🚀 Assignment: Component & Multi-Component Generator Platform

📖 Overview

Build a **stateful**, AI-driven micro-frontend playground where authenticated users can iteratively generate, preview, tweak, and export React components (or full pages), with **all chat history and code edits preserved** across logins.

Assignment carries step wise marking & Full completion is not mandatory, but the project **must be live-hosted** to be evaluated.

🧰 Tech Stack to be Used

- **Backend:** Node.js with Express or NestJS (Candidate's choice of framework), **MongoDB or PostgreSQL** for data persistence.
- **Frontend:** React + **Next.js** (for server-side rendering and routing).
- **LLMs:** Candidate's choice (eg. use models like llama3/4, Gemma, Gemini 2.0 Flash lite, gpt4o-mini from open router for easier access and free tier usage).
- Redis (for caching session state).
- Vercel/Render/AWS/Heroku/Digital Ocean (for live hosting).

Candidates are free to use AI code completion tools like Windsurf, Cursor, Copilot etc to complete the assignment but at the same time they should be completely aware of the code generated by AI

🎯 Core Requirements (Requirements have been broken down in mandatory & good to have/optional requirements for evaluation your project should at least have the mandatory requirements)

1. **Authentication & Persistence (Mandatory)**
    - **Signup / Login** (email+password or OAuth).

    - **Load Previous Sessions**: list and select any saved "work"—including its full chat transcript, generated code, and UI-editor state.
    - **Create New Session**: initialize an empty slate in the DB.
2. **Conversational UI for Generation (Mandatory)**
    - **Side-panel chat** (text + image inputs) → submits prompts to your AI-backend.
    - AI responds with component code (**JSX/TSX + CSS**).
    - Render the generated component **live** in the central viewport as a **micro-frontend**.
3. **Code Inspection & Export (Mandatory)**
    - Below the preview: show **tabs** for "JSX/TSX" and "CSS" (syntax-highlighted).
    - Provide **Copy** and **Download (.zip)** buttons for the entire code (TSX/JSX + CSS).
4. **Iterative Refinement (Optional/Good to Have)**
    - Further prompts in-chat ("Make the button larger and red," etc.) patch the existing component.
    - Re-invoke AI, apply deltas, and re-render.
5. **Statefulness & Resume (Optional/Good to Have)**
    - **Auto-save** after every chat turn or UI-editor change.
    - On logout/login or page reload, users return to the exact same state:
        - **Full chat history**
        - **Latest generated code**
        - **Rendered micro-frontend preview**

🌟 Optional Bonus Requirements

6. **Interactive Property Editor** *(Bonus)* **(Optional/Good to Have)**

- **Element-Click → Property Panel**: click a rendered element (e.g. `<button>`) to open a floating panel with:
  - Size slider (padding/font-size)
  - Color picker (background/text)
  - Text input (content)
  - Border/shadow/radius controls
- **Two-Way Binding**: changing knobs mutates the underlying **JSX/TSX + CSS** live, re-renders the sandbox, and updates the "JSX/TSX" and "CSS" tabs.

7. **Chat-Driven Overrides** *(Bonus)* **(Optional/Good to Have)**
   - After selecting an element, allow an **input prompt** in the chat to target that element:

     "Make this button have 24px vertical padding, a teal-to-blue gradient, and uppercase text."

   - AI applies just that delta to the component code.

🛠️ Technical & Evaluation Checklist (with Weightage)

Step-wise Marking: Each checklist item carries independent marks. **Full completion is not mandatory**, but the project **must be live-hosted** to be evaluated.

Bonus points (up to **+45**) will be awarded for implementing features in **section 6 & 7** (Interactive Property Editor & Chat-Driven Overrides).

Area	What to Look For	Points
Auth & Backend	Secure JWT sessions; password hashing; REST/GraphQL endpoints; DB schema design.	**10 points**

State Management & Statefulness	Robust storage of chat+code+UI state (Redux/Zustand/Context + DB); auto-save & reload.	**15 points**
AI Integration	Clean prompt→response abstraction; streaming; error/loading handling.	**20 points**
Micro-Frontend Rendering	Secure sandbox (iframe); hot-reload without full refresh; isolation.	**10 points**
Code Editor & Export	Syntax highlighting; reliable copy/download; well-structured ZIP.	**10 points**
Iterative Workflow	Clear chat UX; proper turn delineation; incremental patches vs full replaces.	**10 points**
Persistence & Resume	Auto-save triggers; fast load of previous sessions; graceful error recovery.	**10 points**
Polish & Accessibility	Responsive design; keyboard support; ARIA roles; clear loading/empty/error states.	**10 points**
Bonus Features *(Optional)*	Interactive Property Editor & Chat-Driven Overrides (section 6 & 7).	**+45 points**

Total: 95 points (core) + 45 bonus = Max 140 points.

📦 Deliverables

1. **GitHub Repo** with README:

2. **Live Demo URL**
   - Project must be live-hosted (Vercel, Heroku, Render, AWS, etc.) for reviewers to interact directly.
3. **Write-Up** (Markdown):
   - Architecture diagram.
   - State-management & persistence strategy
   - Key decisions & trade-offs (e.g., sandboxing approach, auto-save logic).
