# Fitness Tracker Application — User Guide

Course: CSCI 487 – Software Design & Development

Instructor: Mr. Benjamin Standfield

Technical Writer & Developer: Jordan Broomfield

Semester: Fall 2025

## 1. Introduction

## The Fitness Tracker Application is a health-monitoring system that allows users to track calorie intake, record daily meals, and receive personalized exercise recommendations. The system is built using ASP.NET Core 8.0 (C#) and exposes a set of RESTful API endpoints that can be tested through Swagger UI or Postman.

## This guide walks end-users through how to set up the system, input personal details, and interact with the API to manage fitness data effectively.

## 2. Purpose of the Application

The primary goal is to help users:

* Maintain awareness of their daily nutritional intake.
* Receive exercise suggestions based on their profile.
* View previous calorie logs and monitor progress over time.
* Access a simple web-based interface for interaction and testing (Swagger UI).

## 3. System Requirements

|  |  |
| --- | --- |
| Component | Minimum Requirement |
| Operating System | Windows 10 or 11 (64-bit) / macOS 13+ / Linux (Ubuntu 22+) |
| .NET SDK | Version 8.0 or later |
| IDE (Optional) | Visual Studio 2022 or Visual Studio Code |
| Memory & Storage | ≥ 4 GB RAM and 1 GB disk space |
| Additional Tools | Git (for cloning), Docker (for containerized runs) |
| Testing Tools | Swagger UI / Postman |

## 4. Account Setup and Login

## Launch the API

## Run dotnet run --project src/FitnessTracker.Api/FitnessTracker.Api.csproj or start via Docker.

## Navigate to http://localhost:8080/swagger.

## Register a User

## Go to the POST /api/auth/register endpoint.

## Enter details such as:

## Username

## Password

## Email (optional)

## Click Execute to register.

## Login to the System

## Use POST /api/auth/login.

## Enter your username and password.

## The API returns a JWT token — copy it.

## Authenticate Requests

## In Swagger’s “Authorize” button, paste Bearer <token> so you remain logged in for other requests.

## 5. Adding Calorie Entries

1. Go to **POST /api/calories/log**.
2. Input the following JSON body:

{

“mealName": "Grilled Chicken Sandwich",

"calories": 420,

"date": "2025-10-15"}

1. Click **Execute**.
2. The system will confirm the new entry with a success message and unique entry ID.

## 6. Viewing Calorie History

1. Use **GET /api/calories/history**.
2. The response displays:
   * Meal names
   * Calorie values
   * Dates of entries
3. This helps users monitor daily or weekly progress.

## 7. Exercise Recommendations

* Open **GET /api/exercises/suggest**.
* The system generates personalized suggestions (e.g., jogging, HIIT, strength training).
* Recommendations are calculated based on total calories logged.

## 8. Health Check

## To verify that the application is running correctly, test:

## GET /health Expected response:

## {“status": "Healthy" }

## 9. Using Docker

## Build the Docker image:

## docker build -t fitness-tracker-api .

## 2. Run the container:

## docker run -p 8080:8080 fitness-tracker-api

## 3. Open the API in your browser:

## http://localhost:8080/swagger

## 10. Troubleshooting

|  |  |  |
| --- | --- | --- |
| **Issue** | **Cause** | **Fix** |
| 401 Unauthorized | Missing or expired token | Re-login and add Bearer token in Swagger. |
| 400 Bad Request | Missing field in JSON body | Double-check the required fields. |
| Port already in use | Another service on 8080 | Change the port in launchSettings.json. |
| JWT Validation Failed | Wrong secret key or expired token | Verify appsettings.json and token. |

## 11. Logout and Session

Developers can verify system functionality with the command:

dotnet test

All tests are stored in tests/FitnessTracker.Tests/Test.cs.  
Expected result: *“Test Run Successful”*

Run tests using: dotnet test  
All tests are in tests/FitnessTracker.Tests/Test.cs.

## 13. Maintainers

|  |  |
| --- | --- |
| Role | Name |
| Developer / Technical Writer | Jordan Broomfield |
| Instructor | Mr. Benjamin Standfield |
| Course | CSCI 487 – Software Design & Development |
| Institution | Virginia State University |

## 14. Version History

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Description** |
| V1.0 | 10/16/25 | Initial release with authentication, Docker Support, API Documentation. |

## 15. Conclusion

The Fitness Tracker Application combines modern API design, authentication, and clean architecture to deliver a reliable fitness-tracking backend.  
Users can easily track calories, review data, and test endpoints through Swagger UI.  
This project highlights both practical implementation and clear documentation, meeting CSCI 487 software-development standards.