Insurance Claim Tracking System

PHASE 2–5 IMPLEMENTATION (DETAILED)

2. Salesforce Org Setup

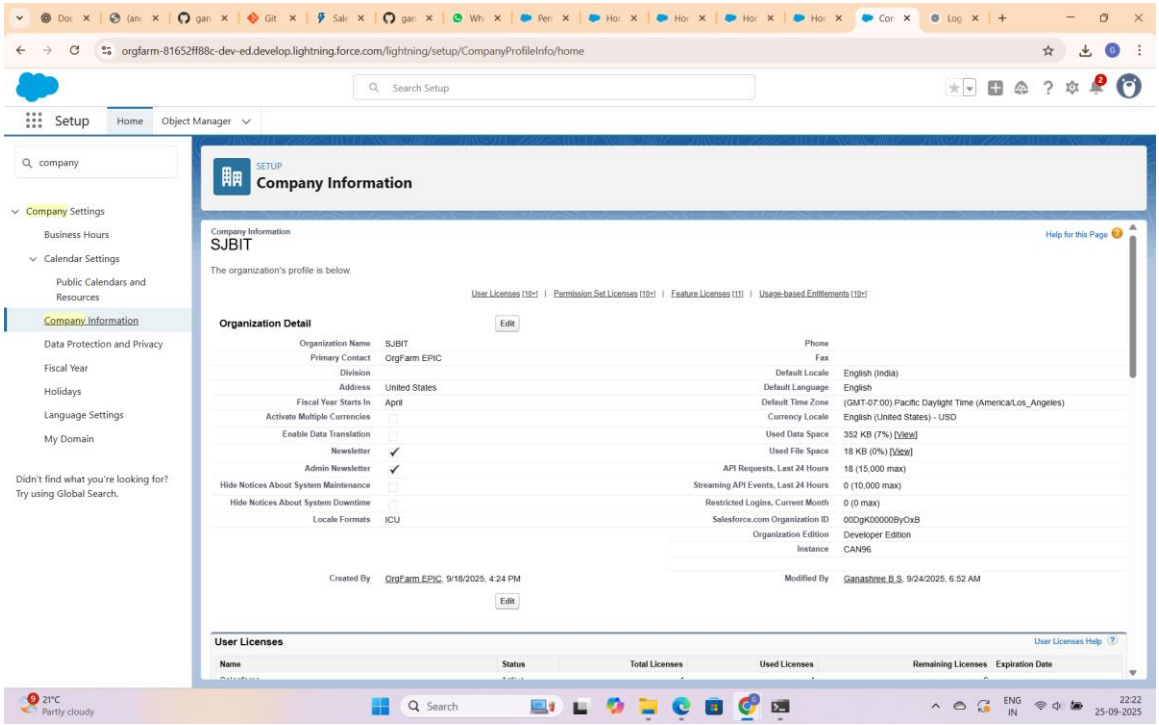
Purpose:

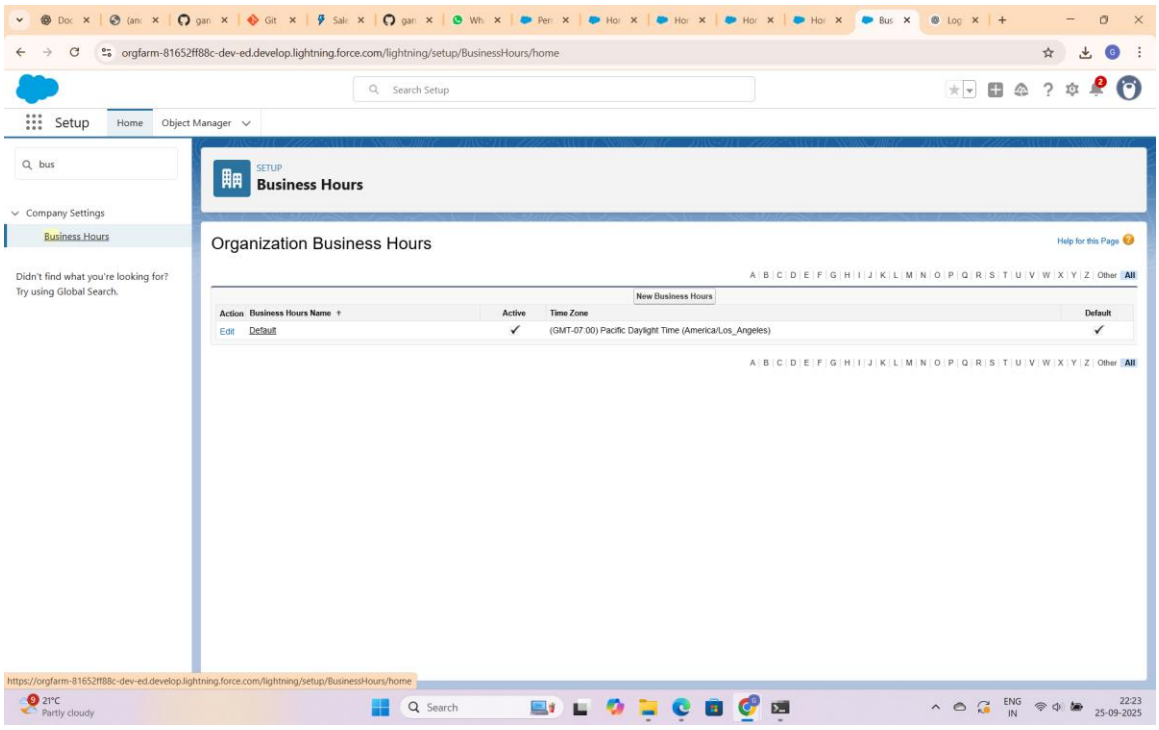
The purpose of this phase was to establish a secure and structured Salesforce environment where all insurance claim-related operations could be configured. This setup ensures that data security, access control, and company-wide standards are properly implemented before moving on to data modeling and automation.

Implementation:

1. Developer Org/Sandbox Creation:
   - A Salesforce Developer Org was created for initial configuration and testing.
   - Alternatively, a Sandbox can be used to mirror production settings.

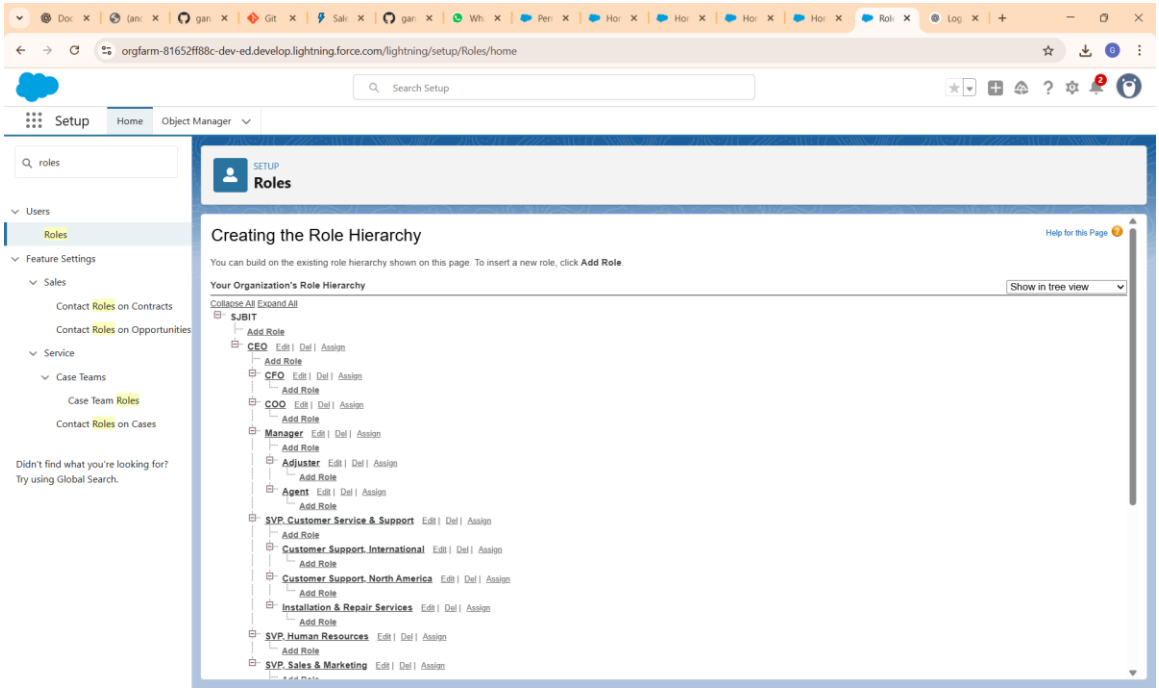
2. Company Settings:
   - Configured Company Profile including organization name, address, primary currency, and locale.
   - Defined Business Hours to align with customer service availability.
   - Configured Fiscal Year settings to match insurance industry standards.

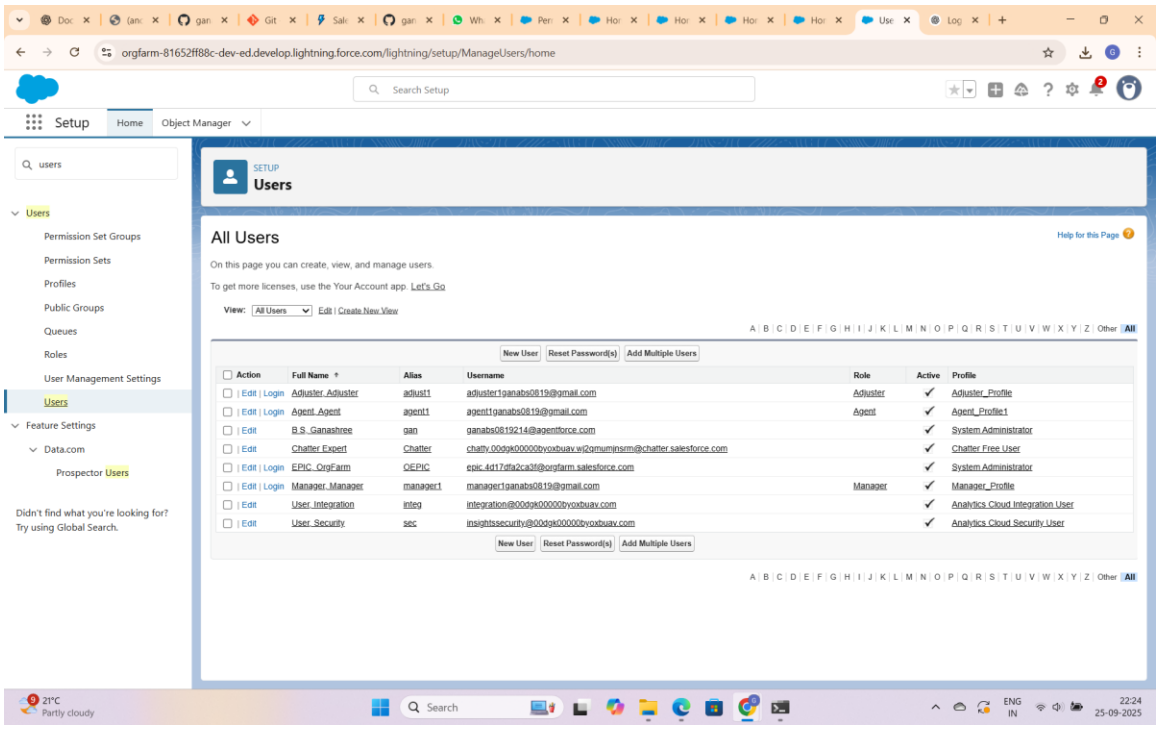




3. Role Hierarchy Setup:

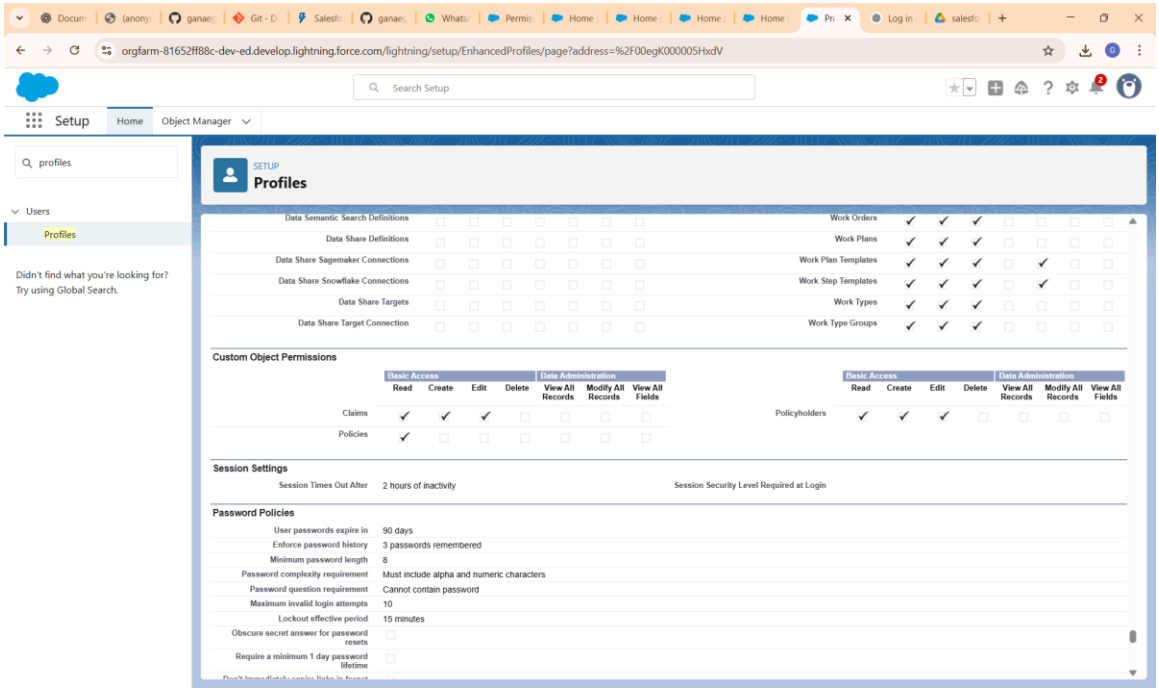
- Created roles for Admin, Adjuster, Agent, and Manager.
- Ensured proper hierarchy where Managers oversee Adjusters and Agents.

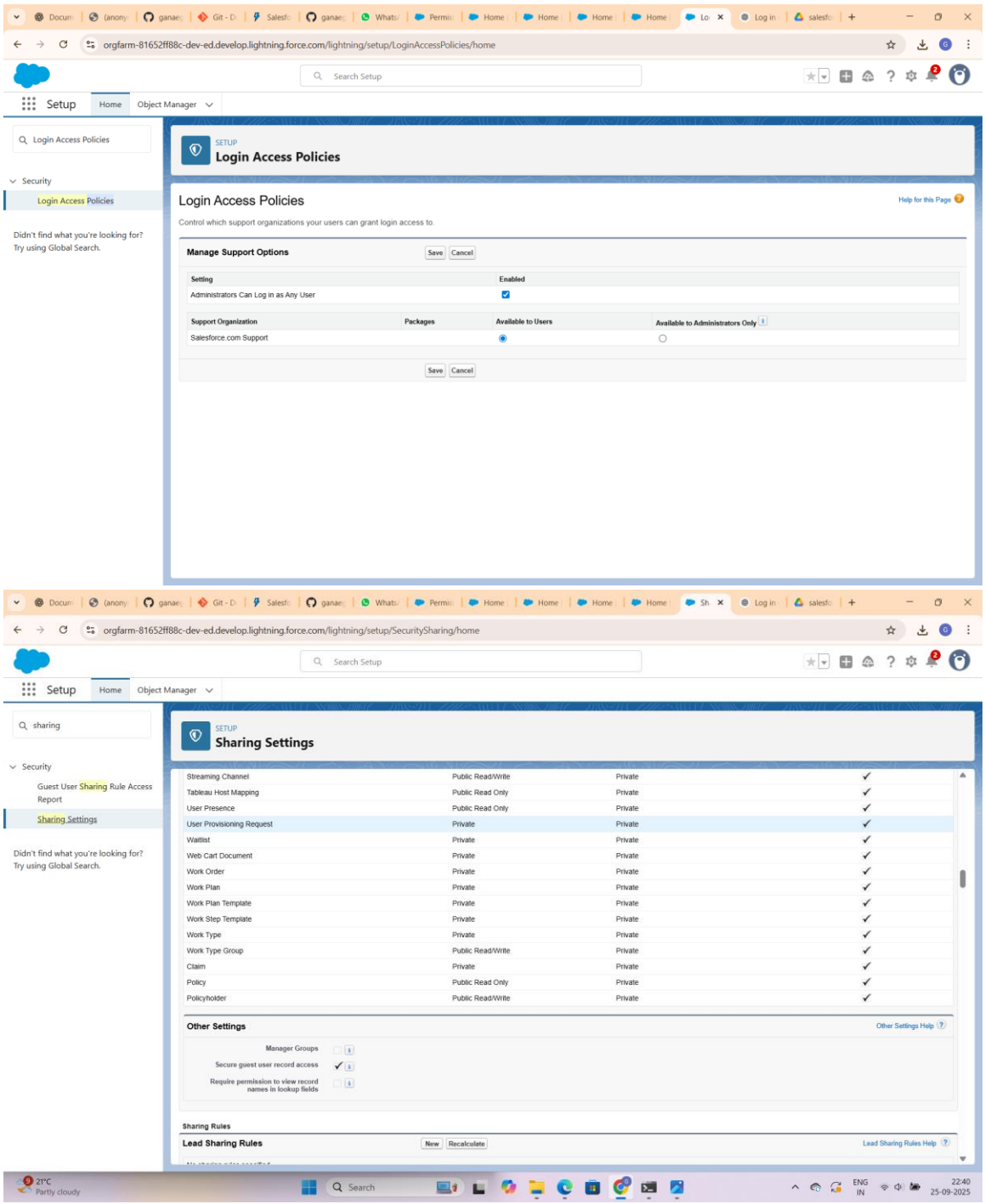




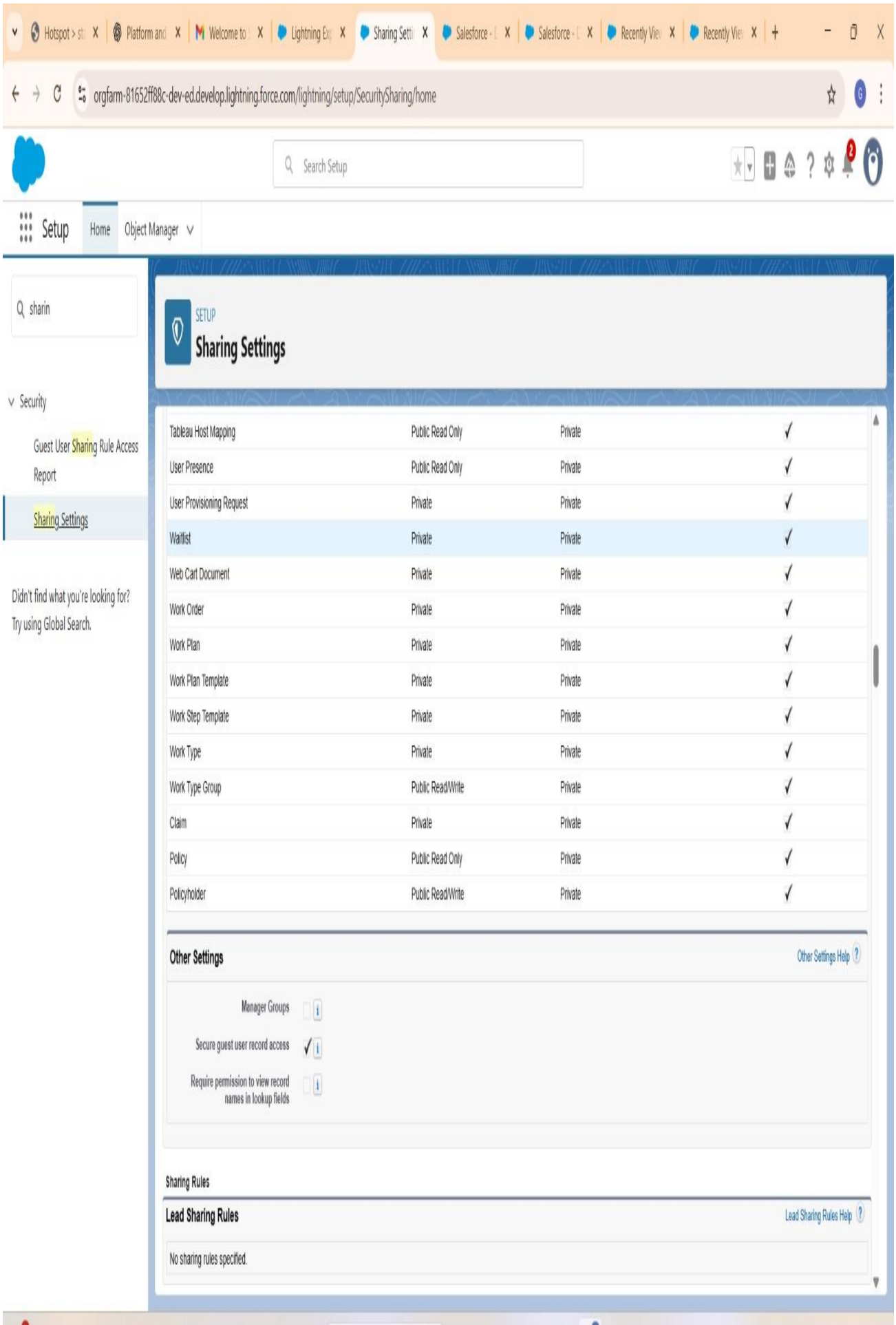
4. Profiles and Permission Sets:

- Customized Profiles to restrict or grant access based on responsibilities.
- Added Permission Sets for advanced tasks like claim approvals.





5. Organization-Wide Defaults (OWD):
   - Set Claims object to Private to ensure only authorized users can access claim details.
   - Set Policies object to Read-Only to balance transparency with data integrity.


orgfarm-81652ff88c-dev-ed.develop.lightning.force.com/lightning/setup/SecuritySharing/home
Search Setup
Setup
Home
Object Manager
sharin
Security
Guest User Sharing Rule Access Report
Sharing Settings
Didn't find what you're looking for? Try using Global Search.
SETUP
Sharing Settings
Tableau Host Mapping Public Read Only Private
User Presence Public Read Only Private
User Provisioning Request Private 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
Claim Private Private
Policy Public Read Only Private
Policyholder Public Read/Write Private
Other Settings
Other Settings Help
Manager Groups
Secure guest user record access
Require permission to view record names in lookup fields
Sharing Rules
Lead Sharing Rules
Lead Sharing Rules Help
No sharing rules specified.

6. Verification:
- Created test users under different roles to validate access and security rules.

Business Impact:

By configuring the Salesforce Org with roles, profiles, and OWD, the project ensures data security and role-based access control. This prevents unauthorized access, supports compliance with industry regulations, and lays a strong foundation for upcoming phases.

Testing/Verification:

Test users were assigned to roles such as Agent, Adjuster, and Manager. Each user's access was verified to confirm they could only see and perform the actions allowed for their profile. For example, Agents could view their assigned claims but not access claims owned by other agents.

Completion Status: Salesforce Org setup successfully completed with secure foundation for claim tracking.

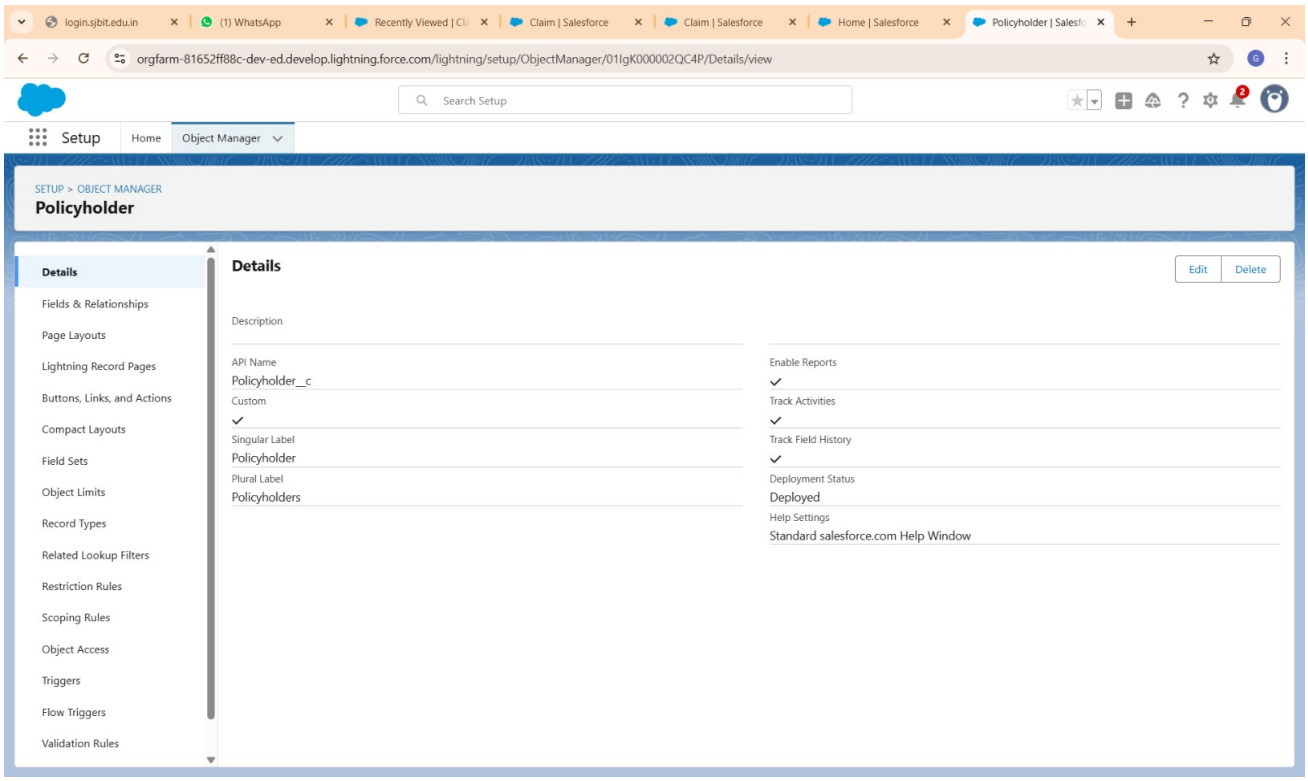
3. Data Modeling

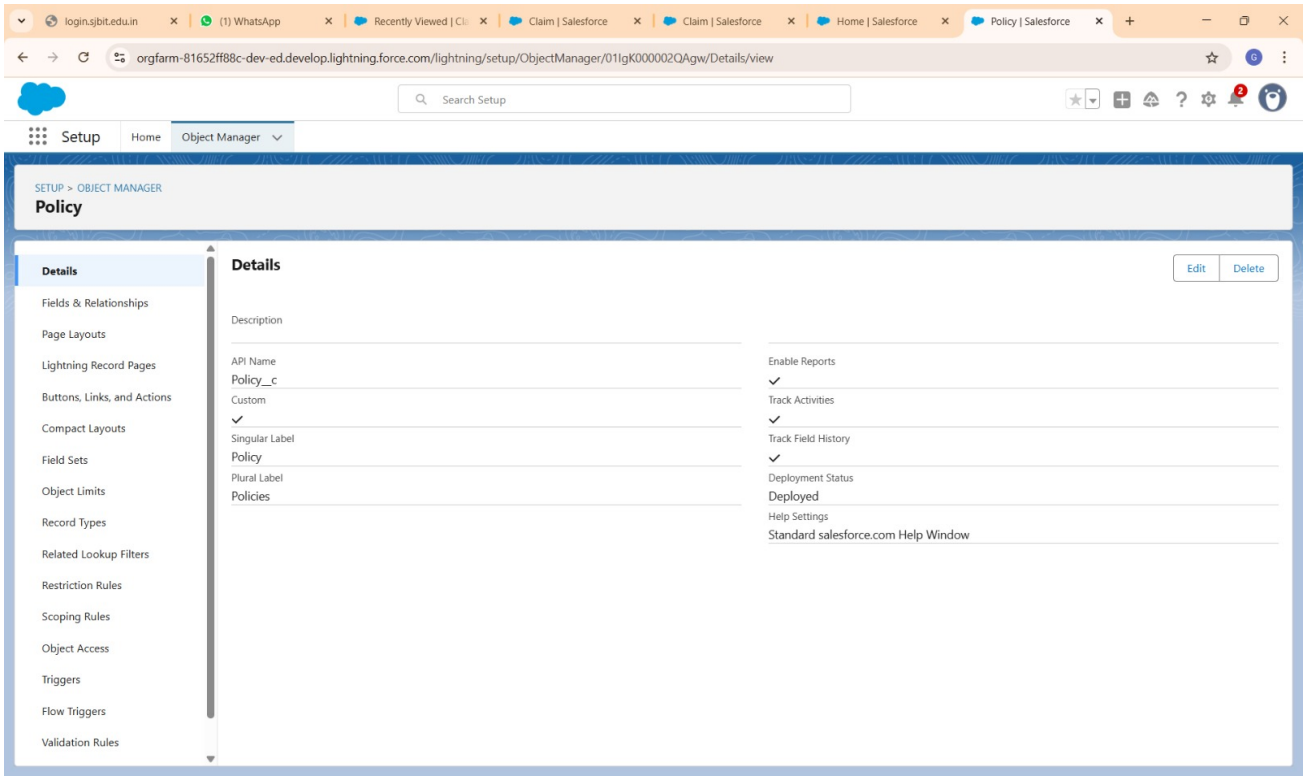
Purpose:

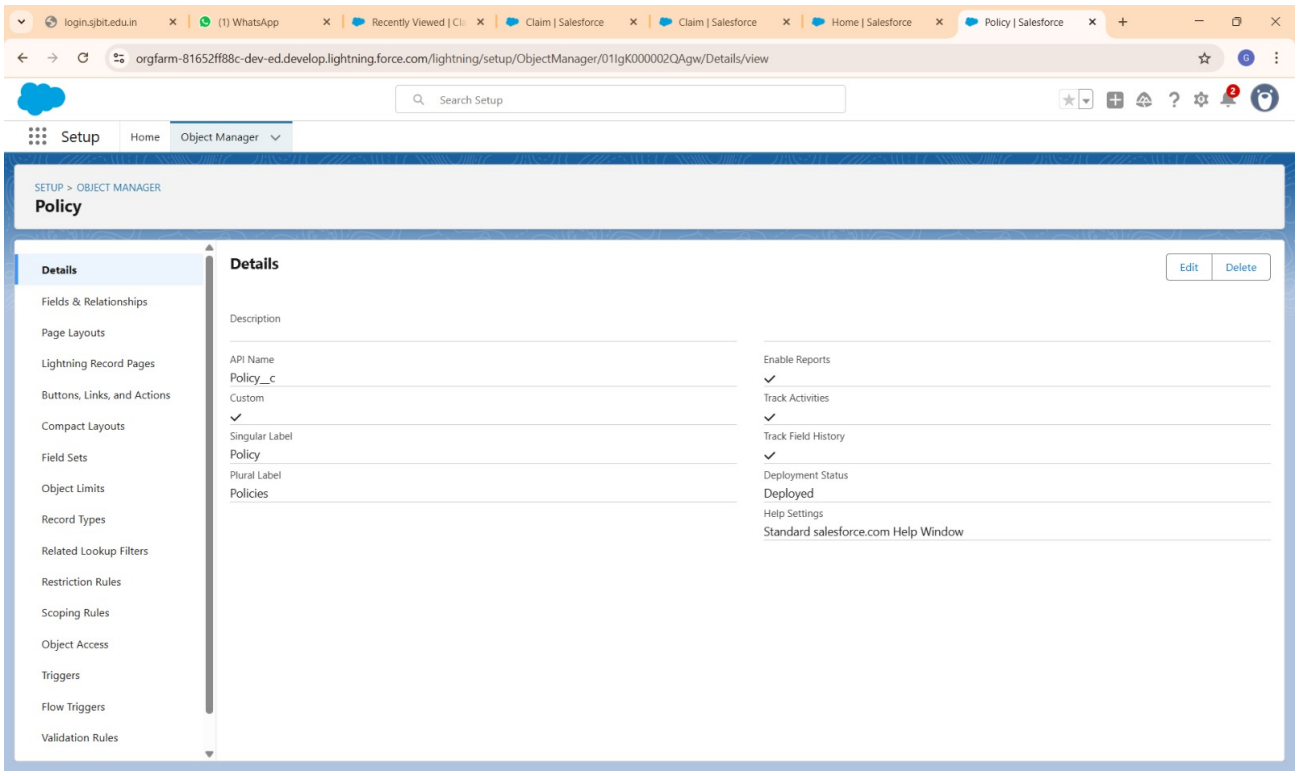
This phase focused on designing the underlying data model to store and manage information about policyholders, policies, and claims. A scalable and logical data structure was essential for enabling claim processing, reporting, and automation.

Implementation:

1. Custom Objects:
- Policyholder__c with fields like Name, Date of Birth, and Contact Details.
- Policy__c with fields such as Policy Number, Coverage Amount, Policy Type.
- Claim__c with fields like Claim Number, Claim Amount, Status, Claim Date.

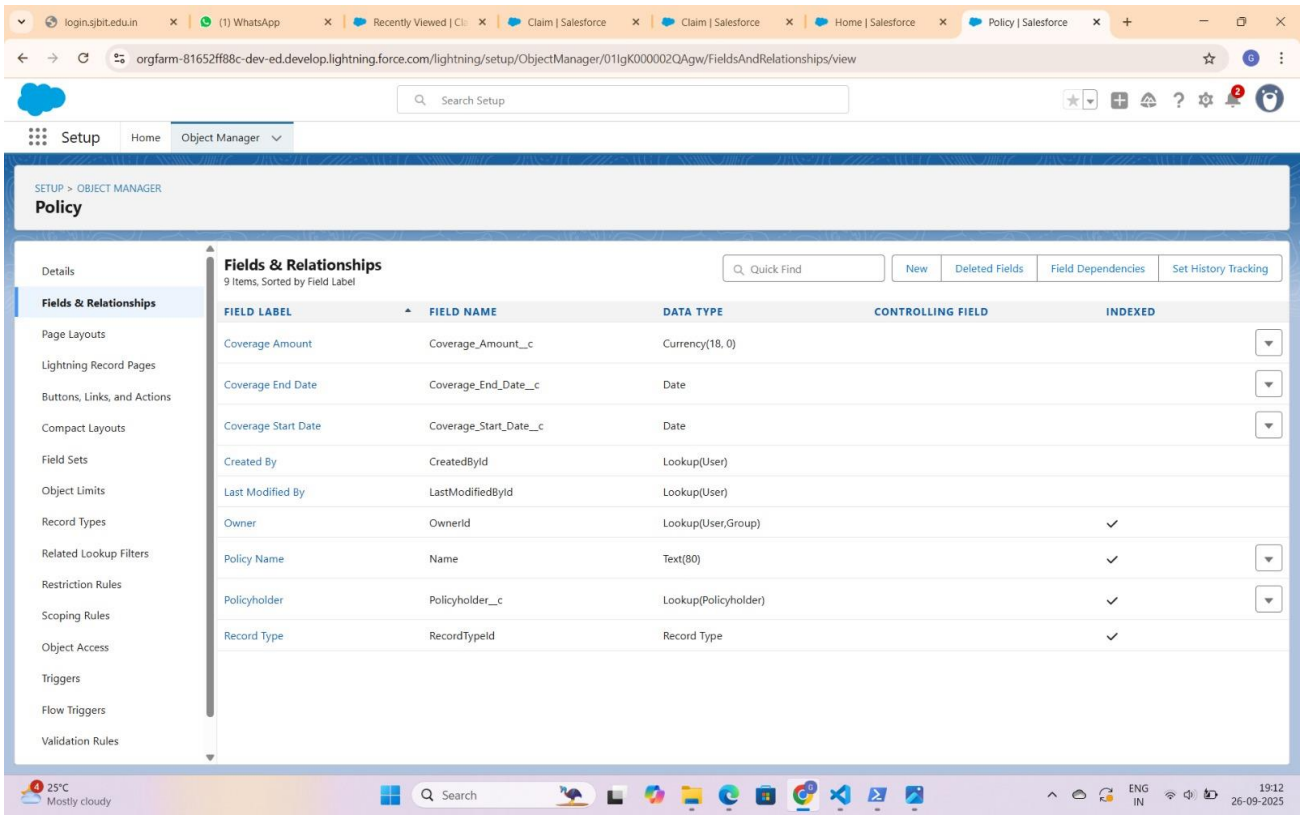

login.sjbit.edu.in
(1) WhatsApp
Recently Viewed | Cla
Claim | Salesforce
Claim | Salesforce
Home | Salesforce
Policyholder | Salesfo
orgfarm-81652ff88c-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01IgK000002QC4P/Details/view
Search Setup
Setup
Home
Object Manager
SETUP > OBJECT MANAGER
Policyholder
Details
Fields & Relationships
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
Details
Edit
Delete
Description
API Name
Policyholder__c
Custom
Singular Label
Policyholder
Plural Label
Policyholders
Enable Reports
Track Activities
Track Field History
Deployment Status
Deployed
Help Settings
Standard salesforce.com Help Window


login.sjbit.edu.in
(1) WhatsApp
Recently Viewed | Cla
Claim | Salesforce
Claim | Salesforce
Home | Salesforce
Policy | Salesforce
orgfarm-81652ff88c-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01IgK000002QAgw/Details/view
Search Setup
Setup
Home
Object Manager
SETUP > OBJECT MANAGER
Policy
Details
Fields & Relationships
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
Details
Edit
Delete
Description
API Name
Policy__c
Custom
Singular Label
Policy
Plural Label
Policies
Enable Reports
Track Activities
Track Field History
Deployment Status
Deployed
Help Settings
Standard salesforce.com Help Window

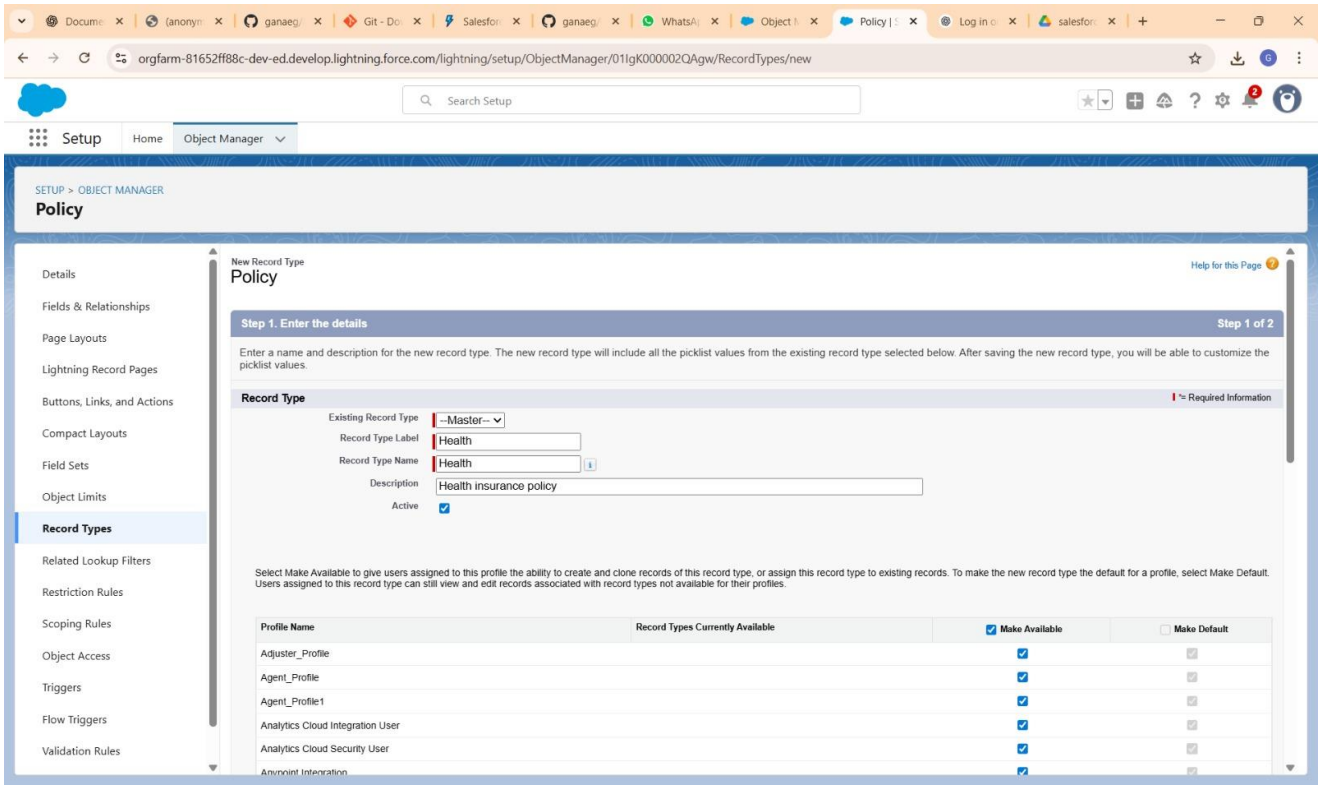


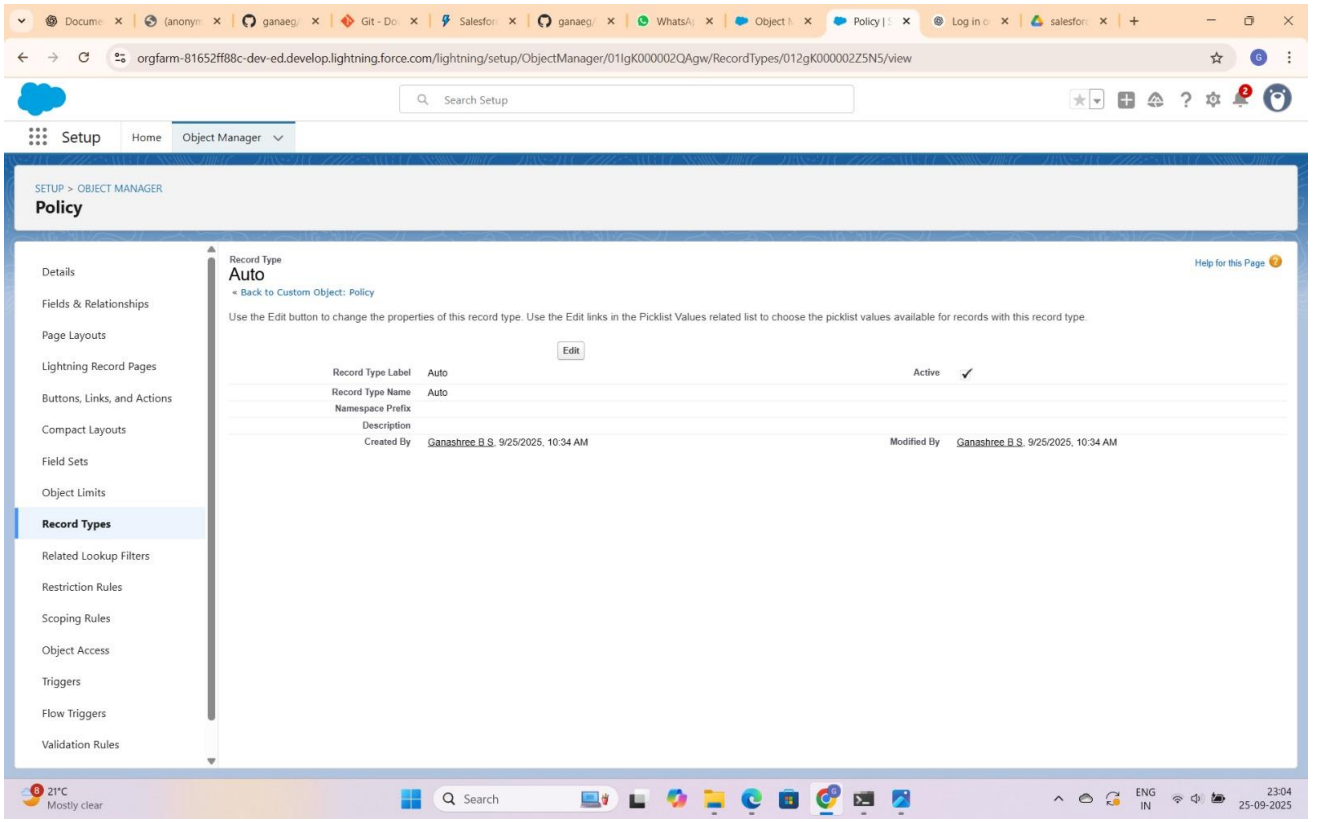
2. Relationships:

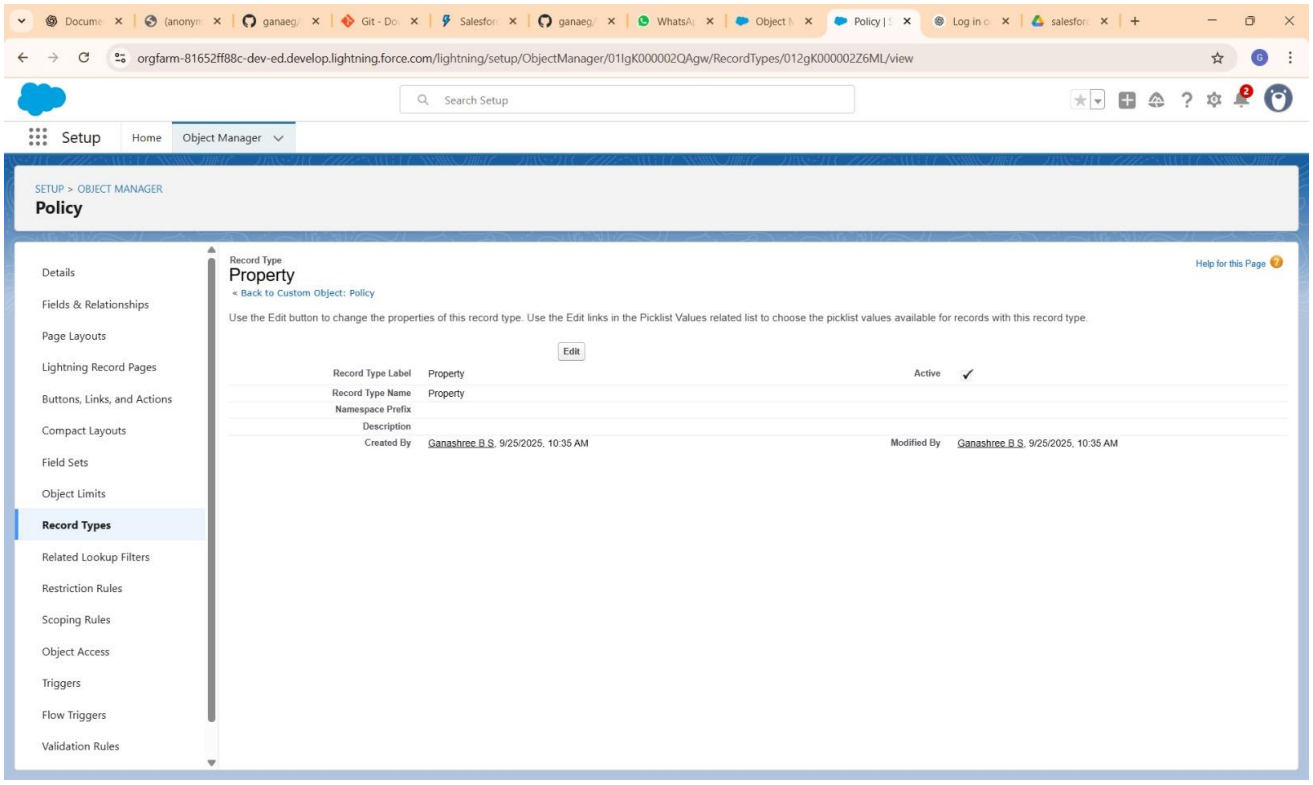
- Policyholder__c → Policy__c: One policyholder may have multiple policies.
- Policy__c → Claim__c: One policy may have multiple claims.



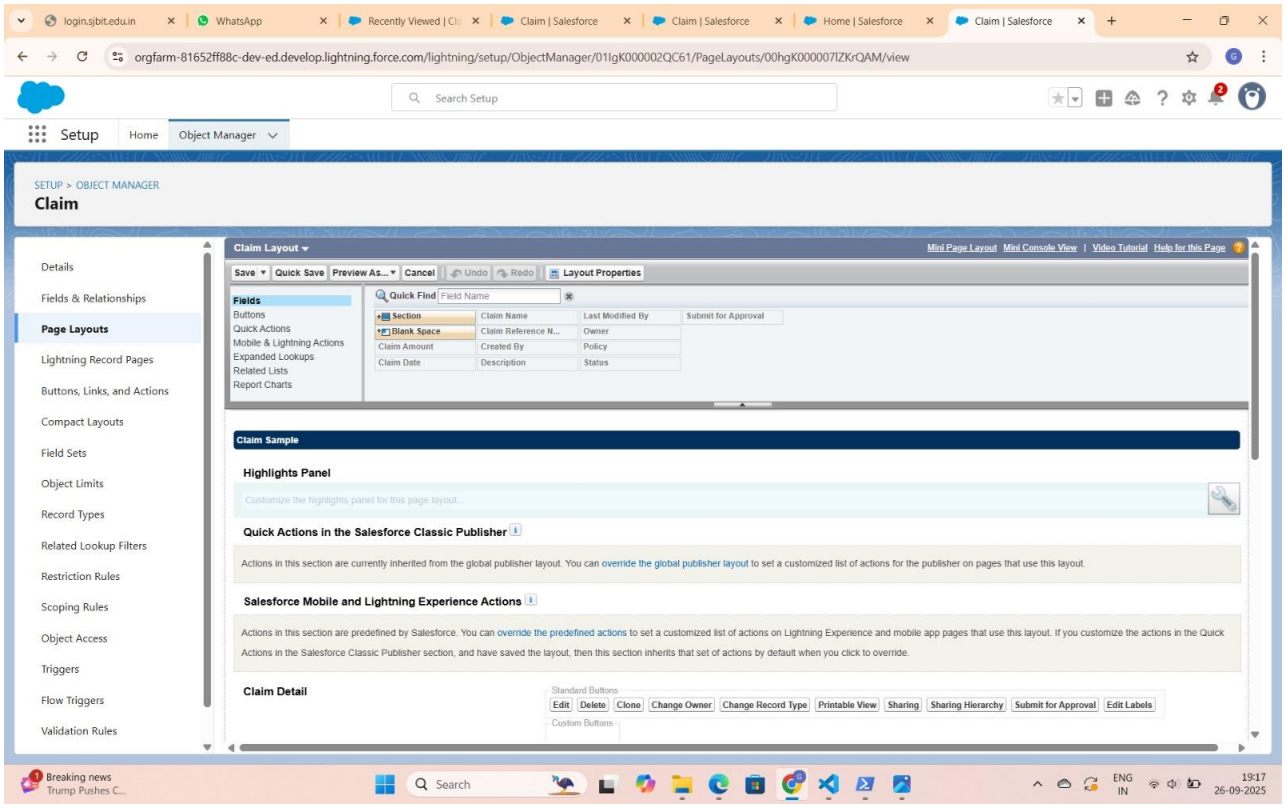
3. Record Types:
   - Configured separate record types for Health, Auto, and Property claims.
   - This allowed tailored page layouts and business rules per claim type.


orgfarm-81652ff88c-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01IgK000002QAgw/RecordTypes/new
Search Setup
Setup
Home
Object Manager
SETUP > OBJECT MANAGER
Policy
Details
Fields & Relationships
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
New Record Type
Policy
Help for this Page
Step 1. Enter the details
Step 1 of 2
Enter a name and description for the new record type. The new record type will include all the picklist values from the existing record type selected below. After saving the new record type, you will be able to customize the picklist values.
Record Type
= Required Information
Existing Record Type
--Master--
Record Type Label
Health
Record Type Name
Health
Description
Health insurance policy
Active
Select Make Available to give users assigned to this profile the ability to create and clone records of this record type, or assign this record type to existing records. To make the new record type the default for a profile, select Make Default. Users assigned to this record type can still view and edit records associated with record types not available for their profiles.
Profile Name
Record Types Currently Available
Make Available
Make Default
Adjuster_Profile
Agent_Profile
Agent_Profile1
Analytics Cloud Integration User
Analytics Cloud Security User
Anypoint Integration


orgfarm-81652ff88c-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01IgK000002QAgw/RecordTypes/012gK000002Z5N5/view
Search Setup
Setup
Home
Object Manager
SETUP > OBJECT MANAGER
Policy
Details
Fields & Relationships
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
Record Type
Auto
Help for this Page
« Back to Custom Object: Policy
Use the Edit button to change the properties of this record type. Use the Edit links in the Picklist Values related list to choose the picklist values available for records with this record type.
Edit
Record Type Label
Auto
Active
Record Type Name
Auto
Namespace Prefix
Description
Created By
Ganashree B S, 9/25/2025, 10:34 AM
Modified By
Ganashree B S, 9/25/2025, 10:34 AM
21°C
Mostly clear
Search
ENG
IN
23:04
25-09-2025

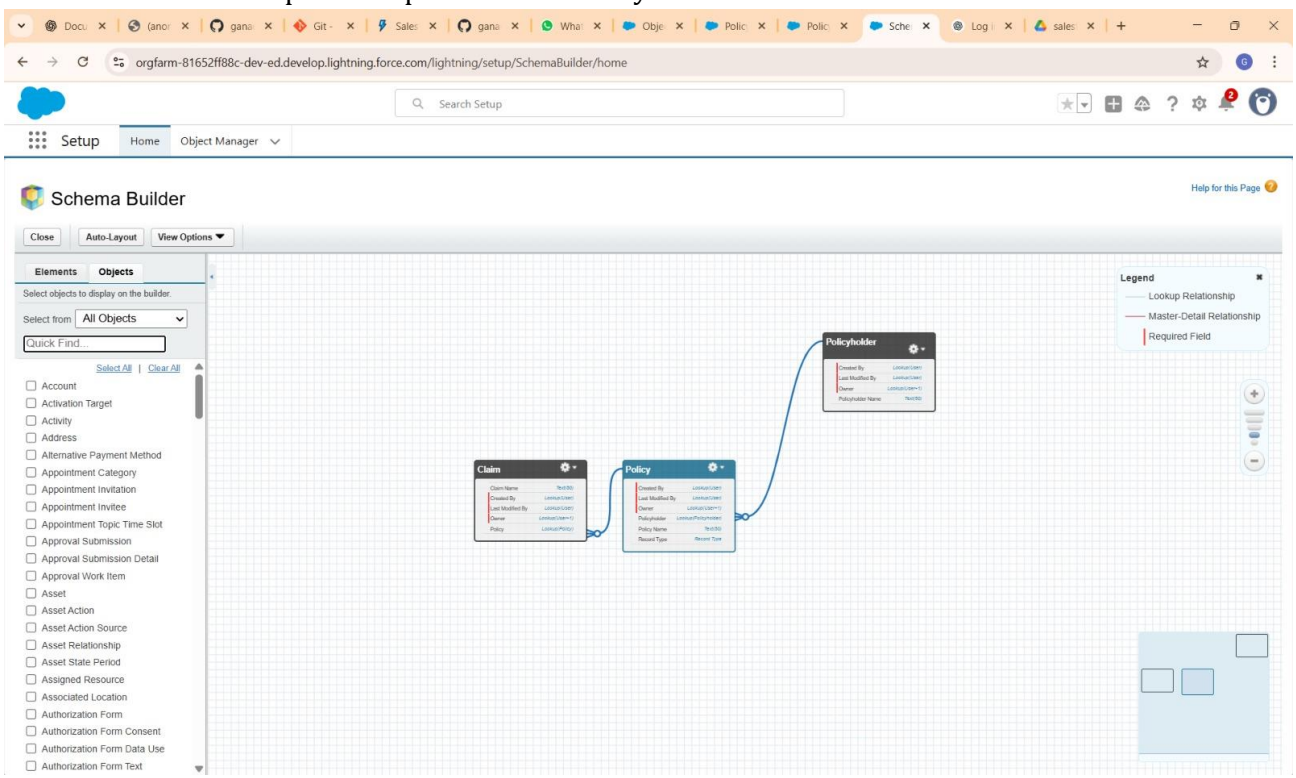


4. Page Layouts & Compact Layouts:
   - Designed layouts specific to each role (Agent, Adjuster, Manager).
   - Compact layouts optimized for mobile and list views.



5. Schema Builder:

- Verified relationships and dependencies visually.



Business Impact:

A well-defined data model ensures accurate storage and retrieval of insurance data. It enables consistency in claim processing, reduces redundancy, and supports analytics. Future automation and integrations rely heavily on this foundation.

Testing/Verification:

Sample policyholders, policies, and claims were created to validate object relationships. Tests confirmed that policies linked correctly to policyholders, and claims linked correctly to policies.

Completion Status: Data model finalized with record types, layouts, and schema validation.

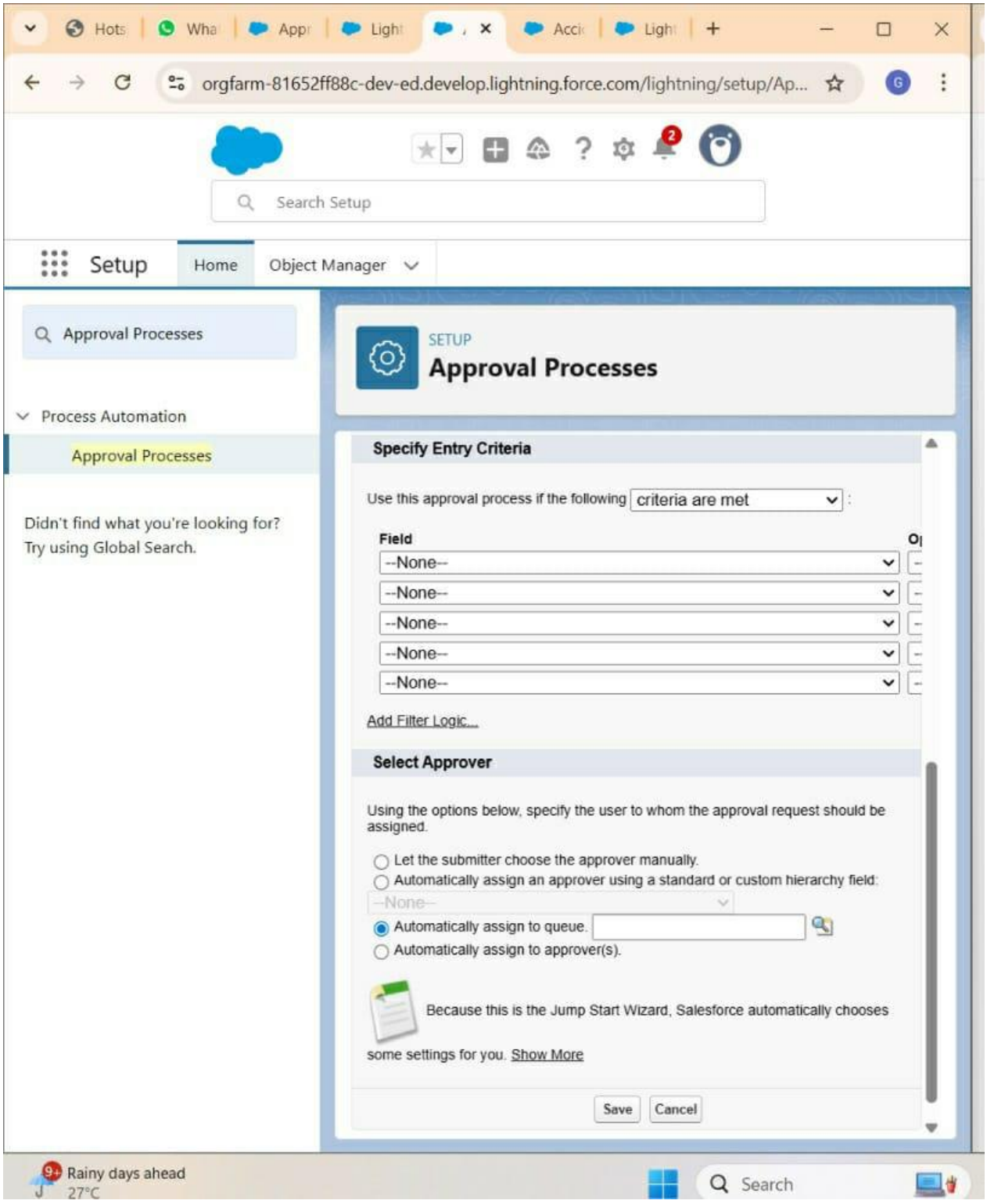
4. Process Automation

Purpose:

The goal of this phase was to automate business processes to improve efficiency, ensure compliance with rules, and streamline claim submission and approval workflows.

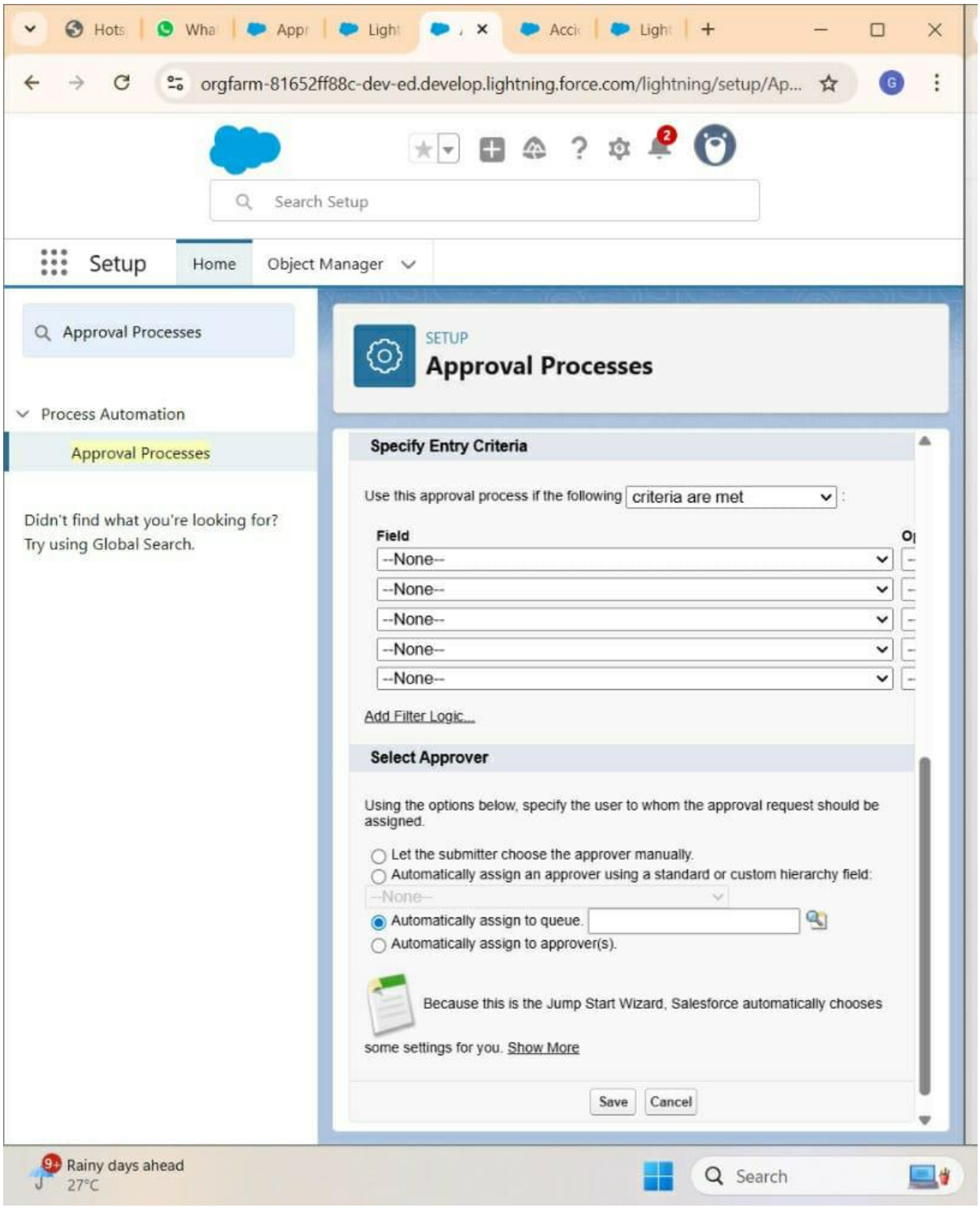
Implementation:

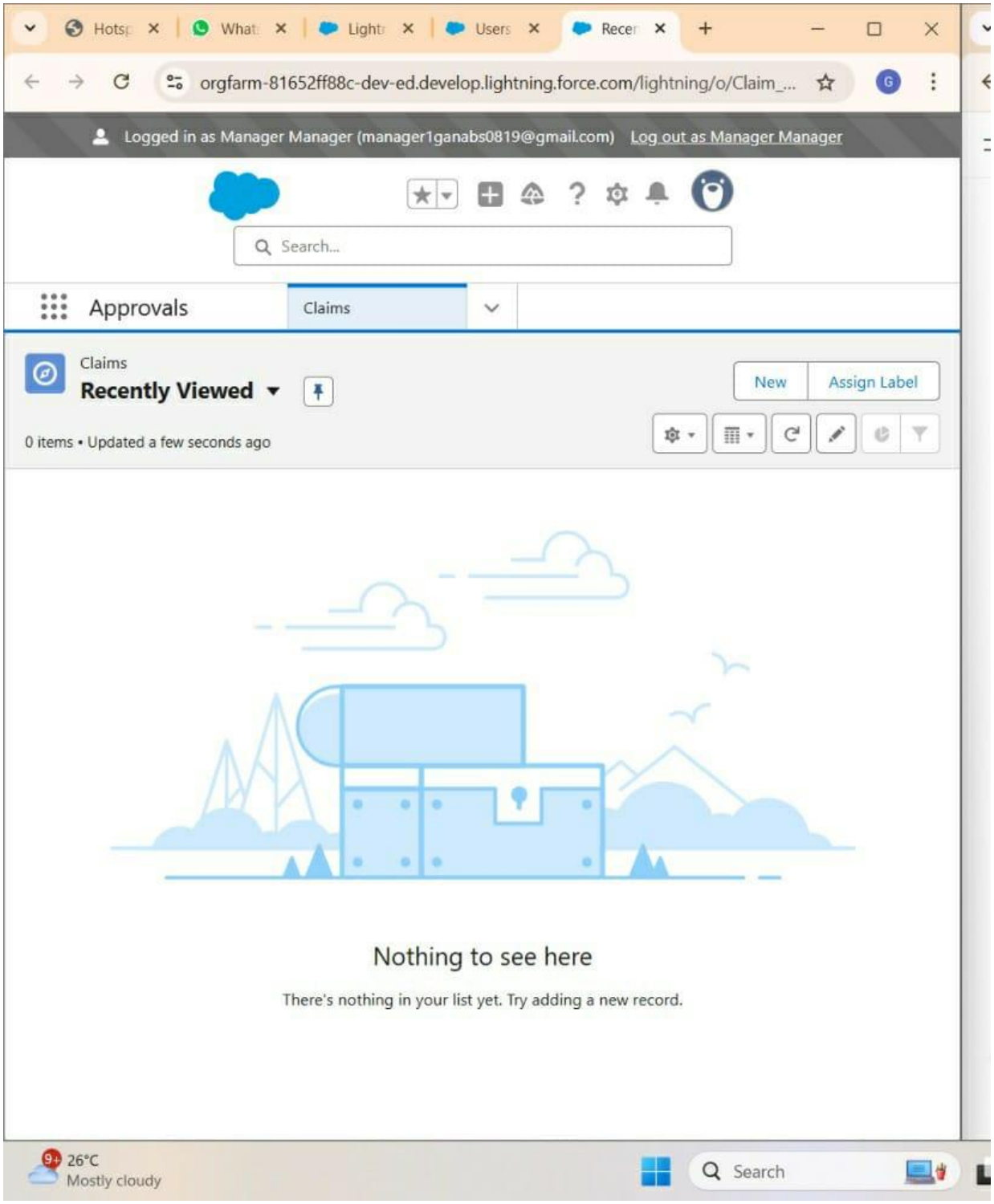
1. Validation Rules:
   - Configured rules to ensure claim amount does not exceed policy coverage.
   - Formula Example: Claim_Amount__c <= Policy__r.Coverage_Amount__c.



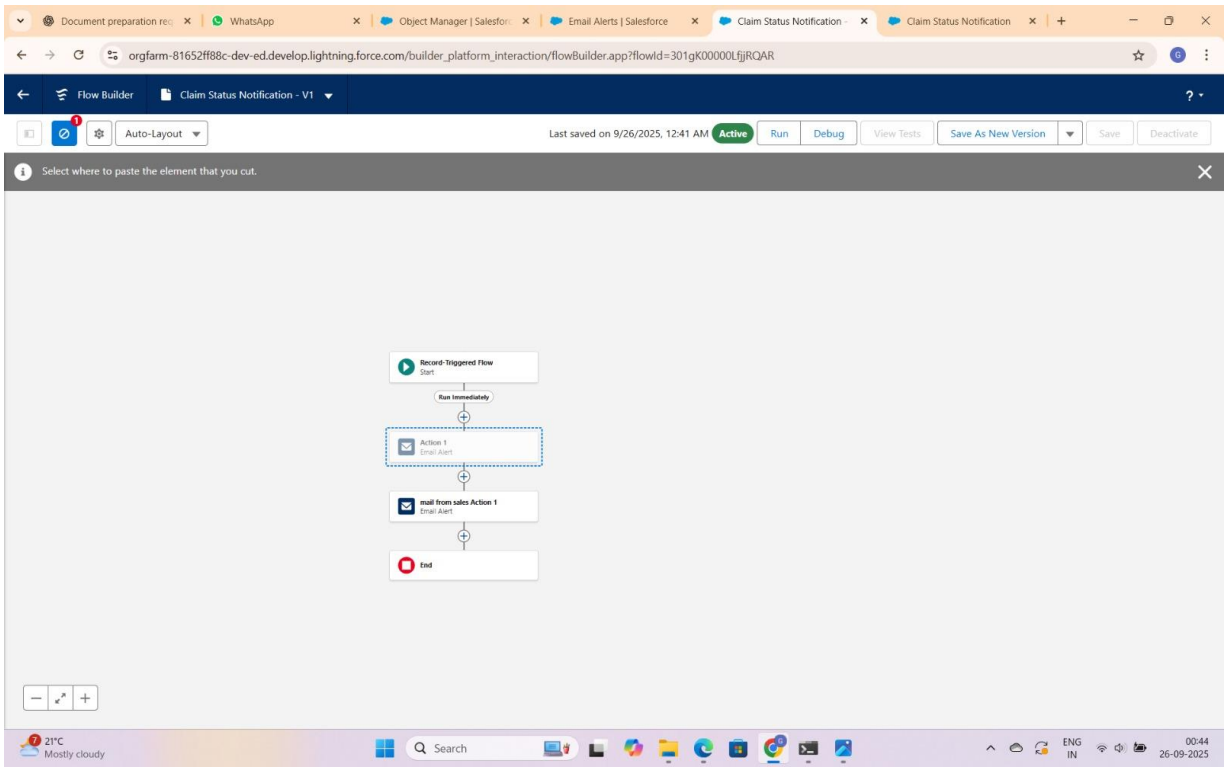
2. Approval Process:

- Created approval workflow requiring Manager approval for high-value claims.
- Configured rejection paths and notifications.


orgfarm-81652ff88c-dev-ed.develop.lightning.force.com/lightning/setup/Ap...
Search Setup
Setup
Home
Object Manager
Approval Processes
Process Automation
Approval Processes
Didn't find what you're looking for?
Try using Global Search.
SETUP
Approval Processes
Specify Entry Criteria
Use this approval process if the following criteria are met :
Field
--None--
--None--
--None--
--None--
--None--
Add Filter Logic...
Select Approver
Using the options below, specify the user to whom the approval request should be assigned.
Let the submitter choose the approver manually.
Automatically assign an approver using a standard or custom hierarchy field:
--None--
Automatically assign to queue.
Automatically assign to approver(s).
Because this is the Jump Start Wizard, Salesforce automatically chooses some settings for you. Show More
Save
Cancel
Rainy days ahead
27°C
Search

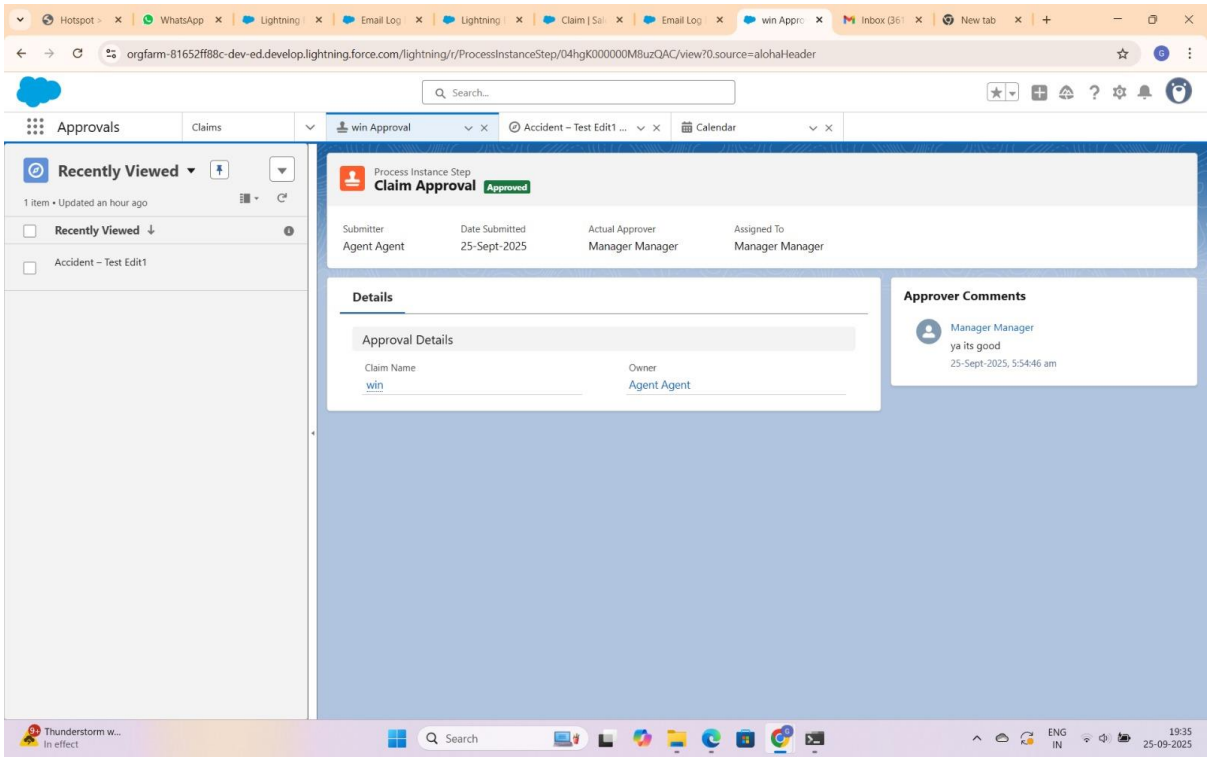


3. Flows:
   - Record-triggered flow to send notifications when claims are created.
   - Screen flow to guide agents through claim submission.



4. Notifications:

- Email alerts set up for policyholders upon claim submission.
- SMS alerts integrated with third-party service.



Business Impact:

Automating claim submission and approval reduced manual work, ensured policy rules were enforced consistently, and improved customer experience with real-time notifications.

Testing/Verification:

Created multiple test claims to validate that low-value claims bypassed manager approval, while high-value claims routed correctly to managers. Notifications were verified via test emails.

Completion Status: Process automation implemented and successfully verified.

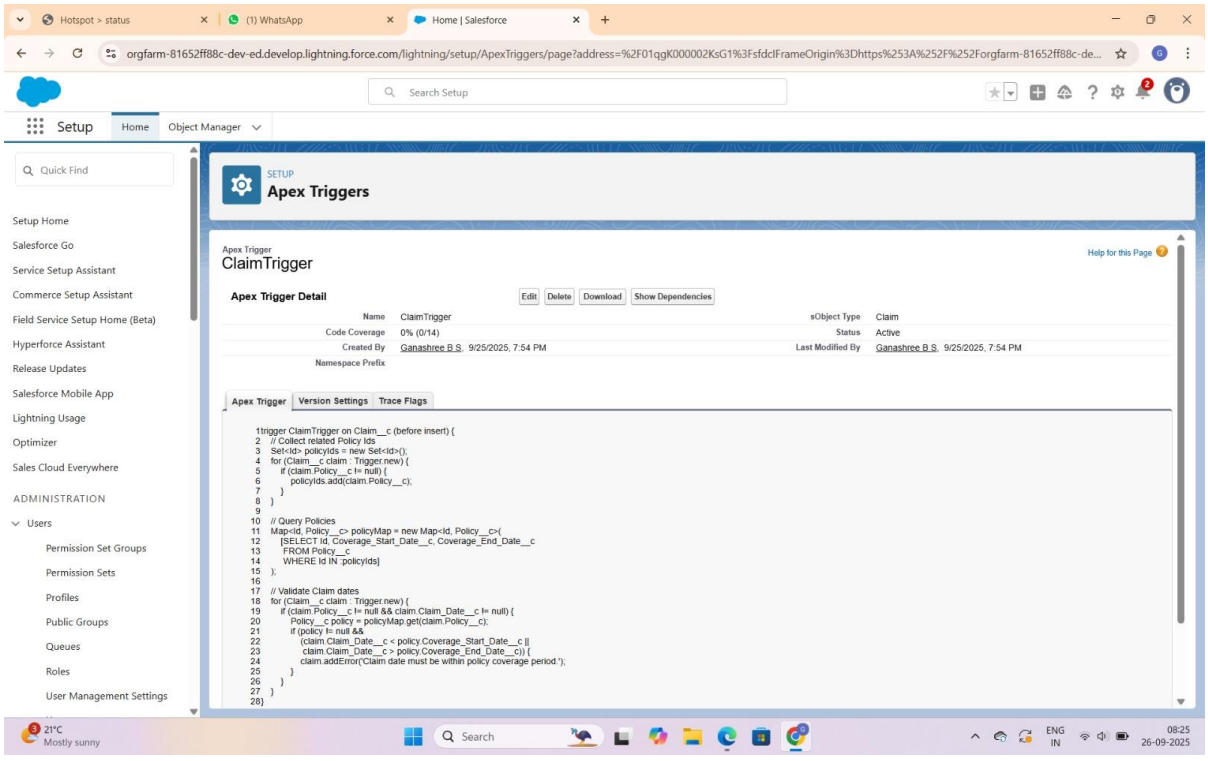
5. Apex Programming

Purpose:

Custom logic was required to extend Salesforce's standard functionality. Apex programming allowed automation of tasks not possible with declarative tools.

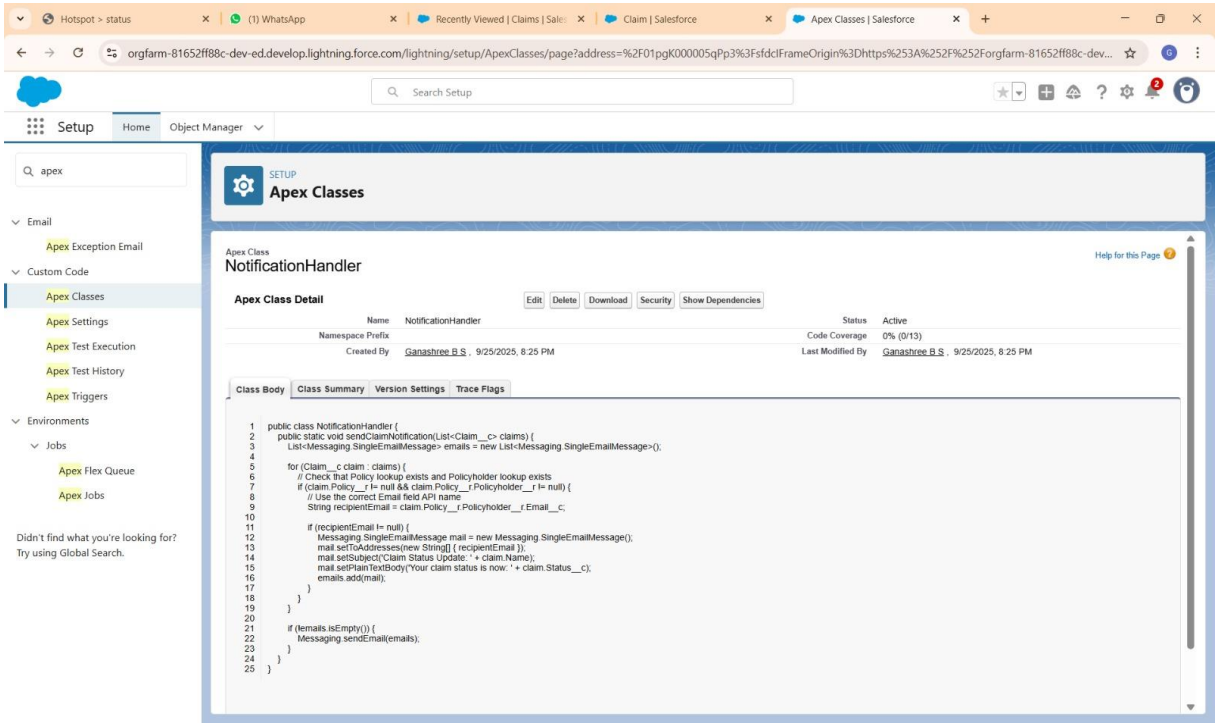
Implementation:

1. Triggers:
   - Before Insert: Ensured claim date cannot be in the future.
   - After Insert: Generated unique claim reference numbers.



2. Apex Classes:
   - ClaimService.cls: Managed claim validation and calculations.

- NotificationHandler.cls: Sent notifications via email/SMS.



3. Batch Apex:
   - Scheduled batch class to generate monthly pending claims report.

4. Test Classes:
   - Created unit tests with 75%+ coverage.
   - Validated logic for triggers, classes, and batch jobs.

Business Impact:

Apex extended Salesforce's power by providing advanced validations and reporting features. Managers gained automated monthly insights, and policyholders benefited from instant notifications.

Testing/Verification:

Ran unit tests in Developer Console with successful code coverage. Deployed triggers and classes to Sandbox for further validation.

Completion Status: Apex components successfully developed, tested, and deployed.