Technical Proposal for PM Bob: AI-Powered Virtual Project Manager

1. Summary of the App's Purpose and Goals

PM Bob is an AI-powered virtual project manager designed to streamline the software development process for individual developers and small teams. Leveraging the Gemini GENai API and Trello API, PM Bob assists in planning, organizing, and executing projects with a focus on timely completion. Its personality-driven assistant modes make project management engaging and adaptive to user preferences.

2. Key Features and User Flows

- Character Modes: Multiple AI personas like PM Bob (strict), FlowJoe (calm), Agile Alex (agile-focused)
- Modular LLM Prompts: Customizable prompt templates for specific tasks and workflows
- Trello Integration: Auto-generation and syncing of project boards and tasks
- Local-First + Sync Later: Offline-first architecture with sync capabilities
- Notification System: Deadline reminders and task updates
- Dashboard UX: Clean and minimal interface with task overview, calendar, and AI suggestions
- Extendable Plugin/Script System: Add custom tools, scripts, and integrations

3. Suggested Tech Stack

Frontend:
- React + TypeScript
- Tailwind CSS for UI
- Redux or Zustand for state management

Backend:
- Node.js + Express or Fastify
- Gemini GENai API for LLM tasks
- Trello API for board and card management
- Plugin architecture via dynamic imports

Database:
- SQLite for local (desktop), PostgreSQL for cloud

- IndexedDB (via Dexie.js) for offline-first data in the browser

Infrastructure:
- Vite for frontend tooling
- Docker for containerization
- GitHub Actions for CI/CD
- Railway, Render, or Fly.io for backend hosting

4. High-Level System Architecture

- Frontend communicates with a centralized backend API
- Backend handles prompt generation, Trello interactions, and plugin execution
- Data sync service ensures offline-first capabilities
- Character personality module determines AI behavior via modular prompts

5. API Design / Data Model Outline

Entities:
- User: { id, name, preferences, characters }
- Project: { id, title, description, tasks, syncedTrelloBoardId }
- Task: { id, content, dueDate, status, assignedTo }
- PromptTemplate: { id, character, templateString, contextType }

Sample Endpoints:
- POST /projects/create
- POST /tasks/add
- POST /sync/trello
- POST /ai/prompt/generate

6. Deployment Strategy

- CI/CD: GitHub Actions for linting, testing, deployment
- Backend: Dockerized deployment to Render or Fly.io
- Frontend: Vercel or Netlify
- DB Sync: CRON or WebSocket based sync for desktop and web versions

7. Other Considerations

- Use tRPC or OpenAPI to keep API consistent across web, mobile, and bot clients
- Abstract core logic into a shared package used by all clients (web, desktop, bots)
- Implement AI safety layers: confirmation before destructive actions, logging prompts
- UX: Use friendly but firm tone in assistant interactions to match PM Bob’s persona