Fitness Tracking App Design Document

1. Introduction

1.1 Purpose of the System

The Fitness Tracking App aims to provide users with a comprehensive platform to track their fitness journey. By logging workouts, setting goals, and analyzing progress, users can monitor their health effectively. The app is designed for individuals of all fitness levels, offering personalized feedback and actionable insights.

1.2 Design Goals

- **Usability**: Ensure an intuitive user interface for web and mobile platforms.
- **Scalability**: Support thousands of users concurrently with seamless performance.
- **Security**: Protect user data with industry-standard encryption and authentication.
- **Maintainability**: Facilitate easy updates and debugging through modular design.

---

2. High-Level Software Architecture

2.1 Subsystem Decomposition

1. **Authentication Module**: Manages user registration, login, and account security.
2. **Workout Management**: Handles CRUD operations for workout data.
3. **Goal Tracking**: Allows users to set, update, and monitor fitness goals.

4. **Progress Analytics**: Generates progress reports and trends.
5. **Admin Management**: Enables system admins to manage users and maintain the app.

2.2 Hardware/Software Mapping

- **Hardware**:
  - Backend server: AWS EC2 instances for scalable compute power.
  - Database server: AWS RDS for reliable storage.
- **Software**:
  - Frontend: React.js (web), Flutter (mobile).
  - Backend: Spring Boot (Java).
  - Database: MySQL.

2.3 Persistent Data Management

- Data is stored in a relational database with tables for `User`, `Workout`, `Goal`, `Progress`, and `Admin`.
- Object-Relational Mapping (ORM) is implemented using Hibernate.
- Daily backups ensure data recovery in case of failures.

2.4 Access Control and Security

- Role-based access control:
  - Regular users can manage their data.
  - Admins have elevated privileges.
- Password encryption with bcrypt.

- Secure communication through HTTPS.

2.5 Boundary Conditions

- Handle invalid user inputs with appropriate error messages.
- Graceful degradation during server downtime with offline storage for mobile users.

---

3. Low-Level Design

3.1 Object Design Trade-Offs

- **Choice of Frameworks**:
  - Spring Boot was selected for its scalability and ease of integration.
- **Database Type**:
  - MySQL provides structured data storage and supports complex queries.

3.2 Final Object Design

- Key entities:
  - `User`, `Workout`, `Goal`, `Progress`, and `Admin`.
- Relationships:
  - `User` to `Workout`: One-to-Many.
  - `User` to `Goal`: One-to-Many.
  - `User` to `Progress`: One-to-One.

3.3 Packages

- **com.fitness.app.entity**: Contains entity classes (`User`, `Workout`, etc.).
- **com.fitness.app.service**: Includes business logic for processing user requests.
- **com.fitness.app.controller**: Handles API endpoints.
- **com.fitness.app.repository**: Manages database operations.

3.4 Class Interfaces

- **User**:
  - Attributes: `id`, `name`, `email`, `password`.
  - Methods: `register()`, `login()`.
- **Workout**:
  - Attributes: `id`, `type`, `duration`, `intensity`.
  - Methods: `logWorkout()`.

3.5 Design Patterns

- **Singleton**: Ensures a single instance of the database connection.
- **Factory**: Creates objects like `User` or `Workout` dynamically.
- **Observer**: Notifies users of milestones or goal completion.

---

4. Improvement Summary

(To be completed during iteration 2.)

- Document changes and improvements made based on feedback.

---

4. Vision

Purpose

The Fitness Tracking App helps users achieve fitness goals by logging workouts, setting targets, tracking progress, and receiving feedback. It is designed for individuals at all fitness levels who want an intuitive platform to stay motivated and track their health journey.

Scope

The app provides features that allow users to:

- Register and securely log in to their accounts.
- Log workout details such as type, duration, and intensity.
- Set fitness goals with deadlines and track progress.
- View detailed analytics of their fitness journey.
- Receive personalized feedback to stay motivated.

Objectives

- Provide an intuitive platform for users of all fitness levels.
- Ensure data security and ease of access.

- Support scalability for thousands of users.
- Offer visual and actionable insights to track fitness progress effectively.

---

5. Supplementary Specs Document

Performance

- The system must handle 100 concurrent users with no noticeable lag.

Security

- Passwords must be encrypted using bcrypt.
- All data must be transmitted over HTTPS to ensure security.

Scalability

- Initially support 10,000 users and scale dynamically based on demand.

Reliability

- Ensure 99.9% uptime with daily backups to prevent data loss.

Portability

- The app must be compatible with major web browsers (Chrome, Firefox, Safari) and mobile platforms (iOS, Android).

---

6. Business/Domain Rules

User Rules

- Each user must have a unique email address.
- Passwords must be at least 8 characters long and include alphanumeric characters.

Workout Rules

- Workout duration must be greater than 0 minutes.
- Intensity levels must be one of the following: `Low`, `Medium`, `High`.

Goal Rules

- Goals must have a target value greater than 0.
- The deadline for a goal must be a future date.

Admin Rules

- Admins can deactivate only active accounts.
- Admin actions must be logged for audit purposes.

---

7. Glossary

Term	**Definition**
User :	A registered individual who uses the app to log workouts and track progress.
Workout:	A session of physical exercise logged by the user, including details like type, duration, and intensity.
Goal:	A fitness objective set by the user, such as losing weight or improving strength.
Progress:	A collection of data points and trends that show the user's fitness journey over time.
Admin:	A user with elevated privileges to manage system and user accounts.

---

8. Logical Architecture

Layers

1. **Presentation Layer**:

   - **Functionality**:
     - Provides the user interface for web and mobile platforms.
     - Handles user input and presents visualizations like workout logs and progress charts.

- **Technologies**:
    - Web: React.js, Bootstrap.
    - Mobile: Flutter or React Native.

2. **Business Logic Layer**:
    - **Functionality**:
        - Processes user input and performs core app functionalities like workout logging and goal tracking.
        - Manages validation, business rules, and application workflows.
    - **Technologies**:
        - Spring Boot (Java).
3. **Data Layer**:
    - **Functionality**:
        - Stores and retrieves data related to users, workouts, goals, and progress.
        - Ensures data consistency and integrity.
    - **Technologies**:
        - MySQL or PostgreSQL for relational database management.
        - Hibernate for ORM.
4. **Integration Layer**:
    - **Functionality**:

        - Acts as the middleware for communication between different system components.
        - Exposes RESTful APIs for mobile and web applications.
    - **Technologies**:
        - Spring Web for API development.

Let me know if further details or diagrams are needed!