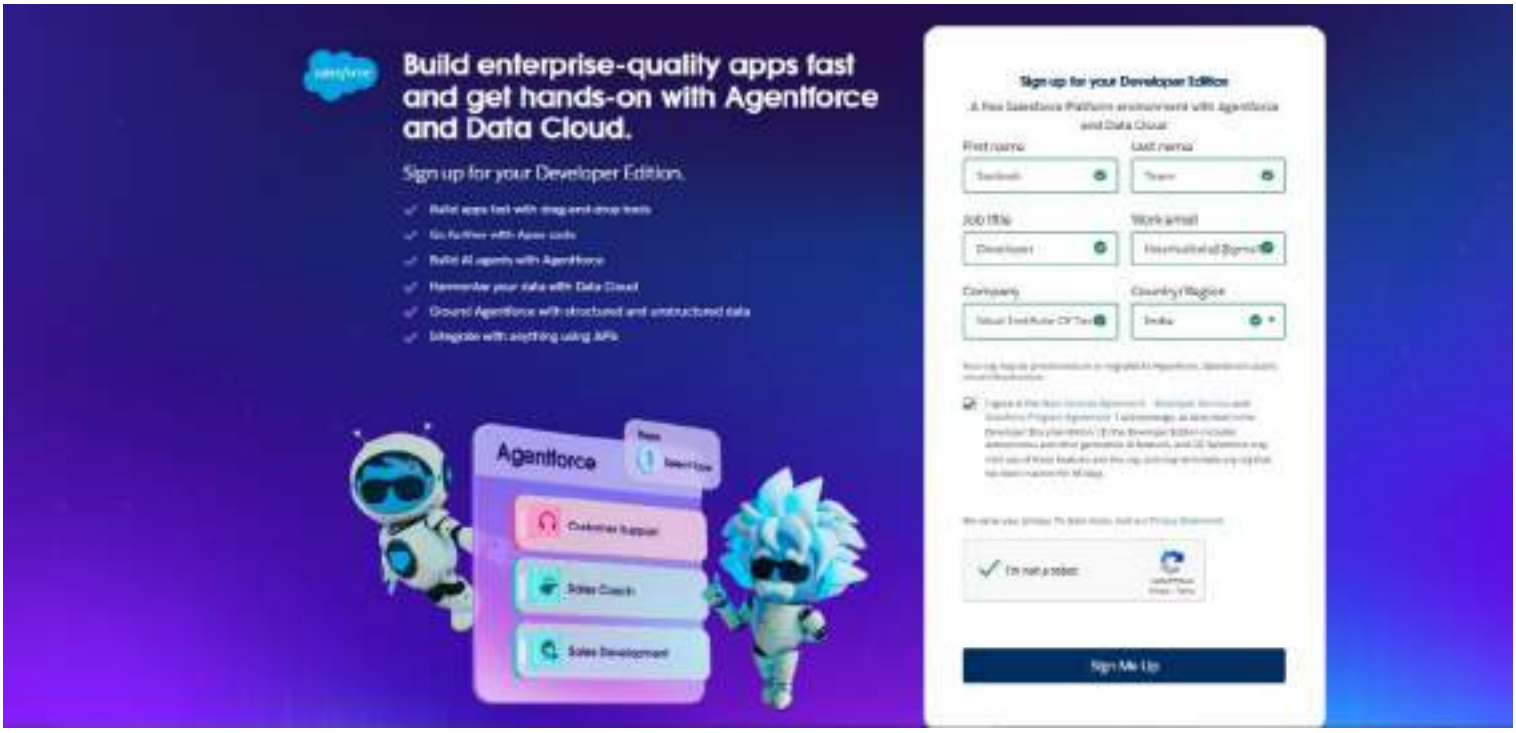
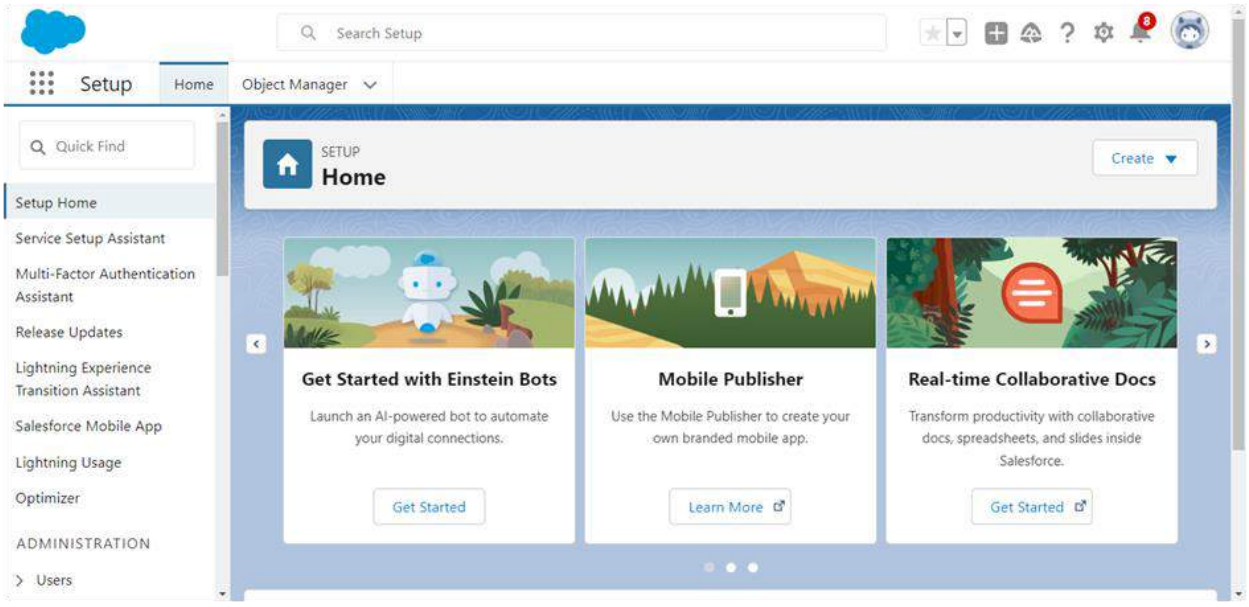
PROJECT DEVELOPMENT PHASE

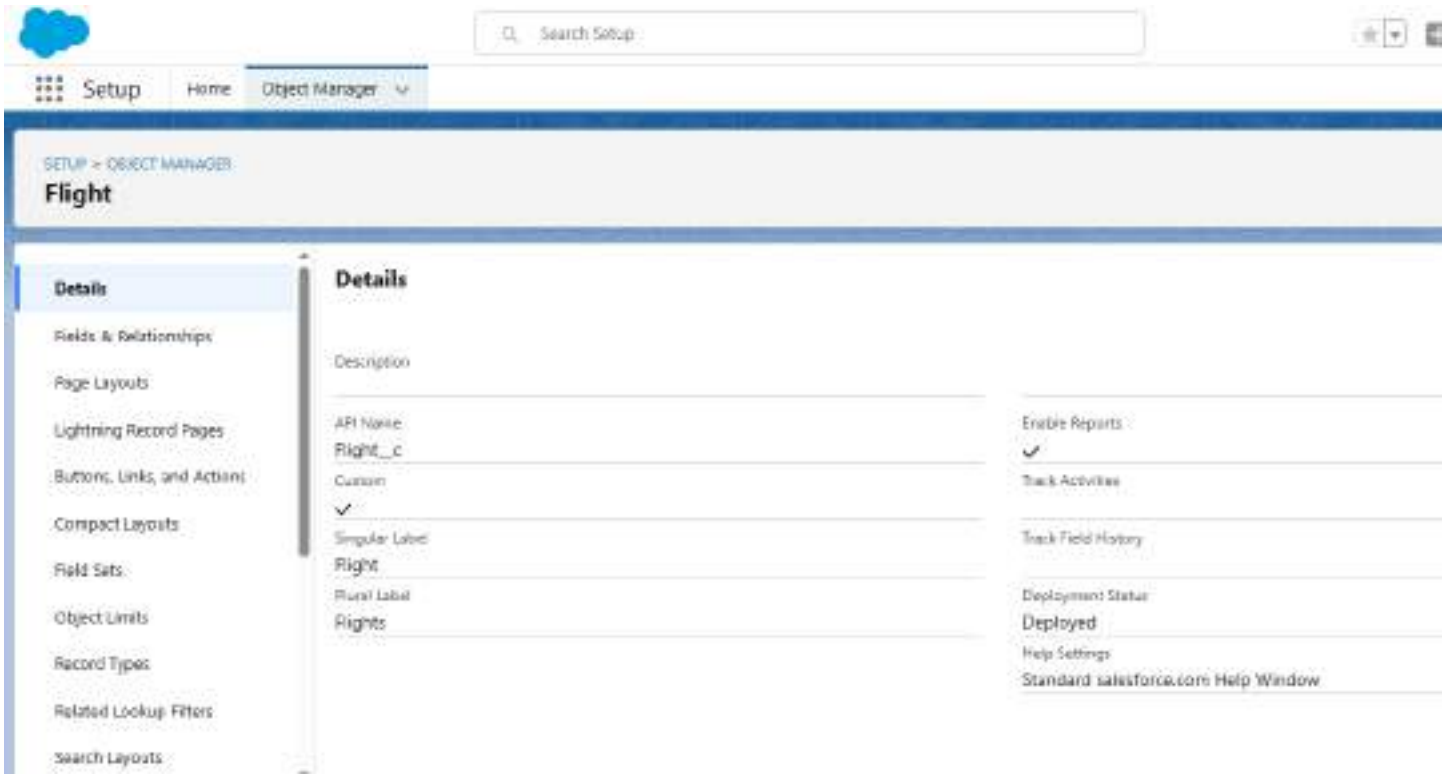
Date	**25-06-2025**
Team Id	**LTVIP2025TMID29760**
Project Name	**AirLine Management System**
College Name	**Ideal Institute Of Technology**

- Created developer org and explored platform features

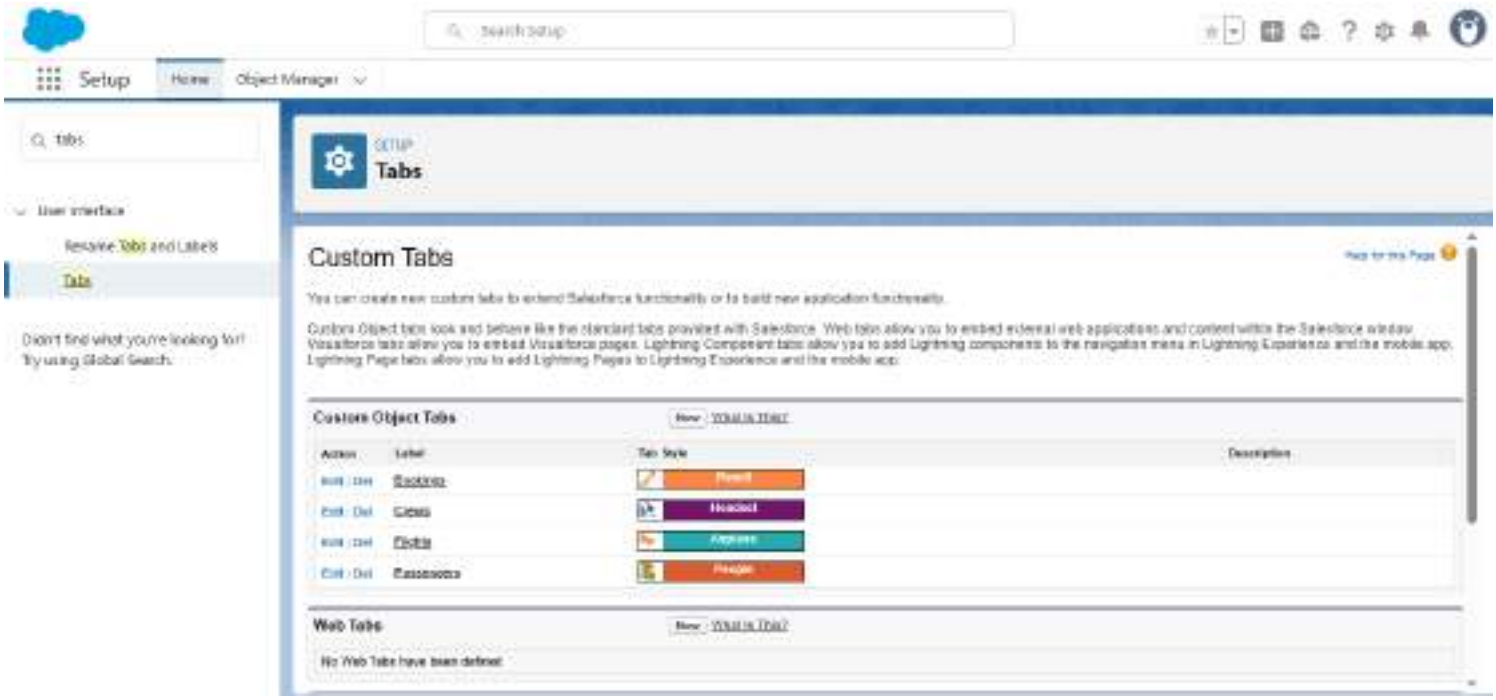




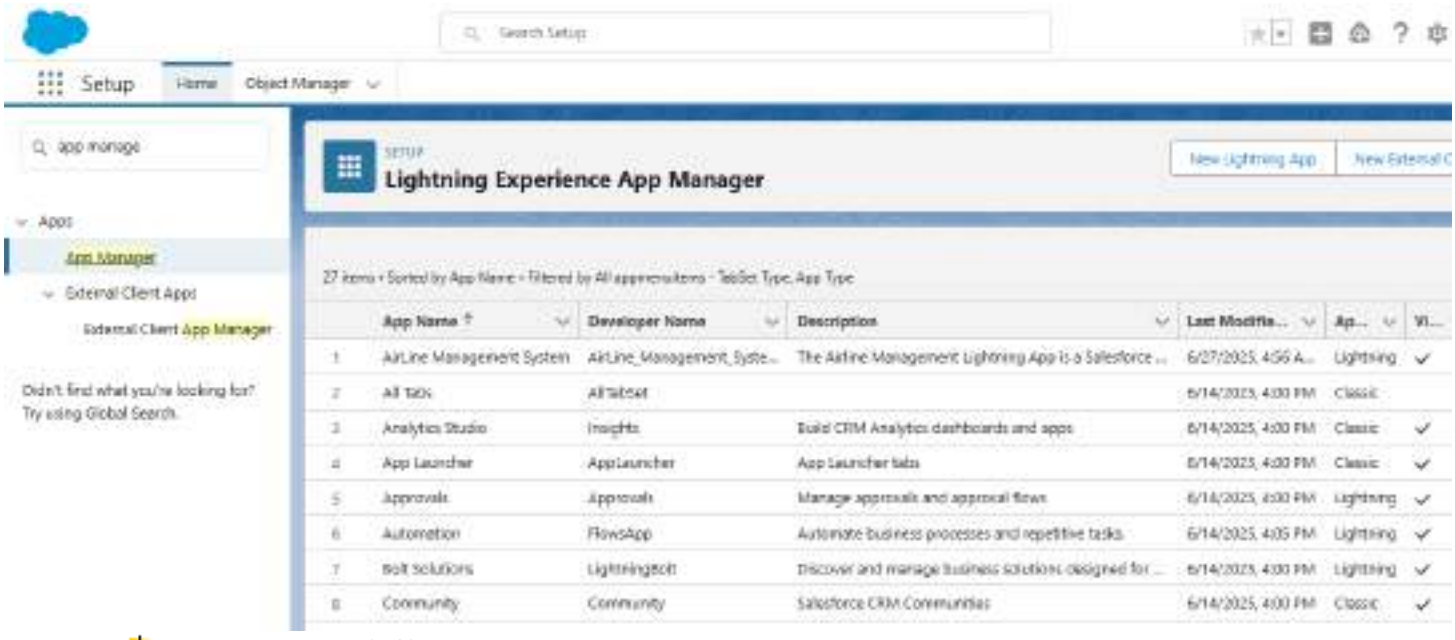
- Object and tab creation



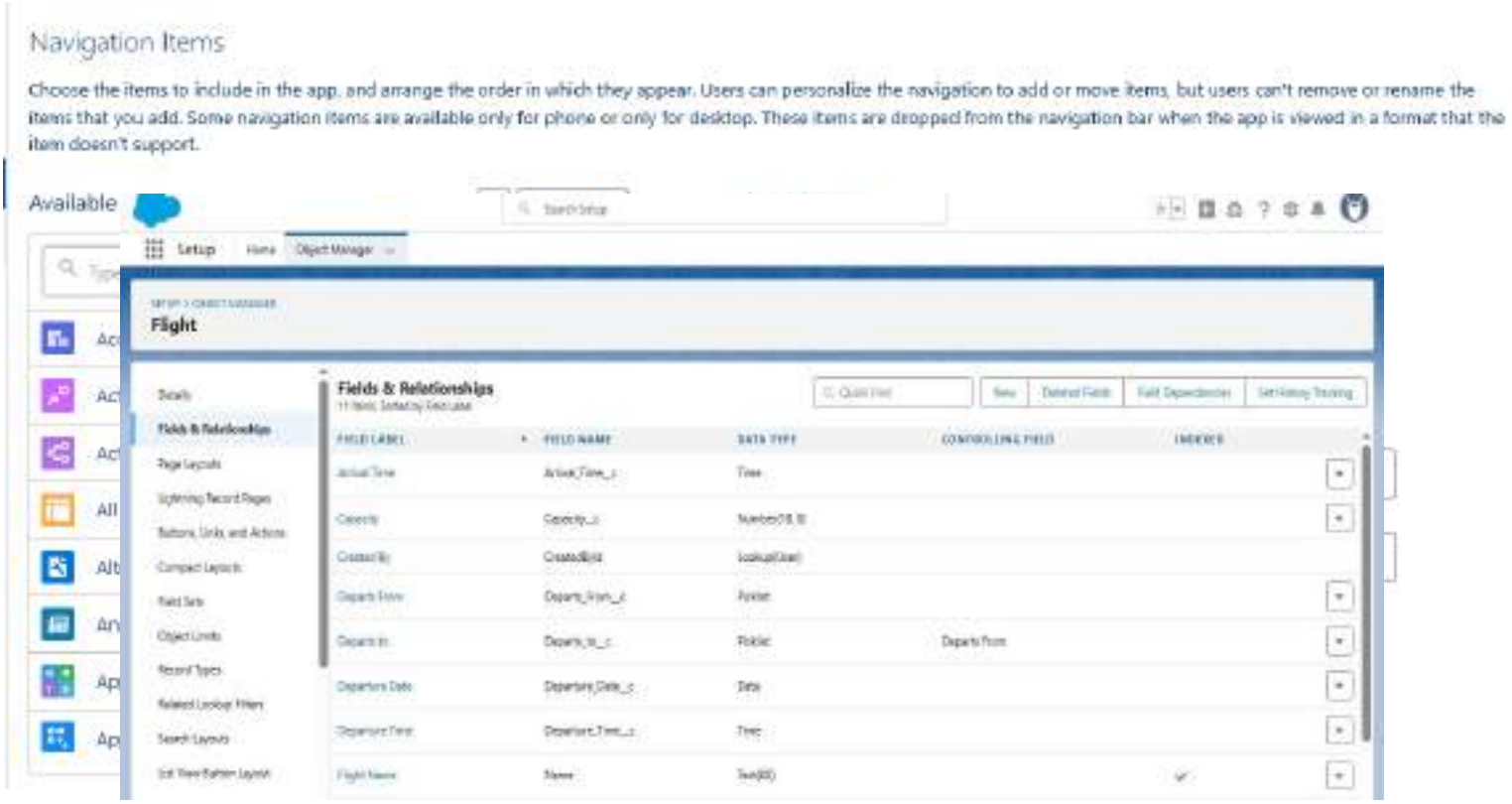
- Designed core objects: Passenger, Booking, Flight, Crew



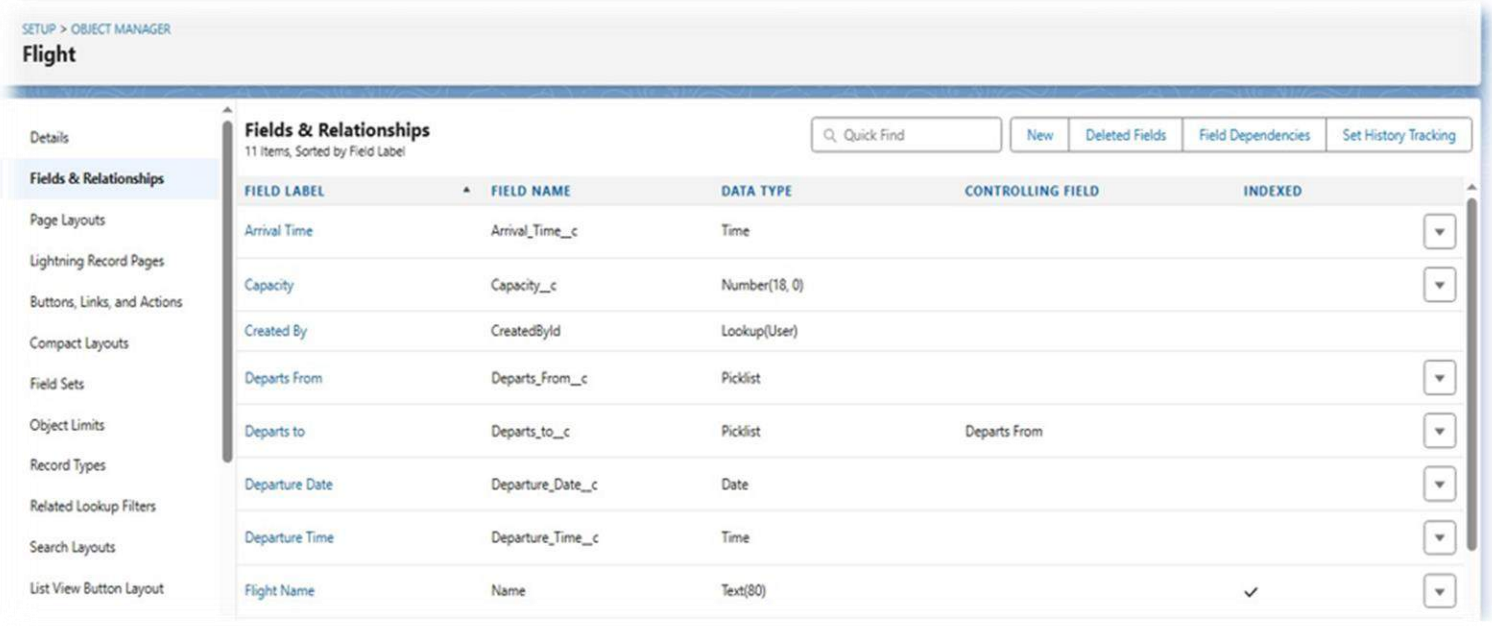
- Lightning App Development



- Created “Airline Management System” app with navigation tabs



- Added custom fields, picklists, lookup relationships & field dependencies

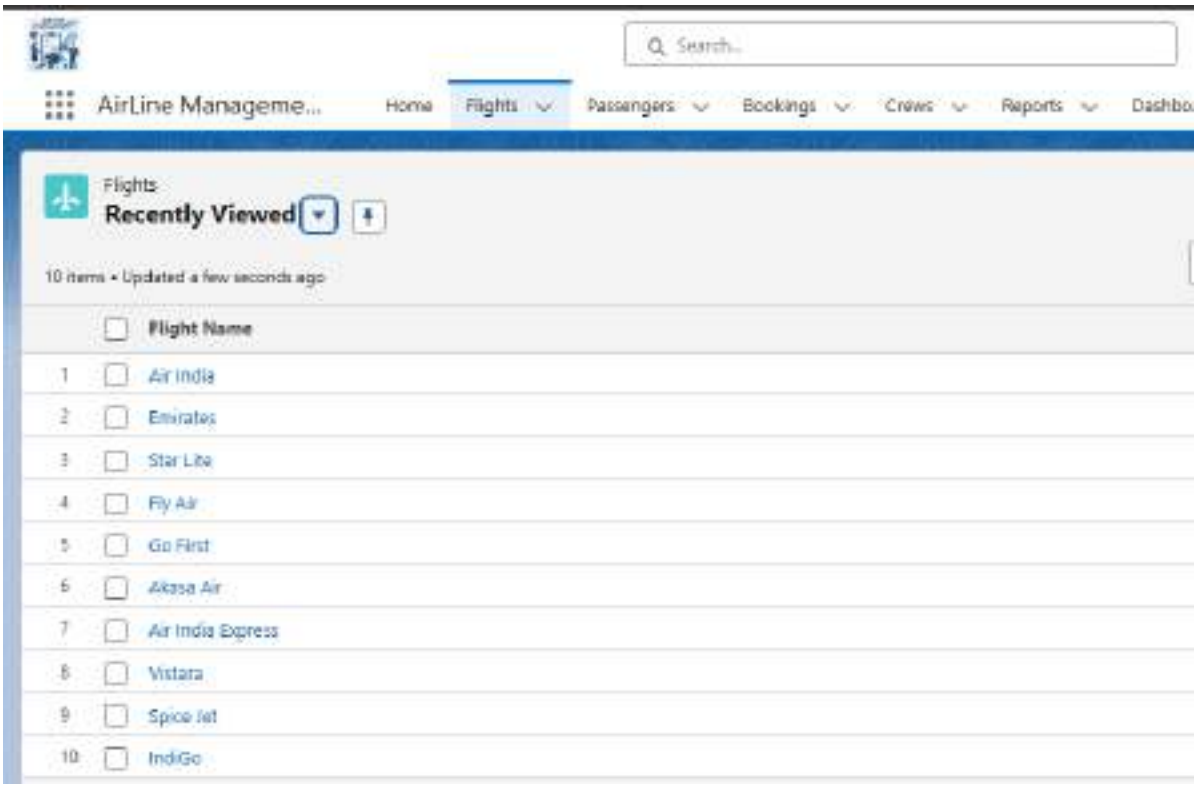


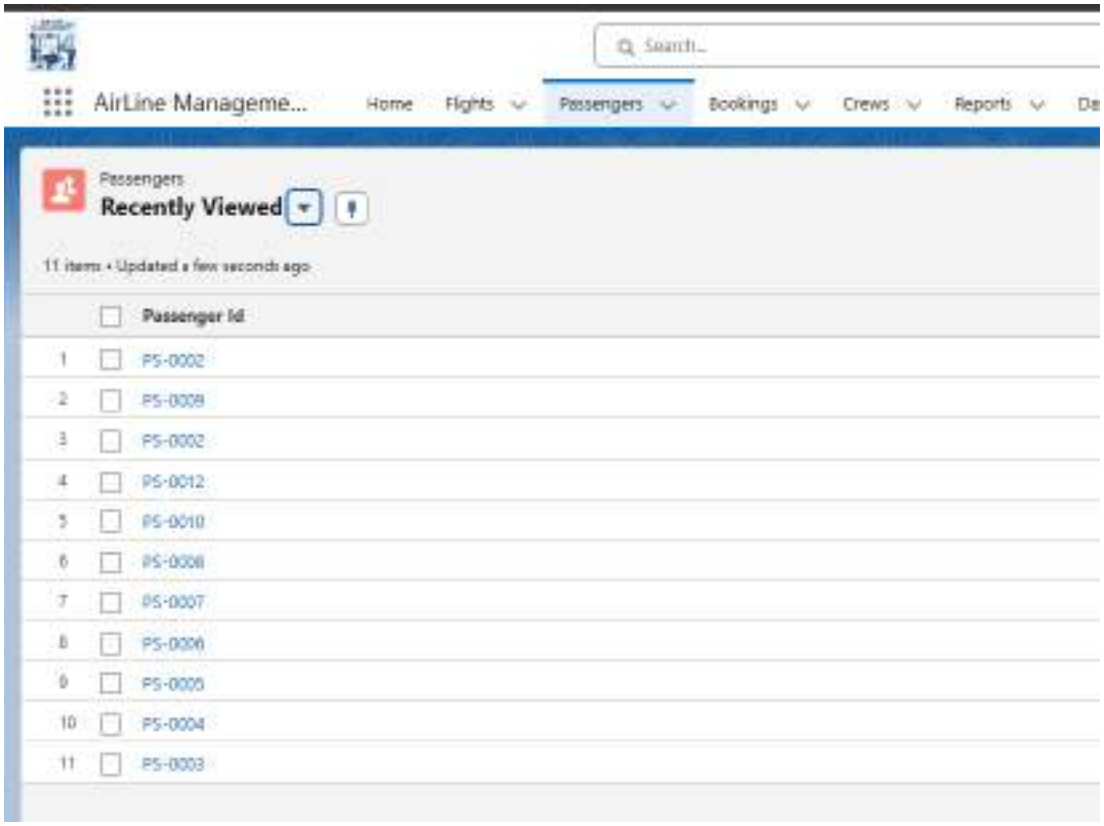
- Schema Builder

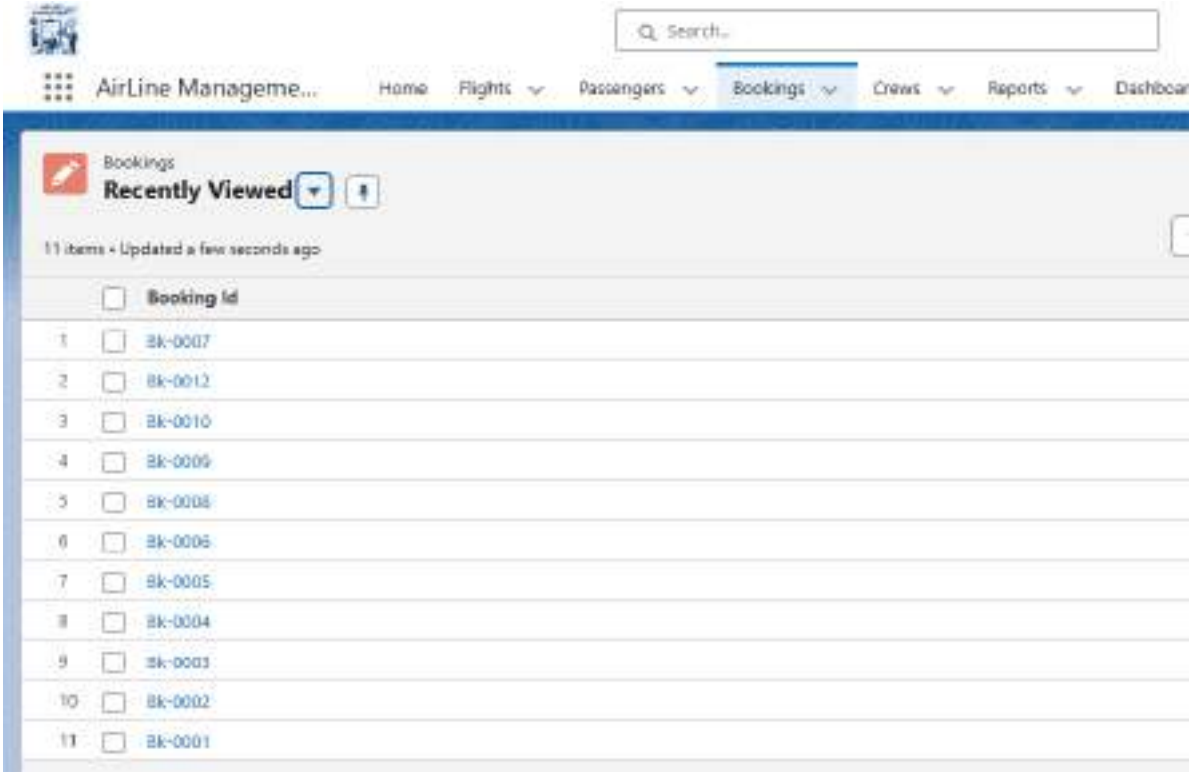
Visualized object relationships using Salesforce Schema Builder

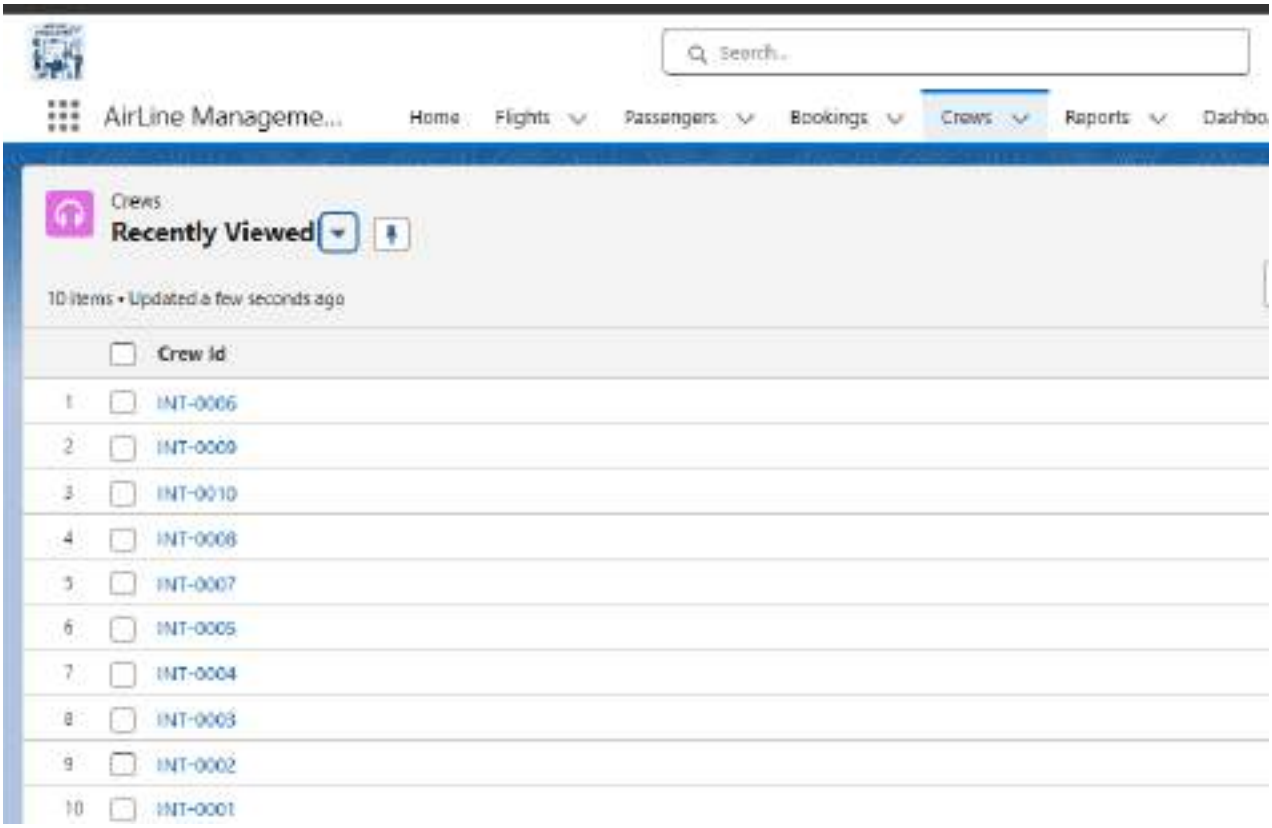
- Data Creation & Record Entry

Entered 10 records per object via Lightning interface

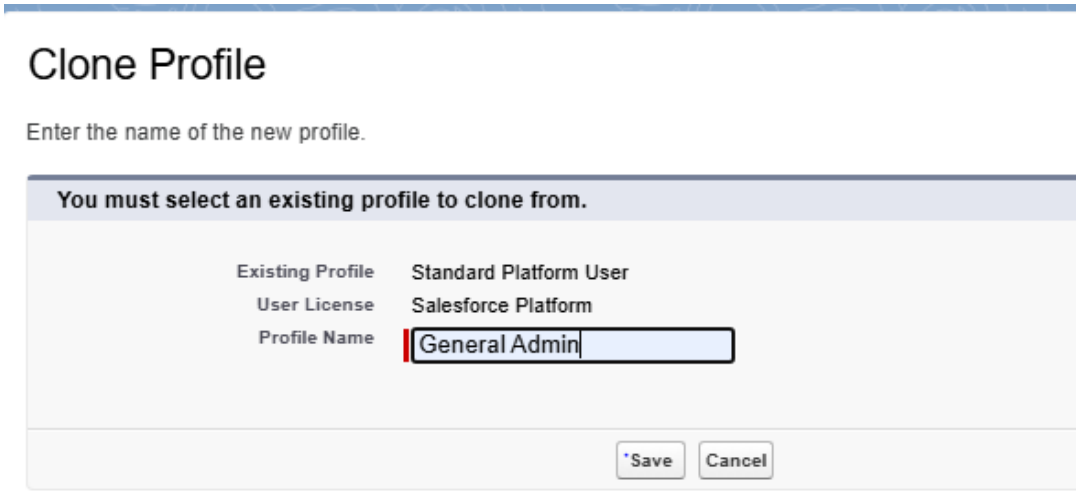


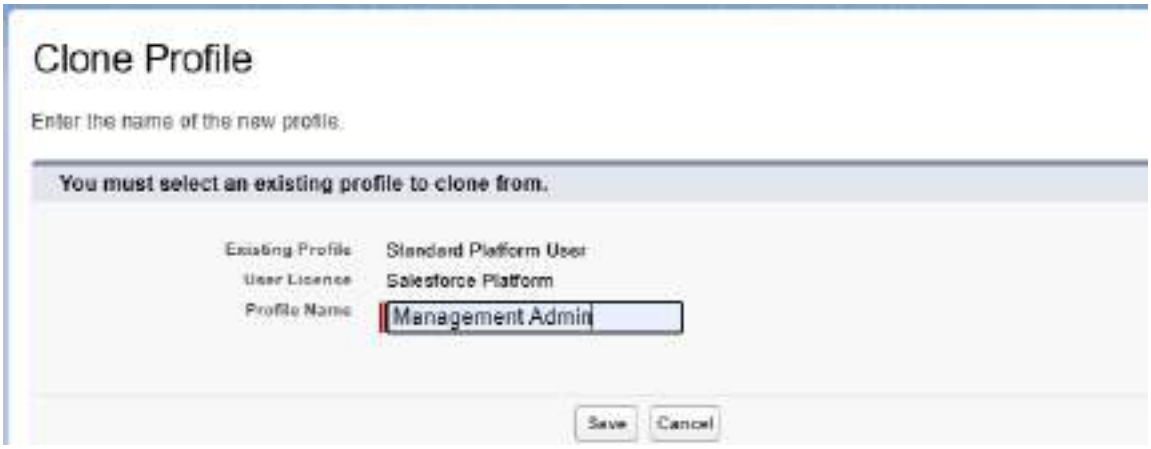

Search...
AirLine Manageme...
Home
Flights
Passengers
Bookings
Crews
Reports
Passengers
Recently Viewed
11 items • Updated a few seconds ago
Passenger Id
1 PS-0002
2 PS-0009
3 PS-0002
4 PS-0012
5 PS-0010
6 PS-0008
7 PS-0007
8 PS-0006
9 PS-0005
10 PS-0004
11 PS-0003

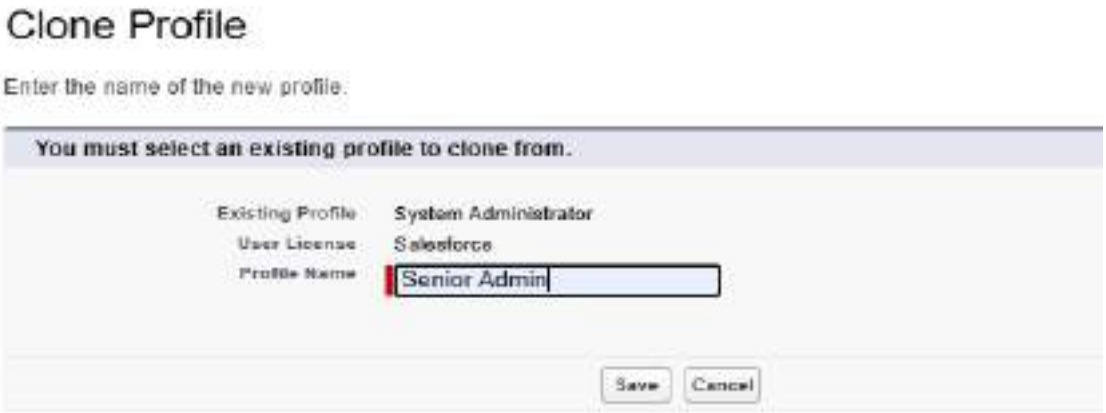

Search...
AirLine Manageme...
Home
Flights
Passengers
Bookings
Crews
Reports
Bookings
Recently Viewed
11 items • Updated a few seconds ago
Booking Id
1 Bk-0007
2 Bk-0012
3 Bk-0010
4 Bk-0009
5 Bk-0008
6 Bk-0006
7 Bk-0005
8 Bk-0004
9 Bk-0003
10 Bk-0002
11 Bk-0001


Search...
AirLine Manageme...
Home
Flights
Passengers
Bookings
Crews
Reports
Dashbo
Crews
Recently Viewed
10 items • Updated a few seconds ago
Crew Id
1 INT-0006
2 INT-0009
3 INT-0010
4 INT-0008
5 INT-0007
6 INT-0005
7 INT-0004
8 INT-0003
9 INT-0002
10 INT-0001

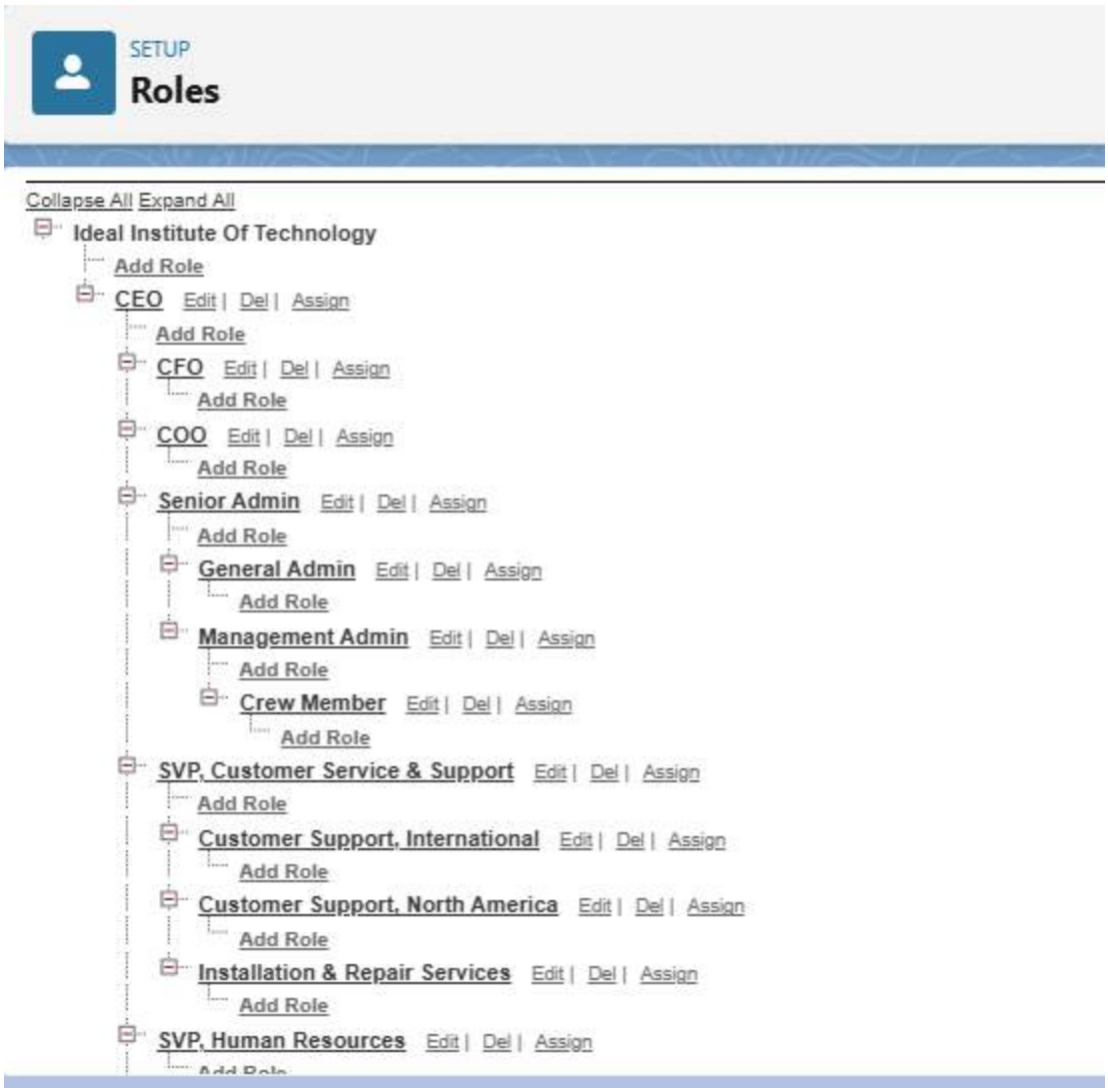
Profile Management

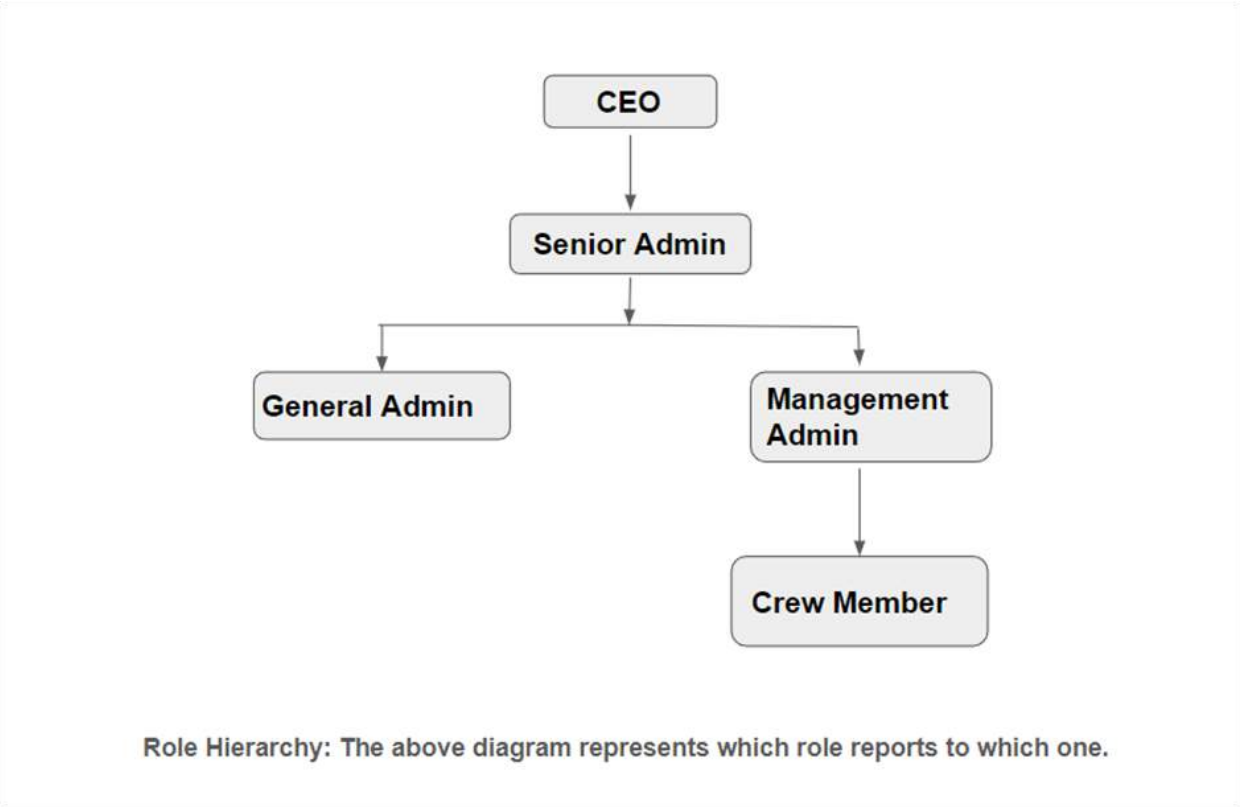






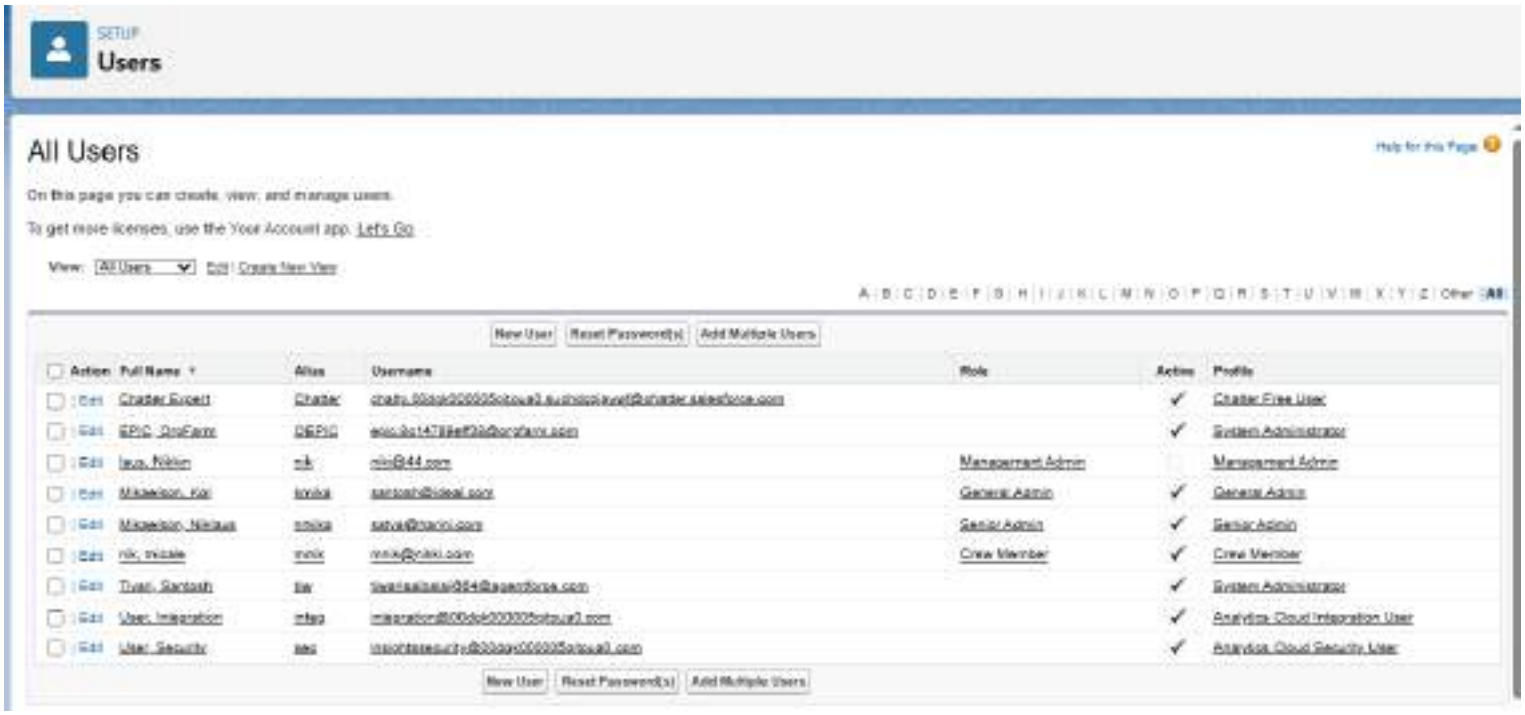
- Role Hierarchy Setup



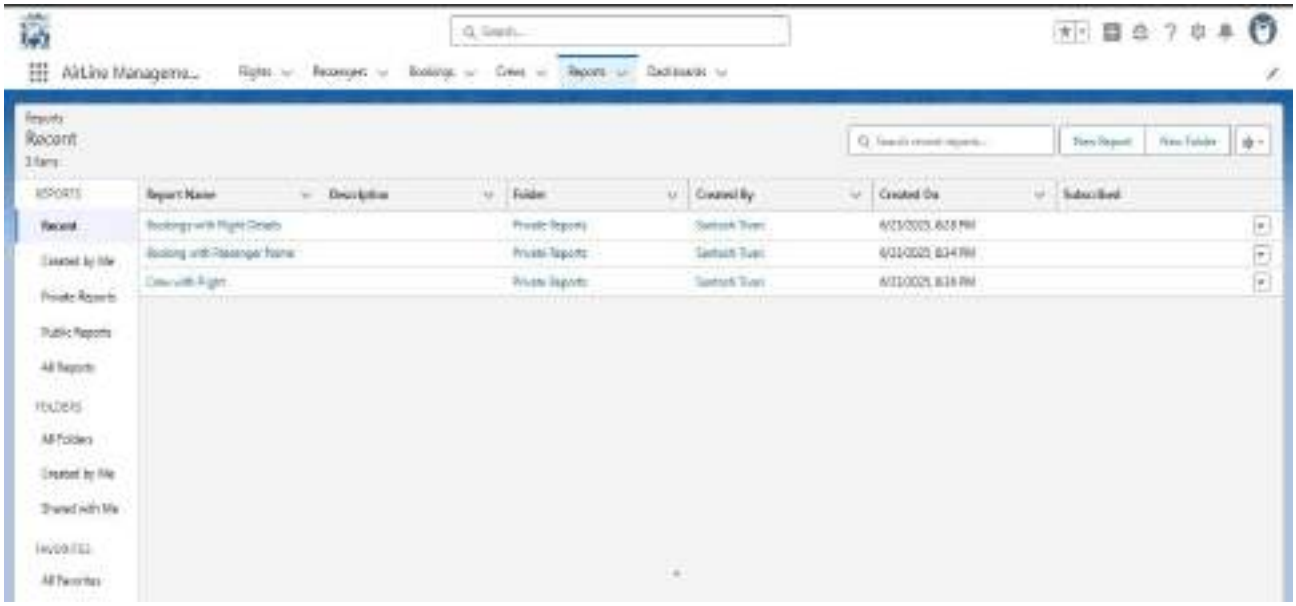


Role Hierarchy: The above diagram represents which role reports to which one.

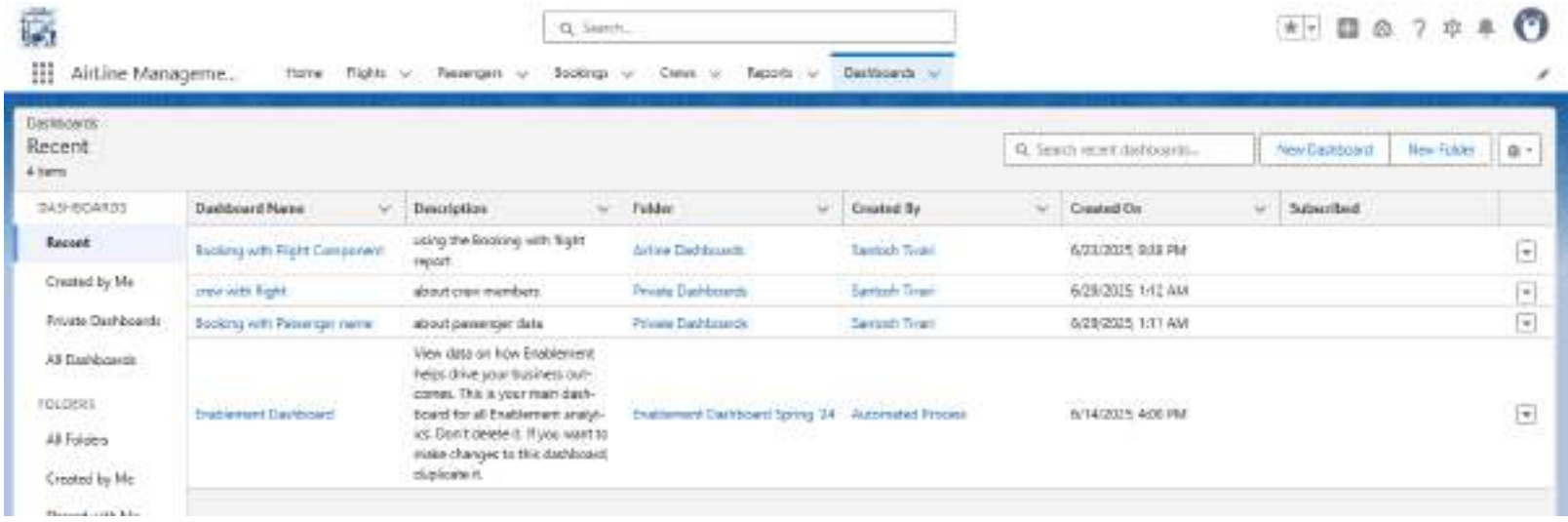
- User Management



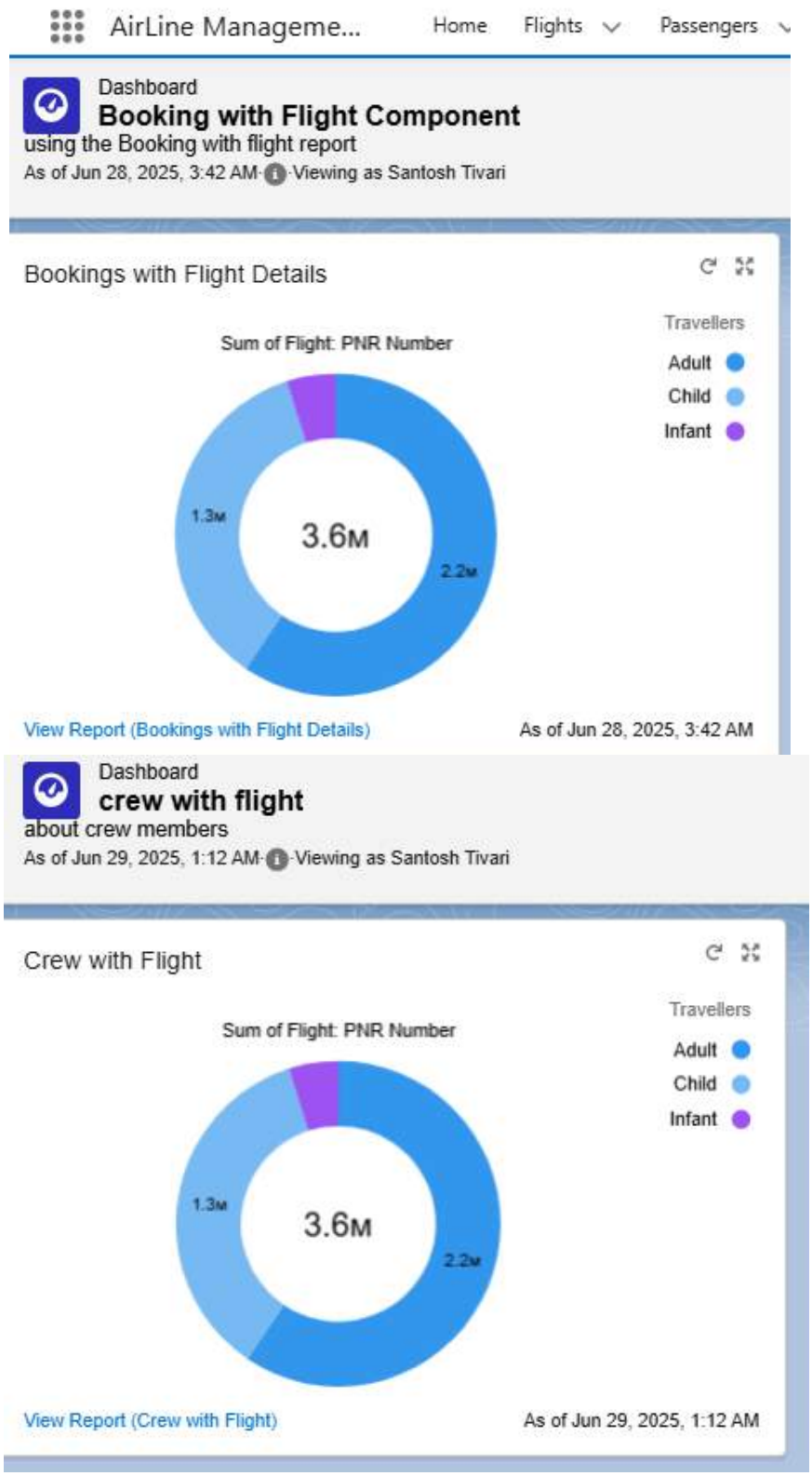
- Reports



- Dashboards



Dashboards display as below:



- Apex Development

Wrote class ,trigger and test class to validate passenger phone input

File ▾ Edit ▾ Debug ▾ Test ▾ Workspace ▾ Help ▾ < >

PhnValid_PassengerObj.apxc

Code Coverage: None ▾ API Version: 58 ▾

```
1 public class PhnValid_PassengerObj {
2     public static void valMethod(List<Passenger__c> newPass){
3         for(Passenger__c p : newPass){
4             if(p.Phone__c == null){
5             p.Phone__c.addError('Please Enter Phone Number');
6             }
7         }
8     }
9 }
```

PassportExpiryValidator.apxc

Code Coverage: None API Version: 64

```apex
public class PassportExpiryValidator {
    public static void validate(List<Passenger__c> passengers) {
        for (Passenger__c p : passengers) {
            if (p.Passport_Expiry_Date__c != null && p.Passport_Expiry_Date__c < Date.today()) {
                p.Passport_Expiry_Date__c.addError('Passport expiry date cannot be in the past.');
            }
        }
    }
}
```

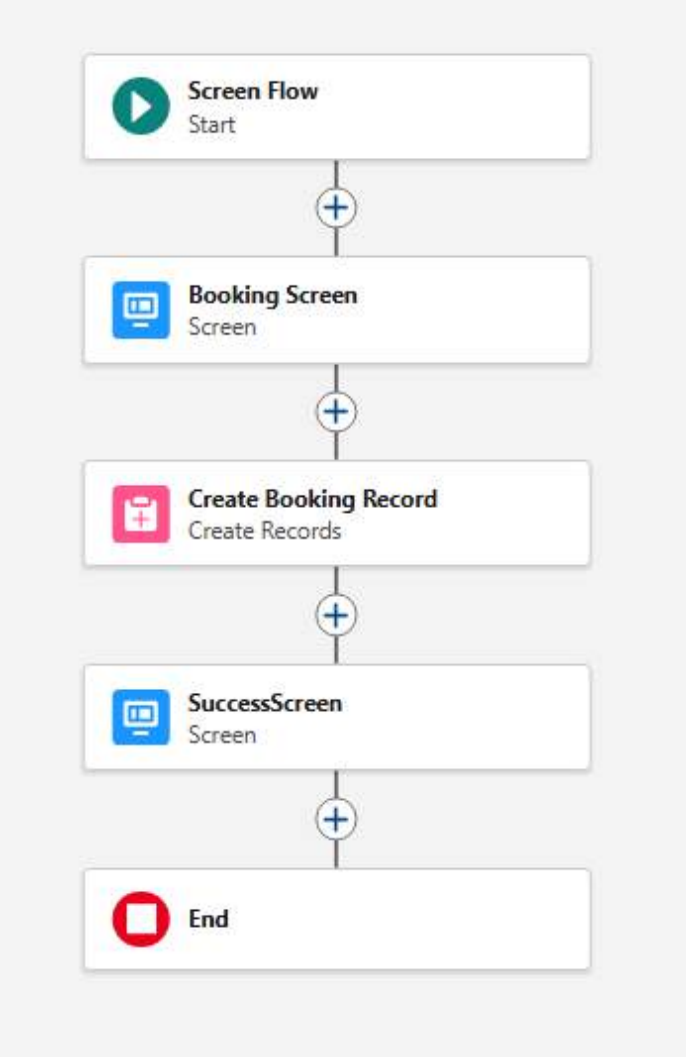
PhnValid_TestClass.apxc

Code Coverage: None API Version: 64

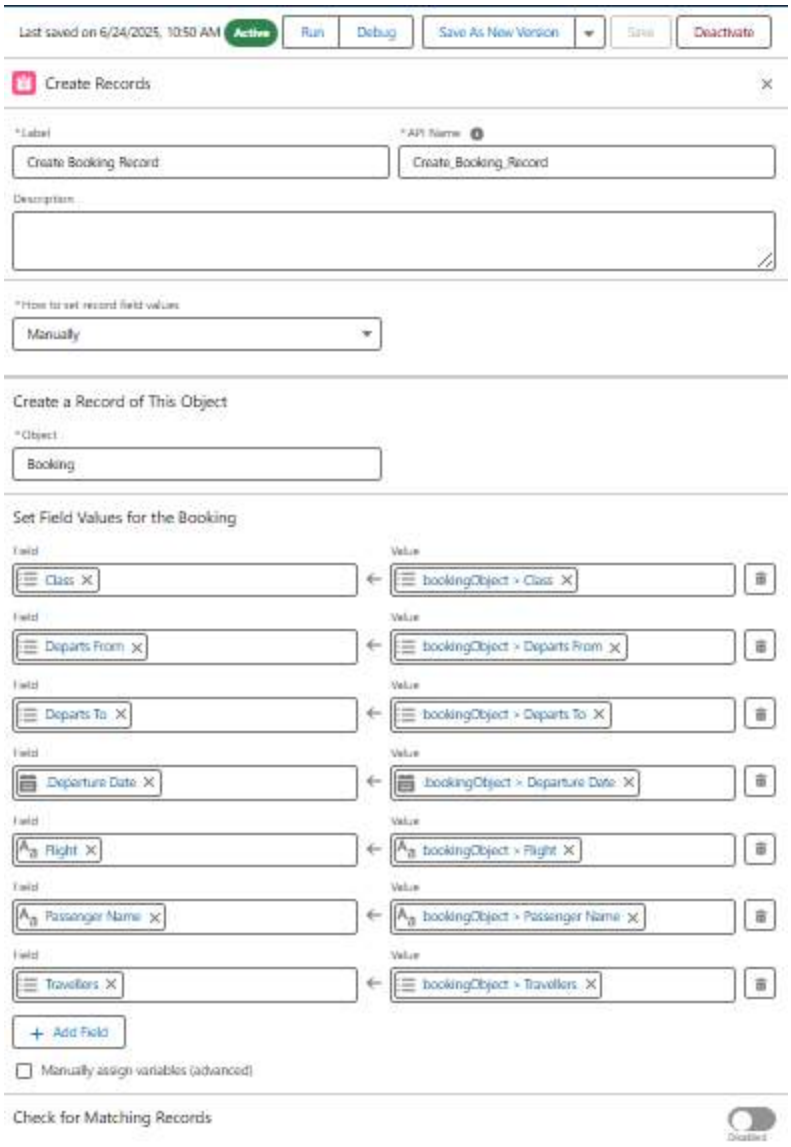
```apex
@isTest
public class PhnValid_TestClass {
    @isTest
    public static void testClass(){
        list <Passenger__c> varlis = new list<Passenger__c>();
        Passenger__c var = new Passenger__c();
        var.Phone__c = null;
        varlis.add(var);
        insert varlis;
        PhnValid_PassengerObj.valMethod(varlis);
    }
}
```

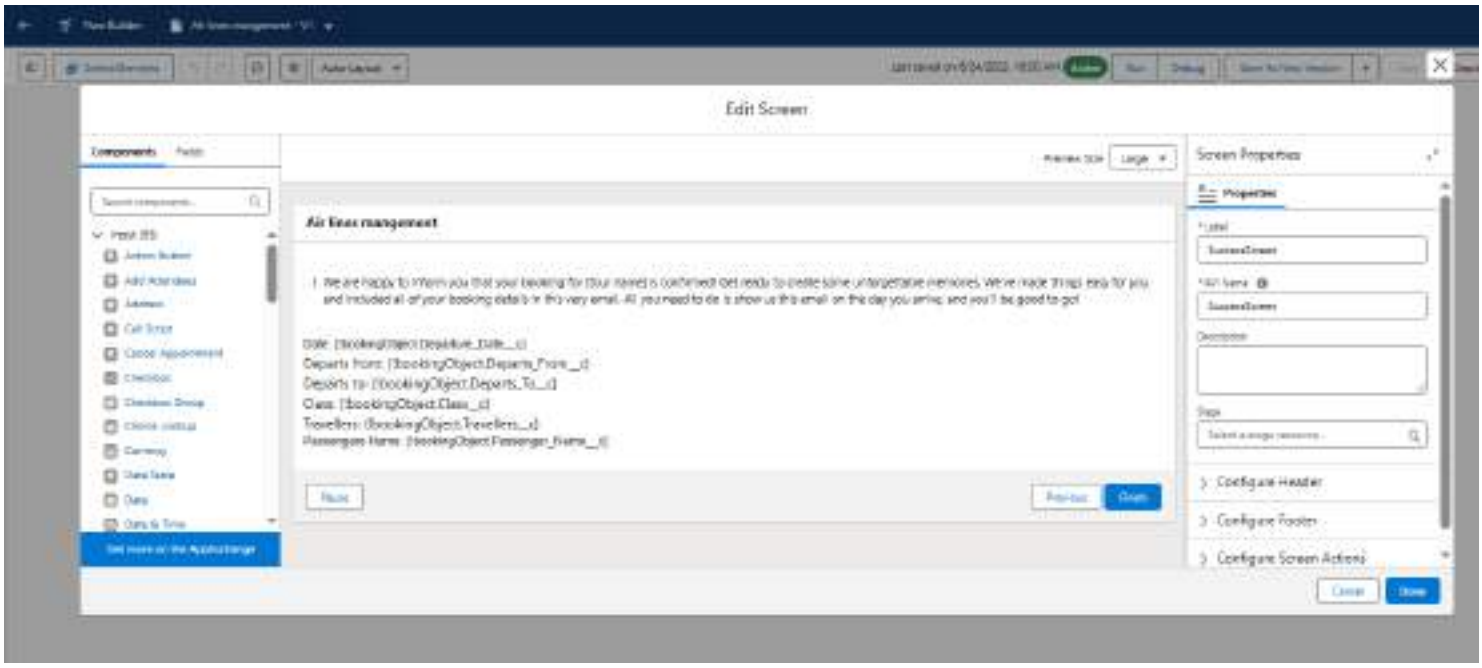
- Flow Automation

Built booking flow with success message confirmation screen



- **Built booking flow with success message confirmation screen**


Last saved on 6/24/2025, 10:50 AM
Active
Run
Debug
Save As New Version
Save
Deactivate
Create Records
*Label
Create Booking Record
*API Name
Create_Booking_Record
Description
*How to set record field values
Manually
Create a Record of This Object
*Object
Booking
Set Field Values for the Booking
Field
Value
Class
bookingObject > Class
Departs From
bookingObject > Departs From
Departs To
bookingObject > Departs To
Departure Date
bookingObject > Departure Date
Flight
bookingObject > Flight
Passenger Name
bookingObject > Passenger Name
Travellers
bookingObject > Travellers
Add Field
Manually assign variables (advanced)
Check for Matching Records
Disabled


Edit Screen
Components
Fields
Screen Properties
Properties
Air lines mangement
Configure Header
Configure Footer
Configure Screen Actions
Cancel
Done

✓ Project Executable Files

In the development of the Airlines Management System, Salesforce's low-code platform was enhanced using Apex programming to implement custom validations and business logic. These executable files represent the backend logic that enforces data integrity and automates specific actions within the system.

The following components were developed as part of the executable logic:

Apex Class: Contains reusable methods that enforce custom validation rules on the Passenger object.

Apex Trigger: Automatically calls the validation method before a Passenger record is inserted, ensuring mandatory fields are checked.

Apex Test Class: Validates the logic through unit testing, ensuring that the trigger behaves correctly and achieves over 75% code coverage—a Salesforce deployment requirement.

Below are the Apex Class, Trigger, and Test Class used in the Airline Management System project:

1. **Apex Class:** Contains reusable methods that enforce custom validation rules on the Passenger object.

Apex Class: PhnValid_PassengerObj.cls

```
public class PhnValid_PassengerObj
        public static void valMethod(List<Passenger__c>
newPass)(
        for(Passenger___c p : newPass)(
        if(p.Phone__c == null)(
             p.Phone__c.addError('please Enter phone
Number');
```

2. Apex Trigger: Automatically calls the validation method before a Passenger record is inserted, ensuring mandatory fields are checked.

Apex Trigger: PhnValidTrigger.trigger

```
 tri gger  PhnValidTri gger on  Pas sen ger_pc
(before  insert) {
        if(tri gger.is Before && tri gger.is Insert)(

PhnValid_PassengerObj.valMethod(tri gger.new);
```

3. **Apex Test Class:** Validates the logic through unit testing, ensuring that the trigger behaves correctly and achieves over 75% code coverage—a Salesforce deployment requirement.

Apex Test Class: PhnValid_TestClass.cls

```
@isTest
public class PhnValid_TestClass (
        @isTest
        public static void testClassb(
        List<Passenger__c> varlis = new
List<Passenger  _c>b;
        Passenger  _c var = new Passenger  _cb;
        var.Phone__c = null;
        varlis.add(var);
        insert varlis;
        PhnValid_PassengerObj.valMethod(varlis);
```

✓ Dataset

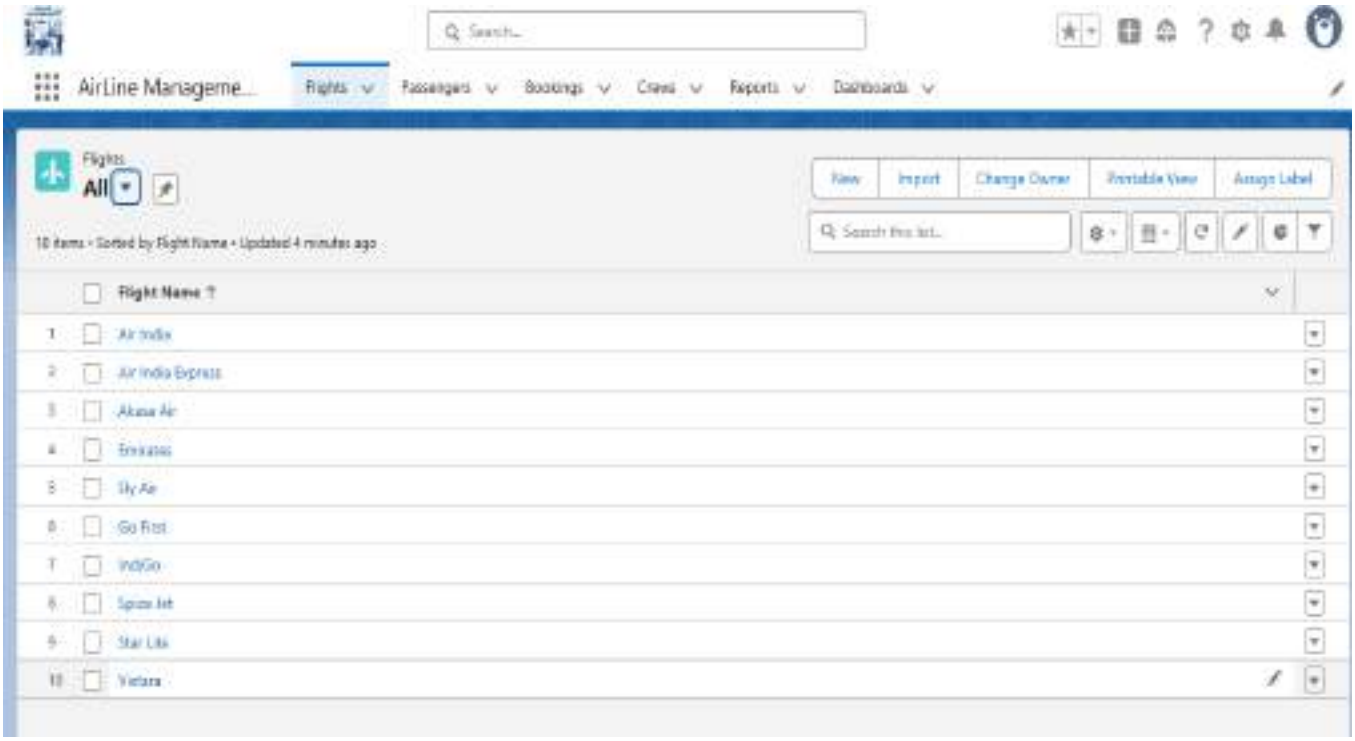
Sample records used for the Airline Management System are shown below. Each object includes a minimum of 10 records

Dataset Creation

As part of the system validation and testing process, datasets were manually created for each key object in the Airlines Management System. A minimum of 10 sample records were added to the following custom objects:

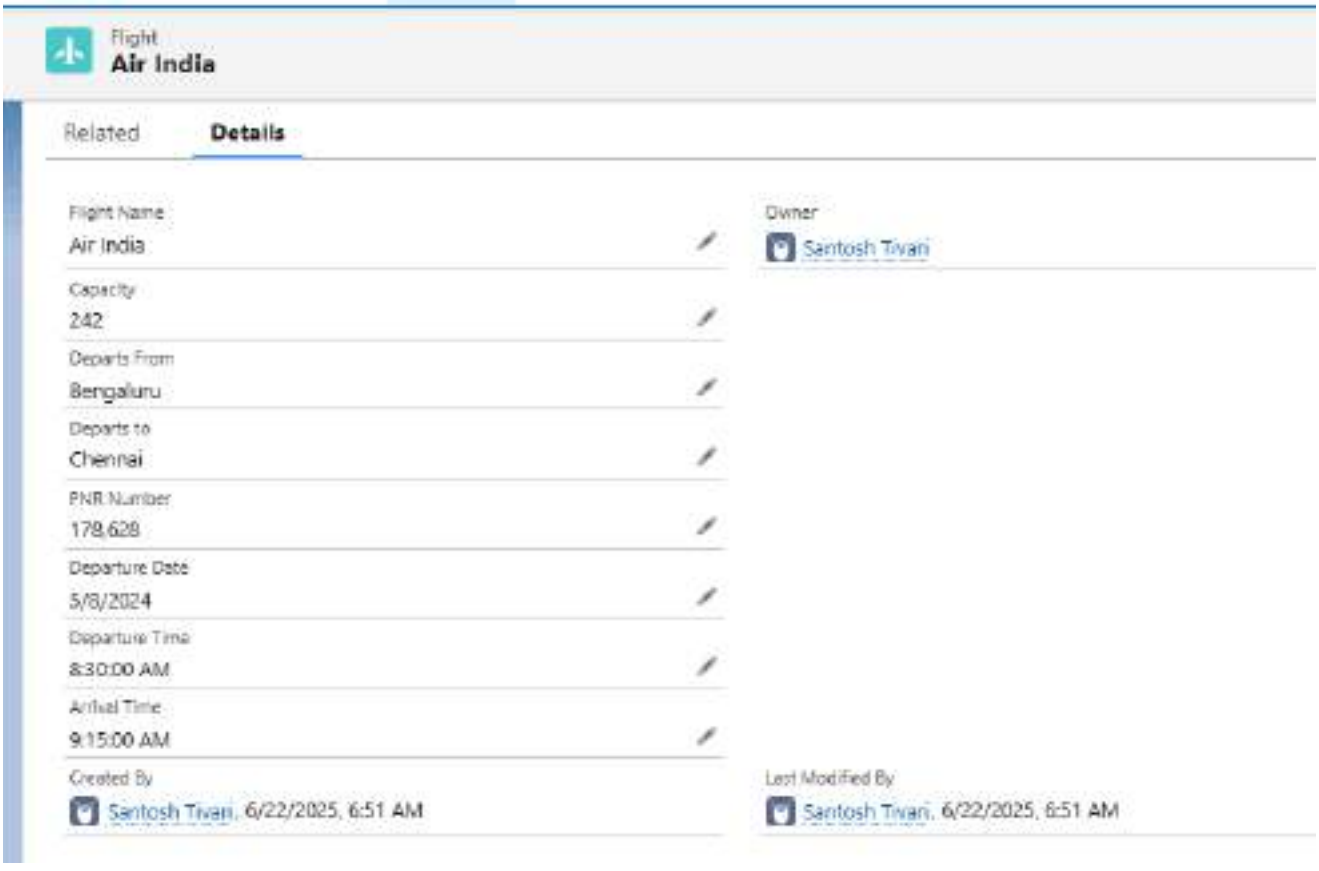
- ✈ Flight
- 🧍 Passenger
- 📄 Booking
- 👮 Crew

✓ FLIGHT

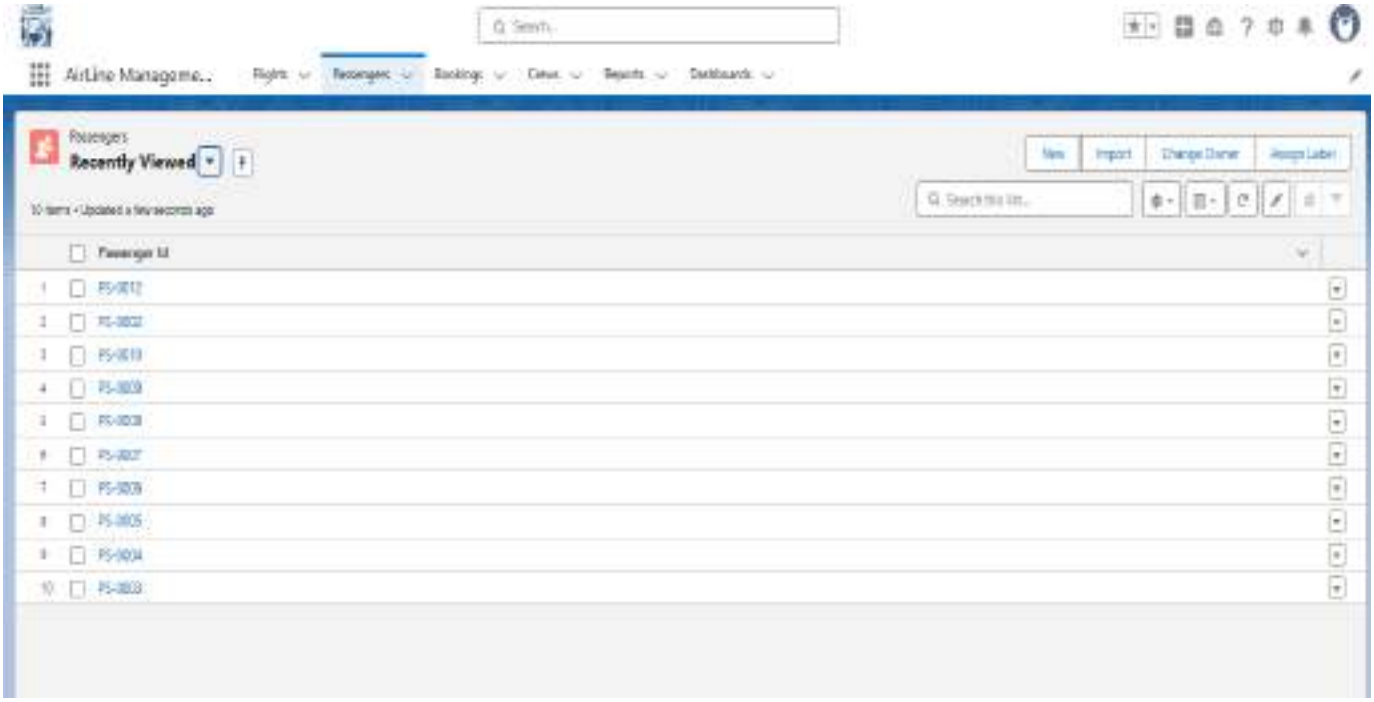


To simulate real-world airline operations, 10 sample records were created for the Flight object within the Salesforce Lightning App. These records include essential details such as:

- Flight Name
- Departs From and Departs To locations
- Capacity
- PNR Number
- Departure & Arrival Times



✓PASSENGER

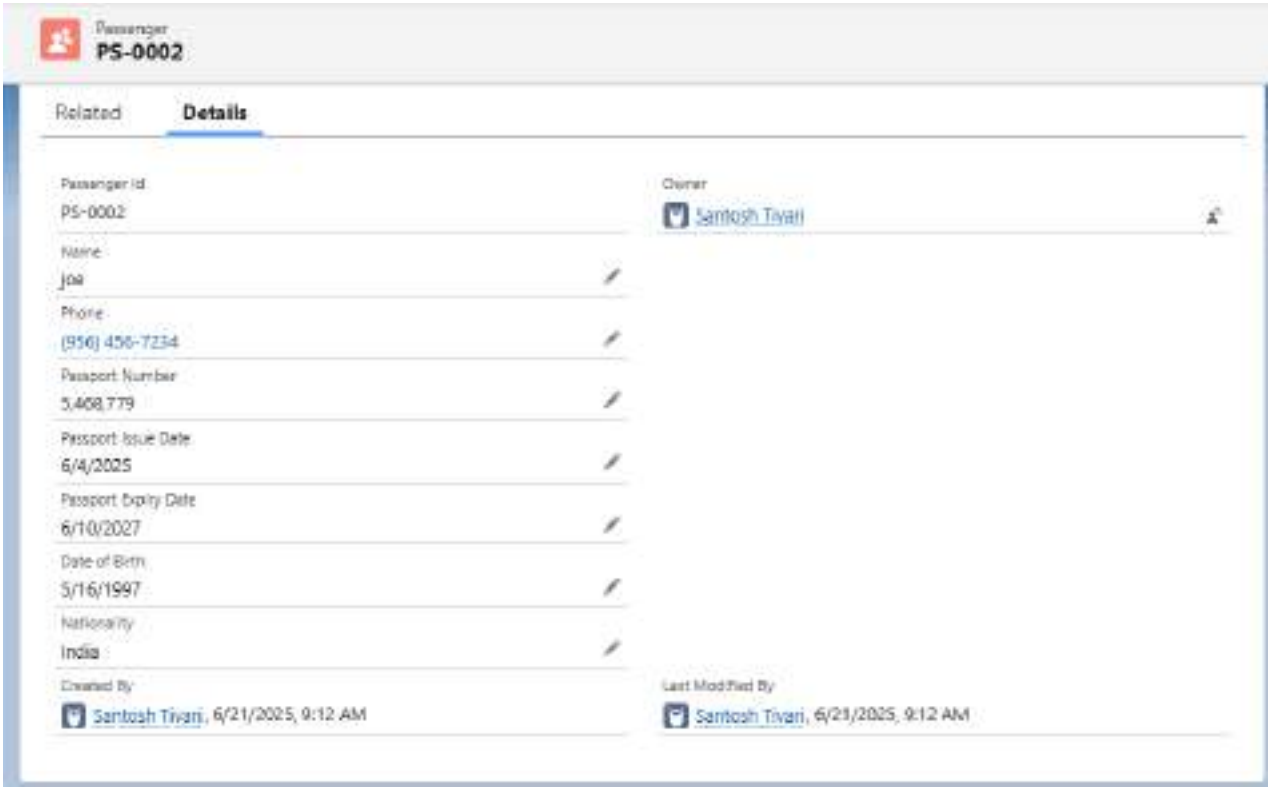


🧍 Passenger Records Creation

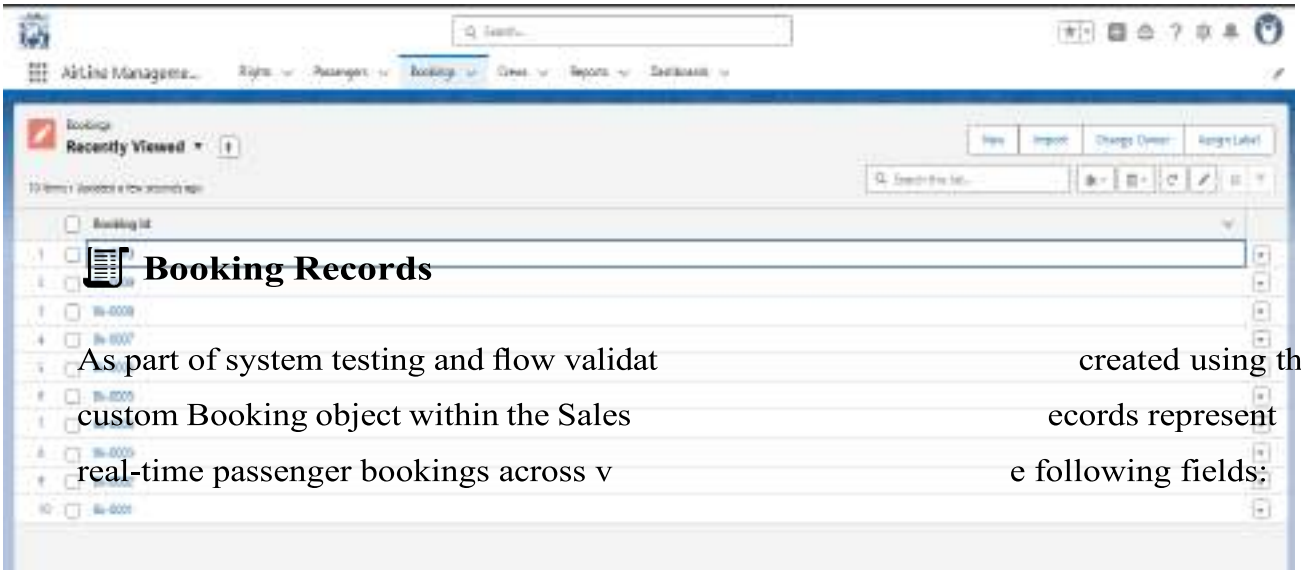
A total of 10 Passenger records were created within the Airlines Manageme represent various travelers booking flights through the platform. Each recor key personal and travel-related details such as:

- Passenger Name
- Phone Number
- Passport Details (Number, Issue Date, Expiry Date)
- Date of Birth
- Nationality (from a predefined picklist)

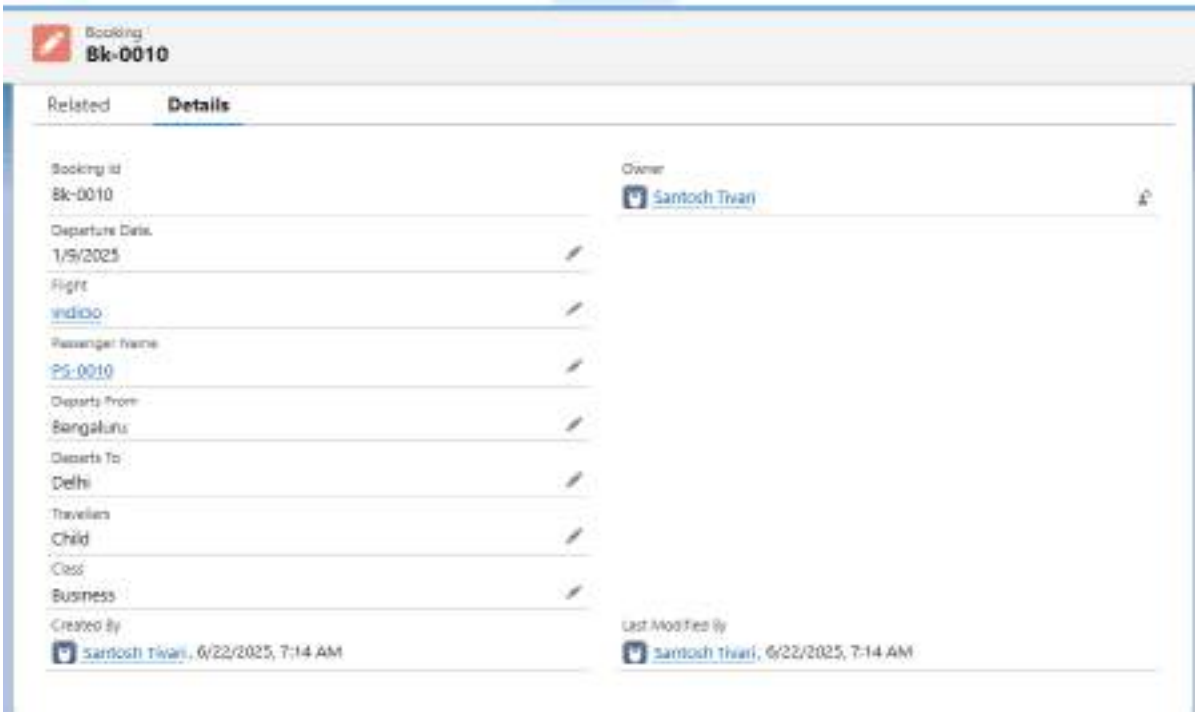
These entries were created using the Passenger tab in the Salesforce Lightning App.



✓BOOKING



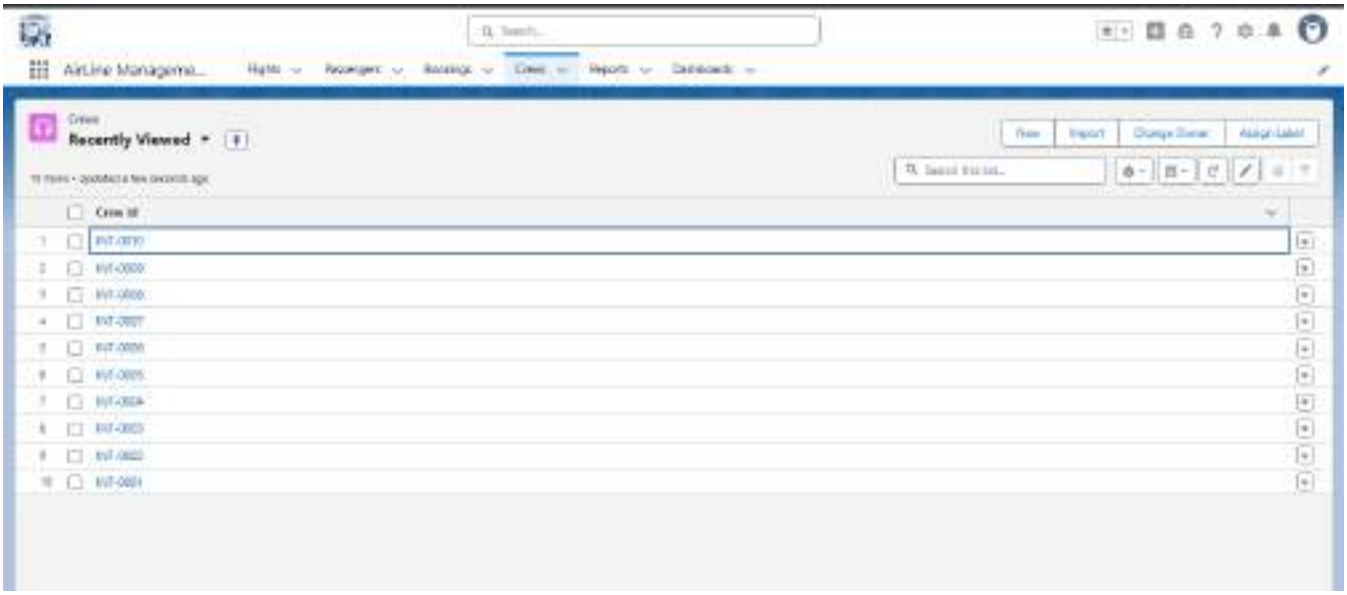
As part of system testing and flow validat created using the custom Booking object within the Sales ecords represent real-time passenger bookings across v e following fields:



- Passenger Name (Lookup to Passenger)
- Flight (Lookup to Flight)
- Departure Date

  Departs From / Departs To
- Class (Economy, Business, First Class)
- Travellers (Adult, Child, Infant)

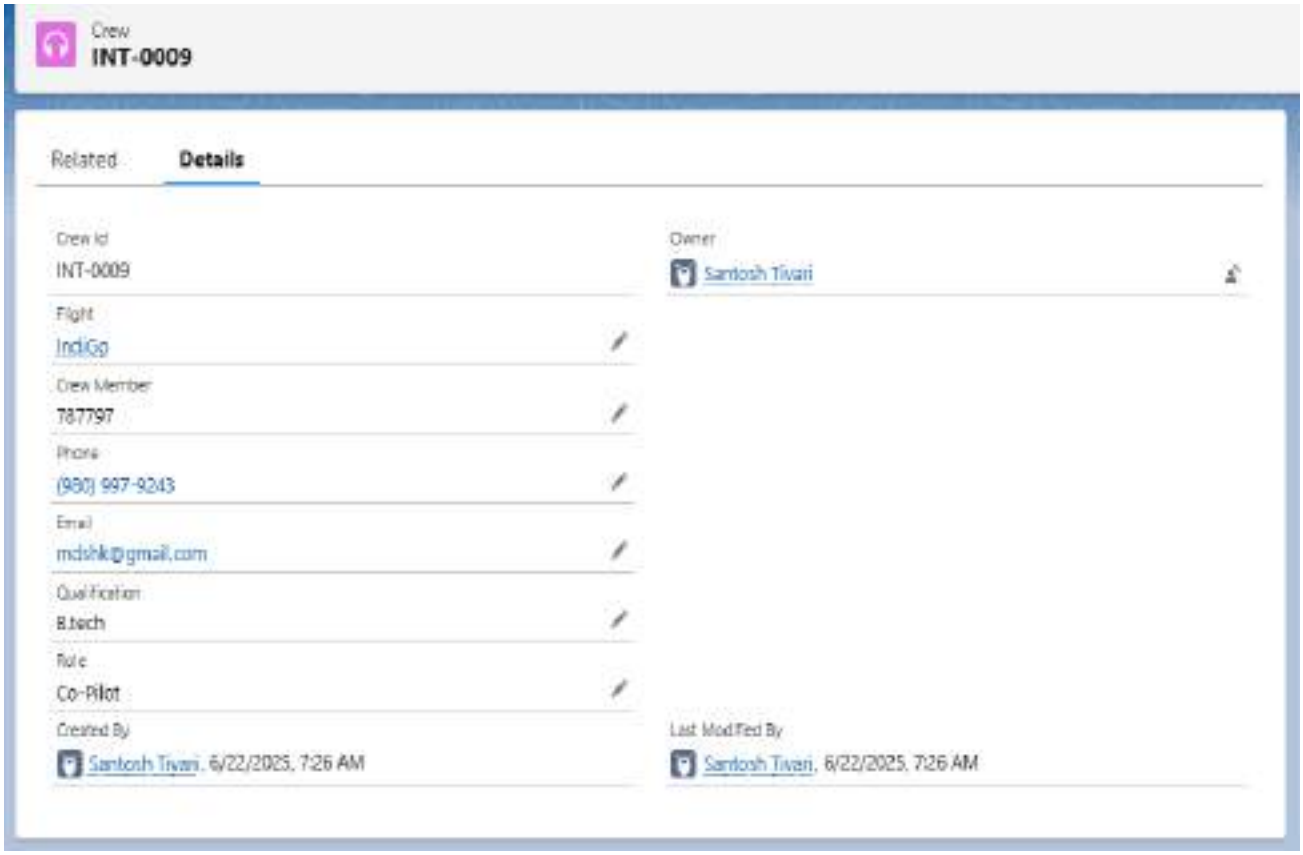
✓CREW



👨‍✈️ Crew Records Creation

To complete the airline operations
the Crew object in the Salesforc
real-life crew assignments for di
fields:

- Crew Member Name
- Phone Number
- Email Address
- Qualification (e.g., Comm
- Role (e.g., Pilot, Co-Pilot,
- Flight (Lookup to Flight o



Overview

It includes Apex classes, triggers, and test classes used to implement core functionalities such as data validation. A sample dataset for major objects like Passenger, Flight, Booking, and Crew is provided to simulate real-world airline operations. The document also includes output screenshots from the Salesforce Org that demonstrate successful execution, error handling, dashboard analytics, and user interface features such as the Lightning App and report views. This project emphasizes the integration of business logic and visual reporting within the Salesforce platform for a streamlined airline management solution.