PLOS (Personal Life Operating System) - Technical Architecture

Core Tech Stack for Individual Development

Frontend

- **Framework**: Next.js 14 (App Router) - provides both frontend and API routes in one project
- **Styling**: TailwindCSS with shadcn/ui components
- **Animations**: Framer Motion for smooth transitions
- **State Management**: Zustand (lightweight, easy to maintain)
- **Forms**: React Hook Form + Zod validation
- **Data Visualization**: Recharts or Chart.js for dashboard analytics

Backend/API

- **Runtime**: Next.js API routes (serverless functions)
- **Database**: Supabase (PostgreSQL + authentication + storage)
- **File Storage**: Supabase Storage (for journal entries, vision boards)
- **Authentication**: Supabase Auth (email, OAuth options)

AI Integration

- **Primary AI**: OpenAI API (GPT-4o or 3.5 Turbo for cost efficiency)
- **Embeddings**: OpenAI embeddings for semantic search in journals
- **Local AI**: Transformers.js for lightweight client-side tasks
- **Sentiment Analysis**: Either OpenAI API or Hugging Face inference API

Progressive Enhancement

- **PWA**: Next-PWA for installable app experience
- **Offline Support**: LocalStorage/IndexedDB for offline data caching
- **Sync**: Custom sync logic when connection is restored

Module-by-Module Technical Implementation

1. Dashboard Page

- **Data Aggregation**: Custom hooks to fetch from multiple endpoints
- **State Persistence**: Zustand store with middleware for persistence

- **Components**:
  - StatCard (reusable for various metrics)
  - MiniChart (for sparklines/small visualizations)
  - QuoteDisplay (with animation)
  - ActivitySummary (grid layout with stats)

2. Physical Health Tracker

- **Data Sources**:
  - CSV parser using PapaParse for fitness data imports
  - API Routes for CRUD operations on health metrics
- **Visualization**:
  - Line/bar charts with Recharts
  - Daily activity timeline component
- **AI Integration**:
  - OpenAI API prompt for health recommendations based on patterns
  - Schedule with node-cron for weekly insights

3. Mental Health Companion

- **Mood Tracking**:
  - Supabase table for mood entries
  - Calendar heatmap visualization
- **AI Chat**:
  - Streaming response implementation with OpenAI
  - Prompt engineering with context from mood history
  - Chat UI with typing indicators and history
- **Meditation Features**:
  - Web Audio API for guided sessions
  - Framer Motion for breathing visualizations

4. Nutrition & Diet Page

- **Food Logging**:
  - Autocomplete component with nutrition database
  - Image analysis with OpenAI Vision API

- **Data Storage**:
  - Supabase tables for meals and nutrition facts
- **Visualization**:
  - Stacked bar charts for macronutrients
  - Progress circles for daily targets
- **AI Integration**:
  - Meal suggestions based on nutritional gaps
  - Recipe generation with dietary preferences

5. Family & Social Life

- **Event Tracking**:
  - Integration with React Calendar component
  - Supabase tables for social activities
- **Time Analysis**:
  - Custom visualization for time distribution
  - Streak tracking with localStorage backup
- **AI Features**:
  - Activity recommendation engine with OpenAI
  - Relationship insight generation

6. Personal Goals & Planner

- **SMART Goals**:
  - React DnD (drag and drop) for task reordering
  - Progress tracking with visual indicators
- **Task Management**:
  - Local state with server sync (offline-first approach)
  - Deadline notifications using web notifications API
- **Vision Board**:
  - Masonry layout with react-masonry-css
  - Image upload with client-side compression

7. Life Journal

- **Editor**:

    - React-Markdown or TipTap for rich editing
    - Autosave with debounce
- **Analysis**:
    - Sentiment analysis through OpenAI API
    - Topic extraction for journaling insights
- **Search**:
    - Vector database approach with OpenAI embeddings
    - Full-text search fallback

Data Architecture

Supabase Tables

1. `users` - Core user profile data
2. `health_metrics` - Daily health statistics
3. `mood_entries` - Mental health tracking
4. `nutrition_logs` - Food and meal tracking
5. `social_events` - Family and social activities
6. `goals` - SMART goals and tasks
7. `journal_entries` - Life journal with rich text
8. `ai_insights` - Stored AI-generated insights

Data Relationships

- One-to-many relationship between users and all tracking tables
- Proper indexing for quick dashboard aggregation
- JSON fields for flexible schema evolution

API Routes Structure

```
/api
  /auth - Authentication endpoints
  /health - Physical health data
  /mental - Mental health and mood
  /nutrition - Food and diet tracking
  /social - Family and social events
  /goals - Goal and task management
  /journal - Journal entries and analysis
  /ai - AI-powered insights and recommendations
```

AI Integration Patterns

1. Direct API Calls

- For real-time chat interactions
- For on-demand analysis

2. Background Processing

- Scheduled insights generation
- Trend analysis overnight

3. Client-Side Processing

- Simple sentiment analysis
- Quick recommendations without API call

4. Hybrid Approach

- Cache common recommendations
- Use streaming for longer responses

Development Approach for Solo Developer

Phase 1: Core Infrastructure (2-3 weeks)

- Set up Next.js with Tailwind and Supabase
- Authentication flow and user profiles
- Basic dashboard with mock data

Phase 2: Module Implementation (8-10 weeks)

- Implement each module one by one, starting with highest value

- Prioritize: Dashboard → Journal → Goals → Health
- Focus on core features before AI integration

Phase 3: AI Enhancement (4-6 weeks)

- Add OpenAI integration for key features
- Implement recommendation systems
- Add sentiment analysis and insights

Phase 4: Polish & Performance (2-3 weeks)

- Optimize loading performance
- Add animations and transitions
- Implement PWA features

Cost Optimization for Individual Developer

OpenAI API Usage

- Implement caching for repeated requests
- Use cheaper models (3.5 Turbo) for non-critical features
- Batch process insights during off-peak hours

Supabase Costs

- Stay within free tier limits initially (10,000 users)
- Implement efficient queries to minimize database operations
- Use edge functions for global performance

Hosting Options

- Vercel Hobby tier for Next.js hosting
- Consider Cloudflare Pages + Workers for edge computing
- Implement proper caching strategies

Mobile-First Implementation

Responsive Design

- Design mobile layouts first, then scale up
- Use viewport units and flexible grids

- Test on various device sizes

Touch Optimization

- Larger touch targets (min 44×44px)
- Swipe gestures with Framer Motion
- Bottom navigation pattern for mobile

PWA Features

- Service worker for offline functionality
- Add to home screen capability
- Push notifications for reminders

Data Security Considerations

- Encrypt sensitive data at rest
- Implement proper row-level security in Supabase
- Add 2FA for user accounts
- Regular backups for user data

Testing Strategy

- Component tests with React Testing Library
- E2E tests with Playwright for critical flows
- Manual testing on multiple devices
- Synthetic monitoring for production

Deployment & CI/CD

- GitHub Actions for automated testing
- Vercel for preview deployments
- Feature flags for gradual rollout
- Analytics integration for usage monitoring