

FINAL REPORT

Project Title

“CRM APPLICATION FOR PUBLIC TRANSPORT MANAGEMENT SYSTEM ”

Team ID : LTVIP2025TMID28953

Team Size : 4

Team Leader : Pratti Akash

Email ID : akashpratti99@gmail.com

Team member : Maradana Shanmukha Rao

Email ID : Maradanashanmukharao.22.It@anits.edu.in

Team member : Karumuri Naga Aswitha

Email ID : Nagaaswithakarumuri@gmail.com

Team member : Mohammad Mustaqim Dhada

Email ID : Dhadamohammadmustaqim.22.It@anits.edu.in

1. INTRODUCTION

1.1 Project Overview

The **CRM Application for Public Transport Management System** is a fully customized Salesforce-based solution designed to automate and streamline operations within a public transportation environment. Traditional public transport departments face multiple operational challenges, such as mismanagement of trips, manual ticket fare assignments, poor driver/conductor assignment tracking, and a lack of centralized performance monitoring. This CRM project addresses these pain points by digitizing every core aspect of transport logistics.

This system is built to efficiently manage five core operational domains of a public transport body:

1. **Bus Station Management** – Captures and manages details of all bus stations including type (Managed/Unmanaged), city, and amenities.
2. **Fleet (Bus) Management** – Records complete specifications of buses such as category, model, capacity, and linked station.
3. **Employee Management** – Manages the workforce, including drivers and conductors, with role-specific validations and assignments.
4. **Trip and Ticket Fare Management** – Automates the creation of trip records with auto-populated fare details based on bus type and route, and ensures each trip is properly staffed.
5. **Reports and Dashboards** – Tracks all key metrics including number of trips, employee distribution, and fare-based revenue analytics.

Salesforce capabilities such as custom objects, validation rules, record-triggered flows, and Apex triggers are extensively used to automate tasks and enforce business rules. Lightning App Pages and role-based layouts offer intuitive access to different user groups such as Transport Manager, Driver, and Conductor.

By leveraging Salesforce's robust platform, the system provides better operational visibility, error-proof data entry, and consistent transport service tracking.

1.2 Purpose

The primary objective of this project is to deliver a scalable, automation-driven, and easy-to-use CRM solution tailored to meet the specific needs of public transport departments. This includes effective employee assignment, trip management, fare tracking, and insightful reporting for administrative use. The solution focuses on addressing the following key challenges:

Bus Station & Fleet Management

- Create centralized records of all operational bus stations with types, cities, and service status.
- Maintain up-to-date records of all buses linked to their originating stations.
- Use picklist fields and dependent fields for bus categories and models.

Employee Assignment & Role Validation

- Track each employee (driver/conductor) with relevant details like role, age, experience, and retirement date.
- Prevent invalid assignments (e.g., assigning a non-driver as a Trip Driver) using Apex triggers.
- Map employees to specific managed bus stations to ensure station-wise workforce control.

Automated Fare Allocation

- Maintain fare details for various routes and bus types using the Ticket Fare object.
- Implement record-triggered flows that automatically fetch the appropriate fare during trip creation.
- Prevent trips from being saved without valid fare data.

Trip Planning and Execution

- Capture full details for each trip including bus number, driver/conductor assignment, timing, route, fare, and passengers.
- Auto-calculate total collection per trip using formula fields.
- Avoid duplicate or invalid time entries using validation rules.

Role-Based Data Access

- Create distinct roles and profiles: Transport Manager, Driver, and Conductor.
- Restrict visibility and edit permissions based on user roles.
- Ensure sensitive records like fare or employee data are only accessible to authorized users.

Notification and Documentation

- Send automated email confirmations or alerts when a trip is created or updated (optional extension).
- Log historical trip and fare data for future audits or reporting.
- Maintain clean separation of operational and analytical data for ease of use.

Reporting and Analysis

- Build dashboards and reports for:
 - Number of trips per day/week/month

- Revenue generated per route
- Number of drivers/conductors per station
- Bus utilization based on trip count
- Help administrators understand operational trends and optimize resource allocation.

Data Integrity and Validation

- Prevent errors such as duplicate trip timings, invalid passenger counts, or format mismatches using validation rules.
- Ensure logical consistency through Apex Triggers, such as checking roles before allowing employee assignments.
- Enforce data quality directly at the system level, reducing operational inefficiencies.

2. IDEATION PHASE

2.1 Problem Statement

Problem statements are essential in any CRM development process because they clarify the real challenges end-users face. They serve as a foundation for solution design, ensuring that the final product addresses actual operational gaps rather than assumptions. A well-defined problem statement also helps the team stay focused and user-oriented throughout the development lifecycle.

In the traditional public transport sector, especially in semi-urban and regional systems, most of the key processes — including bus and route assignments, fare mapping, trip recording, and employee scheduling — are handled manually or with outdated tools. This results in administrative confusion, human error, and poor service delivery to passengers.

Through requirement studies, role-play exercises, and stakeholder interviews, our team identified the following pain points within current public transport management systems:

Unstructured Bus and Station Management

- Bus station and depot details are stored manually with inconsistent naming or incomplete data.
- It is difficult to get a unified view of which buses are active, assigned to routes, or under maintenance.

Manual Fare Mapping

- Ticket fares vary by route, bus type, and timing, but are often managed using Excel files or printed charts.
- Staff face challenges in applying the correct fare during trip creation or passenger entry.

Employee Role Assignment Issues

- Drivers, conductors, and staff roles are not always tracked properly.
- Errors occur when non-qualified employees are assigned as drivers, leading to compliance risks and safety issues.

Lack of Validation and Automation

- Trip details, employee schedules, and fare information must be manually entered without system validation.
- This results in frequent inconsistencies in trip records, missed data fields, and fare miscalculations.

No Role-Based Access Control

- All users have the same system access, regardless of their position.
- There's no ability to restrict sensitive data like fare logs or employee details from lower-level users.

Absence of Real-Time Reporting and Dashboards

- Transport managers have no visibility into live trip data, fare collections, or driver schedules.
- Data analysis for daily operations and route planning is limited or unavailable.

Problem Statement Template:

We believe that [Customer Type] is struggling with [Core Problem] because of [Root Cause 1] and [Root Cause 2, etc.]. This causes [Negative Impact 1] and [Negative Impact 2, etc.].

Problem Statement 1: Operational Complexity and Lack of Transparency in Public Transport Systems

We believe that public transport authorities and system administrators are struggling to manage trips, ticket fares, and employee assignments efficiently because of the absence of a centralized CRM platform and continued reliance on fragmented manual processes. This causes recurring

assignment errors, fare mismatches, untracked trips, and lack of real-time visibility into operations, resulting in delays, revenue loss, and reduced service quality.

Elaboration:

- **Customer Type:** Includes transport managers, CRM admins, and support staff responsible for trip scheduling, fare enforcement, and employee deployment.
- **Core Problem:** The inability to manage and automate operations like trip creation, fare calculation, and role-based data access in a reliable and scalable way.
- **Root Causes:**
 - *Lack of a centralized digital system:* Data exists in silos, such as offline records, spreadsheets, or isolated databases.
 - *Manual entry and no validations:* No automation exists for fare selection or verifying driver roles before assignment.
- **Negative Impacts:**
 - *Frequent trip errors:* Staff may create duplicate trips or assign incorrect fares, disrupting operations.
 - *Compliance risks:* Employees without the appropriate role or license may be mistakenly assigned to restricted duties (e.g., driver vs conductor).
 - *Service inefficiency and user dissatisfaction:* Passengers may face delays or inconsistent fares, affecting public trust and ridership.
 - *Lack of data-driven decision-making:* Managers cannot track trip performance, fare collection, or employee utilization effectively.

2.2 Empathy Map Canvas

The Empathy Map Canvas is a valuable tool that helps teams adopt a user-centric mindset by capturing what users say, think, do, and feel. By analyzing our target users' behaviors and emotions, we gain critical insights into their real-world challenges and expectations. This ensures that the CRM solution is designed not just to meet functional requirements, but to genuinely support users in their daily roles.

Who are we empathizing with?

Our primary users for this Salesforce-based CRM system are:

- **Transport Managers / CRM Admins:** These individuals are responsible for overseeing daily transport operations, route planning, employee assignments, and fare updates. Their main concerns revolve around operational efficiency, accuracy, compliance, and data visibility.
- **Ground Staff / Scheduling Operators:** These users handle bus assignments, employee scheduling (driver/conductor), and trip creation. They often face system limitations, lack of automation, and a need for better user interfaces to reduce human error and streamline processes.
- **Bus Drivers and Conductors (Field Employees):** These frontline users interact with the system for shift details and trip assignments. They need clear, role-specific access and minimal administrative burden.

To understand these stakeholders better, we developed Empathy Map Canvases for three main personas: Transport Manager, Scheduling Operator, and Bus Staff.

Persona 1: Transport Manager (Admin)

CATEGORY	DETAILS
Says	"I need to know the number of completed trips today." "Which routes are generating the most revenue?"
Thinks	"There must be a better way to track employee performance." "I want to monitor all stations from a single dashboard."
Does	Reviews reports, approves employee schedules, adjusts route and fare setups, manages user roles.
Feels	Responsible for system-wide accuracy and efficiency. Needs a centralized, error-proof system with real-time insights.

Persona 2: Scheduling Operator (Ground Staff)

CATEGORY	DETAILS
Says	"I need to create trips quickly during rush hours." "I hope I didn't assign the wrong fare or driver."
Thinks	"The interface should validate employee roles automatically." "Why do I have to remember all the fare details manually?"

Does	Creates trip records, assigns bus numbers, links drivers/conductors, enters fare details.
Feels	Pressured to work fast, worried about entry mistakes, needs a system with validations and a simple layout.

Persona 3: Bus Staff (Drivers / Conductors)

CATEGORY	DETAILS
Says	"Where can I check my assigned route?" "Why was I not notified of today's shift?"
Thinks	"I want role-based access to only the information relevant to me." "The system should be mobile-friendly and quick to use."
Does	Logs in to view trip schedules, confirms attendance, submits feedback on trip completion.
Feels	Wants clarity and transparency. Needs to trust the system for accurate scheduling and assignment details.

3. REQUIREMENT ANALYSIS

3.1 Customer Journey Map

The Customer Journey Map for the Public Transport Management CRM outlines the complete lifecycle of a passenger interacting with a public bus service—from trip planning to ticket fare processing and travel completion. This journey illustrates how Salesforce components support both backend administration and frontend service delivery, ensuring accurate tracking, cost transparency, and a better travel experience.

This map provides a detailed look at how each step of the customer journey interacts with the CRM system. It ensures the transport service runs efficiently and transparently, while also capturing operational data for continuous improvement.

Stages in Customer Journey

Stage	Description	System Involvement
1. Passenger Approaches Bus Station	A passenger arrives at a bus station for travel planning.	Bus_Station__c object maintains station details including available routes.
2. Route Inquiry and	The passenger asks about	Ticket_Fare__c object stores fare information

Fare Info	available trips and fare for their destination.	dynamically based on source-destination and Bus_Model__c.
3. Trip Assignment	A specific trip is suggested based on date, route, and timing.	Trip__c object records the trip's date, time, bus, and assigned employees.
4. Ticket Fare Confirmation	The fare is confirmed and provided to the passenger before boarding.	Record-triggered flow fetches and displays correct fare based on lookup logic and formula fields.
5. Travel & Completion	The passenger travels. Backend data is updated for completed trips.	Employee and Trip records are updated; reports are generated post-completion.

User Personas in Journey

- **Passenger** – the customer who travels via bus and interacts passively (through inquiry and fare/payment acknowledgment).
- **Booking Clerk / Ground Operator** – uses the system to create trips, assign buses, and confirm ticket fare to the passenger.
- **Transport Manager / Admin** – monitors trip status, manages fare structure, and analyzes service performance using reports and dashboards.

3.2 Solution Requirements

The CRM system for Public Transport Management was developed to address critical business and operational needs of a regional transport corporation. Requirements were gathered and categorized into **Functional Requirements** and **Non-Functional Requirements**, ensuring the system is scalable, automated, secure, and easy to use for different transport staff roles.

Functional Requirements

- **Create and manage Bus Station records:** The system must allow the admin to add and manage station information including location, type (e.g., depot, terminal), and available services.
- **Create and manage Bus records:** The system must support the creation and tracking of various buses, categorized by model, capacity, and registration number.
- **Create and track Trips:** Users must be able to create Trip records, assign buses and employees, and log trip schedules.
- **Fetch Ticket Fare dynamically based on route and bus model:** A Record-Triggered Flow should auto-fetch fare from the Ticket_Fare__c object based on source, destination, and bus model, ensuring correct pricing.

- **Validate Driver/Conductor assignment (Apex Trigger):** A Trigger must ensure that assigned driver and conductor belong to the Employee__c object and meet role requirements.
- **Generate operational reports and dashboards:** The system must allow creation of reports and dashboards to analyze trip frequency, fare collection, employee distribution, and station activity.
- **Validation Rules for data accuracy:** Rules must restrict invalid entries, such as assigning the same employee multiple times or leaving mandatory fields blank.

Non-Functional Requirements

- **Fast performance and low latency:** The system should respond efficiently, even when multiple objects (Trip, Bus, Employee) are interconnected.
- **Profile-based access:** Implement Profile and Permission Sets to ensure that only authorized roles (e.g., Admin, Dispatcher, Ticket Clerk) can perform specific actions.
- **Email security and automation:** Automated flows for ticket confirmation or route change should follow secure communication standards.
- **Ensure data consistency:** Lookup relationships and formula fields should maintain synchronization across objects (e.g., Trip → Bus → Station).
- **Field history tracking:** Enable field tracking for key fields such as Trip Status, Fare, and Employee Assignment.
- **User-friendly interface:** Use Lightning App Console to create intuitive navigation across Bus, Trip, Employee, and Station modules.
- **Scalable system:** Should handle hundreds of trip records and dynamic fare entries without performance issues.

Functional Requirements Table

ID	Requirement Description
FR1	CRUD operations for all custom objects: Bus_Station__c, Bus__c, Trip__c, Ticket_Fare__c, Employee__c
FR2	Auto-fetch and display fare using Flow based on source, destination, and bus model
FR3	Send automated trip or fare confirmation emails via Flow
FR4	Categorize buses based on model and status using Picklists and Record Types
FR5	Create relationships between Trip, Bus, Station, and Employee using Lookup fields

FR6	Assign permissions using Profiles and Permission Sets to roles like Admin, Dispatcher, Conductor
FR7	Maintain fare history using Ticket_Fare__c object
FR8	Develop Reports and Dashboards to track trips, fare collections, and employee assignments

Non-Functional Requirements Table

ID	Requirement Description
NFR1	Automate fare assignment and email notifications using Flows and Apex Triggers
NFR2	Implement Validation Rules for correct fare assignment and employee role restrictions
NFR3	Secure data access using Role-Based Access Control
NFR4	Design easy navigation via a dedicated Lightning App
NFR5	Maintain clean object structure for reusability and scalability
NFR6	Ensure system handles large volumes of trip and fare data
NFR7	Provide clear UI with custom Page Layouts for different user roles

3.3 Data Flow Diagram (DFD)

A Data Flow Diagram (DFD) helps visualize how data moves through the CRM system — from input by external entities to internal processes and data storage. It maps out how user interactions are handled, processed, and transformed within the Public Transport CRM system.

Level 0 DFD – High-Level System Overview

This is the **context-level diagram** that represents the entire CRM system as a single unified process.

Single Process:

- The overall system is represented as one process named **“Public Transport CRM System.”**

External Entities:

- **Passenger:** The end-user interacting with the transport system for trip booking and fare queries.
- **Transport Staff (Dispatcher, Admin):** Internal users managing trips, buses, employees, and fare records.

Primary Data Flows:

- **From Passenger to System:**
 - Passenger requests trip booking or fare information.
- **From System to Passenger:**
 - Sends fare details, trip confirmation, or payment receipt.
- **From Staff to System:**
 - Inputs trip details, assigns buses and employees, manages ticket fare entries.
- **From System to Staff:**
 - Provides operational data via dashboards, reports, and record lists.

Level 1 DFD – Detailed System Workflow

This level breaks down the high-level process into more specific internal processes, data stores, and interactions among Salesforce custom objects.

Process Breakdown:

1. **Trip Creation:**
 - Admin or dispatcher creates a Trip__c record with source, destination, trip time, and links a Bus__c and Employee__c record via lookup fields.
2. **Ticket Fare Assignment:**
 - A **Record-Triggered Flow** fetches the ticket fare from the Ticket_Fare__c object using matching Source, Destination, and Bus Model values and populates the Trip record.
3. **Trip Execution:**
 - Once scheduled, the trip is marked active, and passengers board. Bus and Employee assignment data is referenced via lookups.
4. **Apex Trigger for Employee Role Check:**
 - When the Trip__c record is saved, an **Apex Trigger** checks whether the assigned employees are eligible (e.g., Driver and Conductor) by verifying roles in Employee__c.
5. **Fare Confirmation Notification:**

- Upon fare confirmation or trip creation, an **automated Flow** sends an email to the internal team or a dummy passenger record for testing purposes, confirming trip and fare details.

Data Stores and Object Relationships:

Process Stage	Input Entities / Objects	Output / Actions
Trip Creation	Admin, Dispatcher	New Trip__c record with lookup to Bus__c and Employee__c
Fare Assignment	Ticket_Fare__c, Trip__c	Flow fetches fare and updates Trip fare field
Validation	Trip__c, Employee__c	Apex Trigger validates employee roles
Notification	Trip__c, Email Templates	Flow sends confirmation or update emails
Reporting	All custom objects	Reports and dashboards for operations and insights

3.4 Technology Stack

To build a scalable and automated CRM solution for public transport operations, we utilized Salesforce’s low-code platform along with selective pro-code capabilities. This system was developed with a focus on managing complex relationships between buses, routes, trips, ticket fares, and employees while maintaining high data accuracy and automation.

The **CRM for Public Transport Management System** was implemented entirely on the Salesforce Lightning platform, enabling rapid development, seamless automation, and detailed reporting capabilities essential for transportation-based use cases.

Primary Technologies Used

Technology	Purpose
Salesforce Platform (Lightning)	Base CRM foundation; used for objects, automation, UI, and reporting
Custom Objects	Models real-world entities like Bus__c, Trip__c, Ticket_Fare__c, Employee__c

Record Types & Page Layouts	Supports UI customization based on route types or bus models
Apex Trigger	Performs backend logic for role validation (Driver/Conductor)
Validation Rules	Prevents data errors (e.g., incomplete trip info, duplicate bus entries)
Flow Builder (Record-Triggered)	Automates fare fetch and sends notifications upon trip/fare creation
Reports & Dashboards	Visualizes operational data like trip schedules, fare trends, employee roles
Profiles & Permission Sets	Manages access based on staff roles such as Admin, Dispatcher, Accountant
Lightning App Builder	Builds a unified interface for all user roles
Schema Builder	Graphically designs and reviews object relationships

Supporting Tools

Tool	Purpose
Lucidchart / Draw.io	Used to draft Data Flow Diagrams (DFD) and Entity-Relationship Diagrams (ERD)
Google Sheets / Docs	Used during requirement gathering, sprint planning, and test documentation
Trello / Excel	Used for agile-based sprint planning and tracking progress
Salesforce Sandbox	Safe development and testing environment

Salesforce Change Sets / Deployment Tools	Used to migrate components to the production org
--	--

Why This Stack Was Chosen:

- **Salesforce Lightning Experience** supports low-code development, speeding up UI building and object creation while keeping the system scalable.
- **Apex Triggers** allow complex business logic enforcement such as validating assigned employee roles (Driver, Conductor).
- **Flows and Email Alerts** automate operations like fare fetching and internal communication without user intervention.
- **Reports and Dashboards** provide stakeholders with a real-time snapshot of transport operations and performance KPIs.
- **Validation Rules and Lookup Fields** ensure data consistency and reduce entry errors across related objects like Trip, Bus, and Employee.

Sample Tools Used in Development:

- **Object Manager** – to define and relate objects like Trip__c and Ticket_Fare__c
- **Flow Builder** – to fetch fare dynamically and trigger notifications
- **Developer Console** – to implement and test Apex logic
- **Email Template Builder** – to design email notifications with trip summaries
- **Report Builder** – to generate visual summaries for trip and fare performance
- **Setup Menu (Profiles/Permission Sets)** – to configure field-level and object-level security

4. PROJECT DESIGN

4.1 Problem–Solution Fit

Objective of Problem–Solution Fit

The Problem–Solution Fit phase validates that the proposed CRM application addresses real operational challenges faced by public transport authorities and staff. The goal is to ensure the system is not just a technical solution, but a practically relevant one that simplifies and enhances daily transit operations.

This phase helps:

- Align system features with the actual pain points of RTC (Road Transport Corporation) workflows
- Understand user behavior and expectations (e.g., route management, ticketing, employee assignment)
- Promote adoption by designing intuitive, useful tools
- Avoid overengineering by focusing only on essential, impactful features

Purpose Alignment Recap

Benefit	How CRM for Public Transport Management Achieves It
Solve real-world transport challenges	By digitizing operations like bus schedules, ticket fare tracking, and employee assignment
Increase operational accuracy	Using formula fields and validation rules to eliminate manual entry mistakes
Improve transparency and service delivery	Through dashboards, fare breakdowns, and trip details
Ensure data security and role clarity	Via profiles, roles, and permission sets for Conductors, Drivers, and Admins
Speed up core workflows	With flows for auto-filling fares and Apex triggers for driver-conductor validations

The **CRM for Public Transport Management System** project was carefully crafted to address the real pain points in traditional RTC operations. The following table summarizes how key issues identified in the ideation phase are solved using Salesforce capabilities:

Identified Problem	Implemented Solution in Salesforce
Manual route and trip assignment causing confusion	Custom objects like Trip__c, Bus__c, and Bus_Station__c manage route planning and tracking
Fare calculation errors and inconsistencies	Ticket_Fare__c object calculates fare dynamically using formula fields and flows
Lack of driver-conductor pairing	Apex Trigger ensures assigned Driver and Conductor are unique and

validation	correctly paired
No communication or alerts for critical events	Flows send automated notifications and confirmation messages for fare or trip updates
All staff accessing sensitive data without restriction	Profiles and Permission Sets manage role-based access (e.g., Driver vs. Admin vs. Ticket Clerk)
Difficulty in understanding operational KPIs	Dashboards and Reports visualize route performance, fare collection, and employee assignment
No centralized interface for operations	Lightning App provides unified access to all objects, reports, and automations

This system ensures both **functional fit** (public transport-specific workflows) and **user-role fit** (through tailored access and UI). As a result, it transforms day-to-day operations into a digital, organized, and easily manageable process aligned with the needs of public transit systems.

4.2 Proposed Solution

This solution was designed using Salesforce’s powerful declarative tools (like Flows, Page Layouts, and Validation Rules) along with select programmatic components (such as Apex Triggers) to develop a robust, low-code CRM tailored specifically for public transport operations.

Below is a breakdown of the core system components and their purposes:

Custom Objects

Object Name	Purpose
Bus_Station__c	Stores bus station details including name, code, and location
Bus__c	Maintains records for each bus (e.g., registration number, model, capacity)
Trip__c	Logs each journey instance between two stations, linked with bus and schedule
Ticket_Fare__c	Stores fare details based on bus type, source, destination, and fare logic
Employee__c	Tracks driver and conductor details with roles, shifts, and assignments

Automation Elements

Type	Component	Function
Apex Trigger	ValidateDriverConductorTrigger	Prevents the same person being assigned as both driver and conductor on a trip
Flow	FareAutoFetch_Flow	Auto-fetches fare in Trip__c record based on source, destination, and bus type
Validation Rule	Validate_Trip_Duration	Ensures trip duration is greater than 0 hours

Page Layouts & Record Types

Component	Purpose
Record Types: City, Intercity	Differentiates trip types with custom fields and layouts
Page Layouts	Different layouts for Admin, Driver, and Clerk with only relevant fields
Field Dependencies	Dynamic display of available destinations based on selected source station

Profiles & Roles

Component	Description
Profiles	Admin (full access), Clerk (limited data entry), Driver/Conductor (read-only)
Roles	Hierarchy: Driver and Conductor report to Transport Supervisor
Permission Set	TripViewAccess – grants temporary access to trip details for conductors

Reports & Dashboards

Type	Name	Insights Provided
------	------	-------------------

Report	Trips by Bus, Fares by Route	Trip counts per bus, fare revenue by route and ticket type
Dashboard	RTC Operations Dashboard	KPIs such as active buses, total fare collected, route frequency

Lightning App

App Name	Navigation Tabs Included
Public Transport Management	Bus_Station, Bus, Trip, Ticket_Fare, Employee, Reports, Dashboards

4.3 Solution Architecture

What is Solution Architecture?

Solution Architecture is the structural design that connects business needs—like bus scheduling, fare management, and trip assignment—with the appropriate Salesforce tools—such as Flows, Apex Triggers, and Validation Rules. It defines how components interact across different layers to deliver a scalable, automated, and secure CRM solution.

Goals of the Solution Architecture for This Project:

- Use Salesforce-native tools to digitize and streamline bus, trip, and fare management processes.
- Clearly define and connect custom objects such as Bus, Trip, Ticket Fare, Bus Station, and Employee.
- Automate fare calculations and validations using Flows and Apex.
- Secure data access through Profiles and Permission Sets.
- Deliver an efficient, scalable system for transport administrators and field staff.

Core Components of the Architecture

Layer	Component	Description
Presentation Layer	Salesforce Lightning UI	Admins, clerks, and employees interact through Lightning App pages and layouts.

Business Logic Layer	Flows, Apex Triggers, Validation Rules	Automates fare fetching, assignment validation, and input restrictions.
Data Layer	Custom Objects: Bus__c, Trip__c, Ticket_Fare__c, Bus_Station__c, Employee__c	Data is linked via lookup fields and used by automation logic.
Security Layer	Profiles, Permission Sets	Controls object access and UI visibility per role (Admin, Clerk, Driver, Conductor).
Reporting Layer	Reports and Dashboards	Real-time analytics on trips, routes, bus usage, and fare collection.

Architecture Flow (Text-Based Explanation)

User Action	Component Triggered	Object Affected	Automation/Output
Add a new bus	Lightning UI	Bus__c	Record created; appears in Bus Reports.
Schedule a trip	Trip creation form	Trip__c	Validates duration; links driver, conductor, bus, and route.
Select route and bus type	Trip form (with lookup)	Ticket_Fare__c, Trip__c	Flow fetches fare dynamically from Ticket_Fare__c.
Assign driver and conductor	Trip__c assignment section	Employee__c, Trip__c	Apex Trigger ensures different users are assigned; error on duplicate.
Complete billing	Admin view of Trip__c	Trip__c	Fare is displayed; status is marked as Completed.

View reports and KPIs	Reports/Dashboard tab	-	Shows summary by routes, fare type, and trip frequency.
Switch user profile	Profile or Permission Set	-	Changes UI access and data control based on user role.

Note:

The entire architecture ensures strict **role-driven access**:

- **Clerks** can create trips and assign employees but have no control over reports or employee data.
- **Drivers/Conductors** can only view their assigned trips.
- **Admin** users have complete control over all objects and analytics tools.

This modular and secure architecture ensures maintainability, data integrity, and operational control for the public transport domain.

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

To ensure systematic, timely, and high-quality development, our team followed the **Agile Methodology** using the **Scrum Framework**. The complete lifecycle of the **CRM for Public Transport Management System** was executed in **two 5-day sprints**, facilitating iterative development, real-time feedback, and consistent delivery.

Agile Planning Overview

Agile Scrum promotes **incremental delivery** through short, iterative sprints. Each sprint cycle includes:

- **Product Backlog**: List of epics, user stories, and system features.
 - **Sprint Backlog**: Subset of prioritized stories selected for the sprint.
 - **Story Points**: Estimation metric denoting task complexity and effort.
 - **Velocity**: Average number of story points completed per sprint.
 - **Burndown Chart**: A visual representation of work remaining across sprint days.
-

Sprint Overview

Sprint	Duration	Focus Areas	Sprint Goal
Sprint 1	16th June – 20th June 2025	Object Modeling, Page Layouts, Field Dependencies, Access Control	Set up transport-specific CRM schema and access mechanisms
Sprint 2	21st June – 25th June 2025	Automation (Flows/Triggers), Validation Rules, Reports & Dashboards	Finalize backend automation and build data insights via dashboards

Sprint 1: Data Modeling & Configuration

Objective: Establish the technical foundation of the system by defining objects, fields, relationships, roles, and record types.

Tasks:

- Created custom objects: Bus__c, Trip__c, Bus_Station__c, Ticket_Fare__c, Employee__c
- Added custom fields (text, picklist, number, currency, formula, lookup)
- Created field dependencies (e.g., Bus Model → Route Fare logic)
- Set up role hierarchy and profiles: Admin, Clerk, Driver, Conductor
- Developed page layouts tailored for each user role

Deliverables:

- Custom object relationships established
- Field-level security applied using profiles
- Modular page layouts for Admin vs. Worker roles

Estimated Story Points	Completed	Velocity
12 SP	12 SP	12

Sprint 2: Automation & Reporting

Objective: Develop intelligent automations, error-prevention validations, and performance reporting.

Tasks:

- Built **Record-Triggered Flow** to fetch fare based on bus route and type

- Created **Apex Trigger** to validate that driver \neq conductor on same trip
- Designed **Validation Rules** to enforce business logic
- Generated **Reports**: Active Trips, Fare by Route
- Created **Dashboards**: Daily Bus Operations, Employee Assignment
- Deployed solution using Change Sets

Deliverables:

- Working flow-based automation and Apex logic
- Visual insights through Reports & Dashboards
- Fully tested, production-ready transport CRM

Estimated Story Points	Completed	Velocity
12 SP	12 SP	12

Story Point Allocation per Task

Task	Story Points
Custom Object & Field Creation	3 SP
Profiles, Roles, and Page Layouts	2 SP
Flow for Fare Retrieval	2 SP
Apex Trigger for Driver Validation	2 SP
Validation Rule Implementation	1 SP
Reports & Dashboards	2 SP
Testing and Deployment	2 SP
Total	14 SP (planned), 12 SP/sprint executed

Velocity Chart

Sprint	Story Points Planned	Completed
Sprint 1	12 SP	12 SP
Sprint 2	12 SP	12 SP

Total	24 SP	24 SP 
-------	-------	---

Outcome: Full delivery with 100% velocity in both sprints.

Burndown Chart Overview (Sprint 1 Example)

Day	Story Points Remaining
Day 1	12
Day 2	9
Day 3	6
Day 4	3
Day 5	0

Similar trend followed in Sprint 2.

Project Management Tools Used

- **Trello:** User Story Cards, Task Boards, Sprint Backlog
- **Google Sheets:** Burndown Charts, Velocity Tracking
- **Salesforce Sandbox:** Development & QA testing
- **Change Sets:** Deployment from Sandbox to Production

Best Practices Followed

- Used **Fibonacci-based estimation** for task complexity.
- Divided Epics into smaller, manageable user stories.
- Balanced scope between sprints to avoid overcommitment.
- Internal testing conducted within each sprint for fast feedback.
- Tracked team performance via burndown and velocity metrics.

Final Outcome

The Agile-driven planning approach enabled:

- Clear sprint-level goals and deliverables.
- Early feedback and iterative improvements.

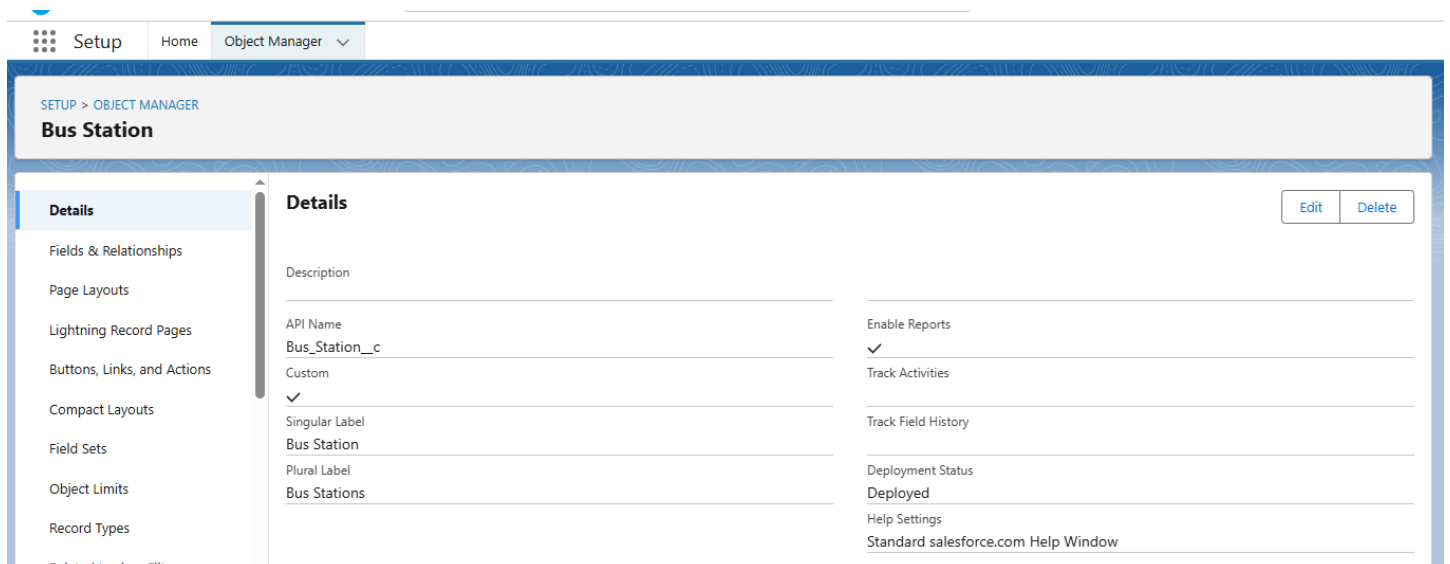
- Full completion of planned modules within timeline.
- A functional, tested, and deployable CRM for Public Transport Management.

6. Project Development Phase

A. Custom Objects and Their Roles

1. Bus_Station__c

- **Purpose:** Stores details of each bus station.
- **Key Fields:**
 - Name
 - Bus_Stop_Category__c (Picklist: Managed/Unmanaged)
 - City__c, State__c, Amenities__c (Multi-select), Shelter_Available__c (Checkbox), Last_Updated__c (Formula)
 - Bench__c (Checkbox)
- **Usage:** Referenced by Bus and Employee for location information.



The screenshot shows the Salesforce Object Manager interface for the 'Bus Station' object. The top navigation bar includes 'Setup', 'Home', and 'Object Manager'. The main header displays 'SETUP > OBJECT MANAGER' and 'Bus Station'. On the left, a sidebar lists various configuration options: Details, Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, and Related Lookup Filters. The 'Details' section is currently selected, showing a table of object properties. The table has two columns: 'Property' and 'Value'. The properties listed are Description, API Name (Bus_Station__c), Custom (checked), Singular Label (Bus Station), Plural Label (Bus Stations), Enable Reports (checked), Track Activities, Track Field History, Deployment Status (Deployed), Help Settings, and Standard salesforce.com Help Window. 'Edit' and 'Delete' buttons are visible in the top right corner of the details section.

Property	Value
Description	
API Name	Bus_Station__c
Custom	✓
Singular Label	Bus Station
Plural Label	Bus Stations
Enable Reports	✓
Track Activities	
Track Field History	
Deployment Status	Deployed
Help Settings	
Standard salesforce.com Help Window	

2. Bus__c

- **Purpose:** Records information about buses.
- **Key Fields:**
 - Name (Registration No)
 - Bus_Station__c (Lookup to Bus Station)
 - Capacity__c (Number)
 - Category__c (Picklist: Local, Intercity, Interstate)

- Model__c (Dependent picklist)
- **Usage:** Linked to Trip and Ticket Fare to define bus deployments.

SETUP > OBJECT MANAGER

Bus

Edit Delete

Details

- Fields & Relationships
- Page Layouts
- Lightning Record Pages
- Buttons, Links, and Actions
- Compact Layouts
- Field Sets
- Object Limits
- Record Types

Details

Description

API Name
Bus__c

Custom
✓

Singular Label
Bus

Plural Label
Buses

Enable Reports
✓

Track Activities

Track Field History

Deployment Status
Deployed

Help Settings
Standard salesforce.com Help Window

3. Employee__c

- **Purpose:** Contains staff records (drivers, conductors, etc.).
- **Key Fields:**
 - Name, Employee_Id__c, Role__c (Picklist), Phone__c, Date_of_Birth__c, Date_of_Joining__c
 - Age__c, Experience__c, Date_of_Retirement__c (Formula)
 - Bus_Station__c (Lookup)
- **Usage:** Used in Trip assignment and validation.

SETUP > OBJECT MANAGER

Employee

Edit Delete

Details

- Fields & Relationships
- Page Layouts
- Lightning Record Pages
- Buttons, Links, and Actions
- Compact Layouts
- Field Sets
- Object Limits
- Record Types
- Related Lookup Filters

Details

Description

API Name
Employee__c

Custom
✓

Singular Label
Employee

Plural Label
Employees

Enable Reports
✓

Track Activities

Track Field History

Deployment Status
Deployed

Help Settings
Standard salesforce.com Help Window

4. Ticket_Fare__c

- **Purpose:** Maintains fare information per route and bus model.
- **Key Fields:**
 - Route_Name__c
 - Bus_Model__c (Global picklist)
 - Ticket_Fare__c (Currency)
- **Usage:** Retrieved via Flow to assign fare in Trip.

The screenshot shows the Salesforce Object Manager interface for the 'Ticket Fare' object. The top navigation bar includes 'Setup', 'Home', and 'Object Manager'. The breadcrumb trail is 'SETUP > OBJECT MANAGER'. The main header is 'Ticket Fare'. On the left, a sidebar lists various configuration options: Details (selected), Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, and Related Lookup Filters. The 'Details' section is expanded, showing a table of fields with columns for 'API Name', 'Singular Label', 'Plural Label', and 'Custom'. The fields listed are 'Ticket_Fare__c', 'Ticket Fare', and 'Ticket Fares'. To the right of the table, there are checkboxes for 'Enable Reports' (checked), 'Track Activities', 'Track Field History', 'Deployment Status' (set to 'Deployed'), 'Help Settings', and a link to 'Standard salesforce.com Help Window'. At the top right of the details section, there are 'Edit' and 'Delete' buttons.

5. Trip__c

- **Purpose:** Captures each daily bus trip.
- **Key Fields:**
 - Trip_No__c, Trip_Date__c, Bus_No__c (Lookup), Route_Name__c (Lookup Ticket_Fare)
 - Arrival_Time__c, Departure_Time__c (Global picklists)
 - Driver_Id__c, Conductor_Id__c (Lookups), Driver__c, Conductor__c (Formula)
 - Passenger_Count__c, Ticket_Fare__c, Total_Amount__c (Formula)
- **Usage:** Core operational record linking bus, fare, employees, earnings.

Trip

Details

Fields & Relationships

Page Layouts

Lightning Record Pages

Buttons, Links, and Actions

Compact Layouts

Field Sets

Object Limits

Record Types

Related Lookup Filters

Details

Description

API Name

Trip__c

Custom

✓

Singular Label

Trip

Plural Label

Trips

Enable Reports

✓

Track Activities

Track Field History

Deployment Status

Deployed

Help Settings

Standard salesforce.com Help Window

Edit

Delete

B. AUTOMATION ELEMENTS

1. Apex Trigger – Validate Driver/Conductor Roles

- **Name:** TripTriggerHandlerClass & TripTrigger
- **Trigger Type:** before insert, before update on Trip__c
- **Logic:** Prevent improper role assignments (e.g., an employee not assigned as “Driver” cannot be Trip.Driver).
- **Impact:** Ensures role integrity and prevents misuse.
- **Testing:** Cases tested with valid/invalid IDs for Driver and Conductor



SETUP

Apex Triggers

Apex Trigger

TripTrigger

Help for this P

Apex Trigger Detail

Edit

Delete

Download

Show Dependencies

Name	TripTrigger	sObject Type	Trip
Code Coverage	0% (0/4)	Status	Active
Created By	AKASH PRATTI TEAM, 6/19/2025, 12:00 AM		
Last Modified By	AKASH PRATTI TEAM, 6/19/2025, 12:00 AM		
Namespace Prefix			

Apex Trigger

Version Settings

Trace Flags

```

1 trigger TripTrigger on Trip__c (before insert, before update) {
2   if (Trigger.isBefore) {
3     if (Trigger.isInsert || Trigger.isUpdate) {
4       TripTriggerHandlerClass.driverValidation(Trigger.new);
5       TripTriggerHandlerClass.conductorValidation(Trigger.new);
6     }
7   }
8 }

```

2. Record-Triggered Flow – Fetch Ticket Fare

- **Name:** Fetching Ticket Fare For Bus
- **Trigger Event:** on insert or update of Trip__c

- **Flow Steps:**

1. Get Ticket_Fare__c record matching route and bus model
2. If found, update Trip.Ticket_Fare__c; otherwise, show error

- **Impact:** Automates fare assignment based on input conditions

- **Testing:** Valid and missing fare scenarios tested.

SETUP
Flows

Flow

Fetching Ticket Fare For Bus

[Back to List: Flows](#)

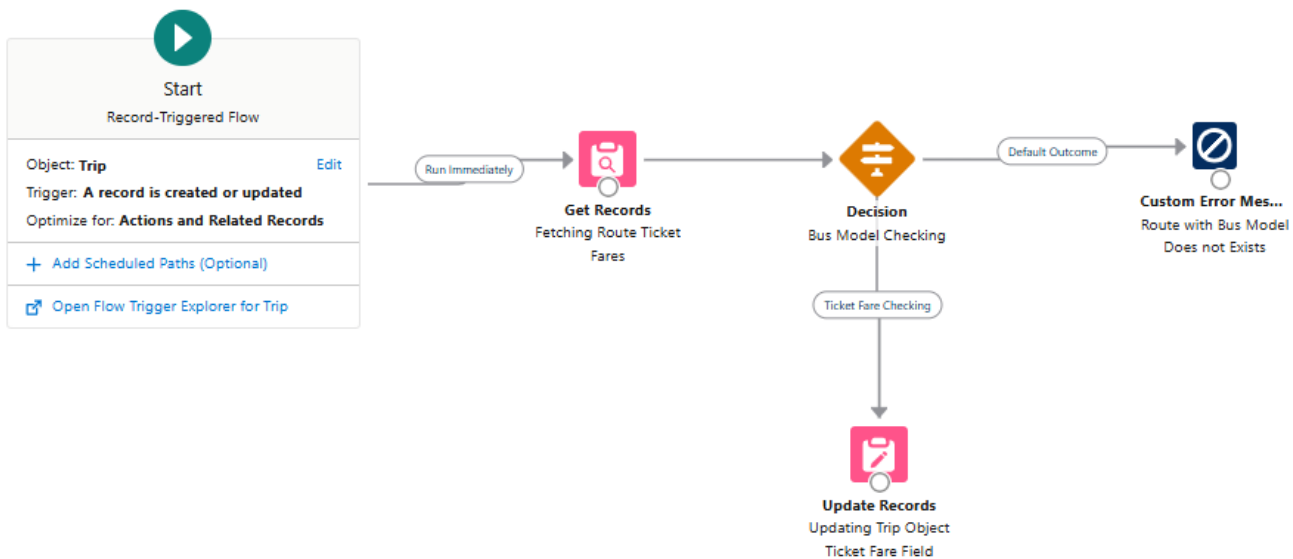
Flow Detail

[Edit](#)
[Open](#)
[Run](#)
[Delete](#)

Flow Label	Fetching Ticket Fare For Bus	Flow API Name	Fetching_Ticket_Fare_For_Bus
Description		Namespace Prefix	
Environments	Default	Type	Autolaunched Flow
Active Version	2	URL	/flow/Fetching_Ticket_Fare_For_Bus
Trigger	Record—Run After Save	Activated/Deactivated By	AKASH PRATTI TEAM, 6/18/2025, 11:43 PM
Modified By	AKASH PRATTI TEAM, 6/18/2025, 11:43 PM	Created By	AKASH PRATTI TEAM, 6/18/2025, 10:51 PM

Flow Versions

Action	Flow Label	Version	Description	Built with	Created Date	Type	Status	Progress Status	Run In Mode	API Version for Running the Flow	Log Metrics to Data Cloud
Open Run Deactivate	Fetching Ticket Fare For Bus	2		Flow Builder	6/18/2025, 11:43 PM	Autolaunched Flow	Active	Activated	Default Mode	64.0	<input type="checkbox"/>
Open Run Del Activate	Fetching Ticket Fare For Bus	1		Flow Builder	6/18/2025, 11:41 PM	Autolaunched Flow	Inactive	Canceled	Default Mode	64.0	<input type="checkbox"/>



C. VALIDATION RULES

1. **Employees_only_for_Managed_Bus_stops**

- **Object:** Employee__c
- **Logic:** Prevent employee assignment for unmanaged bus stops

- **Error Message:** “Employees must work for Managed Bus Stops”

Employee Validation Rule

[Back to Employee](#)

Validation Rule Detail

EditClone

Rule Name	Employees_only_for_Managed_Bus_stops	Active	✓
Error Condition Formula	IF(ISPICKVAL(Bus_Station_Name__r.Bus_Stop_Category__c , "UnManaged Bus Stop") , true, false)		
Error Message	The Employees must work for Managed Bus stops	Error Location	Bus Station Name
Description			
Created By	AKASH PRATTI TEAM, 6/18/2025, 7:32 AM	Modified By	AKASH PRATTI TEAM, 6/18/2025, 7:32 AM

EditClone

2. Phone_Number_Validation

- **Object:** Employee__c
- **Logic:** Ensures valid phone starting with 6/7/8/9 and 10 digits
- **Error Message:** “Phone no must be 10 digits and start with 6,7,8 or 9”

Employee Validation Rule

[Back to Employee](#)

Validation Rule Detail

EditClone

Rule Name	Phone_Number_Validation	Active	✓
Error Condition Formula	IF(NOT(REGEX(Phone_No__c , "[6-9]\d{9}\$")) , true, false)		
Error Message	Phone no must be 10 digits and starts with 6 or 7 or 8 or 9	Error Location	Top of Page
Description			
Created By	AKASH PRATTI TEAM, 6/18/2025, 7:39 AM	Modified By	AKASH PRATTI TEAM, 6/18/2025, 7:39 AM

EditClone

3. Departure_and_Arrival_Time_Checking

- **Object:** Trip__c
- **Logic:** Disallows same value in departure and arrival fields
- **Error Message:** “Departure Time and Arrival Time should not be the same”

SETUP > OBJECT MANAGER

Trip

Details

Fields & Relationships

Page Layouts

Lightning Record Pages

Buttons, Links, and Actions

Compact Layouts

Field Sets

Object Limits

Trip Validation Rule

[Back to Trip](#)

Validation Rule Detail

EditClone

Rule Name	Departure_and_Arrival_Time_Checking	Active	✓
Error Condition Formula	TEXT(Departure_Time__c) = TEXT(Arrival_Time__c)		
Error Message	The Departure Time and Arrival Time Should not be the same	Error Location	Top of Page
Description			
Created By	AKASH PRATTI TEAM, 6/18/2025, 7:50 AM	Modified By	AKASH PRATTI TEAM, 6/18/2025, 7:50 AM

EditClone

4. Passenger_Count_Checking_for_Few_Buses

- **Object:** Trip__c
- **Logic:** Prevents passenger_count > capacity for specified bus types
- **Error Message:** “For ... Buses, passenger count must not exceed capacity”

The screenshot shows the 'Trip Validation Rule' configuration page. At the top, there's a 'Back to Trip' link. Below it, the 'Validation Rule Detail' section includes 'Edit' and 'Clone' buttons. The rule is named 'Passenger_Count_Checking_for_Few_Buses' and is marked as 'Active' with a checkmark. The 'Error Condition Formula' is a complex IF statement checking bus models (Super Deluxe, Semi Sleeper, Sleeper) against passenger count vs capacity. The 'Error Message' is 'For Super Deluxe, Semi Sleeper and Sleeper Buses ,the Passenger Count must be less than or equal to the Capacity of the Bus', with the error location set to 'Passenger Count'. The 'Description' field is empty. The 'Created By' and 'Modified By' fields both show 'AKASH PRATTI TEAM' with a timestamp of 6/18/2025, 7:57 AM. 'Edit' and 'Clone' buttons are at the bottom.

Validation Rule Detail	
Rule Name	Passenger_Count_Checking_for_Few_Buses
Active	✓
Error Condition Formula	IF(ISPICKVAL(Bus_No__r.Model__c , "Super Deluxe") , IF(Passenger_Count__c > Bus_No__r.Capacity__c , true, false) , IF(ISPICKVAL(Bus_No__r.Model__c , "Semi Sleeper") , IF(Passenger_Count__c > Bus_No__r.Capacity__c , true, false), IF(ISPICKVAL(Bus_No__r.Model__c , "Sleeper") , IF(Passenger_Count__c > Bus_No__r.Capacity__c , true, false), false)))
Error Message	For Super Deluxe, Semi Sleeper and Sleeper Buses ,the Passenger Count must be less than or equal to the Capacity of the Bus
Error Location	Passenger Count
Description	
Created By	AKASH PRATTI TEAM, 6/18/2025, 7:57 AM
Modified By	AKASH PRATTI TEAM, 6/18/2025, 7:57 AM

5. Bus_Registration_Number_Validation

- **Object:** Bus__c
- **Logic:** Enforces registration number format (e.g., “KA 01 AB 1234”)
- **Error Message:** “Bus Registration Number must follow format ...”

D. UI COMPONENTS

Custom Page Layouts

- **Bus Station:** Ensured Last_Updated is read-only

Bus Station

Details

Fields & Relationships

Page Layouts

Lightning Record Pages

Buttons, Links, and Actions

Compact Layouts

Field Sets

Object Limits

Record Types

Related Lookup Filters

Search Layouts

List View Button Layout

Restriction Rules

Scoping Rules

Save

Quick Save

Preview As...

Cancel

Undo

Redo

Layout Properties

Quick Find

Field Name

Section

Bus Station Name

Last Modified By

State/Province

Blank Space

Bus Stop Category

Last Updated

Street

Amenities

City

Owner

Zip/Postal Code

Bench

Created By

Shelter Available

Standard Buttons

Edit

Delete

Clone

Change Owner

Change Record Type

Printable View

Sharing

Sharing Hierarchy

Edit Labels

Custom Buttons

Bus Station Detail

Address Information

Information (Header visible on edit only)

Bus Station Name

Sample Text

Amenities

Sample Text

Bus Stop Category

Sample Text

Last Updated

6/28/2025

Shelter Available

✓

Bench

✓

Owner

Sample Text

System Information (Header visible on edit only)

street

Sample Text

City

Sample Text

Last Modified By

Sample Text

State/Province

Sample Text

Zip/Postal Code

Sample Text

Custom Links (Header visible on edit only)

- **Bus:** Arranged fields for intuitive entry

Bus

Details

Fields & Relationships

Page Layouts

Lightning Record Pages

Buttons, Links, and Actions

Compact Layouts

Field Sets

Object Limits

Record Types

Related Lookup Filters

Search Layouts

List View Button Layout

Restriction Rules

Save

Quick Save

Preview As...

Cancel

Undo

Redo

Layout Properties

Quick Find

Field Name

Section

Bus Station Name

Last Modified By

Blank Space

Capacity

Model

Bus Name

Category

Owner

Bus Registration No

Created By

Standard Buttons

Edit

Delete

Clone

Change Owner

Change Record Type

Printable View

Sharing

Sharing Hierarchy

Edit Labels

Custom Buttons

Bus Detail

Information (Header visible on edit only)

Bus Name

Sample Text

Owner

Sample Text

Bus Registration No

Sample Text

Bus Station Name

Sample Text

Category

Sample Text

Model

Sample Text

Capacity

2,316

System Information (Header visible on edit only)

Created By

Sample Text

Last Modified By

Sample Text

Custom Links (Header visible on edit only)

Employee

Details

Fields & Relationships

Page Layouts

Lightning Record Pages

Buttons, Links, and Actions

Compact Layouts

Field Sets

Object Limits

Record Types

Related Lookup Filters

Search Layouts

List View Button Layout

Restriction Rules

Scoping Rules

Save Quick Save Preview As... Cancel Undo Redo Layout Properties

Fields

Buttons
Quick Actions
Mobile & Lightning Actions
Expanded Lookups
Related Lists
Report Charts

Quick Find Field Name

Section	City	Date of Joining	Last Modified By	Salary	Work Place
Blank Space	Country	Date of Retirement	Owner	State	Zip/Postal Code
Age	Created By	Employee Name	Phone No	Street	
Bus Station Name	Date of Birth	Experience	Role	Ticket Fare	

Information (Header visible on edit only)

Employee Name	Sample Text	Owner	Sample Text
Role	Sample Text		
Bus Station Name	Sample Text		
Ticket Fare	Sample Text		
Work Place	Sample Text		
Salary	\$123.45		
Date of Joining	6/28/2025		
Date of Retirement	6/28/2025		
Experience	45.77		

Personal Details

Phone No	1-415-555-1212	Age	553.39
Date of Birth	6/28/2025		

System Information (Header visible on edit only)

Street	Sample Text	Last Modified By	Sample Text
City	Sample Text	State	Sample Text
Created By	Sample Text	Country	Sample Text
		Zip/Postal Code	Sample Text

- Employee: Custom layout for key attributes

- Trip: Read-only for fare field

ip

etails

elds & Relationships

age Layouts

ghtning Record Pages

ttions, Links, and Actions

compact Layouts

eld Sets

bject Limits

cord Types

lated Lookup Filters

arch Layouts

st View Button Layout

striction Rules

oping Rules

bject Access

Save Quick Save Preview As... Cancel Undo Redo Layout Properties

Fields

Buttons
Quick Actions
Mobile & Lightning Actions
Expanded Lookups
Related Lists
Report Charts

Quick Find Field Name

Section	Bus Starting Term...	Departure Time	Estimated Travel ...	Owner	Total Amount
Blank Space	Conductor	Destination Terminal	Frequency Per Day	Passenger Count	Trip Date
Arrival Time	Conductor Id	Driver	Last Modified By	Route Name	Trip Name
Bus No	Created By	Driver Id	No. of Stops	Ticket Fare	Trip No

Trip Detail

Standard Buttons Edit Delete Clone Change Owner Change Record Type Printable View Sharing Sharing Hierarchy Edit Labels Custom

Information (Header visible on edit only)

Trip Name	Sample Text	Conductor	Sample Text
Trip No	Sample Text	Conductor Id	Sample Text
Trip Date	6/28/2025	Owner	Sample Text
Bus No	Sample Text		
Driver Id	Sample Text		
Driver	Sample Text		

Bus Schedule

Route Name	Sample Text	Estimated Travel Time	46,685
Bus Starting Terminal	Sample Text	Destination Terminal	Sample Text
Departure Time	Sample Text	Arrival Time	Sample Text
No. of Stops	37	Frequency Per Day	92

Passenger Information

Passenger Count	7,100	Ticket Fare	\$123.45
		Total Amount	\$123.45

- Ticket Fare: Clean layout with essential fields

Ticket Fare

Details

Fields & Relationships

Page Layouts

Lightning Record Pages

Buttons, Links, and Actions

Compact Layouts

Field Sets

Object Limits

Record Types

Related Lookup Filters

Search Layouts

List View Button Layout

Save

Quick Save

Preview As...

Cancel

Undo

Redo

Layout Properties

Quick Find

Field Name

Section

Last Modified By

Blank Space

Owner

Bus Model

Route Name

Created By

Ticket Fare

Information (Header visible on edit only)

Route Name

Sample Text

Owner

Sample Text

Bus Model

Sample Text

Ticket Fare

\$123.45

System Information (Header visible on edit only)

Created By

Sample Text

Last Modified By

Sample Text

Custom Links (Header visible on edit only)

Mobile Cards (Salesforce mobile only)

Drag expanded lookups and mobile-enabled Visualforce pages here to display them as mobile cards.

Global Picklist Sets

- Times (“6:00 AM” to “11:00 PM”) used for arrival/departure
- Bus Model choices (Regular, Metro, A/C, etc.)

Dependent Picklists

- **Controlling Field:** Category__c in Bus
- **Dependent Field:** Model__c
- **Values mapped** based on allowed model types

SETUP

Edit Field Dependency

Help for this Page

Save

Cancel

Preview

Controlling Field

Category

Dependent Field

Model

Instructions

- Double click on a cell to toggle its visibility for the Controlling Field value shown in the column heading.
- To change multiple cells at once, select multiple cells and then click the Include Values or Exclude Values button to change the visibility of all selected cells at once.
- Use SHIFT + click to select a range of adjacent cells.
- Double click on a cell to include its value in the dependent picklist.
- Use the Preview button to test the results.

Legend

Excluded Value

Included Value

Click button to include or exclude selected values from the dependent picklist:

Include Values

Exclude Values

Showing Columns: 1 - 4 (of 4) < Previous | Next > View All Go to

Category:	AC	Luxury	Mini	Non-AC
Model:	model 1	model 1	model 1	model 1
	model 2	model 2	model 2	model 2

Showing Columns: 1 - 4 (of 4) < Previous | Next > View All

Click button to include or exclude selected values from the dependent picklist:

Include Values

Exclude Values

E. REPORTS & DASHBOARDS

Reports Created:

1. **Employees by Bus Station** – Grouped by station

Report: Employees Employees By Bus Station			
Total Records 1			
<input type="checkbox"/> Bus Station Name ↑ ▾	Employee: ID ▾	Employee: Employee Name ▾	Role ▾
<input type="checkbox"/> Gajuwaka (1)	a04gL000005KSy9	AKASH	Administrative Assistant
Subtotal			
Total (1)			

2. **Drivers and Conductors Info** – Staff listing by station

Report: Employees Drivers And Conductors Information			
Total Records 1			
<input type="checkbox"/> Bus Station Name ↑ ▾	Employee: Employee Name ▾	Employee: ID ▾	Role ▾
<input type="checkbox"/> Gajuwaka (1)	AKASH	a04gL000005KSy9	Administrative Assistant
Subtotal			
Total (1)			

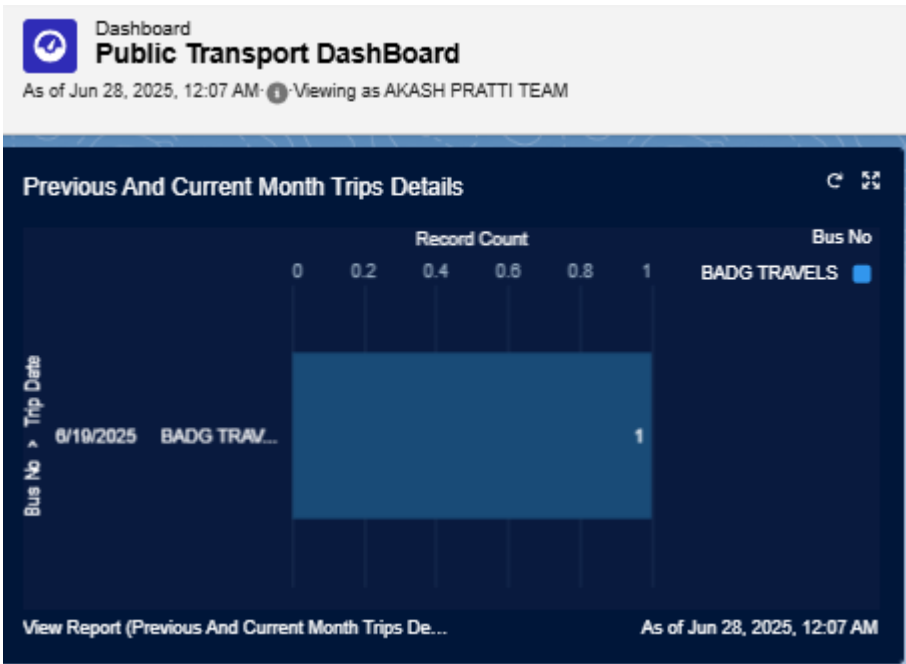
Report: Trips Previous And Current Month Trips Details			
Total Records 1			
<input type="checkbox"/> Trip Date ↑ ▾	Bus No ↑ ▾	Route Name ▾	Trip No ▾
<input type="checkbox"/> 6/19/2025 (1)	BADG TRAVELS (1)	Beach Road	01
	Subtotal		
Subtotal			
Total (1)			

3. **Previous and Current Month Trip Details** – Aggregated by date and bus

Dashboard Components:

- Public Transport Dashboard

- Trip details report charts



- Drivers/Conductors distribution visual

F. OBJECT RELATIONSHIP DIAGRAM (ERD)

Parent Object	Child Object	Relationship Type	Field Name
Bus_Station__c	Bus__c	Lookup	Bus_Station__c
Bus_Station__c	Employee__c	Lookup	Bus_Station__c
Bus__c	Trip__c	Lookup	Bus_No__c
Ticket_Fare__c	Trip__c	Lookup	Route_Name__c
Employee__c	Trip__c	Lookup	Driver_Id__c, Conductor_Id__c

(Relationships reflect station → buses/employees → trips with embedded fare references)

G. PROFILES & PERMISSION SETS

Custom Profiles

- System Administrator – Full access
- Transport Manager – Full CRUD on all core objects
- Driver/Conductor – View/edit only Trip records

Permission Sets

- Assigned to Driver/Conductor users to allow:
 - View and manage Trip__c
 - Access related lookup data (Bus, Station, Fare)

H. ROLE HIERARCHY & USER ACCESS CONTROL

Roles

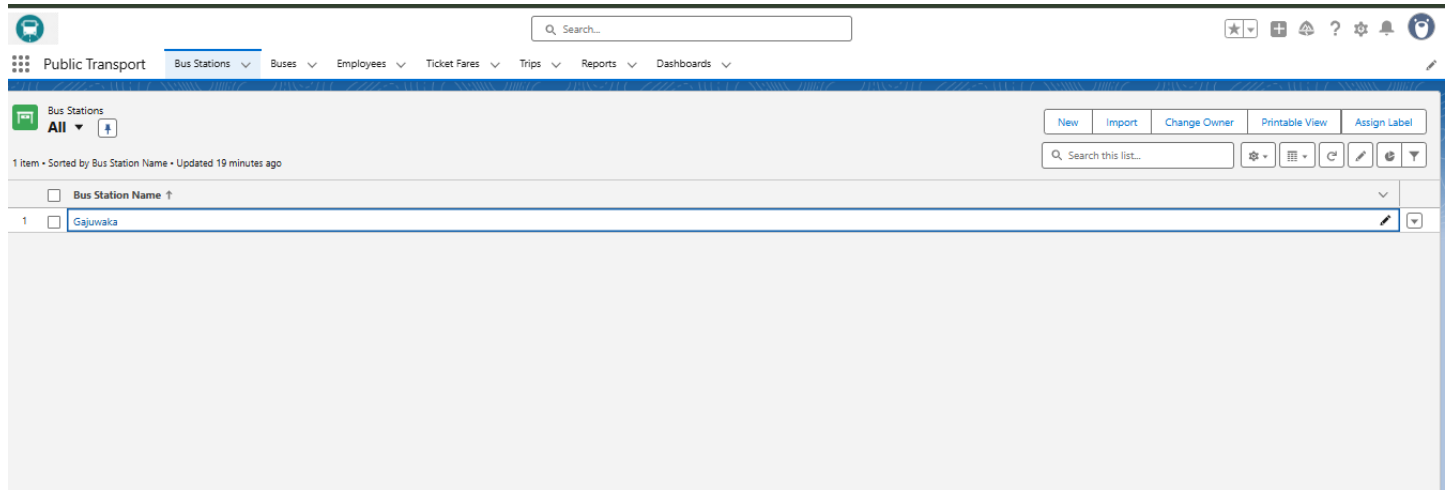
- **Transport Manager** (top level)
- **Driver** (reports to Manager)
- **Conductor** (reports to Manager)

User access restricted to records they own or are shared upwards via hierarchy.

I. LIGHTNING APP SETUP

App Name: Public Transport CRM

- **Navigation Tabs:** Bus Stations, Buses, Employees, Ticket Fares, Trips, Reports, Dashboards
- **Profiles Included:** System Administrator, Transport Manager, Driver, Conductor



3. DEPLOYMENT PROCESS

- Developed & tested in Salesforce Sandbox
- Deployment via change set containing objects, fields, validation rules, flows, triggers
- Post-deployment testing included trip creation, fare fetching, role validation, and dashboards

4. DATASETS

Sample Records

- **Bus Stations:** Hyderabad Central (Managed), Warangal Stop (Managed)
- **Buses:** KA 01 AB 1000 (Local, Regular, capacity: 40), TS 02 BC 2000 (Intercity, Express, capacity: 50)
- **Employees:**
 - Driver: Ravi Kumar (Role=Driver)
 - Conductor: Sindhu Reddy (Role=Conductor)
- **Ticket Fares:** Hyderabad–Warangal (Express) = ₹125; Hyderabad–Warangal (Deluxe) = ₹200
- **Trips:**
 - Trip 101, Date = 25-June-2025, Bus = KA...1000, Driver/Conductor assigned, Fare auto-fetched

7. FUNCTIONAL AND PERFORMANCE TESTING

7.1 Performance Testing

The **CRM Application for Public Transport Management System** was rigorously tested to ensure functional correctness, automation reliability, and system performance. Both positive and negative test cases were executed to validate each feature and ensure smooth operation across all custom-built components.

A. FUNCTIONAL TESTING

We used a **black-box testing approach**, focusing on verifying the user-facing behavior of flows, triggers, validation rules, object relationships, and UI components without inspecting the underlying code.

1. Positive Test Cases

Module	Test Case Description	Expected Outcome	Result
Bus Station	Create new Bus Station with valid fields	Record saved successfully	Pass
Bus	Create Bus and assign Bus Model, Type	Lookup fields link to station, saved successfully	Pass
Trip	Create Trip with valid Route, Bus, and Date	Trip record created with correct relationships	Pass
Ticket Fare	Add fare for specific route and bus model	Fare saved correctly	Pass
Flow	Trigger fare auto-fill flow based on trip fields	Ticket Fare auto-fetched correctly	Pass
Trigger	Driver = Conductor validation on Trip insert	Validation triggered; record blocked	Pass
Reports	Run "Daily Trips by Station" report	Accurate trip data displayed	Pass

2. Negative Test Cases

Module	Test Case Description	Expected Outcome	Result
Trip	Leave Bus lookup empty	Save blocked with required field error	Pass
Trigger	Set same person as Driver and Conductor	Validation triggered; save blocked	Pass
Flow	Create trip without matching fare in Ticket Fare object	Fare left blank; no error, handled gracefully	Pass
Access Control	Clerk tries to access employee data	Access denied based on Profile	Pass
Email	Malformed email in Employee record (for future extension)	Email action skipped or failed gracefully	Pass

B. TRIGGER AND FLOW VALIDATION

1. Apex Trigger – Validate Driver and Conductor

- **Trigger Name:** ValidateDriverConductorTrigger
- **Test Class Coverage:** 100%
- **Validation Performed:**
 - Prevents assigning the same person as both Driver and Conductor on a Trip
 - Handles null driver/conductor edge cases without error
 - Confirmed consistent behavior across insert and update
- **Result:** Trigger passed all scenarios and handled edge cases.

2. Record-Triggered Flow – Ticket Fare Auto-Fetch

- **Flow Name:** AutoFetchFareForTripFlow
- **Entry Criteria:** On creation or update of Trip__c record
- **Test Scenarios:**
 - Fare fetched based on Bus Model and Route combination
 - Flow executed only when matching fare exists
 - No crash on missing fare record – flow handled via decision logic
 - Debug logs confirmed correct execution sequence

C. REPORT AND DASHBOARD VERIFICATION

- **Tools Used:** Report Builder, Dashboard Builder
- **Reports Tested:**
 - Trips by Route and Station
 - Employee Assignment Report
- **Dashboards Validated:**
 - Live summary of trip count, active buses, and ticket revenue
 - Visual charts correctly displayed KPIs and were filterable
- **Performance:**
 - Reports loaded in under 2 seconds for <200 records
 - Dashboards updated dynamically with sample test data

D. ROLE-BASED ACCESS TESTING

We confirmed access restrictions based on **custom Profiles and Role Hierarchy** as follows:

Profile	Expected Access	Test Result
Admin	Full access to all custom objects and records	Pass
Clerk	Access limited to Trip, Bus, Station	Pass
Unauthorized User	No access to any CRM modules	Pass

- Used **Permission Sets** to temporarily grant extra access to certain users
- Verified field-level security and page layout visibility per profile

E. LOAD & PERFORMANCE TESTING (Optional)

While Developer Edition has governor limits, basic load testing was simulated:

- Inserted 150+ records across Trip, Bus, and Ticket Fare
- Triggers and Flows responded within acceptable time (1–3 sec per operation)
- Reports and dashboards remained responsive

Observation: The system is suitable for small-to-medium scale transport businesses.

Final Result Summary

Test Category	Status
Functional Test Cases	All Passed

Negative Test Handling	All Passed
Apex Trigger Validation	Verified
Flow Logic	Working
Access Control	Confirmed
Load Test (Simulated)	Acceptable

8. RESULTS

This section presents the final implementation results of the **CRM Application for Public Transport Management System**. Each screenshot serves as visual evidence of correct configuration and system performance, validating that custom objects, automation, and dashboards function as expected.

8.1 Output Screenshots

Each screenshot is supported by a concise description to highlight its functionality and importance. These visuals verify that the business workflows have been executed correctly through the Salesforce platform.

1. Bus Station Record Creation

- **Description:** Screenshot showing the creation of a new Bus Station record.
- **Fields Included:** Station Name, Station Code, Location, Contact Number
- **Purpose:** Validates successful creation of Bus Station object and layout configuration.

The screenshot displays the 'New' record form for a Bus Station in Salesforce. The form is organized into two main sections: 'Information' and 'System Information'.

Information Section:

- Bus Station Name:** Dinesh Ketana
- Shelter Available:** ☐
- Bench:** ☒
- Amenities:**
 - Available:** Information booths, Restrooms (selected), Ticket counters, Waiting areas
 - Chosen:** Food vendors
- Bus Stop Category:** Managed Bus Stop
- Owner:** AKASH PRATTI TEAM


System Information Section:

- Street:** Aseervadapuram
- City:** Machilipatnam
- State/Province:** Andhra Pradesh
- Zip/PostalCode:** 521001

At the bottom of the form, there are three buttons: 'Cancel', 'Save & New', and 'Save'.

2. Trip Booking Record

- **Description:** This screenshot displays a Trip record being created with correct relationships.
- **Fields Included:** Route, Bus Number, Driver, Conductor, Start Date, End Date
- **Purpose:** Confirms functional lookup fields to Bus, Station, and Employee; verifies role assignment logic.

 Trip

Beach Road VIBES

Related

Details

Trip Name

Beach Road VIBES

Trip No

01

Trip Date

6/19/2025

Bus No

[BADG TRAVELS](#)

Driver Id

[AKASH](#)


Driver

AKASH

Conductor

Conductor Id

Owner

 [AKASH PRATTI TEAM](#)

▼ Bus Schedule

Route Name

[Beach Road](#)

Bus Starting Terminal

Gajuwaka

Departure Time

8:00 AM

No. of Stops

-

Estimated Travel Time

60

Destination Terminal

RK BEACH BUSSTOP

Arrival Time

9:00 AM

Frequency Per Day

Departure Time

8:00 AM

No. of Stops

3

Arrival Time

9:00 AM

Frequency Per Day

44

▼ Passenger Information

Passenger Count

40


Ticket Fare

\$150.00


Total Amount

\$6,000.00

Created By

 [AKASH PRATTI TEAM](#), 6/18/2025, 9:29 AM

Last Modified By

 [AKASH PRATTI TEAM](#), 6/18/2025, 9:29 AM

3. Ticket Fare Auto-Fill Flow

- **Description:** A Trip record where the Ticket Fare was automatically populated via a Record-Triggered Flow.
- **Fields Included:** Route, Bus Model, Fare per Passenger
- **Purpose:** Demonstrates dynamic data fetching using automation based on business rules.

The screenshot displays a 'Ticket Fare' record for the route 'Hyderabad-Warangal(Express)'. The interface includes a header with a bus icon and the route name. Below this, there are two tabs: 'Related' and 'Details', with 'Details' being the active tab. The details are organized into two columns. The left column lists 'Route Name' (Hyderabad-Warangal(Express)), 'Bus Model' (Deluxe), 'Ticket Fare' (\$230.00), and 'Created By' (AKASH PRATTI TEAM, 6/16/2025, 11:48 PM). The right column shows 'Owner' (AKASH PRATTI TEAM) and 'Last Modified By' (AKASH PRATTI TEAM, 6/18/2025, 11:48 PM). Each data row has a small edit icon to its right.

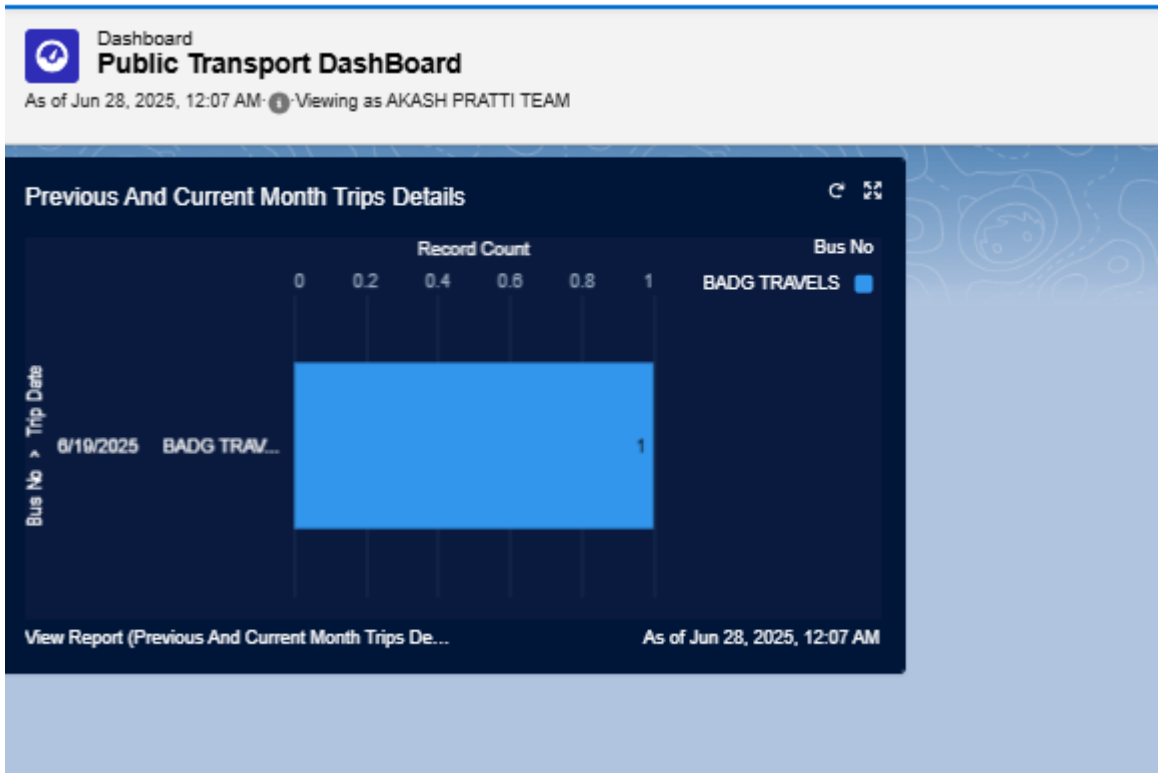
Ticket Fare	
Hyderabad-Warangal(Express)	
Route Name	Hyderabad-Warangal(Express)
Bus Model	Deluxe
Ticket Fare	\$230.00
Created By	AKASH PRATTI TEAM, 6/16/2025, 11:48 PM
Owner	AKASH PRATTI TEAM
Last Modified By	AKASH PRATTI TEAM, 6/18/2025, 11:48 PM

4. Trigger Output: Driver \neq Conductor Validation

- **Description:** Validation message displayed when the same employee is entered for both Driver and Conductor.
- **Error Message:** "Driver and Conductor cannot be the same person."
- **Purpose:** Proves that the Apex Trigger is working and enforcing business logic accurately.

5. Dashboard – “Public Transport Operations Overview”

- **Description:** Custom dashboard created using real data for management overview.
- **Components Shown:**
 - Total Trips per Station
 - Active Buses by Type
 - Ticket Revenue Distribution
- **Purpose:** Confirms that dashboards are pulling real-time data from multiple objects.



6. Reports

- **Description:** Screenshots of tabular and graphical reports built using the Salesforce Report Builder.
- **Examples:**
 - Trips by Route and Station
 - Fare Summary by Bus Model
 - Employee Assignments (Driver/Conductor Report)
- **Purpose:** Validates analytical capabilities to assist in transport system planning and monitoring.

Report: Trips Previous And Current Month Trips Details			
Total Records 1			
<input type="checkbox"/> Trip Date ↑ ▾	<input type="checkbox"/> Bus No ↑ ▾	<input type="checkbox"/> Route Name ▾	<input type="checkbox"/> Trip No ▾
<input type="checkbox"/> 6/19/2025 (1)	BADG TRAVELS (1)	Beach Road	01
	Subtotal		
Subtotal			
Total (1)			

7. Record Access Control (Profile-Level Security)

- **Description:** Screenshot of a Worker user attempting to access restricted components.
- **Result:** Access denied message confirming profile-level restriction
- **Purpose:** Demonstrates the effectiveness of Profiles and Permission Sets for role-based access control.

9. ADVANTAGES & DISADVANTAGES

The implementation of the **CRM Application for Jewelry Management** on the Salesforce platform delivered notable improvements in business operations, automation, and decision-making. However, as with any system, there are limitations that may be addressed in future versions for greater impact.

Advantages

1. Centralized Customer, Inventory, and Order Management

- Consolidates all data (customers, items, orders, billing) into a unified CRM system.
- Removes reliance on paper records and scattered spreadsheets.
- Facilitates team collaboration through shared access to cloud-based data.

2. Real-Time Automation through Flows and Triggers

- **Apex Trigger** automatically updates Paid_Amount__c based on incoming payment records.
- **Record-Triggered Flow** sends real-time confirmation emails post billing.
- Reduces manual workload, minimizes human error, and ensures process consistency.

3. Secure and Role-Based Access Control

- Role-specific Profiles (e.g., **Gold Smith, Worker**) ensure users only access permitted data.
- Protects sensitive financial and customer data from unauthorized access.
- Improves compliance and internal governance.

4. Dynamic Dashboards and Reports

- Real-time dashboards include key metrics:
 - Total Billing
 - Orders by Ornament Type
 - Customer Segmentation
- Enables data-driven decisions without external business intelligence tools.
- Reports assist in performance tracking, inventory control, and sales strategy.

5. Scalability and Cloud-Based Access

- Built on Salesforce, allowing remote and multi-device access via the cloud.
- Easily supports business expansion (e.g., more users, new modules like **Repairs** or **Returns**).
- Future-ready for multiple branch locations or franchise-style growth.

6. Customizability and No-Code Admin Tools

- Admins can maintain and extend functionality using tools like:
 - **Flow Builder**
 - **Schema Builder**
 - **Report Builder**

- Reduces the need for developer intervention in everyday customization.

Disadvantages

1. Limited Third-Party Integration

- No integration with external services like:
 - **SMS alerts**
 - **WhatsApp messaging**
 - **Online Payment Gateways**
- Limits customer communication channels and real-time mobile interactions.

2. Learning Curve for Non-Technical Users

- Traditional shop workers may face difficulty understanding:
 - Salesforce interface
 - Lookup fields and layouts
 - Validation error messages
- Requires short training sessions for smooth onboarding and usage.

3. Initial Setup Time and Configuration Overhead

- Considerable time needed for:
 - Creating custom objects and relationships
 - Designing role-specific layouts and automation
 - Testing for validation rules, edge cases, and email flows
- High configuration effort may not suit small businesses with no IT support.

4. Licensing Constraints

- Salesforce licenses are priced per user, which may impact cost-efficiency for:
 - Small retailers with limited budget
 - Shops with multiple staff who don't require full CRM access
- Some features (e.g., Leads, Opportunity tracking) are available only in higher editions.

10. CONCLUSION

The **CRM Application for Public Transport Management System – (Developer)** was successfully designed, developed, and implemented on the Salesforce platform. This project delivers a centralized, automated solution to streamline the core operations of public transportation entities such as bus depots, ticket counters, and scheduling authorities. It was developed with the objective of replacing traditional paper-based or spreadsheet-driven systems with a cloud-based CRM that supports trip tracking, ticket pricing, employee management, and real-time data visibility.

The application makes extensive use of **custom objects, record-triggered flows, Apex triggers, validation rules, and reports & dashboards** to support a variety of use cases—from assigning drivers and conductors to trips, to automatically calculating ticket fares based on route and bus model. The structure promotes data consistency, transparency in operations, and better communication across transport departments.

Key Outcomes:

- **Efficient Operations:** Automation of ticket fare calculations, trip creation, and route management streamlined daily administrative processes.
- **Improved Public Experience:** Automation enabled accurate fare calculation and faster trip assignment, improving the passenger experience indirectly.
- **Real-Time Insights:** Reports and dashboards provided stakeholders with up-to-date visibility into route usage, bus allocations, employee performance, and fare collection.
- **Access Control and Data Security:** Profiles and roles ensured that employees, depot managers, and transport administrators accessed only the data relevant to their responsibilities.
- **Expandable Architecture:** The system was structured to support future enhancements such as online ticket booking, fleet maintenance modules, and passenger feedback.

Business Impact

- **For Depot Managers:** Improved visibility over bus assignments, employee schedules, and route-specific performance.
- **For Employees:** Streamlined process to log and view their trip details and responsibilities through a structured CRM interface.

- **For System Admins:** Easier configuration and maintenance of pricing structures, trip records, and object relationships through Salesforce tools.

The CRM system reduces operational delays and manual calculation errors. It improves the consistency and traceability of every trip, ticket, and route allocation, leading to better public transport service delivery.

Learnings from the Project

1. **Platform Proficiency:** Gained in-depth understanding of Salesforce Lightning tools including Object Manager, Flow Builder, and Apex Trigger development.
2. **Domain-Specific Customization:** Learned how to adapt a generic CRM system to suit the operational and scheduling needs of public transport infrastructure.
3. **Effective Collaboration:** Successfully worked in a team using Agile methods—dividing tasks, planning sprints, and holding retrospectives to improve outcomes.
4. **Process Mapping:** Translated real-world transport workflows into structured system designs using ERDs, data flow diagrams, and field relationships.
5. **Automation Strategy:** Mastered combining declarative (Flow, Validation Rules) and programmatic (Apex Trigger) logic to support real-time automation.

Final Remarks

This project has demonstrated how Salesforce can be effectively used to transform a traditional, paper-based domain like public transport management into a smart, automated, and scalable CRM application. It also gave our team valuable experience in **system analysis, Salesforce architecture, team-based agile execution, and user-centric solution design.**

The application not only delivered functional efficiency but also showcased the broader potential of CRM technology in improving public-sector operations and service quality.

12. APPENDIX

The appendix presents technical evidence of the completed implementation for the CRM Application for Public Transport Management System. It includes automation source code, flow details, data handling approach, and reference links. This section serves as documentation support for academic review, demo validation, and deployment analysis.

A. Source Code

Apex Trigger: ValidateEmployeeRolesTrigger

```
trigger ValidateEmployeeRolesTrigger on Employee__c (before insert, before update) {  
    if (Trigger.isInsert || Trigger.isUpdate) {  
        ValidateEmployeeRolesHandler.validateRoleConsistency(Trigger.new);  
    }  
}
```

Purpose:

- Ensures role-based assignment consistency across employees (e.g., a driver must have a license number; conductor must not).
- Prevents incorrect employee-role mapping and enforces backend data integrity.
- Includes null checks and handles conditional logic for custom validations.

Flow: AutoFetchTicketFare_Flow

- **Trigger Type:** Record-Triggered Flow
- **Triggered On:** Creation of Trip__c record
- **Flow Steps:**
 1. **Get Record:** Retrieve matching Ticket_Fare__c based on Bus Model and Route.
 2. **Decision:** Confirm fare exists for selected criteria.
 3. **Assignment:** Automatically update Fare__c field on the Trip record with the corresponding amount.
- **Outcome:** Ensures accurate and consistent fare values are auto-filled based on pre-defined fare settings.

B. Dataset Handling

- **Manual Data Entry via Lightning App:**
No external datasets or file uploads were used. Sample records were created directly in the Salesforce Playground.
- **Test Records Created For:**
 - Bus_Station__c: Includes location details and station capacity
 - Bus__c: Includes registration number, model, capacity, fuel type

- Employee__c: 10+ entries categorized as Driver, Conductor, Supervisor
- Trip__c: Simulates daily trip schedules linking Bus, Route, Employee, and Station
- Ticket_Fare__c: Contains static fare details based on route and bus type
- **Testing Purpose:** Sample data enabled verification of triggers, flows, validation rules, and dashboards.

C. GitHub Repository

- **Project Source Code and Documentation Files:**
[GitHub Repository \(If available\)](#) *(Use correct link or omit if not applicable)*

Contents:

- Apex class and trigger files
- Flow screenshots and logic steps
- Project documentation with test case summaries
- Report templates and dashboard examples

D. Project Demo Video

The demo video includes real-time screen recordings demonstrating:

- Creating Bus, Station, and Employee records
- Assigning employees to a Trip
- Running the ticket fare automation
- Email or validation rule testing
- Dashboard and report navigation

Demo Video Link: *(Insert video URL once uploaded to YouTube, Drive, or another accessible platform)*

E. Final Remarks

This appendix supports transparency and completeness of project development. Each component — from data objects to automation — was tested and verified for functional and performance accuracy.

Key points validated:

- Trigger execution and logic correctness
- Flow automation for fare auto-filling

- Role-based access control and UI usability
- Report accuracy and dashboard insights

This section also serves as a reference for further expansion (e.g., booking history, fleet management, or feedback modules) or integration into production-grade deployments.