# Hiking Band: User Manual

Holappa, Heidi Lundén, Jaakko-Juhani Rislakki, Tuomas

ELEC-E8408 Embedded Systems Development, Aalto University. Group I.

## Table of contents

# 1 Introduction

This document is a comprehensive technical document for future developers and maintainers. Section 2 provides a detailed list of dependencies required for the project. Section 3 showcases step-by-step instructions on how to download, setup. compile the project source code and to flash the binary to the smartwatch. Followed by this, section 4 provides key architectural diagrams, a database schema, metadata information and additional structural details that were not yet covered in the Software Requirements Sepcification (SRS).

# 2 List of dependencies

## 2.1 Web application

The Web Application is a Python 3.x application that works on versions 3.10 and greater. The dependencies for the application are listed below and can be found in requirements.txt:

- astroid==3.3.8
- autopep8==2.3.2
- blinker==1.9.0
- click==8.1.8
- dill==0.3.9
- Flask==3.1.0
- greenlet==3.1.1
- isort==5.13.2
- itsdangerous==2.2.0
- Jinja2==3.1.5
- MarkupSafe==3.0.2
- mccabe==0.7.0
- platformdirs==4.3.6
- pycodestyle==2.12.1
- pylint==3.3.3
- python-dotenv==1.0.1
- tomli==2.2.1
- tomlkit==0.13.2
- typing_extensions==4.12.2
- Werkzeug==3.1.3

In this proof-of-concept Hiking Band system, the web application is built to run on a Raspberry Pi3B+ board with a Raspian Operation System. The web application requires a network connection to access external style libraries.

### 2.2 LilyGo T-Watch

The LilyGO T-Watch application is an Arduino application. The project officially supports LilyGo T-Watch V2, but MAY also work with V3 with configuration adjustment. Some code segments in source code are dependent on the chosen smartwatch version.

The smartwatch development has the following dependencies

- arduino-cli v. 1.1
- esp32 libraries v. 2.0.14
- python 3.10 or greater
- pyserial 3.5

## 3 Step-by-step instructions

### 3.1 LilyGO T-Watch: Installation and setup

Follow these instructions to set up the LilyGo Hiking application. Please pay careful attention to version numbers to ensure that installation proceeds successfully.

#### 3.1.1 Arduino-cli and esp32 libraries

1. Install arduino-cli (v.1.1):

https://arduino.github.io/arduino-cli/1.1/

2. Install esp32 libraries (v.2.0.14)

```
arduino-cli core update-index --config-file arduino-cli.yaml

arduino-cli core install esp32:esp32@2.0.14

python3 -m pip install pyserial
```

3. Test your board

```
arduino-cli board list

Port         Protocol Type              Board Name FQBN Core
/dev/ttyUSB0 serial   Serial Port (USB) Unknown
```

### 3.1.2 Compilation and upload to esp32

Use the following table to make your compilation:

| Device | Board/FQBN |
|---|---|
| ESP32_WROOM_32 | esp32:esp32:esp32-poe-iso |
| LILYGO_WATCH_2020_V2 | esp32:esp32:twatch |
| LILYGO_WATCH_2020_V3 | esp32:esp32:twatch |

For example for TWATCH V3:

```
DEVICE="LILYGO_WATCH_2020_V3"
FQBN=esp32:esp32:twatch
arduino-cli compile --fqbn $FQBN \
                    --build-path $(pwd)/build \
                    --build-property "build.extra_flags=-D $DEVICE -D ESP32" .
arduino-cli upload -p /dev/ttyUSB0 \
                   --fqbn esp32:esp32:esp32-poe-iso \
                   --input-dir $(pwd)/build .
```

> **Note**
>
> The device path may not be `/dev/ttyUSB0`. To verify the name of the USB-device, connect the smartwatch with the cable and use command `ls /dev/tty*`.

or

configure config.ini

```
./install.sh
```

> **Tip**
>
> The config.ini contains LilyGo T-Watch versions V2 and V3. To change the T-Watch version, change which version is uncommented. V3 is not officially supported, but both V2 and V3 T-Watches were used during development stage.
>
> - When V2 is selected, the GPS module in V2 is used.
> - With V3 distance is calculated based on an hard coded step length as detailed in the SRS.

### 3.1.3 Debugging

Add read and write access to usb device:

```
chmod 777 /dev/ttyUSB0
```

Read the serial:

```
picocom -b 115200 /dev/ttyUSB0
or
putty
or
screen /dev/ttyUSB0 115200
```

## 3.2 Web Application: Installation and setup

### 3.2.1 Requirements

The web application and the scripts have been designed for a Linux based Operating System. It is recommended to use the application on a Linux based Operating System. The web application officially supports Raspberry Pi3B+ with a Raspian Operating System.

The minimum Python version is 3.10. Versions for dependencies are listed in requirements.txt. Use of virtual environment is adviced, as detailed below in installation instructions.

These instructions assume that the user is using a Linux based Operating System with a bash terminal emulator. The installation may either be done manually or by using a convenience script provided in the project repository.

### 3.2.2 Option 1: Manual installation

First setup the virtual environment

```
python3 -m venv venv
```

Then install dependencies

```
pip install -r requirements.txt
```

If you add new dependencies, create an updated `requirements.txt` with the following command:

```
pip freeze > requirements.txt
```

#### 3.2.3 Option 2: Convenience script

Run the installation script with

```
./install.sh
```

> **Tip**
>
> Using the script is advisable. It for instance verifies that the currently active Python3 installation meets system requirements.

### 3.3 Running the application

Running the application may also be done manually or by using a convenience script.

#### 3.3.1 Option 1: Manually

To run the app use

```
flask --app src/app.py run
```

To debug:

```
flask --app src/app.py --debug run
```

#### 3.3.2 Option 2: Convenience script

To run the app use

```
./start-app.sh
```

To debug:

```
./start-app.sh debug
```

# 4 Technical documentation

## 4.1 Architectural diagrams

## 4.2 Database schema

## 4.3 Metadata

## 4.4 Structural designs