Project Title: Employee Wellness & Productivity Tracker

Phase 6 : User Interface Development

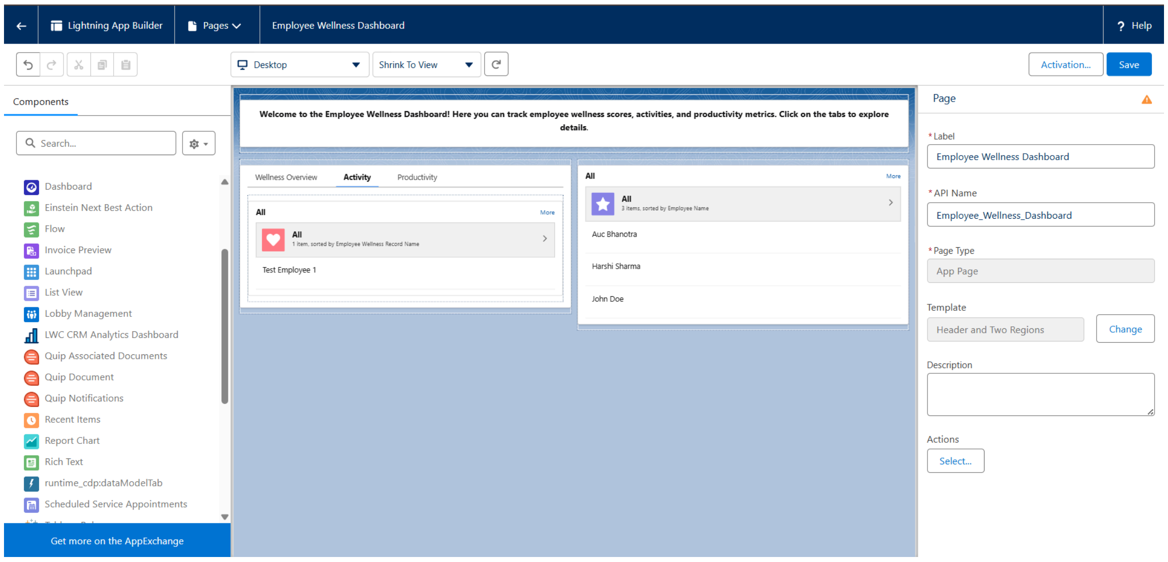
LIGHTNING APP BUILDER

Purpose

The Lightning App Builder allows Salesforce admins and developers to **create custom pages for users without writing code**. In this project, it is used to build an **interactive Employee Wellness Dashboard** where HR and managers can **quickly view employee wellness, activities, and productivity metrics** in one place.

Use Case / Example

- HR wants to **monitor wellness scores** across all employees.
- Managers want to **track activity completion** and **identify employees needing attention**.
- The dashboard provides **visual charts, list views, and optional LWCs** to make data **easy to interpret and**



Welcome to the Employee Wellness Dashboard! Here you can track employee wellness scores, activities, and productivity metrics. Click on the tabs to explore details.

* Label: Employee Wellness Dashboard

* API Name: Employee_Wellness_Dashboard

* Page Type: App Page

Template: Header and Two Regions

TABS	DATA	OBJECT/ SOURCE
Wellness Overview	Report Chart	Employee
Activity	List View	Employee Wellness Activity
Productivity	Report Chart	Productivity Tasks

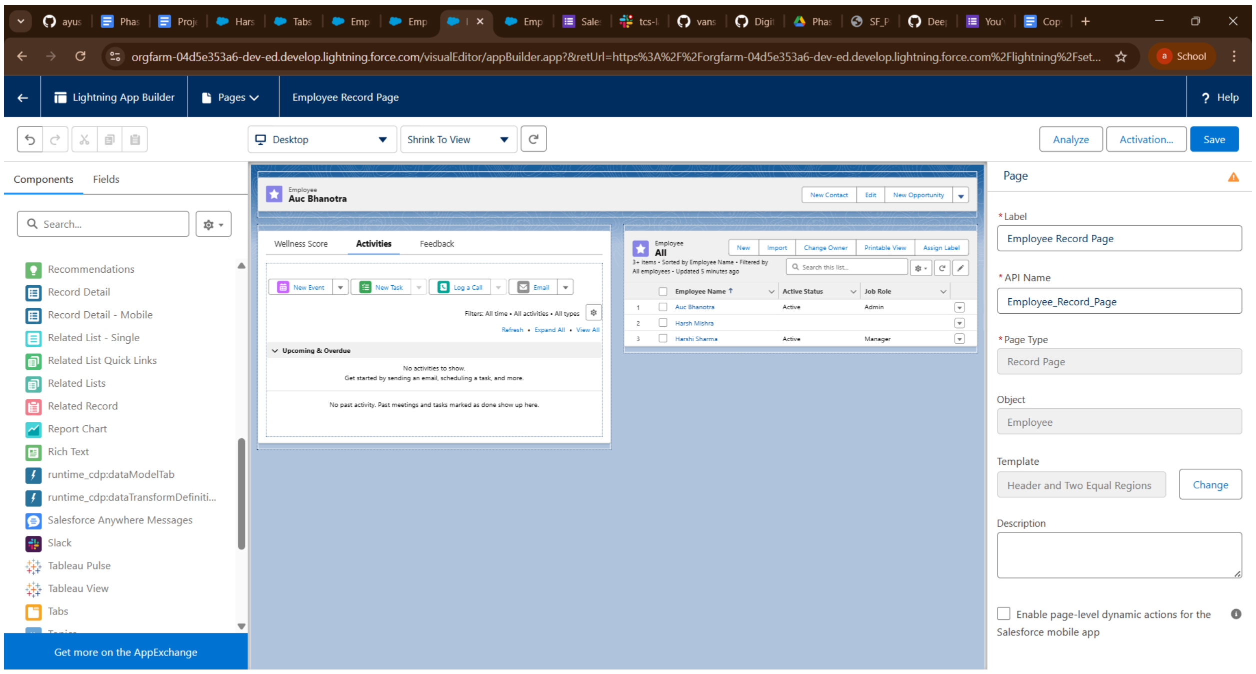
Steps to Build (Optional in PDF)

1. Go to Setup → Lightning App Builder → New → App Page
2. Name: `Employee Wellness Dashboard`
3. Choose One or Two Regions layout
4. Drag Tabs component, Report Charts, List Views, LWCs into the page
5. Configure tabs and components with proper data sources
6. Save → Activate → Assign to HR/Manager profiles

RECORD PAGES

Purpose:

To allow HR and managers to **view and manage individual employee data**, including wellness scores, activities, productivity, and feedback — all in one page.



	Employee Name	Active Status	Job Role
1	Auc Bhanotra	Active	Admin
2	Harsh Mishra		
3	Harshi Sharma	Active	Manager

Page

* Label: Employee Record Page

* API Name: Employee_Record_Page

* Page Type: Record Page

Object: Employee

Template: Header and Two Equal Regions

Description:

☐ Enable page-level dynamic actions for the Salesforce mobile app

Benefits

- Centralized view of **employee wellness and productivity metrics**
- Quick access to **related tasks, activities, and feedback**
- **Easy navigation** through tabs
- Supports **future enhancements** with custom LWCs for interactive dashboards

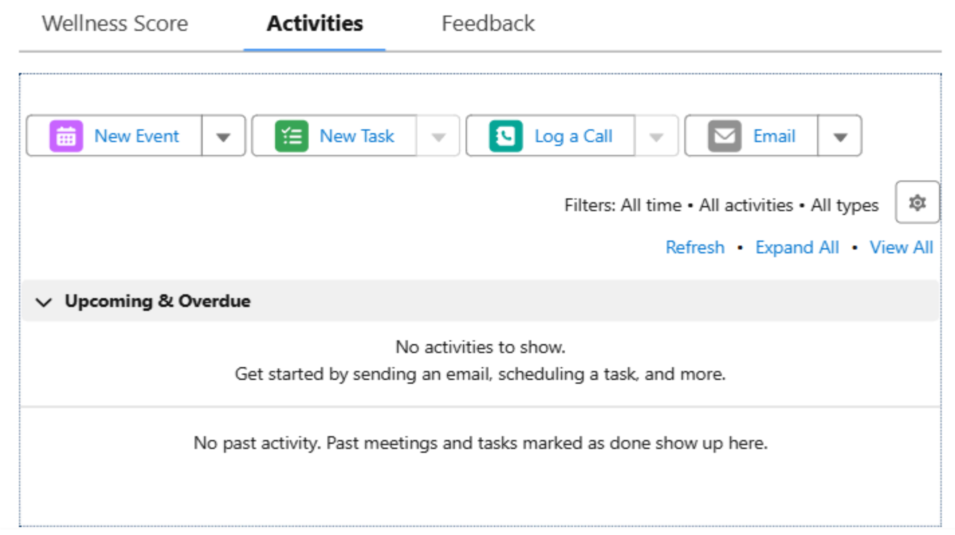
TABS

Purpose

The **Tabs component** organizes content into separate sections on a Record Page, making it easier for HR/managers to view **wellness, activities, feedback, and productivity** without scrolling endlessly.

Use Case

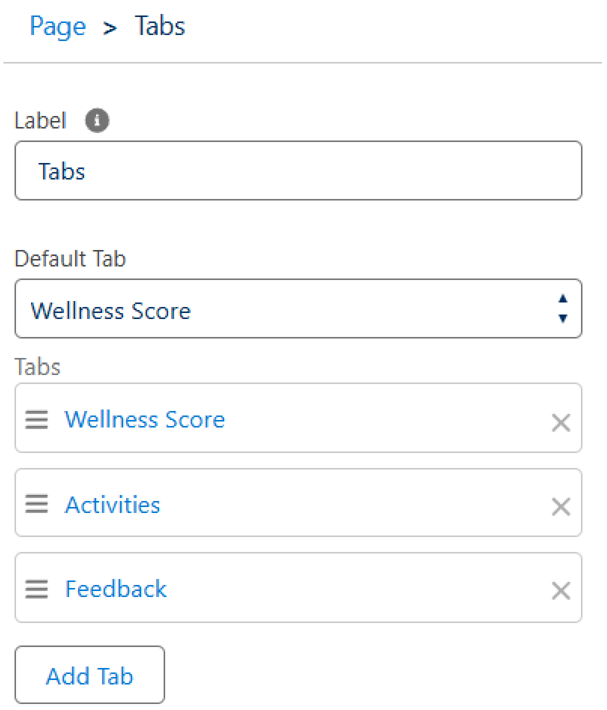
- HR wants to see an employee's **wellness trend and productivity metrics** at a glance.
- Managers want to **track activity completion and feedback**.
- Tabs provide **organized sections**, reducing clutter and making navigation intuitive.



Filters: All time • All activities • All types

No activities to show.
Get started by sending an email, scheduling a task, and more.

No past activity. Past meetings and tasks marked as done show up here.



Page > Tabs

Label

Tabs

Default Tab

Wellness Score

Tabs

- Wellness Score
- Activities
- Feedback

HOME PAGE LAYOUT

Purpose

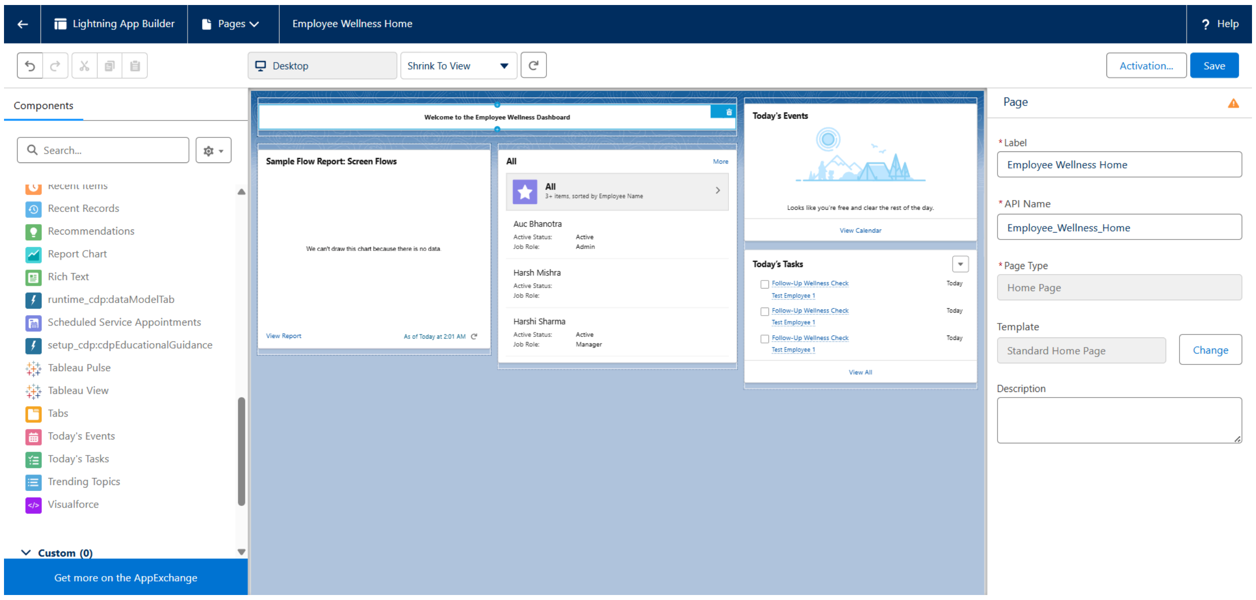
The **Home Page Layout** lets you **design a customized landing page** for your Salesforce app.

- HR and managers can **see key metrics at a glance**, like wellness scores, employee participation, and productivity tasks.
- Provides **quick access to reports, dashboards, and important actions** without navigating multiple pages.

Use Case

- HR opens Salesforce and wants to **quickly identify employees needing attention**.
- Managers want **summary charts, top performers, and upcoming tasks** immediately visible.

- Custom Home Page improves **efficiency and monitoring** for wellness and productivity tracking.



Welcome to the Employee Wellness Dashboard

Sample Flow Report: Screen Flows

We can't draw this chart because there is no data.

View Report — As of Today at 2:01 AM

All — More

All
3+ items, sorted by Employee Name

Auc Bhanotra	
Active Status:	Active
Job Role:	Admin
Harsh Mishra	
Active Status:	
Job Role:	
Harshi Sharma	
Active Status:	Active
Job Role:	Manager

Today's Events

Looks like you're free and clear the rest of the day.

View Calendar

Today's Tasks

Follow-Up Wellness Check / Test Employee 1	Today
Follow-Up Wellness Check / Test Employee 1	Today
Follow-Up Wellness Check / Test Employee 1	Today

View All

Page

* Label	Employee Wellness Home
* API Name	Employee_Wellness_Home
* Page Type	Home Page
Template	Standard Home Page (Change)
Description	

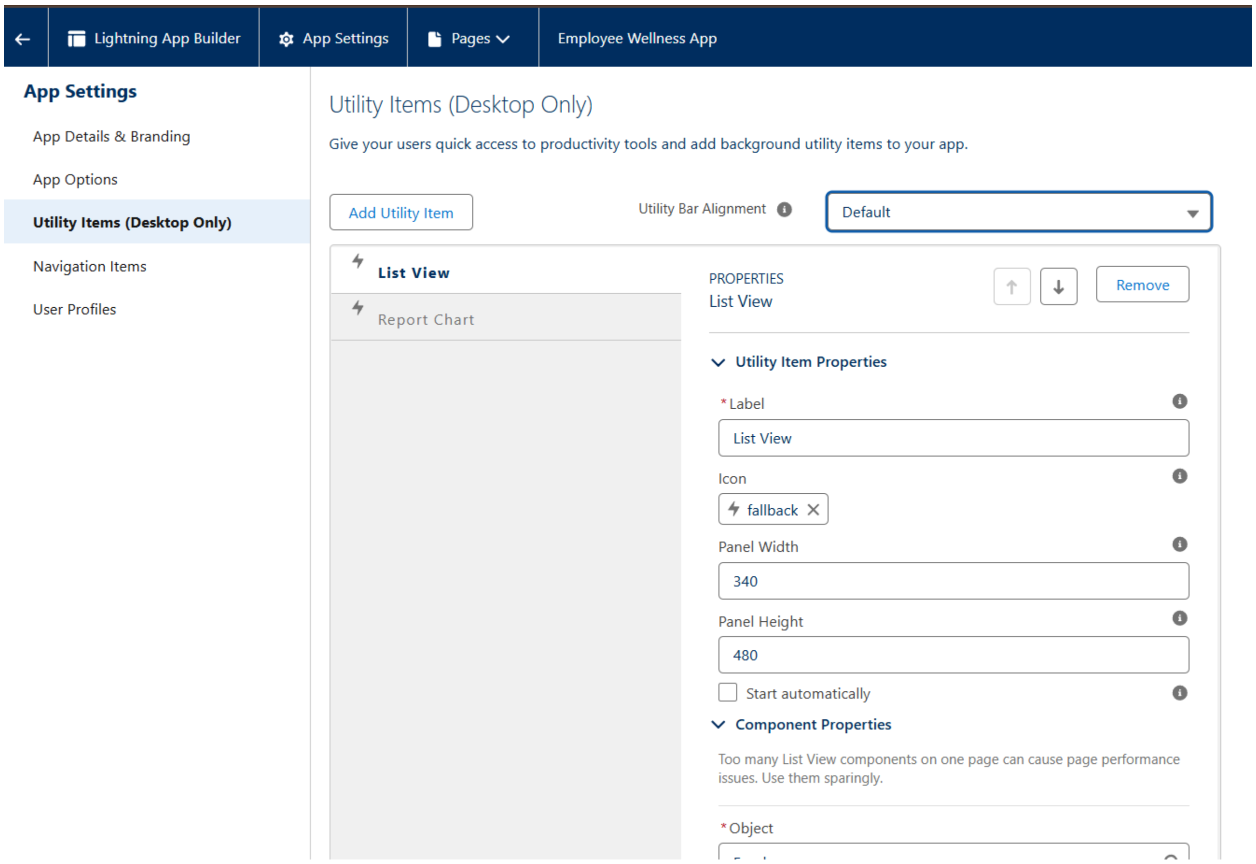
UTILITY BAR

Purpose

- Provides **easy access to frequently used components** without leaving the current page.
- Can include **list views, quick actions, reports, or LWCs**.
- Helps HR and managers **log activities, view tasks, or access key dashboards** quickly.

Use Case

- HR clicks **Utility Bar → Inactive Employees List** to see employees who haven't submitted wellness reports in 30 days.
- Manager clicks **Utility Bar → Log Wellness Activity** while viewing a record page.
- Everything happens **without leaving the current page**, saving time and improving workflow.



Utility Items (Desktop Only)

Give your users quick access to productivity tools and add background utility items to your app.

Utility Bar Alignment: Default

PROPERTIES
List View

Utility Item Properties

*Label: List View

Icon: fallback

Panel Width: 340

Panel Height: 480

Start automatically

Component Properties

Too many List View components on one page can cause page performance issues. Use them sparingly.

*Object

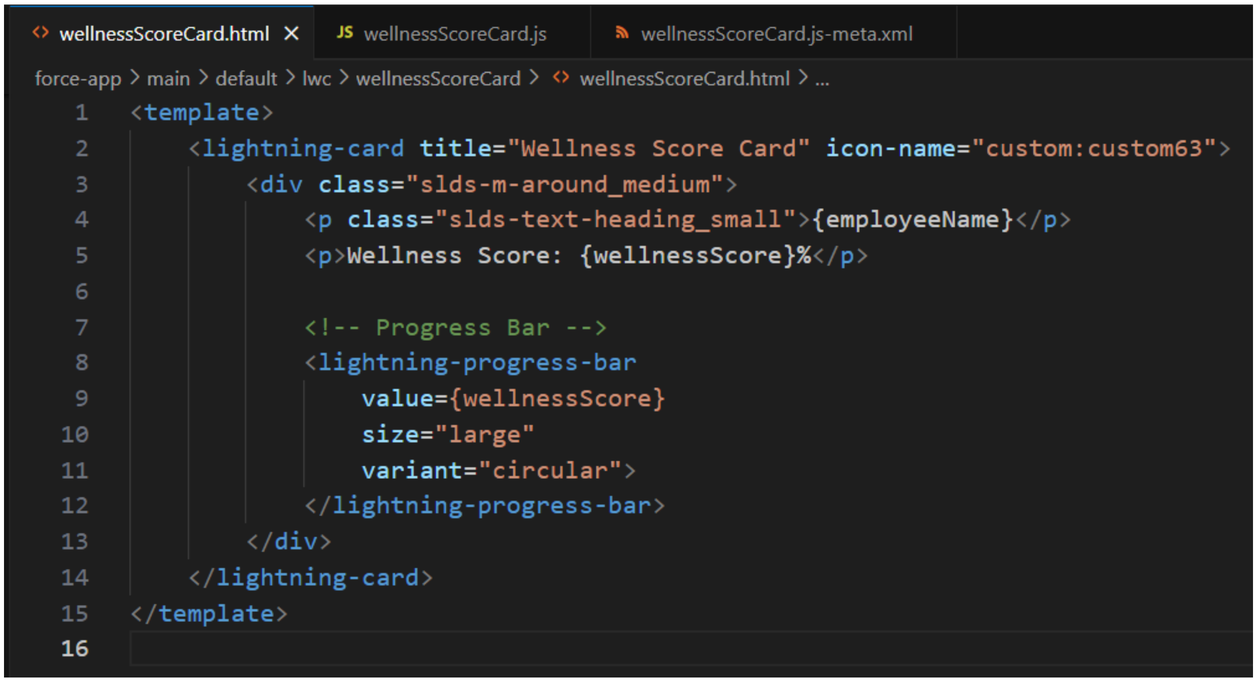
LIGHTNING WEB COMPONENTS (LWS)

Purpose

- **Custom components** that provide modern, fast, and reusable UI pieces in Salesforce.
- Used when **standard components (Reports, Tabs, Lists)** aren't enough for your project.
- LWCs let you **fetch data using Apex or Wire adapters** and display it in a clean, interactive way.

Use Cases in Project:

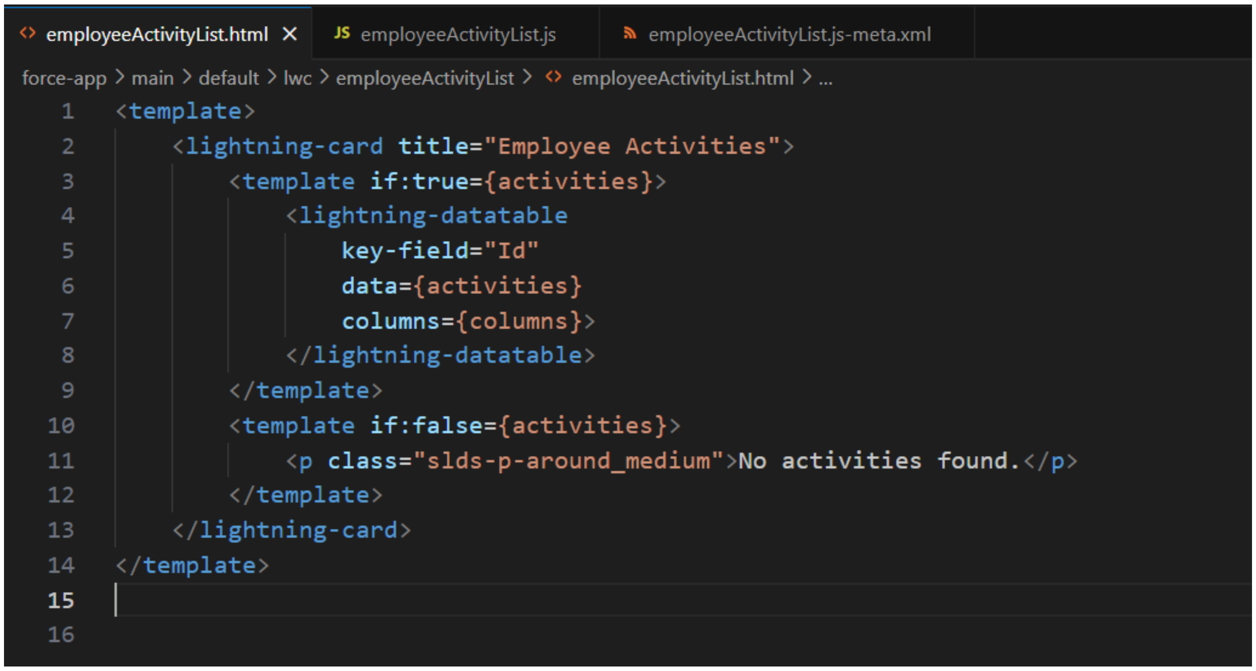
- Wellness Score Card



wellnessScoreCard.html
wellnessScoreCard.js
wellnessScoreCard.js-meta.xml
force-app > main > default > lwc > wellnessScoreCard > wellnessScoreCard.html > ...

```html
<template>
    <lightning-card title="Wellness Score Card" icon-name="custom:custom63">
        <div class="slds-m-around_medium">
            <p class="slds-text-heading_small">{employeeName}</p>
            <p>Wellness Score: {wellnessScore}%</p>

            <!-- Progress Bar -->
            <lightning-progress-bar
                value={wellnessScore}
                size="large"
                variant="circular">
            </lightning-progress-bar>
        </div>
    </lightning-card>
</template>
```

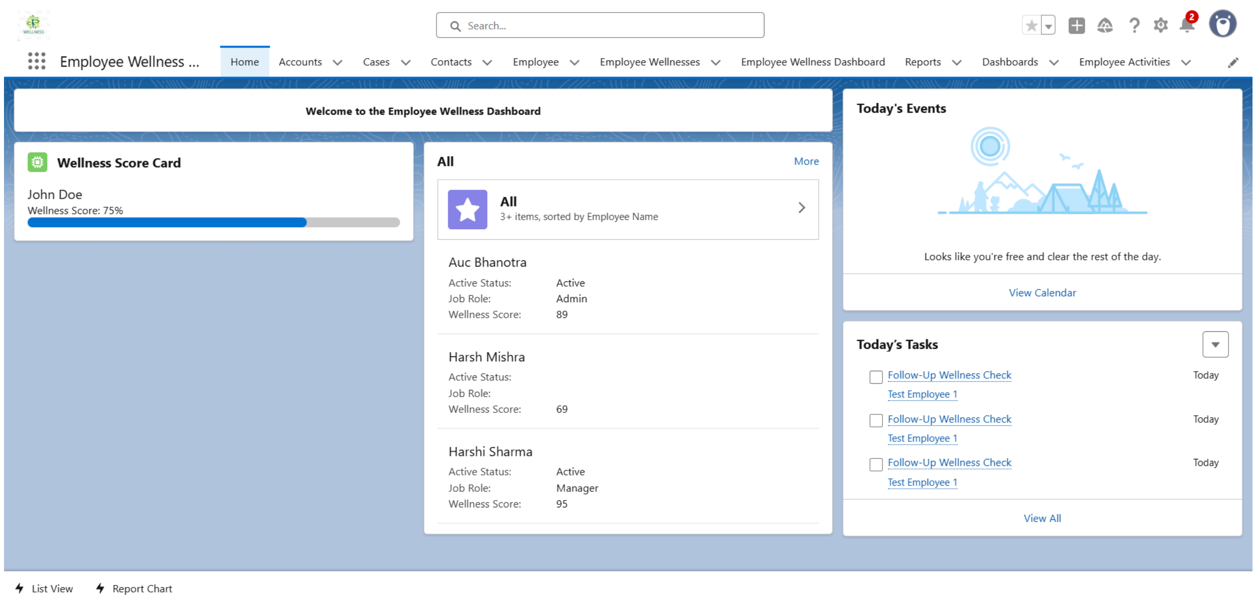
- Employee Activity List



employeeActivityList.html
employeeActivityList.js
employeeActivityList.js-meta.xml
force-app > main > default > lwc > employeeActivityList > employeeActivityList.html > ...

```html
<template>
    <lightning-card title="Employee Activities">
        <template if:true={activities}>
            <lightning-datatable
                key-field="Id"
                data={activities}
                columns={columns}>
            </lightning-datatable>
        </template>
        <template if:false={activities}>
            <p class="slds-p-around_medium">No activities found.</p>
        </template>
    </lightning-card>
</template>
```

Steps: Setup in VS Code ➔ Create LWC ➔ Write Code ➔ Deploy ➔ Add to Lightning App Builder



Welcome to the Employee Wellness Dashboard

Wellness Score Card

John Doe
Wellness Score: 75%

All — More

All
3+ items, sorted by Employee Name

Auc Bhanotra
Active Status: Active
Job Role: Admin
Wellness Score: 89

Harsh Mishra
Active Status:
Job Role:
Wellness Score: 69

Harshi Sharma
Active Status: Active
Job Role: Manager
Wellness Score: 95

Today's Events

Looks like you're free and clear the rest of the day.

View Calendar

Today's Tasks

Follow-Up Wellness Check — Test Employee 1 — Today
Follow-Up Wellness Check — Test Employee 1 — Today
Follow-Up Wellness Check — Test Employee 1 — Today

View All

List View Report Chart

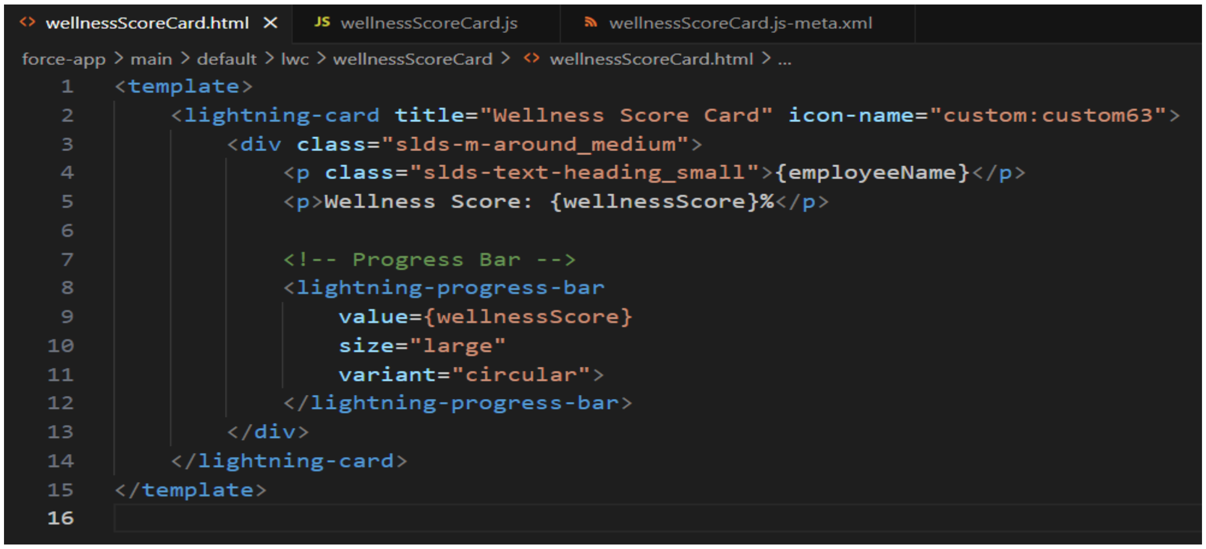
APEX WITH LWC

Purpose

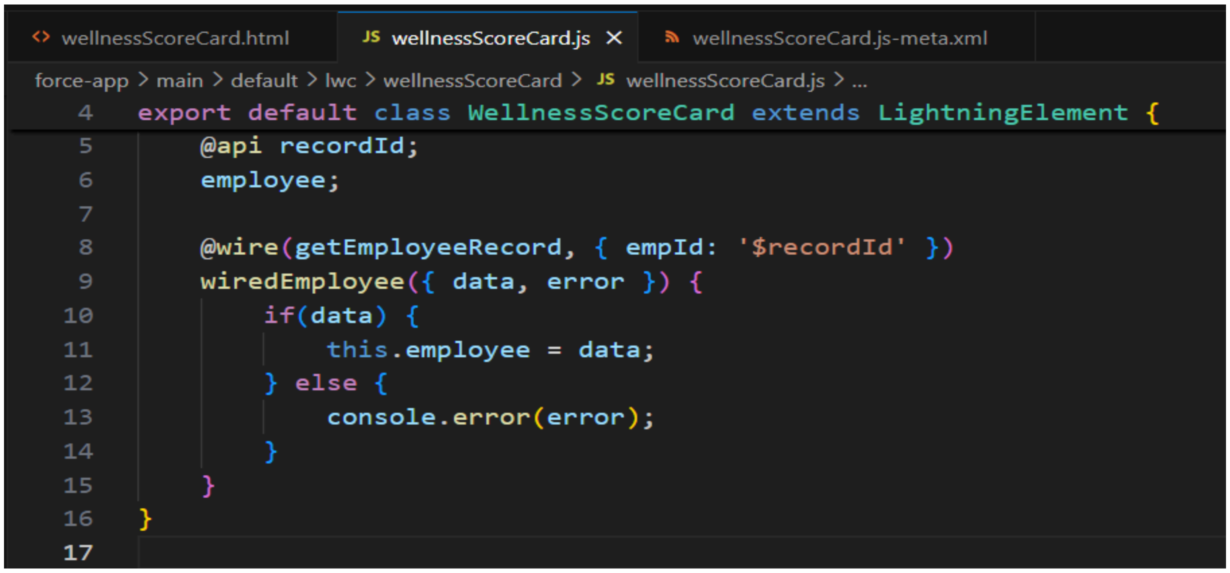
- Apex allows you to **fetch, create, update, or delete Salesforce data** from your LWC.
- You use **@AuraEnabled methods** in Apex and call them in LWC using **Wire adapters** (for reactive data) or **imperative calls** (on button click).

Use Case

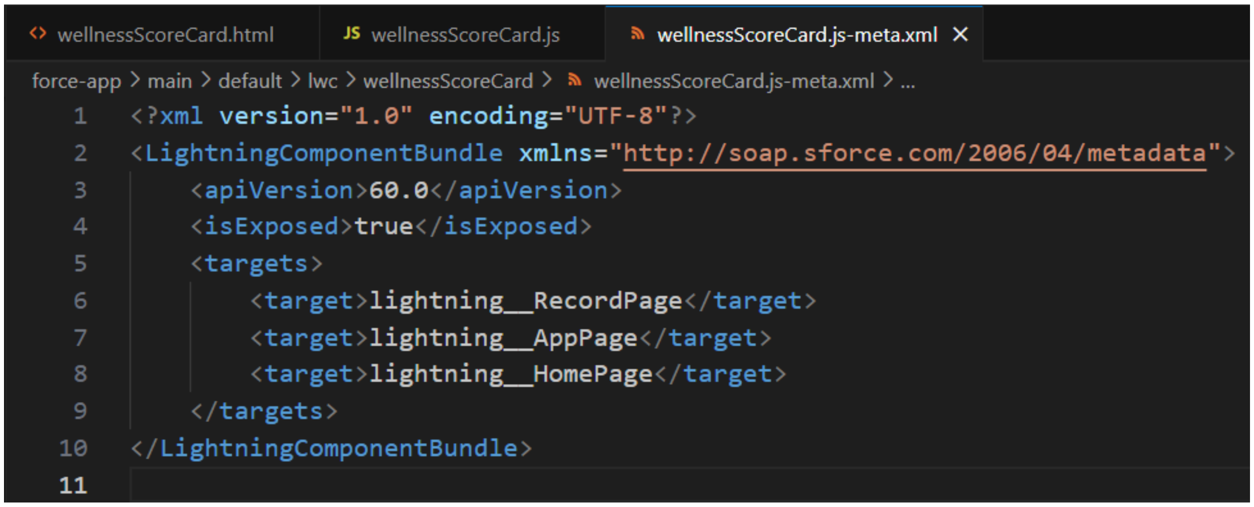
- Show **wellness score** dynamically in `WellnessScoreCard`..
- Update **wellness or productivity scores** using a button in LWC.


wellnessScoreCard.html
wellnessScoreCard.js
wellnessScoreCard.js-meta.xml
force-app > main > default > lwc > wellnessScoreCard > wellnessScoreCard.html > ...

```html
<template>
    <lightning-card title="Wellness Score Card" icon-name="custom:custom63">
        <div class="slds-m-around_medium">
            <p class="slds-text-heading_small">{employeeName}</p>
            <p>Wellness Score: {wellnessScore}%</p>

            <!-- Progress Bar -->
            <lightning-progress-bar
                value={wellnessScore}
                size="large"
                variant="circular">
            </lightning-progress-bar>
        </div>
    </lightning-card>
</template>

```


wellnessScoreCard.html
wellnessScoreCard.js
wellnessScoreCard.js-meta.xml
force-app > main > default > lwc > wellnessScoreCard > wellnessScoreCard.js > ...

```javascript
export default class WellnessScoreCard extends LightningElement {
    @api recordId;
    employee;

    @wire(getEmployeeRecord, { empId: '$recordId' })
    wiredEmployee({ data, error }) {
        if(data) {
            this.employee = data;
        } else {
            console.error(error);
        }
    }
}

```


wellnessScoreCard.html
wellnessScoreCard.js
wellnessScoreCard.js-meta.xml
force-app > main > default > lwc > wellnessScoreCard > wellnessScoreCard.js-meta.xml > ...

```xml
<?xml version="1.0" encoding="UTF-8"?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>60.0</apiVersion>
    <isExposed>true</isExposed>
    <targets>
        <target>lightning__RecordPage</target>
        <target>lightning__AppPage</target>
        <target>lightning__HomePage</target>
    </targets>
</LightningComponentBundle>

```

EVENTS IN LWC

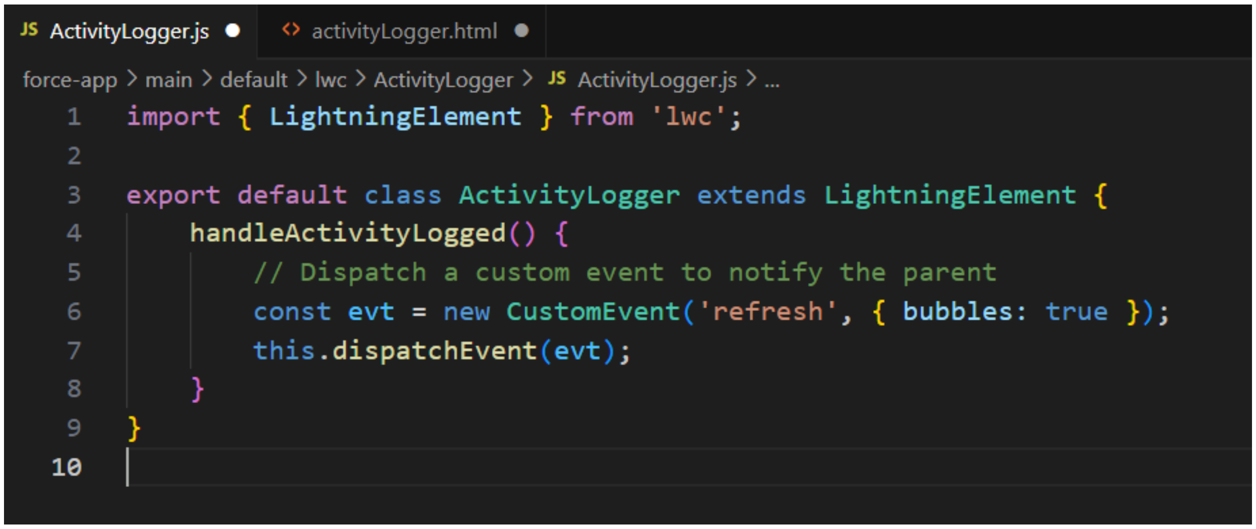
Purpose

- Events allow **communication between components**:
    - **Parent → Child**
    - **Child → Parent**
- Useful for updating UI when data changes without page reload

Use Case in Your Project

- Update **Wellness Score Card** when a manager logs a new activity.
- Refresh **Employee Activity List** after adding a new activity.

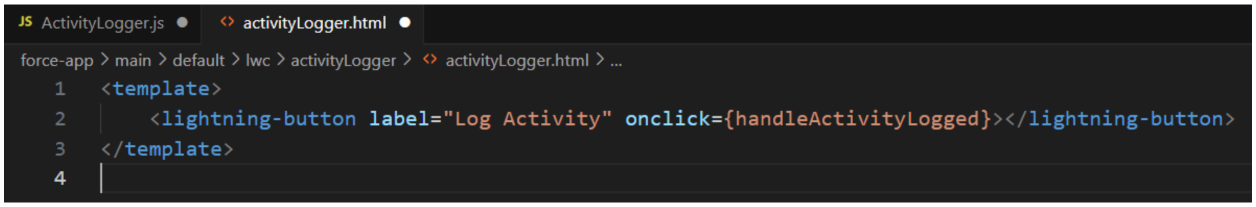
CHILD COMPONENT : - `ActivityLogger`



JS ActivityLogger.js ● | activityLogger.html ●

force-app > main > default > lwc > ActivityLogger > JS ActivityLogger.js > ...

```
import { LightningElement } from 'lwc';

export default class ActivityLogger extends LightningElement {
    handleActivityLogged() {
        // Dispatch a custom event to notify the parent
        const evt = new CustomEvent('refresh', { bubbles: true });
        this.dispatchEvent(evt);
    }
}

```

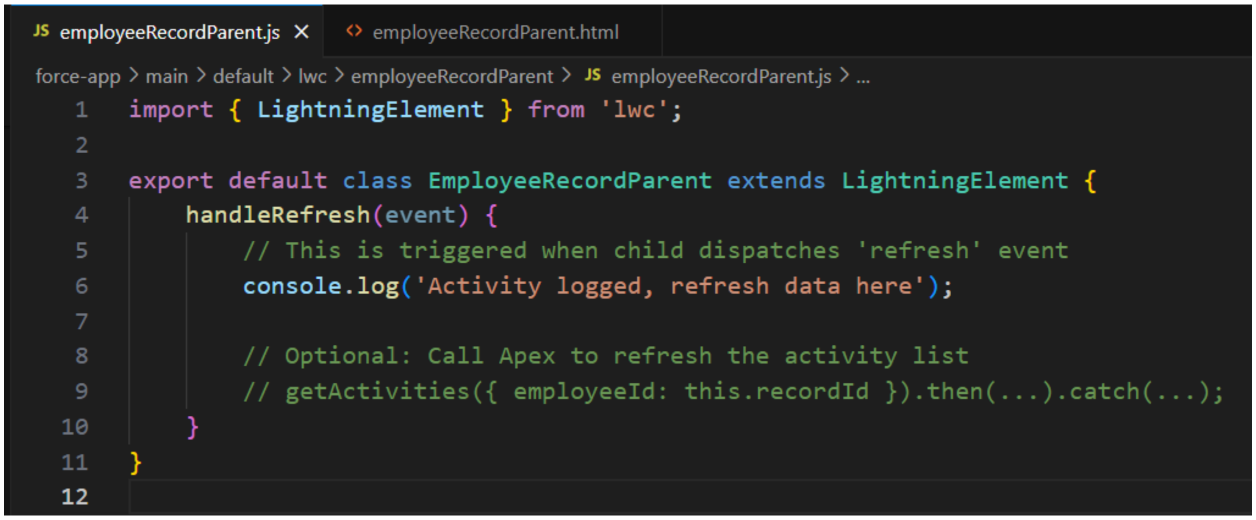


JS ActivityLogger.js ● | activityLogger.html ●

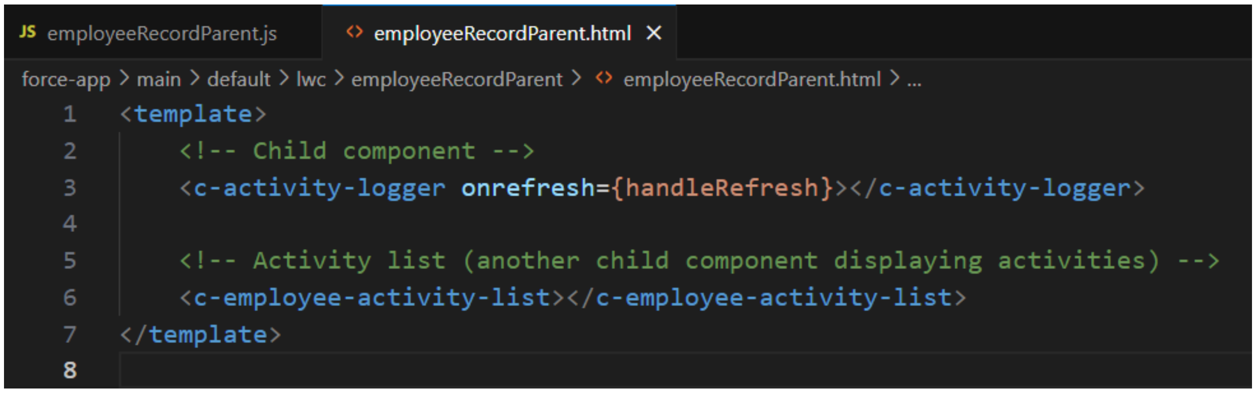
force-app > main > default > lwc > activityLogger > activityLogger.html > ...

```
<template>
    <lightning-button label="Log Activity" onclick={handleActivityLogged}></lightning-button>
</template>

```

PARENT COMPONENT : - EmployeeRecordParent


employeeRecordParent.js
employeeRecordParent.html
force-app > main > default > lwc > employeeRecordParent > employeeRecordParent.js > ...

```
import { LightningElement } from 'lwc';

export default class EmployeeRecordParent extends LightningElement {
    handleRefresh(event) {
        // This is triggered when child dispatches 'refresh' event
        console.log('Activity logged, refresh data here');

        // Optional: Call Apex to refresh the activity list
        // getActivities({ employeeId: this.recordId }).then(...).catch(...);
    }
}
```


employeeRecordParent.js
employeeRecordParent.html
force-app > main > default > lwc > employeeRecordParent > employeeRecordParent.html > ...

```
<template>
    <!-- Child component -->
    <c-activity-logger onrefresh={handleRefresh}></c-activity-logger>

    <!-- Activity list (another child component displaying activities) -->
    <c-employee-activity-list></c-employee-activity-list>
</template>
```

WIRE ADAPTERS

Purpose

- Fetch **Salesforce data reactively** in Lightning Web Components.
- Automatically updates when the record or parameters change.
- Used in the project to **display employee wellness scores** dynamically on the Employee Record Page.

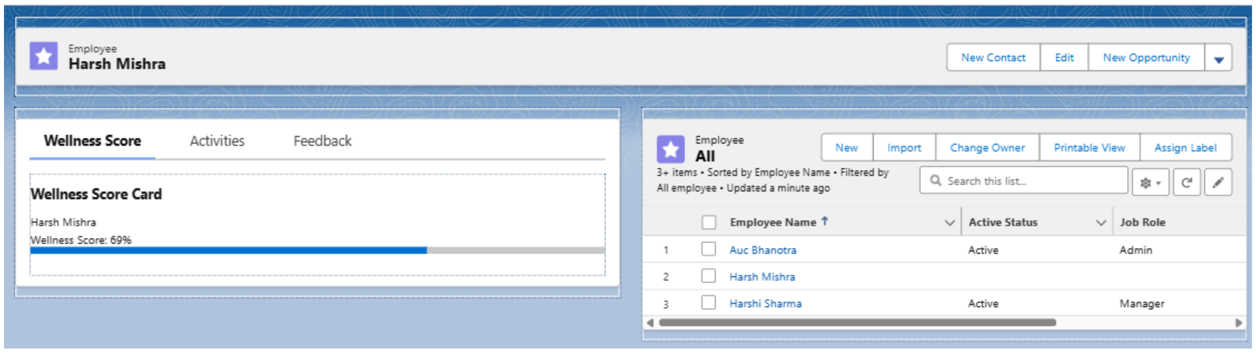
Use Case in Project

- The `WellnessScoreCard` LWC fetches data for a specific employee using the **EmployeeController Apex class**.
- Displays the **employee name** and **wellness score** as a progress bar.

- Updates automatically if the record changes or the page reloads

Steps Performed

1. Created **Apex class `EmployeeController`** with `@AuraEnabled(cacheable=true)` method `getEmployeeRecord`.
2. Created **LWC `WellnessScoreCard`** with JS, HTML, and meta XML files.
3. Used **@wire** in JS to call `getEmployeeRecord` reactively.
4. Displayed the employee data in a **progress bar**.
5. Added LWC to **Employee Record Page** using Lightning App Builder.



Employee
Harsh Mishra

Wellness Score Card
Harsh Mishra
Wellness Score: 69%

Employee
All
3+ items • Sorted by Employee Name • Filtered by All employee • Updated a minute ago

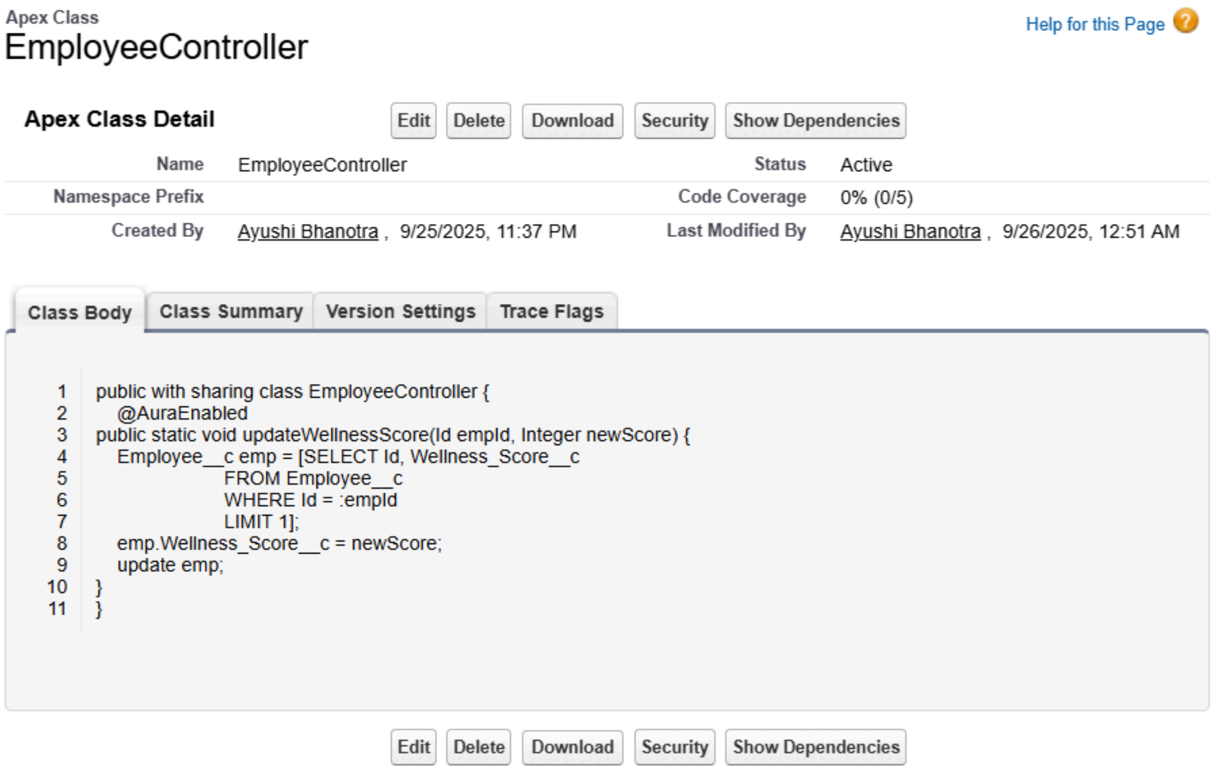
	Employee Name	Active Status	Job Role
1	Auc Bhanotra	Active	Admin
2	Harsh Mishra		
3	Harshi Sharma	Active	Manager

IMPERATIVE APEX CALLS

Purpose

- Call Apex **on demand**, triggered by user actions such as a **button click**.
- Unlike Wire adapters, this method is **not reactive** and runs only when invoked.
- Used in the project to **update employee wellness scores** manually.

Use Case: A manager clicks a button to update an employee's wellness score



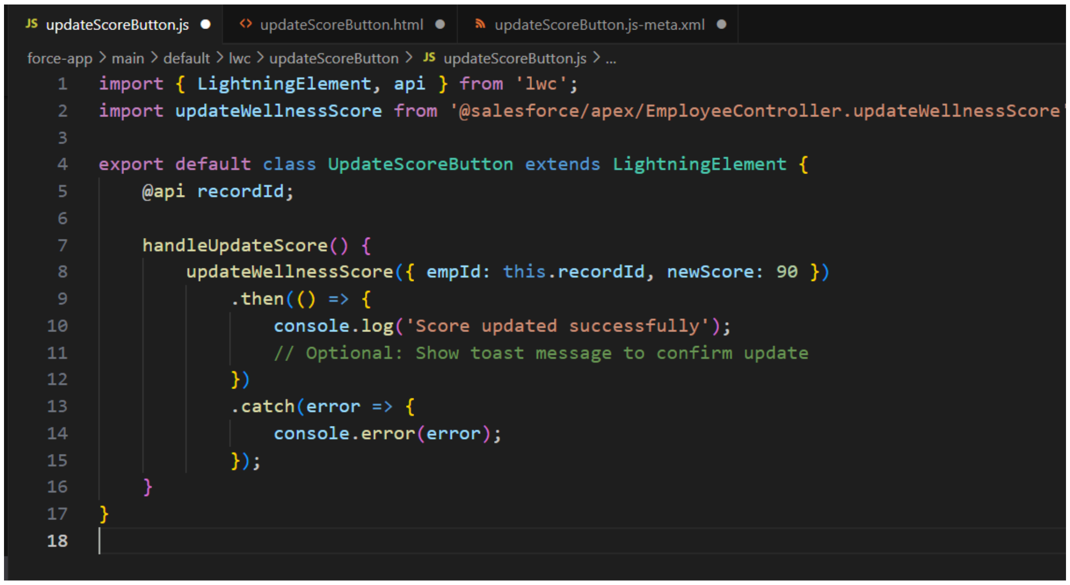
EmployeeController

Apex Class Detail

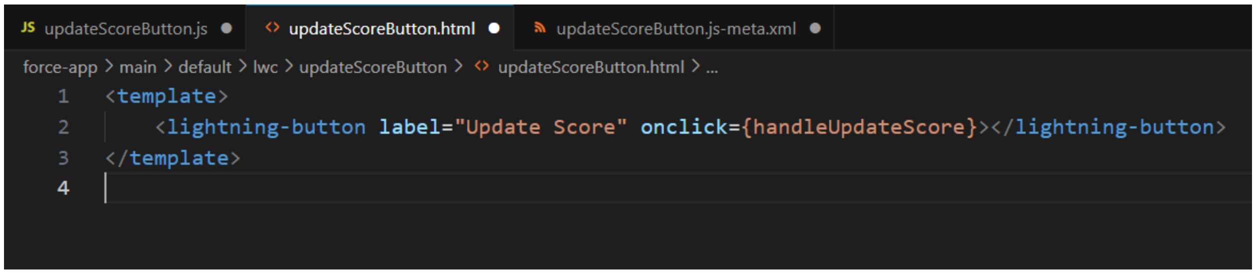
Name	EmployeeController	Status	Active
Namespace Prefix		Code Coverage	0% (0/5)
Created By	Ayushi Bhanotra , 9/25/2025, 11:37 PM	Last Modified By	Ayushi Bhanotra , 9/26/2025, 12:51 AM

```
public with sharing class EmployeeController {
    @AuraEnabled
public static void updateWellnessScore(Id empId, Integer newScore) {
    Employee__c emp = [SELECT Id, Wellness_Score__c
                      FROM Employee__c
                      WHERE Id = :empId
                      LIMIT 1];
    emp.Wellness_Score__c = newScore;
    update emp;
}
}
```

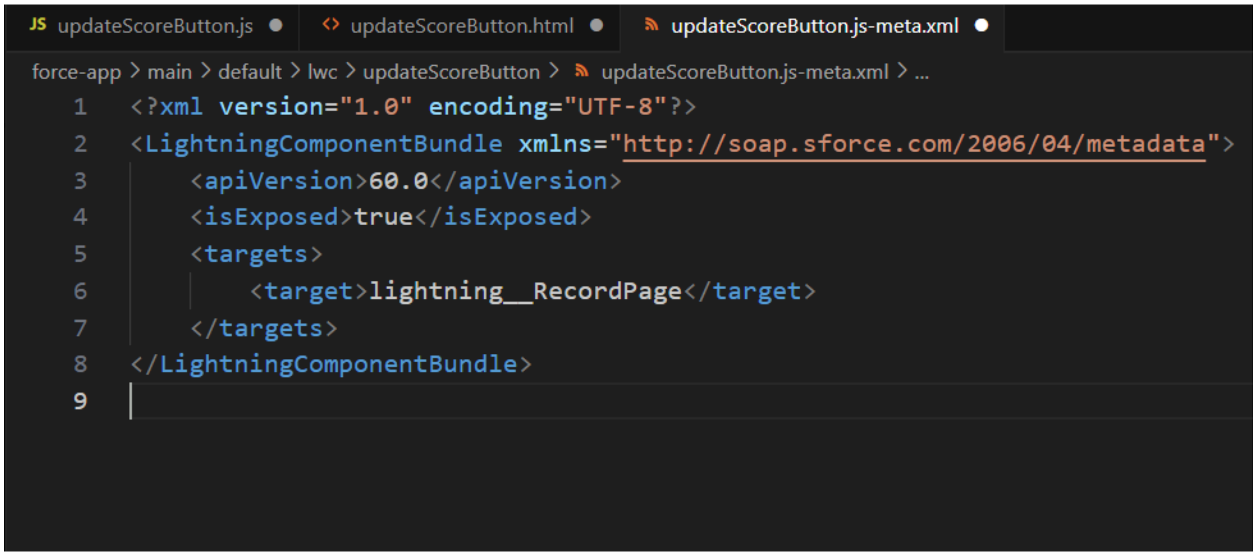
CODE SNIPPETS :


updateScoreButton.js
updateScoreButton.html
updateScoreButton.js-meta.xml
force-app > main > default > lwc > updateScoreButton > updateScoreButton.js > ...

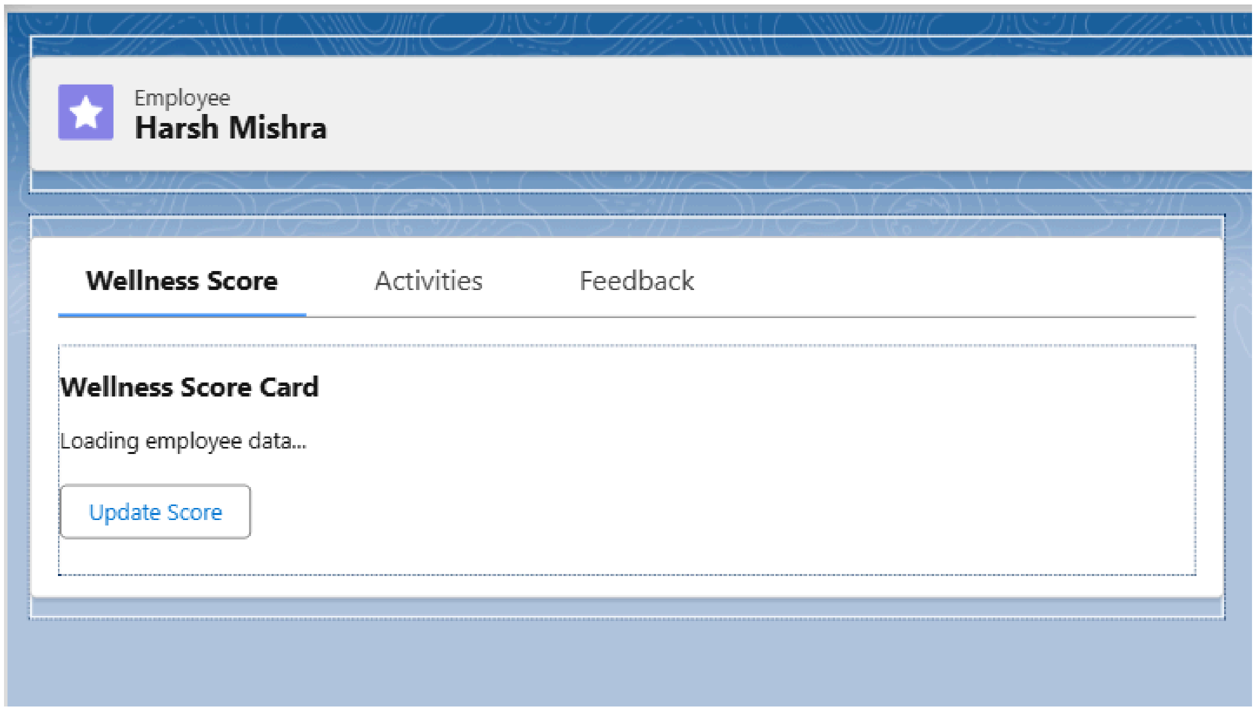
```javascript
import { LightningElement, api } from 'lwc';
import updateWellnessScore from '@salesforce/apex/EmployeeController.updateWellnessScore

export default class UpdateScoreButton extends LightningElement {
    @api recordId;

    handleUpdateScore() {
        updateWellnessScore({ empId: this.recordId, newScore: 90 })
            .then(() => {
                console.log('Score updated successfully');
                // Optional: Show toast message to confirm update
            })
            .catch(error => {
                console.error(error);
            });
    }
}
```


updateScoreButton.js
updateScoreButton.html
updateScoreButton.js-meta.xml
force-app > main > default > lwc > updateScoreButton > updateScoreButton.html > ...

```html
<template>
    <lightning-button label="Update Score" onclick={handleUpdateScore}></lightning-button>
</template>
```


updateScoreButton.js
updateScoreButton.html
updateScoreButton.js-meta.xml
force-app > main > default > lwc > updateScoreButton > updateScoreButton.js-meta.xml > ...

```xml
<?xml version="1.0" encoding="UTF-8"?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>60.0</apiVersion>
    <isExposed>true</isExposed>
    <targets>
        <target>lightning__RecordPage</target>
    </targets>
</LightningComponentBundle>
```



Wellness Score Card

Loading employee data...

NAVIGATION SERVICE

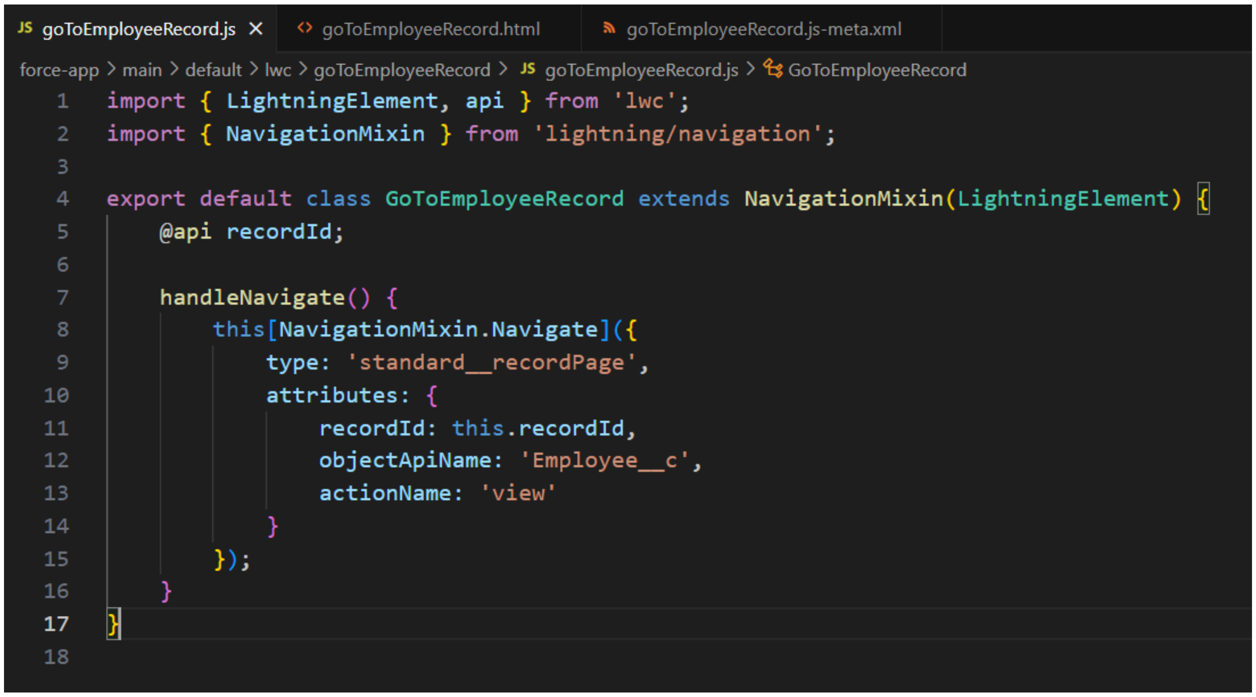
Purpose

- Programmatically navigate to **record pages, list views, dashboards, or external URLs** from a Lightning Web Component.
- Provides **better user experience** by redirecting users after an action, e.g., clicking a button.
- Used in the project to **navigate to an Employee Record Page** directly from a button on a dashboard or activity list.

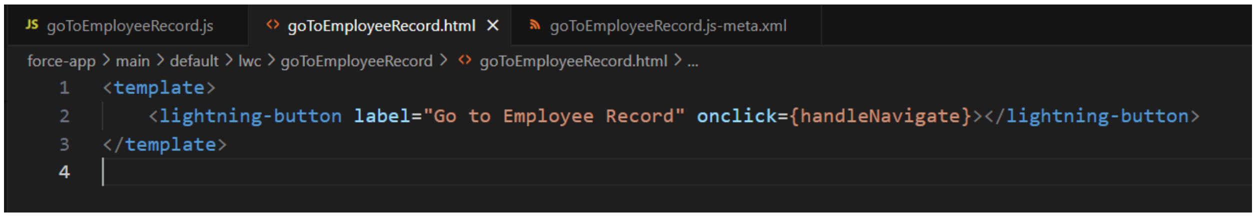
Use Case in Project

- A manager clicks **"Go to Employee Record"** from a dashboard or list.
- The LWC uses **Navigation Service** to take the user directly to the selected employee's record page.

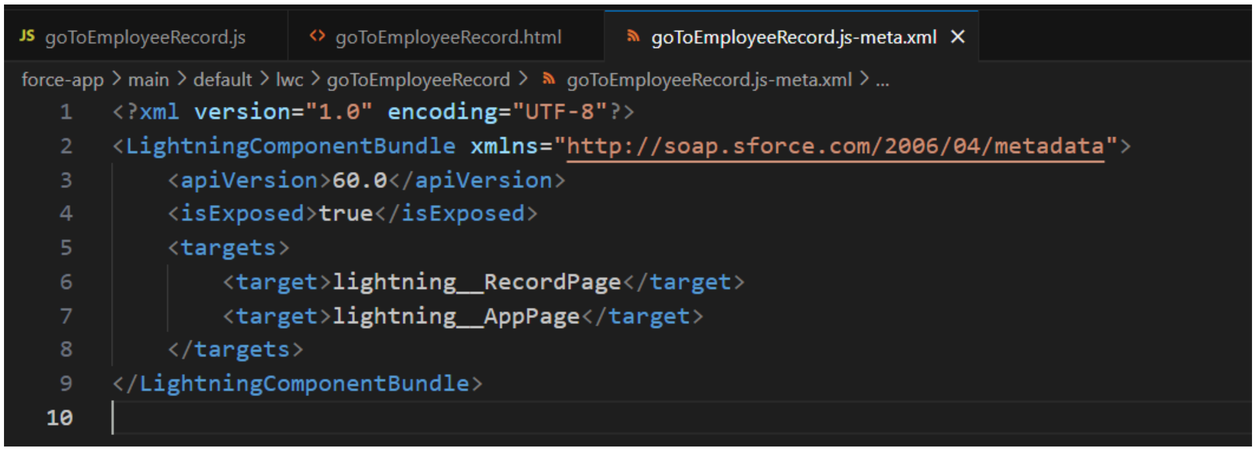
CODE SNIPPETS :


goToEmployeeRecord.js
goToEmployeeRecord.html
goToEmployeeRecord.js-meta.xml
force-app > main > default > lwc > goToEmployeeRecord > goToEmployeeRecord.js > GoToEmployeeRecord

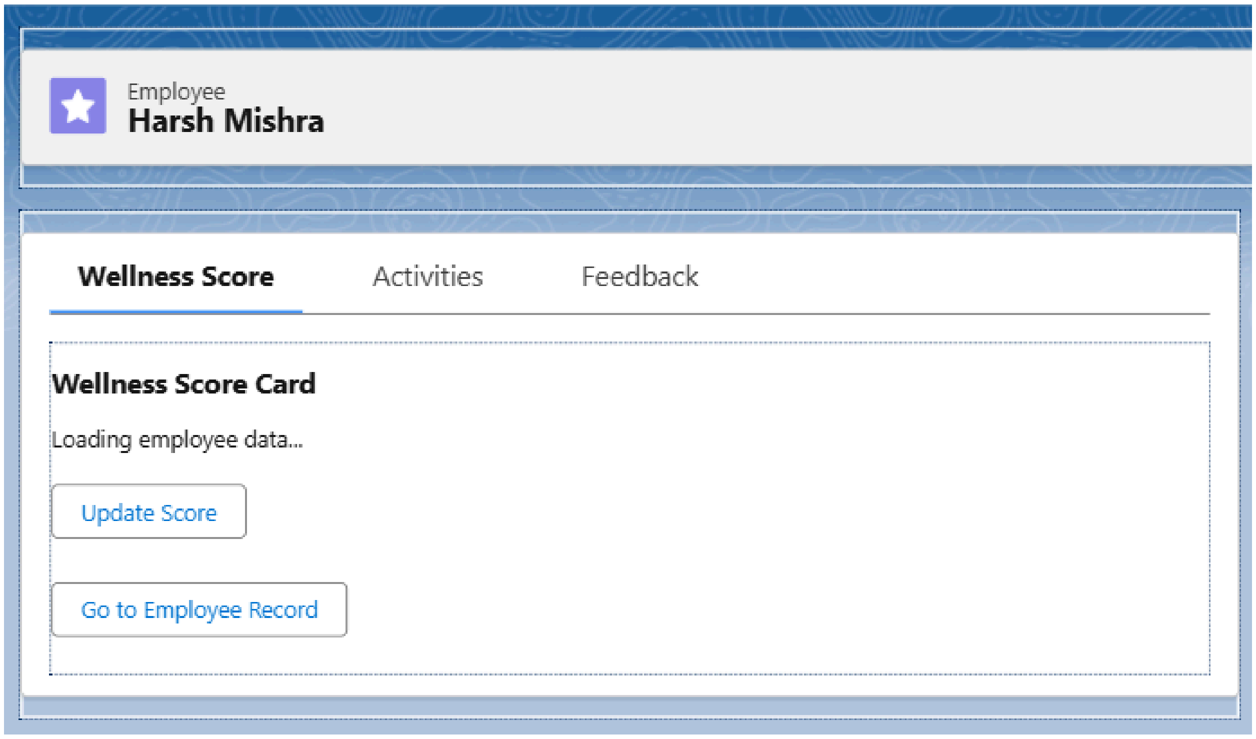
```javascript
import { LightningElement, api } from 'lwc';
import { NavigationMixin } from 'lightning/navigation';

export default class GoToEmployeeRecord extends NavigationMixin(LightningElement) {
    @api recordId;

    handleNavigate() {
        this[NavigationMixin.Navigate]({
            type: 'standard__recordPage',
            attributes: {
                recordId: this.recordId,
                objectApiName: 'Employee__c',
                actionName: 'view'
            }
        });
    }
}
```


goToEmployeeRecord.js
goToEmployeeRecord.html
goToEmployeeRecord.js-meta.xml
force-app > main > default > lwc > goToEmployeeRecord > goToEmployeeRecord.html > ...

```html
<template>
    <lightning-button label="Go to Employee Record" onclick={handleNavigate}></lightning-button>
</template>

```


goToEmployeeRecord.js
goToEmployeeRecord.html
goToEmployeeRecord.js-meta.xml
force-app > main > default > lwc > goToEmployeeRecord > goToEmployeeRecord.js-meta.xml > ...

```xml
<?xml version="1.0" encoding="UTF-8"?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>60.0</apiVersion>
    <isExposed>true</isExposed>
    <targets>
        <target>lightning__RecordPage</target>
        <target>lightning__AppPage</target>
    </targets>
</LightningComponentBundle>

```



Wellness Score Card

Loading employee data...