Design of an SAP Fiori Application for Creating a Sales Activity

Story

Most enterprises have a number of sales agents whose job involves interacting with clients regularly to generate business and obtain feedback about the products/services provided. These meetings happen either at the client office or over the telephone. The details of these meetings need to be recorded to help the organization improve its services and products.

For these people, the time spent in recording the visit/conversation details is not a value adding activity. In case the representatives are travelling between locations, they may not be able to update details in the system immediately. This delay might lead to omission of important details. Thus, it is best if the results are recorded when the details are still fresh in the mind.

A web based app for recording sales activities, on the other hand, enables the agents to record details while on the move using their mobile devices. A simplified user interface allows the users to concentrate more on the task at hand, leading to increased user productivity.

Persona

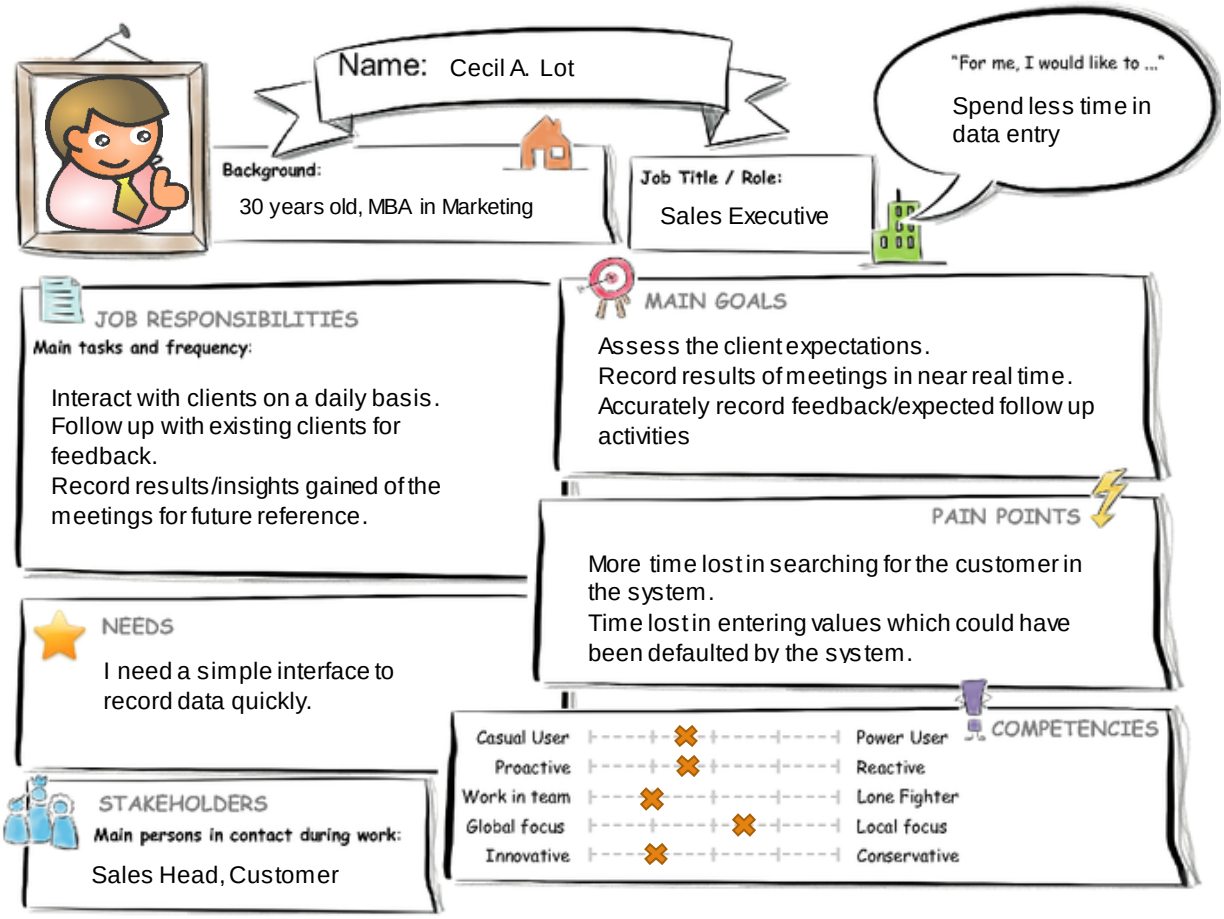

Figure 1 Mr Cecil A. Lot

For the purposes of designing and building this application, a persona of a 30 year old sales representative, working for a MNC is created. The main points regarding Mr Cecil A. Lot are shown in Figure 1.

User Experience Journey

The organization where Cecil works runs its business using SAP ERP. As a result, for recording day to day sales activities, Cecil has to use the Create Sales Activity (VC01N) transaction. His experience in working with the transaction is shown in Figure 2, where the red dots indicate the pain points and the blue dots are the desirable outcomes.

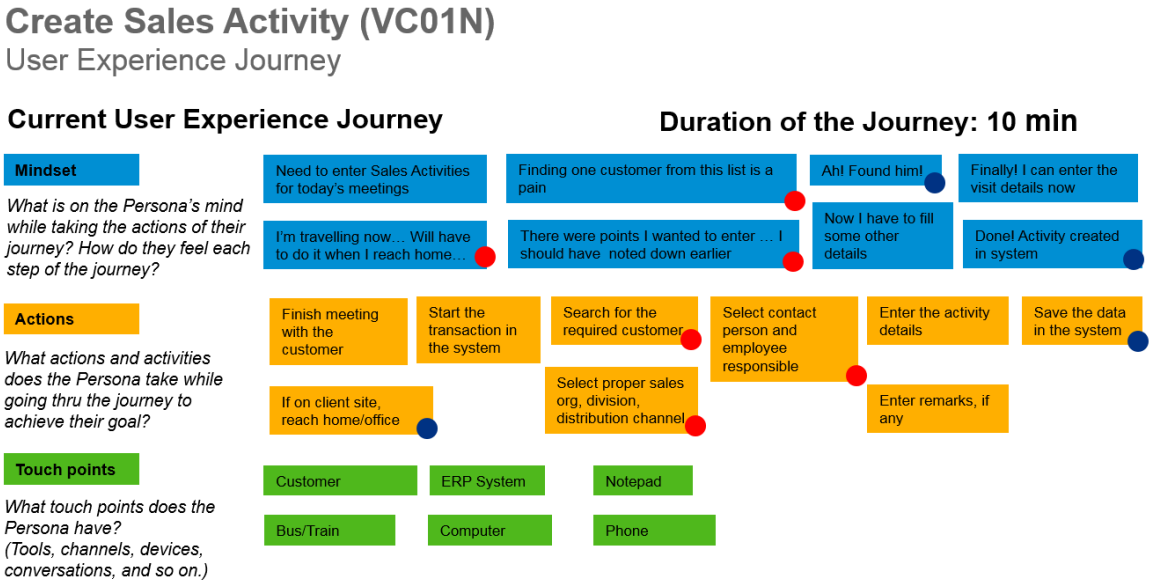

Figure 2 Cecil and VC01N

The journey duration is for the case when Cecil has access to a computer from where he can execute the transaction. It does not include the time spent in travel, in case of meetings at customer premises.

The Fiori Application

To help Cecil, a simplified version of VC01N will be built using SAP Fiori. That way, Cecil and his colleagues can access the application from their phones and tablets in addition to desktops. A mock-up of the application, created using the SAP Fiori Prototyping Kit, is shown in Figure 3. It is a *Fiori Master Detail* application, with the customers assigned to Cecil (or the user) in the master list on the left, and form for creating the Sales Activity on the right.

The OData Model

A partial view of the OData Model which will be used by the application is shown in Figure 4. The structure was first written in an *.edmx* file, which was provided to the New Project Wizard of SAP Web IDE. It contains two types, *Customer* and *SalesActivity*. For the create activity, the type *SalesActivity* will not be bound to any fields on the screen. The OData service will use Cecil's login credentials to determine the list of customers.

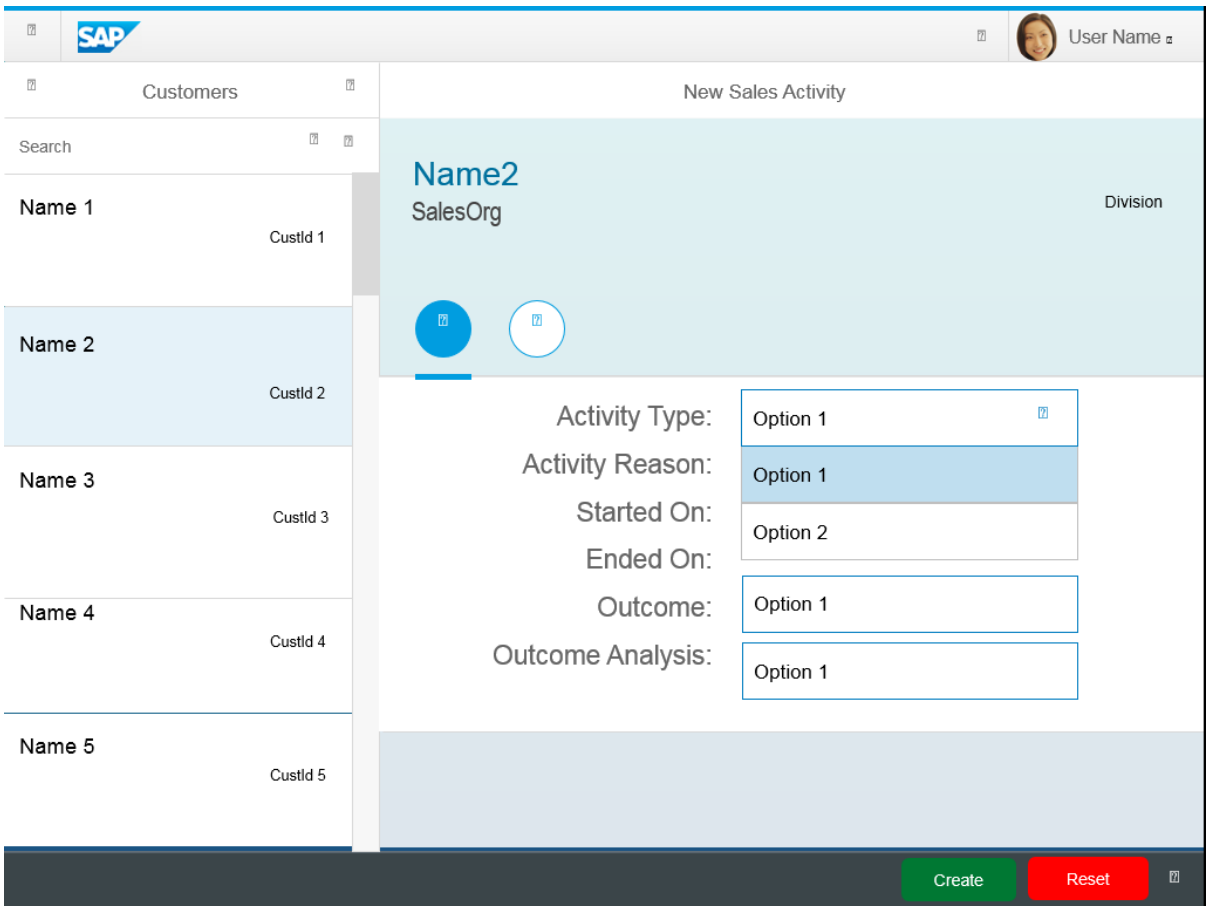

Figure 3 Mock-up of the application

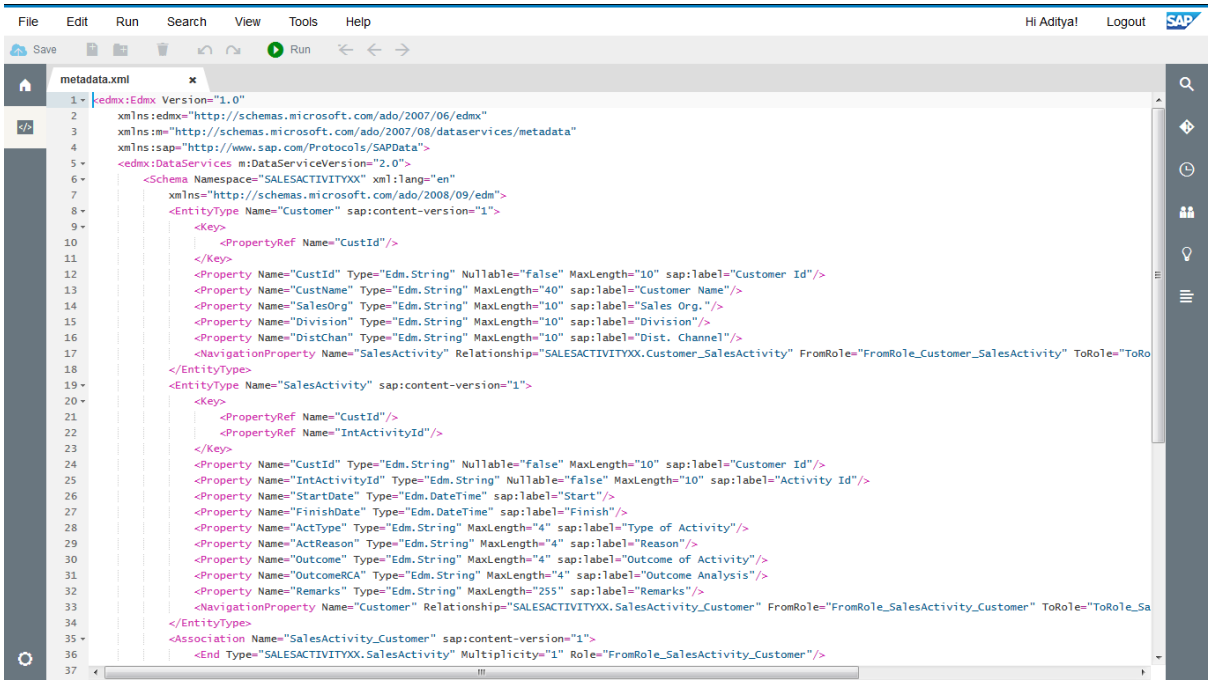

Figure 4 OData Model

The User Interface

Figure 5 shows a screen shot of the application running with mock data in SAP Web IDE. It was created by modifying the views generated by the New Project Wizard.

At present, the Create button does nothing more than display an information message. In the completed application, this will call the service to create the Sales Activity in the backend ERP system.

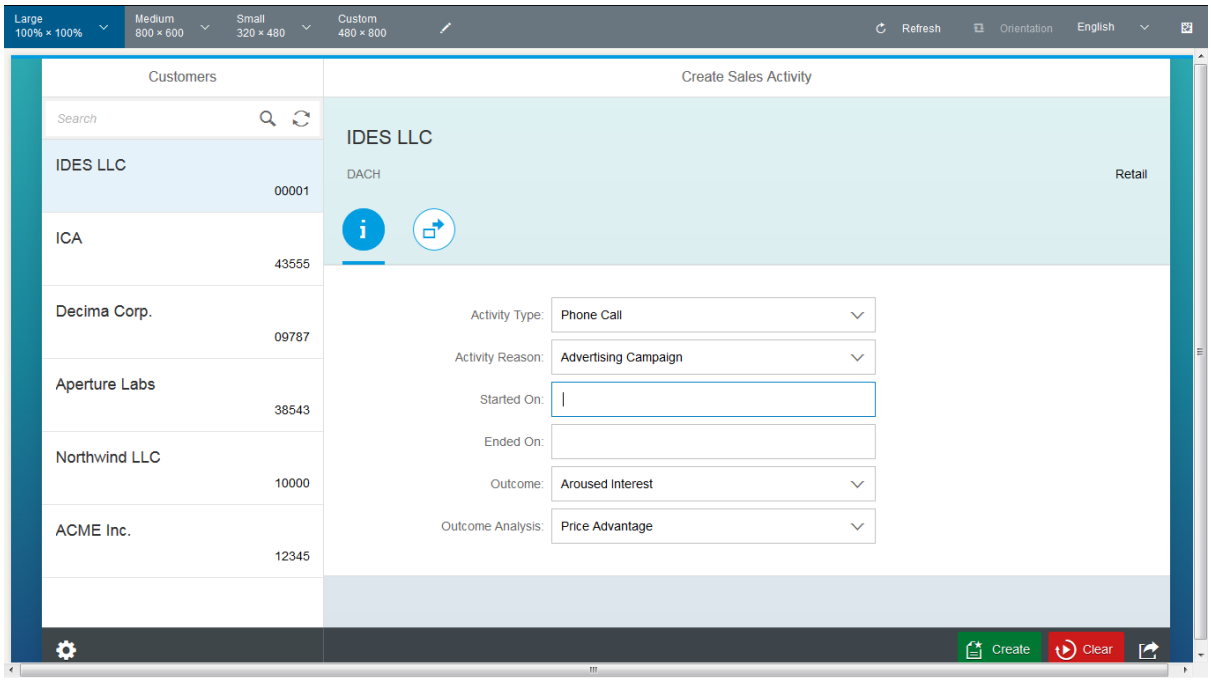

Figure 5 Screen shot of the Fiori application

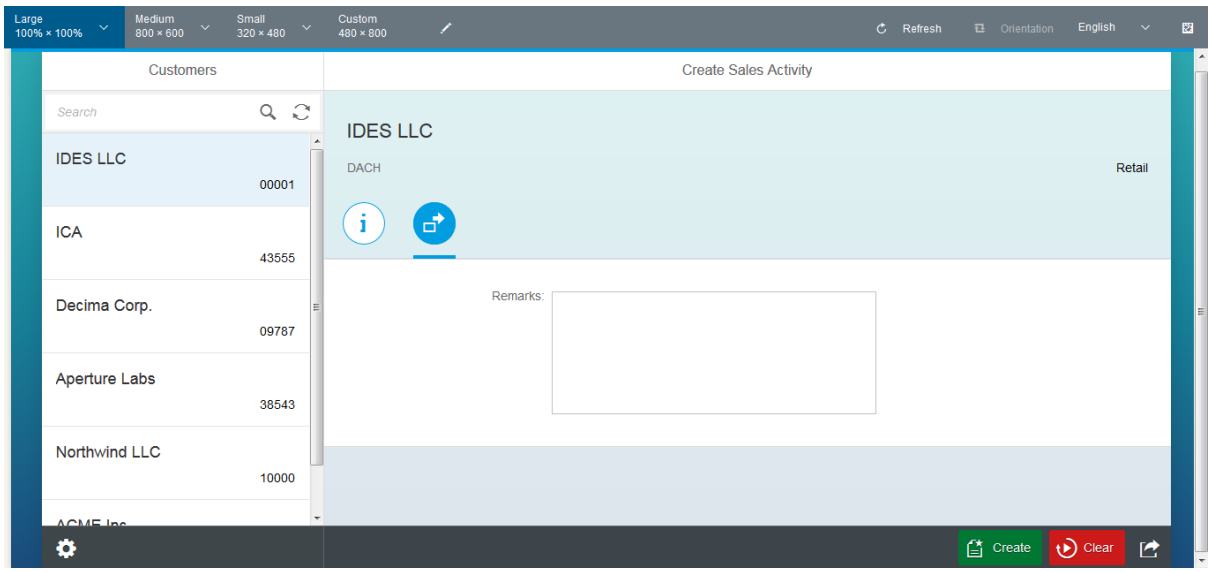

Figure 6 The Remarks input field

The Controller

The dropdowns for *Outcome* and *Outcome Analysis* fields are linked, i.e. the values in the *Outcome Analysis* dropdown change if the *Outcome* is changed. The logic for this is implemented in the

controller, which is shown in Figure 7. For this prototype, the dropdown values are hard coded in the *OnInit* method of the controller, but in a productive environment, they will be provided by the OData service.

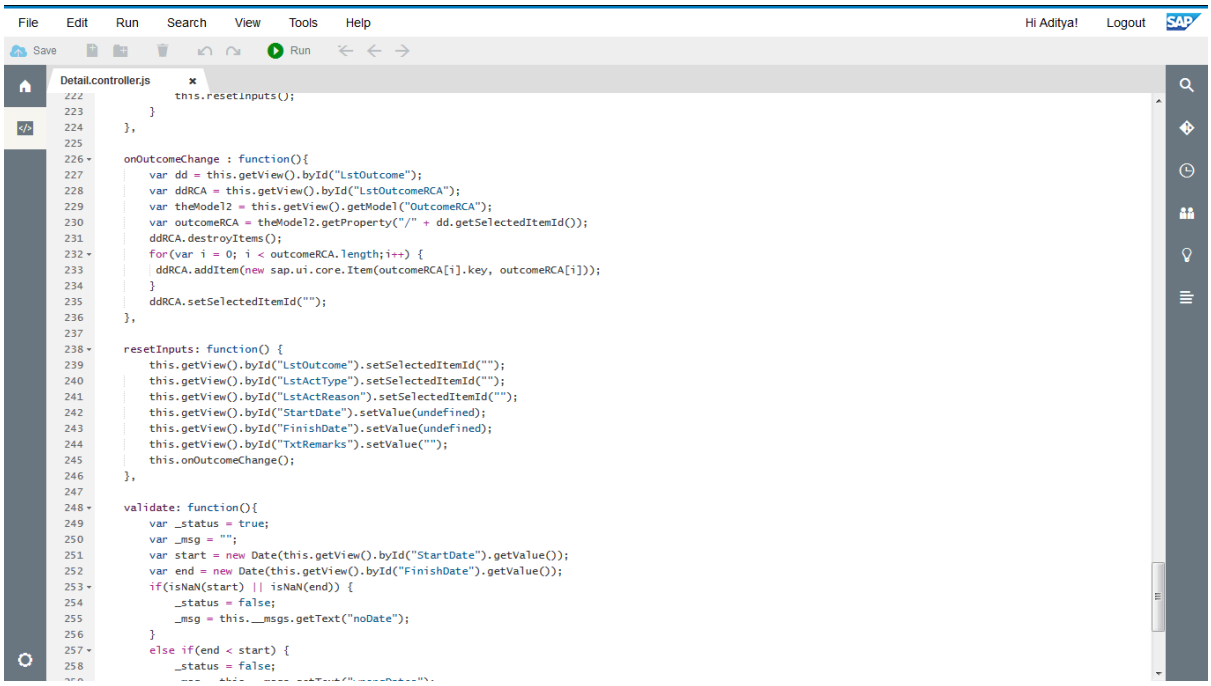

Figure 7 Controller for the Detail view

Screen Shots

This section contains screen shots of the application, depicting some of the interface functionality developed thus far.

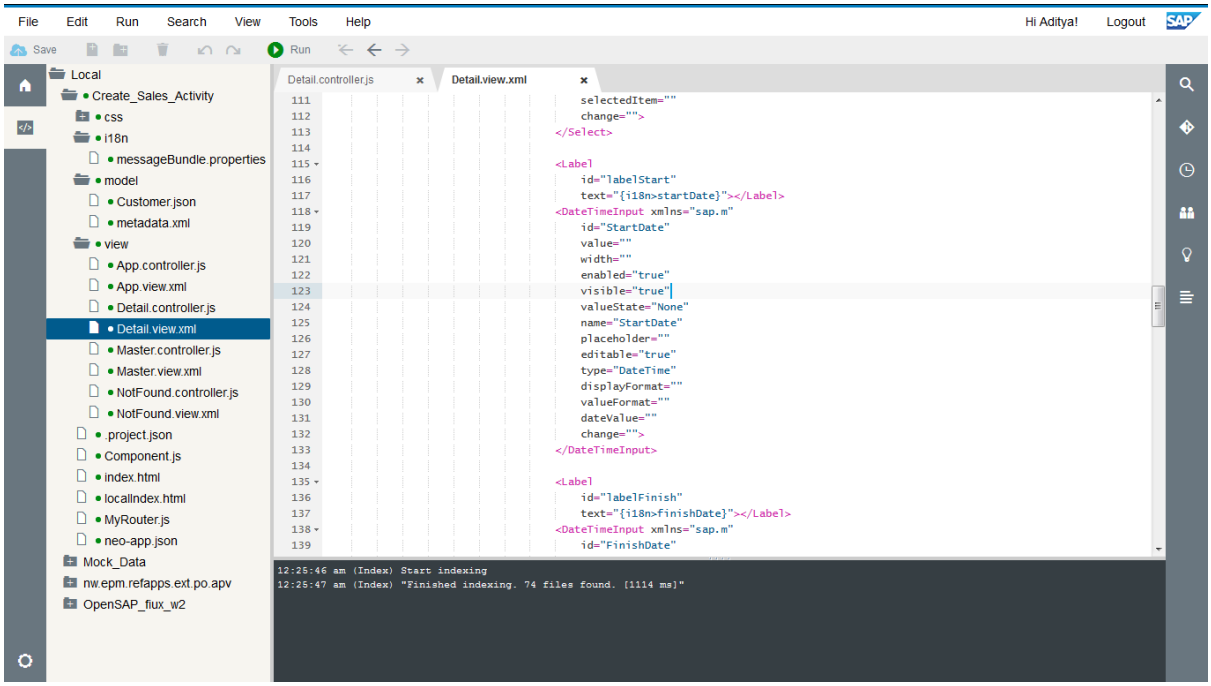

Figure 8 Project Structure

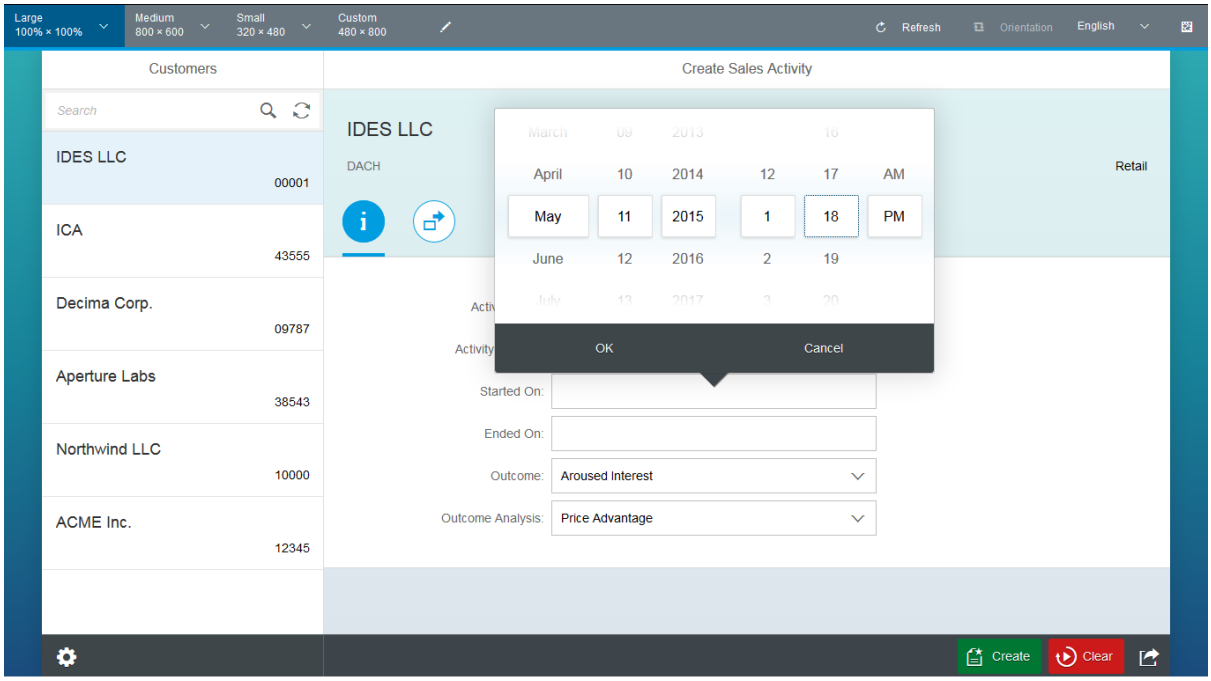

Figure 9 Using the DateTimeInput control for date and time input

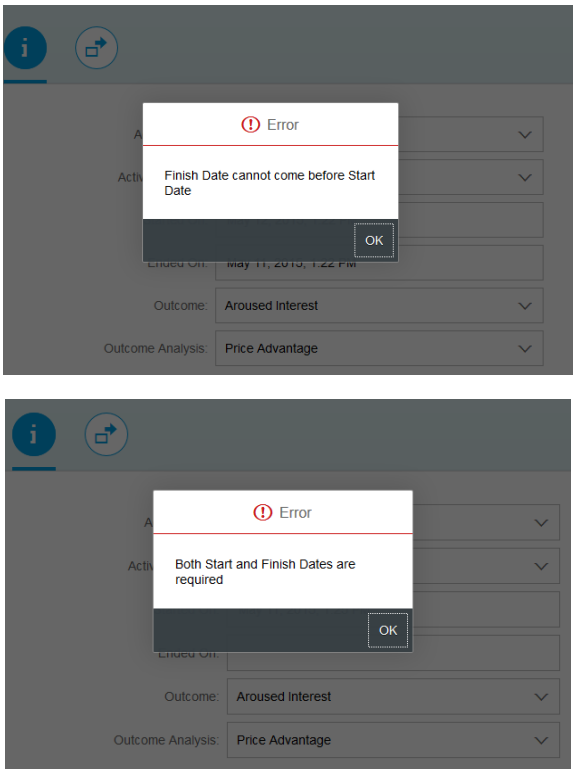

Figure 10 Validations for Date Input

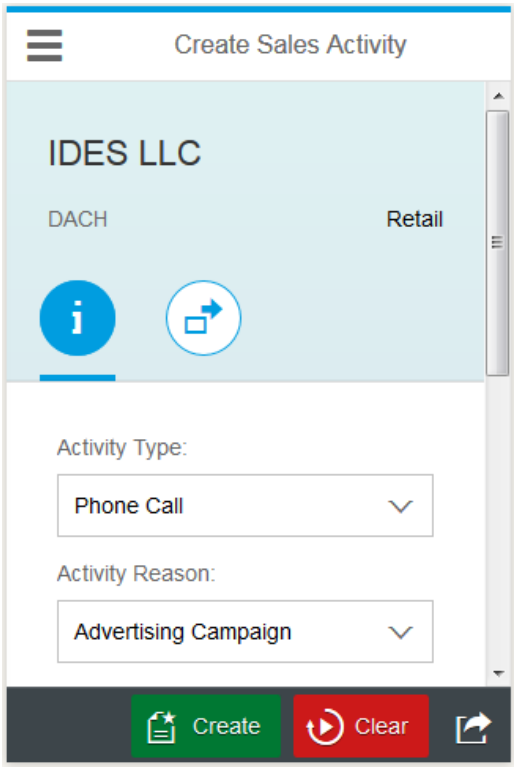

Figure 11 Mobile Layout preview

```
actReason=Activity Reason
startDate=Started On
finishDate=Ended On
outcomeRCA=Outcome Analysis
outcome=Outcome
```

Figure 12 i18n