

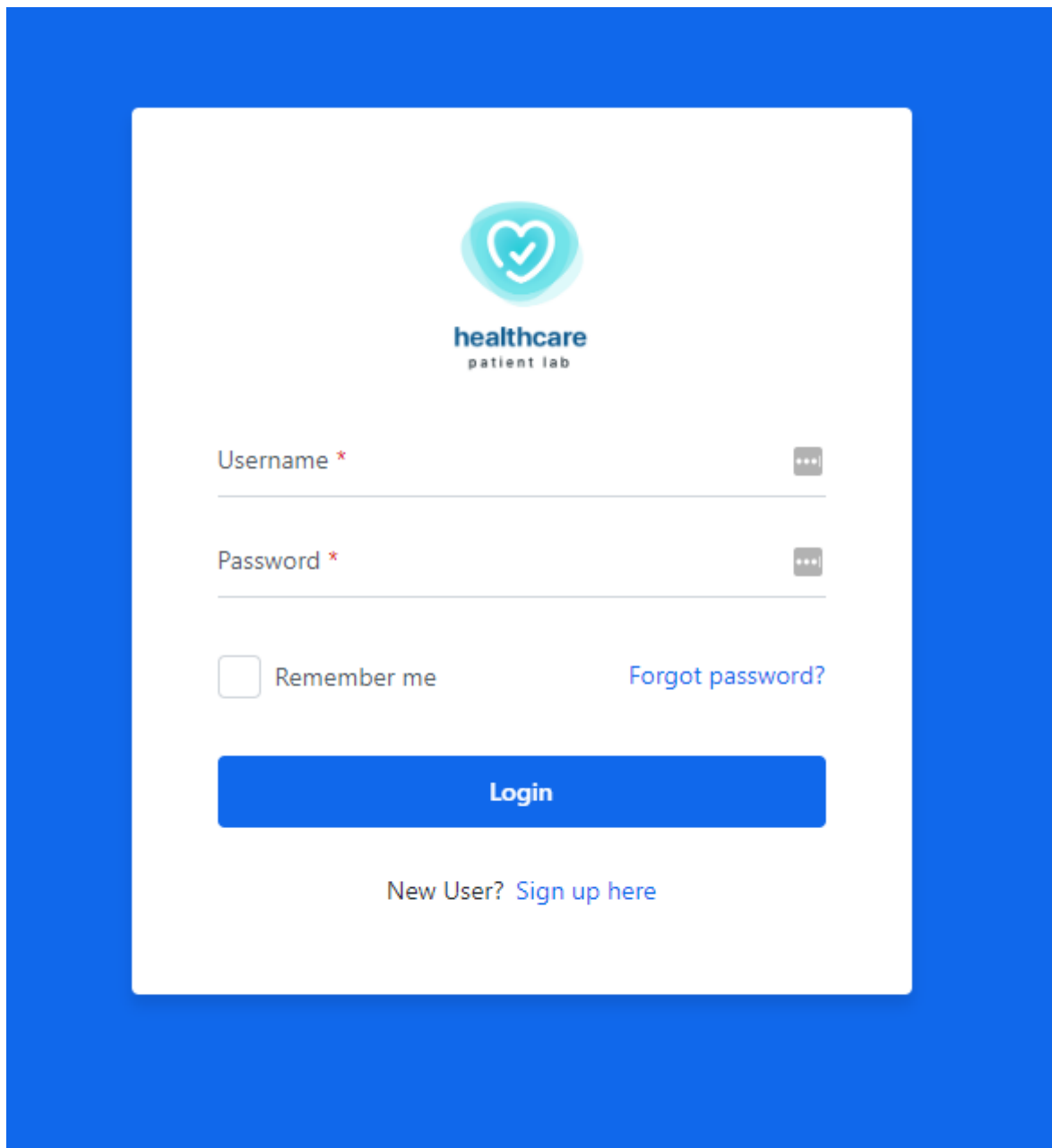
Patient Registration Screen

Table of Contents

Scenario.....	2
How-To.....	4
Setting up the Registration Screen	4
Registration Form	6
Upload Photo	7
Patient Input Fields	13
User Input Fields	16
Mobile Preview	22
Patient Onboarding Fields	25
Menu	28
Wrapping up.....	30
References	30

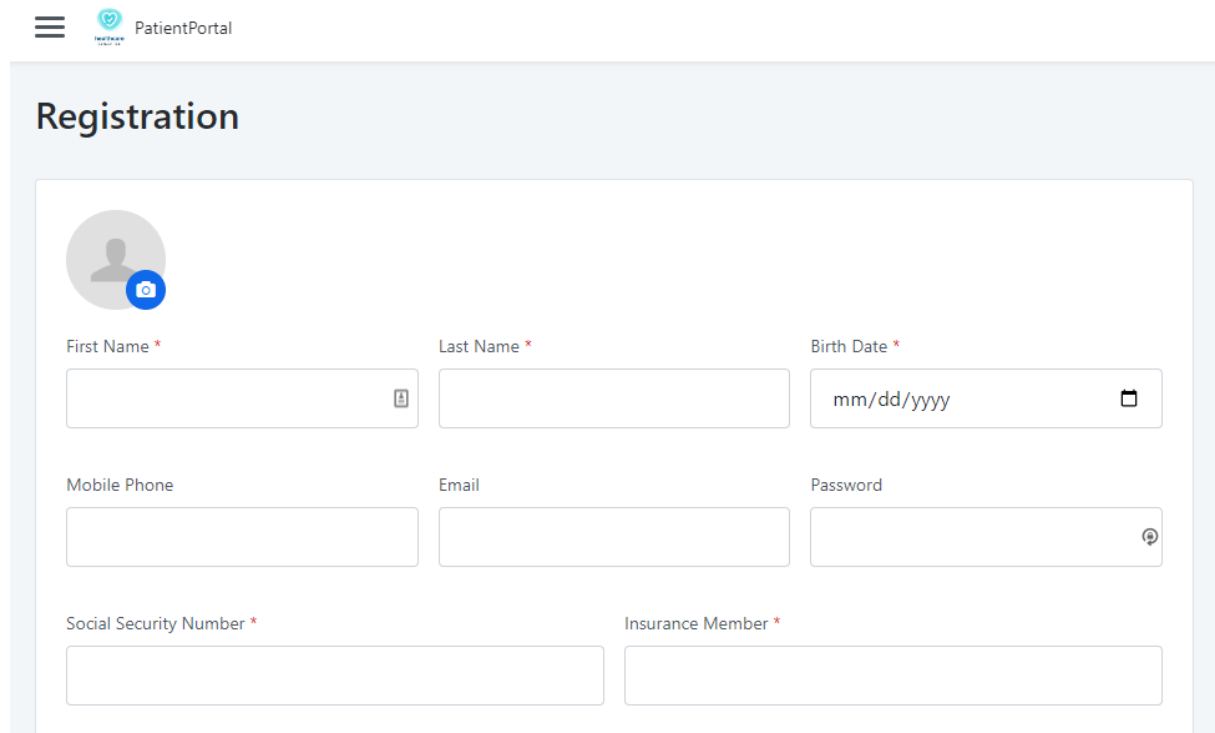
Scenario

At this point of the course, your application should have the registration flow built, with the Login Screen redirecting a user without login to the Registration Screen.

The image shows a login screen for an application named 'healthcare patient lab'. The screen has a white background with a blue border. At the top center is a logo consisting of a teal heart with a white checkmark inside, and the text 'healthcare' in bold and 'patient lab' in a smaller font below it. Below the logo are two input fields: 'Username *' and 'Password *', each with a small grey icon to its right. Below the password field is a checkbox labeled 'Remember me' and a link 'Forgot password?'. A large blue button labeled 'Login' is centered below these elements. At the bottom, the text 'New User?' is followed by a link 'Sign up here'.

In this tutorial, you will build the UI for the Registration Screen. The Screen will have a Form that will allow users to insert the patient information, which includes their name


and birth date, but also email, password and social security number, as well as a picture. At the end of this tutorial, the Registration Screen should look like this:



The image shows a web application header with a hamburger menu icon, a heart icon, and the text "PatientPortal". Below the header is a "Registration" section. It features a profile picture placeholder with a camera icon. The form contains several input fields: "First Name *" and "Last Name *" are single-line text boxes; "Birth Date *" is a date picker showing "mm/dd/yyyy"; "Mobile Phone", "Email", and "Password" are single-line text boxes, with the password field having an eye icon; "Social Security Number *" and "Insurance Member *" are single-line text boxes.

PatientPortal

Registration



First Name * Last Name * Birth Date *

Mobile Phone Email Password

Social Security Number * Insurance Member *

The Registration Screen will not be finished at the end of this tutorial and it will continue on the next ones.

How-To

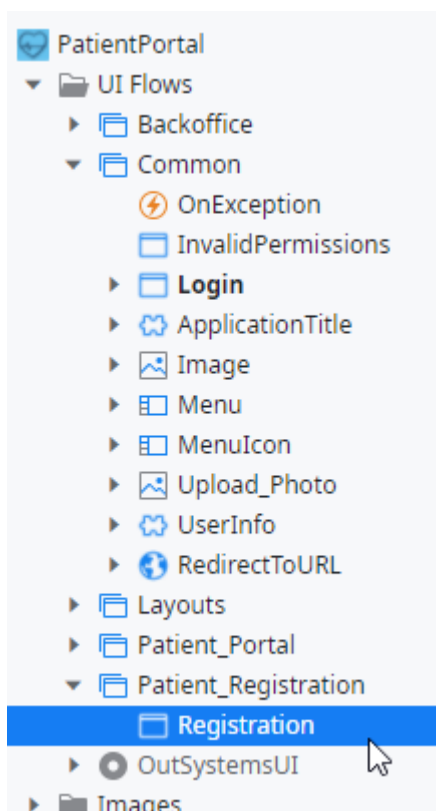
Let's build the Registration Form step-by-step!

Setting up the Registration Screen

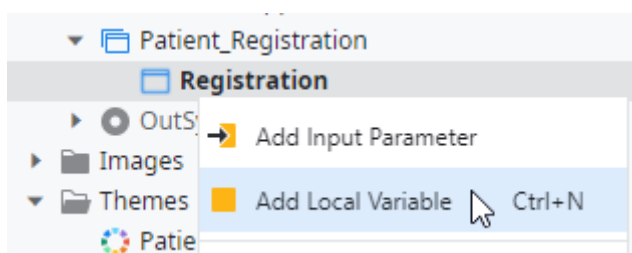
To register a new user you will need to collect a lot of information that will be saved in four different Entities in the database: Patient, PatientOnboarding, PatientImage and User.

But before that, the information inserted by the user in the Form need to be saved temporarily, right when the user types and selects the data on the Screen. You will use Local Variables to store the information temporarily before writing it in the Database.

- 1) In Service Studio, open the **Registration** Screen (if it is not open already).



- 2) Right-click on the Registration Screen and select **Add Local Variable**.



- 3) Set its **Name** to *Patient* and make sure the **Data Type** is set to **Patient**.

The screenshot shows a tree view on the left with 'Patient_Registration' expanded, then 'Registration', and finally 'Patient' selected. The 'Patient' entity structure is listed below it: Id, FirstName, LastName, BirthDate, and UserId. On the right, the 'Patient' Local Variable configuration is shown. The 'Name' field is set to 'Patient', the 'Data Type' is set to 'Patient', and the 'Default Value' is empty.

This will determine that the Local Variable will have a data type with the same structure of the Patient Entity.

- 4) Repeat the previous two steps and create the Local Variables *User*, *PatientImage*, and *PatientOnboarding* with the corresponding Data Types.

The screenshot shows the 'PatientImage' Local Variable configuration. The 'Name' field is set to 'PatientImage', the 'Data Type' is set to 'PatientImage', and the 'Default Value' is empty.

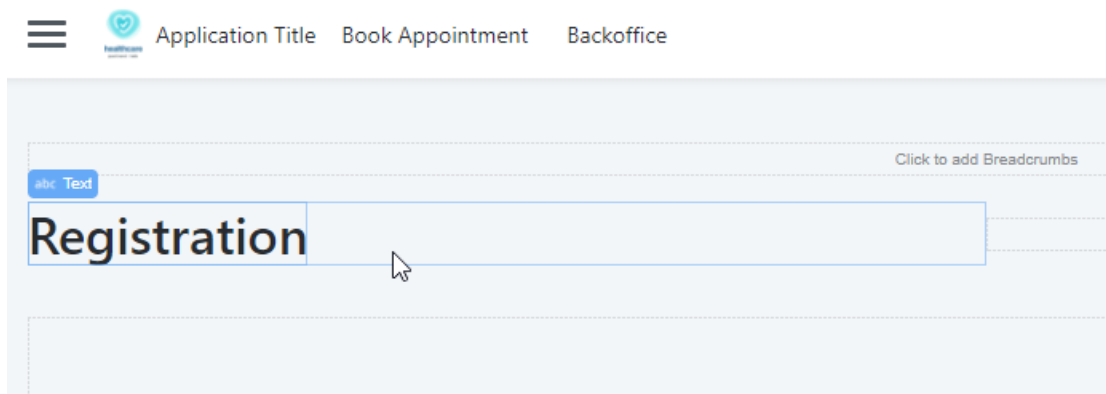
The screenshot shows the 'PatientOnboarding' Local Variable configuration. The 'Name' field is set to 'PatientOnboarding', the 'Data Type' is set to 'PatientOnboarding', and the 'Default Value' is empty.

The screenshot shows the 'User' Local Variable configuration. The 'Name' field is set to 'User', the 'Data Type' is set to 'User', and the 'Default Value' is empty.

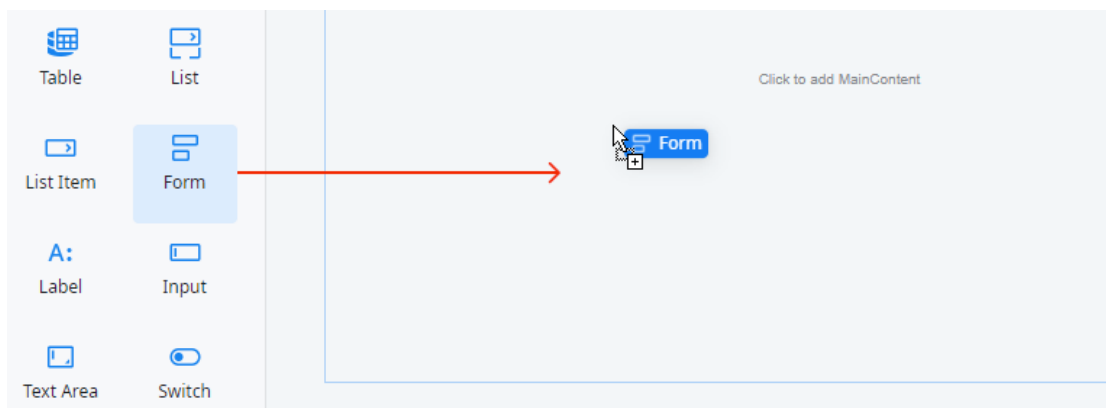
Registration Form

Now, you're ready to build the Form!

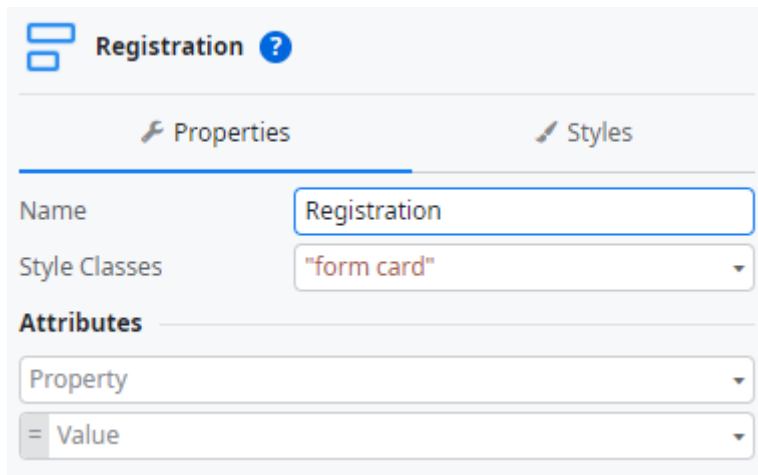
- 1) Click on the **Title** section of the Screen and type *Registration*.



- 2) Drag a **Form** from the left sidebar and drop it on the main section on the Screen.



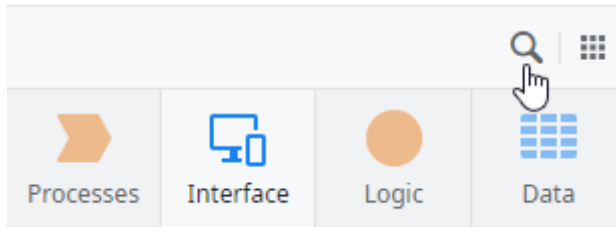
- 3) Set the **Name** of the Form to *Registration* in the properties area on the right sidebar.



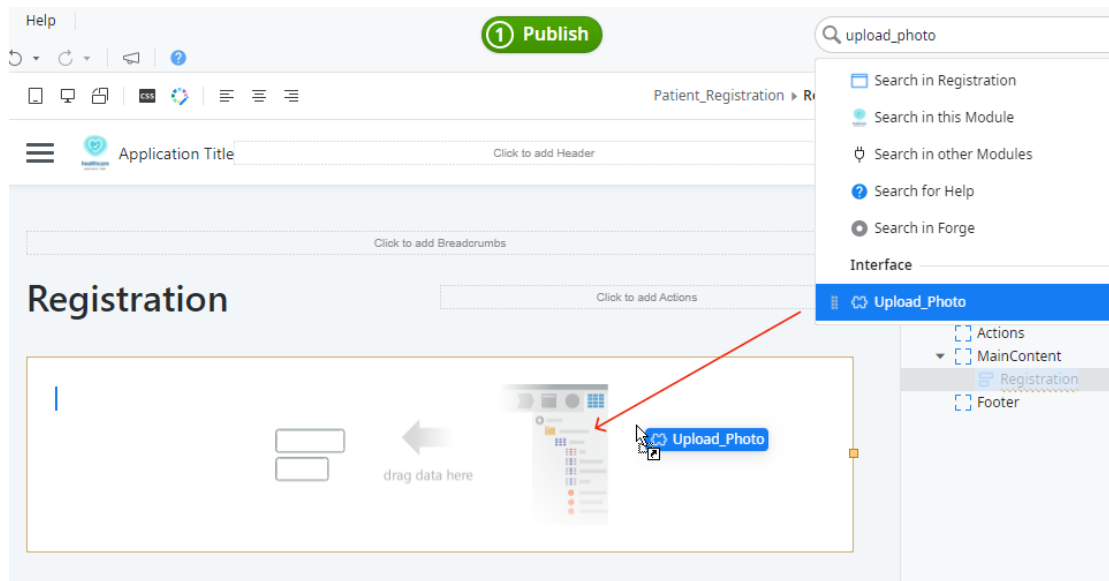
Upload Photo

A patient should be able to upload a picture in their registration. You already have a Block element available that will help you with that.

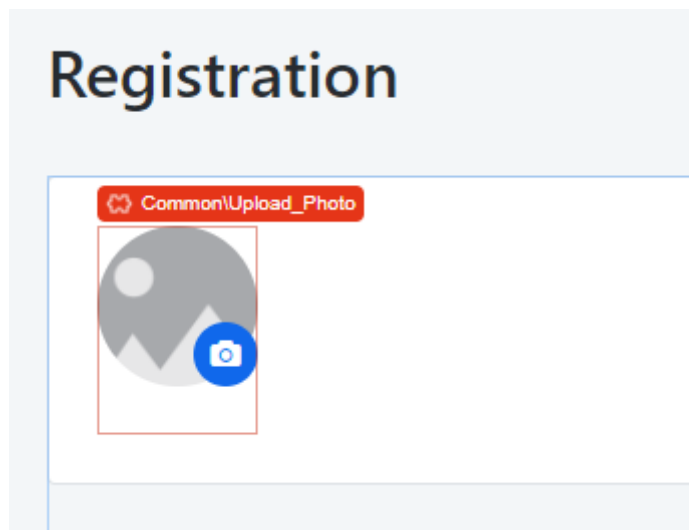
- 1) Click on the Search Icon right above the layer tabs.



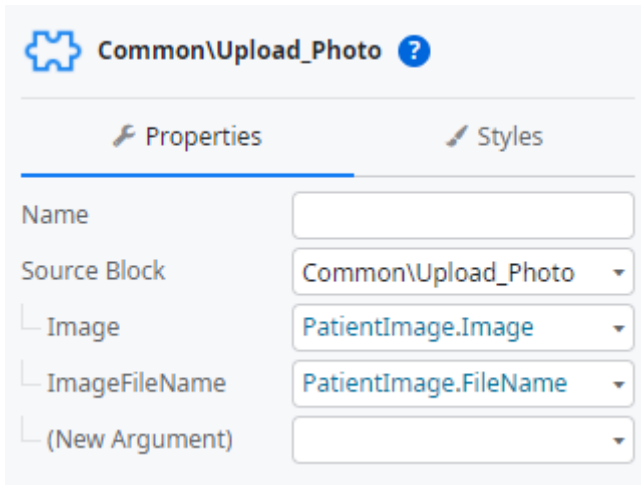
- 2) Type **Upload_Photo** in the search bar, then drag and drop the block inside the Form.



- 3) Click on the **Upload_Photo** Block to see its properties.



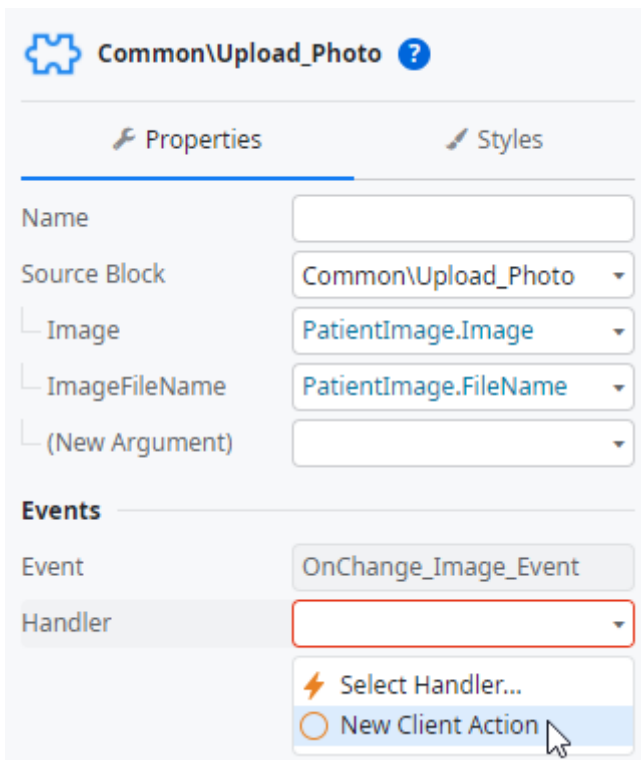
- 4) Set the **Image** parameter to **PatientImage.Image** and the **ImageFileName** parameter to **PatientImage.FileName**.



The screenshot shows the 'Common\Upload_Photo' block in the OutSystems Properties Panel. The 'Properties' tab is active. The 'Image' property is set to 'PatientImage.Image' and the 'ImageFileName' property is set to 'PatientImage.FileName'. The 'Source Block' is set to 'Common\Upload_Photo'. There is also a '(New Argument)' property set to an empty dropdown.

This Block has two objectives. The first one is to allow uploading the photo. The second one is to display the picture when it exists. Here, you are passing the image and file name information to the Block, so it can be displayed properly on the Screen.

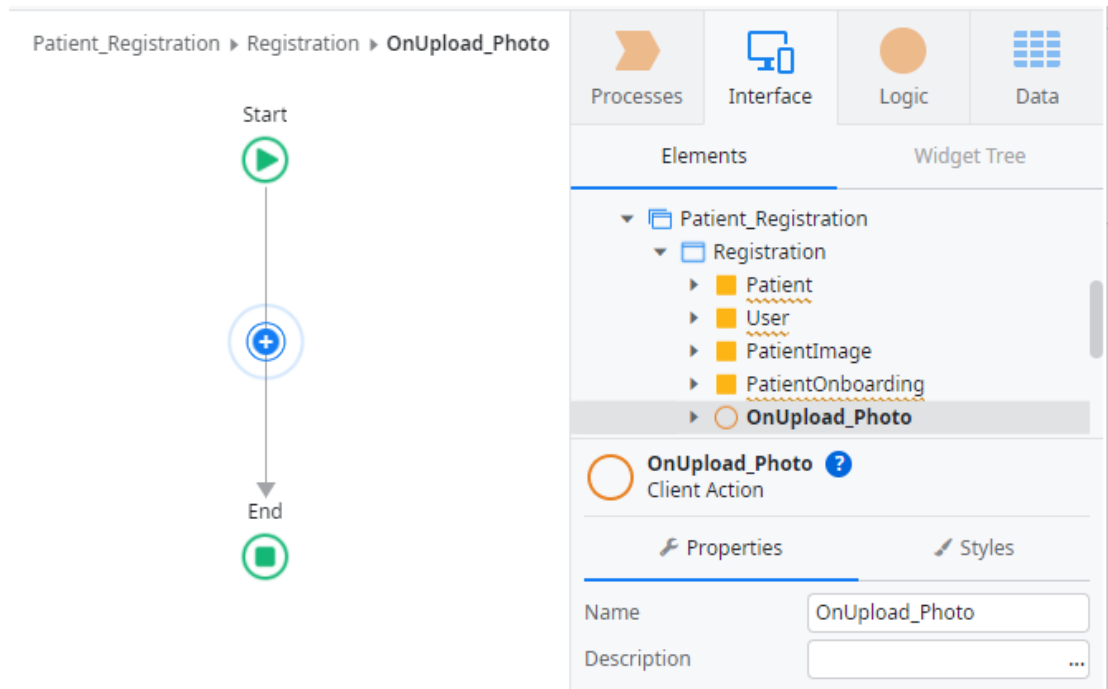
- 5) Then, open the dropdown of the **Handler** property and select **New Client Action**.



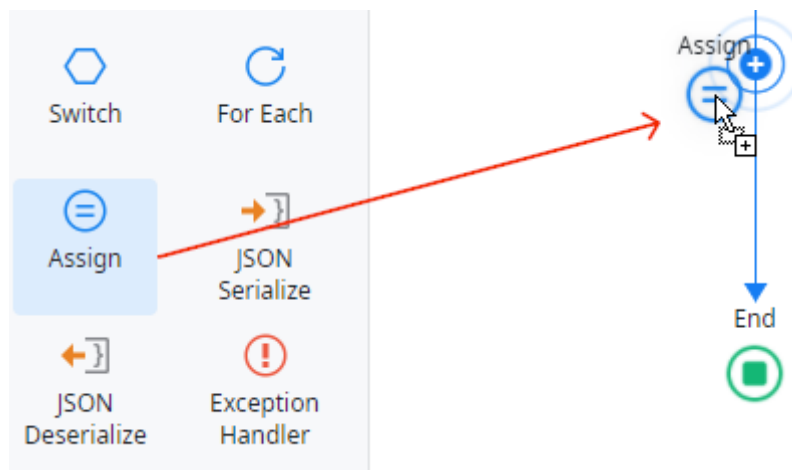
The screenshot shows the 'Common\Upload_Photo' block in the OutSystems Properties Panel. The 'Properties' tab is active. The 'Image' property is set to 'PatientImage.Image' and the 'ImageFileName' property is set to 'PatientImage.FileName'. The 'Source Block' is set to 'Common\Upload_Photo'. There is also a '(New Argument)' property set to an empty dropdown. The 'Events' section is expanded, showing the 'OnChange_Image_Event' event. The 'Handler' property is open, and 'New Client Action' is selected.

A new Action is created and you will be directed to its flow. The Action flow is where you define the logic that will be executed when the Action runs. In this case, the Action will run when the user uploads a picture.

- 6) Rename the Action as *OnUpload_Photo*. Don't forget! A good naming convention is your best friend!

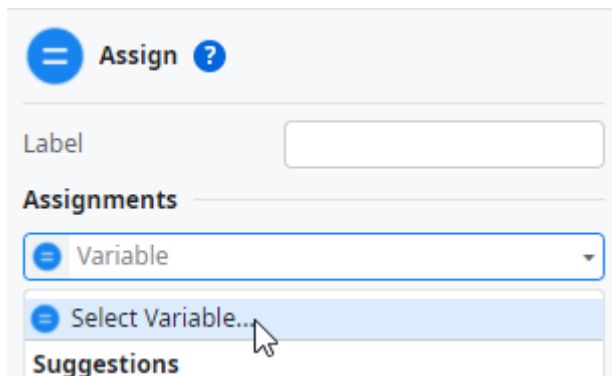


- 7) Select an **Assign** element in the left sidebar, then drag and drop it in the blue region on the center of the flow.

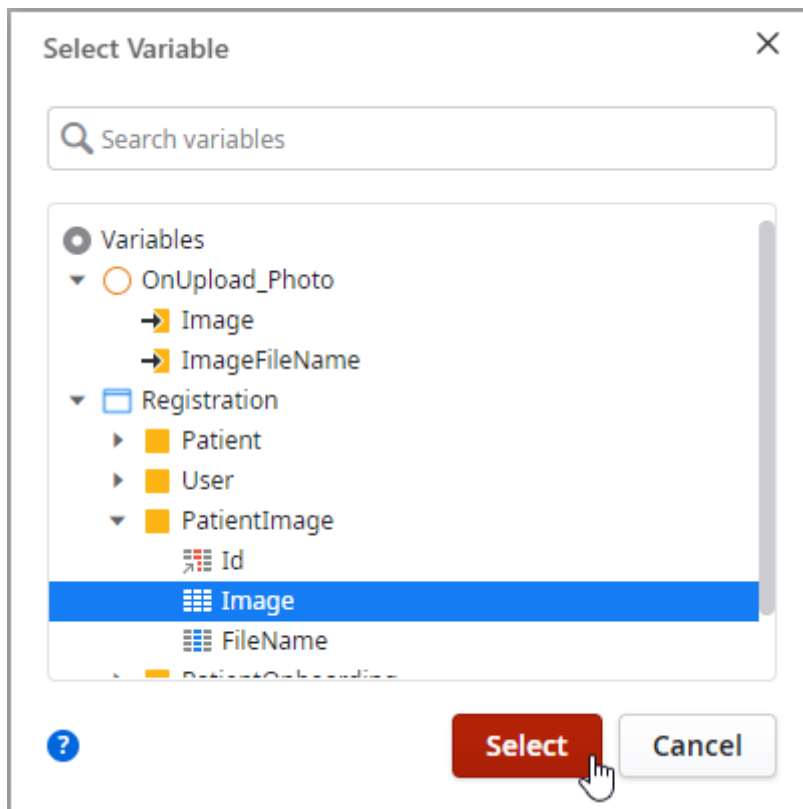


The Assign element is used to assign values to variables. In this case, we want to assign the image and its name uploaded by the user to the PatientImage Local Variable, to save the information temporarily on the Screen.

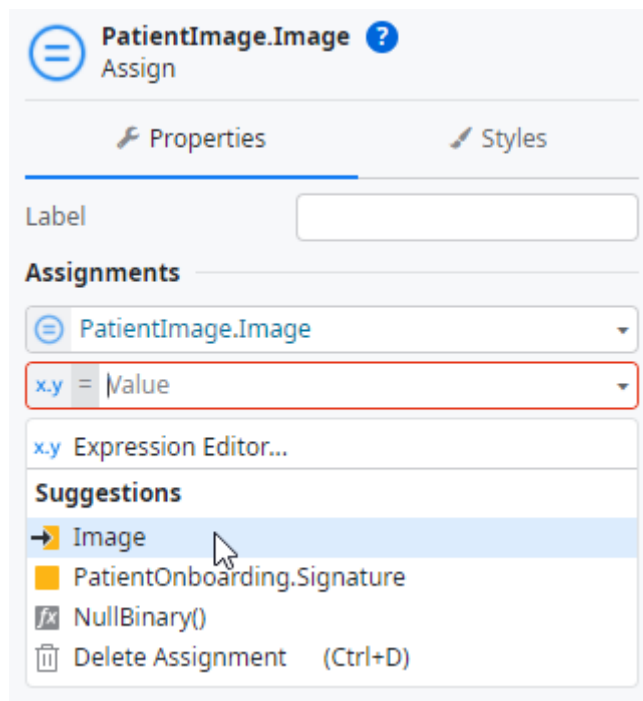
- 8) Click on the Variable property and select **Select Variable...**



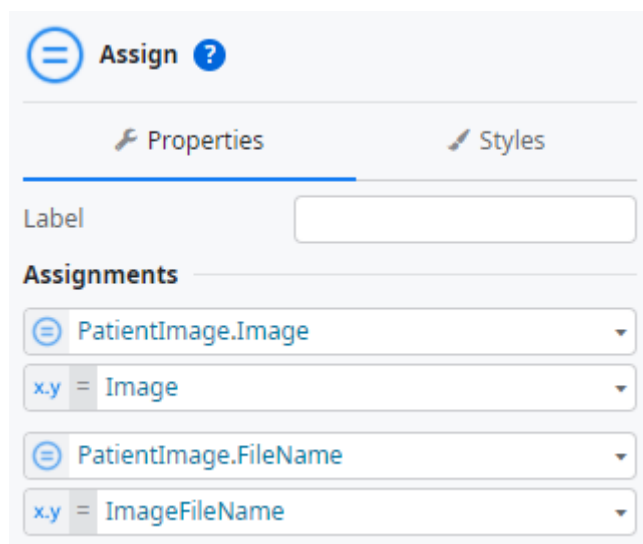
- 9) Expand the **PatientImage** variable and select the **Image** attribute.



- 10) Expand the Value dropdown and select the **Image** Input parameter in the suggestions.



- 11) In the same Assign, select the **PatientImage.FileName** in the Variable dropdown and the **ImageFileName** Input Parameter in the Value property.

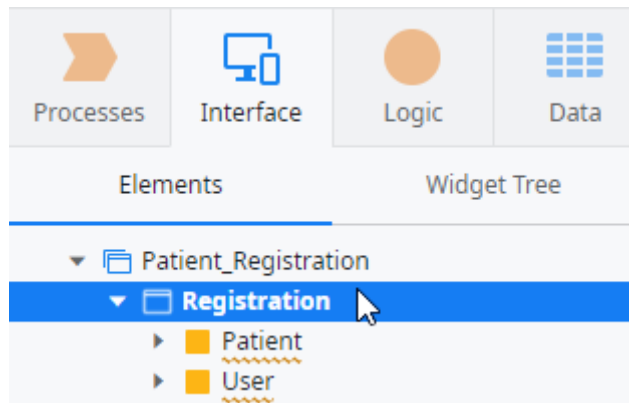


Note: The attribute should appear in the suggestions. In case that doesn't happen, you can select it exactly like you did in the previous steps or just type *PatientImage.FileName*.

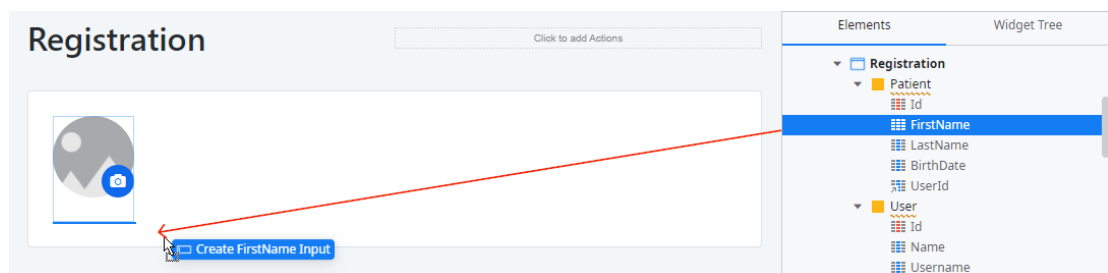
Patient Input Fields

Now that the functionality to upload the photo is ready, let's add more fields to the Form like the first name, last name and birth date.

- 1) Go back to the **Registration** Screen by double clicking on it.

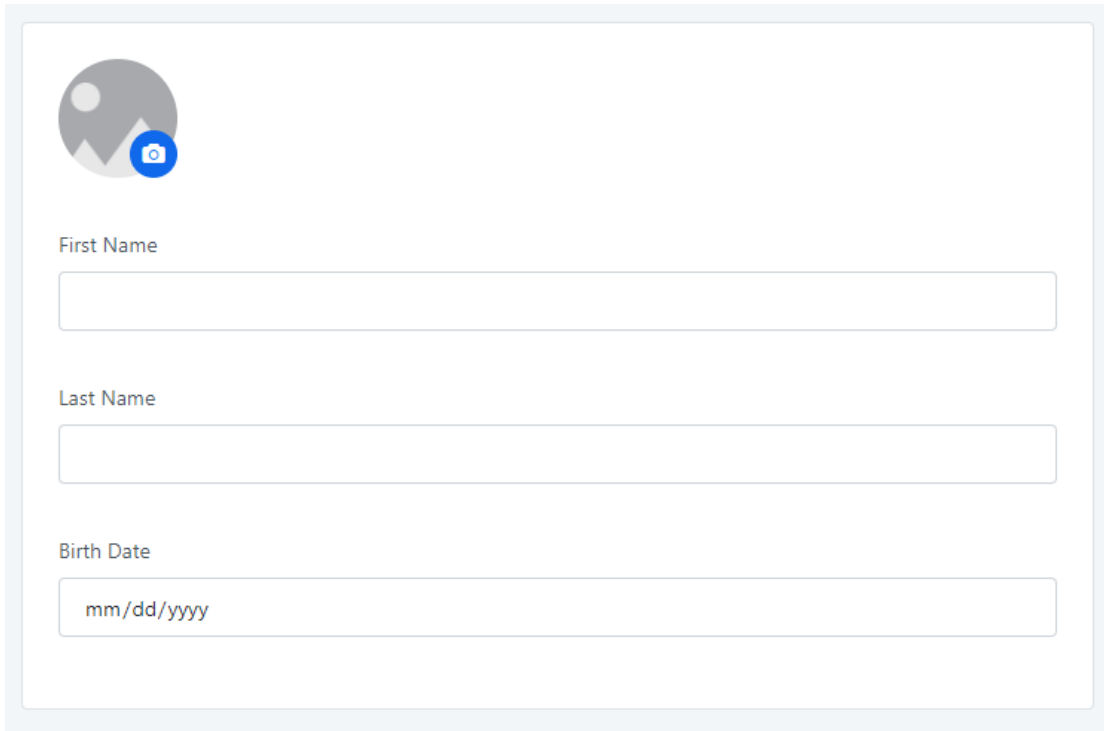


- 2) Expand the **Patient** Local Variable, select the **FirstName** attribute, then drag and drop it inside the Form.



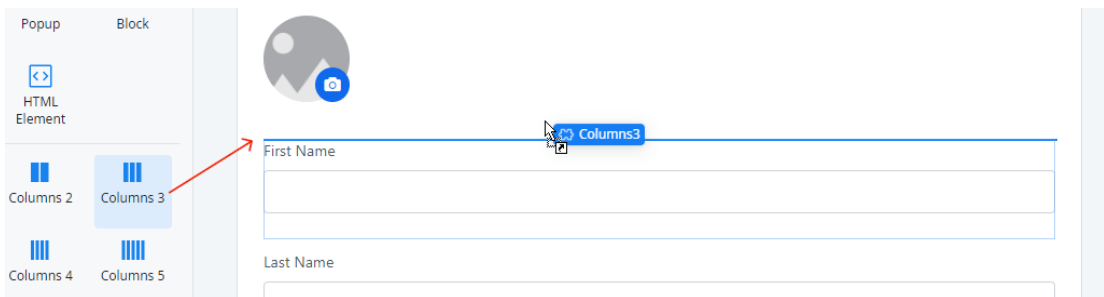
The platform will automatically create a Container with the Label and the Input field of the First Name.

- 3) Repeat the same process for the **LastName** and **BirthDate** attributes.

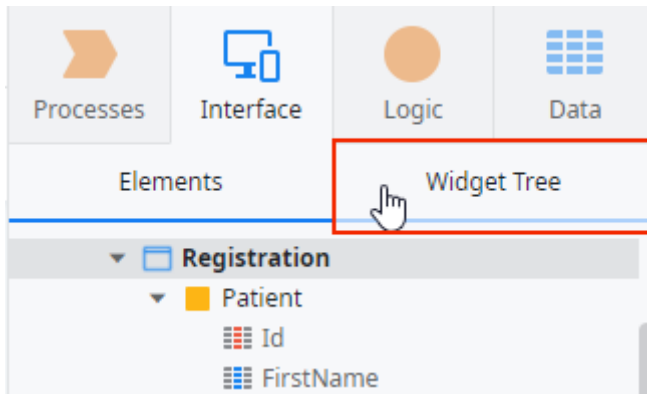


This is fine, but we can make better usage of the space available. So let's use some Columns to reorganize the fields!

- 4) Drag a **Columns 3** element from the left sidebar and drop it inside the Form, between the Upload Photo and the First Name field.

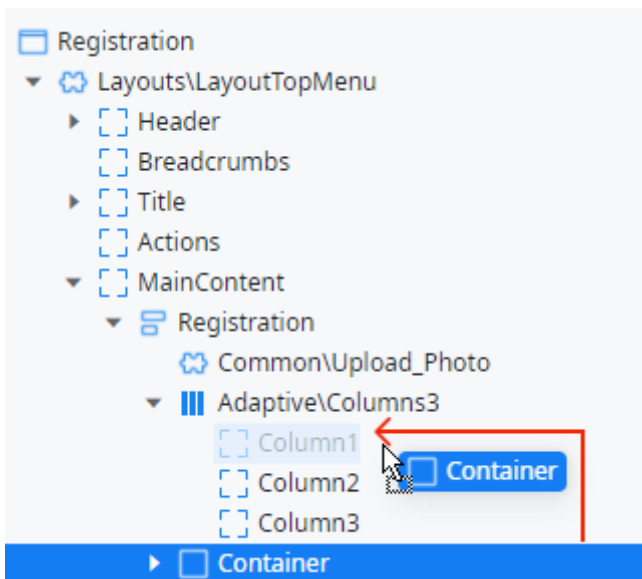


- 5) Click on the **Widget Tree** tab on the right sidebar.



The Widget Tree is available when we have a Screen open on Service Studio. It shows the structure of all the elements inside the Screen, and it is a great helper to set the pieces right where they belong.

- 6) Drag the first **Container** and drop it inside the Column1.

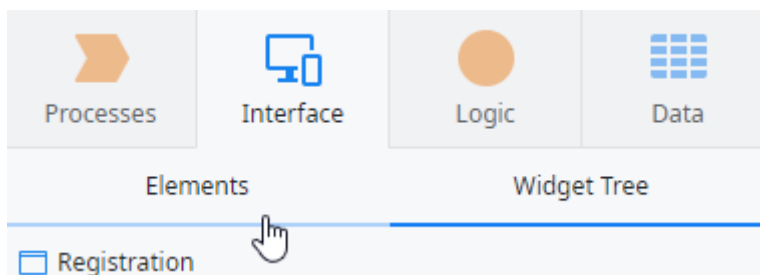


- 7) Do the same with the other Containers for Column2 and Column3 respectively. You can see the difference on the Screen already.

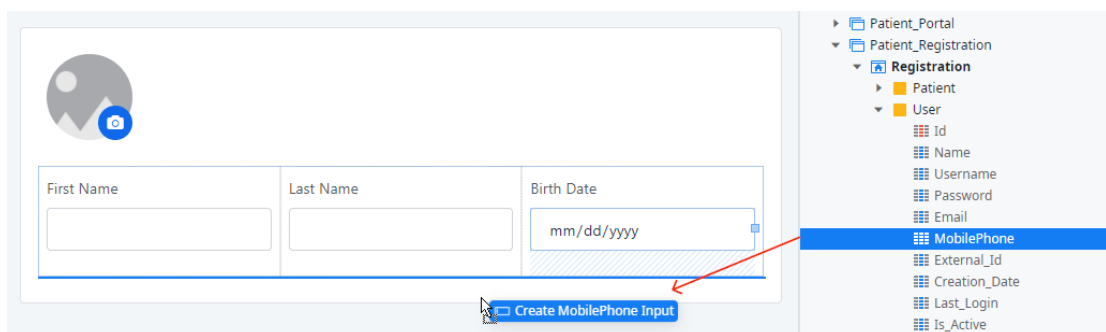
User Input Fields

Let's continue populating the Form with some relevant fields, such as the mobile phone, email and password. These attributes are not in the Patient Entity, since they are part of the already existing User Entity.

- 1) Switch from the Widget Tree to the **Elements** tab.



- 2) Expand the **User** Local Variable on the right sidebar, select the **MobilePhone** attribute, and drag and drop it inside the Form, after the first three fields.



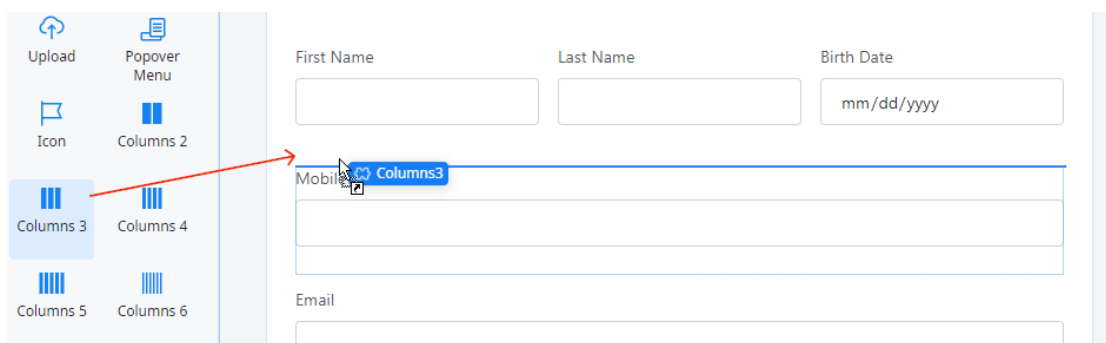
- 3) Do the same for the **Email** attribute under the User Local Variable.

First Name	Last Name	Birth Date
<input type="text"/>	<input type="text"/>	<input type="text" value="mm/dd/yyyy"/>

Mobile Phone

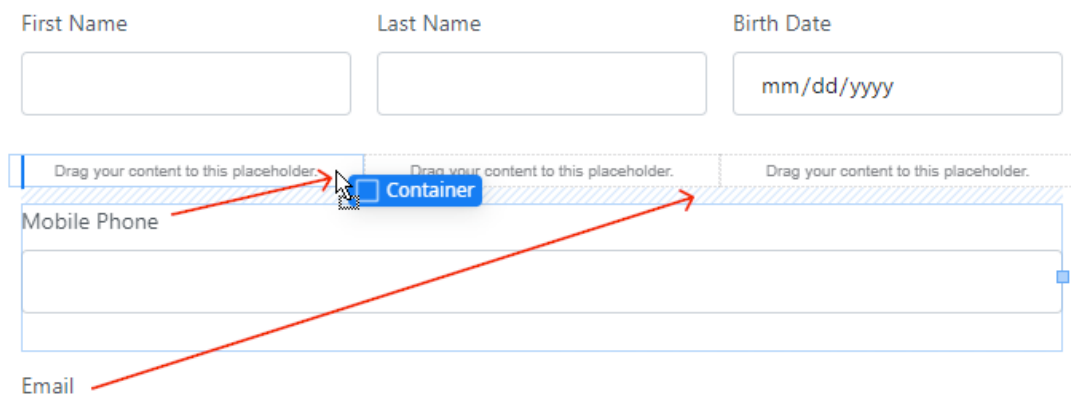
Email

- 1) Drag and drop another **Columns 3** under the one with the First Name, Last Name, and Birth Date.



The screenshot shows a widget palette on the left with various components like 'Upload', 'Popover Menu', 'Icon', and several 'Columns' widgets. A red arrow points from 'Columns 3' in the palette to the 'Mobile' field in the form. The form itself has three columns at the top: 'First Name', 'Last Name', and 'Birth Date'. Below them is a 'Mobile' field and an 'Email' field.

- 2) Drag and drop the Containers with the Mobile Phone and Email fields to the first and second columns respectively.

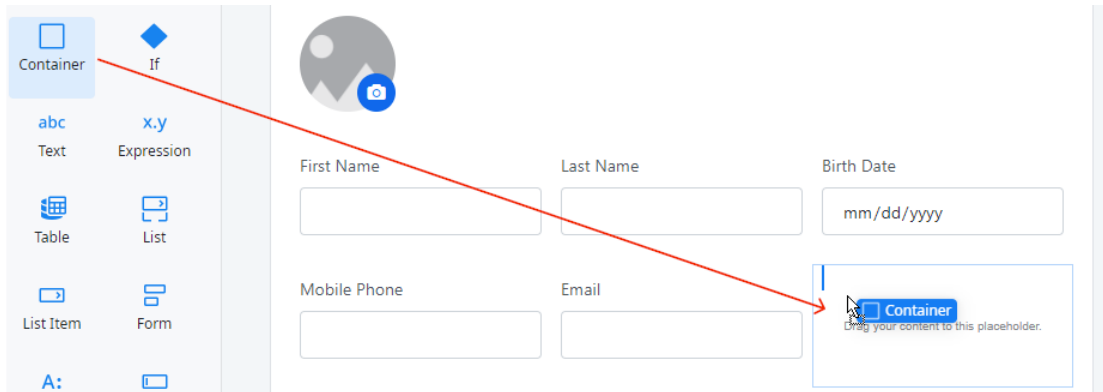


The screenshot shows the form with three columns: 'First Name', 'Last Name', and 'Birth Date'. Below these is a 'Mobile Phone' field and an 'Email' field. A 'Container' widget is being dragged from the palette to the first column placeholder. Red arrows indicate the drag-and-drop actions for the 'Mobile Phone' and 'Email' fields into the first and second columns respectively.

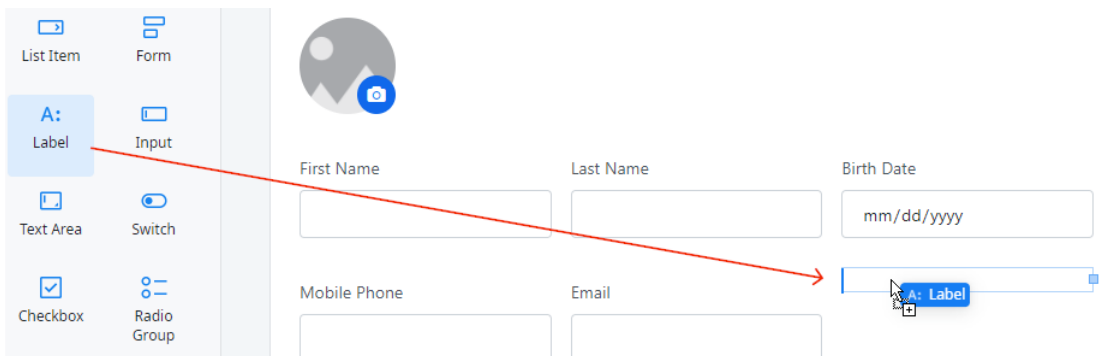
Note: You can use the Widget Tree or drag the Containers directly in the Screen preview.

The steps for the last Column are a bit different than the previous ones. Since the password is a special field, let's build it manually.

- 3) Drag a **Container** from the left sidebar and drop it in the third Column.



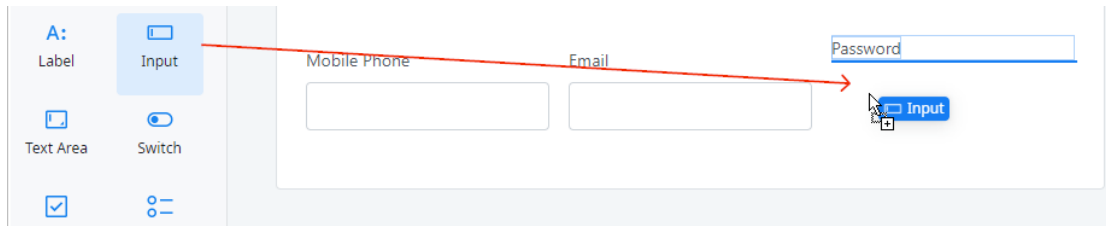
- 4) Drag and drop a **Label** inside the Container.



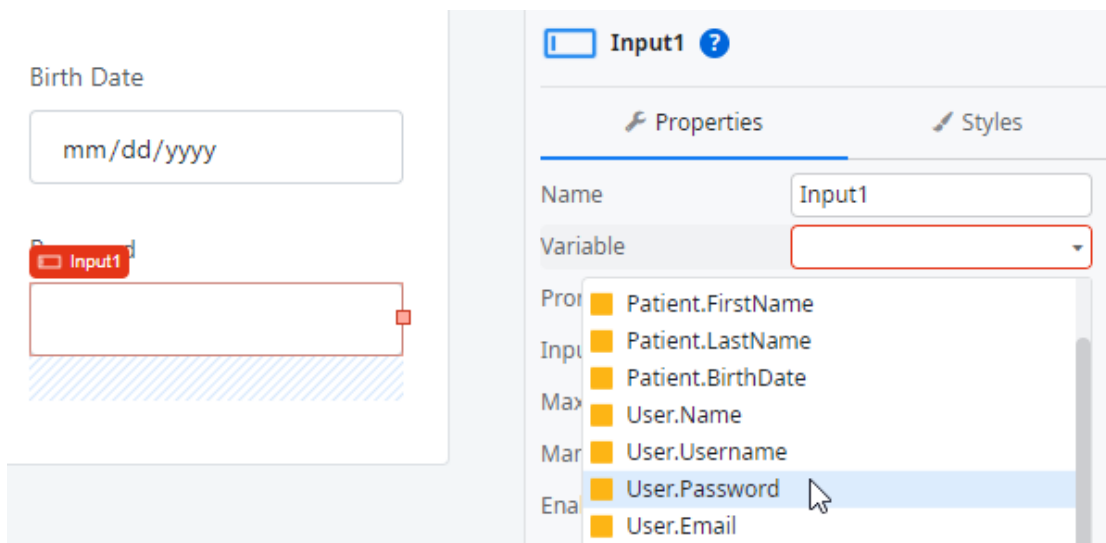
- 5) Type *Password* in the Label Text.



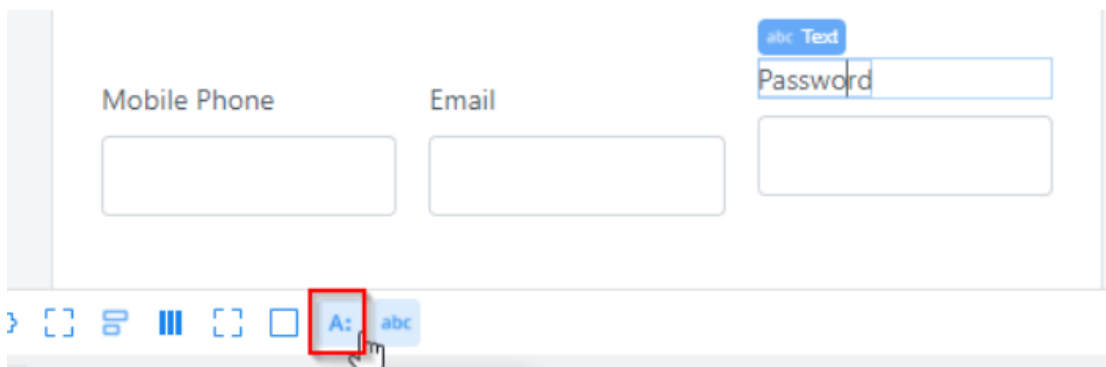
- 6) Drag and drop an **Input** under the Label.



- 7) Set its **Variable** property to **User.Password** on the properties area.

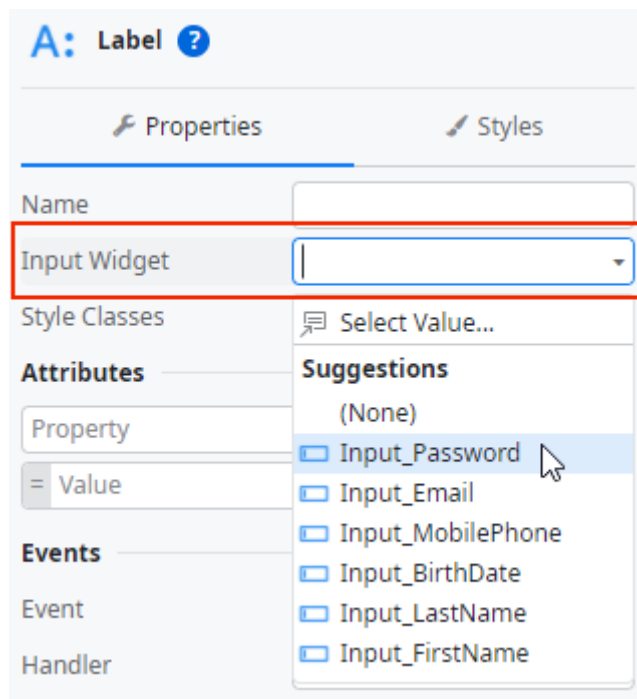


- 8) Select the Password text from the Label and then select the **Label** element, by navigating the breadcrumbs at the bottom of the page in Service Studio.

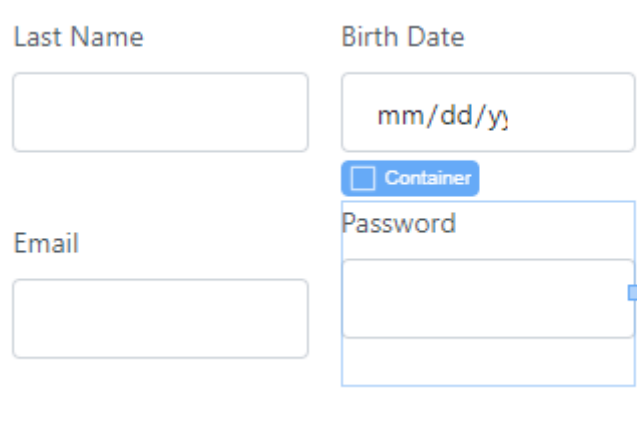


When there are elements nested inside other elements, the breadcrumbs, are very helpful to select the right thing.

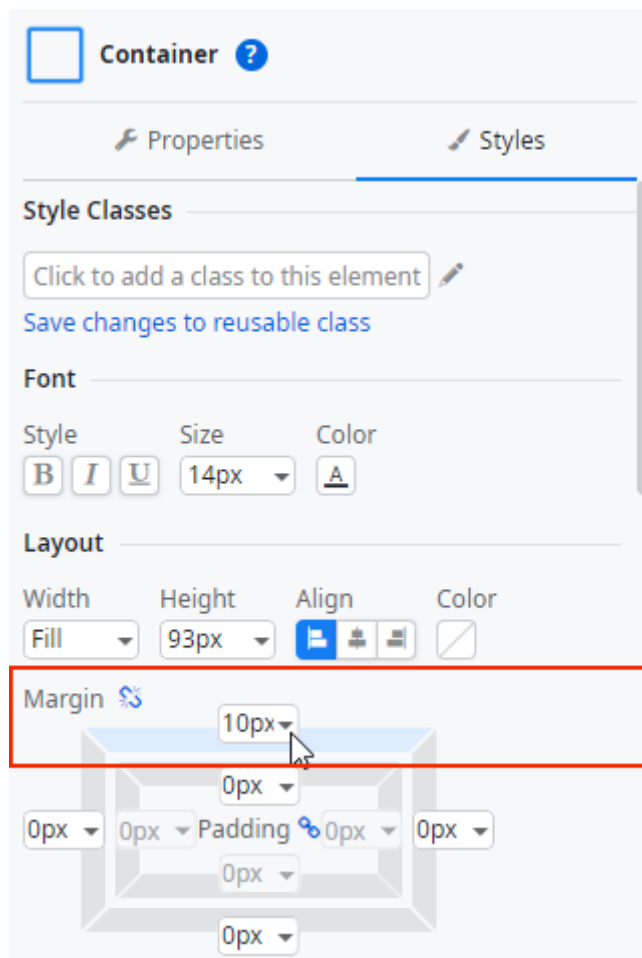
- 9) In the properties of the Label, click on the dropdown of the **Input Widget** property and select the **Input_Password**.



- 10) Click on the **Container** surrounding the password Label and the Input.

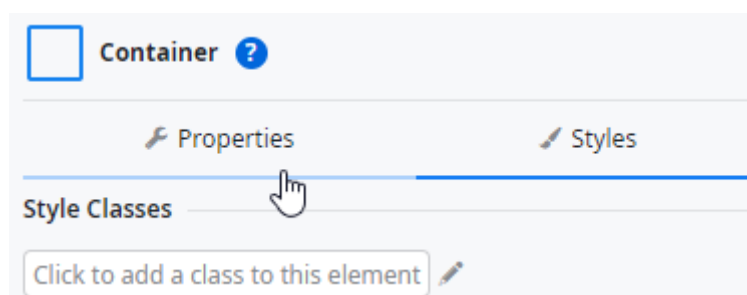


- 11) In the properties area on the right sidebar, you can find a **Styles** tab. Click on it and then set the **margin top** to *10 px*.



The **Styles** tab is available for every visual elements that we drag and drop to the Screen, and it's where you can edit and apply some styles to the elements, such as margin, font, etc.

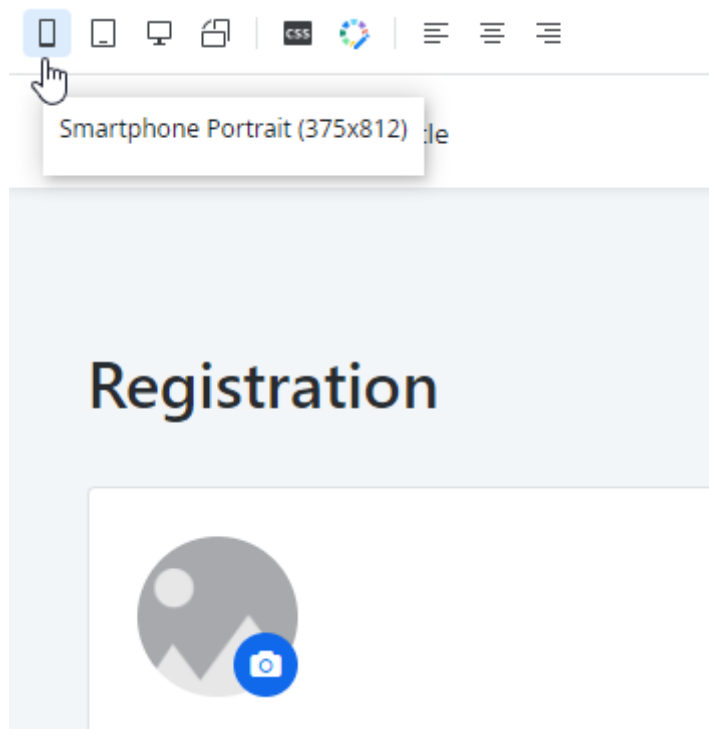
Go back to the Properties area, by selecting the **Properties** tab.



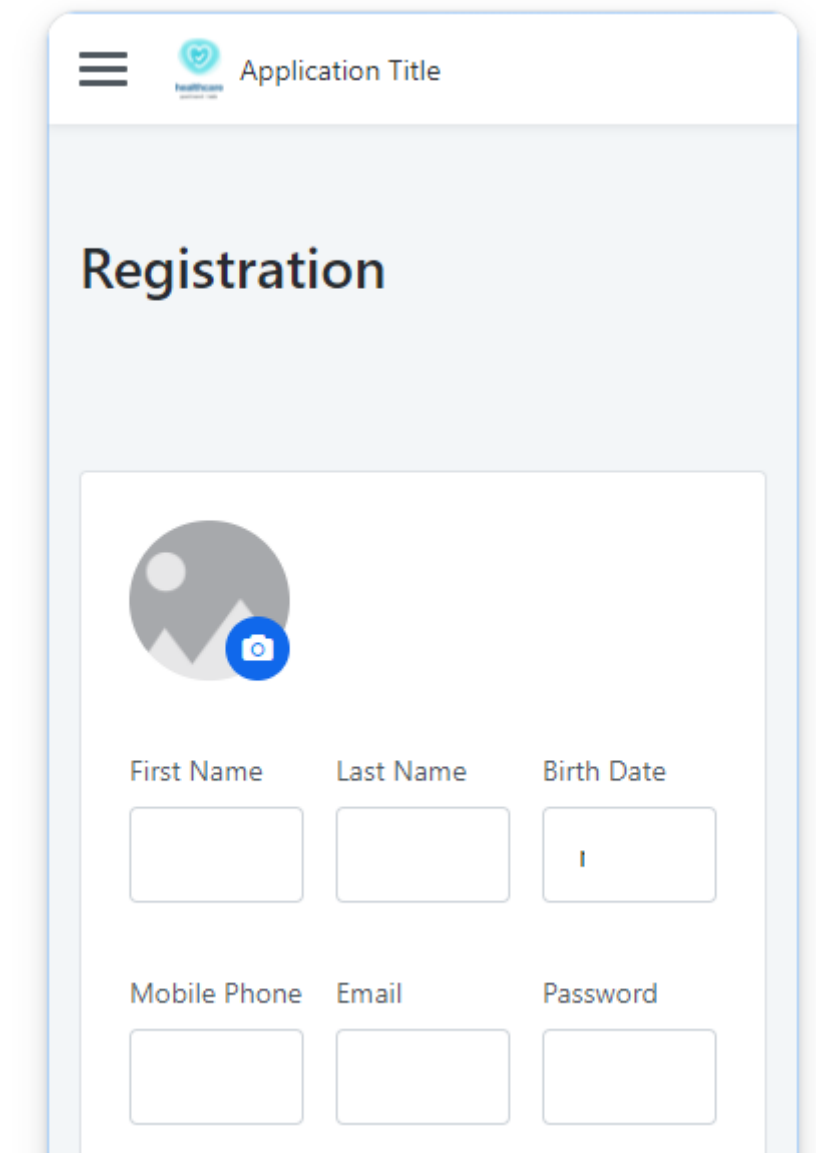
Mobile Preview

Did you know you can see the Screen preview in different formats? Let's make some adjustments, so the Form looks better in a Mobile Screen.

- 1) First, click on the **Smartphone** Icon on the top of the Screen preview to change the preview to a smartphone layout.

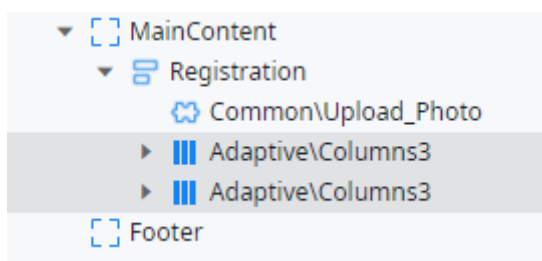


This is how your Form would look like in a smartphone:

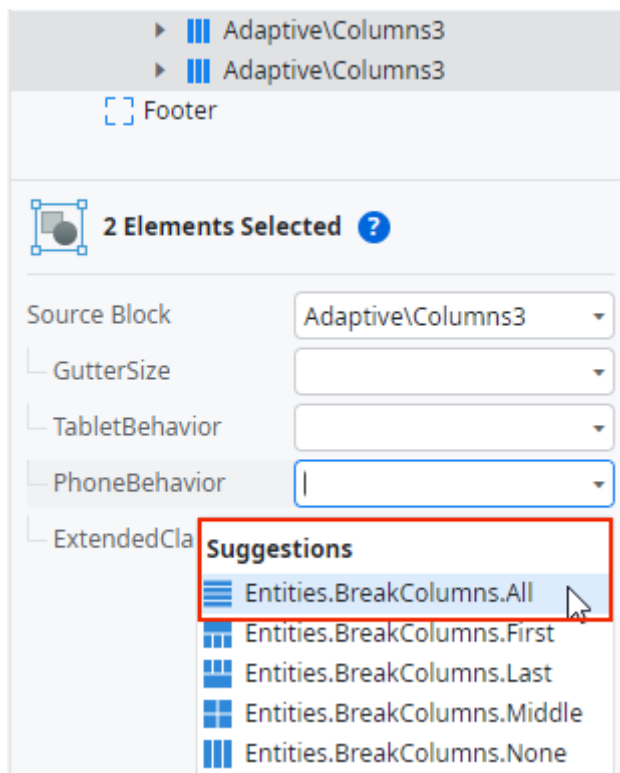


A smartphone mockup showing a registration form. The top bar includes a hamburger menu icon, a logo, and the text "Application Title". The main heading is "Registration". Below it is a white box containing a circular profile picture placeholder with a camera icon. Underneath are six input fields arranged in two rows of three. The first row is labeled "First Name", "Last Name", and "Birth Date". The "Birth Date" field contains the letter "I". The second row is labeled "Mobile Phone", "Email", and "Password".

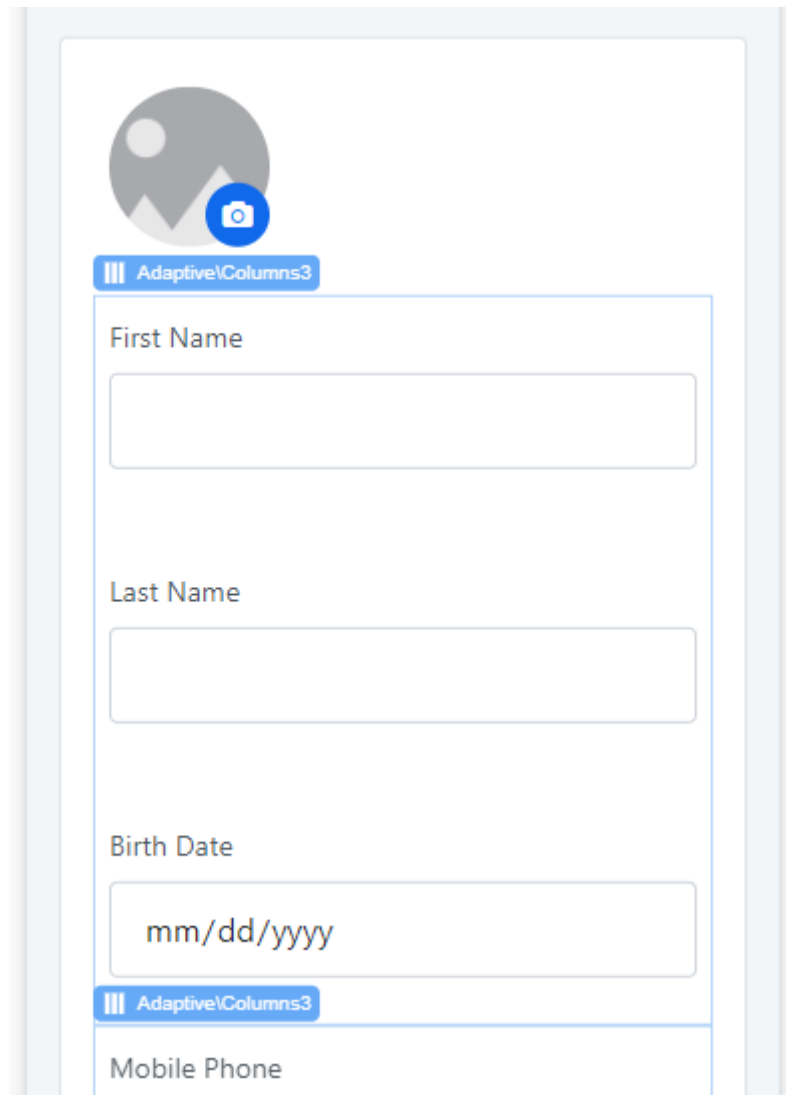
- 2) Open the **Widget Tree** and select both **Columns3** elements.



- 3) Then, click on the **PhoneBehavior** property and select the option **Entities.BreakColumns.All**.



Now, you're "telling" the columns to break when the app is used on a smartphone. OutSystems does the rest for you. That's pretty cool, right?

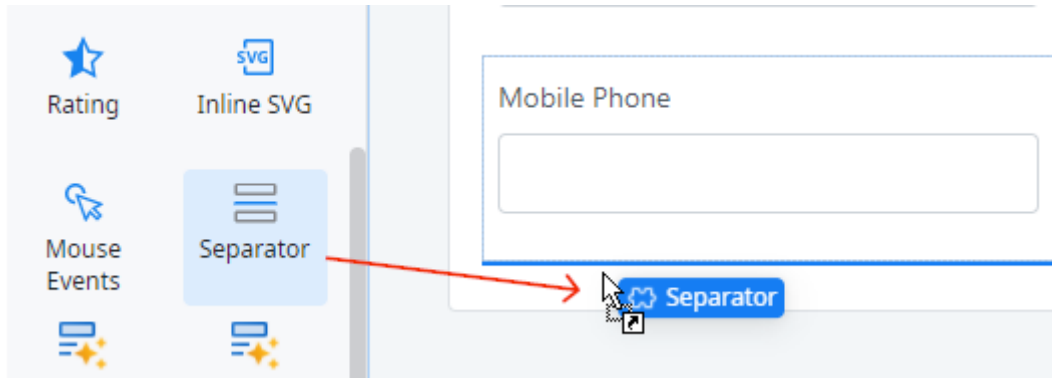


Patient Onboarding Fields

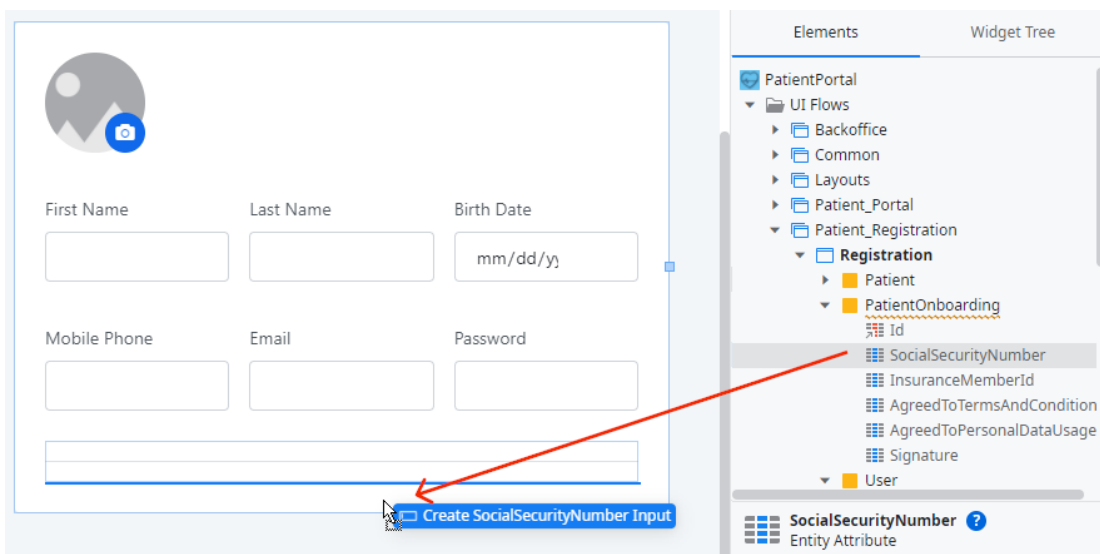
Let's add a couple more fields to the Form, this time for the patient social security and insurance information, which is important for the PatientOnboarding data.

- 1) Click on the Smartphone preview icon again to change the preview back to how it was.

- 2) Drag a **Separator** from the left sidebar and drop it in the Form under the last Columns 3, to give some space between the previous Form fields and the new content.

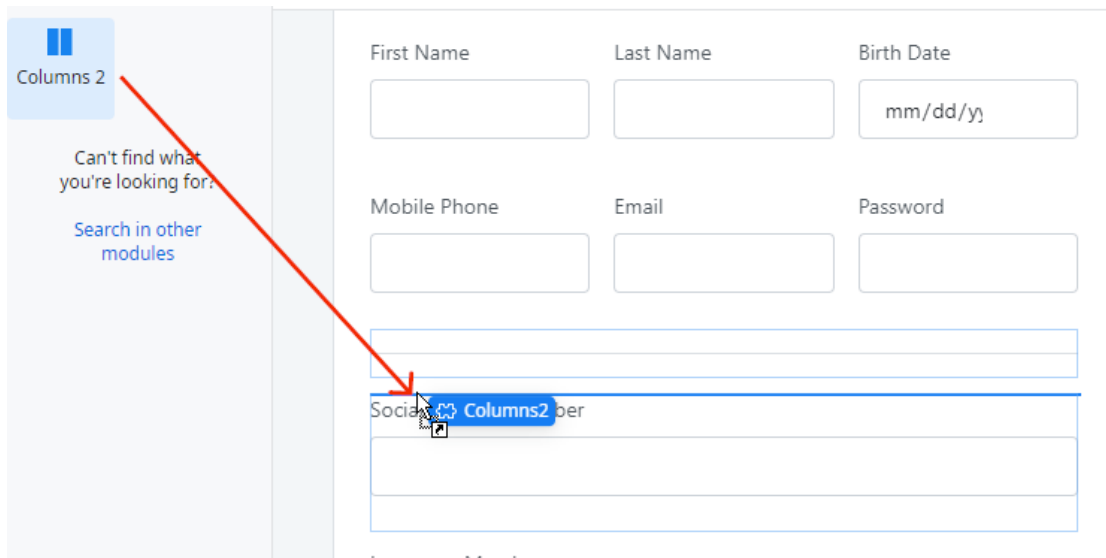


- 3) Expand the **PatientOnboarding** Local Variable, select the **SocialSecurityNumber** attribute, then drag and drop it under the Separator. If you're still seeing the Widget Tree, don't forget to switch to the Elements tab.

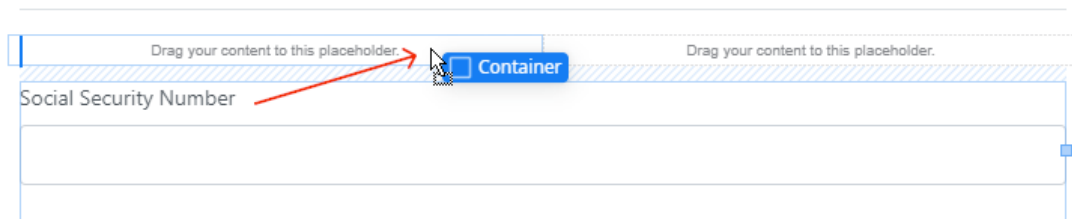


- 4) Do the same for the **InsuranceMemberId** attribute and drop it after the Social Security field.

- 5) Drag a new **Columns 2** element and drop it right after the Separator.



- 6) Drag the Containers with the two new fields to the first and second column of the Columns 2 respectively, just like we did in previous sections.



Your Form should look like this:

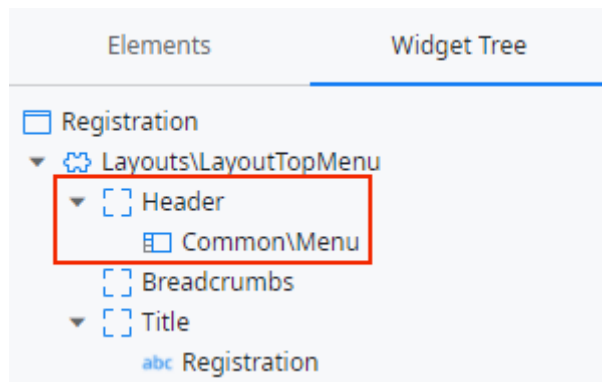
Mobile Phone	Email	Password
<input type="text"/>	<input type="text"/>	<input type="text"/>

Social Security Number	Insurance Member
<input type="text"/>	<input type="text"/>

Menu

Since this Screen is for users that are not registered yet, the menu of the app should not be visible. So let's remove it from the Screen!


- 1) On the Registration Screen, open the **Widget Tree** and expand the **Header** section.




- 2) Right-click the **Common\Menu** Block and delete it.
- 3) Publish the module and open it in the browser.



You will see the Screen you just created. It is looking good, but not ready yet.
We will continue on the next tutorial!

☰  PatientPortal

Registration



First Name *	Last Name *	Birth Date *
<input type="text"/>	<input type="text"/>	<input type="text" value="mm/dd/yyyy"/>
Mobile Phone	Email	Password
<input type="text"/>	<input type="text"/>	<input type="password"/>
Social Security Number *	Insurance Member *	
<input type="text"/>	<input type="text"/>	

Wrapping up

Congratulations on finishing this tutorial. With this exercise, we had the chance to go through some essential aspects of OutSystems, and even and 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) [OutSystems Overview](#)
- 2) [Logic](#)
- 3) [Building Screens with Data](#)

See you in the next tutorial!