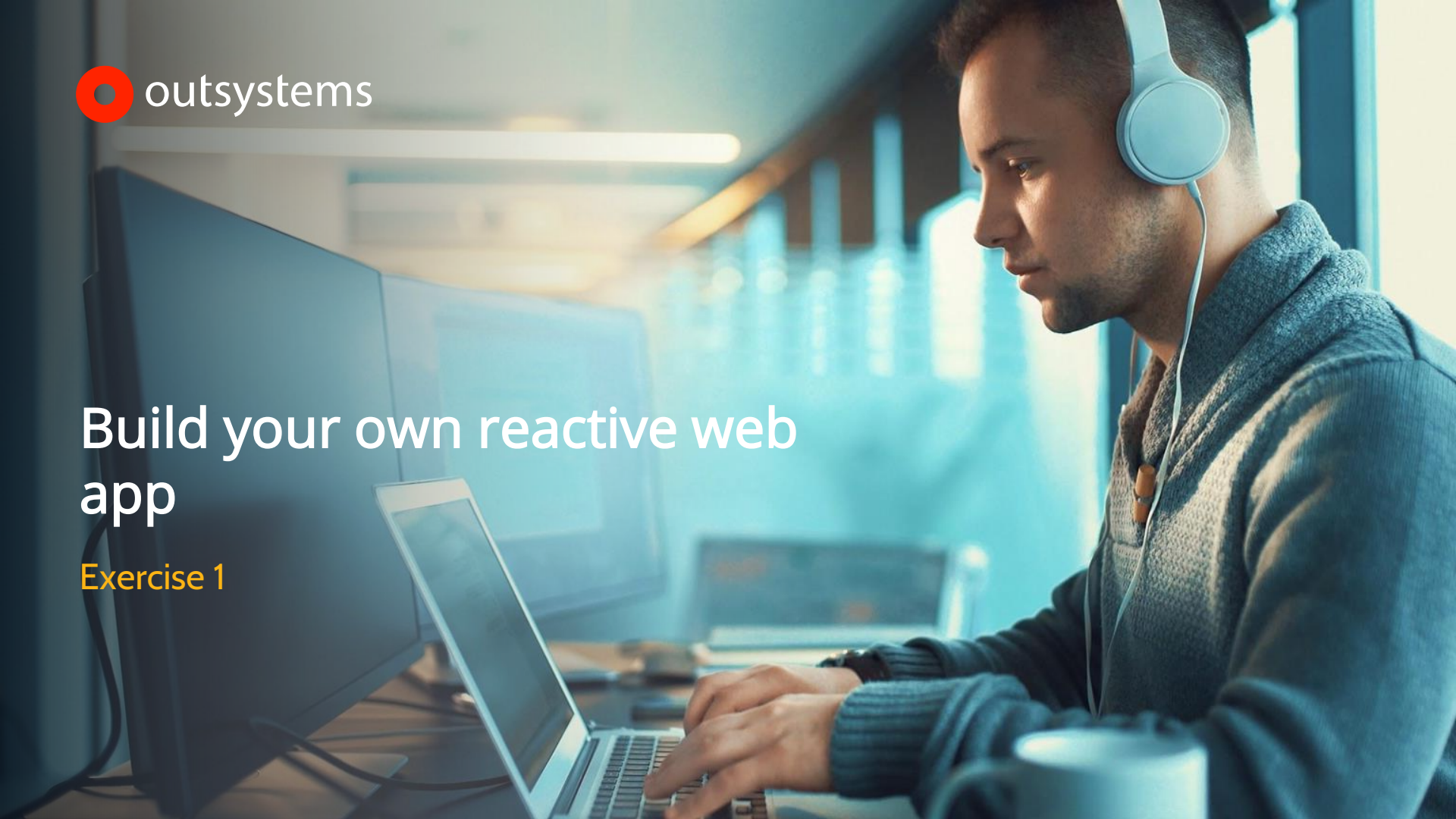




# Build your own reactive web app

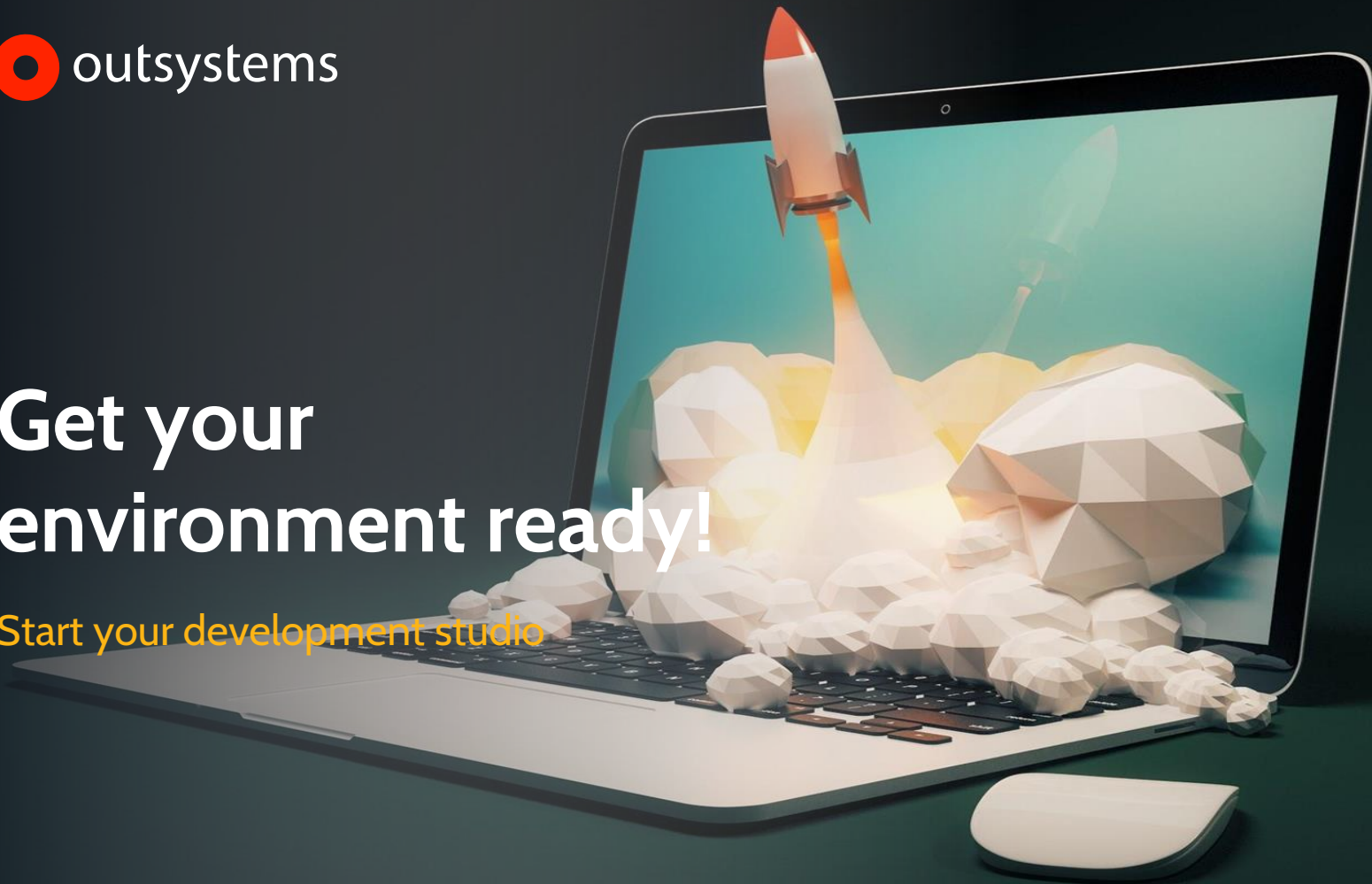
## Exercise 1





# Get your environment ready!

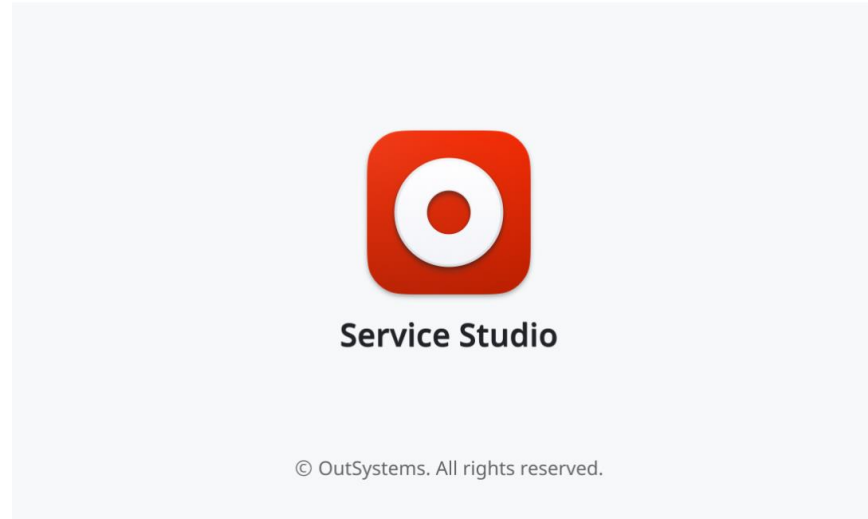
Start your development studio



# Development Studio

Open and start

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



# Web application

## Exercise 1



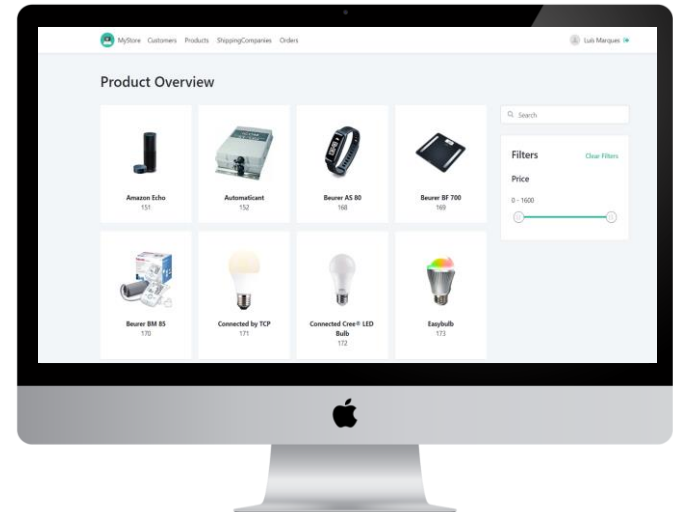
# Context

*“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



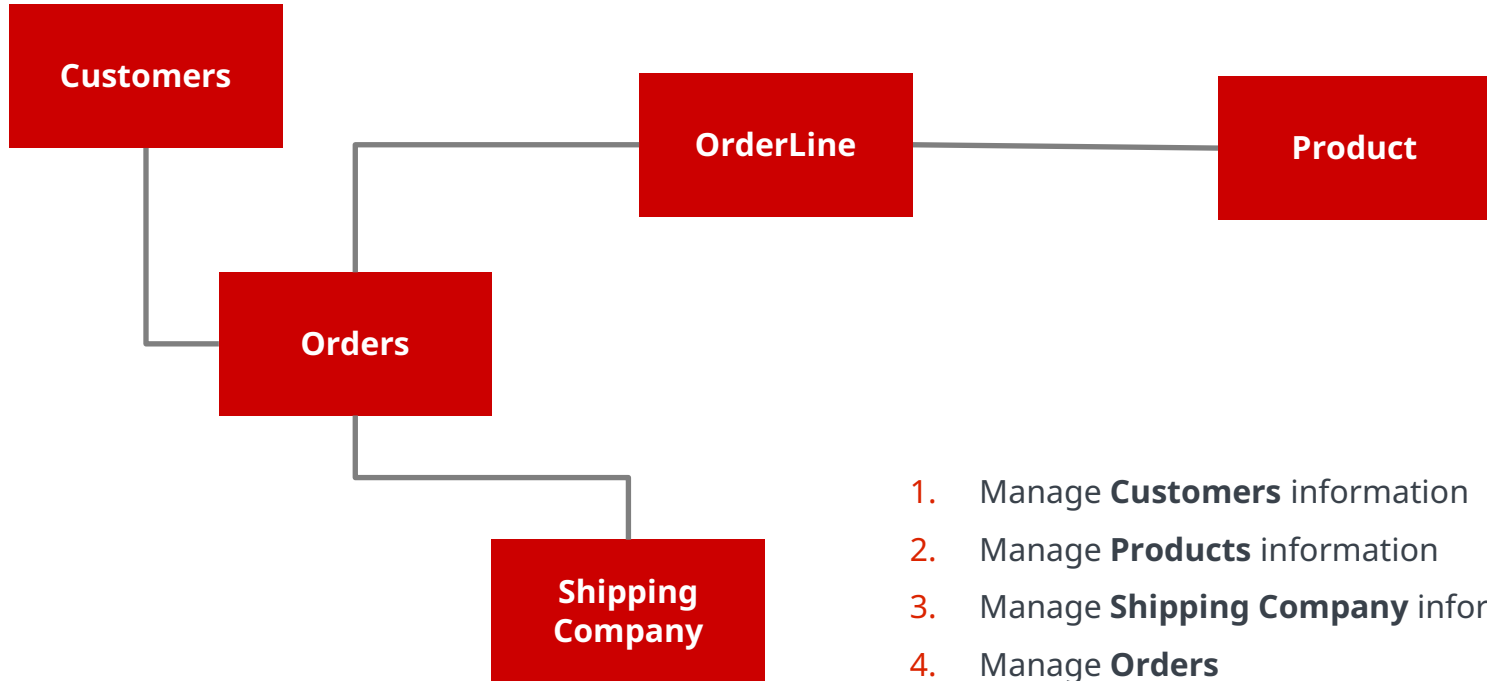
“As a **customer representative** I should be able to search for customers, edit their information, create new customers and register their purchases”.

---

“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.

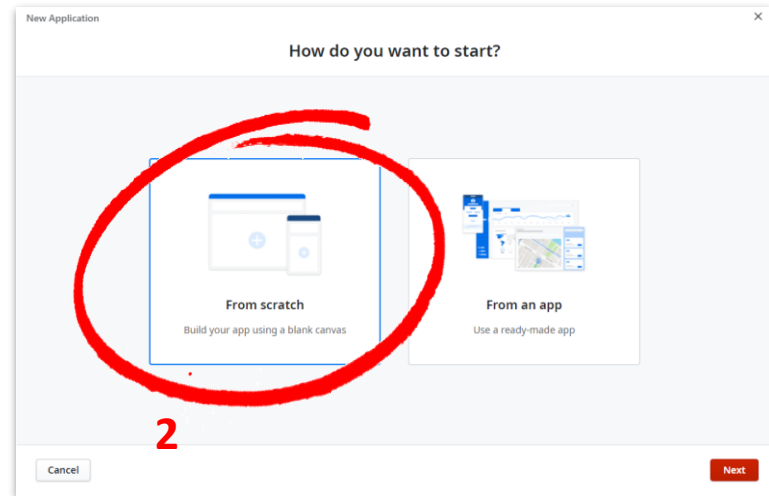
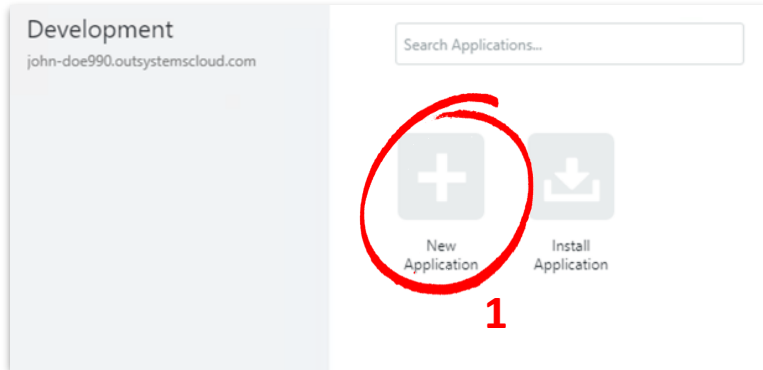
## Section 1

### A

## 1. Starting point

Create a new mobile app

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

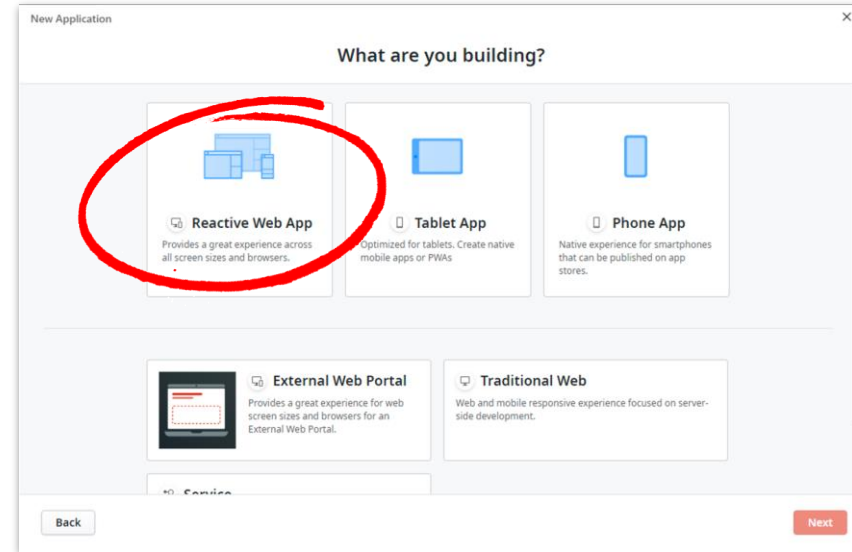


## Section 1 > A > 2. Create a new application

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



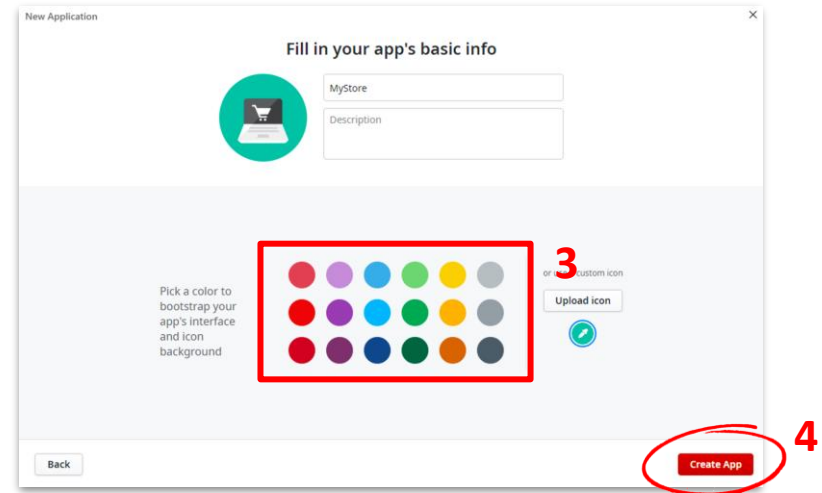
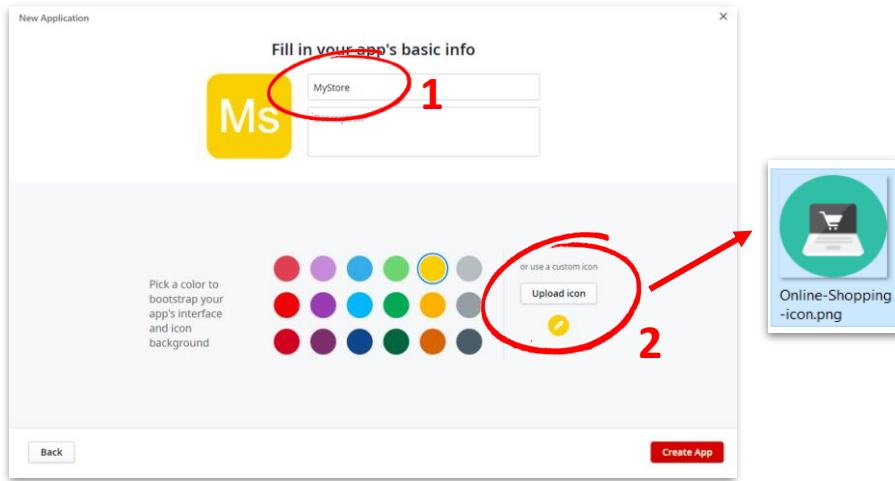
## Section 1

## A

## 2. Application Name & Icon

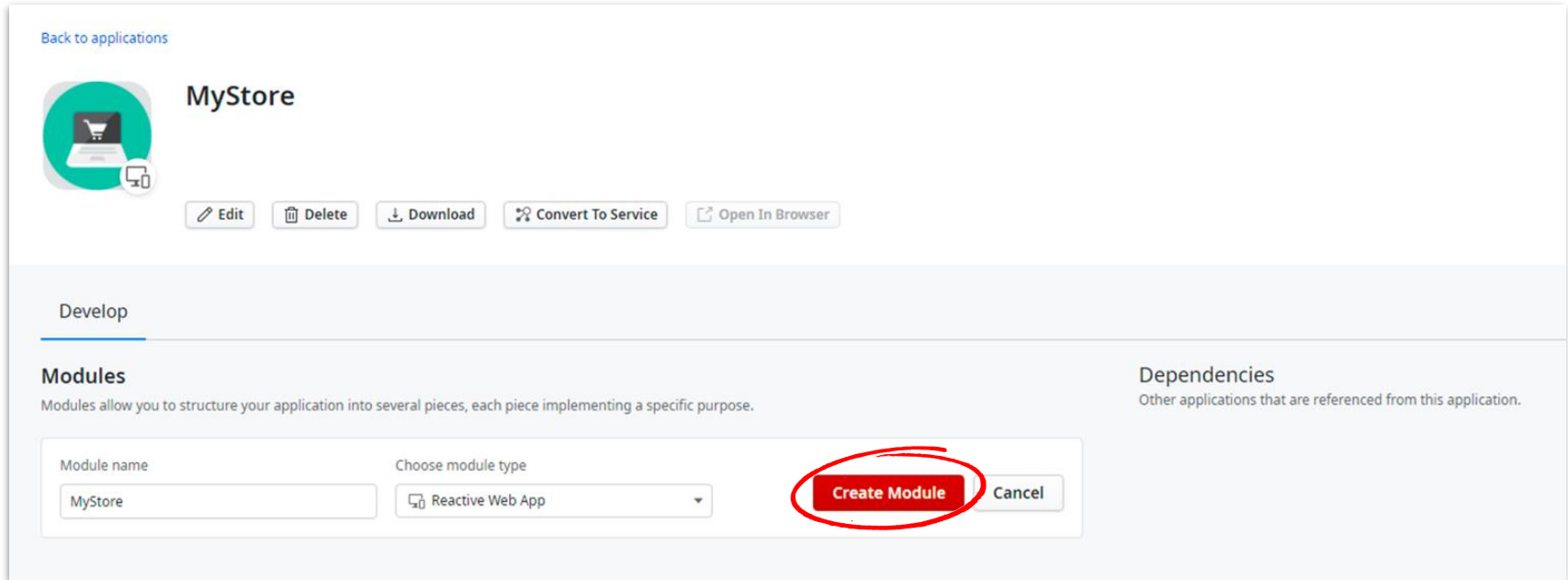
### New Application

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.



The screenshot shows the 'MyStore' application interface. At the top, there is a 'Back to applications' link. Below it, the application name 'MyStore' is displayed next to a shopping cart icon. A row of action buttons includes 'Edit', 'Delete', 'Download', 'Convert To Service', and 'Open In Browser'. The 'Develop' section is active, showing a 'Modules' section with a description: 'Modules allow you to structure your application into several pieces, each piece implementing a specific purpose.' Below this, there is a form with a 'Module name' field containing 'MyStore' and a 'Choose module type' dropdown menu set to 'Reactive Web App'. A red circle highlights the 'Create Module' button, which is next to a 'Cancel' button. To the right, there is a 'Dependencies' section with the text 'Other applications that are referenced from this application.'

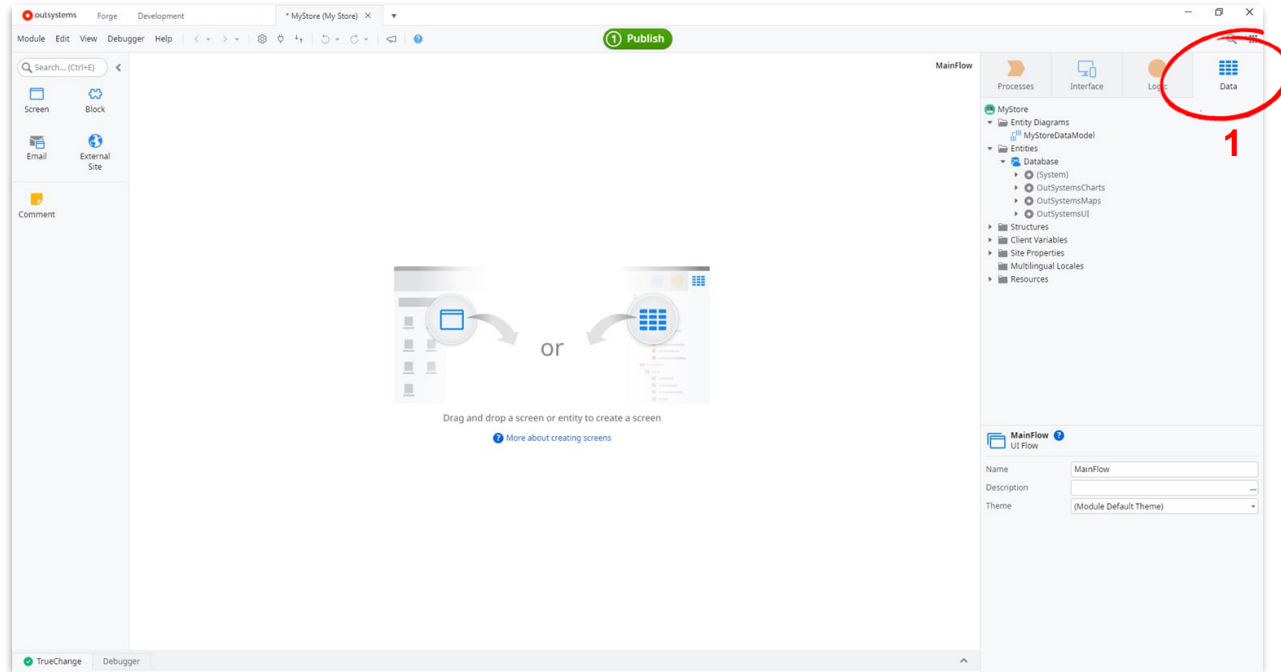
Let's start by creating our first table in our data model. In Outsystems tables are known as **Entities** while columns are known as **Attributes**.

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.



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



## Section 1 > A > 6. Add a New Entity

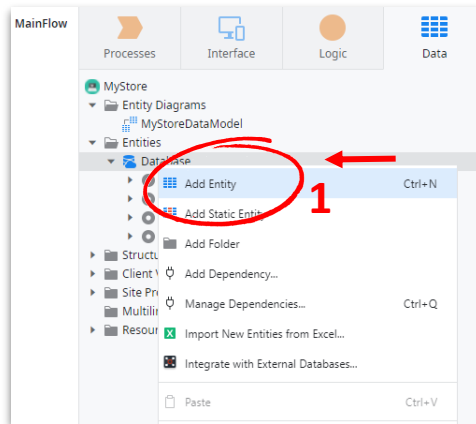
### Create the “Customer” Entity

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.

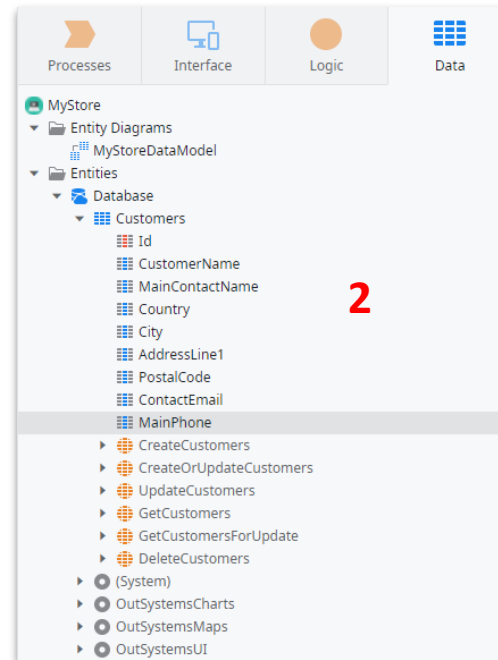


QUICK TIP

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).



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	

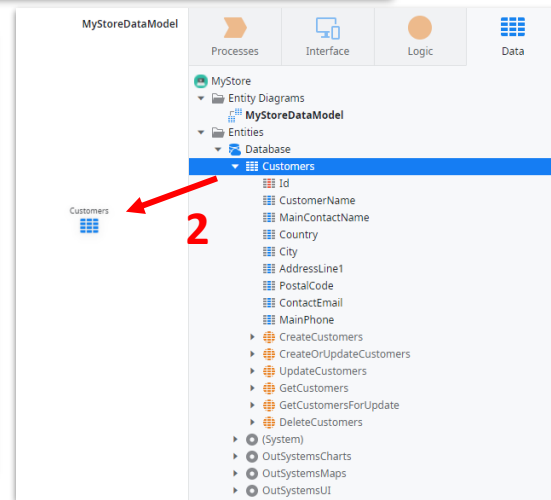
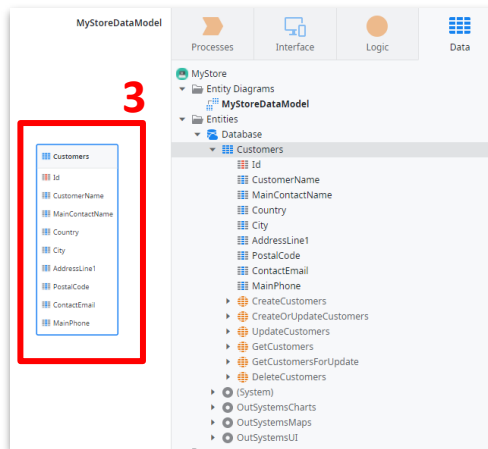
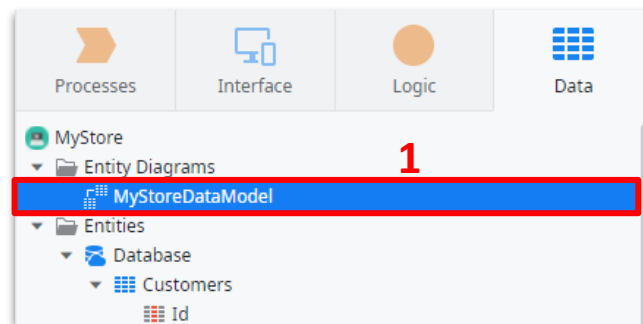


## Section 1 > A > 7. Entity Diagrams

Add the “Customer” Entity to Entity Diagrams

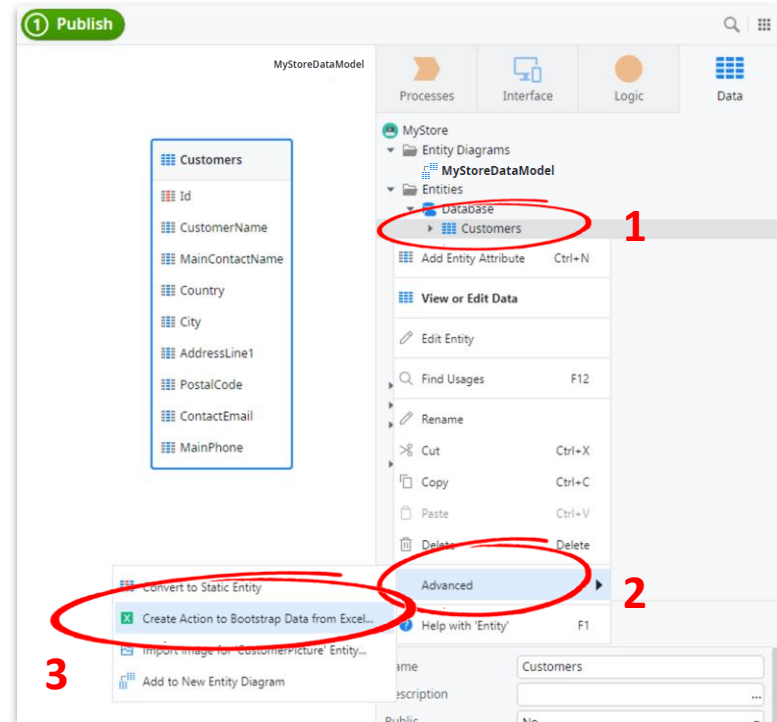
An **Entity Diagram** in OutSystems helps you visualize the relationship between your entities.

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

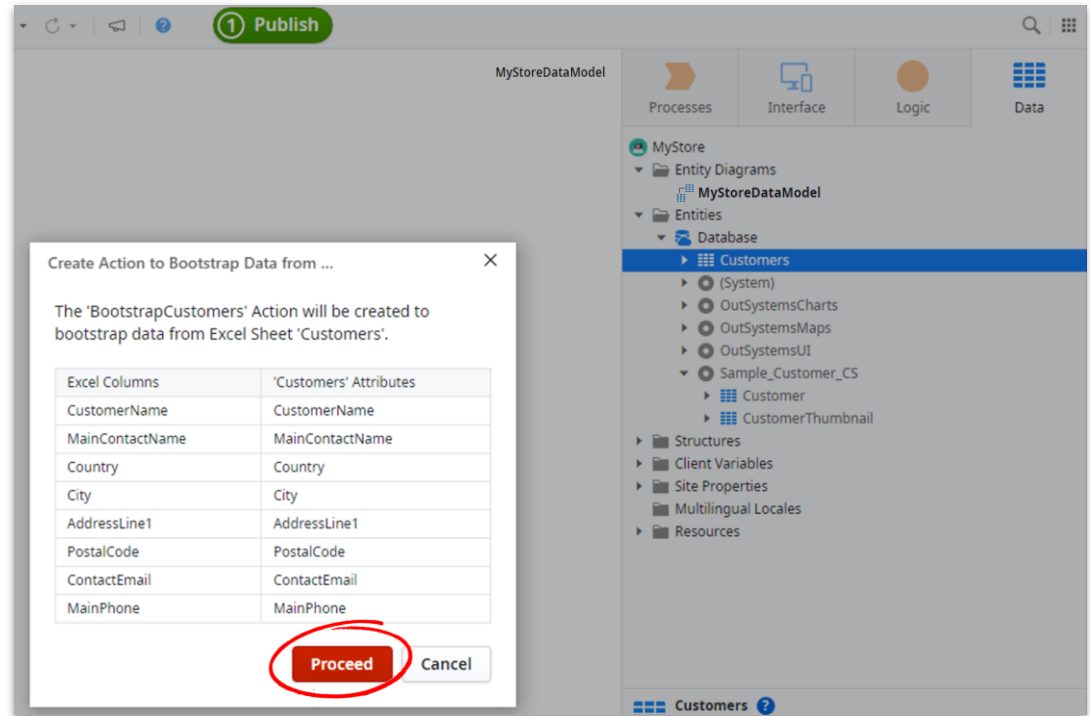


This diagram will show us how the entities are related to each other.

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:

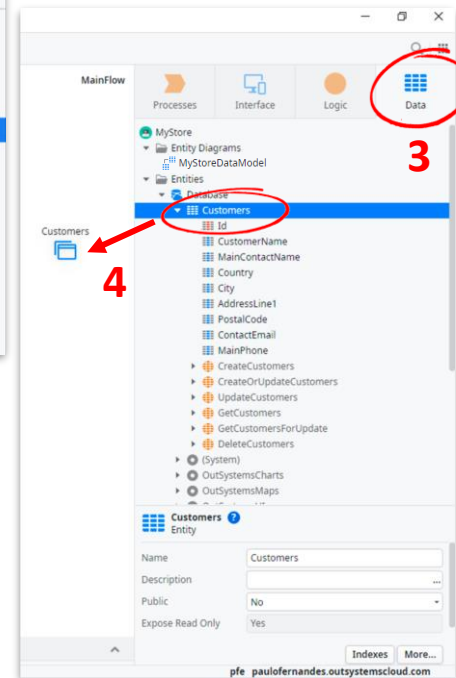
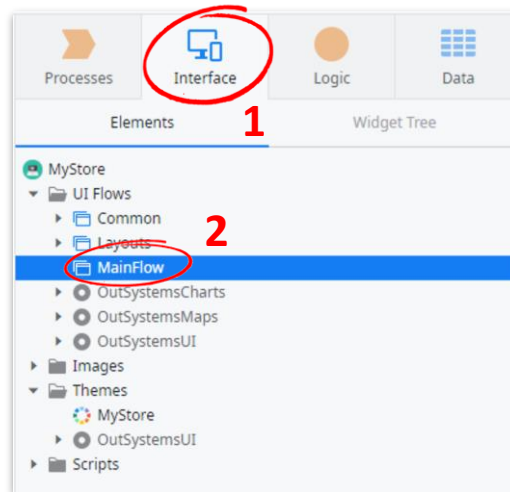
1. From a **blank slate**, inserting each component with drag-and-drop
2. From one of our many **screen templates** - these come with sample data that can be modified for your own needs
3. Using a feature called "**scaffolding**" which automatically creates lists and forms based on your database entities.

We'll explore these different methods during the exercises.

## Section 1 > B > 2. Create List Page

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

## Section 1 > B > 3. Change Interface

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.

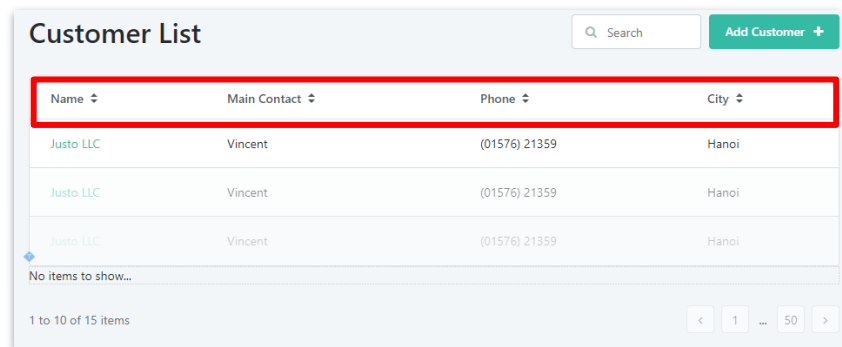
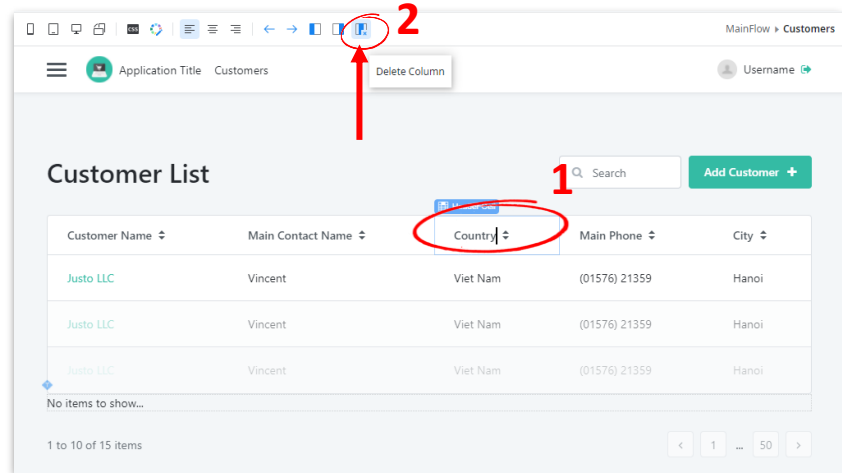
The image shows a three-step process in a development tool:

- Step 1:** The **Interface** tab is selected in the top navigation bar. In the **Elements** panel, the **Customers** entity is highlighted with a red circle and the number 2.
- Step 2:** The **Data** tab is selected. In the **Entity Diagrams** panel, the **MainPhone** attribute is highlighted with a red circle and the number 3. A red arrow points from this attribute to the table in the next screenshot.
- Step 3:** The **Customer List** table is shown. A red arrow points from the **MainPhone** attribute in the previous screenshot to a new column header **MainPhone** being added to the table. The table contains three rows of customer data.

Customer Name	Main Contact Name	Country	City
Justo LLC	Vincent	Viet Nam	Hanoi
Justo LLC	Vincent	Viet Nam	Hanoi
Justo LLC	Vincent	Viet Nam	Hanoi

## 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).

TrueChange		Debugger		1-Click Publish			
1	Uploading	Storing a new version into 'https://		outsystemscloud.com/ServiceCenter'.		14:38	
2	Compiling	Generating and compiling optimized code and database scripts.					14:38
3	Deploying	Updating database model and deploying the web application.					14:38
✓	Done	'MyStore' is now available at 'https://		outsystemscloud.com/MyStore'.		14:38	

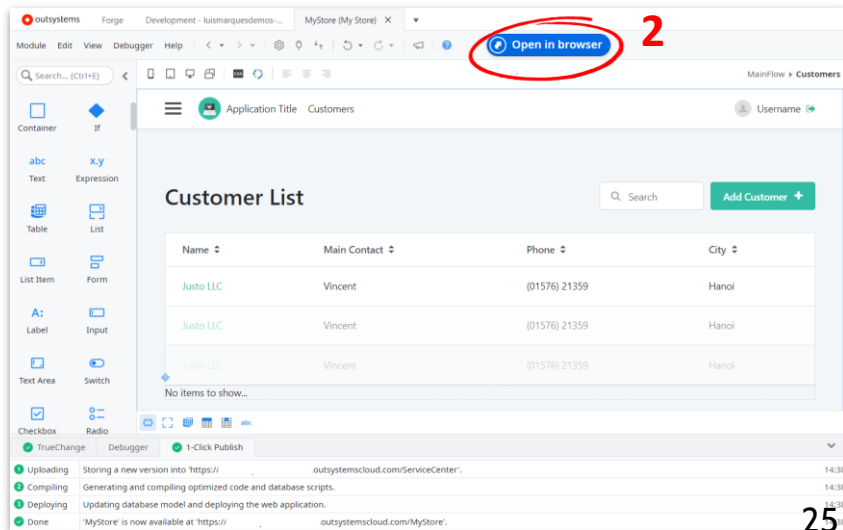
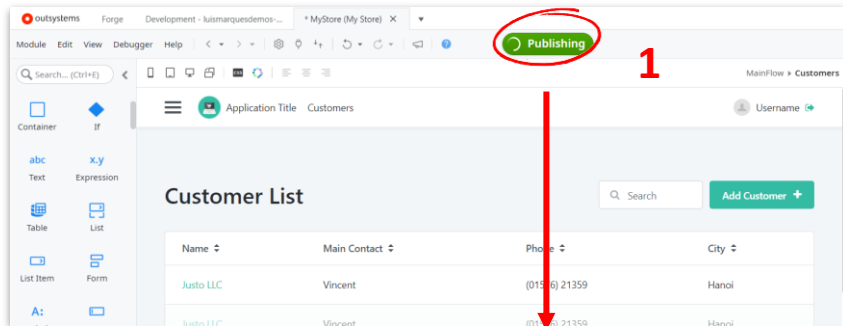
## Section 1

## B

## 6. Deploy and Test – 1-Click-Publish

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!

The image displays three overlapping screenshots of a web application interface. The leftmost screenshot shows a login page with a teal background, featuring a shopping cart icon, the text 'MyStore', and input fields for 'Username', 'Password', and a 'Remember me' checkbox, with a 'Login' button below. The middle screenshot shows the 'Customer List' page, which includes a search bar, an 'Add Customer' button, and a table with columns for Name, Main Contact, Country, Phone, and City. The rightmost screenshot shows the 'Edit Customer' page, which contains a form with fields for Customer Name, Main Contact Name, Country, City, Address Line 1, and Postal Code, alongside a faint illustration of an open cardboard box.

Name	Main Contact	Country	Phone	City
Consectetur Limited	Lamar	Afghanistan	07624 123154	Kabul
Vitae Diam Inc.	Brady			
Enim Industries	Hamish			
Mi Corp.	Yuli			
Cras Eget Industries	Ishmael			
Et Euismod Ltd	Tucker			
Fringilla Donec Inc.	Ezra			
Ac Inc.	Chaney			
Eu Ltd	Barry			
Nisl Sem Corp.	Logan			



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.

## Section 2

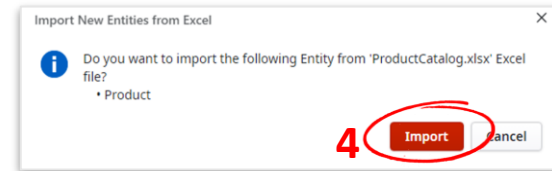
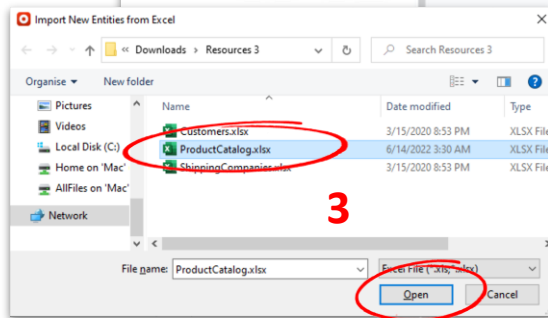
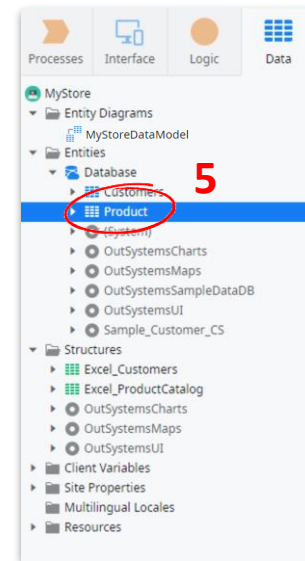
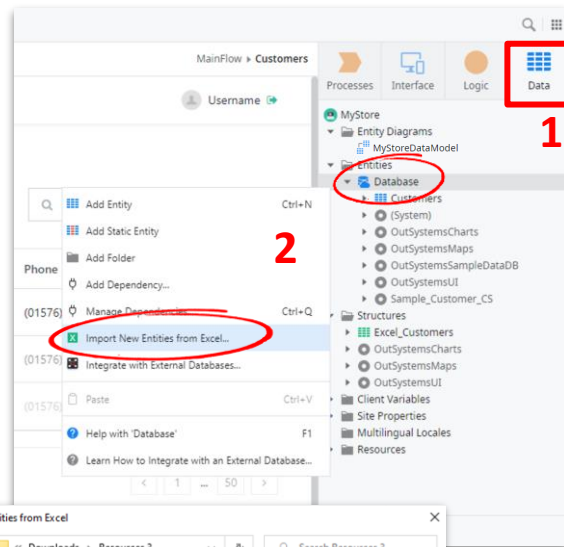
### A

## 1. Create Product Entity

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.



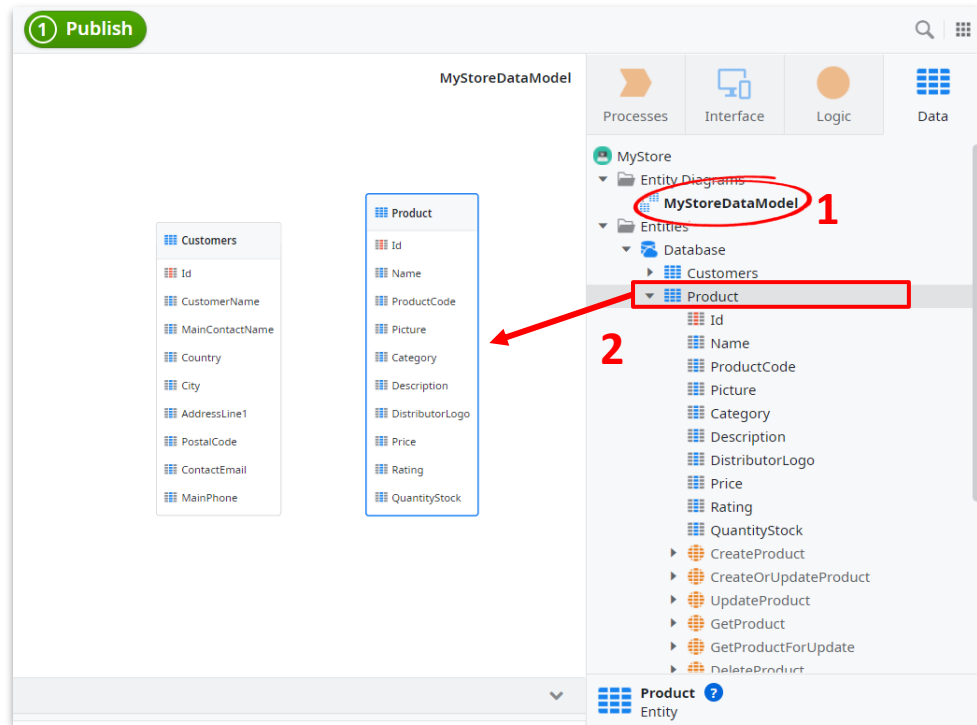
The OutSystems platform provides development **accelerators** to increase productivity on common, patterned and repeating development tasks (**scaffolding**).



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.

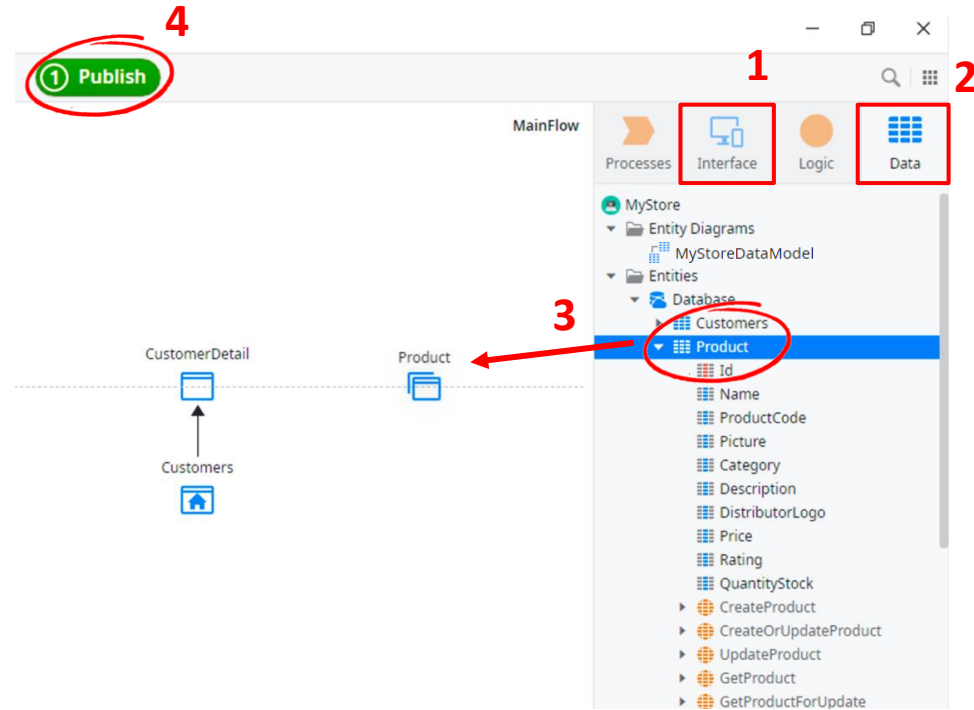


1. Double-click on the “**MainFlow**” on **Interface** menu;
2. Open the “**Data**” tab;
3. Drag the “**Product**” **Entity** to the “**MainFlow**” canvas to create both pages (list and edit) and tune them with the proper information (i.e. *maybe you want to remove the picture URL from the product list,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

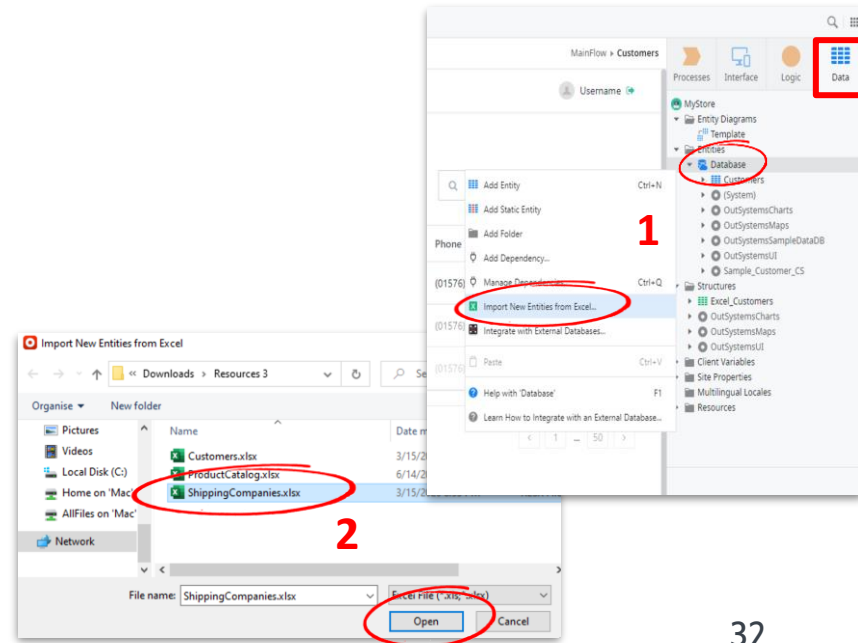
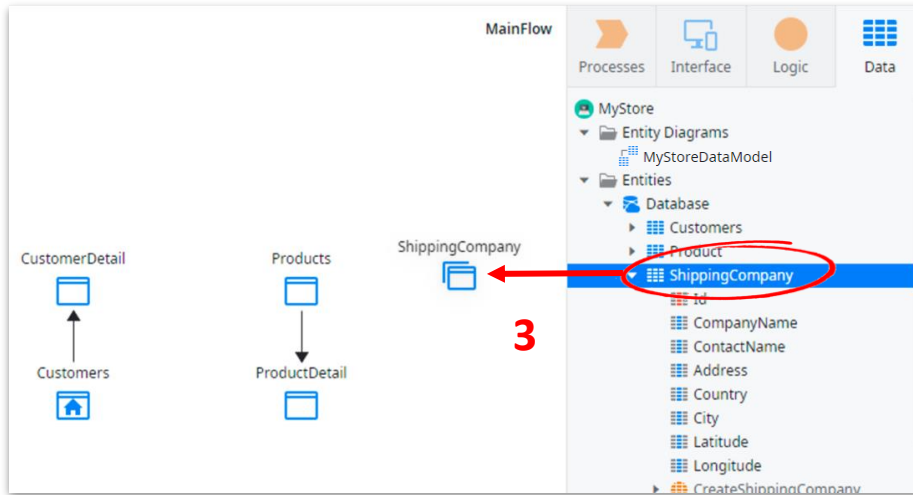


QUICK TIP

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

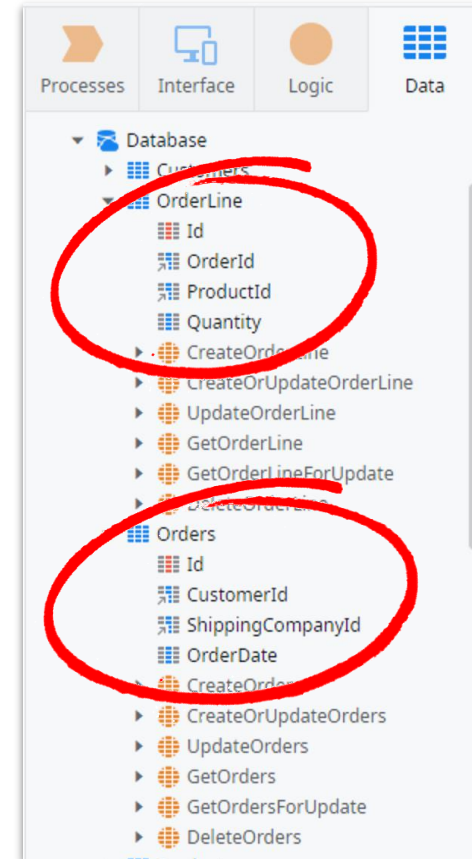
1. Create the “OrderLine” Entity with the following additional attributes:

Name	Data Type
OrderId	Order Identifier
ProductId	Product Identifier
Quantity	Integer



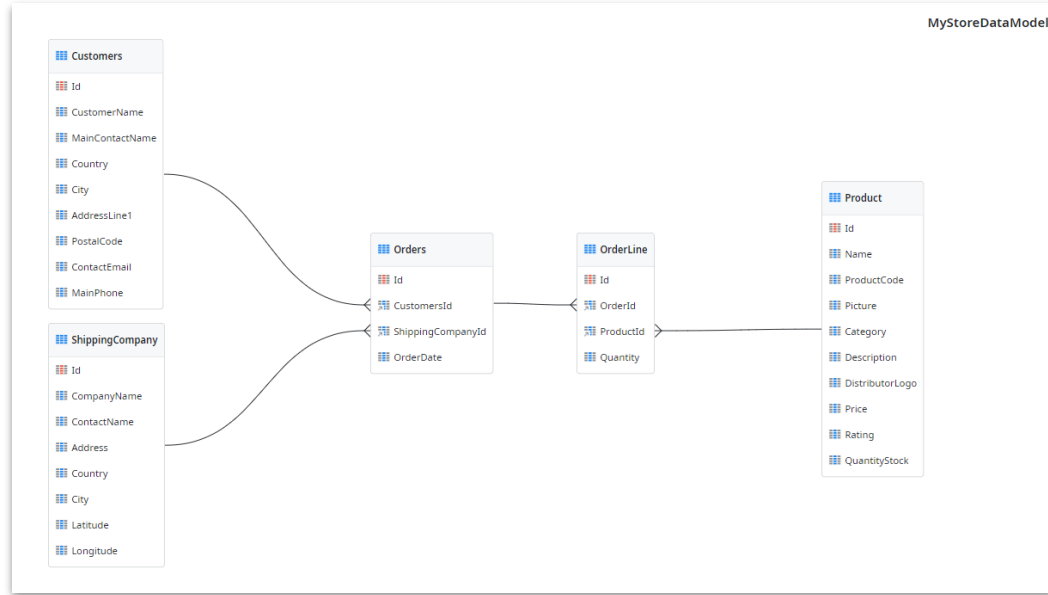
Note that by creating attributes with an existing entity suffix, the data type is automatically mapped to an entity identifier (e.g. the “CustomerId” attribute is automatically defined as an identifier of the Customer entity).

**Please make sure that your data model, including the attributes, are correctly created.**

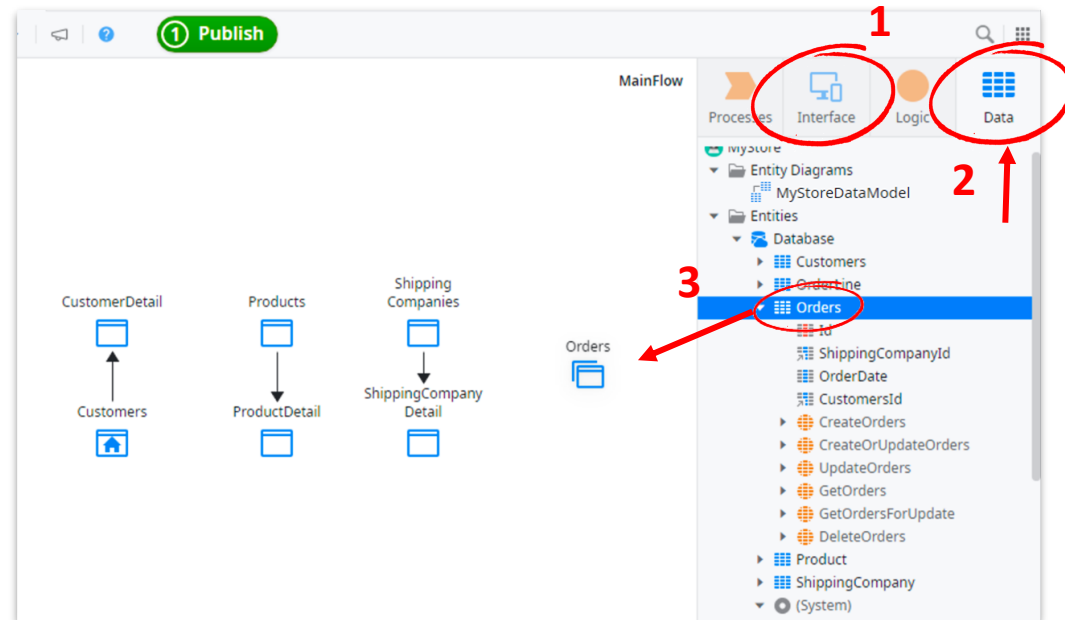


1. Open the **“MyStoreDataModel” Entity Diagram**;
2. Drag the **“Orders” entity** to the **“MyStoreDataModel” diagram**;
3. Drag the **“OrderLine” entity** to the **“MyStoreDataModel” diagram**;
4. Drag the **“ShippingCompany” entity** to the **“MyStoreDataModel” diagram**.

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

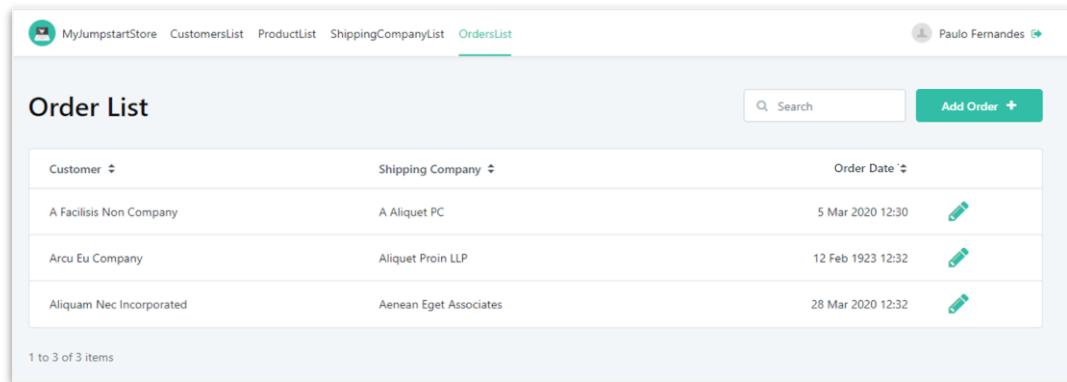


1. Open the **"MainFlow"** on the **"Interface"** Tab;
2. Go to the **"Data"** tab;
3. Drag the **"Orders"** entity to the **"MainFlow"** to create both pages (list and edit).



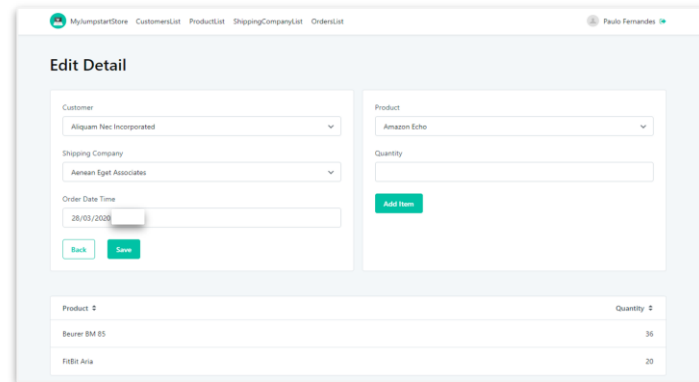
The accelerators are a great way to get a lot of functionality very fast. For simple applications that might be exactly what you need. More often than not you will want to adapt the generated pages to exactly your needs. You can easily do this, again in a visual fashion.

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.



The screenshot shows the 'Order List' page. At the top, there is a navigation bar with 'MyJumpstartStore', 'CustomersList', 'ProductList', 'ShippingCompanyList', and 'OrdersList' (highlighted). A user profile 'Paulo Fernandes' is visible in the top right. Below the navigation, there is a search bar and an 'Add Order +' button. The main content is a table with three columns: 'Customer', 'Shipping Company', and 'Order Date'. Each row has a green pencil icon for editing. At the bottom left, it says '1 to 3 of 3 items'.

Customer	Shipping Company	Order Date
A Facilis Non Company	A Aliquet PC	5 Mar 2020 12:30
Arcu Eu Company	Aliquet Proin LLP	12 Feb 1923 12:32
Aliquam Nec Incorporated	Aenean Eget Associates	28 Mar 2020 12:32



The screenshot shows the 'Edit Detail' page. At the top, there is a navigation bar with 'MyJumpstartStore', 'CustomersList', 'ProductList', 'ShippingCompanyList', and 'OrdersList'. A user profile 'Paulo Fernandes' is visible in the top right. The page has two main sections. The left section contains a 'Customer' dropdown (selected: 'Aliquam Nec Incorporated'), a 'Shipping Company' dropdown (selected: 'Aenean Eget Associates'), and an 'Order Date Time' input field (value: '28/03/2020'). There are 'Back' and 'Save' buttons. The right section contains a 'Product' dropdown (selected: 'Amazon Echo') and a 'Quantity' input field. There is an 'Add Item' button. At the bottom, there is a table with two columns: 'Product' and 'Quantity'.

Product	Quantity
Beurre BM 05	36
FaBib Aria	20

## Section 2

### C

## 1. Remove link to Customer detail

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.



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



The screenshot shows the software interface with the 'Orders' screen. The 'Interface' tab is selected in the top right. A context menu is open over a 'Link' widget in the 'Customer' column. The 'Remove Link' option is highlighted in the menu. The breadcrumb path at the bottom of the main view is visible, showing the path from the container to the widget.

## Section 2

### C

## 2. Add new column to the orders table

The screenshot shows a web application editor interface. The main area displays a table titled "Order List" with columns for "Customers" and "Shipping Company". The "Order Date" column header is circled in red with a red "1" next to it. A tooltip "Add New Column Right" is visible above the table. A green "Add Order +" button is also visible. The right sidebar shows a widget tree with "Orders" selected.

Customers	Shipping Company	Order Date
Justo LLC	Arcu Morbi Inc.	OrderDate
Justo LLC	Arcu Morbi Inc.	OrderDate
Justo LLC	Arcu Morbi Inc.	OrderDate

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 screenshot shows a software development environment with a table titled "Order List". The table has columns for "Customers", "Shipping Company", and "Order Date". The first row is a header row. The second row contains data: "Justo LLC", "Arcu Morbi Inc.", and "OrderDate". A red arrow points from the "Icon" widget in the toolbox to the "OrderDate" cell in the second row. A "Pick an Icon" dialog box is open, showing a search bar with "pencil" entered and a list of icons including "pencil". The "pencil" icon is selected. The "Select" button is highlighted.

**i** 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”**.

The screenshot illustrates the steps to link a pencil icon to an order detail page. It shows a table with 'Order Date' columns, a 'Select Icon' menu, a 'Link to' menu, and a 'Properties' tab for the link.

**1** Right-click the pencil icon in the table.

**2** Choose **Link to** > **MainFlow\OrderDetail**.

**3** In the **Properties** tab, set the **OrderId** value to **GetOrders.List.Current.Orders.Id**.

**Properties**

Property	Value
Name	
Confirmation ...	
Enabled	True
Visible	True
Style Classes	

**Attributes**

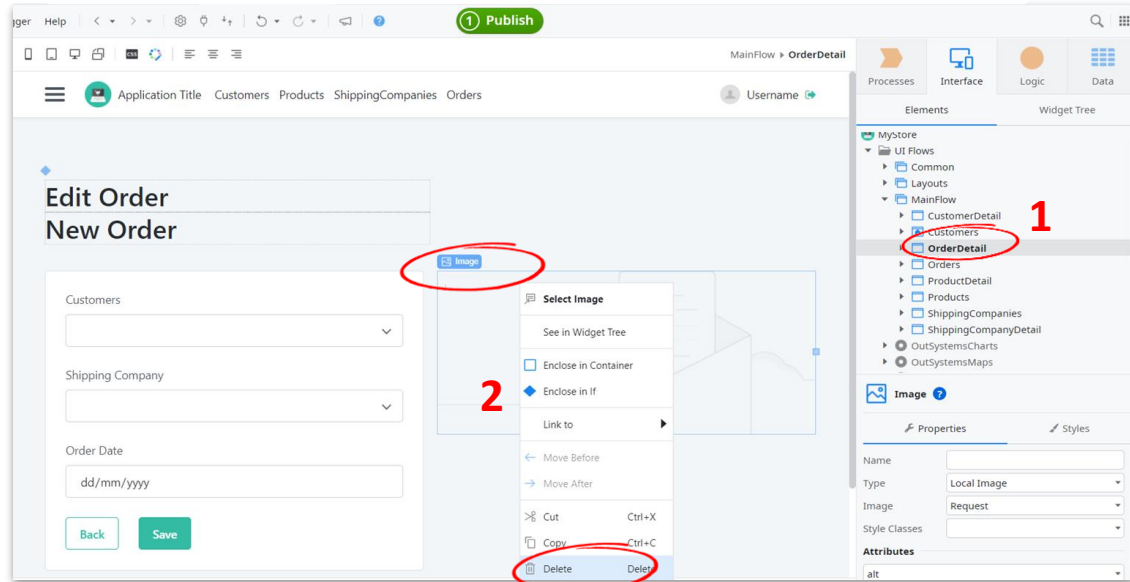
Property	Value
Property	
Value	

**Events**

Event	Handler
On Click	MainFlow\OrderDetail
OrderId	GetOrders.List.Current.Orders.Id
(New Argu...	

**i** 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.

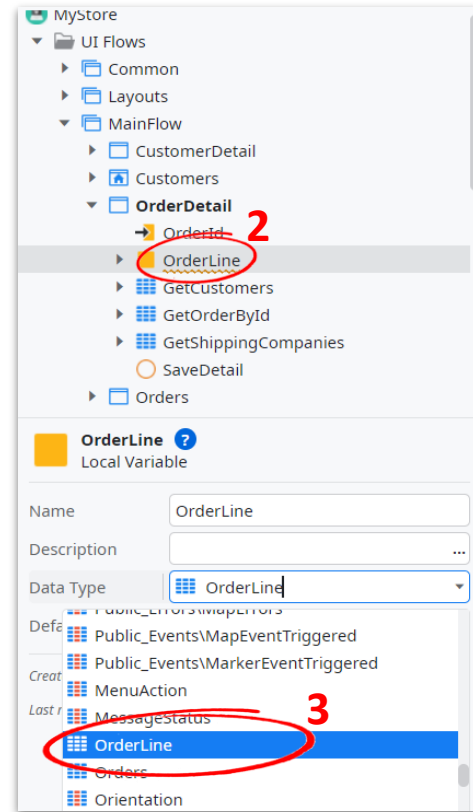
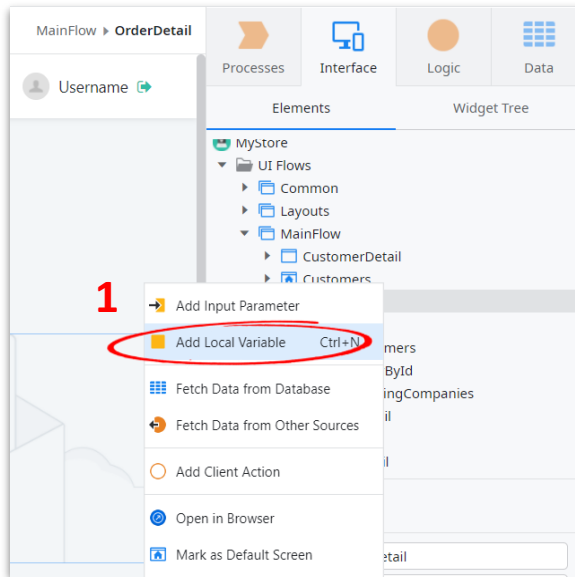
## Section 2 > C > 5. Delete Image



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”**.

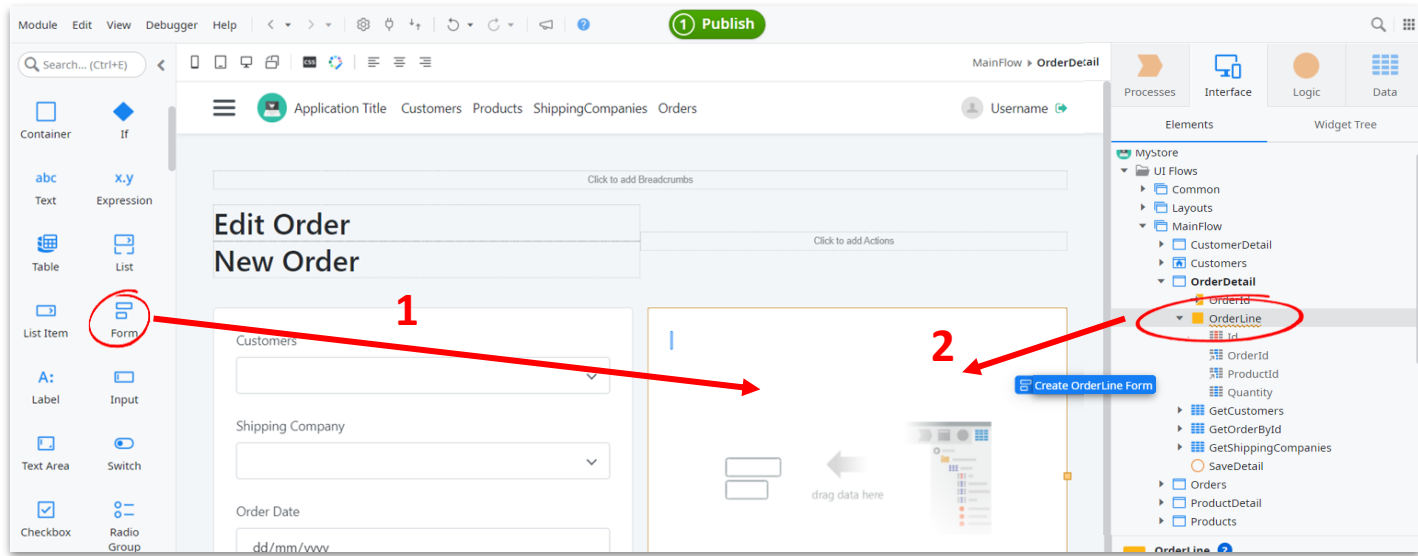


You can also use the keyboard shortcut: **Ctrl+N** to create a local variable



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

## Section 2 > C > 7. Add form to order detail screen



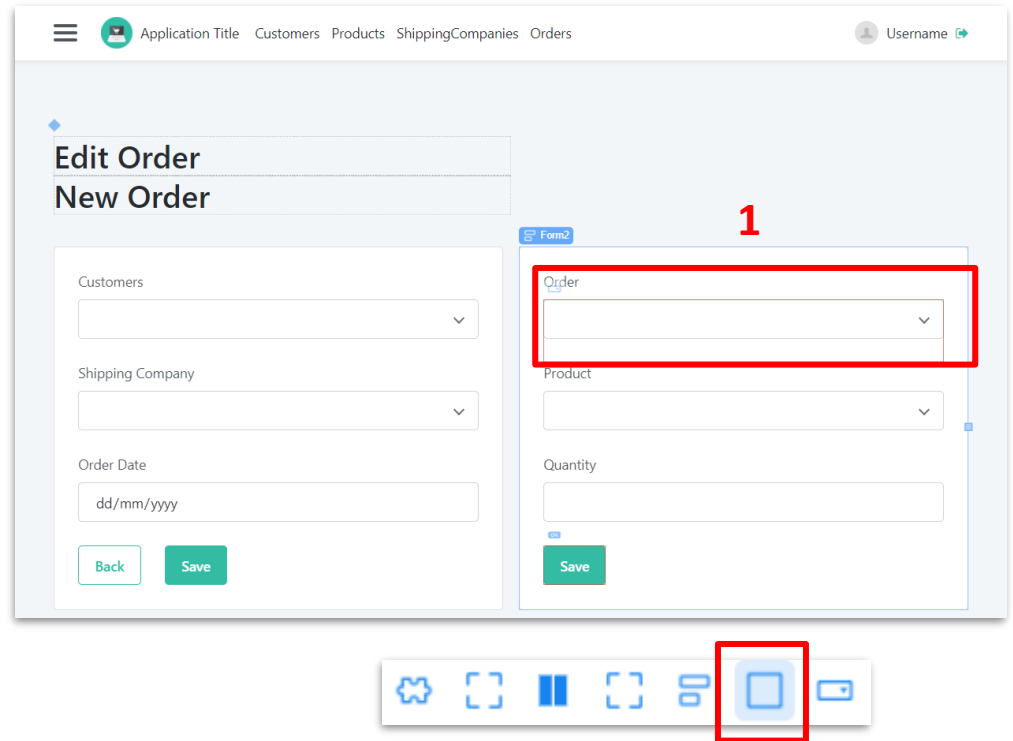
1. Drag the **“Form”** widget, from the toolbox pane in the left, to the column space in the right. This will create a blank form;
2. Then **drag the “OrderLine”** local variable to the **“Form”** widget. This will automatically add to the form the **“OrderLine”** local variable fields.

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

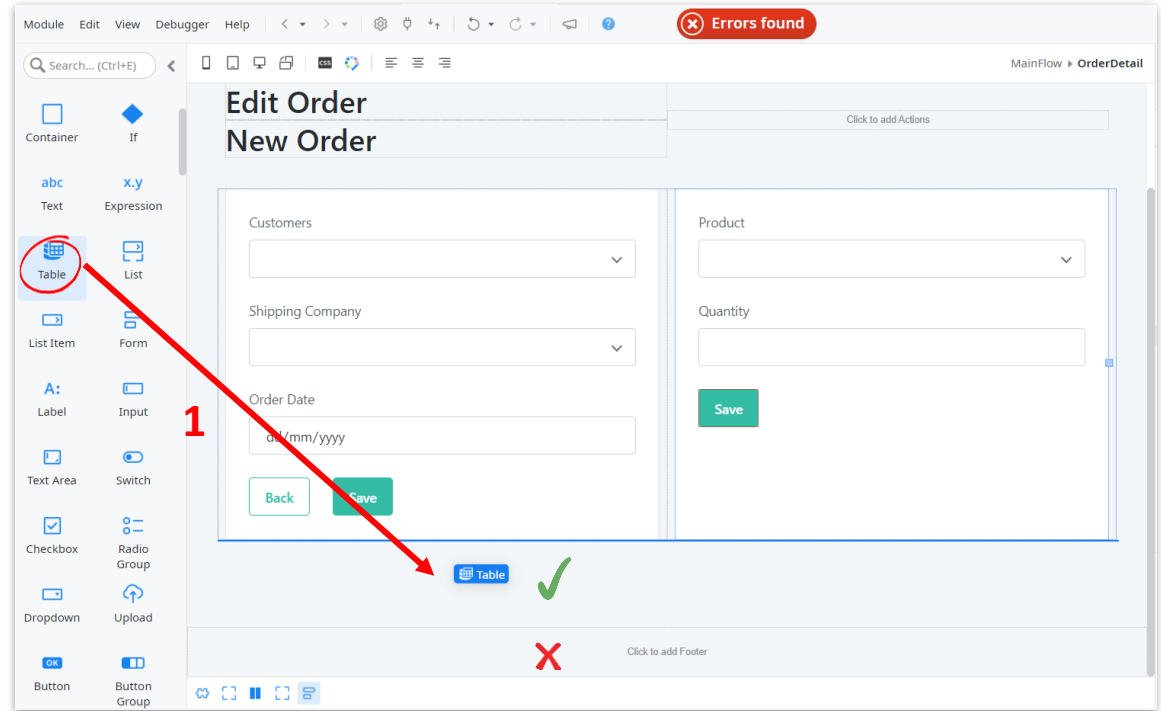
1. Delete the “Order” Label and Dropdown (delete the container wrapping both elements).



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.

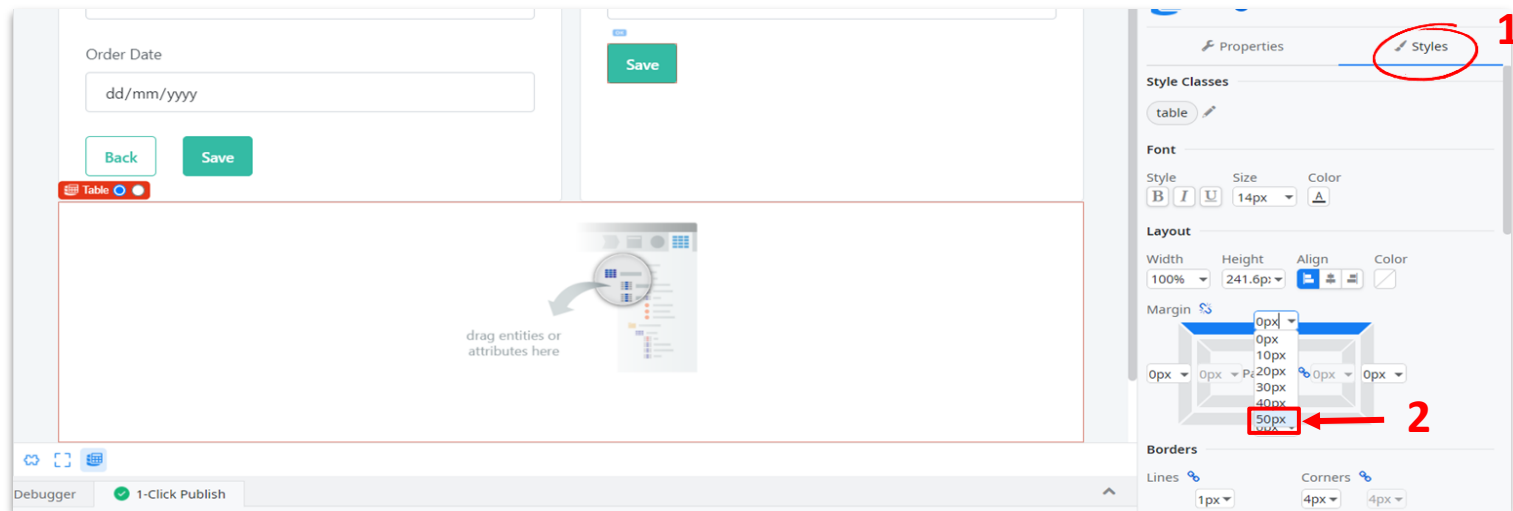


## Section 2 > C > 10. Format Table

Add some space between the table and the forms

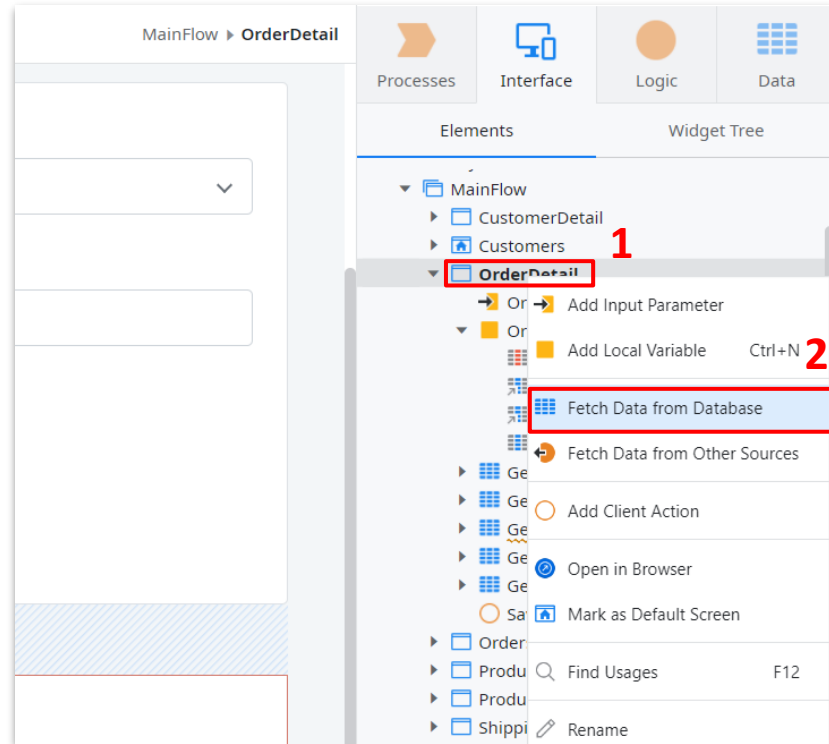
You now have table below the forms. But there's no space in between these elements. Let's add that space:

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”**;

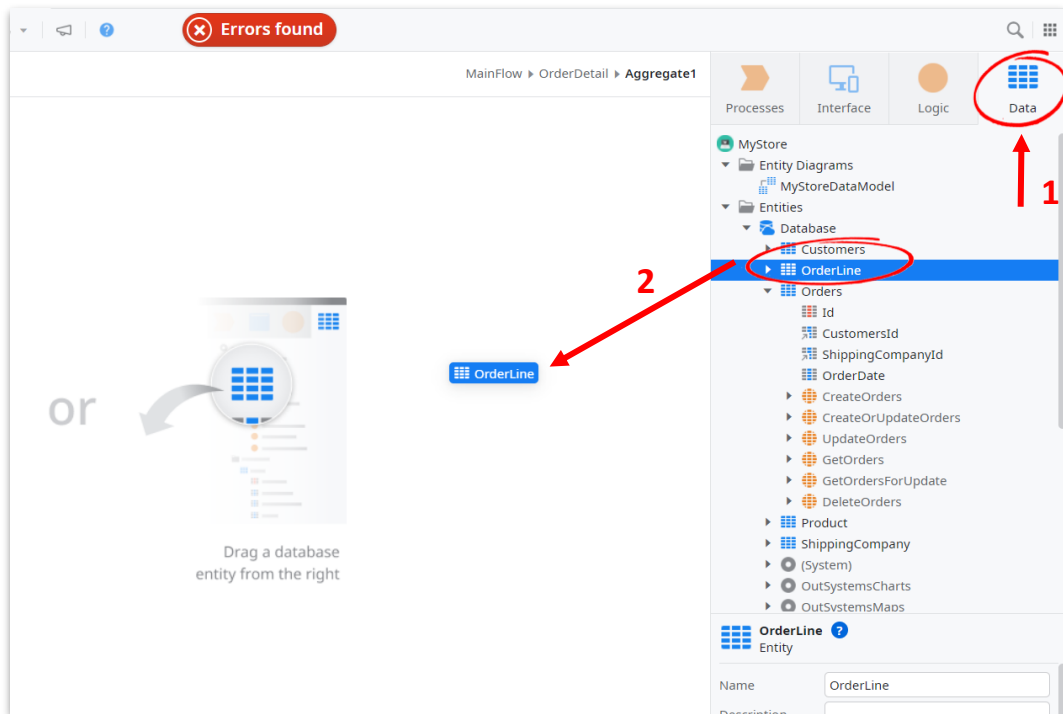


1. Navigate to the **“Data”** tab;
2. Drag the **“OrderLine”** Entity to the new **Aggregate** to fetch information from the **“OrderLine”** Entity.



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.



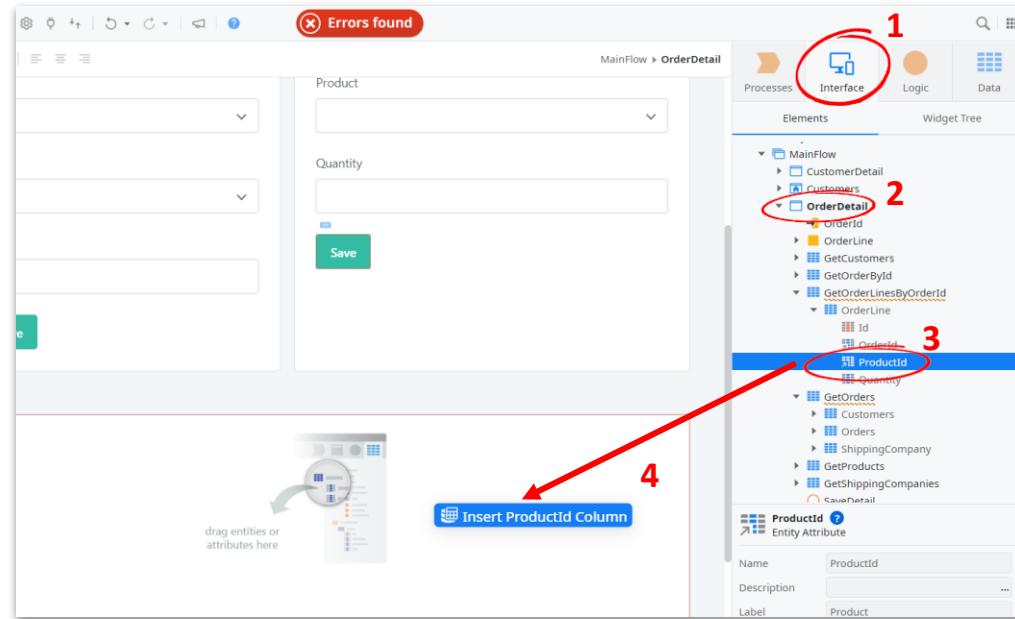
1. Click on the **"No Filters"** tab;
2. Click **"Add filter"**;
3. Enter the following query:  
*"Orderline.OrderId = OrderId"*;
4. Click **"Close"**.



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.

The screenshot shows the OutSystems query editor interface. The main window displays the 'GetOrderLines' aggregate with tabs for '1 Source', 'No Filters', 'No Sorts', and 'No Test Values'. The 'No Filters' tab is selected and circled in red with a '1'. Below the tabs, the 'Add filter' button is circled in red with a '2'. A dialog box titled 'OrderLine.OrderId = OrderId Condition' is open, showing the filter expression 'OrderLine.OrderId = OrderId' entered in the text field, which is circled in red with a '3'. Below the text field, there is a confirmation message 'The expression is ok (Type: Boolean)' and a set of operators. At the bottom right of the dialog, the 'Close' button is circled in red with a '4'.

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 screenshot displays a software development tool interface for a form titled "MainFlow > OrderDetail". The form contains several input fields: "Customers", "Shipping Company", "Order Date" (with a date format "dd/mm/yyyy"), "Product", and "Quantity". Below the form is a table with the following content:

Product
Withings Pulse OX
Withings Pulse OX
Withings Pulse OX

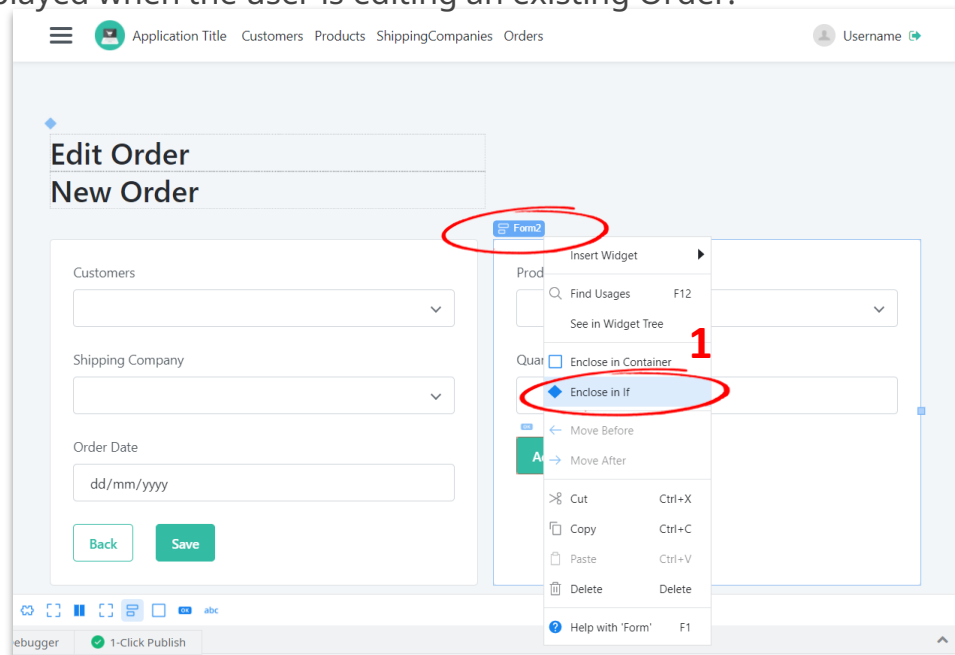
The interface also features a "Widget Tree" on the right side. The tree shows a hierarchy of elements, including "MyStore", "UI Flows", "Common", "Layouts", "MainFlow", "CustomerDetail", "Customers", "OrderDetail", "OrderLine", "GetCustomers", "GetOrderById", "GetOrderLinesByOrderId", "OrderLine", "Id", "OrderId", "Product", "Quantity", "Name". The "Quantity" attribute is highlighted with a red circle. A red arrow points from the "Quantity" attribute to the "Insert Quantity Column" button in the table, which is also highlighted with a red circle and labeled with a red "1". The "Add Item" button in the form is also highlighted with a red circle and labeled with a red "2".

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.



The screenshot illustrates the process of adding a condition to an 'If' widget in a Telerik UI editor. The main canvas shows a form with 'Product' and 'Quantity' fields and an 'Add Item' button. The 'Widget Tree' on the right shows the 'If' widget selected. The 'Properties' tab for the 'If' widget shows the 'Condition' dropdown set to 'Expression Editor...'. The 'Expression Editor' dialog is open, showing the expression 'OrderId <> NullIdentifier()' and a 'Close' button. The 'If Condition' dialog also shows the expression and a 'Close' button. Red circles and numbers 1, 2, and 3 highlight the 'Expression Editor...' button, the expression input, and the 'Close' button respectively.

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).

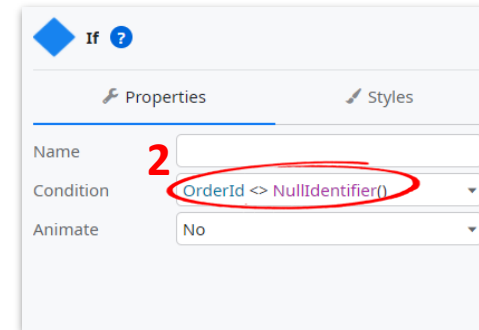
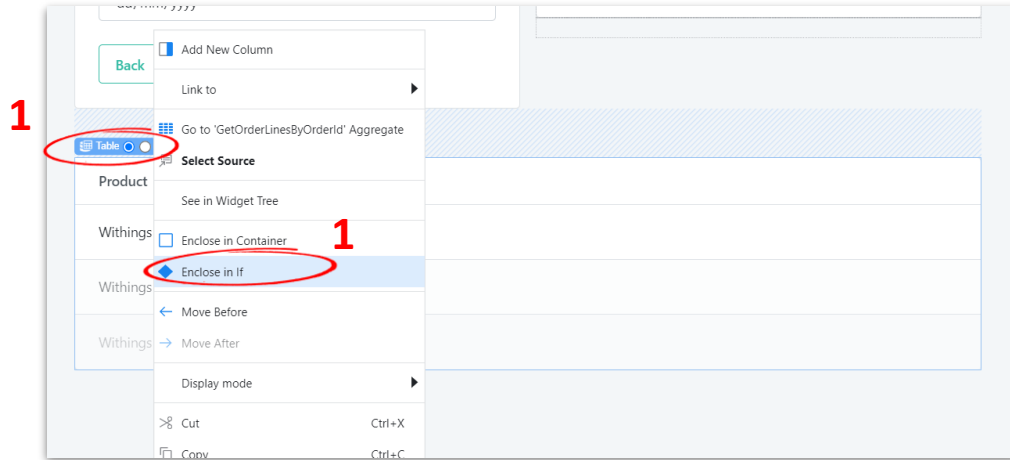


QUICK TIP

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()"*.



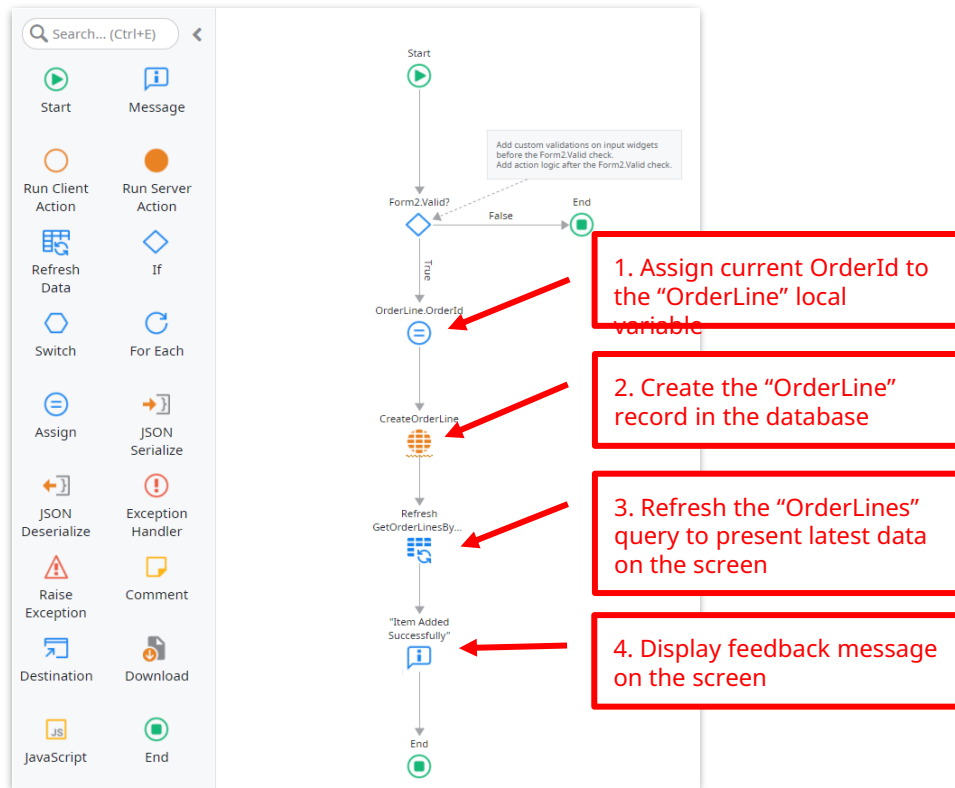
## Section 2

### 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.

The top screenshot shows a form with fields for Shipping Company, Quantity, and Order Date. A red circle highlights the 'Add Item' button, with a red '1' next to it. The bottom screenshot shows the 'AddItemOnClick' Client Action logic flowchart. A red box highlights the flowchart, and a red '2' is next to it. The flowchart starts with a 'Start' node, followed by a 'Form2.Valid?' decision diamond. If 'True', it goes to an 'End' node. If 'False', it goes to another 'End' node. A tooltip above the flowchart says: 'Add custom validations on input widgets before the Form2.Valid check. Add action logic after the Form2.Valid check.' The right-hand side of the bottom screenshot shows the 'Widget Tree' with 'AddItemOnClick' circled in red.

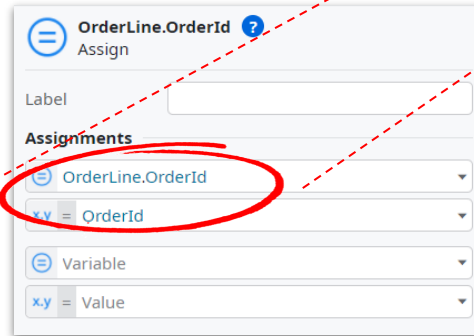
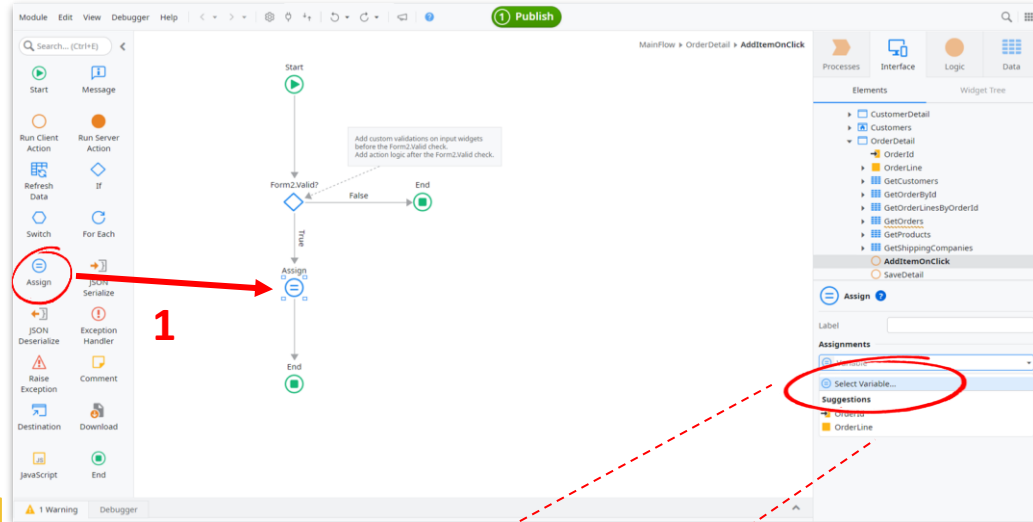
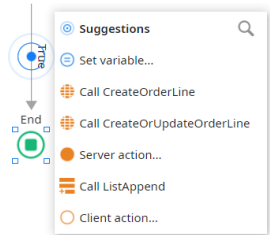
## Section 2 >> D >> 2. Add Assign Activity to Flow

1. Drag an **“Assign”** node to the **“AddItemOnClick”** action flow;
2. In the properties of the assign node, set the Variable of the assign node 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

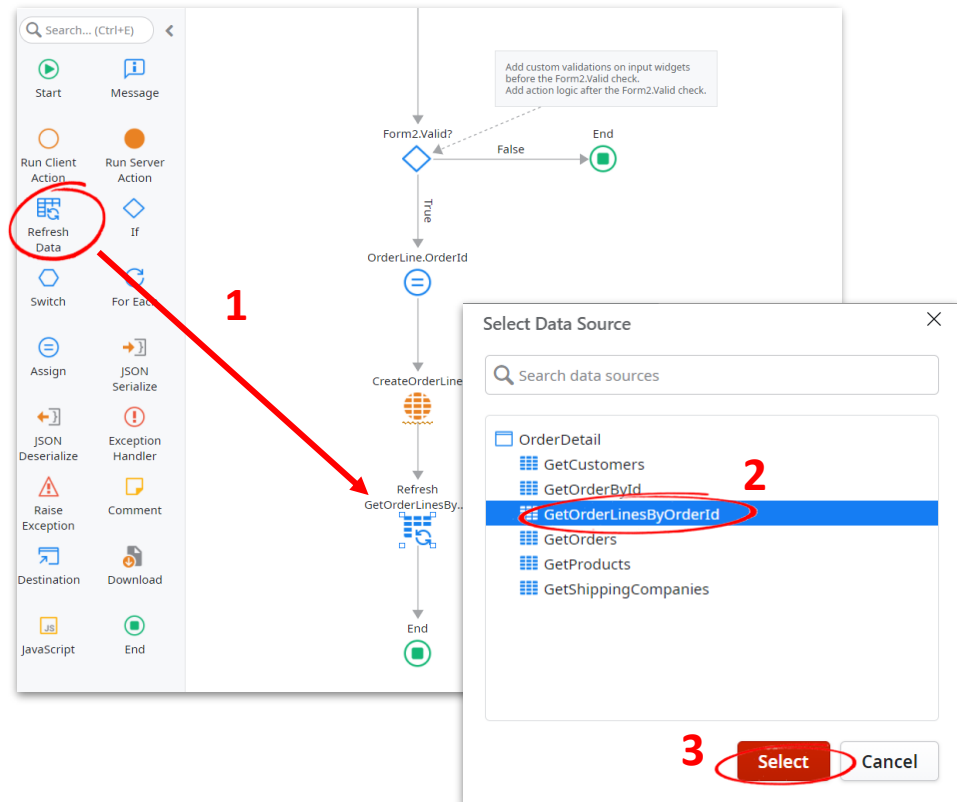
1. Navigate to the **“Data”** tab;
2. Drag the **“CreateOrderLine”** Server Action from the **“OrderLine”** Entity below the assign activity that you have just created;
3. Set the source value to the **“OrderLine”** attribute.



## Section 2 > D > 4. Refresh Screen Data

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.



## Section 2 > D > 5. Add Feedback Message

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.

The screenshot displays the Oracle APEX IDE interface. On the left, the 'Widgets' panel shows the 'Message' widget circled in red, with a red arrow labeled '1' pointing to its placement in the 'AddItemOnClick' action flow. The flow includes steps: True, OrderLine.OrderId, CreateOrderLine, Refresh GetOrderLinesBy..., and 'Item Added Successfully'. On the right, the 'Properties' panel for the 'Message' widget is shown, with the 'Type' dropdown set to 'Info' circled in red, and a red arrow labeled '2' pointing to it. The 'Message' text field contains 'Item Added Successfully'.

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).

The screenshot shows the Development Studio interface. The top menu bar includes Module, Edit, View, Debugger, and Help. A red circle highlights the 'Publish' button in the top right corner. The main canvas displays a flowchart with the following steps: True, OrderLine.OrderId, CreateOrderLine, Refresh GetOrderLinesBy..., 'Item Added Successfully', and End. The right sidebar shows a widget tree with 'AddItemOnClick' selected. A message box at the bottom right displays 'Item Added Successfully'.

## End of Section 2 - Product & Shipping

MyStore Customers Products ShippingCompanies Orders Luis Marques

### Order List

Search Add Order +

Customers	Shipping Company	Order Date
A Facilis Non Company	A Aliquet PC	27 Jul 2022
Arcu Eu Company	Aliquet Proin LLP	
Aliquam Nec Incorporated	Aenean Eget Associates	

1 to 3 of 3 items



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.

**Jump Start Training**

MyStore Customers Products ShippingCompanies Orders Luis Marques

### Edit Order

Customers: Aliquam Nec Incorporated

Shipping Company: Aenean Eget Associates

Order Date: 29/07/2022

Product: FITBit Aria

Quantity: 20

Back Save Add Item

Product	Quantity
Beurer BM 85	
FITBit Aria	

# Section 3

Image Gallery with fluid layout



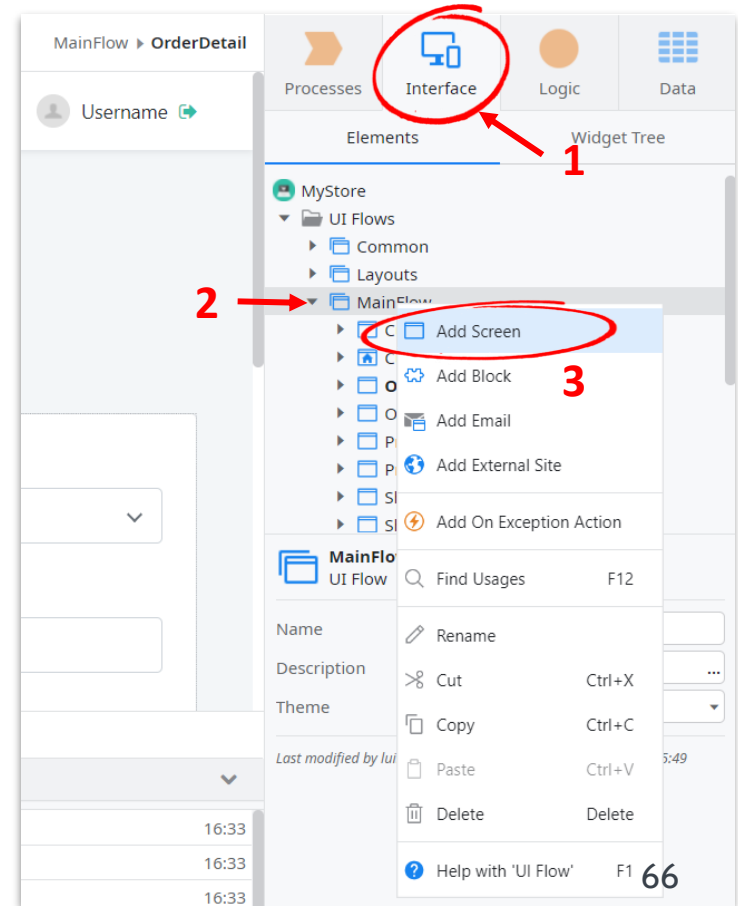
- | The platform provides screen templates to help developers create world-class user interfaces easily. Not only do the templates contain UI design, but it also has pre-created actions to accelerate development of the page actions such as filters.
- | OutSystems UI templates pre-creates your screens 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.

## Section 3

### A

## 1. New Web Screen

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



## Section 3

A

## 2. Select Template Screen

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

New Screen

Employees directory Employees list and detail

Search template

1

Four column gallery

Inspection Detail

Invoice Detail

List with chart

Location detail

Onboarding

Four column gallery

Give us feedback

Product Overview

Screen name: Homepage

2

Screen composition

Product gallery filters

3

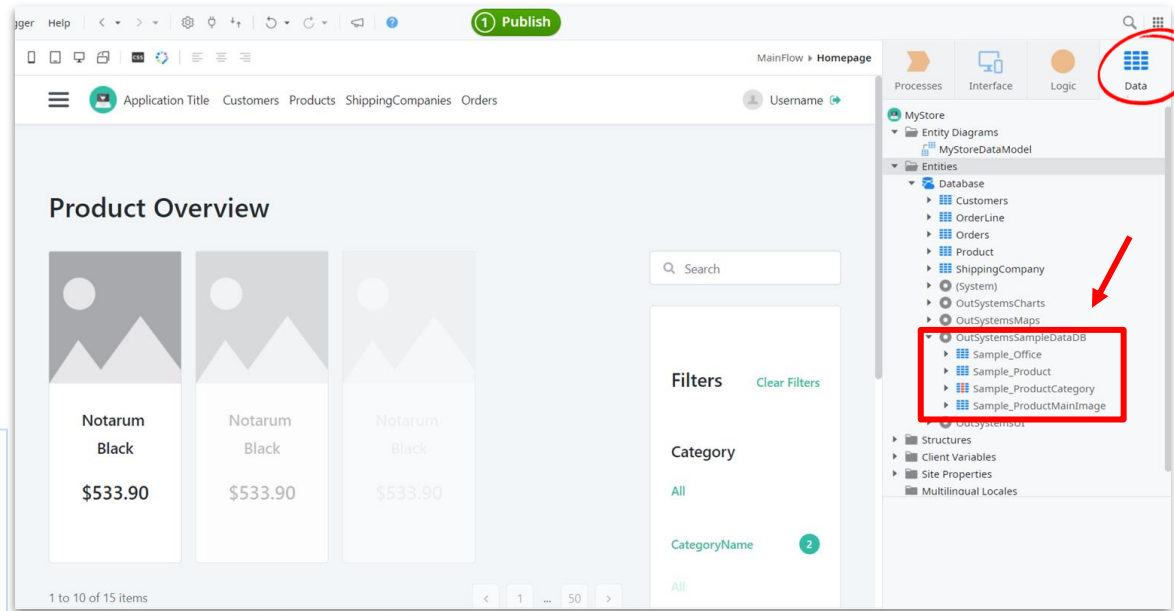
Create Screen Cancel

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.

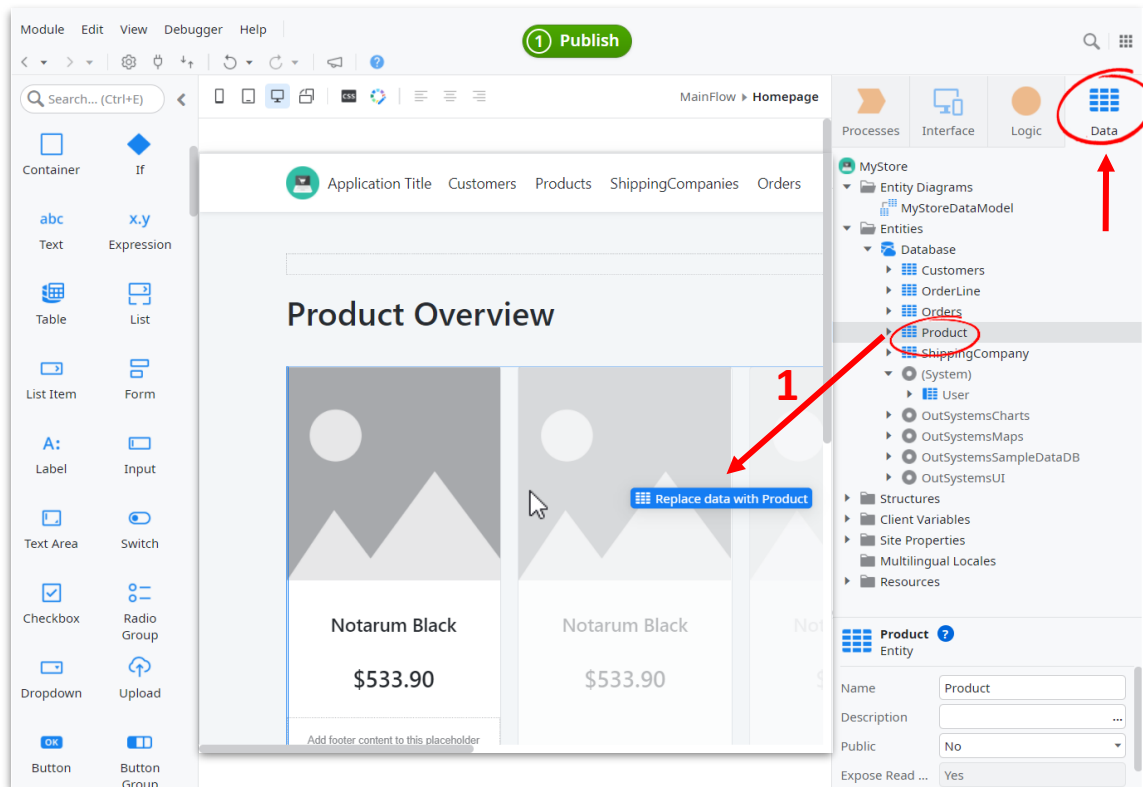


## Section 3 > B > 1. Add Gallery Web Pattern

### 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.



The screenshot shows the development environment with the 'Product Overview' screen. A red arrow points to a button labeled 'Replace data with Product' on the screen. Another red arrow points to the 'Data' menu item in the top right corner of the interface. The 'Product' entity is highlighted in the right-hand pane. The 'Product Overview' screen displays a list of products, including 'Notarum Black' with a price of '\$533.90'.

## 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”**.

The screenshot shows a web application editor interface. On the left, a sidebar contains various widgets, with the 'Image' widget highlighted by a red circle. A red arrow labeled '1' points from this widget to an 'Image' placeholder on a 'Product Overview' page. The page displays three product cards, each with the text 'Withings Pulse OX' and '213'. Below the cards, it says '1 to 10 of 15 items'. On the right, a properties panel for the 'Image' widget is shown. The 'Type' dropdown is set to 'External URL', which is circled in red. The 'Value' field contains the expression 'GetProducts.List.Current.Product.Picture', which is also circled in red. A red circle labeled '2' is placed near the suggestions list in the 'Value' field.

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.

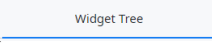
You can customize any template since it's just a starting point that gives you most of what you needed.

## Section 3

### C

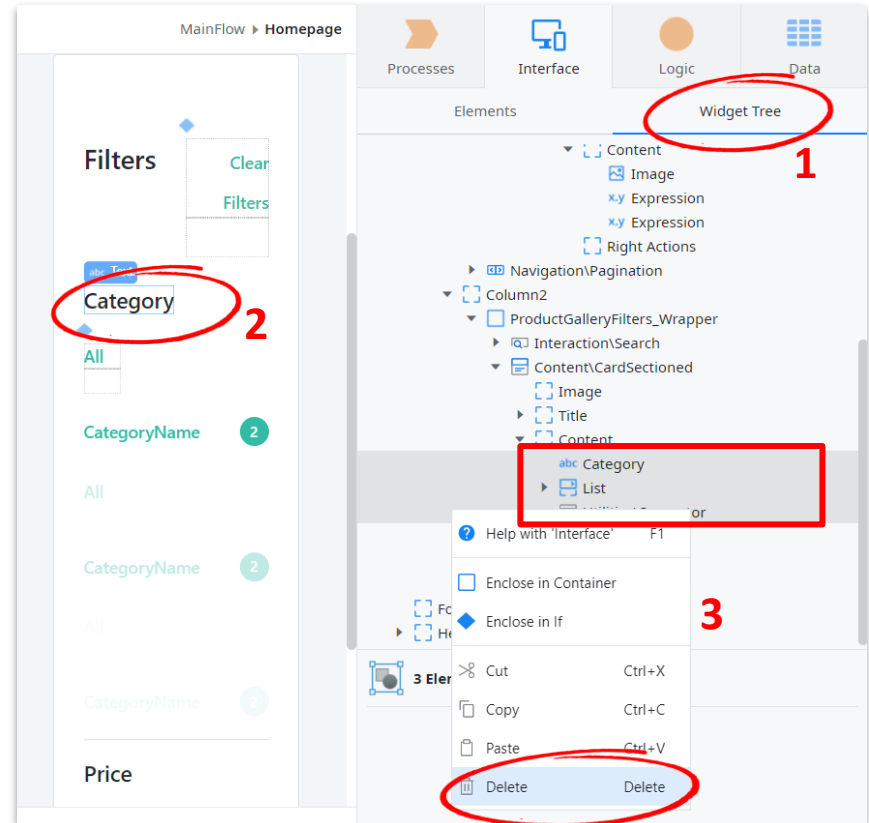
## 1. Delete Filters

### Delete Category Filters

1. Click to open the **“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”.



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

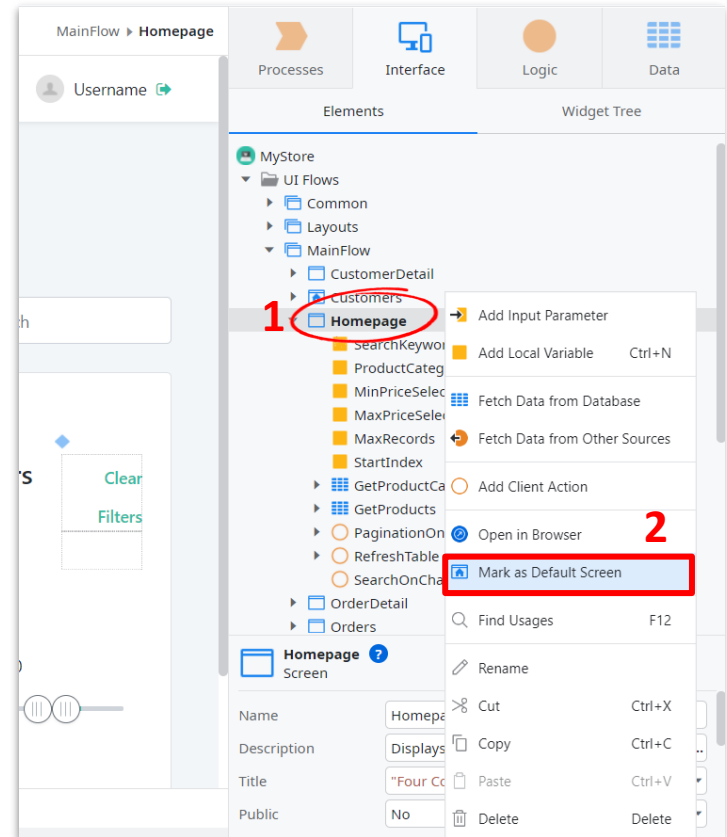


## Section 3

### C

## 2. Make Homepage the Default Screen

1. Set this page as the Entry Point by right-clicking the **“Homepage”** screen;
2. And selecting the **“Mark as Default Screen”** option.

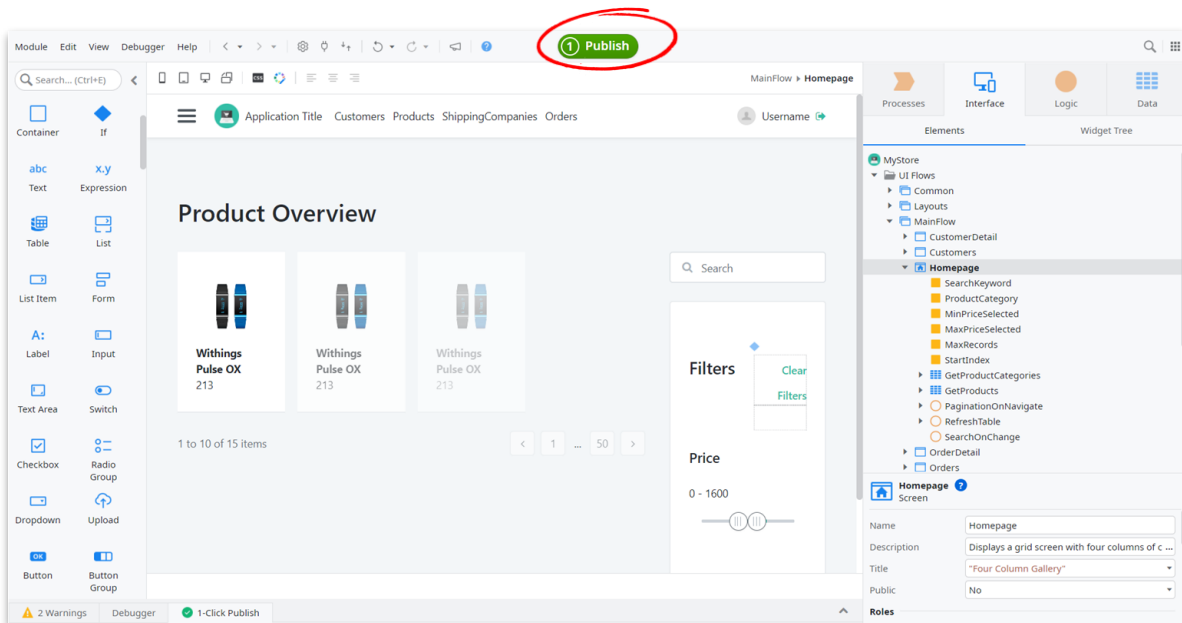


1. Click the “1-Click Publish” to initiate the deployment process.

*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.*




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


# End of Section 3 - Image Gallery with fluid layout

MyStore Customers Products ShippingCompanies Orders Luis Marques


## Product Overview




**Amazon Echo**  
151




**Automaticant**  
152




**Beurer AS 80**  
168




**Beurer BF 700**  
169




**Beurer BM 85**  
170



**Connected by TCP**  
171



**Connected Cree® LED Bulb**  
172




**Easybulb**  
173

**Filters** [Clear Filters](#)

**Price**

0 - 1600



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



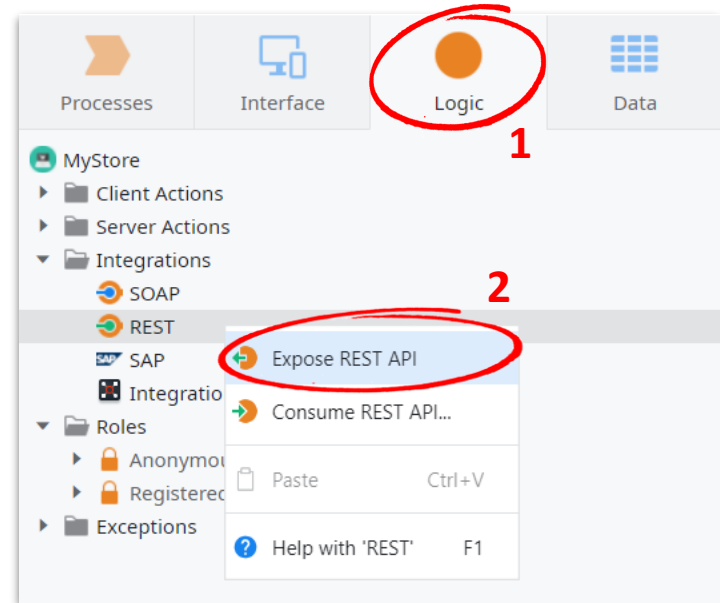
# Section 4

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.

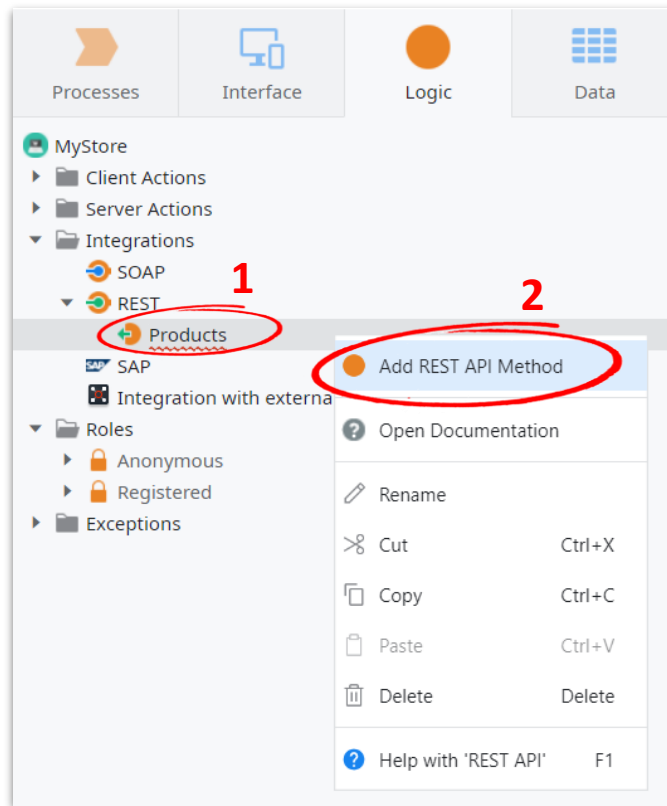
Please note: If you wanted to consume a service from another module, there is a much easier and standard way to do this. Your instructor will demonstrate how to do this later in the day.

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



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”**.

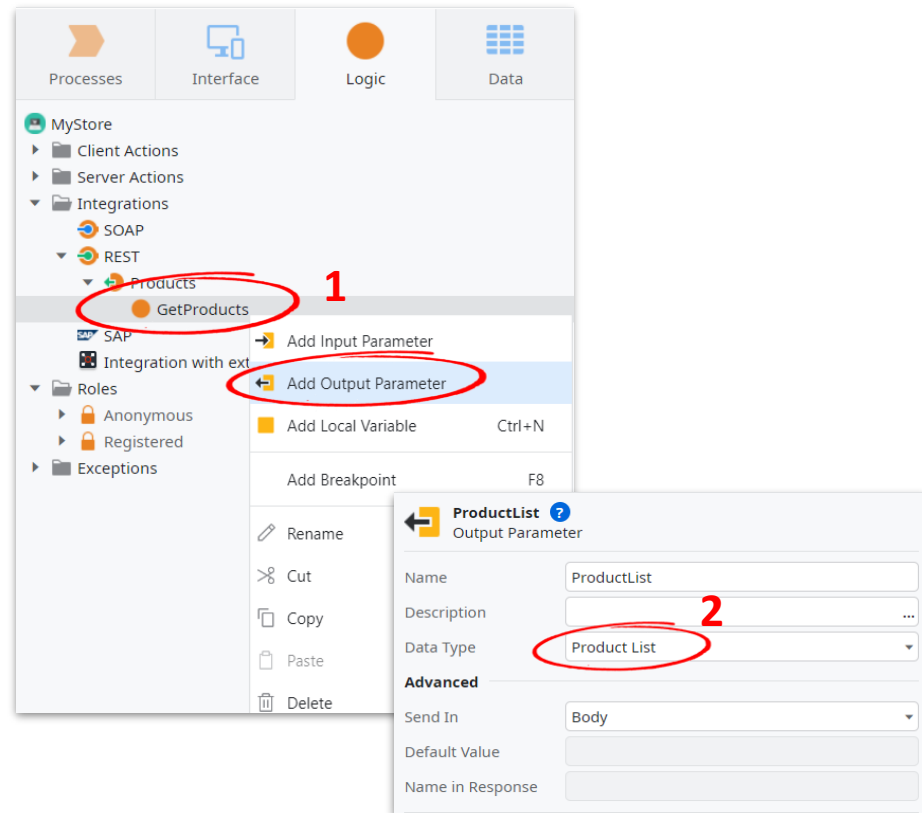
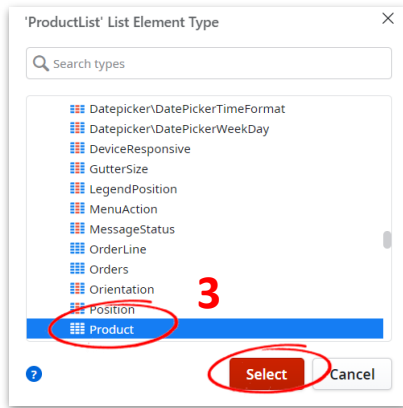
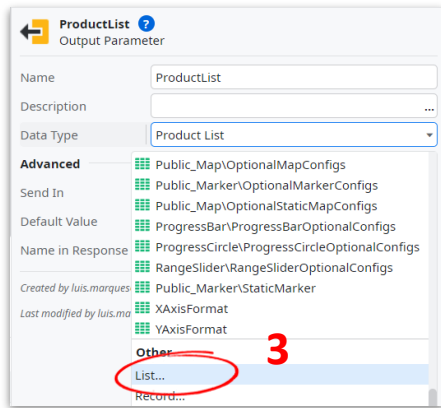


## Section 4

### A

## 3. Expose the product catalog as a REST API

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**”.

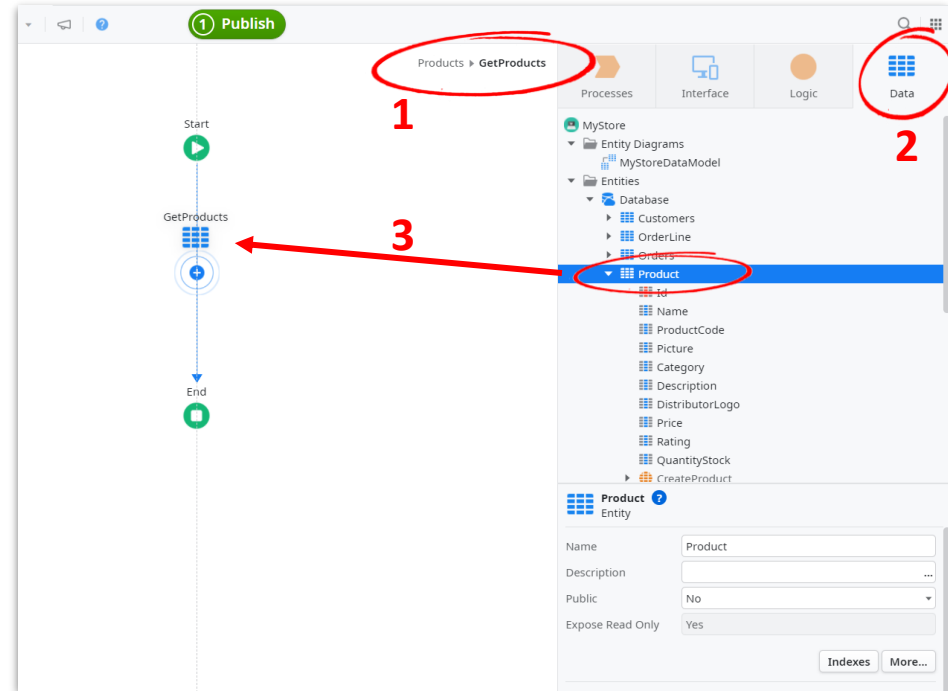


We have created the Web Service and one method. Let's fill the method with logic in a low code way.

## Section 4 > B > 1. REST API Implementation

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.



## Section 4 > B > 2. REST API Implementation

### Assign Output Parameter

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



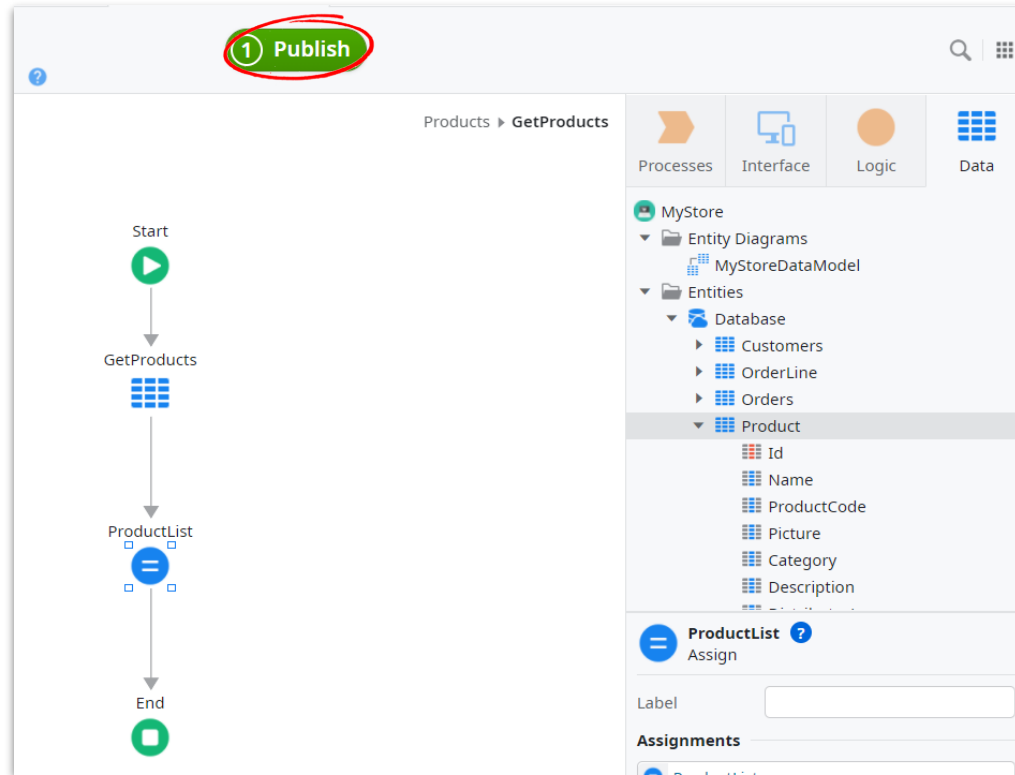
QUICK TIP

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.

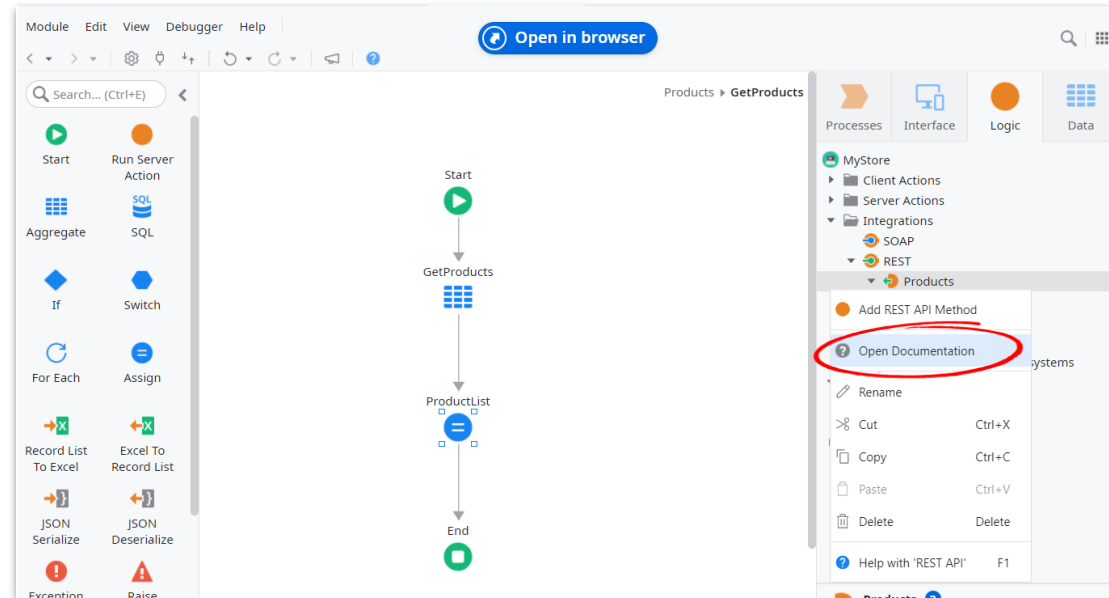
More information about AI Assisted Development [here](#)

The screenshot displays the OutSystems development environment. The main canvas shows a flowchart with the following nodes: Start, GetProducts, ProductList, and End. A red circle highlights the 'Assign' node in the left-hand palette, with a red arrow labeled '1' pointing to the 'ProductList' node. A secondary window titled 'ProductList Assign' is open, showing a dropdown menu where 'x.y = GetProducts.List' is selected and circled in red, with a red arrow labeled '2' pointing to the 'x.y Expression Editor...' field below it.

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”**.



## 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.

## Section 4

### C

## 1. Expose the Product Entity as Public

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.

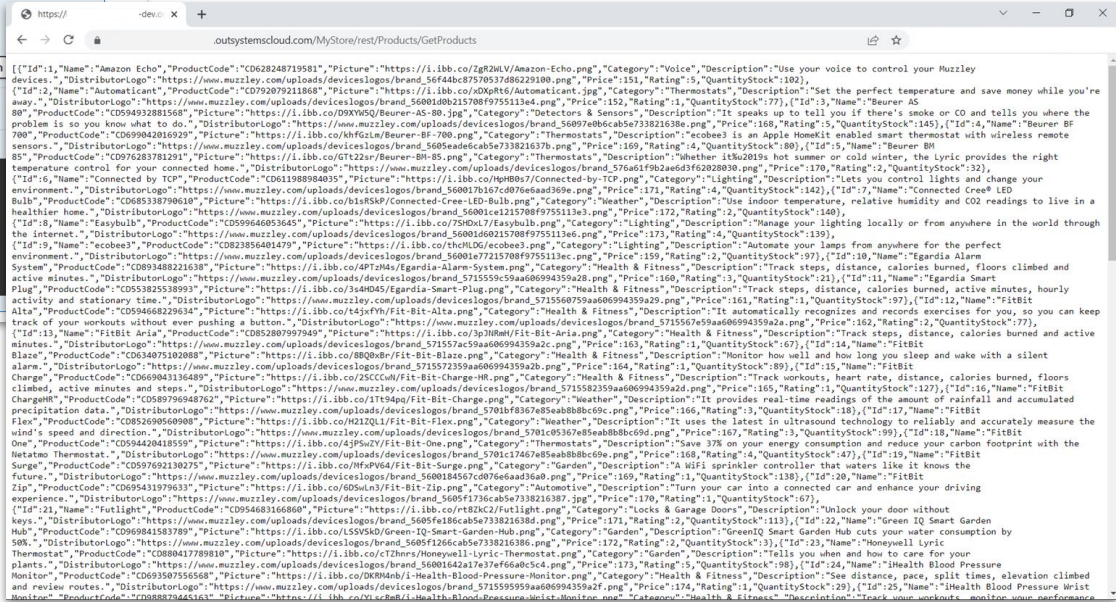
The screenshot shows the Microsoft Dynamics CRM interface. The 'Data' tab is selected in the top right corner. The 'Entity Diagrams' tree on the left shows the 'Product' entity selected. The 'Product' entity properties pane on the right shows the 'Public' property set to 'Yes'. A 'Publish' button is visible in the top right corner. Red annotations highlight the 'Publish' button (4), the 'Data' tab (1), the 'Product' entity (2), and the 'Yes' value in the 'Expose Read...' dropdown (3).

# End of Section 4 - REST API

## Products

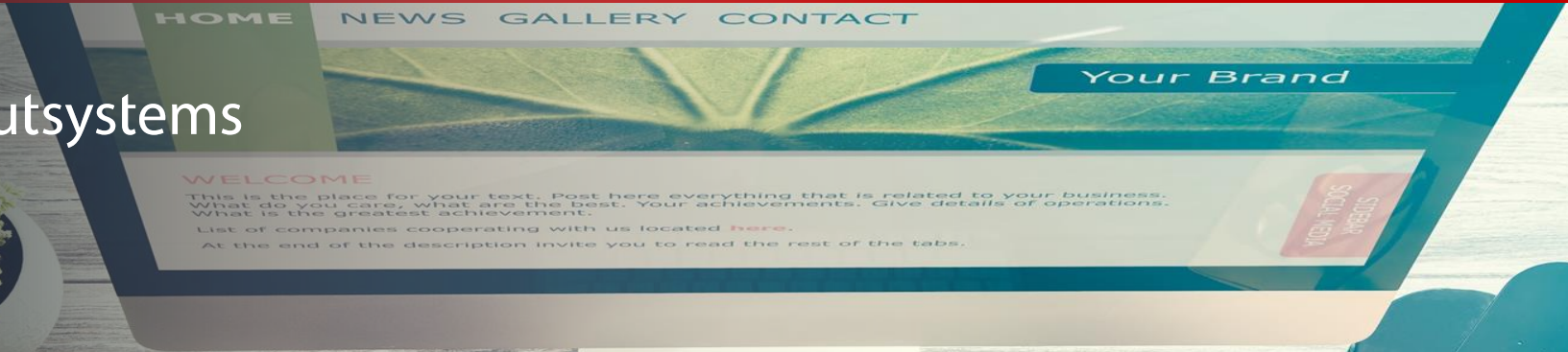
[ Base URL: /MyStore/rest/Products/ ]  
swagger.json

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)	<pre>Example Value   Model {   "id": 1234567891234567,   "name": "Amazon Echo",   "productCode": "C628248719581",   "picture": "https://i.ibb.co/2RqMLV/Amazon-Echo.png",   "category": "Voice",   "description": "Use your voice to control your Muzley devices.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_56f44bc87570537d62229100.png",   "price": 151,   "rating": 5,   "quantityStock": 1802,   "id": 2,   "name": "Automatic",   "productCode": "C0792079211888",   "picture": "https://i.ibb.co/xDqRt6/Automatic.jpg",   "category": "Thermostats",   "description": "Set the perfect temperature and save money while you're away.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_56089108215788795113e4.png",   "price": 152,   "rating": 1,   "quantityStock": 77,   "id": 3,   "name": "Beurer AS 80",   "productCode": "CD59493281566",   "picture": "https://i.ibb.co/D9XW5Q/Beurer-AS-80.jpg",   "category": "Detectors &amp; Sensors",   "description": "It speaks up to tell you if there's smoke or CO and tells you where the problem is so you know what to do.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_56097f68c4b5e73821638e.png",   "price": 168,   "rating": 5,   "quantityStock": 145,   "id": 4,   "name": "Beurer BF 700",   "productCode": "CD599842816539",   "picture": "https://i.ibb.co/4hKqLw/Beurer-BF-700.png",   "category": "Thermostats",   "description": "ecobee3 is an Apple HomeKit enabled smart thermostat with wireless remote sensors.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_5605e4e4e4b5e738216376.png",   "price": 169,   "rating": 4,   "quantityStock": 80,   "id": 5,   "name": "Beurer BH 85",   "productCode": "CD976283781291",   "picture": "https://i.ibb.co/GT22sr/Beurer-BH-85.png",   "category": "Thermostats",   "description": "Whether it's hot summer or cold winter, the Lyric provides the right temperature control for your connected home.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_576a1f9b2aed6376202830.png",   "price": 170,   "rating": 2,   "quantityStock": 32,   "id": 6,   "name": "Connected by TCP",   "productCode": "CD611088968035",   "picture": "https://i.ibb.co/0h4B8z/Connected-by-TCP.png",   "category": "Lighting",   "description": "Lets you control lights and change your environment.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_560017b167cd0766aad369e.png",   "price": 171,   "rating": 4,   "quantityStock": 142,   "id": 7,   "name": "Connected Cree® LED Bulb",   "productCode": "CD685338790610",   "picture": "https://i.ibb.co/01s18AP/Connected-Cree-LED-Bulb.png",   "category": "Weather",   "description": "Use indoor temperature, relative humidity and CO2 readings to live in a healthier home.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_56001e215708f975511e3.png",   "price": 172,   "rating": 2,   "quantityStock": 140,   "id": 8,   "name": "Easybulb",   "productCode": "CD599646053645",   "picture": "https://i.ibb.co/7SHdL7/5Hub.png",   "category": "Lighting",   "description": "Manage your lighting locally or from anywhere in the world through the internet.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_5600160215708f975511e6.png",   "price": 173,   "rating": 4,   "quantityStock": 130,   "id": 9,   "name": "ecobee3",   "productCode": "CD821895401479",   "picture": "https://i.ibb.co/thoDLC/ecobee3.png",   "category": "Lighting",   "description": "Automate your lamps from anywhere for the perfect environment.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_56001e77215708f975511e3.png",   "price": 159,   "rating": 2,   "quantityStock": 97,   "id": 10,   "name": "Egardia Alarm System",   "productCode": "CD89348221638",   "picture": "https://i.ibb.co/4PTzMa/Egardia-Alarm-System.png",   "category": "Health &amp; Fitness",   "description": "Track steps, distance, calories burned, floors climbed and active minutes.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_5715560739a606994359a29.png",   "price": 161,   "rating": 1,   "quantityStock": 97,   "id": 12,   "name": "FitBit Alta",   "productCode": "CD59468229634",   "picture": "https://i.ibb.co/t4jYfV/Fit-Bit-Alta.png",   "category": "Health &amp; Fitness",   "description": "It automatically recognizes and records exercises for you, so you can keep track of your workouts without ever pushing a button.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_5715560739a606994359a29.png",   "price": 162,   "rating": 2,   "quantityStock": 77,   "id": 13,   "name": "FitBit Aria",   "productCode": "CD852807997949",   "picture": "https://i.ibb.co/3pJlMh/Fit-Bit-Aria.png",   "category": "Health &amp; Fitness",   "description": "Track steps, distance, calories burned and active minutes.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_571557ac59a606994359a29.png",   "price": 163,   "rating": 1,   "quantityStock": 67,   "id": 14,   "name": "FitBit Blaze",   "productCode": "CD53407180288",   "picture": "https://i.ibb.co/8RQ08r/Fit-Bit-Blaze.png",   "category": "Health &amp; Fitness",   "description": "Monitor how well and how long you sleep and wake with a silent alarm.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_571557359a606994359a29.png",   "price": 164,   "rating": 1,   "quantityStock": 189,   "id": 15,   "name": "FitBit Charge",   "productCode": "CD609843136489",   "picture": "https://i.ibb.co/25CCuM/Fit-Bit-Charge-HR.png",   "category": "Health &amp; Fitness",   "description": "Track workouts, heart rate, distance, calories burned, floors climbed and steps.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_570167f83e78e8a8b88b0c9e.png",   "price": 165,   "rating": 1,   "quantityStock": 127,   "id": 16,   "name": "FitBit ChargeFX",   "productCode": "CD58795948762",   "picture": "https://i.ibb.co/1tE4qD/Fit-Bit-Charge.png",   "category": "Weather",   "description": "It provides real-time readings of the amount of rainfall and accumulated precipitation data.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_570167f83e78e8a8b88b0c9e.png",   "price": 166,   "rating": 3,   "quantityStock": 18,   "id": 17,   "name": "FitBit Flex",   "productCode": "CD85269560808",   "picture": "https://i.ibb.co/0Y2JL2/Fit-Bit-Flex.png",   "category": "Weather",   "description": "It uses the latest in ultrasound technology to reliably and accurately measure the wind's speed and direction.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_570167f83e78e8a8b88b0c9e.png",   "price": 167,   "rating": 3,   "quantityStock": 99,   "id": 18,   "name": "FitBit One",   "productCode": "CD594420418559",   "picture": "https://i.ibb.co/4JpSwZ/Fit-Bit-One.png",   "category": "Thermostats",   "description": "Save 37% on your energy consumption and reduce your carbon footprint with the Netatmo Thermostat.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_5701174678e8a8b88b0c9e.png",   "price": 168,   "rating": 4,   "quantityStock": 47,   "id": 19,   "name": "FitBit Surge",   "productCode": "CD59769113027",   "picture": "https://i.ibb.co/0F4P64/Fit-Bit-Surge.png",   "category": "Garden",   "description": "A wifi sprinkler controller that waters like it knows the future.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_5608184567cd0766aad36a0.png",   "price": 169,   "rating": 1,   "quantityStock": 138,   "id": 20,   "name": "FitBit Zip",   "productCode": "CD695431979633",   "picture": "https://i.ibb.co/0dW0u3/Fit-Bit-Zip.png",   "category": "Automotive",   "description": "Turn your car into a connected car and enhance your driving experience.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_5605f1736cab5e738216387.jpg",   "price": 170,   "rating": 1,   "quantityStock": 67,   "id": 21,   "name": "Futlight",   "productCode": "CD954683166860",   "picture": "https://i.ibb.co/r7BRK2C/Futlight.png",   "category": "Locks &amp; Garden Doors",   "description": "Unlock your door without keys.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_5605f186cab5e73821638a.png",   "price": 171,   "rating": 2,   "quantityStock": 113,   "id": 22,   "name": "GreenIQ Smart Garden Hub",   "productCode": "CD596841583789",   "picture": "https://i.ibb.co/L5SV5d0/Green-IQ-Smart-Garden-Hub.png",   "category": "Garden",   "description": "GreenIQ Smart Garden Hub cuts your water consumption by 50%.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_5605f126cab5e738216386.png",   "price": 172,   "rating": 2,   "quantityStock": 3,   "id": 23,   "name": "Honeywell Lyric Thermostat",   "productCode": "CD88043789810",   "picture": "https://i.ibb.co/cZlMns/Honeywell-Lyric-Thermostat.png",   "category": "Garden",   "description": "Tells you when and how to care for your plants.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_56081842427e376f66a0c54.png",   "price": 173,   "rating": 5,   "quantityStock": 98,   "id": 24,   "name": "iHealth Blood Pressure Monitor",   "productCode": "CD69350755658",   "picture": "https://i.ibb.co/0K9W6h/i-Health-Blood-Pressure-Monitor.png",   "category": "Health &amp; Fitness",   "description": "See distance, pace, split times, elevation climbed and review routes.",   "distributorLogo": "https://www.muzley.com/uploads/deviceslogos/brand_5715595959a606994359a29.png",   "price": 174,   "rating": 1,   "quantityStock": 29,   "id": 25,   "name": "iHealth Blood Pressure wrist Monitor",   "productCode": "CD88829464316",   "picture": "https://i.ibb.co/YVscr8H/i-Health-Blood-Pressure-wrist-Monitor.png",   "category": "Health &amp; Fitness",   "description": "Track your workouts, monitor your performance</pre>



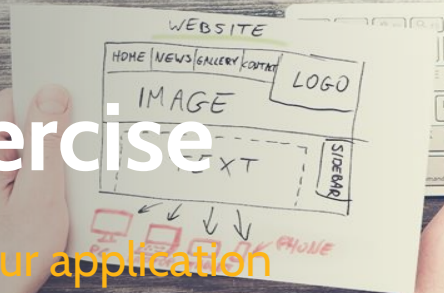
To test the "GetProduct" REST API copy and paste the "Request URL" in the browser and check if it returns list of products

## Jump Start Training



# Bonus exercise

Ideas to improve your application



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!

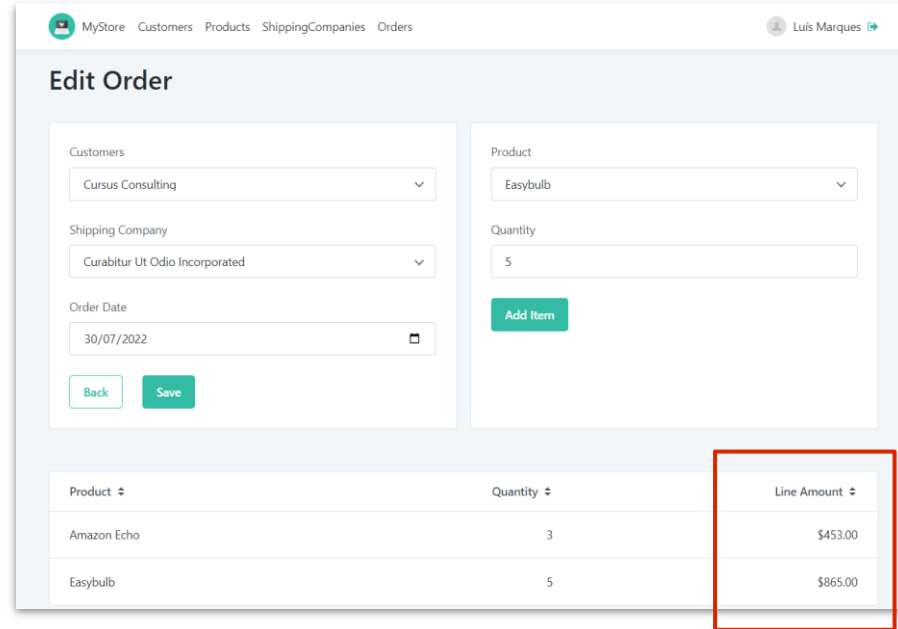


Bonus exercises are optional!

## 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

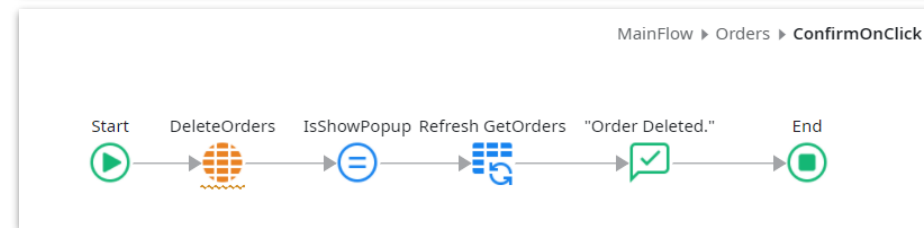
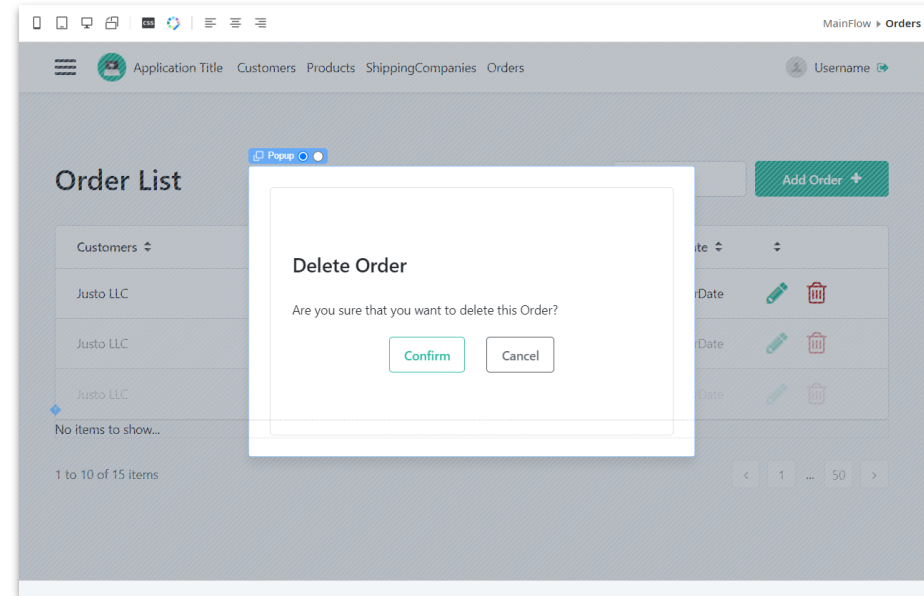
- In the “Homepage” screen open up the “GetProducts” Aggregate
- Switch back to the “Homepage” UI, and click on the “PriceRangeSliderInterval”.
  - Change the minimum value to 50;
  - Change the maximum value to 300.
- Now, for both the local variables **MinPriceSelected** and **MaxPriceSelected**, click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

The screenshot displays the DevExpress IDE interface for configuring a widget. At the top, the 'GetProducts' aggregate is shown with a filter expression: `Product.Price > MinPriceSelected and Product.Price < MaxPriceSelected`. Below this, a 'PriceRangeSliderInterval' widget is visible, showing a range of 50 - 300. The 'Properties' panel for this widget is open, showing the 'MinValue' set to 50 and 'MaxValue' set to 300. A dropdown menu is open, showing a list of local variables, with 'MinPriceSelected' selected. The 'MinPriceSelected' local variable properties are also visible, showing its name, description, data type (Currency), and default value (50).

## 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.





**Congratulations!**

You have successfully built a web app



# Jumpstart training

This material is owned by OutSystems and may only be used in the ways described in this Copyright Notice:

You may take temporary copies necessary to read this document

You may print a single copy of this material for personal use

You must not change any of this material or remove any part of any copyright notice

You must not distribute this material in any shape or form