Aterges: Frontend Development Plan

Version: 1.0
Lead: Solopreneur (acting as Frontend Developer / Product Manager)
Primary Tool: Bolt (for initial scaffolding and component generation)

1. Core Objective

To build a modern, responsive, and intuitive web application that serves as the primary client interface for the Aterges platform. The frontend will provide a seamless user experience, from onboarding and authentication to interacting with the core conversational AI and managing account settings.

2. Guiding Principles

- **User-Centric Design:** Every component and workflow must be designed with simplicity and ease of use in mind, catering to both technical and non-technical users.
- **Performance First:** The application must be fast and responsive. We will leverage Next.js's capabilities for optimized builds and fast page loads.
- **Security by Default:** Adherence to modern web security practices is non-negotiable, especially in handling user authentication and data display.
- **Modularity and Scalability:** Components will be built to be reusable and the architecture will support the addition of new features and agents without major refactoring.

3. Core Technical Stack

This stack is final and should be implemented with the specified versions to ensure security and stability.

- **Framework: Next.js** (v15.4.0+)
- **Library: React** (v19.1.0+)
- **Language: TypeScript** (v5.8+)
- **UI Components: Shadcn/ui** (latest stable)
- **Deployment: Vercel**

4. Development Roadmap

Phase 0: The Skeleton & Authentication (Completed)

This phase, intended as the initial input for Bolt, establishes the foundational structure.

- **[✅] Repository Setup:** GitHub monorepo with a /frontend directory.
- **[✅] Page Scaffolding:** Creation of public pages (/, /login, /signup) and the protected dashboard (/app/dashboard).

- **[✅] Layouts:** PublicLayout and AppLayout components are defined.
- **[✅] Branding:** The Aterges logo, Geist font, and Vercel-inspired Light/Dark mode themes are implemented.
- **[✅] API Contract:** The frontend is ready to communicate with the backend API for authentication (/auth/signup, /auth/login, /api/me) using JWTs.

Phase 1: The Core Chat Experience (Current Focus)

This phase transforms the empty dashboard into a functional product.

- **[✅] Chat UI Components:** Creation of ChatInputForm, ChatMessage (with Markdown rendering), and ChatHistory components.
- **[✅] State Management:** The ChatInterface component manages the conversation state (messages, isLoading).
- **[✅] API Integration:** The handleSubmit function is wired to call the POST /api/query endpoint, sending the user's prompt and displaying the AI's response.
- **[✅] UX Enhancements:** Loading indicators, disabled states, and error handling (via toast notifications) are implemented.

Phase 2: Core Application Features (Next Steps)

Once the backend can provide the necessary data, the frontend will expand to include essential SaaS features.

- **[ ] Build "Account Settings" Page (/app/settings):**
  - **Profile Tab:** Display user email and allow name changes.
  - **Security Tab:** Implement the password change form.
  - **Plan & Billing Tab:** Display the user's current plan and a placeholder for Stripe integration.
  - **API Dependency:** Requires simple CRUD endpoints for user profile management.
- **[ ] Build "Integrations" Page (/app/integrations):**
  - **Goal:** Enable the "Bring Your Own Key" (BYOK) model.
  - **UI:** A view to list connected data sources (e.g., "Google Cloud Project - *project-123*").
  - **Workflow:** An "Add Integration" form where users can name a connection and paste their service-account.json credentials.
  - **API Dependency:** Requires CRUD endpoints for managing user integrations.
- **[ ] Implement Conversation History:**
  - **Goal:** Make conversations persistent.
  - **UI:** Modify AppLayout to include a sidebar listing past chats.
  - **Functionality:** Allow users to start new chats, load old ones, rename them, and delete them.

    - **API Dependency:** This is the most complex dependency, requiring a full set of CRUD endpoints for managing chat sessions and messages.
- **[ ] Quality of Life (QoL) Improvements:**
    - Implement "Copy" button for code blocks in chat messages.
    - Implement "Regenerate Response" button in the chat interface.

5. Key Deliverables

- A fully responsive and production-ready web application deployed on Vercel.
- A component library built with Shadcn/ui that is consistent with the Aterges brand identity.
- A secure and seamless user authentication and session management experience.
- A robust and user-friendly chat interface that serves as the core product experience.
- Complete implementation of Account Settings, Integrations management, and Conversation History.