**Online Food Delivery CRM Project**

**Problem Statement:**

The food delivery industry handles thousands of daily orders from websites, apps, and partner channels. Without a centralized CRM, restaurants struggle to manage orders, assign deliveries, track status, and handle customer complaints. Delivery agents rely on manual updates, managers lack visibility, and executives don’t get real-time insights.

A **Salesforce-based Online Food Delivery CRM** is needed to:

* Automate order capture and assignment
* Enable real-time delivery tracking
* Centralize customer feedback and complaints
* Help managers monitor operations
* Provide executives with performance dashboards

**Phase 1: Problem Understanding & Industry Analysis**

**Requirement Gathering**

* Talk to stakeholders (restaurant managers, delivery agents, executives, customers).
* **Example requirements:**  
  - Track all restaurants and their menu items.  
  - Allow managers to assign orders to delivery agents.  
  - Prevent delivery conflicts (no double assignment).  
  - Capture customer complaints/feedback.  
  - Generate sales and delivery performance reports.

**Stakeholder Analysis**

* **Restaurant Managers** → Manage restaurants, menu items, orders, and customer complaints.
* **Delivery Agents** → Update delivery status, ensure timely order completion.
* **Executives / Management** → View performance dashboards, monitor KPIs.
* **System Administrator (CEO)** → Configure Salesforce org, oversee security and data integrity.

**Business Process Mapping**

* **Order Placement** → Customer places order → System captures order → Assigned to Restaurant Manager.
* **Delivery Assignment** → Restaurant Manager assigns order to Delivery Agent.
* **Delivery Update** → Delivery Agent updates status (Assigned → Out for Delivery → Delivered).
* **Feedback Handling** → Complaints/feedback logged and monitored by Manager.
* **Reporting** → Executives analyze performance metrics via dashboards.

**Industry-Specific Use Case Analysis**

* Real-time order tracking similar to Swiggy/Zomato.
* Menu & restaurant management within CRM.
* Delivery agent assignment and status updates.
* Customer complaints handling for service improvement.
* Performance dashboards for management review.

**AppExchange Exploration**

* Explore AppExchange packages:
* Food Delivery Management Apps for reference.
* Maps & Route Optimization apps for delivery tracking.
* SMS/Email Notification apps for customer communication.

......................................................................................................................................................

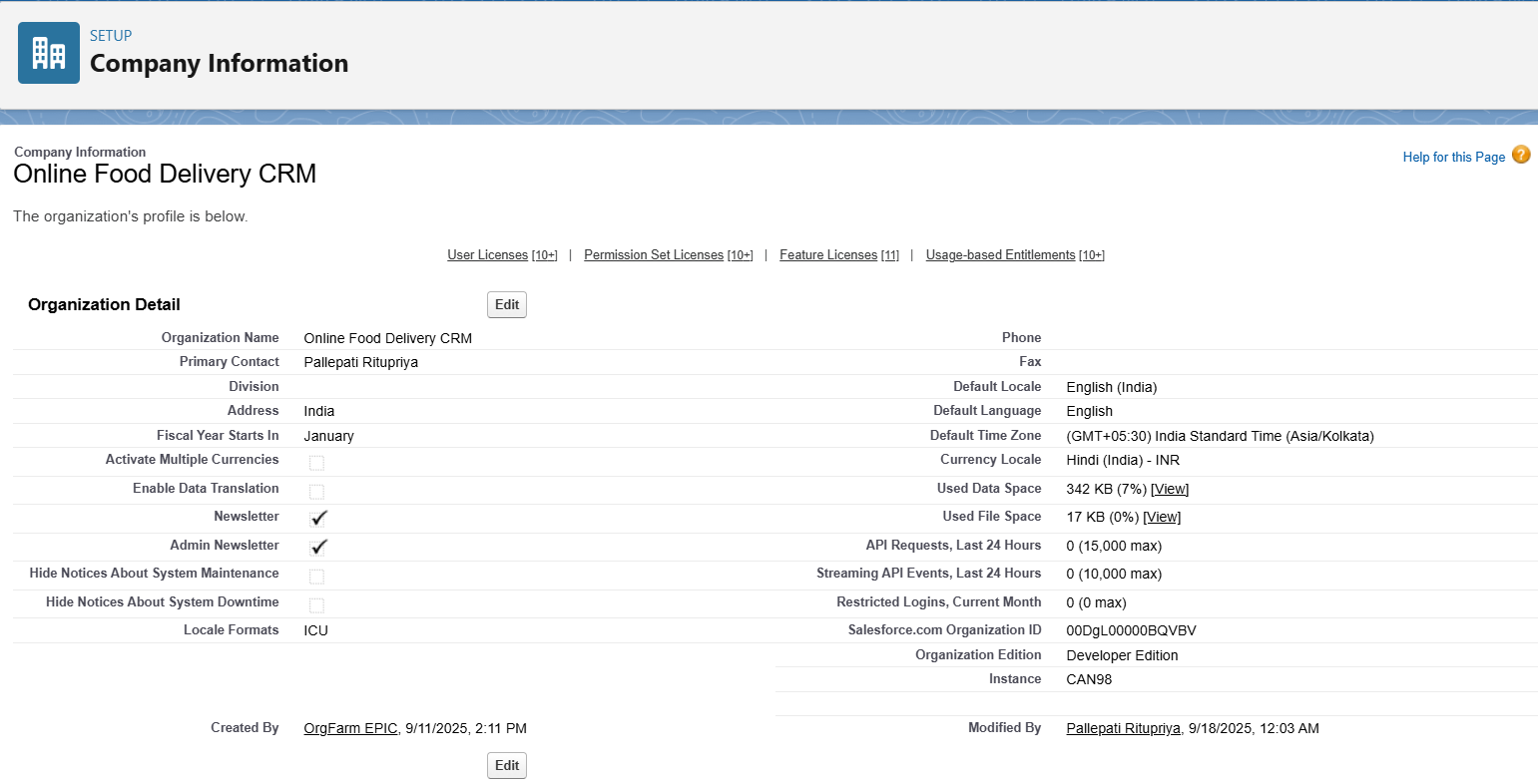
**Phase 2: Org Setup & Configuration**

**Salesforce Editions**

* Using Salesforce Developer Edition Org (free, permanent).
* Supports custom objects, flows, Apex, and dashboards.

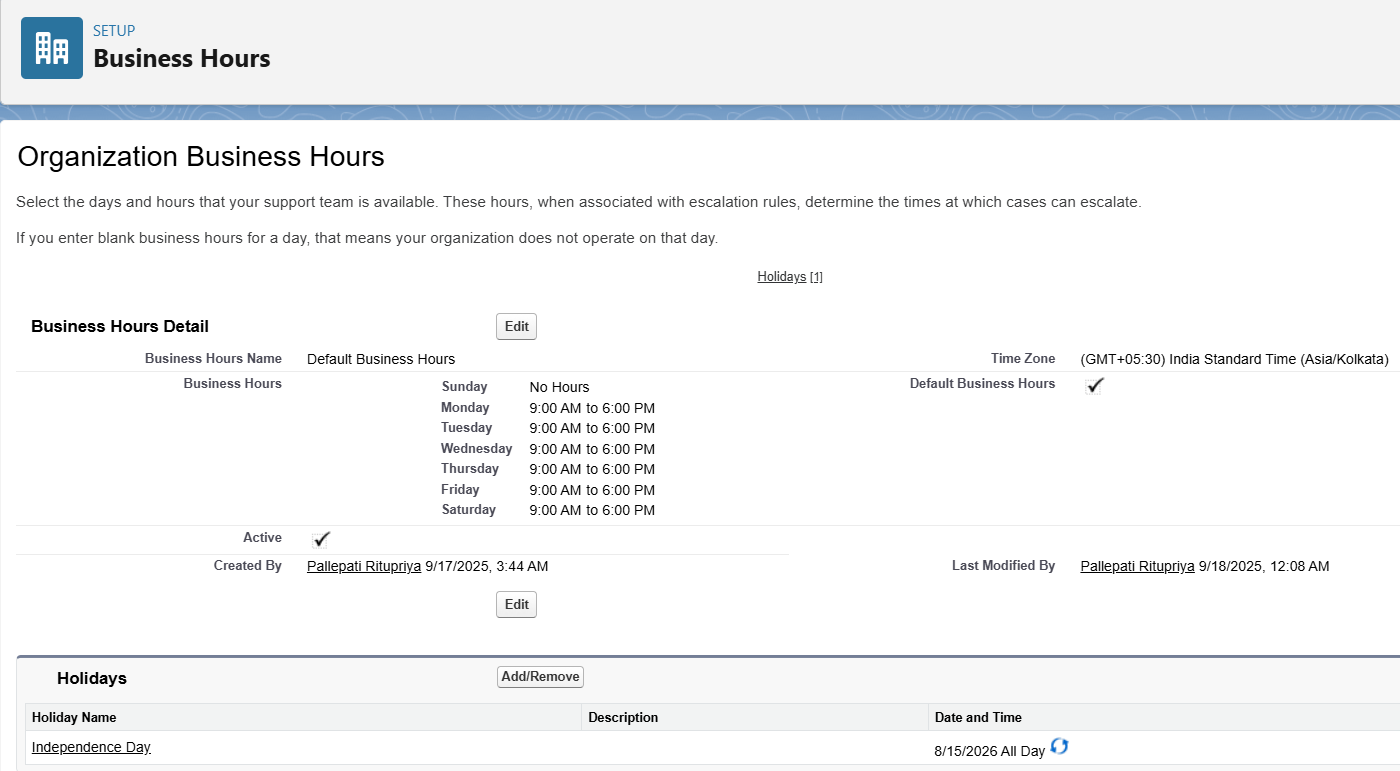
**Company Profile Setup**

* Setup → Company Information → Edit.
* **Company Name:** Online Food Delivery CRM
* **Default Locale:** English (India)
* **Time Zone:** Asia/Kolkata (IST)
* **Currency:** INR
* Saved changes for org-wide consistency.



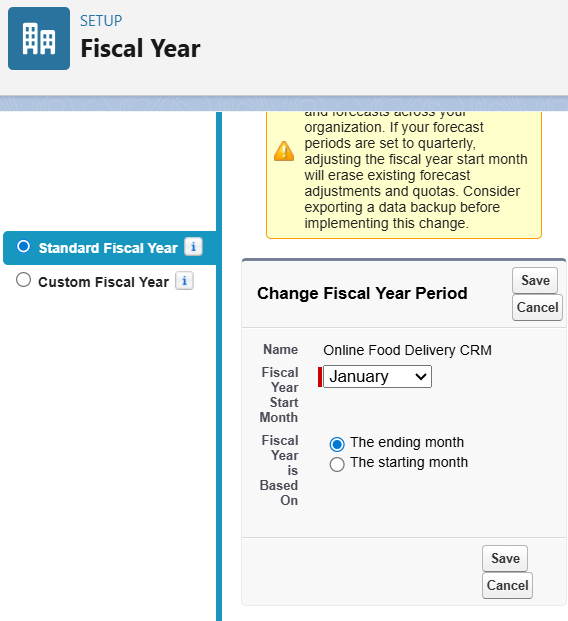
**Business Hours & Holidays**

* Setup → Business Hours → New Business Hours.
* **Working Hours:** 9:00 AM – 6:00 PM (Mon–Sat).
* Setup → Holidays → Add national holidays (e.g., 15th August – Independence Day).
* Ensures workflows/approvals don’t trigger during holidays.



**Fiscal Year Settings**

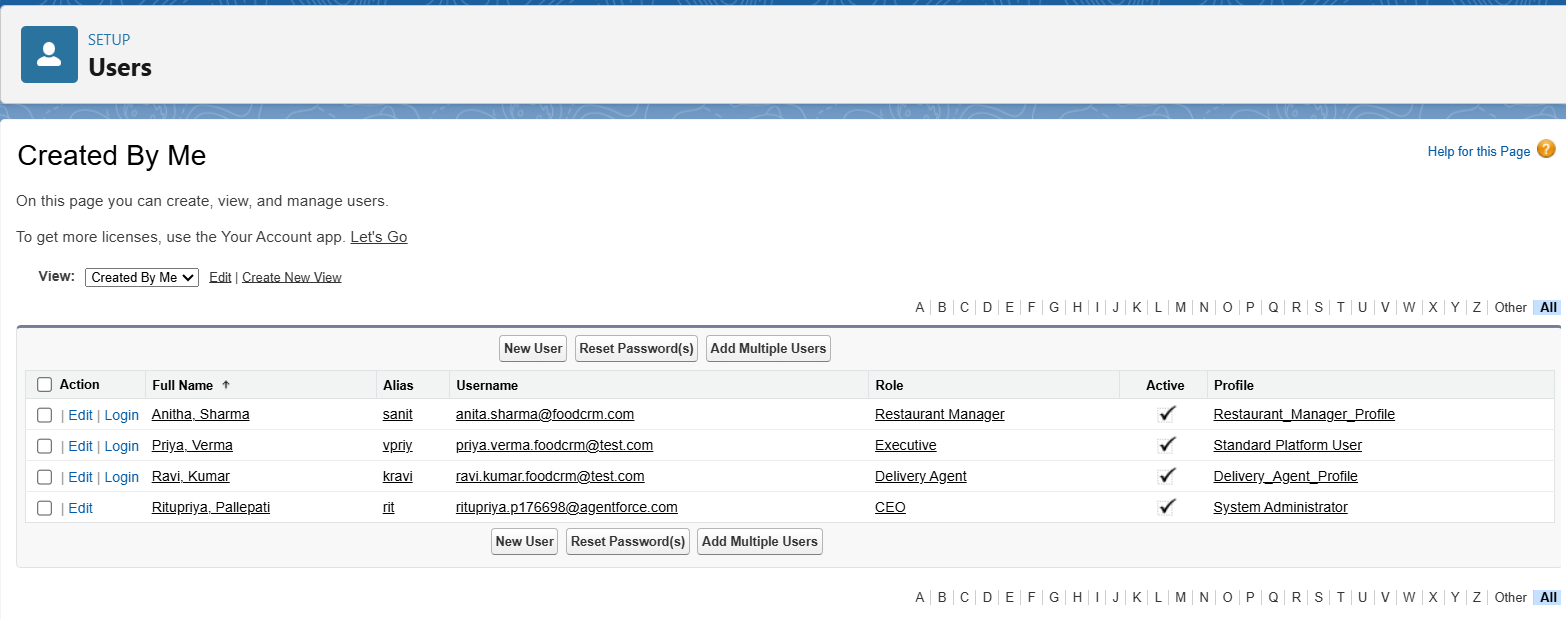
* Setup → Fiscal Year → Standard (Jan–Dec).
* Useful for revenue and sales reporting.



**User Setup & Licenses**

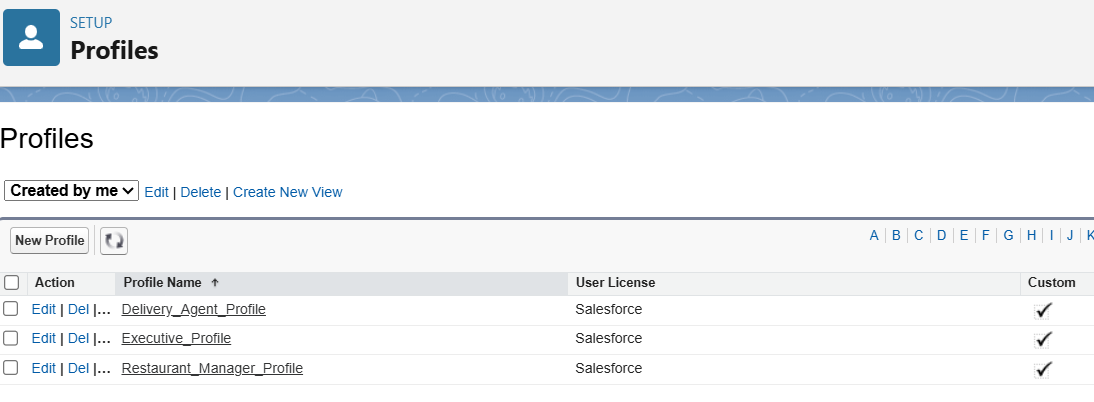
Created project-specific users:

* Me (Pallepati Ritupriya) = System Administrator (CEO) → Full access
* Anita Sharma (Restaurant Manager) → Salesforce License
* Ravi Kumar (Delivery Agent) → Salesforce License
* Priya Verma (Executive/Management) → Salesforce Platform License



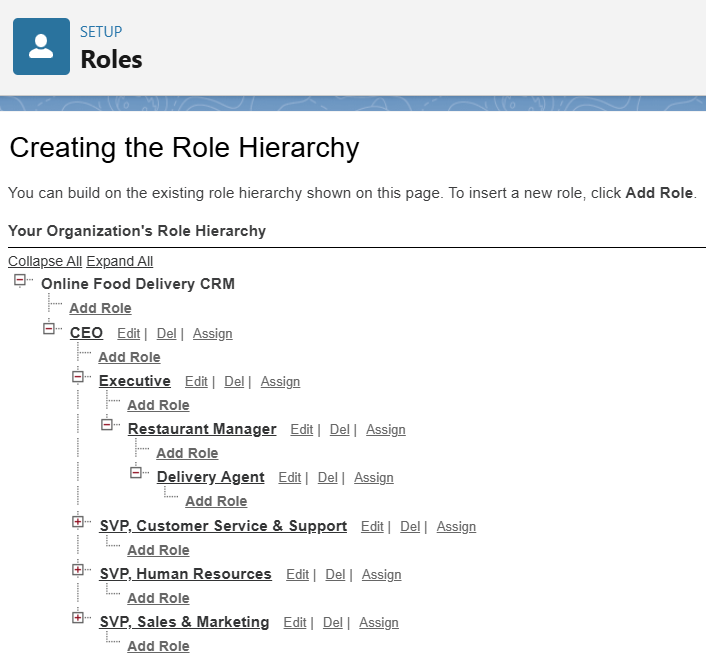
**Profiles**

* Cloned **Standard User** into custom profiles:
* Restaurant\_Manager\_Profile
* Delivery\_Agent\_Profile
* Executive\_Profile



**Roles**

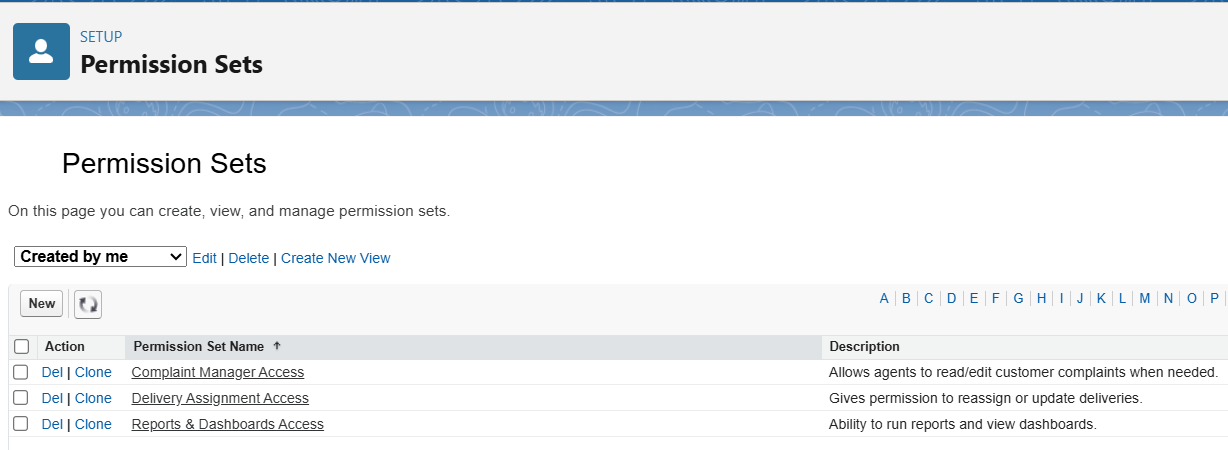
* **CEO (System Admin)** → Top-level, full access to all records.
* **Executive / Management** → Reports to CEO, sees all managers & agents records.
* **Restaurant Manager** → Reports to Executive, manages restaurants, menus, orders, and complaints.
* **Delivery Agent** → Reports to Manager, sees and updates only their assigned orders/deliveries.



**Permission Sets**

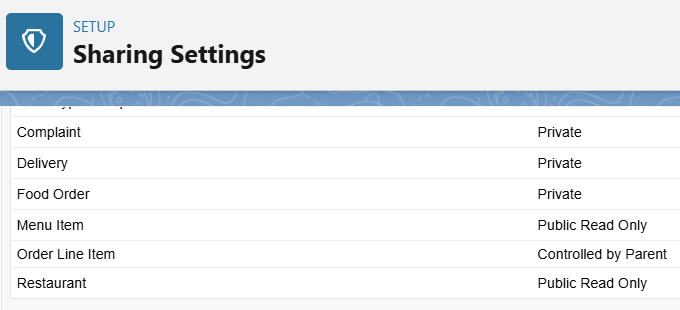
Permission sets provide extra access without changing user profiles.

* **Reports & Dashboards Access** → Grants executives (Priya Verma) ability to run reports and view dashboards.
* **Complaint Manager Access** → Allows delivery agents (Ravi Kumar) to read/edit customer complaints when needed.
* **Delivery Assignment Access** → Gives restaurant managers (Anita Sharma) permission to reassign or update deliveries.



**Organization-Wide Defaults (OWD)**

* Setup → Sharing Settings → Organization-Wide Defaults
* Configure default access for each object:
* **Restaurant\_\_c:** Public Read Only
* All users can view restaurants, but only managers/admins can edit.
* **Menu\_Item\_\_c:** Public Read Only
* All users can see menu items.
* **Food\_Orders\_\_c:** Private
* Only the owner (customer) and assigned staff can see the order.
* **Order\_Line\_Item\_\_c:** Controlled by Parent
* Inherits access from Food\_Orders\_\_c.
* **Delivery\_\_c:** Private
* Only assigned delivery agent and manager/admin can see.
* **Complaint\_\_c:** Private
* Only owner (customer) and support/admin can see.
* This ensures baseline record visibility and data security for all users.

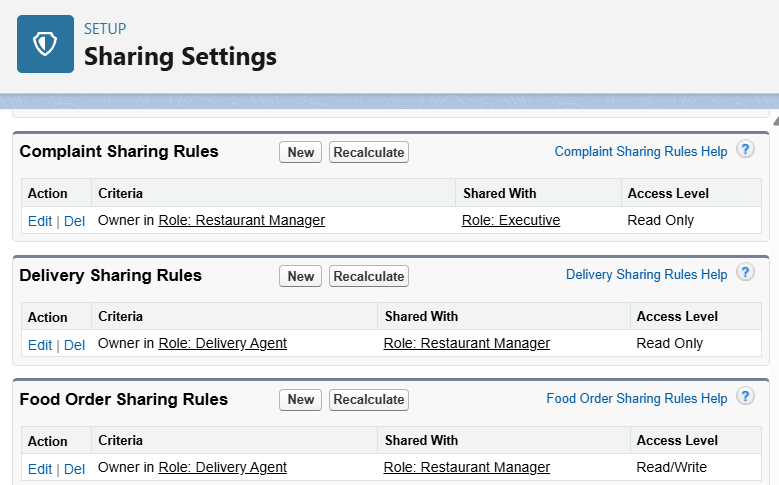


**Sharing Rules**

* Setup → Sharing Settings → Sharing Rules
* Sharing Rules grant additional access beyond OWD for specific business scenarios, ensuring that users in certain roles can view or edit records as needed.

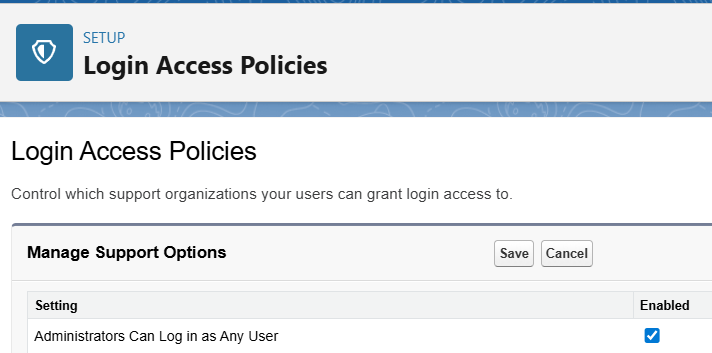
**Rules:**

1. **Complaint Sharing Rule**
   * **Access Level:** Read Only
   * Allows executives to view complaints raised by restaurant managers.
2. **Delivery Sharing Rule**
   * **Access Level:** Read Only
   * Enables restaurant managers to view deliveries assigned to their agents.
3. **Food Order Sharing Rule**
   * **Access Level:** Read/Write
   * Lets restaurant managers view and update orders handled by delivery agents.



**Login Access Policies**

* Setup → Login Access Policies.
* Enabled Administrators Can Log in as Any User.
* Allows Admin to test features as Manager, Agent, or Executive.



**Deployment Basics**

* Development in VS Code + Salesforce CLI.
* Metadata synced via package.xml.
* Source code stored in GitHub repo.
* **Deployment flow:**  
  VS Code (local changes) → Push to Salesforce Org → Commit to GitHub.

......................................................................................................................................................

**Phase 3: Data Modeling & Relationships**

**Standard & Custom Objects**

**Standard Objects:**

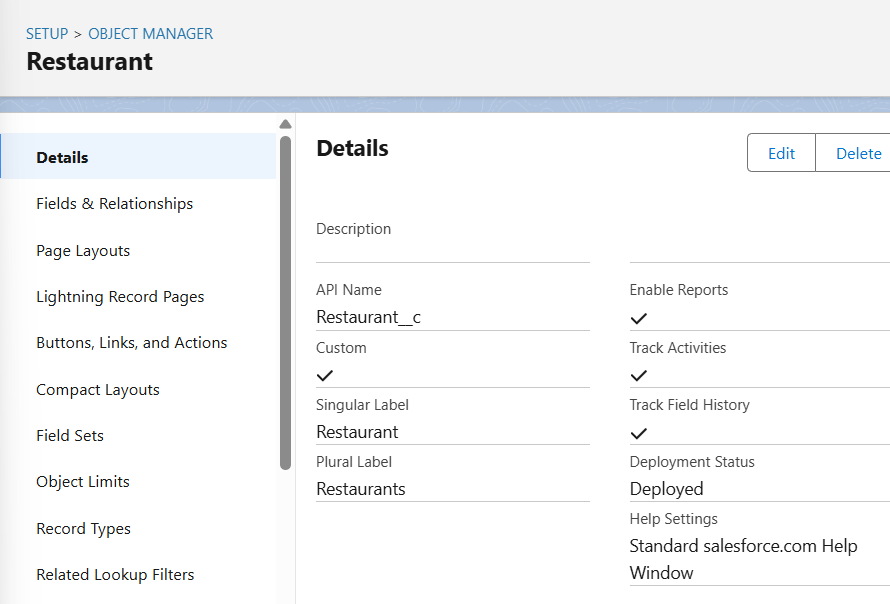
* **Contact:** Represents customers placing orders.
* **User:** Represents internal users like Restaurant Managers, Delivery Agents, and Executives.

**Custom Objects:**

* Setup → Object Manager → Create → Custom Object
* Used to manage business-specific data that is not captured by standard objects like Account, Contact, or User.

1. **Restaurant\_\_c**

**Purpose:**Stores information about restaurants in the Food Delivery CRM, including contact details, location, and cuisine type.

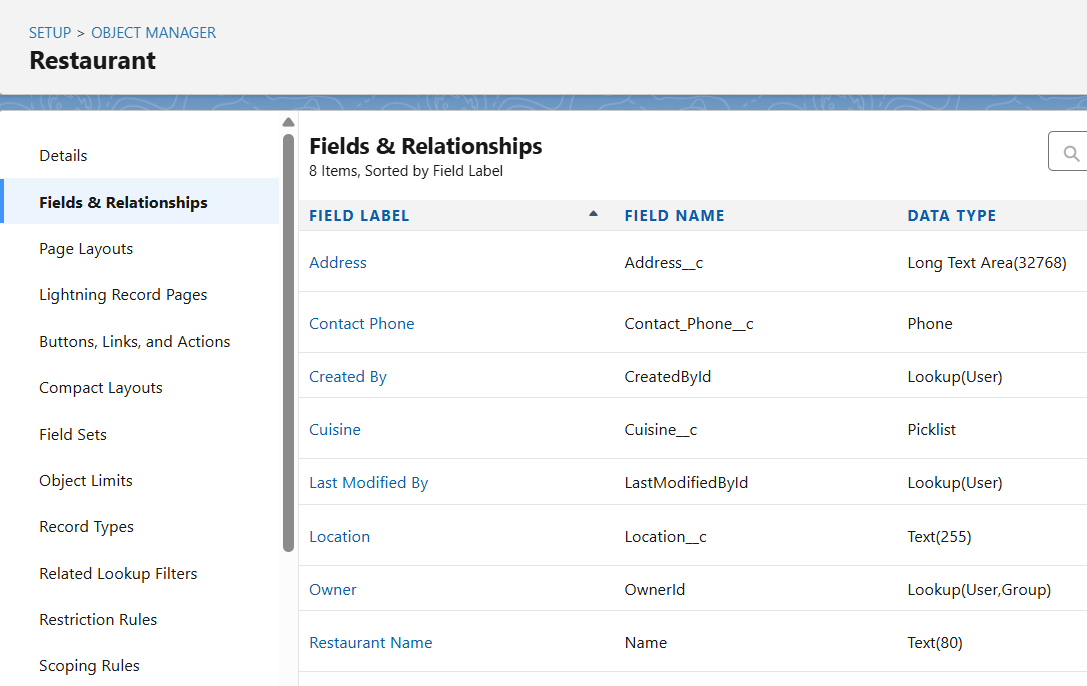
****

**Fields:**

* Restaurant Name (Name, Text 80): The name of the restaurant.
* Address\_\_c (Long Text Area 32768): Full address of the restaurant.
* Location\_\_c (Text 255): Short location or area name.
* Contact\_Phone\_\_c (Phone): Restaurant contact number.
* Cuisine\_\_c (Picklist): Type of cuisine offered (e.g., Indian, Italian).
* OwnerId (Lookup to User/Group): The user or group who owns the record.
* CreatedById (Lookup to User): System field indicating who created the record.
* LastModifiedById (Lookup to User): System field indicating who last modified the record.

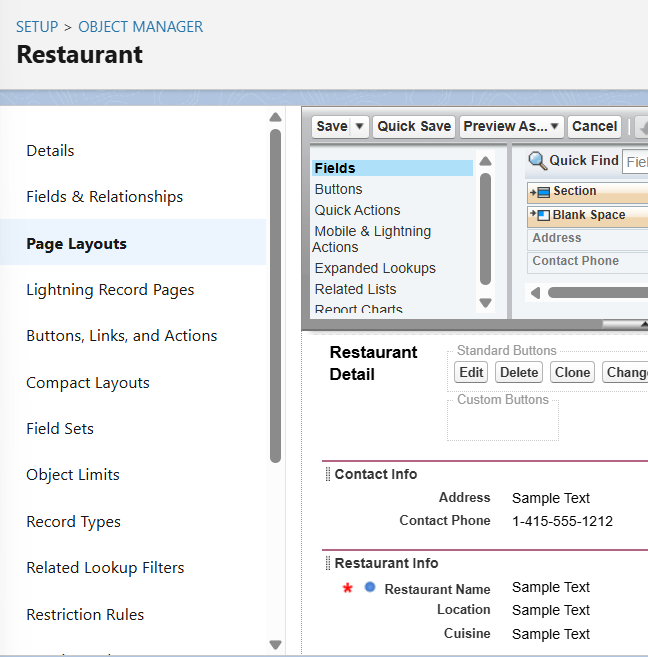
**Relationships:**

* Menu\_Item\_\_c → Restaurant\_\_c (Lookup): Links menu items to the restaurant.
* Food\_Orders\_\_c → Restaurant\_\_c (Lookup): Links orders to the restaurant.



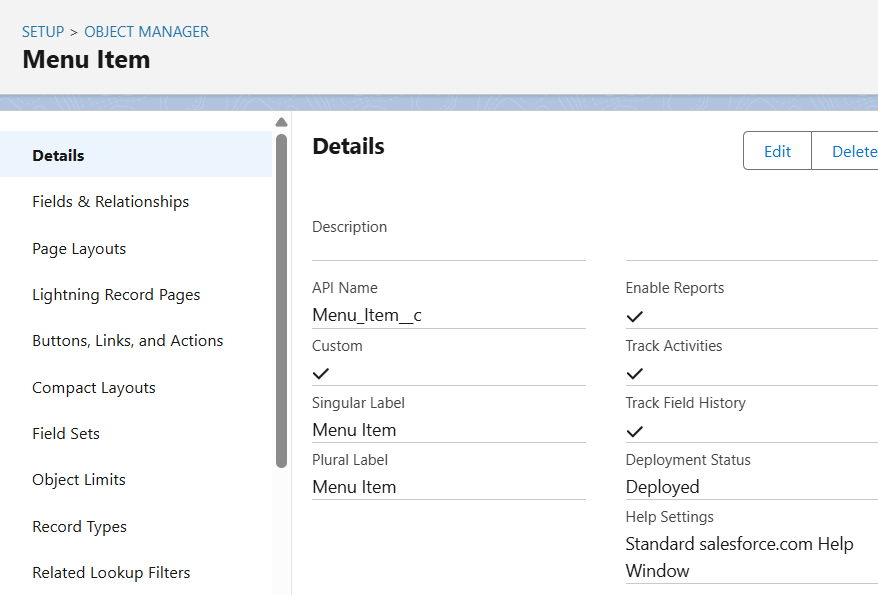
**Page Layouts:**

* Contact Info Section: Address, Contact Phone
* Restaurant Info Section: Restaurant Name, Location, Cuisine



1. **Menu\_Item\_\_c**

**Purpose:**  
Stores details of food items offered by restaurants, including price, category, availability, and description.

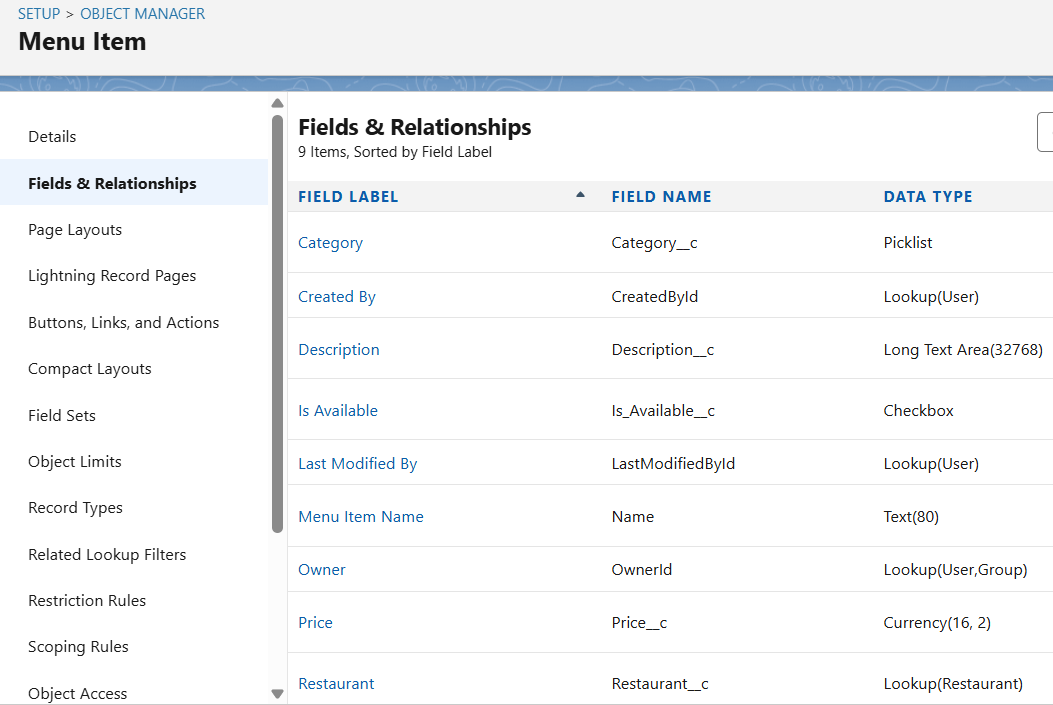


**Fields:**

* Menu Item Name (Name, Text 80): Name of the menu item.
* Price\_\_c (Currency 16,2): Price of the menu item.
* Category\_\_c (Picklist): Type/category of the food item (e.g., Appetizer, Main Course).
* Description\_\_c (Long Text Area 32768): Detailed description of the item.
* Is\_Available\_\_c (Checkbox): Indicates if the item is currently available.
* Restaurant\_\_c (Lookup to Restaurant\_\_c): Associated restaurant for the menu item.
* OwnerId (Lookup to User/Group): Owner of the record.
* CreatedById (Lookup to User): System field indicating who created the record.
* LastModifiedById (Lookup to User): System field indicating who last modified the record.

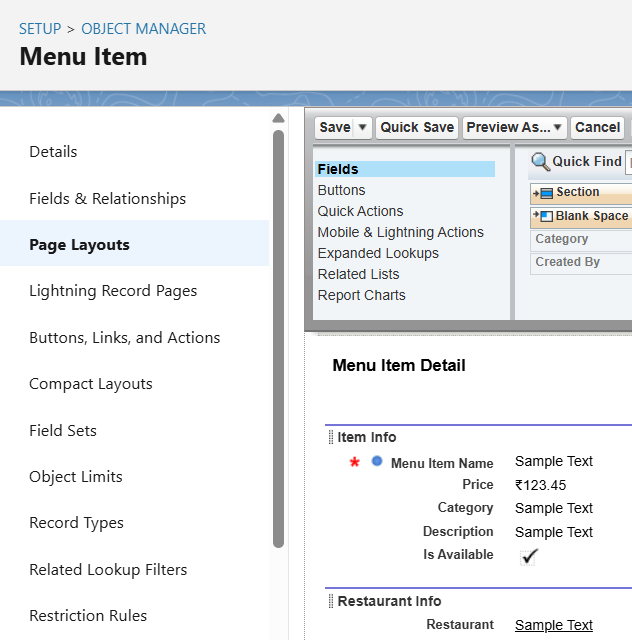
**Relationships:**

* Menu\_Item\_\_c → Restaurant\_\_c (Lookup): Links menu items to the restaurant.
* Order\_Line\_Item\_\_c → Menu\_Item\_\_c (Lookup): Links line items to menu items in orders.



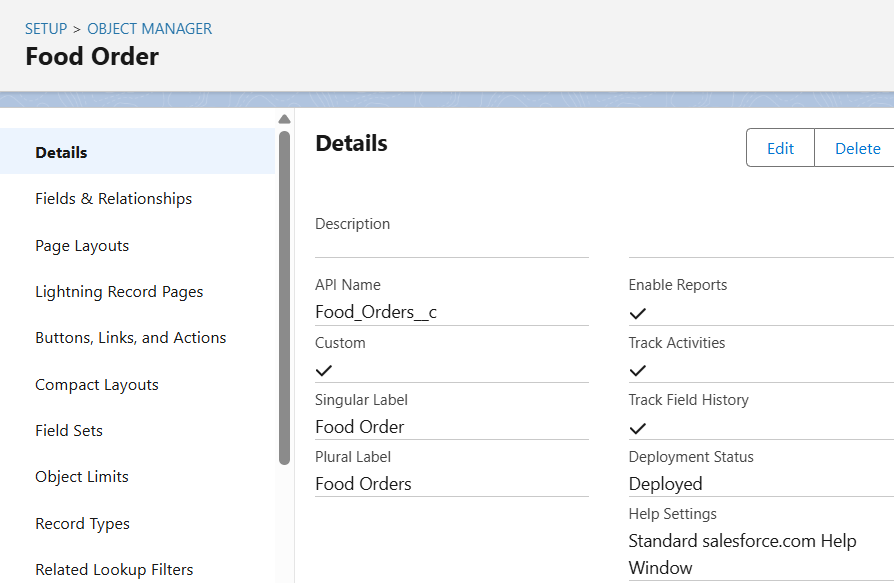
**Page Layouts:**

* Item Info Section: Menu Item Name, Price, Category, Description, Is Available
* Restaurant Info Section: Restaurant



1. **Food\_Orders\_\_c**

**Purpose:**  
Represents customer orders in the Food Delivery CRM, including order details, customer information, restaurant, and payment status.

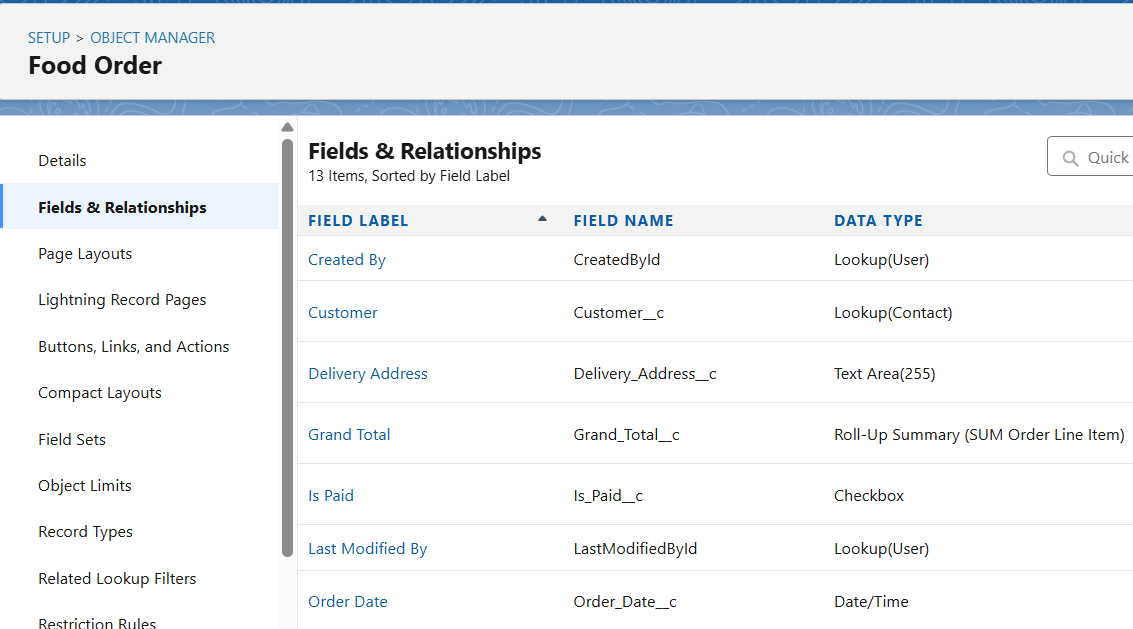


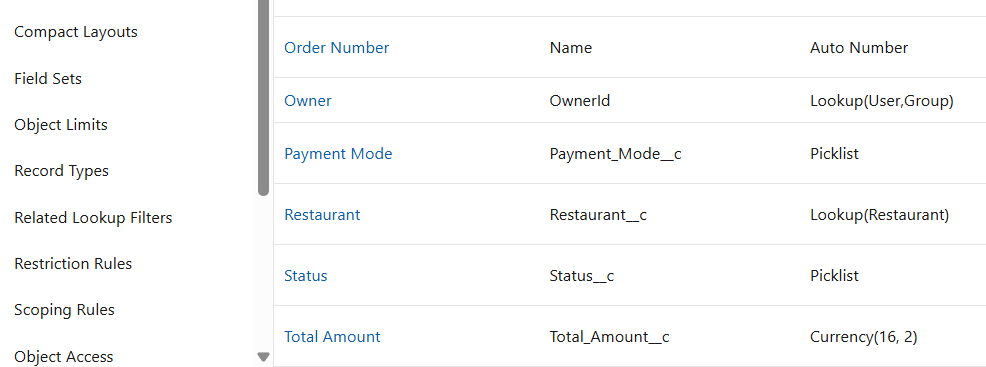
**Fields:**

* Order Number (Name, Auto Number): Unique identifier for the order.
* Customer\_\_c (Lookup to Contact): Customer who placed the order.
* Delivery\_Address\_\_c (Text Area 255): Address where the order should be delivered.
* Restaurant\_\_c (Lookup to Restaurant\_\_c): Restaurant fulfilling the order.
* Order\_Date\_\_c (Date/Time): Date and time when the order was placed.
* Status\_\_c (Picklist): Current status of the order (e.g., Pending, Completed).
* Total\_Amount\_\_c (Currency 16,2): Total amount for the order.
* Grand\_Total\_\_c (Roll-Up Summary, SUM of Order Line Items): Total of all line items.
* Payment\_Mode\_\_c (Picklist): Mode of payment (e.g., Cash, Card, Online).
* Is\_Paid\_\_c (Checkbox): Indicates whether the order has been paid.
* OwnerId (Lookup to User/Group): Owner of the record.
* CreatedById (Lookup to User): System field indicating who created the record.
* LastModifiedById (Lookup to User): System field indicating who last modified the record.

**Relationships:**

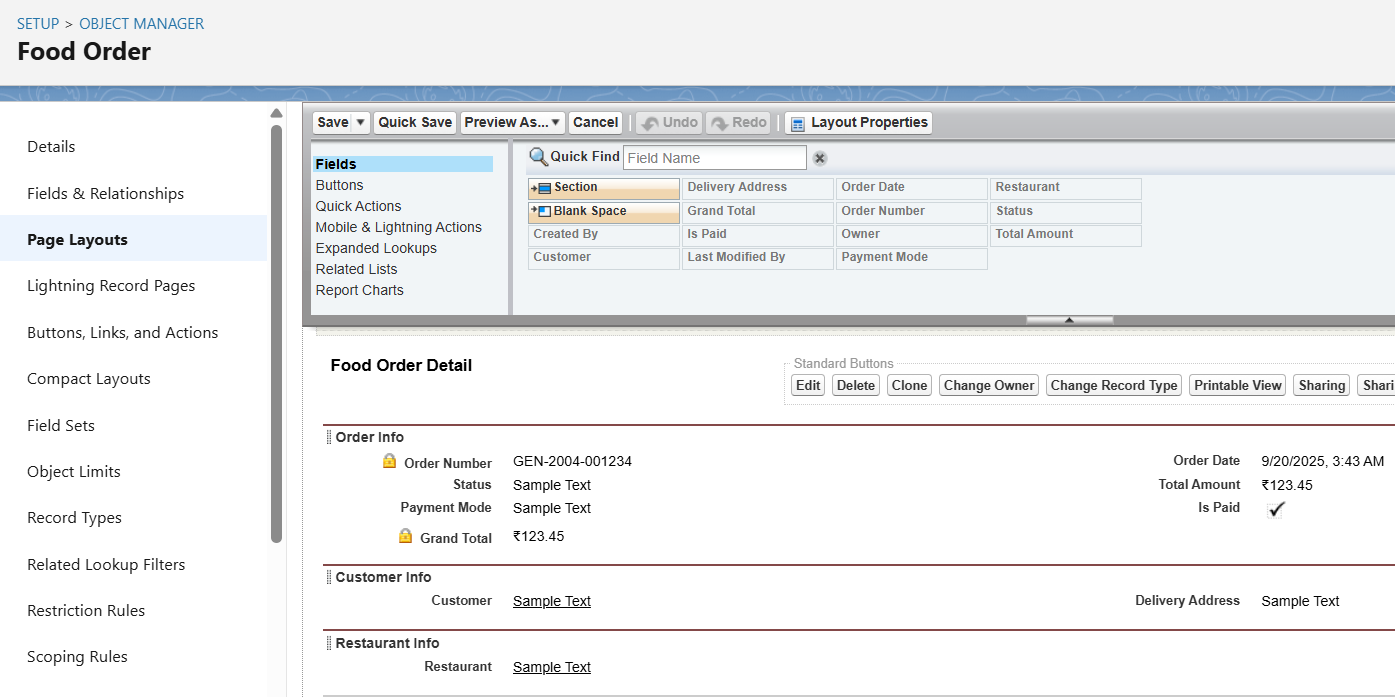
* Food\_Orders\_\_c → Customer\_\_c (Lookup to Contact): Links order to customer.
* Food\_Orders\_\_c → Restaurant\_\_c (Lookup): Links order to restaurant.
* Order\_Line\_Item\_\_c → Food\_Orders\_\_c (Master-Detail): Links line items to the parent order.
* Delivery\_\_c → Food\_Orders\_\_c (Lookup): Links delivery record to the order.
* Complaint\_\_c → Food\_Orders\_\_c (Lookup): Links complaints to the order.





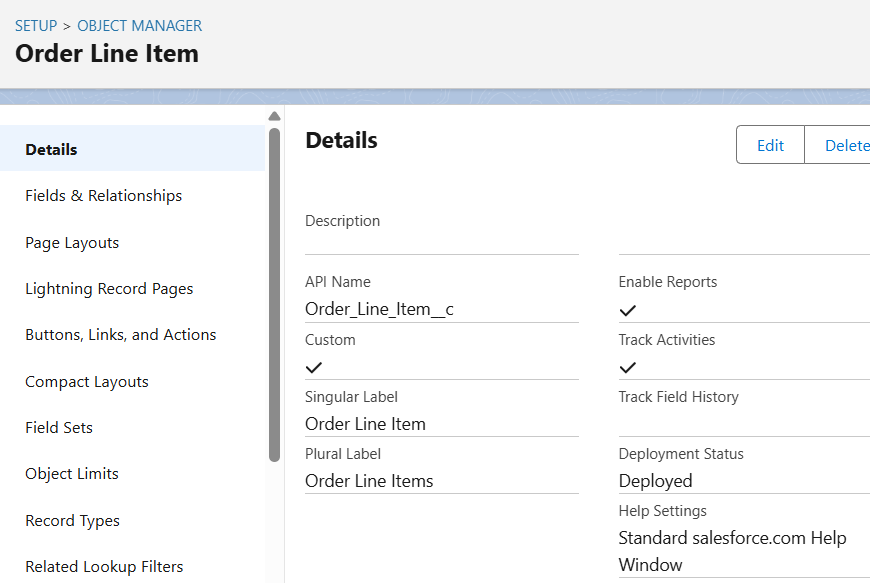
**Page Layouts:**

* Order Info Section: Order Number, Order Date, Status, Total Amount, Payment Mode, Is Paid, Grand Total
* Customer Info Section: Customer, Delivery Address
* Restaurant Info Section: Restaurant



1. **Order\_Line\_Item\_\_c**

**Purpose:**  
Represents individual items within a food order, including the menu item, quantity, pricing, and calculated total.

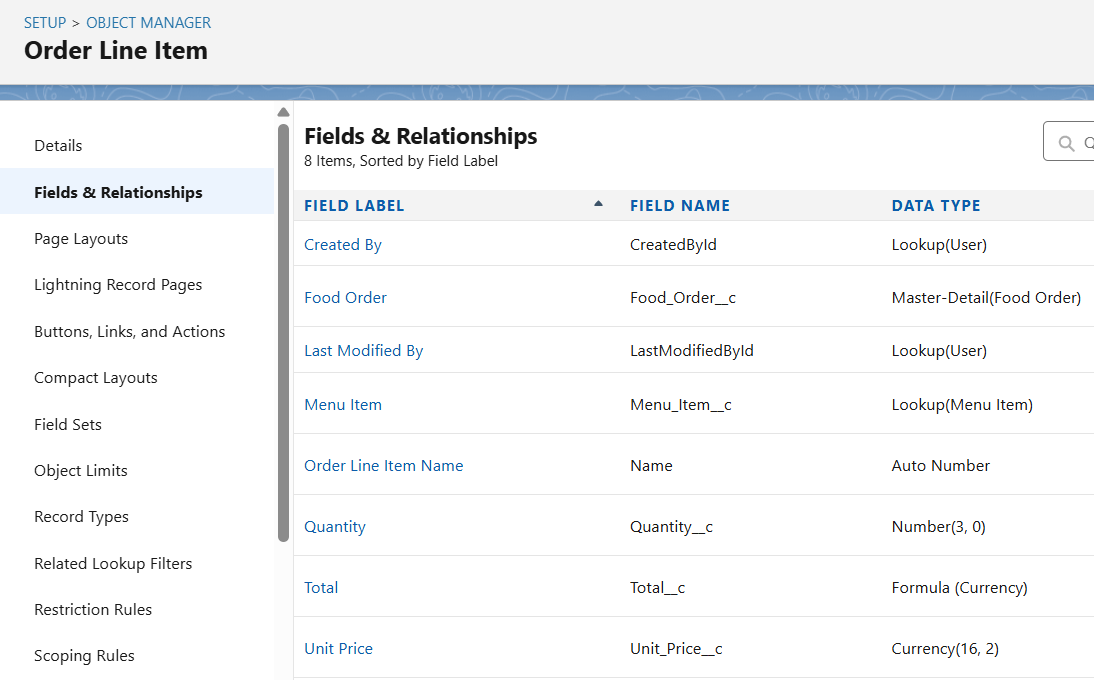


**Fields:**

* Order Line Item Name (Name, Auto Number): Unique identifier for the line item.
* Food\_Order\_\_c (Master-Detail to Food\_Orders\_\_c): Parent order to which this line item belongs.
* Menu\_Item\_\_c (Lookup to Menu\_Item\_\_c): Menu item included in this line item.
* Quantity\_\_c (Number 3,0): Number of units ordered.
* Unit\_Price\_\_c (Currency 16,2): Price per unit of the menu item.
* Total\_\_c (Formula, Currency): Calculated total for this line item (Quantity × Unit Price).
* OwnerId (Lookup to User/Group): Owner of the record.
* CreatedById (Lookup to User): System field indicating who created the record.
* LastModifiedById (Lookup to User): System field indicating who last modified the record.

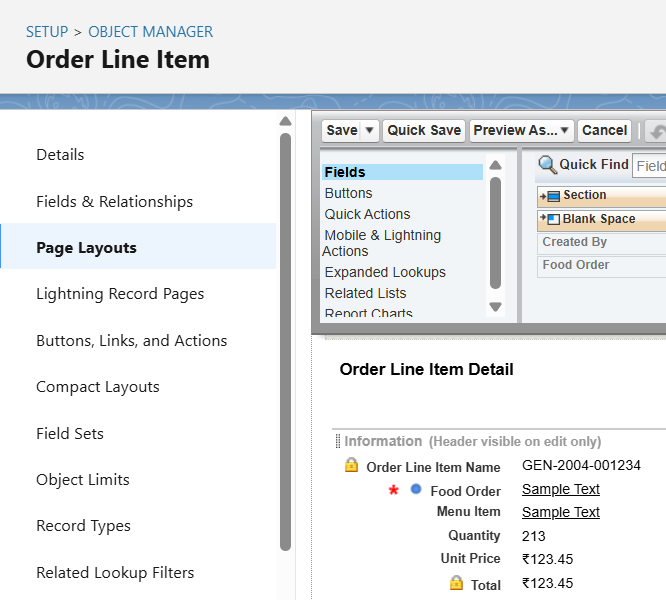
**Relationships:**

* Order\_Line\_Item\_\_c → Food\_Orders\_\_c (Master-Detail): Links line item to parent food order.
* Order\_Line\_Item\_\_c → Menu\_Item\_\_c (Lookup): Links line item to the menu item being ordered.



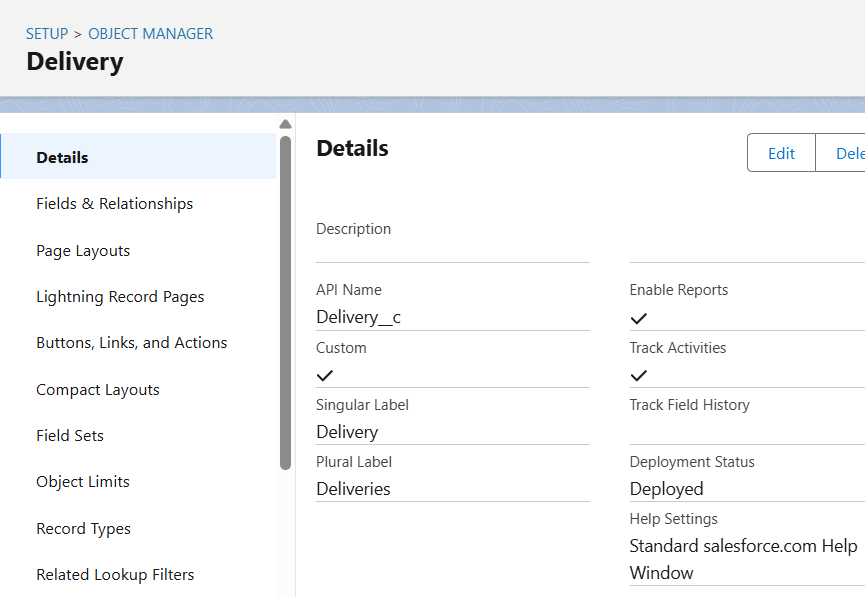
**Page Layouts:**

* Information Section: Order Line Item Name, Food Order, Menu Item, Quantity, Unit Price, Total



1. **Delivery\_\_c**

**Purpose:**  
Tracks the delivery details of food orders, including assigned delivery agent, estimated time of arrival (ETA), and delivery status.

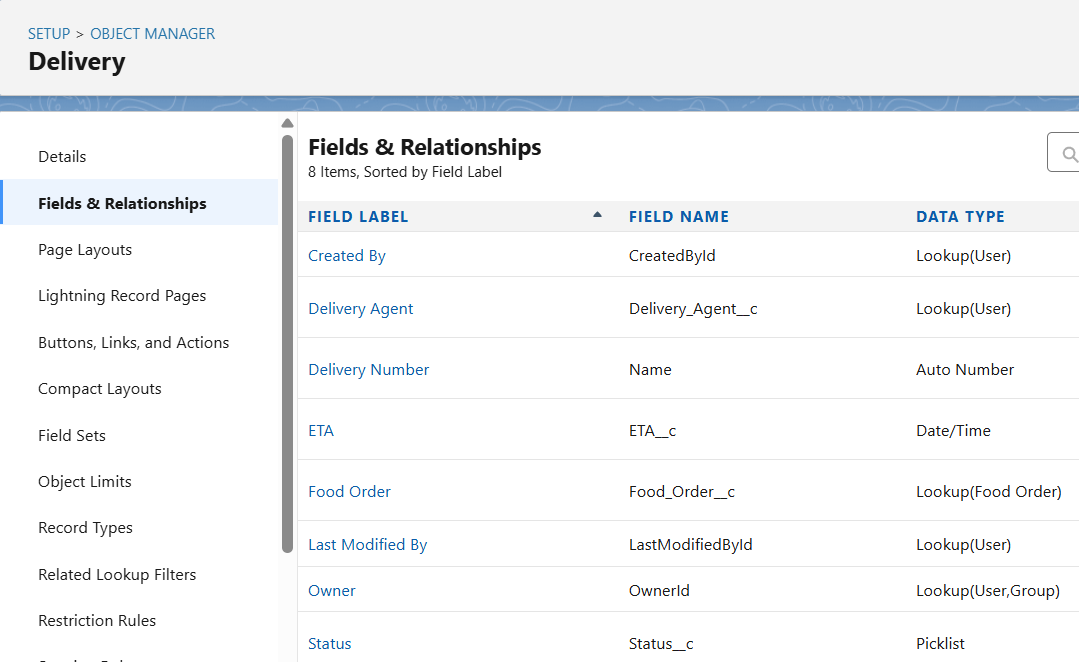


**Fields:**

* Delivery Number (Name, Auto Number): Unique identifier for each delivery.
* Delivery\_Agent\_\_c (Lookup to User): Delivery agent assigned to this delivery.
* Food\_Order\_\_c (Lookup to Food\_Orders\_\_c): Links the delivery to its associated food order.
* ETA\_\_c (Date/Time): Estimated time of arrival for the delivery.
* Status\_\_c (Picklist): Current status of the delivery (e.g., Assigned, Out for Delivery, Delivered).
* OwnerId (Lookup to User/Group): The owner of the record.
* CreatedById (Lookup to User): System field indicating who created the record.
* LastModifiedById (Lookup to User): System field indicating who last modified the record.

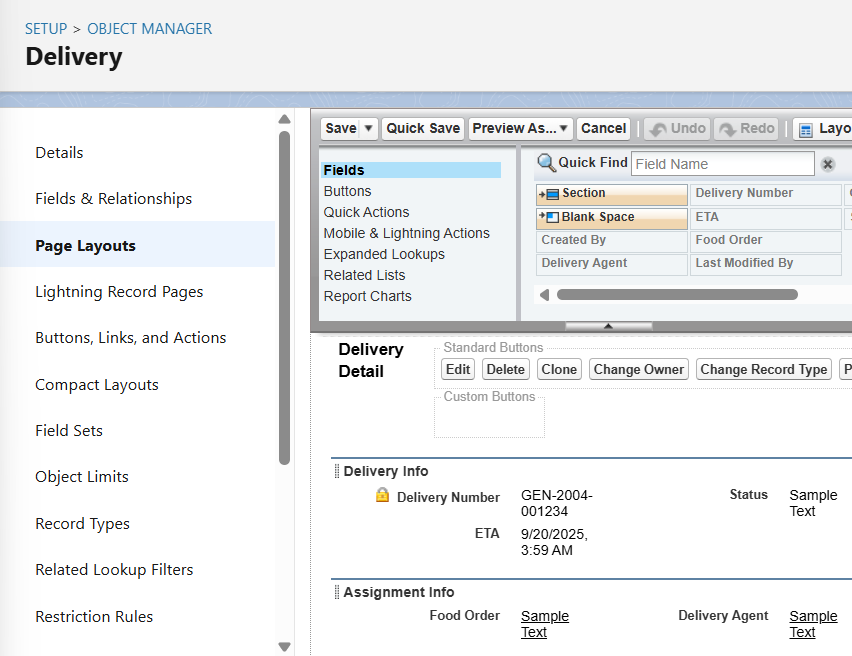
**Relationships:**

* Delivery\_\_c → Food\_Orders\_\_c (Lookup): Associates each delivery with a specific food order.
* Delivery\_\_c → User (Lookup): Links the delivery record to the assigned delivery agent.



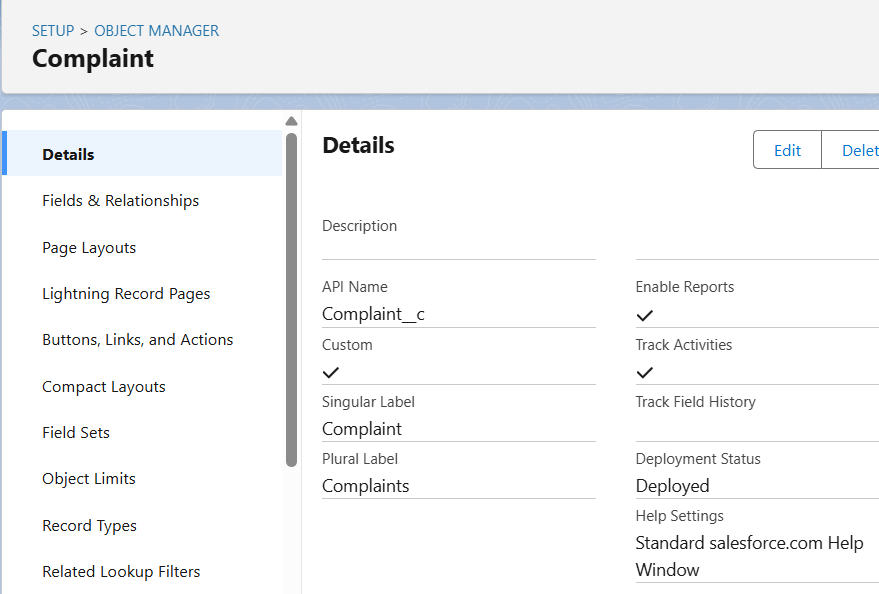
**Page Layouts:**

* Delivery Info Section: Delivery Number, Status, ETA
* Assignment Info Section: Food Order, Delivery Agent



1. **Complaint\_\_c**

**Purpose:**  
Captures customer complaints related to food orders, including complaint type, status, and customer feedback.

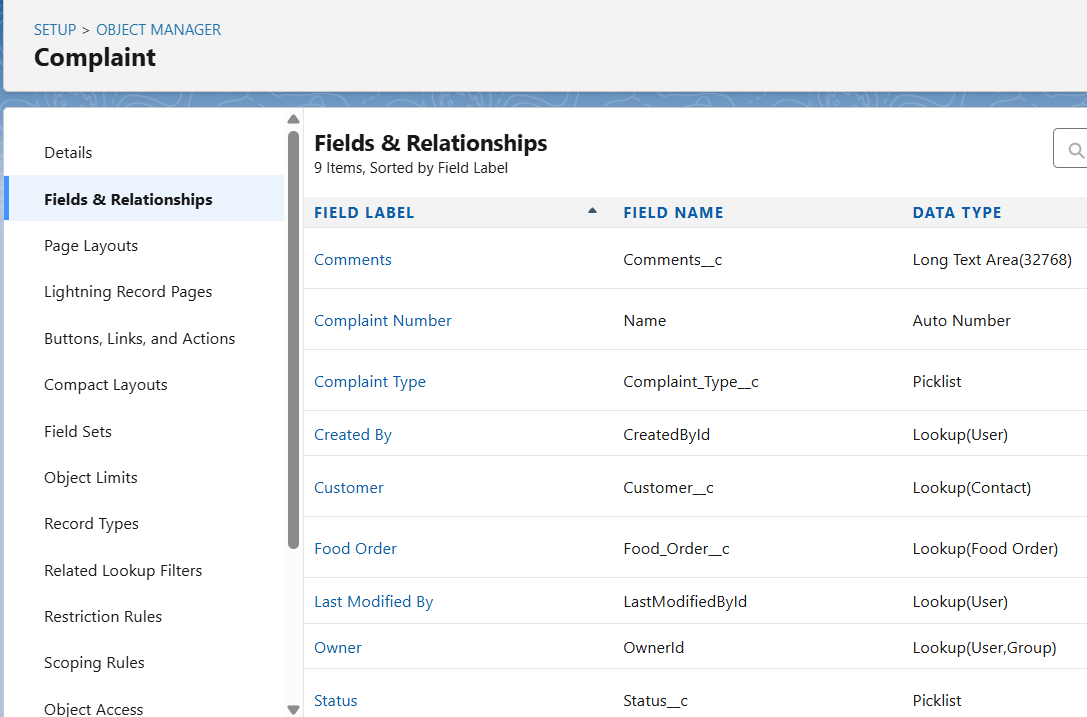


**Fields:**

* Complaint Number (Name, Auto Number): Unique identifier for each complaint.
* Complaint\_Type\_\_c (Picklist): Type of complaint (e.g., Late Delivery, Wrong Item, Quality Issue).
* Status\_\_c (Picklist): Current status of the complaint (e.g., Open, In Progress, Resolved, Closed).
* Comments\_\_c (Long Text Area 32768): Detailed description or comments provided by the customer.
* Food\_Order\_\_c (Lookup to Food\_Orders\_\_c): Links the complaint to the related food order.
* Customer\_\_c (Lookup to Contact): Customer who raised the complaint.
* OwnerId (Lookup to User/Group): Owner of the record (e.g., Support Agent).
* CreatedById (Lookup to User): System field indicating who created the record.
* LastModifiedById (Lookup to User): System field indicating who last modified the record.

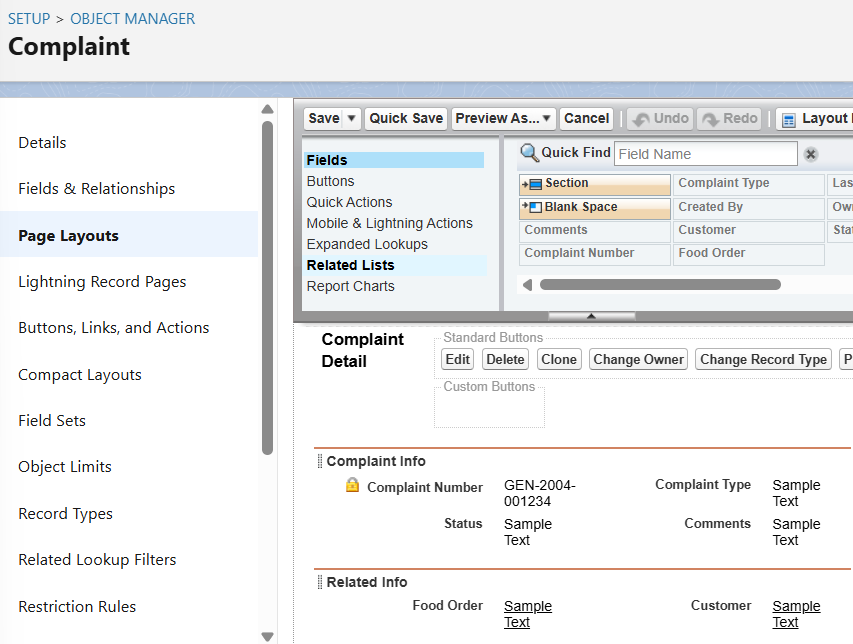
**Relationships:**

* Complaint\_\_c → Food\_Orders\_\_c (Lookup): Associates the complaint with a specific order.
* Complaint\_\_c → Contact (Lookup): Links the complaint to the customer who submitted it.



**Page Layouts:**

* Complaint Info Section: Complaint Number, Complaint Type, Status, Comments
* Related Info Section: Food Order, Customer

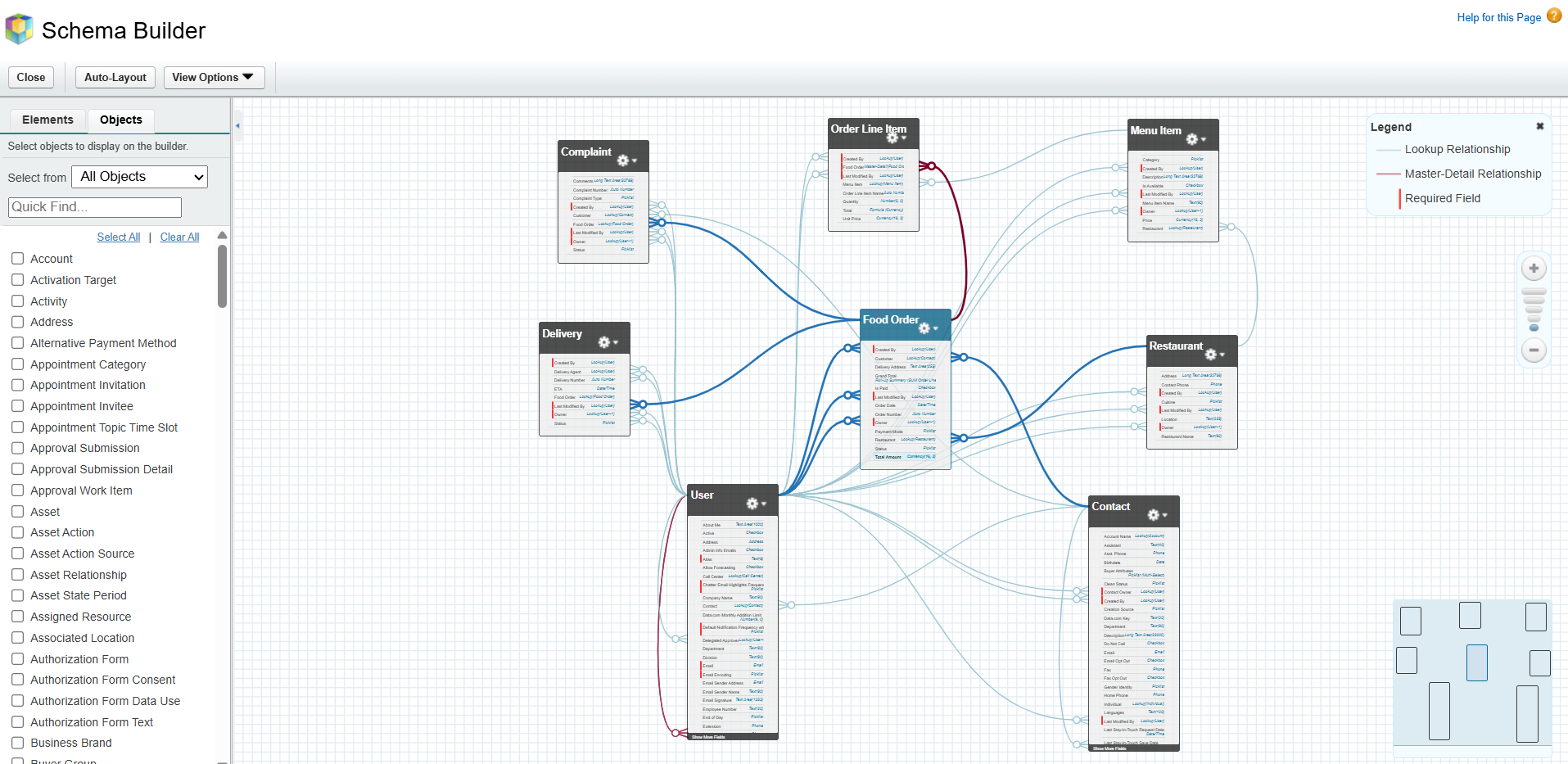


**Lookup vs Master-Detail vs Hierarchical Relationships**

* **Restaurant\_\_c ↔ Menu\_Item\_\_c → Lookup**  
  A menu item is linked to a restaurant, but deleting a restaurant doesn’t automatically delete its menu items.
* **Food\_Orders\_\_c ↔ Restaurant\_\_c → Lookup**  
  An order is associated with a restaurant, but restaurants don’t own the orders.
* **Food\_Orders\_\_c ↔ Contact (Customer) → Lookup**  
  Orders are linked to customers (Contacts), but customers don’t “own” the order records.
* **Food\_Orders\_\_c ↔ Order\_Line\_Item\_\_c → Master-Detail**  
  Line items cannot exist without a parent order. Deleting a Food Order deletes all related Order Line Items.
* **Delivery\_\_c ↔ Food\_Orders\_\_c → Lookup**  
  Each delivery record is linked to an order, but the delivery doesn’t own the order.
* **Complaint\_\_c ↔ Food\_Orders\_\_c → Lookup**  
  Complaints are tied to specific orders, but they can exist independently.
* **Complaint\_\_c ↔ Contact (Customer) → Lookup**  
  Complaints are linked to the customer who raised them, but customers don’t own the complaints.

**Schema Builder**

* Setup → Object Manager → Schema Builder
* A visual tool in Salesforce that lets you view and manage objects, fields, and relationships in a single interface.



......................................................................................................................................................

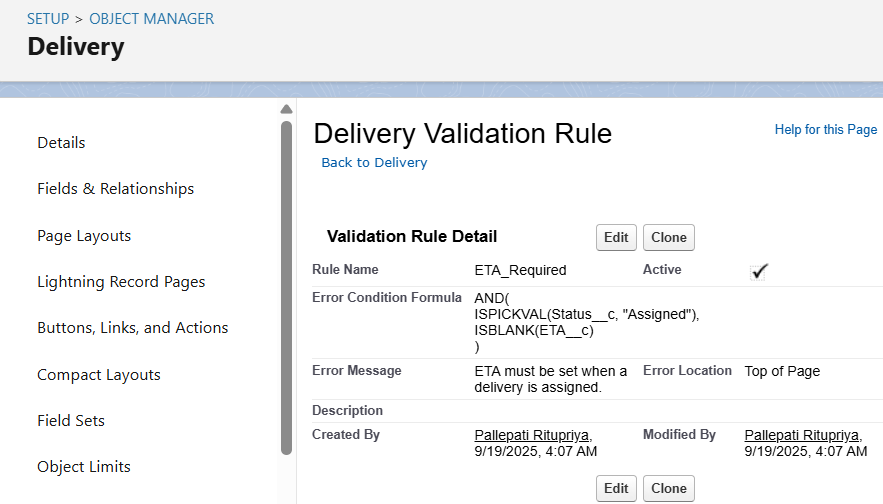
**Phase 4: Process Automation (Admin)**

**Validation Rules  
•** Setup → Object Manager → Select Object → Validation Rules  
• Validation Rules enforce data integrity and business logic, preventing incomplete or invalid data entry.

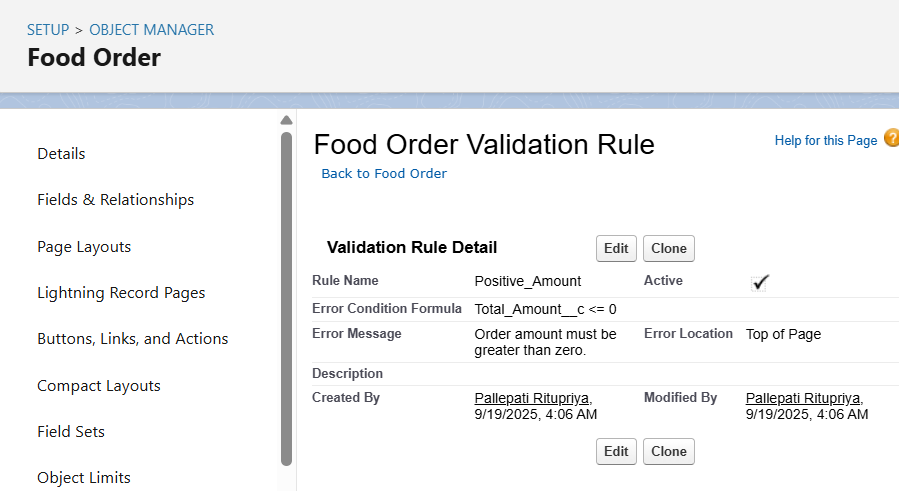
**Rules:**

1. **Delivery – ETA\_Required**

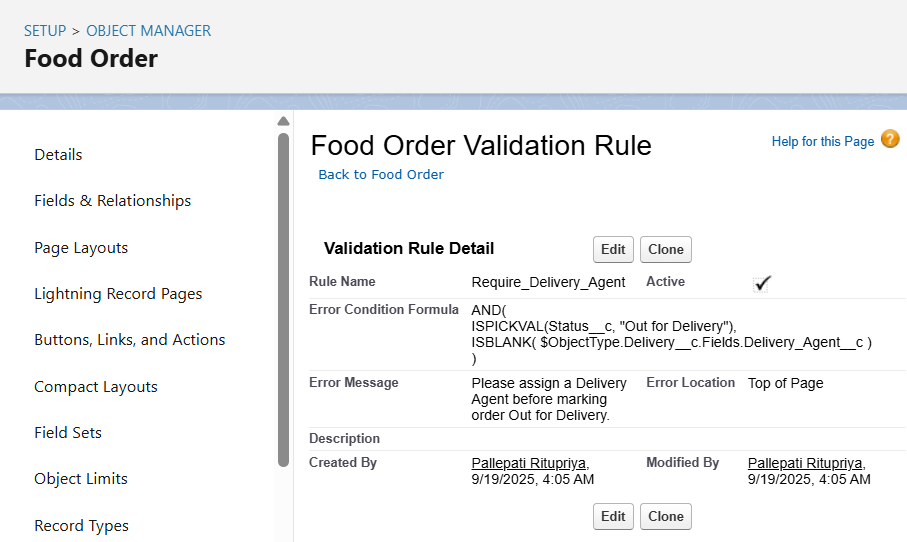
• Error Condition Formula: AND(ISPICKVAL(Status\_\_c, "Assigned"), ISBLANK(ETA\_\_c))  
• Error Message: ETA must be set when a delivery is assigned.



1. **Food Order – Positive\_Amount**  
   • Error Condition Formula: Total\_Amount\_\_c <= 0  
   • Error Message: Order amount must be greater than zero.



1. **Food Order – Require\_Delivery\_Agent**  
   • Error Condition Formula: AND(ISPICKVAL(Status\_\_c, "Out for Delivery"), ISBLANK($ObjectType.Delivery\_\_c.Fields.Delivery\_Agent\_\_c))  
   • Error Message: Please assign a Delivery Agent before marking order Out for Delivery.

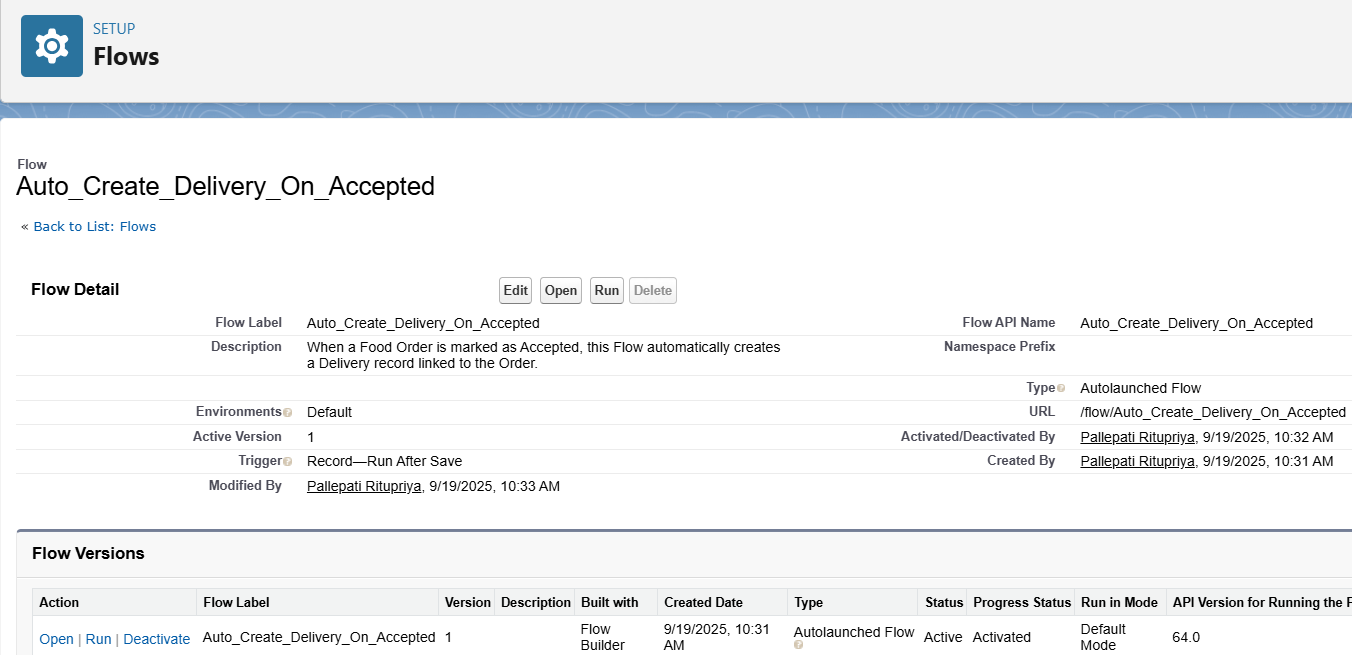


**Flow Builder**

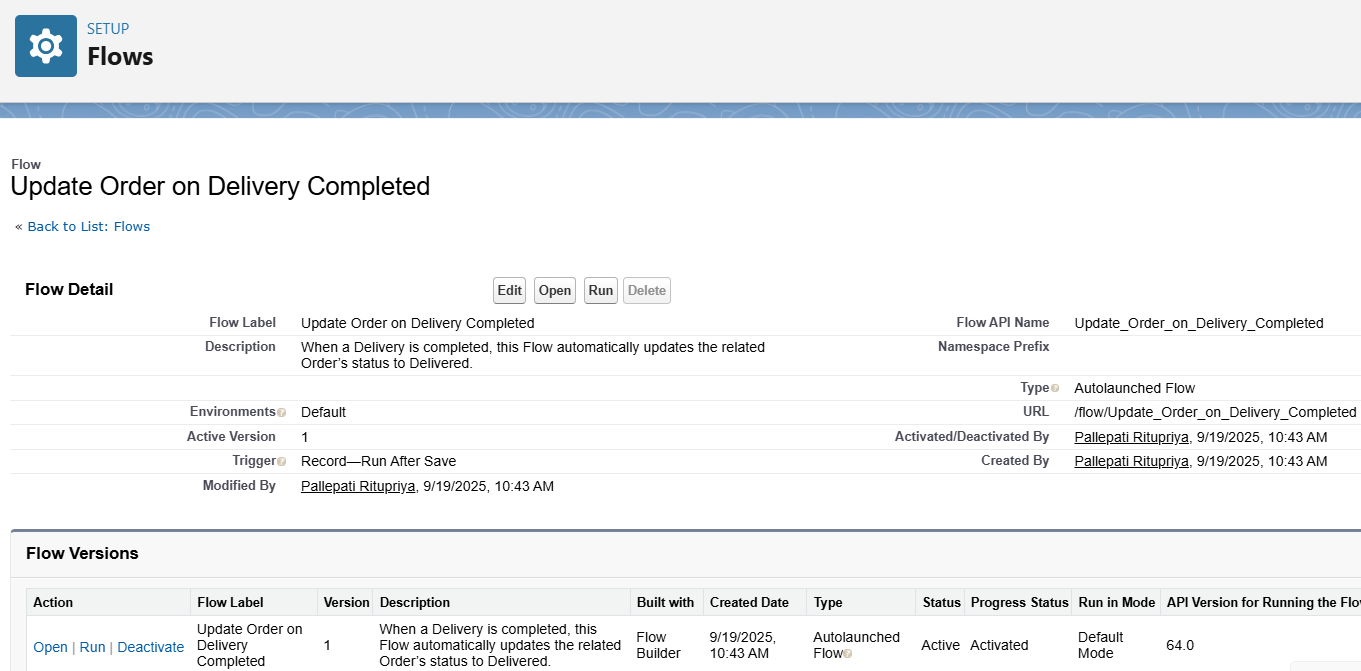
* Setup → Flow Builder → New Flow → Record-Triggered → Auto-launched Flow
* Record-Triggered Flows run automatically when a record is created or updated, enabling automation without user intervention.

**Flows in Food Delivery CRM:**

1. **Auto\_Create\_Delivery\_On\_Accepted**  
   • Trigger: Record—Run After Save  
   • Description: When a Food Order is marked as Accepted, this flow automatically creates a Delivery record linked to the order.



1. **Update\_Order\_on\_Delivery\_Completed**  
   • Trigger: Record—Run After Save  
   • Description: When a Delivery is completed, this flow automatically updates the related Food Order status to Delivered.

****

**Purpose / Outcome:**

* Automates business processes to reduce manual work.
* Ensures data consistency and real-time updates across related objects.
* Supports accurate tracking of orders and deliveries.

......................................................................................................................................................

**Phase 5: Apex Programming (Developer)**