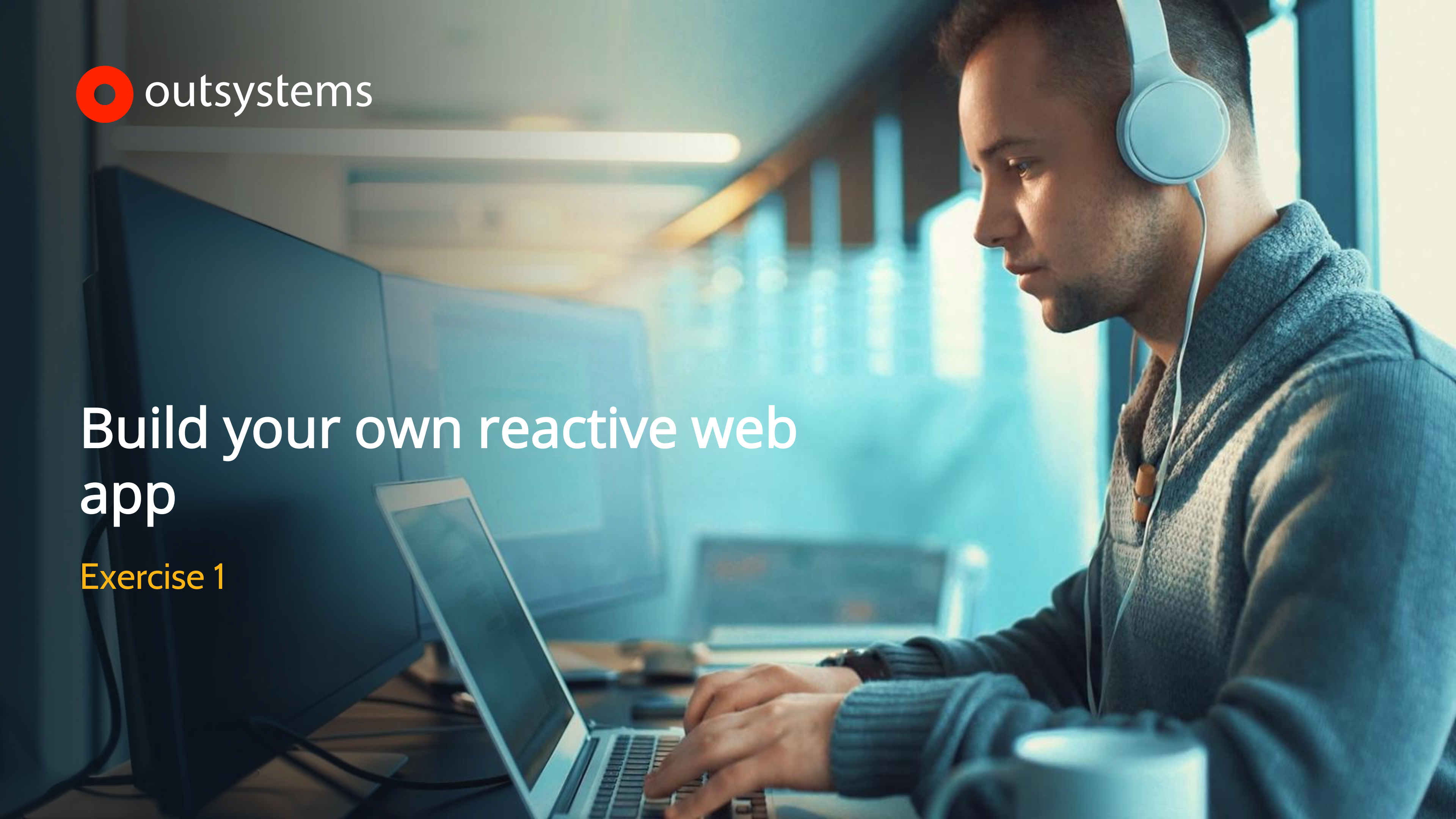


Build your own reactive web app

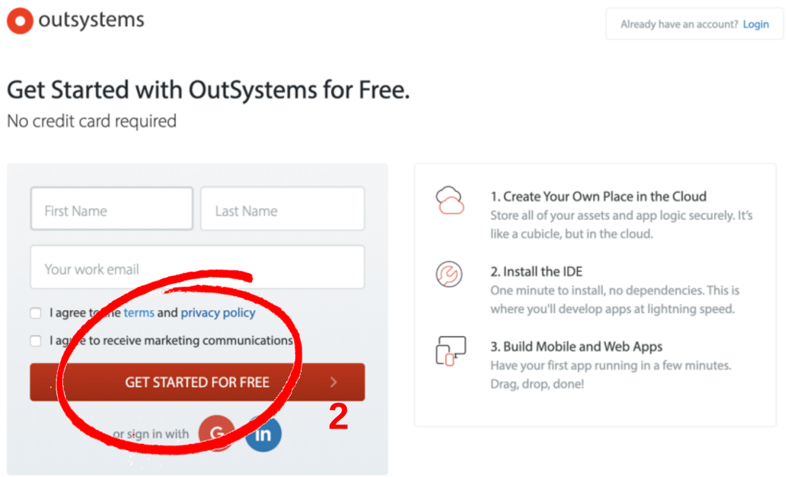
Exercise 1

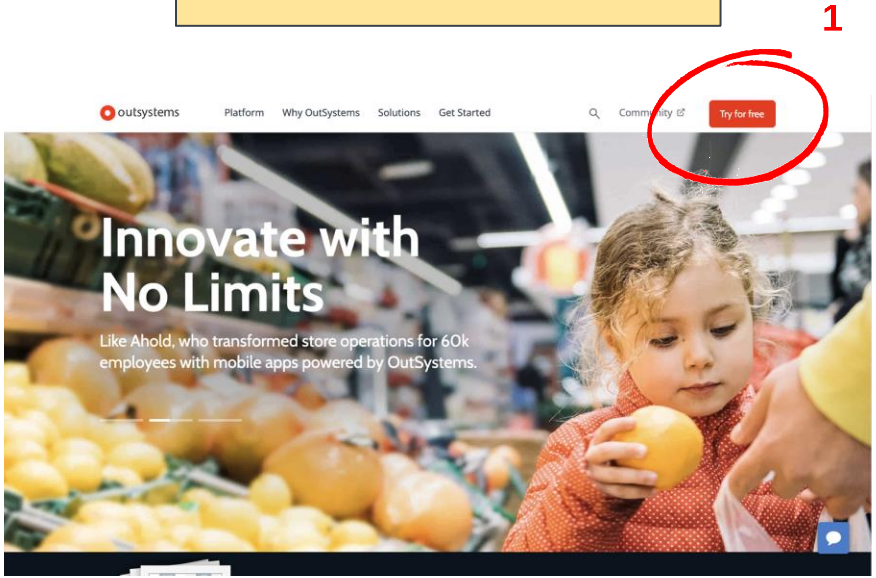
Get your personal environment

Register & Download

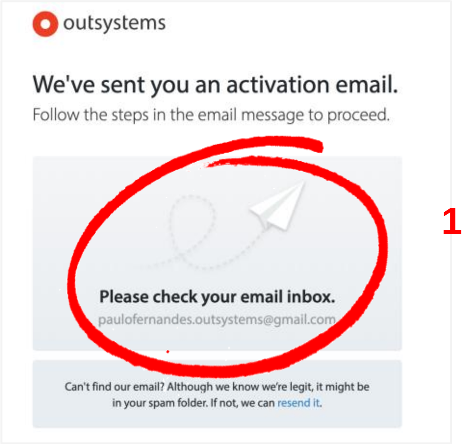
If you have done this already, skip to page 9 to begin the exercise.

Go to **www.outsystems.com** and get yourself a personal Environment by providing your information and follow the indicated steps.

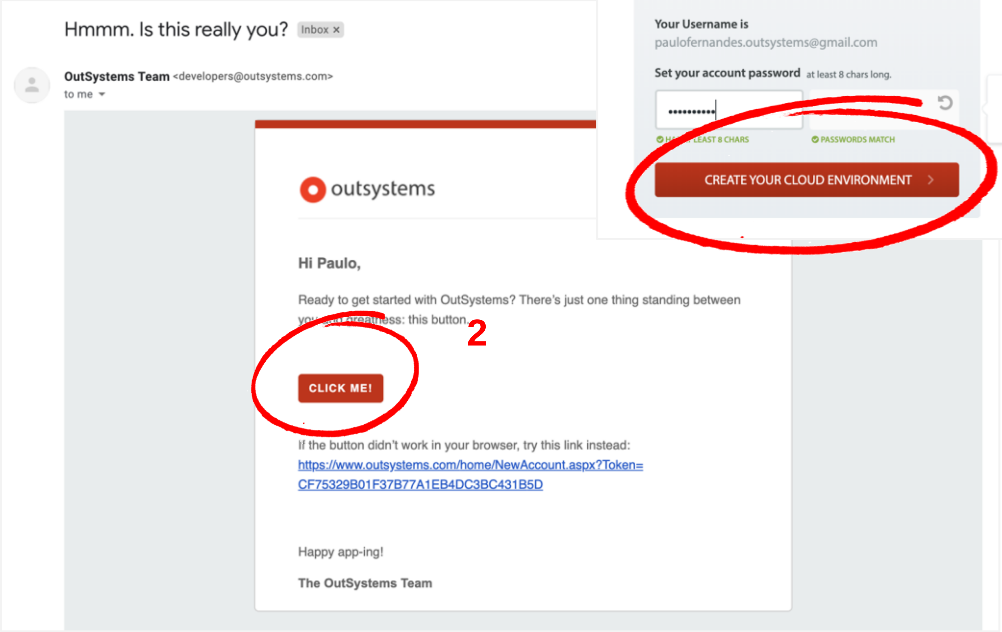




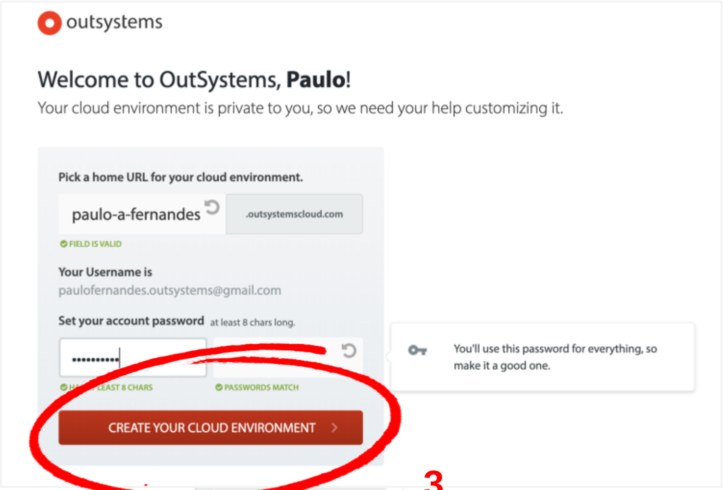
Register & Download



1



2

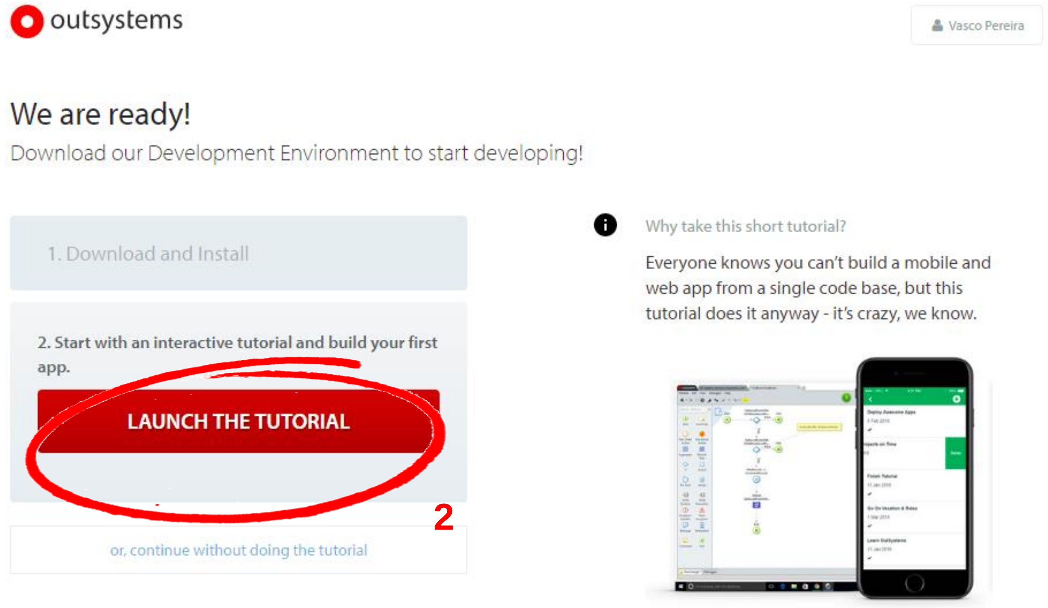


3

Note your **cloud environment address**, username and password. Don't forget it :)

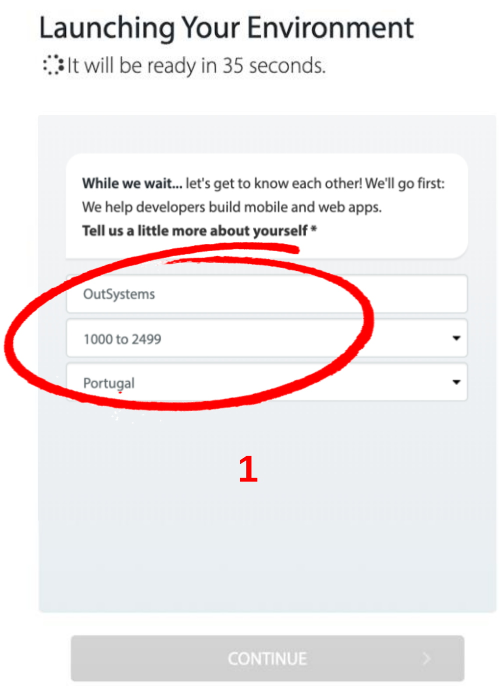
Development Studio

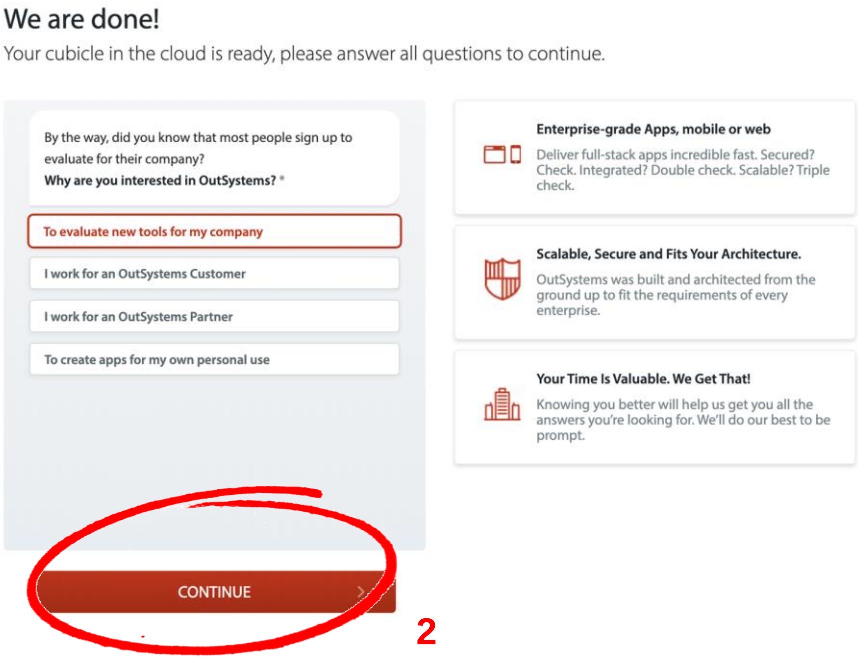
Install and launch



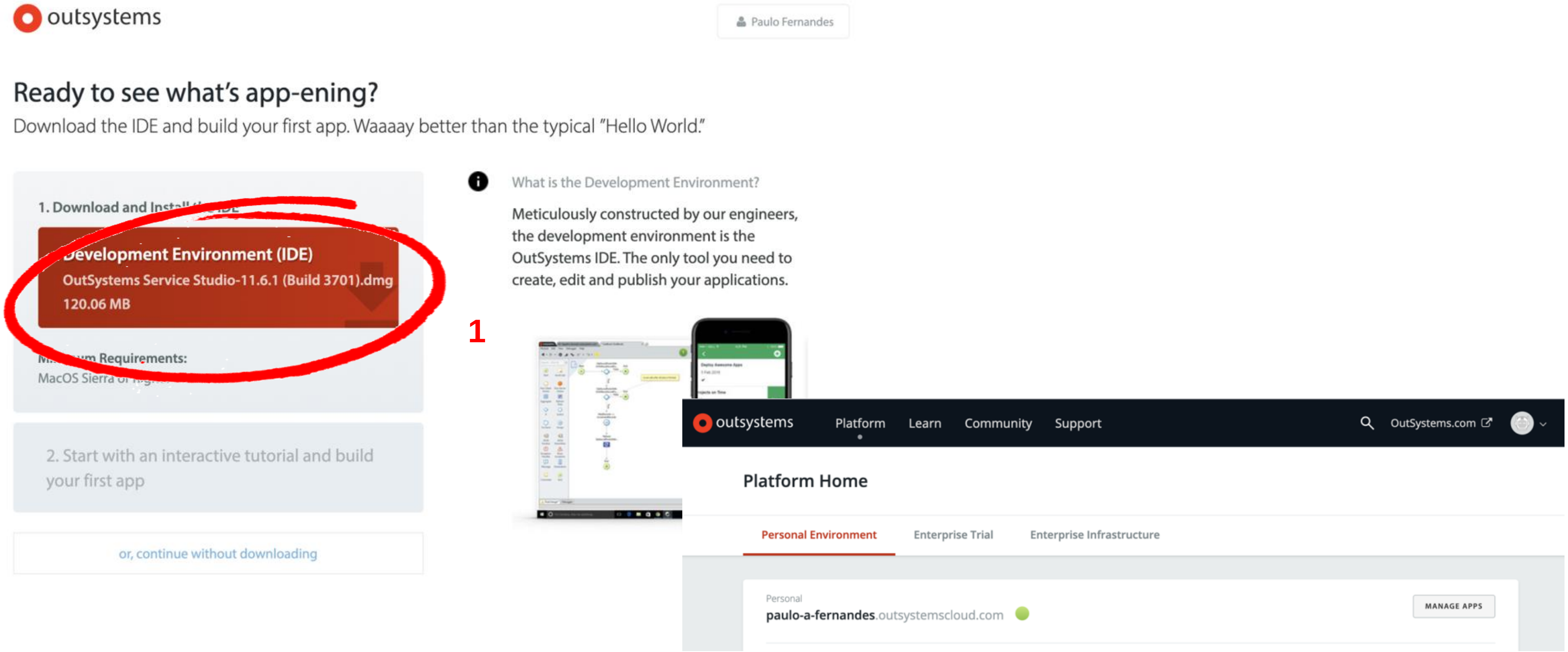
Register & Download

outsystems





Register & Download



Download Jump Start Materials

1. Bitte rufen Sie folgenden Link auf unserer Webseite auf:

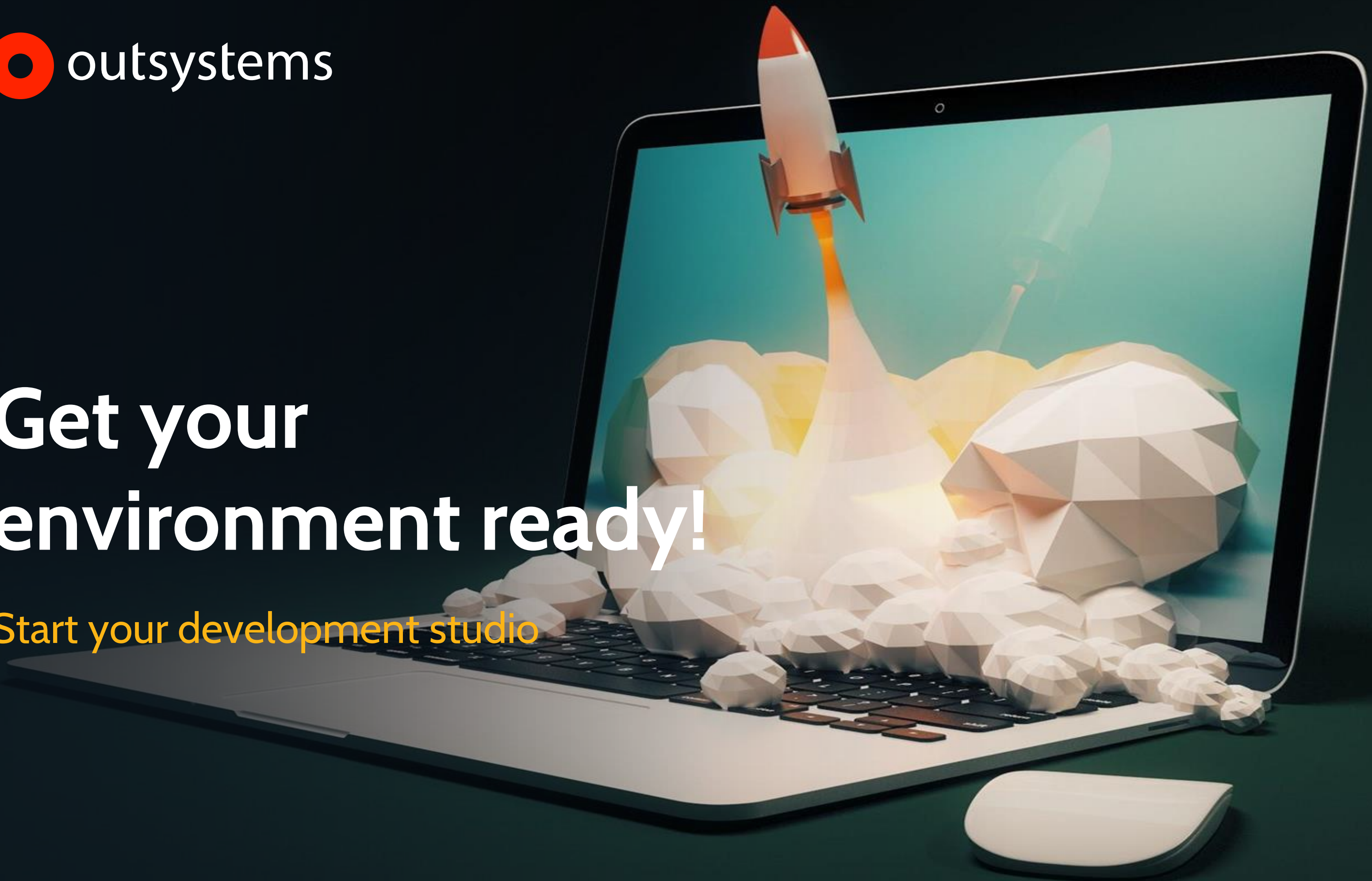
   https://www.agentbase.de/aktionen/vorbereitung-outsystems-jump-start-workshop/

2. Laden Sie alle Materialien herunter

outsystems

Get your environment ready!

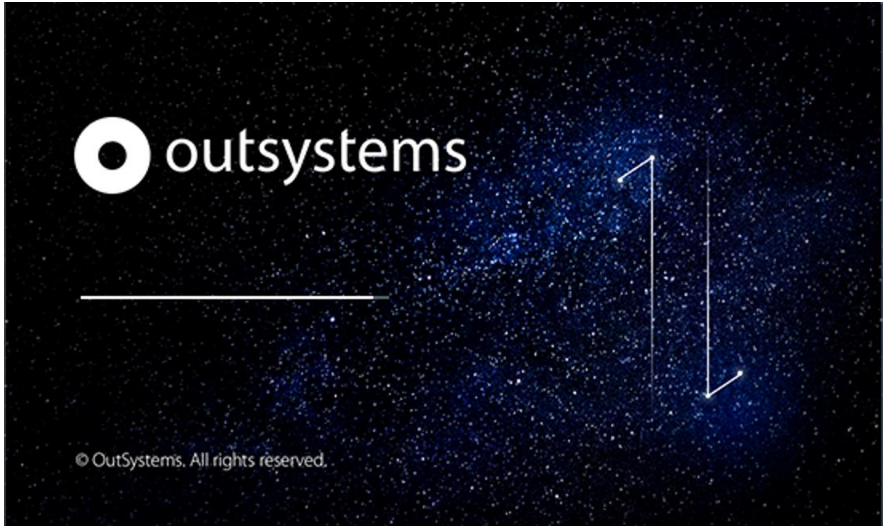
Start your development studio



Development Studio

Open and start

1. Open **Development Studio (Service Studio)**
2. Be awesome and **build your own web and mobile apps**.



Web Application

Exercise 1

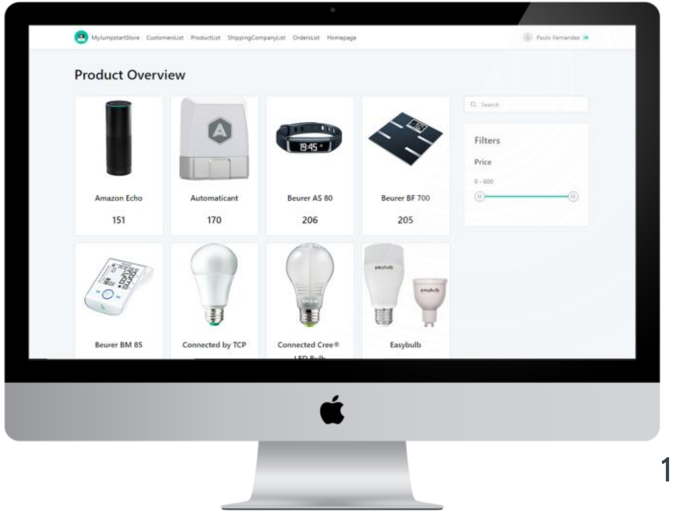
Context

“Company ACME Goods is a retail distribution company that acquires its products directly from the producers and sell them to small/medium shops, although is considering to sell them directly as well.

ACME Goods is looking for an application to manage their existing customer base information, manage their inventory and relations with their shipping companies.

Main Needs:

Customer Information Management
Inventory management
Shipping Companies management

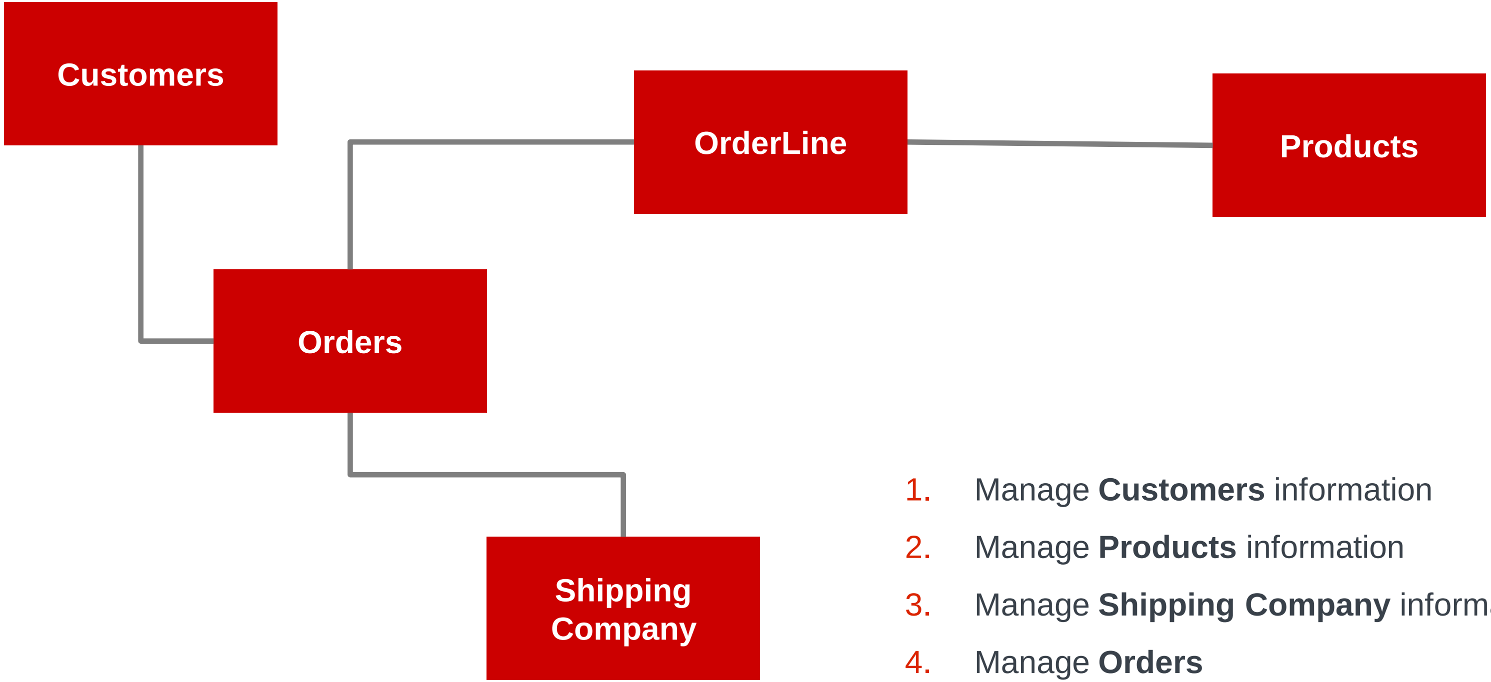


“As a **customer representative** I should be able to search for customers, edit their information, create new customers and register their purchases.”

As a **inventory manager** I should be able to access the products available, register new products, edit information about a product and associate products with suppliers.

Application Design

Basic Information Architecture



1. Manage **Customers** information
2. Manage **Products** information
3. Manage **Shipping Company** information
4. Manage **Orders**

outsystems

Section 1

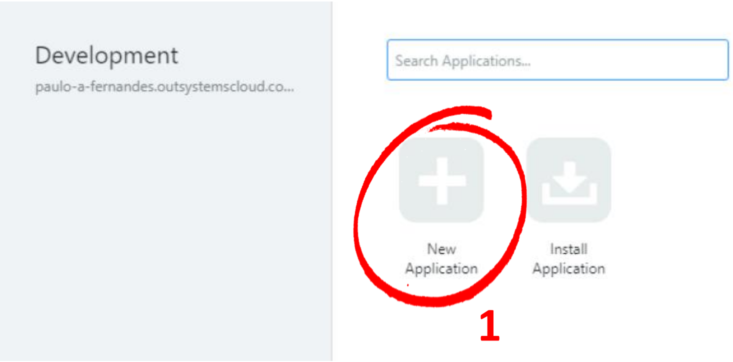
Customer Management

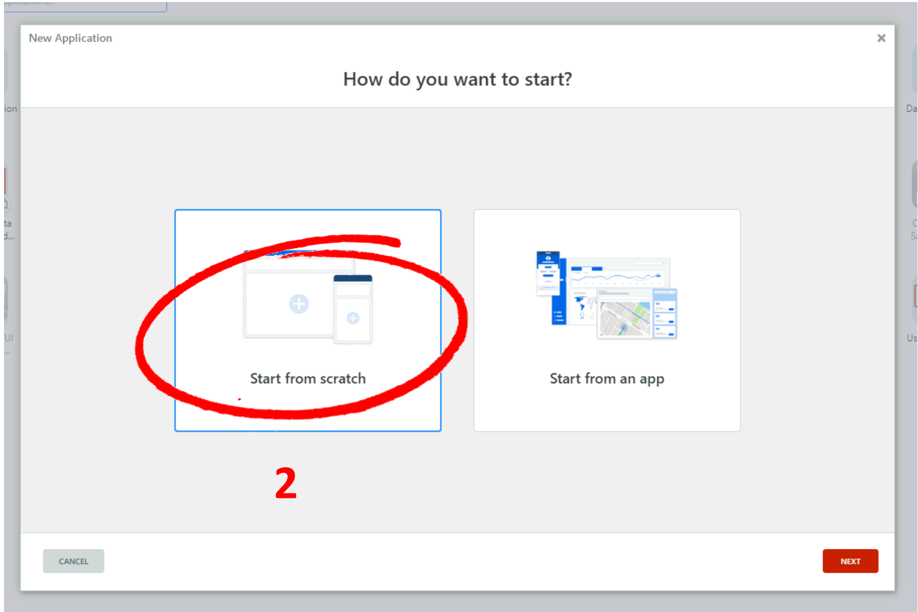
In the following steps you'll learn how to create your own responsive web application.

An application can contain one or more modules that act as a means of subdividing components. We will create a first module for our application.

Section 1 › A › 1. Create a New App

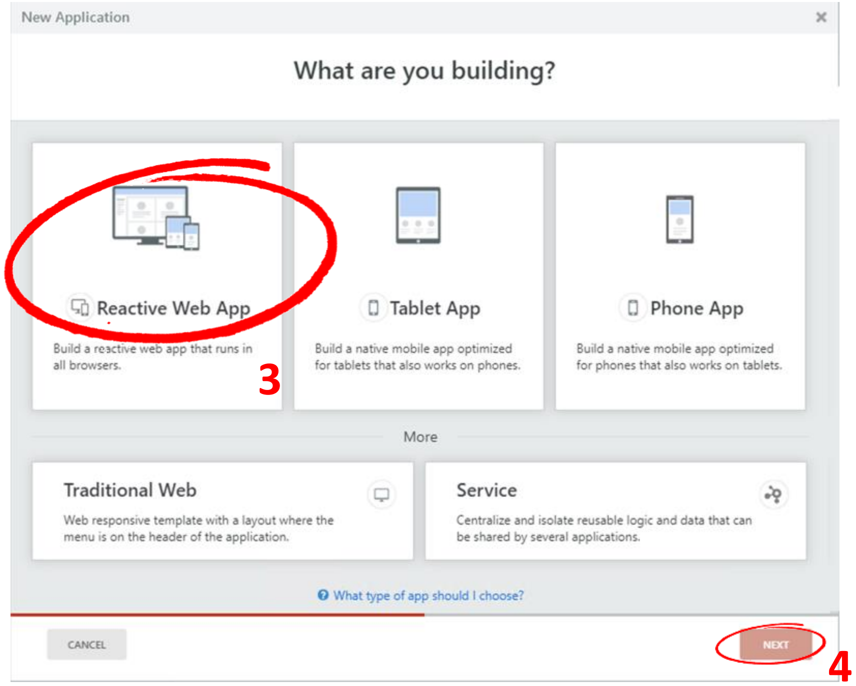
Review applications on our environment and Create a new web app





Section 1 A 1. Create a New App

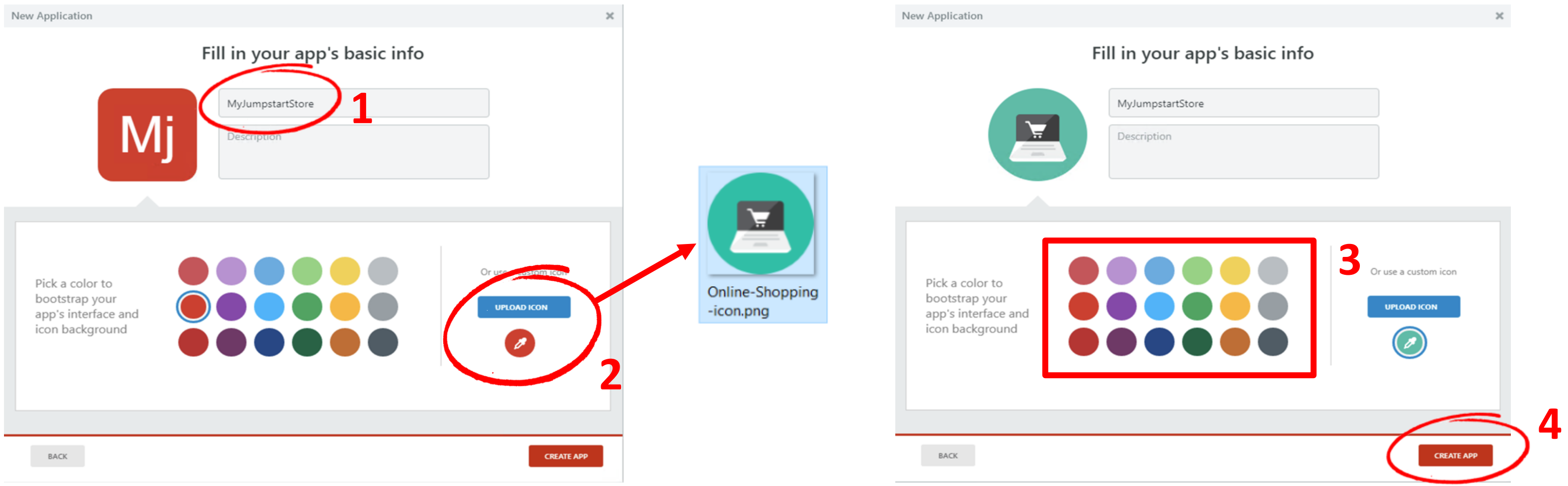
Review applications on our environment and Create a new web app



*You can select Web (Reactive & Traditional), Mobile (Tablet & Phone) App or Service. You should **choose Reactive Web App**, or choose mobile when you want to create an optimized experience for mobile devices, with touch friendly behaviors, etc and leverage the device features such as geo location, camera, etc*

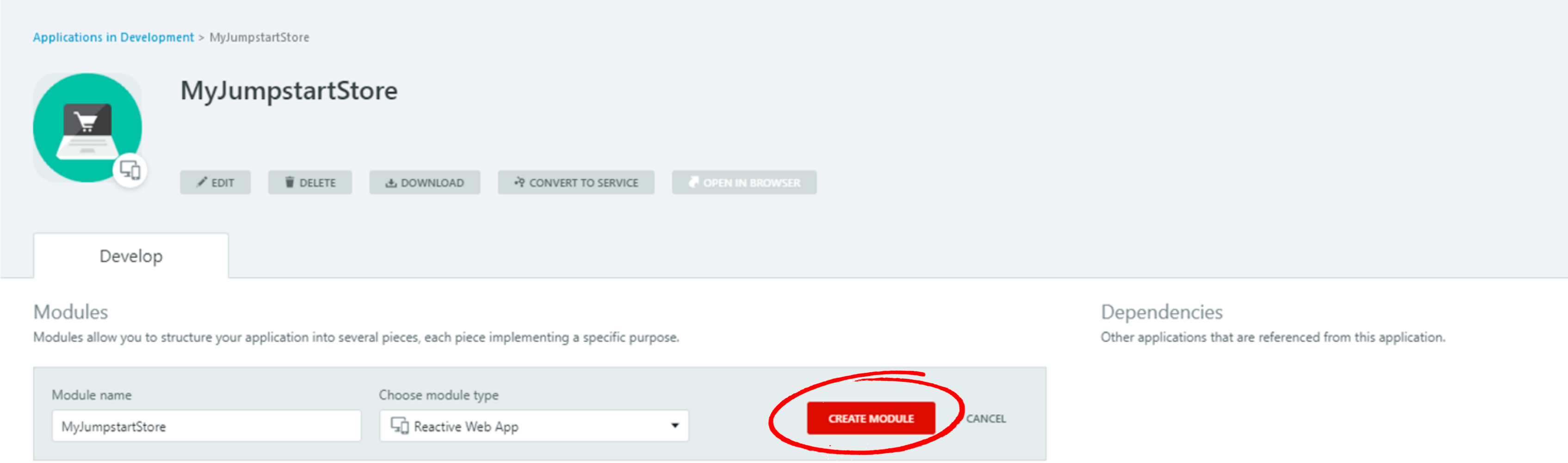
New Application

1. Type **Name** of your application “**MyJumpstartStore**”
2. Upload the **icon** provided from Jumpstart Resources Material folder
3. Color palette will be automatically determine based on icon color. Select a **color palette**, if you want to change.
4. Click “Create App”



Create Reactive Web App Module

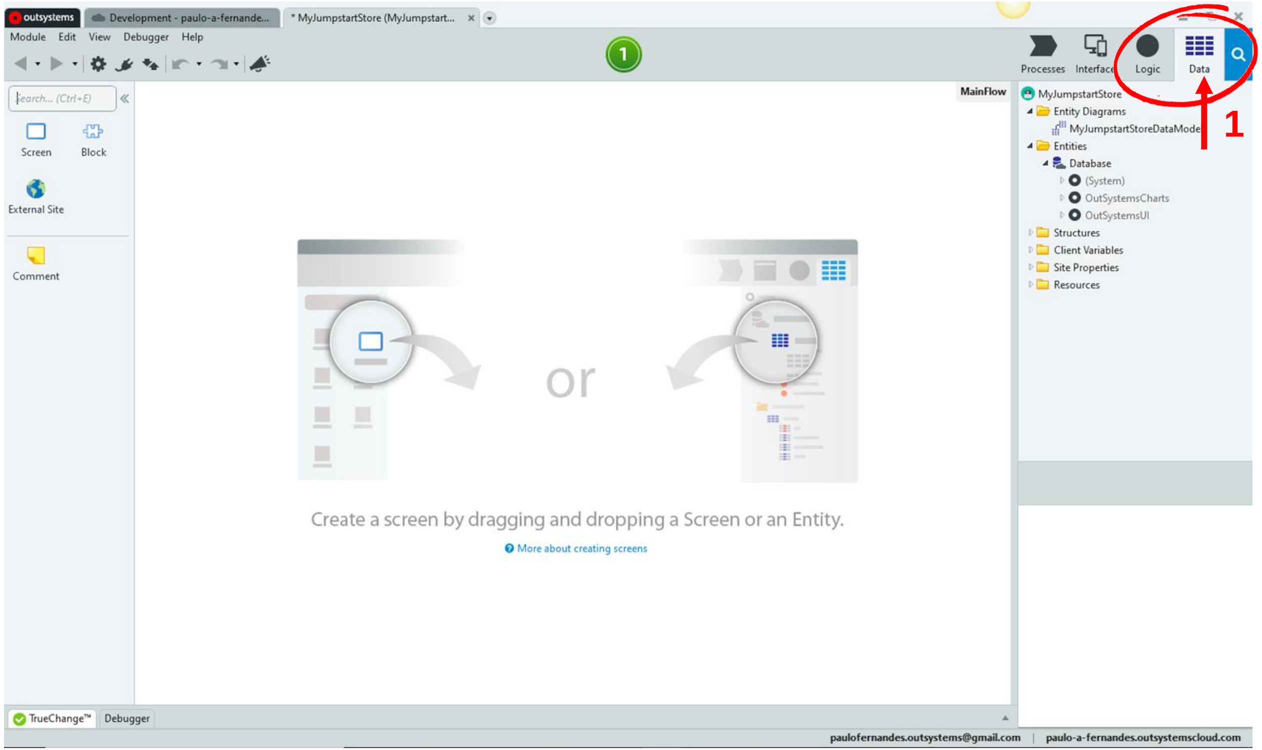
1. Leave the default Module Name
2. Click Create Module button



Let’s start by creating our first table in our data model. In Outsystems tables are known as **Entities** while columns are known as **attributes**.

After creating our first Entity we will populate it with data coming from an Excel file. This process is also known as **bootstrap.**

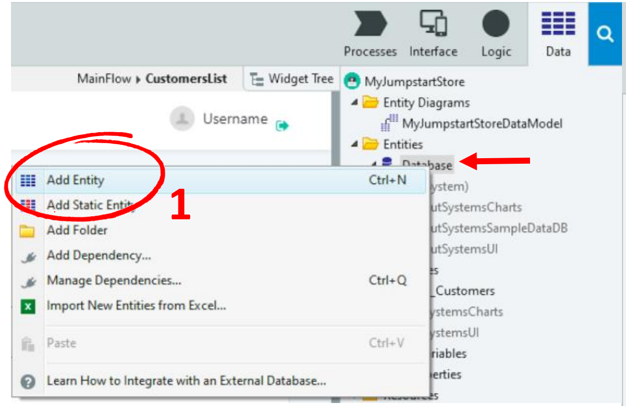
1. Access the **data modeling section** by clicking on the “Data” tab

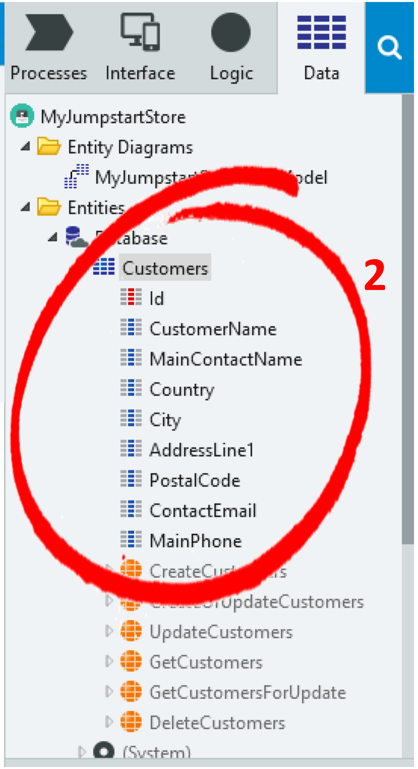


QUICK TIP Try to use keyboard shortcut: **Ctrl+4** to access Data tab

Create Customer Entity

1. Choose “Add Entity” to **create a new entity** to store information by right clicking on “Database” under Entities.
2. Create the “Customers” entity with the following attributes:





Name	Data Type	Length
CustomerName	Text	50
MainContactName	Text	50
Country	Text	50
City	Text	50
AddressLine1	Text	50
PostalCode	Text	50
ContactEmail	Email	
MainPhone	Phone	

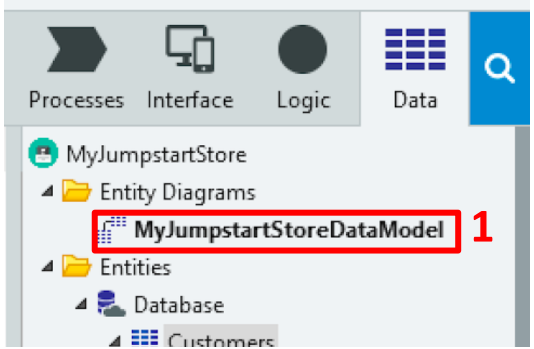
QUICK TIP

1. Try to use the keyboard shortcut: **Ctrl + N** to create Entity or Attributes
2. Use keyword for attributes name, will be automate attribute Data Type (example: *Phone for Phone Data Type)

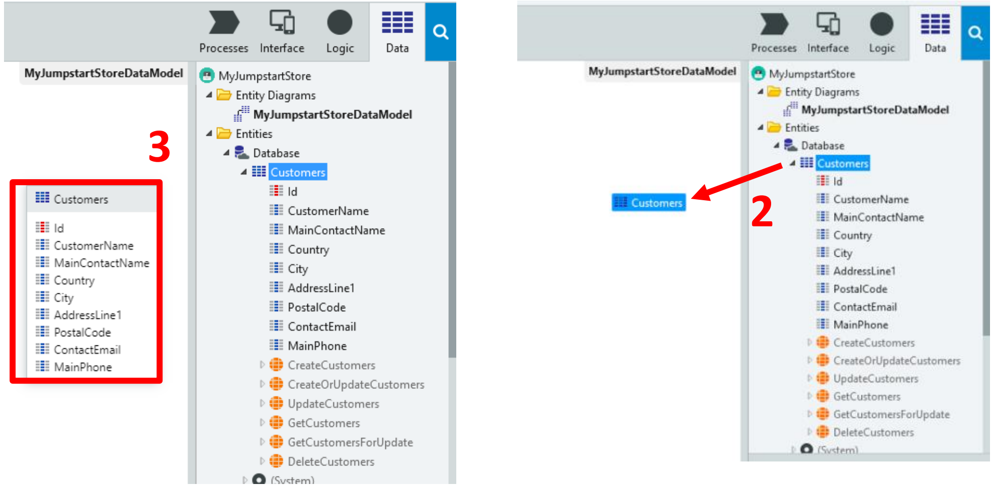
Add customer entity to Entity Diagrams

An **Entity Diagram** in OutSystems helps you visualize the relationship between your entities.

1. Go to the Data tab > Entity Diagrams and double click the “**MyJumpStartStoreDataModel**”
2. Drag the “Customers” entity to the diagram.



This diagram will show us how the entities are related to each other.

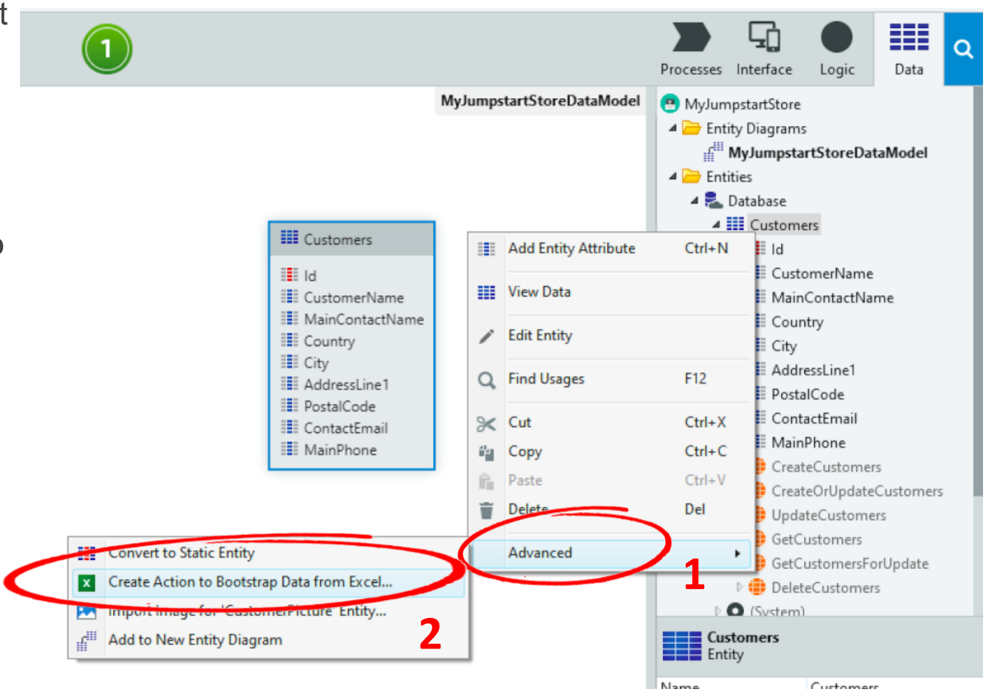


Section 1 > A > 8. Bootstrap Sample Data

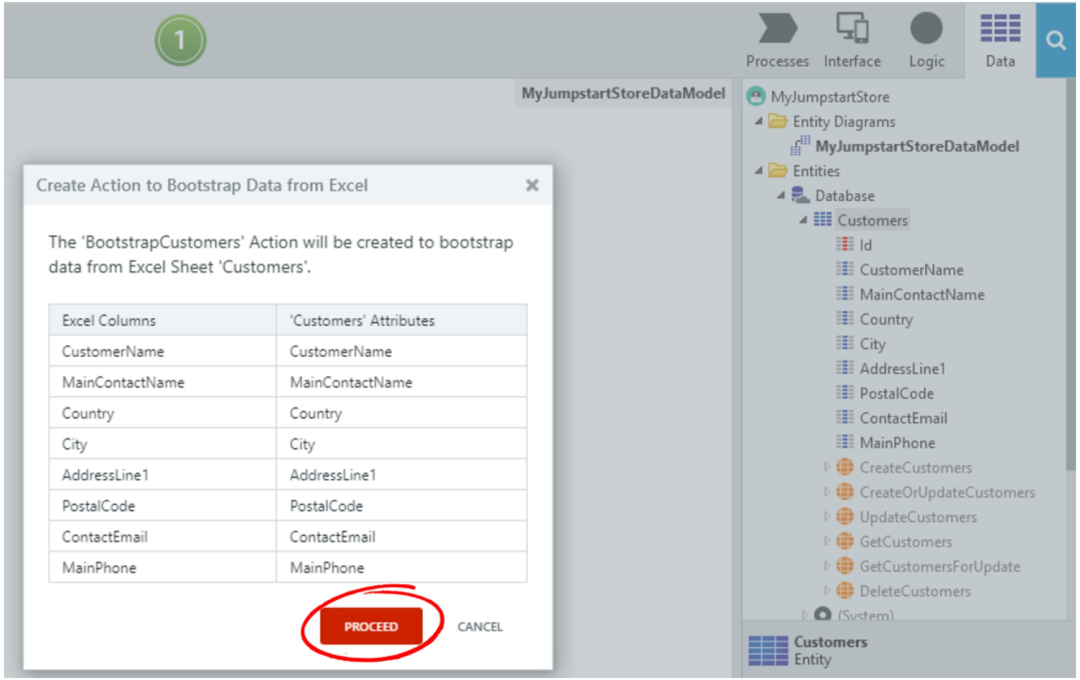
1. After the creation of the “Customers” entity we want to **bootstrap** some information to use as **sample data**.

   To do this right click on the “Customers” entity then go to “**Advanced**” and then click “**Create Action to Bootstrap Data from Excel...**”

2. Use the “Customers.xls” for the bootstrap.



1. If you have created the entity attributes correctly (**attribute name is case sensitive**), the bootstrap will match every column from the excel file. Click "**Proceed**" to continue.
2. If you find some mismatching, please correct the attribute names



Now that we have data, let‘s create a set of screens to display an overview of all customer data, a screen to edit existing data and to create new entries.

There are multiple ways we can quickly build screens in OutSystems:

1. From a **blank slate**, inserting each component with drag-and-drop
2. From one of our many **screen templates** - these come with sample data that can be modified for your own needs
3. Using a feature called “**scaffolding**” which automatically creates lists and forms based on your database entities.

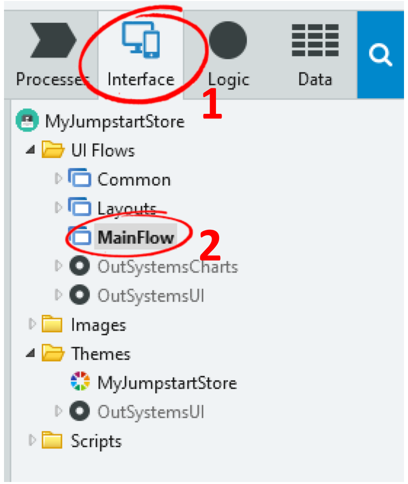
We’ll explore these different methods during the exercises.

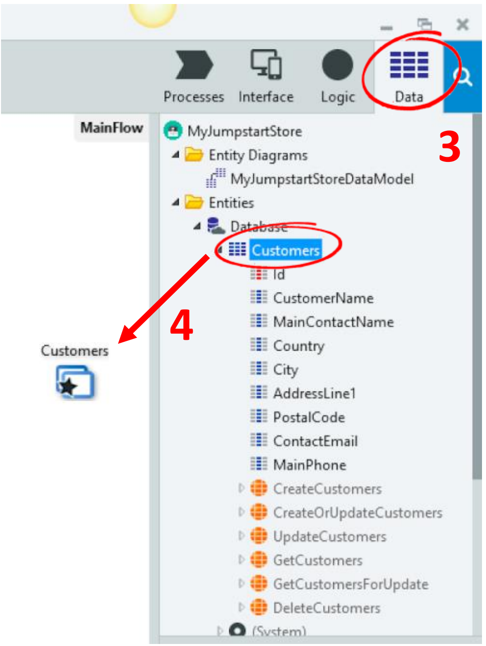
2. Create List Page

List page with paging and search

1. Go to the “**MainFlow**” folder, by going to “**Interface**” tab and double click on it.
2. Then go to “**Data**” tab and drag the “**Customers**” entity to the “**MainFlow**”.

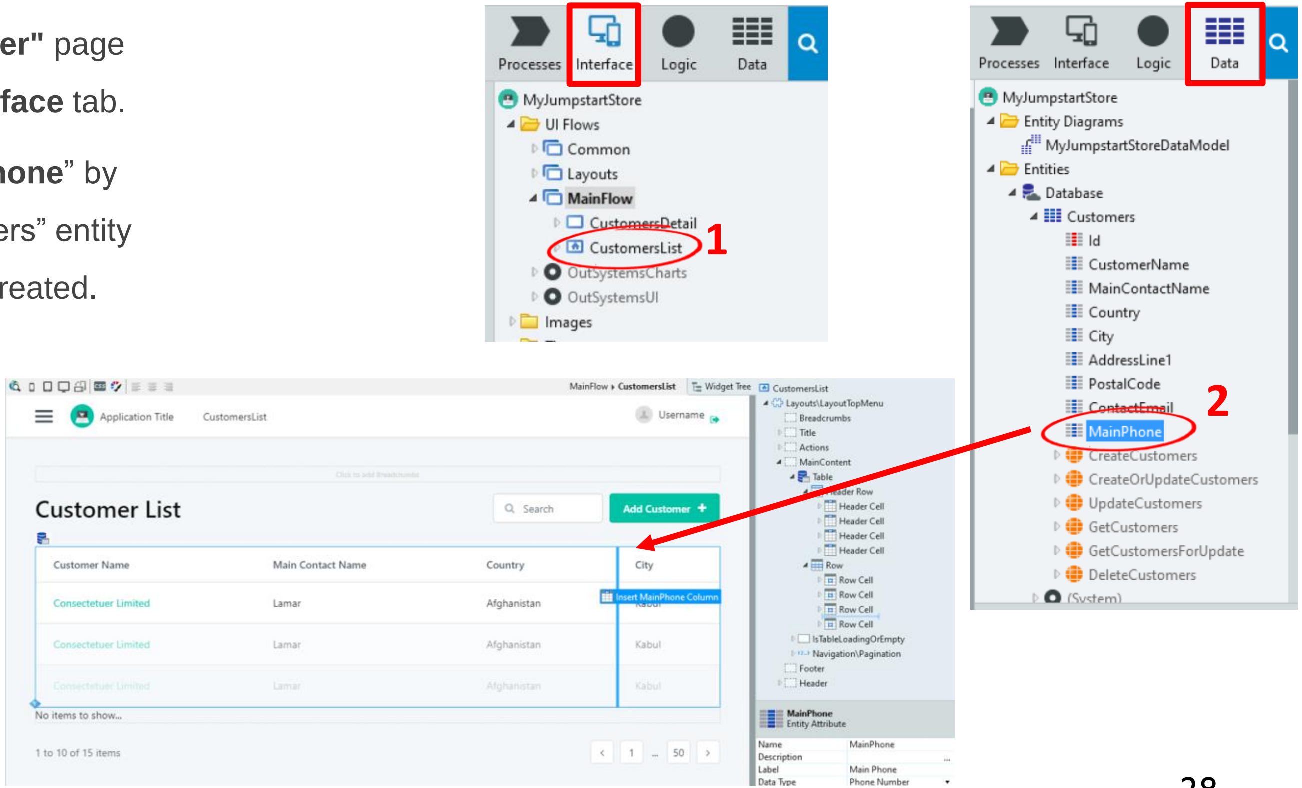
1. OutSystems has interface accelerators (called “**scaffolding**”) to automatically create pages based on Entities.
2. When you drag an entity to the screen, OutSystems auto-creates a **listing** and **detail** page





Add data to interface

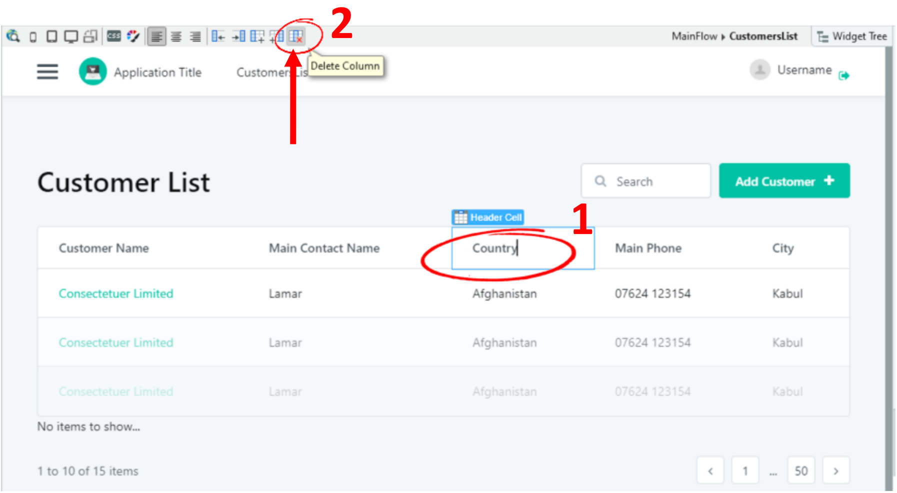
1. Double click on the **"Customer"** page under **Main Flow** In the **Interface** tab.
2. Add a **new column** "**Main Phone**" by dragging it from the "Customers" entity and dropping it on the table created.

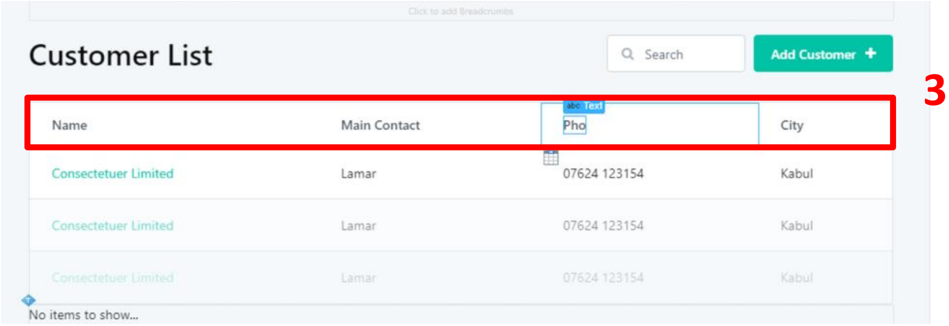


Section 1 B 4. Change Interface

Delete data from interface

1. Select column “Country” of the table
2. Use the “**Delete Column**” action to delete the “Country” column.
3. Change the labels
   - “Customer Name” to “Name”
   - “Main Contact Name” to “Main Contact”
   - “Main Phone” to “Phone”.

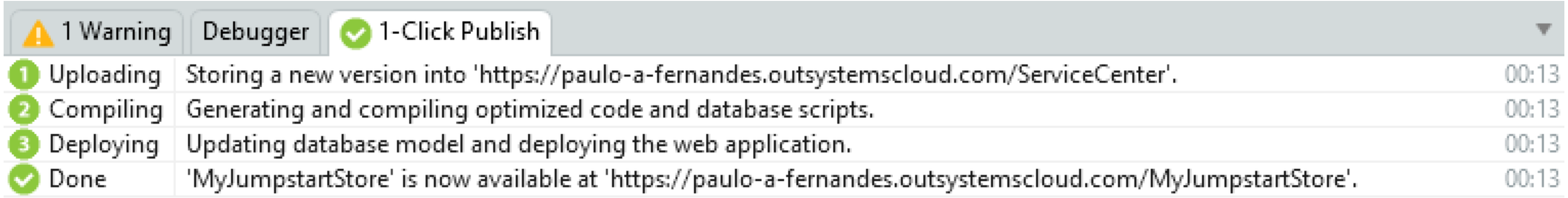




Section 1 › B › 5. Deploy and Test – 1-Click-Publish

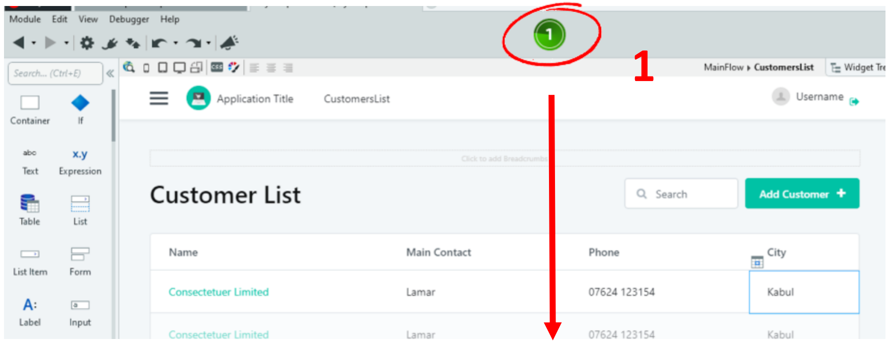
Once you are done developing, you need to deploy the application to the cloud. This is accomplished with only 1 click. This triggers the upload of the meta model to the cloud, compiling it to standard components, deploying the components & updating the database, so your application is ready to run.

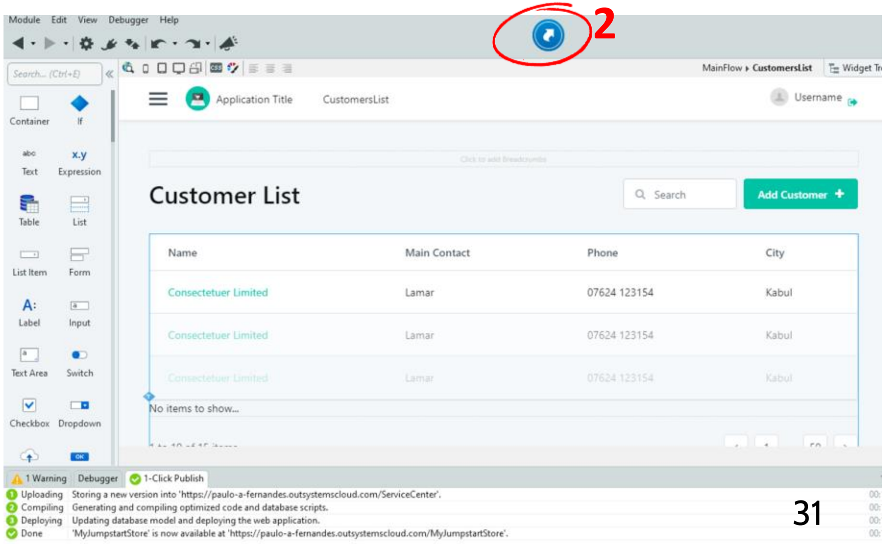
You can monitor this process in the 1-Click Publish tab (placed in the bottom of your development environment, and opens automatically when the process starts).



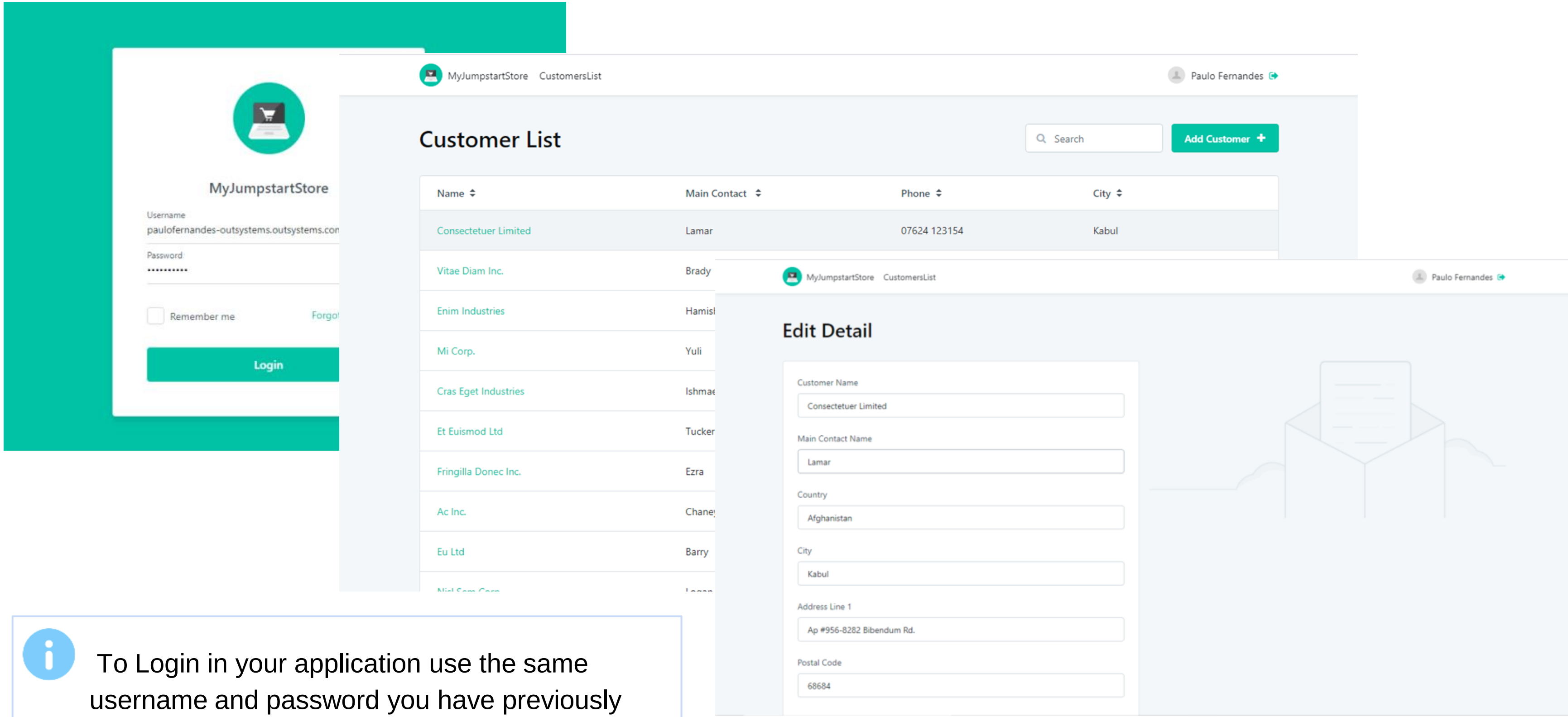
1. Click the "**1-Click publish**" to initiate the deployment process with a single click.

2. Click on the blue icon to see your web application running.
   *Once the deployment process is finished, the "**1-Click publish**" is updated to a blue icon.*

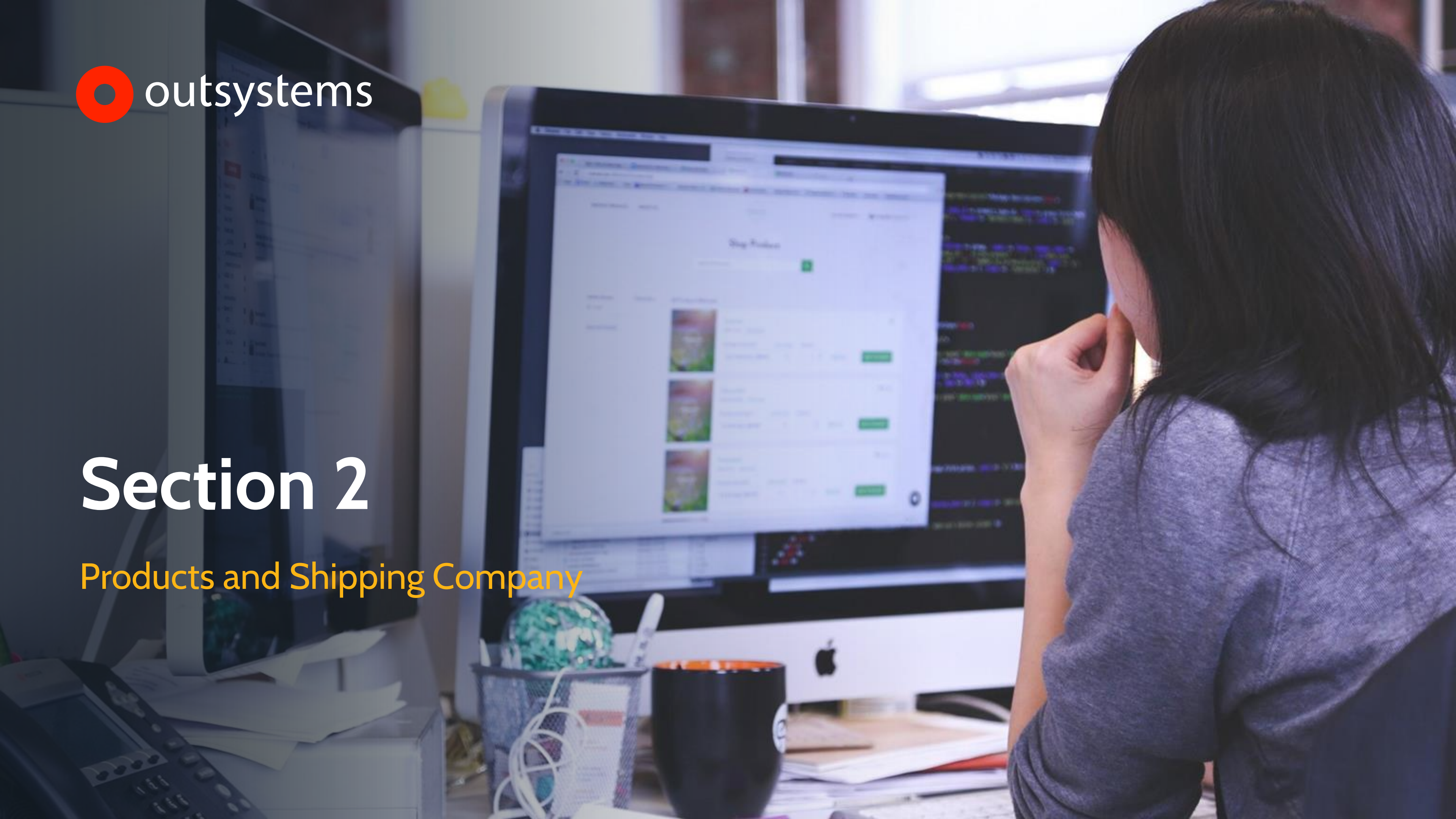




End of Section 1 - Your First Application!



To Login in your application use the same username and password you have previously defined for your OutSystems account.



outsystems

Section 2

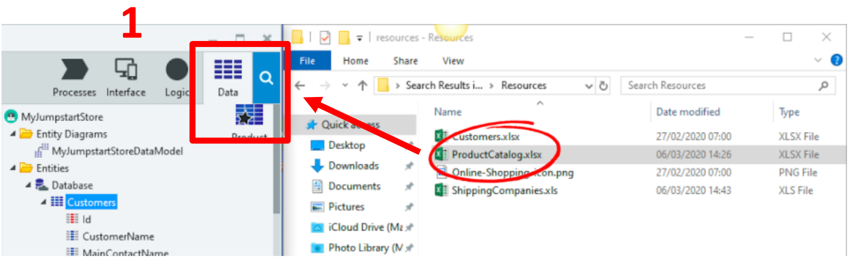
Products and Shipping Company

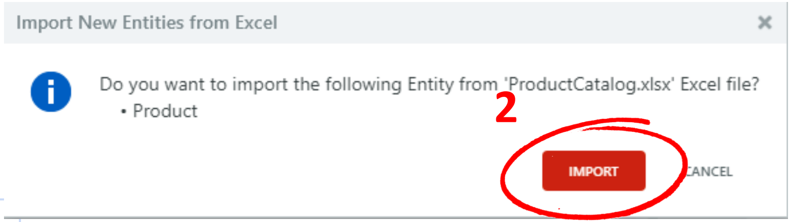
A. Use Excel to Bootstrap Data Model and Data

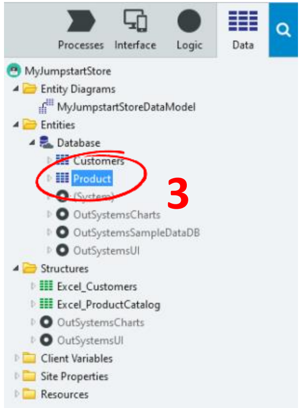
Another accelerator within the platform is to drag excel files onto the data section. This will automatically create the corresponding entities as well as actions to bootstrap the data itself from the excel file content.

We will use this together with the screen scaffolding patterns we already know to fast-track parts of our application.

1. Drag the "**ProductCatalog.xlsx**" to the Studio. Make sure you drop it on the top of the "Data" tab.
2. Click **Import** when prompted for confirmation.
3. You'll notice that the "Product" entity is automatically created.



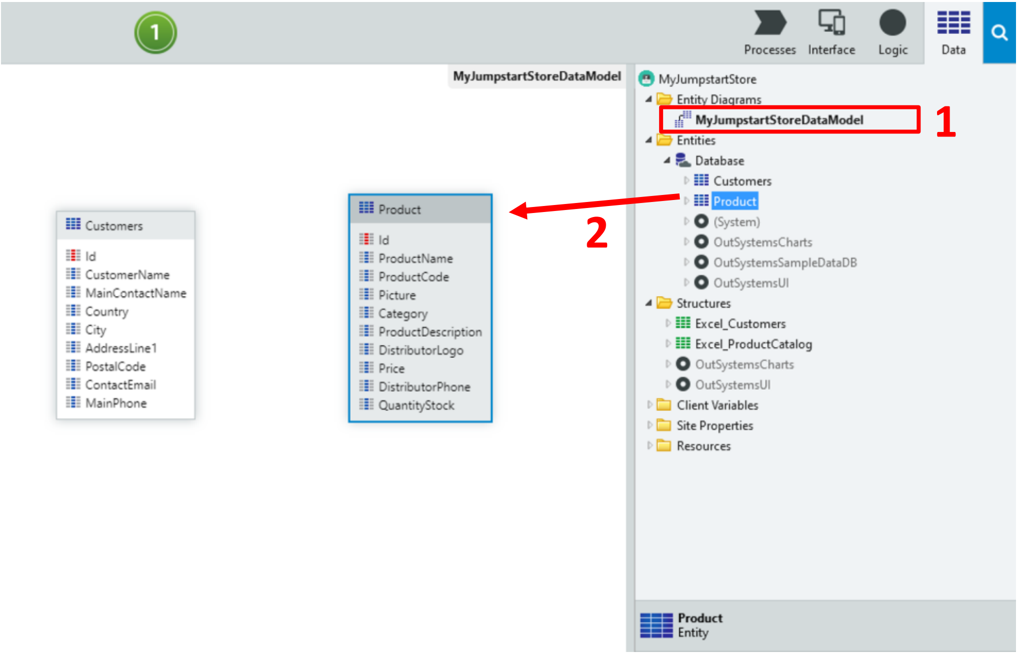




The OutSystems platform provides development **accelerators** to increase productivity on common, patterned and repeating development tasks (**scaffolding**).

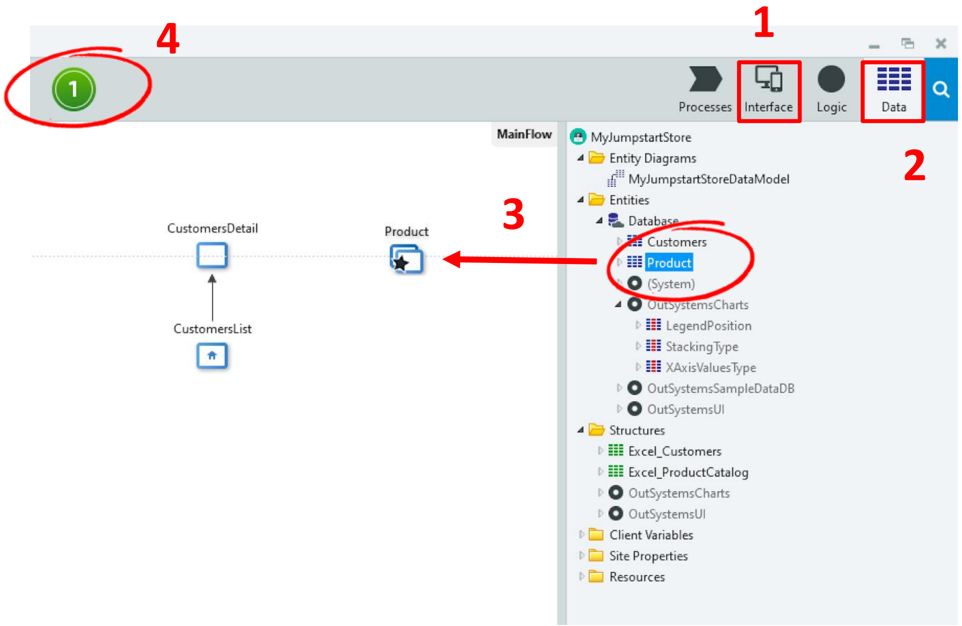
2. Add Product Entity to Diagram

1. Open the Entity Diagram "**MyJumpStartStoreDataModel**"
2. Drag the "**Product**" entity into the canvas.



As what you can see, our Customer entity and Product entity do not have a relationship yet.

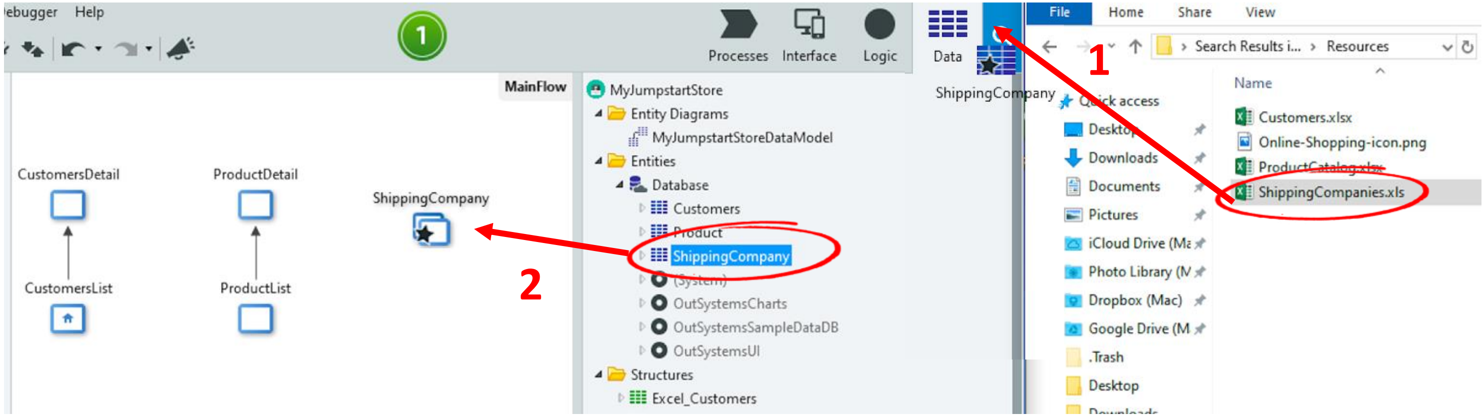
1. Open "**MainFlow**" on **Interface** menu
2. Open the **Data** tab
3. Drag the **"Product" entity** to the "**MainFlow**" canvas to create both pages (list and edit) and tune them with the proper information (i.e. *maybe you want to remove the picture URL from the product list, and display other attribute*)
4. **Deploy** the application



QUICK TIP

To speed-up, try to use keyboard shortcut **Ctrl+F**, and type "**MainFlow**" to directly open the object.

1. Repeat the previous steps to create a “**ShippingCompany**” entity. Use the “**ShippingCompanies.xls**” excel file.
2. Add the corresponding pages by dragging the ShippingCompany entity to the “MainFlow” (list screen with search and paging, and detail screen).



For the orders we want to manually model the data, and then use a scaffolding pattern again to create the basic pages.

You will do that in the **Data** section of Service Studio.

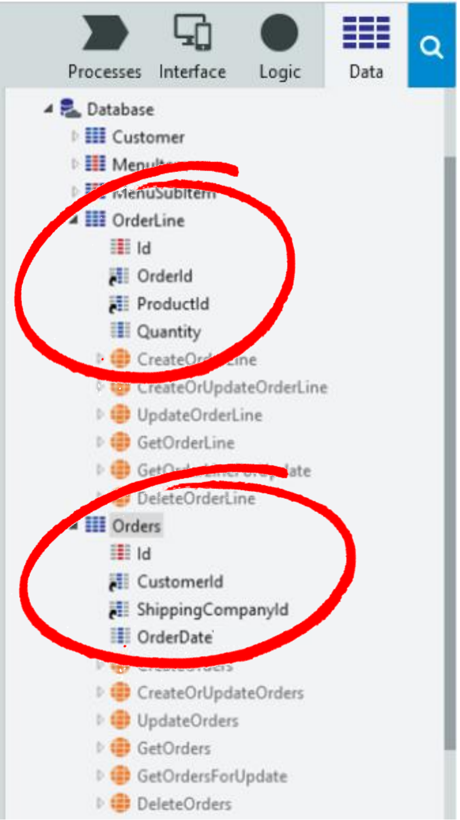
Section 2 › B › 1. Create Order and Order lines

1. Create the entity “Orders” with the following additional attributes:

Name	Data Type
CustomerId	Customer Identifier
ShippingCompanyId	ShippingCompany Identifier
OrderDate	Date

1. Create the entity “OrderLine” with the following additional attributes:

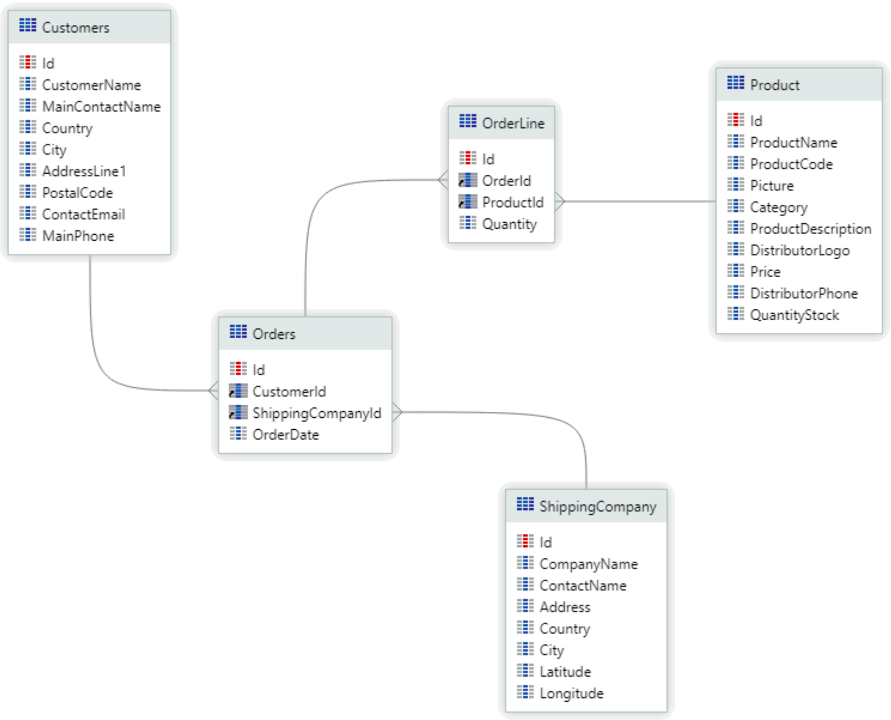
Name	Data Type
OrderId	Order Identifier
ProductId	Product Identifier
Quantity	Integer



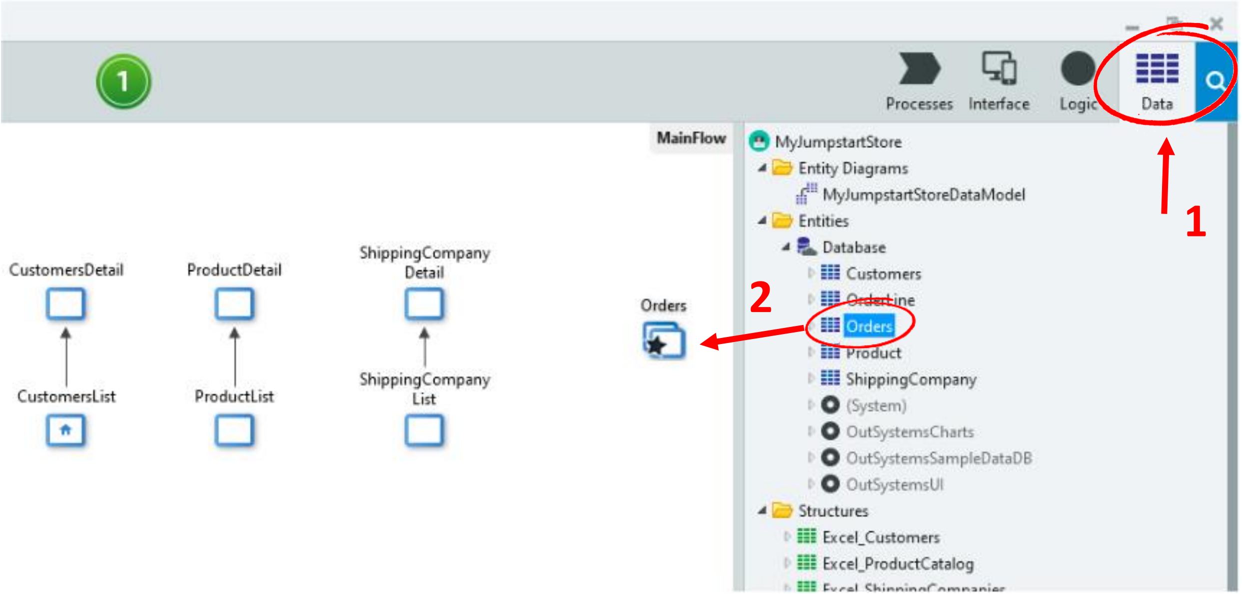
Note that by creating attributes with an existing entity suffix, the data type is automatically mapped to an entity identifier (e.g. the “CustomerId” attribute is automatically defined as an identifier of the Customer entity).
Please make sure that your data model, including the attributes, are correctly created.

1. Drag the “Orders” entity to the "MyJumpStartStoreDataModel" **diagram.**
2. Drag the “OrderLine” entity to the "MyJumpStartStoreDataModel" **diagram.**
3. Drag the “ShippingCompany” entity to the "MyJumpStartStoreDataModel" **diagram.**

The “MyJumpStartStoreDataModel” diagram should be similar to the one on the left.

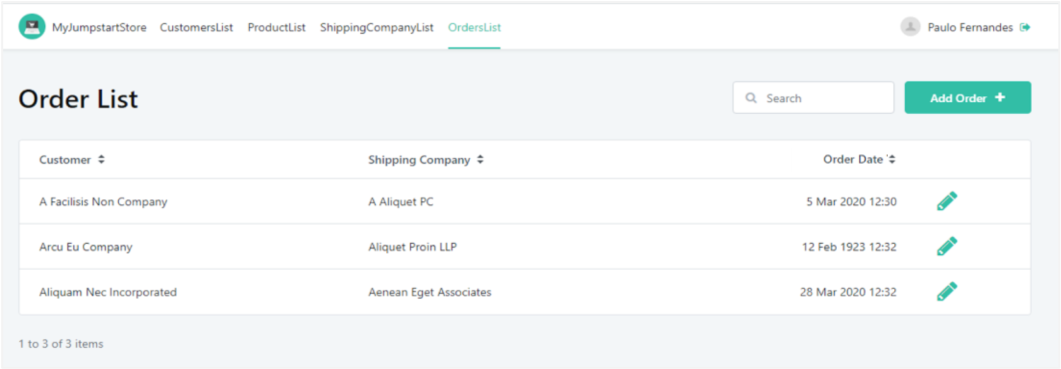


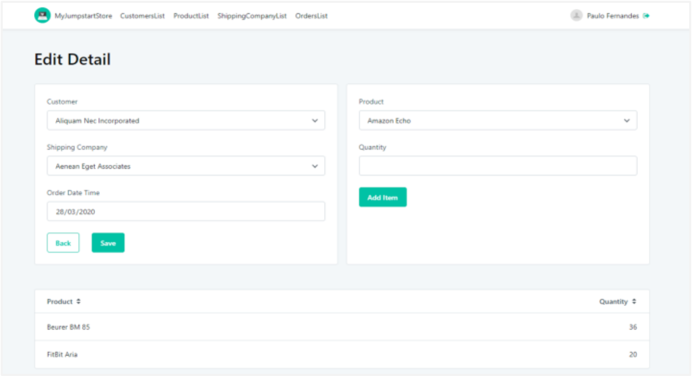
1. Drag the “Orders” entity to the “MainFlow” to create both pages (list and edit)



The accelerators are a great way to get a lot of functionality very fast. For simple applications that might be exactly what you need. More often than not you will want to adapt the generated pages to exactly your needs. You can easily do this, again in a visual fashion.

In the next steps you will take the new order pages and change them to cover some specific needs. We will adapt these pages to enable users to add order lines to existing orders. In the end of this section your Order Listing and Detail pages should look like the screens illustrated below.

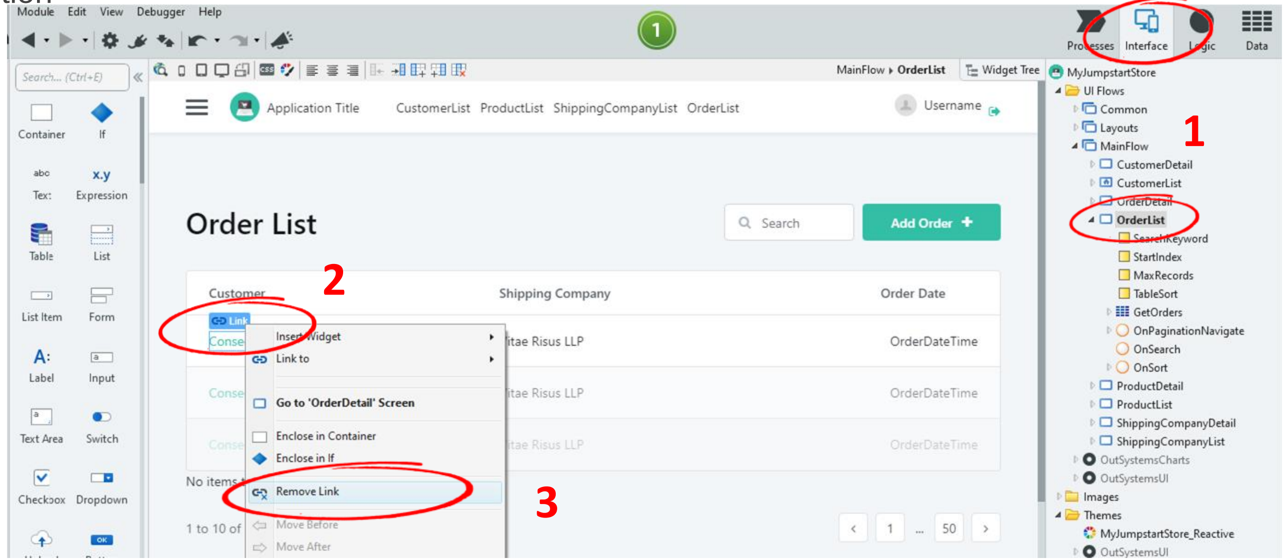




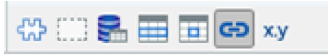
1. Go to the **Interface tab** and open the **“Orders”** page

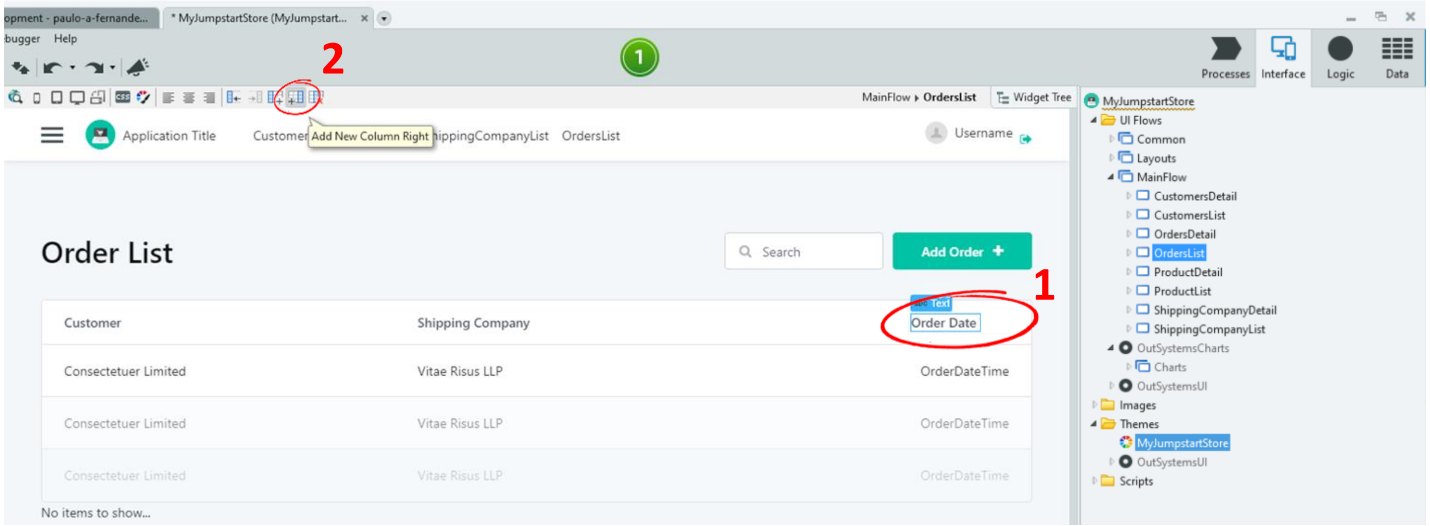
2. Select the **first element** of the **"Customer"** column, and right-click the enclosing **Link** widget

3. Click on the **“Remove Link”** option



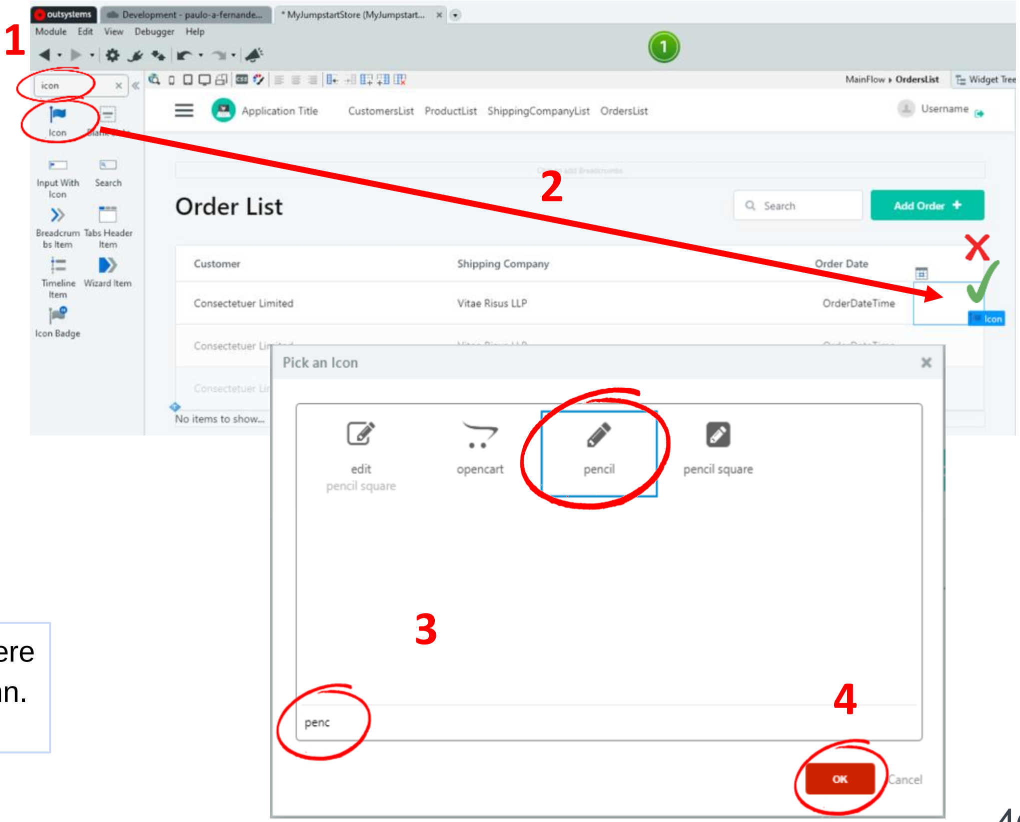
To help you select an enclosing element, you can use the **breadcrumbs** on the bottom of the main view:





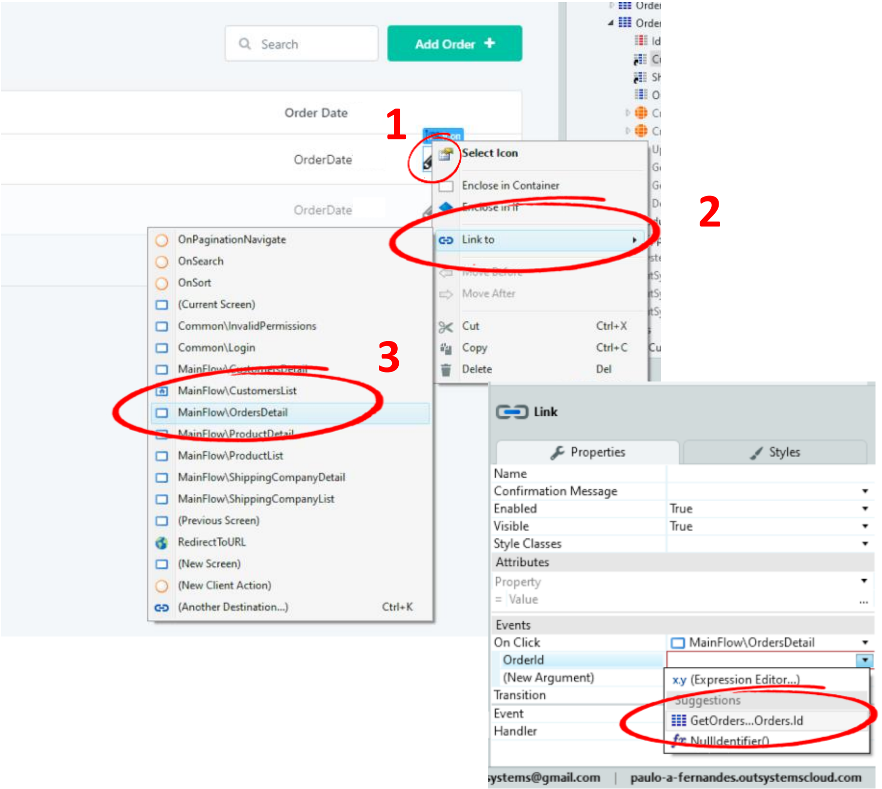
1. Still in the OrdersList screen, click on the header of the **"Order Date"** table column
2. Click on the **“Add New Column Right”** button to **add a new column** to the right of the Order Date

1. Go to the **toolbox search bar** (top left) and search for **"Icon"**.
2. Then, **drag the Icon** widget to the last column of the Order List table. Make sure you drop it on the second row (under the header cell).
3. A new window will show up. **Search** for and choose the "**pencil" icon**
4. Click **"OK"**

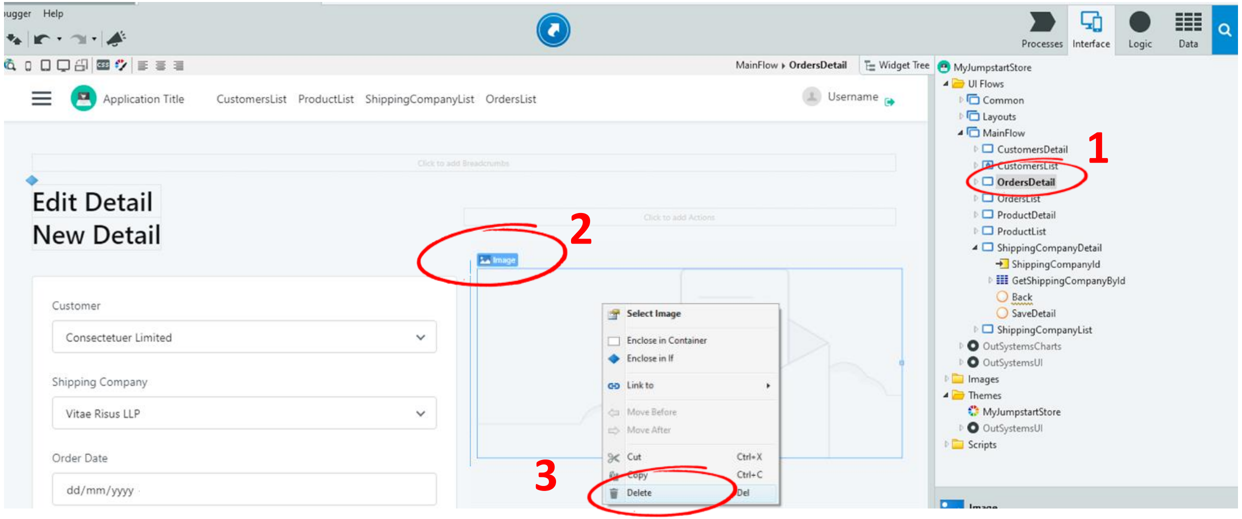


The first row of the table is the Header row where you can enter the header name for each column.

1. In the Orders List table, **right click** the new Pencil icon
2. Choose "**Link to**" > "**MainFlow/OrderDetail**"
3. In the **Properties** tab set the **OrderId** value to "**GetOrders.List.Current.Orders.Id**".



You have just created a link to the "OrderDetail" page. Users can now click on the Pencil icon to open the Order Detail page of a specific order.



We will now adapt the Order Detail page to enable users to add new Order Lines to an Order. Let's start this process by deleting the Image that is displayed by default on the screen. To do that:

1. Open the **"OrdersDetail"** page
2. **Right click** on the Image widget and select the **Delete** option.

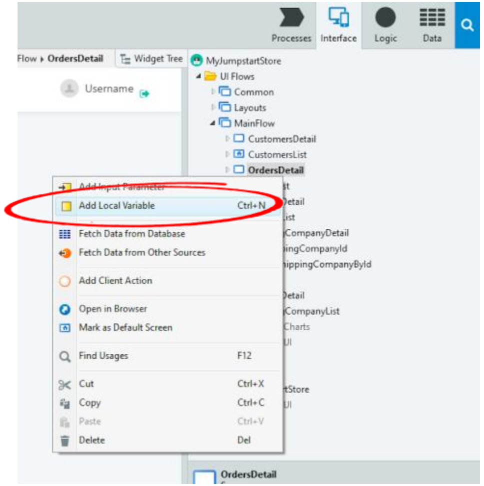
6. Add local variable

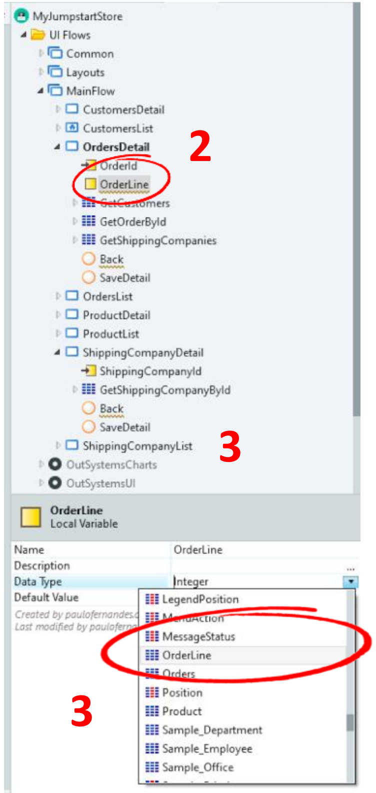
1. **Right click** the OrderDetail page and choose the **Add Local Variable** option
2. **Rename** the local variable to “OrderLine”
3. In the Properties tab set the **Data Type to OrderLine**.

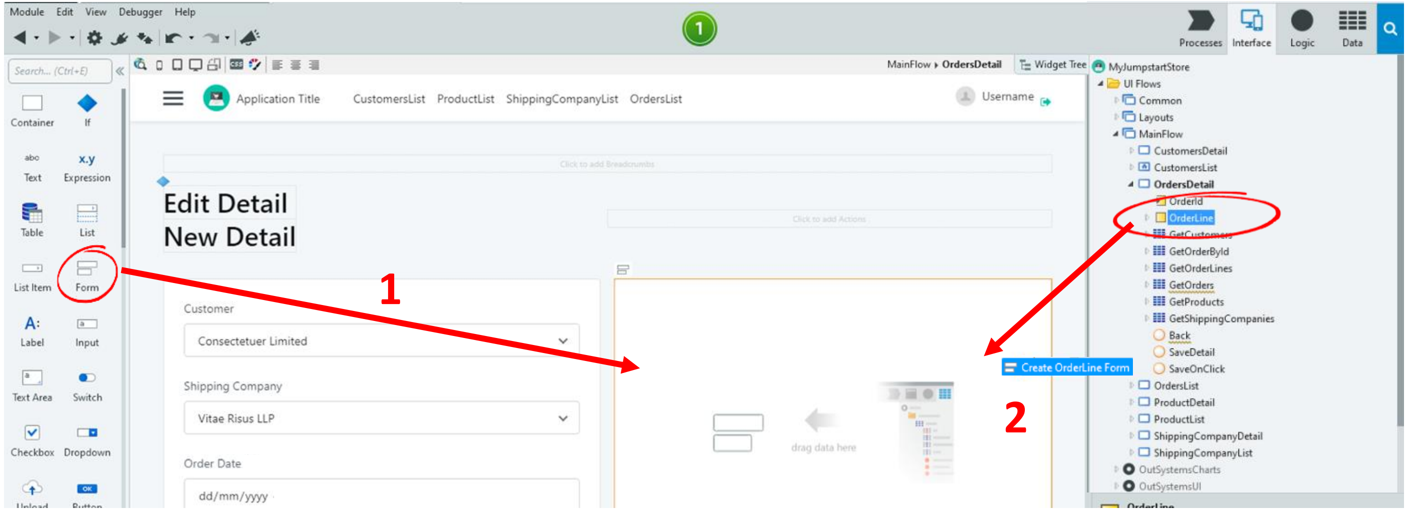
> QUICK TIP
> You can also use the keyboard shortcut: **Ctrl+N** to create a local variable

> By setting the data type to OrderLine the local variable will inherit the OrderLine entity structure.

1





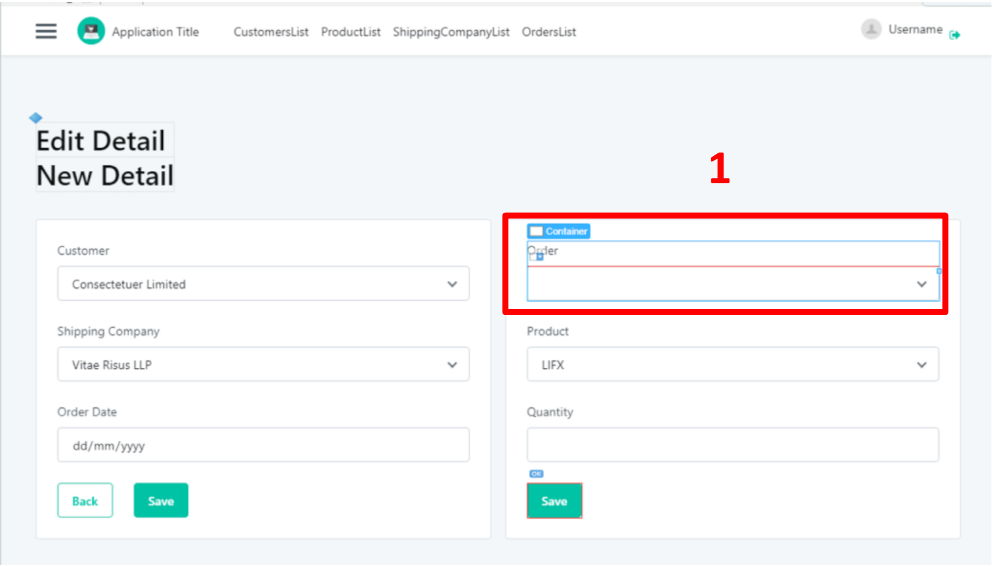


1. **Drag the Form widget** from the Toolbox pane in the left to the column space in the right. This will create a blank form.

2. Then **drag the OrderLine local variable** to the **Form widget**. This will automatically add to the form the “OrderLine” local variable fields.

The Order dropdown should not be displayed in this page since we are already in the context of a specific order.

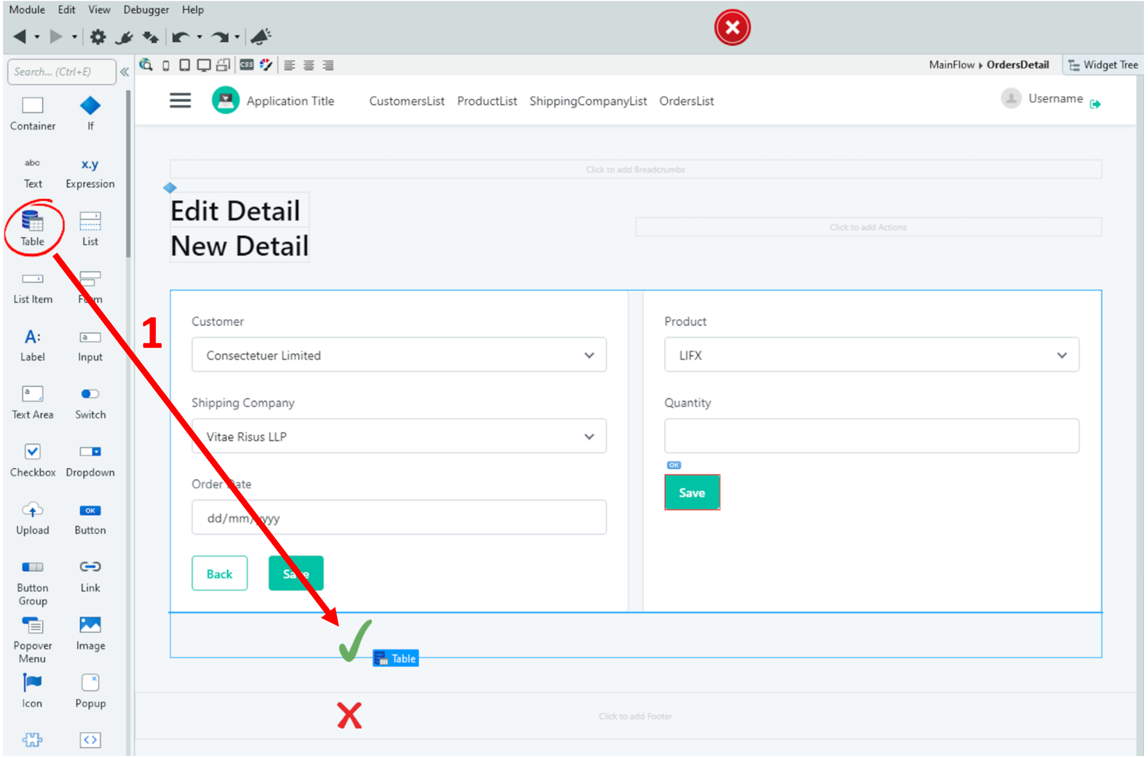
1. Delete the Order Label and Dropdown (delete the container wrapping both elements).



Later in the exercise we will ensure that the Order Id is mapped to the OrderLine local variable.

1. **Drag a Table component** below the two forms.

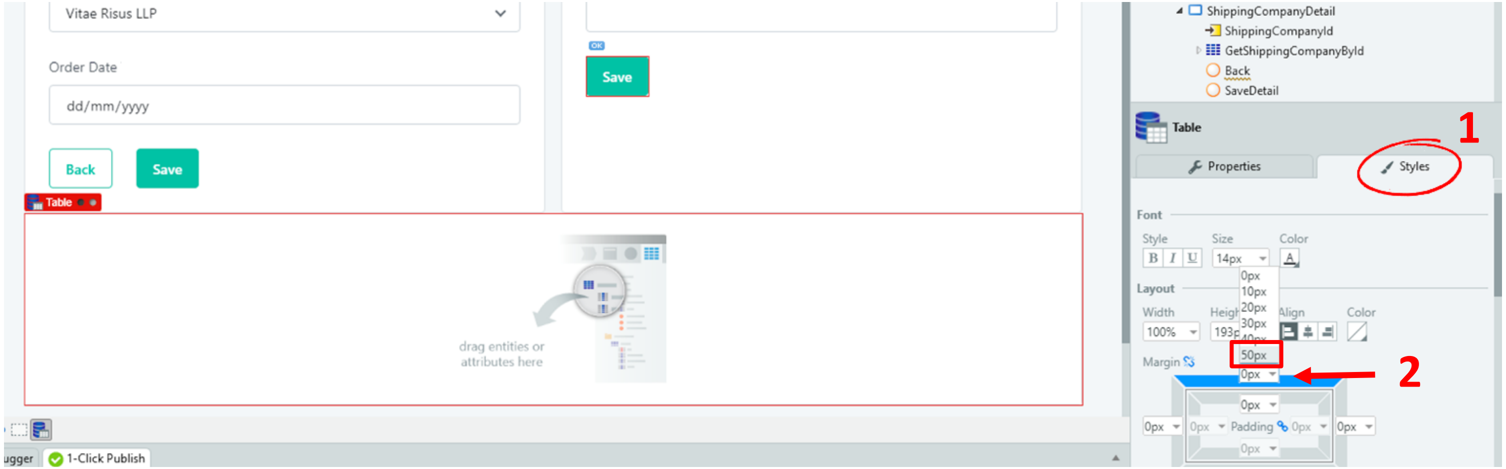
   Make sure you are not dragging it to the Footer of the page.



Add space between the table and forms

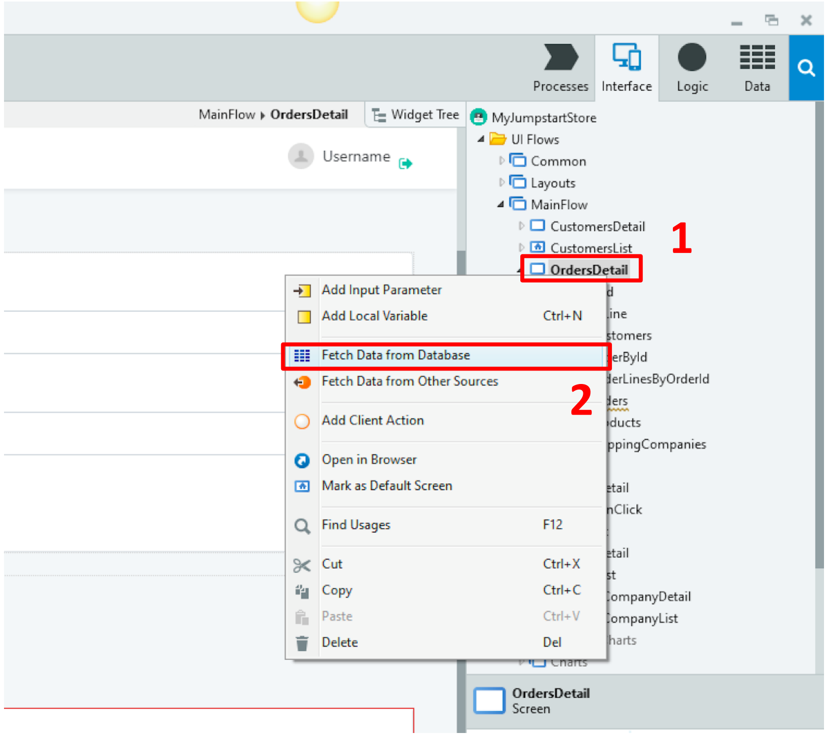
You now have table below the forms. But there's no space in between these elements. Let's add that space:

1. With the **table widget** still selected, click the **Styles tab.**
2. Then change the style attributes to have a **Margin-top** of **50px**.



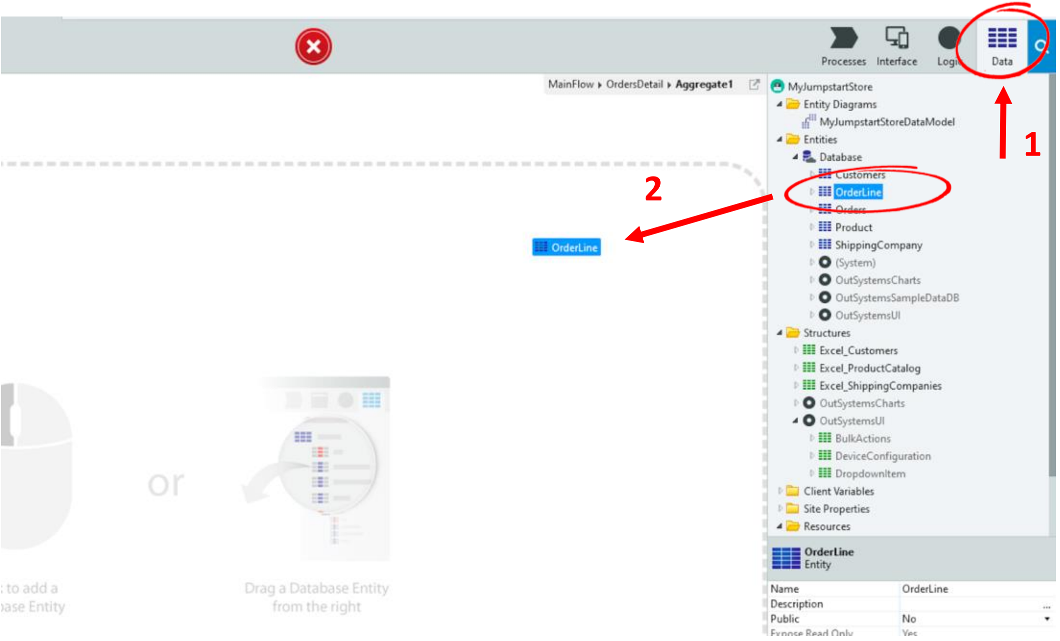
Now that we have added the necessary spacing to the table, lets populate it with order line information from the "OrderLine" entity. For that:

1. **Right click** in the "OrderDetail" page
2. Then click on **"Fetch Data from Database"**



Section 2 › C › 12. Fetch order line information from DB

1. Navigate to the **“Data” section**
2. **Drag the “OrderLine”** entity to the new **aggregate** to fetch information from the Order Line table



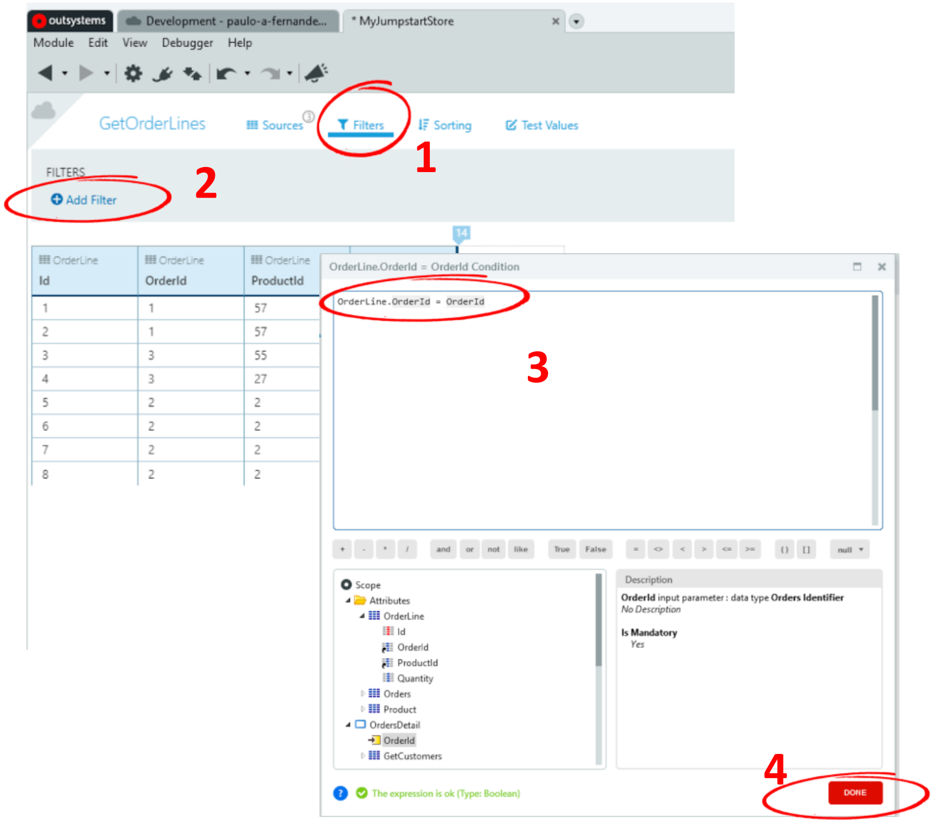
In web applications you can use an Aggregate to fetch data from the server database.

An aggregate allows you to fetch data from the DB using an optimized query.

1. Click **Filters**
2. Click **Add filter**
3. Enter the following query:

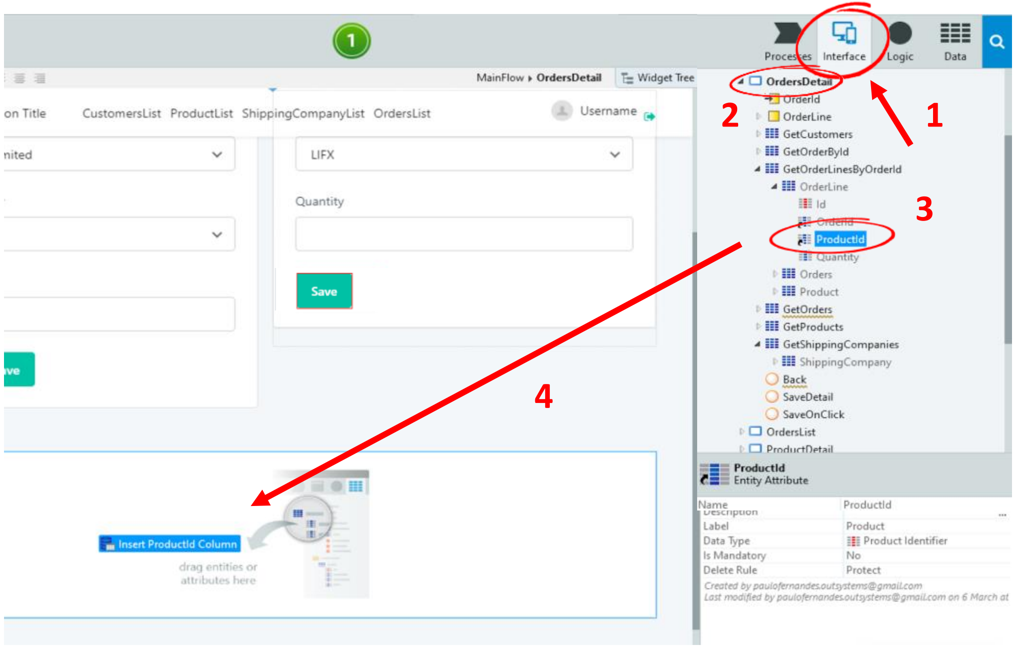
   Orderline.OrderId = OrderId

4. Click **"Done"**



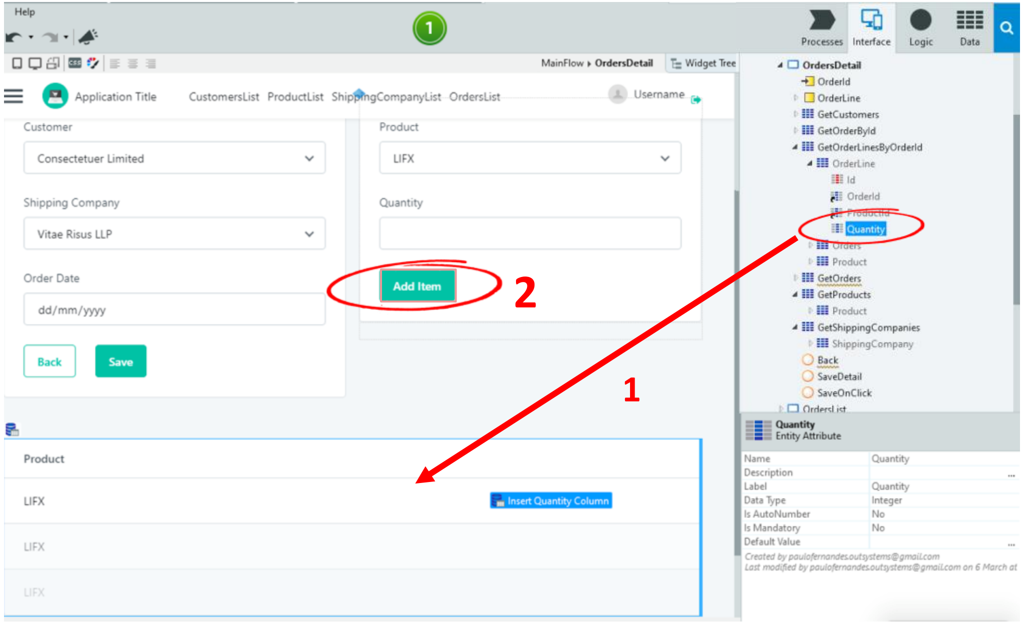
When designing queries, it is common to add conditions to filter results and get exactly what you want from the database. In OutSystems, this is done by adding Filters to Aggregates, as described above.

1. Navigate to the **“Interface” section**
2. Double click the **“OrderDetail”** screen to open the Order Detail screen
3. Expand the “**OrderDetail”** screen, then expand the **“GetOrderLinesByOrderId”** aggregate and then the **“OrderLine”**
4. **Drag** the **“ProductId”** attribute to the table in the screen



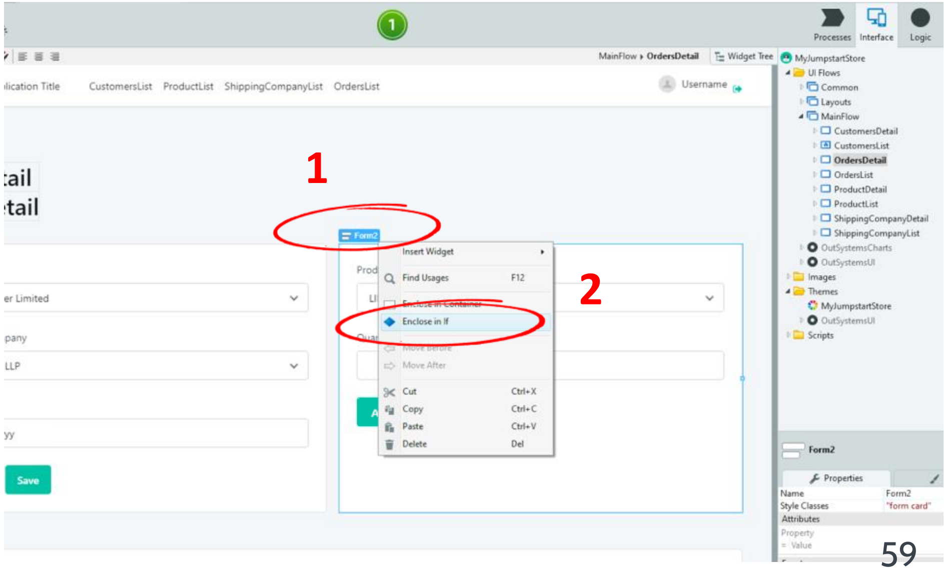
1. Drag the **“Quantity”** attribute to the right side of the **“OrderLines” table**.

2. Then, rename the **“Save”** button to **“Add Item”**.



The Order Detail page is used to edit but also to create orders. We only want users to be able to add items (new order lines) to an order once this order is created. We will use the **If widget** to make sure the Form (used to add items) and the Order Lines table are only displayed when the user is editing an existing Order.

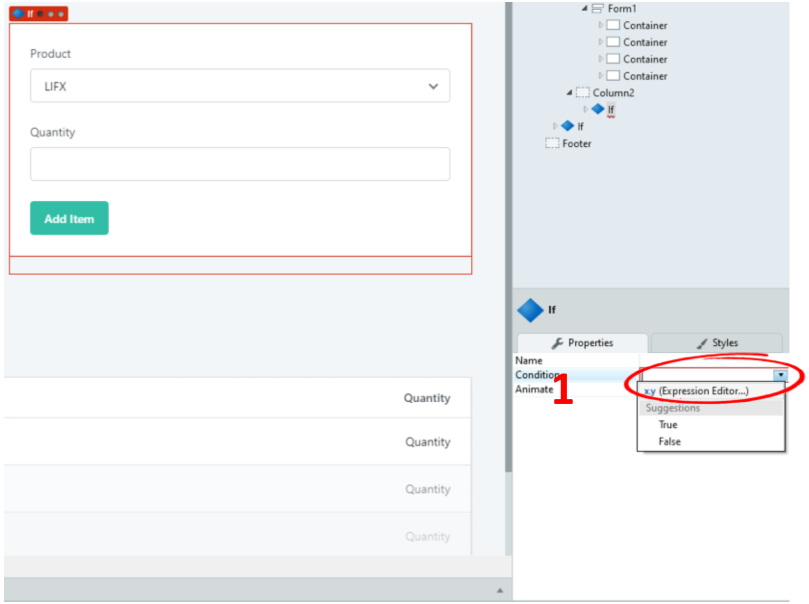
1. **Right click** on the form in the right (Form2) > **"Enclose in If"**

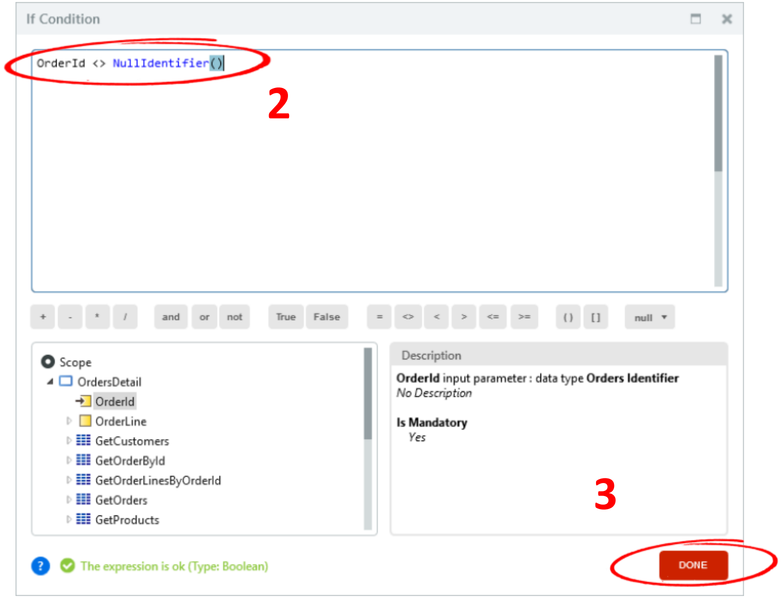


The If widget allows you to control the content that is displayed in the screen based on a condition.

You specify the content of the True and False branches at design time. At runtime the condition is evaluated and only one of the alternatives is displayed into the screen.

Section 2 › C › 17. Add Condition to Display Form





1. In the **If widget** Properties tab, open the condition and click on **“(Expression Editor...)”.**

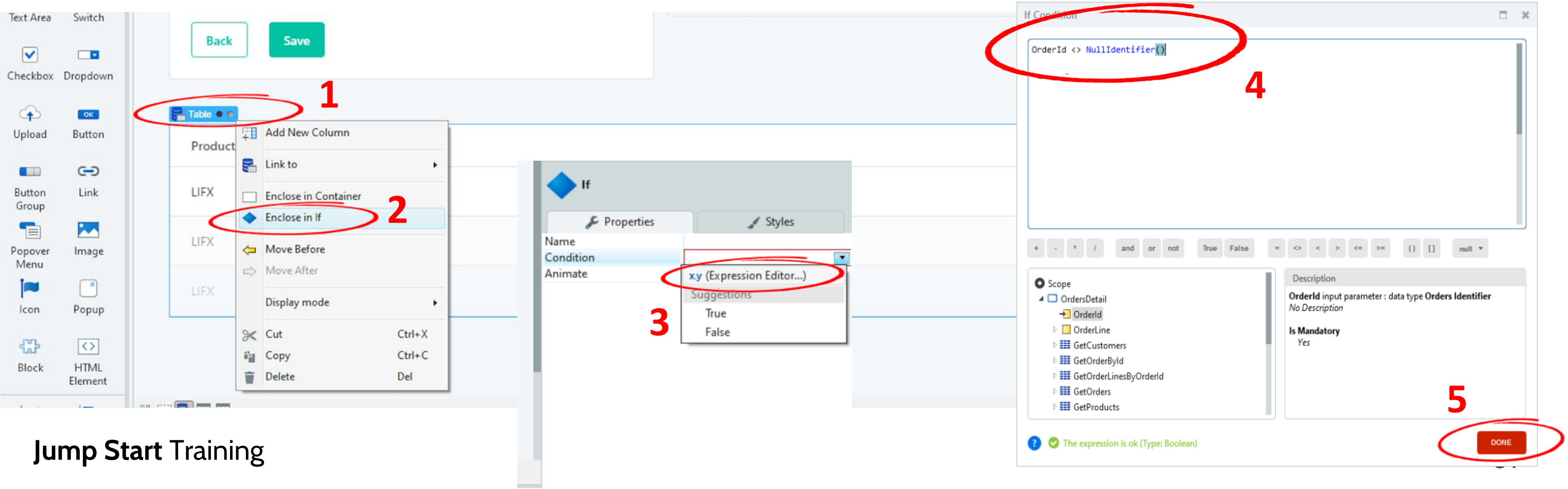
2. In the expression editor input the expression “*OrderId <> NullIdentifier()*” and press **“Done”**.

QUICK TIP

You can also double-click on “Condition” to quickly open the Expression Editor

Repeat the same steps for the OrderLines Table:

1. Right click on the Order Line table and then **“Enclose in If”**
2. In the Properties tab, open the condition and click on **“(Expression Editor...)”**
3. In the expression editor input the expression “*OrderId <> NullIdentifier()*” and press **“Done”**.



Section 2 D. Create Logic to add Order Lines to an Order

Now that you have finished adapting the look and feel of the Order List and Detail pages lets work on the logic to add new items (order lines) to an existing Order.

In the following steps you will create the logic of the "Add Item" button.

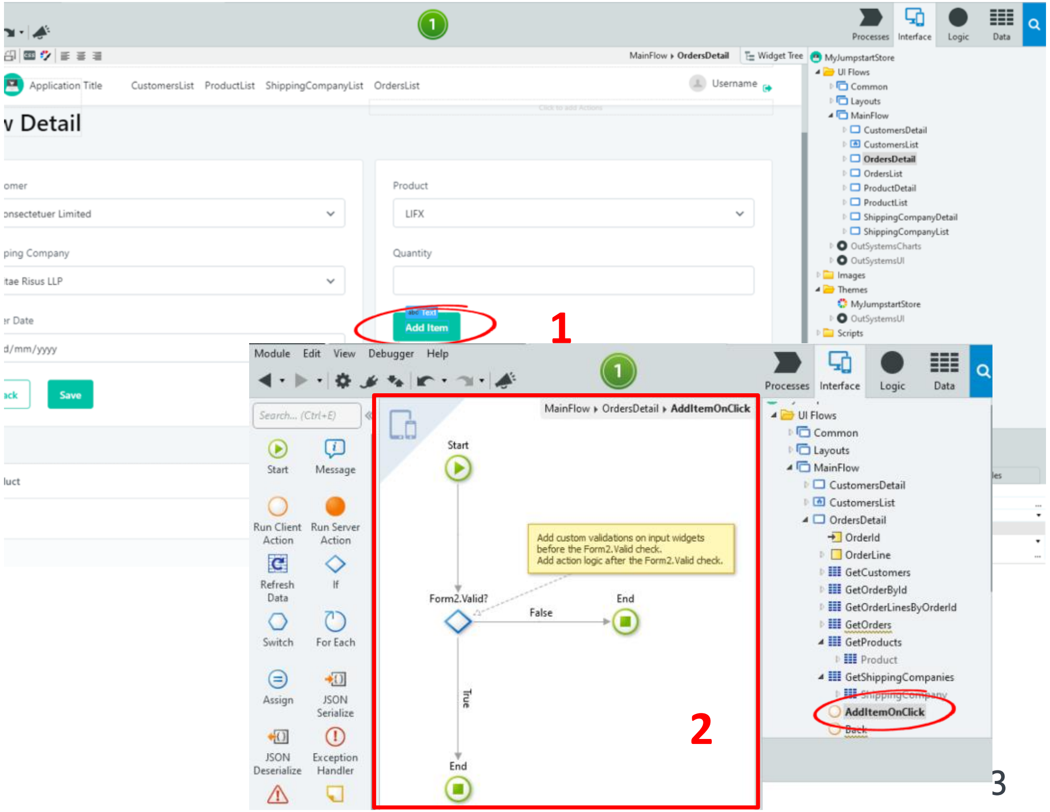
In the end of this section your action flow should look like the one in the image on the right.

Section 2 | D | 1. Add order id on the table

Let’s create the “Add Item” Button logic

1. **Double click** the “Add Item” button to create a **New Client Action.**
   Make sure to double-click the Button widget, and not the Text widget inside it

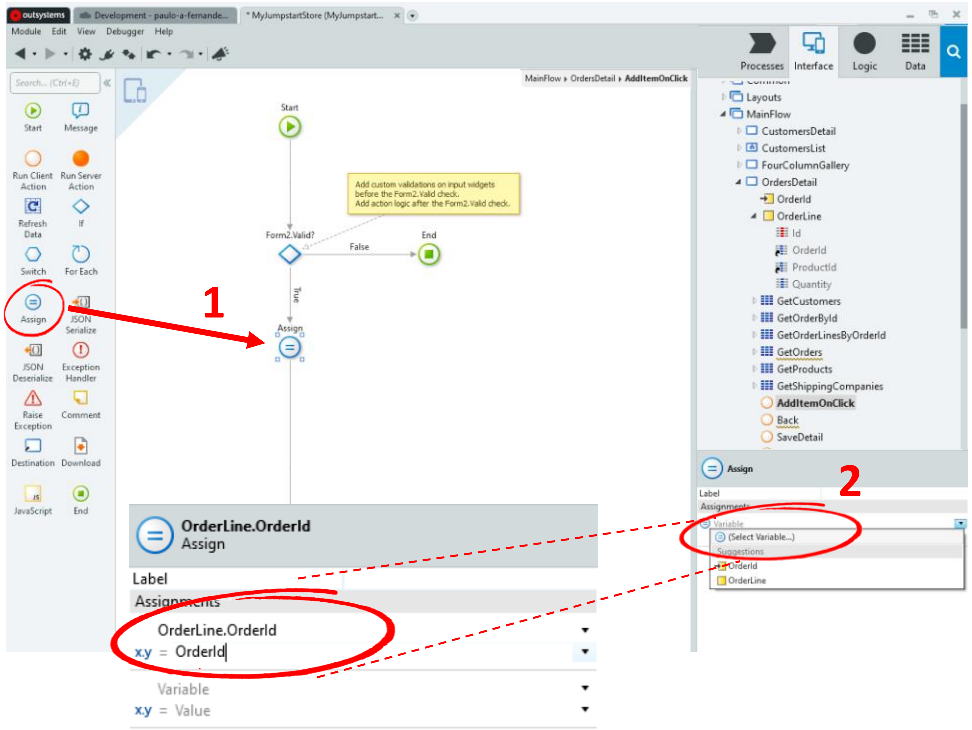
2. Notice that a new **“AddItemOnClick” Client Action** is created and automatically opened so that we can start working on its logic.



When developing apps you create both logic that runs on the **server**, and logic that runs on the **client** device - just like on mobile apps.

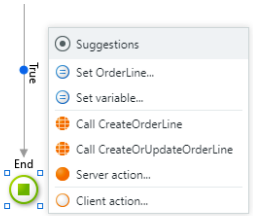
1. Drag an **"Assign"** node to the "AddItemOnClick" action flow.

2. In the properties of the assign node, set the Variable of the assign note to **OrderLine.OrderId** and the Value to **OrderId**



QUICK TIP

Click on the logic flow to see what OutSystems **AI engine** suggests you could do next, based on similar patterns to yours.

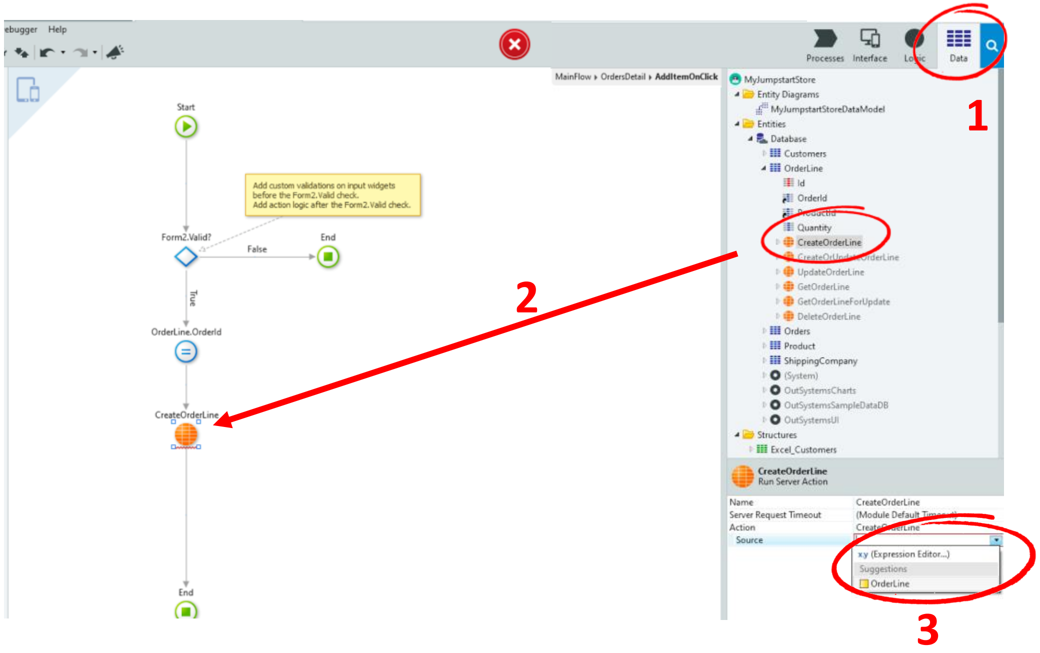


Section 2 D 3. Add Server Action to Create Order Line

1. Navigate to the **“Data” section**

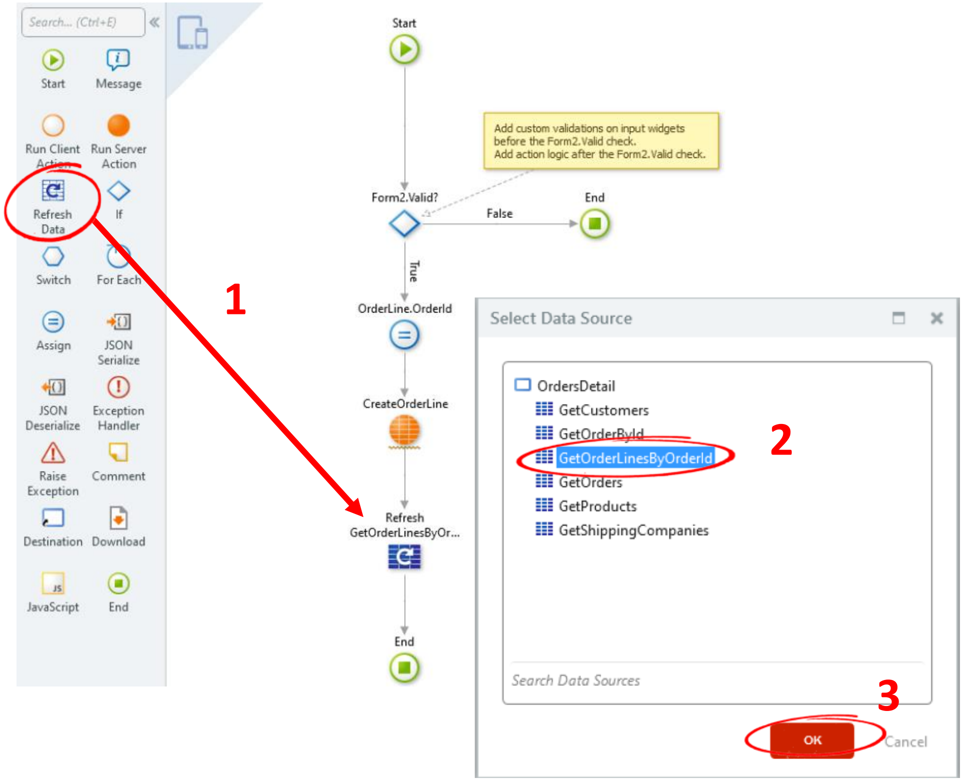
2. Drag the **“CreateOrderLine” Server Action** from the **“OrderLine” entity** below the assign activity that you have just created.

3. Set the source value to the “OrderLine” attribute.



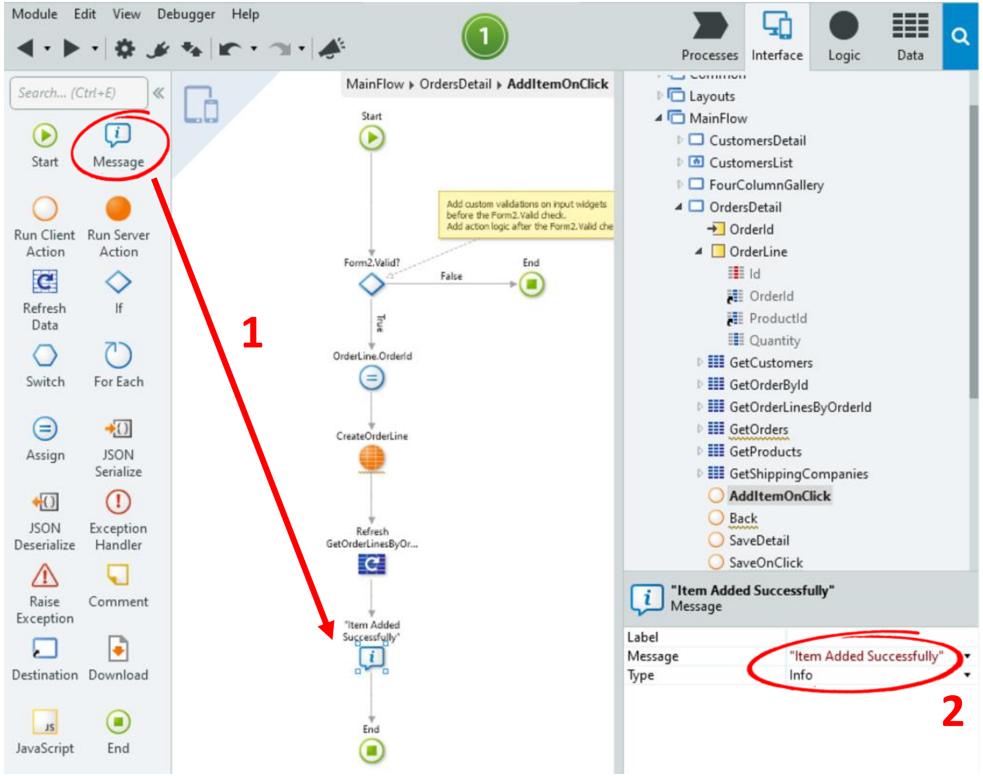
After creating the order line in the database, we need to refresh the OrderLines table in the screen to fetch the latest records from the database.

1. Use the **Refresh Data** widget.
2. When prompted for the data source to be refreshed, select **GetOrderLinesByOrderId**.
3. Click "**OK**" Button.



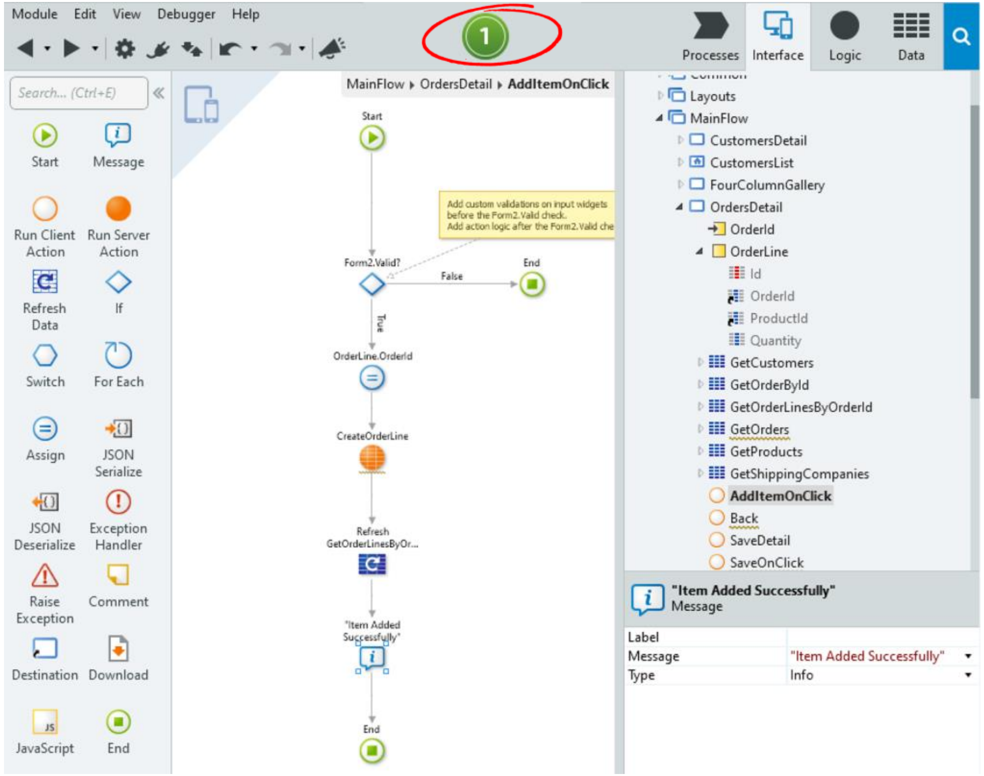
1. Add the **“Message” widget** to the Action Flow just below the “Refresh Data”.

2. Go to the properties panel: for the Message write **“Item Added Successfully”** for the type choose the option **“Success”.**



The Message provides a feedback message to the user. It has predefined types for "info", "success", "warning" and "error" messages.

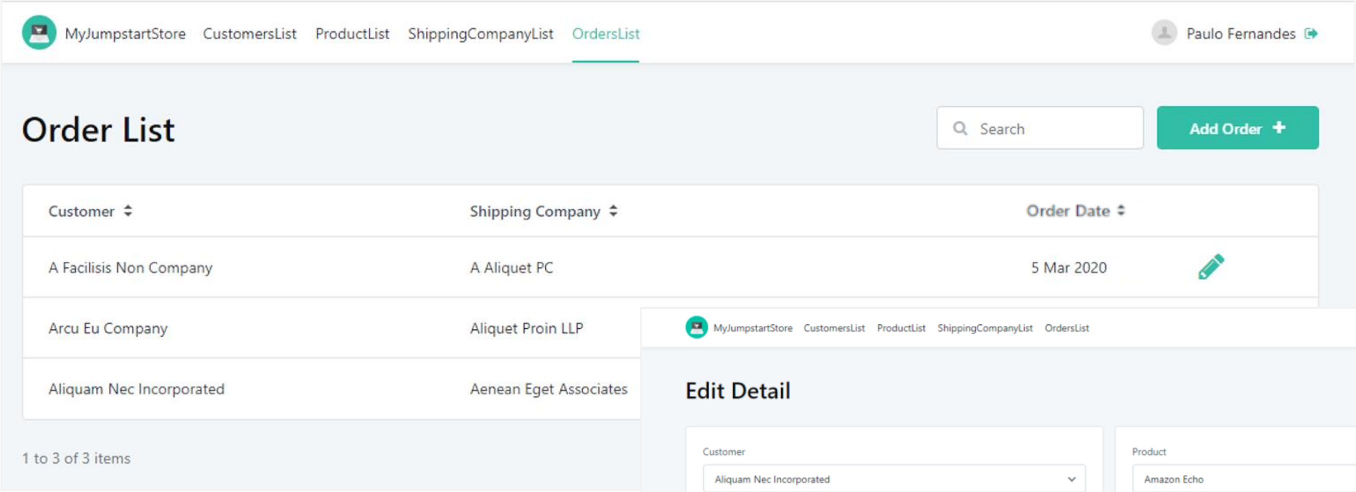
1. Click the “**1-Click publish**” to initiate the deployment process

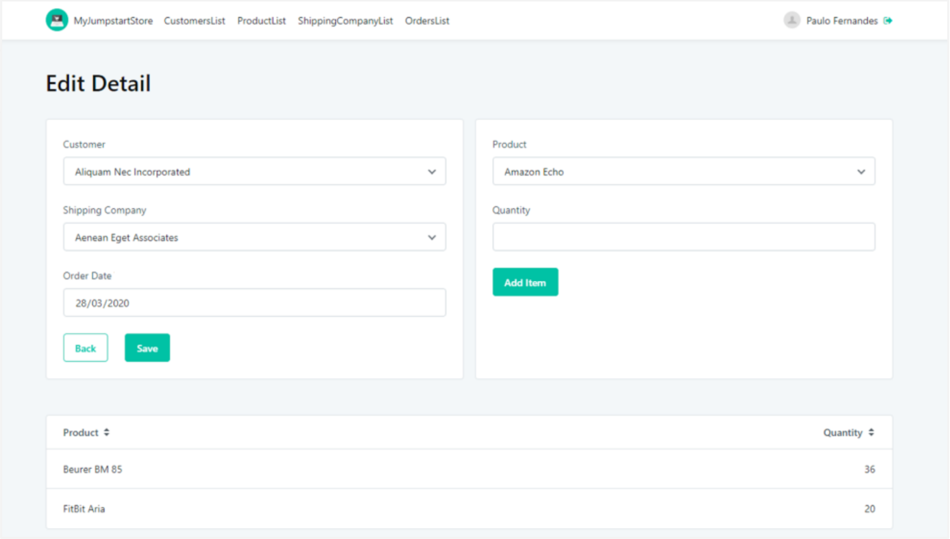


This will start the deployment and publication process with one simple click.

During this process, the Development Studio will upload the model definition of your application, and the server will generate standard code C#, React, HTML5, CSS3, Javascript, SQL, optimize it, compile it and publish it on an application server (IIS).

End of Section 2 - Product & Shipping





To test your application go to the OrderList page and create a new order. After creating the order you can click on the pencil to access its detail and start adding new order line items.

outsystems

Section 3

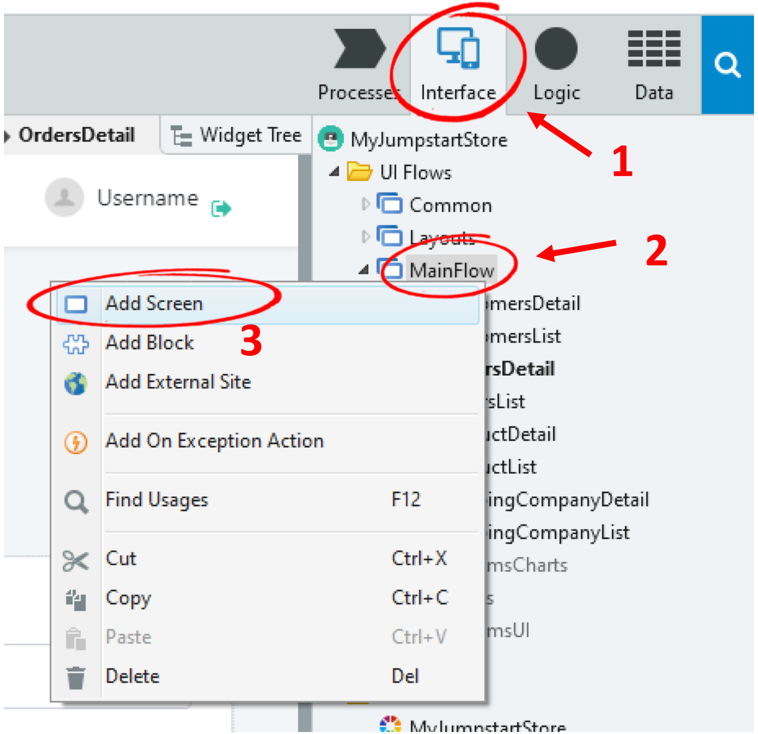
Image Gallery with fluid layout

The platform provides screen templates to help developers create world-class User Interfaces easily. Not only do the Templates contain UI design, but it also has pre-created actions to accelerate development of the page actions such as filters.

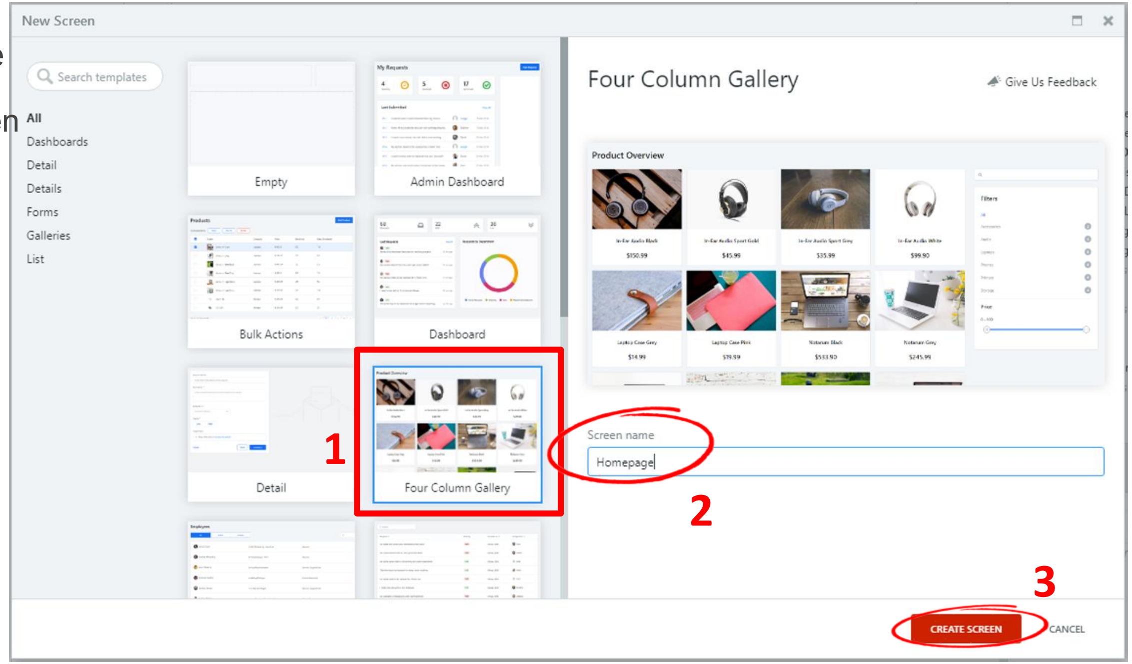
OutSystems UI templates pre-creates your screens with sample data that can be easily replaced so you don’t have to start from scratch.

In this chapter, we will create a Product Gallery page that contains filters and search capabilities.

1. On **Interface tab**, right click on ”**MainFlow**” and choose create a “**Add Screen”** to create a new web screen.

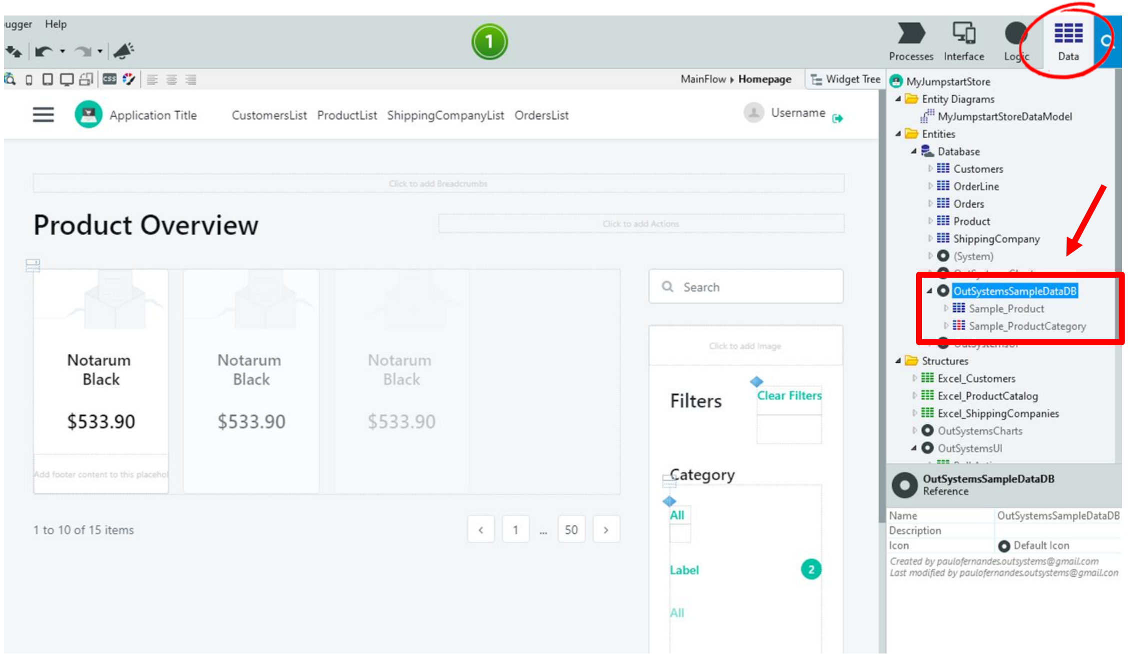


1. Select **Four Column Gallery** template
2. Add the title “**Homepage**” under Screen name
3. Click “Create Screen”



1. A page with pre-built template is automatically created.

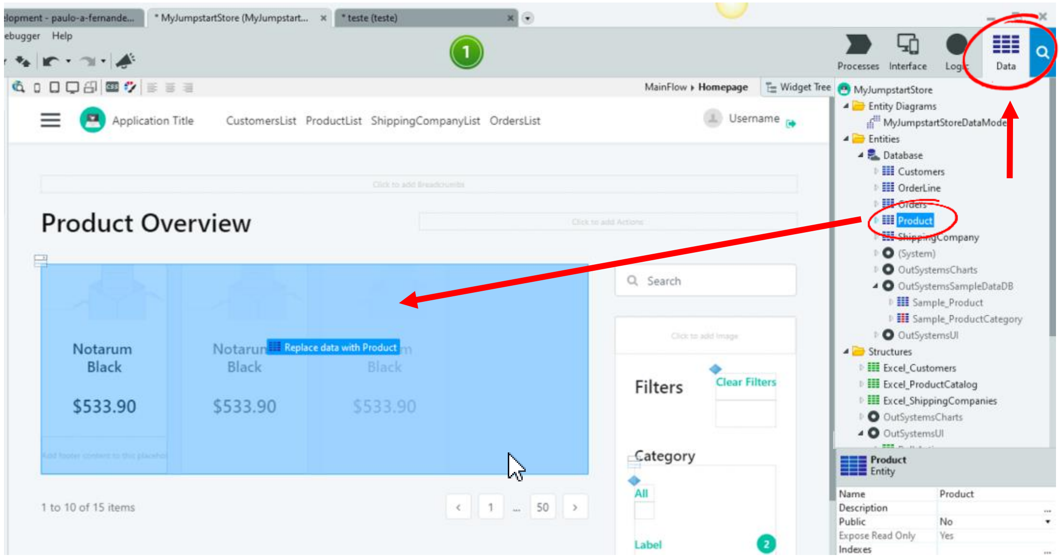
   Notice that it's using Sample Data "**Sample_Product**" and "**Sample_ProductCategory**".



The Screens created from the built-in Screen Templates have sample data that can you can replace manually or semi-automatically. Having sample data in the Screen is a good way to see how the Screen is designed and to get inspired for developing your app.

Replace Sample Data

1. In the **Homepage** screen, drag the **Product** entity onto **Product Overview section**. Make sure it indicates “**Replace Data with Product**”.

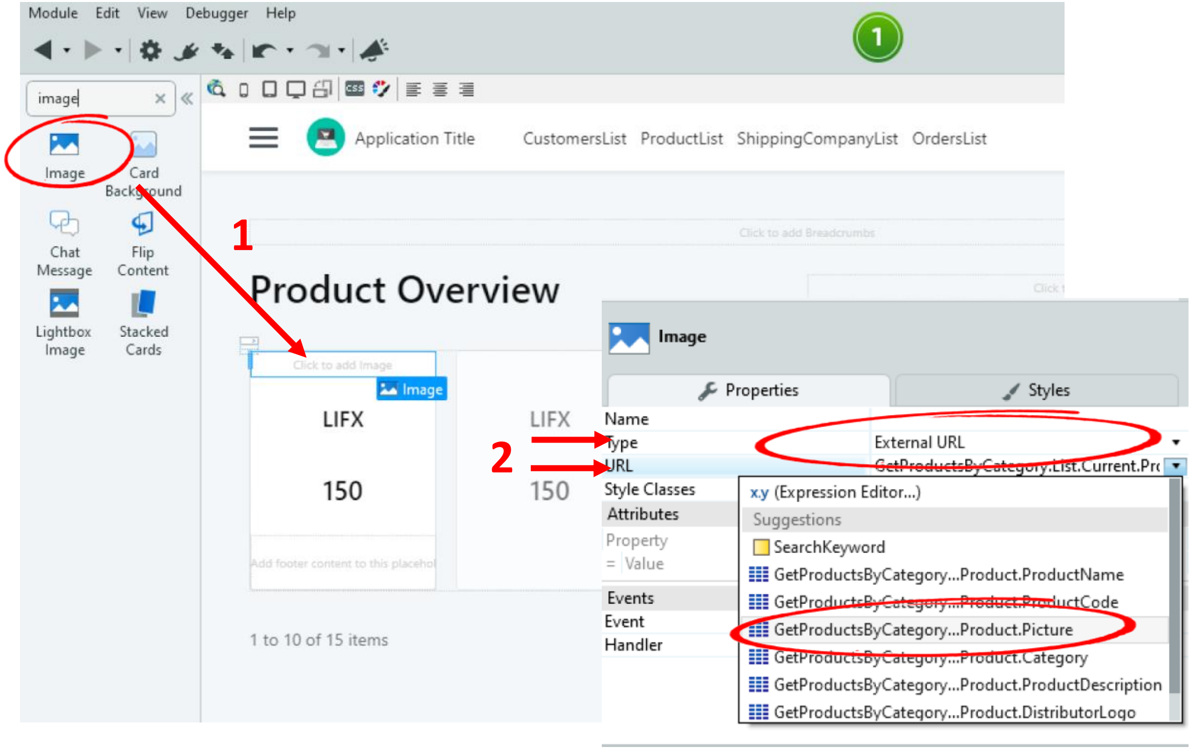


You can quickly adapt your screen templates to show data that is relevant to you. This is done by dragging and dropping the corresponding Entity over the Widget. This changes the data sources and updates the user interface automatically.

Add Product Image

1. **Drag an "Image"** widget to the Image section of the Product Gallery
2. In the properties of the Image widget node, change the Type to **External** and set the URL to

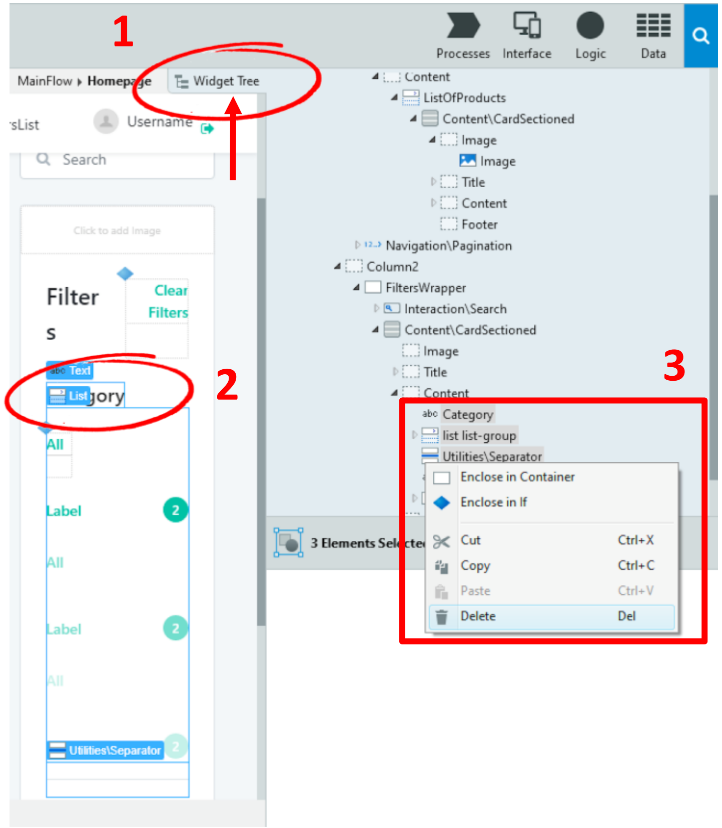
   GetProductsByCategory.List.Current.Product.Picture



- In the next steps we will adapt our screen and delete some of the filters from the initial template we don’t want to use.
- You can customize any template since it's just a starting point that gives you most of what you needed.

Delete Category Filters

1. Click to open the "**Widget Tree**" (Widget Tree)
2. Click on the "**Category**" title to locate the UI elements in the widget tree.
3. Delete the **Category Filter Controls**:
   a. Category
   b. SecondaryList
   c. Utilities/Separator



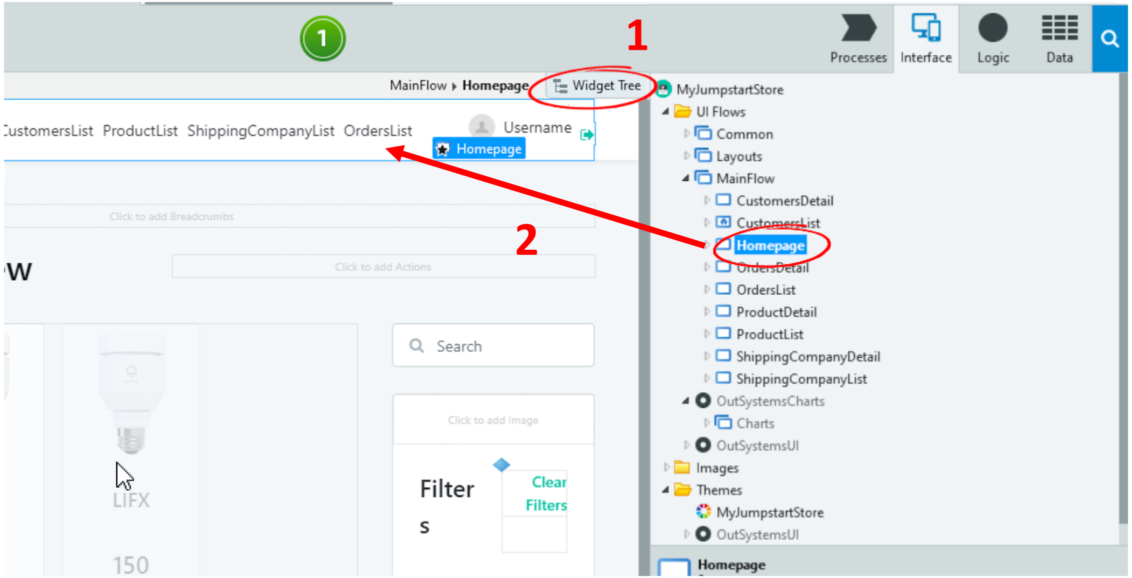
QUICK TIP

Try to use keyboard shortcut: **Ctrl+W** to show or hide Widget Tree

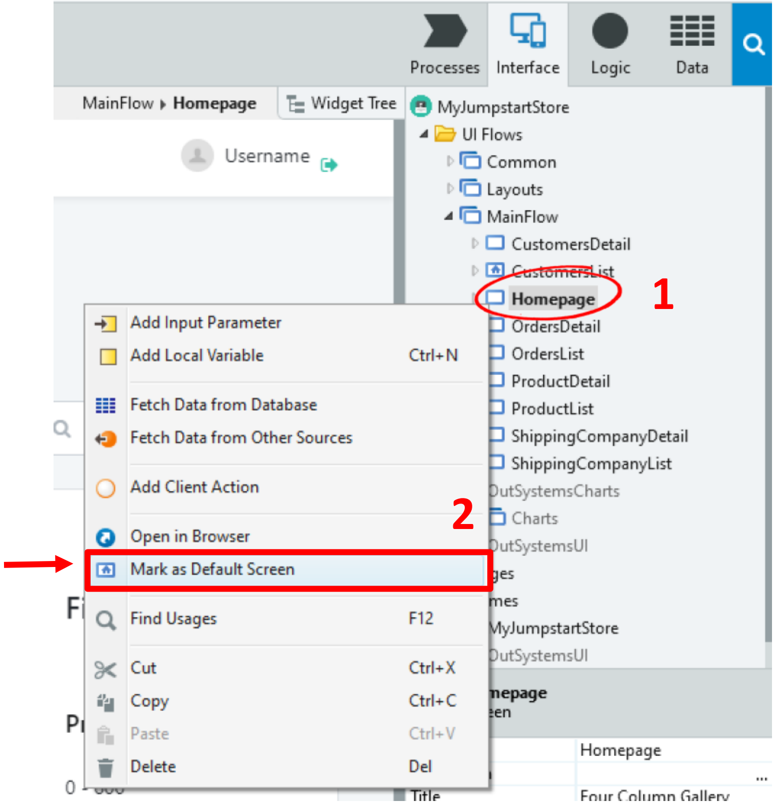
Section 3 C 2. Add page to menu area

Add Homepage to Menu

1. Go back to the normal view by clicking the “**Widget Tree**” button again
2. Switch to the “**Interface**” tab if needed, and add the **Homepage** to the Menu bar by dragging and dropping the screen into the Menu area.

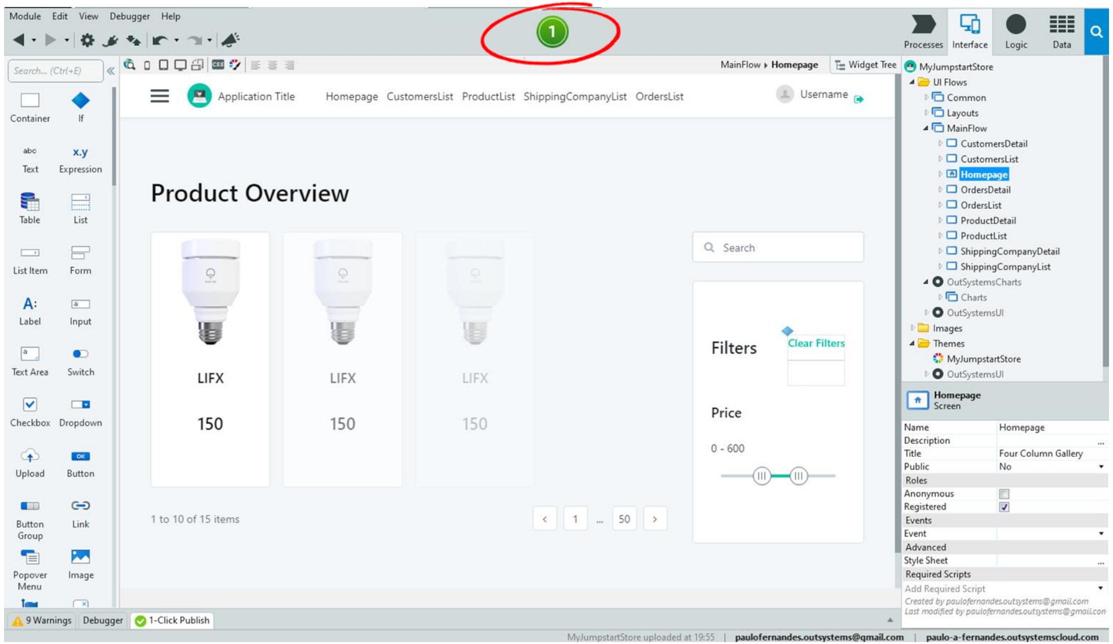


1. Set this page as the Entry Point by double clicking the **Homepage** screen and selecting the **Mark as Default Screen** option.



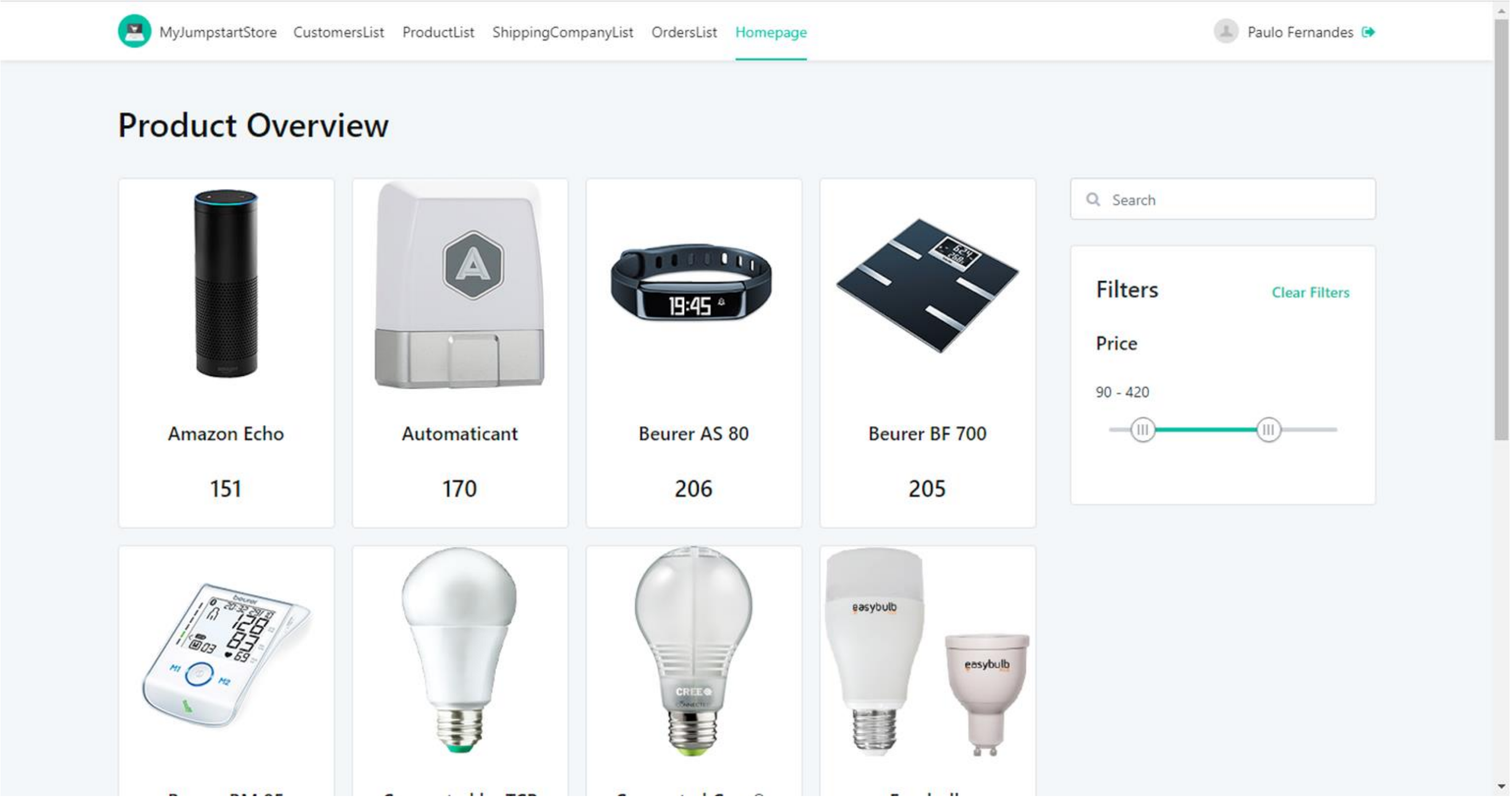
1. Click the "**1-Click Publish**" to initiate the deployment process.
   This will start the deployment and publication process with one simple click.

2. Click on the blue icon to see your web application running.
   *Once the deployment process is finished the "**1-Click publish**" is updated to a blue icon.*



Try to use keyboard shortcut: **F5** to do 1-Click Publish

End of Section 3 - Image Gallery with fluid layout



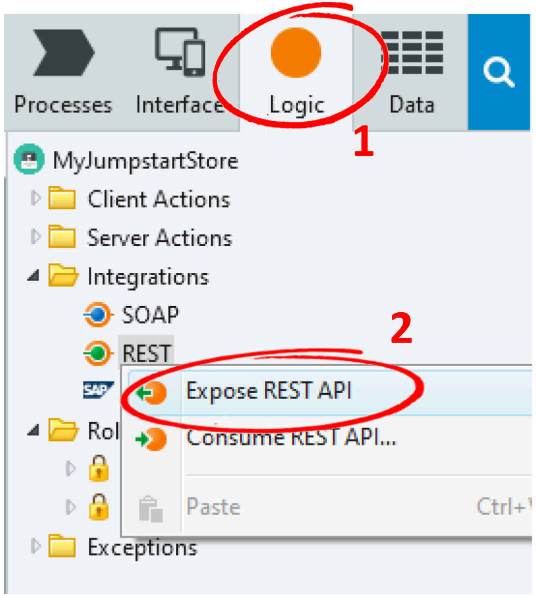
Section 4

Expose REST API with your products

We want to showcase how you can expose and consume a REST Web Service. In this section we will expose part of the functionality as a REST Web Service and later on the mobile application we will consume this REST Web Service.

Please note: If you wanted to consume a service from another module, there is a much easier and standard way to do this. Your instructor will demonstrate how to do this later in the day.

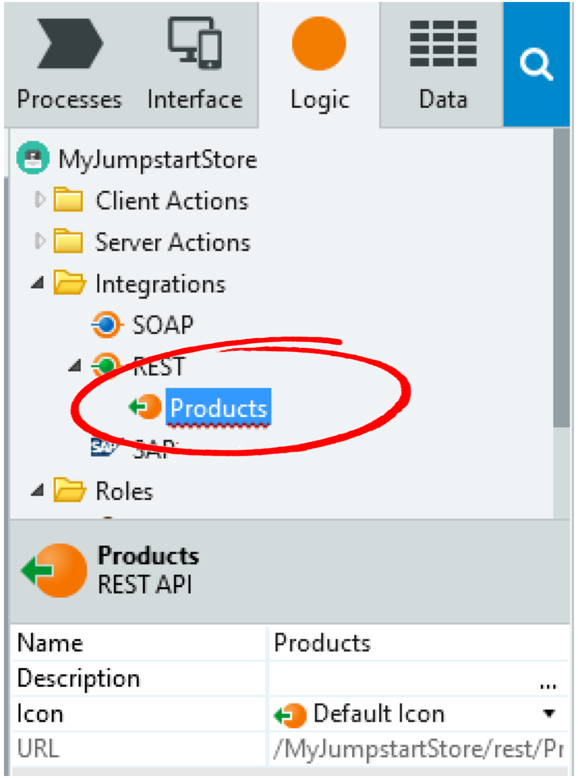
1. From the "**Logic**" section go to the Integrations folder
2. **Right click** the REST tab and select **Expose REST API**



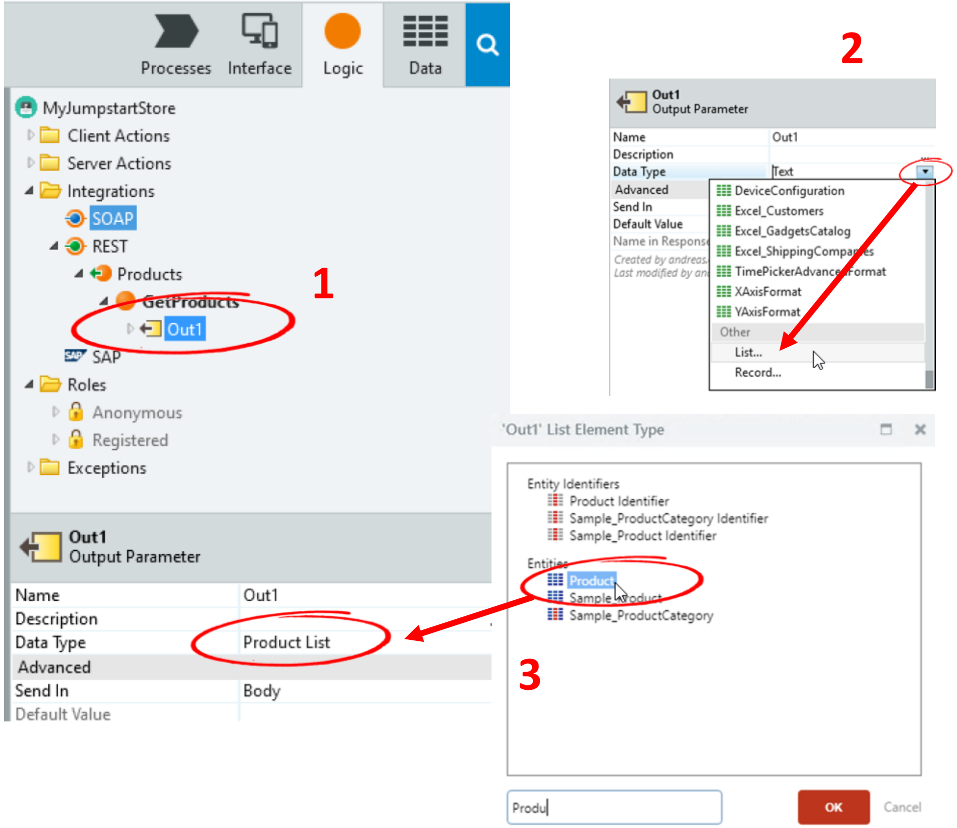
QUICK TIP

1. Try to use keyboard shortcut: **Ctrl+3** to access Logic tab

1. Rename the REST Integration to **"Products"**
2. Add a new REST API Method and rename it to **"GetProducts"**



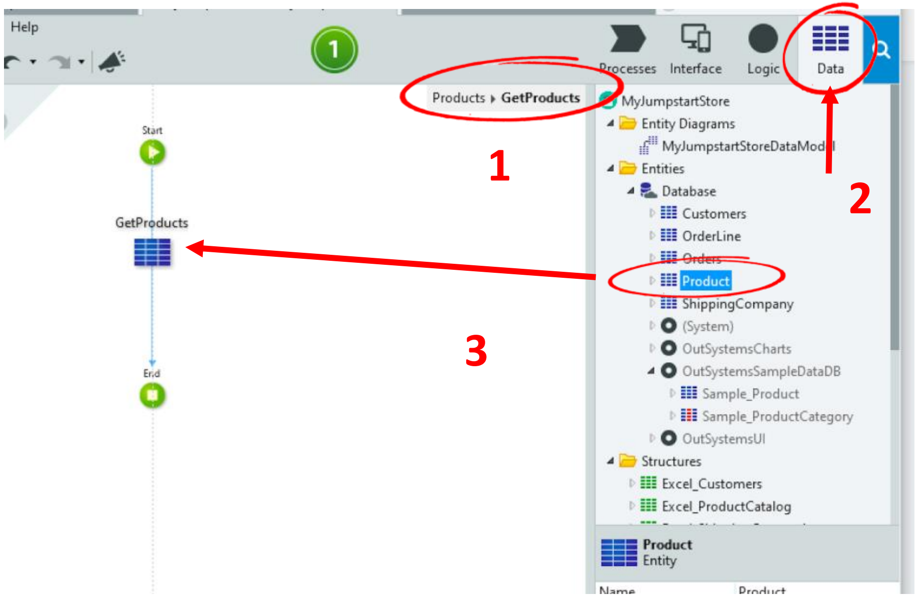
1. Add an **output parameter** named **Out1**. Make sure the Data type is set to **Product List**.



We have created the Web Service and one method. Let‘s fill the method with logic in a low code way.

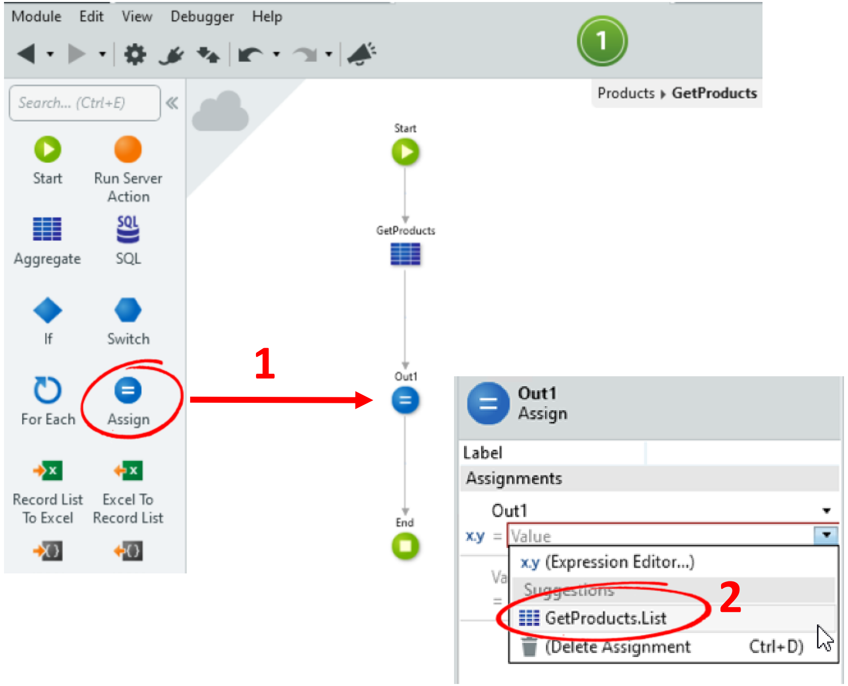
Add Product Aggregate

1. On the “Logic” tab, double click the “GetProducts” REST API method.
2. Go to the “Data” tab
3. Drag the “Product” Entity on top of the “GetProducts” REST API method flow.

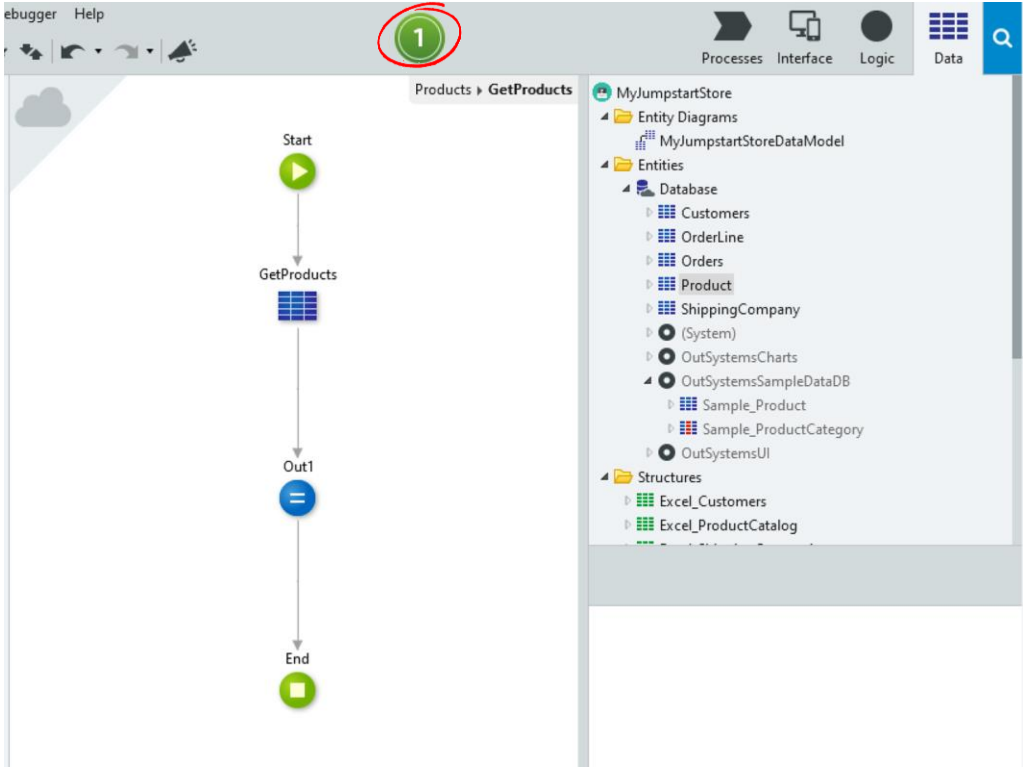


Assign Output Parameter

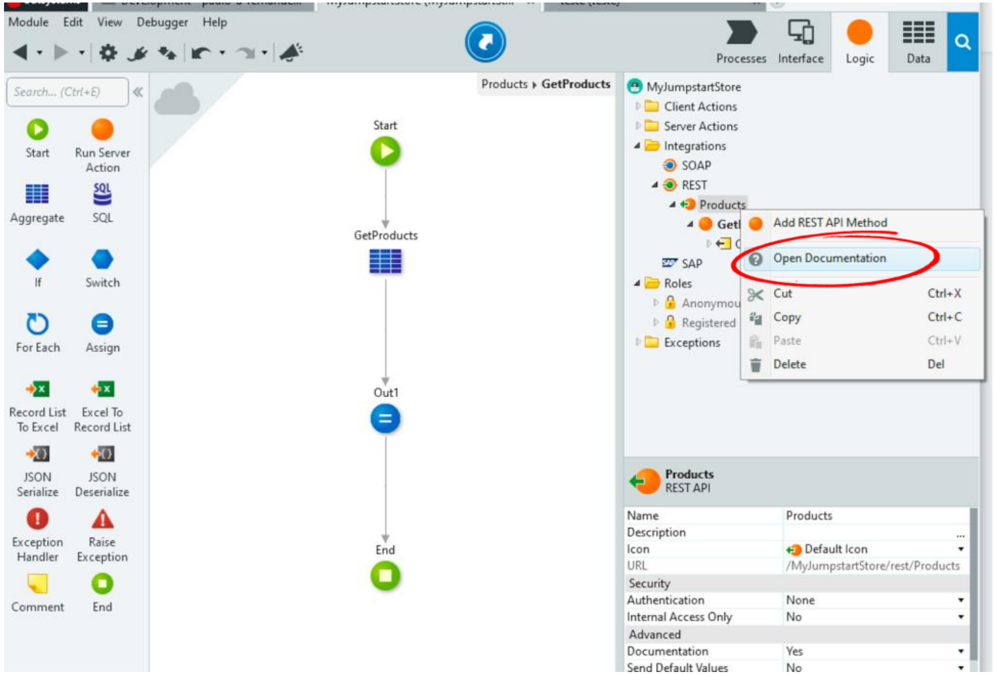
1. Add an **“Assign”** step.
2. Assign to the **“Out1”** parameter the **“GetProducts.List”** (result of the query).



1. Click the “**1-Click publish**” to initiate the deployment process.



1. After the deployment, **right click** on top of the **Products API** and click "**Open Documentation**".



outsystems REST API Documentation

Products

[API DEFINITION URL: https://paulo-a-fernandes.outsystemscloud.com/MyJumpstartStore/rest/Products/swagger.json]

List Operations | Expand Operations

GET /GetProducts

Request URL

```
https://paulo-a-fernandes.outsystemscloud.com/MyJumpstartStore/rest/Products/GetProducts
```

Response Messages

HTTP Status Code	Description	Content-Type	Data Type (Model | Example)
200		application/json	see below

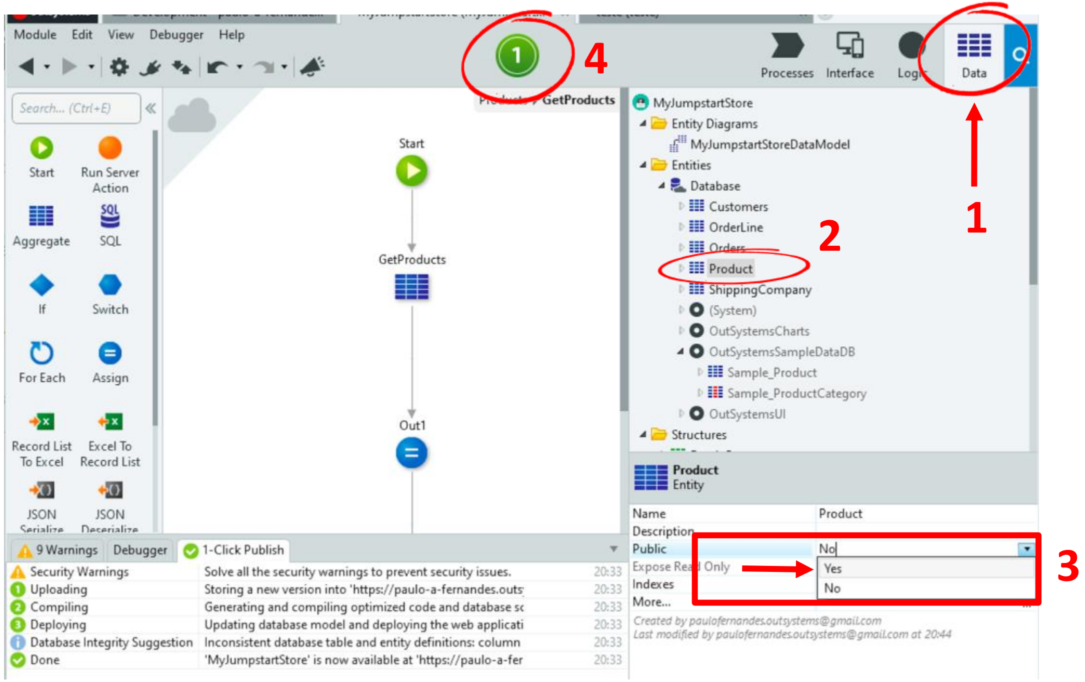

```
[
  {
    "Id": 1234567891234567,
    "ProductName": "string",
    "ProductCode": "string",
    "Picture": "string",
    "Category": "string",
    "ProductDescription": "string",
    "DistributorLogo": "string",
    "Price": 0,
    "DistributorPhone": 0.1,
    "QuantityStock": 0
```


[BASE URL: /MyJumpstartStore/rest/Products]

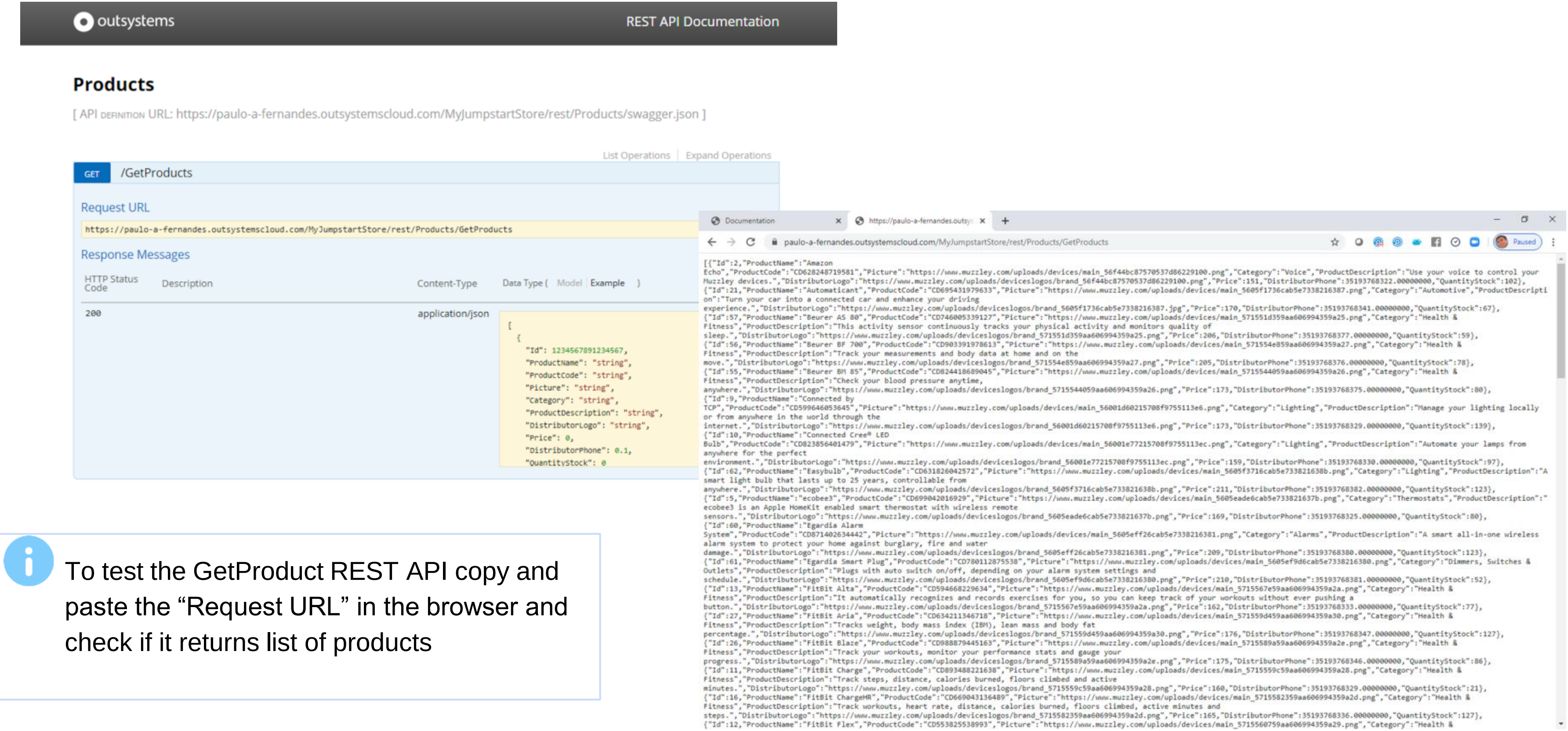
C. Reusable Product Entity

In a later exercise, we will reuse the Product entity in our Mobile Application. We need to expose the Product entity as public to make it accessible to other OutSystems apps..

1. Go to the Product **Entity** and set the **Public** property to **Yes**.
2. Click the “**1-Click publish**” to initiate the deployment process.



End of Section 4 - REST API



To test the GetProduct REST API copy and paste the “Request URL” in the browser and check if it returns list of products

outsystems

Bonus Exercise

Ideas to improve your application

In this section you will find two bonus exercises that do not include a step-by-step description on how to solve them. These exercises will help you develop your skills and knowledge on the OutSystems Platform.

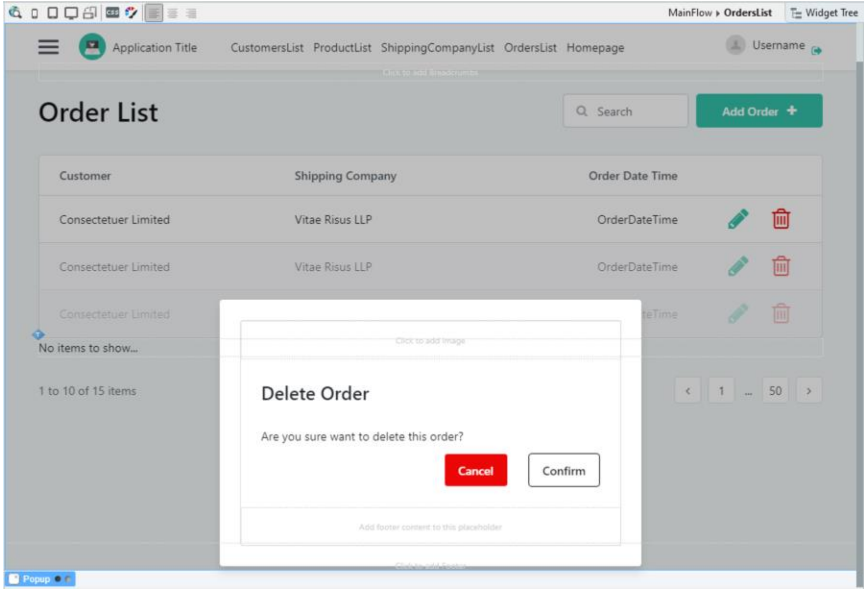
Good Luck!

Bonus exercises are optional!

1. Enable users to delete orders

Deleting is a sensitive operation so we would like to have a confirmation message popping-up before proceeding with this operation.

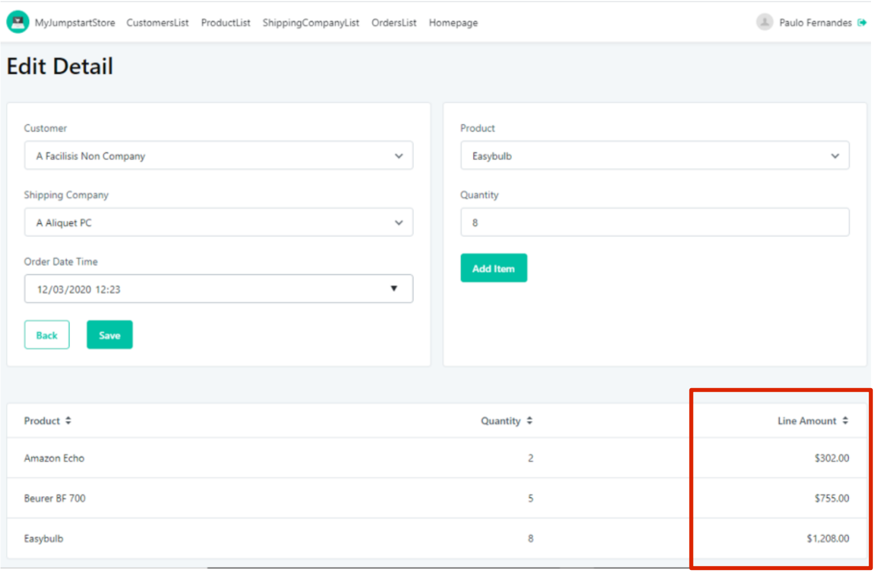
- Add a new Icon on the Order List table;
- On the Order List screen use the pop-up widget (and the card sectioned widget) to display a confirmation message;
- Create a Boolean local variable to control when the popup should be displayed (The “Trash” icon should show the pop-up; The Cancel button should hide the pop-up; the Confirm button should start a new client action to delete the corresponding order.
- Finally create the business logic to delete the corresponding order;



Display Amount/Cost for each Order Line

For each Order Line lets calculate its cost and display it on the Order Line table that is on the Order Detail screen. For that:

- Add a new attribute “LineAmount” to the OrderLine entity;
- Then, on the “OrdersDetail” page adapt the “Add Item” button logic to:
  1. Obtain the price information from the Product Entity based on the product selected by the user (tip: use an aggregate, its filters and the OrderLine variable).
  2. Update the existing assign activity to assign a new value to the new “LineAmount” attribute (this value should be the product between the OrderLine quantity and the Product price).
- On the Order Detail screen drag the “LineAmount” attribute from the OrdersLine Entity to the OrderLine table.



Congratulations!

You have successfully built a web app

outsystems

Jumpstart training