Thermal Grace - VHUB
Comfort-as-a-Service IoT System

Software Architecture Design

Artur Kraskov
Semester 7
ICT & OL, Delta
Fontys 2025-2026

Contents

Introduction

Within the scope of VHUB Comfort-as-a-Service system development a prototype was realized. The current document reports the software architecture design.

Background

Software design is made to satisfy requirements and fit in the context of user stories. Previously the project plan and the requirements document were compiled [1-2]. Hardware architecture was designed [3].

Software Architecture

Requirements summary

The system must be able to accumulate data from various wireless sensors. It must collect user feedback. Weather data must be fetched via an API. It must compute PMV/PPD values and query LLM for analysis and advice. New sensors or access points must be easily integrable and removable - the algorithm must still be capable of providing an accurate analysis, sensing and predicting perceived thermal comfort.

Stakeholders and Needs

Stakeholder group A — Researchers

- **Needs:** validity/traceability of measurements, reproducible computations, export/logging, interpretability, modularity, adaptability.
- **Concerns:** data quality, model transparency, auditability.

Stakeholder group B — Office users/visitors/workers

- **Needs:** easy feedback input, understandable comfort guidance, minimal friction.
- **Concerns:** privacy, data use, clarity, trust.

C4 levels 1 & 2

Level 1 (Context): system + external dependencies + people.
Level 2 (Container): main runtime pieces (Streamlit app, MQTT broker, sensor nodes, external APIs, CSV storage, survey app).
Note: Component-level is intentionally abstract/omitted because the system is evolving.

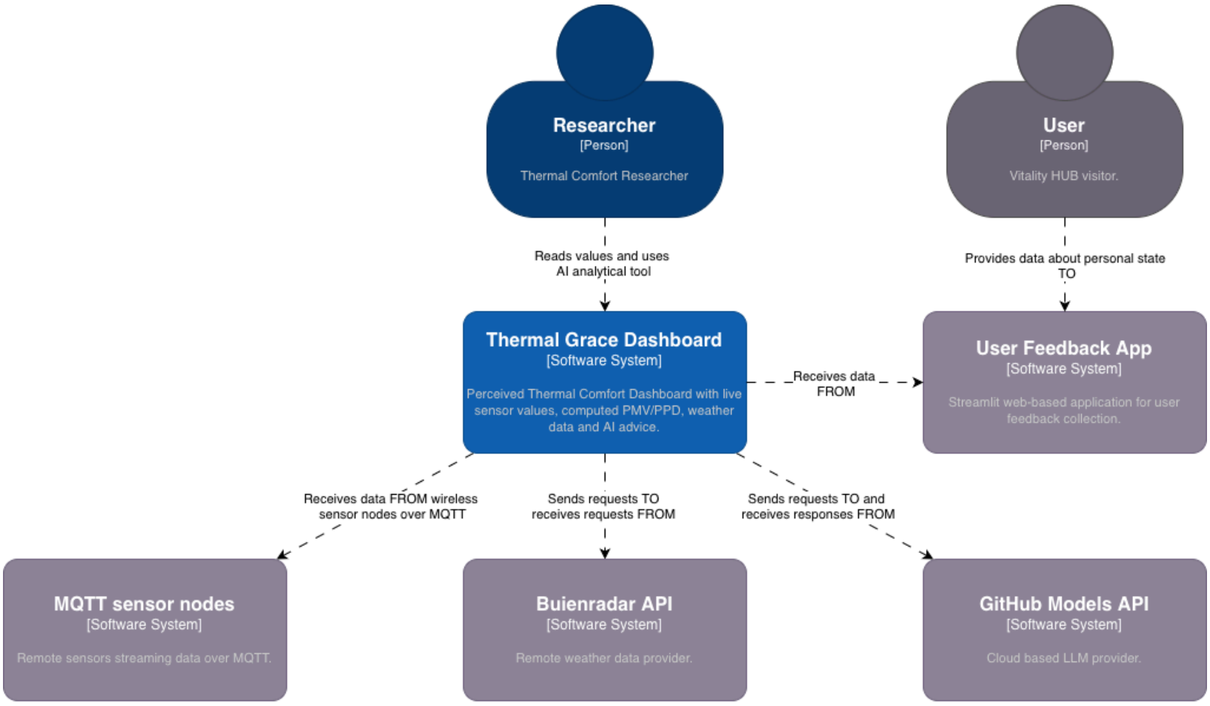

Image 1: C4 diagram level 1 - context
High resolution: https://drive.google.com/file/d/1dxyRCSqdI8iYIdqm1-Kn6hhx_9YyO0t3

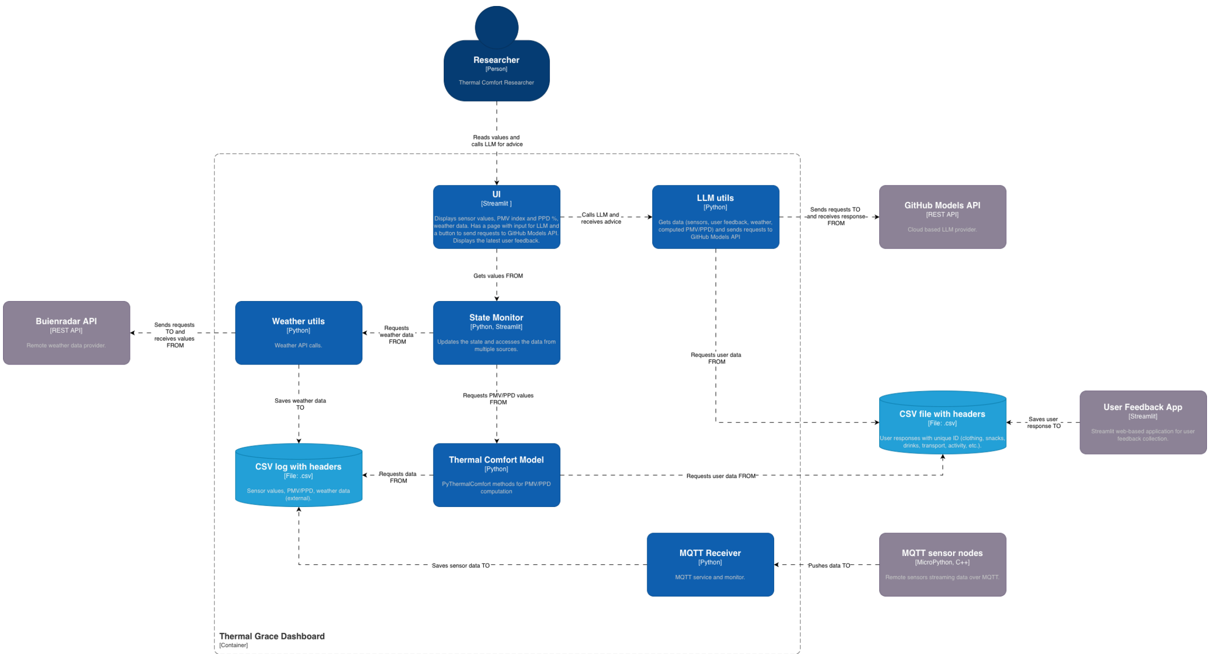

Image 2: C4 diagram level 2 - container
High resolution: https://drive.google.com/file/d/1_d4d4YvSziXnISVNDAA3kGB_XrYcdbUx

The current implementation meets the requirements in a way that it allows connecting more API or data points by adding new utils. New sensors can be added and connected to the MQTT receiver, which can accept new topics. Thermal Comfort index and % computation can be explored and researched via PyThermalComfort module. All the data is logged into .csv files, which can further be utilized for data analytics and AI models training. Users provide the feedback on their current feeling of comfort and share data about their travel, snack, clothes and other parameters important for PMV/PPD computation. Each entry has a UID, no name or sensitive info is stored or asked for.

System Inventory: Existing vs New/Custom-Built

Existing systems/services

- **MQTT Broker (Mosquitto / managed broker):** message transport for sensor data.
- **Buienradar API:** external weather provider (REST API, HTTP).
- **GitHub Models API:** external LLM provider (REST API, HTTP).
- **PyThermalComfort - Thermal comfort computation module:** PMV/PPD logic.

New/custom-built systems in this project

- **Thermal Grace Dashboard (Streamlit app):** UI + background services + computations.
- **MQTT ingest logic:** topic parsing, validation, snapshot update.
- **User Feedback App:** Streamlit survey app + response storage (CSV).
- **Sensor node firmware/scripts:** Pico publishers for CO_2 and air/mmWave (if considered part of the overall system).

Quality Attributes

Security

Threats to address

- Unauthorized access to MQTT topics (spoofed sensor data).
- Leakage of API tokens (GitHub Models / weather).
- Exposure of user feedback (personal/contextual data).
- Dashboard endpoint exposure on the network.

Controls / decisions to document

MQTT security

- Broker authentication (username/password or client certs).
- Topic ACLs (publish-only for sensors; subscribe-only for dashboard).
- TLS for MQTT (if feasible on the network).

API credential handling

- Store tokens in environment variables / secret store (not in repo).
- Rotation strategy; least-privilege tokens.

Network boundaries

- Dashboard access limited to trusted network/VPN; optional reverse proxy with auth.
- Firewall rules: only required ports open (MQTT, Streamlit).

Data protection & privacy

- Minimize Personally Identifiable Information (PII) in feedback; clarify data retention.
- Access control to CSV files; file permissions; backups.

Logging and auditability

- Do not log secrets; sanitize payload logs.
- Record message provenance (topic, timestamp) for traceability.

Secure defaults

- Disable DEBUG logs in production (or redact).
- Input validation for MQTT payloads and feedback fields.

Scalability

Expected growth vectors

- More sensor nodes / more topics.
- Higher message frequency.
- More concurrent dashboard viewers.
- Longer retention / larger CSV files.

Scalability decisions

MQTT extensibility

- New sensors added by publishing to new topics; dashboard subscribes via patterns and routing rules.

Storage evolution path

- CSV as initial store; migration plan to SQLite/Postgres/InfluxDB if file size becomes a bottleneck.
- Partitioning/rolling files (daily/weekly CSV) to keep reads fast.

Performance safeguards

- Rate-limit LLM calls; cache recent advice; handle API rate limits/timeouts.
- Avoid full-file CSV reads on every UI refresh (use tail/last N rows).

Deployment scaling

- Single instance initially; scale-out path (multiple dashboard instances) requires shared storage + shared state (DB + broker remains central).

Architecture Constraints and Assumptions

- Runs on a single host (initially) with local CSV files.
- External APIs depend on internet availability.
- Sensors rely on Wi-Fi stability and broker uptime.
- Comfort model assumptions - some values are fixed and hardcoded.

Technology Choices and Rationale

- Why MQTT (pub/sub, decoupling, easy sensor onboarding).
- Why Streamlit (fast iteration, research prototype).
- Why CSV (transparent, easy export) + acknowledged limits.
- Why external LLM API (capability vs hosting cost/complexity).

Conclusion

The described software architecture design represents a Minimal Viable Product (MVP) for Vitality HUB's Perceived Thermal Comfort IoT system. It is reproducible made with supported libraries and frameworks. Implemented on accessible hardware. It will be updated after the realization of the final version of the prototype. Level 3 - Components will be added into the C4 model.

Reference

1. Kraskov A., Kaszuba B., (2025), Vitality HUB Perceived Thermal Comfort | Project Plan, Thermal Comfort-as-a-Service (CaaS) IoT system, FHICT Delta, [Online], Available: https://drive.google.com/file/d/1BXHXID4_SPatpJrU1mDwF0ZKnESI2W_U
2. Kraskov A., Kaszuba B., (2025), Vitality HUB Perceived Thermal Comfort | Requirements, Thermal Comfort-as-a-Service (CaaS) IoT system, FHICT Delta, [Online], Available: https://drive.google.com/file/d/1NooyuNM97BFz3jYOX2aPSRkbFZ7f5yJF
3. Kraskov A., (2025), Thermal Grace - VHUB Comfort-as-a-Service IoT System, Hardware Architecture Design, FHICT Delta, [Online], Available: https://drive.google.com/file/d/1sWcylxS85DdrUbDKQ6F5CqS8aSHxkFRN