WISE-WARE Software Requirements Specification

Contents

Introduction

Product scope

WISE-WARE is a home automation system that integrates smart home devices and services from many brands into a single intelligent, ambient assistive living space. Data from the devices used is aggregated for analysis, whether assisting the automation of the home, helping a resident's care services monitor their health, or contributing to research.

Intended audience

The intended audience is older adults living with frailty, or those vulnerable to frailty.

Product value

The target audience will find value in being able to automate aspects of their home to make it more assistive and convenient, while being able to avoid the biggest issues that commonly prevent users from benefiting from smart homes - ease of use and brand lock-in.

Intended use

The intended audience will use WISE-WARE to monitor, control and automate various aspects of their home, such as lighting, climate control, and security.

General description

WISE-WARE is a system that runs on a Raspberry PI and supervises containerised services like Home Assistant and Apache Kafka. These services are integrated with each other to allow the consolidation of their individual functions and collected data into a single entity, opening the door for future developments in machine learning and ambient assistive intelligence.

F - Functional Requirements

F1. WISE-WARE shall provide a user interface for monitoring and controlling various aspects of the smart home environment.

Details: This functionality leverages containerised services like Home Assistant to perform tasks and allow actions like:

- Lighting: Users shall be able to turn lights on/off, adjust brightness, and set schedules.
- Climate Control: Users shall be able to adjust thermostats, control fans, and set temperature schedules.
- Security: Users shall be able to view live security camera feeds, receive alerts for security breaches, and remotely control locks.

The UI for each service may differ, but all will be accessible through a web browser with a responsive interface optimised for mobile devices, and all will be integrated to function together.

Reference: For more information on how Home Assistant has been used in smart homes, refer to: https://www.home-assistant.io/examples/.

F2. WISE-WARE shall integrate with a variety of smart home devices, allowing users to control them from a central hub.

Details: Supported devices are typically distinguished by their communication protocol(s), such as BLE, WiFi and Zigbee. Some are supported out-of-the-box like BLE and WiFi, while others require additional addon modules like Zigbee.

Reference: For a list of devices and services compatible with Home Assistant - a component of WISE-WARE with the easiest integration processes - refer to: https://www.home-assistant.io/integrations/.

F3. WISE-WARE shall collect data from integrated smart home devices and securely store it for analysis.

Details: WISE-WARE uses Apache Kafka as an event streaming platform to collect and durably store data from various sensors and devices. This data can then be used for further analysis or automation purposes. User privacy is a priority, and data will remain on-site by default, but can be transmitted externally with the user's permission.

Reference: For an introduction to the concepts important to understanding Apache Kafka, refer to: https://docs.confluent.io/kafka/introduction.html.

F4. WISE-WARE shall allow users to create custom automations using a user-friendly interface.

Details: WISE-WARE includes Node-RED, a visual programming tool, to empower users to create automations without extensive coding knowledge. Users can create rules that trigger actions based on sensor data or other events within the smart home environment, and customise (or have customised on their behalf) their WISE-WARE to suit their unique needs.

Reference: For more information about Node-RED and how it is used, refer to: https://nodered.org/about/.

E - External Interface Requirements

Note: External interface requirements over the following pages are split into “External User Interface” requirements and “External Hardware Interface” requirements.

EU - User Interface Requirements

EU1. WISE-WARE’s UIs shall be accessible through modern web browsers on various devices.

Details: The UI shall be compatible with the latest two versions of popular web browsers (e.g., Chrome, Firefox, Safari) at the time of release. This ensures compatibility with most mobile and desktop devices used by consumers. The security considerations of a particular implementation of WISE-WARE will dictate the minimum supported browser version to ensure compatibility with secure protocols.

EU2. WISE-WARE’s UIs shall adhere to accessibility guidelines to ensure usability for users with disabilities.

Details: The UI shall comply with the World Wide Web Consortium's (W3C) Web Content Accessibility Guidelines (WCAG) 2.1 Level AA conformance criteria. This ensures the UI is accessible to users with a wide range of disabilities.

Reference: For more information on what constitutes the Web Content Accessibility Guidelines, refer to: https://www.w3.org/TR/WCAG21/

EU3. WISE-WARE’s UIs shall be responsive and adapt its layout for optimal viewing on various devices with different screen sizes.

Details: The UI shall be designed to function seamlessly across a range of viewports, including desktops, tablets, and smartphones.

EH - Hardware Interface Requirements

EH1. WISE-WARE shall run on a Raspberry Pi model 4B or Raspberry Pi model 5.

Details: These models provide sufficient processing power and memory to run WISE-WARE effectively. Other Raspberry Pi models may be compatible in the future, but initial development and testing will target these specific models.

Reference: For more information about the different models of Raspberry PI, refer to their website: https://www.raspberrypi.com/products/

EH2. WISE-WARE requires a Wi-Fi router to function normally, and internet access (and it’s SSID and password) for initial setup, software updates, and certain optional functionalities.

Details: A stable internet connection allows WISE-WARE to download updates and potentially enable features like remote management, contributions to data sets, or cloud analysis (depending on implementation). Internet access is not required for continued day-to-day function, although the router’s networking capabilities will still be used for local communication and coordination.

EH3. Bluetooth connectivity for WISE-WARE is handled by the host machine itself.

Details: The Raspberry Pi has built-in Bluetooth functionality that WISE-WARE can leverage for compatible smart home devices. This will be used for handling communication with assistive technologies that communicate through Bluetooth or a compatible protocol (such as Bluetooth Low Energy)

EH4. For integration with Zigbee-based smart home devices, an additional Zigbee controller component is required.

Details: Zigbee is a low-power wireless communication technology commonly used in smart home devices. If users want to integrate Zigbee devices, they will need a separate Zigbee controller component, connected to the Raspberry Pi via USB.

Reference: For more information about Zigbee technologies and specifications, refer to: https://zigbeealliance.org/

EH5. For installation, advanced configuration, or troubleshooting a device with a terminal application and SSH capability is required to connect to the Raspberry Pi hosting WISE-WARE.

Details: A computer with a terminal program (like Command Prompt on Windows or Terminal on Mac) and SSH capability can be used for advanced tasks on the Raspberry Pi. This is typically used for troubleshooting or configuration not accessible through the user interface. The control device must be on the same network as the Raspberry Pi.

Reference: For an introduction to SSH, what it is and how it can be used, refer to:
https://www.hostinger.co.uk/tutorials/ssh-tutorial-how-does-ssh-work

EH6. (See EU1) WISE-WARE's services are accessible through a modern web browser on various user devices, therefore an end-user's access or control device for WISE-WARE must support modern web technologies.

Details: WISE-WARE leverages web technologies to deliver its user interfaces. Users can access and control their smart home environment through a web browser on various devices like computers, tablets, and smartphones. The web browser should support modern web standards like HTML5, CSS3, and ECMAScript 2018 or later. These technologies ensure a smooth and visually appealing user experience. The user device's web browser should be released within the last 2 years to benefit from the latest security updates and compatibility features.

Reference: For introductions to HTML, CSS and JavaScript (ECMAScript), refer to:

- https://developer.mozilla.org/en-US/docs/Web/HTML
- https://developer.mozilla.org/en-US/docs/Web/CSS
- https://developer.mozilla.org/en-US/docs/Web/JavaScript

ES - Software Interface Requirements

ES1. WISE-WARE shall use a containerisation platform to isolate and manage individual software components.

Details: Docker is used as the containerisation platform. Docker containers provide a lightweight and isolated environment for each software component, ensuring smooth operation in the event of individual component failure, as well as making WISE-WARE modular and reproducible.

Reference: For more information on what Docker is and how it is commonly used, refer to: https://www.docker.com/

ES2. WISE-WARE shall use a core home automation platform to manage connected smart home devices.

Details: Home Assistant acts as the central hub for device integration, automation creation, and communication with various smart home devices and services. Due to the large library of community-source integrations, automations and documentation, integration with and extension of the initial WISE-WARE configuration is easy and effective.

Reference: For more information about Home Assistant, the principal home automation technology in WISE-WARE, refer to: https://www.home-assistant.io/

ES3. WISE-WARE shall use a data streaming platform to collect and store data from integrated smart home devices.

Details: Apache Kafka facilitates real-time data collection and storage from various sensors and devices, and is extensively used in other data-heavy contexts (streaming platforms like Netflix, for example). Many extant modules and connectors exist that allow Apache Kafka to be integrated with other workflows and products. The data aggregated can be used for further analysis or automation purposes, both in and outside of the home. Apache Kafka is supported by another Apache software, Apache Zookeeper.

Reference: For an introduction to the concepts important to understanding Apache Kafka, refer to: https://docs.confluent.io/kafka/introduction.html

ES4. WISE-WARE shall provide a user-friendly interface for creating custom automations without extensive coding knowledge.

Details: WISE-WARE includes Node-RED, which offers a visual programming environment, allowing users to create automations by connecting nodes that represent specific actions or triggers. This is integrated with Home Assistant, so can access the states and services involving integrated smart home technologies easily. The extensive collection of “flows” (the term for automations created in Node-RED) produced by the community make learning and adopting Node-RED easier, either for the user or a delegate that configures their automations for them.

Reference: For more information about Node-RED and how it is used, refer to: https://nodered.org/about/.

CI - Communication Interface Requirements

CI1. WISE-WARE shall use web-based communication interfaces accessible through modern web browsers on various devices.

Details: Users will interact with WISE-WARE through a web browser on devices like computers, tablets, and smartphones. The web interface shall communicate with WISE-WARE's core functionalities using standard web protocols (HTTP/HTTPS). This allows users to access and utilise the services in WISE-WARE without needing to write interfacing software themselves.

CI2. WISE-WARE shall expose RESTful APIs for communication between the user interface and integrated services.

Details: RESTful APIs shall be used to facilitate seamless communication between the user interface and integrated services like Home Assistant and Node-RED. These APIs shall provide a standardised way to exchange data and perform operations, enabling the user interface to interact with the underlying smart home automation and control features. This also allows developers to extend the interfacing functionality of WISE-WARE without compromising the existing web-interfaces (see CI1, above).

CI3. WISE-WARE shall support communication with smart home devices using industry-standard wireless protocols.

Details: WISE-WARE shall communicate with smart home devices through protocols like Bluetooth Low Energy (BLE), Wi-Fi, and Zigbee. These protocols are widely adopted in the smart home industry and shall allow efficient communication with a range of devices.

CI4. WISE-WARE shall utilise lightweight data formats for efficient data exchange between components and smart devices.

Details: Data exchange between WISE-WARE components and smart devices shall utilise lightweight data formats like JSON (JavaScript Object Notation) and specific formats used by integrated services like Home Assistant. JSON is a human-readable and easy-to-parse data format that facilitates efficient data transfer and integration with web-based systems.

CI5. WISE-WARE shall produce data in a lightweight JSON format and transmit it to Apache Kafka for storage and analysis.

Details: To improve compatibility and adhere to industry-standard data structures, WISE-WARE shall produce data, collected from integrated smart home devices and sensors, to the Apache Kafka event streaming platform for durable storage and subsequent analysis or integration with other data processing pipelines. This shall be done in a lightweight JSON format.

NF - Non-Functional Requirements

NF1. Uptime Requirement (95%)

WISE-WARE shall achieve a minimum uptime of 95% since it performs an assistive role and does not handle health-critical functions. Based on our research, this is a common, standard uptime for smart home and assistive technology of this type.

NF2. Recovery Time Objective (RTO)

NF2.1. Individual container failure

In cases where manual intervention is not required during system failure, WISE-WARE shall be able to restart and reconnect its component containers within 3-5 minutes. This is based on performance benchmarks for the services used, and varies between services (and dependencies that are also caused to restart).

NF2.2. Full system failure

In cases where manual intervention is required to completely shut down and restart WISE-WARE after a system failure, the system shall be able to restart and reconnect within 10 minutes; this is a combination of the typical startup times for the host machine used in stock WISE-WARE, and that of it’s contained services (see NF2.1).

Ongoing causes of failure may require additional time for diagnosis and repair, depending on the nature and severity of the issue.

NF3. Recovery Point Objective (RPO)

In the event of storage hardware failure, WISE-WARE may lose historical data that has not been backed up (e.g., due to lack of user permission). This data loss shall not impede everyday functioning unless a specific application of WISE-WARE is extended to recycle and make use of historical data, such as configuring or tuning automations based on probabilities calculated from historical data.

NF4. Redundancy and Resilience

NF4.1. Container isolation

WISE-WARE's containerised architecture shall ensure that individual service failures typically do not impact other services. Each container shall be configured to automatically restart in the event of failure, unless manually stopped.

NF4.2. Exceptions to 4.1

Exceptions to NF4.1 include services with dependencies, such as:

- Apache Kafka's dependency on Apache Zookeeper, where the failure of the latter shall cause the former to fail and require a restart.
- Home Assistant's production to Apache Kafka depends on Kafka's function, so in the event of Kafka's failure, Home Assistant's data production feature shall cease until Kafka is restarted.

NF4.3. Extensions of NF4.2

Additional exceptions to NFR4.1 may be introduced as WISE-WARE is further extended and shall be logically deduced during development. Most extensions leading to these exceptions will be of the dependency type discussed in NF4.2.

NF5. Performance

NF5.1. Maximum container capacity

On a Raspberry Pi 4B, WISE-WARE shall be capable of running approximately 4-6 different containerized services simultaneously without performance issues at user-scale, in addition to the 5 included in stock WISE-WARE. The specific number is highly dependent on the performance demands of the additional services, and those of the included dependencies.

NF5.2. Power consumption

WISE-WARE’s power consumption shall be low; at rest/during routine operation, below 3W, and under temporary, heavy load below 10W.

For reference, the power consumptions of the Raspberry PI models 4B and 5 are:
- RPi 4B - ~2.7W at rest, ~6.4W under heavy load
- RPi 5 - ~2.7W at rest, ~8.9W under heavy load

NF6. Vertical Scalability

NF6.1. Hardware performance

WISE-WARE's performance on a single host can be improved by installing an M.2 drive for increased and faster storage. This is possible on a Raspberry PI 4B, and alternatives (storage extensions) are available on other platforms.

NF6.2. Service additions

WISE-WARE’s abilities on a single host can be extended by installing additional containerised services. These services should be installed by alteration to the Docker Compose file template, and their dependency relationships with other services should be considered, tested and documented.

NF6.3. Assistive technology additions

WISE-WARE's abilities on a single host can be extended by installing additional integrations in Home Assistant to communicate with or coordinate new assistive technologies.

NF6.4. Peripheral devices

Peripheral modules for access to new communication protocols can be installed and integrated with WISE-WARE. These may also require additional changes to configuration files or similar to install and fully integrate.

On a Raspberry PI 4B, this is typically done through either USB-connected transceivers, or GPIO modules. Alternative host machines should have their own slots, ports or connectors to accommodate the peripherals needed for a given application.

NF7. Security

NF7.1. User data is not transmitted without consent

User data must stay on-site, in Kafka, outside of specific situations previously consented to by the user which have not had that consent revoked since.

NF7.2. User data transmission is not required for function

In cases where consent for external data transmission is not provided, or is withdrawn, WISE-WARE’s user-side functionality will not be affected.

NF7.3. Exceptions to 7.2

Exceptions to NF7.2 are extended services, reliant on the transmission of that data, which a particular implementation of WISE-WARE also provides.

NF8. Capacity

NF8.1. Container images

Approximately 4.0 GB of storage is required for the container images used by the core WISE-WARE system.

NF8.2. Accommodating use and extension

Additional storage space will be consumed during use and with extension of WISE-WARE. To accommodate this data, at least 32GB of additional usable storage space will be included in installations of WISE-WARE, and a policy for ensuring that the filling of that space does not impede function will be included (such as deletion of data past a certain age, or selective retaining of particular data).

NF9. Compatibility

NF9.1. OS compatibility

WISE-WARE is compatible with operating systems that have a maintained Docker solution available, such as Debian or Alpine. Stock WISE-WARE uses Raspberry Pi OS Lite, in particular.

NF9.2. Hardware compatibility

WISE-WARE is compatible with hardware platforms that are able to be installed with and run operating systems that fulfil NF9.1. Stock WISE-WARE uses a Raspberry PI 4B or Raspberry PI 5.

As exemplars of the hardware requirements, excerpts from their technical specifications are included here, as well as links to their datasheets for reference:

Raspberry PI 4B	Raspberry PI 5
• Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.8GHz • 1GB, 2GB, 4GB or 8GB LPDDR4-3200 SDRAM (depending on model) • 2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE • H.265 (4kp60 decode), H264 (1080p60 decode, 1080p30 encode) • OpenGL ES 3.1, Vulkan 1.0 • Micro-SD card slot for loading operating system and data storage …	• Broadcom BCM2712 2.4GHz quad-core 64-bit Arm Cortex-A76 CPU, with cryptography extensions, 512KB per-core L2 caches and a 2MB shared L3 cache • VideoCore VII GPU, supporting OpenGL ES 3.1, Vulkan 1.2 • Dual 4Kp60 HDMI® display output with HDR support • 4Kp60 HEVC decoder • LPDDR4X-4267 SDRAM (4GB and 8GB SKUs available at launch) • Dual-band 802.11ac Wi-Fi® • Bluetooth 5.0 / Bluetooth Low Energy (BLE) • microSD card slot, with support for high-speed SDR104 mode …
https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-datasheet.pdf	https://datasheets.raspberrypi.com/rpi5/raspberry-pi-5-product-brief.pdf

NF10. Maintainability

NF10.1. Software extensibility

WISE-WARE shall support continuous integration through various mechanisms, including:

- Addition of new containers

- Exposing RESTful APIs for communication with existing containers
- Leveraging Home Assistant integrations, templates, and blueprints
- Creating custom nodes for Node-RED

NF10.2. Logging

The logs produced by containerised services shall be accessible to the end user through Portainer. These can also be used for troubleshooting by delegated maintainers of a given installation, as well as developers.

NF10.3. BI for maintenance

WISE-WARE shall support integration with business intelligence (BI) solutions like PowerBI and Grafana to enable monitoring of the system's performance, in addition to wellbeing/assistance -related analysis tasks.

NF11. Usability

NF11.1. Configuration usability

The main WISE-WARE software system shall be easy to use for configuration-users. This subgroup of end-users are assumed to either have some amount of comfort with technical systems, or have delegated configuration to someone who does.

The learning curve for services varies, with some being more difficult to configure and extend than others. For reference, the difficulties of use and configuration of the services in stock WISE-WARE are:

- Home Assistant: Easy to use but moderately more difficult to configure and extend.
- Node-RED: Easy to use and visually intuitive for creating automations, preferable to Home Assistant's default automation and scripting system.
- Portainer: Provides a user-friendly GUI for container management, compared to the default CLI-based controls of Docker.
- Apache Kafka: Not user-friendly but not intended for direct user interaction. Kafka Connect can be used to ease integration in certain installations.

NF11.2. Intelligent environment usability

The assistive, intelligent environment enabled by WISE-WARE shall be easy to use and live in for end-users. These end-users are assumed to have minimal knowledge or comfort with configuring technical systems, and do not need technical knowledge to enjoy the benefits of the intelligent environment.

NF12. Documentation

WISE-WARE shall be accompanied by comprehensive documentation, including user manuals and developer guides, targeting different audience groups as appropriate.