Bachelor's thesis

Information and Communications Technology

2025

Sebastian Sopola

Smart Home Automation

TURKU AMK
TURKU UNIVERSITY OF
APPLIED SCIENCES

Bachelor’s Thesis | Abstract

Turku University of Applied Sciences

Information and Communications Technology

2025 | 45

Sebastian Sopola

Smart Home Automation

In the modern world, technology enables people to transform their homes into smart homes through automation. The primary objective of this thesis was to develop a smart home automation system that extends the existing capabilities of homes and introduces new features. The goal was to create a useful and user-friendly solution that automates basic household functions.

The system aimed to offer features such as automatic heating control, real-time home status information, and remote control. The system's performance was evaluated based on criteria such as user-friendliness, economic savings, and stability. The results showed that the smart home system can enhance the basic functions of homes, provide remote control capabilities, increase knowledge of home functions, and bring cost savings, highlighting the real benefits of the system.

The thesis demonstrates the potential of smart home automation in modern homes. The experiences and data gained from this work provide valuable information and expertise for the further development of the system.

Keywords:

Smart home, home management, monitoring, automation, savings, user-friendliness, IoT

Opinnäytetyö (AMK) | Tiivistelmä

Turun ammattikorkeakoulu

Tieto- ja viestintätekniikka

2025 | 45

Sebastian Sopola

Älykoti

Nykyaikana teknologia mahdollistaa sen, että ihmiset voivat muuttaa kotinsa älykodeiksi automaation avulla. Tässä opinnäytetyössä kehitettiin älykotien automaatiojärjestelmä, joka laajensi kotien nykyisiä ominaisuuksia ja toi uusia toimintoja. Tavoitteena oli luoda yhtenäinen ja käyttäjäystävällinen ratkaisu, joka automatisoi kodin perustoiminnot.

Opinnäytetyössä suunniteltiin järjestelmä, joka tarjoaa ominaisuuksia kuten automaattisen lämmityksen säätelyn, reaaliaikaisen kodin tilannetiedon ja etäohjauksen. Järjestelmän suorituskykyä arvioitiin käyttäjäystävällisyyden, taloudellisten säästöjen ja vakauden perusteella. Tulokset osoittivat, että älykotijärjestelmä voi parantaa kotien perustoimintoja, tarjota etäohjausmahdollisuuksia ja lisätä tietoa kodin toiminnoista sekä tuoda kustannussäästöjä, mikä korostaa järjestelmän etuja.

Opinnäytetyö osoittaa älykotien automaation potentiaalin nykyaikaisissa kodeissa. Opinnäytetyöstä saadut kokemukset ja data tarjoavat arvokasta tietoa ja osaamista järjestelmän jatkokehitystä varten.

Asiasanat:

Älykoti, etäohjaus, seuranta, automaatio, säästöt, käyttäjäystävällisyys, IoT

Contents

Figures

Pictures

List of abbreviations

Adax API	An API service provided by Adax company to connect to their devices and interact with them
API	A set of rules or protocols that allow different software applications to communicate and interact with each other
Bluetooth	A wireless technology standard for exchanging data over short distances
H100	A specific hub model used in smart home systems
HVAC	Heating, ventilation and air conditioning system
IDPS	Intrusion detection and prevention system used in communication systems
IoT	A network of interconnected devices that collect, exchange, and act on data to create smarter environments
LLM	A type of machine learning model designed for natural language processing tasks such as language generation
PC	A computer
TP-Link	A Chinese company that manufactures network equipment and smart home products
WPA2	An encrypted security protocol that protects internet traffic on wireless networks
Wi-Fi	A family of wireless network protocols which support local area networking of devices and Internet access
Zigbee	A low-cost, low-power, wireless mesh network standard that allows different devices to communicate together

Z-Wave	A wireless communications protocol user primarily for home automation

1 Introduction

In the modern world, technology empowers individuals to transform their homes into smart homes through automation. Smart home automation offers numerous benefits, including improved cost efficiency, energy efficiency, insights into household functions, automating tasks, and enhancing comfort. By leveraging smart home systems, users can reduce utility bills and create a more convenient living environment. [1]

However, smart home systems face several challenges. Multiple brands and manufacturers produce devices with little to no cross compatibility, leading to fragmented systems. The development of a new IoT technology called Matter aims to bridge this gap, providing cross-platform and cross-manufacturer compatibility. [2]

This thesis is part of studies in the Embedded Systems study track at Turku University of Applied Sciences. The goal is to develop a smart home automation system that is both useful and user-friendly. The system is designed to extend the capabilities of existing home functions and introduce new features that enhance household management.

The key components of the system include a server, platform, hub, smart thermostats, sensors, and a mobile app. These components work together to offer a variety of functionalities, including physical heating controls, remote heating controls, automatic heating controls, real-time data monitoring, and access to historical data.

The system's performance will be evaluated based on the following criteria: user-friendliness, energy cost savings, and overall stability. The results are expected to demonstrate that the smart home system can enhance basic household functions, provide remote control capabilities, increase knowledge of home functions, and bring cost savings. These outcomes will highlight the real benefits of the smart home system.

By addressing the limitations of current systems and offering new functionalities, this thesis aims to showcases the potential of smart home automation in modern living environments. The insights gained from this thesis will contribute valuable information and personal experience to the ongoing development of smart home technologies.

2 Requirements

In this chapter, the requirements of the smart home automation system are described to ensure proper system functionality. To ensure the system is a reasonable choice, both hardware and software costs **must** be kept reasonable. Additionally, the system **should** have low maintenance requirements for both hardware and software. The platform **should** support the hardware to some extent to ease the design and implementation phase without sacrificing customizability.

The server **must** be able to process data from various sensors, execute scripts, support wireless network connectivity, and be able to store data. To make the smart home automation system a reasonable choice, the server hardware **must** be affordable, reliable, small, and energy efficient.

The platform **must** provide interfaces, services, and functionalities needed to integrate and manage IoT devices. The platform **must** handle data exchange, device coordination, and automation logic. Overall development environment to create custom system with user interface functionality.

The hub **must** serve as the communication bridge between the server and IoT devices. The hub **must** support at least one of the communication protocols such as ZigBee, Z-Wave, or Wi-Fi. The need for a hub is not mandatory, but the work span of the sensors and the extra work by acquiring a hub and possible third-party software that comes with the hub must be weighed.

The smart thermostat **must** support remote control, automation system compatibility, and **must** support receiving commands to perform actions via at least one communication protocol (ZigBee, Z-Wave, Wi-Fi, Bluetooth). It **should** also function as a traditional thermostat when necessary.

Sensors **must** have a reasonable work span of approximately 365 days, support compatibility with the smart home automation system, and operate within a temperature range of +30 to -30 degrees Celsius. Additionally, sensors

should support precision in measuring, robustness to withstand various environmental conditions, and efficient energy usage to extend battery life.

The mobile app **must** provide a customizable interface for interacting with the smart home automation system. The mobile app **must** allow users to remotely monitor and control their home devices. The mobile app **should** support notifications to users for important events and updates.

The system **must** be designed to collect, process, and act on real-time data provided by sensors. Data **must** be stored in a database after processing. Stored data **should** be accessible via a web application for display.

The system **must** allow manual adjustment of heating settings directly on the thermostat. The physical interface **must** provide immediate control over the home's heating units.

The system **must** enable users to adjust heating settings via a mobile app or web interface. It **must** support remote monitoring and control of the heating system. There **should** be customizable areas and central heating control options.

The system **must** automate the adjustment of heating settings based on predefined criteria (e.g., time of day, temperature preferences, room size). It **must** optimize energy costs as a primary rule and sustain a comfortable living environment based on heating cost optimization processes and comfort criteria. The system **should** provide both central and room-specific automatic heating control.

The system must:

- Include temperature and humidity sensors.
- Enable responsive and accurate system operations with real-time data.
- Store all data, including logs of actions, adjustments and changes.
- Provide historical data for analysis.

3 Design

In this chapter, the key components and key functionalities of the smart home automation system are outlined and described. Each component and functionality play a critical role in ensuring the system's overall operation and purpose.

3.1 Key components

Figure 1 below shows the system design, showcasing each element along with the direction of the data flow. The diagram includes communication protocols and the relationships between components, providing a comprehensive overview of the system architecture.

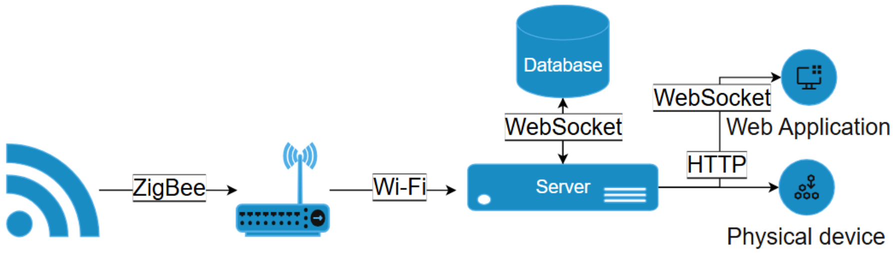


Figure 1. System design.

The server acts as the central unit that runs the smart home automation system. It processes data from various sensors, executes scripts, and stores data. [3]

The platform is the software framework that powers the smart home automation system. It provides the interfaces, services, and functionalities needed to integrate and manage IoT devices. The platform handles data exchange, device coordination, and automation logic. [4]

A hub serves as the communication bridge between the server and IoT devices. It can support multiple or one of the many communication protocols like ZigBee, Z-Wave, and Wi-Fi. A hub ensures that various smart devices can communicate effectively with a server. [5]

A smart thermostat controls the heating and cooling of the home. Smart thermostats can receive commands to perform actions via many communications protocols such as ZigBee, Z-Wave, Wi-Fi, and Bluetooth. Alternatively, it can act as a “dumb” thermostat, functioning as traditional thermostat as well. [6]

A sensor collects real-time data about the environment, such as temperature, humidity, motion, or light. This data is sent to the server for processing and decision-making, making the system responsive and accurate. [7]

The mobile app provides a user-friendly interface for interacting with the smart home automation system. It allows users to remotely monitor and control their home devices and receive notifications. The app ensures easy access and management of the smart home from anywhere. [8]

3.2 Key functionalities

Physical heating control enables manual adjustment of heating settings directly on the thermostat. This functionality allows users to interact with the physical interface of the thermostat to set the desired temperature and switch between different modes (e.g., heating, cooling, off). The physical interface provides users with immediate control over their home’s heating units, ensuring functionality even during system downtimes.

Remote heating control enables users to adjust heating settings via a mobile app or web interface, regardless of their location. This functionality allows users to monitor and control their home’s heating system remotely. Users can set temperatures, receive alerts, and be prompted to act. There are two types of thermostats in this system: room-specific thermostats, which control the heating for individual rooms, and central thermostat, which manages heating for the entire house. [9]

Automatic heating control automates the adjustment of heating settings based on predefined criteria such as time of day, temperature preferences, and room size. This functionality optimizes energy costs and maintains a comfortable living environment without requiring manual intervention, except in extreme situations where remote intervention is possible. Automatic heating controls are divided into central and room-specific categories, allowing for both central and room-specific heating control. [9]

Data functionality provides both real-time and historic data features. Real-time enables responsive and accurate system operations, while historical data provides valuable insights over time. These data types contribute to various system functionalities, ensuring more responsive, accurate and informed decision-making. [10]

4 Implementation

In this chapter, the development process of the smart home automation system, including the development, implementation solutions, encountered obstacles, and observations made throughout the development process is detailed.

4.1 System topology

The smart home automation system is implemented to collect, process, and act on real-time data provided by sensors. Automation functionality is capable of fast and accurate responses because of the real-time data stream. Furthermore, data is stored in a database after processing. This stored data can then be requested by a web application for display and further interaction. [11]

Figure 2 below shows the system topology which presents the physical layout of all devices, including the communication medium (e.g., wireless network).

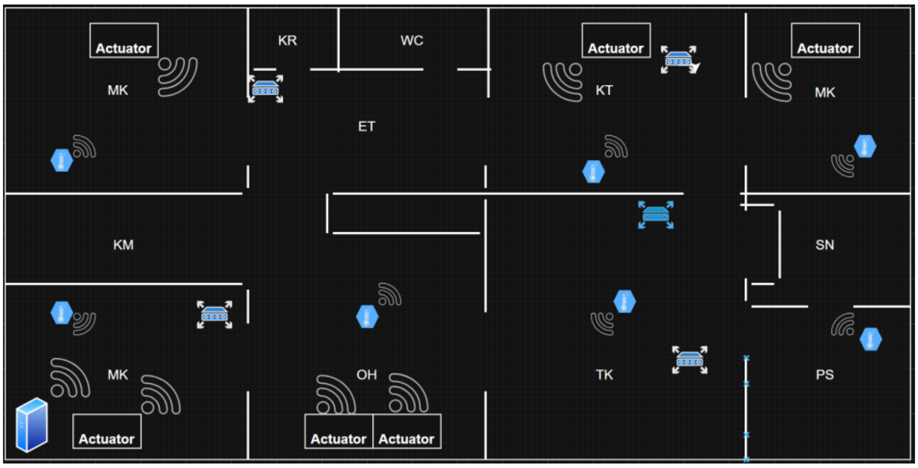
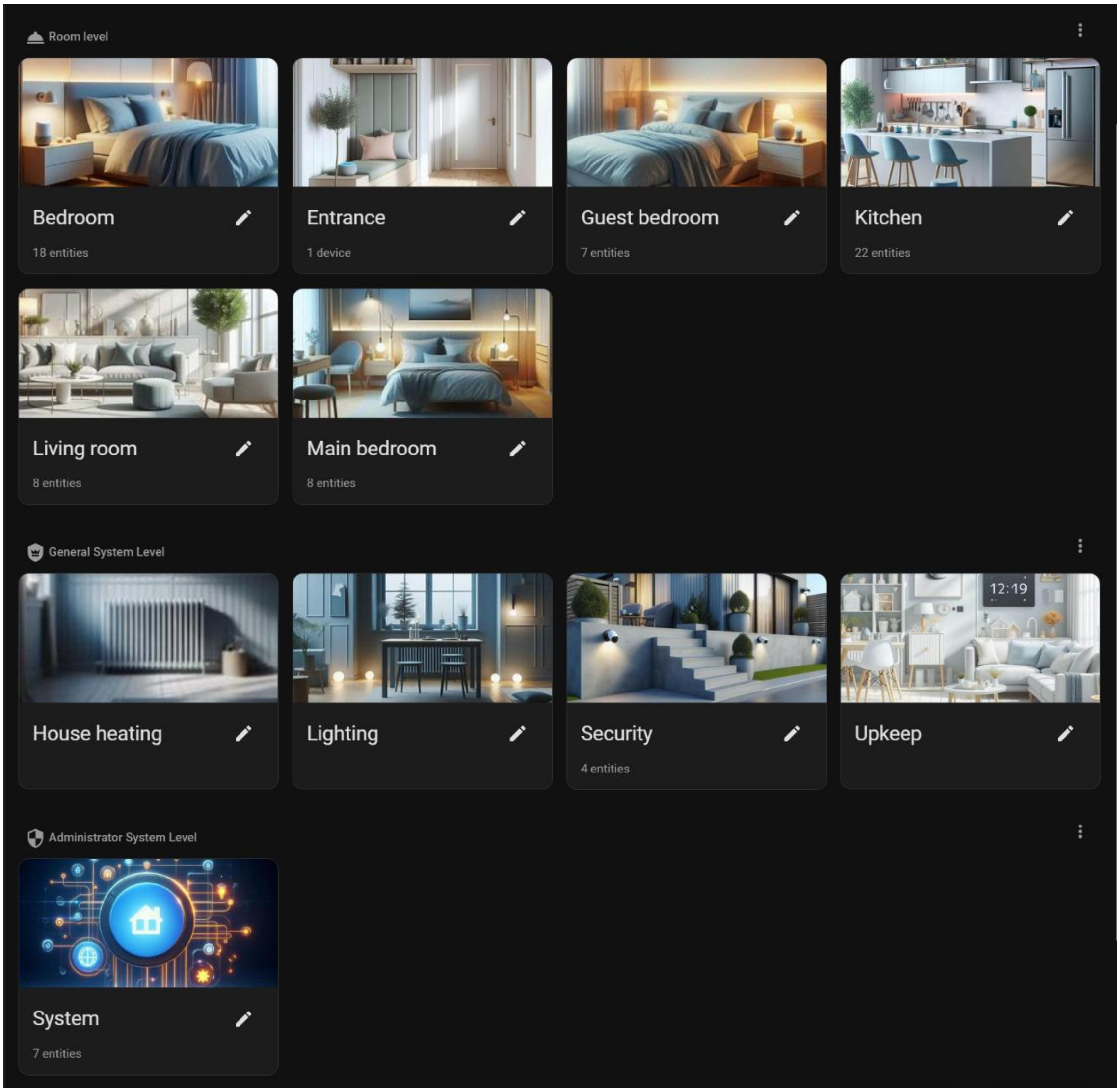


Figure 2: System topology.

The implementation is both area-based and functionality-based, meaning that some functions control specific rooms independently, without interference from system controls. Meanwhile, other functions provide fine-tuning controls for specific rooms and can be affected by system configuration changes.

Picture 1 shows the different smart home automation permission configurations and access rights groups along with the members. [12]



Picture 1: Permissions and access rights.

The decision to create a system to communicate over a wireless network interface was a clear choice. It’s affordable, reliable, easy to set up, and supports the loads that a smart home automation system may provide. All communication traveling through wireless network access points is encrypted (WPA2) by the network devices. Additionally, network events are detailed in weekly security reports. Both measures improve the security of the network and home. The smart home automation system operates in a restricted network environment, which is not connected to the rest of the network. The network security implements an Intrusion Detection and Prevention System (IDPS) feature, which monitors who, when, where, and how a new connection to the smart home automation system network is made. [13][14]

Figure 3 shows the technical implementation of restricted network.

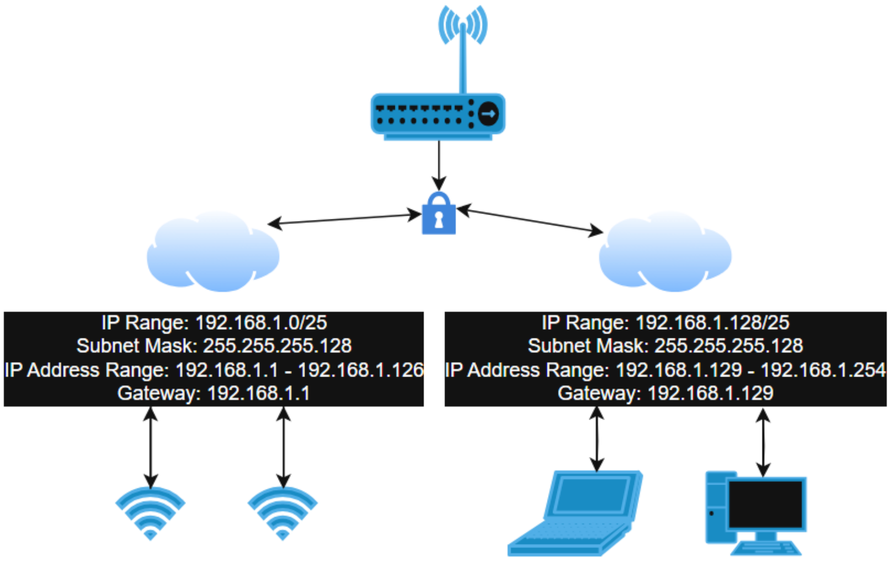


Figure 3: Restricted network.

Creating a suitable network with internet access was straightforward. The process involved researching affordable yet capable hardware with security features. While the process did not present any significant issues, it required a few hours of work. Some of this time was spent learning the network device interface and creating the necessary network configuration.[15]

Sensor data is sent by T310 sensors via the ZigBee communication protocol to a hub, which then forwards the data to the cloud. The server retrieves the sensor data via the TP-Link API by the polling method. [16][17][18]

Figure 4 shows how the sensor sends readings to a hub, which then continues data transmission to the cloud, where the server retrieves the data.

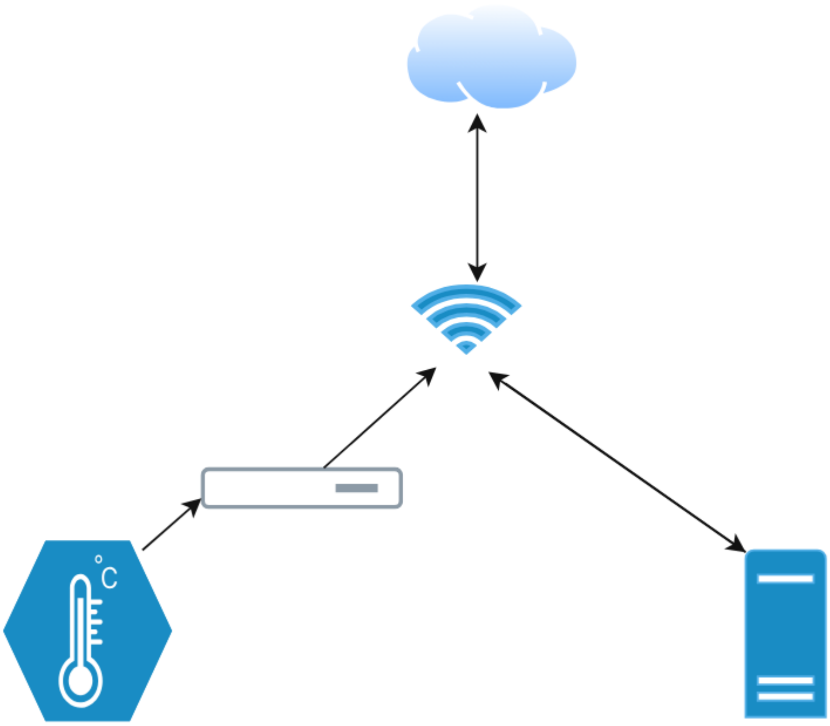


Figure 4: Sensor function.

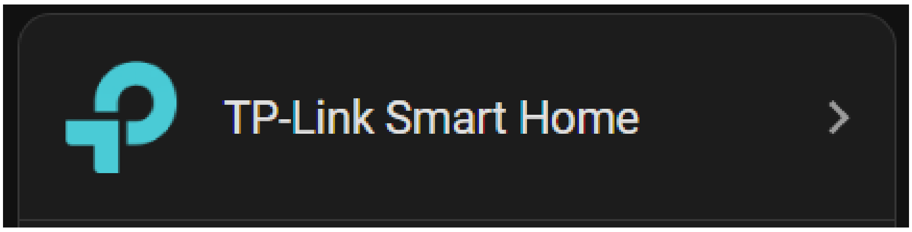
The original plan was to use the Sonoff ZigBee Pro as a hub, which offers full control over the hub device software, supports more IoT devices per hub, and does not require an additional cloud connection. This setup promised the best performance and security at an affordable price. [19]

However, T310 wouldn't connect to the Tasmota Zigbee bridge. Despite efforts to resolve the issue by examining the hub's communication and verifying that the sensors worked with the manufacturer's own hub, the problem persisted. Resolving it would have required purchasing a sniffer to analyse the ZigBee packets further. [20]

This challenge led to the switch to the H100 hub. While the H100 hub introduced an additional dependency on the TP-Link API to link the IoT devices to the smart home automation system, it was necessary to set up the system. Additionally, the sensors were cheaper when bundled with the hub. [21]

Consequently, it was decided to abandon the initial plan and use the H100 hub instead. This decision resulted in a downgrade in security and potential hardware limitations when scaling up the system, as the chosen hub had significantly lower specifications. Fortunately, setting up the TP-Link API to establish the communication channel between the sensor and server was completed quickly.

Picture 2 shows the software toolkit used to set up the API connection and integrate the sensors into the smart home automation server, including access to the TP-Link Cloud.



Picture 2: API connection.

The data provided by the sensors is received and processed by the system. It's also stored in a database after processing. After the steps the system creates function specific sensor components which provide interfaces to handle or control certain system functions. And provide information to be easily retrievable by a system function upon need.

Picture 3 shows the heating control sensor, which is created, managed and deleted using the MQTT discovery feature. The code snippet demonstrates how to use the MQTT discovery feature to create an entity. Once created, this entity can be used like any regular entity or configure via MQTT topics for more advanced actions that regular entities do not support. Although MQTT entities do not exist in the system configuration file, they can still be accessed as regular entities. [22]

```yaml
create_room_heating_interface:
  # 1. Create {room}_heating_interface_sensor
  - service: mqtt.publish
    data:
      topic: "homeassistant/sensor/{{room}}_heating_system_interface/config"
      payload: >
        {
          "name": "{{room}} Heating System Interface",
          "unique_id": "{{room}}_heating_system_interface",
          "icon": "mdi:radiator",
          "state_topic": "heating_system_interface/{{room}}_heating_system_interface/state",
          "json_attributes_topic": "heating_system_interface/{{room}}_heating_system_interface/attributes"
        }
      retain: true
  # 2. Initialise attributes field
  - service: mqtt.publish
    data:
      topic: "heating_system_interface/{{room}}_heating_system_interface/attributes"
      payload: >
        {
          "start_heating_process": "{{ states('input_boolean.' + room + '_start_heating_process') }}",
          "data_collection_process": "{{ states('input_boolean.' + room + '_data_collection_process') }}",
          "room_conditions": {
            "maximum_temperature_reached": "{{ states('input_boolean.' + room + '_at_maximum_temperature') }}",
            "minimum_temperature_or_below": "{{ states('input_boolean.' + room + '_at_or_below_minimum_temperature') }}",
            "between_min_max_temperature": "{{ states('input_boolean.' + room + '_between_minimum_and_maximum_temperature') }}"
          },
          "heating_options": {
            "preheat_maximum": "{{ states('input_boolean.' + room + '_preheat_maximum') }}",
            "maintain_heat": "{{ states('input_boolean.' + room + '_maintain_heat') }}",
            "delay_heat": "{{ states('input_boolean.' + room + '_delay_heat') }}",
            "minimum_heat": "{{ states('input_boolean.' + room + '_minimum_heat') }}"
          },
          "control_thermostats": "{{ states('input_boolean.' + room + '_control_thermostats') }}"
        }
      retain: true
  fields:
    room:
      name: Room name
      description: Provide the name of the room
      required: true
      selector:
```

Picture 3: Control sensor.

Implementing the automatic heating process proved to be the most difficult part of the work. After several iterations, it was decided to start over, this time with a detailed design plan. The entire automatic heating process was divided into three layers. As each layer was developed, new details were added. This approach proved to be the best choice because it clarified which technical solutions were viable and how everything should be interconnected. Implementing the design was relatively easy once the system architecture was established, making the technical part straightforward. This ease of implementation was largely due to thoroughly reviewing the software framework documentation and understanding its features earlier. Additionally, the platform community's support was invaluable, especially when small but significant details were overlooked in the documentation. [23]

The automatic heating process starts every hour based on predefined criteria. It gathers all data from the interface sensor to create an overview of the entire home environment and each room. The system checks that all components are functioning properly and evaluates the need for heating based on the data. The automatic heating system also retrieves additional data from the web, such as electricity prices, in a preprocessed format. The program considers three factors: heat energy sustainability, current prices, current price trend, and upcoming trends, comparing them against the results of the thermodynamics analysis. Before sending new orders to smart thermostats, the system performs final checks and updates any changes in the settings. These new orders take effect for the next hour. If there are changes during the hour, the system adapts the remaining time and restarts the process at the beginning of the next hour.

Figure 5 shows data processing and heating control.

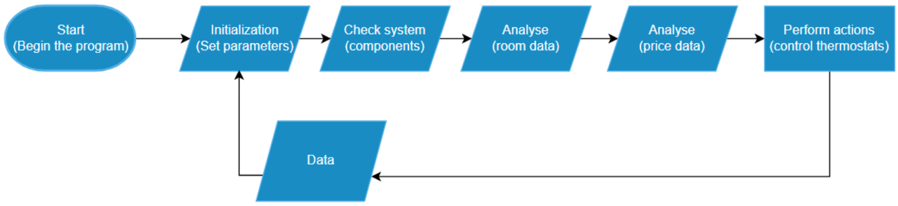
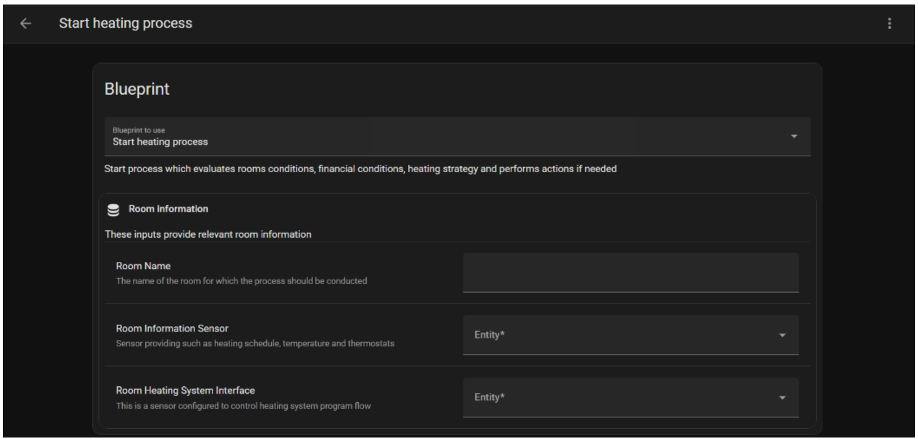


Figure 5: Data processing and heating control.

The automatic heating system is designed to allow any integrated HVAC units to be added to the overall system. Each area-based system component is configurable via a user interface. This means that new units can be set up within the existing system framework to function independently and become part of the overall system. HVAC units can be added, replaced, etc. making the system scalable, flexible, and easy to maintain for non-technical users. [24][25]

Picture 4 shows how integrated HVAC devices can be added to the automatic heating system.



Picture 4: HVAC devices.

Core system entities, such as sensors, devices, and interfaces, can be added traditionally by updating the system configuration files. These configurations cannot be changed with built-in features, but the pyscript integration provides an I/O feature for system configuration files. Note: Do not use this feature unless familiar with how Home Assistant handles processes. Incorrect usage can break the system or interfere with its base functions. [26][27]

Picture 5 shows how to configure core system entities which should be considered immutable.

```
- platform: template
  sensors:
    kitchen_room_information:
      friendly_name: "Kitchen Room Information Sensor"
      value_template: "{{ states('sensor.kitchen_temperature_and_humidity_sensor_temperature') }}"
      attribute_templates:
        room_temperature: "{{ states('sensor.kitchen_temperature_and_humidity_sensor_temperature') }}"
        room_minimum_temperature: "{{ states('input_number.kitchen_minimum_temperature') }}"
        room_maximum_temperature: "{{ states('input_number.kitchen_maximum_temperature') }}"
        room_target_temperature: "{{ states('input_number.kitchen_target_temperature') }}"
        room_living_space: "{{ states('input_number.kitchen_living_space') }}"
        room_heating_schedule_start_time: "{{ states('input_datetime.kitchen_heating_schedule_start_time') }}"
        room_heating_schedule_end_time: "{{ states('input_datetime.kitchen_heating_schedule_end_time') }}"
        room_thermostat: "climate.keittio"
```

Picture 5: System configuration file.

The smart home automation system implements a feature called Smart sensor monitoring and alert system. This feature keeps track of sensor behavior, status, and other values. It includes a custom-built alert system that notifies predefined user via mobile application in the event of anomalies, such as IoT devices going offline for unknown reasons or having low battery levels. Additionally, this feature is configurable via a user interface, making it easy to add new sensors, such as water leak sensors. [28]

Picture 6 shows how smart sensor monitoring and alert system functions.

```yaml
- id: Bathroom_Toilet_sensor_Unavailable
  alias: Bathroom Toilet Sensor Unavailable
  trigger:
    - platform: state
      entity_id: binary_sensor.vesivuotoanturi_wc_pontto_moisture
      to: 'unavailable'
      for:
        minutes: 5
  action:
    - service: notify.mobile_app_sebastians_iphone
      data:
        title: "Anturi Epäkunnossa"
        message: >
          Vessan WC Pytyn vesivuotoanturi on ollut 'unavailable' tilassa viimeiset 5 minuuttia. Tarkasta laitteen yhteys.
```

Picture 6: Monitoring and alerts.

The smart home automation system also includes a feature called Unified communication broadcast. This feature provides one-way communication between the server and all users. It is designed to send information, alerts, etc. to all users simultaneously. The underlying functionality is compatible with assistant and other LLM models that support function calling features. The trick is to make the Unified communication broadcast feature detectable by AI assistants, enabling AI assistants to use it. AI assistants can send custom messages based on text or audio input without using the custom-built interface. An AI assistant integration into the Unified communication broadcast is compatible with catchphrases as well for further improvement. [29][30]

Picture 7 shows how messages are displayed to the user.

```yaml
- id: send_raw_weather_data_to_extended_openai
  alias: Send Raw Weather Data To Extended OpenAI integration for data processing
  trigger:
    - platform: time
      at: "08:00:00"
  action:
    - service: conversation.process
      metadata: {}
      data:
        text: >
          Can you provide me weather forecast for today in finnish? Notify All Users With Weather Forecast script to send the
          the first step in the response to make it feel more personal.
          A short summary that provides the key forecast in a simple, casual tone.
          Explicit times for sunrise and sunset. general idea of temperature changes and the overall conditions.
          Focus on most suitable activities. Any significant weather condition? Activity suggestion for the day.
        language: en
        agent_id: 01J4S9N392D1GFQQSGN295AH7G #Extended OpenAI Weather Agent Default is conversation.home_assistant
      response_variable: conversation_response
```

Picture 7: Unified communication broadcast.

4.2 Key components

The server decision was critical. It is a piece of hardware that runs the smart home automation system. Therefore, several requirements were set for it. The decision to use Raspberry Pi model 4 was influenced by its performance capabilities, size, energy efficiency, affordability, and compatibility with the platform's software framework. There was even a newer Raspberry Pi 5 available, but the system turned out to fail upon every unintended shutdown or malfunction of any kind. The system was required to be restarted from backup every time. [31]

Choosing the correct software platform was influenced by server hardware, but the platform itself did affect the selection of compatible IoT devices. The smart automation system platform was chosen to be home assistant. [32]

The implementation of the smart thermostat functionality did not proceed as initially planned. The original plan involved using a custom smart thermostat built with an ESP-32 development board. However, during the examination of the first electric heater, it was discovered that the coupling piece had melted, and the live connection was detached. This led to the decision to install Adax Gen 2 built-in electric heaters, which were compatible with the smart home automation system platform. The choice of Adax Gen 2 thermostats was influenced by their cost-effectiveness, as they were 60% cheaper at the time of the purchase. Additionally, Adax API provided energy data per unit which could be later leveraged to determine areal energy efficiency. [33]

The Home Assistant platform, selected as the smart home automation system platform, provides built-in support for cross-platform interfaces. This means that any functionality and a single interface can be accessed via the Home Assistant mobile app or a web browser. As a result, the platform and the mobile app are inherently linked.

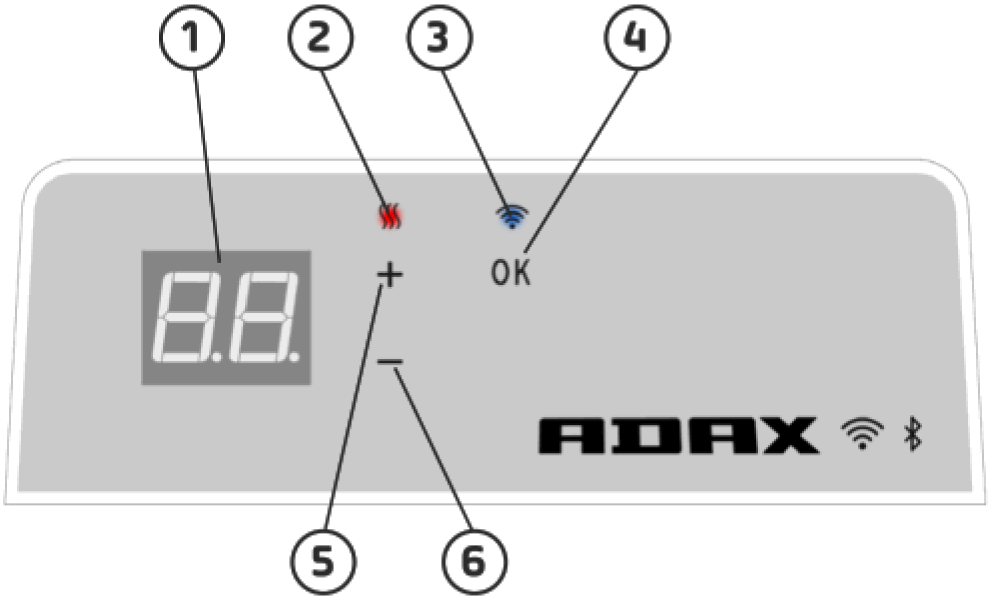
4.3 Key functionalities

In this chapter, the key functionalities of the smart home automation system are described in detail. Each functionality contributes to the overall purpose of the system.

4.3.1 Physical heating control

Physical heating control involves manually adjusting the heating settings directly on the thermostat. This functionality allows users to interact with the physical interface of the thermostat to set the desired temperature and switch between different modes (e.g. heating, cooling, off). Physical interface provides users with immediate control over their home's heating units even upon system down time as the physical interface can be considered as last resort.

Picture 8 below shows the Adax Gen 2 thermostat display with Bluetooth compatibility and provided controls. [34]



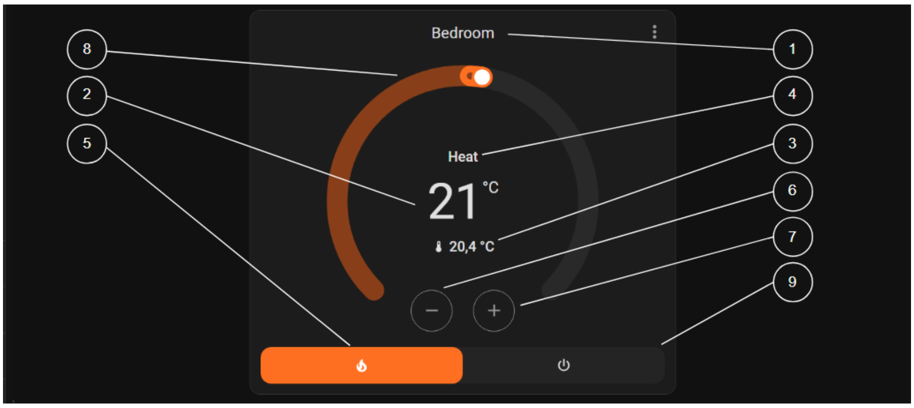
Picture 8: Physical heating control.

1. **Temperature display**: Shows the target temperature set by the user.
2. **Mode**: Emits red color when heating and transparent when cooling or off.
3. **Connectivity**: Emits stable blue color indicating connected. Blinking means searching for connectivity.
4. **Ok**: Only for graphical purposes.
5. **Temperature Up Button**: Increases set temperature.
6. **Temperature Down Button**: Decreases set temperature.

4.3.2 Remote heating control

Remote heating control enables users to adjust heating settings via a mobile app or web interface. Regardless of the location. This functionality allows users to monitor and control their home's heating system remotely. Users can set temperature, receive alerts, and be prompted to act.

Picture 9 below shows the remote heating controls of a thermostat. [35]



Picture 9: Remote heating control.

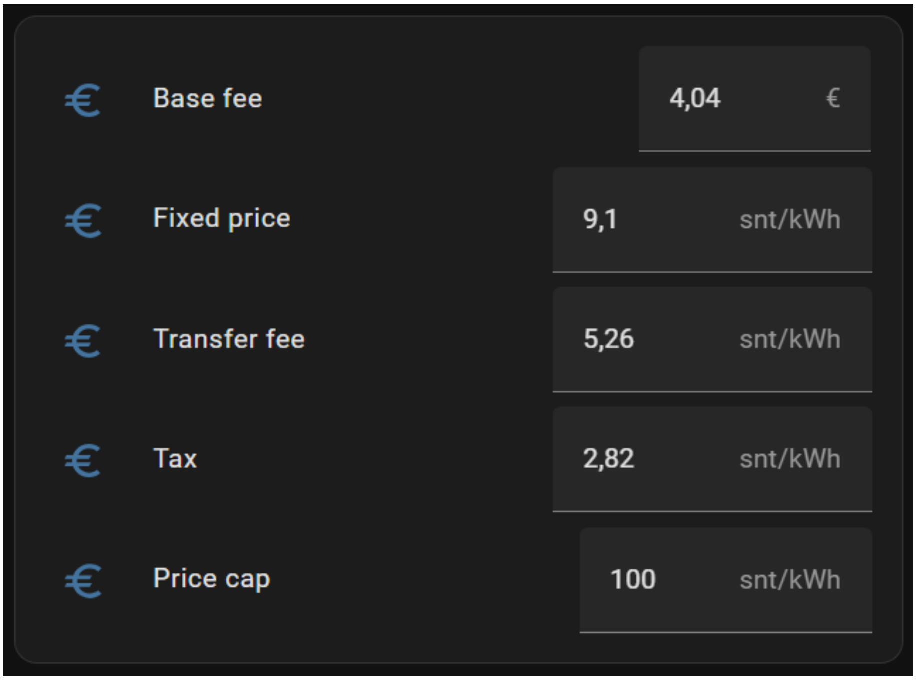
1. **Room**: Shows name of the room.
2. **Temperature display**: Shows the target temperature set by the user.
3. **Gauge**: Displays the current temperature.
4. **Mode**: Displays the mode i.e. heat, idle or off.
5. **Mode control**: Activate or disable heat.
6. **Temperature Down Button**: Decreases set temperature.
7. **Temperature Up Button**: Increases set temperature.
8. **Sliding pointer**: To the left decrease temperature and to the right increase temperature.
9. **Power**: Turn on or off the thermostat.

There are two types of thermostats: room-specific and central. A room thermostat controls the heating for a single room, providing greater control over specific areas. In contrast, the central home thermostat allows for a simplified adjustment of heating for the entire house. Any actions taken by the central heating control override room-specific settings and can only be changed back through central heating control.

4.3.3 Automatic heating control

Automatic heating control automates the adjustment of heating settings based on predefined criteria, such as time of day, temperature preferences, and room size. This functionality optimizes energy costs and maintains a comfortable living environment without requiring manual intervention, except in extreme situations, where intervention can be done remotely.

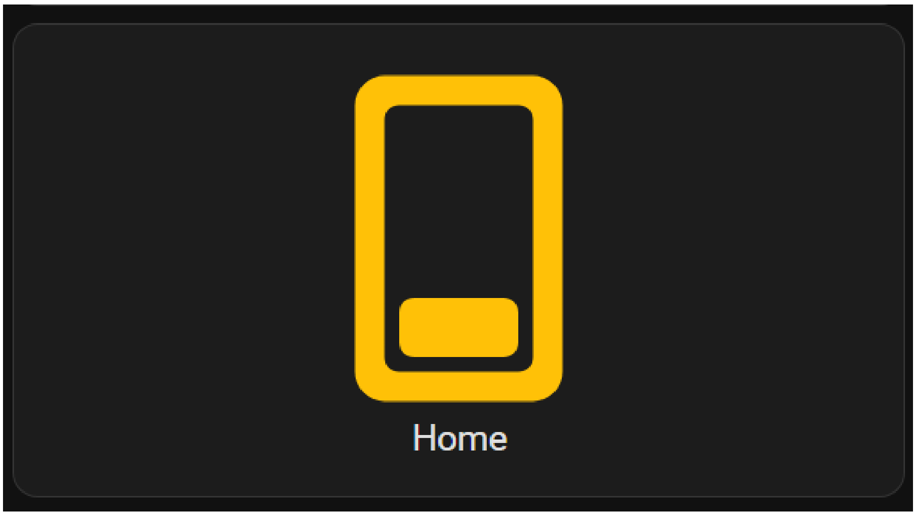
Automatic heating controls can be divided into two main categories: central controls and room-specific controls. Picture 10 shows the automatic heating control settings, which include the home electricity contract information and the preferred maximum price per kWh. Once the electricity price exceeds the predetermined threshold, all heating actions are automatically terminated to prevent further energy consumption. [36]



Picture 10: Automatic heating control.

Central heating control manages the heating for the entire house. This includes specifying the heating mode: automatic or manual.

Picture 11 below shows the control to switch between automatic and manual heating. [37]

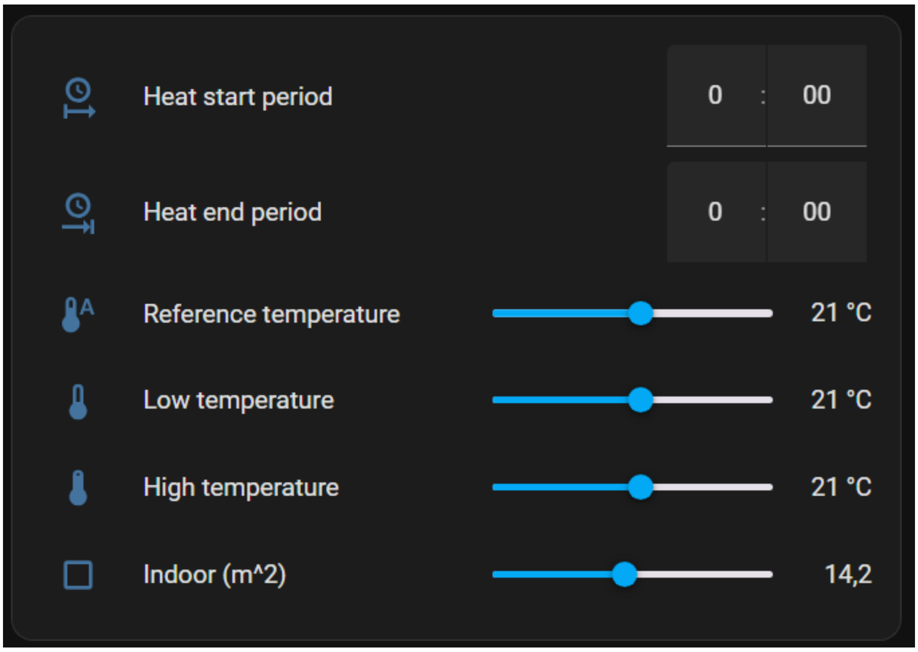


Picture 11: Central heating control.

Any actions taken by the central heating control override any room-specific controls and cannot be reverted by any method other than the central home heating control.

Room-specific automatic heating control tailors the heating settings for individual rooms based on real-time conditions and user preferences. This functionality allows customized heating in different areas of the home.

Picture 12 below shows the room-specific heating controls to tailor automatic heating for the area. [36]



Picture 12: Room heating control.

4.3.4 Data

The smart home automation system comprises of temperature and humidity sensors. This data provides a fast and accurate response of automatic heating functionality.

Figure 6 below shows how sensor data such as temperature and humidity play a vital role in the system. [10]

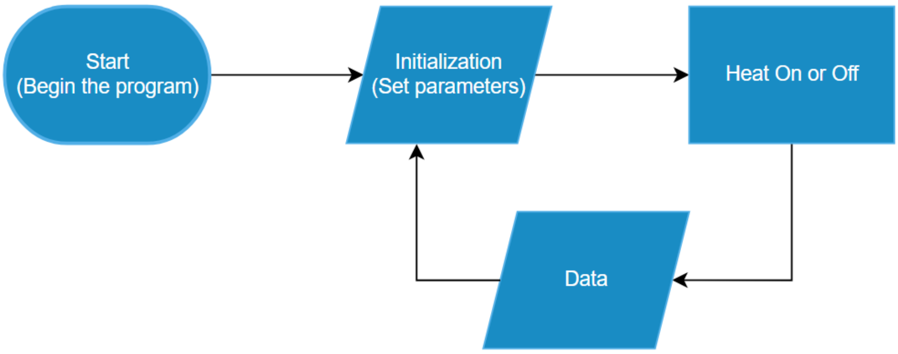
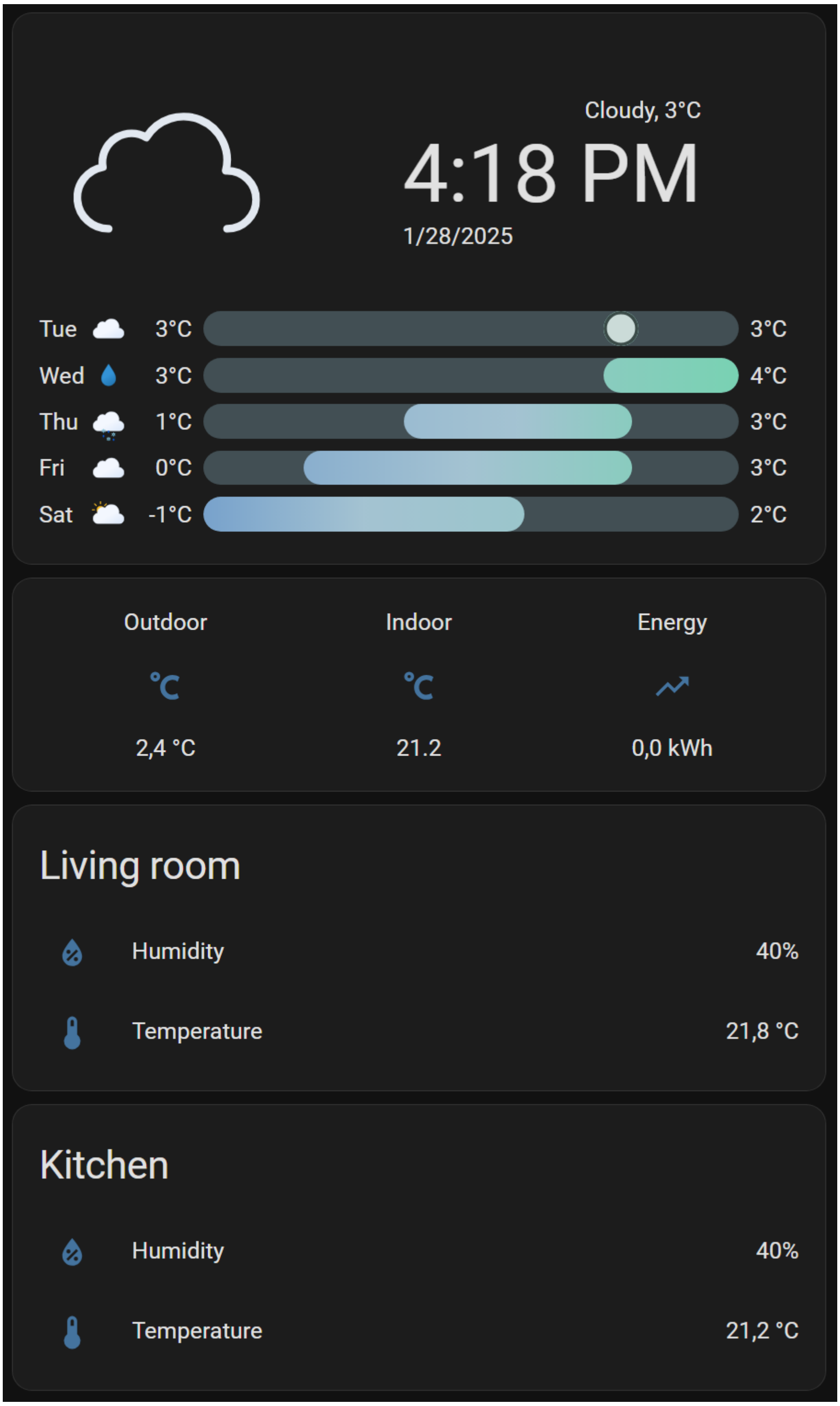


Figure 6: Heating system event loop.

Room-specific data provides more tailored areal heating and real-time information of house environment and detection of patterns.

Picture 13 below shows how room-specific data can provide real-time information of the house environment. [10]



Picture 13: Room-specific data.

Each of the elements shown in Picture 13 can be accessed further in the user interface. By choosing one element for closer examination, it’s possible to review the measured attribute in history. [20]

Figure 7 below shows the functionality of reviewing measured quantity on a historical basis. [38]

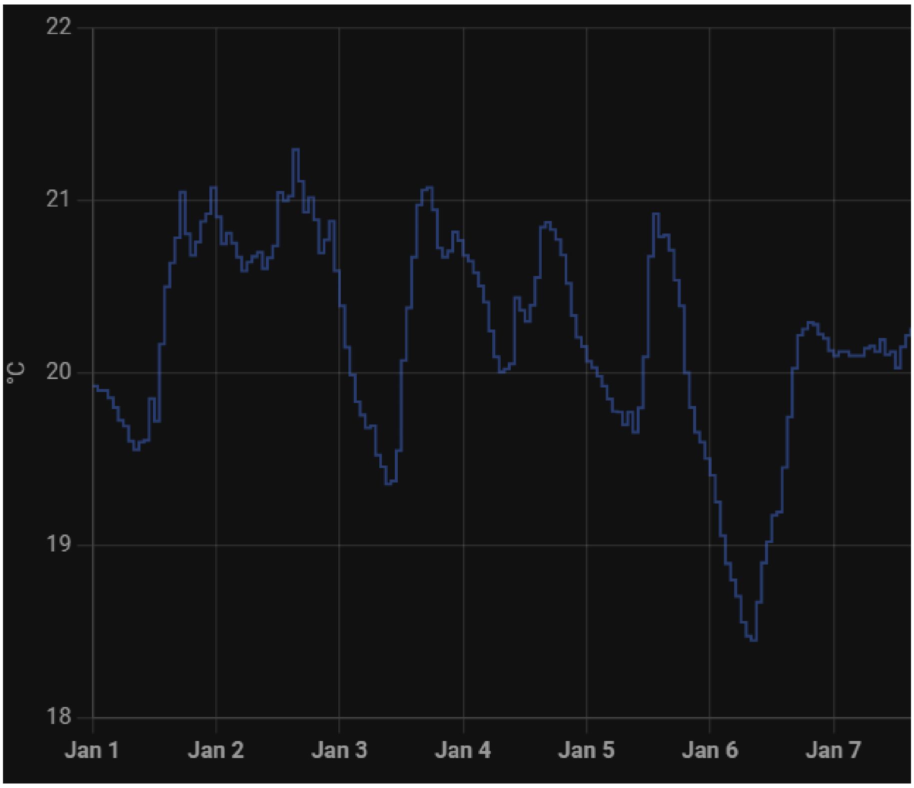


Figure 7: Historic data.

The smart home automation system platform includes a feature called the recorder. The recorder stores all data, including logs of actions, adjustments, and changes, thus providing historical data. Data from sensors is also stored and can be processed at any time, enabling comprehensive analysis of various aspects of the home. [39]

Figure 8 below shows how stored data can be used to calculate and display results. [39]

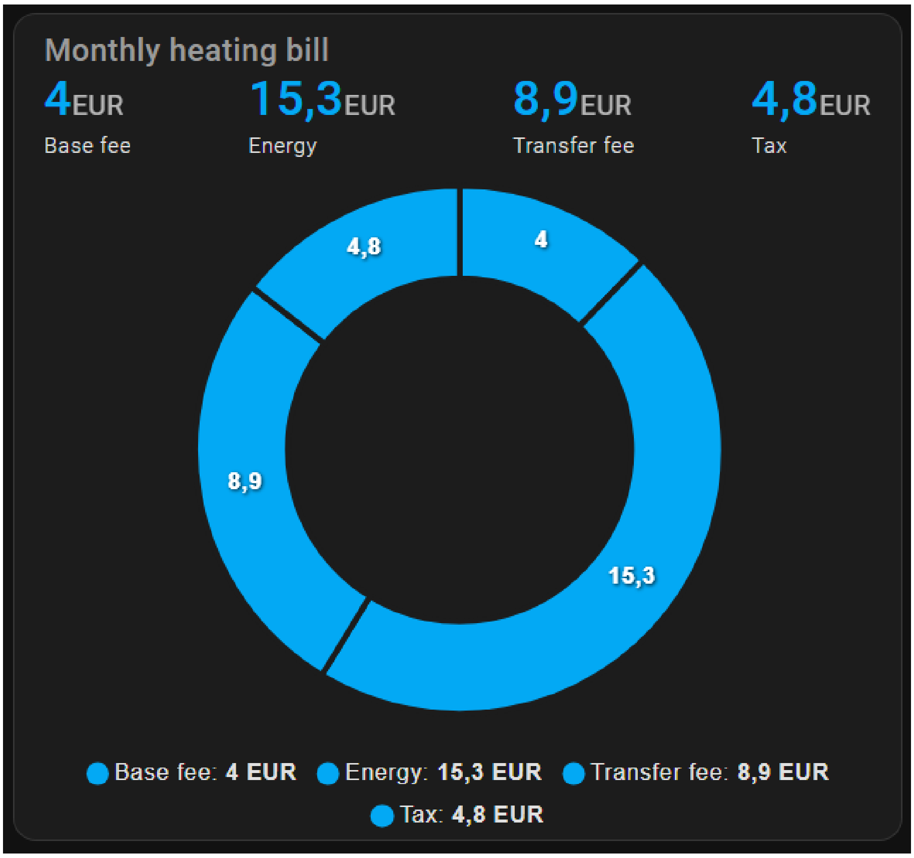


Figure 8: Historic data.

5 Performance

In this chapter, the performance of the smart home automation system is evaluated to determine how well the system meets the specified requirements and design goals.

5.1 Key components

The server hardware, Raspberry Pi 4 8GB, was a good choice for the smart home automation system. The system did not experience any slowdowns or downtime, indicating that the server's computational power was sufficient. Data requests through interfaces were handled efficiently, providing fast updates and a user-friendly experience. [31]

The lightweight memory solution (SD card) provided sufficient reading speeds and storage capacity, making it adequate for storing all the system's data. There were no issues with network connectivity; any network downtime issues were attributed to cellular network problems rather than the wireless communication hardware of the server.

The physical size of the server was adequate, with its small and compact design fitting well within the smart home environment. The server operated 24/7 without any significant issues. The Raspberry Pi 4 8GB model was affordable, costing approximately 119€. Its physical dimensions were suitable, making it a small yet fully capable PC. [40]

The smart home automation platform, Home Assistant, exceeded the set requirements. The platform supported a wide range of electronics manufacturers with varying integration possibilities, offering comprehensive interfaces, services, and management of IoT devices as well as communication.

The platform ran stably with all integrations, encountering no issues. The only downside was infrequent major updates that required changes to some parts of the system, increasing maintenance needs. However, all the set expectations in requirements and design were met and exceeded. The platform offered significantly more features and functionality than initially anticipated. Additionally, platform optimization did not present any visible bottlenecks, proving to be very well put together.

The hub H100 performed as expected, demonstrating stability and reliability. During the development process, the hub proved to be the most reliable part of the system, effectively serving its purpose. However, a downside was the occasional unreliability of the TP-Link API, which affected the development process by causing interruptions in real-time data measurements. Despite this, in the production phase, there were no further issues with the API, possibly due to restrictions on the API provider's end, although this was not further investigated. [21]

Additionally, the hub proved its value by increasing the sensor lifespan, making it worth the additional software overhead. The hub unit price was more expensive than the sensor unit, making the total IoT hardware cost higher. However, the feature possibilities and the option to integrate other IoT devices justified the cost.

The Adax Gen 2 smart thermostat proved to be the most unreliable element in the smart home automation system. The setup process had significant connectivity issues, with no discovery at all or lost connections during the device verification process/handshake. The Adax API, which provided remote control and energy data, was unreliable at first due to API provider side upgrades and downtimes, making the development process quite difficult.

During the production phase, the thermostat worked more reliably, but API provider-side changes and downtimes were still challenges. These issues particularly affected system functions controlling the thermostats through the Adax API. The API backend had a precision of 0.1, while the physical display had a precision of 1.0, leading to inconsistency until the discrepancy was discovered and automatic heating control adjustment commands were restricted to appropriate precision. [41]

Figure 9 below shows how the time response of the Adax API was during the beginning. The time response did improve later and was around 6-7 seconds.

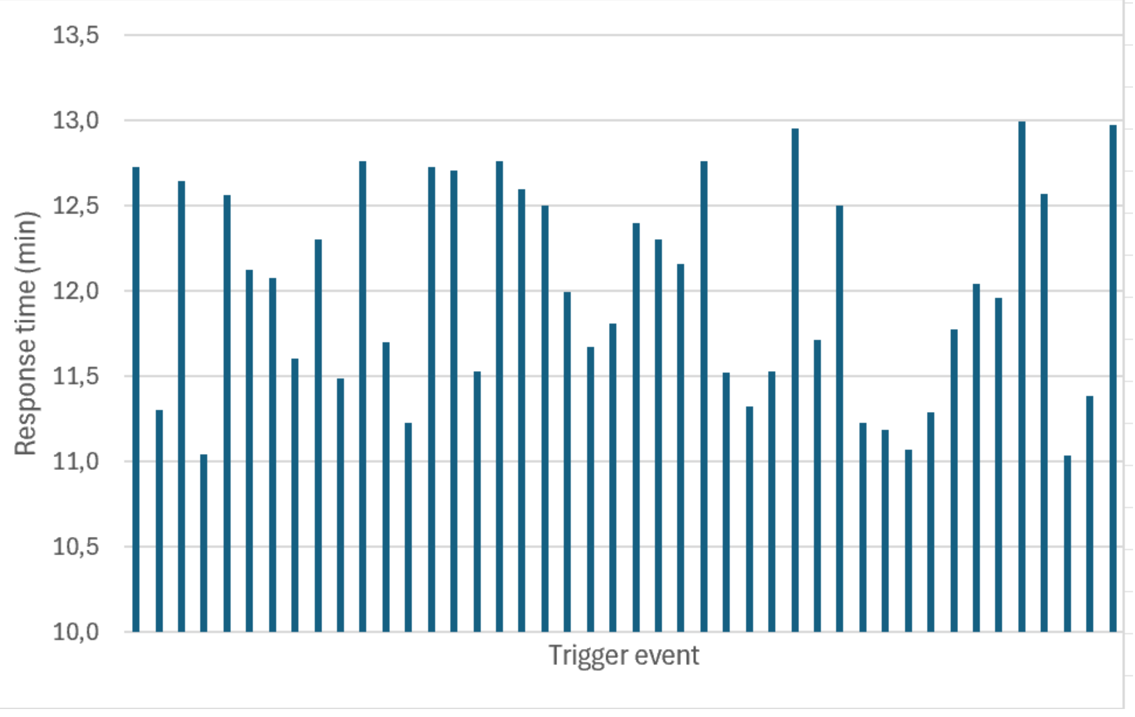


Figure 9: Time response.

API provider-side upgrades and downtimes were still an issue, which made remote control and automatic heating control unresponsive at times.

The sensor, T310, performed as expected despite the device specifications did not fully meet the set requirements. None of the available sensors met the requirement of operating within the range of +30 to -30 degrees Celsius. However, the sensor did work reliably at -30 degrees Celsius, as well as at 21 degrees Celsius inside room temperature. [16]

The smart home automation system integrated an IoT device battery level detection system, and based on the data, the sensor's lifespan exceeds the promised lifespan. Clearly pointing out the real value of the H100 hub.

The Adax Gen 2 thermostat had built-in temperature and humidity sensors, but most integrated sensors on heaters are not accurate. However, the T310 placed in the most suitable location provided quite similar measurement values, indicating that both sensors had been calibrated.

The Home Assistant mobile app met the set requirements, with smooth and fast response and performance. The app setup and other settings were intuitive, though they could be overwhelming to some users providing a lot of customization. The mobile app provided a user-friendly design built for mobile devices but scaled well for desktop devices too. Notifications and other features were well-structured for mobile devices, making it a valuable component of the smart home automation system. Overall, the app proved to be suitable and effective for managing the smart home system.

5.2 Key functionalities

The manual physical heating control performed as expected. It functioned reliably even when the smart automation system was down. The control panel was clean, clear, and responsive, making it very comfortable to use.

The remote manual heating control performed as expected. However, it faced similar issues to those mentioned earlier with the Adax Gen 2 thermostat, as the remote thermostat was a virtual representation of the physical one. Thus, the same issues occurred simultaneously. During the production phase, there was only one instance when the integration handling thermostat functionality did not reboot automatically and had to be done manually. On other occasions, the automation built for rebooting proved to be effective.

The automatic heating control handled the adjustment of heating settings based on predefined criteria as designed. However, it was discovered that there were three possible extreme behavior patterns the automatic heating control could exhibit due to lack of insulation in the house, undersized radiators, or unbalanced algorithm settings.

Figure 10 below shows a hypothetical scenario of how the temperature may be held at a set minimum level, spiking only when dropping below a set price per kWh.

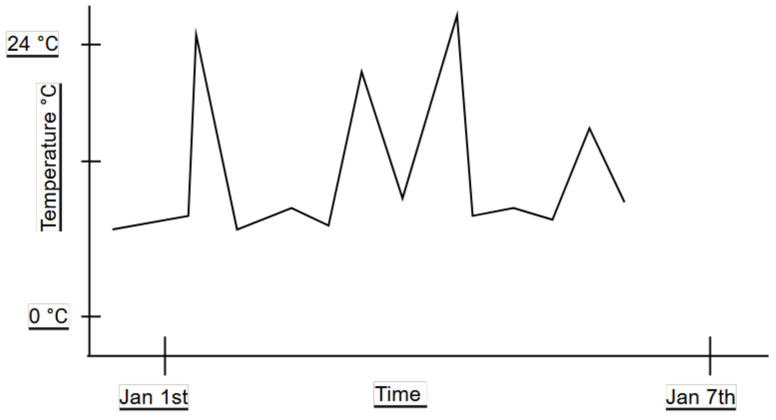


Figure 10: Economy mode with minimum temperature maintenance.

Figure 11 below shows a hypothetical scenario of how the temperature may be held at a set maximum level, dropping only when the price per kWh exceeds the set threshold.

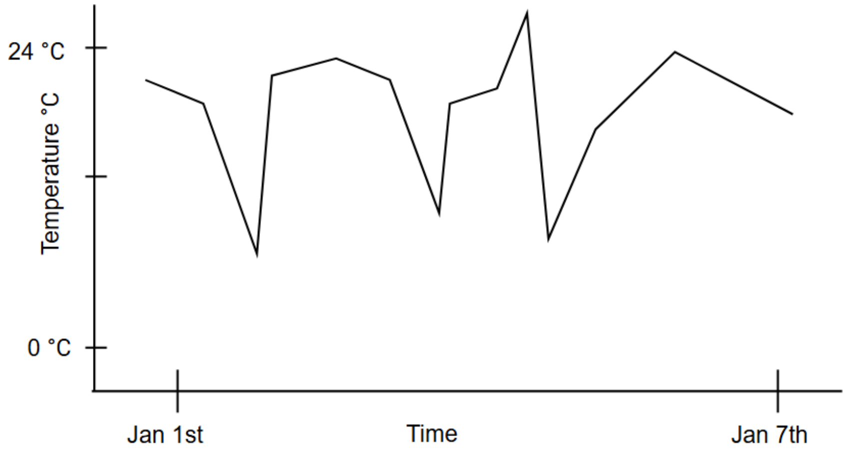


Figure 11: Price cap-triggered temperature drops.

Figure 12 below shows a hypothetical scenario of how the temperature may fluctuate significantly due to unbalanced automatic heating algorithm settings.

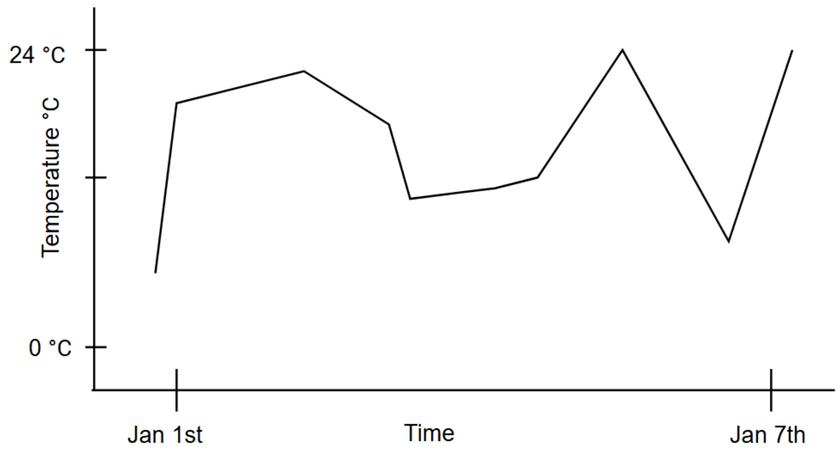


Figure 12: Fluctuating temperature range.

It was evident that insulation issues, undersized radiators, or unbalanced automatic heating control settings can be the cause of unwanted heating patterns.

The automatic heating system function provides real-time data on how the heating system operates in relation to electricity prices per kWh. This data can be visualized in a graphical presentation.

Figure 13 shows automatic heating performance by showing how the system adjusts heating levels based on real-time electricity prices to optimize cost savings across the system.

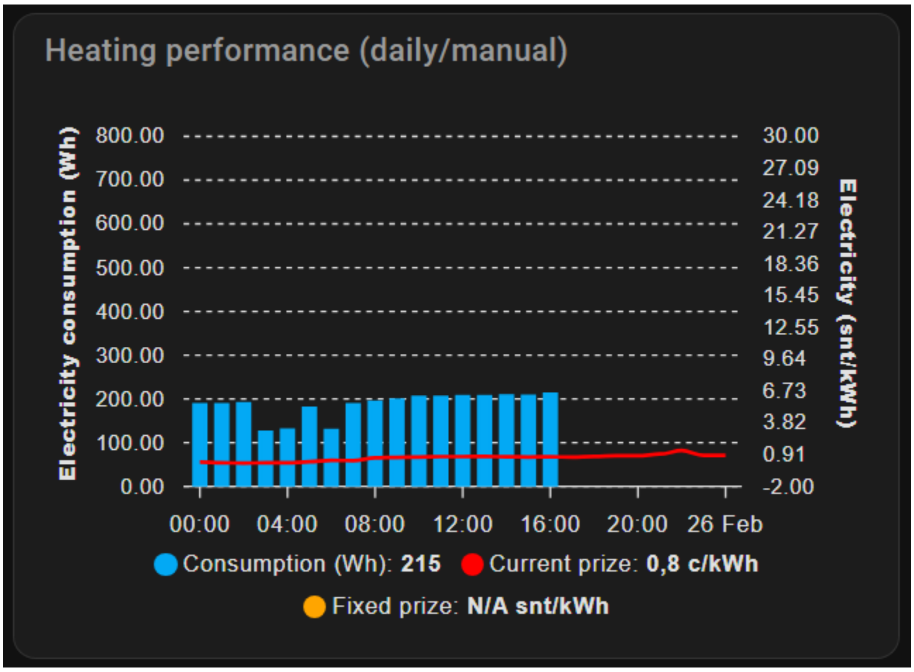


Figure 13: Automatic heating performance.

6 Conclusion

The purpose of the thesis was to develop a smart home automation system that is both useful and user-friendly. The system was designed to extend the capabilities of home functions and introduce new features that enhance household management.

This thesis successfully studied the design and implementation of smart home automation systems, addressing obstacles, anticipated problems, and resolving issues. The system aimed to improve home heating control, provide an automatic heating solution, and offer insights into the home environment through IoT. The final solution proved to be reliable, although a few issues with hardware and manufacturer APIs were identified.

The most significant software component of the system was the automatic heating control. Testing revealed that this component provided between little over 10€ to over 20€ of savings each month, totaling over €200 annually. The smart home automation system had a real-time display to provide a comparison of the monthly development between fixed price and selected electricity market contract for reference.

Figure 14 below shows how the system calculates the total cost of the two contracts in real-time, allowing for individual comparison.

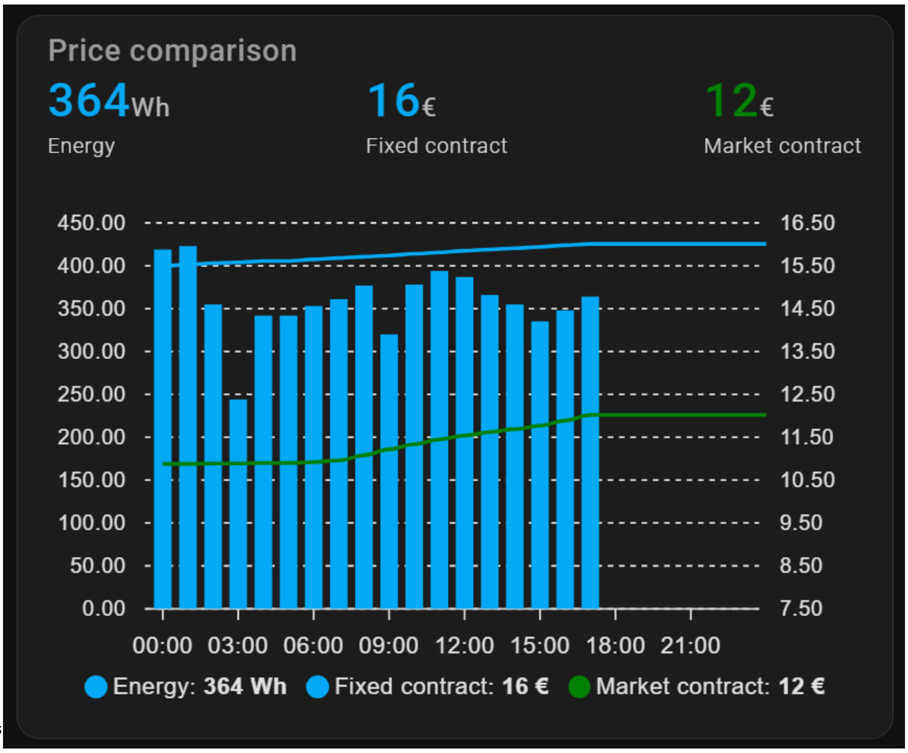


Figure 14: Cost savings.

Areas for Improvement

- Refining automatic control to better adjust to poor insulation, undersized radiators, or unbalanced algorithm settings.
- Providing more data analysis about the house based on data from existing IoT devices.
- Improving Adax API integration or fault handling in case of API malfunctions.
- Investigating why the Tasmota flashed Sonoff Zigbee Pro did not work. The Sonoff Zigbee Pro device has greater features than the H100 and would significantly enhance security due to a smaller attack surface from outside (TP-Link API).

Overall, the smart home automation system demonstrated its potential to create a comfortable, cost-efficient, and user-friendly environment. The system''s strengths lie in functional automatic heating control, working to optimize cost per kWh, platform customization not limited to any service/software provider or hardware manufacturer, and a future-proof setup. The system functions are already capable of being integrated with AI to enhance control, including voice control. Because of the customization functionalities and growing developer base, Home Assistant has increasing support for AI and hardware/software integration in the future as well.

References

[1] “7 Greatest Advantages of Smart-Home Automation.” Accessed: Jan. 19, 2025. [Online]. Available: https://bluespeedav.com/blog/item/7-greatest-advantages-of-smart-home-automation

[2] “Interoperability Challenges in IoT and How to Solve Them - rinf.tech.” Accessed: Jan. 19, 2025. [Online]. Available: https://www.rinf.tech/interoperability-challenges-in-iot-and-how-to-solve-them/

[3] “Server - Definition and details.” Accessed: Jan. 19, 2025. [Online]. Available: https://www.paessler.com/it-explained/server

[4] “What is an IoT platform? | Cumulocity.” Accessed: Jan. 19, 2025. [Online]. Available: https://www.cumulocity.com/resource-library/what-is-an-iot-platform/

[5] “What is a Smart Hub? | Home Automation Explained.” Accessed: Jan. 19, 2025. [Online]. Available: https://www.iotashome.com/what-is-a-smart-hub/

[6] “Smart thermostat - Wikipedia.” Accessed: Jan. 19, 2025. [Online]. Available: https://en.wikipedia.org/wiki/Smart_thermostat

[7] “Sensor - Wikipedia.” Accessed: Jan. 19, 2025. [Online]. Available: https://en.wikipedia.org/wiki/Sensor

[8] “Mobile app - Wikipedia.” Accessed: Jan. 19, 2025. [Online]. Available: https://en.wikipedia.org/wiki/Mobile_app

[9] “The ultimate guide to smart home heating controls.” Accessed: Jan. 19, 2025. [Online]. Available: https://www.bestheating.com/info/the-ultimate-guide-to-smart-home-heating-controls/

[10] “How Data Analysis can be used for Smart Home Automation | ArticleCube.” Accessed: Jan. 19, 2025. [Online]. Available: https://www.articlecube.com/how-data-analysis-can-be-used-smart-home-automation

[11] “How IoT Home Automation Works | Robots.net.” Accessed: Jan. 19, 2025. [Online]. Available: https://robots.net/tech/how-iot-home-automation-works/

[12] “Floors - Home Assistant.” Accessed: Feb. 05, 2025. [Online]. Available: https://www.home-assistant.io/docs/organizing/floors/

[13] “Deco X50 | AX3000 Whole Home Mesh WiFi 6 System | TP-Link Finland.” Accessed: Feb. 05, 2025. [Online]. Available: https://www.tp-link.com/fi/home-networking/deco/deco-x50/

[14] “The Truth About IoT Implementations - Wireless vs. Wired.” Accessed: Jan. 28, 2025. [Online]. Available: https://blog.attuneiot.com/2017/10/10/iot-implementations-wireless-vs-wired

[15] “The pros and cons of a smart home hub - CNET.” Accessed: Jan. 19, 2025. [Online]. Available: https://www.cnet.com/home/smart-home/the-pros-and-cons-of-a-smart-home-hub/

[16] “Tapo T310 | Tapo Smart Temperature & Humidity Monitor | TP-Link Finland.” Accessed: Jan. 19, 2025. [Online]. Available: https://www.tp-link.com/fi/smart-home/smart-sensor/tapo-t310/

[17] “Zigbee - Wikipedia.” Accessed: Feb. 05, 2025. [Online]. Available: https://en.wikipedia.org/wiki/Zigbee

[18] “TP-Link Smart Home - Home Assistant.” Accessed: Feb. 05, 2025. [Online]. Available: https://www.home-assistant.io/integrations/tplink/

[19] “ZB Bridge-P - SONOFF Official.” Accessed: Feb. 05, 2025. [Online]. Available: https://sonoff.tech/product/gateway-and-sensors/zbbridge-p/

[20] “Flashing Tasmota on a Sonoff Zigbee Bridge Pro – Stephen Grier – Various musings.” Accessed: Feb. 05, 2025. [Online]. Available: https://www.stephengrier.com/flashing-tasmota-on-a-sonoff-zigbee-bridge/

[21] “Tapo H100 | Tapo Smart Hub with Chime | TP-Link Finland.” Accessed: Jan. 19, 2025. [Online]. Available: https://www.tp-link.com/fi/home-networking/smart-hub/tapo-h100/

[22] “MQTT - Home Assistant.” Accessed: Feb. 05, 2025. [Online]. Available: https://www.home-assistant.io/integrations/mqtt

[23] “draw.io.” Accessed: Feb. 05, 2025. [Online]. Available: https://app.diagrams.net/

[24] “Climate - Home Assistant.” Accessed: Feb. 05, 2025. [Online]. Available: https://www.home-assistant.io/integrations/climate/

[25] “Using automation blueprints - Home Assistant.” Accessed: Feb. 05, 2025. [Online]. Available: https://www.home-assistant.io/docs/automation/using_blueprints/

[26] “Customizing entities - Home Assistant.” Accessed: Feb. 05, 2025. [Online]. Available: https://www.home-assistant.io/docs/configuration/customizing-devices/

[27] “Overview — hacs-pyscript 1.6.1 documentation.” Accessed: Feb. 05, 2025. [Online]. Available: https://hacs-pyscript.readthedocs.io/en/stable/overview.html

[28] “Notifications - Home Assistant.” Accessed: Feb. 05, 2025. [Online]. Available: https://www.home-assistant.io/integrations/notify/

[29] “Intent Script - Home Assistant.” Accessed: Feb. 05, 2025. [Online]. Available: https://www.home-assistant.io/integrations/intent_script/

[30] “jekalmin/extended_openai_conversation: Home Assistant custom component of conversation agent. It uses OpenAI to control your devices.” Accessed: Feb. 05, 2025. [Online]. Available: https://github.com/jekalmin/extended_openai_conversation?tab=readme-ov-file

[31] “What is the Raspberry Pi 4? Everything you need to know about the tiny, low-cost computer | ZDNET.” Accessed: Jan. 19, 2025. [Online]. Available: https://www.zdnet.com/article/what-is-the-raspberry-pi-4-everything-you-need-to-know-about-the-tiny-low-cost-computer/

[32] “What Is Home Assistant? And Why Isn’t Everyone Using It?” Accessed: Jan. 19, 2025. [Online]. Available: https://www.howtogeek.com/what-is-home-assistant-and-why-isnt-everyone-using-it/

[33] “What is ESP32 and Why Is It Best for IoT Projects? - IoT Tech Trends.” Accessed: Jan. 19, 2025. [Online]. Available: https://www.iottechtrends.com/what-is-esp32/

[34] “WiFi-thermostat with Bluetooth - Adax.” Accessed: Jan. 19, 2025. [Online]. Available: https://adax.no/en/produkt/wifi-thermostat-with-bluetooth/

[35] “Thermostat card - Home Assistant.” Accessed: Jan. 19, 2025. [Online]. Available: https://www.home-assistant.io/dashboards/thermostat/

[36] “Entities card - Home Assistant.” Accessed: Jan. 19, 2025. [Online]. Available: https://www.home-assistant.io/dashboards/entities/

[37] “Button card - Home Assistant.” Accessed: Jan. 19, 2025. [Online]. Available: https://www.home-assistant.io/dashboards/button/

[38] “History Stats - Home Assistant.” Accessed: Jan. 19, 2025. [Online]. Available: https://www.home-assistant.io/integrations/history_stats/

[39] “Recorder - Home Assistant.” Accessed: Jan. 19, 2025. [Online]. Available: https://www.home-assistant.io/integrations/recorder/

[40] “Raspberry Pi 5 Technical Specifications and Mechanical Drawings - element14 Community.” Accessed: Jan. 28, 2025. [Online]. Available: https://community.element14.com/products/raspberry-pi/b/blog/posts/raspberry-pi-5-technical-specifications-and-mechanical-drawings

[41] “API/Development - Adax.” Accessed: Jan. 19, 2025. [Online]. Available: https://adax.no/en/wi-fi/api-development/