Task 3.2: Coding Conceptualization - Lift lit

Feature Conceptualization: Calendar Sync

The Calendar Sync feature is a crucial component of the *Lift Lit* application, allowing users to seamlessly integrate their existing schedules with their fitness routines. This feature ensures that workouts are intelligently scheduled without disrupting academic or personal commitments.

Data Collection

The Calendar Sync feature relies on the following data points:

1. **Calendar Events**: Data such as event titles, timings, and durations will be collected from external platforms like Google Calendar and Notion.
2. **User Availability**: Users can manually input time preferences or mark periods as "non-workout times" to customize schedules.
3. **Fitness Goals**: The app will consider the user's fitness preferences (e.g., cardio, strength training) and recommended durations for each workout session.

To collect this data, the app will use APIs provided by platforms like Google Calendar and Notion. Users will have to grant permission for the app to access calendar events via authentication, ensuring data security.

Results Presentation

The results of this feature will be presented as:

1. A **weekly workout schedule**, showing synced events alongside allocated workout sessions.
2. **Push notifications** to remind users of upcoming workout sessions.
3. A **progress dashboard** displaying adherence to scheduled workouts and suggesting adjustments for better time management.

Coding Conventions

The following coding conventions will be followed:

1. **Modular Design**: The feature will be developed as an independent module to allow easy integration with other app features.
2. **Clean and Readable Code**: Descriptive variable names and comments will ensure clarity.
3. **Error Handling**: Robust mechanisms will handle API call failures or incorrect data input.
4. **Security Measures**: Sensitive data such as API keys will be securely stored, and tokens will be used for user authentication.

Architecture of the Calendar Sync Feature

When the Code Runs

The Calendar Sync feature operates in three main phases:

1. **Initialization Phase**: Runs during the initial app setup when the user connects their calendar. This phase retrieves calendar permissions and sets up API credentials.
2. **Sync Phase**: Runs periodically (e.g., once daily) to fetch updated calendar events and integrate them into the app's scheduling algorithm.
3. **Real-Time Updates**: Triggered when users add or modify events in their calendar or the app's schedule, ensuring that workouts remain optimized.

Starting and Ending Points

1. **Starting Point**: The process starts when the user selects the "Sync Calendar" option in the app and chooses their preferred platform (e.g., Google Calendar).
    - This triggers an authentication request, followed by an API call to fetch calendar events.
2. **Ending Point**: The process ends once the fetched events are successfully analyzed, and workout sessions are scheduled without overlapping important tasks.
    - The app then displays the updated schedule on the dashboard and sends a confirmation notification to the user.

Design Choices

- **Starting Point Justification**: By initiating the process upon user action, the app maintains control over when and how external data is accessed, ensuring user consent and system stability.
- **Ending Point Justification**: The feature ends with visible feedback to the user, reinforcing trust and providing clarity on the scheduling outcomes.