# ­­my clinic

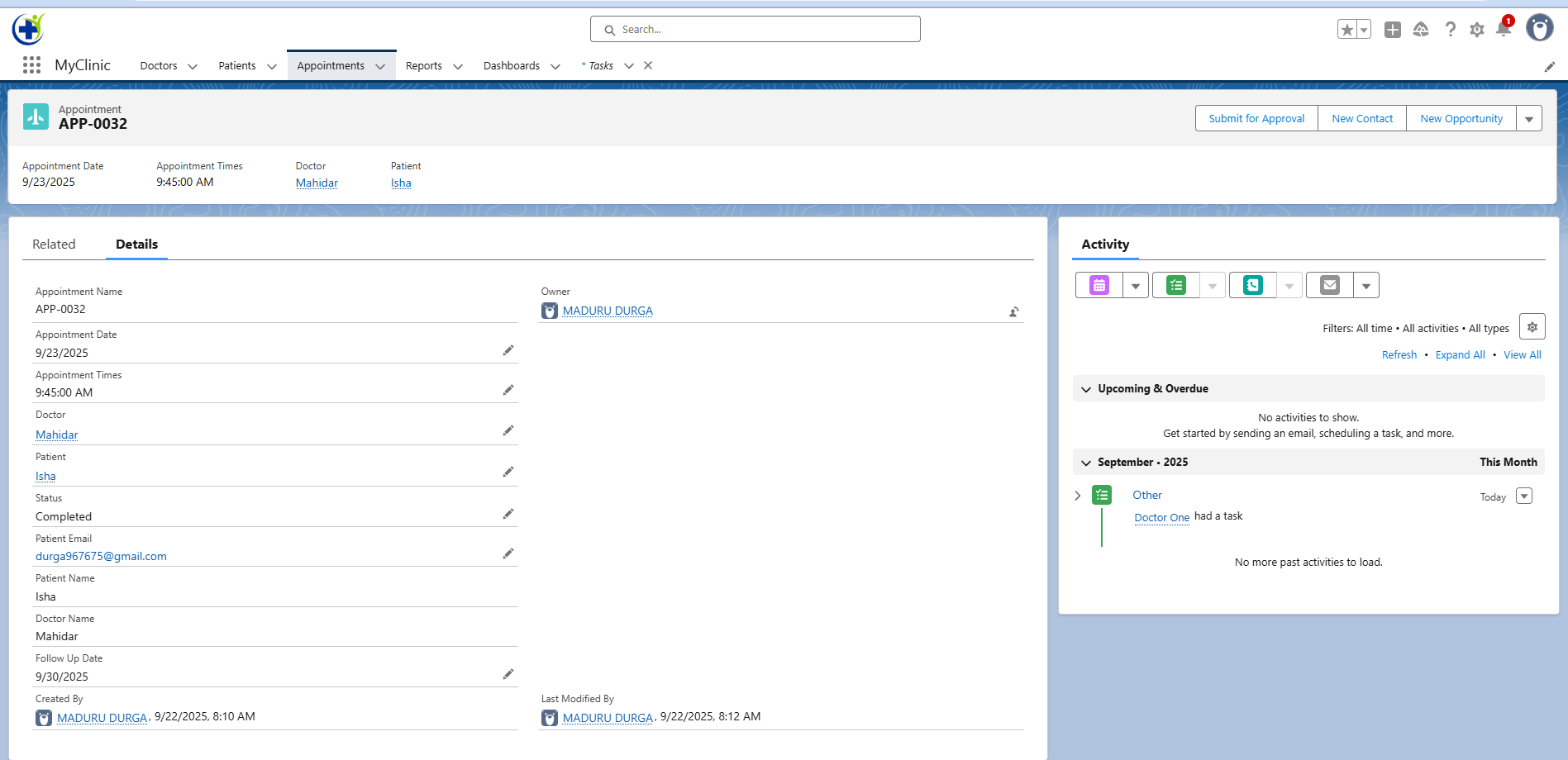
# Smart Appointment Booking – CRM Project

## Phase 6: User Interface Development

* Goal: The primary goal of Phase 6 is to design and deliver an intuitive, efficient, Provide fast, user-friendly navigation and modern user interface for the Smart Appointment Booking System.

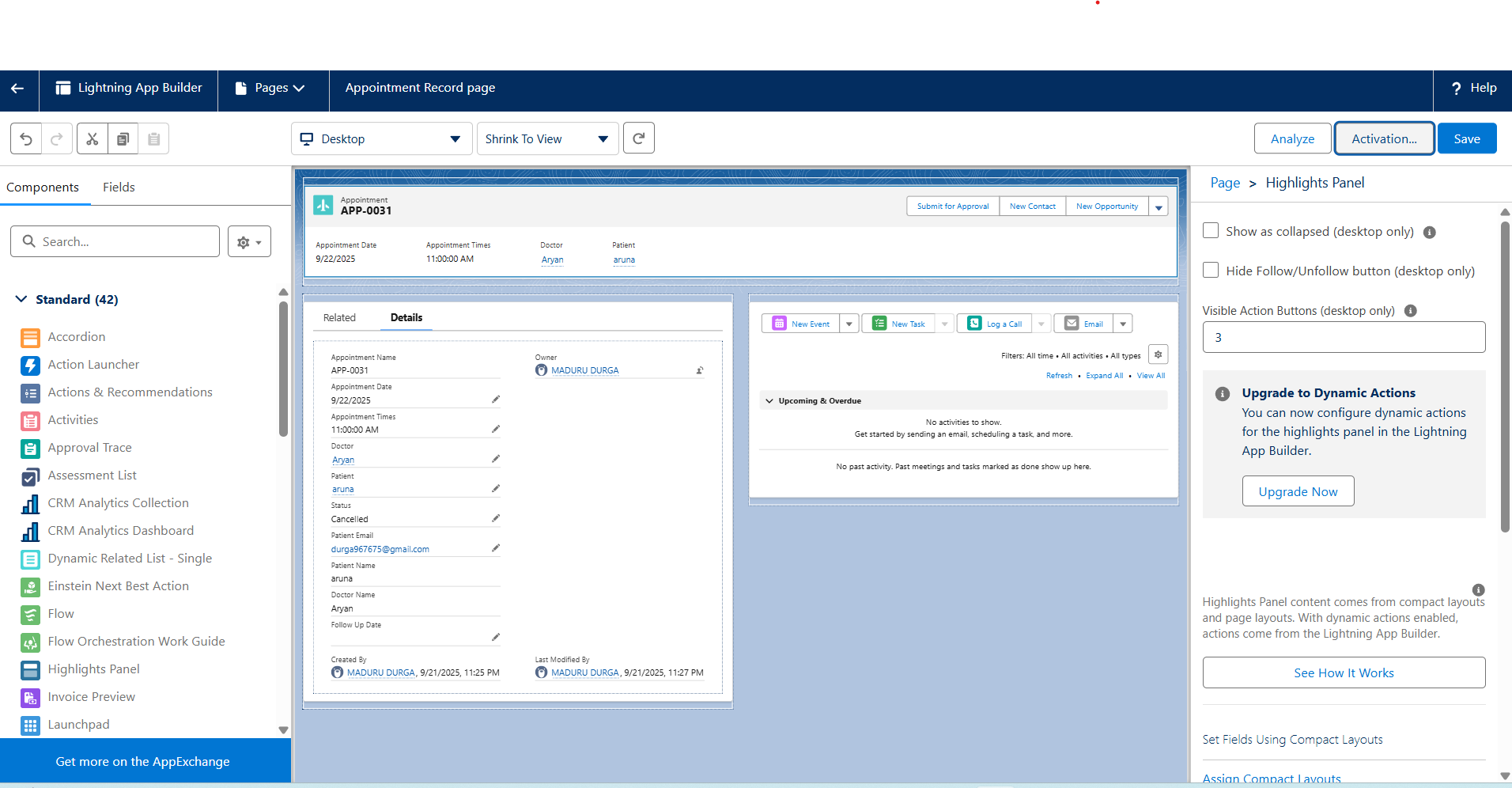
### 1. Lightning App Builder

* Created a dedicated Salesforce app for the Smart Appointment Booking System using the App Launcher, named "MyClinic." This app brings together all core objects such as Appointments, Doctors, Patients, and supporting navigation for users**.**
* The custom app provides an integrated workspace for users to manage appointments, view patient and doctor information, and access custom reports or dashboards**.**

****

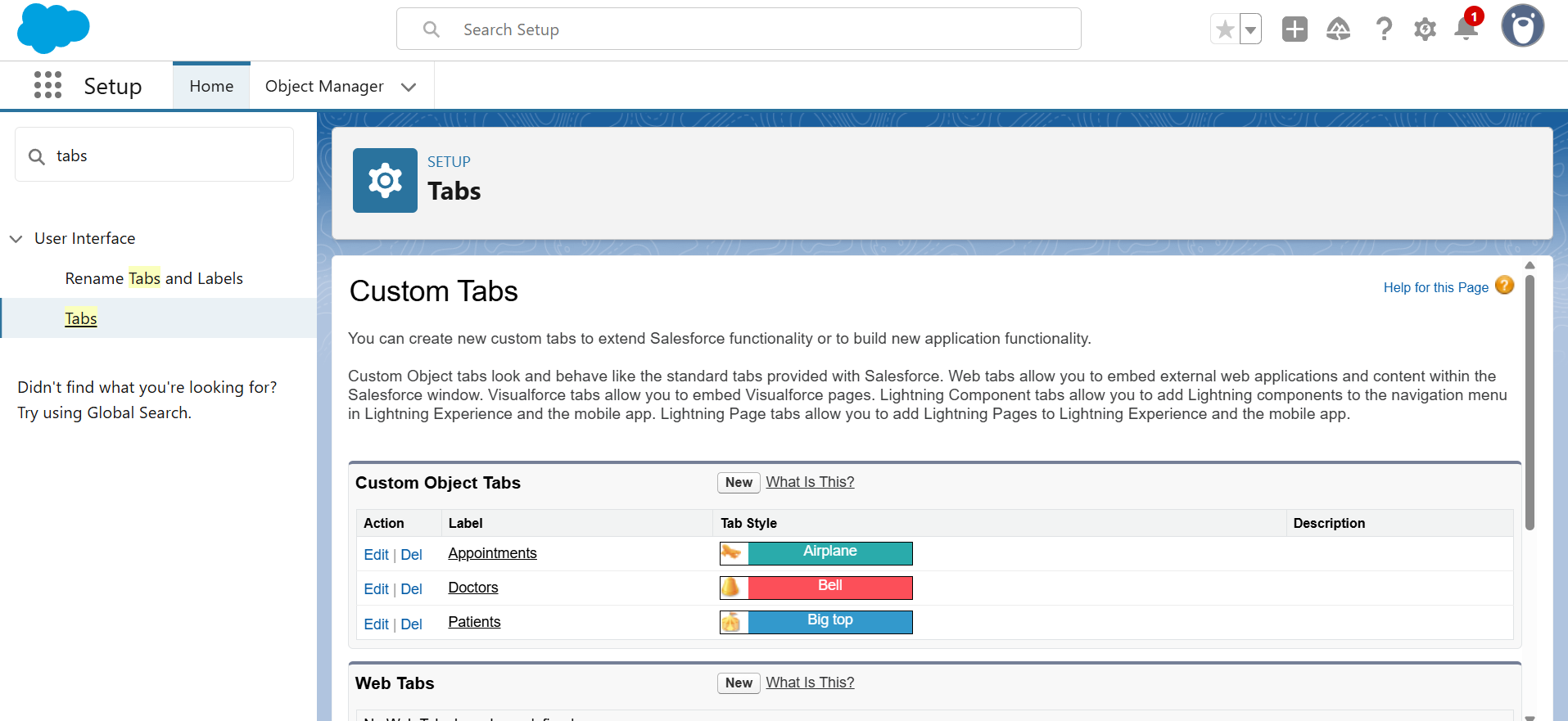
### 2.Record Pages

* We created a custom Appointment Record Page in Lightning App Builder by adding components like Highlights Panel, Tabs (Details & Related), Related List – Approval History, and Activity Timeline.
* Finally, we activated it as the default record page for the MyClinic App.



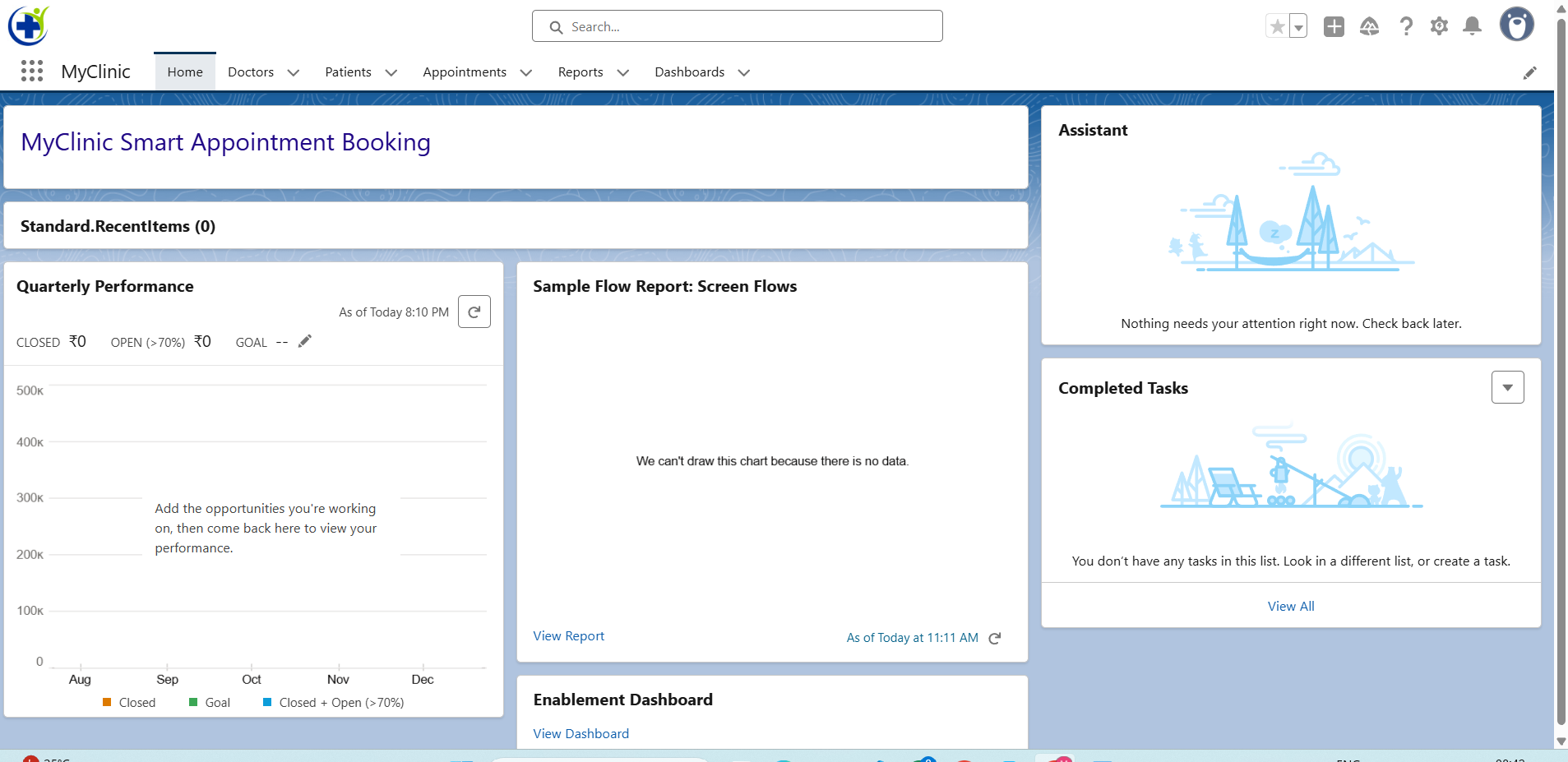
### 3. Tabs

* We created custom tabs for our key objects — Appointment, Doctor, and Patient.
* These tabs allow users to easily access records directly from the navigation bar in the MyClinic Lightning App. Each tab is linked to its respective object and provides a list view of records.



### 4. Home Page Layouts

* We created a custom Home Page using Lightning App Builder and activated it for the MyClinic app.
* This ensures that when users open the app, they see the Home tab with key components like recent records and dashboards for quick access.
* The Home tab helps doctors and staff quickly monitor clinic activities in one place.



### 5. Utility Bar

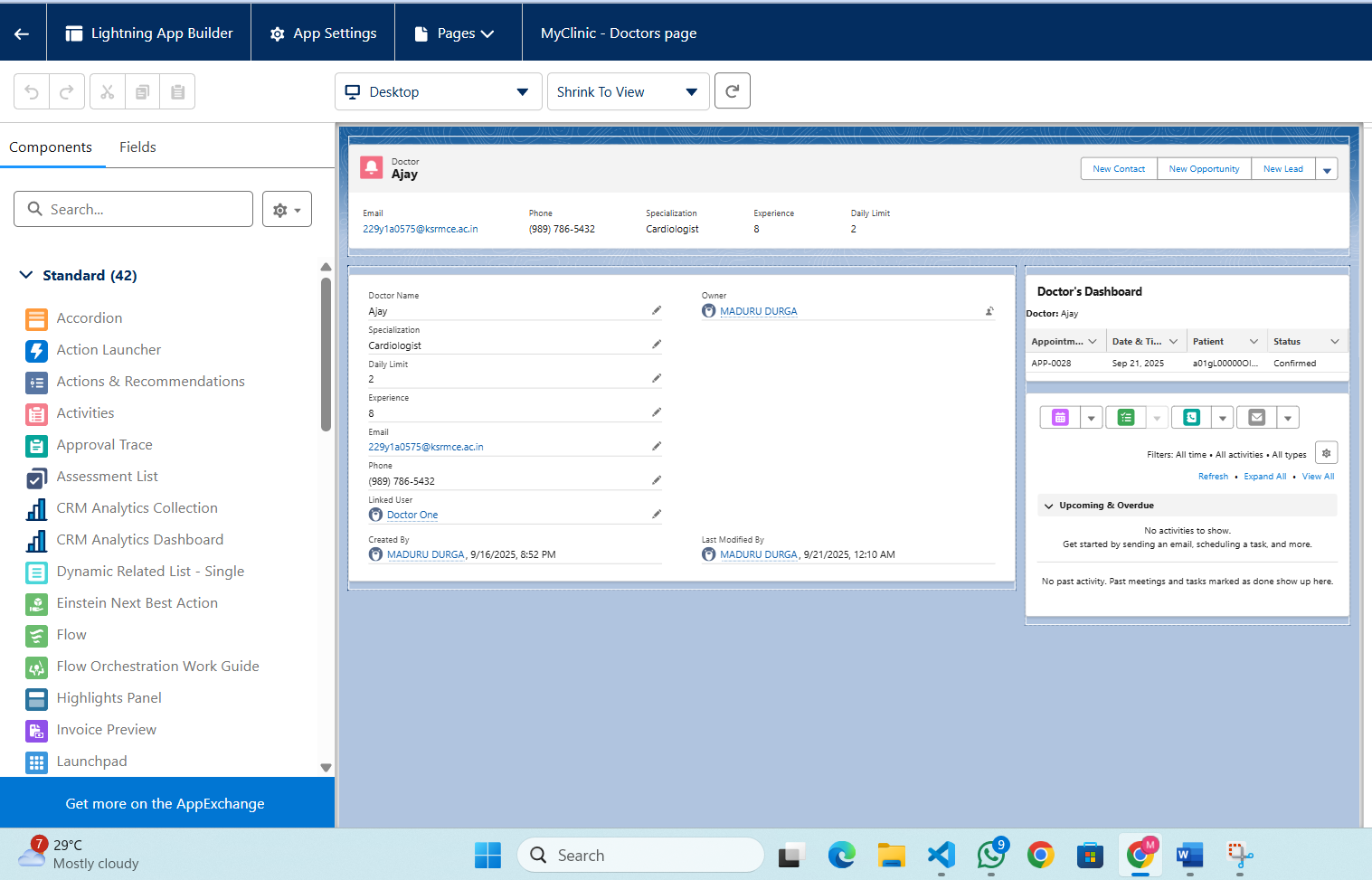
* We customized the Utility Bar in MyClinic App by adding Recent Items, Notes, My Appointments, To-do List and History, giving doctors and admins quick access to important records and notes from anywhere in the app. This improves efficiency and saves navigation time.

A screenshot of a computer

AI-generated content may be incorrect.

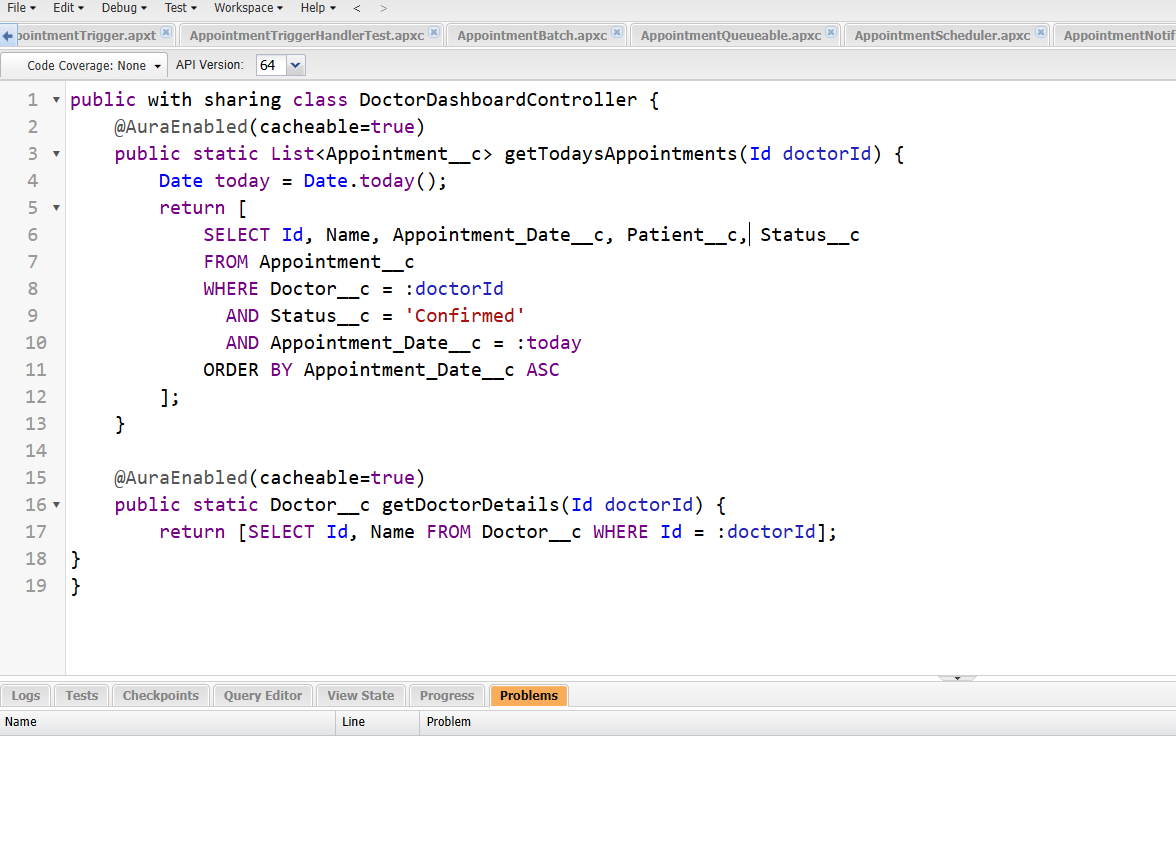
### 6. LWC(Lightning Web Component)

* Created Doctor Dashboard Lightning Web Component (LWC) provides doctors with a real-time view of their daily appointments.
* It displays appointment details, patient names, and statuses in a simple, interactive table.
* The dashboard automatically filters only confirmed appointments scheduled for the current day. This component enhances workflow efficiency and user experience for healthcare professionals.



### 7. Apex With LWC

* The Doctor Dashboard LWC uses an Apex class (DoctorDashboardController) to access appointment and patient data securely from Salesforce.
* The Apex methods are annotated with @AuraEnabled, allowing the LWC to call them via Salesforce's Lightning Data Service.
* This integration enables efficient server-side filtering—so only today's confirmed appointments for the current doctor are shown. By combining Apex and LWC, the solution delivers flexible, real-time healthcare dashboards tailored to each user's needs.

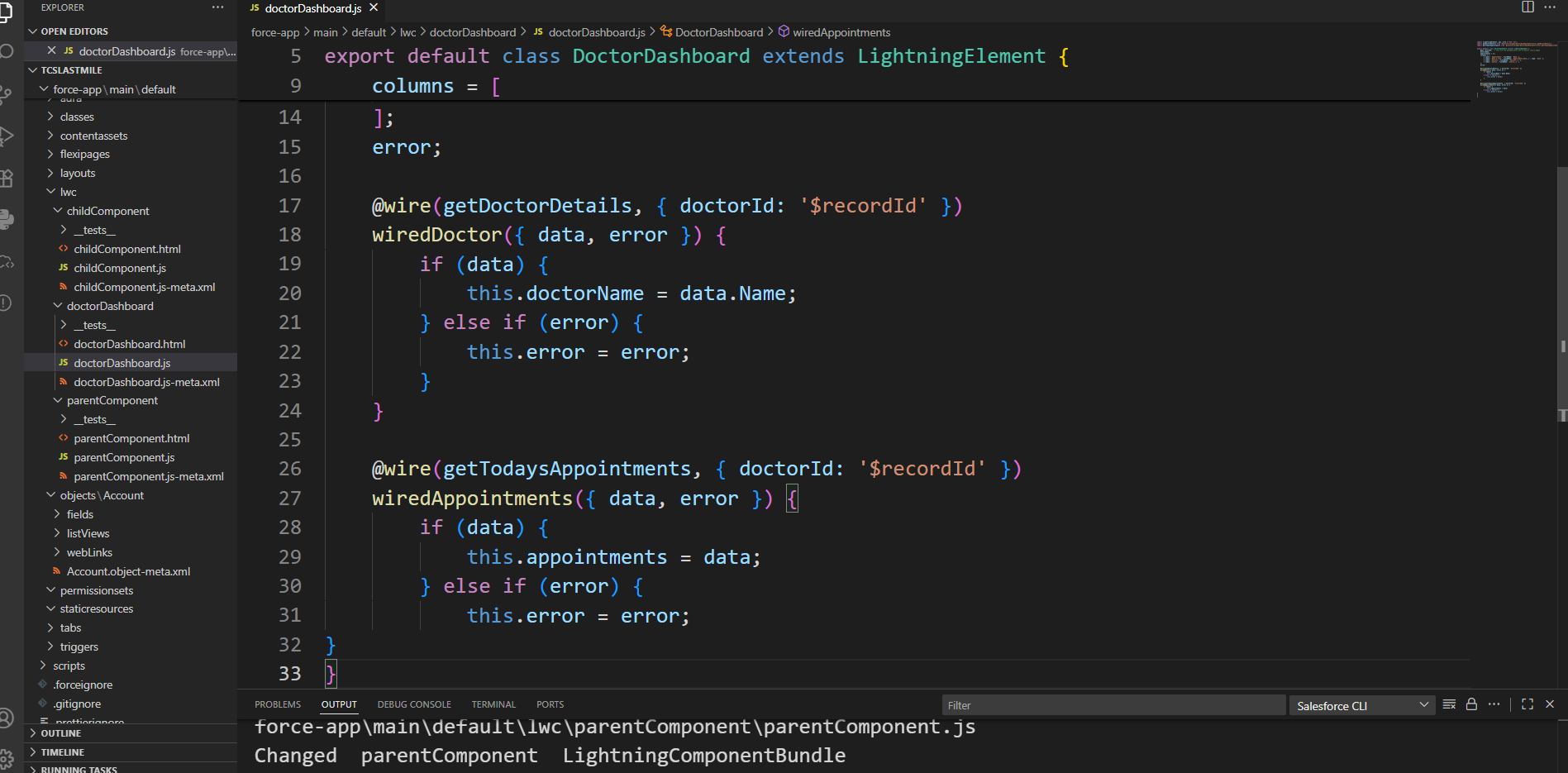


### 8. Events in LWC

* Events in LWC are not required for this project because the application's functionality does not depend on communication between parent and child components. Since all features are handled within single components or via Apex, event handling introduces unnecessary complexity.

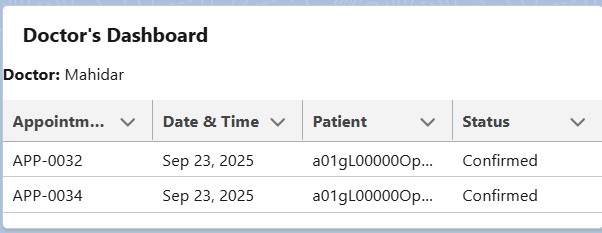
### 9. Wire Adapters

* The concept of wire adapters was already covered during the development of the Doctor Dashboard component.
* In that component, data from Salesforce was fetched and displayed reactively using either the @wire decorator with Apex methods or standard wire adapters.
* This allowed real-time display and automatic updates of appointment and doctor data in the dashboard, ensuring our application stays in sync with the latest changes from the backend.
* No additional wire adapter demonstration is needed for this project since it was implemented in the Doctor Dashboard feature.



### 10. Imperative Apex Calls

* In the Doctor Dashboard component, Imperative Apex Calls are used to fetch appointment and doctor data from Salesforce only when required.
* This approach gives complete control over when backend queries are run and enables parameters to be passed dynamically. After a user action (such as selecting a doctor record), the component calls the appropriate Apex method and updates the UI with the returned results.
* Imperative calls ensure the dashboard loads correct, up-to-date data and handles errors, making the interface responsive to user input and backend changes.



### 11. Navigation Service

* The Doctor's Dashboard component includes Navigation Service functionality, allowing users to navigate to standard Salesforce pages directly from within the dashboard. In this implementation, a "Go To Home" button is provided as an example of how custom navigation can be triggered through LWC..

