**🔹 Architecture Flow**

1. **Frontend (Next.js + Tailwind + Framer Motion)**
   * Renders the UI (server-side or static for SEO + speed)
   * Handles routing, animations, and user interactions
   * Sends API requests (via Axios/Fetch) to the backend
2. **Backend (FastAPI + SQLAlchemy + PostgreSQL)**
   * FastAPI exposes REST API endpoints
   * SQLAlchemy handles database queries with type safety
   * PostgreSQL stores all structured data
   * Authentication handled via **FastAPI Users / JWT**
3. **Authentication**
   * JWT tokens issued by FastAPI on login/register
   * Tokens stored securely (usually HttpOnly cookies or local storage depending on strategy)
   * Protected routes validated on backend using JWT middleware
4. **Hosting**
   * **Frontend:** Deployed on **Vercel** → global CDN, blazing-fast delivery
   * **Backend:** Deployed on **Railway** or **Render** → easy Python hosting
   * **Database:** Hosted on **Supabase** (managed PostgreSQL with free tier)
5. **Dev Tools**
   * **GitHub Actions:** Automates testing + deployment on every push
   * **Black:** Python code formatting
   * **ESLint + Prettier:** JavaScript/React linting and formatting
   * **Pytest:** Automated testing for backend logic

**🚀 Benefits of This Stack**

* **Modern UI** → Next.js + Tailwind = fast, responsive, mobile-friendly
* **Scalable Backend** → FastAPI is async + super fast
* **Strong Data Layer** → PostgreSQL + SQLAlchemy = enterprise-ready
* **Secure Auth** → JWT-based authentication
* **Free-Friendly** → Vercel, Railway, and Supabase all have generous free tiers
* **Professional DevOps** → GitHub Actions + linting = clean CI/CD pipeline