Weber Brain Application: Light Treatment Mobile App Scope and Architecture

# ## 1. Scope

The Weber Brain Application mobile app will serve as a user-friendly interface for controlling Weber Brain Device Controller and the connected modules. The app will be developed using Flutter for cross-platform compatibility (iOS and Android) and will incorporate Bluetooth Low Energy (BLE) for device communication and control.

## ### 1.1 Key Features

1. User Authentication and Profile Management
2. BLE Device Pairing and Management
3. Light Treatment Session Control  
   (start, stop, adjust intensity, output power, frequency and duration)
4. Treatment Schedule Management
5. Light Treatment Programs
6. Session History and Reporting
7. User Settings and Preferences
8. Device Firmware Updates
9. Reminders and Notifications

# ## 2. Architecture

The mobile app will follow a clean architecture pattern, separating concerns and ensuring scalability and testability.

## ### 2.1 High-Level Architecture

1. Presentation Layer (UI)
2. Business Logic Layer (BLoC)
3. Domain Layer
4. Data Layer
5. Device Communication Layer (BLE)

## ### 2.2 Detailed Architecture

### #### 2.2.1 Presentation Layer

* Flutter Widgets
* Screen-specific Views (e.g., Device Control, Schedule, History)
* Reusable UI Components (e.g., LED Control Widget, Timer Widget)

### #### 2.2.2 Business Logic Layer

* Flutter BLoC for state management
* Use Cases (e.g., StartTreatmentSession, UpdateDeviceSettings)

### #### 2.2.3 Domain Layer

* Business Entities (e.g., TreatmentSession, LightProgram, DeviceSettings)
* Repository Interfaces

### #### 2.2.4 Data Layer

* Remote Data Source (API Client for user data and settings sync)
* Local Data Source (SQLite or Hive for offline storage)
* Repository Implementations

### #### 2.2.5 Device Communication Layer

* BLE Service (for device control and status updates)
* Command Protocols (for sending instructions to the device)

## ### 2.3 Key Components

1. BLE Manager: Handles device discovery, connection, and communication.
2. Treatment Controller: Manages treatment sessions, including timing and intensity control.
3. Schedule Manager: Handles treatment scheduling and reminders.
4. Program Manager: Manages predefined and custom light treatment programs.
5. Session History: Tracks and stores treatment session data.
6. Firmware Update Service: Manages device firmware updates.
7. API Client: Manages communication with the backend for data sync and user management.
8. Local Database: Stores user preferences, schedules, and session history.
9. Authentication Manager: Handles user authentication and token management.

# ## 3. Technology Stack

* Framework: Flutter
* State Management: Flutter BLoC
* Local Database: Hive
* BLE Communication: flutter\_blue\_plus
* API Communication: Dio with interceptors for request/response logging and error handling

## Secure Storage:

Base: flutter\_secure\_storage

Enhanced: Custom encryption layer using encrypt package

Hardware-backed (optional): Keychain Services (iOS) / StrongBox Keymaster (Android)

## Authentication:

* Token-based authentication for API requests
* Biometric authentication using local\_auth package

## Encryption:

* AES encryption for sensitive data using encrypt package
* SSL/TLS for network communications
* Dependency Injection: get\_it
* Routing: auto\_route
* Logging: logger package for debug builds, stripped in release builds
* Error Reporting: Sentry.io for crash reporting and performance monitoring
* Code Obfuscation: Flutter's built-in R8 compiler for Android and Bitcode for iOS
* Certificate Pinning: Implemented in Dio for API requests to prevent man-in-the-middle attacks
* Secure Window: flutter\_windowmanager to prevent screenshots and screen recording on Android

## Security Measures:

1. **Data at Rest:**

* Sensitive data encrypted before storing using AES encryption
* Utilizes Keychain (iOS) and EncryptedSharedPreferences (Android) via flutter\_secure\_storage
* Optional: Integration with hardware security modules for high-risk data

1. **Data in Transit:**

* All network communications over HTTPS
* Certificate pinning to prevent man-in-the-middle attacks

1. **Authentication and Authorization:**

* JWT for API authentication
* Biometric authentication for app access
* Role-based access control for features and data

1. **Secure Coding Practices:**

* Input validation and sanitization
* Prevention of SQL injection in local database queries
* Regular dependency updates to patch known vulnerabilities

1. **Device-level Security:**

* Detection of root/jailbreak status
* Secure window to prevent screenshots of sensitive data
* Automatic log-out after a period of inactivity

1. **Audit and Logging:**

* Secure audit logging for sensitive operations
* Remote logging for security events with PII removed

## Compliance:

* GDPR compliance measures for EU users
* CCPA compliance for US users (?)

# ## 4. Data Flow

1. User interacts with UI (e.g., adjusts light intensity)
2. UI triggers events in BLoC
3. BLoC processes events and calls appropriate Use Cases
4. Use Cases interact with Repositories or BLE Manager
5. BLE Manager sends commands to the light treatment device
6. Device status updates are received via BLE and propagated back through the layers
7. Session data is stored locally and synced with the backend when online

# ## 5. Security Considerations

* Implement secure local storage for user data and device settings
* Use HTTPS for all API communications
* Implement certificate pinning
* Secure BLE communication with encryption if supported by the device
* Implement proper error handling to prevent unauthorized device control

# ## 6. Testing Strategy

* Unit Tests for BLoC, Use Cases, and Repositories
* Widget Tests for UI components, especially custom controls
* Integration Tests for key user flows (e.g., starting a treatment session)
* BLE Mock Service for simulating device interactions during development

# ## 7. Deployment and Distribution

* CI/CD pipeline for automated building and testing
* Beta distribution through TestFlight (iOS) and Google Play Beta (Android)
* App Store and Google Play Store for production releases

# ## 8. Future Considerations

* N/A