Vendor Management System - Design Document

Prepared by: Anushka Dwivedi

Institution: Indian Institute of Information Technology, Allahabad

Date: 10/11/2025

Table of Contents

Table of Contents

1. Introduction

The Vendor Management System (VMS) is designed to streamline and organize vendor, driver, and cab onboarding processes within a hierarchical framework. It ensures efficient communication between super vendors, sub-vendors, and operational entities through a unified web-based interface. This system replaces traditional manual processes with a digital, centralized platform to improve visibility, accountability, and data-driven decision-making.

1.1 Objectives

- To provide a scalable and secure solution for vendor onboarding and management.
- To establish clear vendor hierarchies (SuperVendor, RegionalVendor, CityVendor, LocalVendor).
- To enable efficient vehicle and driver management under each vendor.
- To ensure role-based access and data visibility.
- To integrate easily with existing enterprise or logistics tools.

1.2 Scope

The system encompasses vendor registration, approval workflows, sub-vendor linkage, and operational hierarchy management. It provides dashboards for monitoring and control across levels, integrating vehicle and driver onboarding modules. Future extensions include automated reporting, analytics, and integration with transport management systems.

1.3 Technologies Used

- Node.js – for building scalable backend services and RESTful APIs.
- Express.js – chosen for its minimalistic and flexible middleware framework.
- MongoDB Atlas – preferred for its cloud-native architecture, scalability, and document-based data modeling.
- HTML/CSS – for building responsive, simple frontend interfaces.
- GitHub – for version control and collaboration.

2. System Analysis

2.1 Existing System

Traditional vendor management relies on manual spreadsheets, email communications, and siloed record-keeping. This leads to inconsistencies, lack of accountability, and difficulty in tracking hierarchical relationships between vendors.

2.2 Proposed System

The proposed system introduces automation, structured workflows, and role-based controls. By maintaining all data in a cloud-based repository, it allows quick access, data validation, and vendor performance insights. Super vendors can monitor their network, approve sub-vendors, and manage fleets seamlessly.

3. System Design

3.1 Architecture Overview

The system follows a three-tier architecture comprising Presentation, Application, and Database layers.
[Space for System Architecture Diagram]

3.2 Module Design

1. Vendor Management – Handles vendor profiles, approvals, and hierarchy linkage.
2. Sub-Vendor Onboarding – Allows creation of subordinate vendors under parent vendors.
3. Vehicle & Driver Management – Tracks assigned vehicles and drivers for each vendor.
4. Authentication & Roles – Ensures secure login, password hashing, and JWT-based session management.
5. Permission Control – Manages access to operations based on vendor roles.

3.3 Database Design

[Space for ER Diagram]

The database is implemented using MongoDB Atlas, enabling horizontal scalability and flexible schema evolution. Each collection represents entities like Vendor, Vehicle, and Driver, with embedded or referenced relationships for hierarchy representation.

Example – Vendor Schema:

```
{
  name: String,
  username: String,
  password: String,
  contactInfo: String,
  role: [SuperVendor, RegionalVendor, CityVendor, LocalVendor],
  region: String,
  parentVendorId: ObjectId (ref: 'Vendor'),
  isActive: Boolean,
  permissions: {
    fleetOnboarding: Boolean,
    subVendorCreation: Boolean
  }
}
```

3.4 API Design

The API layer is implemented using Express.js to handle vendor and sub-vendor operations securely.

Endpoint	Method	Description
/api/vendors	POST	Create a new vendor
/api/vendors/:id	GET	Fetch vendor details by ID
/api/vendors/:id	PUT	Update vendor details
/api/vendors/:id	DELETE	Delete vendor (admin access)

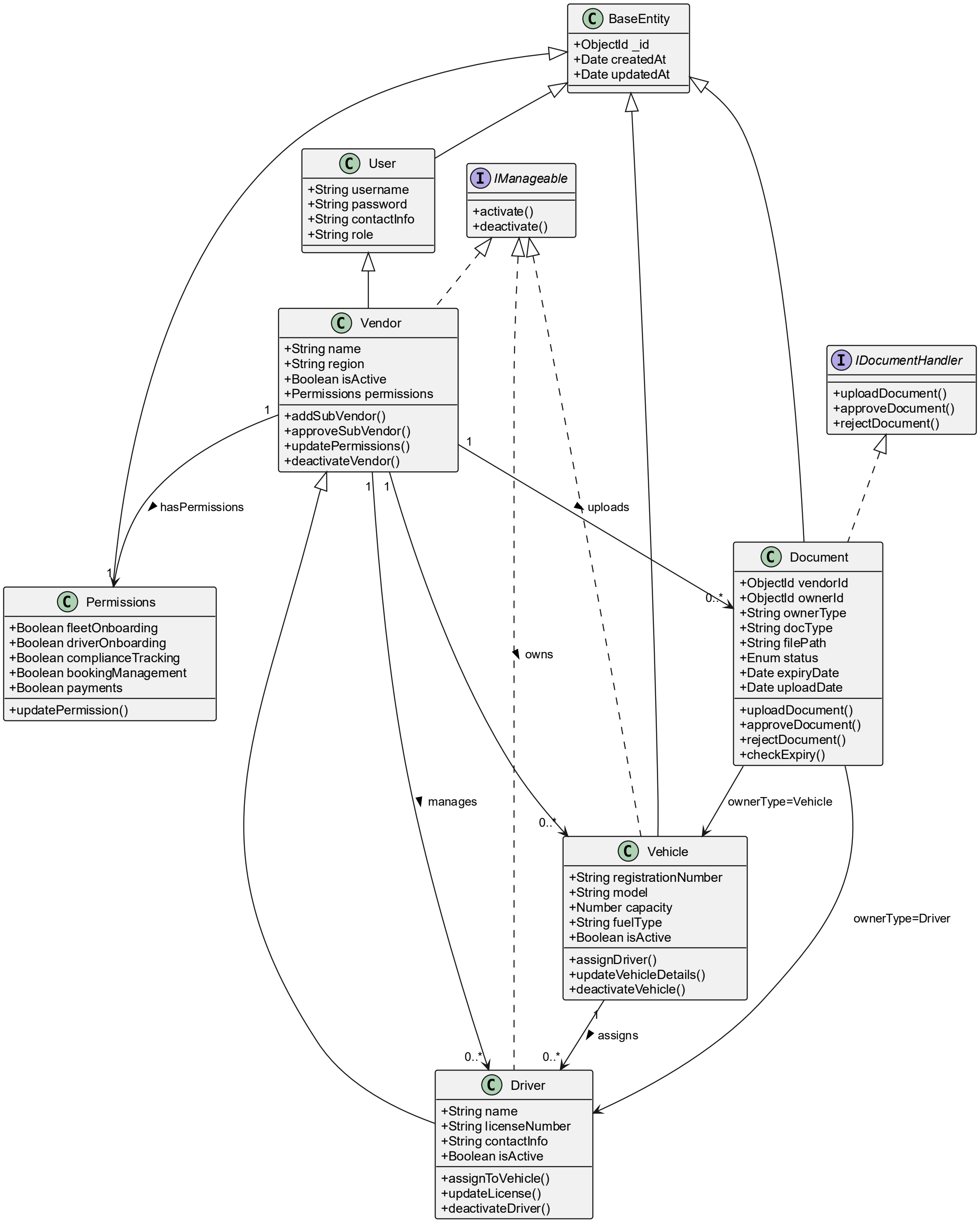
3.5 Design Decisions

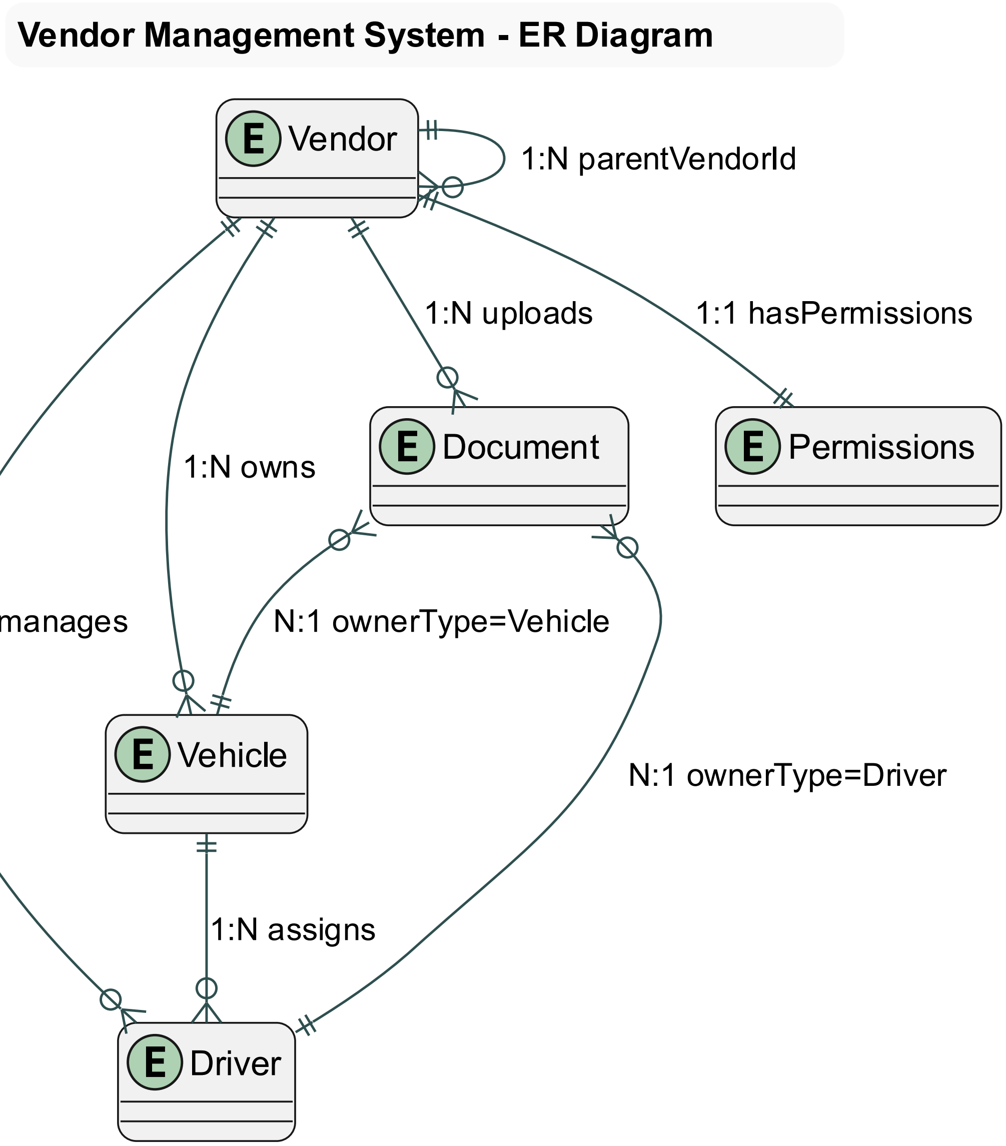
- **MongoDB Atlas** was chosen over SQL databases due to its flexible document structure, better suited for hierarchical vendor relationships.
- **Node.js with Express** was selected for its non-blocking I/O and modular middleware support, ensuring scalable request handling.
- **RESTful APIs** ensure interoperability with future mobile or partner integrations.
- Security design includes password hashing (bcrypt), JWT-based authentication, and validation middleware.

4. Implementation Details

The implementation follows modular coding practices with separate layers for routing, controllers, and models. GitHub is used for version control, enabling easy collaboration and CI/CD integration.

Vendor Management System - UML Class Diagram





5. Testing

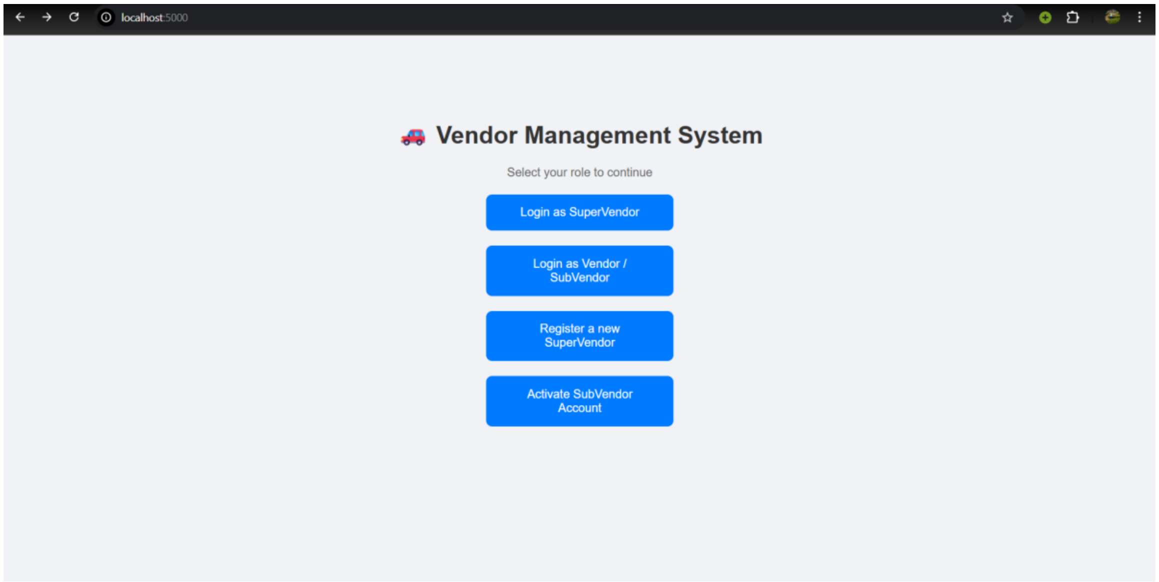
Testing involved unit testing of API endpoints, integration testing across modules, and manual verification for UI workflows.

Sample Test Cases:

Test Case ID	Description	Expected Output	Result
TC001	Create Super Vendor	Vendor added successfully	Pass
TC002	Add Sub Vendor	Sub vendor linked	Pass

		correctly	
TC003	Access Control Check	Unauthorized access blocked	Pass

6. Results and Screenshots



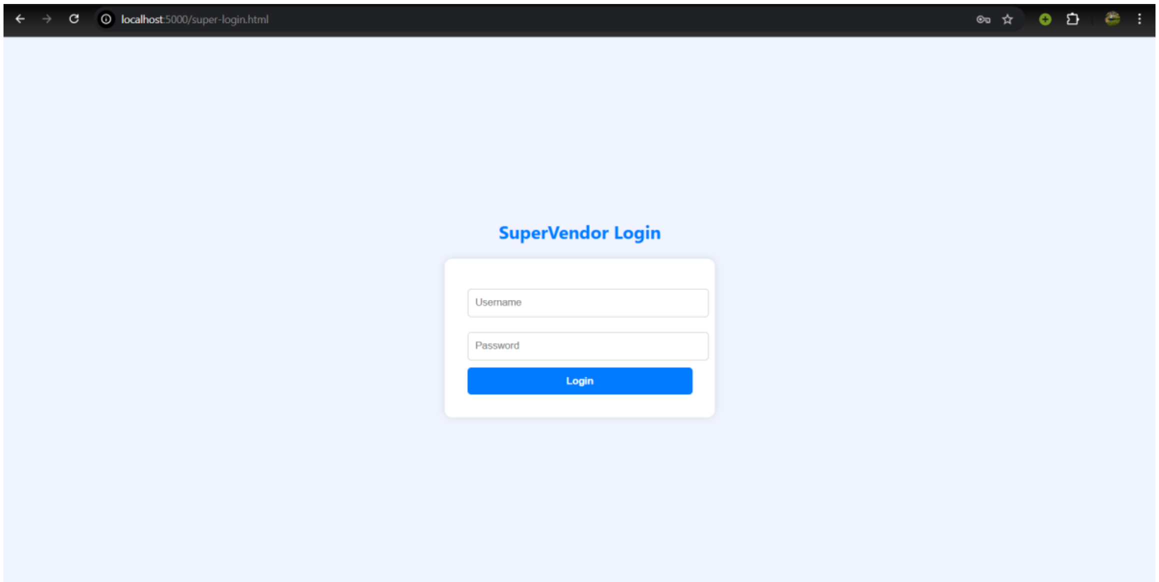
localhost:5000

🚗 Vendor Management System

Select your role to continue

Login as SuperVendor
Login as Vendor / SubVendor
Register a new SuperVendor
Activate SubVendor Account

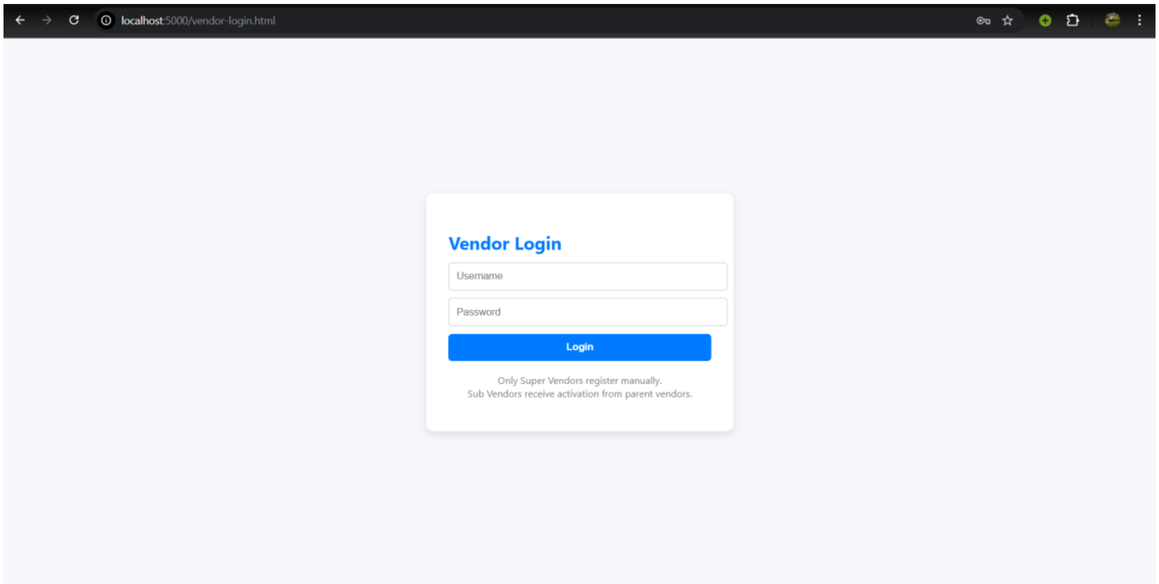
Main Dashboard of Vendor Management System



localhost:5000/super-login.html

SuperVendor Login

Super Vendor Login Page

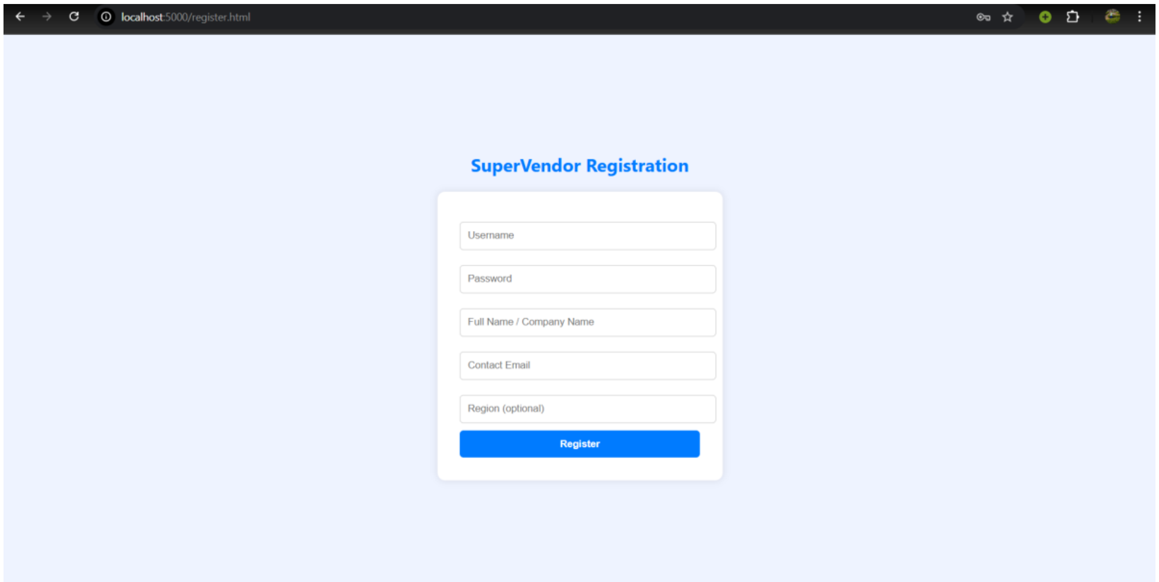


localhost:5000/vendor-login.html

Vendor Login

Only Super Vendors register manually.
Sub Vendors receive activation from parent vendors.

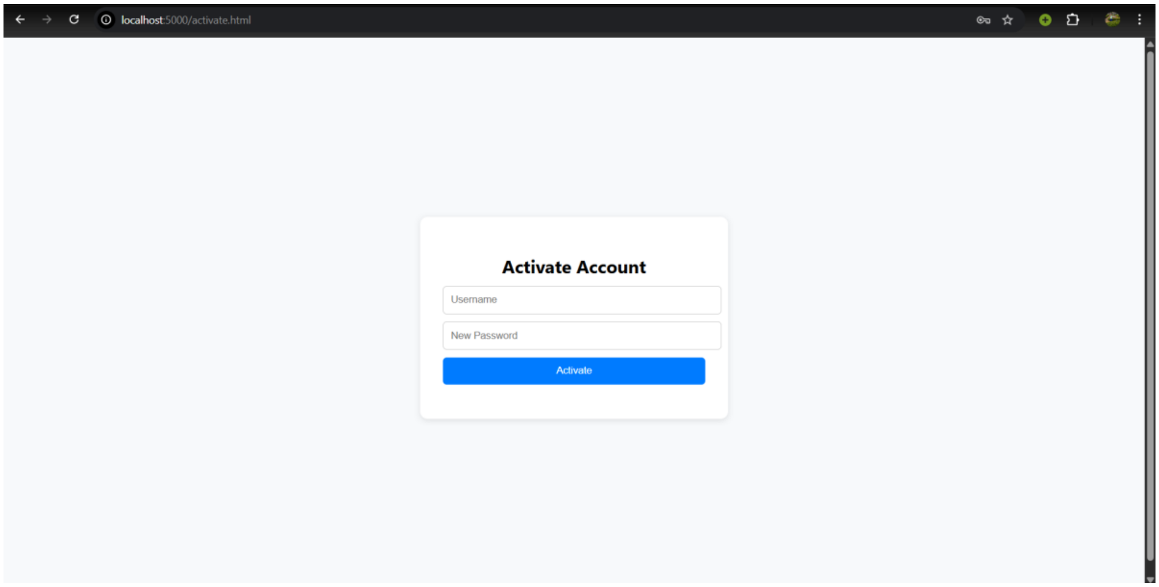
Vendor Login Page



localhost:5000/register.html

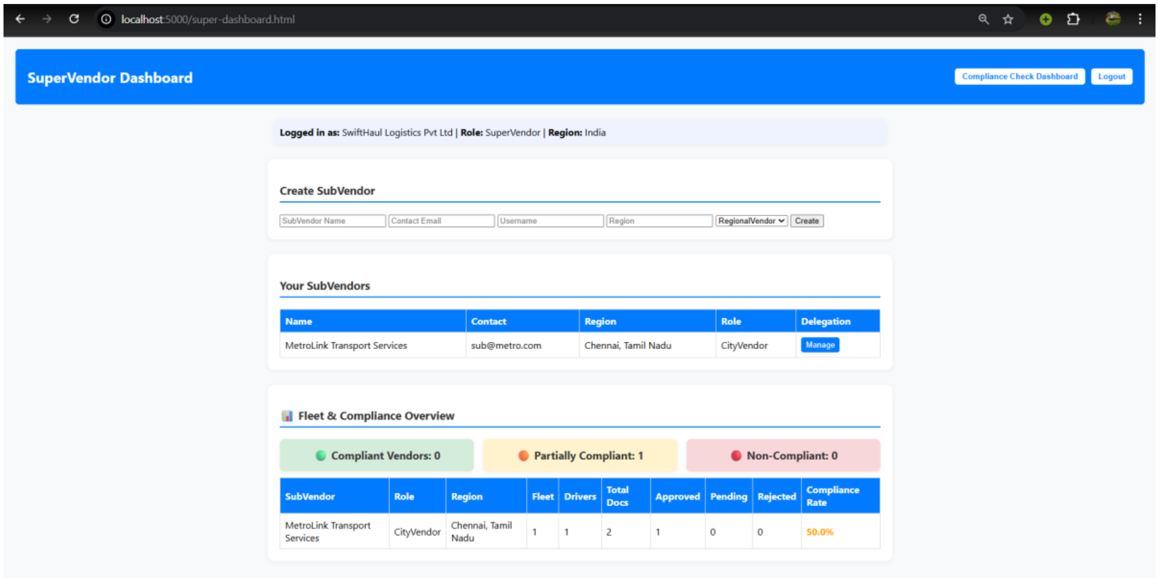
SuperVendor Registration

Super Vendor Registration Page



Activate Account

Activate Account page for the sub vendors created by the super vendor


localhost:5000/super-dashboard.html
SuperVendor Dashboard
Compliance Check Dashboard
Logout

Logged in as: SwiftHaul Logistics Pvt Ltd | **Role:** SuperVendor | **Region:** India

Create SubVendor

Your SubVendors

Name	Contact	Region	Role	Delegation
MetroLink Transport Services	sub@metro.com	Chennai, Tamil Nadu	CityVendor	Manage

Fleet & Compliance Overview

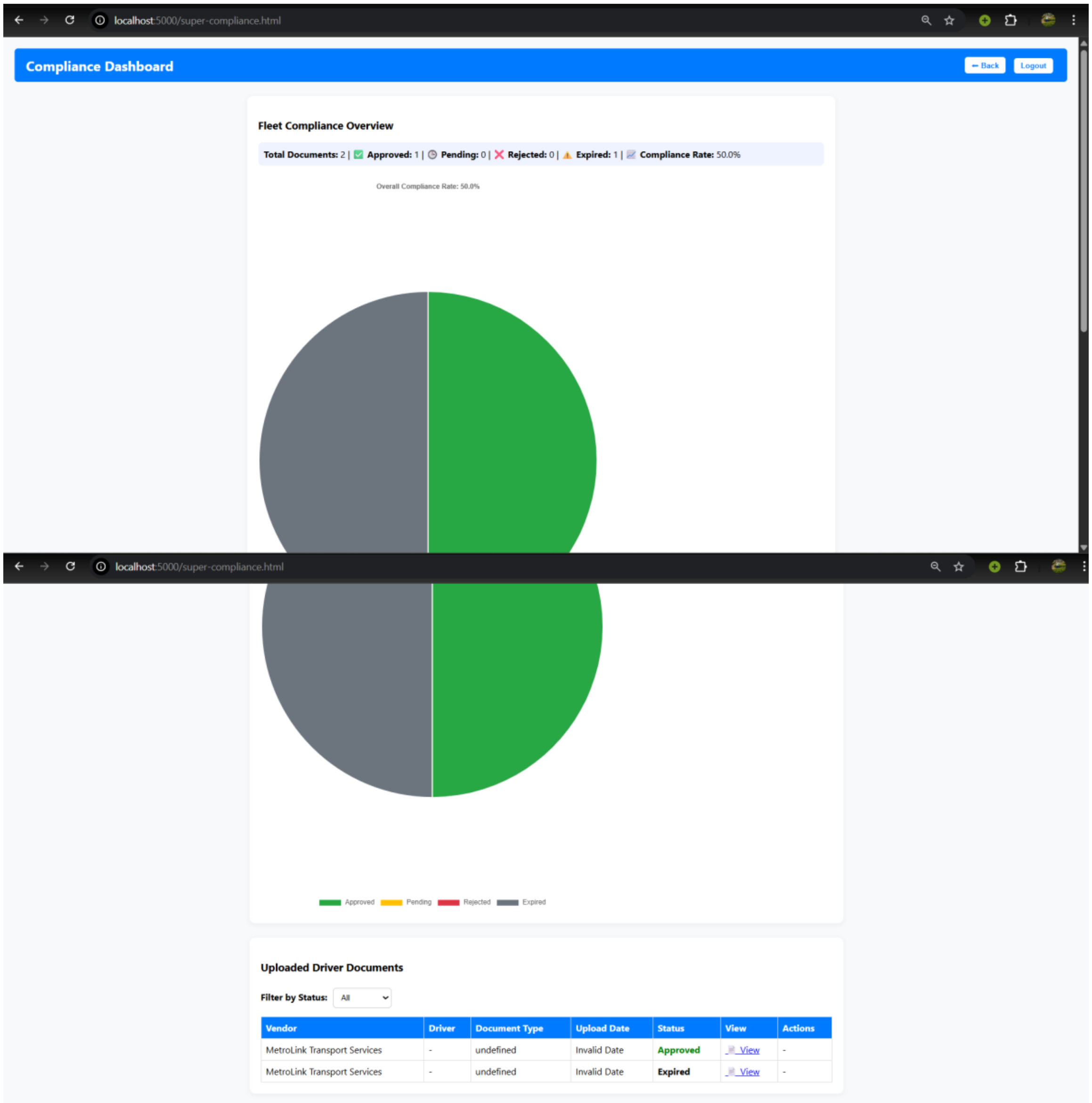
● Compliant Vendors: 0

● Partially Compliant: 1

● Non-Compliant: 0

SubVendor	Role	Region	Fleet	Drivers	Total Docs	Approved	Pending	Rejected	Compliance Rate
MetroLink Transport Services	CityVendor	Chennai, Tamil Nadu	1	1	2	1	0	0	50.0%

Super Vendor Dashboard


localhost:5000/super-compliance.html
← Back
Logout
Overall Compliance Rate: 50.0%
🟩 Approved 🟨 Pending 🟥 Rejected ⬛ Expired
localhost:5000/super-compliance.html

Compliance Dashboard

Fleet Compliance Overview

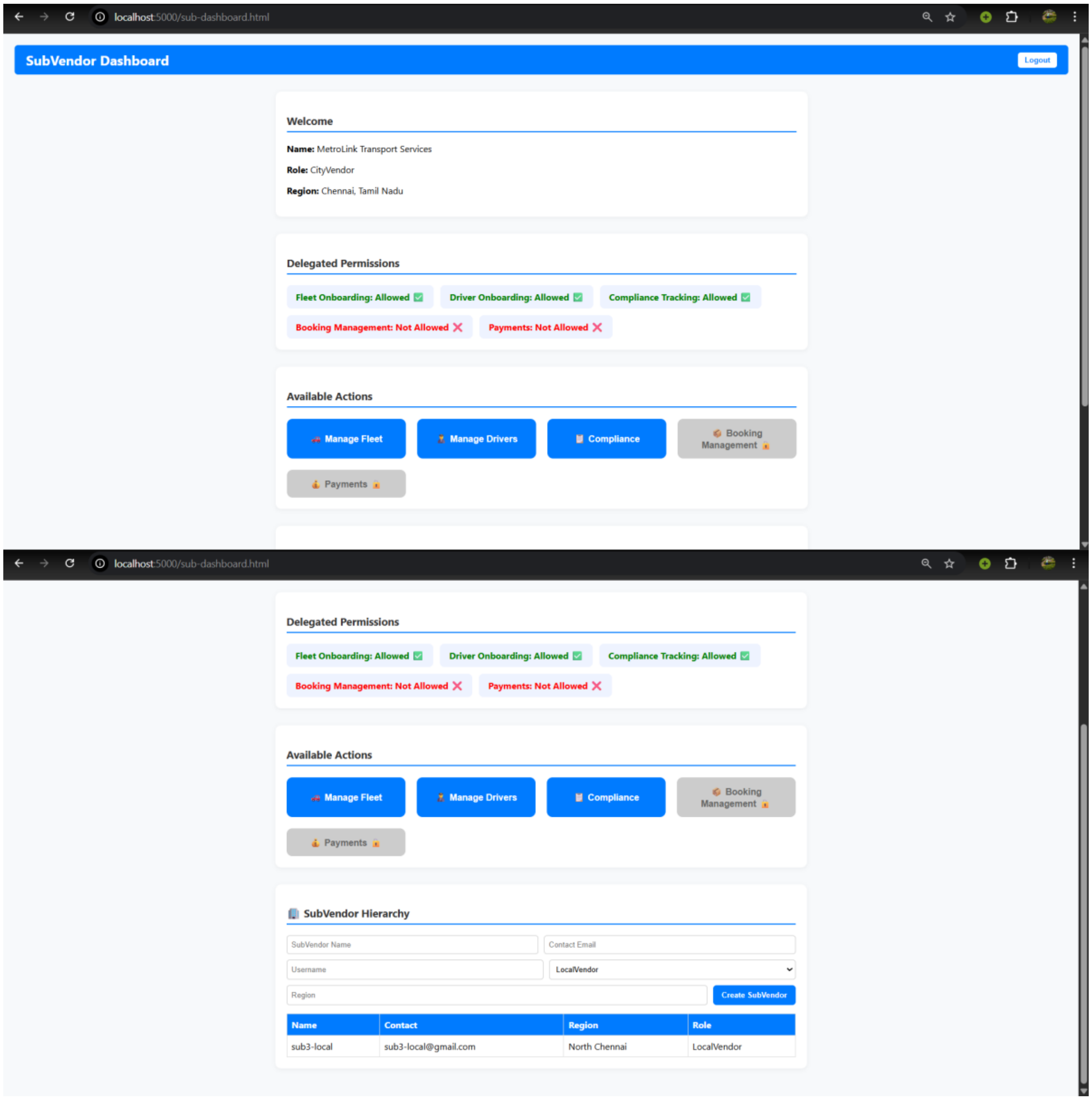
Total Documents: 2 | ✅ **Approved:** 1 | 🕒 **Pending:** 0 | ❌ **Rejected:** 0 | ⚠ **Expired:** 1 | 📈 **Compliance Rate:** 50.0%

Uploaded Driver Documents

Filter by Status: All

Vendor	Driver	Document Type	Upload Date	Status	View	Actions
MetroLink Transport Services	-	undefined	Invalid Date	Approved	View	-
MetroLink Transport Services	-	undefined	Invalid Date	Expired	View	-

Compliance Dashboard



Welcome

Name: MetroLink Transport Services

Role: CityVendor

Region: Chennai, Tamil Nadu

Delegated Permissions

Fleet Onboarding: Allowed ✅ | Driver Onboarding: Allowed ✅ | Compliance Tracking: Allowed ✅

Booking Management: Not Allowed ❌ | Payments: Not Allowed ❌

Available Actions

Delegated Permissions

Fleet Onboarding: Allowed ✅ | Driver Onboarding: Allowed ✅ | Compliance Tracking: Allowed ✅

Booking Management: Not Allowed ❌ | Payments: Not Allowed ❌

Available Actions

🏢 SubVendor Hierarchy

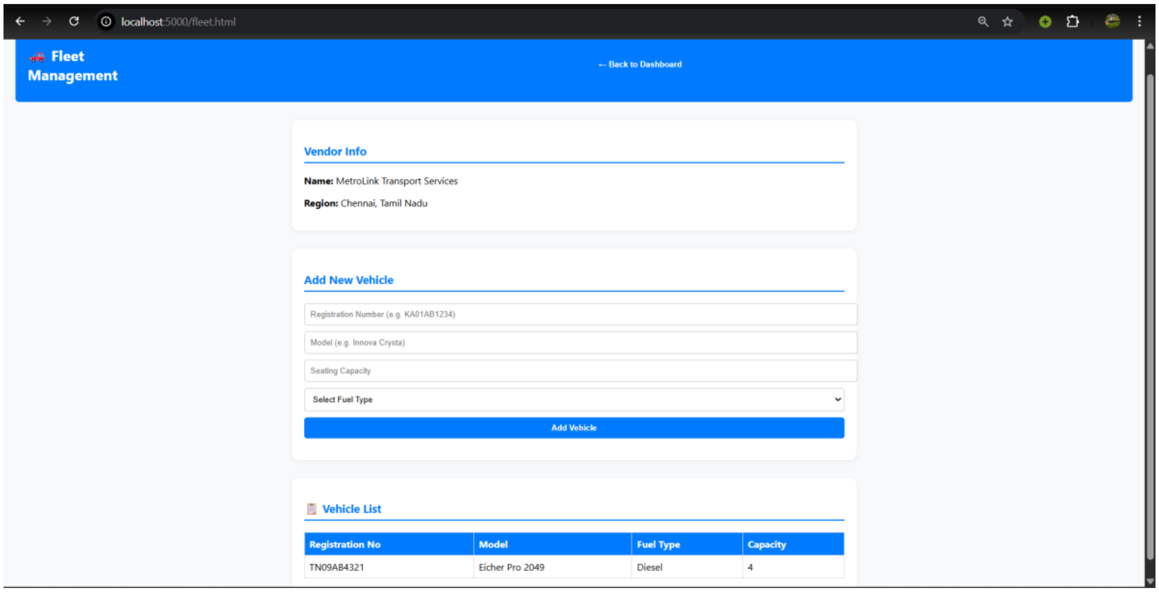
SubVendor Name | Contact Email

Username | LocalVendor ▾

Region | Create SubVendor

Name	Contact	Region	Role
sub3-local	sub3-local@gmail.com	North Chennai	LocalVendor

Sub Vendor Dashboard



localhost:5000/fleet.html

Fleet Management

← Back to Dashboard

Vendor Info

Name: MetroLink Transport Services

Region: Chennai, Tamil Nadu

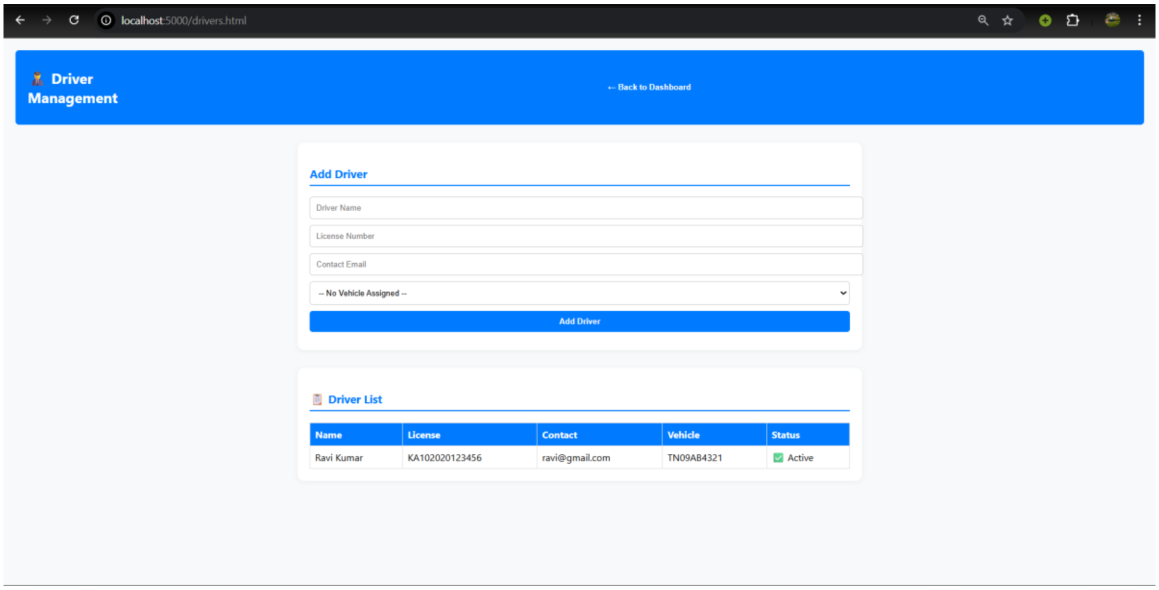
Add New Vehicle

Registration Number (e.g. KA01AB1234)
Model (e.g. Innova Crysta)
Seating Capacity
Select Fuel Type
Add Vehicle

Vehicle List

Registration No	Model	Fuel Type	Capacity
TN09AB4321	Eicher Pro 2049	Diesel	4

Fleet Management



localhost:5000/drivers.html

Driver Management

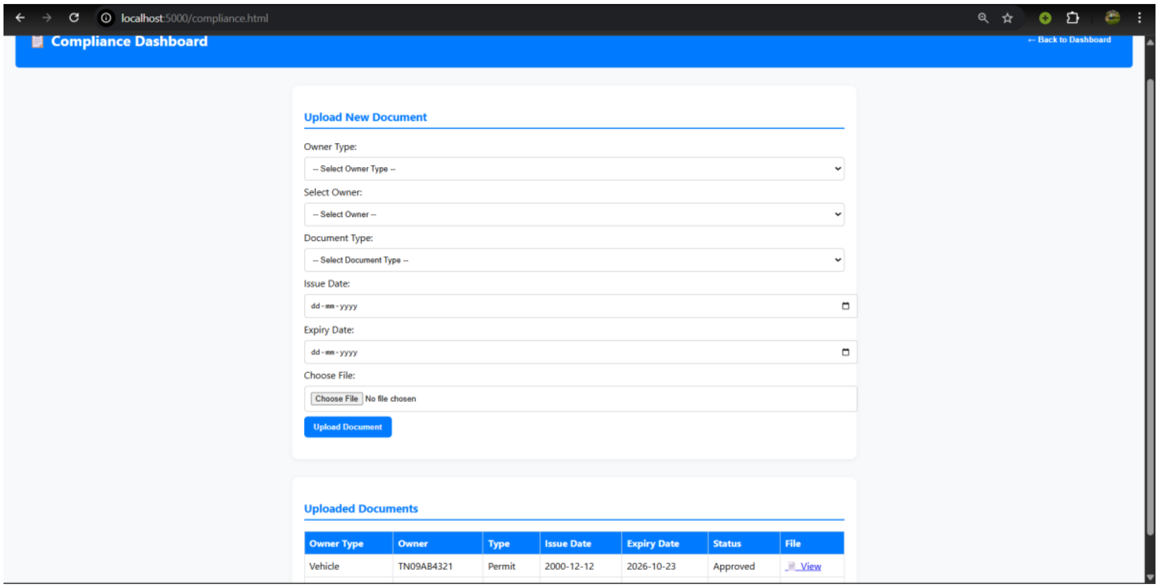
← Back to Dashboard

Add Driver

Driver List

Name	License	Contact	Vehicle	Status
Ravi Kumar	KA102020123456	ravi@gmail.com	TN09AB4321	✓ Active

Driver Management



Compliance Dashboard

← Back to Dashboard

Upload New Document

Owner Type:

-- Select Owner Type --

Select Owner:

-- Select Owner --

Document Type:

-- Select Document Type --

Issue Date:

dd-mm-yyyy

Expiry Date:

dd-mm-yyyy

Choose File:

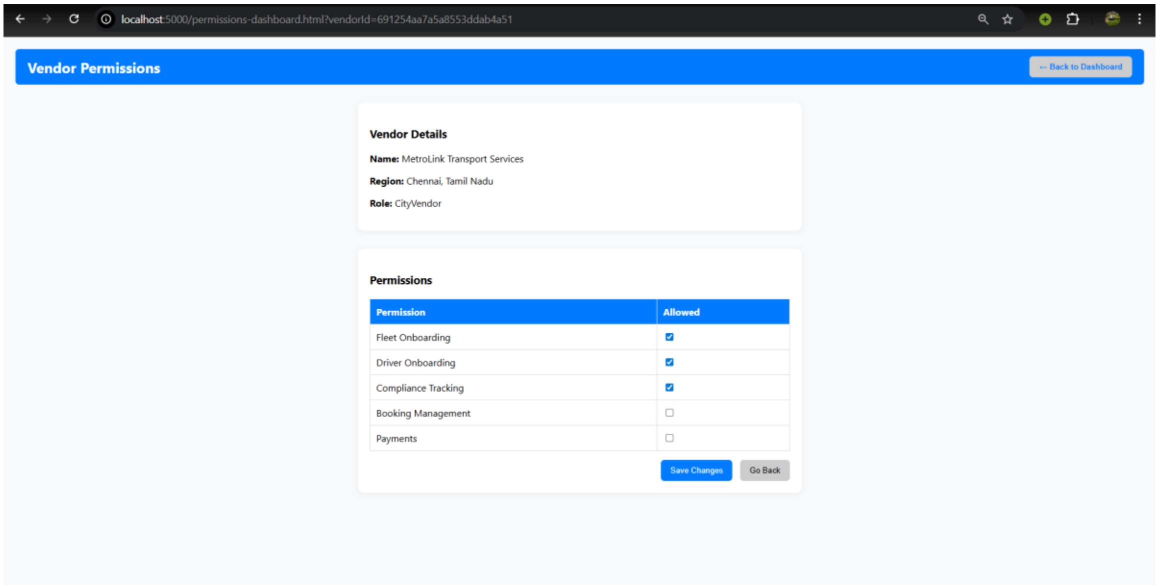
Choose File No file chosen

Upload Document

Uploaded Documents

Owner Type	Owner	Type	Issue Date	Expiry Date	Status	File
Vehicle	TN09AB4321	Permit	2000-12-12	2026-10-23	Approved	View

Compliance Dashboard – Documents Upload for Sub Vendor



localhost:5000/permissions-dashboard.html?vendorId=691254aa7a5a8553ddab4a51

Vendor Permissions

← Back to Dashboard

Vendor Details

Name: MetroLink Transport Services

Region: Chennai, Tamil Nadu

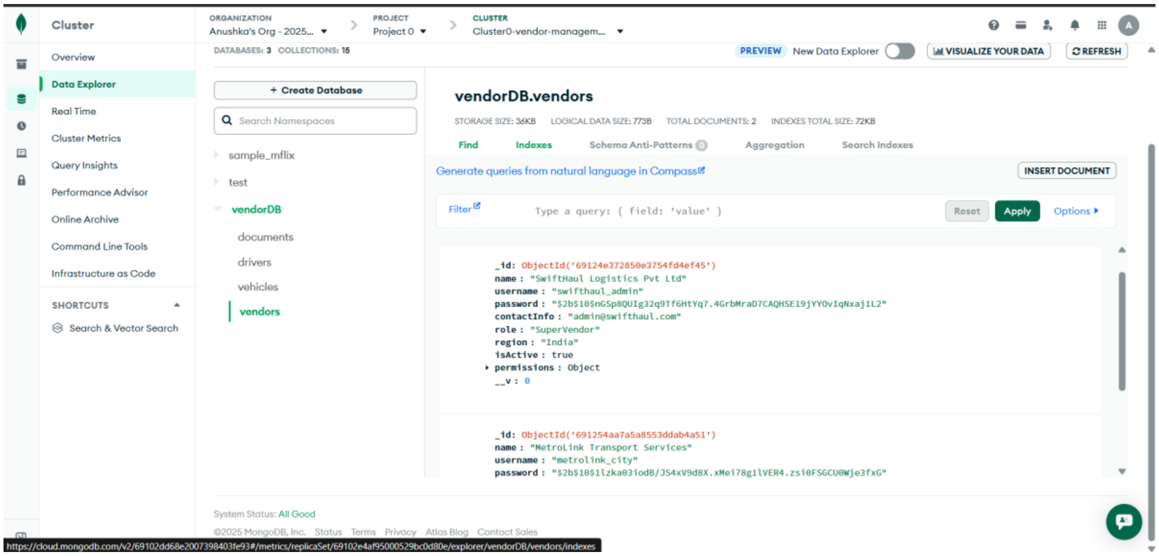
Role: CityVendor

Permissions

Permission	Allowed
Fleet Onboarding	☑
Driver Onboarding	☑
Compliance Tracking	☑
Booking Management	□
Payments	□

Save Changes Go Back

Manage Vendor Permission Dashboard for Super Vendor



```
_id: ObjectId('69124e372850e3754fd4ef45')
name : "SwiftHaul Logistics Pvt Ltd"
username : "swifthaul_admin"
password : "$2b$10$nGSp8QUIg32q9Tf6HtYq7.4GrbMraD7CAQHSE19jYYOvIqNxaj1L2"
contactInfo : "admin@swifthaul.com"
role : "SuperVendor"
region : "India"
isActive : true
permissions : Object
__v : 0

_id: ObjectId('691254aa7a5a8553ddab4a51')
name : "MetroLink Transport Services"
username : "metrolink_city"
password : "$2b$10$1lzka03iodB/JS4xV9d8X.xMei78g1lVER4.zsi0FSGCU0Wje3fxG"
```

https://cloud.mongodb.com/v2/69102dd68e2007398403fe93#/metrics/replicaSet/69102e4af95000529bc0d80e/explorer/vendorDB/vendors/indexes

MongoDB Atlas

7. Conclusion and Future Scope

The Vendor Management System successfully automates vendor and fleet onboarding workflows, improving operational efficiency and transparency. Future improvements include predictive analytics for vendor performance, integration with transport management systems, and an enhanced React-based front-end dashboard.

8. GitHub Repository

https://github.com/anushka-23-10/Vendor-Cab-and-Driver-Onboarding-Vendor-Hierarchy-Management