

# Workshop LiveApps

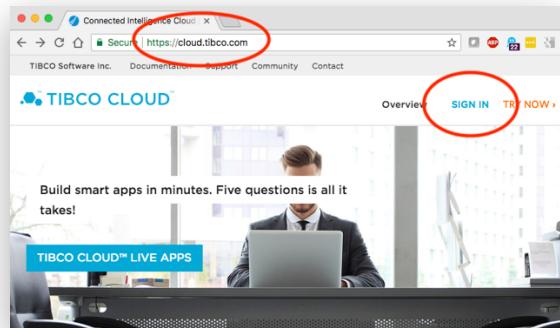


## Table of Contents

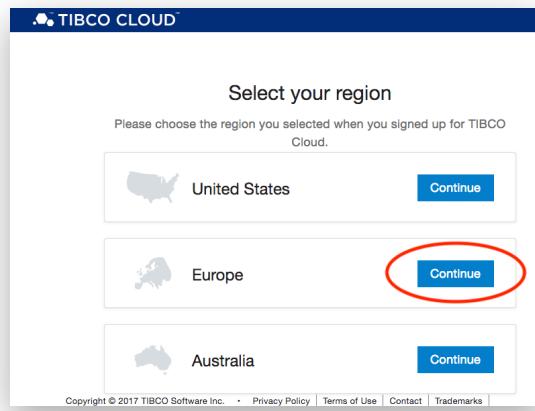
<i>Initial Setup .....</i>	<b>3</b>
<i>Create a new Live App using Wizard .....</i>	<b>4</b>
<i>Fill in some more details.....</i>	<b>9</b>
<i>Test the first version of the App .....</i>	<b>11</b>
<i>Create phase ‘Completed’ .....</i>	<b>14</b>
<i>Publish the App .....</i>	<b>16</b>
<i>Administration Overview .....</i>	<b>18</b>
<i>Improve the App.....</i>	<b>19</b>
Assign tasks to groups .....	19
Improve the data structure.....	22
<i>Service Integration .....</i>	<b>23</b>
Create Flogo Service .....	23
Test the service functionality in Flogo .....	27
Build and run the service executable.....	28
Create an API Specification in TCI .....	29
Create a TCI Wrapper Service using Web Integrator.....	31
Run and test the Web Integrator App .....	34
Embed the service interaction in the Live App.....	36
Run a final test.....	40
<i>Links Used:.....</i>	<b>42</b>
<i>Optional: Error when testing .....</i>	<b>43</b>

## Initial Setup

Start your browser  
Go to <https://cloud.tibco.com/>  
and sign in.

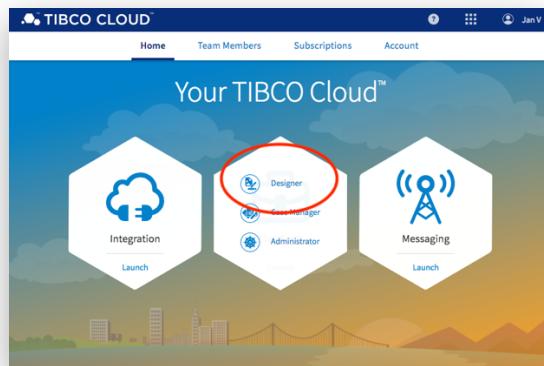


Select your region and continue.

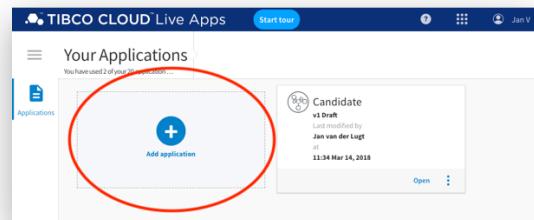


## Create a new Live App using Wizard

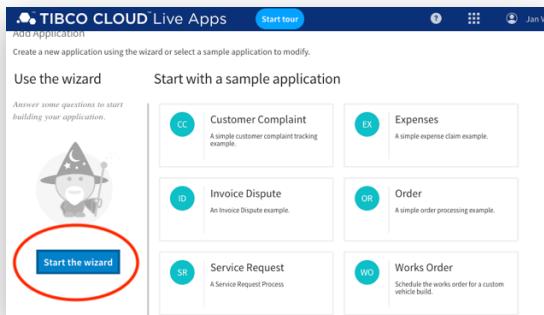
Hover over “Live Apps” then select “Designer” to **start the Live Apps designer**.



From “Your Applications” select “**Add Application**”.



From the “Add Application” screen select “**Start the Wizard**”



In Step 1 fill out the subject of the application as “New Customer”

A screenshot of the "Create an application" wizard, Step 1 of 5. It asks "What is the **subject** of your application?". A red circle highlights the input field where "New Customer" is typed. The "Next >" button is visible at the bottom right.

In Step 2 **enter the states** the new customer can move through. Add the following states (You can just type one name, then press <enter> to add the next etc.):

- Entered
- Welcome mail sent
- Validated
- Accepted
- Rejected

Then **select** state "Entered" to be the **first state** when you start.

Create an application X

### Step 2 of 5

As the 'New Customer' is processed, what **states** might it move through before you are finished working on it?

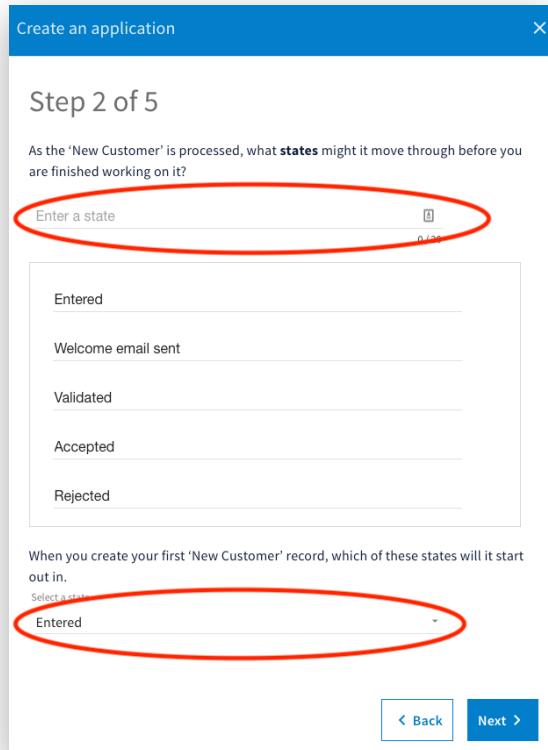
Enter a state [ ] 0 / 20

Entered  
Welcome email sent  
Validated  
Accepted  
Rejected

When you create your first 'New Customer' record, which of these states will it start out in.  
Select a state

Entered [ ]

< Back Next >



In Step 3 **add the data** that is used for a new customer. For now just enter:

- First name
- Last name
- Email address

We will add more data later on.

Press Next >

Create an application X

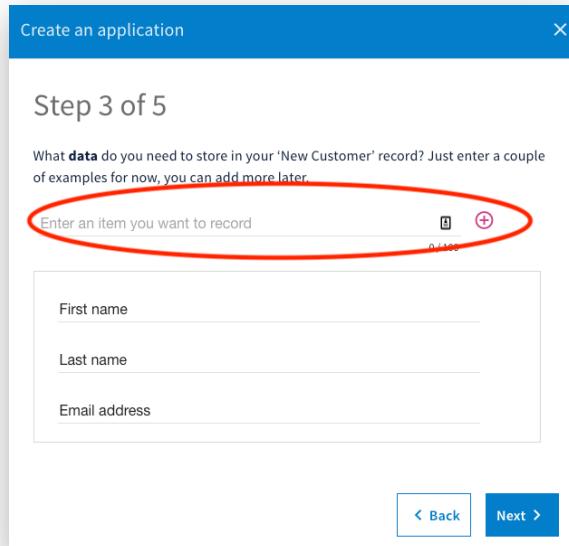
### Step 3 of 5

What **data** do you need to store in your 'New Customer' record? Just enter a couple of examples for now, you can add more later.

Enter an item you want to record [ ] 0 / 100

First name  
Last name  
Email address

< Back Next >



In Step 4 **name a start task** for that happens for each new customer.  
 Fill out “Send welcome mail”.  
 Then select “Welcome mail sent” to be the state the new customer is in after the start task is performed.

Press [Next >](#)

Create an application X

### Step 4 of 5

Is there a follow up task that **always** happens after the ‘New Customer’ is created? If so, please name the task below

18 / 100

What **state** will the ‘New Customer’ be in when this task is complete?  
[Select a state](#)

▼

[< Back](#) [Next >](#)

In Step 5 **add other actions** than can happen as part of processing a new customer.  
 Add the following actions:

- Validate
- Accept
- Reject

Note that for each action another step in the wizard is added where we can choose when these actions are available.

Press [Next >](#)

Create an application X

### Step 5 of 8

What actions **might** happen to the ‘New Customer’ as a normal part of processing?  
 Each new action you add will need to be configured in an additional step

0 / 100

Validate 8 / 100

Accept 6 / 100

Reject 6 / 100

[< Back](#) [Next >](#)

In Step 6 we will select **when the “Validate” action can be performed.**

Check “Welcome email sent”

Then we will **select the next state** after validation.

Select “Validated”.

Press

**Next >**

Create an application X

### Step 6 of 8

In what states will your users be able to perform a 'Validate' action?

Entered  
 Welcome email sent  
 Validated  
 Accepted  
 Rejected

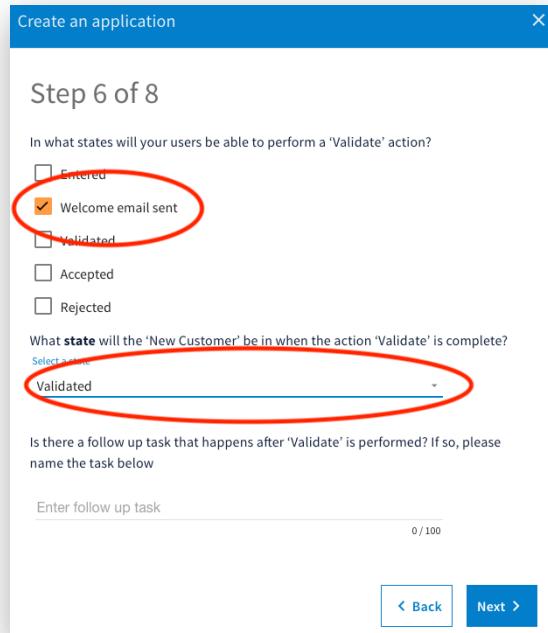
What **state** will the 'New Customer' be in when the action 'Validate' is complete?  
Select a state  
Validated

Is there a follow up task that happens after 'Validate' is performed? If so, please name the task below

Enter follow up task

0 / 100

[< Back](#) [Next >](#)



In Step 7 we will select **when the “Accept” action can be performed.**

Check “Validated”

Then we **select the next state** to be “Accepted”

Then we **name a follow-up task** as “Send acceptance email”

The next state will not change so we select “Do not change the state”.

Press

**Next >**

Create an application X

### Step 7 of 8

In what states will your users be able to perform a 'Accept' action?

Entered  
 Welcome email sent  
 Validated  
 Accepted  
 Rejected

What **state** will the 'New Customer' be in when the action 'Accept' is complete?  
Select a state  
Accepted

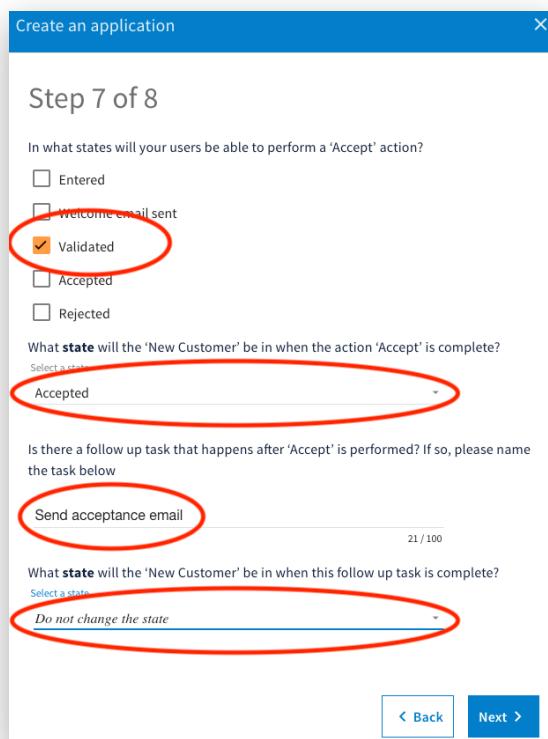
Is there a follow up task that happens after 'Accept' is performed? If so, please name the task below

Send acceptance email

21 / 100

What **state** will the 'New Customer' be in when this follow up task is complete?  
Select a state  
Do not change the state

[< Back](#) [Next >](#)



In Step 8 we will select **when the “Reject” action can be performed.**

Check “Validated”

Then we **select the next state** to be “Rejected”

Then we **name a follow-up task** as “Send rejection email”

The next state will not change so we select “Do not change the state”.

Press **Create**

Create an application X

Step 8 of 8

In what states will your users be able to perform a ‘Reject’ action?

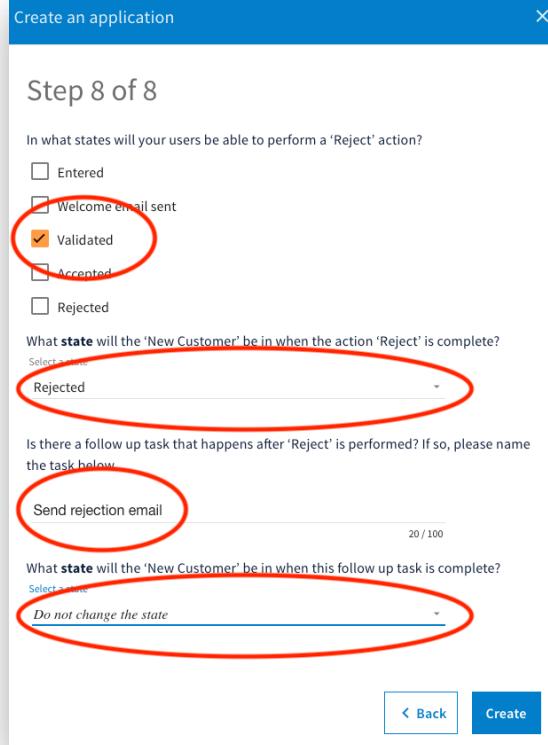
Entered  
 Welcome email sent  
 Validated  
 Accepted  
 Rejected

What **state** will the ‘New Customer’ be in when the action ‘Reject’ is complete?  
Select state  
Rejected

Is there a follow up task that happens after ‘Reject’ is performed? If so, please name the task below.  
Send rejection email 20 / 100

What **state** will the ‘New Customer’ be in when this follow up task is complete?  
Select state  
Do not change the state

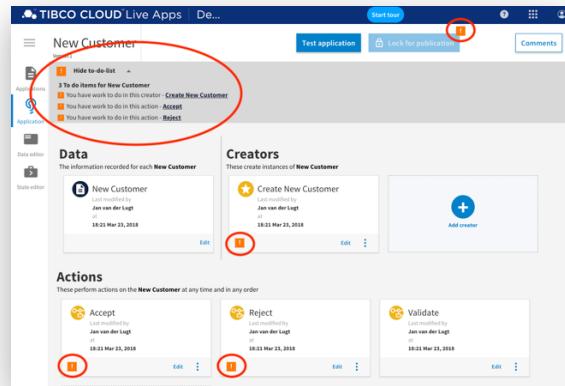
[Back](#) **Create**



## Fill in some more details

Now the app structure has been created we need to **fill in some details** before we can test or publish the app.

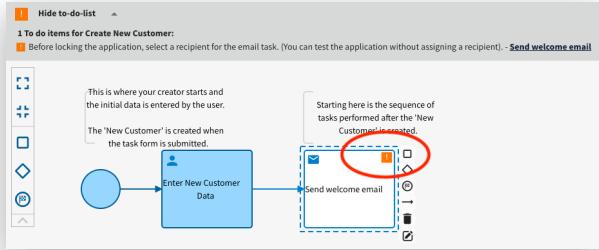
Click the small triangle next to “View to-do list” to get an overview.



Click on the ! symbol at the “Create New Customer” creator

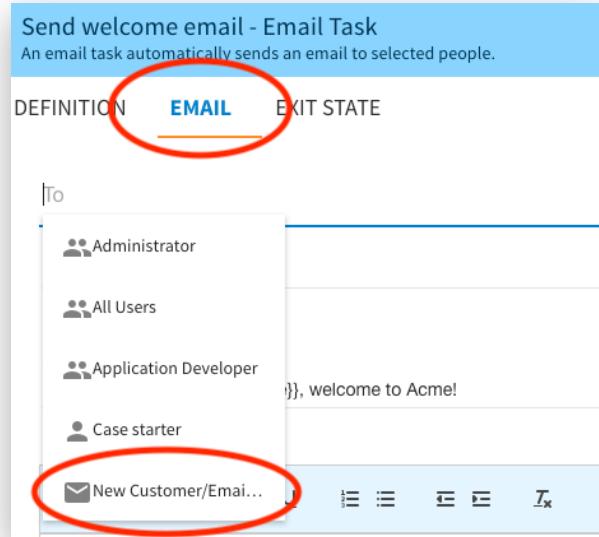
Then on the same symbol at task “Send welcome email”.

There we can fix the missing email recipient.



On this page, the Definition and Exit state should be OK, so select “EMAIL” tab to show the email details and **select an email recipient**.

Click on “To”. Note you get a list of internal email addresses and on the bottom data belonging to New Customer. **Select “New Customer/Email address”** from the list

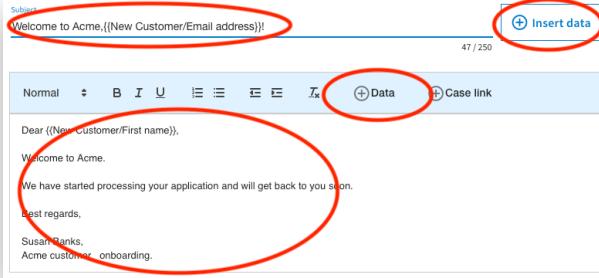


Now fill out the rest of the email details.

Type a **subject**, and make it personal by adding the customer's first name using **Insert data**

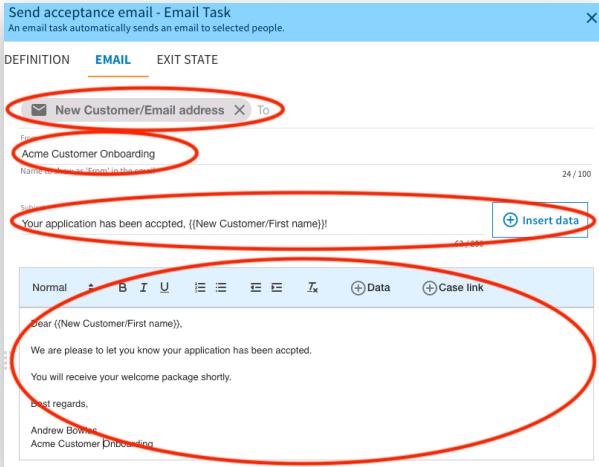
Then **type message body** and use **+Data** from the body title bar

When done, click the large "X" right under your username on the top right of the screen to return to the application.



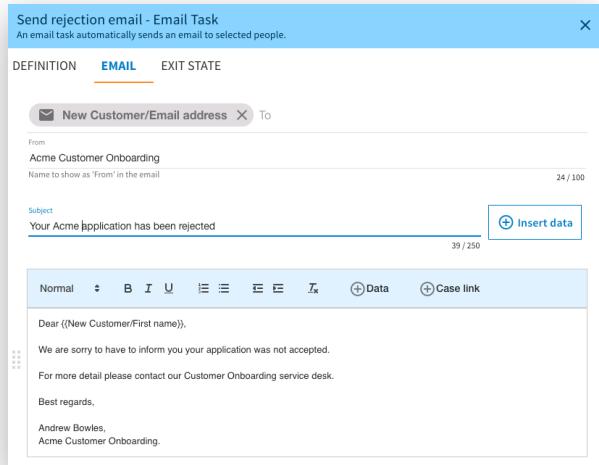
Perform a similar action **to fill in the details for the “Send acceptance email” task.**

When done, click the large "X" right under your username on the top right of the screen to return to the application.

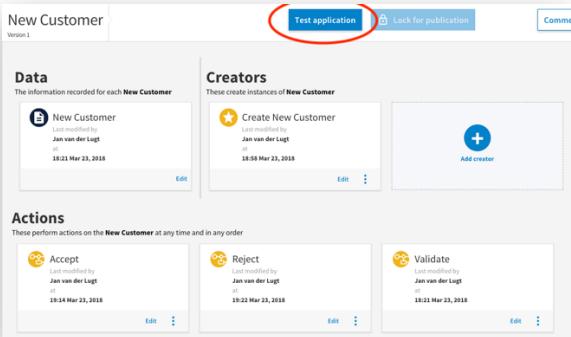
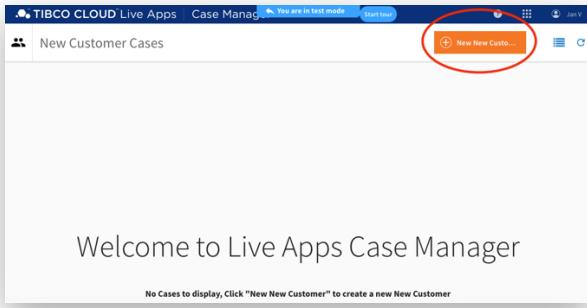
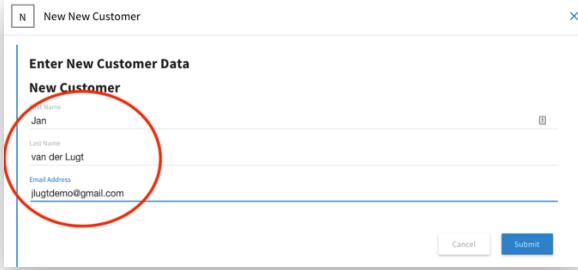
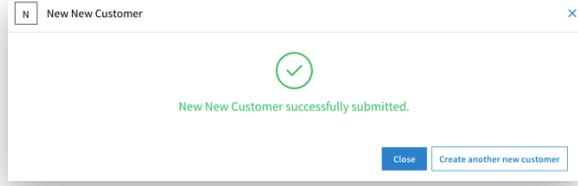


And finally, perform a similar action **to fill in the details for the “Send rejection email” task.**

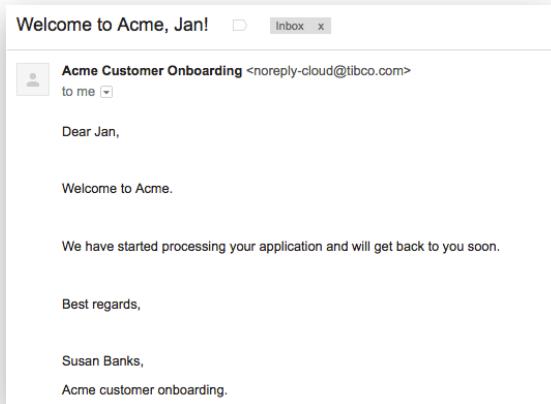
When done, click the large "X" right under your username on the top right of the screen to return to the application.



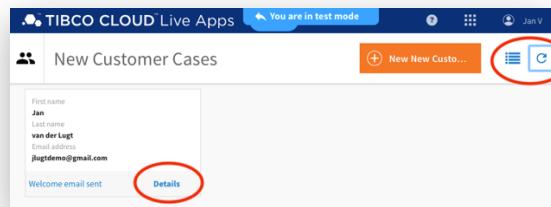
## Test the first version of the App

<p>Now all To-do's should be taken care of so it's time <b>to test the app</b>.</p> <p>Click on "Test Application" on the top of the application title bar.</p> <p>The test engine will perform some final validations.</p>	
<p>Press "New Customer" to <b>create a new "new customer" case</b>.</p> <p>Note you can see you are in test mode at the center top of the screen.</p>	
<p>Here you can see the data fields we've added in the wizard earlier.</p> <p><b>Fill out some data.</b> Make sure you enter a valid email address and press "Submit".</p>	
<p>You will get a <b>confirmation</b> the new customer data was successfully submitted</p>	

You can also **check your mailbox**. Please note that during testing all test messages are sent to the email address associated with your cloud account so it's easier to check all email traffic from your developer account.

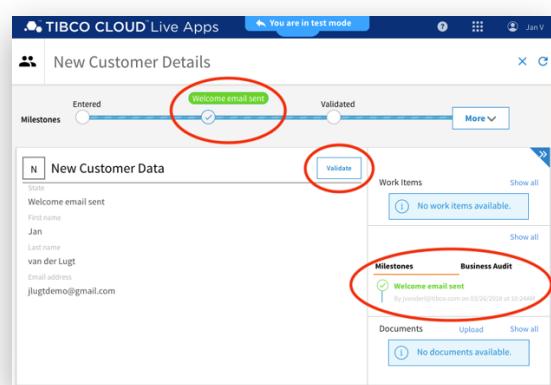


Back on the test screen, if you press the refresh button on the top right your new customer data will appear.  
Press “Details” to **view the customer details**.



Note the **state** now is at “Welcome email sent”, which also is reflected in the “Milestones” section on the right.

The only possible action in this stage is “**Validate**”. Press that button to **move to the next stage**.



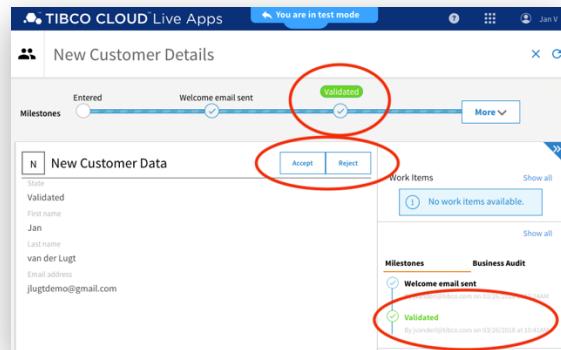
During this stage you can **correct any data** if you wish.

Then press “Submit”.

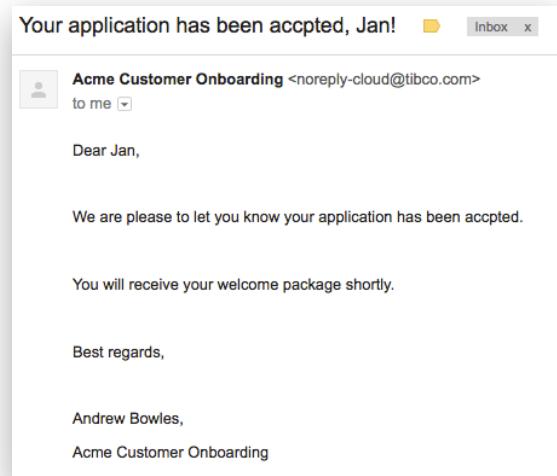
The screenshot shows the 'New Customer' form with fields for First Name (Jan), Last Name (van der Lugt), and Email Address (jlugtdemo@gmail.com). It includes 'Cancel' and 'Submit' buttons at the bottom.

Note that the milestone is now at “Validated” and you have two **new actions** “Accept” and “Reject”.

Go ahead and press “Accept”.



Another **email is sent** to your mailbox.

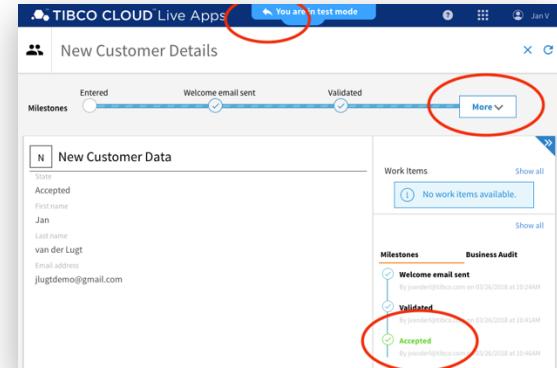


So now the state has changed to “Accepted” and **another milestone is added**.

Note that on the top bar this doesn’t show, but you can press “More” or resize your screen.

We will **group states** “Accepted” and “Rejected” **into a phase** called “Completed” because this makes more sense.

Press the back arrow at the top of the screen to exit test mode.



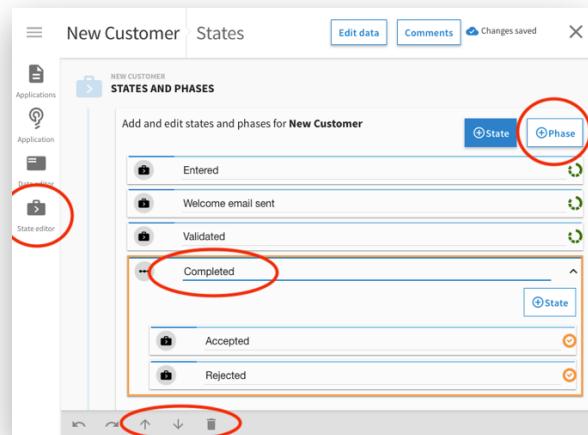
## Create phase ‘Completed’

Back in the designer, select “State Editor” on the left navigation bar.

Press “Phase” to add a new phase and name it “Completed”

Then select state “Accepted” and use the arrow buttons on the bottom of the screen to move the state into the “Completed” phase.

Do the same for “Rejected” and delete the “New state” using the trashcan at the bottom.

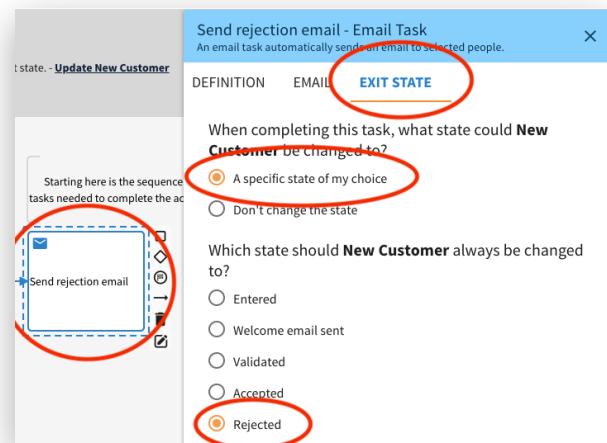
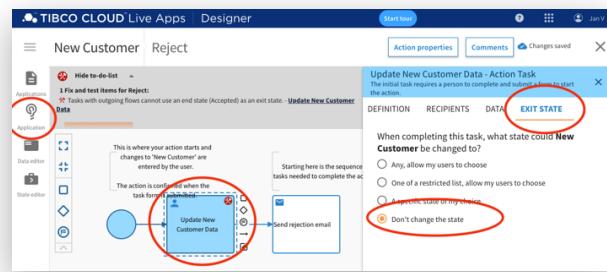


Time to fix some issues after the states have moved.

Select “Application” from the left, and action “Reject”. Then select the “Update new customer data” task, tab “EXIT STATE” and then change the value to “Don’t change the state”. We’ll move that action till after the email is sent.

So now, select the “Send rejection email” task and “EXIT STATE” and change the state to “Rejected”.

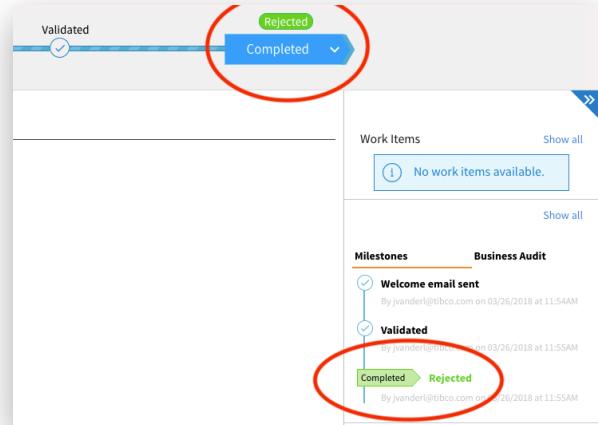
Make the same changes to the “Accept” action.



So now we can **re-run a test**.

This time at the end, choose  
“Reject” as final action.

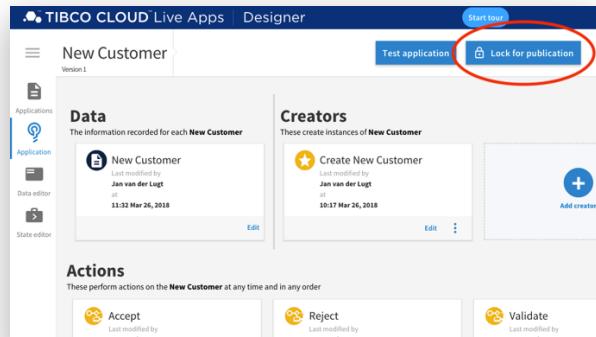
The milestone display should look  
like this.



## Publish the App

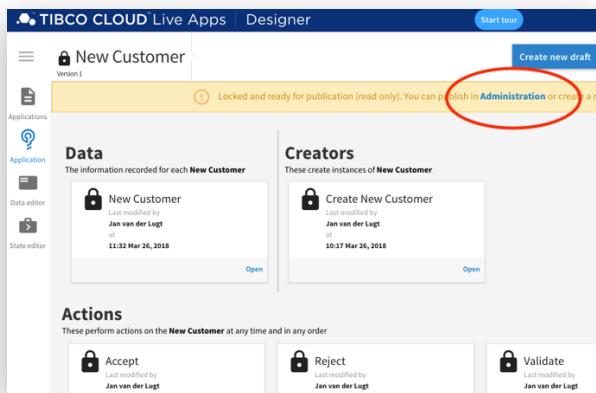
Now it's time to publish the first version of our live App!

Press "Lock for publication" on the top of the screen.

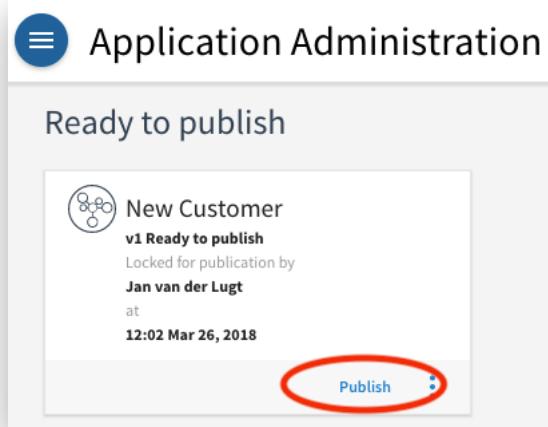


Note now all the app components are locked and you are guided to go to "Administration" to actually perform the publication.

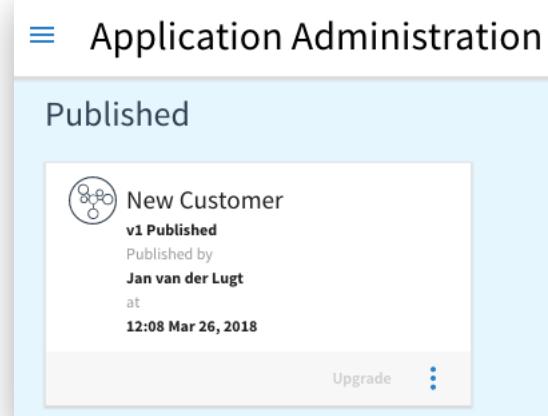
Press "Administration" from here.



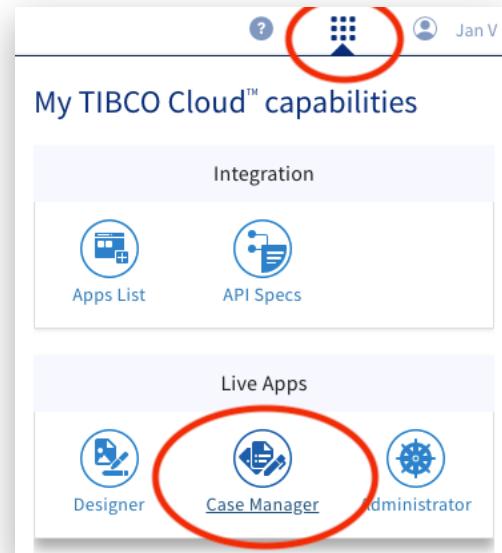
In "Application Administration" find the "New Customer" app under "Ready to publish" and press "Publish".



**“New Customer” app has now moved to “Published” state and is ready to be used.**



To start working with the new app, select “Live Apps” and then “Case Manager”



Select “Create Case” and then “New Customer”.

**Run through onboarding** the customer as you did during testing earlier. Do one that is accepted and another that is rejected.

Create your first case

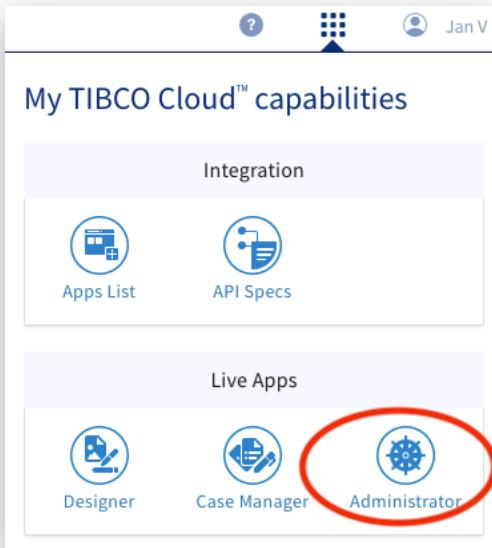
+ Create case

N New Customer

## Administration Overview

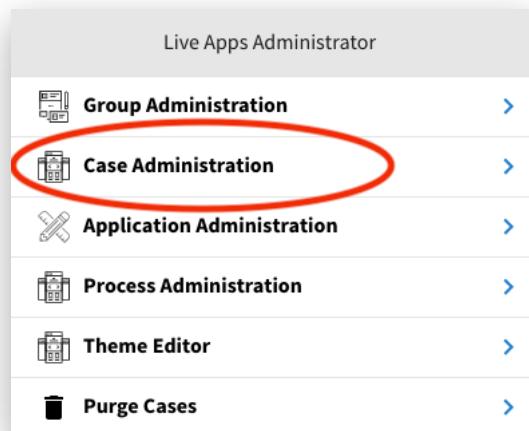
Now it's time to **look at some administrative stuff.**

Select "Live Apps" and then "Administrator" from the main menu.

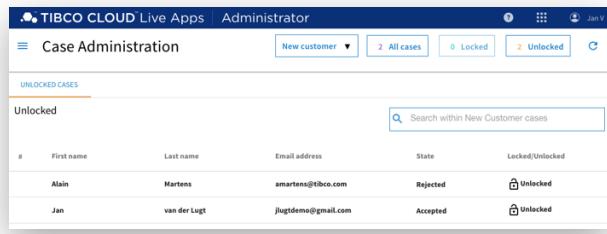


Group we'll get to in a bit.

Select "Case Administration" to **show all cases in all stages.**

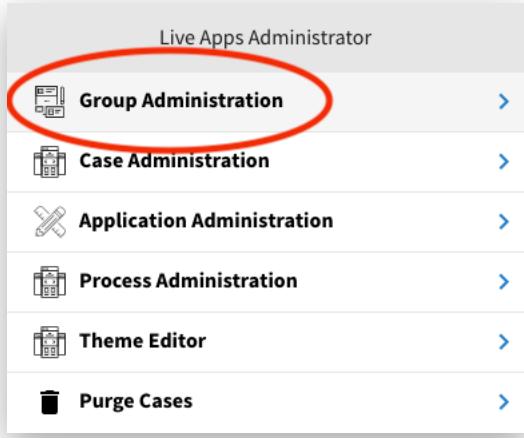
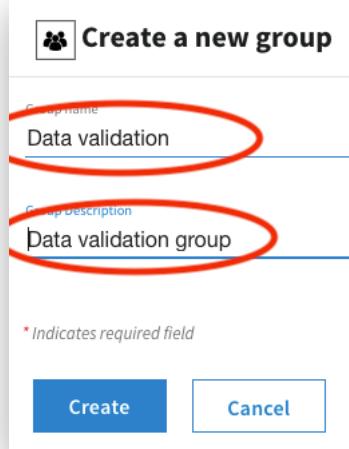
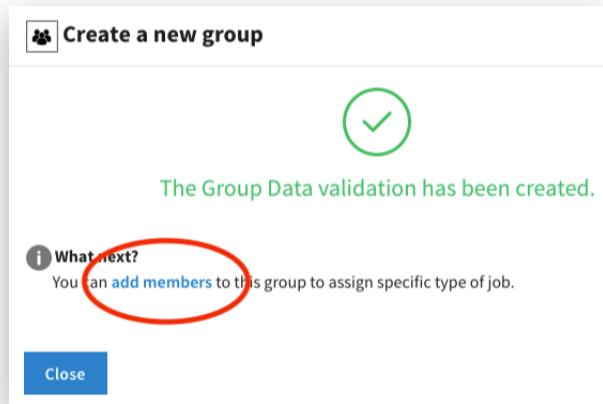


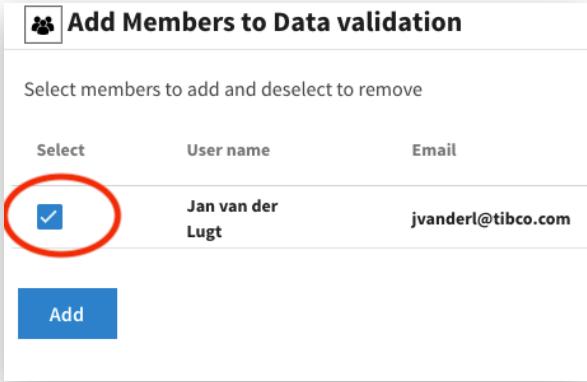
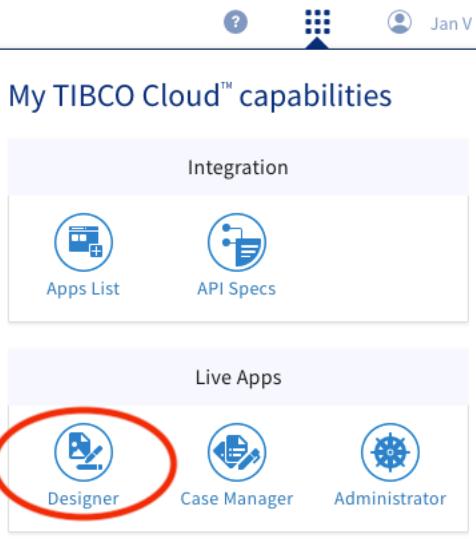
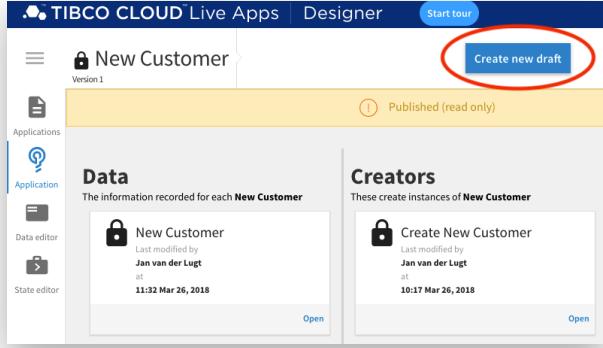
**Check out all features** of the case administration.



## Improve the App

### Assign tasks to groups

<p>Now we will <b>make some changes</b> to our existing process. One item is to <b>assign tasks to groups</b> of people.</p> <p>Select “Group Administration” from the administration menu.</p>	
<p>We will <b>create a group</b> for “Data validation” and one for “Credit checking”.</p> <p>Select “Create custom groups”</p> <p>Fill out group name and group description and press “Create”</p>	
<p>You will see a <b>confirmation screen</b> that the group has been created.</p> <p>From here you have the option to <b>start adding members</b> to the group directly.</p>	

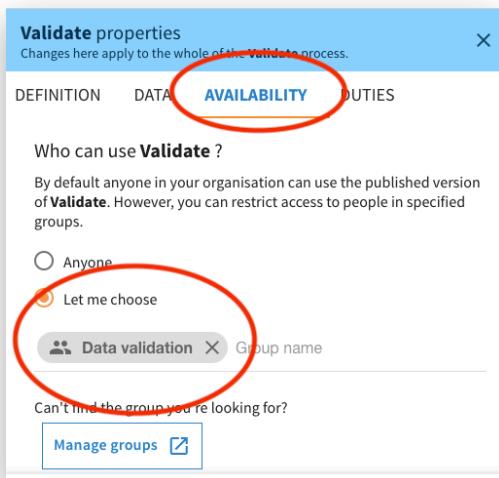
<p><b>Select the members</b> and press “Add”</p> <p>You will now see a tab with a group “DATA VALIDATION” and the list of members.</p> <p>Press “Create custom groups” again and repeat the steps to create a group called “Credit checking”.</p>	
<p><b>Now we will improve the project:</b> First item we'll address is to <b>assign tasks to groups</b>.</p> <p>From the main menu, select “Live Apps” and then “Designer”.</p> <p>Then select our “New Customer” app.</p>	
<p>In order to change a published app we need to <b>create a new draft</b> of the app.</p> <p>Click the corresponding button.</p>	

Select the “Validate” action and then “Action properties” to get to the place where you can **change properties** for the whole **action**.

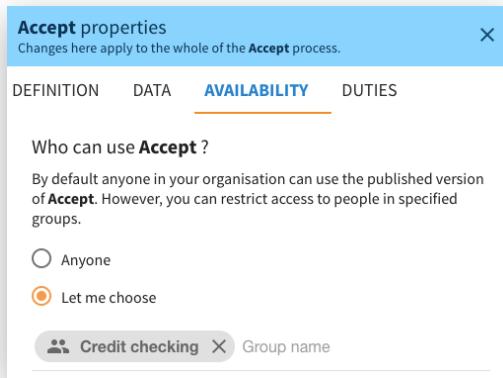


Now select the “AVAILABILITY” tab and “Let me choose”.

Then click on “Group name” and select the group “Data validation” you created earlier from the list.

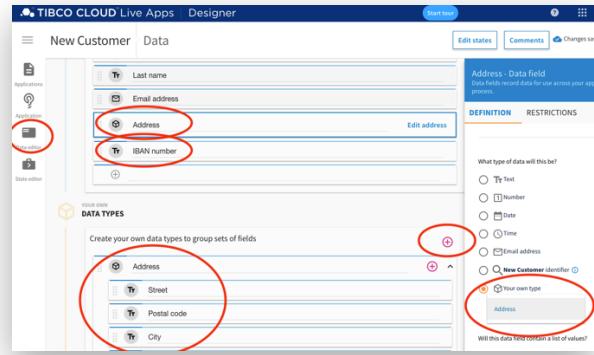


Similarly, Change the actions “Accept” and “Reject” to be available to group “Credit checking”.

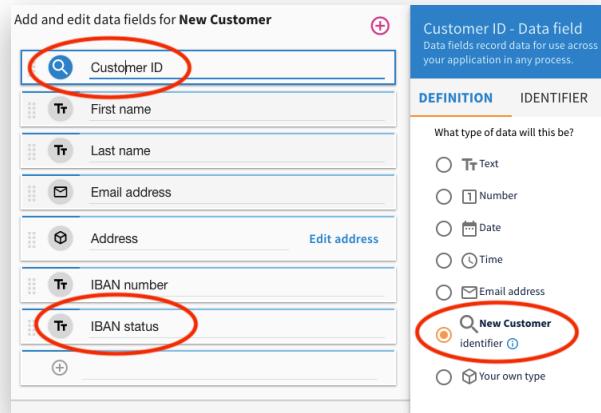


## Improve the data structure

Now let's **improve the data** used for a new customer.  
 Select "Data Editor" on the left.  
**Create a new data type** called "Address" by pressing the red "+" next to "Data Types".  
**Add fields** "Street", "Postal Code" and "City" to the data type.  
 Now add a data field called "Address" that is of type "Address". Select "Your own type" on the right, and select the data type you created.  
**Also add a text field** called "IBAN number" we will use that in the next step.



In that step we will create an automated step in the process that will check if a valid IBAN number has been entered. So in our data editor **we also need an IBAN status field** that is only accessible by the data validator person in the task after the automated check.  
**Also create a Customer ID**, which is an **automatically created identifier**.



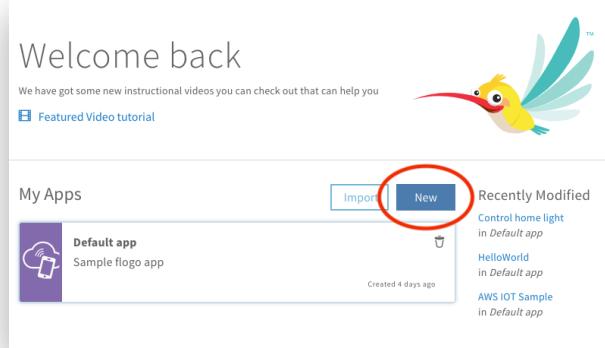
# Service Integration

## Create Flogo Service

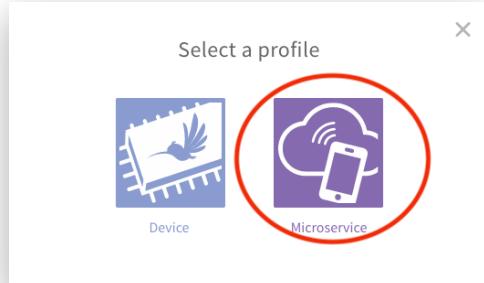
Now we will leave TIBCO Cloud Live Apps and concentrate on **building the IBAN check microservice in Flogo**.

First start flogo web by running:  
docker run -d -p 3303:3303  
flogo/flogo-docker eula accept.  
Then start a browser and open:  
<http://localhost:3303>

Then press “New” to start creating a new flogo app.

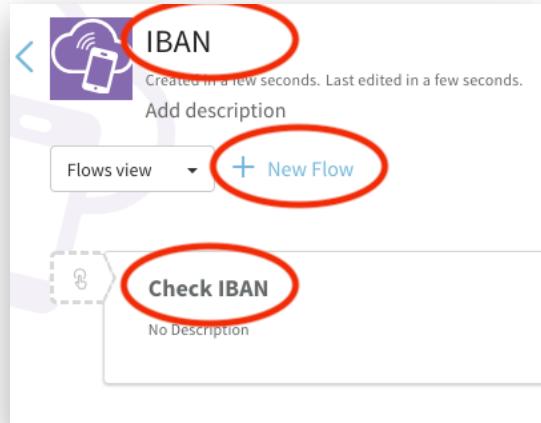


**Select profile “Microservice”**

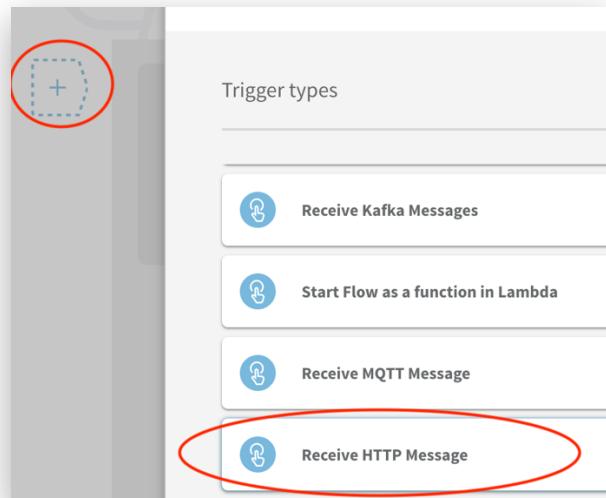


**Name the app “IBAN” and select “New Flow”.**

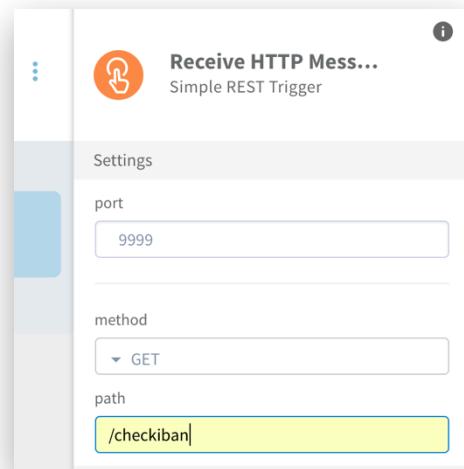
**Name your flow “Check IBAN”.**



As we want this flow to be used as a **restful microservice using REST over HTTP**, we will select “Receive HTTP Message as trigger”

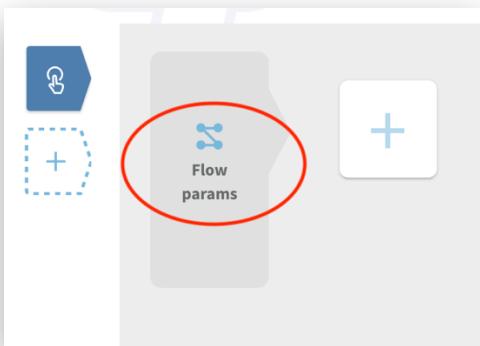


Then we **configure the trigger** to listen to port 9999, expose a “GET” operator, that can be found on “/checkiban”.



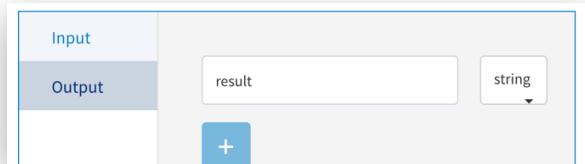
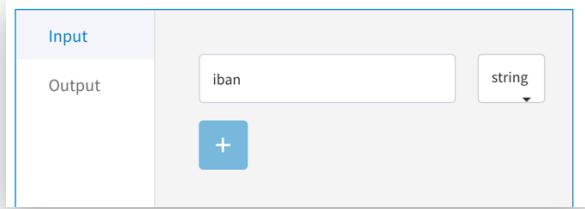
Then we **configure the inputs and outputs** for the flow.

The idea is to feed it an IBAN number and get a simple result string as return.



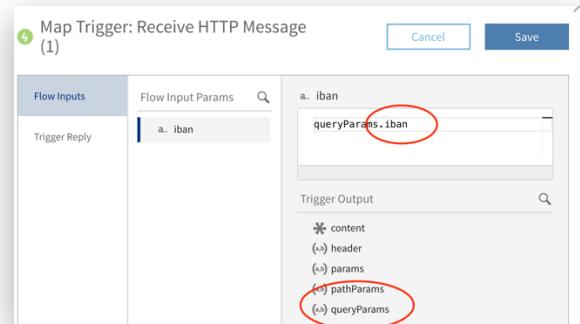
Here we **create the input and output parameters**.

We will map them next.



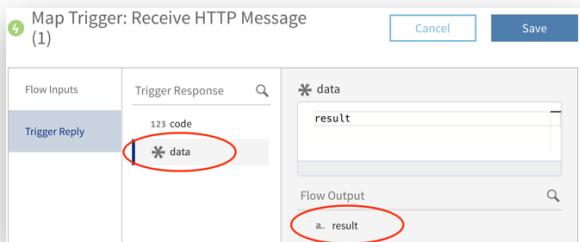
Now we will **map the input parameters**.

Input parameter "iban" will get its' value from queryparameter "iban" note the syntax in the mapper. We manually added ".iban" after selecting queryParams from the lower section



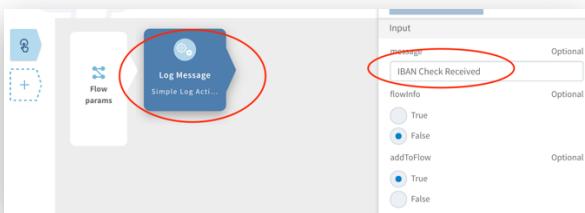
Now we will **map the output parameters**. An HTTP Receiver will always return a code and a data object.

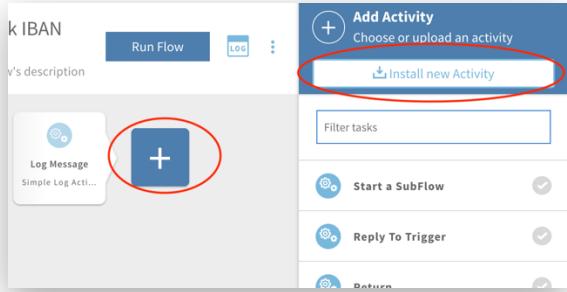
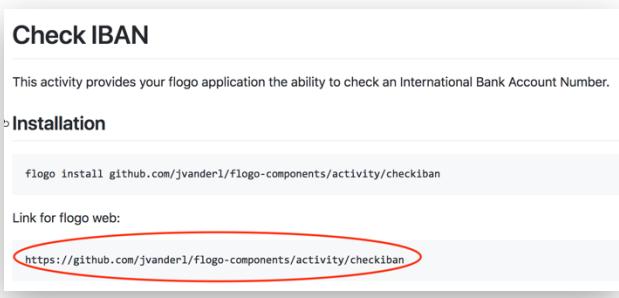
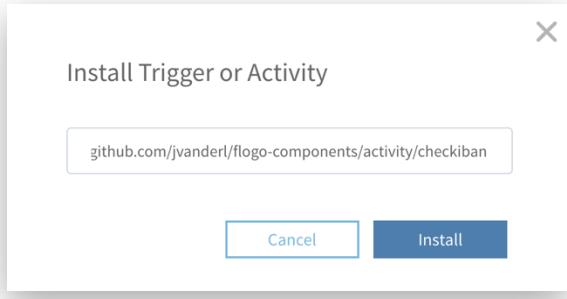
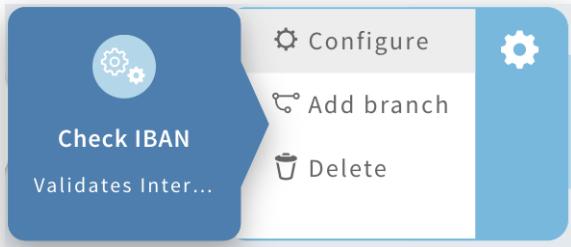
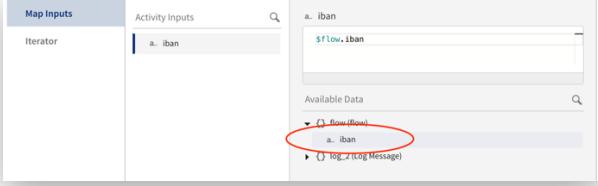
Here we will just map result string to this object.



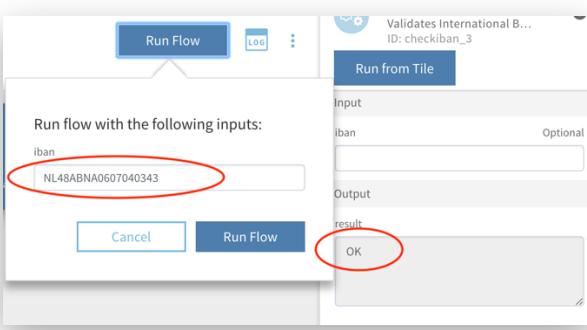
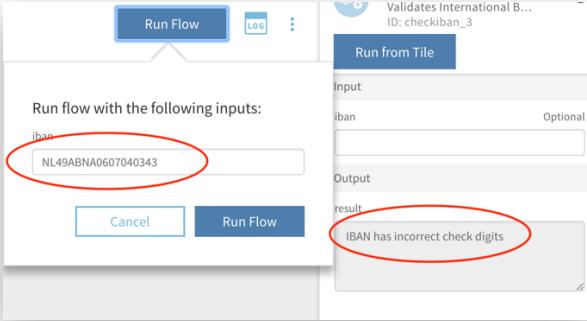
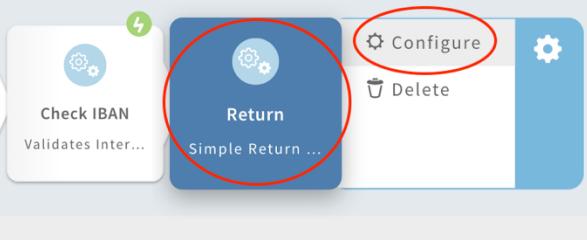
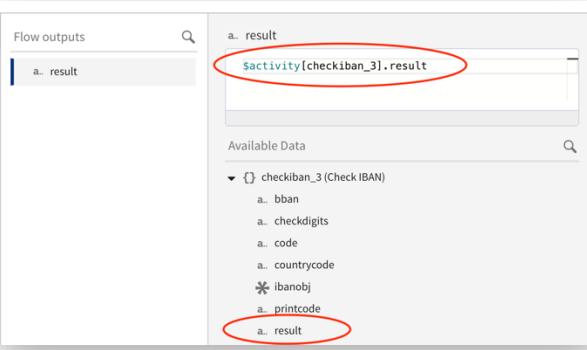
Now we start **creating the flow logic**.

Just for logging purposes we add a standard "Log Message" activity and fill out a fixed message "IBAN Check Received".

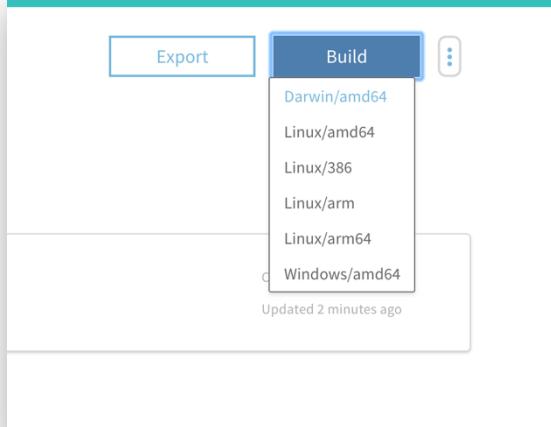
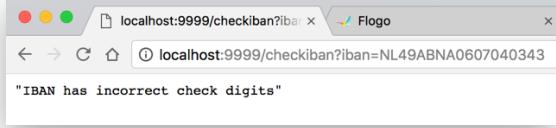
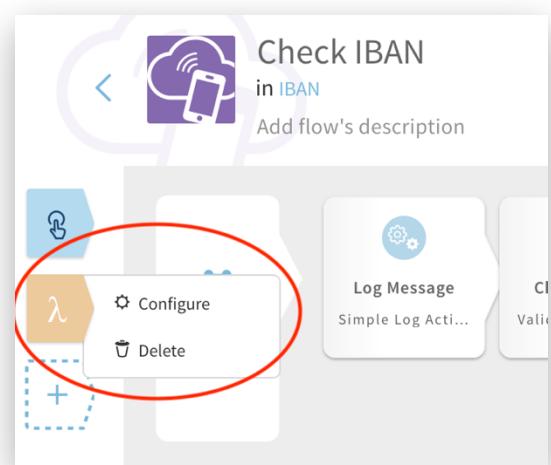


<p><b>Now we will add a custom activity</b> I created to do the actual check of the IBAN number. Press the “+” to add, and then “Install new Activity” to tell flogo where to find this new activity.</p>	
<p><b>Find the IBAN activity</b> on my github: <a href="https://github.com/jvanderl/flogo-components">https://github.com/jvanderl/flogo-components</a></p> <p><b>Copy the link location</b> for flogo web.</p>	
<p>Back in flogo web, in the pop-up box <b>paste the link location</b> to the IBAN activity and press “Install”</p>	
<p><b>Now we map the inputs</b> to the activity by selecting the gear logo on the “Check IBAN” activity, and then select “Configure”.</p>	
<p>In the mapper window select “<b>Map Inputs</b>”, activity input “iban” and then select the “\$flow.iban” to make sure we <b>get the iban number from the flow input</b>.</p>	

## Test the service functionality in Flogo

<p><b>Time to do the first test run!</b></p> <p>Click on “Run Flow” from the top of the screen. This will pop-up a window in which you can enter the input parameters for testing. Fill out a valid IBAN and press “Run Flow”. Check the result in the output on the right.</p>	
<p><b>Now run the test again with an invalid IBAN.</b></p> <p>Check out the result again.</p> <p>If this is all good we can finalize our flow.</p>	
<p>Here we <b>add</b> the last <b>activity</b> which is a <b>return</b> of the flow output back to the HTTP receiver.</p> <p>Select “+” to add and select “Return” from the list of available activities.</p> <p>Now press “Configure” to start mapping the result.</p>	
<p>Note the activity has many more outputs, but we only need “result” for our exercise.</p> <p>So click on “result” and then press “Save”</p>	

## Build and run the service executable

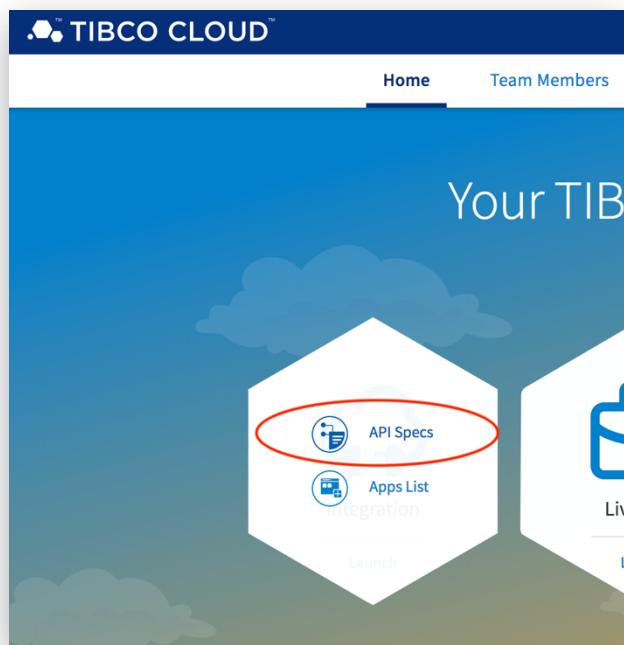
<p>Congrats, we just built our first microservice in flogo!</p> <p>Now we can <b>build a single executable</b> from this.</p> <p>From the highest level in the flogo app, press “Build” and select the target architecture of this service. In my case “Darwin/amd64” because I’m running a mac.</p> <p><b>Save the file</b> to a location on your local disk.</p>	
<p>On Mac, I need to <b>grant “execute” rights</b> to the microservice by issuing “chmod +x &lt;appname&gt;”.</p> <p>Now start the microservice.</p>	<pre>jans-mpb:iban jvanderl\$ chmod +x iban_darwin_amd64 jans-mpb:iban jvanderl\$ ./iban_darwin_amd64 2018-03-26 16:31:26.476 INFO [engine] - Engine Starting... 2018-03-26 16:31:26.477 INFO [engine] - Starting Services... 2018-03-26 16:31:26.477 INFO [engine] - Started Services 2018-03-26 16:31:26.477 INFO [engine] - Starting Triggers... 2018-03-26 16:31:26.477 INFO [engine] - Trigger [ receive_http_message ] started 2018-03-26 16:31:26.477 INFO [engine] - Triggers Started 2018-03-26 16:31:26.477 INFO [engine] - Engine Started</pre>
<p>To <b>check this works</b> issue a simple get from your browser: <a href="http://localhost:9999/checkiban?iban=NL49ABNA0607040343">http://localhost:9999/checkiban?iban=NL49ABNA0607040343</a>. Check the response.</p>	
<p>I prepared a public version of this flow running as FAAS on AWS Lambda.</p> <p>We will use this one but wrap it from TIBCO Web Integrator to make it available to our LiveApp.</p>	

## Create an API Specification in TCI

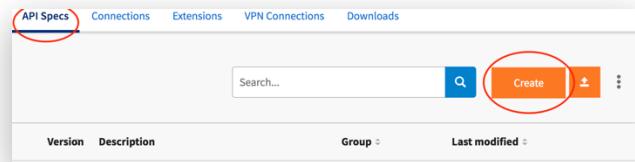
First, we will **create an API Spec** for TCI.

Go back to main section in  
<https://cloud.tibco.com>

Now hover over “Integration” and select “API Specs”



Press “Create” to **define a new API Spec.**



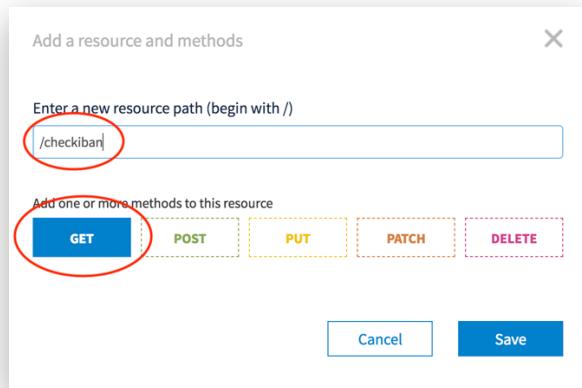
Fill out some details.  
Name the API “checkiban”.

Press “**Create**”.

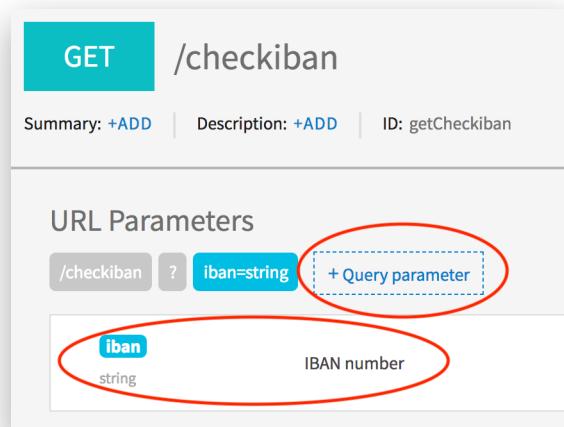
A screenshot of a modal dialog box titled 'Create an API specification'. It contains four input fields: 'API name' (with 'checkiban' typed in), 'Version' (with '1.0.0'), 'Select Group' (with 'Default'), and a 'Cancel' button. At the bottom right is a large, prominent 'Create' button, which is highlighted with a red circle.

Now we will add a resource path and a method.  
Make the pat “/checkiban” and select “GET” as method.

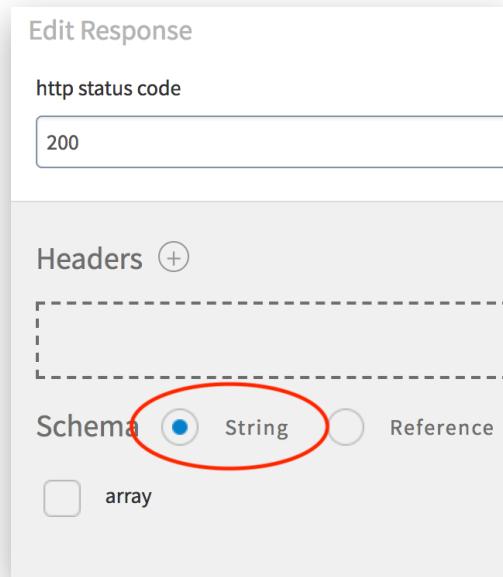
Press “Save”.



Now we'll add a query parameter string called “iban”.



Then make sure the result is a simple “String” type.



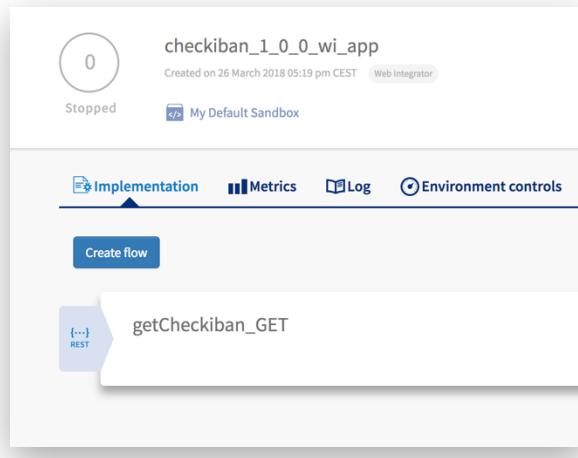
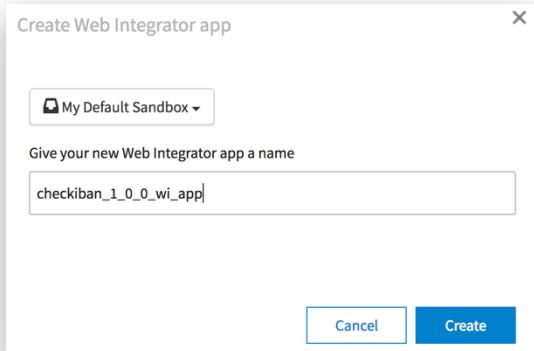
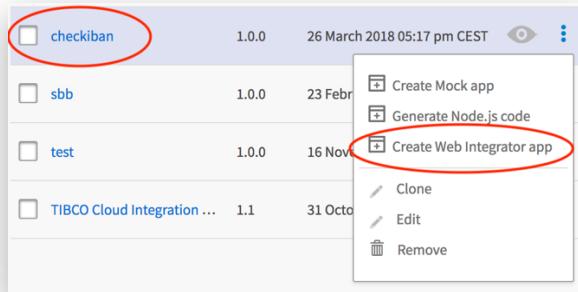
## Create a TCI Wrapper Service using Web Integrator

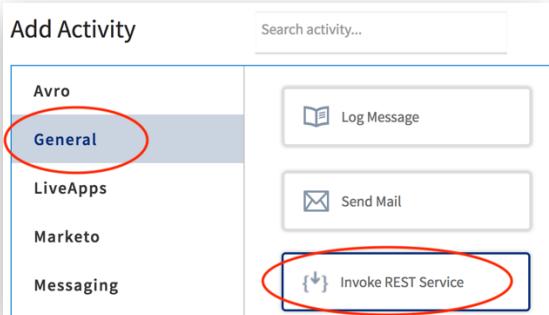
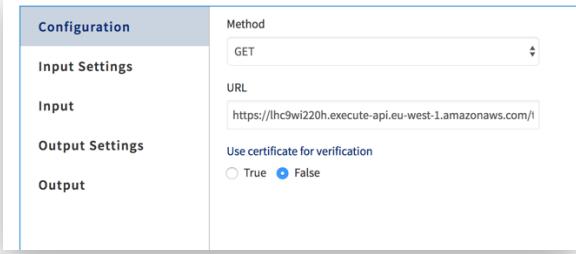
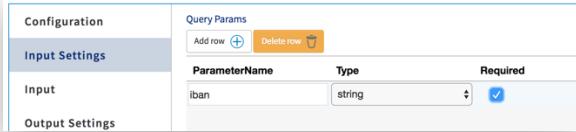
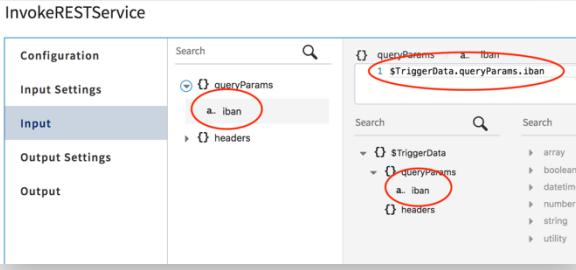
So now we've created the API definition, we can easily **create a Web Integrator app** from here.

Select the “checkiban” API definition, and select “Create Web Integrator app” from the list you see when you click on the 3 vertical dots on the far right.

On the resulting pop-up leave everything default and press “**Create**”.

Now we'll have an empty Web Integrator flow with all the inputs and outputs pre-mapped.



<p>Click on “getCheckiban_GET” to open the flow definition and <b>add an activity for invoking our AWS lambda based REST service.</b></p> <p>First select “General” then “Invoke REST Service”.</p>	
<p>Move the invoke activity between the existing HTTP activities. And select it to show the activity configuration.</p>	
<p>In “Configuration” select method “GET” and use the following URL:</p> <p><a href="https://lhc9wi220h.execute-api.eu-west-1.amazonaws.com/tst/checkiban">https://lhc9wi220h.execute-api.eu-west-1.amazonaws.com/tst/checkiban</a></p>	
<p>In “Input Settings”, <b>create a string parameter for iban</b> and set it to “Required”.</p>	
<p>In “Input”, <b>map the input iban</b> to the newly created iban input for the REST invoke activity.</p>	

Before we can map the output we first need to know what the original service responds with. I use **Postman** to call the AWS Lambda service, but a browser will also do.

**Copy the response** you're your clipboard.

```

{
  "iban": {
    "BBAN": "ABNA0607040343",
    "CheckDigits": "48",
    "Code": "NL48ABNA0607040343",
    "CountryCode": "NL",
    "CountrySettings": {
      "Format": "UO4E10",
      "Length": 18
    },
    "PrintCode": "NL48 ABNA 0607 0403 43"
  },
  "result": "OK"
}

```

Now back in Web Integrator, under “Output Settings” **paste the example data**.

Web Integrator will derive the message structure automatically so you can easily map the output in the next step.

```

{
  "iban": {
    "BBAN": "ABNA0607040343",
    "CheckDigits": "48",
    "Code": "NL48ABNA0607040343",
    "CountryCode": "NL",
    "CountrySettings": {
      "Format": "UO4E10",
      "Length": 18
    }
  }
}

```

Now select the “Reply to HTTP Message” activity to open up its configuration.

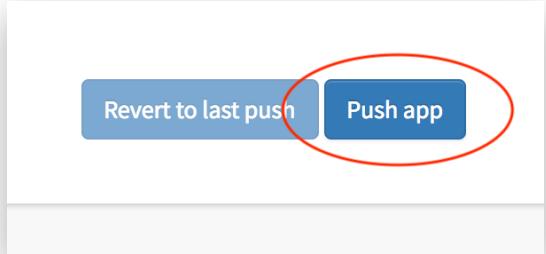
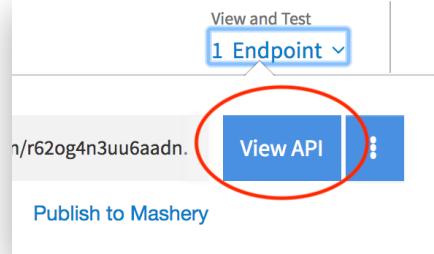
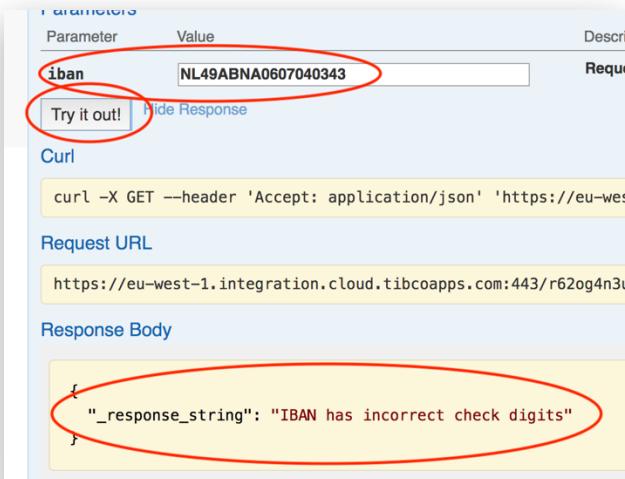


Under “Input”, make sure the “**result**” from the AWS API call is **mapped to “\_response\_string”**.

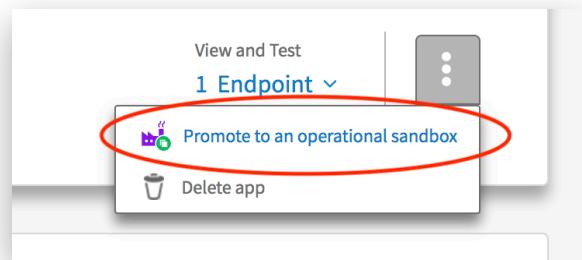
Configuration	Search
Input Settings	data a. _response_string
Input	data a. _response_string

Search	Search
\$invokeRESTService 1. statusCode 2. responseBody 3. iban 4. result 5. headers 6. error 7. \$triggerData	array boolean datetime number string utility

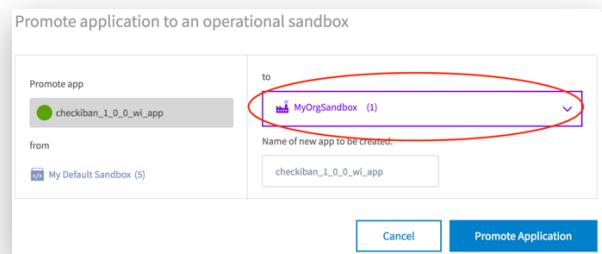
## Run and test the Web Integrator App

<p>All Set, now we can <b>create a running instance</b> of this Web Integrator Microservice by pressing “Push app” on the main screen.</p>	
<p><b>Follow the workflow</b> while in the background this microservice is updated, built and deployed. You should have “1/1 Running” when deployment is done.</p>	
<p>Now from the running Web Integrator app we can <b>check out the API</b>. Press “View API” on the right.</p>	
<p>This will pop-up the API screen. Fill out a false IBAN and press “<b>Try it out!</b>”  <b>See the result</b> in the response Body.</p>	

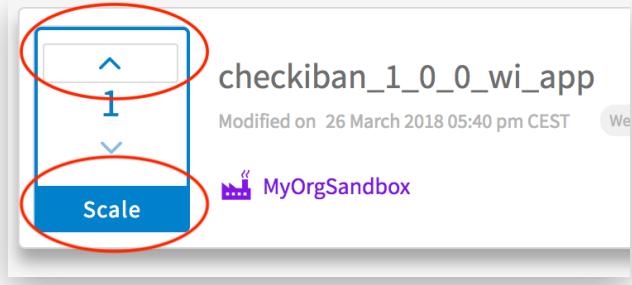
When all is good, we can **promote the microservice to an operational sandbox** so it gets exposed to LiveApps.  
Press the icon with 3 vertical dots on the side and select “Promote to an operational sandbox”



Select “MyOrgSandbox” and press “Promote Application”.



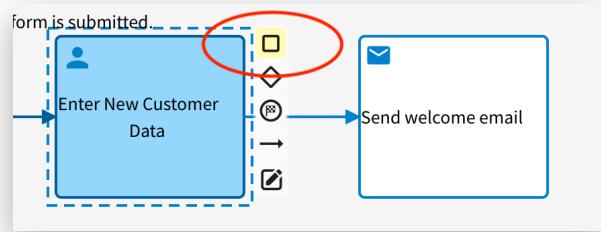
Make sure we have a running instance by **scaling the app to 1**.



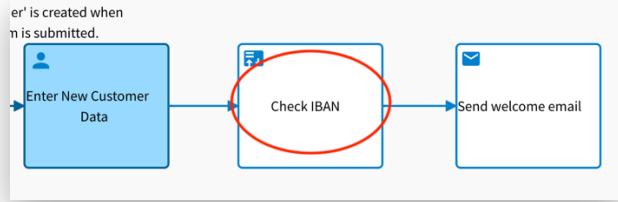
## Embed the service interaction in the Live App

Now all we need to do is **change the LiveApp Creator** for New Customer to embed the IBAN check.  
Back in Live Apps Designer, select our “New Customer” application and edit the creator “Create New Customer”.

**Insert a new task** by pressing the little square icon on the top right of task “Enter New Customer Data”.



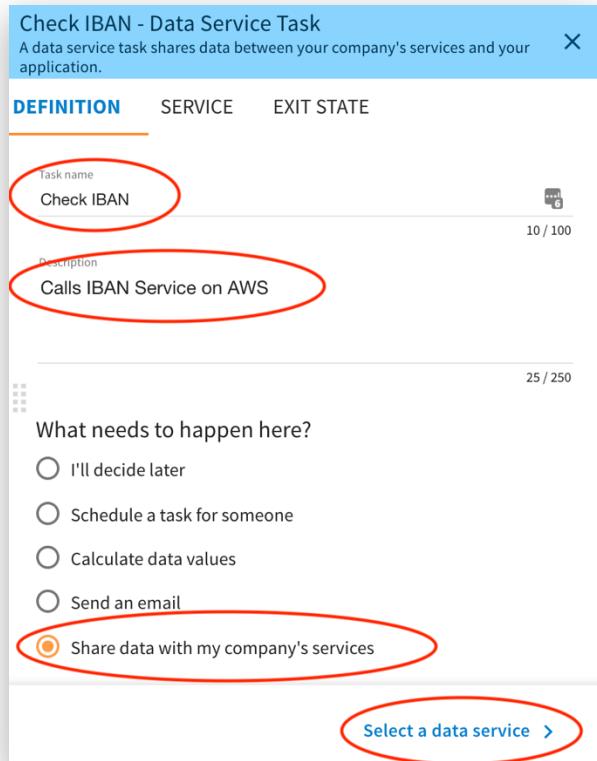
**Drop the new task** on the line **between the two existing tasks** and click on it to open its configuration.



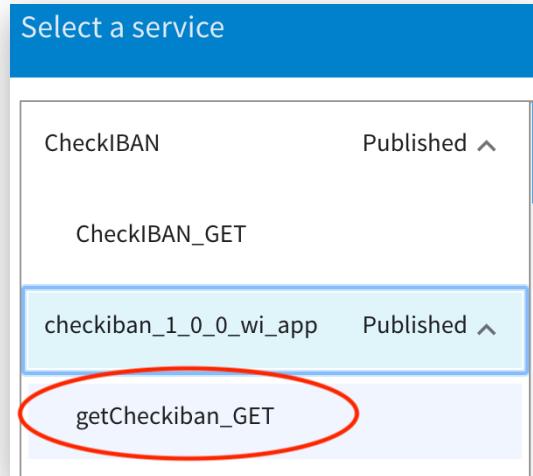
Here we set the name “Check IBAN” and optionally enter a description.

Select “Share data with my company’s services” to tell Live Apps we are calling an automated service here.

Press “Select a data service”.

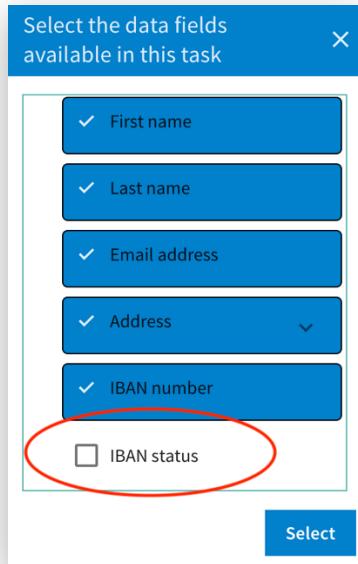


Now select our recently promoted Web Integrator service.

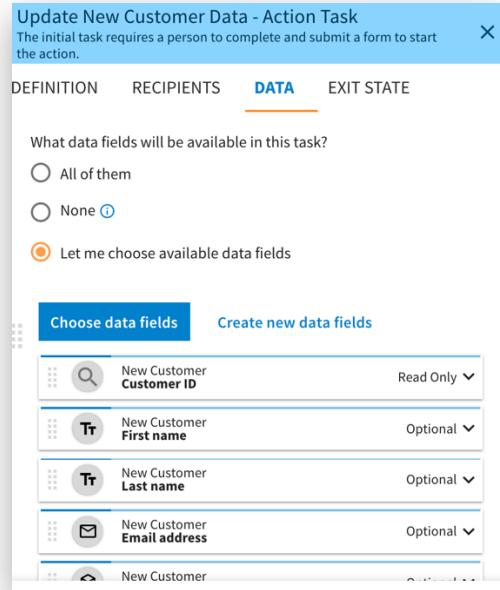


<p><b>Map the service input “iban” to New Customer data field “IBAN number”.</b></p>	<p>Check IBAN - Data Service Task A data service task shares data between your company's services and your application.</p> <p>DEFINITION SERVICE <b>INPUTS</b> OUTPUTS EXIT STATE</p> <p>What inputs will your app provide to the service? Decide what fields to assign from your app to the company data service.</p> <table border="1"> <thead> <tr> <th>Service fields</th> <th>Your data fields</th> </tr> </thead> <tbody> <tr> <td>iban</td> <td>Assign from New Customer ▾ IBAN number</td> </tr> </tbody> </table>	Service fields	Your data fields	iban	Assign from New Customer ▾ IBAN number		
Service fields	Your data fields						
iban	Assign from New Customer ▾ IBAN number						
<p><b>Map the service output “_response_string” to New Customer data field “IBAN status” to make sure the result becomes part of the customer data.</b></p>	<p>Check IBAN - Data Service Task A data service task shares data between your company's services and your application.</p> <p>DEFINITION SERVICE INPUTS <b>OUTPUTS</b> EXIT STATE</p> <p>What outputs will the service provide to your app? Decide what fields to assign from the company data service to your app.</p> <table border="1"> <thead> <tr> <th>Service fields</th> <th>Your data fields</th> </tr> </thead> <tbody> <tr> <td>Response status</td> <td>Assign to ▾</td> </tr> <tr> <td>Response/_response_string</td> <td>Assign to ▾ New Customer ▾ IBAN status</td> </tr> </tbody> </table>	Service fields	Your data fields	Response status	Assign to ▾	Response/_response_string	Assign to ▾ New Customer ▾ IBAN status
Service fields	Your data fields						
Response status	Assign to ▾						
Response/_response_string	Assign to ▾ New Customer ▾ IBAN status						
<p>In our case, the <b>result</b> is only needed while we are validating the entered data, so we can <b>hide it from the other states</b>. Select “Accept” from the actions, then select task “Update New Customer Data” and under tab “DATA” select “Let me choose available data fields” and press “Choose data fields”.</p>	<p>Update New Customer Data - Action Task The initial task requires a person to complete and submit a form to start the action.</p> <p>DEFINITION RECIPIENTS <b>DATA</b> EXIT STATE</p> <p>This is where your action starts and changes to ‘New Customer’ are entered by the user. The action is confirmed when the task form is submitted.</p> <p>What data fields will be available in this task?</p> <ul style="list-style-type: none"> <li><input type="radio"/> All of them</li> <li><input type="radio"/> None</li> <li><input checked="" type="radio"/> Let me choose available data fields</li> </ul> <p>Choose data fields Create new data fields</p>						

Now select all fields except "IBAN status" and press "Select".



Now the tasks' data section displays the selected data fields.



## Run a final test

All there's left to do is a **final test run**.

From the application screen, select "Test application" and fill out some data.

Notice the difference since last time we tried:

- Customer ID is generated automatically
- Address is added and grouped
- IBAN number is added
- IBAN result does not show yet.

N New New Customer

Customer ID	000001
First Name	Jan
Last Name	van der Lugt
Email Address	jlugtdemo@gmail.com
<b>Address</b>	
Street	Lichtenauerlaan 122-140
Postal Code	3062ME
City	Rotterdam
IBAN Number	NL48ABNA0607040343

Next, during validation state, you can see the automated **IBAN check service was called**, as you can see by the IBAN status.

Milestones

Entered ✓ Welcome email sent ✓

Customer ID  
000001

First name  
Jan

Last name  
van der Lugt

Email address  
jlugtdemo@gmail.com

Address

Street  
Lichtenauerlaan 122-140

Postal code  
3062ME

City  
Rotterdam

IBAN number  
NL48ABNA0607040343

IBAN status OK

## Links Used:

TIBCO Cloud

<https://cloud.tibco.com/>

Flogo Web

<http://localhost:3303>

Local Flogo test “OK”

<http://localhost:9999/checkiban?iban=NL48ABNA0607040343>

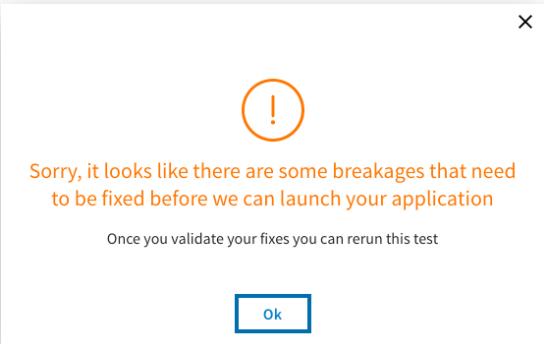
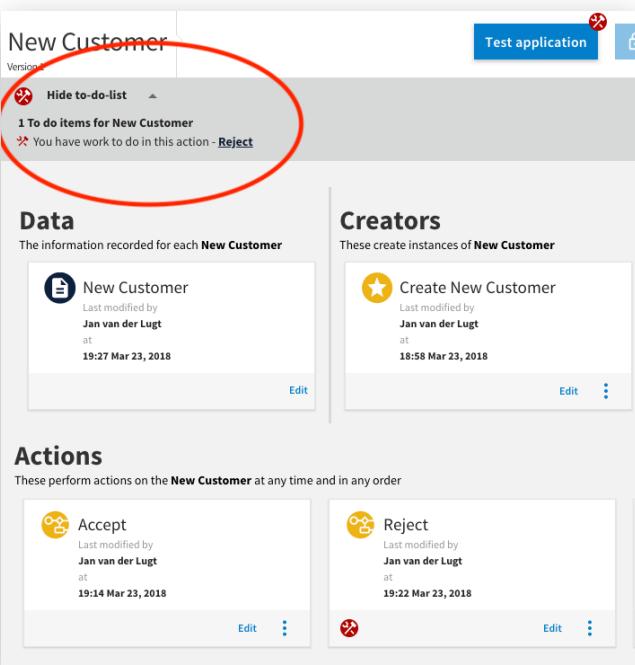
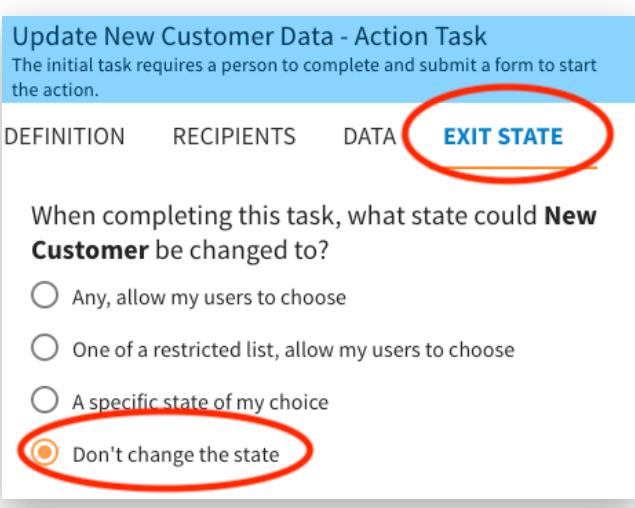
Local Flogo test “Incorrect Check Digits”

<http://localhost:9999/checkiban?iban=NL49ABNA0607040343>

AWS Lambda based service “OK”

<https://lhc9wi220h.execute-api.eu-west-1.amazonaws.com/tst/checkiban?iban=NL48ABNA0607040343>

## Optional: Error when testing

<p>So the some errors were found that need to be fixed.</p> <p><b>Click “OK” to check out what’s wrong.</b></p>	
<p>Notice the red icons indicating this is where we need to fix somtething. Click on the small triangle near “View to-do list” and click on “Reject” from there.</p> <p>The error now states we cannot set an exit state when there’s a follow-up task after.</p> <p>Click on the red icon in the “Update New Customer Data”.</p>	
<p>Select tab “Exit State” and select “Don’t change the state”</p>	

Now select the “Send rejection email” task.

Change “EXIT STATE” to specific state “Rejected”.

