

# Appointments List

## Table of Contents

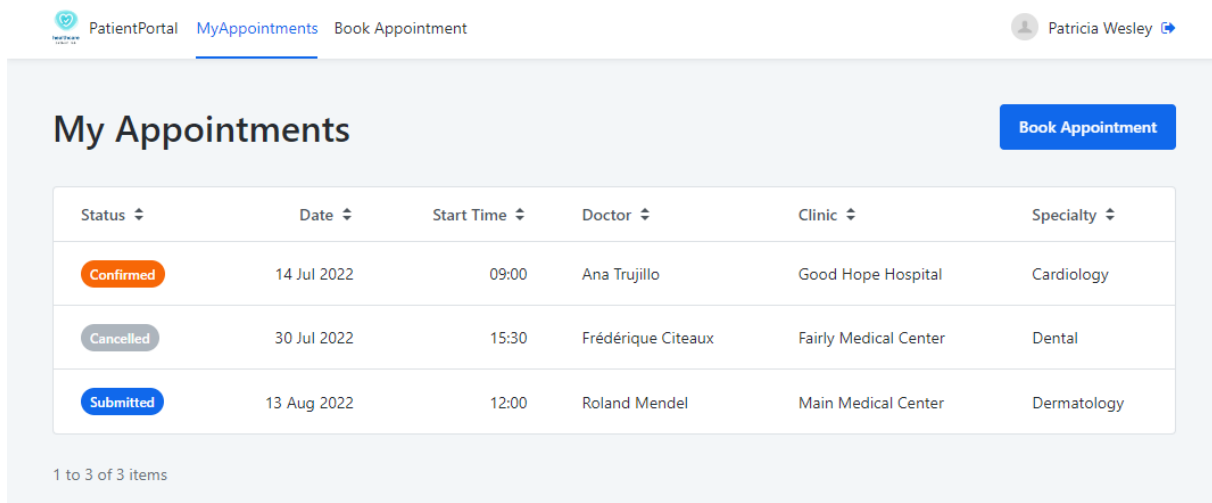
<b>Scenario.....</b>	<b>2</b>
<b>How to.....</b>	<b>4</b>
Patient's List of Appointments	4
Create the MyAppointments Screen	4
Redirecting the Login	7
List of Appointments	10
Customizing the List of Appointments	12
Fetching the Appointments	17
What if there are no Appointments? - UI	22
Schedule a New Appointment	27
Redirecting to the MyAppointments Screen	32
Menu	34
Testing	38
<b>Wrapping up.....</b>	<b>39</b>
References	39

## Scenario

In the previous tutorials, you worked on the Registration process, so your application should allow a user to register as a patient at this point.

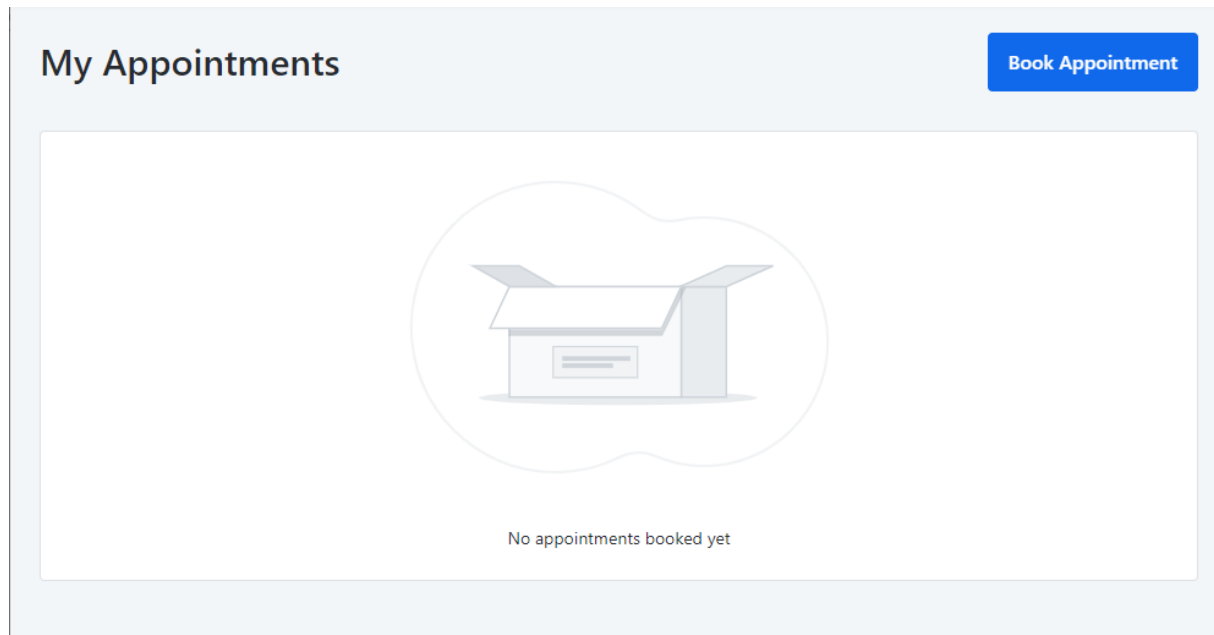
Also, a patient can book an appointment to a clinic, for a given speciality and choose the preferred doctor. However, the user cannot see its list of appointments yet! That's what you will work on during this tutorial.

The objective is to create a new Screen to list the Patient's appointments. The list should be displayed in a tabular layout and should show the status of the appointment, the date, the start time, the doctor, the clinic and the specialty. The Screen should be the default Screen of the app and when a patient logs in, they should be redirected to this Screen.



Like the image shows, the Screen should also have a button to allow the patient to book a new appointment, thus it should navigate to the Book Appointment Screen. When the appointment is confirmed, the user should return to its list of appointments.

When there are no appointments, the Screen should display some visuals indicating that there are no appointments created yet.



Finally, the Menu should have an option to access this My Appointments Screen quickly. However, this option should only be available to users with the Patient role.

# How to

Now that you know the scenario, let's implement it by following this how-to guide!

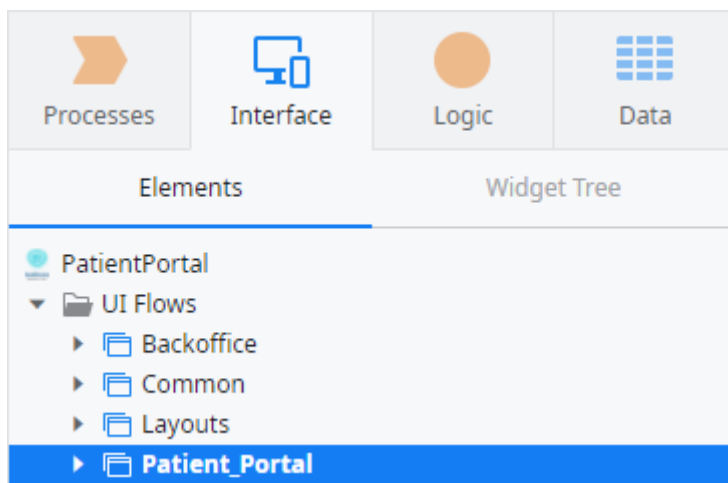
## Patient's List of Appointments

The list of appointments will be displayed in a new Screen of the application. When a patient logs in, the app should redirect them to this new Screen, where the patient will be able to see the list of scheduled appointments. So, in this section, you will create an empty Screen and define the logic of the Login Screen to properly redirect the user after logging in.

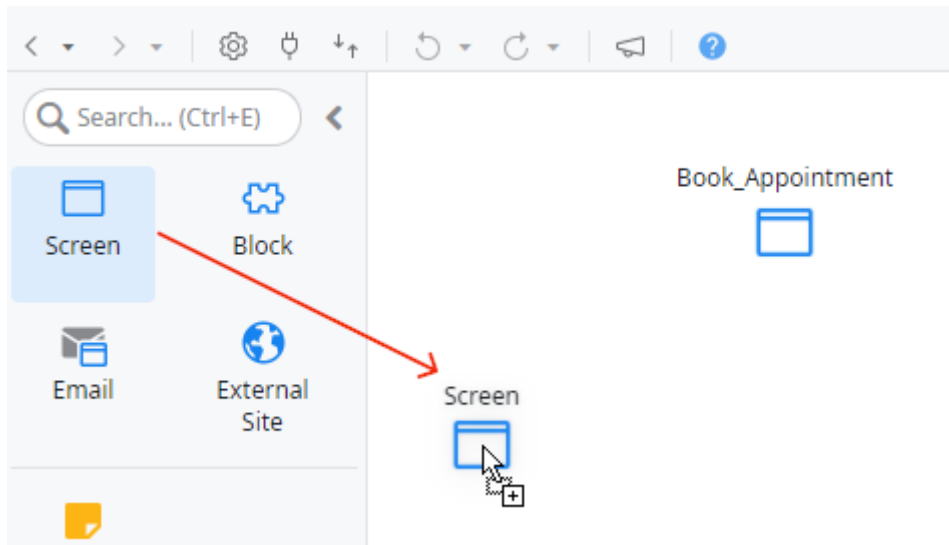
## Create the MyAppointments Screen

Let's start by creating the MyAppointments Screen.

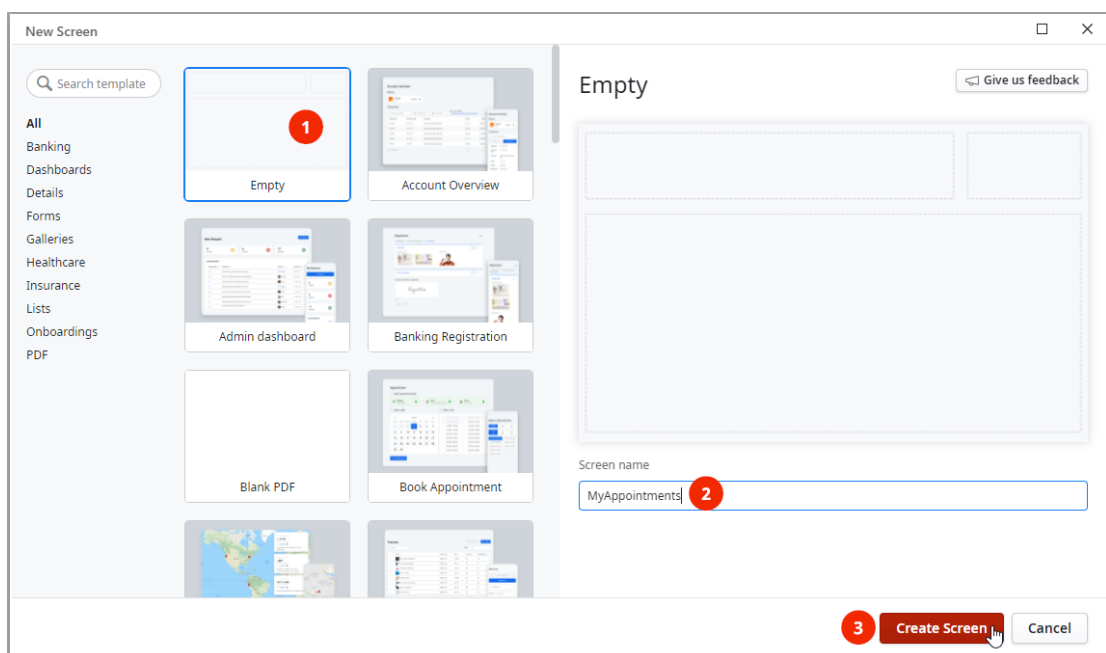
- 1) In the Interface tab, double-click on the **Patient\_Portal** UI Flow to open it.



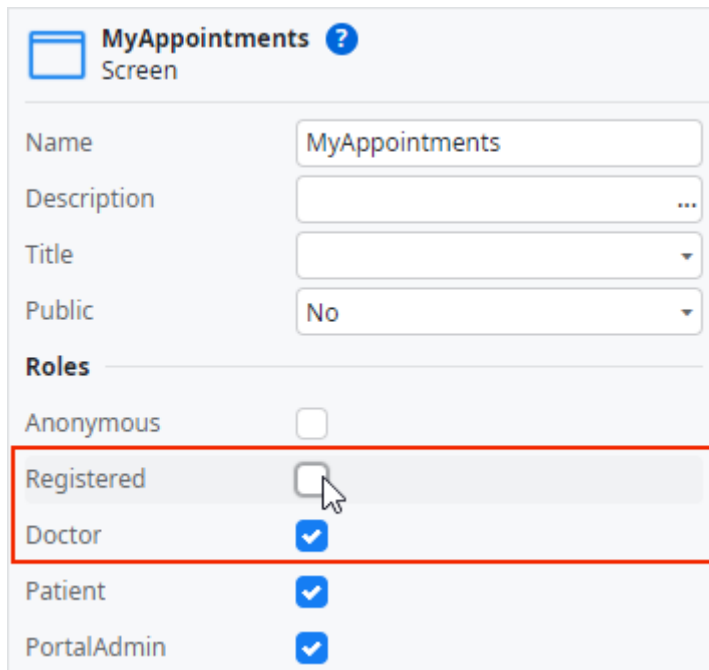
- 2) Drag a **Screen** from the left sidebar and drop it in the flow.



- 3) Select an **Empty** Screen, type *MyAppointments* in the Screen name, then click on **Create Screen**.



- 4) Go to the Screen's properties and uncheck the **Registered** and the **Doctor** roles.



**MyAppointments** ?  
Screen

Name: MyAppointments

Description: ...

Title: ...

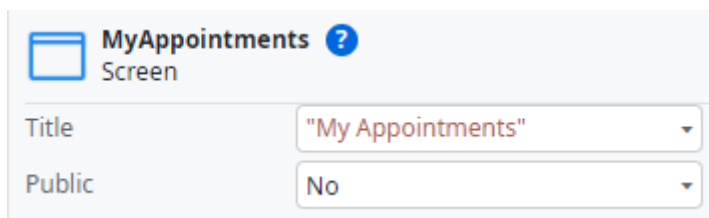
Public: No

**Roles**

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

This means that only users that have the Patient and PortalAdmin roles can access this Screen.

- 5) Still in the **MyAppointments** Screen properties, type *"My Appointments"* in the **Title**.

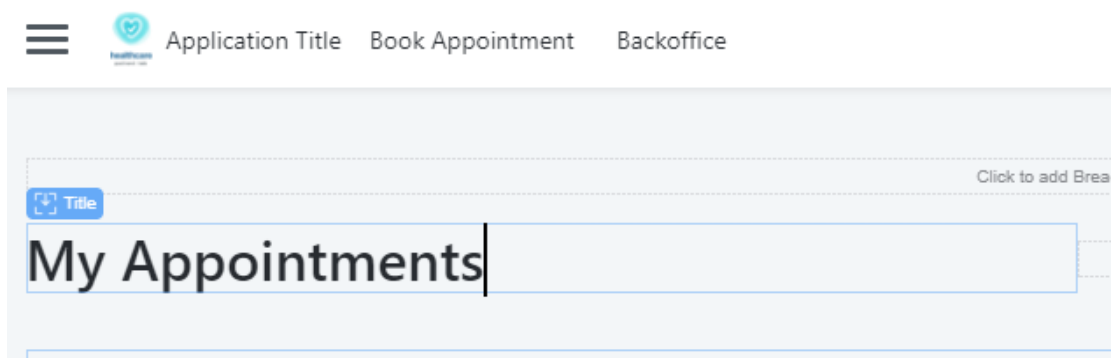


**MyAppointments** ?  
Screen

Title: "My Appointments"

Public: No

- 6) Now, on the Screen itself, click on the Title section and type *My Appointments*.



Application Title Book Appointment Backoffice

Click to add Break

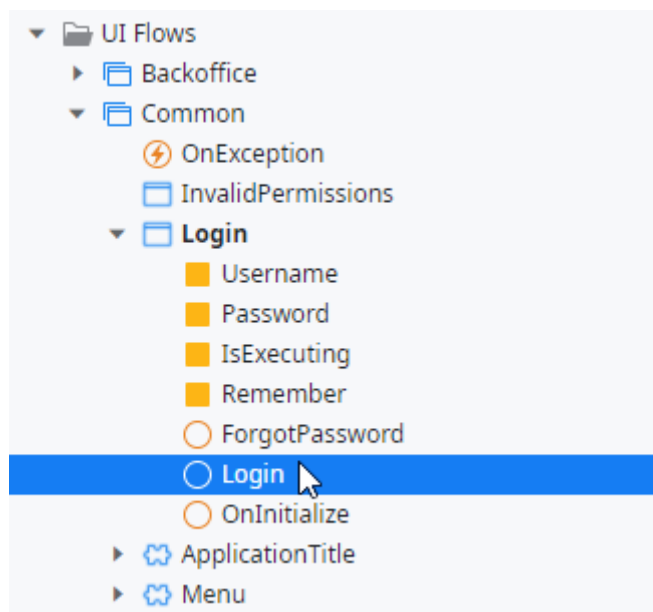
**Title**

My Appointments

## Redirecting the Login

Whenever a user logs in, the logic behind the scenes redirects the user to a Screen or an external URL. Here, you want to have the MyAppointments Screen to be the first Screen the patient sees after the login. For that, you need to go to the Login Screen and adjust its logic.

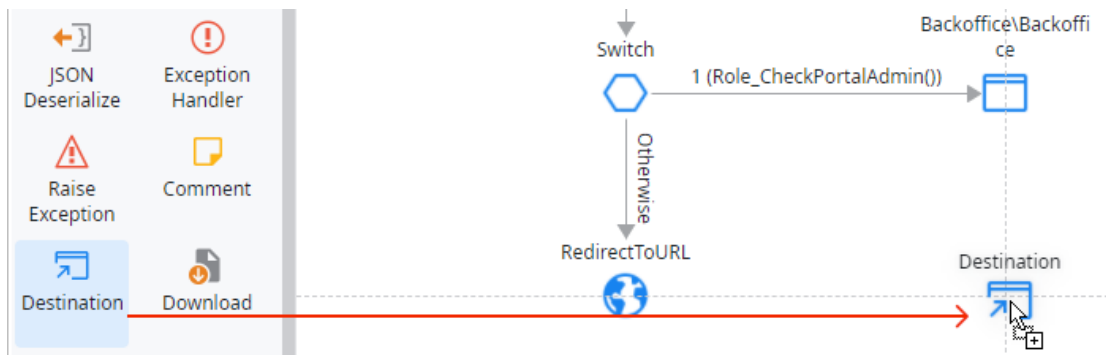
- 1) In the Interface tab, expand the **Common** UI Flow, then the **Login** Screen, and double-click on the **Login** Client Action to open it.



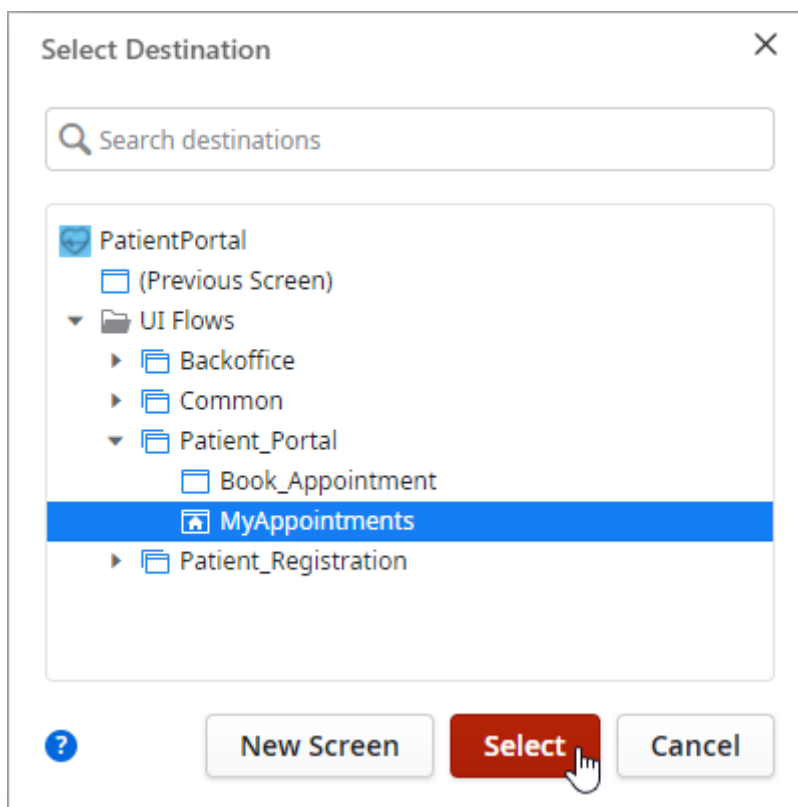
This Client Action defines what happens when a user successfully logs in. So far, if a user has the PortalAdmin Role, it redirects them to the Backoffice Screen. Now we want to add another option: if a user has the Patient role, they should be redirected to the My Appointments Screen.

**Note:** The Switch element splits the Action flow into two or more paths. Each path has its own condition and the flow of the Action follows the first condition that is true. If no condition is evaluated to True, the Otherwise branch is followed. It's like an if, but with more options!

- 2) Drag a **Destination** node from the left sidebar and drop it to the right of the RedirectToURL node.

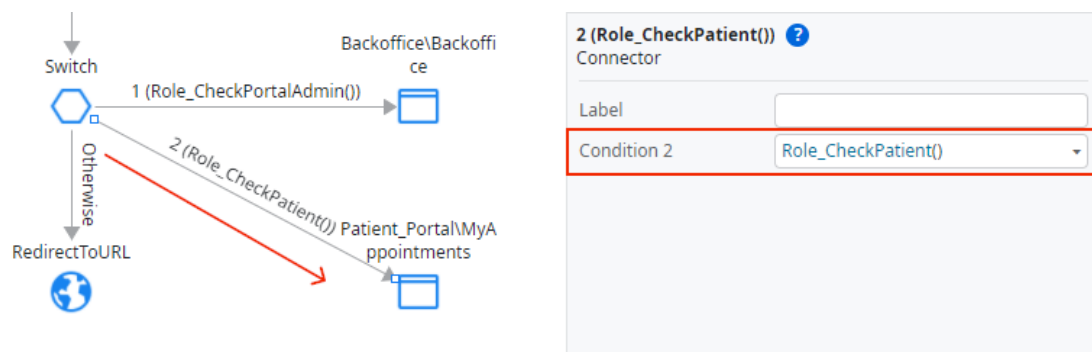


- 3) Select the **MyAppointments** Screen inside the Patient Portal UI Flow in the dialog.



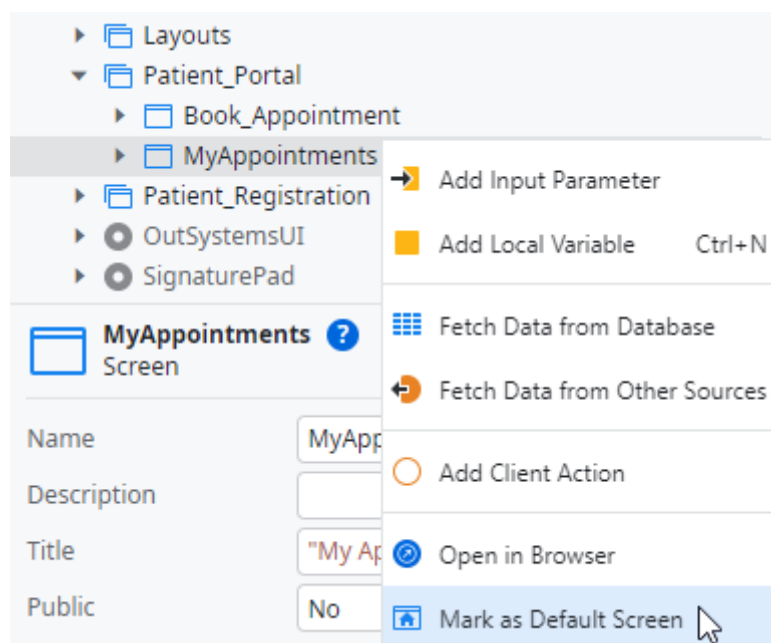


- 4) Connect the Switch to the Destination and add the the function *Role\_CheckPatient()* to the **Condition 2**.



This function was created for you in the quickstart of this app and it checks if the currently logged in user has the Patient Role.

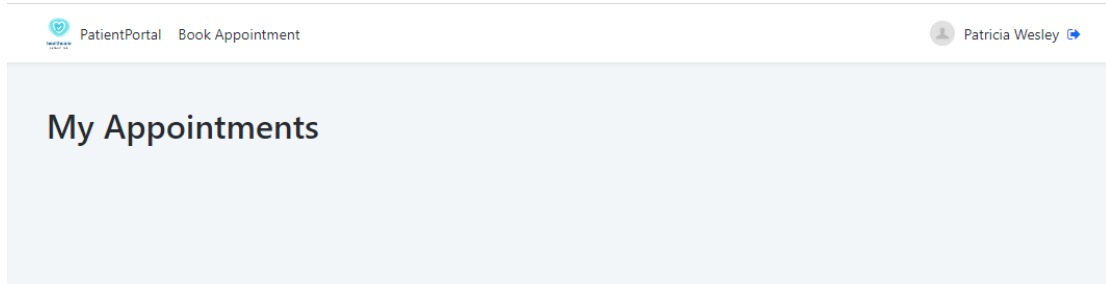
- 5) Mark the MyAppointments Screen as the **Default Screen**.



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



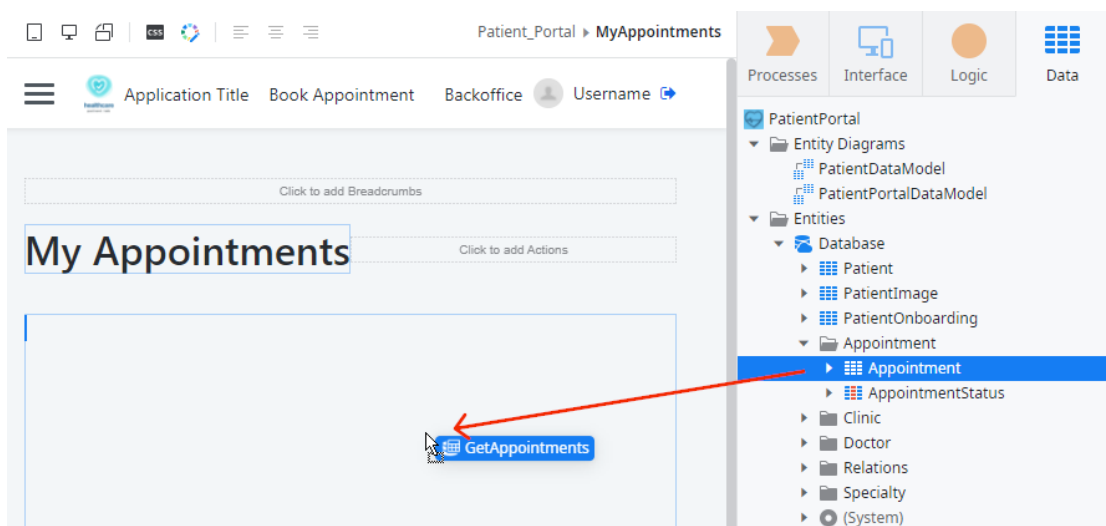
- 7) Login with a user you have created in the previous tutorials to see the MyAppointments Screen. If you were already logged in, logout and login again to see the redirect happening.



## List of Appointments

So far, the Screen only has a title. In this section, you will create the list with all the appointments of a patient, including a pagination functionality, with just one drag and drop!

- 1) Open the **MyAppointments** Screen by double-clicking on it. Also, switch to the Data tab on the top right corner of Service Studio.
- 2) Expand the **Appointment** folder, select the **Appointment** Entity, then drag and drop it into the main content area of the Screen.



A popup will appear so you can select which attributes you want to use. This is going to define which attributes of the Appointment Entity will be displayed on the Screen. By default, all attributes are selected. But let's not show them all. For instance, there's no need to display the user, since all of the appointments in this Screen belong to the same User.

- 3) Unselect all attributes except for the **DoctorId**, **Date**, **StartTime**, **StatusId**, **ClinicId**, and **SpecialtyId**, then click on the **Select** button.

Select data □ ×

Appointment/Appointment

☐ UserId

☒ DoctorId

☒ Date

☒ StartTime

☒ StatusId

☒ ClinicId

☒ SpecialtyId

☐ Diagnosis

☐ IsActive

A table with the list of appointments will appear, displaying the attributes you selected. Notice how the columns already have a title and sorting functionality. Also, the pagination is created automatically. Awesome, right?

abc Text

## My Appointments

Doctor ↕	Date ↕	Start Time ↕	Status ↕	Clinic ↕	Specialty ↕
Yoshi Latimer	Date	StartTime	Submitted	Main Medical Center	Pediatrics
Yoshi Latimer	Date	StartTime	Submitted	Main Medical Center	Pediatrics
Yoshi Latimer	Date	StartTime	Submitted	Main Medical Center	Pediatrics

No items to show...

1 to 10 of 15 items

<

1

...

50

>

## Customizing the List of Appointments

The table created automatically saved us a lot of time. But it doesn't mean you cannot modify it and adapt it to your requirements!

- 1) Click on the Header of the **Status** column to select it. Then, click three times on the arrow pointing left on the Toolbar above the Screen preview.

The screenshot displays a web application interface for "My Appointments". The top navigation bar features a hamburger menu icon, a search bar, and a user profile icon labeled "Username". The main content area shows a table of appointments. Red circles and arrows highlight specific elements: a red circle with the number "2" points to the left arrow icon in the top navigation bar, and a red circle with the number "1" points to the "Header Cell" label above the "Status" column in the table.

Navigation Bar: Patient\_Portal ▶ MyAppointments

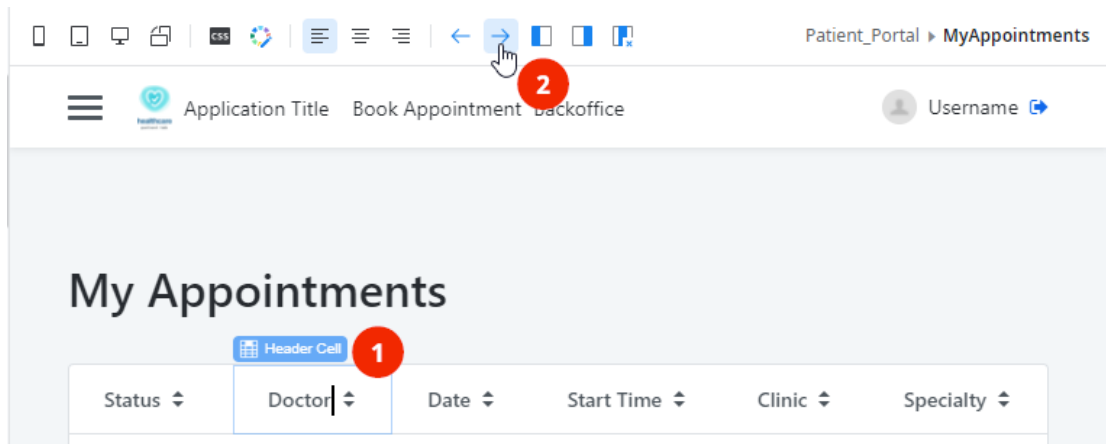
Application Title Book Appointment Backoffice

# My Appointments

Doctor	Date	Start Time	Status	Clinic	Specialty

**Note:** the Toolbar has other options that you can use to customize your tables. You can delete, move, and even create columns.

- Now use the arrow pointing right to move the **Doctor** column after the Start Time.



You can change the columns order however you want! This is our proposal:

### My Appointments

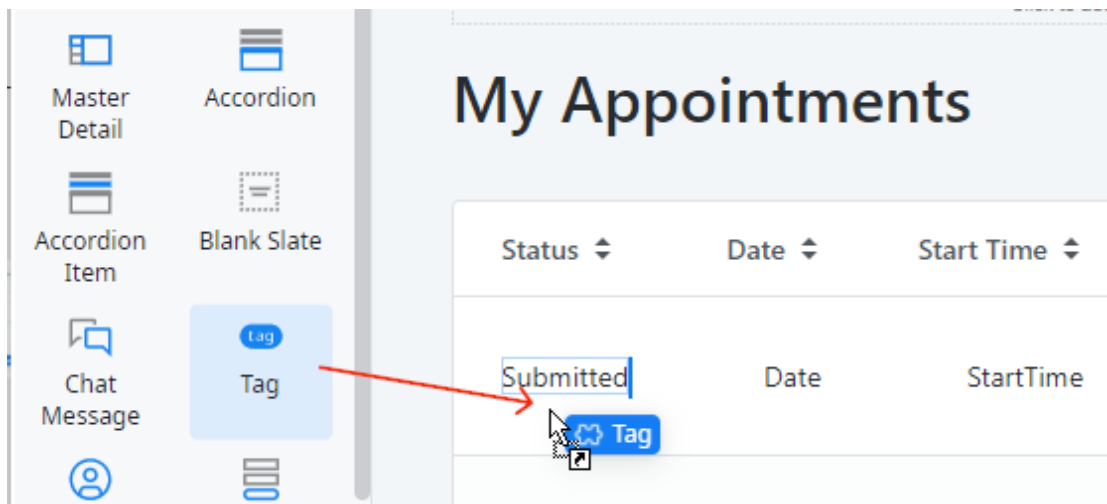
Status	Date	Start Time	Doctor	Clinic	Specialty
Submitted	Date	StartTime	Yoshi Latimer	Main Medical Center	Pediatrics
Submitted	Date	StartTime	Yoshi Latimer	Main Medical Center	Pediatrics
Submitted	Date	StartTime	Yoshi Latimer	Main Medical Center	Pediatrics

No items to show...

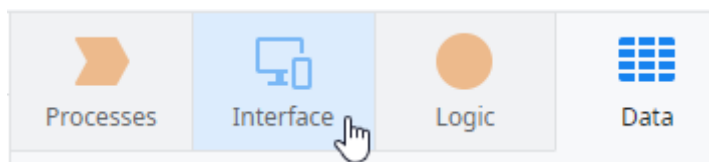
1 to 10 of 15 items

Now let's use a color code to identify the different appointment statuses.

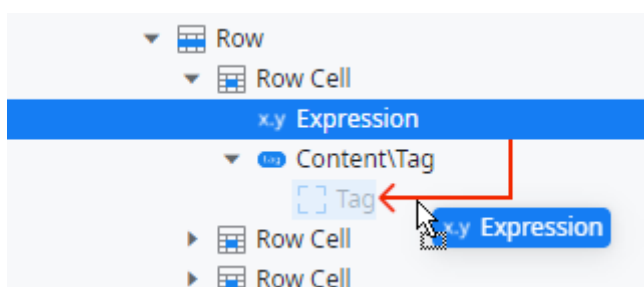
- 3) Drag a **Tag** element from the left sidebar and drop it inside the first element of the Status column.



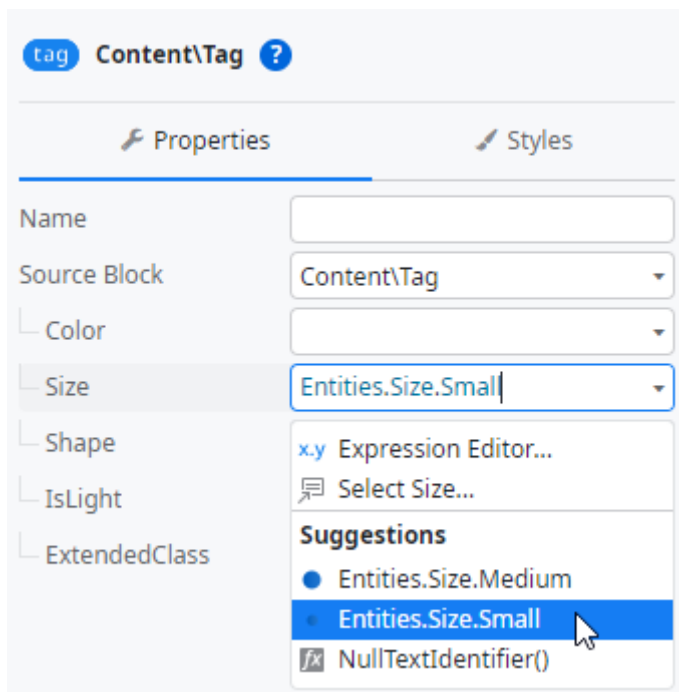
- 4) Click on the **Interface** tab, then open the Widget Tree.



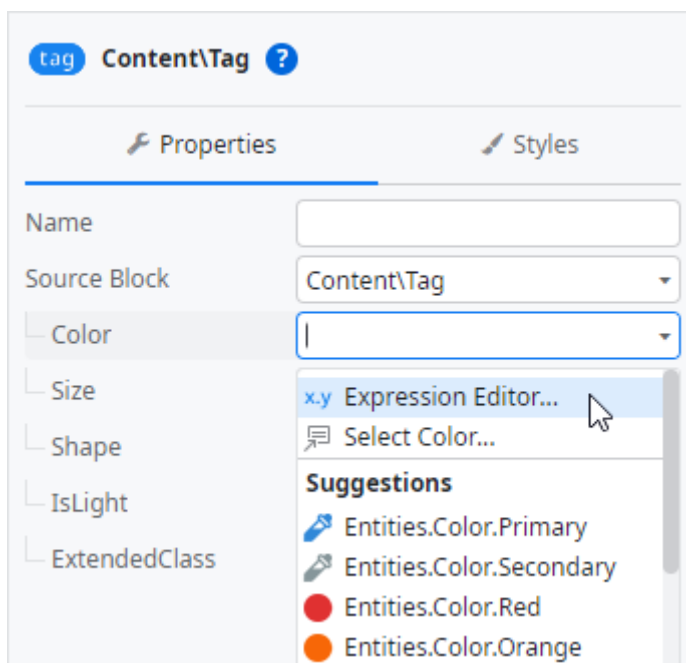
- 5) Select the Expression that contains the Status, then drag and drop it inside the placeholder in the Tag widget.



- 6) Still using the Widget Tree, click on the Tag element to open its properties, then click on the dropdown of the **Size** property and select **Entities.Size.Small**.

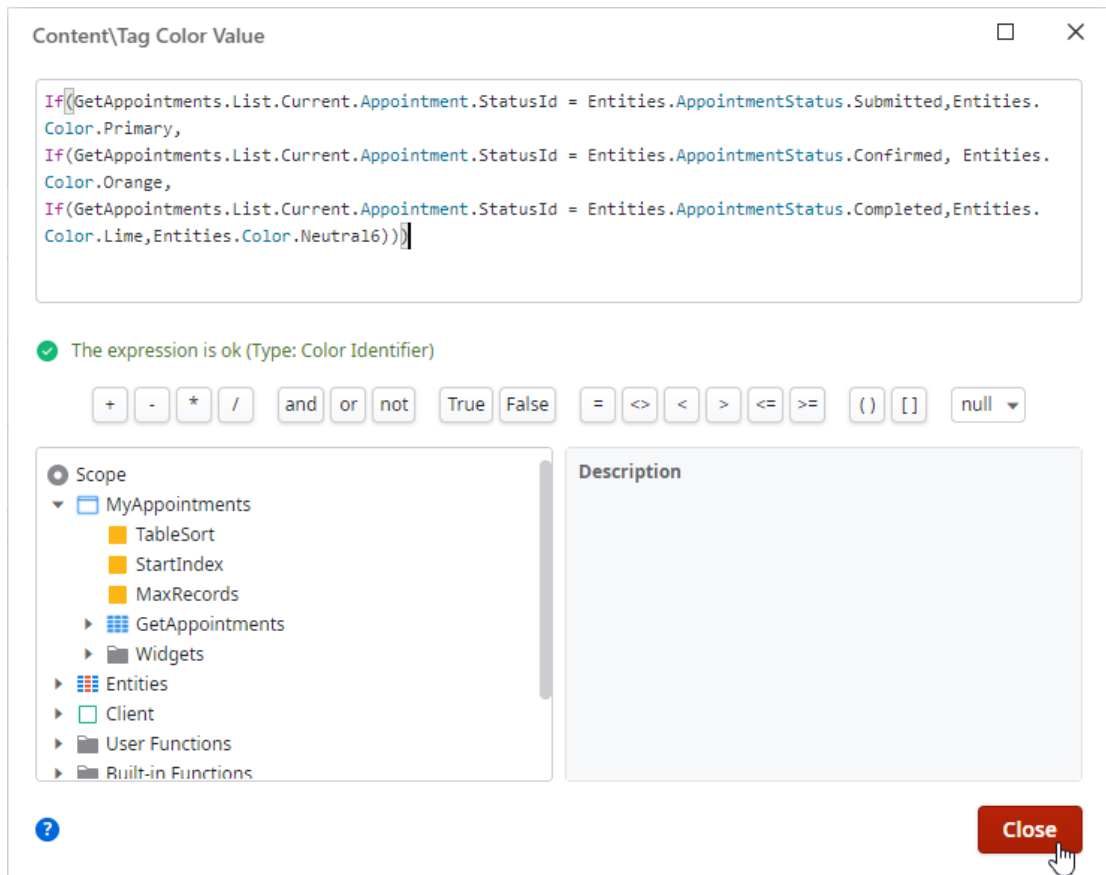


- 7) Click on the **Color** property and select **Expression Editor....**



- 8) Copy the code below and paste it in the Color Value dialog, then click on Close.

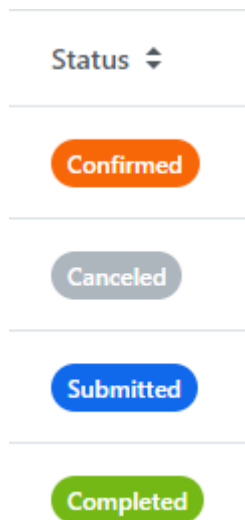
```
If(GetAppointments.List.Current.Appointment.StatusId =
Entities.AppointmentStatus.Submitted,Entities.Color.Primary,
If(GetAppointments.List.Current.Appointment.StatusId =
Entities.AppointmentStatus.Confirmed, Entities.Color.Orange,
If(GetAppointments.List.Current.Appointment.StatusId =
Entities.AppointmentStatus.Completed,
Entities.Color.Lime,Entities.Color.Neutral6)))
```



This code may seem long, but it is actually pretty simple. If the Status is Submitted, the color will be the same as the primary color of the app, which is blue. If the Status is confirmed, it will be orange. If the status is completed it will have a lime green color. Otherwise, the colors will be neutral / grey. These colors are already built-in the platform.



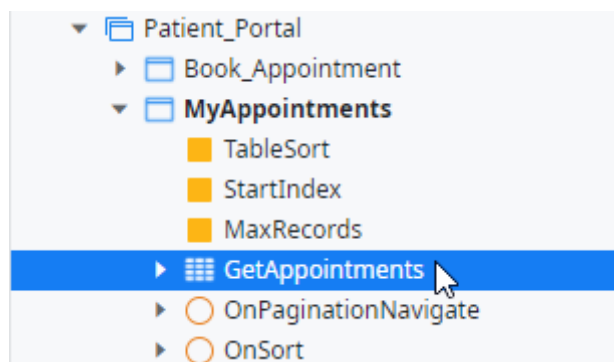
This is how the tags with different statuses are going to look like when the app is ready:



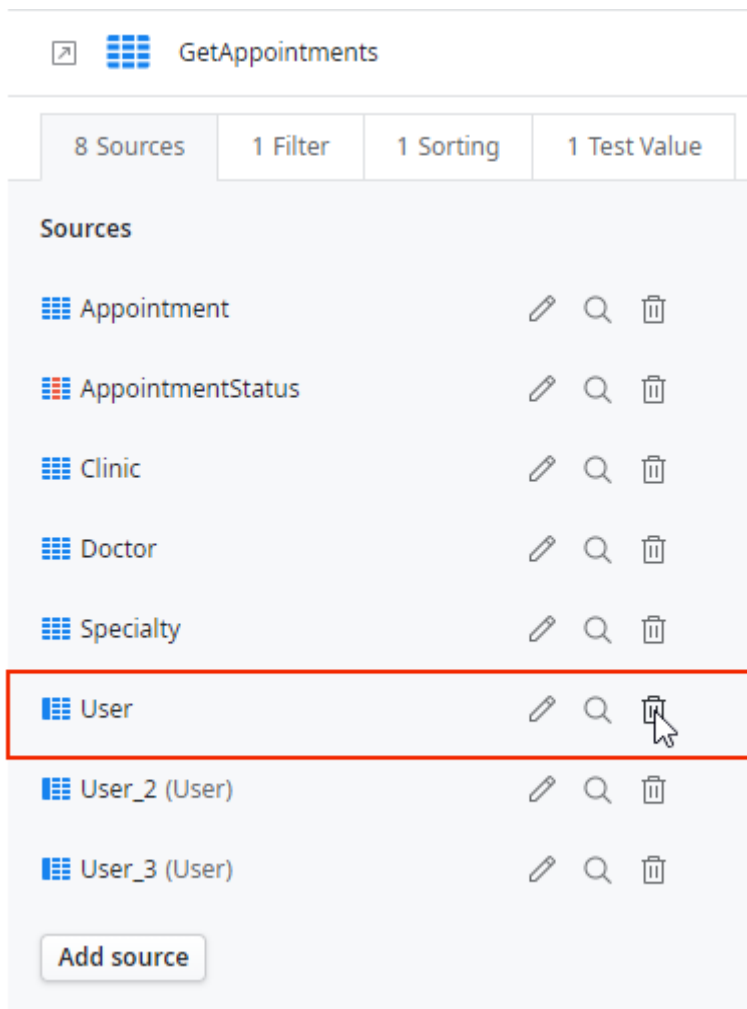
## Fetching the Appointments

The **GetAppointments** Aggregate was created automatically when you dragged the Entity to the Screen. This Aggregate is fetching all appointments from the database. But you don't want that. You just want the appointments of the patient that is currently logged in.

- 1) Switch back from the Widget Tree to the Elements tab, expand the **MyAppointments** Screen and double-click on the **GetAppointments** Aggregate to open it.



- 2) Click on the trash can icon next to the **User** Entity in the Aggregate's Sources section to delete it.













- 3) Do the same for the **User\_2** and **User\_3**.

A few error messages will appear since these Sources are being used in some Joins.

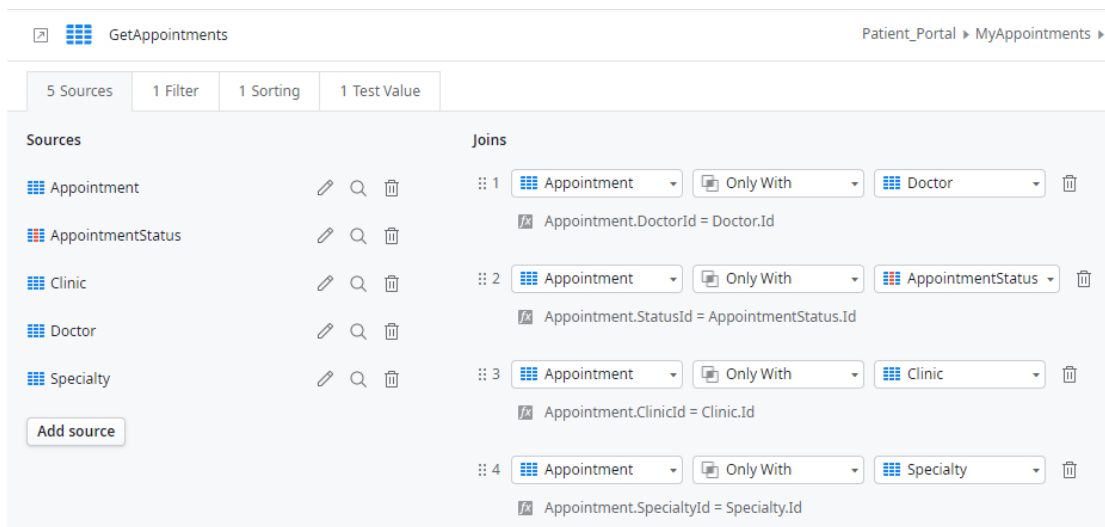
- 4) Click on the trash can icon next to each of the Joins with errors to remove them.

**Joins**

:: 1	Appointment	Only With	Select Source	
fx  Appointment.UserId = User.Id				
:: 2	Appointment	Only With	Doctor	
fx Appointment.DoctorId = Doctor.Id				
:: 3	Appointment	Only With	AppointmentStatus	
fx Appointment.StatusId = AppointmentStatus.Id				
:: 4	Appointment	Only With	Clinic	
fx Appointment.ClinicId = Clinic.Id				
:: 5	Appointment	Only With	Specialty	
fx Appointment.SpecialtyId = Specialty.Id				
:: 6	Appointment	With or Without	Select Source	
fx  Appointment.CreatedBy = User_2.Id				
:: 7	Appointment	With or Without	Select Source	
fx  Appointment.ModifiedBy = User_3.Id				

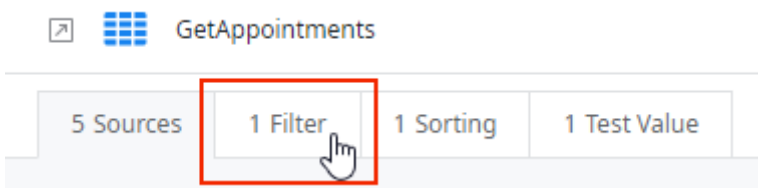
**Add join**

This is how the GetAppointments Sources section should look like:

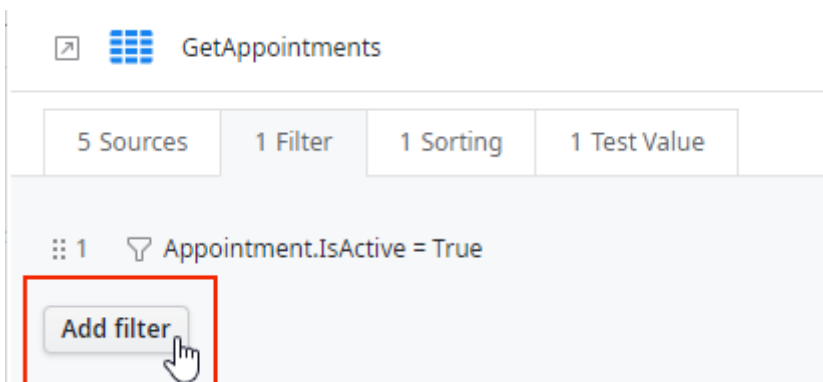


Now let's add a filter so the aggregate only retrieves the appointments of the user logged in.

- 5) Click on the **Filter** tab.

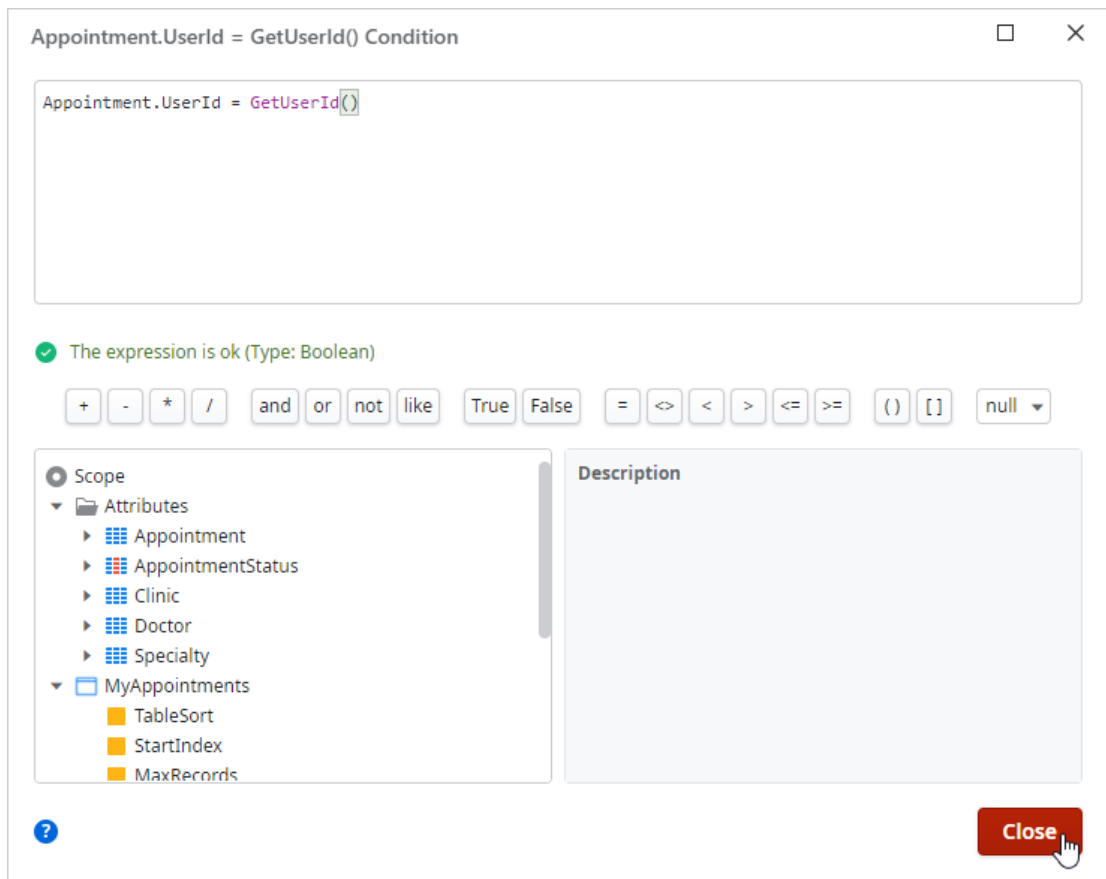


- 6) Click on the **Add filter** button.



- 7) Add the following condition, to make sure you just get appointments with a UserId that matches the user currently logged in, then click on Close.

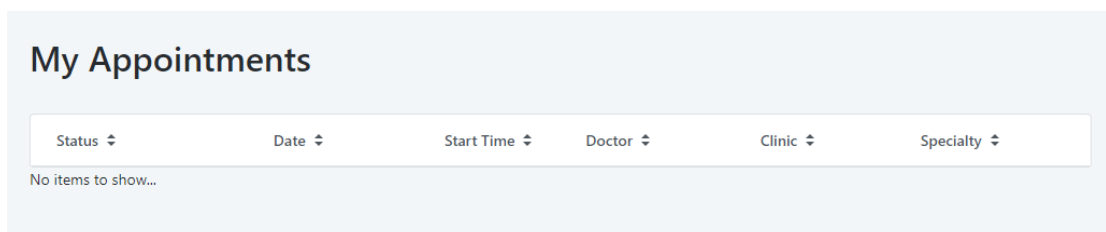
```
Appointment.UserId = GetUserId()
```



8) Publish the module and open it in the browser.



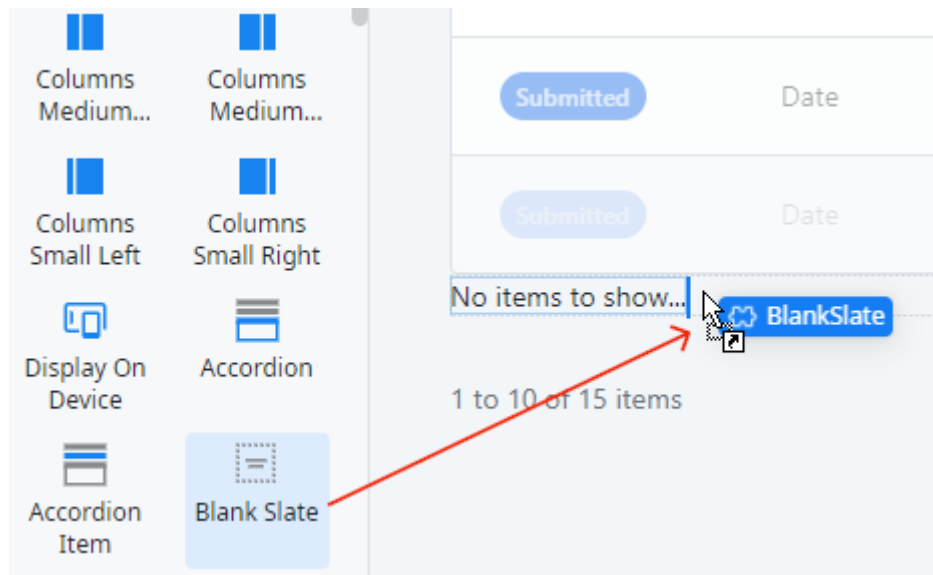
Navigate to the MyAppointments Screen. Do you have any appointments yet? If you don't, the interface is a bit boring, isn't it? Let's work on that!



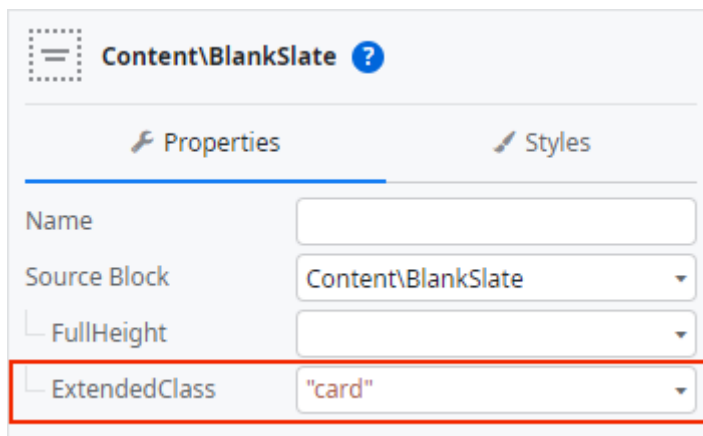
## What if there are no Appointments? - UI

Now let's add a Blank Slate to the MyAppointments Screen. You will replace the *No items to show...* text with something a bit more elaborate.

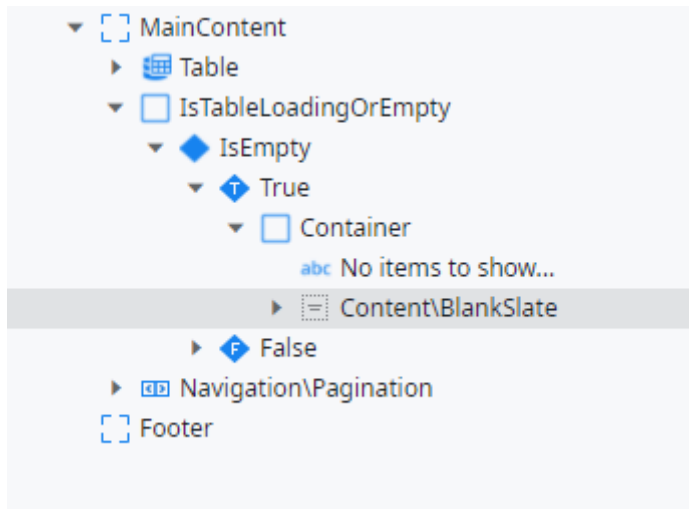
- 1) Drag a **Blank Slate** from the left sidebar and drop it on the Screen, right after the *No items to show...* text.



- 2) In the Blank Slate properties, set the **ExtendedClass** property to "card".

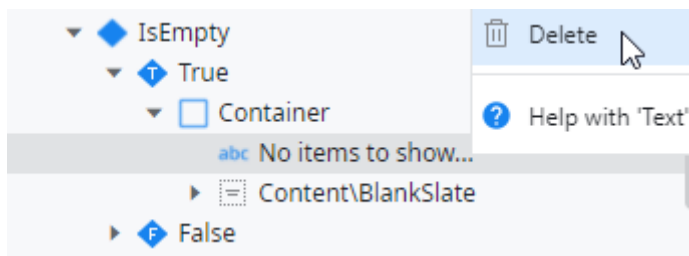


- 3) Open the Widget Tree to see the page structure.

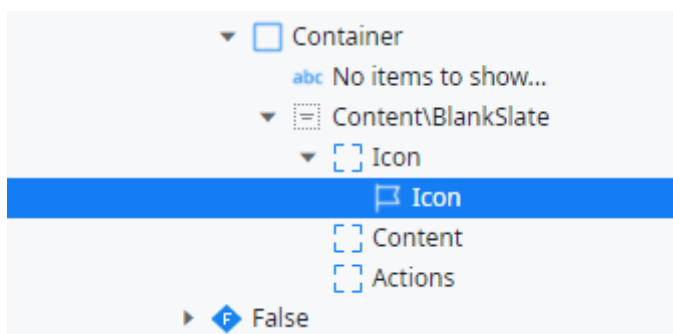


The Blank Slate is in the True branch of the IsEmpty If. This means it will only appear if the Appointments table is empty.

- 4) Right-click on the *No items to show...* text and delete it.

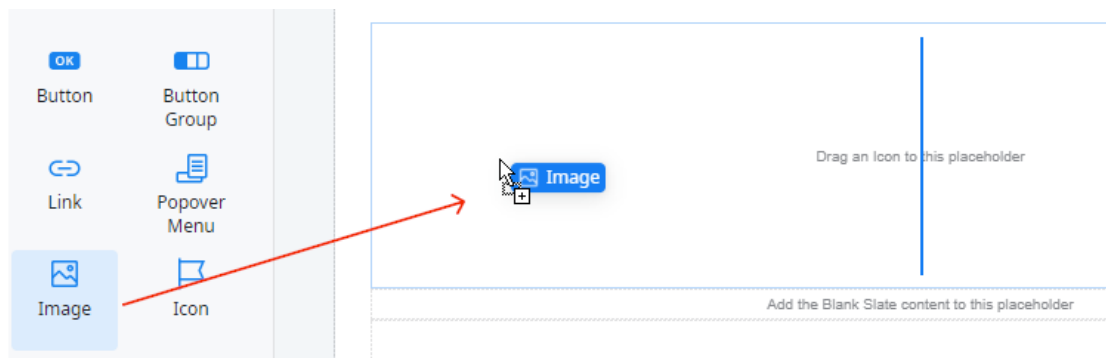


- 5) Now, expand the Blank Slate to see the structure inside it. You will see three placeholders: one for the Icon, one for Content and one for Actions.

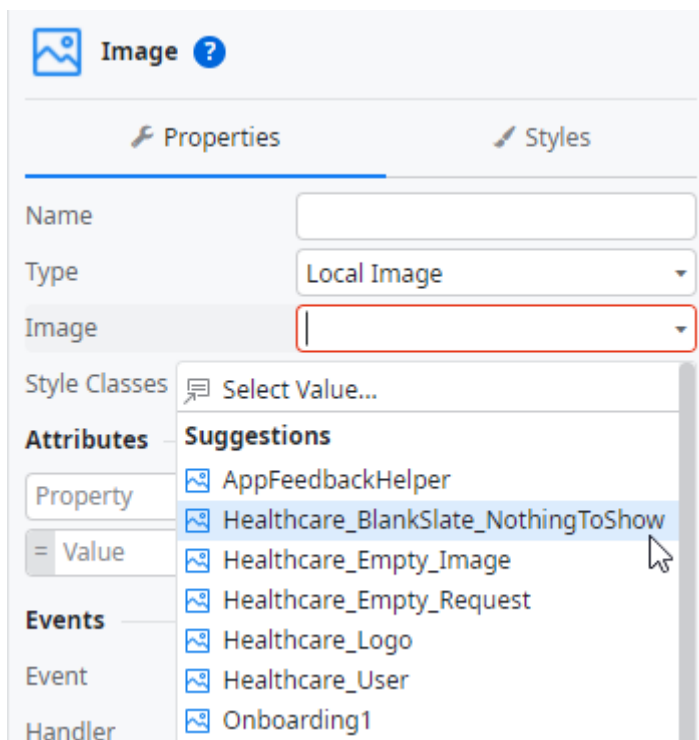


- 6) Delete the **Icon** element inside the Icon placeholder.

- 7) Drag and drop an **Image** inside the Icon placeholder (where the Icon was before).

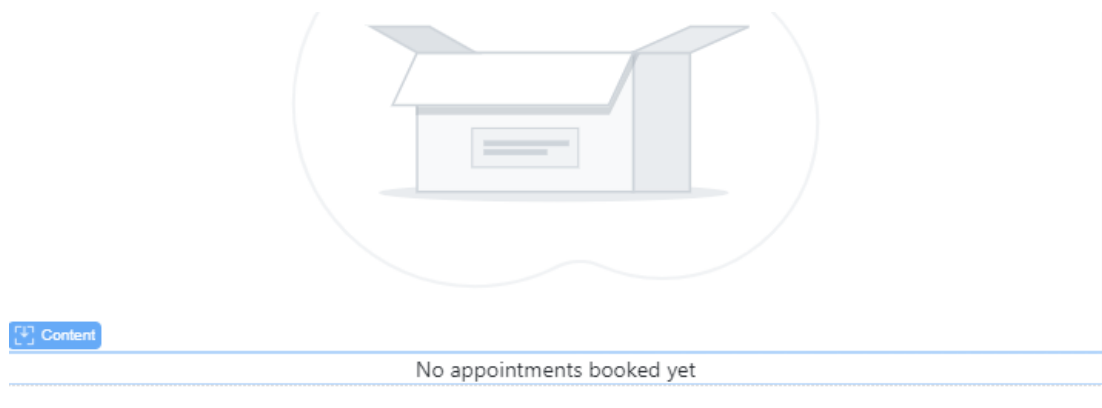


- 8) In the Image property, select the **Healthcare\_BlankSlate\_NothingToShow**.



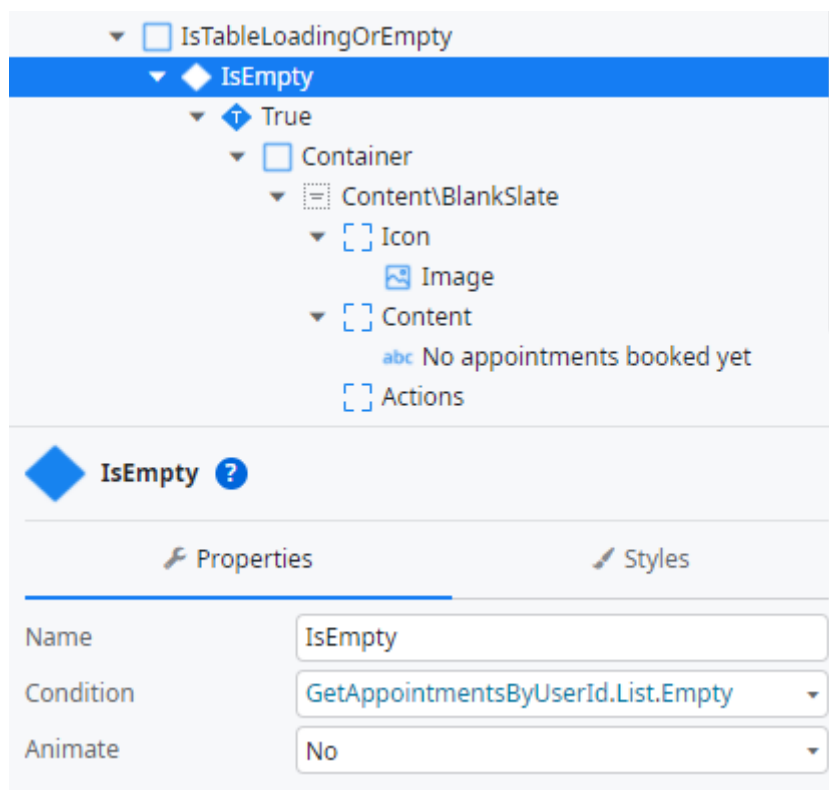


- 9) Click on the **Content** placeholder, right below the Icon placeholder, and type *No appointments booked yet*.

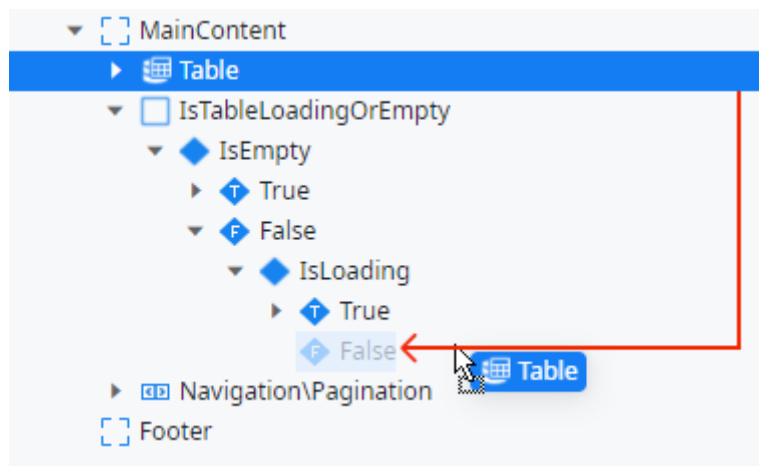


Now, you want to make sure that this Blank Slate only appears if there are no appointments. So, let's tweak the If condition.

- 10) Open the Widget Tree again, click on the **IsEmpty** If, and change the **Condition** to: `GetAppointmentsByUserId.List.Empty`



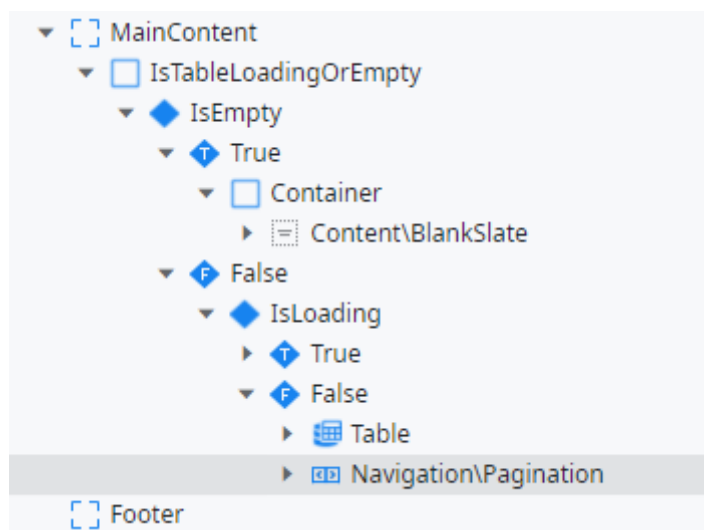
- 11) Using the Widget Tree, drag the **Table** and drop it inside the False branch of the **IsLoading** If.



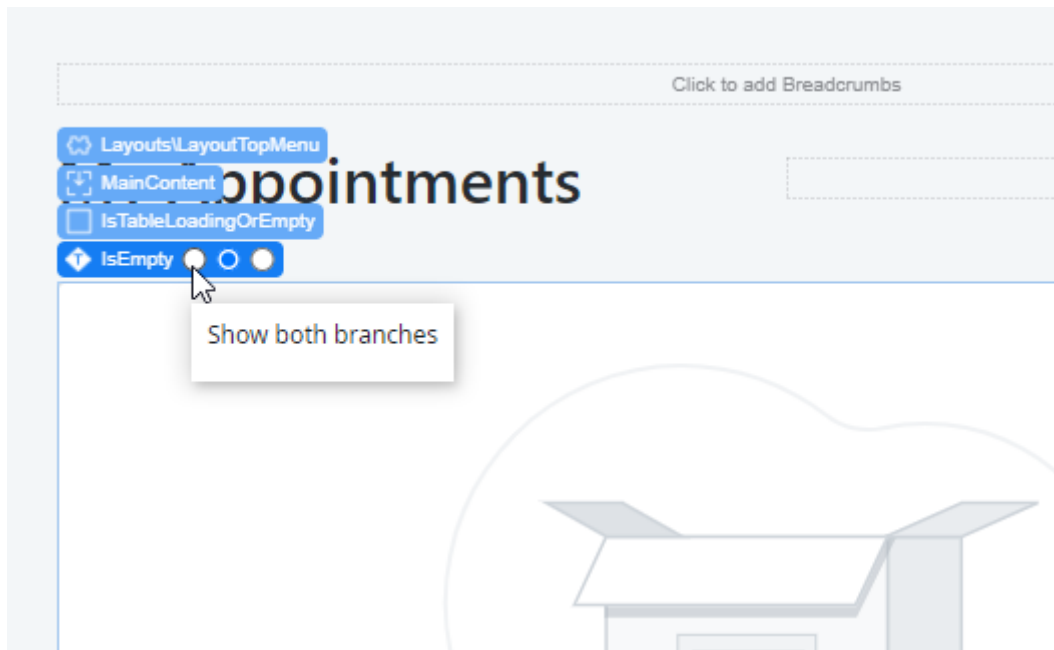
**Note:** You might have to expand the False branch of the IsEmpty If, then the IsLoading If first!

What are you doing here? If the list of appointments is not empty, and it's not loading yet, the Table appears.

- 12) Do the same for the **Navigation\Pagination**. Your Widget Tree should look like this:



- 13) Click on the IsEmpty If in the Screen preview and select the first bullet to show both branches.

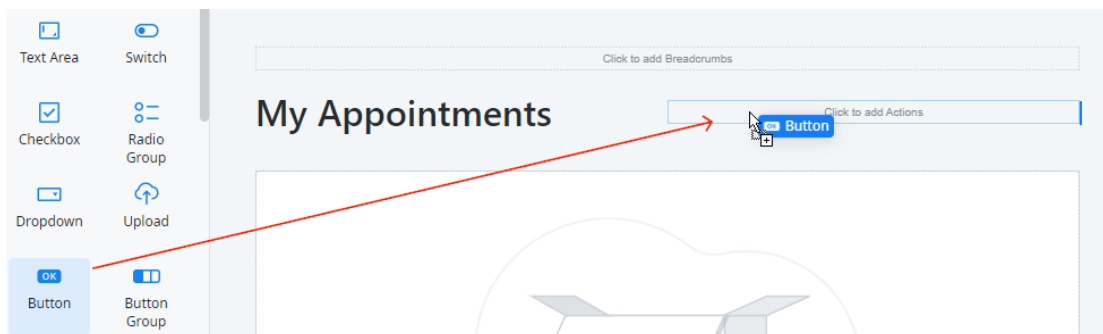


This affects how you see the Screen preview in Service Studio, but it has no impact on the Screen in the browser.

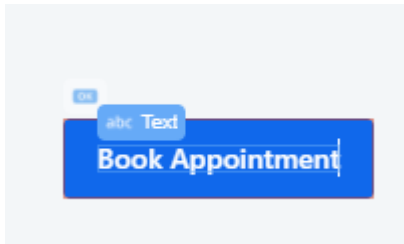
## Schedule a New Appointment

The list of appointments is ready. The next objective is to give the patient the option to schedule a new appointment. The logic for booking an appointment already exists, you just need to use it. So, add a Book Appointment Button and redirect the user to the Book Appointment Screen.

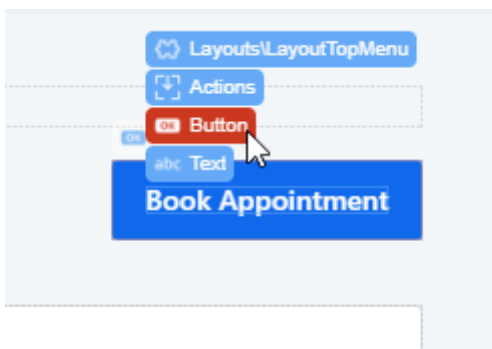
- 1) Still in the MyAppointments Screen, drag and drop a **Button** to the Actions placeholder in the Screen preview.



- 2) Type *Book Appointment* in the Button's text.

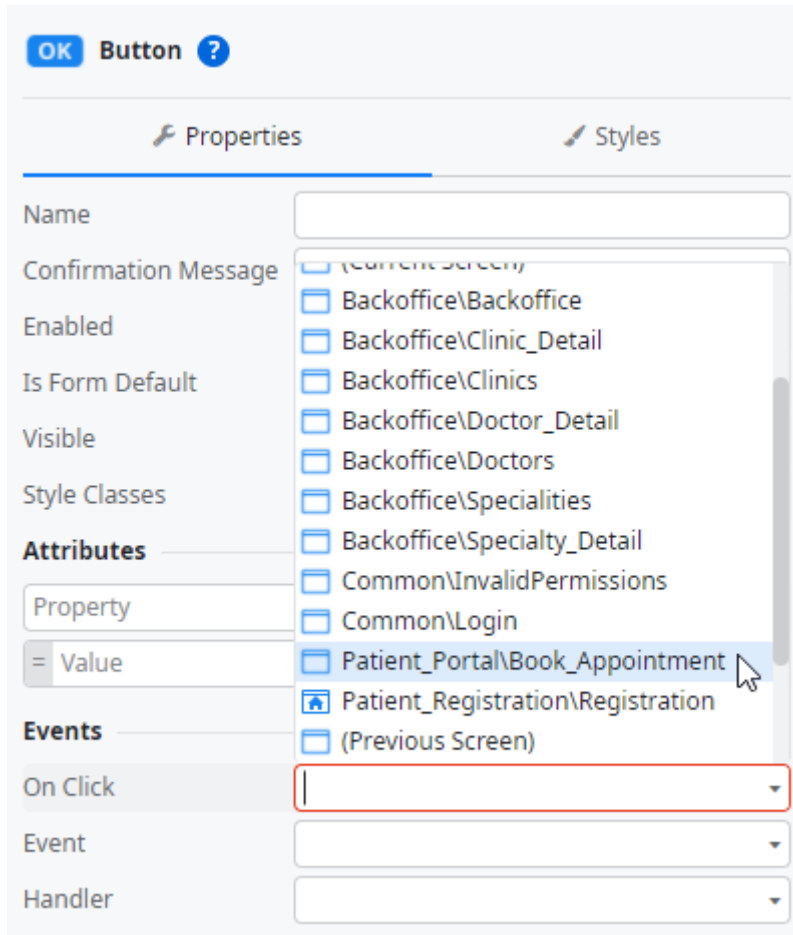


- 3) Click on the Button to select it and see its properties. If you click on top of the text, you will see the Text element's properties, so make sure you have the Button selected.



The Button has an error, since a mandatory property has not been defined yet. Let's fix that.

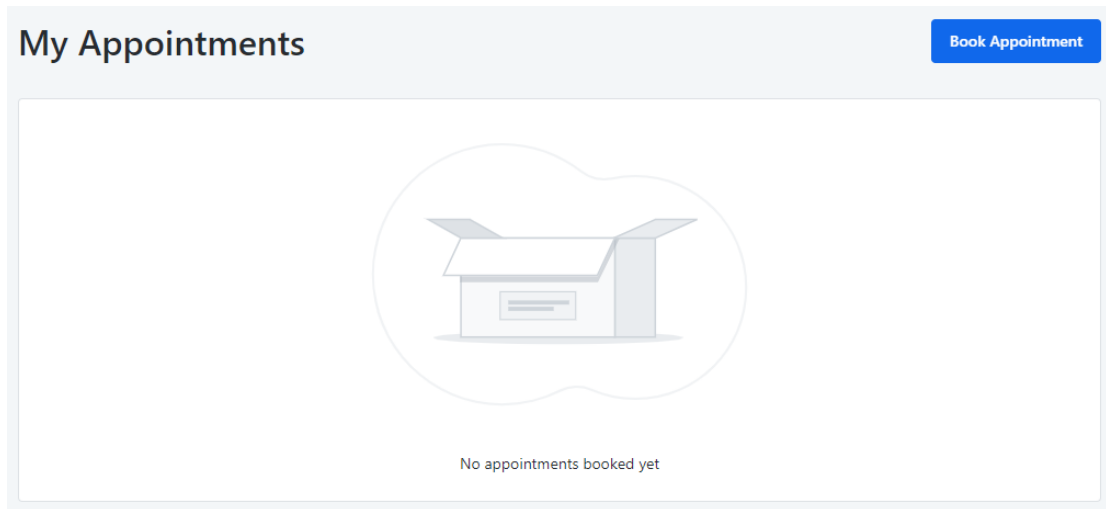
- 4) Click on the dropdown of the **On Click** property and select the **Book\_Appointment** Screen. So, when the user clicks on the Button, they will navigate to the Book\_Appointment Screen.



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



- 6) Login as a patient and see the new and improved MyAppointments Screen.



- 7) Click on the **Book Appointment** Button, then create a new appointment and confirm it. For testing purposes, the quickstart app has logic to create 4 users with the Doctor role in the database. On the next tutorials you will use these users to test the Doctor experience in the app. Why is this important here? It may be relevant to schedule an appointment for one of these doctors, to make sure you can also interact with these appointments as doctors in the future. The doctors created as users are:

- André Fonsenca - Cardiology
- Ann Devon - Dental
- Ana Trujillo - Cardiology


- Antonio Moreno - Endocrinology


## Appointment details

Jul 14 | 10:00 AM

AT Ana Trujillo

Cardiology



 **Good Hope Hospital**  
4669 Arbor Court, Eagon, Minnesota

[Call](#) [Get directions](#)

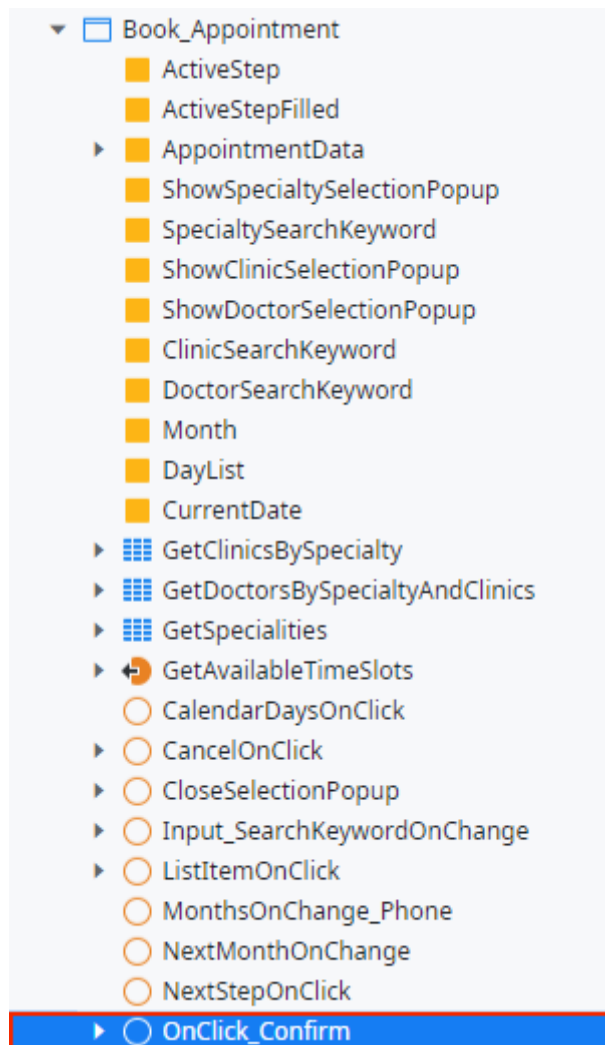
[Cancel](#) [✓ Confirm](#)

You will notice that you get a success message after confirming, but you navigate to the Book Appointment Screen again. It would be better to navigate to the MyAppointments Screen.

## Redirecting to the MyAppointments Screen

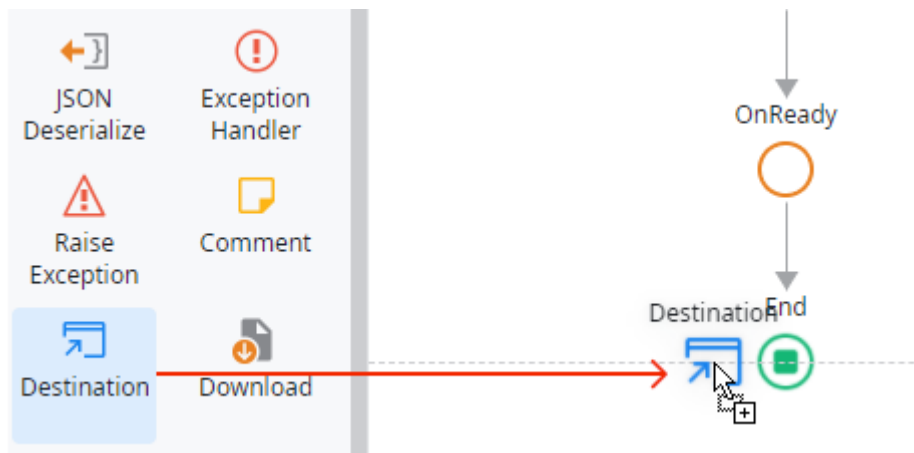
Now you are going to change the Confirm Appointment Action to make sure it redirects the user back to the MyAppointments Screen.

- 1) Expand the Book\_Appointment Screen in the Interface tab, then double-click on the **OnClick\_Confirm** Client Action.

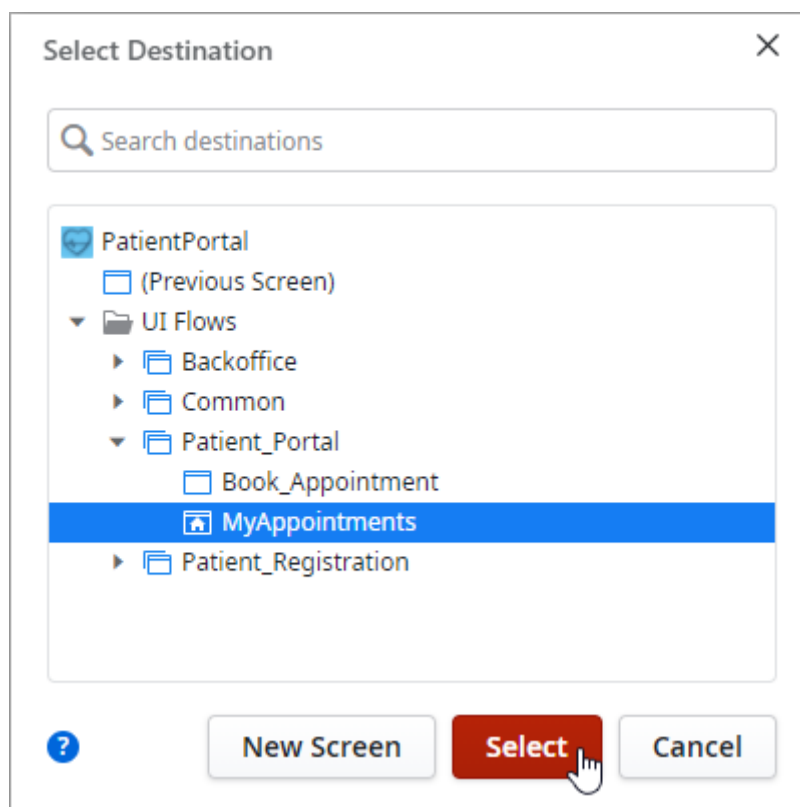




- 2) Scroll down to the end of the Action Flow, then drag a **Destination** node and drop it on the last End node, to replace it.



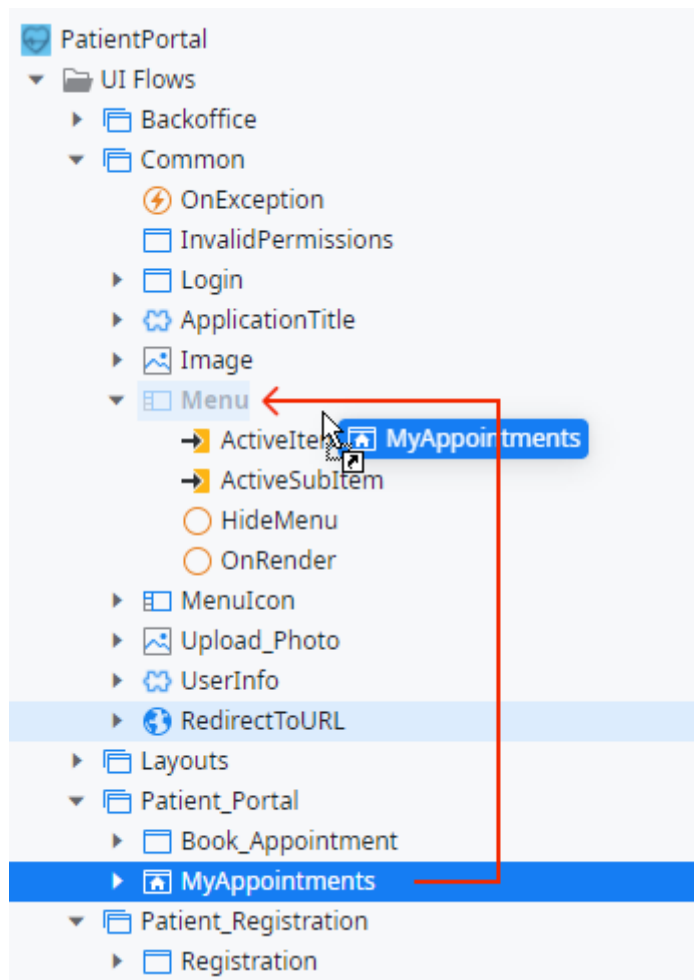
- 3) Select the **MyAppointments** Screen in the dialog.



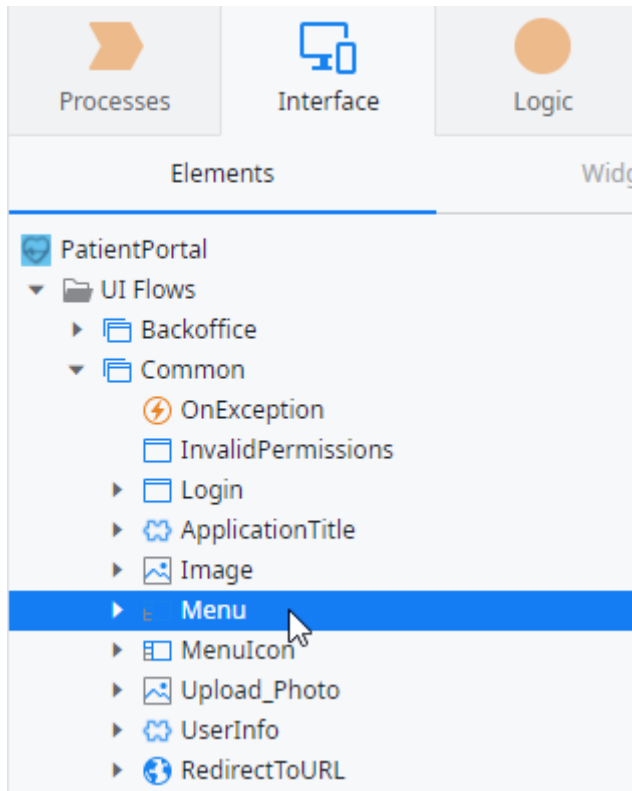
## Menu

The finish, let's just add the MyAppointments Screen to the Menu of the app, so a patient (and only a patient!) can access it whenever they want.

- 1) Still in the Interface tab, drag the MyAppointments Screen and drop it on the **Menu** Block, inside the Common UI Flow.

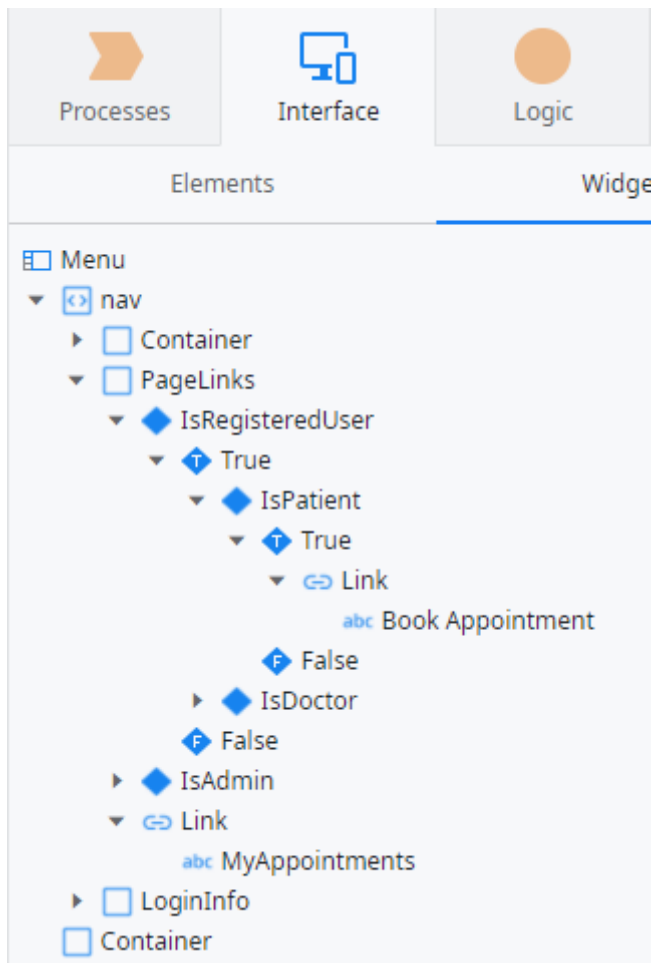


- 2) Double-click on the **Menu** Block inside the Common UI Flow.



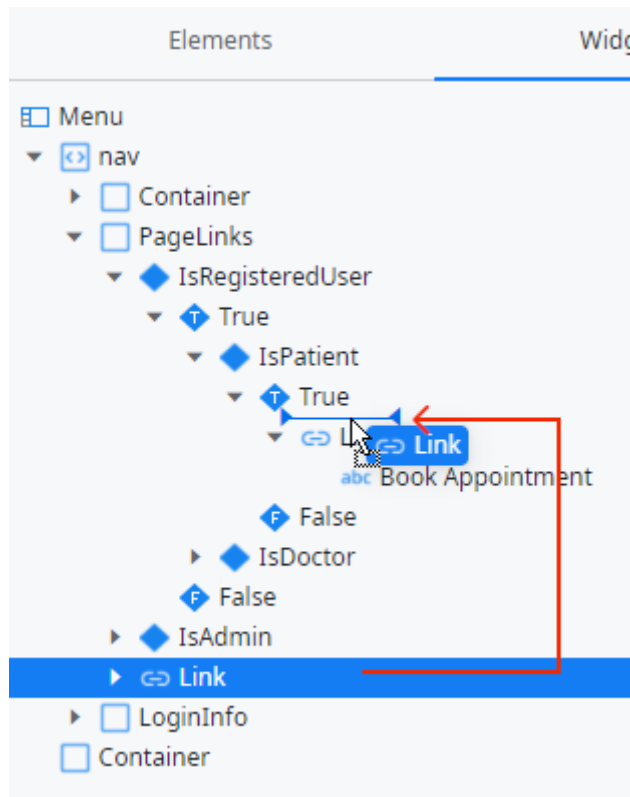
- 3) Open the Widget Tree and expand the **nav** and **Page Links** Containers, as well as the **IsRegisteredUser** and the **IsPatient** If. Now, inside this last If, expand the True branch and Link. This is the Menu entry for the Book Appointment Screen. This sequence of Ifs helps us organize who should have access to

what in the Menu. In this case, you are inside the True branch of the IsPatient If, meaning that only Patients can see these menu entries.



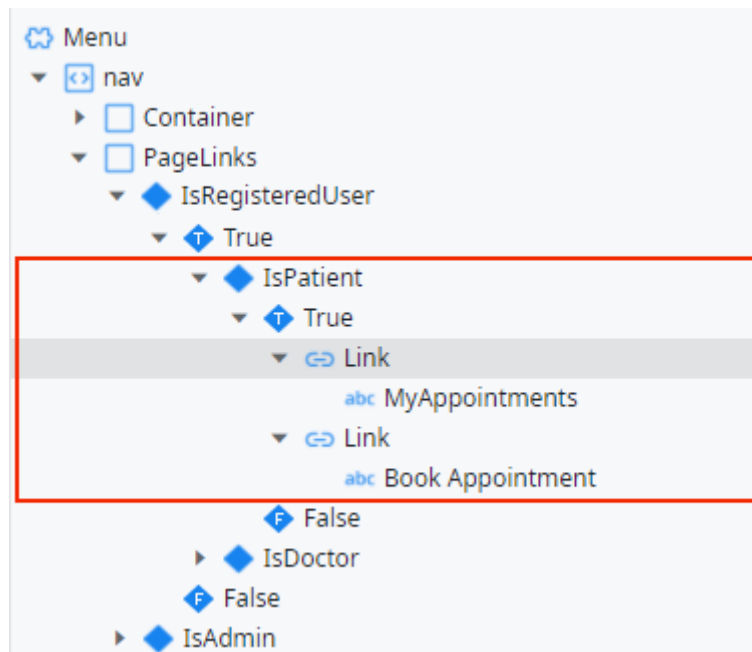
Right now, any user can see the link to the MyAppointments Screen, and we don't want that!

- 4) Drag the Link for the **MyAppointments** Screen and drop it inside the **True** Branch of the IsPatient If.



**Note:** You can move the link before or after the Book Appointment Link. The order won't affect the final functionality.

Now, only Patients can see the link to the My Appointments Screen.



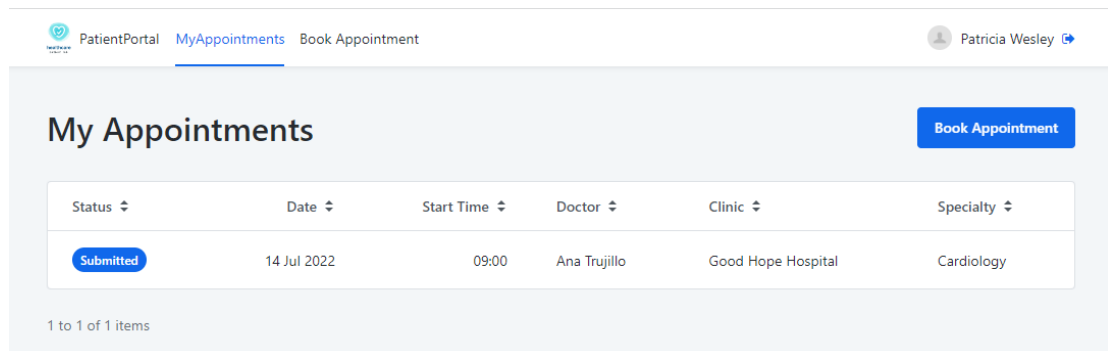
- 5) Publish the module.



## Testing

Let's open the app so you can test the latest modifications.

- 1) Login as a Patient to see the **MyAppointments** Screen.



At this point, you should already have an appointment created. Add more appointments, so you can see the entire flow. For now, all the Appointments will have the Submitted Status. In the next lessons you will learn how to change that.

- 2) Make sure the Menu option to access the MyAppointments Screen is available and that it works!

## Wrapping up

Congratulations on finishing this tutorial and this lesson. This one was a bit shorter. We will come back in later tutorials to the patient's experience, but in the next ones you will focus on the Doctor.

## 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) [Switch](#)
- 2) [OutSystems UI Blank Slate](#)
- 3) [OutSystems UI Flows](#)
- 4) [User Roles](#)

**See you in the next tutorial!**