

**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.

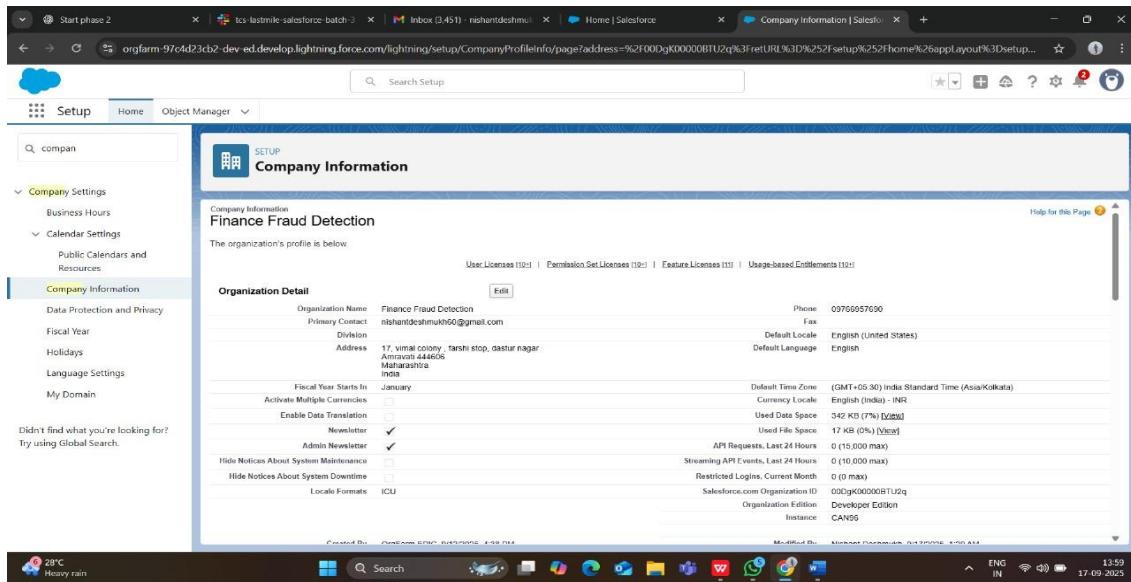
# Phase 2: Org Setup & Configuration

## 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 ✓

The screenshot shows the 'Business Hours Detail' section of the Salesforce setup. A table lists business hours from Sunday to Saturday, with most days set to 'No Hours'. The 'Time Zone' is set to '(GMT+05:30) India Standard Time (Asia/Kolkata)'. A 'Default Business Hours' button is present. Below the table, it says 'Active' and shows 'Created By: Nishant.Deshmukh 9/16/2025, 5:39 AM' and 'Last Modified By: Nishant.Deshmukh 9/16/2025, 5:55 AM'. A 'Holidays' section lists four specific dates: Christmas (12/25/2025 All Day), Gandhi Jayanti (10/2/2025 All Day), Independence Day (8/15/2025 All Day), and Republic Day (1/26/2025 All Day). The sidebar on the left shows 'Company Settings' with 'Business Hours' selected.

## 4. Fiscal Year Settings

- Used Standard Fiscal Year (Jan–Dec).

The screenshot shows the 'Organization Fiscal Year Edit: Finance Fraud Detection' page. It asks to specify the fiscal year type for the organization. A note states: 'Your organization can change the fiscal year start month, and specify whether the fiscal year name is set to the starting or ending year. For example, if your fiscal year starts in April 2025 and ends in March 2026, your Fiscal Year setting can be either 2025 or 2026.' A warning message in a yellow box says: 'Changing the fiscal year shifts fiscal periods and impacts opportunities and forecasts across your organization. If your forecast periods are set to quarterly, adjusting the fiscal year start month will erase existing forecast adjustments and quotas. Consider exporting a data backup before implementing this change.' A 'Change Fiscal Year Period' section allows selecting 'Standard Fiscal Year' (selected) or 'Custom Fiscal Year'. It shows 'Fiscal Year Start Month: January' and 'Fiscal Year is Based On: The ending month'. Buttons for 'Save' and 'Cancel' are at the bottom. The sidebar on the left shows 'Company Settings' with 'Fiscal Year' selected.

## 5. User Setup & Licenses

Created Users:

- Fraud Manager
- Fraud Analyst
- Compliance Officer

The screenshot shows the Salesforce Setup interface with the URL <https://orgfarm-97cf0d23cb2-dev-ed.lightning.force.com/lightning/setup/ManageUsers/page?address=%CF005gK000007p6t%3fmoredirect%3D1%26UserEntityOverride%3D1>. The page title is "Users". The user details for "Ajit Chauhan" are displayed, including Name (Ajit Chauhan), Alias (achauhan), Email (rishabhchauhan60@gmail.com [Verify]), Username (ajit.chauhan60@gmail.com), Nickname (achauhan), Title (Marketing Manager), Company (Company), Department (Division), Address (Address), Time Zone (GMT+05:30) India Standard Time (Asia/Kolkata), Locale (English (United States)), Language (English), Delegated Approver (Manager), and various user roles and settings like Marketing User, Office User, Knowledge User, etc.

The screenshot shows the Salesforce Setup interface with the URL <https://orgfarm-97cf0d23cb2-dev-ed.lightning.force.com/lightning/setup/ManageUsers/page?address=%CF005gK000007p6t%3fmoredirect%3D1%26UserEntityOverride%3D1>. The page title is "Users". The user details for "Sunil Kumar" are displayed, including Name (Sunil Kumar), Alias (skumar), Email (rishabhchauhan60@gmail.com [Verify]), Username (sunilkumar60@gmail.com), Nickname (skumar), Title (Analyst), Company (Company), Department (Division), Address (Address), Time Zone (GMT+05:30) India Standard Time (Asia/Kolkata), Locale (English (United States)), Language (English), Delegated Approver (Manager), and various user roles and settings like Marketing User, Office User, Knowledge User, etc.

The screenshot shows the Salesforce Setup interface with the URL <https://orgfarm-97cf0d23cb2-dev-ed.lightning.force.com/lightning/setup/ManageUsers/page?address=%CF005gK000007p6t%3fmoredirect%3D1%26UserEntityOverride%3D1>. The page title is "Users". The user details for "Parwesh Jaiswal" are displayed, including Name (Parwesh Jaiswal), Alias (parwesh), Email (rishabhchauhan60@gmail.com [Verify]), Username (parweshjaiswal60@gmail.com), Nickname (parwesh), Title (Fraud Analyst), Company (Company), Department (Division), Address (Address), Time Zone (GMT+05:30) India Standard Time (Asia/Kolkata), Locale (English (United States)), Language (English), Delegated Approver (Manager), and various user roles and settings like Marketing User, Office User, Knowledge User, etc.

The screenshot shows the Salesforce Setup interface with the URL <https://orgfarm-97cf0d23cb2-dev-ed.lightning.force.com/lightning/setup/ManageUsers/page?address=%CF005gK000007p6t%3fmoredirect%3D1%26UserEntityOverride%3D1>. The page title is "Users". The user details for "Parwesh Jaiswal" are displayed, including Name (Parwesh Jaiswal), Alias (parwesh), Email (rishabhchauhan60@gmail.com [Verify]), Username (parweshjaiswal60@gmail.com), Nickname (parwesh), Title (Fraud Analyst), Company (Company), Department (Division), Address (Address), Time Zone (GMT+05:30) India Standard Time (Asia/Kolkata), Locale (English (United States)), Language (English), Delegated Approver (Manager), and various user roles and settings like Marketing User, Office User, Knowledge User, etc.

## 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).

The screenshot shows the Salesforce Setup interface under the Profiles section. The left sidebar lists various categories like Hyperforce Assistant, Users, Data, Feature Settings, and Products. The main content area displays the 'Fraud Manager Profile' details. The profile name is 'Fraud Manager Profile', user license is 'Salesforce', and it is a 'Custom Profile'. The 'Profile Detail' section includes fields for Name, User License, Description, Created By, and Modified By. The 'Page Layouts' section shows standard object layouts for Global, Email Application, Home Page Layout, Account, Alternative Payment Method, and Appointment Invitation. It also lists location group assignments for Marketing, Object Milestones, Operating Hours, and Opportunity. The bottom of the page shows a toolbar with Edit, Clone, Delete, and View Users buttons.

This screenshot shows the same Salesforce Setup interface but for the 'Fraud Analyst Profile'. The profile details are identical to the Fraud Manager profile. The 'Page Layouts' section shows standard object layouts for Global, Email Application, Home Page Layout, Account, Alternative Payment Method, and Appointment Invitation. It also lists location group assignments for Marketing, Object Milestones, Operating Hours, and Opportunity. The bottom of the page shows a toolbar with Edit, Clone, Delete, and View Users buttons.

The screenshot shows the Salesforce Setup interface under the 'Profiles' section. A profile named 'Compliance Profile' is selected. The page lists various permissions such as 'Lead Edit Access', 'Enabled Apex Class Access', etc. It also shows 'Page Layouts' for different objects like 'Global', 'Email Application', 'Home Page Layout', and 'Object'. A 'Custom Profile' checkbox is checked.

## 7. Roles

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

The screenshot shows the Salesforce Setup interface under the 'Roles' section. It displays a hierarchical tree of roles. At the top is 'Finance Fraud Detection', which branches into 'CEO', 'CFO', 'Compliance Officer', 'COO', 'Fraud Manager', 'Fraud Analyst', 'SVP, Customer Service & Support', 'SVP, Human Resources', and 'SVP, Sales & Marketing'. Each role has edit, delete, and assign buttons next to it.

## 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.**

Sharing Settings			
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	✓
Fraud Case	Private	Private	✓
Transaction	Private	Private	✓

## 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.

Day	Start Time	End Time
Sunday	All Day	All Day
Monday	8:30 PM PDT	5:30 AM PDT
Tuesday	8:30 PM PDT	5:30 AM PDT
Wednesday	8:30 PM PDT	5:30 AM PDT
Thursday	8:30 PM PDT	5:30 AM PDT
Friday	8:30 PM PDT	5:30 AM PDT
Saturday	All Day	All Day

## 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

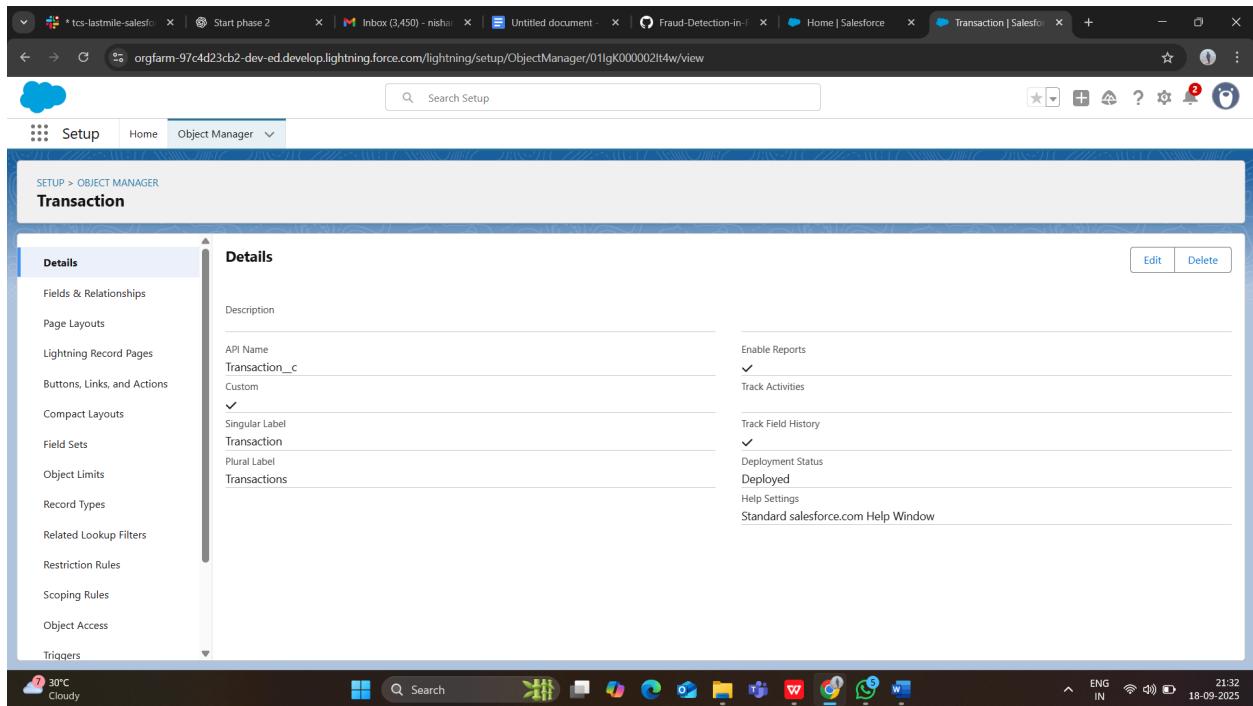
## 1. Standard & Custom Objects

- **Standard:**

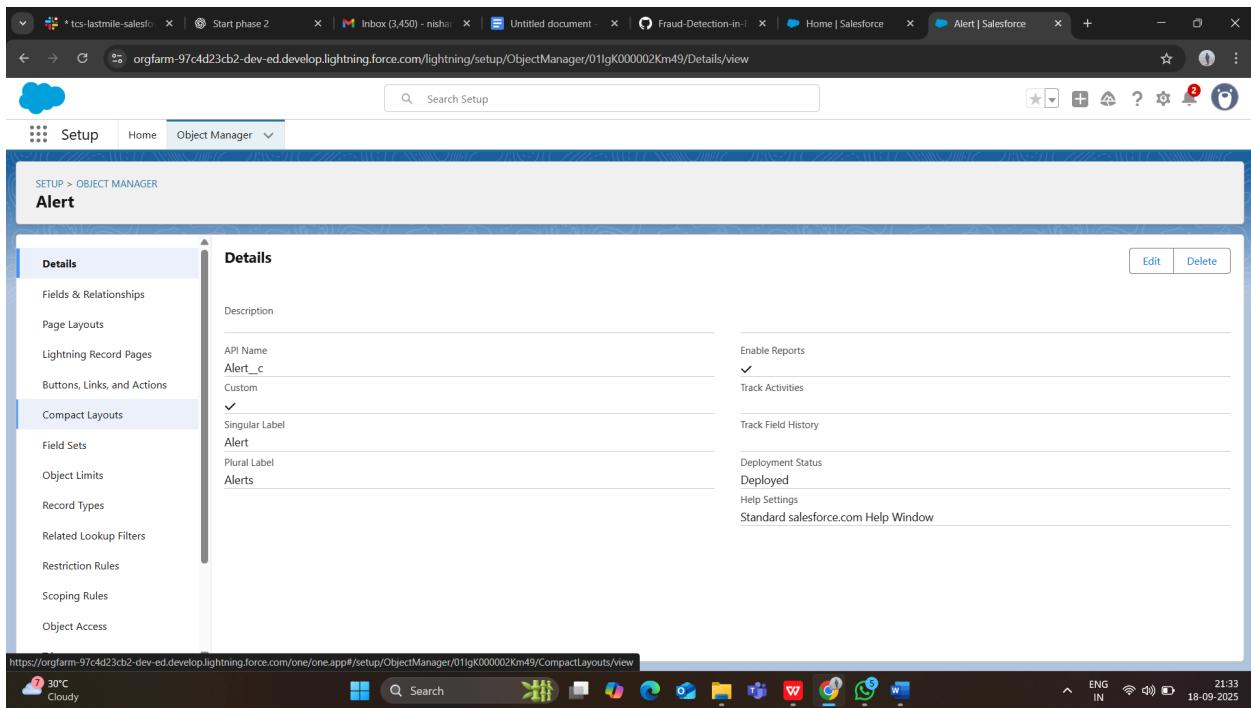
- **Account** (represents customer or business)
- **Contact** (customer representatives).

- **Custom:**

- **Transaction** (each financial activity).



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

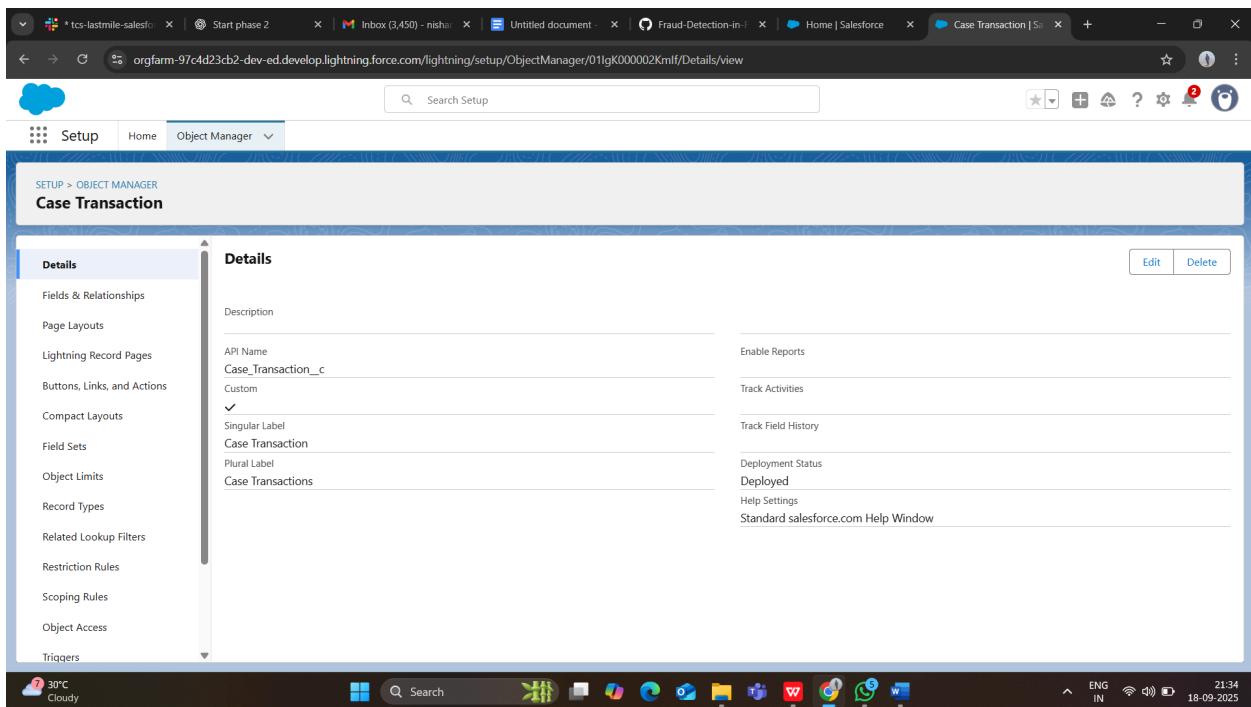


The screenshot shows the Salesforce Object Manager interface for creating a new object named 'Alert'. The left sidebar lists various configuration options: Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts (selected), Field Sets, Object Limits, Record Types, Related Lookup Filters, Restriction Rules, Scoping Rules, and Object Access. The main 'Details' pane shows the following fields:

- Description
- API Name: Alert\_c
- Custom: ✓
- Singular Label: Alert
- Plural Label: Alerts
- Enable Reports: ✓
- Track Activities
- Track Field History
- Deployment Status: Deployed
- Help Settings: Standard salesforce.com Help Window

The status bar at the bottom indicates it's 30°C Cloudy, the time is 21:33, and the date is 18-09-2025.

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



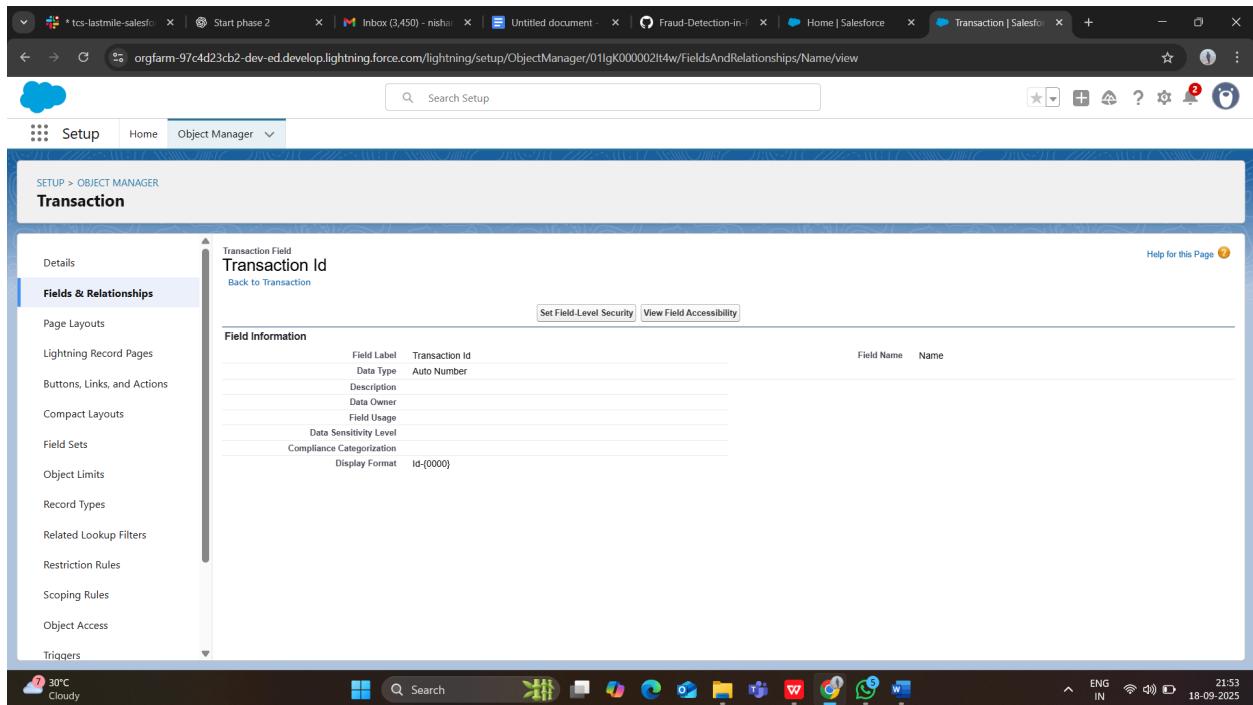
The screenshot shows the Salesforce Object Manager interface for creating a new object named 'Case Transaction'. The left sidebar lists various configuration options: Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts (selected), Field Sets, Object Limits, Record Types, Related Lookup Filters, Restriction Rules, Scoping Rules, Object Access, and Triggers. The main 'Details' pane shows the following fields:

- Description
- API Name: Case\_Transaction\_\_c
- Custom: ✓
- Singular Label: Case Transaction
- Plural Label: Case Transactions
- Enable Reports
- Track Activities
- Track Field History
- Deployment Status: Deployed
- Help Settings: Standard salesforce.com Help Window

The status bar at the bottom indicates it's 30°C Cloudy, the time is 21:34, and the date is 18-09-2025.

## 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).

The screenshot shows the Salesforce Setup Object Manager. The left sidebar is titled 'FIELDS & RELATIONSHIPS' under 'Alert'. The main content area displays the 'Alert ID' field information, which is an Auto Number field labeled 'Alert ID'. The page includes standard Salesforce navigation buttons like 'Set Field-Level Security' and 'View Field Accessibility'.

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

The screenshot shows the Salesforce Setup Object Manager. The left sidebar is titled 'FIELDS & RELATIONSHIPS' under 'Case Transaction'. The main content area displays the 'Transaction' custom field definition, which is a Master-Detail field. It has an API name of 'Transaction\_\_c' and is related to the 'Case' object. The 'Master-Detail Options' section shows it is related to 'Case 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.

The screenshot shows the Salesforce Setup interface for the Object Manager. The left sidebar has a 'Page Layouts' section selected under the 'Object Manager' category. The main content area displays a table titled 'Page Layouts' with one item: 'Transaction Layout'. The table includes columns for 'PAGE LAYOUT NAME', 'CREATED BY', and 'MODIFIED BY'. Both 'Nishant Deshmukh' are listed under their respective columns. The top navigation bar shows the URL as 'orgfarm-97c4d23cb2-dev-ed.lightning.force.com/lightning/setup/ObjectManager/01lgK0000002lt4w/PageLayouts/view'.

PAGE LAYOUT NAME	CREATED BY	MODIFIED BY
Transaction Layout	Nishant Deshmukh, 9/16/2025, 8:15 AM	Nishant Deshmukh, 9/18/2025, 4:04 AM

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

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

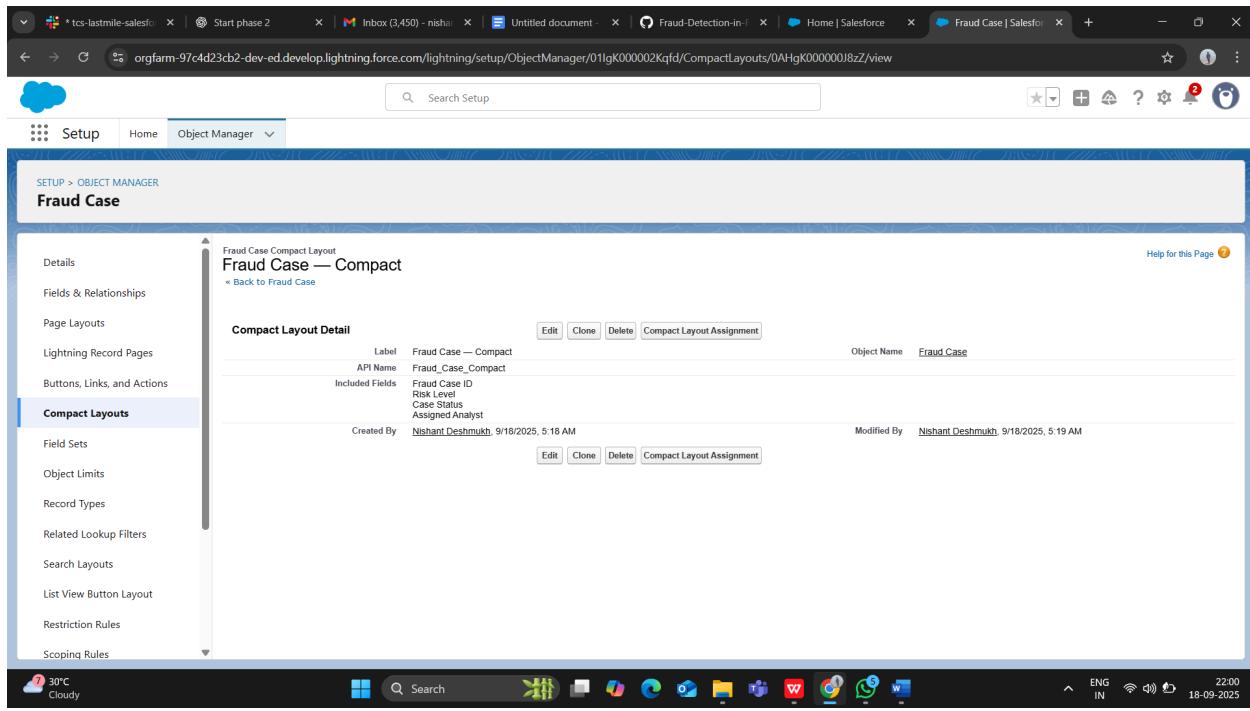
- Page Layouts** tab is selected in the sidebar.
- Page Layouts** section header: 1 Items, Sorted by Page Layout Name.
- Alert Layout** listed in the table:
 

PAGE LAYOUT NAME	CREATED BY	MODIFIED BY
Alert Layout	Nishant Deshmukh, 9/17/2025, 7:33 AM	Nishant Deshmukh, 9/18/2025, 9:27 AM

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

## 5. Compact Layouts

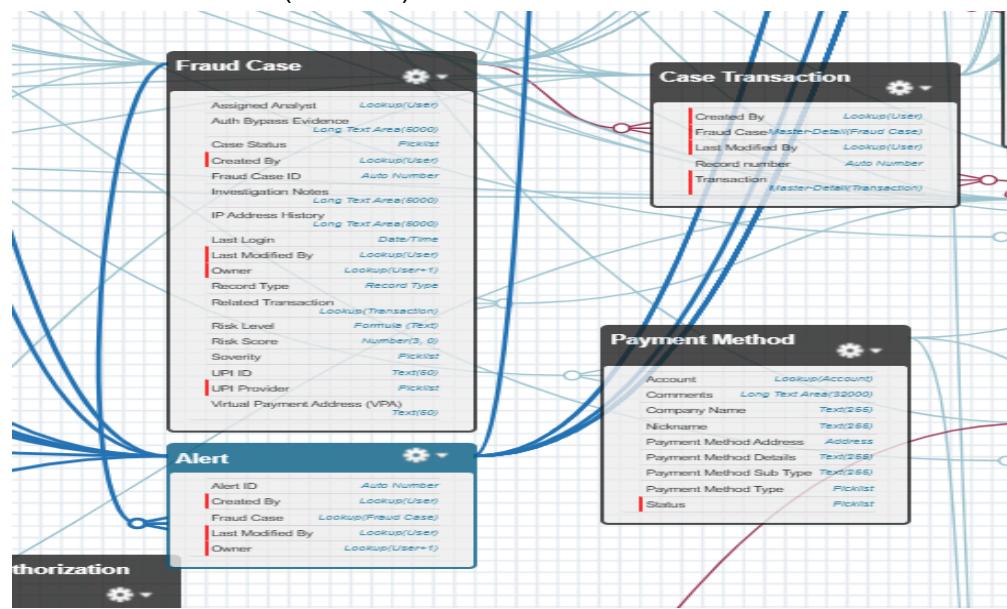
- **Mobile (Transaction)**: Amount, Date, Status.
- **Mobile (Alert)**: Alert Type, Risk Score, Status.



## 6. Schema Builder

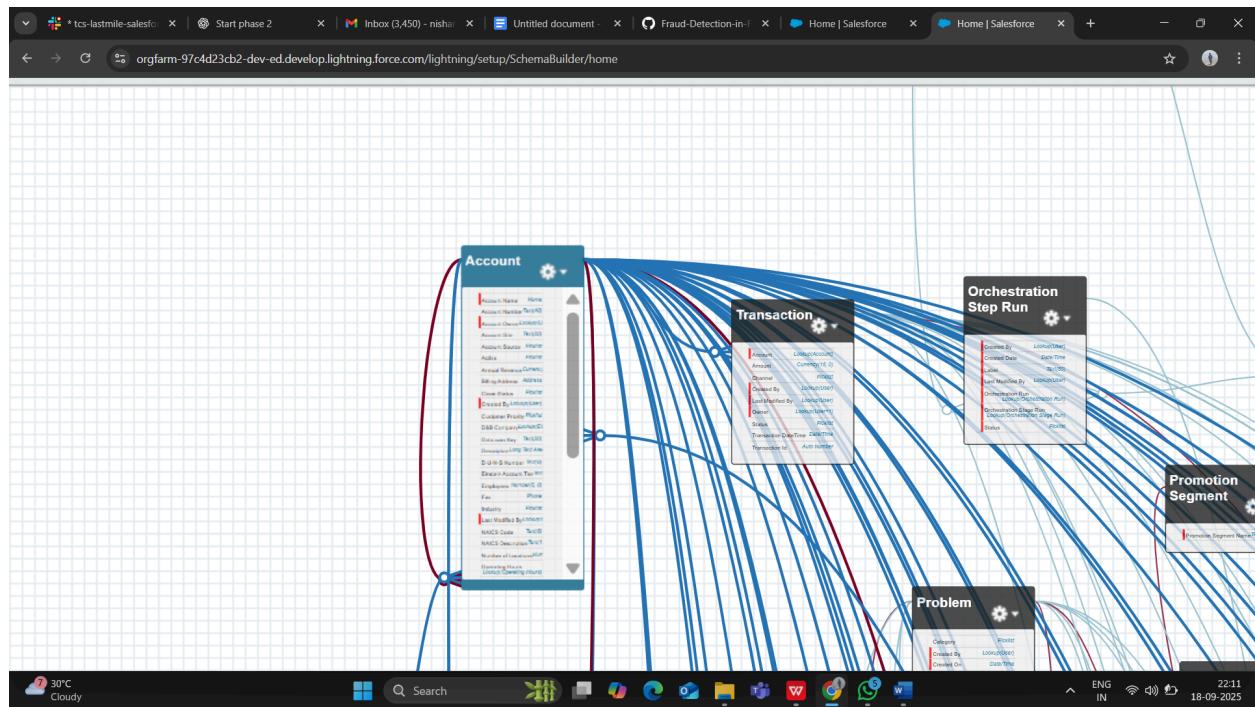
- Use **Schema Builder** to visually design:
  - Account → Transaction (Lookup).
  - Transaction → Alert (Lookup).

- 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

Fraud Case Validation Rule Help for this Page 

Back to Fraud Case

Validation Rule Detail		Edit	Clone
Rule Name	VR_Notes_Required_On_Closed	Active	<input checked="" type="checkbox"/>
Error Condition Formula	AND([ISPICKVAL(Case_Status__c, "Closed"), ISBLANK(Investigation_Notes__c)])		
Error Message	Please add Investigation Notes before closing the case	Error Location	Investigation Notes
Description			
Created By	Nishant Deshmukh, 9/19/2025, 12:10 AM	Modified By	Nishant Deshmukh, 9/19/2025, 12:10 AM
		Edit	Clone

- **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

Object Manager ▾

### Fraud Case Validation Rule

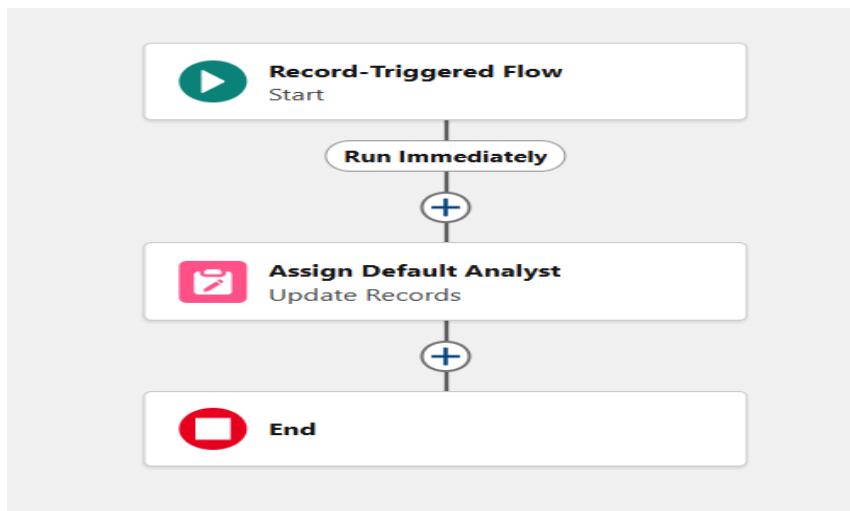
[Back to Fraud Case](#)

Validation Rule Detail		Active	
Rule Name	VR_RiskScore_Range		
Error Condition Formula	OR(Risk_Score__c < 0, Risk_Score__c > 100)		
Error Message	Risk Score must be between 0 and 100.	Error Location	Risk Score
Description	Risk Score must be between 0 and 100.	Modified By	<a href="#">Nishant Deshmukh</a> , 9/19/2025, 12:09 AM
Created By	<a href="#">Nishant Deshmukh</a> , 9/19/2025, 12:09 AM		
	<a href="#">Edit</a>	<a href="#">Clone</a>	

## 2. Record-Triggered Flows

### A) Auto-Assign Analyst

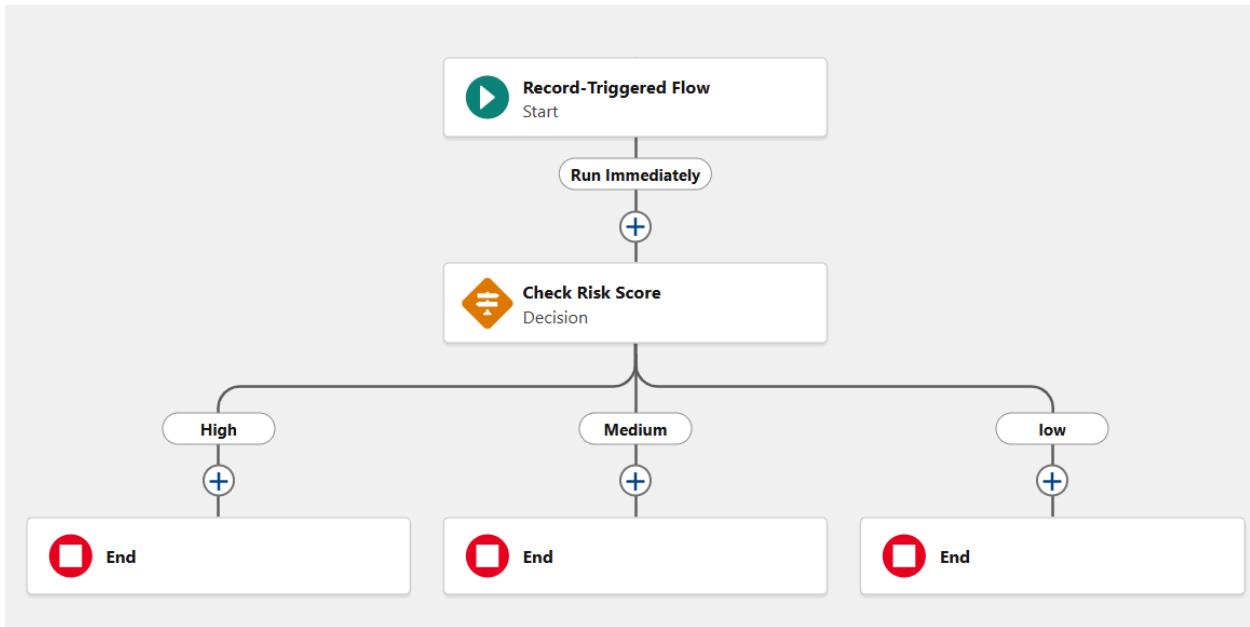
- **Object:** Fraud Case
- **Trigger:** When record is created
- **Condition:** Assigned\_Analyst\_\_c IS NULL
- 
- **Action:** Update Records → Assigned\_Analyst\_\_c = Default Fraud Analyst



---

## B) Auto-Calculate Severity from Risk Score

- **Object:** Fraud Case
- **Trigger:** When record is created or updated
- **Logic (Decision element):**
  - If  $\text{Risk\_Score\_c} \geq 80 \rightarrow \text{Severity\_c} = \text{High}$
  - If  $\text{Risk\_Score\_c} \geq 50 \rightarrow \text{Severity\_c} = \text{Medium}$
  - Else  $\rightarrow \text{Severity\_c} = \text{Low}$
- **Action:** Update Records  $\rightarrow$  update  $\text{Severity\_c}$

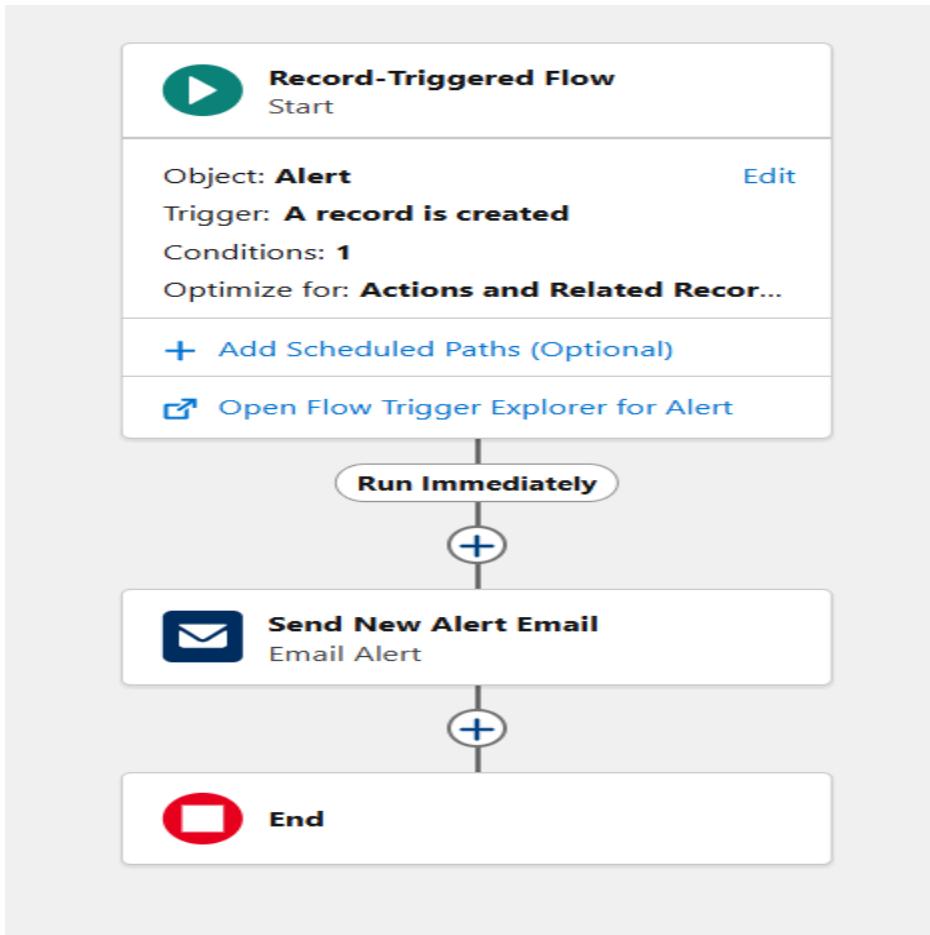


---

## C) Fraud Alert Notification (Email)

- **Object:** Alert\_\_c
- **Trigger:** When Alert is created

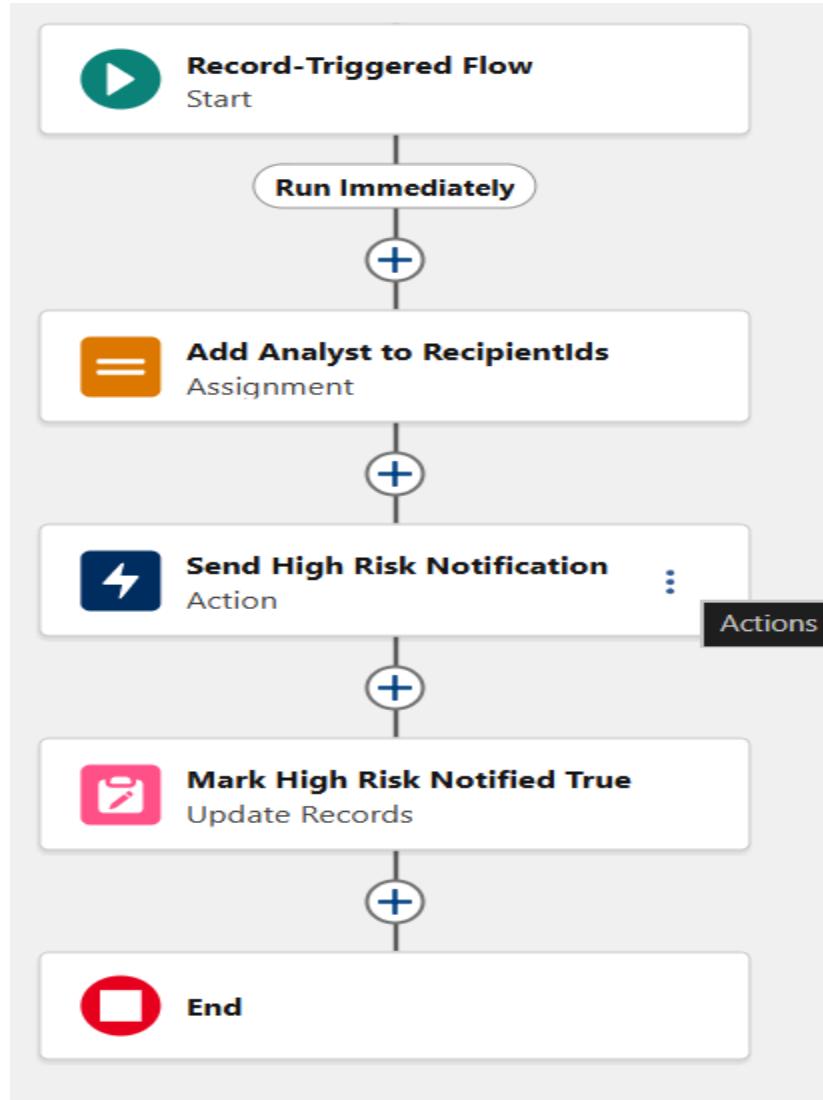
- **Condition:** Status\_\_c = "Open"
- **Action:** Email Alert → send template “New Fraud Alert Notification”
- **Recipient:** Assigned Analyst



#### 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

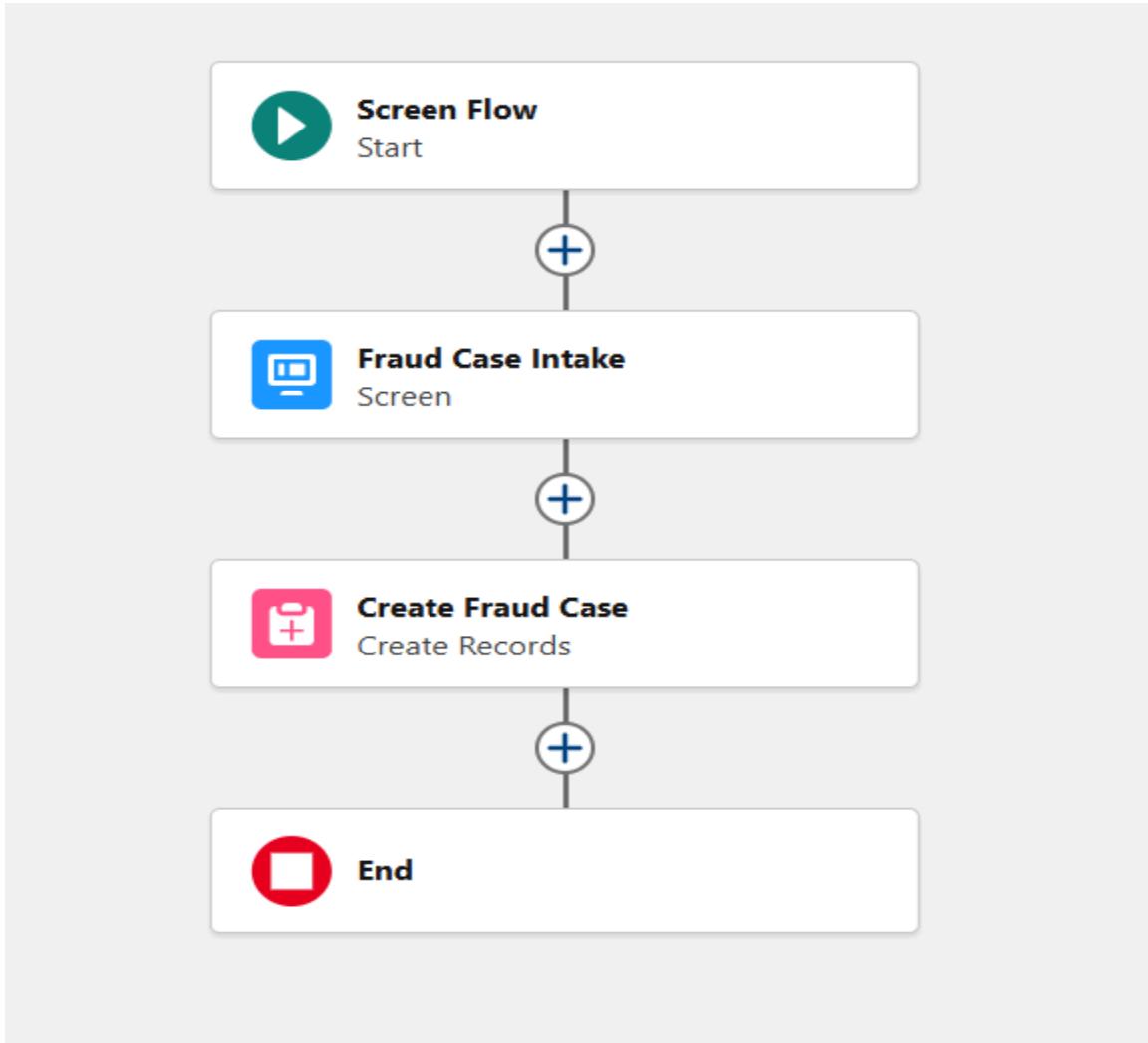


### 3. Screen Flow

#### Fraud Case Intake Form

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

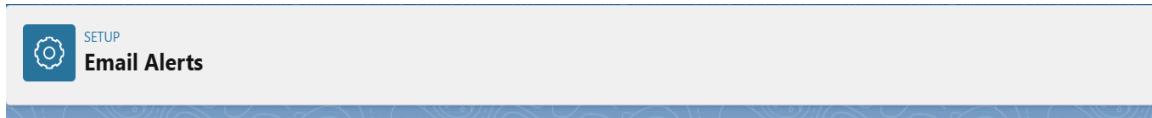
1. Screen → Enter: Customer, Transaction, Risk Score, Investigation Notes
  2. Create Records → Fraud Case record
  3. Decision → If  $\text{Risk\_Score\_c} \geq 80$ , show *High-Risk warning* message
- **Placement:** Added to Fraud Analyst Lightning Page



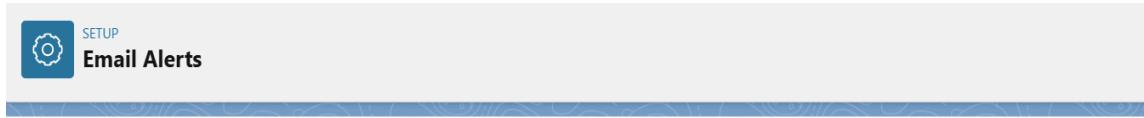
---

#### 4. Email Alerts

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



- Case Rejected Notification



- New Fraud Alert Notification



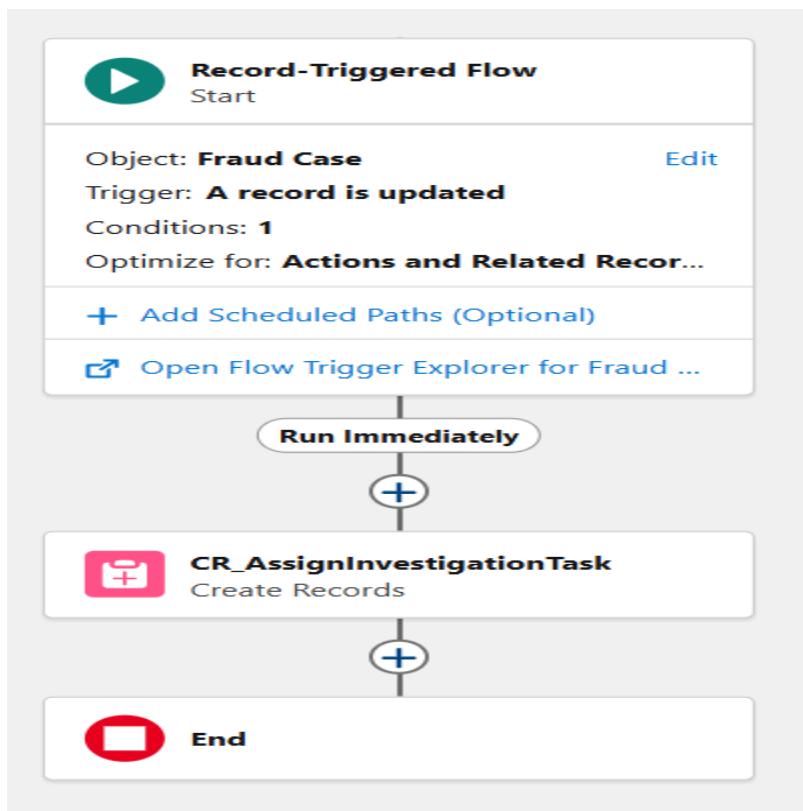
- Usage: Linked inside Flows (Record-Triggered, Post-Approval).

## 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



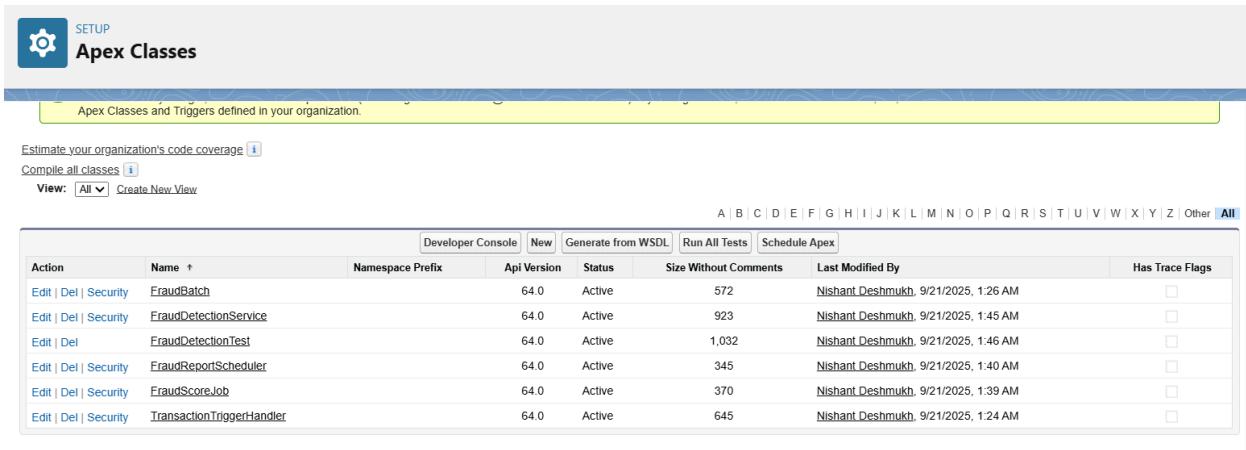
# Phase 5: Apex Programming (Developer) – Fraud Detection System

**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).



The screenshot shows the Salesforce Apex Classes page. At the top, there's a navigation bar with 'SETUP' and a gear icon. Below it, the title 'Apex Classes' is displayed. A green banner at the top of the main content area says 'Apex Classes and Triggers defined in your organization.' Below this, there are buttons for 'Estimate your organization's code coverage' and 'Compile all classes'. The 'View' dropdown is set to 'All'. A 'Create New View' button is also present. At the bottom of the page, there's a navigation bar with links labeled from A to Z, followed by 'Other' and a 'All' link. The main content area displays a table of Apex classes with columns for Action, Name, Namespace Prefix, Api Version, Status, Size Without Comments, Last Modified By, and Has Trace Flags. The table lists six classes: FraudBatch, FraudDetectionService, FraudDetectionTest, FraudReportScheduler, FraudScoreJob, and TransactionTriggerHandler. All classes listed are Active and have a size of 64.0.

Action	Name	Namespace Prefix	Api Version	Status	Size Without Comments	Last Modified By	Has Trace Flags
Edit   Del   Security	FraudBatch		64.0	Active	572	Nishant Deshmukh	9/21/2025, 1:26 AM
Edit   Del   Security	FraudDetectionService		64.0	Active	923	Nishant Deshmukh	9/21/2025, 1:45 AM
Edit   Del	FraudDetectionTest		64.0	Active	1,032	Nishant Deshmukh	9/21/2025, 1:46 AM
Edit   Del   Security	FraudReportScheduler		64.0	Active	345	Nishant Deshmukh	9/21/2025, 1:40 AM
Edit   Del   Security	FraudScoreJob		64.0	Active	370	Nishant Deshmukh	9/21/2025, 1:39 AM
Edit   Del   Security	TransactionTriggerHandler		64.0	Active	645	Nishant Deshmukh	9/21/2025, 1:24 AM

## 2. Apex Triggers

**Trigger:** **TransactionTrigger**

- Fires **before insert & update** on **Fraud\_Case\_\_c**.
- Prevents invalid risk scores.
- Ensures Investigation Notes are added before closing.

The screenshot shows the Salesforce Apex Triggers setup page. At the top, there's a header with a gear icon labeled "SETUP" and "Apex Triggers". Below the header, the title "Apex Trigger" and "TransactionTrigger" are displayed. On the right, there's a "Help for t" link. The main section is titled "Apex Trigger Detail" with tabs for "Apex Trigger", "Version Settings", and "Trace Flags". The "Apex Trigger" tab is selected. It shows the trigger code:

```
trigger TransactionTrigger on Transaction__c (before insert, before update) {
    if(Trigger.isBefore && Trigger.isInsert){
        TransactionTriggerHandler.beforeInsert(Trigger.new);
    }
    if(Trigger.isBefore && Trigger.isUpdate){
        TransactionTriggerHandler.beforeUpdate(Trigger.new, Trigger.oldMap);
    }
}
```

Below the code, there are buttons for "Edit", "Delete", "Download", and "Show Dependencies". To the right, there's a table with columns for Name, sObject Type, Status, and Last Modified By. The table shows:

Name	sObject Type	Status
TransactionTrigger	Transaction	Active

Details for the trigger include:

- Name: TransactionTrigger
- Code Coverage: 75% (3/4)
- Created By: Nishant Deshmukh, 9/21/2025, 1:23 AM
- Last Modified By: Nishant Deshmukh, 9/21/2025, 2:22 AM
- Namespace Prefix: (empty)

### 3. Trigger Design Pattern

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

The screenshot shows the Apex Classes page in Salesforce. The top navigation bar includes 'SETUP' and 'Apex Classes'. Below the header, the page title is 'TransactionTriggerHandler'. A toolbar with buttons for 'Edit', 'Delete', 'Download', 'Security', and 'Show Dependencies' is visible. The main content area displays the 'Apex Class Detail' for 'TransactionTriggerHandler'. The class details table includes columns for Name (TransactionTriggerHandler), Namespace Prefix, Created By (Nishant Deshmukh, 9/21/2025, 1:23 AM), Status (Active), Code Coverage (44% (4/9)), and Last Modified By (Nishant Deshmukh, 9/21/2025, 1:24 AM). Below the table, tabs for 'Class Body', 'Class Summary', 'Version Settings', and 'Trace Flags' are present. The 'Class Body' tab is selected, showing the following Apex code:

```

1 public class TransactionTriggerHandler {
2
3     public static void beforeInsert(List<Transaction__c> newList){
4         for(Transaction__c tx : newList){
5             if(tx.Amount__c != null && tx.Amount__c > 100000){
6                 tx.Status__c = 'Suspicious';
7             }
8         }
9     }
10
11    public static void beforeUpdate(List<Transaction__c> newList, Map<Id, Transaction__c> oldMap){
12        for(Transaction__c tx : newList){
13            Transaction__c oldTxn = oldMap.get(tx.Id);
14            if(tx.Amount__c != oldTxn.Amount__c && tx.Amount__c > 100000){
15                tx.Status__c = 'Suspicious';
16            }
17        }
18    }
19 }

```

## 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

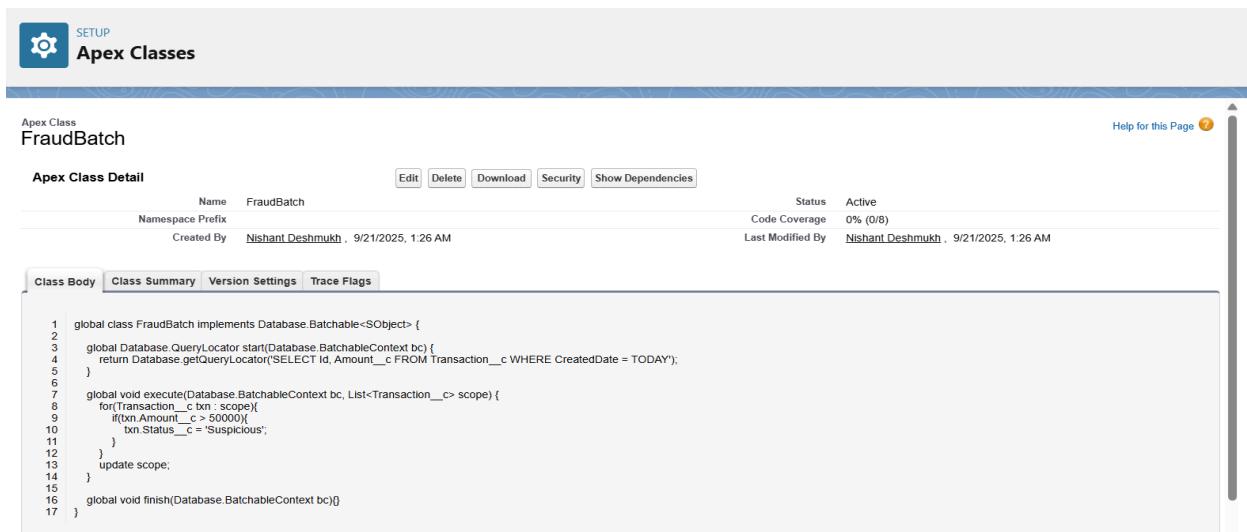
- IF logic checks **Risk\_Score\_\_c** between 0–100.
- If out of range → addError message displayed.
- IF Case is being closed without notes → throw error.

---

## 6. Batch Apex

**Class:** FraudBatch

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



The screenshot shows the Salesforce Setup Apex Classes page. At the top, there's a blue header bar with the word "SETUP" and a gear icon. Below it, the title "Apex Classes" is displayed above a search bar. The main content area has a light gray background. It shows a table for the "FraudBatch" class with columns for Name, Status, and Active status. The "Name" column shows "FraudBatch". The "Status" and "Active" columns both show "Active". Below the table, there are tabs for "Class Body", "Class Summary", "Version Settings", and "Trace Flags". The "Class Body" tab is selected, displaying the Apex code for the FraudBatch class. The code implements the Database.Batchable<SObject> interface and contains logic to query transaction records and update their status based on amount.

Name	FraudBatch	Status	Active
Namespace Prefix		Code Coverage	0% (0/8)
Created By	Nishant Deshmukh, 9/21/2025, 1:26 AM	Last Modified By	Nishant Deshmukh, 9/21/2025, 1:26 AM

```
1 global class FraudBatch implements Database.Batchable<SObject> {
2
3     global Database.QueryLocator start(Database.BatchableContext bc) {
4         return Database.getQueryLocator('SELECT Id, Amount__c FROM Transaction__c WHERE CreatedDate = TODAY');
5     }
6
7     global void execute(Database.BatchableContext bc, List<Transaction__c> scope) {
8         for(Transaction__c tx : scope){
9             if(tx.Amount__c > 50000){
10                 tx.Status__c = 'Suspicious';
11             }
12         }
13         update scope;
14     }
15
16     global void finish(Database.BatchableContext bc){}
17 }
```

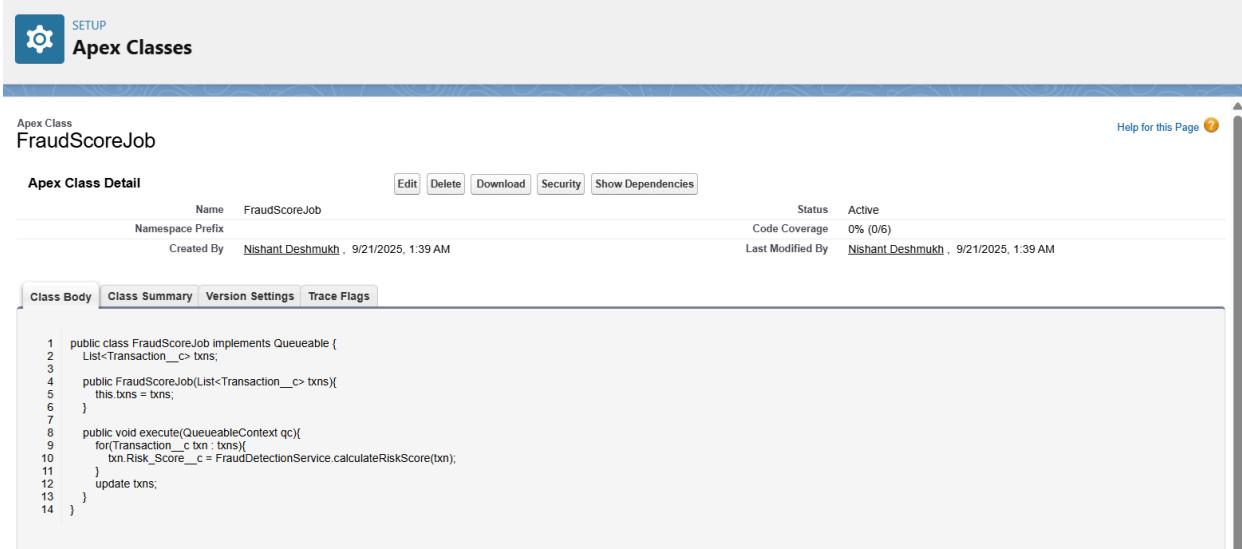
---

## 7. Queueable Apex

**Class:** FraudScoreJob

- Runs asynchronously.

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



The screenshot shows the Apex Class Detail page for the FraudScoreJob class. The class implements Queueable and performs a bulk calculation of fraud severity for a set of Transaction\_\_c objects. The execute method iterates over the transactions, calls calculateRiskScore on each, and updates the Risk\_Score\_\_c field. The code coverage is 0%.

```

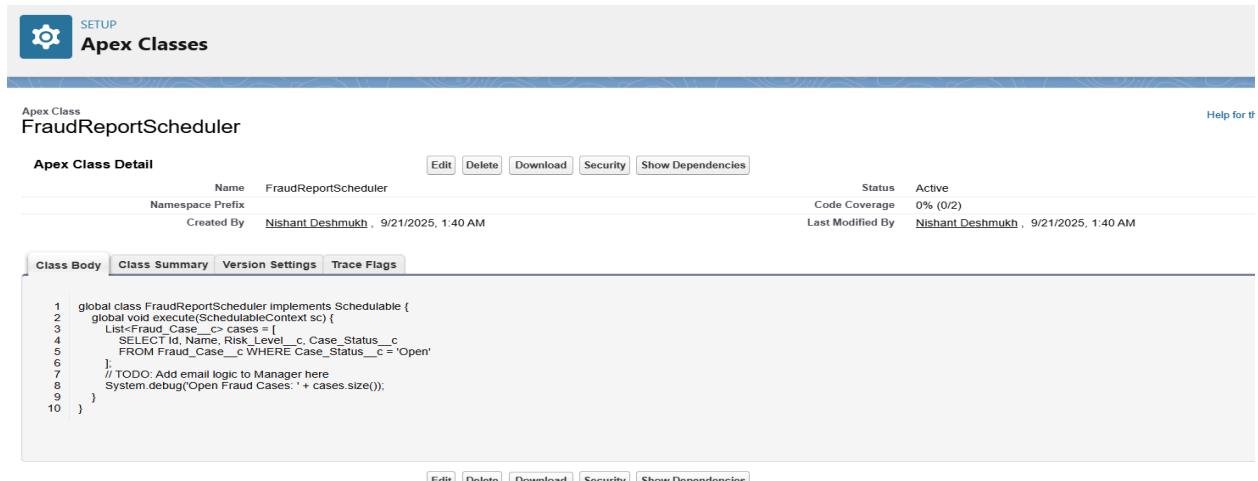
1 public class FraudScoreJob implements Queueable {
2     List<Transaction__c> txns;
3
4     public FraudScoreJob(List<Transaction__c> txns){
5         this.txns = txns;
6     }
7
8     public void execute(QueueableContext qc){
9         for(Transaction__c txn : txns){
10             txn.Risk_Score__c = FraudDetectionService.calculateRiskScore(txn);
11         }
12         update txns;
13     }
14 }

```

## 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.



The screenshot shows the Apex Class Detail page for the FraudReportScheduler class. It implements Schedulable and performs a scheduled task to find open fraud cases and log them to the console. The code coverage is 0%.

```

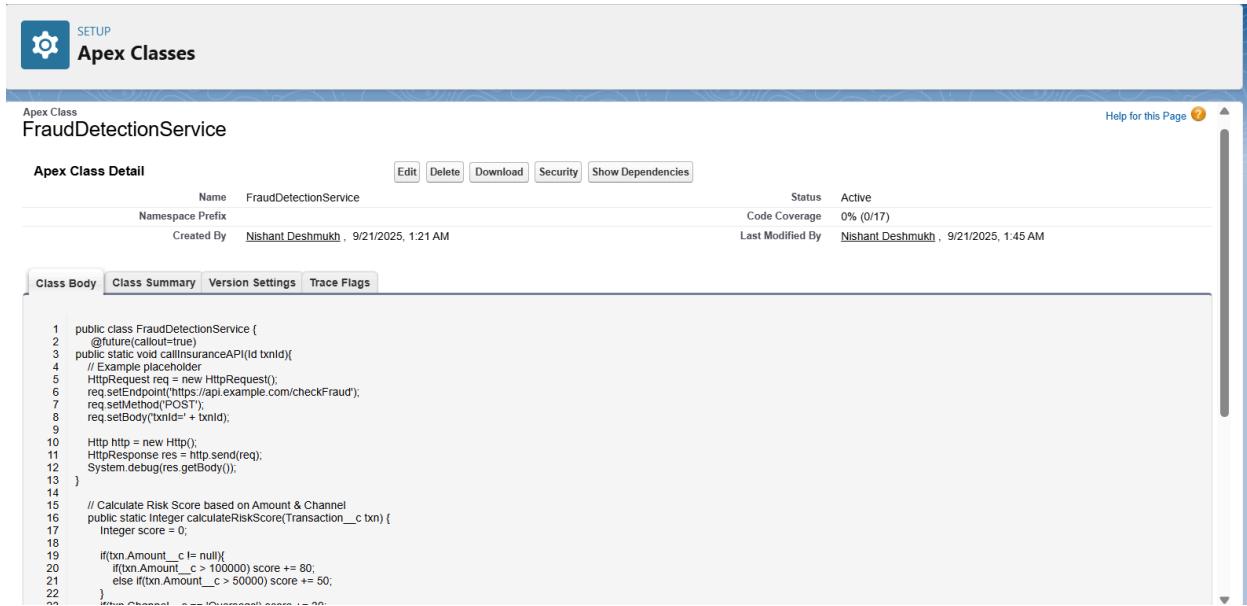
1 global class FraudReportScheduler implements Schedulable {
2     global void execute(SchedulableContext sc) {
3         List<Fraud_Case__c> cases = [
4             SELECT Id, Name, Risk_Level__c, Case_Status__c
5             FROM Fraud_Case__c WHERE Case_Status__c='Open'
6         ];
7         // TODO: Add email logic to Manager here
8         System.debug('Open Fraud Cases: ' + cases.size());
9     }
10 }

```

## 9. Future Methods (Callouts)

Class: FraudDetectionService

- Executes external fraud check **asynchronously**.



The screenshot shows the Salesforce Apex Classes page for the 'FraudDetectionService' class. The page includes a header with 'SETUP' and 'Apex Classes' buttons, and a top navigation bar with 'Edit', 'Delete', 'Download', 'Security', and 'Show Dependencies' buttons. Below the header, the class details are listed: Name (FraudDetectionService), Namespace Prefix (empty), Created By (Nishant Deshmukh, 9/21/2025, 1:21 AM), Status (Active), and Code Coverage (0% /0/17). The last modified by field is also listed. The main content area displays the Apex code for the class:

```
1 public class FraudDetectionService {
2     @future(callout=true)
3     public static void callInsuranceAPI(Id txnid){
4         // Example placeholder
5         HttpRequest req = new HttpRequest();
6         req.setEndpoint('https://api.example.com/checkFraud');
7         req.setMethod('POST');
8         req.setBody('txnid=' + txnid);
9
10        Http http = new Http();
11        HttpResponse res = http.send(req);
12        System.debug(res.getBody());
13    }
14
15    // Calculate Risk Score based on Amount & Channel
16    public static Integer calculateRiskScore(Transaction__c tx) {
17        Integer score = 0;
18
19        if(tx.Amount__c != null){
20            if(tx.Amount__c > 100000) score += 80;
21            else if(tx.Amount__c > 50000) score += 50;
22        }
23    }
24}
```

## 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).

The screenshot shows the Salesforce Setup Apex Classes page. At the top, there's a blue header bar with the word "SETUP" and a gear icon. Below it, the page title is "Apex Classes". Underneath, the specific class name "FraudDetectionTest" is displayed. On the right side of the header, there's a "Help for this Page" link. The main content area has a table titled "Apex Class Detail" with columns for Name, Namespace Prefix, Last Modified By, Status, and Active. The "Name" column shows "FraudDetectionTest", "Namespace Prefix" is empty, "Last Modified By" is "Nishant Deshmukh, 9/21/2025, 2:21 AM", "Status" is "Active", and "Created By" is "Nishant Deshmukh, 9/21/2025, 1:45 AM". Below this table, there are tabs for "Class Body", "Class Summary", "Version Settings", and "Trace Flags". The "Class Body" tab is selected, showing the Apex code for the class:

```

1  @isTest
2  public class FraudDetectionTest {
3      @isTest static void testITransactionTrigger(){
4          Transaction__c tx = new Transaction__c(
5              Transaction__Date__c = System.now(),
6              Amount__c = 200000,
7              Channel__c = 'Online',
8              Status__c = 'Success'
9          );
10         insert tx;
11     }
12     Transaction__c check = [SELECT Status__c FROM Transaction__c WHERE Id = :tx.Id];
13     System.assertEquals('Suspicious', check.Status__c);
14 }
15
16 @isTest static void testQueueableJob(){
17     Transaction__c tx = new Transaction__c(
18         Transaction__Date__c = System.now(),
19         Amount__c = 80000,
20         Channel__c = 'POS',
21         Status__c = 'Success'
22     );

```

## 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.

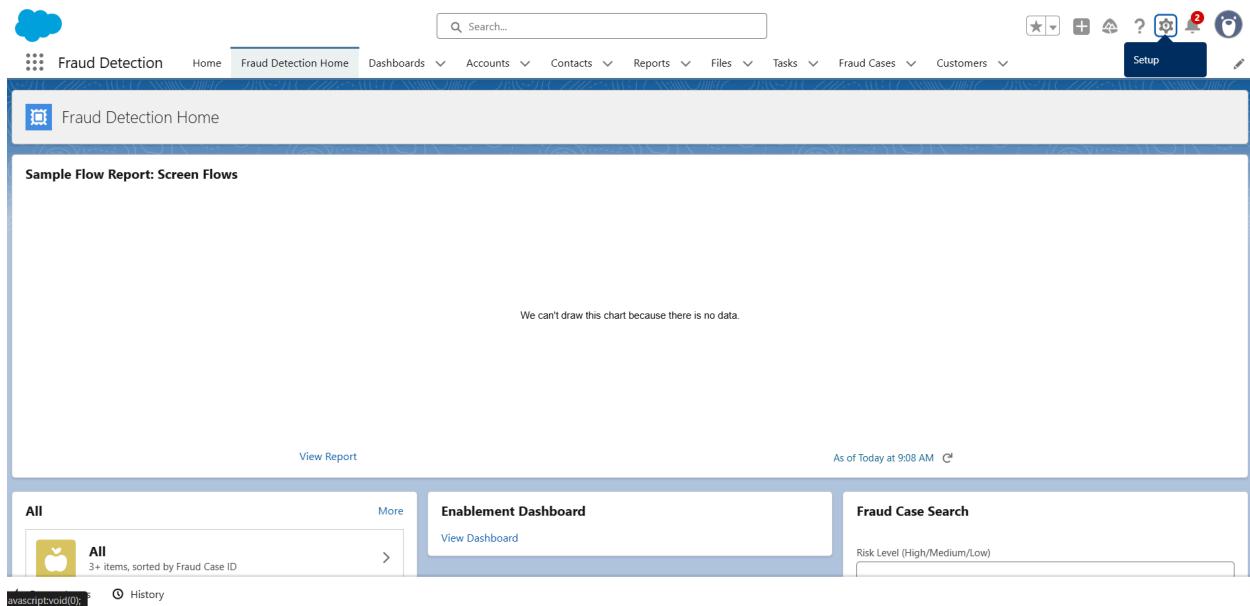
# Phase 6: User Interface Development (Fraud Detection)

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.

The screenshot shows a Lightning Page Detail page for a page named "Fraud\_Detection\_Home". The page has a "Name" field set to "Fraud\_Detection\_Home", a "Label" field set to "Fraud Detection Home", and a "Description" field. There are "Edit", "Clone", and "Delete" buttons at the top and bottom of the page.

## 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.

## Lightning Component Detail

 Help for this Page 

**Lightning Web Component Bundle Detail**

Name	fraudCaseSearch	Label	fraudCaseSearch
Created By	Nishant Deshmukh, 9/22/2025, 9:06 AM	Description	
Modified By	Nishant Deshmukh, 9/22/2025, 10:52 AM		

LWC Dependencies

Search for Lightning Component (Case Sensitive)

**Search**

Name	Label	Type	Namespace Prefix	Api Version
> fraudCaseSearch	fraudCaseSearch	LWC		59

### ● **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.

## Lightning Component Detail

 Help for this Page 

**Lightning Web Component Bundle Detail**

Name	fraudCaseManager	Label	fraudCaseManager
Created By	Nishant Deshmukh, 9/22/2025, 10:52 AM	Description	
Modified By	Nishant Deshmukh, 9/22/2025, 10:52 AM		

LWC Dependencies

Search for Lightning Component (Case Sensitive)

**Search**

Name	Label	Type	Namespace Prefix	Api Version
> fraudCaseManager	fraudCaseManager	LWC		64

## 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.

The screenshot shows the Apex Class Detail page for the class 'FraudCaseService'. The page includes tabs for 'Apex Class Detail', 'Class Body' (which is selected), 'Class Summary', 'Version Settings', and 'Trace Flags'. The 'Class Body' tab displays the following Apex code:

```
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.

The screenshot shows the Salesforce Setup interface for creating a Named Credential. The top navigation bar includes 'SETUP > NAMED CREDENTIALS' and 'Bank API Credential'. On the right, there are 'Edit' and 'Delete' buttons. The main form fields are:

Label	Name
Bank API Credential	BankAPI_Credentials
URL	
https://fraudapi.bank.com	
Enabled for Callouts	<input checked="" type="checkbox"/>
Authentication	
External Credential	<a href="#">Bank API External Credential</a>
Client Certificate	
Callout Options	
Generate Authorization Header	<input checked="" type="checkbox"/>
Allow Formulas in HTTP Header	<input type="checkbox"/>
Allow Formulas in HTTP Body	<input type="checkbox"/>

## 2. External Services

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

The screenshot shows the Salesforce Setup interface with the path 'SETUP > NAMED CREDENTIALS'. A card titled 'Bank API External Credential' is displayed. It includes fields for Label ('Bank API External Credential'), Name ('BankAPI\_External\_Cred'), Authentication Protocol ('Basic Authentication'), Managed Package Access, and Created By Namespace. There are 'Edit' and 'Delete' buttons at the top right.

Label	Name
Bank API External Credential	BankAPI_External_Cred

Authentication Protocol: Basic Authentication

Managed Package Access

Created By Namespace: [Namespace](#)

Edit Delete

## 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  $\geq 80$ .
- External systems subscribed to the event channel for real-time alerts.

Custom Fields & Relationships						
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.

The screenshot shows the Change Data Capture setup page in the Salesforce Setup interface. At the top left is a blue square icon with a white gear symbol. To its right is the word "SETUP". Below these are the section titles "Change Data Capture" and "Available Entities". A search bar with the placeholder "Type to filter list..." is located below the section title. To the right of the search bar is a "Selected Entities" section containing the text "Transaction (Transaction\_\_c)".

---

## 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.

### Remote Site Details

[Help for this Page](#)

Remote Site Detail		<a href="#">Edit</a>	<a href="#">Delete</a>	<a href="#">Clone</a>
Remote Site Name	BankFraudAPI			
Remote Site URL	<a href="https://api.bankfraudcheck.com">https://api.bankfraudcheck.com</a>			
Disable Protocol Security	<input type="checkbox"/>			
Description	Bank Fraud Detection API endpoint for callouts			
Active	<input checked="" type="checkbox"/>			
Created By	Nishant Deshmukh	9/23/2025, 10:52 AM		
		<a href="#">Edit</a>	<a href="#">Delete</a>	<a href="#">Clone</a>

# 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.

Transaction Duplicate Rule  
Prevent\_Duplicate\_Transactions

[Help for this Page](#)

Duplicate Rule Detail		<a href="#">Edit</a> <a href="#">Delete</a> <a href="#">Clone</a> <a href="#">Deactivate</a>	Order 1 of 1 [ <a href="#">Reorder</a> ]
Rule Name	Prevent_Duplicate_Transactions		
Description			
Object	Transaction		
Record-Level Security	Enforce sharing rules		
Action On Create	Block	Operations On Create	<input type="checkbox"/> Alert <input type="checkbox"/> Report
Action On Edit	Allow	Operations On Edit	<input type="checkbox"/> Alert <input type="checkbox"/> Report
Alert Text	This transaction already exists in the system.		
Active	<input checked="" type="checkbox"/>		
Matching Rule	<input checked="" type="checkbox"/> Match_Transaction_ID <input checked="" type="checkbox"/> Mapped	Matching Criteria	Transaction: Name EXACT MatchBlank = FALSE
Conditions			
Created By	Nishant Deshmukh, 9/24/2025, 2:14 AM	Modified By	Nishant Deshmukh, 9/24/2025, 2:14 AM
<a href="#">Edit</a> <a href="#">Delete</a> <a href="#">Clone</a> <a href="#">Deactivate</a>			

## 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.

Fraud Cases by Risk Level				
Total Records	Total Risk Score			
8	425			
<input type="checkbox"/> Risk Level	<input type="checkbox"/> Fraud Case: Fraud Case ID	<input type="checkbox"/> Case Status	<input type="checkbox"/> Risk Score	<input type="checkbox"/> Fraud Case: Owner Role
<input type="checkbox"/> High (5)	FC-0002	Open	85	-
	FC-0003	Open	80	-
	FC-0004	Open	85	-
	FC-0006	Under Investigation	85	-
	FC-0007	Under Investigation	90	-
Subtotal			425	
<input type="checkbox"/> Low (3)	FC-0001	-	-	-
	FC-0005	Open	-	-
	FC-0008	Under Investigation	-	-
Subtotal			0	
Total (8)			425	

- **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.

The screenshot shows the 'Custom Report Types' section in a software application. At the top, there's a header with a gear icon and the text 'Custom Report Types'. Below the header, the title 'Transactions with Fraud Cases' is displayed. To the right of the title are four buttons: 'Preview Layout', 'Edit Layout', 'Clone', and 'Delete'. A 'Close' button is located at the far right. A note below the title states: 'Below is the information for this custom report type. You can click the buttons on this to preview or update information for the custom report type.' The main area is divided into two sections: 'Details' and 'Object Relationships'. The 'Details' section contains fields like Display Label (Transactions with Fraud Cases), API Name (Transactions\_with\_Fraud\_Cases), Description (Shows transaction details along with associated fraud case info), Created By (Nishant Deshmukh, 9/24/25, 10:04 PM), Store in Category (other), Deployment Status (Deployed), and Modified By (Nishant Deshmukh, 9/24/25, 10:04 PM). The 'Object Relationships' section includes a diagram illustrating the relationship between 'Transactions (A)' and 'Fraud Cases (B)'. It shows two overlapping circles labeled 'A' and 'B', with a shaded intersection. An arrow points down to a box containing two columns of horizontal bars, one blue and one orange, labeled 'A' and 'B' respectively. The blue bar has three horizontal lines, and the orange bar has four.

---

## 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.