

Dynamic Route Optimization for Last-Mile Delivery in Salesforce CRM

Problem Statement

In TCS's Last-Mile operations, current route planning methods are insufficiently adaptive and often static. They fail to account for real-world conditions such as traffic fluctuations, road closures, delivery time windows, vehicle constraints, and customer availability. This leads to sub-optimal routes, increased fuel/time costs, frequent delays, missed or failed deliveries, and poor visibility into deviations. The lack of dynamic response to real-time disruptions reduces operational efficiency, degrades customer satisfaction, and limits scalability.

Phase 1: Problem Understanding & Industry Analysis

Goal: Understand what we're building and why for Last Mile CRM.

1. Requirement Gathering

To develop a dynamic and adaptive route optimization system within Salesforce CRM, a detailed requirement study was conducted across multiple operational levels. The goal was to capture both functional and non-functional requirements that would drive intelligent routing decisions.

- Predict delivery ETAs with high accuracy using live traffic and GPS data.
- Plan and reoptimize routes dynamically based on real-time conditions.
- Assign deliveries intelligently to agents considering constraints like shift times, capacity, and location.
- Provide real-time tracking and status visibility for all stakeholders.
- Alert system for delays, deviations, or high-risk deliveries.
- Digital proof of delivery with photo/signature capture.
- Seamless integration with Salesforce modules, mapping APIs, and logistics systems.

2. Stakeholder Analysis

Each stakeholder plays a unique role in ensuring successful last-mile delivery. The system's design must address their distinct requirements and expectations.

- **Operations / Logistics Manager:** Seeks visibility on performance KPIs, operational costs, and delivery efficiency.
- **Dispatch / Routing Team:** Requires dashboards to plan, monitor, and modify routes dynamically when disruptions occur.
- **Field Agents / Drivers:** Need mobile interfaces for navigation, delivery confirmation, and status updates.
- **Customer Support:** Relies on accurate, real-time delivery status to resolve customer queries efficiently.
- **Customers:** Expect timely updates, accurate ETAs, and smooth delivery experiences.
- **Salesforce Admin / IT Team:** Ensures secure system setup, scalability, and integration with Salesforce ecosystems.

3. Business Process Mapping

The proposed process leverages Salesforce CRM to automate and optimize delivery workflows:

Flow: Order received → System predicts ETA & suggests optimal route → Dispatch assigns to

agent → Agent starts journey → Real-time tracking updates CRM → Route deviations trigger alerts → Delivery completed → Feedback recorded → Data stored for continuous improvement.

Exception Management: In case of delays, system recalculates routes, reassigns deliveries if needed, and sends automated notifications to customers.

4. Industry-Specific Use Case Analysis

In logistics, real-world conditions are unpredictable. Systems must adapt dynamically to maintain efficiency and reliability. This section outlines key challenges specific to the last-mile delivery sector.

- High traffic density in urban areas significantly affects delivery times.
- Frequent changes in customer availability require dynamic re-routing.
- Vehicle limitations (capacity, fuel range) influence delivery sequence planning.
- External factors like weather or road closures often disrupt delivery schedules.
- Customer satisfaction heavily depends on real-time updates and transparency.

5. App / Solution Landscape

An evaluation of Salesforce AppExchange solutions and third-party integrations was conducted to assess capabilities for dynamic route planning and optimization.

- **Existing Solutions:** Tools like Routific, Onfleet, and Locus provide strong optimization but lack deep Salesforce integration.
- **Gap Identified:** Most solutions offer static routing and limited real-time re-optimization features.
- **Decision:** Build custom Salesforce components using Apex, Flow, and Einstein AI to support dynamic routing, predictive ETA modeling, and automated alerts.

6. Conclusion

The Phase 1 analysis of Dynamic Route Optimization for Last-Mile Delivery in Salesforce CRM establishes a solid foundation for the project. By understanding stakeholder needs, mapping operational workflows, and analyzing industry use cases, this phase highlights the potential of combining AI-driven analytics with Salesforce CRM capabilities. The proposed system aims to deliver measurable business value — reducing delivery time by up to 20%, lowering fuel costs, and enhancing customer satisfaction through real-time visibility and proactive communication. Future phases will focus on technical architecture, data modeling, and the development of optimization algorithms using Salesforce Einstein and third-party mapping APIs.

Dynamic Route Optimization

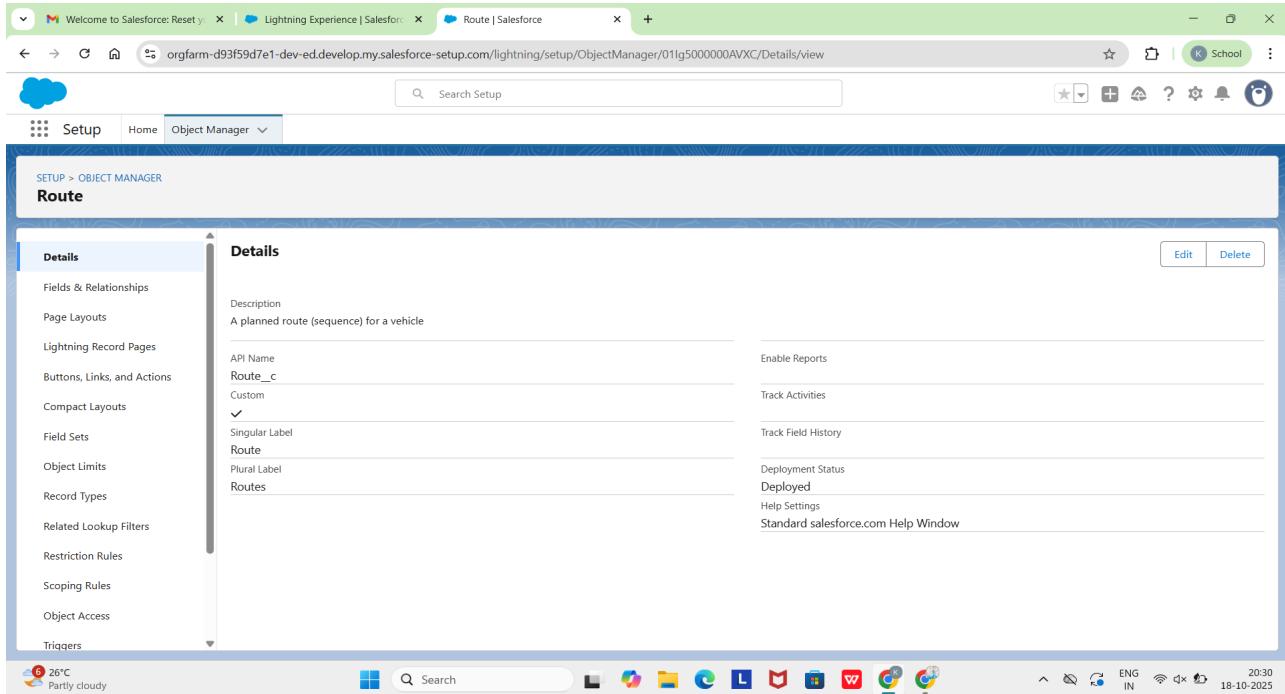
Salesforce-Based Delivery Management System

Phase 2: Org Setup & Configuration

Academic Project Report

Step 1: Salesforce Editions

Checked the Salesforce Edition by navigating to Setup → Company Settings → Company Information. The org used for this project is a Developer Edition suitable for custom app development and testing.



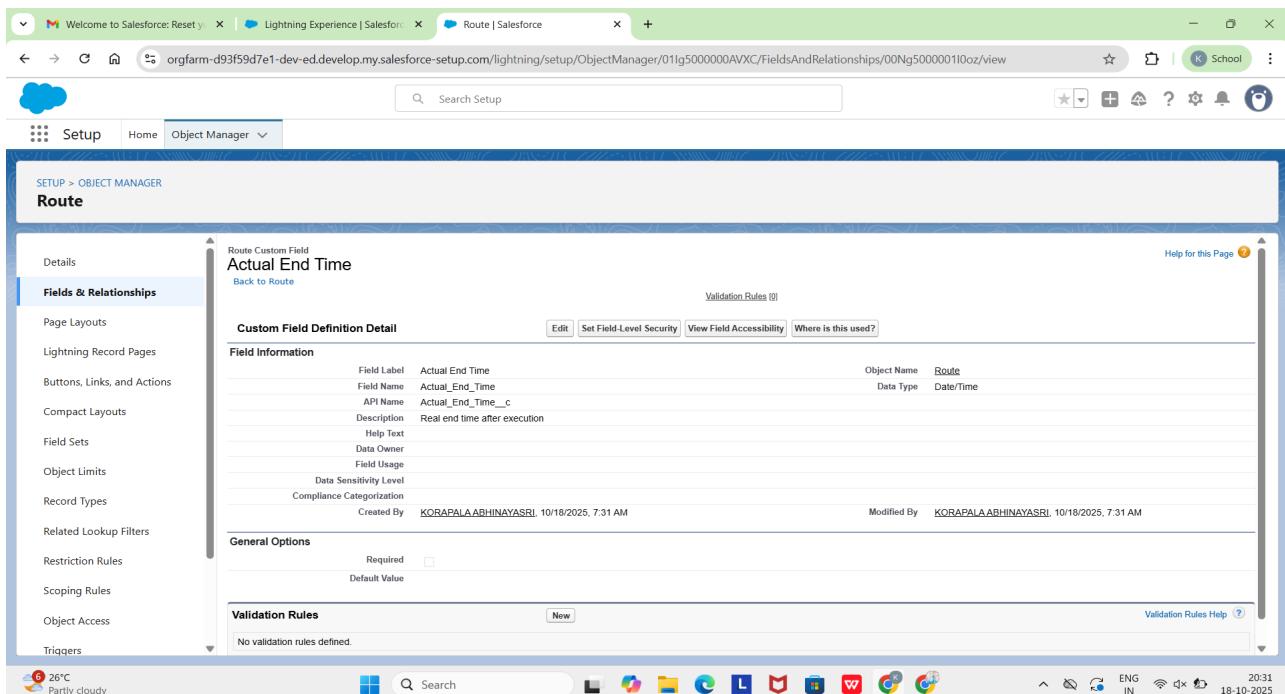
The screenshot shows the Salesforce Setup interface for the 'Route' object. The left sidebar lists various setup categories like Fields & Relationships, Page Layouts, and Triggers. The main 'Details' tab is selected, displaying the following information:

Description	Value
Description	A planned route (sequence) for a vehicle
API Name	Route__c
Custom	✓
Singular Label	Route
Plural Label	Routes
Enable Reports	
Track Activities	
Track Field History	
Deployment Status	Deployed
Help Settings	Standard salesforce.com Help Window

The bottom of the screen shows the standard Salesforce navigation bar with icons for Home, Object Manager, and other setup tools.

Step 2: Company Profile Setup

Configured company profile under Setup → Company Settings → Company Information. Details entered include organization name, address, primary contact, timezone, locale (English - India), and currency (INR).



The screenshot shows the Salesforce Setup interface for the 'Route' object, specifically the 'Fields & Relationships' tab. A custom field named 'Actual End Time' is being configured. The 'Custom Field Definition Detail' section includes the following details:

Field Label	Actual End Time	Object Name	Route
Field Name	Actual_End_Time	Data Type	Date/Time
Description	Real end time after execution		
Help Text			
Data Owner			
Field Usage			
Data Sensitivity Level			
Compliance Categorization			

The 'General Options' section shows 'Required' checked. The 'Validation Rules' section indicates 'No validation rules defined'.

The bottom of the screen shows the standard Salesforce navigation bar with icons for Home, Object Manager, and other setup tools.

Step 3: Business Hours & Holidays

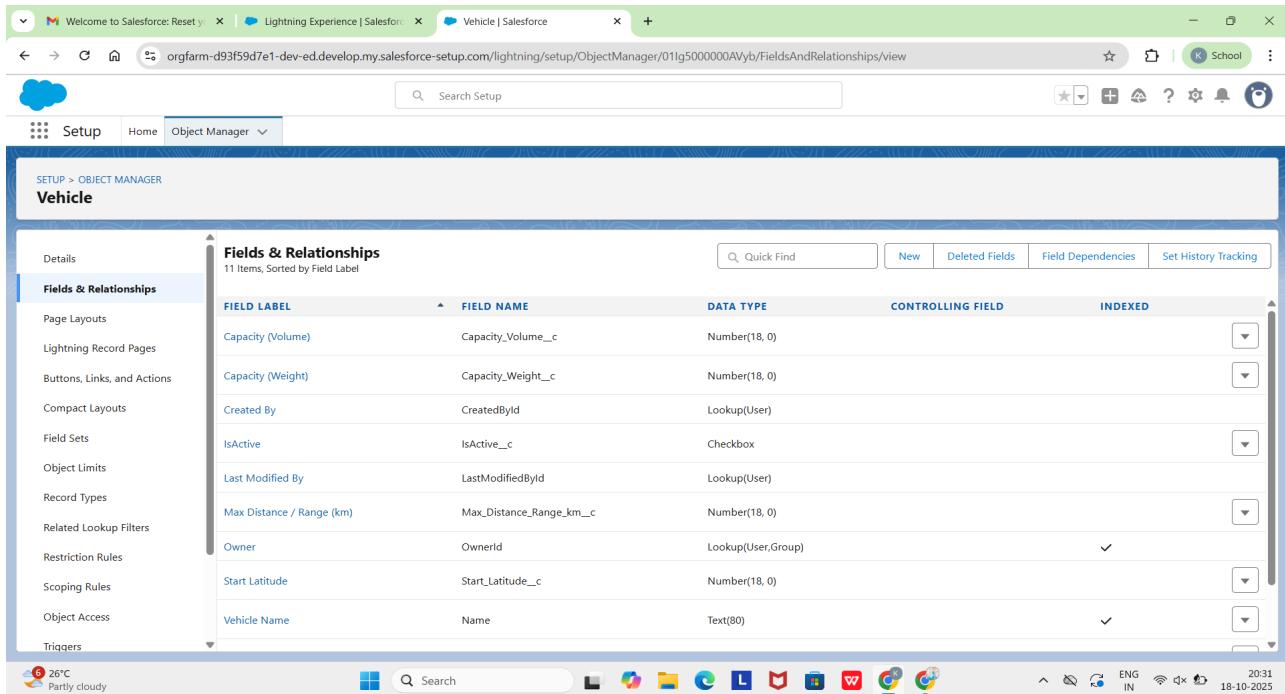
Set business hours as 9:00 AM to 6:00 PM and added standard holidays such as Republic Day, Independence Day, and Diwali under Setup → Holidays.

Step 4: Fiscal Year Settings

Used Standard Fiscal Year (April to March) to align with general financial reporting. Custom fiscal year was not enabled.

Step 5: User Setup & Licenses

Created different users like Admin, Dispatcher, Driver, and Manager with Salesforce or Platform licenses. Assigned profiles and roles to ensure proper access based on responsibility.



The screenshot shows the Salesforce Object Manager interface for the 'Vehicle' object. The left sidebar lists various setup options like Page Layouts, Lightning Record Pages, and Buttons, Links, and Actions. The main content area is titled 'Fields & Relationships' and displays 11 items, sorted by Field Label. The table includes columns for FIELD LABEL, FIELD NAME, DATA TYPE, CONTROLLING FIELD, and INDEXED. The fields listed are: Capacity (Volume), Capacity_Volume__c, Number(18, 0); Capacity (Weight), Capacity_Weight__c, Number(18, 0); Created By, CreatedById, Lookup(User); IsActive, IsActive__c, Checkbox; Last Modified By, LastModifiedById, Lookup(User); Max Distance / Range (km), Max_Distance_Range_km__c, Number(18, 0); Owner, OwnerId, Lookup(User,Group); Start Latitude, Start_Latitude__c, Number(18, 0); and Vehicle Name, Name, Text(80). The interface also includes a search bar, a quick find button, and buttons for New, Deleted Fields, Field Dependencies, and Set History Tracking.

Step 6: Profiles

Configured custom profiles for Admin, Dispatcher, and Driver. Admin has full system access, Dispatcher can manage deliveries and routes, and Drivers can view assigned deliveries only.

Step 7: Roles

Set up role hierarchy: System Admin (top) → Dispatcher (reports to Admin) → Driver (lowest level). Managers can view records of the users reporting to them, ensuring visibility control.

Step 8: Permission Sets

Created permission sets: ps_RoutingAccess (for additional access to route objects) and ps_DriverTracking (for delivery status update permissions). Assigned to selected users in addition

to their profiles.

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

- Setup Path:** SETUP > OBJECT MANAGER > Status Log
- Section:** Fields & Relationships
- Page Layouts:** Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Restriction Rules, Scoping Rules, Object Access, Triggers.
- Current Page:** Status Log New Custom Field
- Step 1: Choose the field type** (Step 1 of 5)
 - Data Type:** None Selected (selected), Auto Number, Formula, Roll-Up Summary, Lookup Relationship (selected), Master-Detail Relationship.
 - Description of Lookup Relationship:** Creates a relationship that links the object to another object. The relationship field allows users to click on a lookup icon to select a value from a popup list. The other object is the source of the values in the list.
 - The relationship field is required on all detail records.
 - The ownership and sharing of a detail record are determined by the master record.
 - When a user deletes the master record, all detail records are deleted.
 - You can create roll-up summary fields on the master record to summarize the detail records.
 - Description of Master-Detail Relationship:** Creates a special type of parent-child relationship between this object (the child, or "detail") and another object (the parent, or "master") where:
 - The relationship field is required on all detail records.
 - The ownership and sharing of a detail record are determined by the master record.
 - When a user deletes the master record, all detail records are deleted.
 - You can create roll-up summary fields on the master record to summarize the detail records.
- Help for this Page**
- Next** and **Cancel** buttons

Step 9: Organization-Wide Defaults (OWD)

Set OWDs under Setup → Security → Sharing Settings. DeliveryOrder and Vehicle objects are Private, while Route and StatusLog are Public Read/Write to allow coordination between dispatchers and drivers.

Step 10: Sharing Rules

Configured sharing rules to share Delivery Orders with Dispatchers and Drivers in the same route region. Also shared route records with Managers for tracking and reporting.

Step 11: Login Access Policies

Enabled admin login access for troubleshooting and delegated admin access for dispatchers when required.

Step 12: Dev Org Setup

Created and configured the Salesforce Developer Org (free edition) for the Dynamic Route Optimization project. All customization and configurations were tested in this org before deployment.

Step 13: Sandbox Usage

Created Developer Sandbox for testing automation rules and triggers such as workflow updates and validation rules. Ensured proper isolation of development and production environments.

Step 14: Deployment Basics

Used outbound Change Sets to migrate metadata from sandbox to production. Additionally, explored Salesforce CLI (SFDC) for automated deployments.

The screenshot shows the Salesforce Status Log - New Relationship page. The URL in the browser is <https://orgfarm-d93f59d7e1-dev-ed.my.salesforce.com/lightning/setup/ObjectManager/01lg500000AW0D/FieldsAndRelationships/new>. The page title is "Status Log New Relationship". On the left, there's a sidebar with "SETUP > OBJECT MANAGER" and a list of categories: Details, Fields & Relationships (which is selected), Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Restriction Rules, Scoping Rules, Object Access, and Triggers. The main content area is titled "Step 2. Choose the related object". It has a dropdown menu labeled "Related To" with "Route Stop" selected. At the bottom right of the content area are "Previous", "Next", and "Cancel" buttons. The top navigation bar includes tabs for "Setup", "Home", and "Object Manager". The top right corner shows various icons for user profile, help, and system status. The bottom of the screen shows the Windows taskbar with various pinned icons and the date/time: 18-10-2025.

Custom Objects and Relationships

The following custom objects were created in Salesforce to support route planning, delivery tracking, and vehicle management.

Object Label	Object API Name	Description / Purpose
Delivery Order	DeliveryOrder__c	Represents delivery orders to be routed and fulfilled.
Route	Route__c	A planned route assigned to a vehicle and driver.
Route Stop	RouteStop__c	Stops within a route linked to delivery orders.
Vehicle	Vehicle__c	Details of vehicles used for deliveries.
Driver	Driver__c	Information about drivers and assigned vehicles.
Status Log	StatusLog__c	Logs route and stop-level events and status changes.
Traffic Alert	TrafficAlert__c	Optional object for tracking external traffic or delay alerts.

Relationships Between Objects

- Route → RouteStop (One-to-Many)
- RouteStop → DeliveryOrder (Lookup)
- Route → Vehicle (Lookup)
- Route → Driver (Lookup)
- StatusLog → Route / RouteStop (Lookup)
- TrafficAlert → Route / RouteStop (Optional Lookup)

Conclusion

In Phase 2 of the Dynamic Route Optimization project, the Salesforce Org was successfully configured with core administrative, security, and data components. Custom objects such as Delivery Order, Route, Route Stop, and Vehicle were developed to establish a comprehensive routing model. Role hierarchies, profiles, permission sets, and OWD ensured secure and efficient collaboration between dispatchers and drivers. Finally, deployment and sandbox configurations were implemented for a structured development lifecycle.

Dynamic Route Optimization

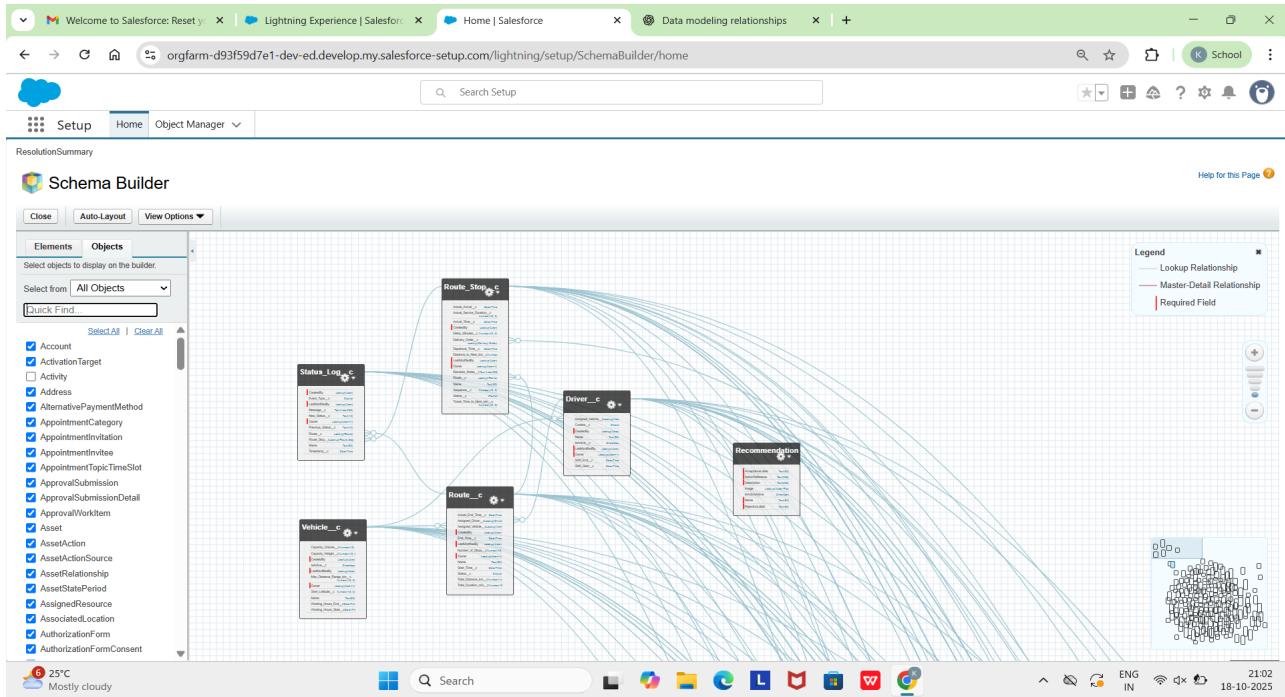
Salesforce-Based Delivery Management System

Phase 3: Automation & Business Logic

Academic Project Report

Step 1: Schema Builder Overview

The Schema Builder provides a visual representation of all objects, fields, and relationships in the Salesforce environment. In this project, custom objects such as Delivery Orders, Routes, Route Stops, Vehicles, Drivers, and Status Logs were created and connected using lookup relationships. This ensures seamless data connectivity and enables route tracking and optimization.



Step 2: Custom Tabs Creation

Custom tabs were created to provide easy navigation to each custom object within the Salesforce app. Tabs such as Delivery Orders, Routes, Route Stops, Vehicles, Drivers, and Status Logs were added to enhance usability.

The screenshot shows the 'Tabs' setup page in Salesforce. It displays a table of 'Custom Object Tabs' with columns for Action, Label, Tab Style, and Description. The tabs listed are:

Action	Label	Tab Style	Description
Edit Del	Delivery Orders	Truck	Tab for managing delivery orders in route optimization
Edit Del	Drivers	People	
Edit Del	Routes	Map	
Edit Del	Route Stops	Phone	
Edit Del	Status Logs	Mail	
Edit Del	Vehicles	Truck	

Below this are sections for 'Web Tabs' (No Web Tabs have been defined), 'Visualforce Tabs' (No Visualforce Tabs have been defined), and 'Lightning Component Tabs' (No Lightning Component Tabs have been defined).

Step 3: Lightning App Setup

A new Lightning App named 'Dynamic Route Optimization' was created using the App Manager. The app includes navigation items for all primary custom objects, allowing streamlined access to manage routes, vehicles, and delivery orders.

The screenshot shows the 'App Manager' setup page. A 'New Lightning App' dialog is open, listing 'Available Items' and 'Selected Items'. The selected items include:

- Delivery Orders
- Routes
- Route Stops
- Vehicles
- Drivers
- Status Logs

At the bottom of the dialog, there is a progress bar with three steps: '15 My Service Journey', '16 Platform', and '17 Queue Management'. The 'Next' button is visible at the end of the bar.

Step 4: Profiles and Permission Management

Profiles were configured to manage object-level and field-level access for different roles such as Admin, Dispatcher, and Driver. Each profile specifies permissions like Read, Create, Edit, and

Delete for custom objects. Password policies and session timeouts were also enforced for security compliance.

The screenshot shows the Salesforce Setup interface under the Profiles section. On the left, there's a search bar with 'prof' typed in and a sidebar with 'Users' and 'Profiles' selected. The main area contains several tables:

- Data Org Feature Allow Lists:** A grid of checkboxes for various features across different objects.
- Custom Object Permissions:** Two grids for Delivery Orders, Routes, and Route Stops, defining permissions for Read, Create, Edit, Delete, View All Records, Modify All Records, and View All Fields.
- Session Settings:** Set 'Session Times Out After' to '2 hours of inactivity' and 'Session Security Level Required at Login' to 'None'.
- Password Policies:** Set 'User passwords expire in' to '90 days', 'Enforce password history' to '3 passwords remembered', 'Minimum password length' to '8', and 'Password complexity requirement' to 'Must include alpha and numeric characters'.

Step 5: Record Creation Examples

Sample records were created to validate object relationships and workflow triggers. For instance, a new Delivery Order record was created with address and timing details. Similarly, a Route record was created linking driver and vehicle information.

The screenshot shows the Salesforce Lightning interface for creating a new Delivery Order. The page title is 'New Delivery Order'. The 'Information' section contains the following fields:

- * Delivery Order Name: DO-001
- Order Number: 67478g
- Street: Jubilee Hill
- City: Kadapa
- State: Andhra Pradesh
- Postal Code: 516004
- Country: India

On the right, there are buttons for Import, Change Owner, and Assign Label. At the bottom, there are 'Cancel', 'Save & New', and 'Save' buttons. The status bar at the bottom shows 'Finance headline India reported 6...' and the date '18-10-2025'.

Welcome to Salesforce: Reset | Recently Viewed | Delivery Orders | Route-01 | Route | Salesforce | Data modeling relationships

orgfarm-d93f59d7e1-dev-ed.develop.lightning.force.com/lightning/r/Route__c/a01g500001SSSDAA4/view

Dynamic Route Opt... Delivery Orders Routes Route Stops Vehicles Drivers Status Logs Reports Dashboards

Route Route-01

Route "Route-01" was created.

Related Details

Route Name: Route-01 Owner: KORAPALA ABHINAYASRI

Start Time: End Time:

Total Distance (km): 230.00 Total Duration (min): 147.00

Assigned Vehicle: Van-01 Assigned Driver: Abhinayarsi

Status: nProgress Number of Stops: 3 Actual End Time:

Extremely humid Now

21:46 ENG IN 18-10-2025

Step 6: Custom Report Types

A custom report type was created under Setup → Reports & Dashboards → Report Types. The report 'Routes with Route Stops' allows generating combined data insights to monitor route efficiency and stop performance.

Welcome to Salesforce: Reset | Recently Viewed | Delivery Orders | Recent | Reports | Salesforce | Report Types | Salesforce | Data modeling relationships

orgfarm-d93f59d7e1-dev-ed.develop.my.salesforce-setup.com/lightning/setup/CustomReportTypeLightning/new

Setup Home Object Manager

report

Feature Settings Analytics Reports & Dashboards Report Types Security

Report Types

Primary Object: Routes

Display Label: Routes with Route Stops API Name: Routes_with_Route_Stops

Description: Shows all routes and their associated stops

Note: Description will be visible to users who create reports.

Store in Category: Other Reports

Set Availability

Cancel Next

25°C Mostly cloudy

21:54 ENG IN 18-10-2025

Step 7: Automation Tools Overview

Automation tools such as Workflow Rules, Process Builder, and Flows are essential to streamline operations. In this phase, triggers can be used to automatically update Route status when all

associated Route Stops are marked as Completed. Flows may also be introduced to assign drivers dynamically based on route capacity and priority.

Conclusion

In Phase 3 of the Dynamic Route Optimization project, Salesforce automation and app configuration were implemented. The creation of custom tabs, a Lightning App, and schema design ensured better navigation and data integrity. Profiles and permission sets reinforced security and access control. Additionally, the configuration of reports and planned automation workflows set the foundation for efficient delivery management within Salesforce.

Phase 4: Automation Implementation

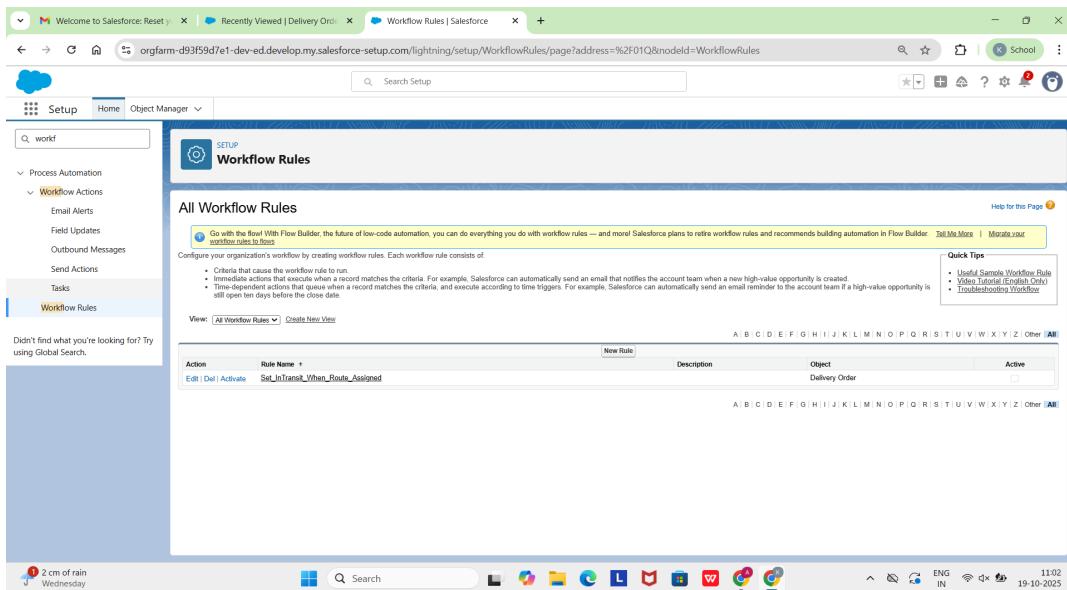
Project Title: Dynamic Route Optimization

Introduction

This phase focuses on automating key business processes in the Dynamic Route Optimization project using Salesforce automation tools such as Workflow Rules, Validation Rules, Process Builder, Flows, and Email Alerts. The goal of automation is to enhance accuracy, consistency, and reduce manual intervention in delivery routing, driver assignments, and notifications.

Workflow Rules

Workflow Rules in Salesforce automate standard internal procedures and processes. For this project, a workflow rule named 'Set_InTransit_When_Route_Assigned' was created on the Delivery Order object. The rule automatically updates the delivery status to 'In Transit' when a route is assigned. It ensures that once a delivery is linked with a route, its operational stage reflects accurately without manual updates.



Validation Rules

Validation Rules are used to maintain data integrity by preventing incorrect data from being saved. In this project, a rule named 'Require_Geocode' was created to ensure that both Latitude and Longitude values are entered before saving a Delivery Order record. This prevents incomplete data from affecting route optimization and geolocation-based tracking.

The screenshot shows the Salesforce Setup interface under the Object Manager section for the Delivery Order object. The Validation Rules section is selected. A single validation rule is listed:

Rule Name	Error Location	Error Message	Active	Modified By
Require_Geocode	Latitude	Please provide both Latitude and Longitude before saving	✓	KORAPALA ABHINAYASRI, 10/18/2025, 10:32 PM

Record-Triggered Flow

A Record-Triggered Flow was designed to automatically create a Route Stop record whenever a new Delivery Order is created. This automation helps reduce manual data entry and ensures that each delivery is immediately linked to its respective route stop. The flow sets key field values such as Route, Delivery Order, Sequence, Status, and Remarks.

The screenshot shows the Flow Builder interface with a flow titled "Record-Triggered Flow". The flow consists of the following steps:

```

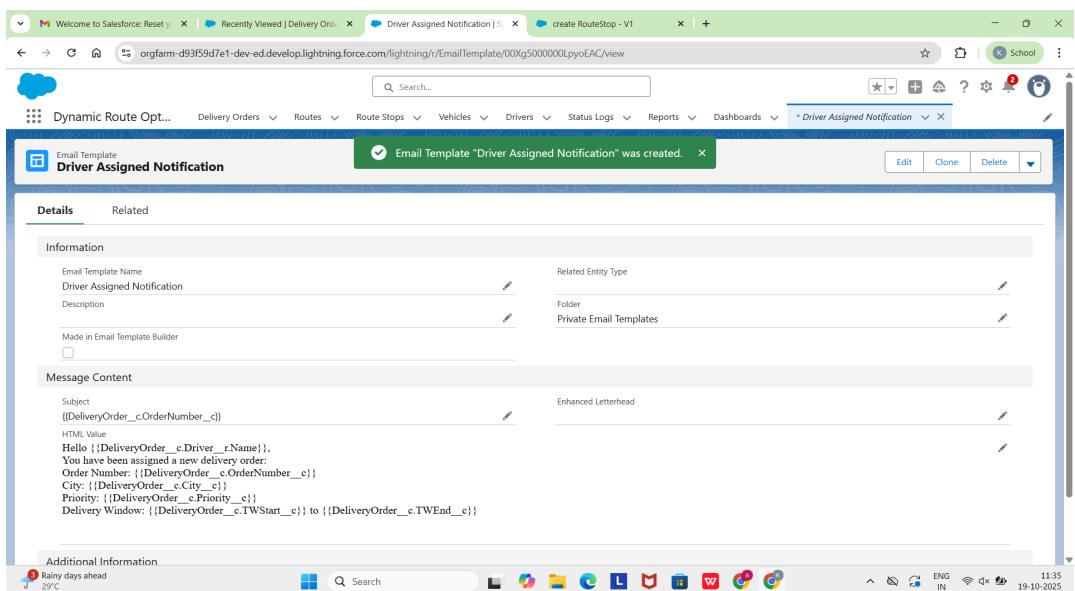
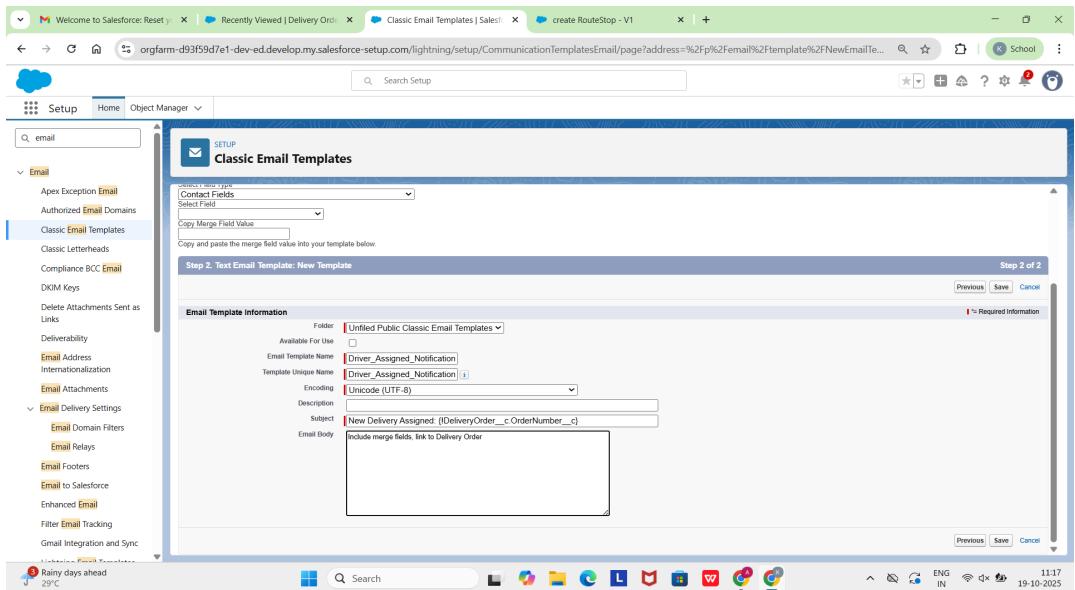
graph TD
    Start((Record-Triggered Flow Start)) --> Run[Run Immediately]
    Run --> CreateRouteStop[Create RouteStop]
    CreateRouteStop --> End((End))
    
```

The "Create RouteStop" step is currently selected, as indicated by a blue highlight. To the right of the flow, there is a configuration panel for the "Create Records" action:

- Create Records** (Action Type: Manually)
- Create a Record of This Object** (Object: Route Stop)
- Set Field Values for the Route Stop**
- Field: `Route` (Value: `# Triggering_Delivery_Order_c > Route`)
- Field: `Delivery Order` (Value: `# Triggering_Delivery_Order_c > Record ID`)
- Field: `Sequence` (Value: `1`)
- Field: `Status` (Value: `NotStarted`)
- Field: `Remarks / Notes` (Value: `"Auto-created from Delivery Order"`)

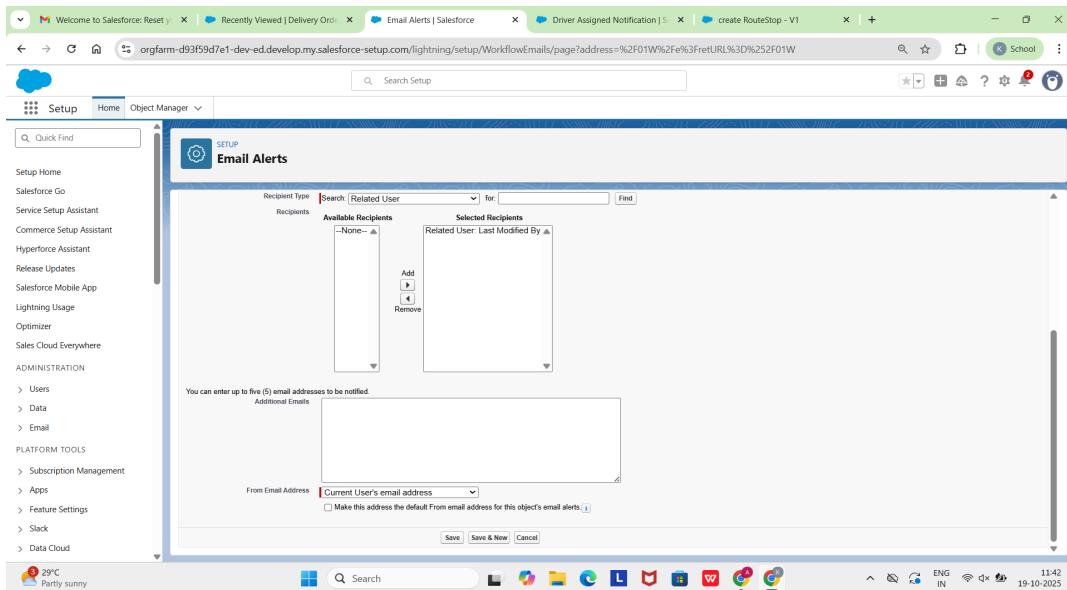
Email Templates

Salesforce Email Templates are used to send structured, professional notifications automatically. In this project, both Classic and Lightning Email Templates were created. The 'Driver_Assigned_Notification' template sends delivery assignment details to drivers, including order number, city, priority, and delivery window. This helps maintain clear communication between dispatchers and drivers.



Email Alerts

An Email Alert was configured to automatically notify drivers once a delivery is assigned. This alert uses the 'Driver_Assigned_Notification' email template and is triggered by the workflow rule. It ensures that the assigned driver receives an instant notification with all relevant delivery details, improving communication and operational efficiency.



Automation Summary Table

Automation Type	Name	Object	Purpose
Workflow Rule	Set_InTransit_When_Route_Assigned	Delivery Order	Auto-update status when route assigned
Validation Rule	Require_Geocode	Delivery Order	Ensure Latitude & Longitude before save
Flow	Create_RouteStop	Route Stop	Auto-create route stops on new delivery
Email Template	Driver_Assigned_Notification	Delivery Order	Notify driver about assigned delivery
Email Alert	Driver_Assigned_Notification	Delivery Order	Send automatic email to driver

Conclusion

The automation features implemented in Salesforce significantly streamlined the operations of the Dynamic Route Optimization system. By leveraging workflow rules, flows, and email alerts, the organization achieved faster task execution, reduced manual errors, and improved real-time coordination between delivery staff and managers. These automated processes ensure scalability and efficiency across all logistics operations.

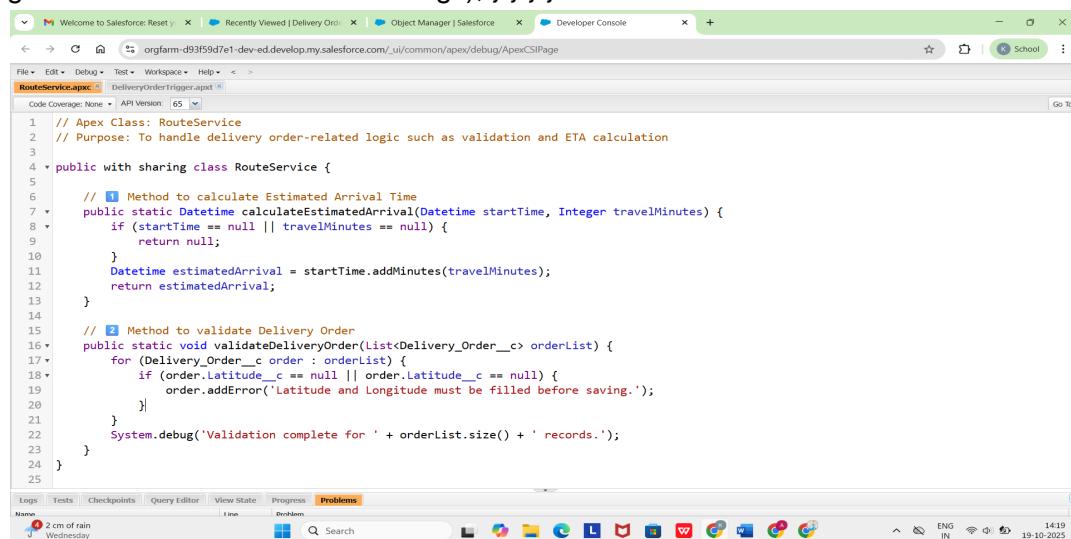
Dynamic Route Optimization — Phase 5 Apex Development

This document provides a detailed explanation of Apex programming concepts implemented in Phase 5 of the Dynamic Route Optimization project. It includes Apex Classes, Triggers, SOQL/SOSL queries, and asynchronous Apex such as Batch, Queueable, Scheduled, and Future methods, along with Exception Handling examples. Each section contains explanations, sample code, and corresponding screenshots.

1. Apex Classes & Objects

Apex Classes are used to encapsulate business logic. In this project, the RouteService class performs calculations such as estimating delivery times and validating delivery orders. It ensures that key fields like Latitude and Longitude are not left empty before saving the record.

```
public with sharing class RouteService { public static Datetime calculateEstimatedArrival(Datetime startTime, Integer travelMinutes) { if (startTime == null || travelMinutes == null) return null; return startTime.addMinutes(travelMinutes); } public static void validateDeliveryOrder(List<Delivery_Order__c> orderList) { for (Delivery_Order__c order : orderList) { if (order.Latitude__c == null || order.Longitude__c == null) { orderaddError('Latitude and Longitude must be filled before saving.'); } } }
```

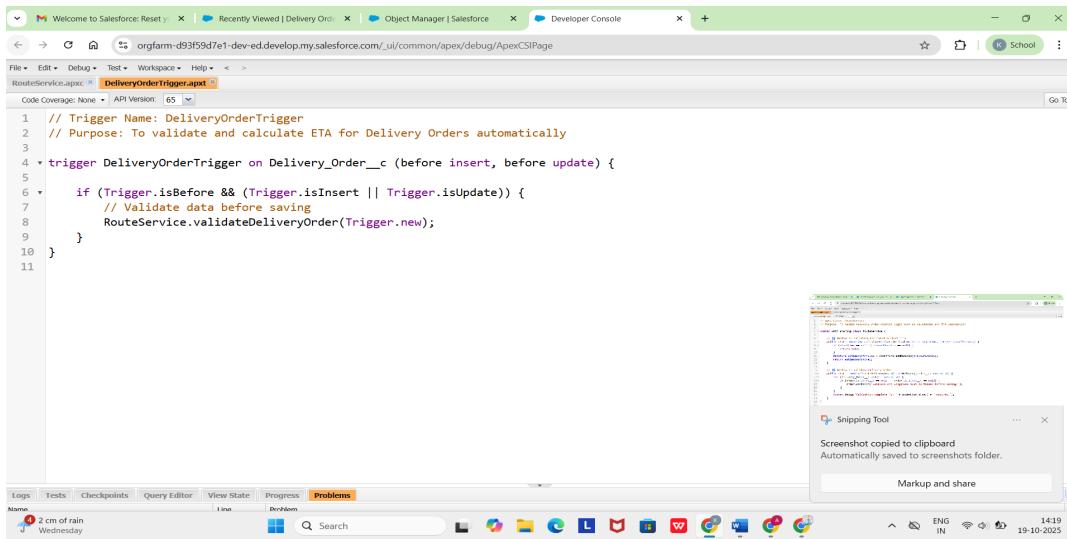


The screenshot shows the Salesforce Developer Console interface. The top navigation bar includes 'Welcome to Salesforce: Reset', 'Recently Viewed | Delivery Ord...', 'Object Manager | Salesforce', and 'Developer Console'. Below the navigation is a toolbar with 'File', 'Edit', 'Debug', 'Test', 'Workspace', 'Help', and a search bar. The main area displays the 'RouteService.apxc' file content. The code is annotated with comments explaining the purpose of each method: 'To handle delivery order-related logic such as validation and ETA calculation'. The 'calculateEstimatedArrival' method takes a start time and travel minutes, returning null if either is null. The 'validateDeliveryOrder' method iterates through a list of delivery orders, adding an error message to each if its latitude or longitude is null. A System.debug statement is present at the end of the validation loop. The bottom of the screen shows the browser's taskbar with various icons and the date/time '19-10-2025 14:19'.

2. Apex Triggers

Apex Triggers automate actions before or after database operations. The DeliveryOrderTrigger calls the RouteService class to validate data before inserting or updating records.

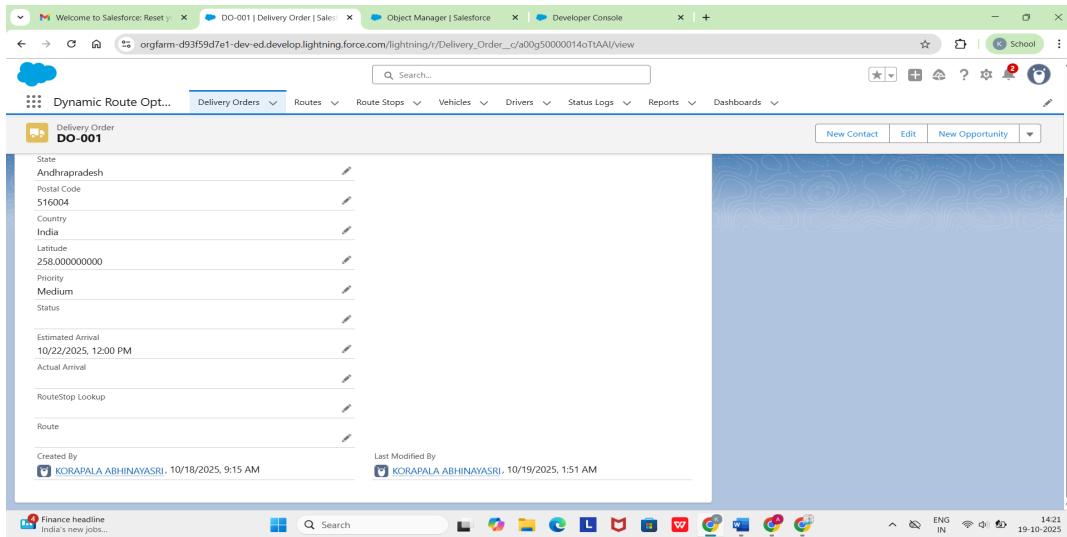
```
trigger DeliveryOrderTrigger on Delivery_Order__c (before insert, before update) { if
(Trigger.isBefore && (Trigger.isInsert || Trigger.isUpdate)) {
RouteService.validateDeliveryOrder(Trigger.new); } }
```



3. SOQL & SOSL Queries

SOQL (Salesforce Object Query Language) retrieves records from objects, while SOSL searches text across multiple objects. Below is a simple query example to fetch assigned Delivery Orders.

```
List orders = [ SELECT Id, Name, Status__c FROM Delivery_Order__c WHERE
Status__c = 'Assigned' ]; System.debug(orders);
```



4. Batch Apex

Batch Apex is designed for handling large data volumes asynchronously. It processes records in manageable batches and executes logic efficiently.

```
global class DeliveryOrderBatch implements Database.Batchable {
    Database.QueryLocator start(Database.BatchableContext bc) { return
```

```

Database.getQueryLocator('SELECT Id, Status__c FROM Delivery_Order__c'); } global
void execute(Database.BatchableContext bc, List scope) { for (Delivery_Order__c order : scope) { order.Status__c = 'Assigned'; } update scope; } global void
finish(Database.BatchableContext bc) { System.debug('Batch Process Completed.'); } }

```

The screenshot shows the Salesforce Developer Console interface. The top navigation bar includes tabs for Welcome to Salesforce, Reset, DO-001 | Delivery Order | Sales, Object Manager | Salesforce, and Developer Console. The main area has tabs for Logs, Tests, Checkpoints, Query Editor (which is selected), View State, Progress, and Problems. The Query Editor tab contains the Apex code for the batch process. The Execution Log pane on the left shows timestamped events and details for each operation. The bottom status bar shows the date (19-10-2025), time (14:36), and weather (30°C Partly sunny).

5. Queueable Apex

Queueable Apex provides an easy way to run asynchronous jobs. It allows for more complex logic and chaining of multiple jobs. Here, we update delivery orders to 'In Transit' when queued.

```

public class DriverAssignmentQueueable implements Queueable { public void
execute(QueueableContext context) { List orders = [SELECT Id, Status__c FROM
Delivery_Order__c WHERE Status__c = 'Assigned']; for (Delivery_Order__c o : orders) {
o.Status__c = 'In Transit'; } update orders; } }

```

The screenshot shows the Salesforce Developer Console interface. The top navigation bar includes tabs for Welcome to Salesforce, Reset, DO-001 | Delivery Order | Sales, Object Manager | Salesforce, and Developer Console. The main area has tabs for Logs, Tests, Checkpoints, Query Editor (which is selected), View State, Progress, and Problems. The Query Editor tab contains the Apex code for the queueable batch process. The Execution Log pane on the left shows timestamped events and details for each operation. The bottom status bar shows the date (19-10-2025), time (14:46), and weather (30°C Rainy days ahead).

6. Scheduled Apex

Scheduled Apex allows you to run classes at specific times automatically. The DailyDeliveryScheduler executes the Batch Apex daily to process delivery orders.

```
global class DailyDeliveryScheduler implements Schedulable { global void execute(SchedulableContext sc) { Database.executeBatch(new DeliveryOrderBatch(), 100); } }
```

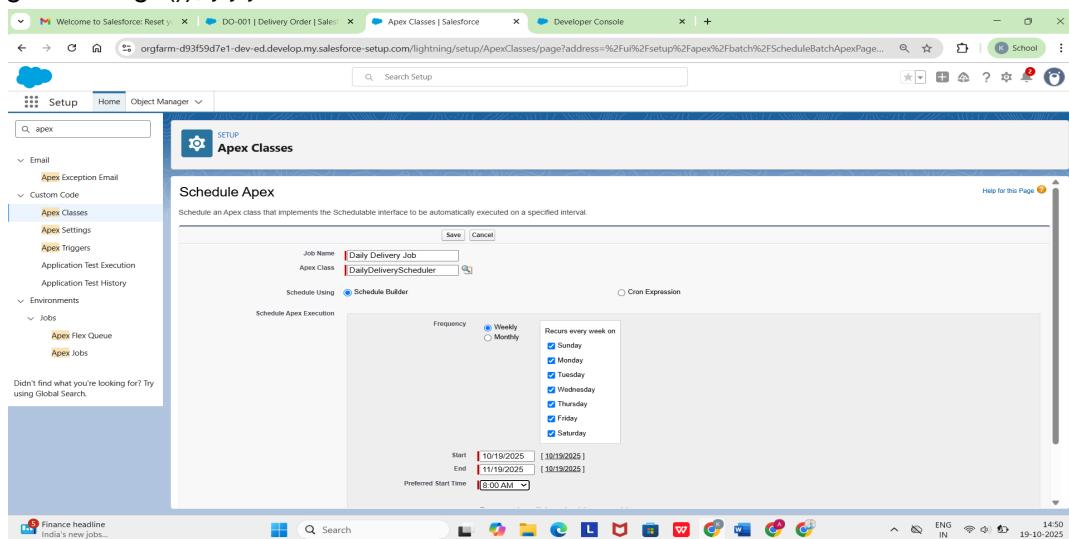
```
1 public class DriverAssignmentQueueable implements Queueable {
2
3     public void execute(QueueableContext context) {
4         List<Delivery_Order__c> orders = [SELECT Id, Status__c FROM Delivery_Order__c WHERE Status__c = 'Assigned'];
5
6         for (Delivery_Order__c o : orders) {
7             o.Status__c = 'In Progress';
8         }
9
10        update orders;
11        System.debug('DriverAssignmentQueueable');
12    }
13 }
14
```

The screenshot shows the Salesforce Developer Console with the code editor open. A modal window titled "Enter Apex Code" contains the code above. Below the code editor is a table titled "Logs" showing the execution history of the class. The logs table has columns: User, Application, Operation, Time, Status, Read, and Size. The log entries show successful executions of the ApexTestHandler and Batch Apex methods.

7. Future Methods & Exception Handling

Future methods allow asynchronous operations like sending notifications or external calls without blocking the main execution. Exception handling ensures smooth error management.

```
public class DeliveryNotificationService { @future public static void sendEmailAsync(String email, String message) { System.debug('Email sent to: ' + email + ' | Message: ' + message); } }
public class DeliveryUpdateService { public static void updateOrderStatus(Delivery_Order__c order, String newStatus) { try { order.Status__c = newStatus; update order; } catch (DmlException e) { System.debug('Error updating order: ' + e.getMessage()); } } }
```



8. Summary

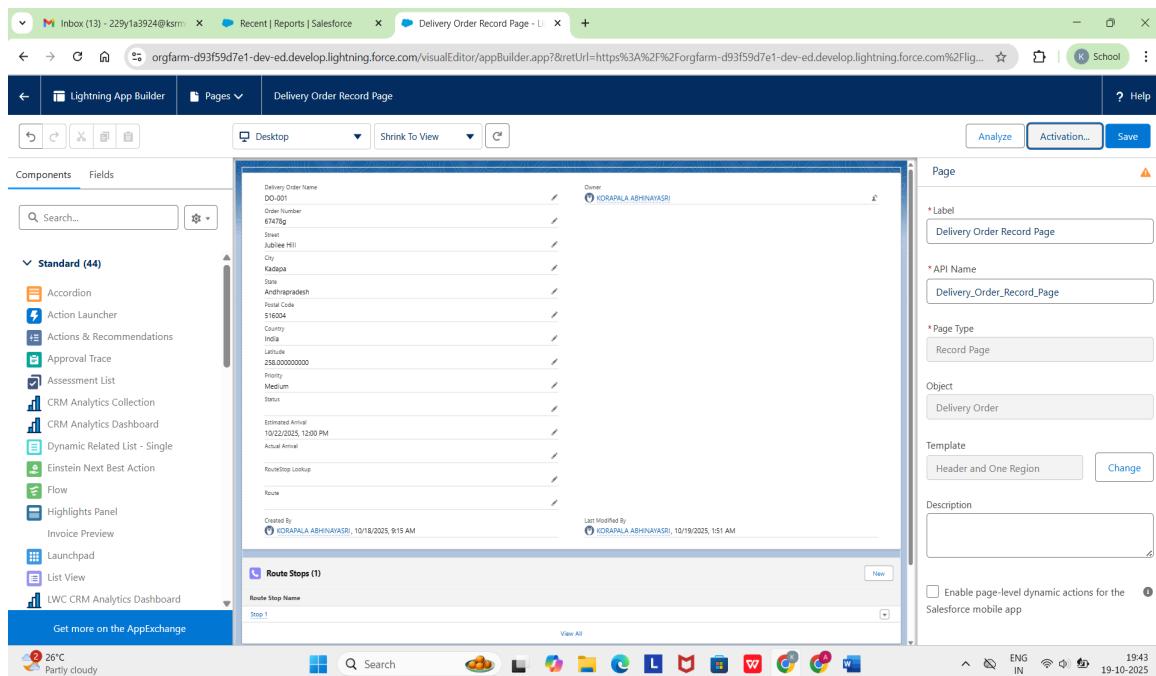
This phase covered the implementation of core Apex programming concepts such as Classes, Triggers, SOQL/SOSL, and asynchronous Apex features. The integration of Batch, Queueable, and Scheduled Apex provides efficient background processing. Exception handling and future methods add reliability and performance to the project.

Dynamic Route Optimization — Phase 6: User Interface Development

In this phase, we focus on building the user interface for the Dynamic Route Optimization application using Salesforce's Lightning App Builder and Lightning Web Components (LWC). The goal is to create an interactive, modern interface that displays delivery orders, route details, and related information in real time. This phase combines Salesforce declarative tools like record pages and tabs with custom LWC and Apex backend logic.

Lightning App Builder and Record Pages

To begin, open Setup in your Salesforce Developer Org and search for "App Builder." Choose Lightning App Builder and click on "New → Record Page." Enter the label as "Delivery Order Record Page" and select the Delivery Order object. Choose the "Header and One Region" template and click Finish. On this page, you can customize what fields and components are visible. Add sections such as delivery details, route stop information, and estimated arrival time. Once configured, click Save and then Activation to make the page available for users.



Tabs, Home Page Layouts, and Utility Bar

Next, navigate to Setup → Tabs, and create tabs for your custom objects such as Delivery Orders, Routes, and Route Stops. These tabs allow users to quickly access each record type from the Salesforce app navigation bar. You can also customize the Home Page Layout to include key performance metrics, quick links, and charts related to delivery performance. Add a Utility Bar from the App Manager to show tools like Recent Items or Notes at the bottom of the app for quick access.

Lightning Web Component (LWC) Setup

After building the record page, you'll move to the programmatic side using Visual Studio Code (VS Code). Open VS Code and connect it to your Salesforce Org using the Salesforce CLI. Create a Lightning Web Component by pressing Ctrl + Shift + P, selecting SFDX: Create Lightning Web Component, and naming it deliveryOrderList. This component will display delivery orders dynamically. It retrieves data from the Apex backend using both wire adapters (for real-time data refresh) and imperative Apex calls (for user-triggered actions).

Example LWC JavaScript Code:

```
// deliveryOrderList.js import { LightningElement, wire } from 'lwc'; import getDeliveryOrders from '@salesforce/apex/RouteService.getDeliveryOrders'; export default class DeliveryOrderList extends LightningElement { deliveryOrders; error; @wire(getDeliveryOrders) wiredOrders({ data, error }) { if (data) { this.deliveryOrders = data; this.error = undefined; } else if (error) { this.error = error; this.deliveryOrders = undefined; } } }
```

Example LWC HTML Code:

```
{error}
```

Apex Integration with LWC

To support this component, create an Apex class named RouteService in your Salesforce Developer Console or VS Code. This class should include the following method to fetch delivery orders dynamically:

```
public with sharing class RouteService { @AuraEnabled(cacheable=true) public static List<Delivery_Order__c> getDeliveryOrders() { return [SELECT Id, OrderNumber__c, City__c, Status__c FROM Delivery_Order__c LIMIT 10]; } }
```

Deploying and Adding LWC to Record Page

Once both the LWC and Apex class are ready, deploy them to your org using Ctrl + Shift + P → SFDX: Deploy Source to Org. After deployment, go back to Lightning App Builder, open the Delivery Order Record Page, and drag your new deliveryOrderList component from the Custom section onto the page layout. Click Save and Activate. Now, you can see a dynamic table showing all delivery orders directly on your record page.

Testing the User Interface

After publishing, open the Delivery Orders tab and select a record. You should now see your custom Lightning Web Component along with all standard fields. Verify that changes in the database reflect automatically in the LWC table. If there are issues such as missing data or access errors, check your Apex permissions and debug logs. This integration ensures that your users can interact with live delivery order data within an intuitive, responsive interface built using Lightning Web Components and Salesforce's Lightning framework.

Phase 7: Integration & External Access

This phase covers how Salesforce integrates with external systems, secures API access, and streams data changes in real time. Below are concise descriptions and setup guidance for each integration capability you'll commonly use when connecting Salesforce to other platforms.

Named Credentials

Named Credentials centralize the URL and authentication details for external endpoints so that callouts can use them securely without hard-coding credentials. Create a Named Credential in Setup > Named Credentials, set the endpoint URL, choose an authentication provider or username/password, and reference it from Apex HTTP callouts or external data sources. This simplifies credential rotation and enforces secure storage of secrets. Use Named Credentials wherever possible instead of embedding auth tokens in code.

External Services

External Services lets admins import an OpenAPI (Swagger) definition or schema for an external REST API and register its actions inside Salesforce. After registering, Flow Builder and Apex can call those external actions declaratively. To configure: upload an OpenAPI spec under Setup > External Services, provide authentication (Named Credential), and then use the generated actions in Flows or Apex. This reduces custom coding for predictable API operations and makes maintenance easier.

Web Services (REST / SOAP)

Salesforce supports both REST and SOAP web services. You can build Apex REST endpoints (via @RestResource) to expose data to external systems, or consume external SOAP/REST services via Apex callouts. For SOAP, generate Apex classes from WSDL (Setup > Apex Classes > Generate from WSDL). For REST, use HttpRequest/Http classes or register external services. Always secure endpoints using OAuth, named credentials, or IP restrictions.

Callouts

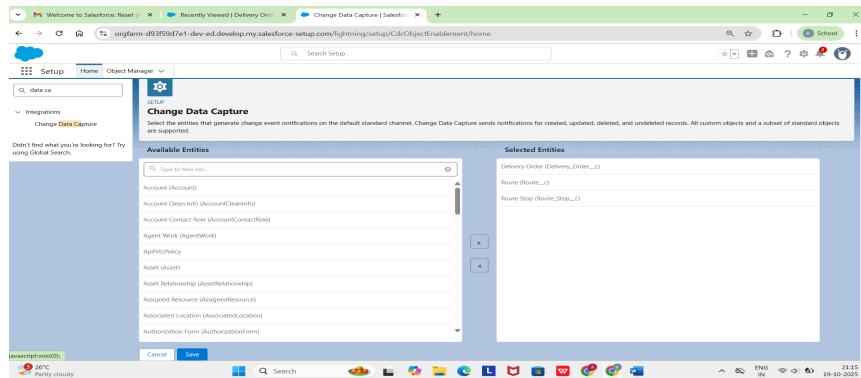
Callouts are outbound HTTP/HTTPS requests Salesforce makes to external services. Implement callouts in Apex using Http, HttpRequest, and HttpResponse classes. Callouts can be synchronous (from Apex) or asynchronous (Queueable, Future). Ensure the target URL is allowed in Remote Site Settings or use Named Credentials to avoid permission errors. Remember governor limits and always handle timeouts and retries sensibly.

Platform Events

Platform Events provide a publish/subscribe model within Salesforce and to external systems. Define an event object, publish events from Apex, Flows, or external systems, and subscribe via CometD, Apex triggers, or external subscribers. Use Platform Events for decoupled integrations requiring near-real-time communication (e.g., notify an external dispatcher when a route changes). They are resilient and scalable for event-driven integration patterns.

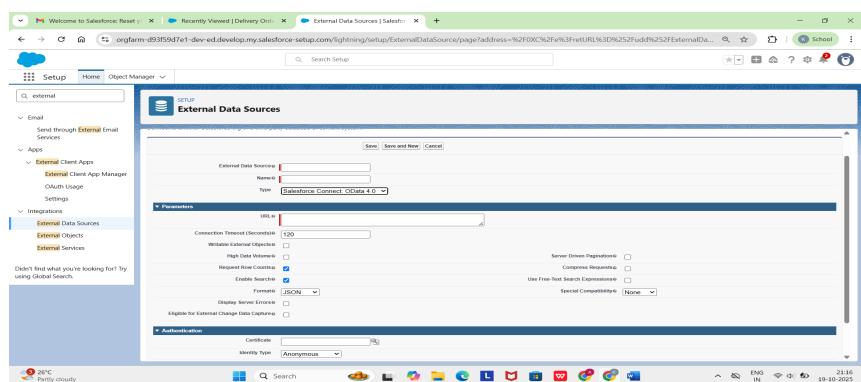
Change Data Capture (CDC)

Change Data Capture streams data change events (create/update/delete/undelete) for selected objects on the standard channel. Consumers (external systems, middleware) can subscribe and react to changes in near real-time. Enable CDC in Setup > Change Data Capture and pick the objects (e.g., Delivery Order, Route, Route Stop). Use CDC for synchronizing systems without polling; it's efficient and reduces integration complexity.



Salesforce Connect

Salesforce Connect surfaces external data in Salesforce using external objects (no data storage in Salesforce). It supports OData and other adapters. To configure, go to Setup > External Data Sources and create a new External Data Source (Salesforce Connect: OData 4.0). Set the URL and authentication (Named Credential). After validation, create External Objects and use them in record pages and reports like native objects.



API Limits

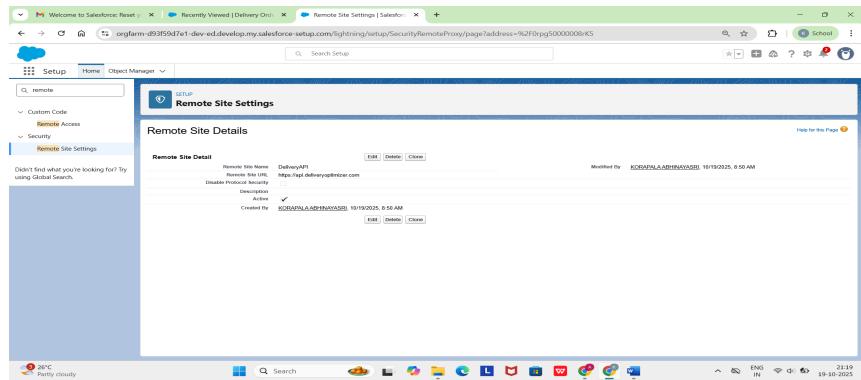
Salesforce enforces API usage limits (daily API calls, concurrent long-running requests) depending on edition and licenses. Monitor limits in Setup > System Overview or via REST endpoints. Design integrations to minimize API consumption: use Bulk API for large imports, Change Data Capture for change-based sync, and caching or batching where appropriate. Handle HTTP 429 / 503 responses gracefully with backoff.

OAuth & Authentication

OAuth is the recommended approach to authenticate external clients and services with Salesforce. Use connected apps (Setup > App Manager) to obtain client id/secret and configure OAuth flows (Web Server, JWT, Username-Password, etc.). For server-to-server integrations, JWT Bearer Token flow or OAuth with refresh tokens is common. Always scope permissions to least privilege and rotate credentials regularly.

Remote Site Settings

Remote Site Settings is a legacy approach to allow Apex callouts by registering external endpoints (Setup > Remote Site Settings). Enter the remote URL and activate it so Apex HttpRequest calls succeed. If using Named Credentials, Remote Site Settings is not required; Named Credentials encapsulate the endpoint and optional auth for safer callouts.



Conclusion

These integration features form a robust toolkit for connecting Salesforce with external applications. Combine change streaming (CDC, Platform Events), secure authentication (Named Credentials, OAuth), and efficient data access (Salesforce Connect) to build scalable, secure, and maintainable integrations. Always account for API limits, secure credentials, and prefer declarative options when available.

Phase 8: Data Management & Deployment

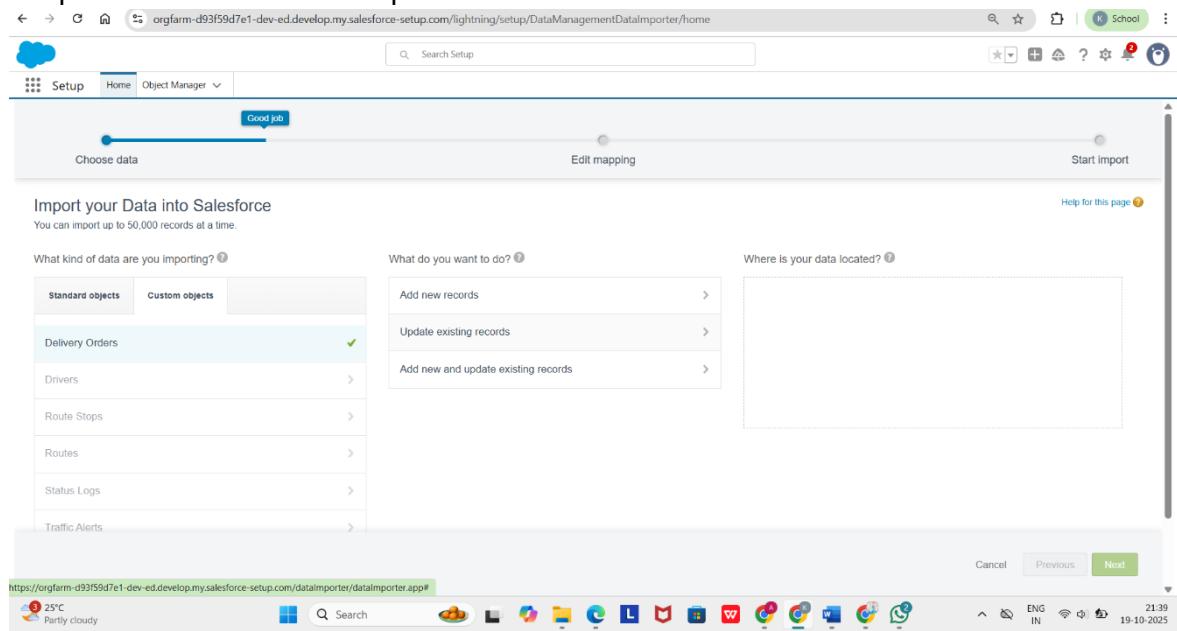
Objective: In this phase, we focus on managing data efficiently and deploying customizations from one Salesforce environment to another. This ensures smooth migration, data consistency, and secure backups, forming the foundation for stable and scalable CRM operations.

1. Data Import Wizard

Purpose: Simplifies importing data such as Accounts, Contacts, Leads, or Custom Object records directly through Salesforce UI.

- 1 Navigate to Setup → Data Import Wizard → Launch Wizard.
- 2 Choose the object (e.g., Account, Contact, or Custom Object).
- 3 Select the operation — Add new, Update existing, or Upsert.
- 4 Upload your CSV file and map fields correctly.
- 5 Run the import and review the success/error files.

Use Case: Ideal for smaller datasets under 50,000 records. Best used for quick imports where mapping and deduplication are simple. It is helpful when end-users or admins need to upload data without developer assistance.



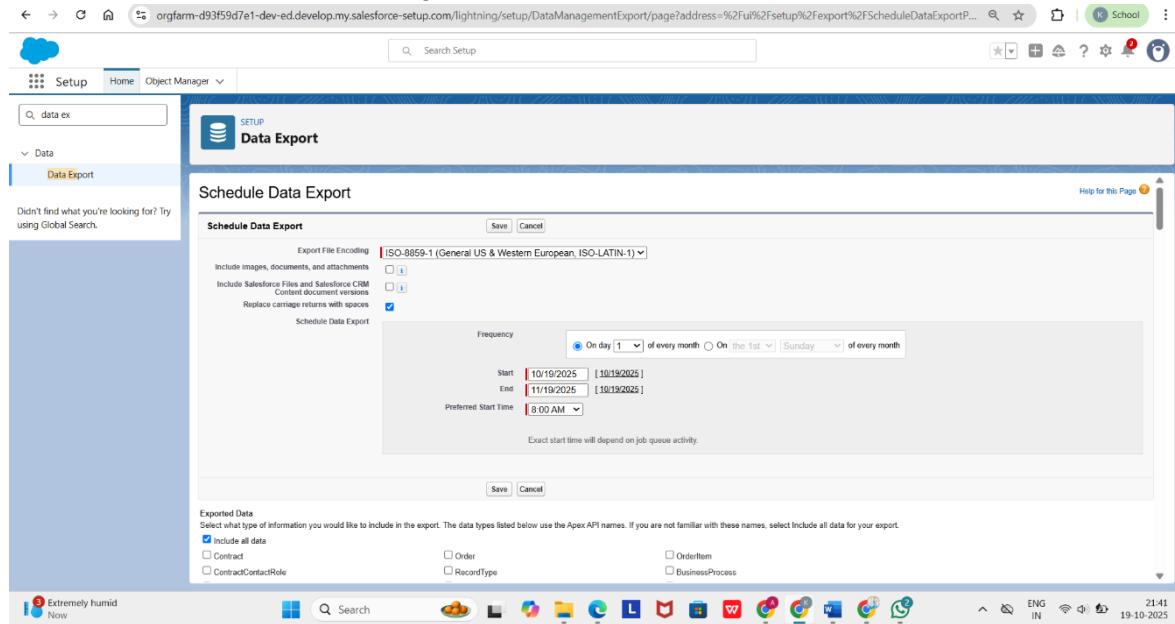
2. Data Loader

Purpose: A client application for bulk import, export, update, or delete operations.

- 1 Download and install Salesforce Data Loader.

- 2 Login using Salesforce credentials or OAuth.
- 3 Choose the operation (Insert, Update, Upsert, Delete, Export).
- 4 Select the CSV file and map fields.
- 5 Run the operation and review success/error logs.

Use Case: Used when large volumes of data need to be migrated, such as initial data loads, mass updates, or regular data synchronization. It allows automation and provides better control over error handling and performance.



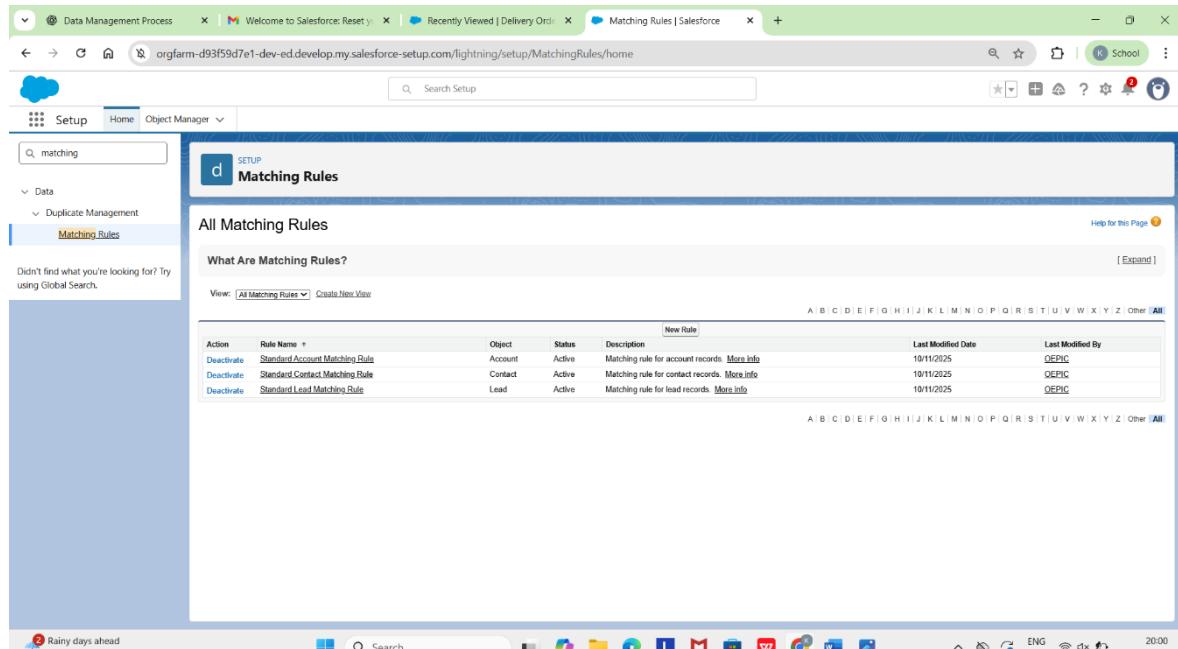
3. Duplicate & Matching Rules

Purpose: Prevent creation of duplicate records and maintain clean data.

- 1 Go to Setup → Matching Rules → New and define matching criteria (e.g., Email, Phone).
- 2 Create a Duplicate Rule using the matching rule and set behavior (Block, Alert, or Report).
- 3 Activate both rules.
- 4 Test by creating records to ensure duplicates are flagged.

Use Case: Keeps CRM data clean by automatically identifying potential duplicates during record creation or update. This improves data integrity, reporting accuracy, and customer

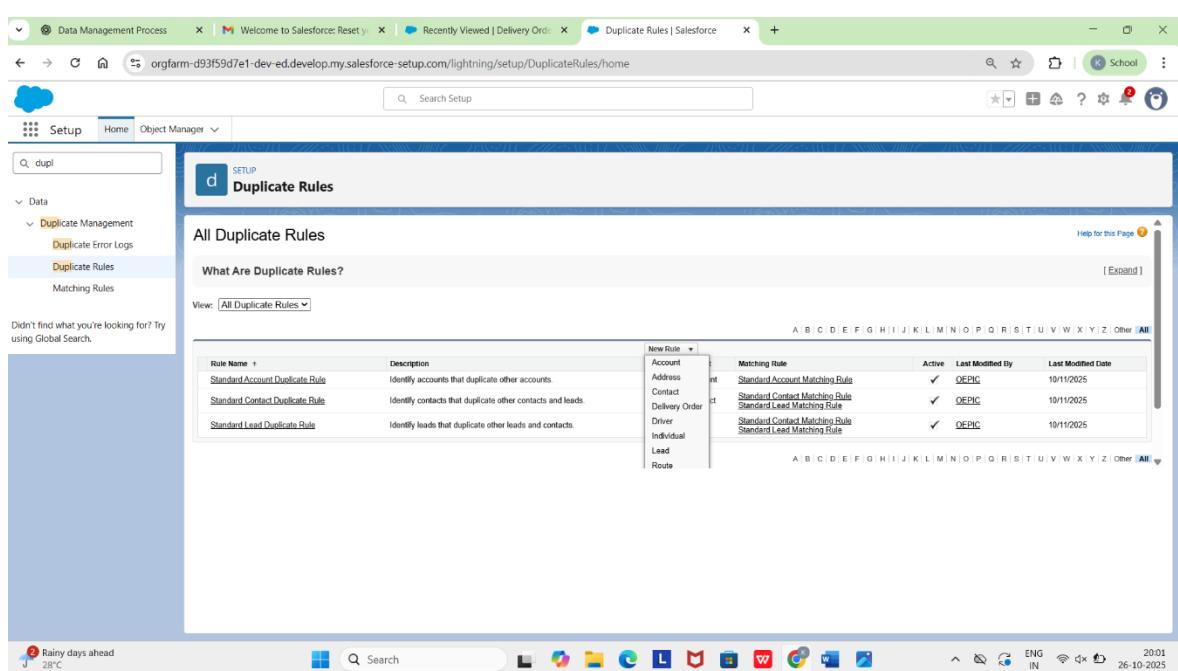
satisfaction.



The screenshot shows the Matching Rules page in the Salesforce Setup. The sidebar has a search bar for 'matching' and a 'Matching Rules' section under 'Duplicate Management'. The main area displays three standard matching rules:

Action	Rule Name	Object	Status	Description	Last Modified Date	Last Modified By
Deactivate	Standard Account Matching Rule	Account	Active	Matching rule for account records. More info	10/11/2025	OEPIC
Deactivate	Standard Contact Matching Rule	Contact	Active	Matching rule for contact records. More info	10/11/2025	OEPIC
Deactivate	Standard Lead Matching Rule	Lead	Active	Matching rule for lead records. More info	10/11/2025	OEPIC

Below the table are navigation links for letters A through Z and an 'All' link. The status bar shows it's 26-10-2025, 20:00, ENG IN, and there are 28°C and Rainy days ahead icons.



The screenshot shows the Duplicate Rules page in the Salesforce Setup. The sidebar has a search bar for 'duplic' and a 'Duplicate Rules' section under 'Duplicate Management'. The main area displays four standard duplicate rules:

Rule Name	Description	Account	Address	Contact	Delivery Order	Driver	Individual	Lead	Route	Matching Rule	Active	Last Modified By	Last Modified Date
Standard Account Duplicate Rule	Identify accounts that duplicate other accounts.	✓	✓	✓	✓	✓	✓	✓	✓	Standard Account Matching Rule	✓	OEPIC	10/11/2025
Standard Contact Duplicate Rule	Identify contacts that duplicate other contacts and leads.	✓	✓	✓	✓	✓	✓	✓	✓	Standard Contact Matching Rule	✓	OEPIC	10/11/2025
Standard Lead Duplicate Rule	Identify leads that duplicate other leads and contacts.	✓	✓	✓	✓	✓	✓	✓	✓	Standard Lead Matching Rule	✓	OEPIC	10/11/2025

Below the table are navigation links for letters A through Z and an 'All' link. The status bar shows it's 26-10-2025, 20:01, ENG IN, and there are 28°C and Rainy days ahead icons.

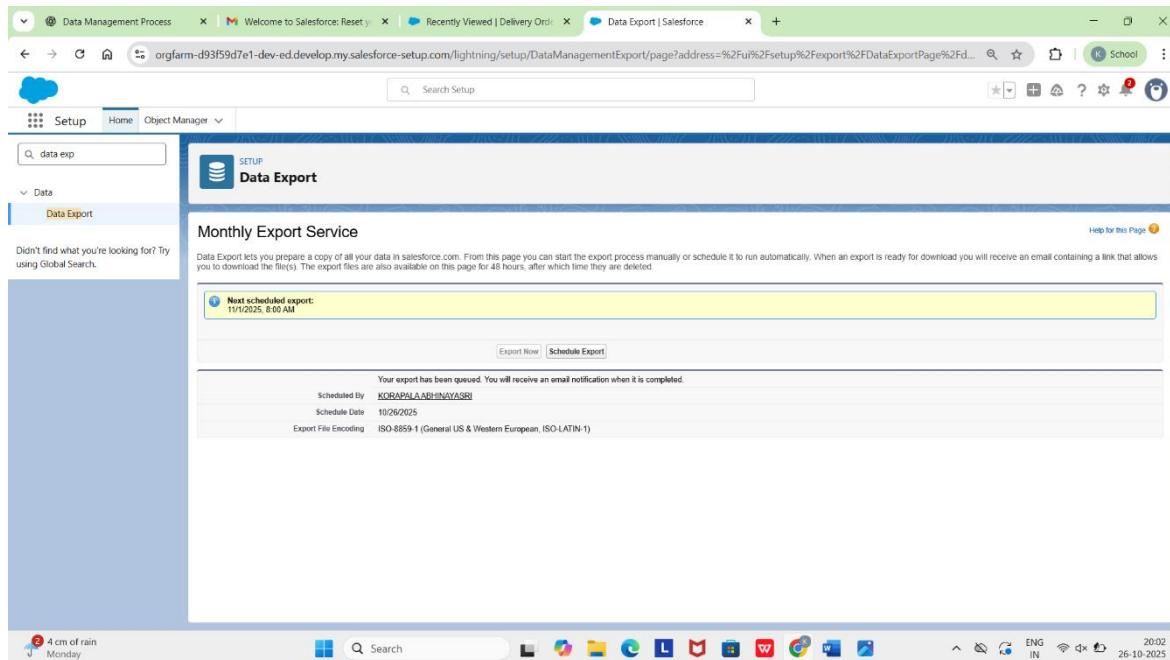
4. Data Export & Backup

Purpose: To ensure data safety and compliance by creating regular backups.

- 1 Navigate to Setup → Data Export.
- 2 Choose objects to include and decide whether to include attachments/files.
- 3 Click Export Now or Schedule Export for weekly/monthly backups.

4 Download the ZIP file once available.

Use Case: Ensures business continuity by maintaining secure backups of all Salesforce data. This process helps restore accidentally deleted records, recover from corruption, and meet data retention or compliance requirements.

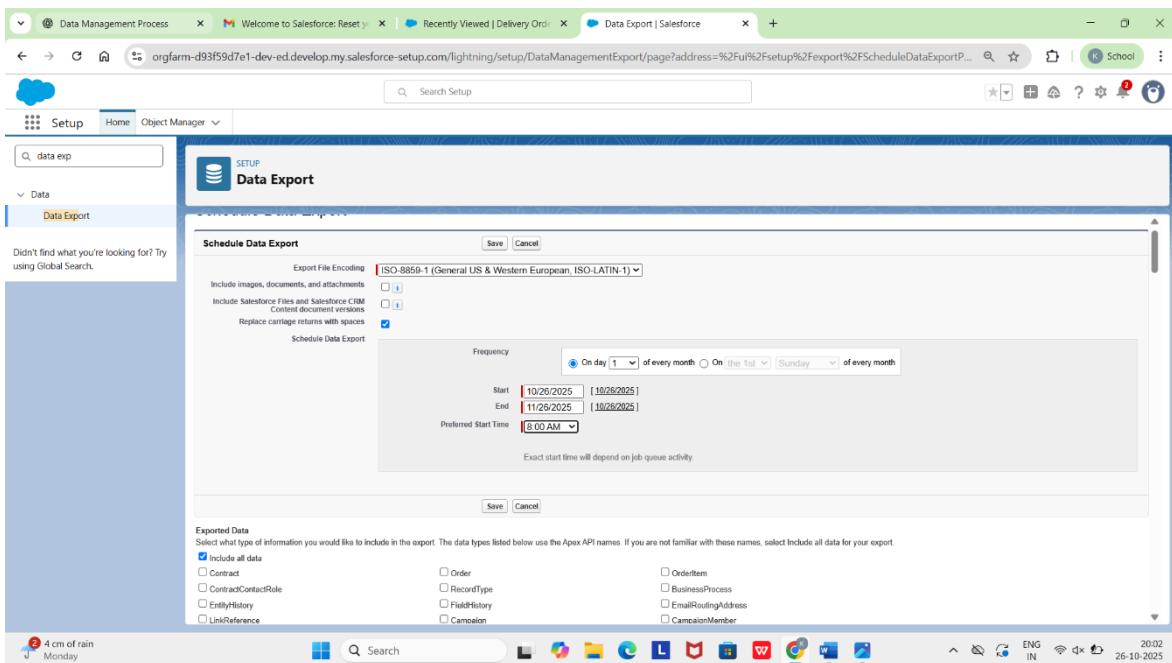


5. Change Sets

Purpose: To deploy metadata (custom objects, triggers, layouts, etc.) between connected Salesforce orgs (e.g., Sandbox → Production).

- 1 In Sandbox, go to Setup → Outbound Change Sets → New.
- 2 Add components (Apex Classes, Page Layouts, etc.).
- 3 Upload to target org.
- 4 In Production, go to Inbound Change Sets, validate, and deploy.

Use Case: Simplifies deployment between environments while maintaining control and traceability. It is ideal for admins and smaller teams who prefer a point-and-click migration method for tested configurations.



6. Managed vs Unmanaged Packages

Purpose: Packaging enables distribution and reusability of Salesforce components.

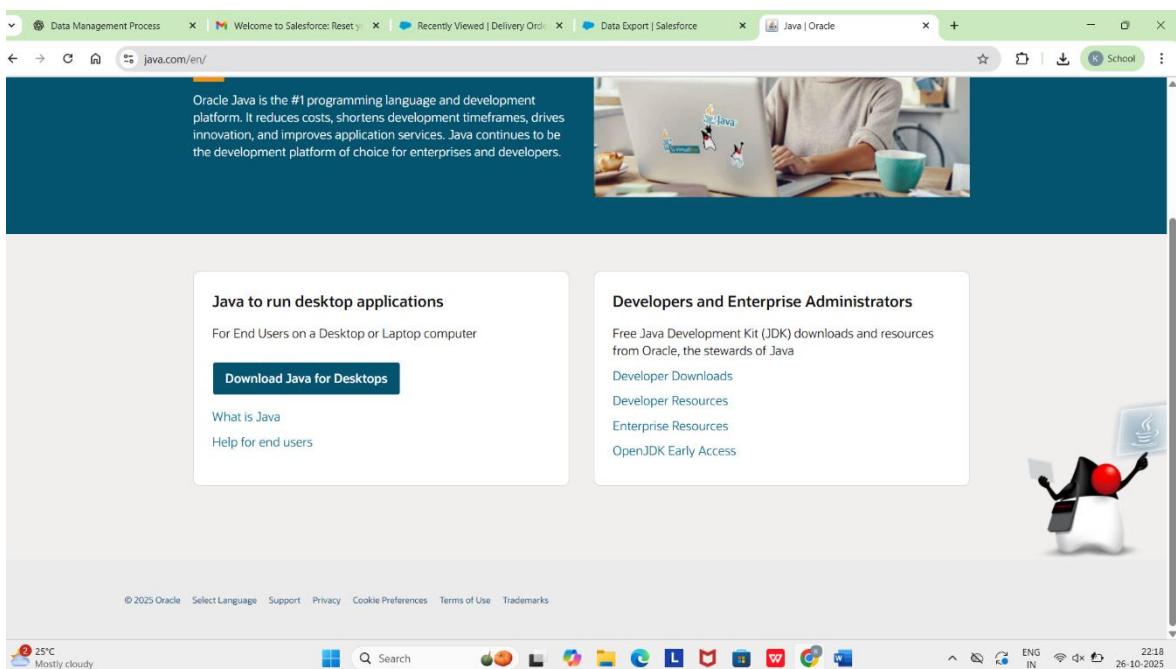
Use Case: Managed packages are used by developers and ISVs to distribute upgradable solutions through AppExchange. Unmanaged packages are used when sharing source code or configurations for learning, customization, or client-specific implementations.

7. ANT Migration Tool

Purpose: A Java-based command-line tool for retrieving and deploying metadata via XML manifests.

- 1 Install Java and Apache Ant.
- 2 Download the Salesforce ANT Migration Tool.
- 3 Configure build.properties with org credentials.
- 4 Edit package.xml to define components.
- 5 Run commands: ant retrieve / ant deploy.

Use Case: Suitable for automated deployments where consistency and repeatability are required. It enables version control integration and bulk metadata transfers across multiple Salesforce environments.

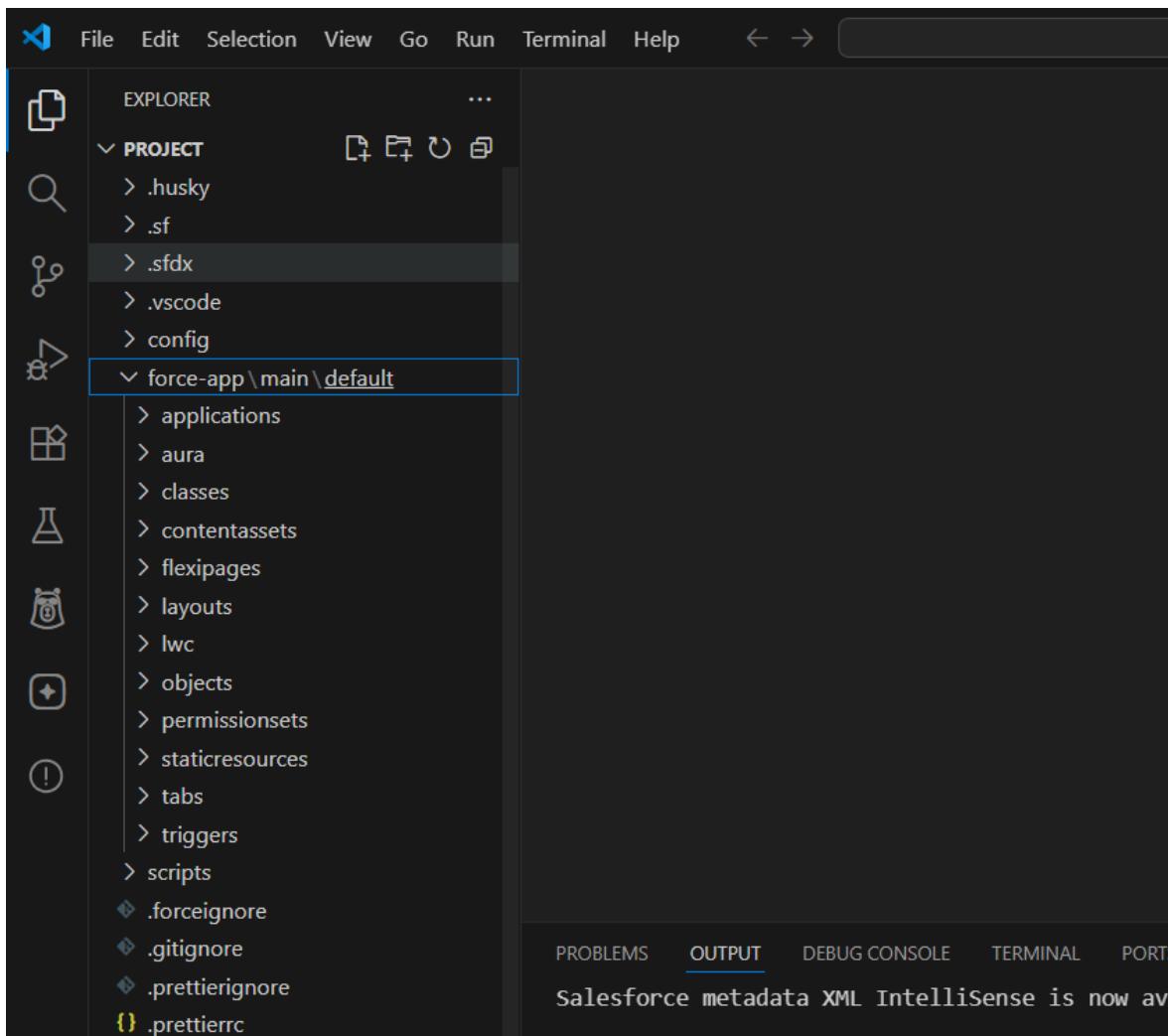


8. Visual Studio Code (VS Code) & Salesforce CLI (SFDX)

Purpose: The modern development and deployment environment for Salesforce developers.

- 1 Install VS Code and Salesforce CLI (SFDX).
- 2 Create a new project using: `sfdx force:project:create -n MyProject`.
- 3 Authorize your org using: `sfdx auth:web:login -a DevHubAlias`.
- 4 Retrieve or deploy metadata using CLI commands.
- 5 Use Git for version control and CI/CD integrations.

Use Case: Provides a unified development platform supporting source tracking, testing, and deployment. Ideal for continuous integration (CI) and continuous delivery (CD) pipelines, improving collaboration among development teams.



Conclusion: Data Management & Deployment are vital for ensuring Salesforce environments remain synchronized, reliable, and secure. By leveraging tools like Data Loader, Change Sets, and SFDX, organizations can automate deployment, prevent errors, and maintain data integrity across all environments.

Salesforce Reports, Dashboards, and Security Review Document

1. Reports (Tabular, Summary, Matrix, Joined)

In this stage, we review and validate the various types of reports configured in the system:
Tabular, Summary, Matrix, and Joined.

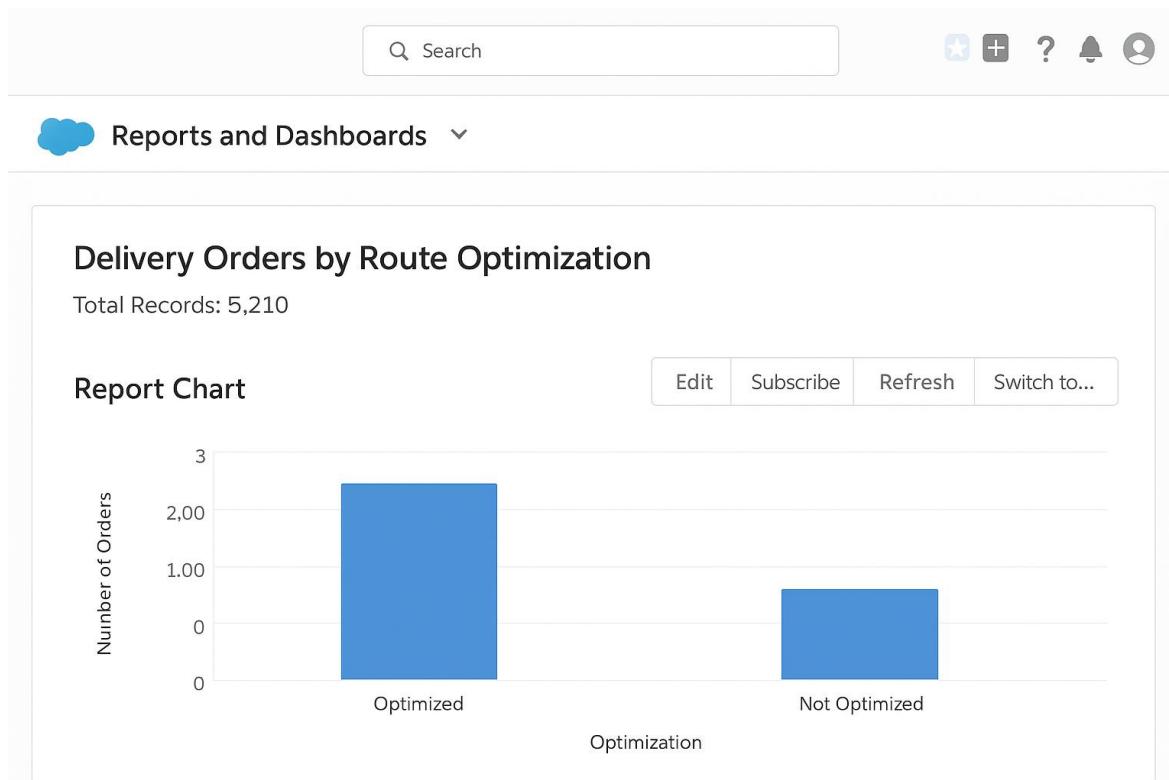
- **Tabular reports** provide flat lists—useful for delivery orders lists and data exports.
- **Summary reports** allow grouping and subtotals (e.g., grouping DeliveryOrders by city or driver).
- **Matrix reports** enable two-dimensional grouping (e.g., number of stops by vehicle by time window).
- **Joined reports** let you combine multiple report types into one (e.g., combining Route, Vehicle, and DeliveryOrder data).

During this phase we:

- Ensure each report type aligns with business metrics (e.g., on-time vs delayed orders).
- Confirm filters, groupings, and charts display correctly for end-users.
- Optimize for performance and ensure correct fields/objects are included.
- Document and classify reports by use-case (Dispatcher, Manager, Executive).

The screenshot shows the Salesforce Reports page. At the top, there are tabs for 'Recent' and 'Reports'. Below the tabs, there's a search bar and buttons for 'New Report' and 'New Folder'. On the left, a sidebar lists categories like 'Recent', 'REPORTS' (with 'Recent' selected), 'FOLDERS', and 'FAVORITES'. The main content area displays a table of recent reports. The columns are 'Report Name', 'Description', 'Folder', 'Created By', 'Created On', and 'Subscribed'. There are two rows visible:

Report Name	Description	Folder	Created By	Created On	Subscribed
Deliveries per Route	Private Reports	KORAPALA ABHINAYASRI	10/18/2025, 9:40 AM		
Sample Flow Report: Screen Flows	Which flows run, what's the status of each interview, and how long do users take to complete the screens?	Public Reports	Automated Process	10/11/2025, 12:04 AM	



2. Report Types

Report Types in Salesforce define which objects and fields are available for reporting.

During this phase we:

- Review existing report types and ensure they include custom objects (DeliveryOrder, RouteStop, Vehicle, etc.).
- Create or modify custom report types to support new use-cases (e.g., “Delivery Orders with Route Stops and Vehicle Assignment”).
- Test each report type for correct object relationships and field availability.
- Document naming conventions and maintain a registry of report types.

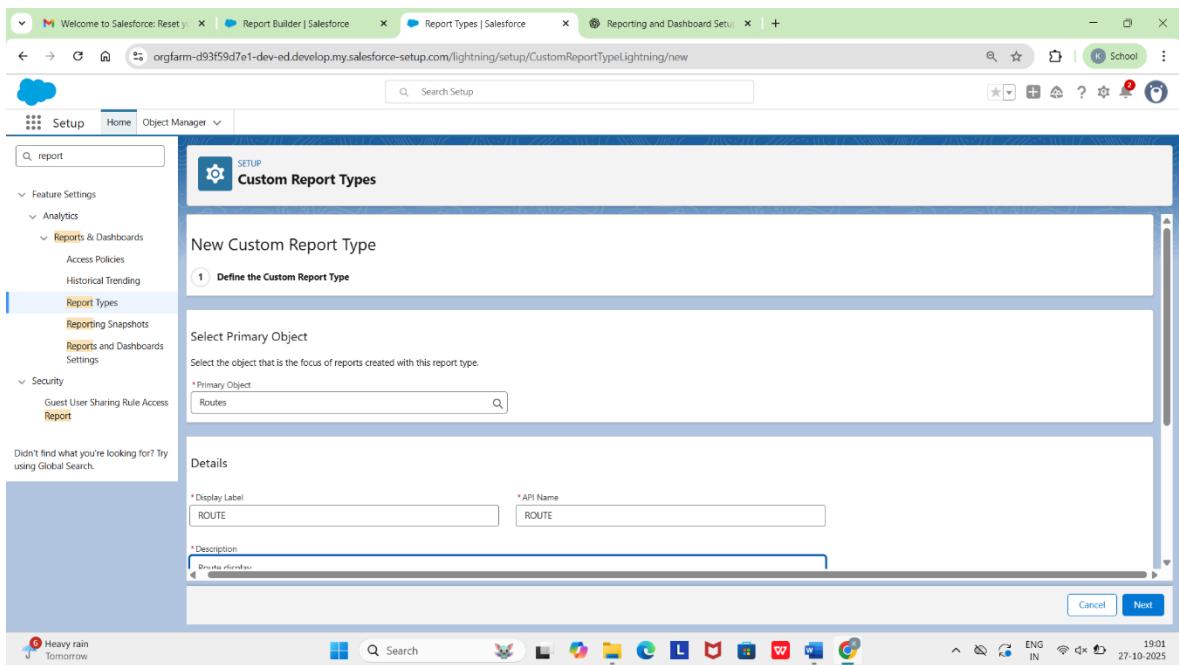
Sample Flow Report: Screen Flows

Flow API Name	Interview Status	Sum of Element Duration in Minutes	Average Element Duration in Minutes	Record Count
Lenovo Vantage	...	No records found		

No records returned in preview. Try running the report or editing report filters.

- Set the Created Date filter to All Time.
- Edit other filters in the filter panel.

Learn More



3. Dashboards

Dashboards visualize key performance indicators.

****During this phase:****

- Consolidate dashboard components (charts, gauges, tables) for routing performance metrics.
- Validate that components reflect accurate data.

- Check layout, color themes, and responsiveness.
- Ensure dashboard folders and naming conventions are standardized.

The screenshot shows the Salesforce Lightning interface with two browser tabs open. The top tab is titled 'New Dashboard | Salesforce' and displays a 'New Dashboard' modal. The modal has fields for 'Name' (set to 'Roote'), 'Description' (set to 'Route'), and 'Folder' (set to 'Private Dashboards'). The bottom tab is also titled 'Roote | Salesforce' and shows a 'Select Report' modal. The left sidebar of the modal lists 'Reports' (Recent, Created by Me, Private Reports, Public Reports, All Reports) and 'Folders' (Created by Me, Shared with Me, All Folders). The main area of the modal shows a search bar and a results section for 'Deliveries per Route' created by 'KORAPALA ABHINAYASRI' on 'Oct 18, 2025, 9:40 AM - Private Reports'. At the bottom right of the modal are 'Cancel' and 'Selected' buttons.

4. Dynamic Dashboards

Dynamic dashboards allow personalized views for users (drivers, dispatchers, managers).

****During this phase:****

- Identify dashboards that should be dynamic and set “Run as Logged-in User”.
- Verify user permissions and sharing settings.
- Review limits (e.g., Developer Edition supports 3 dynamic dashboards).
- Test across profiles to ensure correct data filtering.

5. Sharing Settings

Sharing settings control record visibility across users.

****During this phase:****

- Review OWD for custom objects (DeliveryOrder__c, Route__c, etc.).
- Validate Sharing Rules, Manual Sharing, and Apex Sharing.
- Ensure role hierarchy aligns with business needs.
- Test visibility for roles (driver, manager, operations head).

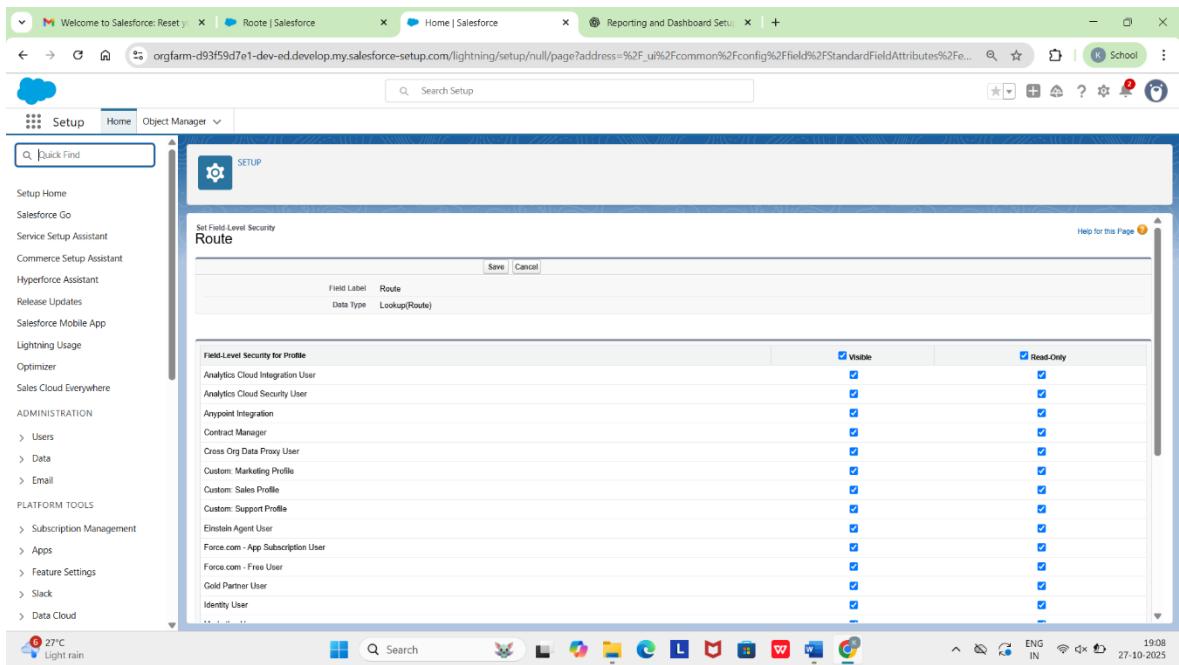
The screenshot shows the Salesforce Sharing Settings page. The URL is orgfarm-d93f59d7e1-dev-ed.develop.my.salesforce-setup.com/lightning/setup/SecuritySharing/page?address=%2Fp%2Fshare%2OrgDefaultSharing%3Fid%3D00Dg50000... . The page title is "Sharing Settings". The main section is titled "Organization-Wide Sharing Defaults Edit". It displays a table with three columns: "Object", "Default Internal Access", and "Default External Access". The "Object" column lists various Salesforce objects like Lead, Account, Contact, Opportunity, Case, etc. The "Default Internal Access" column contains dropdown menus with options like "Public Read/Write/Transfer", "Public Read/Write", "Controlled by Parent", etc. The "Default External Access" column also contains dropdown menus with options like "Private", "Public Read Only", "Public Full Access", etc. There are also checkboxes for "Grant Access Using Hierarchies". A note at the top states: "Edit your organization-wide sharing defaults below. Changing these details will cause all sharing rules to be recalculated. This could require significant system resources and time depending on the amount of data in your organization. Setting an object to Private makes records visible to record owners and those above them in the role hierarchy, and access can be extended using sharing rules." A "Help for this Page" link is also present.

6. Field-Level Security

Defines who can view/edit specific fields.

****During this phase:****

- Hide or restrict sensitive fields (driver contact, vehicle timings).
- Validate mandatory fields are visible and required for relevant users.
- Review and adjust profiles/permission sets.
- Ensure compliance with data policies.



7. Session Settings

Controls login session security.

****During this phase:****

- Verify session timeout (e.g., 30 minutes for mobile users).

- Check IP restrictions for sensitive profiles.
- Confirm MFA and session-based permissions.
- Document exceptions and compensating controls.

8. Login IP Ranges

Defines trusted IP ranges for login.

****During this phase:****

- Review profiles with locked IP ranges.
- Validate restrictions for admin and operations profiles.
- Recommend VPN/secure network access.
- Document exceptions and monitor login anomalies.

The screenshot shows the 'View Setup Audit Trail' page in the Salesforce Setup interface. The page title is 'View Setup Audit Trail'. It displays a table of recent setup activities with columns for Date, User, Source Namespace Prefix, Action, Section, and Delegate User. The table lists various actions such as Requested an export, Remote Proxy insert DeliveryAPI, Added Change Data Capture entity Rout__c to channel ChangeEvents, and Created Lightning Page: Delivery Order Record Page. The page also includes a sidebar with navigation links like Feature Settings, Sales, Contact Intelligence View, Lead Intelligence View, and Security, and a search bar at the top.

Date	User	Source Namespace Prefix	Action	Section	Delegate User
10/19/2025, 7:32:49 AM PDT	229x1a3924482@agentforce.com		Requested an export	Data Export	
10/19/2025, 8:50:14 AM PDT	229x1a3924482@agentforce.com		Remote Proxy insert DeliveryAPI: https://api.deliveryoptimizer.com	Security Controls	
10/19/2025, 8:46:11 AM PDT	229x1a3924482@agentforce.com		Added Change Data Capture entity Rout__c to channel ChangeEvents	Change Data Capture	
10/19/2025, 8:46:11 AM PDT	229x1a3924482@agentforce.com		Added Change Data Capture entity Order__c to channel ChangeEvents	Change Data Capture	
10/19/2025, 8:46:10 AM PDT	229x1a3924482@agentforce.com		Added Change Data Capture entity Rout_Stop__c to channel ChangeEvents	Change Data Capture	
10/19/2025, 7:13:03 AM PDT	229x1a3924482@agentforce.com		Created Lightning Page: Delivery Order Record Page	Lightning Pages	
10/19/2025, 2:24:28 AM PDT	229x1a3924482@agentforce.com		Changed DeliveryUpdateService Apex Class code	Apex Class	
10/19/2025, 2:24:28 AM PDT	229x1a3924482@agentforce.com		Changed DeliveryNotificationService Apex Class code	Apex Class	
10/19/2025, 2:24:10 AM PDT	229x1a3924482@agentforce.com		Created DeliveryUpdateService Apex Class code	Apex Class	
10/19/2025, 2:22:07 AM PDT	229x1a3924482@agentforce.com		Created DeliveryNotificationService Apex Class code	Apex Class	
10/19/2025, 2:19:05 AM PDT	229x1a3924482@agentforce.com		Changed DailyDeliveryScheduler Apex Class code	Apex Class	
10/19/2025, 2:18:45 AM PDT	229x1a3924482@agentforce.com		Created DailyDeliveryScheduler Apex Class code	Apex Class	
10/19/2025, 2:17:40 AM PDT	229x1a3924482@agentforce.com		Changed DriverAssignmentQueueable Apex Class code	Apex Class	
10/19/2025, 2:17:20 AM PDT	229x1a3924482@agentforce.com		Created DriverAssignmentQueueable Apex Class code	Apex Class	
10/19/2025, 2:16:36 AM PDT	229x1a3924482@agentforce.com		Changed DeliveryOrderBatch Apex Class code	Apex Class	
10/19/2025, 2:16:14 AM PDT	229x1a3924482@agentforce.com		Created DeliveryOrderBatch Apex Class code	Apex Class	
10/19/2025, 1:49:04 AM PDT	229x1a3924482@agentforce.com		Changed Delivery Order Trigger code: DeliveryOrderTrigger	Apex Trigger	
10/19/2025, 1:47:26 AM PDT	229x1a3924482@agentforce.com		Changed RouteService Apex Class code	Apex Class	
10/19/2025, 1:16:56 AM PDT	229x1a3924482@agentforce.com		Created Route Trigger code: RouteTrigger	Apex Trigger	
10/19/2025, 1:13:47 AM PDT	229x1a3924482@agentforce.com		Created Delivery Order Trigger code: DeliveryOrderTrigger	Apex Trigger	

9. Audit Trail

Tracks metadata and data changes.

****During this phase:****

- Review 180-day Setup Audit Trail logs.
- Ensure Field History Tracking is enabled on key fields.
- Review privileged changes and validate approvals.
- Establish audit log governance and anomaly reporting.

Welcome to Salesforce: Reset | Root | Salesforce | View Setup Audit Trail | Salesforce | Reporting and Dashboard Setup | +

orgfarm-d93f59d7e1-dev-ed.develop.my.salesforce-setup.com/lightning/setup/SecurityEvents/home

Setup Home Object Manager

Search Setup

View Setup Audit Trail

View Setup Audit Trail

The last 20 entries for your organization are listed below. You can download your organization's setup audit trail for the last six months (Excel .csv file).

Help for this Page ?

Date	User	Source Namespace Prefix	Action	Section	Delegate User ?
10/26/2025, 7:32:49 AM PDT	229y1a3924482@agentforce.com		Requested an export	Data Export	
10/19/2025, 8:50:14 AM PDT	229y1a3924482@agentforce.com		Remote Proxy Insert DeliveryAPI: https://api.deliveryoptimizer.com	Security Controls	
10/19/2025, 8:46:11 AM PDT	229y1a3924482@agentforce.com		Added Change Data Capture entity Route__c to channel ChangeEvents	Change Data Capture	
10/19/2025, 8:46:11 AM PDT	229y1a3924482@agentforce.com		Added Change Data Capture entity Delivery_Order__c to channel ChangeEvents	Change Data Capture	
10/19/2025, 8:46:10 AM PDT	229y1a3924482@agentforce.com		Added Change Data Capture entity Route_Stop__c to channel ChangeEvents	Change Data Capture	
10/19/2025, 7:13:03 AM PDT	229y1a3924482@agentforce.com		Created Lightning Page: Delivery Order Record Page	Lightning Pages	
10/19/2025, 2:24:29 AM PDT	229y1a3924482@agentforce.com		Changed DeliveryUpdateService Apex Class code	Apex Class	
10/19/2025, 2:24:28 AM PDT	229y1a3924482@agentforce.com		Changed DeliveryUpdateService Apex Class code	Apex Class	
10/19/2025, 2:24:10 AM PDT	229y1a3924482@agentforce.com		Created DeliveryUpdateService Apex Class code	Apex Class	
10/19/2025, 2:22:07 AM PDT	229y1a3924482@agentforce.com		Created DeliveryNotificationService Apex Class code	Apex Class	
10/19/2025, 2:19:05 AM PDT	229y1a3924482@agentforce.com		Changed DailyDeliveryScheduler Apex Class code	Apex Class	
10/19/2025, 2:18:45 AM PDT	229y1a3924482@agentforce.com		Created DailyDeliveryScheduler Apex Class code	Apex Class	
10/19/2025, 2:17:40 AM PDT	229y1a3924482@agentforce.com		Changed DriverAssignmentQueueable Apex Class code	Apex Class	
10/19/2025, 2:17:20 AM PDT	229y1a3924482@agentforce.com		Created DriverAssignmentQueueable Apex Class code	Apex Class	
10/19/2025, 2:15:36 AM PDT	229y1a3924482@agentforce.com		Changed DeliveryOrderBatch Apex Class code	Apex Class	
10/19/2025, 2:15:14 AM PDT	229y1a3924482@agentforce.com		Created DeliveryOrderBatch Apex Class code	Apex Class	
10/19/2025, 1:49:04 AM PDT	229y1a3924482@agentforce.com		Changed Delivery Order Trigger code: DeliveryOrderTrigger	Apex Trigger	
10/19/2025, 1:47:26 AM PDT	229y1a3924482@agentforce.com		Changed RouteService Apex Class code	Apex Class	
10/19/2025, 1:16:55 AM PDT	229y1a3924482@agentforce.com		Created Route Trigger code: RouteTrigger	Apex Trigger	
10/19/2025, 1:13:47 AM PDT	229y1a3924482@agentforce.com		Created Delivery Order Trigger code: DeliveryOrderTrigger	Apex Trigger	

27°C Light rain

Search

ENG IN

19:10 27-10-2025