بسم الله الرحمن الرحيم

خطة إنشاء مشروع إدارة عمليات الحجز

نظرة عامة على المشروع (MVP Overview)

- فكرة المشروع: إطلاق MVP لمنصة (Web & Mobile) بتشتغل كـ Marketplace بيربط مقدمي الخدمات بالعملاء لحجز المواعيد.
- الهدف من الـ MVP: اختبار الفكرة في السوق (Market Validation) بأهم الـ Core Features فقط، وإثبات إن النظام قادر يحل مشكلة الحجز اليدوي.
- الفئة المستهدفة: مقدمي الخدمات (عيادات/صالونات) والعملاء اللي بيدوروا على حجز سريع.
- القيمة اللي بيقدمها: تبسيط عملية الحجز، تنظيم الجداول، وتوفير الـ Effort على الطرفين.

المزايا الأساسية للـ :

هنركز هنا على الـ "Must-Haves" بس:

- تسجيل الدخول (Basic Auth): للعميل ومزود الخدمة والإدمن.
- إدارة الخدمات: مزود الخدمة يقدر يضيف خدماته بأسعارها ومدتها.
- نظام الحجز الأساسي: تقويم (Calendar) بيعرض الأوقات المتاحة، والعميل يقدر يحجز بناءً عليها.
- حالة الموعد (Booking Status): (مؤكد، ملغي، مكتمل).
- الدفع الإلكتروني (Basic Payment): دمج بوابة دفع أساسية لضمان جدية الحجز.
- لوحة تحكم مبسطة للإدارة (Basic Admin Dashboard): لإدارة المستخدمين ومتابعة الحجوزات.

البنية التقنية (Tech Stack)

- تطبيق الموبايل: Flutter، عشان نبني للـ iOS والـ Android بكود واحد ونوفر وقت التطوير للـ MVP. هنطبق Clean Architecture عشان الكود يبقى Scalable بعدين.
- الواجهة الأمامية للويب: Angular، ممتازة لبناء لوحة تحكم سريعة (SPA).
- الباك إند (Backend Options):
- الخيار الأول: Laravel (PHP): اختيار ممتاز جداً للـ MVP لأن الـ Ecosystem بتاعه مليان Packages جاهزة للـ Auth، الـ Payment، والـ APIs، وده هيسرع الـ Development Cycle بشكل رهيب.
- الخيار التاني: Dart Vania: اختيار عبقري لو عايزين نوحد الـ Stack مع تيم الموبايل (Full-Stack Dart)، وده هيقلل الـ Context Switching ويخلي الـ Communication أسرع بين التيم.
- قاعدة البيانات: MySQL، لأن العلاقات بين الحجوزات والمستخدمين واضحة ومحتاجة Relational DB.
- التصميم: Figma

النظام Architecture

- البنية: Client-Server Architecture بسيطة. الموبايل والويب بيكلموا الـ Backend (سواء كان Larave أو Vania) عن طريق RESTful APIs.
- الـ APIs: هترجع Data بصيغة JSON، وهتكون Documented بشكل واضح عشان تيم الـ Frontend.

- الأمان: استخدام JWT (JSON Web Tokens) للـ Authentication وتأمين الـ Endpoints.

مخطط قاعدة البيانات (MVP Database Schema)

جداول الـ MVP هتكون كالتالي:

- Users (id, name, email, password, role)
- Providers (id, user_id, business_name, description)
- Services (id, provider_id, name, price, duration)
- Bookings (id, client_id, service_id, booking_datetime, status)
- Payments (id, booking_id, amount, status)

خطة التنفيذ (Agile Scrum للمرحلة الحالية)

- هنقسم الشغل لـ Sprints قصيرة، كل Sprint مدته أسبوعين بيركز على تجميع Features شغالة وقابلة للـ Testing.
- التركيز كله هيكون على الـ Local Development وتجهيز الـ Repositories لتسليم الكود.

الجدول الزمني للتنفيذ (45 يوم)

Phase 1: تصميم الـ MVP (المدة: 15 يوم)

- تصميم شاشات الموبايل والويب الأساسية على Figma والموافقة عليها.

Phase 2: التطوير البرمجي (المدة: 30 يوم)

- Sprint 1 (تجهيز الأساسيات): بناء الداتابيز، الـ Auth APIs، وتجهيز الـ Base code للـ Flutter والـ Angular.

- Sprint 2 (نواة النظام): برمجة الـ Services، نظام الـ Calendar، والـ Bookings Logic في الباك إند وربطه بالفرونت.
- Sprint 3 (اللمسات الأخيرة): ربط بوابة الدفع الأساسية، وإنهاء لوحة التحكم للإدمن.
- Sprint 4 (الـ Handoff & QA): عمل Testing محلي (Local QA)، إصلاح الـ Bugs، وتجهيز الكود للتسليم النهائي للعميل.

الفريق المطلوب للـ MVP

فريق Lean جداً عشان نقلل التكلفة والوقت:

- 1 Project Manager
- 1 UI/UX Designer
- 1 Flutter Developer
- 1 Backend Developer (Laravel OR Dart Vania)
- 1 Angular Developer

المخرجات النهائية للمشروع (Final Deliverables)

بما إن الاتفاق على عدم النشر، المخرجات هتكون كالتالي:

1. Source Code: الكود المصدري النظيف بالكامل للـ (Flutter، Backend، Angular) مرفوع على مستودع (GitHub/GitLab).
2. Database Dump: ملف الـ SQL الخاص بقاعدة البيانات شامل الـ Seeders والـ Migrations.
3. Figma Files: لينكات التصميم الأصلية.

4. API Documentation: ملف Postman Collection أو Swagger بيشرح كل الـ Endpoints وكيفية تشغيلها.
5. README File: ملف تقني بيشرح للـ Developers إزاي يعملوا Run للكود ده Local عندهم خطوة بخطوة.

المخاطر المحتملة (Risks & Mitigation)

- الخطر (Scope Creep): الرغبة في إضافة مميزات جديدة خارج نطاق الـ MVP (زي نظام شات، أو تقييمات معقدة).
- الحل: الالتزام الصارم بالـ Features المكتوبة في العقد فقط. أي ميزة إضافية بتتسجل في الـ Backlog للنسخة الـ V2.