**Project Design Document**

**Project Name:** HealthTracker

**TABLE OF CONTENTS**

1. **System Overview**
   1. **Brief Project Description**
   2. **System Architecture**
   3. **Technology Stack**
2. **Implementation Details**
   1. **Codebase Structure**
   2. **Key Implementations**
   3. **Component Interfaces**
3. **Use Case Support in Design**
   1. **Use Case Selection**
   2. **Requirement Mapping**
   3. **Use Case Design**
   4. **Demo Requirement**
4. **Design Decisions**
   1. **Technology Comparisons**
   2. **Decision Justifications**
5. **GitHub Commit Requirement**
   1. **Code Implementations & Interfaces**
   2. **Technology Comparisons**

**LIST OF CONTRIBUTORS**

* **Kerem ELMA**
* **Kaan ÖNEN**
* **Mehmet ESKİ**

1. **System Overview**
   1. **Brief Project Description**

HealthTracker is a comprehensive health tracking application designed to help users manage their daily health habits. It aims to centralize the tracking of important health activities, such as medication schedules, doctor appointments, fitness goals, and hydration levels, in one platform. The application is developed to improve user wellness by ensuring they never miss a crucial health task and promoting a proactive approach to maintaining overall health.

* 1. **System Architecture**

Since we are a development team of 4 people and are developing a small-scale project, it would be appropriate to use a monolithic structure for the project's general system architecture.

* 1. **Technology Stack**

The project we are developing will run on two different platforms. For the part that will run on the Android platform, blablabla. For the part that will run on the web platform, Python, HTML, and CSS will be used. For the backend, the Python framework Django will be used, and for the frontend, the JavaScript framework React.js will be used.

1. **Implementation Details**
   1. **Codebase Structure**

Since the project's backend will be built on Django; router modules, form, model, and view modules that will interact with the pages will be located under the `backend/` path. Once the backend is completed, frontend development will begin under the `frontend/` path.

* 1. **Key Implementations**

Key implementations to be made:

* User login/register system
* Graphically represented statistics on the frontend
* Trackers such as water and appointments
  1. **Component Interfaces**

Since there will be no database in the project, components will be needed on the backend side to process form data and return it in its final format. Similarly, communication between the backend and frontend will be established through these JSON files. Appropriate interfaces will also be included.

1. **Use Case Support in Design**
   1. **Use Case Selection**
   2. **Requirement Mapping**
   3. **Use Case Design**
   4. **Demo Requirement**
2. **Design Decisions**
   1. **Technology Comparisons**
   2. **Decision Justifications**
3. **GitHub Commit Requirement**
   1. **Code Implementations & Interfaces**
   2. **Technology Comparisons**