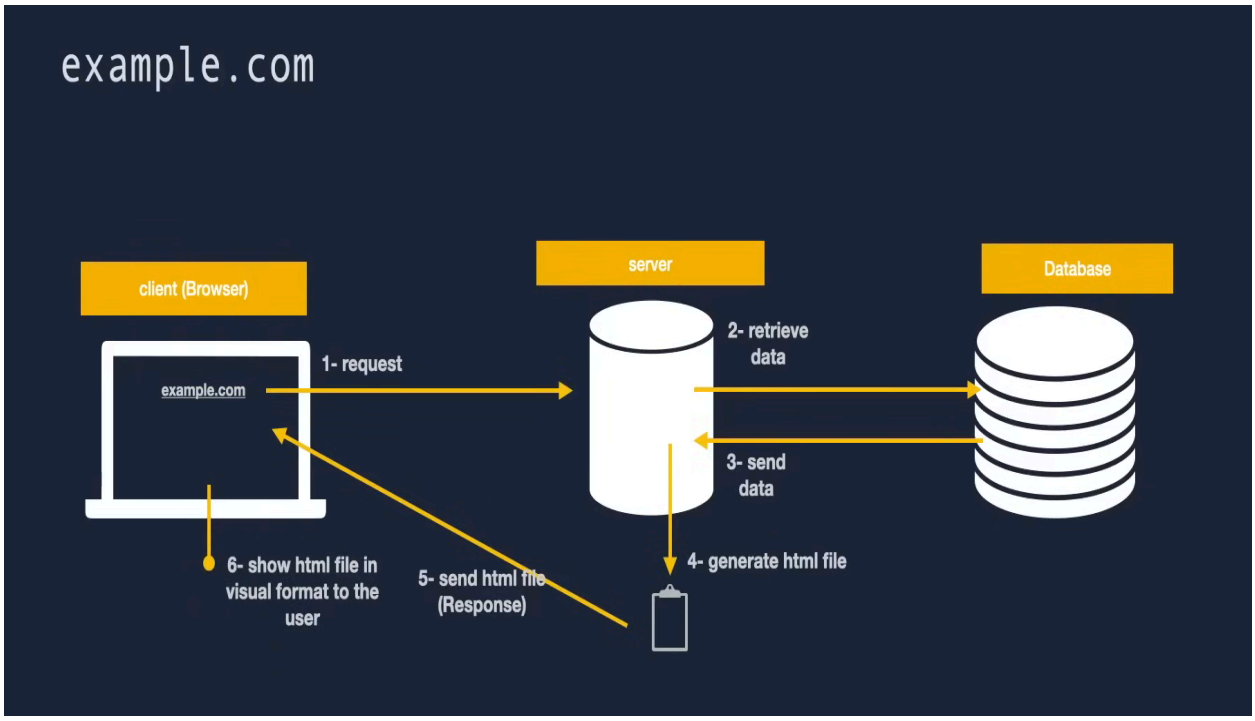
اول شي نفهم كيف عملية الوصول الى المواقع الالكترونية او كيف بتشتغل المواقع
لنفترض انو بدي ادخل على موقع اسمو example.com

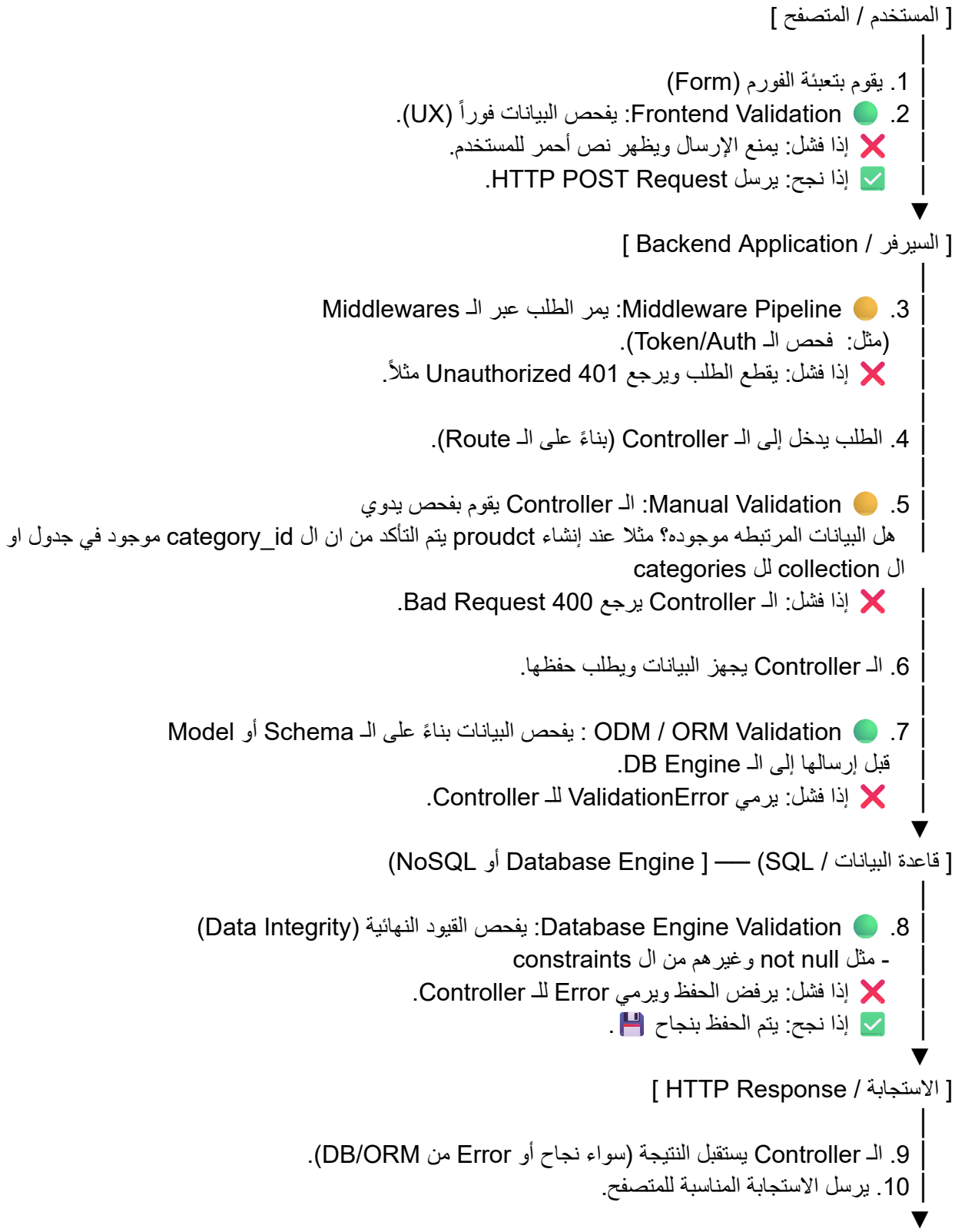
1. لما نكتب اسم الموقع example.com في المتصفح هون الـ DNS رح يحول اسم الموقع إلى عنوان الـ IP الخاص بالسيرفر وبعدها يرسل المتصفح Request إلى السيرفر
2. هسا هذا ال request رح يروح على ال server وحسب الطلب لو بدو بيانات بروح بجيبها من الداتابيس
3. بعد ما ال server جاب البيانات من قاعدة البيانات رح يروح يعمل generate ل html file ويعبي البيانات فيه وهاي العملية اسمها rendering او رح يجهز البيانات على شكل json في حالة ال SPA
4. لما تكتمل علمية ال generate لل html file وقتها رح يرجعو ك response للمتصفح والمتصفح رح يعرض المحتويات على شكل مرئي ألوان نصوص وصور مشان تظهر للمستخدم



هسا نرجع للسؤال اللي وهو على اي layer من ال layers الموقع تتم تطبيق ال validation ؟
الجواب كان على كل الطبقات frontend and backend and database طيب ليش؟ مشان دائما بالويب في قاعدة وهي Never Trust the Client مثلا لو تم تخطي علمية ال validatoin من جهة ال client يكون في validation من جهة ال server + database

طبقة ال frontend ليش مهمة؟ مشان تحسين تجربة المستخدم UX
وال backend ؟ مشان ال security حماية البيانات من اي طلبات عشوائية تجاوزت المتصفح وتطبيق ال Business Logic
وال database ؟ مشان سلامة البيانات

مثال لو بدنا نعمل create item


[المستخدم / المتصفح]
1. يقوم بتعبئة الفورم (Form)
2. 🟢 Frontend Validation: يفحص البيانات فوراً (UX).
❌ إذا فشل: يمنع الإرسال ويظهر نص أحمر للمستخدم.
✅ إذا نجح: يرسل HTTP POST Request.
[السيرفر / Backend Application]
3. 🟡 Middleware Pipeline: يمر الطلب عبر الـ Middlewares
(مثل: فحص الـ Token/Auth).
❌ إذا فشل: يقطع الطلب ويرجع 401 Unauthorized مثلاً.
4. الطلب يدخل إلى الـ Controller (بناءً على الـ Route).
5. 🟡 Manual Validation: الـ Controller يقوم بفحص يدوي
هل البيانات المرتبطه موجوده؟ مثلا عند إنشاء proudct يتم التأكد من ان ال category_id موجود في جدول او
ال collection لل categories
❌ إذا فشل: الـ Controller يرجع 400 Bad Request.
6. الـ Controller يجهز البيانات ويطلب حفظها.
7. 🟢 ODM / ORM Validation : يفحص البيانات بناءً على الـ Schema أو Model
قبل إرسالها إلى الـ DB Engine.
❌ إذا فشل: يرمي ValidationError للـ Controller.
[قاعدة البيانات / Database Engine] ── (SQL أو NoSQL)
8. 🟢 Database Engine Validation: يفحص القيود النهائية (Data Integrity)
- مثل not null وغيرهم من ال constraints
❌ إذا فشل: يرفض الحفظ ويرمي Error للـ Controller.
✅ إذا نجح: يتم الحفظ بنجاح 💾 .
[الاستجابة / HTTP Response]
9. الـ Controller يستقبل النتيجة (سواء نجاح أو Error من DB/ORM).
10. يرسل الاستجابة المناسبة للمتصفح.