**📘 LikhilLMS – A Lightweight Tutor & Institute Management System**

**🎯 Overview**

**LikhilLMS** is a web-based Learning Management System designed specifically for **freelance tutors, coaching institutes, and small educational organizations**.  
The focus is on **simplicity, automation, and affordability** — unlike heavy platforms (Classplus, Teachmint), it is lightweight but includes all the **essential tutor workflows**: authentication, scheduling, attendance, payments, student management, and admin approvals.

**🚀 Core Features**

**👨‍🏫 User Roles**

* **Super Admin** → Manages platform-wide settings.
* **Organization Admin** → Creates and manages their institute’s profile, tutors, and students.
* **Tutor** → Conducts classes, manages students, adds payment/attendance.
* **Student** → Attends scheduled classes, linked to tutors/institutes.

**🔐 Authentication & Authorization**

* Built on **Supabase Auth** with email + password.
* Role-based access via profiles table (admin, tutor, student).
* **Admin approval flow** → tutors register, but need admin approval before becoming active.
* Supports **solo tutors** (without any organization).
* Session management using **Flask session cookies**.

**🗂️ Student Management**

* Tutors/Admins can **add students** with details (name, email, courses).
* Students linked to tutors (foreign key tutor\_id).
* Students linked to courses for progress tracking.

**📅 Class Scheduling**

* Tutors schedule classes with date, time, duration.
* Uses **Supabase (Postgres)** for storing schedules.
* Integrated with **n8n + Google Calendar** automation (classes auto-created in calendar).
* **Conducted class → moved to attendance table** automatically.

**📝 Attendance Management**

* Conducted classes shift into **attendance table**.
* Each record has → tutor, student(s), course, duration, date.
* UUID-based unique IDs for safe multi-tenant data.

**💰 Payment Records**

* Tutors can record **payment entries** per student:
  + Course fees
  + Advance hours
  + Advance amount
  + Cleared/uncleared payments
* Payments linked to tutor\_id → only visible to the right tutor.

**📊 Dashboards**

* **Admin Dashboard**:
  + Pending tutor approvals
  + Active tutors list
* **Tutor Dashboard**:
  + Upcoming classes
  + Attendance logs
  + Payment records
  + Student/course management

**⚙️ Tech Stack**

**Backend**

* **Flask (Python)** → lightweight web framework.
* **Supabase (Postgres + Auth + Storage)** → database + authentication.
* **n8n** → automation (class scheduling + Google Calendar sync).

**Frontend**

* **HTML + Jinja2 templates (Flask)**
* **Bootstrap 5** for styling.
* **FontAwesome icons** for UI polish.

**Database**

* **Supabase Postgres** with RLS (Row Level Security) policies.
* Tables:
  + auth.users → authentication (Supabase managed)
  + profiles → role, org mapping, full\_name, email, active status
  + organizations → institute details
  + students → student records
  + class\_schedule → planned classes
  + attendance → conducted classes
  + payment\_records → fee tracking

**DevOps**

* **GitHub** for version control.
* **Render / Hostinger VPS** for deployment.
* .env file for secret keys (Supabase URL, ANON key, Service key).

**🔮 Future Enhancements**

* 📧 Email notifications (class reminders, fee due reminders).
* 📱 Student app (React Native / Flutter) for attendance & classes.
* 📊 Analytics → tutor performance, payment summaries, student progress.
* 💳 Stripe/Razorpay integration for online fee payments.
* 🔗 Multi-tenant SaaS → each org gets their own subdomain (abc.likhillms.com).

**📌 Why LikhilLMS?**

* 🎯 Tailor-made for **small tutors & coaching centers** (not enterprise-heavy).
* 🔑 **Role-based authentication** with **solo tutor support**.
* ⚡ **Fast + lightweight** (Flask + Supabase) → deployable in days, not months.
* 🛠️ Built for extensibility: tutors can start small (attendance + payments) and scale into full SaaS LMS.

**✅ Example User Workflow**

1. A **tutor** signs up → goes into profiles as pending tutor.
2. **Admin** approves them from dashboard → tutor becomes active.
3. Tutor **adds students + schedules classes**.
4. When a class is finished → record shifts into **attendance table**.
5. Tutor records **payment entries** → visible in tutor dashboard.
6. **Admin** monitors all tutors & their activities from central dashboard.

⚡ Bhai, yeh documentation tumhara **README.md** ban sakta hai GitHub repo ke liye.  
Bas thoda formatting aur logo add kar do, ready ho jaayega.

👉 Kya tu chaahta hai main abhi isko proper **GitHub README file** ke format me convert karke de du (headings, badges, usage guide, screenshots placeholders)?