

Phase 2: Org Setup & Configuration

Purpose:

The purpose of this phase was to establish a secure and structured Salesforce environment where all insurance claim-related operations could be configured. This setup ensures that data security, access control, and company-wide standards are properly implemented before moving on to data modeling and automation.

(Org Setup & Configuration This phase involves preparing the Salesforce Developer Org for the application build.)

1. Salesforce Edition:

Purpose: The purpose of this phase was to establish a secure and structured Salesforce environment where all insurance claim-related operations could be configured. This setup ensures that data security, access control, and company-wide standards are properly implemented before moving on to data modelling and automation.

Implementation: -

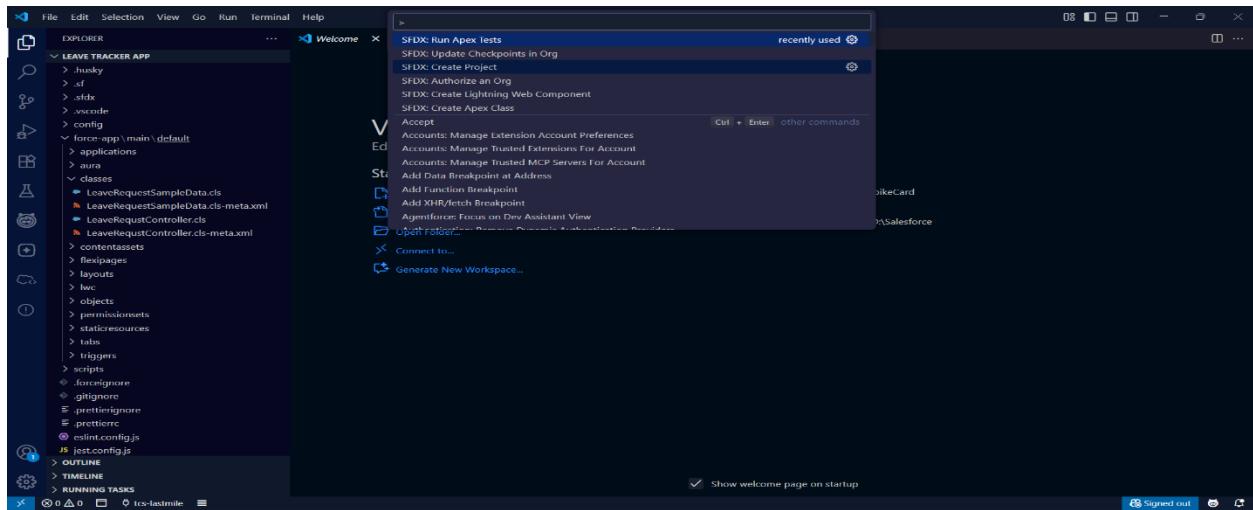
a. Developer Org/Sandbox Creation:

- A Salesforce Developer Org was created for initial configuration and testing.
- Alternatively, a Sandbox can be used to mirror production settings.

To create the org, we are using the lwc (lightning web component/vs code) and also mostly for all the things and implementation like object creation/implementation we are using vs code and then deploy to the org

Step 1:-

- On VScode open command palatte or add (>) symbol and write (SFDX: Create A PROJECT) and name it “Leave Tracker App”

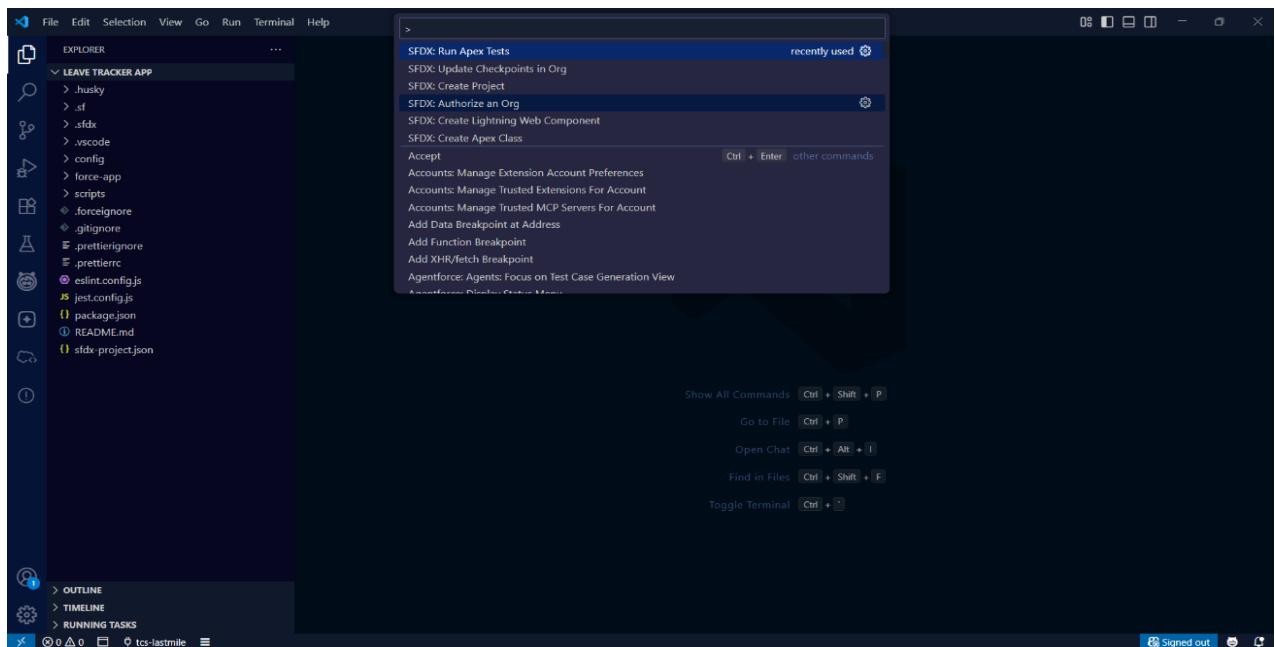


- A new project file will get created and then save it according to the system.

Step 2: -

Authorize the org:

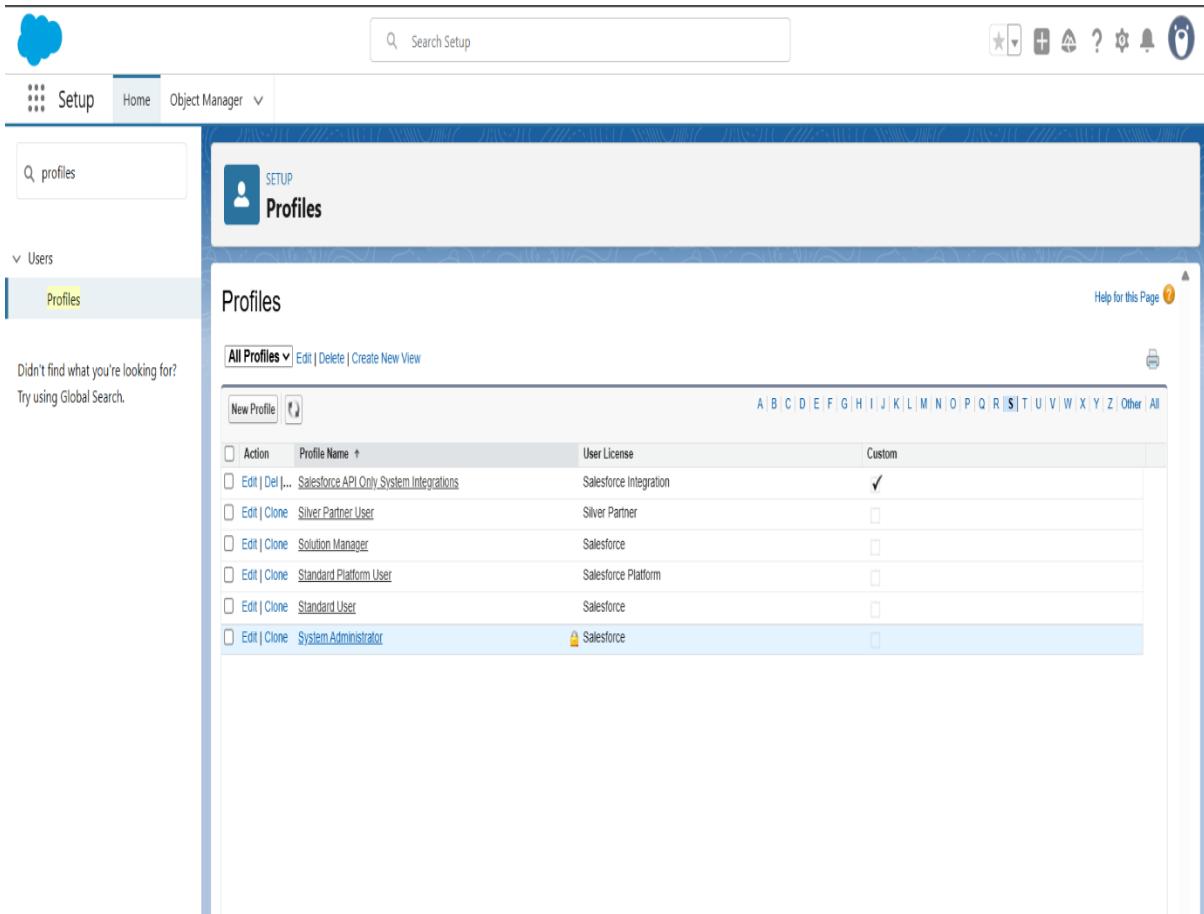
- In this we have to authorize the org to the salesforce account, for that in the same command pallatte search for (SFDX: Authorize an org).
- Press enter, then a new screen will show on web browser on that we have to login using the salesforce credentials.



- After connecting the account to the project, it will show how many logs have been created to the vs code, so choose according to the account you want to create the project on it.
- Now you have created the project titled “Leave Tracker App” and also the “Salesforce Edition” is also comes under the dev org setup.

2. Permission Sets:

- Customized Profiles to restrict or grant access based on responsibilities.
- Added Permission Sets for advanced tasks like claim approvals.



The screenshot shows the Salesforce Setup interface with the following details:

- Header:** Includes a blue cloud icon, a search bar labeled "Search Setup", and various navigation icons.
- Left Sidebar:** Shows a "Setup" icon, "Home", "Object Manager", and a search bar for "profiles". Below it, there's a "Users" section with a "Profiles" tab selected, and a message: "Didn't find what you're looking for? Try using Global Search."
- Central Content:** A title bar with "SETUP" and "Profiles". Below it, a "Help for this Page" link and a printer icon.
- Table View:** A grid titled "Profiles" with columns: "Action", "Profile Name", "User License", and "Custom". The table lists several profiles:

Action	Profile Name	User License	Custom
<input type="checkbox"/> Edit Del ...	Salesforce API Only System Integrations	Salesforce Integration	✓
<input type="checkbox"/> Edit Clone	Silver Partner User	Silver Partner	□
<input type="checkbox"/> Edit Clone	Solution Manager	Salesforce	□
<input type="checkbox"/> Edit Clone	Standard Platform User	Salesforce Platform	□
<input type="checkbox"/> Edit Clone	Standard User	Salesforce	□
<input type="checkbox"/> Edit Clone	System Administrator	Salesforce	□

The screenshot shows the Salesforce Setup interface with the 'Profiles' tab selected. The top navigation bar includes 'Setup', 'Home', 'Object Manager', and a search bar. On the left, a sidebar lists 'Users' and 'Profiles'. A message says 'Didn't find what you're looking for? Try using Global Search.' The main content area is titled 'Profiles' and contains sections for 'Custom Object Permissions', 'Session Settings', and 'Password Policies'. Under 'Custom Object Permissions', there are two tables: one for 'Data Share Target Connection' and another for 'Work Type Groups'. Under 'Session Settings', it shows 'Session Times Out After: 2 hours of inactivity' and 'Session Security Level Required at Login'. Under 'Password Policies', various password requirements are listed, such as 'User passwords expire in 90 days' and 'Minimum password length 8'. At the bottom are 'Edit', 'Clone', and 'View Users' buttons.

The screenshot shows the Salesforce Setup interface with the 'Leave Request Field-Level Security' page for the 'System Administrator' profile. The top navigation bar includes 'Setup', 'Home', 'Object Manager', and a search bar. On the left, a sidebar lists 'Users' and 'Profiles'. A message says 'Didn't find what you're looking for? Try using Global Search.' The main content area is titled 'Leave Request Field-Level Security for profile System Administrator' and includes a 'Help for this Page' link. It displays a table of field-level security settings for the 'System Administrator' profile. The table has columns for 'Field Name', 'Field Type', 'Read Access', and 'Edit Access'. Fields listed include 'Created By', 'From Date', 'Last Modified By', 'Leave Request Id', 'Manager Comment', 'Owner', 'Reason', 'Status', 'To Date', and 'User'. Most fields have 'Read Access' checked, while 'Edit Access' is mostly unchecked. At the bottom are 'Edit' and 'Back to Profile' buttons.

Phase 3: Data Modeling & Relationships

Purpose:

This phase focused on designing the underlying data model to store and manage information about policyholders, policies, and claims. A scalable and logical data structure was essential for enabling claim processing, reporting, and automation.

Implementation:

1. Custom/Standard object creation:

- a. LeaveRequest__c (includes: From_Date__c, To_Date__c, Manager_Comment__c, Reason__c, Status__c and User__c)

The screenshot shows the Salesforce Setup interface, specifically the Object Manager section for the 'Leave Request' object. The left sidebar lists various configuration tabs: Details, Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Restriction Rules, Scoping Rules, Object Access, and Triggers. The 'Details' tab is selected. The main pane displays the 'Leave Request' object's details, including its API Name (LeaveRequest__c), which is marked as 'Custom'. Other details shown include Singular Label (Leave Request) and Plural Label (Leave Requests). On the right side of the main pane, there are buttons for 'Edit' and 'Delete'. The top navigation bar includes links for Setup, Home, and Object Manager, along with a search bar and various system icons.

The screenshot shows the Salesforce Setup interface with the following details:

- Header:** Includes the Salesforce logo, a search bar labeled "Search Setup", and various navigation icons.
- Breadcrumbs:** "SETUP > OBJECT MANAGER".
- Page Title:** "Leave Request".
- Left Sidebar:** A navigation menu with the following items:
 - Details
 - Fields & Relationships** (selected)
 - Page Layouts
 - Lightning Record Pages
 - Buttons, Links, and Actions
 - Compact Layouts
 - Field Sets
 - Object Limits
 - Record Types
 - Related Lookup Filters
 - Restriction Rules
 - Scoping Rules
 - Object Access
 - Triggers
- Table:** "Fields & Relationships" section, displaying 10 items sorted by Field Label.

Field	Type	Description	Actions
Created By	CreatedBy	Lookup(User)	▼
From Date	From_Date__c	Date	▼
Last Modified By	LastModifiedBy	Lookup(User)	▼
Leave Request Id	Name	Auto Number	✓
Manager Comment	Manager_Comment__c	Text Area(255)	▼
Owner	OwnerId	Lookup(User,Group)	✓
Reason	Reason__c	Text Area(255)	▼
Status	Status__c	Picklist	▼
To Date	To_Date__c	Date	▼
User	User__c	Lookup(User)	✓

Phase 4: Process Automation (Admin) (Only client-side validation)

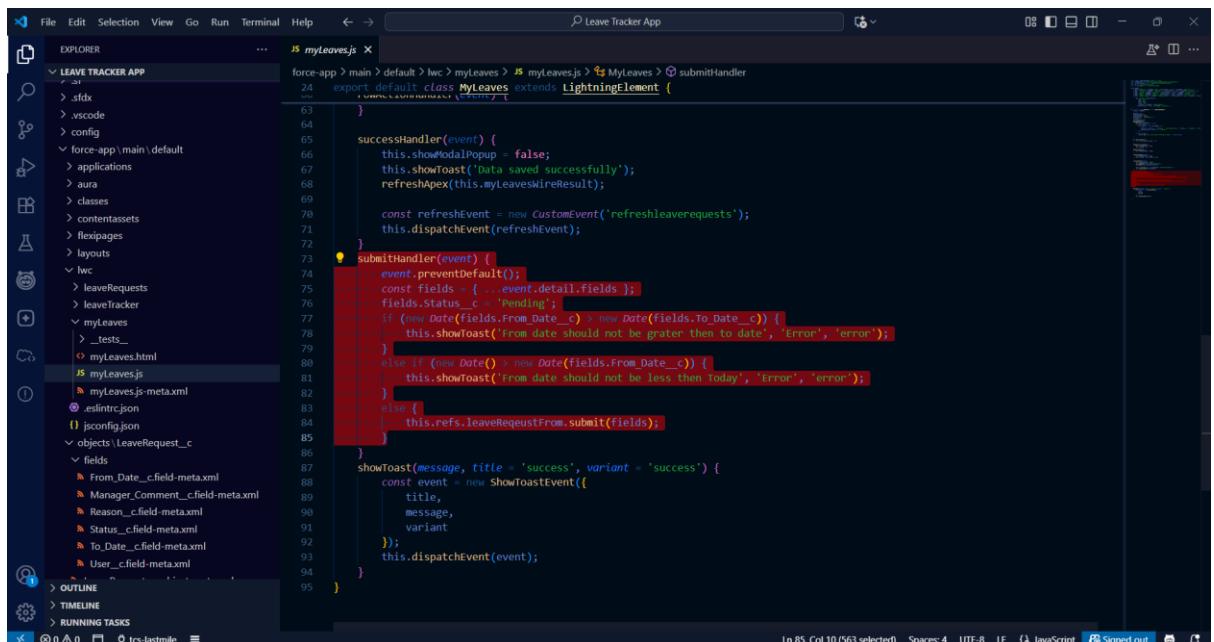
Purpose:

The goal of this phase was to automate business processes to improve efficiency, ensure compliance with rules, and streamline claim submission and approval workflows.

Implementation: -

1. Custom validation using form submit event:

We are using lwc and Vs code to create the validation in the submit handler.

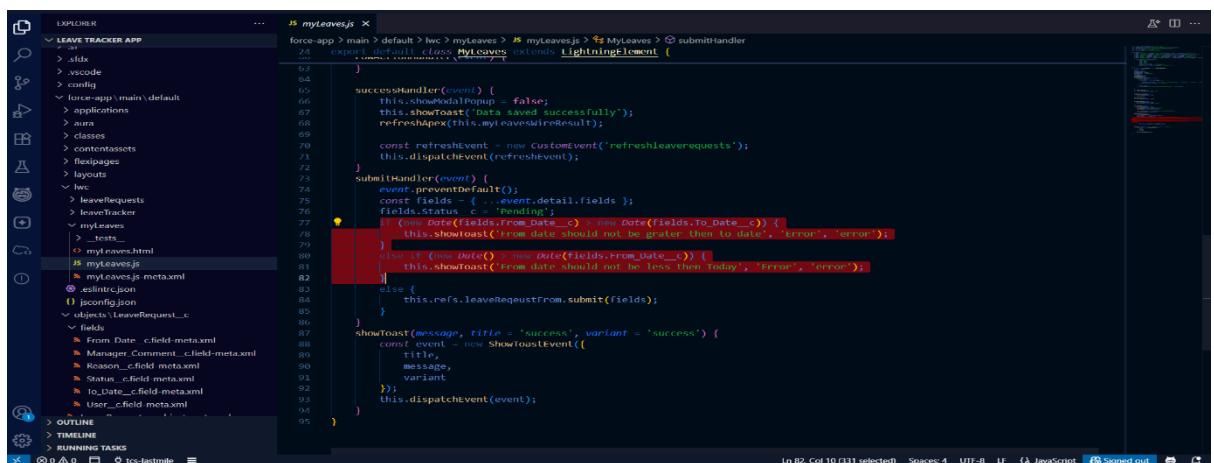


```
force-app > main > default > lwc > myLeaves > JS myLeaves.js > submitHandler

export default class MyLeaves extends LightningElement {
    ...
    successHandler(event) {
        ...
        const fields = { ...event.detail.fields };
        fields.status__c = 'Pending';
        if (new Date(fields.From_Date__c) > new Date(fields.To_Date__c)) {
            this.showToast('From date should not be greater than to date', 'Error', 'error');
        } else if (new Date() > new Date(fields.From_Date__c)) {
            this.showToast('From date should not be less than today', 'Error', 'error');
        } else {
            this.refs.leaveRequestFrom.submit(fields);
        }
        ...
        showToast(message, title = 'success', variant = 'success') {
            const event = new ShowToastEvent({
                title,
                message,
                variant
            });
            this.dispatchEvent(event);
        }
    }
}
```

2. Date validation logic:

This is for the logic for the from date to date for both MyLeaves and Request.



```
force-app > main > default > lwc > myLeaves > JS myLeaves.js > submitHandler

export default class MyLeaves extends LightningElement {
    ...
    successHandler(event) {
        ...
        const fields = { ...event.detail.fields };
        fields.status__c = 'Pending';
        if (new Date(fields.From_Date__c) > new Date(fields.To_Date__c)) {
            this.showToast('From date should not be greater than to date', 'Error', 'error');
        } else if (new Date() > new Date(fields.From_Date__c)) {
            this.showToast('From date should not be less than today', 'Error', 'error');
        } else {
            this.refs.leaveRequestFrom.submit(fields);
        }
        ...
        showToast(message, title = 'success', variant = 'success') {
            const event = new ShowToastEvent({
                title,
                message,
                variant
            });
            this.dispatchEvent(event);
        }
    }
}
```

LEAVE TRACKER APP

myLeaves.js

```
force-app > main > default > lwc > myLeaves > JS MyLeaves > submitHandler
24 export default class MyLeaves extends LightningElement {
25
26     popupCloseHandler() {
27         this.showModalPopup = false;
28     }
29
30     rowActionHandler(event) {
31         this.showModalPopup = true;
32         this.recordId = event.detail.row.Id;
33     }
34
35     successHandler(event) {
36         this.showModalPopup = false;
37         this.showToast('Data saved successfully');
38         refreshApex(this.myLeavesWireResult);
39
40         const refreshEvent = new CustomEvent('refreshleaverequests');
41         this.dispatchEvent(refreshEvent);
42     }
43
44     submitHandler(event) {
45         event.preventDefault();
46         const fields = { ...event.detail.fields };
47         fields.Status__c = 'Pending';
48         if (new Date(fields.From_Date__c) > new Date(fields.To_Date__c)) {
49             this.showToast('From date should not be grater then to date', 'Error', 'error');
50         }
51         else if (new Date() > new Date(fields.From_Date__c)) {
52             this.showToast('From date should not be less then Today', 'Error', 'error');
53         }
54         else {
55             this.refs.leaveRequestForm.submit(fields);
56         }
57         showToast(message, title = 'success', variant = 'success') {
58             const event = new ShowToastEvent({
59                 title,
60             });
61             this.dispatchEvent(event);
62         }
63     }
64 }
```

Ln 74, Col 32 (335 selected) Spaces: 4 UTF-8 LF ↻ JavaScript Signed out

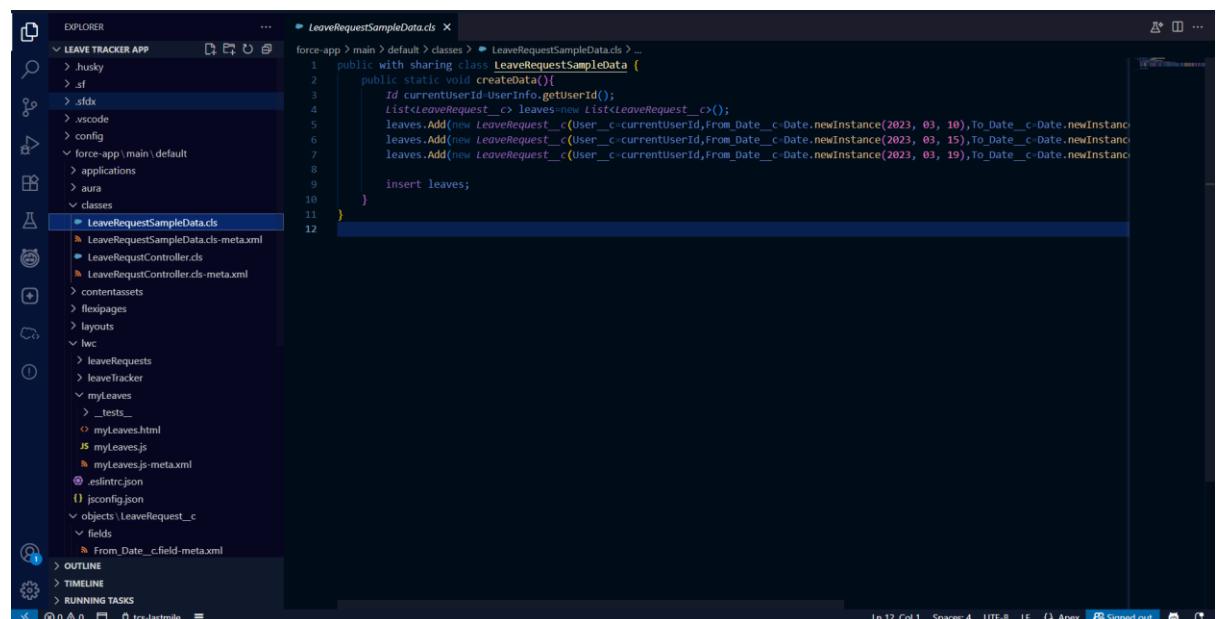
Phase 5: Apex Programming (Developer)

Purpose: Custom logic was required to extend Salesforce's standard functionality. Apex programming allowed automation of tasks not possible with declarative tools.

Implementation:

1. Classes & Object (Apex classes setup):

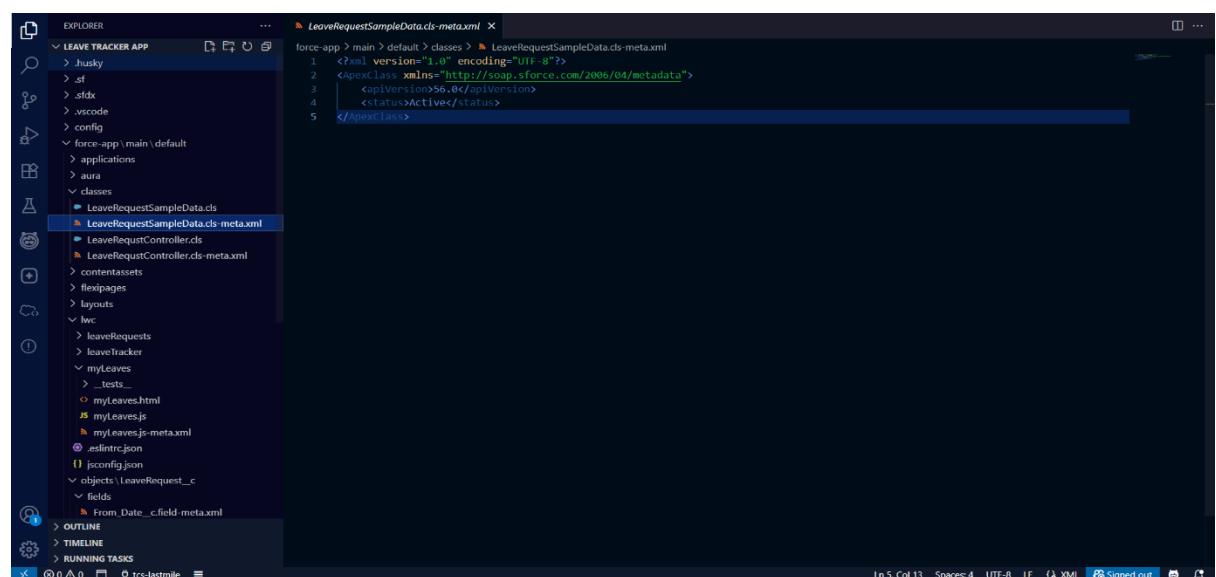
- LeaveRequestSampleData
- LeaveRequestController



The screenshot shows the VS Code interface with the Explorer sidebar open, displaying the project structure for 'LEAVE TRACKER APP'. The 'LeaveRequestSampleData.cls' file is selected in the list. The main editor tab shows the Apex code for 'LeaveRequestSampleData.cls'. The code creates a new list of 'LeaveRequest__c' objects and adds three instances to it, setting their 'From__Date' and 'To__Date' fields to specific dates (2023-03-10, 2023-03-15, and 2023-03-19) and their 'User__c' field to the current user's ID.

```
public with sharing class LeaveRequestSampleData {
    public static void createData() {
        Id currentUserInfo = UserInfo.getUserId();
        List<LeaveRequest__c> leaves = new List<LeaveRequest__c>();
        leaves.Add(new LeaveRequest__c(User__c=currentUser, From__Date__c=Date.newInstance(2023, 03, 10), To__Date__c=Date.newInstance(2023, 03, 15)));
        leaves.Add(new LeaveRequest__c(User__c=currentUser, From__Date__c=Date.newInstance(2023, 03, 15), To__Date__c=Date.newInstance(2023, 03, 19)));
        leaves.Add(new LeaveRequest__c(User__c=currentUser, From__Date__c=Date.newInstance(2023, 03, 19), To__Date__c=Date.newInstance(2023, 03, 20)));
        insert leaves;
    }
}
```

LeaveRequestSampleData.cls



The screenshot shows the VS Code interface with the Explorer sidebar open, displaying the project structure for 'LEAVE TRACKER APP'. The 'LeaveRequestSampleData.cls-meta.xml' file is selected in the list. The main editor tab shows the XML metadata for the Apex class. It defines the class name as 'LeaveRequestSampleData', specifies the API version as 56.0, and sets the status to 'Active'.

```
<?xml version='1.0' encoding='UTF-8'?>
<apexClass xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>56.0</apiVersion>
    <status>Active</status>
</apexClass>
```

LeaveRequestSampleData.cls-meta.xml

The screenshot shows the Visual Studio Code interface with the following details:

- EXPLORER**: Shows the project structure for "LEAVE TRACKER APP".
- EDITOR**: Displays the code for `LeaveRequestController.cls`. The code implements two static methods: `getMyLeaves()` and `getLeaveRequests()`, both using `@AuraEnabled(cacheable=true)`.
- STATUS BAR**: Shows "Ln 24, Col 2" and "Signed out".

```
• LeaveRequestController.cls X
force-app > main > default > classes > LeaveRequestController.cls > LeaveRequestController
1 public with sharing class LeaveRequestController {
2     @AuraEnabled(cacheable=true)
3     public static List<LeaveRequest__c> getMyLeaves(){
4         try {
5             List<LeaveRequest__c> myLeaves=new List<LeaveRequest__c>();
6             myLeaves=[SELECT Id,Name,From_Date__c,To_Date__c,Reason__c,Status__c,Manager_Comment__c FROM LeaveRequest__c WHERE User__r.ManagerId=:UserInfo.getUserId() ORDER BY CreatedDate DESC];
7             return myLeaves;
8         } catch (Exception e) {
9             throw new AuraHandledException(e.getMessage());
10        }
11    }
12
13    @AuraEnabled(cacheable=true)
14    public static List<LeaveRequest__c> getLeaveRequests(){
15        try {
16            List<LeaveRequest__c> myLeaves=new List<LeaveRequest__c>();
17            myLeaves=[SELECT Id,Name,From_Date__c,To_Date__c,Reason__c,Status__c,Manager_Comment__c,User__r.ManagerId,User__r.Name
18                      WHERE User__r.ManagerId=:UserInfo.getUserId() ORDER BY CreatedDate DESC];
19            return myLeaves;
20        } catch (Exception e) {
21            throw new AuraHandledException(e.getMessage());
22        }
23    }
24 }
```

LeaveRequestController.cls

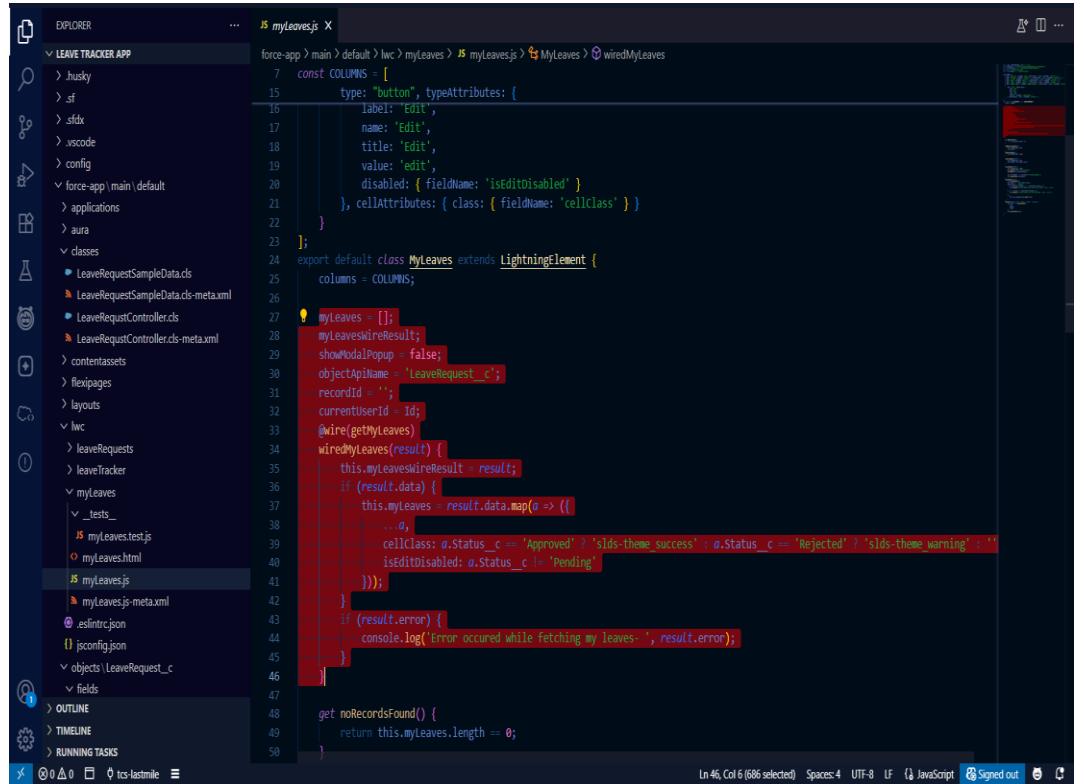
The screenshot shows the Visual Studio Code interface with the following details:

- EXPLORER**: Shows the project structure for "LEAVE TRACKER APP".
- EDITOR**: Displays the XML code for `LeaveRequestController.cls-meta.xml`. It defines an Apex class with an API version of 56.0.
- STATUS BAR**: Shows "Ln 1, Col 1" and "Signed out".

```
<?xml version="1.0" encoding="UTF-8"?>
<ApexClass xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>56.0</apiVersion>
    <status>Active</status>
</ApexClass>
```

LeaveRequestController.cls-meta.xml

- Now to connect to the objects/classes the (@wire) is used to connect



```

    const COLUMNS = [
      { type: "button", typeAttributes: {
        label: "Edit",
        name: "Edit",
        title: "Edit",
        value: "edit",
        disabled: { fieldName: 'isEditDisabled' }
      }, cellAttributes: { class: { fieldName: 'cellClass' } } }
    ];
    export default class MyLeaves extends LightningElement {
      columns = COLUMNS;
      myLeaves = [];
      myLeavesWireResult;
      showModalPopup = false;
      objectApiName = 'LeaveRequest__c';
      recordId = '';
      currentUserID = Id;
      @wire(getMyLeaves)
      wiredMyLeaves(result) {
        if (result.data) {
          this.myLeavesWireResult = result;
          this.myLeaves = result.data.map(o => ({
            ...o,
            cellClass: o.Status__c === 'Approved' ? 'slds-theme_success' : o.Status__c === 'Rejected' ? 'slds-theme_warning' : '',
            isEditDisabled: o.Status__c === 'Pending'
          }));
        }
        if (result.error) {
          console.log('Error occurred while fetching my leaves...', result.error);
        }
      }
      get noRecordsFound() {
        return this.myLeaves.length === 0;
      }
    }
  
```

The screenshot shows the VS Code interface with the 'myLeaves.js' file open in the editor. The code defines a class 'MyLeaves' extending 'LightningElement'. It includes a static array 'COLUMNS' for column definitions, a state variable 'myLeaves' to store data, and a state variable 'myLeavesWireResult' to store the result of the wire call. The 'wiredMyLeaves' method processes the wire result by mapping each object to a new one with a 'cellClass' based on its status and an 'isEditDisabled' flag. If there's an error, it logs the error message. The 'noRecordsFound' getter returns true if the 'myLeaves' array has zero length.

(note: the LeaveRequestSampleData is a dummy data for testing the app)

Phase 6: User Interface Development

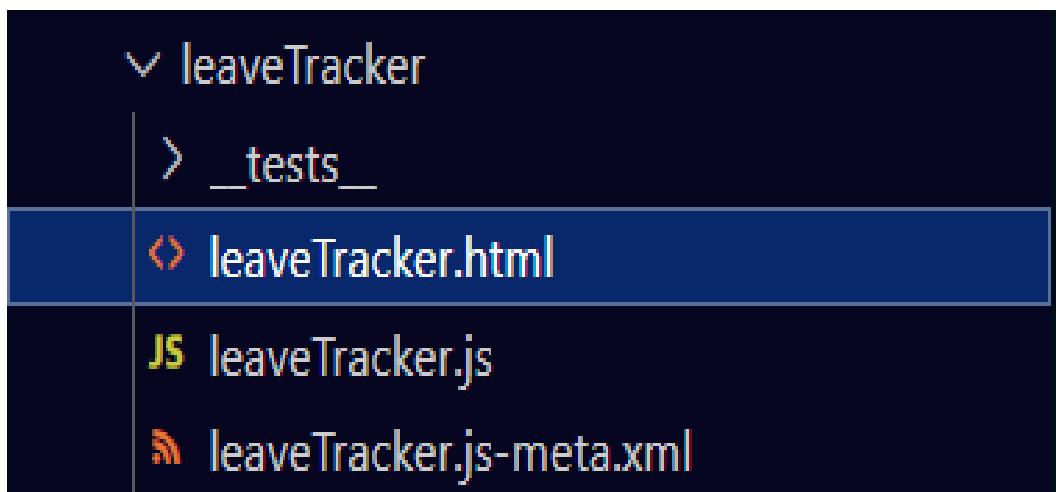
Purpose:

The purpose of this phase was to design an interactive and user-friendly interface for agents, managers, and policyholders using Salesforce Lightning Web Components (LWC). This ensures that claim submissions, claim tracking, and managerial dashboards are intuitive, efficient, and aligned with Salesforce Lightning Design System (SLDS) standards.

Implementation:

1. Parent component using LWC:

- Using LWC (lightning web component) the parent component is created the project
- The parent component is ‘leave Tracker’.
- SFDX: Create Lightning Web Component from the command palette and give the name “leaveTracker”.



The screenshot shows the VS Code interface with the following details:

- Explorer View:** Shows the project structure under "LEAVE TRACKER APP".
 - force-app/main/default:
 - classes
 - LeaveRequestSampleData.cls
 - LeaveRequestSampleData.cls-meta.xml
 - LeaveRequestController.cls
 - LeaveRequestController.cls-meta.xml
 - contentassets
 - flexpages
 - layouts
 - IWC
 - leaveRequests
 - _tests_
 - leaveRequests.html
 - leaveRequests.js
 - leaveRequests.js-meta.xml
 - leaveTracker
 - _tests_
 - leaveTracker.html
 - leaveTracker.js
 - leaveTracker.js-meta.xml
 - myLeaves
 - _tests_
 - myLeaves.test.js
 - myLeaves.html
 - myLeaves.js
 - myLeaves.js-meta.xml
 - .eslintrc.json
 - .jsonconfig.json
 - Editor View:** The file "leaveTracker.html" is open in the editor.

```
<template>
  <lightning-card>
    <lightning-tabc>
      <lightning-tab title="My Leaves" label="My Leaves">
        <>my-leaves onrefreshleaverequests=[refreshLeaveRequestHandler]</>c-my-leaves>
      </lightning-tab>
      <lightning-tab title="Leave Requests" label="Leave Requests">
        <>leave-requests IWC:ref="myLeavesComp"</>c-leave-requests>
      </lightning-tab>
    </lightning-tabc>
  </lightning-card>
</template>
```
 - Bottom Status Bar:** Shows "In 1, Col 1 Spaces:4 UTF-8 LF {} HTML" and "Signed out".

leaveTracker.html

The screenshot shows the VS Code interface with the following details:

- Explorer View:** Shows the project structure under "LEAVE TRACKER APP".
 - force-app/main/default:
 - classes
 - LeaveRequestSampleData.cls
 - LeaveRequestSampleData.cls-meta.xml
 - LeaveRequestController.cls
 - LeaveRequestController.cls-meta.xml
 - contentassets
 - flexpages
 - IWC
 - layouts
 - leaveRequests
 - _tests_
 - leaveRequests.html
 - leaveRequests.js
 - leaveRequests.js-meta.xml
 - leaveTracker
 - _tests_
 - leaveTracker.html
 - leaveTracker.js
 - leaveTracker.js-meta.xml
 - myLeaves
 - _tests_
 - myLeaves.test.js
 - myLeaves.html
 - myLeaves.js
 - myLeaves.js-meta.xml
 - .eslintrc.json
 - .jsonconfig.json
- Editor View:** The file "leaveTracker.js" is open in the editor.

```
import { api, LightningElement, wire } from 'lwc';
export default class LeaveTracker extends LightningElement {
  refreshLeaveRequestHandler(event) {
    this.refs.myLeavesComp.refreshGrid();
  }
}
```

leaveTracker.js

```

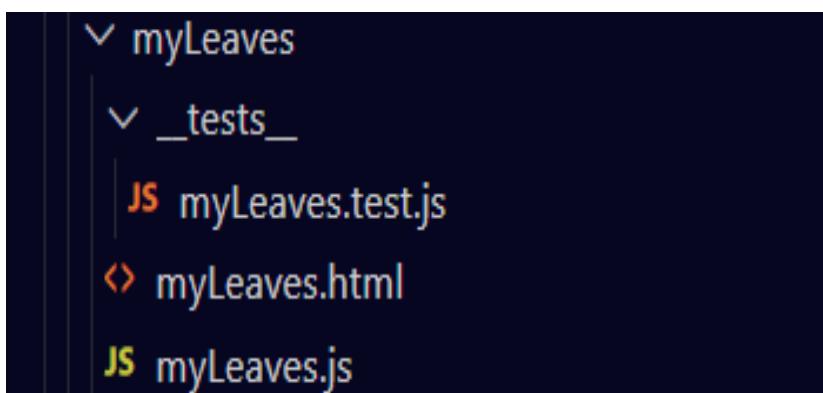
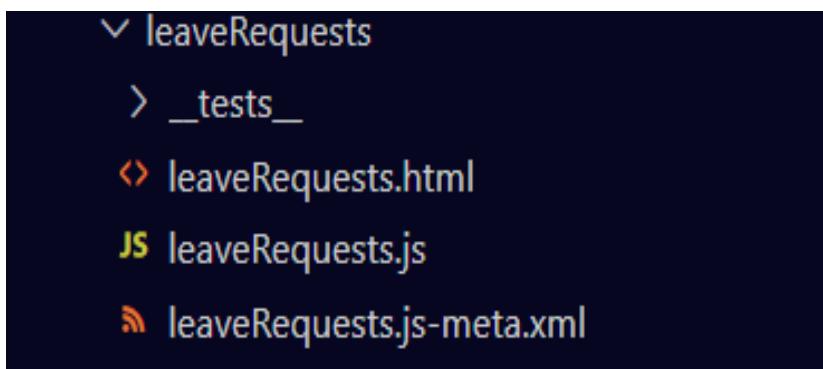
<?xml version="1.0" encoding="UTF-8"?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>56.0</apiVersion>
    <isExposed>true</isExposed>
    <targets>
        |   <target>lightning__AppPage</target>
    </targets>
</LightningComponentBundle>

```

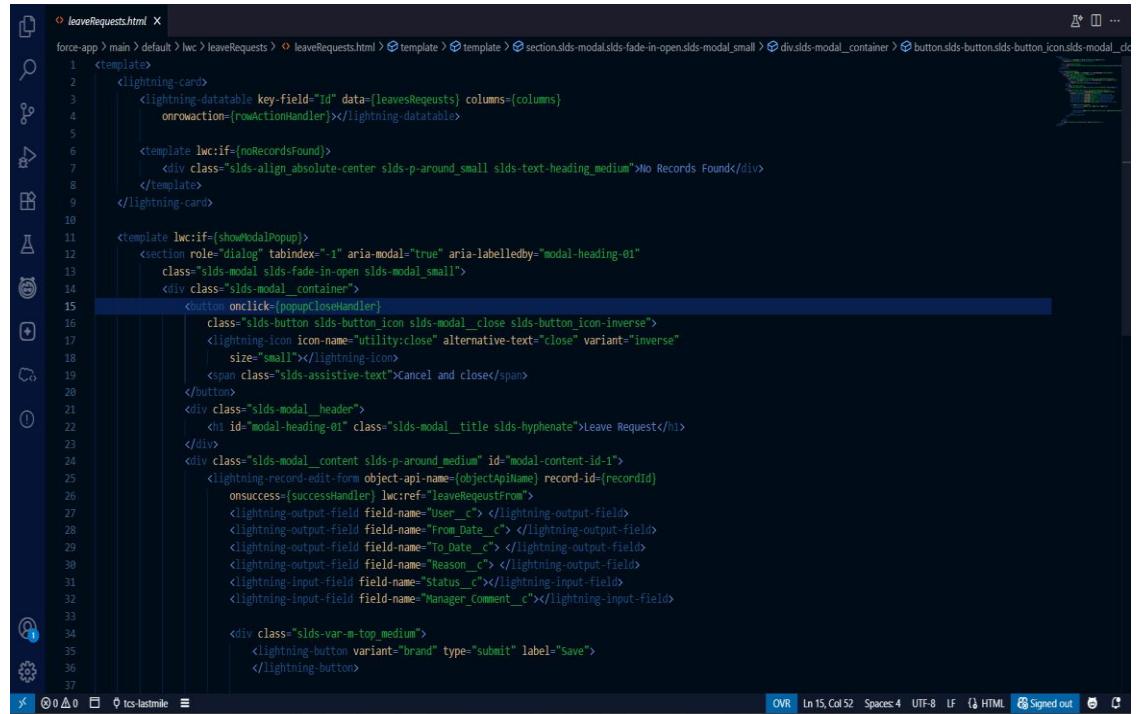
leaveTracker.js-meta.xml

2. Child component using LWC:

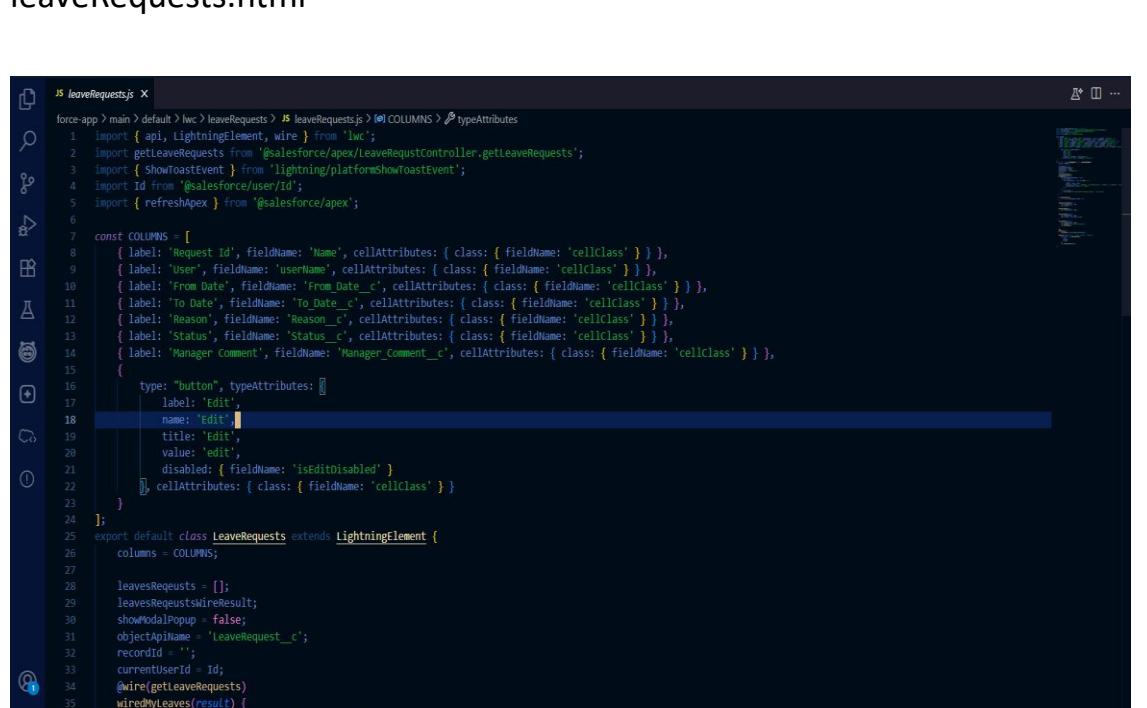
- leaveRequests
- myLeaves



- The Screen Shots for leaveRequests including its code:



```
force-app > main > default > lwc > leaveRequests > leaveRequests.html
1  <template>
2    <lightning-card>
3      <lightning-data-table key-field="id" data=[leavesRequests] columns={columns}
4        onrowaction=[rowActionHandler]></lightning-data-table>
5
6      <template lwc:if={noRecordsFound}>
7        <div class="slds-align_absolute-center slds-p-around_small slds-text-heading_medium">No Records Found</div>
8      </template>
9    </lightning-card>
10
11   <template lwc:if={showModalPopup}>
12     <section role="dialog" tabindex="-1" aria-modal="true" aria-labelledby="modal-heading-01"
13       class="slds-modal slds-fade-in-open slds-modal_small">
14       <div class="slds-modal__container">
15         <button onclick={popupCloseHandler}>
16           class="slds-button slds-button_icon slds-modal__close slds-button__icon-inverse">
17             <lightning-icon icon-name="utility:close" alternative-text="close" variant="inverse"
18               size="small"></lightning-icon>
19             <span class="slds-assistive-text">Cancel and close</span>
20         </button>
21         <div class="slds-modal__header">
22           <h1 id="modal-heading-01" class="slds-modal__title slds-hyphenate">Leave Request</h1>
23         </div>
24         <div class="slds-modal__content slds-p-around_medium" id="modal-content-id-1">
25           <lightning-record-edit-form object-api-name={objectApiName} record-id={recordId}
26             onsuccess={successHandler} lwc:ref="leaveRequestForm">
27             <lightning-output-field field-name="User_c"></lightning-output-field>
28             <lightning-output-field field-name="From Date_c"></lightning-output-field>
29             <lightning-output-field field-name="To Date_c"></lightning-output-field>
30             <lightning-output-field field-name="Reason_c"></lightning-output-field>
31             <lightning-input-field field-name="Status_c"></lightning-input-field>
32             <lightning-input-field field-name="Manager Comment_c"></lightning-input-field>
33
34             <div class="slds-var-m-top_medium">
35               <lightning-button variant="brand" type="submit" label="Save"></lightning-button>
36           </div>
37         </lightning-record-edit-form>
38       </div>
39     </div>
40   </template>
41
42   <script>
43     import { LightningElement, wire } from 'lwc';
44     import getLeaveRequests from '@salesforce/apex/LeaveRequestController.getLeaveRequests';
45     import { ShowToastEvent } from 'lightning/platformShowToastEvent';
46     import Id from '@salesforce/user/Id';
47     import { refreshApex } from '@salesforce/apex';
48
49     const COLUMNS = [
50       { label: 'Request Id', fieldName: 'Name', cellAttributes: { class: { fieldName: 'cellClass' } } },
51       { label: 'User', fieldName: 'userName', cellAttributes: { class: { fieldName: 'cellClass' } } },
52       { label: 'From Date', fieldName: 'From Date_c', cellAttributes: { class: { fieldName: 'cellClass' } } },
53       { label: 'To Date', fieldName: 'To Date_c', cellAttributes: { class: { fieldName: 'cellClass' } } },
54       { label: 'Reason', fieldName: 'Reason_c', cellAttributes: { class: { fieldName: 'cellClass' } } },
55       { label: 'Status', fieldName: 'Status_c', cellAttributes: { class: { fieldName: 'cellClass' } } },
56       { label: 'Manager Comment', fieldName: 'Manager Comment_c', cellAttributes: { class: { fieldName: 'cellClass' } } },
57     ];
58
59     export default class LeaveRequests extends LightningElement {
60       columns = COLUMNS;
61
62       leavesRequests = [];
63       leavesRequestsWireResult;
64       showModalPopup = false;
65       objectApiName = 'LeaveRequest_c';
66       recordId = '';
67       currentUserId = Id;
68       @wire(getLeaveRequests)
69       wiredLeaves(result) {
70         this.leavesRequestsWireResult = result;
71         if (result.data) {
72           this.leavesRequests = result.data;
73         }
74       }
75     }
76   </script>
77
78   <style>
79     .slds-modal__close {
80       color: #0070C0 !important;
81     }
82   </style>
83
84   <script>
85     window.addEventListener('load', () => {
86       const modal = document.querySelector('.slds-modal');
87       modal.style.display = 'block';
88     });
89   </script>
90
91   <style>
92     .slds-modal__close {
93       color: #0070C0 !important;
94     }
95   </style>
96
97   <script>
98     window.addEventListener('load', () => {
99       const modal = document.querySelector('.slds-modal');
100      modal.style.display = 'block';
101    });
102  </script>
103
104  <style>
105    .slds-modal__close {
106      color: #0070C0 !important;
107    }
108  </style>
109
110  <script>
111    window.addEventListener('load', () => {
112      const modal = document.querySelector('.slds-modal');
113      modal.style.display = 'block';
114    });
115  </script>
116
117  <style>
118    .slds-modal__close {
119      color: #0070C0 !important;
120    }
121  </style>
122
123  <script>
124    window.addEventListener('load', () => {
125      const modal = document.querySelector('.slds-modal');
126      modal.style.display = 'block';
127    });
128  </script>
129
130  <style>
131    .slds-modal__close {
132      color: #0070C0 !important;
133    }
134  </style>
135
136  <script>
137    window.addEventListener('load', () => {
138      const modal = document.querySelector('.slds-modal');
139      modal.style.display = 'block';
140    });
141  </script>
142
143  <style>
144    .slds-modal__close {
145      color: #0070C0 !important;
146    }
147  </style>
148
149  <script>
150    window.addEventListener('load', () => {
151      const modal = document.querySelector('.slds-modal');
152      modal.style.display = 'block';
153    });
154  </script>
155
156  <style>
157    .slds-modal__close {
158      color: #0070C0 !important;
159    }
160  </style>
161
162  <script>
163    window.addEventListener('load', () => {
164      const modal = document.querySelector('.slds-modal');
165      modal.style.display = 'block';
166    });
167  </script>
168
169  <style>
170    .slds-modal__close {
171      color: #0070C0 !important;
172    }
173  </style>
174
175  <script>
176    window.addEventListener('load', () => {
177      const modal = document.querySelector('.slds-modal');
178      modal.style.display = 'block';
179    });
180  </script>
181
182  <style>
183    .slds-modal__close {
184      color: #0070C0 !important;
185    }
186  </style>
187
188  <script>
189    window.addEventListener('load', () => {
190      const modal = document.querySelector('.slds-modal');
191      modal.style.display = 'block';
192    });
193  </script>
194
195  <style>
196    .slds-modal__close {
197      color: #0070C0 !important;
198    }
199  </style>
200
201  <script>
202    window.addEventListener('load', () => {
203      const modal = document.querySelector('.slds-modal');
204      modal.style.display = 'block';
205    });
206  </script>
207
208  <style>
209    .slds-modal__close {
210      color: #0070C0 !important;
211    }
212  </style>
213
214  <script>
215    window.addEventListener('load', () => {
216      const modal = document.querySelector('.slds-modal');
217      modal.style.display = 'block';
218    });
219  </script>
220
221  <style>
222    .slds-modal__close {
223      color: #0070C0 !important;
224    }
225  </style>
226
227  <script>
228    window.addEventListener('load', () => {
229      const modal = document.querySelector('.slds-modal');
230      modal.style.display = 'block';
231    });
232  </script>
233
234  <style>
235    .slds-modal__close {
236      color: #0070C0 !important;
237    }
238  </style>
239
240  <script>
241    window.addEventListener('load', () => {
242      const modal = document.querySelector('.slds-modal');
243      modal.style.display = 'block';
244    });
245  </script>
246
247  <style>
248    .slds-modal__close {
249      color: #0070C0 !important;
250    }
251  </style>
252
253  <script>
254    window.addEventListener('load', () => {
255      const modal = document.querySelector('.slds-modal');
256      modal.style.display = 'block';
257    });
258  </script>
259
260  <style>
261    .slds-modal__close {
262      color: #0070C0 !important;
263    }
264  </style>
265
266  <script>
267    window.addEventListener('load', () => {
268      const modal = document.querySelector('.slds-modal');
269      modal.style.display = 'block';
270    });
271  </script>
272
273  <style>
274    .slds-modal__close {
275      color: #0070C0 !important;
276    }
277  </style>
278
279  <script>
280    window.addEventListener('load', () => {
281      const modal = document.querySelector('.slds-modal');
282      modal.style.display = 'block';
283    });
284  </script>
285
286  <style>
287    .slds-modal__close {
288      color: #0070C0 !important;
289    }
290  </style>
291
292  <script>
293    window.addEventListener('load', () => {
294      const modal = document.querySelector('.slds-modal');
295      modal.style.display = 'block';
296    });
297  </script>
298
299  <style>
300    .slds-modal__close {
301      color: #0070C0 !important;
302    }
303  </style>
304
305  <script>
306    window.addEventListener('load', () => {
307      const modal = document.querySelector('.slds-modal');
308      modal.style.display = 'block';
309    });
310  </script>
311
312  <style>
313    .slds-modal__close {
314      color: #0070C0 !important;
315    }
316  </style>
317
318  <script>
319    window.addEventListener('load', () => {
320      const modal = document.querySelector('.slds-modal');
321      modal.style.display = 'block';
322    });
323  </script>
324
325  <style>
326    .slds-modal__close {
327      color: #0070C0 !important;
328    }
329  </style>
330
331  <script>
332    window.addEventListener('load', () => {
333      const modal = document.querySelector('.slds-modal');
334      modal.style.display = 'block';
335    });
336  </script>
337
338  <style>
339    .slds-modal__close {
340      color: #0070C0 !important;
341    }
342  </style>
343
344  <script>
345    window.addEventListener('load', () => {
346      const modal = document.querySelector('.slds-modal');
347      modal.style.display = 'block';
348    });
349  </script>
350
351  <style>
352    .slds-modal__close {
353      color: #0070C0 !important;
354    }
355  </style>
356
357  <script>
358    window.addEventListener('load', () => {
359      const modal = document.querySelector('.slds-modal');
360      modal.style.display = 'block';
361    });
362  </script>
363
364  <style>
365    .slds-modal__close {
366      color: #0070C0 !important;
367    }
368  </style>
369
370  <script>
371    window.addEventListener('load', () => {
372      const modal = document.querySelector('.slds-modal');
373      modal.style.display = 'block';
374    });
375  </script>
376
377  <style>
378    .slds-modal__close {
379      color: #0070C0 !important;
380    }
381  </style>
382
383  <script>
384    window.addEventListener('load', () => {
385      const modal = document.querySelector('.slds-modal');
386      modal.style.display = 'block';
387    });
388  </script>
389
390  <style>
391    .slds-modal__close {
392      color: #0070C0 !important;
393    }
394  </style>
395
396  <script>
397    window.addEventListener('load', () => {
398      const modal = document.querySelector('.slds-modal');
399      modal.style.display = 'block';
400    });
401  </script>
402
403  <style>
404    .slds-modal__close {
405      color: #0070C0 !important;
406    }
407  </style>
408
409  <script>
410    window.addEventListener('load', () => {
411      const modal = document.querySelector('.slds-modal');
412      modal.style.display = 'block';
413    });
414  </script>
415
416  <style>
417    .slds-modal__close {
418      color: #0070C0 !important;
419    }
420  </style>
421
422  <script>
423    window.addEventListener('load', () => {
424      const modal = document.querySelector('.slds-modal');
425      modal.style.display = 'block';
426    });
427  </script>
428
429  <style>
430    .slds-modal__close {
431      color: #0070C0 !important;
432    }
433  </style>
434
435  <script>
436    window.addEventListener('load', () => {
437      const modal = document.querySelector('.slds-modal');
438      modal.style.display = 'block';
439    });
440  </script>
441
442  <style>
443    .slds-modal__close {
444      color: #0070C0 !important;
445    }
446  </style>
447
448  <script>
449    window.addEventListener('load', () => {
450      const modal = document.querySelector('.slds-modal');
451      modal.style.display = 'block';
452    });
453  </script>
454
455  <style>
456    .slds-modal__close {
457      color: #0070C0 !important;
458    }
459  </style>
460
461  <script>
462    window.addEventListener('load', () => {
463      const modal = document.querySelector('.slds-modal');
464      modal.style.display = 'block';
465    });
466  </script>
467
468  <style>
469    .slds-modal__close {
470      color: #0070C0 !important;
471    }
472  </style>
473
474  <script>
475    window.addEventListener('load', () => {
476      const modal = document.querySelector('.slds-modal');
477      modal.style.display = 'block';
478    });
479  </script>
480
481  <style>
482    .slds-modal__close {
483      color: #0070C0 !important;
484    }
485  </style>
486
487  <script>
488    window.addEventListener('load', () => {
489      const modal = document.querySelector('.slds-modal');
490      modal.style.display = 'block';
491    });
492  </script>
493
494  <style>
495    .slds-modal__close {
496      color: #0070C0 !important;
497    }
498  </style>
499
500  <script>
501    window.addEventListener('load', () => {
502      const modal = document.querySelector('.slds-modal');
503      modal.style.display = 'block';
504    });
505  </script>
506
507  <style>
508    .slds-modal__close {
509      color: #0070C0 !important;
510    }
511  </style>
512
513  <script>
514    window.addEventListener('load', () => {
515      const modal = document.querySelector('.slds-modal');
516      modal.style.display = 'block';
517    });
518  </script>
519
520  <style>
521    .slds-modal__close {
522      color: #0070C0 !important;
523    }
524  </style>
525
526  <script>
527    window.addEventListener('load', () => {
528      const modal = document.querySelector('.slds-modal');
529      modal.style.display = 'block';
530    });
531  </script>
532
533  <style>
534    .slds-modal__close {
535      color: #0070C0 !important;
536    }
537  </style>
538
539  <script>
540    window.addEventListener('load', () => {
541      const modal = document.querySelector('.slds-modal');
542      modal.style.display = 'block';
543    });
544  </script>
545
546  <style>
547    .slds-modal__close {
548      color: #0070C0 !important;
549    }
550  </style>
551
552  <script>
553    window.addEventListener('load', () => {
554      const modal = document.querySelector('.slds-modal');
555      modal.style.display = 'block';
556    });
557  </script>
558
559  <style>
560    .slds-modal__close {
561      color: #0070C0 !important;
562    }
563  </style>
564
565  <script>
566    window.addEventListener('load', () => {
567      const modal = document.querySelector('.slds-modal');
568      modal.style.display = 'block';
569    });
570  </script>
571
572  <style>
573    .slds-modal__close {
574      color: #0070C0 !important;
575    }
576  </style>
577
578  <script>
579    window.addEventListener('load', () => {
580      const modal = document.querySelector('.slds-modal');
581      modal.style.display = 'block';
582    });
583  </script>
584
585  <style>
586    .slds-modal__close {
587      color: #0070C0 !important;
588    }
589  </style>
590
591  <script>
592    window.addEventListener('load', () => {
593      const modal = document.querySelector('.slds-modal');
594      modal.style.display = 'block';
595    });
596  </script>
597
598  <style>
599    .slds-modal__close {
600      color: #0070C0 !important;
601    }
602  </style>
603
604  <script>
605    window.addEventListener('load', () => {
606      const modal = document.querySelector('.slds-modal');
607      modal.style.display = 'block';
608    });
609  </script>
610
611  <style>
612    .slds-modal__close {
613      color: #0070C0 !important;
614    }
615  </style>
616
617  <script>
618    window.addEventListener('load', () => {
619      const modal = document.querySelector('.slds-modal');
620      modal.style.display = 'block';
621    });
622  </script>
623
624  <style>
625    .slds-modal__close {
626      color: #0070C0 !important;
627    }
628  </style>
629
630  <script>
631    window.addEventListener('load', () => {
632      const modal = document.querySelector('.slds-modal');
633      modal.style.display = 'block';
634    });
635  </script>
636
637  <style>
638    .slds-modal__close {
639      color: #0070C0 !important;
640    }
641  </style>
642
643  <script>
644    window.addEventListener('load', () => {
645      const modal = document.querySelector('.slds-modal');
646      modal.style.display = 'block';
647    });
648  </script>
649
650  <style>
651    .slds-modal__close {
652      color: #0070C0 !important;
653    }
654  </style>
655
656  <script>
657    window.addEventListener('load', () => {
658      const modal = document.querySelector('.slds-modal');
659      modal.style.display = 'block';
660    });
661  </script>
662
663  <style>
664    .slds-modal__close {
665      color: #0070C0 !important;
666    }
667  </style>
668
669  <script>
670    window.addEventListener('load', () => {
671      const modal = document.querySelector('.slds-modal');
672      modal.style.display = 'block';
673    });
674  </script>
675
676  <style>
677    .slds-modal__close {
678      color: #0070C0 !important;
679    }
680  </style>
681
682  <script>
683    window.addEventListener('load', () => {
684      const modal = document.querySelector('.slds-modal');
685      modal.style.display = 'block';
686    });
687  </script>
688
689  <style>
690    .slds-modal__close {
691      color: #0070C0 !important;
692    }
693  </style>
694
695  <script>
696    window.addEventListener('load', () => {
697      const modal = document.querySelector('.slds-modal');
698      modal.style.display = 'block';
699    });
700  </script>
701
702  <style>
703    .slds-modal__close {
704      color: #0070C0 !important;
705    }
706  </style>
707
708  <script>
709    window.addEventListener('load', () => {
710      const modal = document.querySelector('.slds-modal');
711      modal.style.display = 'block';
712    });
713  </script>
714
715  <style>
716    .slds-modal__close {
717      color: #0070C0 !important;
718    }
719  </style>
720
721  <script>
722    window.addEventListener('load', () => {
723      const modal = document.querySelector('.slds-modal');
724      modal.style.display = 'block';
725    });
726  </script>
727
728  <style>
729    .slds-modal__close {
730      color: #0070C0 !important;
731    }
732  </style>
733
734  <script>
735    window.addEventListener('load', () => {
736      const modal = document.querySelector('.slds-modal');
737      modal.style.display = 'block';
738    });
739  </script>
740
741  <style>
742    .slds-modal__close {
743      color: #0070C0 !important;
744    }
745  </style>
746
747  <script>
748    window.addEventListener('load', () => {
749      const modal = document.querySelector('.slds-modal');
750      modal.style.display = 'block';
751    });
752  </script>
753
754  <style>
755    .slds-modal__close {
756      color: #0070C0 !important;
757    }
758  </style>
759
760  <script>
761    window.addEventListener('load', () => {
762      const modal = document.querySelector('.slds-modal');
763      modal.style.display = 'block';
764    });
765  </script>
766
767  <style>
768    .slds-modal__close {
769      color: #0070C0 !important;
770    }
771  </style>
772
773  <script>
774    window.addEventListener('load', () => {
775      const modal = document.querySelector('.slds-modal');
776      modal.style.display = 'block';
777    });
778  </script>
779
780  <style>
781    .slds-modal__close {
782      color: #0070C0 !important;
783    }
784  </style>
785
786  <script>
787    window.addEventListener('load', () => {
788      const modal = document.querySelector('.slds-modal');
789      modal.style.display = 'block';
790    });
791  </script>
792
793  <style>
794    .slds-modal__close {
795      color: #0070C0 !important;
796    }
797  </style>
798
799  <script>
800    window.addEventListener('load', () => {
801      const modal = document.querySelector('.slds-modal');
802      modal.style.display = 'block';
803    });
804  </script>
805
806  <style>
807    .slds-modal__close {
808      color: #0070C0 !important;
809    }
810  </style>
811
812  <script>
813    window.addEventListener('load', () => {
814      const modal = document.querySelector('.slds-modal');
815      modal.style.display = 'block';
816    });
817  </script>
818
819  <style>
820    .slds-modal__close {
821      color: #0070C0 !important;
822    }
823  </style>
824
825  <script>
826    window.addEventListener('load', () => {
827      const modal = document.querySelector('.slds-modal');
828      modal.style.display = 'block';
829    });
830  </script>
831
832  <style>
833    .slds-modal__close {
834      color: #0070C0 !important;
835    }
836  </style>
837
838  <script>
839    window.addEventListener('load', () => {
840      const modal = document.querySelector('.slds-modal');
841      modal.style.display = 'block';
842    });
843  </script>
844
845  <style>
846    .slds-modal__close {
847      color: #0070C0 !important;
848    }
849  </style>
850
851  <script>
852    window.addEventListener('load', () => {
853      const modal = document.querySelector('.slds-modal');
854      modal.style.display = 'block';
855    });
856  </script>
857
858  <style>
859    .slds-modal__close {
860      color: #0070C0 !important;
861    }
862  </style>
863
864  <script>
865    window.addEventListener('load', () => {
866      const modal = document.querySelector('.slds-modal');
867      modal.style.display = 'block';
868    });
869  </script>
870
871  <style>
872    .slds-modal__close {
873      color: #0070C0 !important;
874    }
875  </style>
876
877  <script>
878    window.addEventListener('load', () => {
879      const modal = document.querySelector('.slds-modal');
880      modal.style.display = 'block';
881    });
882  </script>
883
884  <style>
885    .slds-modal__close {
886      color: #0070C0 !important;
887    }
888  </style>
889
890  <script>
891    window.addEventListener('load', () => {
892      const modal = document.querySelector('.slds-modal');
893      modal.style.display = 'block';
894    });
895  </script>
896
897  <style>
898    .slds-modal__close {
899      color: #0070C0 !important;
900    }
901  </style>
902
903  <script>
904    window.addEventListener('load', () => {
905      const modal = document.querySelector('.slds-modal');
906      modal.style.display = 'block';
907    });
908  </script>
909
910  <style>
911    .slds-modal__close {
912      color: #0070C0 !important;
913    }
914  </style>
915
916  <script>
917    window.addEventListener('load', () => {
918      const modal = document.querySelector('.slds-modal');
919      modal.style.display = 'block';
920    });
921  </script>
922
923  <style>
924    .slds-modal__close {
925      color: #0070C0 !important;
926    }
927  </style>
928
929  <script>
930    window.addEventListener('load', () => {
931      const modal = document.querySelector('.slds-modal');
932      modal.style.display = 'block';
933    });
934  </script>
935
936  <style>
937    .slds-modal__close {
938      color: #0070C0 !important;
939    }
940  </style>
941
942  <script>
943    window.addEventListener('load', () => {
944      const modal = document.querySelector('.slds-modal');
945      modal.style.display = 'block';
946    });
947  </script>
948
949  <style>
950    .slds-modal__close {
951      color: #0070C0 !important;
952    }
953  </style>
954
955  <script>
956    window.addEventListener('load', () => {
957      const modal = document.querySelector('.slds-modal');
958      modal.style.display = 'block';
959    });
960  </script>
961
962  <style>
963    .slds-modal__close {
964      color: #0070C0 !important;
965    }
966  </style>
967
968  <script>
969    window.addEventListener('load', () => {
970      const modal = document.querySelector('.slds-modal');
971      modal.style.display = 'block';
972    });
973  </script>
974
975  <style>
976    .slds-modal__close {
977      color: #0070C0 !important;
978    }
979  </style>
980
981  <script>
982    window.addEventListener('load', () => {
983      const modal = document.querySelector('.slds-modal');
984      modal.style.display = 'block';
985    });
986  </script>
987
988  <style>
989    .slds-modal__close {
990      color: #0070C0 !important;
991    }
992  </style>
993
994  <script>
995    window.addEventListener('load', () => {
996      const modal = document.querySelector('.slds-modal');
997      modal.style.display = 'block';
998    });
999  </script>
1000 </script>
```



```
JS leaveRequests.js
force-app > main > default > lwc > leaveRequests > leaveRequests.js
1  import { LightningElement, wire } from 'lwc';
2  import getLeaveRequests from '@salesforce/apex/LeaveRequestController.getLeaveRequests';
3  import { ShowToastEvent } from 'lightning/platformShowToastEvent';
4  import Id from '@salesforce/user/Id';
5  import { refreshApex } from '@salesforce/apex';
6
7  const COLUMNS = [
8    { label: 'Request Id', fieldName: 'Name', cellAttributes: { class: { fieldName: 'cellClass' } } },
9    { label: 'User', fieldName: 'userName', cellAttributes: { class: { fieldName: 'cellClass' } } },
10   { label: 'From Date', fieldName: 'From Date_c', cellAttributes: { class: { fieldName: 'cellClass' } } },
11   { label: 'To Date', fieldName: 'To Date_c', cellAttributes: { class: { fieldName: 'cellClass' } } },
12   { label: 'Reason', fieldName: 'Reason_c', cellAttributes: { class: { fieldName: 'cellClass' } } },
13   { label: 'Status', fieldName: 'Status_c', cellAttributes: { class: { fieldName: 'cellClass' } } },
14   { label: 'Manager Comment', fieldName: 'Manager Comment_c', cellAttributes: { class: { fieldName: 'cellClass' } } },
15 ]
16
17 export default class LeaveRequests extends LightningElement {
18   columns = COLUMNS;
19
20   leavesRequests = [];
21   leavesRequestsWireResult;
22   showModalPopup = false;
23   objectApiName = 'LeaveRequest_c';
24   recordId = '';
25   currentUserId = Id;
26
27   @wire(getLeaveRequests)
28   wiredLeaves(result) {
29     this.leavesRequestsWireResult = result;
30     if (result.data) {
31       this.leavesRequests = result.data;
32     }
33   }
34
35   handleRowAction(event) {
36     const row = event.detail.row;
37     const editButton = row.querySelector('.edit');
38     const editLabel = row.querySelector('.editLabel');
39     const editTitle = row.querySelector('.editTitle');
40     const editValue = row.querySelector('.editValue');
41     const editDisabled = row.querySelector('.editDisabled');
42     const cellAttributes = row.cellAttributes;
43
44     if (editButton) {
45       editButton.click();
46     }
47     if (editLabel) {
48       editLabel.click();
49     }
50     if (editTitle) {
51       editTitle.click();
52     }
53     if (editValue) {
54       editValue.click();
55     }
56     if (editDisabled) {
57       editDisabled.click();
58     }
59     if (cellAttributes) {
60       cellAttributes.click();
61     }
62   }
63
64   handleEditClick() {
65     const editButton = this.querySelector('.edit');
66     const editLabel = this.querySelector('.editLabel');
67     const editTitle = this.querySelector('.editTitle');
68     const editValue = this.querySelector('.editValue');
69     const editDisabled = this.querySelector('.editDisabled');
70     const cellAttributes = this.cellAttributes;
71
72     if (editButton) {
73       editButton.click();
74     }
75     if (editLabel) {
76       editLabel.click();
77     }
78     if (editTitle) {
79       editTitle.click();
80     }
81     if (editValue) {
82       editValue.click();
83     }
84     if (editDisabled) {
85       editDisabled.click();
86     }
87     if (cellAttributes) {
88       cellAttributes.click();
89     }
90   }
91
92   handleEditLabelClick() {
93     const editLabel = this.querySelector('.editLabel');
94     const editTitle = this.querySelector('.editTitle');
95     const editValue = this.querySelector('.editValue');
96     const editDisabled = this.querySelector('.editDisabled');
97     const cellAttributes = this.cellAttributes;
98
99     if (edit
```

```

<?xml version="1.0" encoding="UTF-8"?>
<lightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>56.0</apiVersion>
    <isExposed>true</isExposed>
</lightningComponentBundle>

```

leaveRequests.js-meta.xml

- The ScreenShots for myLeaves:

```

<template>
    <lightning-card>
        <lightning-button-icon icon-name="utility:add" variant="border-filled" slot="actions"
            onclick="newRequestClickHandler"></lightning-button-icon>
        <lightning-dataTable key-field="Id" data="myLeaves" columns={columns}
            onrowaction={rowActionHandler}></lightning-dataTable>
    </template>
    <template lwc:if={noRecordsFound}>
        <div class="slds-align_absolute-center slds-p-around_small slds-text-heading_medium">No Records Found</div>
    </template>
</lightning-card>

<template lwc:if={(showModalPopup)}>
    <section role="dialog" tabindex="-1" aria-modal="true" aria-labelledby="modal-heading-e1"
        class="slds-modal slds-fade-in-open slds-modal_small">
        <div class="slds-modal__container">
            <button onclick={popupCloseHandler}>
                <lightning-icon icon-name="utility:close" alternative-text="close" variant="inverse">
                    <span>Close</span>
                </lightning-icon>
            </button>
            <div class="slds-modal__header">
                <h1 id="modal-heading-e1" class="slds-modal__title slds-hyphenate">Modal header</h1>
            </div>
            <div class="slds-modal__content slds-p-around_medium" id="modal-content-id-1">
                <lightning-record-edit-form object-api-name={objectApiName} record-id={recordId}>
                    <onsuccess>{successHandler}</onsuccess>
                    <onsubmit>{submitHandler}</onsubmit>
                    <lwc:ref="leaveRequestForm"></lwc:ref>
                    <lightning-input-field field-name="User_c" value={currentUserId}></lightning-input-field>
                    <lightning-input-field field-name="From Date_c"></lightning-input-field>
                    <lightning-input-field field-name="To Date_c"></lightning-input-field>
                    <lightning-input-field field-name="Reason_c"></lightning-input-field>
                    <div class="slds-var-m-top_medium">
                        <lightning-button variant="brand" type="submit" label="Save"></lightning-button>
                    </div>
                    <lightning-button label="Cancel" class="slds-m-left_small" onclick={popupCloseHandler}></lightning-button>
                </lightning-record-edit-form>
            </div>
        </div>
    </section>
</template>

```

myLeaves.html

```

    const COLUMNS = [
        {
            label: 'Manager Comment', fieldName: 'Manager_Comment_c', cellAttributes: { class: { fieldName: 'cellClass' } }
        }
    ];
    export default class MyLeaves extends LightningElement {
        columns = COLUMNS;
        myLeaves = [];
        myLeavesWireResult;
        showModalPopup = false;
        objectApiName = 'LeaveRequest__c';
        recordId = '';
        currentUserID = Id;
        @wire(getMyLeaves)
        wiredMyLeaves(result) {
            this.myLeavesWireResult = result;
            if (result.data) {
                this.myLeaves = result.data.map(a => ({
                    ...a,
                    cellClass: a.Status__c === 'Approved' ? 'slds-theme_success' : a.Status__c === 'Rejected' ? 'slds-theme_warning' : '',
                    isEditDisabled: a.Status__c !== 'Pending'
                }));
            }
            if (result.error) {
                console.log('Error occurred while fetching my leaves-', result.error);
            }
        }
    }

```

myLeaves.js

```

<xml version="1.0" encoding="UTF-8?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>56.0</apiVersion>
    <isExposed>true</isExposed>
</LightningComponentBundle>

```

myLeaves.js-meta.xml

(note: All the ‘Parent and ‘Child’ component’s “-meta.xml” file’s [`<isExposed>` `</isExposed>`] should be ‘true’.)

3. Lightning App Creation/Implementation:

After completion of the codes and deploying it on the salesforce org we have to create a Lightning App and add the “custom component” developed using LWC and VScode in it.

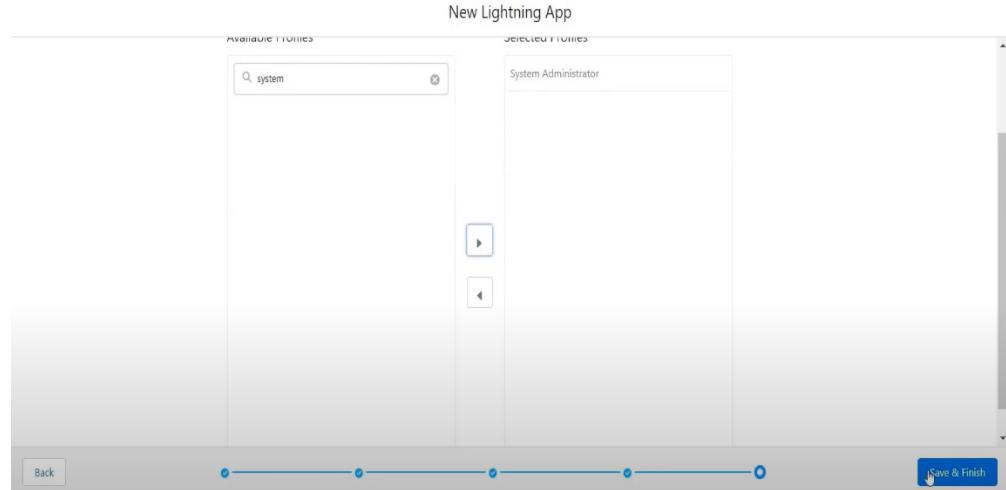
Step 1:

Goto setup→App manager→New Lightning App

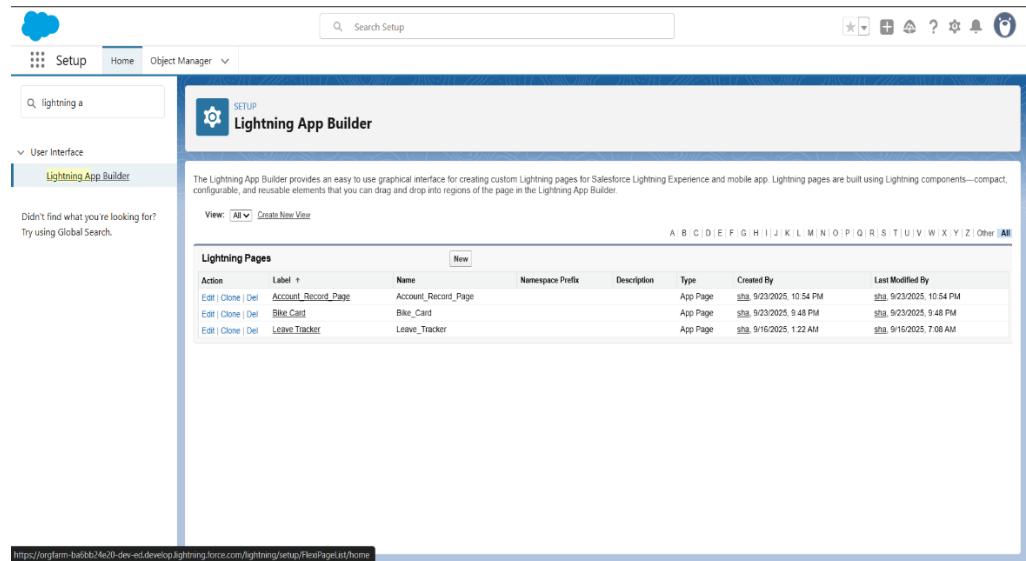
The screenshot shows the Salesforce Setup Home page. In the top left, there's a search bar with "app manager" typed in. Below it, the "App Manager" link is highlighted in the "External Client App Manager" section of the sidebar. The main content area features three cards: "Data Cloud" (Setup), "Get Started with Einstein Bots" (Setup), and "Mobile Publisher" (Setup). Below these cards is a section titled "Most Recently Used" which lists "System Administrator" (Profile), "Collaboration" (Custom App), and "User" (User). At the bottom of the page, the URL "orgfarm-ba6bb24e20-dev-ed.lightning.force.com/lightning/setup.../home" is visible.

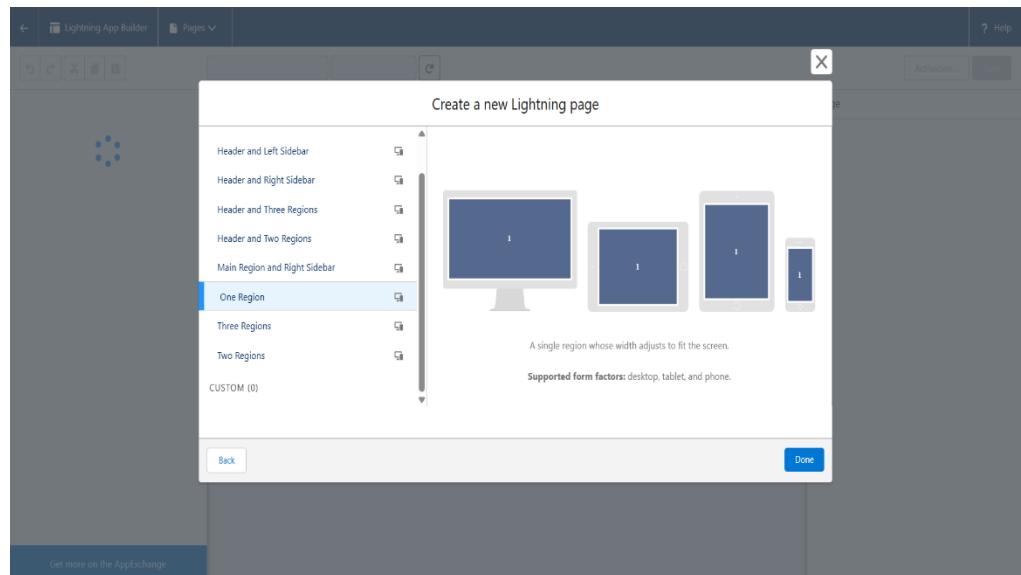
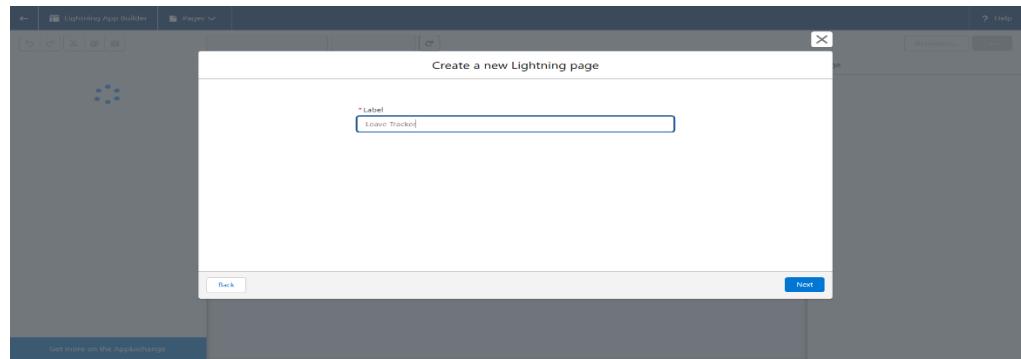
The screenshot shows the "New Lightning App" configuration page. The title "New Lightning App" is at the top. The page is divided into two main sections: "App Details" and "App Branding". Under "App Details", there are fields for "App Name" (set to "Leave Tracker App"), "Developer Name" (with a placeholder "Enter a developer name..."), and "Description" (with a placeholder "Enter a description..."). Under "App Branding", there are fields for "Image" (with a placeholder "Upload") and "Primary Color Hex" (set to "#0070D2"). A checkbox "Org Theme Options" is checked, with a note below it stating "Use the app's image and color instead of the org's custom theme". At the bottom right, there is a "Next" button. The bottom of the screen shows a list of recent apps, including "Data Manager", "Digital Experiences", and "Leave Tracker App".

- After inserting the App name “Leave Tracker App” click on Next→Next→Next→Next→Now select “System Administrator” and click Save & Finish.

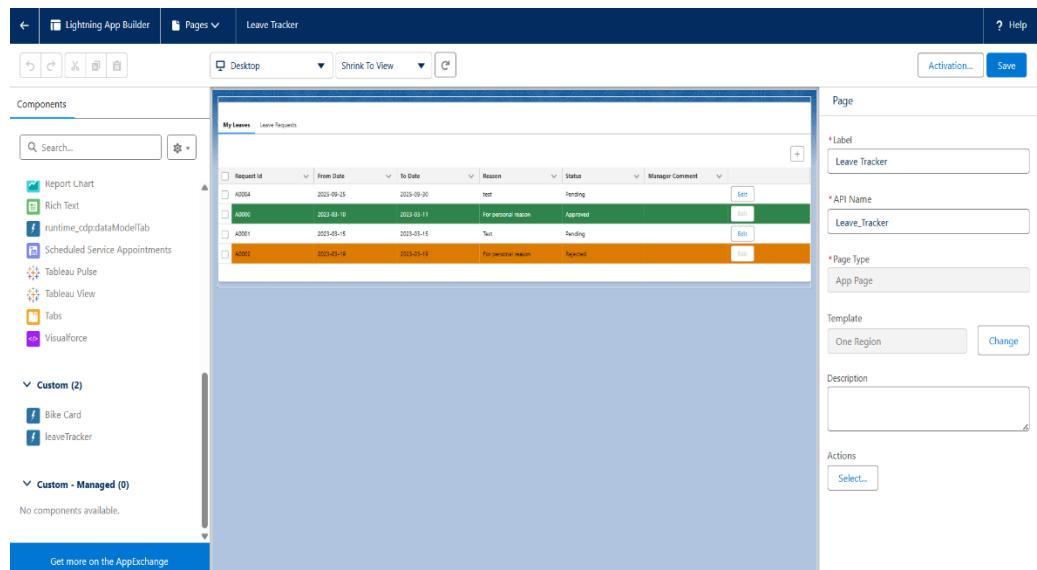


- Now goto home→ Lightning App Builder→ New and label it “leave Tracker”. (select the App page before it then hit Next then select One region).

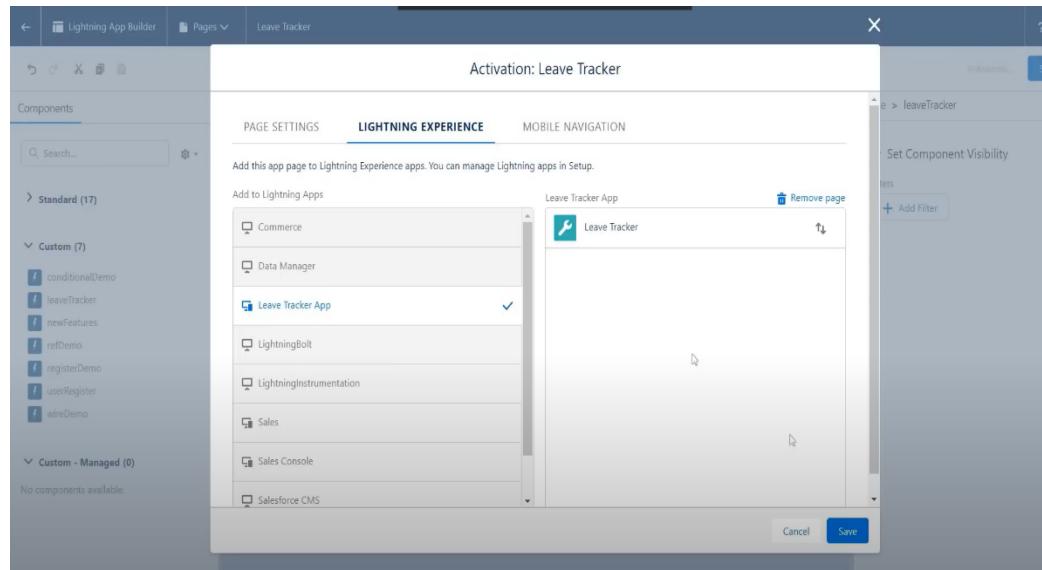




- Now drag and drop the Custom 'leave Tracker' and then save and activate.



- Now save it and activate it (also in the Lightning Experience add this to Leave Tracker App we created Earlier).



- Also I implemented the some other things like:
 - a. Wire decorator implementation
 - b. Edit button click handler
 - c. Add button implementation
 - d. Grid auto-refresh using refreshApex (imperative Apex call)
 - e. Custom event creation and dispatch
 - f. Component communication

These all are implemented using VS code and then deployed using [SFDX: Deploy this source to org].
(note: I have pushed all the codes in the GitHub Repository, so you will abe to see these implimentations in the code).

Phase 7: Integration & External Access (not applicable for the project/only Conceptually)

Purpose:

This phase aimed to connect Salesforce with external systems, such as a policyholder portal and third-party payment services, ensuring seamless data exchange and notification delivery.

1. Request and Reply Integration:

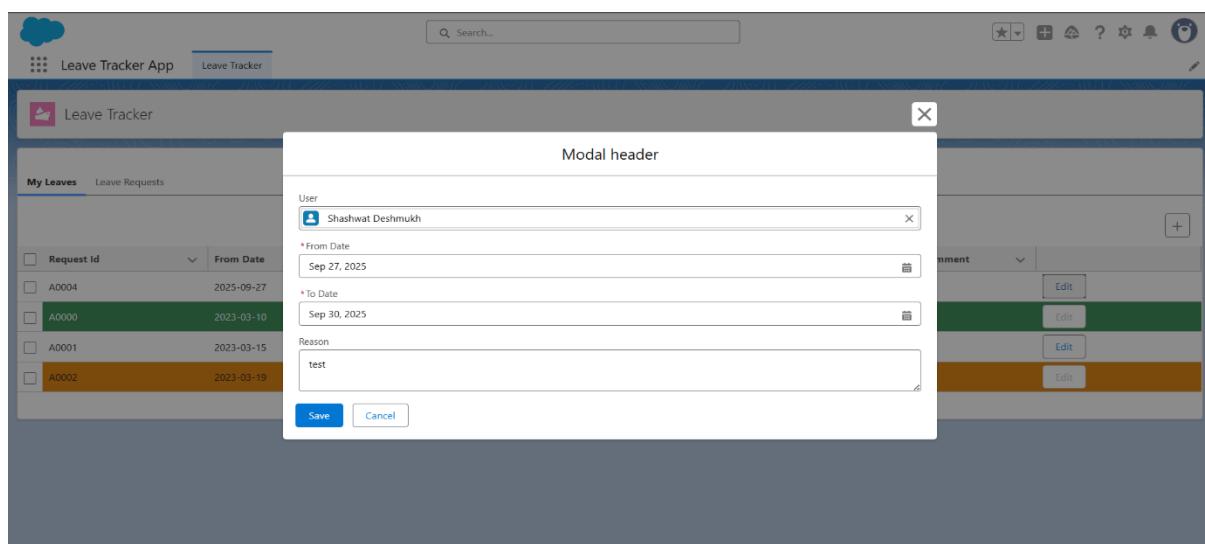
The Leave Tracker App implements Request & Reply pattern for real-time interactions with external systems. When employees submit leave requests, the system can instantly verify leave balances from external payroll systems and return approval status.

2. Batch Data Synchronization:

For large-scale data operations, the app utilizes Batch Data Synchronization pattern to sync employee data, leave policies, and historical leave records. This ensures data consistency across multiple systems without impacting real-time user experience.

3. Data Virtualization with Salesforce Connect

External employee data from HRMS systems can be accessed in real-time using Salesforce Connect without storing duplicate records. This zero-copy approach maintains data integrity while providing unified access to employee information.





Search...



<input type="checkbox"/> A0020	OrgFarm EPIC	2025-09-27	2025-09-29	uo	Pending	<button>Edit</button>	
<input type="checkbox"/> A0019	OrgFarm EPIC	2025-09-25	2025-09-25	t	Pending	<button>Edit</button>	
<input type="checkbox"/> A0018	OrgFarm EPIC	2025-09-18	2025-09-21	..	Pending	<button>Edit</button>	
<input type="checkbox"/> A0017	OrgFarm EPIC	2025-09-18	2025-09-19	you	Pending	<button>Edit</button>	
<input type="checkbox"/> A0016	OrgFarm EPIC	2025-09-18	2025-09-22	dexter	Pending	<button>Edit</button>	
<input type="checkbox"/> A0015	OrgFarm EPIC	2025-09-17	2025-09-30	rest	Pending	<button>Edit</button>	
<input type="checkbox"/> A0014	OrgFarm EPIC	2025-09-20	2025-09-21	domain	Approved	approved!!!	<button>Edit</button>
<input type="checkbox"/> A0013	OrgFarm EPIC	2025-09-17	2025-09-27	Pending	<button>Edit</button>	
<input type="checkbox"/> A0012	OrgFarm EPIC	2025-09-18	2025-09-26	emergency	<button>Edit</button>		
<input type="checkbox"/> A0011	OrgFarm EPIC	2025-09-17	2025-09-26	..	Pending	<button>Edit</button>	
<input type="checkbox"/> A0010	OrgFarm EPIC	2025-09-17	2025-09-27	leave	Rejected	can't	<button>Edit</button>
<input type="checkbox"/> A0009	OrgFarm EPIC	2025-09-17	2025-09-26	.	Pending	<button>Edit</button>	
<input type="checkbox"/> A0008	OrgFarm EPIC	2025-09-18	2025-09-25	test3	Approved	approved!	<button>Edit</button>
<input type="checkbox"/> A0007	OrgFarm EPIC	2025-09-18	2025-09-21	test2	Approved	yes ok	<button>Edit</button>
<input type="checkbox"/> A0006	OrgFarm EPIC	2025-09-18	2025-09-30	best	Pending	<button>Edit</button>	
<input type="checkbox"/> A0005	OrgFarm EPIC	2025-09-17	2025-09-24	test	Rejected	nope	<button>Edit</button>

Phase 8: Data Management & Deployment

Purpose:

The purpose of this phase was to ensure accurate data migration, prevent duplication, and deploy the solution across different Salesforce environments securely.

1. VS Code & SFDX:

VS Code (Visual Studio Code) is a free, open-source code editor developed by Microsoft that serves as the primary development environment for Salesforce developers. It's lightweight, extensible, and available across Windows, Linux, and macOS platforms.

SFDX (Salesforce DX) is Salesforce's modern development platform that includes a commandline interface (CLI) and development tools designed for source-driven development. It represents the official future of Salesforce development, offering enhanced capabilities for working with metadata, scratch orgs, and version control systems.

2. How they work together:

VS Code and SFDX integrate seamlessly through the Salesforce Extension Pack, which brings SFDX functionality directly into the VS Code interface. This combination enables developers to create projects, authorize orgs, retrieve source code, deploy changes, and manage Salesforce metadata all within a unified development environment.

After Completing the code and debugging all the errors it is then deployed to the salesforce org using [SFDX Deploy this source to org]

EXPLORER ... JS myLeaves.js X myLeaves.html

LEAVE TRACKER APP

- > .husky
- > .sf
- > .sfdx
- > .vscode
- > config
- > force-app\main
 - > applications
 - > aura
 - > classes
 - LeaveRequest
 - LeaveRequest
 - > contentassets
 - > flexipages
 - > layouts
 - > lwc
 - > objects\LeaveRequest
 - > fields
 - From_Date_c
 - Manager_Co
 - Reason_c.f
 - Status_c.f
 - To_Date_c.f
 - User_c.field
 - LeaveRequest
 - > permissionsets
 - > staticresources
 - > tabs
- > OUTLINE
- > TIMELINE
- > RUNNING TASKS

New File... New Folder... Reveal in File Explorer Open in Integrated Terminal Shift+Alt+R

Find in Folder... Shift+Alt+F

Add Folder to Chat

Cut Ctrl+X Copy Ctrl+C Paste Ctrl+V

Copy Path Shift+Alt+C Copy Relative Path Ctrl+K Ctrl+Shift+C

Rename... F2 Delete Delete

SFDX: Clear Code Analyzer Violations from Selected Files or Folders

SFDX: Delete from Project and Org

SFDX: Deploy This Source to Org

SFDX: Diff Folder Against Org

SFDX: Generate Manifest File

SFDX: Retrieve This Source from Org

SFDX: Scan Selected Files or Folders with Code Analyzer

```
force-app > main > default > lwc > myLeaves.js
24   export default class MyLeave {
73     submitHandler(event) {
80       else if (new Date() >
81         this.showToast(
82           }
83         else {
84           this.showError(
85         )
86       }
87     }
88   }
89 }
```

Phase 9: Reporting, Dashboards & Security Review

Purpose:

The goal of this phase was to provide managers and agents with actionable insights through Salesforce reports and dashboards, enabling better decision-making and performance tracking.

1. Reports & Dashboards:

Leave Summary Reports: Create tabular reports showing employee leave requests by status (Pending, Approved, Rejected) and summary reports grouped by department and manager. Include matrix reports for comparing leave usage across different time periods.

Leave Analytics Dashboard: Build dashboards displaying key metrics like total leave requests, pending approvals, and manager workload distribution. Add dynamic dashboard components that filter data based on user roles.

Custom Report Types: Create report types combining Leave Request records with User data to enable detailed leave pattern analysis.

2. Security Implementations:

Sharing Settings: Configure Organization-Wide Defaults for Leave Request object to Private, with sharing rules allowing managers to see their team's requests and HR to access all records.

Field Level Security: Restrict access to sensitive fields like Manager Comments to appropriate user profiles only. Ensure employees can only edit their own pending requests.

Audit Trail: Enable field history tracking on critical fields like Status and Approval Date for compliance monitoring.

Reports

- Employee Leave Report: Total leaves taken by each employee in a period.
- Leave Type Report: Breakdown of Sick Leave, Casual Leave, etc.
- Pending Requests Report: Leaves still awaiting manager approval.

The screenshot shows a Salesforce Lightning Report titled "Report: Employee Leave Requests" with a sub-header "New Employee Leave Requests Report". The report displays 25 total records in a table format. The columns include: Leave Request ID, From Date, To Date, Full Name, Leave Request Id, Reason, and Status. The data shows various leave requests from different employees with different reasons and statuses (Pending, Approved, Rejected).

	Leave Request ID	From Date	To Date	Full Name	Leave Request Id	Reason	Status
1	a00gl.00000JULKH	9/28/2025	9/30/2025	OrgFarm EPIC	A0004	test.	Pending
2	a00gl.00000JURJE	9/18/2025	9/30/2025	OrgFarm EPIC	A0003	bu	Pending
3	a00gl.00000JlaEU	3/10/2023	3/11/2023	Shashwat Deshmukh	A0000	For personal reason	Approved
4	a00gl.00000JuEV	9/28/2025	9/29/2025	Shashwat Deshmukh	A0001	Test	Pending
5	a00gl.00000JuEW	3/19/2023	3/19/2023	Shashwat Deshmukh	A0002	For personal reason	Rejected
6	a00gl.00000JuCwC	9/18/2025	9/21/2025	OrgFarm EPIC	A0007	test2	Approved
7	a00gl.00000JUID4	9/17/2025	9/27/2025	OrgFarm EPIC	A0010	leave	Rejected
8	a00gl.00000JUmqm	9/17/2025	9/27/2025	OrgFarm EPIC	A0013	---	Pending
9	a00gl.00000JuqN	9/17/2025	9/24/2025	OrgFarm EPIC	A0005	test	Rejected
10	a00gl.00000JuHtv	9/18/2025	9/22/2025	OrgFarm EPIC	A0016	dexter	Pending
11	a00gl.00000JUyv	9/18/2025	9/19/2025	OrgFarm EPIC	A0017	you	Pending
12	a00gl.00000JuJtr	9/18/2025	9/21/2025	OrgFarm EPIC	A0018	--	Pending
13	a00gl.00000JuTyd	9/17/2025	9/26/2025	OrgFarm EPIC	A0009	-	Pending
14	a00gl.00000Ju7p	9/18/2025	9/26/2025	OrgFarm EPIC	A0012	emergency	-

Audit Trail

- Track all changes to LeaveRequest__c records.
- Example: Who approved/rejected a leave, and when.
- Provides compliance visibility for HR audits.

The screenshot shows the "View Setup Audit Trail" page. The left sidebar lists various setup categories like Contact Intelligence, Lead Intelligence, and Security. The main content area shows a table of audit entries with columns: Date, User, Source Namespace Prefix, Action, Section, and Delegate User. The table lists numerous actions performed by the user "shashvatdeshmukh1277@agentforce.com" on different dates, such as changing Lightning Web Components and Apex classes.

Date	User	Source Namespace Prefix	Action	Section	Delegate User
9/26/2025, 11:35:02 PM PDT	shashvatdeshmukh1277@agentforce.com		Completed Owner Rule: Leave Request recalculation: employee Leave	Sharing Rules	
9/26/2025, 11:35:01 PM PDT	shashvatdeshmukh1277@agentforce.com		Created LeaveRequest Owner Sharing Rule employee Leave	Sharing Rules	
9/26/2025, 11:35:01 PM PDT	shashvatdeshmukh1277@agentforce.com		Initiated Owner Rule: Leave Request recalculation: employee Leave	Sharing Rules	
9/26/2025, 11:32:50 PM PDT	shashvatdeshmukh1277@agentforce.com		Completed sharing rule recalculation: Leave Request	Sharing Rules	
9/26/2025, 11:32:49 PM PDT	shashvatdeshmukh1277@agentforce.com		Initiated sharing rule recalculation: Leave Request	Sharing Rules	
9/26/2025, 11:32:37 PM PDT	shashvatdeshmukh1277@agentforce.com		Changed myLeaves Lightning Web Component	Lightning Components	
9/26/2025, 5:16:23 PM PDT	shashvatdeshmukh1277@agentforce.com		Changed myLeaves Lightning Web Component	Lightning Components	
9/26/2025, 5:15:45 PM PDT	shashvatdeshmukh1277@agentforce.com		Changed myLeaves Lightning Web Component	Lightning Components	
9/26/2025, 5:11:25 PM PDT	shashvatdeshmukh1277@agentforce.com		Changed LeaveRequestController Apex Class code	Apex Class	
9/26/2025, 5:11:25 PM PDT	shashvatdeshmukh1277@agentforce.com		Changed LeaveRequestSampleData Apex Class code	Apex Class	
9/26/2025, 5:11:23 PM PDT	shashvatdeshmukh1277@agentforce.com		Changed myLeaves Lightning Web Component	Lightning Components	
9/26/2025, 5:11:23 PM PDT	shashvatdeshmukh1277@agentforce.com		Changed leaveTracker Lightning Web Component	Lightning Components	
9/26/2025, 5:11:23 PM PDT	shashvatdeshmukh1277@agentforce.com		Changed leaveRequests Lightning Web Component	Lightning Components	
9/26/2025, 5:11:13 PM PDT	shashvatdeshmukh1277@agentforce.com		Changed myLeaves Lightning Web Component	Lightning Components	
9/26/2025, 5:06:32 PM PDT	shashvatdeshmukh1277@agentforce.com		Changed myLeaves Lightning Web Component	Lightning Components	
9/26/2025, 4:19:02 PM PDT	shashvatdeshmukh1277@agentforce.com		Changed Lightning Page: Leave Tracker	Lightning Pages	
9/26/2025, 4:18:59 PM PDT	shashvatdeshmukh1277@agentforce.com		Changed Lightning Page: Leave Tracker	Lightning Pages	

Phase 10: Final Presentation & Demo Day

Purpose:

The final phase ensured smooth knowledge transfer to stakeholders through demos, documentation, and training. It prepared the project for production use.

1. Project Presentation:

Business Impact Overview: Present how the Leave Tracker App transformed manual leave processes into an automated Lightning Web Component solution. Highlight key improvements like real-time updates and streamlined approval workflows.

Technical Showcase: Demonstrate the LWC architecture including wire decorators, component communication between "My Leaves" and "Leave Requests" tabs, and Lightning Data Services integration.

2. Live Demo:

End-to-End Walkthrough: Show complete leave request process from employee submission through manager approval, highlighting modal popups, form validations, and automatic grid refresh without page reload.

Component Communication Demo: Demonstrate how creating requests in one tab automatically updates the other tab through custom events and parent-child component interaction.

3. Documentation & Feedback:

Handoff Materials: Provide technical documentation covering component structure, deployment using VS Code/SFDX, and administrator configuration guides.

User Training: Create step-by-step guides for both employees and managers covering the new Lightning Web Component interface.

Code Repository: Upload clean, documented code to GitHub demonstrating LWC best practices, wire adapters, and component communication patterns.

Demo Video: Create screen recording showcasing the Leave Tracker App functionality with technical explanations suitable for portfolio presentation.

This simplified approach focuses on the core deliverables that directly relate to the Leave Management System demonstrated in the video, emphasizing practical implementation over extensive theoretical details.