

Create the Employee Directory App

Table of Contents

Outline.....	2
Scenario	2
Data Model	2
Employee List Screen	3
Employee Detail Screen	4
How-To.....	6
Resources	6
Getting Started	6
Import Data from an Excel File	11
Create Relationships Between the Data	14
Publish the Application and View Data	21
Using Accelerators to Create Screens	23
Wrapping up.....	28
References	28

Outline

In this tutorial, you will build an Employee Directory application from scratch using Service Studio, the OutSystems IDE.

You will start by creating the **Employee Directory** application, define its data model and then create two Screens: one to list employees and one to see and edit the details of an employee (don't worry, it is easier than it sounds).

In the end, you will be able to open your app in the browser and try it out.

Scenario

We want to build a web application to manage the employees of a company. In this tutorial, you will create the Employee Directory application and its first two Screens:

- Employee List: lists all the employees and their most important information.
- Employee Detail: has a Form that allows viewing and editing the detailed information of an employee, including the name, email, hiring date, job position, among others.

You will start from an empty app, with no data or Screens. The first step will then be creating the employee data, from an Excel file.

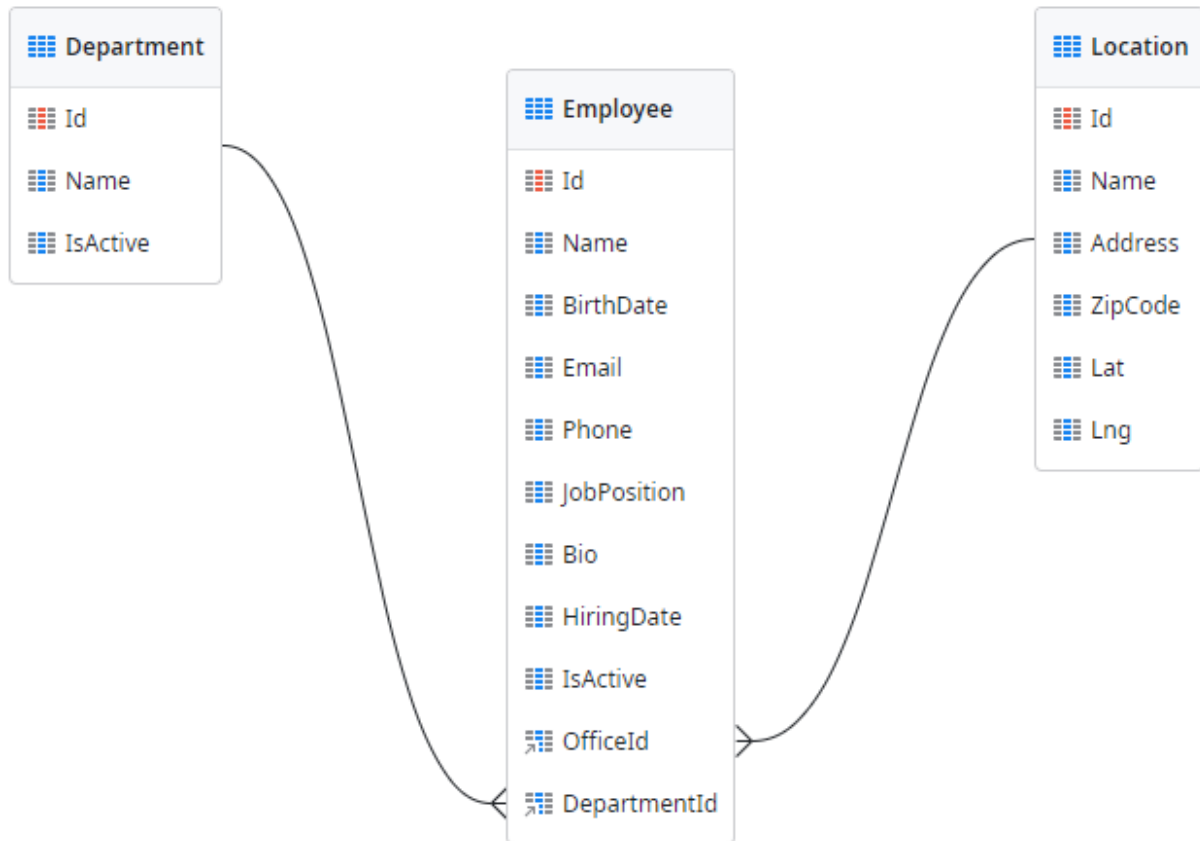
Data Model

This application will have a data model with three database tables, which in OutSystems are called Database Entities:

- Employee, with the employee information;
- Location, with the office information;
- Department, with the department information.

Besides the employees' basic information, such as name, birth date, email, phone, and job position, an employee will also have an office and a department associated with it,

creating then a relationship between the Employee and the Location and Department Entities.



Employee List Screen

The Employee List Screen displays a list of all the employees in a tabular layout, with the name, birth date, email, and phone information.

The Screen will also have a text input field to filter the employees, by name, right next to a button that will allow the user to add a new employee.

The Screen will look like the following image:

EmployeeDirectory Employees Login

Employee List

Search Add Employee +

Name	Birth Date	Email	Phone
Patricia Wesley	25 Dec 1986	patricia.wesley@example.com	1-555-723-3191
Edward Williams	9 Oct 1980	edward.williams@example.com	1-555-491-7977
Andrea McCarthy	14 Dec 1986	andrea.mccarthy@example.com	1-555-445-1521
Ann Olivarria	16 Aug 1978	ann.olivarria@example.com	1-555-720-9353
Bridget Hernandez	10 Nov 1982	bridget.hernandez@example.com	1-555-843-3944
Carla Hansen	30 Dec 1986	carla.hansen@example.com	1-555-228-7916
Charlotte Anderson	15 Sep 1982	charlotte.anderson@example.com	1-555-788-4083
Cheryl Fleet	10 Apr 1980	cheryl.fleet@example.com	1-555-253-1007
Christina Sharp	20 Jul 1980	christina.sharp@example.com	1-555-234-8671
Christopher Shaw	5 Oct 1991	christopher.shaw@example.com	1-555-895-9275

1 to 10 of 55 items < 1 2 3 4 ... 6 >

Employee Detail Screen

The Employee Detail Screen displays all the fields of a specific employee, such as the name, birth date, email, hiring date, and department.

A user of this application can navigate to this Screen in two ways:

- when clicking on the name of the employee in the Employee List Screen.
- when clicking on the option to add a new employee.

As mentioned before, this Screen will be used to view the details of the employee, but will also allow a user to modify and save that data. For that purpose, the Screen should also have a Save button that executes the logic to save the employee information in the database. Right next to it, we will also have a Back button to return to the Employee List Screen.

The Screen will look like the following image:

EmployeeDirectory Employees Login

Edit Employee

Name *
Patricia Wesley

Birth Date *
12/25/1986

Email *
patricia.wesley@example.com

Phone *
1-555-723-3191

Job Position *
Sales Manager West

Bio
Top-ranked sales manager, contributed to record sales and new account developpr

Hiring Date *
03/31/2020

Is Active
☒

Department
Accounting

Location
Boston, MA

Back Save

How-To

Now that you know the scenario, let's build the Employee Directory app by following this how-to guide! Are you ready? Let's do it together!

Resources

This how-to guide has two resources that you will need:

- The **Employee Directory.oap** file, which is the quickstart for the application that you will create. Don't worry, we just created the empty app to get you start quickly, but didn't create any functionality yet.
- The **Bootstrap.xls** excel file with some data about employees, locations, and departments. Feel free to add more data if you want to, or even create your Excel file with your data. However, make sure you keep the same columns and column headers, at least for now, so you don't get a mismatch between your data and this guide.

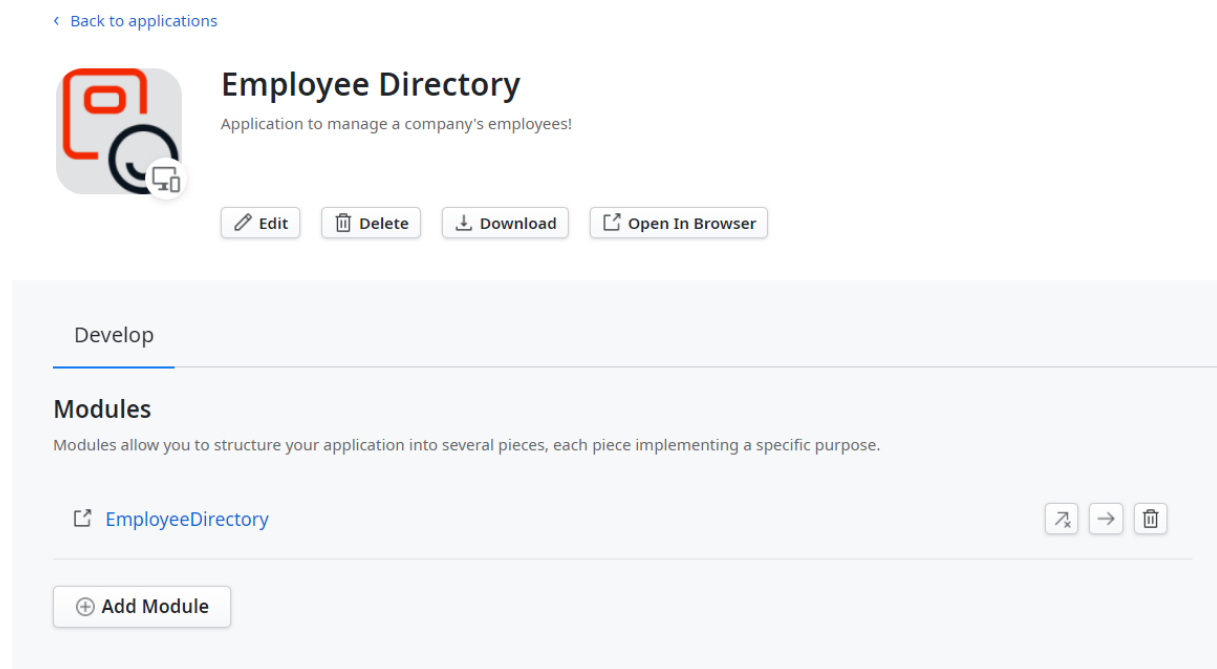
Please download them from the **Lesson materials** on the course page.

Getting Started

Let's start this tutorial by installing the Quickstart file, **Employee Directory.oap**.

This file has an application with one module, named **EmployeeDirectory**, one icon and a small description.

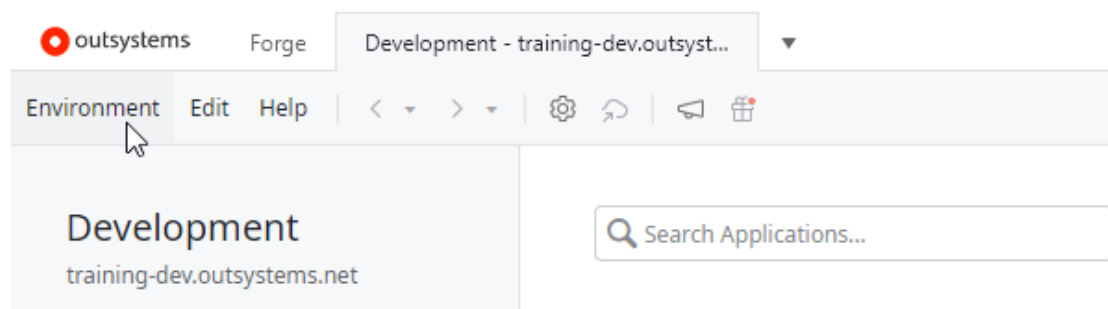
The module is pretty much empty at this point and it's where you will create the Screens and logic of the Employee Directory web application.



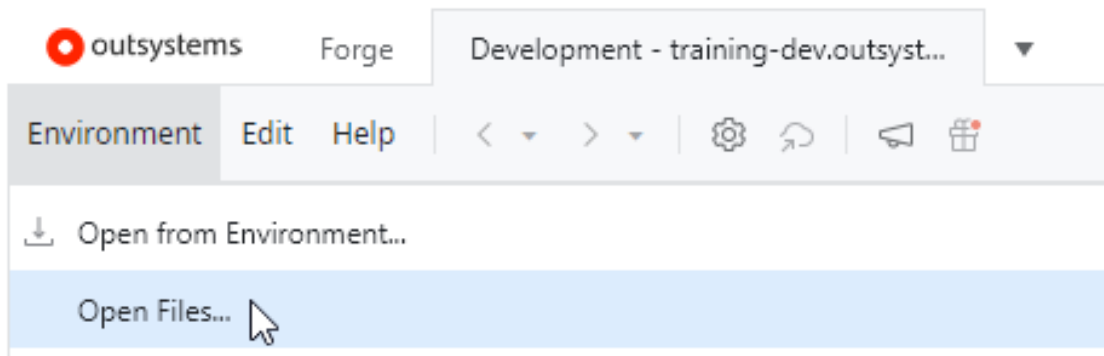
So, install Service Studio if you haven't done it so far, connect to your OutSystems account (in Service Studio) and download the exercise resources from the **Lesson Materials**.

You are now ready to start!

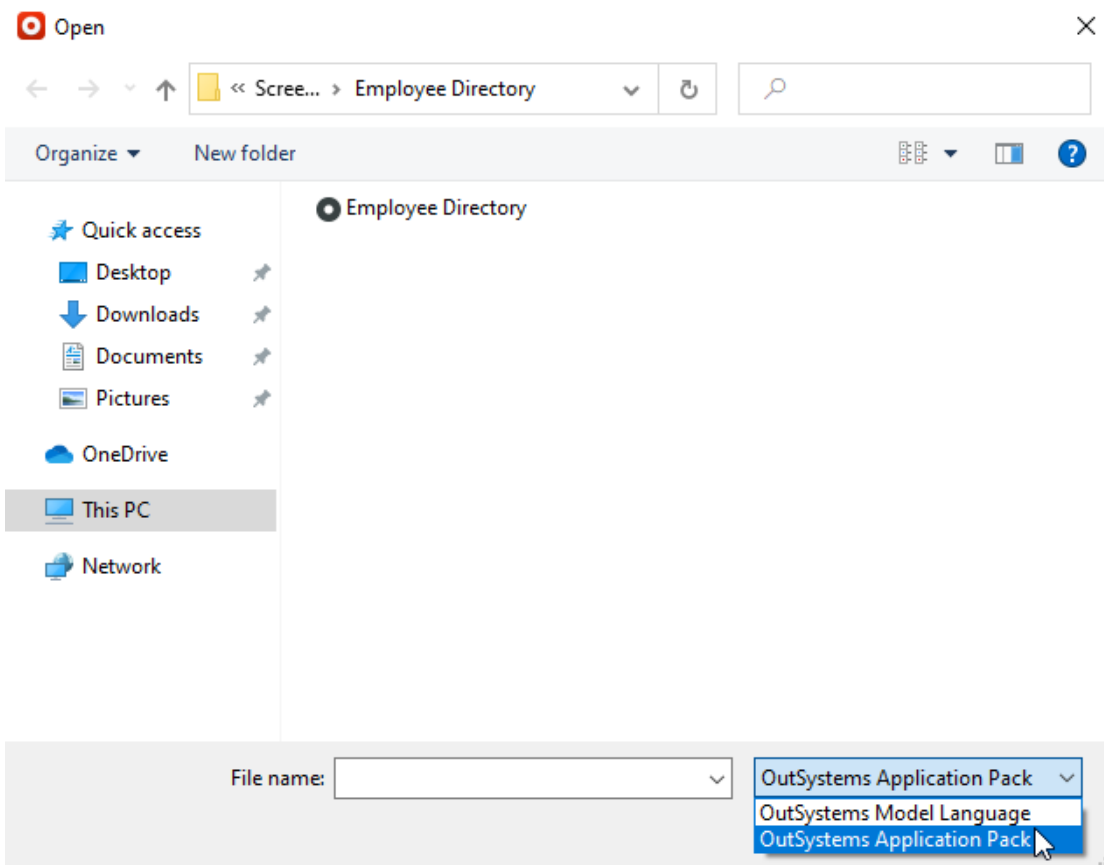
- 1) In Service Studio's main window, select the **Environment** menu on the top left of your window.



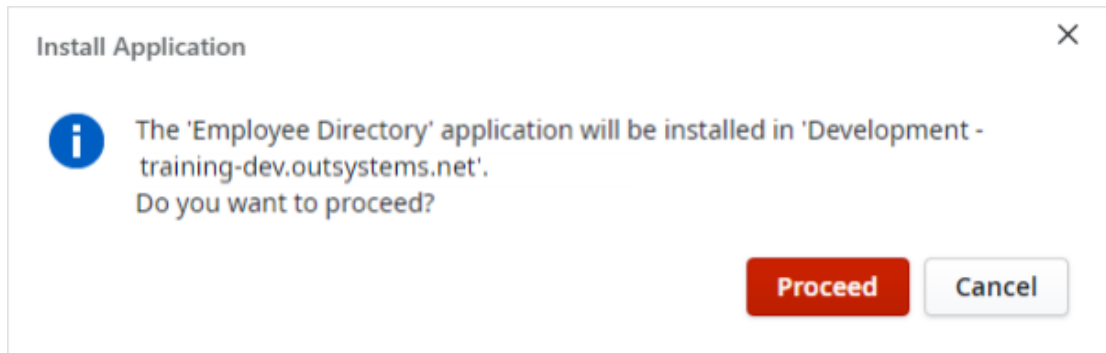
2) Select **Open Files....**



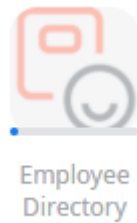
3) In the new dialog, navigate to the folder where you have the resource file and change the file type to OutSystems Application Pack (.oap). You should now see the file **Employee Directory.oap**. Double-click the file to open it.



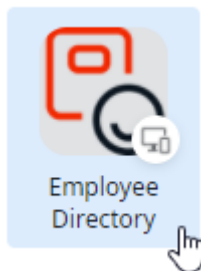
- 4) In the new confirmation dialog that appears in Service Studio, select **Proceed**.



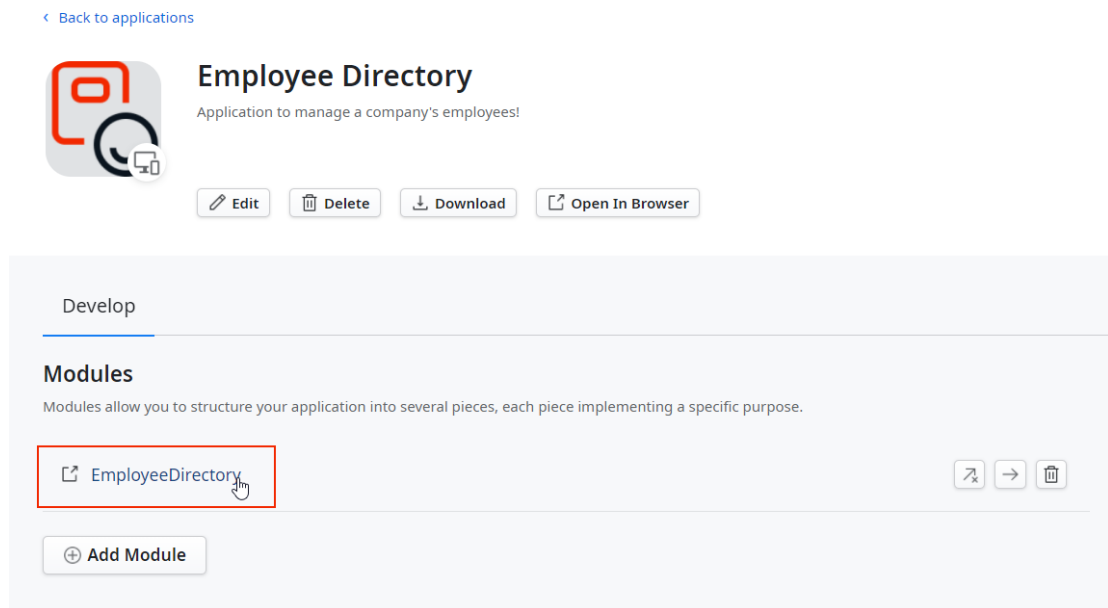
- 5) The application will begin installing automatically. Give it some time until it's finished.



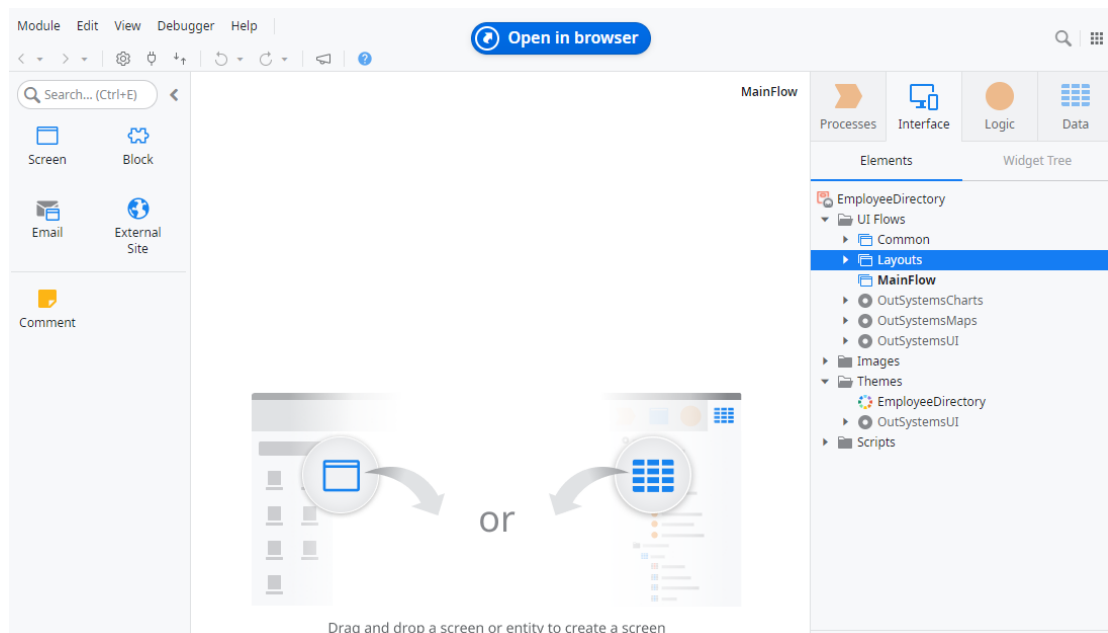
- 6) When the progress bar finishes, click on the application icon to open the application in Service Studio.



- 7) The application has only one module. Let's open it by clicking on the module's name!



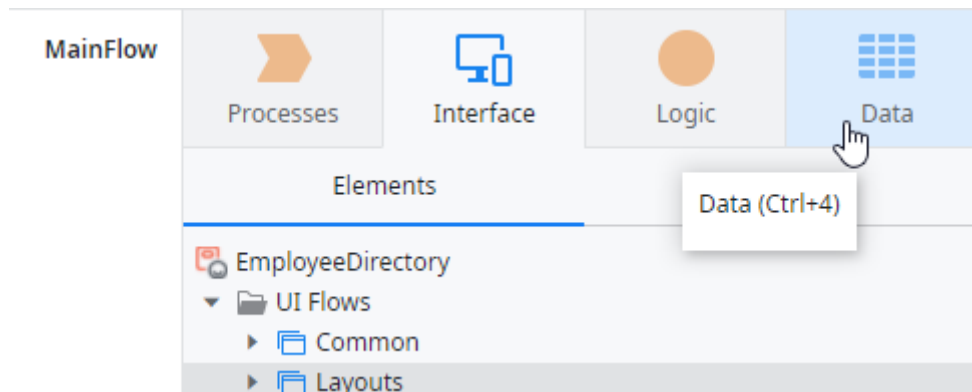
- 8) You are now inside the module. An application is a set of modules and it's within the modules where you develop the elements from your applications, from UI and business logic to data.



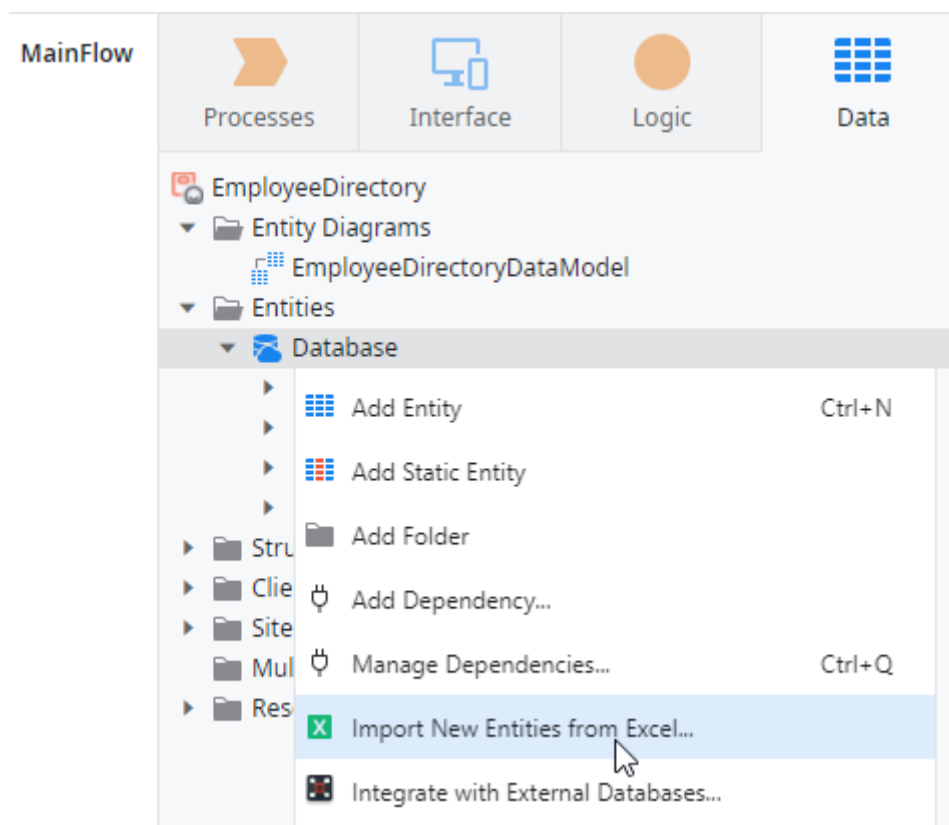
Import Data from an Excel File

Now, that you have the app installed, let's start by adding some data. In this section, you will import an Excel file with data for employees, locations and departments, which will automatically create three Database Entities.

- 1) In Service Studio, select the Data tab in the top-right corner. This will open an area of Service Studio where we can see and define the data for our app.



- 2) Expand the Entities folder, right-click the Database element and select **Import New entities from Excel...**



- 3) In the new dialog that appears, navigate to the folder where you have the **Bootstrap.xlsx** and select it.
- 4) A dialog will appear showing the name of the Entities that will be created. Make sure all three entities appear here: Employee, Location, and Department. Click **Import**.

Import New Entities from Excel



Do you want to import the following Entities from 'Bootstrap.xlsx' Excel file?

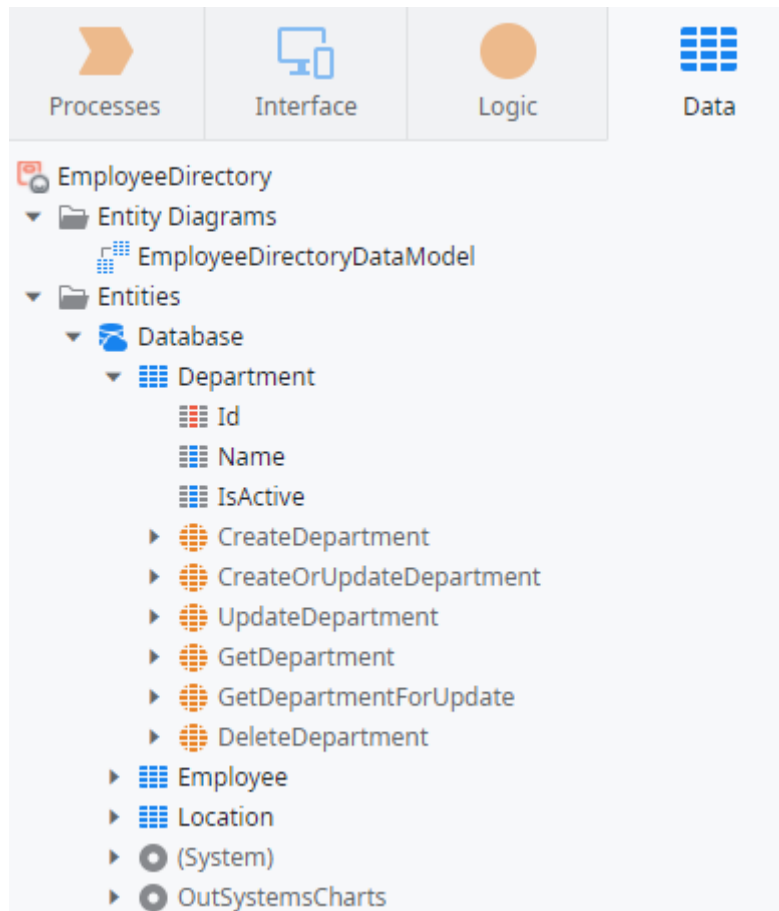
- Employee
- Location
- Department

Import

Cancel

OutSystems analyzes the Excel file and creates one Entity per sheet in the Excel file, using the name of the sheet.

The three entities will be created in the Database, and you can expand each one to see that the attributes were also automatically created.



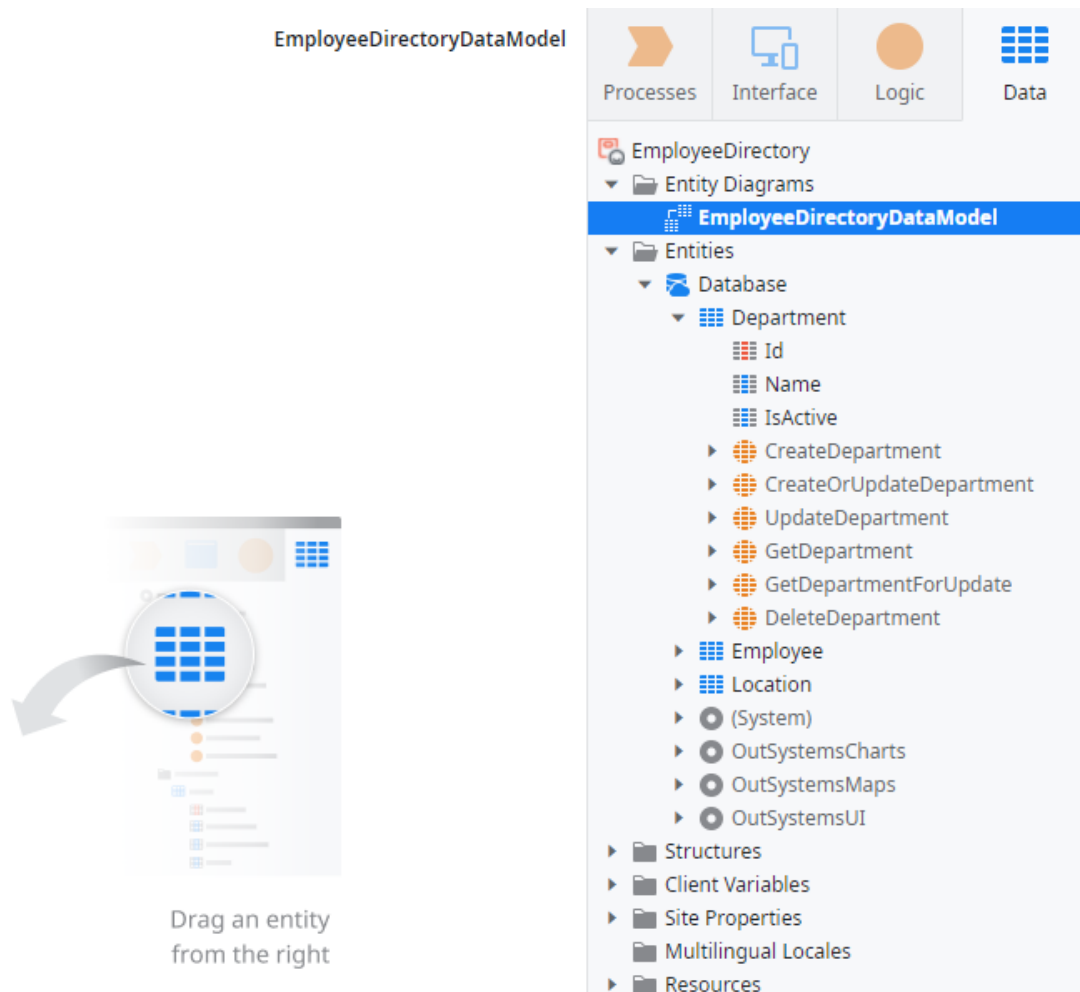
An attribute refers to a column in the Excel file. OutSystems goes through each column, analyzes the content in each row, and creates an attribute with the name of the column and with the appropriate data type.

This is just a way to import data to OutSystems. You can create your data model from scratch, or even import data from other data sources. You can also edit, or create your own, Excel files. Bear in mind that the sheet and column names will also be the names of your Entities and attributes.

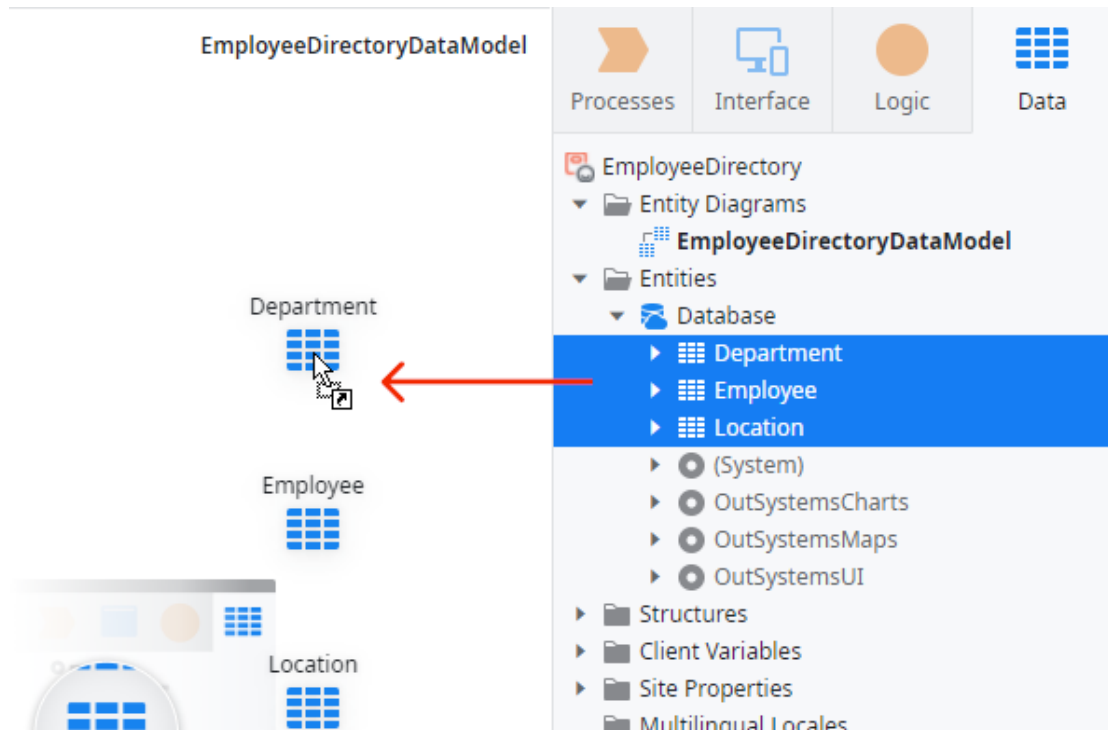
Create Relationships Between the Data

Now that you have created the Entities, it's time to create relationships between the data. In this application, an Employee belongs to one Department and works at an office (in a given Location).

- 1) Still in the Data tab, expand the **Entity Diagrams** folder (if it is not expanded already). Double-click the **EmployeeDirectoryDataModel** to open it.



- 2) Select the **Department**, **Employee** and **Location** Entities (you can use CTRL+click to select several elements). Drag the selected Entities and drop them to the data model you have just opened.



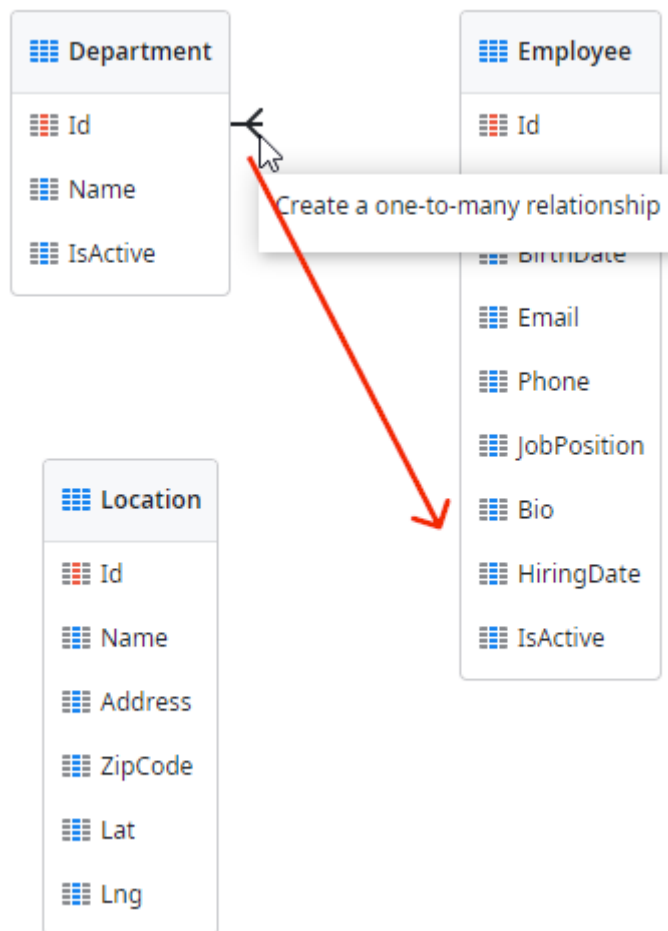
You will have something like the image below.

Department
Id
Name
IsActive

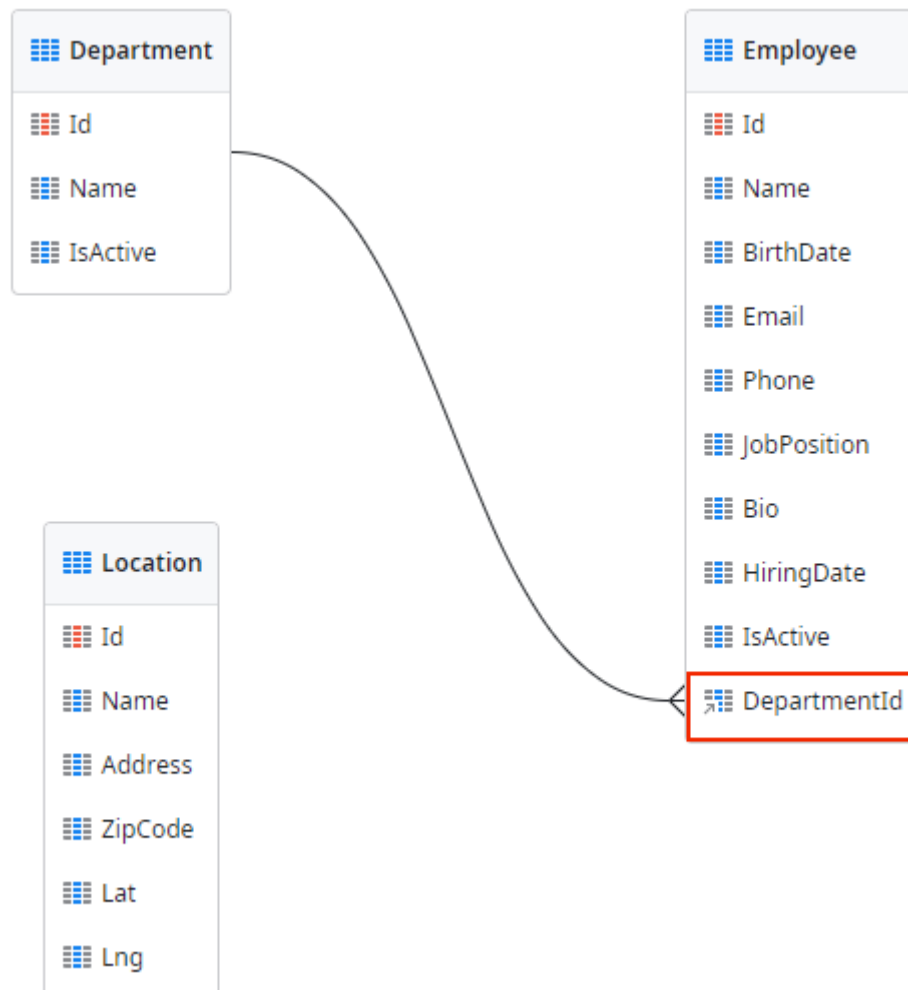
Location
Id
Name
Address
ZipCode
Lat
Lng

Employee
Id
Name
BirthDate
Email
Phone
JobPosition
Bio
HiringDate
IsActive

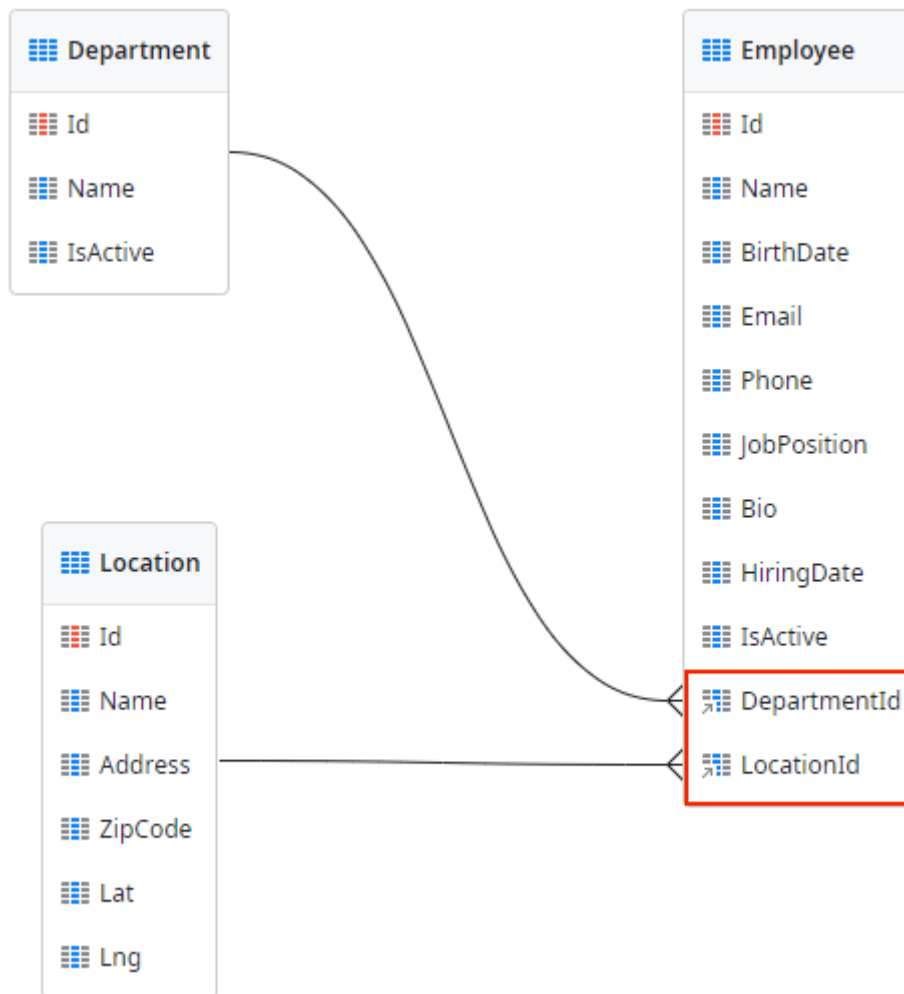
- 3) Hover the mouse in the Department Entity, select the connector that appears close to the Id attribute, and drag it to the Employee Entity.



A new attribute was created in the Employee entity called *DepartmentId*. This creates a one-to-many relationship, meaning that a Department has many employees and an employee belongs to one Department.



- 4) Repeat the same step for the **Location** Entity. Don't forget to drag **from** the Location Entity **to** the Employee.



Notice that the two attributes created automatically in the Employee Entity have a slightly different icon. This happens because they are attributes linked to another entity. If you click on the attribute you can see they have special identifiers.

- 5) We can at any time change the names of the attributes. Let's focus again on the right sidebar of the Data tab, expand the Employee Entity and select the **LocationId** attribute. Set its name to *OfficeId*.

The screenshot shows the OutSystems Data tab interface. At the top, there are four tabs: Processes, Interface, Logic, and Data. The Data tab is selected. Below the tabs, a tree view shows the project structure: EmployeeDirectory > Entity Diagrams > EmployeeDirectoryDataModel > Entities > Database > Employee. The Employee entity is expanded, showing its attributes: Id, Name, BirthDate, Email, Phone, JobPosition, Bio, HiringDate, IsActive, DepartmentId, and OfficeId. The OfficeId attribute is highlighted with a red box. Below the tree view, the configuration for the OfficeId attribute is shown. The attribute is named 'OfficeId' and is an 'Entity Attribute'. The configuration table below shows the following values:

Property	Value
Name	OfficeId
Description	
Label	Office
Data Type	Location Identifier
Is Mandatory	No
Delete Rule	Protect

Publish the Application and View Data

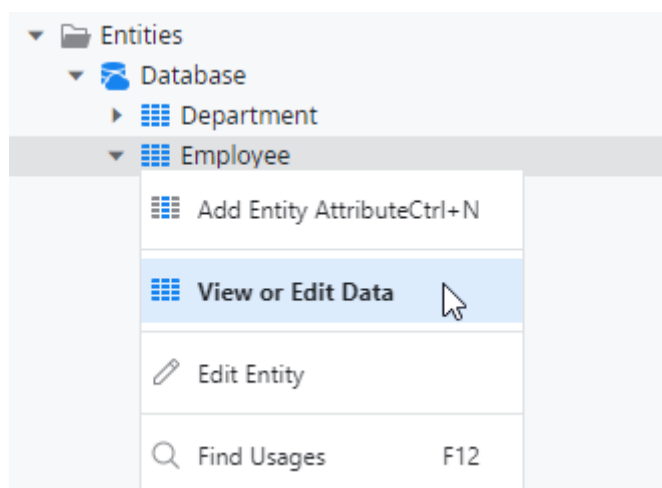
The data model is now ready, so now we need to publish this application in the OutSystems server to actually save the data in the database, and to be able to view it in our application.

- 1) Publish the module by clicking on the Publish button.



The publishing process will upload your module to the server, compile and generate the code, update the database and deploy it. When the process is over, the app is ready to be accessed by your users.

- 2) It's time to see our data. In the Data tab, right-click the Employee Entity and select **View or Edit Data**.



- 3) You will see a table with all the Employee attributes filled out with the data we imported from the Excel file. Move the bar to the right to see the other attributes.

No Filters

1 Sorting

<div><div></div><div>Employee</div></div> <div><div></div><div>Id</div></div>	<div><div></div><div>Employee</div></div> <div><div></div><div>Name</div></div>	<div><div></div><div>Employee</div></div> <div><div></div><div>BirthDate</div></div>	<div><div></div><div>Employee</div></div> <div><div></div><div>Email</div></div>	<div><div></div><div>Employee</div></div> <div><div></div><div>Phone</div></div>	<div><div></div><div>Employee</div></div> <div><div></div><div>JobPosition</div></div>
55	Jerome Wilde	1977-06-29	jerome.wilde@example.com	1-555-360-5047	UI Designer
54	Janice Spacey	1985-03-30	janice.spacey@example.com	1-555-943-2061	Development Manager
53	Jacob Norton	1984-04-03	jacob.norton@example.com	1-555-330-181	Security Officer
52	Justin Carter	1991-08-07	justin.carter@example.com	1-555-316-3103	Services Representative Consultant
51	Hannah Bridges	1987-08-27	hannah.bridges@example.com	1-555-696-141	Front-End Developer
50	Harrison Fitzgerald	1984-09-25	harrison.fitzgerald@example.com	1-555-398-7940	Full-Stack Developer
49	Hamilton Jones	1991-08-09	hamilton.jones@example.com	1-555-879-4586	Software Tester
48	Haley Scott	1981-03-15	heley.scott@example.com	1-555-672-9322	Human Resources Consultant East
47	George Ramos	1989-04-19	george.ramos@example.com	1-555-349-4530	VP Personal Assistant
46	Emily Sutton	1979-06-02	emily.sutton@example.com	1-555-519-9228	Lead Software Tester
45	Emily Banks	1988-12-05	emily.banks@example.com	1-555-247-2664	Sales Representative
44	Carl Vedder	1979-04-27	carl.vedder@example.com	1-555-409-3087	Lead Developer
43	Casey O'Donnel	1992-04-12	casey.odonnel@example.com	1-555-345-8539	UI Designer
42	Charles Ament	1986-02-26	charles.ament@example.com	1-555-762-8771	Back-End Developer
41	Camila Stevenson	1987-01-23	camila.stevenson@example.com	1-555-323-5801	Software Tester

- 4) The *DepartmentId* and *OfficeId* columns are empty, since we have just created these attributes. But you can use this window to add some data. Click on a dropdown in a row of one of these columns, to see the Department and Location options. Populate some employee data with some departments and offices.

Employee HiringDate	Employee IsActive	Employee DepartmentId	Employee OfficeId
2020-03-31	true	0	0
2020-03-31	true		0
2020-03-31	true	1 Human Res	0
2020-03-31	true	2 Services	
2020-03-31	true	3 Services Su...	1 Human Resources
2020-03-31	true	4 Services Su...	
2020-03-31	true	5 Credit Cont...	0
2020-03-31	true	6 Accounting	
2020-03-31	true	7 Management	0
2020-03-31	true	8 Marketing	

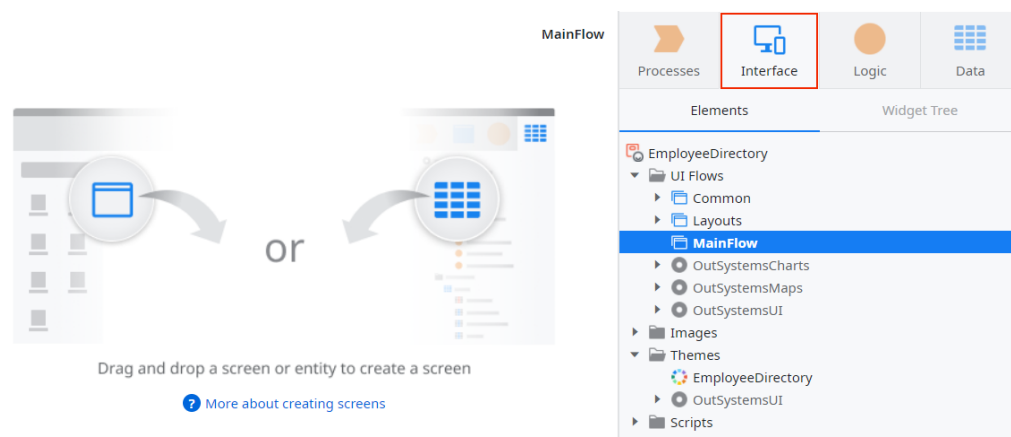
- 5) In the end, you should have something like the picture below. Click on **Apply** to save your changes.

Employee HiringDate	Employee IsActive	Employee DepartmentId	Employee OfficeId
2020-03-31	true	1 Human Resc ▾	1 Australia ▾
2020-03-31	true	2 Services ▾	2 Japan ▾
2020-03-31	true	3 Services Sup ▾	3 Netherlands ▾
2020-03-31	true	4 Services Sup ▾	4 Portugal ▾
2020-03-31	true	5 Credit Contr ▾	5 Singapore ▾

Using Accelerators to Create Screens

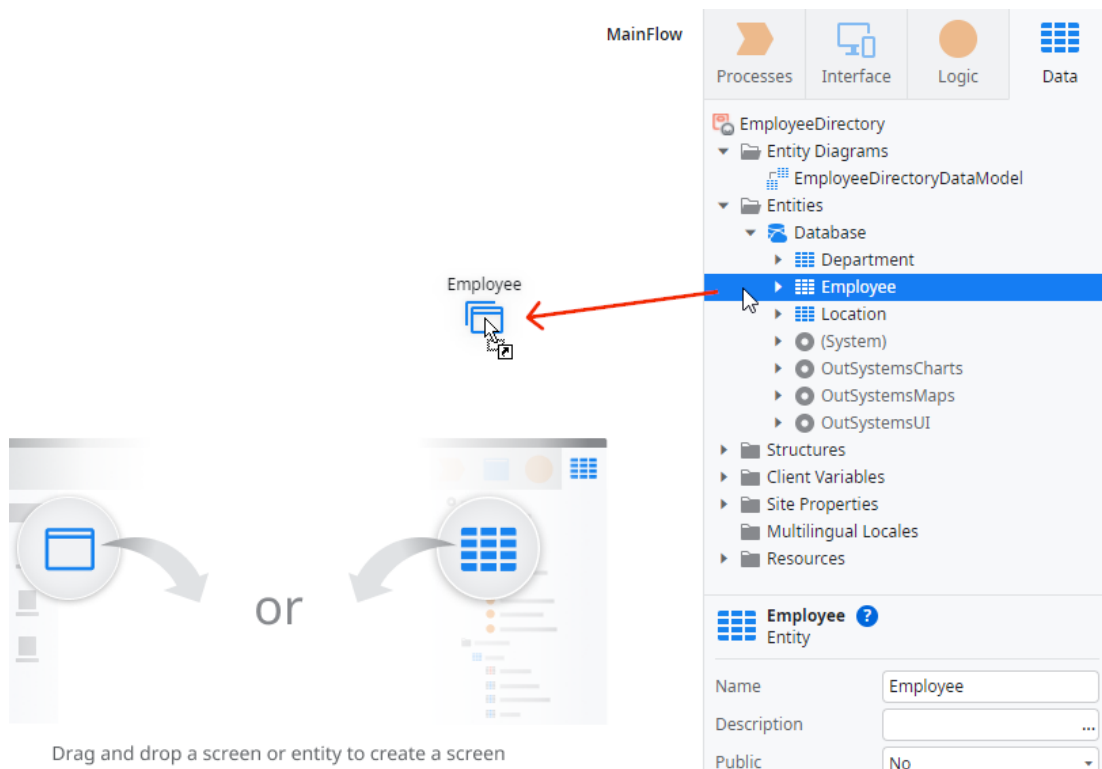
Now that the data model is ready, it is time to see the data on a proper Screen! And we're going to create two Screens with a couple of steps! Don't believe us? We'll show how easy it is!

- 1) Click on the **Interface** tab, on the top right corner of Service Studio, and double-click the **MainFlow** element.

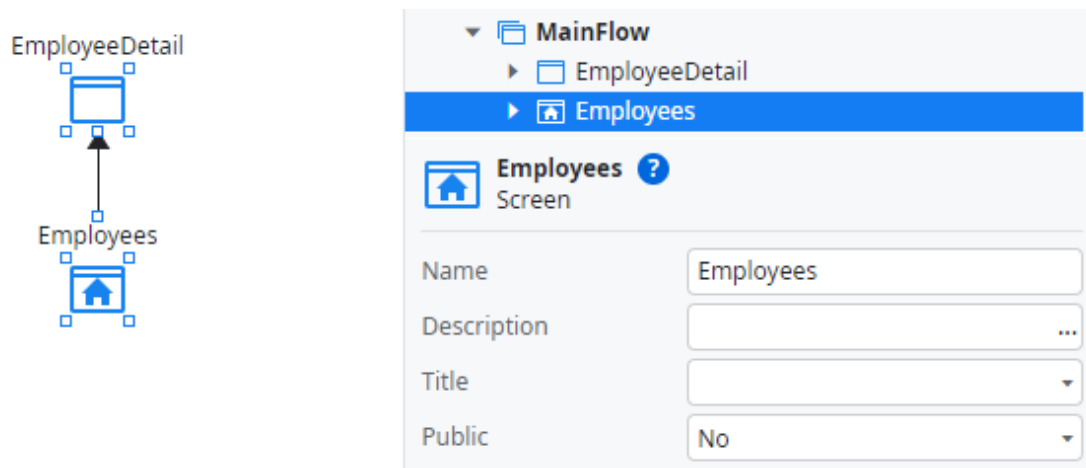


The MainFlow is where your Screens will be created. For now, we don't have Screens yet.

- 2) Switch back to the **Data** tab, drag the Employee Entity and drop it on the MainFlow area, just like the next image shows.



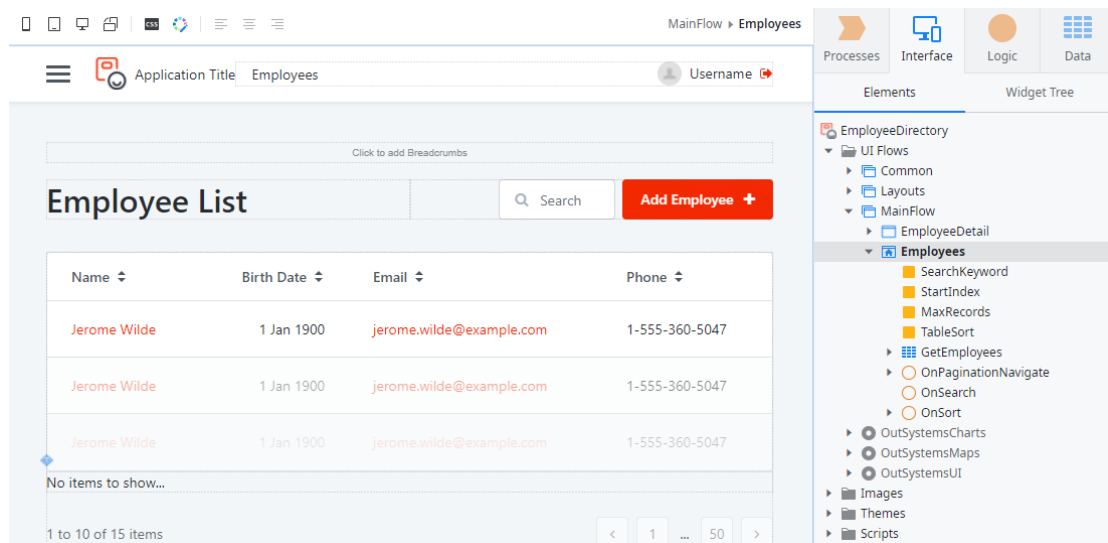
And just like that, you have created two Screens!



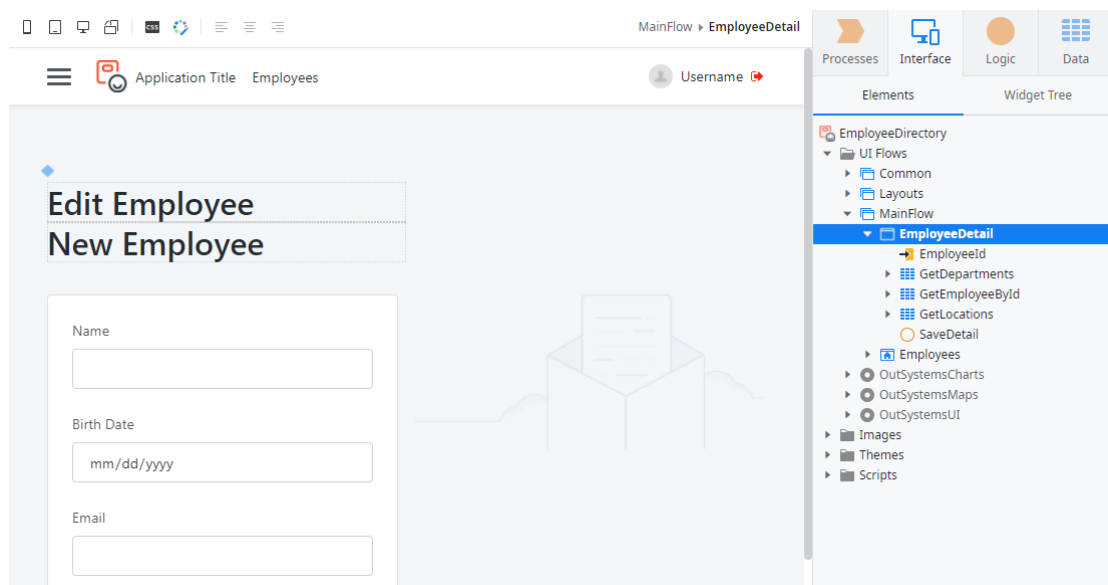
You can rename them if you want, but for now, we will use the automatically created names.

- 3) Double-click on the **Employees** Screen to see what was created. You can see that the Screen was automatically created with all the functionality that we

need: list of employees, search by employee name and the option to add a new employee.



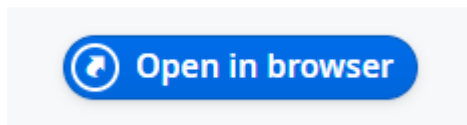
The same thing happens with the **EmployeeDetail** Screen.



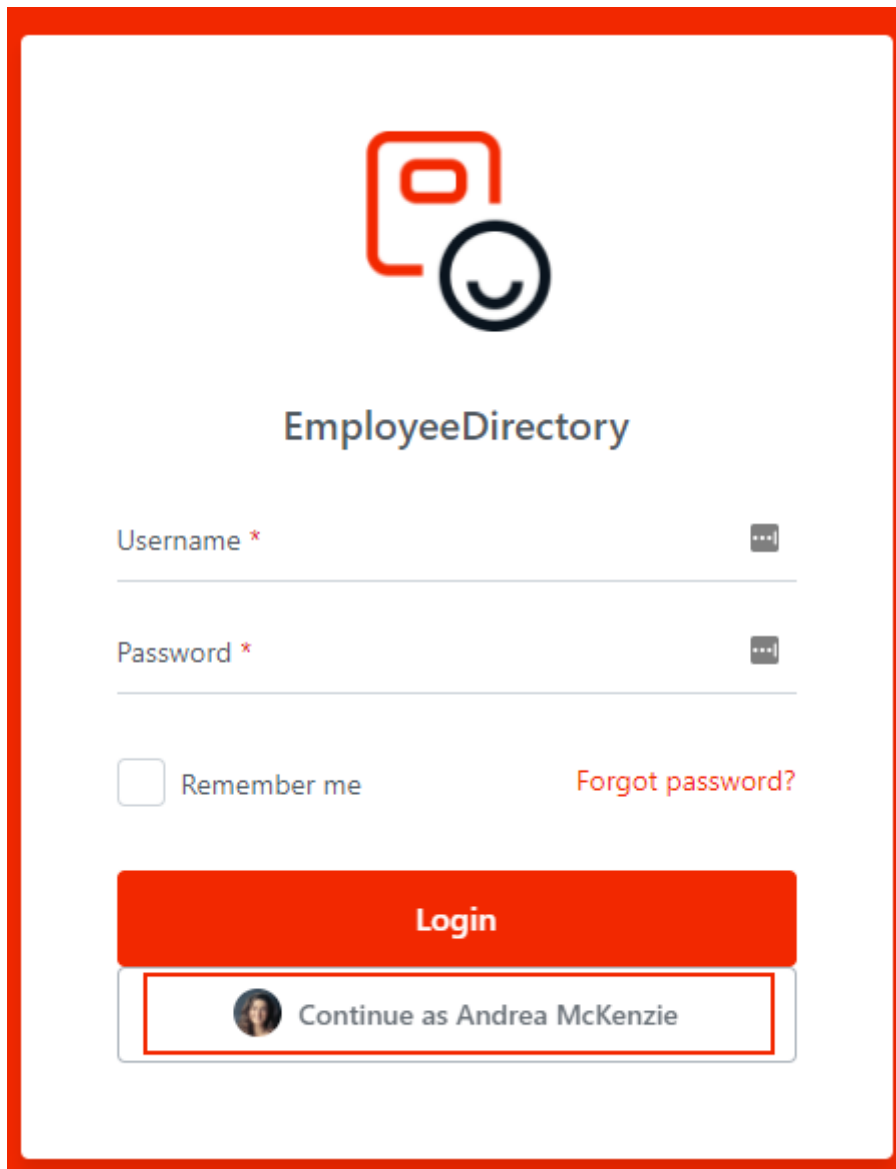
OutSystems uses the attributes in the Entity and creates a list and a detail Screen accordingly, following the application's layout. This saves us a lot of time and it can be customized if needed.


- 4) **Publish** the module again to save the last changes you did.

- 5) When the publishing is finished, a blue option appears: **Open in browser**. Click on it to open the application in the browser.





- 6) You should see a login screen. Use the sample login option available to log in.

A login screen for "EmployeeDirectory" enclosed in a red border. At the top is a logo consisting of a red square with a white 'O' inside, and a black circle with a white 'C' inside. Below the logo is the text "EmployeeDirectory". There are two input fields: "Username *" and "Password *", each with a small "..." icon to its right. Below the password field is a checkbox labeled "Remember me" and a link "Forgot password?". At the bottom is a red "Login" button. Below the "Login" button is a button with a user profile picture and the text "Continue as Andrea McKenzie".




EmployeeDirectory

Username * 

Password * 

☐ Remember me [Forgot password?](#)

Login

 Continue as Andrea McKenzie

7) Now you can test and interact with your app in the browser.

EmployeeDirectory Employees Andrea McKenzie

Employee List

 [Add Employee +](#)

Wrapping up

Congratulations on finishing this tutorial. With this exercise, we had the chance to go through some essential aspects of OutSystems 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) [Service Studio Overview](#)
- 3) [Intro to OutSystems Development](#)
- 4) [Modeling Data](#)
- 5) [Bootstrap and Entity Using an Excel File](#)

See you in the next tutorial!