

# Appointment Detail

## Table of Contents

<b>Scenario.....</b>	<b>2</b>
<b>How-To.....</b>	<b>3</b>
Setting Up the Appointment_Detail Screen	3
Fetching the Data	6
Fetching an Appointment: Create the Aggregate	7
Fetching an Appointment: Adding the Data Sources	8
Fetching an Appointment: Filtering the Data	12
Fetching Patient's Image	14
Fetching the Patient Onboarding Info	17
Building the UI	19
Screen Title	20
Setting Up the UI for the Patient Information	22
Patient Image	23
Patient Entity Information	26
User Entity Information	31
Patient Onboarding Info	34
Navigate Back to the Appointments List	37
Testing and Results	40
<b>Wrapping up.....</b>	<b>42</b>
References	42

## Scenario

At this point of the application, the doctor experience only has a list of its appointments, where the doctors can see them in a tabular layout, and then choose an appointment to start or review it.


Appointments						
Pick Appointment	Status ▾	Patient ▾	Date ▾	Start Time ▾	Clinic ▾	Specialty ▾
<a href="#">Start Appointment</a>	Submitted	Patricia Wesley	16 Jul 2022	12:30	Main Medical Center	Dental
<a href="#">Start Appointment</a>	Submitted	Ann Marie Pitt	21 Jul 2022	11:00	Main Medical Center	Dental
<a href="#">Start Appointment</a>	Submitted	Krissa Tesena	30 Jul 2022	11:30	Main Medical Center	Dental

1 to 3 of 3 items

However, these Buttons still lead to an empty Screen. In this tutorial, you will start implementing the UI for this Screen. The Screen should have a Card with:

- Patient Image
- Patient's full Name and Birth Date
- Patient's Mobile Phone and Email (from the User Entity)
- Patient's Social Security Number and Insurance Number (from the PatientOnboarding Entity).

The Screen should look like the following screenshot:

Tesena, Krissa Appointment			
	Name	Birthday	
	Krissa Tesena	1995-06-04	
	Mobile Phone	Email	
	987456	kris@tesena.com	
Social Security Number		Insurance Member	
987456		9874565	

# How-To

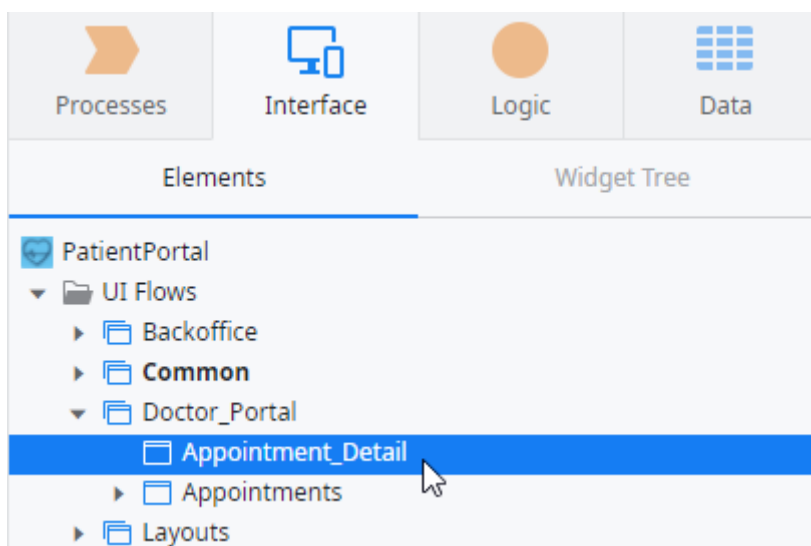
Now that you know the scenario, let's build those features in the Patient Portal app by following this how-to guide!

## Setting Up the Appointment\_Detail Screen

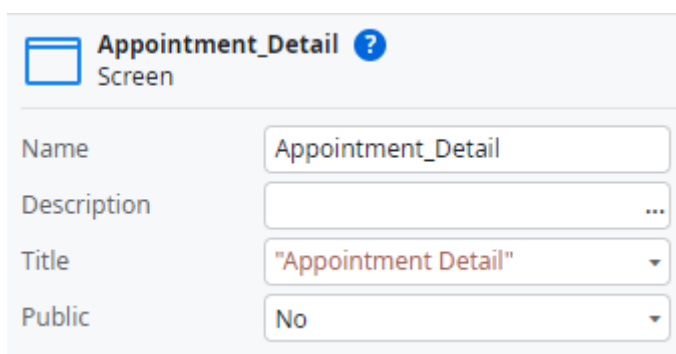
The Appointment\_Detail Screen was created in the previous tutorial, when you defined the On Click behavior for the Start / Review Appointment Button.

Let's go back to that Screen and set up the Title, Roles and Input Parameter.

- 1) Open the **Appointment\_Detail** Screen in the Doctor\_Portal flow.

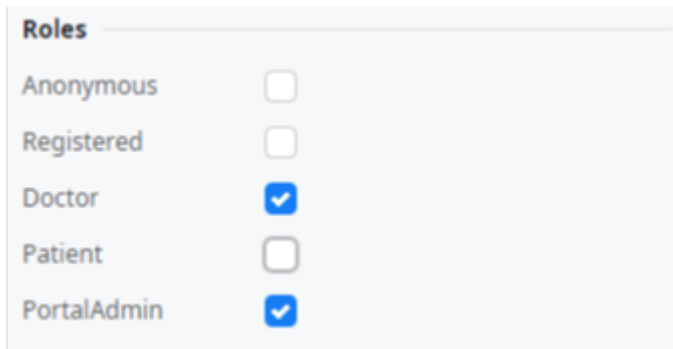


- 2) Set the **Title** to "Appointment Detail" in the Screen's properties.



Only a user with the Doctor or Portal Admin roles can access this Screen. So we need to set this up in the Screen properties as well.

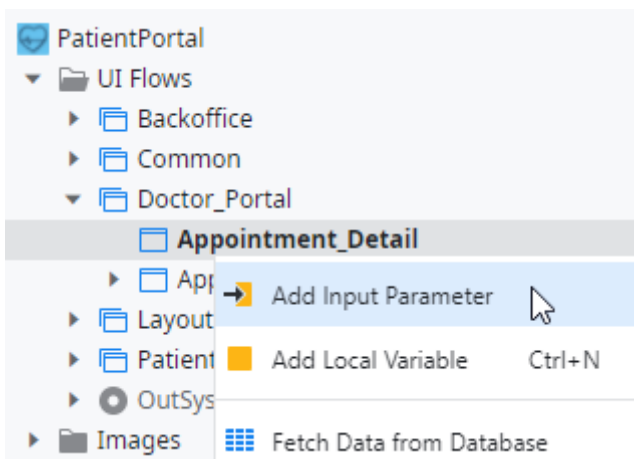
- 3) This Screen should only be accessible by a **Doctor** or **PortalAdmin**. So, unselect the Registered and Patient Roles in the roles section of the properties.



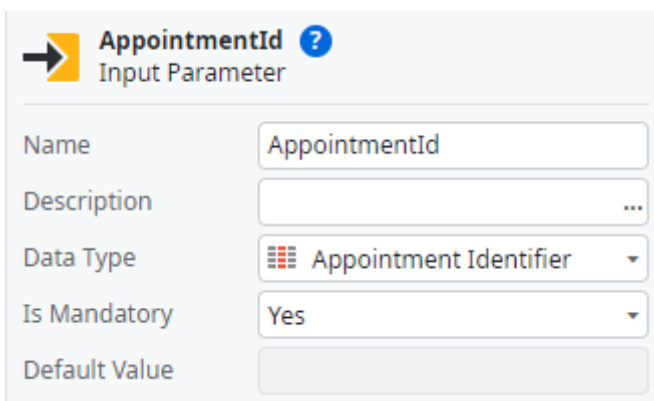
Roles	
Anonymous	<input type="checkbox"/>
Registered	<input type="checkbox"/>
Doctor	<input checked="" type="checkbox"/>
Patient	<input type="checkbox"/>
PortalAdmin	<input checked="" type="checkbox"/>

The Appointment\_Detail Screen will show the details of one specific Appointment, not all of them. So you will need to define an Input Parameter on the Screen, that will have the identifier of the appointment being displayed.

- 4) Right-click on the Screen and select **Add Input Parameter**.



- 5) Set its **Name** to *AppointmentId*.



AppointmentId ? Input Parameter	
Name	AppointmentId
Description	...
Data Type	Appointment Identifier
Is Mandatory	Yes
Default Value	

The **Data Type** should automatically change to Appointment Identifier. If that

does not happen, you can change the Data Type in the dropdown of the property.

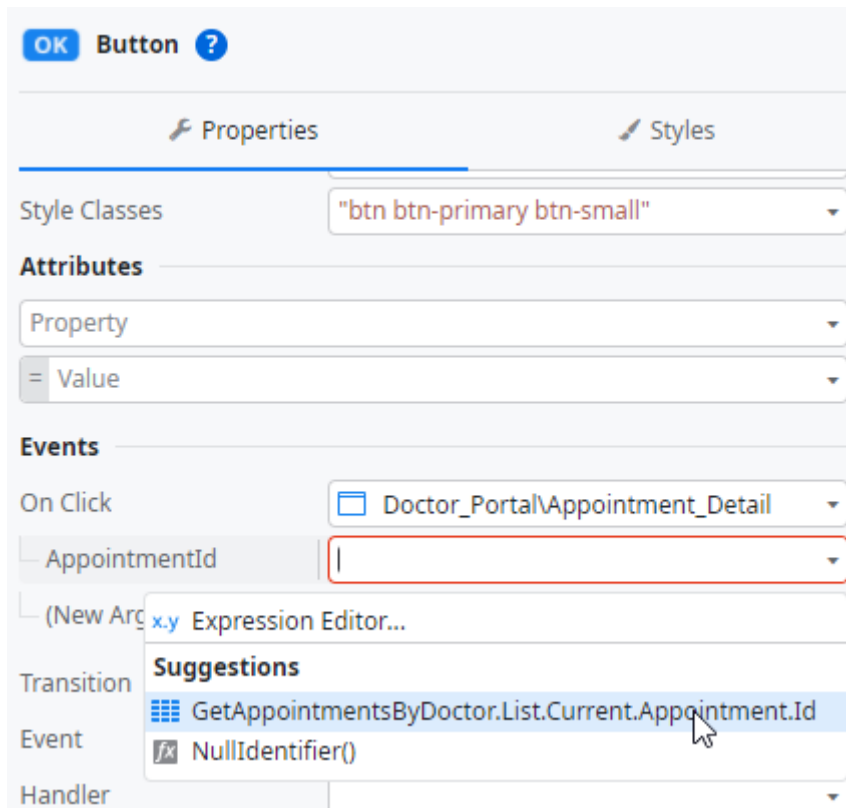
You now have an error on your app. That's because the button to start / review an appointment has this Appointment\_Detail Screen as destination. Since now you have an input parameter on the new Screen, you need to go back to the Appointments Screen and set a value for the input parameter.

- 6) Double-click on the error to go back to the Appointments Screen and to the properties of the Button.

- 7) Double-click on the error to go back to the Appointments Screen and to the properties of the Button.

✖ 1 Error, 2 Warnings		Debugger
✖ Required Property Value	A valid expression must be set for parameter 'AppointmentId'.	
⚠ Unused Element	Input Parameter 'AppointmentId' is never used in Screen 'Appointr	

- 8) Set the **AppointmentId** to  
`GetAppointmentsByDoctor.List.Current.Appointment.Id`



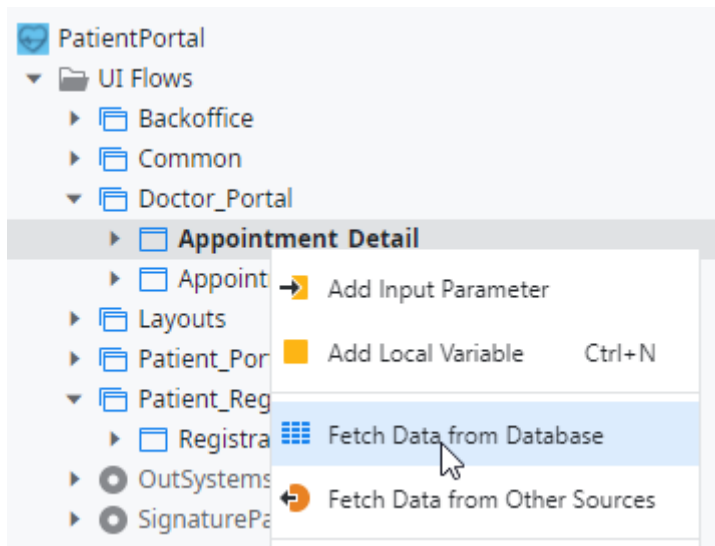
## Fetching the Data

In this section, you will fetch the data required to display all the info regarding an appointment. First, you will fetch the appointment, based on its identifier, as well as the information regarding its patient and doctor. Then, you will fetch the patient's image, so the doctor can easily identify their patient. Finally, you will fetch the patient's onboarding info.

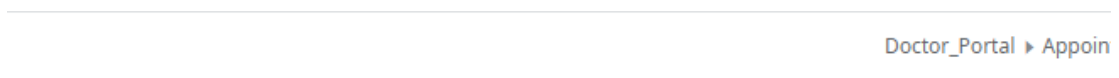
## Fetching an Appointment: Create the Aggregate

Now that we have the Id of the Appointment, let's create an Aggregate to fetch the respective appointment.

- 1) Right-click on the Screen and select **Fetch Data from Database**.



- 2) An empty Aggregate will appear. Click in the preview area to select a data source.



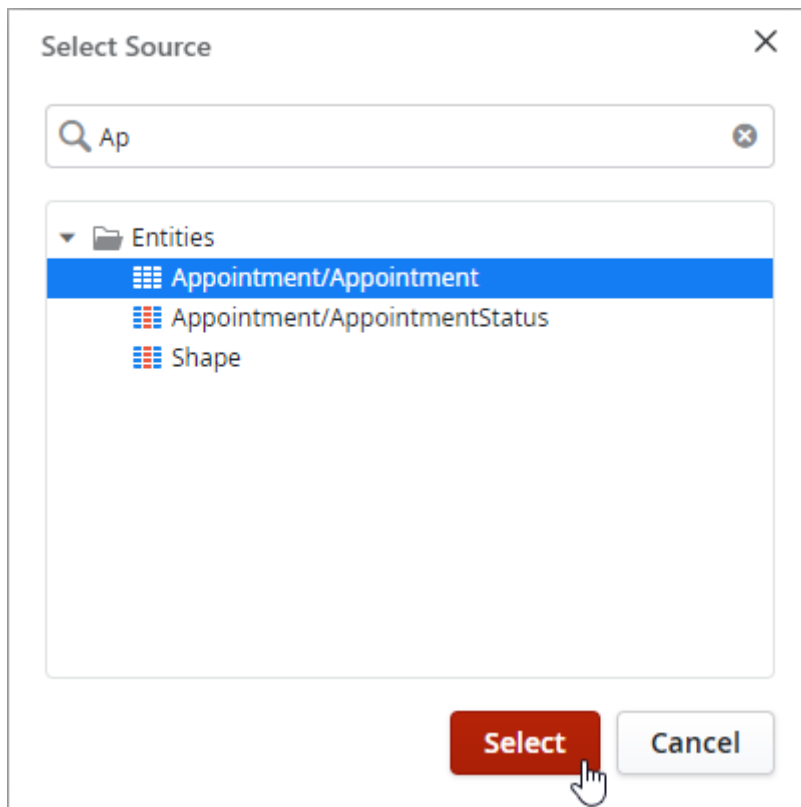
Click to add a  
database entity

or



Drag a database  
entity from the right

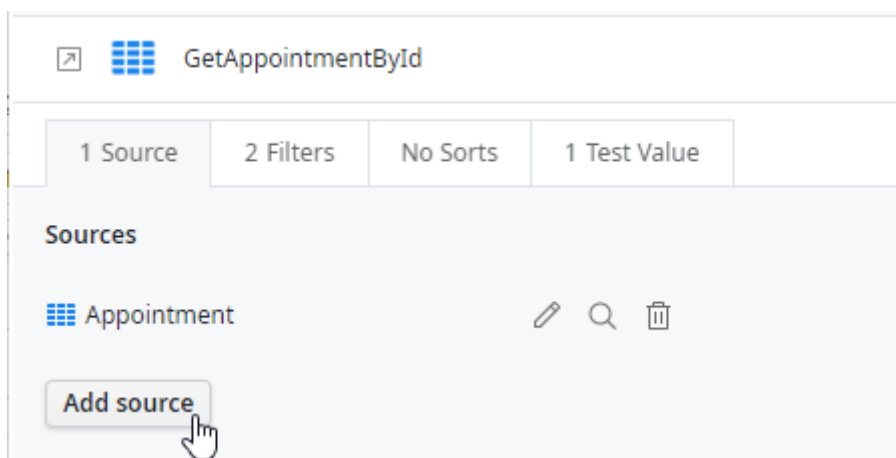
- 3) Select the **Appointment** Entity in the dialog, then click on **Select**.



## Fetching an Appointment: Adding the Data Sources

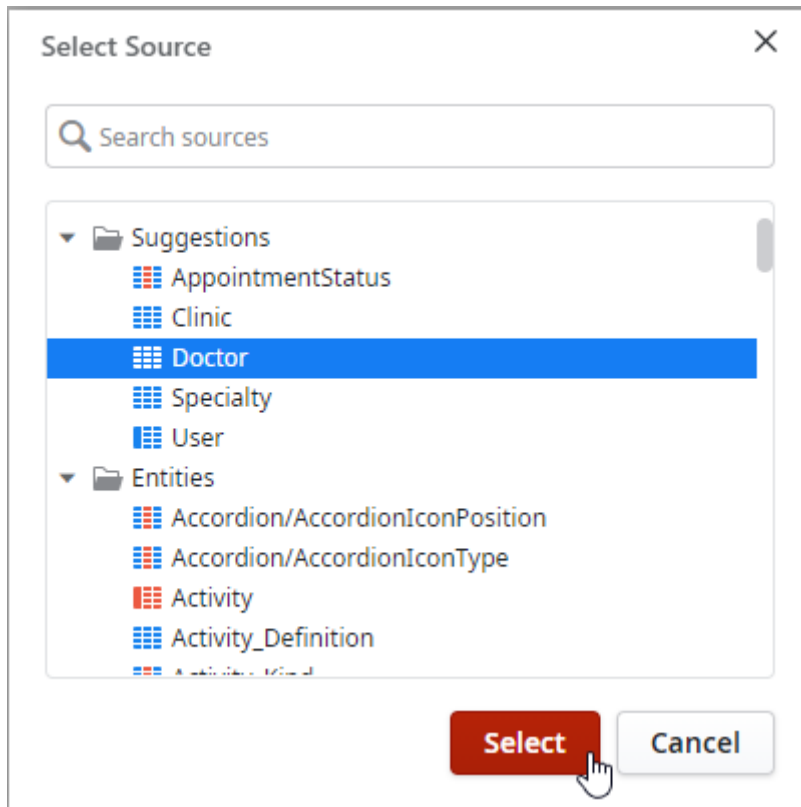
Since you need information stored in different Entities, this Aggregate must fetch data from more than one data source. You can quickly do that by adding more sources to the Aggregate. For this Screen you need information from the Doctor, Patient and User.

- 1) Click on **Add source**.





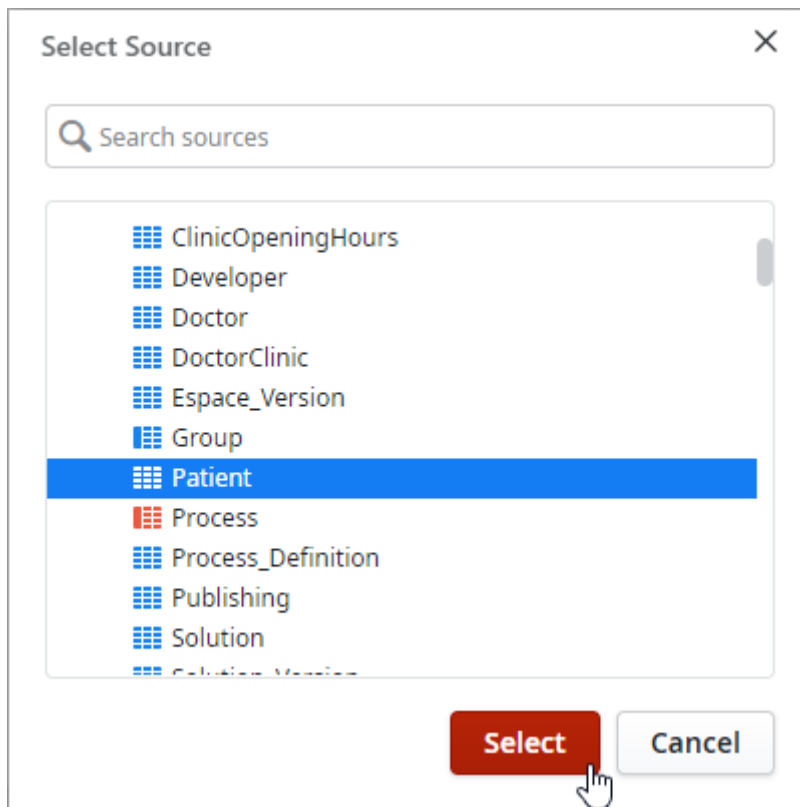
- 2) Select the **Doctor** Entity in the dialog, then click on **Select**.



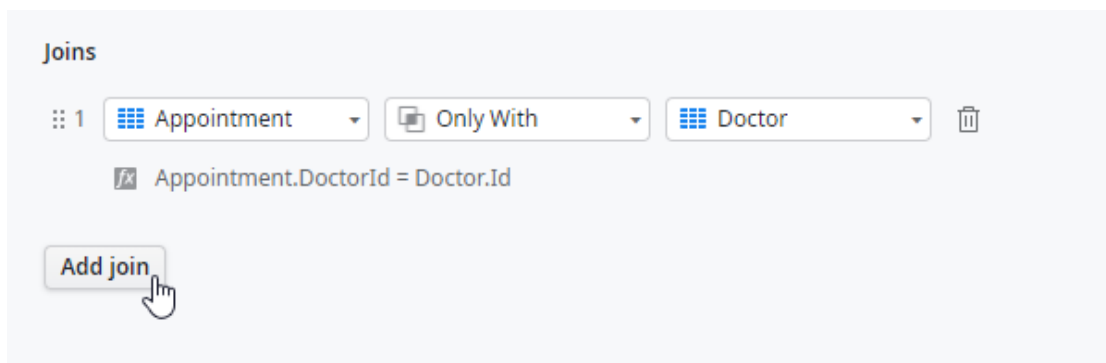
A join will automatically be created with the condition `Appointment.DoctorId = Doctor.Id`

We also need to fetch the Patient's information. So we'll add two more sources, Patient and User.

- 3) Add a new source and select the **Patient** Entity.



- 4) Click on **Add join**.



- 5) Select the **Appointment** Entity in the first dropdown and the **Patient** Entity in the third one. Keep the Join as Only with.

Joins

1 Appointment Only With Doctor Appointment.DoctorId = Doctor.Id

2 Appointment Only With Select Source

Condition must be set.

Add join

Appointment

Doctor

Patient

- 6) Click on the error in the Join condition and add the following condition:

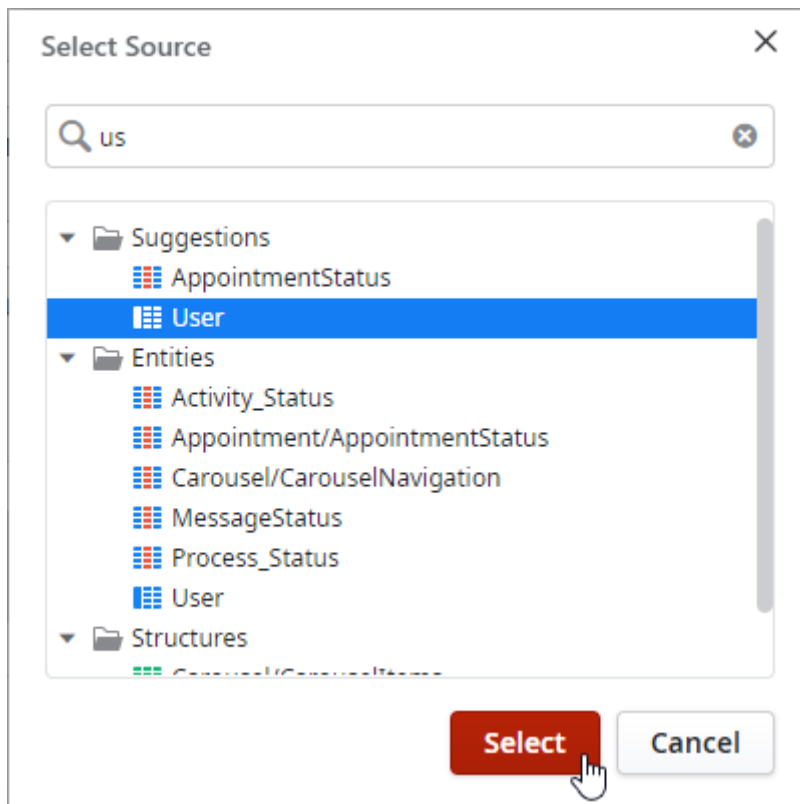
`Appointment.UserId = Patient.UserId`

Appointment.UserId = Patient.UserId Condition

Appointment.UserId = Patient.UserId

✓ The expression is ok (Type: Boolean)

- 7) Click on **Add source** again and select the **User** Entity.

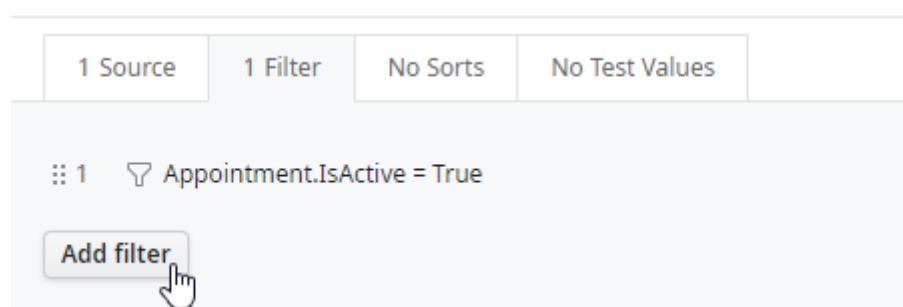


Now the join was created automatically for you!

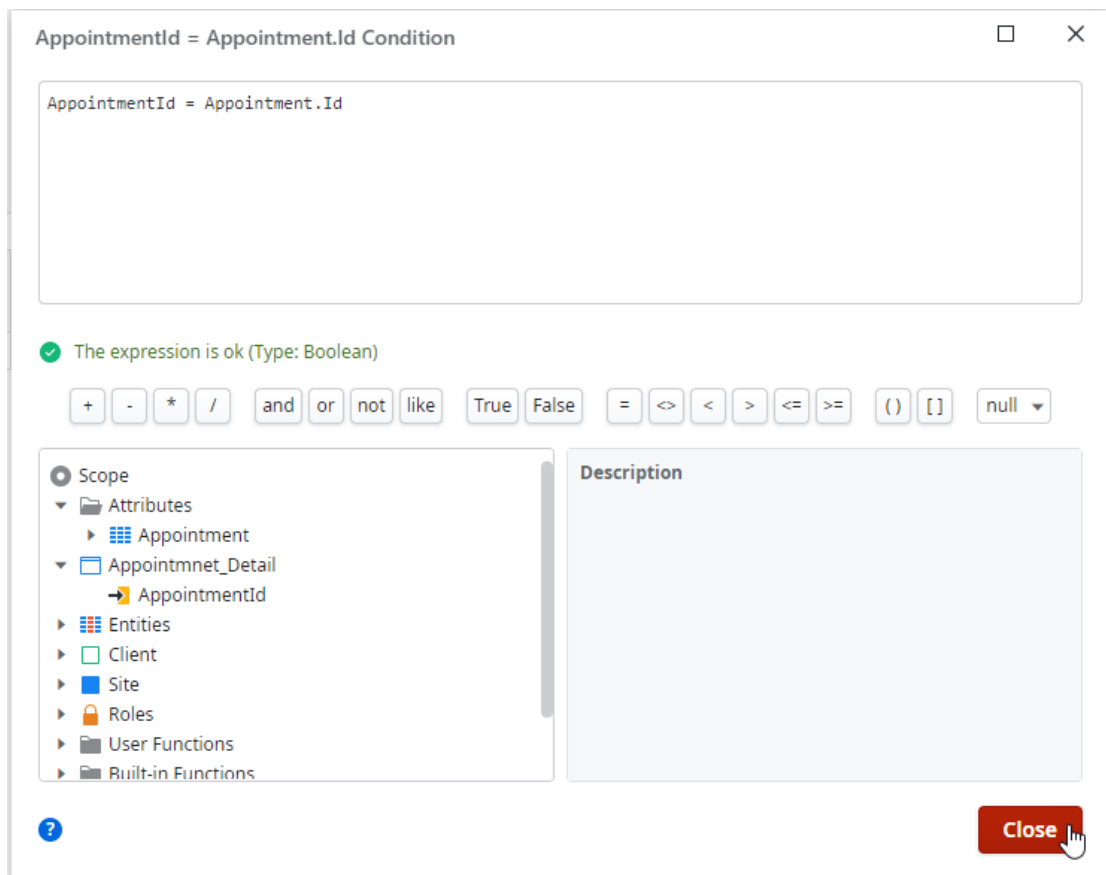
## Fetching an Appointment: Filtering the Data

Now that you have the sources, you have to filter the data to just return one appointment: the one that matches the AppointmentId Input Parameter.

- 1) Select the **Filter** tab and click on the button **Addfilter**.

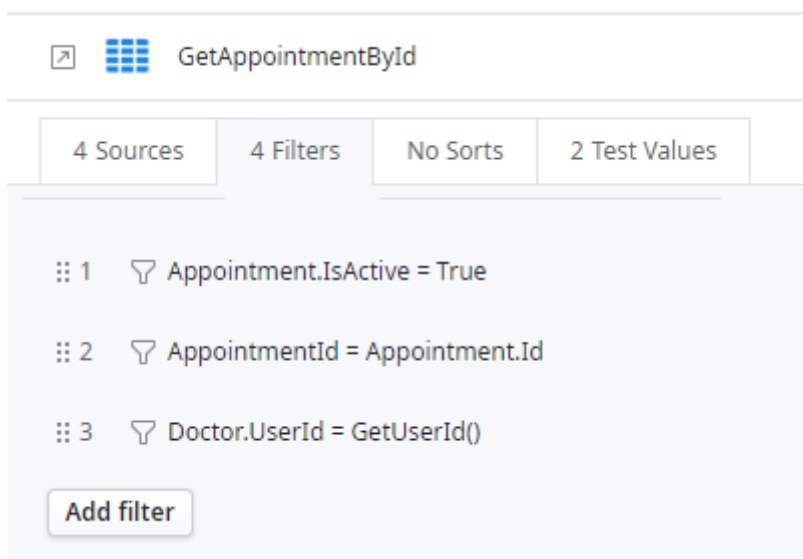


- 2) A dialog will appear so you can add the filter condition. Add the condition below: `AppointmentId = Appointment.Id`



This means that only the appointment with the Appointment Id from the Input Parameter will be fetched.

- 3) Add another filter with the following condition: `Doctor.UserId = GetUserId()`



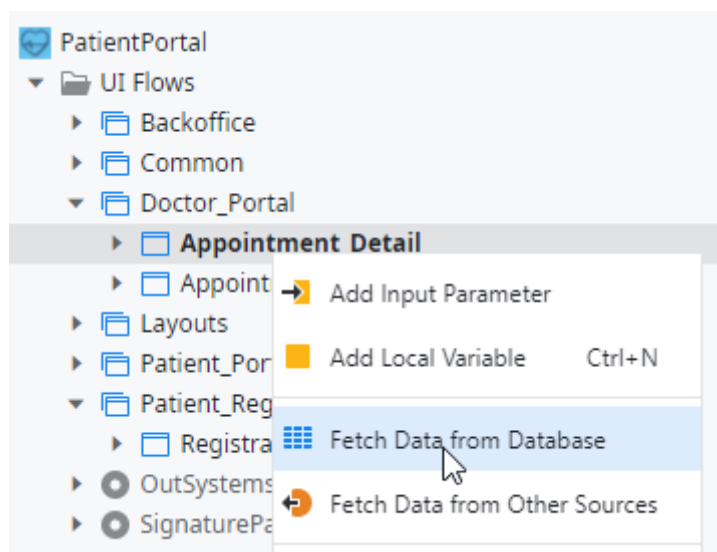
This will filter the Aggregate by the doctor that is currently logged in. The *GetUserId()* function exists by default in OutSystems and automatically returns the identifier of the User currently logged in.

Now it's time to fetch the data related with the patient's image.

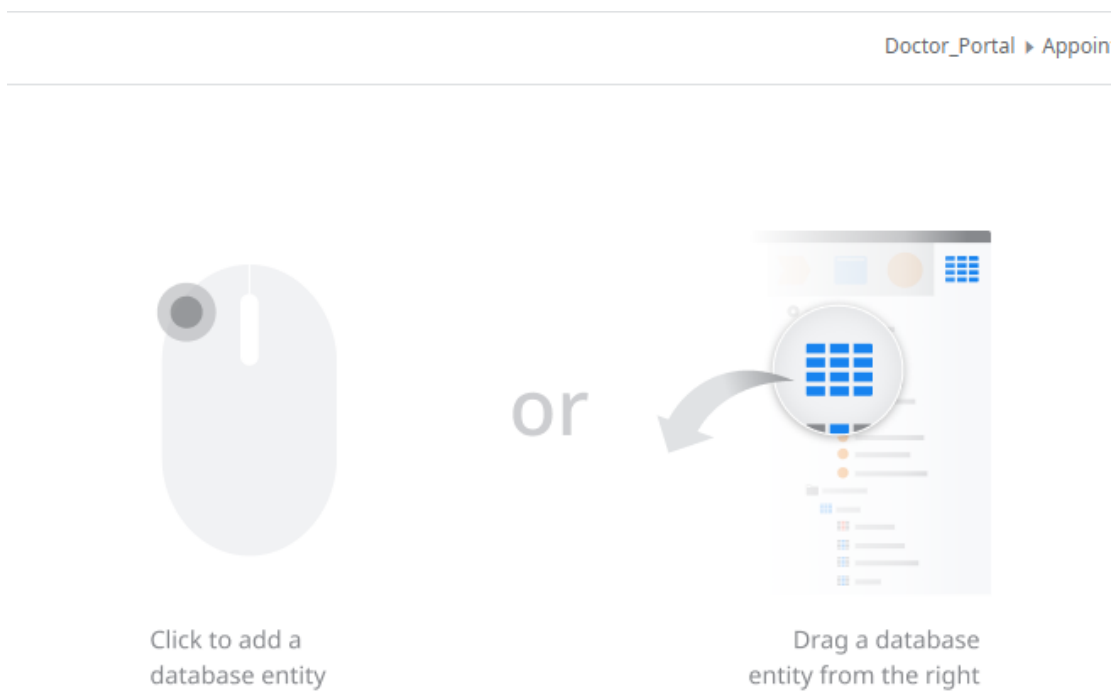
## Fetching Patient's Image

Now, you will create a second Aggregate to fetch the image of the patient that is associated with this appointment.

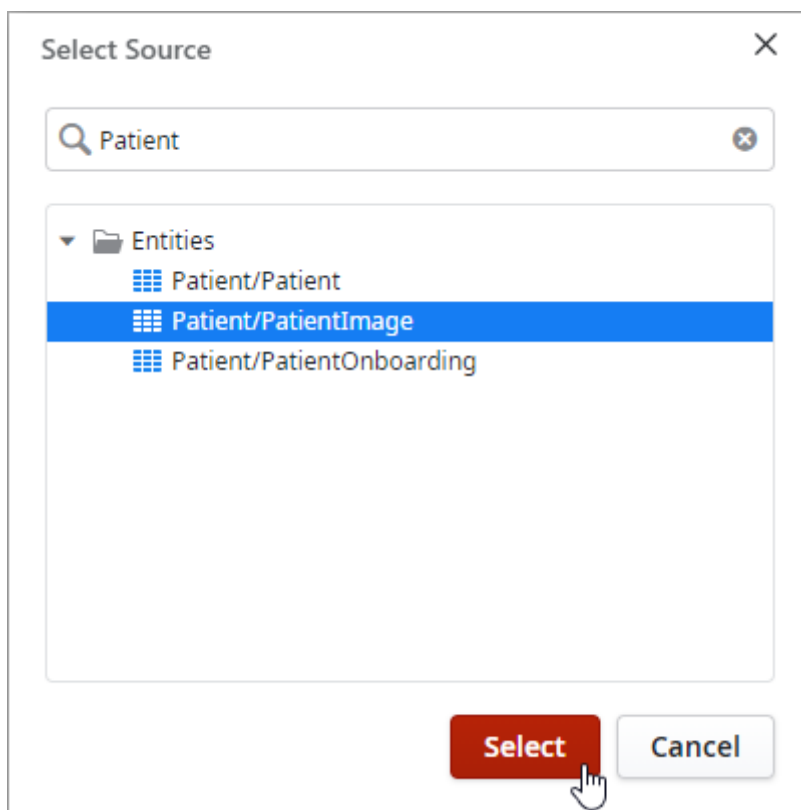
- 1) Right-click on the **Appointment\_Detail** Screen in the Interface tab, then select **Fetch Data from Database**



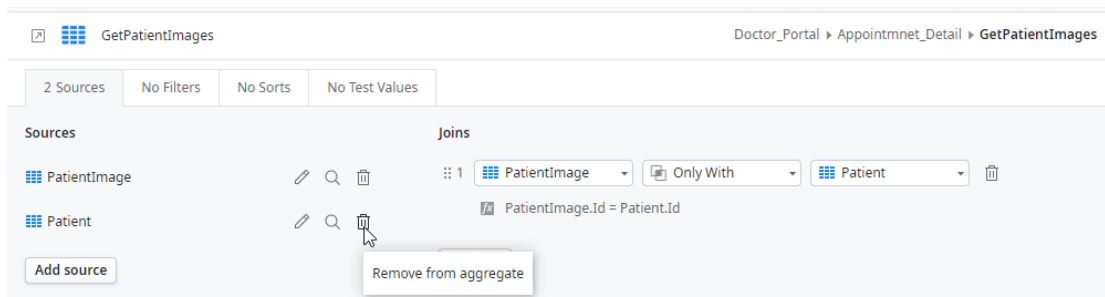
- 2) Click on the preview area of the Aggregate to add a new data source.



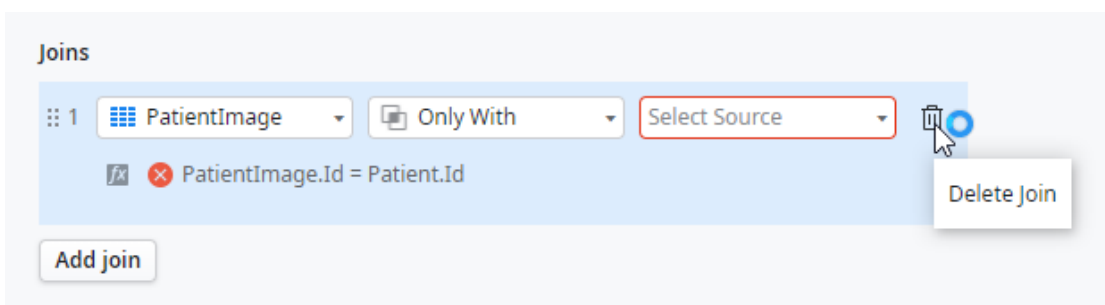
- 3) Select the **PatientImage** Entity in the dialog and click **Select**.



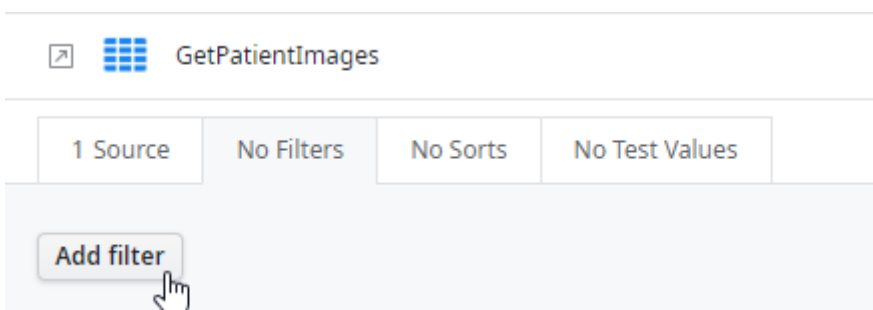
- 4) Remove the **Patient** Entity from the Sources, since you just want the image and nothing else.



- 5) Also, delete the Join since it will not be necessary.



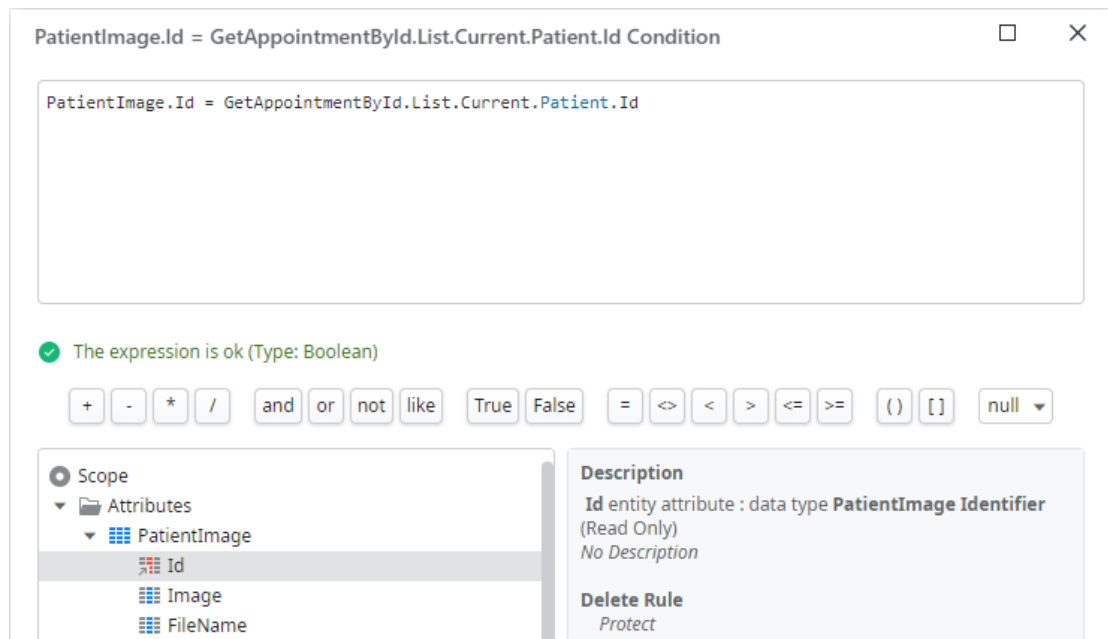
- 6) Click on the **Filter** tab and then on the **Add filter** button.



- 7) Add the condition below to the filter:



```
PatientImage.Id = GetAppointmentById.List.Current.Patient.Id
```



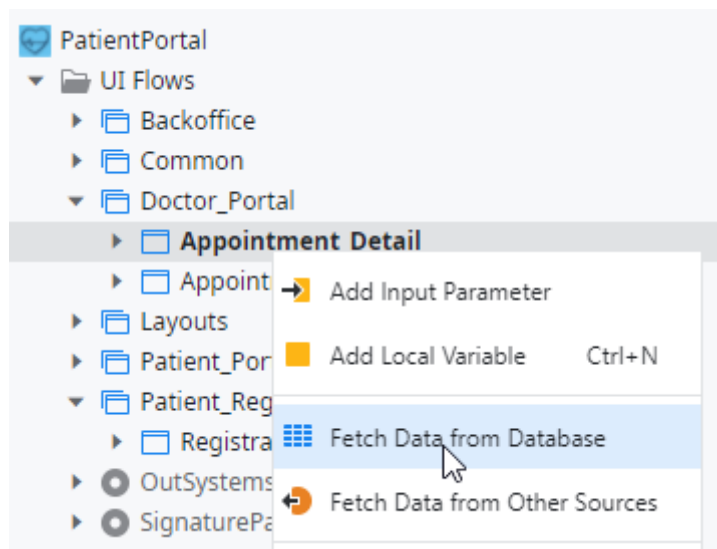
This Aggregate will only fetch the image that corresponds to the patient associated with the appointment displayed on the Screen. Since you already created an Aggregate that gives you that info (GetAppointmentsById), you can leverage its result on this filter.

## Fetching the Patient Onboarding Info

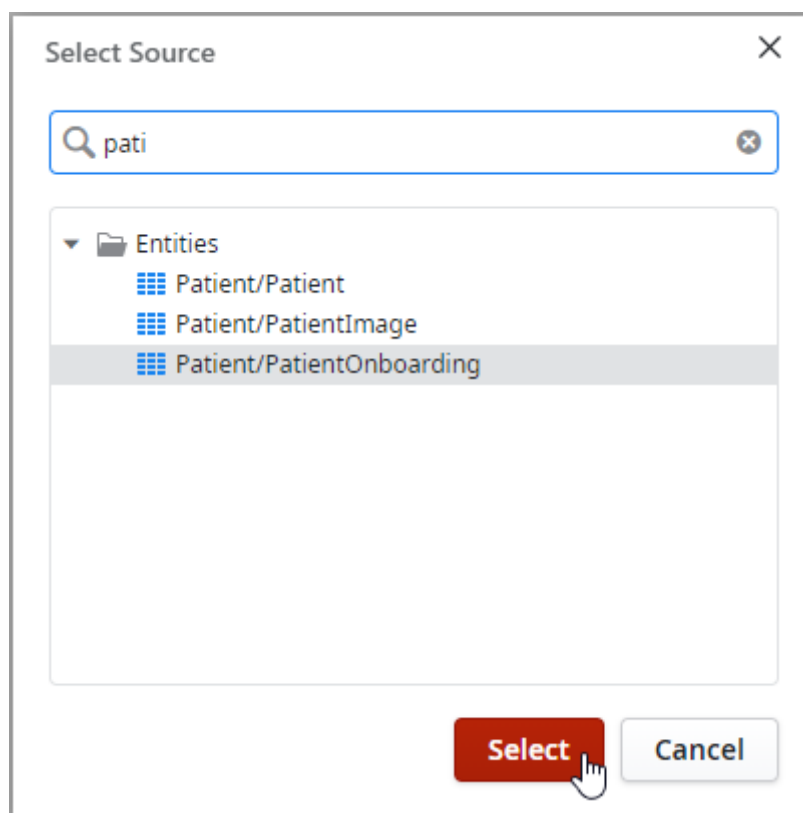
The Social Security Number and the Insurance Member Id are also pieces of information that may be relevant for the doctor. So, you will need a final Aggregate to

fetch data from the PatientOnboarding Entity, as well as set up the interface in the Screen.

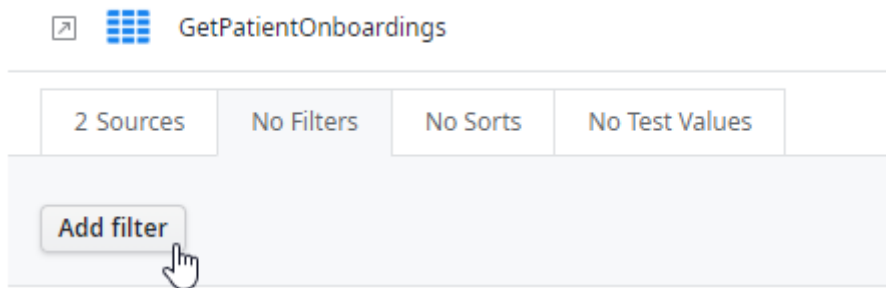
- 1) Right-click on the **Appointment\_Detail** Screen and select **Fetch Data from Database**.



- 2) Select the Entity **PatientOnboarding** as the source in the dialog.

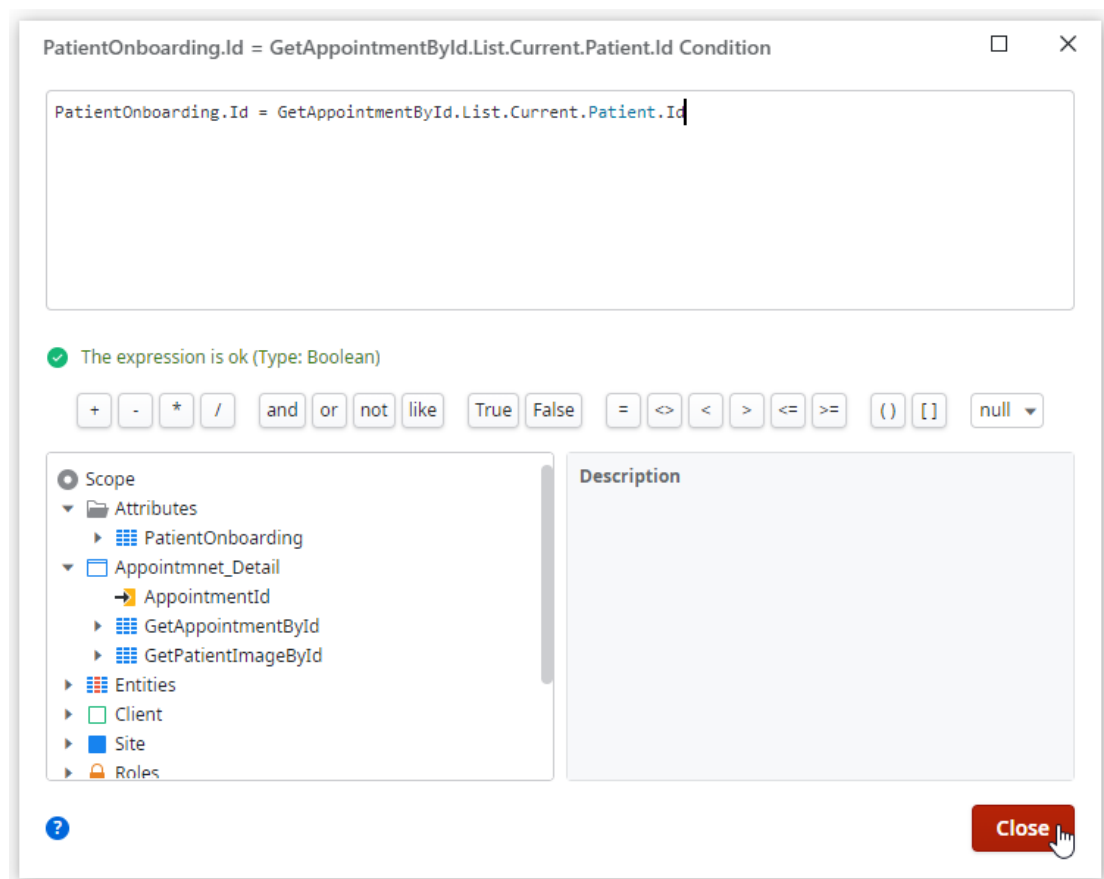


- 3) Click on the **Filter** tab, then click on the button **Add filter**.



- 4) Add the condition below:

`PatientOnboarding.Id = GetAppointmentById.List.Current.Patient.Id`



After you add the filter the Aggregate's name will change to *GetPatientOnboardingById*

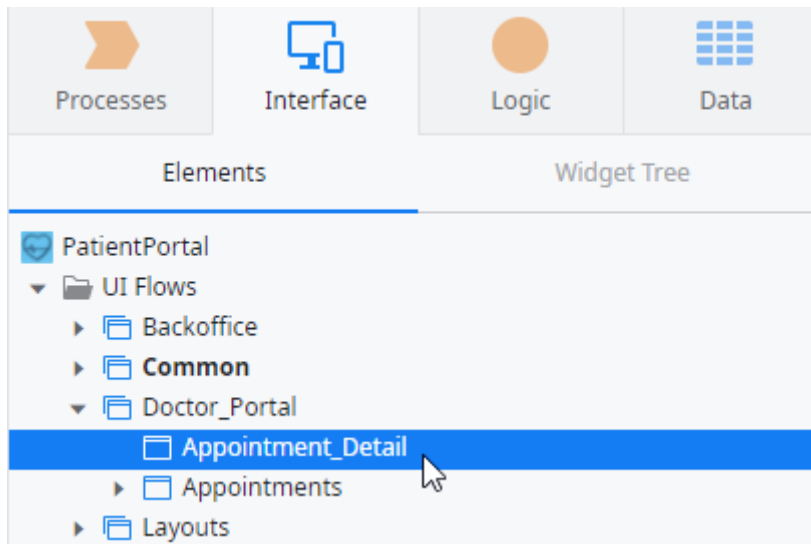
## Building the UI

Now that you have the Aggregates fetching the information from the database, it's time to define the UI. Let's start by the title of the Screen.

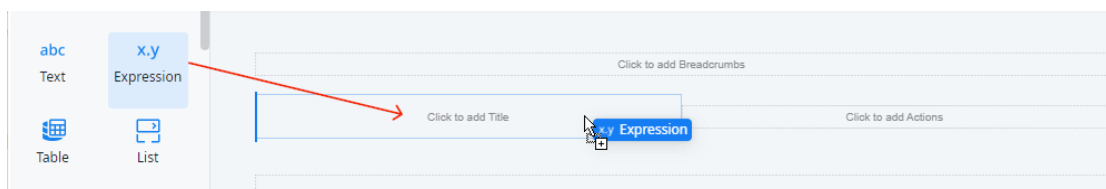
## Screen Title

The title of the Screen will have the name of the patient with the following format: Last Name, First Name.

- 1) Go back to the **Appointment\_Detail** Screen.



- 2) Drag an **Expression** from the left sidebar and drop it on the Screen's Title placeholder.



- 3) Set the Expression to  
`GetAppointmentById.List.Current.Patient.LastName+", " +`

`GetAppointmentById.List.Current.Patient.FirstName + " Appointment"`  
and click on the **Close** button.

Expression Value □ ×

```
GetAppointmentById.List.Current.Patient.LastName+", " + GetAppointmentById.List.Current.Patient.FirstName + " Appointment"
```

✓ The expression is ok (Type: Text)

+ - \* / and or not True False = <> < > <= >= ( ) [ ] null ▼

Scope

- Appointment\_Detail
  - AppointmentId
  - GetAppointmentById
- Entities
- Client
- User Functions
- Built-in Functions
- Resources

Description

?

Close

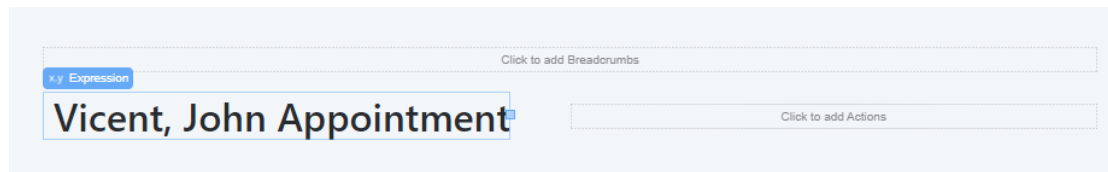
- 4) Click on the **Expression** to open its properties and set the **Example** property to *Vincent, John Appointment*.

x.y Expression ?

Properties Styles

Name	<input type="text"/>
Value	<code>GetAppointmentById.List.Current.Patient.L</code> ▼
Example	<code>Vincent, John Appointment</code>
Style Classes	▼

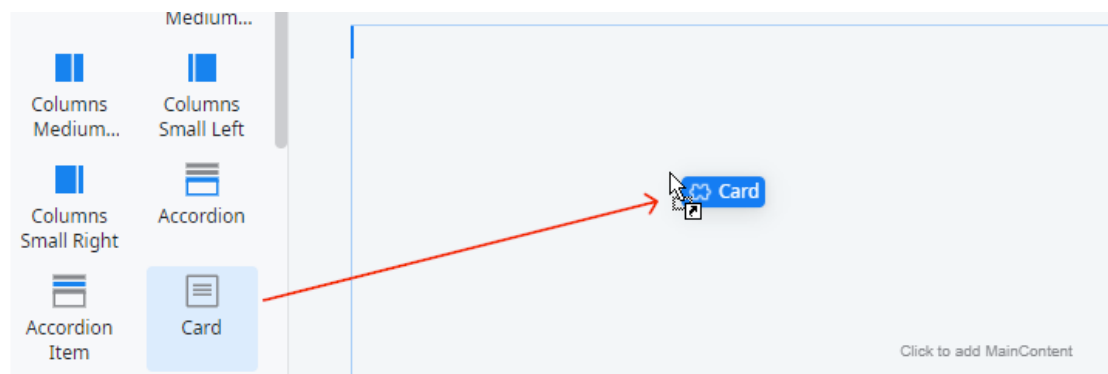
This step is not mandatory, but it helps developers to preview what the expression will look like before publishing it. So, this is a good practice.



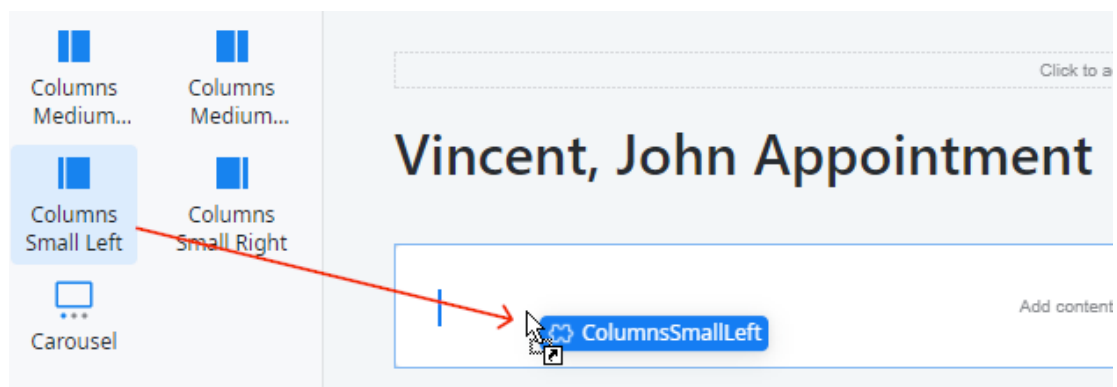
## Setting Up the UI for the Patient Information

Now, you will use an OutSystems UI component, called Card, to help you display the patient information. This is just a different option for the UI of your applications.

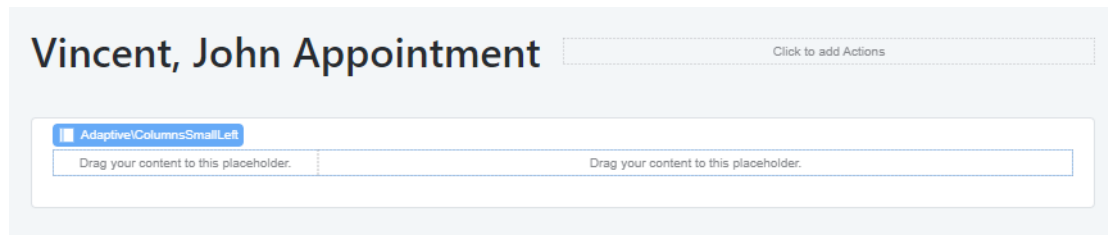
- 1) Drag a **Card** from the left sidebar and drag it to the main content area of the Screen.



- 2) Drag a **ColumnsSmallLeft** and drop it inside the Card.



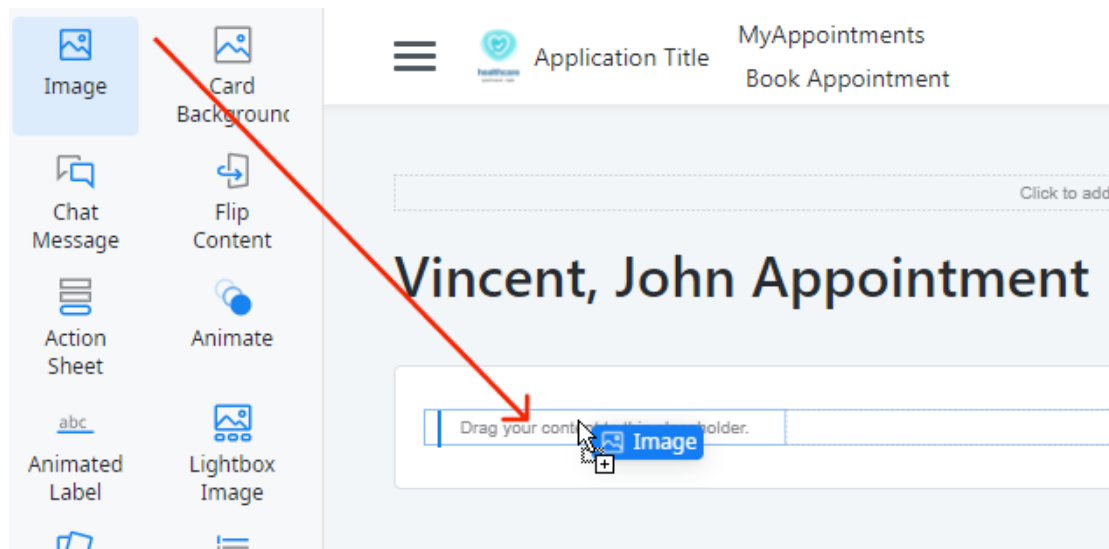
The smaller column on the left will display the Patient's picture (if they have one). But you haven't fetched that data yet! So you'll do that in the next section.



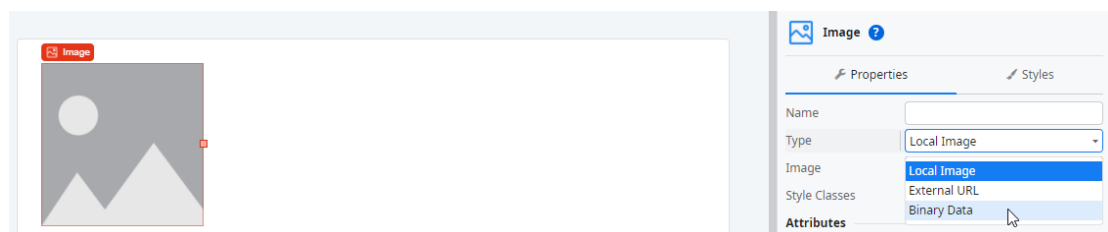
## Patient Image

The Aggregate to fetch the Image is done, so it's time to work on its UI.

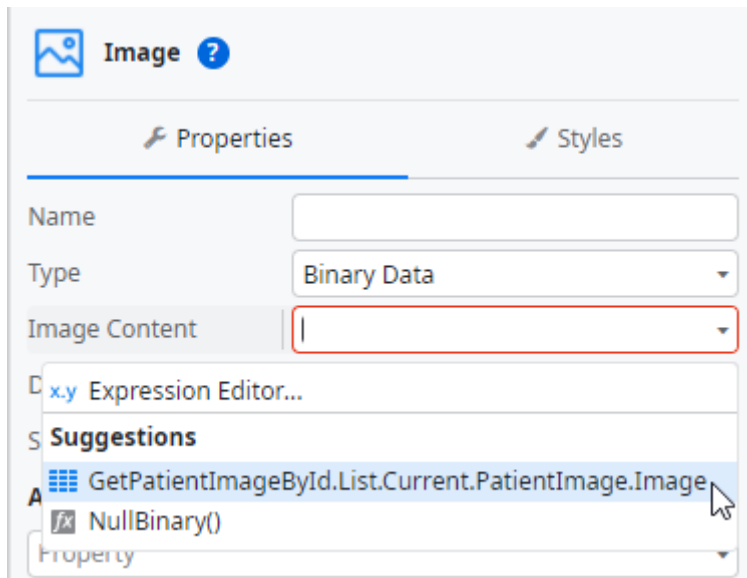
- 1) Drag an **Image** from the left sidebar and drop it inside the First column of the Card in the Screen preview.



- 2) In the properties of the Image, set the **Type** to **Binary Data**.

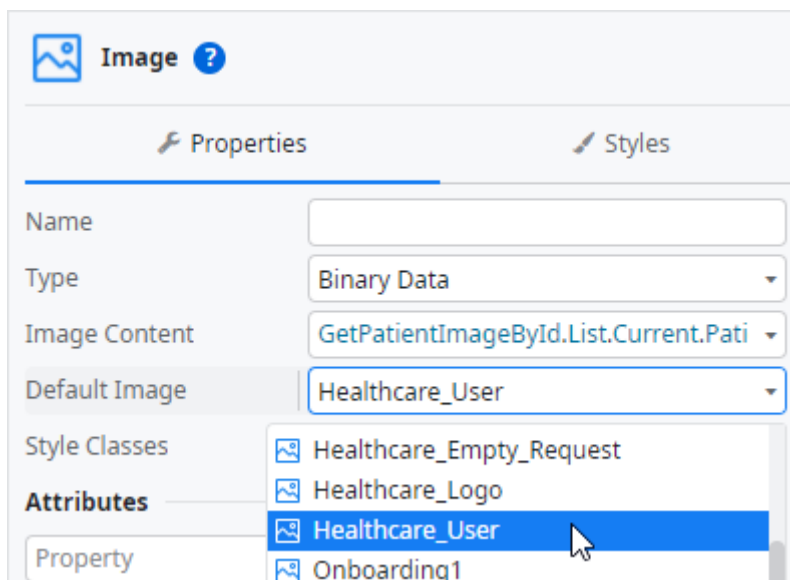


- 3) Still in the properties, set the **Image Content** property to the **GetPatientImageById.List.Current.PatientImage.Image** in the suggestions. This will tie the element on the Screen with the output of the Aggregate.



Let's make the image look nice and proportional. This is important because the Patient can upload images of different sizes and shapes or no image at all. So let's tweak the Style properties.

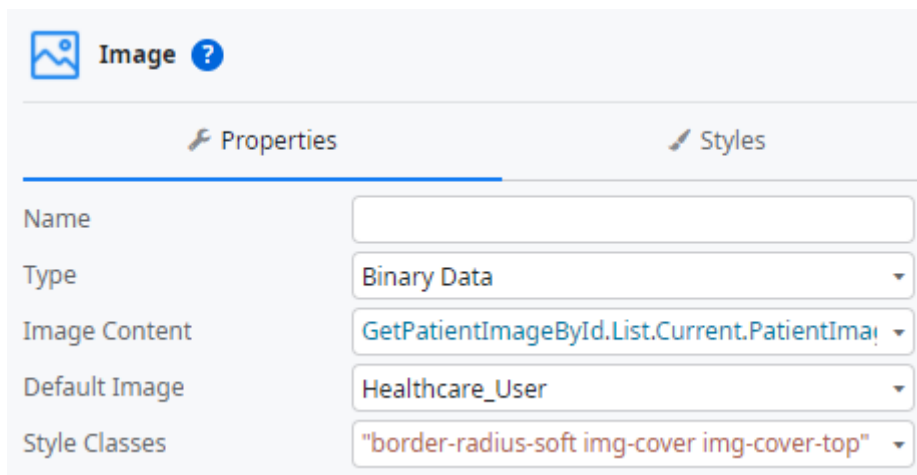
- 4) In the Image's properties, set the **Default Image** property to **Healthcare\_User**.



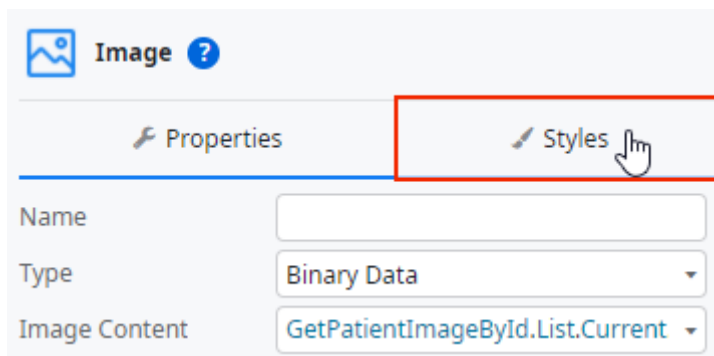
This is going to display a default image if the Patient does not upload a picture.



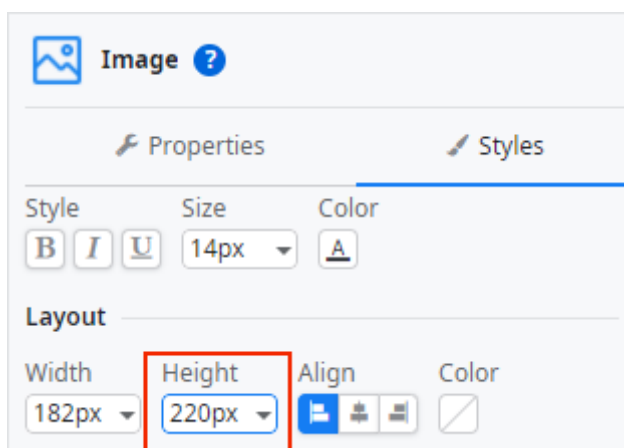
- 5) Set the **Style classes** property to "border-radius-soft img-cover img-cover-top" .



- 6) Click on the **Styles** tab to open the Styles editor.



- 7) Set the **Height** property to 220px.

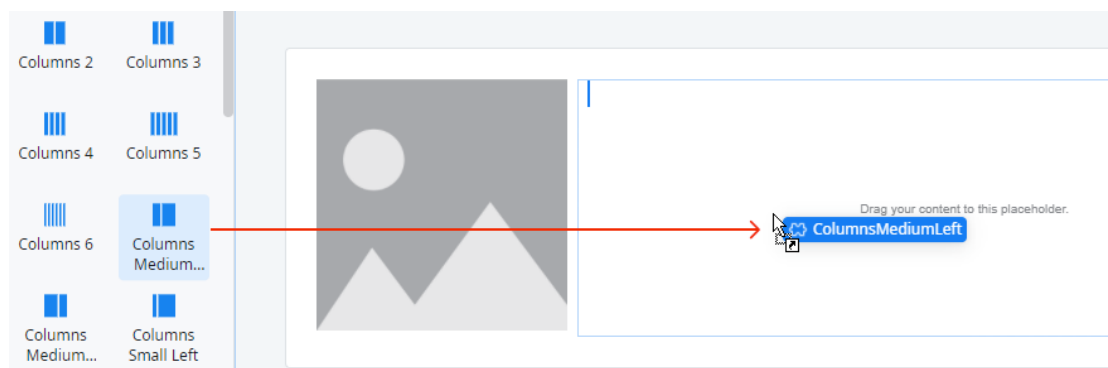


This will maintain the image's proportion.

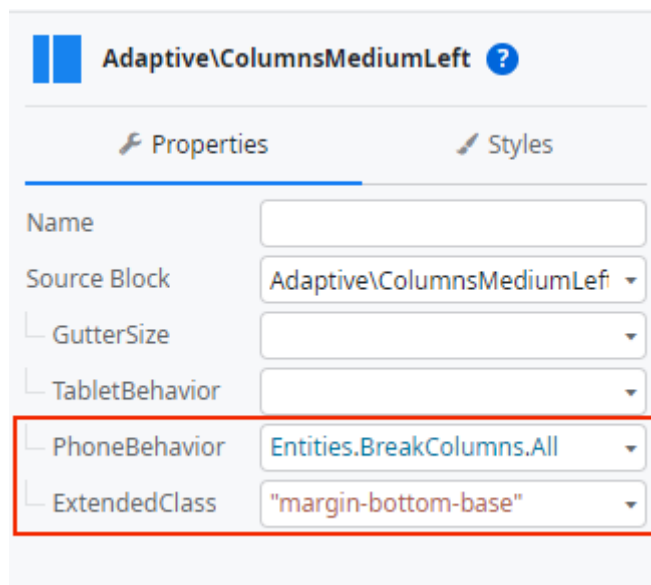
## Patient Entity Information

The right column, which is the bigger one, will contain important patient information, such as the name, birthday, mobile phone, etc.

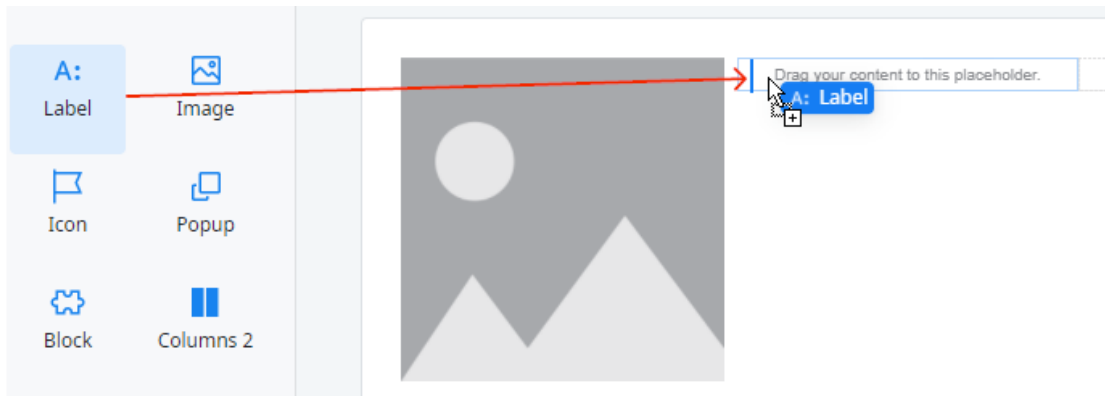
- 1) Back on the Screen, drag a **ColumnsMediumLeft** from the left sidebar and drop it inside the right column.



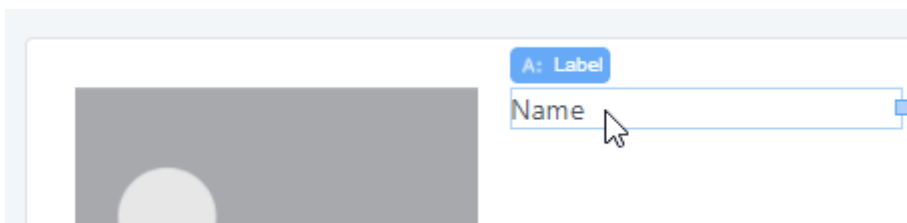
- 2) Click on the **PhoneBehavior** property and select **Entities.BreakColumns.All**. Also, set the **ExtendedClass** property to "margin-bottom-base".



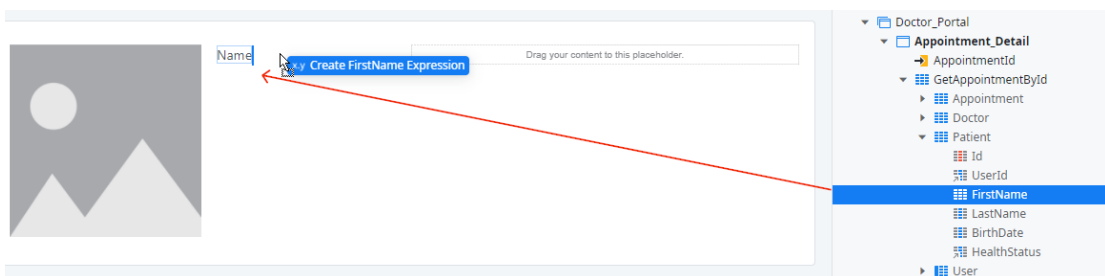
- 3) Drag and drop a **Label** inside the first column.



- 4) Click on the Label and type *Name*.

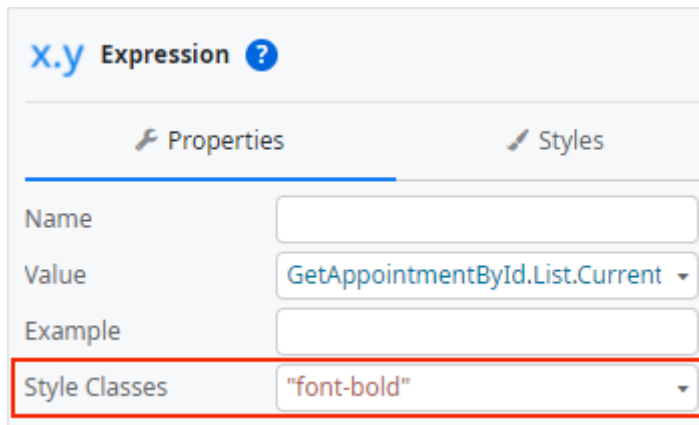


- 5) In the right sidebar, expand the Appointment\_Detail Screen, the GetAppointmentsById Aggregate and the Patient Entity inside it. Drag the **FirstName** attribute and drop it after the Label.



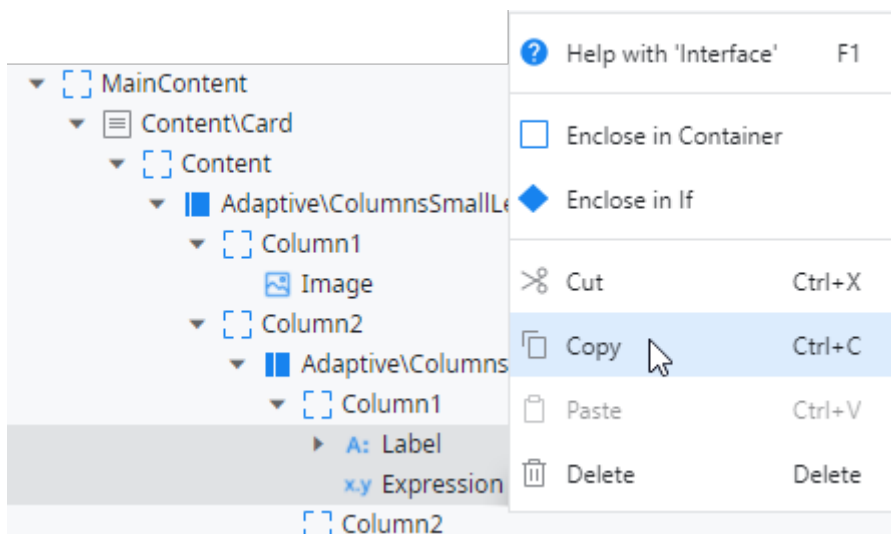
Use the Widget Tree to make sure the Expression is next to the label and **not** inside it.

- 6) In the Expression's properties, switch back to the Properties view (you may still be in the Styles tab), and set the **Style Class** to "**font-bold**" to highlight the content.



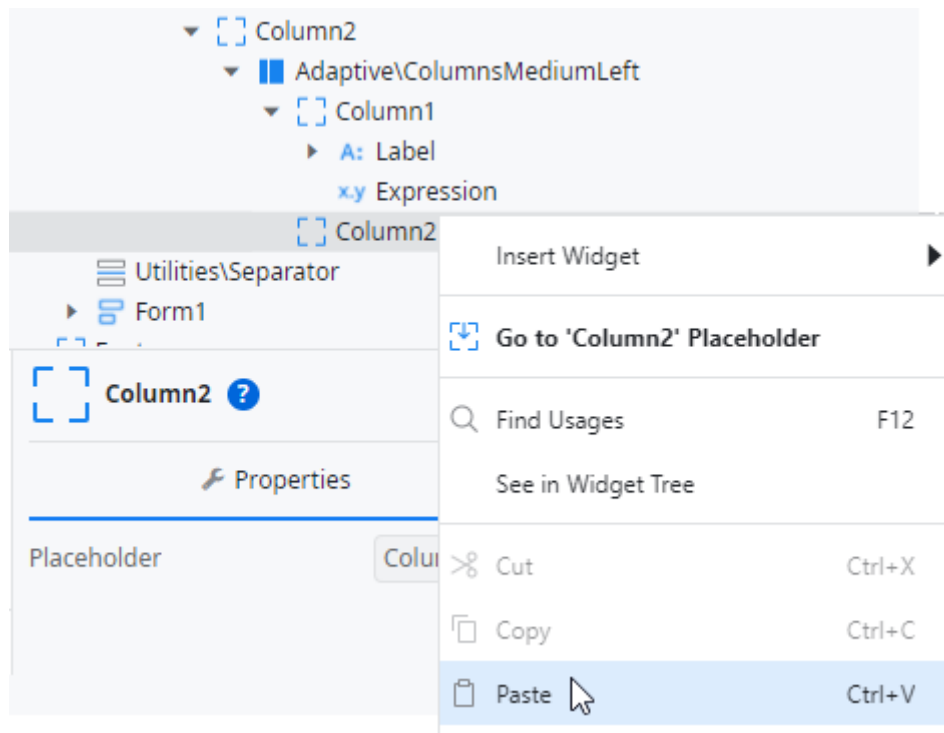
Now you need to do something similar for the Patient's Birthdate, so let's cut some corners.

- 7) Select the Label and the Expression for the name, then right-click on them and select **Copy**.



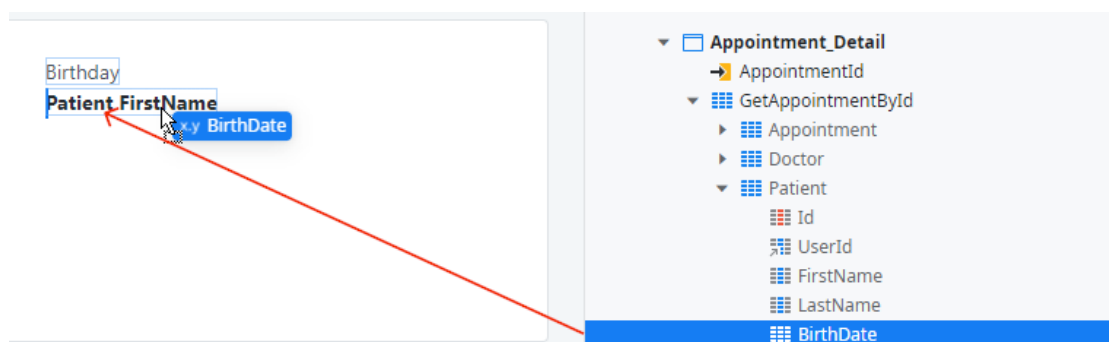
**Note:** the keyboard shortcut CTRL + C can also be used to copy elements.

- 8) Right-click on the Column2 and select **Paste** (or CTRL + V) .



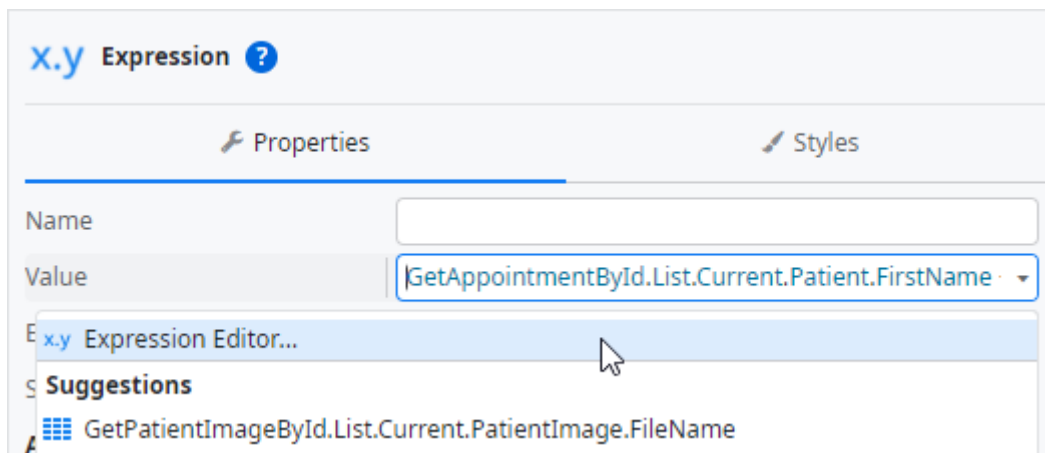
You will have two columns that look exactly the same. But now, you can simply replace the data.

- 9) Click on the **Label** on the Column2 and type *Birthday*.
- 10) In the right sidebar, go back to the Elements tab. You may still see the Patient Entity expanded, from when you dragged the FirstName attribute to the Screen. Now, drag the **BirthDate** attribute and drop it **on top of** the FirstName Expression in the Column2.



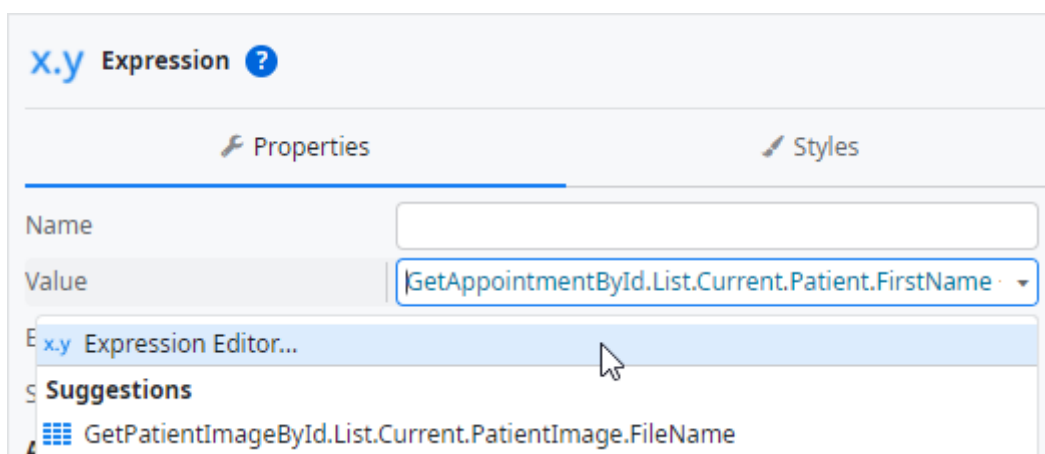
Last but not least, it's weird to just have the first name associated with the patient. Let's add the last name as well.

- 11) Click on the **FirstName** Expression to open its properties, then select **Expression Editor** in the **Value** property.

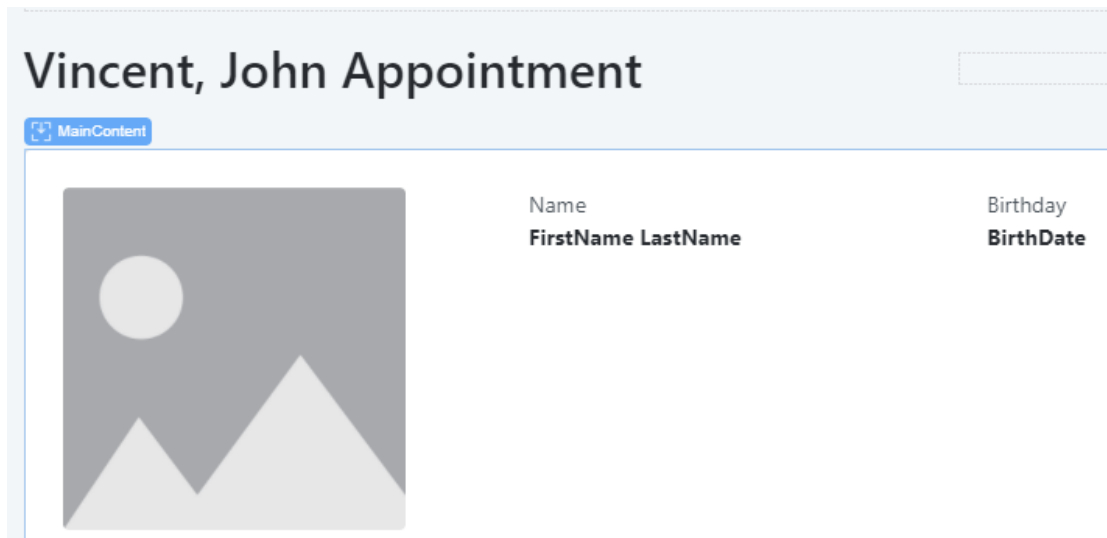


- 12) Change the Expression Value to:

`GetAppointmentById.List.Current.Patient.FirstName + " " +  
GetAppointmentById.List.Current.Patient.LastName`



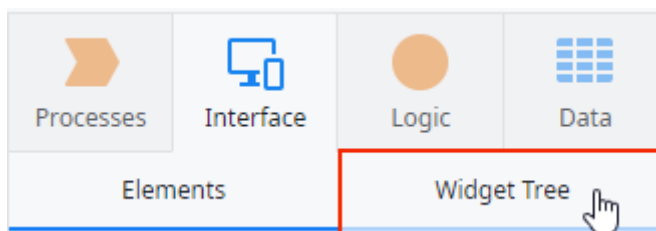
It's possible to concatenate values, characters and strings in these Expressions.



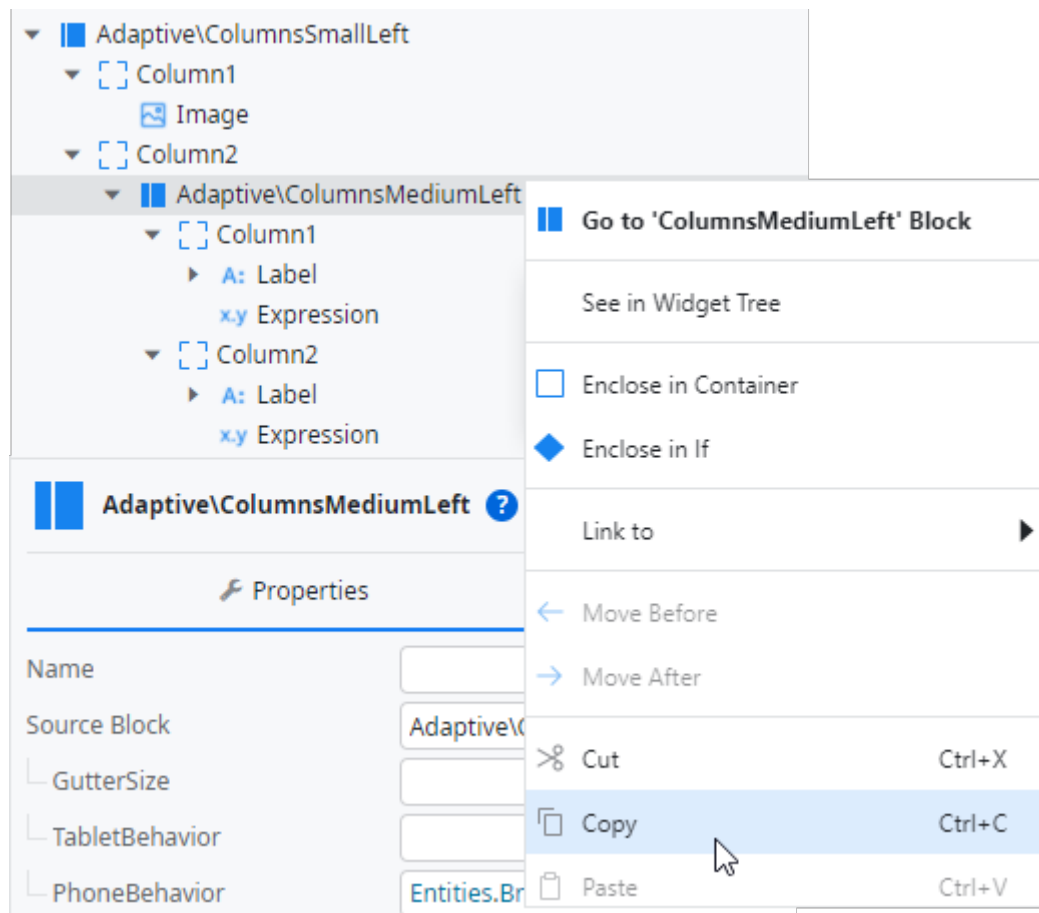
## User Entity Information

Let's continue filling in the Card, this time with the Phone Number and the Email. These attributes are part of the User Entity.

- 1) Open the Widget Tree.

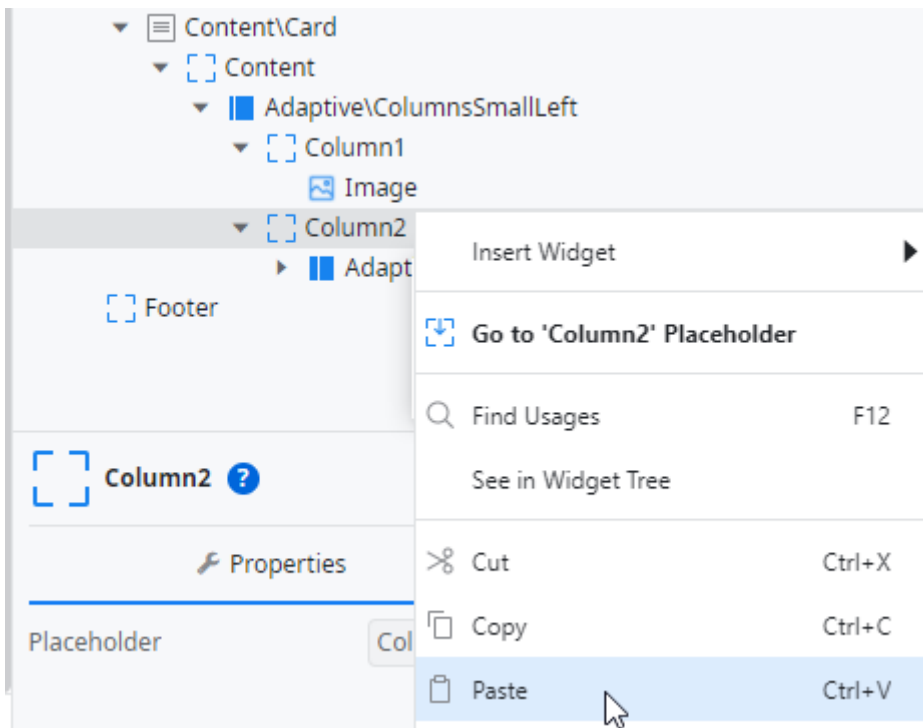


- 2) Right-click on the **ColumnsMediumLeft** element with the Name and Birthday, then select **Copy**.

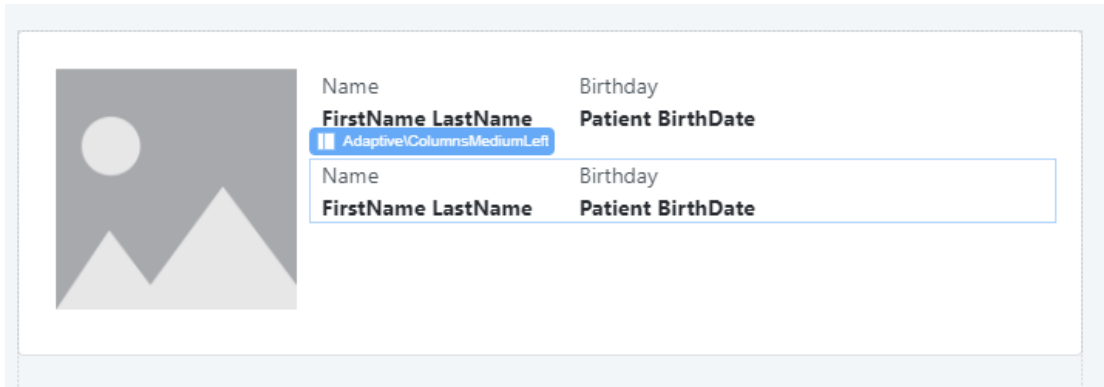




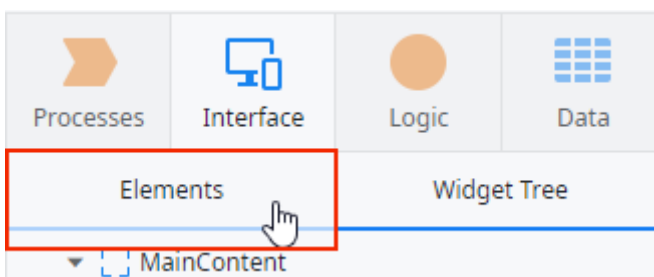
- 3) Right-click on the second column of the ColumnsSmallLeft and select Paste.



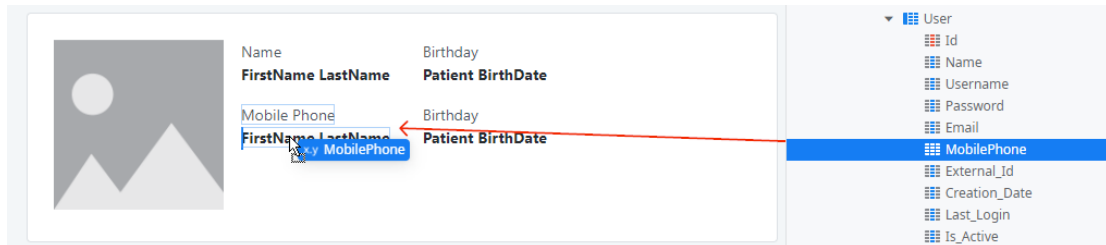
**Note:** the properties will be exactly the same!



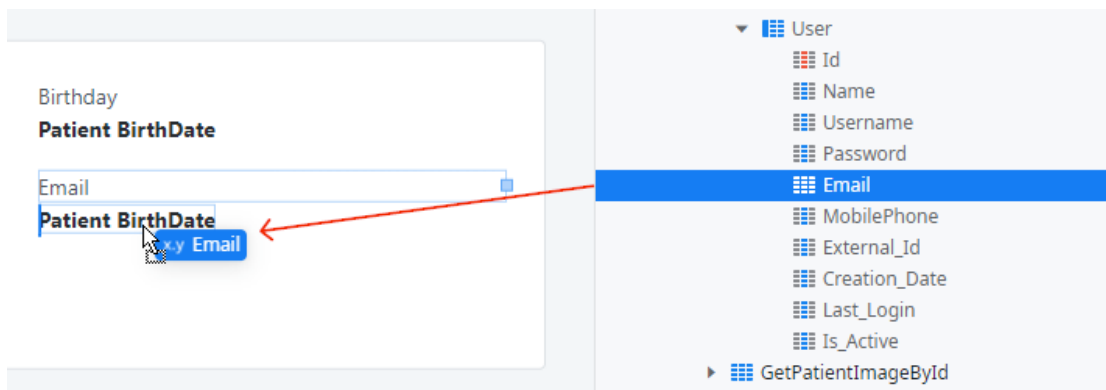
- 4) Change the text of the **Label** in the **first column** to *Mobile Phone*.
- 5) Go back to the **Elements** tab.



- 6) Expand the **User** Entity inside the **GetAppointmentsById** Aggregate, and drag the **MobilePhone** attribute on top of the Expression under the new Mobile Phone Label.



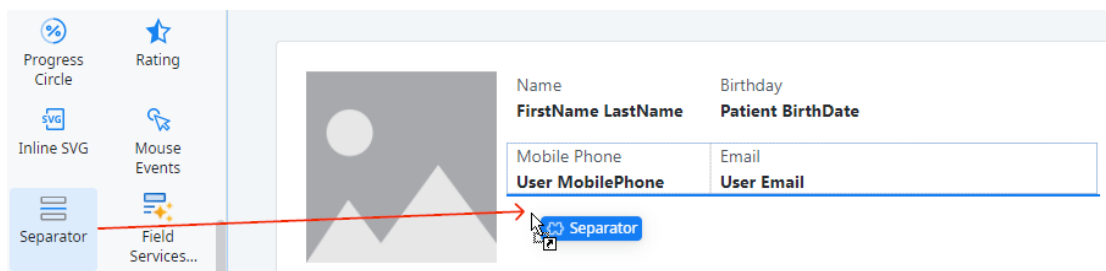
- 7) In the **second column**, change the **Label** to *Email*.
- 8) Now, drag the **Email** attribute of the **User** Entity on top of the Expression under the Email Label, just like you did above for the mobile phone.



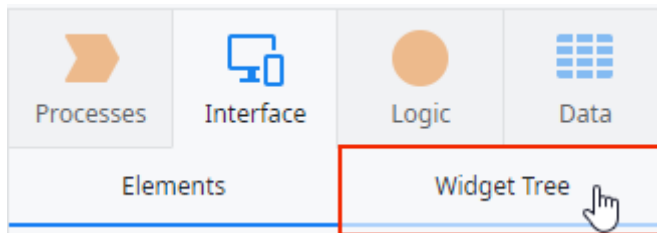
## Patient Onboarding Info

You are just missing the Patient Onboarding information at this point, which include the social security number and the insurance member id.

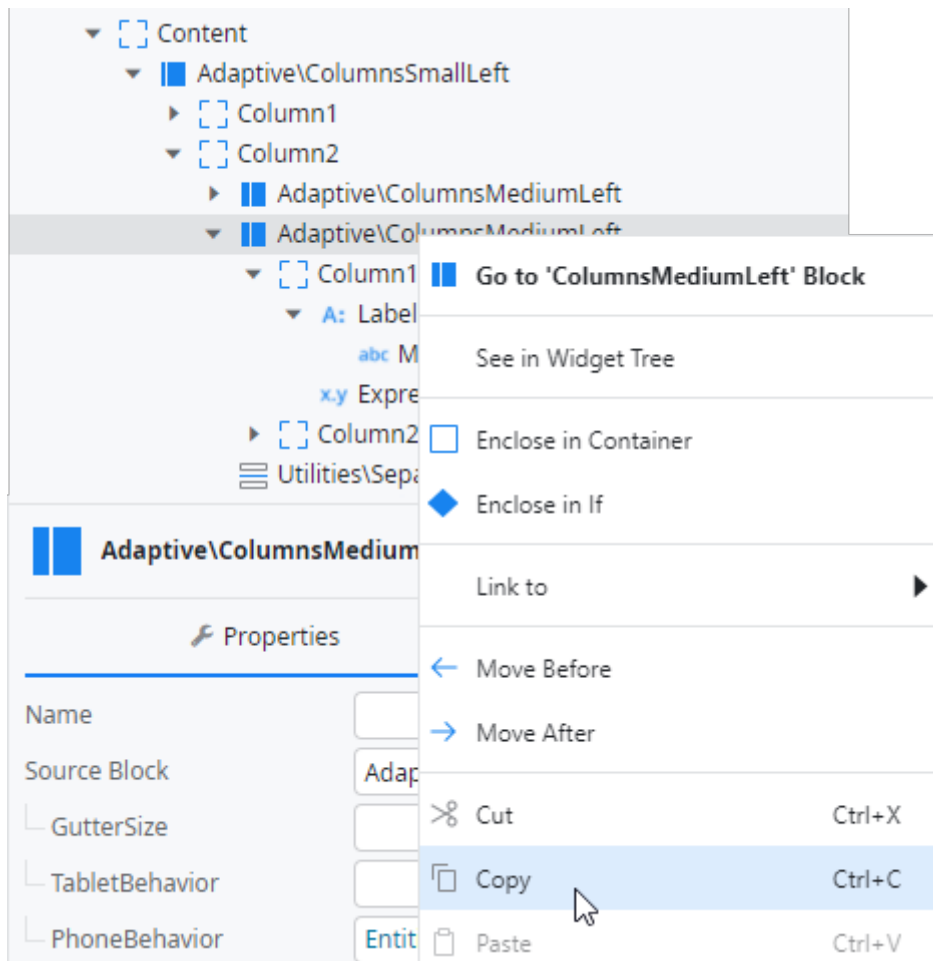
- 1) Drag and drop a **Separator** under the last columns created in the previous section.



- 2) Open the **Widget Tree**.

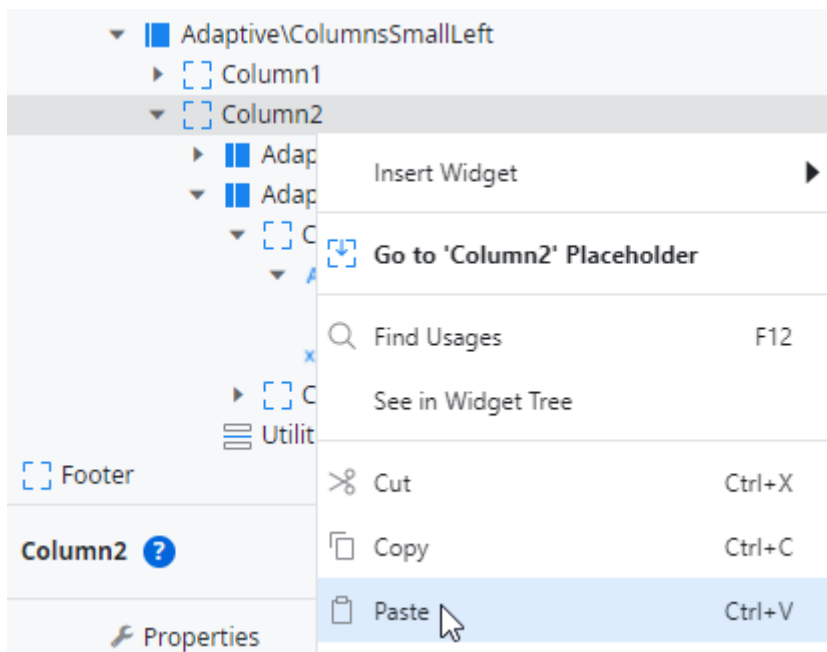


- 3) Right-click on the **ColumnsMediumLeft** element and select **Copy**.

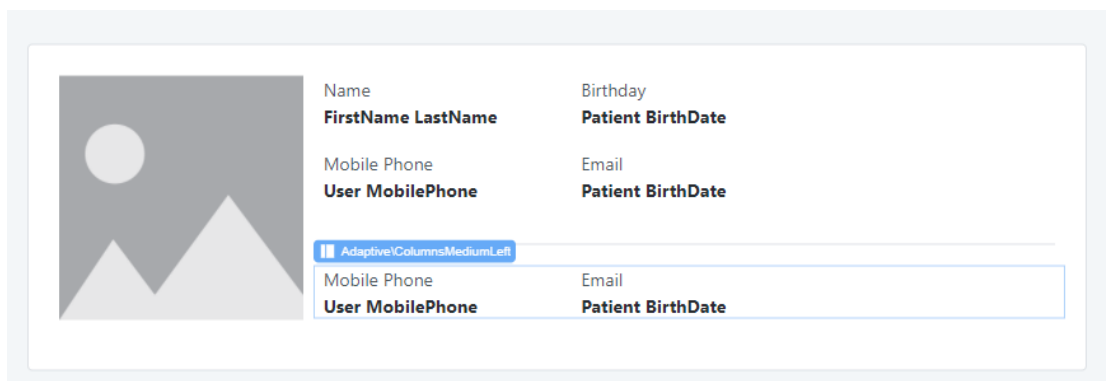


**Note:** it doesn't matter which one of the two ColumnsMediumLeft element you select, since they have the same look and feel.

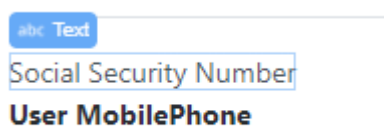
- 4) Right-click on the **Column2** inside the **ColumnsSmallLeft** element, then select **Paste**.



The Columns structure is ready, now you need to change the Labels and replace the Expressions like you did before.

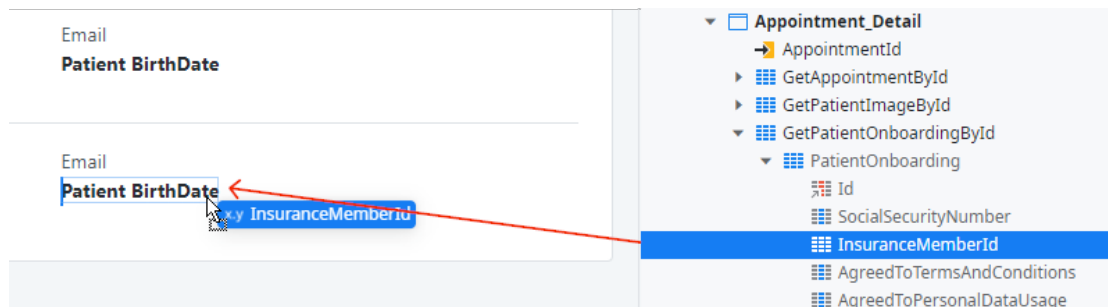


- 5) Type *Social Security Number* in the first Label.

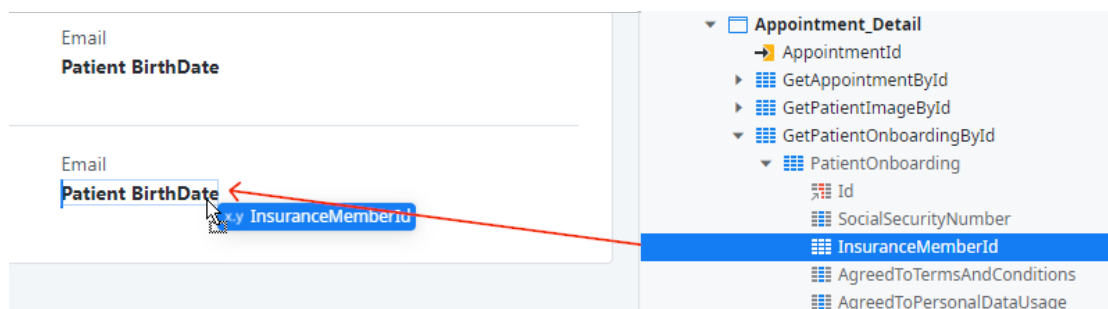


- 6) In the right sidebar, switch back from the Widget Tree to the **Elements** tab. Expand the **PatientOnboarding** Entity inside the **GetPatientOnboardingById**

Aggregate and drag the **SocialSecurityNumber** attribute on top of the Expression under the Social Security Number Label.



- 7) Type *Insurance Member* in the second Label.
- 8) Drag the **InsuranceMemberID** attribute on top of the Expression under the Insurance Member Label.



And now the Screen has all the information that it needs about the patient. You will come back to this Card in the next tutorial.

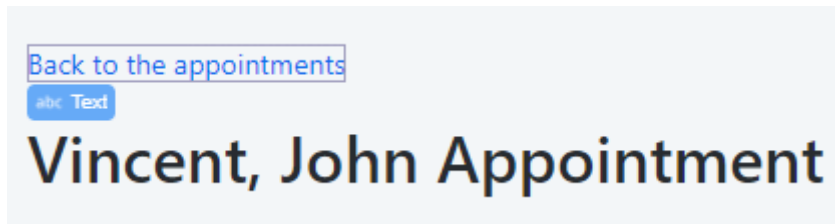
## Navigate Back to the Appointments List

So far, the Doctor can click on an Appointment and open the corresponding details. However, the Doctor might want to go back to the Appointments List. So you will need to add a link to the previous Screen.

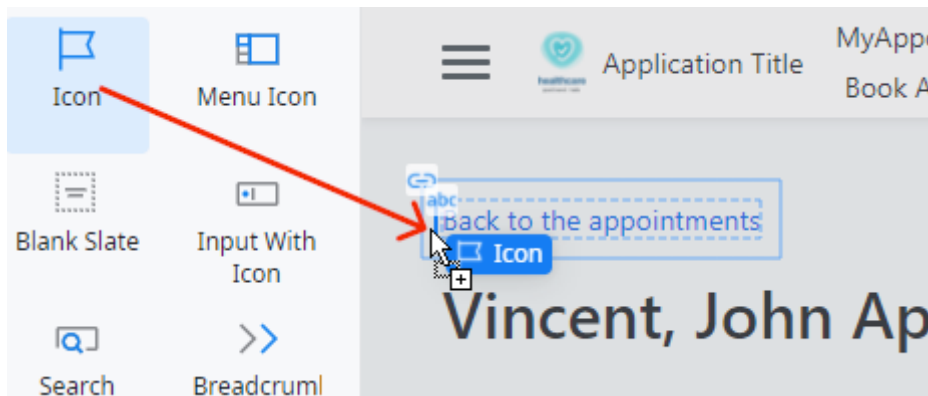
- 1) Still in the **Appointment\_Detail** Screen, drag and drop a **Link** inside the breadcrumbs placeholder, right above the Screen Title.



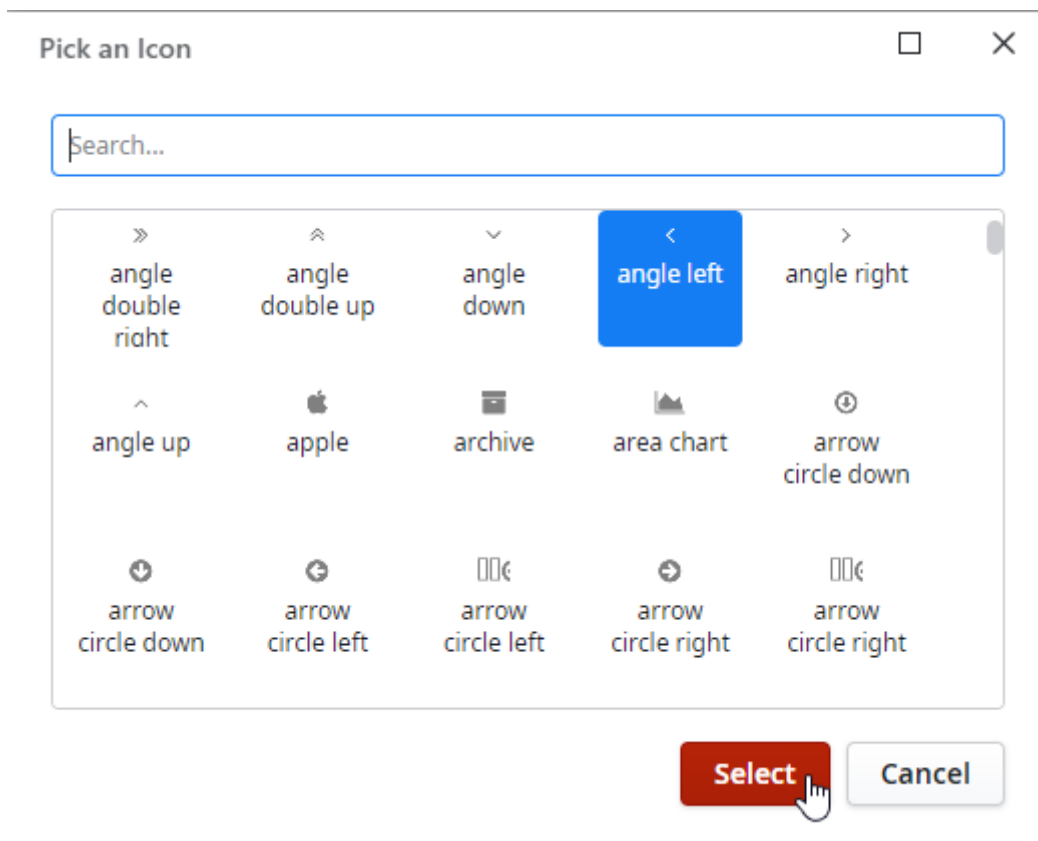
- 2) Then, type *Back to the Appointments* inside the Link.



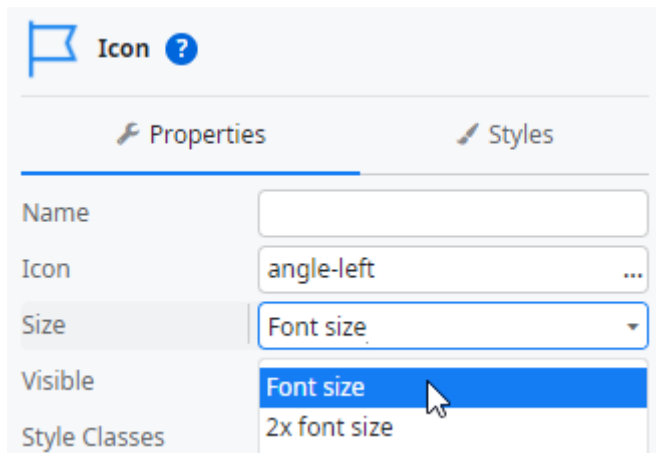
- 3) Drag and drop an **Icon** on the left of the Link's text.



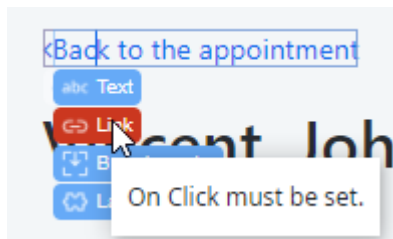
- 4) Select the **angle left** icon in the dialog and click **Select**.



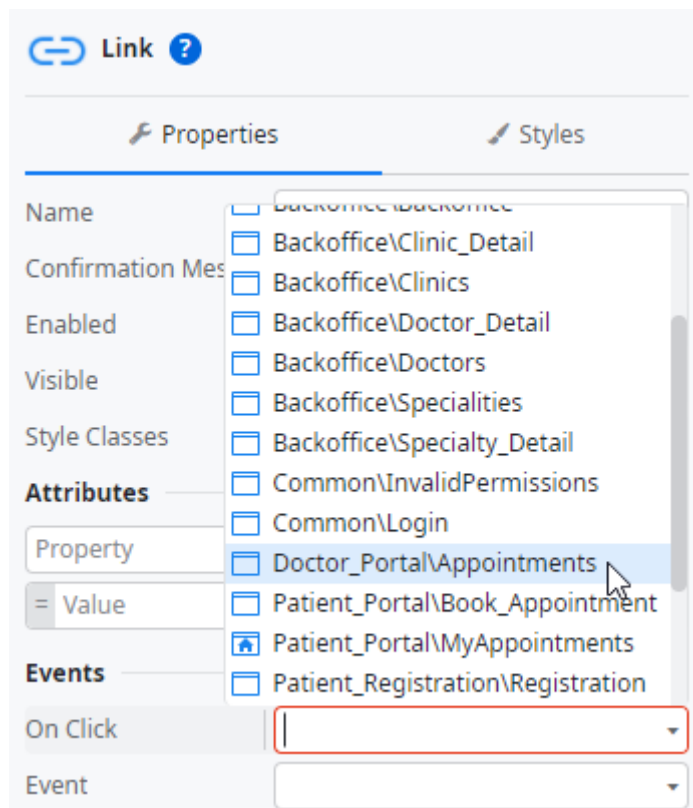
- 5) In the Icon's properties, set the **Size** to **Font size** to keep it proportional to the text.



- 6) Select the **Link** to open its properties.



- 7) Set the **OnClick** property to the **Appointments** Screen from the Doctor\_Portal Flow.



You're done, let's test the app!

## Testing and Results

- 1) Publish the module and open it in the browser.



- 2) Login as a patient, book an appointment with the doctor Ann Devon, then logout.
- 3) Log in as a doctor using the email and password below:
- Username: anndevon@gmail.com
  - Password: 1q2w3e4r



- 4) Click on the **Start Appointment** button.


### Appointments

Pick Appointment	Status ▾	Patient ▾	Date ▾	Start Time ▾	Clinic ▾	Specialty ▾
<b>Start Appointment</b>	Submitted	Patricia Wesley	16 Jul 2022	12:30	Main Medical Center	Dental
<b>Start Appointment</b>	Submitted	Ann Marie Pitt	21 Jul 2022	11:00	Main Medical Center	Dental
<b>Start Appointment</b>	Submitted	Krissa Tesena	30 Jul 2022	11:30	Main Medical Center	Dental

1 to 3 of 3 items

You will see the Patient Information Card and the custom Screen Title created in this tutorial.

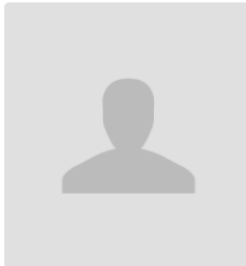
### Tesena, Krissa Appointment



Name	<b>Krissa Tesena</b>	Birthday	<b>1995-06-04</b>
Mobile Phone	<b>987456</b>	Email	<b>kris@tesena.com</b>
Social Security Number	<b>987456</b>	Insurance Member	<b>9874565</b>

Notice that the Screen layout would still be the same even if you select a patient without an image.

### Wesley, Patricia Appointment



Name	<b>Patricia Wesley</b>	Birthday	<b>2000-07-19</b>
Mobile Phone	<b>123456</b>	Email	<b>patricia@wesley.com</b>
Social Security Number	<b>1234567</b>	Insurance Member	<b>1234567</b>

## Wrapping up

Congratulations on finishing this tutorial. With this exercise, we had the chance to go through some essential aspects of OutSystems, and even get to know more about the platform.

## References

If you want to deep dive into the subjects that we have seen in this exercise, we have prepared some useful links for you:

- 1) [Image](#)
- 2) [Aggregate](#)
- 3) [Aggregates 101](#)
- 4) [Building Screens with Data](#)

**See you in the next tutorial!**