**FitiTracker**

**Project Overview**

FitiTracker is a web-based fitness tracking application that allows users to log and view their workout routines, track progress, and manage exercises efficiently. The project integrates Django as the backend framework and Bootstrap for front-end styling. Users can add exercises to their workout logs, specify details like sets, reps, and weight, and retrieve their workout history.

**Distinctiveness and Complexity**

**Distinctiveness**

FitiTracker is unique because it combines dynamic form management with Django's robust backend capabilities. Unlike many static fitness tracking tools, FitiTracker provides:

* **Dynamic formsets**: Users can add and remove exercises dynamically on the same page without refreshing.
* **Custom CSS styling with Bootstrap integration**: Combines a sleek, modern UI with tailored design elements.
* **Relational Data Models**: Utilizes normalized models for workouts and exercises, enabling scalable data handling.

The name ‘FitiTracker is the first of its kind in the Kenyan fitness industry. In Swahili, ‘fiti’ means to be in good health or ‘cool’ in slang. Combining this with the fitness tracking purpose of the app created the name ‘FitiTracker’.

**Complexity**

* **Dynamic Forms**: The formset dynamically adjusts the number of forms and updates the total form count using JavaScript, seamlessly integrating with Django's inline formset management.
* **Client-Server Interaction**: Proper CSRF token integration ensures secure data submission while maintaining a responsive UI.
* **Integration Challenges**: Combining Bootstrap's predefined styles with custom CSS required balancing aesthetic cohesion and individuality.
* **Modular Design**: Separates the logic into Django apps and ensures scalability.

**Design Approach**

**Backend Design**

* **Django Models**:
  + WorkoutDate: Tracks unique workout dates.
  + Workout: Links to WorkoutDate and stores workout names.
  + Exercise: Tracks specific exercises tied to workouts.
* **Forms and Formsets**:
  + Custom Django forms manage user input with robust validation.
  + Inline formsets allow easy linkage of exercises to workouts.

**Frontend Design**

* **Bootstrap Integration**: Provides a responsive and visually appealing layout.
* **Custom CSS**: Allows for unique branding while preserving Bootstrap's responsiveness.
* **JavaScript**: Handles dynamic form management, ensuring smooth user interaction.

**File Structure and Content**

**Backend Files**

1. **models.py**:
   * Defines the data scheme for workouts and exercises.
2. **forms.py**:
   * Contains forms for managing workout and exercise input.
3. **views.py**:
   * Handles rendering templates and processing form submissions.
4. **urls.py**:
   * Maps URLs to views for navigation.

**Frontend Files**

1. **Templates**:
   * home.html: Landing page for the application.
   * log.html: Page for logging workouts.
   * view.html: Displays logged workout history.
2. **Static Files**:
   * style.css: Custom CSS for branding and design.
   * script.js: JavaScript for dynamic form functionality and mobile responsiveness.

**Configuration Files**

* **settings.py**: Configures Django settings, including static file paths and database configurations.
* **urls.py**: Central URL dispatcher for the Django project.

**How to Run the Application**

1. **Clone the Repository**:

git clone <repository-url>

cd fitapp

1. **Set Up Virtual Environment**:

python -m venv myvenv  
myvenv\Scripts\activate #for WIndows

source venv/bin/activate #for MacOS

1. **Install Dependencies (if not already installed)**: pip install -r requirements.txt
2. **Apply Migrations**:

python manage.py makemigrations

python manage.py migrate

1. **Run the Development Server**: python manage.py runserver
2. **Access the Application**: Open your browser and navigate to <http://127.0.0.1:8000>.
3. **Use the application**: Log a completed workout or view logged workouts.

**Additional Information**

* **Dynamic Form Functionality**: Users can add multiple exercises per workout without reloading the page. The JavaScript handles dynamic form generation and updates the Django formset's management form.
* **CSRF Protection**: All forms include CSRF tokens for secure data handling.
* **Extensibility**: The modular design allows for easy addition of new features, such as user authentication or analytics.