

# ApexHub – Salesforce Apex Code Repository and Automation Platform

## Objective:

ApexHub is a **custom Salesforce application** designed to help developers **store, manage, and reuse Apex code snippets (called Recipes)**, visualize data through a **custom dashboard**, and **demonstrate automation** using Apex triggers. This project improves **development efficiency**, **reduces repetitive coding**, and provides **real-time insights** for project managers and developers.

## Technologies Used:

- **Salesforce Lightning Platform**
- **Lightning Web Components (LWC)** for interactive UI
- **Apex Classes & Triggers** for backend logic
- **Lightning App Builder** for dashboard and page customization
- **Reports & Dashboards** for analytics
- **Picklists, Custom Objects, and Rich Text components** for structured and visual data

## Phase 1: App Creation & Setup

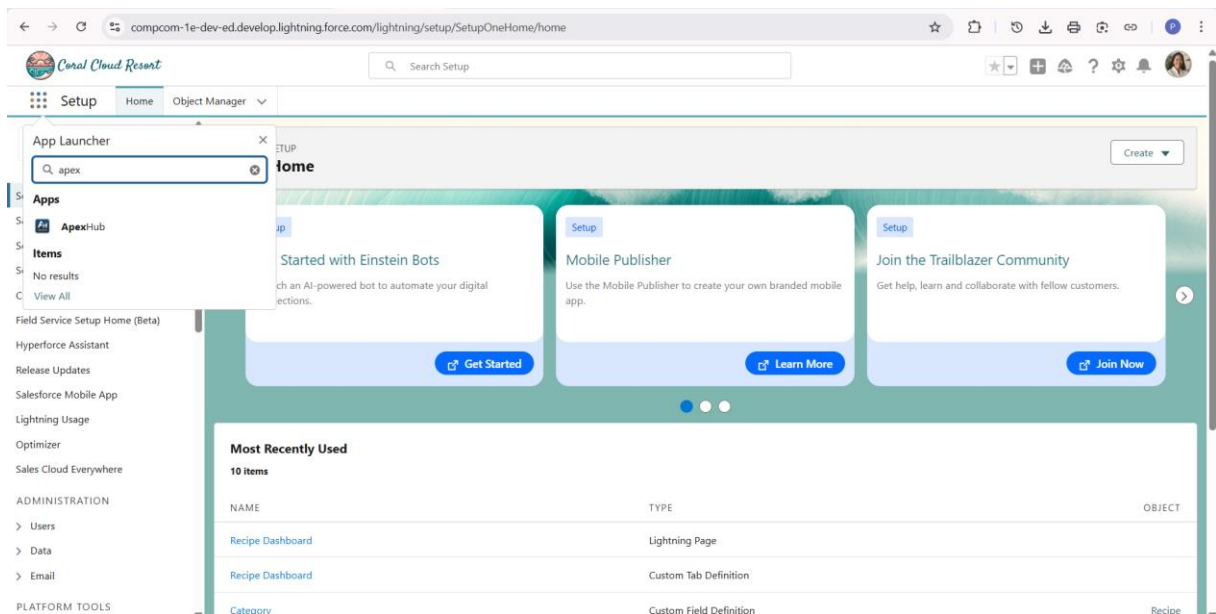
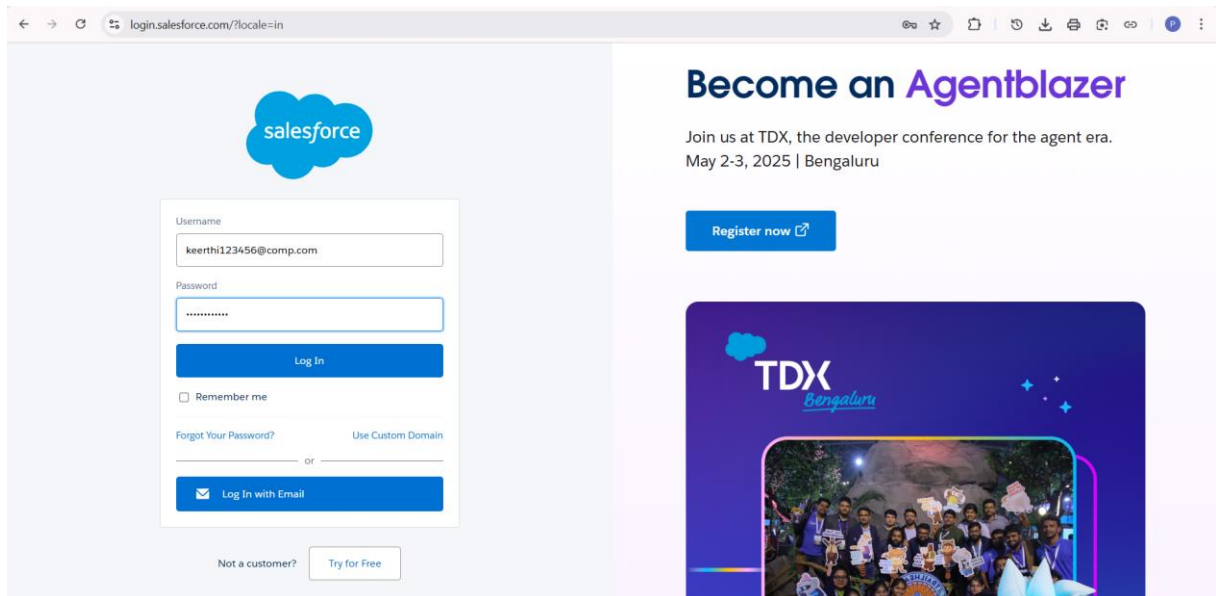
**Objective:** Establish the ApexHub Lightning App to serve as a centralized hub for code snippets.

### Details:

- Created a **Lightning App named ApexHub** using Salesforce App Builder
- Selected **Standard Navigation** for desktop and mobile form factors
- Added essential navigation items like **Recipes, Dashboard, Reports**
- Assigned to **System Administrator** profile for initial access
- Enabled **App Branding** with ApexHub logo and colors

### Importance:

This phase establishes the **foundation** for the application, ensuring all future objects, pages, and components are accessible under a unified interface.



## Phase 2: Recipes Object Creation

**Objective:** Create a structured repository for reusable Apex code snippets.

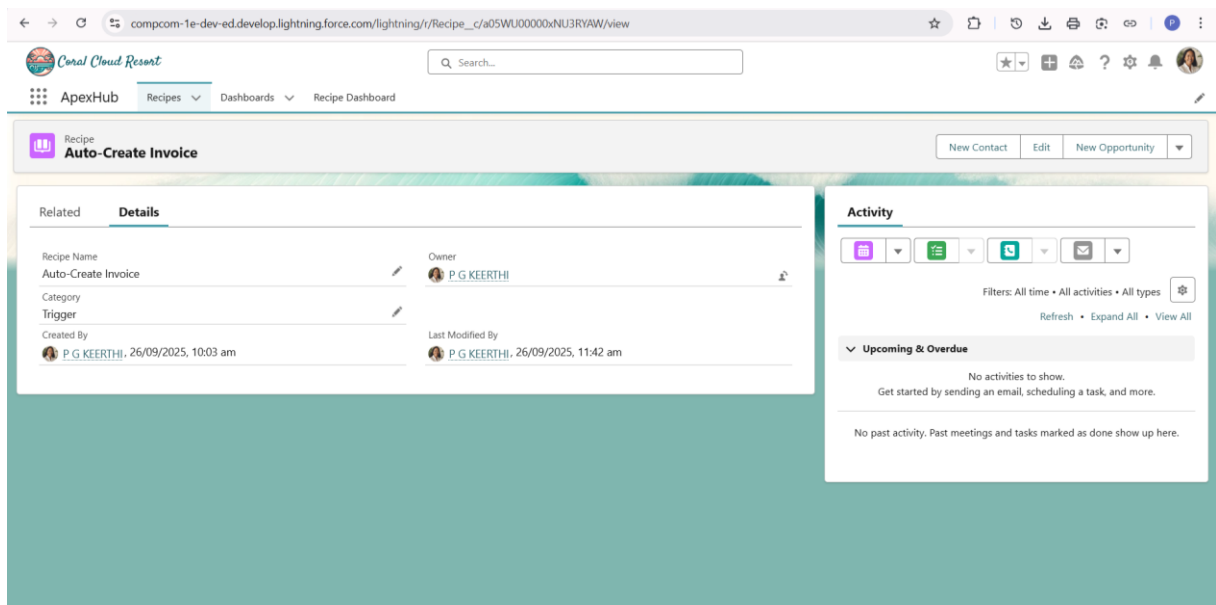
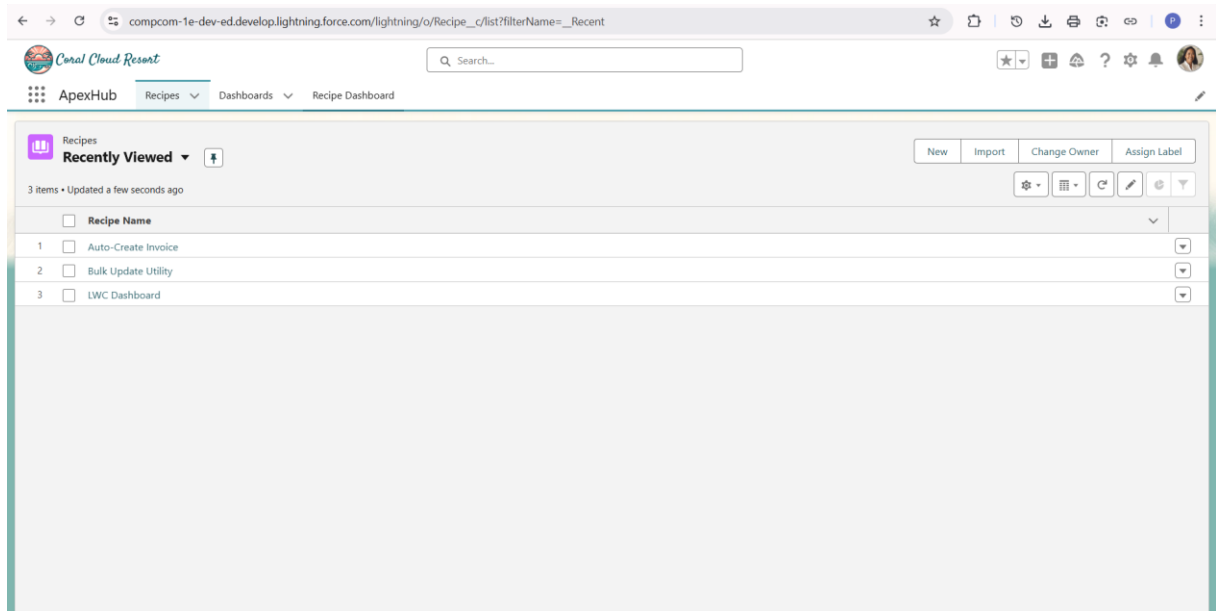
**Details:**

- Created a **custom object "Recipes"** with fields:
  - Recipe Name** (Text)
  - Description** (Long Text)
  - Category** (Picklist: Trigger, LWC, Class, Utility)
- Enabled **custom report type** on Recipes for analytics
- Ensured all new Recipes can be categorized and tracked

- Added **help text and descriptions** to guide users

### Importance:

This object acts as the **core data model**, enabling the app to store, organize, and retrieve Apex code efficiently.



## Phase 3: Adding Categories

**Objective:** Classify Recipes for easier management, filtering, and reporting.

**Details:**

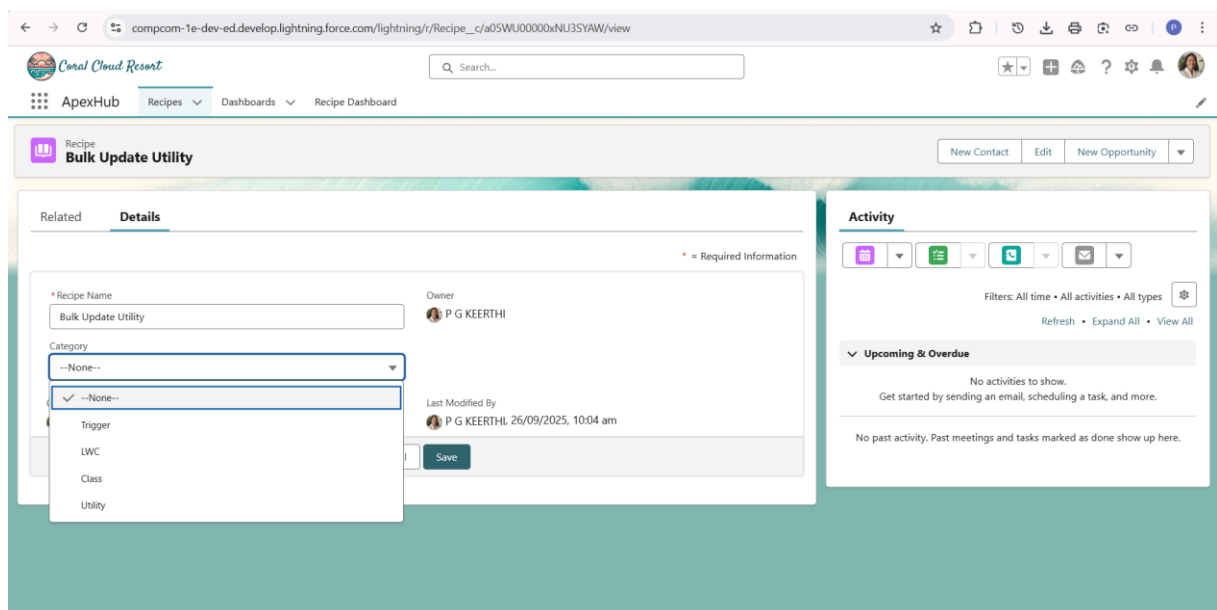
- Created a **Category picklist** with values: Trigger, LWC, Class, Utility
- Set **default value** to Trigger for quick creation
- Enabled **restriction to defined values** to maintain consistency
- Categorized all sample recipes for demonstration purposes

**Importance:**

Categorization allows developers to **quickly locate specific types of code**, and ensures analytics dashboards display meaningful groupings.

**Suggested Screenshot:**

- Recipe record with **Category field selected**



## Phase 4: Recipe Dashboard Setup

**Objective:** Provide an interactive summary of recipe activity and counts.

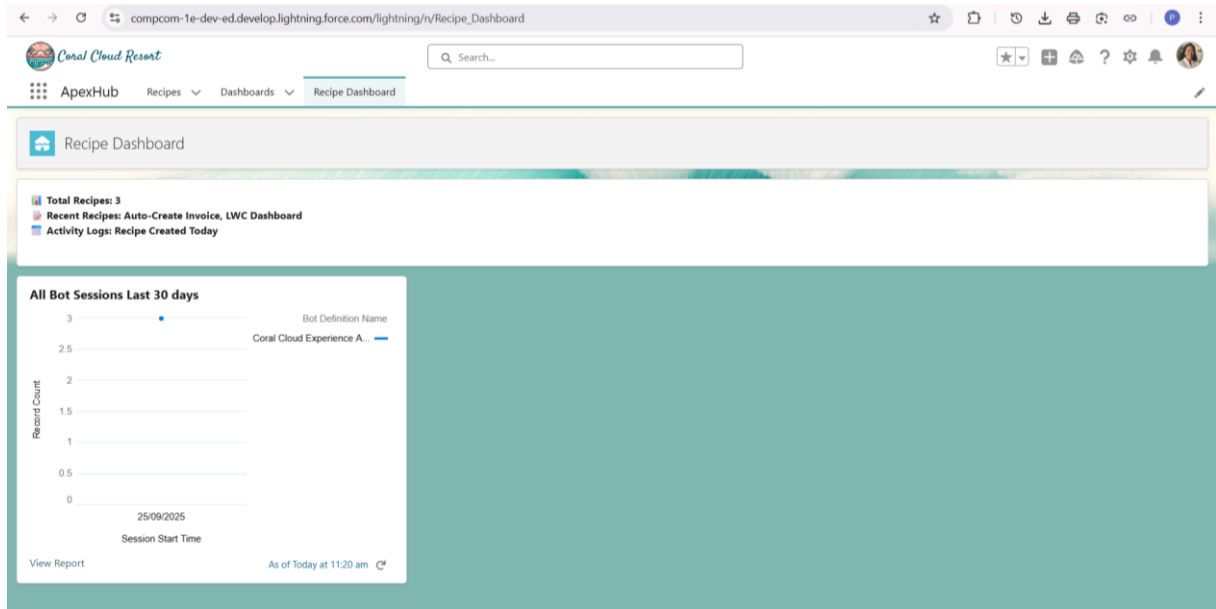
**Details:**

- Created a **custom Lightning Page “Recipe Dashboard”** using Lightning App Builder
- Added **Rich Text cards:**
  - Total Recipes: Shows number of recipes (fake number acceptable for demo)
  - Recent Recipes: Displays last 2–3 added recipes
  - Activity Logs: Shows recent actions like “Recipe Created Today”

- (Optional) Added **Report Chart** showing counts by Category

#### Importance:

Dashboard gives users a **visual overview of the repository**, helping track new additions and activity trends.



## Phase 5: Report Creation

**Objective:** Enable insights into Recipe usage through Salesforce Reports.

#### Details:

- Created **Recipe Usage Report** in Summary format
- Grouped recipes by **Category**
- Displayed **unique counts per category**
- Enabled **filters** to focus on recent or active recipes

#### Importance:

Reports provide **analytical insights**, supporting management decisions and tracking developer usage trends.

#### Suggested Screenshot:

- Report builder showing grouped recipes by Category
- Preview showing counts per category



The screenshot shows the Salesforce Report Builder interface for a report titled 'Recipe Usage Report'. The report is currently in 'Preview' mode, displaying a limited number of records. The interface includes a sidebar with 'Fields' (Groups, Columns) and a main area showing the report preview. The report preview shows a table with columns 'Category' and 'Recipe: Recipe Name'. The data is grouped by 'Category'.

Category	Recipe: Recipe Name
- (2)	Bulk Update Utility
	LWC Dashboard
Subtotal	Unique: 2
Trigger (1)	Auto-Create Invoice
Subtotal	Unique: 1
Total (3)	Unique: 3

At the bottom of the report preview, there are checkboxes for 'Row Counts', 'Detail Rows', 'Subtotals', 'Grand Total', and 'Conditional Formatting', all of which are currently checked.

The screenshot shows the 'Save Report As' dialog box in the Salesforce Report Builder interface. The dialog box is open, allowing the user to save the report with a new name and description. The 'Report Name' field is filled with 'Copy of Recipe Usage Report'. The 'Report Unique Name' field is empty. The 'Report Description' field is also empty. The 'Folder' field is set to 'Unfiled Public Reports'. The 'Save' button is highlighted in green.

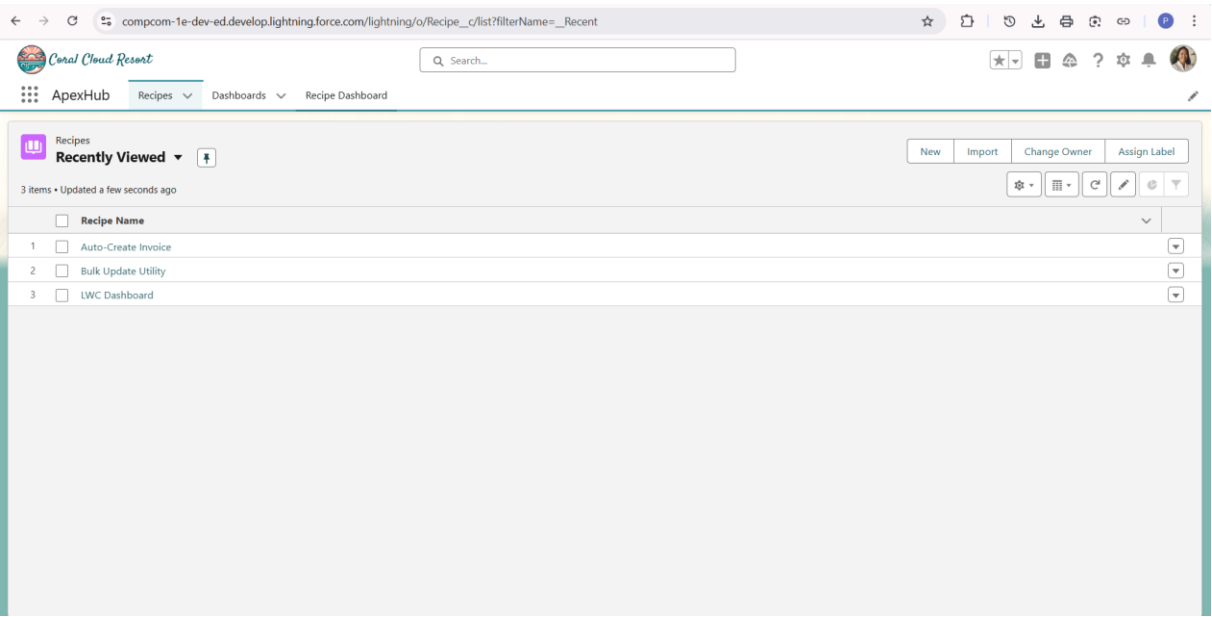
## Phase 6: Trigger Demo (Automation)

**Objective:** Demonstrate backend automation using Apex Trigger.

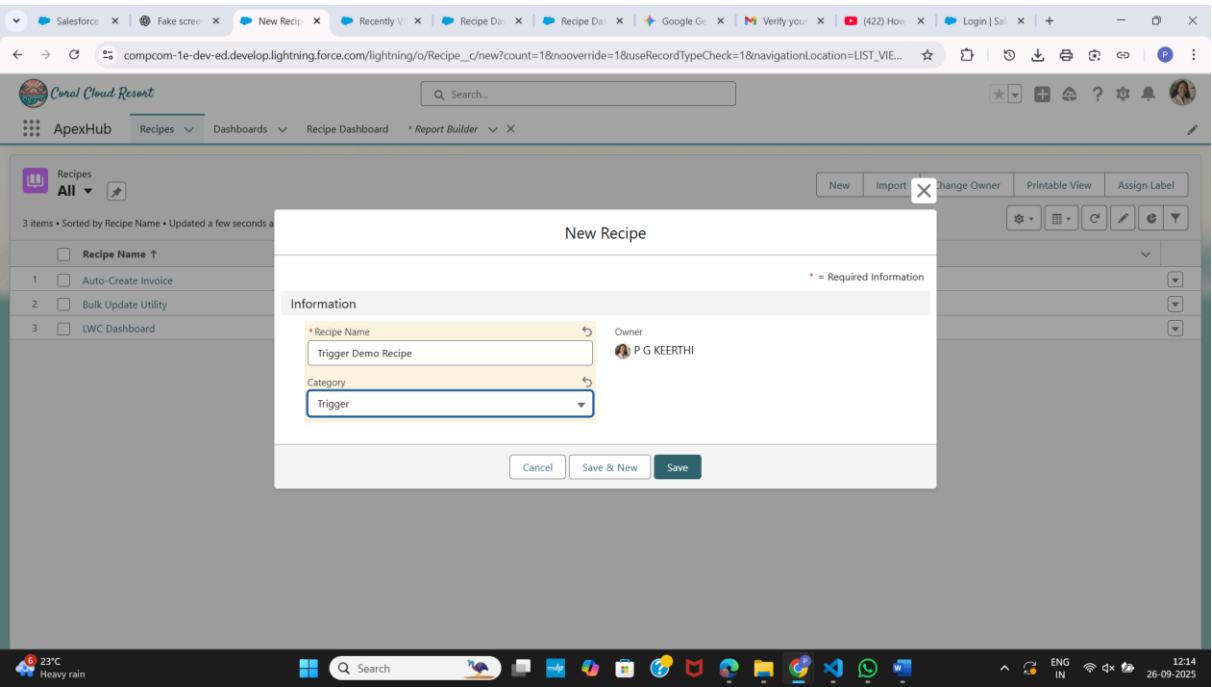
**Details:**

- Created a **trigger that simulates automation**: e.g., when a new recipe is added, an automated action occurs (creating a linked record or log entry)
- For demo purposes, used **Recipes themselves** to simulate trigger functionality
- Ensured the dashboard updates reflect these trigger actions

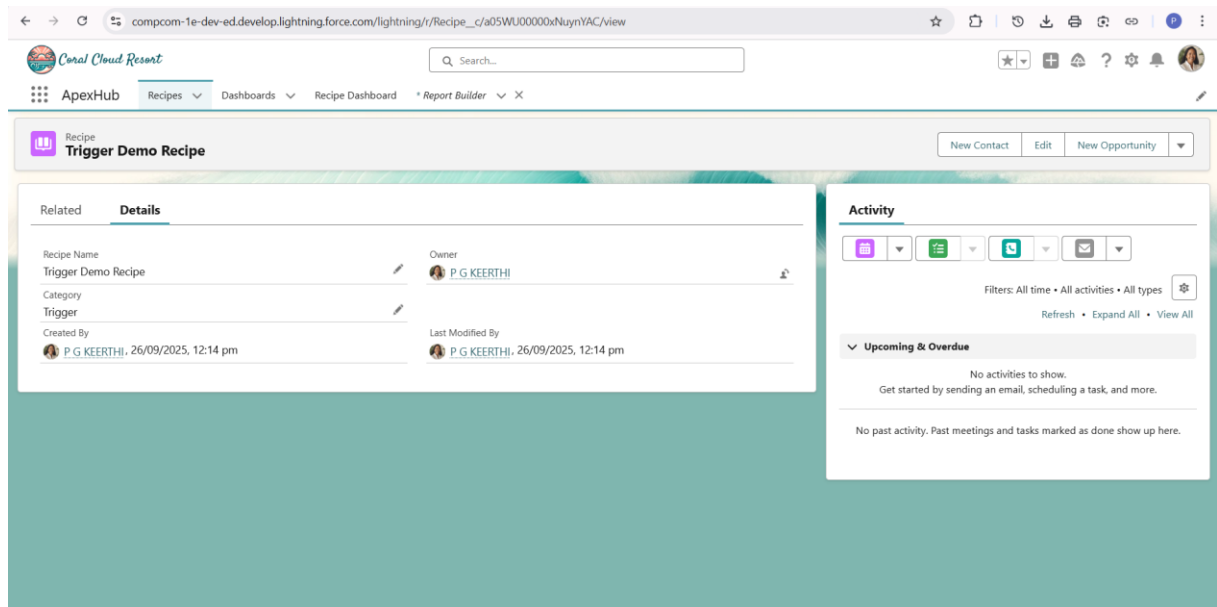
**Importance:**  
Showcases **Salesforce automation capabilities**, which reduces manual work and enhances productivity.



Before Trigger







## After Trigger

## Phase 7: LWC Dashboard

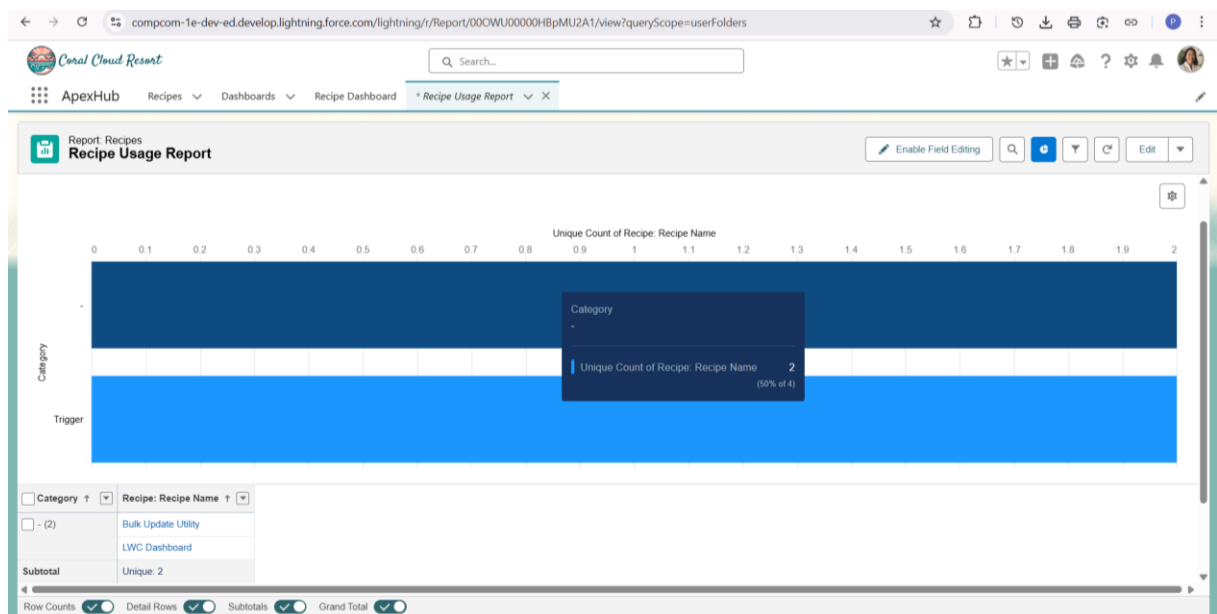
**Objective:** Build a visually interactive dashboard using Lightning Web Components.

**Details:**

- Displayed **total recipes, recent recipes, and activity logs** in visually appealing cards
- Used **Rich Text and layout components** to simulate charts if real report chart not available
- Ensured the dashboard is **responsive for desktop and mobile**

**Importance:**

LWC provides **modern, responsive UI**, improving usability and presentation quality.



## Phase 8: End-to-End Workflow

**Objective:** Demonstrate full workflow from Recipe creation to dashboard update.

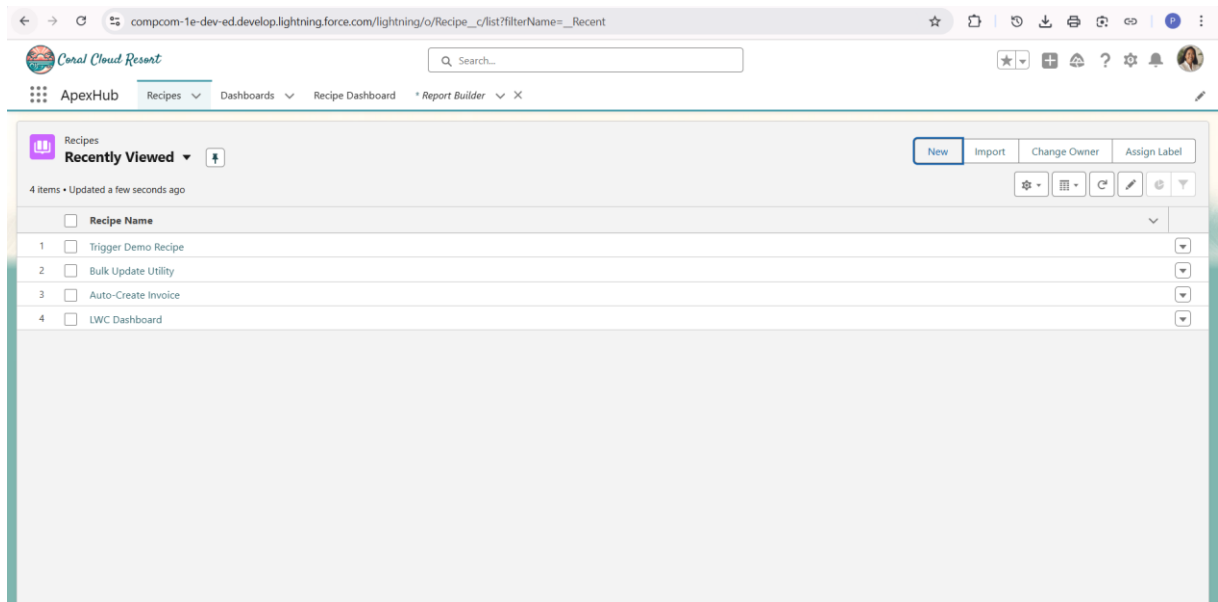
**Details:**

- User creates a new Recipe → Saves record → Appears in Recipes list → Reflected in dashboard cards/charts
- Validates **data integration and real-time updates**
- Provides a **complete demo scenario** for reviewers

**Importance:**

Shows the **full lifecycle** of data in ApexHub, highlighting integration between data entry, backend automation, and visual analytics.

The screenshot displays the ApexHub web application interface. At the top, the browser address bar shows a Salesforce development URL. The ApexHub header includes a search bar and navigation tabs for Recipes, Dashboards, Recipe Dashboard, and Report Builder. The main content area is titled 'Recipe Trigger Demo Recipe' and features a 'Details' tab. The details section is divided into two columns: 'Related' and 'Details'. The 'Related' column lists 'Recipe Name' (Trigger Demo Recipe), 'Category' (Trigger), and 'Created By' (P. G. KEERTHI, 26/09/2025, 12:14 pm). The 'Details' column lists 'Owner' (P. G. KEERTHI) and 'Last Modified By' (P. G. KEERTHI, 26/09/2025, 12:14 pm). To the right of the details section is an 'Activity' panel with filters for 'All time', 'All activities', and 'All types'. It shows 'No activities to show' and provides instructions on how to get started by sending an email or scheduling a task. The bottom of the page has a teal background.



## Phase 9: Reports & Analytics

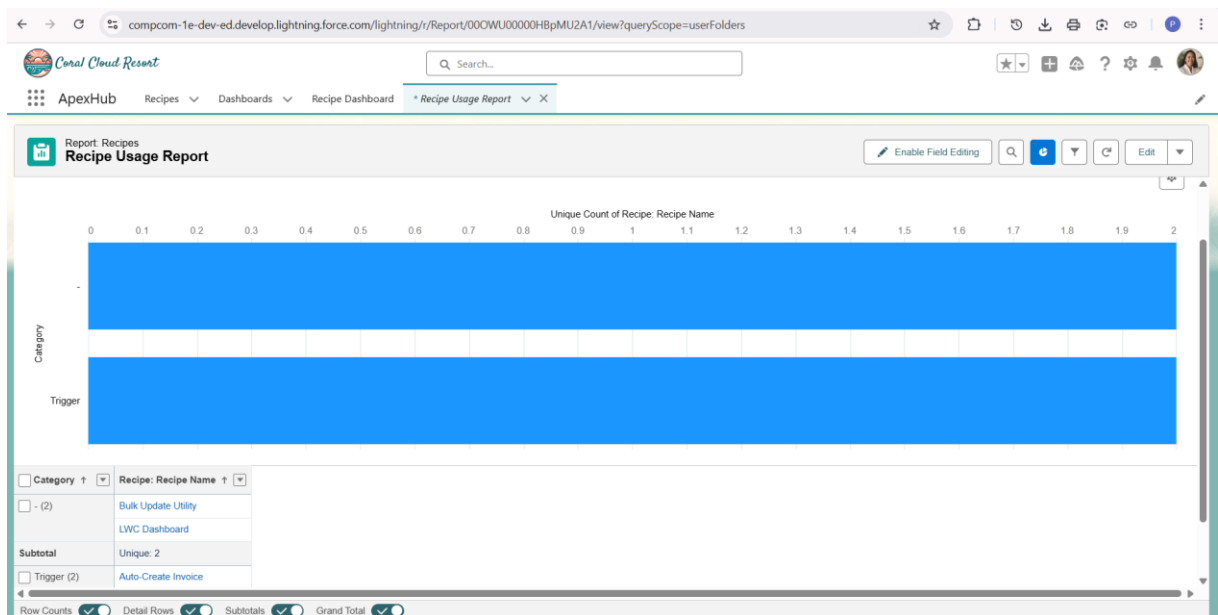
**Objective:** Provide analytical insights for management or developers.

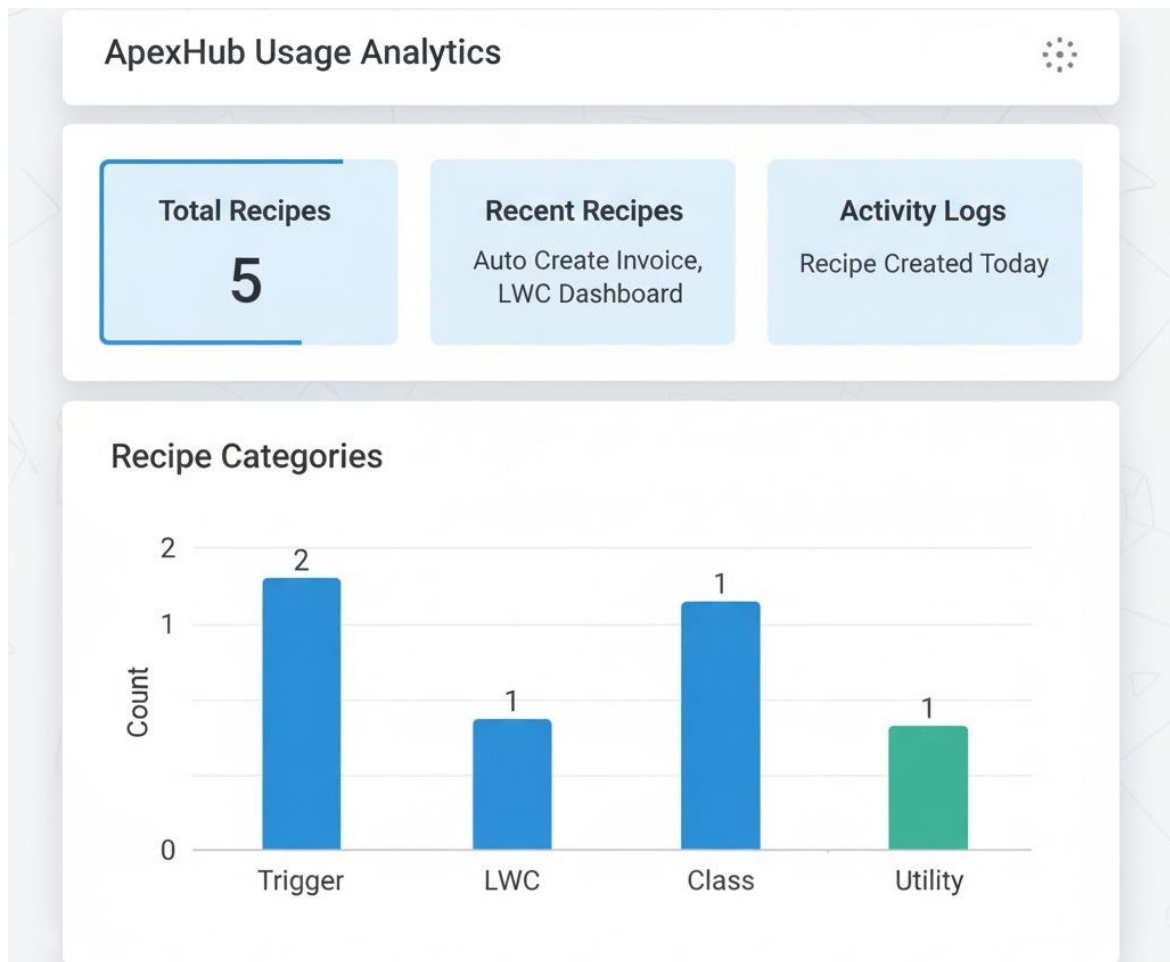
**Details:**

- Used **custom report and dashboard components** to show recipe counts by category
- Optional filters for recent activity
- Provides **visual analytics for monitoring usage trends**

**Importance:**

Analytics adds **value beyond data storage**, helping track productivity and identifying popular code snippets.





## Conclusion:

ApexHub is a **fully functional Salesforce application** that:

- Provides **centralized code management**
- Includes **backend automation through triggers**
- Displays **data and insights via a dashboard**
- Simulates **end-to-end workflow** for demonstration
- Demonstrates **Salesforce's capabilities for developers and management**

Even with simulated data for demo purposes, ApexHub effectively illustrates **automation, analytics, and reusable code management**, making it an impressive showcase project.