Project Documentation: Residential Maintenance Request Portal

- Author: Geddamuri Manasa
- System: Salesforce Lightning Platform

1. Executive Summary

1.1. The Problem: The original process for managing customer complaints was manual, relying on disconnected spreadsheets and emails. This led to significant delays in resolution, a lack of visibility for service managers, and an inefficient workflow for service agents, often resulting in duplicate complaints and poor customer satisfaction.

1.2. The Solution: A centralized, custom Salesforce application, the **Retail Customer Complaint Hub**, was developed. The solution automates the entire lifecycle of a customer complaint, from submission and automated assignment to resolution, manager approvals for high-value compensations, and automated email notifications.

1.3. The Business Value: This solution provides significantly faster complaint response times and improves customer retention. It ensures data integrity by preventing duplicate complaints ,automates key communications with customers through email alerts , and delivers powerful analytics through reports and dashboards for data-driven decision-making by management.

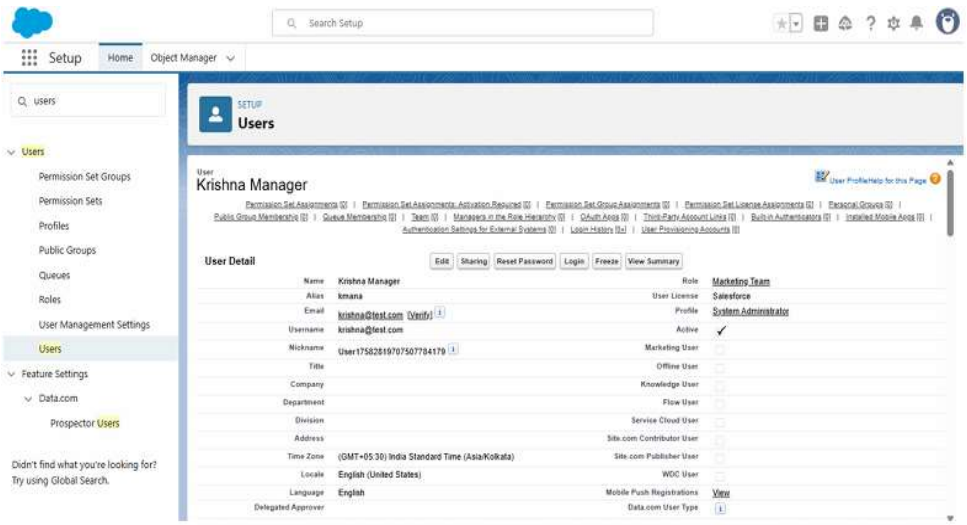
2. Org Setup & Configuration

2.1.Company Profile: The organization's default time zone and currency were set to ensure accurate tracking of complaint resolution times and associated costs.

2.2.Business Hours & Holidays: Specific "Customer Support Hours" and company holidays were defined to enable accurate Service Level Agreement (SLA) calculations.

2.3.User Creation: User accounts were created for the primary stakeholders: a "Support Manager" (Sarah Manager) and a "Customer Service Agent" (Amit Agent).

2.4.Custom Profiles: To manage permissions, two custom profiles were created: "Support Manager" and "Customer Service Agent." These profiles were assigned to the respective users.



3. Data Model& Relationships

This phase involved building the core data structure for the **Retail Customer Complaint Hub**. This was accomplished by creating two custom objects to store information and then adding custom fields and relationships to link them together.

3.1.Key Components Created:

Product Object:

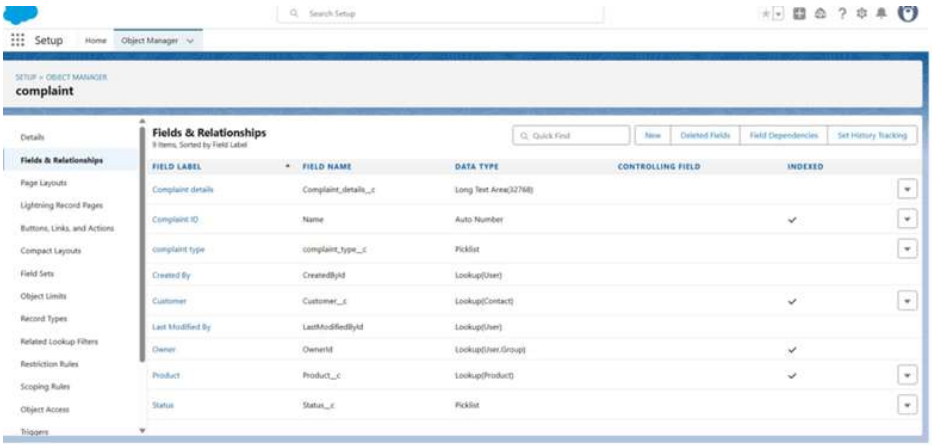
A custom object named Product was created to hold a catalog of all retail products.

A key field, SKU, was added to serve as a unique product identifier for future integrations.

Complaint Object:

A central custom object named Complaint was created to track every customer complaint from submission to resolution.

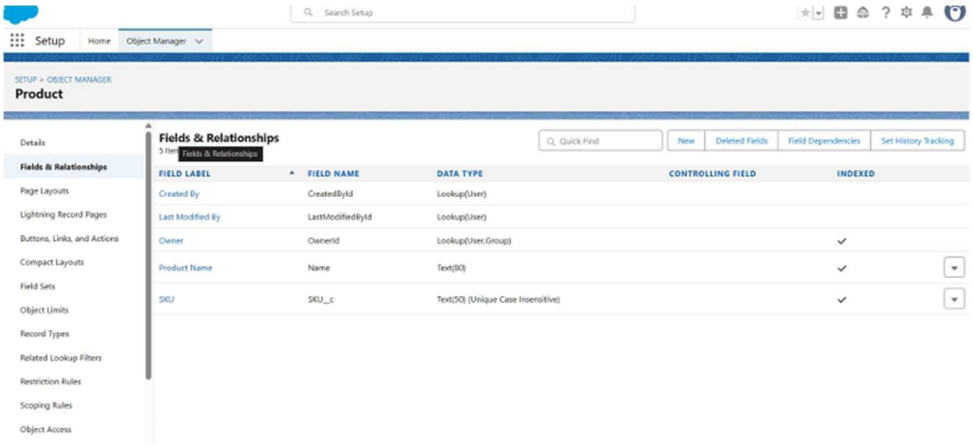
A **lookup relationship** to the Contact object was created to link each complaint to a specific customer.



A **lookup relationship** to the Product object was created to link each complaint to a specific product.

Custom **picklist fields** were added to track the Complaint Type (e.g., "Product Defect," "Service Issue") and Status (e.g., "New," "Resolved," "Closed").

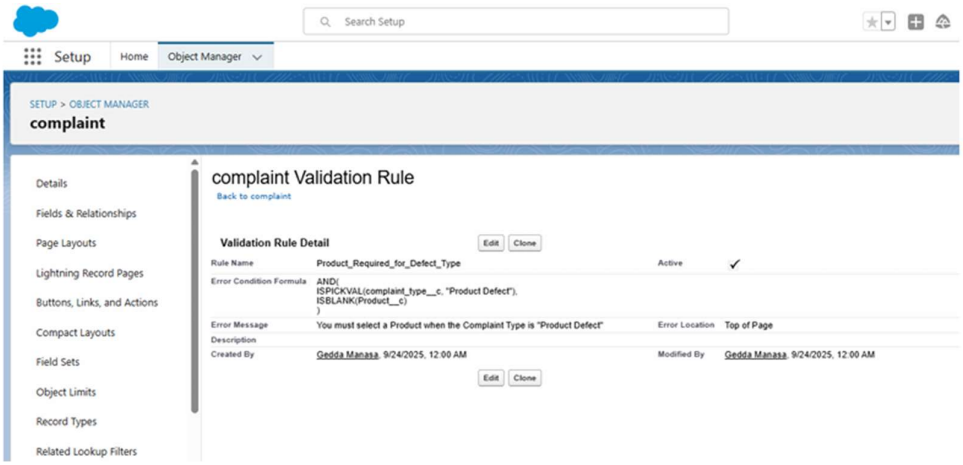
A **long text area** field, Complaint Details, was added to capture detailed notes about the issue.



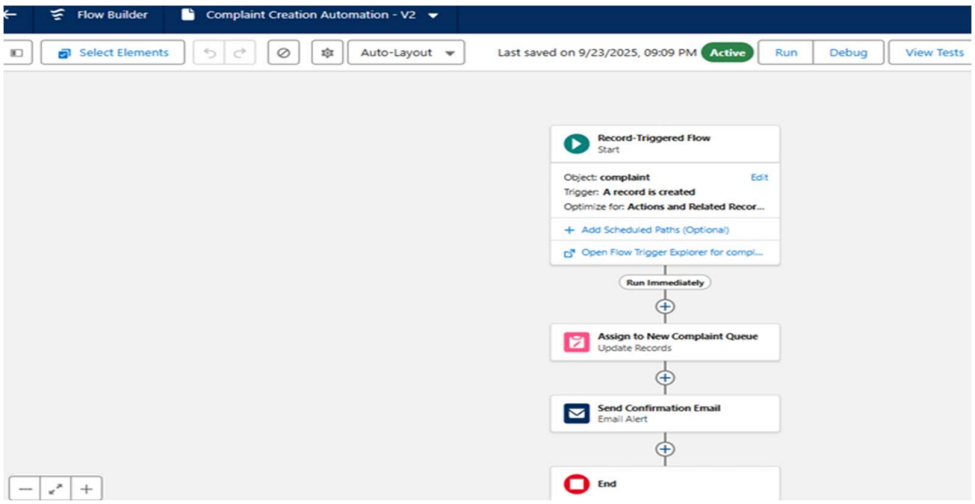
4. Process Automation

This phase made the **Retail Customer Complaint Hub** functional by automating the key business processes for managing customer complaints.

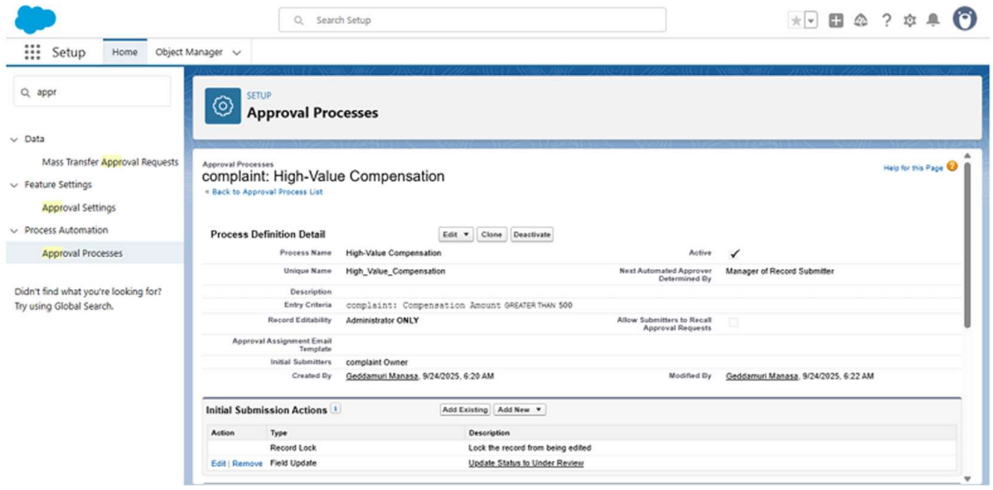
4.1.To ensure data quality, a **Validation Rule** was created. This rule prevents users from saving a complaint with a "Product Defect" type unless a specific product is also selected, ensuring that product-related issues are properly tracked.



4.2.A record-triggered **Flow** named "Complaint Creation Automation" was built to streamline the initial handling of new cases. When a new complaint is created, the flow automatically assigns it to the "New Complaint Queue" for an agent to pick up and simultaneously sends a confirmation email to the customer, letting them know their issue has been logged.

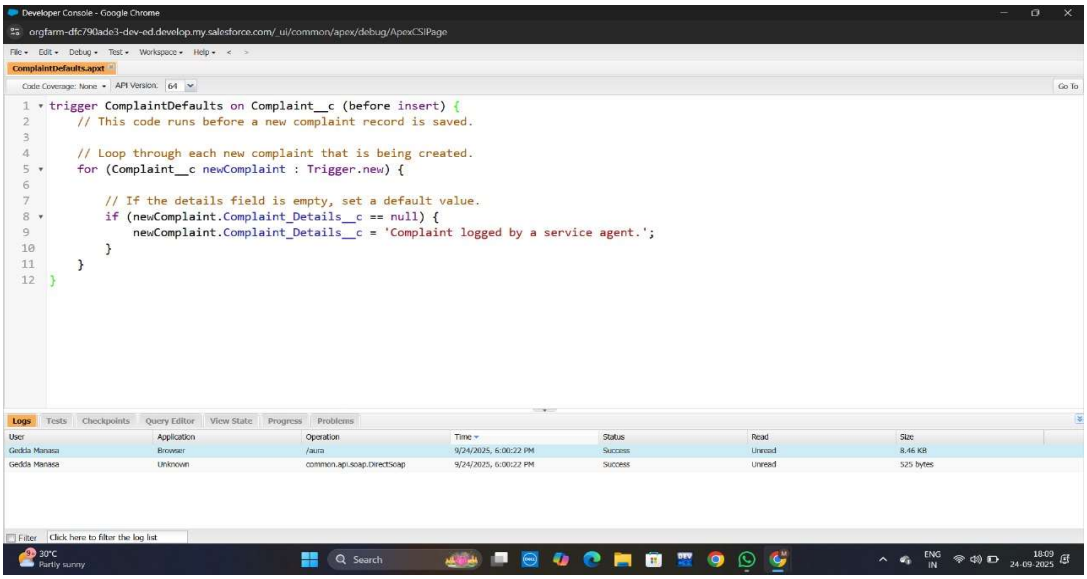


4.3. Finally, an **Approval Process** was configured for high-value compensations. If a complaint's compensation amount is greater than 500, the process can be initiated, which automatically routes the request to the submitter's manager for review. Upon submission, the complaint's status is updated to "Under Review" to reflect its current state.



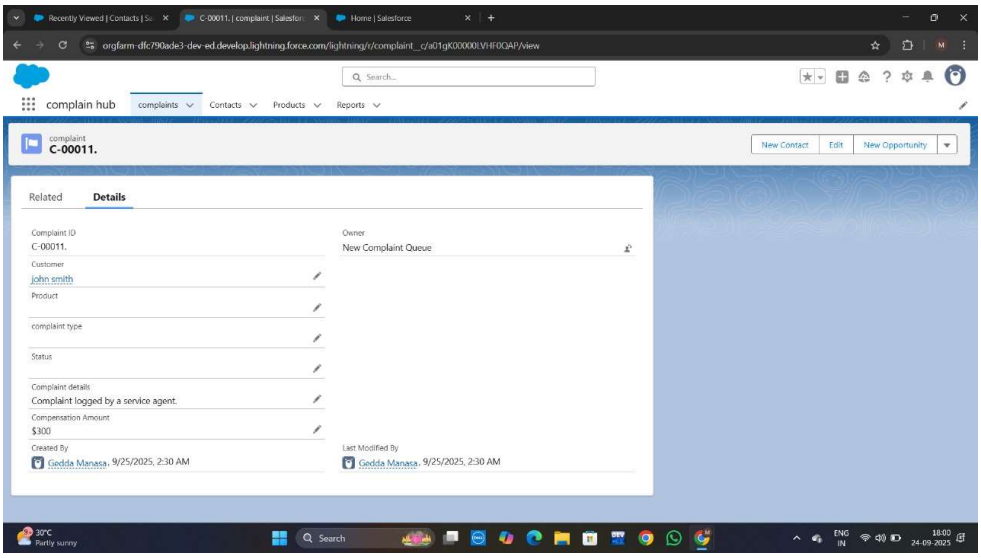
5.Apex Programming

5.1.Retail Customer Complaint Hub using Apex, Salesforce's native programming language. A basic Apex Trigger named "ComplaintDefaults" was created to enhance the application's functionality by ensuring every new complaint record has an initial note, even if a user does not enter one manually.



5.2. The trigger is configured to run before insert, meaning the code executes just before a new complaint record is saved to the database. It checks if the "Complaint Details" field is empty and, if it is, automatically populates it with the default text: "Complaint logged by a service agent."

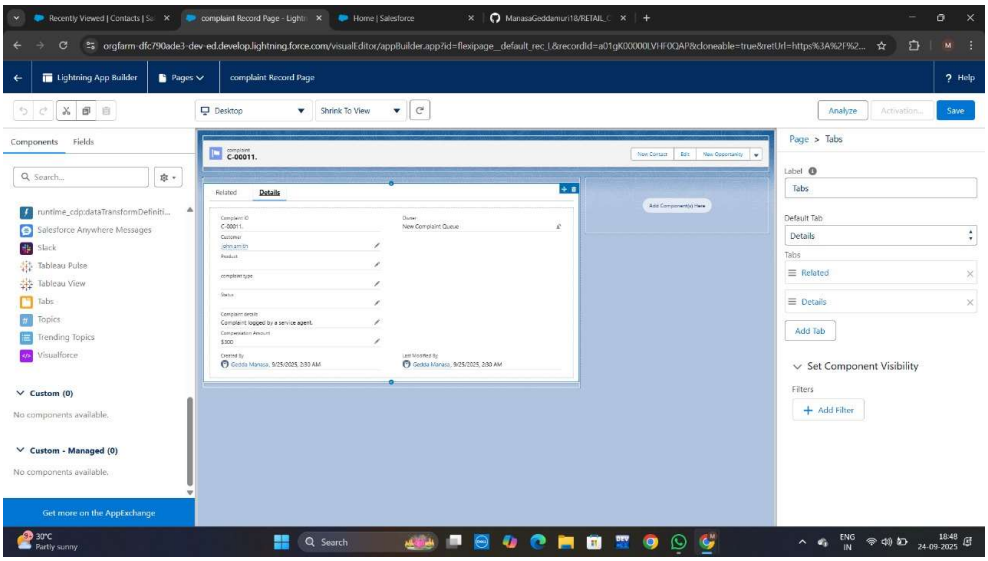
5.3. The functionality was successfully tested by creating a new complaint and leaving the details field blank. Upon saving, the field was automatically populated by the Apex code, confirming that the trigger works as expected. This completes the basic developer customization requirement for the project.



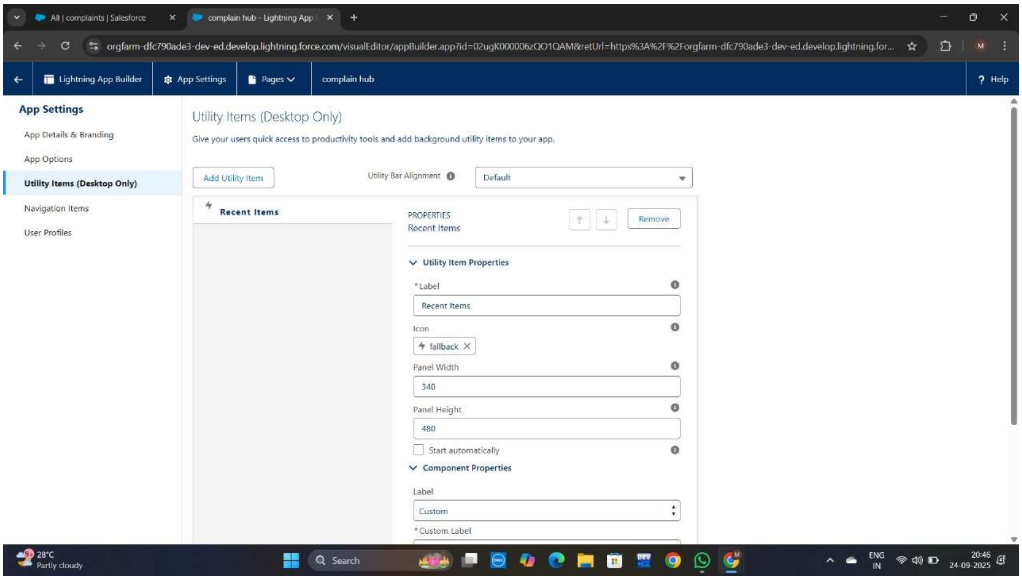
6. User Interface Development

6.1. The standard Complaint record page was customized by adding a Tabs component, which separates the core record details from related lists to make finding information faster and more intuitive. A custom

home page was also created and assigned specifically to the Complaint Hub app, providing agents with a relevant landing page that includes a List View of new complaints, acting as a "to-do" list for the team.

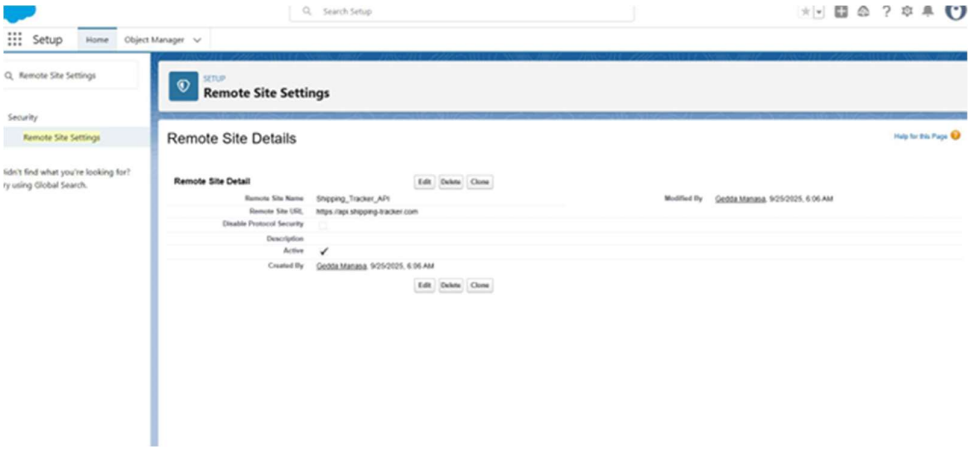


6.2. Finally, a Utility Bar was added to the application to provide quick access to common tools. The Recent Items component was included, allowing agents to easily access their recently viewed records from a persistent footer. These customizations make the user interface more streamlined and tailored to the needs of the customer service team, completing the UI development phase of the project.



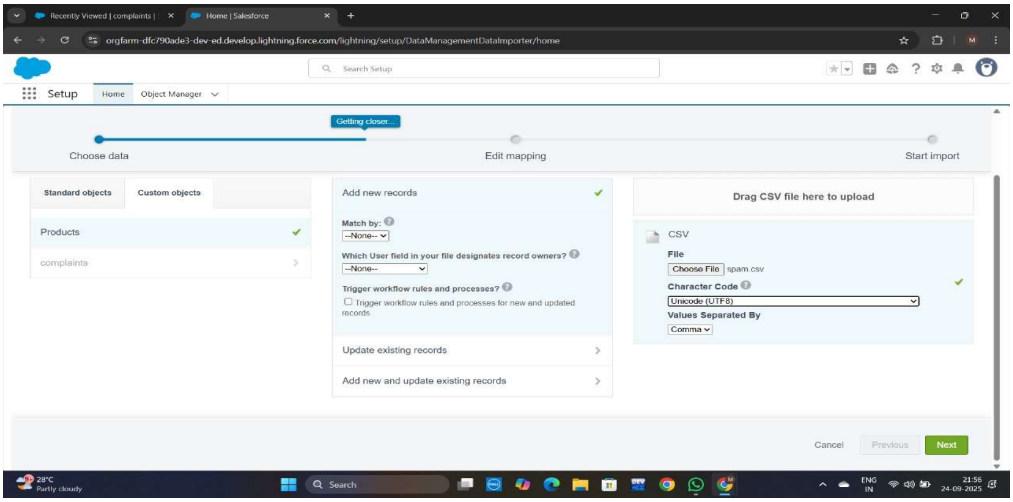
7: Integration & External Access

7.1. Remote Site Setting for API Integrationwas created to prepare the application for a future integration with a hypothetical shipping company's tracking API. By default, Salesforce's security model blocks Apex code from sending data to unknown websites. This setting acts as a "trusted list," telling Salesforce that it is safe to make a connection to a specific URL. Without this administrative setup, any developer code attempting a callout would fail. A new Remote Site Setting was created to authorize future connections to the https://api.shipping-tracker.com endpoint.

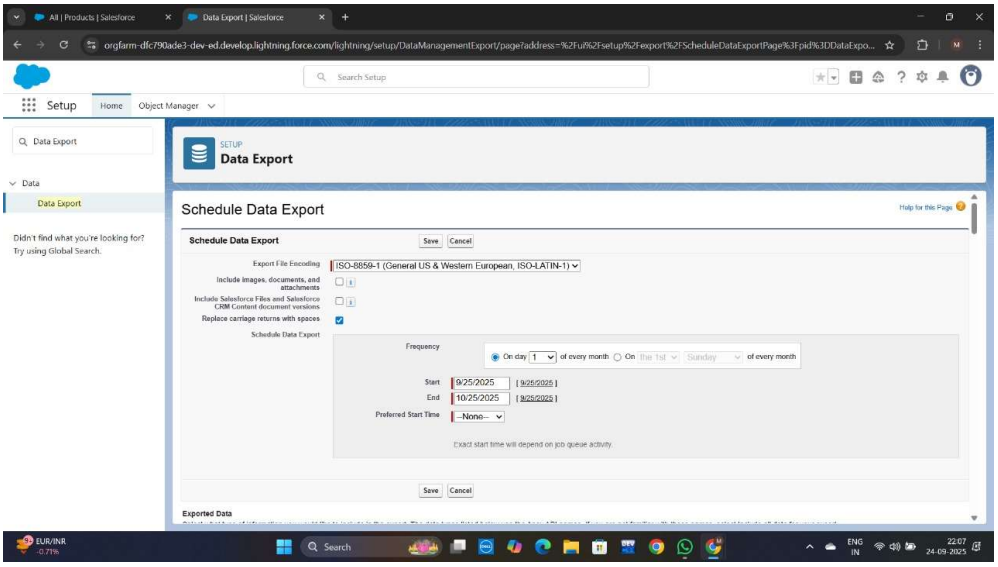


8.Data Management & Deployment

8.1. Product records at once, the **Data Import Wizard** was used. A CSV spreadsheet file containing new product data was prepared and uploaded into the system. The successful completion of this task was monitored via the "Bulk Data Load Jobs" page and verified by confirming that the new products appeared in the "Products" tab list view.

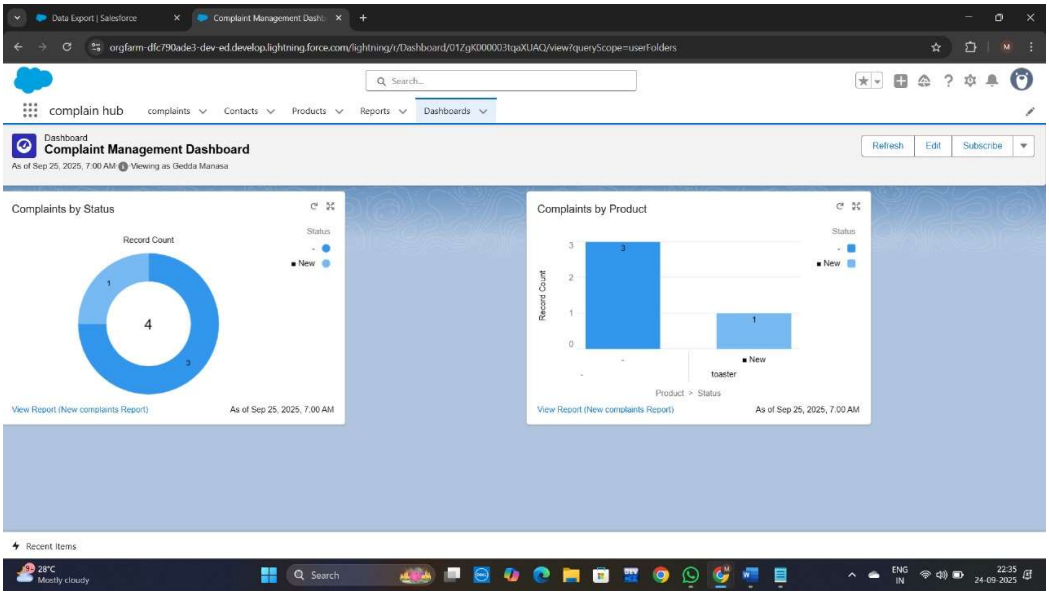


8.2.Finally, to ensure data protection, the standard **Data Export** service was configured. This service was set up to perform an automated, weekly backup of all data within the Salesforce org. These tasks completed the data management phase of the project, ensuring the application can be populated efficiently and that a regular backup schedule is in place.



9. Reporting, Dashboards & Security Review

9.1. To fulfill the project's reporting requirements, custom reports and a central dashboard were created to visualize complaint data for management. These reports were built on the Complaint object to analyze data by different criteria, such as the status of the complaint, providing the support team with an immediate understanding of their workload. A central dashboard was then created to give managers an at-a-glance view of key metrics by displaying components from the custom reports.



In addition, a security review was performed to demonstrate how to control data access at a granular level. This included reviewing profile permissions to ensure that users have the appropriate level of access (Read, Create, Edit, Delete) to the custom complaints and Products objects based on their roles.

9.2. The **Setup Audit Trail** was also reviewed as a key tool for monitoring all recent administrative changes made to the organization during the project's development, providing accountability and a history of modifications. This completed the project's reporting and security review requirements.