Fleet Management Dashboard - Software Design Document (SDD)

1. Introduction

Purpose:
This document describes the design, architecture, and functionality of the Fleet Management Dashboard system. It enables fleet operators to monitor vehicles, track analytics, manage maintenance, and optimize operations.

Scope:
The system provides real-time tracking, fleet analytics, maintenance management, driver management, and revenue monitoring.

2. System Overview

The Fleet Management Dashboard is a web-based platform that provides real-time visualization and analytics of fleet operations.

Main Features:
• Real-time vehicle tracking
• Fleet analytics dashboard
• Driver and vehicle management
• Maintenance tracking
• Revenue and performance analysis

3. System Architecture

Architecture Type: Three-tier architecture

Layers:
• Presentation Layer (Frontend)
• Application Layer (Backend)
• Data Layer (Database)

Flow:

User → Frontend → Backend API → Database → GPS Devices

4. User Interface Components

Main UI Components:

Sidebar Navigation:
• Dashboard
• Vehicles
• Drivers
• Maintenance
• Reports
• Settings

Dashboard Widgets:
• Fleet summary (total, active, idle vehicles)
• Real-time map tracking
• Maintenance cost analytics
• Revenue analytics
• Vehicle status table

5. Functional Requirements

Authentication:
• User login/logout
• Role-based access control

Vehicle Management:
• Add, edit, delete vehicles
• View vehicle status

Driver Management:
• Add and assign drivers

GPS Tracking:
• Real-time vehicle location tracking
• Route history visualization

Maintenance Management:

• Schedule and track maintenance

Reports:
• Generate fleet, revenue, and maintenance reports

6. Non-Functional Requirements

Performance:
• Supports 1000+ vehicles
• Real-time updates within seconds

Security:
• HTTPS encryption
• Secure authentication

Usability:
• Responsive and user-friendly interface

Reliability:
• 99.9% uptime

7. Database Design

Main Tables:

Users:
• user_id, name, email, password, role

Vehicles:
• vehicle_id, vehicle_number, model, status, driver_id

Drivers:
• driver_id, name, license_number, contact

GPS_Data:
• gps_id, vehicle_id, latitude, longitude, timestamp

Maintenance:
• maintenance_id, vehicle_id, date, cost, description

Revenue:
• revenue_id, vehicle_id, amount, date

8. API Design

Authentication:
POST /api/login
POST /api/logout

Vehicles:
GET /api/vehicles
POST /api/vehicles
PUT /api/vehicles/{id}
DELETE /api/vehicles/{id}

GPS:
GET /api/gps/{vehicle_id}
POST /api/gps/update

Maintenance:
GET /api/maintenance
POST /api/maintenance

9. Technology Stack

Frontend:
• React.js
• Tailwind CSS
• Chart.js

Backend:
• Node.js
• Express.js

Database:
• PostgreSQL / MongoDB

Other:

• Google Maps API
• Cloud Hosting (AWS, Azure, GCP)

10. Deployment Architecture

Client Browser → Frontend Server → Backend Server → Database Server

11. Future Enhancements

• Predictive maintenance using AI
• Fuel optimization
• Driver behavior analytics
• Mobile app support
• IoT integration

12. Conclusion

This system provides an efficient solution for fleet tracking, analytics, and management. It improves operational efficiency, reduces maintenance costs, and enhances decision-making.