

FoodieConnect Salesforce CRM

Name : M.Lakshmi Narayana Varma

Email: yarmamudhuluri@gmail.com

1. Introduction

FoodieConnect Pvt. Ltd. is building a Salesforce-powered CRM tailored for the food delivery industry. This phase covers organizational setup, user management, access control, deployment basics, data management, integration setup, and reporting. The goal is to ensure smooth operations, collaboration, and real-time insights for the business.

2. Company Profile Setup

The company information has been configured to reflect FoodieConnect's business identity and operating details.

Organization Name: FoodieConnect Pvt. Ltd.

Primary Contact: Mudhuluri Lakshminarayananarma

Phone: 9381478293

Address: 26-2-696, Chandramouli Nagar, Near Telephone Exchange, 9th Street, Nellore, Andhra Pradesh, India – 524004

Default Locale: English (India)

Default Language: English

Time Zone: India Standard Time (GMT+05:30, Asia/Kolkata)

Currency Locale: INR (English - India)

Multiple Currencies: Not enabled

 **SETUP**
Company Information

User Licenses [10+] | Permission Set Licenses [10+] | Feature Licenses [11] | Usage-based Entitlements [10+]

Organization Detail		Edit	Deactivate Org
Organization Name	FoodieConnect Pvt. Ltd.	Phone	9381478293
Primary Contact	mudhuluri.lakshminarayananarma	Fax	
Division		Default Locale	English (India)
Address	26-2-696, Chandramouli Nagar, Near Telephone Exchange, 9th Street Nellore Andhra Pradesh India, 524004 Nellore 524004 Andhra Pradesh IN	Default Language	English
Fiscal Year Starts In	Custom Fiscal Year	Default Time Zone	(GMT+05:30) India Standard Time (Asia/Kolkata)
Activate Multiple Currencies	<input type="checkbox"/>	Currency Locale	English (India) - INR
Enable Data Translation	<input type="checkbox"/>	Used Data Space	406 KB (8%) [View]
Newsletter	<input checked="" type="checkbox"/>	Used File Space	50 KB (0%) [View]
Admin Newsletter	<input checked="" type="checkbox"/>	API Requests, Last 24 Hours	0 (15,000 max)
Hide Notices About System Maintenance	<input type="checkbox"/>	Streaming API Events, Last 24 Hours	0 (10,000 max)
Hide Notices About System Downtime	<input type="checkbox"/>	Restricted Logins, Current Month	0 (0 max)
Locale Formats	ICU	Salesforce.com Organization ID	00DdM00000d6IBT
		Organization Edition	Developer Edition
		Instance	IND136
Created By	mudhuluri.lakshminarayananarma, 13/09/2025, 5:06 pm	Modified By	mudhuluri.lakshminarayananarma, 13/09/2025, 7:37 pm

[Edit](#) [Deactivate Org](#)

3. Business Hours & Holidays

Business Hours Name: FoodieConnect Support Hours

Schedule: Sunday to Saturday – 24 Hours

Default Business Hours: Enabled

Active: Yes

Holidays Configured: None

The screenshot shows the 'Business Hours Detail' section of a configuration interface. At the top, there's a header with a gear icon labeled 'SETUP' and 'Business Hours'. Below the header, the title 'Organization Business Hours' is displayed. A note says: 'Select the days and hours that your support team is available. These hours, when associated with escalation rules, determine the times at which cases can escalate.' Another note states: 'If you enter blank business hours for a day, that means your organization does not operate on that day.' A 'Holidays' link is present. The main content is a table titled 'Business Hours Detail' with columns: 'Business Hours Name' (FoodieConnect Support Hours), 'Edit' button, 'Time Zone' (GMT+05:30) India Standard Time (Asia/Kolkata), and 'Default Business Hours' (checkbox checked). The table lists days from Sunday to Saturday, all set to '24 Hours'. At the bottom, it shows 'Active' status with a checked checkbox, 'Created By' (mudhuluri lakshminarayananarma) on 13/09/2025, 5:22 pm, and 'Last Modified By' (mudhuluri lakshminarayananarma) on 13/09/2025, 5:22 pm, with an 'Edit' button.

4. Fiscal Year Settings

Fiscal Year 2026:

Start Date: 01-Oct-2025

End Date: 30-Sep-2026

The screenshot shows the 'Custom Fiscal Years' section of a configuration interface. At the top, there's a header with a gear icon labeled 'Fiscal Year'. Below the header, a note says: 'This page allows you to define and edit custom fiscal years, including the names used in reports and forecasts.' Another note says: 'Click the New button to define a new fiscal year. Click Edit to edit a previously defined fiscal year.' A 'New' button is visible. The main content is a table titled 'Custom Fiscal Years' with columns: 'Action' (Edit), 'Year' (2026), 'FY Start Date' (01/10/2025), and 'FY End Date' (30/09/2026).

5. Profiles

Profiles created under the Salesforce Platform license:

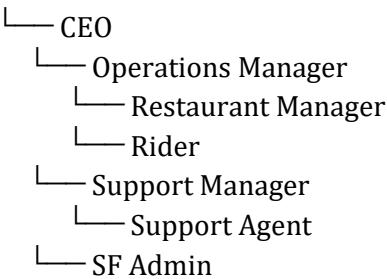
- FoodieConnect Admin – for system administrators with high-level access.
- FoodieConnect Manager – for operational and business managers.
- FoodieConnect Rider – for delivery staff with limited access to orders and tasks.
- FoodieConnect Support – for customer support representatives.

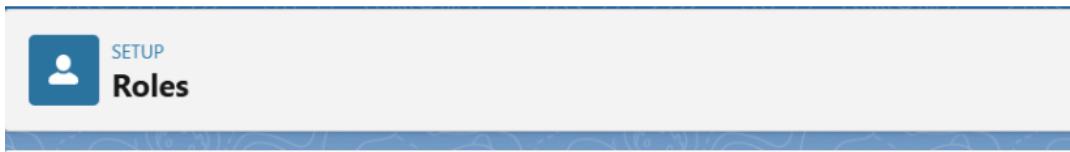
All Profiles		Edit Delete Create New View	
<input type="checkbox"/> Action	Profile Name	User License	Custom
<input type="checkbox"/>	Edit Del ... FoodieConnect Admin	Salesforce Platform	✓
<input type="checkbox"/>	Edit Del ... FoodieConnect Manager	Salesforce Platform	✓
<input type="checkbox"/>	Edit Del ... FoodieConnect Rider	Salesforce Platform	✓
<input type="checkbox"/>	Edit Del ... FoodieConnect Support	Salesforce Platform	✓

6. Roles

Role Hierarchy:

FoodieConnect Pvt. Ltd.





Creating the Role Hierarchy

You can build on the existing role hierarchy shown on this page. To insert a new role, click **Add Role**.

Your Organization's Role Hierarchy

[Collapse All](#) [Expand All](#)



7. Organization-Wide Defaults (OWD)

Customer: Public Read/Write

Delivery: Public Read/Write

Issue: Public Read/Write

Menu Item: Public Read/Write

Order: Controlled by Parent

Payment: Public Read/Write

Provider: Public Read/Write

Resource: Public Read/Write

Restaurant: Public Read/Write

8.Sharing Rules

- Orders Sharing Rule: Managers and Admins have full access to Orders owned by Riders.
- Payments Sharing Rule: Managers have full access to all Payments.
- Customer Sharing Rule: Support Agents have read/write access to Customers.
- Restaurant Sharing Rule: Operations Managers and Admins have full access to Restaurants.



SETUP

Sharing Settings

AppLog	Public Read/Write	Private	<input checked="" type="checkbox"/>
Customer	Public Read/Write	Private	<input checked="" type="checkbox"/>
Delivery	Public Read/Write	Private	<input type="checkbox"/>
Issue	Public Read/Write	Private	<input checked="" type="checkbox"/>
Menu Item	Public Read/Write	Private	<input checked="" type="checkbox"/>
Order	Controlled by Parent	Controlled by Parent	
Payment	Public Read/Write	Private	<input checked="" type="checkbox"/>
Provider	Public Read/Write	Private	<input checked="" type="checkbox"/>
Resource	Public Read/Write	Private	<input checked="" type="checkbox"/>
Restaurant	Public Read/Write	Private	<input checked="" type="checkbox"/>

9. Standard & Custom Objects

- Restaurant:** Stores partner restaurant details.
- Menu Item:** Lists food items offered by restaurants.
- Customer:** Tracks customer details and loyalty points.
- Order:** Manages all orders placed.
- Delivery:** Monitors delivery assignments and status.
- Payment:** Records payment information.
- Issue (Support Case):** Logs customer complaints or queries.

10. Compact Layouts

Compact Layout Detail

Label		Restaurant Highlights	Object Name	Restaurant
API Name	Restaurant_Highlights			
Included Fields	Name Location Cuisine Type Rating			
Created By	mudhuluri lakshminarayanaarma	14/09/2025, 4:43 pm	Modified By	mudhuluri lakshminarayanaarma
				14/09/2025, 4:43 pm

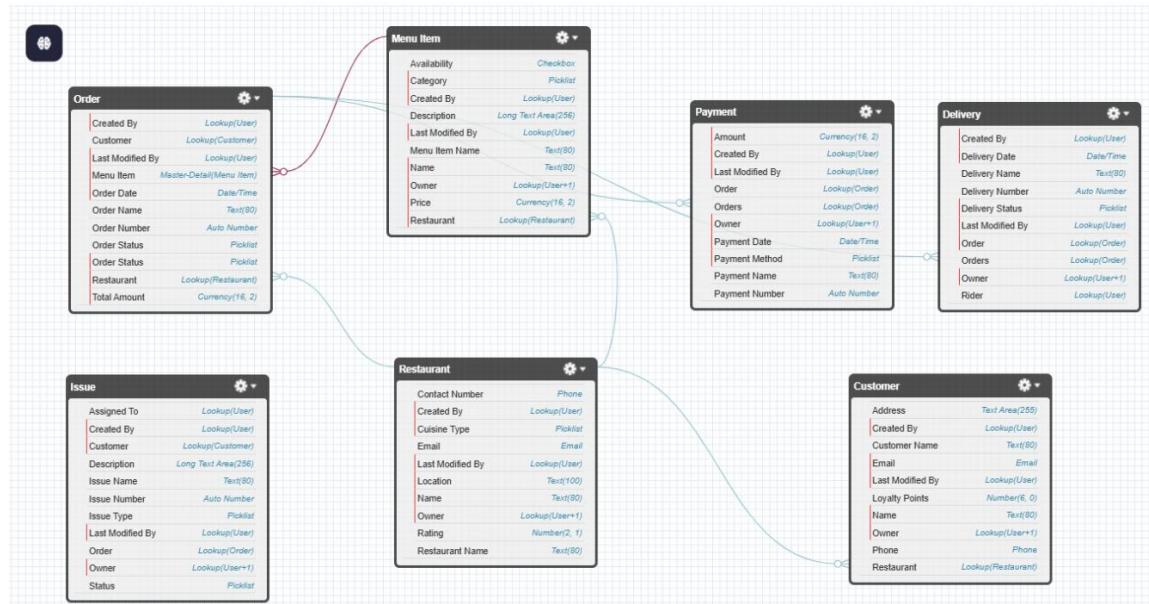
Restaurant Highlights

Menu Item Compact Layout
Menu Item Highlights
[« Back to Menu Item](#)

Compact Layout Detail

Label		Menu Item Highlights	Object Name	Menu Item
API Name	Menu_Item_Highlights			
Included Fields	Name Price Availability Category			
Created By	mudhuluri lakshminarayanaarma	14/09/2025, 4:46 pm	Modified By	mudhuluri lakshminarayanaarma
				14/09/2025, 4:46 pm

11. Schema Builder & Relationships



- **Lookup Relationships:** Connect related records, e.g., Menu Items → Restaurant.
- **Master-Detail Relationships:** Maintain tighter control, e.g., Orders → Menu Items.

12. Validation Rules

Purpose: Ensure data integrity by preventing invalid records from being saved.

Implemented Rules:

- Order_Total_Positive
 - Object: Order
 - Formula: Total_Amount_c <= 0
 - Error Message: "Order Total Amount must be greater than 0."
 - Status: Active

Order Validation Rule

[Back to Order](#)

Validation Rule Detail

Rule Name	Order_Total_Positive	Edit Clone	Active <input checked="" type="checkbox"/>
Error Condition Formula	Total_Amount_c >= 0		
Error Message	Order Total Amount must be greater than 0		Error Location Total Amount
Description			Created By mudhuluri lakshminarayana varma , 14/09/2025, 5:39 pm

Modified By [mudhuluri lakshminarayana varma](#), 14/09/2025, 5:39 pm

[Edit](#) [Clone](#)

Payment Validation Rule

[Back to Payment](#)

Validation Rule Detail

Rule Name	Payment_Amount_Positive	Edit Clone	Active <input checked="" type="checkbox"/>
Error Condition Formula	Amount_c <= 0		
Error Message	Payment amount must be greater than zero.		Error Location Amount
Description			Created By mudhuluri lakshminarayana varma , 19/09/2025, 4:55 pm

Modified By [mudhuluri lakshminarayana varma](#), 19/09/2025, 4:55 pm

[Edit](#) [Clone](#)

Menu Item Validation Rule

[Back to Menu Item](#)

Validation Rule Detail

Rule Name	MenuItem_Price_Positive	Edit Clone	Active <input checked="" type="checkbox"/>
Error Condition Formula	Price_c < 0		
Error Message	Menu item Price cannot be negative		Error Location Price
Description			Created By mudhuluri lakshminarayana varma , 14/09/2025, 5:42 pm

Modified By [mudhuluri lakshminarayana varma](#), 14/09/2025, 5:42 pm

[Edit](#) [Clone](#)

13. Email Alerts

Purpose: Automatically notify customers when key actions occur.

Implemented Alerts:

- Payment Received Email Alert
 - Object: Payment
 - Email Template: Payment Received - Customer
 - Recipients: Customer email (configured for testing to owner email)

SETUP Email Alerts

Email Alert
Send email to customer when payment is received

[Rules Using This Email Alert](#) | [Approval Processes Using This Email Alert](#) | [Entitlement Processes Using This Email Alert](#)

Email Alert Detail

Description	Send email to customer when payment is received	Edit Delete Close
Unique Name	Send_email_to_customer_when_payment_is_received	Email Template Object
From Email Address	Current User's email address	Payment Received - Customer
Recipients	User_muthulalakshminarayanaarma	Payment
Additional Emails		
Created By	muthulalakshminarayanaarma, 19/09/2025, 4:22 pm	Modified By muthulalakshminarayanaarma, 19/09/2025, 4:22 pm
	Edit Delete Close	

Rules Using This Email Alert
This alert is currently not used by any rules

Approval Processes Using This Email Alert
This alert is currently not used by any approval processes

Entitlement Processes Using This Email Alert
This alert is currently not used by any entitlement processes

SETUP Classic Email Templates

Text Email Template
Payment Received - Customer

Preview your email template below.

Email Template Detail

Email Templates from Salesforce	Unified Public Classic Email Templates	Edit Delete Clone
Email Template Name	Payment Received - Customer	Available For Use <input checked="" type="checkbox"/>
Template Unique Name	Payment_Received_Customer	Last Used Date
Encoding	Unicode (UTF-8)	Times Used
Author	muthulalakshminarayanaarma [Change]	
Description	Sends a confirmation email to customer when payment is received.	Modified By muthulalakshminarayanaarma, 19/09/2025, 4:17 pm
Created By	muthulalakshminarayanaarma, 19/09/2025, 4:17 pm	
	Edit Delete Close	

Email Template

[Send Text and Verify Merge Fields](#)

Plain Text Preview

```

Subject: Your Payment #([Payment.Payment_Number__c] for Order #([Payment.Order__c]) has been received.

Hello ([MultiValue([Payment.Order__r,"Sir or Madam"])].

We have successfully received your payment.

Payment Details:
- Payment Number: ([Payment.Payment_Number__c])
- Order Number: ([Payment.Order__c])
- Amount: ([Payment.Amount__c])
- Payment Method: ([Payment.Payment_Method__c])
- Payment Date: ([Payment.Payment_Date__c])

Thank you for choosing FoodieConnect!
We're excited to continue serving you.

Best regards,
FoodieConnect Support Team

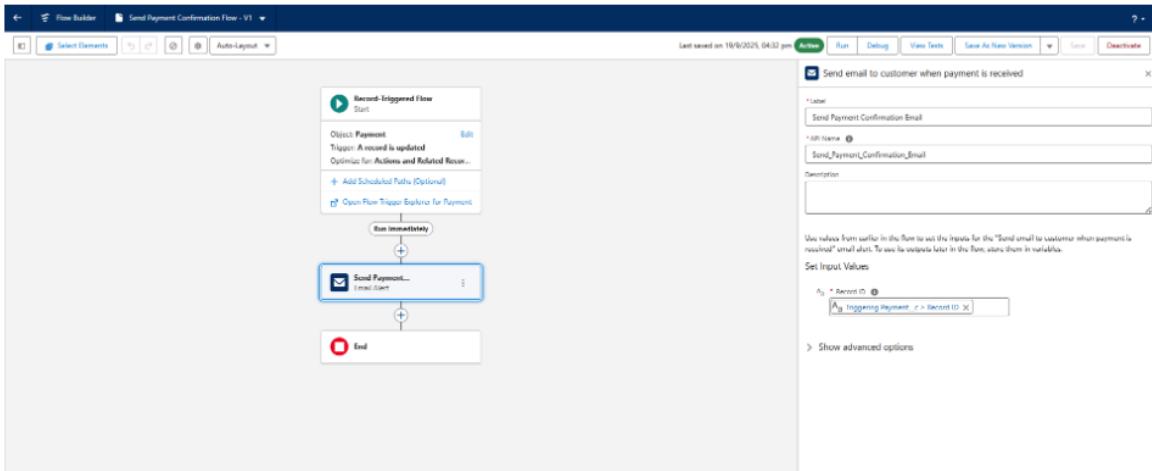
```

14. Flow Builder (Record-Triggered Flows)

Purpose: Automate actions based on record changes.

Implemented Flows:

- **Payment Confirmation Flow**
 - Triggered when a **Payment** record is created.
 - Automatically sends the **Payment Received Email Alert**.



15. Classes & Objects

RestaurantService

Handles restaurant-related operations such as fetching active restaurants and their menu items based on location.

Key Methods:

- `getActiveRestaurantsByLocation(String location)` → Fetches active restaurants and their menu items for a specific location
- `updateRestaurantRating(Set<Id> restaurantIds)` → Updates restaurant ratings based on published reviews

```

public with sharing class RestaurantService {
    // Method to get restaurants by location (active field removed)
    public static List<Restaurant__c> getRestaurantsByLocation(String location) {
        return [
            SELECT Id, Name, Name__c, Contact_Number__c, Email__c, Rating__c, Cuisine_Type__c, Location__c,
            (SELECT Id, Name, Price__c, Description__c
            FROM Menu_Items__r
            ORDER BY Name)
            FROM Restaurant__c
            WHERE Location__c = :location
            ORDER BY Rating__c DESC NULLS LAST
            LIMIT 50
        ];
    }
}

```

OrderHandle

Manages order lifecycle, including auto-generating order numbers, validating order status, creating delivery records, and updating customer loyalty points.

Key Methods:

- `beforeInsert(List<Order__c> newOrders)` → Sets default status and generates order numbers
- `afterInsert(List<Order__c> newOrders)` → Creates delivery records and updates loyalty points
- `beforeUpdate(List<Order__c> newOrders, Map<Id, Order__c> oldMap)` → Validates order status changes
- `afterUpdate(List<Order__c> newOrders, Map<Id, Order__c> oldMap)` → Updates related deliveries



The screenshot shows a Salesforce code editor with the tab 'OrderHandle.apxc' selected. The code is written in Apex and defines a class named 'OrderHandle' with several static methods for managing order lifecycle events.

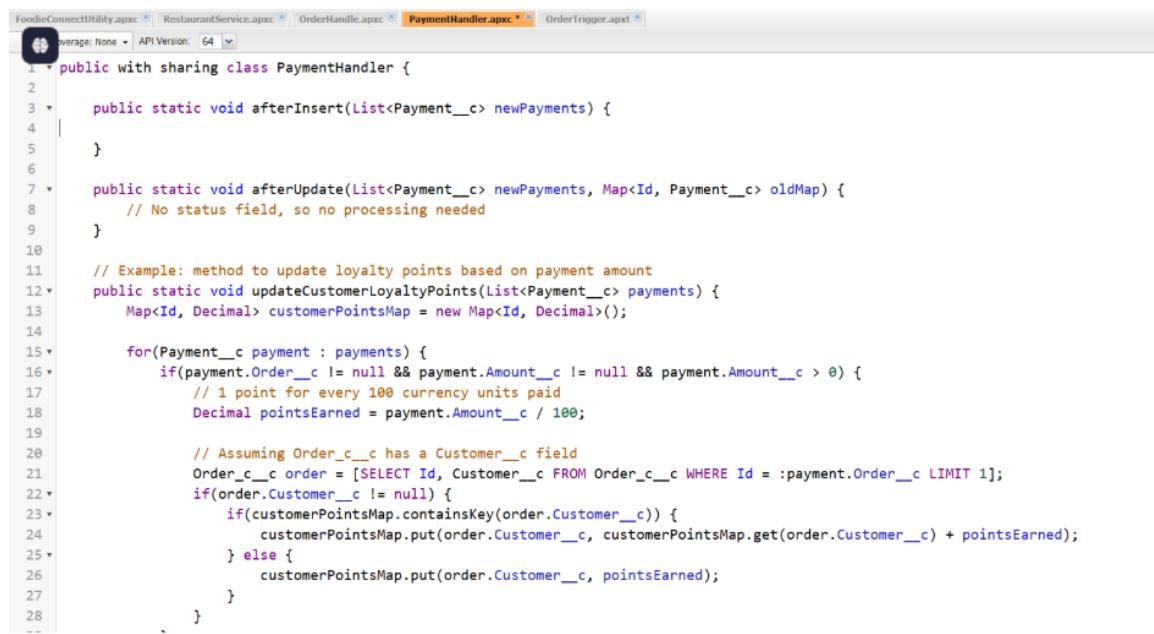
```
1 public with sharing class OrderHandle {
2
3     public static void beforeInsert(List<Order__c> newOrders) {
4         // Set default status if not provided
5         for(Order__c order : newOrders) {
6             if(order.Order_Status__c == null) {
7                 order.Order_Status__c = 'New';
8             }
9         }
10    }
11
12    public static void afterInsert(List<Order__c> newOrders) {
13        // Update customer loyalty points
14        updateCustomerLoyaltyPoints(newOrders);
15    }
16
17    public static void beforeUpdate(List<Order__c> newOrders, Map<Id, Order__c> oldMap) {
18        validateOrderStatusChange(newOrders, oldMap);
19    }
20
21    public static void afterUpdate(List<Order__c> newOrders, Map<Id, Order__c> oldMap) {
22        // No related deliveries to update in current schema
23    }
24
25    private static void validateOrderStatusChange(List<Order__c> newOrders, Map<Id, Order__c> oldMap) {
26        for(Order__c newOrder : newOrders) {
27            Order__c oldOrder = oldMap.get(newOrder.Id);
28        }
29    }
30}
```

PaymentHandler

Manages payments and integrates payment data with orders. Updates order status after payment creation.

Key Methods:

- `afterInsert(List<Payment__c> newPayments)` → Updates related orders after payment creation
- `afterUpdate(List<Payment__c> newPayments, Map<Id, Payment__c> oldMap)` → Handles updates in payment information



The screenshot shows the Salesforce code editor with the tab 'PaymentHandler.apxc' selected. The code implements the PaymentHandler interface with two methods: afterInsert and afterUpdate. The afterUpdate method includes logic to update customer loyalty points based on payment amount. The code uses Apex syntax with annotations like `public with sharing class PaymentHandler {` and `public static void afterInsert(List<Payment__c> newPayments) {`.

```
1 public with sharing class PaymentHandler {
2
3     public static void afterInsert(List<Payment__c> newPayments) {
4
5     }
6
7     public static void afterUpdate(List<Payment__c> newPayments, Map<Id, Payment__c> oldMap) {
8         // No status field, so no processing needed
9     }
10
11    // Example: method to update loyalty points based on payment amount
12    public static void updateCustomerLoyaltyPoints(List<Payment__c> payments) {
13        Map<Id, Decimal> customerPointsMap = new Map<Id, Decimal>();
14
15        for(Payment__c payment : payments) {
16            if(payment.Order__c != null && payment.Amount__c != null && payment.Amount__c > 0) {
17                // 1 point for every 100 currency units paid
18                Decimal pointsEarned = payment.Amount__c / 100;
19
20                // Assuming Order__c has a Customer__c field
21                Order__c order = [SELECT Id, Customer__c FROM Order__c WHERE Id = :payment.Order__c LIMIT 1];
22                if(order.Customer__c != null) {
23                    if(customerPointsMap.containsKey(order.Customer__c)) {
24                        customerPointsMap.put(order.Customer__c, customerPointsMap.get(order.Customer__c) + pointsEarned);
25                    } else {
26                        customerPointsMap.put(order.Customer__c, pointsEarned);
27                    }
28                }
29            }
30        }
31    }
32 }
```

Apex Triggers

OrderTrigger

Implements the trigger handler pattern for the Order__c object.

Trigger Events:

- Before Insert → Auto-generates order numbers and sets default status
- After Insert → Creates delivery records and updates customer loyalty points
- Before Update → Validates order status changes to prevent invalid transitions
- After Update → Updates delivery statuses based on order status changes

```

trigger OrderTrigger on Order_c__c (before insert, before update, after insert, after update) {
    if(Trigger.isBefore) {
        if(Trigger.isInsert) {
            OrderHandle.beforeInsert(Trigger.new);
        }
        if(Trigger.isUpdate) {
            OrderHandle.beforeUpdate(Trigger.new, Trigger.oldMap);
        }
    }
    if(Trigger.isAfter) {
        if(Trigger.isInsert) {
            OrderHandle.afterInsert(Trigger.new);
        }
        if(Trigger.isUpdate) {
            OrderHandle.afterUpdate(Trigger.new, Trigger.oldMap);
        }
    }
}

```

SOQL Queries

Used SOQL to fetch orders, customers, and restaurant details.

Example Queries:

SELECT Id, Customer_r.Name, Total_Amount__c FROM Order_c WHERE Order_Status__c = 'Confirmed'

SELECT Id, Order_Number__c, Total_Amount__c, Customer_c FROM Order_c LIMIT 10

Query Results - Total Rows: 1			
Id	Order_Number__c	Total_Amount__c	Customer_c
a06dM00000f7jHQ43	ORD-0001	250	006dM00000001QAI

History
Executed
SELECT Id, Customer_r.Name, Total_Amount__c FROM Order_c WHERE Order...
SELECT Id, Order_Number__c, Total_Amount__c, Customer_c FROM Order_c ...

OrderHandlerTest

Covers unit testing for order insertion, delivery creation, and status validation.

Test Cases:

- Verify auto-generated order numbers are unique and sequential

- Confirm delivery records are created for orders with 'Confirmed' status
- Prevent status regression (e.g., changing 'Delivered' orders back to 'Confirmed')
- Validate loyalty points are correctly calculated and assigned
- Test bulk order processing for performance

Test Coverage Features:

- @isTest annotation for test classes and methods
- Test.startTest() and Test.stopTest() for governor limit reset
- System.assert() methods for validation
- Test data factory methods for reusable test data creation

Screenshot of the Salesforce IDE showing the OrderHandlerTest.apxc code and the Progress tab of the Test Results window.

```

FoodieConnectUtility.apxc RestaurantService.apxc OrderHandle.apxc PaymentHandler.apxc OrderTrigger.apxc OrderHandlerTest.apxc
Coverage: None API Version: 64
1 @isTest
2 private class OrderHandlerTest {
3
4     @testSetup
5     static void setupTestData() {
6         // Create test restaurant
7         Restaurant__c restaurant = new Restaurant__c(
8             Name = 'Test Restaurant',
9             Cuisine_Type__c = 'Italian',
10            Contact_Number__c = '9876543210',
11            Location__c = 'Test Location',
12            Rating__c = 4.5
13        );
14        insert restaurant;
15
16        // Create test customer
17        Customer__c customer = new Customer__c(
18            Name = 'Test Customer',
19            Phone__c = '9876543211',
20            Email__c = 'test@example.com'
21        );
22        insert customer;
23    }
24
25    @isTest
26    static void testOrderInsert() {
27        // Get test data
28        Restaurant__c restaurant = [SELECT Id FROM Restaurant__c LIMIT 1];
29        Customer__c customer = [SELECT Id FROM Customer__c LIMIT 1];
}

```

Progress Tab Data:

Step	Label	Status	Start	End	Duration (ms)	Handler	Errors	Details
90	Getting members of ApexClassMember for containerId=1d0f0000000000000000000000000000	Finished	1:02:59	1:02:59	240			
89	-2	Getting deployment for id=1d0f0000000000000000000000000000	Finished	1:02:59	1:02:59	259		
88	-1	Creating deployment for containerId=1d0f0000000000000000000000000000	Finished	1:02:57	1:02:57	240		
87	-3	Creating or Updating containerMember for containerId=1d0f0000000000000000000000000000	Finished	1:02:56	1:02:57	377		
86	-1	Getting members of ApexTriggerMember for containerId=1d0f0000000000000000000000000000	Finished	12:51:53	12:51:53	176		
85	-2	Getting deployment for id=1d0f0000000000000000000000000000	Finished	12:51:53	12:51:53	132		
84	-3	Creating deployment for containerId=1d0f0000000000000000000000000000	Finished	12:51:51	12:51:51	208		
83	-1	Creating or Updating containerMember for containerId=1d0f0000000000000000000000000000	Finished	12:51:49	12:51:49	179		
82	-4	Getting members of ApexTriggerMember for containerId=1d0f0000000000000000000000000000	Finished	12:51:45	12:51:45	142		
81	-2	Getting deployment for id=1d0f0000000000000000000000000000	Finished	12:51:45	12:51:45	129		
80	-1	Creating deployment for containerId=1d0f0000000000000000000000000000	Finished	12:51:43	12:51:44	213		
79	-4	Creating or Updating containerMember for containerId=1d0f0000000000000000000000000000	Finished	12:51:43	12:52:43	150		
78	-1	Getting members of ApexClassMember for containerId=1d0f0000000000000000000000000000	Finished	12:51:37	12:51:37	114		

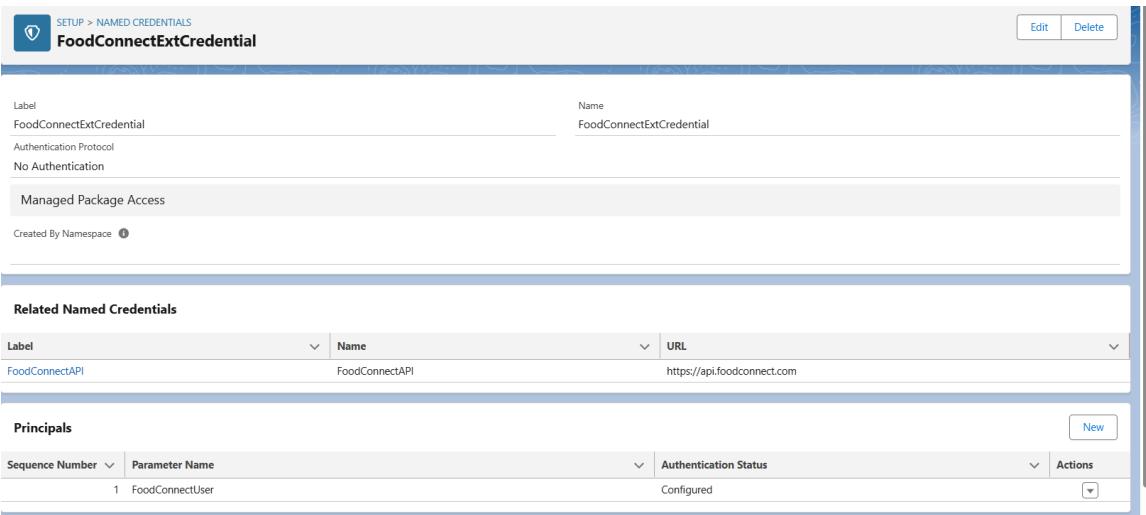
16.Named Credentials



The screenshot shows the 'SETUP > NAMED CREDENTIALS' interface for creating a new named credential named 'FoodConnectAPI'. The form includes fields for Label (FoodConnectAPI), URL (https://api.foodconnect.com), and Enabled for Callouts (checked). Under Authentication, it is set to 'External Credential' and points to 'FoodConnectExtCredential'. A summary bar at the bottom indicates the status is 'Completed'.

- **Named Credential Created:** FoodConnectAPI
- **URL:** https://api.foodconnect.com
- **Authentication:** Custom Header (Authorization: Bearer DUMMY_API_KEY)
- **Purpose:** Simplifies callouts to external APIs, securely stores credentials, and handles authentication automatically.
- **Status:** Completed
-

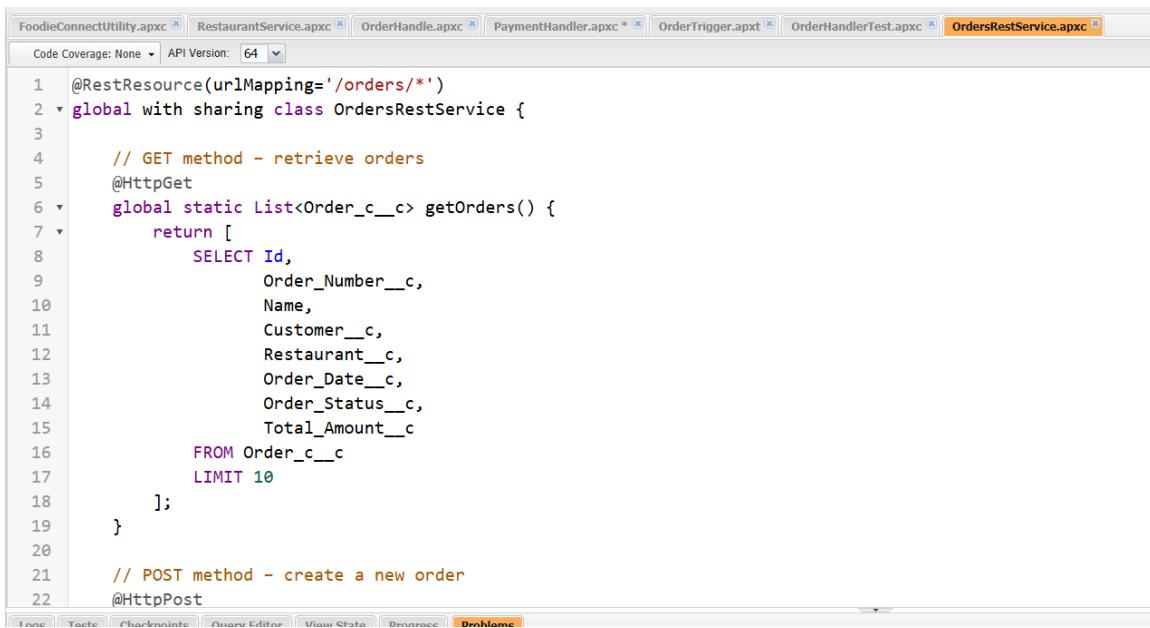
17. External Credentials



The screenshot shows the 'SETUP > NAMED CREDENTIALS' interface for the external credential 'FoodConnectExtCredential'. It lists the label 'FoodConnectExtCredential', authentication protocol 'No Authentication', and managed package access. In the 'Related Named Credentials' section, 'FoodConnectAPI' is listed with its URL. The 'Principals' section shows one principal named 'FoodConnectUser' with an authentication status of 'Configured'.

- **Status:** Configured
- **Purpose:** Enhanced security for external system authentication
- **Notes:** Used in conjunction with Named Credentials for advanced authentication scenarios

18. Web Services (REST/SOAP)



```

FoodieConnectUtility.apxc RestaurantService.apxc OrderHandle.apxc PaymentHandler.apxc OrderTrigger.apxc OrderHandlerTest.apxc OrdersRestService.apxc
Code Coverage: None API Version: 64
1  @RestResource(urlMapping='/orders/*')
2  global with sharing class OrdersRestService {
3
4      // GET method - retrieve orders
5      @HttpGet
6      global static List<Order_c__c> getOrders() {
7          return [
8              SELECT Id,
9                  Order_Number__c,
10                 Name,
11                 Customer__c,
12                 Restaurant__c,
13                 Order_Date__c,
14                 Order_Status__c,
15                 Total_Amount__c
16             FROM Order_c__c
17             LIMIT 10
18         ];
19     }
20
21     // POST method - create a new order
22     @HttpPost

```

The screenshot shows the Salesforce Apex code editor with the OrdersRestService.apxc file open. The code defines a REST service with a GET method to retrieve orders and a POST method to create a new order. The GET method uses a query to select specific fields from the Order_c__c object, including Id, Order_Number__c, Name, Customer__c, Restaurant__c, Order_Date__c, Order_Status__c, and Total_Amount__c, with a limit of 10 records.

- **SOAP Web Service:** OrdersSoapService
 - `getOrders()` → Retrieves orders
 - `createOrder(customerId, restaurantId, totalAmount)` → Creates a new order
- **Status:** Implemented and functional in Apex
- **Purpose:** Enables external or internal systems to interact with Salesforce order data via SOAP.

19. Callouts

- **Status:** Configured using Named Credential
- **Purpose:** Apex callouts can access external API endpoints (<https://api.foodconnect.com>) without needing separate Remote Site Settings.
- **Example Usage:** `HttpRequest req = new HttpRequest();
req.setEndpoint('callout:FoodConnectAPI/orders'); req.setMethod('GET');`

```
Http http = new Http(); HttpResponse res = http.send(req); System.debug(res.getBody());
```

20. Platform Events

The screenshot shows the 'Platform Event Definition Detail' for 'OrderEvent'. Key details include:

- Singular Label:** OrderEvent
- Plural Label:** OrderEvents
- Object Name:** OrderEvent
- API Name:** OrderEvent__e
- Event Type:** High Volume
- Publish Behavior:** Publish Immediately
- Created By:** mudhuluri.lakshminarayana varma, 25/09/2025, 12:18 pm
- Description:** "Event triggered when a new order is created in Food Connect, including order details and status."
- Deployment Status:** Deployed
- Modified By:** mudhuluri.lakshminarayana varma, 25/09/2025, 12:18 pm

Standard Fields table:

Action	Field Label	Field Name	Data Type	Controlling Field	Indexed
Created By	CreatedBy	CreatedBy	Lookup(User)		
Created Date	CreatedDate	CreatedDate	Date/Time		
Event UUID	EventUuid	EventUuid	Text(36)		
Replay ID	ReplayId	ReplayId	External Lookup		

- Event Created:** OrderEvent__e
- Type:** High Volume
- Publish Behavior:** Publish Immediately
- Status:** Implemented
- Purpose:** Used to notify external systems or trigger processes in real-time when orders are created or updated.

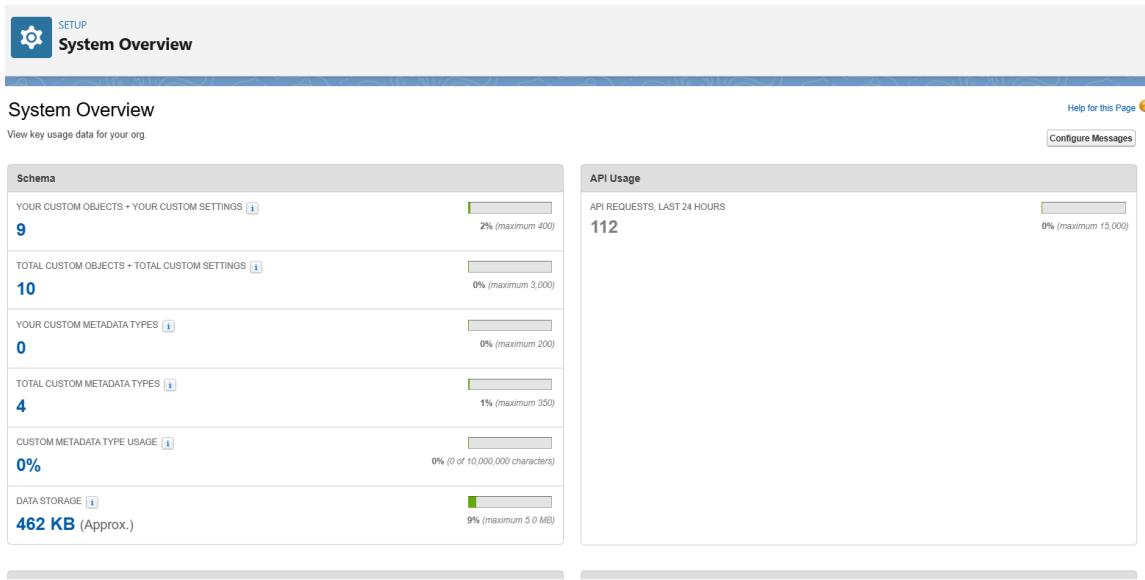
21. Change Data Capture

The screenshot shows the 'Change Data Capture' setup page. It lists entities available for selection and those currently selected:

Available Entities	Selected Entities
Account (Account)	Order (Order_c__c)
Account Clean Info (AccountCleanInfo)	Issue (Issue_c)
Account Contact Role (AccountContactRole)	Payment (Payment_c)
Asset (Asset)	Delivery (Delivery_c)
Asset Relationship (AssetRelationship)	Customer (Customer_c)
Assigned Resource (AssignedResource)	

- **Objects Enabled:** Order_c
- **Purpose:** Automatically tracks Create, Update, Delete, Undelete operations and publishes events.
- **Status:** Implemented
- **Note:** Used for real-time integration with external systems.

22. API Limits



- **Current Usage:** 112 requests in last 24 hours (0% of 15,000 daily limit)
- **Status:** Checked
- **Notes:** Project is within safe API usage limits; monitoring only.

23. OAuth & Authentication

SETUP

Auth. Providers

Auth. Provider

Auth. Provider Detail	
Auth. Provider ID	0SOdM000001j2BZ
Provider Type	Salesforce
Name	FoodConnectOAuth
URL Suffix	FoodConnectOAuth
Consumer Key	From external API / Salesforce connected app
Consumer Secret	Click to reveal
Authorize Endpoint URL	https://login.salesforce.com/services/oauth2/authorize
Token Endpoint URL	https://login.salesforce.com/services/oauth2/token
Use Proof Key for Code Exchange (PKCE) Extension	<input checked="" type="checkbox"/> i
Default Scopes	full refresh_token
Include Consumer Secret in SOAP API Responses	<input checked="" type="checkbox"/> i
Custom Error URL	
Custom Logout URL	
Registration Handler Type	None
Portal	
Icon URL	
Use Salesforce MFA for this SSO Provider	<input type="checkbox"/> i
Salesforce Configuration	
Test-Only Initialization URL	https://foodcom-dev-ed.my.salesforce.com/services/auth/test/FoodConnectOAuth
Existing User Linking URL	https://foodcom-dev-ed.my.salesforce.com/services/auth/link/FoodConnectOAuth
OAuth-Only Initialization URL	https://foodcom-dev-ed.my.salesforce.com/services/auth/oauth/FoodConnectOAuth
Callback URL	https://foodcom-dev-ed.my.salesforce.com/services/authcallback/FoodConnectOAuth

- **Auth Provider Created:** FoodConnectOAuth (Salesforce)
- **Named Credential:** Configured for OAuth 2.0 authentication
- **Status:** Implemented
- **Notes:** OAuth 2.0 authentication implemented for secure API access.

23. Remote Site Settings

SETUP

Remote Site Settings

Remote Site Details

Remote Site Detail	
Remote Site Name	FoodConnectAPI
Remote Site URL	https://api.foodconnect.com
Disable Protocol Security	<input type="checkbox"/>
Description	External API for Food Connect callouts
Active	<input checked="" type="checkbox"/>
Created By	mudhuluri lakshminarayana varma, 25/09/2025, 12:32 pm
	Edit Delete Clone

- **Status:** Configured
- **Notes:** Additional remote site settings configured for backup callout endpoints and external service integrations.

24.Data Import Wizard

Bulk Data Load Job
750dM00000XYDB9

View the details of a bulk data load job.

< Back to List: Bulk Data Load Jobs

Bulk Data Load Job Detail

Job ID	750dM00000XYDB9	Submitted By	mudhuluri lakshminarayana varma	Job Type	Bulk V1	Status	Closed
Start Time	25/09/2025, 1:06 pm IST	Operation	Insert	Queued Batches	0	Total Processing Time (ms)	107
End Time	25/09/2025, 1:06 pm IST	In Progress Batches	0	Completed Batches	1	API Active Processing Time (ms)	43
Time to Complete (hh:mm:ss)	00:00	Object	Customer	Failed Batches	0	Apex Processing Time (ms)	0
External ID Field		Content Type	CSV	Progress	100%		
Concurrency Mode	Parallel	API Version	64.0	Records Processed	3		
				Records Failed	0		
				Retries	0		

Reload

Bulk Data Load Job
750dM00000XYdIT

View the details of a bulk data load job.

< Back to List: Bulk Data Load Jobs

Bulk Data Load Job Detail

Job ID	750dM00000XYdIT	Submitted By	mudhuluri lakshminarayana varma	Job Type	Bulk V1	Status	Closed
Start Time	25/09/2025, 1:08 pm IST	Operation	Insert	Queued Batches	1	Total Processing Time (ms)	0
End Time		In Progress Batches	0	Completed Batches	0	API Active Processing Time (ms)	0
Time to Complete (hh:mm:ss)		Object	Menu Item	Failed Batches	0	Apex Processing Time (ms)	0
External ID Field		Content Type	CSV	Progress	0%		
Concurrency Mode	Parallel	API Version	64.0	Records Processed	0		
				Records Failed	0		
				Retries	0		

Abort Reload

Purpose:

The Data Import Wizard is a Salesforce tool designed to simplify the process of importing data from external sources like CSV files into Salesforce standard and custom objects. It is particularly helpful for admins and users needing to upload data without scripting or programming knowledge. The wizard supports objects such as Customers, Orders, Payments, Restaurants, and more.

Detailed Steps Implemented:

- Accessing the Wizard:** The process begins by navigating to Salesforce Setup and typing "Data Import Wizard" in the Quick Find box. Selecting the Data Import Wizard from the results launches the import interface.
- Object Selection:** Users choose the object they want to import data into. For FoodieConnect, both standard and custom objects such as Customers were selected.

- **File Upload:** A CSV file containing the data (e.g., customer details) is uploaded via the wizard.
- **Field Mapping:** The wizard attempts automatic mapping of CSV file headers to Salesforce object fields. Example mappings for Customer data include:
 - CSV First_Name → Salesforce field First Name
 - CSV Last_Name → Salesforce field Last Name
 - CSV Email → Salesforce field Email
 - CSV Phone → Salesforce field Phone
 - CSV Address → Salesforce field Address
 - CSV Loyalty_Points → Salesforce field Loyalty Points
 - Any unmapped fields can be manually corrected before proceeding.
- **Validation & Import:** The wizard verifies the mapping, allowing corrections before starting the import process.
- **Import Success:** Successfully imported records are now available in the Salesforce CRM, ensuring data integrity and completeness.

Outcome:

The import process was smooth and accurate, resulting in a fully populated Customer database in Salesforce that reflects the source CSV data.

25. Data Loader

Purpose:

The Data Loader is a more powerful Salesforce client application ideal for handling large-scale data operations. It supports bulk data insertions, updates, upserts, and exports – especially useful for objects and volumes not supported by the Data Import Wizard.

Detailed Steps Implemented:

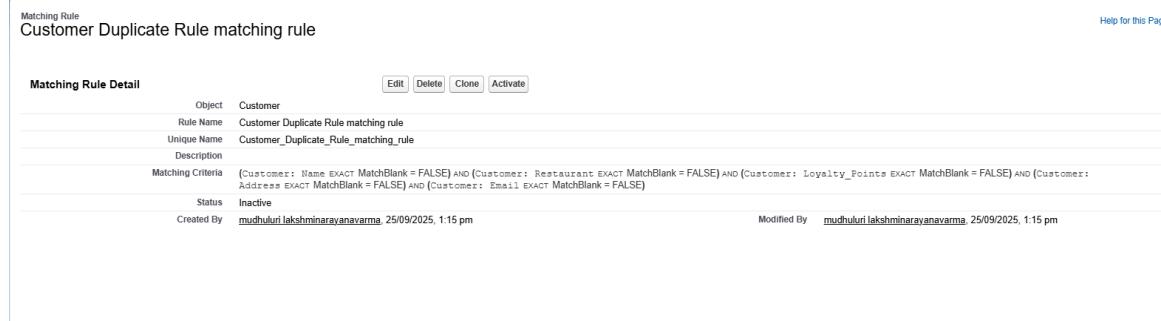
- **Installation:** The Data Loader application was downloaded and installed locally.
- **Connection Setup:** Configured connection credentials including Salesforce username, password, and security token to authenticate API access.
- **Bulk Operations:** Used to import large datasets for Orders and Payments which exceeded the Data Import Wizard's practical limits.

- **Data Integrity:** Care was taken to maintain proper data relationships (such as lookup and master-detail relationships) to ensure data consistency across related objects.
- **Error Handling:** The Data Loader's detailed error reports helped quickly identify and fix any issues during upload.

Outcome:

The Data Loader enabled efficient handling of large datasets, significantly supporting testing and validating CRM workflows from order placement to payment and delivery.

26. Duplicate Rules



The screenshot shows the 'Matching Rule Detail' page for a rule named 'Customer Duplicate Rule matching rule'. The page includes fields for Object (Customer), Rule Name (Customer Duplicate Rule matching rule), Unique Name (Customer_Duplicate_Rule_matching_rule), Description, Matching Criteria (a complex logical expression involving Name, Restaurant, Loyalty Points, Address, and Email fields), Status (Inactive), Created By (mudhuluri lakshminarayana varma, 25/09/2025, 1:15 pm), and Modified By (mudhuluri lakshminarayana varma, 25/09/2025, 1:15 pm). Action buttons for Edit, Delete, Clone, and Activate are visible at the top right.

Purpose:

To maintain a clean and reliable customer database, Duplicate Rules are essential. They help detect, prevent, and manage duplicate records, reducing inconsistencies and improving user trust in the data.

Detailed Steps Implemented:

- **Setup Navigation:** Accessed through Setup under Duplicate Management.
- **Rule Creation:** A Duplicate Rule for the Customer object was created and configured.
- **Matching Logic:** The rule matches records based on critical criteria like Customer Name, Restaurant affiliation, Loyalty Points, Address, and Email.
- **User Experience:** When duplicates are detected during record creation or edits, users are alerted with options for resolution.
- **Reporting:** Duplicate detection is logged for administrative oversight.

Outcome:

Duplicate entries are minimized, which enhances CRM data quality and operational efficiency.

27. Data Export & Backup

The screenshot shows the Salesforce Setup interface under the 'Data Export' section. At the top, there's a message about the 'Monthly Export Service'. Below it, a table lists a single scheduled export entry:

Next scheduled export:	None
Scheduled By:	mudhuluri lakshminarayana varma
Schedule Date:	25/09/2025
Export File Encoding:	ISO-8859-1 (General US & Western European, ISO-LATIN-1)

Below this is a file list table:

Action	File Name	File Size
download	WE_00dM00000d6IBtUAM_1.ZIP	1.1K

At the bottom of the page, there's a file list interface with columns for Name, Date modified, Type, and Size, showing two CSV files: 'Customer_c.csv' and 'Menu_Item_c.csv'.

Purpose:

Regular data backups are critical for disaster recovery, compliance, and offline access. Salesforce's Data Export feature automates this process.

Detailed Steps Implemented:

- Access:** Located in Setup under Data Export.
- Scheduling:** Configured manual and recurring export schedules to cover all relevant Salesforce objects including Customers, Orders, Payments, etc.
- Export Format:** Data was exported as CSV files packaged inside ZIP archives, using the ISO-8859-1 encoding to accommodate special characters.
- Download & Storage:** Exported ZIP files downloaded and safely stored outside Salesforce for redundancy.
- Restore Readiness:** Data is ready for import back into Salesforce or migration to other environments if needed.

Outcome:

A reliable backup strategy is in place ensuring data safety and availability.

28. Change Sets

Purpose:

Change Sets simplify deployment by transferring metadata changes (like new fields or triggers) between Salesforce orgs, typically from sandbox to production environments.

Implementation Status:

Although Change Sets were noted, they were not visible or actively used in this FoodieConnect org because all work was done within a single environment, making Change Sets optional.

Outcome:

No deployment disruptions from Change Sets; future deployments can consider using Change Sets if multi-org rollout is introduced.

29. Unmanaged vs Managed Packages

The screenshot shows the Salesforce Package Manager interface. At the top, there's a header with a 'SETUP' button, a 'Package Manager' title, and a 'Help' link. Below the header, the package name 'FoodieConnect_Unmanaged' is displayed, along with a 'Back to Package List' link. The main area is divided into two sections: 'Package Detail' and a table of components.

Package Detail

	Value	Action
Package Name	FoodieConnect_Unmanaged	Edit
Language	English	Delete
Notify on Apex Error	mudhuluri lakshminarayanaarma	Upload
Created By	mudhuluri lakshminarayanaarma	25/09/2025, 5:06 pm
Description	FoodieConnect CRM App (Restaurants, Orders, Payments).	Last Modified By mudhuluri lakshminarayanaarma 25/09/2025, 5:06 pm

Components [Versions]

Action	Component Name	Parent Object	Type	Included By
<u>Address</u>		Customer	Custom Field	<u>Customer</u>
All		Payment	List View	<u>Payment</u>
All		Delivery	List View	<u>Delivery</u>
All		Restaurant	List View	<u>Restaurant</u>
All		Customer	List View	<u>Customer</u>
All		Menu Item	List View	<u>Menu Item</u>
All		Issue	List View	<u>Issue</u>
All		Order	List View	<u>Order</u>
<u>Amount</u>		Payment	Custom Field	<u>Payment</u> <u>Payment_Amount_Positive</u> <u>Payment_Highlights</u>
<u>Assigned To</u>		Issue	Custom Field	<u>Issue</u>
<u>Availability</u>		Menu Item	Custom Field	<u>Menu Item</u> <u>Menu_Item_Highlights</u>
<u>Category</u>		Menu Item	Custom Field	<u>Menu Item</u> <u>Menu_Item_Highlights</u>
<u>Contact Number</u>		Restaurant	Custom Field	<u>Restaurant</u>
<u>Cuisine Type</u>		Restaurant	Custom Field	<u>Restaurant</u> <u>Restaurant_Highlights</u>
<u>Customer</u>		Tab		<u>FoodieConnect</u>
<u>Customer</u>		Custom Object		<u>Customer</u> All <u>OrderHandle</u>
<u>Customer</u>		Issue	Custom Field	<u>Issue</u> <u>Issue_Highlights</u>
<u>Customer</u>		Order	Custom Field	<u>Order</u> <u>Order_Highlights</u> <u>OrderHandle</u>
Customer Layout		Customer	Page Layout	<u>Customer</u>
<u>Customer Highlights</u>		Customer	Compact Layout	<u>Customer</u>
<u>Delivery</u>		Tab		<u>FoodieConnect</u>
<u>Delivery</u>		Custom Object		<u>Delivery</u> All
<u>Delivery Date</u>		Delivery	Custom Field	<u>Delivery</u>
Delivery Layout		Delivery	Page Layout	<u>Delivery</u>
<u>Delivery Number</u>		Delivery	Custom Field	<u>Delivery</u> <u>Delivery_Highlights</u>
<u>Delivery Status</u>		Delivery	Custom Field	<u>Delivery</u> <u>Delivery_Highlights</u>

Purpose:

Packaging bundles all necessary components (objects, fields, validation rules, Apex code) for reuse or distribution in other Salesforce orgs.

Detailed Steps Implemented:

- Created an Unmanaged Package named FoodieConnect_Unmanaged containing:
 - Core Objects: Customer, Order, Payment, Delivery, Restaurant, Menu Item, Issue.
 - Custom fields and validation rules enforcing business logic (e.g., positive order totals).
 - User interface elements: Page layouts, tabs, Lightning pages, and utility bar components.
 - Apex classes and triggers for business automation.
- Attempted to upload the package to the AppExchange or another org but failed due to missing dependent components (fields/layouts referenced in triggers and validation rules).

Reports

- **Tabular Reports:** Simple lists for Customers, Orders, Payments[reports.png].
- **Summary Reports:** Orders grouped by Status, Deliveries grouped by Rider[reports.png].
- **Matrix Reports:** Orders vs Restaurants matrix with total sales[reports.png].
- **Joined Reports:** Combined Customer & Order reports for cross-object insights[reports.png].

Outcome: Management efficiently tracks performance, deliveries, and payments.

30. Report Types

Custom Report Types were created for advanced cross-object insights:

- Customer with Orders
- Orders with Payments
- Delivery with Orders

Purpose: Enable complex reporting beyond standard Salesforce objects.

Status: Types deployed and available to all users.

31. Dashboards

Custom dashboards provide instant KPIs:

- **FoodieConnect Overview:** Orders, Payments, Customer Growth[reports.png].
- **Delivery Performance:** Deliveries by status and rider[reports.png].

Components: Charts, tables, metrics.

Outcome: Executives and managers get at-a-glance insights.

32. Dynamic Dashboards

- Displays data specific to the logged-in user's role.

- Example: Delivery staff see assigned deliveries; Admin views all data.

Benefit: Personalized, role-specific analytics without duplicating dashboards.

33. Sharing Settings

Organization-Wide Defaults (OWD):

- Customer: Public Read/Write
- Delivery: Public Read/Write
- Issue: Public Read/Write
- Menu Item: Public Read/Write
- Order: Controlled by Parent
- Payment: Public Read/Write
- Restaurant: Public Read/Write

Sharing Rules:

- Customers → Support Agents (Read/Write)
- Orders → Managers & Admins (Full Access)
- Payments → Managers (Full Access)

Outcome: Proper access and security across user roles.

Action	Criteria	Shared With	Access Level
Edit Del	Owner in Role: Rider	Role: food connect CEO	Read Only

34. Field Level Security

- **Admin:** Full field access
- **Support:** Customer & Issue fields
- **Riders:** Delivery and Order fields only
- **Customers:** Own customer record and order history

Sensitive fields (Payment info, Loyalty Points) locked down.

Customer Custom Field
Phone

[Back to Customer](#)

[Validation Rules \[0\]](#)

Custom Field Definition Detail		Edit	Set Field-Level Security	View Field Accessibility	Where is this used?
Field Information					
Field Label	Phone	Object Name	Customer		
Field Name	Phone	Data Type	Phone		
API Name	Phone__c				
Description					
Help Text					
Data Owner					
Field Usage					
Data Sensitivity Level					
Compliance Categorization					
Created By	mudhuluri lakshminarayana varma , 13/09/2025, 7:06 pm	Modified By	mudhuluri lakshminarayana varma 13/09/2025, 7:06 pm		

35. Session Settings

- Session Timeout: 2 hours
- Force logout on timeout: Enabled
- Lock sessions to domain/IP: Admins only
- Lightning Web Security: Enabled
- Multi-Factor Authentication: Admin & Managers (high assurance)

Benefit: Secures logins and prevents unauthorized access.

SETUP [Session Management](#)

[Help for this Page](#)

User Session Information

View information about or delete active user sessions. Location shows the approximate location of the IP address from where the user logged in. To show more geographic information, such as approximate city and postal code, create a custom view to include those fields. Due to the nature of geolocation technology, the accuracy of geolocation fields (for example, country, city, postal code) may vary.

[View: All](#) [Create New View](#)

Remove	Username	Session ID	Parent Session ID	Session Type	User Type	Login Type	Created	Updated	Valid Until	Source IP	Location
<input type="checkbox"/>	foodieconnect@food.com	0Akdm00000x00Ak		UI	Standard	Application	26/09/2025, 12:33 pm	26/09/2025, 12:39 pm	26/09/2025, 2:39 pm	103.168.80.48	India
<input type="checkbox"/>	foodieconnect@food.com	0Akdm00000x0Hxv	0Akdm00000x00Ak	Content	Standard	Application	26/09/2025, 12:33 pm	26/09/2025, 12:33 pm	26/09/2025, 2:33 pm	103.168.80.48	India
<input type="checkbox"/>	foodieconnect@food.com	0Akdm00000x0RL2	0Akdm00000x0TjB	InternalServiceCall	Standard	Application	26/09/2025, 12:23 pm	26/09/2025, 12:23 pm	26/09/2025, 2:23 pm	103.168.80.48	India
<input type="checkbox"/>	foodieconnect@food.com	0Akdm00000x0TjB		UI	Standard	Application	26/09/2025, 12:23 pm	26/09/2025, 12:30 pm	26/09/2025, 2:30 pm	103.168.80.48	India
<input type="checkbox"/>	foodieconnect@food.com	0Akdm00000x0TrX	0Akdm00000x00Ak	InternalServiceCall	Standard	Application	26/09/2025, 12:34 pm	26/09/2025, 12:33 pm	26/09/2025, 2:33 pm	103.168.80.48	India
<input type="checkbox"/>	foodieconnect@food.com	0Akdm00000x0WPP	0Akdm00000x00Ak	Aura	Standard	Application	26/09/2025, 12:33 pm	26/09/2025, 12:33 pm	26/09/2025, 2:33 pm	103.168.80.48	India

36. Login IP Ranges

- Admins:** Restricted to office IP range (e.g., 103.168.80.48)
- Delivery & Customers:** No IP restrictions (mobile access supported)

Admin logins protected; other users can access anywhere.

Login IP Ranges			Help (?)
Action	IP Start Address	IP End Address	Description
Edit Del	0.0.0.0	255.255.255.255	

37. Audit Trail

- Tracks changes across all setup activities:
 - Object creation (Customer, Order, Payment, Delivery, Issue)
 - Validation, Sharing, Duplicate Rules
 - Login IP range updates
- Audit Trail export available in CSV for reporting.

Full transparency of config changes and user actions.

38. Deployment

A custom Lightning App named FoodieConnect has been created with access to Restaurants, Menu Items, Customers, Orders, Deliveries, Payments, and Issues.

Customer Name	Actions
giri vardhan	View
harsha vardhan	View
k.Nikhilesh Reddy	View
M.Lakshmi Narayana varma	View