Introduction to Liferay Objects Workshop

So these are the exercises that will be completed during the workshop.

When using the `prepared` branch, skip ahead to Exercise 05

When using the `completed` branch, all of these steps have been completed.

Exercise 01 - Create the Picklists

We will be using three Picklists for our implementation. Navigate to the Picklists control panel and create the following Picklists with the given items:

Picklist - Bootcamps

Name: `Bootcamps` External Reference Code: `objects-bootcamps` Items:

Name	Key
`Objects`	`Objects`
`Client Extensions`	`ClientExtensions`
`Commerce`	`Commerce`

Picklist - Registration Status

Name: `Registration Status` External Reference Code: `objects-registration-status` Items:

Name	Key
Pending	Pending
Approved	Approved
Denied	Denied

Picklist - Dates

Name: `Dates` External Reference Code: `objects-bootcamp-dates` Items:

Name	Key
2025 Q1	2025Q1
2025 Q2	2025Q2
2025 Q3	2025Q3

Name	Key
2025 Q4	2025Q4

Permissions

Grant the *View* permission to the *Guest* role for all of the Picklists.

Exercise 02 - Create the Object

In this portion of the workshop, we will be preparing the new object.

Start by creating a new Object Folder:

Label: `Bootcamps` Name: `Bootcamps`

Within the new folder, create a new custom object:

Label: `Bootcamp Request` Plural Label: `Bootcamp Requests` Object Name: `BootcampRequest`

In the object fields tab, add the following:

Label	Field Name	Type	Extra Settings
`Previous Experience`	`previousExperience`	Long Text	
`Attending Date`	`attendingDate`	Picklist	Dates Picklist, Mandatory
`Attending Bootcamp`	`attendingBootcamp`	Picklist	Bootcamps Picklist, Mandatory
`Registration Status`	`registrationStatus`	Picklist	Registration Status Picklist, Mandatory, Mark as State, Default Value Pending

In the object relationships tab, add a relationship:

Label: `Attendee` Name: `attendee` Type: *One to Many* One Record Of: `User` Many Records Of: `Bootcamp Request`

Exercise 03 - Add Object Actions

We're going to add three actions to the Bootcamp Request object.

Open the Bootcamp Request object and on the actions tab, add the following:

Action Auto-Assign Attendee

Action Label: `Auto-Assign Attendee` Action Name: `autoAssignAttendee` Description: `Sets the current user as the attendee in the Bootcamp Request.` Trigger When: *On After Add* Action Then: *Update an Object Entry* Values:

Field	Input Method	New Value
Attendee	Unchecked	`currentUserId`

Action Approve Registration

Action Label: `Approve Registration` Action Name: `approveRegistration` Description: `Approves the given registration request.` Trigger When: *Standalone* Action Then: *Update an Object Entry* Values:

Field	Input Method	New Value
Registration Status	Checked	`Approved`

Error Message: `Failed approving the request.`

Action Deny Registration

Action Label: `Deny Registration` Action Name: `denyRegistration` Description: `Denies the given registration request.` Trigger When: *Standalone* Action Then: *Update an Object Entry* Values:

Field	Input Method	New Value
Registration Status	Checked	`Denied`

Error Message: `Failed denying the request.`

Exercise 04 - Completing the Object

Before publishing the object, we will update the state manager settings.

Open the Bootcamp Request object and go to the State Manager tab and edit the settings for the Registration Status:

State Name	Next Status
`Pending`	`Approved` `Denied`
`Approved`	`Denied`
`Denied`	`Approved`

On the Details tab, set the following:

Scope: *Company* Panel Link: *Object* Enable Entry History in Audit Framework: *On*

Publish the object.

Grant the *View* permission to the *Class Requestor* and *Class Approver* roles.

Adjust Role Permissions

The two custom roles need to be updated to grant appropriate permissions on entries.

For the *Class Requestor* role, find the *Bootcamp Requests* object, grant the *View* application permission, and for the *Bootcamp Requests* grant the *Add Object Entry* permission. This will give the Requestor the permission to add new bootcamp requests.

For the *Class Approver* role, find the *Bootcamp Requests* object, grant the *View* application permission, and for the *Bootcamp Request* grant the *View*, *Update*, *action.approveRegistration*, and *action.denyRegistration* permissions. This will allow the Approver the permission to see and update all requests.

Exercise 05 - Create the Registrations Page

Note: All further exercises will occur in the Masterclass site.

From the *Pages* control panel, create a new page using `Main-1` as the template. Name the page `Registrations`.

Publish the page. Set the permissions on the page so the *Course Requestor* role can view the page, and remove the *View* permission from the *Guest* and *Site Member* roles.

Go to the *Navigation Menus* control panel, to the *Main Navigation*, and add the *Registrations* page under the *Courses* navigation item.

Test the page by logging in as `student@masterclass.com` and verify the page appears.

Exercise 06 - Create the Registration Form

Log back in as `admin@masterclass.com` and return to the *Pages* control panel and edit the *Registrations* page.

In the drop zone, drop a *Container* on the page. Inside of the container, drop a *Form Container*.

For the mapping, choose the *Bootcamp Request*.

Remove the *Attendee* and *Registration Status* fields from the form.

Add a *Grid Container* with 2 columns to the form container, put the bootcamp and date fields into the grid.

Play with the styles of the elements to make the spacing and UI better.

Publish the page. You'll get a warning about missing required fields, but that's okay to publish anyway.

Test the page by logging in as `student@masterclass.com`, navigate to the page, and complete and submit the form. Use the navigation to return to the page to submit additional requests. Verify the additions by logging back in as `admin` and use the *Bootcamp Requests* control panel.

Exercise 07 - Create the My Registrations Page

As the `admin`, create a new page in the *Pages* control panel using the `Private Area` template. Name the page `My Registrations`.

Add a *Container* to the Drop Zone, then a *Collection Display* fragment to the container. Configure the collection display fragment to use the *Bootcamp Requests* collection provider.

Add a *Masterclass Heading* fragment mapped to the *Attending Bootcamp* and *Masterclass Text Block* fragments for the *Attending Date* and *Registration Status* fields.

Style as you'd like, then Publish the page.

In the *Navigation Menus* control panel, edit the *Private Area Navigation* and add the *My Registrations* page.

Test the page by logging in as `student@masterclass.com`, use the *User* menu to get to *My Learning* and find *My Registrations* on the left side, then verify the registrations are listed.

Bonus Points

Earn bonus points by modifying the *Masterclass* fragments like *Masterclass Navbar* and *Masterclass Private Header* to include the new *My Registrations* page.

Exercise 08 - Create the Approvals Segment

Create a new segment called *Approvals*, with the condition for the *Regular Role* equals *Class Approver*.

Exercise 09 - Create the Approvals View

Edit the *Private Area* master page template to allow a *Button* fragment in the drop zone.

Edit the *My Registrations* page, create a new experience named *Approvals* and use the *Approvals* segment. Ensure the experience is on top to have priority.

Add a *Masterclass Text Block* and map to the *Attendee* field to show the user who is requesting the course.

Add two *Button* fragments. Set the button types to *Action*, then on the *Action* child, map one to the *Approve Registration* (both the *Mapping* and the *Action*), and the other to *Deny Registration*. After choosing the Action, check both of the *Reload Page After* checkboxes.

Publish the page.

Test the page by logging in as `student` and note that the buttons are not visible.

Next test by logging in as `teacher@masterclass.com` and note that the buttons are visible and clicking them should change the state of the registrations.

Exercise 10 - Add the React Custom Element Client Extension

So far, we have only used Liferay OOTB tools for the form, for the listing as well as for the approvals.

Sometimes the requirements may necessatate a custom solution which we can solve using *Custom Element Client Extensions*.

A React-based custom element client extension has been pre-created for you, it is in the `client-extensions` folder.

Navigate to the `client-extensions/lr-course-registrations` folder and issue the command `blade gw clean deploy` to build and deploy the client extension.

If you can't build the project, a pre-built `lr-course-registrations.zip` file is in the root of the workspace. Copy this to the `bundles/osgi/client-extensions` folder (you might need to create this folder) to deploy the client extension.

Create a new page and drop the *Liferay React Course Registrations* on it, then publish the page. By default this is the *requestor* view.

Edit the page, find the Action menu for the *Liferay React Course Registrations* fragment and choose *Configuration*. In the dialog that opens, set the *Properties* to `admin=true`, *Save* and close the dialog and Publish the page. This view is the *approver* view.

The code for the custom element is in the repository. It leverages Liferay's React, but it also includes React Router and it demonstrates how to use the Headless APIs to access the Picklists as well as the *Bootcamp Request* objects. As a React application, the UI can be as simple or complex as requirements mandate.