CalendrAI: Supplemental Implementation Document

Project Overview

CalendrAI is a modern, AI-powered calendar and task management application that seamlessly integrates natural language processing (NLP) with traditional calendar functionality. Users can manage schedules through both standard CRUD operations and AI-assisted natural language commands.

Technology Stack

Frontend

- **Next.js** — React framework for server-side rendering, routing, and optimization
- **Tailwind CSS** — Utility-first CSS framework for a custom, responsive pastel-themed UI
- **React** — JavaScript library for building interactive UIs
- **Shadcn UI** — Accessible, customizable component library with a consistent design system
- Additional supporting libraries for enhanced functionality like Radix.

Backend

- **Next.js API Routes** — Serverless functions for backend processes

- **Firebase** — Cloud platform for database, authentication, and hosting. Chosen because it was easy and quick to set up.

    - **Firestore** — NoSQL database to store user data, calendar events, and tasks

    - **Firebase Authentication** — Secure authentication with Google OAuth integration

AI Integration

- **OpenAI API**

    - GPT-4o-mini model integration for natural language understanding

    - Function calling for task/event management based on user commands

---

Core Architecture

Authentication System

CalendrAI uses Google Authentication integrated via **NextAuth.js** with Firebase:

- **Implementation** — Firebase Authentication with Google OAuth via NextAuth.js

- **Security** — Middleware-based route protection and validation

- **Session Management** — Cookie-based sessions with token verification. Will expire eventually.

Database Structure

Firestore database architecture:

- **Users Collection**
    - User ID (Google Auth)
    - Name
    - Email
    - Email verification status
    - **Notes Sub-collection** — Stores user events and tasks
        - Description
        - Start and end times
        - Categories, tags, priorities
        - Color coding
    - **Usage Sub-collection** — Tracks:
        - Credits remaining
        - CreatedAt timestamps

    - **Conversations Collection** — Stores AI chat histories
        - User messages
        - AI responses
        - Associated timestamps and actions

API Security

Robust measures to secure APIs:

- **Middleware Protection**
    - Verifies secure JWT ID tokens from the frontend in the backend before conducting any processes.
    - Validates user authorization
    - Blocks unauthorized access
- **Credits System**
    - Initial credit balance per user
    - Credits consumed based on token usage
    - Real-time updates to prevent abuse of OpenAI API

Key Features

Calendar System

Views and Navigation

- **Week View** — Interactive weekly calendar with time slots
- **Day Navigation** — Move between days and weeks easily
- **Today Focus** — Jump to the current day's events with a red line.

Event Management

- **Create Events**
  - Title, description
  - Start and end times
  - Categories, priorities, color coding, tags
- **Edit Events**
  - Modal-based editing
  - Drag-and-drop repositioning
  - Resizing event durations
- **Delete Events**
- **Search and Filter Events**
  - By text, category, tag, or time period

AI Integration

Natural Language Processing

OpenAI integration enables AI-driven schedule management:

- **Command Parsing**
  - Examples:
    - "Schedule a meeting with John tomorrow at 3 PM."
    - "Move my dentist appointment to Friday and add a work session this evening."
    - "What's on my calendar for next week?"
- **Agent Loop System**
  - Creates, updates, deletes, and searches tasks/events until it fulfills all of its' goals.
- **Contextual Understanding**
  - Maintains conversation history for better context handling

Implementation Details

- API Handler: /api/ai/route.ts
- Structured tool definitions for function calling

- Credit-based tracking for OpenAI API token usage

Caching Strategy

Performance optimization through:

- **Client-Side Caching** — In-memory caching for quick access
- **Data Prefetching** — Load probable next-step data
- **Debounced Updates** — Limit excessive writes to the database during rapid user interactions

---

Frontend Components

Main UI Components

- **WeekCalendar** — Core weekly calendar visualization
- **CommandBar** — Natural language input field(expandable and supports audio input)
- **ChatSidebar** — Conversational AI interface
- **TaskList** — Task management with filtering(a simple non-calendar view)

User Interface Features

- **Interactive Calendar Events** — Drag, drop, and resize events
- **Responsive Design** — Fully functional across devices
- **Dark/Light Mode Toggle**

Performance Optimizations

- **Code Splitting** — Reduces initial load times
- **Debounced Updates** — Prevents unnecessary re-renders
- **Optimized Rendering** — Using `useMemo` and `useCallback` hooks
- **Firestore Optimization** — Caching and lazy loading implemented to reduce read/writes.

Security Considerations

- **API Route Protection** — Authentication checks on all endpoints
- **Input Validation** — Sanitize user input before processing
- **Secure Environment Variables** — Credentials kept safe
- **Token Verification** — Firebase ID tokens validated on every request