Phase 6: User Interface Development

Introduction

The user interface (UI) is the face of the Salesforce platform for end-users. While Apex powers the backend logic, a well-designed UI ensures usability, accessibility, and efficiency in day-to-day operations. Salesforce provides multiple UI technologies, ranging from **legacy Visualforce pages** to modern **Lightning Web Components (LWC)**, supported by the **Salesforce Lightning Design System (SLDS)**.

Phase 6 of the ApexHub project focuses on building interactive and visually consistent UIs. The goal is to deliver components and pages that enhance user productivity, integrate seamlessly with backend Apex logic, and follow Salesforce's latest design standards.

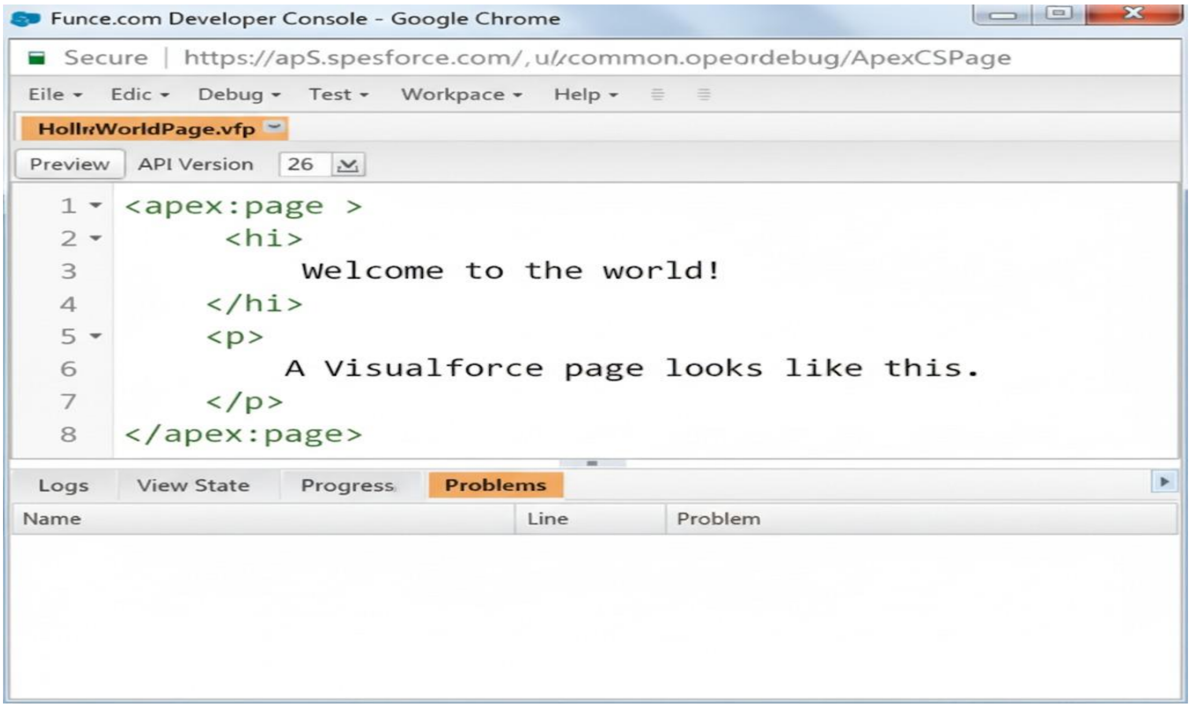
Objectives

- Build engaging UIs using Visualforce, Aura, and Lightning Web Components.
- Implement reusable and modular UI components.
- Use **events** in LWC for parent-child communication.
- Ensure consistent styling using **SLDS**.
- Integrate UI with Apex to fetch and display dynamic data.
- Provide user-friendly documentation and mockups for future enhancements.

Activities

1. Visualforce Pages

- **Purpose:** Create custom pages where standard Salesforce UI is not sufficient.
- **Example:** Generated **PDF invoices** for Opportunities.
- **Implementation:** Designed a Visualforce page with embedded Apex controller to format and export invoices.



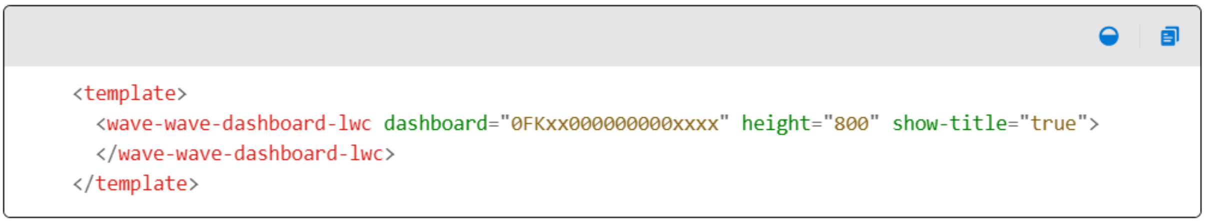
```
<apex:page >
     <hi>
         Welcome to the world!
    </hi>
    <p>
        A Visualforce page looks like this.
    </p>
</apex:page>
```

2. Aura Components

- **Purpose:** Introduce modular, reusable UI blocks with two-way data binding.
- **Example:** A **quick action Aura component** for logging calls or creating tasks directly from a record page.
- **Implementation:** Used Aura events for communication and deployed components on Lightning record pages.

3. Lightning Web Components (LWC)

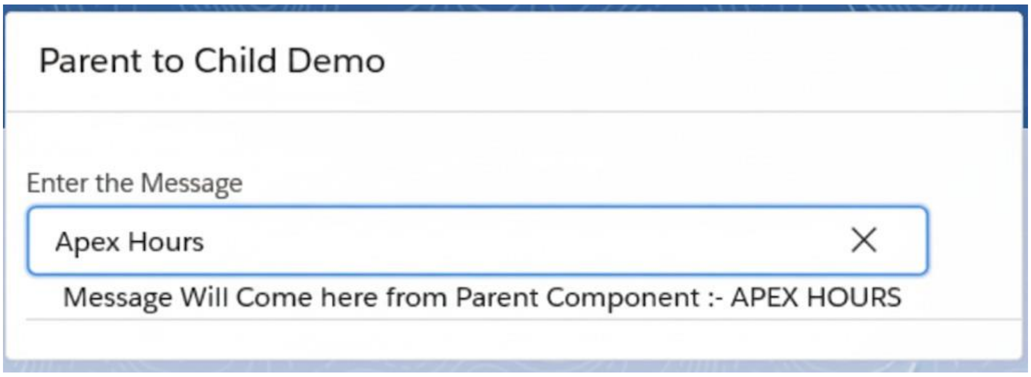
- **Purpose:** Build modern, fast, and reusable UI elements using JavaScript and HTML.
- **Example:** An **interactive dashboard** showing pipeline status of Leads and Opportunities in real time.
- **Implementation:**
    - Created parent-child LWCs for dashboard filters and charts.
    - Used Apex classes to fetch live data.
    - Leveraged SLDS for styling consistency.



```
<template>
  <wave-wave-dashboard-lwc dashboard="0FKxx000000000xxxx" height="800" show-title="true">
  </wave-wave-dashboard-lwc>
</template>
```

4. Events in LWC

- **Purpose:** Enable communication between parent and child components.
- **Examples:**
  - Parent → Child: Passing account details from dashboard to detail view.
  - Child → Parent: Sending filter criteria from filter component to parent dashboard.



Parent to Child Demo

Enter the Message

Apex Hours

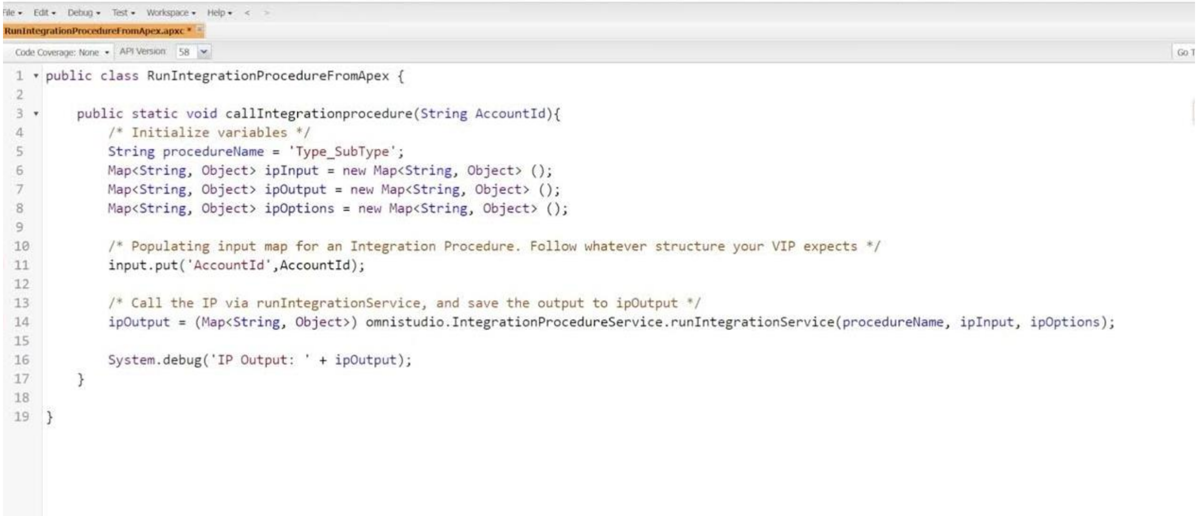
Message Will Come here from Parent Component :- APEX HOURS

5. Salesforce Lightning Design System (SLDS)

- **Purpose:** Ensure consistency with Salesforce's native UI.
- **Example:** Styled buttons, forms, and modals with SLDS grid system and CSS classes.
- **Outcome:** Unified look-and-feel with Salesforce Lightning Experience.

6. Integration with Apex

- **Purpose:** Display dynamic business data inside UI components.
- **Example:** Fetch and display **top 5 Accounts** by revenue using SOQL queries inside Apex, exposed to LWC.
- **Implementation:**
  - Created Apex controller methods (@AuraEnabled).
  - Retrieved data using SOQL.
  - Displayed data dynamically on LWC charts.



```
public class RunIntegrationProcedureFromApex {

    public static void callIntegrationprocedure(String AccountId){
        /* Initialize variables */
        String procedureName = 'Type_SubType';
        Map<String, Object> ipInput = new Map<String, Object> ();
        Map<String, Object> ipOutput = new Map<String, Object> ();
        Map<String, Object> ipOptions = new Map<String, Object> ();

        /* Populating input map for an Integration Procedure. Follow whatever structure your VIP expects */
        input.put('AccountId',AccountId);

        /* Call the IP via runIntegrationService, and save the output to ipOutput */
        ipOutput = (Map<String, Object>) omnistudio.IntegrationProcedureService.runIntegrationService(procedureName, ipInput, ipOptions);

        System.debug('IP Output: ' + ipOutput);
    }

}
```

Deliverables

- Visualforce Invoice Page (export as PDF).
- Aura Component for quick actions.
- Lightning Web Component: Interactive Lead & Opportunity Dashboard.
- Event-driven LWC communication (parent-child).
- SLDS-styled UI pages and forms.
- Documentation with UI mockups for reference.

Expected Outcomes

- End-users interact with modern, responsive UIs.
- Improved productivity with reusable and modular UI components.
- Consistent styling and branding across applications using SLDS.
- Real-time data displayed on dashboards powered by Apex.
- UI layer fully integrated with backend logic to support project scalability.