

Mood Journal: An Intelligent Journal with Deep Learning-Powered Sentiment Analysis

Ahmad Pataparla (2201017), Anushka Srivastava (2201030),
Kollipara Sai Surya Narayana (2201109), Kondragunta Surya Teja (2201111)

November 7, 2025

Introduction

Mental wellness applications typically rely on manual mood tracking, which introduces subjective bias and lacks automated pattern recognition. This project implements a Progressive Web Application (PWA) using SvelteKit with integrated sentiment analysis, conversational AI, and wellness recommendation features.

The system uses DistilBERT for 3-class sentiment classification (positive, negative, neutral), keyword-based similarity matching for entry retrieval, and Groq's Llama 3.1 8B model for AI-assisted journaling. The implementation includes offline functionality via service workers, gamification through 22 achievements, and analytics visualizations.

Problem Statement

This project addresses the following objectives: (1) automate mood classification to reduce manual input burden; (2) implement pattern recognition through NLP techniques; (3) provide AI-assisted journaling features; (4) enable offline functionality through PWA architecture; (5) add gamification elements for user engagement.

Objectives: Develop a full-stack PWA with authentication, implement sentiment analysis using DistilBERT with lexicon-based fallback, build an AI companion using Groq's Llama 3.1 8B model, design gamification system with 22 achievements, achieve PWA compliance with offline functionality, and implement testing with security features.

Solution Approach

Technology Stack: Frontend: SvelteKit 2.43 with Svelte 5, TypeScript 5.9, shadcn-svelte UI, Tailwind CSS 4.1, Chart.js 4.5. Backend: Node.js with SvelteKit SSR, Drizzle ORM 0.44, SQLite, Argon2 password hashing. AI/ML: Python Flask with DistilBERT (lxyuan/distilbert-base-multilingual-cased-sentiments-student), Groq API with Llama 3.1 8B Instant.

Sentiment Analysis: DistilBERT multilingual model performs 3-class classification (positive, negative, neutral) with ~350ms inference time on CPU. Lexicon-based classifier (87 keywords, 168 stopwords) serves as fallback with ~10–15ms processing time when Python server unavailable.

NLP Pipeline: TF-IDF-inspired keyword extraction (653-line implementation) identifies top 5 thematic keywords per entry. Keywords stored in database for future analysis.

AI Companion: Groq Llama 3.1 8B Instant model generates responses (2-4 sentences) using 10-message conversation history for context. System prompt includes journal entry content, configured with temperature 0.7 and max 300 tokens. Fallback mechanism returns default response on API failures.

Gamification: 22 achievements across 5 categories (Milestones, Consistency, Quality, Exploration, Wellness) with XP system (Level N requires $N \times 100$ XP). Includes progress bars, notifications, and streak tracking.

PWA Implementation: Service worker caches static assets with network-first strategy for API responses. 162-line manifest.json with 9 icon sizes for installation on iOS, Android, Windows, Mac, Linux. Responsive design with mobile bottom navigation, dark mode, and offline functionality.

Security: Argon2id password hashing, session management (httpOnly cookies, 30-day expiration), account lockout (5 failed attempts, 15-minute duration), rate limiting (100 req/15min general, 20 req/15min auth), local SQLite storage.

Experimental Results

Sentiment Analysis Performance: DistilBERT processes first 512 tokens with observed ~ 300 – 400 ms inference time on CPU. Lexicon fallback (87 keywords, 60% threshold) processes in ~ 10 – 15 ms. Dual-mode architecture with automatic fallback on timeout (2s health check, 10s analysis) or API errors.

AI Companion Characteristics: Groq Llama 3.1 8B generates responses typically 100–150 tokens with configured parameters (temperature 0.7, max tokens 300). 10-second response timeout with fallback. Conversation history retrieval < 50 ms from SQLite.

NLP Pipeline Efficiency: Keyword extraction processes entries in milliseconds. Keywords stored per entry for filtering and search functionality.

System Implementation: PWA with manifest (9 icon sizes) and service worker for offline caching. Responsive design with breakpoint-based layouts. Dark mode with localStorage persistence. Cross-platform installability on iOS, Android, Windows, Mac, Linux.

Feature Completeness: Markdown journal entries with sentiment analysis, Chart.js analytics dashboard, AI companion with Groq API key, multi-format export (PDF via jsPDF, JSON, Markdown), 22 achievements across 5 categories.

Testing Coverage: Vitest unit tests for 653-line NLP pipeline (sentiment analysis, keyword extraction), authentication flows (login, registration, password hashing), database operations. Playwright E2E tests for user journeys (registration, entry creation), mobile responsive testing, PWA functionality, AI conversations, export functionality.

Conclusion

This project implements a mental wellness application with the following technical components: full-stack SvelteKit PWA with TypeScript, SQLite with Drizzle ORM, secure authentication (Google OAuth + Argon2), testing suite (Vitest + Playwright), Python Flask with DistilBERT for sentiment analysis, dual-mode sentiment classification with lexicon fallback, NLP pipeline (TF-IDF keyword extraction, Jaccard similarity), AI companion (Groq Llama 3.1 8B), and gamification (22 achievements, XP system).

Technical Contributions: (1) Hybrid sentiment analysis combining DistilBERT with lexicon fallback for reliability; (2) NLP pipeline without model training requirements; (3) Local SQLite architecture for simplified deployment; (4) PWA with offline support and cross-platform installation; (5) Gamification system for user engagement.

Applications: Automated sentiment tracking, keyword-based entry tagging, AI-assisted journaling. Exportable data formats (PDF, JSON, Markdown) for sharing with healthcare providers. Sentiment visualizations for tracking emotional states.

Code Repository: <https://github.com/knightofcookies/mood-journal-app>