**Framework Comparison Report**

**Domain: Web Application**

**Frontend Comparison: React vs. Next.js**

| **Feature** | **React (TypeScript)** | **Next.js (TypeScript)** |
| --- | --- | --- |
| **Architecture** | Component-based UI library | Full-stack framework with SSR/SSG |
| **Routing** | Uses React Router | File-based routing built-in |
| **Server-Side Rendering (SSR)** | Requires extra setup (Next.js, Gatsby) | Native support |
| **Static Site Generation (SSG)** | Requires additional tools | Built-in support |
| **Performance** | Needs optimization for SSR | Optimized by default (Automatic Static Optimization) |
| **Flexibility** | Fully customizable | More opinionated but structured |
| **Community & Ecosystem** | Huge ecosystem, widely adopted | Growing rapidly, used for modern web apps |
| **Ease of Learning** | Simple for SPA development | More concepts to learn (SSR, API routes) |

**Final Verdict for Frontend**

* **React** is ideal for a highly flexible and fully client-side rendered SPA. It allows complete control over state management, routing, and component-based architecture.
* **Next.js** is better if you need **better SEO, SSR, SSG, and API routes** without setting up additional tools. However, it’s more opinionated.
* **Choice for this project:** **React** (due to its flexibility and focus on frontend development).

**Backend Comparison: Spring Boot vs. Node.js (NestJS)**

| **Feature** | **Spring Boot** | **NestJS (Node.js)** |
| --- | --- | --- |
| **Language** | Java | TypeScript (built on Express) |
| **Performance** | Heavier, but optimized for large applications | Lightweight and faster for I/O operations |
| **Scalability** | Excellent for enterprise-level applications | Scales well for microservices |
| **Security** | Built-in Spring Security, OAuth2, JWT support | Requires third-party libraries (e.g., Passport.js) |
| **Development Speed** | More boilerplate, but robust | Faster development with decorators and modules |
| **Database Support** | Works well with relational DBs (JPA, Hibernate) | Works well with NoSQL and SQL DBs |
| **Community & Ecosystem** | Mature, enterprise-focused | Rapidly growing in the JavaScript ecosystem |
| **Ease of Learning** | Steep learning curve | Easier for JS/TS developers |

**Final Verdict for Backend**

* **Spring Boot** is better for applications needing robust security, scalability, and deep database integration (especially relational databases like SQLite).
* **NestJS (Node.js)** is great for faster development, real-time applications, and microservices, but requires additional tools for security and complex logic.
* **Choice for this project:** **Spring Boot** (because it provides a stable, scalable, and well-structured backend for long-term maintainability).