Software Requirements Specification

for

Embedded Systems Mini-project Hiking application

Group 7

Prepared by C. Sulighetean (1011195), G. Pelesz (1011399), O. Lagerroos (655879), S. Nordström (653101)

Aalto University

24.03.2022

Table of Contents

Revision History

Name	**Date**	**Reason For Changes**	**Version**
First draft published	04.03.2022		0.1
First complete version published	24.03.2022	Corrected first draft based on feedback	1.0

1. Introduction

1.1 Purpose

The purpose of this document is to describe the application of Hiking Tour Assistant (HTA) which is run via the smartwatch LiLyGo TWatch 2020. The system of HTA records and displays hiking trip statistics, and thus creates a platform for any hiking activities. The final version of this document can be beneficial at low-level design and implementation of the software.

This document describes the whole system required for HTA. The main components of HTA are forementioned smartwatch to record data, Raspberry Pi 3B+ (Hub) to store sessions and transfer data, Unicorn Hat HD Matrix LED to display data, and locally run website to display and remove session data. The document structure follows the standard given in IEEE Std 830-1998.

1.2 Scope

The software product is the Hiking Tour Assistant application. This application provides the feature of recording trip statistics, such as traveled distance and step count. This recorded data can be stored to a local database which can display the statistics, accessible and readable only for the user. The application offers the user the possibility to analyze the collected results and stay motivated while continuing the hiking.

The smartwatch should record step count and GPS data, and based on that, show the current step count and the travel distance. This data can be synchronized with the Hub (via Bluetooth) which can also calculate the estimate of the burnt calories, and display the data on the LED Matrix and on the website (accessible in LAN).

The system devices and data can be accessed only by the user (or by the maintainer if software development or system updates are required) which guarantees the safety of the system and the data. The user interface of each platform is developed in such a way that it is easy to use and understand. The system of HTA is recommended to use only for low-speed outdoor activities such as hiking, jogging or playing golf.

1.3 Definitions, acronyms, and abbreviations

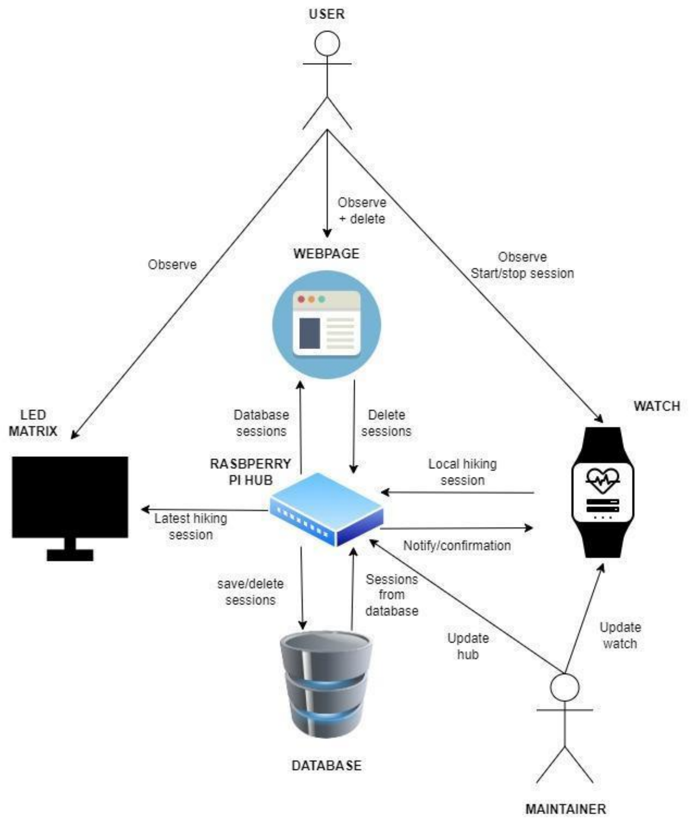
HTA	Hiking Tour Assistant
Hub	Raspberry Pi 3B+
API	Application Programming Interface
REST API	Representational State Transfer (Framework for web address)
SQL API	Structured Query Language (language for database)
WATCH API	TTGO Watch Library
GPS	Global Positioning System
LAN	Local Area Network
Bluetooth	Communication protocol
LED Matrix	The display of the system, Unicorn Hat HD Matrix LED
SPI	Serial Peripheral Interface

SSH	Secure Shell (Security protocol for maintainer to connect to Hub)
UART	Universal Asynchronous Receiver/Transmitter (computer hardware device for asynchronous serial communication
Ethernet	Data communication through a cable
RAM	Random access memory (Short term computer memory)
SRAM	Static RAM (Part of ram, read write memory)
PSRAM	A combinational form of a dynamic RAM

1.4 References

“IEEE Recommended Practice for Software Requirements Specifications," in IEEE Std 830-1998 , vol., no., pp.1-40, 20 Oct. 1998, doi: 10.1109/IEEESTD.1998.88286

2. Overall Description



2.1 Product Perspective

This product represents a simpler and affordable smartwatch for people looking to monitor their hiking sessions. There are other companies who offer a higher number of features and higher accuracy, but at a different cost. Some examples are Polar, Garmin, Suunto, Apple and Huawei which offer their smartwatches at a price range between €100 and, €1000. The product is self-contained, as each part of the system is defined in this document and the system works as a whole.

2.1.1 System interfaces

2.1.1.1 Users - are utilizing the smartwatch screen and button to record and display the hiking session and the Hub to synchronize and display the sessions.
2.1.1.2 Maintainers - are utilizing the Hub to update the system and solve software issues, can update the watch through the Hub if wired connection to the watch is established and the maintainer has access to the Hub.

2.1.2 User interfaces

2.1.2.1 Smartwatch: Users shall use the smartwatch to record the hiking sessions. The smartwatch shall have a screen and button to be able to support the visualization, and the hiking recording requirements.
2.1.2.2 LED Display: The LED Display shows for the user the last hiking session statistics, after the synchronization process of the Hub with the smartwatch.
2.1.2.3 Web page: Users are able to observe, select and delete previous hiking sessions through the web page.

2.1.3 Hardware interfaces

2.1.3.1 Major Hardware components
2.1.3.2 Wired Interfaces
2.1.3.2.1 SPI interface between the LED display and the Hub
2.1.3.2.2 Interface between the smartwatch and the GPS transceiver
2.1.3.3 Connection between Hub and LAN
2.1.3.4 Bluetooth interface between Hub and smartwatch

2.1.4 Software interfaces

2.1.4.1 REST API: through which the Hub's database can be accessed to read and delete entries. The web page is using this API
2.1.4.2 SQL API: to connect the Hub with the database in which the hiking sessions will be stored. All the system components need to access the database through this API separately.
2.1.4.3 WATCH API: of the chosen smartwatch to control the functionalities of the smartwatch

2.1.5 Communication interfaces

2.1.5.1 Bluetooth - between smartwatch and Hub to synchronize hiking session data. The Hub and the Watch shall implement a simple data exchange protocol, and a way to communicate the automatic connection and processed data statuses.
2.1.5.2 SSH (over TCP/IP) to access the Hub over the LAN network so that the maintainer can troubleshoot problems of the Hub
2.1.5.3 UART - between maintainer and smartwatch to program the smartwatch and between GPS transceiver and smartwatch to receive coordinates while the session is ongoing
2.1.5.4 SPI - between LED matrix and Hub to display information on the LED Display
2.1.5.5 Ethernet / HTTP - between the Hub and the local network, and between the Hub and a Browser on the local area network to be able to serve the website.

2.1.6 Memory constraints

2.1.6.1 The smartwatch should have at least 16 MB of flash memory, 520 kilobytes of SRAM memory, and 8 megabytes of PSRAM memory
2.1.6.2 The Hub should have at least 1 gigabyte of RAM and 32 gigabytes of Flash memory

2.1.7 Operations

2.1.7.1 Turn on smartwatch

To turn on the smartwatch, the user shall verify the inside battery of the smartwatch to be charged over 20%. The correct behavior shall not be expected under this percentage.

2.1.7.2 Start session

To start a hiking session, the user shall press on the button situated on the side panel of the smartwatch while the screen shows the start session screen

2.1.7.3 Stop session

To stop a hiking session, the user shall press on the button situated on the side panel of the smartwatch while the smartwatch shows the session statistics screen

2.2 Product Functions

2.2.1 Watch: Start & Stop session

The watch can start and stop the recording of a session by pushing the button on the watch's lateral side.

2.2.2 Watch: Record session (store, process)

The watch shall record and store an ongoing session by continuously accumulating km and step count data along with position coordinates. After stopping the recording of the session, the watch shall store the accumulated km, step count and position data of the session, including the session's ID. On starting a new session, the latest recordings shall be overwritten.

2.2.3 Watch: Show ongoing session data

The watch shall show the latest data (km, step count) during an ongoing session on the watch's screen.

2.2.4 Watch: Send data to the Hub

The watch shall send out the stored session data (session_id, km, step count, coordinates) upon synchronizing with the Hub. During an established connection with the Hub, the watch shall send out its locally stored session's data, if there is any session stored locally. After successfully sending the locally stored session to the Hub, the watch shall delete the locally stored session. The watch shall not send out data when there is an ongoing session.

2.2.5 Hub: Synchronize with watch

The Hub shall continuously try to synchronize with the watch until a connection has been made. In order for the connection to be established, the user shall stay close to the device. Upon an established connection, the Hub should wait for incoming session data.

2.2.6 Hub: Calculate burnt calories

Upon receiving a session from the watch, the Hub shall calculate the burnt calories. The calculation of the calories should be based on the formula found at the following link:

- https://www.verywellfit.com/pedometer-steps-to-calories-converter-3882595

From this article, we are assuming that each step corresponds to around 0.04 calories. From the website below we know that hiking approximately burns 6 times as much as simple walking. So the formula that the system uses should be 0.04*6=0.24kcal/step.

- https://www.omnicalculator.com/sports/calories-burned

2.2.7 Hub: Store sessions

When an established connection has been made with the watch and the Hub is receiving data from the watch, the Hub shall store the incoming session including the calculated burnt calories that was calculated upon receiving the session.

2.2.8 Hub: Host website to manage sessions

The Hub shall host a website on LAN. The website shall show all the stored sessions. The website shall provide a delete button for each session to delete them from the Hub's database.

2.2.9 Hub: Delete session

Upon a request to delete a session, the Hub should delete the session from its database.

2.2.10 Hub: Display the latest session (km, step count, calories)

The Hub shall display the latest session's data (km, step count, calories) on the connected LED display.

2.3 User Characteristics

2.3.1 Basic intuition with using digital devices.
2.3.2 The user shall be able to read and follow basic instructions.
2.3.3 The user shall not be visually impaired

2.4 Constraints

2.4.1 Project constraints

The fundamental project constraint is the pre-defined hardware (resource constraint) to make the system work with:

Project component	Hardware
Watch / Data recorder	Smart Wrist Watch LiLyGo v3 (ESP32)
Hub	Raspberry Pi 3B+
Display	Unicorn Hat HD Matrix LED

The selection of the hardware naturally affects the frameworks and software environments which can be used to implement the designed system. In this case, the project constraints create the restrictions to the system and its implementation as discussed in the section 2.4.2.

2.4.2 System constraints

The system should be implemented respecting the given hardware to enable communication, data transfer and functionalities between the system components. The table below shows how the hardware limits the usage of the possible tools available.

Hardware	Possible tools / compatibility
Smart Wrist Watch LiLyGo v3 (ESP32)	PlatformIO Libraries: TTGO Watch Library ESP32 board API
Raspberry Pi 3B+	Raspbian OS Python (v.3.7 or 3.9.10+) Libraries: BluePy, SQLite3
Unicorn Hat HD Matrix LED	40-pin header Raspberry Pi models Python 2.6 or 2.7 or 3.0+

Additionally, the communication between the watch and the Hub must happen either via Bluetooth 4.2. The Hub shall communicate with the local network via Ethernet and with the LED over SPI.

2.5 Assumptions and dependencies

2.5.1 The Hub will be a Raspberry Pi 3B+
2.5.2 The smartwatch will be a Smart Wrist Watch LiLyGo V2 2020

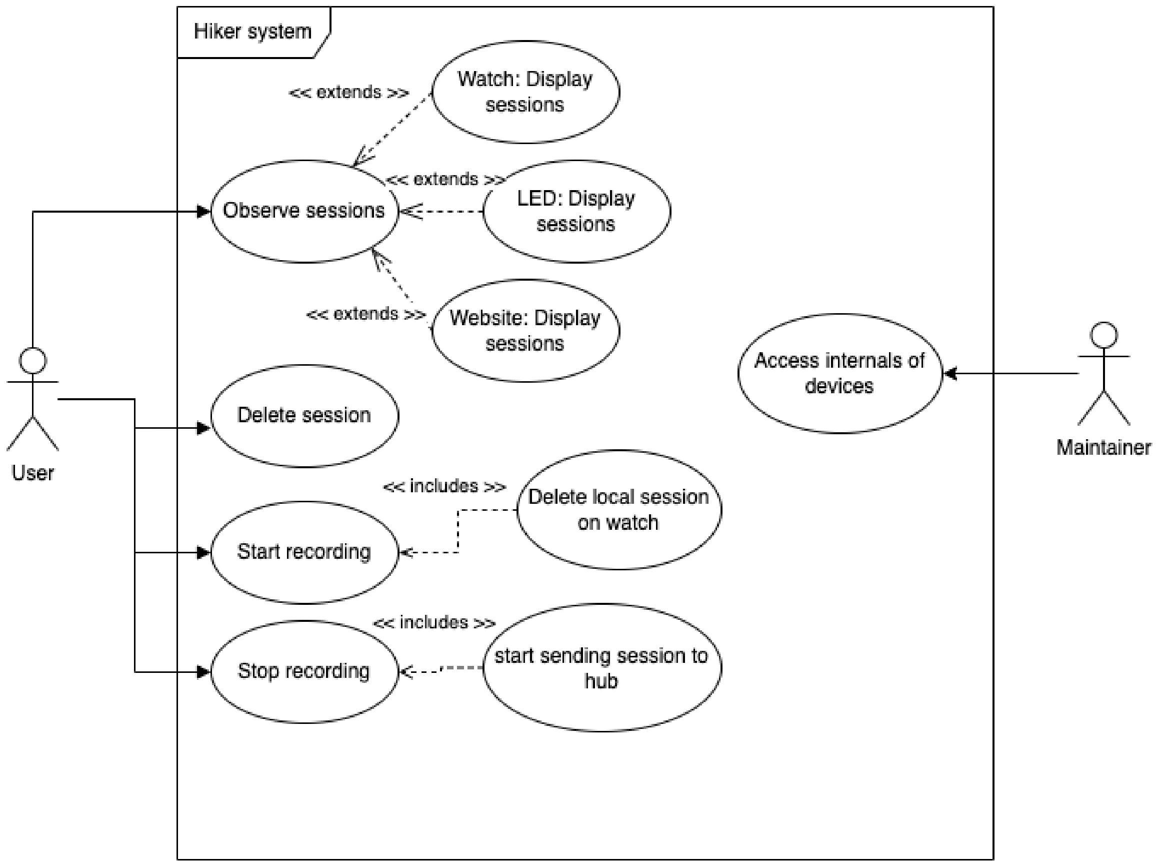
2.6 Apportioning of requirements

2.6.1 Site adaptation requirements: later stages of the development will plausibly reveal several data or initialization sequences in different scenarios.
2.6.2 User characteristics: we will have a better understanding of the user, after the prototype is ready.

3. Specific requirements

3.1 External Interfaces

3.1.1 User Interfaces



3.1.1.1 Smartwatch screen – user: customer, maintainer
3.1.1.2 Smartwatch button – user: customer
3.1.1.3 LED Display – user: customer
3.1.1.4 Website – user: customer

3.1.2 Hardware interfaces

3.1.2.1 SPI interface between LED and Hub

Name of item	Serial Peripheral Interface
Description of purpose	Transfer information from Hub to LED
Source of input / destination of output	Hub GPIO
Valid range, accuracy, and/or tolerance	-
Units of measure	Kbps
Timing	25 ms
Relationships to other inputs/outputs	-
Screen formats/organization	-
Window formats/organization	-
Data formats	-
Command formats	-
End messages	-

3.1.2.2 Interface between smartwatch and GPS transceiver

Name of item	UART
Description of purpose	Move data from GPS transceiver to ESP32
Source of input / destination of output	GPS transceiver / ESP32 in smartwatch
Valid range, accuracy, and/or tolerance	-
Units of measure	Kbps
Timing	1 Hz
Relationships to other inputs/outputs	-
Screen formats/organization	-
Window formats/organization	-
Data formats	-
Command formats	-

End messages	-

3.1.2.3 Connection between Hub and LAN

Name of item	Ethernet
Description of purpose	Connect Hub to local network
Source of input / destination of output	Hub Ethernet / Local network
Valid range, accuracy, and/or tolerance	cable length
Units of measure	Kbits
Timing	50 Kbit/s
Relationships to other inputs/outputs	-
Screen formats/organization	-
Window formats/organization	-
Data formats	-
Command formats	-
End messages	-

- The Hub and the computer running the website shall be connected to the same local network

3.1.2.4 LED Matrix

Name of item	Unicorn HAT HD Matrix LED
Description of purpose	Display information
Source of input / destination of output	LED pins / Hub GPIO 8-11
Valid range, accuracy, and/or tolerance	-
Units of measure	candela
Timing	update every 50 ms
Relationships to other inputs/outputs	-
Screen formats/organization	16x16 Pixels
Window formats/organization	-

Data formats	-
Command formats	-
End messages	-

3.1.2.5 Smartwatch button

Name of item	Button
Description of purpose	Start / stop hiking session
Source of input / destination of output	User finger / GPIO Microcontroller
Valid range, accuracy, and/or tolerance	-
Units of measure	reaction time (ms)
Timing	10ms
Relationships to other inputs/outputs	Primary interrupt
Screen formats/organization	-
Window formats/organization	-
Data formats	-
Command formats	-
End messages	-

3.1.3 Software interfaces

3.1.3.1 REST API

Name of item	REST API
Description of purpose	Display information on Hub
Source of input / destination of output	Hub / Browser on External Device
Valid range, accuracy, and/or tolerance	
Units of measure	ms
Timing	
Relationships to other inputs/outputs	-

Screen formats/organization	-
Window formats/organization	-
Data formats	HTTP Requests
Command formats	-
End messages	-

3.1.3.2 SSH interface for maintainer accessing Hub

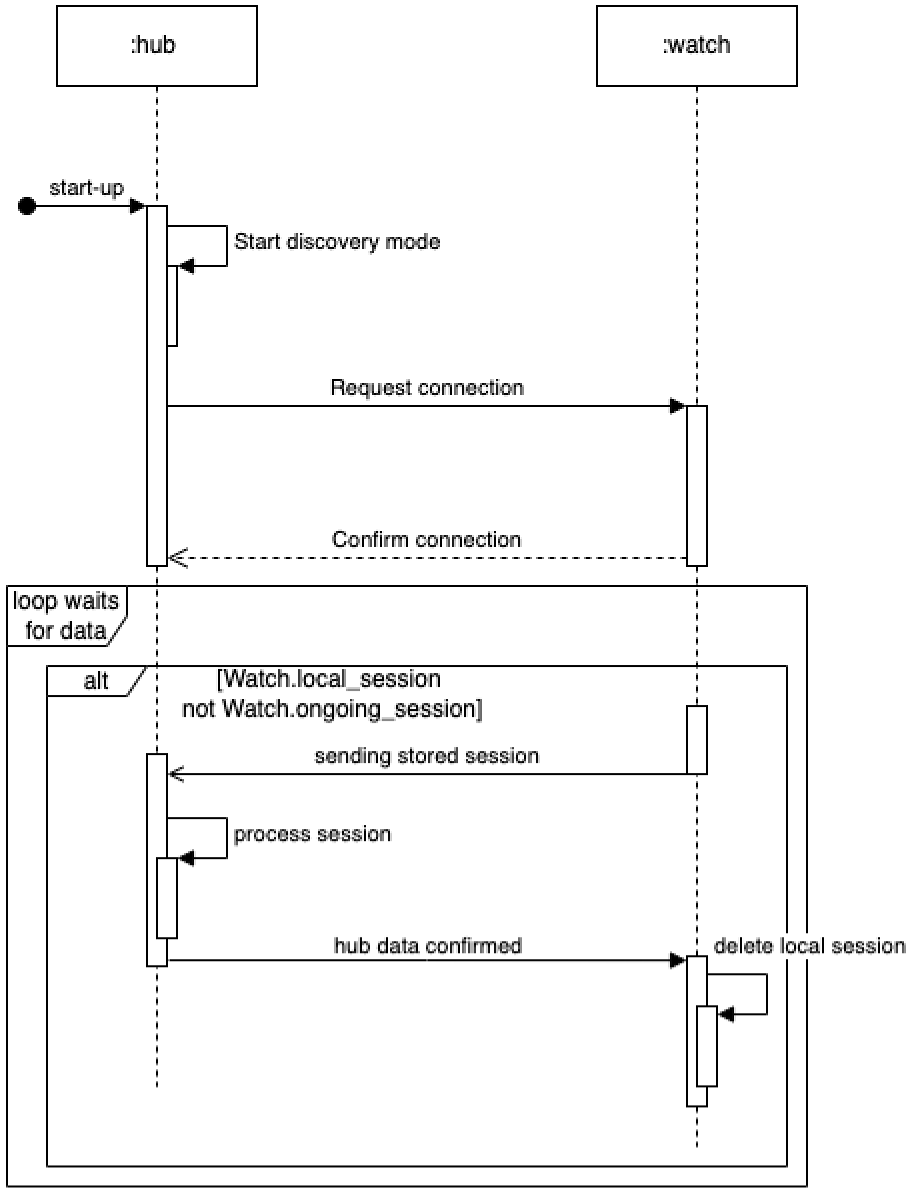
Name of item	SSH interface
Description of purpose	Maintainer access to the Hub
Source of input / destination of output	External device / Hub SSH port
Valid range, accuracy, and/or tolerance	-
Units of measure	Kbits
Timing	100 Kbits
Relationships to other inputs/outputs	-
Screen formats/organization	-
Window formats/organization	-
Data formats	SSH Protocol
Command formats	-
End messages	-

3.2 Functions

3.2.1 **Hub synchronization sequence**

The Hub will actively try to connect to the watch at all times when turned on. When the watch is in reach and available for connection, it will automatically connect to the Hub. After the connection is established, the watch will send session information to the Hub. If the session is received, the Hub will send a confirmation message to the watch. If a confirmation message is received, the watch will delete the stored session. This can be seen in section 3.2.1.1

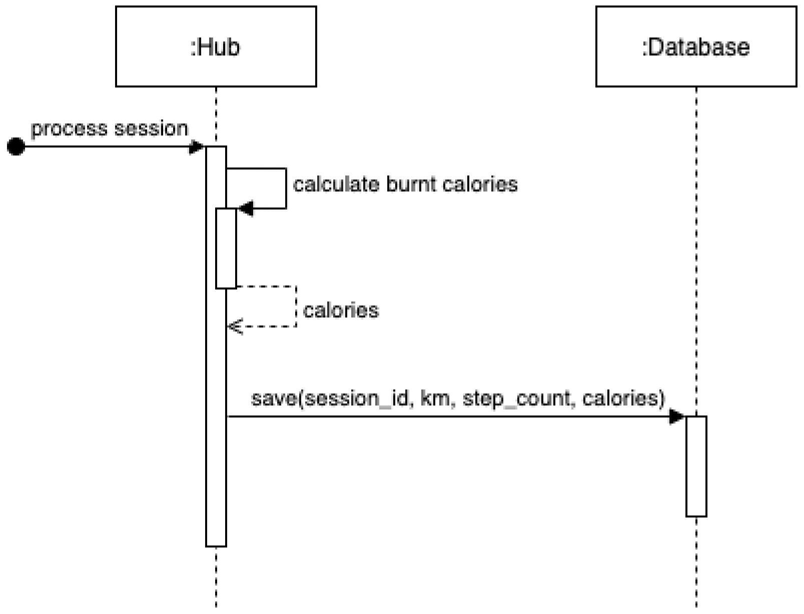
3.2.1.1 Hub synchronization sequence diagram



3.2.2 Hub processes incoming data from smartwatch (calculate burnt calories, save to database)

When the watch has transferred a hiking session to the Hub, the Hub will calculate the burnt calories before storing the session data received from the watch. Can be seen in Diagram 3.2.2.1

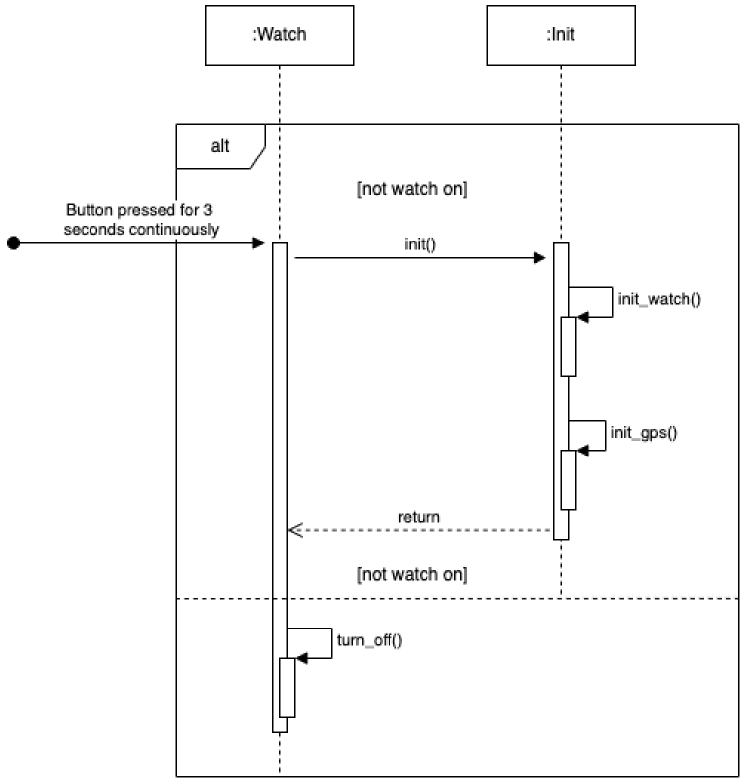
3.2.2.1 Hub processes incoming data from smartwatch diagram



3.2.3 Smartwatch long button press

If the button is pressed longer than 3 second continuously the watch will turn on/off depending on whether it was turned off or on initially. When turned on the watch will first go through initialization of the watch parameters before initializing and connecting to the GPS. Diagram can bee seen in 3.2.3.1

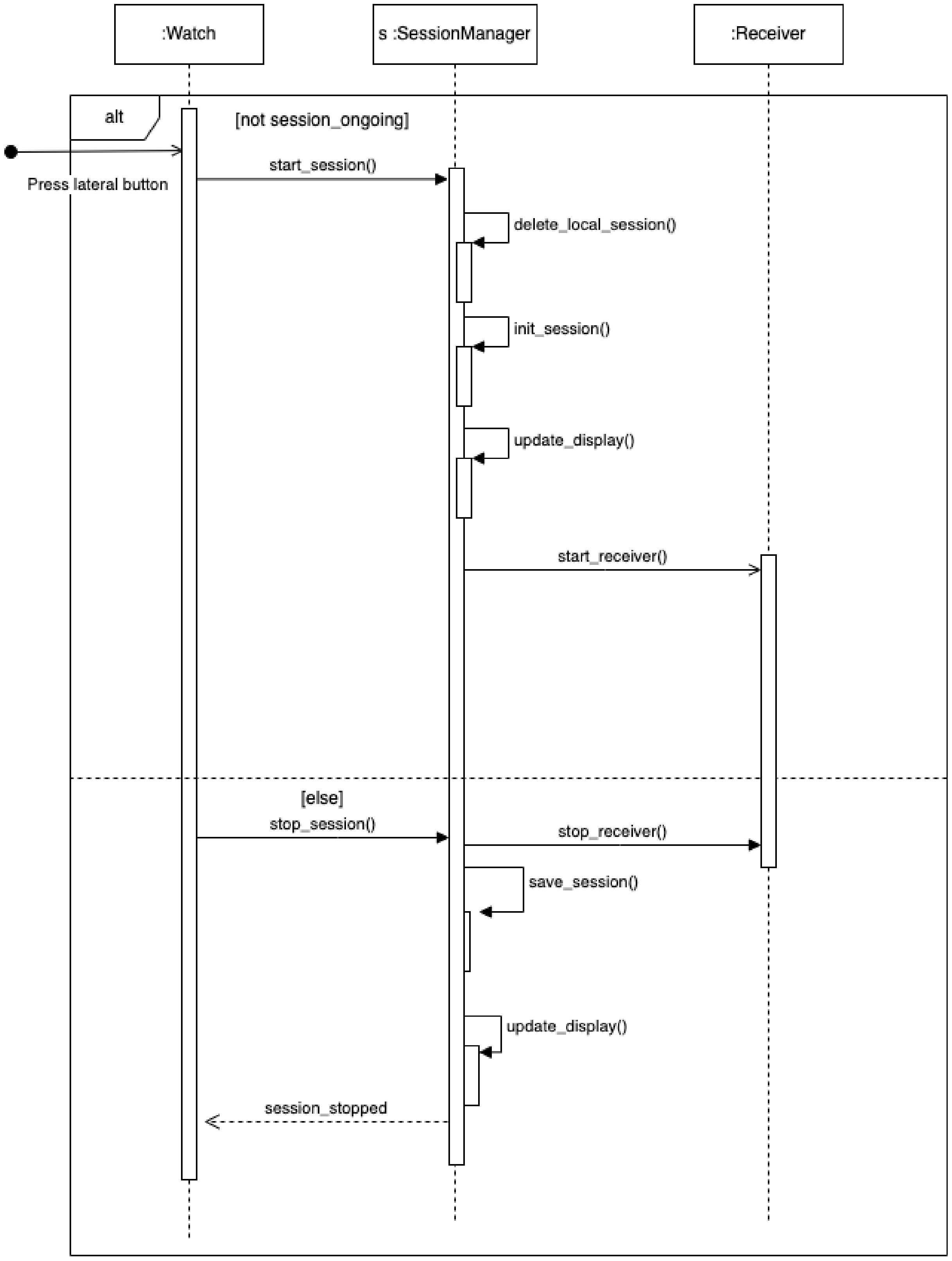
3.2.3.1 Smartwatch long button press diagram



3.2.4 **Smartwatch short button press**

When the button is pressed on a turned on watch, the watch will switch to hiking mode. If an old session is present when starting a new one, the previous session will be deleted and replaced. After the session is deleted, the watch will initialize the session before updating the display to show that the session is starting. After this the watch will move into the receiver loop which will be described in 3.2.5. When the button is pressed one time during an ongoing session, the session will be stopped. After the stop signal, the watch will first go into saving the session before updating the display and return to being available for Bluetooth transfer. The diagram can be seen at 3.2.4.1

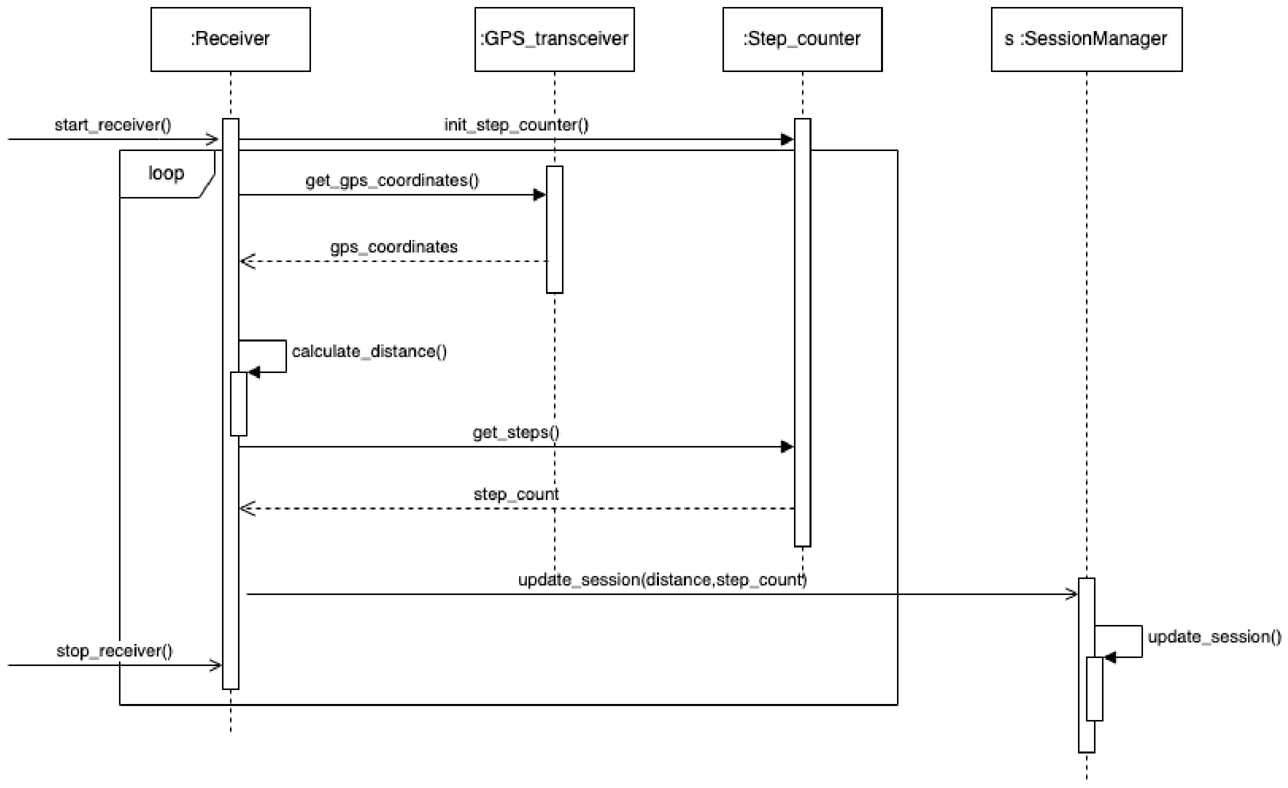
3.2.4.1 Smartwatch short button press diagram



3.2.5 Smartwatch receiving GPS data

After the session initialization is done, the watch switches into the receiver state. In the receiver loop, the watch will communicate with the GPS transceiver to get coordinates with a frequency of 1 Hz. When the coordinates are gathered, the watch will calculate the elapsed distance. When distance is calculated, the watch will fetch data from the step counter before updating the display with live data. Diagram available at 3.2.5.1

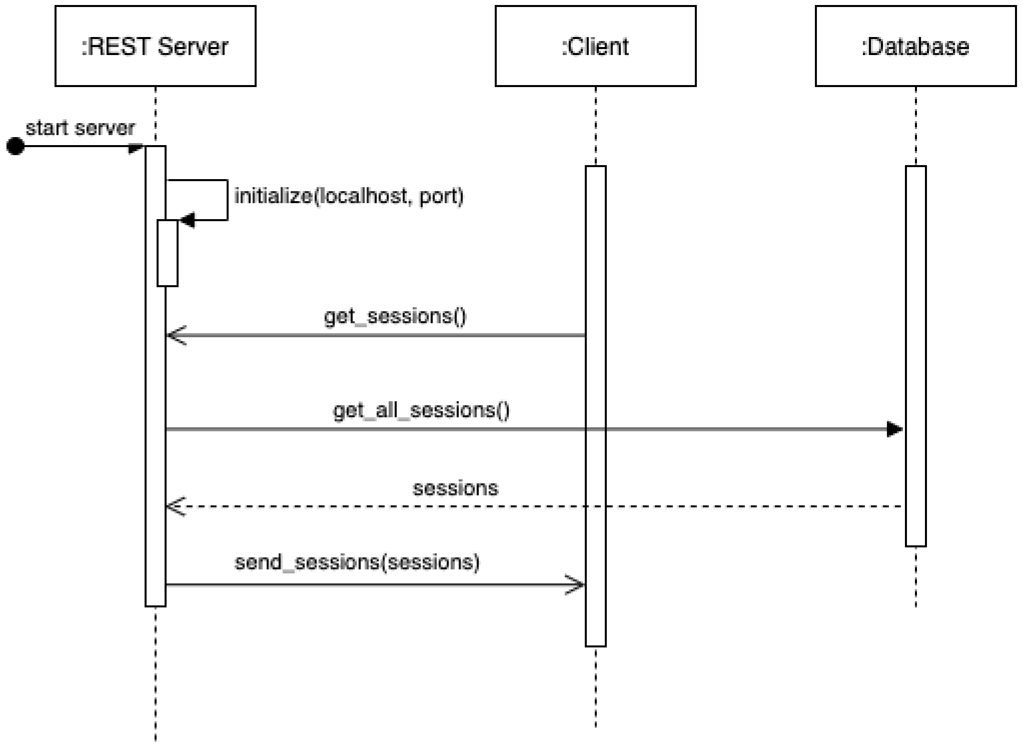
3.2.5.1 Smartwatch receiving GPS data diagram



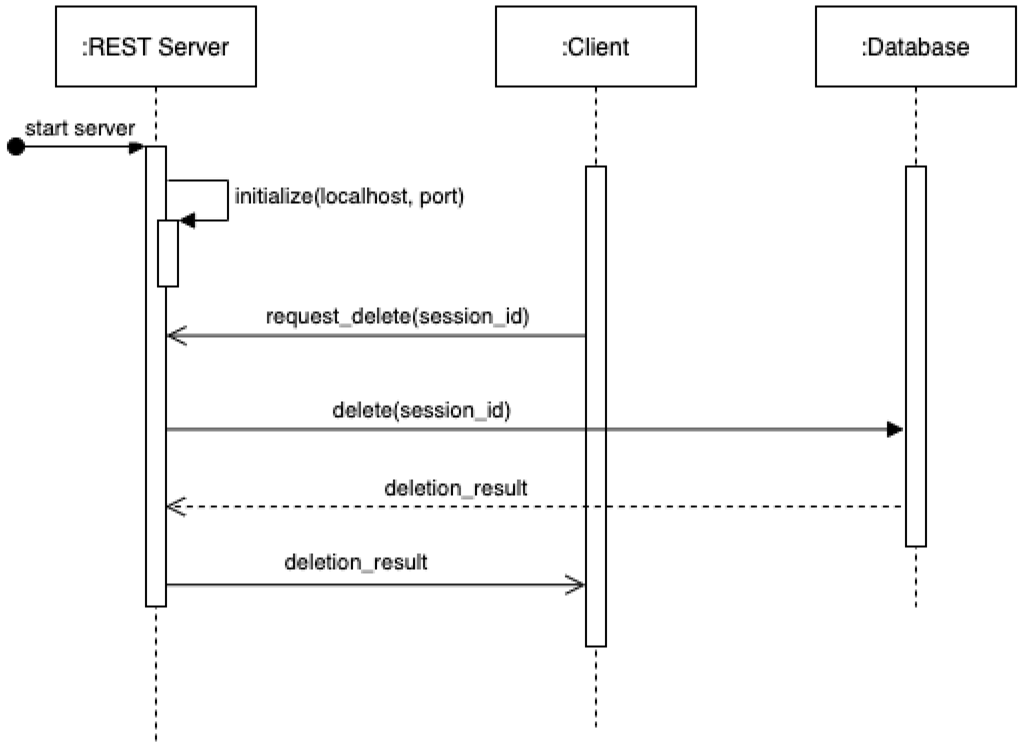
3.2.6 REST API

The client can make two requests through the REST API, GET and DELETE session. When the user makes a GET session request to the REST API server, the server will then communicate with the database and collect all sessions stored on the database before returning and display the info to the user. Visualized in 3.2.6.1. Similar to this, when the user makes a DELETE request with a specific session ID to the REST server it will communicate further with the database and delete the corresponding session. When done, the REST server will return the deletion status to the user. The DELETE sequence can be seen in 3.2.6.2.

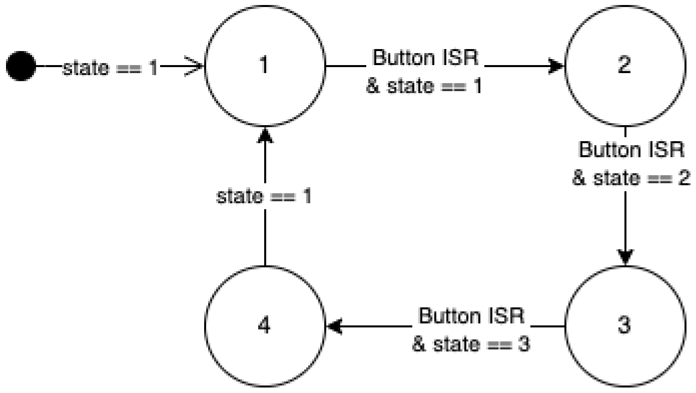
3.2.6.1 Get sessions



3.2.6.2 Delete session



3.2.7 State diagram for Watch



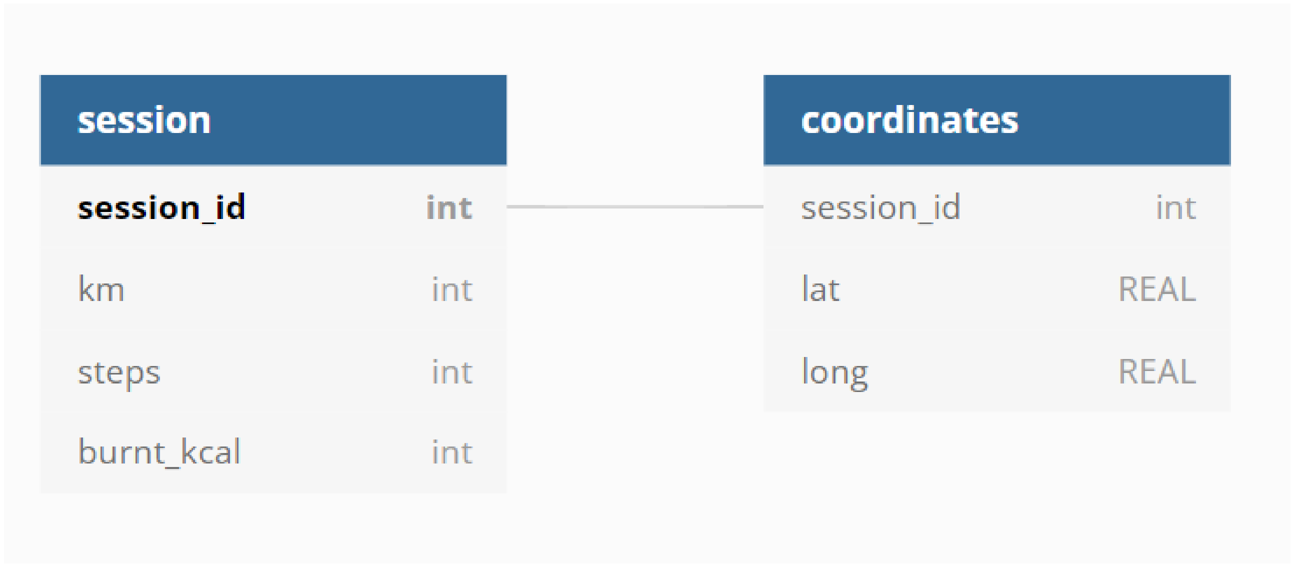
The states are defined as follows. State 1 is the initial state where the watch is available for Bluetooth data transfer, and for initializing the hike session. State 2 is the initializing of the hiking session objects. In State 3 the watch goes into the receiver loop visualized in 3.2.5.1. State 4 is saving the session before returning to the beginning of the loop.

3.3 Performance requirements

Requirement / *Component*	**Traffic**	**Timing**	**Security**
System	The data shall be controlled using the Hub.	–	The system can support only one user at a time (either the user or the maintenance person).
Watch	The watch shall be able to store one trip at a time.	The watch shall update the screen with the trip statistics every 10ms. The watch shall be able to update its global position at a frequency of 1Hz.	The watch shall be able to connect to the external device within 2 meters of clear passage.
Hub	The Hub shall be able to process the data under 2 seconds.	–	–
Led	–	The LED matrix shall	–

		be able to update the display every 50 ms.	
Website	–	–	The website shall display the data to the user only through the website hosted by LAN.

3.4 Logical database requirements



3.4.1 Frequency of use

The database shall be able to handle 10 requests per second.

In an extreme case:

- The Hub receives 1 session per second and saves it to the database.
- The database is accessed twice in a second through the website, with the website sending one delete instruction to the database per second
- The LED matrix depending on the message length has different writing speeds but in an extreme case it accesses the database 1 time per second.

3.4.2 Accessing capabilities

The database shall be able to be accessed concurrently.

3.5 Design constraints

3.5.1 Watch Memory 16 MB

3.5.1.1 With a sample rate of 1 Hz for the GPS where every sample is 2 values of float32 leads to approximately 182 hours of hiking data. This is calculated with room for the program and other data gathering.

3.5.1.2 When synced with Hub, all hiking data is deleted from watch after being transferred to the database.

3.6 Software system attributes

3.6.1 Reliability

3.6.1.1 In order to perform session synchronization, the Hub and the smartwatch should be within 2 meters from each other, without any obstacles between the two devices.

3.6.1.2 The battery of the smartwatch should be charged enough (preferably over 20%), to ensure successful data synchronization and session recording.

3.6.1.3 The external device that accesses the website through a browser shall have a steady connection through Ethernet with the Hub to work sufficiently.

3.6.1.4 The operating temperature of the smartwatch and Hub is between 10 °C and 40 °C.

3.6.2 Availability

3.6.2.1 The Hub has to be powered up to have full functionalities.

3.6.2.2 The Hub shall be connected to the internet via Ethernet port.

3.6.3 Security

3.6.3.1 Web page it is only on the local network and behind the router firewall.

3.6.3.2 The watch is attached to the hand of the user and can be accessed by no one else.

3.6.4 Maintainability

3.6.4.1 The user interface should be modular and easy to extend with additional views.

3.6.4.2 Hub shall use REST API so that new clients can be developed on top of the system.

3.6.5 **Portability**

3.6.5.1 The Hub code is dependent on Linux OS, apart from that it needs a device with a Bluetooth module and SPI controller to control the LED matrix.

3.6.5.2 The Hub implementation is dependent on Python 3.x.