FLOWX.AI

BUILDING BLOCKS / Node / message-send-received-task-node

Contents

BUILDING BLOCKS / Node / Message send/Message received task nodes

Message send task and **message received**

The fallback content to display on prerendering
are used to handle the interaction between a running process and any external systems.

Message send task

This node is used to configure messages that should be sent to external systems.

Configuring a message send task node

Node configuration is done by accessing the **Node Config** tab. You have the following configuration options for a message send task node:

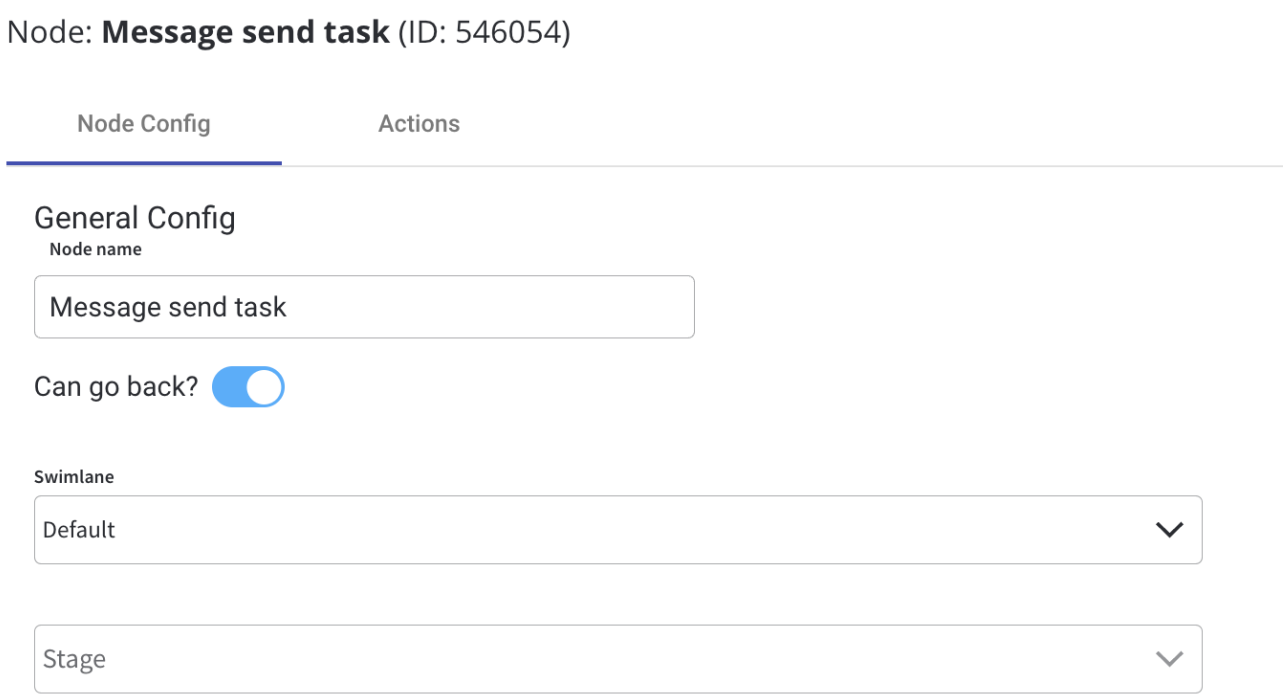
General Config

Inside the General Config you have the following properties:

- **Node name** - the name of the node
- **Can Go Back** - switching this option to true will allow users to return to this step after completing it

> **INFO**
>
> When encountering a step with `canGoBack` switched to false, all steps found behind it will become unavailable.

- **Swimlane** - choose a swimlane (if there are multiple swimlanes on the process) to make sure only certain user roles have access only for certain process nodes - if there are no multiple swimlanes, the value is **Default**
- **Stage** - assign a stage to the node

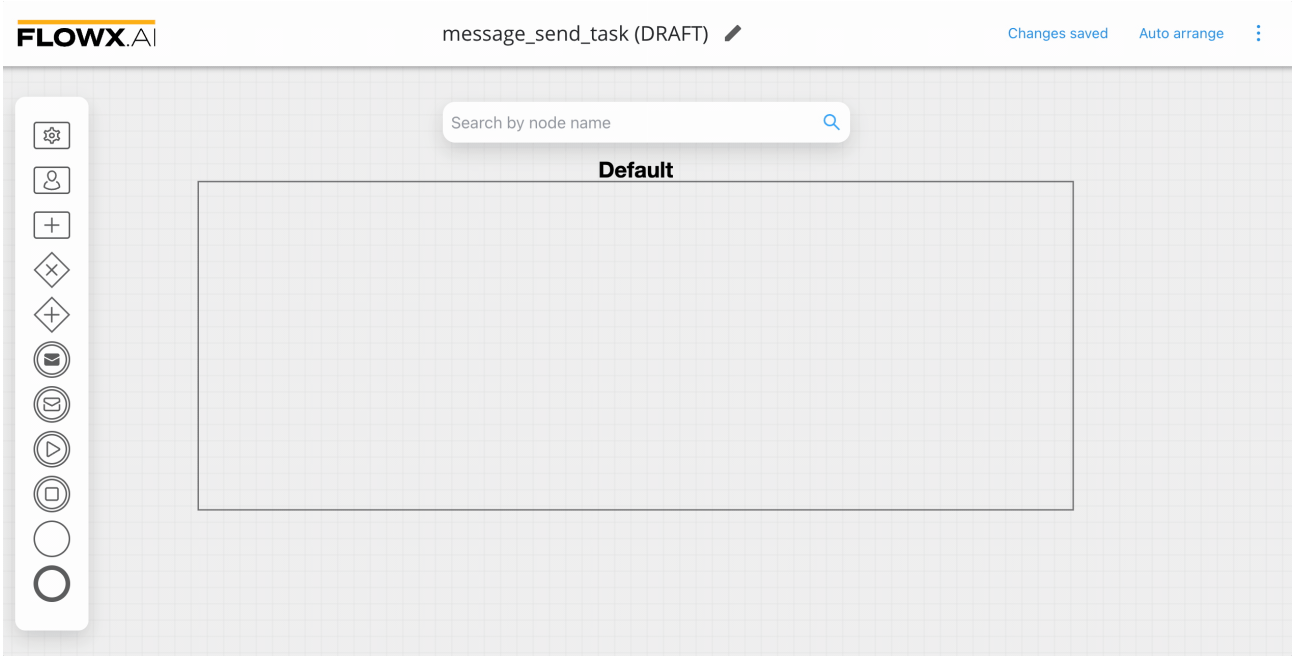


To configure a message send task node, we first need to add a new node and then configure an

The fallback content to display on prerendering
(**Kafka Send Action** type):

1. Open
   The fallback content to display on prerendering
   and start configuring a process.
2. Add a **message send task** node.
3. Select the **message send task** node and open **node configuration**.

4. Add an
   The fallback content to display on prerendering
   , the type of the action set to **Kafka send**.
5. ❗ A few action parameters will need to be filled in depending on the selected action type.



Multiple options are available for this type of action and can be configured via the FLOWX.AI Designer. To configure and add an action to a node, use the

The fallback content to display on prerendering
tab at the node level, which has the following configuration options:

- Action Edit
- Back in steps (for Manual actions)
- Parameters
- Data to send (for Manual actions)

Action Edit

- **Name** - used internally to make a distinction between different actions on nodes in the process. We recommend defining an action naming standard to be able to easily find the process actions
- **Order** - if multiple actions are defined on the same node, the running order should be set using this option
- **Timer expression** - it can be used if a delay is required on that action. The format used for this is ISO 8601 duration format (for example, a delay of 30 seconds will be set up as `PT30S`)
- **Action type** - should be set to **Kafka Send Action** for actions used to send messages to external systems
- **Trigger type** (options are Automatic/Manual) - choose if this action should be triggered automatically (when the process flow reaches this step) or manually (triggered by the user); in most use cases, this will be set to automatic
- **Required type** (options are Mandatory/Optional) - automatic actions can only be defined as mandatory. Manual actions can be defined as mandatory or optional.
- **Repeatable** - should be checked if the action can be triggered multiple times
- **Autorun Children** - when this is switched on, the child actions (the ones defined as mandatory and automatic) will run immediately after the execution of the parent action is finalized

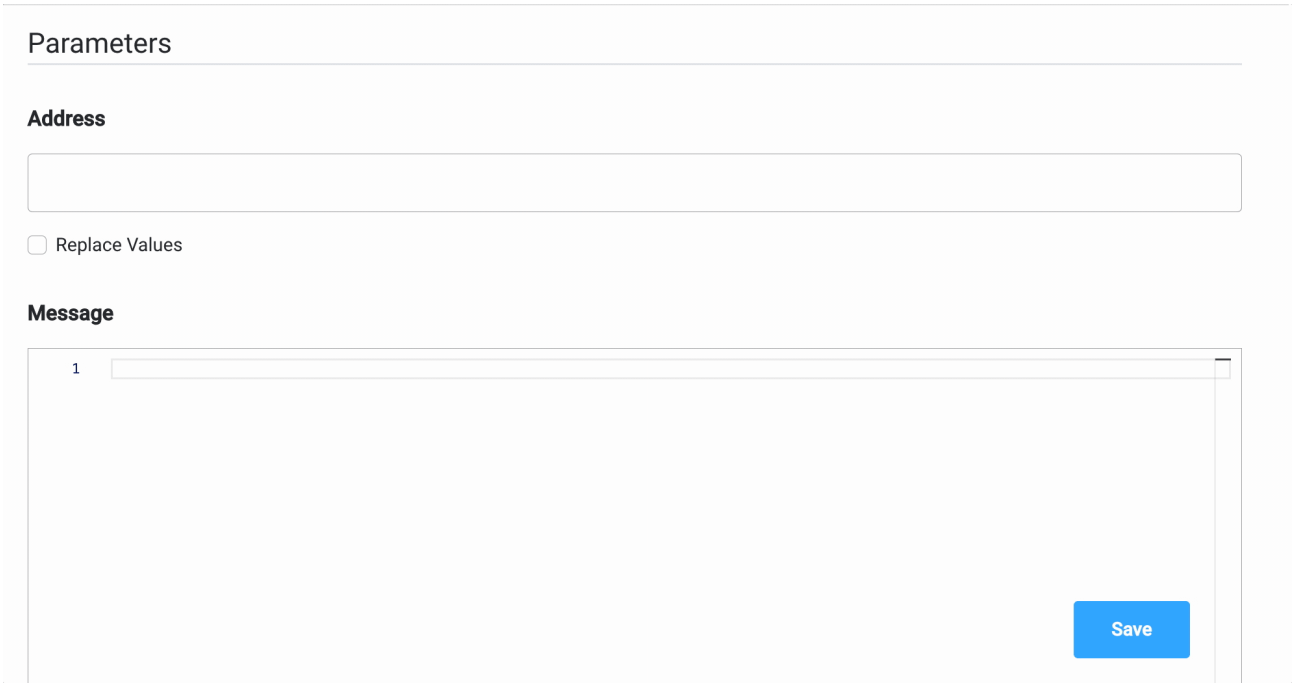
Back in steps

- **Allow BACK on this action** - back in process is a functionality that allows you to go back in a business process and redo a series of previous actions in the process, or more details, check **Moving a token backwards in a process** section

Data to send

- **Keys** - are used when data is sent from the frontend via an action to validate the data (you can find more information in the User Task configuration section)

> **DANGER**
>
> **Data to send** option is configurable only when the action **trigger type** is **Manual**.



For more information about what Kafka is, check the following sections:

» Intro to Kafka

» Kafka documentation

Example of a message send event

Send a message to a CRM integration to request a search in the local database:

Action Edit

- **Name** - pick a name that makes it easy to figure out what this action does, for example, `sendRequestToSearchClient`

- **Order** - 1
- **Timer Expression** - this remains empty if we want to action to be triggered as soon as the token reaches this node
- **Action type** - Kafka Send Action
- **Trigger type** - *Automatic* - to trigger this action automatically
- **Required type** - *Mandatory* - to make sure this action will be run before advancing to the next node
- **Repeatable** - false, it only needs to run once

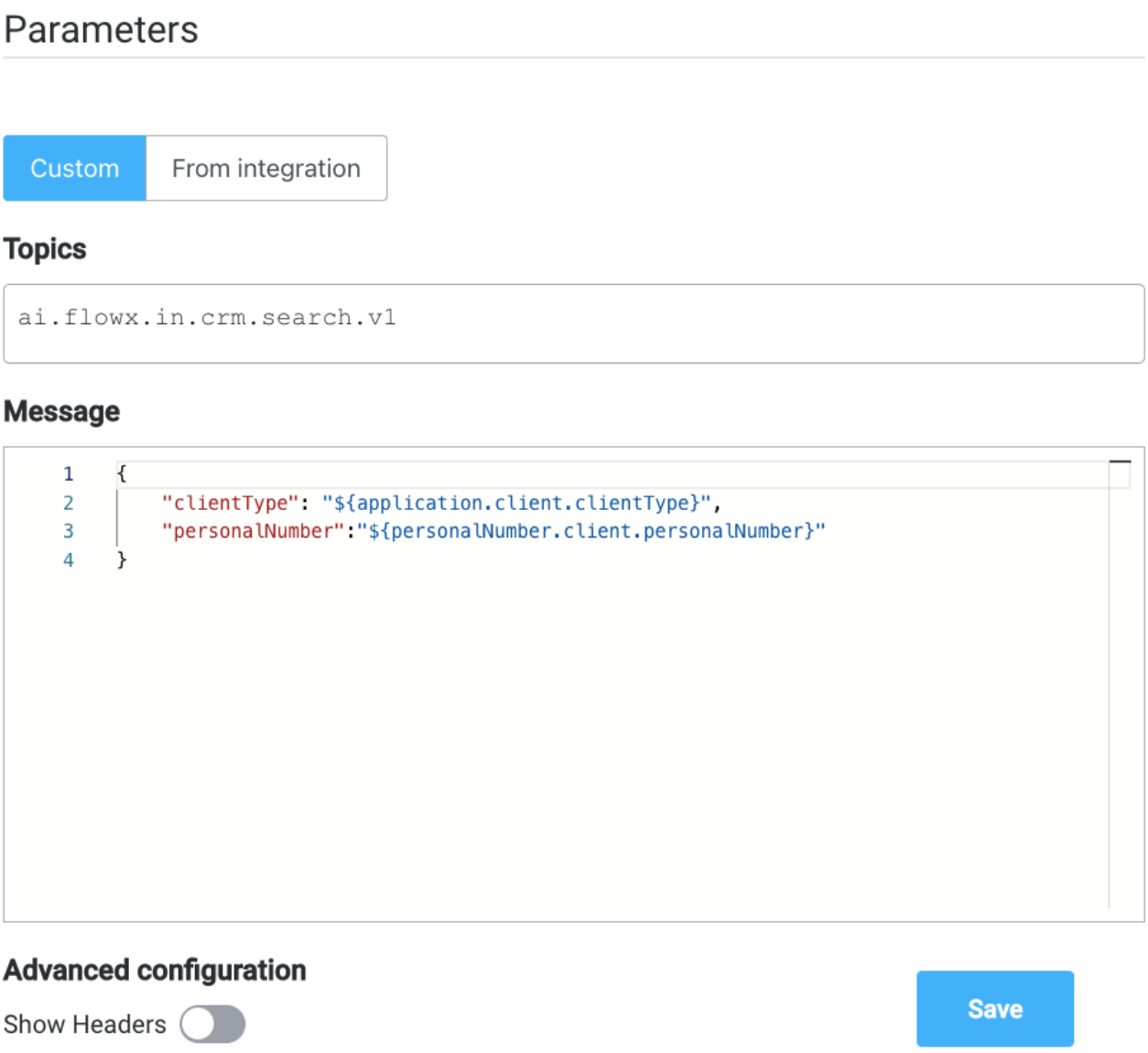
Parameters

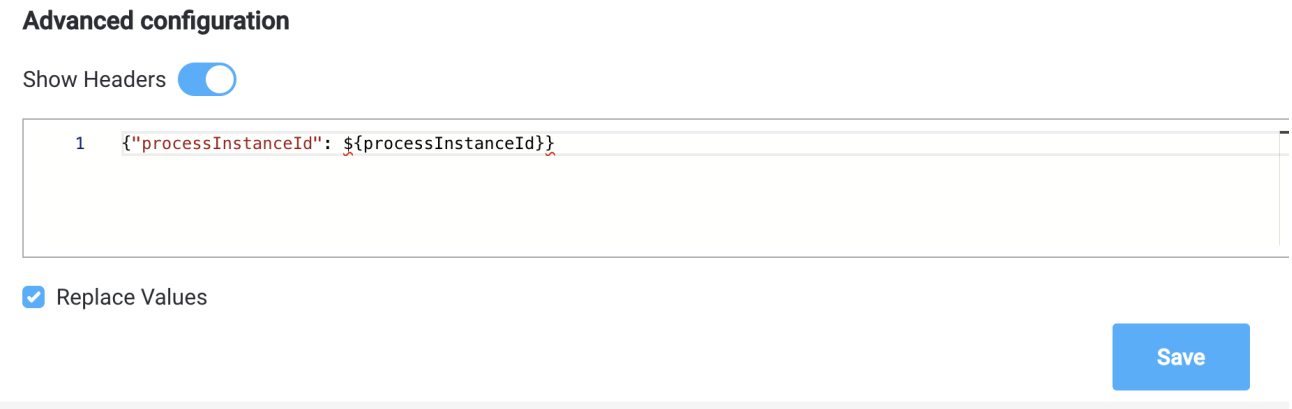
> **INFO**
>
> Parameters can be added either using **Custom** option (where you configure everything on the spot), or by using **From integration** and import parameters already defined in an integration.
>
> More details about **Integrations management** you can find here.

Custom

- **Topics** - `ai.flowx.in.crm.search.v1` the Kafka topic on which the CRM listens for requests
- **Message** - `{ "clientType": "${application.client.clientType}", "personalNumber": "${personalNumber.client.personalNumber}" }` - the message payload will have two keys, `clientType` and `personalNumber`, both with values from the process instance
- **Headers** - `{"processInstanceId": ${processInstanceId}}`


Action Edit
ID: 540663
Name
sendRequestToSearchClient
Order
1
Timer Expression
Kafka Send Action
Automatic
Manual
Mandatory
Optional
Repeatable


Parameters
Custom
From integration
Topics
ai.flowx.in.crm.search.v1
Message
{
"clientType": "${application.client.clientType}",
"personalNumber":"${personalNumber.client.personalNumber}"
}
Advanced configuration
Show Headers
Save

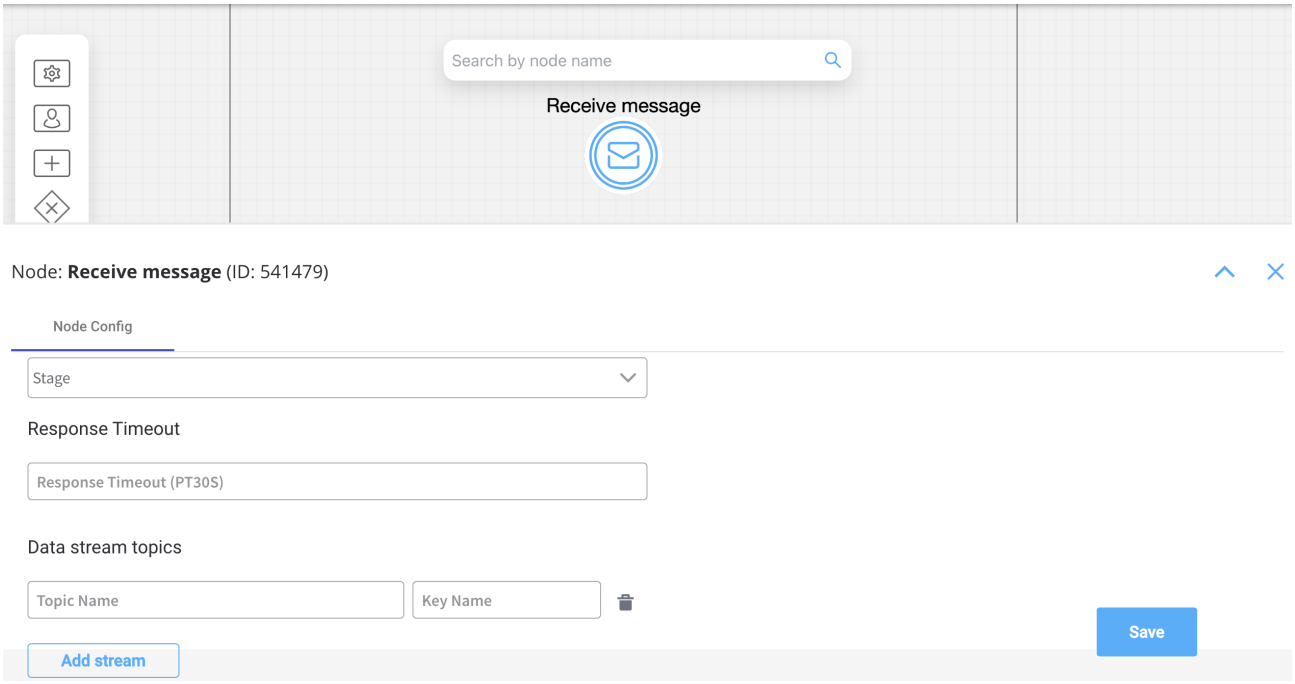


Message receive task

This type of node is used when we need to wait for a reply from an external system.

The reply from the external system will be saved in the process instance values, on a specified key. If the message needs to be processed at a later time, a timeout can be set using the ISO 8601 format.

For example, let's think about a CRM microservice that waits to receive requests to look for a user in a database. It will send back the response when a topic is configured to listen for the response.



Configuring a message receive task node

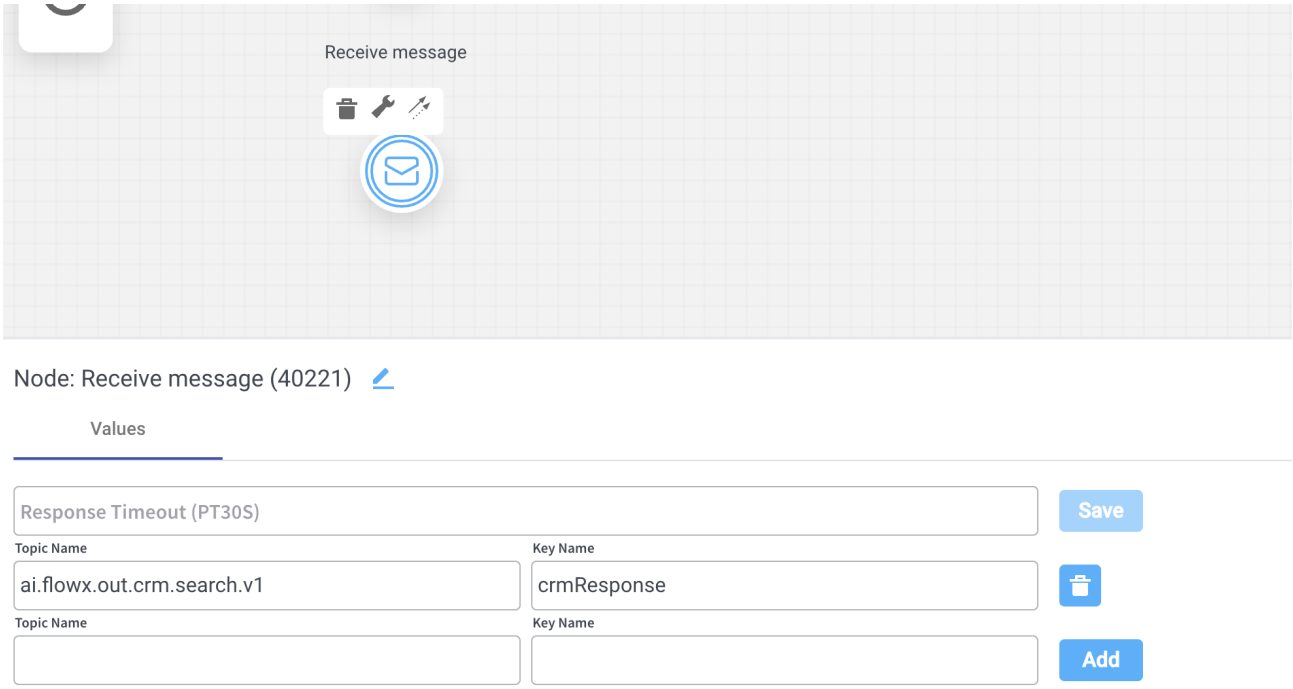
The values you need to configure for this node are the following:

- **Topic name** - the topic name where the process engine listens for the response (this should be added to the platform and match the topic naming rule for the engine to listen to it) - `ai.flowx.out.crm.search.v1`

> **DANGER**
>
> A naming pattern must be defined on the process engine to use the defined topics. It is important to know that all the events that start with a configured pattern will be consumed by the Engine. For example, `KAFKA_TOPIC_PATTERN` is the topic name pattern that the Engine listens to for incoming Kafka events.

- **Key Name** - will hold the result received from the external system, if the key already exists in the process values, it will be overwritten - `crmResponse`

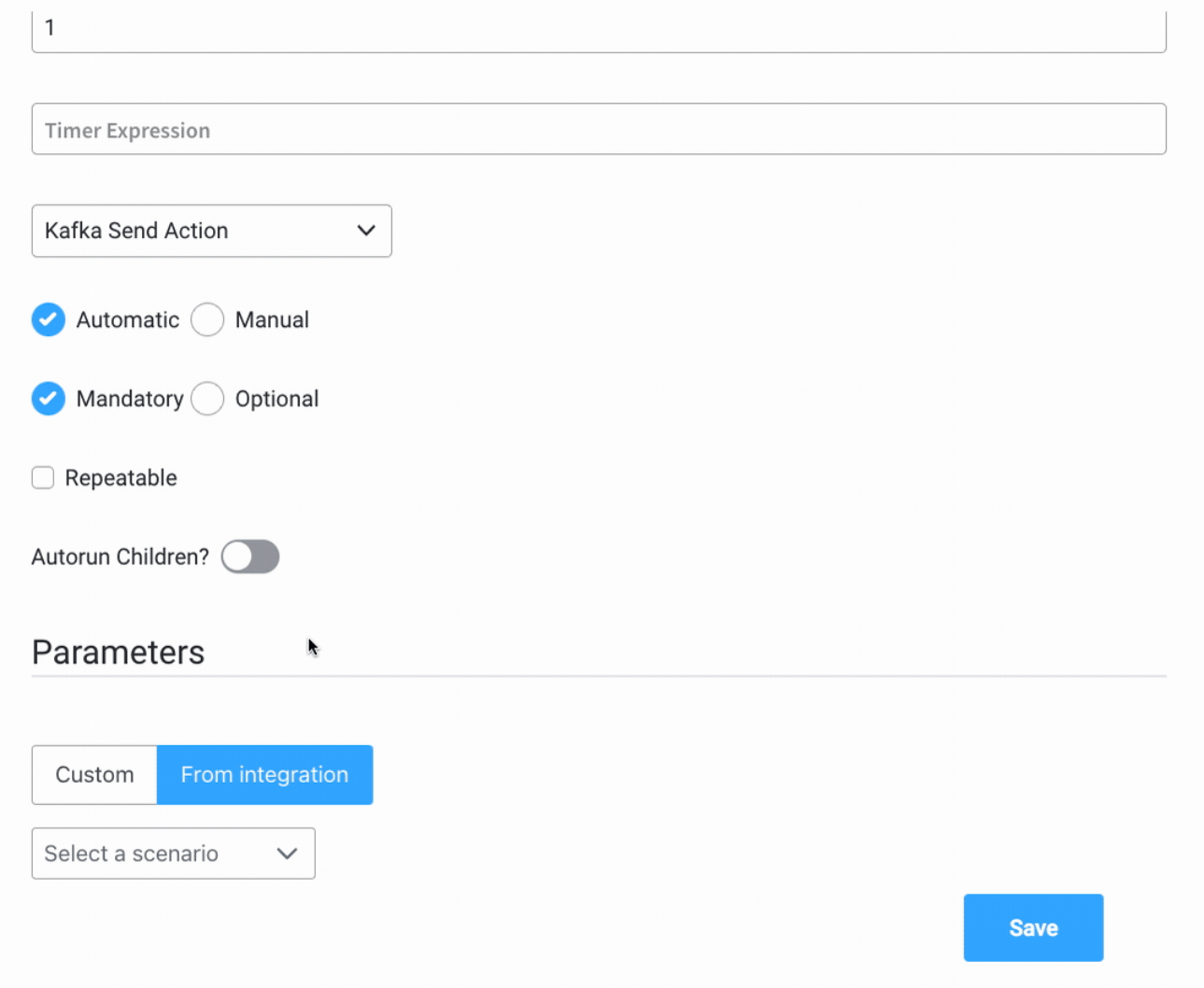
For more information about Kafka configuration, click here.



From integration

After defining one integration (inside Integration management) you can open a compatible node and start using already defined integrations.

- **Topics** - topics defined in your integration
- **Message** - the **Message data model** from your integration
- **Headers** - all integrations have `processInstanceId` as a default header parameter, add any other relevant parameters



Was this page helpful?