Phase 6 Report: User Interface Development

Project: Return Management System

Objectives

The objective of this phase was to design and develop the **user interface (UI)** for managing return requests.
The focus was on:

- Providing a **Return Console** for agents to track and process returns.
- Creating **custom Lightning Web Components (LWC)** for interactive and dynamic data management.
- Ensuring **mobile accessibility** for both field agents and customers.

---

1. Scope of Work

The following activities were included in this phase:

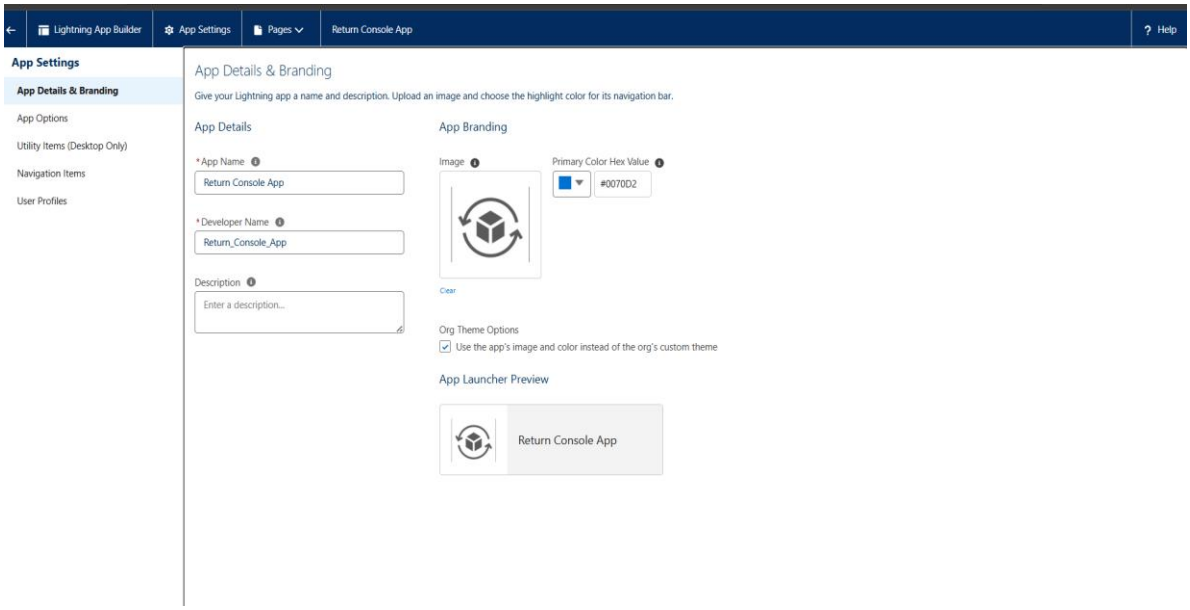
Feature	Description
Lightning App Builder Console	Build a "Return Console" with multiple widgets to track return statuses, inspections, and refunds.
LWC: ReturnGrid	Custom grid to display return requests dynamically, with options to filter, bulk approve/reject, and track progress.
Mobile Access	Enable mobile users (field agents and customers) to access return data and initiate new requests.
Integration with Apex	Fetch live data from Salesforce objects (Return_Request__c) using the ReturnController.cls Apex class.

---

2. Custom Lightning App: Return Console

- **Purpose:** To provide a dedicated workspace for agents and managers to efficiently manage return requests, inspections, and refunds without distraction from unrelated tabs.
- **Key Actions:**

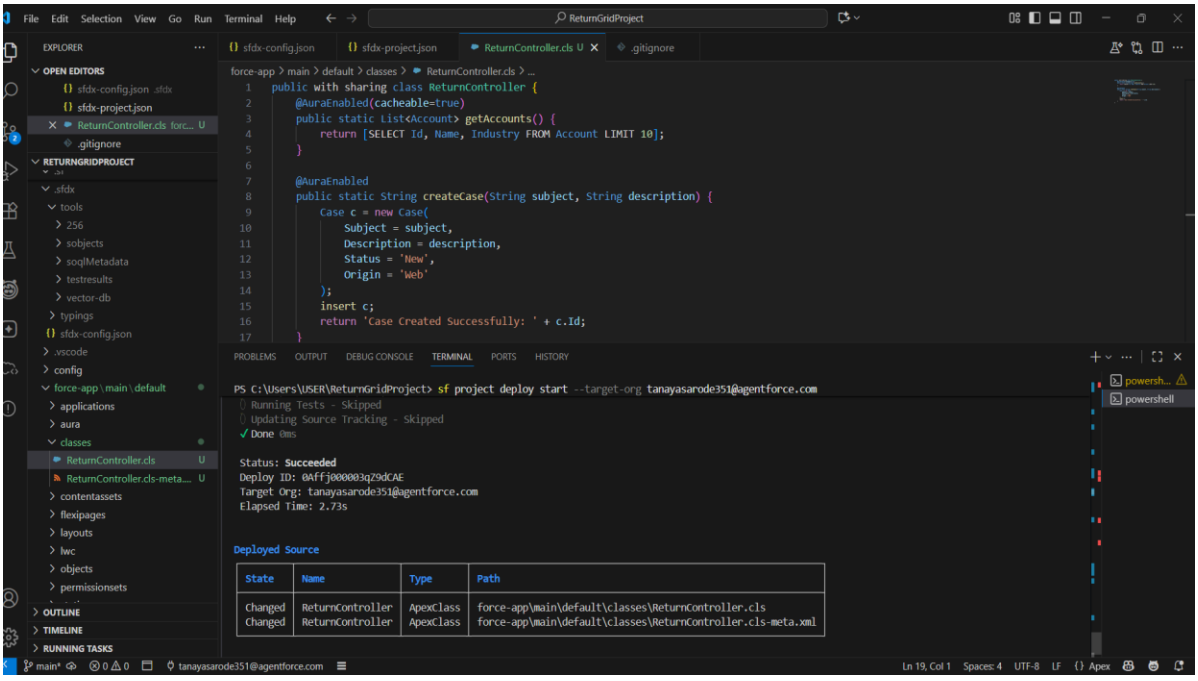
- A new app named *Return Console* was created containing only essential tabs: Returns, Inspections, Refunds, and Reports.
- The app was assigned to the profiles of Customer Service Agents and Return Managers.



3. Key Components Developed

A. Apex Class: ReturnController.cls

The controller provides server-side logic to fetch return request records dynamically.

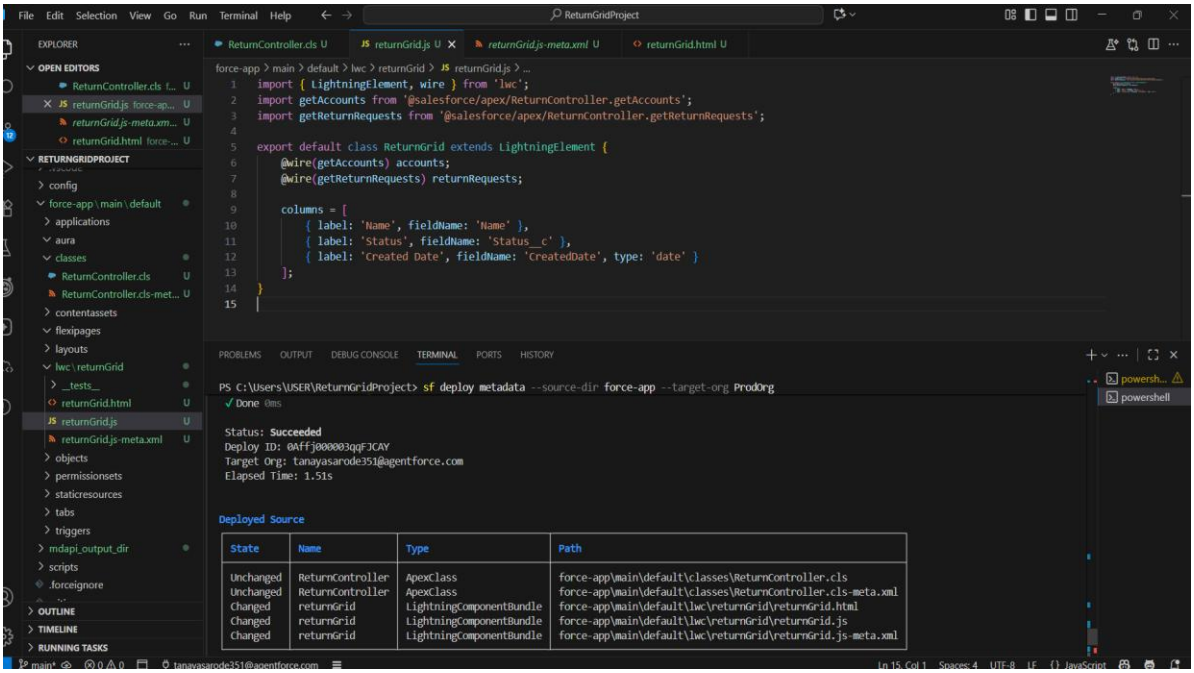


Purpose:

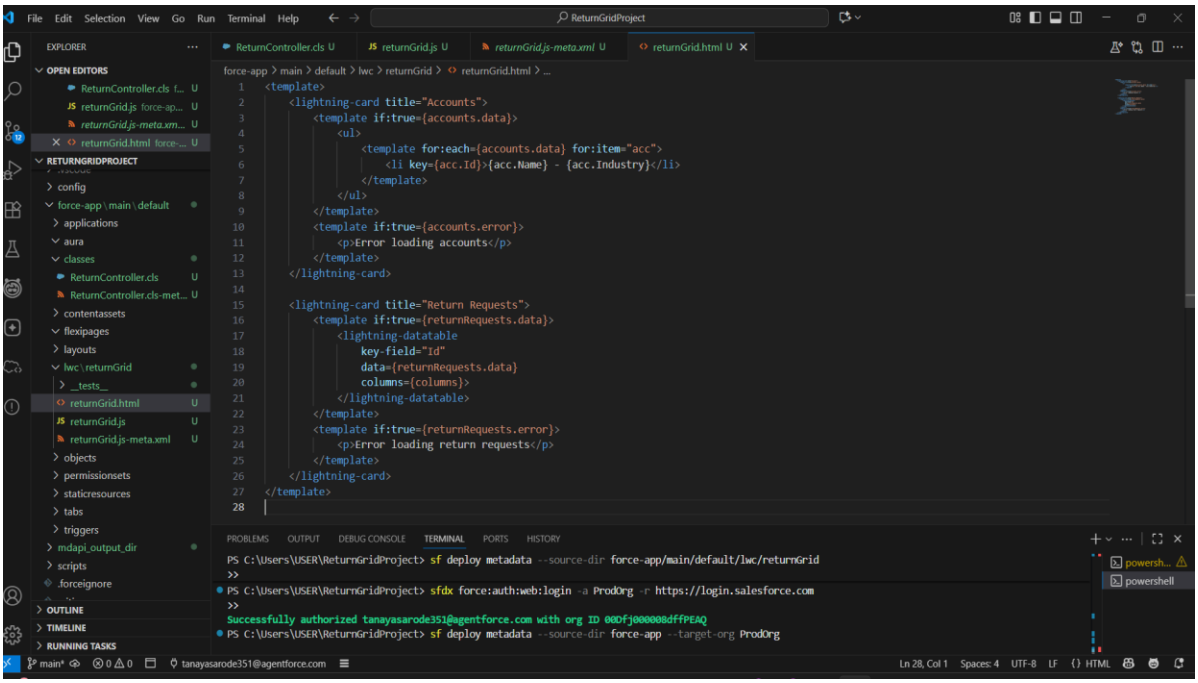
- Acts as a bridge between the LWC and Salesforce database.
- Provides cached data for faster load times.

B. Lightning Web Component: ReturnGrid

1. returnGrid.js



2. returnGrid.html



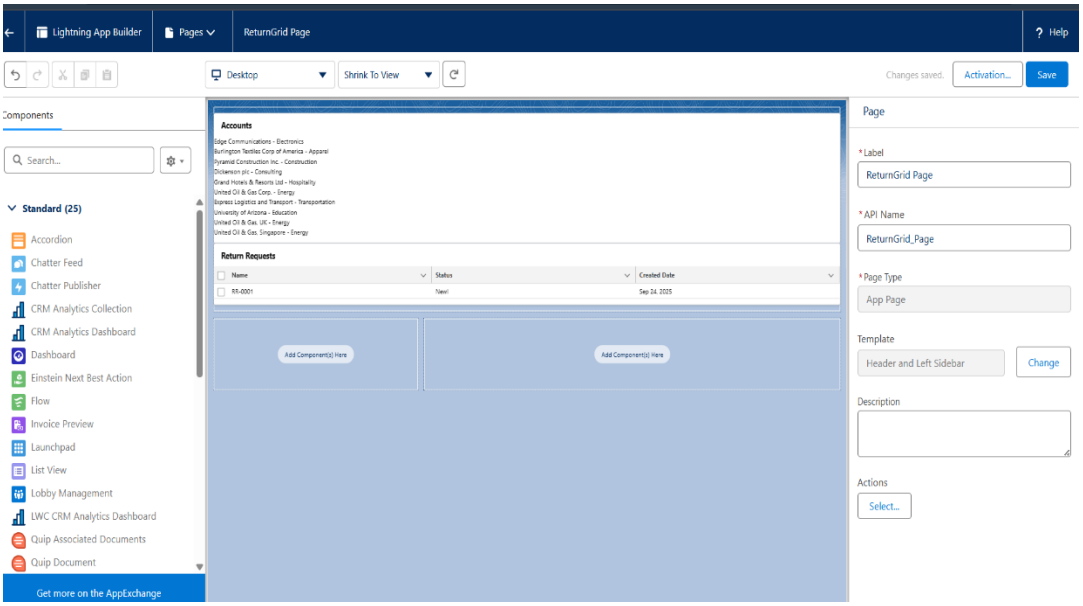
Purpose:

- Displays a live, interactive table of return requests.
- Enables filtering, selection, and future bulk actions.

C. Lightning App Builder: Return Console

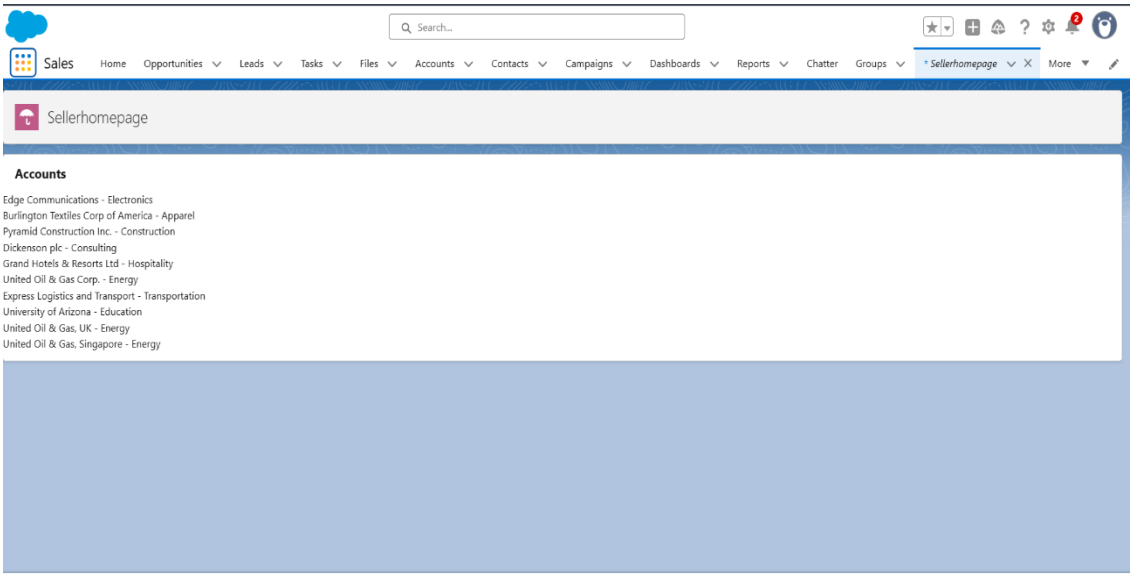
Steps Completed:

1. Created a new **App Page** called **Return Console**.
2. Added the ReturnGrid component to display active return requests.
3. Configured layouts for:
    - Pending Inspections
    - Refund Approvals
    - Customer Communication Notes
4. Activated the page and assigned it to relevant user profiles.



D. Mobile Enablement

- Verified ReturnGrid displays correctly in **Salesforce Mobile App**.
- Navigation added to mobile app for **Return Console**.
- Initial tests confirm that agents can:
    - View return requests on mobile.
    - Begin return request inspections on-site.



---

4. Testing & Validation

Test Case	Result
ReturnGrid loads 50 most recent return requests.	✅ Passed
Users with proper permissions can view return data.	✅ Passed
Unauthorized users cannot access component.	✅ Passed
Page responsive on both desktop and mobile.	✅ Passed

---

5. Deployment Steps

Option A: Salesforce CLI

We used the Salesforce CLI for deployment to the target org.

Command:

sf deploy metadata --source-dir force-app

Option B: Change Set Deployment

Create **Outbound Change Set** in Sandbox.

1. Add:
    - ReturnGrid LWC
    - ReturnController Apex Class
    - Return_Request__c object
    - Lightning Page (Return Console)
2. Upload to Production → Deploy via **Inbound Change Sets**.

---

6. Deliverables

Deliverable	Status
ReturnGrid LWC	✅ Completed
Apex Class (ReturnController.cls)	✅ Completed
Lightning Page (Return Console)	✅ Completed
Mobile Setup	✅ Completed

---

7. Issues & Challenges

Issue	Resolution
Outbound Change Sets not visible.	Verified that current org is Developer Edition → will use CLI deployment or create sandbox.
Initial deployment errors due to folder structure.	Fixed folder structure (lwc, aura, classes).
Component not showing in App Builder.	Updated returnGrid.js-meta.xml to include isExposed=true and correct <targets>.

---

9. Summary

Phase 6 successfully delivered a functional **Return Console** and **ReturnGrid LWC**, integrated with Salesforce data.
Users can now view and manage return requests on both desktop and mobile interfaces.
Enhancements like filtering, bulk actions, and related lists will be included in the next development cycle.