✅ Updated Project Summary

🔹 Project Overview

A full-stack AI-powered social media automation tool that:

- Accepts a **title** and **tone** (Motivational, Funny, Polite, Rude).
- Generates AI-based comment suggestions using **OpenAI or Gemini**.
- Allows users to select platforms (**Twitter, LinkedIn, Instagram**, etc.).
- Posts the selected comment to all chosen platforms.
- Stores user actions and metadata in **MongoDB**.

🔹 Tech Stack

- **Frontend**: ReactJS
- **Backend**: Spring Boot
- **Database**: MongoDB
- **Authentication**: Spring Security with Basic Auth
- **Session Management**: LocalStorage

🔹 Core Features

- User **Signup/Login** with Basic Auth
- Persistent login using **localStorage**
- AI-generated comments with **tone selection**
- Multi-platform posting
- Activity logging in MongoDB (user, comment, tone, platform, timestamp)

Workflow

1. User signs up or logs in.
2. Inputs a title, selects tone and platforms.
3. Clicks "Generate Comments" AI returns 3 suggestions.
4. User selects one and clicks "Post".
5. The comment is posted to all selected platforms.
6. Metadata is saved in MongoDB.
7. Session persists until logout.

Frontend Structure (ReactJS)

- AuthContext.jsx: Manages login state and token
- Login.jsx, Signup.jsx, Home.jsx: Pages
- ToneSelector.jsx, PlatformSelector.jsx, CommentOptions.jsx: UI components
- App.jsx, index.js: Routing and entry point

□ Backend Structure (Spring Boot + MongoDB)

- **Controllers**: AuthController, PostController
- **Config**: SpringSecurity.java using SecurityFilterChain
- **Models**: User, PostRequest, PostHistory
- **Services**
  :
  - AIService: Connects to OpenAI/Gemini
  - SocialPostService: Handles posting (mocked or real)
  - UserService: Manages user data

- **Repositories**: UserRepository, PostHistoryRepository

□ Spring Security Configuration

- Uses SecurityFilterChain for cleaner configuration
- Enables **Basic Auth** and disables **CSRF** for stateless APIs
- **CORS** configured for frontend integration
- Passwords hashed using BCryptPasswordEncoder
- Public endpoints: /auth/signup, /auth/login
- All other endpoints require authentication

□ Additional Notes

- AI API calls are securely made from the backend
- Posting to platforms can be mocked or real using APIs
- MongoDB logs all user activity
- UI will be polished with a **purple theme** after core functionality is stable