ÇANKAYA UNIVERSITY

Software Requirements Specification

daily: an AI powered journaling application

Fatih Kapiz 202011064
Ümit Mete Şahin 202011021
Onurcan Erenel 202011071
Ahmet Buğra Yaka 201911068
Mehmet Mert Türkmen 201911064

1. Introduction

1.1 Purpose

The purpose of this document is to describe the application, Daily (stylized as daily) an AI powered journaling application. This app aims to create a fun and new way of journaling and writing diaries by introducing artificial intelligence, data analysis and anonymous journal sharing features. This document includes detailed information about requirements of the project. It reflects the identified constraints and proposed software functionalities. Moreover, this document explains how users interact with the application.

1.2 Scope of Project

In the realm of personal development, our journaling application emerges as a powerful tool, guiding users on a transformative journey of self-discovery and growth. At its core, the platform is designed to cultivate clarity and self-reflection, offering users a space to navigate the labyrinth of their thoughts and emotions. Through carefully crafted features, individuals can embark on a journey of introspection, gaining valuable insights into their inner world.

One of the pivotal aspects of our project lies in encouraging users to set and achieve personal goals. This feature adds a dynamic dimension to the journaling experience, fostering a sense of accomplishment and progress. Goal-setting functionalities, coupled with intuitive progress trackers and reminders, become integral components, steering users towards realizing their aspirations.

Communication skills, often underestimated in the context of personal development, take center stage in our application. Here, the platform serves as more than just a repository of thoughts; it becomes a training ground for expressing oneself articulately. Writing prompts are strategically incorporated to prompt users to delve deeper into their thoughts, fostering a communication style that is both expressive and refined.

Integrating technology, our application incorporates Natural Language Processing (NLP) to analyze and derive insights from users' journal entries. This not only adds a layer of sophistication to the user experience but also provides personalized feedback and recommendations for further growth.

1.3 Glossary

API (Application Programming Interface) - A set of protocols, tools, and definitions for building application software, enabling communication between different software components.

AI (Artificial Intelligence) - Artificial intelligence is the simulation of human intelligence processes by machines, especially computer systems. Specific applications of AI include expert systems, natural language processing, speech recognition and machine vision.

CI/CD Pipeline (Continuous Integration/Continuous Deployment) - An automated process for software development, integrating code changes and facilitating continuous delivery and deployment.

Data Analysis - Data analysis is the action of collecting, preprocessing data and later analyzing relations and making assumptions based on the data.

Deep Learning - A machine learning subfield that uses models that incorporate multiple layers of neurons.

GDPR (General Data Protection Regulation) - A regulation in EU law on data protection and privacy in the European Union and the European Economic Area.

Generative Image AI - An artificial intelligence system that utilizes neural networks to craft images based on some text input.

Golang - An open-source programming language used for backend server development.

Machine Learning - A subfield of artificial intelligence that uses statistics and mathematics to make predictions, based on input data.

Natural Language Processing - An artificial intelligence concept that utilizes special neural network models to extract information from text.

Neural Network - A machine learning method that uses an interconnected set of nodes that process an input and create outputs based on mathematical formulas.

React Native - A framework for building native apps using React, a JavaScript library for building user interfaces.

Reinforcement Learning - Reinforcement learning is another approach to machine learning, where after each action, the agent gets feedback in the form of reward or punishment (a positive or a negative numerical value).

Smart Mobile Device - A portable computing device such as a smartphone or tablet.

Text summarisation - Text summarisation is an action performed by NLP models that takes an input text, captures the essence of the text and generates a summarized version of it.

1.4 Overview of the Document

The second part of the document describes functionalities of the daily: an AI powered journaling application. Informal requirements are described and it is a context for technical requirement specification in the Requirements Specification chapter.

Requirement specification chapter is written for software developers and details of the functionality of the simulation are described in technical terms.

Both of the sections describe the functionalities of the same product. However, it is described differently because they are intended for different audiences.

2. Overall Description

2.1 Product Perspective

Daily, an AI powered journaling application with image generation is a mobile journaling project that incorporates several artificial intelligence methods to provide an interactive mobile journaling platform. The project has several aspects. It provides a journaling system, analyzes dailies, provides statistics about writing habits and emotions, collects dailies in albums, generates images based on extracted emotions and keywords for these dailies.

The platform also improves user interactivity by providing a daily sharing and reading mechanism for users who are interested in sharing their stories and reading about others'.

2.1.1 User Characteristics

2.1.1.1 User Demographics

- Participants must be at least 13 years of age.

2.1.1.2 User Technical Proficiency

- Participants must own a smart mobile device.
- Participants must have the ability to read and write.
- Participants must have familiarity using mobile operating systems iOS or Android.

2.1.1.3 Admin

- Admin must have a knowledge of website moderation.
- Admins must be familiar with database management systems.

3. Requirements Specification

3.1 Constraints, Assumptions and Dependencies

- **Device Compatibility:** Must be compatible with a range of mobile devices.
- **Network Dependency:** Relies on internet connectivity for synchronization and AI analysis.
- **Storage Limits:** Limited by the device's and server's storage capacities.
- **Processing Power:** Dependent on the processing capabilities of the user's device for the frontend.
- **User Tech Savviness:** Assumes a basic level of user familiarity with mobile apps.
- **Stable Internet Connection:** Assumes users have regular access to a stable internet connection.
- **Regular Usage:** Assumes users will regularly input journal entries for effective analysis.
- **Data Accuracy:** Assumes the data input by users is accurate and truthful.
- **Third-Party APIs:** Dependence on external APIs for features like sentiment analysis or data storage.
- **Mobile Operating Systems:** Dependent on iOS and Android operating systems for updates and compatibility.
- **Cloud Services:** Relies on cloud services for backend data storage and AI processing.
- **Open-Source Libraries:** Utilizes open-source libraries in React Native, Golang, and Python, subject to their availability and updates.

3.2 Functional Requirements

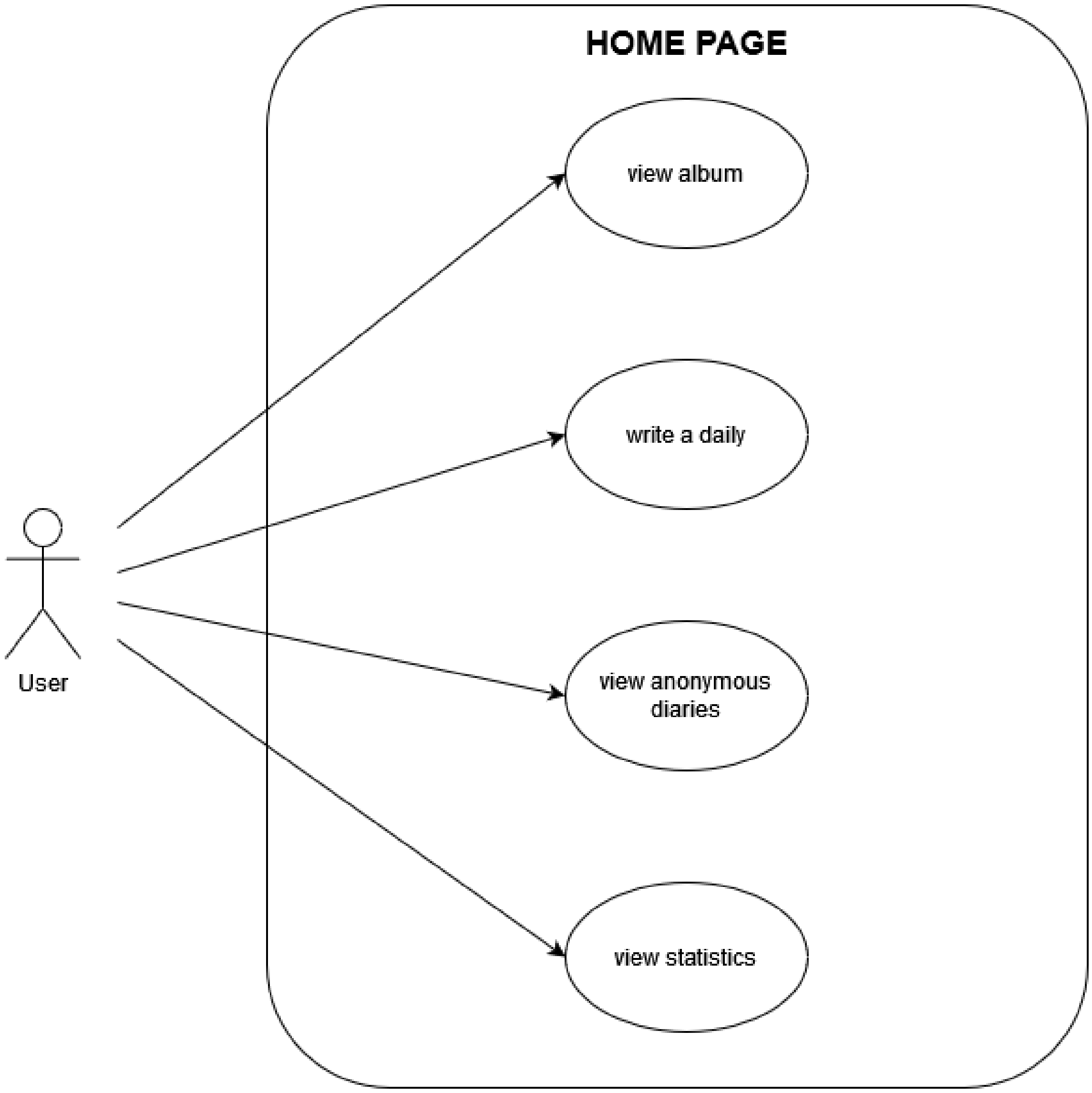
3.2.1 Homepage Use case

Use Case:

User:

- □ View album
- □ Write a daily
- □ View shared anonymous images
- □ View statistics

Diagram:



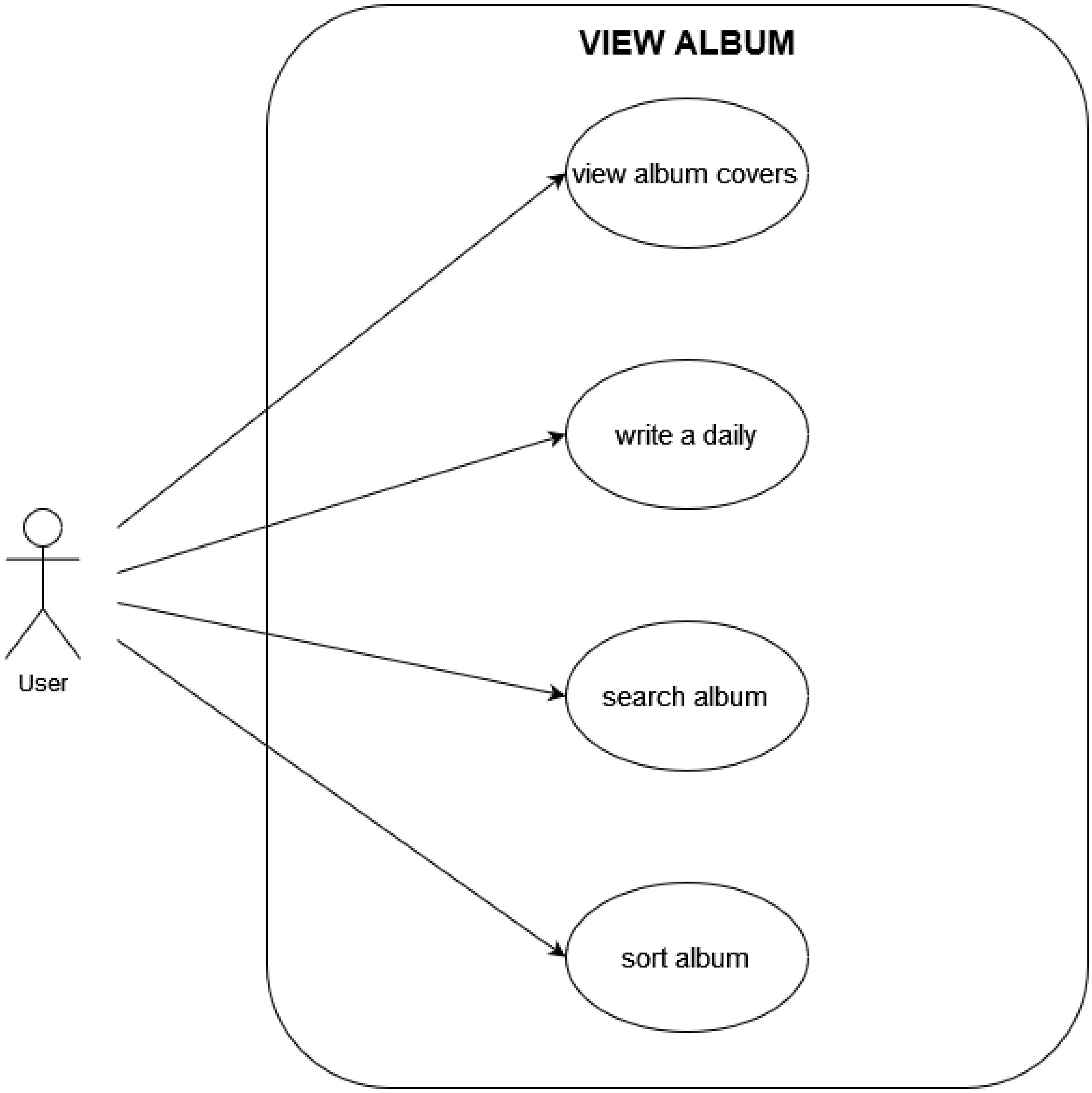
3.2.2 View Album Use Case

Use Case:

User:

- □ View album covers
- □ Write a daily
- □ Search album
- □ Sort album

Diagram:



3.2.3 Statistics Use Case

Use Case:

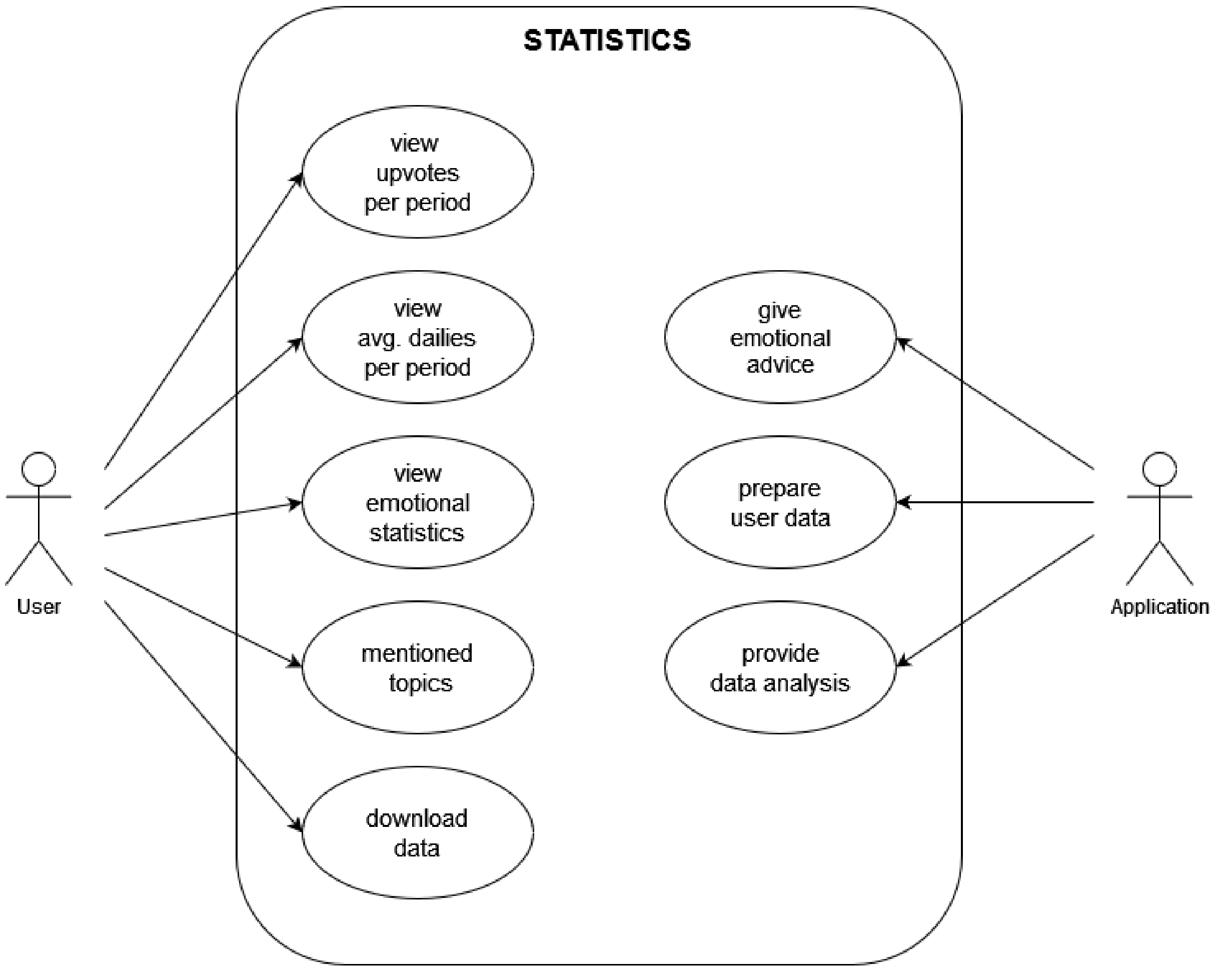
User:

- □ View upvotes per period
- □ View average pages written per period
- □ View emotional statistics
- □ View mentioned topics
- □ Download user data

Application:

- □ Provide emotional advice
- □ Prepare user data
- □ Prepare data analysis

Diagram:



3.2.4 Write a Daily Use Case

Use Case:

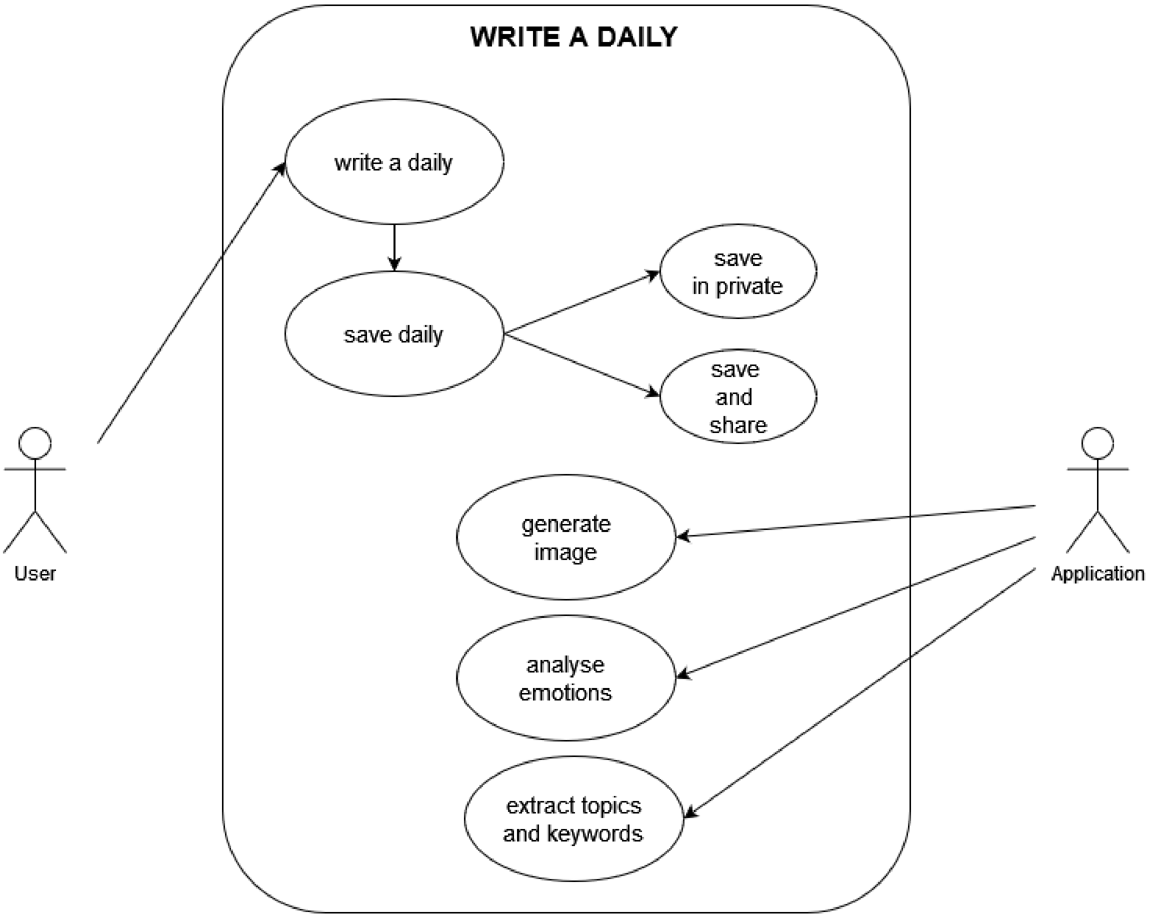
User:

- □ Write a daily
- □ Save a daily
    - □ Save in private
    - □ Save and share

Application:

- □ Generate image
- □ Analyse emotions
- □ Extract topic and keyword

Diagram:



3.2.5 View Anonymous Dailies

Use Case:

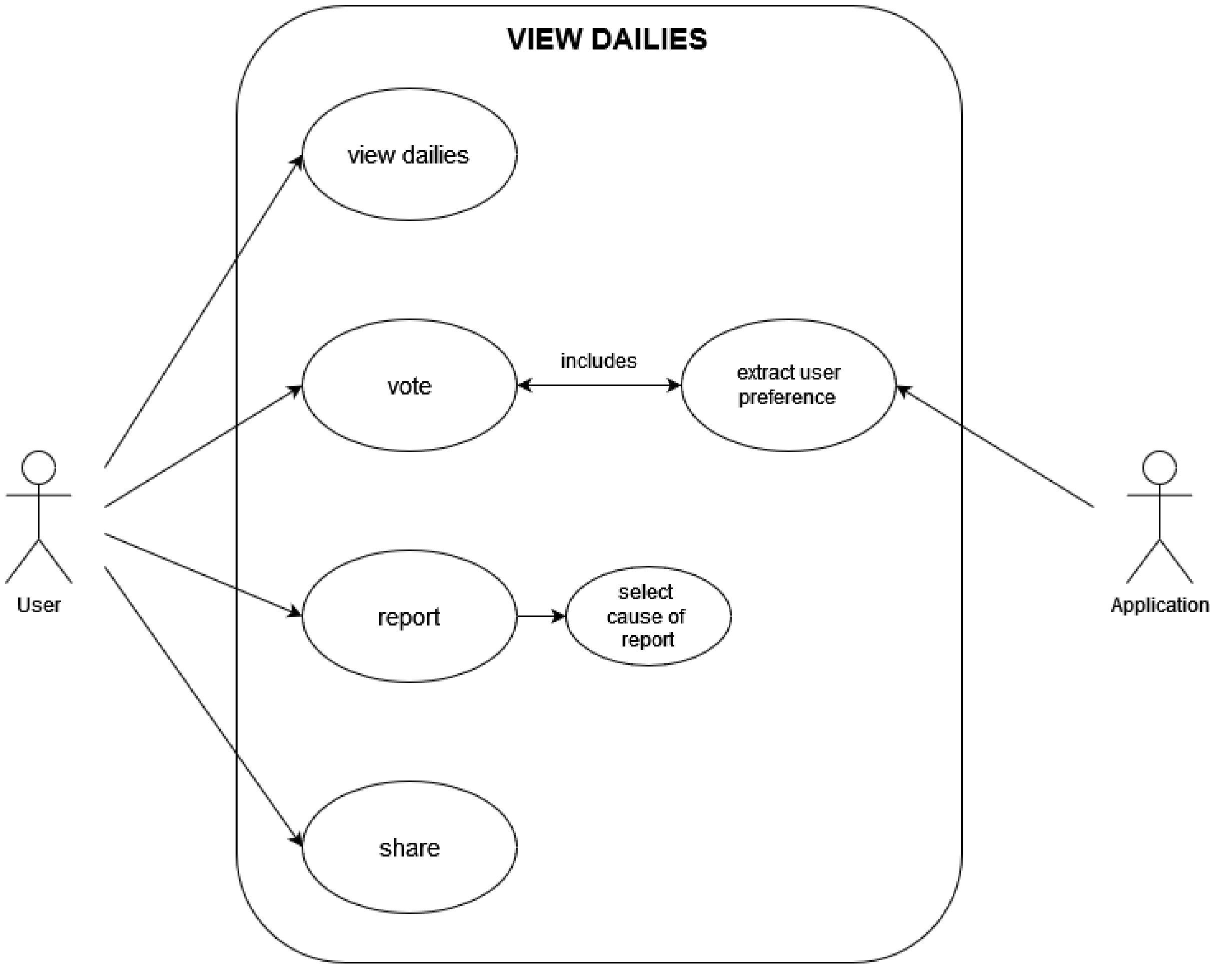
User:

- □ View dailies
- □ Vote
- □ Report
  - □ Select cause of report
- □ Share

Application:

- □ Extract user preferences based on user votes

Diagram:



-

3.3 Non Functional Requirements

Platform Compatibility and Performance:

React Native Limitations: Ensure compatibility with both iOS and Android platforms, keeping in mind the different performance characteristics and design guidelines of each.

Performance Overheads: React Native can have performance limitations for computation-intensive tasks compared to native development.

Backend Server Constraints (Golang):

Scalability and Load Management: The server must efficiently handle varying loads, potentially requiring load balancing and efficient resource management.

Database Interactions: Optimize database queries and ensure efficient data handling, especially with large journal entries or high user volumes.

Python AI System Constraints:

Computational Resources: AI and machine learning models can be resource-intensive. Assess the need for specialized hardware or cloud resources for processing.

Data Privacy and Security: Handling personal journal entries requires stringent data privacy measures and compliance with regulations like GDPR or HIPAA.

Integration and Communication Between Components:

API Design and Management: Robust and secure APIs for communication between the frontend, backend, and AI system.

Data Serialization and Transfer: Efficient data serialization formats to minimize latency and bandwidth usage.

User Interface and User Experience:

Cross-Platform UI Consistency: Ensuring a consistent UI/UX on different device types and operating systems.

Accessibility and Localization: Designing for accessibility and potential localization needs.

Development and Deployment Infrastructure:

Version Compatibility: Ensuring compatibility between different versions of React Native, Golang, and Python libraries.

CI/CD Pipeline: Streamlining the development process with a continuous integration and deployment pipeline.

Testing and Quality Assurance:

Cross-Platform Testing: Comprehensive testing strategies for both Android and iOS platforms.

AI Model Accuracy and Bias Testing: Ensuring the AI provides accurate and unbiased analysis of journal entries.

Regulatory and Compliance Constraints:

Data Protection Laws: Compliance with data protection laws in different regions (e.g., GDPR in Europe).

Intellectual Property: Use of open-source libraries and dependencies within legal constraints.

Market and User Constraints:

User Demographics and Behavior: Tailoring features and performance according to the target user base.

Competitive Features: Keeping up with features offered by competitors in the journaling app market.

Dependency Management:

External Libraries and Frameworks: Managing dependencies and ensuring compatibility across different libraries used in React Native, Golang, and Python.

Network and Connectivity:

Offline Functionality: Designing for scenarios where users might have limited or no internet connectivity.

Data Synchronization: Efficient synchronization of journal entries between the device and server.