وصف مشروع: نظام إدارة إداري متكامل (ERP) - معهد Alpha Institute

1. نظرة عامة (Overview)

لمعهد تعليمي (ERP) المطلوب بناء نظام إدارة إداري شامل في الكويت (**Alpha Institute**، سابقاً "Rakan Institute"). الهدف الأساسي هو **أتمتة عمليات المعهد** وتقليل الاعتماد على عدد كبير من الموظفين الإداريين، عبر نظام واحد مركزي يدير الطلاب، المعلمين، الحضور، والمدفوعات المالية.

اللغة الأساسية للواجهة: **عربي** (RTL)
العملة: **الدينار الكويتي (KWD)**

---

2. السياق التجاري (Business Context)

المعهد يحتاج نظام يدير:

- تسجيل الطلاب ومتابعة بياناتهم
- جدولة الحصص والحضور
- المدفوعات والأرصدة المالية للطلاب (بشكل مرتبط تلقائياً بالحضور والحصص)
- إدارة المعلمين والمناهج
- تعرض مؤشرات الأداء (Dashboard) لوحة تحكم لصاحب المعهد/الإدارة (KPIs) الرئيسية

---

3. الأدوار والبوابات المطلوبة (Roles & Portals)

النظام يجب أن يدعم عدة بوابات (Portals) منفصلة حسب الدور:

الدور	الوصف
Admin (الإدارة)	صلاحية كاملة: إدارة الطلاب، المعلمين، المالية، التقارير، الإعدادات
Teacher (المعلم)	يشوف جدوله، يسجل الحضور، يتابع طلابه
Student (الطالب)	يشوف جدوله، حضوره، ورصيده المالي

الدور	الوصف
Parent (ولي الأمر)	يتابع بيانات ابنه: الحضور، الدفعات، التقدم الدراسي

> **مرجع تصميم (Vision Reference):** فيه prototype يوضح Google AI Studio أولي تم تصميمه على الشكل المطلوب — داشبورد عربي بأربع بوابات منفصلة (Admin/Student/Teacher/Parent)، مع فكرة داخل كل بوابة لمساعدة المستخدم. هذا AI دمج مساعد والتصميم البصري فقط، وليس UX يُستخدم كمرجع للـ حقيقي أو قاعدة بيانات auth كان بدون) كأساس تقني علائقية).

---

4. المتطلبات الوظيفية الأساسية (Functional Requirements)

4.1 المصادقة والصلاحيات (Auth & Permissions)

- تسجيل دخول آمن (JWT + httpOnly cookies)
- نظام أدوار وصلاحيات (Role-Based Access Control) للأدوار الأربعة أعلاه
- Rate limiting على محاولات تسجيل الدخول

4.2 إدارة الطلاب (Student Management)

- CRUD كامل لبيانات الطلاب (مع pagination)
- حالة الطالب (Status): نشط/موقوف/متخرج... إلخ (مضبوط enum)
- ربط الطالب بالحصص والمعلمين

4.3 النظام المالي (Payments)

- تسجيل الدفعات لكل طالب
- **حساب الرصيد تلقائياً** بناءً على الدفعات مقابل الحصص المستهلكة
- ربط الدفعات بالحصص (Payment ↔ Lesson relation)
- عرض الوضع المالي داخل ملف الطالب

4.4 الحضور (Attendance)

- تسجيل حضور/غياب لكل حصة
- **تحقق مالي تلقائي متقاطع**: عند تسجيل الحضور، يتحقق النظام من الرصيد ويحدّثه

4.5 لوحة التحكم (KPI Dashboard)

- مؤشرات مترابطة: عدد الطلاب، الإيرادات، نسب الحضور، حالة المعلمين... إلخ

- تصميم بصري غني (رسوم بيانية، بطاقات، جداول) — ليس مجرد نصوص

4.6 إدارة المعلمين

- بيانات المعلمين، جداولهم، ربطهم بالحصص والطلاب

---

5. المتطلبات التقنية (Technical Requirements)

الطبقة	التقنية
Framework	Next.js
Database	PostgreSQL (مستضاف على Neon)
ORM	Prisma — مع علاقات صحيحة (Foreign Keys)، enums لـ Role/StudentStatus/LessonStatus، وموديل ActivityLog لتتبع العمليات
Auth	JWT مع httpOnly cookies
Validation	Zod على كل المدخلات (API-level validation)
Deployment	Vercel (تطبيق Next.js)

الطبقة	التقنية
Domain	Hostinger (DNS فقط، مو استضافة التطبيق)

ملاحظات أمنية وهيكلية مهمة:

- على Next.js + PostgreSQL **لا يجوز** استضافة مثل (Shared Hosting) استضافة مشتركة Hostinger — لازم فصل: Vercel للتطبيق + Neon لقاعدة البيانات + Hostinger للدومين فقط
- بين الجداول relations كل الـ (طالب ↔ دفعة ↔ حصة ↔ حضور) يجب أن تكون حقيقية في Prisma، وليست حقول Foreign Keys غير مرتبطة (strings) نصية
- كل الحقول ذات الحالات المحددة (status fields) يجب أن تكون typed enums، وليست نصوص حرة
- يجب تفادي أي hardcoded مربوطة بـ fetch calls localhost في الكود

---

6. الوضع الحالي للمشروع (Current State)

- **تأسيسي مُصحح** (خطوات 1-13) **Prompt** تم إعداد يغطي: إعداد المشروع، مخطط قاعدة البيانات، auth آمن مع rate limiting، Zod validation، CRUD الطلاب مع pagination، واجهة إدارة الطلاب، API المدفوعات مع حساب الرصيد التلقائي، الواجهة المالية الحضور مع التحقق المالي API، داخل ملف الطالب مترابطة، واختبارات التكامل KPI المتقاطع، لوحة
- التطوير المحلي شغال (localhost:3000)
- **المشكلة المفتوحة حالياً:** لوحة التحكم (Dashboard) تفتقر للعناصر البصرية والمكونات (مجرد labels تحتاج إعادة تصميم بصري — (أساسية

---

7. التسليمات المتوقعة (Deliverables)

1. نظام كامل يعمل على الخطوات 1-13 المذكورة أعلاه
2. تصميم يقارب تجربة الـ AI Studio prototype كمرجع UX فقط) لوحة تحكم بصرية غنية
3. نشر فعلي على Vercel + Neon + ربط الدومين عبر Hostinger
4. توثيق أساسي لكيفية التشغيل والصيانة

---

8. ملاحظة للمهندس

هذا المشروع كان يُبنى سابقاً بمساعدة أداة AI coding (OpenCode) لكن ظهرت مشاكل جوهرية في الكود الناتج غير صحيحة، عدم وجود Prisma ناقص، علاقات (auth error handling أو pagination). المطلوب من المهندس مراجعة البنية الحالية والبناء عليها بشكل احترافي ومستقر، وليس بالضرورة البدء من الصفر.