Prescripto.ai: Abstract Idea & Concept

Project Title

Prescripto.ai

Problem Statement

Managing multiple medical prescriptions can be a complex and often stressful task for individuals and their caregivers. This complexity can lead to missed doses, confusion regarding medication purposes and timings, potential for adverse drug interactions, and a general lack of accessible, clear information. Furthermore, for common, minor ailments, individuals often seek quick, reliable initial guidance on self-care or over-the-counter (OTC) options but lack an accessible, AI-guided starting point that also responsibly directs them to professional medical advice when necessary.

Solution Overview

Prescripto.ai is envisioned as an innovative, AI-powered React component, meticulously designed to align with the Empire UI track's emphasis on integrating AI to enhance interactivity and developing components that highlight unique, accessible, and user-first UX. This component, built using Tailwind CSS for styling and leveraging Radix UI primitives for foundational accessibility, aims to simplify medication management and offer preliminary guidance for minor health concerns. It is designed to be a reusable and customizable element for broader health applications.

Technology Stack

Frontend:

  - React: (Mandatory for the Empire UI Track)
  - TypeScript: (Recommended for type safety and better maintainability)
  - Styling: Tailwind CSS: This will be used for all custom styling, ensuring a utility-first approach for a responsive, customizable, and modern design consistent with the Empire UI ecosystem.
  - Accessibility Primitives: Radix UI: Unstyled, accessible UI primitives will be used as a foundation for building custom, accessible interactions within the component.
  - State Management: React Context API + useReducer, or a lightweight library like Zustand or Jotai for managing component state and communication with the backend.
- AI & Backend Integration:
  - AI Models & APIs:
    - Gemini API: Leveraged for advanced Natural Language Processing (NLP) tasks, such as parsing complex prescription details from user input, generating concise summaries for medications, and enhancing the conversational capabilities of the Minor Ailment Advisor.

    - Custom Small AI Model(s): A dedicated small-scale AI model (potentially rule-based or a lightweight ML model) will be developed and hosted on the backend for specific tasks like initial prescription validation, dosage sanity checks, or refined drug interaction lookups based on our curated dataset.
- Backend:
    - Node.js with Express.js: A dedicated backend server built with Express.js will be developed. Its primary roles include:
        - Serving as an API endpoint for the React frontend component.
        - Managing and proxying requests to the Gemini API.
        - Hosting and running the custom small AI model for prescription-related tasks.
        - Handling business logic related to AI processing and data retrieval before sending results back to the component.
- Client-Side AI Assistance (for UI Interactivity):
    - Basic input validation and pattern matching directly in the React component for immediate user feedback before backend calls

- Data Management:
    - Curated Datasets: JSON files or simple data structures will store foundational drug information, interaction rules, and ailment guidance for use by both the backend AI models and client-side logic. MongoDB will be used.

- Development Tools:
    - Vite or Create React App for frontend project setup and development server.
    - ESLint and Prettier for code linting and formatting.
    - Tools for backend development (e.g., Nodemon for Express.js).