Project Summary: AI-Powered Social Media Auto-Poster

Overview

This full-stack project allows users to generate AI-powered social media comments using a title input and selected tone. The user selects platforms (Twitter, LinkedIn, Instagram, etc.), generates comments using an AI API (OpenAI or Gemini), chooses one, and posts it to all selected platforms.

It uses:

ReactJS for the frontend

Spring Boot for the backend

MongoDB for persistent storage

Spring Security with Basic Auth

LocalStorage for session persistence

▯ Core Features:

User Signup/Login with Basic Auth

Token stored in localStorage for persistent login

AI-generated comments with tone options (Motivational, Funny, Polite, Rude, etc.)

Platform selection for posting (Twitter, LinkedIn, Instagram, etc.)

Store selected post and user action logs in MongoDB

▯ Project Workflow

User signs up or logs in.

Enters a title, selects tone and platforms.

Clicks "Generate Comments" – AI returns 3 comment suggestions.

User selects one comment.

Clicks "Post" – selected comment is posted to all selected platforms.

Comment + metadata (user, timestamp, platform, tone) is saved to MongoDB.

Login session persists using localStorage until logout.

□ Frontend Structure (ReactJS)

```
src/
□ □ □context/
□ □ □ □AuthContext.jsx // Manages login state and token
□ □ □pages/
□ □ □ □Login.jsx // User login page
□ □ □ Signup.jsx // User registration page

□ □ □ Home.jsx // Main UI for title input, tone & platform selection

□ □ components/

□ □ □ ToneSelector.jsx // Dropdown or toggle buttons for tone selection

□ □ □ PlatformSelector.jsx // Checkboxes for platforms like Twitter, LinkedIn, etc.

□ □ □ CommentOptions.jsx // Displays generated comments and lets user select one

□ □ App.jsx // Routing + context wrapper

□ □ index.js // Entry point
```

□ Backend Structure (Spring Boot + MongoDB)

```
src/main/java/com/ai/poster/

□ □ controller/
```

```
│   ├── AuthController.java // Handles signup & login (basic auth)

│   ├── PostController.java // Handles comment generation and posting

├── config/

│   ├── WebSecurityConfig.java // Spring Security config for Basic Auth

├── model/

│   ├── User.java // User entity

│   ├── PostRequest.java // Incoming request model

│   ├── PostHistory.java // Stored comment + platform + tone + timestamp

├── service/

│   ├── AIService.java // Connects with OpenAI or Gemini API

│   ├── SocialPostService.java // Handles posting to APIs (mocked or real)
```

```
□ □  □ UserService.java // Handles CRUD for user data

□  □ repository/

□ □  □ UserRepository.java

□ □  □ PostHistoryRepository.java
```

□ Summary Notes

Basic Auth is used instead of JWT.

The "token" is simply a base64-encoded username:password saved in localStorage.

AI API calls are made securely from the backend.

Posting to real platforms can be simulated or implemented using their APIs.

MongoDB logs all activity including user, comment, tone, and platform data.

Purple theme and final UI/UX polish will be applied after functionality is stable.

Let me know if you'd like a downloadable version or we can begin coding phase-by-phase now.