Insight Write📝

CS 3337-03 202301 Group #7

Team Members: Miguel Gonzalez Torres, Tony Lau, Hoang Le, Keyvan Mahmoodzadeh Kani, Will May, Hayk Vardapetyan, Jian Verdad, Niyusha Zarnegar.

Niyusha Zarnegar

Introduction

Team Member Names and Contributions:

- **Full Team Contributions:** Documentation, Software Design, Project Planning
- **Miguel Gonzalez Torres:** Backend Development(python)
- **Tony Lau:** Backend Development (Python) & Testing
- **Hoang Le:** Front-End Development (HTML/CSS/JS)
- **Keyvan Mahmoodzadeh Kani**: Backend (Python) Development & Railway Database
- **Will May:** Front-End Development (HTML/CSS/JS)
- **Hayk Vardapetyan:** Backend Google Login Integration
- **Jian Verdad**: Backend Development (Python) & Railway Database
- **Niyusha Zarnegar:** Database, Server Setup & Testing, Architecture

- Link to application (if web based):
- Link to GitHub: Insight Write Repository

Miguel Gonzalez Torres

Our Vision!

Who is the Customer

- Individuals who are interested in personal growth and self-expression

Problem

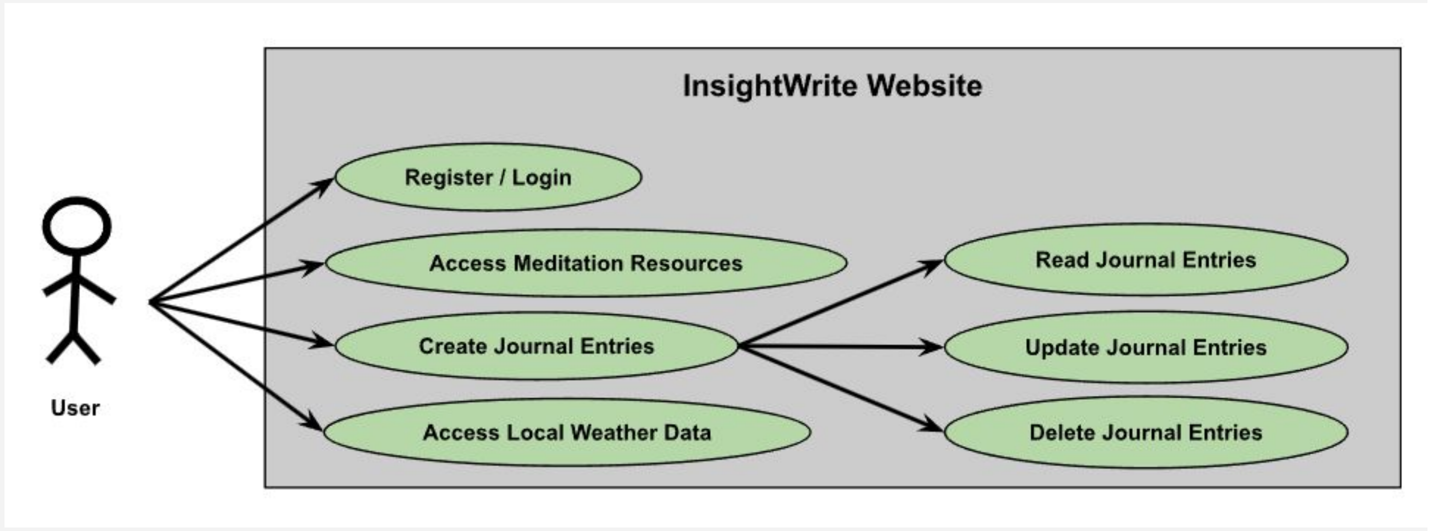
- Lack of existing applications which allows the user to freely express themselves
- Many users feel disconnect in their journaling journey cause of a lack of feedback

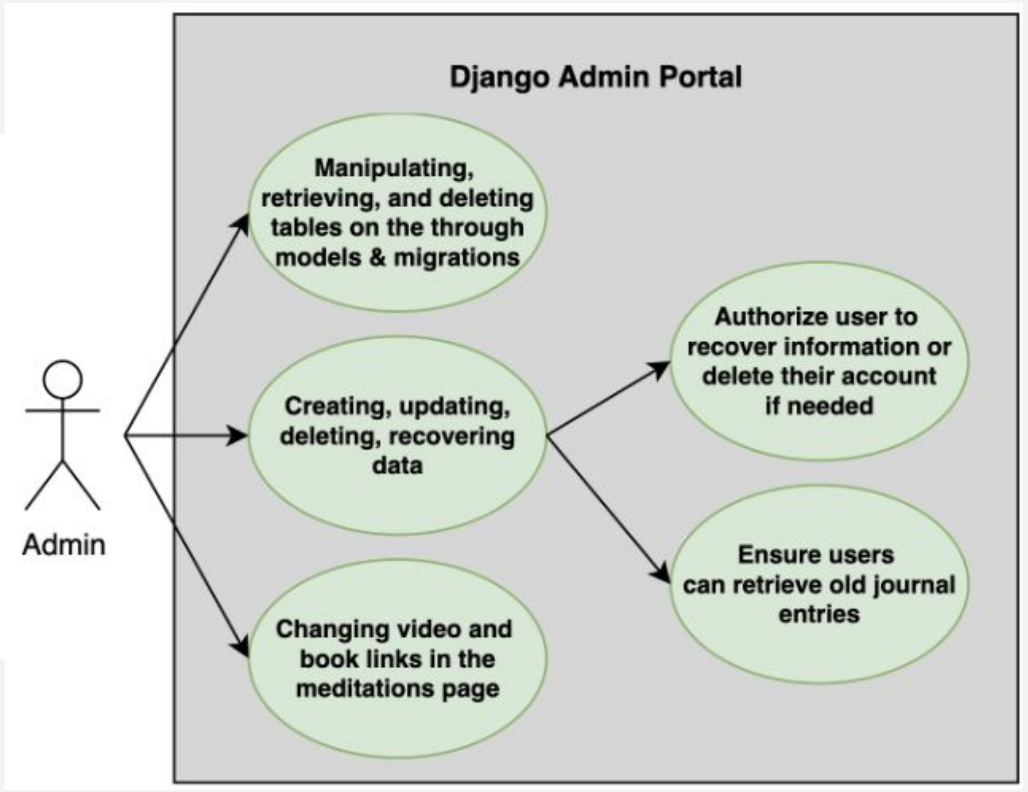
Digital Solution

❖ Our application: A platform with a focus on allowing our user to express themselves through journaling.
  - ➢ Facilitates personal growth
  - ➢ Provides a private space to reflect
  - ➢ Improve mental health

Jian Verdad

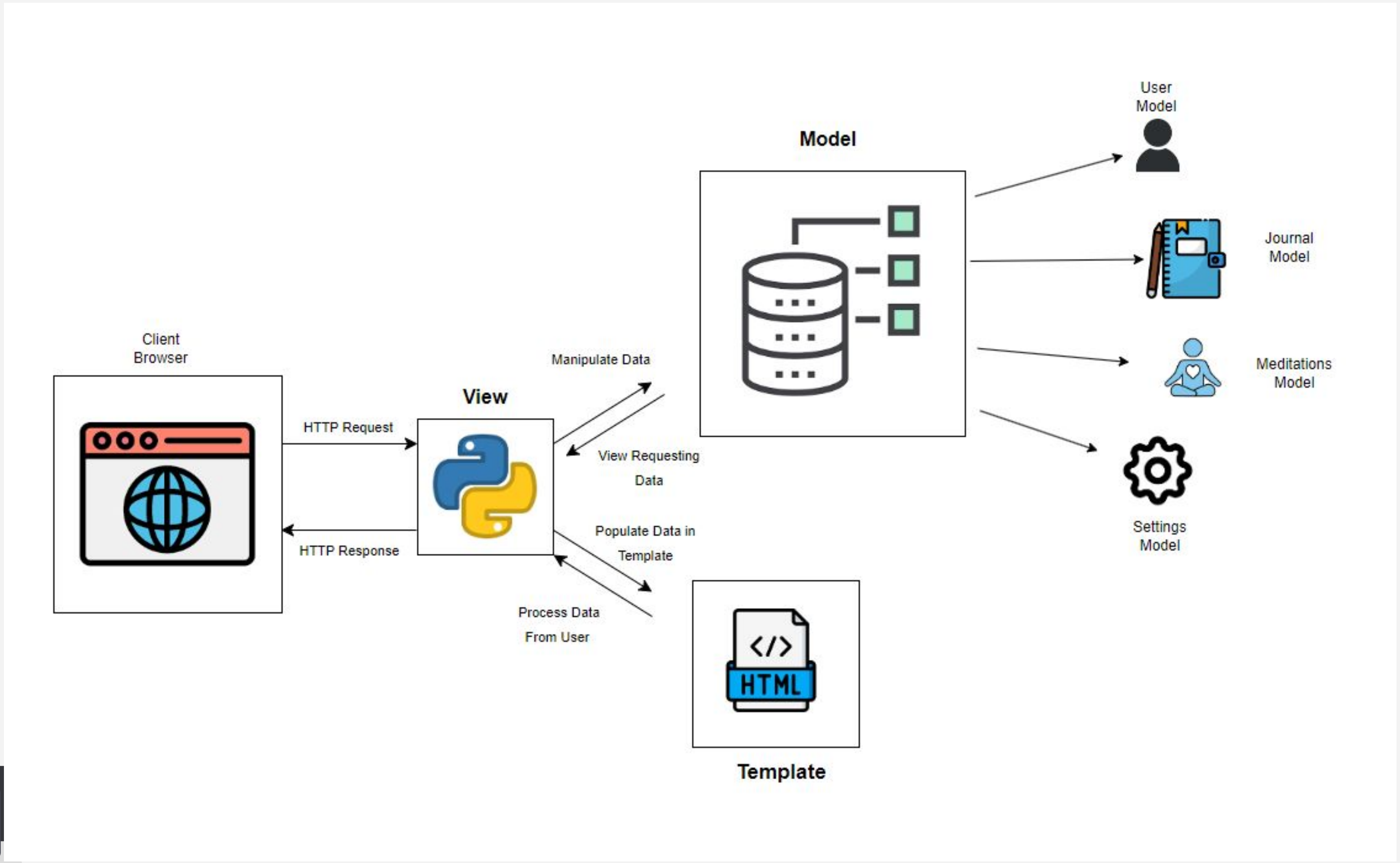
Application Requirements




Django Admin Portal
Admin
Manipulating, retrieving, and deleting tables on the through models & migrations
Creating, updating, deleting, recovering data
Changing video and book links in the meditations page
Authorize user to recover information or delete their account if needed
Ensure users can retrieve old journal entries

Hayk Vardapetyan

Architectural Design



Non functional requirements that favor this architecture for our web application include extensive security measures in place by MVT / MVC architectures for authentication and encryption, as well as allowing an easy way to be compatible with other third party systems.

Technologies Used & Implemented

Hoang Le

Frontend:

- **JavaScript**, **HTML**, and **CSS** → Building user interface

Backend:

- **Python**
  - **Django Framework:** Provides structured web application development
  - **Settings, Models, Views, URLs:** Components to configure, interact with the database, handle logic, and map web requests

Database:

- **PostgreSQL** on **Railway** → Store and save users' accounts and journals data.

Website Hosting:

- **Google Cloud**

Authentication and Authorization:

Django Allauth → Login with Google

API Integration:

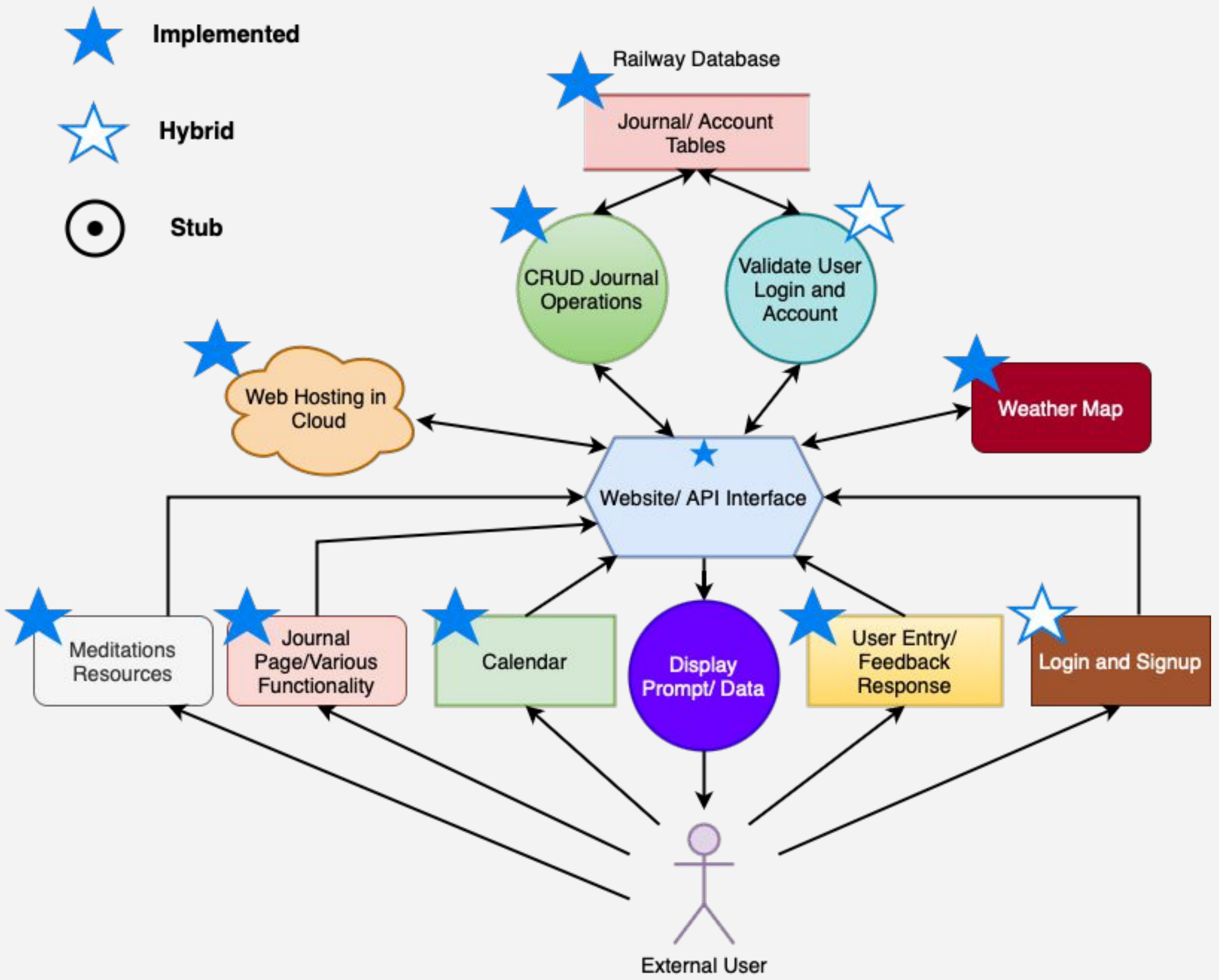
OpenWeatherMap API → Retrieve weather information

- **Third Party Libraries:**
  - Bootstrap → Navigation bar
  - Boxicons, Ionicons → Website Icons
  - Font Awesome → Animations
- **Resources:**
  - Pexels → Background images for our website

Processes - Detailed Design

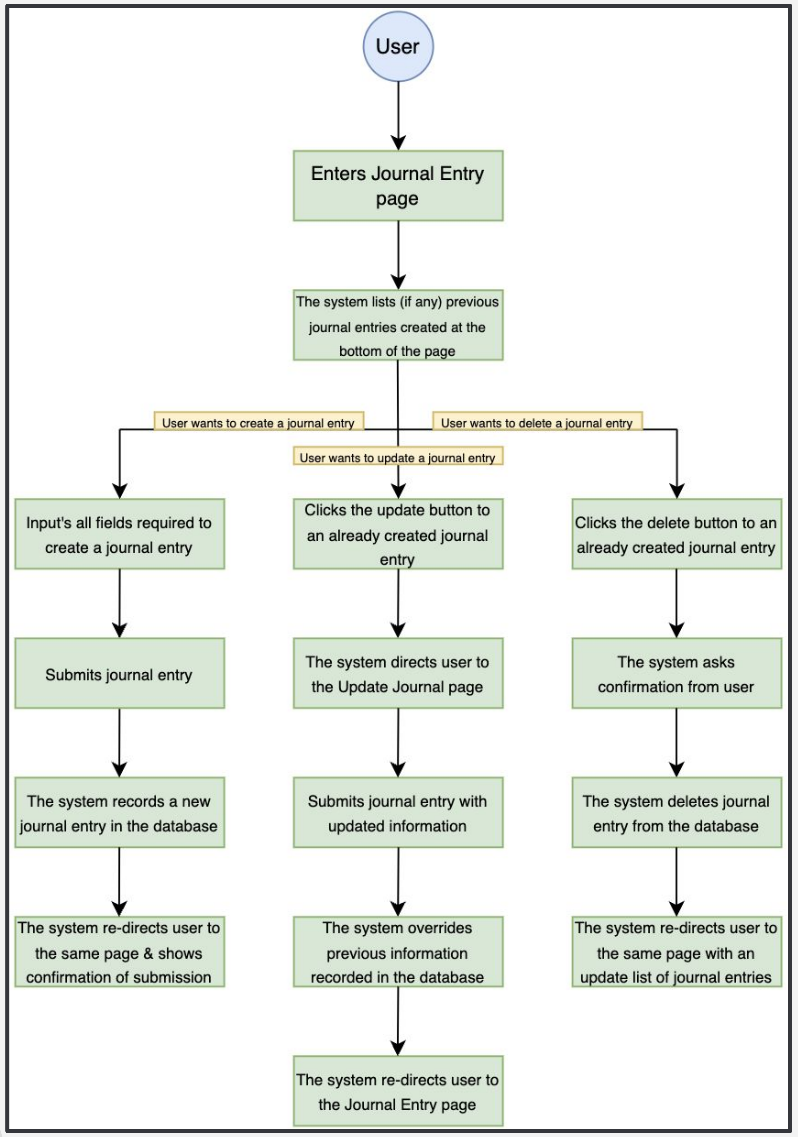
Will May

DFD Level 1 Diagram of Insight Write



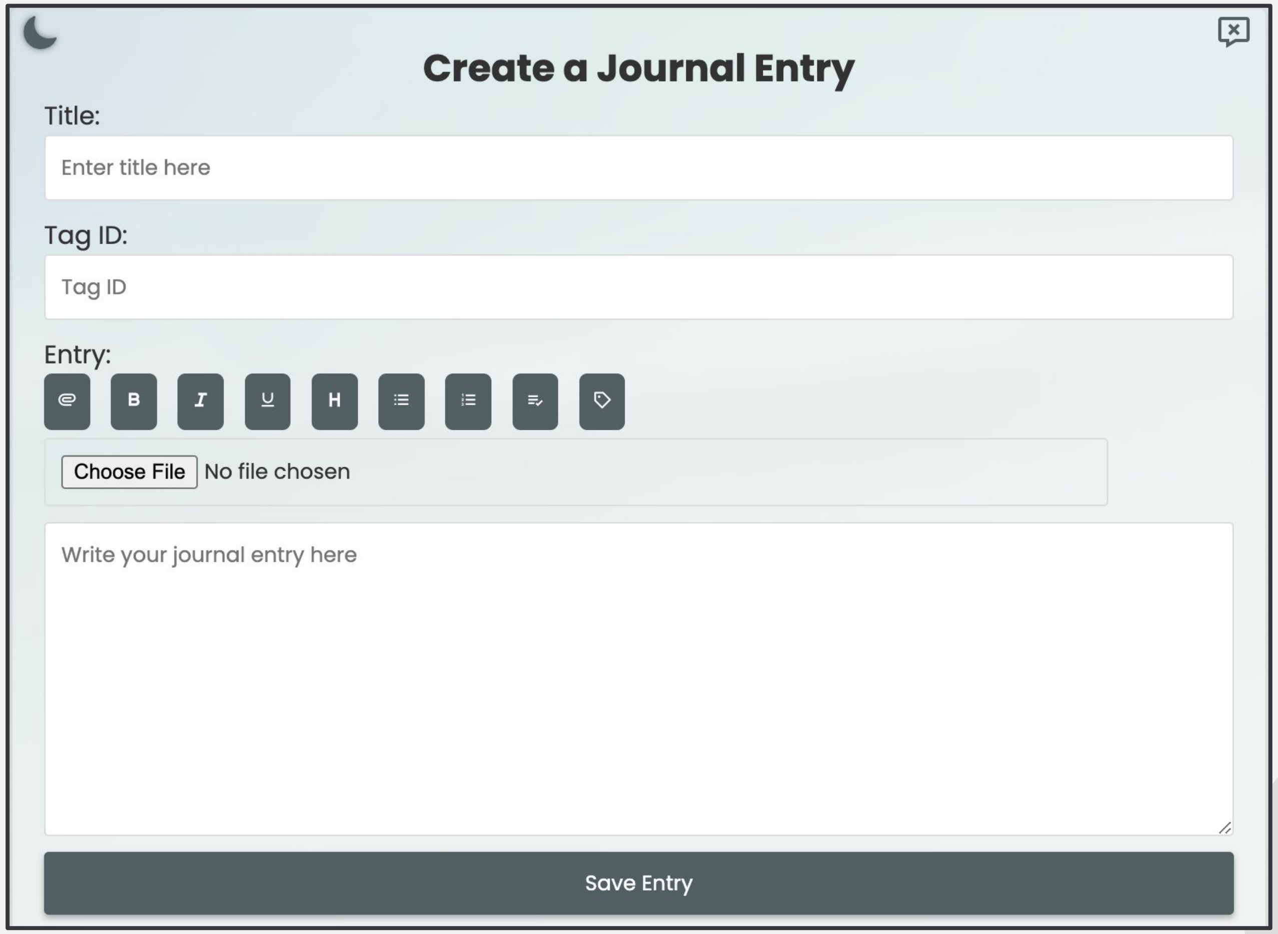
Keyvan M Kani

Behavioral Diagram of Journal Entry



User
Enters Journal Entry page
The system lists (if any) previous journal entries created at the bottom of the page
User wants to create a journal entry
User wants to update a journal entry
User wants to delete a journal entry
Input's all fields required to create a journal entry
Submits journal entry
The system records a new journal entry in the database
The system re-directs user to the same page & shows confirmation of submission
Clicks the update button to an already created journal entry
The system directs user to the Update Journal page
Submits journal entry with updated information
The system overrides previous information recorded in the database
The system re-directs user to the Journal Entry page
Clicks the delete button to an already created journal entry
The system asks confirmation from user
The system deletes journal entry from the database
The system re-directs user to the same page with an update list of journal entries

Journal Entry Management:
Handles the creation, retrieval, update, and deletion (CRUD) operations for user journal entries.



Niyusha Zarnegar

Lessons Learned

- Docker Issues/Managing separate deployment branch
- Agile Development was very helpful in communications
- How to better define features and stories
- Learned more about Railway, PostgreSQL, and Google Cloud
- Organizing the repository
- Locally hosting webpages
- The intricacies of how frontend and backend interact
- Integrating APIs
- Database management

Niyusha Zarnegar

Production Deployment Architecture and Choices

- Remote Database allows for scaling and resiliency
- Dockerized web server

Tony Lau

Column Roadmap (24.0, 24.1, 24.2, Future Enhancements)

24.0	24.1	24.2	Future
. Selection of Website Hosting - Platform: Github Pages . UX/UI Design . Change Management Plan . Unit Testing and Test Plan - PyTest . Frameworks and Tech Stack: - HTML/CSS/JS - Bootstrap - Django Web Framework - MongoDB API - NLTK API	Software Test Plan (STP) Software Requirements Document (SRD) Software Design Document (SDD) Journaling Webpage Login Webpage Setting Web Page Design Database Design	Framework Changes - Django Web Framework to Railway - PostgreSQL Databases - Google Cloud Mostly done with Journal webpage Finalized Signup/Login webpage Dashboard Webpage Meditation Webpage Database Implementation Calendar and adding events Soft API Integration: - Django allauth - OpenWeatherMap.org Journal CRUD operations - Designs, fonts, colors, etc. - Mini-draggable calendar - Text-Formatting (Bold, Italics, Hyperlink, etc.) Basic Javascript on all webpages Basic Python operations with database	Finalize Journal webpage (ex: Tags) Finalize Calendar (ex: email notifications) Unique data verification Encryption/Security Email Verification Setting Web Page (including User Accessibility features) Autosave Data and Information Save images and gifs to database Setting Editing User Profile Page Recovering Trashed Journal Entries

Demo

Q & A