**🧠 VISIÓN AMPLIADA DEL PROYECTO**

**Sistema de Reservas Online para Hotel  
✅ Soporte para múltiples roles (cliente, recepcionista, administrador)  
✅ Reservas de habitaciones (extensible a otros recursos)  
✅ Panel de administración, autenticación JWT, y despliegue profesional**

**🔩 ARQUITECTURA GENERAL**

* **Frontend: Angular (con lazy loading y guards por roles)**
* **Backend REST: Node.js con Express + PostgreSQL**
* **Backend paralelo (extra experiencia): Spring Boot + JPA + Security**
* **Base de Datos: PostgreSQL (o MySQL opcional)**
* **Herramientas: Git, Trello, Notion, Docker, Swagger, Postman**

**🧭 PLAN MAESTRO DETALLADO**

**🚀 Pre-Sprint 0: Planificación + Notion/Trello**

* **Organiza tareas y cronograma realista**
* **Crea dashboard en Notion con:**
  + **Objetivos del proyecto**
  + **Documentación técnica**
  + **Checklist de cada sprint**
  + **Historias de usuario**
  + **Glosario técnico**
* **Setup en Trello (sprints, backlog, QA, bugs, docs)**

**🏁 SPRINT 1: Setup del Proyecto + Base de Datos**

**Objetivo: Tener la estructura base del repo, el diagrama ER y scripts SQL.**

**📌 Extras opcionales útiles:**

* **Script para reiniciar DB (reset-db.sh)**
* **Añadir prettier, eslint, y hooks con Husky**

**✅ *Consejo para volverte autónomo*: Practica hacer tu diagrama ER en draw.io tú solo. Empieza con 3 tablas y valida relaciones (1:N, N:M).**

**🏁 SPRINT 2: Backend Node.js - Usuarios + Auth**

**Objetivo: Registrar, loguear y autenticar usuarios con JWT y roles.**

**📌 Pasos extra:**

* **Añade validaciones con express-validator**
* **Usa .env para manejar claves (dotenv)**
* **Estructura tu backend en /routes, /controllers, /models, /middleware**

**✅ *Tip ninja*: Aprende a usar Postman con variables de entorno y colecciones para testear más rápido.**

**🏁 SPRINT 3: Backend Node.js - CRUD Reservas**

**Objetivo: Crear, listar, actualizar y cancelar reservas con validaciones.**

**📌 Ampliaciones útiles:**

* **Middleware para validar disponibilidad**
* **Endpoint para ver solo reservas del usuario autenticado**
* **Logs en consola con timestamps (morgan)**

**✅ *Sugerencia*: Esquematiza qué validaciones harías si fueras recepcionista. Aplica esa lógica en tu endpoint.**

**🏁 SPRINT 4: Frontend Angular - Base + Login**

**Objetivo: Estructura del proyecto, rutas y auth funcional.**

**📌 Suma estos pasos:**

* **Usa interceptors para insertar JWT en headers**
* **Guarda el token en localStorage**
* **Crea servicio de Auth reutilizable**

**✅ *Tip*: Intenta hacer un esquema de flujo de login en Figma o draw.io (te ayuda a visualizar mejor la lógica del token).**

**🏁 SPRINT 5: Frontend Angular - Reservas**

**Objetivo: Crear, visualizar y cancelar reservas desde el frontend.**

**📌 Mejoras a considerar:**

* **Componente de calendario interactivo (ngx-bootstrap o angular-calendar)**
* **Filtro por fechas para buscar reservas**
* **Confirmación con SweetAlert2**

**✅ *Consejo*: Intenta primero maquetar sin funcionalidad, luego conecta con los servicios y valida.**

**🏁 SPRINT 6: Admin Panel + Gestión Roles**

**Objetivo: Dashboard para visualizar todo y CRUD de habitaciones.**

**📌 Extensiones clave:**

* **Vista tipo tabla (usando Angular Material o PrimeNG)**
* **Paginación y filtros**
* **CRUD completo de habitaciones**
* **Mostrar diferentes componentes según rol**

**✅ *Tip*: Revisa el JWT en consola para entender cómo funcionan los roles y modifica el frontend acorde.**

**🏁 SPRINT 7: Spring Boot + JPA**

**Objetivo: Replicar backend en Java para ganar dominio de ese stack.**

**📌 Extras útiles:**

* **Lombok para evitar boilerplate**
* **Spring Security + JWT**
* **Swagger para documentación automática**

**✅ *Consejo pro*: Intenta mapear primero las entidades en papel o draw.io antes de pasar al código.**

**🏁 SPRINT 8: Docker, Swagger, Testing, Deploy**

**Objetivo: Profundizar en DevOps y dejar el sistema productivo.**

**📌 Incluye también:**

* **Docker Compose para Node + DB**
* **Swagger UI con OpenAPI 3.0**
* **Tests con Postman o jest (mínimo para usuarios y reservas)**
* **Despliegue:**
  + **Vercel para Angular**
  + **Render o Railway para el backend**

**✅ *Tip clave*: Automatiza al menos un deploy (o documentación Swagger) para ganar confianza.**

**🛠️ CONSEJOS PARA VOLVERTE AUTÓNOMO SIN DEJARME DE LADO**

| **Habilidad** | **Recomendación** |
| --- | --- |
| **📘 Documentación** | **Usa Notion para registrar TODO: endpoints, modelos, errores comunes.** |
| **👨‍💻 Práctica con código** | **Reescribe funciones que te doy, intenta cambiarlas y rompelas para entenderlas.** |
| **🧪 Testing básico** | **Aprende a usar Postman con colecciones y variables de entorno.** |
| **🔄 GitFlow** | **Trabaja con ramas: main, dev, feature/xyz, y haz pull requests.** |
| **🤔 Preguntas buenas** | **Cuando me pidas ayuda, dime: “qué hice”, “qué esperé”, “qué pasó”.** |
| **📂 Organización** | **Cada módulo (auth, reservas, admin) debe tener su propia carpeta/servicio.** |