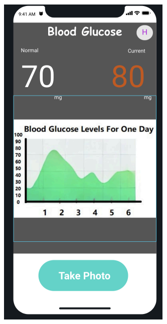
Software Components

1. Render Functions (React Components)

Each of these render functions will be React components. They handle UI rendering tasks.

Components:



- `renderWelcomePage()`
    - **Utility**: The main page that the user interacts with when launching the application. It provides an entry point to other features. As well as displaying the user's data after it is returned.
    - renderSettings() and renderCamera() are subcomponents of renderWelcomePage()
- `renderSettings()`
    - **Utility**: Renders the settings page where users can configure options such as Dexcom integration or other preferences.
    - **Sub-components**:
        - `exitSettings()`: Used to exit the settings page and return to the previous page.
        - `renderViewLogs()`: Displays logs, of Meals and glucose data
        - `renderSetupGlucose()`: Allows the user to set up glucose monitoring by entering relevant data.
        - `loginToDexCom()`
            - **Utility**: Logs the user into Dexcom services, enabling them to access glucose data.
- `renderCamera()`
    - **Utility**: Renders a camera interface, allowing users to take a picture of their food.
    - **Sub-components**:
        - `takePicture()`: Allows users to capture a picture within the camera interface.
        - `exit()`: Exits the camera interface and returns to the previous screen.

2. Dexcom API Functions

These functions are responsible for communicating with the Dexcom API to manage glucose data.

Components:

- `storeGlucoseData()`
  - **Utility**: Stores glucose data received from the Dexcom API into the database.
- `fetchGlucoseData()`
  - **Utility**: Fetches glucose data from the Dexcom API for use in the app.
- `exchangeCodeForToken()`
  - **Utility**: Exchanges authentication codes for an access token to interact with Dexcom's API.

3. OpenAI API Functions

These functions are responsible for integrating with OpenAI to process data.

Components:

- `OpenAICall()`
  - **Utility**: Sends a request to OpenAI's API to process for amount of carbs in the picture

4. Data Storage and Processing

These functions handle data storage and backend processing.

Components:

- `GenerateDistributableLogs()`
  - **Utility**: Creates a CSV from the local database of glucose and insulin.
- `FetchGlucose()`
  - **Utility**: Fetches glucose data from the database for processing and analysis.
- `logData()`
  - **Utility**: Logs data into the database, likely in structured form, for future retrieval.
- `calculateGlucose()`
  - **Utility**: Calculates glucose levels based on data fetched from the Dexcom API or stored data, potentially providing insights or predictions.

Users

1. Admin User: Administers application settings and configurations, defines access groups and user permissions, handles system backups, analyzes application performance metrics and user engagement, configures and monitors API integrations, and oversees all aspects of the application and infrastructure.
2. End User (Subscriber): uploads food images to receive carbohydrate and glucose-insulin ratio feedback, views historical data and recommendations, and authenticates Dexcom account.

Capacity & Performance

1. Server Capacity
    a. Limitations on the number of concurrent users it can handle effectively. This is dependent on the server's hardware specifications and the applications design. Because of our limited intended outreach and non-intensive resource allocation, this will not be a current concern.
    b. The number of external API requests (Dexcom and OpenAI) may be limited based on service agreements or excessive token costs.
2. Storage Capacity: The database will have a finite amount of storage for user data, image uploads, and historical records. Once the limit is reached, data management strategies (like archiving or purging old data) can be needed.
3. Query Performance: As the amount of data grows, query performance may degrade, especially if the database isn't optimized. This can lead to slower response times for users retrieving historical data logs or recommendations.
4. Upload/Download Speeds: Users' internet speeds can affect how quickly they can upload images and receive feedback. Slow connections may lead to timeouts or a poor user experience.
5. Image Size: Larger image files may take longer to upload and process, impacting user experience. There may also be limits on file sizes imposed by the API.
6. Processing Time: The time taken by the OpenAI API to analyze images and return results can introduce delays, especially during peak usage times.
7. Device Performance: The app's performance can vary based on the user's device capabilities (CPU, RAM). Older or less powerful devices may struggle with processing or displaying data efficiently. Our application is not resource intensive and should work efficiently on modern devices.
8. Storage Space: Users may have limited storage space on their devices for app data and images, potentially affecting the app's functionality, like meal and insulin logging.