Building Satellite Rollbase Applciation for an existing OpenEdge application

Authors:

Ganesh Cherivirala

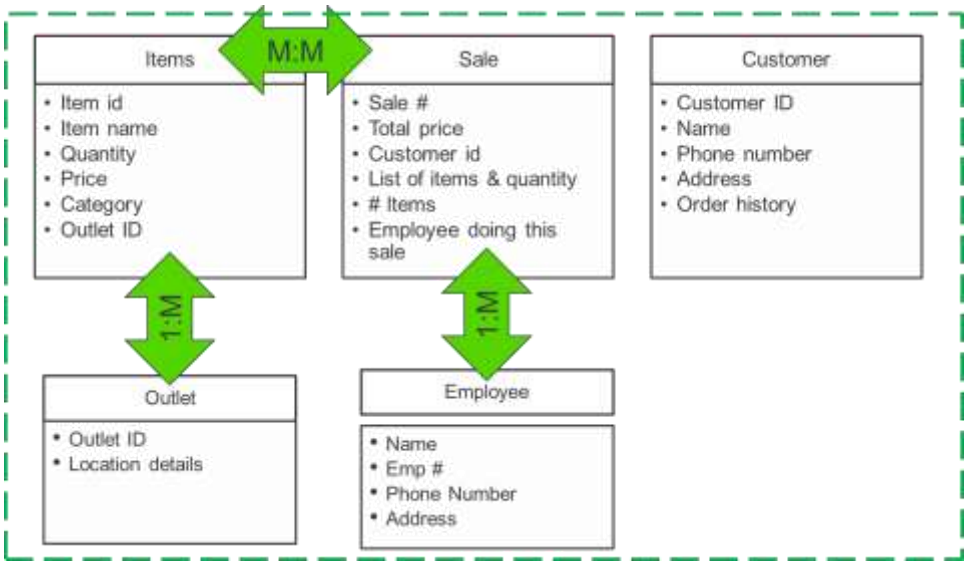
Dr Ganesh Neelakanta Iyer

© 2016 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

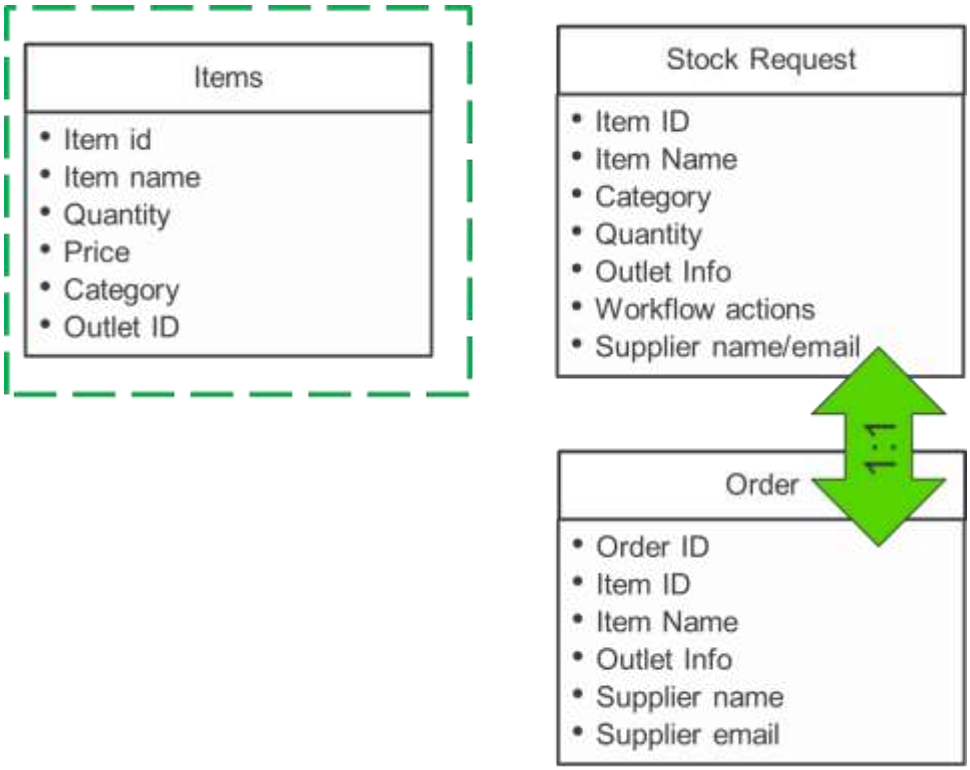
In this whitepaper, we describe one use case for understanding how to build a satellite application in Rollbase for complementing an existing OpenEdge application.

Problem Statement:

For an existing OpenEdge application point of sale system (e.g. Subway, Post Office), we want to build a Satellite Stock Replenishment Rollbase application for the items sold in the point of sale system. The schema of the existing Rollbase application looks as below:

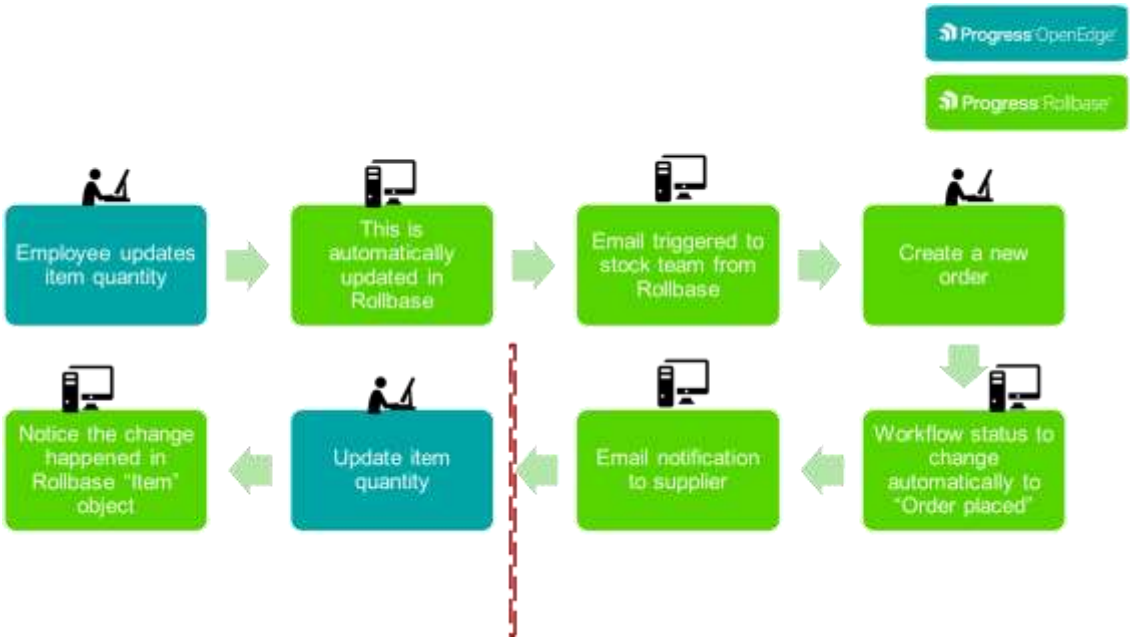


The stock replenishment app's schema is as below:

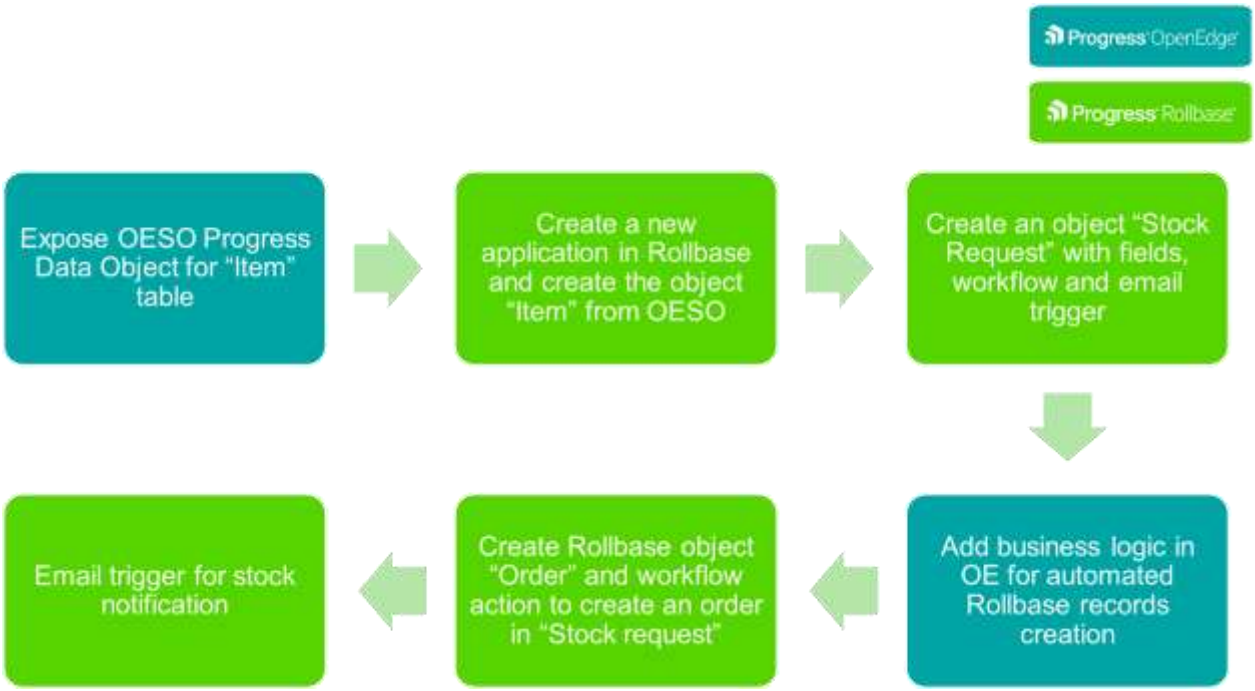


Note: The above schemas are just samples to understand the high level picture. We might use some different names/fields in the following example.

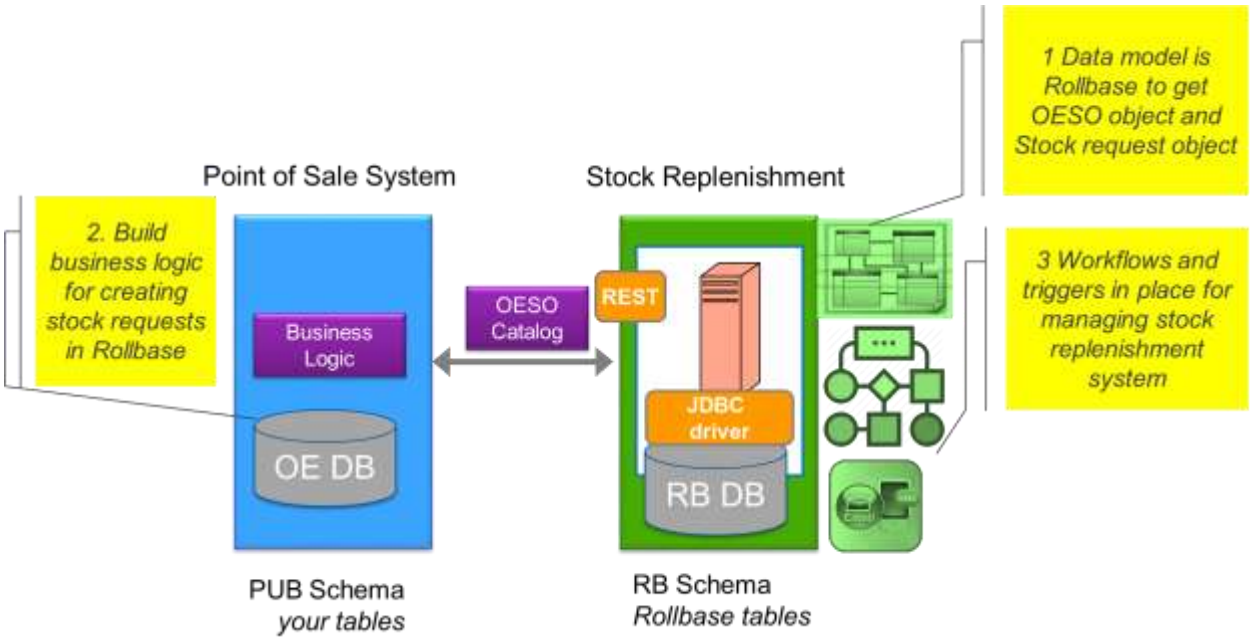
Application run time flow is illustrated below.



For the above application to be functioning, the development flow for the Satellite application is as illustrated below:



In the rest of this document, we are going to build the application for which the broad steps are illustrated below:



Pre-requisites:

OpenEdge Data object service is ready using table for the ITEMS and accessible using Catalog URI (JSON)

Kendo UI, which is acting as POS (point of sales system) is available and using OpenEdge Data object service (Using ITEMS table) in a browser.

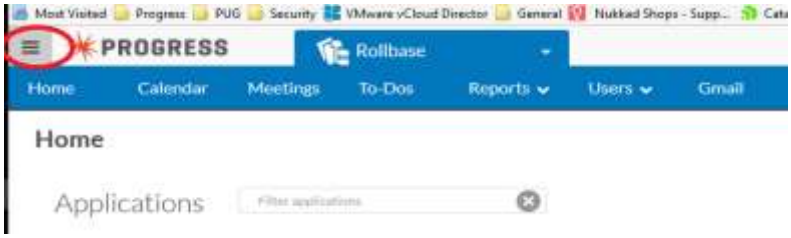
Drag a column header and drop it here to group by that column

ITEMID	SERIALNUMBER	NAME	QUANTITY	CATEGORY	SUBCATEGORY	OUTLETNAMEITEMS
1	Serial1	heritage	1000	Eatables	Cheese	Tewksbury, MA
2	Serial2	Heritage	1001	Eatables	Butter	Chelmsford, MA
3	Serial3	Smallboard	77	Boards	Billboard	Chelmsford, MA
4	Serial4	Knife	399	Household	Kn	Chelmsford, MA
5	Serial5	Heritage butter	99	Eatables	cake3	Chelmsford, MA
6	Serial6	Harry Potter	666	Books	Fiction	Tewksbury, MA

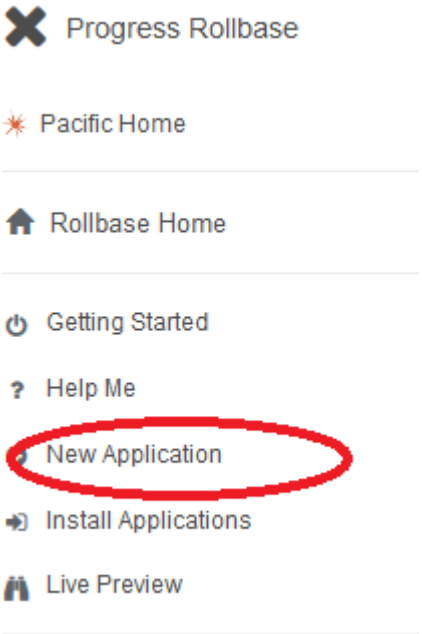
Start building the Rollbase application

Start creating Rollbase “New Application” using below option in Rollbase.

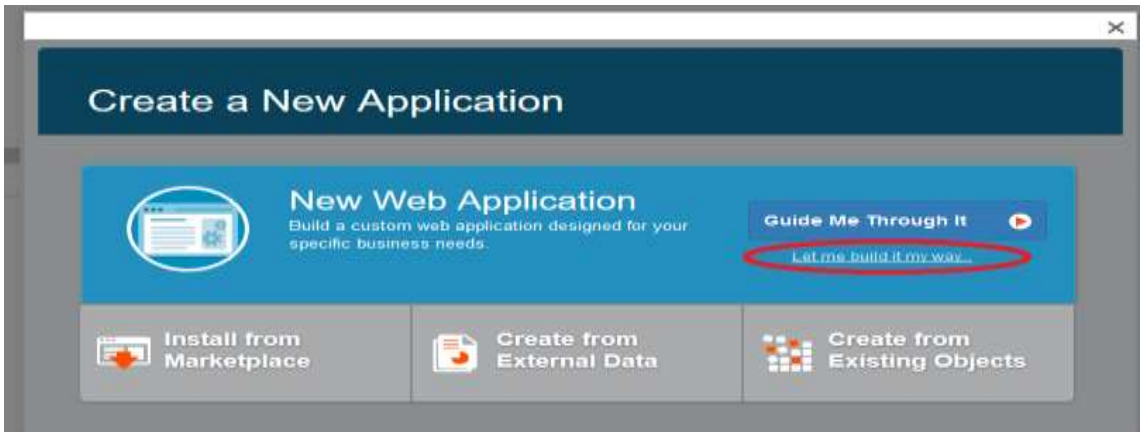
Click on hamburger menu



Choose “*New Application*” option.

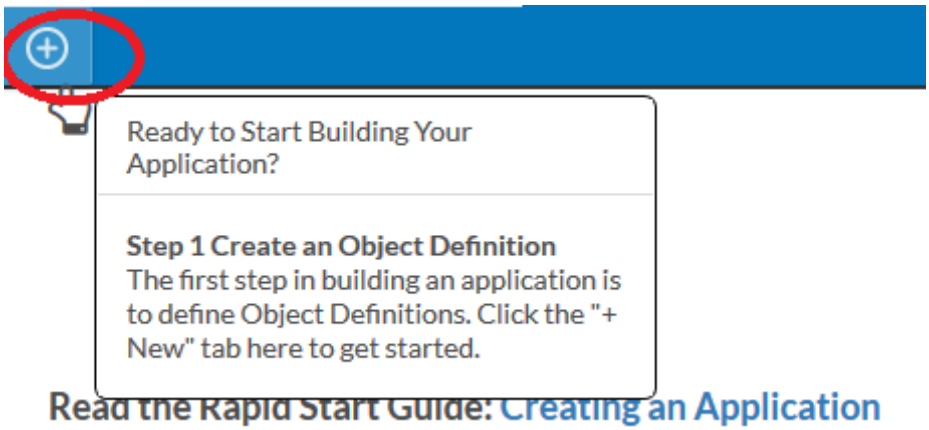


Create a New application dialog popup, select “*let me build it my way*” option.

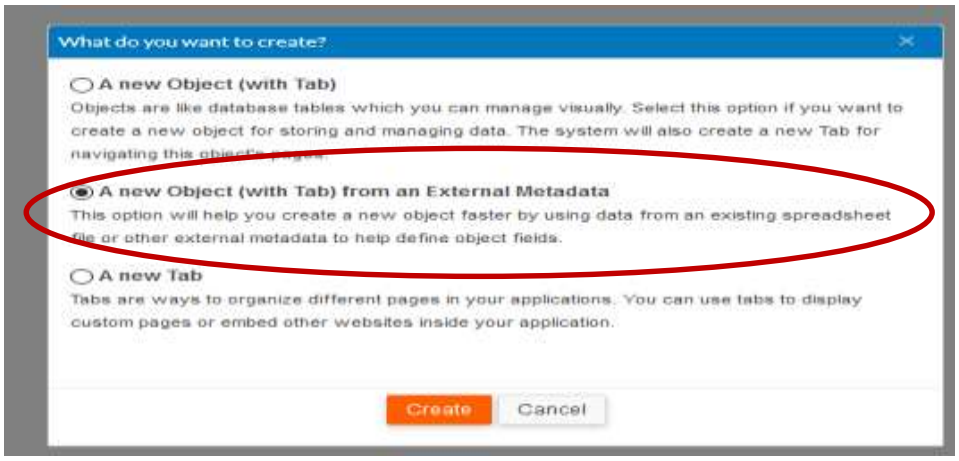


Fill up the Application Name (*Stock Replenishment*) then save. A new application is created.

Click *plus* icon



select a new object (with Tab) from external metadata in What do you want to create? dialog.



Click "*create*" button and select "*OpenEdge Service*" option.

Click on "*Browse*" button and provide the Catalog URI of existing OpenEdge Service.

Catalog URI: http://HOSTNAME:PORT/Items/static/ItemsService.json

Click on Open button and wait for a while until it loads.

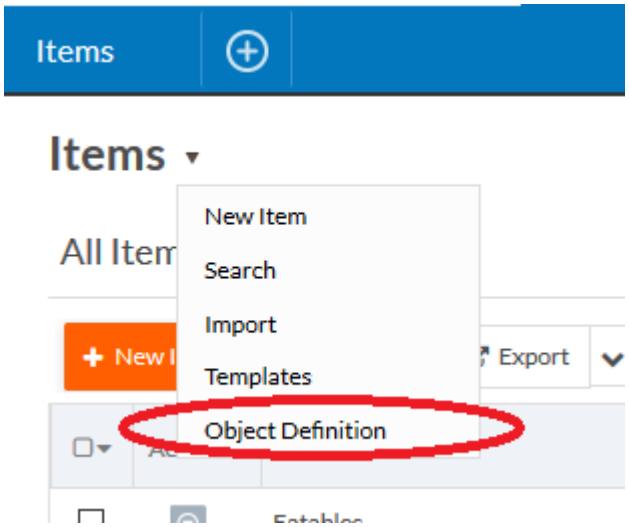
Provide a **service URI** as: http://HOSTNAME:PORT/<Service name>

Click on *NEXT* button until "*Create*" Button is visible and then click.

Now an OpenEdge object (aka OESO object) in Rollbase is created successfully.

Click on Return to <Application name> link at top right to access the created object.

Access the *object definition*



Click *Edit* option and navigate to “OpenEdge Service” section, set “*Clear List Cache After*” to 0 from 5 and save the object.

Click on Return to <Application name> link at top right to access the object.

Access the records in Kendo UI and cross check the records in OESO object and notice both are identical.

Start creating the Rollbase native object with below integration name.

Click PLUS icon and select “A new Object (with TAB)” and provide application name as below.

Object Name/ Singular Name	Integration Name
Stock Management	Stock_Management

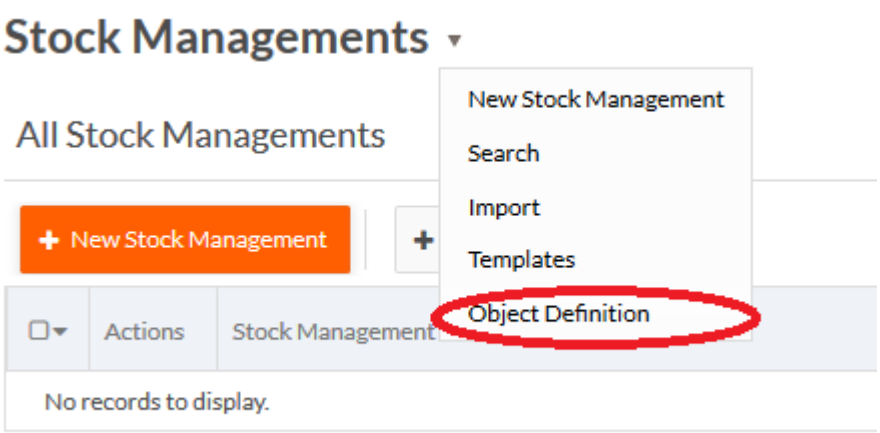
Integration Name: Rollbase identifies everything in Rollbase uniquely using the concept of Integration Name. Whenever you refer to anything in Rollbase (including app name, object name, field names, relationship names etc, you should always refer via the integration name.

Scroll down to navigate to “Advanced Attribute Description” under Object attributes.

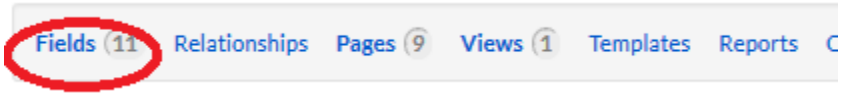
Select the Workflow checkbox.

Click on save and then following save button.

Navigate to object definition of Stock Managements and start creating the required fields as below.



Click on *Fields*.



Click on *New Field*, while creating the select All under views

Field Type	Field Name	Integration Name	Other details
Auto number	Stock Management ID	Stock_Management _ID_SM	Display Format as "SM#{YYYY}-{0000000}" Starting Number as " 1 "
Text	ITEM NAME	ITEM_NAME_PUG	
Text	QUANTITY	QUANTITY_PUG	
Text	CATEGORY	CATEGORY_PUG	
Text	SUBCATEGORY	SUBCATEGORY_PUG	
Text	OUTLET INFO	OUTLET_INFO_PUG	
Integer	ITEMID	ITEMID_PUG	
Picklist	SUPPLIER NAME	SUPPLIER_NAME_PUG	Provide the values as e.g. "Cargil Sysco"
Email	SUPPLIER EMAIL	SUPPLIER_EMAIL_PUG	

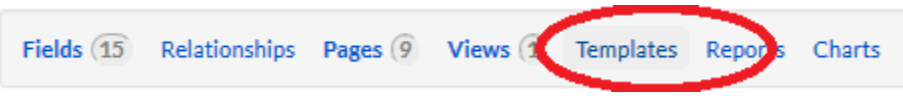
Navigate to the top of the page.

<u>Adding business logic for an email notification whenever a new record is created in this object</u>

It consists of two steps. Create an email template and then create an email trigger.

Creating a template:

Select *Templates* option



Click on *NEW EMAIL TEMPLATE* button.

Give the name as “New Stock request mail”, subject as “New Stock Request” and provide the mail template.

E.g. Mail template

Hi,

There is stock request raised for the item {!ITEM_NAME_PUG#value} under the category {!CATEGORY_PUG#value}. Please ship ASAP at {!OUTLET_INFO_PUG#value}.

Quantity requested is: {!QUANTITY_PUG}

Regards,

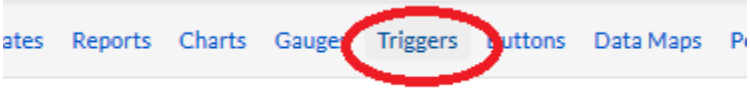
POS ({!OUTLET_INFO_PUG#value})

NOTE: The template tokens in the above sample (e.g. {!ITEMNAME_PUG#value}) needs to be modified as per the tokens in your application.

Then, save the mail template.

Creating an email trigger

Select *Triggers* option.



Click on *NEW TRIGGER* button to send an email, whenever we have a new stock request.

Select “*Send Email*” and click NEXT button.

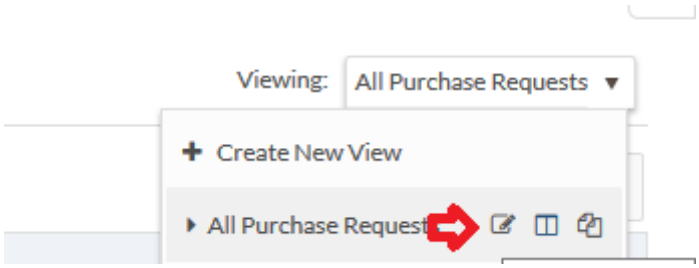
Check “After Update” under Timing section

Give the Trigger Name (Trigger mail for stock request), chose the available Email Template (i.e. Stock request mail)

Give the send to address and save.

Click on Return to <Application name> link at top right to access the object.

Click on viewing drop down and select filter option.



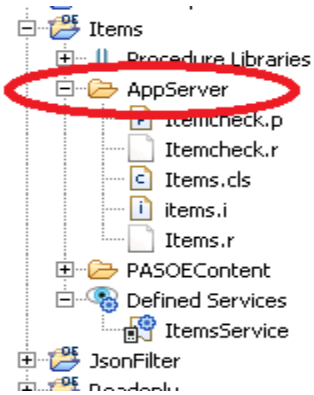
Remove the *createdby*, *createdat* etc, Include Workflow action, workflow status and save the view.

Adding required business logic in OpenEdge:

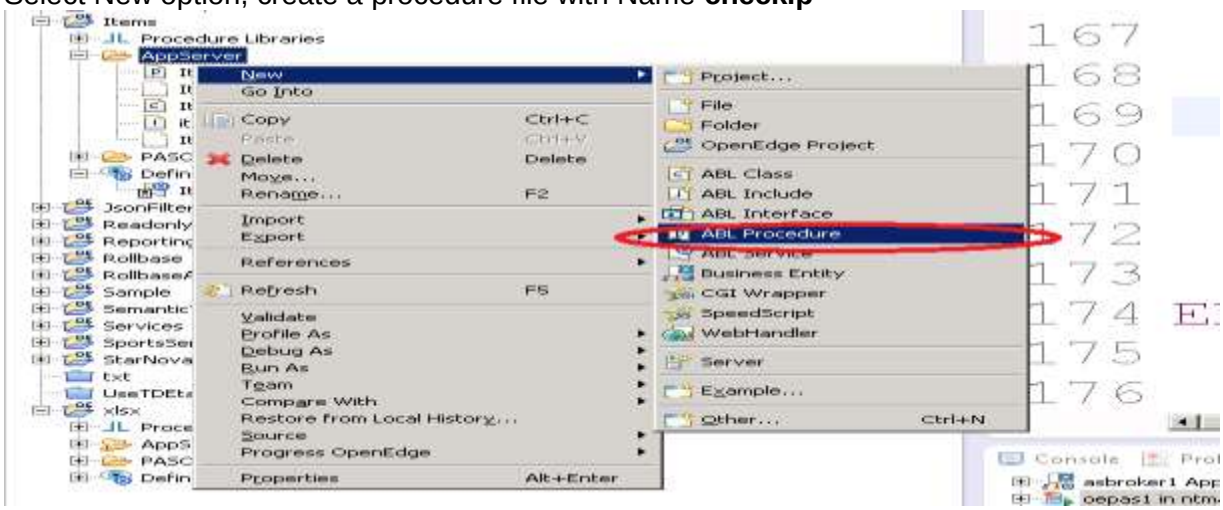
Requirement:

Access OpenEdge environment and start consuming the Rollbase rest API to get the login session followed by creation of record in Rollbase object, when Item Quantity goes less than 10.

Open the OpenEdge Developer Environment.
Expand the Item service project in development environment and right click on App server.



Select New option, create a procedure file with Name **check.p**



Copy the code in below block paste in created Itemcheck.p file and save.

Also make sure, highlighted code of lines are properly specified based on requirement.

```
/*------------------------------------------------------------------------
    File        : Itemcheck.p
    Purpose  :

    Syntax     :

    Description : This procedure file contains the logic to create a new record in
                  Rollbase application, whenever the  items quantaty goes below 10.

    Author(s)   : Ganesh Cherivirala
    Created      : Tue Jun 28 12:35:36 PDT 2016
    Notes        : There are two Rollbase APIs used in this code. Provide the parameters
                   as appropriate. They are highlighted.
  ----------------------------------------------------------------------*/


/* ***************************  Definitions  ************************** */

BLOCK-LEVEL ON ERROR UNDO, THROW.

/* ********************  Preprocessor Definitions  ******************** */


/* ***************************  Main Block  *************************** */
USING OpenEdge.Core.WidgetHandle FROM PROPATH.
USING OpenEdge.Net.HTTP.ClientBuilder FROM PROPATH.
USING OpenEdge.Net.HTTP.IHttpClient FROM PROPATH.
USING OpenEdge.Net.HTTP.IHttpRequest FROM PROPATH.
USING OpenEdge.Net.HTTP.IHttpResponse FROM PROPATH.
USING OpenEdge.Net.HTTP.RequestBuilder FROM PROPATH.
USING Progress.Json.ObjectModel.JsonObject FROM PROPATH.
USING Progress.Json.ObjectModel.ObjectModelParser FROM PROPATH.
USING Progress.Lang.Object FROM PROPATH.


DEFINE VAR      oClient      AS IHttpClient.
DEFINE VAR      oReq         AS IHttpRequest.
DEFINE VAR      oResp        AS IHttpResponse.
DEF    VAR      httpURL      AS CHAR.
DEFINE VARIABLE hXmlDoc      AS HANDLE           NO-UNDO.
```

```
DEF    VAR      oEntity      AS Object            NO-UNDO.
DEFINE VARIABLE hDataSet     AS HANDLE            NO-UNDO.
DEFINE VARIABLE hDataSet2    AS HANDLE            NO-UNDO.
DEFINE VARIABLE h_SessionId  AS CHAR              NO-UNDO.
DEFINE VARIABLE Qty          AS INTEGER           NO-UNDO.
DEF    VAR      l1           AS LONGCHAR.
DEF    VAR      JsonObj      AS JSOnObject.
DEFINE VARIABLE myParser     AS ObjectModelParser NO-UNDO.
DEFINE VARIABLE rbSessionUrl AS CHARACTER         INITIAL
"https://www.rollbase.com/rest/api/login?loginName=<UserName>&password=<Password>".

/* change the "i" file name to reflect the available I file*/
{"items.i"}

DEFINE INPUT PARAMETER DATASET FOR dsITEMS.
DATASET dsITEMS:WRITE-JSON ("file","C:/temp1.json",TRUE).


FOR EACH Itemscheck:
   DISPLAY QUANTITY.
   MESSAGE "in checkitem - " QUANTITY VIEW-AS ALERT-BOX.
   IF QUANTITY < 10 THEN
      // Dynamic dataset handles used for reading the XML
      CREATE DATASET hDataSet.
      CREATE DATASET hDataSet2.

      // JsonObject to store the XML to JSON
      JsonObj = NEW JsonObject().

      // ObjectModelParser to parse the longchar while casting it to JSONObject
      myParser = NEW ObjectModelParser().

      //base HTTP URL while performing GET operation
   //httpURL =
"https://www.rollbase.com/rest/api/login?loginName=user1_prod&password=Jv02Eo02".
   httpURL = rbSessionUrl.

//Perform a GET Request
   oClient = ClientBuilder:Build()
         :Client.

   oReq = RequestBuilder
         :Get(httpURL)
         :Request.

   oResp = oClient:Execute(oReq).

   oEntity = oResp:Entity.
```

```
//If the response entity is XML then read individual elements
   IF TYPE-OF(oEntity, WidgetHandle) THEN
   DO:
      hXmlDoc = CAST(oEntity, WidgetHandle):Value.
      hDataSet:READ-XML("handle",hXmlDoc,"empty",?,?,?).

      hDataSet:WRITE-JSON ("longchar",l1,TRUE).

      JsonObj = CAST(myParser:Parse(l1),JsonObject).

    //Check if the status is OK. If yes, then look for the sessionId

IF(JsonObj:GetJsonObject("NewDataSet"):GetJsonArray("resp"):GetJsonObject(1):GetCharact
er("status") = "ok") THEN

         h_SessionId =
JsonObj:GetJsonObject("NewDataSet"):GetJsonArray("resp"):GetJsonObject(1):GetCharacter(
"sessionId").
      ELSE
         RETURN ERROR "Failed while performing GET operation ".

   END.




//Append the dynamic sessionId to the httpURL before performing the POST operation
   MESSAGE "In Side IF block - " QUANTITY VIEW-AS ALERT-BOX.


httpURL = "https://www.rollbase.com/rest/api/createRecord?sessionId="
      + h_SessionId
      + "&objName=Stock_Management1&useIds=false&OUTLET_INFO_PUG="
      + String (OUTLETNAMEITEMS)
      + "&QUANTITY_PUG="
      + String(QUANTITY)
      + "&CATEGORY_PUG="
      + String(CATEGORY)
      + "&ITEM_NAME_PUG="
      + String(NAME).

   oClient = ClientBuilder:Build()
         :Client.

   oReq = RequestBuilder
         :Post(httpURL,JsonObj)
         :Request.


   oResp = oClient:Execute(oReq).
```

```
   oEntity = oResp:Entity.


//If the response entity is XML then read individual elements
   IF TYPE-OF(oEntity, WidgetHandle) THEN
   DO:

      hXmlDoc = CAST(oEntity, WidgetHandle):Value.
      hDataSet2:READ-XML("handle",hXmlDoc,"empty",?,?,?).
      hDataSet2:WRITE-JSON ("longchar",l1,TRUE).

      JsonObj = CAST(myParser:Parse(l1),JsonObject).

//Check if the status is OK. If yes, then extract the individual elements

IF(JsonObj:GetJsonObject("NewDataSet"):GetJsonArray("resp"):GetJsonObject(1):GetCharact
er("status") = "ok") THEN
      DO:
         MESSAGE "id is - "  +
JsonObj:GetJsonObject("NewDataSet"):GetJsonArray("resp"):GetJsonObject(1):GetJsonArray
("data"):GetJsonObject(1):GetCharacter("id")  VIEW-AS ALERT-BOX.
         MESSAGE "objName is - "  +
JsonObj:GetJsonObject("NewDataSet"):GetJsonArray("resp"):GetJsonObject(1):GetJsonArray
("data"):GetJsonObject(1):GetCharacter("objName") VIEW-AS ALERT-BOX.
         MESSAGE "Msg is - "  +
JsonObj:GetJsonObject("NewDataSet"):GetJsonArray("resp"):GetJsonObject(1):GetJsonArray
("data"):GetJsonObject(1):GetCharacter("Msg") VIEW-AS ALERT-BOX.

      END.

      ELSE
      DO:
         MESSAGE "Post operation failed for the httpURL "  + httpURL.
         RETURN ERROR "Post failed".
      END.
   END.

   CATCH e AS Progress.Lang.Error :
      DEFINE VARIABLE iLoop AS INTEGER NO-UNDO.
      DO iLoop = 1 TO e:NumMessages:
         PUT UNFORMATTED
            'ERROR: ' e:GetMessage(iLoop) SKIP.
      END.


   END CATCH.

END.
```

Till now, we have written the code in procedure file to get the Rollbase user session followed by record creation in stock request object in Rollbase using REST APIs. It's time to invoke the procedure file.

As we all know that the BE/.cls is the acting as an interface for the service for client communication (Kendo UI). My interface have the submit method, I will call the procedure file using RUN comment with input as DATASET.
NOTE: If there is no submit method in BE, we need to call procedure file in the methods (e.g. Update, create).

Open the *.cls file under app server and update the submit method to call above created file.

```
/*

RUN itemcheck.p (INPUT DATASET dsITEMSCHECK BY-REFERENCE).

*/
```

Wait until changes are published to server. Navigate to servers view and confirm the server status is published.

Access OpenEdge Item service in the Kendo UI and update any item quantity to have less than 10.

Access the Purchase request object in Rollbase and notice a new record is created with the details of quantity, category, sub category, soon.

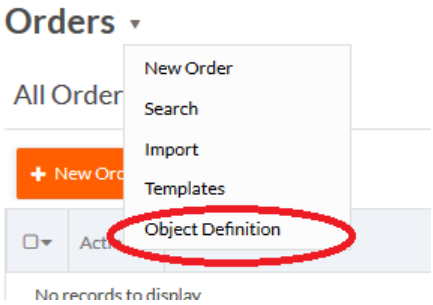
Additional Business Logic in Rollbase

Great! Record has been created automatically. As a next step, we would like to add some more business logic in Rollbase to help the stock replenishment team to create an order in an automatic way and notify the supplier about the same.

Click PLUS icon and select “A new Object (with TAB)” and provide details as below, Save

Object Name/ Singular Name	Integration Name
Order	Order_PUG

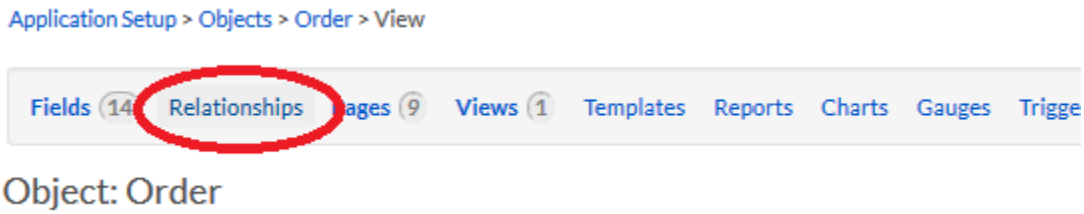
Navigate to object definition and start creating the required fields.



Field Type	Field Name	Integration Name	Other details
AUTO NUMBER	ORDER ID	ORDER_ID_PUG	Display Format as “OR#-{0000000}”. Starting Number as “ 1 “.
Text	ORDER NAME	ORDER_ NAME_PUG	
Text	ORDER QUANTITY	ORDER_QUANTITY_PUG	
Text	SUPPLIER NAME	SUPPLIER_NAME_PUG_OR	
Email	SUPPLIER EMAIL	SUPPLIER_EMAIL_PUG_OR	

Create *one order one purchase request Relationship*.

Navigate to top and select the relationship.



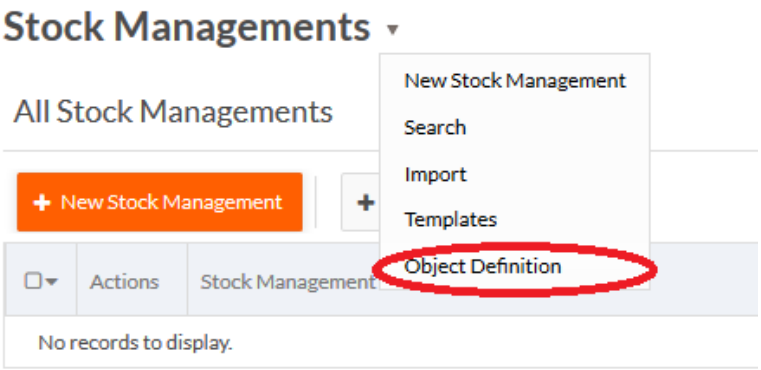
Click New Relationship
Select Purchase Request radio button and then click on NEXT.
Select “one order one Stock Management” in Cardinality and save.

In order to create an Stock request based on user click, we need have a trigger. With little coding.

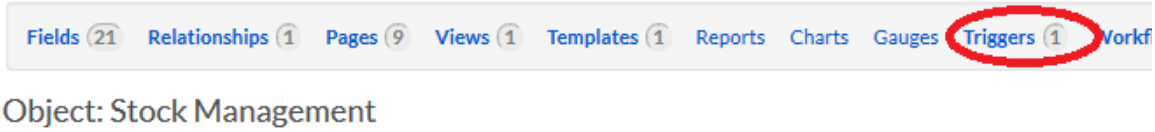
Click on Return to <Application name> link at top right to access the object.

Click on Stock Management object

Click Object definiation.



Navigate to top of the page and click on TRIGGERS option.



Click on NEW TRIGGER. Select **OBJECT SCRIPT** Radio option, and click NEXT. Give the Name as “Place the Order”

Use below code in FORMULA RETUN FIELD and save

```
var x = new Array();

x["Order_Quantity_PUG"]="{!QUANTITY_PUG#value}";

x["Order_Name_PUG"]="{!ITEM_NAME_PUG#value}";

x["SUPPLIER_NAME_PUG_OR"]="{!SUPPLIER_NAME_PUG#value}";

x["SUPPLIER_EMAIL_PUG_OR"]="{!SUPPLIER_EMAIL_PUG}";

var newId = rbv_api.createRecord("Order_PUG1", x);

rbv_api.attach("R248177719", "Stock_Management1", {!id}, "Order_PUG1", newId);
```

NOTE: The template tokens in the above code (e.g. "Stock_Management_ID_OR", "{!Stock_Management_ID_SM}", R247279297 etc) needs to be modified as per the tokens in your application. Details of how to use these APIs can be found here.

In this we are setting the quantity of item as the quantity given while requesting the stock.. while placing the order along with Supplier name, Email.
Creating a new record in ORDER OBJECT (rbv_api.createRecord).
later we are attaching the created object to Stock management object to view.

Navigate to top of the page and click on “Workflow *STATUSES”*

Click Edit and modify the Status Name as “Stock Request made” from DEFAULT.

SAVE. Click **New Status** button and create another status “*Order placed*”. Navigate to top of the page, Click WORKFLOW ACTIONs option

Click on NEW WORKFLOW ACTION.

Select **Run Trigger** radio button and click on NEXT button.

Give the name as “Place Order”,

Select “Change status to’ as “Order request made’,

Select NONE radio option for USE WEB PAGE

Select “place the Order” trigger in the Available Triggers and move to Selected Triggers the save the action.

Navigate to top of the page and click on “Workflow *STATUSES”*

Click Edit for the Stock Requested.
Select “place the Order” trigger in the Available action and move to Selected actions.
SAVE.

Access the item service in Kendo UI (which we are treating as POS). Update the existing record quantity t have less than 10 and update.

As per the todays use-case, A new email reach with subject <Mail subject> e.g. Stock request. A new record as been created under purchase request RB object with an option to create the order.

Access the order object and check for record count.

Back to purchase request and click on Create Order with success message.

Access order object and check for the new record.

We assume vender delivered the order quantity at inventory and quantity is updated accordingly.

We can update the quantity in Kendo UI and same will be reflected item object in Rollbase.

Summary:

- Rollbase is an ideal platform for creating enterprise productivity applications rapidly with low code
- Ideal for satellite applications for existing OpenEdge applications
- The power of Rollbase is 80% mouse clicks and 20% coding for building any enterprise productivity application

About Progress

Progress (NASDAQ: PRGS) is a global leader in application development, empowering the digital transformation organizations need to create and sustain engaging user experiences in today's evolving marketplace. With offerings spanning web, mobile and data for on-premise and cloud environments, Progress powers startups and industry titans worldwide, promoting success one customer at a time. Learn about Progress at www.progress.com or 1-781-280-4000.

Worldwide Headquarters

Progress, 14 Oak Park, Bedford, MA 01730 USA Tel: +1 781 280-4000 Fax: +1 781 280-4095 On the Web at: www.progress.com Find us on facebook.com/progresssw twitter.com/progresssw youtube.com/progresssw For regional international office locations and contact information, please go to www.progress.com/worldwide

Progress, Progress Rollbase and Telerik are trademarks or registered trademarks of Progress Software Corporation and/or one of its subsidiaries or affiliates in the U.S. and/or other countries. Any other trademarks contained herein are the property of their respective owners. © 2016 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved. Rev 16/05 | 60322-0091