Building Forms Using Widgets Exercise

Table of Contents

knowledge@outsystems.com

www.outsystems.com

Outline

In this exercise, we will design a user interface for entering, displaying, and modifying Project details. This will be achieved by incorporating specific widgets into the form, aligning each with the attributes of the Project Entity.

Upon completing this exercise, you can preview the application in a web browser, ensuring that all relevant information is readily accessible to users.

Scenario

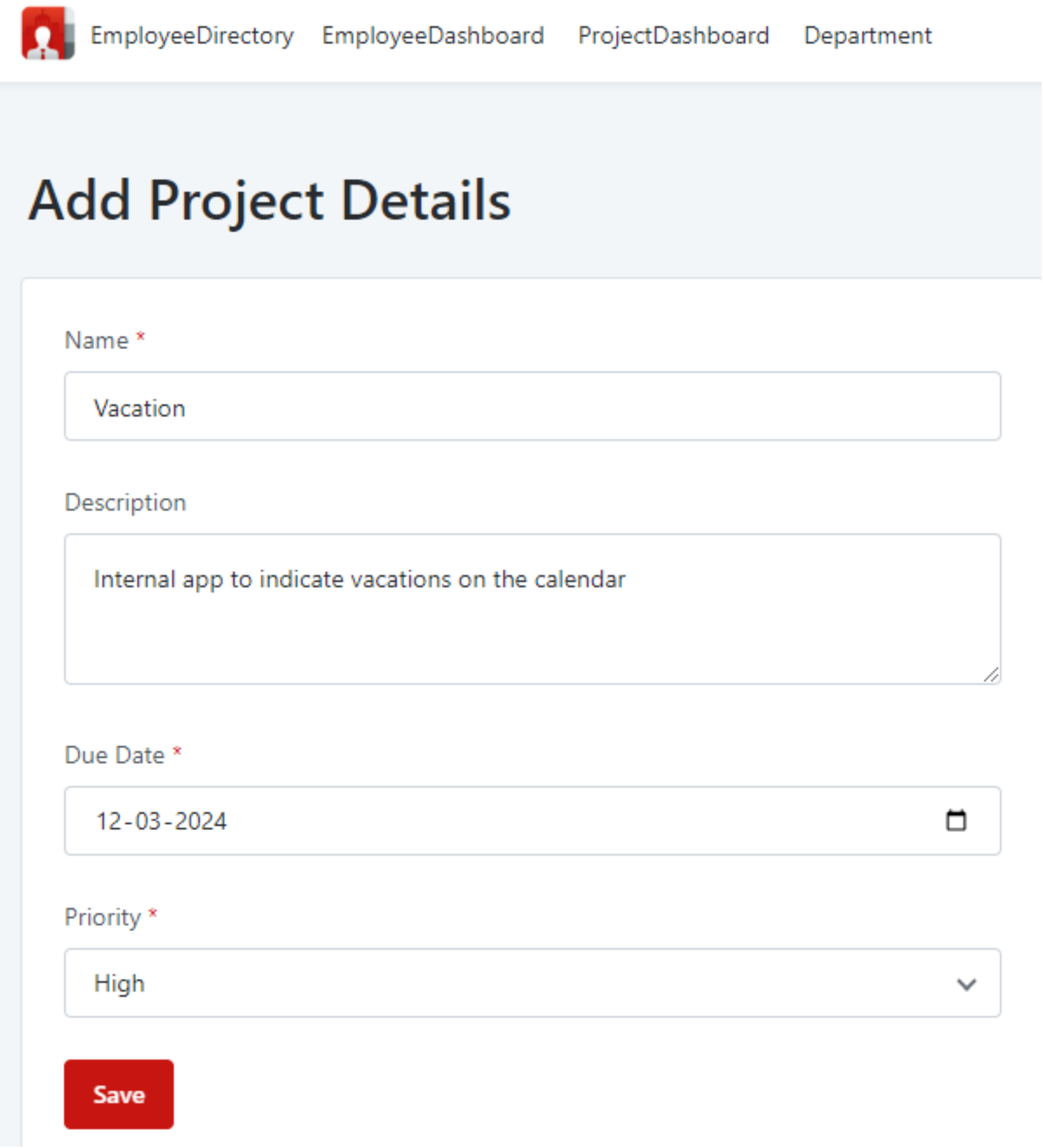
In this exercise, you will enhance the existing **Employee Directory** app, which consists of a single module containing essential entities, screens, and aggregates established in previous activities.

The module comprises six distinct screens, which have some Aggregates defined.

The ProjectDetails Screen has the **GetProject** Aggregate, which inputs and retrieves project data from the database.

Building upon this application, the focus of this exercise is to design a form within the ProjectDetails Screen to manually create an input widget for each attribute associated with a project.

Upon completing this exercise, the ProjectDetails Screen should mirror the visual representation depicted in the accompanying screenshot.



Add Project Details

Name *

Vacation

Description

Internal app to indicate vacations on the calendar

Due Date *

12-03-2024

Priority *

High

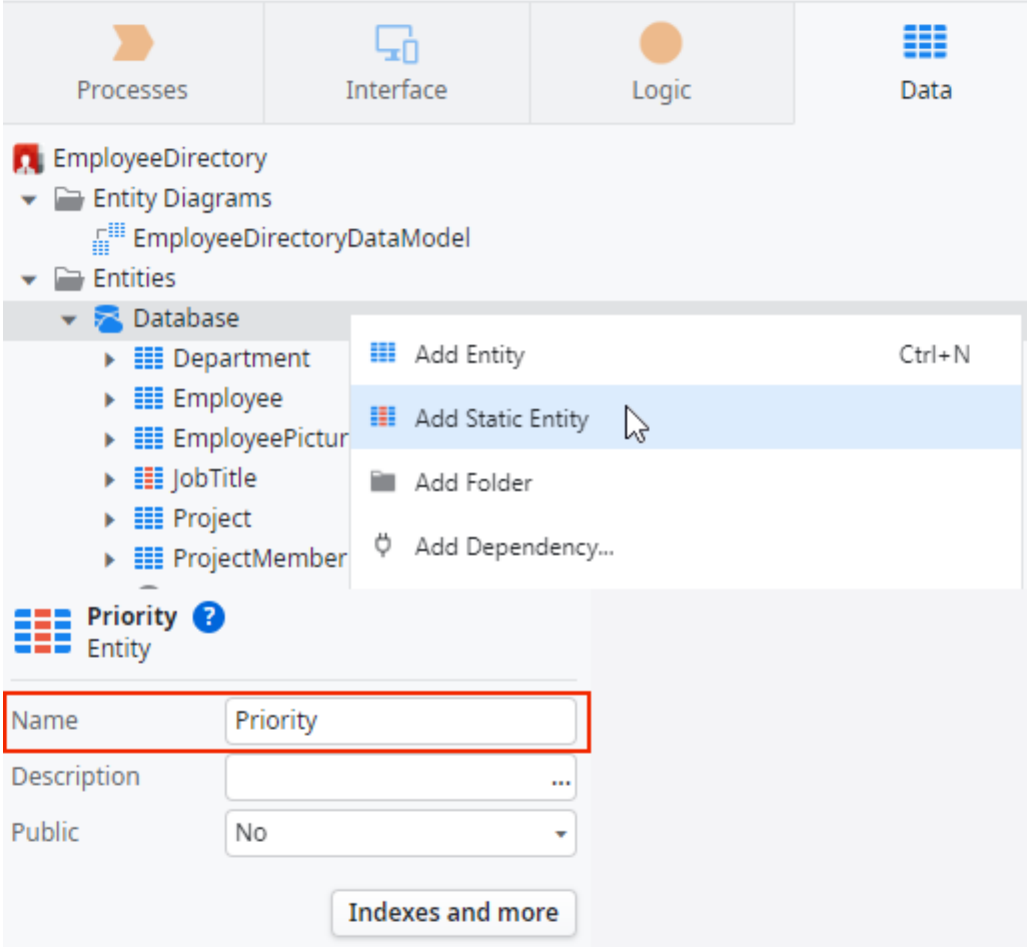
How-To

This section will show you how to do this exercise with a thorough step-by-step description. **If you already finished the exercise on your own, great! You don't need to do it again**. If you didn't complete the exercise, that's fine! We are here to help you.

Create Static Entity

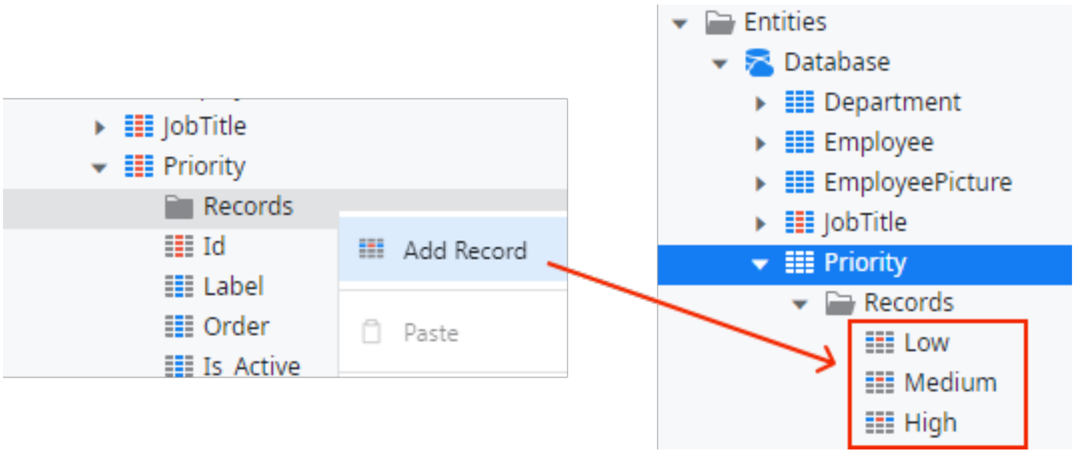
Assigning a unique priority level to each project based on its significance is essential for determining the project's completion due date. In this section, we will create a Priority Static Entity to hold priority records, subsequently we will link them to the corresponding Project Entity.

1) In the Data tab, add a Static Entity and name it *Priority*.

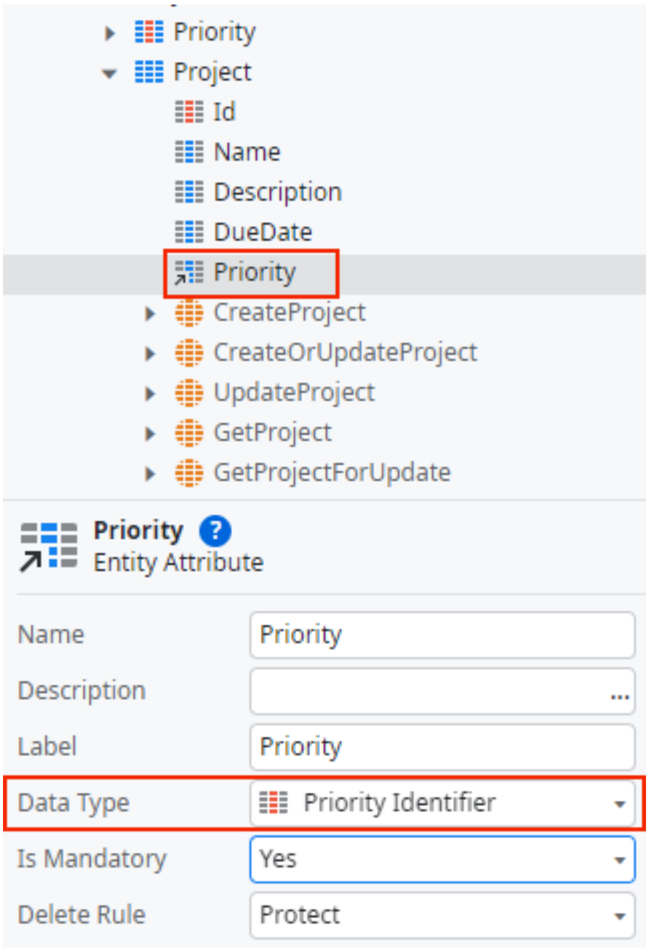


Name: Priority
Description:
Public: No

2) Populate it with the necessary records, as shown in the provided screenshot.

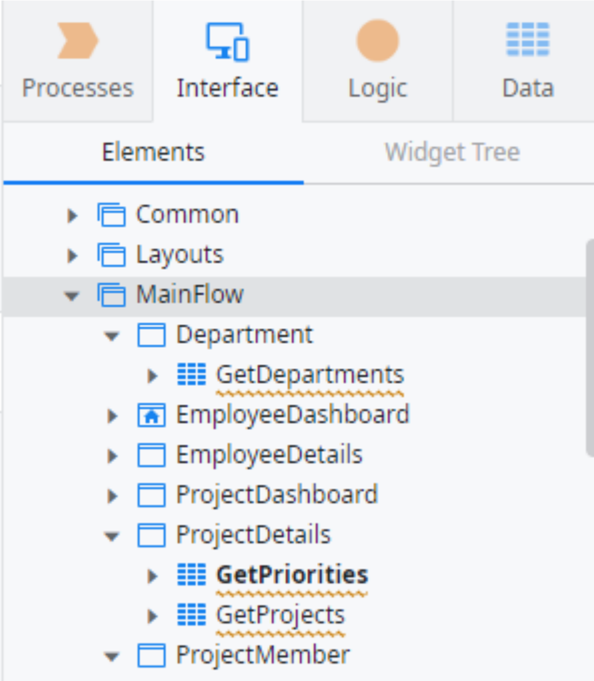


3) Add a attribute to the **Project** Entity, name it *Priority*, and set its Data Type as a *Priority Identifier*, making it a foreign key. Set Is Mandatory to *Yes*.



Name	Priority
Description	
Label	Priority
Data Type	Priority Identifier
Is Mandatory	Yes
Delete Rule	Protect

4) Create the *GetPriorities* Aggregate to the ProjectDetails Screen.

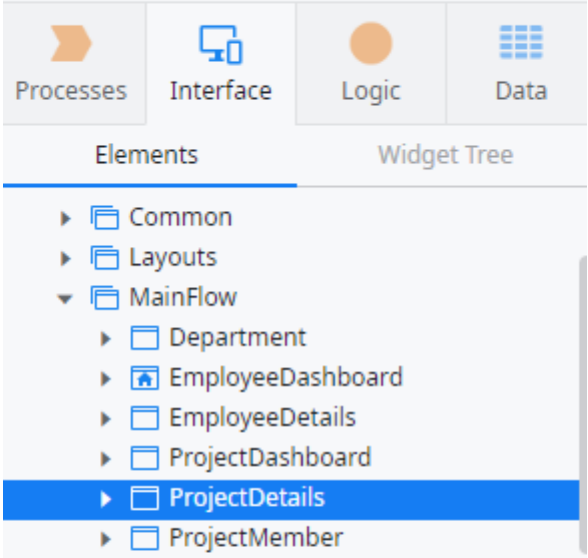


5) Publish the module to the server to save the latest changes.

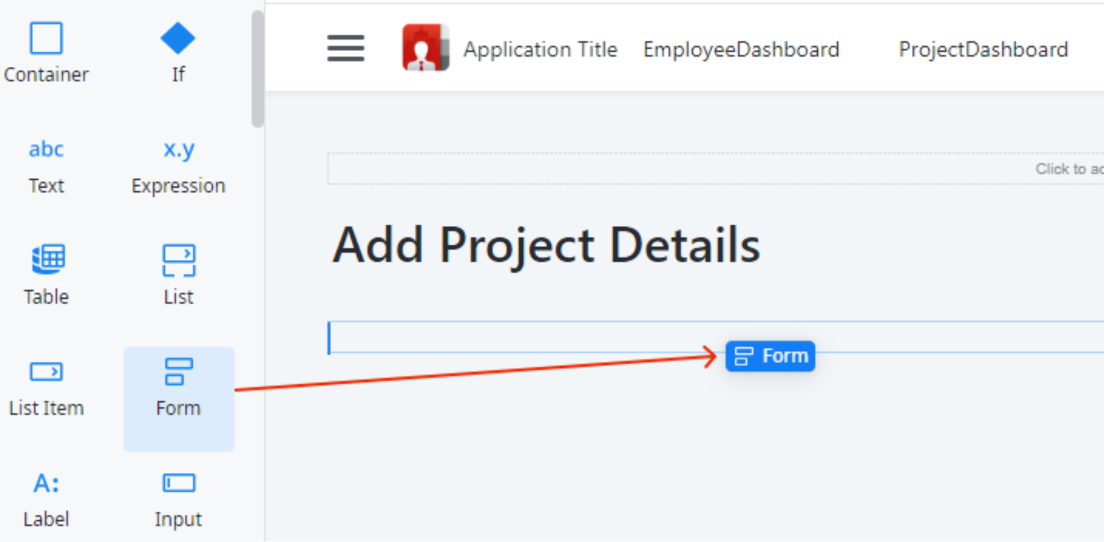
Create a Form Using Widgets

In this segment, we will construct the ProjectDetails Screen by incorporating various widgets, including Forms, Labels, Input fields, Dropdowns. Follow these steps to implement the desired interface on the ProjectDetails Screen:

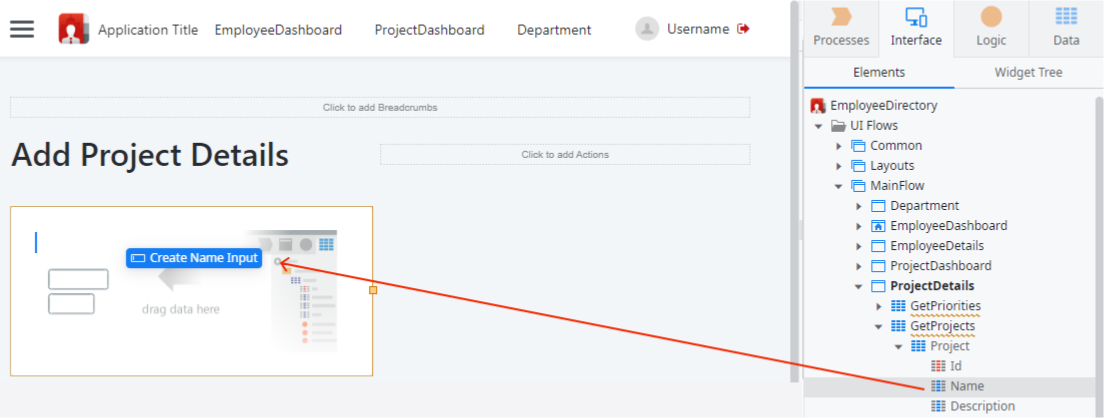
1) Open the **ProjectDetails** Screen by double-clicking on it.



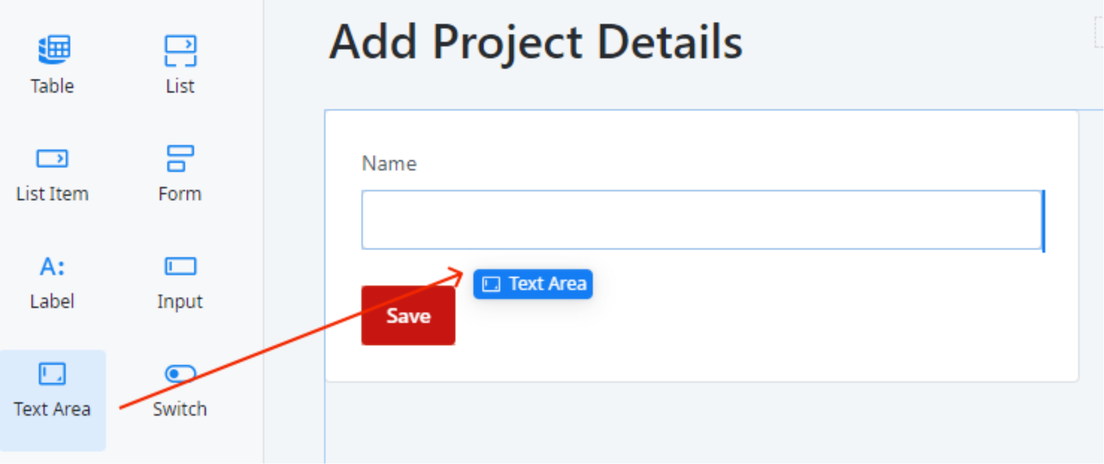
2) Insert a **Form** into the Container on the left in the **MainContent** area by dragging and dropping it.



3) Within the ProjectDetails Screen, expand the **GetProjects** Aggregate and, under the Project Entity, drag the **Name** attribute onto the Form.

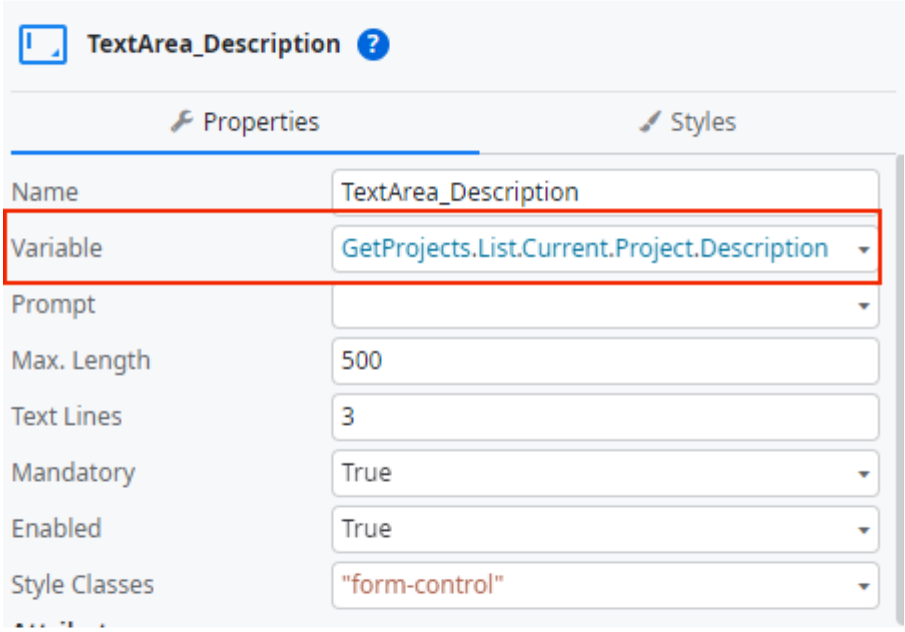


4) Add a **Text Area** widget between the Name input and the Save button.



5) In the **Properties** section, rename TextArea1 to *TextArea_Description*.

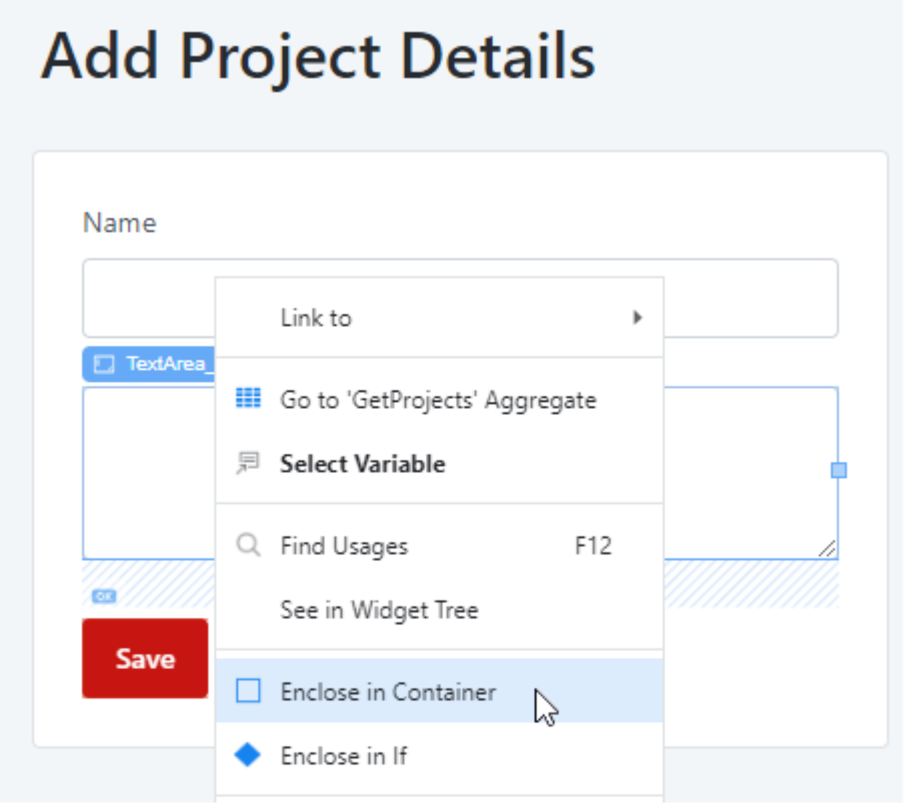
6) Double-click the **Variable** property of the Text Area and set its value to `GetProjects.List.Current.Project.Description`.



TextArea_Description

Name	TextArea_Description
Variable	GetProjects.List.Current.Project.Description
Prompt	
Max. Length	500
Text Lines	3
Mandatory	True
Enabled	True
Style Classes	"form-control"

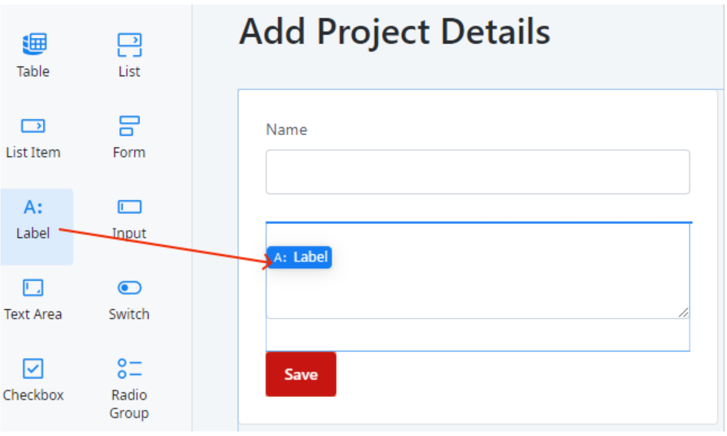
7) Right-click on the Text Area and opt for **Enclose in Container**.

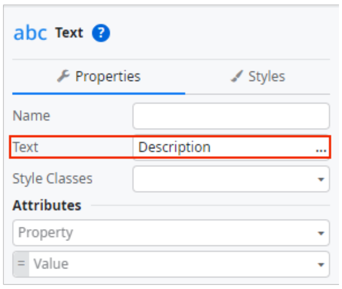


Add Project Details

Name

8) Place a **Label** widget before the Text Area within the new container and label it as *Description*.



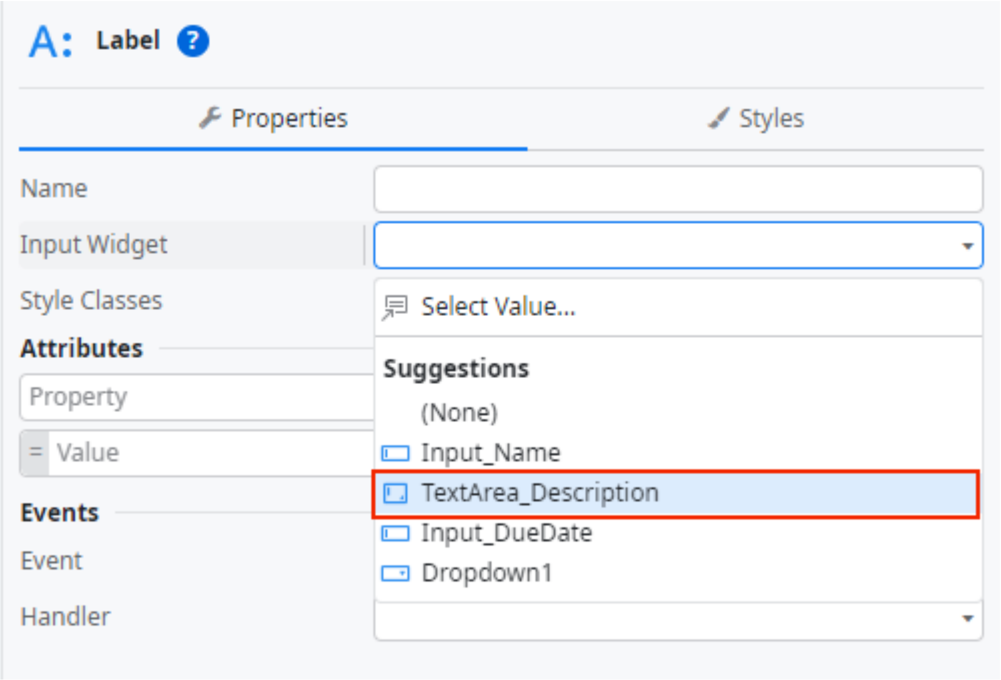


Name:

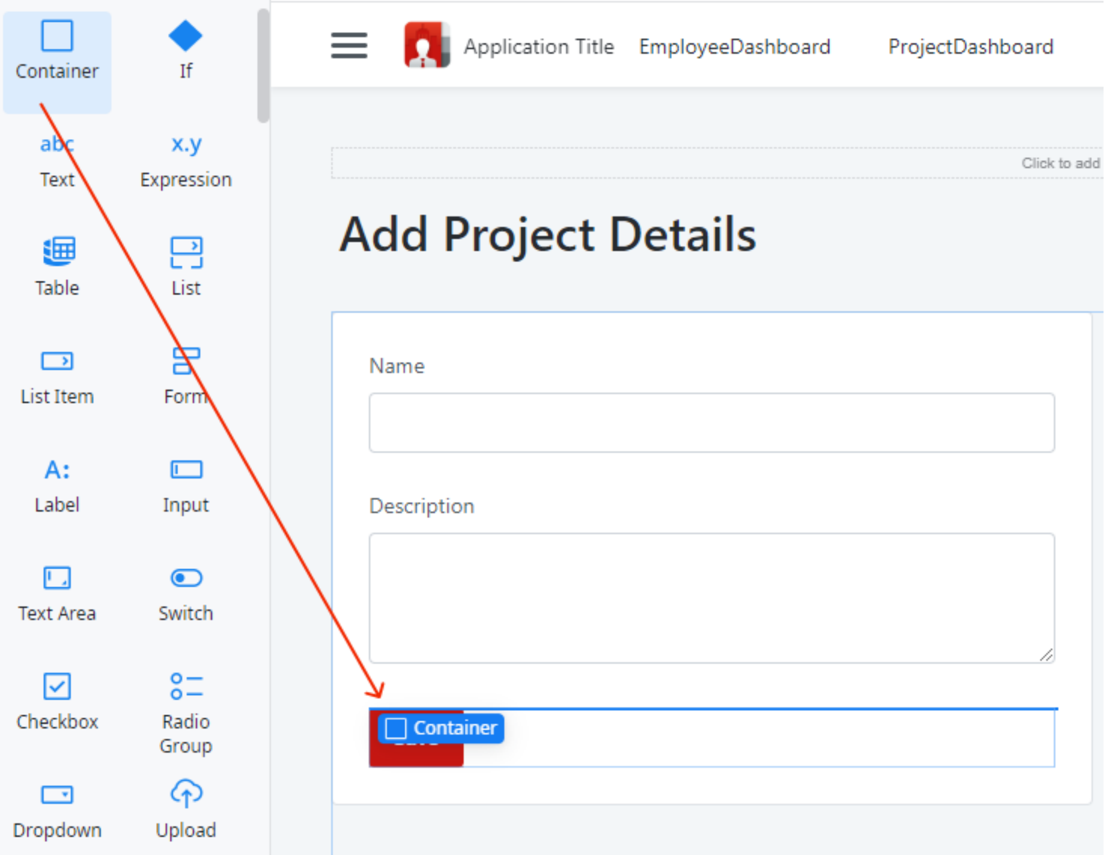
Text: Description

Style Classes:

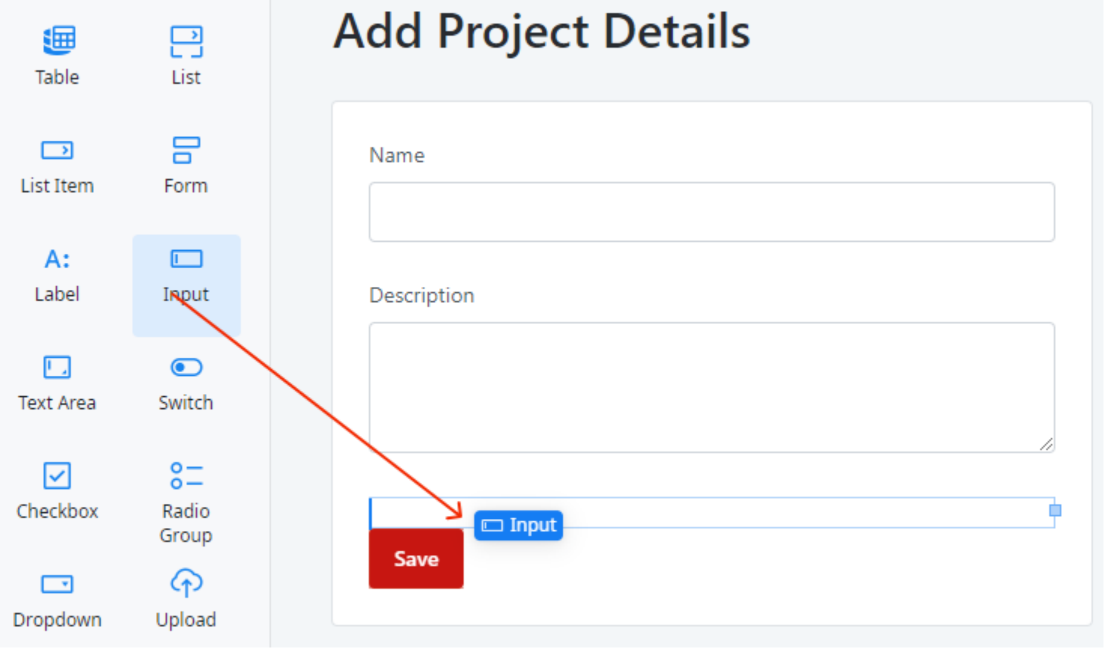
9) Associate the Label with the Text Area by setting the **Input Widget** property to *TextArea_Description*.



10) Add another **Container** below the one comprising the Description label and input but preceding the Save button.

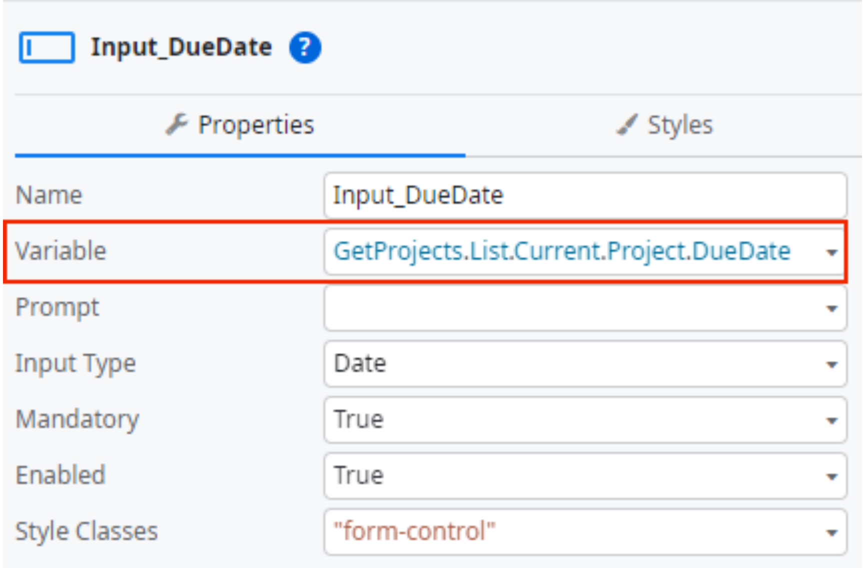


11) Drop an **Input** widget into the newly created container.



12) Rename the widget to *Input_DueDate*

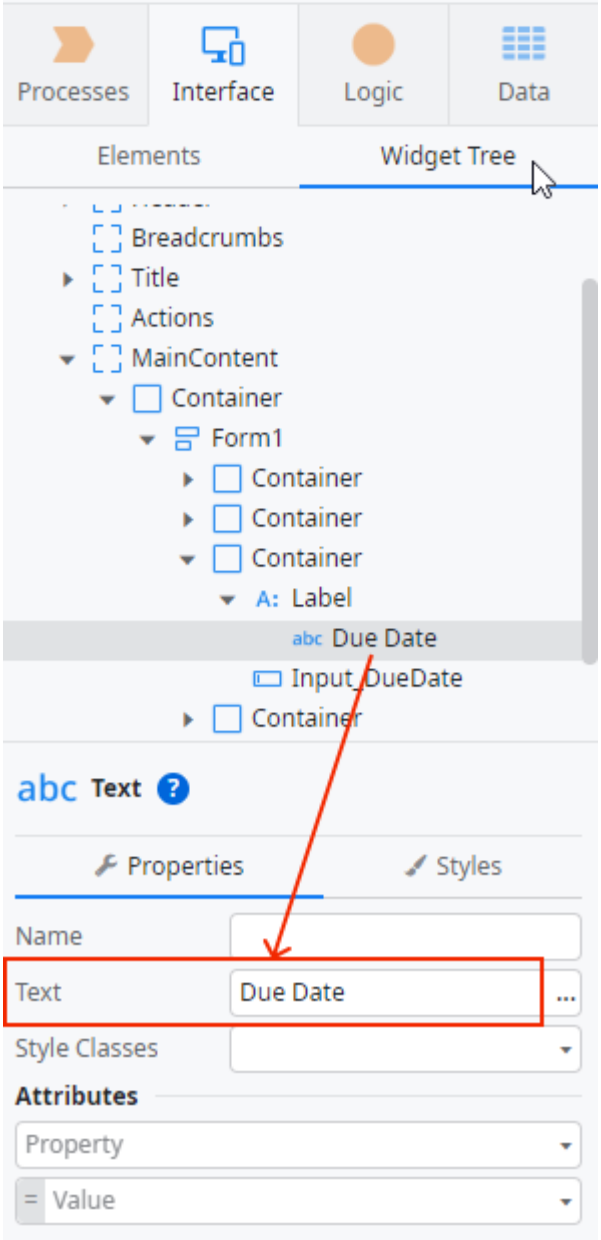
13) Set the **Variable** property of the Input widget to `GetProjects.List.Current.Project.DueDate`.



Input_DueDate

Name	Input_DueDate
Variable	GetProjects.List.Current.Project.DueDate
Prompt	
Input Type	Date
Mandatory	True
Enabled	True
Style Classes	"form-control"

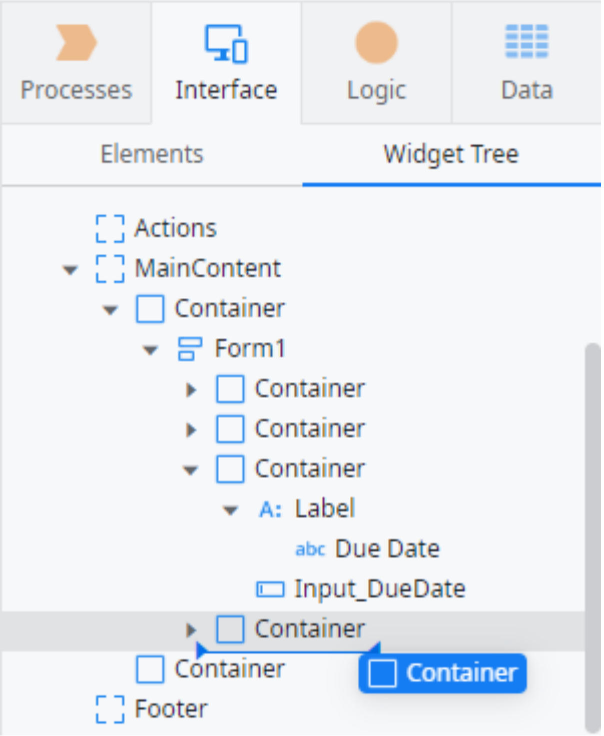
14) Similar to the Description, insert a new Label just before the Input widget within the Container, labeling it as *Due Date*.



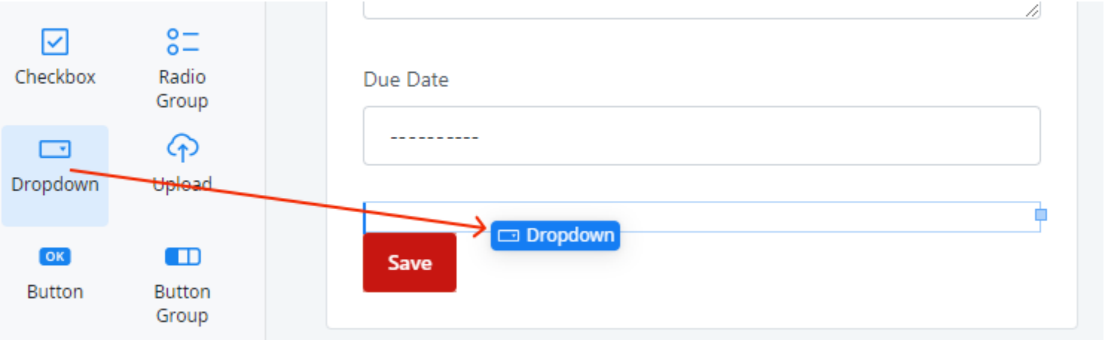
Name
Text: Due Date
Style Classes

15) Set its Input Widget property to *Input_DueDate*.

16) Add a final Container between the container surrounding the Due Date input field and label, but before the Save button.



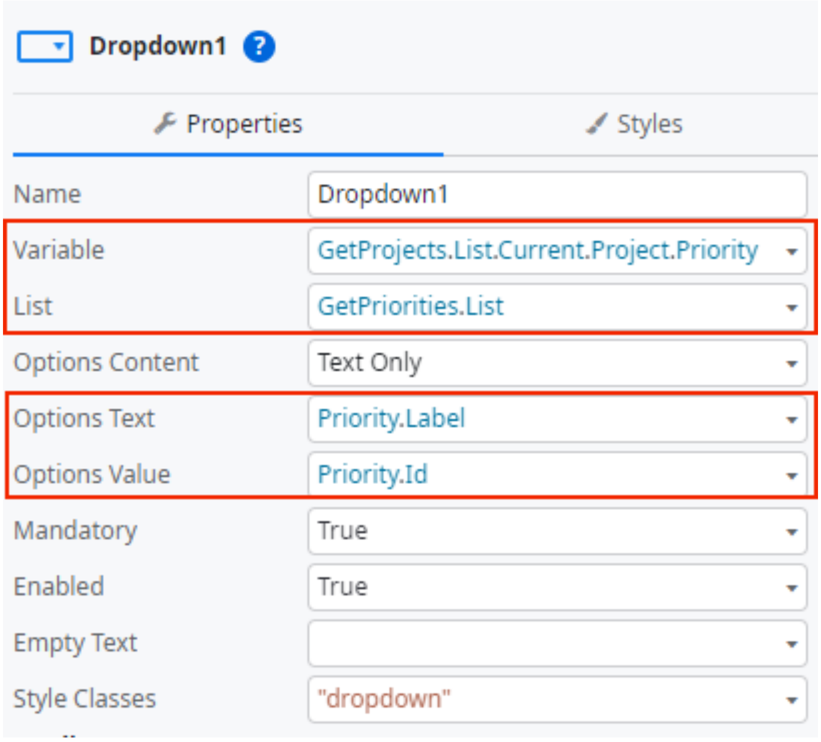
17) Place a **Dropdown** widget inside the new container.



18) In the Dropdown, we want to select the **Priority** of the project. For that, the Dropdown must list all the priorities. Since the Priority attribute is of type Priority Identifier, the value chosen by the user must have the same data type. set the following properties accordingly:

- **Variable**: `GetProjects.List.Current.Project.Priority`
- **List**: `GetPriorities.List`
- **Options Text**: `Priority.Label`

- **Options Value**: `Priority.Id`

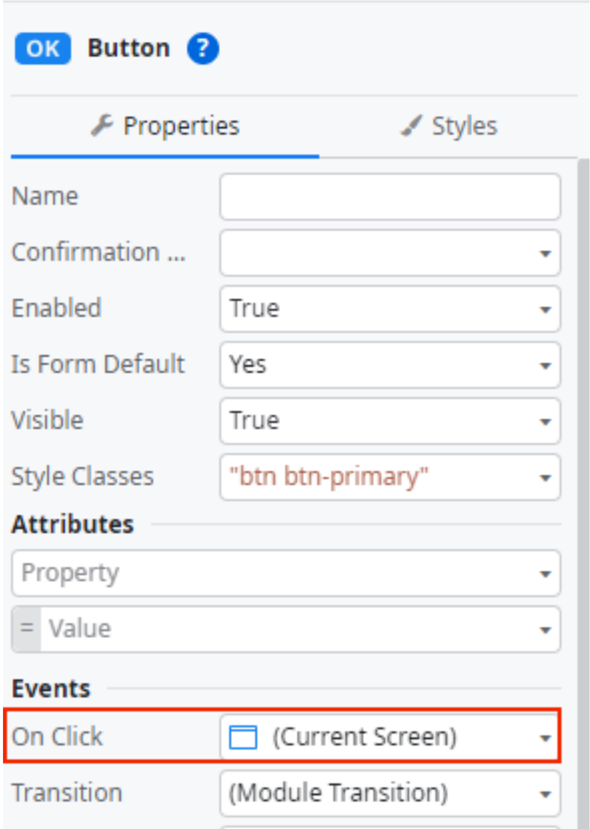


Dropdown1

Name	Dropdown1
Variable	GetProjects.List.Current.Project.Priority
List	GetPriorities.List
Options Content	Text Only
Options Text	Priority.Label
Options Value	Priority.Id
Mandatory	True
Enabled	True
Empty Text	
Style Classes	"dropdown"

19) Just like we did for the Description and Due Date, define a **Label** for the Dropdown, name it *Priority*, and set its Input Widget property to *Dropdown1*.

20) Set the **On Click** property of the Save button to the **Current screen**.



Name:
Confirmation ...:
Enabled: True
Is Form Default: Yes
Visible: True
Style Classes: "btn btn-primary"

Attributes

Property
= Value

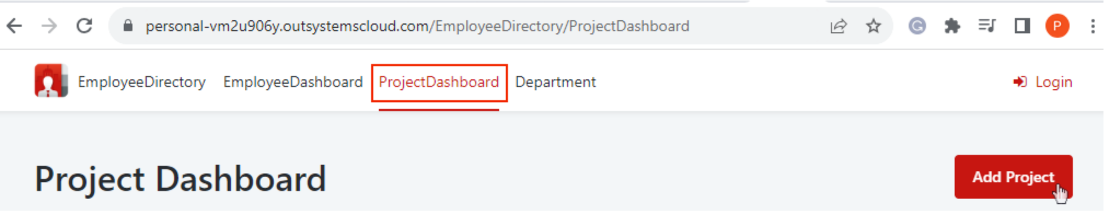
Events

On Click: (Current Screen)
Transition: (Module Transition)

21) Publish the module to the server to save the latest changes.

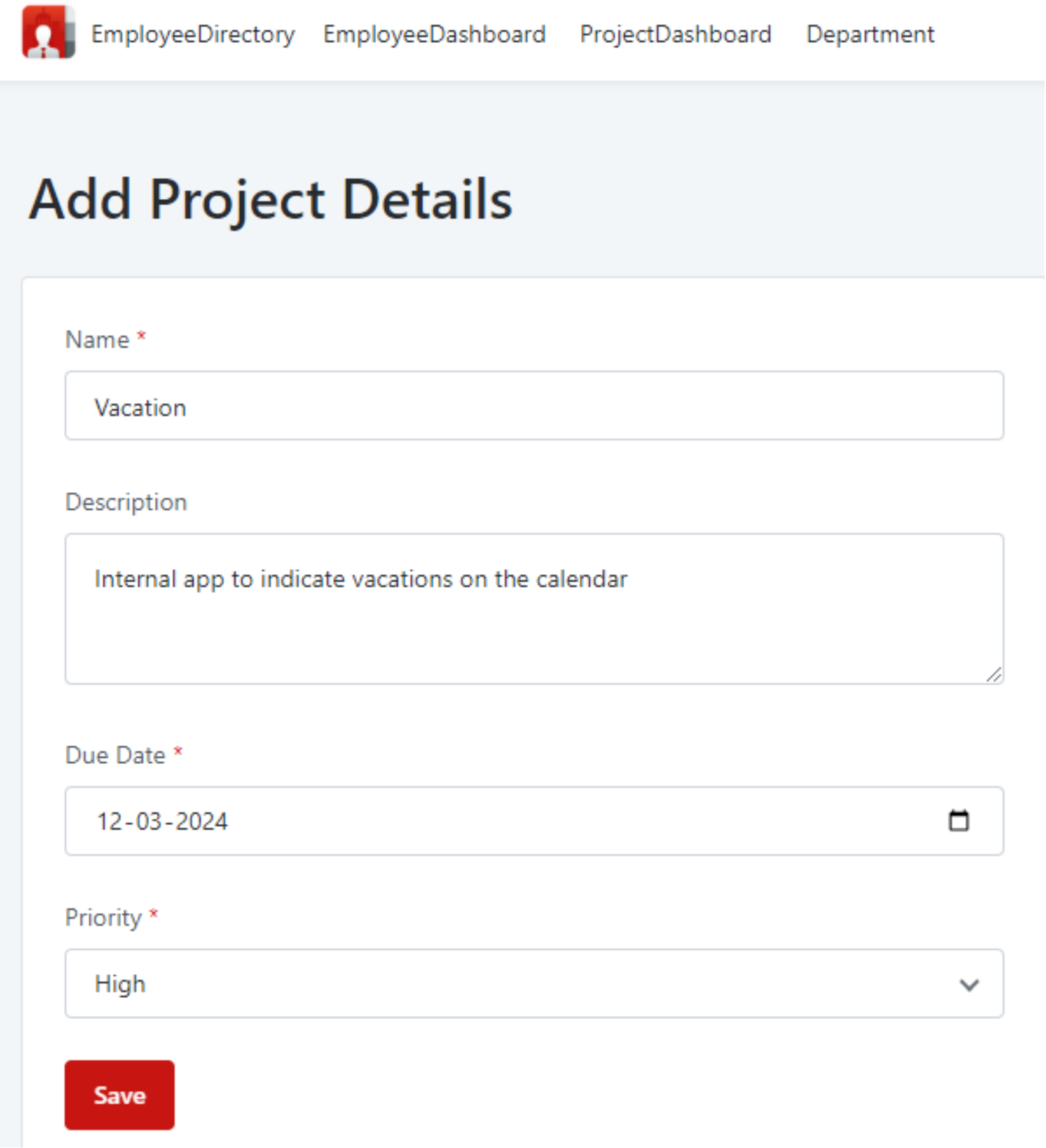
22) Open the application in the browser and navigate to the **ProjectDashboard** Screen through the app menu.

23) Click the **Add Project** button to transition to the ProjectDetails Screen.



personal-vm2u906y.outsystemscloud.com/EmployeeDirectory/ProjectDashboard

24) Observe the newly added widgets in this section. As you can see, the form fetches and displays the first record within the Project Entity.



Add Project Details

Name *

Vacation

Description

Internal app to indicate vacations on the calendar

Due Date *

12-03-2024

Priority *

High

In the Logic lesson, we will explore the process of presenting an empty form upon launching the screen to input and save new project data into the database.

Congratulations on the successful creation of input widgets in the form that efficiently inputs and presents the retrieved data from the database.