Project Design Phase

The Project Design Phase defines the logical, technical, and functional foundation of the solution. It ensures that your proposed Salesforce CRM not only solves the right problems but is also scalable, maintainable, and aligned with Salesforce architecture principles. Where validated problems transform into structured, scalable, and implementable solutions.

In our project, "Airlines Management System," this phase bridges the gap between ideation and execution by converting insights from the previous requirement analysis well-structured CRM solution tailored for airline operations.

◆ Problem–Solution Fit

Problem Recap:

Airlines operate in a complex environment requiring real-time tracking of flights, airplanes, crew members, airports, and fare management. However, many airline processes are still managed manually or isolated in separate systems, leading to:
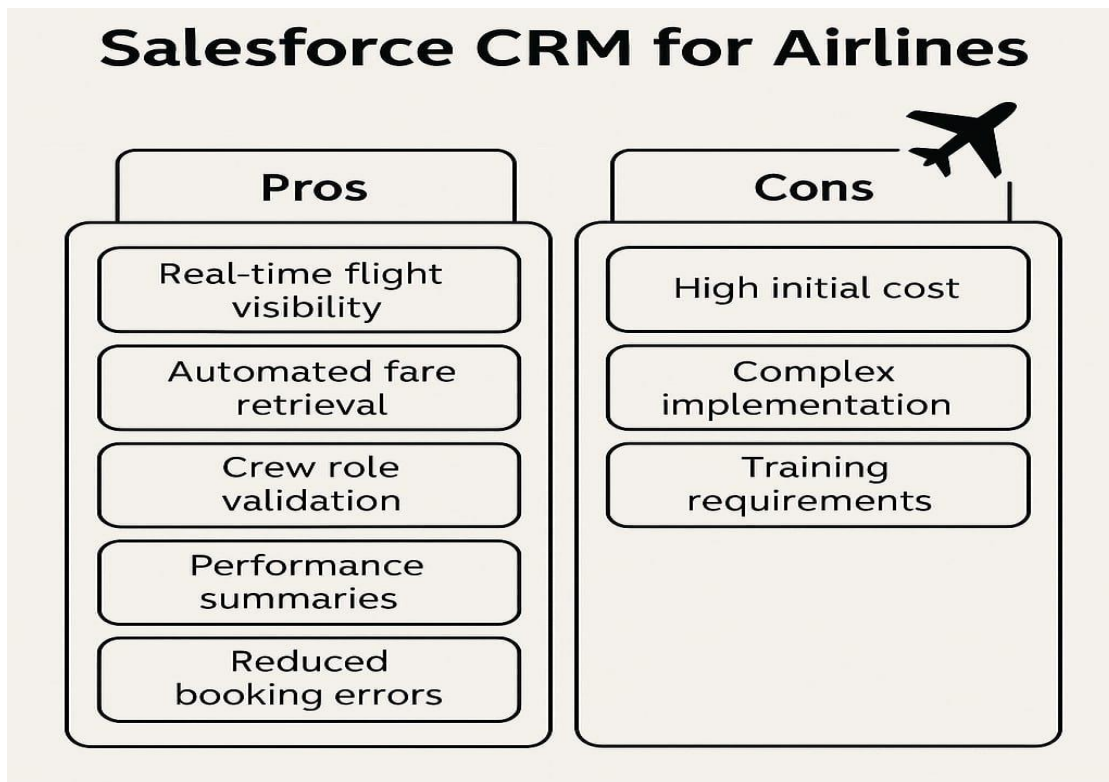
- Data entry errors and duplications

- Lack of real-time visibility for flight scheduling and operations

- Improper assignment of Pilots/Cabin crew

- Challenges in fare management, flight reporting, and occupancy tracking

Does the Proposed Solution Fit?

Yes. The Salesforce CRM solution for Airlines Management directly addresses the critical pain points in airline operations. The solution:

- Introduces **object-level control** for Airplanes, Airports, Employees (Pilots, Cabin Crew), Flights, and Ticket Fares

- Enables **automated fare retrieval** through Flows

- Validates pilots and cabin crew assignments using **Apex Triggers**

- Summarizes performance using **Reports and Dashboards**

- Uses formula fields to reduce calculation errors

- Creates centralized views using **Lightning App Builder**

Thus, it directly fits the core operational pain points of Airlines workflows.

Salesforce CRM for Airlines

Pros	Cons
Real-time flight visibility	High initial cost
Automated fare retrieval	Complex implementation
Crew role validation	Training requirements
Performance summaries	
Reduced booking errors	

◆ Proposed Solution

How Our CRM Will Solve the Identified Problems

Our proposed Airlines CRM application is designed to digitize and streamline Airline operations using Salesforce's declarative and programmatic capabilities.
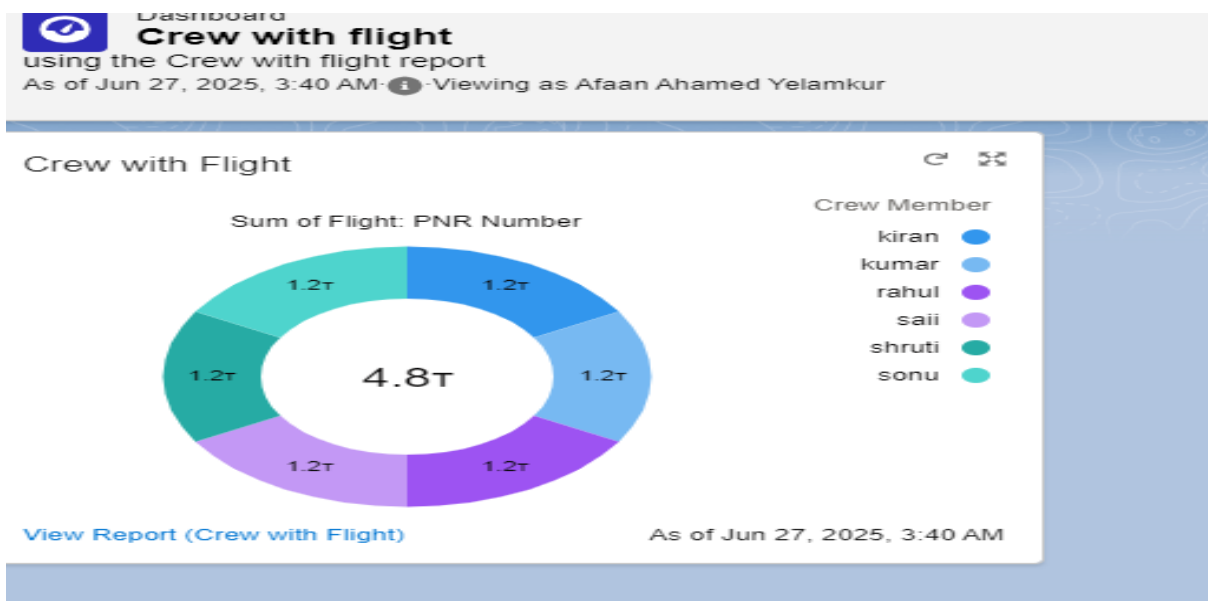
Key Functional Features:

- **Custom Objects**:

  - Airport, Airplane, Flight, Ticket Fare, Employee

- **Automation & Validation**:

  - Role verification (Pilot/Cabin crew) via **Apex Triggers**

  - Fare calculation automation via **Flows**

  - Input control via **Validation Rules**

- **Formula Fields for Efficiency**:

  - Age, Experience, Date of Retirement for employees

  - Total Fare = Passenger Count × Ticket Fare

  - Pilot & Cabin crew Names (auto-derived via lookups)

- **UI & Navigation**:

  - Public Transport App using **Lightning App Builder**

  - Tabs for all custom objects

- Page layouts designed by object and role

- **Reports and Dashboards**:

  - Trip Summary Reports

  - Cabin crew with Flight Assignment Reports

  - Monthly Revenue and Passenger Count Dashboards

Together, these components will offer a centralized, role-driven, and insight-ready system.

◆ Solution Architecture

Visualizing the Technical Structure and Object Relationships

Object Relationship Overview:

Object	Key Fields / Features
Airport	Name, Category, Amenities, Address
Airplane	Linked to Airport, Category, Model, Capacity
Airline Employee	Name, Role (Picklist), DOB, Experience, Lookup to Airport
Flight	Linked to Airplane, Pilot, Cabin Crew, Ticket Fare, Flight Date, Passenger Count, Total Fare
Ticket Fare	Route, Airplane Model, Fare

Lookups:

- Employee → Airport
- Flight → Airplane, Employee (Pilot), Employee (Cabin Crew), Ticket Fare
- Airplane → Airport

Formulas:

- Total_Amount__c = Passenger_Count__c * Ticket_Fare__c

- Pilot_Name____c = Pilot_Id____r.Employee_Name__c

- Cabin_crew_Name_c = Cabin_crew_Id_r.Employee_Name__c

Automation:

- Flows for fare fetch logic

- Triggers for role validation

- Reports & dashboards for output

Summary

The Project Design Phase ensured that our Airlines CRM not only met the users' needs but also followed Salesforce best practices in object modeling, validation, automation, and user experience. This clear blueprint guided our execution in upcoming development and configuration phases.