DevScript Blog App - Architecture and Flow

Architecture

The application is divided into two major parts:

Frontend (React.js)

- Built with Vite for faster development.
- Routing is handled using react-router-dom .
- Authentication: On login/signup, the JWT token is saved in
- Role-based UI: The token is decoded to check if the user is

Main Pages:

- / → Home Page showing list of blogs.
- /blogs/:id → Detailed blog page.
- /login → Login page.
- /signup → Sign-up page.
- /admin → Admin dashboard (only for admins).

Backend (Node.js + Express.js)

Category	API Endpoint	Method	Description
Authentication APIs	`/api/auth/signup`	POST	Sign up a new user
Authentication APIs	`/api/auth/login`	POST	Log in an existing user
Blog APIs	`/api/blogs/`	GET	Fetch all blogs
Blog APIs	`/api/blogs/`	POST	Create a new blog (admin only)
Blog APIs	`/api/blogs/:id`	DELETE	Delete a blog by ID (admin only)
Blog APIs	`/api/blogs/:id`	GET	Retrieve a particular blog by ID

- **JWT**:Issued on successful login and sent in response.
- **MongoDB**: Stores user and blog data.

Middlewares:

- Token is verified on the server for protected routes.

Flow of the Application

---

1. User opens the app and lands on the Home page.
2. User can:

- Browse blogs after logging in.
- Click Login or Sign Up to authenticate.

3. On login/signup:

- Token is generated and stored in localStorage .
- Token contains id and role .

4. Navbar conditionally displays:

- Login/Signup options if not logged in.
- Dashboard + Logout if logged in.
- Dashboard is shown only if the role is admin .

5. Blogs are fetched via
6. Clicking on a blog navigates to a particular blog (/blog/:id).
7. Admin users can access dashboard.