📘 Multi-Tenant Transport / Logistics SaaS

Architecture, Changes & Final Conclusion (Regenerated)

✂️ System ka Purpose (Simple Words)

Ye system ek **Multi-Tenant Transport / Logistics SaaS** hai.

👉 **Multi-Tenant** ka matlab: - Ek hi software - Multiple businesses use kar sakte hain - Lekin **har business (tenant) ka data alag aur secure hota hai**

Is system me har business ko hum **Account (accountId)** bolte hain.

✒️ Tenant (Account) Concept

Har tenant ke paas: - Apna accountId - Apna login system - Apna data

```
Tenant (Account)
 ├── Users (ADMIN / STAFF)
 ├── Suppliers
 ├── Companies
 ├── Vehicles
 └── Trips
```

👈Ek tenant dusre tenant ka data **kabhi nahi dekh sakta**.

🛏️ Account Types (Business Type)

Tenant ka type hota hai: - SUPPLIER - COMPANY - VEHICLE

Important Rule

> **Tenant ka type sirf permissions ke liye hota hai** Data structure same rehta hai

Isliye: - Supplier tenant bhi Companies & Vehicles bana sakta hai - Company tenant bhi Suppliers & Vehicles bana sakta hai - Vehicle tenant bhi Suppliers & Companies bana sakta hai

---

🌂Authentication (Login System)

JWT Based Authentication

Flow: 1. User login karta hai 2. JWT token generate hota hai 3. Token me ye data hota hai: - userId - accountId - role - accountType

protect Middleware

- JWT verify karta hai
- User ko DB se validate karta hai
- `req.user` me user attach karta hai

👉🏻Har request ab account-safe ho jati hai

🌡️ Authorization (Role & AccountType)

authorizeRoles Middleware

Ye 2 cheez check karta hai:

Check	Kyo
Role (ADMIN/STAFF)	Power control
AccountType	Business rule

Example: - Supplier create → Sirf ADMIN + SUPPLIER/COMPANY

🎒Database Design (Final)

Common Rule (Sab Models ke liye)

- `accountId` mandatory
- `isDeleted` for soft delete
- timestamps enabled

Account Model

- Root of tenant
- System isolation ka base

User Model

- Login system
- Role based access

- Account linked

Supplier / Company / Vehicle Models

Design philosophy: - Business records only - Simple fields - Easy future extension

Impact: - Naya field add karna easy - Koi API break nahi hoti

---

Trip Model (Future-Proof)

Trip me: - Supplier - Company - Vehicle - Rates - Load

Extra added: - `profit` - `isDeleted`

Profit formula:

```
(companyRatePerTon - vehicleRatePerTon) * totalTonLoad
```

🎓Soft Delete (Important Change)

Kya hai?

Data delete karne ke bajay:

```
isDeleted: true
```

Kyo?

- Data loss nahi hota
- Reports accurate rehti hain
- Audit possible hota hai

Impact

- Existing code safe
- Sirf queries me filter laga

🎩Audit Logging System

Kya log hota hai?

- Kis user ne

- Kya action kiya
- Kis record pe
- Kab kiya

Kahan lagaya?

👉Controllers me (NOT middleware)

Lagaya gaya hai: - CREATE - UPDATE - DELETE

Impact: - Debugging easy - Compliance ready

🎽Reports System

Reports Available

Report	Use
Supplier-wise	Business volume
Vehicle-wise	Performance
Company-wise	Revenue
Profit Summary	Overall profit

Route base:

```
/api/v1/reports
```

ADMIN only access

🕐Profit Auto Calculation

Kya kiya?

- Profit manually nahi bhejna
- Backend automatically calculate karta hai

Benefit

- Human error zero
- Reports consistent

✂️✂️Validation Rules

Trip create/update pe: - Rates > 0 - Load > 0

Benefit: - Galat data enter nahi hoga

✂️✒️Migration & Backward Compatibility

Kya delete nahi kiya

- 👈Koi model delete nahi
- 👈Koi route remove nahi

Kya add kiya

- Optional fields
- New reports APIs

Result: - Old code kaam karta rahega

✂️🛏️Overall Impact Summary

Area	Result
Security	Strong
Scalability	SaaS ready
Maintainability	High
Reporting	Advanced
Future changes	Easy

🔗Final Conclusion

Ye system ab:

> **Production-ready Multi-Tenant Transport SaaS**

Features: - Account isolation - Role based access - Audit logging - Soft delete - Powerful reports

Agar future me: - Naye fields - Naye reports - Naye roles

aate hain → system break nahi hoga.

📌Rule for Future Developers

> Controller = business logic Route = mapping only accountId = mandatory Soft delete = default

---

📄Swagger-Style API Documentation (Readable Format)

🔐Auth APIs

POST /api/v1/auth/register

Create new tenant admin user

Request:

```
{
  "name": "Admin",
  "email": "admin@test.com",
  "password": "password",
  "accountType": "SUPPLIER",
  "role": "ADMIN"
}
```

Response:

```
{ "token": "JWT_TOKEN" }
```

POST /api/v1/auth/login

Login user

Request:

```
{
  "email": "admin@test.com",
  "password": "password"
}
```

Response:

```
{ "token": "JWT_TOKEN" }
```

Users APIs

POST /api/v1/users

Create staff user (ADMIN only)

Headers:

```
Authorization: Bearer <token>
```

Request:

```
{
  "name": "Staff",
  "email": "staff@test.com",
  "password": "password",
  "role": "STAFF"
}
```

🗒️Supplier APIs

POST /api/v1/suppliers

Create supplier

Role: ADMIN AccountType: SUPPLIER / COMPANY

Request:

```
{
  "name": "ABC Supplier",
  "mobile": "9999999999"
}
```

GET /api/v1/suppliers

Get suppliers list

💼Company APIs

POST /api/v1/companies

Create company

🔐Vehicle APIs

POST /api/v1/vehicles

Create vehicle

🔒 Trip APIs

POST /api/v1/trips

Create trip

Request:

```
{
  "supplierId": "id",
  "companyId": "id",
  "vehicleId": "id",
  "totalTonLoad": 10,
  "companyRatePerTon": 2000,
  "vehicleRatePerTon": 1500
}
```

Auto calculated: - profit

📊Reports APIs

Base URL:

```
/api/v1/reports
```

GET /profit-summary

Overall profit report

Response:

```
{
  "totalTrips": 50,
  "totalProfit": 125000
}
```

🔍Audit Logs (Internal)

Automatically created on: - CREATE - UPDATE - DELETE

No public API exposed by default

😢End of Regenerated Document