School Transportation Management System (STMS) — System Documentation

Overview

A full-stack platform for schools to manage student transportation with live tracking, route planning, attendance, notifications, and analytics. Roles include Admin, Driver, Parent, and Student (optional).

Key Objectives

Improve safety, reduce late arrivals, streamline communication with parents, and give administrators operational visibility via dashboards and reports.

Modules & Features

Authentication & Roles

Role-based access for Admin, Driver, Parent, and Student using JWT or session auth. Password reset, audit logs, and optional 2FA for Admins.

Student Management

CRUD for student profiles, assignment to buses/routes, pickup and drop points, parent linkage, and bus-ride attendance via QR/RFID or manual check-in.

Bus & Route Management

Manage buses (capacity, status, maintenance), assign drivers, define routes with stop sequence, and show ETA using a maps API.

Driver Management

Driver onboarding, license and compliance tracking, shift schedules, and SOS alert that pings Admin and Parents for the current route.

Real-Time Tracking

GPS tracking for buses with live map, stop-by-stop ETA, delay detection, and historical trip playback for incident review.

Notifications

Push notifications and SMS for arrival alerts, boarding/deboarding, delays, and emergencies using Firebase Cloud Messaging or similar.

Payments (Optional)

Transport fee management, invoicing, online payments, and receipt history with export to PDF/Excel.

Reports & Dashboard

KPIs like on-time percentage, average delay, ridership, fuel logs, incidents; exportable reports for daily/weekly/monthly periods.

Tech Stack & Architecture

Frontend

React (or Vue) with a component library (Material UI or Tailwind), map component integration, and offline-first caching for the driver app.

Backend

Node.js with Express (or Django/FastAPI) exposing REST/GraphQL APIs, background jobs for notifications and ETA computation.

Database

MySQL/PostgreSQL for relational data or MongoDB; Redis for caching sessions/ETAs; object storage (e.g., S3-compatible) for files.

Integrations

Maps API for routing/ETAs, Firebase/OneSignal for push notifications, email/SMS gateway for critical alerts, payment gateway for fees.

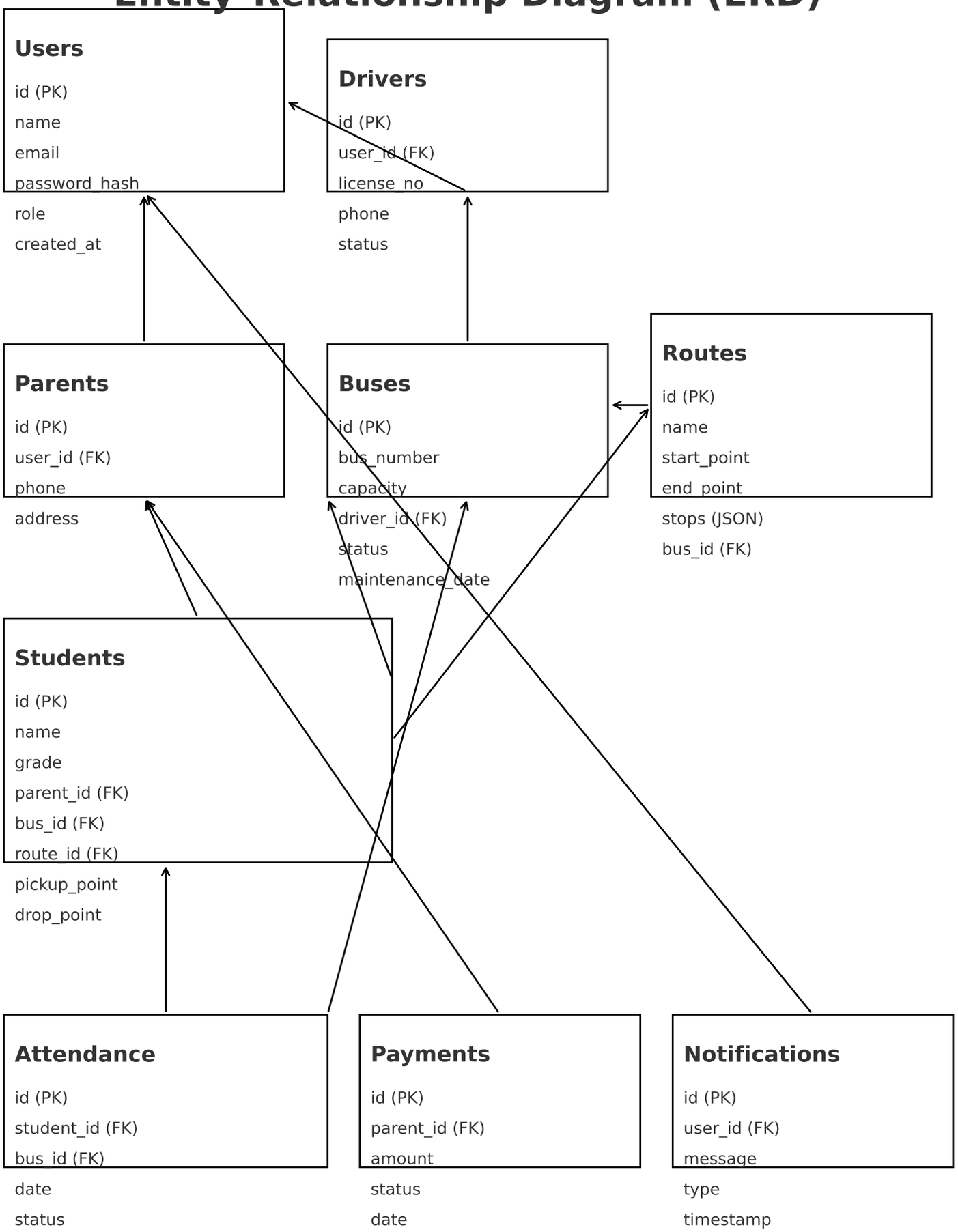
Security

RBAC, hashed passwords, JWT rotation, rate limiting, IP allowlists for admin panel, audit logs, and field-level encryption for PII.

Scalability

Stateless APIs, message queues for event processing, horizontal scaling of location-update consumers, and CDN for assets.

Entity–Relationship Diagram (ERD)



API Endpoints (Sample)

Auth

POST /auth/login, POST /auth/register, POST /auth/refresh, POST /auth/logout

Students

GET /students, POST /students, GET /students/:id, PATCH /students/:id, DELETE /students/:id

Buses & Routes

GET /buses, POST /buses, GET /routes, POST /routes, PATCH /routes/:id

Tracking

POST /telemetry (device -> server), GET /routes/:id/live, GET /trips/:id/replay

Attendance

POST /attendance/scan, GET /attendance?date=YYYY-MM-DD

Notifications

POST /notify, GET /notifications?userId=

Development Roadmap

Phase 1 — Foundations

Set up auth, RBAC, database schema, and CRUD for Students, Drivers, Buses.

Phase 2 — Routing & Tracking

Route creation, device telemetry ingestion, live map & ETAs.

Phase 3 — Attendance & Alerts

QR/RFID flow, boarding/deboarding events, push notifications.

Phase 4 — Payments & Reports

Fee plans, invoices, payments, dashboards, exports.

Phase 5 — Polish & Security

Comprehensive testing, performance tuning, monitoring, and SOC2-ready controls.