MYWAVE.ai

MyWave Technical Document

A GUIDE FOR USERS OF THE MYWAVE PLATFORM |

JANUARY 2024

Contents

Document Details

Document Control

Version	Date	Author	Sign-Off
1.0	March 2024	Ollie Hermans	Amy Johnson
2.0	March 2024	Alexandra Foster	Amy Johnson
3.0	April 2024	Ollie Hermans	Amy Johnson

Purpose

This document is a user-friendly guide for MyWave platform users. It explains the tasks and fields used by SysAdmins, Developers, and Wave Creators. The goal is to help users easily navigate and understand the platform. By offering clear instructions on key features, it empowers users to make the most of MyWave for effective Wave process creation and deployment.

Target Audience

This document is tailored for partners/individuals involved in the use and administration of the MyWave platform.

MyWave Introduction

MyWave is a Generative AI driven conversational Co-Pilot for employees and end customers.

It enables any business process to be transformed by generating and automating digital processes for customers, employees, and assets.

It can remove administrative burdens, improve system usability, and help customers and employees to achieve their goals.

MyWave adapts processes in real-time based on context, intent, business, personal and machine data for each individual user.

Key Links

MyWave Innovation Hub: https://innovation-hub.training.app.mywave.me/

MyWave Website: MyWave.ai – Generative AI for business

Further Assistance

For further assistance, email: support@mywavesupport.zendesk.com

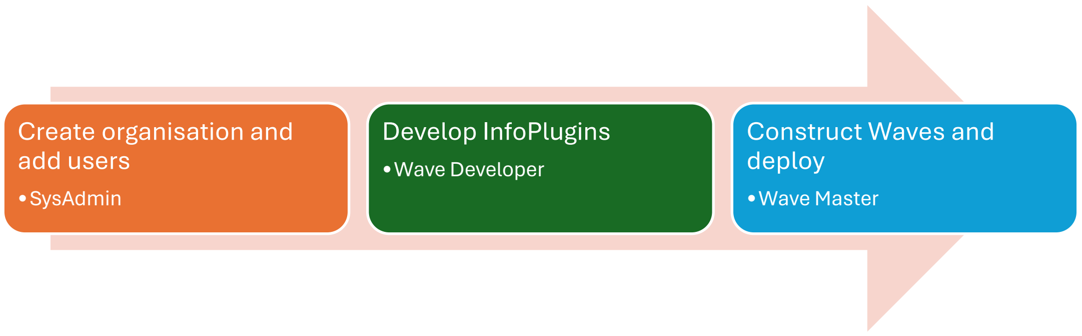
Roles

There are three primary roles for administering the MyWave platform. A user can have multiple roles assigned:

Role	Tasks
SysAdmin	• Create and edit organisations • Add and manage users.
Wave Developer	• Add and customise InfoPlugins • Distribute software.
Wave Master	• Create user journeys (Wave processes).

Overview

The process for setting up the MyWave platform for end users is summarised as:



Create organisation and set up users

Users with the SysAdmin role can create or edit organisations and add or edit users.

Organisations are the highest-level entities encompassing workspaces and their respective Waves.

Organisation

Create a new organisation

1. Click **...** next to User Settings on the left-hand sidebar.
2. Select **Organisations**
3. Select **Create New Organisation**.
4. Enter the name of the organisation.
5. Click **Save**.

Change the name of an organisation

1. Click **...** next to User Settings on the left-hand sidebar.
2. Select **Organisations**.
3. Click the edit icon next to the organisation you wish to rename.
4. Update the name of the organisation and click the **tick**.

Change selected organisation

1. Click **...** next to User Settings on the left-hand sidebar.
2. Select **Organisations**.
3. Click on the organisation name you wish to change to.
4. Click **Yes**.

Users

Users with the SysAdmin role can add and manage users in an organisation. Users can be members of multiple organisations.

Add a user to an organisation

1. Click **...** next to your organisation name on the left-hand sidebar.
2. Select **Users**.
3. Click **Add User**.
4. Enter the user's email address and select the roles to be assigned for the organisation.
5. Click **Save**.

Edit the roles of a user

1. Click **...** next to your organisation name on the left-hand sidebar.
2. Select **Users**.
3. Select the user to edit.
4. Select or deselect the roles to be assigned to the user for the organisation.
5. Click **Save**.

Remove a user

1. Click **...** next to your organisation name on the left-hand sidebar.
2. Select **Users**.
3. Click the Delete icon of the user to remove.
   Note: You cannot remove yourself from an organisation.
4. Click **Yes, delete** to confirm.

Creating InfoPlugins

Users with the Wave Developer role can create and deploy InfoPlugins and Repositories for use within Waves.

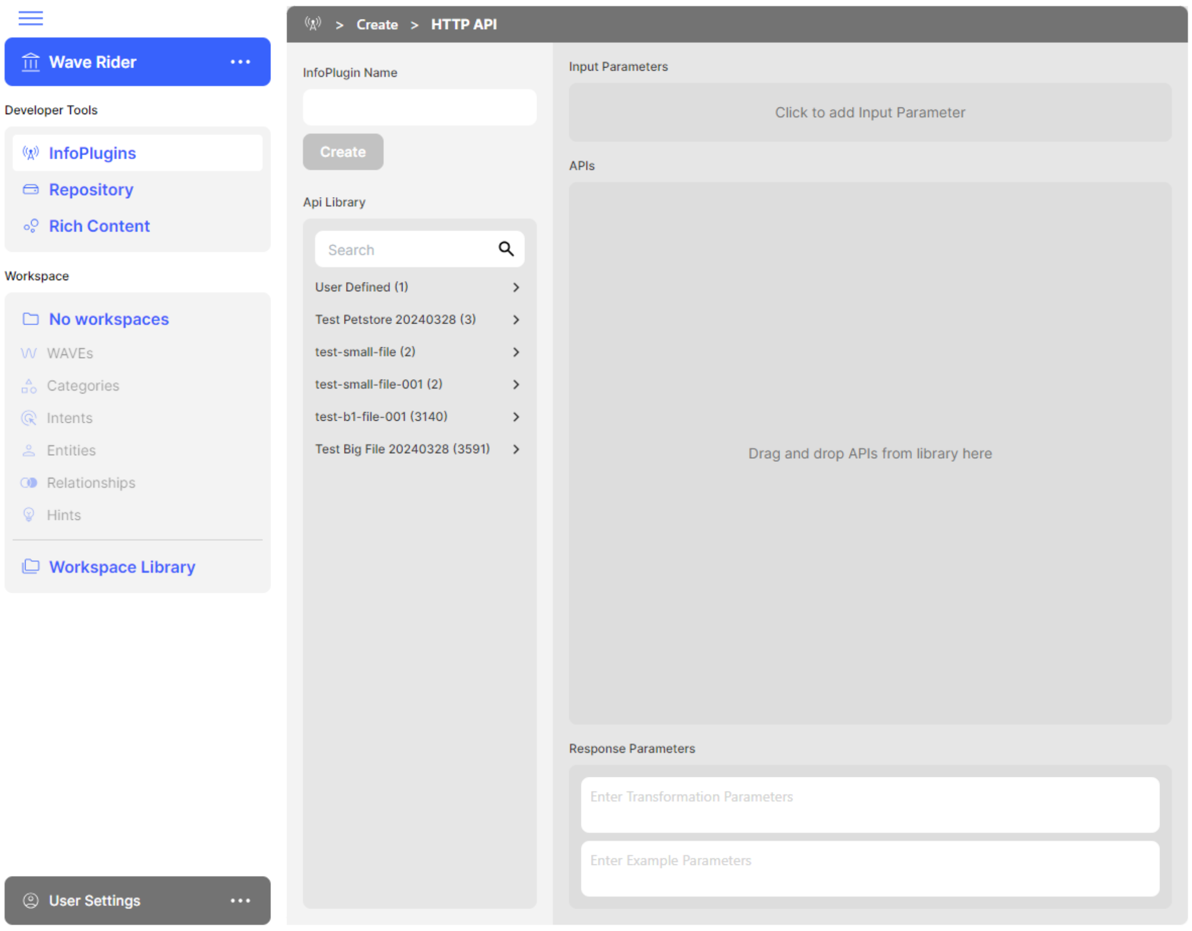
InfoPlugins serve as the integration point between MyWave and a third party, enabling the platform to orchestrate data across multiple systems. They can be considered a superset of APIs or a 'bridge' between MyWave and other platforms. One of the benefits of InfoPlugins lies in their ability to chain together different API connections.

For instance, the Post Sales Order InfoPlugin creates a sales order in SAP Business One based on the information collected from the user. In this case, the InfoPlugin connects MyWave and SAP Business One.

There are two main types of InfoPlugin, either HTTP API or Java based.

HTTP API InfoPlugin

HTTP API InfoPlugin allows you to connect MyWave with third-party platforms like SAP Business One and SAP S/4HANA. You select one or more APIs and then configure each of them as required.



Create a new InfoPlugin (HTTP API)

1. Select the **InfoPlugins** tab from the left-hand menu.
2. Click **Create New InfoPlugin**.
3. Select the HTTP API as the **Available Implementation** and click **Next**.
4. This will open the InfoPlugin Builder screen window, which consists of the following components:

Component	Description	Example
InfoPlugin Name	This is the name that will be used to access the InfoPlugin connection within MyWave **Note:** Assign a meaningful name as this will be the key for accessing the connection throughout the tool.	CloseSalesOpportunity
Input Parameters	These are the input values the Wave Master will need to provide to the InfoPlugin.	Name: SalesOpportunityID Type: String

Response parameters	These are the values that will be returned by the InfoPlugin. They can be used by the Wave Master in both the prompt and block of conditionals.	Transformation Parameters:
		{"itemCode": "${parameters.item_code}", "itemName": "${response.step1.productName}", "age": ${response.step2.age} }}
		Example Parameters:
		{"itemCode": "P123","itemName": "Leather Jacket", "age": 22 }

API Library	A list of all the available APIs, grouped by category.	Invoke_action_Close
Builder Section	Where the developer drags and drops API connections.	n/a

5. Enter an appropriate name for the InfoPlugin.
6. Enter any Inputs that the Wave Master will have to provide.
7. Select an API and drag and drop it into the 'Drag and drop' section. This will provide a template for that specific API to get you started.
8. Check and complete the following fields as necessary:

Component	Description	Example
baseUrl	The starting point for API requests, including the domain and optional subdirectories.	Direct: https://localhost/service-root Using a config file: ${config.innovation_hub_intergration_service.base_url}
path	The specific endpoint or resource within the API.	Direct: /SalesOpportunities{{SequentialNo}}/SAPB1.Close Using a config file: ${config.innovation_hub_intergration_service.path}
httpMethod	Specifies the action type (GET, POST, PUT, DELETE) for interacting with the server.	GET
headers	Additional information sent with the request, like	{"accept":"application/json", "Token token=${PLATFORM_ACCOUNT_TOKEN}"}

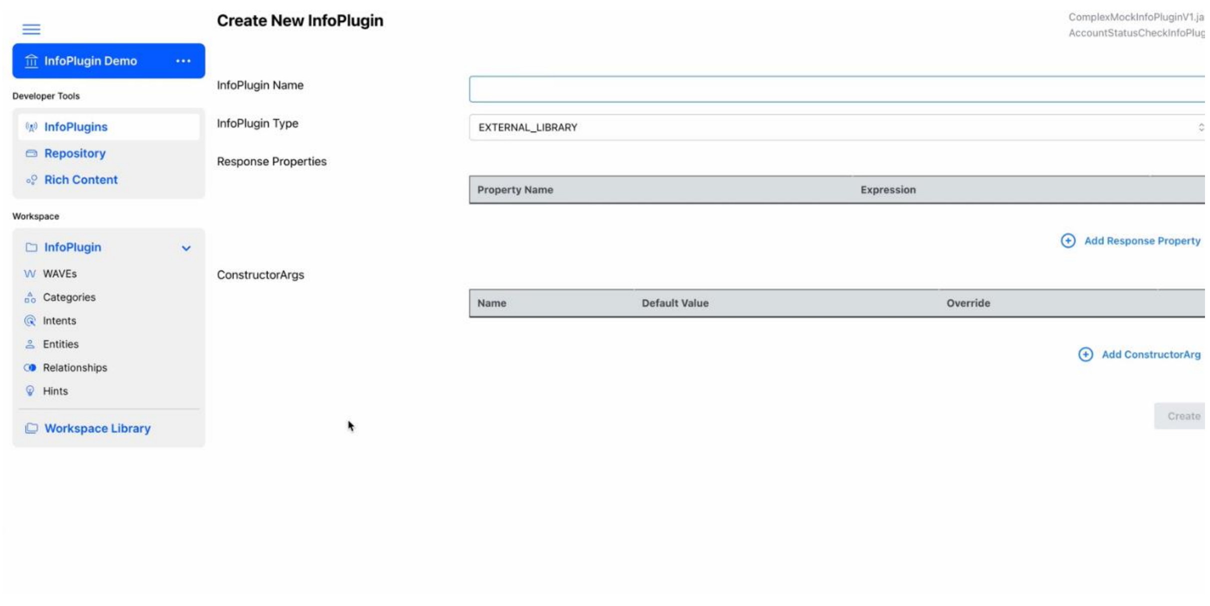
	authentication tokens or content type.	
queryBody	Parameters or data appended to the URL for filtering or specifying the request.	{“SequentialNo”:”${parameters. SalesOpportunityID}”
requestBody	Data sent to the server for operations like creating or updating resources. Always empty for GET requests.	{“SaleItem”: “table”, “SaleLocation”: "Oceania”}

9. Finally, hit the **Create** button.

Your InfoPlugin will now be accessible to Wave Masters when creating conversations. For more information, please see this section of this manual.

JAR InfoPlugin

JAR InfoPlugin (Java-based) allows you to upload a .jar file to MyWave. You configure an InfoPlugin externally and then upload it to MyWave using a .jar file.



Create New InfoPlugin

InfoPlugin Name

InfoPlugin Type: EXTERNAL_LIBRARY

Response Properties

Property Name	Expression

ConstructorArgs

Name	Default Value	Override

Create a new InfoPlugin (JAR, Java-based)

Unlike an HTTP API InfoPlugin, most of the work for the JAR InfoPlugin is done in the Java code of the JAR file itself before being uploaded to the Innovation Hub. Custom Java plugins provide developers with the flexibility to deliver tailored solutions beyond the scope of out-of-the-box InfoPlugins.

The Repository tab is where the .JAR file for a Java-based InfoPlugin is stored. Once the file has been added, you then need to configure the InfoPlugins themselves.

See the Developer Guide [TBC] for more information on creating custom Java plugins for use in the repository section.

1. Select Repository from the left-hand menu.
2. Click **Upload New File**.
3. Select the .jar file to upload (you can also drag and drop the file).
4. Select the InfoPlugins tab from the left-hand menu.
5. Click **Create New InfoPlugin**.
6. Select one of your newly added InfoPlugins from the **Available Implementations** drop-down list.
7. Complete the following fields as necessary:

Field	Description	Example
InfoPlugin Name	This is the name that will be used to access the InfoPlugin connection within MyWave. Note: Assign a meaningful name as this will be the key for accessing the connection throughout the tool.	CloseSalesOpportunity
InfoPlugin Type	There are three main types of InfoPlugin for Java plugins: External Library: This type of InfoPlugin involves a Java call that returns a single object with attributes, similar to the Response Properties example above. It is typically used in updateAnswerBinding and in true/false conditions within the conversation. External Library with List: In this type, the return consists of an array of single objects. It is typically used when you need to populate a multiple-choice list with data.	EXTERNAL_LIBRARY

	External Library Ignore: This type of InfoPlugin involves a Java call where you are not expecting to use any of the return from the info plugin. For instance, if you use an info plugin for sending emails, as long as it does not fail, you are not interested in processing anything returned from the email service.	
Response Properties	These are the values that will be returned by the InfoPlugin. They can be used by the Wave Master in both the prompt and block of conditionals.	Property Name: itemCode Expression: parameters.item_code
ConstructorArgs	Used to override ConstructorArgs (values used during the running of the InfoPlugin) which were set in the Java file.	Retry Timeout default value = 2 Override = 3

8. Repeat for each newly added InfoPlugin.

Once configured, these InfoPlugins will now be accessible to Wave Masters when creating conversations. For more information, please see this section of this manual.

Software Distribution

As a developer, you can build your own custom solutions using the MyWave SDK. Instructions on how to install the SDK into your development environment are available from the **Software Distribution** section.

Construct Process Waves

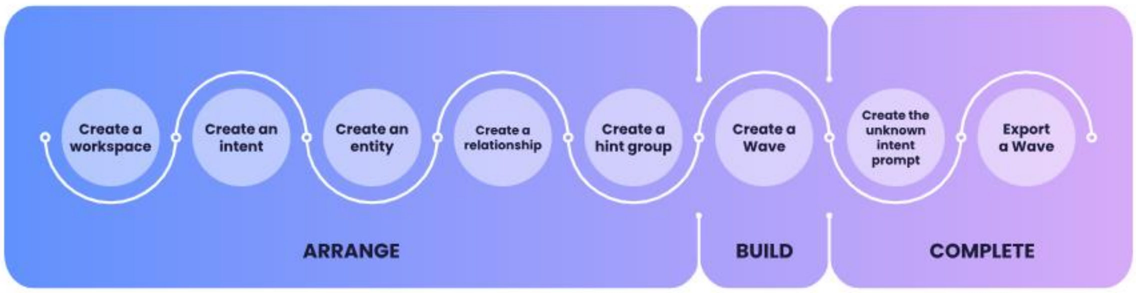
Users with the Wave Master role can create or edit configured Waves.

The process of creating a Wave consists of three stages:

Arrange is the first stage where you set up the key elements required to get started.

Build is the core stage where you create the Wave for your customer.

Complete is the final stage where you perform the last steps and export the Wave, so your customers can start using it.



There are several object types to set up and configure:

Object type	Purpose
Workspace	Encompasses all elements to include in a deployment. An organisation can have multiple workspaces.
Intent	An intent is the purpose or outcome of a Wave, e.g. create a sales order.
Entity	An entity is an object that the platform remembers and can refer to during Waves. There is always a Person entity that stores information about the current user, e.g. name, etc.
Relationship	A relationship is a way to connect entities together. For example, the Person entity could have a 'Someone I know' option and the user will be prompted to enter their name, making it available for future use.
Hint	A hint is a prompt text that displays on the platform so the user knows the types of questions or actions they can perform.
Category	Categories allow you to customise your Waves based on certain criteria. For example, a 'Clothing' category for 'item' can contain jeans, shoes, etc. Categories can also be used to help the platform recognise similar items, e.g. a 'TV' category could contain similar words such as television.
Wave Model	A Wave Model is a journey, starting with an intent triggered by the user or other system. It uses intents, entities, and relationships to identify and request information from the user or system so that the platform can execute that action.

Workspaces

Create a new workspace

1. Select **Workspace Library** from the left-hand menu.
2. Click **Create New Workspace**.
3. Enter the workspace name and click **Save**.

To change a workspace name, click the Edit icon of the workspace and update the name.

Intents

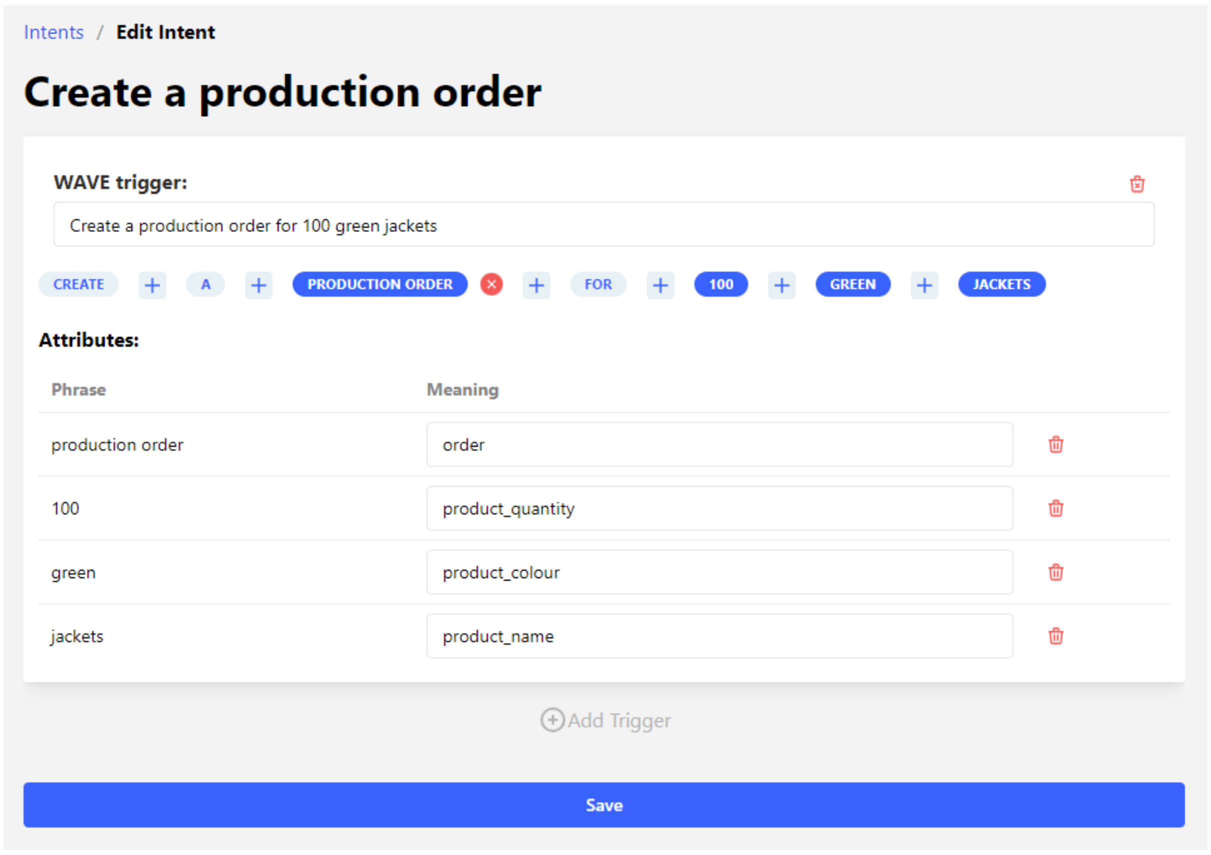
An intent refers to the purpose or desired outcome of a Wave process, such as creating a sales order or adding a timesheet item.

There are two methods to trigger Intents in the Innovation Hub:

1. Using a Large Language Model (LLM)
2. Using Natural Language Processing (NLP)

Both methods are configured in the same way within the Innovation Hub; however, LLM deployments require significantly less training data. For instructions on linking your LLM to the Waves, please refer to the Deployment documentation.

For more detailed insights into the differences between NLPs and LLMs in the context of this product, please consult: NLP vs LLM.



Create a production order for 100 green jackets

Phrase	Meaning
production order	order
100	product_quantity
green	product_colour
jackets	product_name

Create an intent structure

To create an intent:

1. Select your **Workspace** from the left-hand menu.
2. Select the **Intents** tab.
3. Click Add Intent.
4. Enter the intent name and click **Save**.

Note: if you are using LLM, you do not need to enter a WAVE trigger, so can stop at this step.

5. Enter a **Wave Trigger**, the sentence the platform looks for to recognise the intent. For example: 'I want to buy a green jacket'.
6. Select the key words in the entered sentence to set as identified attributes and enter the meaning. For example:
    - Green = Colour
    - Jacket = Item.
7. Click **Add Trigger** to add an additional trigger for this intent and repeat steps 6 and 7. For example: 'I want to buy a large green jacket'.
   Note: Words in the Wave process trigger can be combined to create a single phrase by clicking the plus icon.

For LLM-triggered implementations, only basic intents and categories are required. The LLM configuration file will be generated during deployment.

Create the unknown intent prompt

The unknown intent prompt displays to the user if they enter something that the platform cannot understand. Intents can be added to guide the user in the right direction from this output. Multiple intents can be added, and the user will be able to select what it is they were trying to do.

To create the unknown intent prompt:

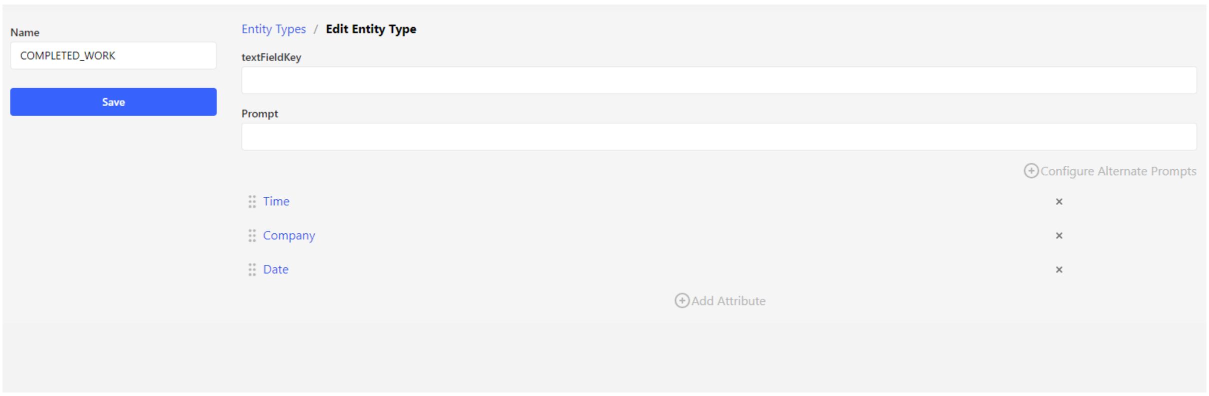
1. Select your **Workspace** from the left-hand menu.
2. Select **Workspace Library**.
3. Click **Export**.
   Result: Dialog displays, stating that the Unknown Intent Conversation has not been configured.
4. Click **Yes**.

5. Complete the following fields and click **Save**:

Field	Description	Example
Unknown Intent Prompt	The text presented to the user when they input something the platform doesn't understand.	I'm sorry, I didn't quite get that. Here's what I can help you with right now:
Description	The text presented to the user as an option for what to do next.	Buy a product
Intent	The intent linked with this option.	Product Purchase
Attribute	The attribute of the category linked with this option.	Item
Prompt	An optional field that serves as a prompt for instances where multiple Waves share the same intent. The user's response to this prompt informs the system about which specific Wave process it should use.	What do you want to buy? Possible answers could include 'a jacket', initiating the clothing Wave process, or 'a TV', triggering the electronics Wave process.

Note: To edit an existing unknown intent prompt, select **Waves**, then choose the **Unknown intent** Wave.

Entities



An entity is an object that the platform remembers and can refer to during Wave process execution. There is always a Person entity that stores information about the current user, e.g. name, etc.

Create a new entity

1. Select your **Workspace** from the left-hand menu.
2. Select the **Entities** tab.
3. Click **Add Entity**.

4. Complete the following fields and click **Save**:

Field	Description	Example
Name	The display name of the entity.	Client
textFieldKey	The internal name of the entity.	Client
Prompt	The question asked to the user to initiate the entity creation.	Sure, I can help you with that. May I please have the client's name?

5. Click **Configure Alternate Prompts** to add additional prompt questions for the entity.

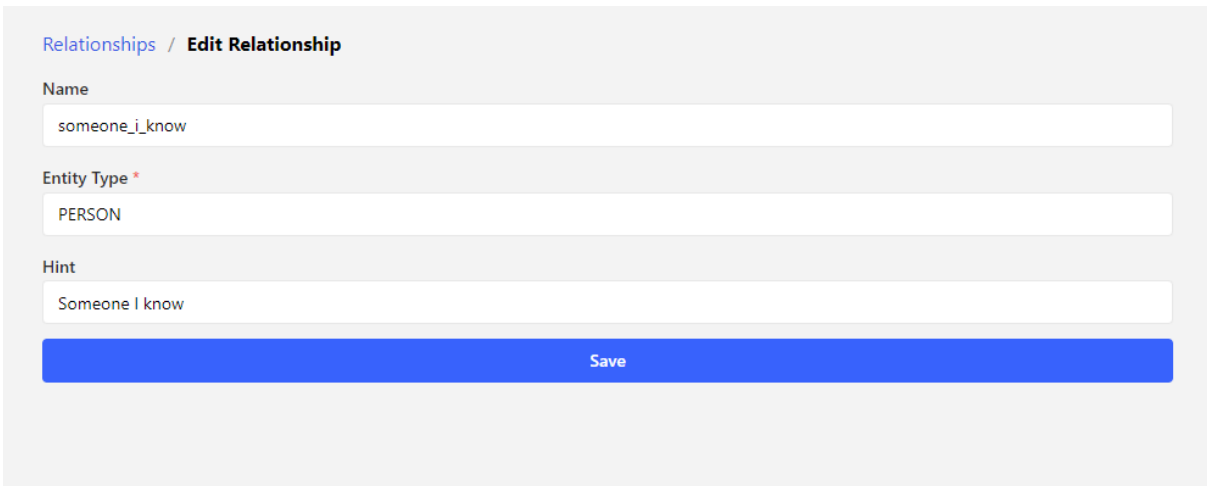
Entities have one or more attributes that store types of information about that entity. For example, the Person entity can have name, size, and gender attributes.

Add an attribute to an entity

1. From the Edit Entity Type screen, click Add Attribute.
2. Complete the following fields and click **Save**:

Field	Description	Example
Name	The attribute name.	Size
Label	The display name of the attribute.	Size
Requires Confirmation	If true, a designer can ask the user to confirm the value in a consent interaction. If false, it won't appear in user consent interactions.	True
Can Delete	If true, users can remove the entered value. If set to false, the platform won't allow deletion of this attribute's value from the entity. For instance, if a date of birth is set to not be deleted, it can't be removed once it's set.	True
Data Type	The type of the saved value.	String, Multiple Strings, Integer, Boolean
Options	Options available for the user to select from. **Note:** The first attribute in an entity cannot have options.	Small, Medium, Large

Relationships



Relationships / **Edit Relationship**

Name
someone_i_know

Entity Type *
PERSON

Hint
Someone I know

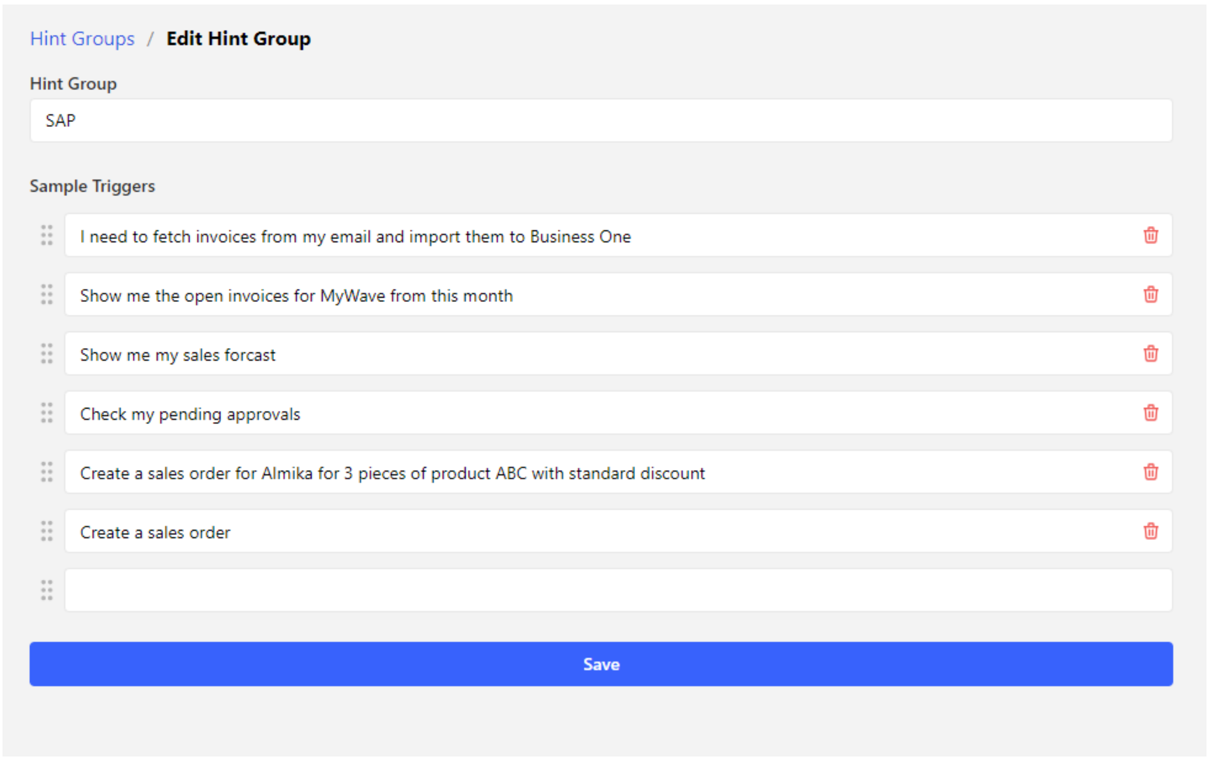
This is where you can set up methods to fill in entities. For instance, the PERSON entity could have a 'Someone I know' option. If this is selected, the user would be prompted for person's name and create a new entity for that individual, making it available for future use.

Create a new relationship

1. Select your **Workspace** from the left-hand menu.
2. Select the **Relationships** tab.
3. Click Add Relationship.
4. Complete the following fields and click **Save**:

Field	Description	Example
Name	The name of the relationship. **Note:** Do not use spaces.	someone_i_know
Entity Type	The type of entity the value is added to.	PERSON
Hint	The prompt text displayed to the user.	Add the name of the person you know.

Hints



Hint Group

SAP

Sample Triggers

I need to fetch invoices from my email and import them to Business One

Show me the open invoices for MyWave from this month

Show me my sales forcast

Check my pending approvals

Create a sales order for Almika for 3 pieces of product ABC with standard discount

Create a sales order

Hints are the 'suggestions' or options available to users when they access the MyWave User Interface. They serve as quick buttons that users can utilise to complete commonly accessed tasks.

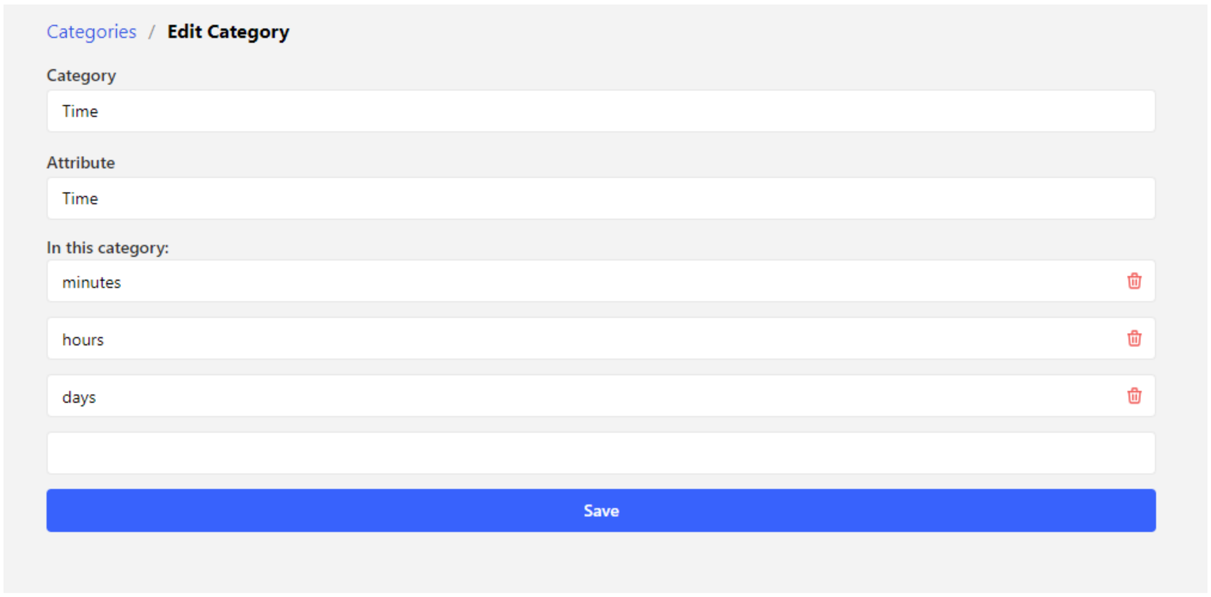
Hints can be autogenerated by clicking **Autogenerate Hints**. This takes the first trigger from each of the defined intents.

Create a Hint Group

1. Select your **Workspace** from the left-hand menu.
2. Select the **Hints** tab.
3. Click Add Hint Group.
4. Complete the following fields and click **Save**:

Field	**Description**	**Example**
Hint Group	The category for grouping hints together.	Product Purchase
Sample Triggers	The example triggers to display.	I want to buy a jacket.

Categories


Categories / **Edit Category**

Category	Time
Attribute	Time

In this category:

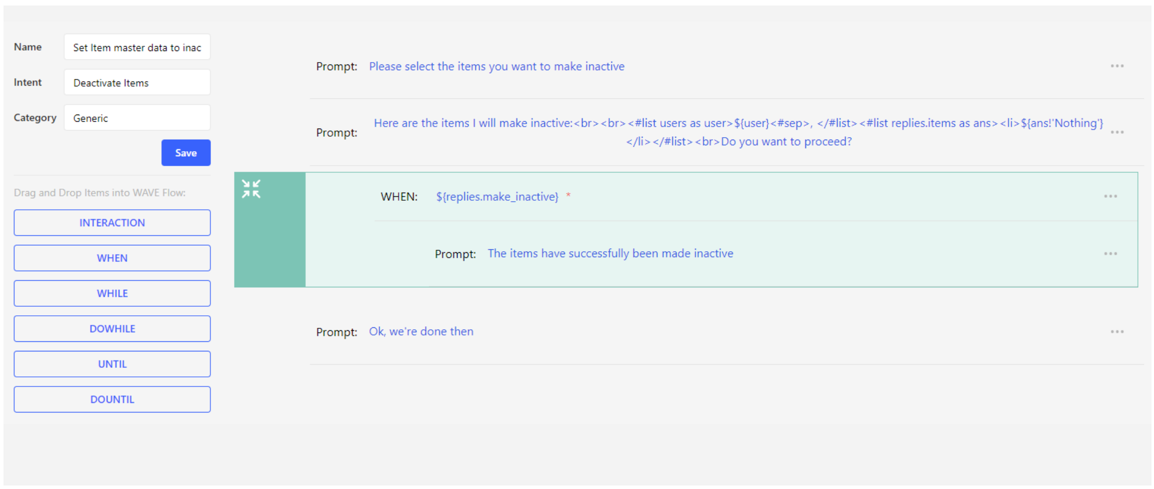
- minutes
- hours
- days

Create a new category

1. Select your **Workspace** from the left-hand menu.
2. Select the **Categories** tab.
3. Click **Create Category**.
4. Complete the following fields and click **Save**:

Field	Description	Example
Category	The category name.	Clothing
Attribute	The intent attribute associated with this category.	Item
In this category	The words within the category.	Hat, scarf, jacket

Wave Models


Name: Set Item master data to inac
Intent: Deactivate Items
Category: Generic

Prompt: Please select the items you want to make inactive

Prompt: Here are the items I will make inactive:

<#list users as user>${user}<#sep>, </#list><#list replies.items as ans><li>${ans!'Nothing'}</li></#list>
Do you want to proceed?

WHEN: ${replies.make_inactive} *

Prompt: The items have successfully been made inactive

Prompt: Ok, we're done then

A Wave is the journey for a selected Intent. It comprises one or more items, each representing a different prompt type. These items engage with the user, gathering information to fulfil a specific intent.

All Waves should start with a Pre-Filled My Cloud interaction, with an entity type of PERSON, to ask the person their name. This enables specific customisation for each individual user.

Create a Wave process

1. Select your **Workspace** from the left-hand menu.
2. Select the **WAVEs** tab.
3. Click **Create New Wave**.
4. Complete the following fields:

Field	Description	Example
Name	The name of the Wave	Show sales forecast
Intent	The intent associated with the conservation.	Display Sales Information
Category	The category associated with the Wave (if using NLP).	Sales Forecasts

5. Click the **Add an Item** icon and select **INTERACTION**. (after selecting your first item, you can drag and drop to add additional ones.
6. Select Pre-Fill My Cloud for the Interaction Type and ensure the Entity Type is PERSON.
7. Click **Save**.
8. Add additional items to the Wave process Model by clicking the **Add an Item** icon and selecting the relevant item type, see below.
9. Repeat step 8 for additional items.
   Hint: Click the Wave process name above the form to return to the list of items.
10. Click **Save**.

Interaction items

The Interaction item is the most used prompt type, used for engaging with users by asking questions or prompts to obtain responses. For example: 'What would you like to purchase?'

There are several interaction types, each with their own properties and behaviours.

Prompts

Prompts in interaction types serve as questions to ask the user. All interaction types, except Pre-Filled My Cloud, have a prompt.

Variables

In prompts, you can incorporate previously obtained information using variables.

${replies.variable_name} is used to access information already provided by the user within the Wave process. For example, if you had asked them for a size, it could be ${replies.size}.

${intentAttribute.variable_name} is used to access information within an intent associated with the Wave process. For instance, if you had asked them what they wanted to buy, and they chose a 'Jacket', ${intentAttribute.item} would be 'jacket'.

For example: 'Hi ${replies.user_name}, who do you want to buy the ${intentAttribute.item} for?'. Where:

- ${replies.user_name} is the user's name entered earlier in the Wave process (where the Pre-Filled My Cloud interaction type has been used with the PERSON entity)
- ${intentAttribute.item} is the identified attribute from the intent, e.g. 'I want to buy a **jacket**'.

You can also use {response.attribute} to access variables stored within an InfoPlugin.

Alternate prompts can be configured to introduce more variation in the prompts presented to the user. If provided, the system will randomly select one of the alternative prompts, enhancing the natural flow of the Wave Execution —especially beneficial when a user engages in similar Waves multiple times. Each alternate prompt should only differ in language; the intent of the prompt should remain consistent. For example, you might have "Hi, what is your name?" and "Hi, can you provide me with your name?".

Interaction types

There are seven interaction types available for different situations.

All interaction types are associated with a mood value, serving as an indicator to the front-end. This mood value can be used to customise the display, such as modifying the font color, for each interaction type based on its mood. The default value is "Calm," and additional setup is necessary on the front end to incorporate extra moods.

Pre-Filled My Cloud

The Pre-Filled My Cloud interaction type retrieves stored information from the user's MyWave. If the information has not been previously entered, the selected entity's prompt displays. For example: 'What is your name?'

Present

This interaction type is used to gather and store data within an entity for future Wave process items. For example, it can be utilised to inquire about the size of a jacket a user wants to purchase, with the platform recording this information against the currently selected entity option.

The present interaction type requires at least one field.

Field	Description	Example
Field	The name of the field.	applicant_name
Field Type	The input type of the field.	Text, Multiple choice, email, etc.
Label	The label visible to the user on the front-end.	Applicant name
Hint	The placeholder text the user sees.	Enter the applicant's name
Optional	Whether the field is required or not. **Note:** Only relevant if the interaction involves multiple fields.	True
Number of Fractional Digits	The number of decimal places to display. **Note:** Currency and Number field types only.	2
Default Value Expression	A dynamic way of setting the default value.	todays dates or ${replies.user_name}
Default Value	The default value for the field. **Note:** Currency and Number fields display Default Number Value. Does not display for Multiple Choice.	100

Min and Max Value	The minimum and maximum values possible. **Note:** Currency, Date and Number field types only (Date displays Min/Max Days)	0, 100
Validation Error Message	This error message appears if the user has not entered a value in a required field, or if the entry doesn't meet the field's criteria. **Note:** Not on Lookup or Multiple Choice fields.	Please enter an applicant's name to proceed.
Owner Entity	This refers to the specific option within the entity where the information will be recorded. If the platform is unaware, it will prompt the user.	person_to_buy_for
Pre Filled With	This indicates the item within the selected entity option that pre-fills this field if available.	Size
Application Strategy	Follow Cloud Definition: Utilises the setting selected under the entity item. Confirm Required: The chosen value will be displayed in any relevant user consent interactions, allowing the user to modify it through that interaction. Confirm Not Required: Skips this field if there is a pre-filled value. This also overrides the Confirm Required setting on the attribute in the entity definition. Force User Input: Presents an entry field, even if the user has previously entered a value on a prior date. If a value has been previously selected, it will populate this field, providing the option to make changes.	Confirm Required

Text field type specific fields

Field	Description	Example
Is Multi-Line	Is the text field a multi-line input field.	False
Case	This serves as an indicator for the front-end, specifying the case in which the input should be provided.	No Case Required
Restriction	A regular expression used to instruct and validate a user's input.	Check for credit card number: r'^\d{16}$'

Multiple Choice field type specific fields

Field	Description	Example
Enable selection of more than one option	Can the user select more than one option?	True
Enable user entered option	Can the user enter their own value?	False
User entered option label	The field label to display for user-entered options	Other
Options	The options to select from. Click **Add options** to add more.	Medium, Large

Lookup field type specific fields

The lookup field section is still a work in progress, so the table below may be subject to change. Please reach out to support@mywavesupport.zendesk.com if you have any questions.

Field	Description	Example
API Endpoint	The API endpoint is set to refence a previously created MyWave config file.	${config.getcustomers}
API Param	The value sent to the API which will be used for the lookup.	{customerName}
Minimum Number of Characters to Trigger Lookup	The API lookup will be sent once the required number of characters has been entered.	3
Waiting to Trigger Lookup Message	This message will be displayed until the user enters enough characters.	You need to enter at least 3 characters
Work in Progress	More information on this field to come	

Request User ID

This interaction type is utilised for user authentication on the front end. The user will be prompted to log in if not already authenticated. If already logged in, it is skipped.

Field	Description	Example
Unauthenticated Error Msg	This error message appears if the user is not authenticated to perform the task.	You must be authenticated to do that

Confirmation

This interaction type is used to ask the user a Yes/No (Boolean style) question.

Field	Description	Example
Label	The label visible to the user on the frontend.	Are you over 18 years old?
Key	The key used to access the answer in future prompts and conditions, assessed using variables: ${replies.person_to_buy_for}.	person_to_buy_for
Positive Answer	The label for a positive answer.	Yes
Negative Answer	The label for a negative answer.	No
Owner Entity	This refers to the specific option within the entity where the information will be recorded. If the platform is unaware, it will prompt the user.	person_to_buy_for
Pre Filled With	This indicates the item within the selected entity option that pre-fills this field if available.	Size
Application Strategy	Follow Cloud Definition: Utilises the setting selected under the entity item. Confirm Required: Shows the field with its pre-filled value, allowing the user to change it. Confirm Not Required: Skips this field if there is a pre-filled value. Force User Input: Displays the empty field even if it has a pre-filled value available.	Confirm Required

Conversation End

This interaction type is used to signal the end of the Wave process. For example: 'Thanks, ${replies.user_name}, we're all done.'

User Consent

This interaction type is used to confirm details with the user before completing an action (for example verifying information before submitting a purchase order).

The user will be presented with the saved values for each field listed under "Consent Fields" and given the option to either edit the values or confirm their correctness. If the user opts to modify a value, they will be redirected to the original question where they initially provided the information.

Field	Description	Example
Action	What will be happening after the consent – this value is just a guide, so it doesn't really matter what is entered.	
Consent Fields	These are the fields you want the user to confirm.	Size, colour

Entities Selection

The entities selection interaction type allows a user to store information about other entities so that they can be referred to later in the Wave. For example, someone they know. The end-user will be able to select either themselves (providing the entity type of the relationship in the Entities Selection is PERSON), someone that they know that the platform already knows about, or a new person.

Field	Description	Example
Label	The name of the entity selection, displayed for the Wave Generator in the back-end.	Someone I know
Key	The key used to access the answer in future prompts.	person_to_buy_for
Relationship	This establishes the connection between the provided input and an entity, managed in the relationships tab.	someone_i_know
Add New Entity Label	This labels the option the user selects to add a new entry.	Someone Else

InfoPlugin assigned values

InfoPlugins in MyWave serve as connectors that facilitate the integration of MyWave with other platforms. These connectors enable actions such as posting a sales order.

For example, to post a sales order on your final “Wave End” item you would:

1. Select the pre-imported InfoPlugin from the dropdown menu
2. Complete the table of pre-filled required parameters using the variables from the Wave or other locations.
   For example, the parameter salesOrder.user_name may be populated with the pre-obtained value ${replies.user_name}.

Field	Description	Example
Select InfoPlugin	The InfoPlugin to which the value will be assigned	Check Business Partner
Assigned Value	This is the name of the variable where you store the response of the info plugin, allowing you to use it in your Wave. It will appear in the list of ${replies.variable} once set.	busines_partner_check
Parameter	The key within the InfoPlugin to which the value will be assigned.	BusinessPartner.partner_name
Value	The value to be passed to the assigned key within the InfoPlugin.	Sysdoc

InfoPlugin Bindings

InfoPlugin bindings work like assigned values but do not store the sent values. Instead, only the final response is stored. This is useful for sensitive data, such as credit card numbers, where it's more secure to store only the response success token.

If you wish to update one or more values with information from the InfoPlugin, you should select 'Add Answer Bindings' and specify the fields you want to update, along with the corresponding response property from the InfoPlugin to replace them

Field	Description	Example
Select InfoPlugin	The InfoPlugin to which the value will be assigned	Check Business Partner
Parameter	The key within the InfoPlugin to which the value will be assigned.	BusinessPartner.partner_name
Value	The value to be passed to the assigned key within the InfoPlugin.	Sysdoc

Field	The field you wish to update with information from the InfoPlugin.	Company
Answer Binding	The response property from the InfoPlugin that will update the selected field.	BusinessPartner.partner_name

Condition items

The other types of items that can be added to a Wave process are conditions:

Condition	Description
When	Execute actions when certain conditions are met, based on saved variables or answers. Optionally, define actions for execution when those conditions aren't met. For example, determine if a store has the requested item in stock or not.
While	Execute actions continuously as long as certain conditions are met. For example, continuously poll an unreliable API until it becomes available.
DoWhile	Similar to the **While** condition, Similar to the While condition, but the conditions are evaluated after the actions are executed at least once. For example, continuously attempt to purchase a product while it's in stock, reassessing availability after each attempt.
Until	Continue executing actions until a specified condition becomes true. For example, keep polling the delivery status until it changes to 'Delivered'.
DoUntil	Similar to the Until condition, except the conditions are evaluated after the actions are executed at least once. For example, keep checking the delivery status until it changes to 'Delivered' after placing an order.

Condition expressions

Condition expressions are defined using FreeMarker. Variables are stated using the ${} operator.

Examples:

User role equals Approver:

- ${replies.user_role} == 'Approver'

User role does not equal approver:

- ${replies.user_role} != 'Approver'

A value is greater then 150, or is 0

- ${replies.price} > 150 || ${replies.price} == 0

A username value has been provided:

- ${replies.user_name}??

A reference code is uppercase and starts with an H

- ${intentAttribute.refrence_code}?upper_case && ${intentAttribute.refrence_code}?ensure_starts_with('H')

There are more conditions available. See the FreeMarker documentation for more information.

Deploying workspaces

After the Wave process models have been configured, they must be deployed to the platform so they can be used by end users.

Export workspace

1. Select your **Workspace** from the left-hand menu.
2. Go to the **Workspace Library** tab.
3. Click **Export**.
   Result: The workspace is exported and saved to your downloads folder.

Accounts

Change your password

1. Click **...** next to user settings on the left-hand menu
2. Select **Change Password**.
3. Enter your current password.
4. Enter your new password and confirmation.
5. Click **Save**.

User Details

The **User Details** section provides an overview of the account details linked to the currently logged-in user.

Other information

NLP vs LLM (in the context of this product)

Natural Language Processing (NLP)

NLP needs training for specific situations and struggles with different sentence structures or spelling errors. It works within set limits.

Large Language Models (LLM)

LLMs are more complex and can handle information presented in different ways. But there's a risk of misunderstanding, especially with nuanced language.