**Day 1: Introduction & Environment Setup**

* Explain **MERN stack & project structure**.
* Install **Node.js**, **MongoDB (or Supabase)**, and **Postman** for testing APIs.
* Set up a **React app**:
* Set up an **Express.js server** with a basic route.

**Day 2: UI Design with Tailwind CSS**

* Create a **simple UI**:
  + Input box for text
  + "Convert to Speech" button
  + Audio playback section
* Use **Tailwind CSS utility classes** for styling.

**Day 3: Backend Setup (Node.js & Express.js)**

* Install dependencies:
* Create a **basic API endpoint** in Express.js.
* Set up **CORS** to allow frontend-backend communication.

**Day 4: Database Setup (Supabase or MongoDB)**

* If using **Supabase**:
  + Create a **Supabase project**.
  + Set up a **table for storing text & audio links**.
  + Install Supabase SDK:
* If using **MongoDB**:
  + Set up a **MongoDB database with Mongoose**.
  + Create a **schema** for storing **text & generated audio URLs**.

**Day 5: Implementing the Text-to-Speech API**

* Choose a **TTS API** (Google Cloud, OpenAI, or Mozilla TTS).
* Set up API calls from the backend to **convert text into speech**.
* Return the **generated speech file URL** to the frontend.

**Day 6: Connecting Frontend to Backend**

* Use **Axios or Fetch API** to send text input from React to Express.
* Display **loading states** while the speech is being generated.
* Show the **generated audio file link** on the frontend.

**Day 7: Playing & Storing Audio**

* Implement an **HTML5 Audio player** to play the generated speech.
* Store audio URLs in **Supabase storage or MongoDB**.
* Display a list of **previously converted texts with audio files**.

**Week 2: Optimization, Deployment & Testing**

**Day 8: UI Enhancements & Tailwind Styling**

* Improve **UI design using Tailwind CSS**.
* Add **buttons, icons, and animations** for better user experience.

**Day 9: Implementing Error Handling**

* Handle **empty text input errors**.
* Show **proper error messages** if API calls fail.

**Day 10: User Authentication (Optional)**

* If needed, add **user authentication with Supabase Auth**.
* Allow users to **save their previous TTS conversions**.

**Day 11: Deployment Preparation**

* Deploy **backend (Express.js) on Render/Vercel**.
* Deploy **frontend (React) on Netlify/Vercel**.
* Ensure **Supabase/MongoDB connections work correctly**.

**Day 12: Final Testing & Debugging**

* Test **all features**.
* Check **backend security** (CORS, API limits).
* Fix **any UI/UX issues**.

**Day 13: Documentation & Code Cleanup**

* Write **README.md** with instructions.
* Clean up **unnecessary console logs and unused code**.

**Day 14: Final Presentation & Submission**

* Conduct a **final project walkthrough**.
* Discuss **key learnings & improvements**.
* Submit or showcase the project.

This **fast-paced 2-week plan** ensures the student builds a **functional TTS project** while learning **MERN, Supabase, and Tailwind CSS**. Let me know if you need any refinements! 🚀Top of Form

Bottom of Form