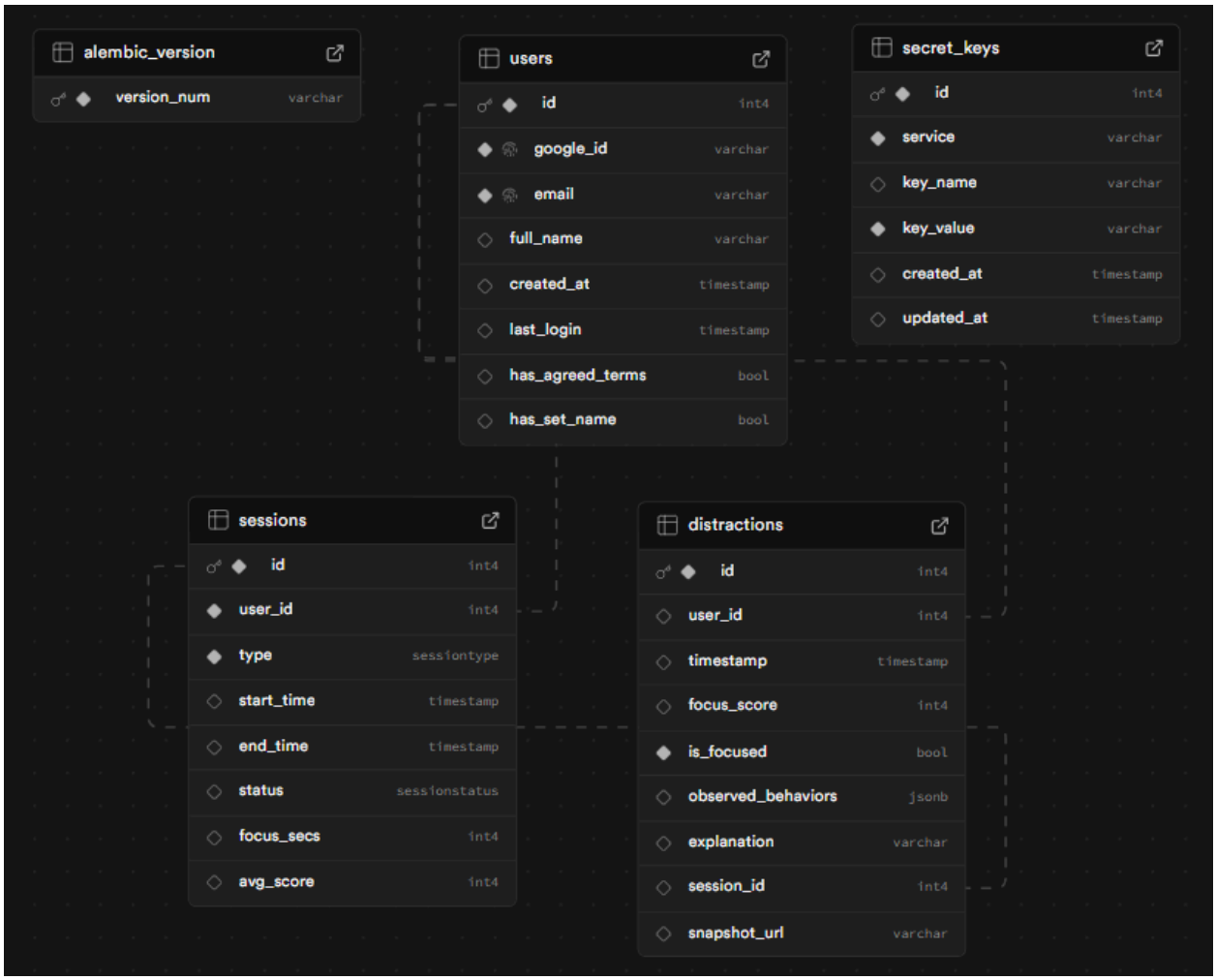
nagAI: Product Requirements

Wooyoung Jung, Jiwoo Kim, Joohyoung Jun
CSE 416, Spring 2025

Functional Requirements:
Detailed list of all functional requirements grouped by feature or page.

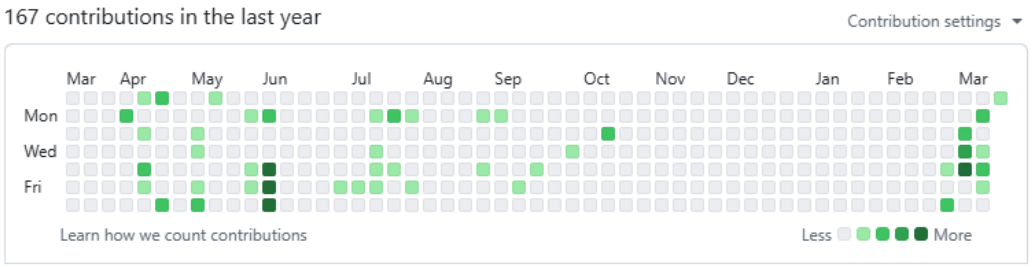
0. Navbar:
    - 0.1: The navigation bar shall allow the user to move to the desired page when the name of the page is clicked.
    - 0.2: The ‘NAGai’ logo on the top left shall navigate to the Main Page when clicked.
    - 0.3: The button with the user’s name, on the top right, shall navigate to the My Account’ page when clicked.
1. Create Account / Log In Page
    - 1.1 User Input & Buttons
        - ~~1.1.1: The page shall allow the user to enter their credentials (username and password).~~
        - 1.1.2: The page shall have a "Continue with Google" button for account creation or login.
    - 1.2 Handling Google OAuth Login
        - 1.2.1: When the "Continue with Google" button is clicked, a popup window shall appear.
        - 1.2.2: If the google_id returned by OAuth does not exist in our database:
            - 1.2.2.1: Create a new user account with:
                - google_id
                - full_name
                - created_at (current timestamp)
                - last_login (current timestamp)
        - 1.2.3: If the google_id already exists, log in the associated user and update last_login.
    - 1.3 Handling New User Sign-Ups (Terms & Conditions)
      1.3.1: If the user is new, they must agree to:
        - 1.3.1.1: Terms & Conditions
        - 1.3.1.2: Privacy Statement
        - 1.3.1.3: The "Continue" button shall be greyed out unless both checkboxes are checked.
        - 1.3.1.4: Clicking "Cancel" navigates the user back to the login page.

- 1.4 Post-Signup Name Editing
    - 1.4.1: After successful signup, show a confirmation page displaying the user's full_name from their Google account.
    - 1.4.2: The user can edit their name in a text field before continuing (30 characters max).
    - 1.4.3: The Gesture Help overlay shall be displayed for first-time login users, so that they can be guided with instructions to use the app.
- 1.5 Error Handling
    - 1.5.1: If "Continue with Google" fails, display a red error message:
        - "Failed to authenticate with Google."
    - 1.5.2: Redirect the user back to the login page.
- 1.6: The page shall securely store new google_id-full name pairs in a database.
    - The google_id-full name pairs shall be stored in a 'user info' table within the MySQL database.



        - Final SQL schemas for our Database.
2. Main Page:
    - 2.1: There shall be a 16:9 square box in which the user's webcam feed is shown.
        - At least 30% of the bottom of the screen shall display the user's desk, and the rest shall display the user's face and body, and their surroundings.
    - 2.2: The 'Distractions' button shall display the 'Distraction Log' modal when clicked.
        - 2.2.1: When a new distraction is detected, the distraction count in the right side of the button will be incremented.
    - 2.3: There shall be a round-shaped timer indicating the remaining time of the current session. The outer border of the circle decreases in length correspondingly.
        - 2.3.1: For focus sessions, the timer shall be highlighted in red (#EB6565).
            - Timer begins counting down from 25:00 all the way down to 00:00.
        - 2.3.2: For breaks, the timer shall be highlighted in blue (#65B3EB).
            - Timer begins counting down from 5:00 all the way down to 00:00.
    - 2.4a: When the distraction level is < 40, the user shall be considered distracted, and the distraction count shall be incremented.
            - The user shall be considered:
            - 2.4a.1.1: 'focused' in the following states:
                a. Sitting still and appearing to be reading/writing material on their desk
                b. Not laying down
                c. Present in front of the webcam
            - 2.4a.1.2: 'distracted' in the following states:
                a. Looking at their phones for > 1 min
                b. Away from keyboard for > 1 min
                c. Laying down on desk/slouching for > 1 min
                d. Appearing to be loitering/socializing with friends for > 2 min
            - Note: the following factors are not final. Due to the black-box nature of our AI API, it may not even be possible to determine focus based on those factors alone.
    - 2.5: There shall be a focus time box bottom right of the page which displays the accumulated focus time on it.

- 2.5.1: The focus time written in the box shall be renewed to 00:00 every midnight.
    - 2.5.1.1: If the user is focusing after midnight from the previous day, the focus time will be renewed after the current focus time is over.
- 2.5.2: If the user clicks on the ‘focus time’ box, it shall show a modal with a calendar view of focus times for the entire month
    - And, the user shall be able to select which month to view their focus data for.



-
    - Reference: GitHub contributions page

3. Distraction Log (displayed in a modal over the current window)
    - 3.1: The log shall be shown in chronological order.
        - 3.1.1: The most recent distraction event shall be shown at the top by default.
        - 3.1.2: By clicking on the switch button, the order of the entries shall be inverted, displaying the oldest (least recent) distraction event on top.
    - 3.2: The distraction log shall be in a table format.
        - 3.2.1: Each row shall show each distraction date of distraction, a description of the event, and the estimated focus score.
        - 3.2.2: If the user clicks on a row, a detailed view shall appear in the same modal. It shall display the following information about a distraction event:
            - The snapshot of the webcam during the distraction event.
                a. The snapshot shall be stored for 7 days, and deleted after that.
            - The timestamp of the captured snapshot.
            - A textual description of the distraction event (e.g., looking at phone, sleeping, etc).
                a. This description shall be part of an output prompt by the AI API we are planning to use.

        - There shall also be a ‘X’ or ‘close’ button in the top-right corner of the detailed view.
            a. When clicked, the entire distraction log shall be displayed in the same modal.
    - 3.2.3: There shall be a ‘X’ or ‘close’ button in the top-right corner of the distraction log.
        - When clicked, the modal shall be closed, and shall navigate to the Main Page.
4. My Account (displayed in a modal over the current window)
    - 4.1: There shall be an editable text field displaying the user’s full name.
        - 4.1.1: The user shall be able to edit their full name (30 characters max).
    - 4.2: The user shall be able to sign out by entering the sign out button.
        - If signed out successfully, the system shall navigate to the Create Account / Login Page.
    - 4.4: There shall be a ‘Continue’ button that saves the changes and closes the modal when clicked.
        - If changes are saved successfully, the system shall navigate to the Main Page.
        - 4.4.2: If changes fail to be saved, display an error message “Failed to save changes.” in red at the bottom of the modal, above the button bar.
    - 4.5: There shall be a ‘Cancel’ button that discards the changes and closes the modal when clicked.

Non-Functional Requirements: Usability, performance criteria, etc.

1. Frontend: The frontend shall be implemented using TypeScript React.
    a. The source code shall be modularized into individual files for components (Navbar, FocusTimer, etc) and pages (LogIn, MainPage, DistractionLog, etc).
    b. As mentioned above, all forms of user input shall be validated (in terms of input length, type, etc) in the frontend before being passed to the backend through the server API. This is to ensure security and prevent undefined behavior.
2. Backend: The backend shall be implemented using an FastAPI server and PostgreSQL database.
    a. The backend API shall provide the ability to GET, POST, and PUT data from the frontend to the backend, and vice versa.
    b. The backend shall encrypt and encapsulate sensitive data, such as username-password pairs.

    c. The backend shall be able to store reasonable amounts of data and handle some volume of traffic.
    d. While it does not require as much rigor as a production-grade application, the backend shall be able to handle some form of concurrency through the use of Javascript keywords such as ‘async’ and ‘await’.
3. Styling: The frontend shall have proper styling, such as div centering, alignment, adaptive width, and good visibility. Buttons should be easily reachable and intuitive to use. Input fields and dropdown menus should be easy to manipulate even in small mobile-based viewports.