1.서론 (Introduction)

- **목적 (Purpose)**: 이 소프트웨어 설계서의 목적은 AI 헬스 트레이너 서비스의 각 구성 요소를 구체적으로 설계하여 개발자들이 소프트웨어를 효율적으로 구현할 수 있도록 돕는 것입니다.
- **범위 (Scope)**: 본 설계서는 AI 헬스 트레이너 웹 애플리케이션의 개인 맞춤형 추천 제공, 달력을 통한 기록 관리, 채팅 기반 대화형 인터페이스, 그리고 회원가입 및 정보 관리 모듈을 포함한 시스템 설계를 다룹니다.
- **참고 문서 (Reference Documents)**: 소프트웨어 요구사항 명세서(SRS), 아키텍처 설계 문서, LLM 활용 AI 설계 문서.

---

2. 시스템 개요 (System Overview)

- **기능 설명 (Functional Overview)**: 본 시스템은 AI 헬스 트레이너로, 개인 맞춤형 운동 및 식단 추천, 채팅을 통한 실시간 상담, 운동 및 식단 기록 관리, 사용자 정보 입력 및 관리 기능을 제공합니다.
- **시스템 아키텍처 (System Architecture)**: 3계층 구조 (Presentation Layer, Business Logic Layer, Data Layer)
  - **Presentation Layer**: 사용자 인터페이스(UI)로, 개인 맞춤형 추천, 채팅 서비스, 운동 기록 및 분석 기능, 사용자 정보 입력을 위한 화면을 제공합니다.
  - **Business Logic Layer**: 사용자의 질문 분석 및 맞춤형 운동/식단 추천, 실시간 피드백, 기록된 운동 및 식단 데이터를 처리하는 로직을 구현합니다.
  - **Data Layer**: 사용자 정보, 운동 및 식단 기록, 시스템에서 처리된 데이터를 데이터베이스에 저장 및 불러오기를 담당하며, CRUD 작업을 수행합니다.

---

3. 모듈 설계 (Module Design)

각 모듈은 독립적으로 설계되어 쉽게 유지보수하고 확장할 수 있도록 설계됩니다.

3.1. 사용자 관리 모듈 (User Management Module)

- **기능 설명**: 사용자의 기본 정보와 신체 정보를 입력, 회원 가입 및 로그인, 사용자의 운동 목표를 관리.

- **입력 (Input)**: 사용자 이름, 이메일, 비밀번호, 신체 정보(키, 몸무게, 체지방량, 골격근량), 운동 목표.
- **출력 (Output)**: 등록 성공 여부, 로그인 토큰, 맞춤형 운동 및 식단 추천.
- **프로세스 흐름:**
  1. 사용자가 회원 가입 시 기본 정보와 신체 정보를 입력.
  2. 입력된 정보를 데이터베이스에 저장.
  3. 비밀번호는 해시 알고리즘을 통해 암호화.
  4. 로그인 시 입력한 비밀번호와 저장된 해시 비밀번호를 비교하여 인증.
- **주요 클래스 및 메서드**:
  - **UserController**
    - **registerUser(UserDto user)**: 회원가입 기능.
    - **login(String email, String password)**: 로그인 기능.
  - **UserService**
    - **validateUserCredentials()**: 사용자 자격 검증.
    - **createUser()**: 사용자 생성.
  - **UserRepository**
    - **save(UserEntity user)**: 사용자 정보 저장.
    - **findByEmail(String email)**: 이메일로 사용자 검색.

3.2. 개인 맞춤형 추천 모듈 (Personalized Recommendation Module)

- **기능 설명**: 사용자의 신체 정보 및 운동 목표를 기반으로 맞춤형 운동 및 식단을 추천.
- **입력 (Input)**: 사용자 신체 정보(키, 몸무게, 체지방량, 골격근량), 운동 목표.
- **출력 (Output)**: 맞춤형 운동 프로그램, 식단 추천.
- **프로세스 흐름**:
  1. 사용자가 신체 정보와 운동 목표를 입력.
  2. 입력된 정보가 데이터베이스에 저장.
  3. LLM을 활용해 사용자의 정보를 분석하여 맞춤형 운동 및 식단 추천 제공.

- **주요 클래스 및 메서드**:
  - **RecommendationController**:
    - **getRecommendations(UserDto user)**: 맞춤형 추천 제공.
  - **RecommendationService**:
    - **generateExercisePlan(UserDto user)**: 운동 계획 생성.
    - **generateDietPlan(UserDto user)**: 식단 계획 생성.
  - **RecommendationRepository**:
    - **saveRecommendations(RecommendationEntity recommendation)**: 추천 데이터 저장.

3.3. 운동 기록 및 분석 모듈 (Exercise Tracking & Analysis Module)

- **기능 설명**: 사용자가 수행한 운동 기록을 저장, 기록된 데이터를 분석하여 사용자가 쉽게 확인할 수 있도록 제공.
- **입력 (Input)**: 사용자의 운동 기록(운동 시간, 운동 종류), 식단 기록.
- **출력 (Output)**: 기록된 운동 및 식단 정보.
- **프로세스 흐름**
  1. 사용자가 운동 및 식단 정보를 입력.
  2. 입력된 정보가 데이터베이스에 저장.
  3. 사용자가 날짜를 클릭하면 해당 날짜의 운동 및 식단 정보를 불러와서 출력.
- **주요 클래스 및 메서드**
  - **ExerciseController**
    - **addExerciseRecord(ExerciseDto exercise)**: 운동 기록 저장.
    - **getExerciseRecordsByDate(String date)**: 특정 날짜의 운동 기록 조회.
  - **ExerciseService**
    - **analyzeExerciseData(UserDto user)**: 사용자 운동 데이터 분석.
  - **ExerciseRepository**
    - **save(ExerciseEntity exercise)**: 운동 기록 저장.

- ◆ **findByDate(String date)**: 날짜별 운동 기록 조회.

3.4. 챗봇 서비스 모듈 (Chatbot Service Module)

- **기능 설명**: 사용자가 운동 및 식단에 관련된 질문을 채팅으로 입력, AI가 해당 질문을 분석하여 맞춤형 답변을 제공.
  - **입력 (Input)**: 사용자의 질문(텍스트 입력).
  - **출력 (Output)**: 질문에 대한 답변(운동 및 식단 정보).
- **프로세스 흐름**:
  1. 사용자가 챗봇에 질문을 입력.
  2. 질문이 서버로 전달되며, LLM을 통해 자연어 처리 후 답변 생성.
  3. 생성된 답변을 사용자에게 실시간으로 제공.
- **주요 클래스 및 메서드**:
  - **ChatController**:
    - ◆ **processUserQuery(String query)**: 사용자 질문 처리.
  - **ChatService**:
    - ◆ **generateResponse(String query)**: LLM을 이용해 응답 생성.
  - **ChatRepository**:
    - ◆ **logChatSession(ChatEntity chat)**: 채팅 세션 기록 저장.

3.5. 데이터베이스 모듈 (Database Module)

- **기능 설명**: 사용자의 신체 정보, 운동 기록, 맞춤형 추천 데이터를 저장하고 관리합니다.
  - **입력 (Input)**: 사용자 정보, 운동 기록, 추천 데이터.
  - **출력 (Output)**: 요청된 데이터.
- **프로세스 흐름**:
  1. 사용자 정보, 운동 기록, 맞춤형 추천 데이터를 저장.
  2. 필요 시 해당 데이터를 요청 및 제공.
- **주요 클래스 및 메서드**:
  - **DatabaseConnection**: 데이터베이스 연결 관리.

- **UserRepository**: 사용자 정보 저장 및 조회.
- **ExerciseRepository**: 운동 기록 저장 및 조회.
- **RecommendationRepository**: 추천 데이터 저장 및 조회.

---

4. 데이터베이스 설계

- **사용자 정보 테이블 (User Info Table)**

1. ID INT PRIMARY KEY AUTO_INCREMENT (사용자번호)
2. EMAIL VARCHAR(255) UNIQUE NOT NULL (이메일 아이디)
3. PASSWORD VARCHAR(255) NOT NULL (비밀번호)
4. NAME VARCHAR(100) UNIQUE NOT NULL (사용자이름)
5. CREATED_AT TIMESTAMP DEFAULT CURRENT_TIMESTAMP (가입일)

- **4.2 신체 정보 테이블 (Body Table)**

1. BODY_ID INT PRIMARY KEY AUTO_INCREMENT
2. USER_ID INT
3. HEIGHT DECIMAL(4,1) NOT NULL (키)
4. WEIGHT DECIMAL(4,1) NOT NULL (몸무게)
5. FAT DECIMAL(4,1) NOT NULL (체지방량)
6. MUSCLE DECIMAL(4,1) NOT NULL (골격근량)
7. FOREIGN KEY (USER_ID) REFERENCES Basic(ID)

- **4.3 운동 목표 테이블 (Exercise Table)**

1. EXERCISE_ID INT PRIMARY KEY AUTO_INCREMENT
2. USER_ID INT
3. TARGRT_WEIGHT DECIMAL(4,1) NOT NULL (목표 몸무게)
4. GOAL VARCHAR(255) NOT NULL (운동 목표)
5. INTENSITY VARCHAR(50) NOT NULL (운동 강도)
6. FOREIGN KEY (USER_ID) REFERENCES Basic(ID)

- **4.4 운동 기록 테이블 (Workout Table)**

1. WORKOUT_ID INT PRIMARY KEY AUTO_INCREMENT
2. USER_ID INT

3. WORKOUT_DATE DATE NOT NULL (날짜)
4. EXERCISE_TYPE VARCHAR(100) NOT NULL (운동 종류)
5. DURATION VARCHAR(100) NOT NULL (운동 시간)
6. FOREIGN KEY (USER_ID) REFERENCES Basic(ID)

- **4.5 식단 기록 테이블 (Diet Table)**

1. DIET_ID PRIMARY KEY AUTO_INCREMENT
2. USER_ID INT
3. DIET_DATE DATE NOT NULL (날짜)
4. MEAL_TYPE VARCHAR(100) NOT NULL (식단 종류)
5. CALORIES INT NOT NULL (칼로리)
6. FOREIGN KEY (USER_ID) REFERENCES Basic(ID)

---

5. 사용자 인터페이스 설계

5.1 화면 구조도

- FITGPT 앱은 메인 화면, 운동 및 식단 기록 화면, 추천 화면, 질문 응답 화면 등으로 구성.

5.2 주요 화면 설계

- **메인 화면**: 운동과 식단 추천 정보를 일목요연하게 제공하며, 사용자가 각 기능에 쉽게 접근할 수 있도록 구성.
- **운동 및 식단 기록 화면**: 사용자가 일별로 운동 및 식단 기록을 입력, 조회, 수정할 수 있도록 설계. 캘린더 형식으로 전체 기록 조회 가능.
- **추천 화면**: 사용자의 신체 정보와 목표에 맞춘 맞춤형 운동 및 식단 정보를 제공하는 화면.
- **질문 응답 화면**: 사용자가 질문을 입력하면 LLM 이 답변을 제공하는 인터페이스.

5.3 사용자 시나리오

- **시나리오 1**: 로그인 후 사용자는 개인정보 및 신체 정보를 입력
- **시나리오 2**: 사용자가 운동 목표에 맞춘 추천 운동과 식단 정보를 확인.
- **시나리오 3**: 사용자가 당일 운동 및 식단 기록을 입력하고 조회.
- **시나리오 4**: 운동 또는 식단 관련 질문을 입력하여 LLM 기반 답변을 확인.

---

6. 보안 설계 (Security Design)

6.1 인증 및 권한 관리

- **JWT 인증**: JSON Web Token(JWT)을 사용하여 사용자가 로그인하면 토큰을 발급하며, 사용자가 애플리케이션을 이용할 때마다 이 토큰을 통해 사용자 인증을 수행. 토큰은 유효 기간을 설정하여 만료 시 자동 로그아웃 되도록 함.

6.2 데이터 암호화

- **비밀번호 해시화**: 사용자의 비밀번호는 해시 알고리즘(Bcrypt)을 사용해 해시화하여 데이터베이스에 저장. 이를 통해 데이터 유출 시에도 비밀번호를 복원할 수 없도록 보호.
- **접속 로그 기록**: 사용자의 접속 및 활동 기록을 남겨 비정상적인 접근이나 로그인 시도를 모니터링할 수 있도록 함. IP 주소, 접근 시간, 접근 경로 등을 기록.

---

7. 성능 및 확장성 고려사항

7.1 성능 요구사항

- 서버 응답 속도: 모든 API 응답은 1 초 이내로 반환되어야 함.
- 앱 로딩 속도: 주요 화면 로딩 시간은 2 초 이내로 유지.

7.2 확장성 방안

- **데이터베이스**: 증가하는 사용자 데이터를 효율적으로 관리하기 위해 데이터를 여러 개의 작은 저장소로 나누어 저장하는 방식을 고려.
- **서버 확장**: 트래픽이 많아질 경우 여러 서버에 작업을 나누어 처리하는 부하 분산 장치를 사용하는 방식을 고려.

---

8. 테스트 계획

8.1 단위 테스트 계획

- API 별 정상 및 예외 상황 테스트
- 입력 데이터 검증 로직 테스트

8.2 통합 테스트 계획

- API 서버와 데이터베이스의 연동 테스트
- **Postman 을 사용하여** 클라이언트와 서버 간 데이터 교환 시나리오 테스트

8.3 시스템 테스트 계획

- 실제 사용자 시나리오 기반의 종합 테스트

---

9. 부록

9.1 변경 이력

- 시스템 설계 및 API 기능 추가/변경 사항 기록

9.2 검토 의견

- 팀원 및 검토자의 피드백과 수정 내역 기록