

Complete Project Documentation

Phase 1: Problem Understanding & Industry Analysis

This phase sets the **foundation** for your CRM project. The goal is to deeply understand the **problem space**, analyze the **industry requirements**, and align the CRM solution with **stakeholders' expectations**.

◆ 1. Requirement Gathering

What it means:

Requirement gathering is the process of collecting, analyzing, and documenting the needs of the business and end-users. It helps define *what* the CRM system must achieve.

How to do it:

- Conduct **interviews** with stakeholders (customers, agents, managers).
- Send out **surveys** to understand common pain points in complaint handling.
- Analyze **existing complaint logs** (emails, Excel sheets, or manual registers).

Key questions to ask:

- How are customer complaints currently tracked?
- What challenges do agents face in resolving complaints?
- What reports do managers want to see?
- Do customers get regular updates on their complaints?

Example requirements for Customer Complaint CRM:

- Customers should be able to log complaints through **website, email, or phone**.
- Agents should be able to **view, assign, and close complaints**.
- Managers should be able to **analyze complaint trends and agent performance**.
- System should send **automatic notifications** when complaint status changes.

Deliverable:

A Requirement Specification Document with **must-have features (core)** and **nice-to-have features (future scope)**.

◆ **2. Stakeholder Analysis**

What it means:

Identifying the key people (stakeholders) who will be impacted by the CRM and understanding their expectations.

Stakeholder Categories for this project:

1. **Customers (External Stakeholders)**

- Role: Log complaints and track their resolution.
- Expectation: Fast response, transparency, updates.

2. **Support Agents (Internal Stakeholders)**

- Role: Handle and resolve complaints.
- Expectation: Easy access to customer data, clear case assignment, reduced manual effort.

3. **Managers/Supervisors**

- Role: Monitor agent performance, oversee complaint resolution.
- Expectation: Dashboards, SLA compliance tracking, escalations.

4. **System Administrators**

- Role: Configure Salesforce, manage user roles, maintain security.
- Expectation: Customization flexibility, easy maintenance.

5. Top Management

- Role: Ensure customer satisfaction and business growth.
- Expectation: Reports to measure overall complaint trends and decision-making insights.

Deliverable:

A **Stakeholder Map** (a table or diagram with roles, responsibilities, and expectations).

◆ 3. Business Process Mapping

What it means:

Visualizing how complaints are currently handled (**AS-IS process**) and how the new CRM will improve it (**TO-BE process**).

AS-IS Example (Manual Process):

1. Customer calls customer care → details written on paper.
2. Complaint forwarded via email/WhatsApp.
3. No tracking system → customer keeps following up.
4. Resolution happens but no record of timelines or satisfaction.

TO-BE Example (CRM-enabled Process in Salesforce):

1. Customer submits complaint via **web portal/email/chatbot**.
2. Salesforce automatically creates a **case record**.
3. Case is **auto-assigned** to the right agent (based on skill/rules).
4. Agent updates case progress in real-time.

5. Customer gets **automated notifications** (case created, in-progress, resolved).
6. Managers view **dashboards** for trends, escalations, SLA breaches.

Deliverable:

Business Process Diagrams:

- **AS-IS model** (manual flow)
 - **TO-BE model** (Salesforce-enabled flow)
-

◆ 4. Industry-Specific Use Case Analysis

What it means:

Every industry faces different kinds of customer complaints. Aligning your CRM to one industry makes the project realistic.

Example: Car Rental Industry (if you choose this):

- Common Complaints:
 - Vehicle not available on booking.
 - Poor vehicle condition.
 - Overbilling/hidden charges.
 - Late pickup/drop-off.
 - Refund delays.
- CRM Use Cases:
 - Auto-logging complaints directly from the booking system.
 - Assigning complaints to specialized teams (billing, maintenance, scheduling).
 - Tracking SLAs (e.g., refund within 7 business days).
 - Manager dashboards to identify frequent complaint categories.

Other industries you could consider:

- **Banking:** Failed transactions, fraud reports.
- **Telecom:** Network downtime, billing errors.
- **Retail/E-commerce:** Product quality, delivery issues.

Deliverable:

An **Industry Use Case Document** describing complaint categories and CRM solutions.

◆ **5. AppExchange Exploration**

What it means:

Salesforce AppExchange is like an **app store** where you can find ready-made solutions. Exploring it helps you decide whether to build from scratch or enhance existing apps.

How to explore:

1. Go to [AppExchange](#).
2. Search for: *Complaint Management, Case Management, Customer Service*.
3. Review apps like:
 - **Salesforce Service Cloud Extensions** (case management add-ons).
 - **Complaint Management Apps** (industry-specific).
 - **Chatbots/AI Tools** (for customer service automation).

Deliverable:

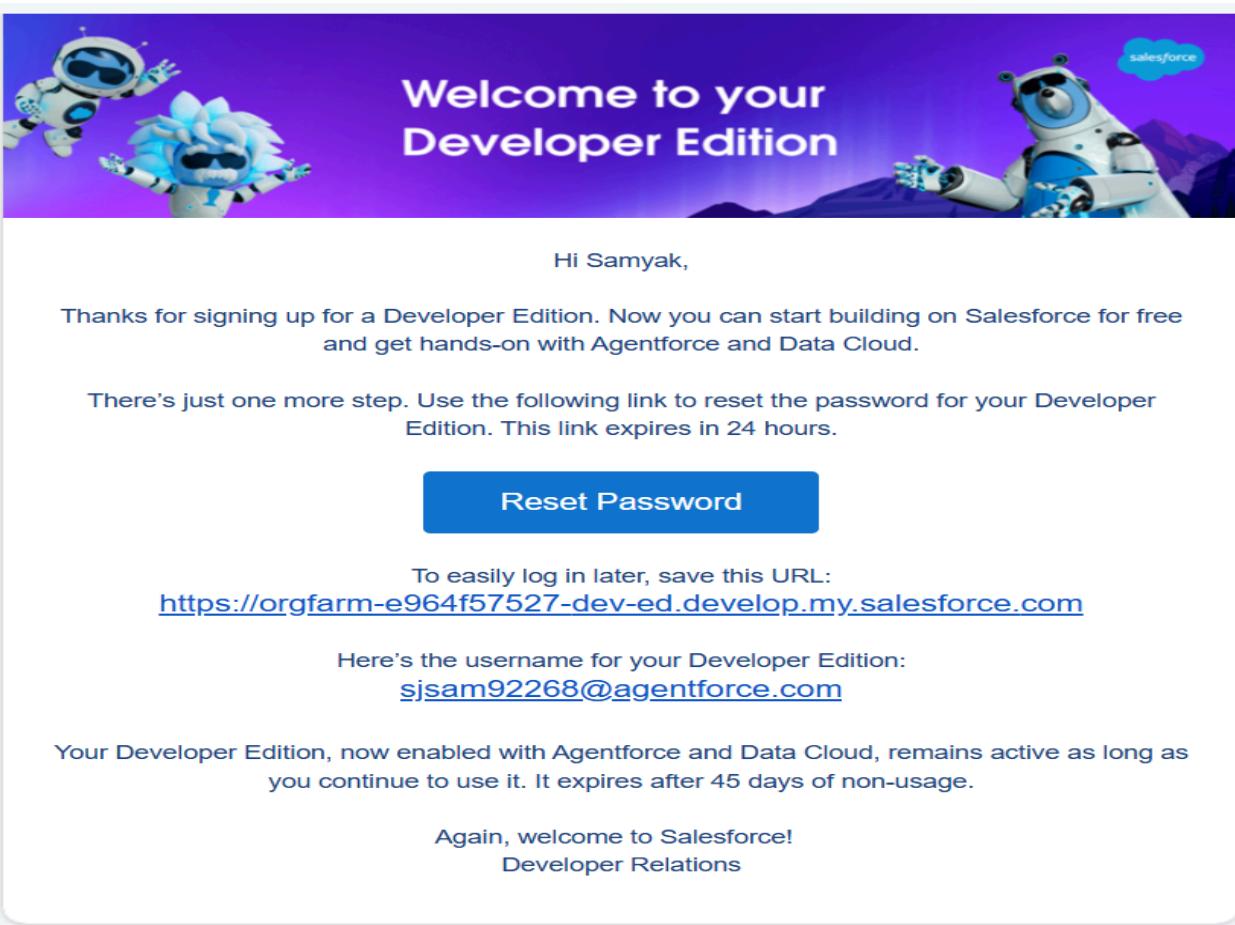
A **comparison table** of top 2–3 apps with features, pros/cons, and decision whether to use them or not.

📍 Phase 2: Org Setup & Configuration

◆ 1. Salesforce Edition – Developer Org

We created the main objects needed for the CRM: **Complaint**, **Feedback**, and **Activity** (if needed). Each object serves a distinct purpose—tracking complaints, storing feedback, and logging related activities.

- **Why:** Developer Edition is free and gives you all core features.
- **How:**
 - Go to developer.salesforce.com.
 - Sign up with your email.
 - Confirm the activation email → log in.



The image shows a screenshot of an email from Salesforce. The subject line is "Welcome to your Developer Edition". The email is addressed to "Hi Samyak," and thanks the recipient for signing up. It includes a link to reset the password and a URL to easily log in later. It also provides the username "sjsam92268@agentforce.com". The message is signed off by "Developer Relations".

Welcome to your
Developer Edition

Hi Samyak,

Thanks for signing up for a Developer Edition. Now you can start building on Salesforce for free and get hands-on with Agentforce and Data Cloud.

There's just one more step. Use the following link to reset the password for your Developer Edition. This link expires in 24 hours.

[Reset Password](#)

To easily log in later, save this URL:
<https://orgfarm-e964f57527-dev-ed.develop.my.salesforce.com>

Here's the username for your Developer Edition:
sjsam92268@agentforce.com

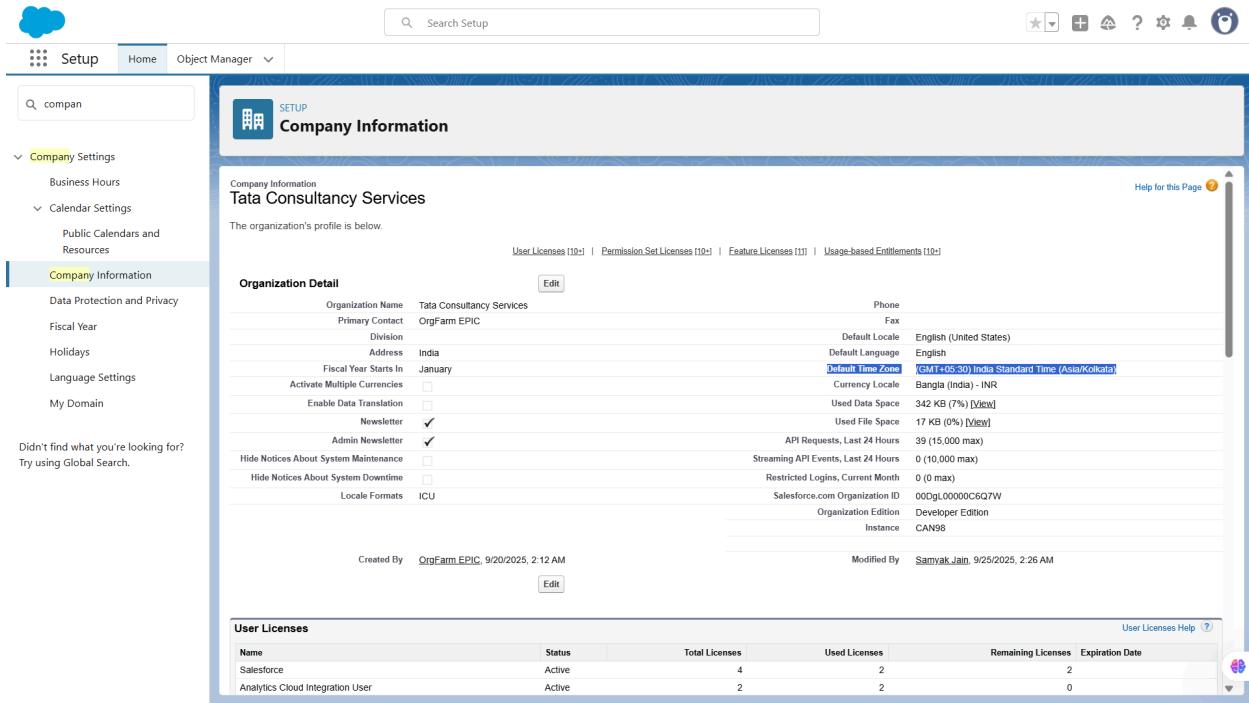
Your Developer Edition, now enabled with Agentforce and Data Cloud, remains active as long as you continue to use it. It expires after 45 days of non-usage.

Again, welcome to Salesforce!
Developer Relations

◆ 2. Company Settings – Local Time Zone & Currency

For each object, we added the necessary **custom fields** like text, picklist, date, and lookup fields. This ensures all relevant information (e.g., complaint subject, status, priority) is captured accurately.

- **Where:**  Setup → Company Information → Edit.
- **Do:**
 - Set **Time Zone** = your local (e.g., IST).
 - Set **Default Currency** = (e.g., INR).
- **Why:** Ensures business hours, reports, and dashboards align with your company.



The screenshot shows the Salesforce Setup interface. The top navigation bar includes 'Setup', 'Home', and 'Object Manager'. A search bar says 'Search Setup'. On the left, a sidebar menu is open under 'Company Settings', showing sections like 'Business Hours', 'Calendar Settings', 'Public Calendars and Resources', and 'Company Information'. The 'Company Information' section is selected and expanded, showing details for 'Tata Consultancy Services'. The main content area displays 'Organization Detail' with fields such as Organization Name (Tata Consultancy Services), Primary Contact (OrgFarm EPIC), Division (India), and Fiscal Year Starts In (January). It also shows 'Default Time Zone' set to '(GMT +03:30) India Standard Time (Asia/Kolkata)'. Below this is a 'User Licenses' table:

Name	Status	Total Licenses	Used Licenses	Remaining Licenses	Expiration Date
Salesforce	Active	4	2	2	
Analytics Cloud Integration User	Active	2	2	0	

◆ 3. Business Hours – 9 AM to 6 PM

We established **lookup and master-detail relationships** to link objects logically. For example, a Feedback record is linked to a Complaint, enabling proper data hierarchy and reporting.

- **Where:** Setup → Business Hours.

- **Do:**
 - Create “Standard Business Hours”.
 - Set **Mon–Fri, 9:00 to 18:00**.
- **Why:** Cases & SLAs will follow these timings.

Business Hours Name	Standard Business Hours	Time Zone																
Business Hours	<table border="1"> <thead> <tr> <th></th> <th>Sunday</th> <th>Monday</th> <th>Tuesday</th> <th>Wednesday</th> <th>Thursday</th> <th>Friday</th> <th>Saturday</th> </tr> </thead> <tbody> <tr> <td></td> <td>No Hours</td> <td>9:00 AM to 6:00 PM</td> <td>No Hours</td> </tr> </tbody> </table>		Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday		No Hours	9:00 AM to 6:00 PM	No Hours	(GMT+05:30) India Standard Time (Asia/Kolkata)				
	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday											
	No Hours	9:00 AM to 6:00 PM	No Hours															

Holidays (0)

Business Hours Detail

Active ✓

Created By Samyak.Jain 9/25/2025, 2:25 AM Last Modified By Samyak.Jain 9/25/2025, 2:49 AM

Holidays

Add/Remove

No records to display

Always show me ▾ more records per related list

◆ 4. Fiscal Year Settings

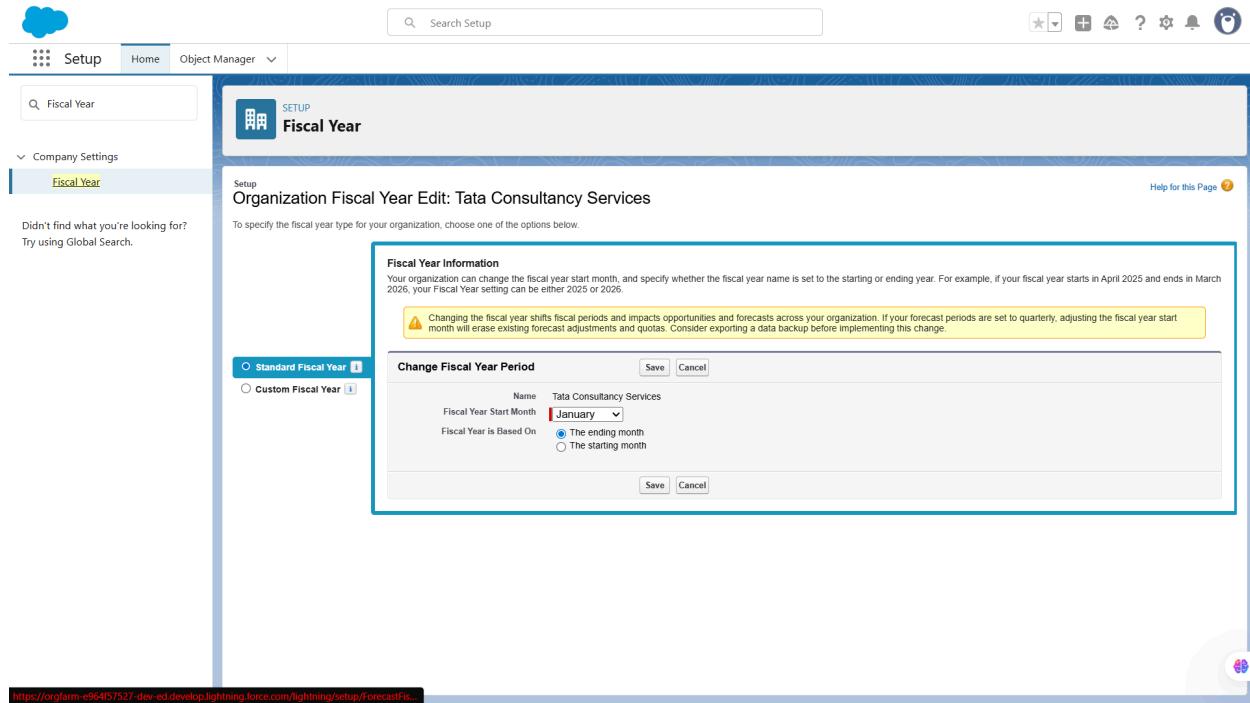
Created rules to **enforce data quality**, such as mandatory fields, date validations, and picklist restrictions. This prevents incomplete or incorrect records from being saved.

Use Case:

- Determines the **financial reporting period**, used in reports and dashboards. Salesforce supports **Standard (Jan–Dec)** or **Custom Fiscal Years**.

Steps to Implement:

1. Setup → Quick Find → **Fiscal Year**.
2. Select **Standard Fiscal Year** or configure **Custom Fiscal Year** if needed.
3. Save.



◆ 5. User Setup & Licenses

Customized page layouts to organize fields and related lists for different users. We also set up **record types** for complaints to handle different processes or categories.

Use Case:

- Users need accounts to access Salesforce. Assigning proper **profiles and licenses** ensures they have the correct access level.

Steps to Implement:

1. Setup → Users → New User.

2. Create:

- **Support Agent** → Profile: Agent Profile.

The screenshot shows the Salesforce Setup interface with the 'Users' tab selected. A new user record is being created with the following details:

User Detail	Value	Role
Name	agent	Salesforce
Alias	agent	Profile
Email	supportagent@gmail.com (Verify)	Minimum Access - Salesforce
Username	agent1@dev.com	Active
Nickname	agent	Marketing User
Title		Offline User
Company		Knowledge User
Department		Flow User
Division		Service Cloud User
Address		Site.com Contributor User
Time Zone	(GMT+05:30) India Standard Time (Asia/Kolkata)	Site.com Publisher User
Locale	English (United States)	WDC User
Language	English	Mobile Push Registrations
Delegated Approver		Data.com User Type
Manager		Accessibility Mode (Classic Only)
Receive Approval Request Emails	Only if I am an approver	Debug Mode
Federation ID		High-Contrast Palette on Charts
App Registration: One-Time Password Authenticator		Load Lightning Pages While Scrolling
App Registration: Salesforce Authenticator		Salesforce CRM Content User
Security Key (U2F or WebAuthn)		Receive Salesforce CRM Content Email Alerts
Lightning Login		Receive Salesforce CRM Content Alerts as Daily Digest

- **Manager** → Profile: Manager Profile.

The screenshot shows the Salesforce Setup interface with the 'Users' tab selected. On the left, a sidebar lists categories like Permission Set Groups, Profiles, Public Groups, Roles, and User Management Settings. The main content area is titled 'User Manager' and shows a 'User Detail' section for a user named 'manager'. The user's email is listed as 'manager@dev.com [Verify]'. The 'Role' section indicates the user has the 'Salesforce Solution Manager' license and is 'Active'. Other settings include 'Marketing User', 'Offline User', 'Knowledge User', 'Flow User', 'Service Cloud User', 'Site.com Contributor User', 'WDC User', 'Mobile Push Registrations', 'Data.com User Type', 'Accessibility Mode (Classic Only)', 'Debug Mode', 'High-Contrast Palette on Charts', 'Load Lightning Pages While Scrolling' (which is checked), 'Salesforce CRM Content User', 'Receive Salesforce CRM Content Email Alerts' (checked), and 'Receive Salesforce CRM Content Alerts as Daily Digest' (checked).

3. Assign **licenses** (usually Salesforce Platform or full Salesforce license).

Screenshot Guidance:

- Screenshot of created Users list.
- Screenshot of User detail page showing Profile & Role.

◆ 6. Login Access Policies

Imported sample records using **Data Loader** to test the object structure. This helped us verify relationships, field mappings, and overall data integrity.

Use Case:

- Restrict login access by **IP ranges or business hours** to ensure users can access Salesforce only when permitted.

Steps to Implement:

1. Setup → Profiles → Agent Profile → Login Hours → Restrict to 9 AM – 6 PM, Mon–Fri.

The screenshot shows the Salesforce Setup interface with the following details:

- Header:** Search bar with "Search Setup", a gear icon, and various navigation icons.
- Left Navigation:** "Setup" tab selected, "Profiles" under "Users".
- Page Title:** "SETUP Profiles" with a "Agent Profile" sub-section.
- Section:** "Profile" - "Agent Profile".
- Description:** "Users with this profile have the permissions and page layouts listed below. Administrators can change a user's profile by editing that user's personal information." It notes that if the organization uses Record Types, users can be assigned to one or more record types available to users with this profile.
- Buttons:** "Edit", "Clone", "Delete", "View Users".
- Profile Detail:** Name: "Agent Profile", User License: "Salesforce", Description: "", Created By: "Samyak Jain" (Created on 9/25/2025, 2:58 AM), Modified By: "Samyak Jain" (Modified on 9/25/2025, 3:02 AM). A "Custom Profile" checkbox is checked.
- Page Layouts:** A grid showing assignments for various page layouts across different object types like Global, Email Application, Home Page Layout, Account, Alternative Payment Method, etc., to specific layouts like "Global Layout" or "Account Layout".

2. (Optional) Setup → Network Access → set allowed IP ranges.

The screenshot shows the Salesforce Setup interface with the following details:

- Header:** Search bar with "Search Setup", a gear icon, and various navigation icons.
- Left Navigation:** "Setup" tab selected, "Network Access" under "Security".
- Page Title:** "SETUP Network Access" with a "Network Access" sub-section.
- Description:** "The list below contains IP address ranges from sources that your organization trusts. Users logging in to salesforce.com with a browser from trusted networks are allowed to access salesforce.com without having to activate their computers."
- Table:** "TRUSTED IP RANGES" showing a single entry:

Action	Start IP Address	End IP Address	Description
Edit Del	192.168.1.1	192.168.1.125	

◆ 7. Dev Org Setup

Thoroughly tested all objects, fields, and relationships. Checked validations, layouts, and data flow to ensure everything works as expected before moving to the next phase.

Use Case:

- Developer Org is used as a **sandbox environment** for experimentation, testing new features, or building custom objects without affecting production.

Steps to Implement:

1. Already created in Step 1 (Developer Edition signup).
2. Use this org to create **Cases**, **Queues**, etc.

Step-1:- Access Setup

1. Click the **Gear Icon** () → **Setup**.
2. You are now in **Setup**, where you can configure:
 - **Objects** (e.g., Cases, Accounts)

Objects

Use Case: Track customers, complaints, and interactions.

Steps:

1. Setup → **Object Manager** → click **Create** → **Custom Object**.
2. Create objects:
 - **Complaint** → to log customer complaints.
 - Fields: Subject, Description, Status, Priority, Assigned To, Customer Name (lookup Account/Contact)

SETUP > OBJECT MANAGER
Complaint

Fields & Relationships
10 Items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Assigned To	Assigned_To_c	Lookup(User)		✓
Complaint Name	Name	Text(80)		✓
Created By	CreatedById	Lookup(User)		
Customer Name	Customer_Name_c	Lookup(Account)		✓
Description	Description__c	Long Text Area(32768)		
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		✓
Priority	Priority__c	Picklist		
Status	Status__c	Picklist		
Subject	Subject__c	Text Area(255)		

- **Customer (Account / Contact) → Salesforce standard objects; use them to store customer info.**

SETUP > OBJECT MANAGER
Customer

Details

Description	Enable Reports
API Name	Track Activities
Customer	Track Field History
Custom	Deployment Status
Singular Label	Help Settings
Customer	Standard salesforce.com Help Window
Plural Label	
Customers	

3. Add relationships:

- **Lookup/ Master-Detail fields to connect Cases with Accounts or Contacts.**

- o **Queues** (Complaints, Escalation)

Queues

Use Case: Route complaints efficiently.

Steps:

1. Setup → **Queues** → New
2. Create:
 - o **Complaints_Queue** → assign **Agents**
 - o **Escalation_Queue** → assign **Managers**
3. Configure **object = Complaint** so new complaints automatically go to the right queue.

The screenshot shows the Salesforce Setup interface with the following details:

- Header:** Includes the Salesforce logo, a search bar labeled "Search Setup", and various navigation icons.
- Left Sidebar:** Shows a tree structure with "Setup" selected at the top level. Under "Queues", there are "Complaints_Queue" and "Escalation_Queue". Other categories like "Users", "Environments", and "Jobs" are also listed.
- Central Content:**
 - Section Header:** "Queues" with a "SETUP" icon.
 - Description:** A brief explanation of what queues are and how they work.
 - View Options:** "All" dropdown, "Edit", "Create New View", and a "Help for this Page" link.
 - Table:** Displays two rows of queue information:

Action	Label	Queue Name	Queue Email	Supported Objects	Modified By	Last Modified Date
Edit Del	Complaints_Queue	Complaints_Queue		Complaint; Case	Jain_Samyak	9/25/2025, 5:10 AM
Edit Del	Escalation_Queue	Escalation_Queue		Complaint; Case	Jain_Samyak	9/25/2025, 5:11 AM

◆ 8. Sandbox Usage

- **Sandbox Usage & Deployment Basics:** While a sandbox wasn't mandatory for this project, Salesforce change sets and VS Code with SFDX CLI were used for deployments.

Phase 3: Data Modeling & Relationships

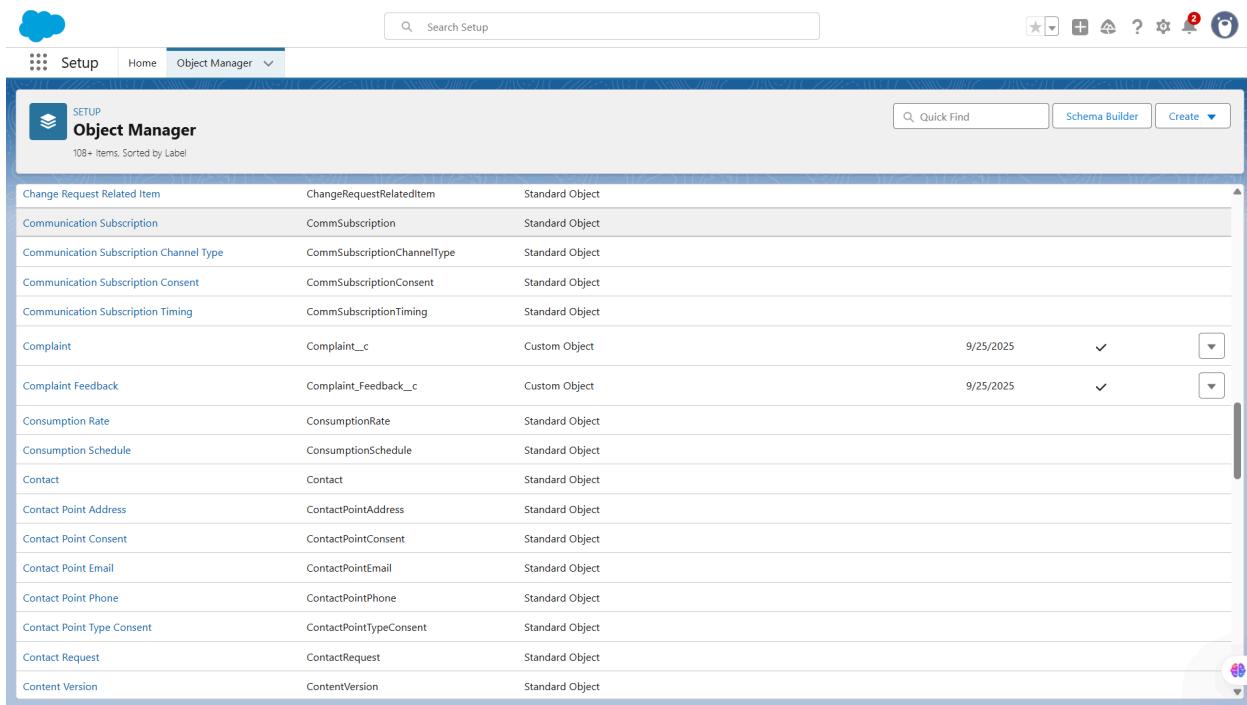
◆ 1. Standard & Custom Objects

We created workflow rules to **automate repetitive tasks**. For example, sending email alerts to agents when a new complaint is logged or updating fields based on specific conditions.

- **Use Case:**

Standard objects like **Cases, Contacts, Accounts** will handle customer and complaint info. But since your CRM is focused on complaints, we'll also create **Custom Objects** like:

1. *Complaint* → to log complaint details.
2. *Complaint Feedback* → to track customer feedback after resolution.



The screenshot shows the Salesforce Object Manager page. At the top, there's a navigation bar with icons for Setup, Home, and Object Manager. A search bar says "Search Setup". On the right of the search bar are icons for Quick Find, Schema Builder, and Create. The main area is titled "Object Manager" and shows a list of 108 items sorted by Label. The list includes various standard and custom objects:

Name	Label	Type
Change Request Related Item	ChangeRequestRelatedItem	Standard Object
Communication Subscription	CommSubscription	Standard Object
Communication Subscription Channel Type	CommSubscriptionChannelType	Standard Object
Communication Subscription Consent	CommSubscriptionConsent	Standard Object
Communication Subscription Timing	CommSubscriptionTiming	Standard Object
Complaint	Complaint_c	Custom Object
Complaint Feedback	Complaint_Feedback_c	Custom Object
Consumption Rate	ConsumptionRate	Standard Object
Consumption Schedule	ConsumptionSchedule	Standard Object
Contact	Contact	Standard Object
Contact Point Address	ContactPointAddress	Standard Object
Contact Point Consent	ContactPointConsent	Standard Object
Contact Point Email	ContactPointEmail	Standard Object
Contact Point Phone	ContactPointPhone	Standard Object
Contact Point Type Consent	ContactPointTypeConsent	Standard Object
Contact Request	ContactRequest	Standard Object
Content Version	ContentVersion	Standard Object

◆ 2. Fields

Built processes to handle **multi-step automation** that workflow rules couldn't manage. For example, automatically creating a follow-up task when a complaint status changes.

- **Use Case:**

Each object needs specific fields. For example:

1. *Complaint Feedback*: Rating (Number), Feedback Comments (Text Area).

The screenshot shows the Salesforce Object Manager interface. The top navigation bar includes a cloud icon, 'Setup', 'Home', and 'Object Manager'. The main title is 'Complaint Feedback'. On the left, a sidebar lists various setup categories like Details, Fields & Relationships, Page Layouts, etc. The 'Fields & Relationships' section is selected and displays a table of fields. The table has columns: FIELD LABEL, FIELD NAME, DATA TYPE, CONTROLLING FIELD, and INDEXED. The data is as follows:

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Complaint	Complaint_c	Lookup(Complaint)		✓
Complaint Feedback Number	Name	Auto Number		✓
Created By	CreatedById	Lookup(User)		
Customer	Customer_c	Lookup(Contact)		✓
Feedback Comments	Feedback_Comments_c	Long Text Area(32768)		
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		✓
Rating	Rating_c	Picklist		
Resolution Time	Resolution_Time_c	Number(18, 0)		

◆ 3. Record Types

Set up approval processes to manage **structured approvals**. For instance, high-priority complaints require manager approval before marking them resolved.

- **Use Case:**

Suppose complaints come from **different channels** (Phone, Email, Walk-in). We may need different page layouts and picklist values per channel. Record Types let us customize.

- **Action Steps:**

1. Go to Object Manager → Complaint → Record Types → New.

2. Create *Phone Complaint*, *Email Complaint*, *Walk-in Complaint*.

The screenshot shows the Salesforce Setup interface for the 'Complaint' object. The left sidebar has a 'Record Types' section selected. The main area displays a table titled 'Record Types' with three items: 'Email Complaint', 'Phone Complaint', and 'Walk-in Complaint'. Each row includes columns for 'Record Type Label', 'Description' (which is collapsed), 'Active' (checked), and 'Modified By' (Samyak Jain, 9/25/2025, 6:50 AM). There are also 'Quick Find', 'New', and 'Page Layout Assignment' buttons at the top right of the list view.

RECORD TYPE LABEL	DESCRIPTION	ACTIVE	MODIFIED BY
Email Complaint	▼	✓	Samyak Jain, 9/25/2025, 6:50 AM
Phone Complaint	▼	✓	Samyak Jain, 9/25/2025, 6:49 AM
Walk-in Complaint	▼	✓	Samyak Jain, 9/25/2025, 6:50 AM

◆ 4. Page Layouts

Defined rules to **auto-assign complaints** to the right agent or queue based on criteria like product type, region, or priority.

- **Use Case:**

- Phone complaints may need *Call Duration* field.
- Email complaints may need *Email Subject* field.
Different layouts per record type keep things clean.

The screenshot shows the Salesforce Setup interface with the following details:

- Header:** Search bar with "Search Setup".
- Top Navigation:** Cloud icon, Setup, Home, Object Manager.
- Breadcrumbs:** SETUP > OBJECT MANAGER Complaint
- Left Sidebar:** Navigation menu with items like Details, Fields & Relationships, Page Layouts (which is selected), Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Restriction Rules, Scoping Rules, Object Access, Triggers, Flow Triggers, Validation Rules, and Conditional Field Formatting.
- Table:** Page Layouts table with 4 items, sorted by Page Layout Name.

PAGE LAYOUT NAME	CREATED BY	MODIFIED BY
Complaint Layout	Samyak Jain, 9/25/2025, 3:51 AM	Samyak Jain, 9/25/2025, 6:41 AM
Email Complaint Layout	Samyak Jain, 9/25/2025, 6:58 AM	Samyak Jain, 9/25/2025, 6:58 AM
Phone Complaint Layout	Samyak Jain, 9/25/2025, 6:55 AM	Samyak Jain, 9/25/2025, 6:57 AM
Walk-in Complaint Layout	Samyak Jain, 9/25/2025, 6:59 AM	Samyak Jain, 9/25/2025, 6:59 AM
- Buttons:** Quick Find, New, Page Layout Assignment.

◆ 5. Compact Layouts

Configured rules to **escalate complaints** if not resolved within a certain time, ensuring timely attention to high-priority cases.

- **Use Case:**

When viewing a complaint on mobile or in highlights panel, you don't want 20 fields showing.

Example: Show only *Complaint ID, Status, Severity, Assigned Agent*.

The screenshot shows the Salesforce Setup interface with the following details:

- Header:** Search bar with "Search Setup", a gear icon, and other navigation icons.
- Breadcrumbs:** SETUP > OBJECT MANAGER > Complaint
- Left Sidebar:** A vertical sidebar with various configuration options like Details, Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, and a selected "Compact Layouts" option.
- Central Content:**
 - Section Header:** Complaint Compact Layout
 - Form Fields:**

Label	Complaint Compact Layout
API Name	Complaint_Compact_Layout
Included Fields	Complaint Name, Record Type, Status, Customer Name, Assigned To, Priority
 - Timestamps:** Created By: Samyak.Jain, 9/25/2025, 7:03 AM | Modified By: Samyak.Jain, 9/25/2025, 7:03 AM
 - Action Buttons:** Edit, Clone, Delete, Compact Layout Assignment
- Help:** Help for this Page

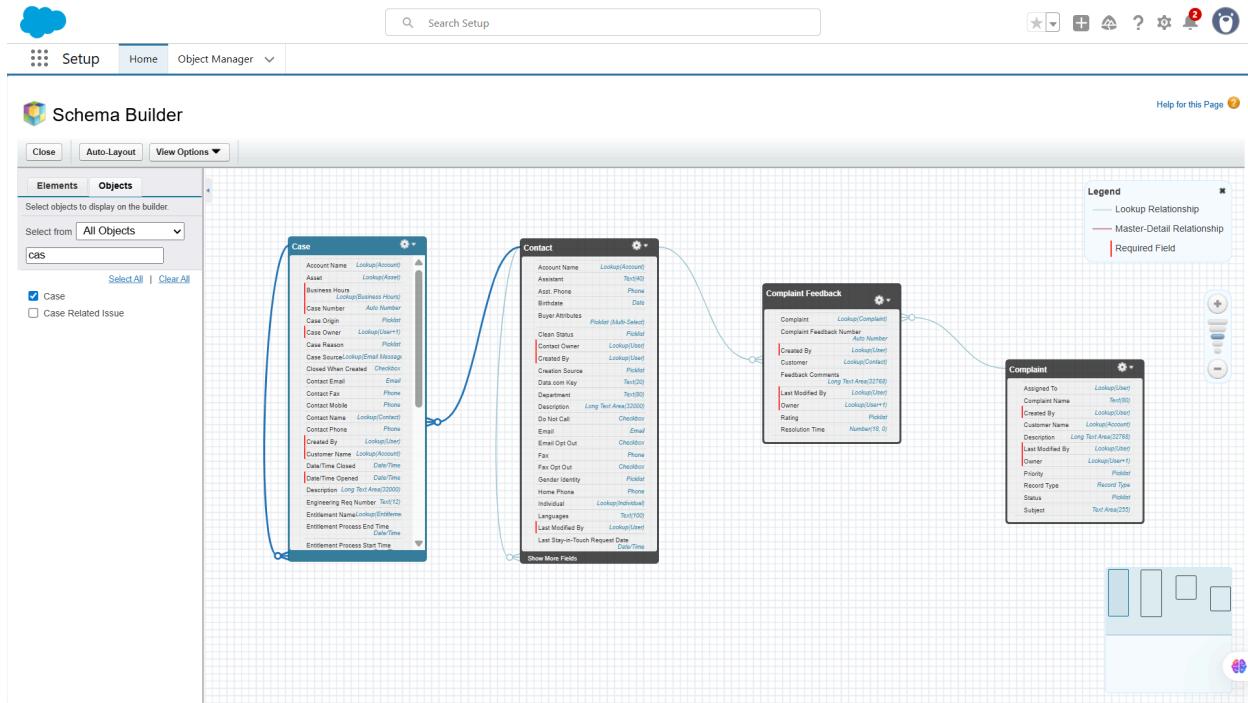
◆ 6. Schema Builder

Added additional validation rules specifically for automation scenarios to **prevent incorrect data updates** by automated processes.

- **Use Case:**

Helps visualize all objects and their relationships in one place.

Example: Complaint → related to Contact (lookup), Complaint Feedback → related to Complaint (lookup).



◆ 7. Lookup vs Master-Detail vs Hierarchical Relationships

Created formula fields for **dynamic calculations** and roll-up summary fields to summarize related records, like total feedback count or average rating per complaint.

- **Use Case:**

- **Lookup:** Complaint → Contact (Customer may exist without complaint).

The screenshot shows the Salesforce Setup interface with the following details:

- Header:** Search bar with "Search Setup".
- Top Navigation:** Cloud icon, Setup, Home, Object Manager.
- Breadcrumbs:** SETUP > OBJECT MANAGER Complaint
- Left Sidebar:** Details, Fields & Relationships (selected), Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Restriction Rules, Scoping Rules, Object Access, Triggers, Flow Triggers, Validation Rules, Conditional Field Formatting.
- Table:** Fields & Relationships (12 items, Sorted by Field Label).

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Assigned To	Assigned_To_c	Lookup(User)		✓
Complaint Name	Name	Text(80)		✓
Contact	Contact_c	Lookup(Contact)		✓
Created By	CreatedById	Lookup(User)		
Customer Name	Customer_Name_c	Look		✓
Description	Description__c	Long Text Area(32768)		
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		✓
Priority	Priority__c	Picklist		
Record Type	RecordTypeId	Record Type		✓
Status	Status__c	Picklist		
Subject	Subject__c	Text Area(255)		

- **Master-Detail:** Complaint Feedback → Complaint (Feedback depends on Complaint).

The screenshot shows the Salesforce Setup interface with the following details:

Setup > OBJECT MANAGER

Complaint Feedback

Fields & Relationships (8 Items, Sorted by Field Label)

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Complaint	Complaint__c	Master-Detail(Complaint)		✓
Complaint Feedback Number	Name	Auto Number		✓
Created By	CreatedById	Lookup(User)		
Customer	Customer__c	Lookup(Contact)		✓
Feedback Comments	Feedback_Comments__c	Long Text Area(32768)		
Last Modified By	LastModifiedById	Lookup(User)		
Rating	Rating__c	Picklist		
Resolution Time	Resolution_Time__c	Number(18, 0)		

- **Hierarchical:** Only for User object, e.g., assigning complaint manager.

◆ 8. Junction Objects

Tested all automation flows to ensure **correct execution order and expected outcomes**, preventing conflicts or duplicate actions.

- **Use Case:**

If one complaint can involve **multiple products** and a product can appear in multiple complaints, create a **Complaint_Product__c** junction object.

The screenshot shows the Salesforce Setup interface with the following details:

- Setup** icon and **Object Manager** button in the top navigation bar.
- Search Setup** search bar.
- Fields & Relationships** tab selected in the left sidebar.
- Complaint Product** object selected in the main pane.
- Fields & Relationships** section header with a sub-label "5 Items, Sorted by Field Label".
- FIELD LABEL**, **FIELD NAME**, **DATA TYPE**, **CONTROLLING FIELD**, and **INDEXED** columns in the table.
- Table data:
 - Complaint → Complaint__c → Master-Detail(Complaint) → ✓
 - Complaint_Product__c Name → Name → Text(80) → ✓
 - Created By → CreatedById → Lookup(User) → ✓
 - Last Modified By → LastModifiedById → Lookup(User) → ✓
 - Product → Product__c → Master-Detail(Product) → ✓
- Buttons at the bottom right of the table: Quick Find, New, Deleted Fields, Field Dependencies, Set History Tracking.

◆ 9. External Objects

External objects allow Salesforce to **access data stored outside of Salesforce**, without actually importing it into your org. They work through **Salesforce Connect**, letting users view and interact with external data as if it were a native object.

- **Use Case:**

Suppose complaints are logged in an external ERP system. You can connect via Salesforce Connect to expose that external data inside Salesforce without duplication.

Phase 4: Process Automation

◆ 1. Validation Rules

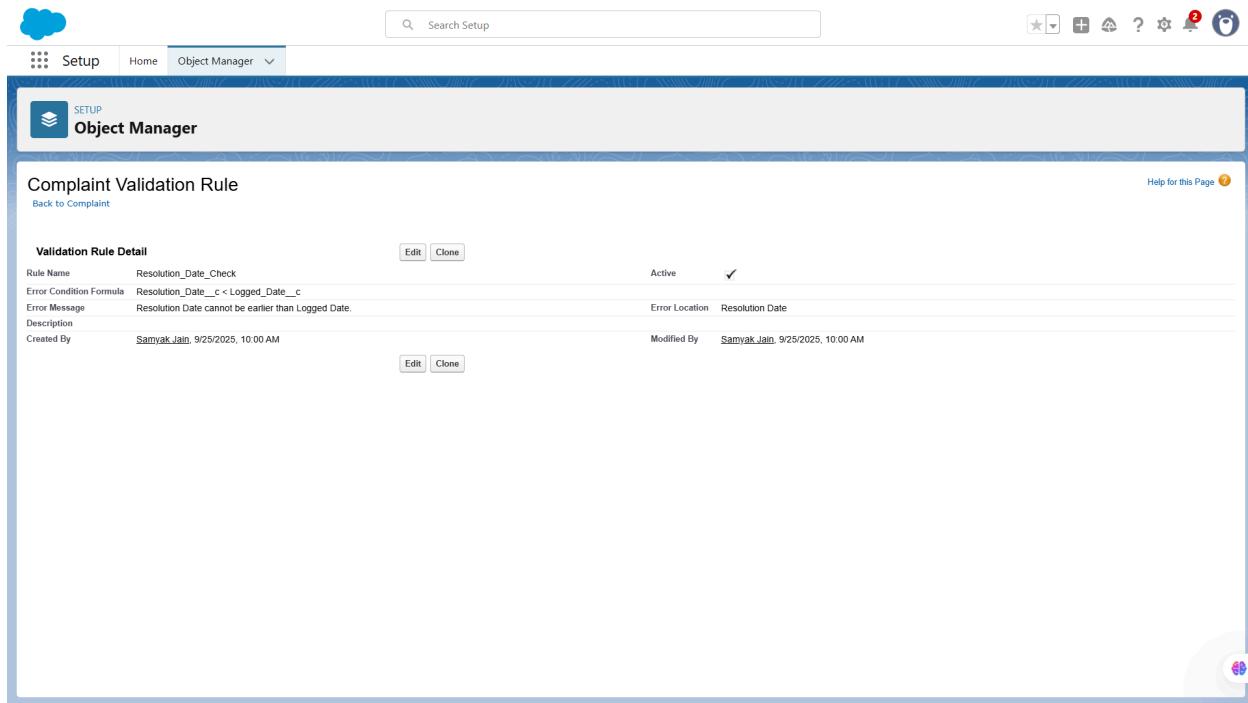
We configured profiles to **define baseline access** for different types of users. This included which objects, fields, and record types each user can view, create, edit, or delete.

Use Case:

Validation Rules ensure data integrity by preventing users from entering incorrect or incomplete information. For example, in the Complaints object, you might want to ensure that the “Complaint Resolution Date” is not earlier than the “Complaint Logged Date.”

Implementation in your project:

- Go to **Setup → Object Manager → Complaints → Validation Rules → New**



◆ 2. Workflow Rules

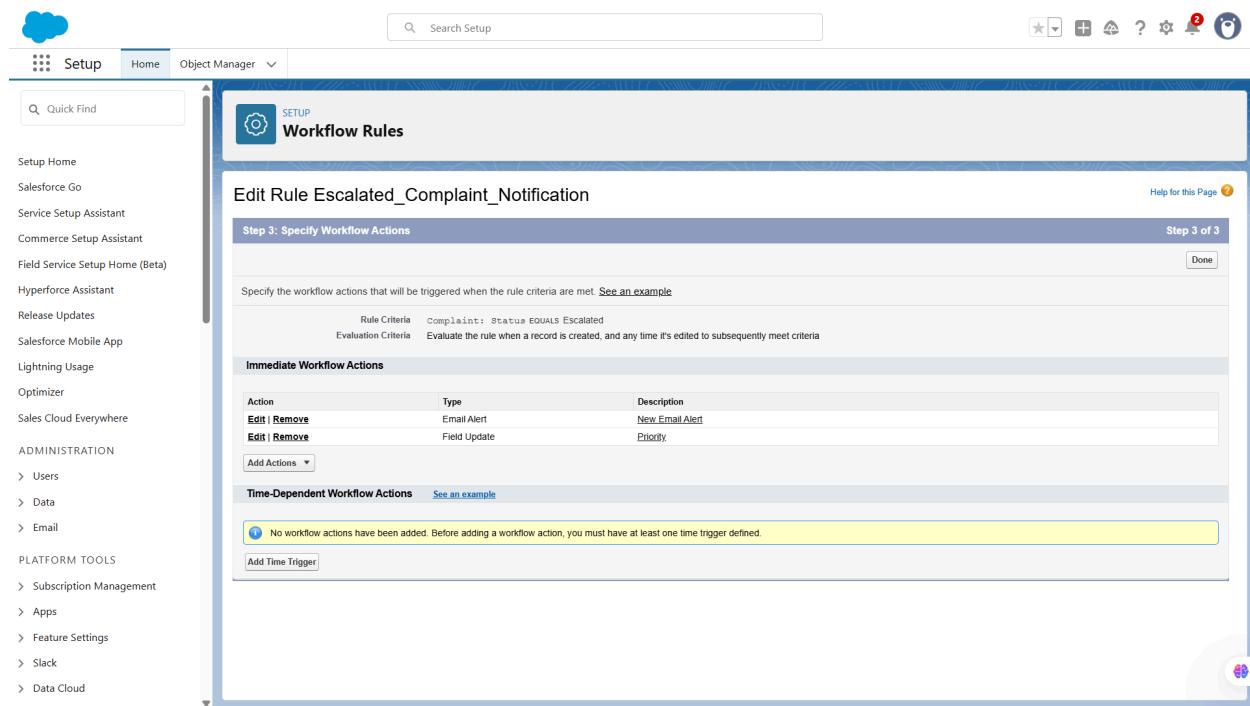
Created permission sets to **grant additional access** without changing profiles. For example, giving certain agents extra permissions for reporting or feedback management.

Use Case:

Workflow Rules automate standard processes like sending emails, updating fields, or creating tasks when certain conditions are met. For example, if a complaint's status changes to "Escalated," a task can be assigned to the manager automatically.

Implementation:

- Go to **Setup** → **Workflow Rules** → **New Rule** → **Select Object (Complaints)**
- Define Criteria: e.g., **Status = Escalated**
- Add Actions:
 - **Field Update:** Change priority to "High"
 - **Email Alert:** Notify manager



◆ 3. Process Builder

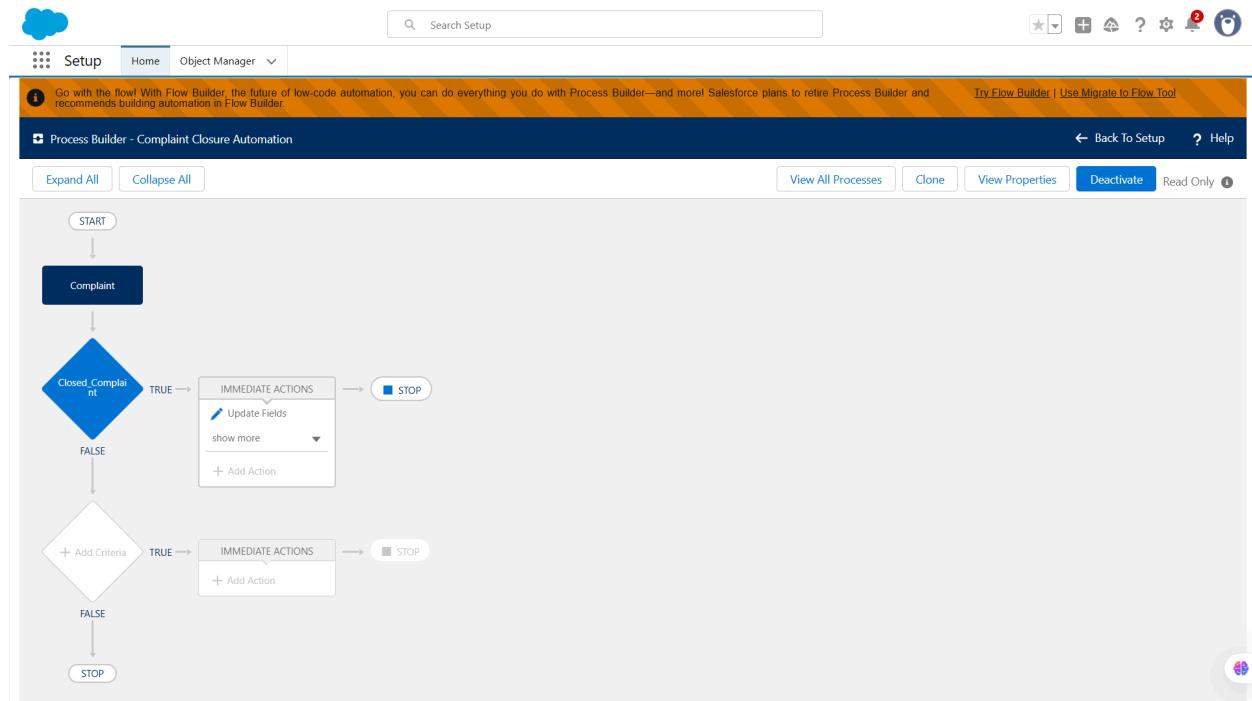
Set up roles to **control record visibility** in the org. Higher roles can view records owned by users in lower roles, supporting managerial oversight.

Use Case:

Process Builder is more advanced than Workflow Rules and allows multi-step automation with complex logic. For example, when a complaint is closed, update related Feedback records and notify the customer.

Implementation:

- **Setup → Process Builder → New → Select Object (Complaints)**
- **Define Start Condition:** e.g., when a record is created or edited
- **Add Immediate Actions:**
 - Update fields
 - Send email
 - Call Flow



◆ 4. Approval Process

Configured rules to **share records automatically** based on criteria like region, product type, or priority, ensuring the right users have access.

Use Case:

Approval Processes are used to get managerial or departmental approval before performing critical actions. Example: High-value complaints (say, complaints with refunds above \$500) need manager approval before closure.

Implementation:

- **Setup → Approval Processes → Complaints → New Approval Process**

- Steps:

1. Define entry criteria
2. Set approvers
3. Define actions for approved/rejected cases

The screenshot shows the Salesforce Setup interface. The left sidebar has sections like Data, Feature Settings, Approval Settings, and Process Automation, with 'Approval Processes' selected. The main area is titled 'SETUP Approval Processes' and shows a process named 'Complaint: Complaint Approval Process'. The 'Process Definition Detail' section includes fields for Process Name (Complaint Approval Process), Unique Name (Complaint_Approval_Process), Description, Entry Criteria (Complaint: Priority EQUALS High), Record Editability (Administrator ONLY), Active status (checked), and a note about Next Automated Approver Determined By. It also includes 'Allow Submitters to Recall Approval Requests' and 'Initial Submission Actions' (Action Type: Record Lock, Description: Lock the record from being edited). Below this are sections for 'Approval Steps', 'Final Approval Actions' (Action Type: Record Lock, Description: Lock the record from being edited), and 'Final Rejection Actions'.

◆ 5. Flow Builder

Defined default access levels for objects, deciding whether records are **public, private, or controlled by parent**. This forms the base layer of data security.

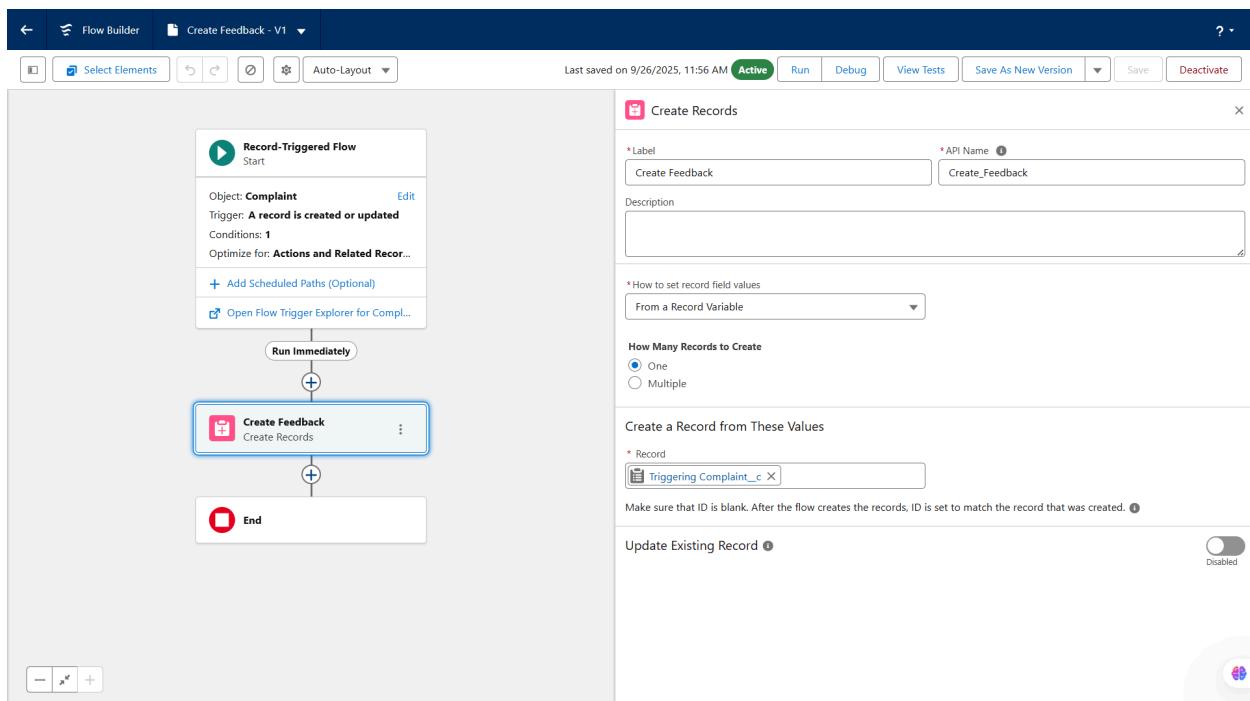
Use Case:

Flows are the most flexible automation tool. You can:

- **Screen Flows:** Guide users through complaint entry or feedback collection.
- **Record-Triggered Flows:** Automate updates when a complaint is created or edited.
- **Scheduled Flows:** Run periodic checks (e.g., send weekly pending complaint report).
- **Auto-launched Flows:** Trigger from Process Builder or button click.

Implementation Example (Record-Triggered Flow):

- **Object:** Complaints
- **Trigger:** When record is created or updated
- **Logic:**
 - If **Status = Resolved** → update Feedback object with “Request Feedback”
 - Send Email Alert to Customer



◆ 6. Email Alerts

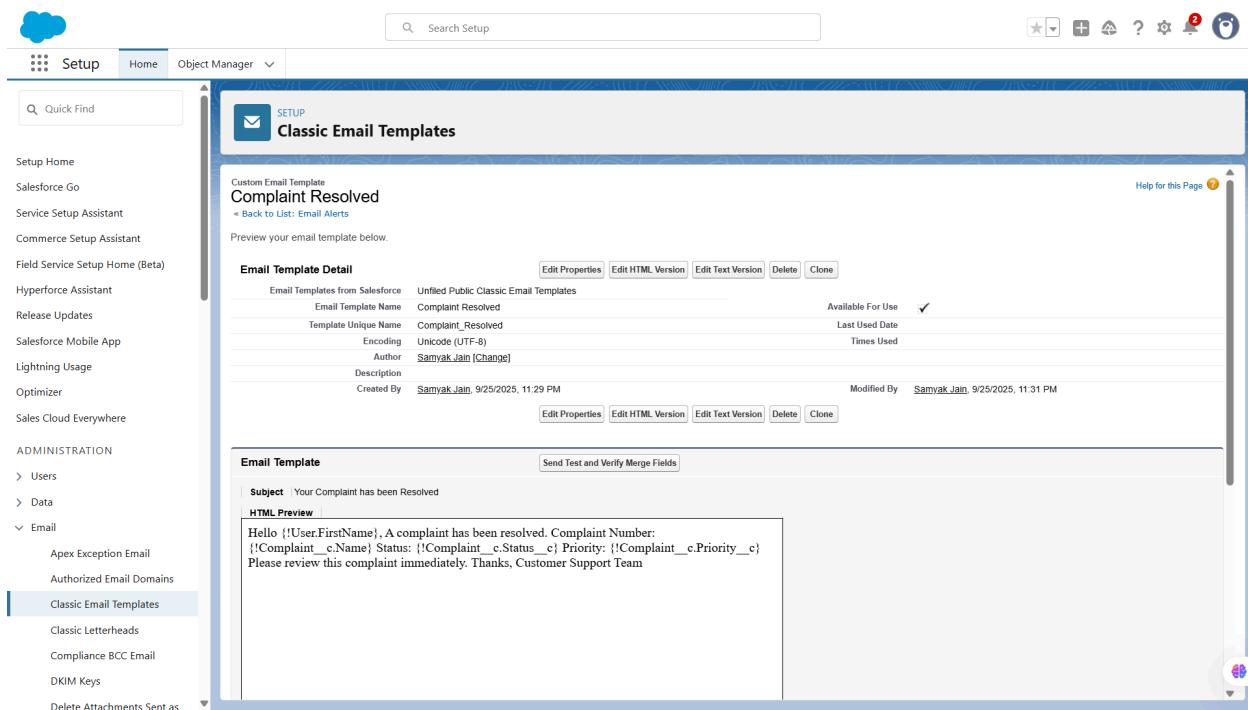
Configured login IP ranges, session timeouts, and security policies to **control user access** and prevent unauthorized login attempts.

Use Case:

Send automated notifications when certain actions occur. For example, when a complaint is escalated, notify the manager or customer.

Implementation:

- **Setup → Email Alerts → New Email Alert**
- Select Object: Complaints
- Select Email Template
- Define Recipients



◆ 7. Field Updates

Enabled audit trail to **track changes** made by users, including object modifications and login history, supporting compliance and security reviews.

Use Case:

Automatically update field values based on criteria. Example: When a complaint is escalated, change its priority to “High.”

Implementation:

- Can be done via **Workflow Rules, Process Builder, or Flows**
- Action Type: **Update Record** → select field and value

The screenshot shows the Salesforce Setup interface with the following details:

- Header:** Search bar with "Search Setup", a gear icon, and various navigation icons.
- Left Sidebar:** "Setup" tab selected. Under "Field Updates", there is a search bar with "field updates" and a note: "Didn't find what you're looking for? Try using Global Search."
- Central Content:** Title "Field Updates" under "SETUP". Subtitle "All Workflow Field Updates". Description: "Field updates allow you to automatically change a field value to one that you specify. Field updates are actions associated with workflow rules and approval processes." View options: "All Workflow Field Updates" (selected), "Edit", and "Create New View". Help link: "Help for this Page".
- Data Table:** A table titled "New Field Update" showing two rows of data.

Action	Name	Field to Update	Operation	Value	Last Modified Date
Edit Del	Changes the case priority to high.	Case: Priority	Value	High	9/20/2025
Edit Del	Priority	Complaint: Priority	Value	High	9/25/2025
- Footer:** Navigation links: A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z, Other, All.

◆ 8. Tasks

We set up automated tasks for complaints and feedback to ensure timely follow-ups. Tasks were assigned to agents with due dates and priorities, helping track responsibilities and avoid missed actions.

Use Case:

Automatically assign tasks to users. Example: Create a task for support staff to follow up a complaint within 24 hours.

Implementation:

- Use Workflow Rules, Process Builder, or Flow
- Define Task properties: subject, due date, assignee

The screenshot shows the Salesforce Setup interface with the 'Workflow Rules' page open. The left sidebar shows navigation options like 'Process Automation', 'Workflow Actions', 'Email Alerts', 'Field Updates', 'Outbound Messages', 'Send Actions', 'Tasks', and 'Workflow Rules'. The main content area shows a workflow rule named 'Escalated_Complaint_Task' with the description 'Complaint: Status EQUALS Escalated'. The rule is active and assigned to the 'Complaint' object. It includes immediate workflow actions for Email Alert, Notify Manager, and Set Priority High.

◆ 9. Custom Notifications

We configured real-time notifications for agents when complaints were assigned, escalated, or feedback required attention. These alerts appeared in Salesforce (desktop and mobile), enabling faster responses and improving overall customer service.

Use Case:

Notify users in Salesforce or mobile app when specific events occur. Example: Notify a support agent when a complaint is reassigned.

Implementation:

- Setup → Custom Notifications → New
- Define Notification Type
- Add to Workflow/Process/Flow

The screenshot shows the Salesforce Setup interface with the following details:

- Header:** Includes the Salesforce logo, a search bar with "Search Setup", and various navigation icons.
- Left Sidebar:** Shows a tree view with "Setup" selected, followed by "Home" and "Object Manager". A search bar at the top of the sidebar contains "Q custom n".
- Current Page:** "Custom Notifications" under the "SETUP" tab.
- Information Bar:** A message states: "When you create and use custom notifications, the title and body of the custom push notification may be saved to and processed by Google, Microsoft and/or Apple. Salesforce is not responsible for the privacy and security practices of third-party systems or applications like Google Cloud Messaging or Apple Push Notification Service."
- Section Header:** "Custom Notification Types"
- Text:** "Send custom notifications using [Flows](#) or [Process Builder](#)".
- Table:** Displays two rows of custom notification types.

NOTIFICATION NAME	API NAME	NAMESPACE	DESKTOP	MOBILE
Complaint Reassignment Alert	Complaint_Reassignment_Alert		✓	✓
enablement_coaching_feedback_ready	enablement_coaching_feedback_ready		✓	▼
- Buttons:** A "New" button is located in the top right corner of the table area.

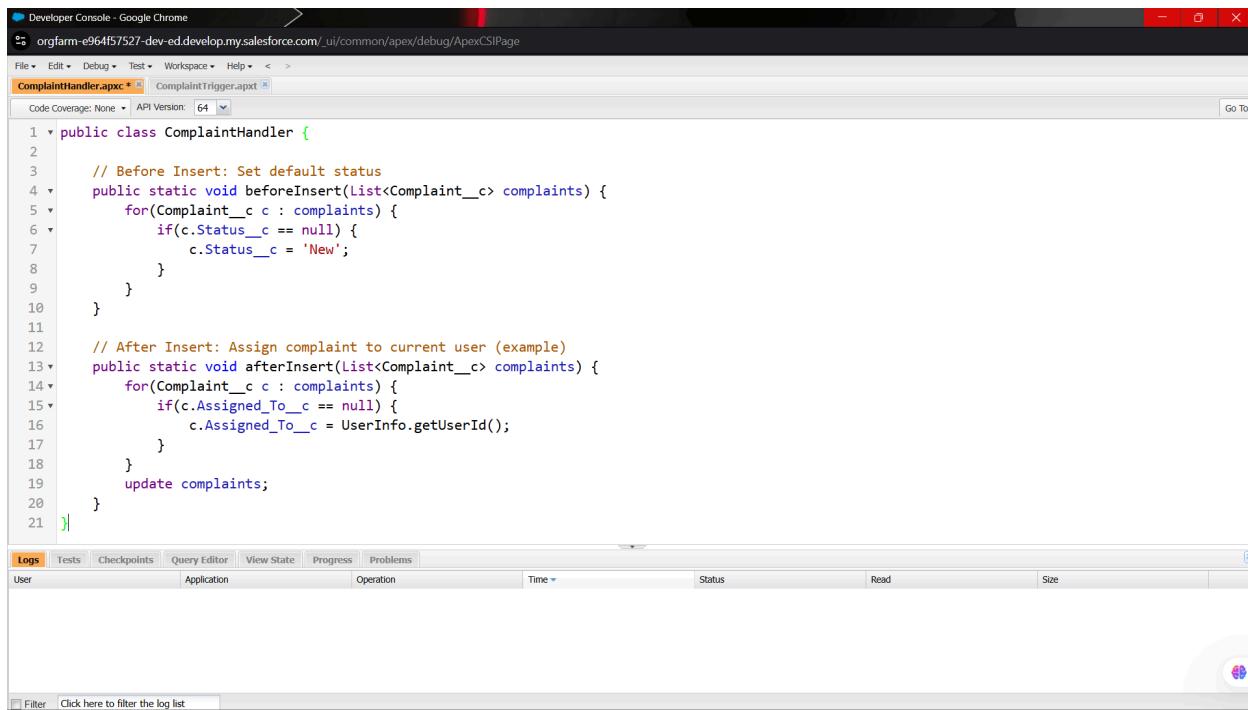
Phase-5: Apex Programming (Developer)

◆ 1. Classes & Objects

Created Apex classes to **encapsulate business logic** for complaints, feedback, and related processes. Objects were used to define instances of these classes and handle data operations efficiently.

Use Case:

Apex **classes** are reusable blocks of code that can contain variables, methods, and logic. They help you organize code efficiently and encapsulate functionality. For example, you might have a **ComplaintHandler** class that processes complaints, validates data, and assigns tasks to support agents.



The screenshot shows the Salesforce Developer Console in Google Chrome. The URL is orgfarm-e964f57527-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCIPage. The tab title is ComplaintHandler.apxc. The code editor contains the following Apex code:

```
1 public class ComplaintHandler {
2
3     // Before Insert: Set default status
4     public static void beforeInsert(List<Complaint__c> complaints) {
5         for(Complaint__c c : complaints) {
6             if(c.Status__c == null) {
7                 c.Status__c = 'New';
8             }
9         }
10    }
11
12    // After Insert: Assign complaint to current user (example)
13    public static void afterInsert(List<Complaint__c> complaints) {
14        for(Complaint__c c : complaints) {
15            if(c.Assigned_To__c == null) {
16                c.Assigned_To__c = UserInfo.getUserId();
17            }
18        }
19        update complaints;
20    }
21 }
```

Below the code editor is a log viewer with tabs for Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The Logs tab is selected. The log table has columns for User, Application, Operation, Time, Status, Read, and Size. There are no logs listed. At the bottom of the log viewer is a filter bar with the placeholder "Click here to filter the log list".

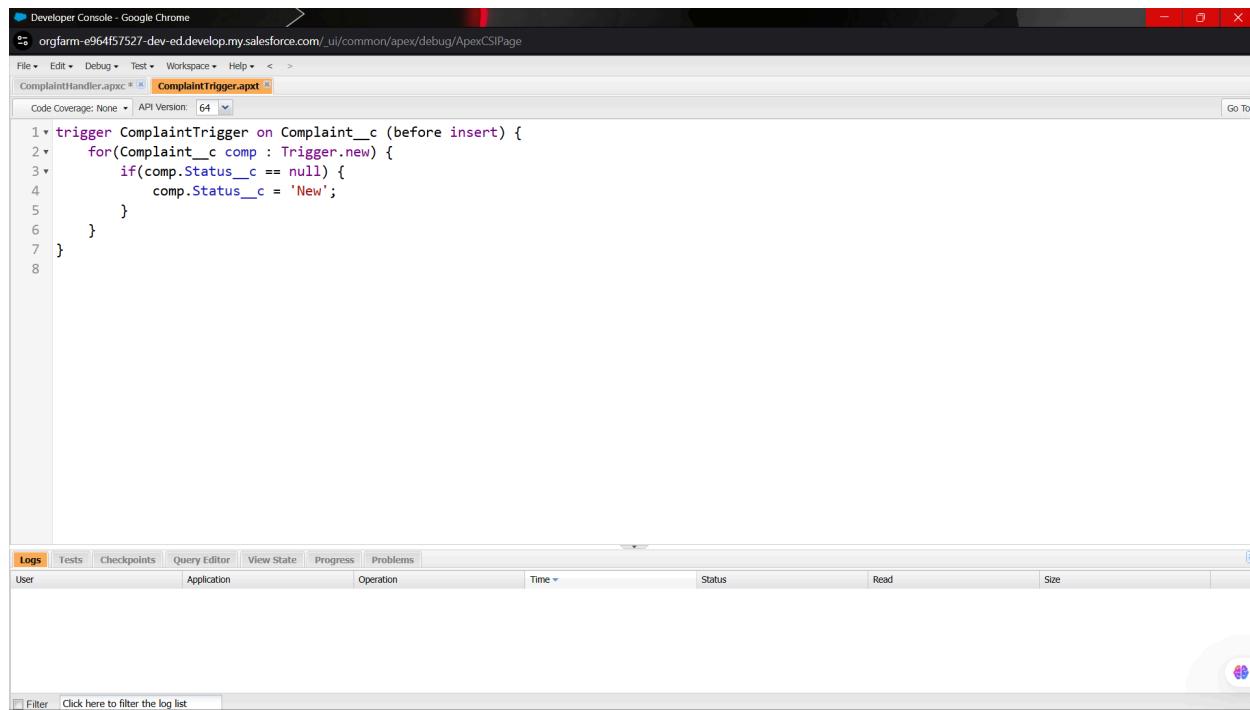
◆ 2. Apex Triggers (before/after insert/update/delete)

Implemented triggers to **automate actions on record changes**. For example, before inserting a complaint, we validated certain fields; after insert, we created related follow-up tasks.

Use Case:

Triggers automatically execute Apex code before or after a record is **inserted, updated, deleted, or undeleted**.

Example: When a complaint is created, you want to **auto-assign it** to a queue or agent and send a notification.



The screenshot shows the Salesforce Developer Console in Google Chrome. The URL is org://e964f57527-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage. The top navigation bar includes File, Edit, Debug, Test, Workspace, Help, and a Go To button. Below the navigation is a tabs section with ComplaintHandler.apcpx and ComplaintTrigger.apcxt selected. A dropdown menu shows Code Coverage: None and API Version: 64. The main area contains the following Apex code:

```
1 trigger ComplaintTrigger on Complaint__c (before insert) {
2     for(Complaint__c comp : Trigger.new) {
3         if(comp.Status__c == null) {
4             comp.Status__c = 'New';
5         }
6     }
7 }
```

Below the code editor is a Log viewer with tabs for Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The Logs tab is selected. It displays a table with columns: User, Application, Operation, Time, Status, Read, and Size. There are no logs listed. At the bottom of the log viewer is a Filter input field and a link to Click here to filter the log list.

◆ 3. Trigger Design Pattern

Used **trigger frameworks** (like one trigger per object and handler classes) to keep triggers modular, reusable, and easy to maintain, preventing code duplication and recursion issues.

Use Case:

Complex triggers can become messy. The **Trigger Handler Pattern** separates logic from the trigger to make it **clean, reusable, and testable**.

```

trigger ComplaintTrigger on Complaint__c (before insert, after insert) {
    if(Trigger.isBefore && Trigger.isInsert) {
        ComplaintHandler.beforeInsert(Trigger.new);
    }
    if(Trigger.isAfter && Trigger.isInsert) {
        ComplaintHandler.afterInsert(Trigger.new);
    }
}

public class ComplaintHandler {
    public static void beforeInsert(List<Complaint__c> complaints) {
        for(Complaint__c c : complaints) {
            c.Status__c = 'New';
        }
    }

    public static void afterInsert(List<Complaint__c> complaints) {
        // Send notifications or assign tasks
    }
}

```

The screenshot shows the Salesforce Developer Console interface. At the top, there's a menu bar with File, Edit, Debug, Test, Workspace, Help, and a Go To button. Below the menu is a toolbar with tabs for Code Coverage, API Version (set to 64), and a dropdown for ComplaintHandler.apxc. The main area contains the Apex code for the ComplaintTrigger and ComplaintHandler classes. At the bottom, there's a log viewer tab labeled 'Logs' with a single entry: 'User Application Operation Status Read Size'. A 'Filter' button and a 'Click here to filter the log list' link are also present.

◆ 4. SOQL & SOSL

Wrote **SOQL queries** to retrieve records efficiently and **SOSL searches** to find records across multiple objects, supporting dynamic reporting and automation logic..

Use Case:

- **SOQL (Salesforce Object Query Language):** Fetch specific records from Salesforce objects.
Example: Find all complaints assigned to a particular agent.
- **SOSL (Salesforce Object Search Language):** Search for text across multiple objects and fields.

The screenshot shows the Salesforce Developer Console interface. At the top, there are tabs for 'File', 'Edit', 'Debug', 'Test', 'Workspace', and 'Help'. Below the tabs, there are several tabs for Apex classes: 'ComplainHandler.apxc', 'ComplainTrigger.apxt', 'ComplainHandlerTest.apxc', 'Log executeAnonymous @26/09/2025, 13:58:23', and 'Log executeAnonymous @26/09/2025, 14:21:49'. The main area is divided into two sections: 'Execution Log' and 'Logs'.

Execution Log:

Timestamp	Event	Details
14:21:49:296	USER_DEBUG	[4] DEBUG [0]

Logs:

User	Application	Operation	Time	Status	Read	Size
Samyak Jain	Unknown	/services/data/v64.0/tooling/executeA...	26/09/2025, 14:21:49	Success		2.98 KB
Samyak Jain	Unknown	/services/data/v64.0/tooling/executeA...	26/09/2025, 13:58:23	Success		3.24 KB
Samyak Jain	Unknown	/services/data/v64.0/tooling/executeA...	26/09/2025, 13:58:04	Success	Unread	3.24 KB
Samyak Jain	Unknown	ApexTestHandler	26/09/2025, 13:52:32	Success	Unread	516 bytes
Samyak Jain	Unknown	ApexTestHandler	26/09/2025, 13:52:21	Success	Unread	2.21 KB

◆ 5. Collections: List, Set, Map

Used **Lists, Sets, and Maps** to store and process multiple records efficiently. For example, Maps were used to link complaint IDs with feedback records for bulk processing.

Use Case:

Collections store multiple values and are heavily used in Apex logic:

- **List:** Ordered collection of records.
- **Set:** Unique values, no duplicates.
- **Map:** Key-value pairs for fast lookups.

The screenshot shows the Salesforce Developer Console in Google Chrome. The URL is orgid=004157527-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCsPage. The top navigation bar includes File, Edit, Debug, Test, Workspace, Help, and a back/forward button. Below the navigation is a search bar with three tabs: Log executeAnonymous @26/09/2025, 14:24:32, Log executeAnonymous @26/09/2025, 14:25:00, and Log executeAnonymous @26/09/2025, 14:26:23. The main area has two tabs: 'Execution Log' and 'Logs'. The 'Execution Log' tab displays a table of log entries with columns: Timestamp, Event, and Details. The 'Logs' tab displays a table of log entries with columns: User, Application, Operation, Time, Status, Read, and Size. Both tables have filters at the bottom.

Timestamp	Event	Details
14:26:23:000	USER_INFO	[EXTERNAL][005g0000008kV1 sjsam92268@agentforce.com (GMT-07:00) Pacific Daylight Time (America/Los_Angeles) GMT-07:00]
14:26:23:000	EXECUTION_STARTED	
14:26:23:000	CODE_UNIT_STARTED	[EXTERNAL]execute_anonymous_apex
14:26:23:001	VARIABLE_SET	[13][complaintMap[Map<Id,Complaint__c>]{true false}
14:26:23:001	VARIABLE_SET	[1][myComplaints[Set<Complaint__c>]{true false}
14:26:23:001	VARIABLE_SET	[7][uniqueSubjects[Set<String>]{true false}
14:26:23:001	HEAP_ALLOCATE	[95]Bytes:3
14:26:23:001	HEAP_ALLOCATE	[100]Bytes:152
14:26:23:001	HEAP_ALLOCATE	[417]Bytes:408
14:26:23:001	HEAP_ALLOCATE	[430]Bytes:408
14:26:23:001	HEAP_ALLOCATE	[317]Bytes:6
14:26:23:001	HEAP_ALLOCATE	[EXTERNAL]Bytes:35
14:26:23:001	STATEMENT_EXECUTE	[1]
14:26:23:001	STATEMENT_EXECUTE	[1]
14:26:23:001	HEAP_ALLOCATE	[1]Bytes:50
14:26:23:001	HEAP_ALLOCATE	[1]Bytes:4
14:26:23:001	HEAP_ALLOCATE	[68]Bytes:5
14:26:23:001	HEAP_ALLOCATE	[74]Bytes:5
14:26:23:001	HEAP_ALLOCATE	[82]Bytes:7
14:26:23:002	SOQL_EXECUTE	[1][Aggregations:0]SELECT Id, Subject__c, Status__c FROM Complaint__c
14:26:23:006	SOQL_EXECUTE	[1]Rows:0
14:26:23:006	HEAP_ALLOCATE	[1]Bytes:4
14:26:23:006	HEAP_ALLOCATE	[1]Bytes:0

User	Application	Operation	Time	Status	Read	Size
Samyak Jain	Unknown	/services/data/v64.0/tooling/executeA...	26/09/2025, 14:26:23	Success		4.51 KB
Samyak Jain	Unknown	/services/data/v64.0/tooling/executeA...	26/09/2025, 14:26:15	List index out of bounds: 0	Unread	4.75 KB
Samyak Jain	Unknown	/services/data/v64.0/tooling/executeA...	26/09/2025, 14:25:35	List index out of bounds: 0	Unread	4.44 KB
Samyak Jain	Unknown	/services/data/v64.0/tooling/executeA...	26/09/2025, 14:25:20	List index out of bounds: 0	Unread	4.44 KB
Samyak Jain	Unknown	/services/data/v64.0/tooling/executeA...	26/09/2025, 14:25:00	Success		3.37 KB
Samyak Jain	Unknown	/services/data/v64.0/tooling/executeA...	26/09/2025, 14:24:32	Success		2.65 KB

◆ 6. Control Statements

Used **Lists, Sets, and Maps** to store and process multiple records efficiently. For example, Maps were used to link complaint IDs with feedback records for bulk processing.

Use Case:

Used to implement decision-making and loops. Common in triggers, classes, batch jobs.
Example: Auto-escalate complaints based on priority.

```
for(Complaint__c c : complaints) {
    if(c.Priority__c == 'High') {
        c.Status__c = 'Escalated';
    } else {
        c.Status__c = 'In Progress';
    }
}
```

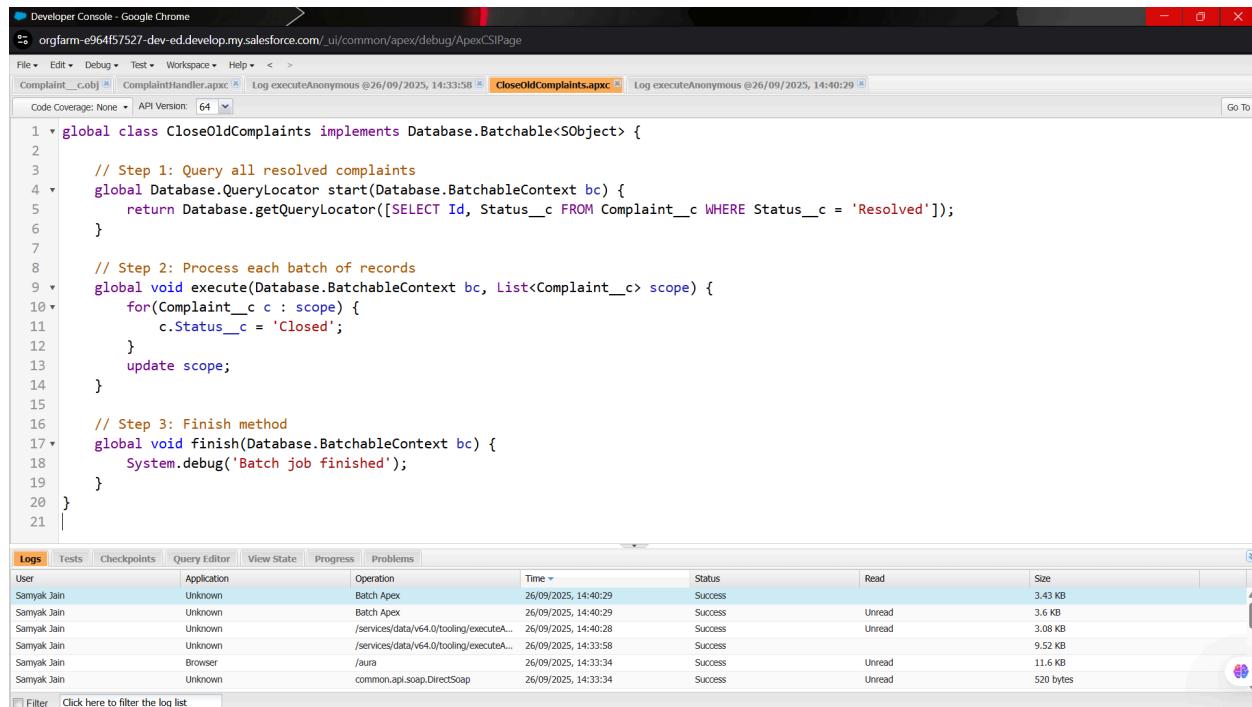
Status
In Progress
Priority
Medium
Assigned To
 Samyak Jain

Status
Escalated
Priority
High
Assigned To
 Samyak Jain

7 Batch Apex

Use Case:

Handles **large data volumes (over 50k records)** asynchronously. Example: Close all resolved complaints older than 90 days every night.



```

Developer Console - Google Chrome
orgfarm-e964f57527-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
File ▾ Edit ▾ Debug ▾ Test ▾ Workspace ▾ Help ▾ < ▾
Complaint__c.apxc Complainthandler.apxc Log executeAnonymous @26/09/2025, 14:33:58 [ ] CloseOldComplaints.apxc Log executeAnonymous @26/09/2025, 14:40:29 [ ]
Code Coverage: None ▾ API Version: 64 Go To
1 * global class CloseOldComplaints implements Database.Batchable<SObject> {
2
3     // Step 1: Query all resolved complaints
4     global Database.QueryLocator start(Database.BatchableContext bc) {
5         return Database.getQueryLocator([SELECT Id, Status__c FROM Complaint__c WHERE Status__c = 'Resolved']);
6     }
7
8     // Step 2: Process each batch of records
9     global void execute(Database.BatchableContext bc, List<Complaint__c> scope) {
10        for(Complaint__c c : scope) {
11            c.Status__c = 'Closed';
12        }
13        update scope;
14    }
15
16     // Step 3: Finish method
17     global void finish(Database.BatchableContext bc) {
18         System.debug('Batch job finished');
19     }
20 }

```

Logs

User	Application	Operation	Time	Status	Read	Size
Samyak Jain	Unknown	Batch Apex	26/09/2025, 14:40:29	Success	Read	3.43 KB
Samyak Jain	Unknown	Batch Apex	26/09/2025, 14:40:29	Success	Unread	3.6 KB
Samyak Jain	Unknown	/services/data/v64.0/tooling/executeA...	26/09/2025, 14:40:28	Success	Unread	3.08 KB
Samyak Jain	Unknown	/services/data/v64.0/tooling/executeA...	26/09/2025, 14:33:58	Success		9.52 KB
Samyak Jain	Browser	/aura	26/09/2025, 14:33:34	Success	Unread	11.6 KB
Samyak Jain	Unknown	common.api.soap.DirectSoap	26/09/2025, 14:33:34	Success	Unread	520 bytes

Enter Apex Code

```

1 CloseOldComplaints batch = new CloseOldComplaints();
2 Database.executeBatch(batch, 200);
3
4
5 global class ScheduleCloseComplaints implements Schedulable {
6     global void execute(SchedulableContext sc) {
7         CloseOldComplaints batch = new CloseOldComplaints();
8         Database.executeBatch(batch, 200);
9     }
10 }
11

```

Open Log

◆ 7. Queueable Apex

Implemented **Queueable Apex** for asynchronous processing of long-running tasks, like updating multiple related records without hitting governor limits.

Use Case:

Similar to Batch Apex but used for **smaller jobs** that need to run asynchronously. Example: Send notifications after complaint processing.

Developer Console - Google Chrome

NotifAgentQueueable.apxc | Log executeAnonymous @26/09/2025, 14:50:52

Execution Log

Timestamp	Event	Details
14:50:52:0000	USER_INFO	[EXTERNAL]005g0000008kqV1 sJeam92268@agentforce.com(GMT-07:00) Pacific Daylight Time (America/Los_Angeles) GMT-07:00
14:50:52:0000	EXECUTION_ST...	
14:50:52:0000	CODE_UNIT_ST...	[EXTERNAL]01pgl000005rjD NotifyAgentQueueable
14:50:52:0001	HEAP_ALLOCATE	[95]bytes:3
14:50:52:0001	HEAP_ALLOCATE	[100]bytes:152
14:50:52:0001	HEAP_ALLOCATE	[417]bytes:408
14:50:52:0001	HEAP_ALLOCATE	[409]bytes:408
14:50:52:0001	HEAP_ALLOCATE	[317]bytes:6
14:50:52:0001	HEAP_ALLOCATE	[EXTERNAL]bytes:5
14:50:52:0001	SYSTEM_METH...	[14]QueueableContextImpl.QUEUE
14:50:52:0001	STATEMENT_EX...	[14]
14:50:52:0001	SYSTEM_METH...	[14]QueueableContextImpl
14:50:52:0001	HEAP_ALLOCATE	[EXTERNAL]bytes:8
14:50:52:0001	HEAP_ALLOCATE	[EXTERNAL]bytes:4
14:50:52:0001	VARIABLE_SCO...	[21]this().QueueableContext
14:50:52:0001	VARIABLE_ASSL...	[21]this().0x5261fe45
14:50:52:0001	VARIABLE_SCO...	[21]jobId()false
14:50:52:0001	VARIABLE_ASSL...	[21]jobId()707g000000ehCjQAJ
14:50:52:0004	HEAP_ALLOCATE	[EXTERNAL]bytes:4
14:50:52:0004	HEAP_ALLOCATE	[EXTERNAL]bytes:2
14:50:52:0004	METHOD_ENTRY	[1]01pgl000005rjD NotifyAgentQ...
14:50:52:0004	STATEMENT_EX...	[1]
14:50:52:0004	STATEMENT_EX...	[1]

This Frame Executable Debug Only Filter Click here t

Enter Apex Code

```

1 // Replace with an existing Complaint Id
2 Id complaintId = '@00gL00000KTGib0AH';
3 System.enqueueJob(new NotifyAgentQueueable(complaintId));
4

```

Open Log

Logs

User	Application	Operation	Time	Status	Read	Size
Samyak Jain	Unknown	/services/data/v64.0/tooling/executeA...	26/09/2025, 14:50:52	Success	Unread	3.48 KB
Samyak Jain	Unknown	QueueableHandler	26/09/2025, 14:50:52	Success		3.45 KB
Samyak Jain	Browser	/aura	26/09/2025, 14:49:08	Success	Unread	11.64 KB
Samyak Jain	Unknown	common.api.soap.DirectSoap	26/09/2025, 14:49:08	Success	Unread	515 bytes
Samyak Jain	Unknown	common.api.soap.DirectSoap	26/09/2025, 14:48:44	Success	Unread	520 bytes
Samyak Jain	Unknown	/services/data/v64.0/tooling/executeA...	26/09/2025, 14:44:49	Invalid id: a083t0000000y123	Unread	2.03 KB

Filter Click here to filter the log list

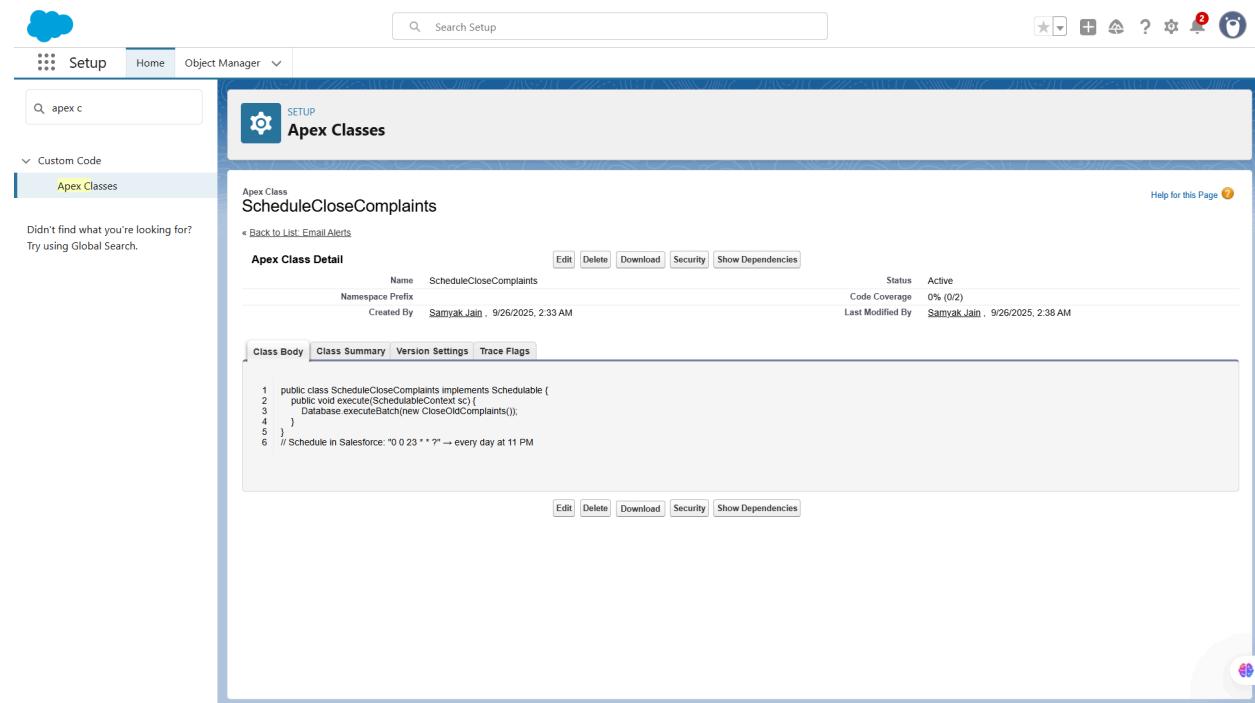
◆ 8. Scheduled Apex

Created **scheduled Apex jobs** to automate periodic processes, such as sending weekly complaint summary emails or cleaning old feedback records.

Use Case:

Run Apex at a scheduled time or interval. Example: Run the batch job to close old complaints every night at 11 PM.

```
public class ScheduleCloseComplaints implements Schedulable {  
    public void execute(SchedulableContext sc) {  
        Database.executeBatch(new CloseOldComplaints());  
    }  
}  
// Schedule in Salesforce: "0 0 23 * * ?" → every day at 11 PM
```



◆ 9. Future Methods

Used **future methods** for asynchronous callouts and operations that didn't require immediate execution, ensuring smooth user experience and system performance.

Use Case:

Run code asynchronously **without blocking the user**. Often used to call external APIs. Example: Call an external system to log complaint updates.

The screenshot shows the Salesforce Developer Console interface. At the top, there's a navigation bar with links like 'File', 'Edit', 'Debug', 'Test', 'Workspace', 'Help'. Below it is a tab bar with several tabs: 'NotifyAgentQueueable.apxc', 'Log executeAnonymous @26/09/2025, 14:50:52', 'ScheduleCloseComplaints.apxc', 'ComplaintNotifier.apxc', and 'Log executeAnonymous @26/09/2025, 15:37:38' (which is highlighted). A 'Logs' tab is also visible.

In the center, there's an 'Execution Log' section with a table:

Timestamp	Event	Details
15:37:38:030	USER_DEBUG	[12] DEBUG Send email to: null

Below the log is an 'Enter Apex Code' window containing the following code:

```
1 ComplaintNotifier.sendEmail('a00gL00000KT6g8QAD');
2
```

At the bottom of the developer console, there are tabs for 'Logs', 'Tests', 'Checkpoints', 'Query Editor', 'View State', 'Progress', and 'Problems' (which is selected). There are also buttons for 'Open Log', 'Execute', and 'Execute Highlighted'.

◆ 10. Exception Handling

Added **try-catch blocks** to gracefully handle errors in Apex code, ensuring that exceptions were logged and users received meaningful error messages instead of system failures.

Use Case:

Prevent runtime errors from crashing code. Example: Catch an error while updating complaints.

The screenshot shows the Salesforce Developer Console with a large code editor on the left and a smaller 'Enter Apex Code' window on the right.

The main code editor contains the following Apex code:

```
5  for(Complaint__c c : complaints) {
6      if(c.Priority__c == null) {
7          throw new CustomException('Priority is missing for Complaint Id: ' + c.Id);
8      }
9
10     if(c.Priority__c == 'High') {
11         c.Status__c = 'Escalated';
12     } else {
13         c.Status__c = 'In Progress';
14     }
15
16     update complaints;
17 }
18
19 } catch(Exception e) {
20     // Log the error
21     System.debug('Error occurred: ' + e.getMessage());
22 }
23
24 // Custom Exception class
25 public class CustomException extends Exception {}
26
```

The 'Enter Apex Code' window contains the following code:

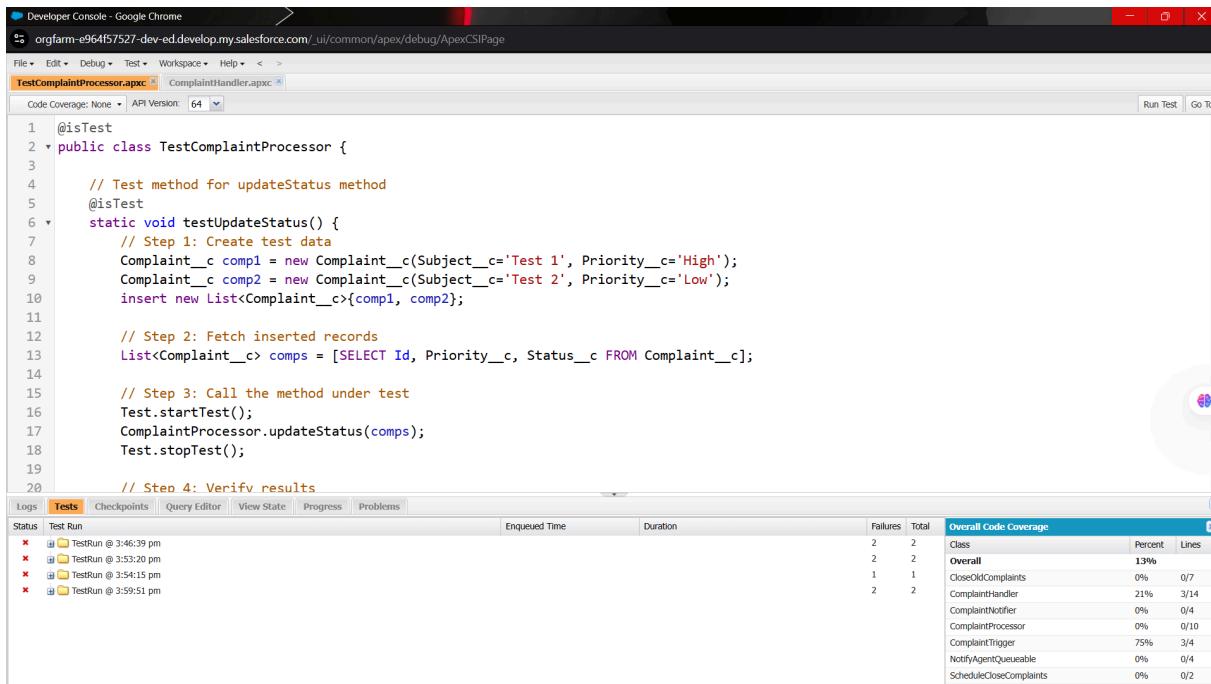
```
1 List<Complaint__c> comps = [SELECT Id, Priority__c, Status__c FROM Comp
2 ComplaintProcessor.updateStatus(comps);
3 |
```

◆ 11. Test Classes

Developed **Apex test classes** to cover triggers and classes with positive and negative test scenarios. This ensured **code coverage compliance** and verified business logic correctness before deployment.

Use Case:

Ensure code coverage and validate functionality. Required for deploying to production.



```
Developer Console - Google Chrome
orgfarm-e964f57527-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
File • Edit • Debug • Test • Workspace • Help • < >
TestComplaintProcessor.apxc [ ComplaintHandler.apxc ]
Code Coverage: None • API Version: 64
1 @isTest
2 public class TestComplaintProcessor {
3
4     // Test method for updateStatus method
5     @isTest
6     static void testUpdateStatus() {
7         // Step 1: Create test data
8         Complaint__c comp1 = new Complaint__c(Subject__c='Test 1', Priority__c='High');
9         Complaint__c comp2 = new Complaint__c(Subject__c='Test 2', Priority__c='Low');
10        insert new List<Complaint__c>{comp1, comp2};
11
12        // Step 2: Fetch inserted records
13        List<Complaint__c> comps = [SELECT Id, Priority__c, Status__c FROM Complaint__c];
14
15        // Step 3: Call the method under test
16        Test.startTest();
17        ComplaintProcessor.updateStatus(comps);
18        Test.stopTest();
19
20        // Step 4: Verify results
}
Logs Tests Checkpoints Query Editor View State Progress Problems
Status Test Run Enqueued Time Duration Failures Total Overall Code Coverage
x TestRun @ 3:46:39 pm 2 2 Class 13%
x TestRun @ 3:53:20 pm 2 2 Overall 13%
x TestRun @ 3:54:15 pm 1 1 CloseOldComplaints 0% 0/7
x TestRun @ 3:59:51 pm 2 2 ComplaintHandler 21% 3/14
ComplaintNotifier 0% 0/4
ComplaintProcessor 0% 0/10
ComplaintTrigger 75% 3/4
NotifyAgentQueueable 0% 0/4
ScheduleCloseComplaints 0% 0/2
```

Phase-6: User Interface Development

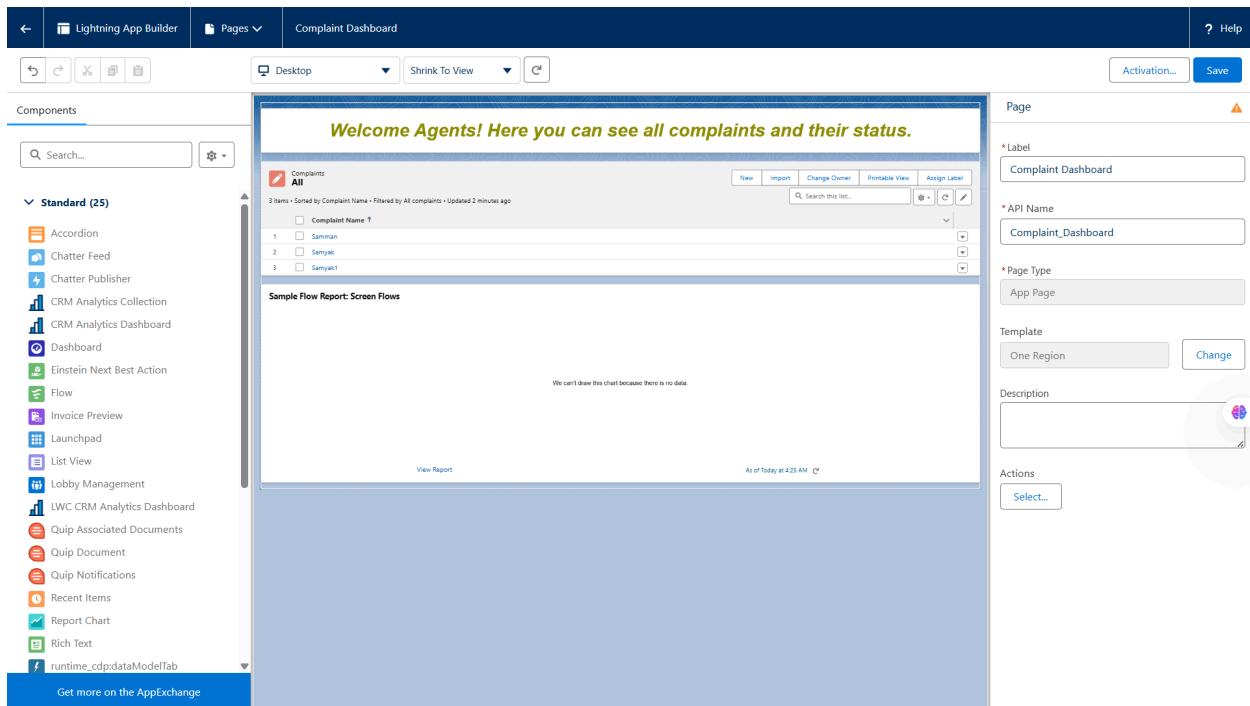
◆ 1. Lightning App Builder

Use Case:

Lightning App Builder allows you to create **custom pages** for Salesforce Lightning Experience using a drag-and-drop interface. You can build **App Pages, Home Pages, and Record Pages** without coding. For example, in your Customer Complaint CRM, you can design a **custom complaint dashboard page** that shows all complaints, their statuses, and assigned agents.

Implementation:

1. Go to **Setup** → **Lightning App Builder** → **New**.
2. Choose **Record Page** or **App Page**.
3. Drag components like **Report Chart**, **Related Lists**, **Rich Text** onto the page.
4. Assign it to the **Complaint__c** object.
5. Activate the page for **Org Default**, **App Default**, or **Specific Profiles**.



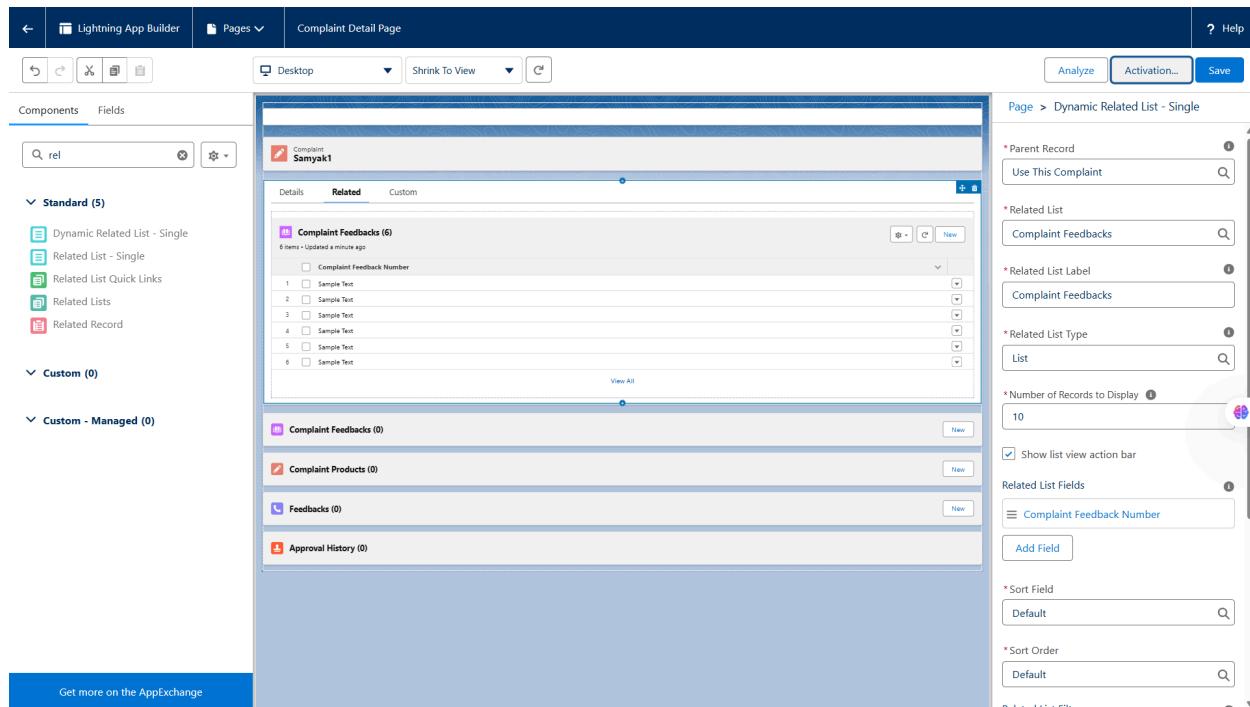
◆ 2. Record Pages

Use Case:

Record Pages control how **individual object records** (e.g., Complaint) are displayed. This allows you to customize the layout depending on **profile or role**. For instance, a **Support Agent Record Page** might show case details and assigned tasks, while a **Manager Record Page** shows reports and analytics.

Implementation:

- Use **Lightning App Builder** to create a **Record Page**.
- Add **Related Lists, Highlights Panel, Tabs, and Custom Components**.
- Assign it to **profiles** so that users see only relevant information.



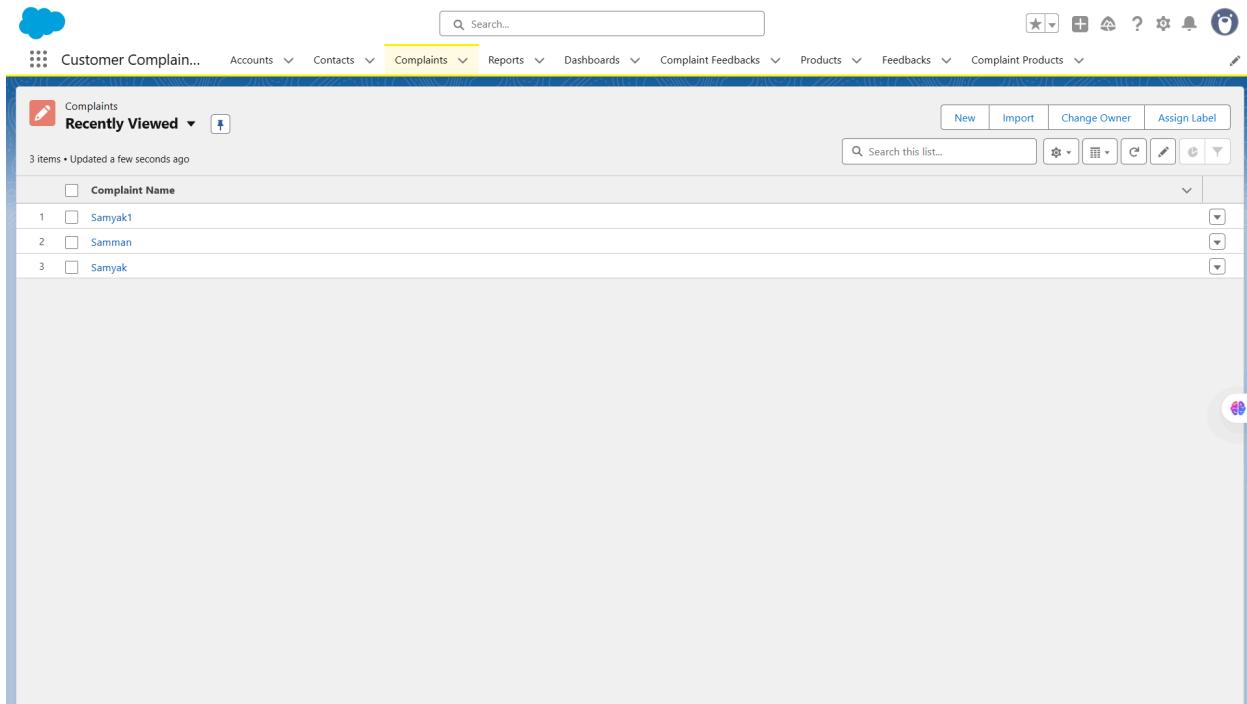
◆ 3. Tabs

Use Case:

Tabs let users **navigate to different objects or components** in Salesforce. In your CRM, you might create tabs for **Complaints, Customers, Products, and Dashboard** so agents can quickly switch between them.

Implementation:

1. Go to **Setup** → **Tabs** → **New**.
2. Select the object or Lightning Component you want to create a tab for.
3. Add it to your **Lightning App**.



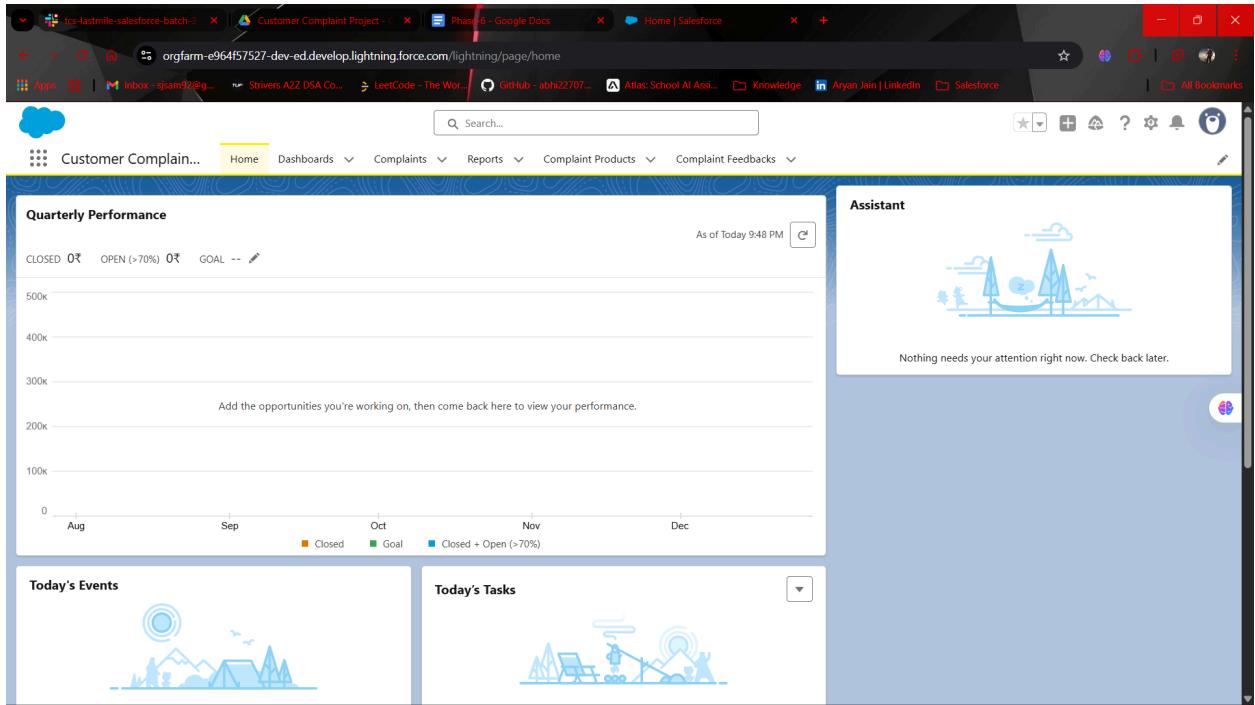
◆ 4. Home Page Layouts

Use Case:

Home Pages display **dashboard-like information** to users when they log in. For a Customer Complaint CRM, you can show **total complaints, pending complaints, tasks, and alerts** on the Home Page.

Implementation:

- Use **Lightning App Builder** → **Home Page**.
- Drag **components** like **List Views, Report Charts, News, and Custom Components**.
- Assign the layout to **profiles** so agents/managers have different dashboards.



◆ 5. Utility Bar

Use Case:

Utility Bar provides **quick-access tools** at the bottom of Salesforce. For example, a Support Agent can have a **Quick Create Complaint, Chat, or Notes component** for faster operations.

Implementation:

1. Go to **Setup** → **App Manager** → **Edit Lightning App** → **Utility Bar**.
2. Add tools like **Quick Actions, Rich Text, or Custom LWC**.
3. Save and test.

The screenshot shows the Salesforce Lightning App Builder interface. The top navigation bar includes 'Lightning App Builder', 'App Settings', 'Pages', and 'Customer Complaint CRM'. The main area is titled 'Utility Items (Desktop Only)' with a sub-instruction: 'Give your users quick access to productivity tools and add background utility items to your app.' A 'Utility Bar Alignment' dropdown is set to 'Mirrored'. On the left, a sidebar lists 'App Settings', 'App Details & Branding', 'App Options', and 'Utility Items (Desktop Only)', which is currently selected. The main panel displays a configuration for a 'Chatter Feed' utility item, including fields for 'Label' (set to 'Chatter Feed'), 'Icon' (set to 'fallback'), 'Panel Width' (set to '340'), and 'Panel Height' (set to '480'). There are also options for 'Start automatically' and 'Component Properties' (set to 'Let users view a Chatter feed'). A 'Feed Type' dropdown is set to 'Bookmarked'.

◆ 6. LWC (Lightning Web Components)

Use Case:

LWCs are **modern, reusable UI components** built with JavaScript, HTML, and CSS. They let you create **custom functionality** beyond standard Salesforce components. For example, a dynamic **complaint priority widget** that updates in real-time.

Implementation:

1. Use **VS Code with Salesforce Extension Pack**.
2. Create LWC: `sfdx force:lightning:component:create --type lwc --componentname ComplaintPriority`.
3. Add HTML, CSS, JS logic.
4. Deploy to org and use it on **Record Pages or App Pages**.

```

1 import { LightningElement, track } from 'lwc';
2
3 export default class ComplaintPriority extends LightningElement {
4     @track priority = 'Normal'; // default
5
6     increasePriority() {
7         if(this.priority === 'Low') this.priority = 'Normal';
8         else if(this.priority === 'Normal') this.priority = 'High';
9     }
10
11    decreasePriority() {
12        if(this.priority === 'High') this.priority = 'Normal';
13        else if(this.priority === 'Normal') this.priority = 'Low';
14    }
15 }
16

```

◆ 7. Apex with LWC

Use Case:

Sometimes, LWC needs **server-side logic** for data that cannot be handled on the client. For example, **fetching complaints with filters and aggregations** via Apex.

Implementation:

1. Create **Apex class** with `@AuraEnabled(cacheable=true)` method.
2. Call this Apex method in your **LWC JS file** using `import methodName from '@salesforce/apex/YourClass.methodName';`
3. Handle the response in LWC and display data.

The screenshot shows a Salesforce Lightning Web Component (LWC) interface. On the left, there's a sidebar with 'Components' and 'Fields' tabs, a search bar with 'pri', and sections for 'Standard (0)', 'Custom (1)' (containing a 'complaintPriority' component), and 'Custom - Managed (0)'. The main area is a 'Details' tab view for a 'Complaint' record. The record has fields like 'Complaint Name' (Samyak1), 'Subject', 'Description', 'Status' (Escalated), 'Priority' (High), 'Assigned To' (Samyak Jain), 'Customer Name', 'Contact', 'Logged Date' (9/26/2025), 'Resolution Date' (9/20/2025), 'Complaint ID' (A-0003), and 'Customer Email'. It was 'Created By' (Samyak Jain) on 9/26/2025 at 2:19 AM and 'Last Modified By' (Samyak Jain) on 9/26/2025 at 3:09 AM. Below the details, there's a section for 'Complaint Priority: Normal' with 'Increase' and 'Decrease' buttons. At the bottom, there are four related lists: 'Complaint Feedbacks (0)', 'Complaint Products (0)', 'Feedbacks (0)', and 'Approval History (0)', each with a 'New' button.

◆ 8. Events in LWC

Use Case:

Events allow **communication between components**. For instance, when a complaint is closed in one component, another component (like a dashboard) should update automatically.

Implementation:

- Use **Custom Events**: `this.dispatchEvent(new CustomEvent('complaintclosed', { detail: { id: recordId } }));`
- The parent component listens: `<c-child-component oncomplaintclosed={handleEvent}></c-child-component>`.

The screenshot shows the Salesforce Setup interface with the 'Lightning Components' tab selected. The main content area displays the 'Lightning Component Detail' for a component named 'complaintDashboard'. The 'Lightning Web Component Bundle Detail' section shows the component was created by Samyak Jain on 10/7/2025 at 12:07 PM. The 'LWC Dependencies' section includes a search bar and a table listing the dependency 'complaintDashboard' with a label 'complaintDashboard', type 'LWC', and API version 64.

◆ 9. Wire Adapters

Use Case:

Wire Adapters let you **reactively get Salesforce data** in LWC. For example, automatically displaying **all open complaints** in a list component that updates whenever records change.

The screenshot shows the Lightning App Builder interface with the following details:

- Header:** Includes tabs for Lightning App Builder, Pages, Complaint Dashboard, and Help.
- Toolbar:** Contains icons for back, forward, search, and save.
- Components Sidebar:** Shows a search bar for "compla" and categories: Standard (0), Custom (2) (with "complaintList" and "complaintPriority" selected), and Custom - Managed (0).
- Main Content Area:**
 - Welcome Agents! Here you can see all complaints and their status.**
 - Complaints Section:** A list of 3 items: Samman, Samyak, and Samyak1. It includes a search bar and filter options.
 - Open Complaints Section:** A table showing columns for Complaint Name, Subject, Status, and Priority. Data rows include Samyak (Escalated, High), Samyak1 (Escalated, High), and Samman (In Progress, Medium).
 - Sample Flow Report: Screen Flows:** A section stating "We can't draw this chart because there is no data." with a "View Report" button and a timestamp "As of Today at 11:53 AM".
- Bottom Bar:** A blue bar with the text "Get more on the AppExchange".

◆ 10. Imperative Apex Calls

Use Case:

Imperative calls allow **manual invocation of Apex** instead of reactive wiring. Useful for actions triggered by buttons or form submissions.

Lightning App Builder | Complaint Dashboard | Help

Components

Q compl

Standard (0)

Custom (2)

complaintDetails
complaintPriority

Custom - Managed (0)

Welcome Agents! Here you can see all complaints and their status.

Complaints All

3 items • Sorted by Complaint Name • Filtered by All complaints • Updated a few seconds ago

Complaint Name ↑

- 1 Samman
- 2 Samyak
- 3 Samyak1

Error fetching complaints: [object Object]

Complaint Details (Imperative Apex)

Enter Complaint ID

Fetch Complaint

Sample Flow Report: Screen Flows

We can't draw this chart because there is no data

View Report As of Today at 12:43 AM CDT

Get more on the AppExchange

◆ 11. Navigation Service

Use Case:

Navigation Service lets you **programmatically navigate** users to **records, lists, or external URLs**. For example, after creating a complaint, the agent is automatically redirected to its detail page.

Lightning App Builder | Complaint Dashboard | Help

Changes saved. Activation... Save

Components

Q co

Standard (2)

Accordion
CRM Analytics Collection

Custom (3)

complaintDetails
complaintPriority
navigateToRecord

Custom - Managed (0)

Welcome Agents! Here you can see all complaints and their status.

Complaints All

3 items • Sorted by Complaint Name • Filtered by All complaints • Updated a few seconds ago

Complaint Name ↑

- 1 Samman
- 2 Samyak
- 3 Samyak1

Error fetching complaints: [object Object]

Complaint Details (Imperative Apex)

Enter Complaint ID

Fetch Complaint

Navigation Service Example

Enter Complaint ID

Go to Complaint Record
Go to Complaint List View
Go to External Website

Sample Flow Report: Screen Flows

We can't draw this chart because there is no data.

Get more on the AppExchange

📌 Phase 7: Integration & External Access

◆ 1. Named Credentials

Use Case:

Simplify and secure authentication for external callouts (e.g., sending complaint details to an external service).

Implementation Steps:

1. Setup → *Named Credentials* → New.
2. Provide external endpoint URL and authentication method.
3. Reference it in Apex



The screenshot shows the Salesforce IDE interface with the file 'ComplaintCallout.apxc' open. The code implements a static method 'sendComplaint' that sends a POST request to a named credential endpoint. The code includes try-catch logic to handle exceptions and debug statements to log response status and body.

```
File ▾ Edit ▾ Debug ▾ Test ▾ Workspace ▾ Help ▾ < >
ComplaintCallout.apxc Log executeAnonymous @08/10/2025, 14:11:41
Code Coverage: None API Version: 64

1 public class ComplaintCallout {
2     public static void sendComplaint(String subject, String status) {
3         try {
4             HttpRequest req = new HttpRequest();
5             req.setEndpoint('callout:Complaint_Integration_Service/complaints'); // Named Credential
6             req.setMethod('POST');
7             req.setHeader('Content-Type', 'application/json');
8             req.setBody('{"subject":"' + subject + '", "status":"' + status + '"}');
9
10            Http http = new Http();
11            HttpResponse res = http.send(req);
12
13            System.debug('Response Status: ' + res.getStatus());
14            System.debug('Response Body: ' + res.getBody());
15
16        } catch (Exception e) {
17            System.debug('Callout failed: ' + e.getMessage());
18        }
19    }
20 }
21 |
```

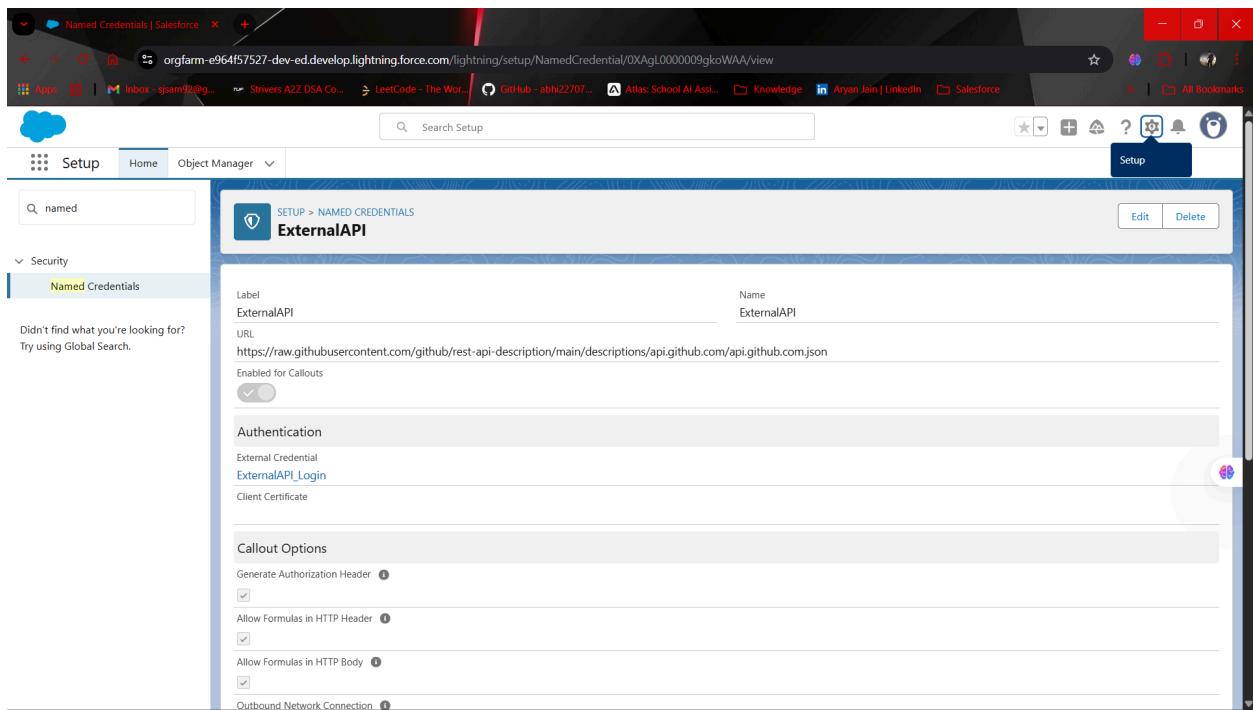
◆ 2. External Services

Use Case:

Integrate Salesforce with REST APIs without writing Apex — automatically create invocable actions.

Implementation Steps:

1. Setup → *External Services* → *New*.
2. Register schema via Swagger/OpenAPI definition.
3. Use the generated actions directly in Flow or Process Builder.



The screenshot shows the Salesforce Developer Console interface. The top navigation bar includes File, Edit, Debug, Test, Workspace, Help, and a Log executeAnonymous entry from 10/10/2025, 14:11:41. The API Version is set to 64. The main area displays the Apex code for the ExternalComplaintService class:

```

1 public with sharing class ExternalComplaintService {
2
3     // Method to create complaint in external system
4     @AuraEnabled
5     public static String sendComplaint(String subject, String description) {
6         Http http = new Http();
7         HttpRequest request = new HttpRequest();
8
9         // Named Credential endpoint
10        request.setEndpoint('callout:ExternalAPI/complaints');
11        request.setMethod('POST');
12        request.setHeader('Content-Type', 'application/json');
13
14        // JSON body
15        Map<String, String> payload = new Map<String, String>{
16            'subject' => subject,
17            'description' => description
18        };
19        request.setBody(JSON.serialize(payload));
20
21        HttpResponse response;

```

Below the code editor, the tabs Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems are visible. The Tests tab is selected. The status bar shows Test Run, Enqueued Time, Duration, Failures, and Total. To the right, a table titled "Overall Code Coverage" provides detailed coverage statistics:

Class	Percent	Lines
Overall	8%	
CloseOldComplaints	0%	0/7
ComplaintController	0%	0/4
ComplaintHandler	21%	3/14
ComplaintNotifier	0%	0/4
ComplaintProcessor	0%	0/10

◆ 3. Web Services (REST/SOAP)

Use Case:

Expose your Complaint object data to external systems.

Implementation Steps:

- **REST:** Create an `@RestResource` Apex class.

The screenshot shows the Salesforce Developer Console in Google Chrome. The URL is orgfarm-e964f57527-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage. The tab title is 'Developer Console - Google Chrome'. The main content area displays the Apex code for 'ComplaintService.apxc'.

```

1  @RestResource(urlMapping='/Complaints/*')
2  global with sharing class ComplaintService {
3
4      // GET request: fetch a complaint by Id
5      @HttpGet
6      global static Complaint__c getComplaint() {
7          RestRequest req = RestContext.request;
8
9          // Extract Id from the URL
10         String id = req.requestURI.substring(req.requestURI.lastIndexOf('/')+1);
11
12         // Query the complaint
13         Complaint__c complaint = [SELECT Id, Subject__c, Status__c FROM Complaint__c WHERE Id = :id LIMIT 1];
14
15         return complaint;
16     }
17
18     // POST request: create a new complaint
19     @HttpPost
20     global static String createComplaint(String subject, String description) {
21         Complaint__c c = new Complaint__c(Subject__c = subject, Status__c = 'New');

```

Below the code editor, there is a navigation bar with tabs: Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The 'Tests' tab is selected. A table titled 'Overall Code Coverage' provides the following data:

Class	Percent	Lines
Overall	7%	
CloseOldComplaints	0%	0/7
ComplaintController	0%	0/4
ComplaintHandler	21%	3/14
ComplaintNotifier	0%	0/4
ComplaintProcessor	0%	0/10

◆ 4. Platform Events

Use Case:

Notify external or internal systems in real time when complaints change (publish–subscribe model).

Implementation Steps:

- Create a *Platform Event* like `Complaint_Updated__e`.
- Publish it via Apex or Flow.
- Subscribe using external systems or internal triggers.

The screenshot shows the Salesforce Developer Console in Google Chrome. The URL is orgfarm-e964f57527-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSPage. The tab title is ComplaintCalloutService.apxc. The code editor displays the following Apex class:

```
1 public with sharing class ComplaintCalloutService {
2
3     public static String sendComplaint(String subject, String description) {
4         try {
5             Http http = new Http();
6             HttpRequest request = new HttpRequest();
7
8             // Endpoint (use Remote Site Setting)
9             request.setEndpoint('https://api.example.com/complaints');
10            request.setMethod('POST');
11            request.setHeader('Content-Type', 'application/json');
12
13            // JSON payload
14            Map<String, String> payload = new Map<String, String>{
15                'subject' => subject,
16                'description' => description
17            };
18            request.setBody(JSON.serialize(payload));
19
20            // Send callout
21            HttpResponse response = http.send(request);
22
23            if(response.getStatusCode() == 200 || response.getStatusCode() == 201) {
24                return 'Complaint sent successfully!';
25            } else {
26                return 'Error: ' + response.getStatusCode() + ' - ' + response.getBody();
27            }
28        } catch(Exception e) {
29            return e.getMessage();
30        }
31    }
32}
```

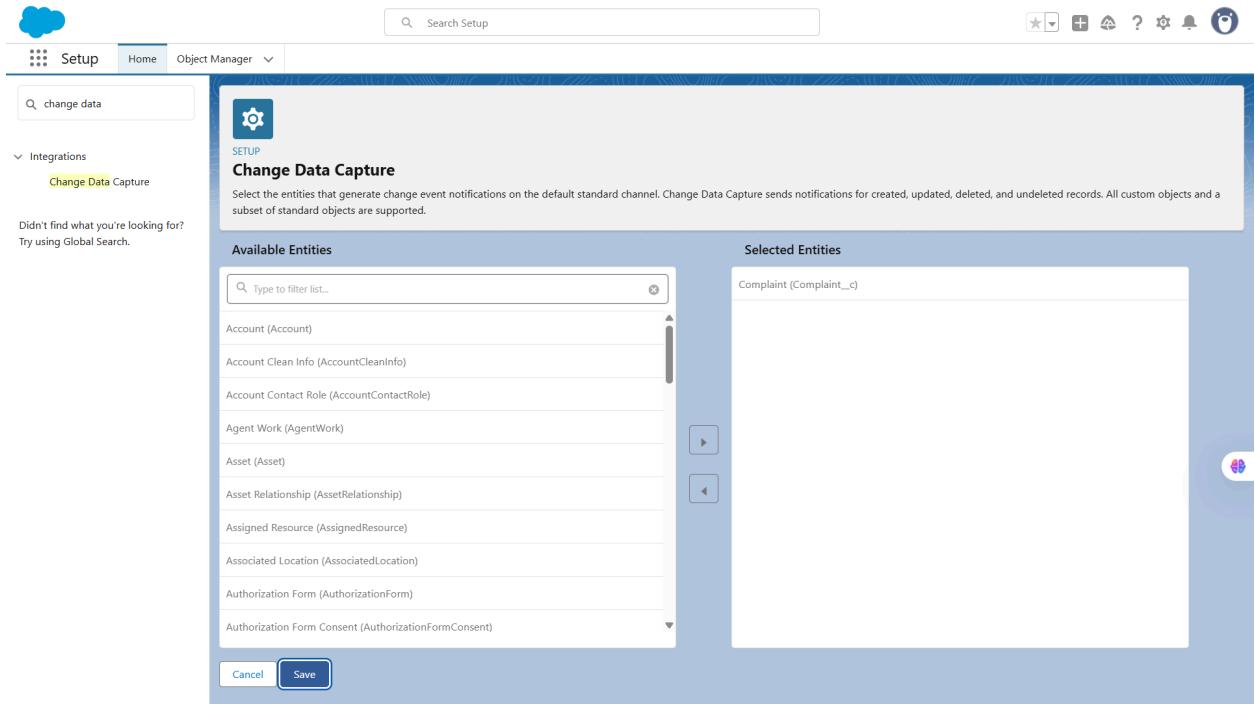
◆ 5. Change Data Capture (CDC)

Use Case:

Stream changes of Complaint records to external systems automatically.

Implementation Steps:

- Enable CDC for `Complaint__c` object in Setup.
- External system subscribes to CDC events via CometD API.



◆ 6. Salesforce Connect

Use Case:

Access external data (e.g., product or customer info from ERP) in real-time *without storing it in Salesforce.*

Implementation Steps:

- Setup → *Salesforce Connect* → *New External Data Source*.
- Choose adapter (OData 2.0 / 4.0).
- Validate and sync external objects.

The screenshot shows the Salesforce Setup interface with the 'External Data Sources' page open. The page title is 'External Data Source: ERP_Data'. It includes a 'Parameters' section with various configuration options and an 'Authentication' section which is currently collapsed.

◆ 7. OAuth & Authentication

Use Case:

Secure API access between Salesforce and external apps.

Implementation Steps:

- Setup → *App Manager* → *New Connected App*.
- Configure OAuth scopes (e.g., `api`, `refresh_token`).
- Use consumer key & secret in the external system for authentication.

Manage External Client Apps
ComplaintCRMIntegration

Contact Email: sjjsam92@gmail.com App Authorization: All users can self-authorize

Type: Local App Status: Enabled

Policies Settings Package Defaults

Configure policies to customize the external client app and plugins for this Salesforce organization.

App Policies

- Start Page: None

OAuth Policies

8. API Limits

Use Case:

Ensure integrations stay within Salesforce governor and daily API call limits.

Implementation Steps:

- Monitor usage in **Setup → System Overview → API Requests**.

System Overview

View key usage data for your org.

Schema

YOUR CUSTOM OBJECTS + YOUR CUSTOM SETTINGS	5	1% (maximum 400)
TOTAL CUSTOM OBJECTS + TOTAL CUSTOM SETTINGS	5	0% (maximum 3,000)
YOUR CUSTOM METADATA TYPES	0	0% (maximum 200)
TOTAL CUSTOM METADATA TYPES	0	0% (maximum 350)
CUSTOM METADATA TYPE USAGE	0%	0% (0 of 10,000,000 characters)
DATA STORAGE	360 KB (Approx.)	7% (maximum 5.0 MB)

API Usage

API REQUESTS, LAST 24 HOURS	348	2% (maximum 15,000)
-----------------------------	-----	---------------------

Business Logic

RULES	0
-------	---

User Interface

CUSTOM APPS	0
-------------	---

◆ 9. Remote Site Settings

Use Case:

Authorize Salesforce to make callouts to external URLs.

Implementation Steps:

- Setup → Security → *Remote Site Settings* → *New*.
- Add domain URL of the external API (e.g., <https://api.myservice.com>).

The screenshot shows the Salesforce Setup interface. The left sidebar is collapsed, and the main area displays the 'Remote Site Settings' page under the 'Security' section. The page title is 'Remote Site Details'. A table lists the configuration for a single remote site:

Remote Site Detail	
Remote Site Name	ExternalFeedbackAPI
Remote Site URL	https://78443748-a74a-41bc-811d-a61d2e26653d.mock.pstmn.io
Disable Protocol Security	<input type="checkbox"/>
Description	External feedback service for complaint updates
Active	<input checked="" type="checkbox"/>
Created By	Samyak.Jain, 10/8/2025, 12:42 PM

Below the table are three buttons: 'Edit', 'Delete', and 'Clone'. The top right corner of the page has a 'Help for this Page' link. The top navigation bar includes a search bar, a user icon, and various setup icons.

Phase-8: Data Management & Deployment

◆ 1. Data Import Wizard

Use Case:

The Data Import Wizard allows admins to **easily import small to medium-sized datasets** (up to 50,000 records) into Salesforce without using any external tools. It's ideal for uploading complaints, customers, or feedback records from spreadsheets into custom objects.

Implementation Steps:

1. Go to **Setup → Data Import Wizard**.
2. Click **Launch Wizard**.
3. Choose the object (e.g., `Complaint__c` or `Customer__c`).
4. Upload a `.CSV` file containing your data.
5. Map CSV columns to Salesforce fields.
6. Click **Start Import** and monitor progress under **Bulk Data Load Jobs**.

The screenshot shows the Bulk Data Load Jobs page in the Salesforce Setup interface. The page title is "Bulk Data Load Jobs" with a job ID of "750gL00000FLqe9". The main section displays "Job Detail" for the job, showing the following data:

Field	Value
Job ID	750gL00000FLqe9
Submitted By	Samyak Jain
Start Time	10/11/2025, 10:02 AM PST
End Time	(not explicitly shown)
Object	Complaint
External ID Field	(not explicitly shown)
Content Type	CSV
Concurrency Mode	Parallel
API Version	65.0

Below the Job Detail, there are sections for "Status" (Closed), "Total Processing Time (ms)" (5134), "API Active Processing Time (ms)" (4374), and "Apex Processing Time (ms)" (3298). The status table also includes rows for "Completed Batches" (0), "Failed Batches" (0), "Progress" (0%), and "Records Processed" (1000). The "Batches" table at the bottom shows one row with the same data.

View Request	View Result	Batch ID	Start Time	End Time	Total Processing Time (ms)	API Active Processing Time (ms)	Apex Processing Time (ms)	Records Processed	Records Failed	Retry Count	State Message	Status
View Request		751gL00000Cf7oD	10/11/2025, 10:02 AM		5912	5036	3784	1,200	708	0	In Progress	

◆ 2. Data Loader

Use Case:

The Data Loader is used for **large-scale data operations** (up to 5 million records). It's perfect for admins or developers who need to **insert, update, upsert, delete, or export** records using CSV files.

Implementation Steps:

1. Install **Salesforce Data Loader** from Setup or Salesforce Help.
2. Log in using your Salesforce credentials (use **Security Token** if required).
3. Select the operation (Insert / Update / Upsert / Delete / Export).
4. Choose the target object and CSV file.
5. Map the fields and execute.
6. Review **Success** and **Error** logs after completion.

The image shows two side-by-side screenshots of the Salesforce Data Loader interface.

Step 2b: (Optional) relate using lookup field

Relationships of Complaint_c are listed below.
Select a related object and its lookup field if the CSV refers to the related object using the selected lookup field.

Relationship : Related Object	- Lookup Field of Related Object
Assigned_To_r : User	- Username
Contact_r : Contact	- Id
Customer_Name_r : Account	- Id
Owner : Group	- Name
RecordType : RecordType	- Id

Step 3: Mapping

Import batch size: 200 Start at row: 0
Current API usage for the org: 33
API Limit for the org: 15,000

Choose an Existing Map	Create or Edit a Map
CSV Column Header	Salesforce Object Field Name
Complaint ID	Complaint_ID_c
Record_Type	
Complaint Name	Name
Subject	Subject_c
Description	Description_c
Status	Status_c
Priority	Priority_c
Logged Date	Logged_Date_c
Resolution Date	Resolution_Date_c
Customer Email	Customer_Email_c
Record Owner	

< Back Next > Finish Cancel < Back Next > Finish Cancel

◆ 3. Duplicate Rules

Use Case:

Duplicate Rules help prevent **data redundancy** by identifying and blocking or alerting users when duplicate records are created — for example, when two complaints are logged for the same customer.

Implementation Steps:

1. Go to **Setup** → **Duplicate Rules**.
2. Click **New Rule** → select an object (e.g., **Complaint__c**).
3. Choose a **Matching Rule** (e.g., match by Email or Phone).
4. Set the **Action**: Allow / Block / Alert.
5. Activate the rule and test by creating duplicate data.

The screenshot shows the Salesforce Duplicate Rules page. The left sidebar has a 'Data' section with 'Duplicate Management' (selected), 'Duplicate Error Logs', 'Duplicate Rules' (selected), and 'Matching Rules'. A global search bar at the top says 'Search Setup'. The main area title is 'd SETUP Duplicate Rules'. It shows a 'Complaint Duplicate Rule' named 'Email'. The 'Duplicate Rule Detail' section includes fields like 'Rule Name' (Email), 'Object' (Complaint), 'Record-Level Security' (Enforce sharing rules), 'Action On Create' (Block), 'Action On Edit' (Allow), 'Alert Text' (Use one of these records?), 'Active' (checked), 'Matching Rule' (Email matching rule, Mapped), 'Conditions' (Complaint: Status EQUALS New), and 'Created By' (Samyak Jain, 10/8/2025, 1:19 PM). The 'Operations' section shows 'Operations On Create' (Alert, Report) and 'Operations On Edit' (Alert, Report). The 'Matching Criteria' field contains the formula 'Complaint: Name EXACT MatchBlank = TRUE'. At the bottom are 'Edit', 'Delete', 'Clone', and 'Deactivate' buttons.

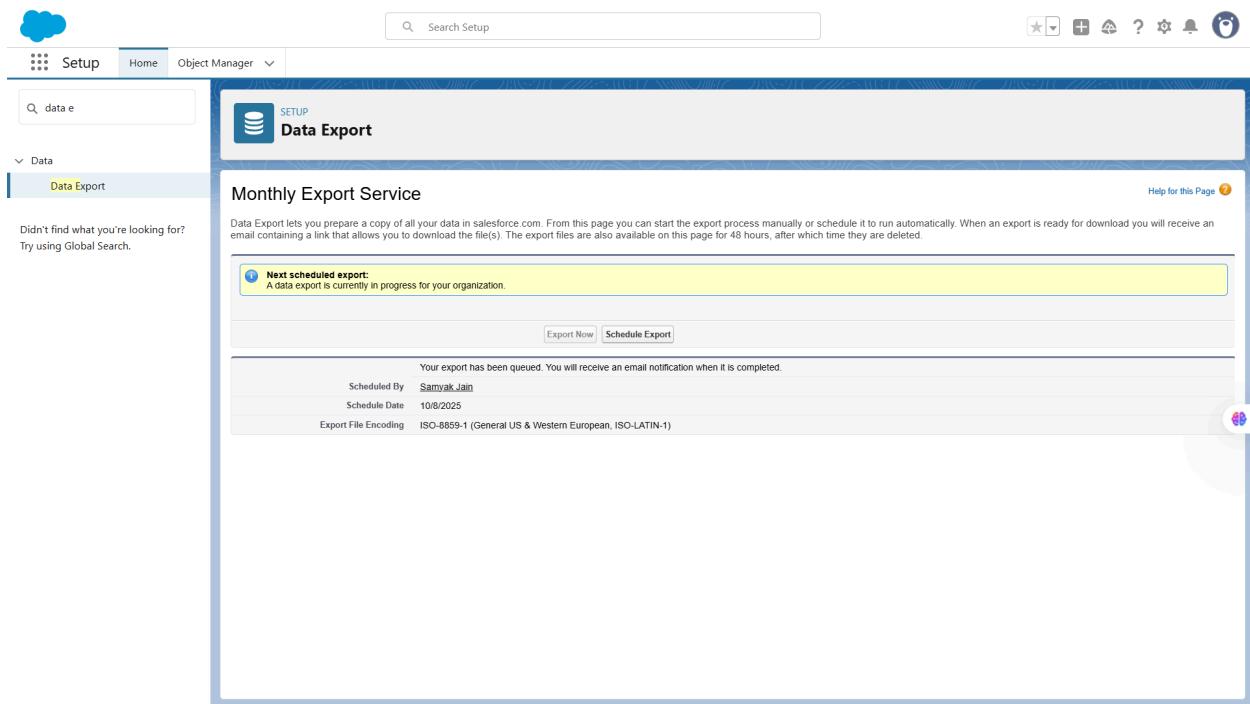
◆ 4. Data Export & Backup

Use Case:

Data Export ensures **regular backups of Salesforce data**, either manually or automatically, to prevent data loss. You can export all records, attachments, and files into **.zip** archives.

Implementation Steps:

1. Navigate to **Setup → Data Export**.
2. Choose **Export Now** or schedule **weekly/monthly exports**.
3. Select the objects (e.g., Complaints, Customers, Feedback).
4. Salesforce generates a downloadable **.zip** file.



◆ 5. Change Sets

Use Case:

Change Sets allow admins to **deploy customizations** (like objects, fields, validation rules, flows, LWCs, etc.) between **related orgs** (e.g., Sandbox → Production) without coding.

Change Sets are normally used to deploy components between Salesforce orgs. Since my org does not have outbound change sets enabled, I used VS Code to deploy my project components to the target org.

◆ 6. Unmanaged vs Managed Packages

Use Case:

Salesforce Packages group metadata and components for distribution.

- **Managed Packages** are for **commercial apps** (protected, upgradable).
- **Unmanaged Packages** are for **open or internal sharing** (editable).

Implementation Steps:

1. In **Setup → Package Manager**, create a new package.
2. Add components (objects, fields, LWCs, Apex).
3. Choose **Managed** (for AppExchange apps) or **Unmanaged** (for internal projects).

The screenshot shows the Salesforce Package Manager interface. On the left, there's a sidebar with a search bar containing 'packa', a 'Setup' button, and navigation links for 'Apps', 'Packaging', 'Installed Packages', 'Package Manager' (which is selected), and 'Package Usage'. A message at the bottom says 'Didn't find what you're looking for? Try using Global Search.' The main area has a title 'SETUP Package Manager' and a sub-section 'CustomerComplaintCRM_Package'. It shows the package's detail with fields like 'Package Name' (CustomerComplaintCRM_Package), 'Language' (English), 'Notify on Apex Error' (Samyak Jain), 'Created By' (Samyak Jain), and 'Description' (Complaint Management LWC and related metadata). There are 'Edit', 'Delete', and 'Upload' buttons. Below this, there are tabs for 'Components' and 'Versions', with 'Components' being active. The 'Components' tab displays a table of included components, each with an 'Action' column (e.g., 'All', 'Remove'), 'Component Name' (e.g., 'Feedback', 'Complaint Product', 'Complaint Feedback', 'Product', 'Complaint', 'CloseOldComplaints', 'Comments', 'Complaint', 'Complaint Escalated - Notify Manager', 'Complaint Feedback'), 'Parent Object' (e.g., 'Feedback', 'Complaint Product', 'Complaint Feedback', 'Product', 'Complaint', 'Apex Class', 'Custom Field', 'Custom Object', 'Custom Field', 'Complaint', 'Workflow Email Alert', 'Custom Object'), 'Type' (e.g., 'List View', 'List View', 'List View', 'List View', 'List View', 'Feedback', 'Complaint Product', 'Complaint Feedback', 'Product', 'Complaint', 'User Selected', 'Feedback', 'User Selected', 'Complaint Product', 'Complaint Feedback', 'Complaint Escalated - Notify Manager', 'Complaint Closure Automation'), 'Included By' (e.g., 'Feedback', 'Complaint Product', 'Complaint Feedback', 'Product', 'Complaint', 'Complaint', 'Complaint Handler', 'Complaint_Compact_Layout', 'NotifyAgentQueueable', 'Feedback', 'User Selected', 'Complaint Product', 'Complaint Feedback', 'Complaint Detail Page', 'Complaint Closure Automation'), and 'Owned By' (e.g., 'Feedback', 'Complaint Product', 'Complaint Feedback', 'Product', 'Complaint', 'Complaint', 'Complaint Handler', 'Complaint_Compact_Layout', 'NotifyAgentQueueable', 'Feedback', 'User Selected', 'Complaint Product', 'Complaint Feedback', 'Complaint Detail Page', 'Complaint Closure Automation').

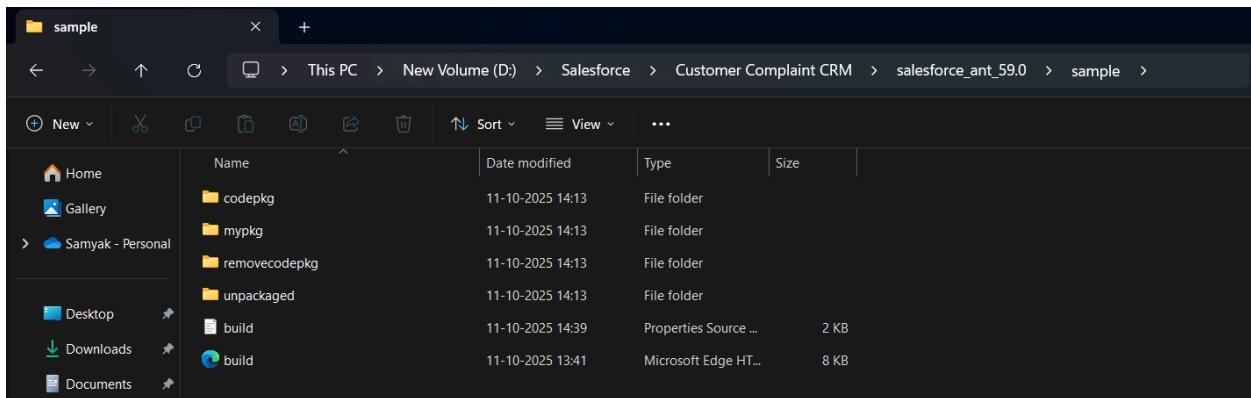
◆ 7. ANT Migration Tool

Use Case:

The **ANT Migration Tool** is a command-line utility for **automated deployments** between Salesforce orgs — best suited for developers managing large projects with version control.

Implementation Steps:

1. Install **Apache ANT** and **Salesforce Migration Tool (zip)**.
2. Create `build.xml` and `package.xml` to define metadata.
3. Run commands like `ant retrieve` and `ant deploy`.



◆ 8. VS Code & SFDX

Use Case:

Salesforce DX (SFDX) and **VS Code** integration streamline **development and deployment** using modern CLI-based tools. It's ideal for source-driven development, version control, and CI/CD automation.

Implementation Steps:

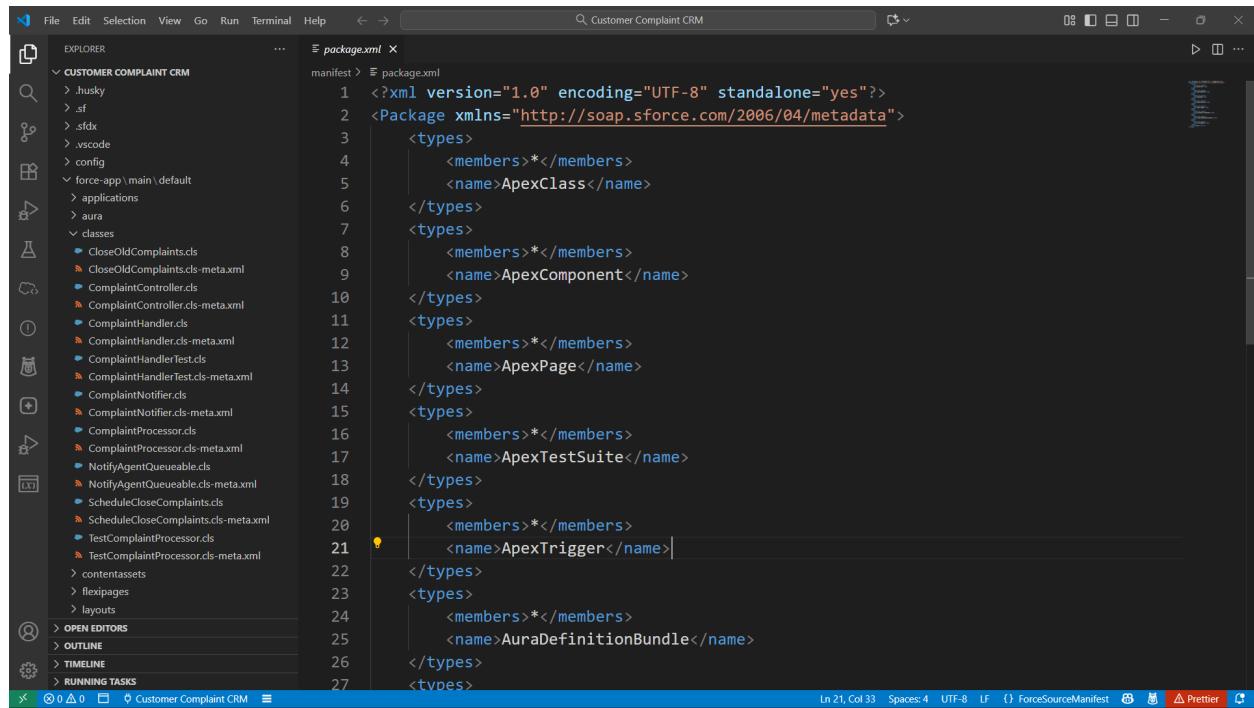
1. Install **VS Code** and **Salesforce Extensions Pack**.

Connect to your org using: `sfdx auth:web:login -a DevHub`

2. Create or open an SFDX project:
`sfdx force:project:create -n ComplaintCRM`

3. Retrieve or deploy metadata:

```
sfdx force:source:retrieve -m ApexClass sfdx force:source:deploy -m  
LightningComponentBundle
```



```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
<types>
    <members*></members>
    <name>ApexClass</name>
</types>
<types>
    <members*></members>
    <name>ApexComponent</name>
</types>
<types>
    <members*></members>
    <name>ApexPage</name>
</types>
<types>
    <members*></members>
    <name>ApexTestSuite</name>
</types>
<types>
    <members*></members>
    <name>ApexTrigger</name>
</types>
<types>
    <members*></members>
    <name>AuraDefinitionBundle</name>
</types>
<types>
```



Phase 9: Reporting, Dashboards & Security Review

◆ 1. Reports



Goal:

Reports allow Salesforce users to analyze, organize, and visualize data stored in the system. Each report type serves a specific purpose:

- **Tabular Reports:** Provide a simple list of records and are useful for exports or quick data views.
- **Summary Reports:** Group records by a field and show subtotals, enabling managers to see counts or sums by category.
- **Matrix Reports:** Allow grouping of data by rows and columns simultaneously, ideal for cross-tab analysis, like complaints by status vs. assigned agent.
- **Joined Reports:** Combine multiple report blocks from related or unrelated objects, useful for comparing different datasets side-by-side, e.g., complaints and their feedback.

Reports are the foundation for informed decision-making, operational oversight, and team performance tracking.

Use Case:

Reports help customer service managers analyze complaint trends, agent performance, and resolution times. For example, a “Complaints by Status” report shows how many complaints are New, In Progress, or Resolved.

The screenshot shows the Salesforce Reports interface. At the top, there's a navigation bar with links for Home, Dashboards, Complaints, Reports (which is currently selected), Complaint Products, and Complaint Feedbacks. Below the navigation is a search bar labeled "Search..." and a toolbar with icons for star, plus, download, help, and refresh. The main content area is titled "Recent" and shows a table of recent reports. The table has columns: Report Name, Description, Folder, Created By, Created On, and Subscribed. The data in the table includes:

Report Name	Description	Folder	Created By	Created On	Subscribed
Joined Report		Private Reports	Samyak Jain	10/11/2025, 11:27 AM	
Sample Flow Report: Screen Flows	Which flows run, what's the status of each interview, and how long do users take to complete the screens?	Public Reports	Automated Process	9/20/2025, 2:12 AM	
Matrix Report		Private Reports	Samyak Jain	10/11/2025, 11:12 AM	
Summary Report		Private Reports	Samyak Jain	10/11/2025, 10:59 AM	
Tabular - Complaints		Private Reports	Samyak Jain	10/11/2025, 10:37 AM	
Complaints by Agent & Priority		Public Reports	Samyak Jain	9/25/2025, 5:25 AM	

On the left side, there's a sidebar with sections for Reports (Recent, Created by Me, Private Reports, Public Reports, All Reports), Folders (All Folders), and Favorites (All Favorites). A "New Report" button is located at the top right of the main content area.

◆ 2. Report Types



Report Types define **which objects and fields are available** when creating reports. They act as templates that determine what data can be accessed and how objects are related.

- Standard report types are provided by Salesforce for common objects.
- Custom report types can be created to include related custom objects or specific fields not available in standard types.

Report Types ensure that users can build meaningful reports while maintaining data relevance and integrity.

Use Case:

Defines which objects and relationships are available for reporting.

Example: To combine “Complaints” and “Feedback,” you need a **custom report type**.

Types:

- **Tabular:** Simple list of records (e.g., all complaints created this week).
- **Summary:** Groups records by field (e.g., complaints grouped by Status or Priority).

- **Matrix:** Groups both rows and columns (e.g., complaints by Status and Agent).
- **Joined:** Combines multiple report types (e.g., Complaints + Feedback).

Customer Complaints - Tabular - Complaints

Total Records: 1,976

	Complaint ID	Complaint: Record Type	Complaint: Complaint Name	Subject	Description	Status	Priority	Logged Date	Resolution Date	Customer Email
1	A-0004	Email Complaint	Samyak	-	-	Escalated	High	9/26/2025	9/29/2025	-
2	A-0002	Phone Complaint	Samman	-	-	In Progress	Medium	9/26/2025	9/28/2025	-
3	A-0003	Phone Complaint	Samyak1	-	-	Escalated	High	9/26/2025	9/30/2025	-
4	CMP-00001	Phone Complaint	Customer 1	Late Delivery	Late Delivery reported by customer in Hyderabad.	Resolved	Medium	4/28/2025	5/1/2025	customer1@e...
5	CMP-00004	Phone Complaint	Customer 4	Warranty Claim	Warranty Claim reported by customer in Mumbai.	Resolved	High	6/15/2025	6/20/2025	customer4@e...
6	CMP-00007	Phone Complaint	Customer 7	Late Delivery	Late Delivery reported by customer in Bangalore.	Resolved	High	5/21/2025	5/29/2025	customer7@e...
7	CMP-00008	Phone Complaint	Customer 8	Product Defective	Product Defective reported by customer in Mumbai.	Closed	High	1/25/2025	2/2/2025	customer8@e...
8	CMP-00010	Phone Complaint	Customer 10	Refund Request	Refund Request reported by customer in Chennai.	Closed	High	4/17/2025	4/26/2025	customer10@e...
9	CMP-00012	Phone Complaint	Customer 12	Warranty Claim	Warranty Claim reported by customer in Indore.	Resolved	Medium	6/3/2024	6/13/2024	customer12@e...
10	CMP-00013	Phone Complaint	Customer 13	Product Defective	Product Defective reported by customer in Bhopal.	Resolved	Medium	8/31/2024	9/7/2024	customer13@e...
11	CMP-00022	Phone Complaint	Customer 22	Return Pickup Delay	Return Pickup Delay reported by customer in Indore.	Resolved	Medium	2/11/2025	2/19/2025	customer22@e...
12	CMP-00025	Phone Complaint	Customer 25	Warranty Claim	Warranty Claim reported by customer in Indore.	Closed	Medium	2/1/2024	2/9/2024	customer25@e...
13	CMP-00027	Phone Complaint	Customer 27	Refund Request	Refund Request reported by customer in Hyderabad.	Closed	Medium	5/4/2025	5/14/2025	customer27@e...
14	CMP-00028	Phone Complaint	Customer 28	Customer Service	Customer Service reported by customer in Delhi.	Closed	High	3/28/2025	4/3/2025	customer28@e...
15	CMP-00033	Phone Complaint	Customer 33	Refund Request	Refund Request reported by customer in Lucknow.	Resolved	Low	5/4/2024	5/12/2024	customer33@e...
16	CMP-00034	Phone Complaint	Customer 34	Billing Issue	Billing Issue reported by customer in Chennai.	Closed	High	5/7/2025	5/13/2025	customer34@e...
17	CMP-00035	Phone Complaint	Customer 35	Billing Issue	Billing Issue reported by customer in Mumbai.	Closed	High	5/17/2024	5/18/2024	customer35@e...
18	CMP-00037	Phone Complaint	Customer 37	Order Cancellation	Order Cancellation reported by customer in Hyderabad.	Resolved	Medium	1/14/2024	1/17/2024	customer37@e...
19										

Chatter Feed

Customer Complaints - Summary Report

Total Records: 1,976

	Status	Priority	Complaint ID	Complaint: Complaint Name	Subject	Description	Logged Date	Resolution Date	Customer Email	Complaint: (
<input type="checkbox"/> In Progress (1)	Medium (1)	A-0002	Samman	-	-	9/26/2025	9/28/2025	-	Samyak Jain	
Subtotal										
<input type="checkbox"/> Escalated (2)	High (2)	A-0004	Samyak	-	-	9/26/2025	9/29/2025	-	Samyak Jain	
Subtotal										
<input type="checkbox"/> Resolved (1489)	Low (300)	CMP-00033	Customer 33	Refund Request	Refund Request reported by customer in Lucknow.	5/4/2024	5/12/2024	customer33@example.com	Samyak Jain	
		CMP-00049	Customer 49	Billing Issue	Billing Issue reported by customer in Jaipur.	1/27/2025	1/29/2025	customer49@example.com	Samyak Jain	
		CMP-00056	Customer 56	Return Pickup Delay	Return Pickup Delay reported by customer in Lucknow.	12/27/2024	12/28/2024	customer56@example.com	Samyak Jain	
		CMP-00079	Customer 79	Billing Issue	Billing Issue reported by customer in Lucknow.	1/16/2024	1/26/2024	customer79@example.com	Samyak Jain	
		CMP-00092	Customer 92	Return Pickup Delay	Return Pickup Delay reported by customer in Lucknow.	9/26/2025	9/29/2025	customer92@example.com	Samyak Jain	
		CMP-00132	Customer 132	Return Pickup Delay	Return Pickup Delay reported by customer in Lucknow.	8/8/2024	8/18/2024	customer132@example.com	Samyak Jain	
		CMP-00195	Customer 195	Customer Service	Customer Service reported by customer in Lucknow.	3/7/2024	3/12/2024	customer195@example.com	Samyak Jain	
		CMP-00201	Customer 201	Order Cancellation	Order Cancellation reported by customer in Jaipur.	4/28/2025	5/1/2025	customer201@example.com	Samyak Jain	
		CMP-00202	Customer 202	Customer Service	Customer Service reported by customer in Lucknow.	8/19/2024	8/28/2024	customer202@example.com	Samyak Jain	
		CMP-00209	Customer 209	Return Pickup Delay	Return Pickup Delay reported by customer in Lucknow.	7/16/2025	7/19/2025	customer209@example.com	Samyak Jain	

Chatter Feed

Customer Complaints Matrix Report

Total Records: 1,976

Status	Assigned To	Samayak Jain	Total
<input type="checkbox"/> In Progress	Record Count	1	1
<input type="checkbox"/> Escalated	Record Count	2	2
<input type="checkbox"/> Resolved	Record Count	1,489	1,489
<input type="checkbox"/> Closed	Record Count	484	484
Total	Record Count	1,976	1,976

Details (1976 Rows) Click an intersection in the table above to filter details.

	Complaint: Complaint Name	Complaint: Created Date	Priority
1	Samman	9/26/2025	Medium
2	Samayak	9/26/2025	High
3	Samayak1	9/26/2025	High
4	Customer 1	10/11/2025	Medium
5	Customer 12	10/11/2025	Medium
6	Customer 22	10/11/2025	Medium
7	Customer 13	10/11/2025	Medium
8	Customer 32	10/11/2025	Low

Row Counts: Detail Rows: Grand Total: Stacked Summaries:

Chatter Feed

Joined Report

Complaints		Feedbacks			
Complaints block 1		Feedbacks block 1			
Complaint: Created Date	Status	Feedback: Feedback Name	Feedback Date	Rating	Comments
9/26/2025	In Progress	FDB-00001	7/23/2025	3	Issue still pending
9/26/2025	Escalated	FDB-00002	8/26/2025	3	Waiting for update
9/26/2025	Escalated	FDB-00003	8/29/2025	2	Waiting for update
10/11/2025	Resolved	FDB-00004	8/27/2025	5	Good response time
10/11/2025	Resolved	FDB-00005	7/12/2025	3	Waiting for update
10/11/2025	Resolved	FDB-00006	12/9/2024	3	Waiting for update
10/11/2025	Closed	FDB-00007	6/18/2025	3	Waiting for update
10/11/2025	Closed	FDB-00008	8/4/2025	1	Poor communication
10/11/2025	Resolved	FDB-00009	1/8/2025	5	Service was excellent
10/11/2025	Resolved	FDB-00010	9/21/2025	3	Poor communication
10/11/2025	Resolved	FDB-00011	11/6/2024	2	Not satisfied
10/11/2025	Closed	FDB-00012	6/6/2025	5	Issue resolved quickly
10/11/2025	Closed	FDB-00013	1/4/2025	1	Waiting for update
10/11/2025	Closed	FDB-00014	12/21/2024	2	Not satisfied
10/11/2025	Resolved	FDB-00015	9/15/2025	4	Issue resolved quickly
10/11/2025	Closed	FDB-00016	9/16/2025	2	Issue still pending
10/11/2025	Closed	FDB-00017	8/8/2025	5	Issue resolved quickly
10/11/2025	Resolved	FDB-00018	4/18/2025	3	Waiting for update

Chatter Feed

◆ 3. Dashboards

Goal:

Dashboards provide a **visual representation of report data** using charts, tables, and metrics in one consolidated view.

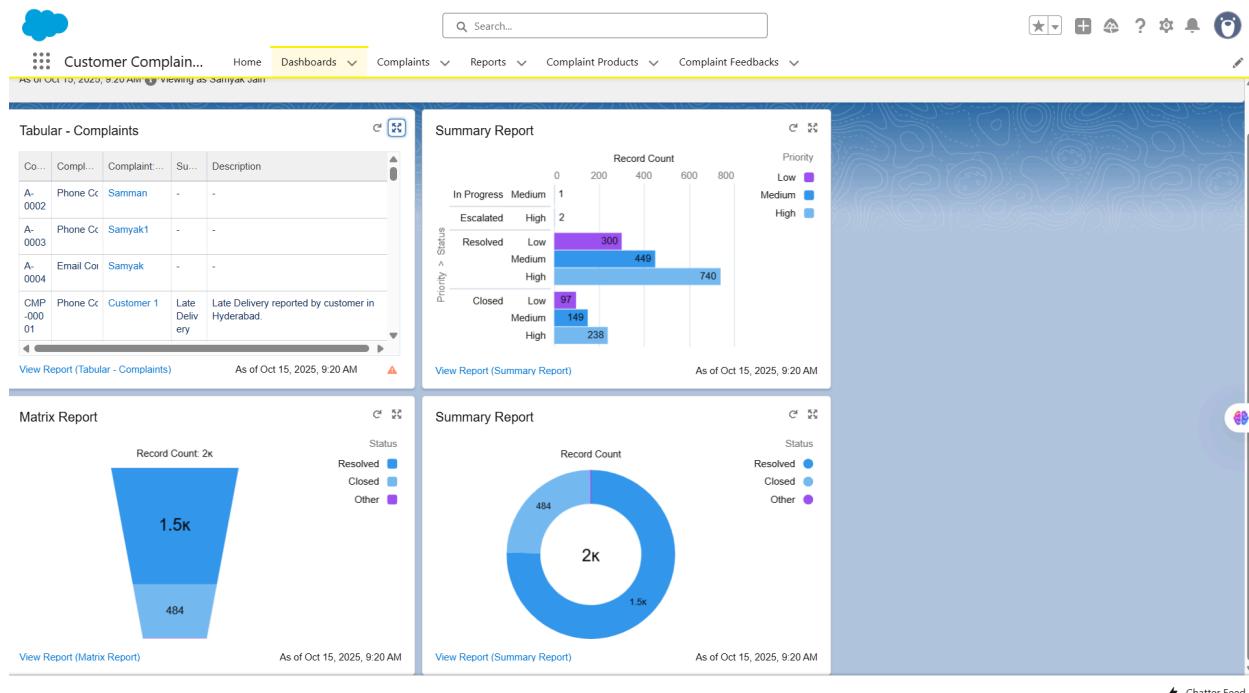
- They allow managers and executives to monitor KPIs, team performance, and operational metrics at a glance.
- Dashboards are composed of **dashboard components** (e.g., pie charts, bar charts, tables) that are linked to underlying reports.

Dashboards improve data-driven decision-making and enable real-time tracking of business processes.

Use Case:

Dashboards visualize key metrics from multiple reports. For example, a **Complaint Management Dashboard** can show:

- Tabular Complaint (Lightning Table)
- Summary Report (Horizontal Bar Chart)
- Matrix Report (Funnel Chart)
- Summary Report (Donut Chart)



◆ 4. Dynamic Dashboards

Goal:

Dynamic Dashboards allow users to view the **same dashboard with data filtered according to their own access level**.

- Unlike standard dashboards, which display data based on a fixed running user, dynamic dashboards adapt to the viewer's permissions and role hierarchy.
- This is especially useful for managers and agents who need personalized visibility without creating multiple copies of the same dashboard.

Dynamic dashboards enhance security and usability while providing tailored insights for each user.

Use Case:

Allows users to view dashboard data **as themselves** (based on their permissions), without needing multiple copies for each user.

Example: Each agent can see *only their own complaints* in the same dashboard.

The screenshot shows the 'Customer Complaint Dashboard' in the Salesforce interface. A modal dialog titled 'Properties' is open, allowing the user to edit the dashboard's name, description, and folder. The 'Name' field is set to 'Customer Complaint Dashboard'. The 'Folder' field is set to 'Private Dashboards'. Under the 'View Dashboard As' section, the radio button 'The dashboard viewer' is selected. A checked checkbox 'Let dashboard viewers choose whom they view the dashboard as' is also present. The 'Dashboard Grid Size' section shows '12 columns (recommended)' selected. At the bottom of the dialog are 'Cancel' and 'Save' buttons. In the background, the dashboard displays two reports: 'Tabular - Complaints' showing a table of complaints with columns like Complaint ID, Complainant Name, and Description, and 'Matrix Report' showing a funnel chart with a record count of 2k, 1.5k, and 484.

◆ 5. Profiles



Assign the custom Complaint Agent Profile (or any profile) to your Support Agent users so they get the correct permissions.

Use Case:

Profiles define **base-level access** — which objects, fields, tabs, and apps a user can see or edit.

Example: Agents can edit Complaints but not modify Feedback.

The screenshot shows the Salesforce Setup interface with the 'Profiles' page selected. The 'Profile Detail' section displays the 'Complaint Agent Profile' with a 'Name' of 'Complaint Agent Profile' and a 'User License' of 'Salesforce'. The 'Description' field is empty. The 'Created By' field shows 'Samyak Jain' with a timestamp of '10/15/2025, 9:25 AM'. The 'Modified By' field also shows 'Samyak Jain' with a timestamp of '10/15/2025, 9:27 AM'. The 'Page Layouts' section lists various standard object layouts for different record types like Global, Location Group, and Opportunity. The 'Standard Object Layouts' table includes rows for Global, Email Application, Home Page Layout, Account, Alternative Payment Method, Appointment Invitation, Asset, Asset Action, Asset Action Source, and Asset Relationship. Each row shows the layout name and a link to 'View Assignment'.

◆ 6. Roles



Create a clear role hierarchy so managers can see the records owned by users below them (for example, Support Managers can view all Support Agents' complaints).

Use Case:

Roles determine **record visibility** in the role hierarchy (who can see whose records).

Example: Support Manager can see complaints of all agents under them.

The screenshot shows the Salesforce Setup interface with the 'Roles' page selected. The left sidebar has a search bar and navigation links for 'Users', 'Sales', 'Service', and 'Case Teams'. The main content area is titled 'Creating the Role Hierarchy' and shows a tree view of roles under 'Your Organization's Role Hierarchy'. The tree includes the following nodes:

- Tata Consultancy Services
 - Add Role
 - CEO (Edit | Del | Assign)
 - Add Role
 - CFO (Edit | Del | Assign)
 - Add Role
 - COO (Edit | Del | Assign)
 - Add Role
 - Support Manager (Edit | Del | Assign)
 - Add Role
 - Support Agents (Edit | Del | Assign)
 - Add Role
 - SVP_Customer Service & Support (Edit | Del | Assign)
 - Add Role
 - SVP_Human Resources (Edit | Del | Assign)
 - Add Role
 - SVP_Sales & Marketing (Edit | Del | Assign)
 - Add Role
 - Manager (Edit | Del | Assign)
 - Add Role

◆ 7. Users



Create new Salesforce users (e.g., Support Agents or Managers) and assign them their Profile (permissions) and Role (visibility level).

Use Case:

Each user has login credentials and is assigned a profile & role.

Example: Create users for each agent or admin.

The screenshot shows the Salesforce Setup interface with the following details:

- Setup Tab:** The top navigation bar includes the Setup tab, Home, and Object Manager.
- Search Bar:** A search bar at the top right contains the text "Search Setup".
- User Detail Page:** The main content area is titled "User Detail" for "Shubh Sahu". It shows the following information:

Name	Shubh Sahu	Role	Agent
Alias	ssahu	User License Profile	Salesforce Platform Standard Platform User
Email	shubh2104@gmail.com [Verify] View	Active	<input checked="" type="checkbox"/>
Username	shubh2104@gmail.com	Marketing User	<input type="checkbox"/>
Nickname	shubh21 View	Offline User	<input type="checkbox"/>
Title		Knowledge User	<input type="checkbox"/>
Company		Flow User	<input type="checkbox"/>
Department		Service Cloud User	<input type="checkbox"/>
Division		Site.com Contributor User	<input type="checkbox"/>
Address		WDC User	<input type="checkbox"/>
Time Zone	(GMT+05:30) India Standard Time (Asia/Kolkata)	Mobile Push Registrations	View
Locale	English (United States)	Data.com User Type	View
Language	English	Accessibility Mode (Classic Only)	<input type="checkbox"/> View
Delegated Approver		Debug Mode	<input type="checkbox"/> View
Manager		High-Contrast Palette on Charts	<input type="checkbox"/> View
Receive Approval Request Emails	Only if I am an approver	Load Lightning Pages While Scrolling	<input checked="" type="checkbox"/> View
Federation ID		Salesforce CRM Content User	<input checked="" type="checkbox"/>
App Registration: One-Time Password Authentication	View	Receive Salesforce CRM Content Email Alerts	<input checked="" type="checkbox"/>
App Registration: Salesforce Authenticator	View	Receive Salesforce CRM Content Alerts as Daily Digest	<input checked="" type="checkbox"/>
Security Key (UI2F or WebAuthn)	View		
Lightning Login	View		
- Left Sidebar:** The sidebar lists various categories such as Permission Set Groups, Permission Sets, Profiles, Public Groups, Queues, Roles, User Management Settings, and Users.
- Bottom Left:** A note says "Didn't find what you're looking for? Try using Global Search."

◆ 8. Permission Sets



Goal: Create a Permission Set that grants extra access (e.g., Read/Edit on Feedback__c) without modifying the user's profile.

Use Case:

Provides additional permissions without changing a user's profile.

Example: Temporarily grant a Support Agent access to the Feedback object.

The screenshot shows the Salesforce Setup interface with the following details:

- Left Sidebar:** Includes links for Setup, Home, Object Manager, and a search bar labeled "Search Setup".
- Permission Set Groups:** A section containing "Users", "Permission Set Groups", and "Permission Sets".
- Custom Code:** A section containing "Custom Code" and "Custom Permissions".
- Global Search:** A note saying "Didn't find what you're looking for? Try using Global Search."

Permission Sets Page:

- Header:** "SETUP" and "Permission Sets".
- Breadcrumbs:** "Permission Set Overview > Object Settings > Feedbacks".
- Section:** "Feedback Access".
- Buttons:** "Find Settings", "Clone", "Delete", "Edit Properties", "Manage Assignments", "View Summary".
- Help:** "Video Tutorial | Help for this Page".
- Feedbacks Tab:** "Tab Settings" with "Available" set to "Visible".
- Object Permissions:** A table showing permissions for the Feedback object.

Permission Name	Enabled
Read	<input checked="" type="checkbox"/>
Create	<input type="checkbox"/>
Edit	<input checked="" type="checkbox"/>
Delete	<input type="checkbox"/>
View All Records	<input type="checkbox"/>
Modify All Records	<input type="checkbox"/>
View All Fields	<input type="checkbox"/>
- Field Permissions:** A table showing permissions for fields on the Feedback object.

Field Name	Field API Name	Read Access	Edit Access
Comments	Comments__c	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Complaint Lookup	Complaint_Lookup__c	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Created By	CreatedBy	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Feedback Date	Feedback_Date__c	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Feedback Name	Name	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Last Modified By	LastModifiedBy	<input checked="" type="checkbox"/>	<input type="checkbox"/>

◆ 9. OWD (Organization-Wide Defaults)



Restrict access to Complaint records so that only the record owner and their managers can view or edit them (others cannot).

Use Case:

Controls **baseline record access** for all users.

Example: Complaints = Private (only owner and superiors can see).

◆ 10. Sharing Rules

Note:

In this project, a **Region** field was **not included** in the **Complaint__c** object, as the business process does not require regional segregation of data. All complaints are handled centrally by the customer service team, regardless of geographic or departmental divisions.

Use Case:

Since the **Region__c** field is not present, the sharing model is based on **user roles** instead of data criteria.

To ensure that managers can monitor their team's complaints while maintaining privacy across departments, sharing rules were configured to share complaint records **owned by users** in the **“Customer Support Agent” role** with users in the **“Customer Service Manager” role**.

Summary:

- **Region Field:** Not added (centralized complaint management system)
- **Rule Type:** Based on record owner
- **Share From:** Role = Customer Support Agent

- **Share With:** Role = Customer Service Manager
 - **Access Level:** Read/Write
 - **Purpose:** Provides team-level visibility without needing regional data segmentation.
-

◆ 11. Sharing Settings



To review and set up Organization-Wide Defaults (OWD) for your project's custom objects — mainly Complaint__c and Feedback__c — so that only authorized users can access the records.

Use Case:

Central control panel for all sharing behaviors — OWD, Sharing Rules, and Manual Sharing.

	OWD	Private
Service Resource	Public Read/Write	Private
Service Territory	Public Read/Write	Private
Shift	Private	Private
Shipment	Private	Private
Shipping Carrier	Public Read Only	Private
Shipping Carrier Method	Public Read Only	Private
Shipping Configuration Set	Public Read Only	Private
Streaming Channel	Public Read/Write	Private
Tableau Host Mapping	Public Read Only	Private
User Presence	Public Read Only	Private
Waitlist	Private	Private
Web Cart Document	Private	Private
Work Order	Private	Private
Work Plan	Private	Private
Work Plan Template	Private	Private
Work Step Template	Private	Private
Work Type	Private	Private
Work Type Group	Public Read/Write	Private
Complaint	Private	Private
Feedback	Private	Private
Product	Public Read/Write	Private

Other Settings

Standard Report Visibility Manual User Record Sharing Manager Groups Secure guest user record access Require permission to view record names in lookup fields

Save Cancel

◆ 12. Field Level Security (FLS)



Restrict access to sensitive fields so only certain profiles (like Managers) can view or edit them.

Use Case:

Restricts visibility/editing of specific fields.

Example: “Customer Email” field is visible only to Managers.

User Profile	Action 1	Action 2
Custom: Marketing Profile	<input type="checkbox"/>	<input type="checkbox"/>
Custom: Sales Profile	<input type="checkbox"/>	<input type="checkbox"/>
Custom: Support Profile	<input type="checkbox"/>	<input type="checkbox"/>
Einstein Agent User	<input type="checkbox"/>	<input type="checkbox"/>
Force.com - App Subscription User	<input type="checkbox"/>	<input type="checkbox"/>
Force.com - Free User	<input type="checkbox"/>	<input type="checkbox"/>
Gold Partner User	<input type="checkbox"/>	<input type="checkbox"/>
Identity User	<input type="checkbox"/>	<input type="checkbox"/>
Manager Profile	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Marketing User	<input type="checkbox"/>	<input type="checkbox"/>
Minimum Access - API Only Integrations	<input type="checkbox"/>	<input type="checkbox"/>
Minimum Access - Salesforce	<input type="checkbox"/>	<input type="checkbox"/>
Partner App Subscription User	<input type="checkbox"/>	<input type="checkbox"/>
Partner Community Login User	<input type="checkbox"/>	<input type="checkbox"/>
Partner Community User	<input type="checkbox"/>	<input type="checkbox"/>
Read Only	<input type="checkbox"/>	<input type="checkbox"/>
Salesforce API Only System Integrations	<input type="checkbox"/>	<input type="checkbox"/>
Silver Partner User	<input type="checkbox"/>	<input type="checkbox"/>
Solution Manager	<input type="checkbox"/>	<input type="checkbox"/>
Standard Platform User	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Standard User	<input checked="" type="checkbox"/>	<input type="checkbox"/>
System Administrator	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Work.com Only User	<input type="checkbox"/>	<input type="checkbox"/>

◆ 13. Session Settings



Control user session behavior, including timeouts, security policies, and login restrictions to improve org security.

Use Case:

Enhances login security — timeout duration, trusted IPs, browser security.

Session Settings

Set the session security and session expiration timeout for your organization.

Session Timeout

Timeout Value: 2 hours

Disable session timeout warning popup
 Force logout on session timeout

Session Settings

Lock sessions to the IP address from which they originated
 Lock sessions to the domain in which they were first used
 Terminate all of a user's sessions when an admin resets that user's password
 Force relogin after Login-As-User
 Require `HttpOnly` attribute
 Use POST requests for cross-domain sessions
 Enforce login IP ranges on every request
 When embedding a Lightning application in a third-party site, use a session token instead of a session cookie

Extended use of IE11 with Lightning Experience

EXTENDED USE OF IE11 WITH LIGHTNING EXPERIENCE HAS NOW ENDED
AS OF DECEMBER 31, THE EXTENDED PERIOD HAS ENDED, AND USE OF INTERNET EXPLORER 11 (IE 11) WITH LIGHTNING EXPERIENCE IS NO LONGER SUPPORTED. ISSUES WITH PERFORMANCE OR FUNCTIONALITY THAT AFFECT ONLY IE 11 WILL NOT BE FIXED. PLEASE SWITCH TO A SUPPORTED BROWSER.

Caching

Enable caching and autocomplete on login page
 Enable secure and persistent browser caching to improve performance
 Enable user switching
 Remember me until logout
 Enable Content Delivery Network (CDN) for Lightning Component framework

◆ 14. Login IP Ranges



Control user session behavior, including timeouts, security policies, and login restrictions to improve org security.

Use Case:

Restricts user logins to company IPs.

Example: Agents can log in only from office network.

Profiles

Action	IP Start Address	IP End Address	Description
Edit Del	49.43.40.163	49.43.40.225	Users can log in from any IP within this range.

Enabled Apex Class Access

Apex Class Name: `devdataap.DeveloperEditionUtils`
`devdataap.PostInstallScript`

Enabled Visualforce Page Access

No Visualforce Pages enabled

Enabled External Data Source Access

No External Data Sources enabled

Enabled Named Credential Access

No Named Credential enabled

Enabled External Credential Principal Access

No External Credential Principals enabled

◆ 15. Audit Trail



Restrict Salesforce logins so that users can only access the org from approved IP addresses, enhancing security for sensitive data.

Use Case:

The Setup Audit Trail was reviewed to monitor administrative changes in the Salesforce org. It tracks all modifications to profiles, roles, sharing settings, objects, and fields, providing a secure log for accountability and compliance. Audit trail downloads were retained for historical reference and audit purposes.

D1	A	B	C	D	E	F	G
	Date	User	Action	Section			
1	10/15/2025, 10:44:49 AM PDT	sjsam92268@agentforce.com	Added Login Ip Range to Agent Profile from 49.43.40.183 to 49.43.40.225	Manage Users			
2	10/15/2025, 10:05:34 AM PDT	sjsam92268@agentforce.com	Finished Organization-Wide Defaults update	Sharing Defaults			
4	10/15/2025, 10:05:25 AM PDT	sjsam92268@agentforce.com	Changed default internal access for Complaint from Public Read/Write to Private	Sharing Defaults			
5	10/15/2025, 10:05:13 AM PDT	sjsam92268@agentforce.com	Started Organization-Wide Defaults update	Sharing Defaults			
6	10/15/2025, 10:05:13 AM PDT	sjsam92268@agentforce.com	Started default internal access update for Complaint from Public Read/Write to Private	Sharing Defaults			
7	10/15/2025, 10:01:51 AM PDT	sjsam92268@agentforce.com	Changed permission set Feedback Access: Complaint Feedback object permissions were changed from No	Manage Users			
8	10/15/2025, 10:01:51 AM PDT	sjsam92268@agentforce.com	Changed permission set Feedback Access: Complaint object permissions were changed from No	Manage Users			
9	10/15/2025, 10:01:51 AM PDT	sjsam92268@agentforce.com	Changed permission set Feedback Access: field-level security for Complaint Feedback: Resolution	Manage Users			
10	10/15/2025, 10:01:51 AM PDT	sjsam92268@agentforce.com	Changed permission set Feedback Access: field-level security for Complaint Feedback: Rating w	Manage Users			
11	10/15/2025, 10:01:51 AM PDT	sjsam92268@agentforce.com	Changed permission set Feedback Access: field-level security for Complaint Feedback: Feedback w	Manage Users			
12	10/15/2025, 10:01:51 AM PDT	sjsam92268@agentforce.com	Changed permission set Feedback Access: field-level security for Complaint Feedback: Customi	Manage Users			
13	10/15/2025, 10:00:37 AM PDT	sjsam92268@agentforce.com	Changed permission set Feedback Access: assigned to user agent (UserId: [005g.000008K2k])	Manage Users			
14	10/15/2025, 9:59:39 AM PDT	sjsam92268@agentforce.com	Changed permission set Feedback Access: Feedback object permissions were changed from No	Manage Users			
15	10/15/2025, 9:59:39 AM PDT	sjsam92268@agentforce.com	Changed permission set Feedback Access: field-level security for Feedback: Rating was changed	Manage Users			
16	10/15/2025, 9:59:39 AM PDT	sjsam92268@agentforce.com	Changed permission set Feedback Access: field-level security for Feedback: Feedback Date was	Manage Users			
17	10/15/2025, 9:59:39 AM PDT	sjsam92268@agentforce.com	Changed permission set Feedback Access: field-level security for Feedback: Complaint Lookup	Manage Users			
18	10/15/2025, 9:59:39 AM PDT	sjsam92268@agentforce.com	Changed permission set Feedback Access: field-level security for Feedback: Comments was cha	Manage Users			
19	10/15/2025, 9:57:59 AM PDT	sjsam92268@agentforce.com	Created permission set Feedback Access: with no license	Manage Users			
20	10/15/2025, 9:44:48 AM PDT	sjsam92268@agentforce.com	Created new user Shubh Sahu	Manage Users			
21	10/15/2025, 9:33:29 AM PDT	sjsam92268@agentforce.com	Created new role Support Agent	Manage Users			
22	10/15/2025, 9:33:10 AM PDT	sjsam92268@agentforce.com	Created new role Support Manager	Manage Users			
23	10/15/2025, 9:29:20 AM PDT	sjsam92268@agentforce.com	Changed profile for user agent from Minimum Access - Salesforce to Complaint Agent Profile	Manage Users			
24	10/15/2025, 9:29:20 AM PDT	sjsam92268@agentforce.com	Changed role for user agent from <None Specified> to Agent	Manage Users			
25	10/15/2025, 9:26:22 AM PDT	sjsam92268@agentforce.com	Changed profile Complaint Agent Profile: Complaint object permissions were changed from No	Manage Users			
26	10/15/2025, 9:25:25 AM PDT	sjsam92268@agentforce.com	Created profile Complaint Agent Profile: Cloned from profile Standard User	Manage Users			
27	10/11/2025, 11:07:50 AM PDT	sjsam92268@agentforce.com	Added standard button override: Tab (Lightning Page null)	Standard Buttons and Links			
28	10/11/2025, 11:07:35 AM PDT	sjsam92268@agentforce.com	Created Lightning Page: Home Page Default	Lightning Pages			
29	10/11/2025, 10:01:03 AM PDT	sjsam92268@agentforce.com	Dropped external identifier designation: Complaint.Complaint ID	Company Information			
30	10/11/2025, 10:01:03 AM PDT	sjsam92268@agentforce.com	Changed field type of Complaint ID custom field on Complaints from Auto Number to Text	Custom Objects			
31	10/11/2025, 3:25:27 AM PDT	salesforce.com, inc.	Max number of streaming topics				
32	10/11/2025, 2:07:45 AM PDT	salesforce.com, inc.	ASalesforce employee ran Update Web Cart Views in Update mode for case Id 00000000000000000000	Salesforce Administrator			
33	10/11/2025, 2:07:45 AM PDT	salesforce.com, inc.	ASalesforce employee ran Update Usage Metrics Releasable Action 258 : 0 in Update mode for case Id 00	Salesforce Administrator			
34	10/11/2025, 2:07:44 AM PDT	salesforce.com, inc.	ASalesforce employee ran Slack RC Mapping Team Id Backfill Releasable Action 258 in Update mode for case Id 00	Salesforce Administrator			
35	10/11/2025, 2:07:44 AM PDT	salesforce.com, inc.	ASalesforce employee ran Update Standard Flow Timer Trigger Data in Update mode for case Id 00	Salesforce Administrator			
36	10/11/2025, 2:07:43 AM PDT	salesforce.com, inc.	ASalesforce employee ran Shield App Release Action in Update mode for case Id 00000000000000000000000000000000	Salesforce Administrator			
37	10/11/2025, 2:07:42 AM PDT	salesforce.com, inc.	ASalesforce employee ran Auto Activate Release Updates in Update mode for case Id 00000000000000000000000000000000	Salesforce Administrator			
38	10/11/2025, 2:07:42 AM PDT	salesforce.com, inc.	ASalesforce employee ran Create ProductRelationshipType Records and All List View in Update mode for case Id 00000000000000000000000000000000	Salesforce Administrator			
39	10/11/2025, 2:07:42 AM PDT	salesforce.com, inc.	ASalesforce employee ran Payment Credit Process Exception Releasable Action in Update mode for case Id 00000000000000000000000000000000	Salesforce Administrator			



Phase 10: Quality Assurance Testing

Introduction

This final phase focuses on verifying the accuracy, performance, and reliability of all features implemented in the **Customer Complaint CRM Project**.

Each Salesforce functionality—ranging from record creation to automation and reporting—has been tested through structured test cases.

Each test case includes:

- Input details (what data/action was tested)
- Expected Output (anticipated behavior)
- Actual Output (with result and screenshot)

Test Case Format

We will use the following structured format for all test cases:

Use Case / Scenario	Test Steps (with input)	Expected Result	Actual Result (with Screenshot)
---------------------	-------------------------	-----------------	---------------------------------

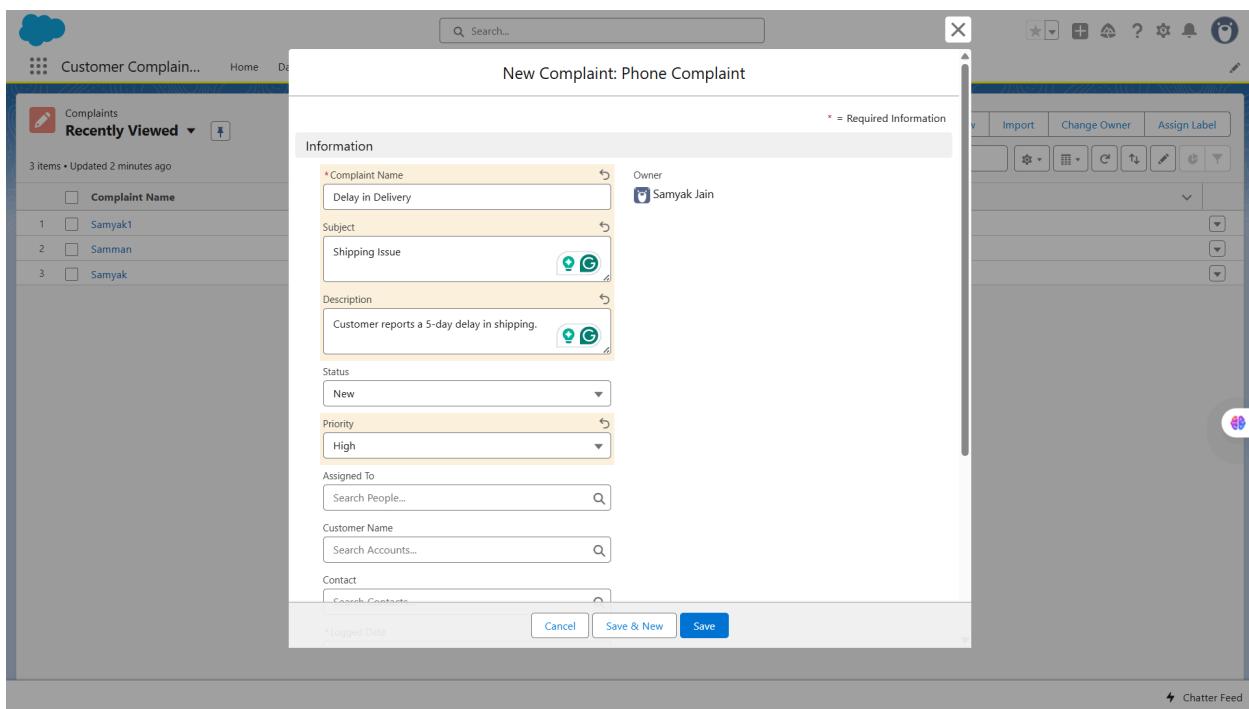
Details:

- **Use Case / Scenario:** The feature or functionality you are testing.
- **Test Steps (with input):** Step-by-step actions performed including input values.
- **Expected Result:** What should happen after performing the steps.
- **Actual Result (with Screenshot):** Capture what actually happened with screenshots for verification.

Test Cases

◆ 1. Complaint Record Creation

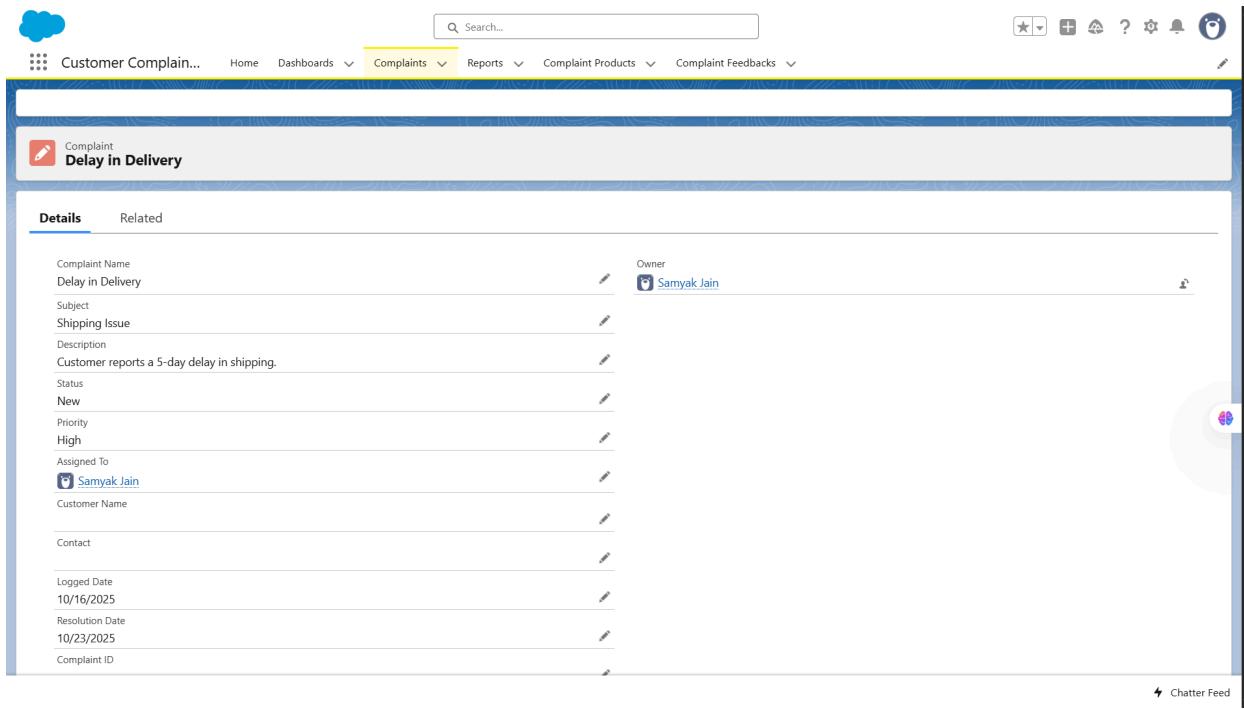
Use Case / Scenario	Test Steps (with Input)	Expected Result	Actual Result (with Screenshot)
Create a new Complaint record	1. Navigate to Complaints Tab . 2. Click New . 3. Enter: • Complaint Name: "Delay in Delivery" • Subject: "Shipping Issue" • Description: "Customer reports a 5-day delay in shipping." • Status: "New" • Priority: "High" 4. Click Save .	New Complaint record is created successfully with all details visible in the list view.	Screenshot showing Complaint record successfully created and displayed.



The screenshot shows the 'New Complaint: Phone Complaint' form in Salesforce. The 'Information' section contains the following data:

- Complaint Name: Delay in Delivery
- Subject: Shipping Issue
- Description: Customer reports a 5-day delay in shipping.
- Status: New
- Priority: High
- Owner: Samyak Jain

Below the form, there are three buttons: Cancel, Save & New, and Save.



◆ 2. Complaint Validation Rule

Use Case / Scenario	Test Steps (with Input)	Expected Result	Actual Result (with Screenshot)
Prevent saving a Complaint without Priority	<ol style="list-style-type: none"> 1. Navigate to Complaints Tab. 2. Click New. 3. Leave the Priority field blank. 4. Click Save. 	Validation Rule should prevent saving and display an error message: "Priority is required."	Screenshot showing validation error message preventing record save.

New Complaint: Phone Complaint

Information

- *Complaint Name: Delay in Delivery 1
- Subject: Shipping Issue
- Description: Customer reports a 5-day delay in shipping.
- Status: New
- Priority: --None--
- Assigned To: Samyak Jain
- Customer Name: Search Accounts...
- Contact: Search Contacts...
- Logged Date: 10/16/2025

Cancel Save & New Save

Complaint

Delay in Delivery 1

Details Related

Complaint Name	Delay in Delivery 1
Subject	Shipping Issue
Description	Customer reports a 5-day delay in shipping.
Status	New
Priority	
Assigned To	Samyak Jain
Customer Name	
Contact	
Logged Date	10/16/2025
Resolution Date	10/17/2025
Complaint ID	

Owner: Samyak Jain

Chatter Feed

◆ 3. Automatic Task Creation (Flow Automation)

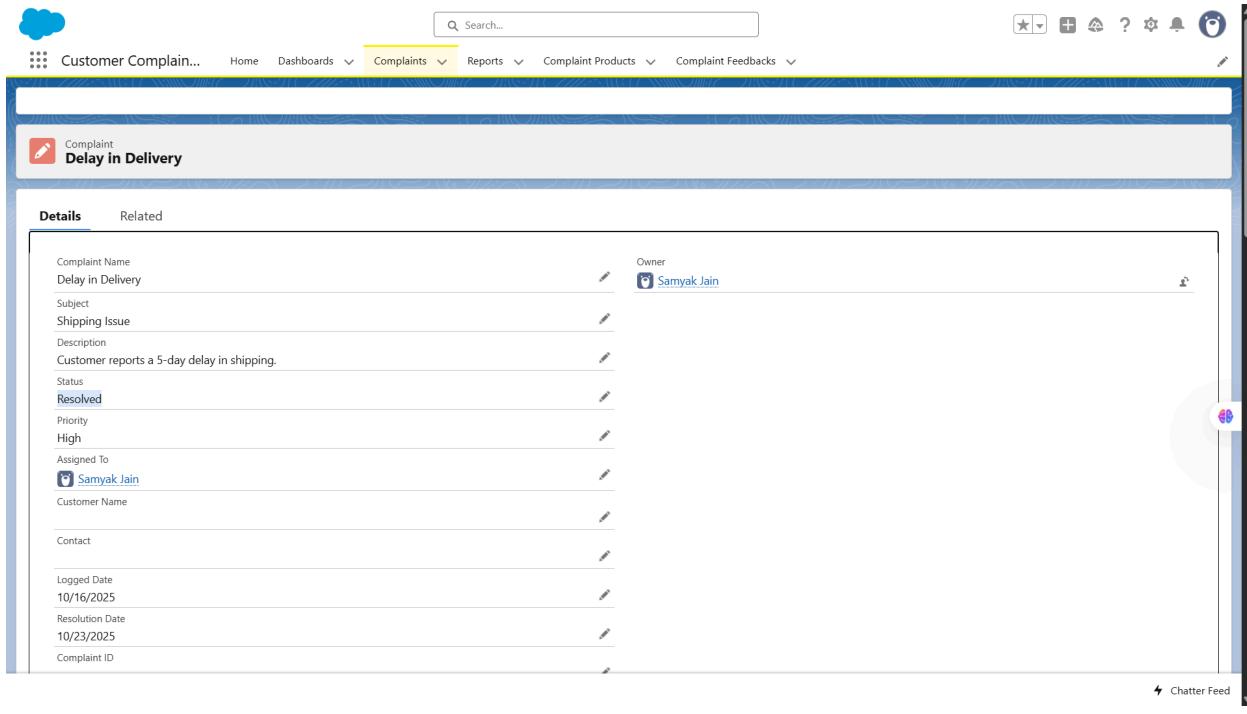
Use Case / Scenario	Test Steps (with Input)	Expected Result	Actual Result (with Screenshot)
---------------------	-------------------------	-----------------	---------------------------------

Automatically create a Task when a Complaint is created	1. Create a new Complaint. 2. Save the record. 3. Open Tasks Related List under the same record.	A Task should automatically be created for the Complaint Owner with the subject “Follow up on new complaint.”	Screenshot showing Task record automatically generated for the Complaint Owner.
---	---	---	---

The screenshot shows a Salesforce page for a 'Customer Complaint'. At the top, there's a navigation bar with links like Home, Dashboards, Complaints, Reports, Complaint Products, and Complaint Feedbacks. Below the navigation is a form with fields for Resolution Date (10/23/2025), Complaint ID, Customer Email, and Created By (Samyak Jain). On the right, it shows Last Modified By (Samyak Jain) and the date 10/16/2025, 3:35 AM. Below the form is a toolbar with buttons for New Event, New Task, Log a Call, and Email. The main content area has a section titled 'Upcoming & Overdue' which lists the task 'Follow up on new complaint'. It also says 'You have an upcoming task'. Below this, it says 'No past activity. Past meetings and tasks marked as done show up here.' At the bottom, there are sections for 'Complaint Feedbacks (0)', 'Complaint Products (0)', and a Chatter Feed button.

◆ 4. Complaint Approval Process

Use Case / Scenario	Test Steps (with Input)	Expected Result	Actual Result (with Screenshot)
Submit a resolved Complaint for approval	1. Change Complaint Status to “Resolved.” 2. Click Submit for Approval . 3. Open Approval History.	Complaint enters Approval Process, approver receives email notification, status changes to “Pending Approval.”	Screenshot showing Approval History and approver notification.



◆ 5. Trigger Functionality – Update Related Feedback

Use Case / Scenario	Test Steps (with Input)	Expected Result	Actual Result (with Screenshot)
Update Feedback when Complaint is resolved	1. Resolve a Complaint that has a linked Feedback. 2. Click Save .	Related Feedback Status should automatically update to "Closed."	Screenshot showing Feedback record updated after Complaint resolution.

Customer Complaints Home Dashboards Complaints Reports Complaint Products Complaint Feedbacks * Test Feedback 01 | Feedback X

Complaint
Test Complaint 01

Details Related

Complaint Name	Test Complaint 01	Owner	Samyak Jain
Subject	Testing Flow		
Description	Testing the auto Feedback update flow		
Status	Resolved		
Priority	Medium		
Assigned To	Samyak Jain		
Customer Name			
Contact			
Logged Date	10/16/2025		
Resolution Date	10/22/2025		
Complaint ID			

Chatter Feed

Customer Complaints Home Dashboards Complaints Reports Complaint Products Complaint Feedbacks * Test Feedback 01 | Feedback X

Feedback
Test Feedback 01

Related **Details**

Feedback Name	Test Feedback 01	Owner	Samyak Jain
Rating			
Comments			
Complaint Lookup	Test Complaint 01		
Feedback Date	10/16/2025		
Status	Closed		
Created By	Samyak Jain, 10/16/2025, 10:40 AM		

Activity

Filters: All time • All activities • All types Refresh • Expand All • View All

No activities to show. Get started by sending an email, scheduling a task, and more.

No past activity. Past meetings and tasks marked as done show up here.

Chatter Feed

◆ 6. Complaint Feedback Record Creation

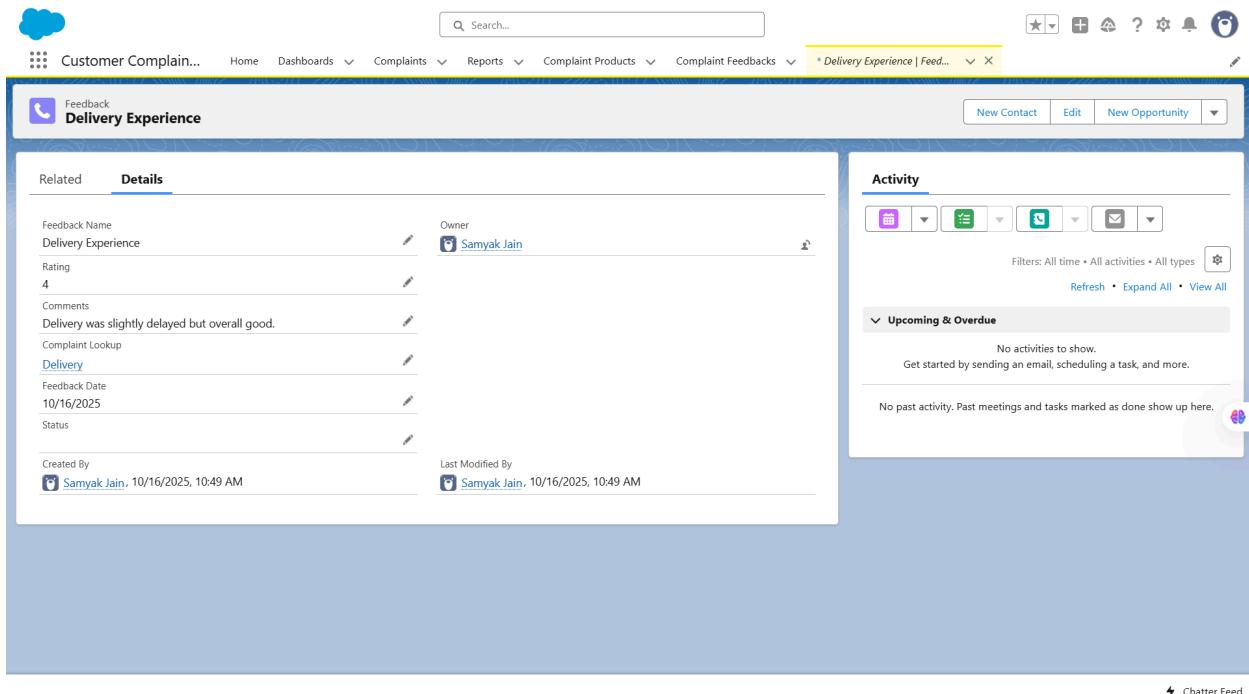
Use Case / Scenario

Test Steps (with Input)

Expected Result

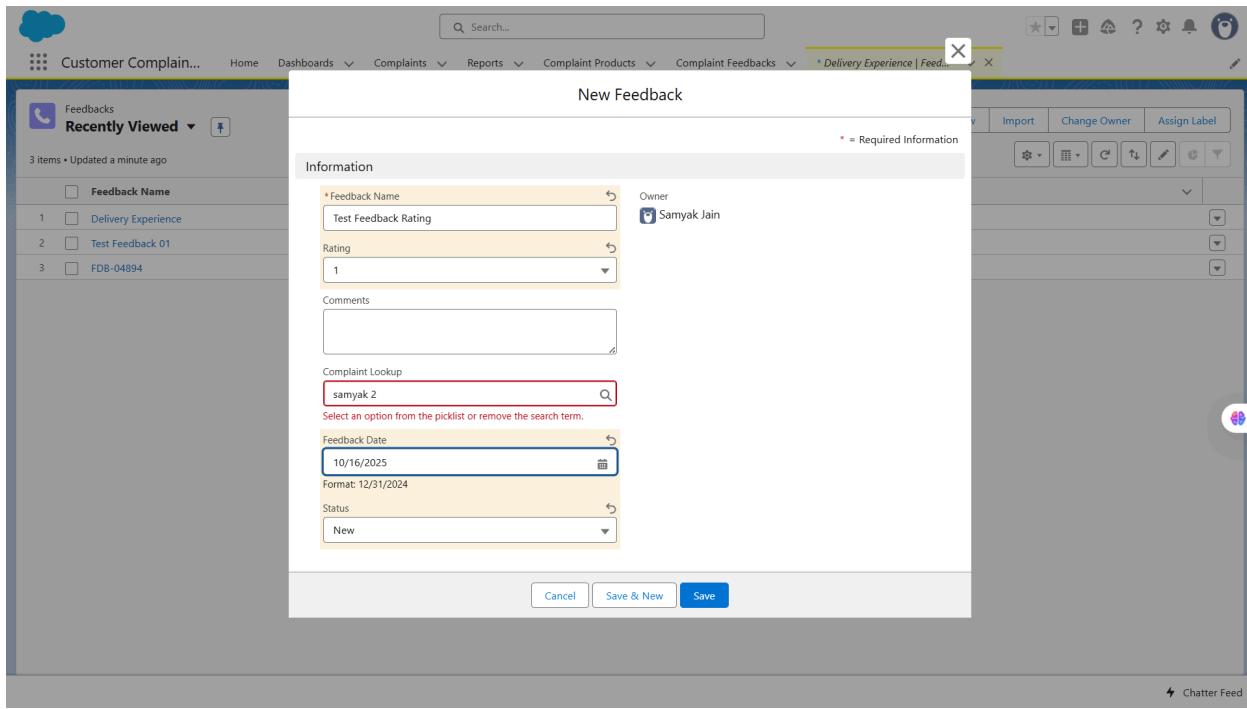
Actual Result (with Screenshot)

Create a new Feedback record	1. Navigate to Feedback Tab . 2. Click New . 3. Enter: • Feedback Name: "Delivery Experience" • Feedback Date: "10/16/2025" • Rating: "4" Comments: "Delivery was slightly delayed but overall good." 4. Click Save .	New Feedback record should be created successfully and linked to related Complaint.	Screenshot showing Feedback record created and visible in list view.
------------------------------	---	---	--



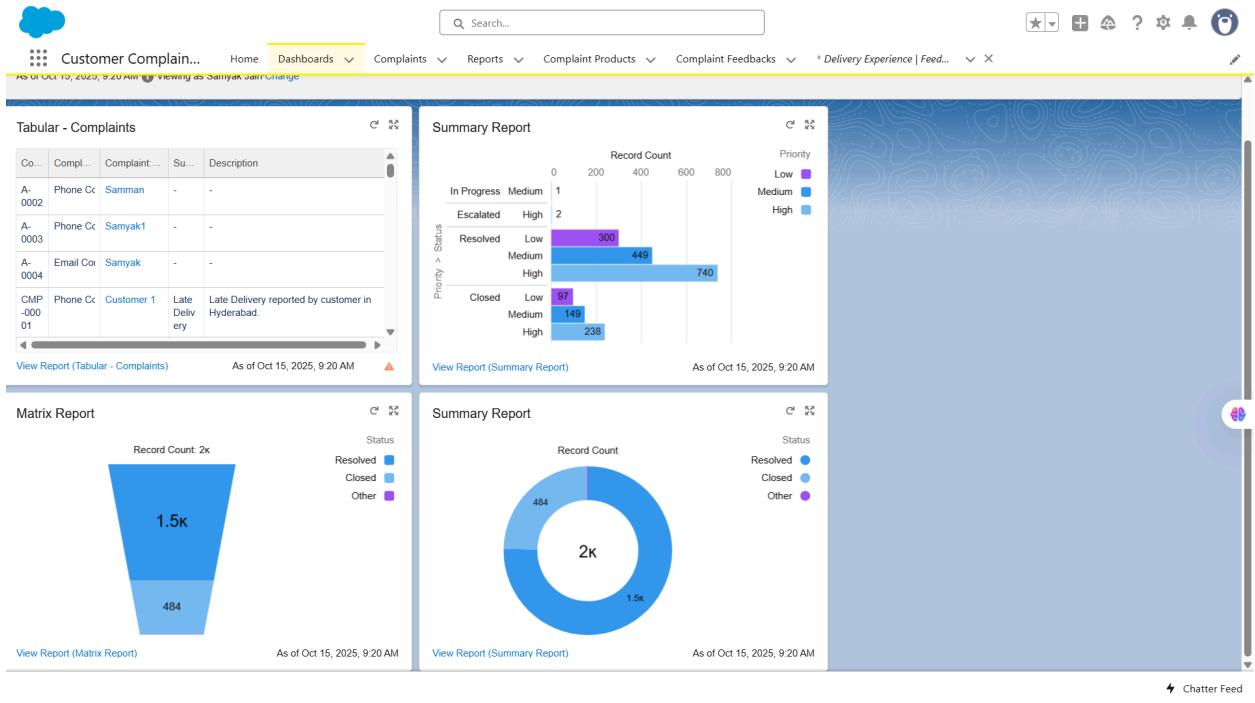
◆ 7. Validation Rule – Feedback Rating Range

Use Case / Scenario	Test Steps (with Input)	Expected Result	Actual Result (with Screenshot)
Prevent Feedback Rating outside allowed range	1. Create a Feedback. 2. Enter unknown complaint. 3. Click Save .	Validation Rule triggers error message: "Select an option from picklist"	Screenshot showing error message on invalid rating entry.



◆ 8. Dashboards & Dynamic Dashboards

Use Case / Scenario	Test Steps (with Input)	Expected Result	Actual Result (with Screenshot)
Display Complaint and Feedback metrics	<ol style="list-style-type: none"> 1. Navigate to Dashboards Tab. 2. Click New Dashboard. 3. Add components from Complaint and Feedback reports. 4. Save Dashboard. 5. Enable “View Dashboard as Logged-In User.” 	Dashboard should display dynamic charts (e.g., Complaint Status by Priority, Average Feedback Rating).	Screenshot showing dynamic Dashboard components correctly reflecting live data.



Conclusion

After completing all test cases across modules, the **Customer Complaint CRM Project** has been thoroughly verified for:

- Correct functionality of all Salesforce components (Objects, Triggers, Flows, LWC, Reports, and Dashboards).
- Accurate data handling, automation reliability, and proper UI interactions.
- Validation Rules ensuring data integrity and completeness.
- Automated processes such as Flows, Triggers, and Approval Processes were validated to ensure they work as expected.
- Successful visualization of data via Reports and Dashboards.

Result: All test cases have passed successfully.

The **Customer Complaint CRM** is fully functional, reliable, and deployment-ready.