Technology Stack Overview

Frontend: **Next.js**

- React-based framework with SSR, SSG, and API routes.
- High performance, scalability, and ease of use.

Backend: **Actix-Web**

- High-performance, asynchronous Rust framework.
- Ideal for high-concurrency applications.

Comparison Document

Frontend: Next.js vs. Others

Feature	Next.js	React	Angular	Vue.js
Key Functionalities	SSR, SSG, API routes, file-based routing	Component-based, flexible	Full-featured, TypeScript	Lightweight, easy to integrate
Performance	Excellent (SSR/SSG optimized)	Good (depends on setup)	Good (can be heavy)	Excellent (lightweight)
Scalability	Scales well for all sizes	Scales well	Scales well	Scales well
Ease of Use	Easy, minimal configuration	Flexible but requires setup	Steeper learning curve	Easy to learn
Tools	Built-in tools for dev, debug, and deploy (Vercel)	Requires additional tools (e.g., Vite)	Angular CLI	Vue CLI, Vite
Ecosystem	Large, backed by Vercel	Largest ecosystem	Strong enterprise	Growing ecosystem
Community	Strong, extensive documentation	Largest community	Smaller than React	Growing community
Pros	SSR/SSG, fast, easy to deploy	Flexible, large ecosystem	Full-featured, TypeScript	Lightweight, easy to learn
Cons	Less flexible than plain React	Requires setup for SSR	Heavy, complex	Smaller ecosystem

Best choice: **Next.js**

- Complete solution with SSR, SSG, and API routes.
- Best for performance, scalability, and ease of use.

Backend: Actix-Web vs. Others

Feature	Actix-Web	Express.js	Django	Spring Boot
Key Functionalities	Async, actor-based, HTTP/1.x/2, WebSockets	Minimalist, flexible	Full-featured, ORM, admin panel	Enterprise-level, Java-based
Performance	Extremely fast (Rust-based)	Good (Node.js)	Good (Python)	Good (Java)
Scalability	Excellent for high concurrency	Good	Good (not for high concurrency)	Good (resource-intensive)
Ease of Use	Steeper learning curve (Rust)	Easy	Easy	Complex
Tools	Cargo for dev, Rust compiler for debugging	NPM, Node.js tools	Django CLI	Maven/Gradle
Ecosystem	Growing (Rust libraries)	Large (NPM)	Large (Python)	Large (Java)

Feature	Actix-Web	Express.js	Django	Spring Boot
Pros	Extremely fast, memory-safe	Easy to use, large ecosystem	Full-featured, easy to use	Enterprise-level, strong typing
Cons	Steeper learning curve	Performance not as fast	Not for high concurrency	Resource-intensive

Best choice: **Actix-Web**

- Unmatched performance and scalability for high-concurrency apps.
- Memory-safe and lightweight, making it future-proof.

Database Comparison: PostgreSQL vs. Others

PostgreSQL vs. MySQL vs. MongoDB vs. SQLite

Feature	PostgreSQL	MySQL	MongoDB	SQLite
Type	Relational (SQL)	Relational (SQL)	NoSQL (Document-based)	Relational (SQL, Embedded)
Key Functionalities	ACID compliance, advanced SQL features, JSON support	ACID compliance, fast for read-heavy workloads	Flexible schema, JSON-like documents	Lightweight, serverless, file-based
Performance	Excellent for complex queries and large datasets	Fast for simple queries and read-heavy workloads	High performance for unstructured data	Fast for small-scale applications
Scalability	Scales well for large applications	Scales well but less robust for complex queries	Horizontally scalable (sharding)	Not designed for large-scale apps
Ease of Use	Moderate (requires more setup)	Easy to set up and use	Easy for unstructured data	Very easy (no server setup needed)
Tools	pgAdmin, psql CLI	MySQL Workbench, CLI	MongoDB Compass, CLI	SQLite CLI, DB Browser for SQLite
Ecosystem	Large, mature ecosystem	Large ecosystem, widely used	Growing ecosystem for NoSQL	Small but sufficient for embedded
Community	Strong, active community	Largest community	Growing NoSQL community	Small but dedicated community
Pros	Advanced features, ACID compliance, JSON support	Fast, easy to use, widely supported	Flexible schema, scalable for NoSQL	Lightweight, no server required
Cons	Steeper learning curve, heavier setup	Less advanced features than PostgreSQL	Not ACID-compliant by default	Not suitable for large-scale apps

---

PostgreSQL is the best choice beacuse:

- **advanced SQL features** (e.g., window functions, CTEs).
- **ACID compliance** and strong data integrity.
- **complex queries** and large datasets.
- **JSON support** alongside relational data.

Why PostgreSQL?

- **Advanced Features**: Supports complex queries, transactions, and JSON.
- **Scalability**: Handles large datasets and high concurrency well.
- **ACID Compliance**: Ensures data integrity and reliability.
- **Flexibility**: Combines relational and NoSQL features (e.g., JSONB).

Next.js (frontend), **Actix-Web (backend)** and **PostgreSQL** is the best choice because:

- **Next.js** offers a modern, performant, and scalable frontend solution.

- **Actix-Web** delivers top-tier backend performance and reliability.
- **PostgreSQL** delivers strong data integrity and JSON support
- Together, they form a robust, high-performance stack for any application size.