**Expense Tracker Application - Phase 6: Lightning UI and App Development**

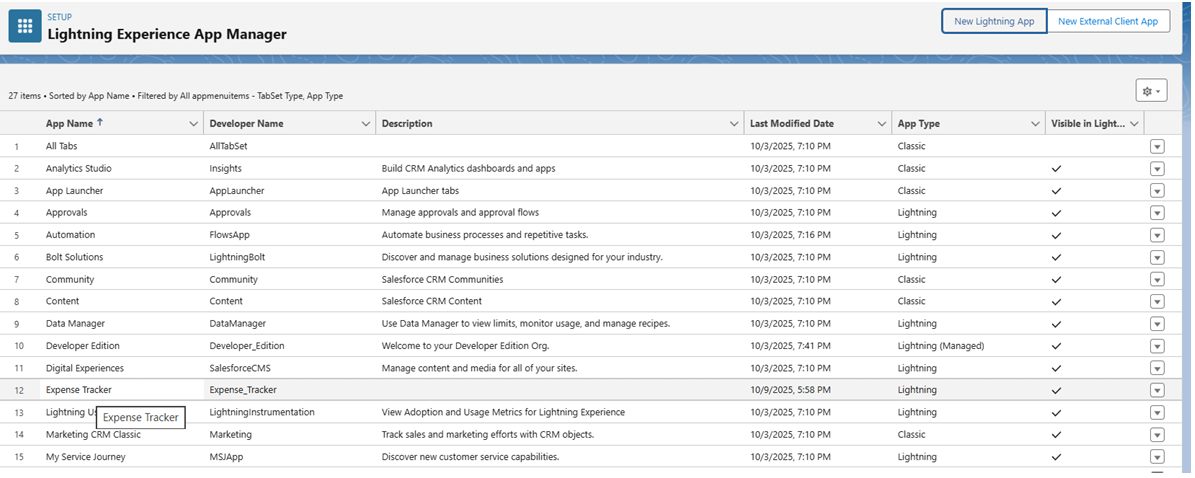
**Objective:** To construct a modern, user-friendly, and cohesive user interface for the Expense Tracker. This phase involves creating a dedicated Lightning App, designing custom record and home pages, and building interactive Lightning Web Components (LWC) for enhanced functionality.

**Step 1: Create the 'Expense Tracker' Lightning App**

* **Purpose:** To group all project-related tabs, pages, and tools into a single, branded application for easy user access.
* **Navigation:** Go to **Setup** → **App Manager** → **New Lightning App**.

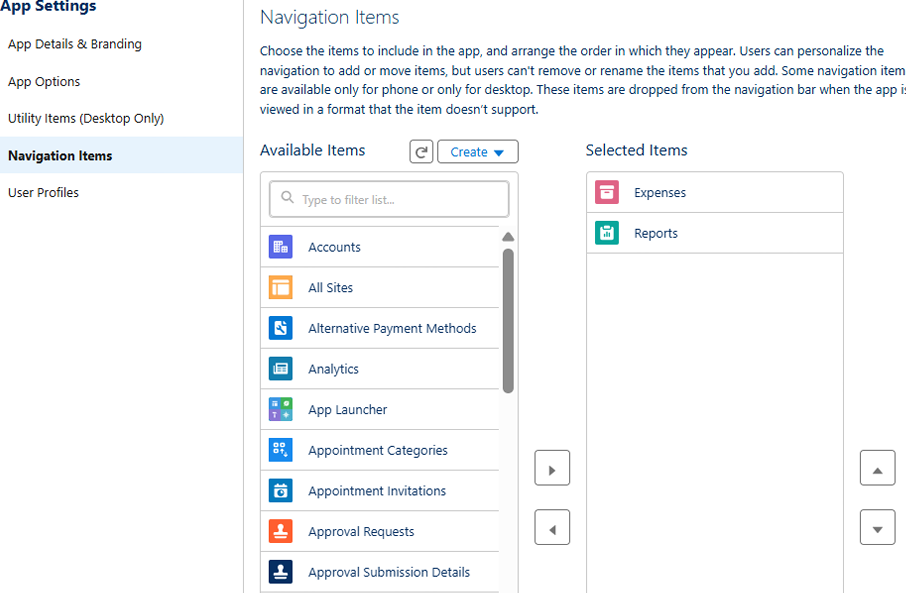
**Configuration:**

1. **App Details:** Enter the **App Name** as Expense Tracker and upload a logo (optional).
2. **Navigation Items:** Add the Expenses tab and the Reports tab to the "Selected Items" list.
3. **Utility Bar (Optional):** Add a utility item, such as the New Expense Quick Action, to allow users to create records from anywhere in the app.
4. **User Profiles:** Assign the app to the relevant user profiles (e.g., Expense Employee, Manager, System Administrator).
5. **Save & Finish**.



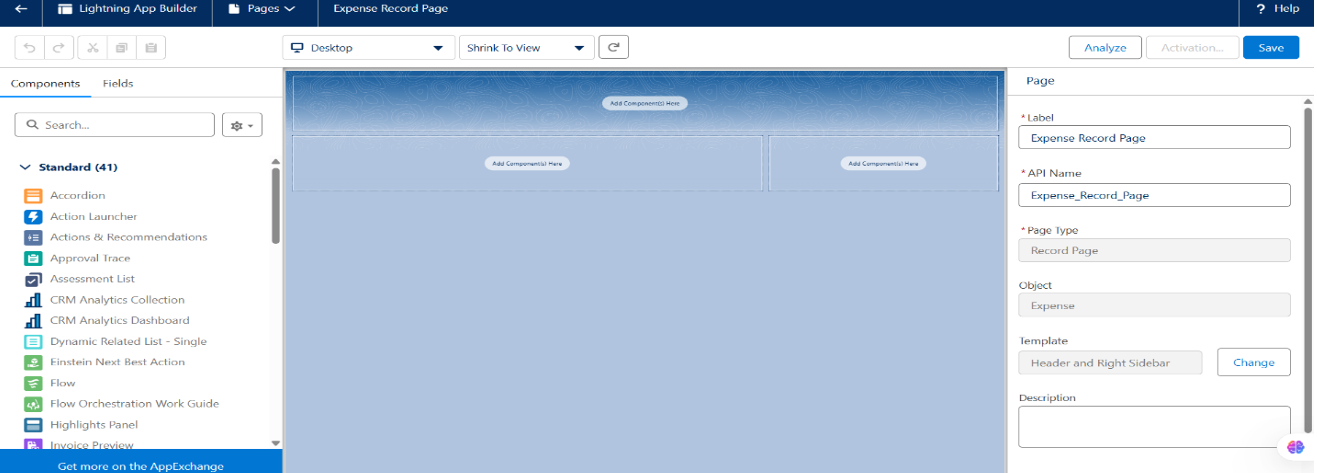
**Step 2: Design the 'Expense' Lightning Record Page**

* **Purpose:** To create an efficient and informative layout for viewing a single expense record, combining key details, related lists, and actions.
* **Navigation:** Go to **Setup** → **Object Manager** → **Expense** → **Lightning Record Pages** → **New** (or Edit existing).



**Layout Configuration:**

1. **Template:** Select a template, such as Header + Right Sidebar.
2. **Main Region:** Drag the **Record Detail** component into the main section. This will display the fields arranged in the page layout, such as:
   * Expense Number
   * Amount
   * Expense Date
   * Category
   * Employee
   * Approval Status
3. **Sidebar:** Drag the **Related Lists** component into the sidebar. Include lists like Approval History and Tasks (if applicable).
4. **Highlights Panel:** Configure the header to show key actions like Submit for Approval and Edit.
5. **Activation:** Click **Activate** and assign the page as the **Org Default** for the Expense object.

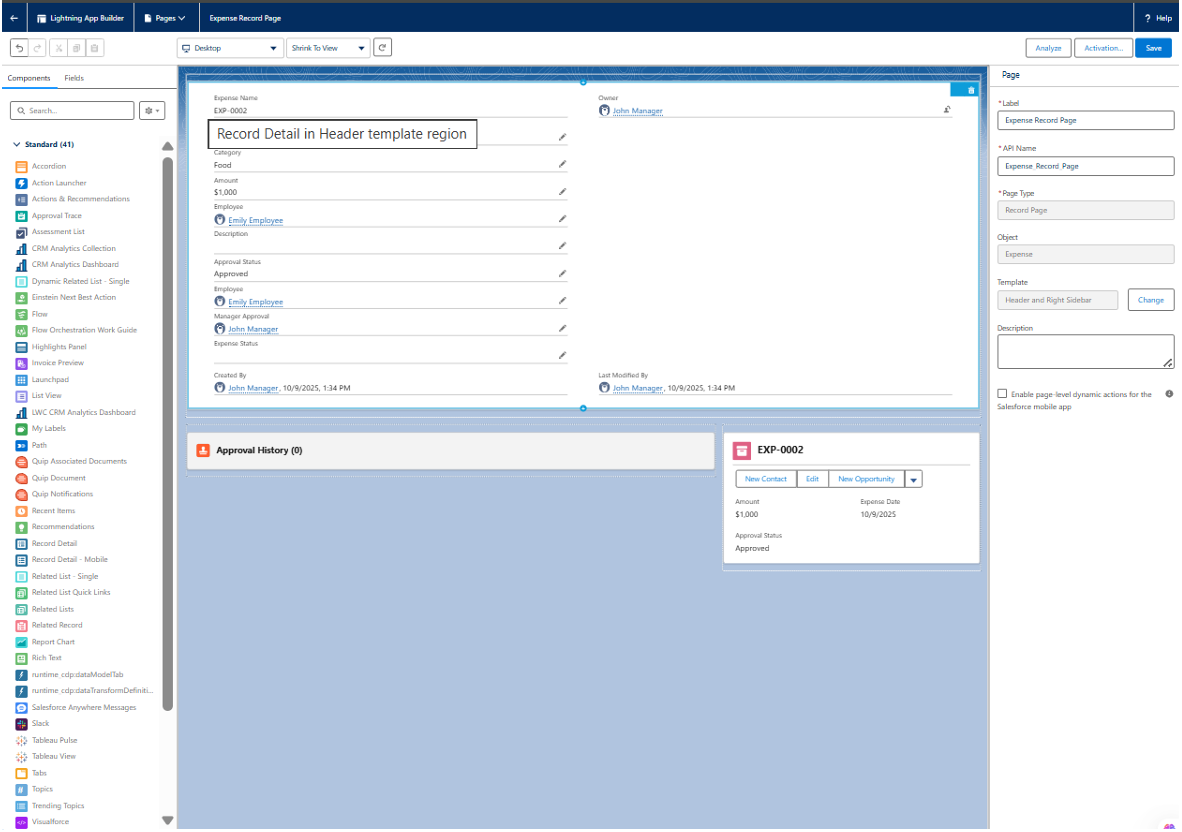


**Step 3: Customize the App Home Page**

* **Purpose:** To provide users with a high-level dashboard summarizing key expense data immediately upon opening the app.
* **Navigation:** Go to **Setup** → **Lightning App Builder** → **New** → **Home Page** (or Edit existing).

**Component Configuration:**

1. Add **Dashboard** or **Report Chart** components to visualize data like Pending Approvals or Total Expenses.
2. Add a **List View** component configured to show Recent Expenses.
3. **Save** and **Activate** the page, assigning it as the default Home Page for the Expense Tracker app and its user profiles.

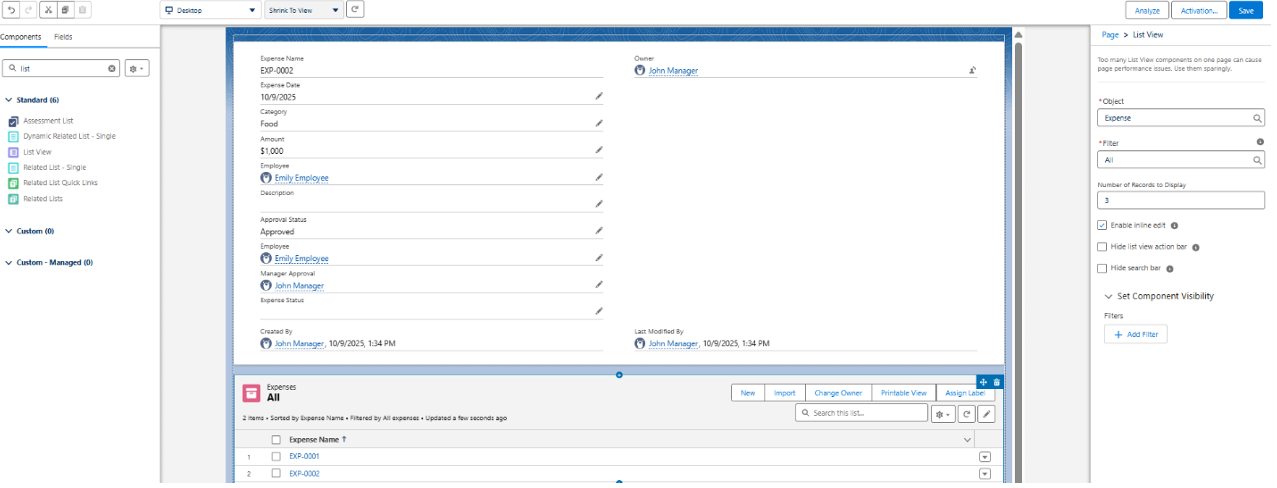


**Step 4: Develop Custom Lightning Web Components (LWC)**

* **Purpose:** To build interactive components that provide functionality beyond standard Salesforce elements, such as custom data tables or one-click approval buttons.
* **Tools:** Use **Visual Studio Code** with the Salesforce Extensions.

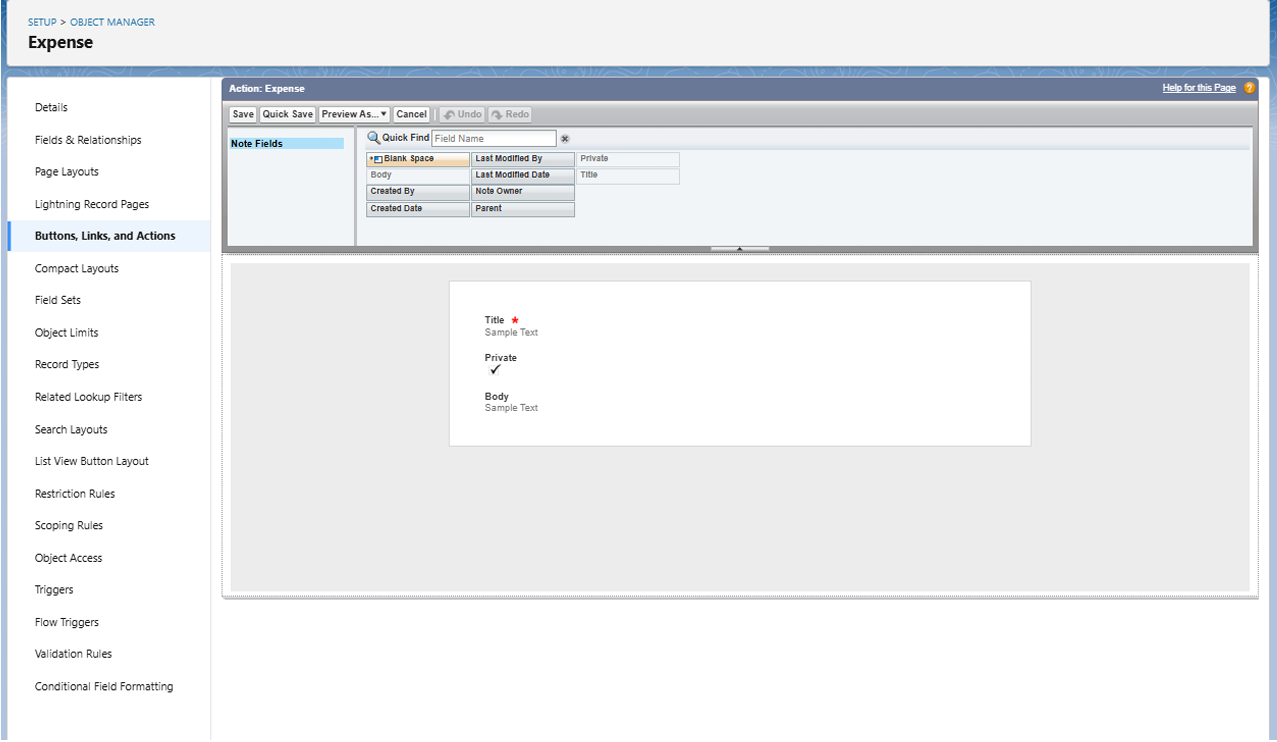
**Part A: Create the Apex Controller**

1. Create an Apex class (e.g., ExpenseController).
2. Add @AuraEnabled methods to make the code callable from an LWC.
   * Create a method to fetch expenses (e.g., getExpenses).
   * Create a method to update an expense (e.g., approveExpense).



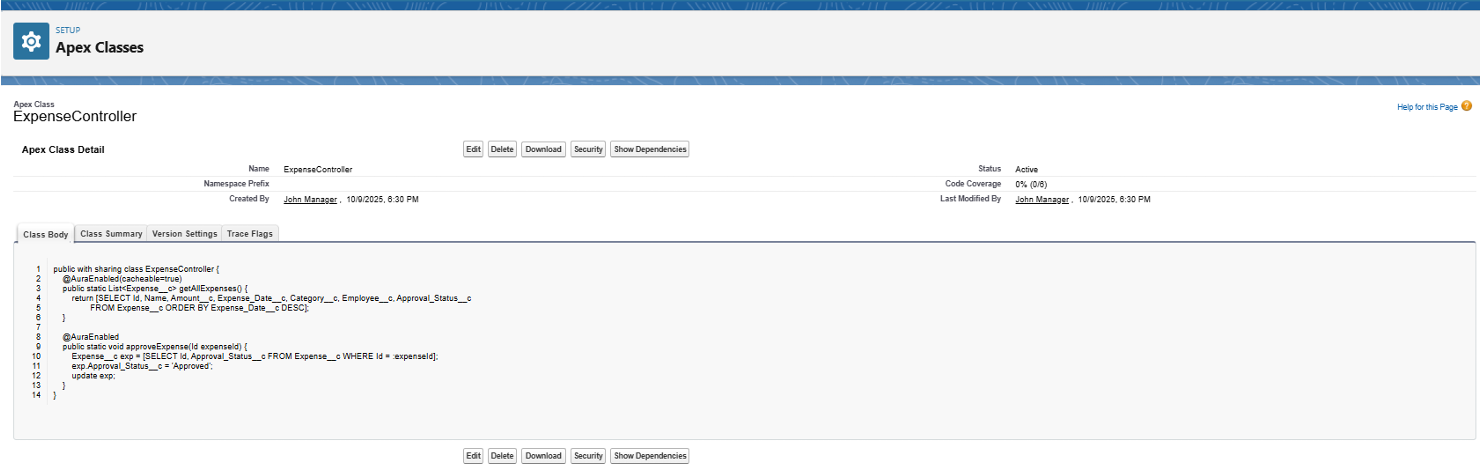
**Part B: Build the LWC (e.g., expenseList)**

1. Create an LWC to display expenses in a table (e.g., expenseList).
2. Use the @wire service to call the getExpenses Apex method and retrieve data dynamically.
3. Display the results in a lightning-datatable.



**Part C: Build an Imperative LWC (e.g., expenseApproveButton)**

1. Create a separate LWC for a custom "Approve" button.
2. When the button is clicked, use an **imperative Apex call** to the approveExpense method, passing the record ID to update the Approval Status.
3. Embed these new LWCs onto the Expense Record Page or Home Page using the Lightning App Builder.

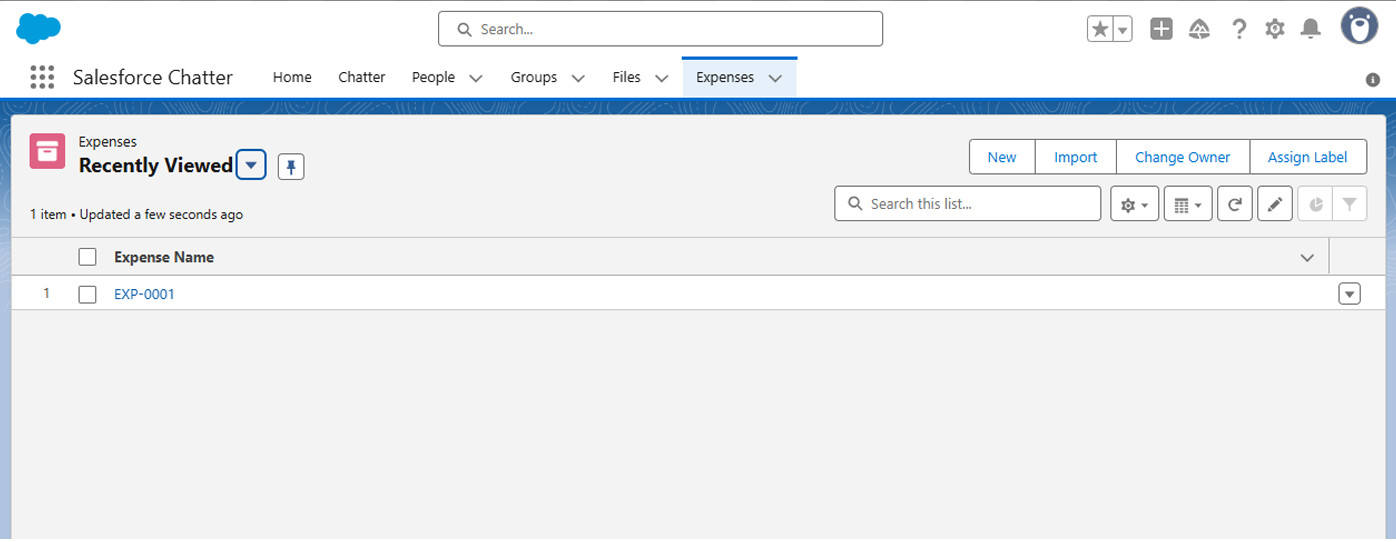


**Step 5: (Optional) Implement Navigation Service**

* **Purpose:** To programmatically redirect users to another page after a custom action is completed, such as navigating back to the home page after an approval.
* **Implementation:** In your LWC JavaScript file, import the NavigationMixin and call the Maps function to redirect the user.

**Step 6: Final User Acceptance Testing**

* **Purpose:** To verify the complete end-to-end user experience from the perspective of different users.
* **Test Cases:**
  1. **Employee Persona:** Log in as an employee. Can you open the Expense Tracker app, create a new expense, and submit it for approval?
  2. **Manager Persona:** Log in as the manager. Can you see the submitted expense? Are the custom approval buttons or actions visible? Can you successfully approve or reject the expense?
  3. **Data-Sync:** Confirm that after an action is taken, the custom LWC (if built) dynamically updates to show the new status. Verify all quick actions and navigation work as expected.



**Phase 6 Complete**

The Expense Tracker application is now fully encapsulated in a user-friendly Lightning App, complete with custom record pages, a dashboard-style home page, and interactive custom components.