EPSON

Monitoring Application for Vibration Measurement System - User's Guide

Notice of Document

Evaluation board/kit and development tool important notice

1. This evaluation board/kit or development tool is designed for use for engineering evaluation, demonstration, or development purposes only. Do not use it for other purposes. It is not intended to meet the requirements of design for finished products.
2. This evaluation board/kit or development tool is intended for use by an electronic engineer and is not a consumer product. The user should use it properly and in a safe manner. Seiko Epson does not assume any responsibility or liability of any kind of damage and/or fire caused by the use of it. The user should cease to use it when any abnormal issue occurs even during proper and safe use.
3. The part used for this evaluation board/kit or development tool may be changed without any notice.

NOTICE : PLEASE READ CAREFULLY BELOW BEFORE THE USE OF THIS DOCUMENT

The content of this document is subject to change without notice.

1. This document may not be copied, reproduced, or used for any other purposes, in whole or in part, without the consent of Seiko Epson Corporation("Epson").
2. Before purchasing or using Epson products, please contact with our sales representative for the latest information and be always sure to check the latest information published on Epson's official web sites and sources.
3. Information provided in this document such as application circuits, programs, usage, etc., are for reference purpose only. Please use the application circuits, programs, usage, etc. in the design of your equipment or systems at your own responsibility. Epson makes no guarantees against any infringements or damages to any third parties' intellectual property rights or any other rights resulting from the information. This document does not grant you any licenses, intellectual property rights or any other rights with respect to Epson products owned by Epson or any third parties.
4. Epson is committed to constantly improving quality and reliability, but semiconductor products in general are subject to malfunction and failure. In using Epson products, you shall be responsible for safe design in your products; your hardware, software and systems are designed enough to prevent any harm or damages to life, health or property even if any malfunction or failure might be caused by Epson products. In designing of your products with using Epson products, please be sure to check and comply with the latest information regarding Epson products (this document, specifications, data sheets, manuals, Epson's web site, etc.). When using the information included in the above materials such as product data, chart, technical contents, programs, algorithms and application circuit examples, you shall evaluate your products both in stand-alone basis and within your overall systems. You shall be solely responsible for deciding whether or not to adopt and use Epson products.
5. Epson has prepared this document and programs provided in this document carefully to be accurate and dependable, but Epson does not guarantee that the information and the programs are always accurate and complete. Epson assumes no responsibility for any damages which you incurred by due to misinformation in this document and the programs.
6. No dismantling, analysis, reverse engineering, modification, alteration, adaptation, reproduction, etc., of Epson products is allowed.
7. Epson products have been designed, developed and manufactured to be used in general electronic applications (office equipment, communications equipment, measuring instruments, home electronics, etc.) and applications individually listed in this document ("General Purpose"). Epson products are NOT intended for any use beyond the General Purpose that requires particular/higher quality or reliability in order to refrain from causing any malfunction or failure leading to harm to life, health or serious property damage or severe impact on society, including, but not limited to listed below. Therefore, you are advised to use Epson products only for the General Purpose. Should you desire to buy and use Epson products for the particular purpose other than the General Purpose, Epson makes no warranty and disclaims with respect to Epson products, whether express or implied, including without limitation any implied warranty of merchantability or fitness for any particular purpose.
   [Particular purpose]
   Space equipment (artificial satellites, rockets, etc.)
   Transportation vehicles and their control equipment (automobiles, aircraft, trains, ships, etc.)
   Medical equipment (other than applications individually listed in this document) / Relay equipment to be placed on sea floor
   Power station control equipment / Disaster or crime prevention equipment / Traffic control equipment / Financial equipment
   Other applications requiring similar levels of reliability as the above
8. Epson products listed in this document and our associated technologies shall not be used in any equipment or systems that laws and regulations in Japan or any other countries prohibit to manufacture, use or sell. Furthermore, Epson products and our associated technologies shall not be used for developing military weapons of mass destruction, military purpose use, or any other military applications. If exporting Epson products or our associated technologies, you shall comply with the Foreign Exchange and Foreign Trade Control Act in Japan, Export Administration Regulations in the U.S.A (EAR) and other export-related laws and regulations in Japan and any other countries and follow the required procedures as provided by the relevant laws and regulations.
9. Epson assumes no responsibility for any damages (whether direct or indirect) caused by or in relation with your non-compliance with the terms and conditions in this document.
10. Epson assumes no responsibility for any damages (whether direct or indirect) incurred by any third party that you assign, transfer, loan, etc., Epson products.
11. For more details or other concerns about this document, please contact our sales representative.
12. Company names and product names listed in this document are trademarks or registered trademarks of their respective companies.

2022.08

Trademark

- Raspberry Pi is a trademark of Raspberry Pi Ltd.
- Microsoft and Windows are trademarks of the Microsoft group of companies.
- EPSON is a registered trademark of Seiko Epson Corporation.
- Other product names are trademarks or registered trademarks of the respective companies

Table of contents

Notice of Document ····· 2

Trademark ····· 3

Revision History ····· 5

1. Related Document ····· 6

2. Introduction ····· 7

3. Specifications ····· 8
   - 3.1. Verified Operating Environment ····· 8
   - 3.2. Input Specifications ····· 8
   - 3.3. Output Specifications ····· 8
     - 3.3.1. Home Screen ····· 8
     - 3.3.2. VC Graph Screen ····· 9
     - 3.3.3. Trend Graph Screen ····· 9
   - 3.4. Configuration Parameters ····· 10

4. Setting Up on Raspberry Pi ····· 11
   - 4.1. File and Folder Structure ····· 11
   - 4.2. Preparation ····· 11
   - 4.3. Transferring the Program to Raspberry Pi ····· 11
   - 4.4. Building the Runtime Environment on Raspberry Pi ····· 12
     - 4.4.1. Creating a Python Virtual Environment ····· 12
     - 4.4.2. Installing Required Packages ····· 12
     - 4.4.3. Application Configuration ····· 12
     - 4.4.4. Installing the Service Registration File ····· 12

5. Running the Application ····· 13
   - 5.1. Preparation ····· 13
   - 5.2. Starting the Application ····· 13
   - 5.3. Measuring Vibration ····· 13
   - 5.4. Running the Hardware Status Monitoring Tool ····· 13
   - 5.5. Running the Summary Application ····· 13

6. Appendix : Developer Guide ····· 14
   - 6.1. Program Structure ····· 14
     - 6.1.1. Subscriber class ····· 14
     - 6.1.2. DataManager class ····· 14
     - 6.1.3. Dash application ····· 14
     - 6.1.4. DataSet class ····· 14
   - 6.2. Running Locally ····· 15
     - 6.2.1. System Requirements ····· 15
     - 6.2.2. Setting Up the Local Environment ····· 15
     - 6.2.3. Running the Program ····· 16
   - 6.3. Reference Links ····· 16

7. Contact Information ····· 17

Revision History

Rev. No.	Rev. Date	Page	Rev. Contents
20250331	2025/3/31	ALL	First edition Corresponding to the release of MSG005-001a_v1.0.0
20250731	2025//7/31	---	Corresponds to MSG005-001a_v1.0.1 release. Corrected description due to missing transfer files. Modified service definition file due to missing startup settings at OS boot.
20251130	2025/11/30	---	Corresponds to MSG005-001a_v1.1.0 release. Added descriptions related to the support for “Machine Condition Monitoring System based on ISO 20816.” Added Trend screen to support the summary application. Modified design and screen names of existing UI. Updated source code structure and revised setup procedures.

1. Related Document

1. “Setup Manual for Vibration Measurement System Using Raspberry Pi Products” Rev.20250731
2. “Operation Manual for Vibration Measurement System Using Raspberry Pi Products” Rev.20250731
3. “VC Evaluation Vibration Measurement System – User’s Guide” Rev.20250331
4. “Machine Condition Monitoring System based on ISO 20816 – User’s Guide” Rev.20251130

2. Introduction

This user guide provides an overview of the Vibration Monitoring System Web Application (hereafter referred to as “the application”), covering the following topics:

- Specification
- Setup procedures
- Execution instructions
- Developer guide

The **"Vibration Measurement System Monitoring Application"** (hereinafter referred to as "this application") is a web application that runs on a Raspberry Pi equipped with either the **"Vibration Measurement System using Raspberry Pi products"** or the **"VC Evaluation Vibration Measurement System"**(collectively referred to as "measurement applications").

This application receives messages via an **MQTT broker**, which acts as an intermediary for messages sent from the measurement applications and their associated **hardware status monitoring tools**. It displays the measurement status, the status of connected vibration sensors, VC evaluation results, and more on a web interface.

Additionally, it receives MQTT messages from the “**Machine Condition Monitoring System Based on ISO 20816”** (hereinafter referred to as “summary application”), which also runs on Raspberry Pi, and displays the aggregated results of past measurement data collected by the measurement applications on the web interface.

By accessing the Raspberry Pi from a PC using a web browser, users can view the web application interface.

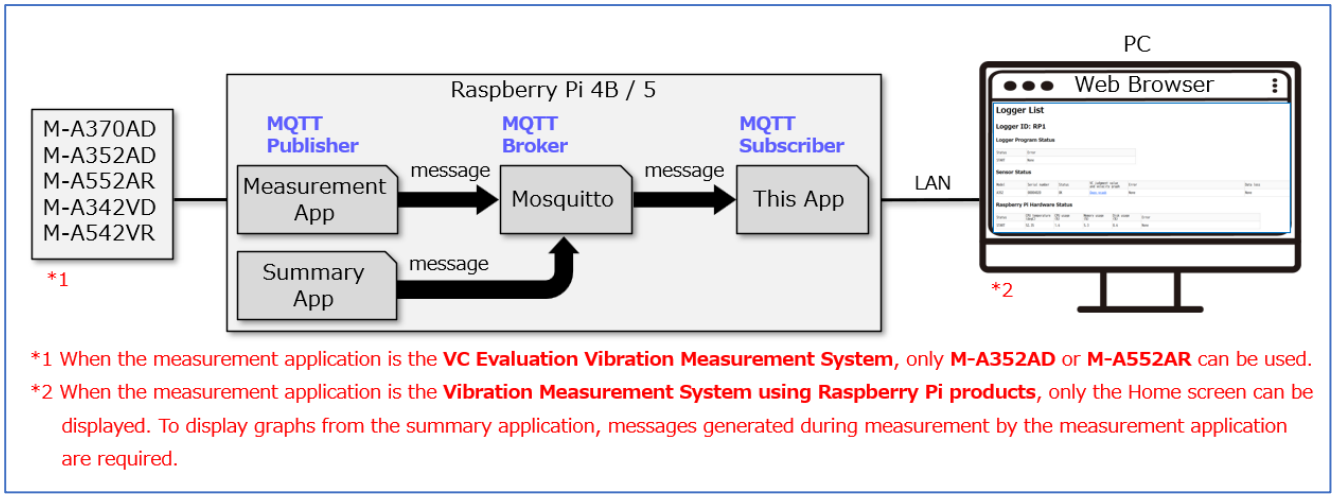


Figure 2-1 Application Diagram

3. Specifications

3.1. Verified Operating Environment

This application has been tested and confirmed to operate on the following Raspberry Pi models:

- Raspberry Pi 4B
- Raspberry Pi 5

It has also been verified to work with the following web browsers:

- Google Chrome
- Microsoft Edge

3.2. Input Specifications

This application receives messages from the following tools:

- Vibration Measurement System using Raspberry Pi products
- Hardware Status Monitor included with the above system
- VC Evaluation Vibration Measurement System
- Machine Condition Monitoring System based on ISO 20816

For detailed message specifications, please refer to the related documents 2, 3 and 4.

3.3. Output Specifications

As this is a web application, the output is presented via web pages. The application consists of two main web screens.

3.3.1. Home Screen

This screen displays the status of the logger and vibration sensors connected to the measurement application via the MQTT broker. It updates every second based on messages received from the measurement applications.

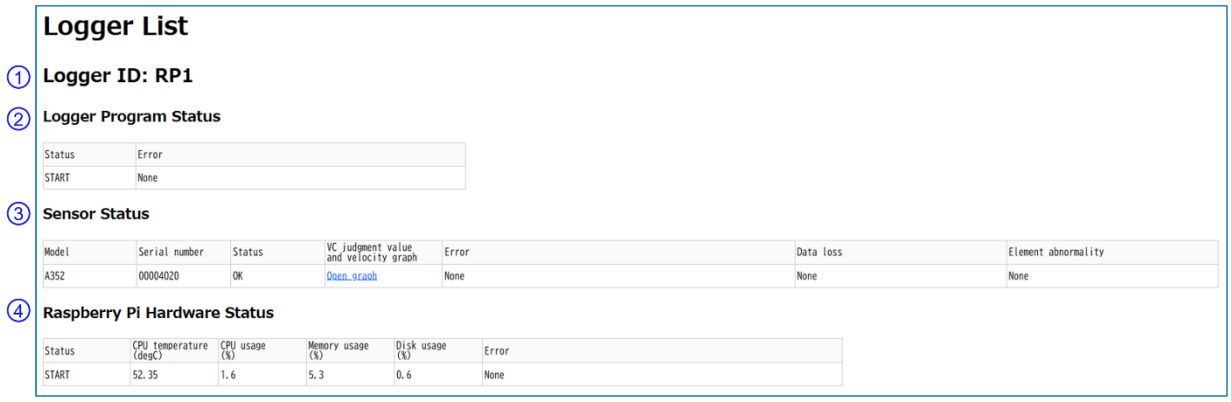


Figure 3-1 Home Screen

1. Displays the ID of the connected logger
2. Shows the status of the connected logger
3. Displays the status of vibration sensors connected to the logger

- In the “VC graph” and “Trend graph” columns, a link labeled “Open graph” will appear when the required data for each graph is available. Clicking the link will open the corresponding graph screen.
- The background color of the “Trend graph” column changes depending on the warning status received from the summary application.
  (Trip warning active Alarm warning active)

4. Displays Raspberry Pi system metrics such as CPU usage

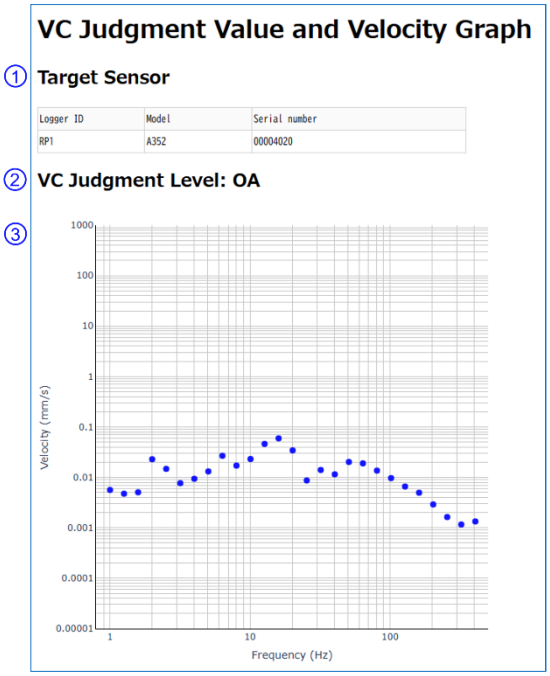
Note: To display the link for “VC graph,” the VC Evaluation Vibration Measurement System must be running.

Note: To display the link for “Trend graph,” the system must be receiving status messages for the corresponding sensor from the **measurement application** and also receiving aggregation messages for that sensor from the **summary application**.

3.3.2. VC Graph Screen

This screen displays the VC evaluation values and velocity graphs of the connected vibration sensors.

Messages sent from the “VC Evaluation Vibration Measurement System” are reflected on the screen every



second..

Figure 3-2 VC Graph Screen

1. Display information about the vibration sensor currently being graphed
2. Shows the VC evaluation value of the vibration data measured by the sensor
3. Displays a velocity graph of the vibration data measured by the sensor

3.3.3. Trend Graph Screen

This screen displays the trend values and change in magnitude graphs aggregated by the **summary application**. Messages sent from the summary application are reflected on the screen at one-minute intervals.

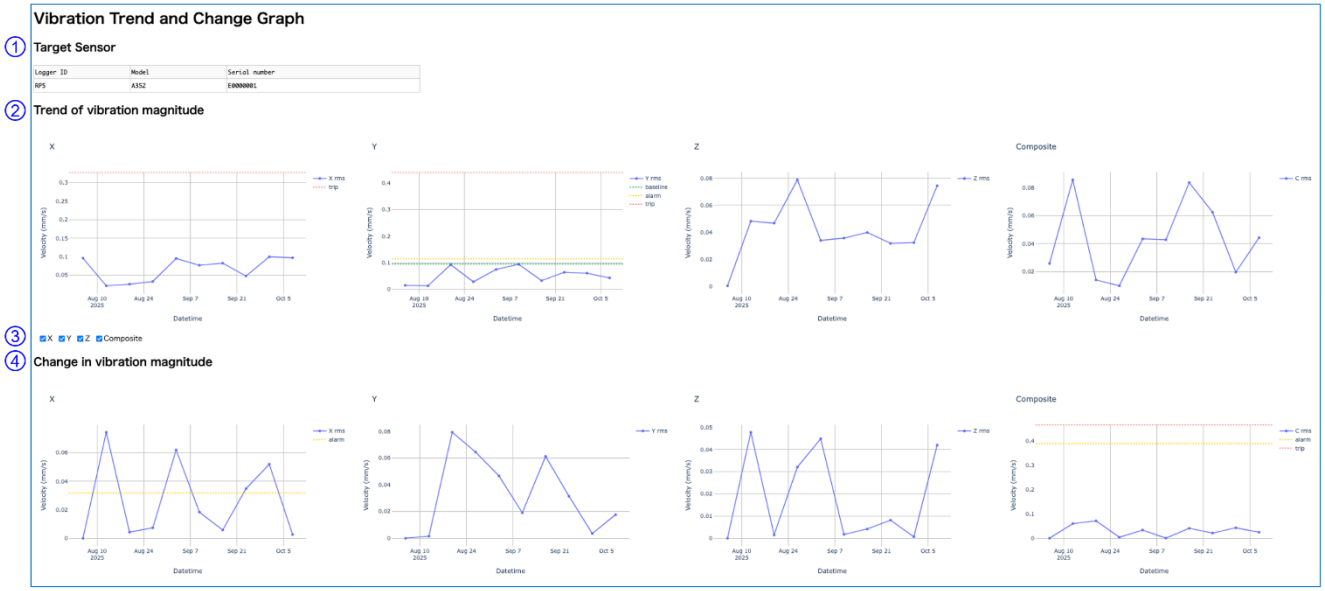


Figure 3-3 Trend Graph Screen

1. Displays information about the vibration sensor shown in the graph
2. Displays the aggregated trend values for each axis: X, Y, Z, and Composite
3. Axis selection checkboxes apply to both trend values and variation values
4. Displays the aggregated variation values for each axis: X, Y, Z, and Composite

3.4. Configuration Parameters

The application reads its configuration from a ".env" file located in the setup directory. The following parameters can be configured.

Table 3-1 Application Configuration Items

Parameter Name	Description	Default Value	Notes
LOG_LEVEL	Log output level	INFO	
APP_HOST	Hostname for accessing the application	192.168.1.52	Specify the hostname used to access via browser
APP_PORT	Port number for accessing the application	8050	
MAX_MESSAGE_COUNT	Number of entries to retain for multi-value data	3	Number of error entries to retain
MQTT_BROKER_HOST	Hostname of the MQTT	localhost	
MQTT_BROKER_PORT	Port number of the MQTT broker	1883	

4. Setting Up on Raspberry Pi

This section explains the steps to set up the application on a Raspberry Pi.

4.1. File and Folder Structure

Extracted Folder Structure:

```
Extracted Folder
├──── pyproject.toml          # Python project definition file
├──── sandbox                 # Folder for test and investigation programs
├──── src / raspi_web         # Source code of the application
│     ├──── data              # Data definitions used in the app
│     ├──── infra             # Code for handling messages from the MQTT broker
│     ├──── ui                # Code for building the web UI
│     └──── util              # Utility code used across multiple components
└──── tests                   # Test code for the application
```

Figure 4-1 File and Folder Structure

4.2. Preparation

Before starting the setup, refer to Related Documents1 and 3 and complete the following preparations:

- Connect the Raspberry Pi and PC to the same network
- Ensure the Raspberry Pi has internet access
- Install the measurement application on the Raspberry Pi
- Install the MQTT broker (Mosquitto) on the Raspberry Pi
- Enable message transmission to the MQTT broker in the measurement application settings

The following instructions assume:

- Raspberry Pi username : `pi`
- Static IP address assigned to Raspberry Pi : `192.168.1.52`

If you are using a different username or IP address, please adjust the commands accordingly.

4.3. Transferring the Program to Raspberry Pi

Run the following commands to transfer the necessary files to the Raspberry Pi:

1. Create the target directory on the Raspberry Pi:
   - `ssh pi@192.168.1.52 "mkdir -p /app/MSG005-001a"`
2. Transfer the program files:
   - `scp -r bin src pyproject.toml .env.default pi@192.168.1.52:/app/MSG005-001a`

4.4. Building the Runtime Environment on Raspberry Pi

Log in to the Raspberry Pi:

- `ssh pi@192.168.1.52`

4.4.1. Creating a Python Virtual Environment

Move to the application directory and create a virtual environment:

- `cd /app/MSG005-001a`
- `python -m venv --upgrade-deps venv`
- `source venv/bin/activate`

4.4.2. Installing Required Packages

Install the required packages in the virtual environment:

- `pip install .`

4.4.3. Application Configuration

Copy the default configuration file and create a working .env file:

- `cp .env.default .env`

Edit the .env file to adjust configuration values as needed. Refer to section 3.43.4 Configuration Parameters for details.

4.4.4. Installing the Service Registration File

Install the service registration file with the following command:

- `sudo cp bin/logger_monitor.service /etc/systemd/system`

Note: The service file assumes the Python virtual environment is located at /app/MSG005-001a/venv. If your environment is in a different location, update the following line in the service file accordingly:

- `ExecStart=/app/MSG005-001a/venv/bin/python -m raspi_web`

To enable the application to start automatically on Raspberry Pi boot:

- `sudo systemctl enable logger_monitor`

5. Running the Application

5.1. Preparation

Before running the application, refer to Related Documents1 and 3 and complete the following preparations:

- Connect the Raspberry Pi and PC to the same network
- Connect the accelerometer to the Raspberry Pi

5.2. Starting the Application

If you have followed the steps in Section 4.4.4 Installing the Service Registration File and configured the application to start automatically on OS boot, the web application will launch using the host and port specified in the configuration file (APP_HOST, APP_PORT).

- `Example: http://192.168.1.52:8050`

If the application is not set to start automatically, you can start it manually with the following command:

- `sudo systemctl start logger_monitor`

Note: When starting the application, the following warning may appear in the logs:

"`WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.`"

This indicates that the internal web server is intended for development use only. Avoid using it in publicly accessible network environments.

To access the application, open a web browser on your PC and enter the following URL:

- `http://192.168.1.52:8050/`

Note: If You Cannot Access the Application, please log in to the Raspberry Pi, enter the following command and check the log output to investigate the issue.

`systemctl status logger_monitor`

5.3. Measuring Vibration

Once the application is running, you can start the measurement application to begin vibration measurement. The monitoring application will display the status of the measurement application and the connected accelerometers.

For details on how to perform measurements, refer to Related Documents 2 and 3.

5.4. Running the Hardware Status Monitoring Tool

By running the hardware status monitoring tool, you can view system metrics such as CPU and memory usage of the Raspberry Pi within the application.

For instructions on how to run the tool, refer to Related Document 2.

5.5. Running the Summary Application

By running the summary application in this state, you can view the aggregated results of past measurement data collected by the measurement application within this application.

For details on how to run the summary application, please refer to Related Documents 4

6. Appendix : Developer Guide

This section provides technical information for developers who wish to extend or modify the application.

- Program Structure
- Running Locally
- Running the Program

6.1. Program Structure

The application consists of the following three main modules:

- `Subscriber class` (Path: `./src/raspi_web/infra/subscriber.py`)
- `DataManager` class (Path: `./src/raspi_web/infra/data_manager.py`)
- `Dash` application (Path: `./src/raspi_web/__main__.py`)

Additionally, the application defines a data structure class:

- `DataSet class` (Path: `./src/raspi_web/data/data_set.py`)

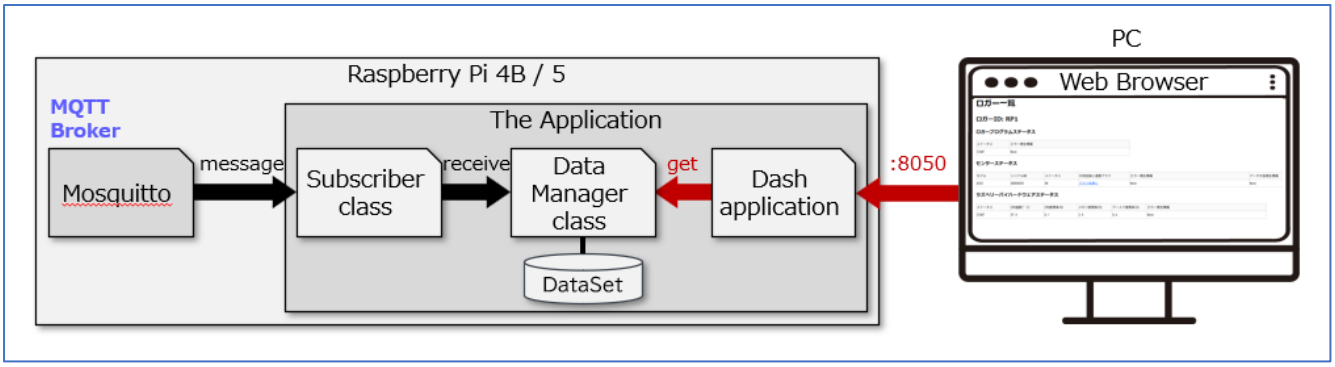


Figure 6-1 Program Structure Diagram

6.1.1. Subscriber class

Receives messages from the MQTT broker and passes them to the DataManager.

6.1.2. DataManager class

Parses messages received from the Subscriber based on their topics and stores data for display in the Dash interface. It also provides data retrieval methods for the Dash UI.

To add a new topic, update the `_init_topic_pattern_list()` method in the source code with the appropriate topic pattern and handler.

6.1.3. Dash application

Builds the web UI. It fetches data from the DataManager every second and updates the screen. Modify this module to add or change UI components.

6.1.4. DataSet class

Defines the data types used by DataManager and the Dash application.

6.2. Running Locally

When adding features or making modifications, development is typically done on a local PC. This section explains:

- How to set up the application locally
- How to run a local MQTT broker

6.2.1. System Requirements

To run the application locally, the following environment is required:

- Python 3.11
- Docker Desktop or another Docker-compatible environment
- Docker Compose v2 (verified with v2.31.0)

6.2.2. Setting Up the Local Environment

Extract the downloaded ZIP file to a folder of your choice on your local PC and set up the environment by following the steps below.

6.2.2.1. Creating Python Virtual Environment

On Windows (PowerShell):

- Launch PowerShell and execute the following command.
  - `py -3.11 -m venv --upgrade-deps venv`
  - `venv\scripts\activate.ps1`
- If `py` command is not found, please try the following alternatives.
  - `python3.11 -m venv --upgrade-deps venv`
  - `venv\scripts\activate.ps1`

On macOS/Linux (Terminal):

- Launch Terminal and execute the following command.
  - `python3.11 -m venv --upgrade-deps venv`
  - `source venv/bin/activate`

6.2.2.2. Installing Packages

Install required packages:

Common to all OS:

- `pip install -e .`

6.2.2.3. Starting the MQTT Broker (Mosquitto)

Use Docker to start the MQTT broker using the `./sandbox/mqtt/compose.yml` file.

On Windows:

- `cd sandbox\mqtt`
- `docker compose up -d`
- `cd ..\..`

On maOS/Linux:

- `cd sandbox/mqtt`
- `docker compose up -d`
- `cd ../..`

6.2.2.4. Creating the Configuration File

Copy the default configuration file and create a working `.env` file:

- `cp .env.default .env`

Edit the `.env` file and set `APP_HOST` to `localhost`. Modify other settings as needed.

6.2.3. Running the Program

6.2.3.1. Starting the Application

Start the web application with the following command:

- `python -m raspi_web`

Access the application in your browser at: `http://localhost:8050/`

6.2.3.2. Sending Sample Messages

A sample message sender script is available at `sandbox/mqtt/publish.py`.

Edit the following fields at the bottom of the script to match the message you want to send:

- `Logger_id`
- `model`
- `serial`
- `topic_type`

Then run:

- `python ./sandbox/mqtt/publish.py`

6.2.3.3. Deleting Sample Messages

Sample messages are sent with the "retain" flag enabled. To delete retained messages, use the script at `sandbox/mqtt/none_publish.py`.

Edit the same fields as above, then run:

- `python ./sandbox/mqtt/none_publish.py`

6.3. Reference Links

- Dash Python User Guider (https://dash.plotly.com/)
  - Official documentation for the Dash web application framework.

7. Contact Information

Seiko Epson Corporation

Sales Headquarters MD Sales Department

Contact Information via the Internet

https://www.epsondevice.com/sensing/en/contact/