Software Requirements Specification: MindScribe

Version: 1.0 **Date:** November 8, 2025

---

1. Introduction

1.1 Purpose

This document specifies the software requirements for **MindScribe**, a web and mobile application designed for personal journaling and automated mood analysis. It provides a private space for users to record their thoughts, which are then analyzed by a Natural Language Processing (NLP) model to provide sentiment insights.

1.2 Intended Audience

This SRS is intended for project stakeholders, including the developer (for building and maintenance) and any potential testers (for validation).

1.3 Scope

The MindScribe application will be a full-stack project consisting of:

- **A React Frontend:** A responsive web dashboard that serves as the primary user interface.
- **A Django Backend:** A RESTful API to manage users, entries, and analysis.

The system **will** provide:

- Secure user registration and login.
- A form to create, submit, and save new journal entries.
- *Instant* server-side sentiment analysis of each new entry.
- A data visualization dashboard showing mood trends, distribution, and recent entries.

The system **will not** provide:

- Social sharing or public "feeds."
- Professional medical advice or diagnosis.
- Entry editing or deletion (in this version).

- A native mobile application (in this version, though the frontend is mobile-responsive).

1.4 Definitions & Acronyms

- **SRS:** Software Requirements Specification
- **NLP:** Natural Language Processing
- **API:** Application Programming Interface
- **REST:** REpresentational State Transfer
- **JWT:** JSON Web Token (for authentication)
- **Sentiment Polarity:** A score (typically -1.0 to +1.0) indicating the negative or positive tone of a text.
- **Sentiment Label:** A human-readable category (e.g., "Positive", "Negative", "Neutral").

---

2. Overall Description

2.1 Product Perspective

MindScribe is a new, standalone web application. It is composed of two primary, decoupled services:

1. **Frontend (Static Site):** A React application that runs in the user's browser.
2. **Backend (Web Service):** A Python/Django API that serves data and performs analysis. These services communicate via a JSON-based REST API.

2.2 Product Features

The high-level features of MindScribe are:

1. **User Authentication:** Securely create and log in to a personal account.
2. **Entry Creation:** A simple, private form to write and save journal entries.
3. **Instant NLP Analysis:** Server-side sentiment analysis is performed immediately on submission.
4. **Data Visualization:** A personal dashboard displays all insights and recent entries.

2.3 User Classes and Characteristics

There is one primary user class:

- **Journaler (User):** A general user who wants a private, digital space to write down their thoughts and gain a better understanding of their emotional patterns over time. This user is assumed to be web-literate and is using a modern web browser.

2.4 Operating Environment

- **Frontend:** The application shall be accessible from any modern web browser (e.g., Chrome, Firefox, Safari, Edge) on desktop or mobile devices.
- **Backend:** The backend service is deployed on **Render** as a web service, running Python and `gunicorn`.
- **Database:** The production database is **PostgreSQL**, hosted by Render.

2.5 Design and Implementation Constraints

- **Backend:** Must be built with Python, Django, and Django REST Framework.
- **Frontend:** Must be built with React.js.
- **NLP Model:** Must use `spaCy` as the core NLP library, combined with `spacytextblob` for sentiment scoring.
- **Deployment:** The entire application (frontend, backend, database) must be deployable on Render.

---

3. System Features (Functional Requirements)

3.1 Feature 1: User Authentication

- **FR-1.1:** The system shall provide a form for new users to register with a unique username and a password.
- **FR-1.2:** The system shall securely hash and store user passwords.
- **FR-1.3:** The system shall provide a form for existing users to log in.

- **FR-1.4:** Upon successful login or registration, the API shall return a JWT authorization token to the client.
- **FR-1.5:** The client shall store this token (e.g., in `localStorage`) to authenticate subsequent requests.
- **FR-1.6:** The client shall provide a "Log Out" button that deletes the token from local storage.
- **FR-1.7:** All API endpoints, *except* for `/api/register/` and `/api/login/`, shall be protected and require a valid JWT token.
- **FR-1.8:** A user shall *only* be able to access their own data.

3.2 Feature 2: Journal Entry Management

- **FR-2.1:** An authenticated user shall be able to submit a new journal entry via a text-area form.
- **FR-2.2:** The system shall associate the new entry with the currently authenticated user.
- **FR-2.3:** The system shall reject the submission of an empty entry.

3.3 Feature 3: NLP & Sentiment Analysis

- **FR-3.1:** The system shall perform NLP analysis *immediately* upon a new entry submission as part of the same API request.
- **FR-3.2:** The analysis shall generate a numerical **sentiment polarity score** (a float between -1.0 and 1.0).
- **FR-3.3:** The system shall derive a **sentiment label** ("Positive", "Negative", or "Neutral") from the score.
    - `Positive`: Score > 0.2
    - `Negative`: Score < -0.2
    - `Neutral`: Score between -0.2 and 0.2
- **FR-3.4:** The system shall save the new entry to the database with its content, user, score, label, and a "PROCESSED" status.
- **FR-3.5:** The API shall return a "success" response to the client after the entry is saved and analyzed.

3.4 Feature 4: Data Visualization Dashboard

- **FR-4.1:** The frontend shall fetch aggregated data from the `/api/dashboard-stats/` endpoint upon loading.
- **FR-4.2:** The dashboard shall display the **Total Entries** count for the logged-in user.
- **FR-4.3:** The dashboard shall display a **Pie Chart** showing the "Mood Distribution" (count of Positive, Negative, and Neutral entries).

- **FR-4.4:** The dashboard shall display a **Line Chart** showing the "Sentiment Trend" (average sentiment score grouped by day).
- **FR-4.5:** The dashboard shall display a list of **Recent Entries**, showing the content, date, mood label, and rounded sentiment score for each.
- **FR-4.6:** The dashboard shall automatically re-fetch its data (and update all charts) after a new entry is successfully submitted, providing instant feedback to the user.

---

4. External Interface Requirements

4.1 User Interfaces

- **UI-1 (Login):** A clean, simple, centered form for login and registration, featuring the "MindScribe" logo and title.
- **UI-2 (Dashboard):** A responsive, multi-column grid layout. It must contain:
  - A header with the "MindScribe" title and a "Log Out" button.
  - A "New Entry" form card.
  - Data cards for "Total Entries," "Mood Distribution," "Sentiment Trend," and "Recent Entries."
- **UI-3 (Branding):** The application shall display "MindScribe" as the browser tab title and use the custom logo as the favicon.

4.2 Software Interfaces

- **API:** The Django backend shall provide a RESTful API that communicates using JSON.
- **Database:** The backend shall interface with a PostgreSQL database (via `psycopg2` and `dj-database-url`).
- **CORS:** The backend shall be configured to only accept cross-origin requests from the deployed frontend URL and local development URLs.

4.3 Hardware Interfaces

No special hardware is required. The application runs on standard web server hardware (via Render) and client devices (computers, smartphones).

---

5. Non-Functional Requirements

- **NFR-1 (Performance):**
    - The login page shall load in under 3 seconds.
    - The dashboard (with all data) shall load in under 4 seconds.
    - A new journal entry submission (including instant analysis) shall complete in under 3 seconds.
- **NFR-2 (Security):**
    - All user passwords must be hashed (handled by Django).
    - All communication between the frontend and backend shall be over HTTPS (handled by Render).
    - The API shall enforce data separation; a user must never be able to see another user's entries.
- **NFR-3 (Usability):**
    - The interface shall be intuitive and require no training.
    - Charts must be clearly labeled.
    - The application must be responsive and usable on both desktop and mobile screens.
- **NFR-4 (Maintainability):**
    - The codebase shall be separated into two distinct projects: `mindscribe-frontend` and `mindscribe-backend`.
    - The backend shall use a `.gitignore` file to exclude virtual environments, cache, and local `db.sqlite3` files.
    - The frontend shall use a `.gitignore` file to exclude `node_modules` and `dist` folders.