Cig++ - Technical Documentation & Code Appendix

Date: January 2025

Team: BNB Tech

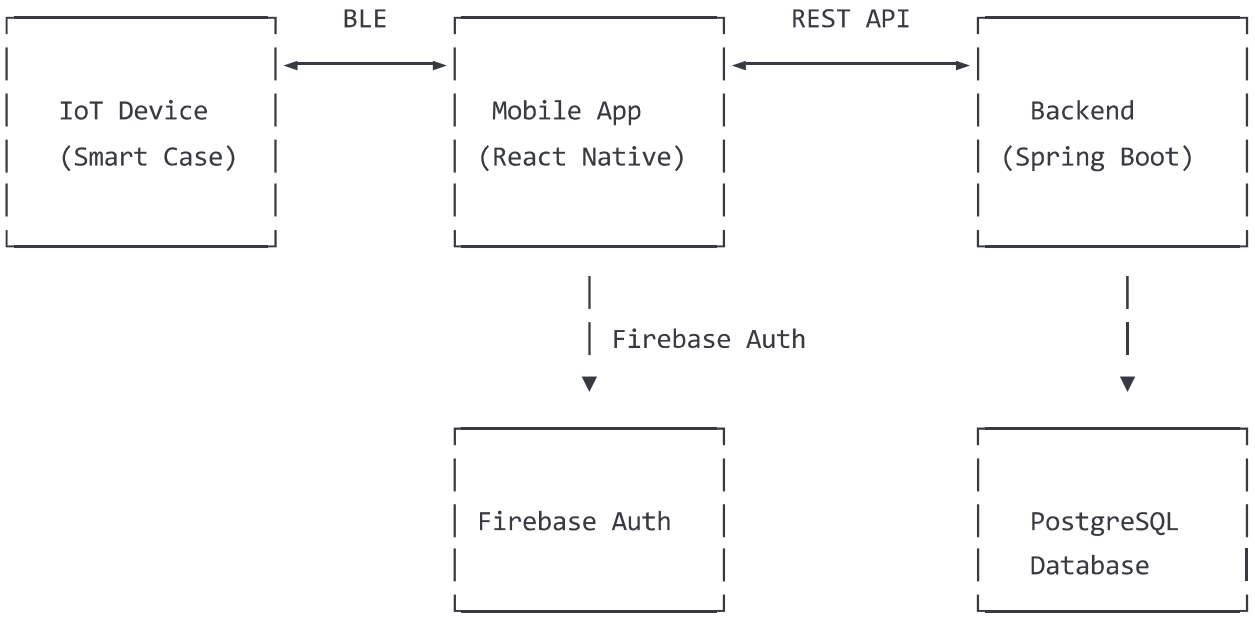
Members: Tomer Geva, Omer Shalev

Table of Contents

---

1. System Architecture

1.1 High-Level Architecture



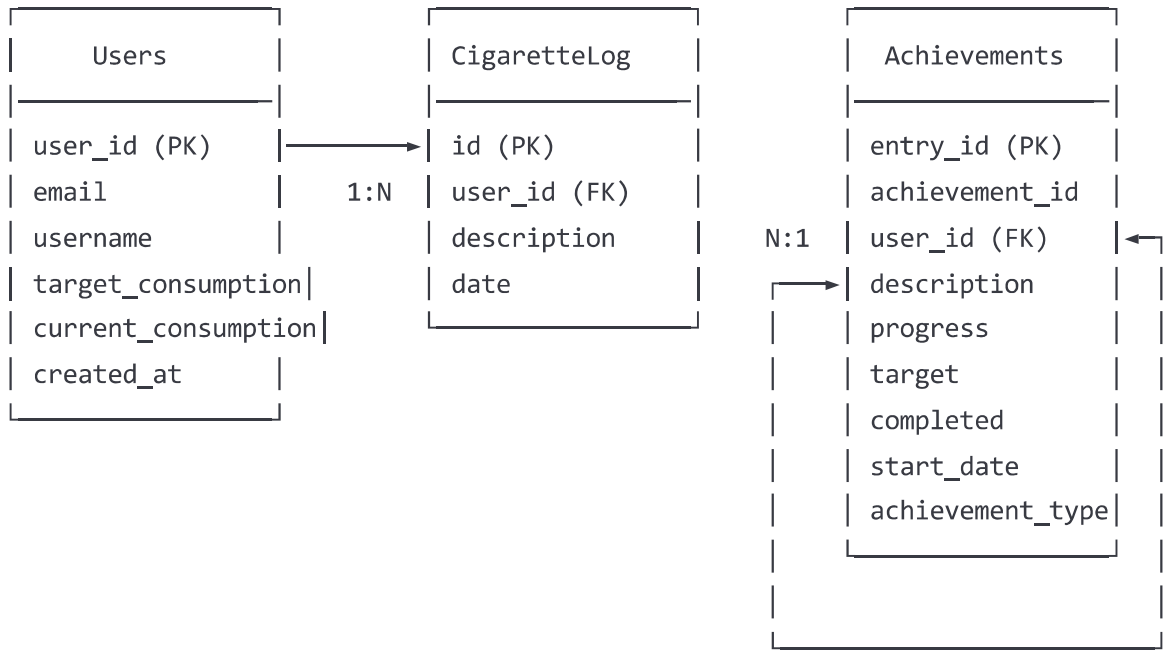
1.2 Technology Stack Summary

Layer	Technology	Purpose
Frontend	React Native + Expo	Cross-platform mobile application
Authentication	Firebase Auth	User authentication and authorization
Backend	Spring Boot (Java)	REST API server and business logic
Database	PostgreSQL	Data persistence
IoT Communication	Bluetooth Low Energy (BLE)	Device-to-app communication
Embedded	C++ (Arduino IDE)	Smart case firmware

---

2. Database Design

2.1 Entity Relationship Diagram



2.2 Database Tables

2.2.1 CigaretteLog Table

```sql
CREATE TABLE cigarette_log (
    id BIGSERIAL PRIMARY KEY,
    user_id VARCHAR(255) NOT NULL,
    description VARCHAR(255),
    date TIMESTAMP NOT NULL,
    CONSTRAINT fk_user FOREIGN KEY (user_id) REFERENCES users(user_id)
);
```

2.2.2 Achievements Table

```sql
CREATE TABLE achievements (
    entry_id BIGSERIAL PRIMARY KEY,
    achievement_id BIGINT NOT NULL,
    user_id VARCHAR(255) NOT NULL,
    description VARCHAR(255) NOT NULL,
    progress INTEGER,
    target INTEGER,
    completed BOOLEAN,
    start_date TIMESTAMP,
    achievement_type VARCHAR(50),
    CONSTRAINT fk_achievement_user FOREIGN KEY (user_id) REFERENCES users(user_id)
);
```

---

3. API Documentation

3.1 Authentication

All API endpoints require Firebase JWT token in Authorization header:

```
Authorization: Bearer <firebase_jwt_token>
```

3.2 Cigarette Log Endpoints

3.2.1 Add Cigarette Log

- **Endpoint:** `POST /api/cigarettes`
- **Request Body:**

```json
{
    "userId": "string",
    "description": "string",
    "date": "2025-01-23T10:30:00Z"
}
```

- **Response:**

```json
{
    "id": 1,
    "userId": "user123",
    "description": "Morning cigarette",
    "date": "2025-01-23T10:30:00Z"
}
```

3.2.2 Get User's Cigarette Logs

- **Endpoint:** `GET /api/cigarettes/{userId}`
- **Response:**

```json
[
    {
        "id": 1,
        "userId": "user123",
        "description": "Morning cigarette",
        "date": "2025-01-23T10:30:00Z"
    }
]
```

3.2.3 Get Logs Between Dates

- **Endpoint:** `GET /api/cigarettes/{userId}/between?startDate={start}&endDate={end}`
- **Parameters:**
  - `startDate`: ISO 8601 timestamp
  - `endDate`: ISO 8601 timestamp

3.2.4 Delete Cigarette Log

- **Endpoint:** `DELETE /api/cigarettes/{id}`
- **Response:** Returns deleted log object

---

4. Frontend Architecture

4.1 Navigation Structure

```
AppNavigator
├── AuthStack (Unauthenticated)
│   ├── LoginScreen
│   └── RegisterScreen
│
└── MainStack (Authenticated)
    ├── HomeTabs
    │   ├── SummaryStack
    │   │   ├── SummaryScreen
    │   │   ├── CigarettesLogsModal
    │   │   ├── AchievementsModal
    │   │   └── DetailedStatsModal
    │   │
    │   ├── ChallengesScreen
    │   └── ProfileStack
    │       ├── ProfileScreen
    │       ├── BTModal
    │       ├── NotificationsModal
    │       ├── ChangeGoalsModal
    │       └── PersonalDetailsModal
```

4.2 State Management Architecture

4.2.1 Context Providers

- **AuthContext:** Manages user authentication state
- **PreferencesContext:** Manages user preferences and settings

4.2.2 Key Components

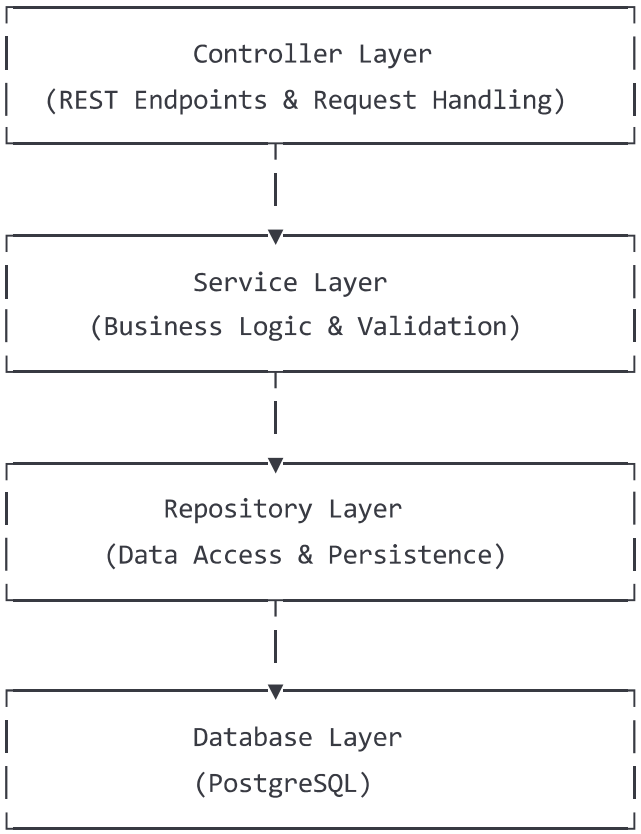
- **ProgressCircleCard:** Visual progress indicator
- **TouchableBox:** Reusable navigation cards
- **CustomInput:** Standardized input component
- **CustomButton:** Standardized button component

4.3 Screen Responsibilities

Screen	Responsibility
SummaryScreen	Main dashboard with progress overview
CigarettesLogsModal	CRUD operations for cigarette logs
AchievementsModal	Display and track user achievements
DetailedStatsModal	Advanced analytics and insights
ProfileScreen	User profile and settings management

5. Backend Architecture

5.1 Layer Architecture



5.2 Key Classes

5.2.1 Controllers

- **CigaretteLogController:** Handles cigarette logging endpoints

5.2.2 Services

- **CigaretteLogServiceImpl:** Business logic for cigarette management

5.2.3 Models

- **CigaretteLog:** Entity representing a cigarette log entry
- **Achievement:** Entity representing user achievements

5.2.4 Security

- **FirebaseAuthenticationFilter:** JWT token validation
- **SecurityConfig:** Spring Security configuration

---

6. Technical Flow Diagrams

6.1 User Registration Flow

```mermaid
sequenceDiagram
    participant U as User
    participant A as Mobile App
    participant F as Firebase Auth
    participant B as Backend
    participant D as Database

    U->>A: Enter credentials & preferences
    A->>A: Validate input
    A->>F: Create user account
    F-->>A: Return user credentials
    A->>B: Save initial preferences
    B->>D: Store user data
    D-->>B: Confirm storage
    B-->>A: Success response
    A->>A: Navigate to main app
```

6.2 Cigarette Logging Flow

```mermaid
sequenceDiagram
    participant D as IoT Device
    participant A as Mobile App
    participant B as Backend
    participant DB as Database

    D->>A: BLE: Cigarette detected
    A->>A: Create log entry
    A->>B: POST /api/cigarettes
    B->>B: Validate & process
    B->>DB: Store log entry
    DB-->>B: Confirm storage
    B-->>A: Success response
    A->>A: Update UI
```

6.3 Achievement System Flow

mermaid

```
flowchart TD
    A[User Action Logged] --> B{Check Achievement Criteria}
    B -->|Criteria Met| C[Update Achievement Progress]
    B -->|Criteria Not Met| D[Continue Tracking]
    C --> E{Achievement Completed?}
    E -->|Yes| F[Mark as Completed]
    E -->|No| D
    F --> G[Show Achievement Notification]
    G --> H[Update UI]
    D --> I[End]
    H --> I
```

---

7. Code Structure

7.1 Frontend Structure

```
src/
├── components/           # Reusable UI components
│   ├── CustomButton.js
│   ├── CustomInput.js
│   ├── ProgressCircleCard.js
│   └── TouchableBox.js
├── contexts/             # React contexts
│   ├── AuthContext.js
│   └── PreferencesContext.js
├── navigation/           # Navigation setup
│   ├── AppNavigator.js
│   ├── AuthStack.js
│   └── ProfileStack.js
├── screens/              # Screen components
│   ├── Login/
│   ├── Register/
│   ├── Summary/
│   └── Profile/
├── constants/            # App constants
│   └── theme.js
└── config/              # Configuration files
    └── firebase/
```

7.2 Backend Structure

```
src/main/java/com/bech/cigpp/
├── controller/           # REST controllers
│   ├── CigaretteLogController.java
│   └── dto/              # Data Transfer Objects
├── service/              # Business logic
│   ├── api/              # Service interfaces
│   └── impl/             # Service implementations
├── model/                # Entity classes
│   ├── CigaretteLog.java
│   └── achievement/
├── repository/           # Data access layer
│   └── CigaretteLogRepository.java
├── config/               # Configuration classes
│   ├── SecurityConfig.java
│   └── FireBaseConfig.java
├── firebase/             # Firebase integration
│   └── FirebaseAuthenticationFilter.java
├── exception/            # Exception handling
│   ├── GlobalExceptionHandler.java
│   └── ResourceNotFoundException.java
└── util/                # Utility classes
    └── CigaretteLogMapper.java
```

---

8. Work Division

8.1 Team Responsibilities

Tomer Geva

IoT Device Integration:

- Bluetooth Low Energy (BLE) communication protocol
- Device pairing and connection management
- Real-time data synchronization between device and app
- Embedded system development (C++/Arduino)

Backend API Development:

- Spring Boot application architecture
- REST API endpoints design and implementation
- Database schema design and optimization
- Firebase JWT token authentication

Data Management:

- PostgreSQL database setup and management
- Data persistence and retrieval optimization
- Achievement tracking system backend
- User data security and privacy implementation

Omer Shalev

Mobile Application Development:

- React Native app architecture and navigation setup
- UI/UX implementation for all screens
- State management with React Context
- Firebase authentication integration
- Custom component development

User Experience:

- Achievement system implementation
- Progress tracking and visualization
- Statistics and analytics display
- Offline capability and data synchronization

8.2 Shared Responsibilities

- **System Integration:** Ensuring seamless communication between frontend and backend
- **Testing:** Unit testing, integration testing, and user acceptance testing
- **Documentation:** Technical documentation, user guides, and API documentation
- **Code Review:** Cross-reviewing each other's code for quality assurance
- **Deployment:** Production deployment and monitoring setup

This documentation serves as a comprehensive technical guide for the Cig++ project, covering all architectural aspects, implementation details, and team collaboration structure.