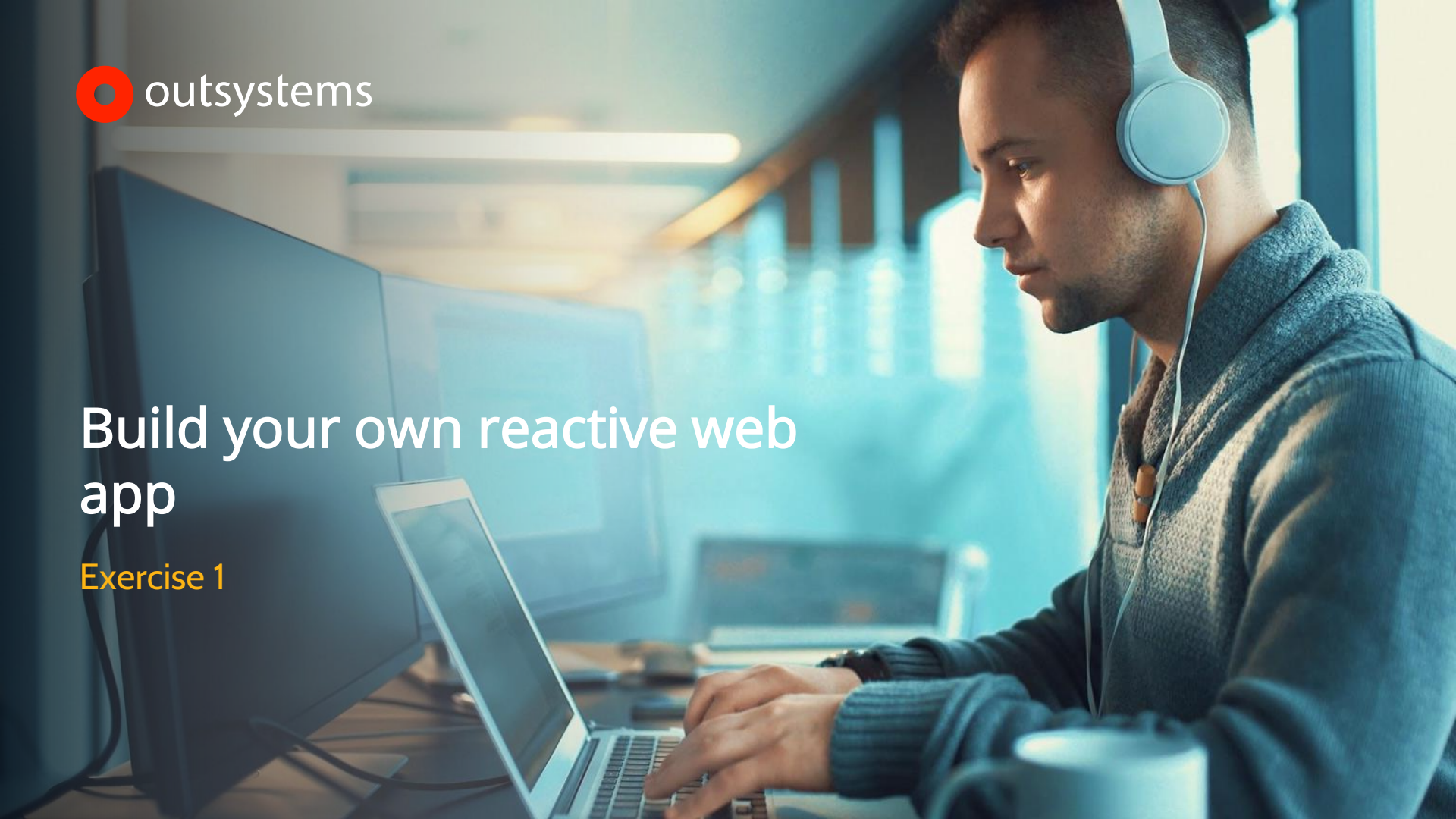
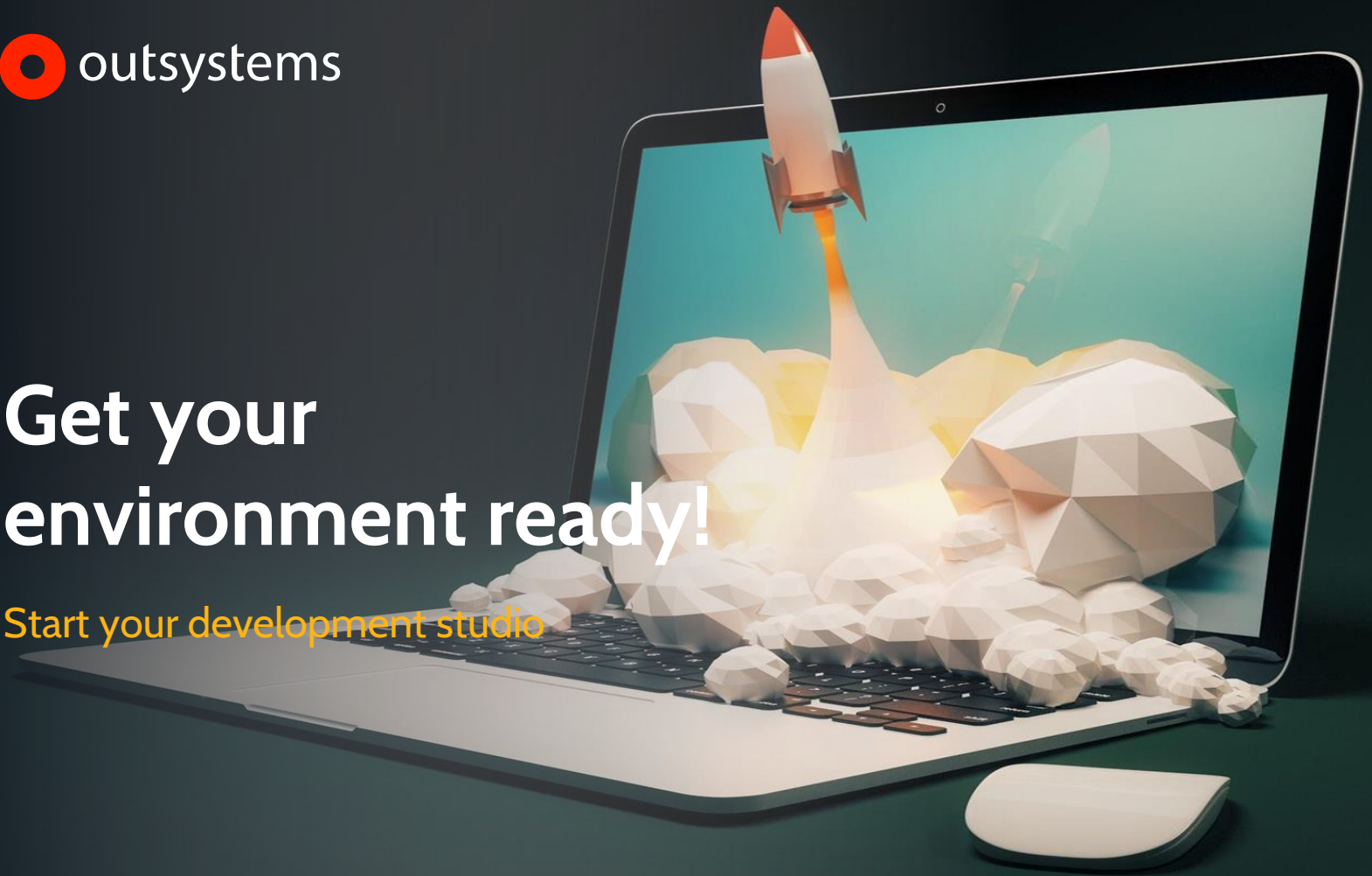
outsystems

Build your own reactive web app

Exercise 1



outsystems



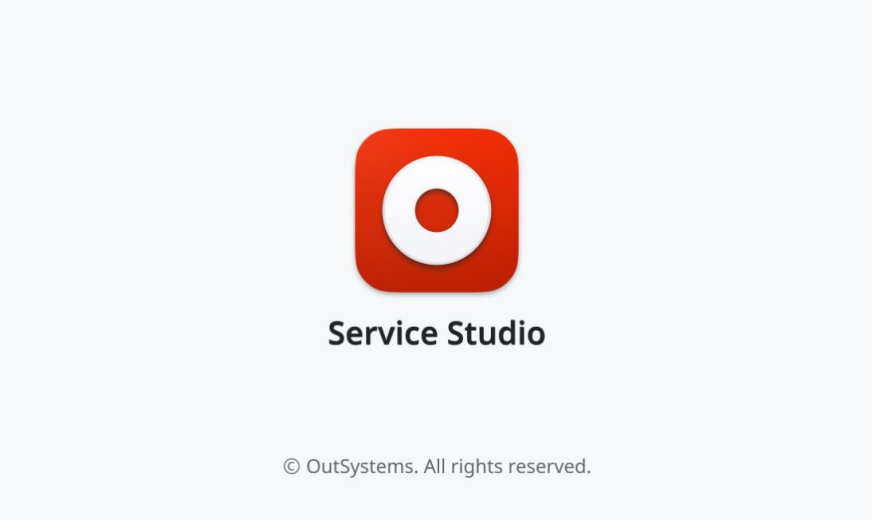
Get your environment ready!

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 app**.



outsystems

Web application

Exercise 1

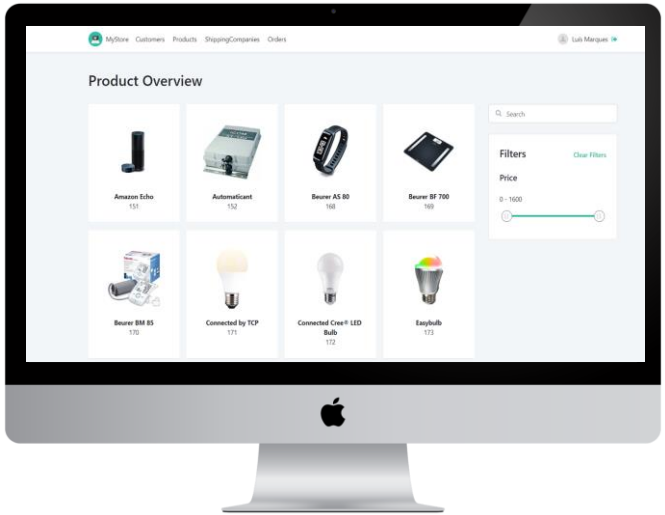
Context

"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

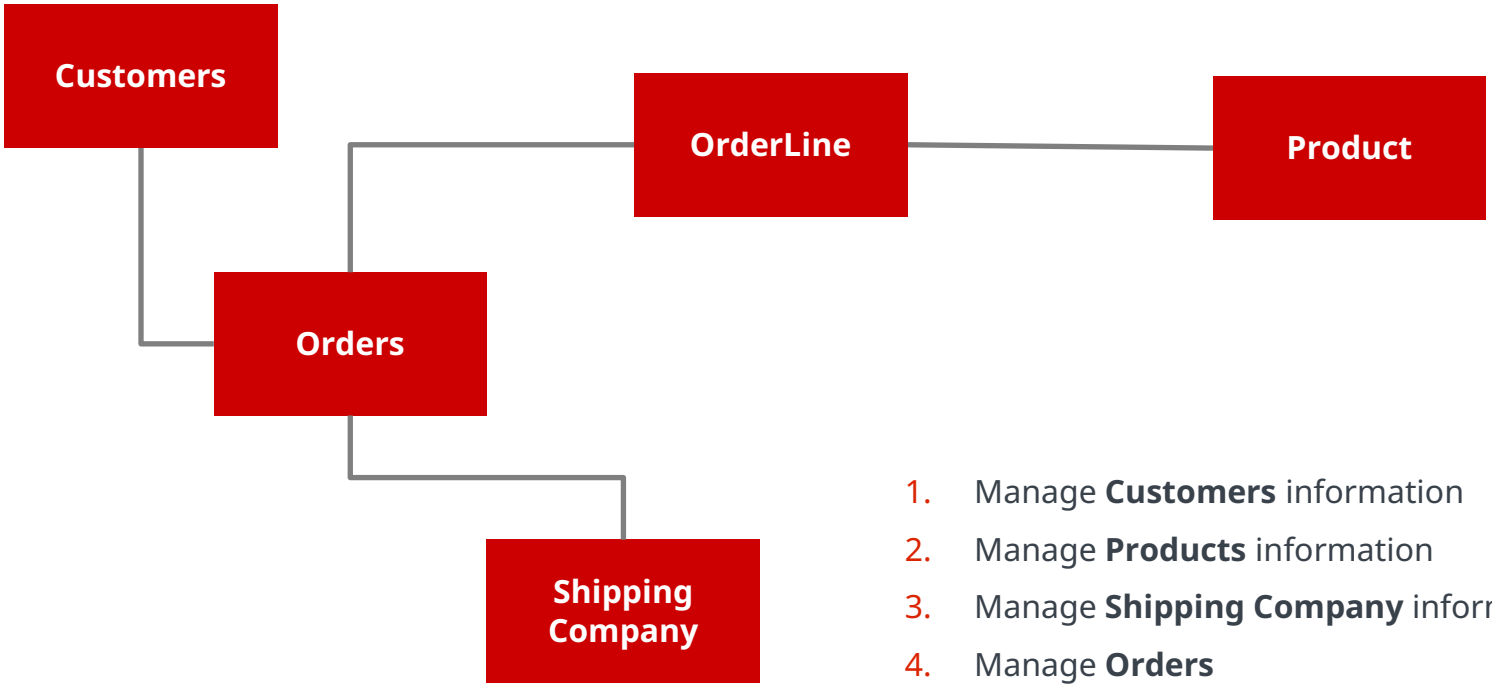


"As a **customer representative** I should be able to search for customers, edit their information, create new customers and register their purchases".

"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**

Section 1

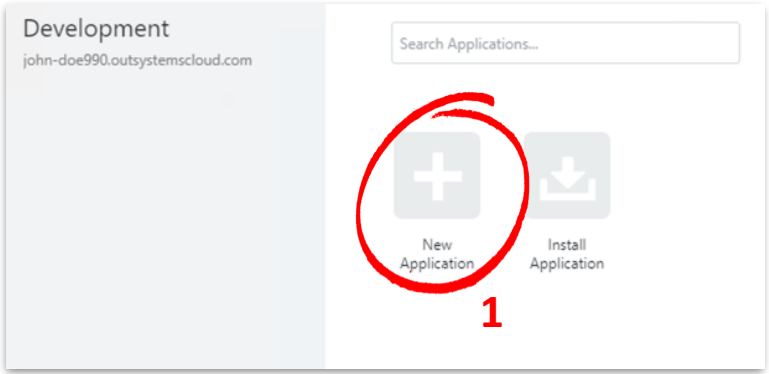
Customer management

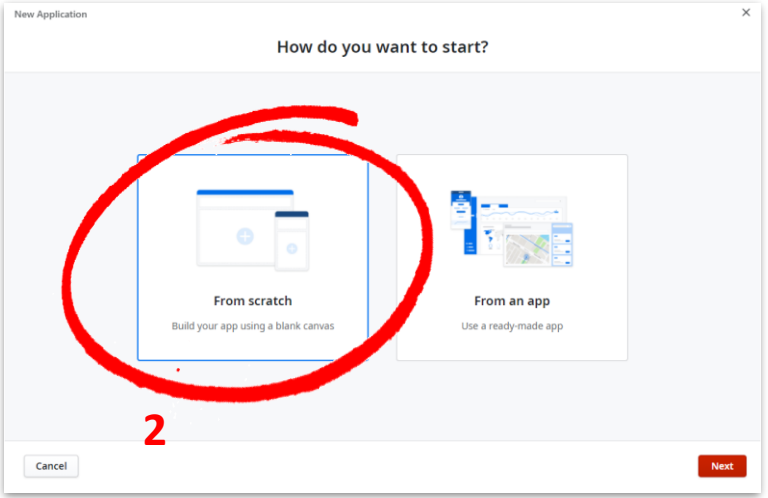
In the following steps you'll learn how to create your own responsive web application.

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.

Create a new mobile app

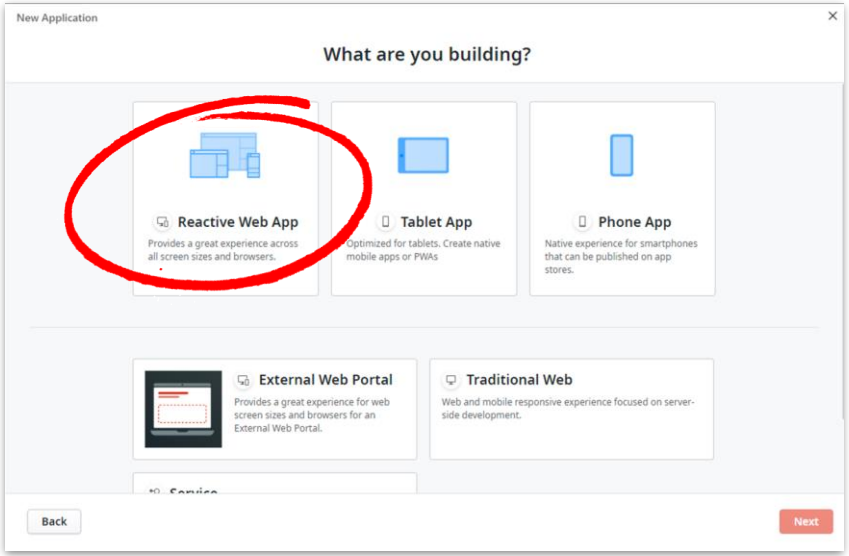
1. Click **“New Application”**;
2. Select **“Start from scratch”**.





Create a new mobile app

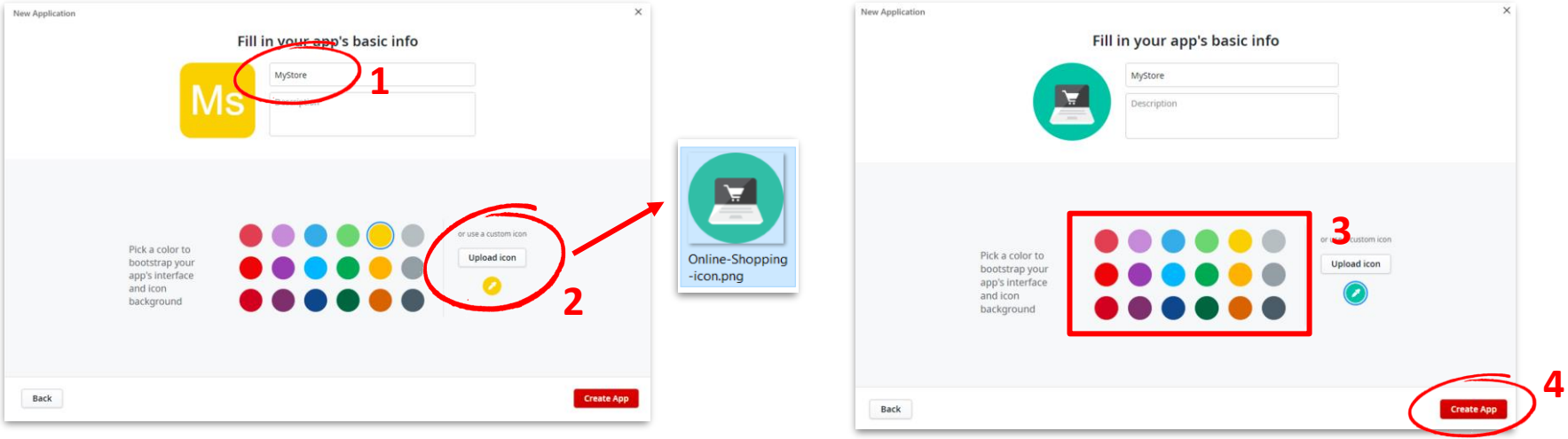
3. Select “**Reactive Web App**”;
4. Click **Next**.



*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*

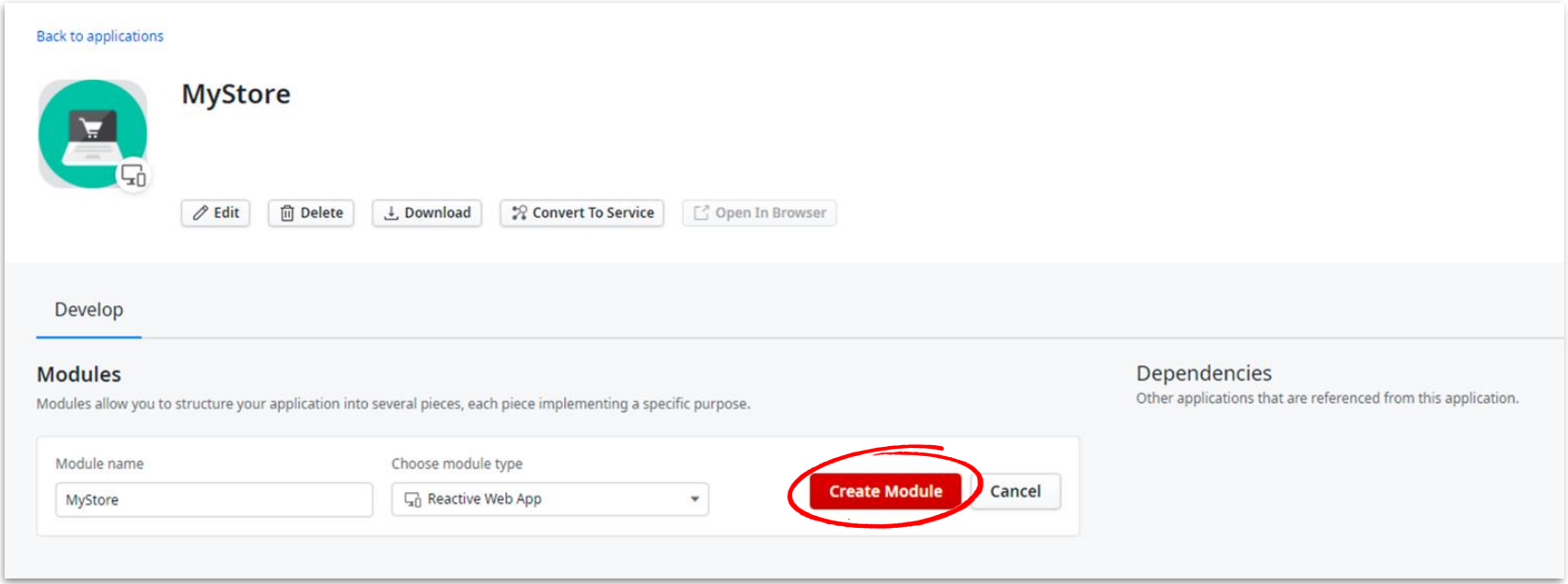
New Application

1. Type the **Name** of your application “**My Store**”;
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

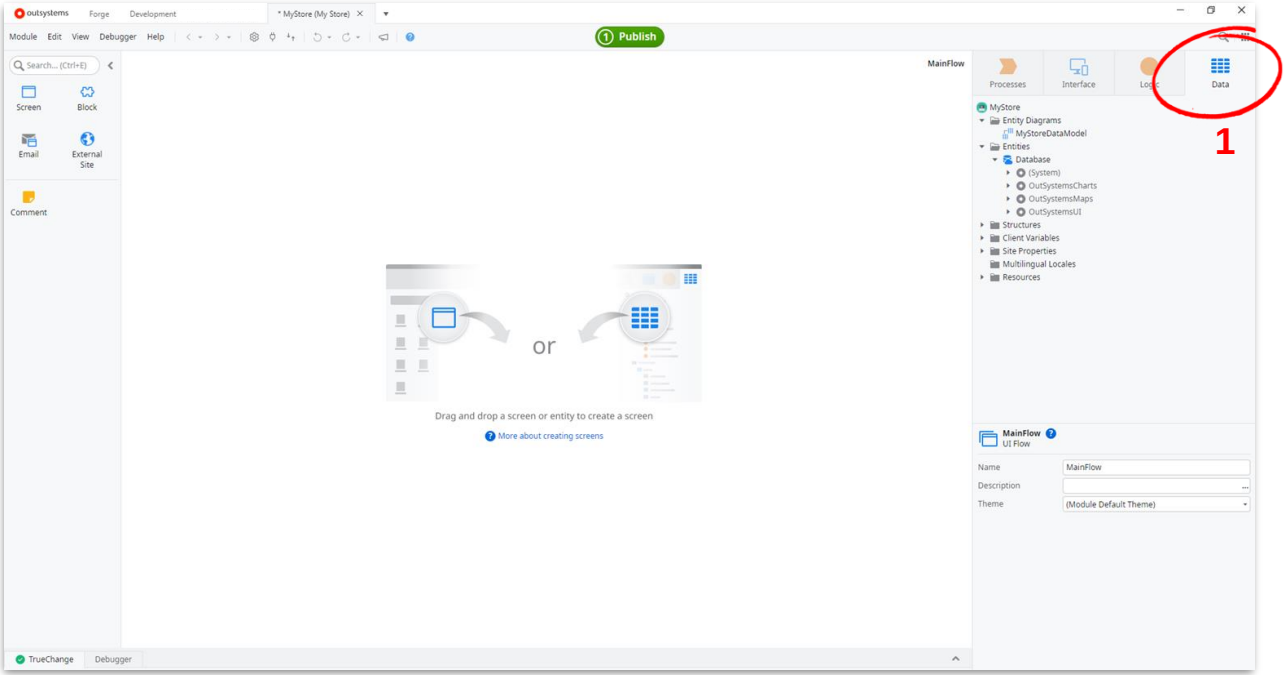
1. Leave the default Module Name;
2. Click **Create Module** button.



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.

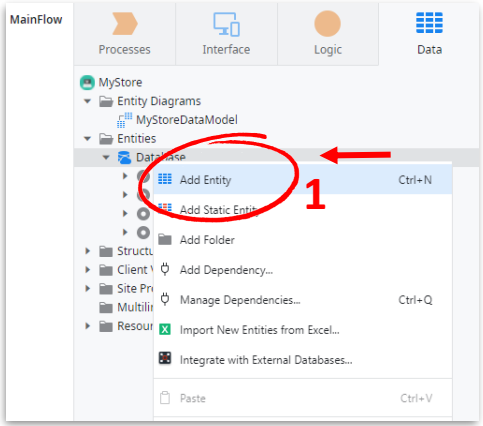


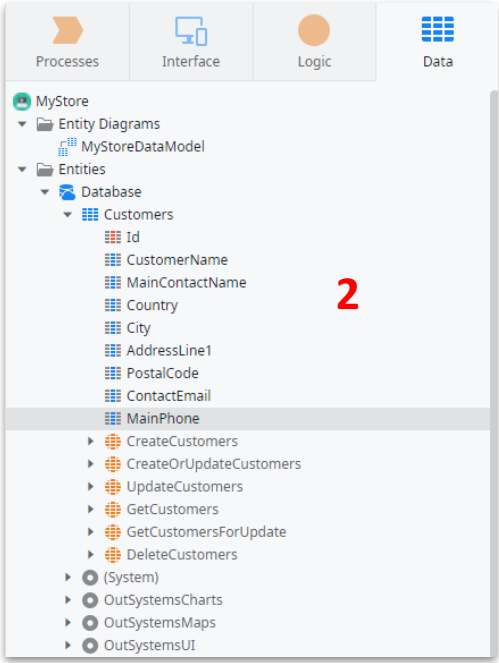
QUICK TIP

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

Create the "Customer" Entity

1. Choose the option "Add Entity" to **create a new entity** to store information, by right-clicking on "Database" under Entities;
2. Create the **"Customers" entity** with the attributes specified in the table below.





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	

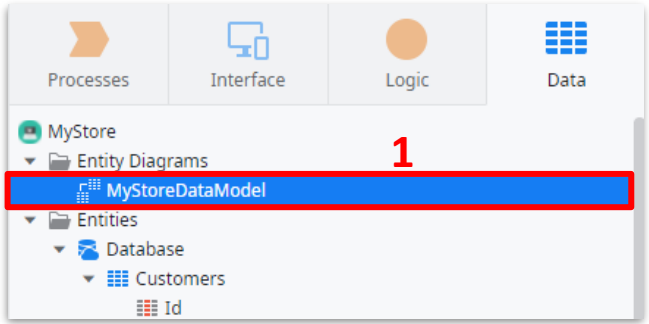
QUICK TIP

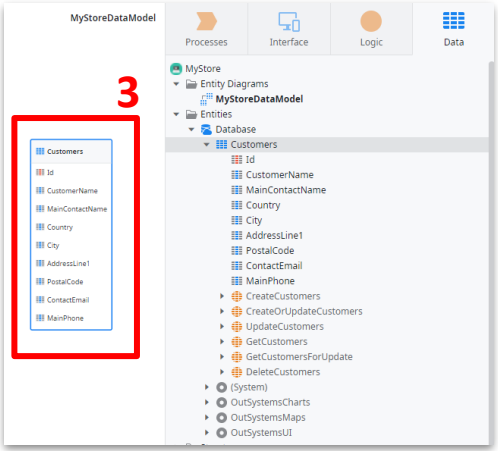
1. Try to use the keyboard shortcut: **Ctrl + N** to create Entity or Attributes;
2. Use keyword for attributes name, will be automate attribute Data Type (example: *Phone for Phone Data Type).

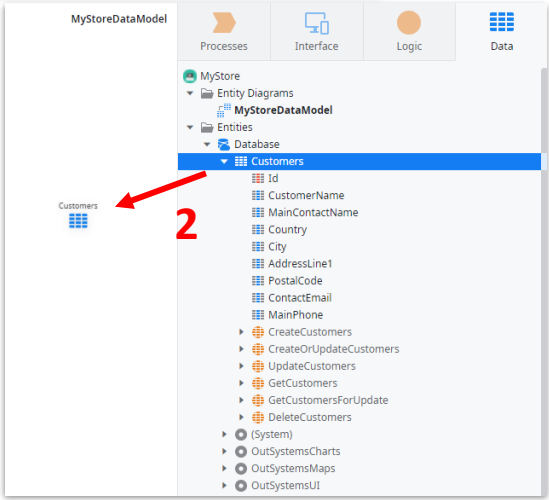
Add the "Customer" Entity to Entity Diagrams

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

1. Go to the "Data" tab > "Entity Diagrams" and double-click on the "**MyStoreDataModel**";
2. Drag the **"Customers"** entity to the diagram.

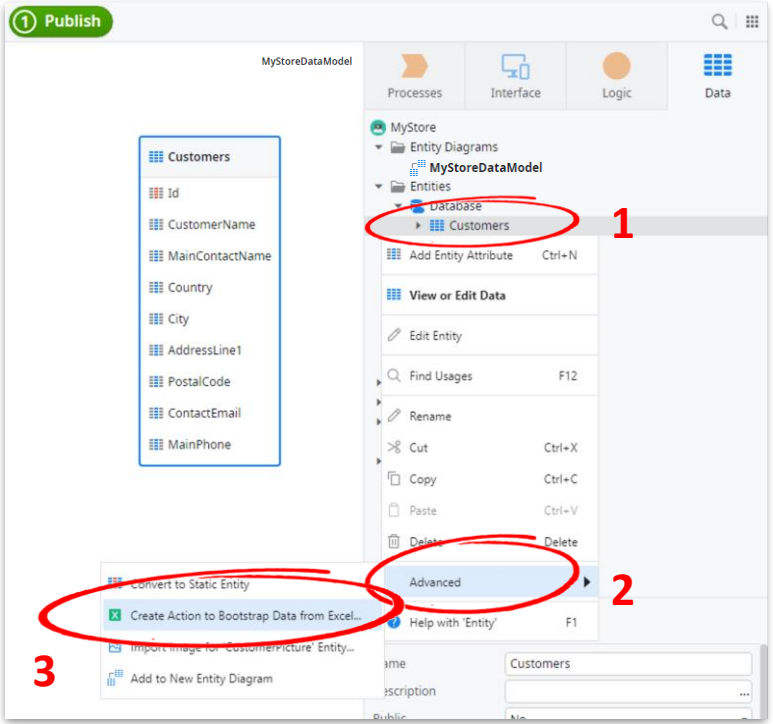




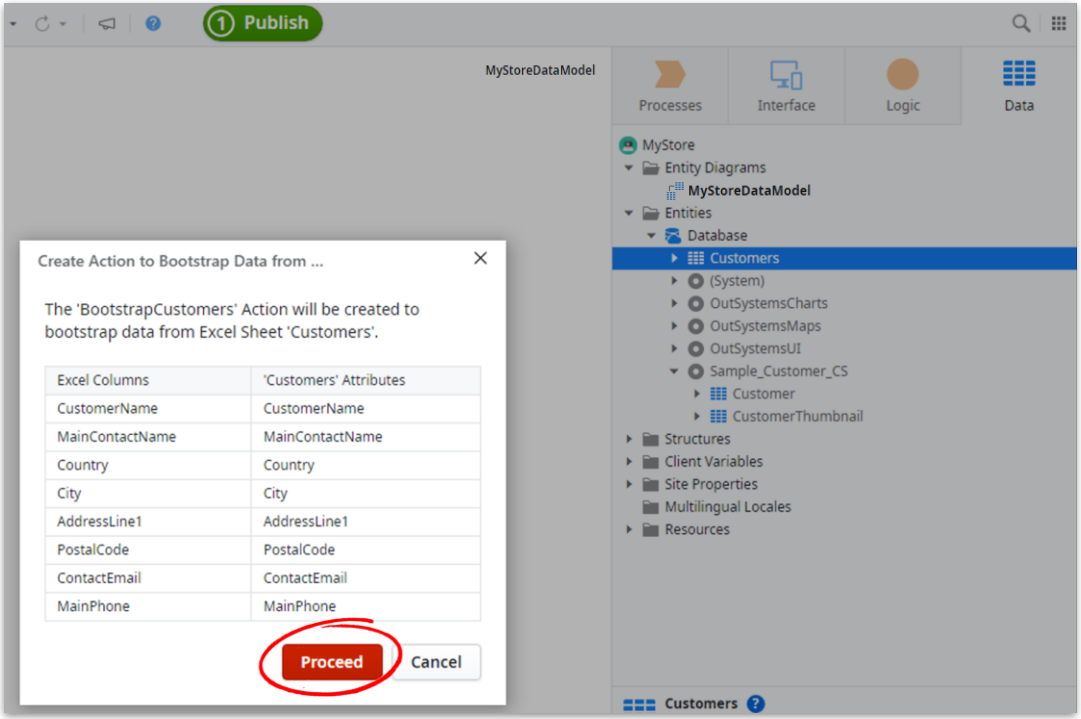


This diagram will show us how the entities are related to each other.

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;
2. Go to “**Advanced**”;
3. Click “**Create Action to Bootstrap Data from Excel…**”. 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.

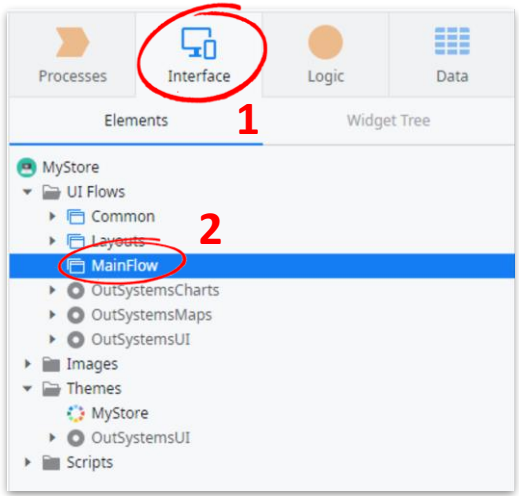
There are multiple ways we can quickly build screens in OutSystems:

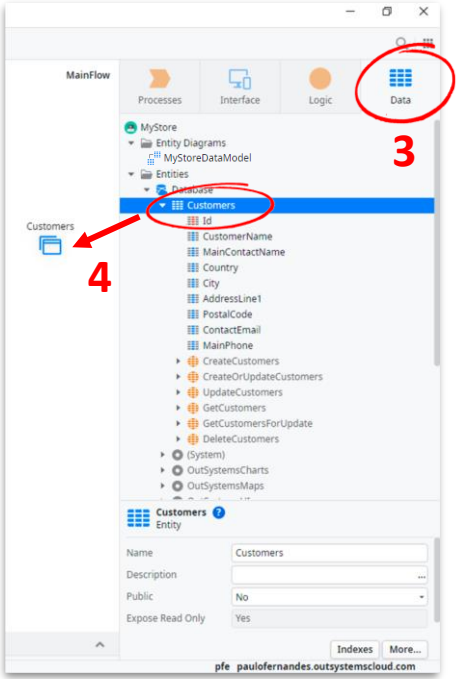
1. From a **blank slate**, inserting each component with drag-and-drop
2. From one of our many **screen templates** - these come with sample data that can be modified for your own needs
3. Using a feature called "**scaffolding**" which automatically creates lists and forms based on your database entities.

We'll explore these different methods during the exercises.

List page with pagination and search

1. Go to the "**Interface**" tab;
2. Double-click on the "**MainFlow**" UI flow;
3. Then go to "**Data**" tab;
4. Drag the "**Customers**" entity to the "**MainFlow**".

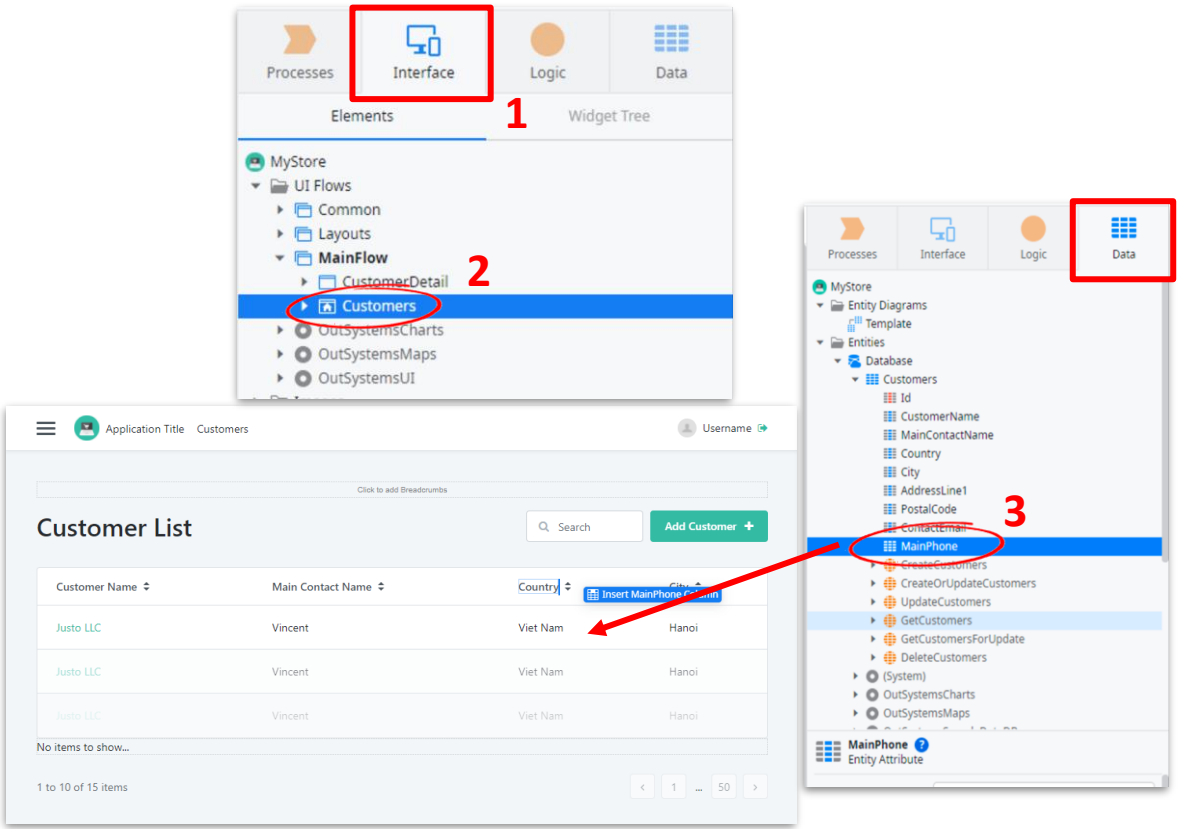




1. OutSystems has interface accelerators (called "**scaffolding**") to automatically create pages based on Entities.
2. When you drag an entity to the screen, OutSystems auto-creates a **listing** and **detail** page

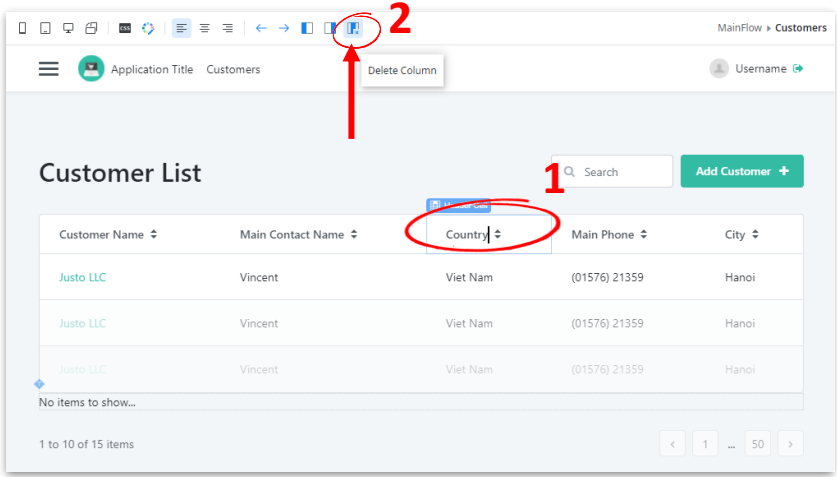
Add data to the interface

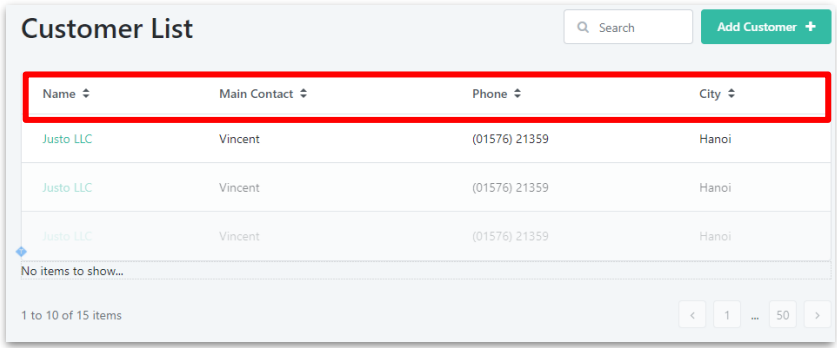
1. Go to the **“Interface” tab**;
2. Double-click on the **"Customers"** screen;
3. Go to the **“Data” tab**, and add a **new** “**Main Phone” column** by dragging it from the “Customers” entity and dropping it on the table created.



Delete data from interface

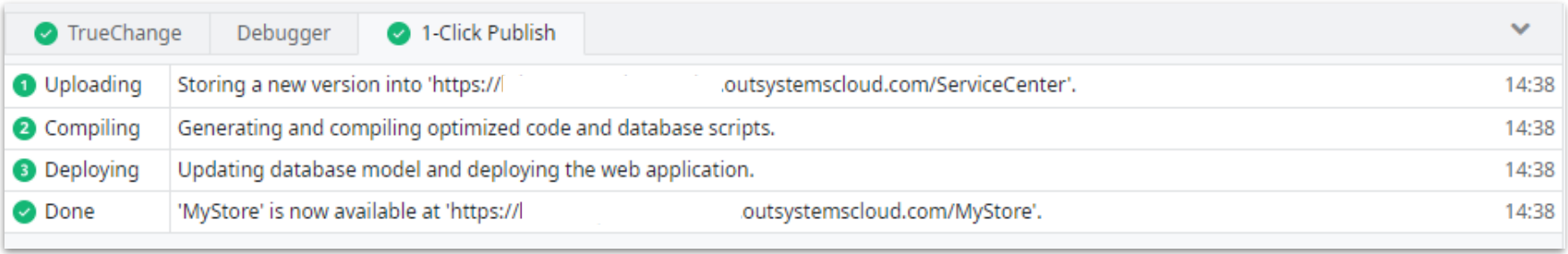
1. Select **column "Country"** of the table;
2. Use the "**Delete Column**" action to delete the "Country" column;
3. Change the labels of the headers of each column, by editing the text
   - "Customer Name" to "Name";
   - "Main Contact Name" to "Main Contact";
   - "Main Phone" to "Phone".





- Once you are done developing, you need to deploy the application to the cloud. This is accomplished with only one 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.

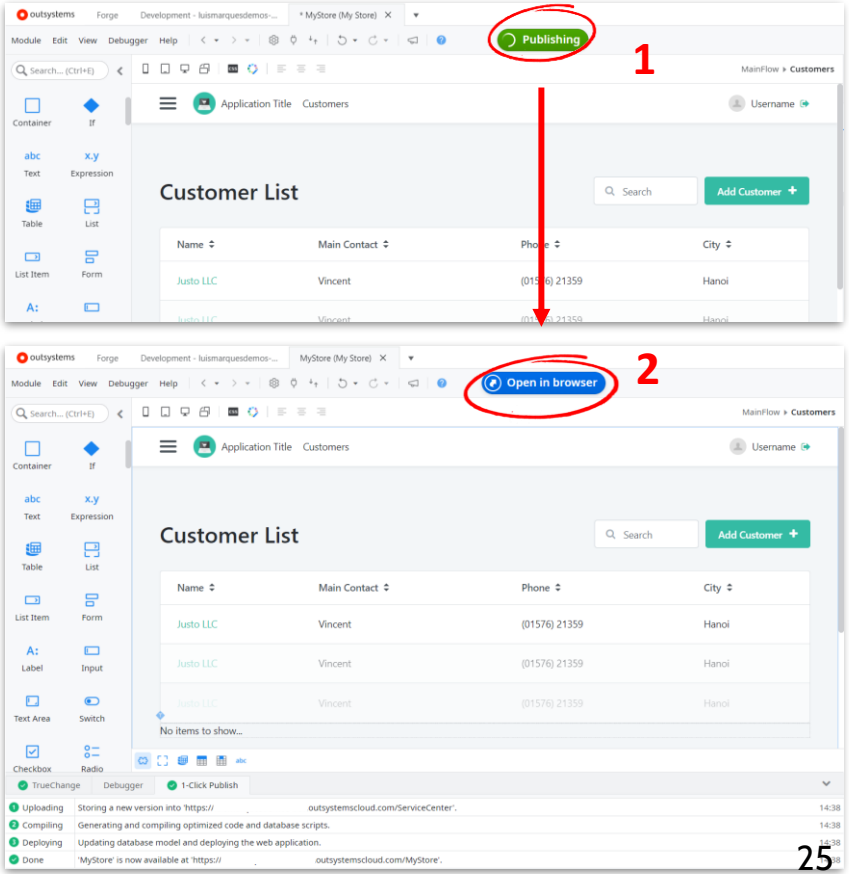
- You can monitor this process in the “1-Click Publish” tab (placed in the bottom of your development environment, and opens automatically when the process starts).



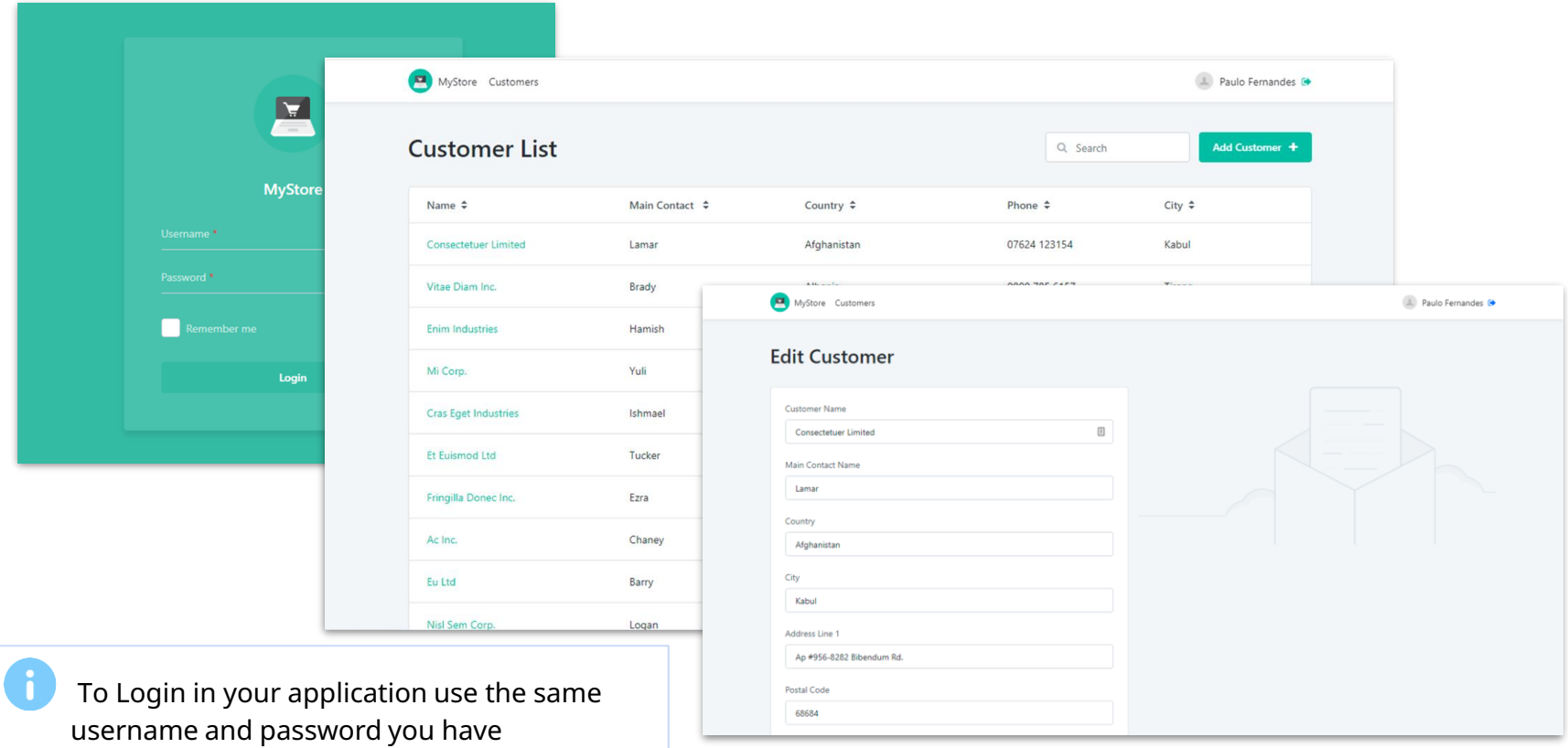
1. Click the "**1-Click publish**" to initiate the deployment process with a single click;

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.

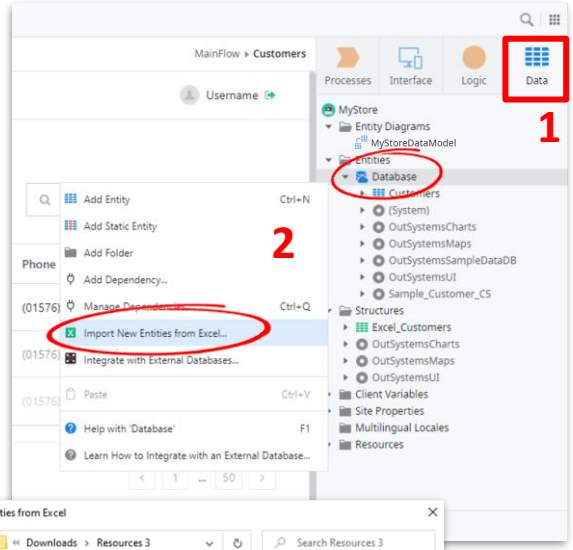
Section 2

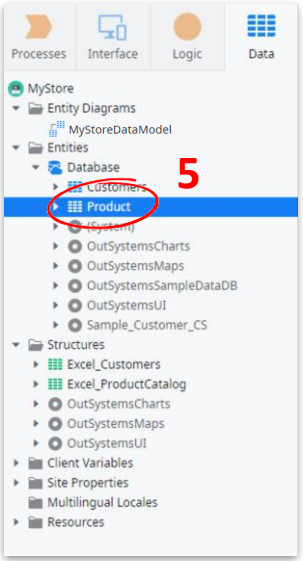
Products and Shipping Company

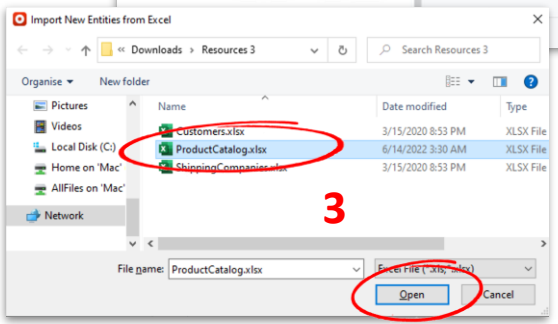
- Another accelerator within the platform is to import Excel files onto the data section. This will automatically create the corresponding entities as well as actions to bootstrap the data itself from the excel file content.

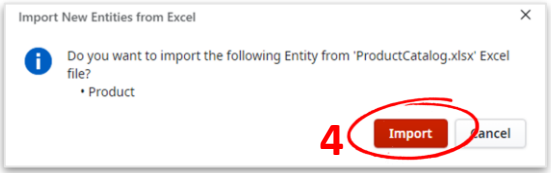
- We will use this together with the screen scaffolding patterns we already know to fast-track parts of our application.

1. Go to the **“Data” tab**;
2. **Right-click on “Database”** and click **“Import New Entities from Excel…”**;
3. Choose the **“ProductCatalog.xlsx”** file;
4. Click “**Import**” when prompted for confirmation;
5. You’ll notice that the **“Product” Entity** is automatically created.



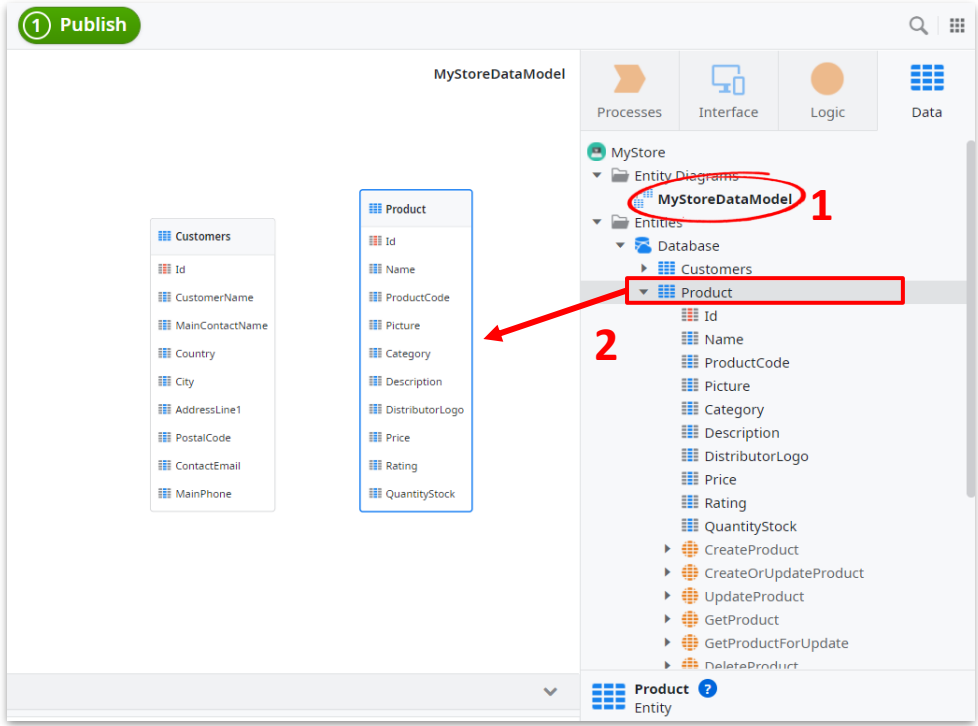






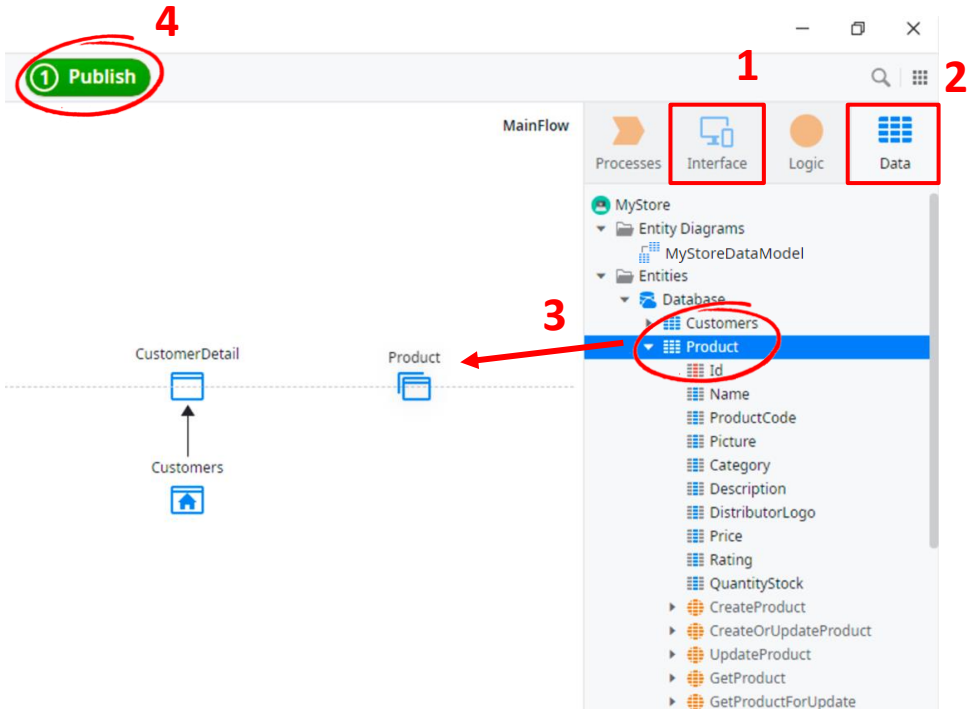
The OutSystems platform provides development **accelerators** to increase productivity on common, patterned and repeating development tasks (**scaffolding**).

1. Open the Entity Diagram “**MyStoreDataModel**”;
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.

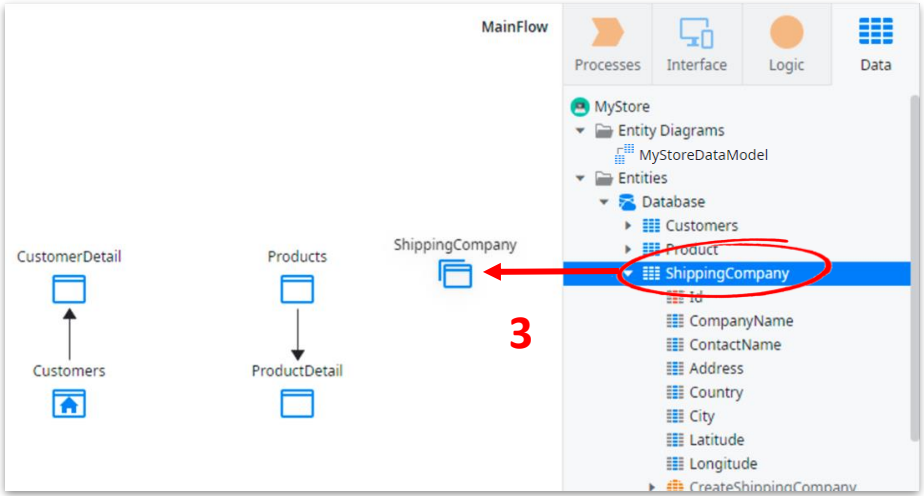
1. Double-click on the "**MainFlow**" on **Interface** menu;
2. Open the "**Data**" tab;
3. Drag the **"Product" Entity** to the "**MainFlow**" canvas to create both pages (list and edit) and tune them with the proper information (i.e. *maybe you want to remove the picture URL from the product list,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

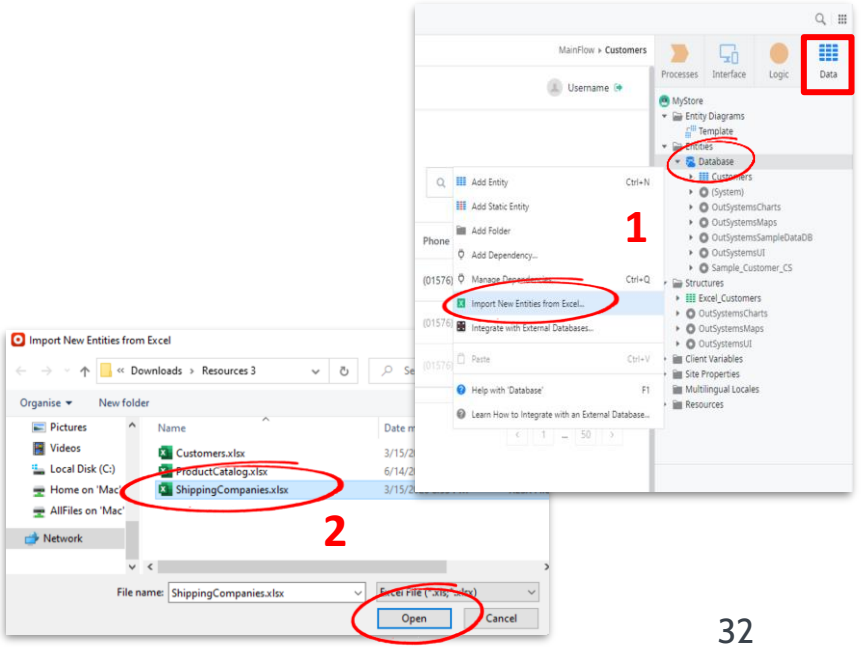


QUICK TIP

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 the “**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”** tab of Service Studio.

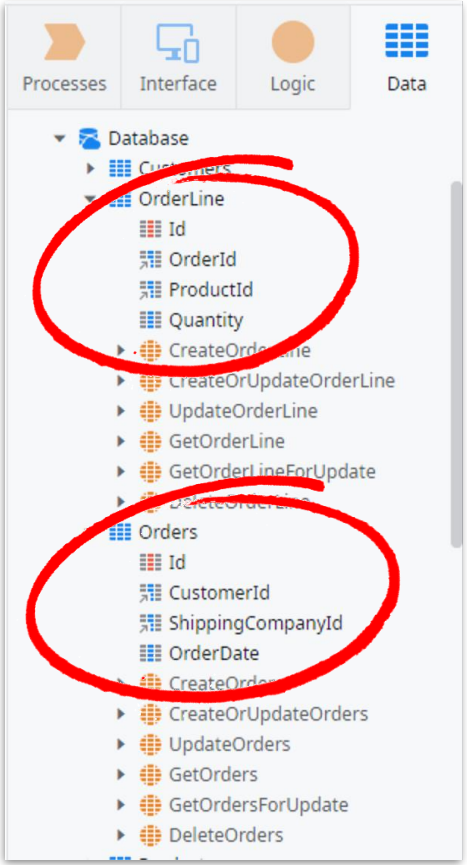
1. Create the “Orders” Entity with the following additional attributes:

Name	Data Type
CustomersId	Customer Identifier
ShippingCompanyId	ShippingCompany Identifier
OrderDate	Date

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

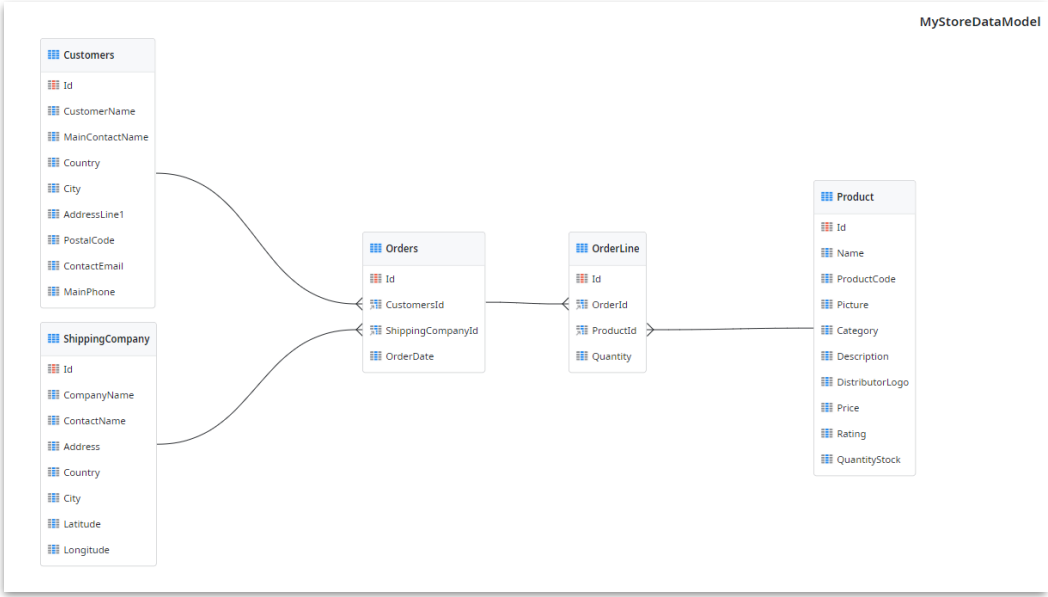
Name	Data Type
OrderId	Order Identifier
ProductId	Product Identifier
Quantity	Integer

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.

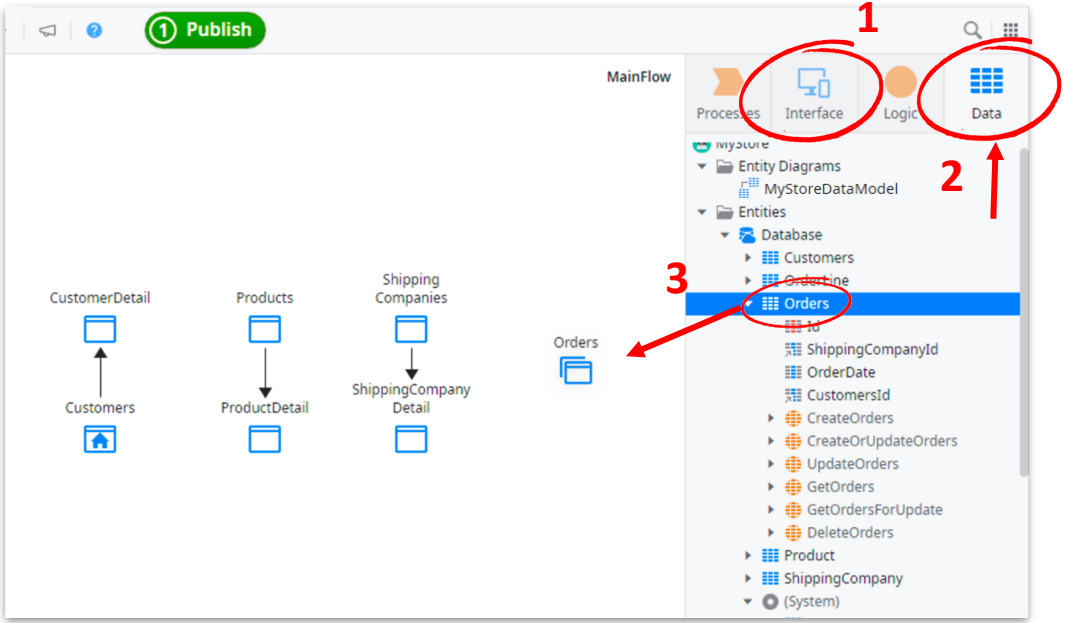


1. Open the **“MyStoreDataModel” Entity Diagram**;
2. Drag the **“Orders” entity** to the “MyStoreDataModel” diagram;
3. Drag the **“OrderLine” entity** to the “MyStoreDataModel” diagram;
4. Drag the **“ShippingCompany” entity** to the “MyStoreDataModel” diagram.

The “MyStoreDataModel” diagram should be similar to the one on the right.

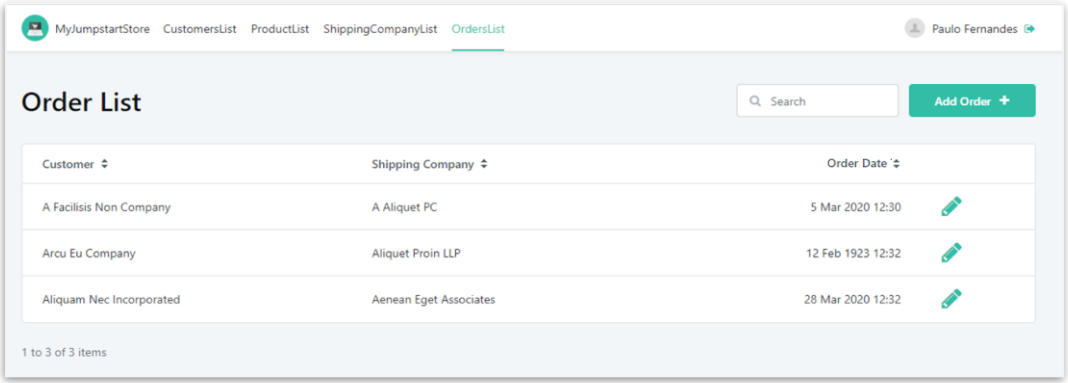


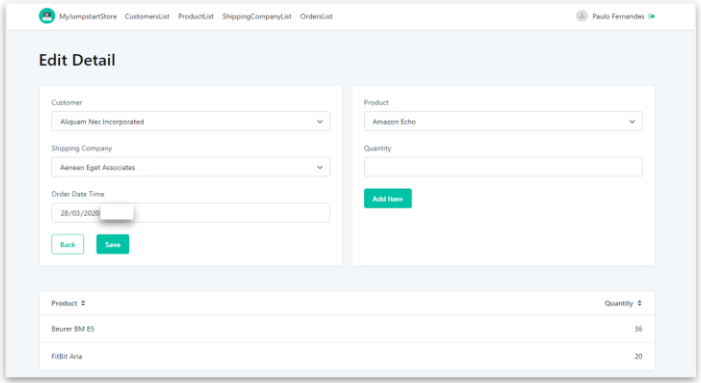
1. Open the **“MainFlow”** on the “**Interface” Tab**;
2. Go to the “**Data” tab**;
3. Drag the **“Orders” entity to the “MainFlow”** to create both pages (list and edit).



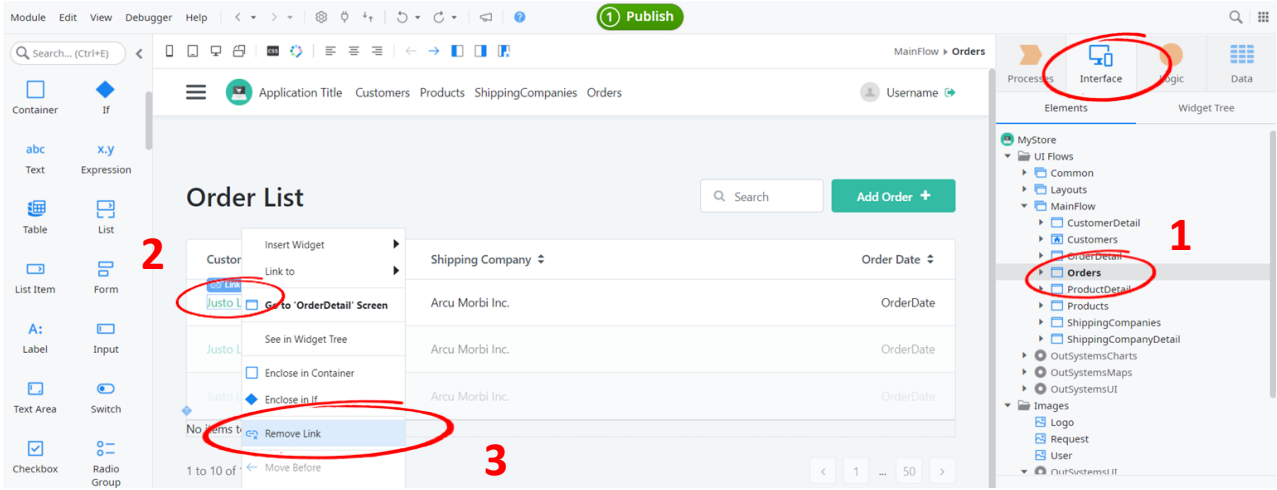
The accelerators are a great way to get a lot of functionality very fast. For simple applications that might be exactly what you need. More often than not you will want to adapt the generated pages to exactly your needs. You can easily do this, again in a visual fashion.

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.



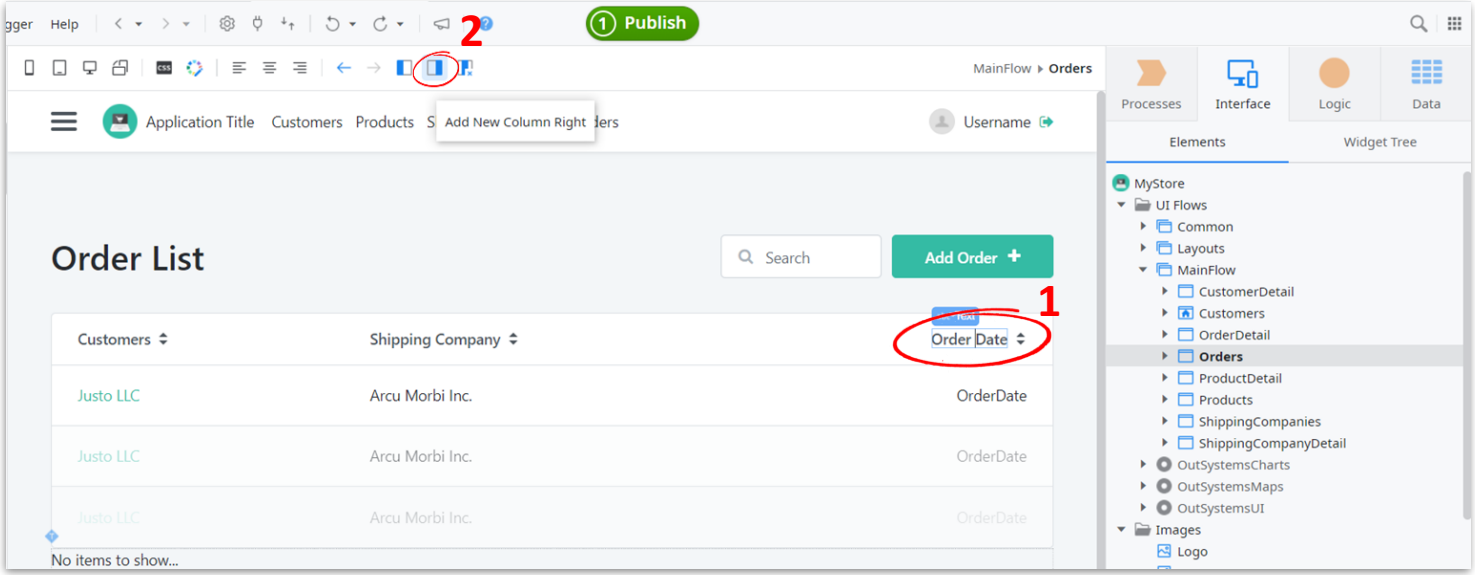


1. Go to the “**Interface” tab** and open the **“Orders”** screen;
2. Select the **first element** of the **"Customer"** column, and right-click the enclosing “**Link”** widget;
3. Click on the **“Remove Link”** option.



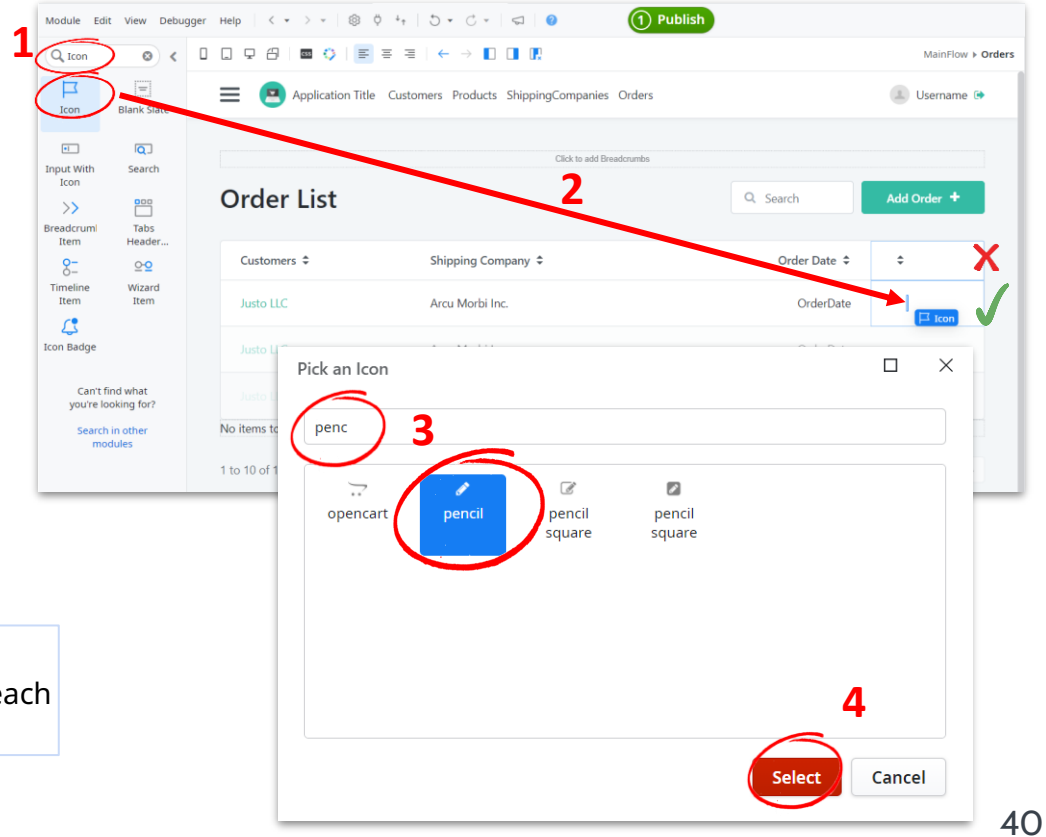
QUICK TIP

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



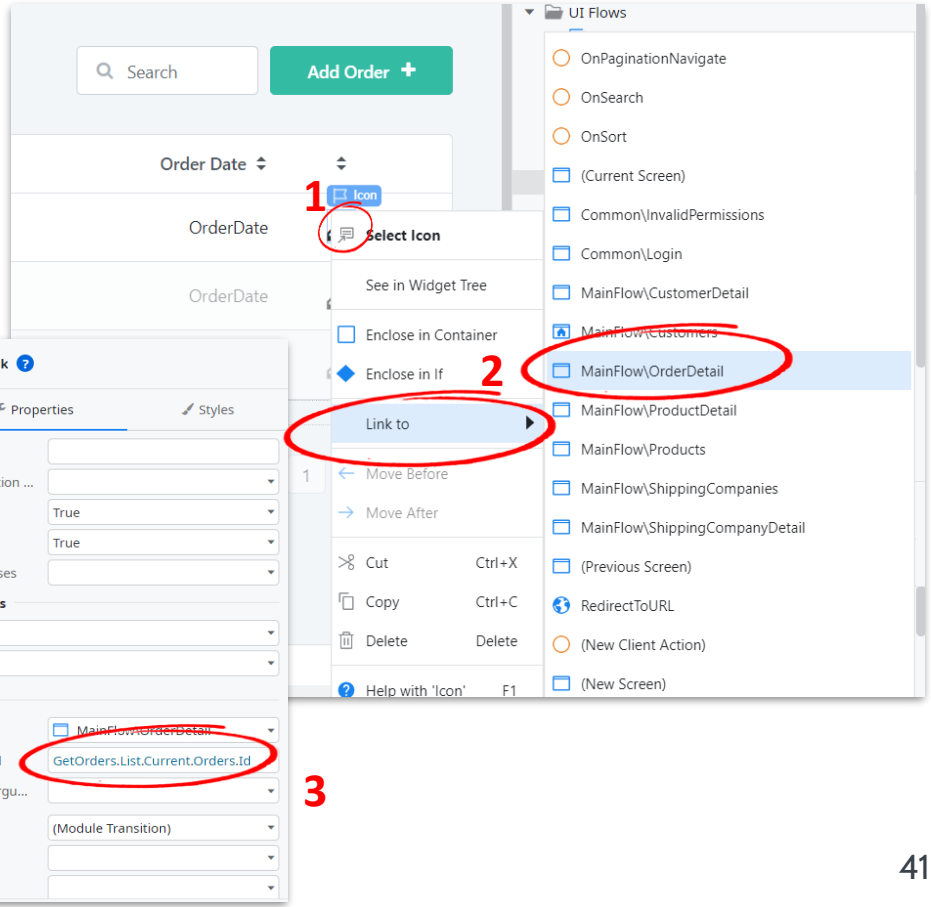
1. Still in the “Orders” screen, click on the header of the **"Order Date"** table column;
2. Click on the **“Add New Column Right”** button to **add a new column** to the right of the Order Date.

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 **“Select”**;

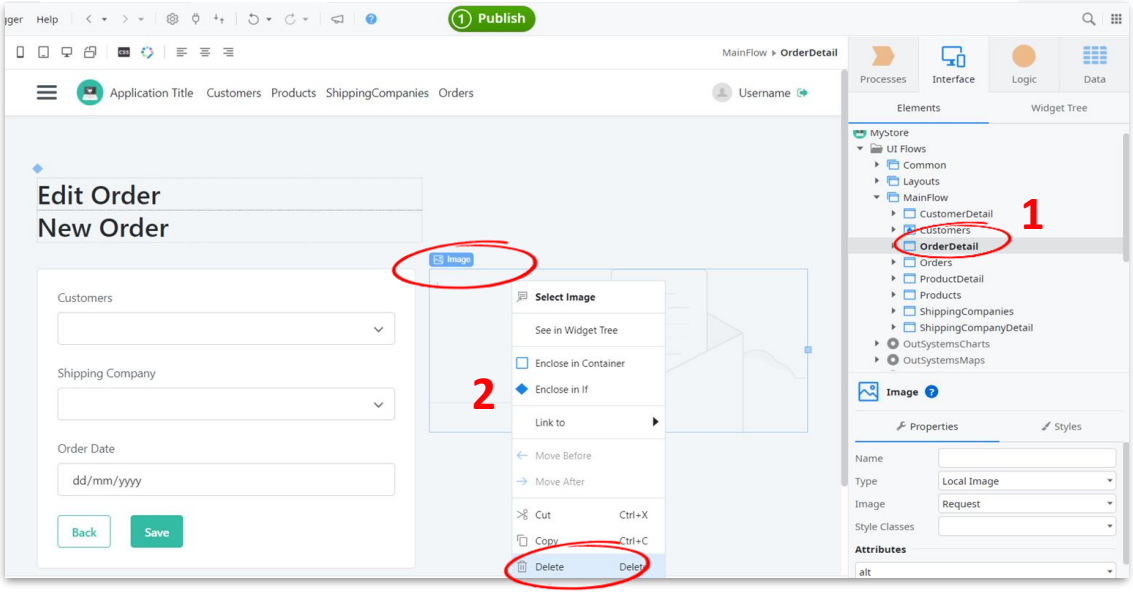


The first row of the table is the Header row where you can enter the header name for each column.

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”**.



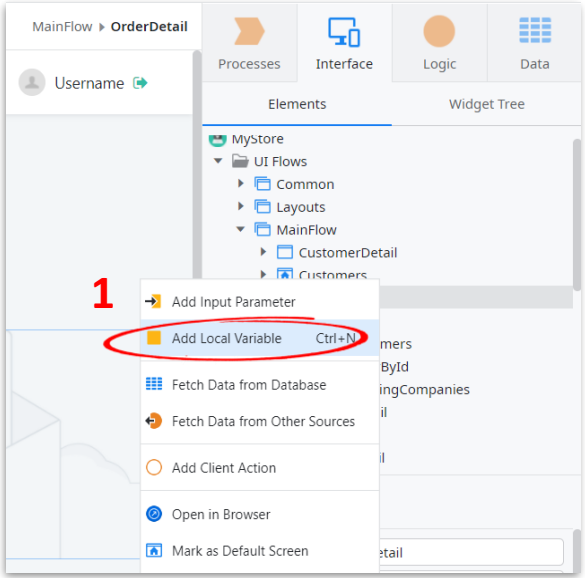
You have just created a link to the “OrderDetail” page. Users can now click on the Pencil icon to open the Order Detail page of a specific order.

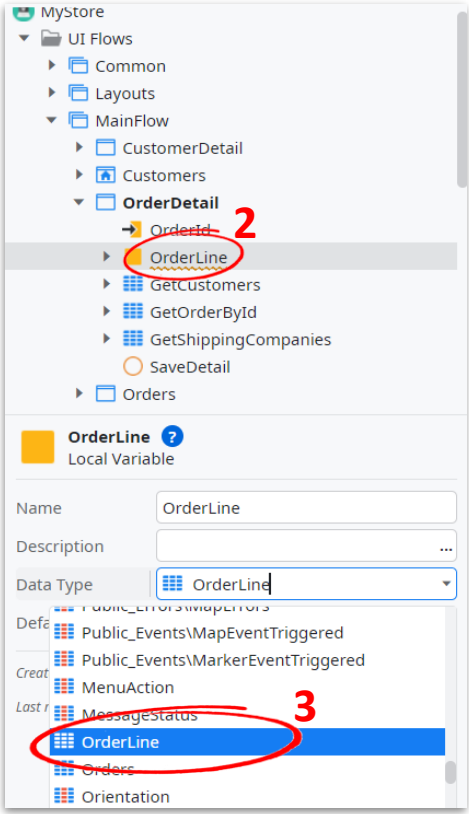


We will now adapt the “OrderDetail” screen to enable users to add new Order Lines to an Order. Let's start this process by deleting the Image that is displayed by default on the screen. To do that:

1. Open the **“OrdersDetail”** screen;
2. **Right-click** on the Image widget and select the “**Delete**” option.

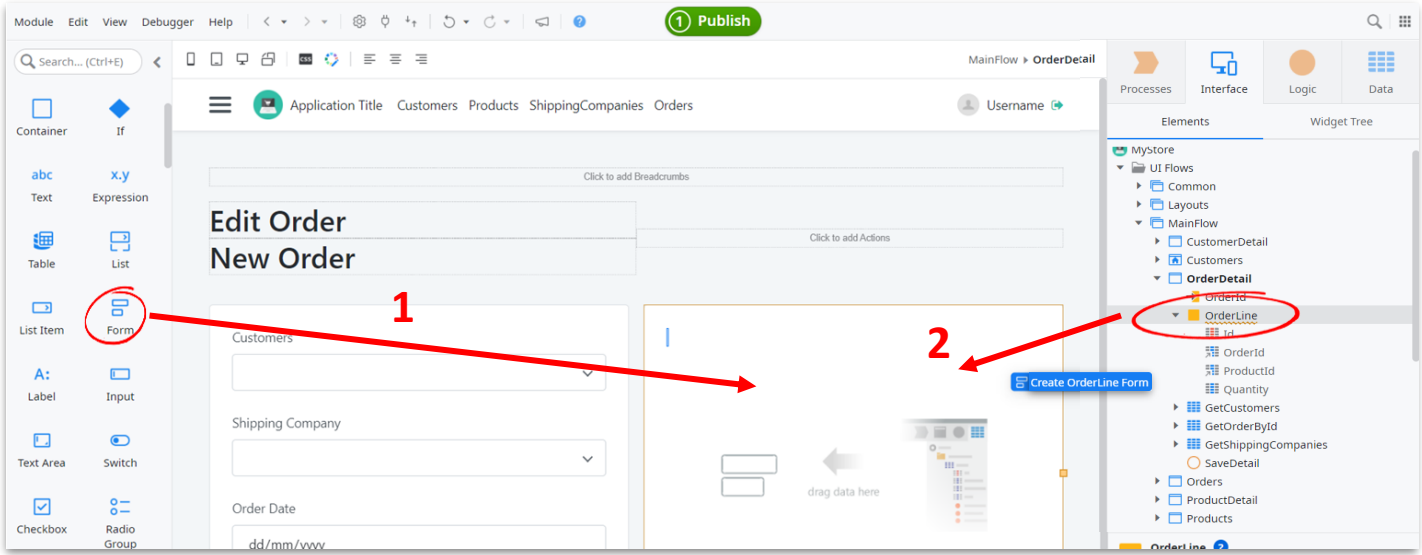
1. **Right-click** on the “OrderDetail” screen and choose the “**Add Local Variable”** option;
2. **Rename** the local variable to “OrderLine”;
3. In the Properties tab, set the “**Data Type” to “OrderLine”**.





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.

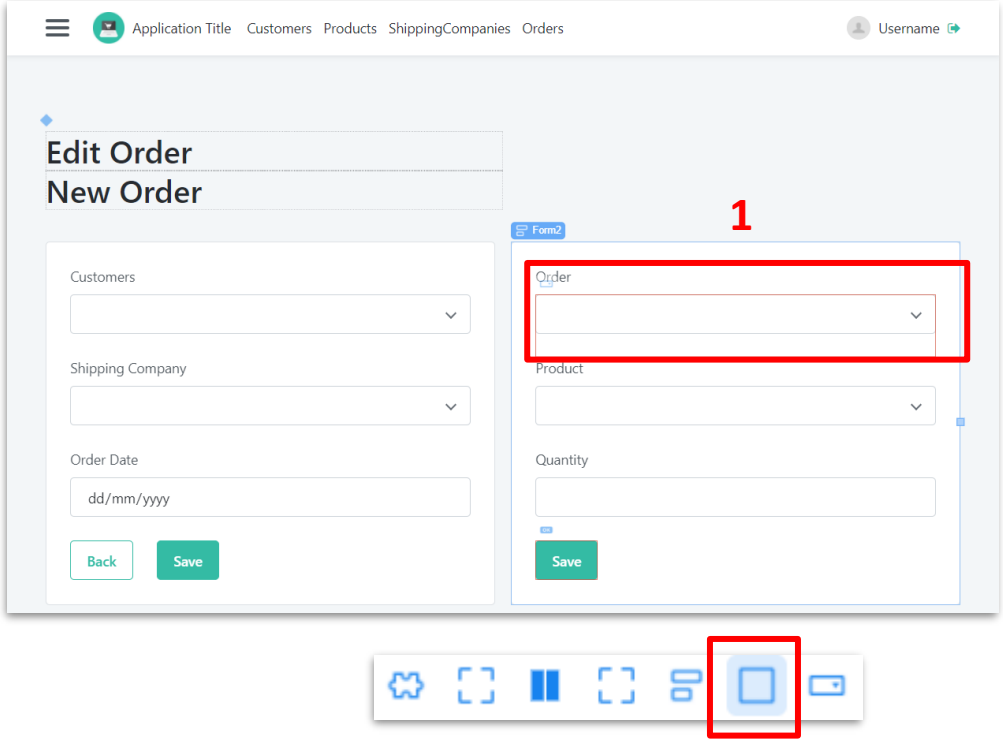


1. **Drag the "Form" widget,** from the toolbox pane in the left, to the column space in the right. This will create a blank form;

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

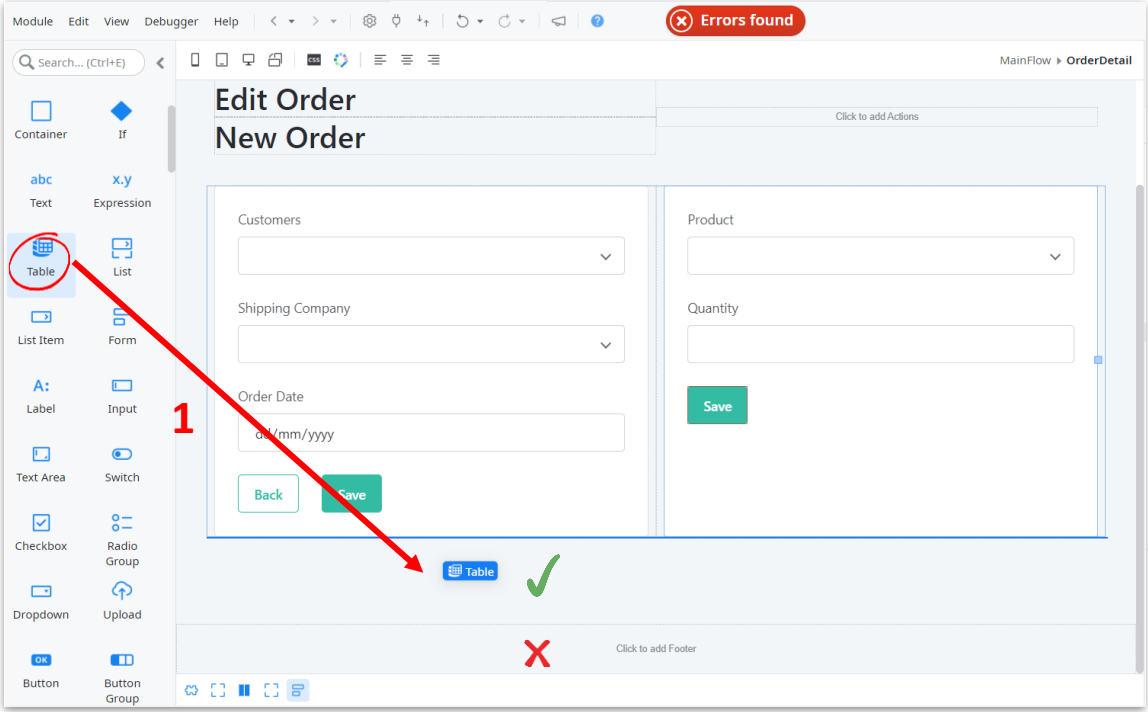
The Order dropdown should not be displayed in this page since we are already in the context of a specific order.

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.

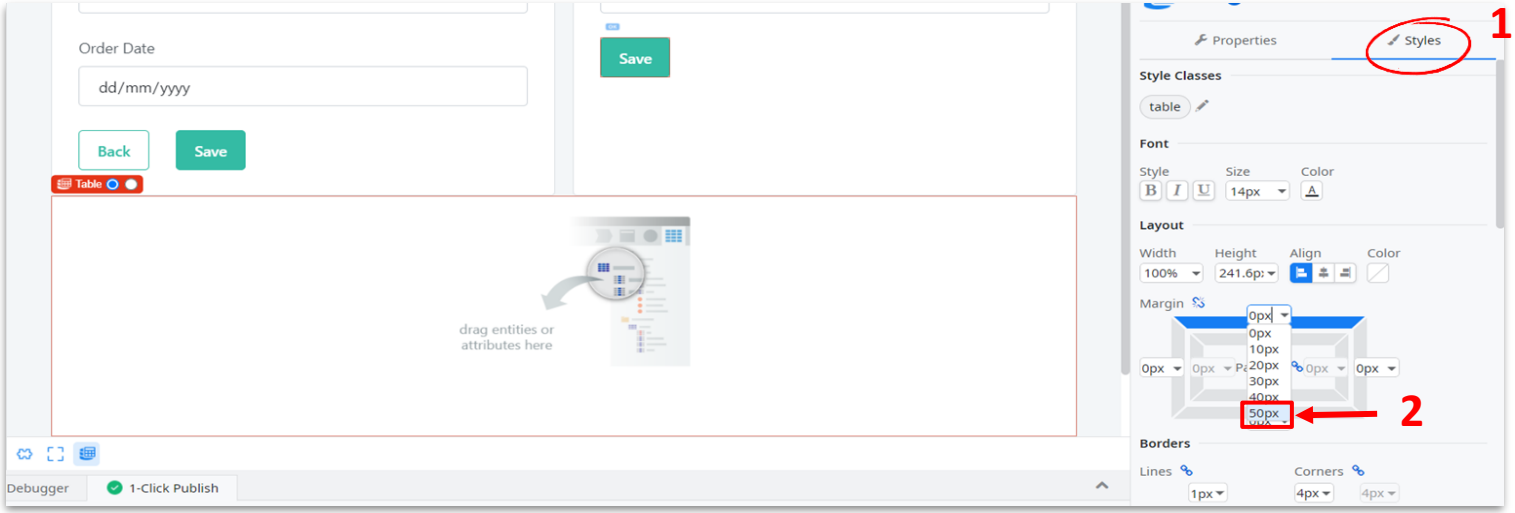
1. **Drag a "Table" component** below the two forms. Make sure you are not dragging it to the Footer of the page.



Add some space between the table and the forms

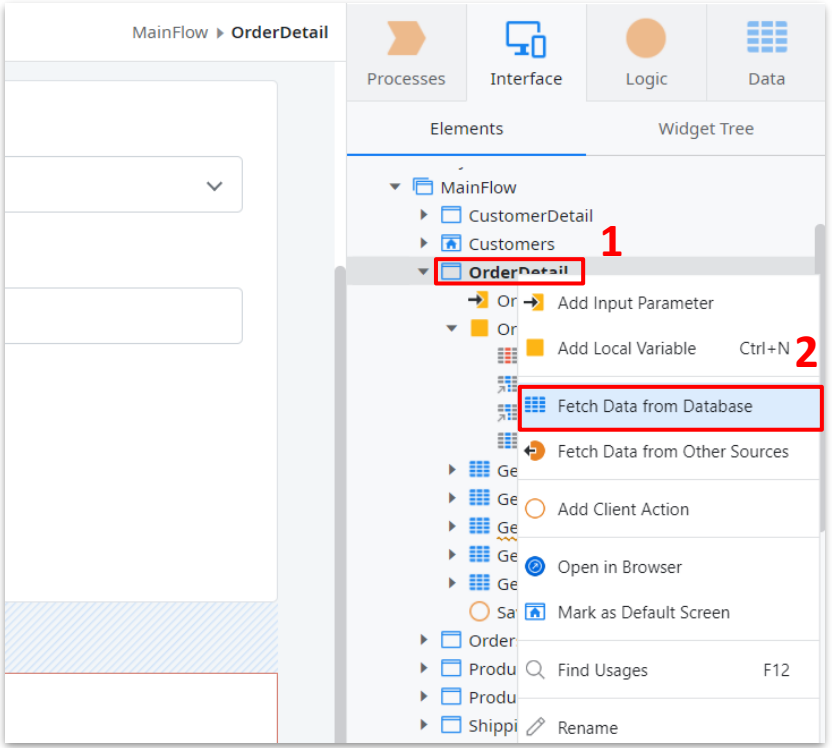
You now have table below the forms. But there's no space in between these elements. Let's add that space:

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



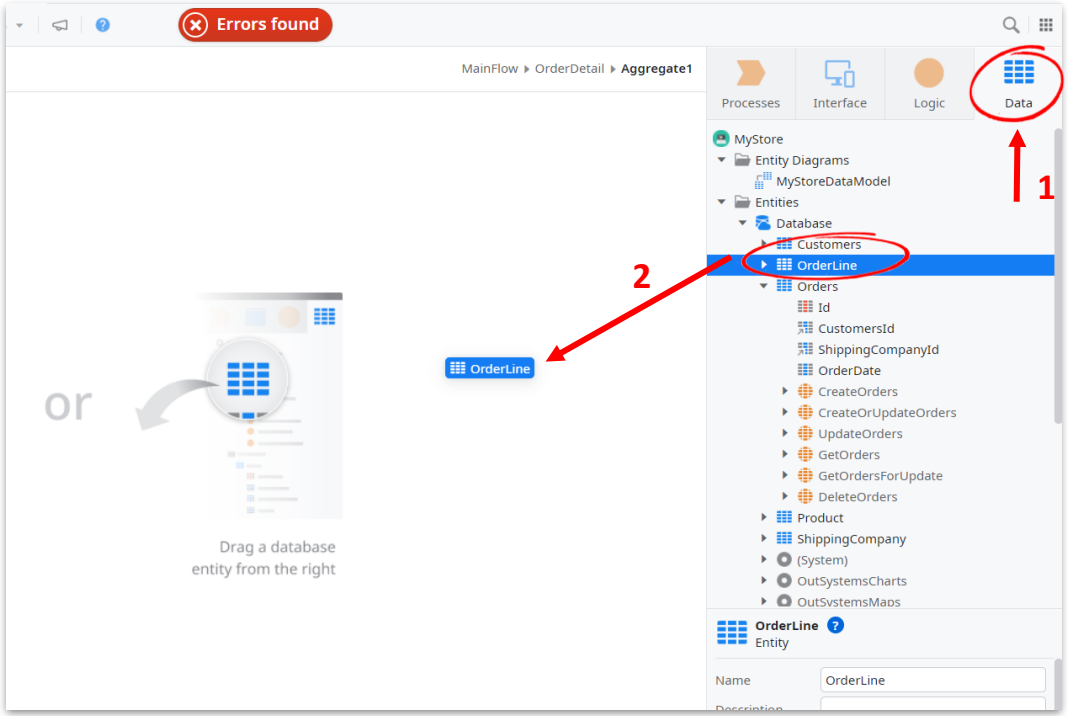
Now that we have added the necessary spacing to the table, lets populate it with order line information from the "OrderLine" Entity. For that:

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



1. Navigate to the **“Data” tab**;

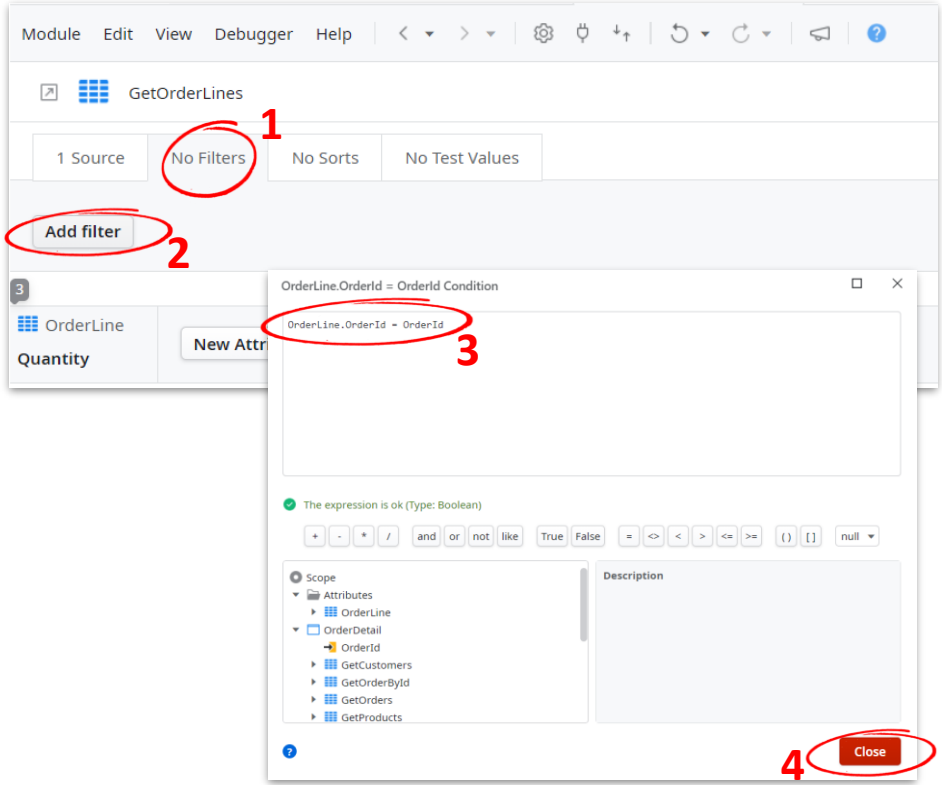
2. **Drag the “OrderLine”** Entity to the new **Aggregate** to fetch information from the “OrderLine” Entity.



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

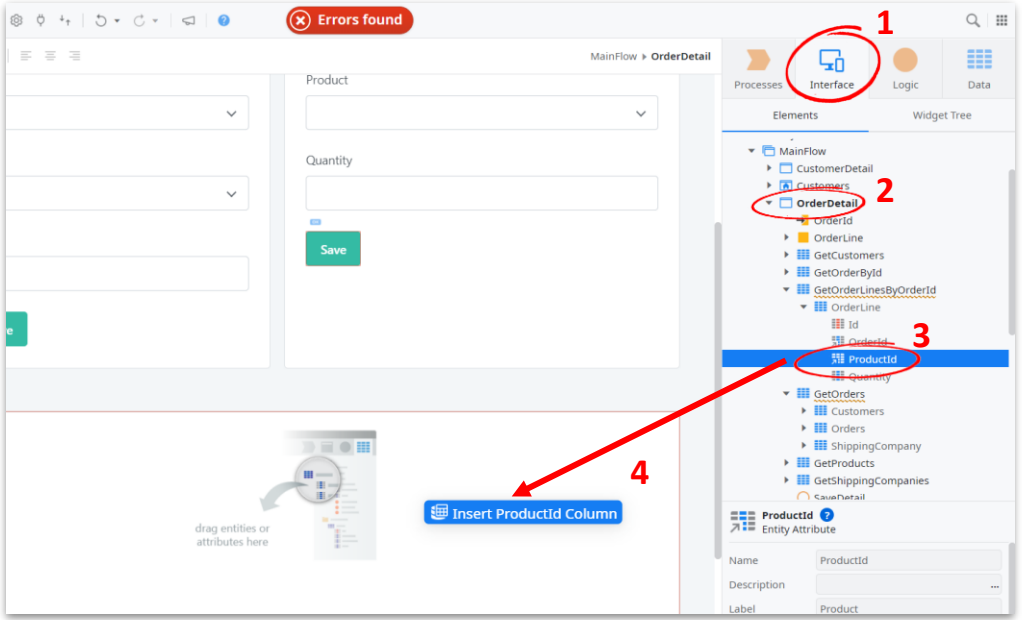
An aggregate allows you to fetch data from the DB using an optimized query.

1. Click on the **“No Filters” tab**;
2. Click “**Add filter**”;
3. Enter the following query:
   “*Orderline.OrderId = OrderId*”;
4. Click **“Close”**.

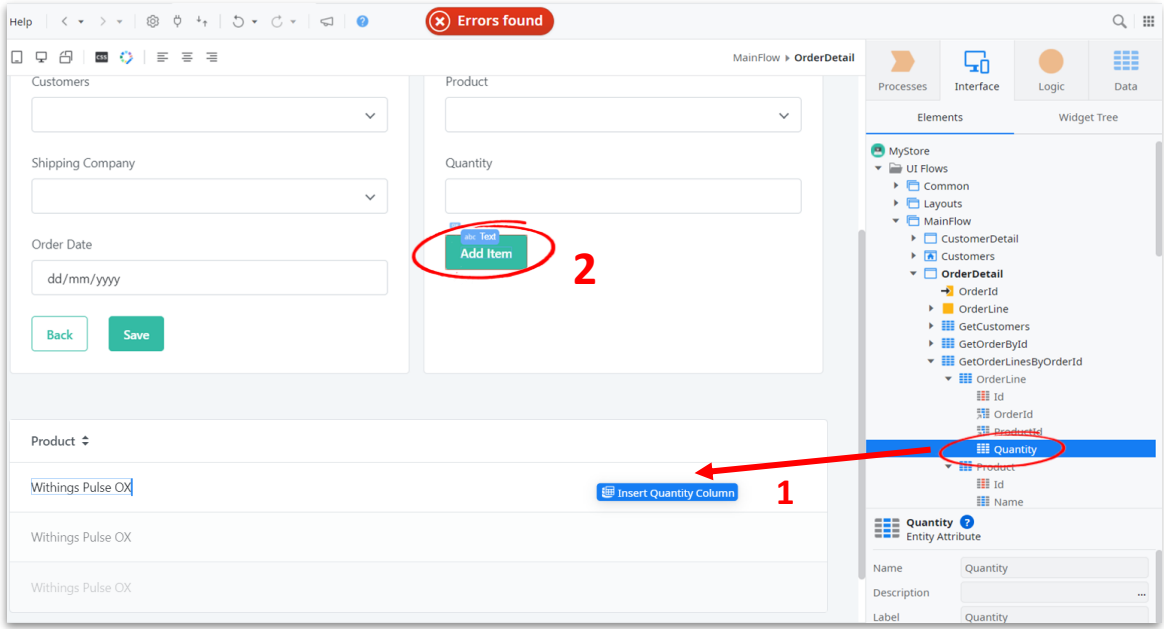


When designing queries, it is common to add conditions to filter results and get exactly what you want from the database. In OutSystems, this is done by adding Filters to Aggregates, as described above.

1. Navigate to the **“Interface” tab**;
2. Double click the **“OrderDetail”** screen to open it;
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;

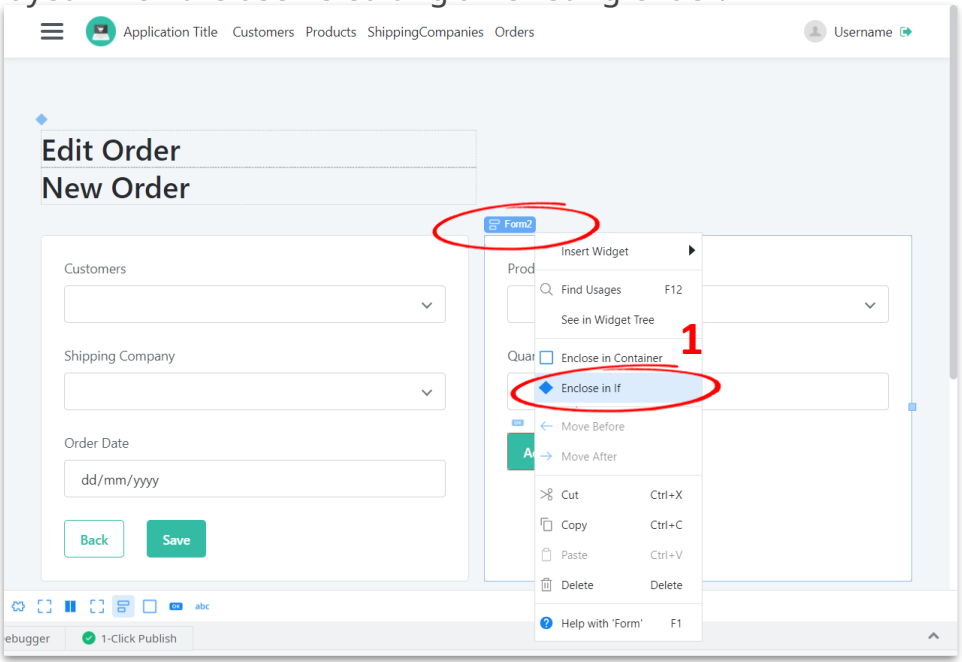


1. Drag the **“Quantity”** attribute to the right side of the **“OrderLines” table**;
2. Then, rename the **“Save”** button to **“Add Item”**.



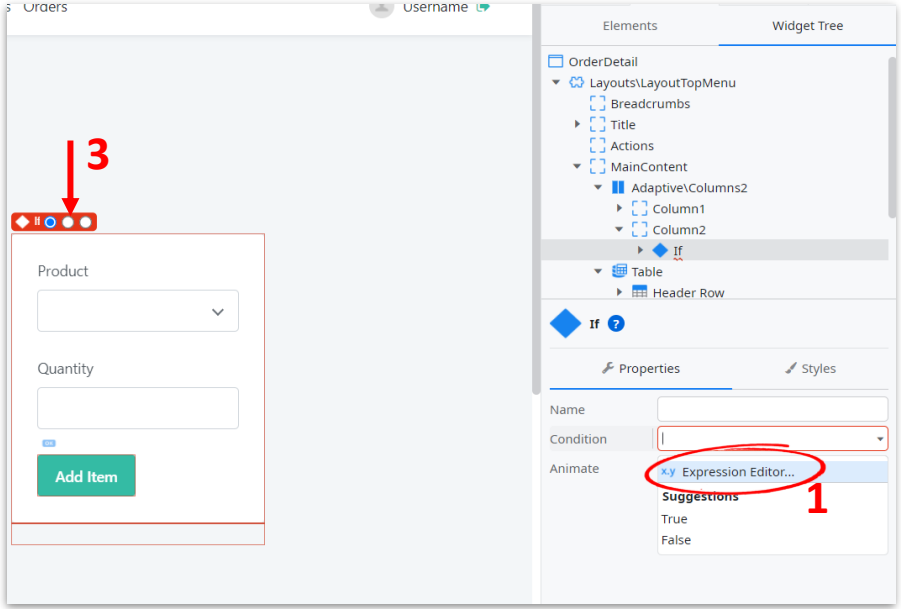
The “OrderDetail” screen is used to edit but also to create Orders. We only want users to be able to add items (new Order Lines) to an order once this Order is created. We will use the “**If” widget** to make sure that the Form (used to add items) and the Order Lines table are only displayed when the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

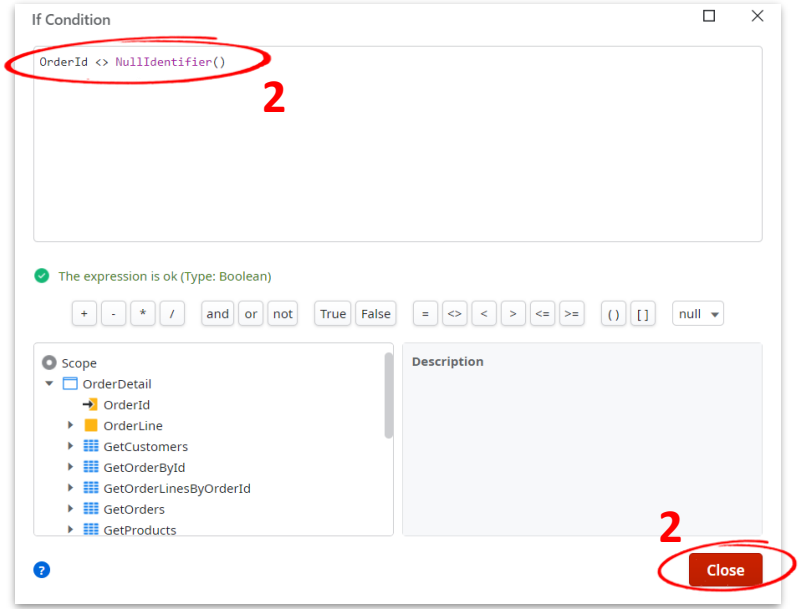
1. **Right click** on the form in the right (Form2) > **“Enclose in If”**.



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.



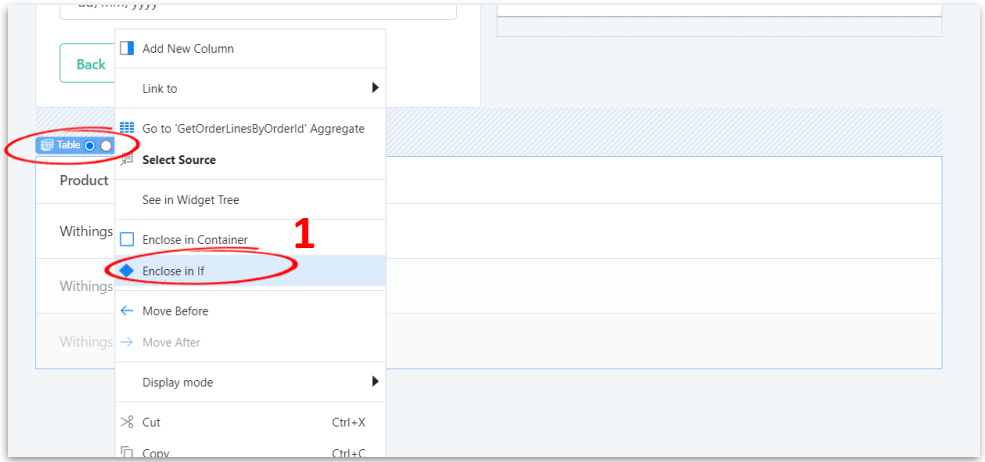


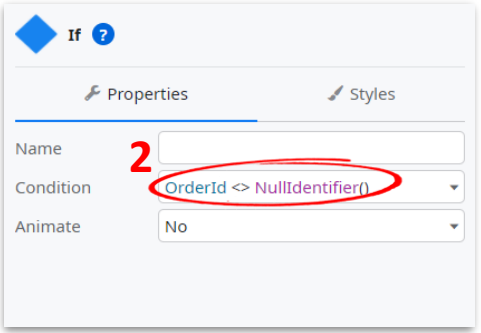
1. In the "**If" widget** Properties tab, open the condition and click on **"Expression Editor…"**;
2. In the expression editor input the expression "*OrderId* <> *NullIdentifier()*" and press **"Close"**;
3. Select the "Display Mode" of the "If" widget, in the canvas, to **"Show True branch only"** (second radio button).

QUICK TIP You can also double-click on "Condition" to quickly open the Expression Editor

Repeat the same steps for the “OrderLines” Table:

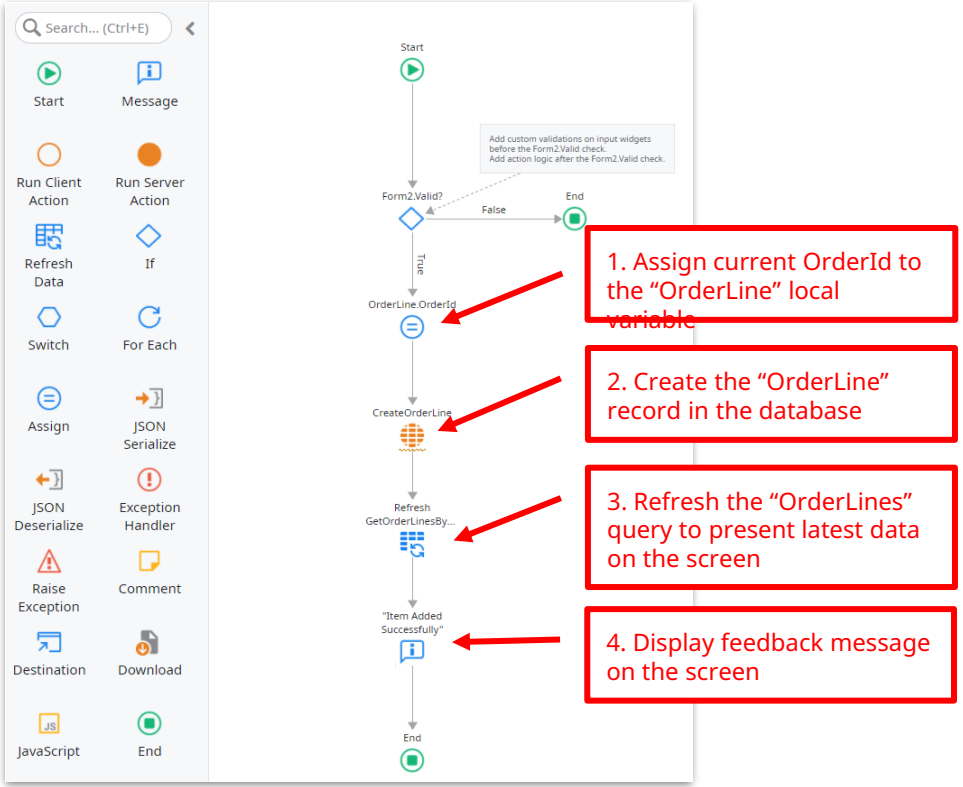
1. Right click on the Order Line table and then **“Enclose in If”**;
2. In the Properties tab, place the cursor in the “Condition” field and paste the expression: “*OrderId <> NullIdentifier()*”.





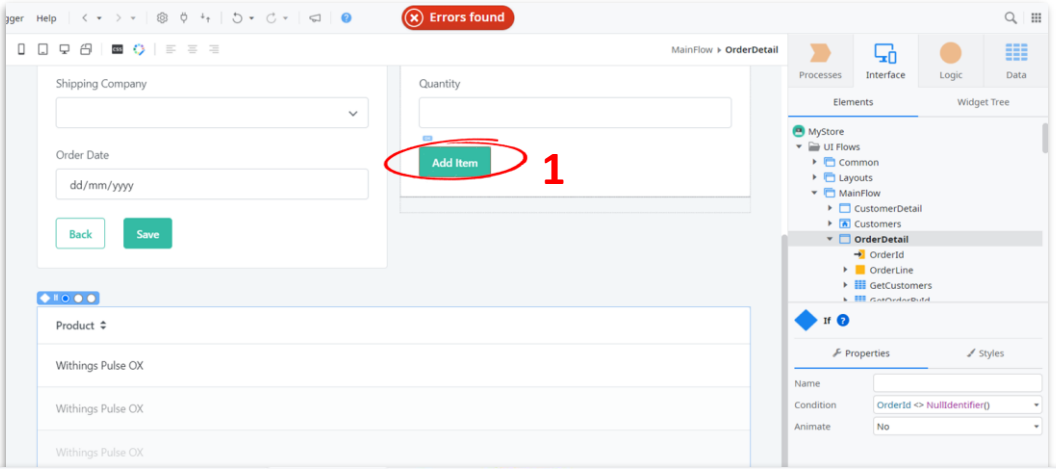
D. Create Logic to add Order Lines to an Order

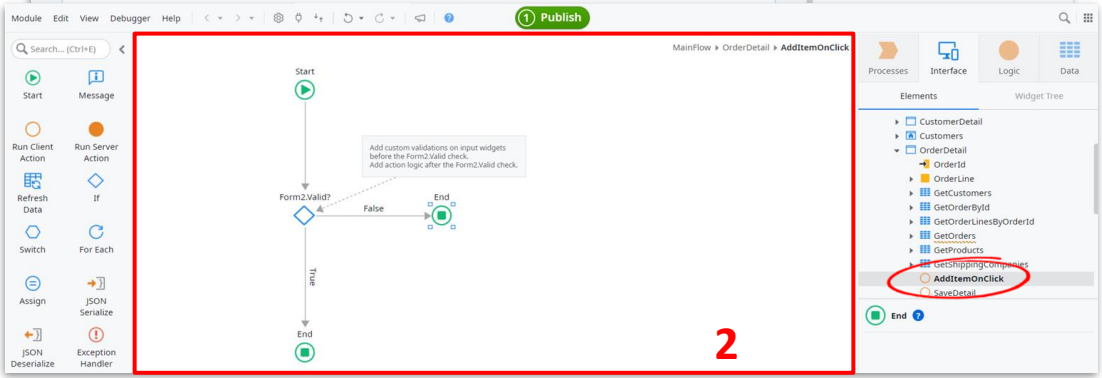
- Now that you have finished adapting the look and feel of the Order List and Detail pages lets work on the logic to add new items (order lines) to an existing Order.
- In the following steps you will create the logic of the “Add Item” button.
- In the end of this section your action flow should look like the one in the image on the right.



Let's create the "Add Item" Button logic

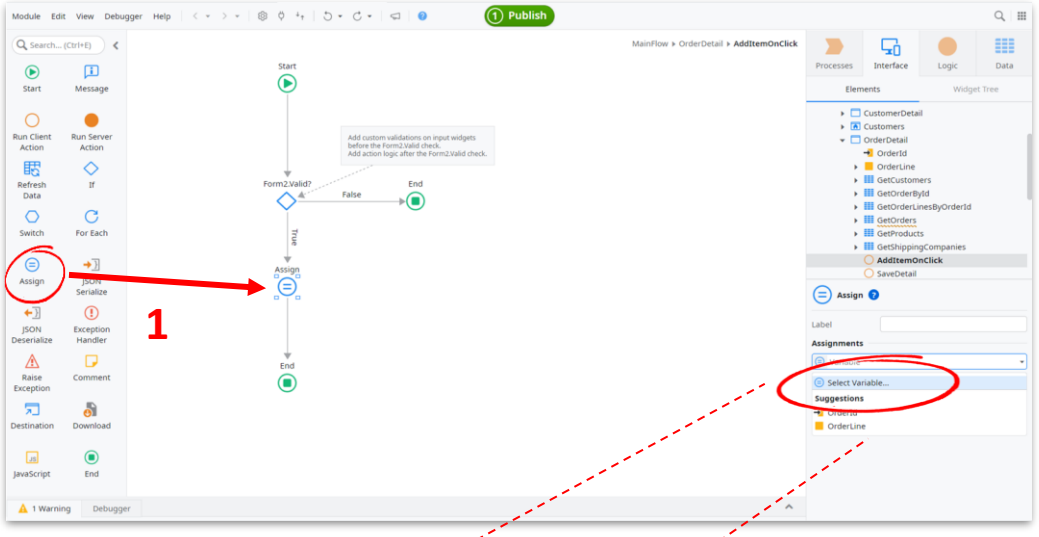
1. **Double-click** the "Add Item" button to create a **new Client Action.**
   Make sure to double-click the Button widget, and not the Text widget inside it;
2. Notice that a new **"AddItemOnClick" Client Action** is created and automatically opened so that we can start working on its logic.

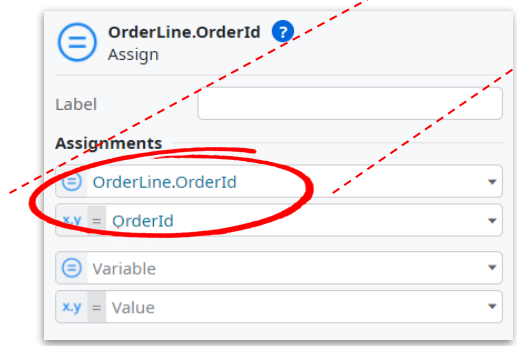




When developing apps you create both logic that runs on the **server**, and logic that runs on the **client** device - just like on mobile apps.

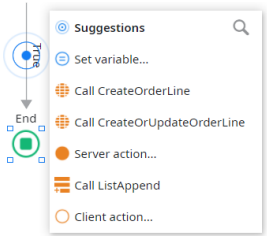
1. Drag an **"Assign"** node to the "AddItemOnClick" action flow;
2. In the properties of the assign node, set the Variable of the assign note to **OrderLine.OrderId** and the Value to **OrderId**.



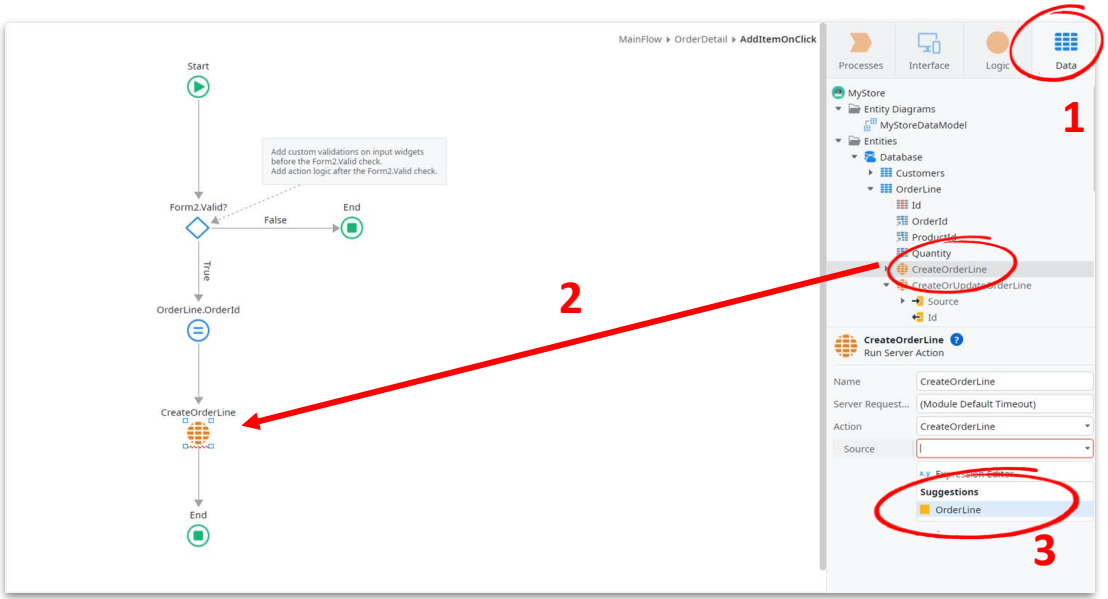


QUICK TIP

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

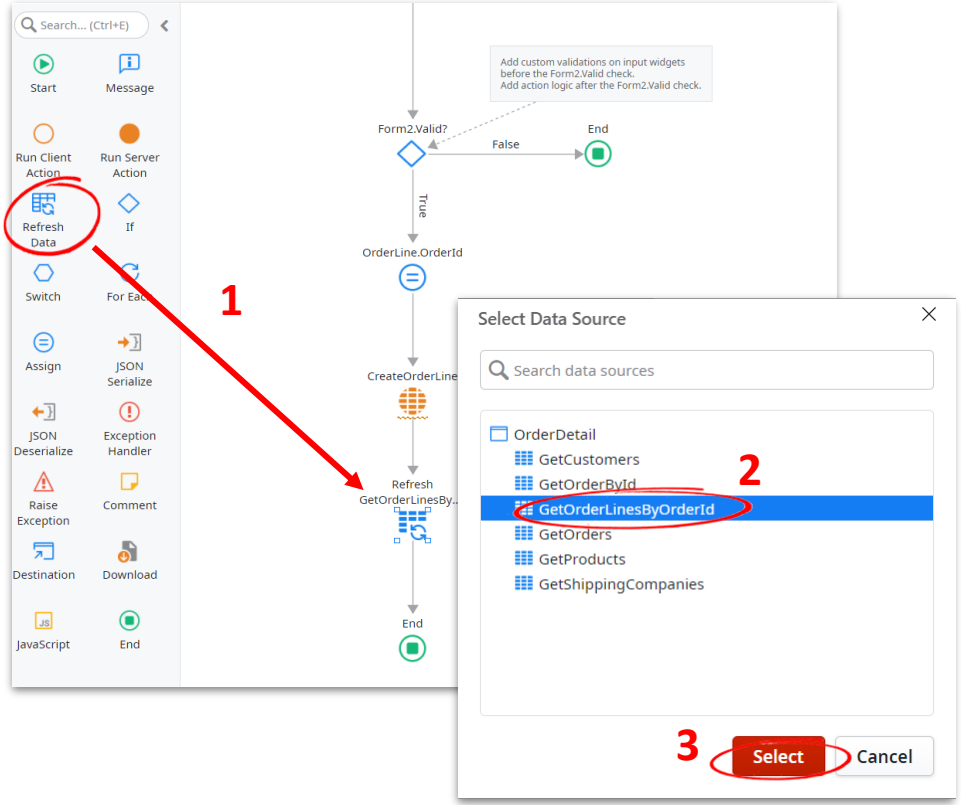


1. Navigate to the **“Data” tab**;
2. Drag the **“CreateOrderLine” Server Action** from the **“OrderLine” Entity** below the assign activity that you have just created;
3. Set the source value to the “OrderLine” attribute.

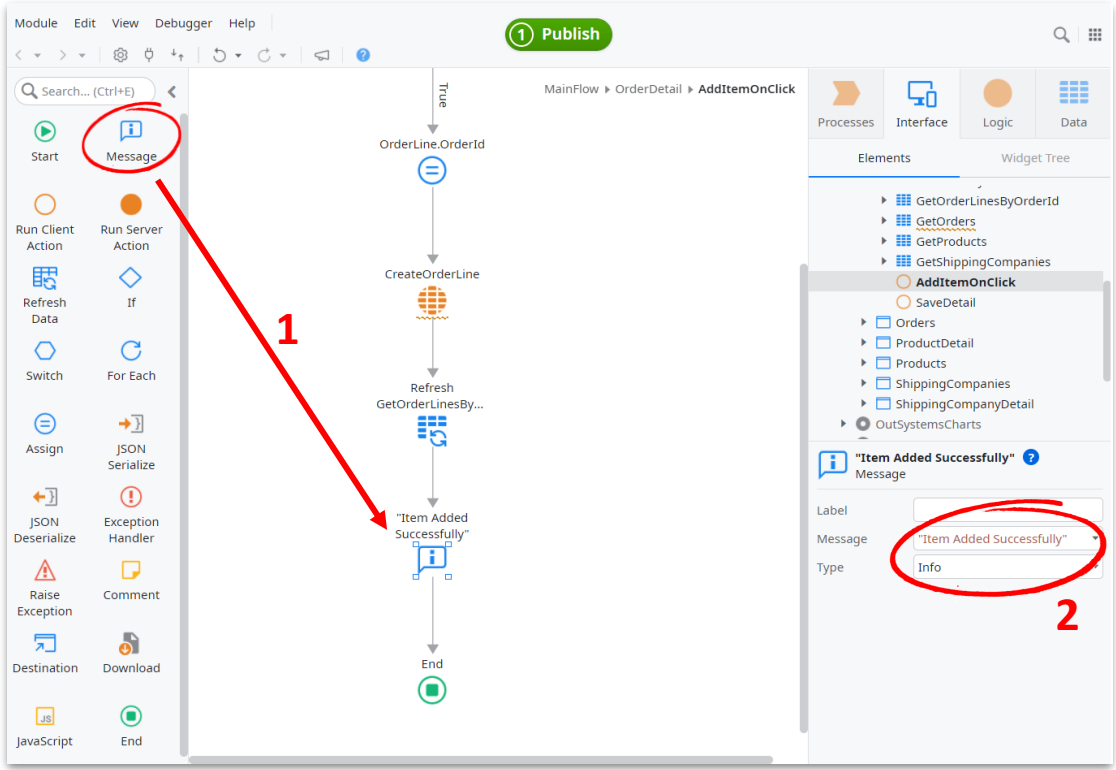


After creating the “OrderLine” in the database, we need to refresh the “OrderLines” table in the screen to fetch the latest records from the database.

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

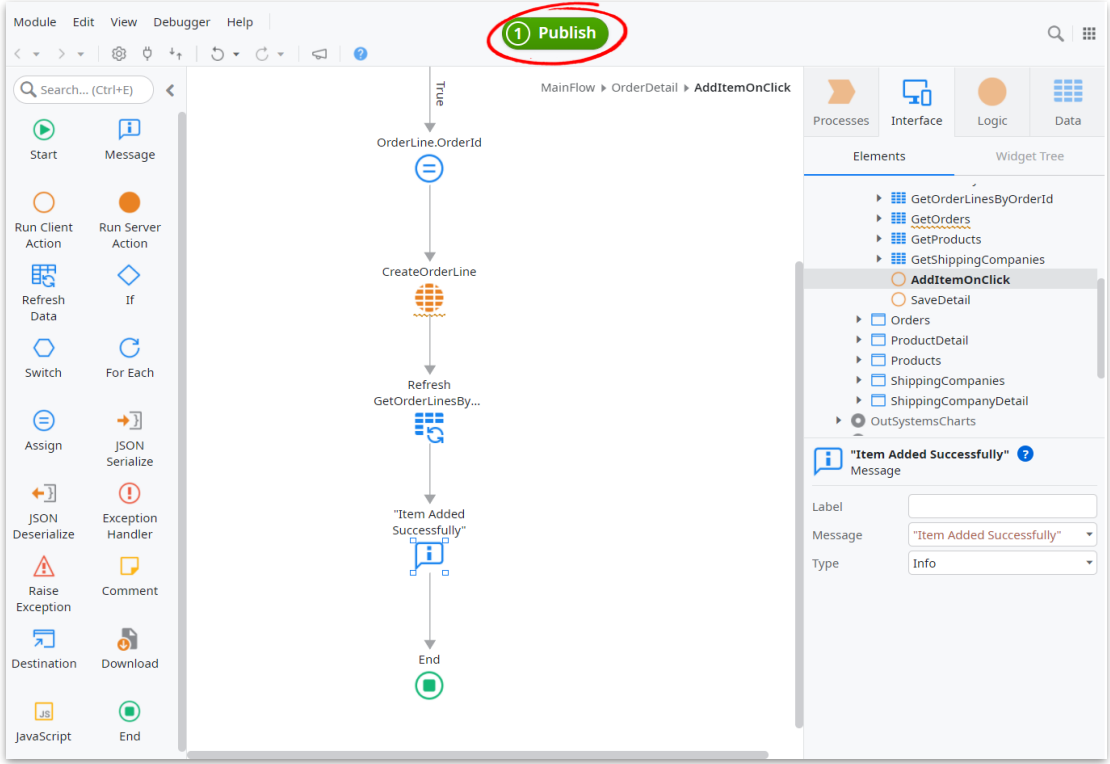


1. Add the **“Message” widget** to the Action Flow just below the “Refresh Data”;
2. Go to the properties panel: for the Message write **“Item Added Successfully”** for the type choose the option **“Success”**.



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

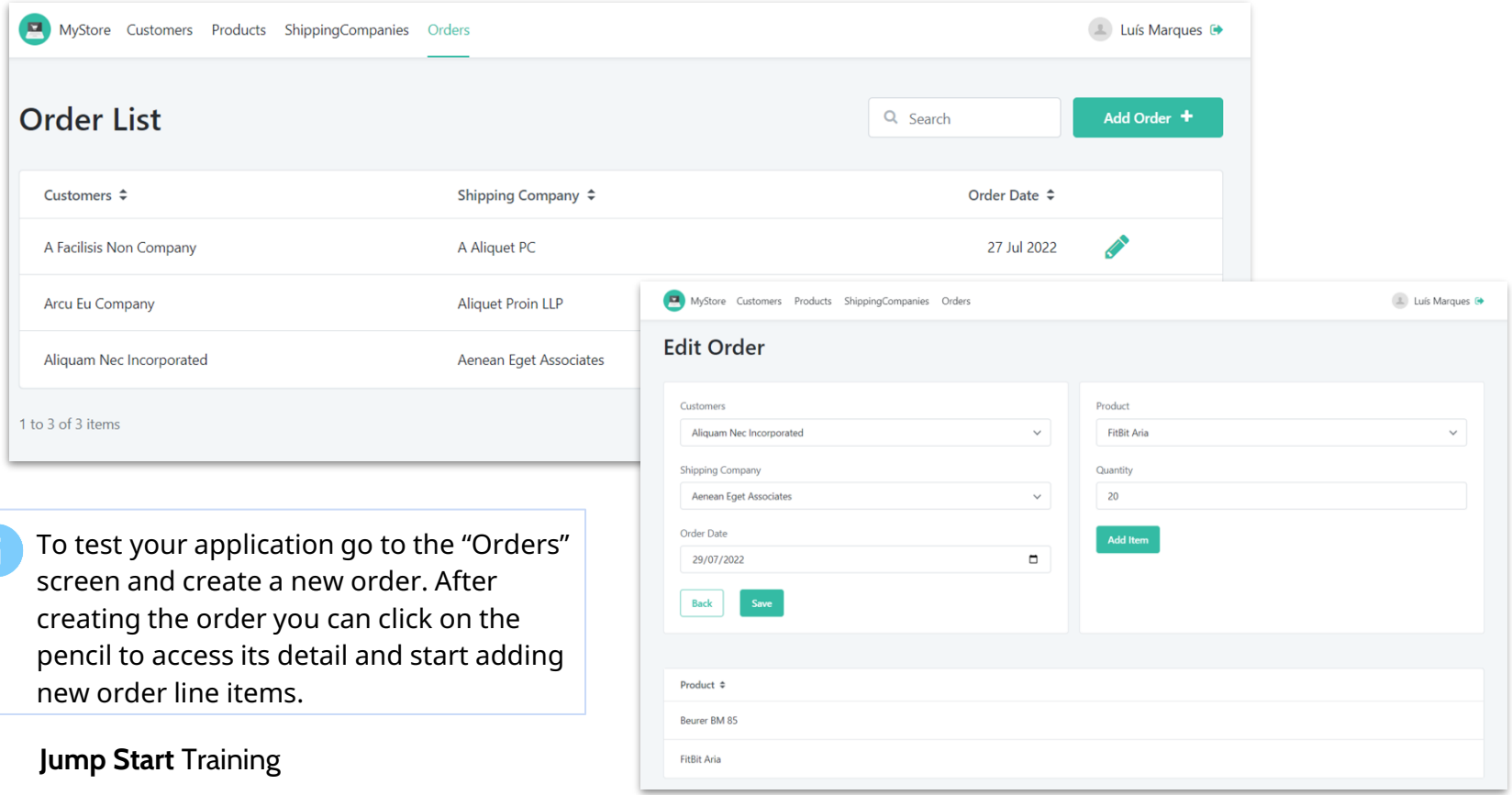
1. Click the "**1-Click publish**" to initiate the deployment process.



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 "Orders" screen 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.

Section 3

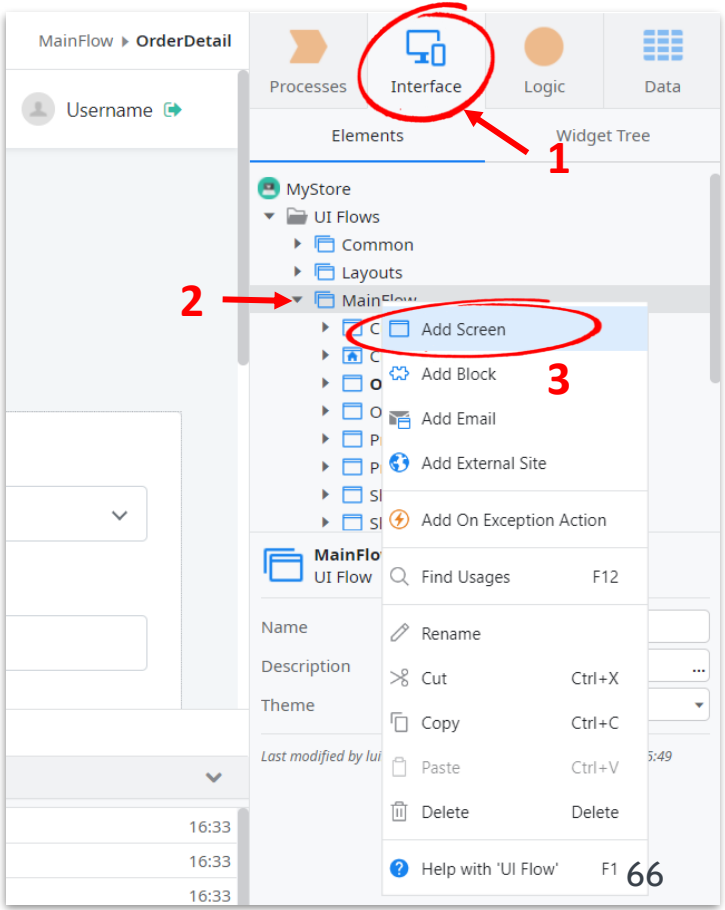
Image Gallery with fluid layout

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.

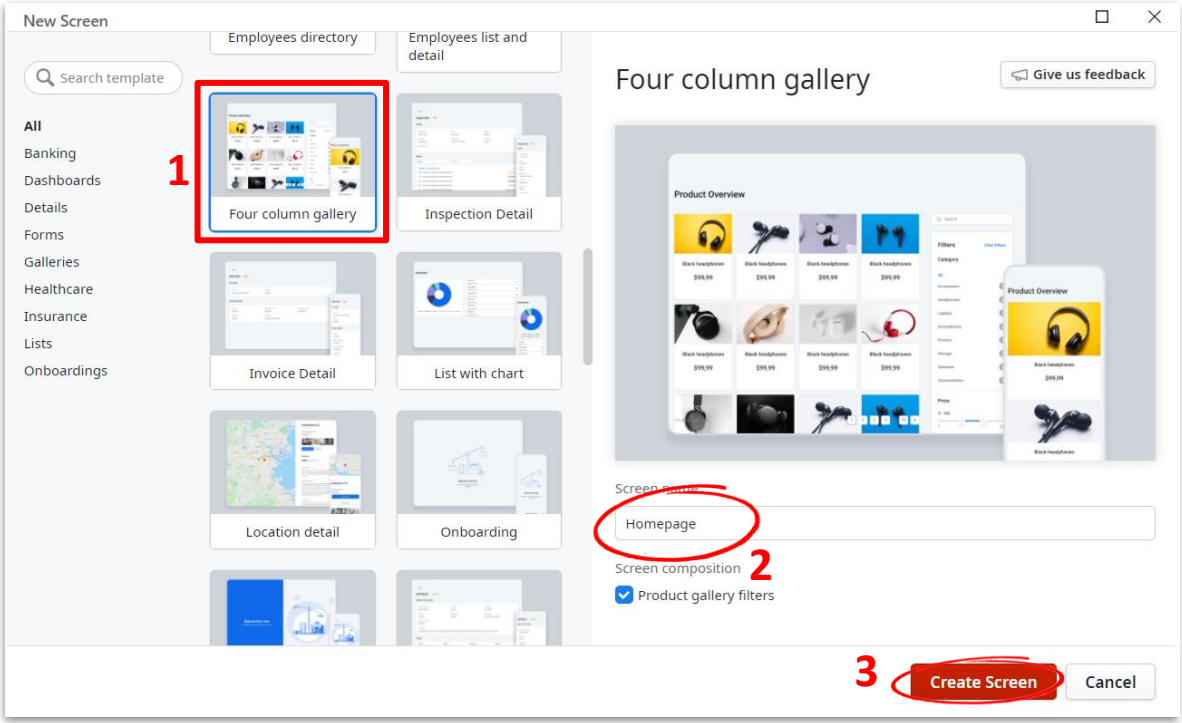
OutSystems UI templates pre-creates your screens with sample data that can be easily replaced so you don’t have to start from scratch.

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

1. Go to the **Interface tab**;
2. Right-click on "**MainFlow**";
3. Select "**Add Screen**" to create a new web screen.

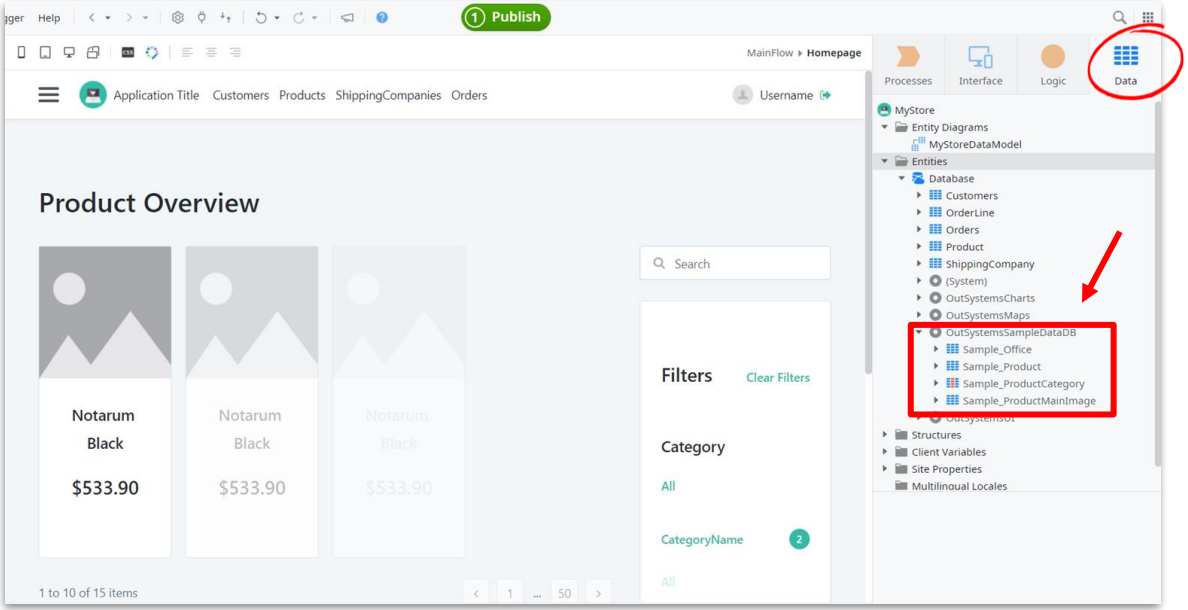


1. Select the “**Four Column Gallery**” template;
2. Add the title “**Homepage**” under Screen name;
3. Click **“Create Screen”**.



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

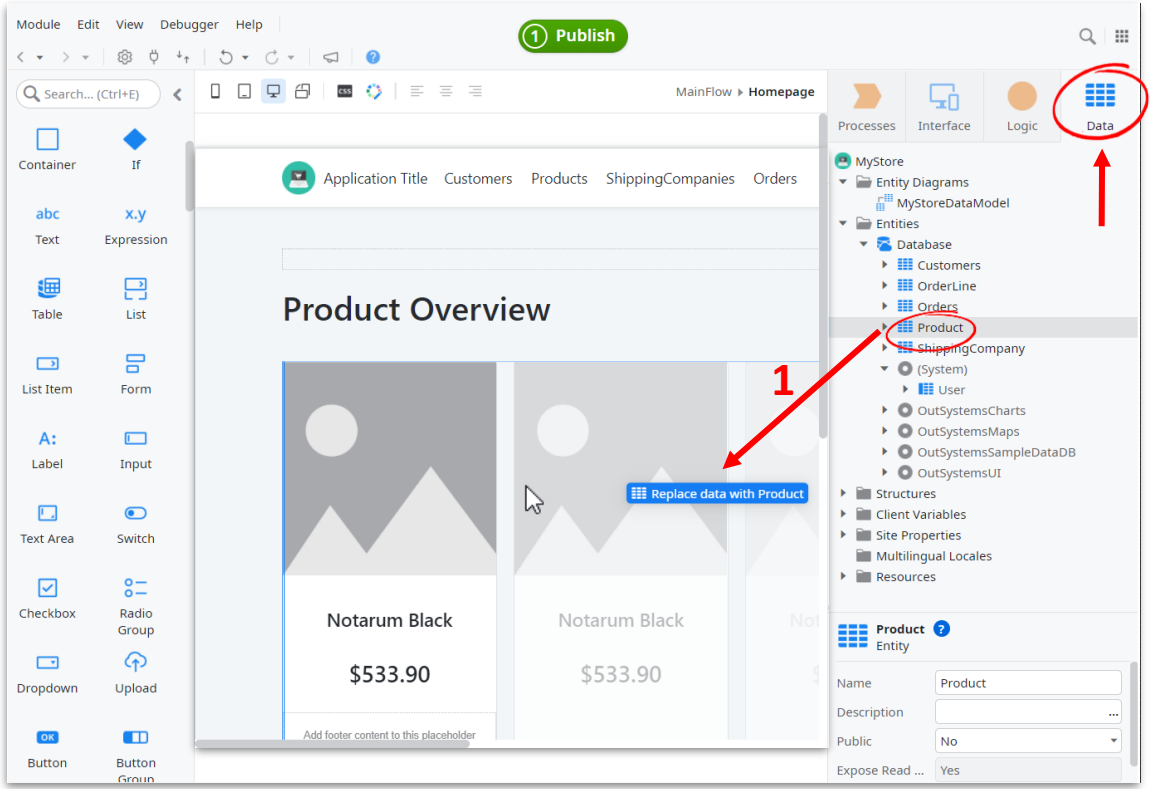
   Notice that it's using Sample Data **"Sample_Office"**, **"Sample_Product"**, **"Sample_ProductCategory"** and **"Sample_ProductMainImage"**.



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

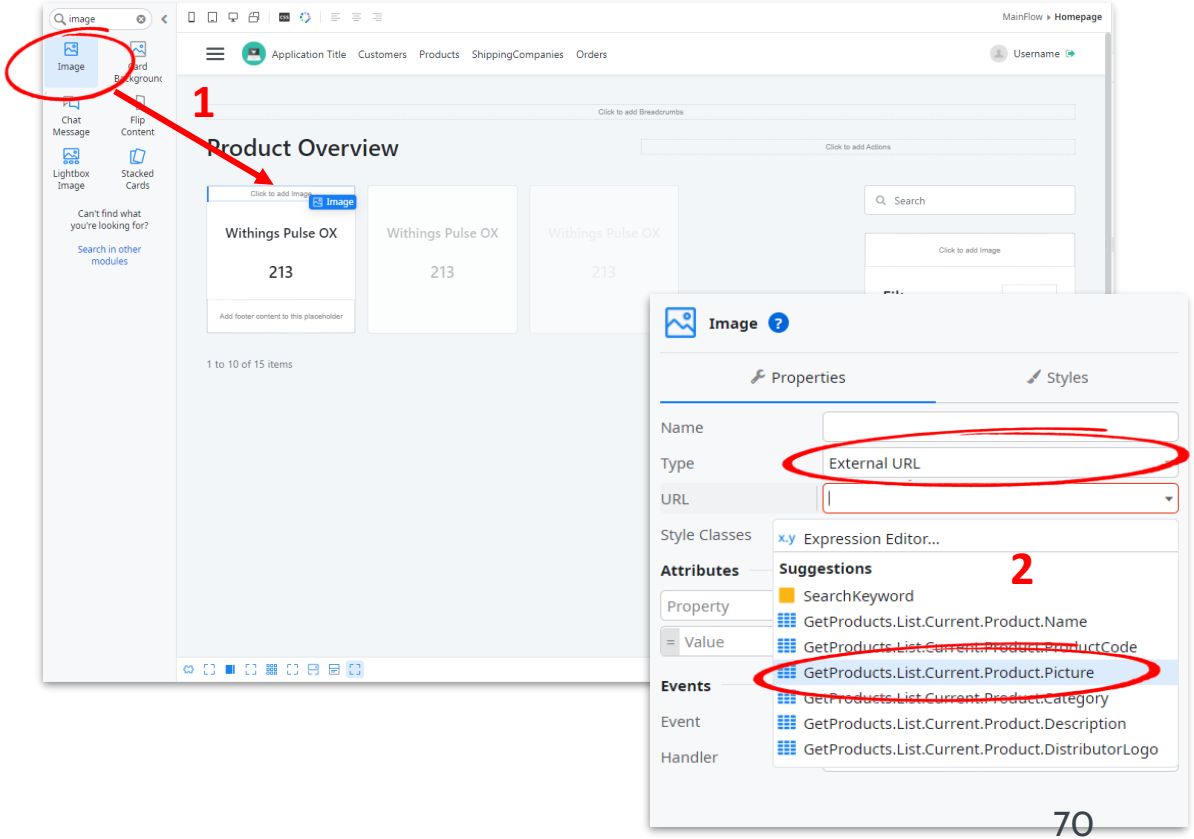
1. In the "**Homepage**" screen, drag the "**Product**" Entity onto "**Product Overview**" **section**. Make sure it indicates "**Replace data with Product**".



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" placeholder of the "Content\CardSectioned" component, within the product gallery;
2. In the properties of the Image widget node, change the Type to "**External URL**" and set the URL to **"GetProductsByCategory.List.Current.Product.Picture"**.

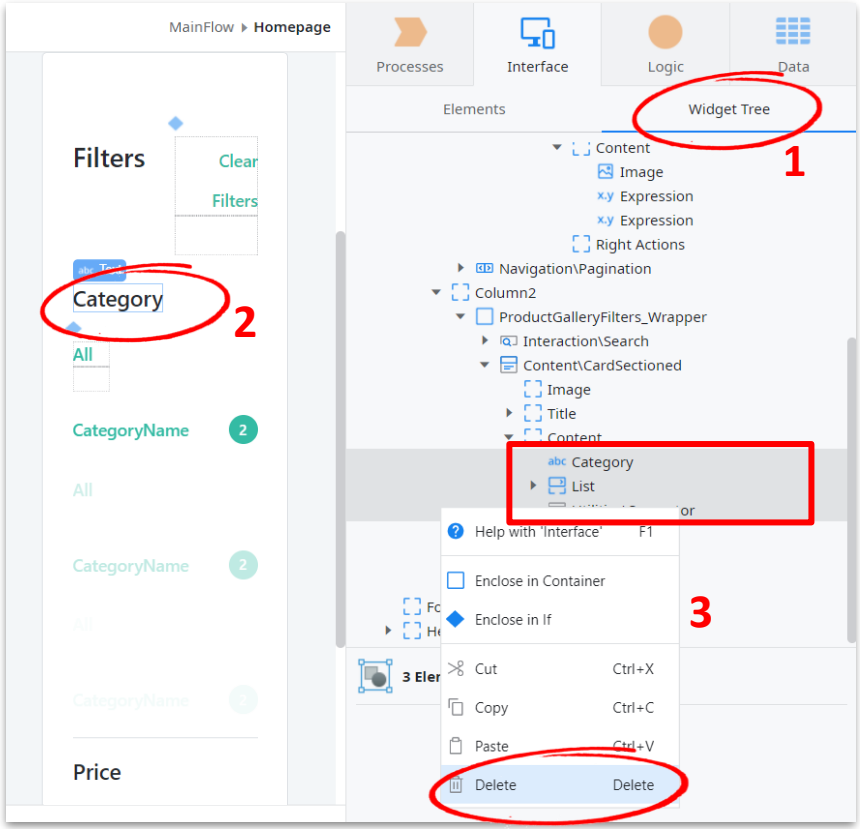


In the next steps we will adapt our screen and delete some of the filters from the initial template we don’t want to use.

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. "List";
   c. "Utilities\Separator".

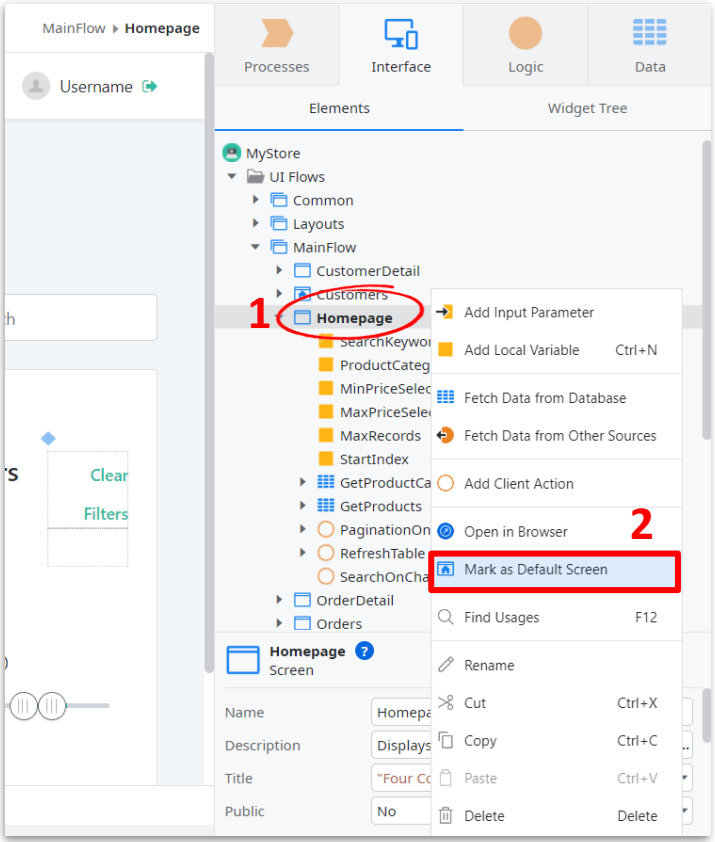


QUICK TIP

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

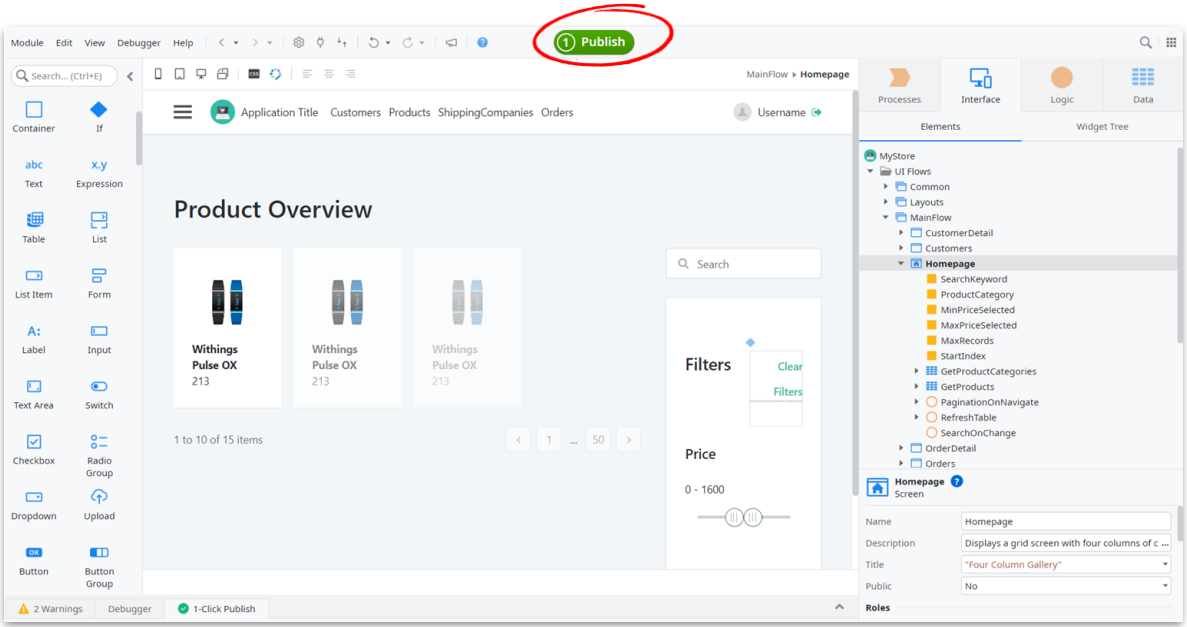
1. Set this page as the Entry Point by right-clicking the “**Homepage”** screen;

2. 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;

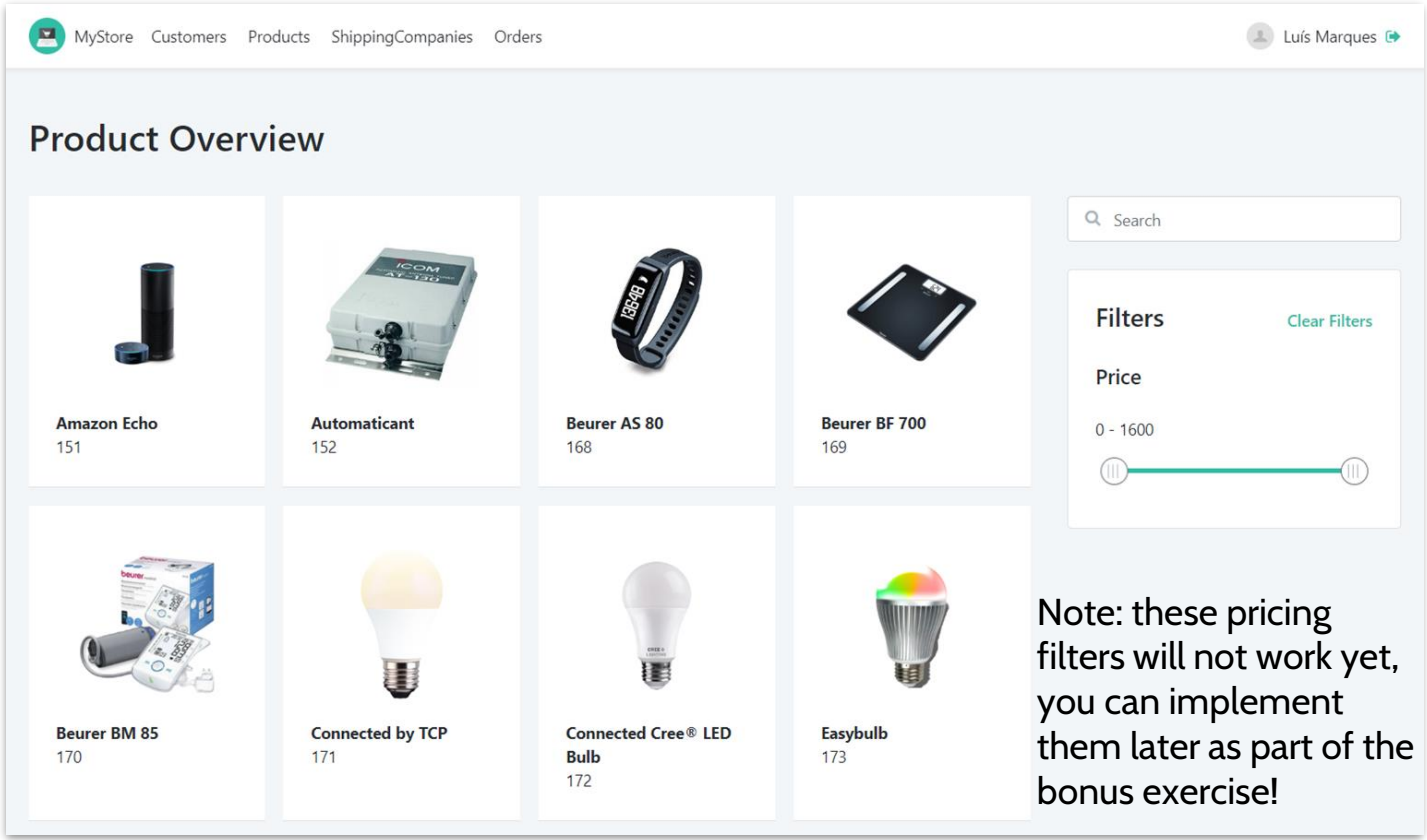
2. Click on the blue icon to see your web application running.
   *Once the deployment process is finished the "**1-Click publish**" is updated to a blue icon.*



QUICK TIP

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

End of Section 3 - Image Gallery with fluid layout



Note: these pricing filters will not work yet, you can implement them later as part of the bonus exercise!

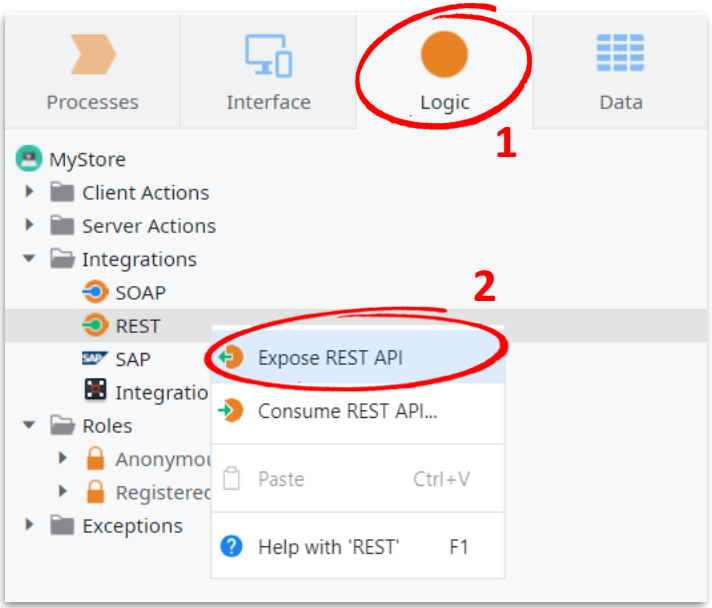
Section 4

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.

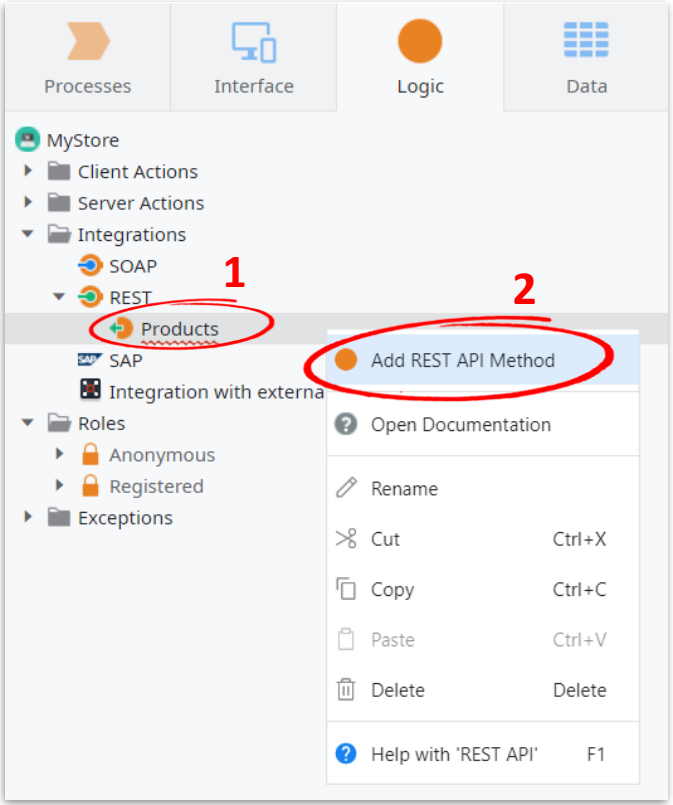
1. From the “**Logic**” tab, go to the “Integrations” folder;
2. **Right-click** on the REST element and select “**Expose REST API**”.



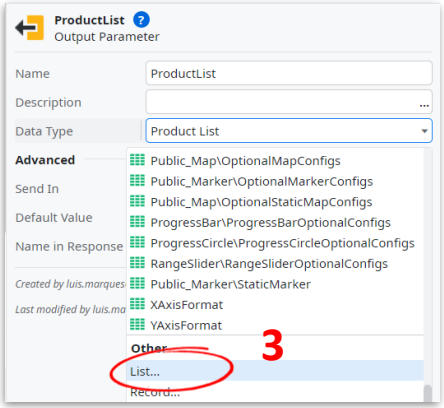
QUICK TIP

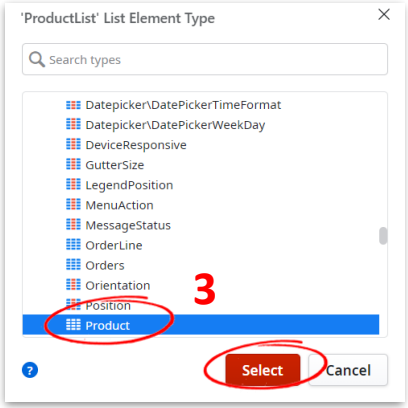
1. Try to use keyboard shortcut: **Ctrl+3** to access Logic tab

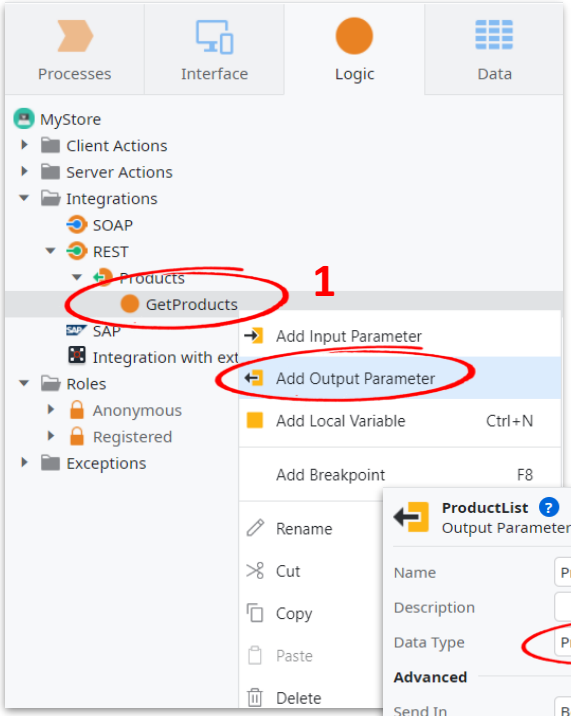
1. Rename the REST Integration to **“Products”**;
2. Right-click on the “Products” Integration and add a new REST API Method and rename it to **“GetProducts”**.

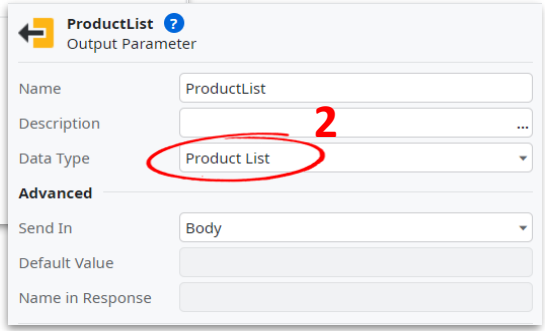


1. Right-click on the “GetProducts” method and add an **output parameter** named **“ProductList”**;
2. Make sure the Data type is set to “**Product List**”;
3. If not, select the **“List…” Data Type** and then **“Product”** and **click “Select”**.





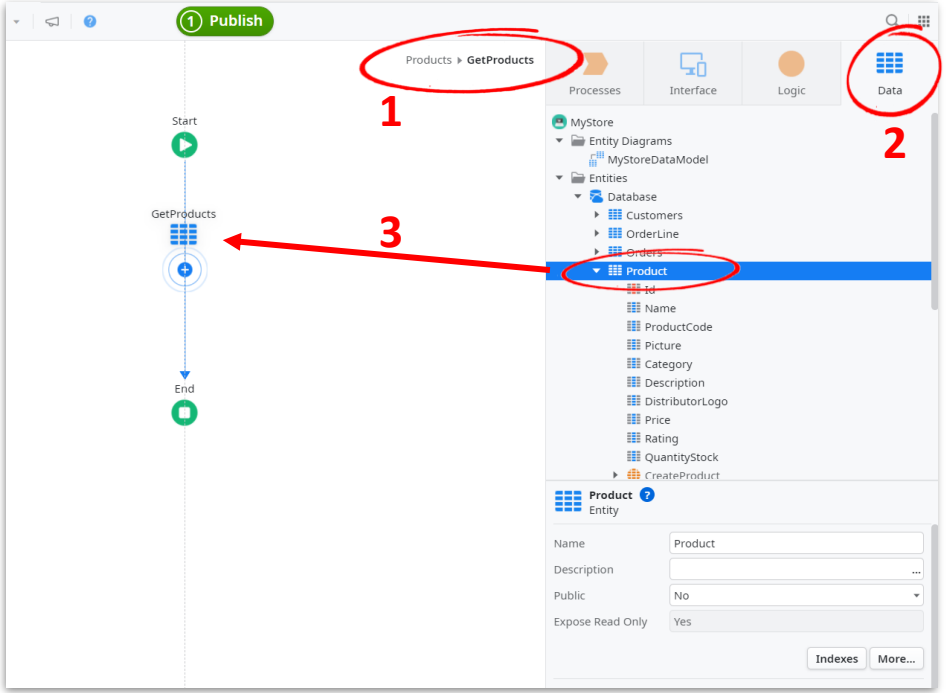




We have created the Web Service and one method. Let‘s fill the method with logic in a low code way.

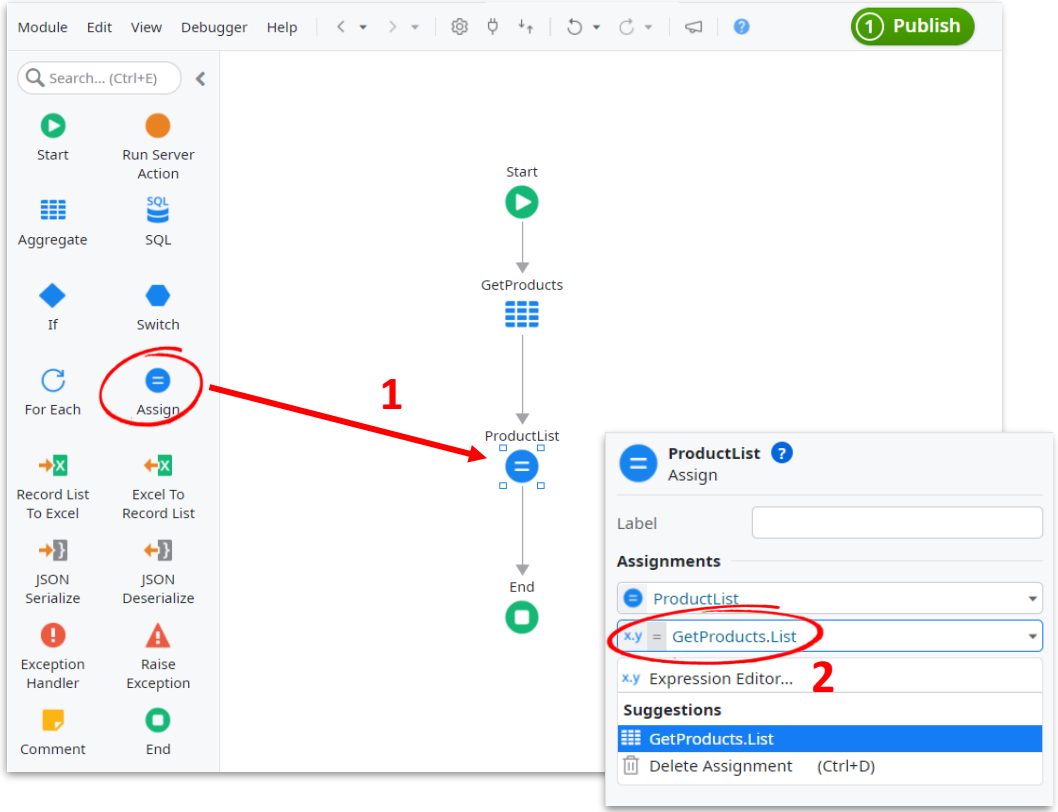
Add the “GetProducts” Aggregate

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”** node;
2. Assign to the **“ProductList”** parameter the “**GetProducts.List”** (result of the query).

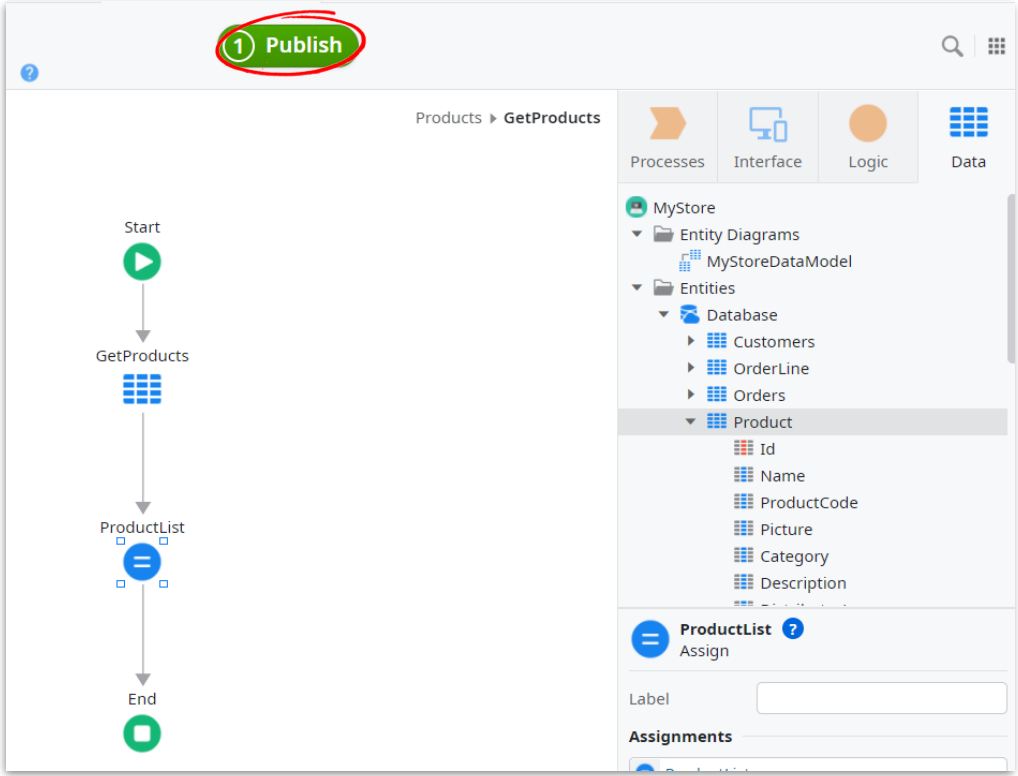


QUICK TIP

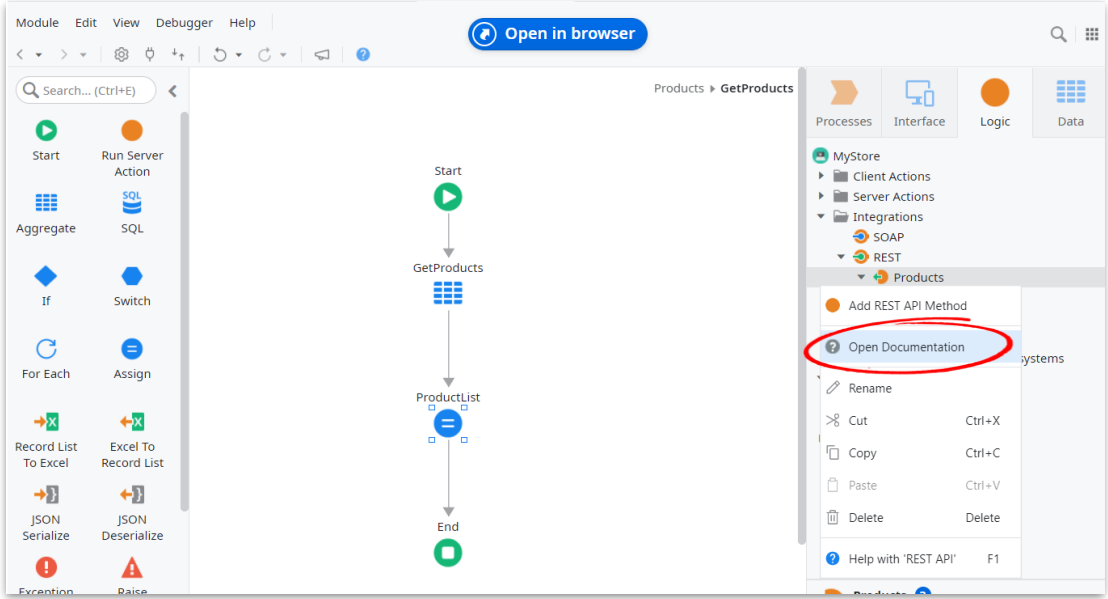
Try hovering over the arrow underneath GetProducts, click on the circle that appears and let the OutSystems AI Assisted Development help you go through the next step.

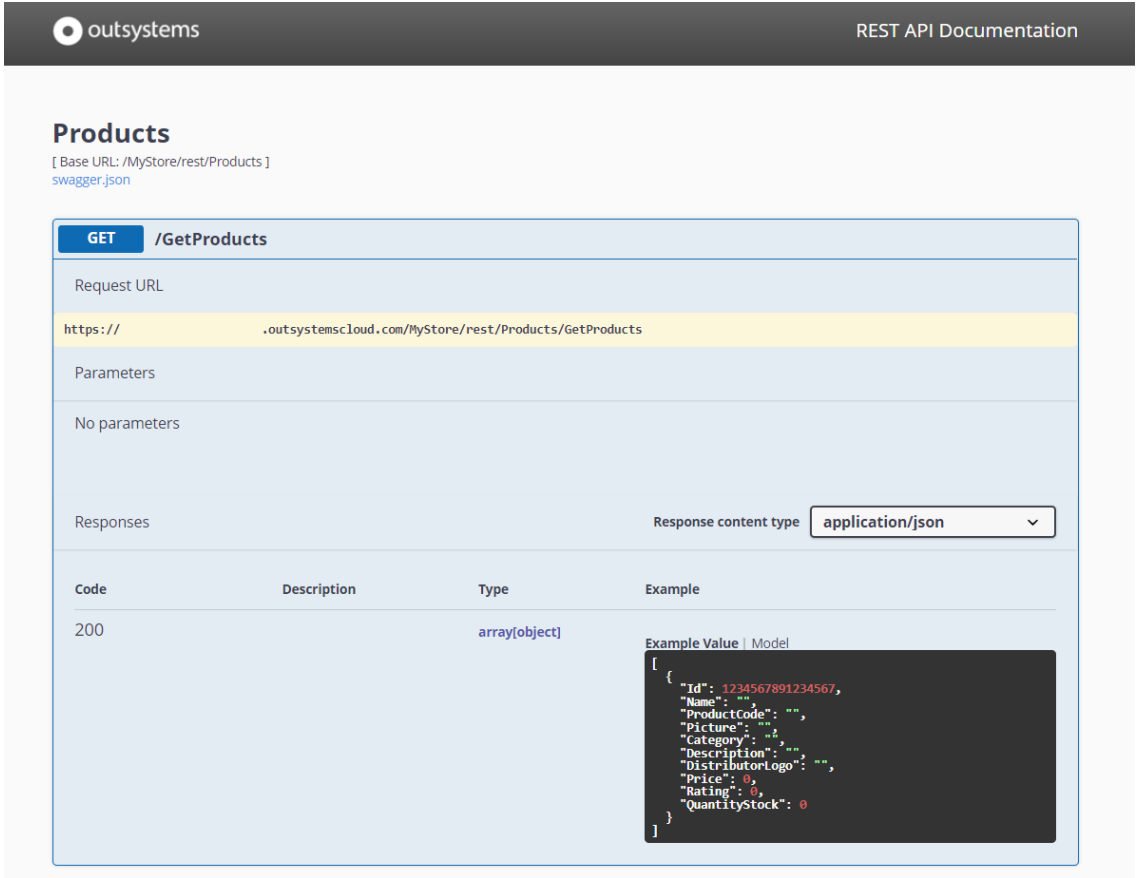
More information about AI Assisted Development here

1. Click the "**1-Click publish**" to initiate the deployment process.



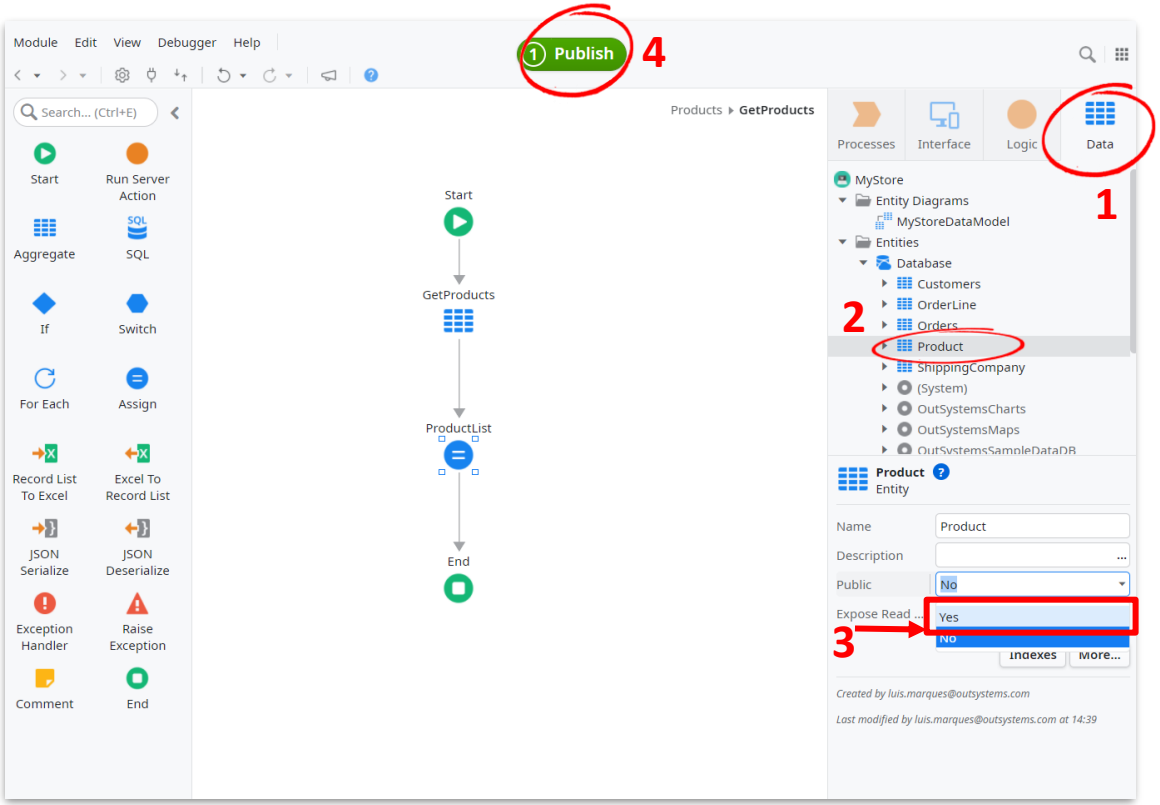
1. After the deployment, **right-click** on top of the “**Products” API** and click “**Open Documentation**”.



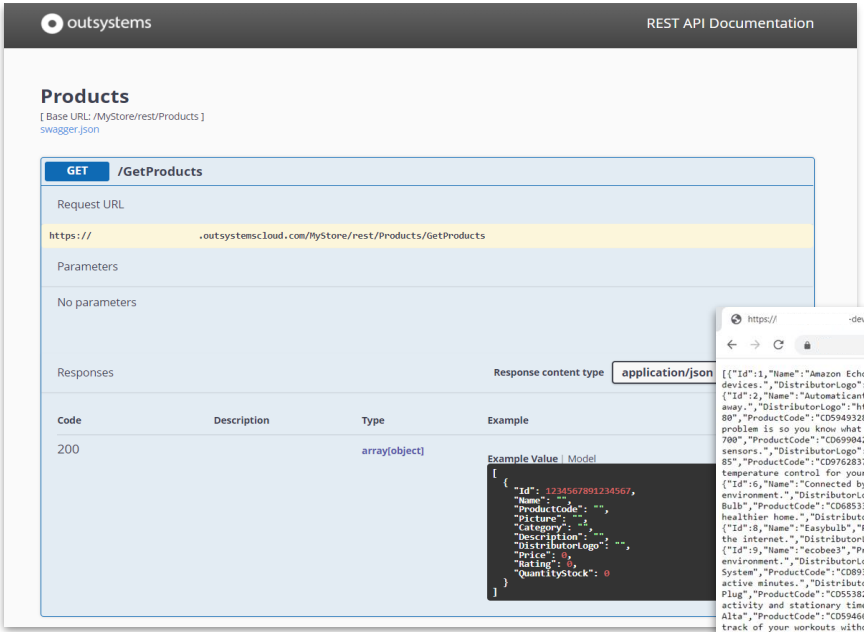

outsystems
REST API Documentation
Products
[Base URL: /MyStore/rest/Products]
swagger.json
GET
/GetProducts
Request URL
https://
.outsystemscloud.com/MyStore/rest/Products/GetProducts
Parameters
No parameters
Responses
Response content type
application/json
Code
Description
Type
Example
200
array[object]
Example Value | Model
[
{
"Id": 1234567891234567,
"Name": "",
"ProductCode": "",
"Picture": "",
"Category": "",
"Description": "",
"DistributorLogo": "",
"Price": 0,
"Rating": 0,
"QuantityStock": 0
}
]

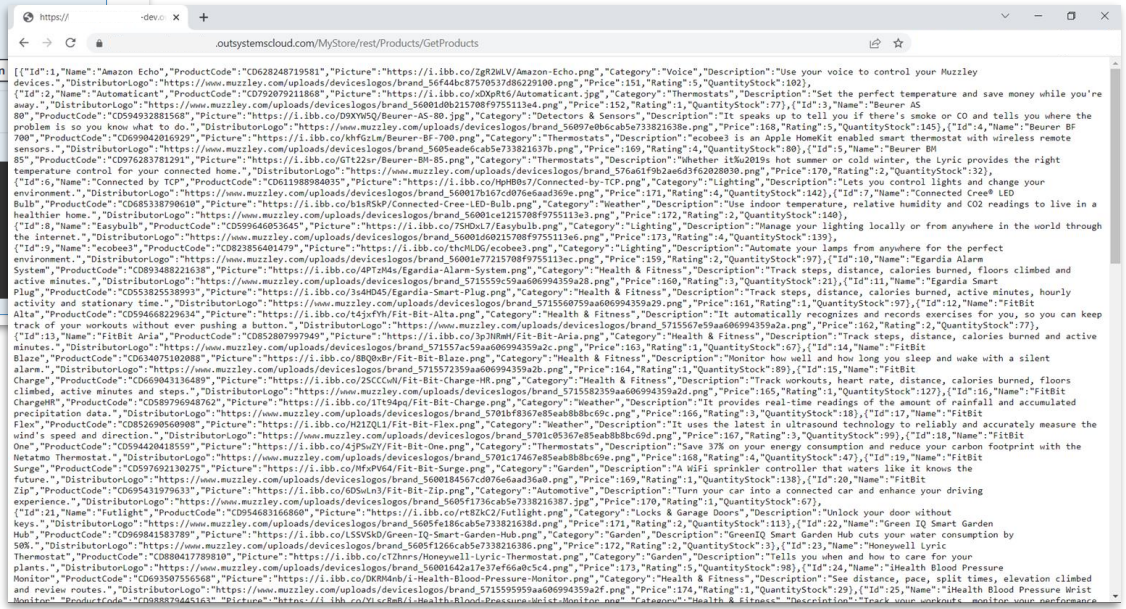
In a later exercise, we will reuse the “Product” Entity in our Mobile Application. We need to expose the “Product” Entity as public to make it accessible to other OutSystems apps.

1. Go to the **“Data” tab**;
2. Select the **“Product” Entity**;
3. Set the **Public** property to **Yes**;
4. 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

A. Bonus Exercises

In this section you will find 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.

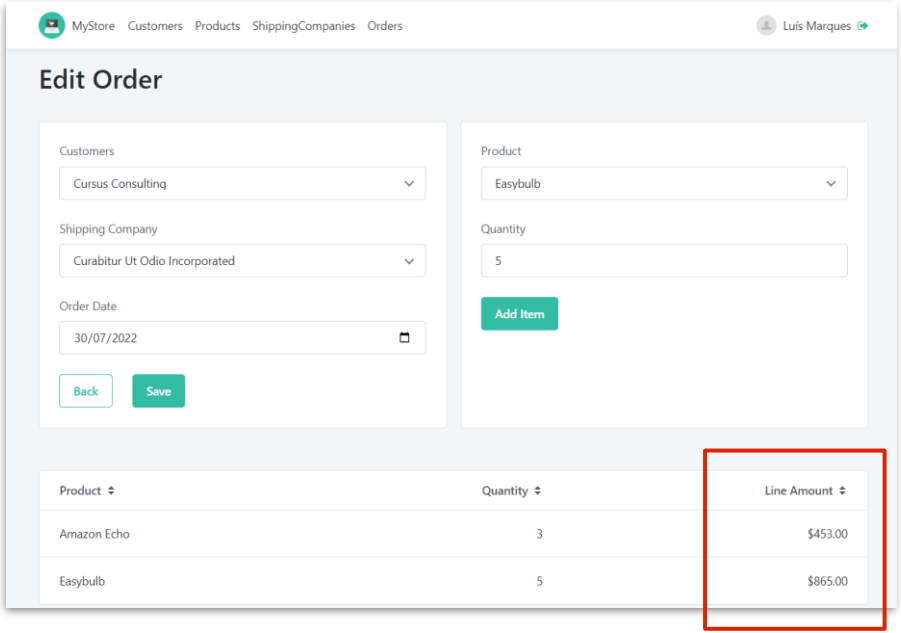
Good Luck!

Bonus exercises are optional!

Display Amount/Cost for each Order Line

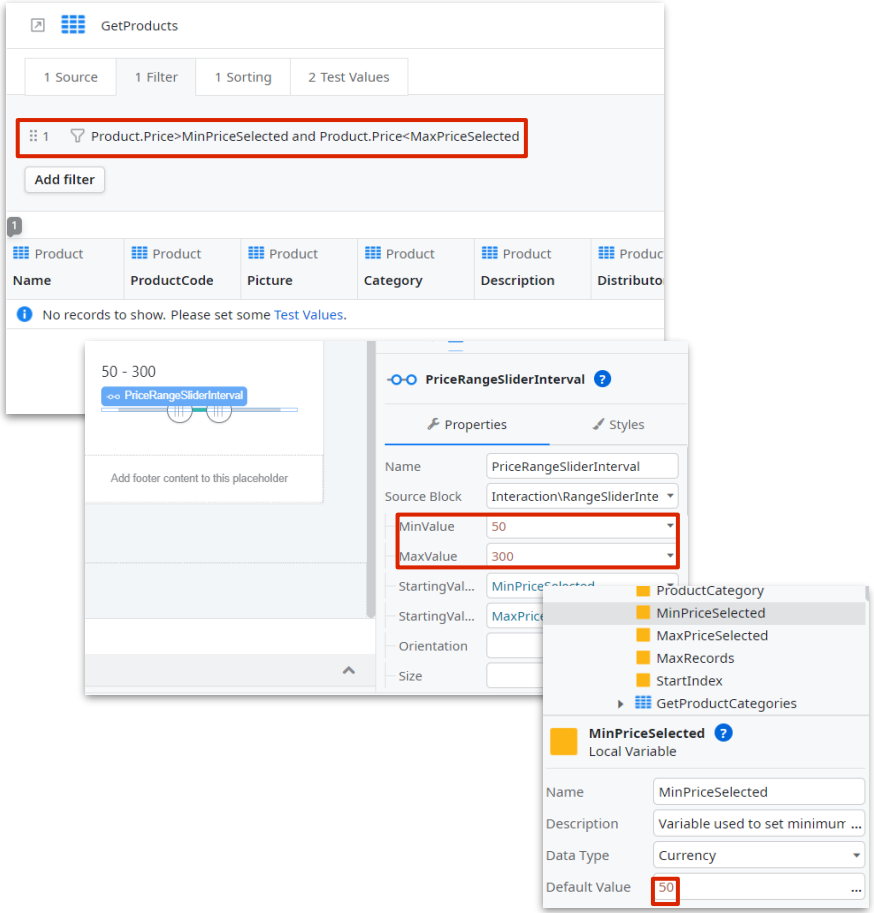
For each Order Line lets calculate its cost and display it on the Order Line table that is on the "OrderDetail" screen. For that:

- 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 or use the "GetProduct" Entity Action);
  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.



Enable filtering your products based on price

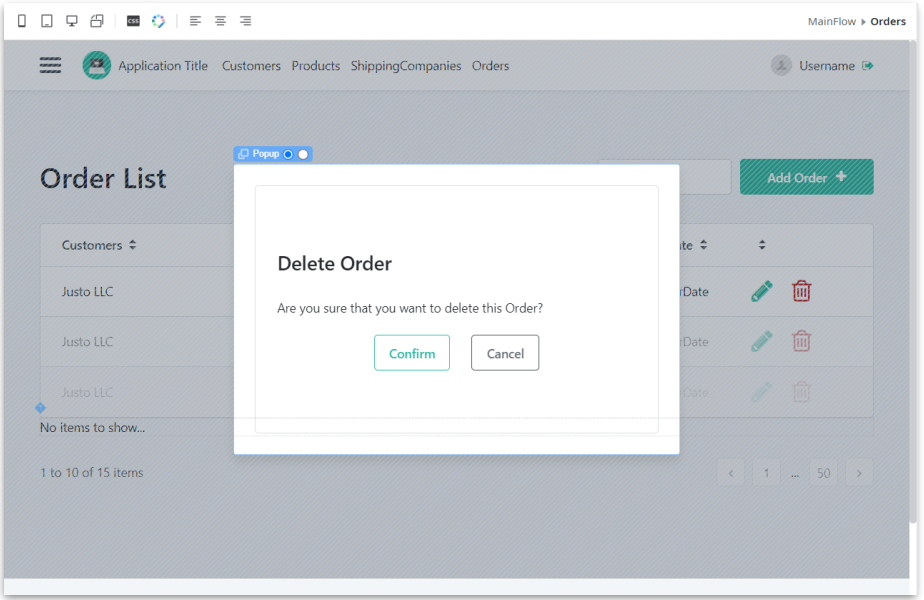
- In the “**Homepage”** screen open up the “**GetProducts”** Aggregate
- Switch back to the “**Homepage”** UI, and click on the “PriceRangeSliderInterval”.
  - Change the minimum value to *50*;
  - Change the maximum value to *300*.
- Now, for both the local variables **MinPriceSelected** and **MaxPriceSelected**, click on the variable and change the default value to *50* and *300* respectively.

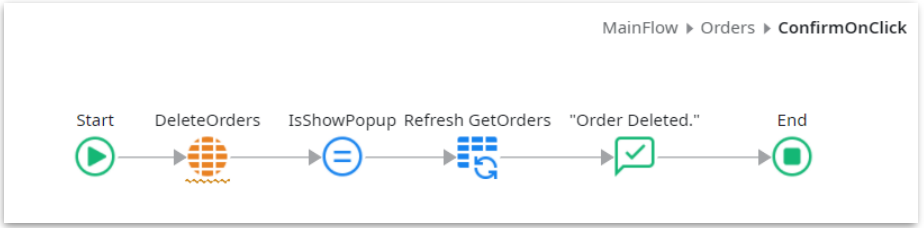


1. Enable users to delete orders

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

- Add the “Trash” Icon to the Order List table;
- On the “Orders” screen use the pop-up widget, and use the card sectioned widget in the pop-up;
- Create a Boolean local variable to control when the popup should be displayed;
- The icon should open the pop-up, the “Cancel” button should hide it and the “Confirm” button should start a new client action to delete the order;
- Finally create the business logic to delete the corresponding order;
    - You will need to change the “Delete Rule” of the **“OrderId” Attribute**, in the **“OrderLine” Entity**, from “Protect” to “Delete”. This ensures that, when an Order is deleted, the associated Order Lines are deleted too.






outsystems
Congratulations!
You have successfully built a web app

outsystems

Jumpstart training