How to create custom workflow in ax 2012

Contents

[Create table (A\_StudentDemoTable). 2](#_Toc508880709)

[Create Base Enums Like CustomEnum and StudentResult. 3](#_Toc508880710)

[Create Form (Student Entry Form). 3](#_Toc508880711)

[Create ListPage (A\_StudentDemoListPage). 4](#_Toc508880712)

[Workflow Configuration 4](#_Toc508880713)

[Create a Workflow Category 4](#_Toc508880714)

[Create a Query 5](#_Toc508880715)

[Create a Workflow Type 6](#_Toc508880716)

[Enable Workflow on a Form 7](#_Toc508880717)

[Create a Submit to Workflow Class 9](#_Toc508880718)

[Create a Workflow Approval 10](#_Toc508880719)

[Create Event Handlers 11](#_Toc508880720)

[Cancelled Method: 11](#_Toc508880721)

[Completed Method: 11](#_Toc508880722)

[Started Method: 11](#_Toc508880723)

[Add Element Level Event Handlers 12](#_Toc508880724)

[Canceled Method: 12](#_Toc508880725)

[changeRequested Method: 12](#_Toc508880726)

[Completed Method: 12](#_Toc508880727)

[Returned Method: 12](#_Toc508880728)

[Started Method: 12](#_Toc508880729)

[Author a Workflow 12](#_Toc508880730)

[Testing Custom Workflow 17](#_Toc508880731)

[Complete Code 23](#_Toc508880732)

[A\_StudentDemoTable 23](#_Toc508880733)

[StudentEntryForm 24](#_Toc508880734)

[StudentResultSubmitManager(Class) – Main Method 24](#_Toc508880735)

[StudentResultEventHandler (Class) – Completed Method 25](#_Toc508880736)

[StudentResultEventHandler (Class) – Canceled Method 25](#_Toc508880737)

[StudentResultEventHandler (Class) – Started Method 26](#_Toc508880738)

[StudentDemoApprovalsEventHandler (Class) – returned Method 26](#_Toc508880739)

[StudentDemoApprovalsEventHandler (Class) – Canceled Method 26](#_Toc508880740)

[StudentDemoApprovalsEventHandler (Class) – ChangedRequested Method 26](#_Toc508880741)

[StudentDemoApprovalsEventHandler (Class) – Completed Method 27](#_Toc508880742)

[StudentDemoApprovalsEventHandler (Class) – Started Method 27](#_Toc508880743)

[CODE FOR RESUBMIT 27](#_Toc508880744)

[StudentDemoApprovalsApprovalEle (Project) 29](#_Toc508880745)

[Declaration Method 30](#_Toc508880746)

[DialogOK 30](#_Toc508880747)

[Init 31](#_Toc508880748)

[ParmCustInvoicetableId 31](#_Toc508880749)

[ParmMenuItemName 32](#_Toc508880750)

[ParmSubmit 32](#_Toc508880751)

[ParmWorkflowComment 32](#_Toc508880752)

[ParmWorkflowControlContext 32](#_Toc508880753)

[ParmWorkflowTemplateName 32](#_Toc508880754)

[ParmWorkflowTableVersion 33](#_Toc508880755)

[ParmWorkflowWorkItemtable 33](#_Toc508880756)

[Resubmit 33](#_Toc508880757)

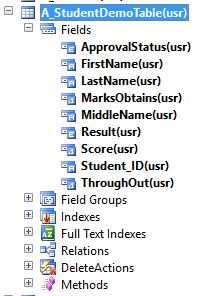
[Submit 33](#_Toc508880758)

[Construct 34](#_Toc508880759)

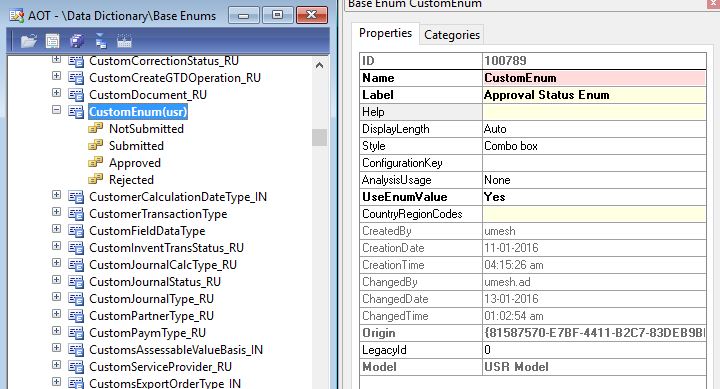
[Main 34](#_Toc508880760)

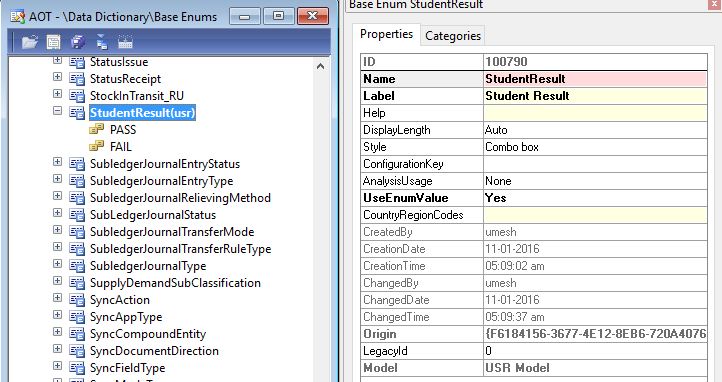
## Create table (A\_StudentDemoTable).

Add Filed Student\_ID (String), FirstName (String), LastName (String), MiddleName (String), Result (Enum), Score (Real), ApprovalStatus (Enum), MarksObtains (Real), ThroughOut (Real). Set ApprovalStatus EnumType CustomEnum and Result to StudentResult.



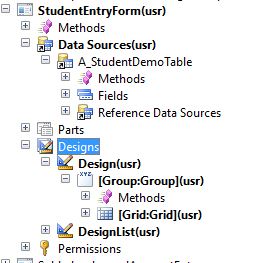
## Create Base Enums Like CustomEnum and StudentResult.





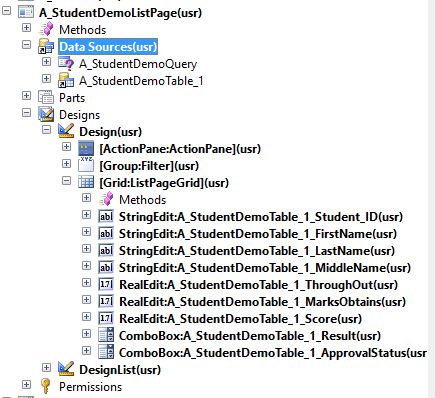
## Create Form (Student Entry Form).

Bind Created table in DataSource. Create Grid Drag and Drop Fields in Grid.



## Create ListPage (A\_StudentDemoListPage).

* For that you have to Create Query after bind Query in ListPage DataSource.
* Add fields in Grid.
* Create Button to Open Form (StudeEntryForm) to fill Data.



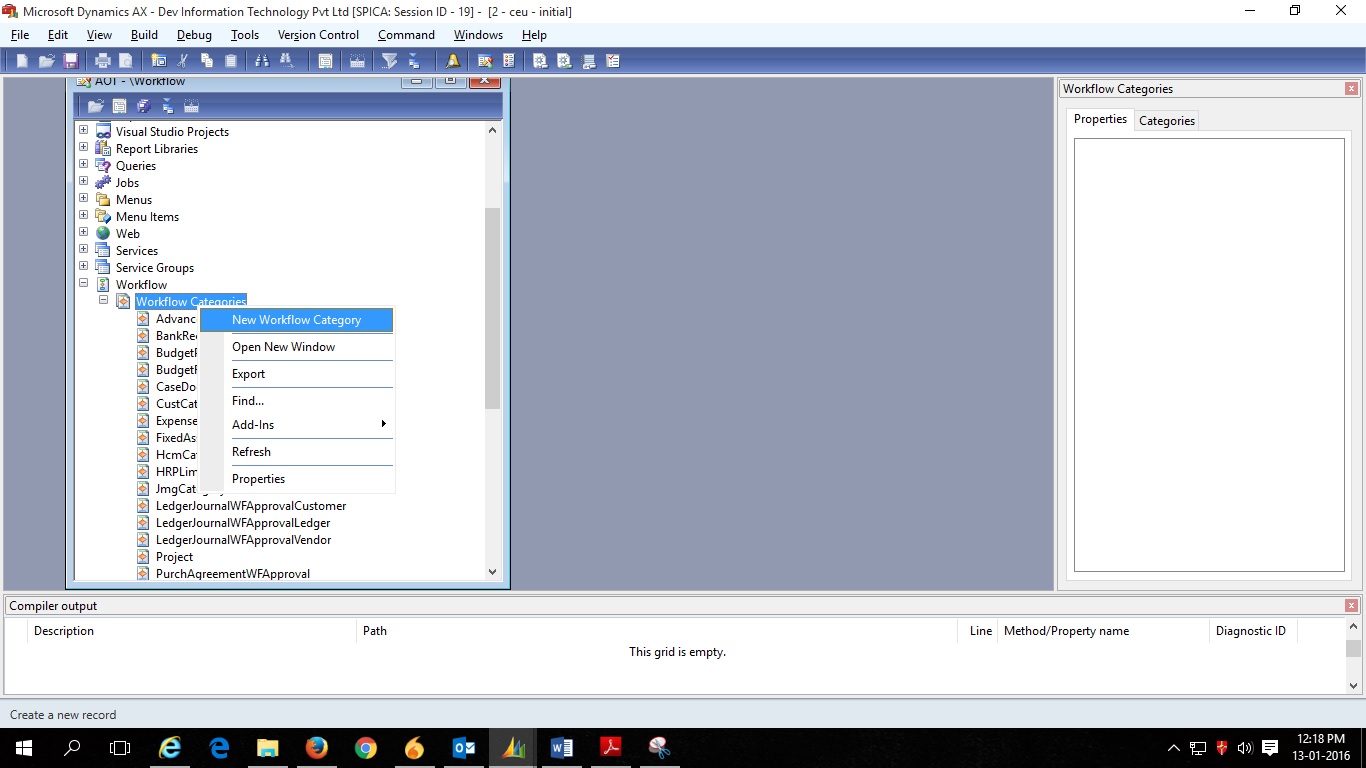
# Workflow Configuration

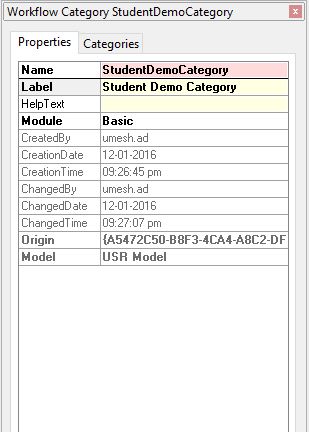
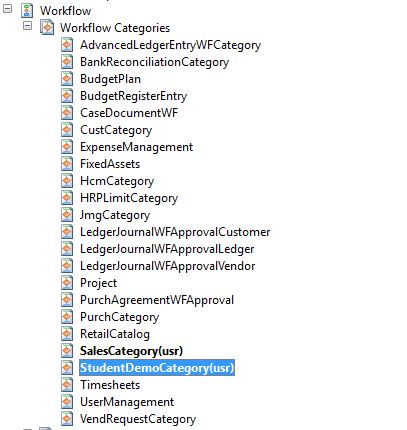
There are three batch jobs that manage workflow processing. These jobs are all run on the Application Object Server (AOS) using the Batch system. To set this up, run System Administration > Setup > Workflow > Workflow infrastructure configuration. Enter batch groups for each process. Batch groups can be used to control which AOS instance each workflow batch job is run on.

## Create a Workflow Category

Open the AOT.

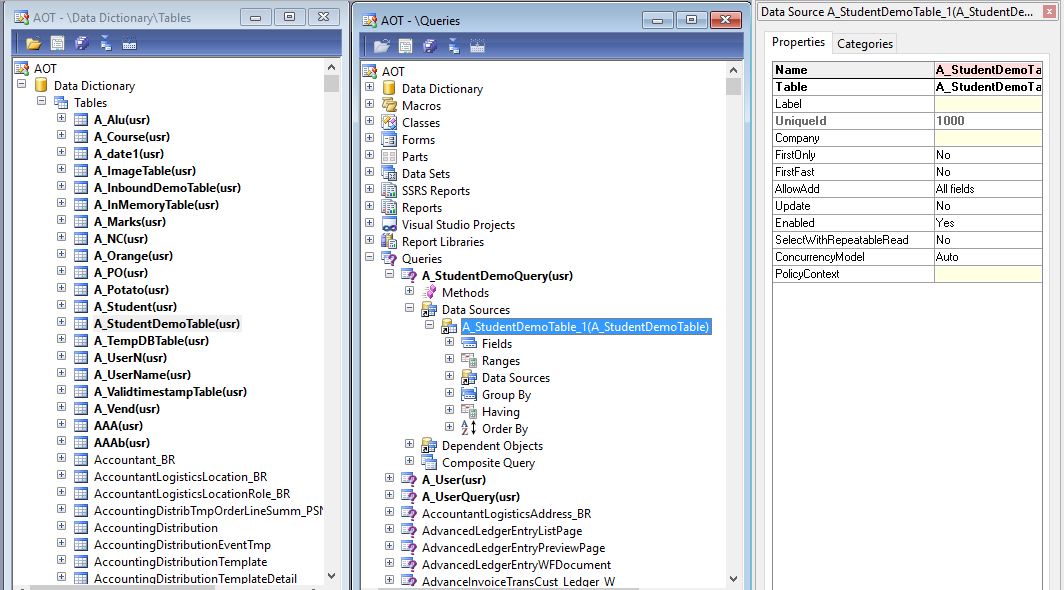
* Expand the Workflow node.
* Right-click on the Workflow Categories node and select New Workflow Category. A new Workflow category called Workflow Category1 will be created.
* Right-click on the newly created workflow category and select Properties.
* Change the name property to StudentDemoCategory.
* Change the label property to Student Demo Category.
* Change the Module property to Basic.
* Right-click on the newly created workflow category and select save.





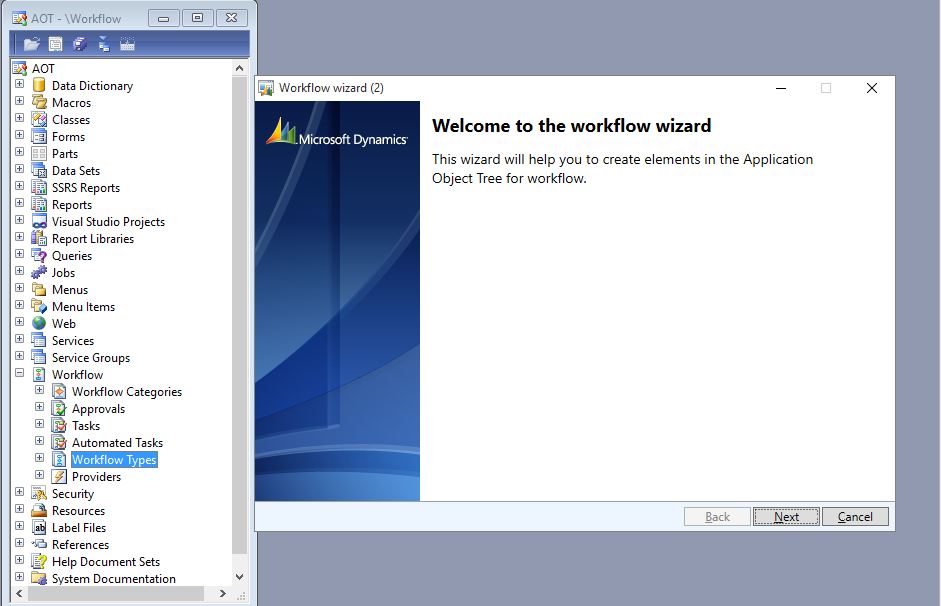
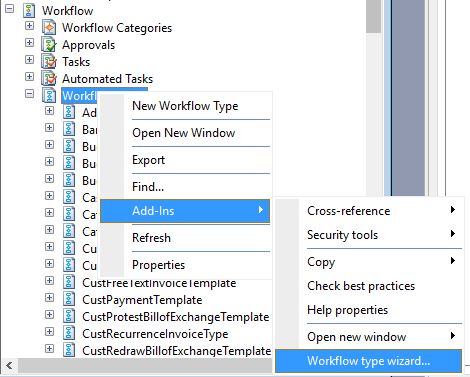
## Create a Query

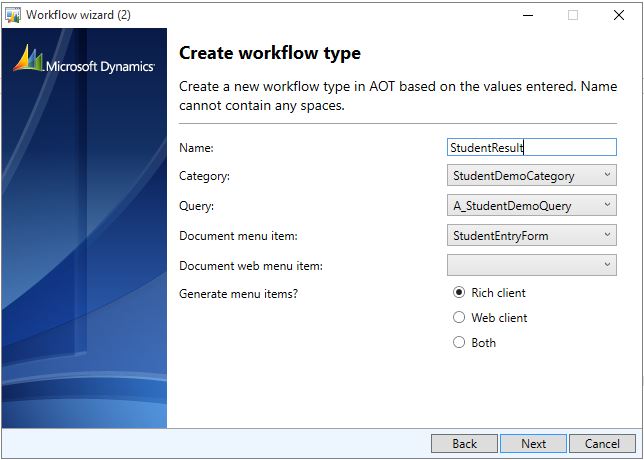
* Open the AOT.
* Right-click on the Queries node and select New Query.
* Rename the query to A\_StudentDemoQuery.
* Expand the newly created query.
* Open another AOT window.
* Expand Data Dictionary > Tables.
* Find the table A\_StudentDemoTable.
* Drag the A\_StudentDemoTable table to the Data Sources node of the A\_StudentDemoQuery query.
* Expand the A\_StudentDemoTable\_1 node
* Right-click on the Fields node and select Properties.
* Set the Dynamics property to Yes and close the property sheet.
* Right click on the A\_StudentDemoQuery query and select Save.

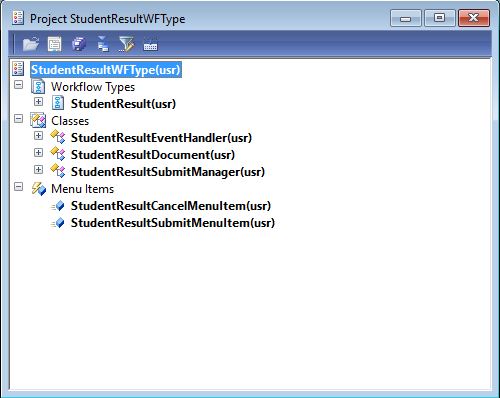


## Create a Workflow Type

* Open the AOT.
* Expand the Workflow node.
* Right-click on the Workflow Types node and select Add-ins > Workflow type wizard.
* Click Next.
* Enter StudentResult in the name.
* Enter StudentDemoCategory in the Category.
* Enter A\_StudentDemoQuery in the query.
* Enter StudentEntryForm in the Document menu item.
* Click Next.
* Click Finish. A development project with a number of newly created elements will be displayed.



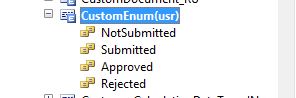


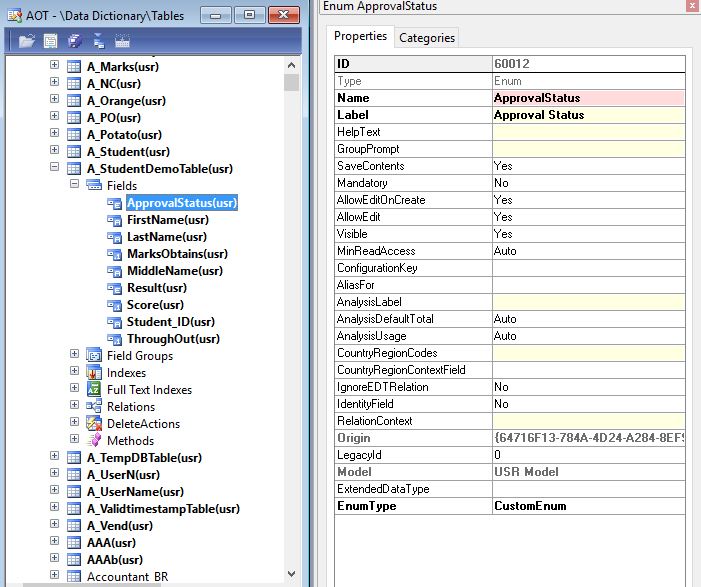


## Enable Workflow on a Form

(This Step is Not Necessary if you have Added ApprovalStatus Enums Fileld in Your table)

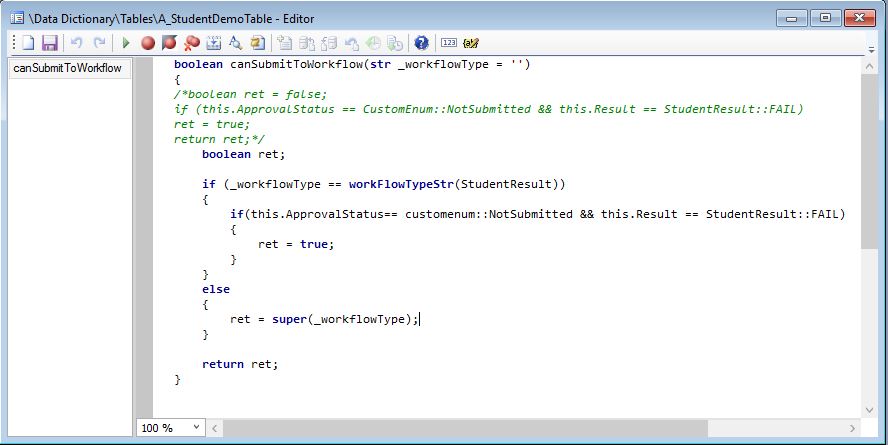
1. Add a WorkflowState Field

* Open the AOT.
* Expand Data Dictionary.
* Right-click on Base Enums and select New Base Enum.
* Rename the new Enum to CustomEnum
* Add four elements to the Enum called NotSubmitted, Submitted, Approved, Rejected.
* Expand Tables > A\_StudentDemoTable.
* Right-click on Fields and select New > Enum.
* Right-click on the newly created field and select Properties.
* Change the Name property to ApprovalStatus.
* Change the EnumType property to CustomEnum.
* Right-click on the A\_StudentDemoTable node and select save.

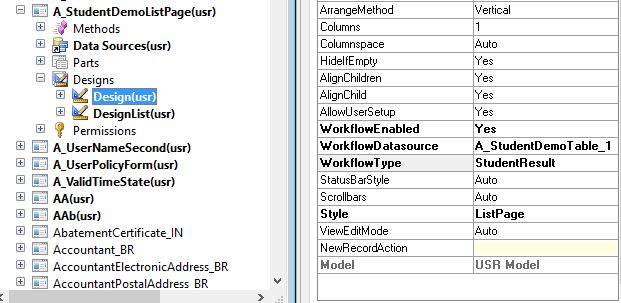


1. Enable Workflow on the Form

* Open the AOT.
* Expand Tables > A\_StudentDemoTable.
* Create a new method and add the method in the following code.

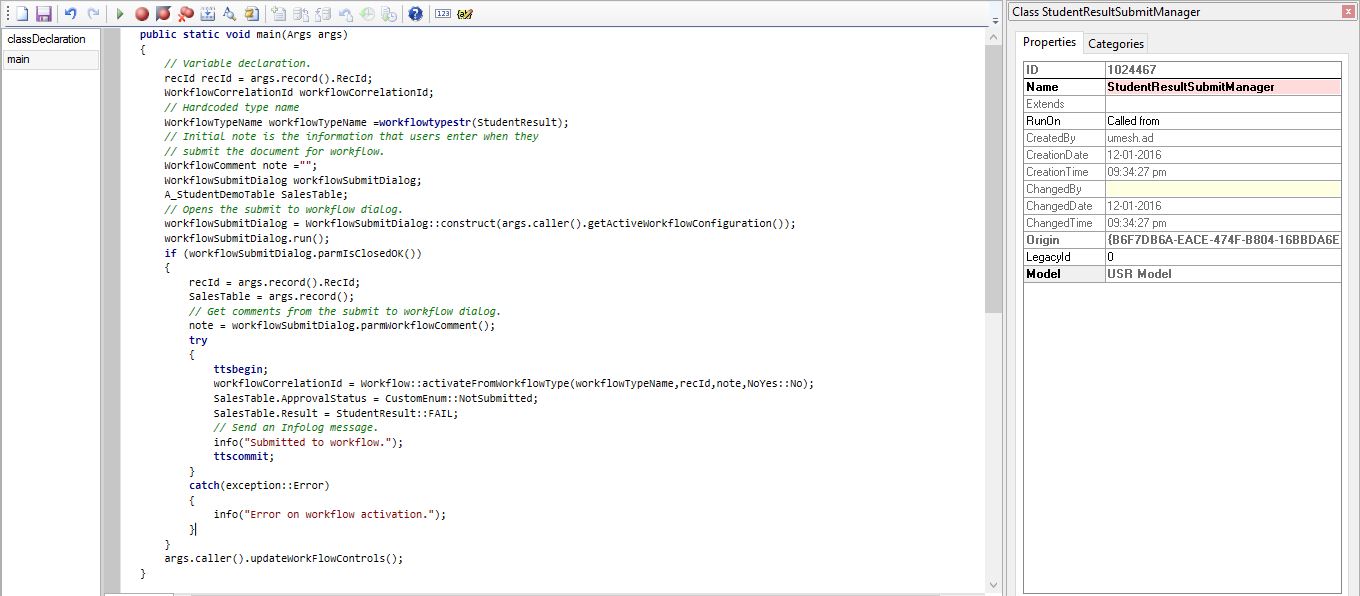


* Save the changes made to the table.
* Expand Forms > A\_StudentDemoListPage > Designs.
* Right-click on the design node and select Properties.
* Change the WorkflowEnabled property to Yes
* Change the WorkflowDatasource property to A\_StudentDemoTable\_1.
* Change the WorkflowType property to StudentResult.
* Save your changes to the form.



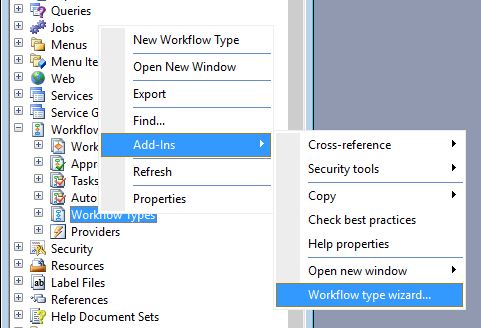
## Create a Submit to Workflow Class

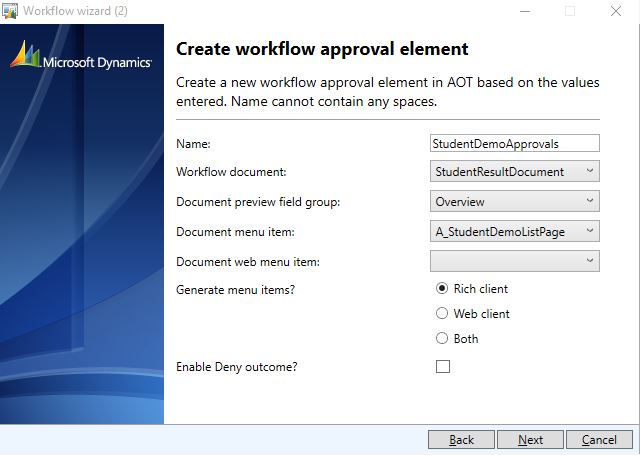
* Press Ctrl+Shift+P to open the development projects window.
* Expand Private.
* Double-click on StudentResultWFType development project.
* In the development project find the class StudentResultSubmitManager.
* Create a new method called submit or you Can call in Main Method, and copy the following code for this method.

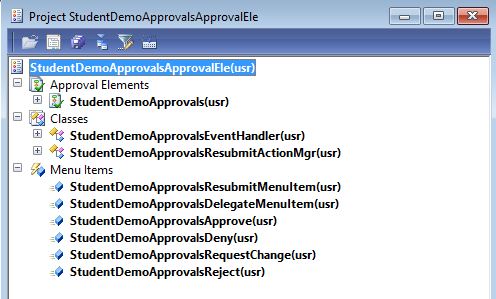


* Modify the code for the main method as shown in the following code.
* Press F8 to save and compile the code.
* Find the menu item StudentResultSubmitMenuItem.
* Change the Label property to Submit.
* Right-click on the StudentResultSubmitMenuItem menuitem and select Save.

## Create a Workflow Approval

* Open the AOT.
* Expand the Workflow node.
* Right-click on Approvals and select Add-ins > Approval wizard.
* Click Next.
* Enter StudentDemoApprovals in Name.
* Enter StudentResulDocument in Workflow document.
* Enter Overview in Document preview field group.
* Enter A\_StudentDemoListPage in Document menu item.
* Click Next.
* Click Finish. A development project with a number of newly created elements is displayed.
* Drag StudentDemoApprovals approval to the Supported elements node on the StudentResult workflow type.
* Save your changes to the StudentResult workflow type

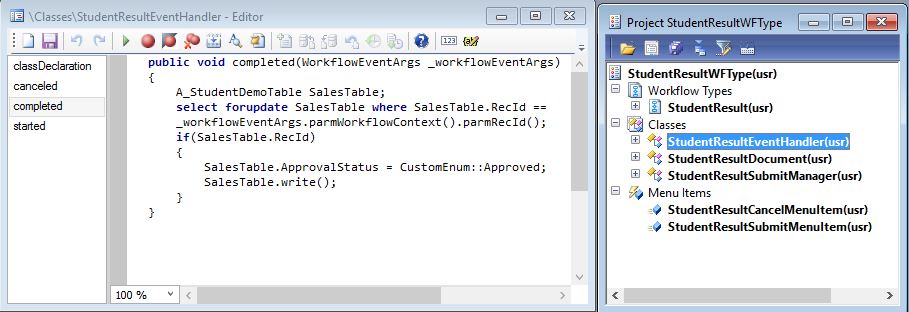




## Create Event Handlers

1. Write code for Event Handler

* Press Ctrl+Shift+P to open the development projects window.
* Expand Private.
* Double-click on StudentResultWFType development project.
* In the development project find the class StudentResultEventHandler.
* The following code needs to be added to the completed method of the StudentResultEventHandler class.



**Or else you can put this Code for Completed Method**

A\_StudentDemoTable::UpdateCustWorkflowState(

\_workflowEventArgs.parmWorkflowContext().parmRecId(),

StudentReqWorkflowState::Completed);

And you can set WorkflowState in all Event of this class like below.

### Cancelled Method:

public void canceled(WorkflowEventArgs \_workflowEventArgs)

{ A\_StudentDemoTable::UpdateCustWorkflowState(\_workflowEventArgs.parmWorkflowContext().parmRecId(), StudentReqWorkflowState::PendingCancellation);

}

### Completed Method:

public void completed(WorkflowEventArgs \_workflowEventArgs)

{ A\_StudentDemoTable::UpdateCustWorkflowState(\_workflowEventArgs.parmWorkflowContext().parmRecId(), StudentReqWorkflowState::Completed);

}

### Started Method:

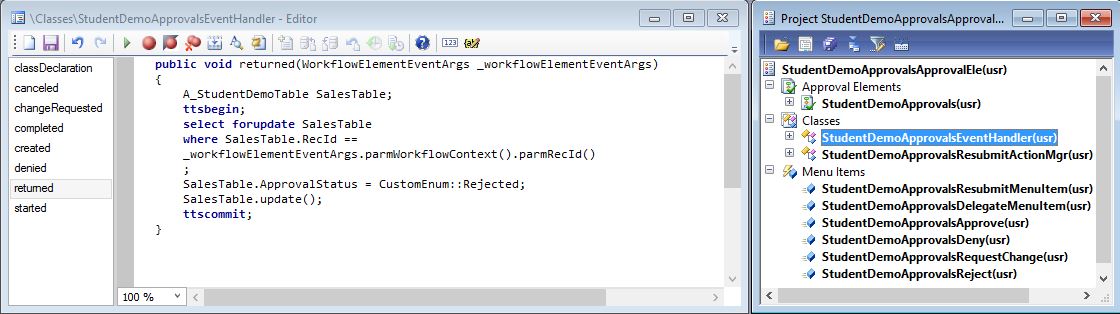
public void started(WorkflowEventArgs \_workflowEventArgs)

{ A\_StudentDemoTable::UpdateCustWorkflowState(\_workflowEventArgs.parmWorkflowContext().parmRecId(), StudentReqWorkflowState::Submitted);

}

## Add Element Level Event Handlers

* In the AOT locate the class StudentDemoApprovalsEventHandler. This class was created by the Approval element wizard.
* Add the following code in the returned method.
* Save your changes to the class.
* In the AOT, right-click on the AOT node and select Incremental CIL generation from X++.



And you can set WorkflowState in all Event of this class like below

### Canceled Method:

public void canceled(WorkflowElementEventArgs \_workflowElementEventArgs)

{ A\_StudentDemoTable::UpdateCustWorkflowState(\_workflowElementEventArgs.parmWorkflowContext().parmRecId(),StudentReqWorkflowState::PendingCancellation);

}

### changeRequested Method:

public void changeRequested(WorkflowElementEventArgs \_workflowElementEventArgs)

{ A\_StudentDemoTable::UpdateCustWorkflowState(\_workflowElementEventArgs.parmWorkflowContext().parmRecId(),StudentReqWorkflowState::ChangeRequest);

}

### Completed Method:

public void completed(WorkflowElementEventArgs \_workflowElementEventArgs)

{ A\_StudentDemoTable::UpdateCustWorkflowState(\_workflowElementEventArgs.parmWorkflowContext().parmRecId(),StudentReqWorkflowState::Approved);

}

### Returned Method:

public void returned(WorkflowElementEventArgs \_workflowElementEventArgs)

{ A\_StudentDemoTable::UpdateCustWorkflowState(\_workflowElementEventArgs.parmWorkflowContext().parmRecId(),StudentReqWorkflowState::Returned);

}

### Started Method:

public void started(WorkflowElementEventArgs \_workflowElementEventArgs)

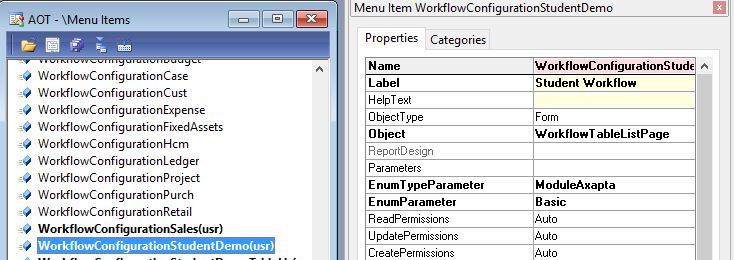
{ A\_StudentDemoTable::UpdateCustWorkflowState(\_workflowElementEventArgs.parmWorkflowContext().parmRecId(),StudentReqWorkflowState::Submitted);

}

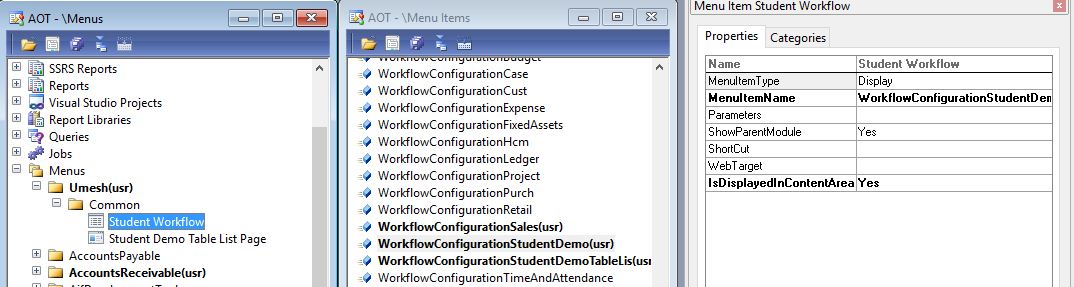
## Author a Workflow

The workflow category we created in the first procedure needs to be added to the main menu.

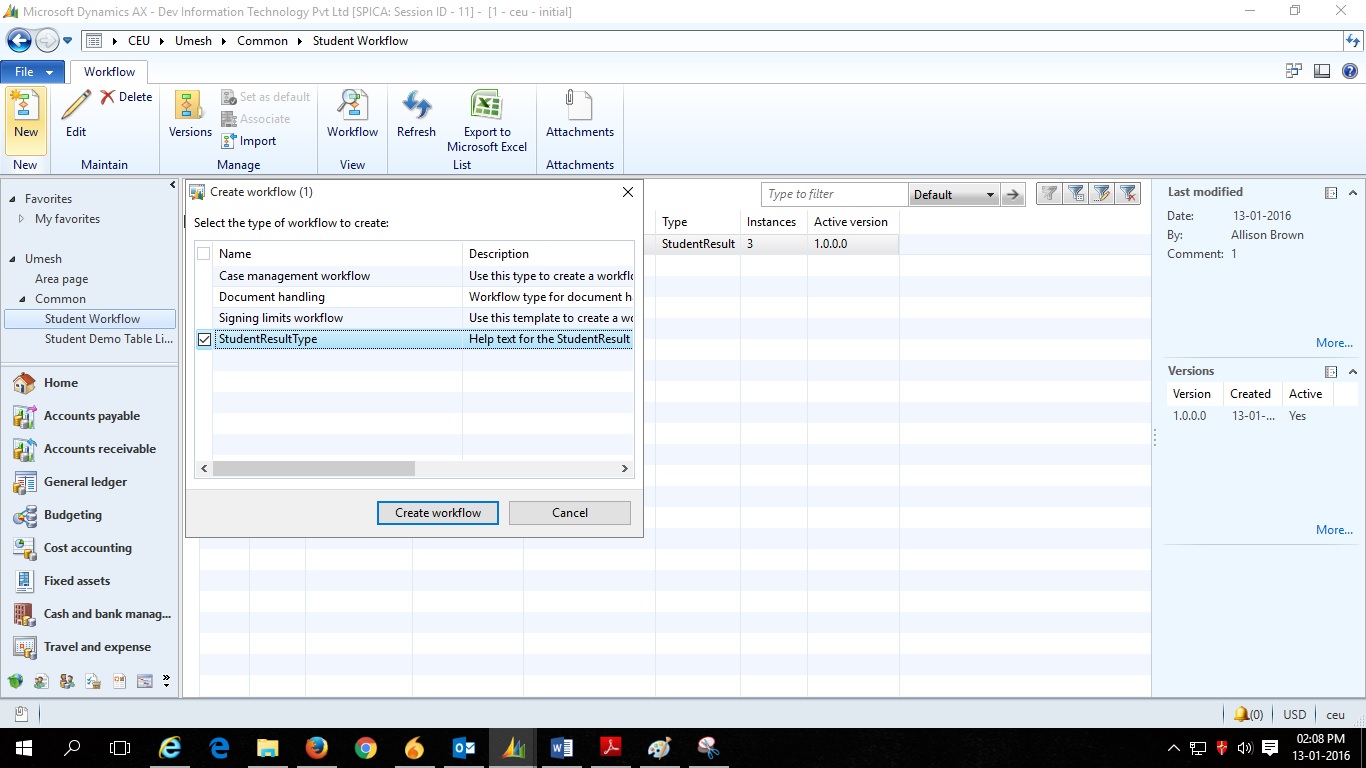
* Create a new display menu item called WorkflowConfigurationStudentDemo.



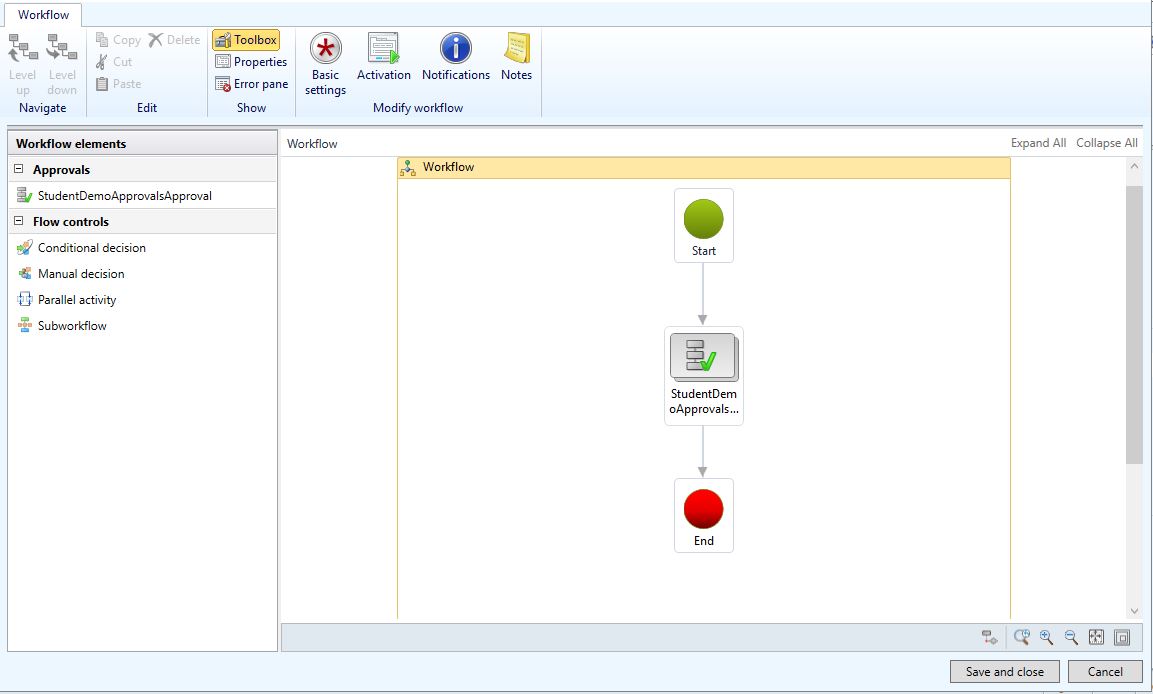
* Set the label to Student Workflow.
* Set the object to WorkflowTableListPage.
* Set the EnumTypeParameter to ModuleAxapta.
* Set the EnumParameter to Basic.
* Add the new menu item to Your Created Menu > menu name.
* In the property sheet for the new node in the menu, set the property IsDisplayedInContentArea to Yes.
* Save your changes to the menu.



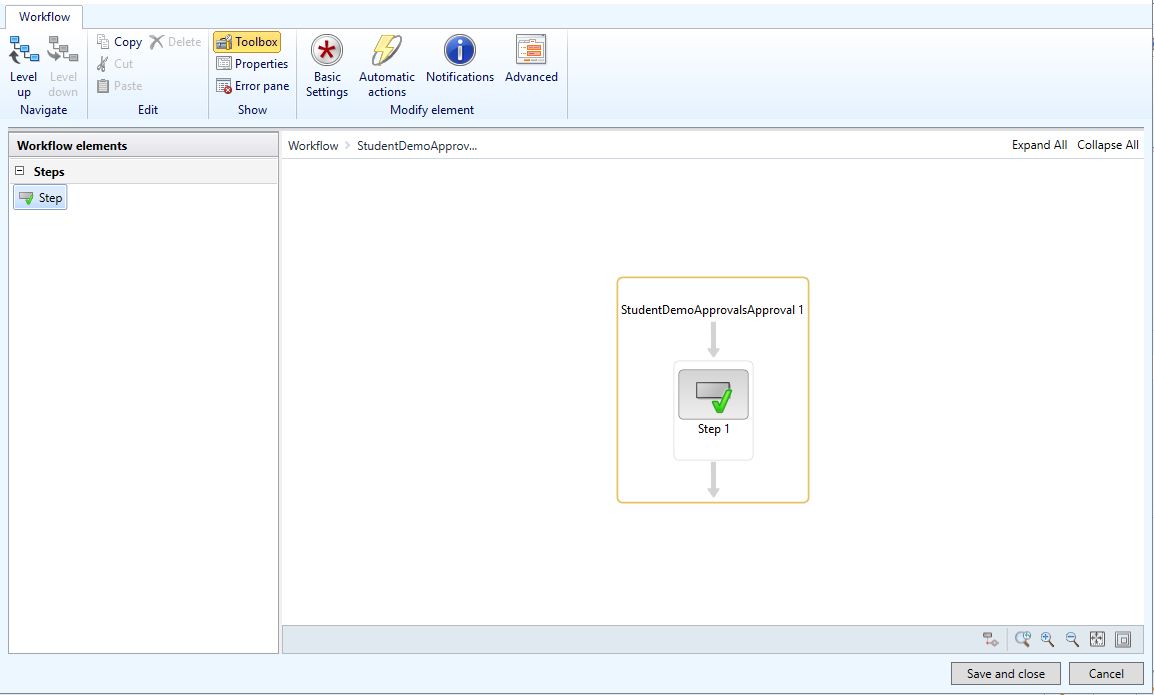
* Open the main menu and select Your Created Menu > Menu Name > WorkflowConfigurationStudentDemo.
* Click New.
* Select StudentResultType and click Create workflow. The workflow editor window opens.



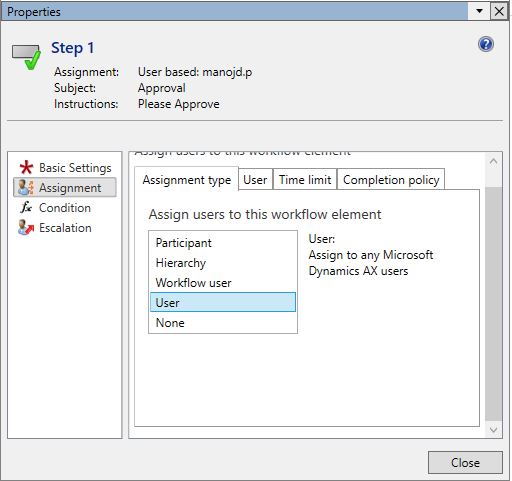
* Drag StudentDemoApprovalsApproval approval from the Workflow elements window to the Workflow window.



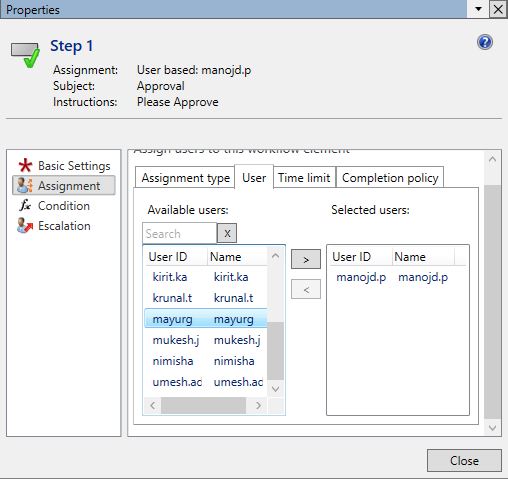
* Drag the bottom node of the Start box to the top node of the Approval box.
* Drag the bottom node of the Approval box to the top node of the End box.



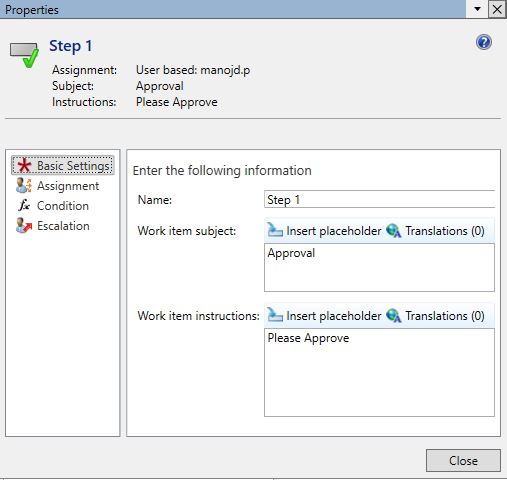
* Double click on the Approval box.
* Click Step 1
* Click Assignment.
* On the Assignment type tab, select User.



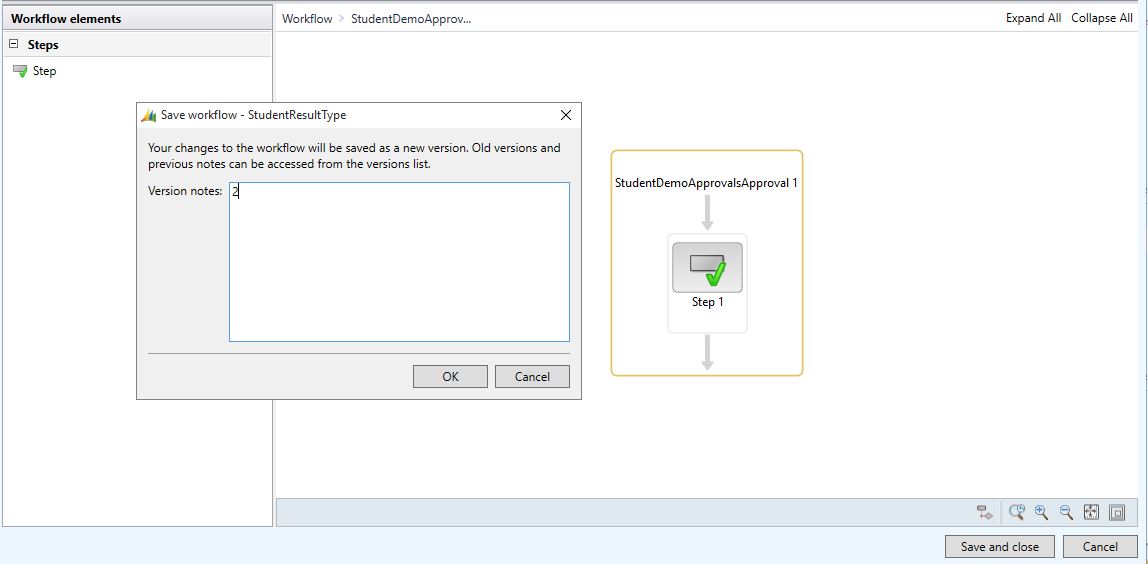
* On the User tab, double click on NewUser.



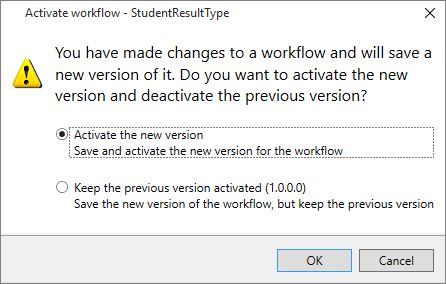
* Click on Basic Settings and then enter a subject and instructions for the approval.



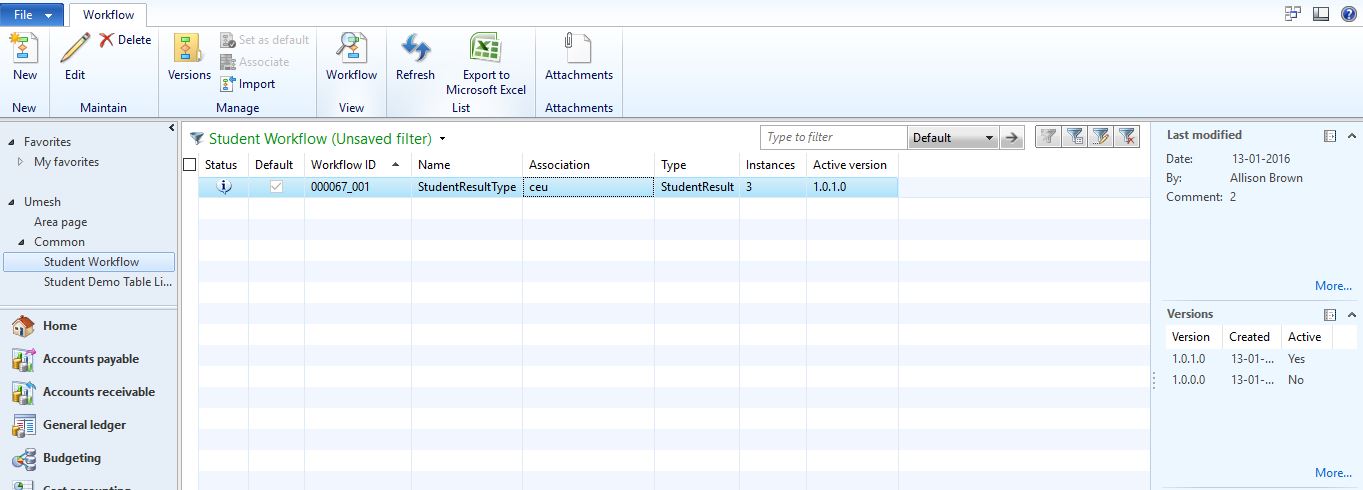
* Click on Close.
* Click on Save and Close.



* Enter some notes for the new workflow if you wish.

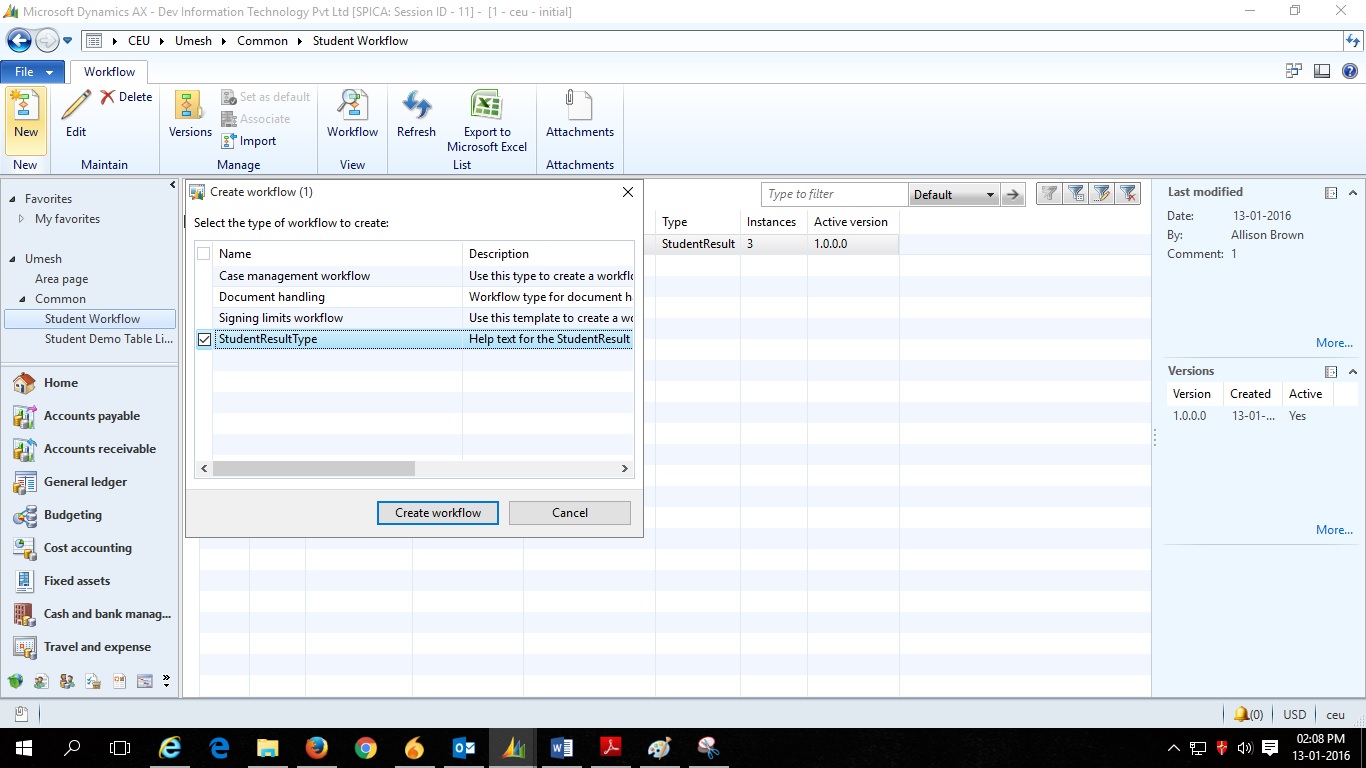


* Select to Activate the new version
* Click OK.
* Workflow Created look like in WorkflowTableListPage.

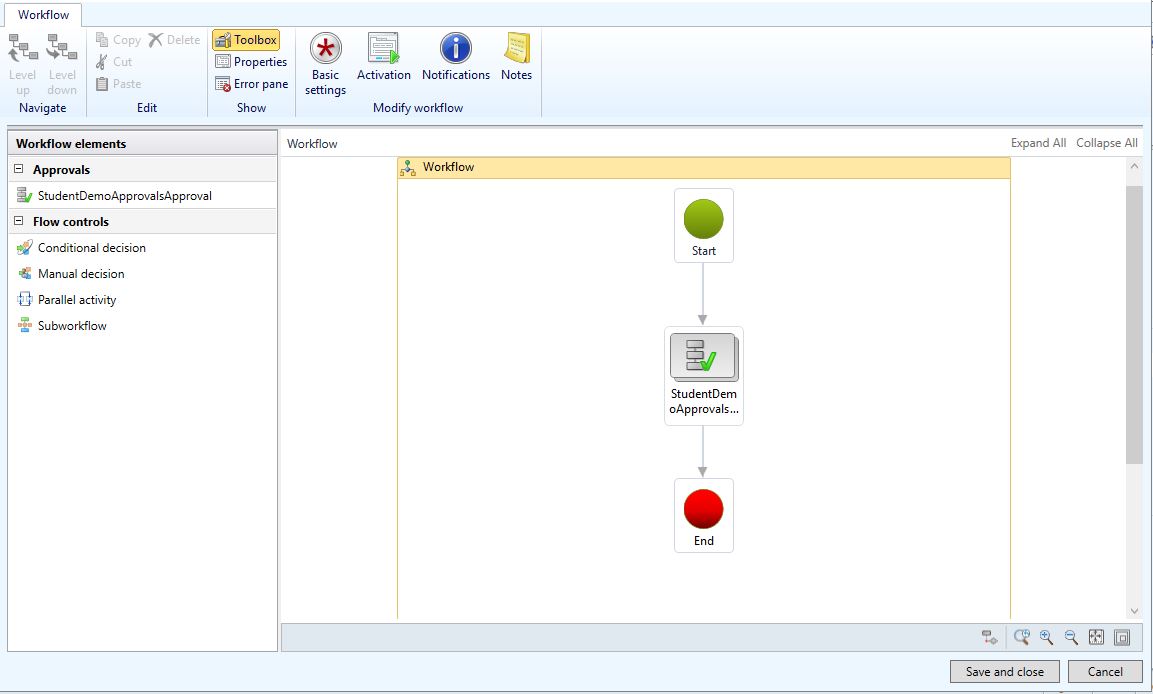


# Testing Custom Workflow

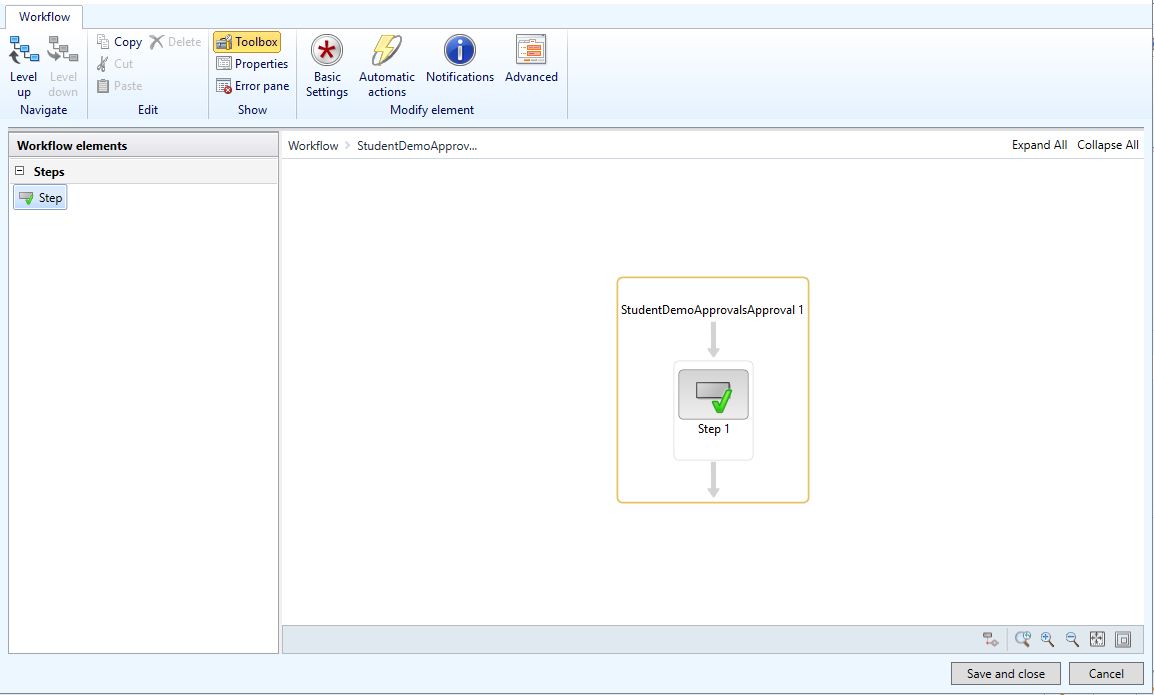
First Open Your Create Menuitem for Creating Workflow. Click New to Create Workflow.



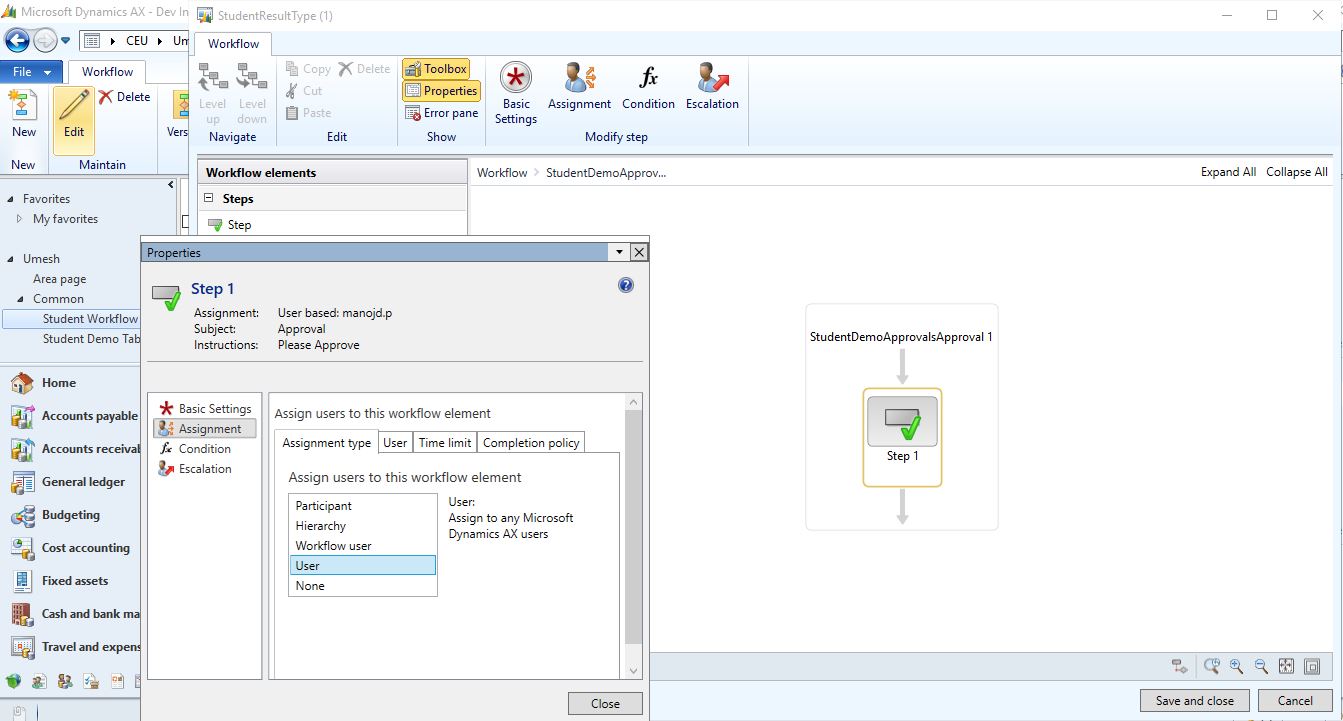
Now Drag and Drop StudentDemoApprovalsApproval into Workflow and Drop lines like below Image.



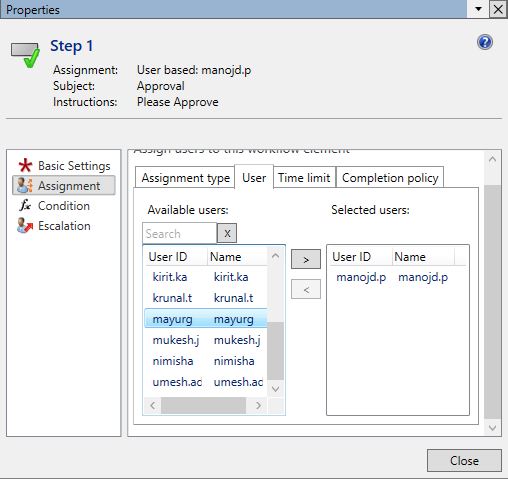
Double Click on StudentDemoApprovalsApproval to Open State and select AssignmentType.



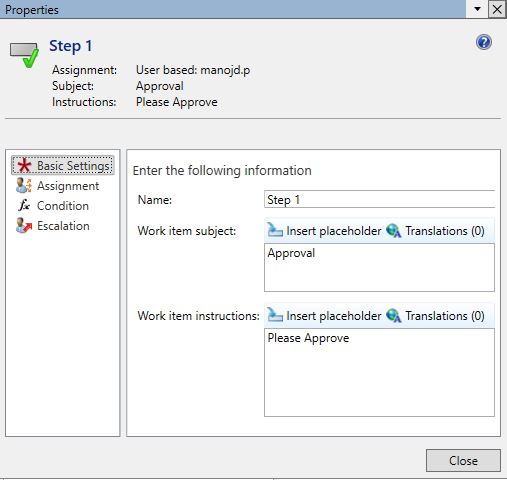
In Assignment select User



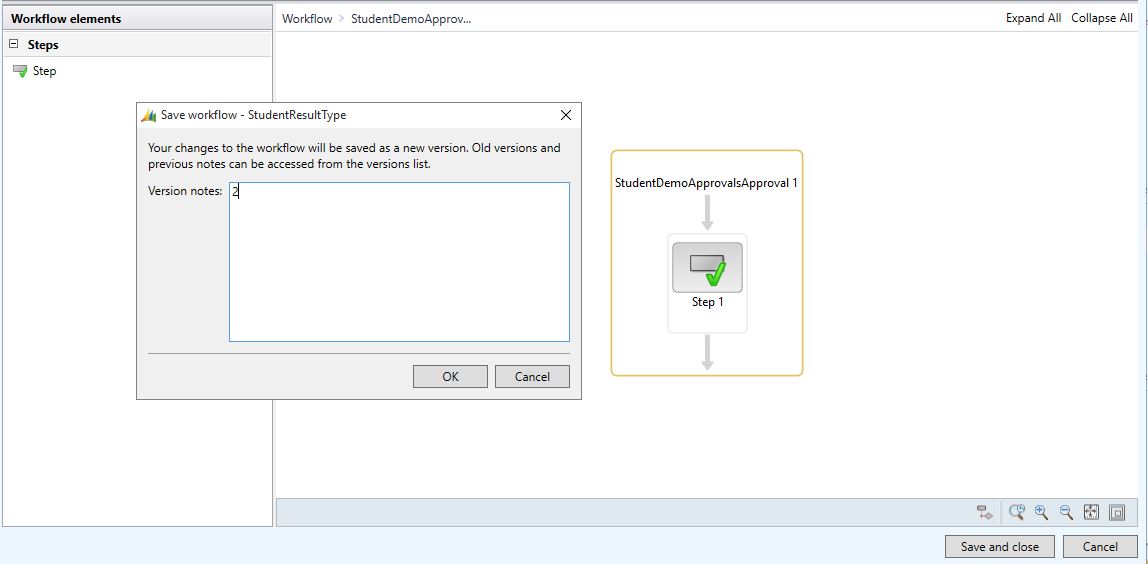
Now go to User Tab And Double Click on Particular User



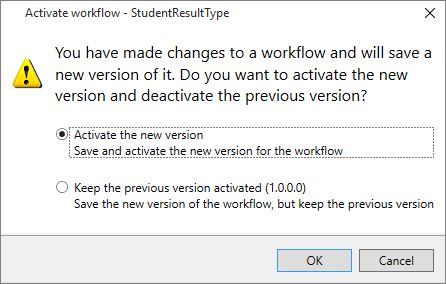
Go to Basic Settings Enter Subjects and Instructions like below image



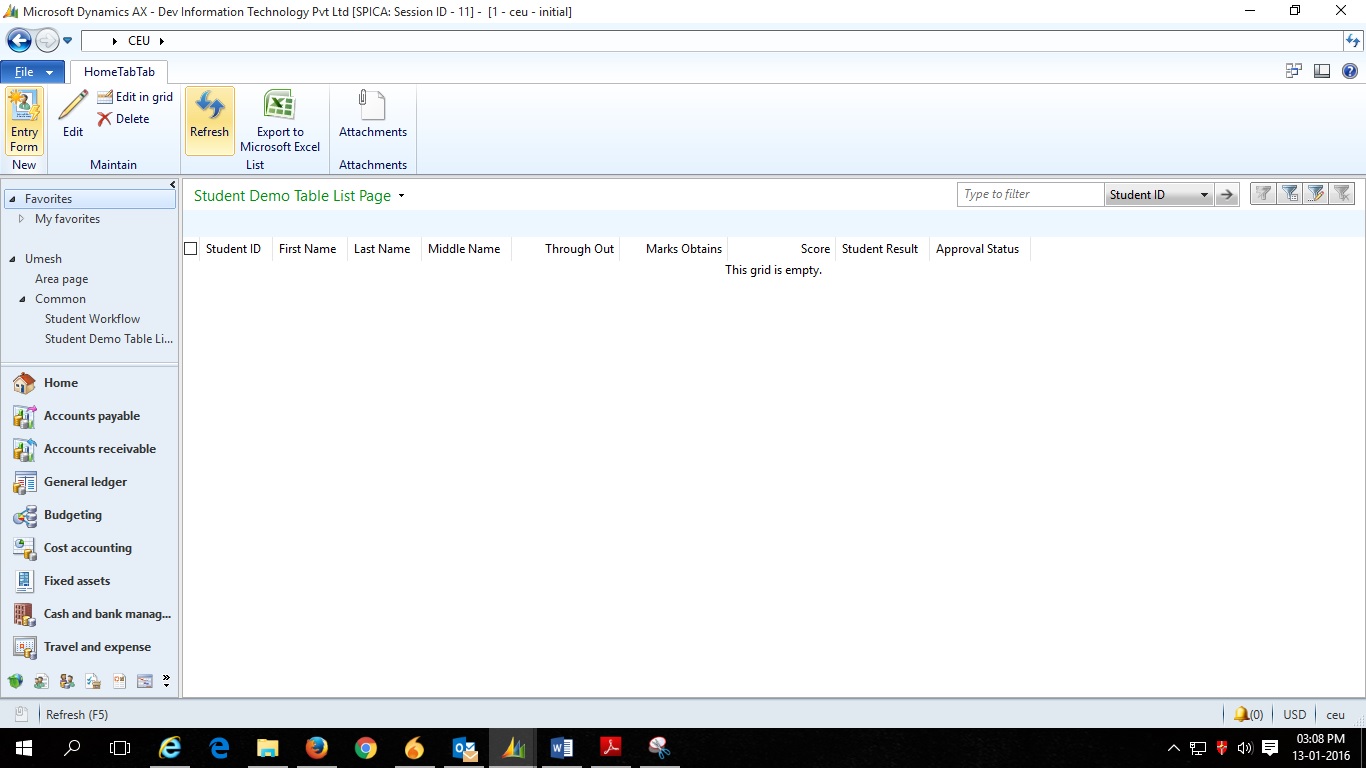
After Click close then (ctl+s) to save and click Save and Close



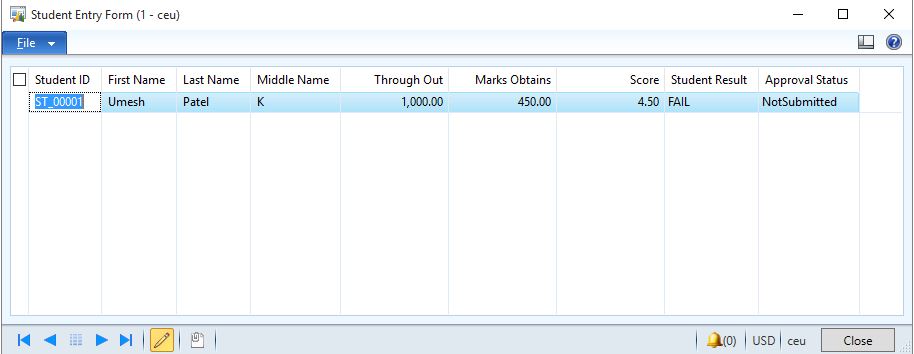
After Select Active the New Version.



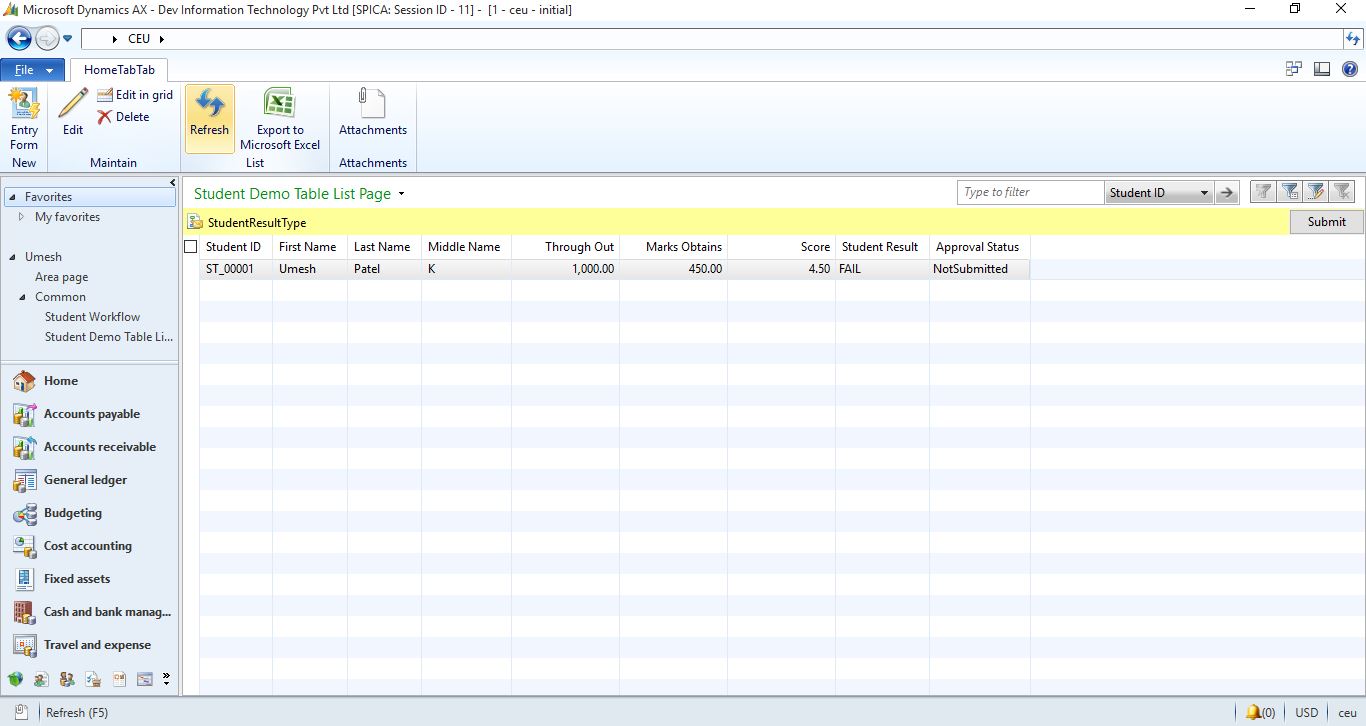
Go to StudentDemoListPage in Click EntryForm to enter Student Details



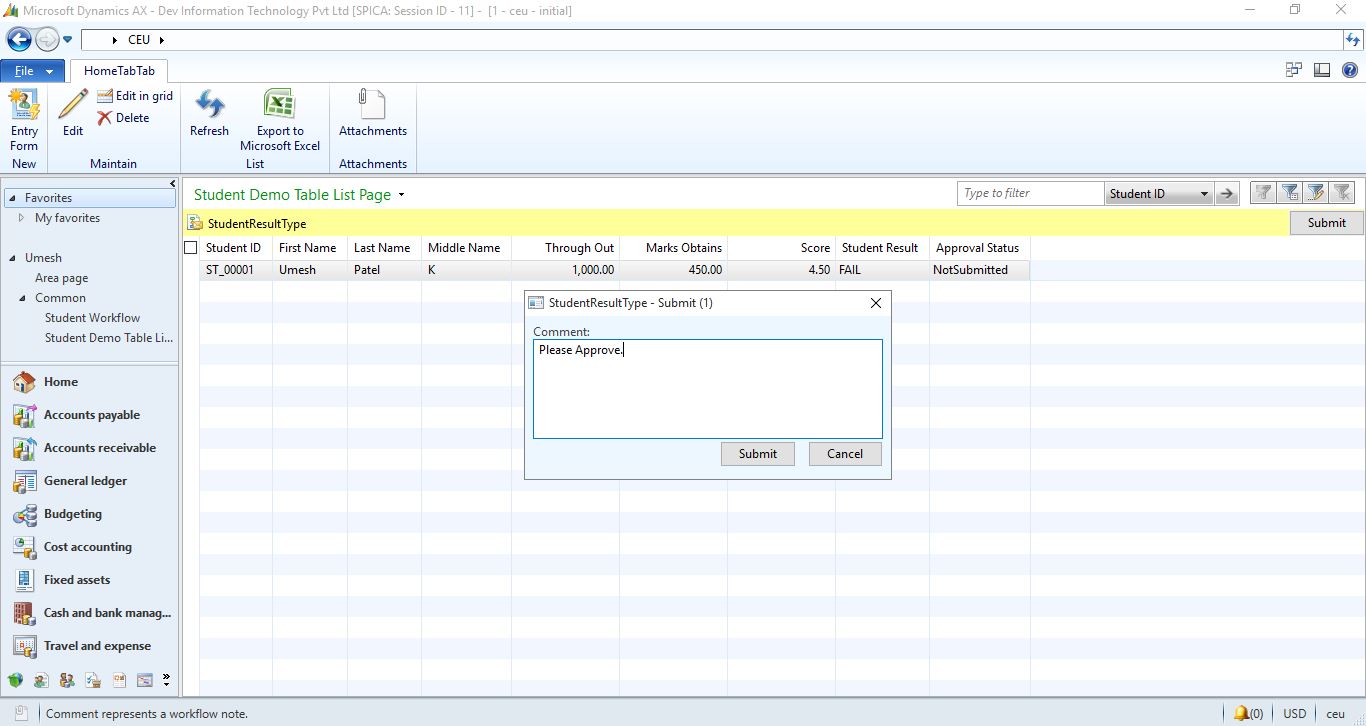
Fill Student Details and Close



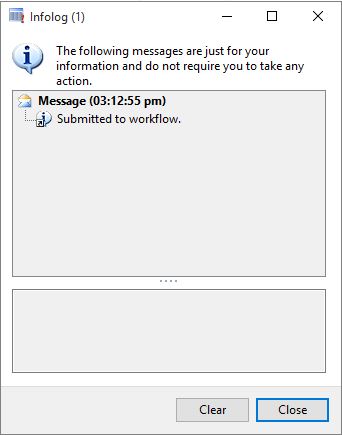
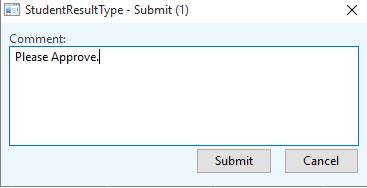
Then Refresh the StudentDemoListPage to Get Data.



Now Submitting the Data Click Submit to Open Comment Window

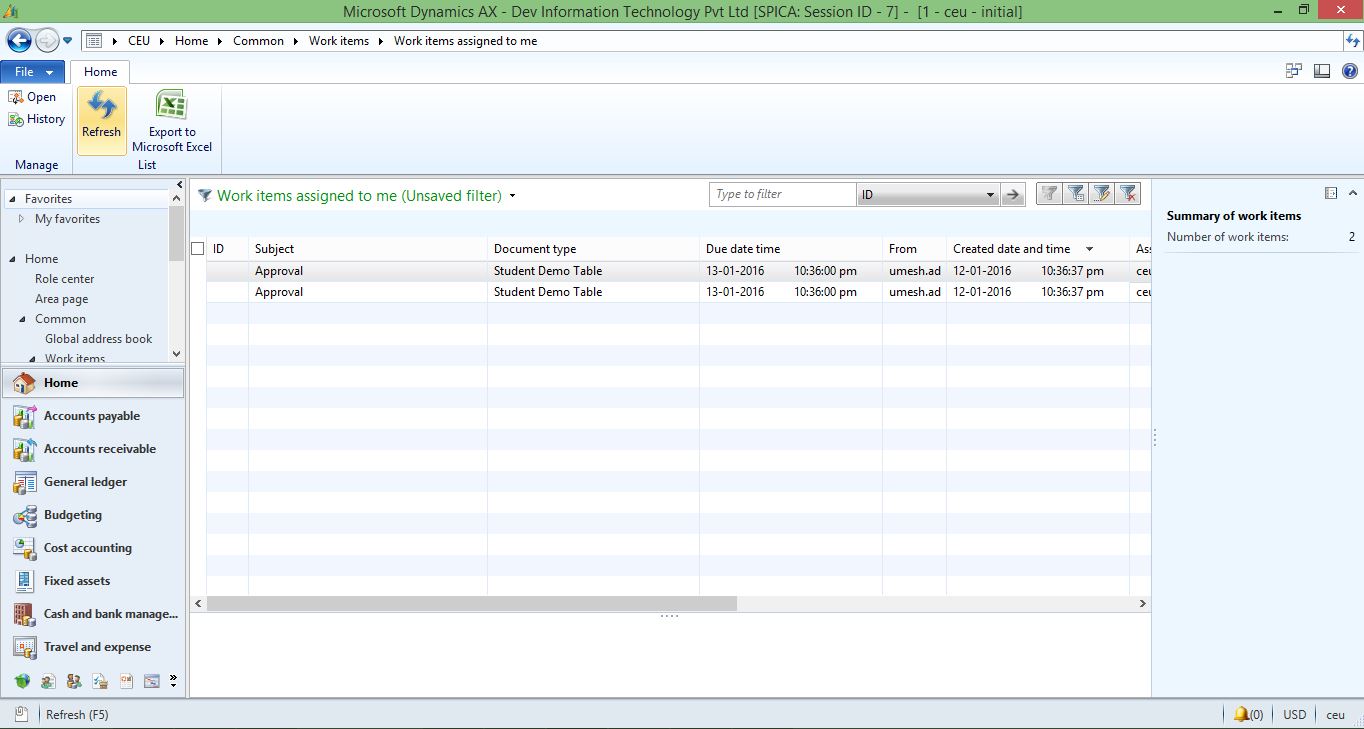


Fill Comment and Press Submit Button. Infolog will open for Display Your Workflow Submitted to Selected User.

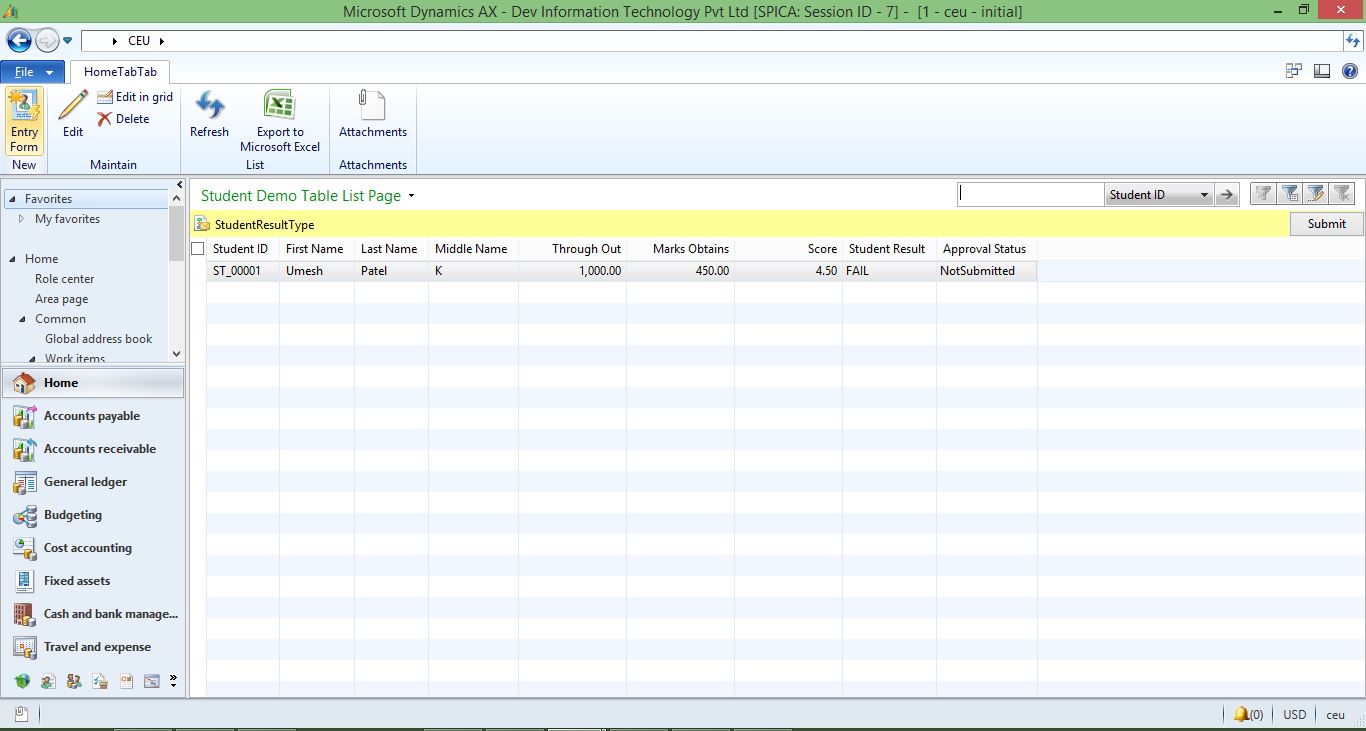


Now go to That User Login and Company. Notification will pop up for Workflow. Open Form and Select

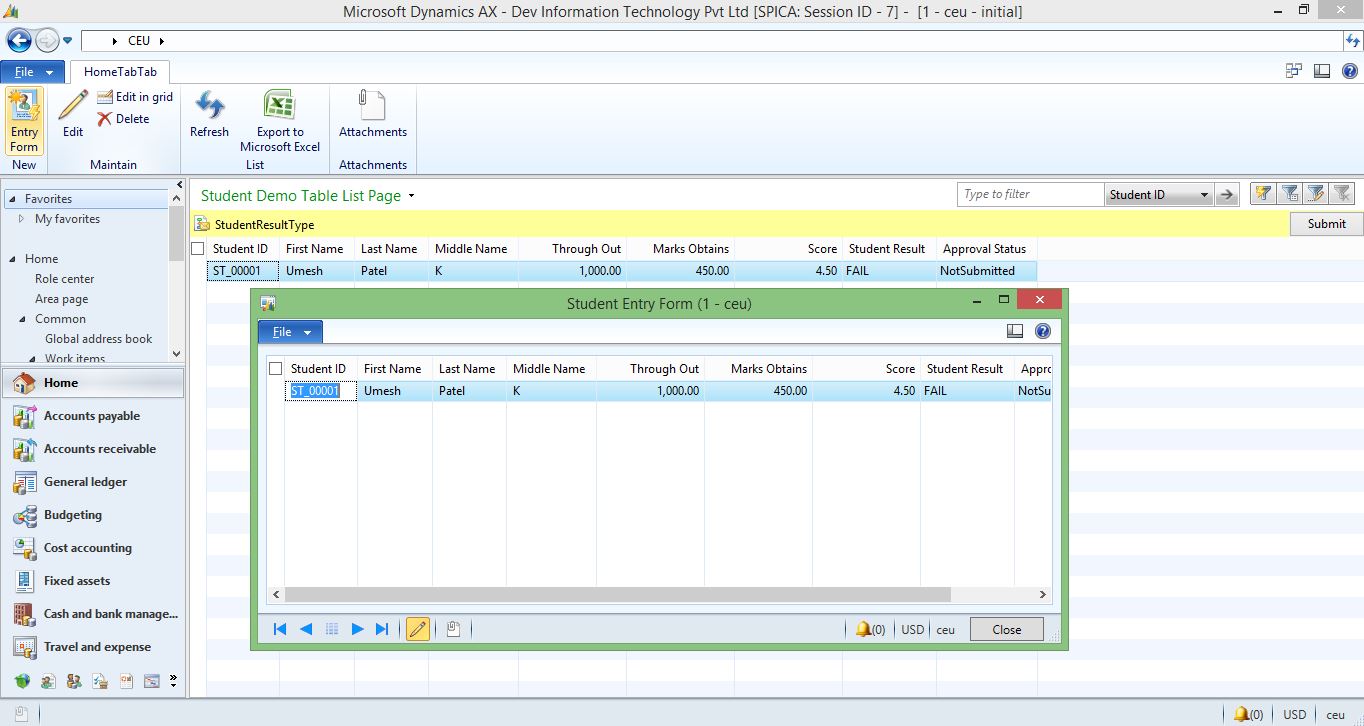
Data and Click Open to Open Form.



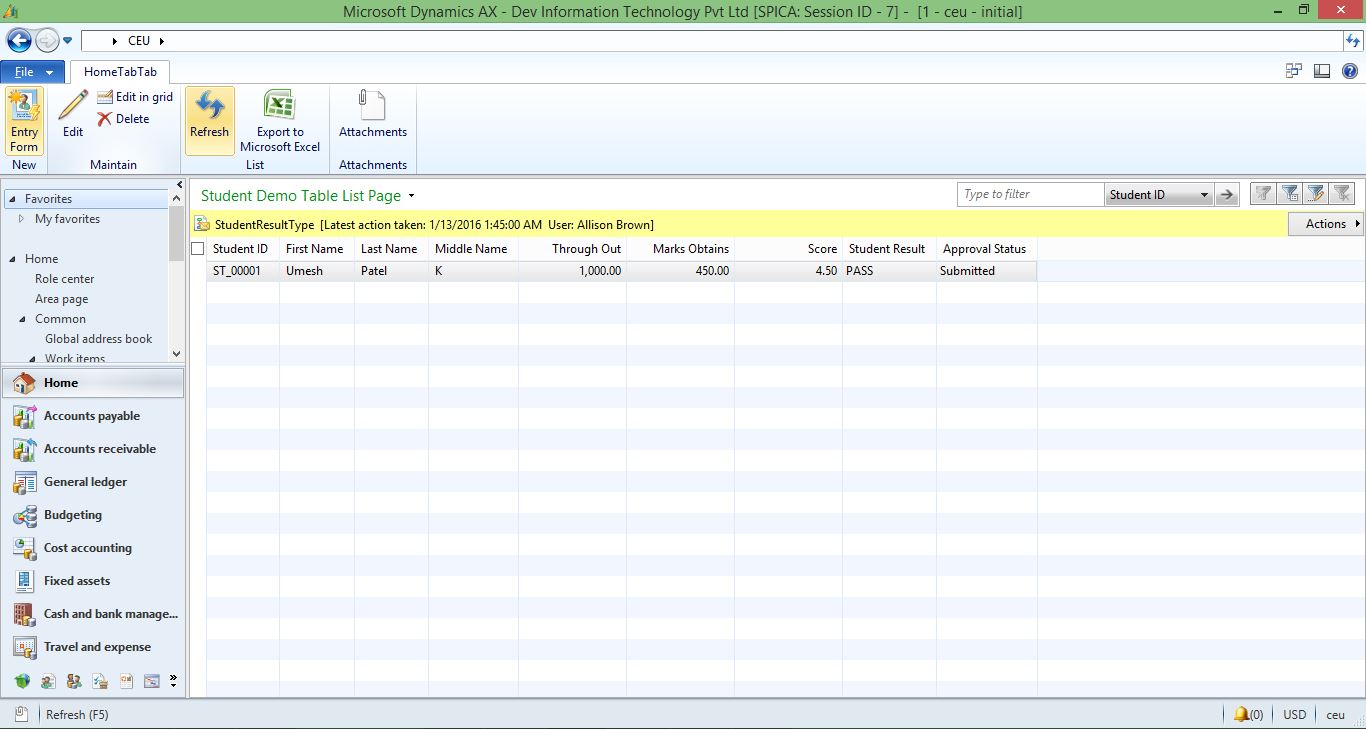
Now Select Record and Edit it to Open StudentEntryForm.



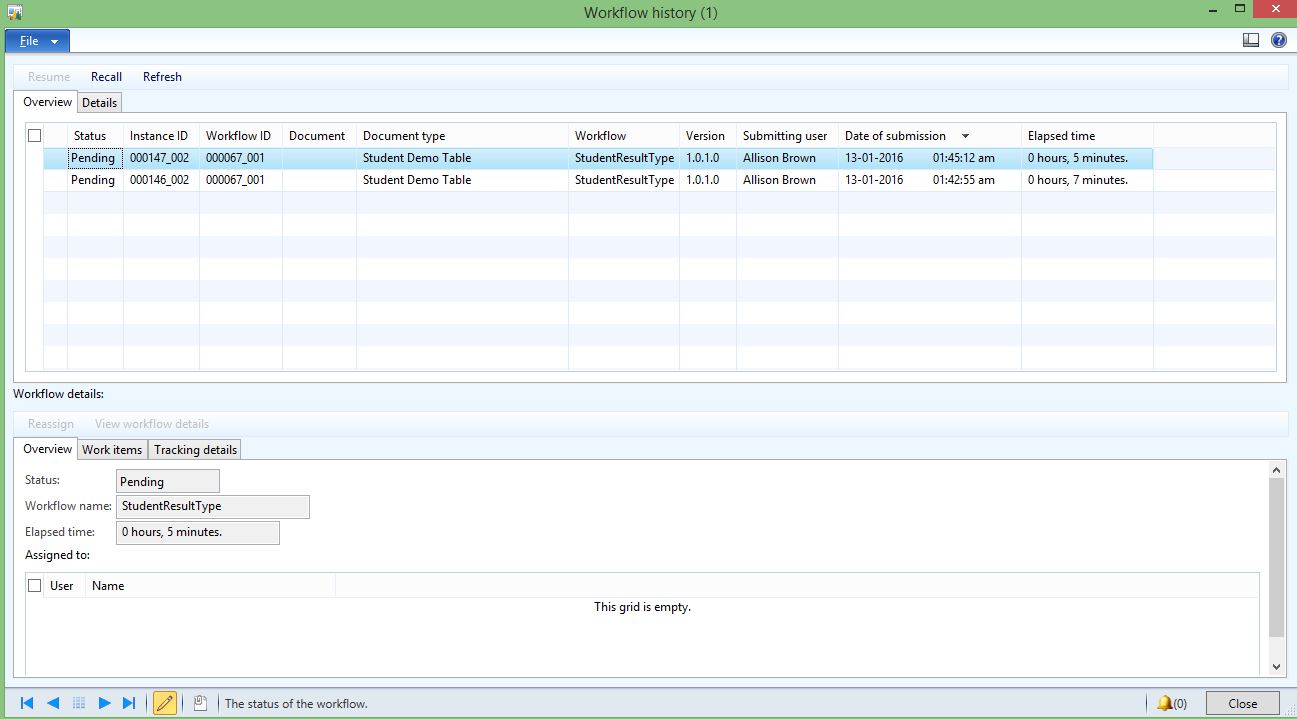
Change Result to PASS and Approval Status to Submitted.



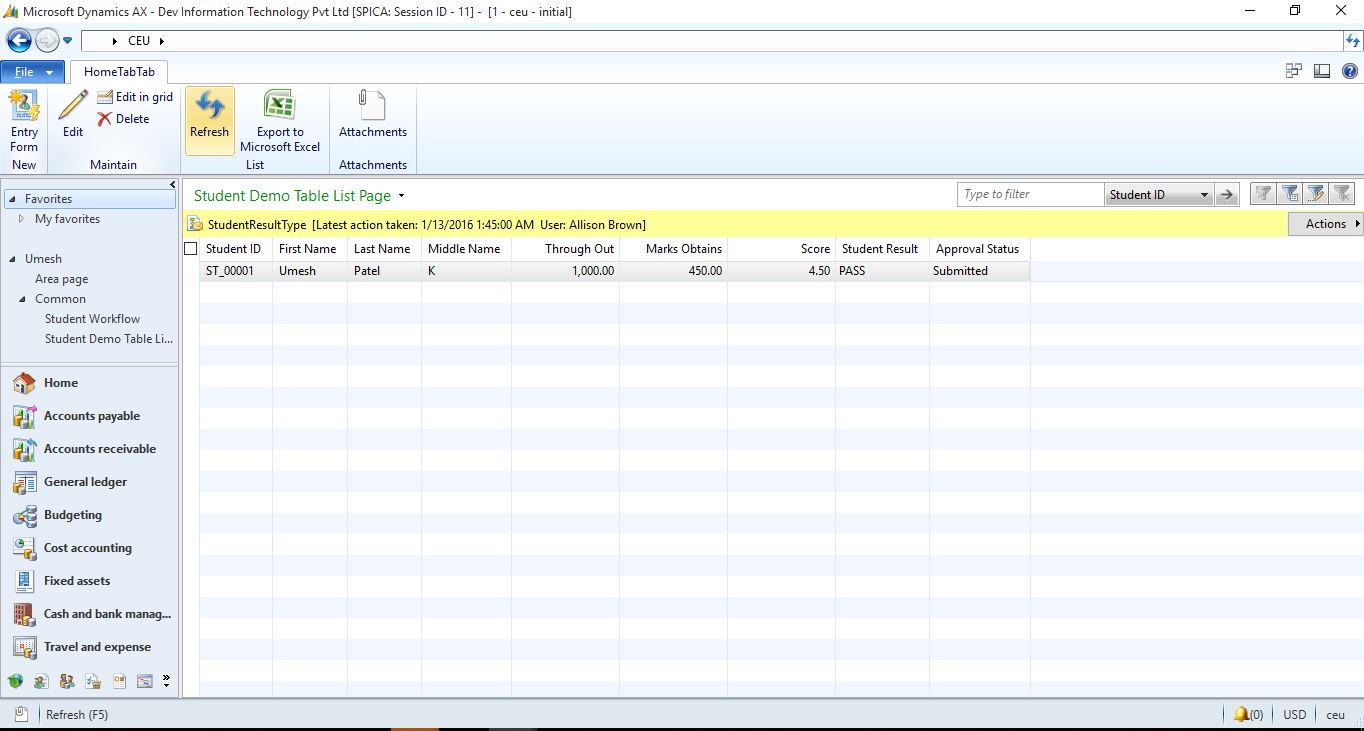
Come back to StudentDemoListPage and Refresh Now Record Changed.



For See History Click Action > View History then Form Will pen and There you can see Record or Workflow History.



Now Comeback to User Login and Refresh StudentDemoListPage You can see submitted Record and ApprovalStatus and Result Wil be changed.



# Complete Code

## A\_StudentDemoTable

#### Create CanSubmitToWorkFlow Method in Table Method

public boolean canSubmitToWorkflow(str \_workflowType = '')

{

boolean ret;

ret = this.RecId != 0 && this.StudentReqWorkflowState ==

StudentReqWorkflowState::NotSubmitted;

return ret;

}

## StudentEntryForm

#### DataSource Create Method

public void create(boolean \_append = false)

{

int max1;

A\_StudentDemoTable \_Student;

str 60 Value;

str 60 \_newCode;

;

//breakpoint;

super(\_append);

select firstOnly Student\_ID from \_Student order by RecId desc;// query to find last number

if(!\_Student.Student\_ID)// When table is blank

{

\_newCode='ST\_'+ strRFix(int2str(1),5, "0");//Assigne value to table field

A\_StudentDemoTable.Student\_ID =\_newCode;

// info(U\_City\_Master.City\_Code);

//StateMaster.StateCode='ST\_'+strRFix(int2str(1), 3, "0");//Assigne value to table field

}

else

{

max1=str2int(subStr(\_Student.Student\_ID,4,5))+1;// When table is not blank, add 1

A\_StudentDemoTable.Student\_ID ='ST\_'+ strRFix(int2str(max1),5, "0");//Assigne value to table field

// info(U\_City\_Master.City\_Code);

}// if Close

}

StudentResultWFType(Project)

## StudentResultSubmitManager(Class) – Main Method

public static void main(Args args)

{

// Variable declaration.

recId recId = args.record().RecId;

WorkflowCorrelationId workflowCorrelationId;

// Hardcoded type name

WorkflowTypeName workflowTypeName =workflowtypestr(StudentResult);

// Initial note is the information that users enter when they

// submit the document for workflow.

WorkflowComment note ="";

WorkflowSubmitDialog workflowSubmitDialog;

A\_StudentDemoTable SalesTable;

// Opens the submit to workflow dialog.

workflowSubmitDialog = WorkflowSubmitDialog::construct(args.caller().getActiveWorkflowConfiguration());

workflowSubmitDialog.run();

if (workflowSubmitDialog.parmIsClosedOK())

{

recId = args.record().RecId;

SalesTable = args.record();

// Get comments from the submit to workflow dialog.

note = workflowSubmitDialog.parmWorkflowComment();

try

{

ttsbegin;

workflowCorrelationId = Workflow::activateFromWorkflowType(workflowTypeName,recId,note,NoYes::No);

SalesTable.ApprovalStatus = CustomEnum::NotSubmitted;

SalesTable.Result = StudentResult::FAIL;

// Send an Infolog message.

info("Submitted to workflow.");

ttscommit;

}

catch(exception::Error)

{

info("Error on workflow activation.");

}

}

args.caller().updateWorkFlowControls();

}

StudentResultWFType(Project)

## StudentResultEventHandler (Class) – Completed Method

public void completed(WorkflowEventArgs \_workflowEventArgs)

{

A\_StudentDemoTable SalesTable;

select forupdate SalesTable where SalesTable.RecId ==

\_workflowEventArgs.parmWorkflowContext().parmRecId();

if(SalesTable.RecId)

{

SalesTable.ApprovalStatus = CustomEnum::Approved;

SalesTable.write();

}

}

Or else you Can Use below Code for Completed Method

A\_StudentDemoTable::UpdateCustWorkflowState(\_workflowEventArgs.parmWorkflowContext().parmRecId(), StudentReqWorkflowState::Completed);

## StudentResultEventHandler (Class) – Canceled Method

public void canceled(WorkflowEventArgs \_workflowEventArgs)

{ A\_StudentDemoTable::UpdateCustWorkflowState(\_workflowEventArgs.parmWorkflowContext().parmRecId(), StudentReqWorkflowState::PendingCancellation);

}

## StudentResultEventHandler (Class) – Started Method

public void started(WorkflowEventArgs \_workflowEventArgs)

{

A\_StudentDemoTable::UpdateCustWorkflowState(\_workflowEventArgs.parmWorkflowContext().parmRecId(), StudentReqWorkflowState::Submitted);

}

StudentDemoApprovalsApprovalEle (Project)

## StudentDemoApprovalsEventHandler (Class) – returned Method

public void returned(WorkflowElementEventArgs \_workflowElementEventArgs)

{

A\_StudentDemoTable SalesTable;

ttsbegin;

select forupdate SalesTable

where SalesTable.RecId ==

\_workflowElementEventArgs.parmWorkflowContext().parmRecId()

;

SalesTable.ApprovalStatus = CustomEnum::Rejected;

SalesTable.update();

ttscommit;

}

Or else you Can Use below Code for Returned Method

A\_StudentDemoTable::UpdateCustWorkflowState( \_workflowElementEventArgs.parmWorkflowContext().parmRecId(),

StudentReqWorkflowState::Returned);

## StudentDemoApprovalsEventHandler (Class) – Canceled Method

public void canceled(WorkflowElementEventArgs \_workflowElementEventArgs)

{

A\_StudentDemoTable::UpdateCustWorkflowState(

\_workflowElementEventArgs.parmWorkflowContext().parmRecId(),

StudentReqWorkflowState::PendingCancellation);

}

## StudentDemoApprovalsEventHandler (Class) – ChangedRequested Method

public void changeRequested(WorkflowElementEventArgs \_workflowElementEventArgs)

{

A\_StudentDemoTable::UpdateCustWorkflowState(

\_workflowElementEventArgs.parmWorkflowContext().parmRecId(),

StudentReqWorkflowState::ChangeRequest);

}

## StudentDemoApprovalsEventHandler (Class) – Completed Method

public void completed(WorkflowElementEventArgs \_workflowElementEventArgs)

{

A\_StudentDemoTable::UpdateCustWorkflowState(

\_workflowElementEventArgs.parmWorkflowContext().parmRecId(),

StudentReqWorkflowState::Approved);

}

## StudentDemoApprovalsEventHandler (Class) – Started Method

public void started(WorkflowElementEventArgs \_workflowElementEventArgs)

{

A\_StudentDemoTable::UpdateCustWorkflowState(

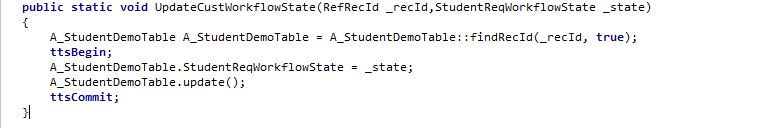
\_workflowElementEventArgs.parmWorkflowContext().parmRecId(),

StudentReqWorkflowState::Submitted);

}

## CODE FOR RESUBMIT

Create New Method in Table and Name it “UpdateCustWorkflowState”.



public static void UpdateCustWorkflowState(RefRecId \_recId,StudentReqWorkflowState \_state)

{

A\_StudentDemoTable A\_StudentDemoTable = A\_StudentDemoTable::findRecId(\_recId, true);

ttsBegin;

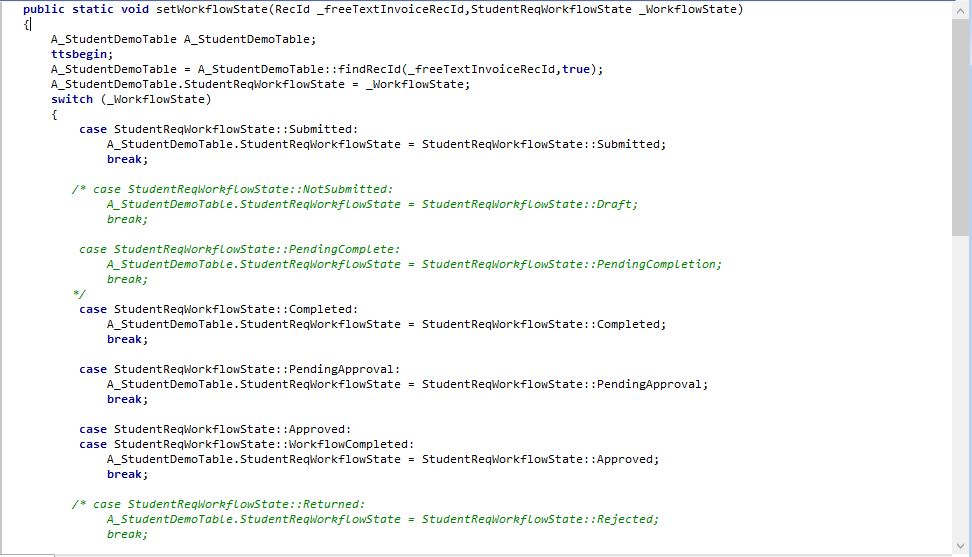
A\_StudentDemoTable.StudentReqWorkflowState = \_state;

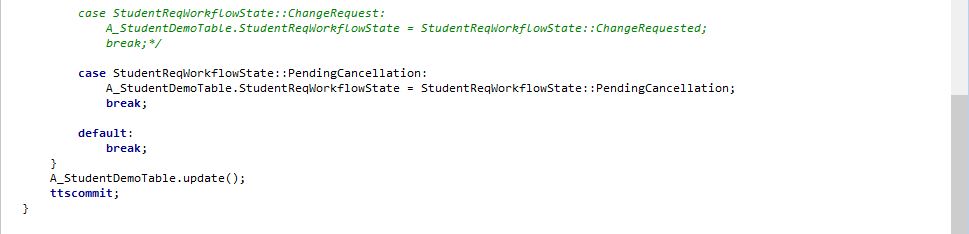
A\_StudentDemoTable.update();

ttsCommit;

}

Create New Method in Table and Name it “setWorkflowState”.





public static void setWorkflowState(RecId \_freeTextInvoiceRecId,StudentReqWorkflowState \_WorkflowState)

{

A\_StudentDemoTable A\_StudentDemoTable;

ttsbegin;

A\_StudentDemoTable = A\_StudentDemoTable::findRecId(\_freeTextInvoiceRecId,true);

A\_StudentDemoTable.StudentReqWorkflowState = \_WorkflowState;

switch (\_WorkflowState)

{

case StudentReqWorkflowState::Submitted:

A\_StudentDemoTable.StudentReqWorkflowState = StudentReqWorkflowState::Submitted;

break;

*/\* case StudentReqWorkflowState::NotSubmitted:*

*A\_StudentDemoTable.StudentReqWorkflowState = StudentReqWorkflowState::Draft;*

*break;*

*case StudentReqWorkflowState::PendingComplete:*

*A\_StudentDemoTable.StudentReqWorkflowState = StudentReqWorkflowState::PendingCompletion;*

*break;*

*\*/*

case StudentReqWorkflowState::Completed:

A\_StudentDemoTable.StudentReqWorkflowState = StudentReqWorkflowState::Completed;

break;

case StudentReqWorkflowState::PendingApproval:

A\_StudentDemoTable.StudentReqWorkflowState = StudentReqWorkflowState::PendingApproval;

break;

case StudentReqWorkflowState::Approved:

case StudentReqWorkflowState::WorkflowCompleted:

A\_StudentDemoTable.StudentReqWorkflowState = StudentReqWorkflowState::Approved;

break;

*/\* case StudentReqWorkflowState::Returned:*

*A\_StudentDemoTable.StudentReqWorkflowState = StudentReqWorkflowState::Rejected;*

*break;*

*case StudentReqWorkflowState::ChangeRequest:*

*A\_StudentDemoTable.StudentReqWorkflowState = StudentReqWorkflowState::ChangeRequested;*

*break;\*/*

case StudentReqWorkflowState::PendingCancellation:

A\_StudentDemoTable.StudentReqWorkflowState = StudentReqWorkflowState::PendingCancellation;

break;

default:

break;

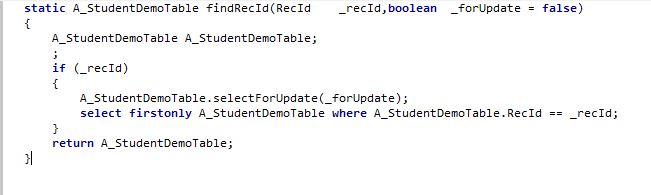
}

A\_StudentDemoTable.update();

ttscommit;

}

Create New Method in Table and Name it “findRecId”.



static A\_StudentDemoTable findRecId(RecId \_recId,boolean \_forUpdate = false)

{

A\_StudentDemoTable A\_StudentDemoTable;

;

if (\_recId)

{

A\_StudentDemoTable.selectForUpdate(\_forUpdate);

select firstonly A\_StudentDemoTable where A\_StudentDemoTable.RecId == \_recId;

}

return A\_StudentDemoTable;

}

## StudentDemoApprovalsApprovalEle (Project)

StudnetApprovalResubmitActionMgr (Class)

Create below method in StudnetApprovalResubmitActionMgr class for reSubmition workflow

### Declaration Method

public class StudnetApprovalResubmitActionMgr

{

RecId StudentTableId;

WorkflowVersionTable workflowVersionTable;

WorkflowComment workflowComment;

boolean submit;

*//WorkflowWorkItemTable workflowWorkItemTable;*

WorkflowWorkItemTable workflowWorkItemTable;

userId userId;

menuItemName menuItemName;

EPWorkflowControlContext workflowControlContext;

workflowTypeName workflowTemplateName;

}

### DialogOK

public boolean dialogOk()

{

WorkflowSubmitDialog workflowSubmitDialog;

WorkflowWorkItemActionDialog workflowWorkItemActionDialog;

boolean ok;

if (menuItemName == menuitemActionStr(StudentWorkflowTypeSubmitMenuItem))

{

workflowSubmitDialog = WorkflowSubmitDialog::construct(this.parmWorkflowVersionTable());

workflowSubmitDialog.run();

this.parmWorkflowComment(workflowSubmitDialog.parmWorkflowComment());

ok = workflowSubmitDialog.parmIsClosedOK();

}

else if (menuItemName == menuitemActionStr(StudnetApprovalResubmitMenuItem))

{

workflowWorkItemActionDialog = WorkflowWorkItemActionDialog::construct(

workflowWorkItemTable,

WorkflowWorkItemActionType::Resubmit,

new MenuFunction(menuitemActionStr(StudnetApprovalResubmitMenuItem), MenuItemType::Action));

workflowWorkItemActionDialog.run();

this.parmWorkflowComment(workflowWorkItemActionDialog.parmWorkflowComment());

ok = workflowWorkItemActionDialog.parmIsClosedOK();

userId = workflowWorkItemActionDialog.parmTargetUser();

}

ok = true;

return ok;

}

### Init

public void init(

RecId \_documentRecordId,

menuItemName \_menuItemName,

WorkflowVersionTable \_workflowVersionTable,

WorkflowWorkItemTable \_workflowWorkItemTable,

EPWorkflowControlContext \_workflowControlContext)

{

if (\_documentRecordId)

{

this.parmCustInvoiceTableId(\_documentRecordId);

this.parmSubmit(\_menuItemName == menuitemActionStr(StudentWorkflowTypeSubmitMenuItem)

|| \_menuItemName == webActionItemStr(EPCustFreeInvoiceSubmitToWorkflow));

this.parmMenuItemName(\_menuItemName);

if (\_workflowControlContext)

{

this.parmWorkflowControlContext(\_workflowControlContext);

this.parmWorkflowWorkItemtable(\_workflowControlContext.getActiveWorkflowWorkItem());

this.parmWorkflowComment(\_workflowControlContext.getWorkflowComment());

this.parmWorkflowTemplateName(\_workflowControlContext.getActiveWorkflowConfiguration().workflowTable().TemplateName);

}

else

{

this.parmWorkflowVersionTable(\_workflowVersionTable);

this.parmWorkflowWorkItemtable(\_workflowWorkItemTable);

this.parmWorkflowTemplateName(this.parmWorkflowVersionTable().workflowTable().TemplateName);

}

}

}

### ParmCustInvoicetableId

public RecId parmCustInvoiceTableId(RecId \_StudentTableId = StudentTableId)

{

StudentTableId = \_StudentTableId;

return StudentTableId;

}

### ParmMenuItemName

public menuItemName parmMenuItemName(menuItemName \_menuItemName = menuItemName)

{

menuItemName = \_menuItemName;

return menuItemName;

}

### ParmSubmit

public boolean parmSubmit(boolean \_submit = submit)

{

submit = \_submit;

return submit;

}

### ParmWorkflowComment

public WorkflowComment parmWorkflowComment(WorkflowComment \_workflowComment = workflowComment)

{

workflowComment = \_workflowComment;

return workflowComment;

}

### ParmWorkflowControlContext

public EPWorkflowControlContext parmWorkflowControlContext(EPWorkflowControlContext \_workflowControlContext = workflowControlContext)

{

workflowControlContext = \_workflowControlContext;

return workflowControlContext;

}

### ParmWorkflowTemplateName

public workflowTypeName parmWorkflowTemplateName(workflowTypeName \_workflowTemplateName = workflowTemplateName)

{

workflowTemplateName = \_workflowTemplateName;

return workflowTemplateName;

}

### ParmWorkflowTableVersion

public WorkflowVersionTable parmWorkflowVersionTable(WorkflowVersionTable \_workflowVersionTable = workflowVersionTable)

{

workflowVersionTable = \_workflowVersionTable;

return workflowVersionTable;

}

### ParmWorkflowWorkItemtable

public WorkflowWorkItemTable parmWorkflowWorkItemtable(WorkflowWorkItemTable \_workflowWorkItemTable = workflowWorkItemTable)

{

workflowWorkItemTable = \_workflowWorkItemTable;

return workflowWorkItemTable;

}

### Resubmit

public void reSubmit()

{

WorkflowWorkItemActionManager::dispatchWorkItemAction(

workflowWorkItemTable,

workflowComment,

userId,

WorkflowWorkItemActionType::Resubmit,

menuItemName,

menuItemName == webActionItemStr(EPCustFreeInvoiceReSubmit));

A\_StudentDemoTable::setWorkflowState(StudentTableId, StudentReqWorkflowState::Submitted);

}

### Submit

public void submit()

{

NoYes activatingFromWeb;

activatingFromWeb = this.parmWorkflowControlContext() == null ? NoYes::No : NoYes::Yes;

Workflow::activateFromWorkflowType(

this.parmWorkflowTemplateName(),

StudentTableId,

this.parmWorkflowComment(),

activatingFromWeb);

A\_StudentDemoTable::setWorkflowState(StudentTableId, StudentReqWorkflowState::Submitted);

}

### Construct

public static StudnetApprovalResubmitActionMgr construct()

{

return new StudnetApprovalResubmitActionMgr();

}

### Main

public static void main(Args args)

{

StudnetApprovalResubmitActionMgr StudnetApprovalResubmitActionMgr;

A\_StudentDemoTable A\_StudentDemoTable;

FormDataSource A\_StudentDemoTableDataSource;

A\_StudentDemoTable = args.record();

A\_StudentDemoTableDataSource = A\_StudentDemoTable.dataSource();

StudnetApprovalResubmitActionMgr = StudnetApprovalResubmitActionMgr::construct();

if (args.menuItemName() == menuitemActionStr(StudentWorkflowTypeSubmitMenuItem) ||

args.menuItemName() == menuitemActionStr(StudnetApprovalResubmitMenuItem))

{

StudnetApprovalResubmitActionMgr.init(

A\_StudentDemoTable.RecId,

args.menuItemName(),

args.caller().getActiveWorkflowConfiguration(),

args.caller().getActiveWorkflowWorkItem(),

null);

}

else

{

StudnetApprovalResubmitActionMgr.init(

A\_StudentDemoTable.RecId,

args.menuItemName(),

null,

null,

args.caller());

}

if (StudnetApprovalResubmitActionMgr.dialogOk())

{

if (StudnetApprovalResubmitActionMgr.parmSubmit())

{

StudnetApprovalResubmitActionMgr.submit();

}

else

{

StudnetApprovalResubmitActionMgr.reSubmit();

}

if (A\_StudentDemoTableDataSource)

{

A\_StudentDemoTableDataSource.research(true);

A\_StudentDemoTableDataSource.refresh();

}

if (args.menuItemName() == menuitemActionStr(StudentWorkflowTypeSubmitMenuItem) ||

args.menuItemName() == menuitemActionStr(StudnetApprovalResubmitMenuItem))

{

args.caller().updateWorkflowControls();

}

}

}

Thank You

Thank You