

Modus.ai

Team 23 - Design Doc

*Rashmi Ananth, Manan Bhatia, Nikita Finchenko,
Angela Kim, Jisoo Kim, Mihira Krishnaswamy*

Contents

Purpose	4
Components	6
Web-client	
Web-server	
ML Engine	
User Database	
Simplified Architecture	9
High Level Interactions Between Components	10
Design Issues	11
Functional Issues	11
How will Modus.ai collect user information for mood analysis?	
What kind of output can the user expect after analysis?	
How will Modus.ai ensure the security of its clients' data?	
Non-Functional Issues	14
What type of implementation will be used for the mood analysis algorithm?	

How will the architecture be structured?
What type of application will Modus.ai be?

Design Details	16
Class Level Diagram	16
Descriptions of Classes and Models	17
User	
JournalEntry	
Journal	
Model	
MoodAnalysis	
Visualization	
UI	
Sequence Overview	21
Sequence of events when user creates an account	
Sequence of events when user logins	
Sequence of events when user submits the journal for analysis	
Sequence of events when user saves a journal entry	
Sequence of events when user provides feedback to Modus.ai	
UI Mockups	26

Purpose

Mental health problems are on the rise among several individuals in today's society. Quite often, these problems occur when individuals repress their emotions or are unable to fully understand them. As human beings, we experience a variety of emotions and have emotional responses to everything in life. To repress those emotions or simply not be able to understand them is unhealthy. Hence, it is necessary for individuals to have a way out for their feelings.

Simply expressing emotions doesn't always help one feel better. It is imperative that one understands the variety of emotions they are experiencing so that they may act on them appropriately. In today's age, we are lucky to have access to numerous mental health resources that individuals may benefit from, but sometimes it may not be easy to pick the right one. Hence, people can benefit from a tool that helps facilitate the silos between their moods and the comprehension of their emotions.

This is where *Modus.ai* comes in. It uses journal entries instead of a set of static questions to analyze the user's mood. This would enhance user autonomy by ensuring additional personalization, and set us apart from our competition. It will analyze individual journal entries per user for various moods/emotions and keep track of the analyses over time to track the user's mental health progress. Hence, the process of journaling will help users express their emotions and the analysis will help them understand those emotions.

Design Outline

Our implementation will use a Three-Tier Model model. On top of a typical client-server interaction, the third tier will be the data processing (ML Engine) unit. The user(s) will be able to use our web-app to query the server. The server will access data stored in a database and analyze it for various emotions or moods. The server will allow multiple users (~1000) to concurrently make requests for analyses. The database will store journal entries for each user.

Components

1. Web-client

- a. Users will be able to access the web-server through our website which will act as our UI.
- b. The client will consist of two modules:
 - i. **User Interface (UI):** This module will be our website and will control all the necessary content and features needed to use *Modus.ai*.
 - ii. **UI Pre-processing Engine:** This module will contain all the relevant API calls to establish connections between the UI and the server. It will also process the data received from the database and the server and prepare it for the website.
- c. The client will send requests made via the website to the web-server. The client may make requests to the web-server for the following reasons (non-exhaustive):

- i. To create an account and/or login to *Modus.ai*.
- ii. To switch between web pages and interact with the UI in general, and specifically due to the following reasons:
- iii. To create and store a journal entry (in our database).
- iv. To access previous journal entries.
- v. To analyze the current entry.
- vi. To view the mood analysis on our dashboard.
- vii. To view the relevant mental health resources.

2. Web-server

- a. The web-server is the central control unit for our application as it will be responsible for sending data and instructions to most other components.
- b. It will be responsible for hosting the web-app on the internet.
- c. The server will respond to the various kinds of client requests mentioned in 1.c.
- d. The server will be responsible for making calls to the database to retrieve or store information
- e. It will also act as a mode of communication between the other components. For e.g. the ML Engine and the Database components.
- f. The server may accommodate up to 1000 concurrent requests.

3. ML Engine

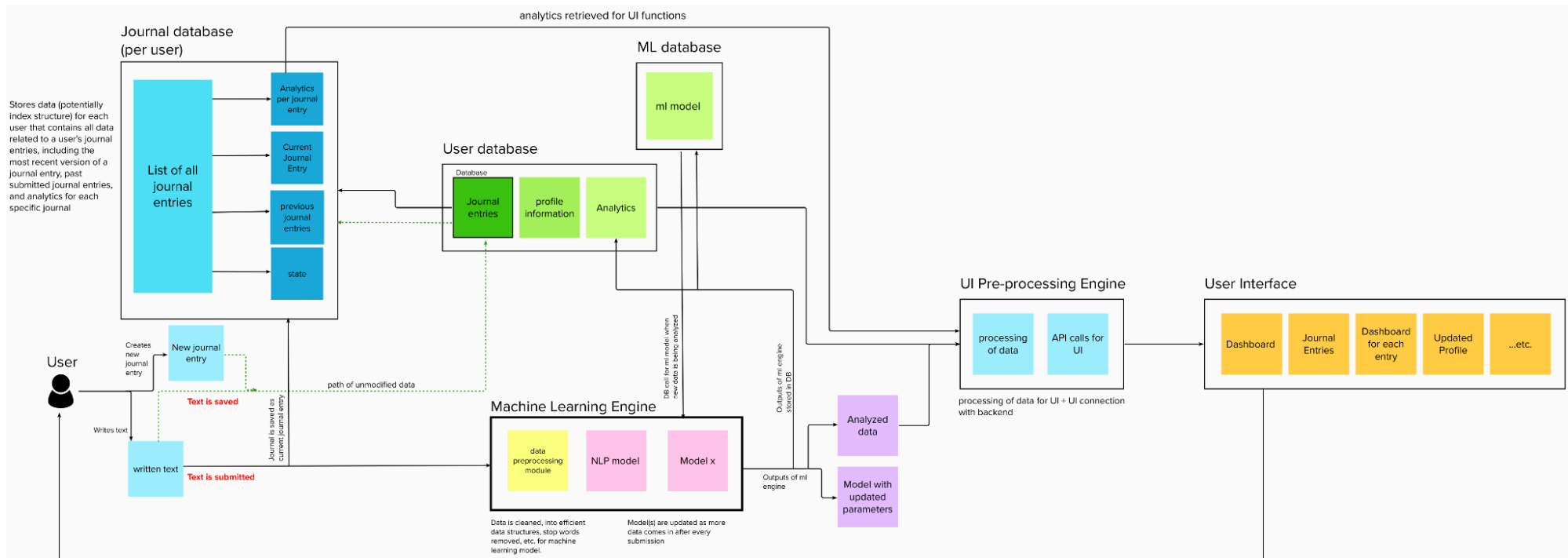
- a. The ML Engine will be the main text analysis component divided into smaller components.
- b. This will include the NLP model itself using the Python Text2Emotion (T2E) library as well as the model image with updated parameters.
- c. The server will retrieve submitted journal entries from the database and pass them to the ML engine when an analysis is requested by the client.
- d. With the help of T2E, the ML Engine will be responsible for cleaning the received data, removing stop words (words that are not relevant for analysis), and sorting it into efficient data structures, internally, before actual analysis is performed.
- e. It will use a scoring system to identify the proportion of specific emotions conveyed in each journal entry. It should be able to detect the following emotions: 'Happy', 'Angry', 'Surprise', 'Sad', 'Fear'. Hence, an example analysis could look like {'Happy': 0.07, 'Angry': 0.09, 'Surprise': 0.16, 'Sad': 0.18, 'Fear': 0.5}.
- f. It is important to note that the sum of the individual proportions will never exceed 1 but it may be less than 1. This is because it is possible that at any given time, less than 100% of the journal entry may be properly analyzed and/or some of the more complex emotions may not be considered by the NLP model for the analysis.
- g. The ML Engine will send the analysis to the UI Pre-processing Engine via the server.
- h. Updated model parameters and relevant information may be stored in an ML database to customize the T2E library in the future.

4. User Database

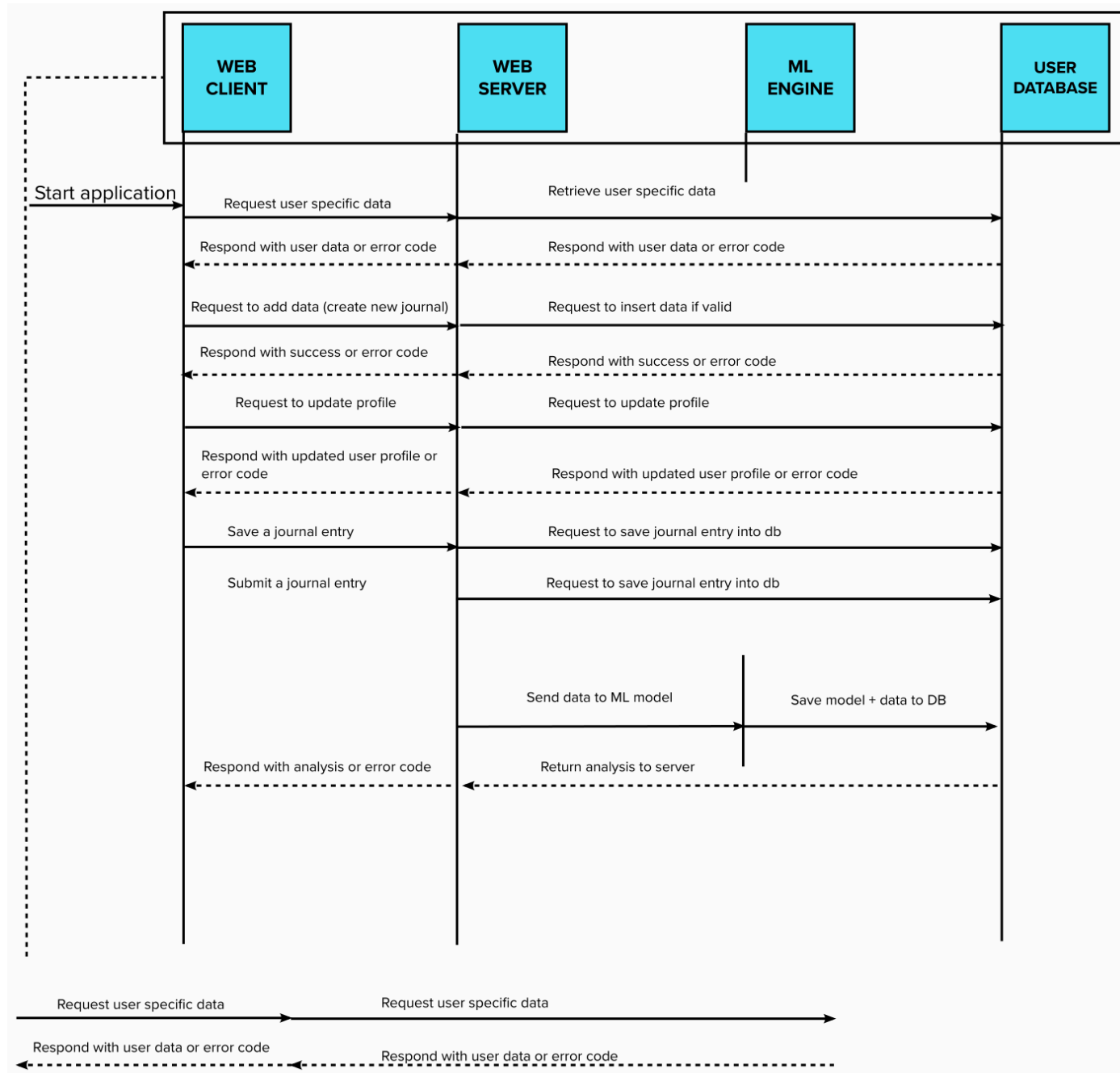
- a. The user database will have two levels. The upper level will contain information related to the user and the lower level will contain information regarding specific journal entries per user.
- b. The upper level will store the following information per user:
 - i. User profile information.
 - ii. Aggregated journal entry analyses from the ML Engine.
 - iii. Reference to the lower level journal database.
- c. The lower level journal database will be nested in the user database and contain the following information per user:
 - i. Reference to the current journal entry.
 - ii. Previous journal entries.
 - iii. Analytics per journal entry.
 - iv. State of the current journal entry: The state may be “*saved*” or “*submitted*”. Saved entries may be edited by the user multiple times. Once submitted, the user may not edit that entry. Submission saves the latest version in the journal database and sends it to the ML Engine for analysis.
- d. The journal entries may be sorted by date in the database.

- e. The database may only be directly queried by the server. Any other component relying on the database may query the database via the server.

Simplified Architecture



High Level Interactions Between Components



Design Issues

Functional Issues

a. How will Modus.ai collect user information for mood analysis?

i. Survey Entry

Input regarding the user's mood would be collected through a series of survey questions related to mood and emotion. Survey questions would be created by the team and stored and retrieved for the user in the database.

ii. Journal Entry (Selected)

Input regarding the user's mood would be collected through a journal entry written by the user. The journal entry will then be stored and processed through the UI pre-processing engine and the NLP model. This solution was selected due to its ability to provide more freedom and utility to the user through this format, as a journal entry would provide the user with more personalization and expression through written text as opposed to survey questions created in advance.

b. What kind of output can the user expect after analysis?

i. Quantitative-based Response

After input is processed, Modus.ai would present the user's mood analysis through a numerical scoring of their journal entry. Users would receive scores on several mood states (such as happiness, anger, sadness, etc.) that would be tracked and recorded over a period of time.

ii. Qualitative-based Response (Selected)

After input is processed, Modus.ai would present the user's mood analysis through a descriptive text in response to their journal entry. An algorithm would be created in order to translate numerical outputs of our NLP model to mood analysis that can be read from our users as well as stored for future reference. This solution was selected as it provides a more understandable and meaningful analysis to our users. In addition, this solution would allow Modus.ai to provide users with resources and help that is less general and more applicable to the user.

c. How will Modus.ai ensure the security of its clients' data?

i. Encryption (Selected)

After mood analysis output is created for the user, its output as well as the respective journal entry will be stored and encrypted in our database to ensure user privacy. This implementation was selected as a solution to security as the privacy of user's, especially when storing private text and information of the user, is crucial for Modus.ai to operate in a confidential and honest manner.

- ii. Username and Password System (Selected)

Before any journal entries and analysis can be created, users must create an account with Modus.ai alongside a username and password to login into the website. After a user has created an account, its information will be encrypted and stored in the database. This implementation was selected as a solution for user security as it provides the user with private and unique access to their journal entries and mood analysis.

Non-Functional Issues

- d. **What type of implementation will be used for the mood analysis algorithm?**

- i. NLTK Sentiment Analysis

Modus.ai would employ the python package within its machine learning engine in order to process user journal entries. Given the user's journal entry, the library would then be able to output a positive, negative, or neutral rating using its sentiment analyzer.

- ii. Text2Emotion (Selected)

Modus.ai would employ the python package within its machine learning engine in order to process user journal entries. Text2Emotion was selected due to its numerical output of 5 categories of emotion which can then be processed into a qualitative explanation for the user. In addition, the T2E library allows for the custom classifiers and training data, allowing the team to further refine and improve the ML engine if time permits.

e. How will the architecture be structured?

i. Client-Server Model

Modus.ai would be built on a client-server architecture. The Modus.ai UI would handle all data collection from the user and the databases will contain and manage user data. Databases will also require the application logic responsible for the processing and analyzing the user journal entries.

ii. Three-Tier Model (Selected)

Modus.ai would be built on a three-tier architecture. The Modus.ai UI, displaying and collecting journal entries, analysis, and other information, would serve as the presentation tier. The ML engine, containing the NLP model responsible for processing user data, would serve as the application tier. Lastly, both databases will serve as the data tier, storing and managing information being passed into the application tier. This model was selected as it most accurately represented the components the team needed to include seen in the design outline.

f. What type of application will Modus.ai be?

i. Mobile Application

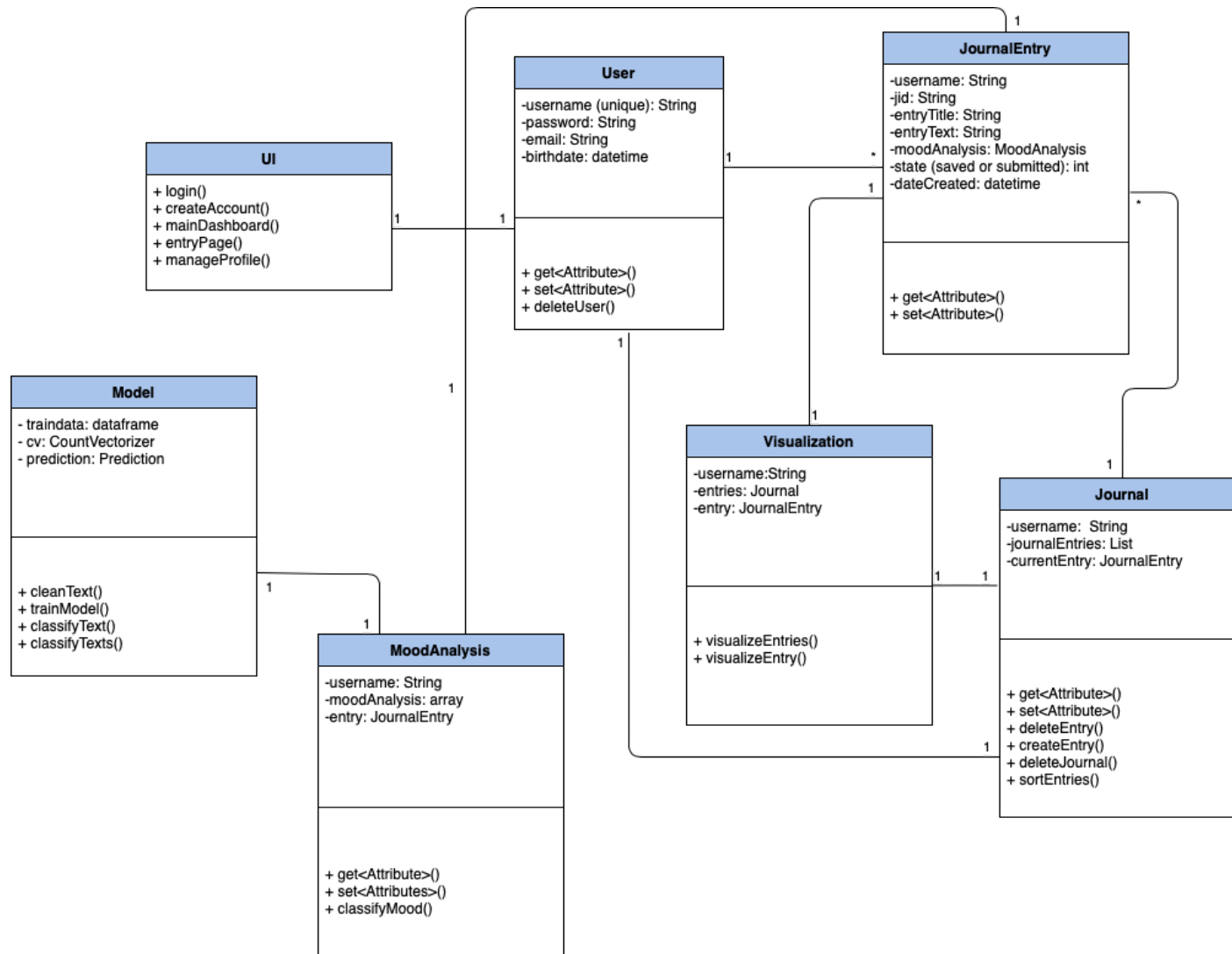
Modus.ai would be created as a mobile application on the Google Play Appstore/ Apple Appstore

ii. Web-based Application (Selected)

Modus.ai would be created as a website hosted on modusai.com (domain name subject to change). This option was selected based on our team's strengths and abilities, which in regards to knowledge and experience, are better suited towards creating web applications.

Design Details

Class Level Diagram



Descriptions of Classes and Models

The classes are designed based on the objects and models used in our web application. Each class has a list of attributes which contains the characteristics of each object or model.

User

- User object will be created when a user signs up for Modus.ai.
- Each user will be prompted to create a username that is unique.
- Each user will create a password.
- Each user will input their email and birthdate.
- Each user will select whether they would like to view the application in light or dark mode.
- Each user will have the ability to reset their password.
- Each user will have the ability to edit their email and birthdate.
- Each user will have the ability to add a picture to their profile.
- Each user will have the ability to delete their account.
- Each user will have a home page which displays a dashboard of their latest journal entry mood analysis, the mood trend of their past journal entries, and buttons to navigate to other pages.
- Each user will have a button which takes them to a page that lists all their journal entries.
- Each user will have a dropdown option to sort the list of journal entries by date created or name.
- Each user will have a search bar to search the list of journal entries by date created or name.
- Each user will have a button to create a journal entry.
- Each user will have a button to delete a journal entry.
- Each user will have a button to submit a journal entry.

- Each user will have a button to save a journal entry.
- Each user will have a toolbar when writing a journal entry which gives them color, style, and font options.
- Each user will be able to view an analysis of their individual journal entries by clicking on them.

JournalEntry

- JournalEntry object is created every time the create journal entry button is pressed.
- Each JournalEntry will have a username which will be used to match a user to a journal entry.
- Each JournalEntry will have a jid which is a unique journal entry id.
- Each JournalEntry will have a title, text, date created, and status (saved or submitted depending on which button is pressed).
- Each JournalEntry object's contents can be changed if the user chooses to edit and save or edit and submit a journal entry using the appropriate buttons.
- Each JournalEntry will have a MoodAnalysis object associated with it which it retrieves using a shared username and unique jid (journal id).

Journal

- Journal object is created when a user creates their account.
- Each Journal will have a username which will be used to match a user to a Journal.
- Each Journal will have a list of JournalEntry objects associated with a username.
- Each Journal will have a currentEntry to keep track of the user's most recently submitted JournalEntry easily.

- If a user deletes a journal entry using the delete entry, the entry will be removed from the JournalEntry list.
- If a user deletes their account, all associated journal entries will be deleted.

Model

- Model contains training data (labeled for mood analysis).
- Model creates a count vector of training data.
- Model stores prediction after inputting test data (text or texts) through model.
- Model cleans text before count vectorization.
- Model is trained and created with training data.

MoodAnalysis

- MoodAnalysis object is created when a user clicks the submit button for a journal entry.
- Each MoodAnalysis has a username and jid which will be used to match a MoodAnalysis object to a JournalEntry object.
- Each MoodAnalysis has an entry which is just the text field of the JournalEntry object.
- Each MoodAnalysis has a moodAnalysis array which stores the result of passing the text field of the JournalEntry into the Model's text classifier.

Visualization

- Visualization object is created when a user submits their journal entry.

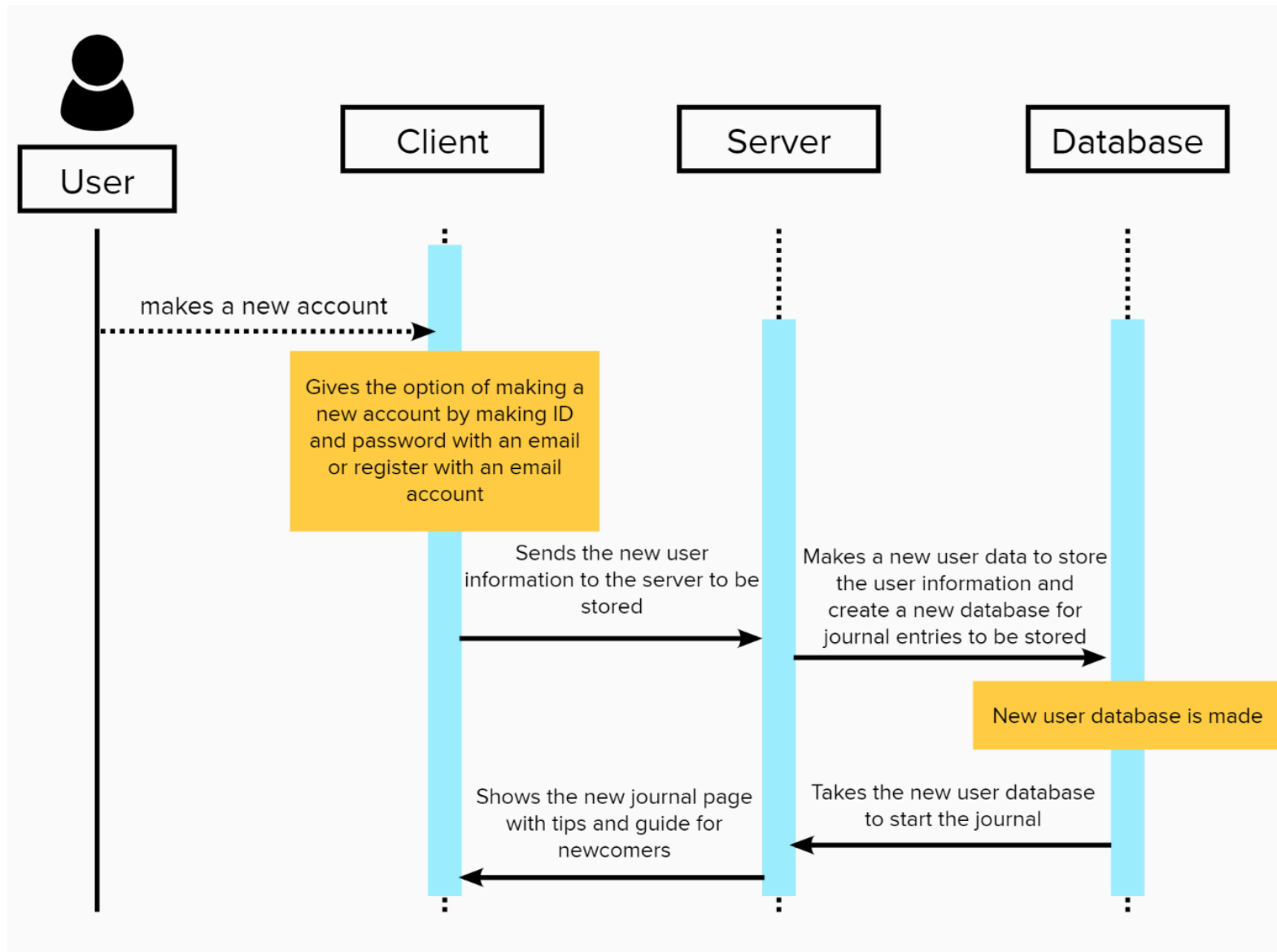
- Each Visualization has a username which will be used to match the generated visualization to a user and display it on the UI.
- Visualization can be created for a single JournalEntry which is stored in the entry field.
- Visualization can be created for a Journal (list of JournalEntry objects) which is stored in the entries field.
- Visualization on a list of entries will generate a progress graph that takes in mood analysis from all entries in the journal associated with a user over time.
- Visualization on a single entry will generate a more in depth visualization of the mood analysis for that entry including displaying keywords which contribute to mood score.

UI

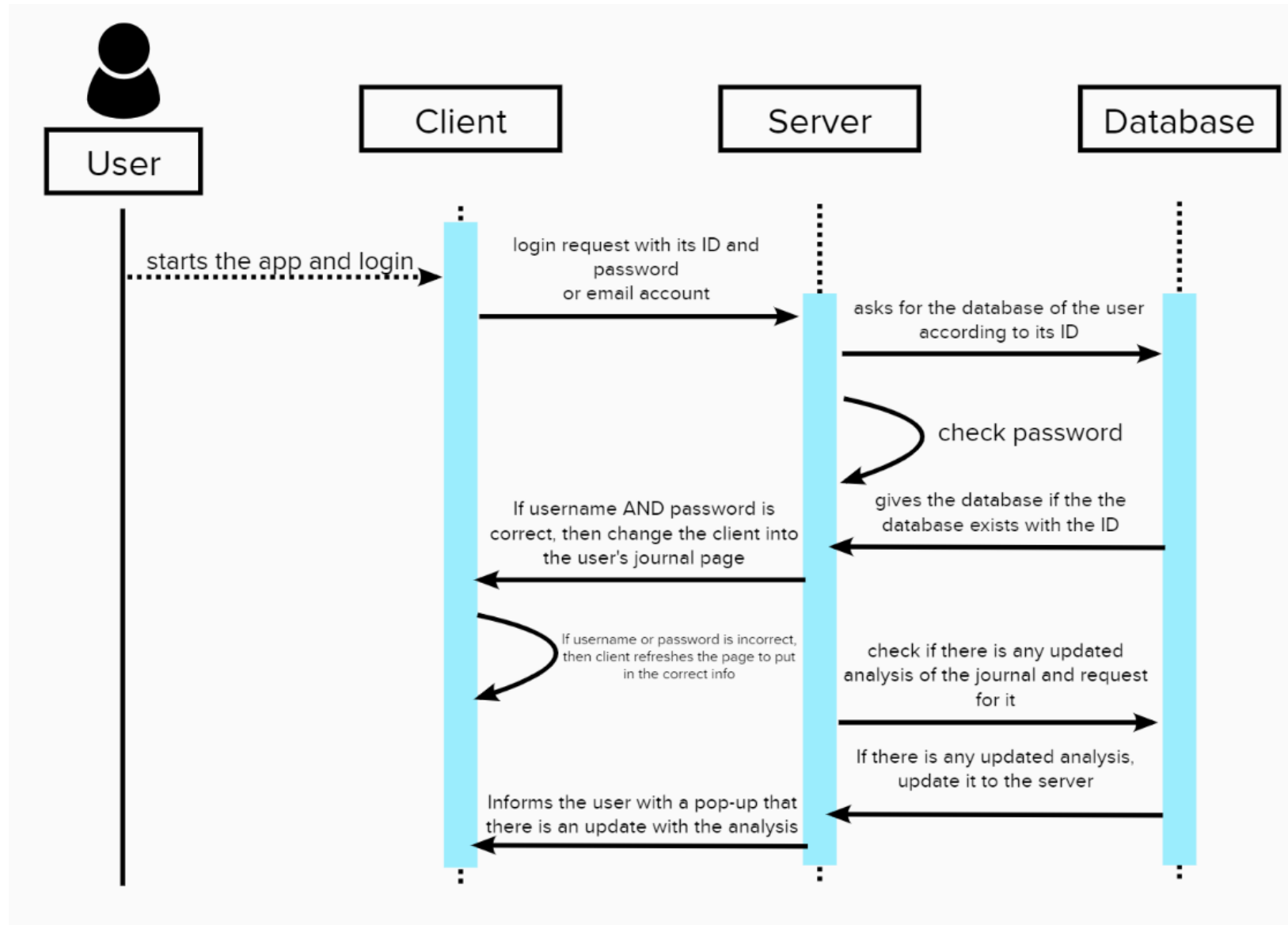
- UI object is created every time a user creates an account.
- UI creates login which displays a login page and contains appropriate buttons, action detection, and text fields.
- UI creates createAccount which displays create account page in case user does not have account.
- UI creates mainDashboard which displays user's mood analytics and options for journaling.
- UI creates entryPage with toolbar and textbox for when newEntry button on mainDashboard is pressed.
- UI creates a manageProfile page when the manageProfile button is pressed and allows the user to update account information.

Sequence Overview

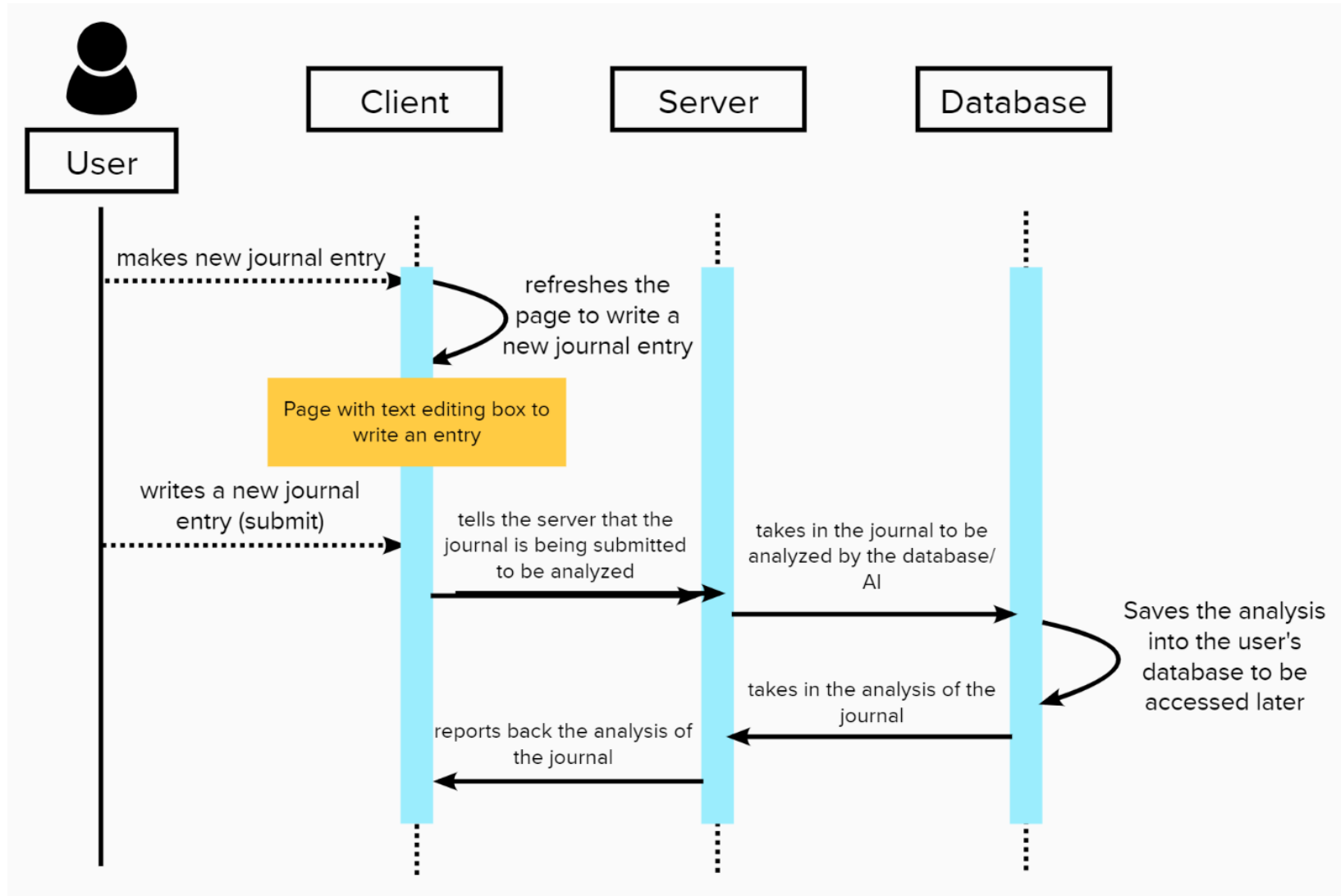
Sequence of events when user creates an account



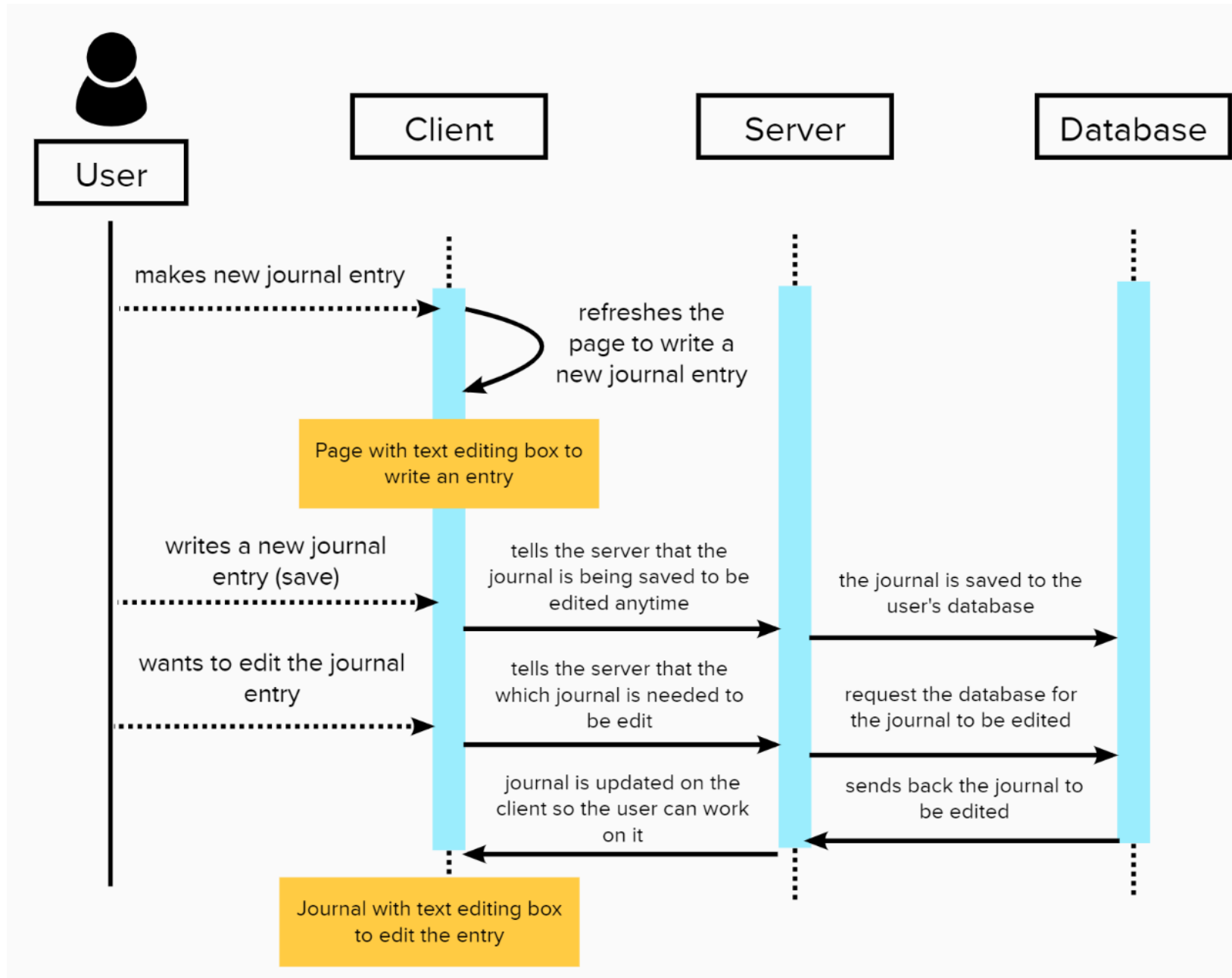
Sequence of events when user logs in



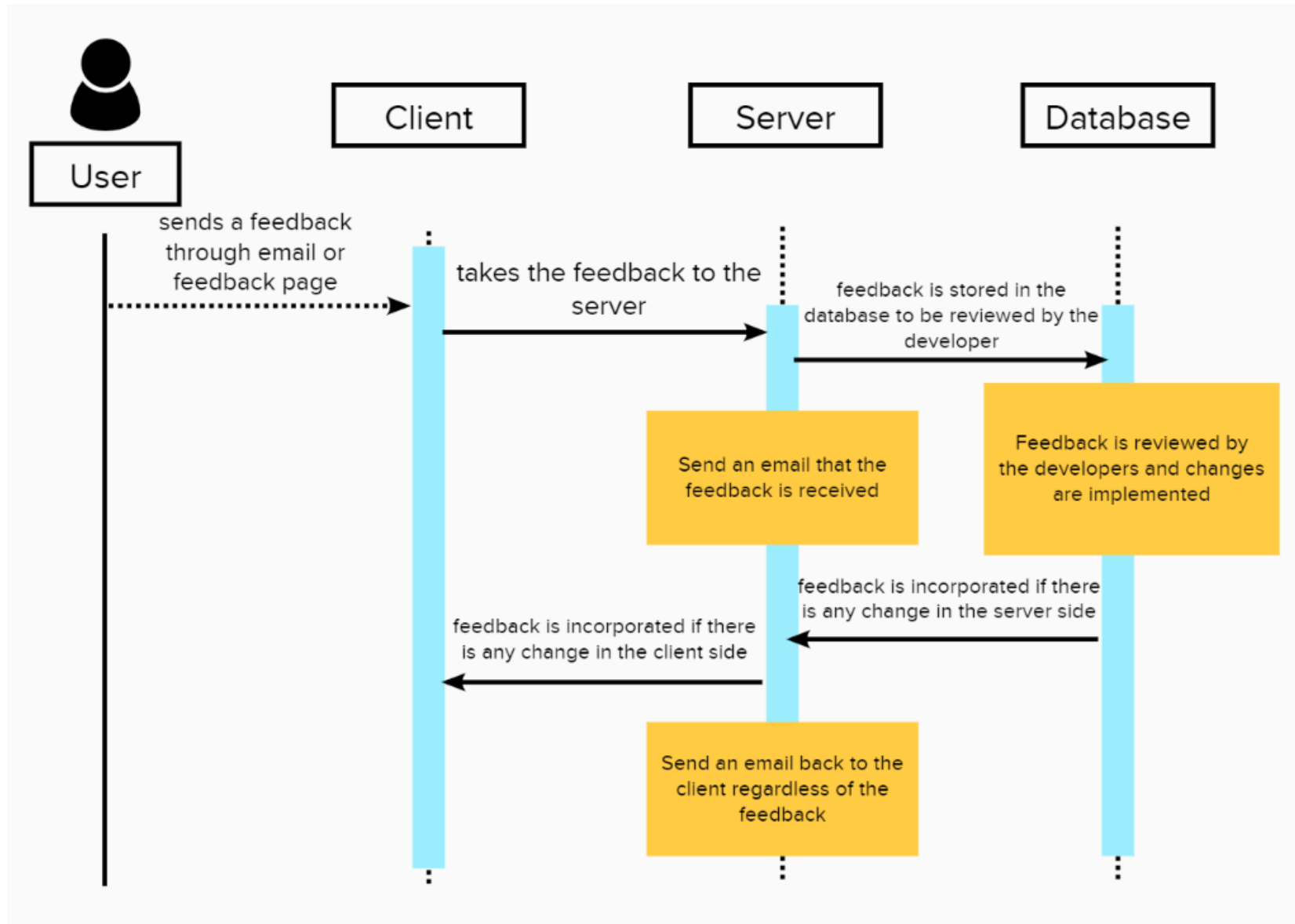
Sequence of events when user submits the journal for analysis



Sequence of events when user saves a journal entry

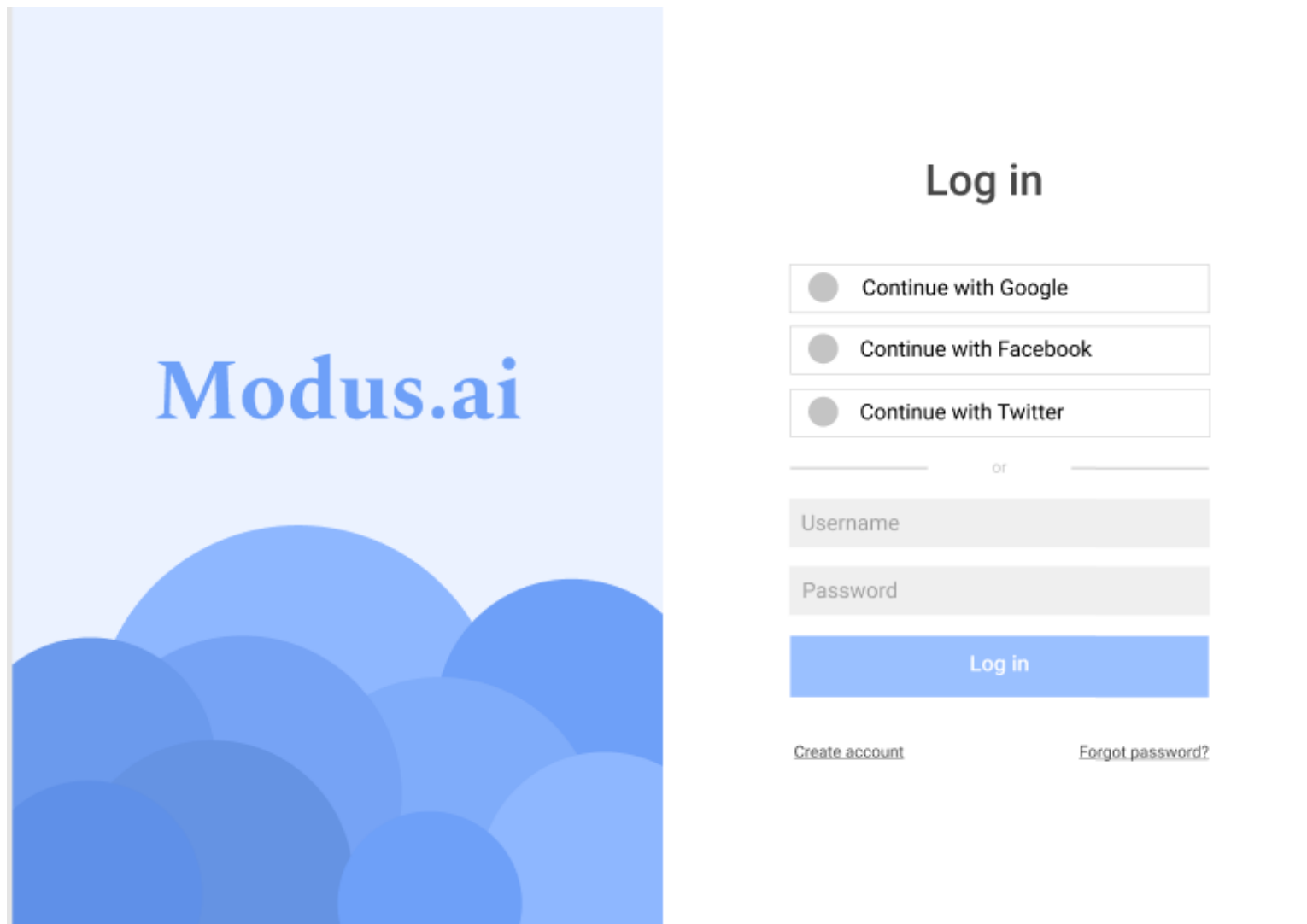


Sequence of events when user provides feedback to Modus.ai



UI Mockups

Login Page



The image shows a UI mockup of a login page. On the left, there is a light blue rectangular area with the text "Modus.ai" in a large, blue, serif font. Below the text, there are several overlapping blue circles of varying shades, creating a decorative wave-like pattern. To the right of this area is a white rectangular box containing the login form. The form has a title "Log in" in a bold, black, sans-serif font. Below the title, there are three social login options, each with a gray circular icon and a text label: "Continue with Google", "Continue with Facebook", and "Continue with Twitter". These options are separated by a horizontal line with the word "or" in the center. Below the line, there are two input fields: "Username" and "Password", both with light gray borders and placeholder text. Below the input fields is a blue button with the text "Log in" in white. At the bottom of the form, there are two links: "Create account" and "Forgot password?", both in a small, black, sans-serif font.

Modus.ai

Log in

☐ Continue with Google

☐ Continue with Facebook

☐ Continue with Twitter

or

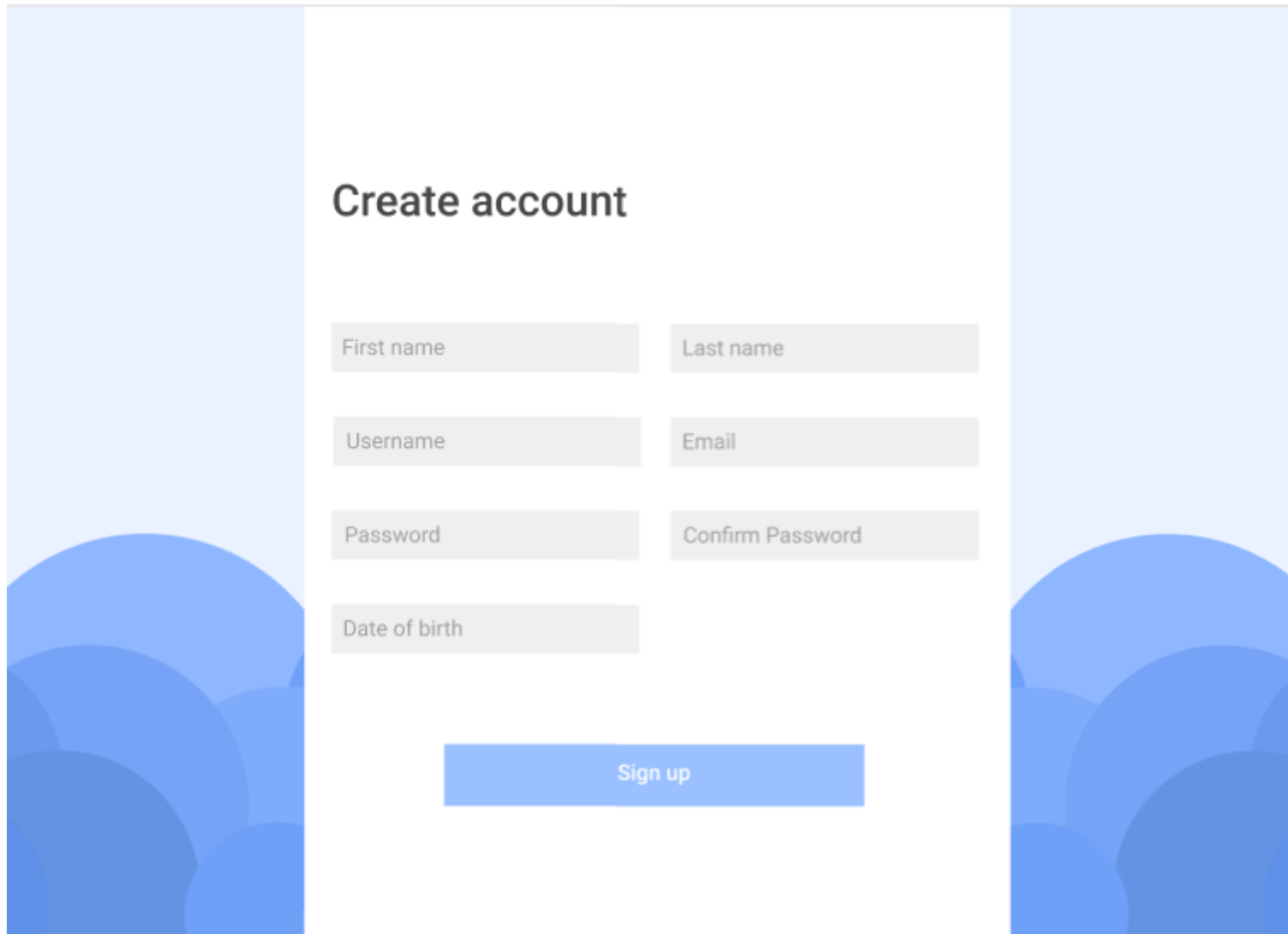
Username

Password

Log in

[Create account](#) [Forgot password?](#)

Signup Page



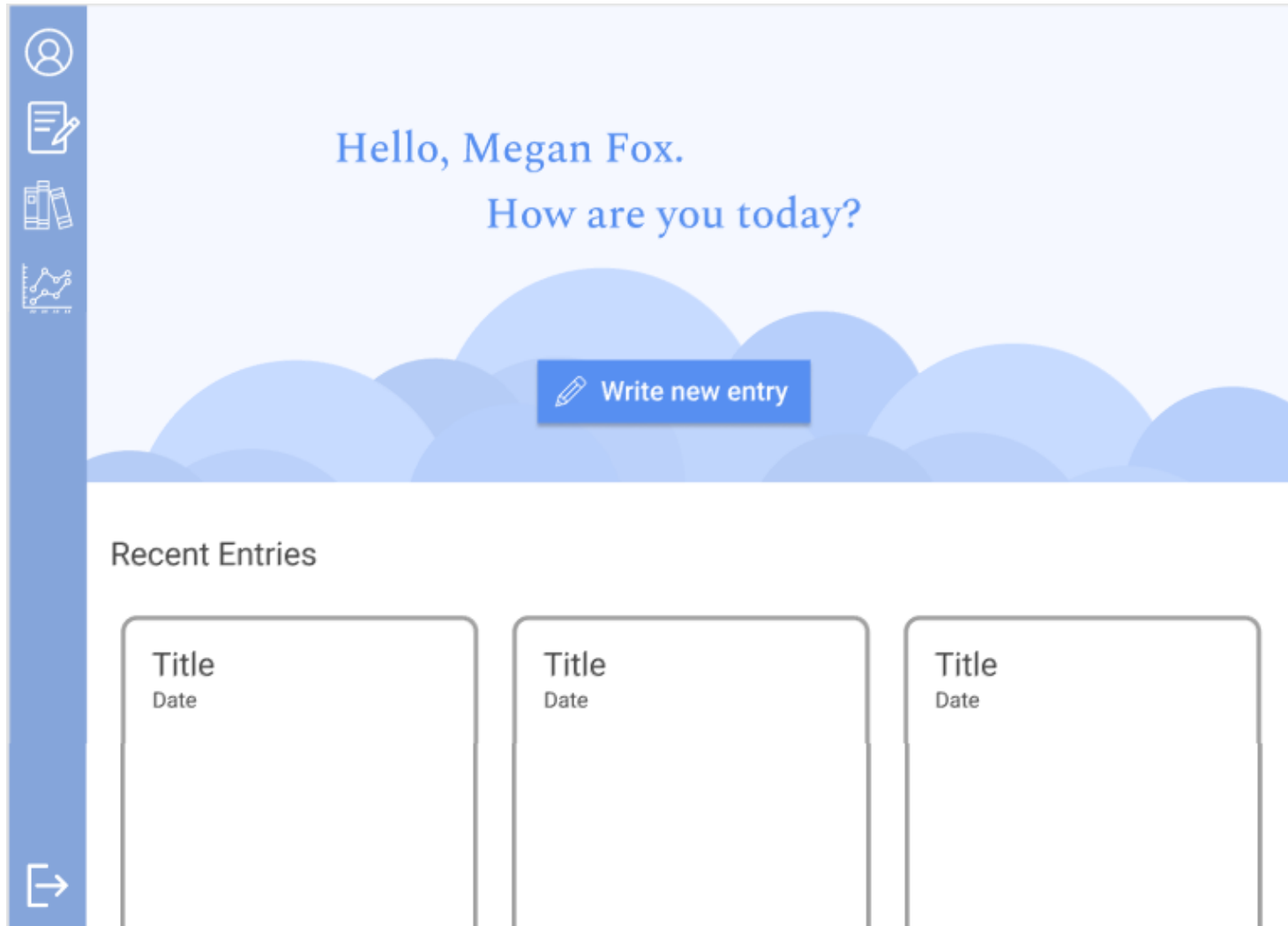
The form is titled "Create account" and is set against a light gray background. It is flanked by two vertical panels, each featuring a light blue background with a decorative pattern of overlapping blue circles at the bottom. The form fields are arranged in a grid: "First name" and "Last name" in the first row, "Username" and "Email" in the second, "Password" and "Confirm Password" in the third, and "Date of birth" in the fourth row. A blue "Sign up" button is centered below the fields.

Create account


First name	Last name
Username	Email
Password	Confirm Password
Date of birth	


Sign up


Main Dashboard





New Entry Page





















Save


Analyze 





Formats ▾








































































































Title

Date