Boutique App CRM

The Boutique Management App is a custom Salesforce application designed to help boutique owners seamlessly manage their customers, products, and sales in one centralized platform.

**Problem Statement**

Boutiques often face challenges in managing inventory, handling customer appointments, and delivering personalized experiences due to reliance on manual processes or basic systems. This leads to issues like overlapping appointments, stock mismatches, and limited visibility into sales trends. To overcome these challenges, a custom Boutique Management App is needed to centralize operations, prevent errors, streamline appointment scheduling, track products efficiently, and provide managers with insightful reports for better decision-making and improved customer satisfaction.

**Phase 1: Problem Understanding & Industry Analysis**

**1. Requirement Gathering**

**Goal: Understand what each stakeholder needs from the boutique management system.**

* **Customers:** Easy browsing of collections, personalized recommendations, loyalty rewards, seamless online & in-store experience.
* **Business Owners:** End-to-end visibility on sales, inventory, customer engagement, and profitability.

**2. Stakeholder Analysis**

**Goal: Identify roles and responsibilities in the ecosystem.**

* **Primary Stakeholders:**
  + **Customers →** End users, purchasing products online or in-store.
  + **Sales Staff →** Assist with transactions, update customer preferences, and upsell products.
  + **Store Manager →** Manages daily operations, inventory, and staff schedules.
* **Secondary Stakeholders:**
  + **Business Owner/Management →** Strategic decisions on sales, marketing, and expansion.
  + **IT/Salesforce Admins →** Maintain the boutique CRM & integrations.
  + **Suppliers/Vendors →** Provide stock and manage restocking cycles.

**3. Business Process Mapping**

**Goal: Understand current challenges vs. how Salesforce can improve operations.**

**Current Process (Manual/Traditional):**

* Customer info tracked manually in diaries/spreadsheets.
* Inventory updated at end of the day → stock mismatches.
* Customer preferences rarely tracked → missed upsell opportunities.
* Marketing mostly word-of-mouth or generic promotions.

**Proposed Process (Salesforce Enabled):**

* Customer interactions captured automatically (POS, website, social media).
* Real-time inventory sync with alerts for low stock.
* Automated loyalty & rewards programs.
* Personalized fashion recommendations powered by Salesforce AI.
* Targeted marketing campaigns (SMS, email, WhatsApp) from Salesforce Marketing Cloud.

**4. Industry-Specific Use Case Analysis**

**Goal: Benchmark against best practices in Boutique/Retail.**

* **Customer Retention:**
  + Many boutiques lose customers due to lack of personalized follow-up.
  + **Solution →** Salesforce CRM with customer history, birthdays, style preferences, automated reminders.
* **Inventory Management:**
  + Stockouts or overstocking are common.
  + **Solution →** Inventory dashboard with predictive restocking.

**5. AppExchange Exploration**

**Goal: Identify existing Salesforce apps to reduce development effort.**

* **Potential Apps for Boutiques:**
  + POS/Inventory Management Apps.
  + Loyalty Management Apps (customer rewards & referrals).
  + Marketing Automation Apps (SMS, WhatsApp, Email campaigns).

**Phase 2: Org Setup & Configuration (Boutique App)**

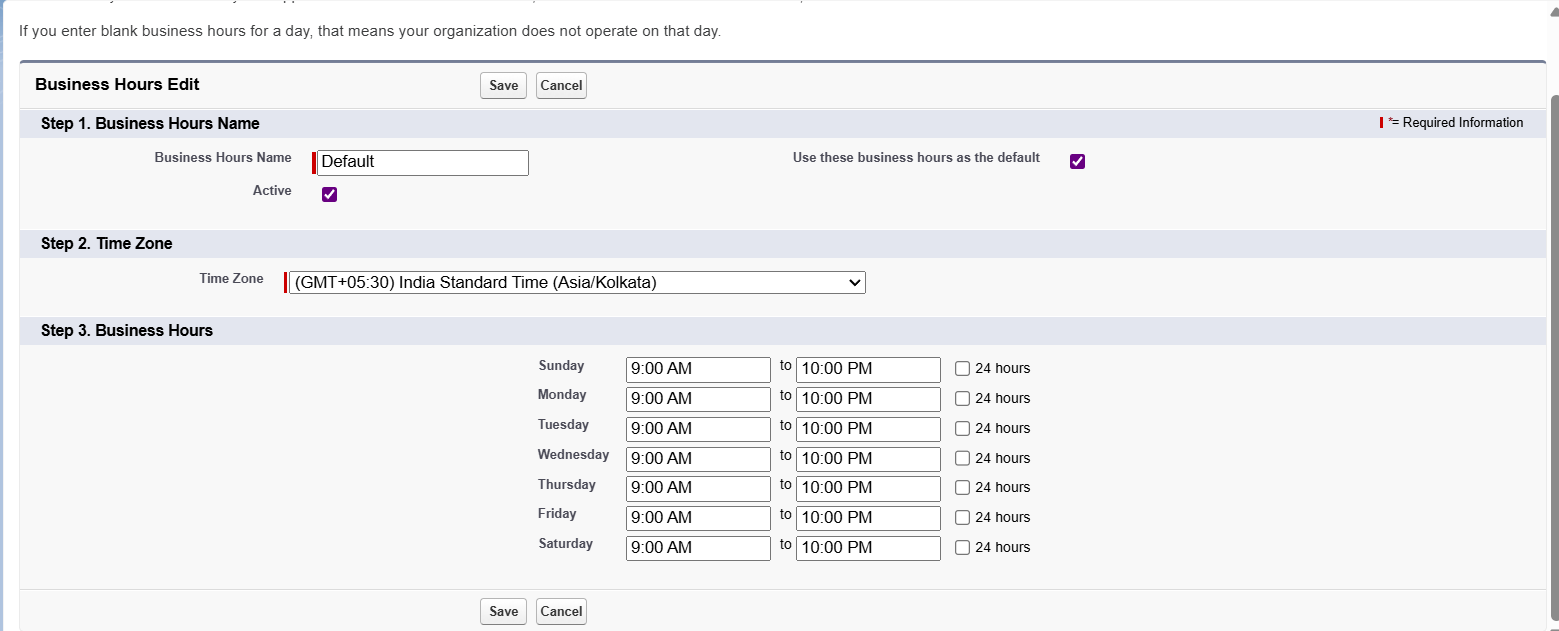
**Goal: Prepare Salesforce environment for the Boutique application.**

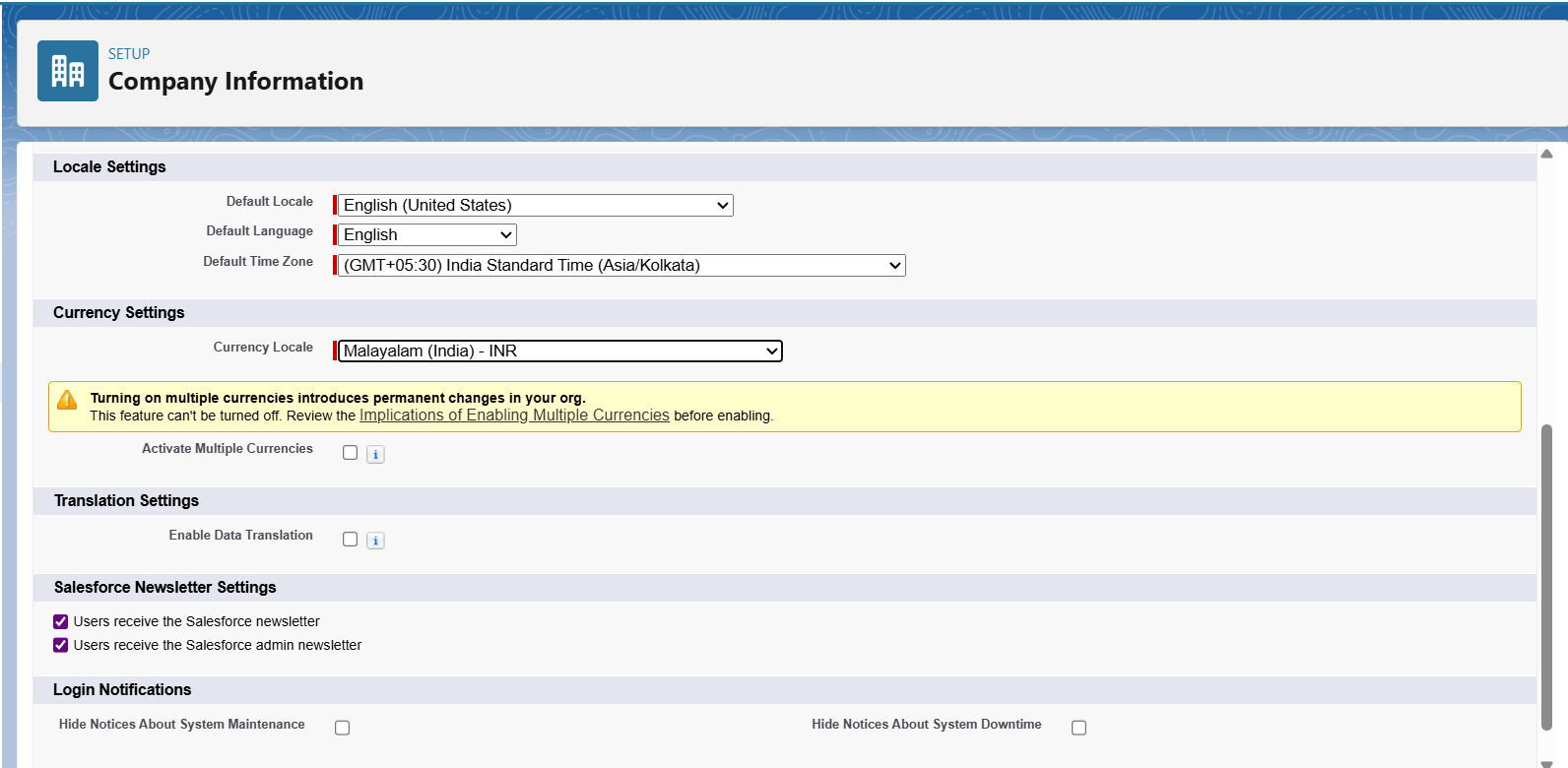
**1. Salesforce Editions**

For the Boutique App, we will use a Salesforce Developer Edition Org (free) to build and test all features such as appointment scheduling, customer management, and order tracking. This Developer Org acts as a safe environment where we can configure, customize, and test without affecting real data. Once everything is finalized and tested, the solution can later be moved to a Production Org, where it will be used by the boutique staff to manage real customers, appointments, and sales transactions.

**2. Company Profile Setup**

For the Boutique App, I set up the company details with the boutique’s name, address, and contact information, configured the time zone to match the boutique’s location, and set the default currency to INR (₹) or USD ($) depending on customer needs.

****

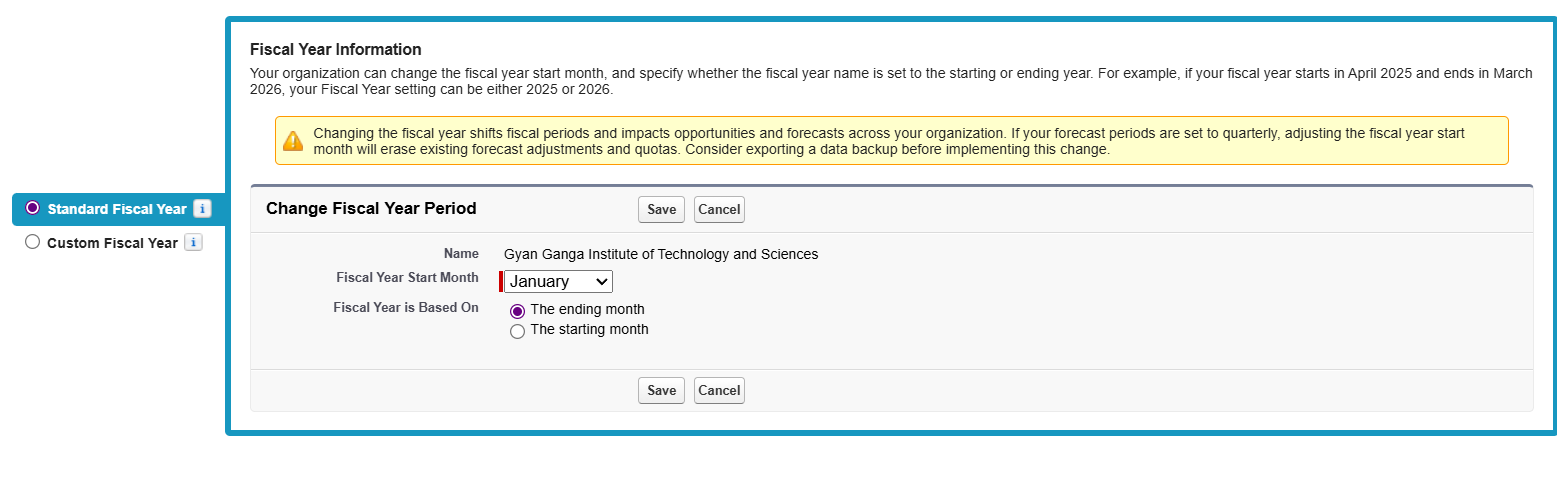
****

**3. Business Hours & Holidays**

For the Boutique App, the business hours are set from 9:00 AM to 10:00 PM, aligning with typical boutique operating times. In addition, public holidays such as Diwali, Christmas, and other major festive days are marked as non-working days to ensure that customers cannot schedule appointments on those dates.

**4. Fiscal Year Settings**

The Standard Fiscal Year (Jan–Dec) is used for the revenue & sales reporting**.**

****

**5. User Setup & Licenses**

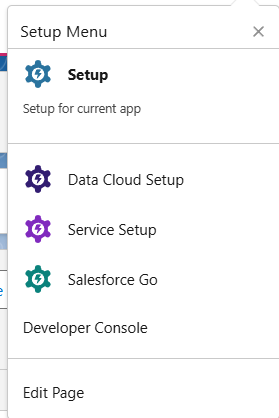
Since it is an application to manage a small-scale business so not much profile setup and licences is need.

**6. Login Access Policies**

For the Boutique App, staff login access is restricted to 10:00 AM – 8:00 PM, ensuring that employees can only log in during active working

**7. Dev Org Setup**

For the Boutique App, the Developer Org serves as a sandbox environment where all configurations, automation, Lightning components, and Apex development can be safely built and tested without affecting real customer data.

****



**8. Deployment Basics**

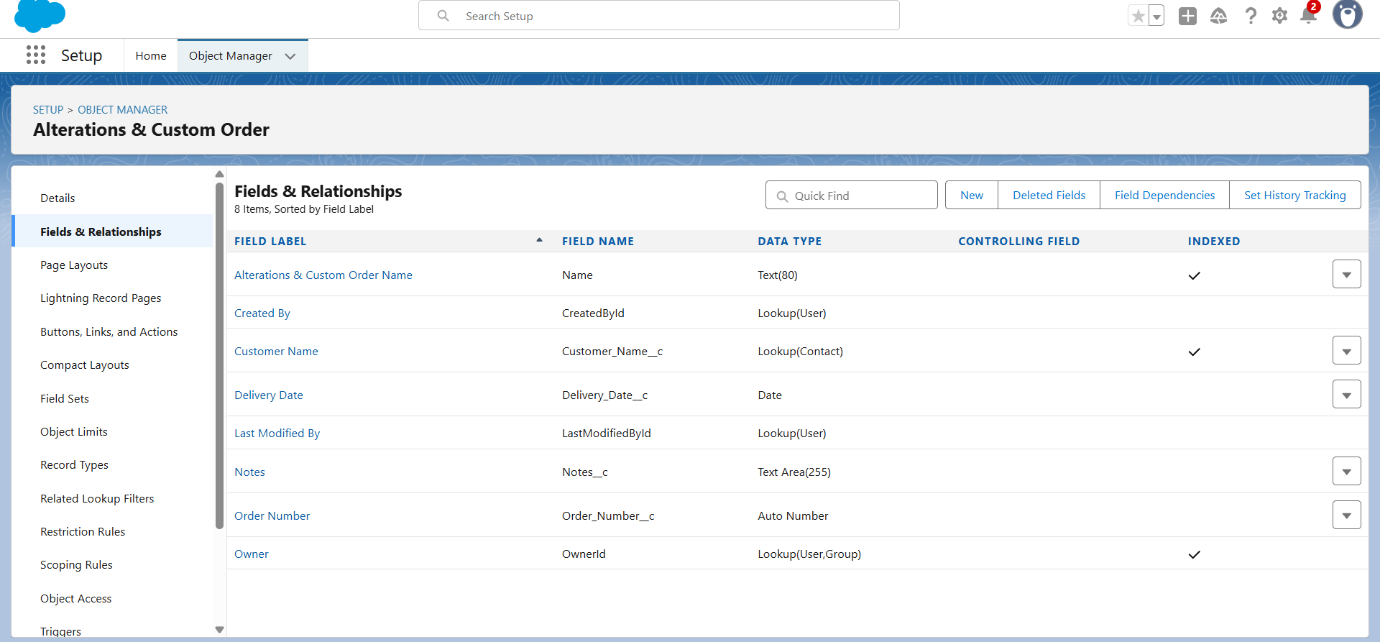
For the Boutique App, deployment involves moving all configurations and code—including custom objects, flows, Apex classes, and Lightning Web Components—from the sandbox environment to the production org. This can be done using tools such as Change Sets, VS Code with Salesforce CLI, or DevOps Center to ensure a smooth and controlled transition to the live environment.

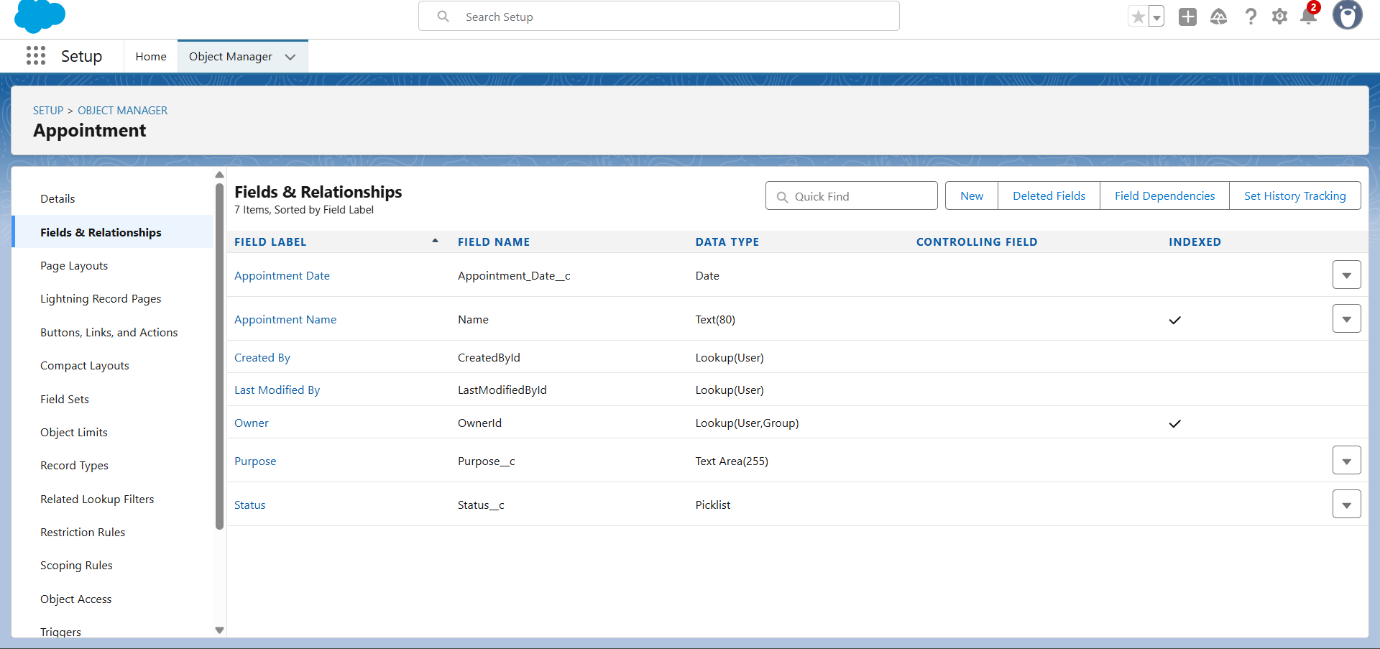
**Phase 3: Data Modelling & Relationships**

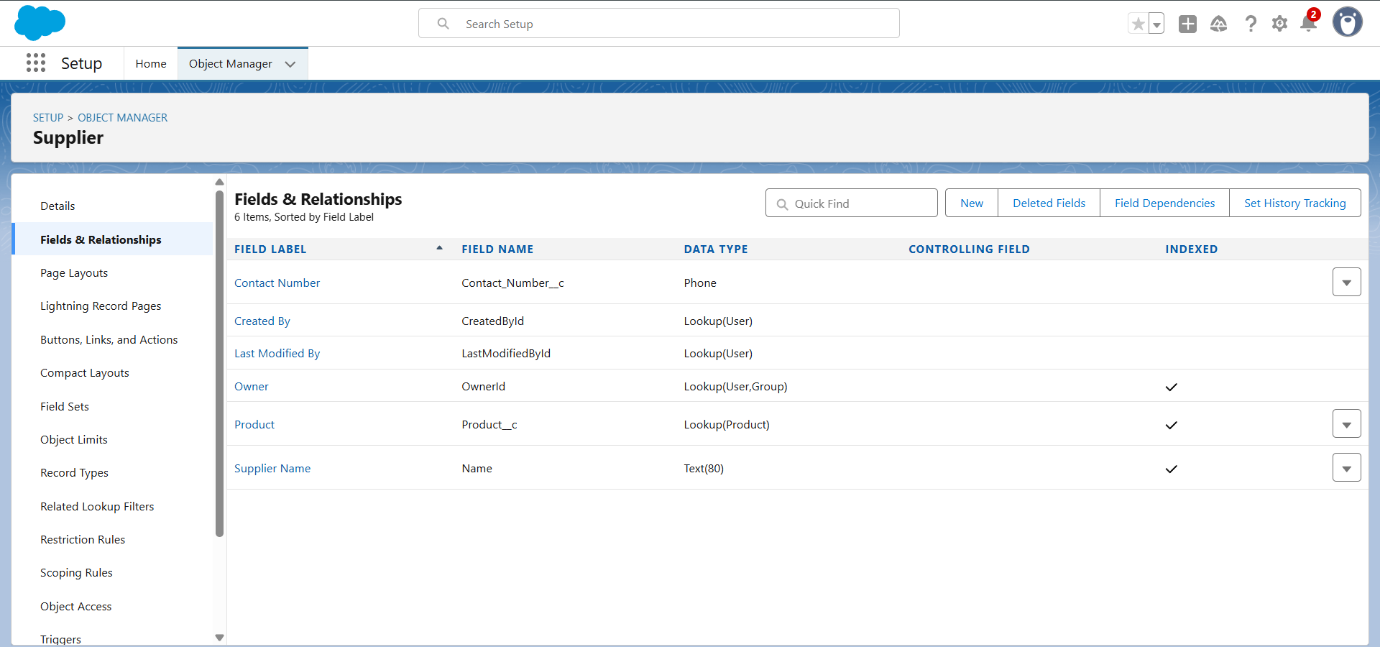
**Goal: Build data structure.**

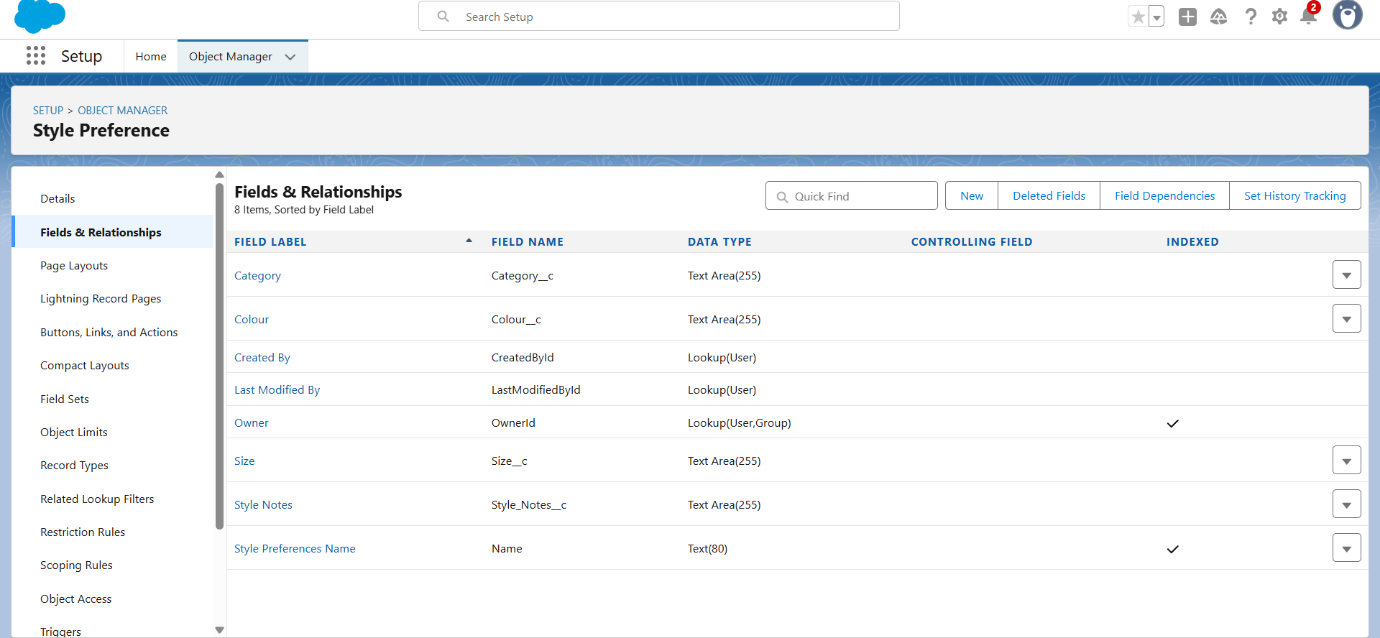
1. **Standard & Custom Objects**
   * Standard: Contact (customers), Products, Orders.
   * Custom: Alterations & Custom Orders, Appointments, Style Preferences, Supplier.
2. **Fields**
   * Alteration & Custom Orders: Alterations & Custom Order Name, Delivery Date, Notes, Customer Name, Order Number
   * Appointments: Appointment Name, Appointment Date, Purpose, Status
   * Style Preferences: Style Preferences Name, Category, Colour, Size, Style Notes
   * Supplier: Supplier Name, Contact Number, Product

**Screenshots**

****

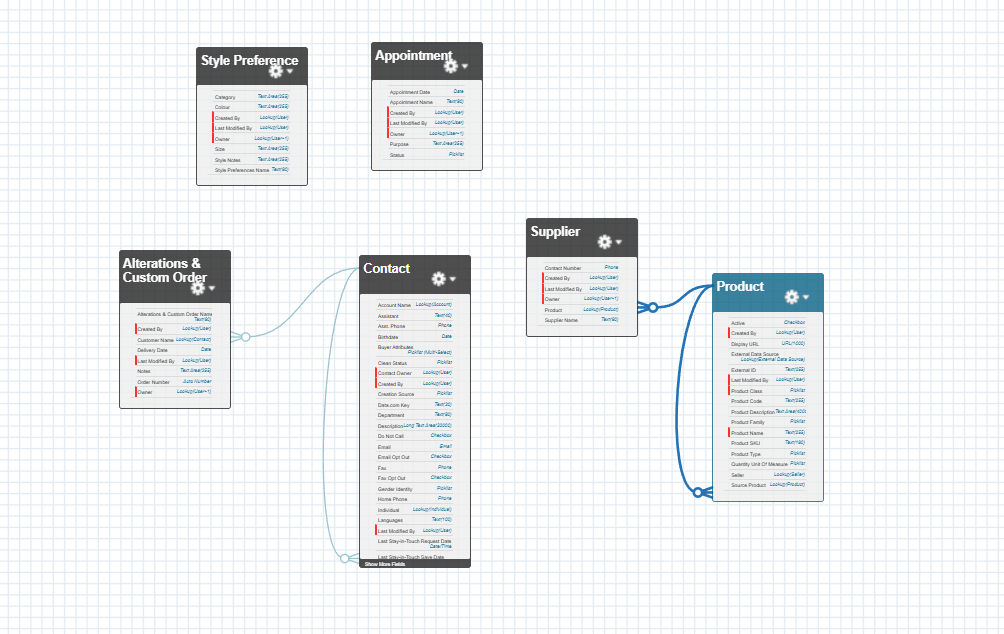
****

****

****

1. **Schema Builder**

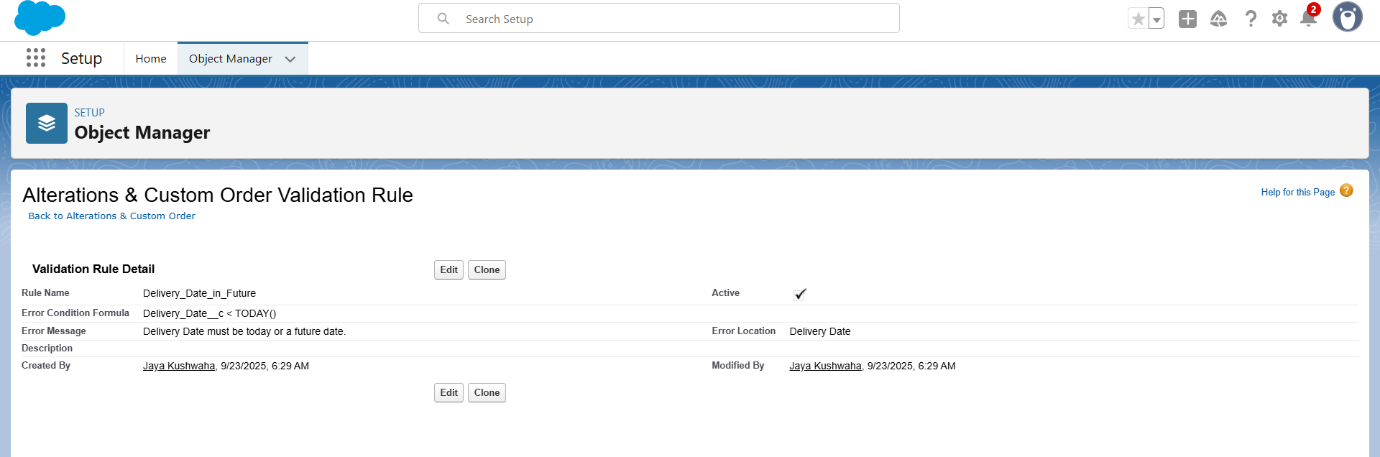
Schema Builder serves as a visual blueprint of how the boutique’s data is structured and connected. It allows you to easily create and manage custom objects like Customers, Appointments, and Products, while also defining the relationships between them, such as linking customers to their appointments or associating products with those appointments. Instead of just looking at lists of fields, Schema Builder presents everything in a clear diagram, making it easier to understand how data flows across the system. For the boutique, this means being able to track how customers schedule appointments, what services or products are included, and how these records interact, ensuring smooth operations and avoiding gaps or duplication in the system.

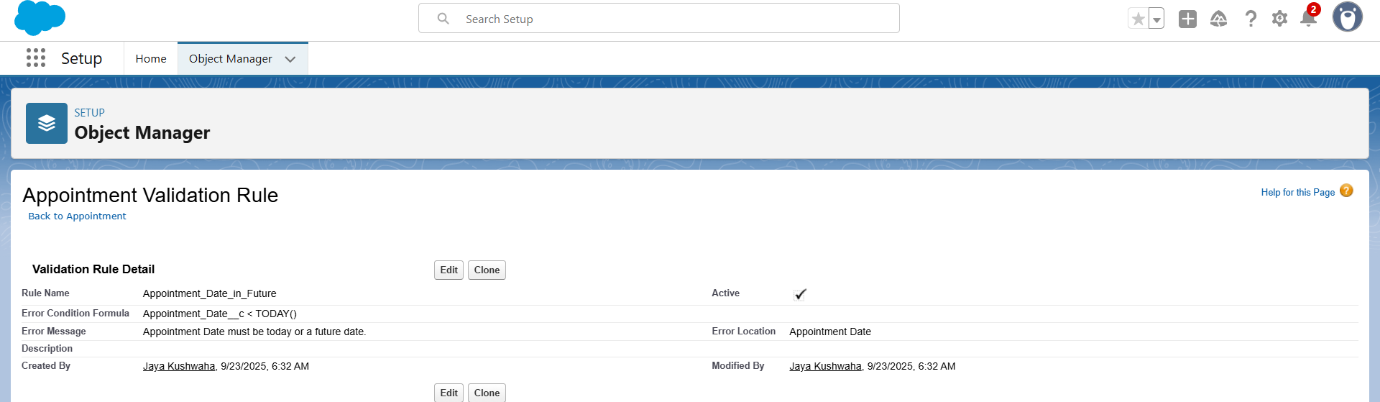
****

**Phase 4: Process Automation (Admin)**

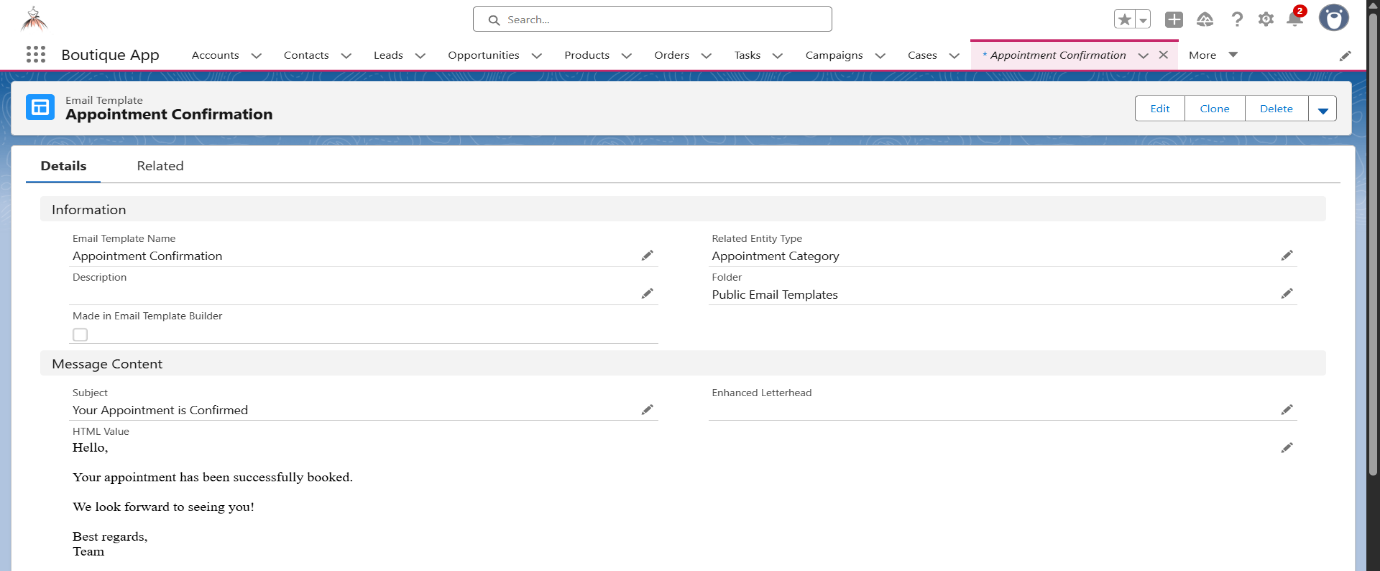
**Goal: Automate tasks.**

1. **Validation Rules**
   * Appointment Date must be not from past.
   * Delivery Date must be not from past.

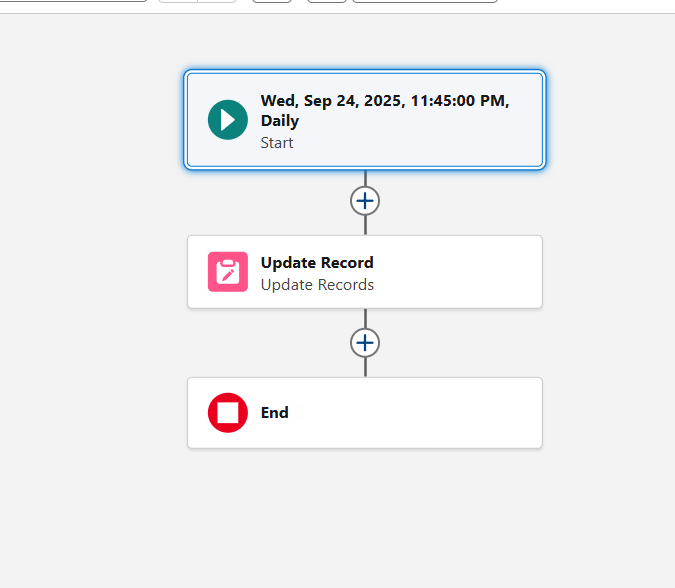




1. **Email Alerts**
   * Customer gets email after approval.



1. **Field Updates**
   * After the Appointment is completed the Status is updated to “Completed”.



**Phase 5: Apex Programming (Developer)**

**Goal: Add advanced logic.**

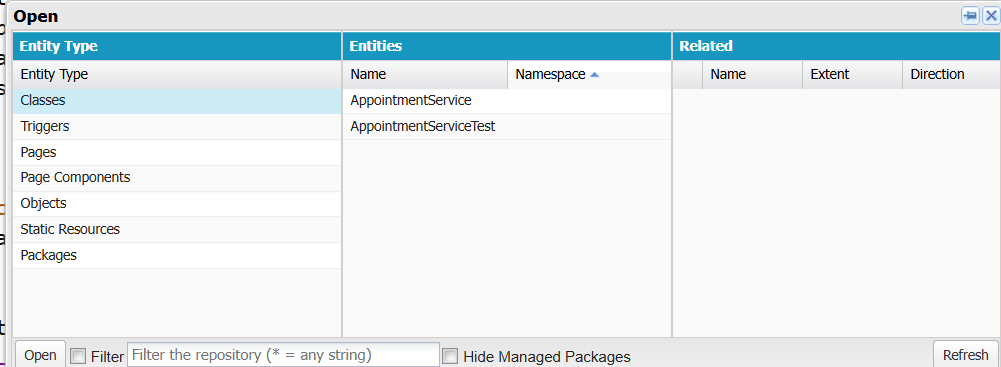
1. **Classes & Objects**

**Service Class: AppointmentService.cls**

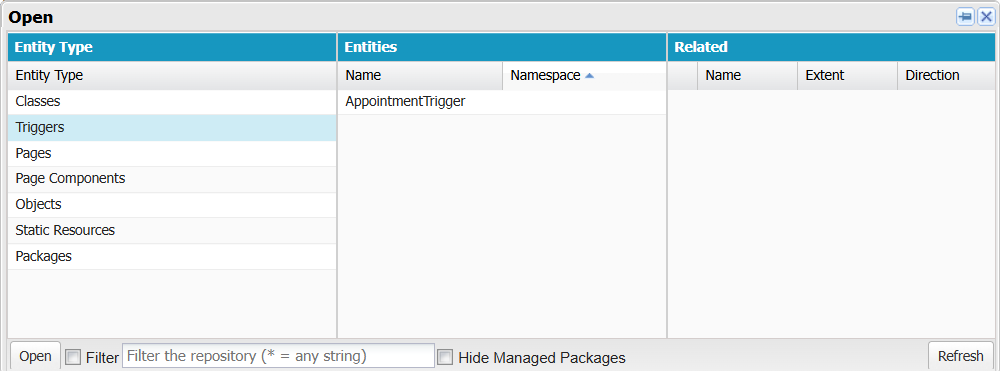
* + This is a **helper class** that holds all the business logic for appointments.  
    It makes your code **reusable** and keeps your trigger clean.
  + **What it does:**
    1. **scheduleAppointment()** → Sets the status to "Scheduled" (if date is today or future).
    2. **completeAppointment()** → Marks an appointment "Completed" only if the date is today or past.
    3. **cancelAppointment()** → Marks an appointment "Cancelled".

**Test Class: AppointmentServiceTest.cls**

* This is used to **prove that your code works** and to get Salesforce’s required **75%+ coverage**.
* **What it does:**
  1. **Creates test Contacts (customers)** → since Appointment\_\_c links to Customer\_\_c.
  2. **Creates test Appointments** → scheduled for today or future.
  3. Tests:
     + Scheduling an appointment (should be "Scheduled").
     + Completing an appointment (status changes to "Completed").
     + Cancelling an appointment (status changes to "Cancelled").



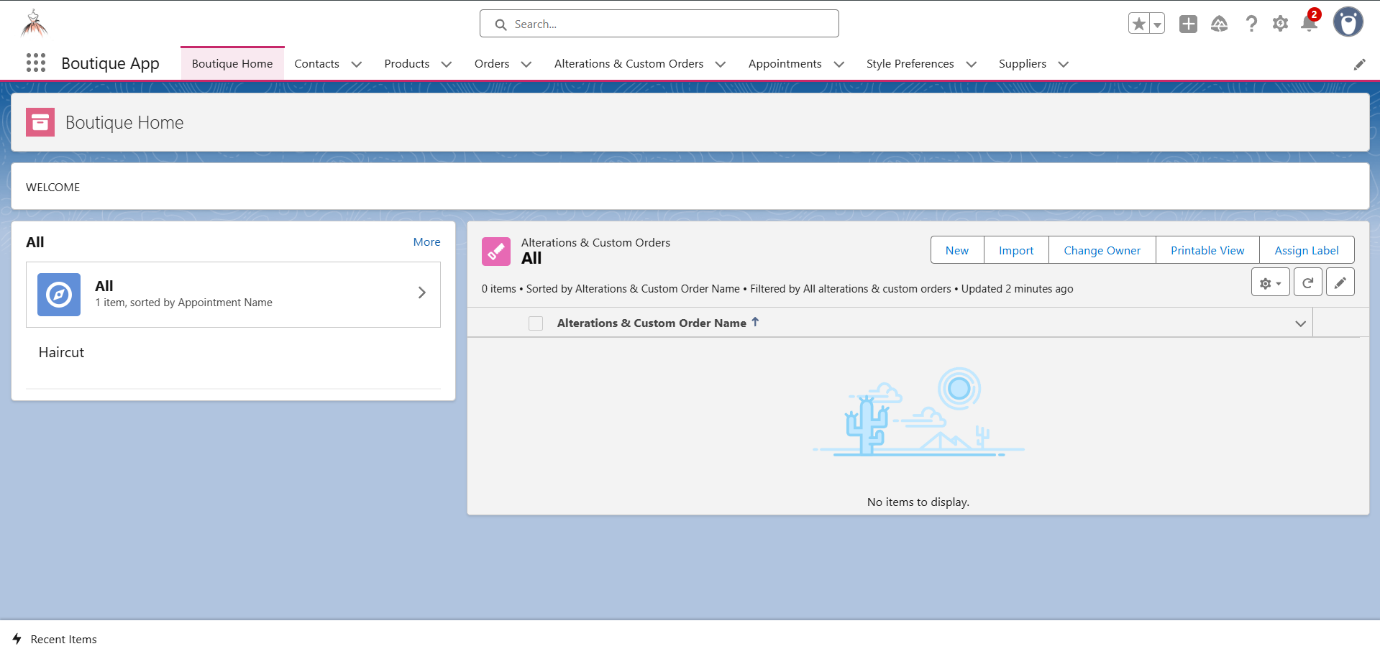
1. **Apex Triggers  
     
   Trigger: AppointmentTrigger**
   * This runs automatically when **Appointment\_\_c** records are inserted or updated.
   * **What it does:**
     1. **On Insert (new appointment):**  
        Calls scheduleAppointment() → ensures new appointments always start with correct status.
     2. **On Update:**
        + If status changed to "Completed" → calls completeAppointment().
        + If status changed to "Cancelled" → calls cancelAppointment().



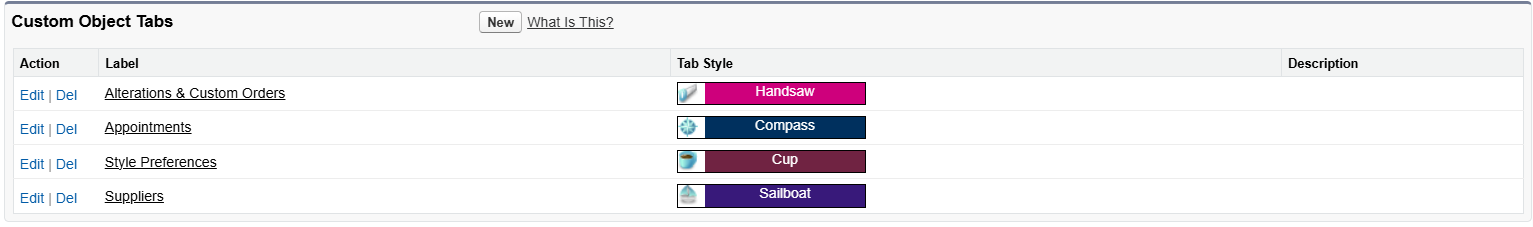
**Phase 6: User Interface Development**

Goal: Make it user-friendly.

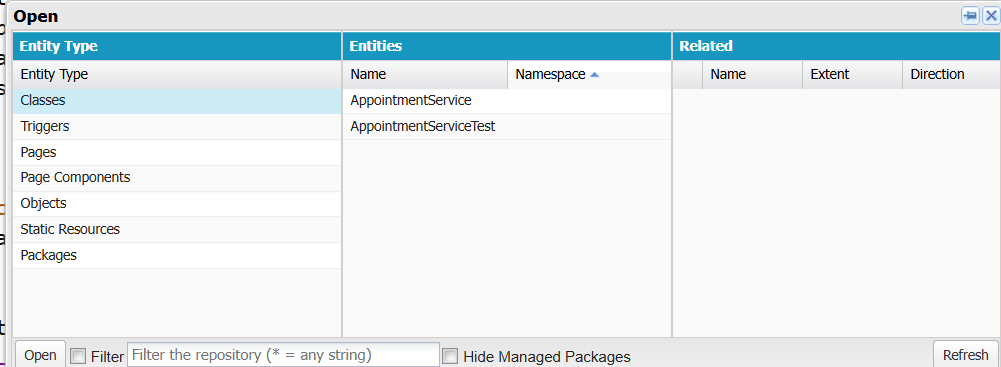
1. **Lightning App Builder**
   * A boutique App is created with the Lightning Web Component for the handling of various tasks and customers in a small business.

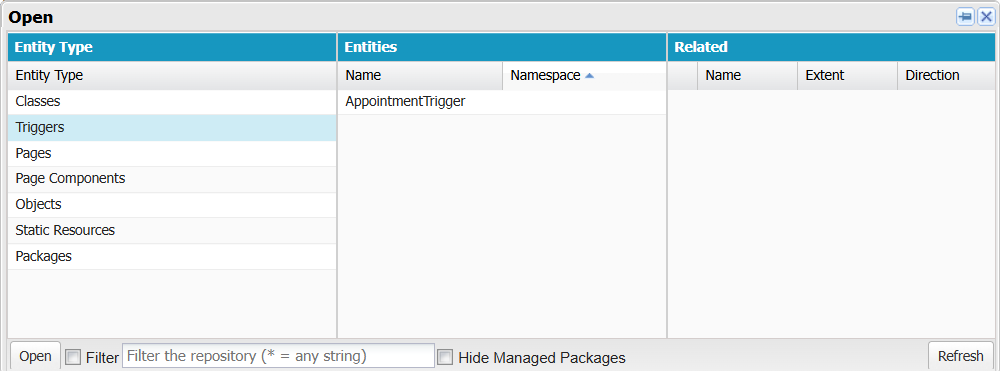


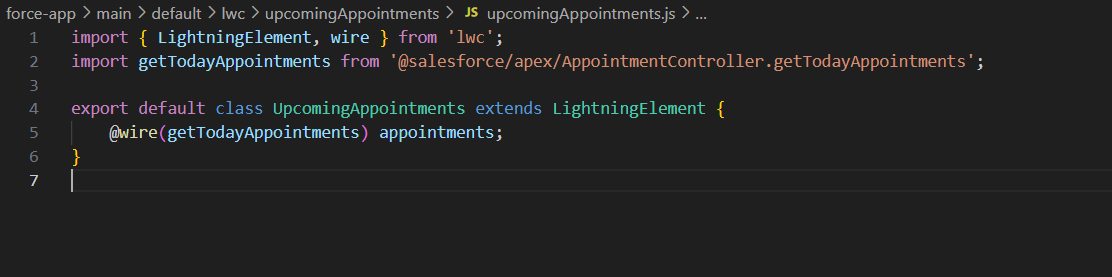
1. **Tabs**
   * Various tabs for the custom objects are added in the app for seamless use by the owner.

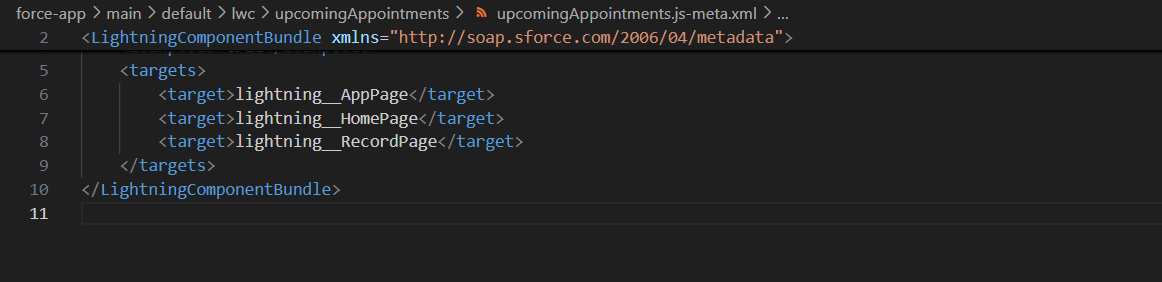


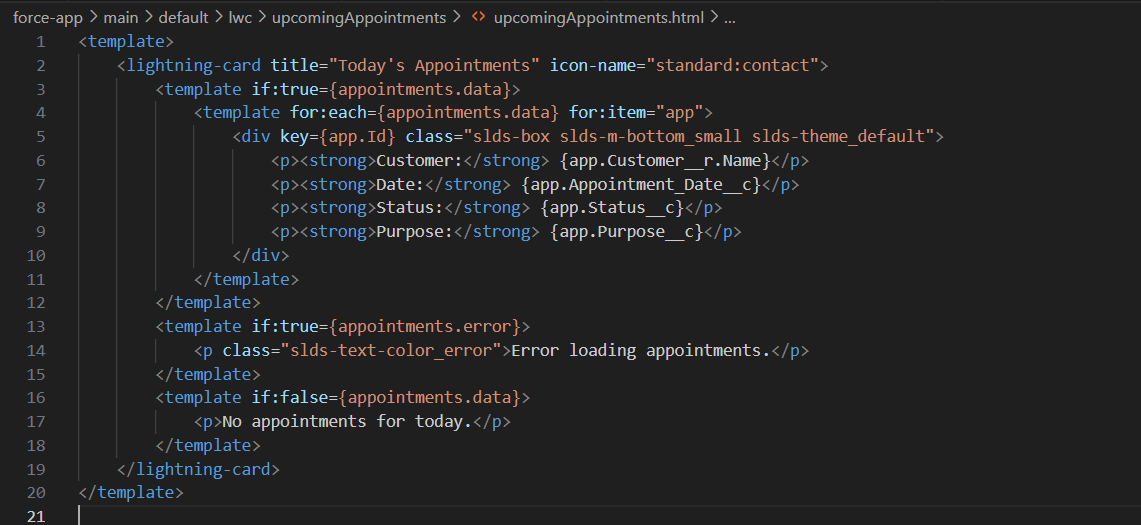
1. **Apex with LWC**
   * Apex is used with lightning web component in integration with vs code for the better use of the application.

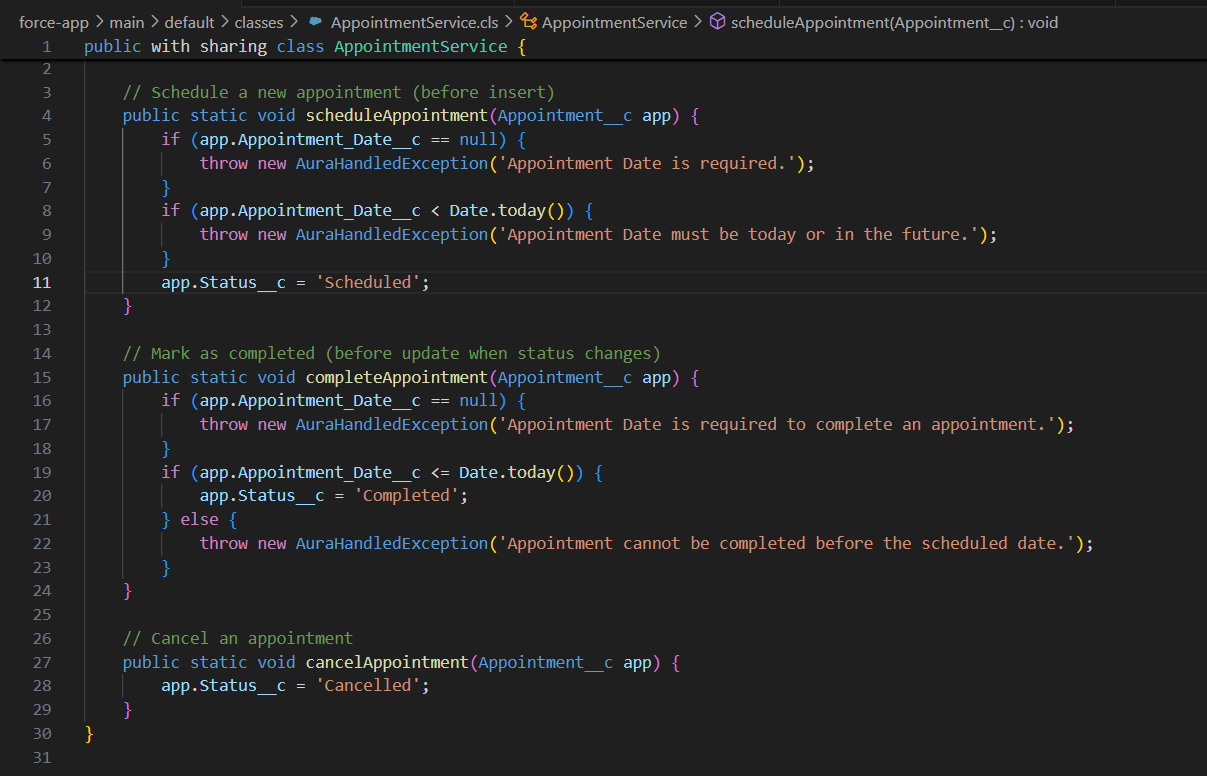


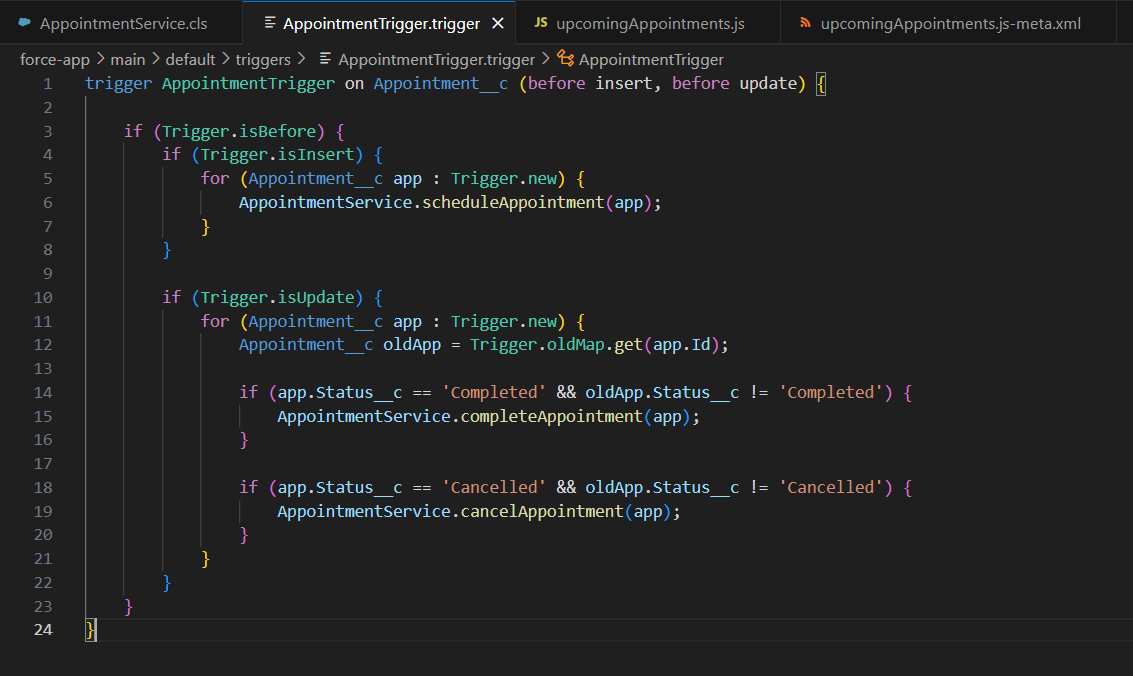




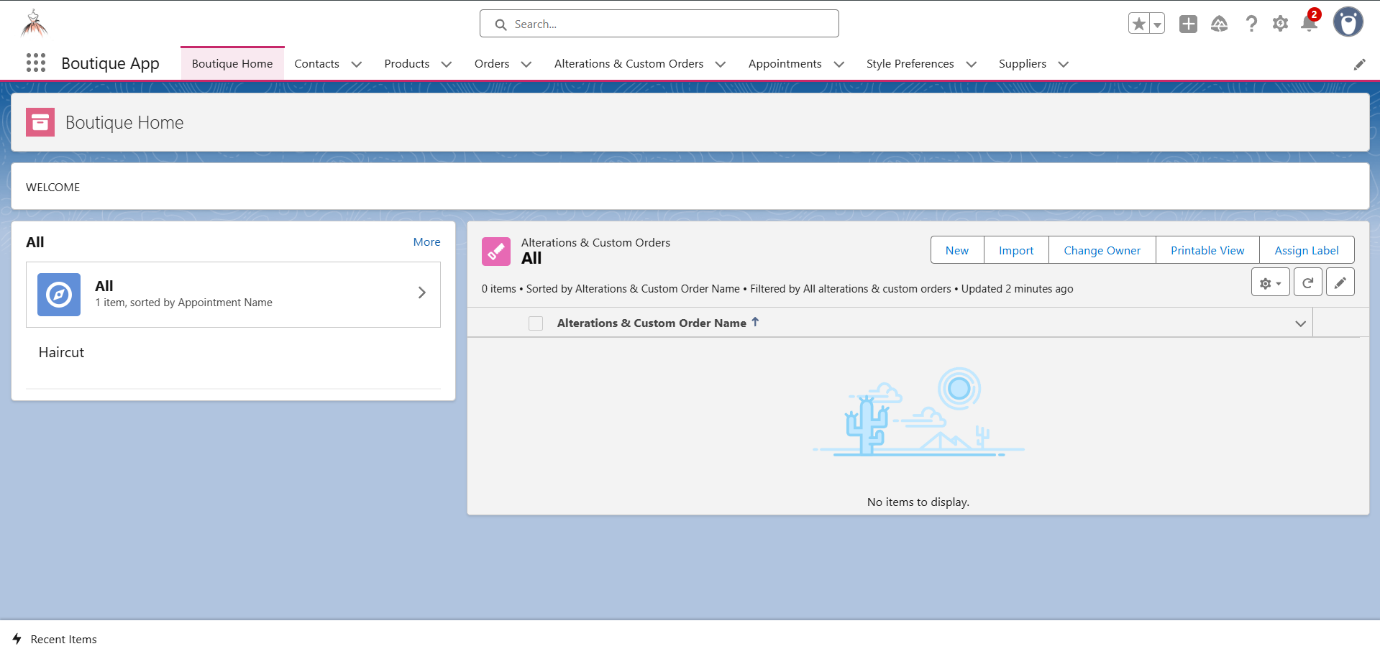








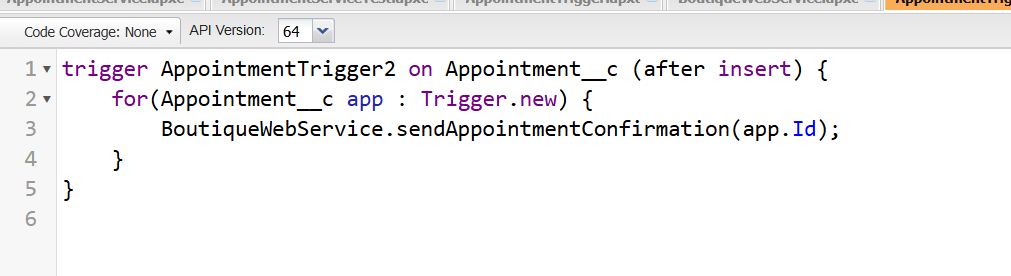
1. **Utility Bar**
   * Quick ‘Recent Items’ action for easy access by the owner.



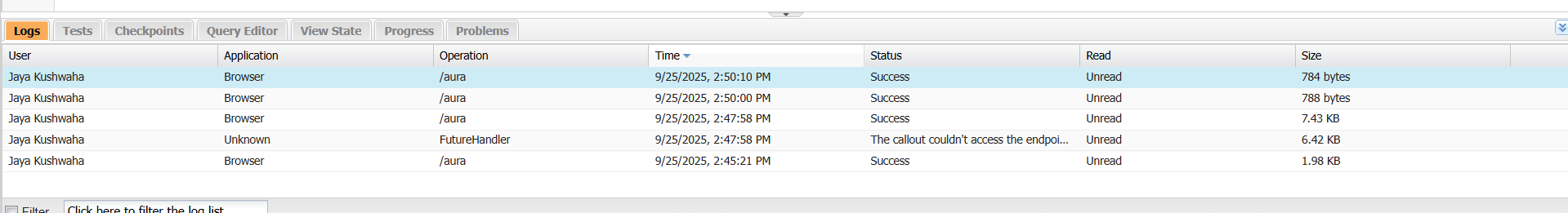
**Phase 7: Integration & External Access**

Goal: Connect with outside systems.

1. **Named Credentials**
   * Store insurance API credentials securely.
2. **External Services**
   * Connect Salesforce to external systems like **email services** using **API**. Also lets us use feature like email when an appointment is created. In this, I’ve use REST API to notify the customer whenever the appointment is created.



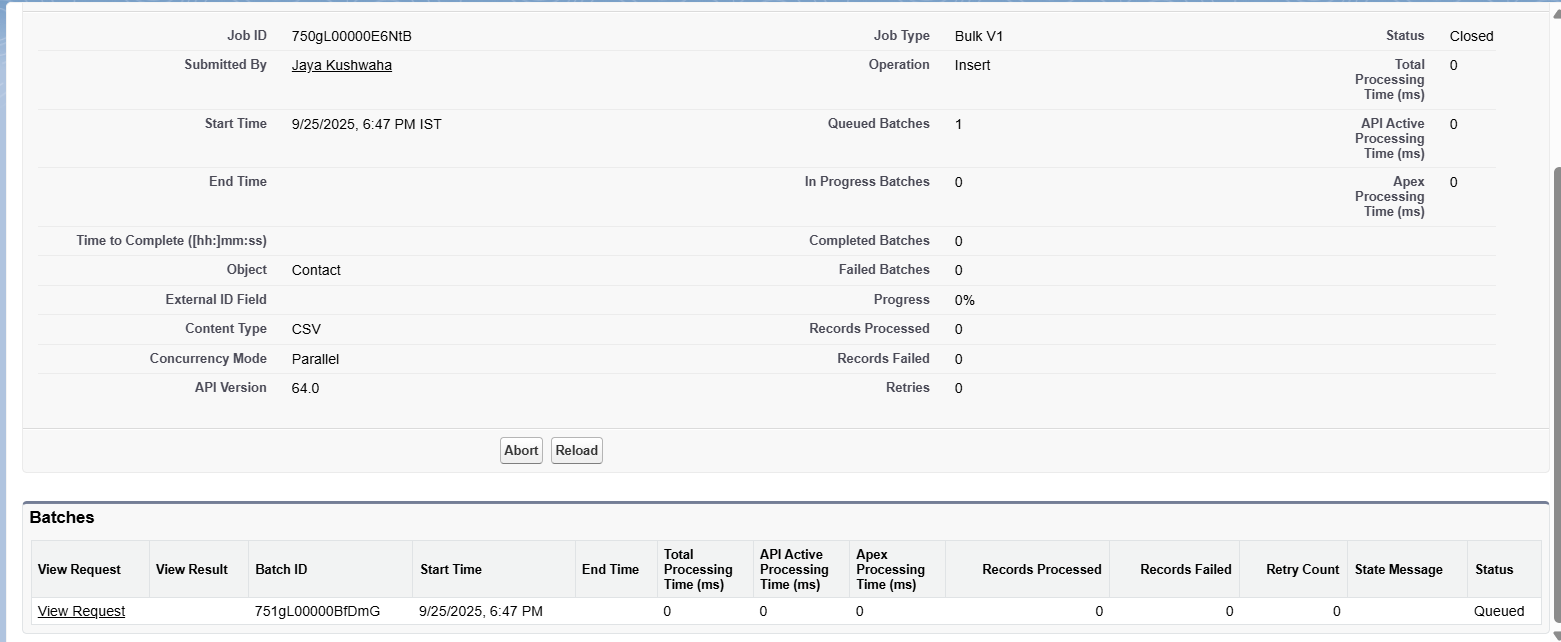




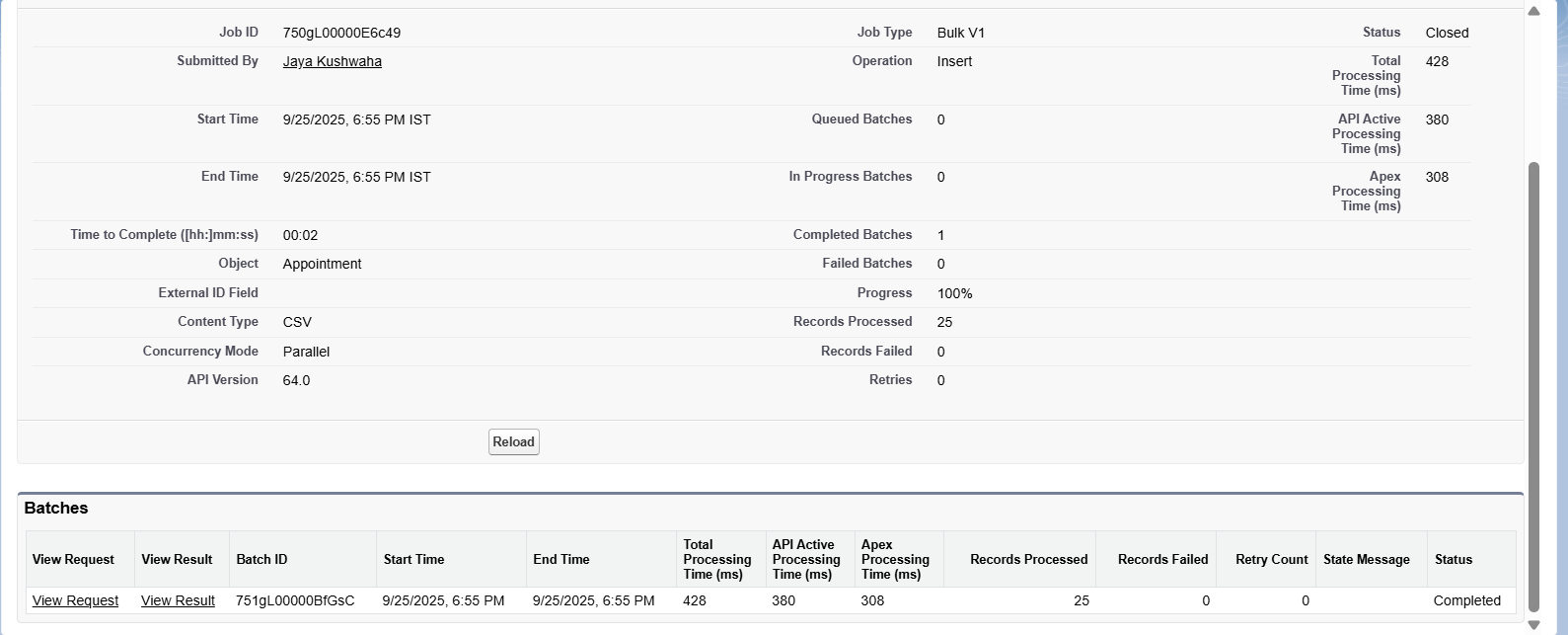
**Phase 8: Data Management & Deployment**

Goal: Manage data and move changes.

1. **Data Import Wizard**
   * Import 50 demo Customer records from the Data Import Wizard.



1. **Data Loader**
   * Import bulk Appointments from the Data Import Wizard.



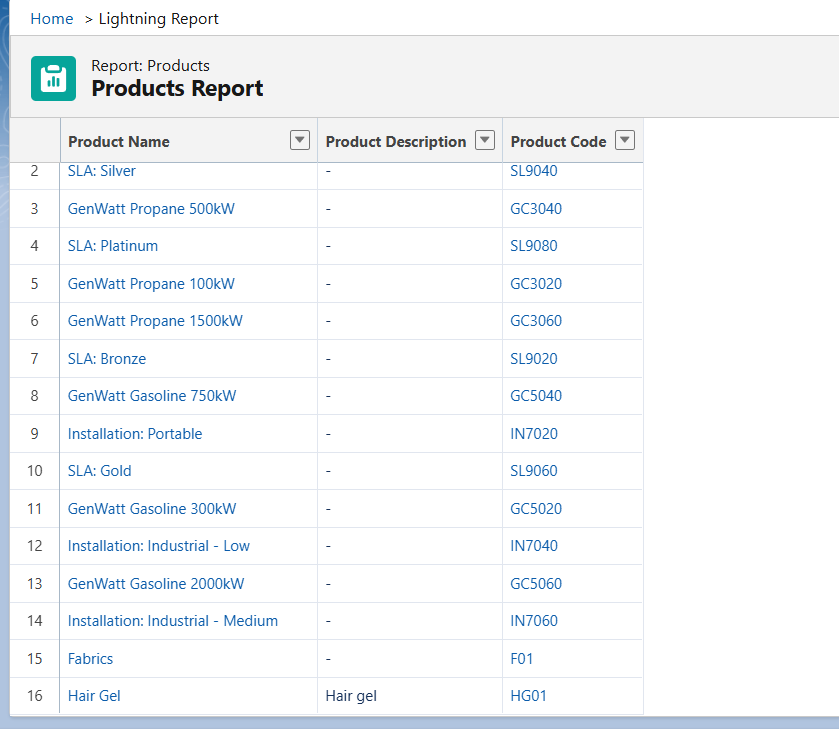
1. **VS Code & SFDX**

Dev-friendly deployments and user-friendly platforms such as VS code is used to integrate the apex programming and other work into the application so that it is seamless.

**Phase 9: Reporting, Dashboards & Security Review**

Goal: Monitor business & secure data.

1. **Reports**Given is a report that shows the products in stock.



1. **Dashboards**

Dashboard for product availability is created for easy access of the details provided in the report.

