HOTEL CUSTOM WORKFLOW

By,

Mohamed Asaruteen A

**Table of Contents**

**1) Introduction --------------------------------------------- 3**

**2)** **Document Scope and Purpose ----------------------- 3**

**3) System Environment ---------------------------------- 3**

**4) UI Design ------------------------------------------------ 3**

**5) Environment Setup ----------------------------------- 4**

**6) Work flow diagram------------------------------------ 17**

**7) Output --------------------------------------------------- 20**

**1 Introduction**

This introduction provides an overview of the *Hotel Custom Workflow* in Liferay DXP 7.1. It includes the purpose and design approach, main component design and high level system environment of the project.

**2 Document Scope and Purpose**

This document provides a description of the technical design for Hotel custom work flow in Liferay DXP, This document’s primary purpose is to describe the technical vision for how business requirements will be realized. This document provides an architectural overview of the system to depict different aspects of the system. This document also functions as a foundational reference point for developers.

**3 System Environment**

* Eclipse with Liferay IDE
* My SQL 5.7
* Gradle
* My SQL Workbench 8.0
* XML Designer
* Liferay CE Tomcat 7.1.2 ga3
* Node
* NPM latest version
* Yo,Gulp and liferay theme generator

**4 UI Design**

Creating Liferay gulp theme using liferay theme generator , It default support Bootstrap4,Clay css and font awesome 3 while we using this components to design our project look and feel

**5 Environment Setup**

Liferay Plugin SDK follows plugin directory where one can plugin can hold multiple portlets and web resources. One Plugin cannot use as dependency into another, but in OSGI context, one module can be added dependency into another by exporting packages.

We will create two modules

* new-workflow-web   :  This module holds web mvc resources and imports business logic from leave-core
* new-workflow-core  :  This module holds service builder generated resources and exports business layer

Edit Service.xml file and update the package-path to some meaningful package name and name space also

<?xml version=*"1.0"*?>

<!DOCTYPE service-builder PUBLIC "-//Liferay//DTD Service Builder 7.0.0//EN" "http://www.liferay.com/dtd/liferay-service-builder\_7\_0\_0.dtd">

<service-builder package-path=***"com.demo.workflow.core"***>

<namespace>demo</namespace>

<entity local-service=***"true"*** name=*"Hotel"* remote-service=***"false"*** uuid=***"true"***>

<!-- PK fields -->

<column name=*"orderId"* primary=***"true"*** type=***"long"*** />

<!-- Group instance -->

<column name=*"groupId"* type=***"long"*** />

<!-- Audit fields -->

<column name=*"companyId"* type=***"long"*** />

<column name=*"userId"* type=***"long"*** />

<column name=*"userName"* type=***"String"*** />

<column name=*"createDate"* type=***"Date"*** />

<column name=*"modifiedDate"* type=***"Date"*** />

<!-- Other fields -->

<column name=*"pizza"* type=***"int"*** />

<column name=*"burger"* type=***"int"*** />

<column name=*"Doughnut"* type=***"int"*** />

<column name=*"status"* type=***"int"*** />

<column name=*"statusByUserId"* type=***"long"*** />

<column name=*"statusByUserName"* type=***"String"*** />

<column name=*"statusDate"* type=***"Date"*** />

<!-- Order -->

<order by=***"asc"***>

<order-column name=***"createDate"*** />

</order>

<!-- Finder methods -->

<finder name=*"userId"* return-type=***"Collection"***>

<finder-column name=***"userId"*** />

</finder>

<finder name=*"status"* return-type=***"Collection"***>

<finder-column name=***"groupId"***></finder-column>

<finder-column name=***"status"***></finder-column>

</finder>

<!-- References -->

<reference entity=*"AssetEntry"* package-path=*"com.liferay.portlet.asset"* />

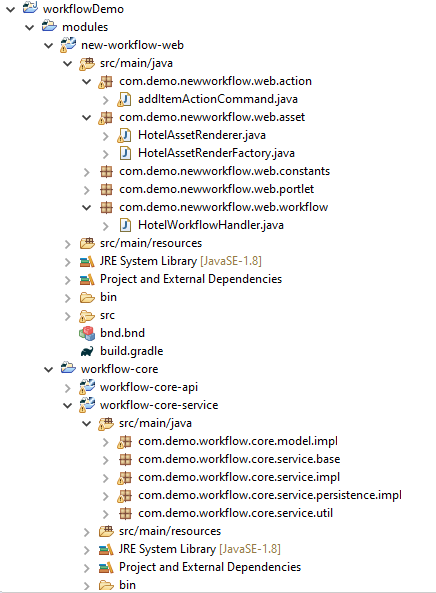
<reference entity=*"AssetTag"* package-path=*"com.liferay.portlet.asset"* />

</entity>

</service-builder>

Now run the gradle task : buildService in new-workflow-core gradle task side panel and update the export packages on leave-core module.

The Project structure is:



Currently we have service and custom portlet once everything done need to be change our custom entity as Asset

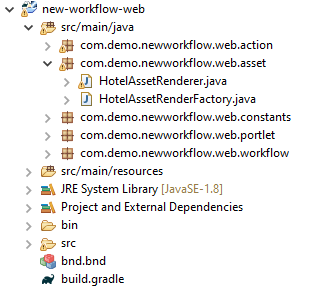
* The Entity need to be displayed on the user interface.  Liferay provides Asset Publisher API for default rendering of custom entity attributes such as title and description.
* Liferay also provides custom AssetRenderer API to display additional attributes and editing entity

Steps for change custom entity to Asset

Create Asset Renderer

Create AssetRenderer Factory OSGI service  to create AssetRenderer

Create the packages “com.demo.newworkflow.web.asset” and Create classes HotelAssetRenderer and HotelAssetRenderFactory



HotelAssetRenderer class is :-

**package** com.demo.newworkflow.web.asset;

**import** java.util.Locale;

**import** javax.portlet.PortletRequest;

**import** javax.portlet.PortletResponse;

**import** javax.servlet.http.HttpServletRequest;

**import** javax.servlet.http.HttpServletResponse;

**import** com.demo.workflow.core.model.Hotel;

**import** com.liferay.asset.kernel.model.BaseJSPAssetRenderer;

**import** com.liferay.portal.kernel.util.ResourceBundleLoader;

**public** **class** HotelAssetRenderer **extends** BaseJSPAssetRenderer<Hotel> {

**private** **final** Hotel hotel;

**private** **final** ResourceBundleLoader resourceBundleLoader;

**public** HotelAssetRenderer(Hotel hotel, ResourceBundleLoader resourceBundleLoader) {

**this**.hotel = hotel;

**this**.resourceBundleLoader = resourceBundleLoader;

}

@Override

**public** Hotel getAssetObject() {

**return** hotel;

}

@Override

**public** **long** getGroupId() {

**return** hotel.getGroupId();

}

@Override

**public** **long** getUserId() {

**return** hotel.getUserId();

}

@Override

**public** String getUserName() {

**return** hotel.getUserName();

}

@Override

**public** String getUuid() {

**return** hotel.getUuid();

}

@Override

**public** String getClassName() {

**return** Hotel.**class**.getName();

}

@Override

**public** **long** getClassPK() {

**return** hotel.getOrderId();

}

@Override

**public** String getSummary(PortletRequest portletRequest, PortletResponse portletResponse) {

**return** "pizza ->" + hotel.getPizza()+ "burger ->" + hotel.getBurger() +"Doughnut ->"+ hotel.getDoughnut() + "by "+hotel.getUserName();

}

@Override

**public** **int** getStatus() {

**return** hotel.getStatus();

}

@Override

**public** String getTitle(Locale locale) {

**return** hotel.getUserName()+" order";

}

@Override

**public** **int** getAssetRendererType() {

**return** **super**.getAssetRendererType();

}

@Override

**public** String getJspPath(HttpServletRequest request, String template) {

**return** "/hotelAssetInfo.jsp";

}

@Override

**public** **boolean** include(HttpServletRequest request, HttpServletResponse response, String template) **throws** Exception {

request.setAttribute("hotelEntry", hotel);

**return** **super**.include(request, response, template);

}

}

HotelAssetRenderFactory class is :-

package com.demo.newworkflow.web.asset;

/\*\*

\* @author PONSELVAM

\*

\*/

import javax.servlet.ServletContext;

import org.osgi.service.component.annotations.Component;

import org.osgi.service.component.annotations.Reference;

import com.demo.newworkflow.web.constants.NewWorkflowWebPortletKeys;

import com.demo.workflow.core.model.Hotel;

import com.demo.workflow.core.service.HotelLocalService;

import com.liferay.asset.kernel.model.AssetRenderer;

import com.liferay.asset.kernel.model.AssetRendererFactory;

import com.liferay.asset.kernel.model.BaseAssetRendererFactory;

import com.liferay.portal.kernel.exception.PortalException;

import com.liferay.portal.kernel.util.ResourceBundleLoader;

@Component(

immediate = true,

property = {

"javax.portlet.name=" + NewWorkflowWebPortletKeys.NewWorkflowWeb,

},

service = AssetRendererFactory.class

)

public class HotelAssetRenderFactory extends BaseAssetRendererFactory<Hotel> {

private HotelLocalService hotelService;

private ResourceBundleLoader resourceBundleLoader;

private ServletContext servletContext;

@Reference(unbind = "-")

protected void setHotelService(HotelLocalService hotelService) {

this.hotelService = hotelService;

}

@Reference(unbind = "-")

public void setResourceBundleLoader(ResourceBundleLoader resourceBundleLoader) {

this.resourceBundleLoader = resourceBundleLoader;

}

@Reference(unbind = "-")

public void setServletContext(ServletContext servletContext) {

this.servletContext = servletContext;

}

public HotelAssetRenderFactory() {

setClassName( HotelAssetRenderFactory.class.getName());

setCategorizable(true);

setLinkable(true);

setPortletId(NewWorkflowWebPortletKeys.NewWorkflowWeb);

setSearchable(true);

setSelectable(true);

}

@Override

public AssetRenderer<Hotel> getAssetRenderer(long classPK, int type) throws PortalException {

Hotel hotel = hotelService.getHotel(classPK);

HotelAssetRenderer assetRenders = new HotelAssetRenderer(hotel, resourceBundleLoader);

assetRenders.setAssetRendererType(type);

assetRenders.setServletContext(servletContext);

return assetRenders;

}

@Override

public String getType() {

return "hotel";

}

@Override

public String getClassName() {

return Hotel.class.getName();

}

}

Once our custom entity everything changed asset we need to assign workflow handler

Create package “com.demo.newworkflow.web.workflow” and create HotelWorkflowHandler.java class like shown below

**package** com.demo.newworkflow.web.workflow;

**import** java.io.Serializable;

**import** java.util.Locale;

**import** java.util.Map;

**import** org.osgi.service.component.annotations.Component;

**import** org.osgi.service.component.annotations.Reference;

**import** com.demo.workflow.core.model.Hotel;

**import** com.demo.workflow.core.service.HotelLocalService;

**import** com.liferay.portal.kernel.exception.PortalException;

**import** com.liferay.portal.kernel.service.ServiceContext;

**import** com.liferay.portal.kernel.util.GetterUtil;

**import** com.liferay.portal.kernel.workflow.BaseWorkflowHandler;

**import** com.liferay.portal.kernel.workflow.WorkflowConstants;

**import** com.liferay.portal.kernel.workflow.WorkflowHandler;

@Component(

property = {"model.class.name=com.demo.workflow.core.model.Hotel"},

service = WorkflowHandler.**class**

)

**public** **class** HotelWorkflowHandler **extends** BaseWorkflowHandler<Hotel>{

**private** HotelLocalService hotelService;

@Reference(unbind = "-")

**protected** **void** setHotelService(HotelLocalService hotelService) {

**this**.hotelService = hotelService;

}

@Override

**public** String getClassName() {

**return** Hotel.**class**.getName();

}

@Override

**public** String getType(Locale locale) {

**return** "hotel";

}

@Override

**public** Hotel updateStatus(**int** status, Map<String, Serializable> workflowContext) **throws** PortalException {

**long** userId = GetterUtil.*getLong*((String)workflowContext.get(WorkflowConstants.***CONTEXT\_USER\_ID***));

**long** orderId = GetterUtil.*getLong*((String)workflowContext.get(WorkflowConstants.***CONTEXT\_ENTRY\_CLASS\_PK***));

ServiceContext serviceContext = (ServiceContext)workflowContext.get("serviceContext");

Hotel hotel = hotelService.updateStatus(userId, orderId, status, serviceContext);

**return** hotel;

}

}

Once Asset and workflow done respective we need to update our HotelLocalServiceImpl file also.

/\*\*

\* Copyright (c) 2000-present Liferay, Inc. All rights reserved.

\*

\* This library is free software; you can redistribute it and/or modify it under

\* the terms of the GNU Lesser General Public License as published by the Free

\* Software Foundation; either version 2.1 of the License, or (at your option)

\* any later version.

\*

\* This library is distributed in the hope that it will be useful, but WITHOUT

\* ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS

\* FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more

\* details.

\*/

package com.demo.workflow.core.service.impl;

import java.util.Date;

import java.util.List;

import com.demo.workflow.core.model.Hotel;

import com.demo.workflow.core.model.Leave;

import com.demo.workflow.core.service.HotelLocalService;

import com.demo.workflow.core.service.HotelLocalServiceUtil;

import com.demo.workflow.core.service.base.HotelLocalServiceBaseImpl;

import com.liferay.asset.kernel.model.AssetEntry;

import com.liferay.counter.kernel.service.CounterLocalServiceUtil;

import com.liferay.portal.kernel.exception.PortalException;

import com.liferay.portal.kernel.model.User;

import com.liferay.portal.kernel.search.Indexer;

import com.liferay.portal.kernel.search.IndexerRegistryUtil;

import com.liferay.portal.kernel.service.ServiceContext;

import com.liferay.portal.kernel.util.ContentTypes;

import com.liferay.portal.kernel.workflow.WorkflowConstants;

import com.liferay.portal.kernel.workflow.WorkflowHandlerRegistryUtil;

/\*\*

\* The implementation of the hotel local service.

\*

\* <p>

\* All custom service methods should be put in this class. Whenever methods are added, rerun ServiceBuilder to copy their definitions into the {@link com.demo.workflow.core.service.HotelLocalService} interface.

\*

\* <p>

\* This is a local service. Methods of this service will not have security checks based on the propagated JAAS credentials because this service can only be accessed from within the same VM.

\* </p>

\*

\* @author Brian Wing Shun Chan

\* @see HotelLocalServiceBaseImpl

\* @see com.demo.workflow.core.service.HotelLocalServiceUtil

\*/

public class HotelLocalServiceImpl extends HotelLocalServiceBaseImpl {

/\*

\* NOTE FOR DEVELOPERS:

\*

\* Never reference this class directly. Always use {@link com.demo.workflow.core.service.HotelLocalServiceUtil} to access the hotel local service.

\*

\*/

public Hotel addItemsInList(ServiceContext serviceContext,int pizzaConut,int burgerCount,int DoughnutCount){

long orderId = CounterLocalServiceUtil.increment(Hotel.class.getName());

Hotel hotel = null;

try{

User user = userLocalService.getUser(serviceContext.getUserId());

hotel = hotelLocalService.createHotel(orderId);

hotel.setPizza(pizzaConut);

hotel.setBurger(burgerCount);

hotel.setDoughnut(DoughnutCount);

hotel.setUserId(serviceContext.getUserId());

hotel.setUserName(user.getFullName());

hotel.setCompanyId(serviceContext.getCompanyId());

hotel.setGroupId(serviceContext.getScopeGroupId());

hotel.setStatus(WorkflowConstants.STATUS\_DRAFT);

hotel.setStatusByUserId(user.getUserId());

hotel.setStatusDate(new Date());

hotel.setStatusByUserName(user.getFirstName());

hotel.setStatusByUserUuid(user.getUserUuid());

hotel = hotelLocalService.addHotel(hotel);

AssetEntry assetEntry = assetEntryLocalService.updateEntry( user.getUserId(), serviceContext.getScopeGroupId(), new Date(),

new Date(), Hotel.class.getName(),hotel.getOrderId(), hotel.getUuid(), 0, null, null, true, false, new Date(), null,

new Date(), null, ContentTypes.TEXT\_HTML, hotel.getUserName(), hotel.getUserName(), null, null, null, 0, 0, null);

Indexer<Hotel> indexer = IndexerRegistryUtil.nullSafeGetIndexer(Hotel.class);

indexer.reindex(hotel);

WorkflowHandlerRegistryUtil.startWorkflowInstance(hotel.getCompanyId(), hotel.getGroupId(), hotel.getUserId(), Hotel.class.getName(),hotel.getPrimaryKey(), hotel, serviceContext);

}catch(PortalException e){

e.printStackTrace();

}

return hotel;

}

public Hotel updateStatus(long userId,long orderId,int status,ServiceContext serviceContext){

Hotel hotel = hotelPersistence.fetchByPrimaryKey(orderId);

hotel.setStatus(status);

hotel.setStatusByUserId(userId);

hotel.setStatusDate(new Date());

User user = null;

try {

user = userLocalService.getUser(userId);

hotel.setStatusByUserName(user.getFullName());

hotel.setStatusByUserUuid(user.getUserUuid());

} catch (PortalException e) {

e.printStackTrace();

}

hotel = hotelPersistence.update(hotel);

try {

if (status == WorkflowConstants.STATUS\_APPROVED) {

// update the asset status to visibile

assetEntryLocalService.updateEntry(Hotel.class.getName(), orderId, new Date(),null, true, true);

} else {

// set leave entity status to false

assetEntryLocalService.updateVisible(Hotel.class.getName(), orderId, false);

}

} catch (Exception e) {

e.printStackTrace();

}

return hotel;

}

public List<Hotel> getItemsByStatus(long groupId,int status){

return hotelPersistence.findBystatus(groupId, status);

}

public List<Hotel> getItemsByStatus(long groupId,int status,int start,int end){

return hotelPersistence.findBystatus(groupId, status, start, end);

}

}

After we need to do gulp build service next gradle refresh for project and do gulp build and deploy for new-workflow-service

Once our service was ready paste below code in new-workflow-web portlet view.jsp and addItemActionCommand.java

view.jsp :-

<%@ include file=*"/init.jsp"* %>

<liferay-portlet:actionURL name=*"add\_items"* var=*"addItems"*>

<portlet:param name=*"mvcActionCommand"* value=*"add\_items"* />

</liferay-portlet:actionURL>

<aui:form action="<%= addItems %>" cssClass=*"container-fluid-1280"* method=*"post"* name=*"fm"*>

<aui:fieldset markupView=*"lexicon"*>

<div class=*"card-deck"*>

<div class=*"card"*>

<img class=*"card-img-top"* src=*"*<%=request.getContextPath()%>*/images/pizza.jpg"* alt=*"Card image cap"*>

<div class=*"card-body"*>

<h4 class=*"card-title mt-2"*>Pizza</h4>

<div class=*"d-flex justify-content-between mt-2"*>

<div class=*"price-field"*>

<h4>$ 4</h4>

</div>

<div class=*"Quantity-field"*>

<aui:select name=*"pizza"* label=*""*>

<aui:option value=*"0"*>0</aui:option>

<aui:option value=*"1"*>1</aui:option>

<aui:option value=*"2"*>2</aui:option>

<aui:option value=*"3"*>3</aui:option>

<aui:option value=*"4"*>4</aui:option>

<aui:option value=*"5"*>5</aui:option>

<aui:option value=*"6"*>6</aui:option>

<aui:option value=*"7"*>7</aui:option>

<aui:option value=*"8"*>8</aui:option>

<aui:option value=*"9"*>9</aui:option>

<aui:option value=*"10"*>10</aui:option>

</aui:select>

</div>

</div>

<p class=*"card-text"*>This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.</p>

</div>

</div>

<div class=*"card"*>

<img class=*"card-img-top"* src=*"*<%=request.getContextPath()%>*/images/buger.jpg"* alt=*"Card image cap"*>

<div class=*"card-body"*>

<h5 class=*"card-title mt-2"*>Burger</h5>

<div class=*"d-flex justify-content-between mt-2"*>

<div class=*"price-field"*>

<h4>$ 3</h4>

</div>

<div class=*"Quantity-field"*>

<aui:select name=*"burger"* label=*""*>

<aui:option value=*"0"*>0</aui:option>

<aui:option value=*"1"*>1</aui:option>

<aui:option value=*"2"*>2</aui:option>

<aui:option value=*"3"*>3</aui:option>

<aui:option value=*"4"*>4</aui:option>

<aui:option value=*"5"*>5</aui:option>

<aui:option value=*"6"*>6</aui:option>

<aui:option value=*"7"*>7</aui:option>

<aui:option value=*"8"*>8</aui:option>

<aui:option value=*"9"*>9</aui:option>

<aui:option value=*"10"*>10</aui:option>

</aui:select>

</div>

</div>

<p class=*"card-text"*>This card has supporting text below as a natural lead-in to additional content.</p>

</div>

</div>

<div class=*"card"*>

<img class=*"card-img-top"* src=*"*<%=request.getContextPath()%>*/images/donut.jpg"* alt=*"Card image cap"*>

<div class=*"card-body"*>

<h4 class=*"card-title mt-2"*>Doughnut</h4>

<div class=*"d-flex justify-content-between mt-2"*>

<div class=*"price-field"*>

<h4>$ 2</h4>

</div>

<div class=*"Quantity-field"*>

<aui:select name=*"Doughnut"* label=*""*>

<aui:option value=*"0"*>0</aui:option>

<aui:option value=*"1"*>1</aui:option>

<aui:option value=*"2"*>2</aui:option>

<aui:option value=*"3"*>3</aui:option>

<aui:option value=*"4"*>4</aui:option>

<aui:option value=*"5"*>5</aui:option>

<aui:option value=*"6"*>6</aui:option>

<aui:option value=*"7"*>7</aui:option>

<aui:option value=*"8"*>8</aui:option>

<aui:option value=*"9"*>9</aui:option>

<aui:option value=*"10"*>10</aui:option>

</aui:select>

</div>

</div>

<p class=*"card-text"*>This is a wider card with supporting text below as a natural lead-in to additional content. This card has even longer content than the first to show that equal height action.</p>

</div>

</div>

</div>

<aui:button type=*"submit"* label=*"Order"* name=*"Order"* cssClass=*"mt-5 pull-right"*></aui:button>

</aui:fieldset>

</aui:form>

addItemActionCommand.java :-

/\*\*

\*

\*/

package com.demo.newworkflow.web.action;

import javax.portlet.ActionRequest;

import javax.portlet.ActionResponse;

import org.osgi.service.component.annotations.Component;

import org.osgi.service.component.annotations.Reference;

import com.demo.newworkflow.web.constants.NewWorkflowWebPortletKeys;

import com.demo.workflow.core.model.Hotel;

import com.demo.workflow.core.service.HotelLocalService;

import com.liferay.portal.kernel.portlet.bridges.mvc.BaseMVCActionCommand;

import com.liferay.portal.kernel.portlet.bridges.mvc.MVCActionCommand;

import com.liferay.portal.kernel.service.ServiceContext;

import com.liferay.portal.kernel.service.ServiceContextFactory;

import com.liferay.portal.kernel.theme.ThemeDisplay;

import com.liferay.portal.kernel.util.ParamUtil;

import com.liferay.portal.kernel.util.PortalUtil;

import com.liferay.portal.kernel.util.WebKeys;

@Component(

property = {

"javax.portlet.name=" + NewWorkflowWebPortletKeys.NewWorkflowWeb,

"mvc.command.name=add\_items"

},

service = MVCActionCommand.class

)

public class addItemActionCommand extends BaseMVCActionCommand{

private HotelLocalService hotelService;

@Reference(unbind = "-")

protected void setLeaveService(HotelLocalService hotelService) {

this.hotelService = hotelService;

}

@Override

protected void doProcessAction(ActionRequest actionRequest, ActionResponse actionResponse) throws Exception {

// TODO Auto-generated method stub

ThemeDisplay themeDisplay = (ThemeDisplay) actionRequest.getAttribute(WebKeys.THEME\_DISPLAY);

long userId = themeDisplay.getUserId();

ServiceContext serviceContext=ServiceContextFactory.getInstance(Hotel.class.getName(),actionRequest);

serviceContext.setUserId(userId);

serviceContext.setCompanyId(PortalUtil.getCompanyId(actionRequest));

int pizzaConut = ParamUtil.getInteger(actionRequest, "pizza");

int burgerCount = ParamUtil.getInteger(actionRequest, "burger");

int DoughnutCount = ParamUtil.getInteger(actionRequest, "Doughnut");

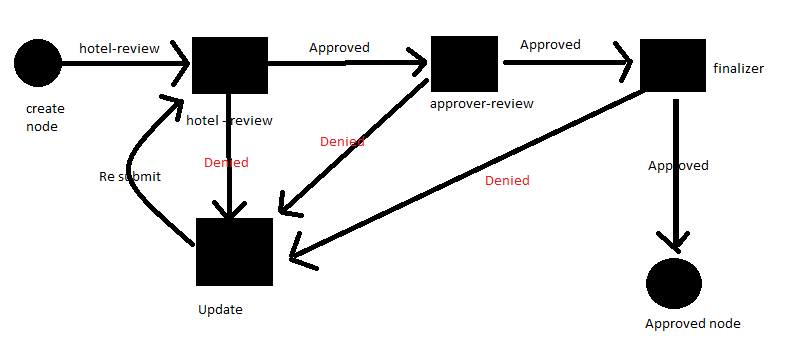
Hotel hotel = hotelService.addItemsInList(serviceContext, pizzaConut, burgerCount, DoughnutCount);

}

}

Once our web and core everything is active mode we need to create workflow xml based on below diagram

**6 Work flow diagram:-**



XML Code:-

<?xml version="1.0"?>

<workflow-definition

xmlns="urn:liferay.com:liferay-workflow\_7.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="urn:liferay.com:liferay-workflow\_7.0.0 http://www.liferay.com/dtd/liferay-workflow-definition\_7\_0\_0.xsd">

<name>Hotel Workflow Approval</name>

<description>Workflow assets must be approved first by reviewer second by approver and then by final approver.</description>

<version>1</version>

<state>

<name>created</name>

<initial>true</initial>

<transitions>

<transition>

<name>Hotel Review</name>

<target>hotel-review</target>

<default>true</default>

</transition>

</transitions>

</state>

<task>

<name>update</name>

<actions>

<notification>

<name>Creator Modification Notification</name>

<template>Your order was rejected by ${userName}, please modify and resubmit.</template>

<template-language>freemarker</template-language>

<notification-type>email</notification-type>

<notification-type>user-notification</notification-type>

<execution-type>onAssignment</execution-type>

</notification>

</actions>

<assignments>

<user/>

</assignments>

<transitions>

<transition>

<name>Resubmit</name>

<target>hotel-review</target>

<default>true</default>

</transition>

</transitions>

</task>

<task>

<name>hotel-review</name>

<actions>

<notification>

<name>Hotel Content Reviewer Notification</name>

<template>You have a new order from ${userName} waiting for your review.</template>

<template-language>freemarker</template-language>

<notification-type>email</notification-type>

<notification-type>user-notification</notification-type>

<execution-type>onAssignment</execution-type>

</notification>

</actions>

<assignments>

<roles>

<role>

<role-type>regular</role-type>

<name>Administrator</name>

</role>

<role>

<role-type>regular</role-type>

<name>Hotel Content Reviewer</name>

</role>

</roles>

</assignments>

<transitions>

<transition>

<name>Reviewer Approved</name>

<target>approver-review</target>

<default>true</default>

</transition>

<transition>

<name>Reviewer Denied</name>

<target>update</target>

<default>false</default>

</transition>

</transitions>

</task>

<task>

<name>approver-review</name>

<actions>

<notification>

<name>Approver Content Review Notification</name>

<template>You have a new order notification from ${userName},waitiing for your review.</template>

<template-language>freemarker</template-language>

<notification-type>email</notification-type>

<notification-type>user-notification</notification-type>

<execution-type>onAssignment</execution-type>

</notification>

</actions>

<assignments>

<roles>

<role>

<role-type>regular</role-type>

<name>Administrator</name>

</role>

<role>

<role-type>regular</role-type>

<name>Hotel Approver Content Reviewer</name>

</role>

</roles>

</assignments>

<transitions>

<transition>

<name>Approver Approval</name>

<target>final-review</target>

<default>true</default>

</transition>

<transition>

<name>Approver Denied</name>

<target>update</target>

<default>false</default>

</transition>

</transitions>

</task>

<task>

<name>final-review</name>

<actions>

<notification>

<name>Final Content Review Notification</name>

<template>You have a new order notification from ${userName},waitiing for your review.</template>

<template-language>freemarker</template-language>

<notification-type>email</notification-type>

<notification-type>user-notification</notification-type>

<execution-type>onAssignment</execution-type>

</notification>

</actions>

<assignments>

<roles>

<role>

<role-type>regular</role-type>

<name>Administrator</name>

</role>

<role>

<role-type>regular</role-type>

<name>Hotel Final Content Reviewer</name>

</role>

</roles>

</assignments>

<transitions>

<transition>

<name>Final Approval</name>

<target>approved</target>

<default>true</default>

</transition>

<transition>

<name>Final Denied</name>

<target>update</target>

<default>false</default>

</transition>

</transitions>

</task>

<state>

<name>approved</name>

<actions>

<action>

<name>approve</name>

<script>

<![CDATA[

import com.liferay.portal.kernel.workflow.WorkflowStatusManagerUtil;

import com.liferay.portal.kernel.workflow.WorkflowConstants;

WorkflowStatusManagerUtil.updateStatus(WorkflowConstants.getLabelStatus("approved"), workflowContext);

]]>

</script>

<script-language>groovy</script-language>

<execution-type>onEntry</execution-type>

</action>

</actions>

</state>

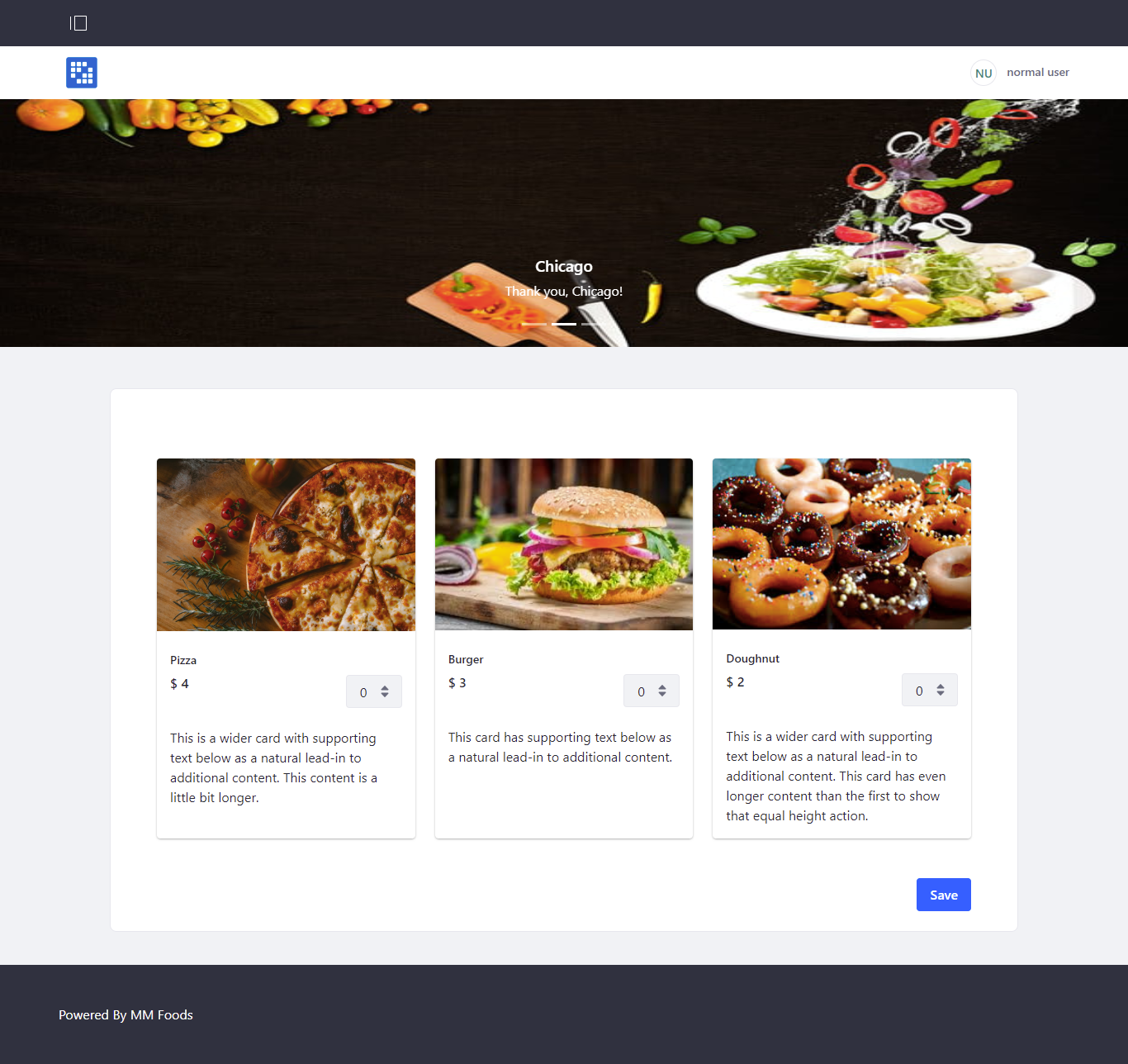
</workflow-definition>

Copy the above XML code and login as a Admin in liferay portal -> control panel -> configuration -> wokflow -> add new work flow

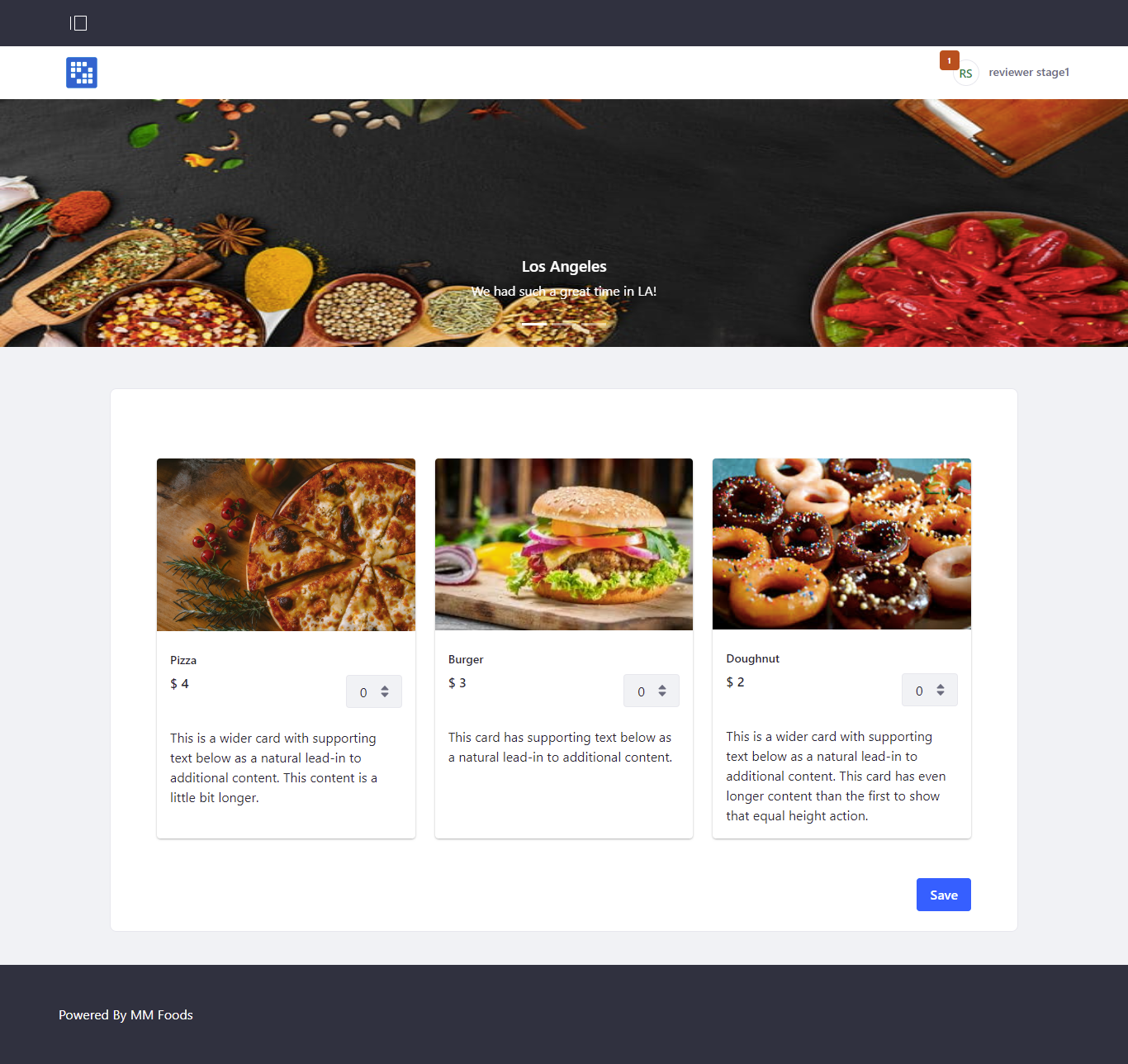
After navigate to schemes tab assign custom workflow to our module. Once everything done login with normal user sumit the form.

**7 Output:-**

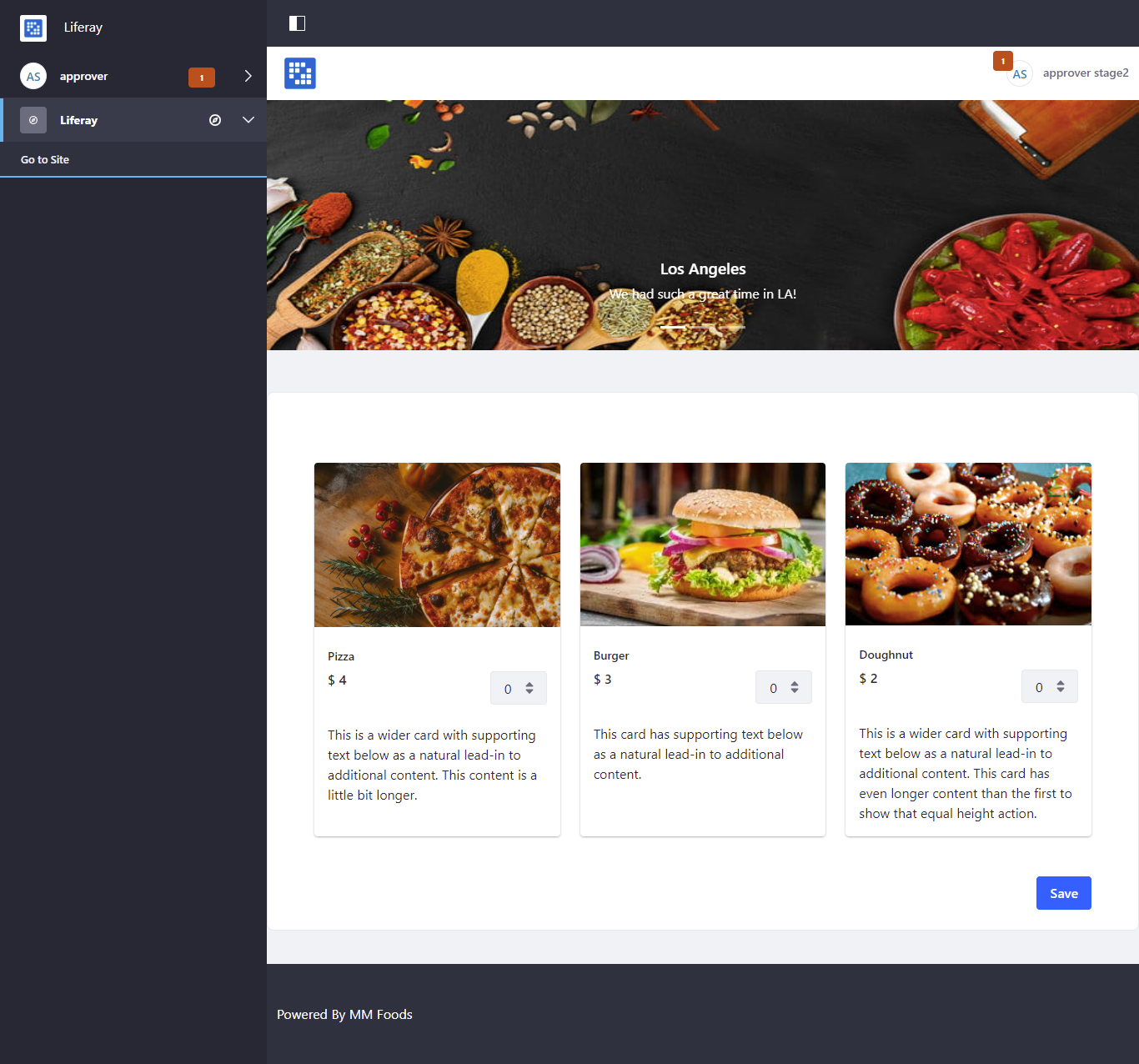
Login as a normal user



Login as a reviewer



Login as an approver:-



Login as a finalizer:-

