Project Overview

This project is a simple social media web application built using **Next.js** for both the front-end and back-end, written in **TypeScript**. It utilizes **Prisma** as the Object-Relational Mapper (ORM) and **PostgreSQL** as the database, hosted on **Supabase**. Authentication is handled through **Auth.js** (formerly known as NextAuth.js), with **Shadcn UI** as the component library and **Tailwind CSS** for styling.

Technology Stack

- **Frontend**: Next.js (TypeScript)
- **Backend**: Next.js API Routes (TypeScript)
- **Database**: PostgreSQL (via Supabase)
- **ORM**: Prisma
- **Authentication**: Auth.js (GitHub OAuth)
- **Styling**: Tailwind CSS, Shadcn UI

Folder Structure

The folder structure follows Next.js conventions.

Technologies Used

Next.js

Next.js is used for server-side rendering (SSR) and building the API backend. It provides fast development features like hot reloading and easy API creation via API routes.

2. TypeScript

TypeScript ensures strong typing, error checking, and better developer experience, providing type safety over JavaScript.

3. Prisma ORM

Prisma is the ORM used for database interactions. It provides type-safe queries and schema management for PostgreSQL databases.

4. PostgreSQL (via Supabase)

PostgreSQL is the relational database used to store all user, post, and interaction data, with **Supabase** acting as the hosting provider.

5. Auth.js (NextAuth.js)

Auth.js is used for authentication, with GitHub as the sole third-party authentication provider. It manages session handling and user authentication flows.

6. Tailwind CSS & Shadcn UI

Tailwind CSS is used for styling the UI in a utility-first manner. **Shadcn UI** provides pre-built, accessible components that complement Tailwind's approach.

Database Schema

The database schema is defined using **Prisma**. Below is an overview of the primary models:

- **User Model**: Represents the user entity with fields like id, name, email, and relationships to posts, likes, and bookmarks.
- **Post Model**: Represents social media posts, with fields for title, content, author, and relationships to users who like or bookmark the post.
- **Like Model**: Tracks likes on posts by linking users and posts through a many-to-many relationship.
- **Bookmark Model**: Tracks bookmarked posts by linking users and posts in a similar many-to-many relationship.

Authentication with GitHub (Auth.js)

The application uses **Auth.js** for GitHub OAuth-based authentication. It handles session management and user login via GitHub's OAuth flow. Key configurations include setting up GitHub as an OAuth provider, managing session tokens, and securing routes based on user roles.

API Routes

Next.js API routes handle the backend logic for posts, user management, and interactions. These routes perform actions like creating posts, fetching user data, and managing likes or bookmarks. Each API route operates on a defined path and interacts with the database using Prisma to persist or retrieve data.

Styles and UI

The UI is constructed with **Shadcn UI** components styled using **Tailwind CSS**. Shadcn UI provides a set of accessible, pre-built components such as buttons, forms, and cards. Tailwind CSS allows for a utility-first approach, where styles are applied directly to elements without the need for traditional CSS classes.

Deployment

- **Database**: Hosted on **Supabase** (PostgreSQL).
- **Application**: Deployed on **Vercel**, which is optimized for **Next.js** applications.
- **Environment Variables**: Set in the deployment environment (Vercel's dashboard or local .env file).

Conclusion

This documentation outlines the structure, technology stack, and design details of the social media web app. It is built with **TypeScript**, **Next.js**, and **Prisma**, using **Tailwind CSS** and **Shadcn UI** for styling. GitHub is used for authentication, and Supabase hosts the PostgreSQL database. This setup is designed to be scalable, secure, and easy to maintain.