1⃣ الفرق بين Single Page Application و Multiple Page Application

الجانب	SPA – تطبيق صفحة واحدة	MPA – تطبيق متعدد الصفحات
تحميل الصفحات	يتم تحميل صفحة HTML واحدة فقط في البداية، والباقي يتم تحديثه ديناميكيًا عن طريق JavaScript	يتم تحميل صفحة HTML جديدة من السيرفر في كل مرة ينتقل فيها المستخدم لصفحة جديدة
الأداء	سريع بعد التحميل الأولي، فقط البيانات تنتقل عبر الـ API وليس كامل الصفحة	أبطأ، كل انتقال يحتاج تحميل الصفحة كاملة والملفات المرتبطة
تجربة المستخدم	سلسة، تشبه التطبيقات، التفاعل سريع	تقليدية، كل حركة يعقبها إعادة تحميل الصفحة
SEO (تحسين محركات البحث)	صعب بعض الشيء، يحتاج Server-Side Rendering أو Pre-rendering	أسهل، كل صفحة لها URL ومحتوى مستقل
تعقيد التطوير	أعلى، يحتاج Frameworks مثل React, Angular, Vue	أبسط، يمكن عمله باستخدام تقنيات السيرفر مثل ASP.NET MVC أو Django
الاستخدام الأنسب	لوحات تحكم، تطبيقات تفاعلية (مثل بريد إلكتروني، شبكات اجتماعية)	مواقع محتوى ثابت أو تسويقي (مثل المدونات، المتاجر الإلكترونية)

أفضل خيار لكل مشروع من مشاريعكم الثلاثة

1. **منصة الترفيه (Streaming + Subscriptions)**
   - **مناسب:** SPA
   - **السبب:** التطبيق يحتاج تفاعل عالي، تحميل فيديوهات سريع، فلترة المحتوى، وكلها أفضل مع SPA.
2. **تطبيق تخطيط الرحلات / حجز السفر**
   - **مناسب:** SPA (أو مزيج Hybrid)
   - **السبب:** البحث عن رحلات وفنادق ونشاطات يحتاج تحديث سريع للبيانات دون إعادة تحميل الصفحة بالكامل.
3. **الدليل الفرعوني التفاعلي ثلاثي الأبعاد**
   - **مناسب:** SPA
   - **السبب:** التفاعل مع 3D، الرسوميات الحية، ونظام الذكاء الاصطناعي يحتاج تحديث ديناميكي وسلس.

الخلاصة: كل المشاريع الثلاثة أفضل مع SPA بسبب التفاعل الديناميكي وسرعة تجربة المستخدم. MPA مناسب فقط لو كانت صفحات ثابتة أو محتوى بسيط.

2️⃣ الفرق بين REST و gRPC

الجانب	REST	gRPC
البروتوكول	HTTP 1.1 / 2، غالبًا JSON	Binary (Protobuf)، HTTP/2
صيغة البيانات	JSON (قابل للقراءة)	Protobuf (صغير وسريع، ثنائي)
الأداء	أبطأ، يحتاج تحويل JSON	أسرع، تحويل بيانات خفيف
البث Streaming	محدود	مدعوم بشكل كامل
نوعية البيانات Type Safety	ضعيف، يعتمد على التوثيق	قوي، يتم توليد الكود تلقائيًا للعميل والخادم
دعم المتصفحات	مدعوم أصليًا	يحتاج gRPC-Web أو Proxy
الاستخدامات	APIs عامة أو مواقع ويب	Microservices، التطبيقات الداخلية، الأداء العالي

الخلاصة:

- **REST:** سهل، مدعوم على كل المتصفحات، سهل القراءة والتصحيح ← جيد للـ APIs العامة.
- **gRPC:** سريع، قوي في التحقق من البيانات، ثنائي وصغير الحجم ← جيد للاتصال الداخلي بين الخدمات، والتطبيقات التي تحتاج أداء عالي أو بث بيانات حي.

مثال على مشاريعكم

المشروع	REST	gRPC
منصة الترفيه	REST للـ APIs العامة (المستخدمين، الاشتراكات)	gRPC داخليًا (معالجة الفيديو، التوصيات)
تطبيق السفر	REST للتعامل مع خدمات خارجية (خطوط طيران، فنادق)	gRPC داخلي (بحث سريع عن التوافر، تسعير ديناميكي)
الدليل الفرعوني	REST لمعلومات عامة	gRPC للتفاعل مع الذكاء الاصطناعي وبيانات 3D بسرعة
