

# Fitness App: System Design Document

**Project Name:** Fitness App

**Release Version:** v1.0 (Initial MVP Release)

**Date:** October 26, 2023

**Prepared By:** Team FitnessApp

---

## Table of Contents

1. Introduction
  2. CRC Cards
    - User
    - Profile
    - Workout
    - Exercise
    - ExerciseAPIService
    - AuthenticationService
  3. System Architecture Diagram
  4. System Decomposition
  5. Error Handling and Exception Management
  6. Conclusion
- 

## 1. Introduction

This document provides an overview of the system design for the Fitness App, detailing the architecture, main classes and their responsibilities, interactions, and error-handling strategies. This initial design may evolve over time to accommodate new features or integrate improved solutions.

---

## 2. CRC Cards

### Class: User

- **Responsibilities:**
  - Register and log in users using Clerk.

- Manage user session information.
  - Retrieve and update user profile information.
  - **Collaborations:**
    - **AuthenticationService:** For validating user credentials.
    - **Profile:** For storing and managing user-specific data.
- 

## Class: Profile

- **Responsibilities:**
    - Store user profile data (e.g., name, email).
    - Enable user updates to profile information.
  - **Collaborations:**
    - **User:** Owned by a single user, interacts with the User class to link profile data.
    - **Database:** Saves and retrieves profile data in the backend database.
- 

## Class: Workout

- **Responsibilities:**
    - Add, remove, and manage exercises within a workout plan.
    - Save and retrieve workout plans for each user.
  - **Collaborations:**
    - **Exercise:** Uses exercises to build a workout plan.
    - **Database:** Interacts with the database for storing and retrieving workout data.
- 

## Class: Exercise

- **Responsibilities:**
    - Display and manage exercise information based on muscle groups.
    - Store exercise details (e.g., name, target muscle, equipment).
  - **Collaborations:**
    - **ExerciseAPIService:** Fetches exercise information from an external API.
    - **Workout:** Used by the Workout class to add exercises to a workout plan.
- 

## Class: ExerciseAPIService

- **Responsibilities:**
  - Connect to external API to fetch exercise data.

- Cache exercise data in the database to reduce repeated API calls.
- **Collaborations:**
  - **Exercise:** Supplies exercise data to the Exercise class.
  - **Database:** Stores cached exercise data for future access.

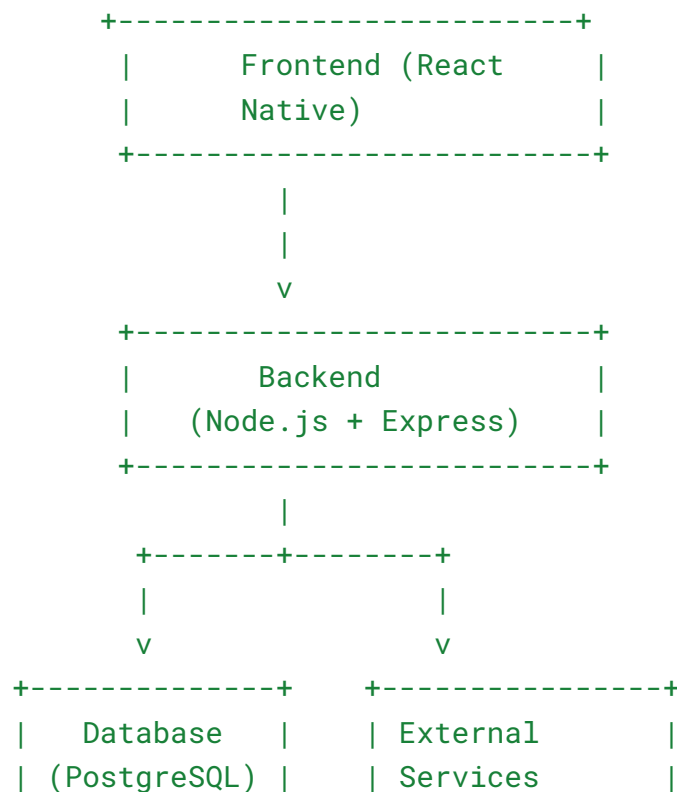
---

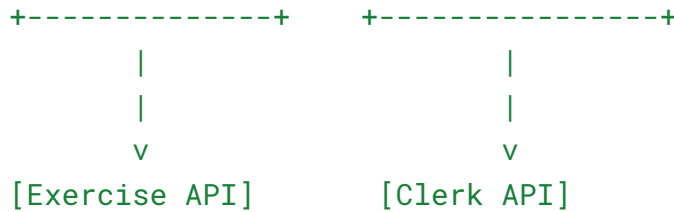
## Class: AuthenticationService

- **Responsibilities:**
    - Manage user registration, login, and session validation.
    - Ensure secure handling of authentication using Clerk.
  - **Collaborations:**
    - **User:** Verifies and manages user sessions.
    - **Clerk API:** Facilitates authentication with Clerk's service.
- 

## 3. System Architecture Diagram

The Fitness App architecture follows a **three-tier structure** consisting of **Frontend**, **Backend**, and **Database** components, with external services for authentication and exercise data.





## Explanation:

1. **Frontend:**
    - **Platform:** React Native.
    - **Purpose:** Manages UI and handles user interactions. It makes API requests to the backend to retrieve or modify data.
  2. **Backend:**
    - **Platform:** Node.js with Express.
    - **Purpose:** Serves as the main server that handles API requests, processes business logic, and communicates with the database and external services.
  3. **Database:**
    - **Platform:** PostgreSQL.
    - **Purpose:** Stores user data, workout plans, and cached exercise data.
  4. **External Services:**
    - **Clerk API:** Manages user authentication and session handling.
    - **Exercise API:** Supplies exercise data categorized by muscle group.
- 

## 4. System Decomposition

### Components and Roles:

1. **Frontend:**
  - Handles all user interactions, such as registering, logging in, and creating workout plans.
  - Sends API requests to the backend and displays data in a user-friendly format.
2. **Backend:**
  - Exposes API endpoints that the frontend interacts with for user, profile, and workout data.
  - Integrates with Clerk API for authentication and the Exercise API for exercise data.
3. **Database:**
  - Acts as the main repository for all app data, including users, profiles, workouts, and cached exercise details.

Each component serves a distinct purpose, supporting scalability and ease of maintenance. This modular structure also allows for independent updates and testing of each layer.

---

## 5. Error Handling and Exception Management

The error-handling strategy for the Fitness App addresses common issues and unexpected exceptions.

### Error Categories and Handling Strategy

1. **User Input Errors:**
  - **Description:** Occur when users enter invalid data.
  - **Handling:** Validate inputs on both frontend and backend, displaying user-friendly error messages if necessary.
2. **Authentication Errors:**
  - **Description:** Related to authentication, such as incorrect credentials.
  - **Handling:** Provide clear error messages for incorrect logins and redirect to login on session expiration.
3. **Network and API Errors:**
  - **Description:** Occur due to network issues or unavailable APIs.
  - **Handling:** Display a message to inform the user of connectivity issues and use cached data if available.
4. **Database Errors:**
  - **Description:** Issues related to accessing or modifying data in the database.
  - **Handling:** Retry operations if the database connection is lost and display a generic error if the issue persists.
5. **Unexpected Errors:**
  - **Description:** Unanticipated errors during runtime.
  - **Handling:** Log the errors for the development team and display a generic error message to the user.

### Anticipated Response Summary

- **User Input Errors:** “Invalid input. Please check your entries.”
- **Authentication Errors:** “Session expired. Please log in again.”
- **Network and API Errors:** “Network issues detected. Please check your connection.”
- **Database Errors:** “Technical issue encountered. Please try again later.”
- **Unexpected Errors:** “An unexpected error occurred. Please restart the app.”