Software Requirements Specification

for

Health App Web

Website and API

Version 0.1
Prepared by
Enrique Alonso Esposito

San Marcos
June 2024

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to define the requirements for the Health Web App, which includes both the website and the API components. This document will provide a detailed description of the functionalities, features, and constraints of the system to ensure that it meets the needs and expectations of its users.

1.2 Scope

The Health Web App is designed to provide users with a comprehensive platform to track their health and fitness data. It includes features such as meal tracking, body weight progression, and daily macro breakdowns. The API will support the web app by providing the necessary backend services.

1.3 Definitions, Acronyms, and Abbreviations

- API: Application Programming Interface
- UI: User Interface
- JWT: JSON Web Token
- SRS: Software Requirements Specification
- SDLC: Software Development Life Cycle

1.4 References

- Angular Documentation
- Python
- JWT Authentication Guide

1.5 Overview

This document includes:

- Overall description of the system
- Functional and non-functional requirements
- System features
- External interface requirements

2. Overall Description

2.1 Product Perspective

The Health Web App is a standalone system that includes a web-based user interface and a backend API. The frontend will be developed using Angular, while the backend API will be developed using Python and Flask

2.2 Product Functions

- User registration and authentication
- Meal tracking
- Body weight progression tracking
- Daily macros breakdown
- Data visualization through charts and graphs
- PDF export of health data

2.3 User Classes and Characteristics

- **End Users:** Individuals who want to track their health and fitness data.
- **Admin Users:** Individuals who manage the system and provide support to end users.

2.4 Operating Environment

- Frontend: Web browsers (Chrome, Firefox, Safari, Edge)
- Backend: Python and Flask API and SQLite for data storage

2.5 Design and Implementation Constraints

- JWT for secure authentication
- Responsive design for mobile and desktop use

2.6 User Documentation

- User manual
- Admin guide
- API documentation

2.7 Assumptions and Dependencies

- Users have access to the internet
- Users have modern web browsers

3. System Features

3.1 User Registration and Authentication

- **Description:** Users can register and log in to the system.
- **Priority:** High
- **Stimulus/Response Sequences:**
    1. User enters registration details.
    2. System validates and stores user data.
    3. User logs in using credentials.
    4. System generates and returns JWT.

3.2 Meal Tracking

- **Description:** Users can log their daily meals.
- **Priority:** High
- **Stimulus/Response Sequences:**
  1. User enters meal details.
  2. System stores meal data.
  3. System updates daily macros breakdown.

3.3 Body Weight Progression Tracking

- **Description:** Users can track their body weight over time.
- **Priority:** Medium
- **Stimulus/Response Sequences:**
  1. Users are prompted to enter their body weight once every day.
  2. User enters weight.
  3. System stores weight data.
  4. System generates weight progression chart.

3.4 Daily Macros Breakdown

- **Description:** Users can view their daily macro nutrients.
- **Priority:** High
- **Stimulus/Response Sequences:**
  1. System calculates daily macros based on meal data.
  2. User views macro breakdown on dashboard.

3.5 Data Visualization

- **Description:** Users can view charts and graphs of their health data.
- **Priority:** Medium
- **Stimulus/Response Sequences:**
  1. System generates charts based on user data.
  2. User views charts on dashboard.

3.6 PDF Export

- **Description:** Users can export their health data as a PDF.
- **Priority:** Low
- **Stimulus/Response Sequences:**
  1. User clicks 'Save as PDF'.
  2. System generates and downloads PDF.

4. External Interface Requirements

4.1 User Interfaces

- **Login Page:** Allows users to log in.
- **Dashboard:** Displays user health data.
- **Meal Entry Form:** Allows users to log meals.
- **Weight Entry Form:** Allows users to log weight.
- **Charts and Graphs:** Visual representation of user data.
- **Workout creation Form**: Allows users to create workouts and add exercises to them.
- **Exercise creation form:** Allows users to search popular exercises or create their own.
- **Schedule maker**: Allows users to drag and drop their workouts into their weekly view
- **PDF export:** Users can turn their workouts into PDF to save to their devices or print.

4.2 Hardware Interfaces

- Not applicable

4.3 Software Interfaces

- **Frontend to Backend:** REST API calls
- **Backend to Database:** SQL queries

4.4 Communications Interfaces

- **Internet:** HTTP/HTTPS for API communication

5. Non-Functional Requirements.

5.3 Software Quality Attributes

- **Usability:** The system should be easy to use and navigate.
- **Maintainability:** The code should be modular and well-documented.

5.4 Business Rules

- Users can only view and modify their own data.