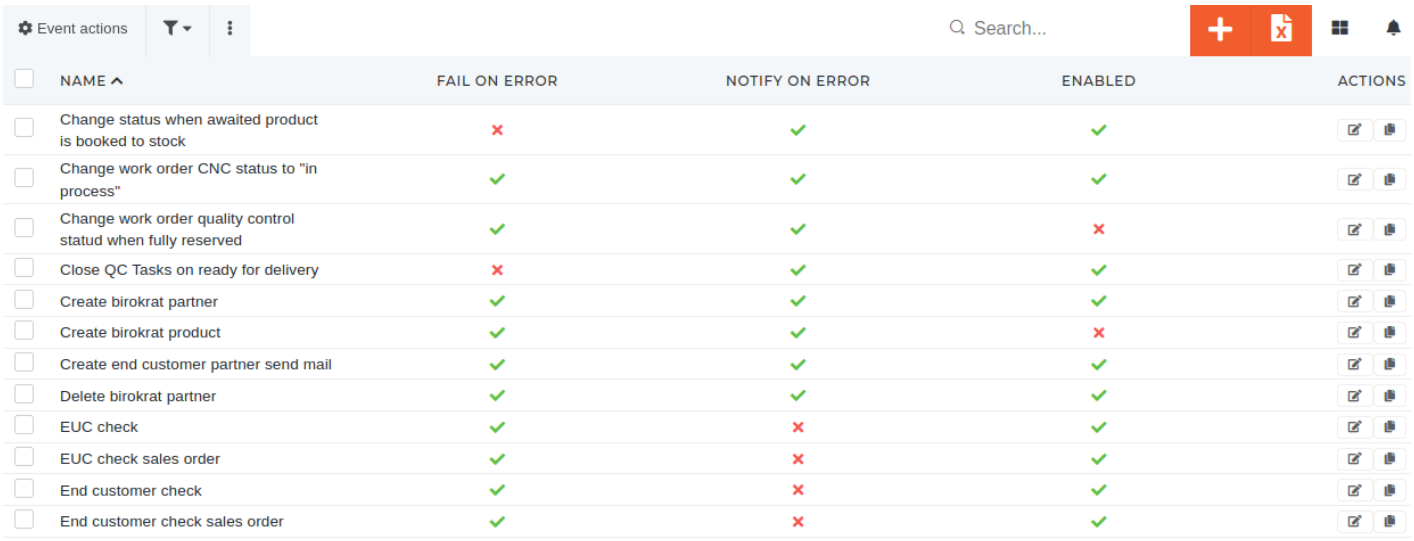
Event actions

- Overview
- Create new event action
- Update existing event action
- Delete event action

Overview

Event actions are automated actions, performed by the system, when a set of conditions is met or a certain event is triggered (updating of a product, changing an order's status, creating a new invoice...).



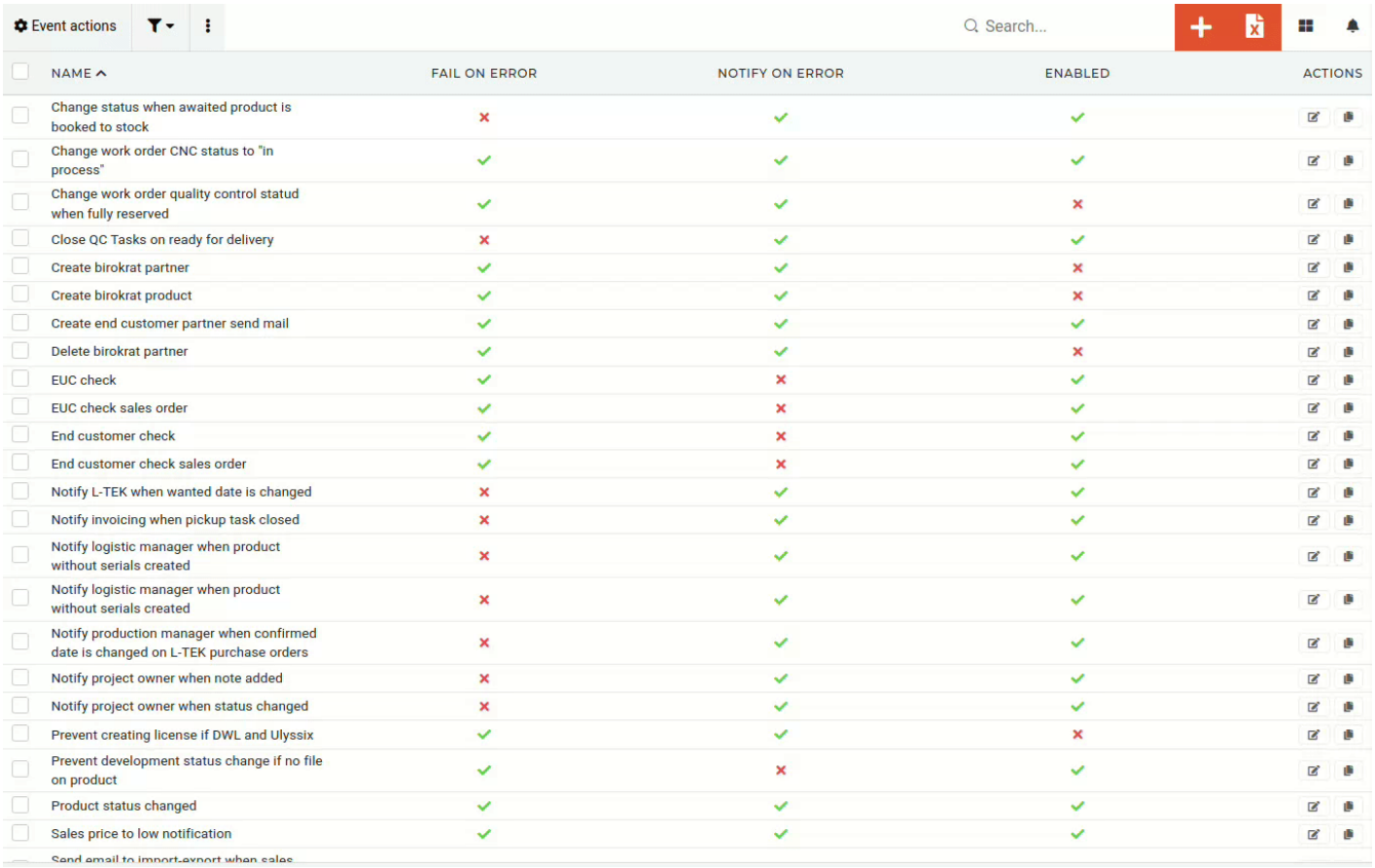
> Event actions are used for a variety of automated tasks, such as sending email notifications, updating statuses, performing hook calls...

Event actions are managed by ERP administrators.

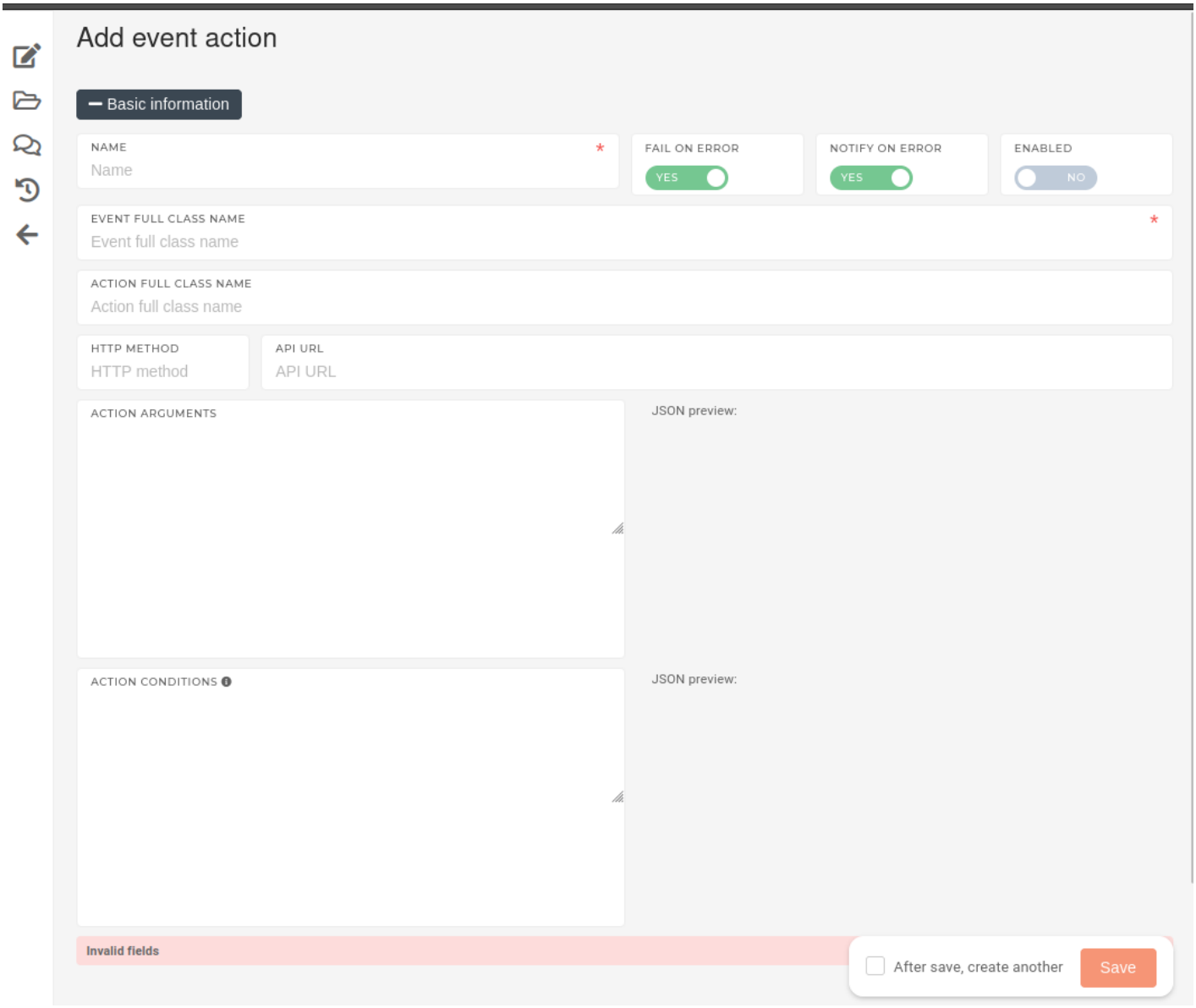
Create new event action

> This action requires the **"create event action"** permission.

To add a new event action, navigate to the "**Add new action**" form, by pressing the "**+**" button in the top right corner above the table.



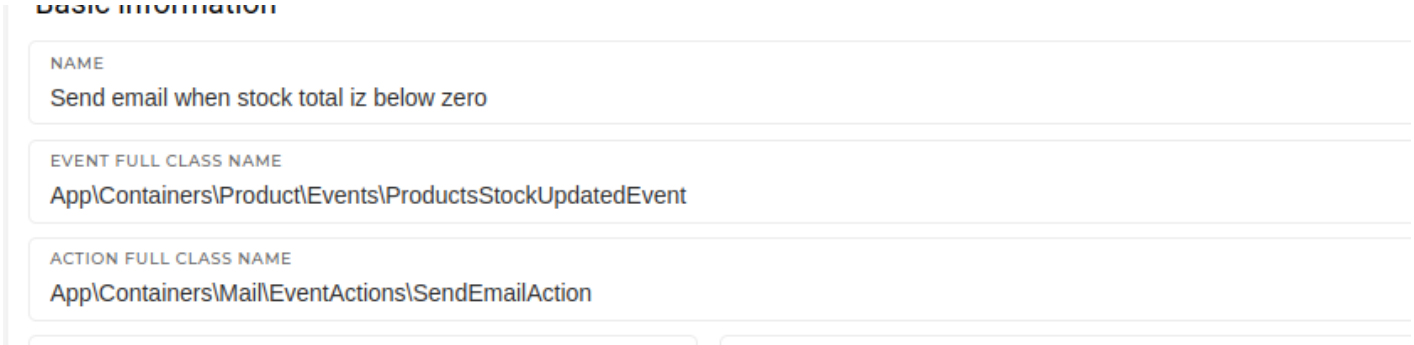
> Event actions should only be added by administrators with access to the code base, due to class names and their full paths being mandatory when creating new event actions. That information is only available to developers.



The creation form consists of several fields, required for creation of a new event action:

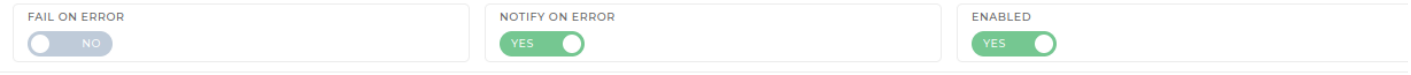
- **Name:**
  - The name of an event actions should be **simple and descriptive,** making it easy to figure out an event action's function from name alone (eg. *Send email to user when new order is created*).

With the name set, the event that triggers the event action and the action that is performed by the event action need to be defined.



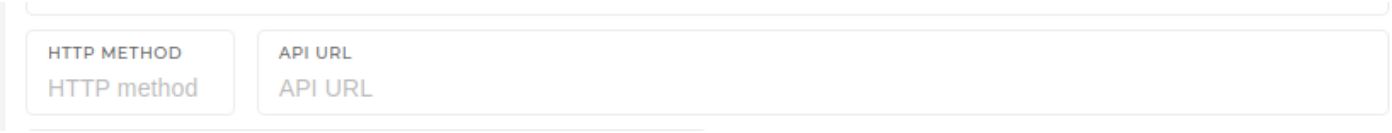
- **Event full class name**:
  - A full path (namespace and class name) of the Event class that needs to be triggered to perform the event action (eg. *App\Containers\Product\Events\ProductsStockUpdatedEvent*).
- **Action full class name**:
  - A full path (namespace and class name) of the Action class that will run once the Event is triggered. (eg. *App\Containers\Mail\EventActions\SendEmailAction*).

With the name, event and action defined, the three top switches can be toggled on/off.



- **Enabled:**
  - Toggle an event action as active or inactive. Inactive event actions will never run.
- **Fail in error:**
  - When this switch is toggled on, the action will be performed as normal, but will stop **immediately** on encountering an error.
- **Notify on error:**
  - When this switch is toggled on, whenever the event action encounters an error, an email is sent to the development team to inform them of a failure when performing the event action.

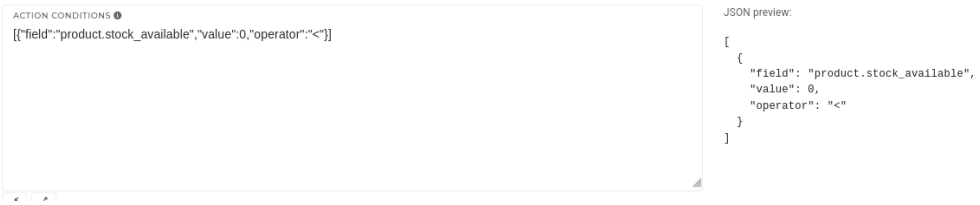
> **Event actions** can also be used to **call external services** via **API** calls when conditions are met.



- **HTTP Method**:
  - The method of the API call. Valid call methods include **POST, GET, PUT, PATCH, and DELETE**.
- **API URL:**
  - The URL of the external service API that will be called. Should be in format: **https://[service]/[endpoint]**

> **Arguments** and **conditions** (currently) need to be written in **JSON format**.

- **ARGUMENTS**:
  - Available arguments depend on selected **"action class".** This data can be provided by people with access to the codebase.
  - eg. **SendEmailAction** class requires:
    - **"to":** the email that will receive the notification email
    - **"html":** the content of the email in HTML
    - **"subject":** the subject of the email to be sent



- **CONDITIONS**:
  - Available conditions depend on selected **"event class".** This data can be provided by people with access to the codebase.
  - **The example in the photo checks if a product's stock_total is lower than 0**
  - eg. The **ProductsStockUpdatedEvent** class has access to the product object, so we can build conditions with fields of a product, eg. product's **"stock_total"**.
    - We can set a condition by making a JSON object that includes product's:
      - **"field":** Any field of a product model record (sku, name, short_name, id, ...)
      - "**value**": The value that the field will be compared to
      - "**operator**": The operation to run (the way to compare the value and the field)

AVAILABLE CONDITION OPERATORS:

"=" or "==="	Check if the field and value are the same. (strict comparison by value and by type)
"=="	Check if the field and value are the same.
"!=="	Check if the field and value are not the same. (strict comparison by value and by type)
"!="	Check if the field and value are not the same.
"<"	Check if the field is less than the value.
">"	Check if the field is mode than the value.
"<="	Check if the field is less than or equal to the value.
">="	Check if the field is more than or equal to the value.

IN	Check if field is one of values in array. Value must be array here.
NOT IN	Check if field is not one of values in array. Value must be array here.
CONTAINS	Check if field contains the value.
DOES_NOT_CONTAIN	Check if field does no contains the value.
IS_TABLE	Check if the record's table is equal to value.
STARTS_WITH	Check if the field starts with value. Field and value must be strings in this case
ENDS_WITH	Check if the field end with value. Field and value must be strings in this case

Update existing event action

Existing event actions should currently never be updated by anyone else but the developers. When a need arises for a different functionality, a new event action should be created, deactivating the old one.

Delete event action

Existing event action should never be deleted, instead being deactivated.