基本設計書

# 1. はじめに

## 1.1 背景

このプロジェクトの目的は、筋力トレーニングの記録と進捗を効率的に管理するためのウェブアプリケーションを提供することです。これにより、ユーザーは自分のトレーニングデータを簡単に記録・管理でき、視覚化された進捗を確認しながらトレーニングプランを最適化できます。

## 1.2 システム概要

このアプリケーションは、ユーザー管理、トレーニング記録、進捗管理、エクスポート機能を提供します。ユーザーは登録後、トレーニングの種目を追加し、セッションごとの記録を行い、進捗をグラフやチャートで確認することができます。さらに、記録をCSVやPDF形式でエクスポートすることが可能です。

# 2. 機能設計

## 2.1 ユーザー管理機能

ユーザー登録: 新規ユーザーは、名前、メールアドレス、パスワードを入力して登録します。  
ログイン/ログアウト: 登録済みのユーザーは、メールアドレスとパスワードを使用してログインし、セッション終了時にログアウトします。  
プロフィール管理: ユーザーは、自分のプロフィール情報を編集できます。

## 2.2 トレーニング記録機能

トレーニング種目の追加: ユーザーは、新しいトレーニング種目（例: ベンチプレス、スクワット）を追加できます。  
トレーニングセッションの記録: 各セッションの重量、回数、セット数、日付などを記録します。  
トレーニングの編集/削除: ユーザーは、過去のトレーニング記録を編集または削除できます。

## 2.3 進捗管理機能

進捗の可視化: 記録されたデータを基に、折れ線グラフや棒グラフを使用して進捗を表示します。  
目標設定と達成状況の管理: ユーザーは、目標を設定し、その達成状況を確認できます。

## 2.4 エクスポート機能

CSVエクスポート: 記録されたトレーニングデータをCSV形式でエクスポートします。  
PDFエクスポート: 記録と進捗をPDF形式でエクスポートします。

## 2.5 セキュリティ機能

ユーザー認証: Spring Securityを使用して、ユーザーの認証を行います。  
アクセス制御: 各ユーザーのデータへのアクセスを適切に制御します。

# 3. 非機能設計

## 3.1 パフォーマンス要件

レスポンスタイム: ユーザー操作に対するレスポンスは1秒以内を目標とします。  
同時接続数: 100人以上の同時接続をサポートします。

## 3.2 可用性要件

稼働時間: システムは99.9%以上の稼働時間を保証します。

## 3.3 拡張性要件

スケーラビリティ: AWSやHerokuを利用し、負荷に応じてサーバーをスケールさせます。

## 3.4 セキュリティ要件

データ暗号化: ユーザーのパスワードや個人情報は暗号化して保存します。  
ログイン試行回数制限: 認証失敗の試行回数に制限を設け、一定回数以上失敗するとアカウントを一時的にロックします。

# 4. UI/UX設計

## 4.1 ユーザーインターフェース設計

直感的なデザイン: ユーザーが直感的に操作できるインターフェースを提供します。  
レスポンシブデザイン: デスクトップおよびモバイルデバイスに対応するレスポンシブデザインを採用します。

## 4.2 ユーザビリティ設計

ヘルプ機能: 各画面にヘルプを提供し、ユーザーが迷わないようにします。

# 5. システム構成設計

## 5.1 フロントエンド

技術スタック: HTML/CSS, JavaScript, Thymeleafを使用します。

## 5.2 バックエンド

技術スタック: Java, Spring Boot, Spring Security, JPA/Hibernateを使用します。

## 5.3 データベース

データベース管理システム: MySQLを採用します。

## 5.4 インフラストラクチャ

インフラ: AWSまたはHerokuを使用し、GitHub Actionsを利用したCI/CDパイプラインを構築します。

# 6. 開発スケジュール

# 1. はじめに

## 1.1 背景

このプロジェクトの目的は、筋力トレーニングの記録と進捗を効率的に管理するためのウェブアプリケーションを提供することです。これにより、ユーザーは自分のトレーニングデータを簡単に記録・管理でき、視覚化された進捗を確認しながらトレーニングプランを最適化できます。

## 1.2 システム概要

このアプリケーションは、ユーザー管理、トレーニング記録、進捗管理、エクスポート機能を提供します。ユーザーは登録後、トレーニングの種目を追加し、セッションごとの記録を行い、進捗をグラフやチャートで確認することができます。さらに、記録をCSVやPDF形式でエクスポートすることが可能です。

# 2. 機能設計

## 2.1 ユーザー管理機能

ユーザー登録: 新規ユーザーは、名前、メールアドレス、パスワードを入力して登録します。

ログイン/ログアウト: 登録済みのユーザーは、メールアドレスとパスワードを使用してログインし、セッション終了時にログアウトします。

プロフィール管理: ユーザーは、自分のプロフィール情報を編集できます。

## 2.2 トレーニング記録機能

トレーニング種目の追加: ユーザーは、新しいトレーニング種目（例: ベンチプレス、スクワット）を追加できます。

トレーニングセッションの記録: 各セッションの重量、回数、セット数、日付などを記録します。

トレーニングの編集/削除: ユーザーは、過去のトレーニング記録を編集または削除できます。

## 2.3 進捗管理機能

進捗の可視化: 記録されたデータを基に、折れ線グラフや棒グラフを使用して進捗を表示します。

目標設定と達成状況の管理: ユーザーは、目標を設定し、その達成状況を確認できます。

## 2.4 エクスポート機能

CSVエクスポート: 記録されたトレーニングデータをCSV形式でエクスポートします。

PDFエクスポート: 記録と進捗をPDF形式でエクスポートします。

## 2.5 セキュリティ機能

ユーザー認証: Spring Securityを使用して、ユーザーの認証を行います。

アクセス制御: 各ユーザーのデータへのアクセスを適切に制御します。

# 3. 非機能設計

## 3.1 パフォーマンス要件

レスポンスタイム: ユーザー操作に対するレスポンスは1秒以内を目標とします。

同時接続数: 100人以上の同時接続をサポートします。

## 3.2 可用性要件

稼働時間: システムは99.9%以上の稼働時間を保証します。

## 3.3 拡張性要件

スケーラビリティ: AWSやHerokuを利用し、負荷に応じてサーバーをスケールさせます。

## 3.4 セキュリティ要件

データ暗号化: ユーザーのパスワードや個人情報は暗号化して保存します。

ログイン試行回数制限: 認証失敗の試行回数に制限を設け、一定回数以上失敗するとアカウントを一時的にロックします。

# 4. UI/UX設計

## 4.1 ユーザーインターフェース設計

直感的なデザイン: ユーザーが直感的に操作できるインターフェースを提供します。

レスポンシブデザイン: デスクトップおよびモバイルデバイスに対応するレスポンシブデザインを採用します。

## 4.2 ユーザビリティ設計

ヘルプ機能: 各画面にヘルプを提供し、ユーザーが迷わないようにします。

# 5. システム構成設計

## 5.1 フロントエンド

技術スタック: HTML/CSS, JavaScript, Thymeleafを使用します。

## 5.2 バックエンド

技術スタック: Java, Spring Boot, Spring Security, JPA/Hibernateを使用します。

## 5.3 データベース

データベース管理システム: MySQLを採用します。

## 5.4 インフラストラクチャ

インフラ: AWSまたはHerokuを使用し、GitHub Actionsを利用したCI/CDパイプラインを構築します。

# 6. 開発スケジュール

各フェーズごとに進捗をレビューし、問題があれば早期に対応します。

# 7. リスク管理

## 7.1 技術的リスク

新技術を採用する際、学習期間を確保します。

## 7.2 プロジェクトリスク

スケジュールの遅延が発生した場合は、早期に対応し、計画の見直しを行います。