

# Hiking Tour Assistant Design Report

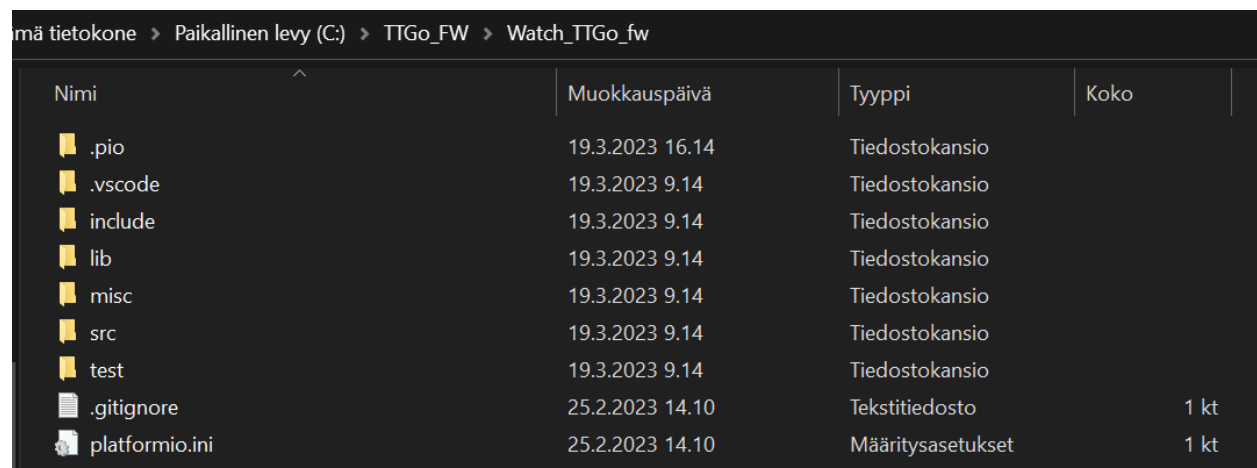
## Introduction

The Hiking Tour Assistant (HTA) is a software application that enables hikers to record their hiking sessions by using a smartwatch. The smartwatch records the steps taken by the user and converts them into distance. Data from the latest session is saved on the watch. The session data can be synchronized with an external hub device and displayed on a web page. The smartwatch used is a LilyGo TWatch and we are using a Raspberry Pi (RPI) as the hub device. The devices function on their own and can communicate with each other by using a Bluetooth connection.

## Program description

Smartwatch program:

The smartwatch program can be found in the 'Watch\_TTGo\_fw' folder. The folder is organized in the following way:



Nimi	Muokauspäivä	Tyyppi	Koko
.pio	19.3.2023 16.14	Tiedostokansio	
.vscode	19.3.2023 9.14	Tiedostokansio	
include	19.3.2023 9.14	Tiedostokansio	
lib	19.3.2023 9.14	Tiedostokansio	
misc	19.3.2023 9.14	Tiedostokansio	
src	19.3.2023 9.14	Tiedostokansio	
test	19.3.2023 9.14	Tiedostokansio	
.gitignore	25.2.2023 14.10	Tekstitiedosto	1 kt
platformio.ini	25.2.2023 14.10	Määrittäjäasetukset	1 kt

- .pio – contains the builds and the library dependencies
- .vscode – contains the VSCode extension files
- include – empty
- lib – empty
- misc – empty
- src – contains the source files of the project, including the main.cpp file
- test – empty

The source files of the project are in the src-folder. The functionality of the app is implemented in the *main.cpp* file. It contains helper functions for initializing the watch, sending data over Bluetooth, saving the session id, steps and distance, deleting the stored session, printing the Bluetooth MAC-address of the watch, and printing the status of the watch (state and irqButton). The two main functions *void setup()* and *void loop()* are used to setup the watch and switch between possible states in the *loop()* method, as shown in the figure below. The *loop()* method is executed at each clock cycle.

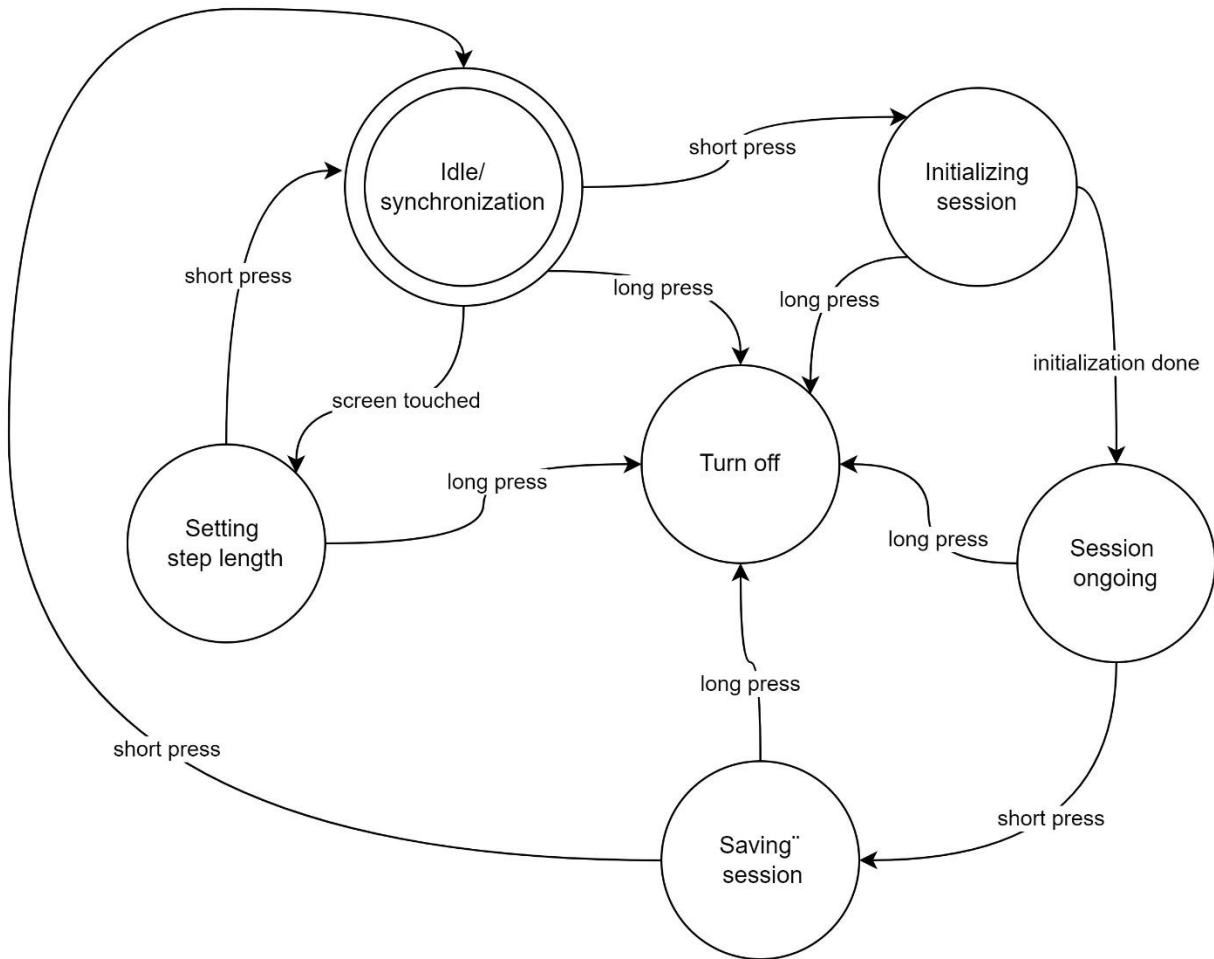


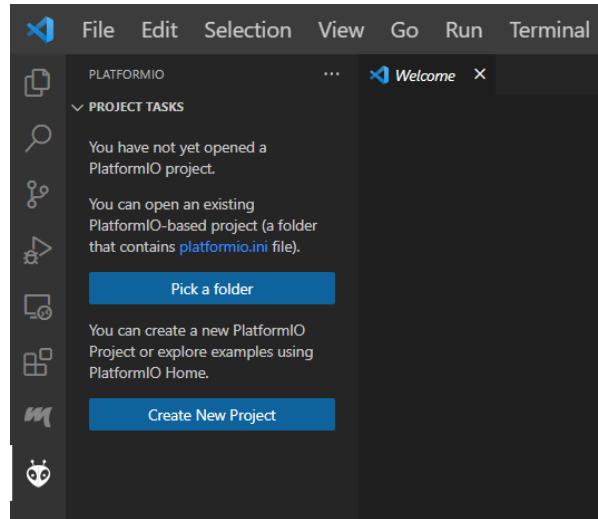
Figure 1: State machine showing how the smartwatch program switches between states. 'Idle' is the start state and 'Turn off' is a terminal state. 'Short press' indicates that the side button of the watch is pressed for less than 4 seconds and 'long press' means that the button is pressed for more than 4 seconds.

Instructions:

The development was done on Windows using VSCode and PlatformIO. Using the same environment is highly recommended.

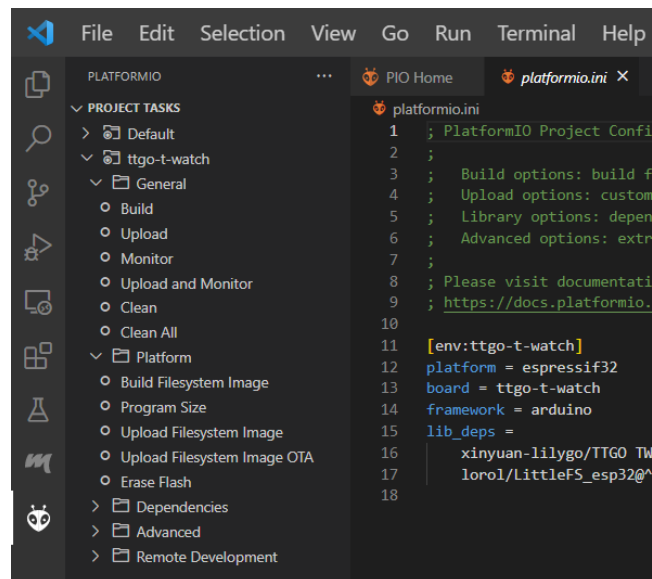
- VSCode can be downloaded here: <https://code.visualstudio.com/>
- PlatformIO installation guide: <https://platformio.org/install/ide?install=vscode>

After installing PlatformIO, you should be able to see the following tab.



Click 'Pick a folder' and choose ...\\TTGo\_FW\\Watch\_TTGo\_fw folder, which can be cloned from the GitHub repository. The folder contains a 'platformio.ini' which initializes the project automatically. **(NOTE: we had some issues running the Project from a folder connected to OneDrive, so we recommend saving the project folder to a physical drive)**

After the initialization of the project, you should see the following screen:



Here you can click on *Build*, to build an image of the project. This image can be then pushed to the LilyGo TWatch by clicking *Upload* assuming that the watch is connected with a USB-cable.

The library dependencies can be found in .pio/libdeps/ttgo-t-watch and they include:

- TTGO TWatch Library
- LittleFS\_esp32

RPi program:

The RPi program can be found in the 'RPi' folder. The folder is organized as follows:

- The root folder contains the Python code files and requirements information
- User data is saved in the 'data' folder
- HTML files used by the web server are in the 'templates' folder

Nimi	Muokauspäivä	Tyyppi	Koko
data	19.3.2023 12.59	Tiedostokansio	
templates	19.3.2023 12.59	Tiedostokansio	
app.py	19.3.2023 10.58	JetBrains PyCharm ...	2 kt
bt.py	19.3.2023 10.58	JetBrains PyCharm ...	7 kt
hike.py	19.3.2023 10.58	JetBrains PyCharm ...	1 kt
receiver.py	19.3.2023 10.58	JetBrains PyCharm ...	2 kt
requirements.txt	19.3.2023 10.58	TXT-tiedosto	1 kt

The program is divided into two parts: synchronization and web server. The data used by the system is in the 'data' folder. It includes a file called 'session', which is where the data from the latest session is saved, and a file called 'user', where the user's weight is saved.

The synchronization program is implemented in the *bt.py*, *hike.py* and *receiver.py* files. The *bt.py* file includes functionality for establishing the Bluetooth connection with the smartwatch and receiving session data from it. The *hike.py* file includes a class representing a hiking session and some functions related to it. The *receiver.py* file includes functionality for processing the session data and a main function to run the synchronization program. To execute the synchronization program, run *receiver.py* on the command line with the command 'python3 receiver.py'.

The web server is implemented in the *app.py* file. In addition, several HTML files corresponding to the different web pages can be found in the 'templates' folder. The web server can be run by running the *app.py* file.

The development was done on the RPi itself using Thonny as an IDE. It is also possible to develop on another platform using any IDE with Python support, such as. The program has the following dependencies:

- Python 3.9.2 or newer
- Flask 2.2.2
- PyBluez 0.23
- Pillow 8.1.2

The Git repository for the project can be found [here](#).