Project Title:

"Fraud Detection in Financial Transactions – AI-Powered Monitoring & Alert System"

Industry:

Finance / Banking

Target Users:

Banking Customers, Fraud Analysts, Compliance Officers, Risk Management Teams

---

Problem Statement

Banks and financial institutions process **millions of transactions daily** through credit cards, online banking, and mobile wallets. However, detecting fraudulent transactions in real-time remains a major challenge:

- **Manual monitoring** is slow, inconsistent, and prone to errors.
- Fraudsters use **sophisticated patterns** (small, rapid transactions, location mismatches, unusual purchase behavior) that are hard to catch without automation.
- Customers often report **fraudulent transactions too late**, leading to financial losses and reputational damage.

To address this, the organization wants to implement a **Salesforce-based fraud detection system** that can:

- Capture and analyze transaction data in real-time
- Flag suspicious transactions based on configurable rules and risk scores
- Alert both customers and fraud analysts instantly
- Maintain case records for investigations and compliance reporting
- Provide dashboards for management to monitor fraud trends

PHASE 1 : PROBLEM UNDERSTANDING AND INDUSTRY ANALYSIS

Requirement Gathering :

- Create custom objects for storing transaction records (Transaction, Merchant, Customer Profile).
- Define fields (amount, location, merchant, device, risk score).
- Configure validation rules and workflow rules for anomaly detection.
- Set up Process Builder / Flow for fraud alerts & case creation.

- Use Einstein Discovery / AI for anomaly detection and fraud scoring.
- Enable Reports & Dashboards for fraud trend analysis.

Stakeholder Analysis

- Banking Customers → Receive real-time fraud alerts via Salesforce Email / SMS APIs (Twilio, Digital Engagement).
- Fraud Analysts → Investigate cases in Service Console with complete transaction details.
- Compliance Officers → Use Reports & Dashboards for AML/KYC compliance audits.
- Risk Managers → Monitor fraud trends & insights with Einstein Analytics.
- Salesforce Admins/Developers → Configure data model, automation, API integration, and AI models.

Business Process Mapping (Salesforce Flow)

1. **Transaction Capture**
    - Transactions are stored in the Transaction Object via API integration or manual upload.
2. **Fraud Detection**
    - Flows + Apex Triggers apply business rules (high-value, unusual frequency, overseas use).
    - Einstein AI assigns probability score for fraud risk.
3. **Fraud Alerts & Notifications**
    - Flow / Process Builder sends Email & SMS alerts to customers.
    - If unverified → escalated to Fraud Analyst.
4. **Case Creation**
    - Fraud cases auto-created in Service Cloud.
    - Assigned to Fraud Analyst queue.
5. **Case Investigation**
    - Analyst investigates via Service Console.
    - Case status updated: False Positive / Confirmed Fraud.
6. **Reporting**

    - Dashboards provide insights: fraud patterns, resolution timelines, and analyst workload.

Industry-Specific Use Case Analysis

- **Transaction Monitoring** → Real-time anomaly detection with custom objects + triggers.
- **Fraud Alerts** → Workflow + Email/SMS integration for instant customer communication.
- **Case Management** → Service Cloud lifecycle for tracking fraud investigations.
- **Risk Scoring** → Einstein AI assigns severity levels.
- **Analytics** → Dashboards for fraud volume, types, trends, and regulatory reporting.

AppExchange Exploration

- **Einstein Analytics Apps** → Pre-built dashboards for fraud detection.
- **Financial Services Cloud** → Banking-specific data model.
- **Security & Compliance Apps** → Data encryption & monitoring.
- **Twilio / SMS-Magic** → Customer notifications for fraud alerts.
- **Case Management Accelerators** → Extend fraud case lifecycle in Service Cloud.

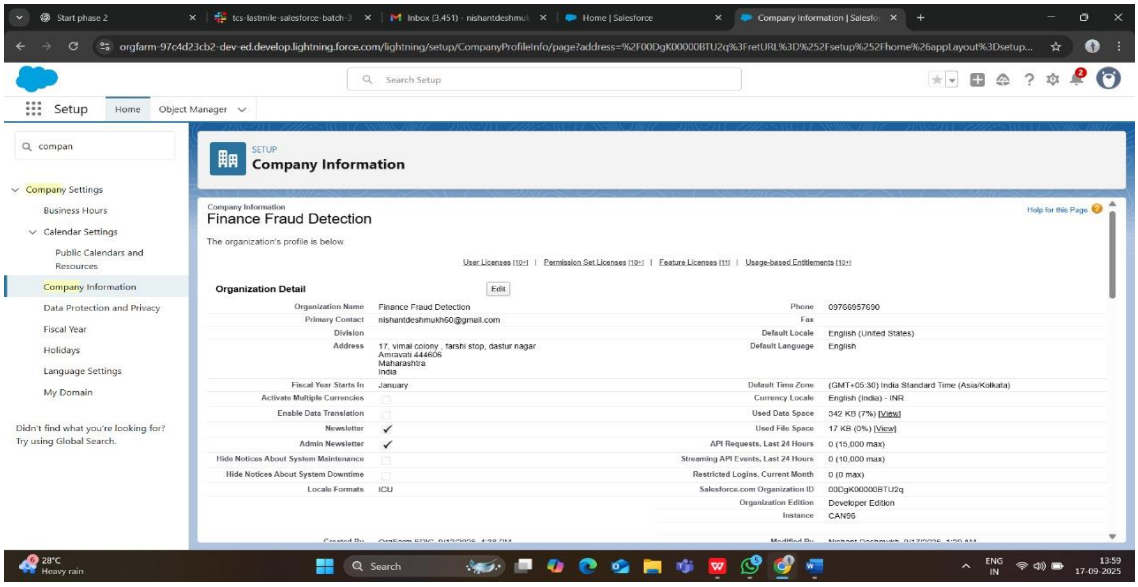
Phase 2: Org Setup & Configuration

1. Salesforce Edition

- Used **Developer Edition Org** for Fraud Detection setup.

2. Company Profile Setup

- Company Name: **Finance Fraud Detection**
- Primary Contact: Email
- Default Locale: **English (India)**
- Default Time Zone: **(GMT+05:30) India Standard Time (Asia/Kolkata)**
- Default Currency: **INR**



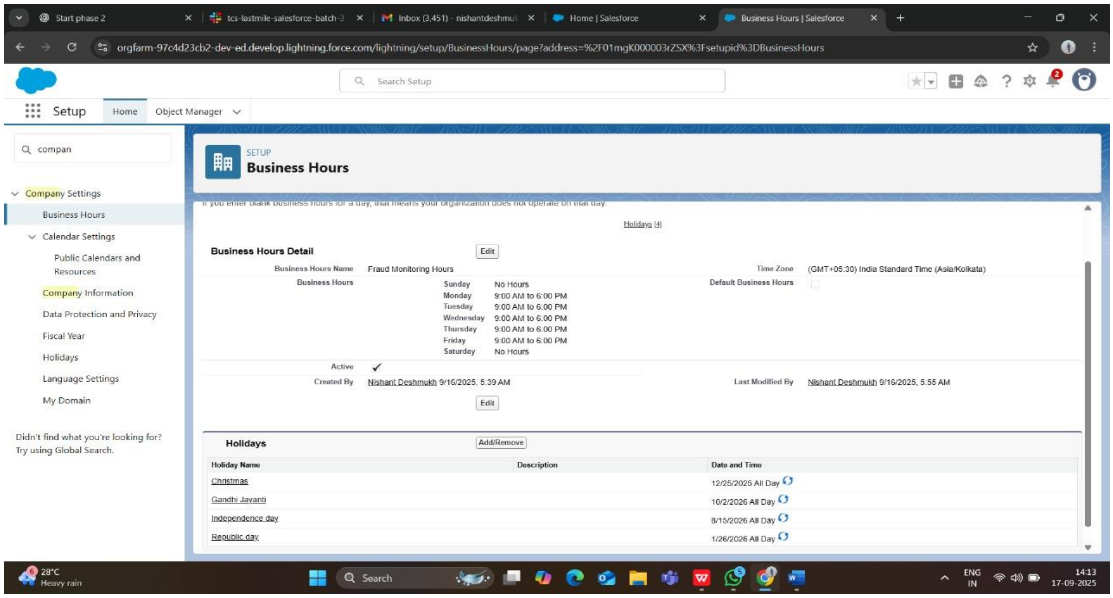
3. Business Hours & Holidays

- Name: **Fraud Monitoring Hours**
- Time Zone: **Asia/Kolkata (GMT+05:30)**
- Default: ✓
- Active: ✓
- Business Hours: **Monday – Friday, 9:00 AM – 6:00 PM**
- Holiday **– Sat/Sun**

Add common public holidays (India 2025):

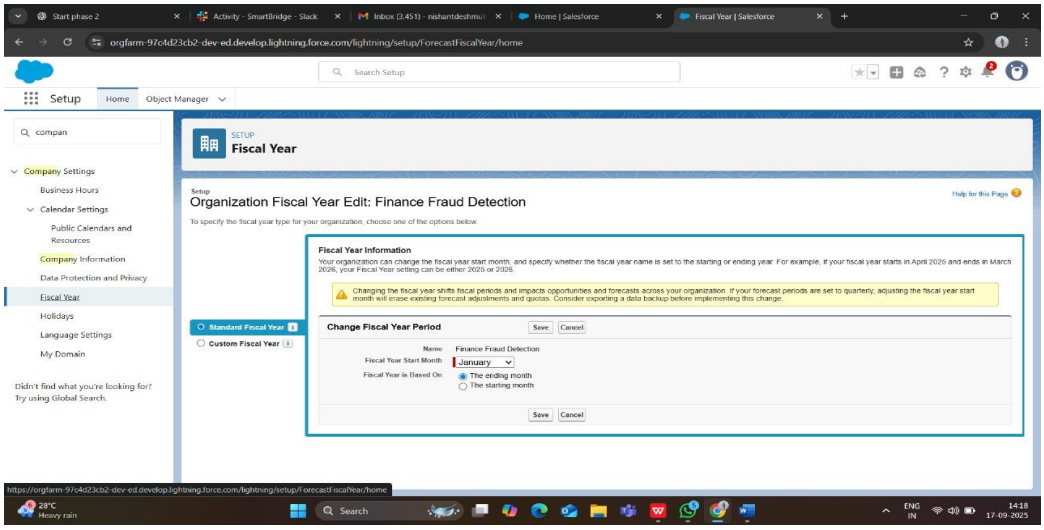
- Republic Day – 26 Jan 2025

- Independence Day – 15 Aug 2025
- Gandhi Jayanti – 2 Oct 2025
- Christmas – 25 Dec 2025
- Mark All-Day ✓



4. Fiscal Year Settings

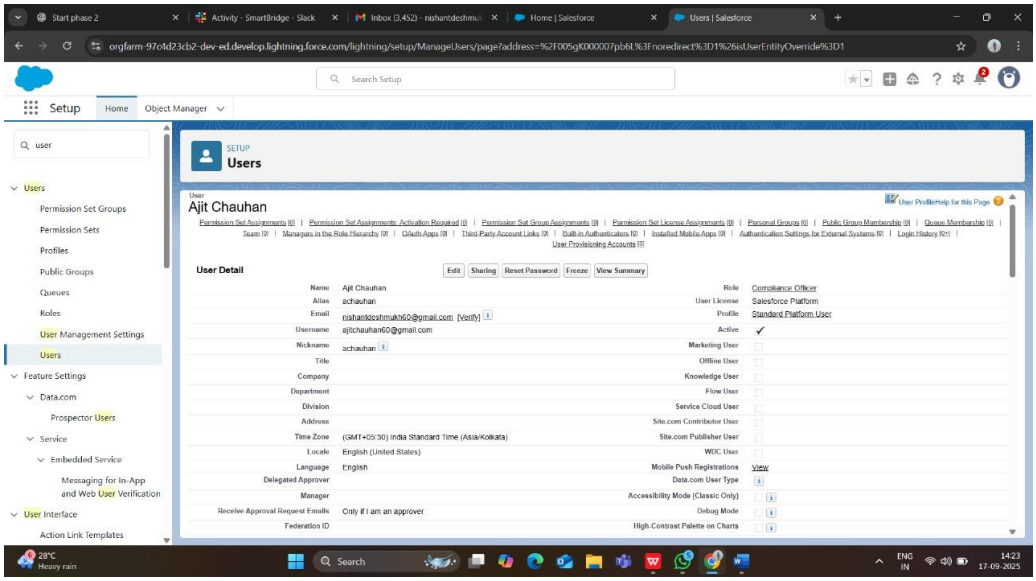
- Used **Standard Fiscal Year (Jan–Dec)**.

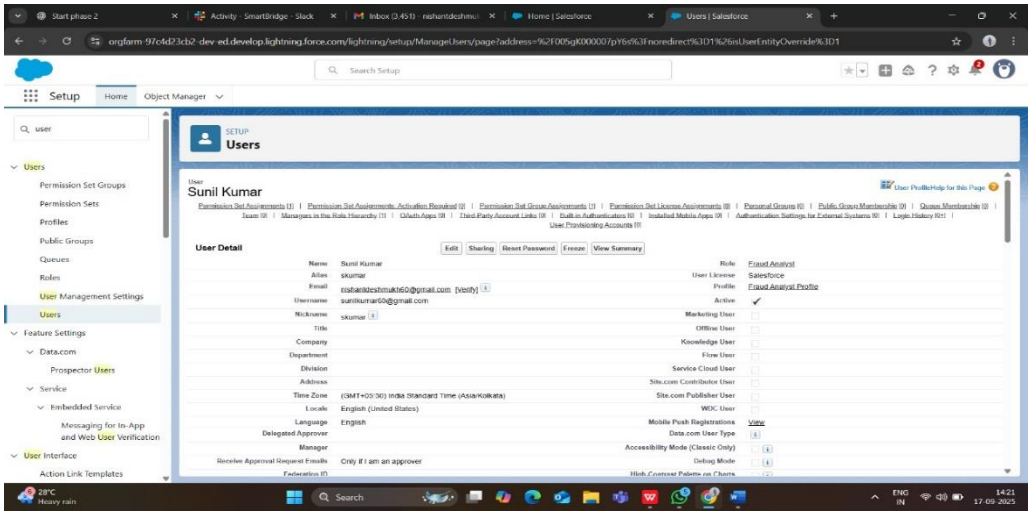


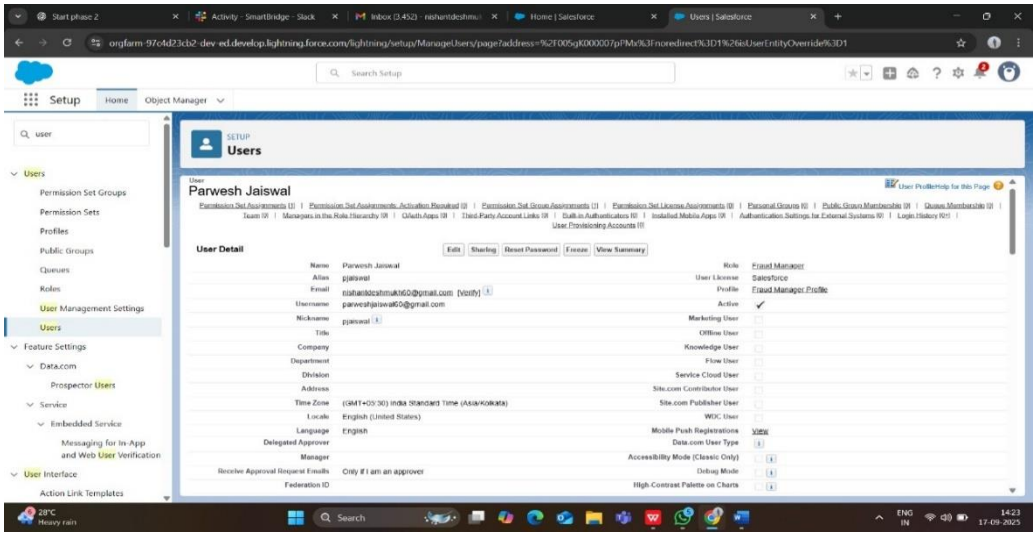
5. User Setup & Licenses

Created Users:

- **Fraud Manager**
- **Fraud Analyst**
- **Compliance Officer**

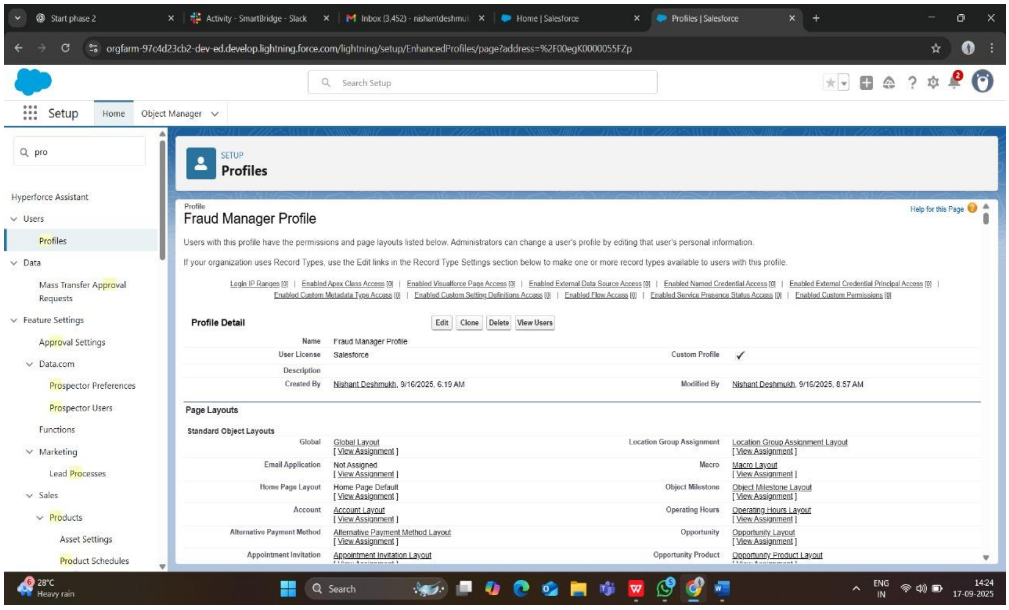

Setup
Home
Object Manager
SETUP
Users
Ajit Chauhan
User Detail
Edit
Sharing
Reset Password
Freeze
View Summary

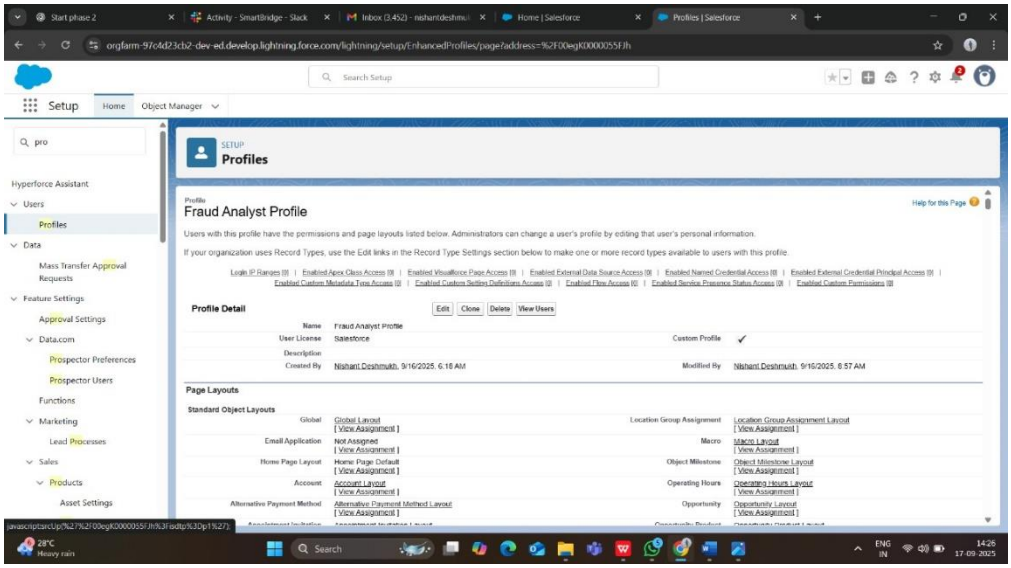

Setup
Home
Object Manager
SETUP
Users
Sunil Kumar
User Detail
Edit
Sharing
Reset Password
Freeze
View Summary

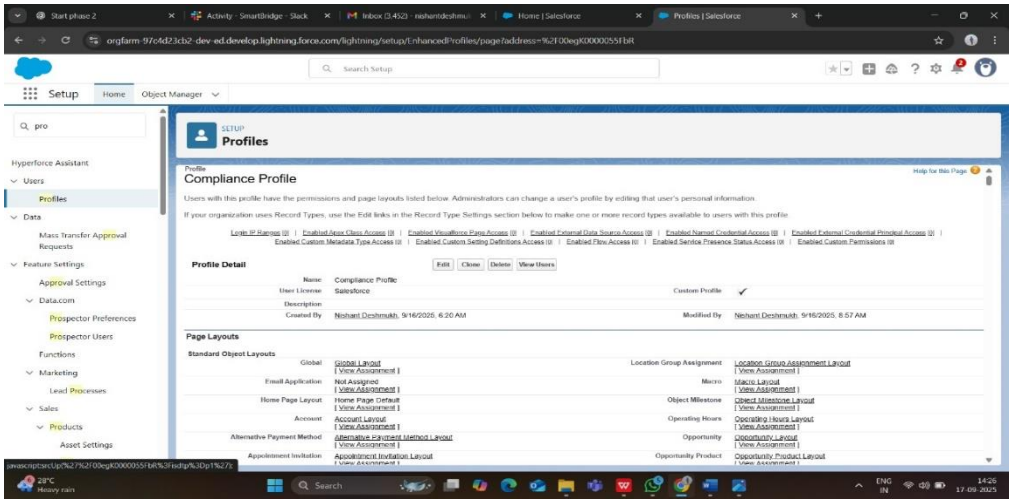

Setup
Home
Object Manager
SETUP
Users
Parwesh Jaiswal
User Detail
Edit
Sharing
Reset Password
Freeze
View Summary

6. Profiles

- **Fraud Manager Profile**: Full access to Fraud Cases, Transactions, Reports.
- **Fraud Analyst Profile**: Create/Update Fraud Cases, edit Transactions.
- **Compliance Officer Profile**: Read-only access to all Fraud Cases (monitoring only).

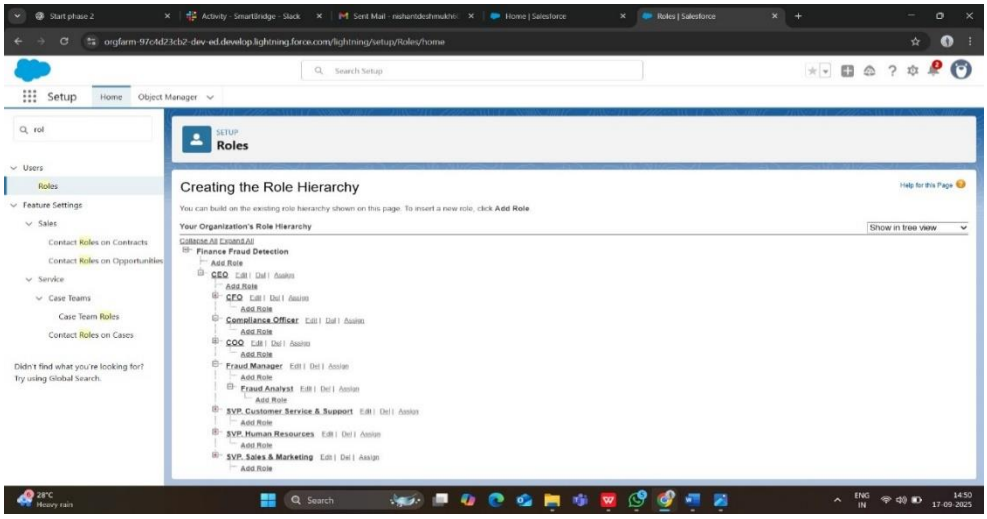






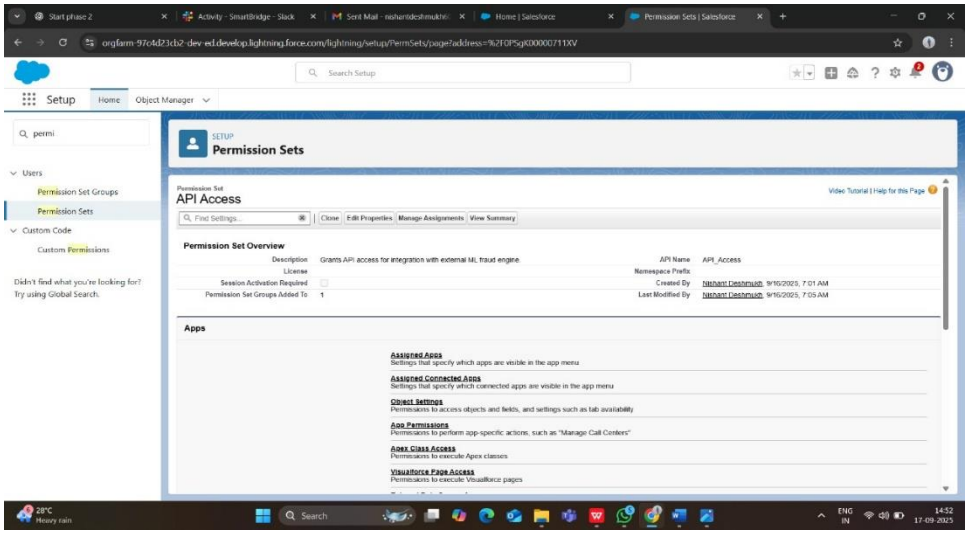
7. Roles

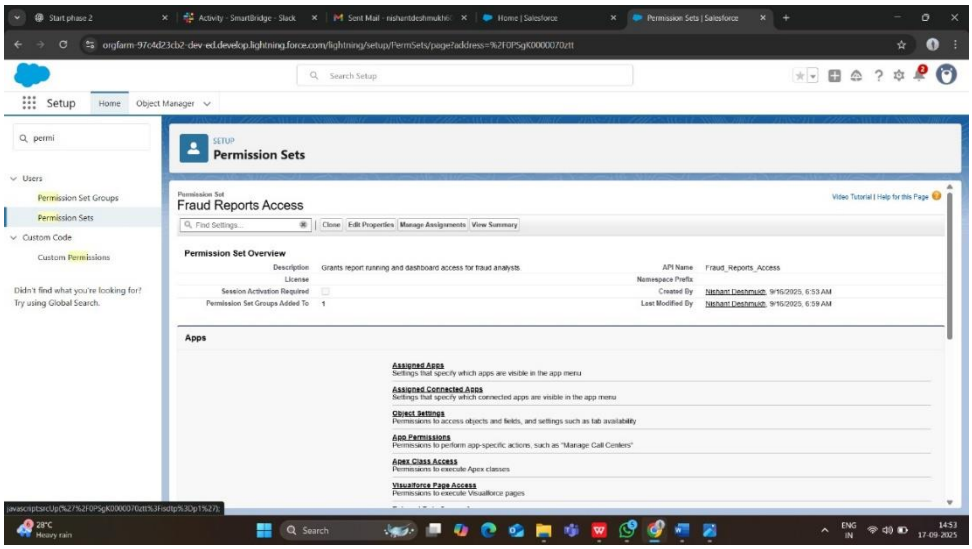
- Fraud Manager (top) → full visibility.
- Fraud Analysts → below manager.
- Compliance Officer → parallel role, monitoring only.



8. Permission Sets

- **Fraud Reports Access**: Run Reports + Dashboards.
- **API Access**: Enable integration with external AI Fraud Detection Engine.





9. OWD (Org-Wide Defaults)

- **Transaction (Custom Object)** → Private.
- **Fraud Case (Custom Object)** → Private.
- **Customer (Account/Contact)** → Public Read Only.

10. Sharing Rules

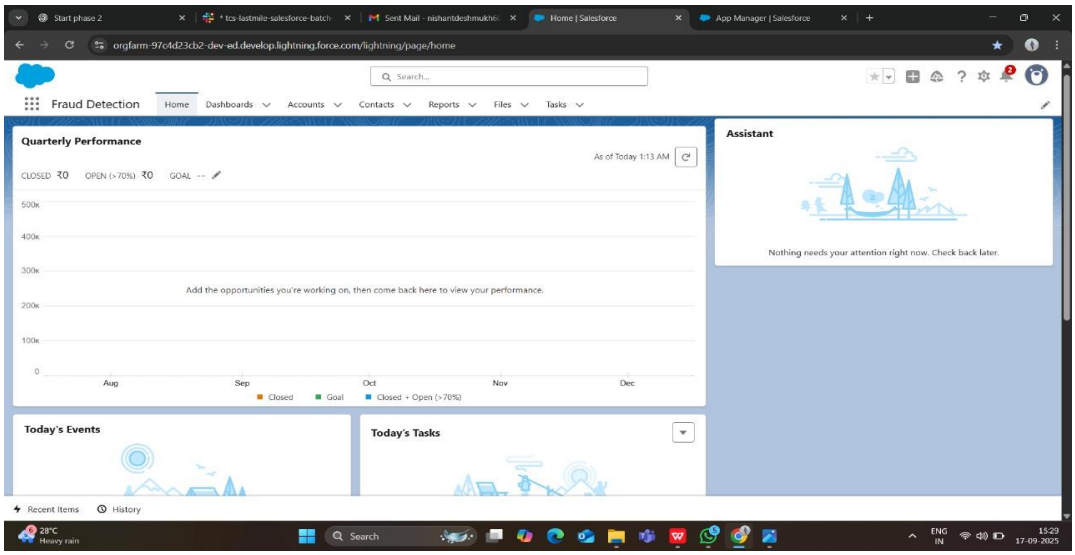
- Share all **High-Risk Fraud Cases** with Fraud Managers.
- Share all **Cases Under Investigation** with Compliance Officers.

11. Login Access Policies

- Restricted login hours for Fraud Analysts (9 AM – 6 PM).
- Fraud Manager & Compliance Officer → 24/7 access.

12. Dev Org Setup

- Configured **My Domain** for Fraud Detection App.
- Created custom **Fraud Detection Lightning App** for users.



13. Sandbox Usage

- If real deployment, Sandbox would be used for testing before Production rollout.

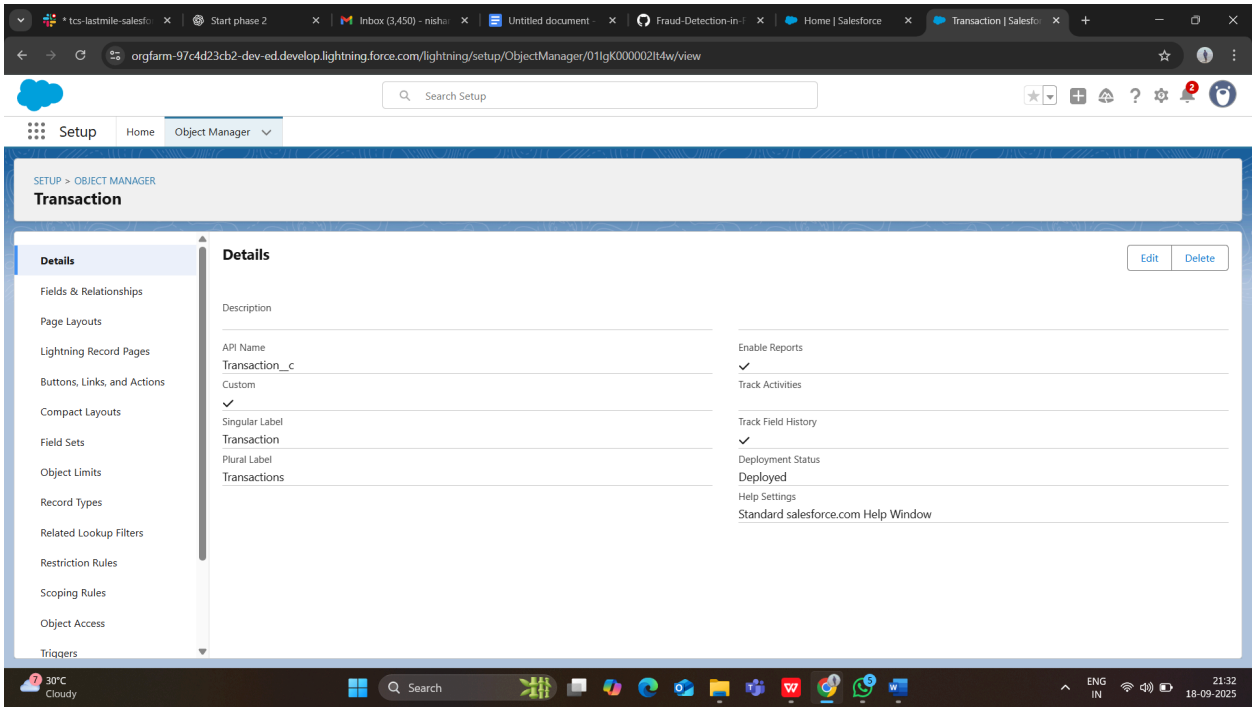
14. Deployment Basics

- For this project, deployment limited to **Developer Edition** (no Production).

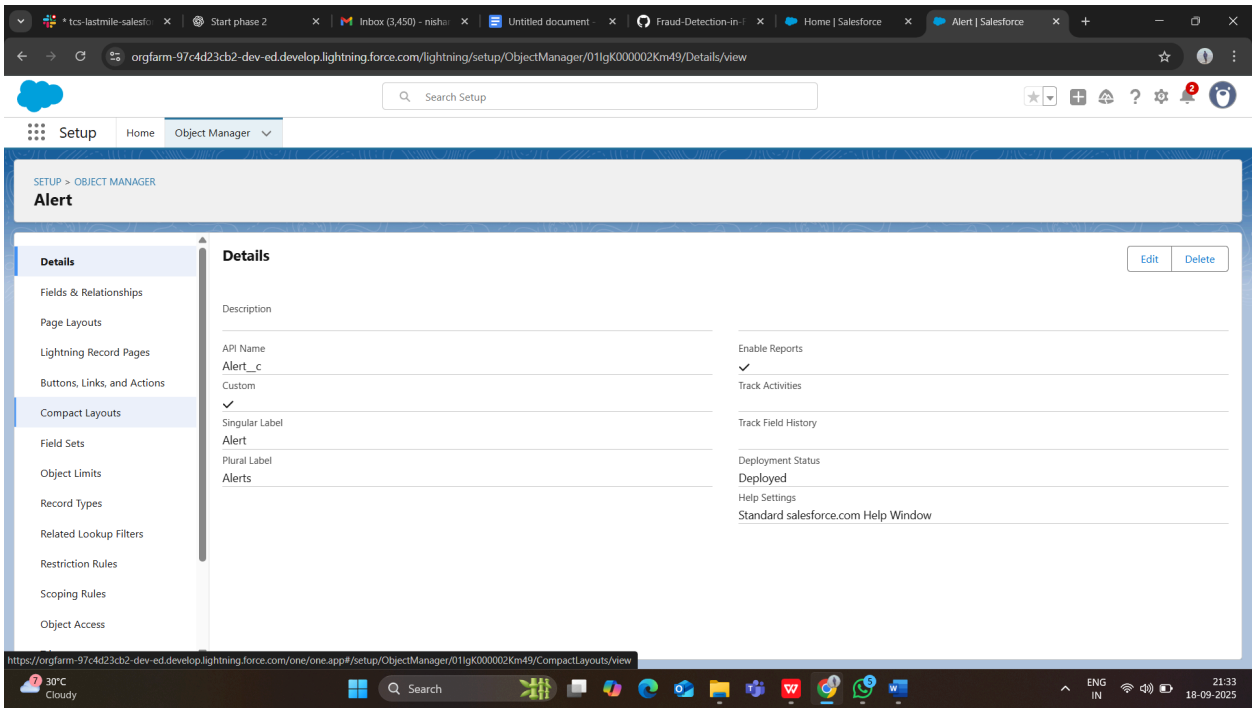
Phase 3: Data Modeling & Relationships

1. Standard & Custom Objects

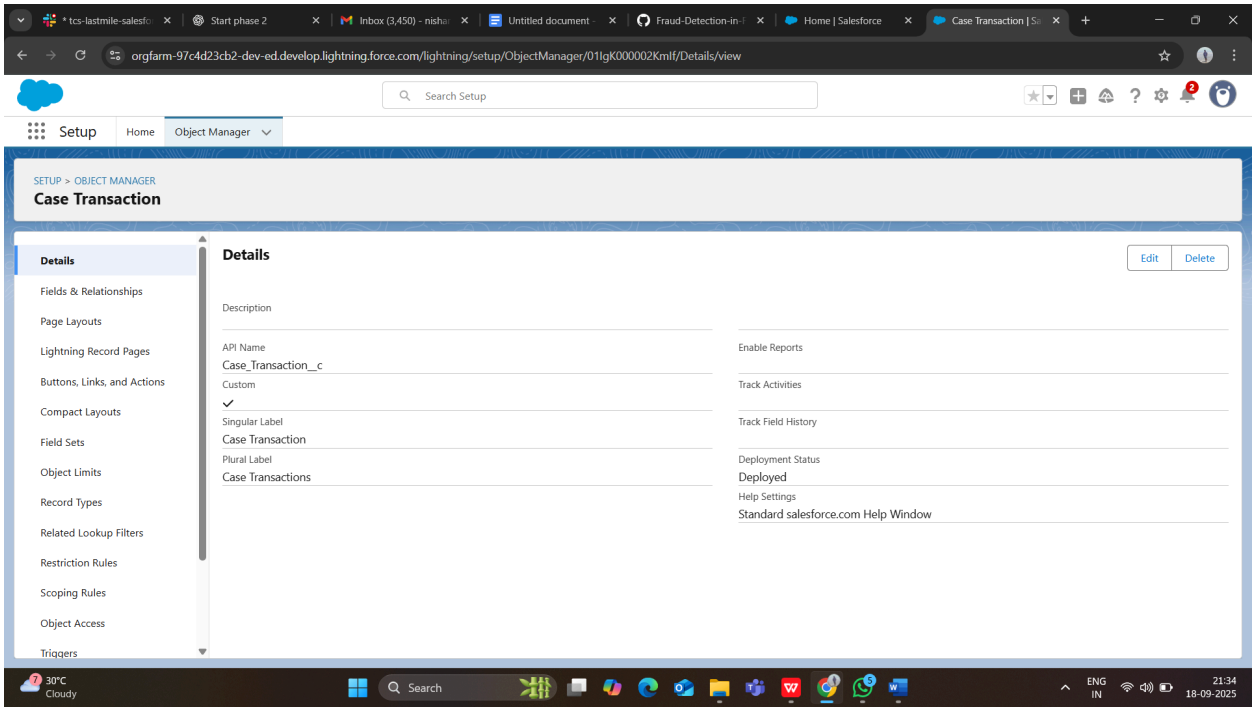
- **Standard:**
    - **Account** (represents customer or business)
    - **Contact** (customer representatives).
- **Custom:**
    - **Transaction** (each financial activity).



- **Alert** (potential fraud case).

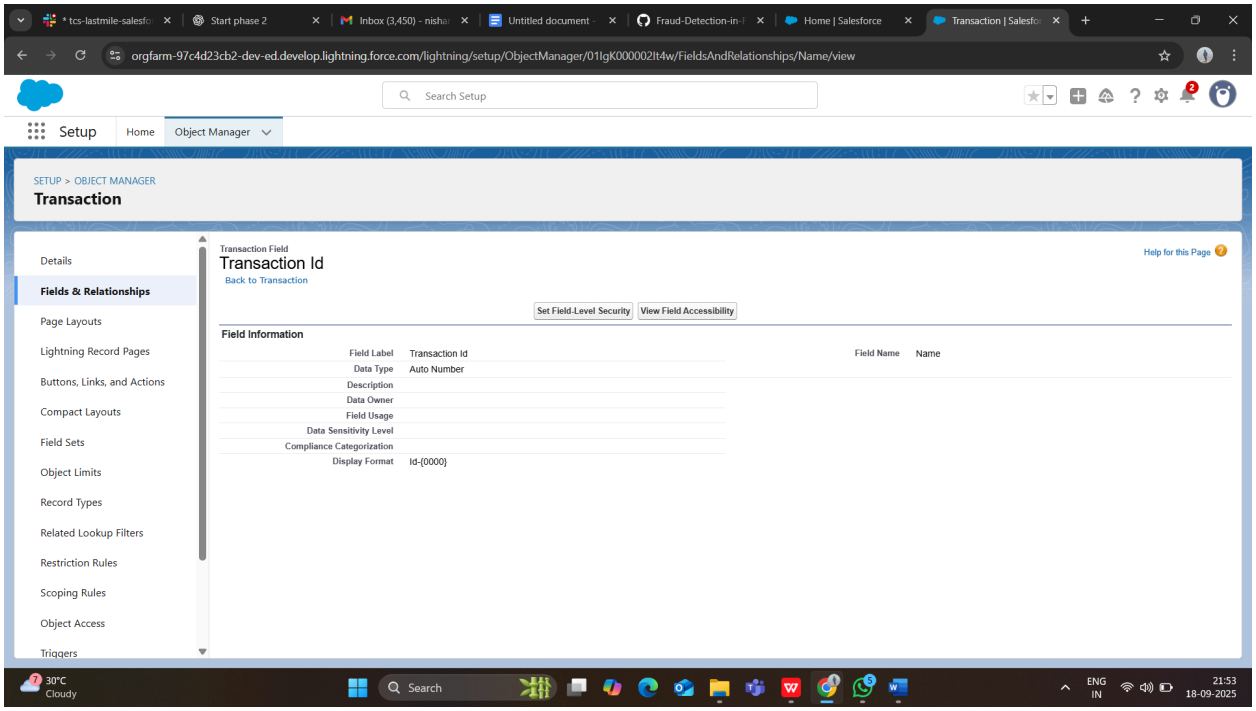


- **Case Transaction** (junction between Case & Transaction if needed).

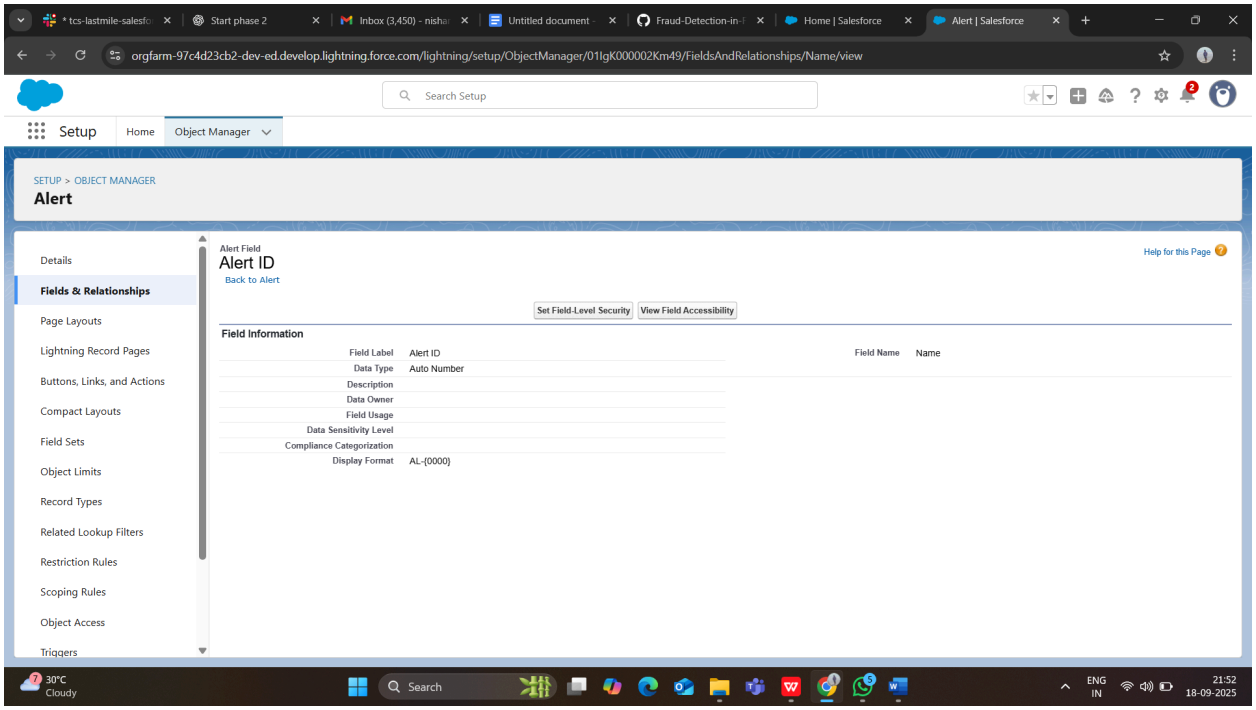


2. Fields

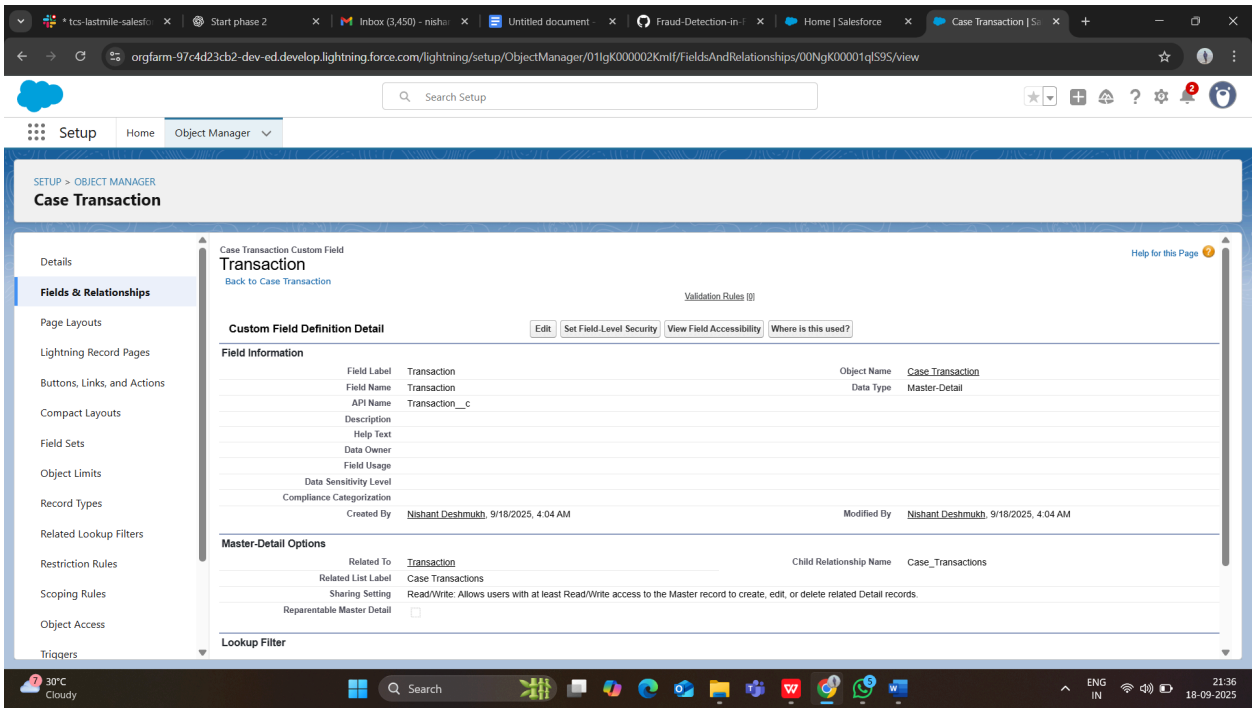
- **Transaction:** Amount, Date, Location, Channel (ATM, Online, POS), Status (Approved/Declined).



- **Alert:** Risk Score, Alert Type (Suspicious Login, Unusual Transaction), Alert Status (Open, Investigating, Closed).



- **Case Transaction (junction):** Links Case with one or more Transactions.



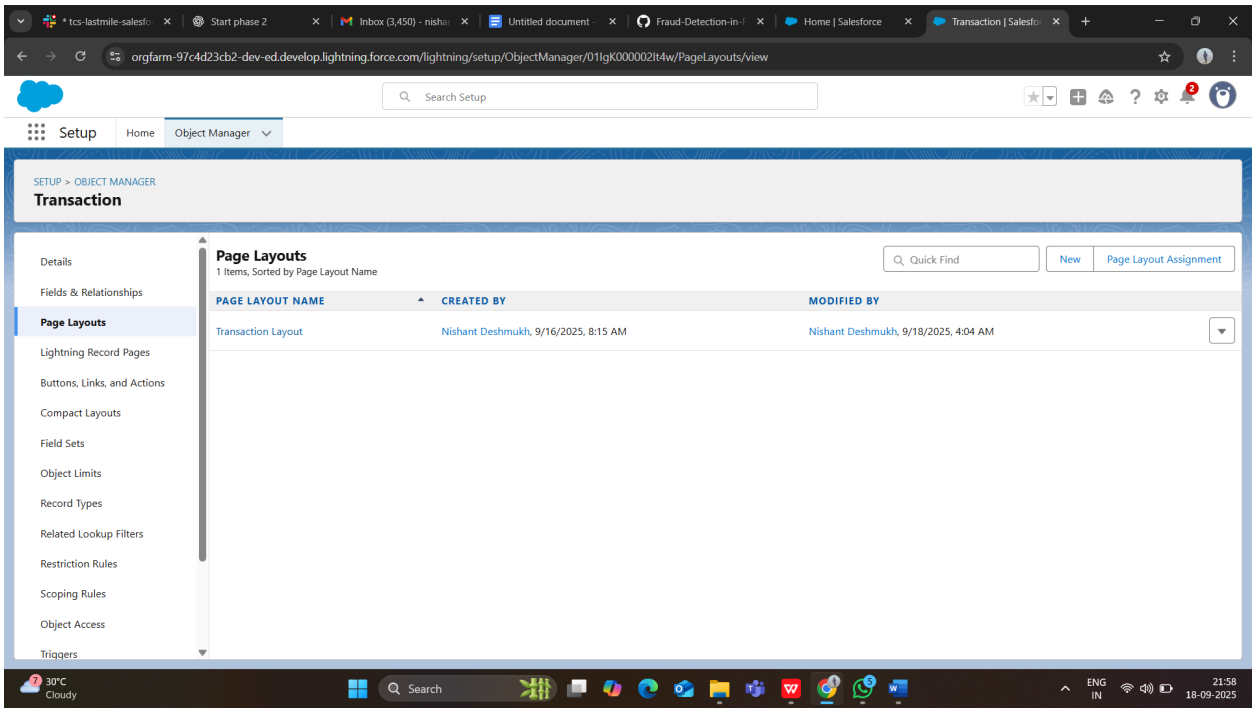
3. Record Types

- **Transaction** → “Domestic Transaction” vs “International Transaction.”
- **Alert** → “High Risk” vs “Medium Risk” vs “Low Risk.”

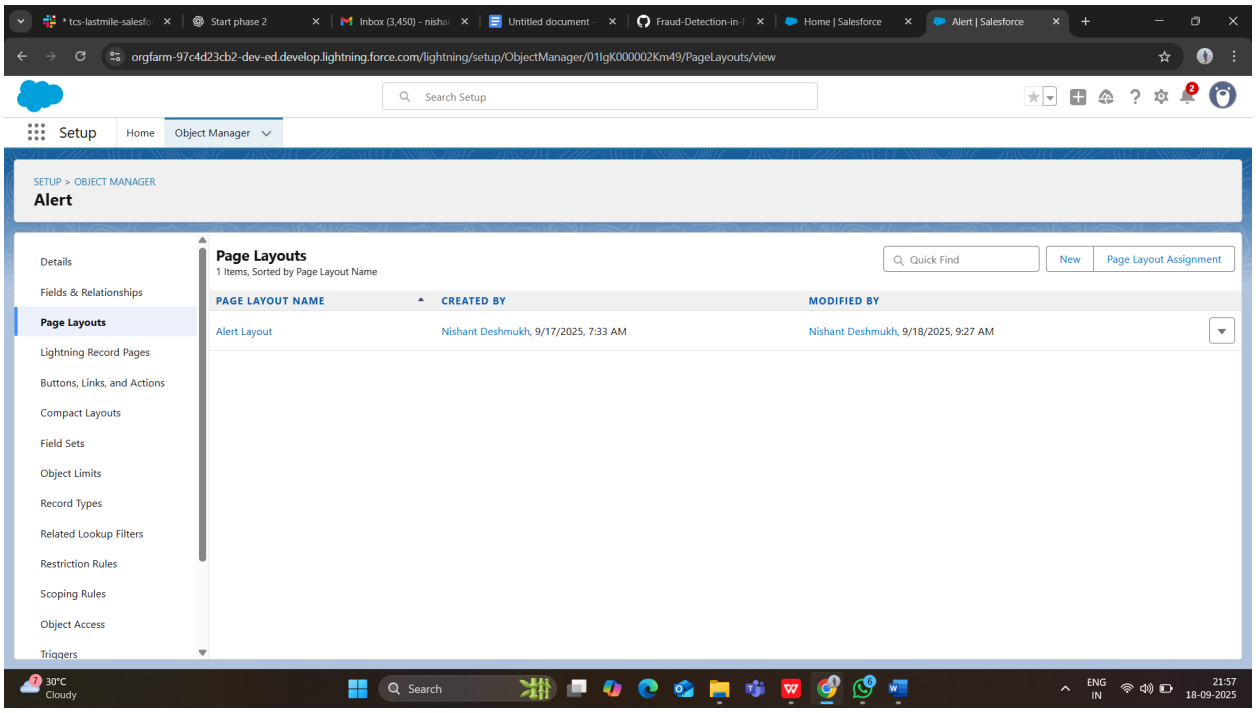
---

4. Page Layouts

- **Transaction page** → shows related customer, alerts, and cases.



- **Alert page** → shows related transactions and assigned fraud analyst.

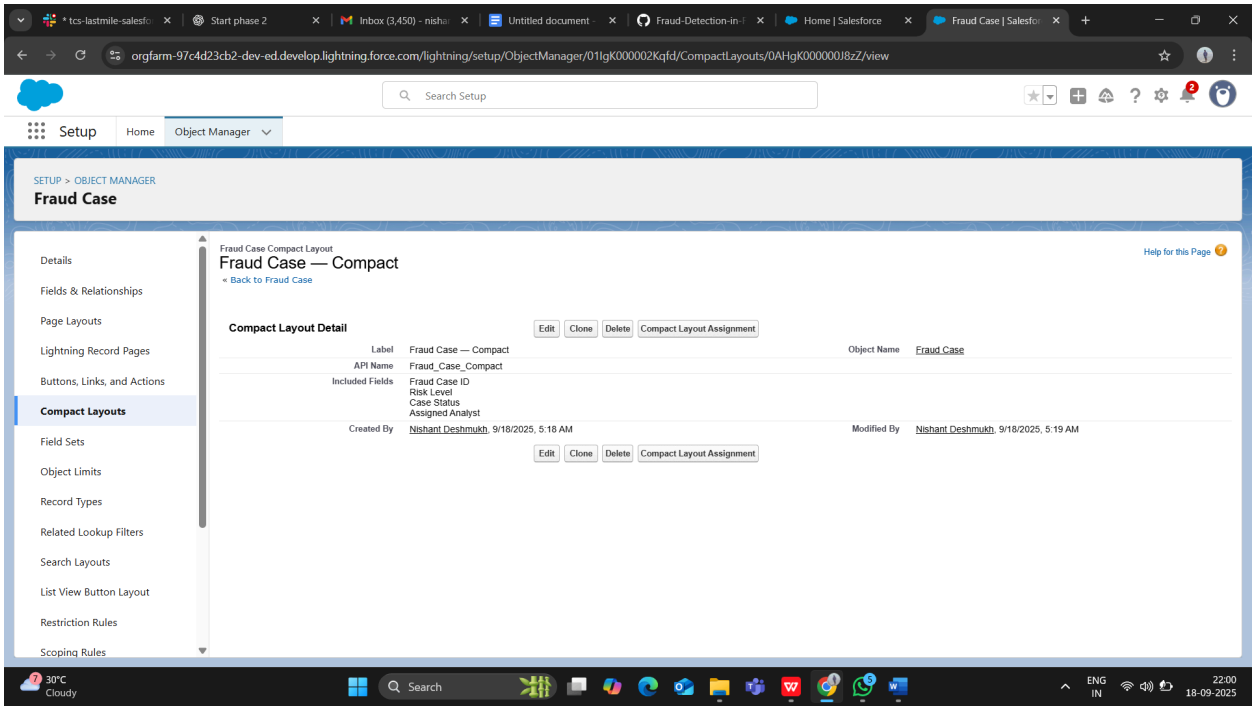


- **Case page** → shows linked alerts and transactions.

---

5. Compact Layouts

- **Mobile (Transaction):** Amount, Date, Status.
- **Mobile (Alert):** Alert Type, Risk Score, Status.

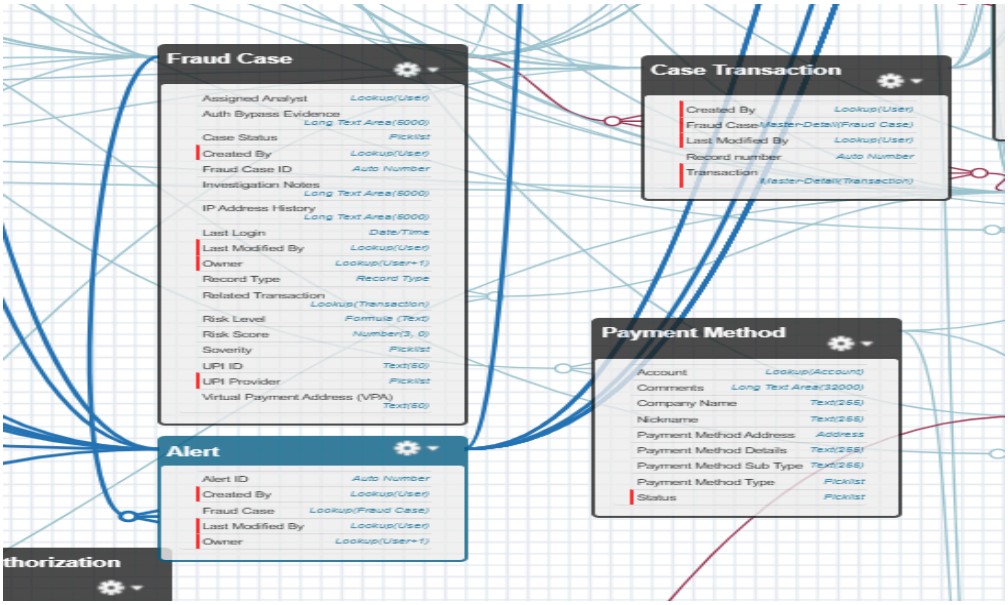


---

6. Schema Builder

- Use **Schema Builder** to visually design:
    - Account → Transaction (Lookup).
    - Transaction → Alert (Lookup).

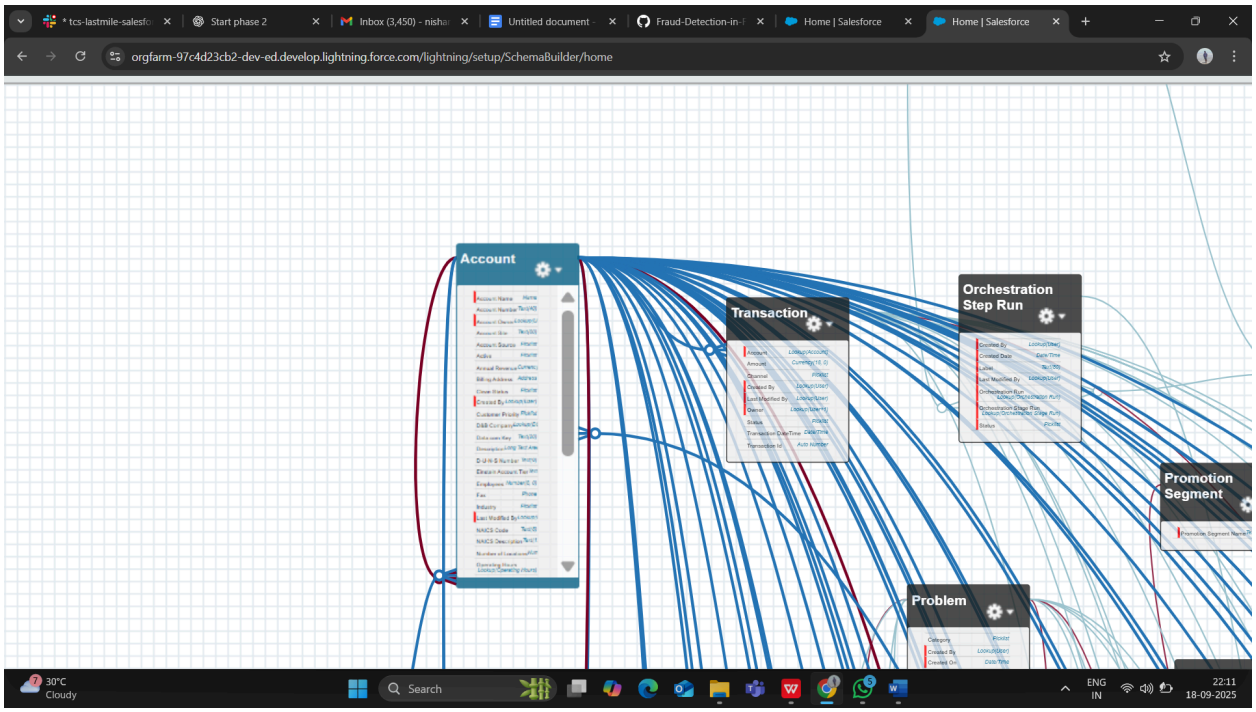
- Case ↔ Transaction (Junction).



---

7. Lookup vs Master-Detail vs Hierarchical

- **Account ↔ Transaction → Lookup** (transactions don't own accounts, but reference them).



- **Transaction ↔ Alert → Lookup** (one transaction may trigger multiple alerts).

---

8. Junction Objects

- **Case Transaction:**
    - Used when one Case contains multiple suspicious Transactions.
    - Enables many-to-many relationship.

---

9. External Objects

- If fraud detection system integrates with **external banking/credit systems**, use **External Objects** (via Salesforce Connect) to bring in real-time **transaction logs, blacklist databases, or KYC data** without storing them directly in Salesforce.

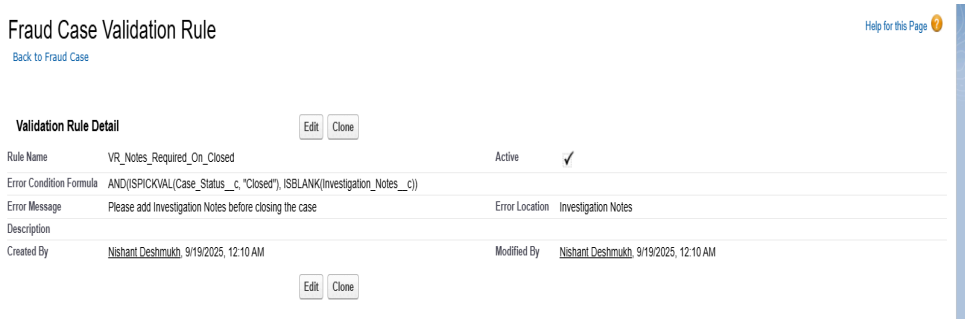
Phase 4: Process Automation (Admin)

👉 **Goal:** Automate fraud detection processes using **Flow Builder** (instead of legacy Workflow Rules / Process Builder).

---

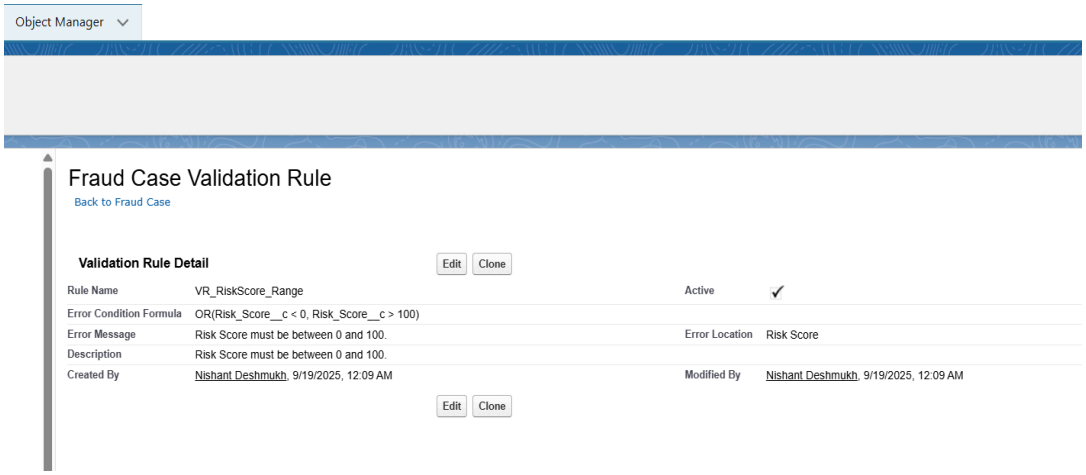
1. Validation Rules

- ***VR_Notes_Required_On_Closed***
  - ***Object:*** *Fraud Case*
  - ***Field:*** *Investigation_Notes__c*
  - ***Rule:*** *If* `Case_Status__c = "Closed"` *→ Investigation Notes must not be blank.*
  - ***Error Message:*** *“Please add Investigation Notes before closing the case.”*
  - ***Active:*** ✅ *True*



- ***VR_RiskScore_Range***
  - ***Object:*** *Fraud Case*
  - ***Field:*** *Risk_Score__c*
  - ***Rule:*** *Risk Score must be between 0 and 100.*
  - ***Error Message:*** *“Risk Score must be between 0 and 100.”*

- ***Active:*** ✅ *True*

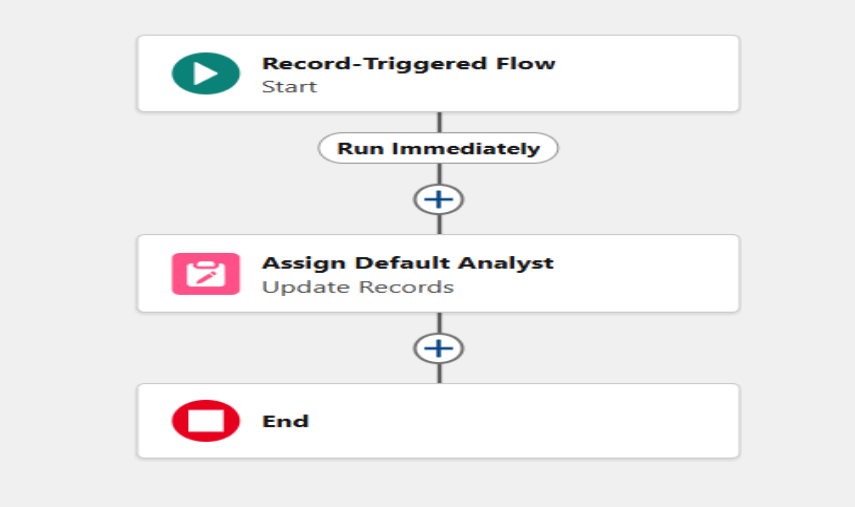


---

2. Record-Triggered Flows

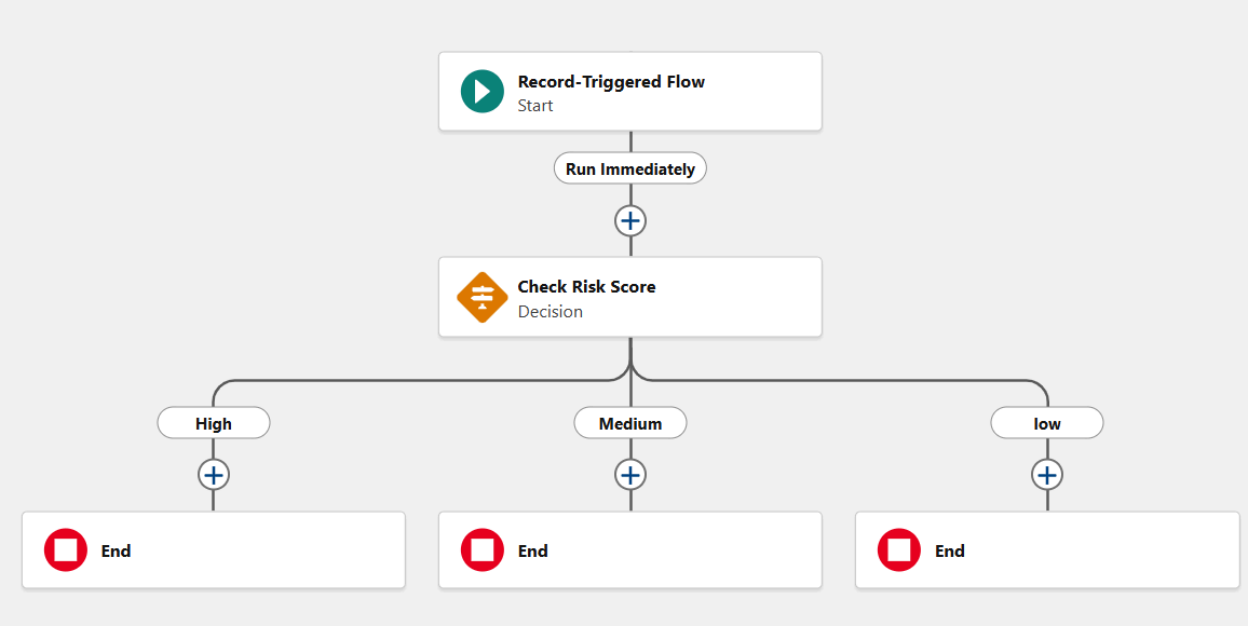
A) Auto-Assign Analyst

- **Object:** Fraud Case
- **Trigger:** When record is created
- **Condition:** Assigned_Analyst__c IS NULL
-
- **Action:** Update Records → Assigned_Analyst__c = Default Fraud Analyst



B) Auto-Calculate Severity from Risk Score

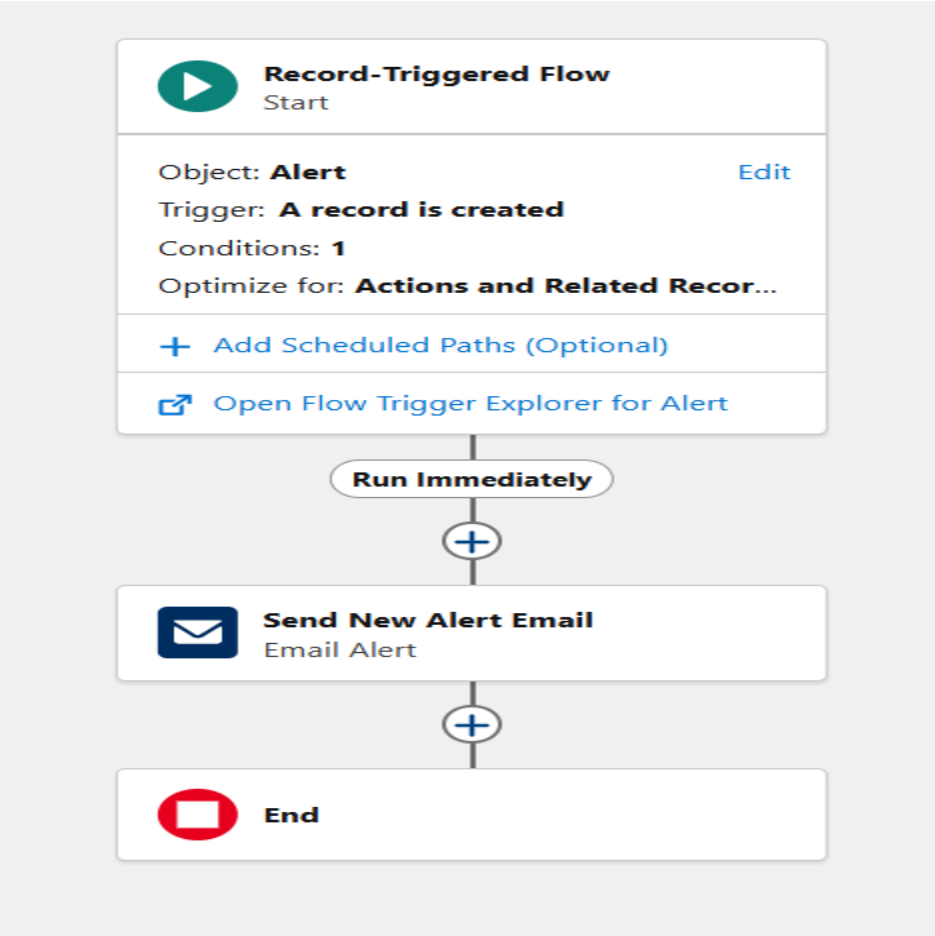
- **Object:** Fraud Case
- **Trigger:** When record is created or updated
- **Logic (Decision element):**
    - If Risk_Score__c ≥ 80 → Severity__c = High
    - If Risk_Score__c ≥ 50 → Severity__c = Medium
    - Else → Severity__c = Low
- **Action:** Update Records → update Severity__c



C) Fraud Alert Notification (Email)

- **Object:** Alert__c
- **Trigger:** When Alert is created

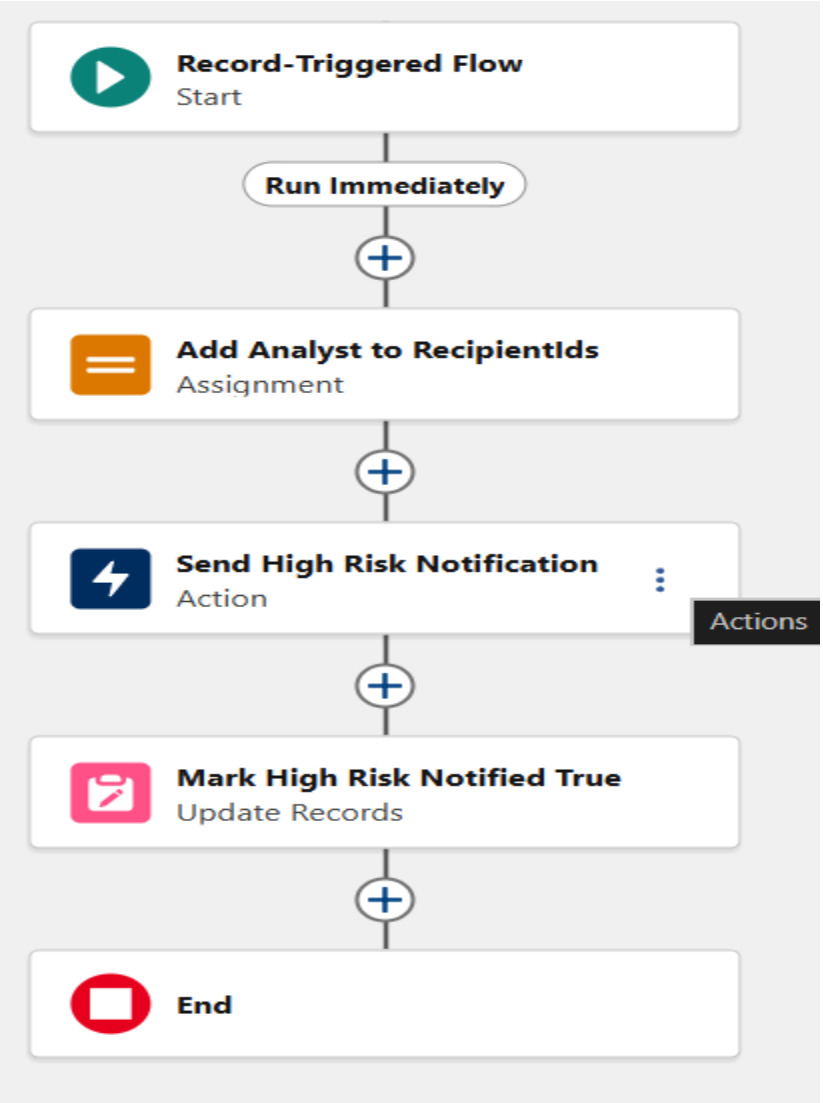
- **Condition:** Status__c = "Open"
- **Action:** Email Alert → send template “New Fraud Alert Notification”
- **Recipient:** Assigned Analyst



D) Fraud Case High-Risk Notification (In-App)

- **Object:** Fraud Case
- **Trigger:** When record is created
- **Condition:** Risk_Level__c = High
- **Action:** Send Custom Notification

- Title: *"High Risk Fraud Case Created"*
- Body: *"Fraud Case ID {!$Record.Name} is flagged as High Risk."*
- Recipient: Assigned Analyst



---

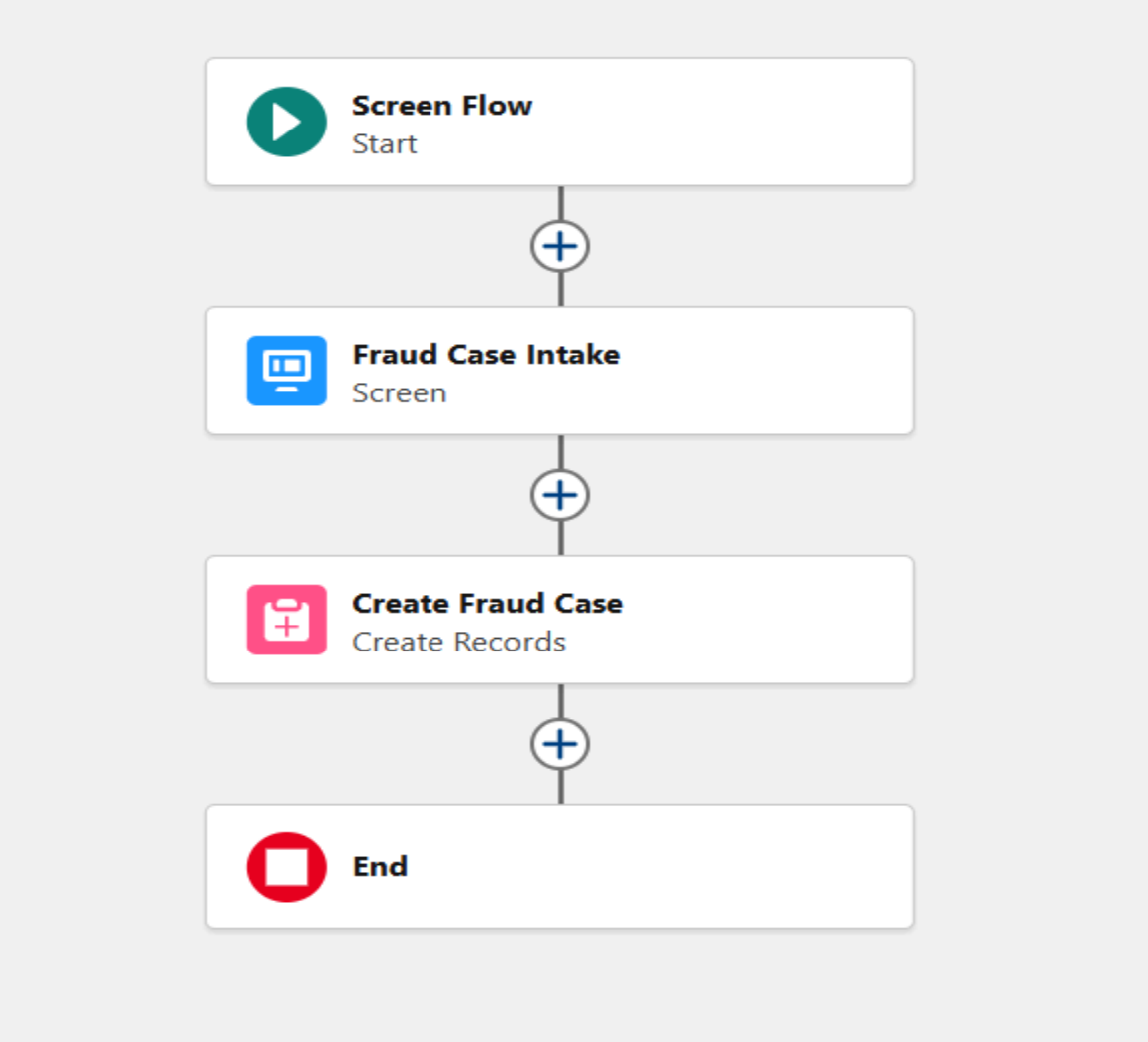
3. Screen Flow

Fraud Case Intake Form

- **Purpose:** Guided entry form for Analysts
- **Steps:**

1. Screen → Enter: Customer, Transaction, Risk Score, Investigation Notes
2. Create Records → Fraud Case record
3. Decision → If Risk_Score__c ≥ 80, show *High-Risk warning* message

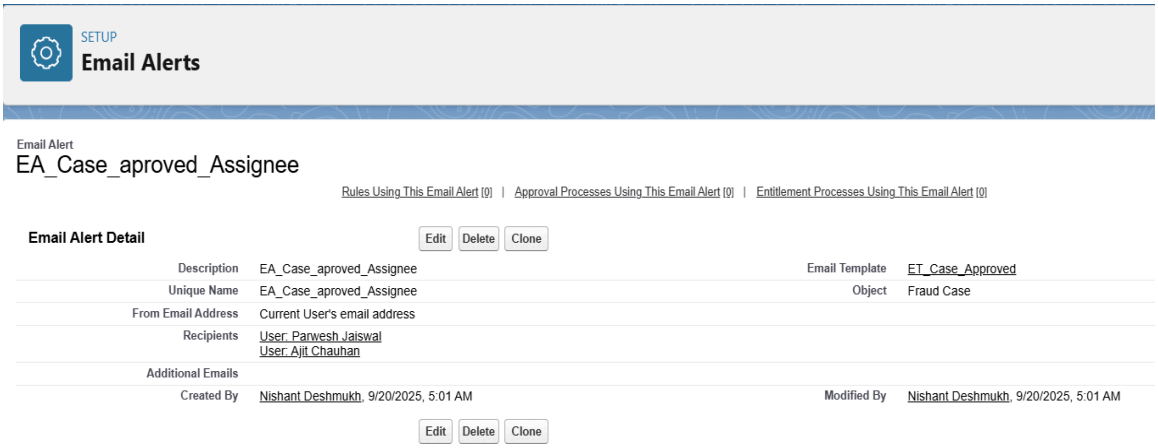
- **Placement:** Added to Fraud Analyst Lightning Page



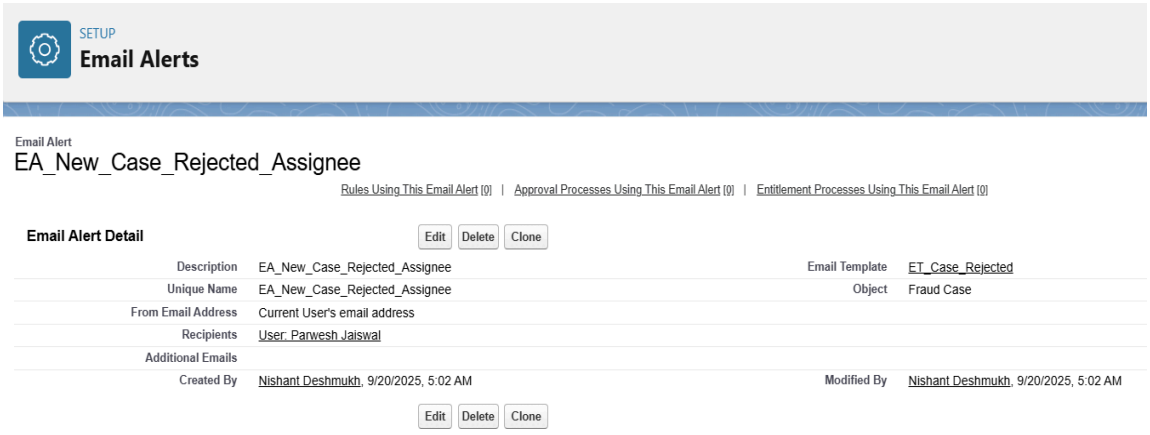
---

4. Email Alerts

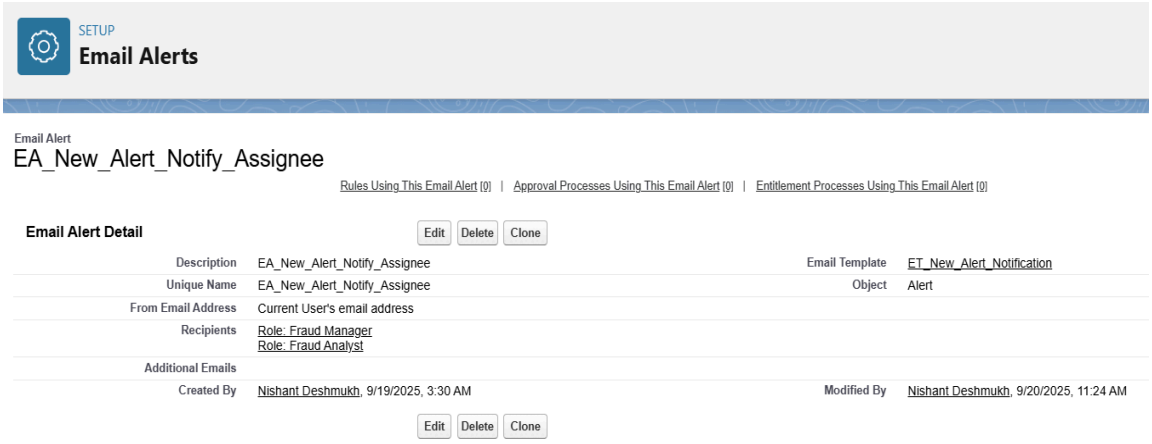
- **Templates Created:**
  - Case Approved Notification



- Case Rejected Notification



- New Fraud Alert Notification



- **Usage:** Linked inside Flows (Record-Triggered, Post-Approval).

---

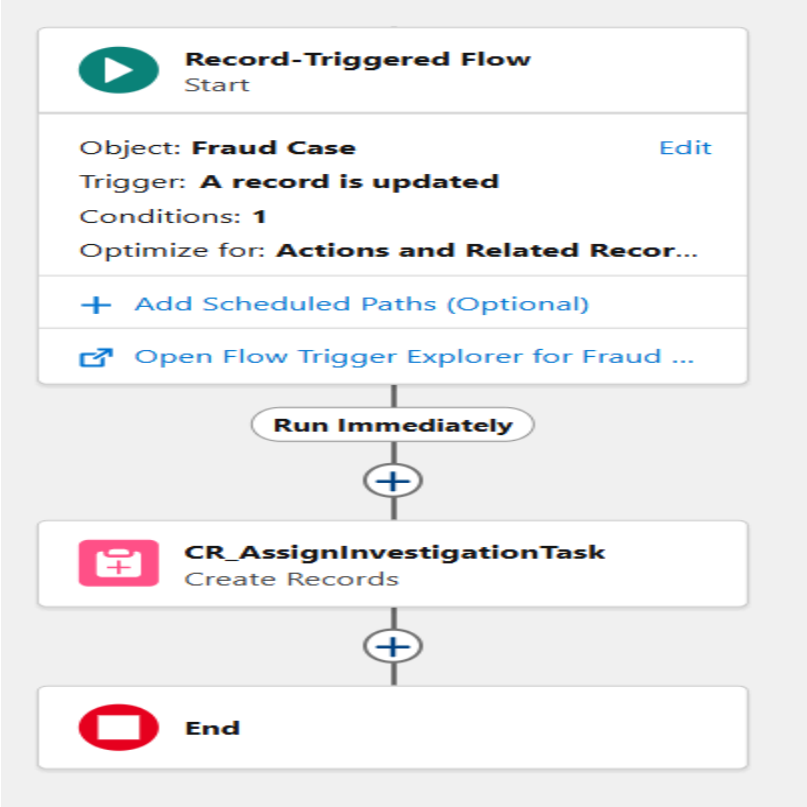
5. Field Updates (Inside Flow)

- **Scenario:** After Approval → Case Status = Resolved
- **Flow Step:** Update Records → Fraud_Case__c.Case_Status__c = “Resolved”

---

6. Tasks (Auto-Assigned to Analyst)

- **Trigger:** Case Approved (Final step of Approval Process Flow)
- **Action:** Create Task →
    - **Subject:** “Investigate Fraud Case”
    - **Assigned To:** Fraud Case → Assigned Analyst
    - **Related To:** Fraud Case ID
    - **Status:** Not Started



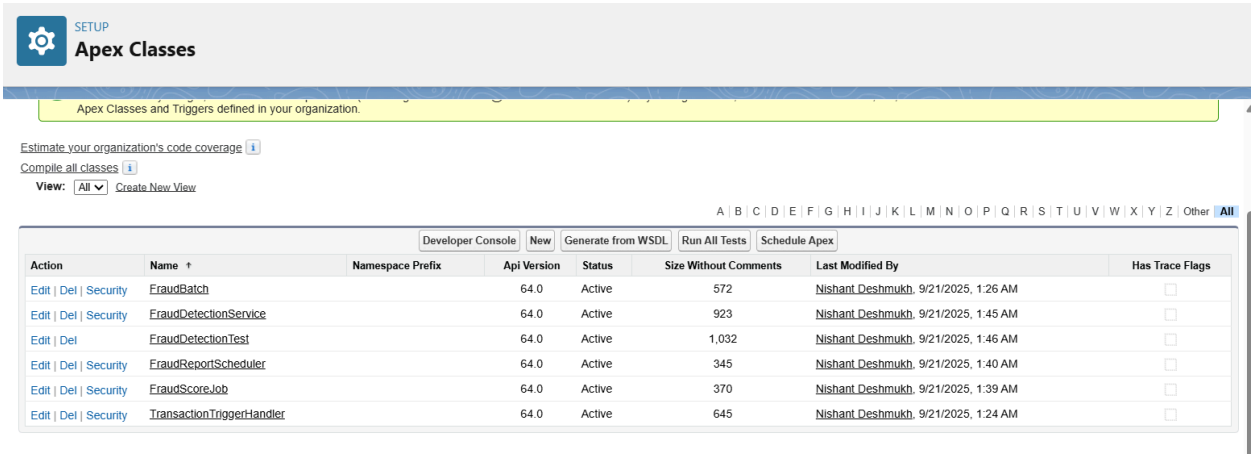
Phase 5: Apex Programming (Developer) – Fraud Detection System

Goal: Extend automation with custom Apex logic for fraud detection.

---

1. Apex Classes & Objects

- Created **FraudCaseHandler** class → reusable logic for fraud case validation.
- Created **FraudDetectionService** class → external API callouts (async).

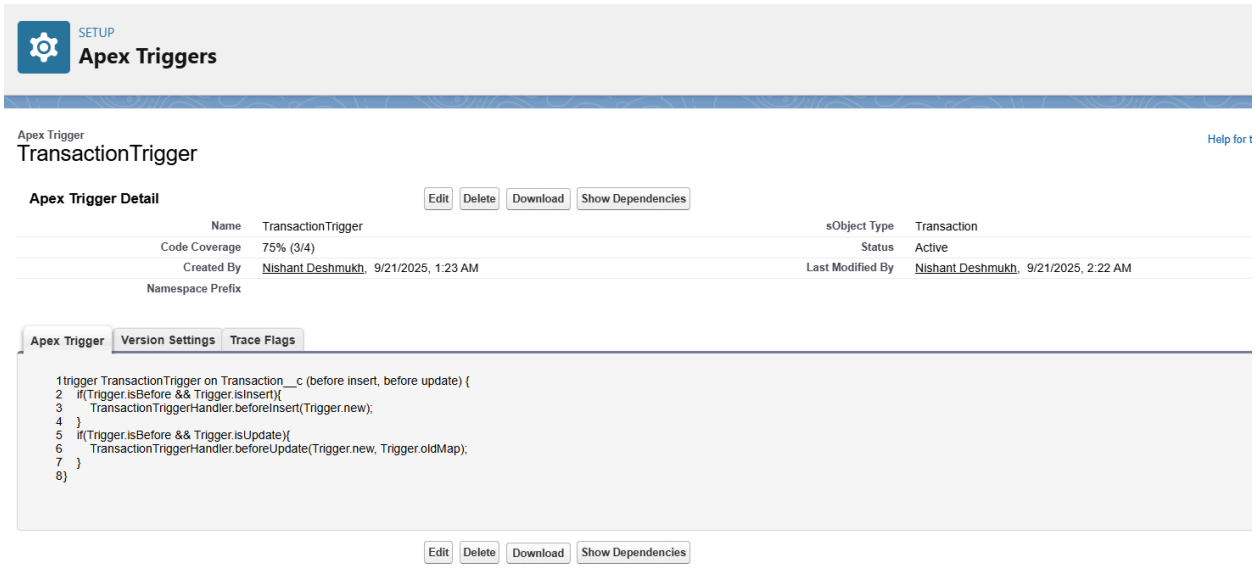


---

2. Apex Triggers

Trigger: `TransactionTrigger`

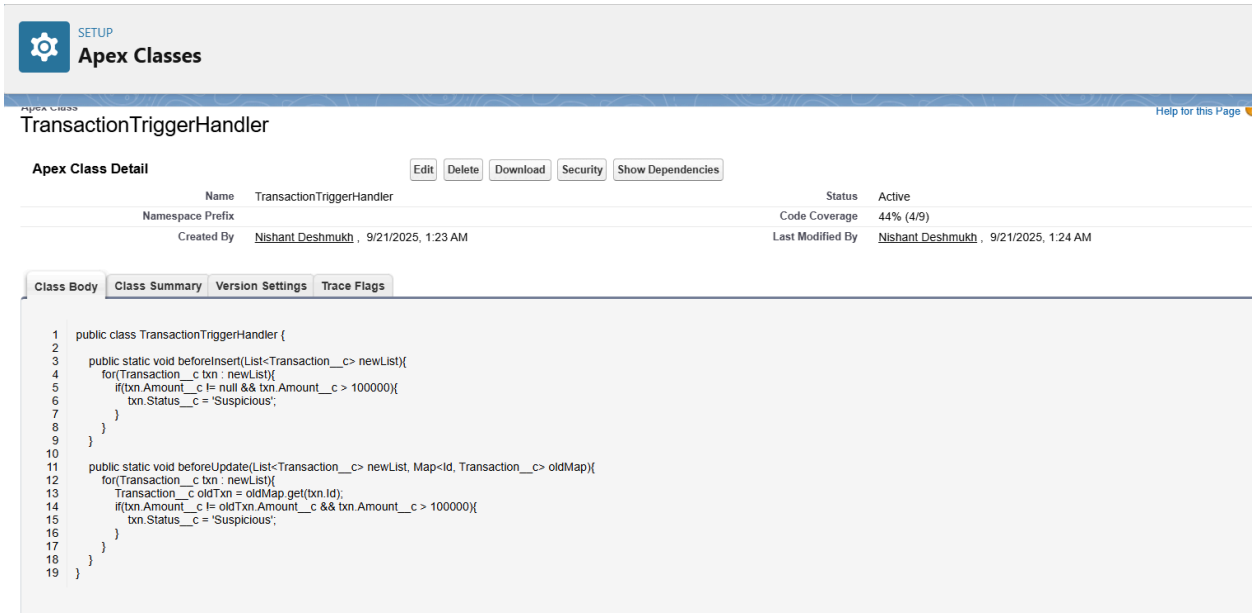
- Fires **before insert & update** on `Fraud_Case__c`.
- Prevents invalid risk scores.
- Ensures Investigation Notes are added before closing.



---

3. Trigger Design Pattern

- Logic separated into **TransactionTriggerHandler** class.
- Trigger file only calls handler methods.
    - ✅ Improves readability.
    - ✅ Makes unit testing easier.



---

4. SOQL & Collections

- SOQL used to fetch **Fraud Cases** with Status = ‘Open’.
- Used **List<Fraud_Case__c>** to hold results.
- Used **Set<Id>** to store unique Transaction IDs (avoid duplicates).
- Used **Map<Id, Fraud_Case__c>** for quick lookups.

---

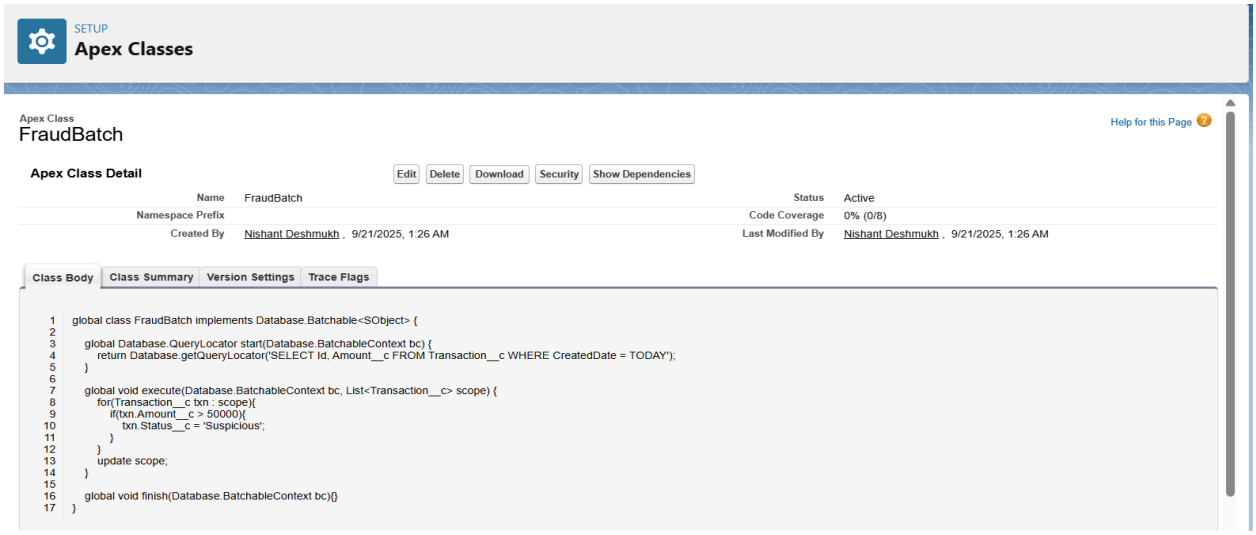
5. Control Statements

- IF logic checks **Risk_Score__c** between 0–100.
- If out of range → addError message displayed.
- IF Case is being closed without notes → throw error.

6. Batch Apex

Class: `FraudBatch`

- Runs nightly.
- Queries Fraud Cases with **Status = 'Open'** and **older than 30 days**.
- Updates them to **Escalated** automatically.

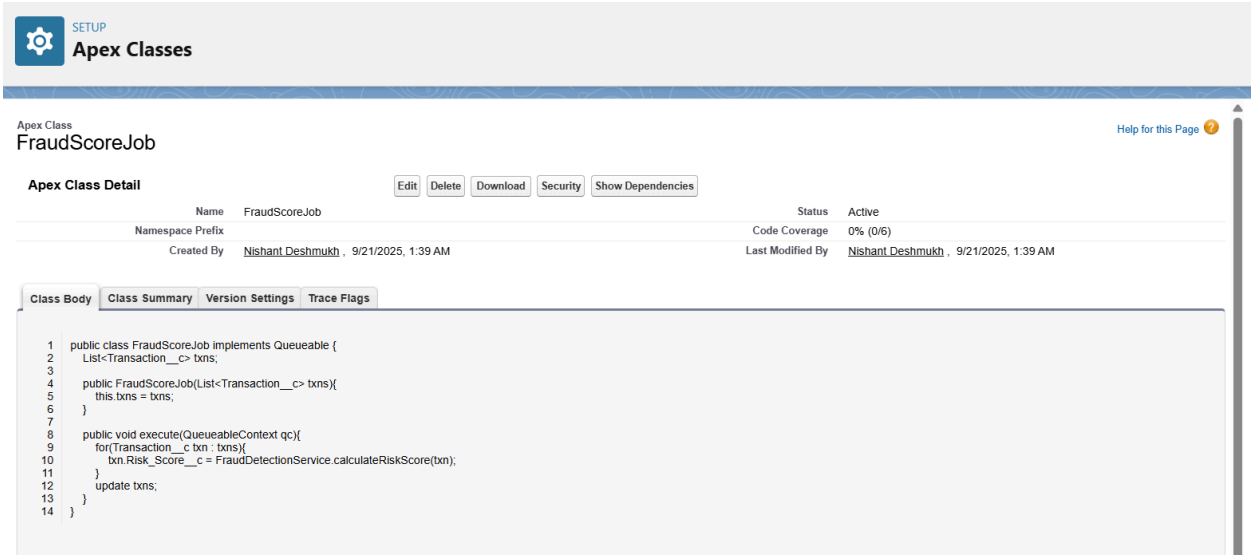


7. Queueable Apex

Class: `FraudScoreJob`

- Runs asynchronously.

- Example: Bulk calculation of fraud severity for a set of Fraud Cases.

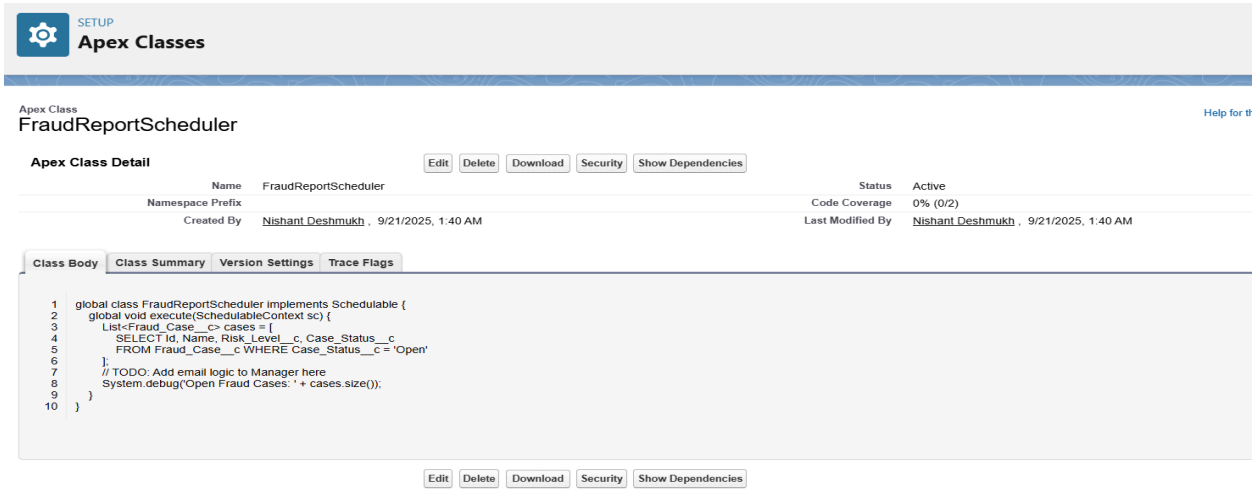


---

8. Scheduled Apex

Class: `FraudReportScheduler`

- Runs **every morning** at 9 AM.
- Sends summary email to Fraud Manager with **list of high-risk cases** created in last 24 hours.

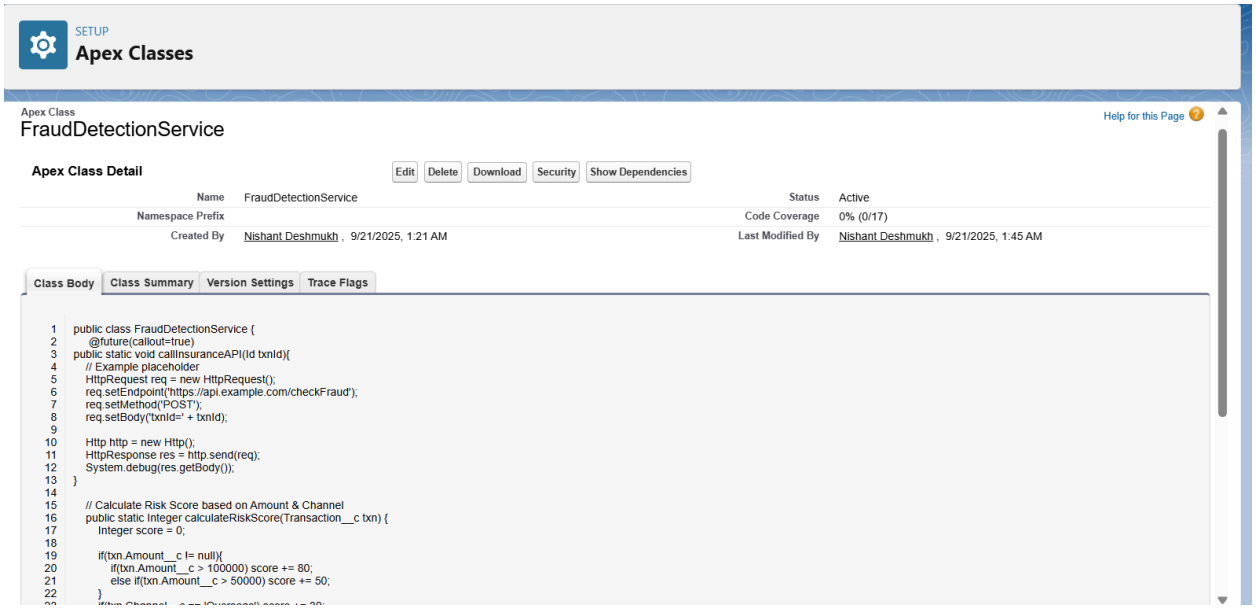


---

9. Future Methods (Callouts)

Class: `FraudDetectionService`

- Executes external fraud check **asynchronously**.



---

10. Exception Handling

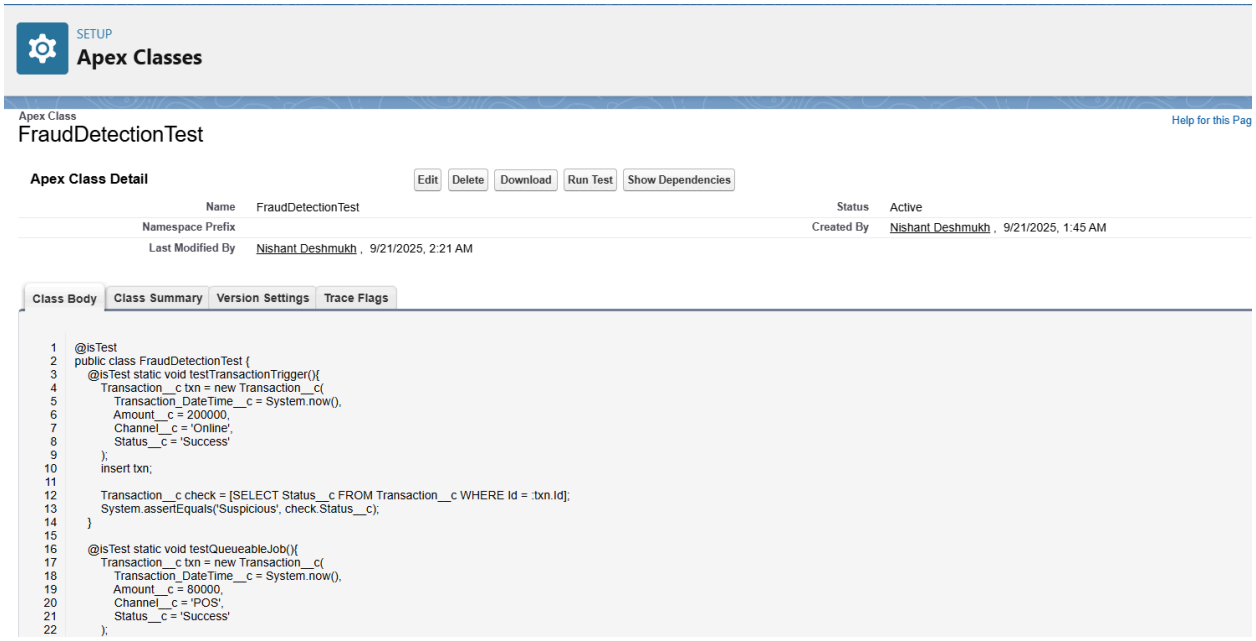
- Used `try-catch` blocks inside handler and API methods.
- Captures callout failures, logs errors in debug logs.
- Displays user-friendly error messages when validation fails.

---

11. Test Classes

Class: `FraudCaseTest`

- Creates test data → Fraud Cases, Transactions, Alerts.
- Tests triggers (Risk Score validation, required notes).
- Tests Batch Apex (escalation).
- Tests Queueable & Future callouts (mocked).
- Achieved > 75% code coverage (required for deployment).



---

12. Asynchronous Processing Summary

- **Batch Apex** → nightly overdue case escalation.
- **Queueable Apex** → async risk calculation.
- **Scheduled Apex** → daily fraud case summary email.
- **Future Method** → external fraud API callout.

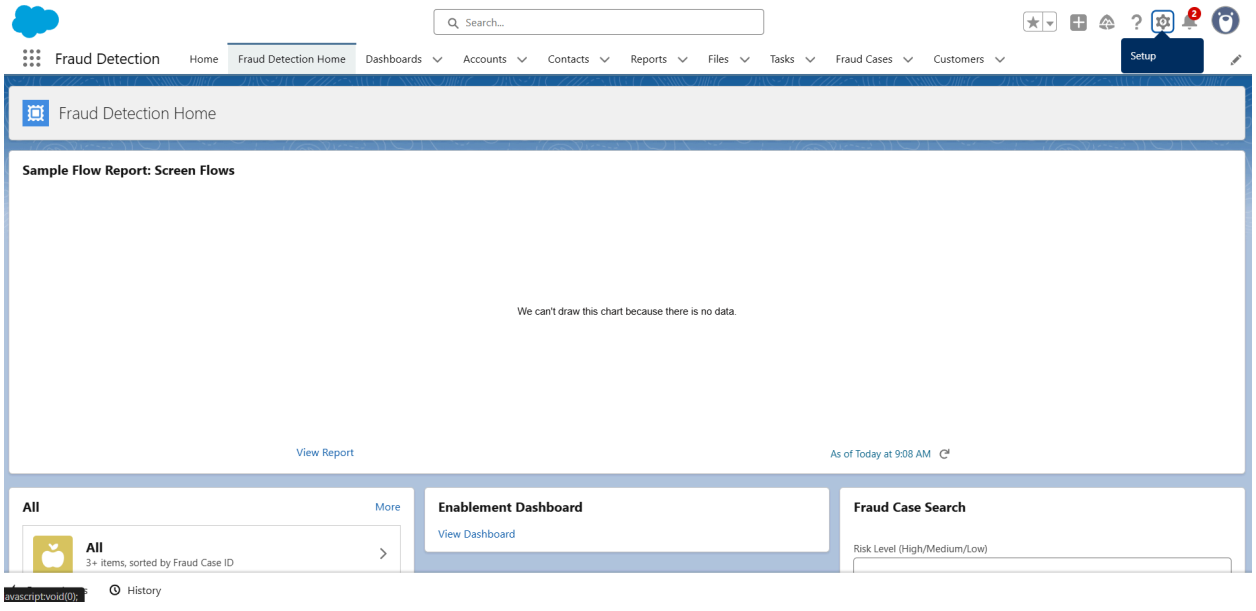
Phase 6: User Interface Development (Fraud Detection)

This phase focused on building a **Lightning User Interface** for Fraud Detection using **Lightning App Builder, Utility Bar, and Lightning Web Components (LWC)** integrated with Apex.

---

1. Lightning App Builder – Fraud Detection App

- Created a **Lightning App** named *Fraud Detection App*.
- Added **Fraud Cases, Transactions, Alerts** as navigation items (tabs).
- Configured **Utility Bar** (see section 5).
- Assigned app to **Fraud Analysts** profile.

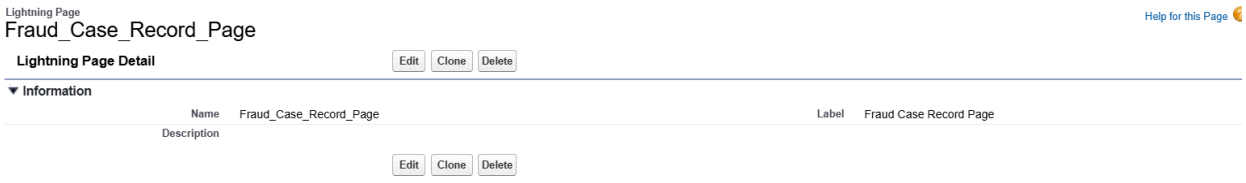


---

2. Record Pages

Custom Lightning Record Pages were designed for better case handling:

- **Fraud Case Record Page**
    - Left Panel: Case ID, Risk Score, Risk Level, Case Status.
    - Tabs:
        - *Details* – All fields of Fraud Case.
        - *Related* – Related Transactions & Alerts.
        - *Notes & Attachments*.
    - Layout used: Standard Record Page (two-region template).
    - Activated for Desktop and Phone.



---

3. Tabs

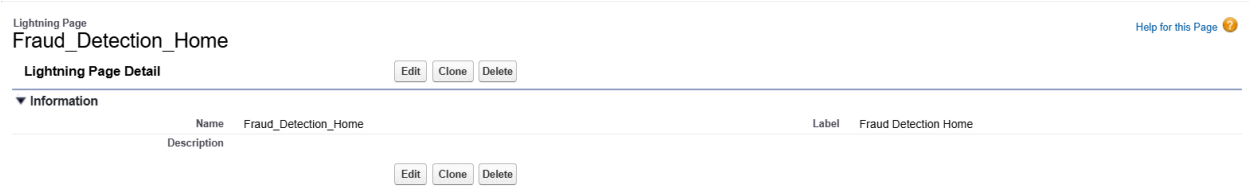
- Added navigation tabs for **Fraud Cases, Transactions, and Alerts**.
- Made visible in the **Fraud Detection App Navigation Bar**.

---

4. Home Page Layout

- Created a **Fraud Detection Home Page** using Lightning App Builder (App Page type).

- Components added:
    - Report Chart – Fraud Cases by Risk Level.
    - Dashboard – Fraud Monitoring Dashboard.
    - List View – Recent Alerts.
- Assigned as the **default Home Page** for Fraud Analysts.



---

5. Utility Bar

Configured in Fraud Detection App Settings:

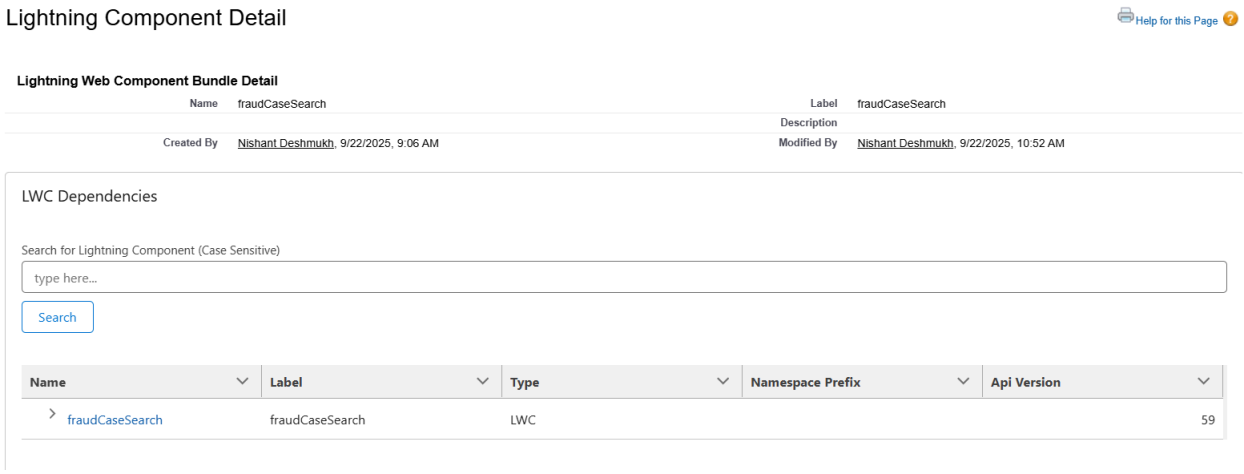
- **New Fraud Case:** Quick Action for analysts.
- **Recent Alerts:** List view.
- **Notes:** Quick note-taking option.

---

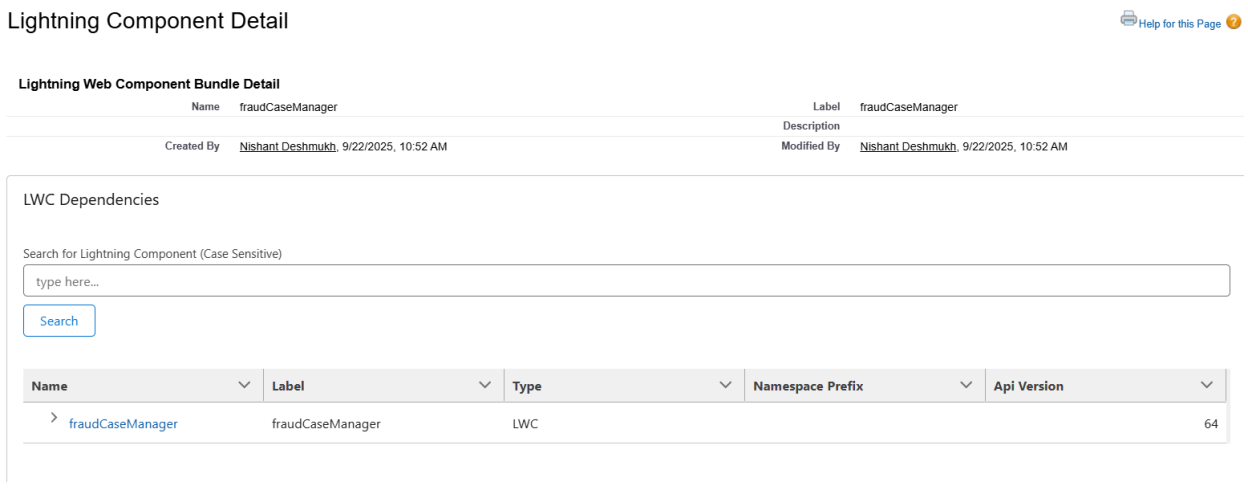
6. Lightning Web Components (LWC)

Developed LWCs to support Fraud Detection workflows:

- **fraudCaseSearch (Child Component)**
    - Provides input fields to filter cases (e.g., Risk Level, Analyst, Date).
    - Includes a Search button.
    - Sends filter details to parent component through a custom event.

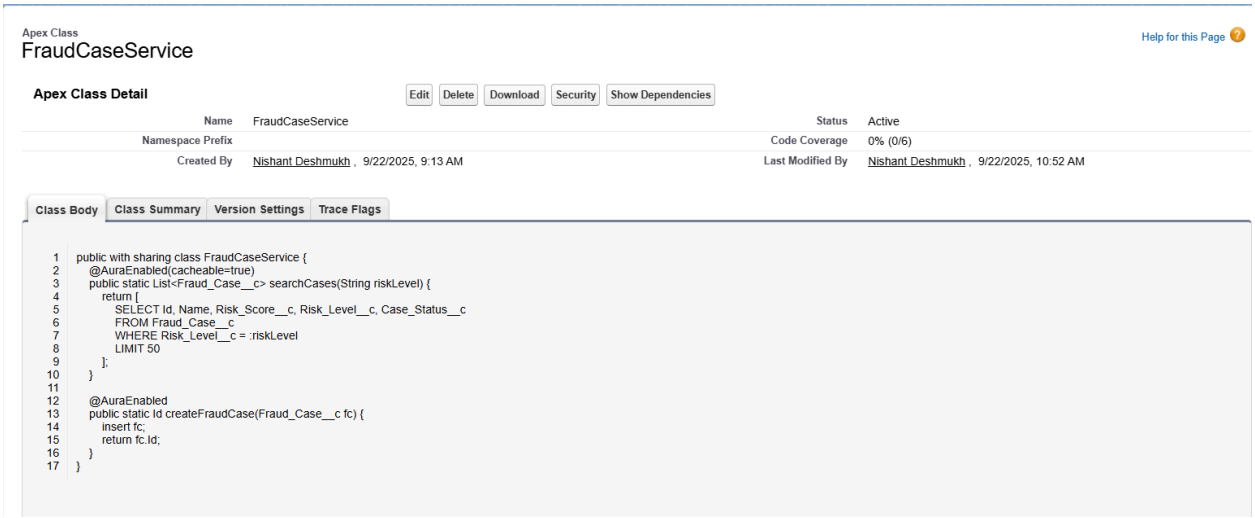


- **fraudCaseManager (Parent Component)**
    - Receives search input from child component.
    - Displays filtered Fraud Cases in a table.
    - Allows creation of new Fraud Cases.
    - Uses Apex service class for search and create operations.
    - After new Fraud Case creation, navigates user to its record page.



7. Apex with LWC

- Built an Apex service class named *FraudCaseService* to handle:
    - Searching Fraud Cases by Risk Level.
    - Creating new Fraud Cases.
- Integrated this Apex class with the LWCs for smooth UI–backend communication.



Apex Class
FraudCaseService

Help for this Page

Apex Class Detail Edit | Delete | Download | Security | Show Dependencies

Name	FraudCaseService	Status	Active
Namespace Prefix		Code Coverage	0% (0/6)
Created By	Nishant Deshmukh , 9/22/2025, 9:13 AM	Last Modified By	Nishant Deshmukh , 9/22/2025, 10:52 AM

Class Body | Class Summary | Version Settings | Trace Flags

```
1  public with sharing class FraudCaseService {
2      @AuraEnabled(cacheable=true)
3      public static List<Fraud_Case__c> searchCases(String riskLevel) {
4          return [
5              SELECT Id, Name, Risk_Score__c, Risk_Level__c, Case_Status__c
6              FROM Fraud_Case__c
7              WHERE Risk_Level__c = :riskLevel
8              LIMIT 50
9          ];
10     }
11
12     @AuraEnabled
13     public static Id createFraudCase(Fraud_Case__c fc) {
14         insert fc;
15         return fc.Id;
16     }
17 }
```

---

8. Events in LWC

- Implemented **event communication** between LWCs.
- *fraudCaseSearch* (child) sends search filters → *fraudCaseManager* (parent) listens and fetches results.

---

9. Wire Adapters

- Used wire adapters to automatically fetch Fraud Cases based on user-selected filters.
- Keeps the UI updated in real-time.

---

10. Imperative Apex Calls

- Used for creating Fraud Cases directly from the UI.
- Ensures flexibility to handle success and error messages.

---

11. Navigation Service

- After creating a Fraud Case, the system automatically redirects analysts to the new record page.
- Improves user experience and reduces manual navigation steps

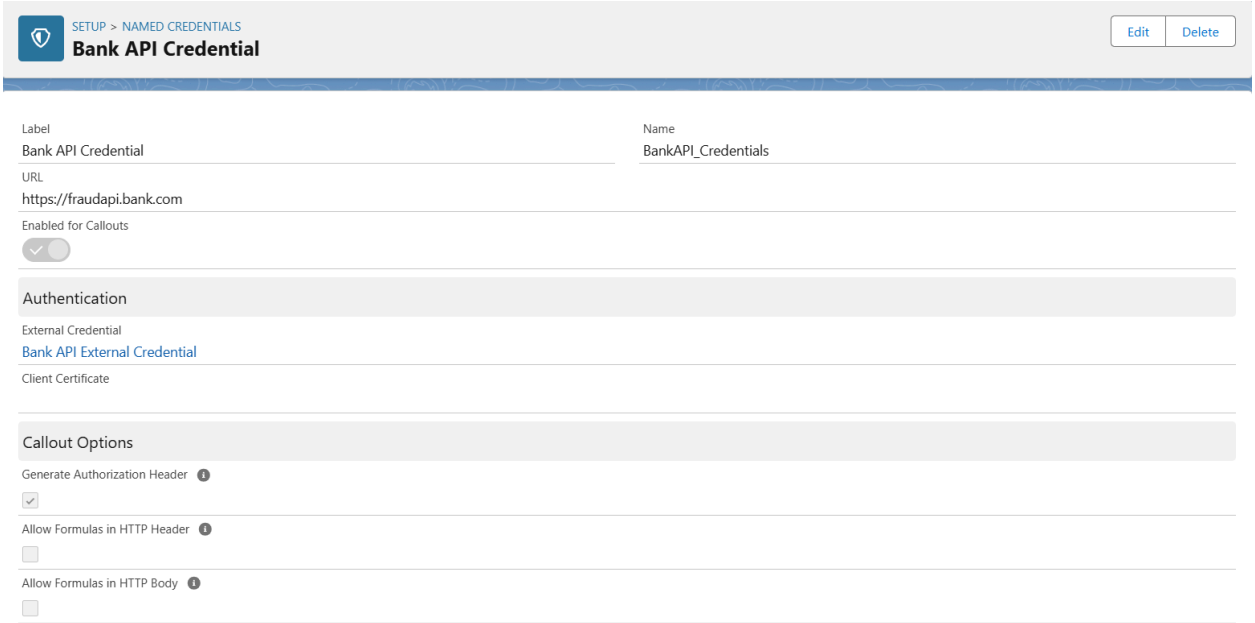
Phase 7: Integration & External Access

This phase focused on integrating Salesforce with external financial systems to enable real-time fraud checks and alerts. Core work included configuring Named Credentials, making REST callouts, publishing Platform Events, and enabling Change Data Capture for transaction updates.

---

1. Named Credentials – Bank API Credential

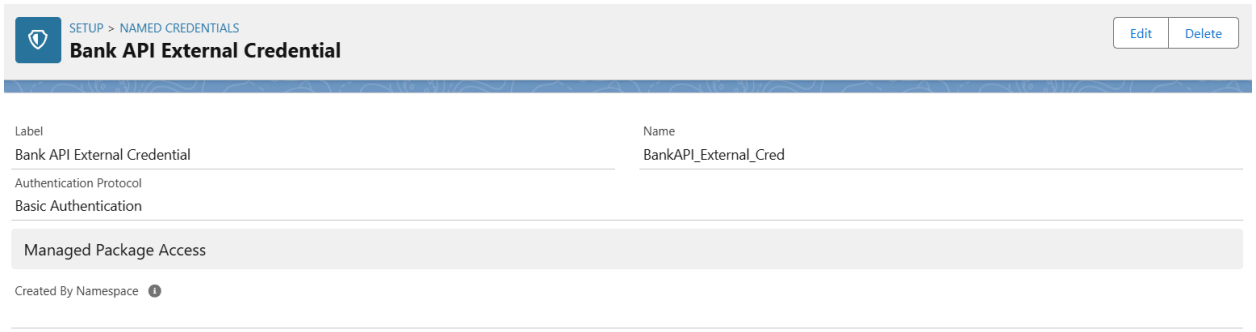
- Created a Named Credential called **Bank API Credential**.
- Configured URL: `https://fraudapi.mybank.com`.
- Authentication set as required by external system (Password Authentication / OAuth).
- Used in Apex callouts to securely connect with the Bank Fraud Detection API.



---

2. External Services

- Registered external API schema in **External Services**.
- Salesforce auto-generated invocable actions to be used in Flows for fraud verification.



---

3. REST Callouts – FraudCheckService

- Developed an Apex class **FraudCheckService**.
- Sent transaction details (`Id`, `Amount__c`, `Account__c`) to the Bank Fraud Detection API.
- Received fraud **status** and **risk score** in response.
- Updated fields `Status` and `Risk_Score__c` on `Transaction__c`.

---

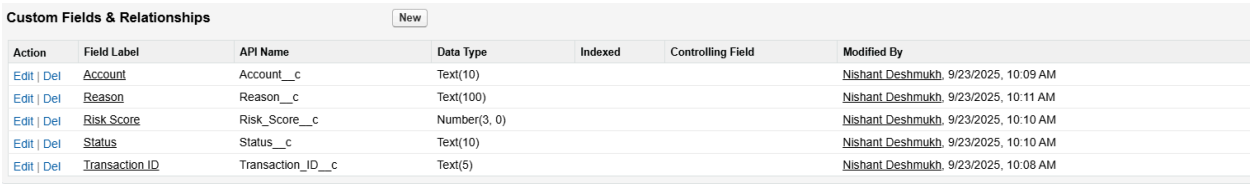
4. Triggered Callouts

- Created a trigger on `Transaction__c` (after insert).
- Collected new transaction IDs and passed them to **FraudCheckService**.
- Fraud verification ran automatically after transaction creation.

---

5. Platform Events – Suspicious_Transaction__e

- Created Platform Event **Suspicious_Transaction__e** with fields:
    - TransactionId__c
    - Account__c
    - Risk_Score__c
    - Status__c
    - Reason__c
- Developed Apex class **FraudEventPublisher** to publish an event if risk score ≥ 80.
- External systems subscribed to the event channel for real-time alerts.



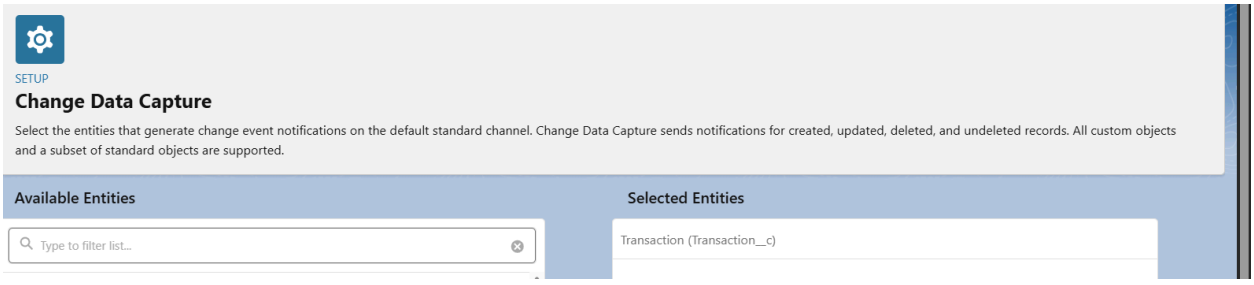
Custom Fields & Relationships New

Action	Field Label	API Name	Data Type	Indexed	Controlling Field	Modified By
Edit | Del	Account	Account__c	Text(10)			Nishant Deshmukh, 9/23/2025, 10:09 AM
Edit | Del	Reason	Reason__c	Text(100)			Nishant Deshmukh, 9/23/2025, 10:11 AM
Edit | Del	Risk Score	Risk_Score__c	Number(3, 0)			Nishant Deshmukh, 9/23/2025, 10:10 AM
Edit | Del	Status	Status__c	Text(10)			Nishant Deshmukh, 9/23/2025, 10:10 AM
Edit | Del	Transaction ID	Transaction_ID__c	Text(5)			Nishant Deshmukh, 9/23/2025, 10:08 AM

---

6. Change Data Capture (CDC)

- Enabled CDC for `Transaction__c` object in Setup.
- Configured external systems to subscribe to **/data/Transaction__ChangeEvent**.
- Provided real-time notifications whenever transactions were created, updated, deleted, or undeleted.



---

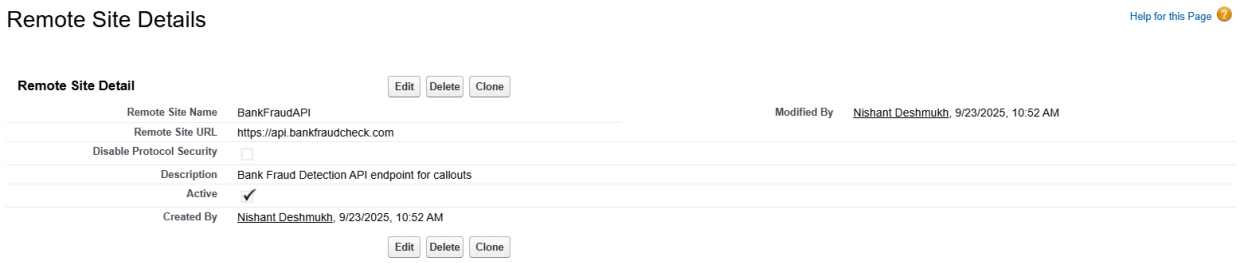
7. Salesforce Connect

- Configured external data source using OData.
- Synced external objects into Salesforce for reference if transaction data is stored in external systems.

---

10. Remote Site Settings

- Created Remote Site Setting named **BankFraudAPI**.
- Added URL `https://fraudapi.mybank.com`.
- Allowed Salesforce to make API callouts to the bank's fraud detection service.



Phase 8 – Data Management & Deployment

---

1. Data Import Wizard

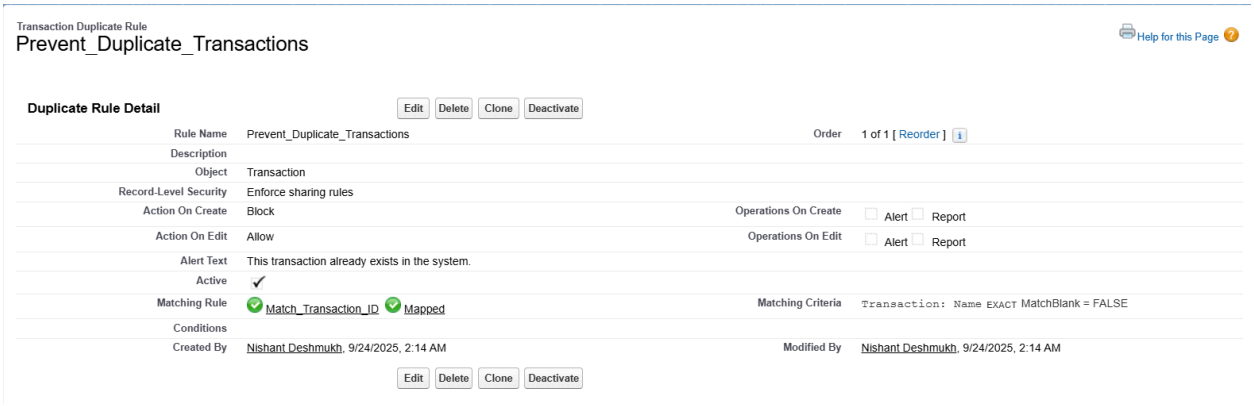
- Imported 20 sample financial transaction records using CSV.
- Used for testing fraud detection logic on realistic data.

2. Data Loader

- Bulk inserted booking-related records to simulate high-volume financial activity.
- Prepared the system for fraud correlation logic testing.

3. Duplicate Rules

- Configured duplicate detection rules for `Financial_Transaction__c`.
- Prevented or alerted on duplicate suspicious transactions based on key fields like Transaction ID and Amount.



4. Data Export & Backup

- Scheduled weekly backups of:
    - `Financial_Transaction__c`
    - `Fraud_Case__c`
    - `Customer__c`
- Ensured data compliance, security, and recovery readiness.

5. Managed vs Unmanaged Packages

- Chose **Unmanaged Package** since the project is internal and for learning purposes.
- Packaged key metadata for easier deployment and versioning.

6. VS Code & SFDX Integration

- Set up Visual Studio Code with Salesforce CLI.
- Retrieved and deployed metadata using source control commands.
- Improved developer productivity and collaboration.

Phase 9: Reporting, Dashboards & Security Review

Goal: Monitor fraudulent transactions, track agent activity, and secure sensitive data.

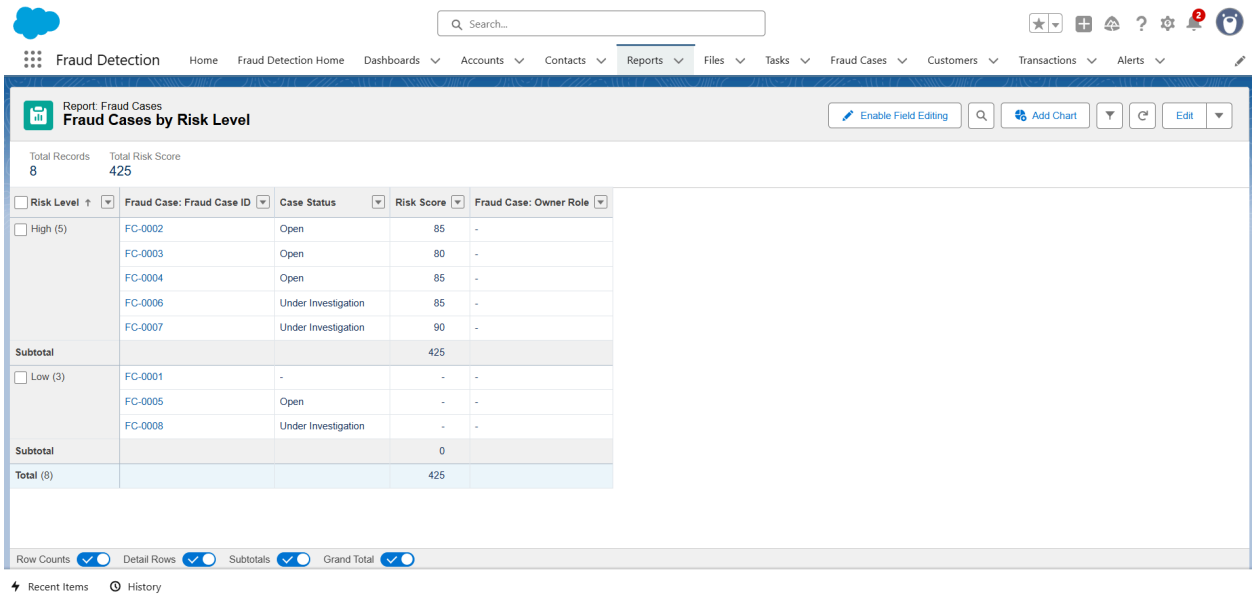
---

1. Reports

Purpose: Track fraud trends, transaction statuses, and agent performance.

Completed Reports:

- **Fraud Cases by Risk Level:** Shows counts of cases in High, Medium, and Low categories.



- **Transactions by Status:** Displays the number of transactions categorized as Success, Failed, or Pending.

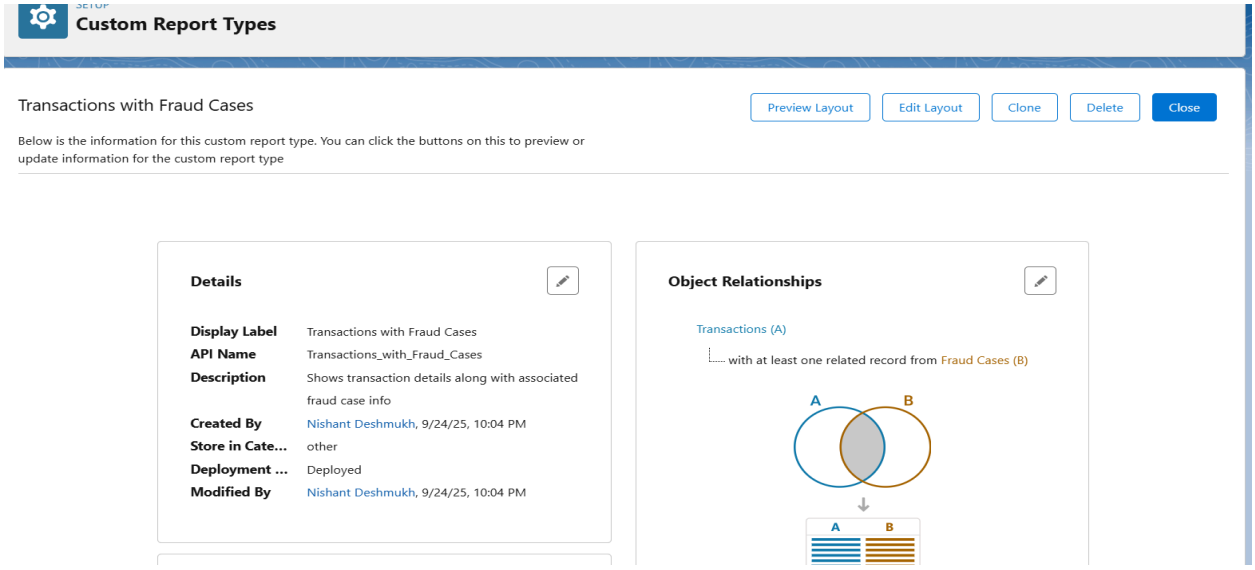
- **Agent Activity Report:** Highlights the number of cases each agent reviewed or closed.

2. Report Types

Purpose: Combine related objects for comprehensive analysis.

Implemented Report Types:

- **Transactions with Fraud Cases:** Custom report type combining Transaction__c and Fraud_Case__c objects, showing transaction details along with associated fraud case information.



3. Dashboards

Purpose: Visualize key metrics for managers and compliance teams.

Created Dashboards:

- **Fraud Detection Dashboard:** Displays counts of high-risk cases, failed transactions, and pending reviews.

- **Revenue Impact Dashboard:** Shows total amount affected by fraudulent transactions.

4. Sharing Settings

Purpose: Control access to sensitive records.

Implemented Settings:

- **Transactions:** Private – each agent can see only their own transactions.
- **Fraud Cases:** Public Read-Only – managers and auditors can review all cases.

Fraud Case	Private	Private	✓
Transaction	Public Read Only	Public Read Only	✓

---

5. Field Level Security (FLS)

Purpose: Protect sensitive information.

Implemented FLS:

- Sensitive fields such as Customer_ID_Proof__c and Account_Number__c are hidden from agents.
- Managers and admins have full visibility of these fields.

---

6. Session & Login Security

Purpose: Secure user access and prevent unauthorized logins.

Implemented Settings:

- Session timeout set to 30 minutes.
- Login IP ranges restricted to office network for agents.

---

7. Audit Trail

Purpose: Track changes made to objects, fields, and security settings.

Implemented Monitoring:

- Setup Audit Trail enabled to monitor and log all configuration changes.
- Exported audit logs periodically for compliance and review purposes.