

AI-Powered Journaling Companion

Case Study

Pooja Munishamappa Raju

February 2026

Contents

1	Project Overview	3
1.1	Key Objectives	3
2	Architecture Design	3
2.1	High-Level Architecture	3
2.2	Design Principles	4
3	Tech Stack	4
3.1	Frontend	4
3.2	Backend	4
3.3	AI / ML Services	4
4	Docker Containerization	5
4.1	Container Architecture	5
4.2	Services	5
4.2.1	PostgreSQL	5
4.2.2	Backend API	5
4.2.3	Frontend Application	5
4.3	Dockerfiles	6
4.3.1	Frontend Dockerfile	6
4.3.2	Backend Dockerfile	6
5	Core Features	6
5.1	Adaptive Entry Modes	6
5.1.1	Unload My Mind	6
5.1.2	Soft Start	6
5.1.3	The Clarity Corner	7
5.2	AI-Powered Analysis	7

6	AI Pipeline	7
6.1	Cognitive Bias Types	7
7	Database Schema	8
7.1	Users Table	8
7.2	Journal Entries Table	8
8	API Reference	9
8.1	User Endpoints	9
8.2	Entry Endpoints	9
8.3	Analytics Endpoints	9
9	Future Enhancements	9
10	Conclusion	9

1 Project Overview

The AI-Powered Journaling Companion is a mental health-focused journaling application that combines voice-to-text input, cognitive bias detection, and comprehensive analytics to help users develop self-awareness through reflective writing.

1.1 Key Objectives

- **Reduce Friction:** Multiple entry modes such as voice and soft-start prompts to encourage daily journaling
- **Cognitive Support:** Detect thinking patterns and provide gentle mirror-style reframes
- **Insight Generation:** Transform raw journal entries into actionable self-awareness
- **Progress Tracking:** Visualize mood trends, patterns, and personal growth

2 Architecture Design

2.1 High-Level Architecture

```
Client Layer
- Welcome Screen
- Adaptive Cards
- Journal Modes (Unload / Soft Start)
- Gallery (Visualization + Wiki)

API Layer
- Users Route
- Entries Route
- Analytics Route
- Wiki Route

Service Layer
- AI Pipeline
  - Gemini Analysis
  - Embeddings Service
  - Cognitive Bias Detection

Data Layer
- PostgreSQL
  - users
  - journal_entries
  - embeddings
  - analytics
```

2.2 Design Principles

1. **Separation of Concerns:** Clear boundaries between frontend, API, services, and data layers
2. **Monorepo Structure:** Frontend and backend maintained in a single repository
3. **AI-First Processing:** Every journal entry flows through the AI pipeline before storage
4. **Privacy-Focused:** Journal entries processed ephemeraly for insights only—never stored, logged, or used for training. All trend analysis stored locally on user’s device. Raw text never shared beyond active session.
5. **Code Quality:** Modular service-layer architecture, comprehensive error handling, input validation on all endpoints, ESLint enforcement, and test coverage for critical paths.

3 Tech Stack

3.1 Frontend

Technology	Version	Purpose
Next.js	16.1.6	React framework with App Router
React	19.2.3	UI library
Material UI	7.3.7	Component library
Emotion	12.x	CSS-in-JS styling
Recharts	3.7.0	Data visualization
Lucide React	0.563.0	Icon library
Web Speech API	Native	Voice-to-text transcription

3.2 Backend

Technology	Version	Purpose
Express.js	5.2.1	REST API framework
PostgreSQL	16.x	Relational database
pg	8.18.0	PostgreSQL client
UUID	13.0.0	Unique ID generation
CORS	2.8.6	Cross-origin middleware

3.3 AI / ML Services

Technology	Version	Purpose
Google Generative AI	0.24.1	Gemini API integration
gemini-2.5-flash	–	Journal analysis model
text-embedding-004	–	Semantic embeddings

4 Docker Containerization

The application is fully containerized using Docker and Docker Compose for consistent development and deployment.

4.1 Container Architecture

```
Docker Compose Stack
- frontend (port 3000)
- backend (port 3001)
- postgres (port 5432)
- shared bridge network
- persistent volumes
```

4.2 Services

4.2.1 PostgreSQL

- Image: pgvector/pgvector:pg16
- Persistent storage using volumes
- Vector embedding support
- Automatic initialization scripts

4.2.2 Backend API

- Node.js 20 Alpine
- Hot reload with nodemon
- Depends on healthy database container

4.2.3 Frontend Application

- Next.js dev server
- Hot reload via volume mounts
- Connects to backend via environment variable

4.3 Dockerfiles

4.3.1 Frontend Dockerfile

```
FROM node:20-alpine
WORKDIR /app
COPY package*.json ./
RUN npm install
COPY . .
ENV NODE_ENV=development
EXPOSE 3000
CMD ["npm", "run", "dev"]
```

4.3.2 Backend Dockerfile

```
FROM node:20-alpine
WORKDIR /app
COPY package*.json ./
RUN npm install
COPY . .
EXPOSE 3001
CMD ["npm", "run", "dev"]
```

5 Core Features

5.1 Adaptive Entry Modes

5.1.1 Unload My Mind

- Voice-to-text journaling
- Real-time transcript display
- Immediate AI analysis
- Cognitive bias detection with mirror reframes

5.1.2 Soft Start

- Guided journaling with floating prompts
- Inactivity-triggered suggestions
- Gentle nudges for empty entries
- Book-like visual aesthetic

5.1.3 The Clarity Corner

1. Wave Timeline (Analytics Visualizations)

The Wave Timeline is an interactive mood visualization that displays sentiment trends over time using a smooth, flowing wave chart.

2. Indexed Domain (Journal Wiki Access)

The Journal Index is a personal wiki system that organizes journal entries by life domains (thematic tags), enabling historical exploration by domain.

5.2 AI-Powered Analysis

Each journal entry produces:

- First-person summary
- Keywords extraction
- Domain and emotion tags
- Sentiment score from -1.0 to +1.0
- Cognitive bias detection with reframes

6 AI Pipeline

Raw Journal Entry

Gemini Analysis

- Summarization
- Keyword extraction
- Tag classification
- Sentiment scoring
- Cognitive bias detection

Text Embeddings (768 dimensions)

PostgreSQL Storage

6.1 Cognitive Bias Types

Bias Type	Description
Catastrophizing	Assuming the worst outcome

All-or-nothing	Black and white thinking
Overgeneralization	Broad conclusions from single events
Should Statements	Rigid internal rules
Personalization	Excessive self-blame
Negative Prediction	Expecting negative outcomes without evidence
Mental Filter	Focusing only on negatives
Labeling	Defining identity by a single trait

7 Database Schema

7.1 Users Table

```
CREATE TABLE users (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  name VARCHAR(255) NOT NULL,
  email VARCHAR(255) UNIQUE NOT NULL,
  timezone VARCHAR(50) DEFAULT 'UTC',
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

7.2 Journal Entries Table

```
CREATE TABLE journal_entries (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  user_id UUID REFERENCES users(id) ON DELETE CASCADE,
  raw_text TEXT NOT NULL,
  summary TEXT,
  keywords TEXT[],
  tags TEXT[],
  sentiment_score DECIMAL(3,2),
  cognitive_biases JSONB,
  embedding DECIMAL[],
  entry_mode VARCHAR(50),
  duration_seconds INTEGER,
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```


8 API Reference

8.1 User Endpoints

- `POST /api/users` – Create or retrieve user
- `GET /api/users/:userId` – Fetch user details

8.2 Entry Endpoints

- `POST /entries` – Create journal entry with AI processing

8.3 Analytics Endpoints

- `GET /analytics/dashboard?userId=` – Dashboard statistics

9 Future Enhancements

- **AI Companion Chat**
- **Community and Therapist Features**
- **Advanced Analytics and Trend Discovery**
- **External Integrations**
- **Guided journaling programs**

10 Conclusion

This project demonstrates a thoughtful, AI-first approach to mental health journaling. The architecture balances user privacy, extensibility, and insight generation while remaining practical for real-world deployment and future scaling.