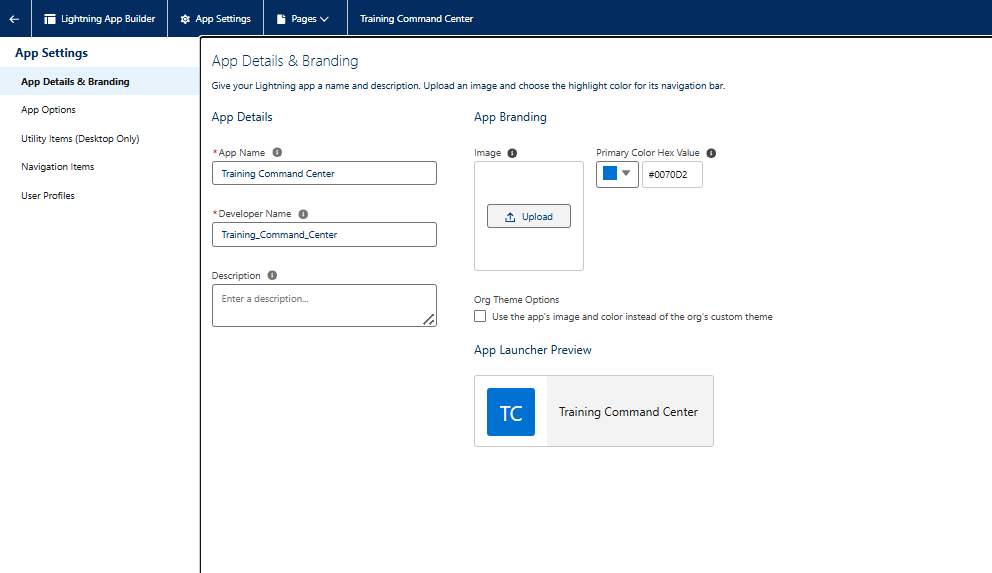
**Phase 6 Report: User Interface Development**

**Executive Summary**

Phase 6 focused on creating an interactive user interface for the **Phishing Awareness Training Tracker** application. Using Salesforce Lightning tools and **Lightning Web Components (LWC)**, I built custom record pages, app pages, and components that provide users with a seamless experience. This phase combined **Lightning App Builder** for declarative UI customization with programmatic features such as **LWC, Apex integration, wire adapters, events, and navigation services**. The result is a highly usable and mobile-friendly UI that empowers both administrators and employees.



**Lightning App Builder & Tabs**

A custom app named **Training Command Center** was created using **Lightning App Builder**. It includes navigation items for Assignments, Completions, Employees, Reports, and Dashboards. Custom tabs were configured for each custom object so that list views and records are directly accessible.

A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect.

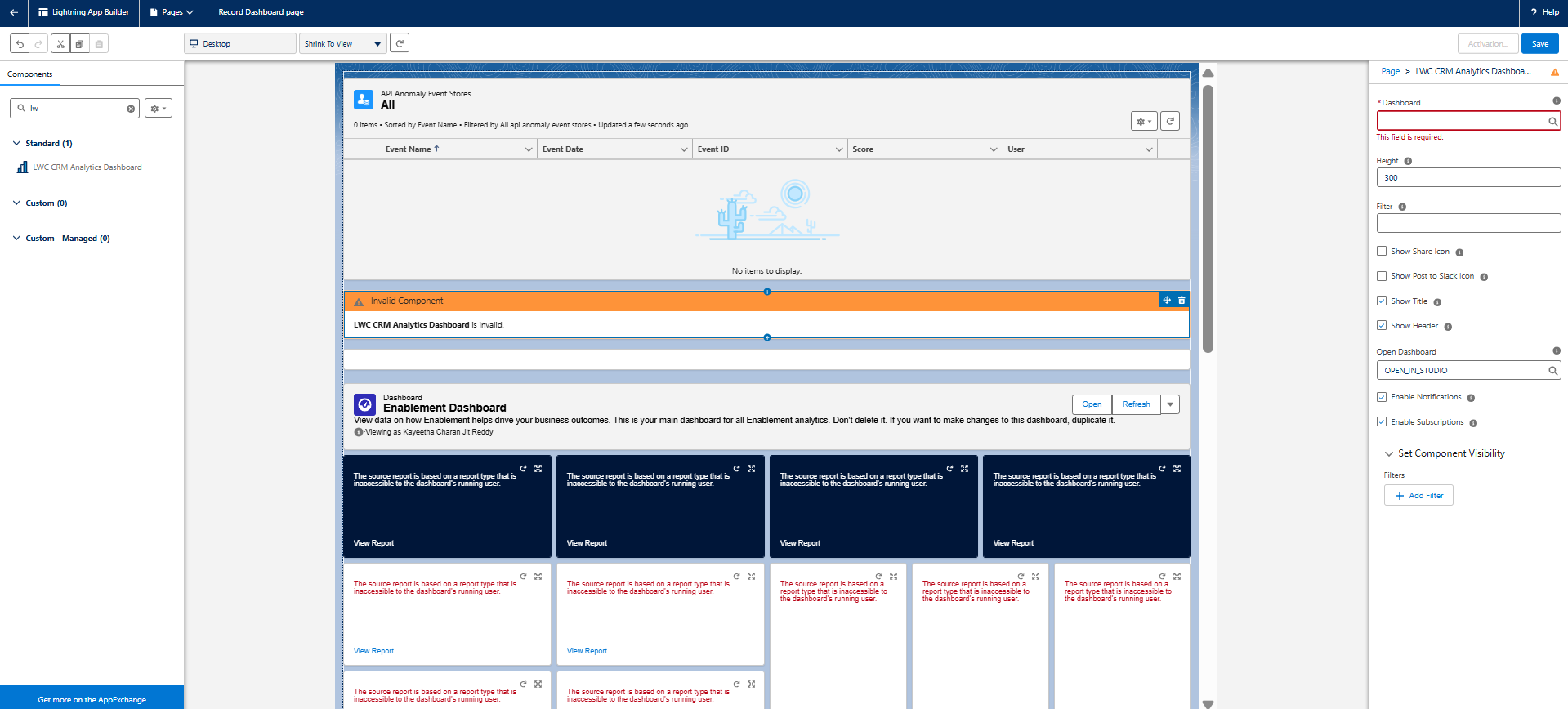
**Record Pages & Home Page Layouts**

**Assignment\_\_c Record Page** was customized with the following layout:

* **Highlights Panel**: Displays Employee, Module, Status, and Due Date.
* **Tabs**: Details, Related Lists (Completions), and Activity.
* **Related Record Component**: Quick view of Employee’s training score.
* **Custom LWC Slot**: For the quick-assign form.

A **custom Home Page** was also built showing:

* Dashboard (Assignments by Status).
* List View (Overdue Assignments).
* Rich Text message for employees.
* Custom LWC component for quick actions.



**Utility Bar**

The **Utility Bar** in the Training Command Center app was configured with:

* **Recent Items** for quick navigation.
* A **custom LWC wrapper** (optional) to provide a quick “Assign Training” action.

A screenshot of a computer

AI-generated content may be incorrect.

**Lightning Web Components (LWC)**

A custom LWC named **assignmentIntakeForm** was developed to allow administrators to quickly assign training. It uses **Apex methods, events, wire adapters, and navigation service**.

**assignmentIntakeForm.html**

<template>

<lightning-card title="Assign Training" icon-name="custom:custom63">

<div class="slds-p-around\_medium">

<lightning-input name="Name" label="Assignment Title" onchange={handleChange}></lightning-input>

<lightning-combobox name="Module\_\_c" label="Module" options={moduleOptions} onchange={handleChange}></lightning-combobox>

<lightning-input type="date" name="Assigned\_Date\_\_c" label="Assigned Date" onchange={handleChange}></lightning-input>

<lightning-input type="date" name="Due\_Date\_\_c" label="Due Date" onchange={handleChange}></lightning-input>

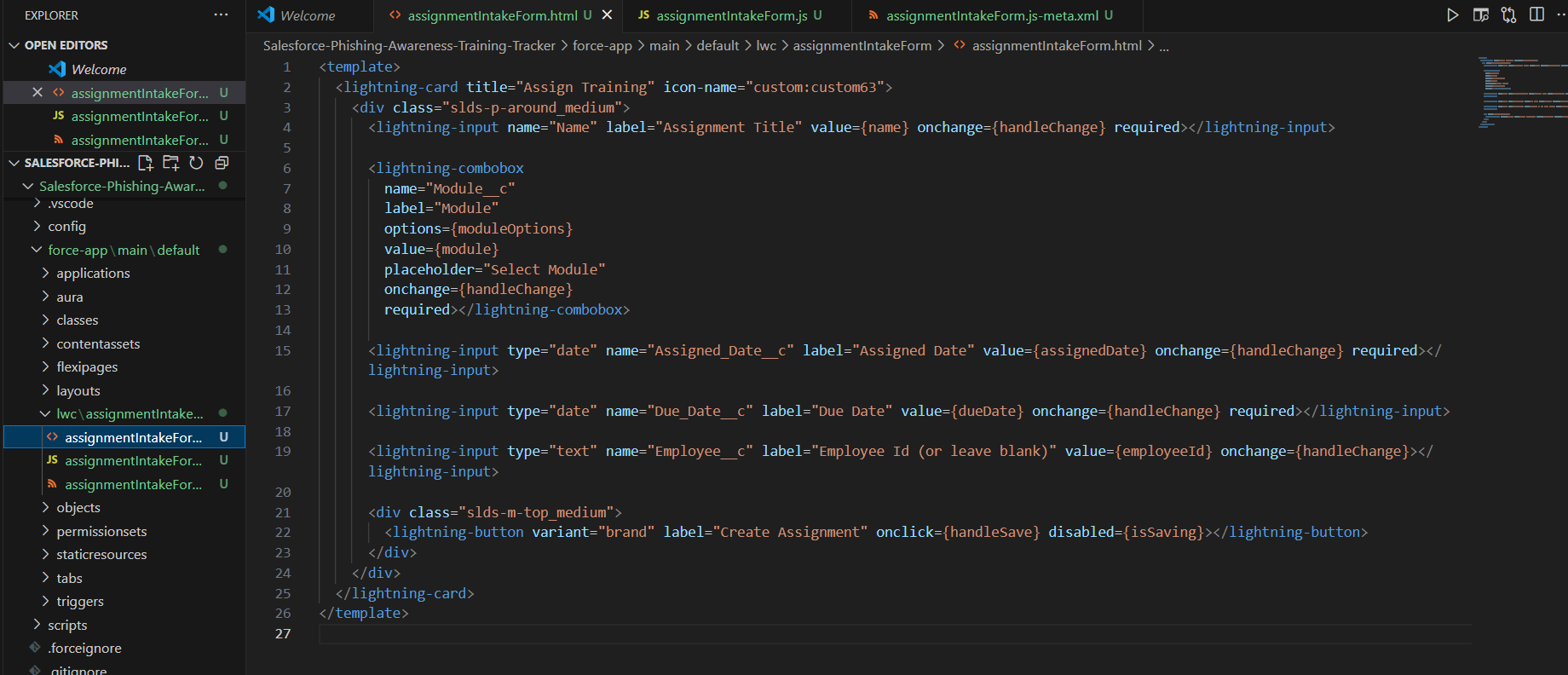
<lightning-input type="text" name="Employee\_\_c" label="Employee Id" onchange={handleChange}></lightning-input>

<lightning-button variant="brand" label="Create Assignment" onclick={handleSave}></lightning-button>

</div>

</lightning-card>

</template>



A screen shot of a computer

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect.

**Apex with LWC**

An Apex controller (AssignmentController) was created to fetch picklist values and save new Assignment records.

**AssignmentController.cls**

public with sharing class AssignmentController {

@AuraEnabled(cacheable=true)

public static Map<String, List<String>> getPicklistValues() {

Map<String, List<String>> picklistMap = new Map<String, List<String>>();

List<String> moduleValues = new List<String>();

Schema.DescribeFieldResult f = Assignment\_\_c.Module\_\_c.getDescribe();

for (Schema.PicklistEntry pe : f.getPicklistValues()) moduleValues.add(pe.getValue());

picklistMap.put('Module\_\_c', moduleValues);

return picklistMap;

}

@AuraEnabled

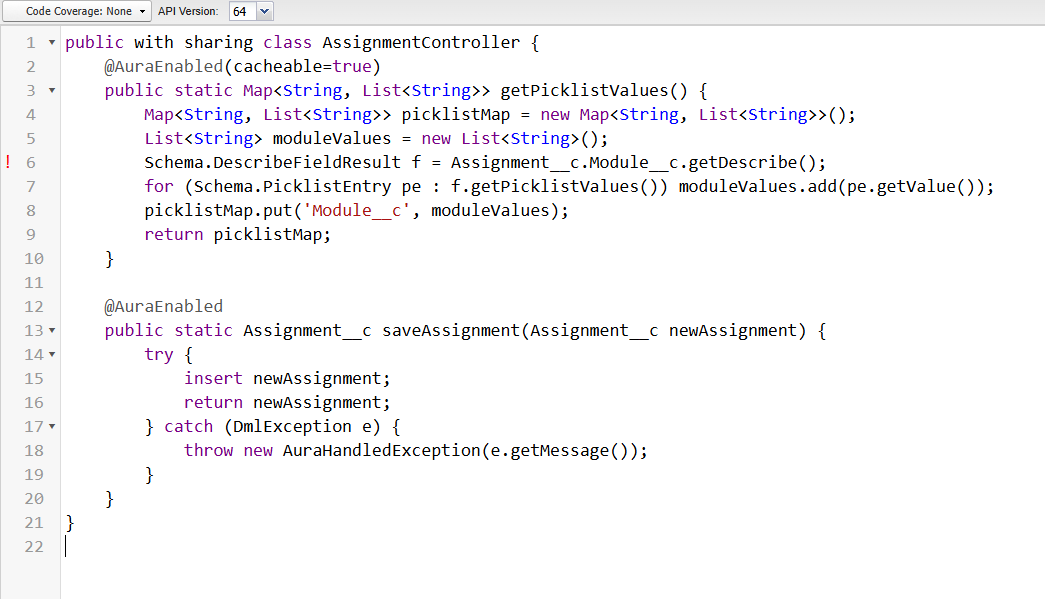
public static Assignment\_\_c saveAssignment(Assignment\_\_c newAssignment) {

insert newAssignment;

return newAssignment;

}

}



**Events, Wire Adapters & Imperative Apex Calls**

* **Wire Adapter:** Used for fetching picklist values via @AuraEnabled(cacheable=true).
* **Imperative Apex:** Used for inserting records (saveAssignment).
* **Events:** ShowToastEvent for success/error, NavigationMixin for redirect to the new record.

**Conclusion**

Phase 6 delivered a modern, interactive interface for the Training Tracker. Administrators can easily assign modules through LWCs, employees can view assignments in tailored record pages, and dashboards keep everyone informed. By combining declarative tools (App Builder, Tabs, Home Page, Utility Bar) with programmatic LWCs integrated with Apex, the project now provides a **seamless and engaging user experience**.