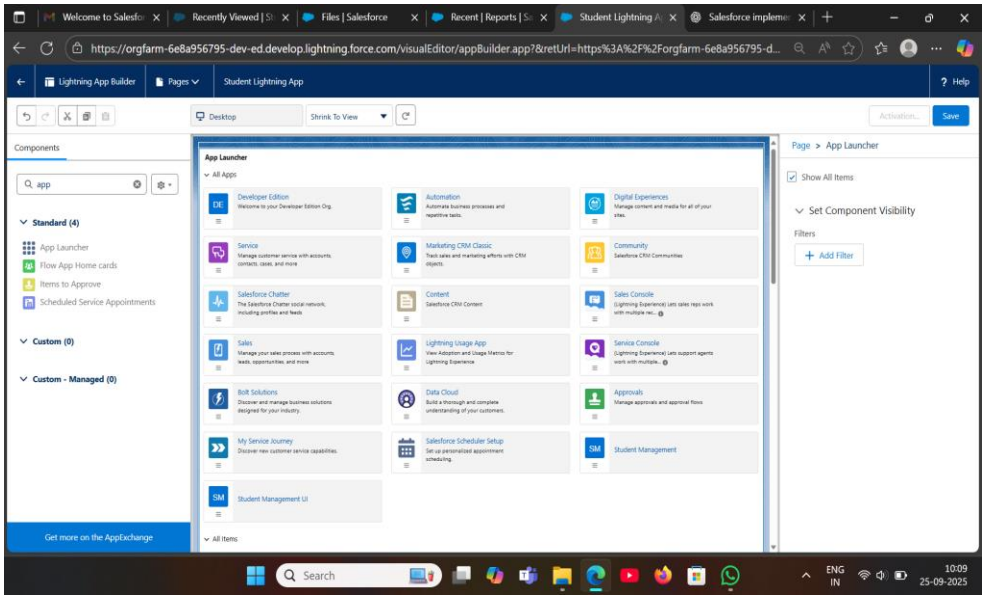
Phase 6: User Interface Development

Goal : To design and implement a user-friendly interface in Salesforce using Lightning tools, LWCs, and Apex integration, enabling users to interact with data efficiently and navigate seamlessly through the application.

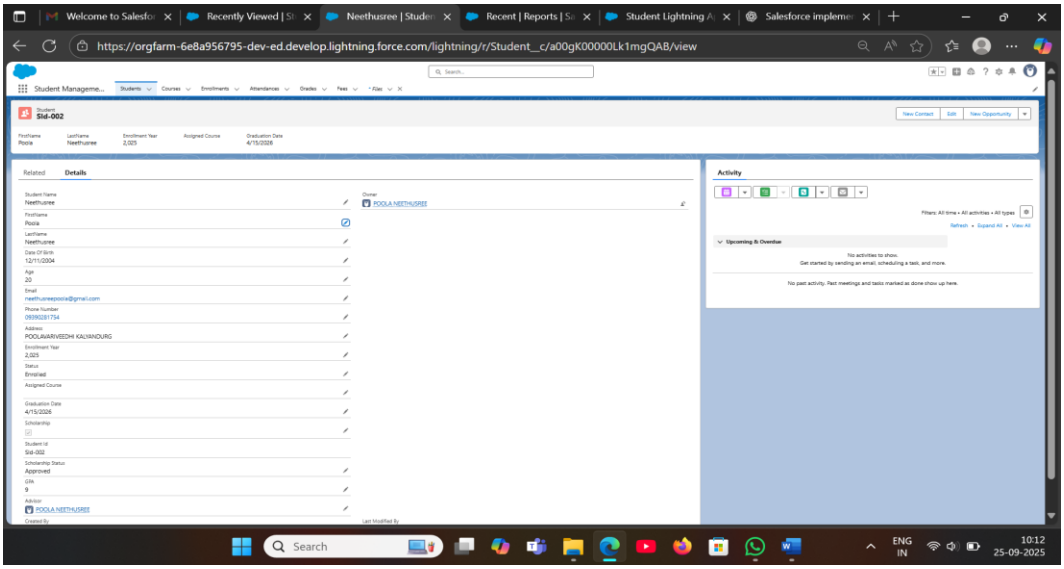
1.Lightning App Builder

- **Purpose:** Drag-and-drop tool to create custom pages without coding.
- **Key Uses:** Build **App Pages**, **Home Pages**, and **Record Pages**.
- **Details:** Place standard components or custom LWCs, save, and activate for users.



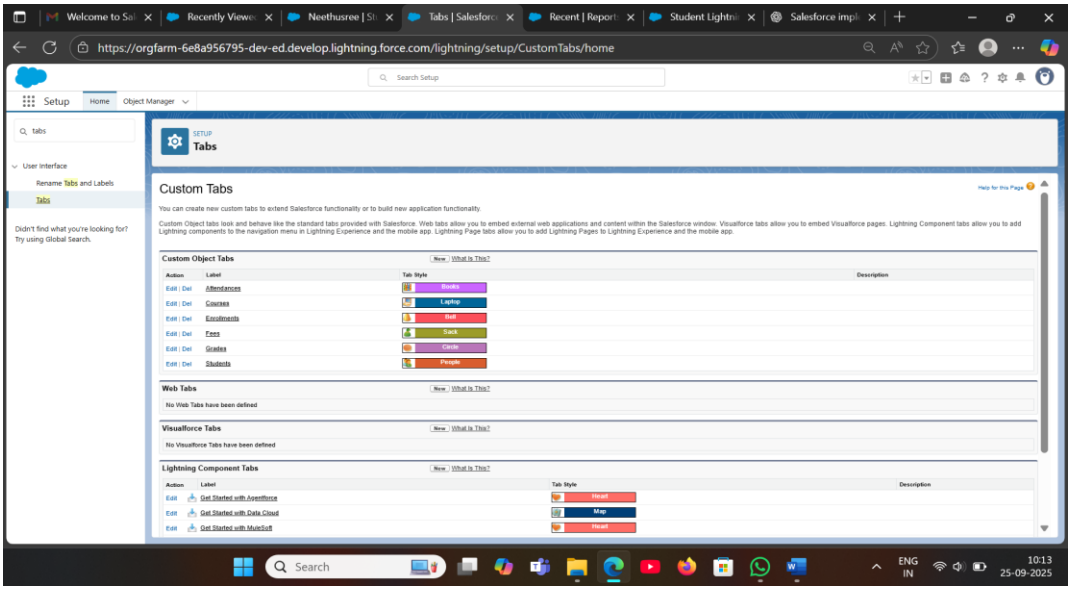
2.Record Pages

- **Purpose:** Customize the layout of individual records.
- **Key Uses:** Display key fields, related lists, and custom LWCs for objects like Student or Course.
- **Details:** Assign pages to apps or profiles to control visibility.



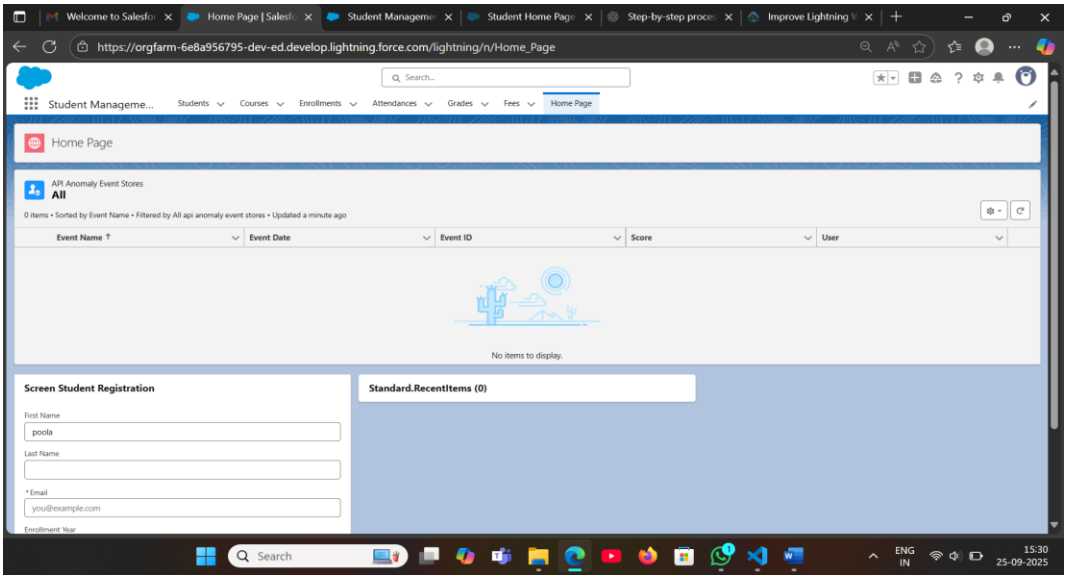
3.Tabs

- **Purpose:** Organize objects, pages, or components into tabs for easy navigation.
- **Key Uses:** Access **Students, Courses, Reports**, or custom pages quickly.
- **Details:** Tabs can host objects, Visualforce pages, web pages, or LWCs.



4. Home Page Layouts

- **Purpose:** Customize the Home Page for better user experience.
- **Key Uses:** Show dashboards, reports, navigation buttons, and custom LWCs.
- **Details:** Layouts define the structure and visibility of components on the Home Page.



5. Utility Bar

- **Purpose:** Quick access to frequently used tools at the bottom of the app.
- **Key Uses:** Add Notes, Messages, or custom LWCs for easy reach.
- **Details:** Can configure behavior, size, and component order.

6. Lightning Web Components (LWC)

- **Purpose:** Build reusable, interactive components for the UI.
- **Key Uses:** Display student lists, details, assignments, certificates, and navigation buttons.
- **Details:** Components consist of HTML (template), JS (functionality), CSS (styling), and meta file (configuration).

7. Apex with LWC

- **Purpose:** Fetch or manipulate Salesforce data using Apex in LWCs.
- **Key Uses:** Display dynamic data like students or courses.
- **Details:** Use **@AuraEnabled methods** in Apex, then call them in JS (Wire or Imperative).

8. Events in LWC

- **Purpose:** Enable communication between components.

- **Key Uses:** Pass data from child to parent, or across sibling components.
- **Details:** Use **Custom Events** (dispatchEvent) or **Lightning Message Service** for cross-component communication.

9. Wire Adapters

- **Purpose:** Automatically fetch Salesforce data reactively.
- **Key Uses:** Display records or lists without manual refresh.
- **Details:** Example: @wire(getRecord, { recordId: '$recordId', fields }) record;

10. Imperative Apex Calls

- **Purpose:** Call Apex methods manually from JS when needed.
- **Key Uses:** Fetch or update data on button click or action.
- **Details:** Example:

```
import getStudents from '@salesforce/apex/StudentController.getStudents';


handleLoadStudents() {
   getStudents()
      .then(result => { this.students = result; })
      .catch(error => { this.error = error; });
}
```

11. Navigation Service

- **Purpose:** Programmatically navigate to records, lists, or external URLs.
- **Key Uses:** Navigate users to **Student details, Course list, or external websites**.
- **Details:** Use NavigationMixin.Navigate() inside LWC methods. Works on **App Pages and Record Pages**, but not standard Home Pages.