README - Flokzu in General

danger on making flow changes

create different versions for the changes and disable the automatically migration of instances.
old instances should remain on old versions while only new instances should run on the new deployed version

general tips

at first roll-out - assign all tasks to self (as administrator) and user

BECAUSE if there are tasks you want to be able to take action in
then you should assign task to self **as well**
(otherwise only viewing rights)

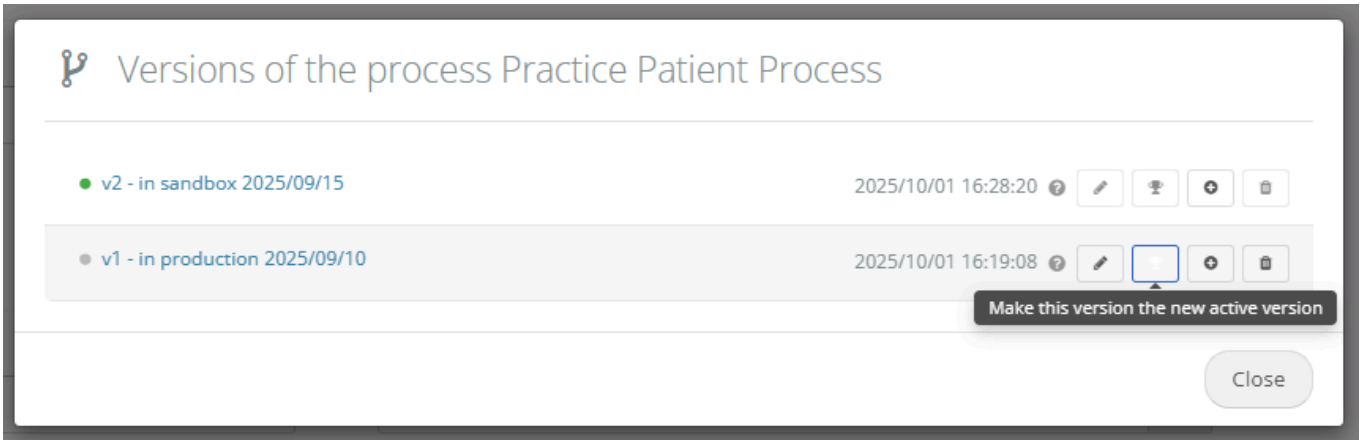
EXAMPLE CASE
LLM generates summaries from one (or field) form fields (and populates text field) *in backend*
if results now visible at very next tasks may need to make manual corrections to resulting field (in which case crucial that said task is editable) till LLM prompt refined may need to make alterations

when deploying new version

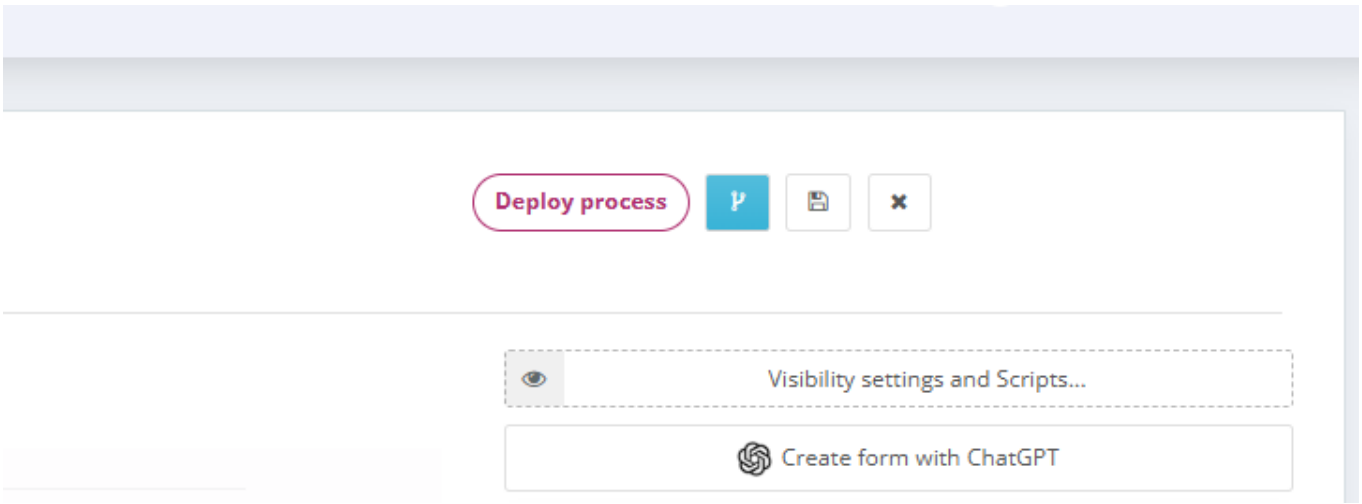
during staggered deployment (with multiple versions)

create tasks eg "INTERAL end of deployment v{ } phase { }" at desired stages of v1 assigned to development team
when flow then duplicated to configure next version "blockers" present (update name v{ +1})
this way able to manage various stages **and** versions of deployment

step one:



step two:



actually also then save and deploy again

warning on changing tasks names with script / visibility changes

× **do NOT do this**

unlike form fields where name can be altered but "identity" retained (info on how system handles change in field name)
a different task name is equivalent to different task

if instances already exists in task and scripts / visibility changed
then this affects those tasks

BUT

if task name is changed
then changes to scripts and/or fields visibility do **not** carry over

(ie. you are stuck with instances behaving as they did with previously named tasks - ito script and fields)

this is especially problematic for new scripts or form fields
and devastating if new form fields are used in flow (at gateways)
will have to introduce an additional tasks directly afterwards where this new field can then be set correctly (since wont appear in renamed task)

✓ when to do this

1. as long as no changes in script / fields also introduced
2. no instances existing in task (or will be created while in updates development)

NOTE - not tested

?! possible way around it

if instances exists in task where name and function (script and/or visibility) have to be updated

---

1. add script and/or change field visibility to tasks as is
2. use real-time report to check that new script and/or field visibility is "applied" to existing tasks (instnaces in tasks)
3. if so then go change name

---

this way the instances already existing in tasks (with old name) will have new script and/or field visibility applied (albeit no change in task name)
and any other instances moving into said task after latest deployment will have updated taks name, script and/or field visibility

warning when using "end-of-process" items

Common Errors

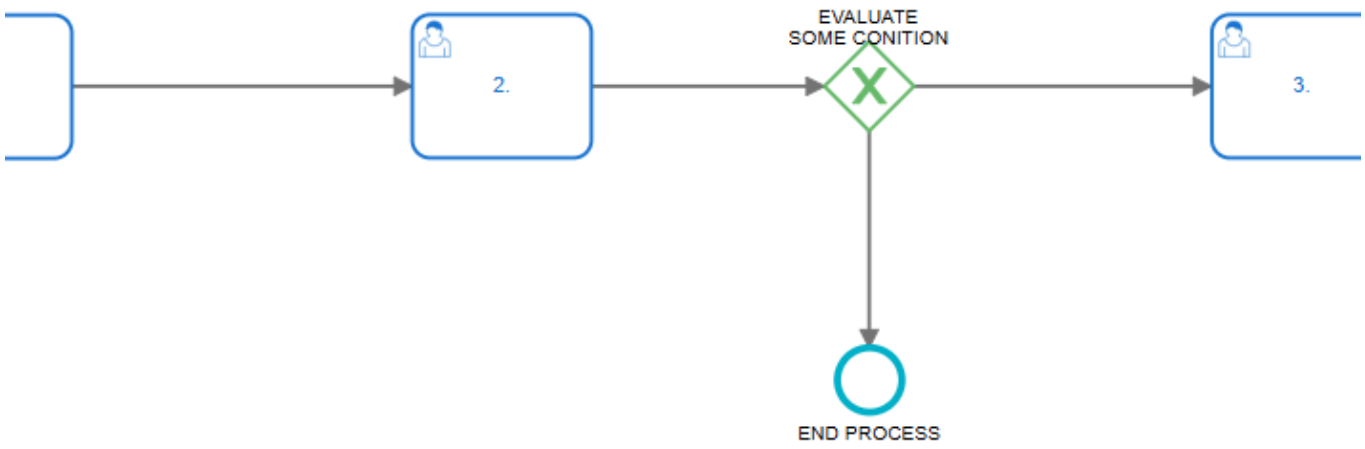
× NB end of process only appears when last of its tokens reach end-of-process

and then in real-time will reflect name of **last** end of process completed

NOTICE:
those conditions are usually form fields
so could be overcome by deleting said field but then all the data from that field for previous instances also lost

with exclusive gateway

here there is a decision after task 2. where if some condition is met the instance moves to task 3. or END OF PROCESS



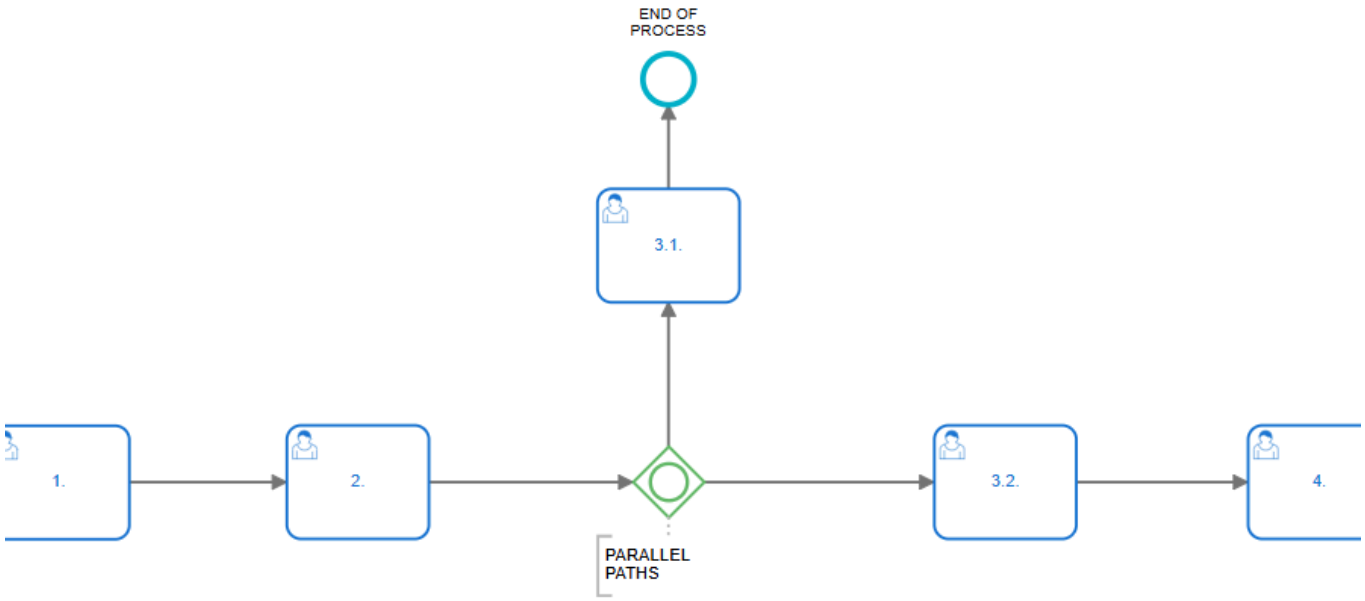
lets say there some instances (called FLOW-1, FLOW-2) where now created with this flow (and lying in task 1. or 2.)
then the flow is updated as follows (exclusive gateway removed)

when those instances FLOW-1, FLOW-2 are completed in task 2., if they meet the previous condition they move to “END OF PROCESS” instead of all moving 3.

on the other hand, any instances created after the update is complete (deployed), the flow will follow 1. → 2. → 3. as expected

with inclusive gateway

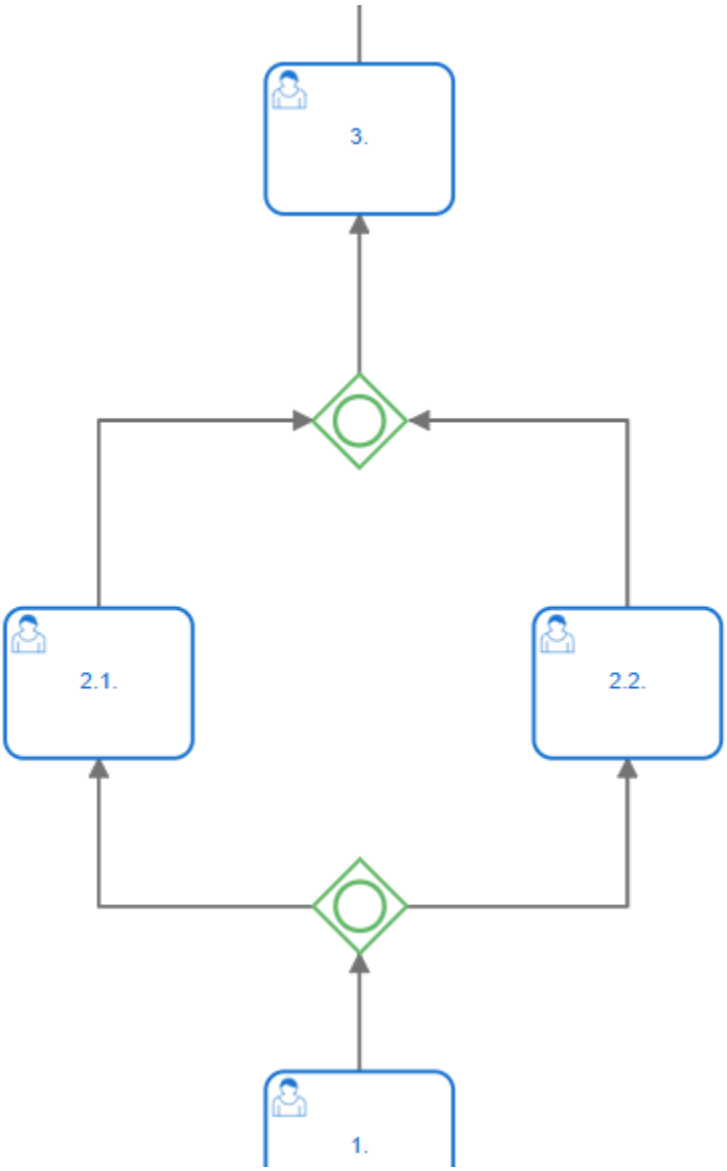
here the instances that move past task 2. split to task 3.1 and 3.2., where 3.1. runs to a END OF PROCESS and 3.2. continues with flow to 4. and so forth



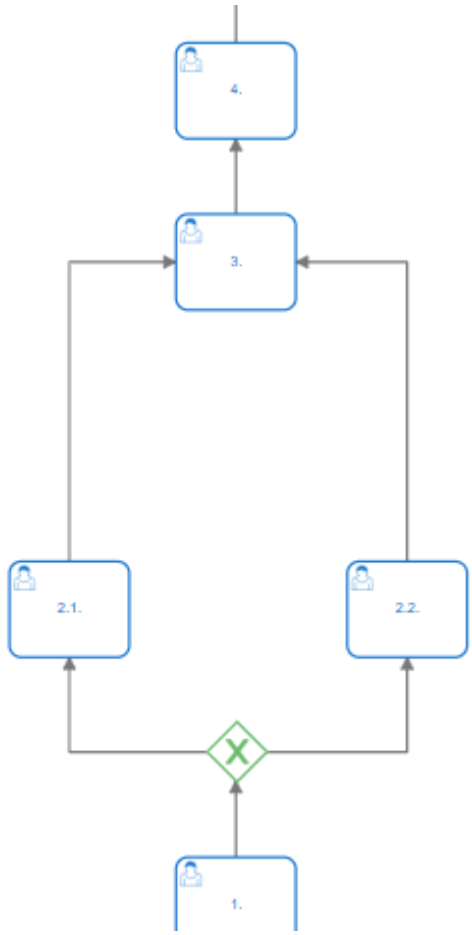
lets say there are instances (called FLOW-1, FLOW-2) where created with this flow ie. as the live / deployed version (and lying in tasks 1. or 2.)
then the flow is updated to remove the inclusive gateway (so that task 3.1 moves directly to 4. and 3.2 no longer exists)

when instances FLOW-1 and FLOW-2 are completed in 2. they move to END OF PROCESS instead of continuing to 3.1. and then 4.

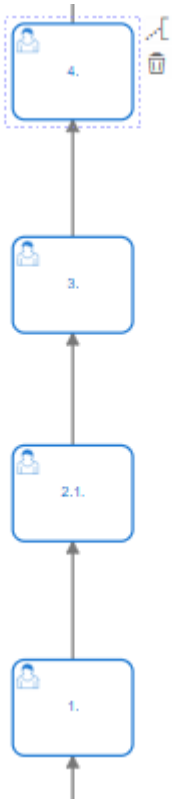
WHY THIS IS STRANGE:
making changes like this works fine (even with tasks still in 2.2. when it is deleted)
but as soon as a “end-of-process” is involved the updates to flow seem to be irrelevant



or



to



warning of field visibility at step that prevents scripts from executing

as default make all fields "involved" in script Editable at given task and use script to set as hidden or read-only

reason being

1. script can retrieve value form read-only field (`.getFieldValue`) but is unable to update (`.setFieldValue`) read-only
2. script cannot retrieve nor update hidden fields

warning on field visibility at step that prevents user interaction

field hidden at task and required by script

at given task

field set to hidden
expect script to make required
but fails

```
function showField(){
    var show = Flokzu.getFieldValue([[Show Other]]);
    if (show === true){Flokzu.setRequired([[Other]])}
    // else {Flokzu.setHidden([[Other]])}
    }
 Flokzu.onInit(showField);
 Flokzu.onChange([[Show Other]], showField);
```

NOTICE: does not work with or without "else" condition

after deployment

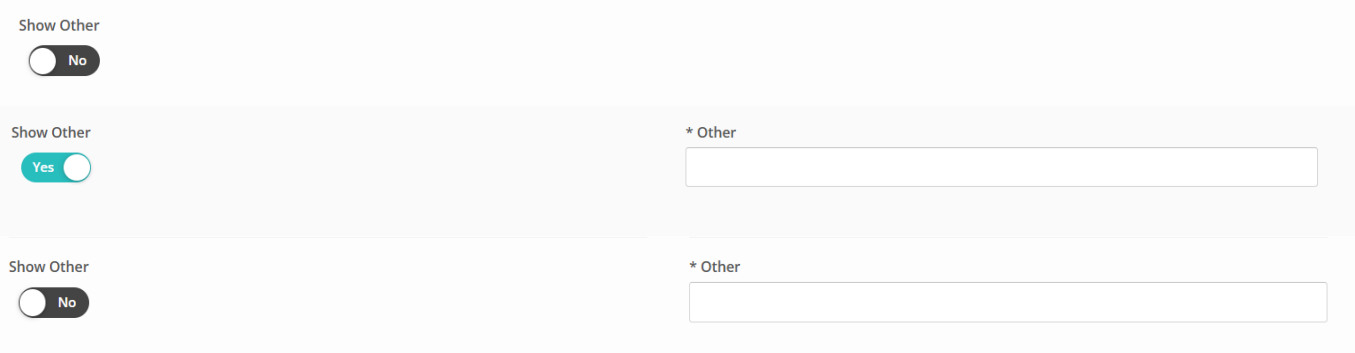
field read-only at task and required with script

at given task

field set to read-only
expect script to make required
semi-succeeds

```
function showField(){
    var show = Flokzu.getFieldValue([[Show Other]]);
    if (show === true){Flokzu.setRequired([[Other]])}}
 Flokzu.onInit(showField);
 Flokzu.onChange([[Show Other]], showField);
```

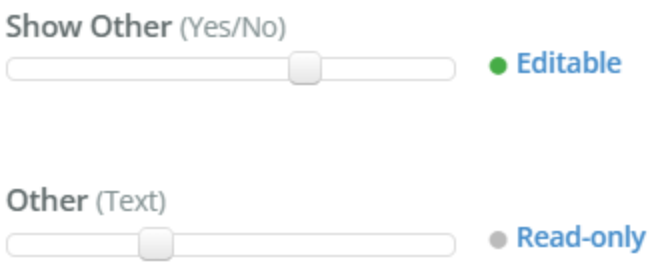
after deployment



could be result of no else condition to reset / change field visibility
demonstrated below

field read-only at script and hidden or required with script

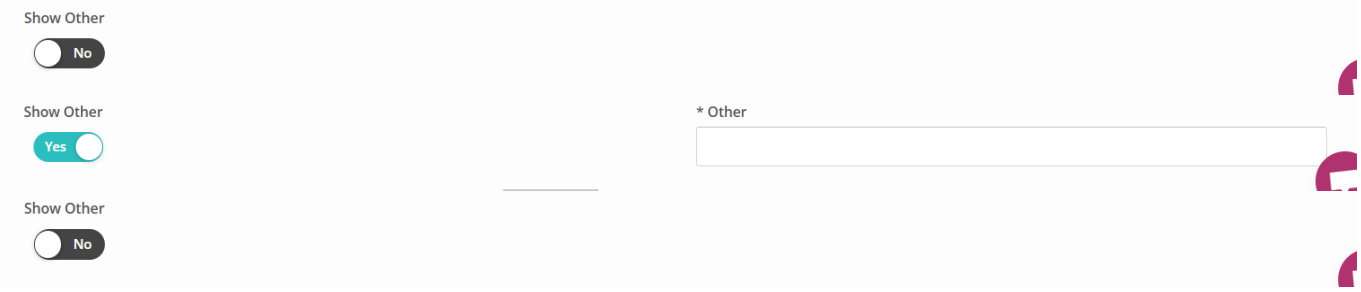
at given task



field set to read-only
expect script to make required OR hide
succeeds

```
function showField(){
    var show = Flokzu.getFieldValue([[Show Other]]);
    if (show === true){Flokzu.setRequired([[Other]])}else {
Flokzu.setHidden([[Other]]) }
    }
 Flokzu.onInit(showField);
 Flokzu.onChange([[Show Other]], showField);
```

after deployment



and continues to work with every change
therefore require else condition

NOTICE
same behavior when using **editable**

RULE OF THUMB
have all fields "used" in script set to editable and let script dictate whether hidden, read-only or required

ALTHOUGH
able to use *read-only fields* if **only requirement** is to "fetch field value" `Flokzu.getFieldValue`

THEREFORE
if you were planning to set the fields to read-only in script

save the extra effort and make read-only in tasks
if however will be hidden could just as well leave editable and hide with script

how to set up External Forms from other processes to interact with current one

external form can be specified in this way s/t they auto-populate when said external form is initialized

in main form

1. set all fields involved to editable
   NOTE: for those fields where plan to set hidden and ONLY use .getFieldValue instead of duplicating work
   can be hidden at step
2. use script to update field with link to external form

```
function externalForm(){
    var id = $('#fkz_ref').text();
    var number = Flokzu.getFieldValue([[Number]]);
    var name = Flokzu.getFieldValue([[Name]]);
    var surname = Flokzu.getFieldValue([[Surname]]);
    var fullname = name + ' ' + surname;
    var first = 'https://app.flokzu.com/public/----?Number=';
    var second = '&Name=';
    var third = '&ID=';
    var URL = first.concat(number, second, fullname, third, id);
    // ALTERNATIVE FORMAT using text field instead of weblink
    // required for cases where https: contains blank spaces
    // var link = '<a href="' + URL + '">Nanme of Link</a>';
    Flokzu.setFieldValue([[External Link]], URL); //
Flokzu.setFieldValue([[External Link]], link);
    // Flokzu.setHidden([[External Link]]); // POST-TESTING
}

Flokzu.onInit(externalForm);
// Flokzu.onChange([[ ]], externalForm);
```

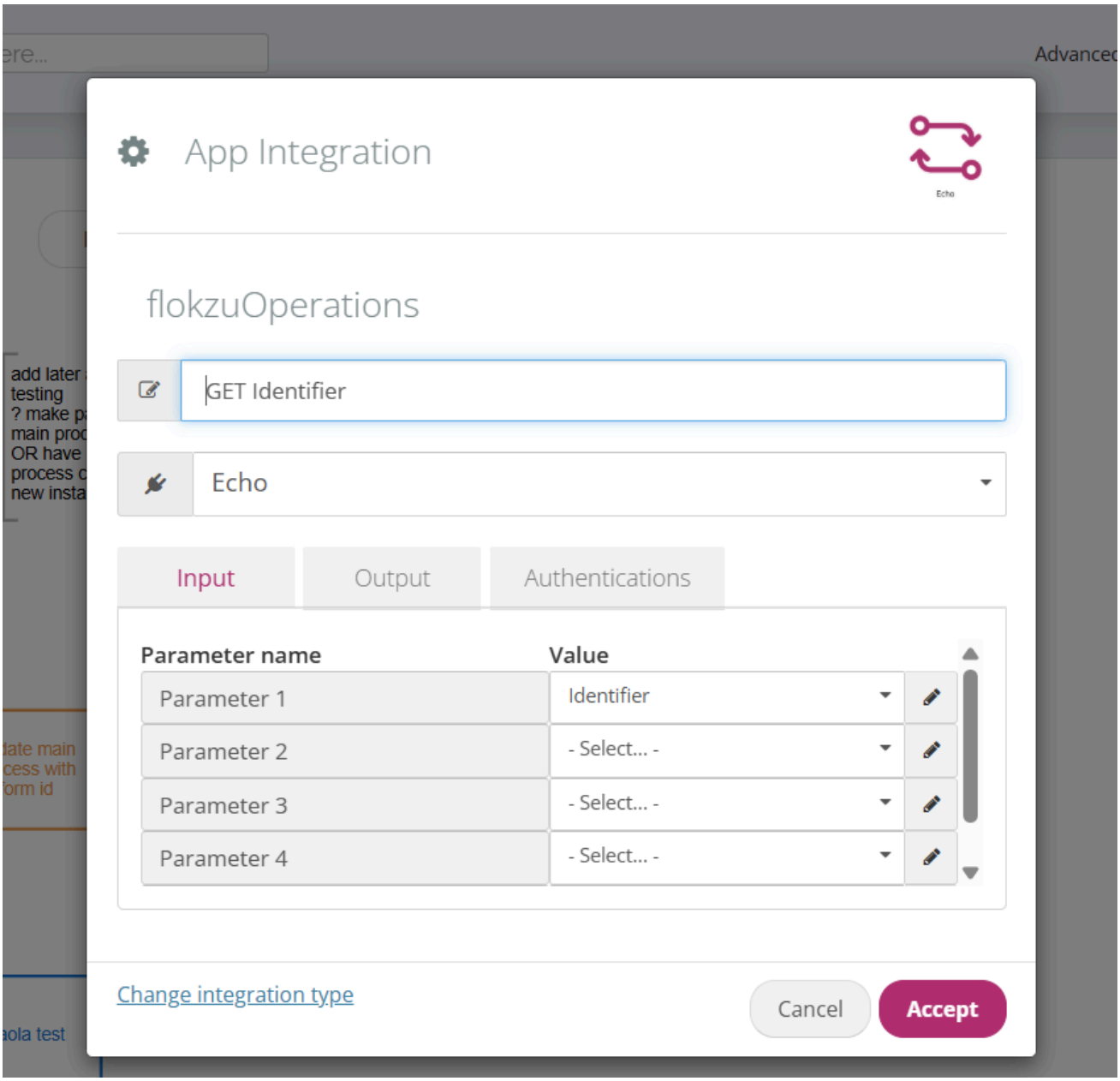
in external form

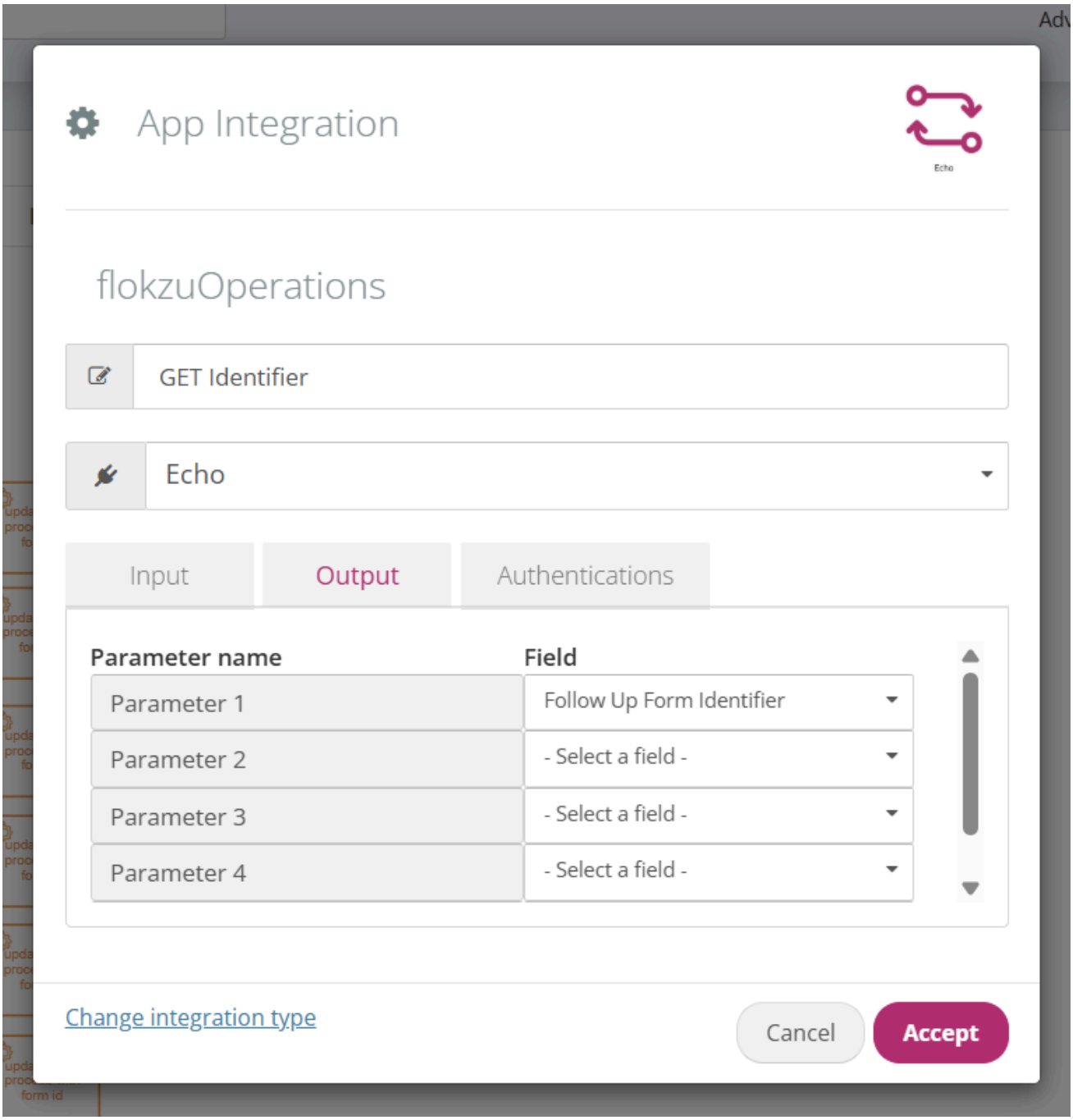
3. create all fields that will be required for url to populate
   editable

4. include field to hold (current) external form process identifier
   integer
   hide
   Flokzu backend able to update hidden fields
5. retrieve form fields from url using script

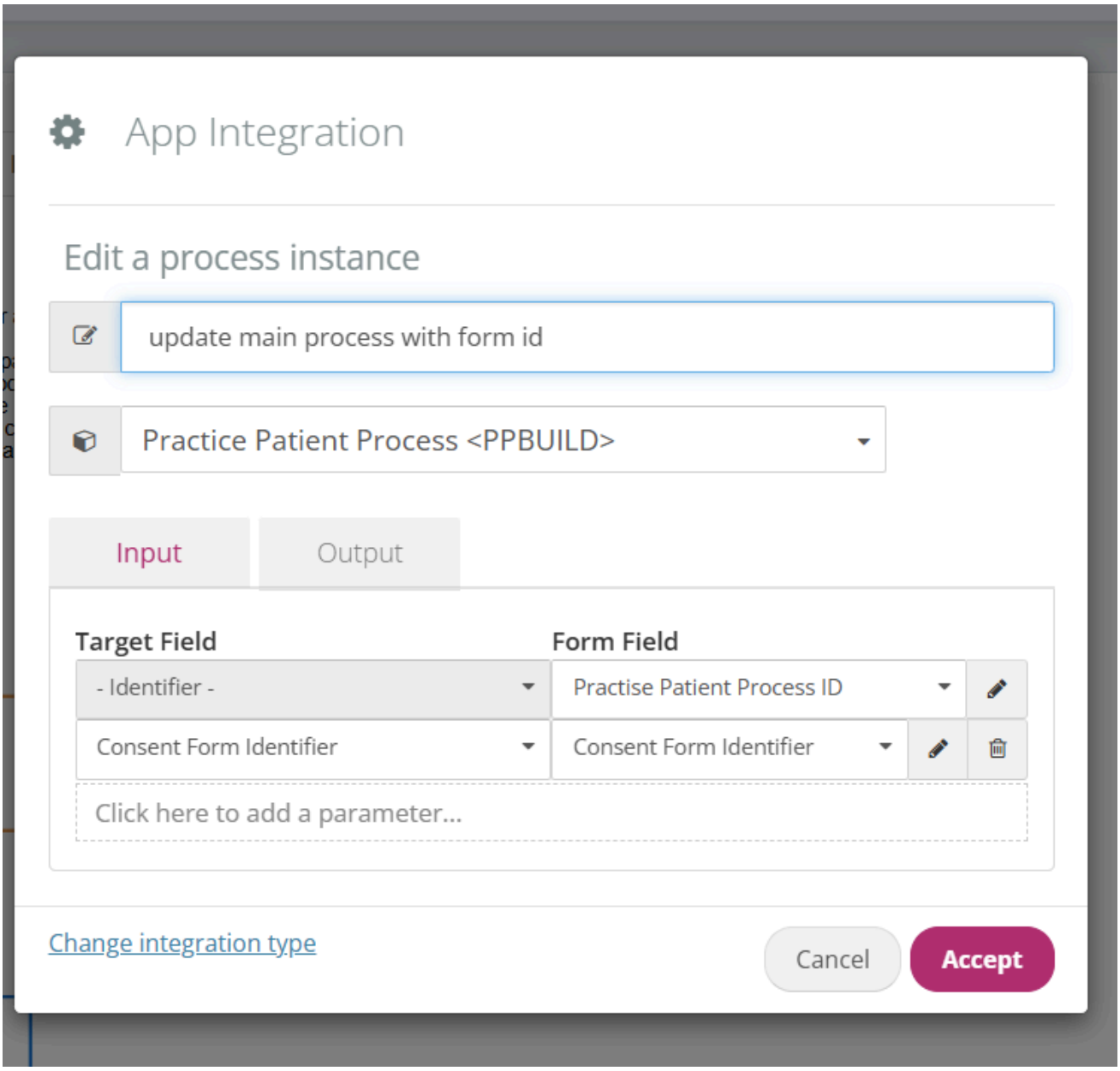
```
// EXAMPLE
// https://app.flokzu.com/public/----?Number=123&Name=Me&ID=3

 $.urlParam = function(name){ // dont touch!!!!
            var results = new RegExp('[\?&]' + name + '=
([^&#]*)').exec(window.location.href);
            console.log(results);
            if (results == null){
               return null;
            }
            else {
               return decodeURI(results[1]) || 0;
 }
        }


function Defaultfields(){

    var number=  $.urlParam('Number');
    var name=  $.urlParam('Name');
    var id = $.urlParam('ID');

console.log(number);

    Flokzu.setFieldValue([[Number]], number);
    // Flokzu.setReadOnly([[Number]]); // POST-TESTING

    Flokzu.setFieldValue([[Name & Surname]], name);
    // Flokzu.setReadOnly([[Name & Surname]]); // POST-TESTING

    Flokzu.setFieldValue([[Main Process ID]], id);
    // Flokzu.setHidden([[Main Process ID]]); // POST-TESTING

}

Flokzu.onInit(Defaultfields);
```

6. get form id


Advanced
App Integration
Echo
flokzuOperations
GET Identifier
Echo
Input
Output
Authentications
Parameter name
Value
Parameter 1
Identifier
Parameter 2
- Select... -
Parameter 3
- Select... -
Parameter 4
- Select... -
Change integration type
Cancel
Accept


App Integration
Echo
flokzuOperations
GET Identifier
Echo
Input
Output
Authentications
Parameter name
Field
Parameter 1
Follow Up Form Identifier
Parameter 2
- Select a field -
Parameter 3
- Select a field -
Parameter 4
- Select a field -
Change integration type
Cancel
Accept

7. then update main process



test

8. create instance and move to task of interest
9. check in main form field with link to external form updated
10. open newly generated link and verify fields that should have auto-populated as expected
11. complete external form
12. refresh main process task and verify external form instance id updated as expected
13. once that is satisfied and to script in main form and in external form]] to set hidden / read only

how field visibility may prevent script form executing / user interaction

fields updated by script

must be editable at that task where script runs

conditionally visible fields

must be editable at that task where script runs

cannot have hidden field be set to required by script (user cannot interact)
therefore for fields that appear based on values of other fields
must be editable and then hidden by script

fields from which values retrieved by script

must be editable or read-only
and then set to hidden when initiated

BUT

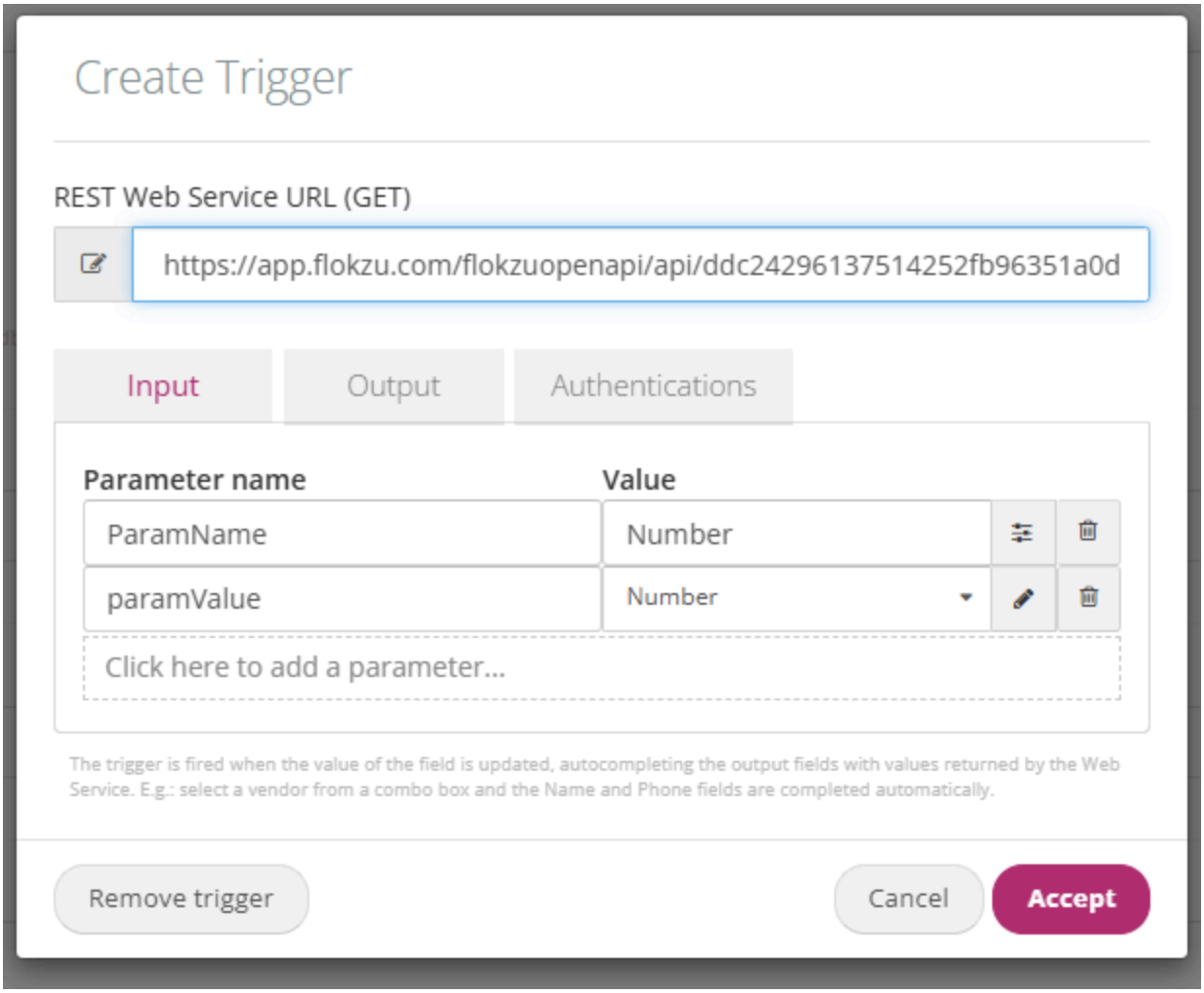
fields updated from database

can be hidden

when setting up db interaction from form field (trigger)

input

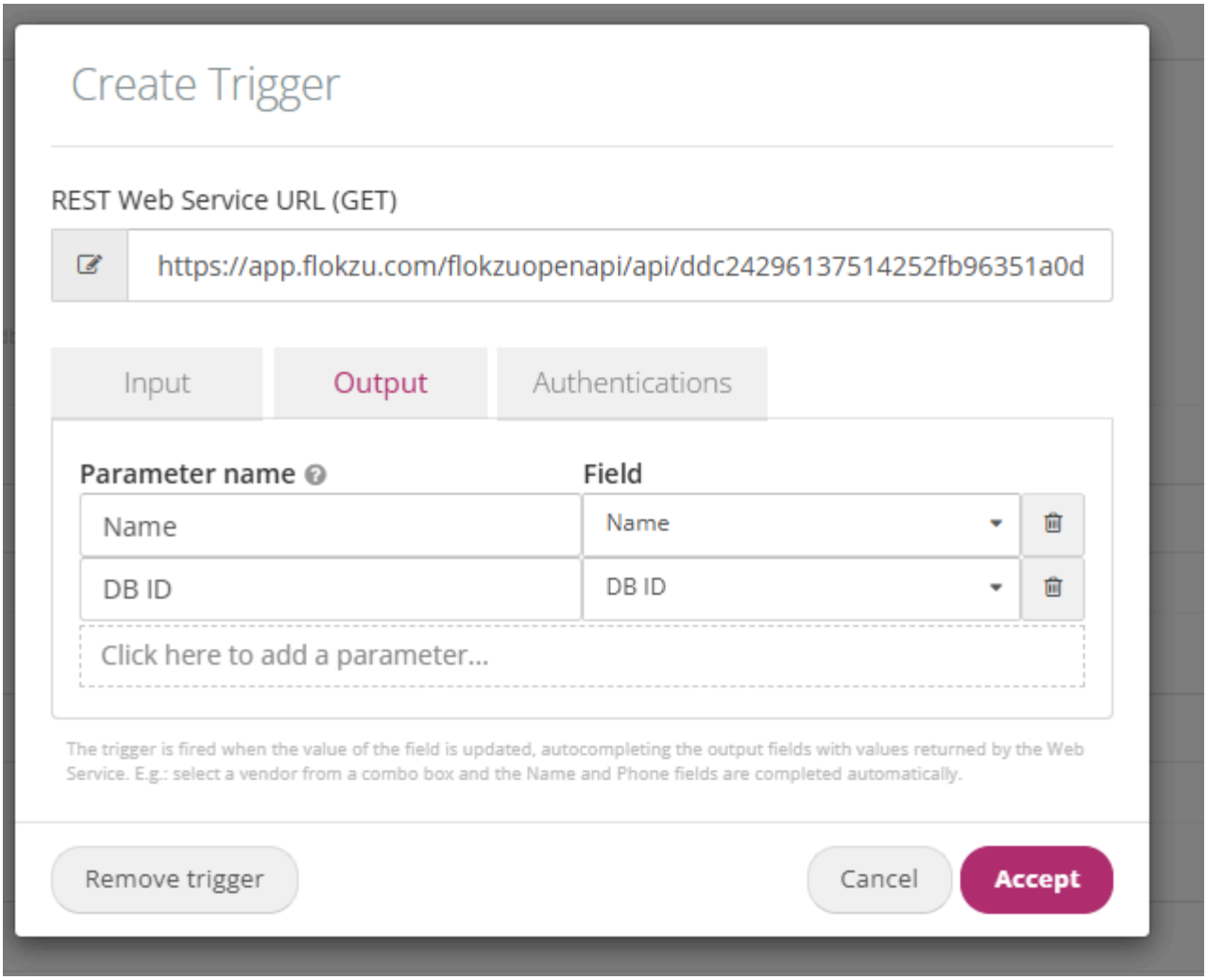
lookup value from db (recommend using unique identifier)



set up in field "Number"

output

how to populate fields



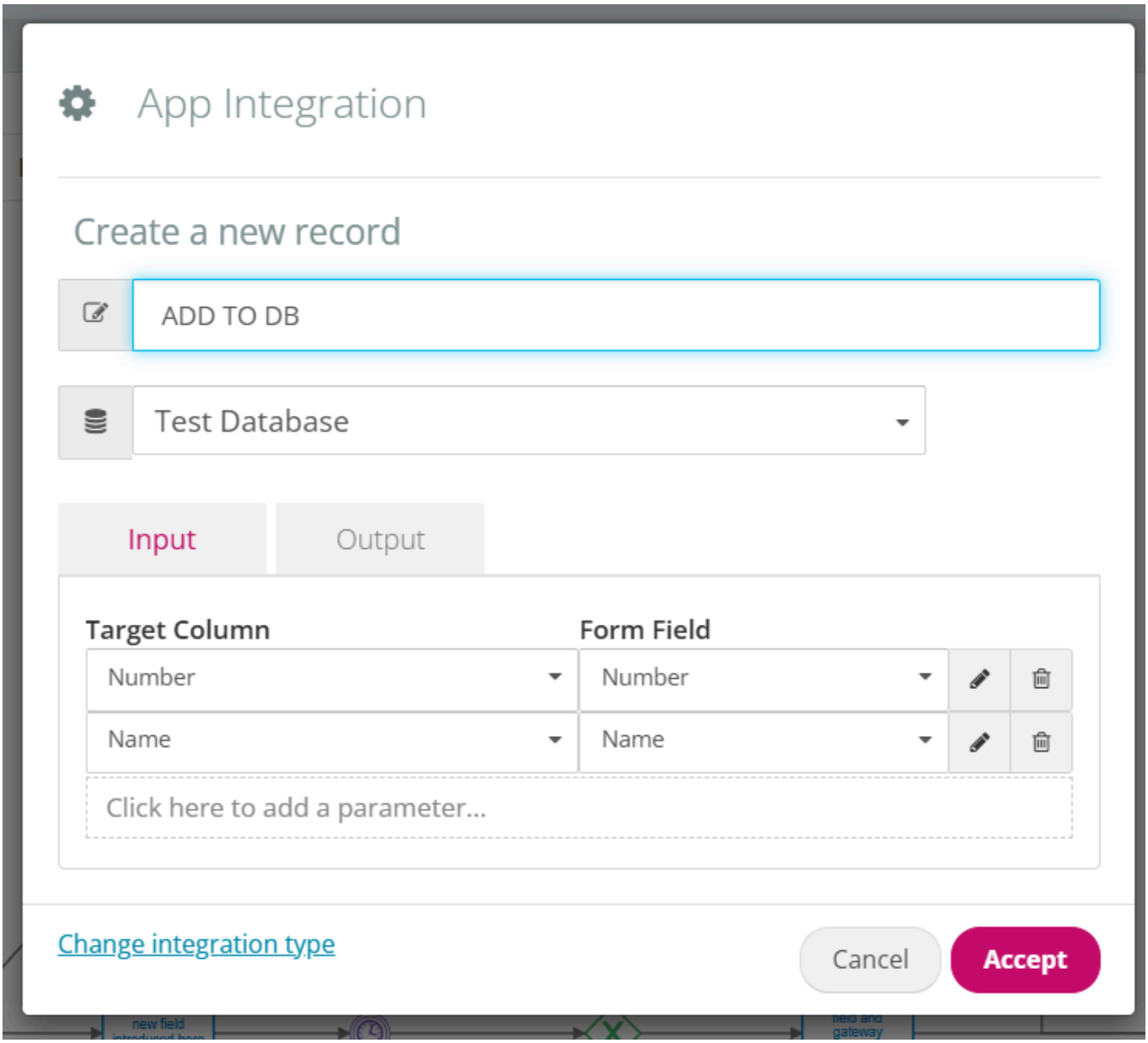
NB **Parameter name** should match db column name **exactly**

WARNING
trying to populate the db ID field this way does **not** work
whereas and return db id to form does work therefore consider flow of add to / edit record in db
only annoynace is that error log will show erros when new db entry is made since unable to find matching record in db
although benift is does work even if no value is given in "unique identifier" field - simply add record to db without any

adding and editing instances in db after task

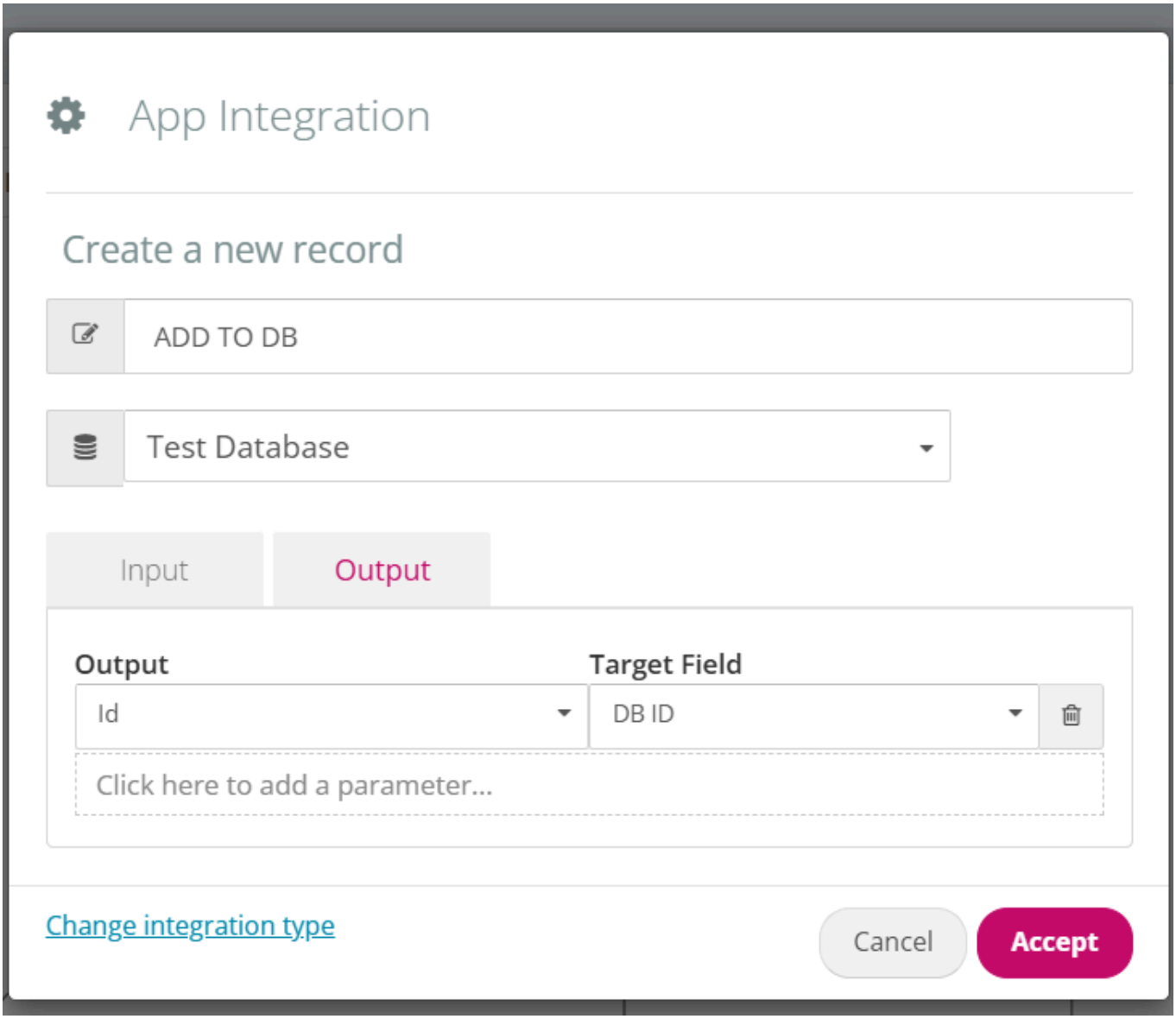
NOTICE
*fields used to store db primary key **must be integer** and **can be** hidden throughout*

add record to db



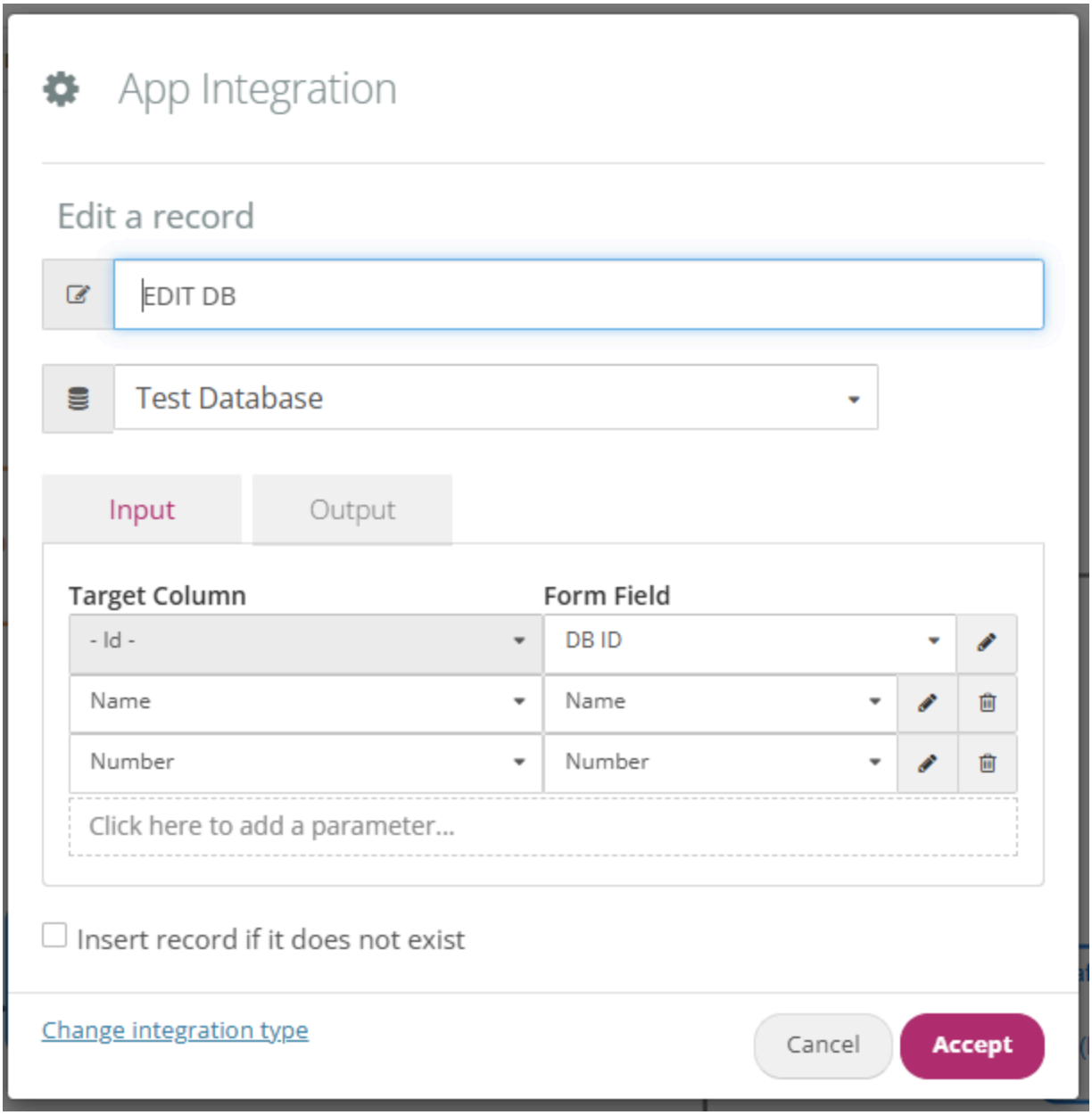
and return db id to form

field can be set hidden



edit record based on db id

based on and return db id to form



add to / edit record in db

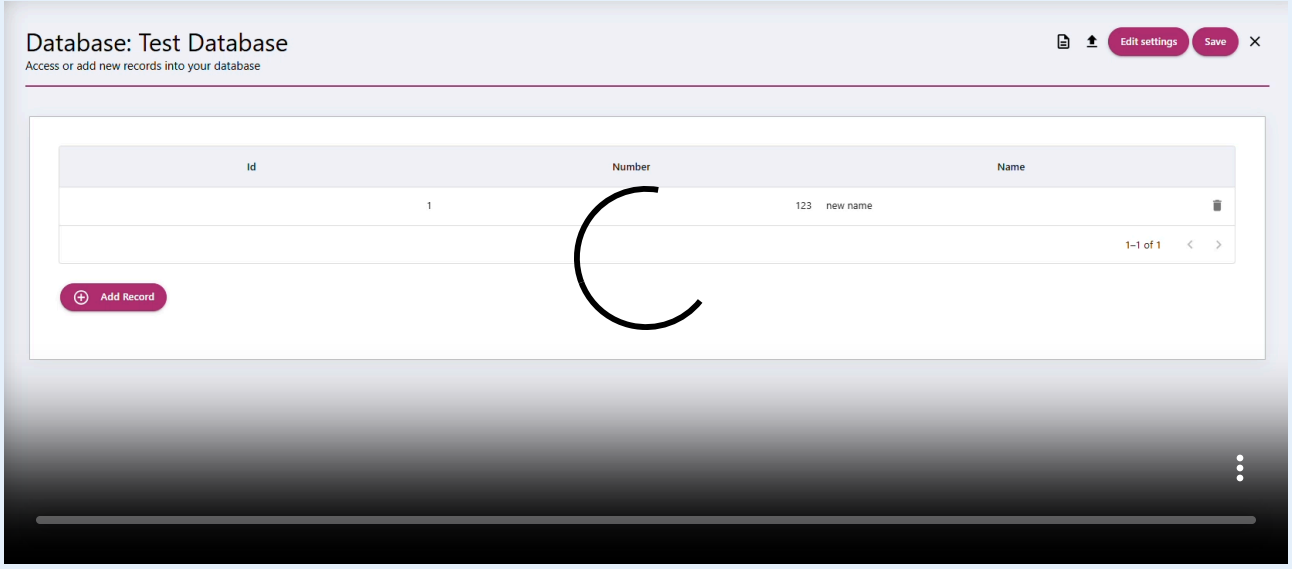
since when setting up db interaction from form field (trigger) > output fails to update ID field (even when et to editable in task) use this flow

HOW THIS WORKS

if new record

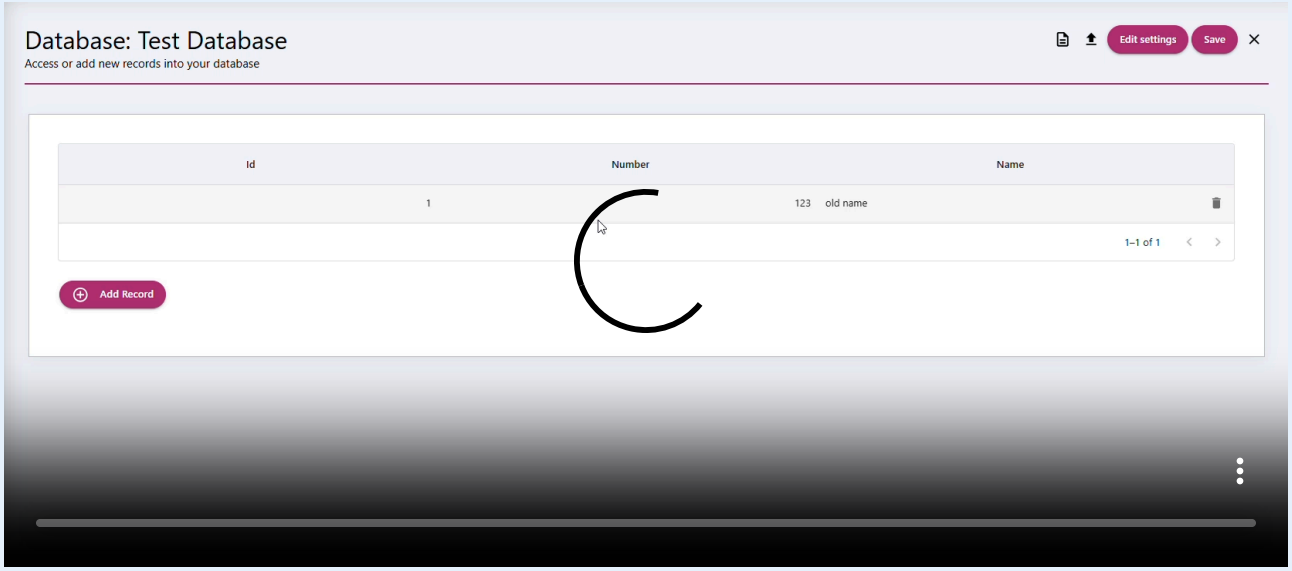
if unique identifier field is not found in db fetch db id (if exists)
then and populate db ID form field does not occur hence blank field
and add / edit record (depending on if db record id exist) adds record to db since no id found

and populate db ID field in form with newly created db id
which can then be used to edit record in edit record based on db id



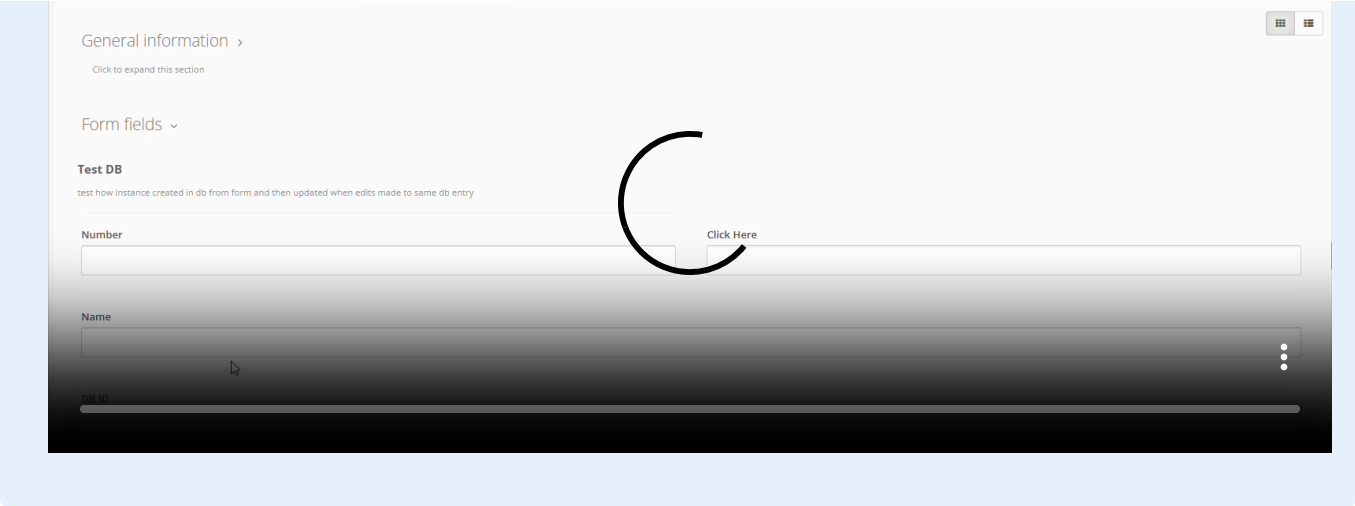
if previous record

unique identifier found in fetch db id (if exists)
therefore db id field populated from and populate db ID form field
which can then edit record in add to / edit record in db instead of adding new (but assuming no fields changed essentially does nothing)
so that changes can be made at same location as with case where no record found in edit record based on db id



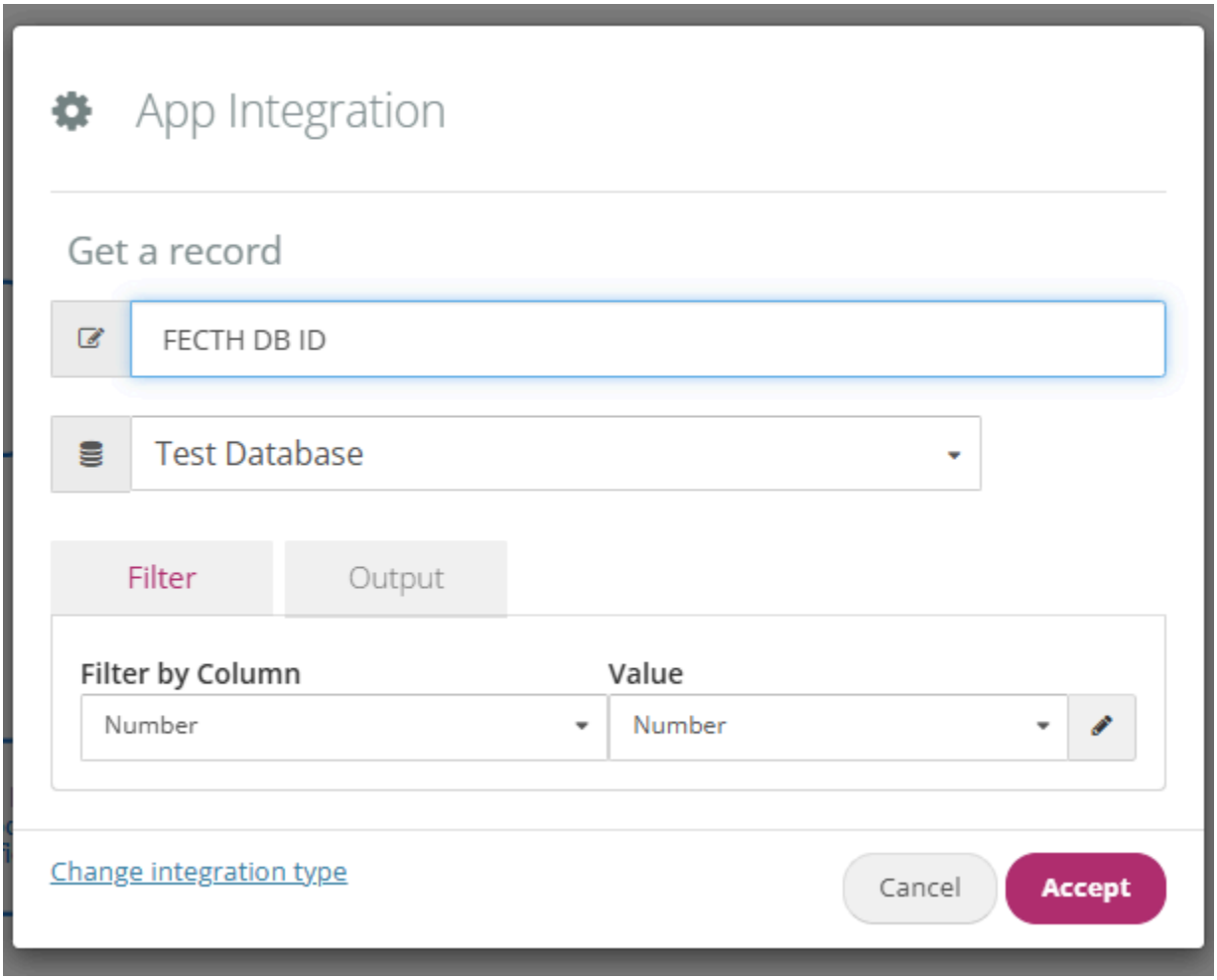
NOTICE

works (updates id ID field) even if field used as unique identifier null

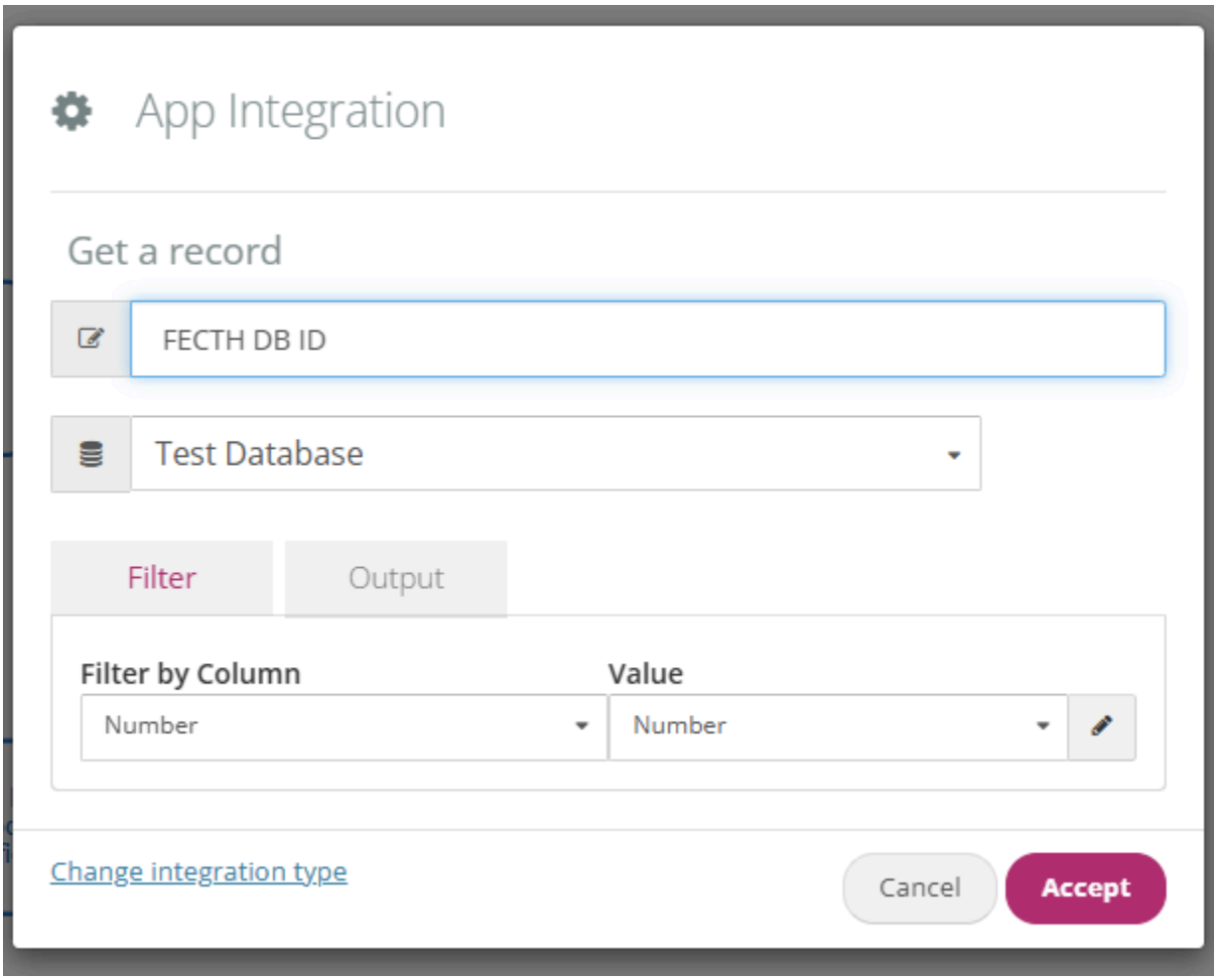


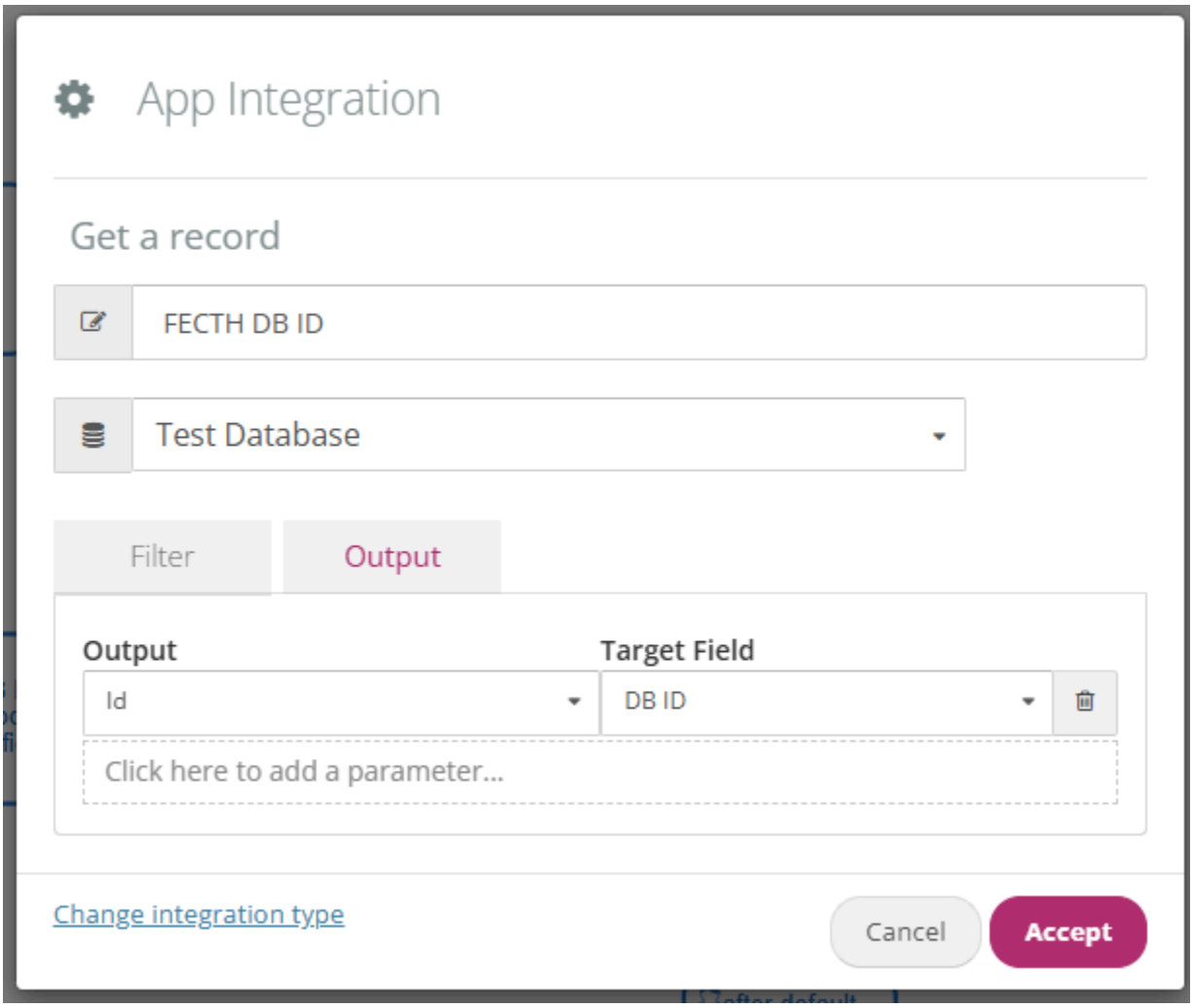
fetch db id (if exists)

NB use same field as "unique identifier" in [when setting up db interaction from form field (trigger) > input]

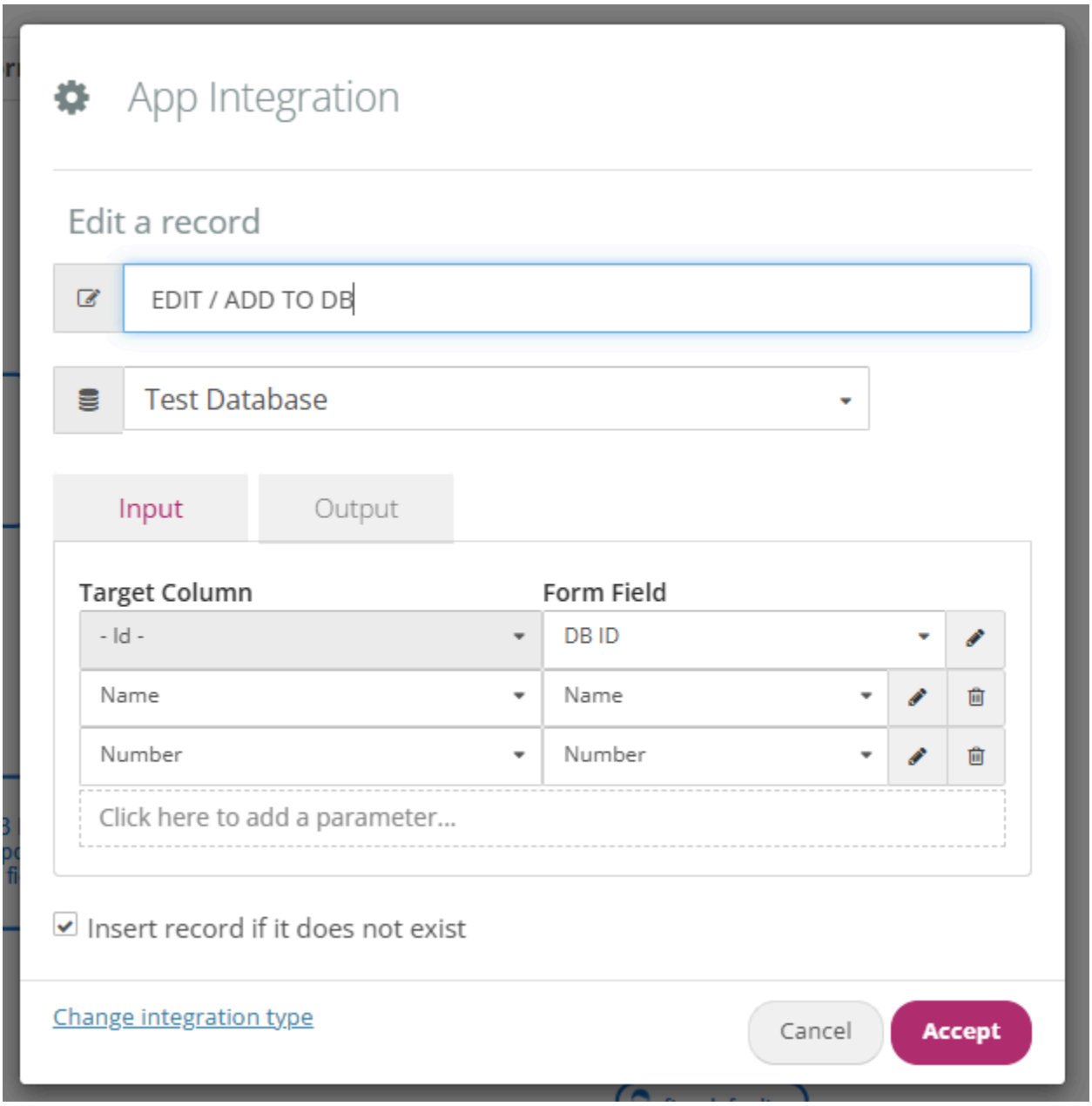


and populate db ID form field

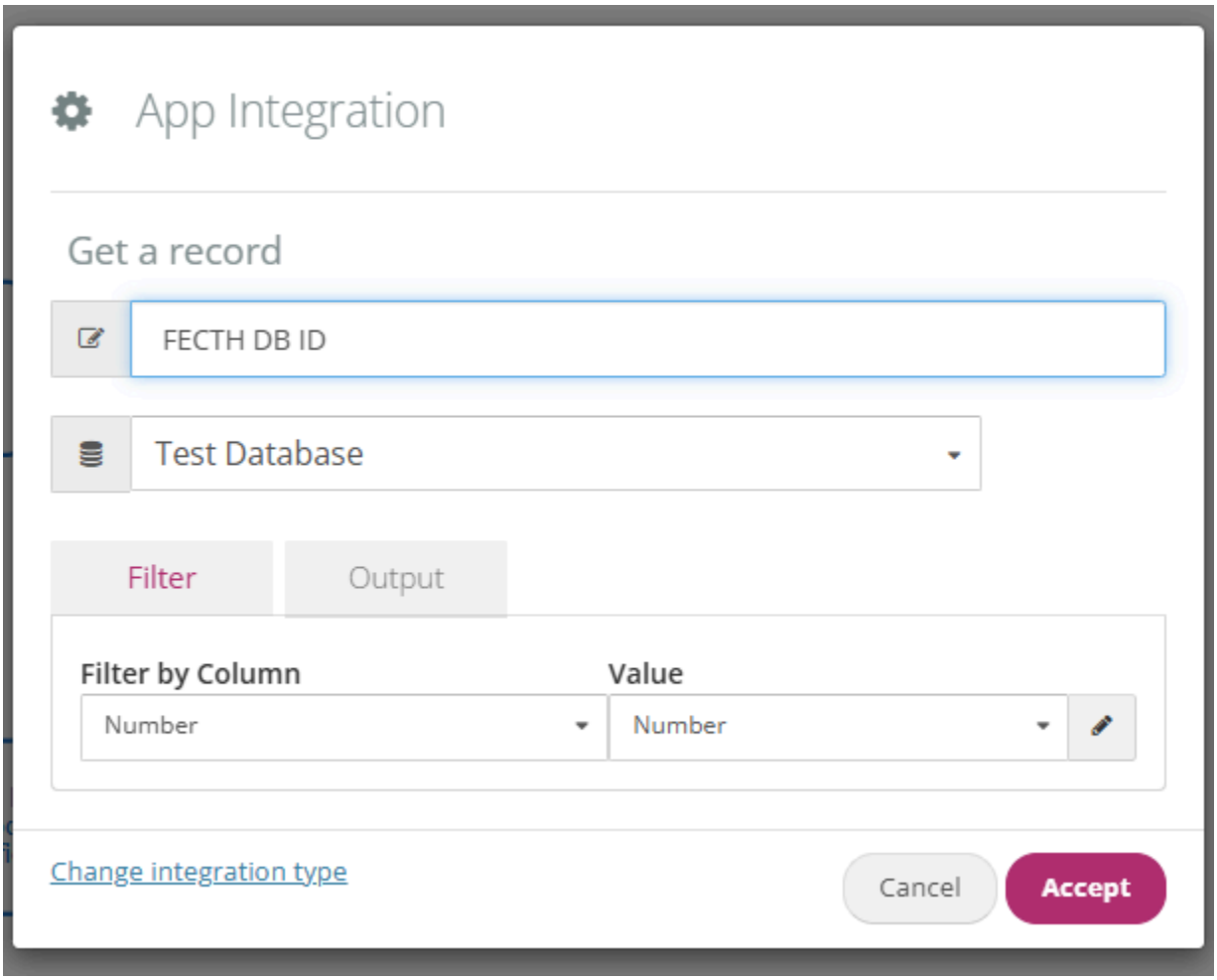

App Integration
Get a record
FECTH DB ID
Test Database
Filter
Output
Filter by Column
Value
Number
Number
Change integration type
Cancel
Accept

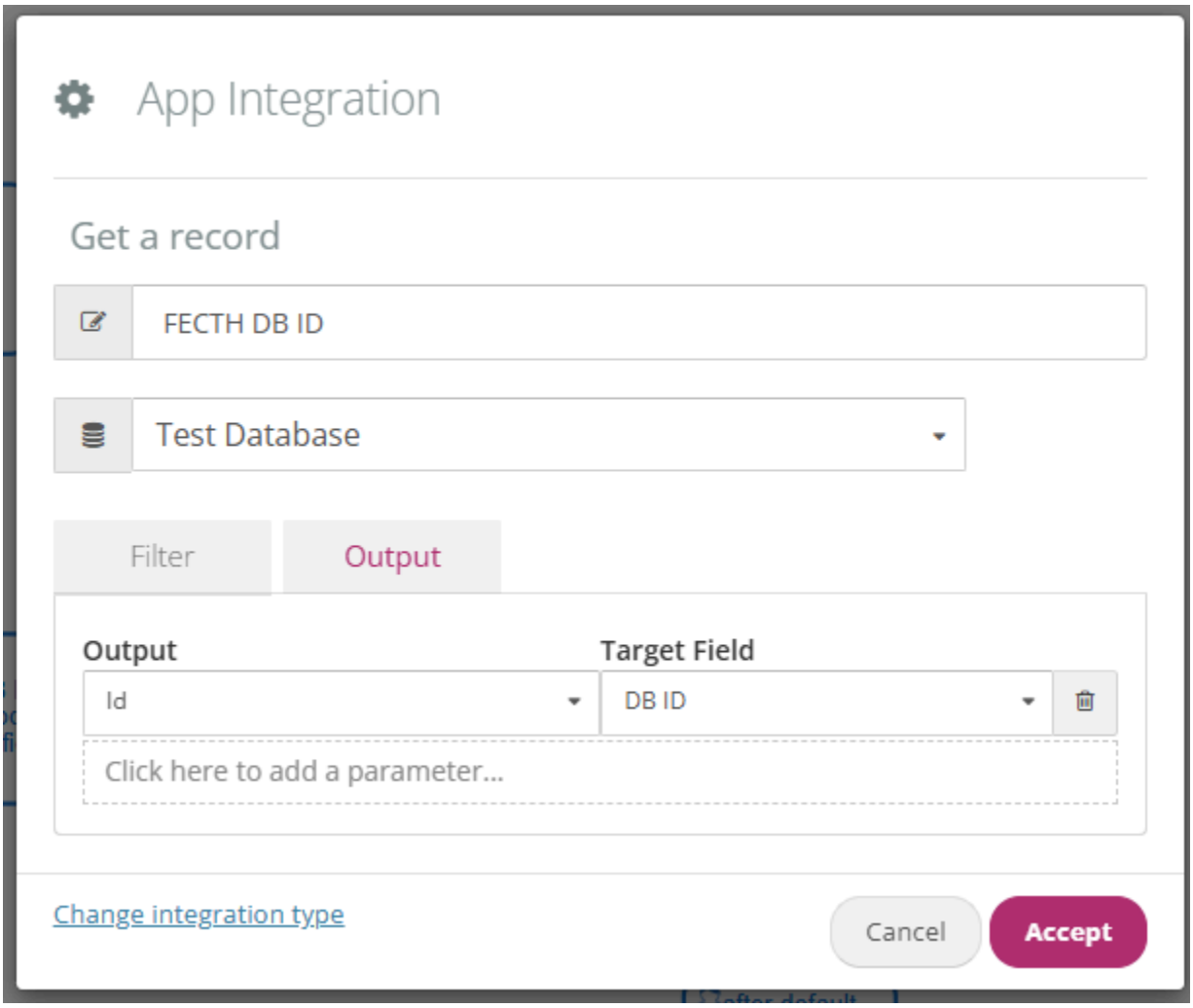


add / edit record (depending on if db record id exist)



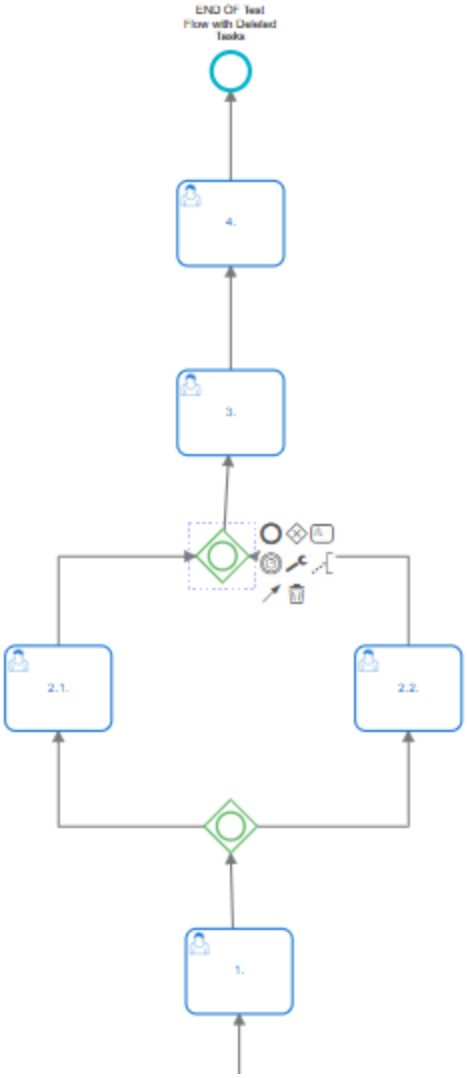
populate db ID field in form


App Integration
Get a record
FECTH DB ID
Test Database
Filter
Output
Filter by Column
Value
Number
Number
Change integration type
Cancel
Accept



info on how system behaves when task in flow deleted

before



after

NOTICE gateways also removed

tasks prior to alteration

tasks in 1. continued with *updated* flow (--> 2.1. --> 3.)
hence

previously created tasks follow newest flow

instances existing in deleted task

tasks in 2.2. (ie. tasks 1. and 2.1. completed) remain in inbox and continue with *previous* flow

since parallel task (2.1.) already completed and waiting at inclusive gateway
therefore

deletion of parallel flow (inclusive gateway) will not cause error in flow
even if existing (in use) tasks removed

instances existing in parallel to deleted task

tasks in 2.1. (ie. tasks 1. and 2.2. completed) remain in inbox and continue *updated* flow
(no loner requiring parallel tasks 2.2. to be completed before progressing to tasks 3.)

info on how system behaves with auto-completed tasks ito gateways

links closely to [info on how system behaves with new form field used in gateway]

when tasks auto-completed values in fields at last update "saved" as if task completed
ie. values set / updated in tasks but not submitted will pull through to rest of process upon auto-completing

WARNING

can be problematic if user alters a field value (eg. drop down) which is later used for gateway condition without updating other fields that may be used in this flow

EXAMPLE:
require user to complete a date field and then update dropdown to "dates specified" so that flow can skip a later task where this to be completed
how this fails:
user updates the dropdown but does not specify a date
the tasks is then not re-initialized later on
and date cannot be set for this instance

SOLUTIONS

NOTICE
should be used exclusively

use script to set fields required based on gateway-used field value

```
function setRequired() {
    var option = Flokzu.getFieldValue([[Dropdown]]);

    if (option) {
        if (option === "Values Updated") {
            Flokzu.setRequired([[Value 1]]);
            Flokzu.setRequired([[Value 2]]);
        }
    }
}

Flokzu.onChange([[Dropdown]], setRequired);
```

TIP
format document so that "Dropdown" is **above** all other fields

so that user updates this field first which then determines field visibility of those below

WARNING
tasks completed on timer can do so even if required fields are empty

use script to revert gateway-used field value as desired if required fields not updated

```
function forceValue() {
    var v1 = Flokzu.getFieldValue([[Value 1]]);
    var v2 = Flokzu.getFieldValue([[Value 2]]);
    var option = Flokzu.getFieldValue([[Dropdown]]);

    // Check if either is empty, null, or undefined
    if (!v1 || !v1) {
        // Only reset if the status is not already "Values Not Updated"
        if (statusField !== "Values Not Updated") {
            Flokzu.setFieldValue([[Dropdown]], "Values Not Updated");
            Flokzu.log("Please add the Value 1 and Value 2 fields before
altering this dropdown");
        }
    }
}

Flokzu.onChange([[Dropdown]], forceValue);
```

TIP
format document so that "Dropdown" is **below** all other fields
so that user attempts to update this field last (and unable to do so if required fields not populated)

NB
explain in field description
eg. "Please update xyz before updating this field value"

WARNING
task with required field autocomplete if still blank

info on how system behaves with new form field used in gateway

{tested for tasks in inbox and at timers}

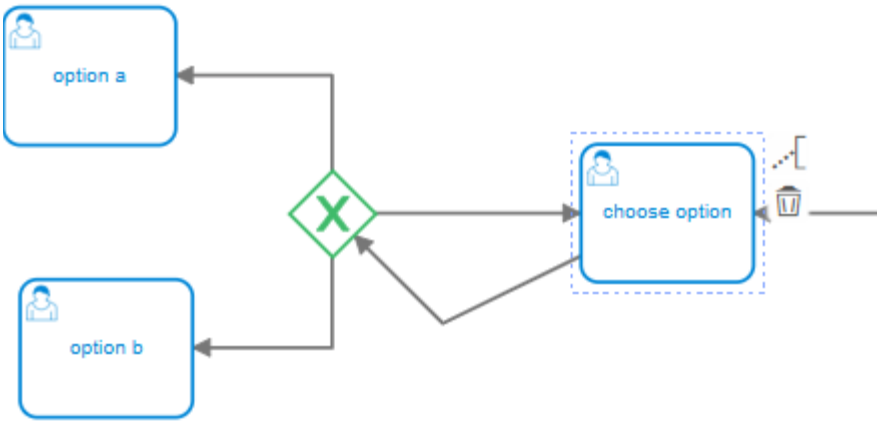
able to update gateway conditions to use new form field for all instances existing before this gateway
but should not delete field from form previously used

as soon as process is updated (to include new field) this field will be present in previously existing instances as well and can therefore be used in gateways
BUT
CRITICAL TO
set default value which is used to direct flow at gateway to flow in which tasks appears where this new field can be updated and "default flow" expected

WARNING
if no condition in one of the exclusive gateway flows then system followed that path even when condition of other flow met

NOTICE
not always the case
able to follow flow where condition met and only if not met follow no-condition flow when exclusive gateway linear

but fails with recursive loops



info on how system handles change in field name

{tested for tasks in inbox and at timers}

WARNING
have to manually update the scripts

NOTE
although field name does not update for instances in timer
as soon as moves out of this field name altered and should be referenced to as new field name in communication eg. `{{Updated Field Name}}`
if old file name used then section of email will be blank