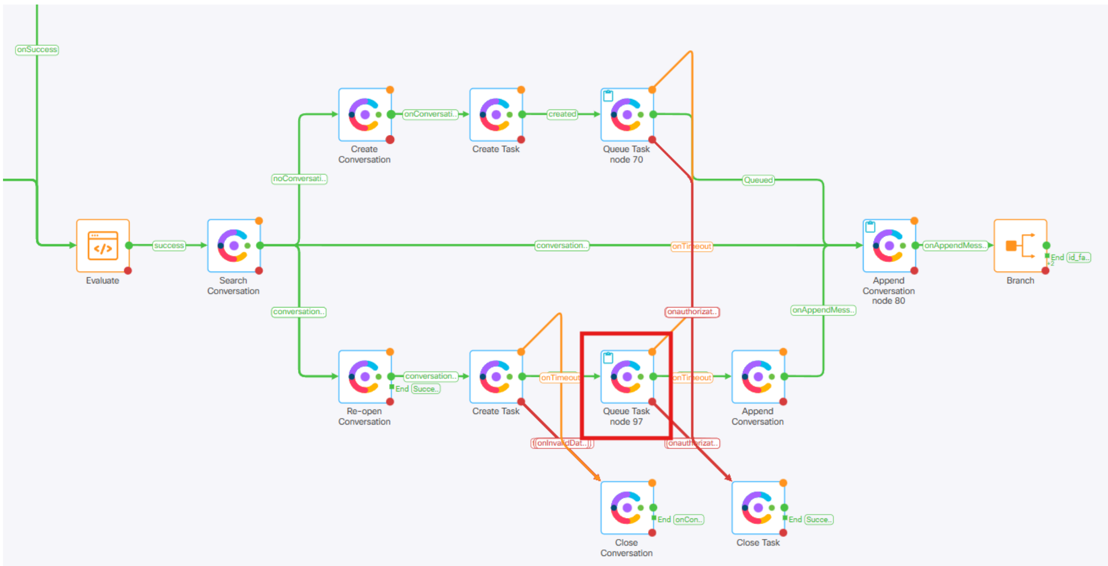


For the Queue Name dropdown we need to select the queue we created at the start of this task, named ‘PodXX Incoming Social’ (remember you have replaced the XX with your relevant pod number)



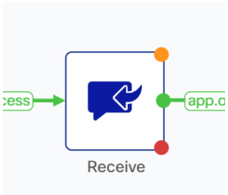
And click Save.

Now let’s repeat this process for Queue Task node 97



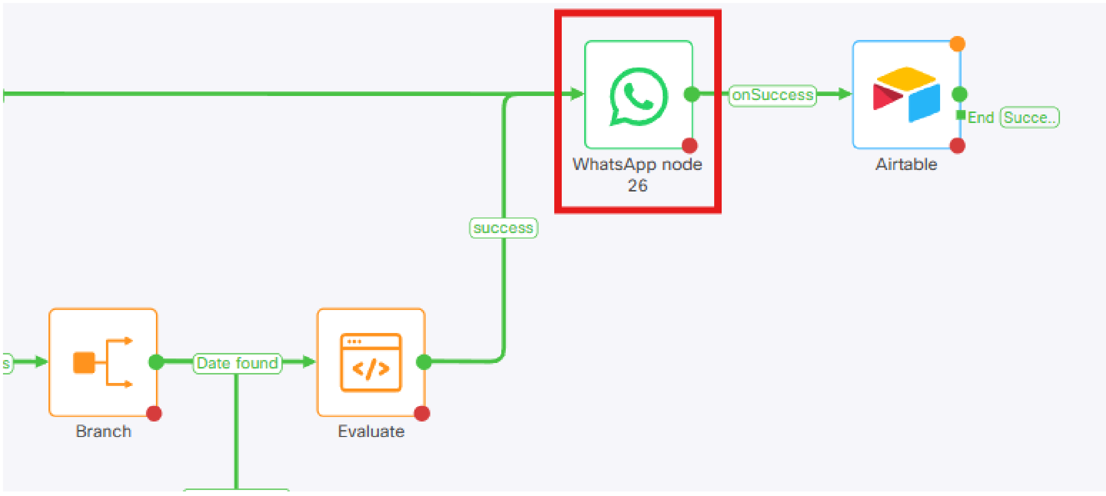
Note: You are now almost ready to publish the flow
When publishing any flows on Webex Connect that you have imported, you will need to open and save all receive nodes, so that the flow recognizes the nodes as configured before we can make the flow live.

To do this locate a Receive node in the flowbuilder UI (see example below)

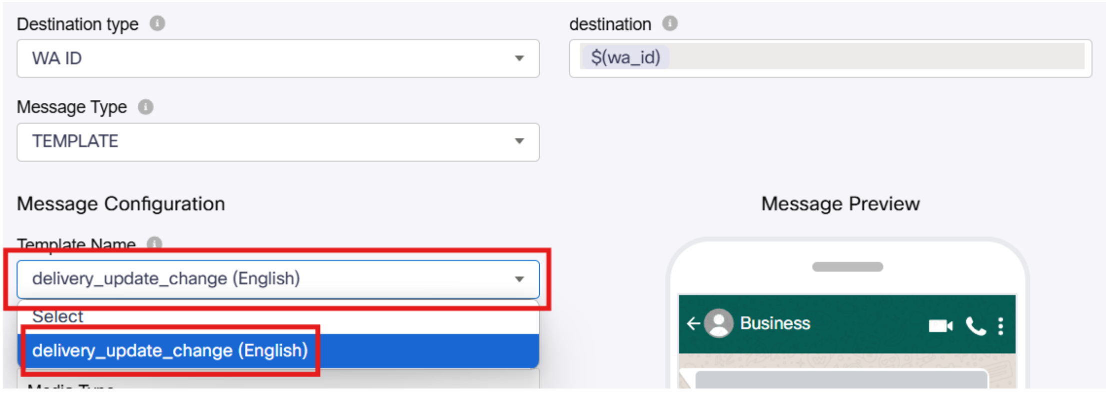


Double click on these to open the node, and then click Save to close the node again.

Because our flow contains a WhatsApp node with a template, we must also configure this node. Locate and open the WhatsApp node that connects to our Airtable node at the end of the flow (see screenshot below for reference).



Now select the template listed in the Template Name dropdown.



You will need to scroll down and fill the variables with the following values:
Image: https://awsprodemailassets.s3.amazonaws.com/0ae1dbc4-e5aa-46c3-8aed-55995aabf5b7/calendar_spaced_1570093244978O3.png
variable1: Select Text as Parameter Type, then enter Value as: $(customerName)
variable2: Select Text as Parameter Type, then enter Value as: $(dayChoice)

Then click Save.

OK, let’s publish the flow. Click Make Live in the top right of your UI, then select our Webex CX LTRCCT2813 WhatsApp asset and click Make Live.

Once the flow is Live you’re ready to test your workflow.

Note: You may need to refresh your browser if Live doesn’t show after a minute.

Testing:
To test, run the same steps you completed for testing v1, but look to trigger the HELP route, as this will escalate to WxCC. You can handle chats as you did in the prior Module, 2.5 & 2.7.

You can also try exiting on the loop escapes. There are 3 looping parts of the flow.

1st is the initial message we send to trigger the message. For your Order ID, remove the 'better' part of the ID so we trigger the looping experience.

2nd is the main options menu. Try replying with messages the branch logic isn't anticipating.

3rd is on the MORE branch. The NLP function is anticipating a date, so try responding without any day/date context.

As demonstration builders, you may think to enhance the experience by 'catching' the unrecognized intents with AI agents. You could also use AI to categorize only, with no responses being sent to the messenger, the types of unrecognized responses, and then enhance your journey based on data observed using the AI agents.

***

Congratulations – you've finished ALL modules in this lab! 🎉