Demo Creating a Mobile App. With OutSystems

Developing mobile apps with **OutSystems** is easy. If you have an Excel file containing your data, you can import it into a database and quickly create a mobile app that enables you to check and manage your data on the go.

To create a mobile app with data that's imported from an Excel file, you need to:

1. Create a database model, and import the data from the Excel file into the database
2. Create a screen that lists the data from the database
3. Synchronize your server database with your local database
4. Implement functionality to synchronize
5. Add a plugin to your application
6. Test the application on your mobile device.

Let's do this! In this example we'll use a sample Excel file with Employee information, and we'll create a simple Contacts mobile app.

1. Before you start

Before you start you must download two files from a GitHub repository https://github.com/Synobsys/Demo.

One file is called `Employee.xlsx` the other is an icon called `contacts.png`. Store these files on your PC.

Make sure you have your own personal environment installed from OutSystems.

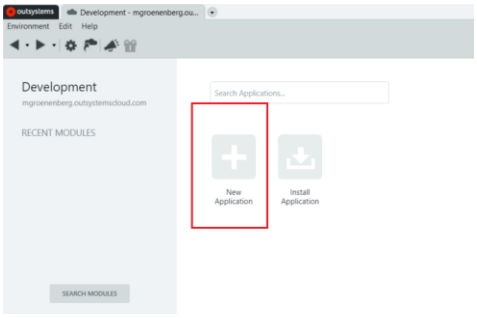
https://www.outsystems.com/home/GetStartedForFree.aspx

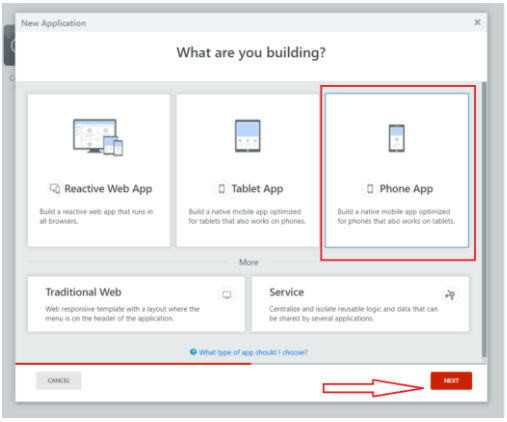
- Create an account.
- Install Service Studio.
- Connect to your personal environment,

Contents

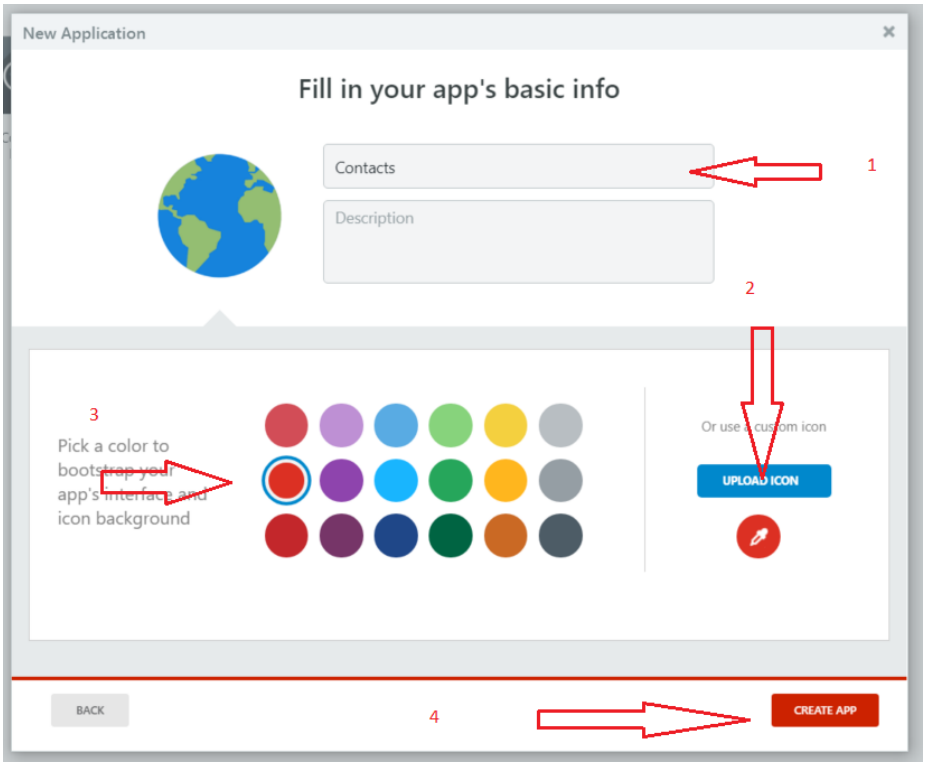
2. Creating your application

In Service Studio, click **New Application**, choose Phone **App**, and name it `Contacts`.

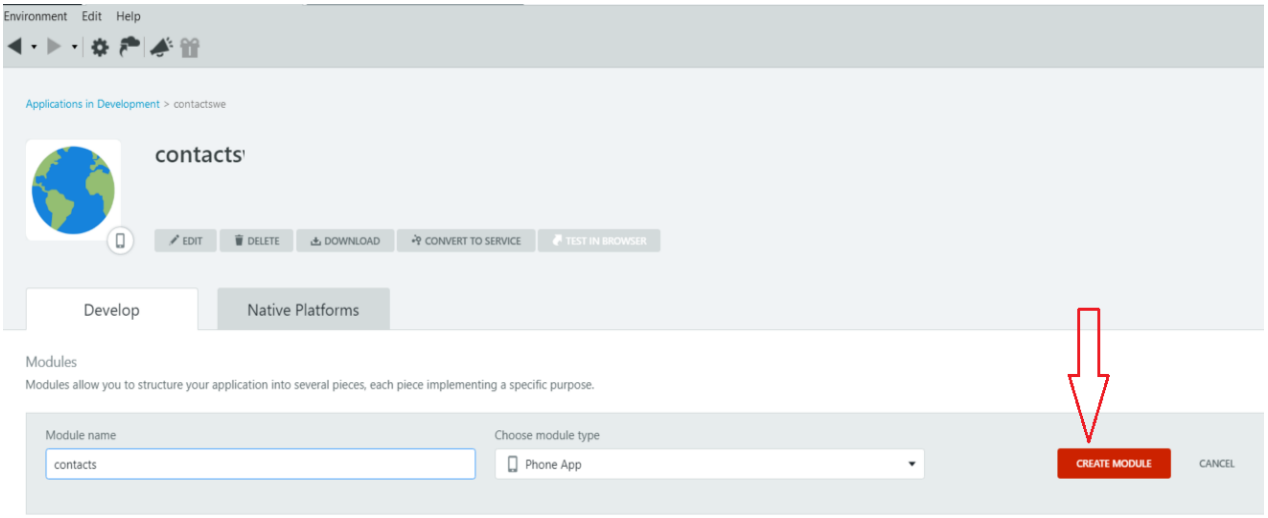




Select the Icon you downloaded from the site `contacts.png`. And select a base color for your application. Push the button **create app.**



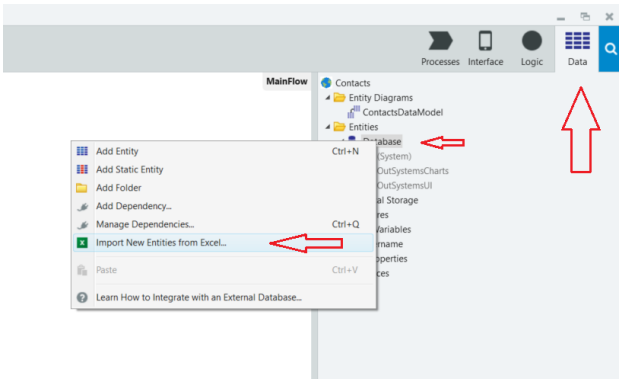
In the new **Contacts** application, create a **Phone App** module, named `Contacts`.



Push the button **CREATE MODULE.**

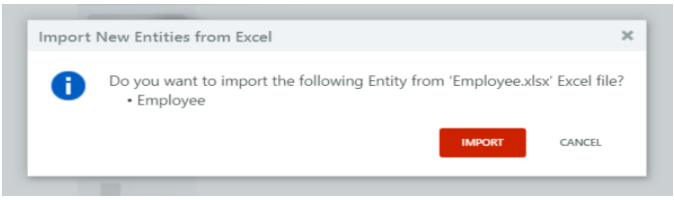
3. Creating the database

Open the D**atabase tab** in the top right header of the screen. Right click with your mouse on **Database** and select the option **Import New Entities from Excel.**

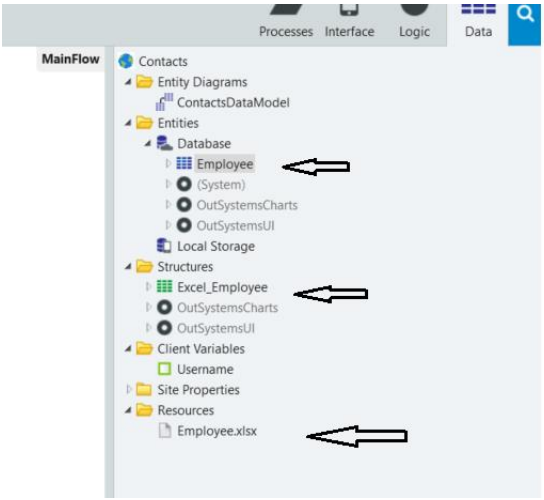


Select the downloaded file "`Employee.xlsx`" which you downloaded in step1 and open this file.

Wait for the confirmation and Import the file.



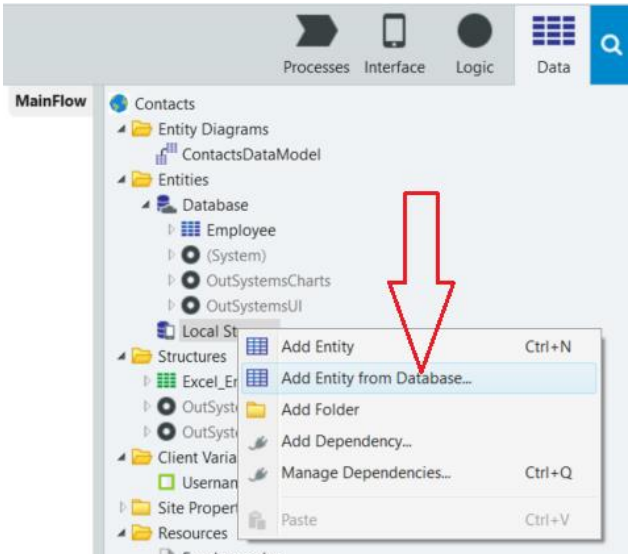
After the import your application looks like this and all definitions are imported by OutSystems.



4. Local Storage

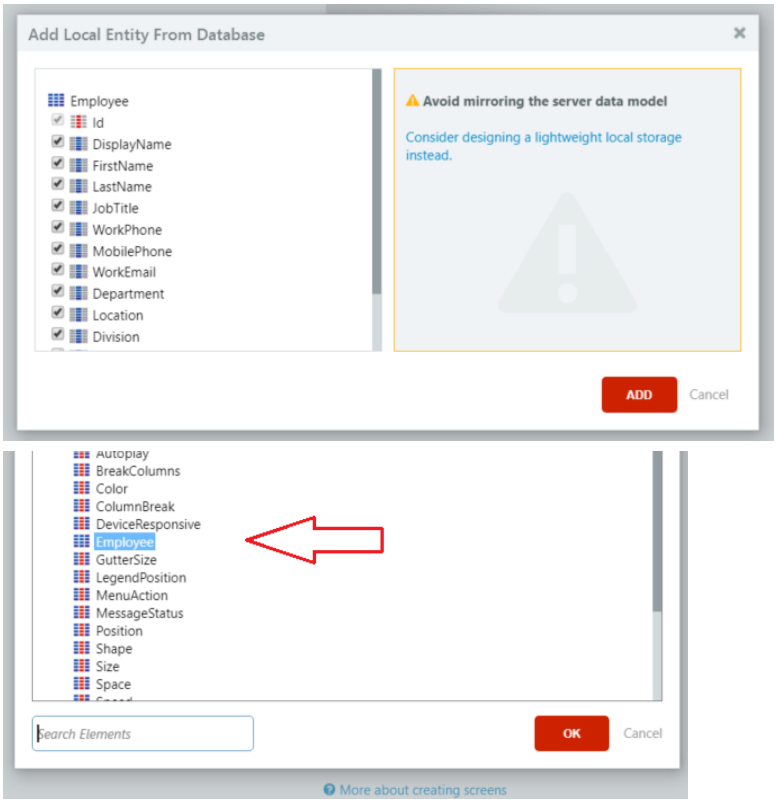
Because this file location is on the Server side, we also want this file on our local device. All Android- and IOS-devices uses SQL-Light and so it is easy to do. We can use our device also when it is not connected to the Internet and getting data from local storage is also faster.

Still in the database tab we click right with the mouse on **Local Storage** and select **Add Entity from Database**…



Select the `Employee` database and confirm with **OK**.

Now we can select what fields we want to use in the Local Storage. Because we do not always need all the fields on our local device and want to keep our Local Storage as small as possible. In this demo we select all the fields.

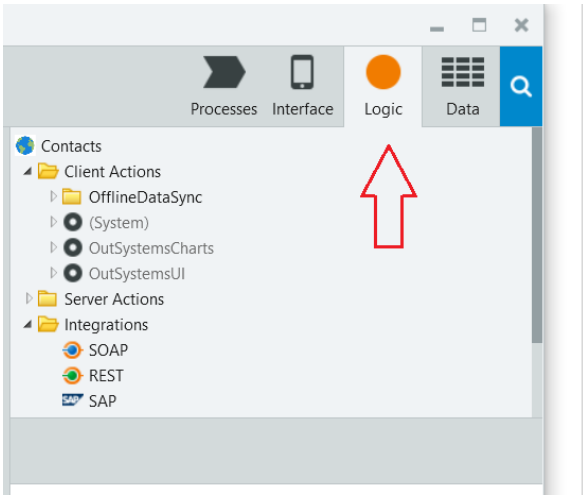


Now we are ready with the Database.

5. Synchronize the Local database

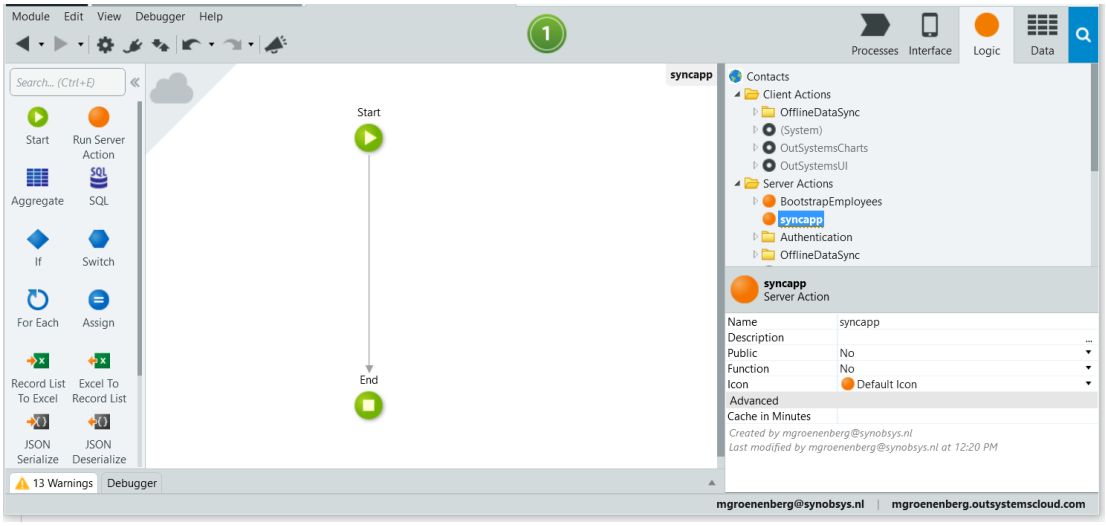
Because the **Database** is somewhere in the cloud and our **Local Database** is on our device, we need to import the records from this cloud database. In this Demo we only import the records one time and do not create a real synchronization for these databases.

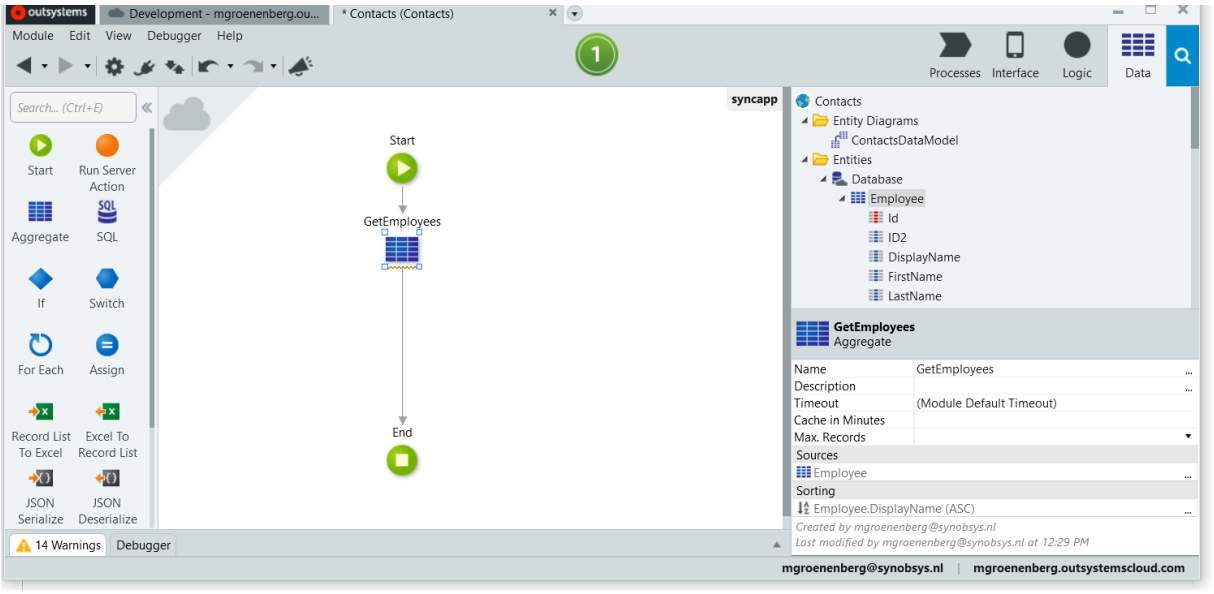
Open the **Logic** perspective



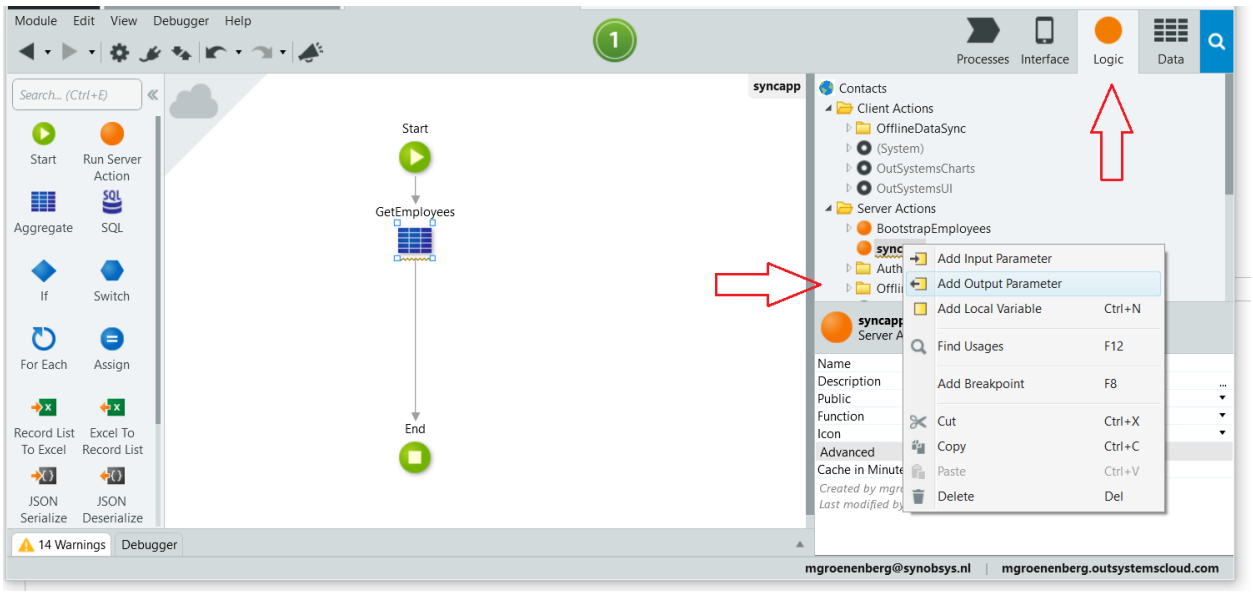
Go to **Server Actions** right click and **Add Server Action** and name it `syncarp`.

Now when you are on your new created action open the **Database Tab** and drag and drop the **Entity `Employee`** on the line of your action. An Aggregate **`GetEmployee`** is created which will retrieve all the records from `Employee`.





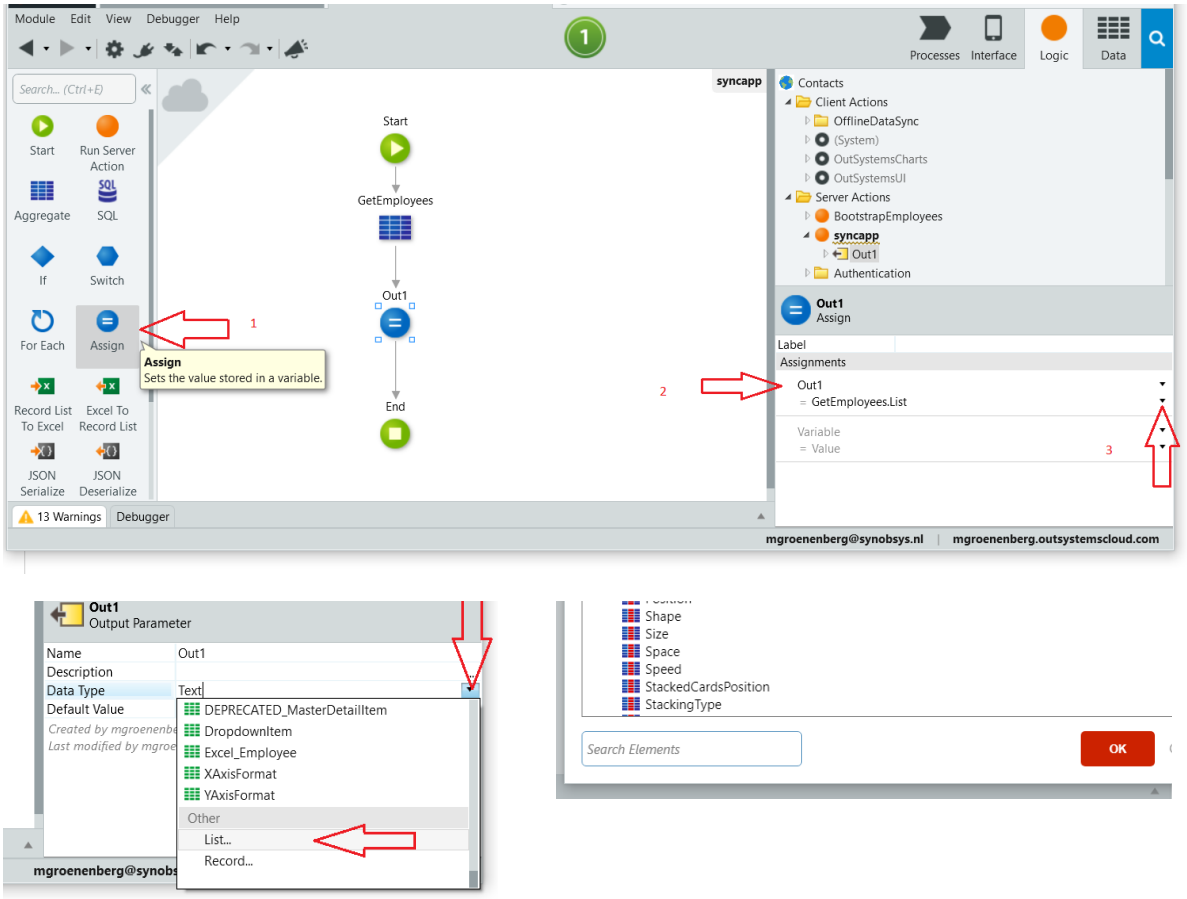
Go back to the Logic tab. Select your action `syncapp`, right click on this action and **Add Output Parameter**. Let the name be "`Out1`".



Change the **Data Type** of this Output Parameter "`Out1`" to **List** and after that select the Entity `Employee`.

In your action `syncapp` select from the left pain the **Assign**, drag and drop this on the action line under the **Aggregate**.

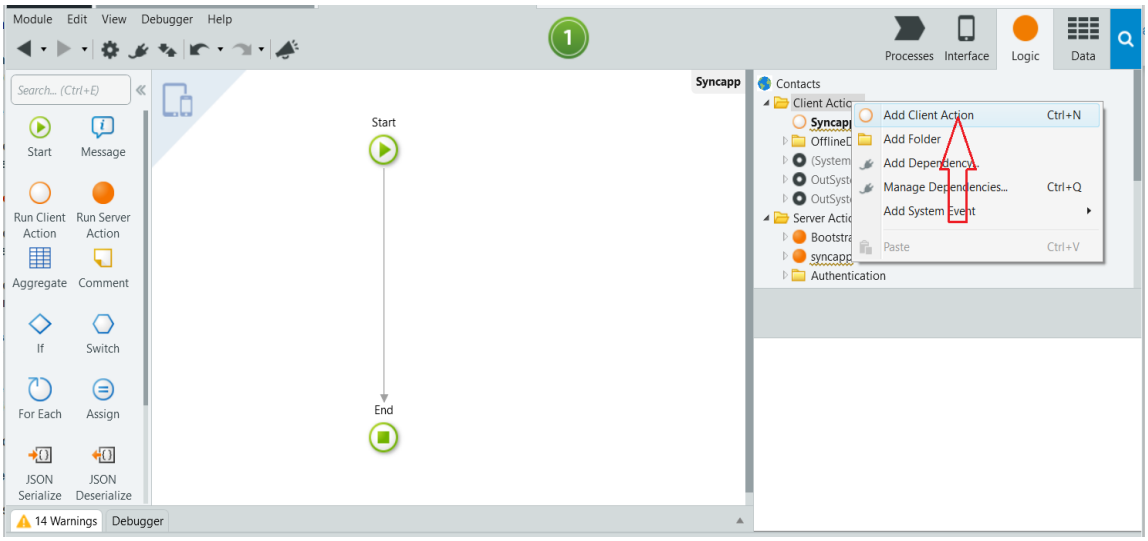
In the Assignments pane (in the right corner) select as variable the `Out1` parameter and as input the list from `GetEmployees`.



Your `syncapp` Server side is ready. Now on the Client side we also need an Action.

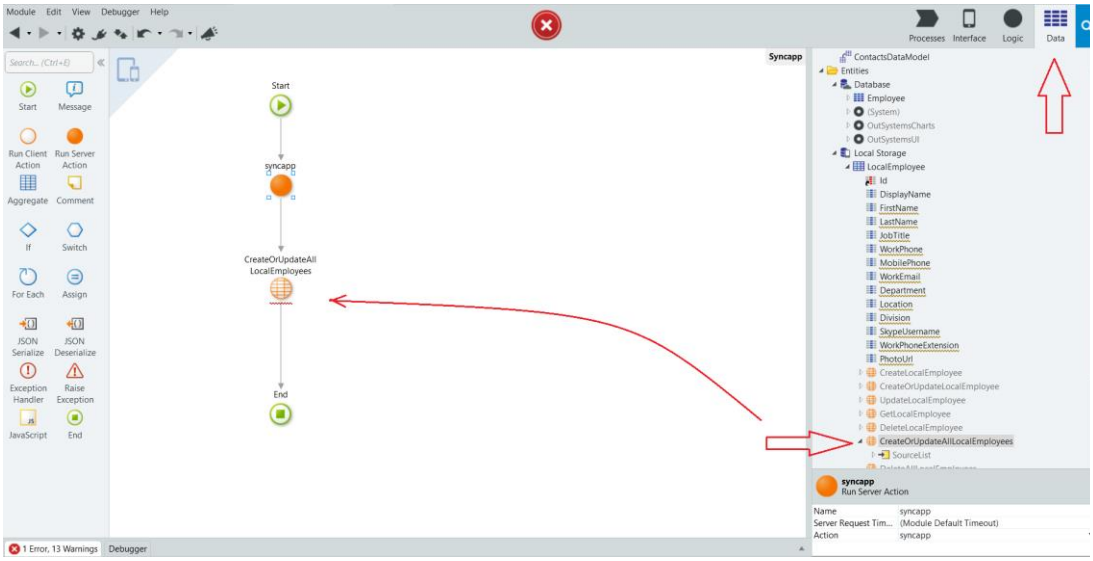
6. Syncapp Client Side.

Right click on the **Client Actions** and **Add a new Client Action**. Name it also `syncapp`.

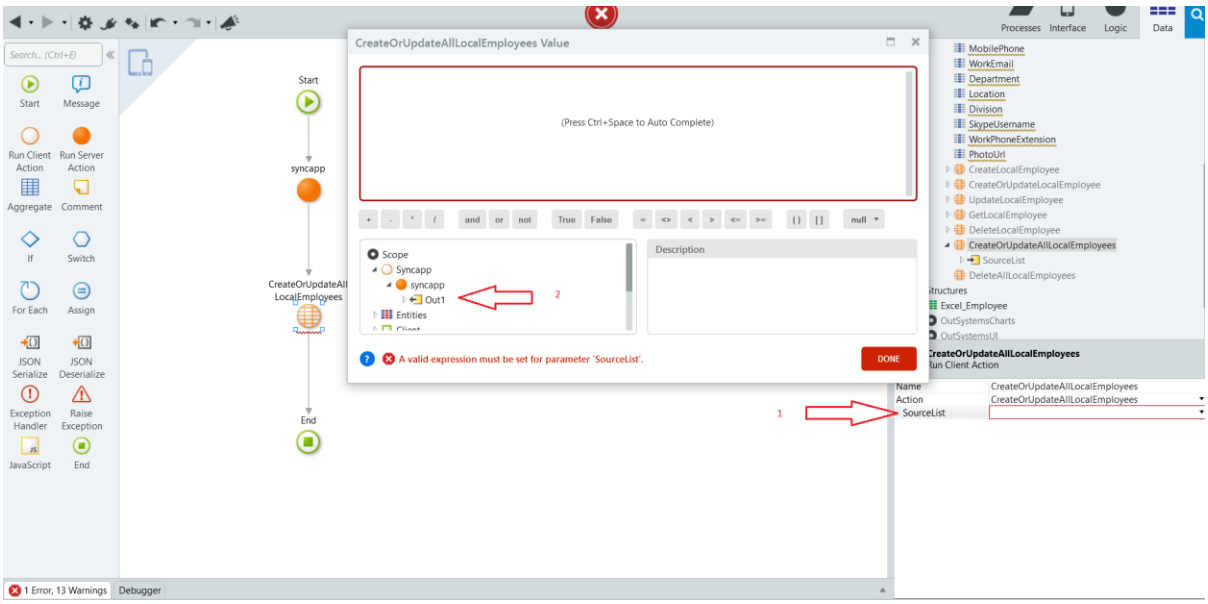


Drag and drop the early created Server Action `syncapp` on the line of the action diagram.

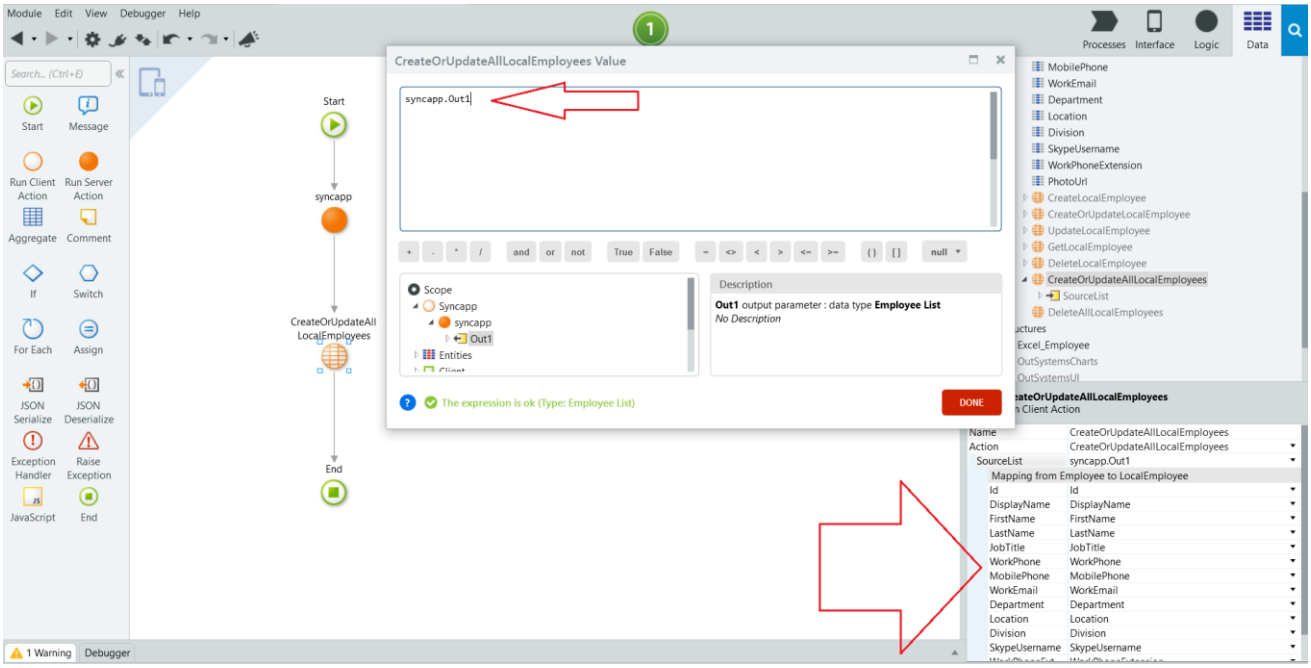
Now open the **Database tab** and go to the **Local Storage**, Select the Entity LocalEmployee and go to the action `CreateOrUpdateAllLocalEmployees`, drag and drop this action on your action diagram.



Double click on the **Source List** of this action so the **Expression Editor** of OutSystems pop's up. Select the output parameter `Out1` of the server action.

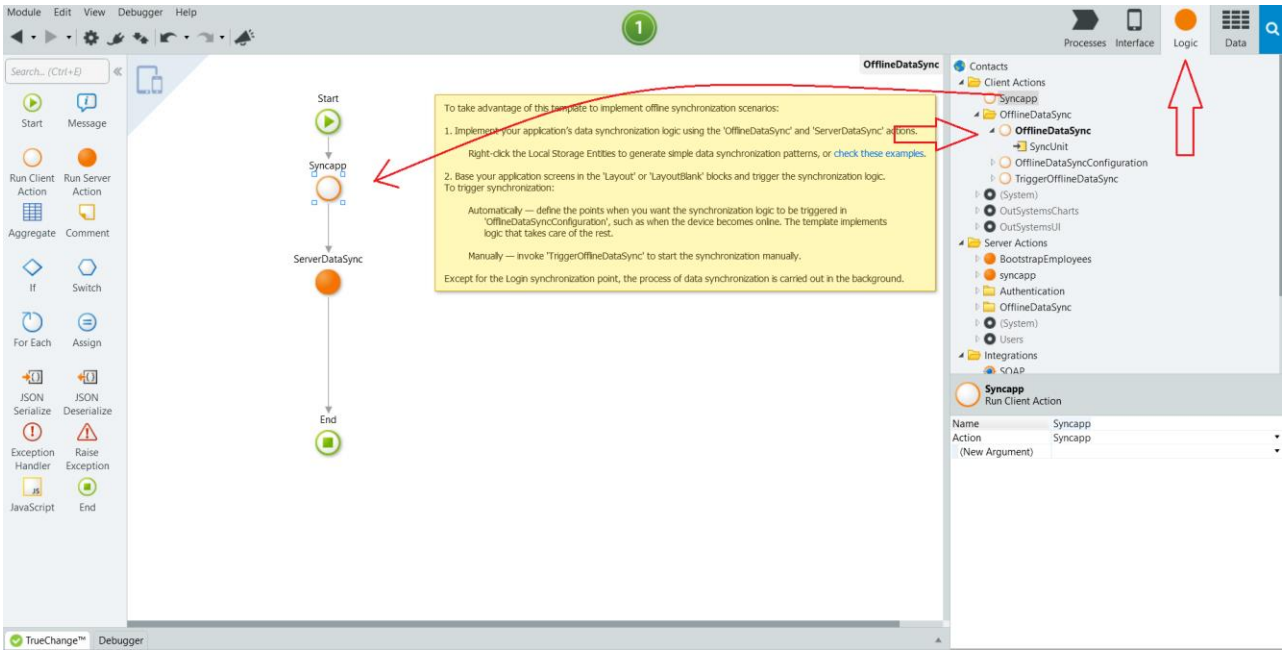


Note that all the fields from the server action are mapped to the fields of the client action.



On the **Logic tab** open the action `OfflineDataSync`.

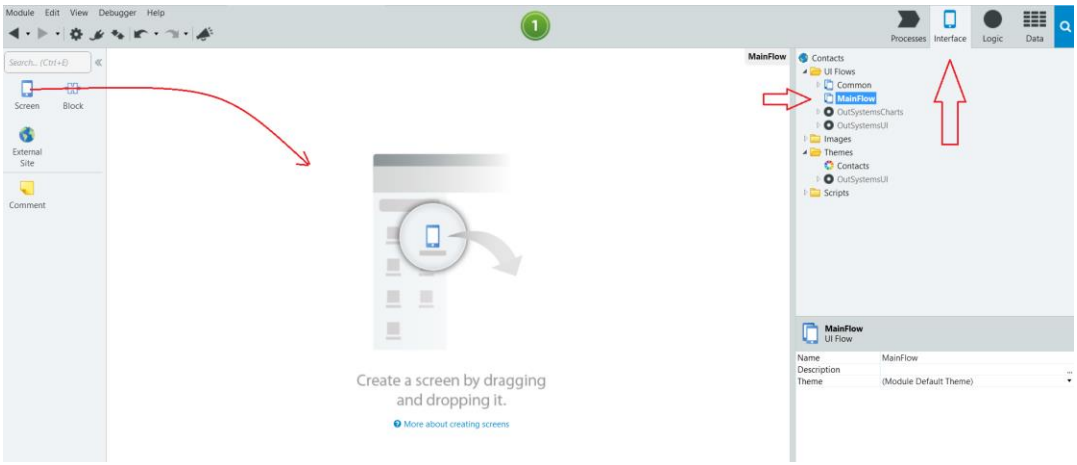
Now drag and drop your new `syncapp` to this action.



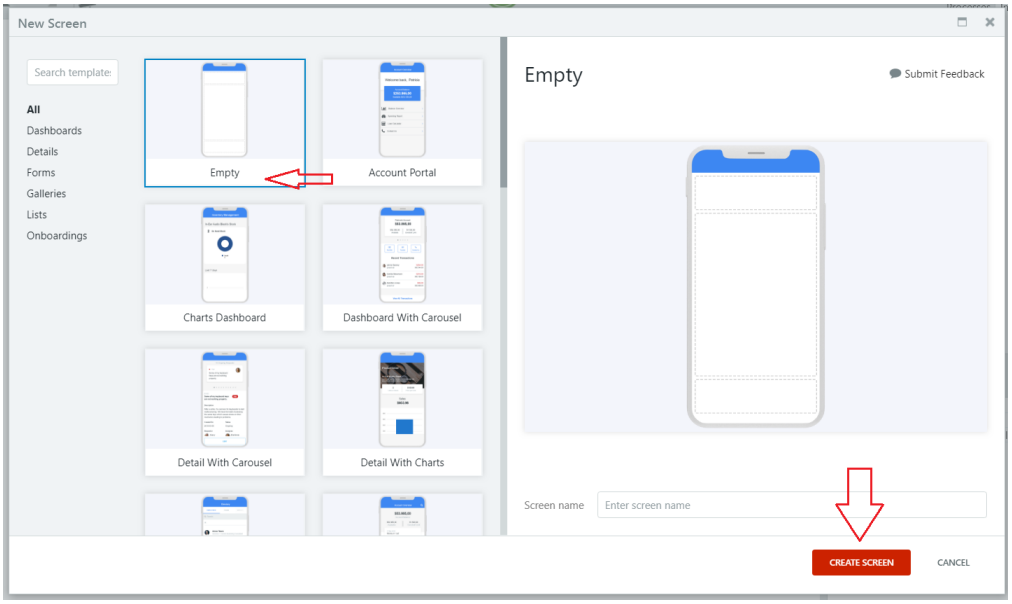
That’s all for the database and synchronization now lets’ go to the Screens.

7. Screens

Go to the Tab **Interface** (right top corner) and select and double click the **Mainflow**. From the Left pain select the **Screen** Icon and drag and drop it on your screen.

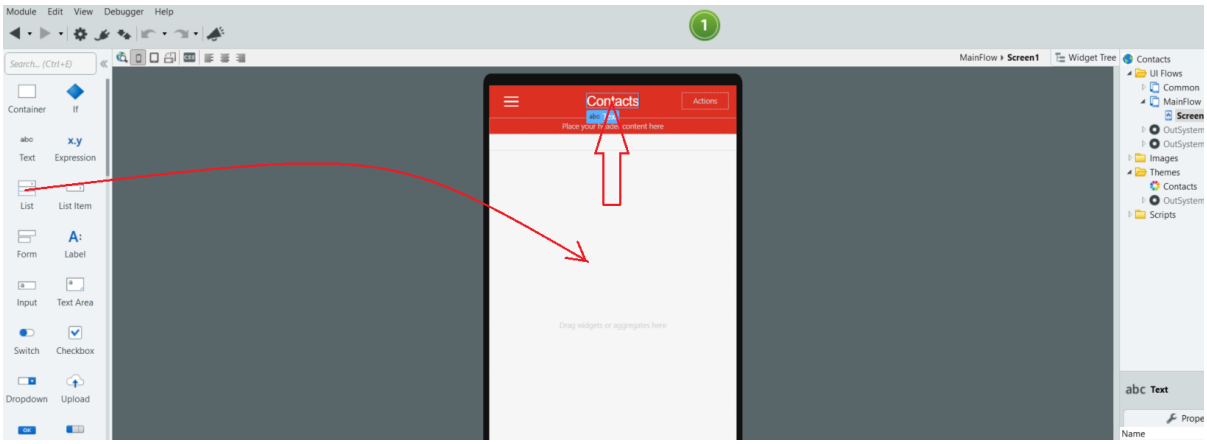


Select the empty screen and hit the Button **CREATE SCREEN**.

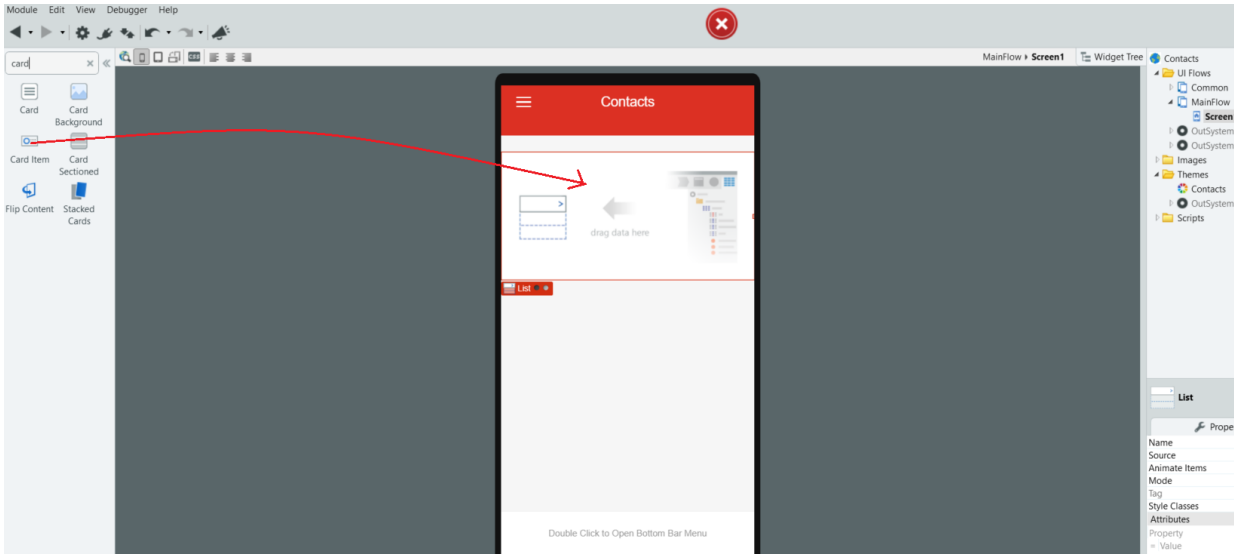


Change the Title to `Contacts`.

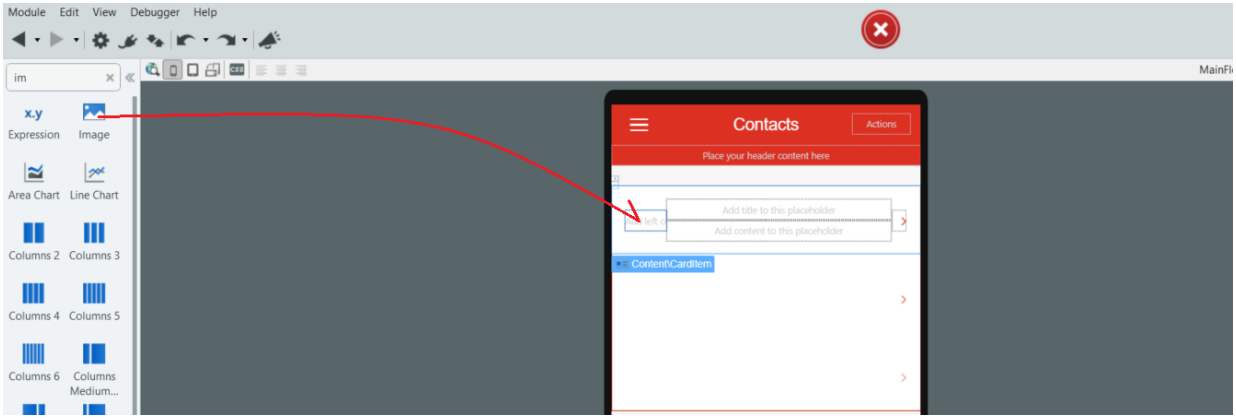
Drag and drop from the left pane the widget **List** on the main section of your screen.



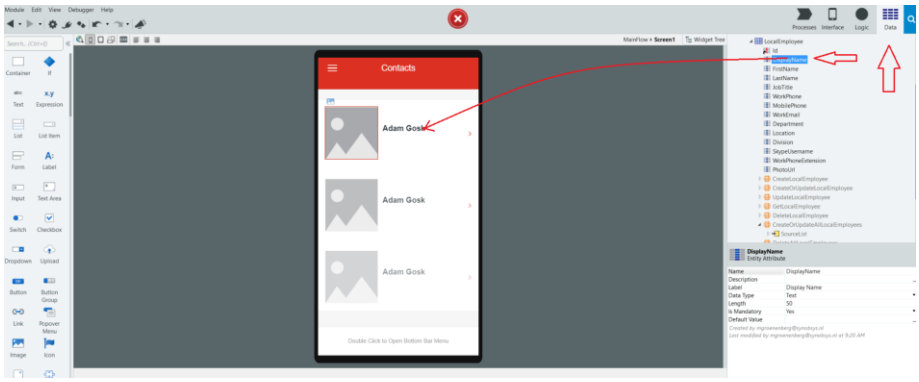
Drag and drop from the left pane the widget **Card Item** on the **List** you just created.



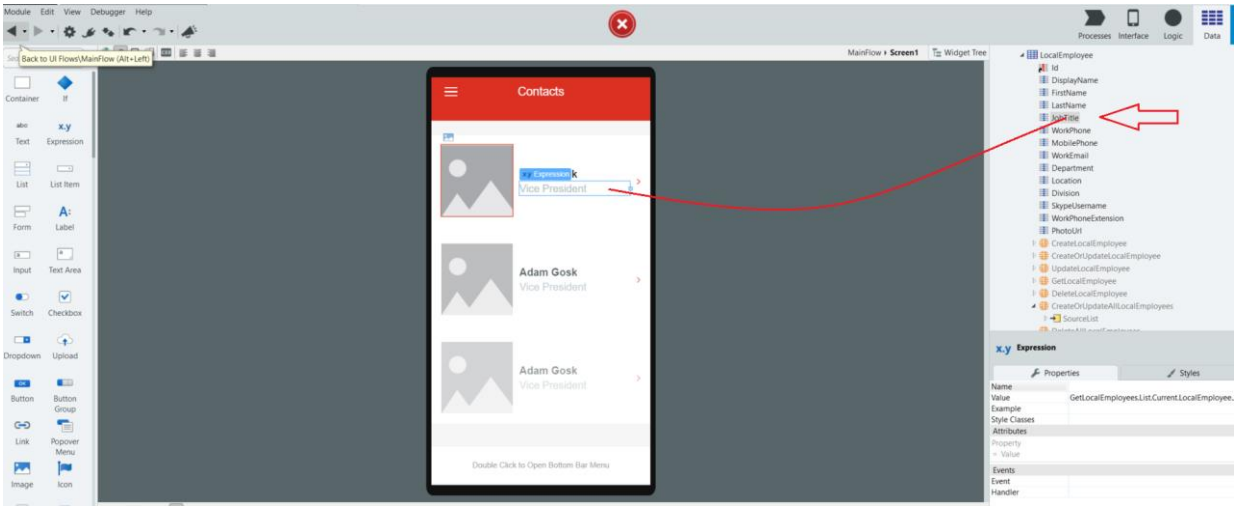
Drag and drop from the left pane the widget **Image** on the left of the **Card Item** you just created.



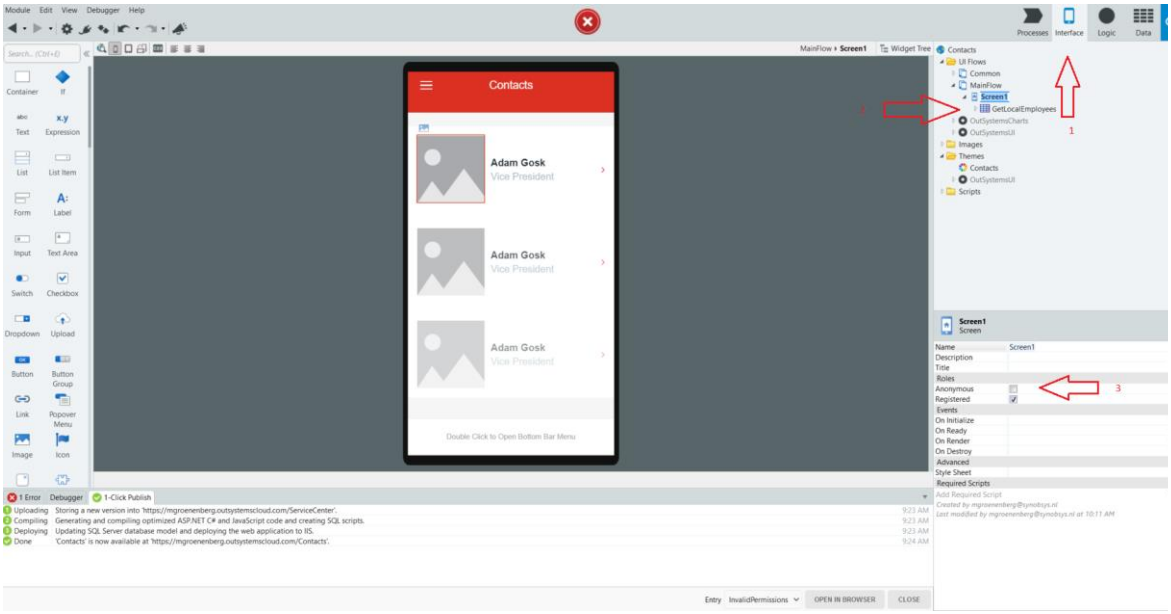
Now open the **Database tab** and go to the **Local Storage** and select the file `LocalEmployee`. From this file select and drag and drop the field `DisplayName` on the first field of your list.



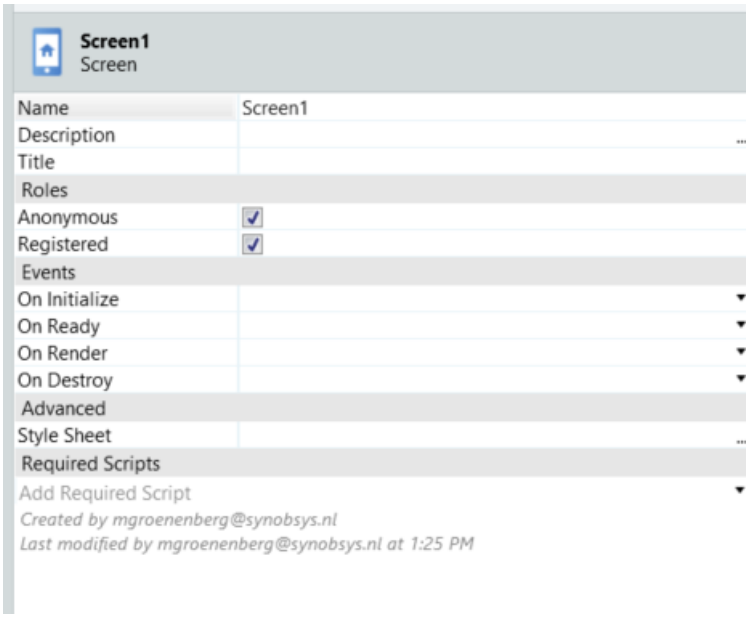
Drag and drop the field `JobTitle` to the second field of the list.



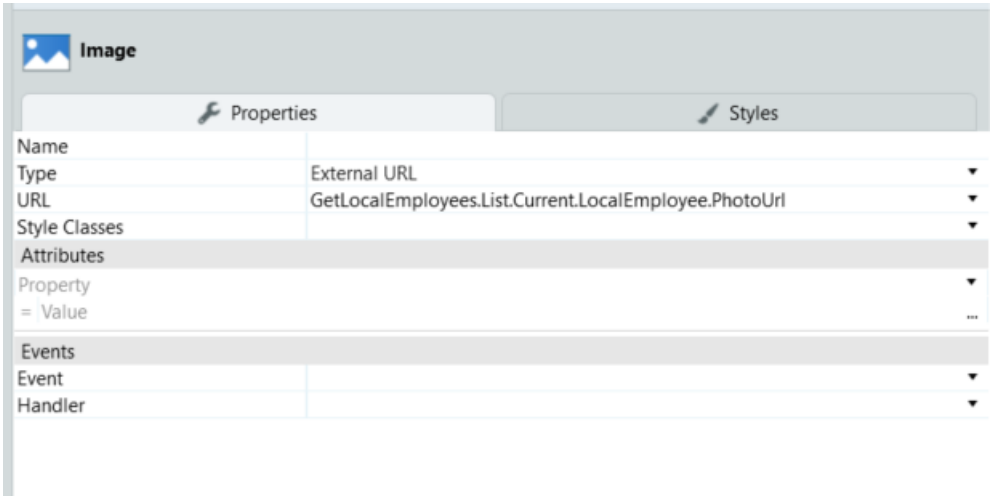
Go back to the **Interface Tab** and notice that under your `Sreen1` an aggregate is created by OutSystems named `GetLocalEmployees`.

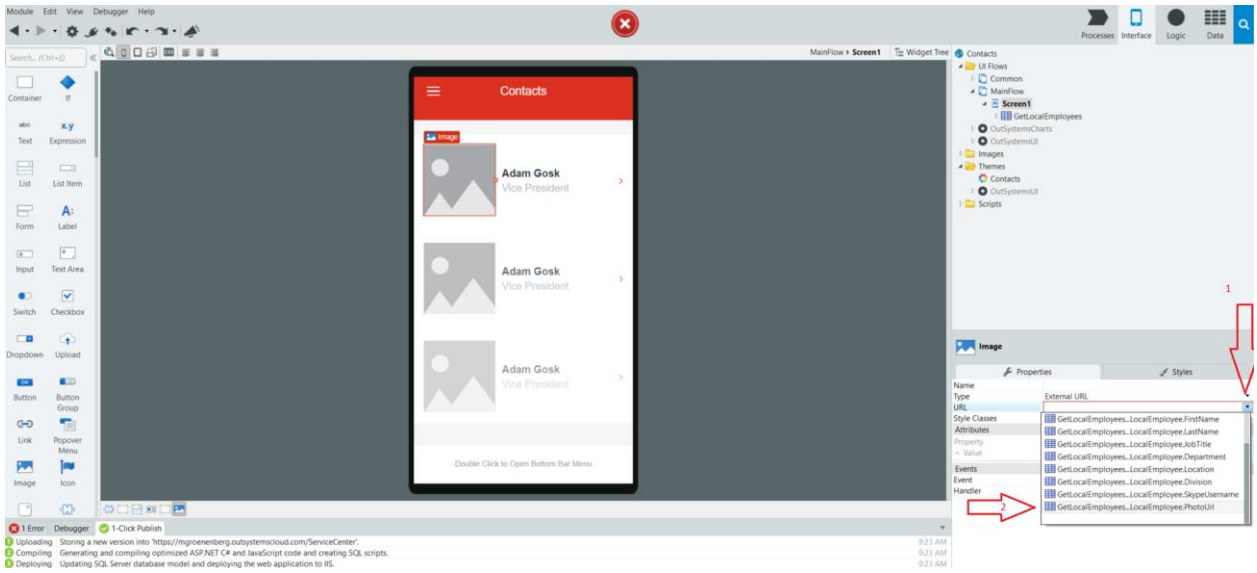


Because we do not want to consider about authorization for this screen we add the **Role** Anonymous to this screen. Click the select box for Anonymous.

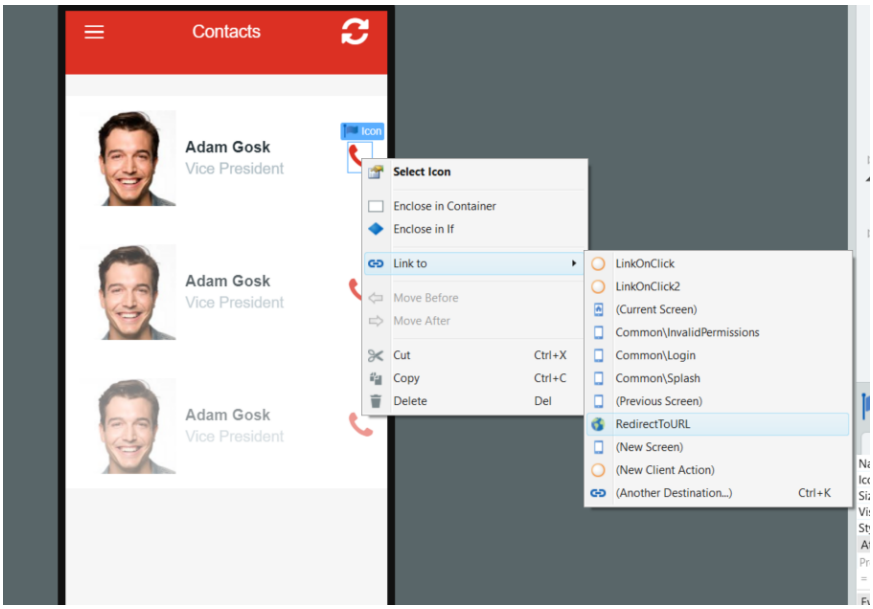


Now right click on the **Image** in your **List** and go to the properties pane and make the type to **External URL** and for the **URL** select the field `GetLocalEmployees.List.Current.LocalEmployee.PhotoUrl`.



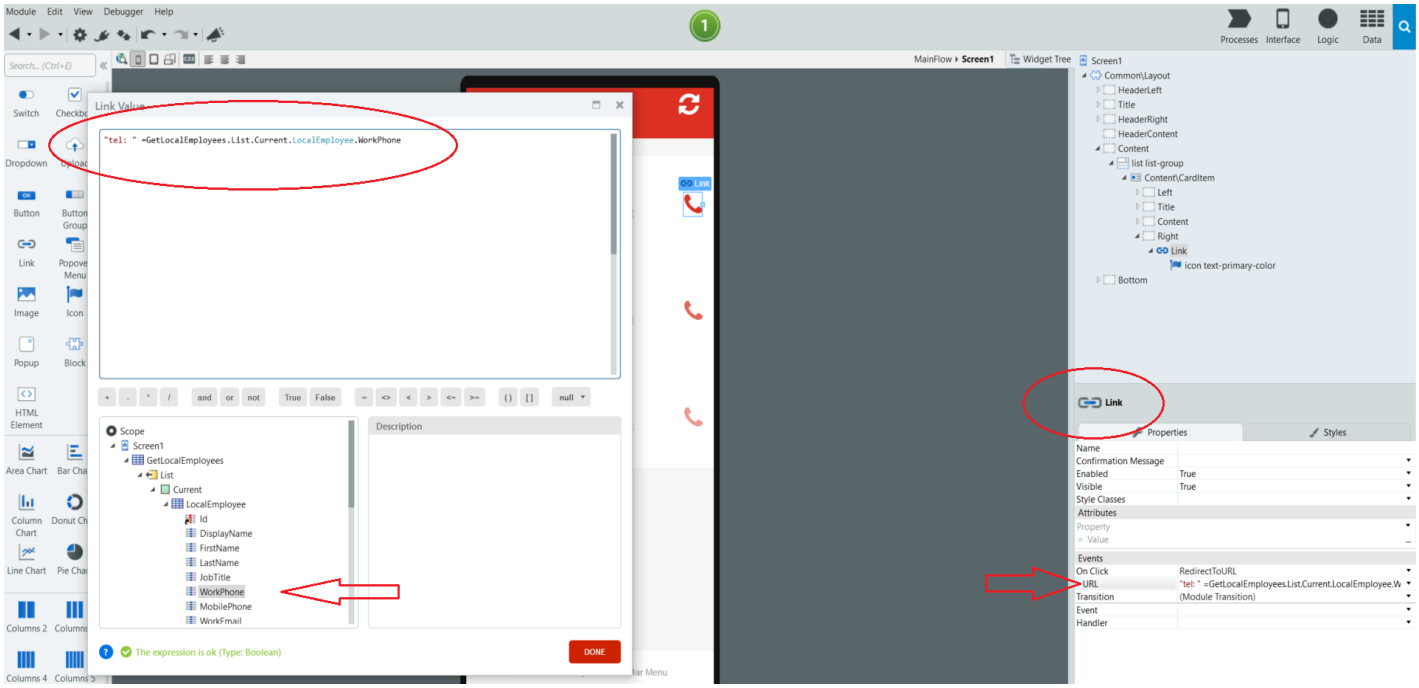


Now click on the Icon “>” in your List and change the type to “phone”. Make the Size to 2x font size. Right click on the icon and select **Link to > RedirectToURL**

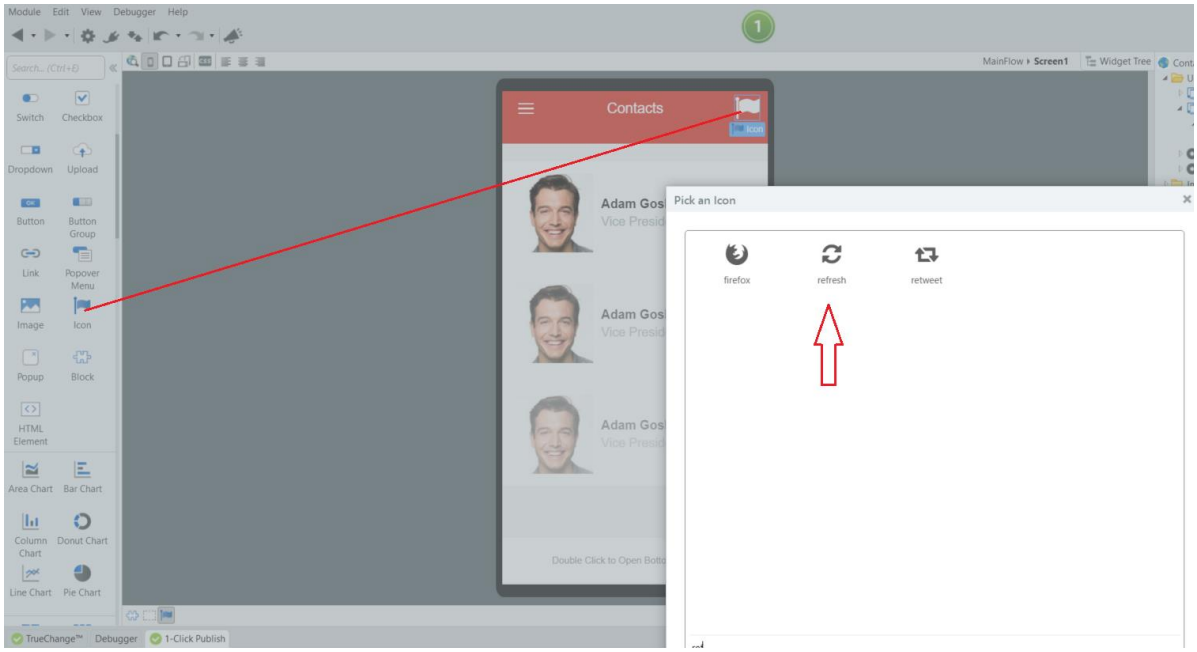


On the **Properties** of this link **Event** Double click on the **URL** parameter and add the text: “tel: ” + <select the workphone from the current LocalEmployee> so it looks like this:

```
"tel: "+GetLocalEmployees.List.Current.LocalEmployee.WorkPhone
```



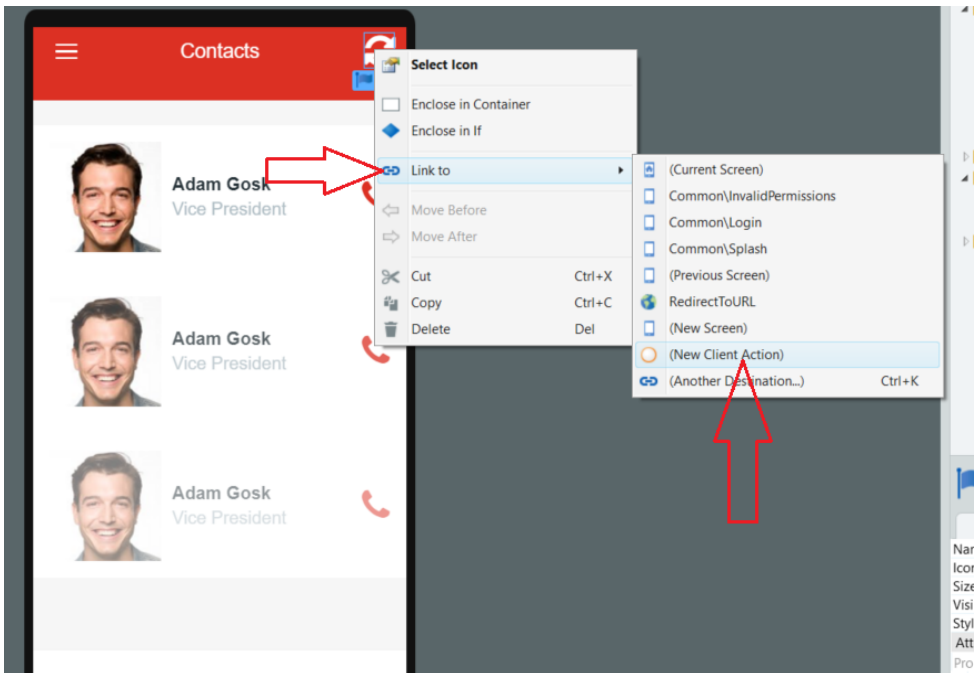
Drag an **Icon** from the left pain to the Action part of the Header and select the **Refresh** Icon.



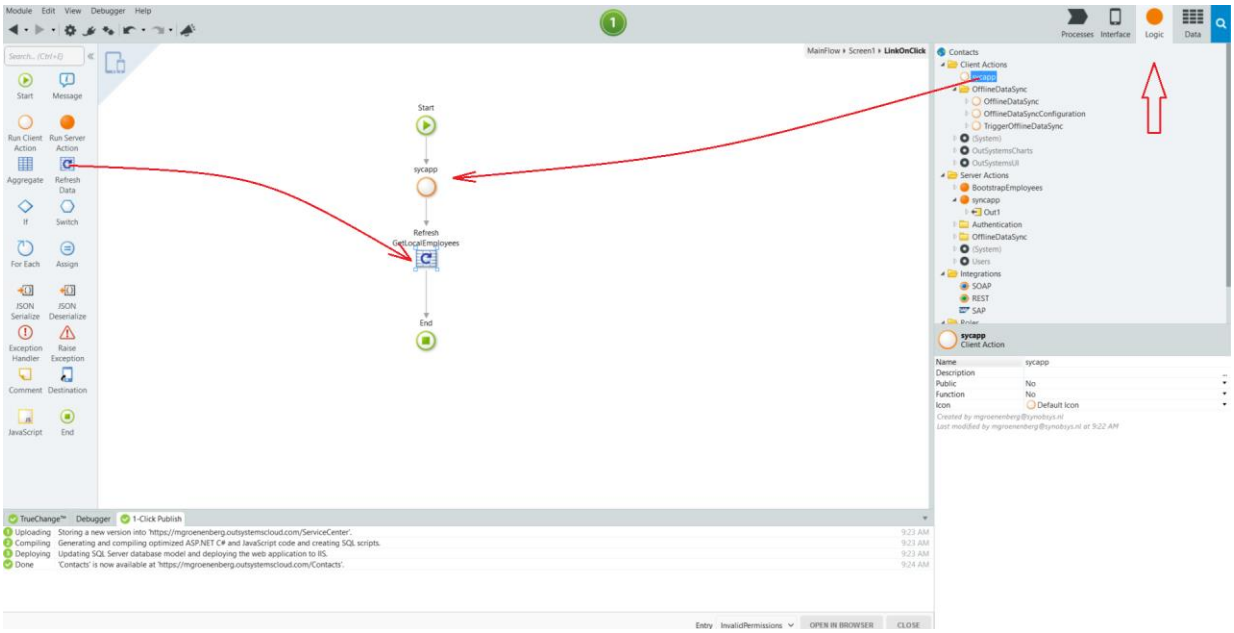
Right click on the new **Icon** and select **Link to** and then **New Client Action.**

A new action is created on your screen named `LinkonClick`**.**

Open this newly created Action and from the **Logic Tab** drag and drop the **Client Action** `syncapp` to the action line of this action.



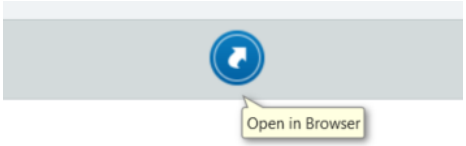
After the syncapp action we insert a **Refresh Table** by dragging and dropping it under the syncapp action and select the default table to refresh GetLocalEmployees



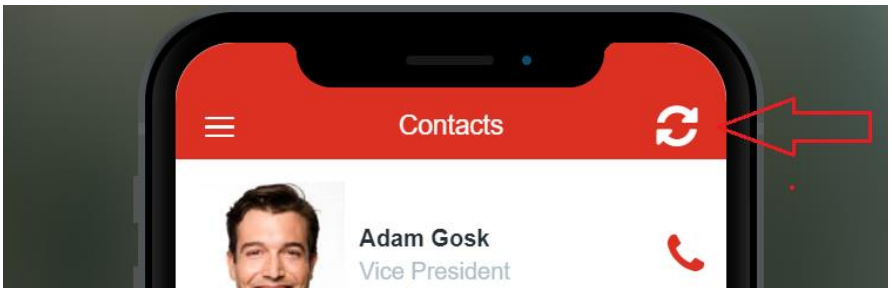
Now we have created a button that will sync the data from the cloud database to our local database and refreshes our screen.

You can test your screen now by pushing the 1 click publish button.

After the image becomes blue you can open it in a browser to test.



When it opens in the browser you can push your refresh button in the header to get your data on the screen.



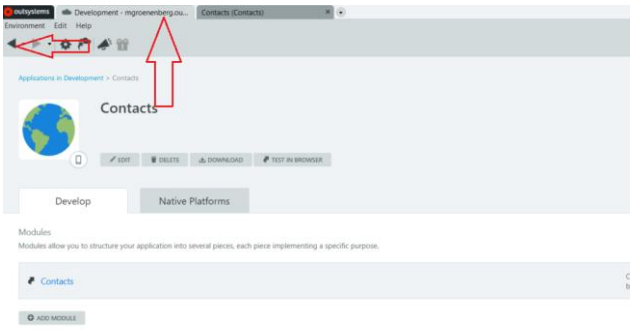
8. Integrating Plugin from the Forge.

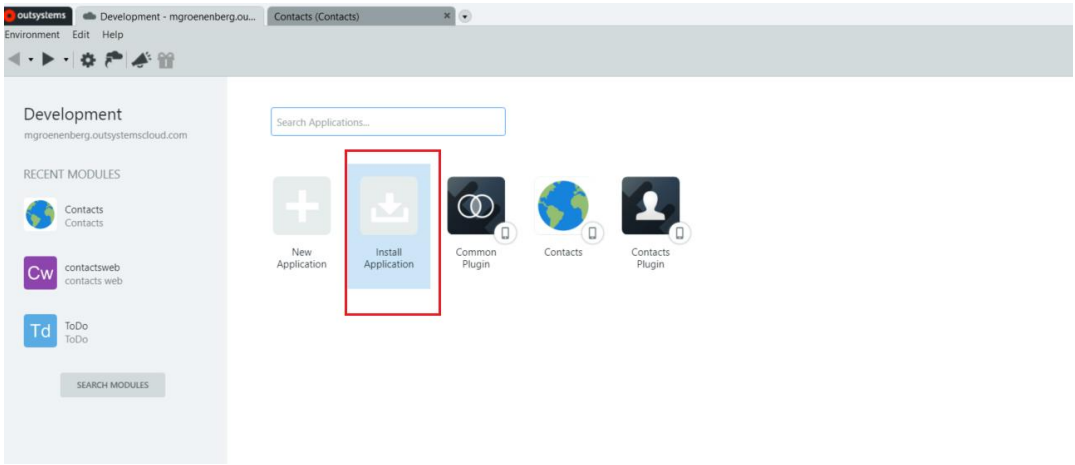
You can consume any Plugin which is available in the Forge.

As an example, we want to add the contact from our employee list to the contact list of our mobile device when we press on the picture.

To do that we need 2 plugins to be installed.

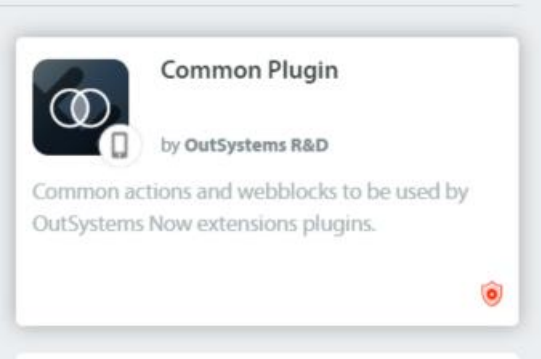
Go back to your personal development environment tab and select the button “back”.





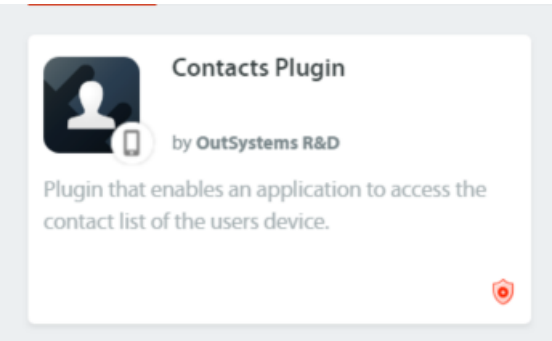
You will now be in your development home screen.

Click **Install Application** and browse the Forge for the **common plugin**.



Select and Install this plugin.

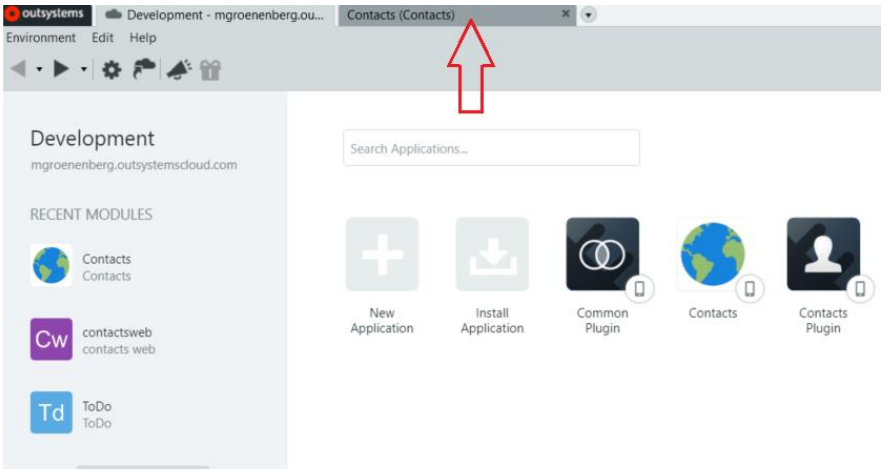
Next search for the Contact plugin.



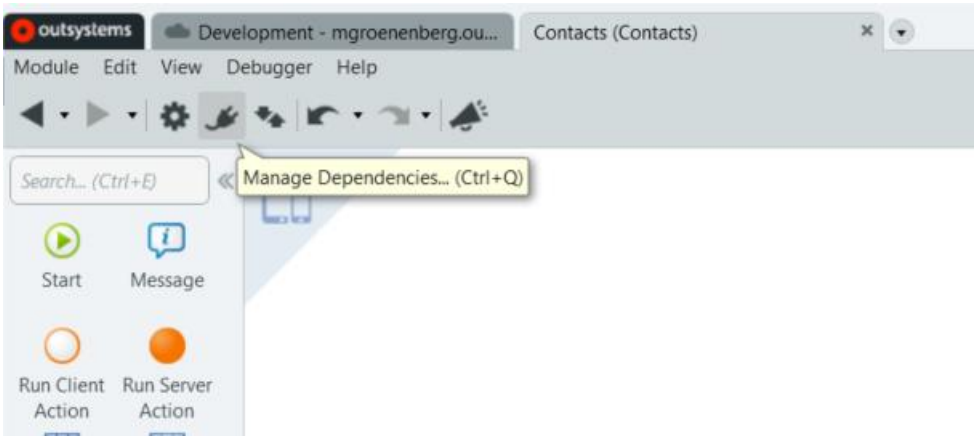
Select and Install this plugin.

Check on your **personal development** home page that the plugins are installed.

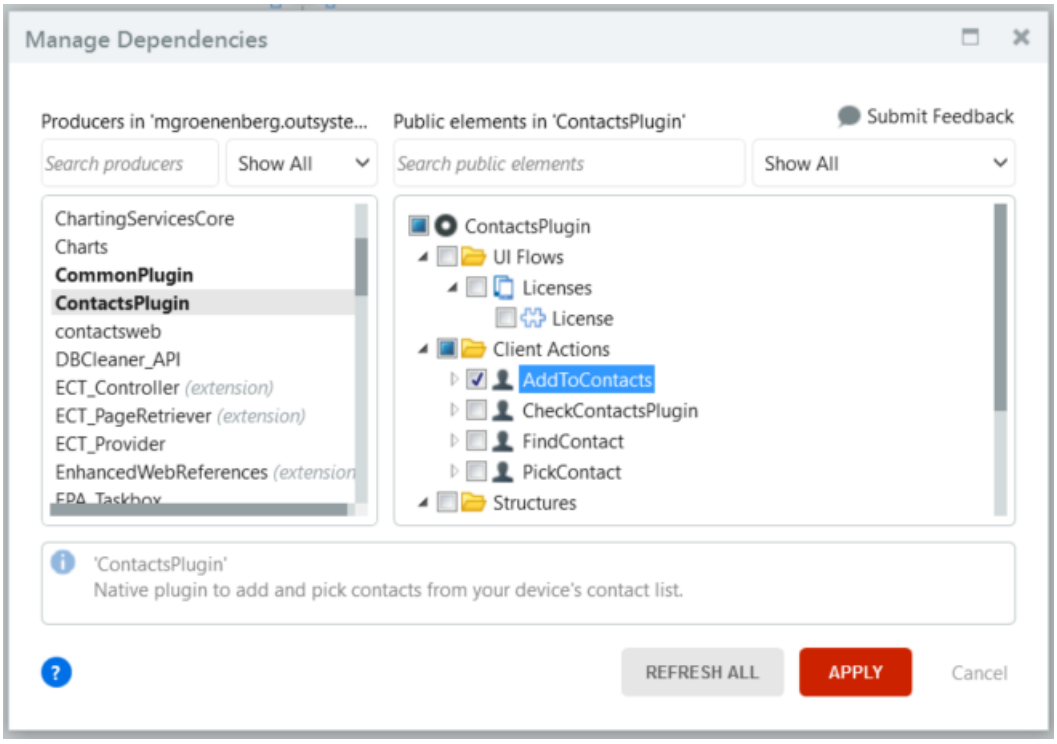
Go back to your **Contacts** Application by selecting the tab on top of your screen.



In your C**ontacts** application click the button **Manage Dependencies.**



In the popup search for the **ContactsPlugin** and select **AddToContacts**.

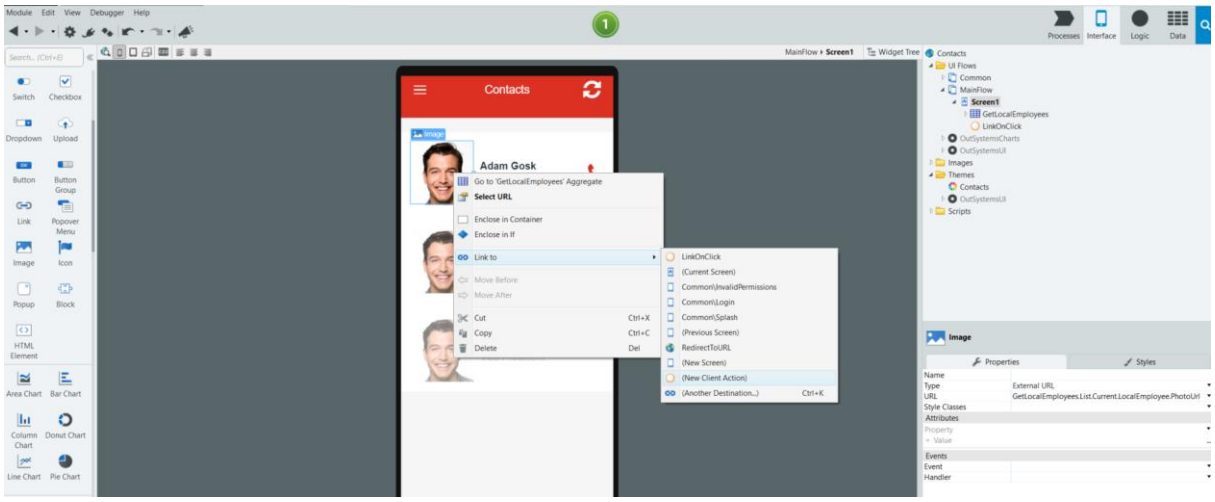


APPLY the changes.

Go back to your `screen1` and right click on the picture in your list.

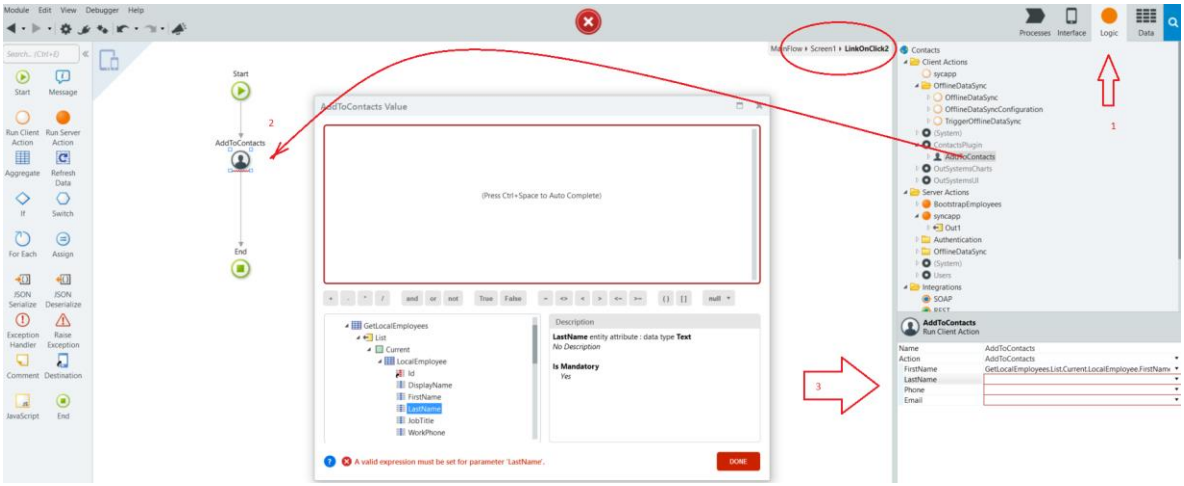
Select **Link to** and then **New Client Action.** A new client action `LinkOnClick2`

is created.

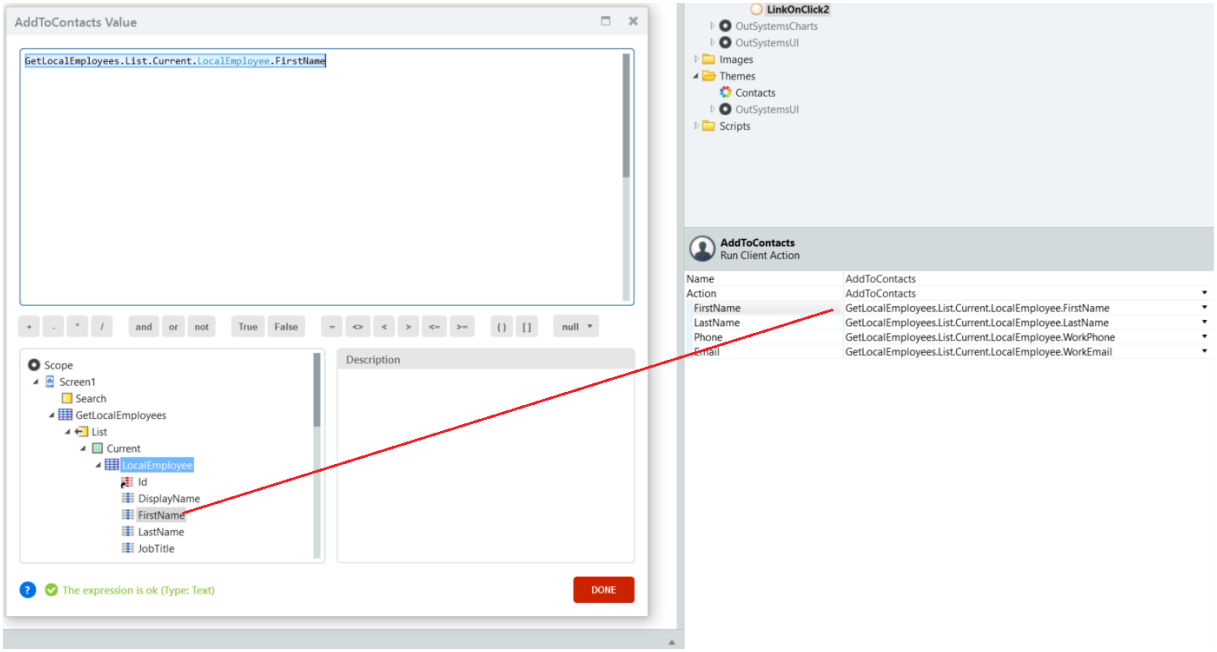


Open this action by double clicking it.

Open the **Logic tab** and drag and drop the action `AddToContacts` on the action diagram.



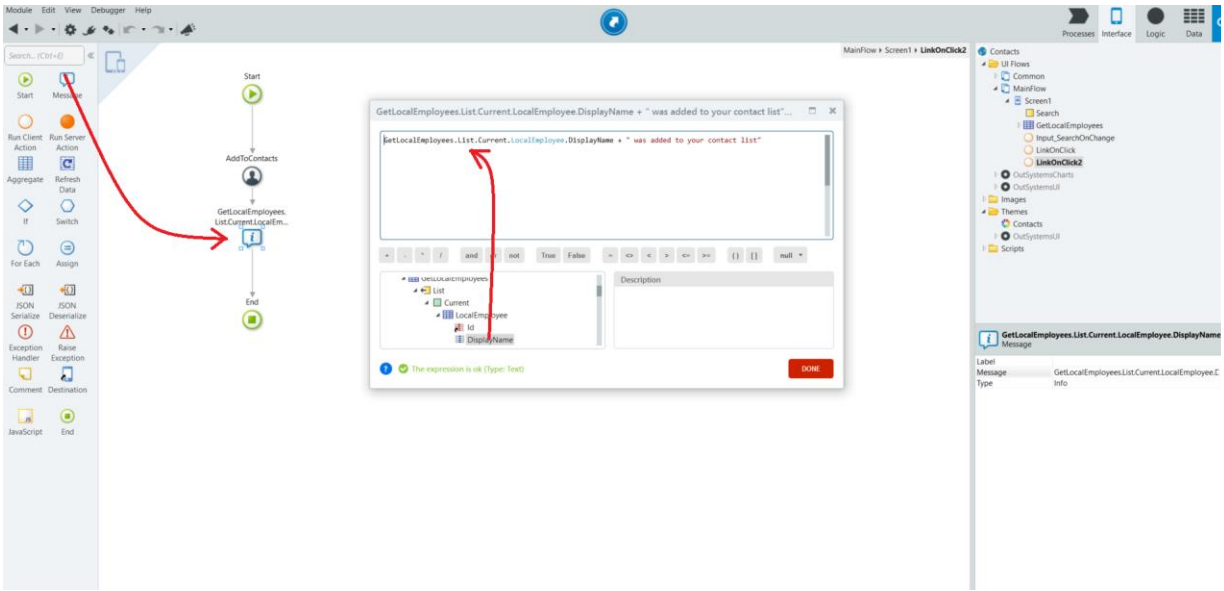
Add the 4 parameters that this action needs by using the **expression editor**.



Add a M**essage** from the left pane to the action to inform the user.

Edit the M**essage** text first select from the list the current `DisplayName` and add the text like:

GetLocalEmployees.List.Current.LocalEmployee.DisplayName + " was added to your contact list"



Now you are ready to test your application by publishing and after that start the browser.

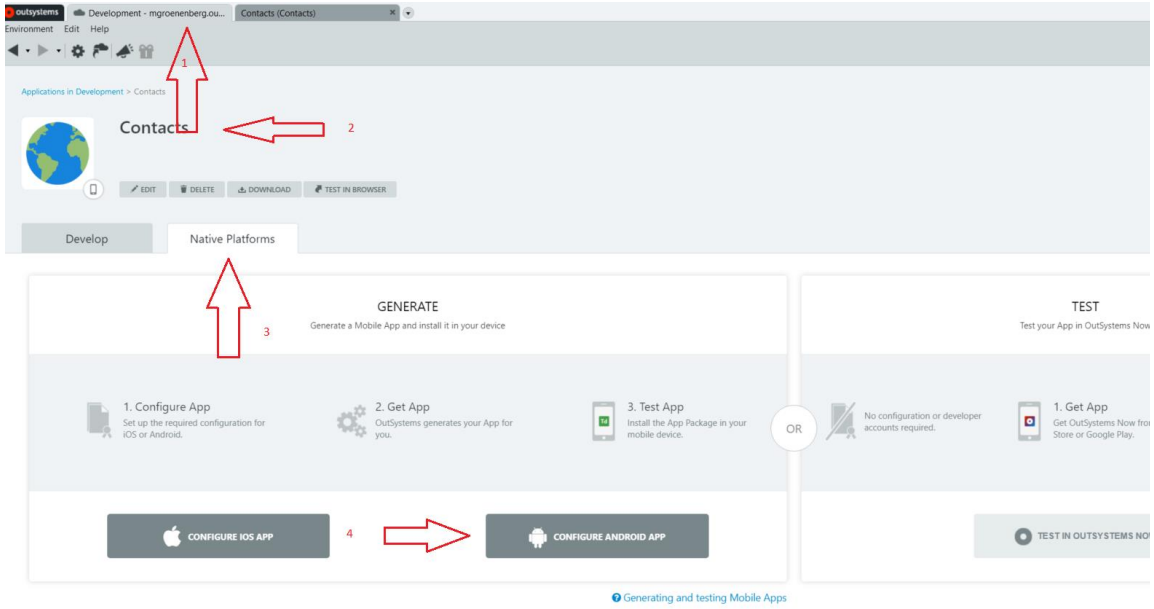
To prove it works we must bring it to your mobile device.

9. Run your application on your Mobile Device (Android)

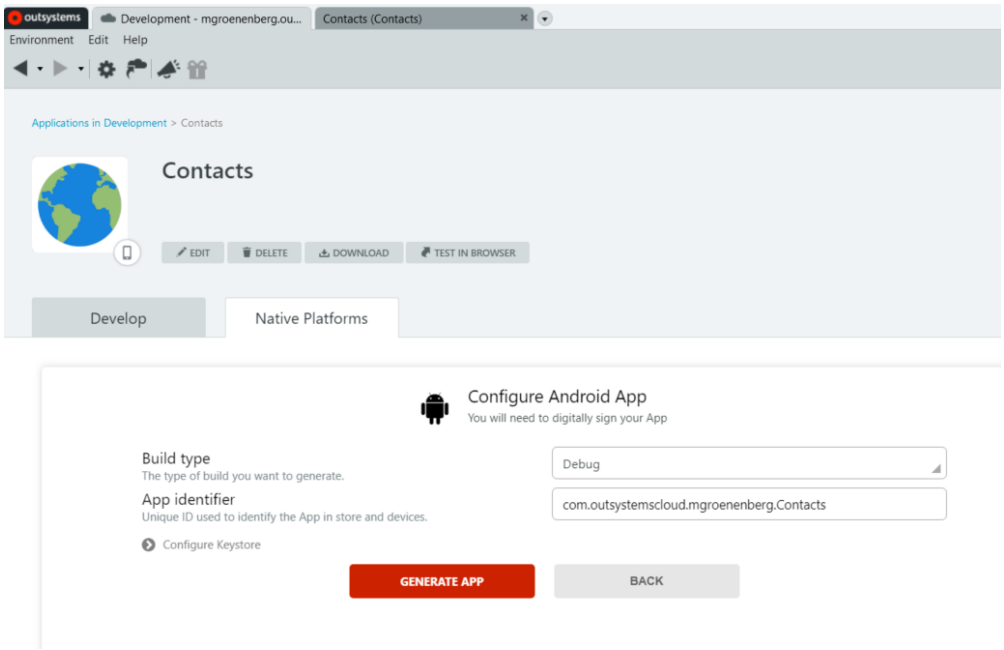
Open the **Development** home page by selecting this tab in top of your screen.

Select the **Contacts** application and the **Native Platform**.

Select **CONFIGURE ANDROID APP**



Select **GENERATE APP**:

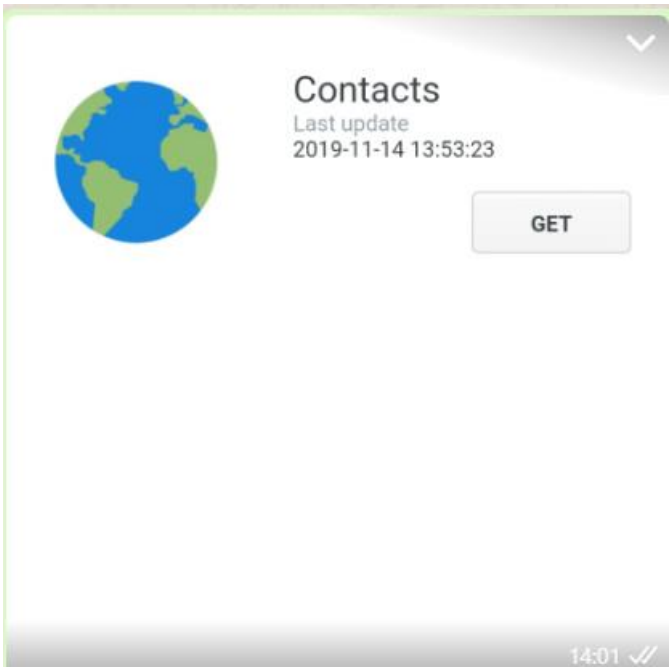


OutSytems will now generate the Android App.

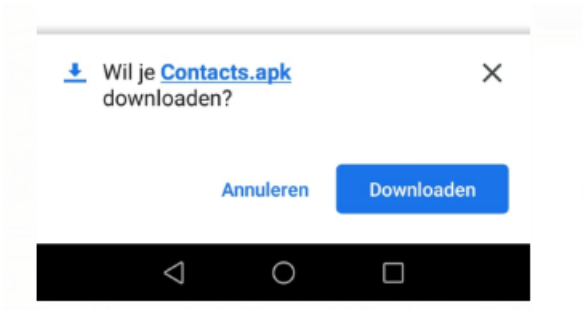
With a QR-code reader application (https://play.google.com/store/apps/details?id=net.grbot&hl=nl) you can with your mobile device directly open the website where your application is stored. Otherwise copy the created link and go there with your browser on your mobile device.

https://<yourname>.outsystemscloud.com/NativeAppBuilder/App?AppKey=49e13ff6-xxxx-xxxx-xxxx-xxxxxxxxxxxx

Open the website and GET the application



Download the application and Install the application.



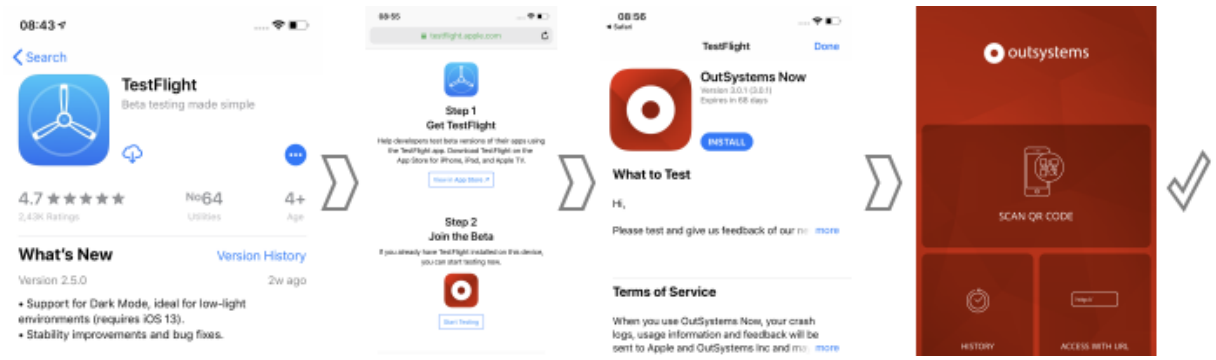
Accept all messages you get from your device.

Your application is now ready to test!

10. Run your application on your Mobile Device (iOS)

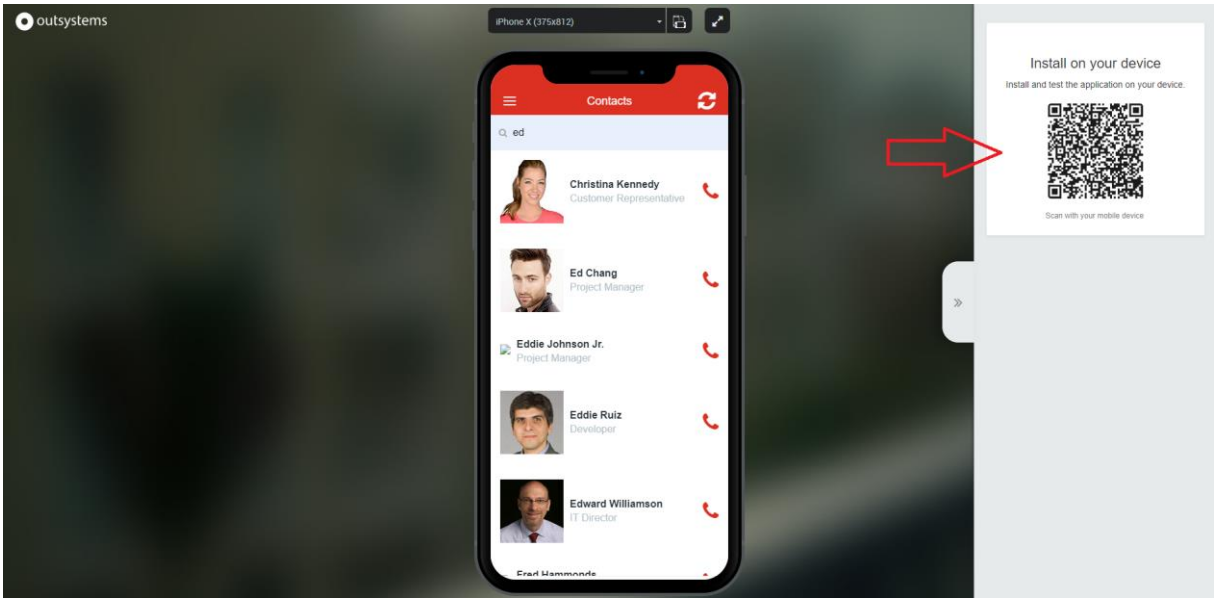
We are using Apple's test tool TestFlight by following these steps:

1. Install TestFlight on the iOS device that you'll use for testing.
2. Open Outsystems Now on your iOS device! (*send this link to your mobile device*)
3. Tap View in TestFlight or Start Testing.
4. Tap Install or Update for the app you want to test.



In Outsystems open the application `Contacts` with the buttons:

Scan with the **Outsystems Now** the QR-code of your browser screen.



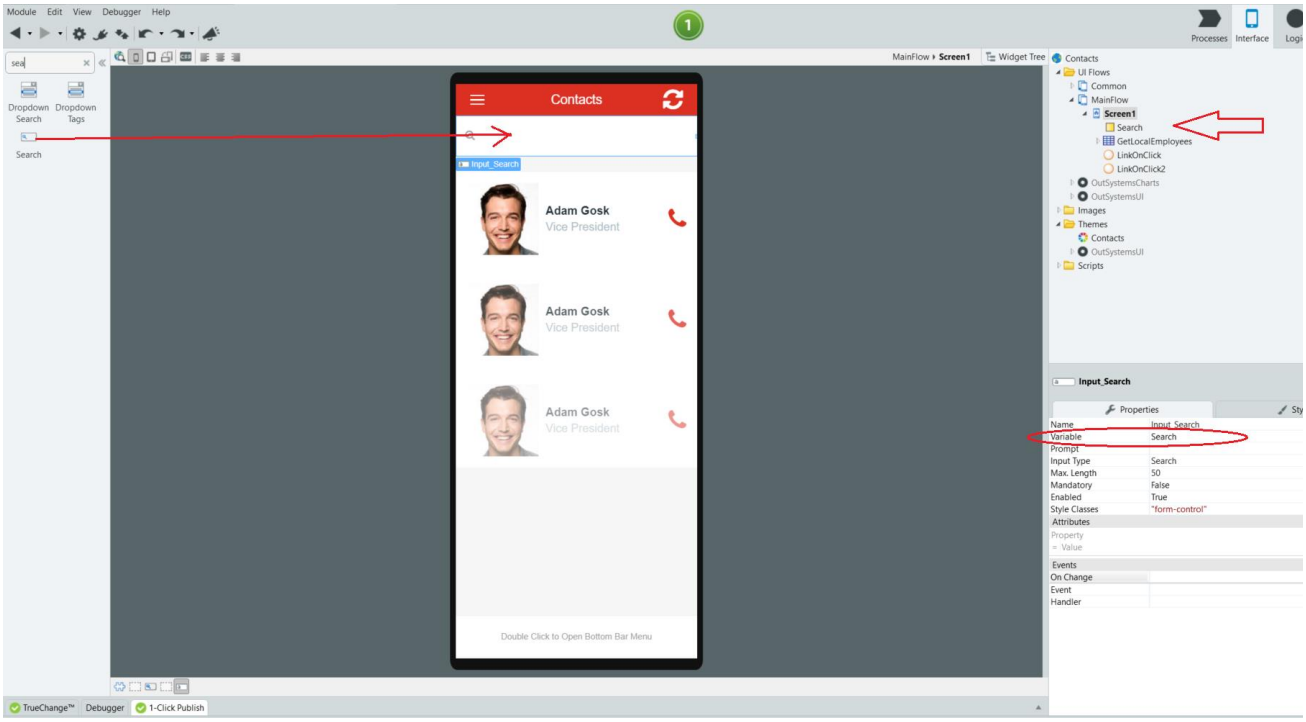
Your application is now ready to test!

11. Adding a search on the Contacts application

Go to the **Interface tab** of the **Contacts** Application. Select `Screen1` , right click and select the **Add Local Variable**. Name it `Search`.

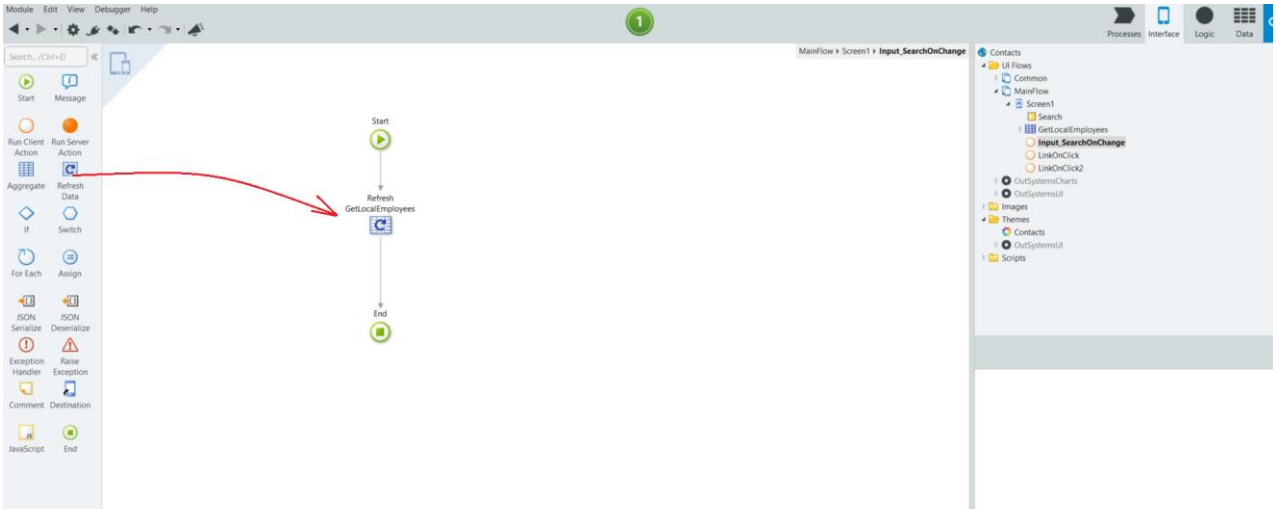
From the left pane search for the **Search** widget and drag it to the Header of your application.

Add the **Local Variable** `Search` as input **variable** of the **Input_Search** widget.



On the **Input_Search** widget double click on the **Events On Change** and make a N**ew Client Action.** Outsystems named it `Input_SearchOnChange`.

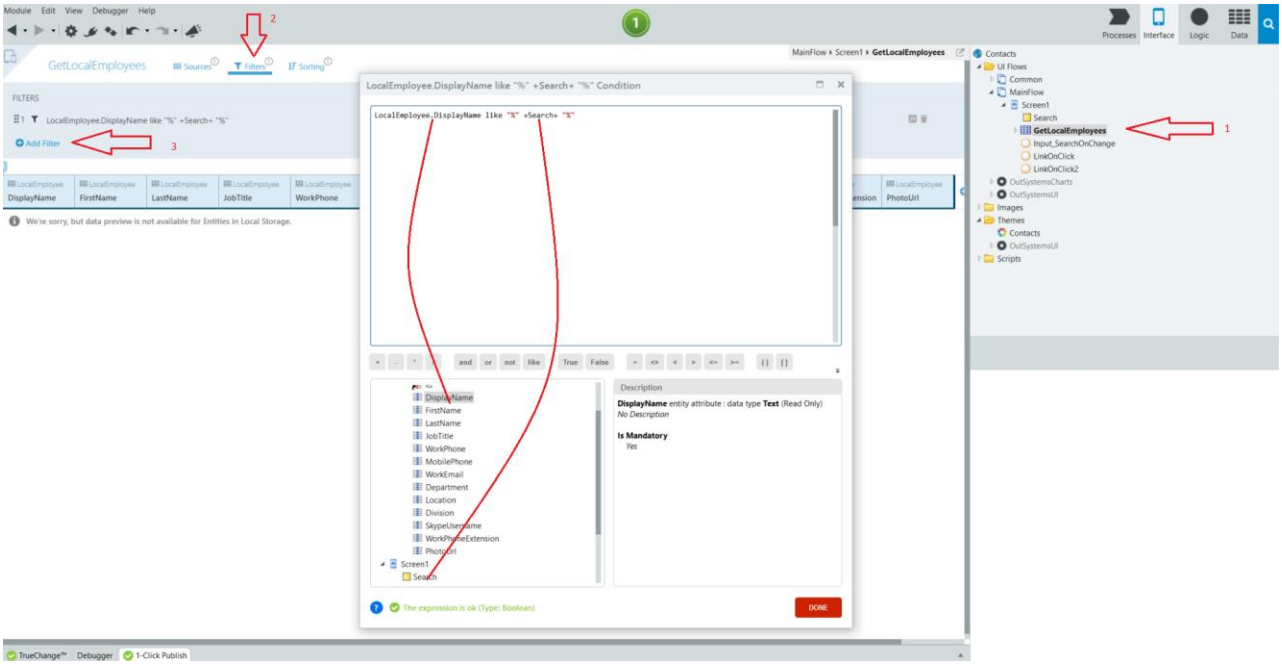
Open this action and drag and drop the **Refresh Data** icon from the left pane on your `Input_SearchOnChange` action.



Change the Aggregate `GetLocalEmployees` (double click) select **Filter** and **add a Filter**.

Select the `LocalEmployee.DisplayName` select `LIKE` and input `"%" +` select the local variable `search` and input a `+ "%".`

So it looks like: `LocalEmployee.DisplayName like "%" +Search+ "%"`



Now you are ready to test your application again by publishing and after that start the browser.

Check your application on your mobile device. Notice that this new version is automatic updated the old application and there is no need to reinstall your application.