Table of Contents

[Case for 2nd interview 3](#_Toc210785355)

[I. Conceptual Design 4](#_Toc210785356)

[I.1. Data collection and usage 4](#_Toc210785357)

[I.1.1. Real-time sensors 4](#_Toc210785358)

[I.1.2. Meta / configuration data 5](#_Toc210785359)

[I.1.3. How data are used 5](#_Toc210785360)

[I.2. Algorithm and logic 5](#_Toc210785361)

[I.2.1. Architectural choices 5](#_Toc210785362)

[I.2.2. Special logic for PCM pre-charging 5](#_Toc210785363)

[I.2.3. Adaptation & learning 6](#_Toc210785364)

[I.2.4. Heuristics / Fallback 6](#_Toc210785365)

[I.2.5. Control flow diagram 6](#_Toc210785366)

[II. User Interface (UI) Proposal 7](#_Toc210785367)

[II.1a. UI / UX design — screens & elements 7](#_Toc210785368)

[II.1.1. Main dashboard (single page) 7](#_Toc210785369)

[II.1.2. Mobile UX 9](#_Toc210785370)

[II.1.3. Setup & advanced pages 9](#_Toc210785371)

[II.1.4. Notifications & alerts & summary 10](#_Toc210785372)

[II.1b. Frontend, backend and tools 11](#_Toc210785373)

[II.1b.1. Frontend (User Interface Layer) 11](#_Toc210785374)

[II.1b.2. Backend (Application & Control Logic Layer) 11](#_Toc210785375)

[II.1b.3 Database Layer 12](#_Toc210785376)

[II.1b.4. Integration & Data Flow Overview 12](#_Toc210785377)

[II.1b.5 Recommended Stack Summary 12](#_Toc210785378)

[II.2. Ensure usability and user management 13](#_Toc210785379)

[II.2.1. Usability Foundations 13](#_Toc210785380)

[II.2.2. Engagement and Interaction Design 13](#_Toc210785381)

[II.2.3. Analytics & Behavioral Insights 13](#_Toc210785382)

[II.2.4. Trust, Transparency & Feedback Loop 14](#_Toc210785383)

[II.2.5. Tools & Methods to Implement 14](#_Toc210785384)

[II.2.6. Outcome 14](#_Toc210785385)

[III. Challenges and Considerations 15](#_Toc210785386)

[III.1. Main technical and user-related challenges 15](#_Toc210785387)

[III.1.1. Technical Challenges 15](#_Toc210785388)

[III.1.2. Data and Algorithmic Challenges 15](#_Toc210785389)

[III.1.3. User-Related Challenges 16](#_Toc210785390)

[III.1.4. Summary Table 17](#_Toc210785391)

[III.2. Security, data privacy, user preferences, and system reliability 17](#_Toc210785392)

[III.2.1. Security 17](#_Toc210785393)

[III.2.2. Data Privacy 17](#_Toc210785394)

[III.2.3. User Preferences & Comfort 17](#_Toc210785395)

[III.2.4. System Reliability & Robustness 18](#_Toc210785396)

[III.2.5. Summary Table 18](#_Toc210785397)

[IV. My Contributions 19](#_Toc210785398)

[IV.1. Background and skills 19](#_Toc210785399)

[IV.1.1. System Architecture & Scalability 19](#_Toc210785400)

[IV.1.2. Data Engineering & Integration 19](#_Toc210785401)

[IV.1.3. Optimization & AI Implementation 19](#_Toc210785402)

[IV.1.4. User Interface & Experience 19](#_Toc210785403)

[IV.1.5. DevOps & System Reliability 19](#_Toc210785404)

[IV.1.6. Data Security & Compliance 20](#_Toc210785405)

[IV.1.7. Leadership & Collaboration 20](#_Toc210785406)

[IV.1.8. Summary Table 20](#_Toc210785407)

[IV.2 Additional ideas or improvements 21](#_Toc210785408)

[IV.2.1. Smarter Control Algorithms 21](#_Toc210785409)

[IV.2.2. Integration with Renewable and Smart Grid Systems 21](#_Toc210785410)

[IV.2.3. Advanced Analytics & Insights 21](#_Toc210785411)

[IV.2.4. Digital Twin Simulation 21](#_Toc210785412)

[IV.2.5. Enhanced User Interface Features 22](#_Toc210785413)

[IV.2.6. System Security & Edge Computing 22](#_Toc210785414)

[IV.2.7. Broader Ecosystem Integration 22](#_Toc210785415)

[IV.2.7. Research & Innovation Opportunities 22](#_Toc210785416)

[IV.2.8. Summary Table 22](#_Toc210785417)

# Case for 2nd interview

**Title: Designing a Smart Heating Strategy for Energy-Efficient Buildings**

**Background:**

You are part of a research team working on a Horizon Europe project focused on improving energy efficiency and human comfort in buildings using digital tools. The project involves a smart floor system embedded with **phase change materials (PCM)** that act as passive thermal storage. A **web application** is being developed to control the heating system based on real-time data, electricity prices, and weather forecasts.

**Case Task:**

Imagine you are tasked with **designing the logic and user interface** for a digital tool that optimizes the heating of the smart floor system. The tool should:

1. **Incorporate real-time temperature data from sensors**
2. **Utilize next-day electricity prices and weather forecasts**
3. **Minimize energy consumption and cost**
4. **Maximize user comfort**

**Deliverables:**

Prepare a presentation (max. 30 minutes) addressing the following:

1. **Conceptual Design:**
   * What kind of data would you collect and how would you use it?
   * What algorithms or logic would you implement to optimize heating?
2. **User Interface (UI) Proposal:**
   * Sketch or describe the interface for both the web and mobile applications. Explain how you define frontend and backend database, tools you will use and why.
   * How would you ensure usability and user engagement?
3. **Challenges and Considerations:**
   * What are the main technical and user-related challenges?
   * How would you address issues like security, data privacy, user preferences, and system reliability?
4. **Your Contribution:**
   * How would your background and skills contribute to solving this challenge?
   * What additional ideas or improvements would you propose?

**Format:** You may use slides, diagrams, or a short written report. Be ready to discuss your ideas during the interview.

# Conceptual Design

A predictive control system (primary: Model Predictive Control) that uses real-time sensor data, next-day electricity prices and weather forecasts to optimally schedule the floor heating power and pre-charge the PCM thermal buffer so cost and energy are minimized while user comfort is guaranteed.

## I.1. Data collection and usage

**NOTE: The data collected in this part should also be stored together with planning / controlling data (created when optimization algorithms in I.2) as historical data for later estimation and reinforcement learning.**

### **I.1.1. Real-time sensors**

(sample period: 1–10 min depending on latency):

* **Room air temperature(s)** (one per room/zone): comfort control and state estimation.
* **Floor surface temperature(s)** (near PCM layer): direct measurement of PCM/floor state and to avoid overheating floor.
* **PCM temperature(s)** (if possible, or inferred from floor temp): to estimate PCM phase state (solid/liquid fraction).
* **Power consumption of heater(s)** (smart meter or per-circuit power sensor): closed-loop energy accounting and cost calculation.
* **Actuator state** (valve/opening, pump speed, on/off): ensure commands are executed and for fault detection.
* **External weather feed** (API): next-day/hourly outdoor temperature, solar irradiance, wind (affects envelope losses).
* **Electricity price feed** (API): hourly next-day prices (or real-time if available).
* **Solar gains / window orientation** (static config or irradiance sensor): to estimate passive gains.
* **Occupancy / presence** (motion sensors, calendar integration, phone geofence): to relax constraints when empty.
* **Humidity (optional)**: can affect perceived comfort and heat capacity of air.
* **(other additional data upon needed)**

### **I.1.2. Meta / configuration data**

**(one-time or rarely changed data)**

* Building envelope parameters (R-values, floor area, ceiling height) — can be estimated/learned.
* PCM properties: melting temperature(s), latent heat per m², effective heat capacity curve, mass/volume (entered in setup or identified).
* Floor heating actuator limits (max power, min on/off interval, ramp rate).
* User comfort preferences (target temp ± tolerance, schedule, “comfort vs cost” slider).
* Safety limits (max floor temperature to protect floor/occupants).

### **I.1.3. How data are used**

* **State estimation:** combine floor/air/PCM temps and power to estimate thermal state (air temp, floor temp, PCM fraction).
* **Schedule heating plan:** optimizer produces heater power schedule (and pump/valve commands) for the next 24 hours.
* **Real-time control decision:** execute heating according to schedule with real-time monitoring based on feedback data from sensors
* **Adaptive learning:** use historical data to refine building thermal model and PCM parameters (system identification).

## I.2. Algorithm and logic

### I.2.1. Architectural choices

* **Core controller:** Model Predictive Control (MPC) with a receding horizon (24 hours horizon, 5–15 min timestep).
* **Local safety controller:** fast PID/relay loop on floor temperature to prevent overheating (executes between MPC updates).
* **Parameter learning:** online system identification (recursive least squares) to update RC thermal model and PCM effective capacity.
* **Long-term adaption:** Reinforcement Learning.

### I.2.2. Special logic for PCM pre-charging

* When next-day prices include low-price windows (e.g., off-peak), schedule **pre-charge**: increase floor heating power before peak hours so PCM melts/charges and then releases during high price/peak cold periods.
* If forecast predicts sunny midday (solar gains), shift pre-charge to later or reduce it.
* If occupancy changes (user home sooner), adapt setpoint and possibly use stored PCM heat to avoid extra energy.

### I.2.3. Adaptation & learning

* Use recursive least squares or Kalman filter to fit the RC model parameters and PCM capacity from logged temp/power data.
* Detect degraded performance or sensor faults via residual monitoring; notify user.

### I.2.4. Heuristics / Fallback

* If price or forecast API fails: run conservative schedule — maintain comfort band with minimal energy and apply simple time-of-day pre-charge based on historical price pattern.
* Safety overrides: never exceed floor temp safety threshold.

### A diagram with colorful labels AI-generated content may be incorrect.I.2.5. Control flow diagram

# User Interface (UI) Proposal

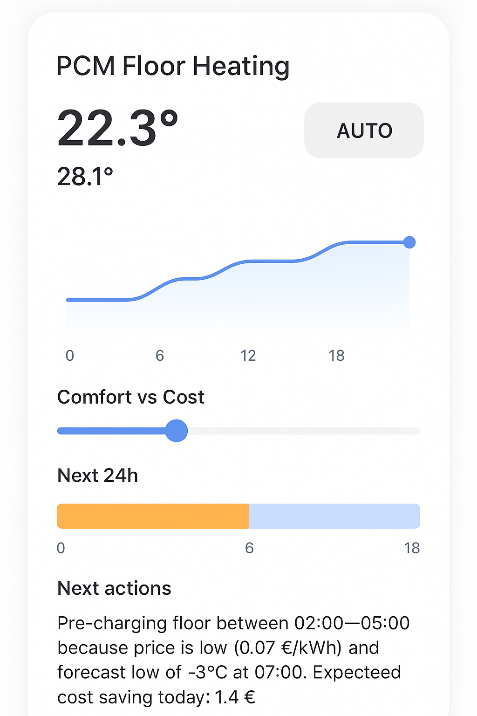
## II.1a. UI / UX design — screens & elements

Design principles: transparency, control, explainability, and a single slider to trade comfort vs cost.

### **A screenshot of a graph AI-generated content may be incorrect.**II.1.1. Main dashboard (single page)

* **Top strip (current state):** current room temp, floor temp, PCM SOC %, current electricity price, "comfort score" (0–100).
* **24-hour timeline** (centerpiece): stacked view with:
  + Hourly electricity price curve (line).
  + Weather forecast (mini bars for outside temp + icons for sun/cloud).
  + Predicted room temp (line).
  + Planned heater power schedule (bar chart under overlay).
  + Shaded band for comfort bounds.
  + Highlighted pre-charge windows (colored).
* **Quick controls:**
  + "Comfort vs Cost" slider (left: cost priority — wider comfort band; right: comfort priority — hold temp tight).
  + Setpoint temperature (per zone) and schedule.
  + Toggle "Auto / Manual" and a manual power slider for immediate override.
* **Explainability box:** short plain-language explanation of next actions, e.g.:  
  “Pre-charging floor between 02:00–05:00 because price is low (0.07 €/kWh) and forecasted low of -3°C at 07:00. Expected cost saving today: 1.4 €.”
* **Energy & Cost preview:** estimated energy usage and cost for next 24 h under current plan + comparison vs. "no optimization".

### II.1.2. Mobile UX



* Condensed dashboard: current temp, next price window, one-tap “force pre-charge” or “eco mode”.
* Push notifications for important events (e.g., planned pre-charge started).

### II.1.3. Setup & advanced pages

* **System configuration wizard:** prompts for PCM melting temp, PCM mass/area (or "auto-identify" button).
* **Sensor map:** show where sensors are located; status (online/offline).
* **Price & weather sources:** show API connections, manual override for price schedule.
* **Advanced tuning:** MPC horizon, timestep, α weight, safety limits, system ID toggle.

### II.1.4. Notifications & alerts & summary

* Alerts for sensor faults, failed actuator commands, or if floor temp limit approached.
* A screenshot of a data analysis

  AI-generated content may be incorrect.Weekly/monthly energy/cost summary and learned thermal parameters.

## II.1b. Frontend, backend and tools

### II.1b.1. Frontend (User Interface Layer)

#### Purpose

Provide real-time visualization, control, and personalization for users (homeowners, facility managers, engineers).

#### Key Features

* Real-time room and floor temperature dashboard
* Visualization of energy consumption and cost trends
* Manual override for heating setpoints
* Historical analytics (weekly/monthly reports)
* Mobile and tablet responsive design

#### Suggested Tools

| **Function** | **Technology** |
| --- | --- |
| Web Framework | **React.js** (for modular UI) |
| Mobile | **React Native** (shared logic with web) |
| Visualization | **Recharts** or **D3.js** for live graphs |
| UI Components | **TailwindCSS** + **ShadCN UI** for sleek design |
| State Management | **Redux Toolkit** or **Recoil** |
| Real-Time Updates | **WebSocket** / **MQTT client** |
| Authentication | **Auth0** or **Firebase Auth** |
| Build & Deployment | **Vercel** or **Netlify** |

### II.1b.2. Backend (Application & Control Logic Layer)

Integrate data streams (sensors, forecasts, pricing), run optimization algorithms, and serve APIs to the frontend.

#### Key Features

* Handle incoming IoT sensor data (temperature, humidity, floor state)
* Fetch weather and electricity price forecasts
* Run optimization algorithms (e.g., MPC, RL)
* Store historical and simulation data
* Manage user profiles and preferences

#### Suggested Tools

| **Function** | **Technology** |
| --- | --- |
| Framework | **FastAPI (Python)** or **Node.js (Express)** |
| IoT Integration | **MQTT broker** (e.g., **Eclipse Mosquitto**) |
| Task Scheduling | **Celery + Redis** (Python) or **BullMQ** (Node.js) |
| Optimization Engine | Python **NumPy**, **SciPy**, or **PyTorch** (for learning-based control) |
| API Design | **REST** + **OpenAPI (Swagger)** for documentation |
| Caching | **Redis** |
| Containerization | **Docker**, orchestrated via **Kubernetes** |
| CI/CD | **GitHub Actions** or **GitLab CI** |

### II.1b.3 Database Layer

Store sensor readings, optimization results, system configuration, and user data.

#### Data Categories

| **Data Type** | **Description** | **Suggested Storage** |
| --- | --- | --- |
| Real-Time Sensor Data | Temperature, humidity, floor PCM state | **Time-series DB (InfluxDB / TimescaleDB)** |
| Historical & Forecast Data | Weather, electricity prices, predictions | **PostgreSQL** |
| Optimization Results | Model parameters, cost functions | **MongoDB** (for flexible schema) |
| User Preferences | Comfort settings, schedules | **PostgreSQL / Firebase** |
| Logs & Events | System diagnostics | **ElasticSearch** (for search and analytics) |

### II.1b.4. Integration & Data Flow Overview

1. **Sensors → MQTT Broker → Backend**
   * Collect temperature, humidity, and PCM thermal data in real-time.
2. **Backend → Optimization Engine**
   * Compute heating schedule using Model Predictive Control (MPC).
3. **Backend → Database**
   * Store current state, historical performance, and optimization results.
4. **Frontend → Backend APIs**
   * Display data visualizations, send user preferences, and enable control actions.
5. **External APIs → Backend**
   * Integrate electricity market and weather forecast data daily.

### II.1b.5 Recommended Stack Summary

| **Layer** | **Tool / Technology** | **Purpose** |
| --- | --- | --- |
| **Frontend** | React + Tailwind + MQTT client | Real-time, responsive user interface |
| **Backend** | FastAPI + Python (MPC/ML models) | Control logic and optimization |
| **Database** | PostgreSQL + InfluxDB | Persistent + time-series data |
| **Communication** | MQTT + REST API | Real-time & scheduled updates |
| **Deployment** | Docker + Kubernetes | Scalability and reliability |
| **Monitoring** | Grafana + Prometheus | System health and analytics |

## II.2. Ensure usability and user management

### II.2.1. Usability Foundations

#### a. User-Centered Design (UCD)

* **Personas:** Define clear user groups — e.g., homeowners, building managers, energy engineers.
* **Context Analysis:** Study when and how users interact (morning preheating, cost alerts, etc.).
* **Iterative Testing:** Conduct usability tests every development phase (A/B or moderated sessions).

#### b. Interface Clarity

* **Dashboard Hierarchy:** Prioritize comfort and cost at the top level; technical details hidden in advanced views.
* **Color Coding:** Use intuitive cues — blue for cooling, orange for heating, green for efficiency.
* **Minimal Cognitive Load:** Simple graphs (temperature vs time, energy cost per day).
* **Mobile-first design:** Ensure one-handed operation, fast load times, and offline fallback.

#### c. Accessibility & Inclusivity

* Support **dark/light themes** for visibility.
* Comply with **WCAG 2.1** for text contrast and touch targets.
* Include **multi-language support** for diverse users (e.g., English, Norwegian).

### II.2.2. Engagement and Interaction Design

#### a. Personalization

* Adaptive comfort modes: *Eco*, *Comfort-first*, *Smart-balance*.
* Personalized daily summaries: *“You saved 22% energy this week”*.
* Dynamic UI: Show different KPIs depending on user type (home vs facility).

#### b. Gamification Elements

* Weekly energy-saving streaks and achievements.
* Comfort-efficiency score badge.
* Community benchmarks (optional privacy-safe comparison).

#### c. Smart Feedback

* **Predictive notifications:**

“Electricity prices will rise tomorrow — preheating recommended.”

* **Voice/Chatbot interface** for quick commands: “Set living room to comfort mode.”
* **Interactive charts:** Tap to simulate “What if” cost scenarios.

### II.2.3. Analytics & Behavioral Insights

#### a. Continuous Improvement

* Collect anonymized UX metrics: time-on-screen, preferred features, actions dropped mid-flow.
* Use **A/B testing** for UI layouts (comfort slider vs temperature graph).
* Apply **heatmaps** to optimize dashboard interactions.

#### b. Engagement KPIs

| **Metric** | **Target** | **Measurement Method** |
| --- | --- | --- |
| Active Users / Month | ≥80% of installations | Analytics tracking |
| Energy Efficiency Improvement | ≥20% vs baseline | Comparison reports |
| Comfort Satisfaction Score | ≥8/10 | Periodic in-app survey |
| Alert Response Rate | ≥60% | Notification interaction logs |

### II.2.4. Trust, Transparency & Feedback Loop

#### a. Explainable Optimization

Show users **why** the system made a decision:

“Preheated at 03:00 due to low night tariff and predicted outdoor temp of -5°C.”

#### b. User Overrides

Allow easy manual control at any time, without penalty — always respecting user trust.

#### c. Feedback Collection

* One-tap feedback for each optimization decision: thumb up / down
* Automatic learning loop: Adjust comfort models based on feedback history.

### II.2.5. Tools & Methods to Implement

| **Category** | **Tools / Frameworks** |
| --- | --- |
| UX Research | Figma, Maze, Hotjar |
| User Analytics | Google Analytics 4, Amplitude |
| A/B Testing | Optimizely or custom backend logic |
| Feedback Collection | In-app micro-surveys (Typeform / custom) |
| Notification System | Firebase Cloud Messaging, Twilio |
| Personalization Engine | Python ML service integrated with user behavior data |

### II.2.6. Outcome

A user experience that is:

* **Engaging** through feedback and personalization
* **Transparent** through explainable AI decisions
* **Empowering** by allowing control without complexity
* **Habit-forming** by rewarding efficient behavior

# Challenges and Considerations

## III.1. Main technical and user-related challenges

### III.1.1. **Technical Challenges**

#### **a. Thermal Modeling Complexity**

* **PCM behavior is nonlinear** — melting/solidifying rates vary with temperature gradients and material thickness.
* Building envelope and occupant heat gains add unpredictable variability.
* → **Solution:** Use hybrid models combining physical equations and machine learning (e.g., physics-informed neural networks).

#### **b. Real-Time Data Integration**

* Synchronizing **sensor data**, **weather forecasts**, and **electricity prices** is difficult due to latency, missing values, and different update intervals.
* → **Solution:** Implement buffering and timestamp alignment mechanisms (e.g., Kafka or MQTT with QoS).

#### **c. Optimization Computation Load**

* Model Predictive Control (MPC) requires solving optimization problems at every time step.
* High computation cost can cause delays, especially for multi-room systems.
* → **Solution:** Use **edge computing** for local decisions and **cloud processing** for long-term learning.

#### **d. Scalability & Interoperability**

* Integration with existing **Building Management Systems (BMS)** or **smart home ecosystems** (e.g., KNX, Home Assistant) can be complex.
* → **Solution:** Use standardized protocols (BACnet, Modbus, MQTT, or RESTful APIs).

#### **e. Sensor Calibration & Reliability**

* Drift, noise, and sensor placement can distort thermal data and affect optimization accuracy.
* → **Solution:** Implement automatic calibration and anomaly detection algorithms.

#### **f. Security & Data Privacy**

* IoT systems are vulnerable to attacks on sensors, cloud APIs, or user devices.
* → **Solution:** Apply **end-to-end encryption (TLS)**, **secure OTA updates**, and **role-based access control**.

### III.1.**2. Data and Algorithmic Challenges**

#### **a. Data Quality**

* Inconsistent or missing data due to sensor failures or network downtime.
* → **Solution:** Use data validation and interpolation routines before feeding models.

#### **b. Forecast Uncertainty**

* Weather and electricity price forecasts are inherently probabilistic.
* → **Solution:** Use **stochastic optimization** or **ensemble forecasting** to handle uncertainty.

#### **c. Model Adaptability**

* Energy behavior changes over seasons and occupancy patterns.
* → **Solution:** Use **online learning algorithms** that adapt to feedback over time.

#### **d. Explainability of AI Decisions**

* Users may distrust “black-box” decisions.
* → **Solution:** Add explainable AI layers that show simple justifications like:

“System preheated due to low tariff and expected outdoor temperature drop.”

### III.1.**3. User-Related Challenges**

#### **a. Trust and Transparency**

* Users may feel uncomfortable with an automated system controlling heating.
* → **Solution:** Always provide **manual override** and clear visualization of what the system is doing and why.

#### **b. Comfort vs. Cost Trade-off**

* Different users have different tolerances: one prefers warmth, another prefers savings.
* → **Solution:** Use **personalized comfort profiles** (Eco, Balanced, Comfort-first).

#### **c. Cognitive Load**

* Too much data (graphs, temperatures, costs) can overwhelm users.
* → **Solution:** Keep UI minimal and use **progressive disclosure** (simple summary first, details on tap).

#### **d. Engagement and Feedback**

* Users may ignore notifications or stop engaging once installed.
* → **Solution:** Introduce **gamification**, weekly reports, or energy badges to sustain interest.

#### **e. Multi-user Coordination**

* Different household members may have conflicting preferences.
* → **Solution:** Enable **zonal or role-based control** (e.g., “child room comfort mode,” “whole-home eco mode”).

### III.1.**4. Summary Table**

| **Challenge Type** | **Example Issue** | **Recommended Strategy** |
| --- | --- | --- |
| **Thermal Modeling** | Nonlinear PCM dynamics | Physics-informed hybrid models |
| **Data Integration** | Asynchronous feeds | MQTT buffering + timestamp sync |
| **Optimization Load** | Slow MPC response | Edge + cloud hybrid processing |
| **Forecast Uncertainty** | Weather/price errors | Stochastic or ensemble optimization |
| **User Trust** | Reluctance to automate | Transparent decisions, manual override |
| **Usability** | Data overload | Simple, progressive UI |
| **Engagement** | Notification fatigue | Reports and feedback loops |

## III.2. Security, data privacy, user preferences, and system reliability

### III.2.1. Security

**Goal:** Prevent unauthorized access to devices, data, and control systems.  
**Approach:**

* **End-to-End Encryption (E2EE):** Use TLS 1.3 for all data in transit and AES-256 for data at rest.
* **Authentication & Authorization:** Implement OAuth 2.0 / OpenID Connect for user authentication; role-based access control (RBAC) for admins, installers, and users.
* **Secure Firmware & OTA Updates:** Digitally sign updates; use hardware-based trust (e.g., TPM or secure boot).
* **Network Hardening:** Segment IoT devices from the public network using VPN tunnels and firewalls.
* **Continuous Security Audits:** Regular penetration testing and automated vulnerability scanning.

### III.2.2. Data Privacy

**Goal:** Protect personal and behavioral data while maintaining system intelligence.  
**Approach:**

* **Data Minimization:** Only collect what’s essential — e.g., room temperature, energy consumption, and user-set preferences.
* **Anonymization:** Replace user identifiers with pseudonyms when data is transmitted to the cloud for analysis.
* **User Consent & Control:** GDPR-compliant privacy settings allowing users to view, delete, or export their data.
* **Local Processing:** Run real-time control logic and sensitive analytics on the **edge gateway**, reducing cloud exposure.
* **Retention Policy:** Automatically delete or archive data after a defined period (e.g., 12 months).

### III.2.3. User Preferences & Comfort

**Goal:** Align automation with individual comfort levels and behavioral patterns.  
**Approach:**

* **Personalized Profiles:** Create user-specific temperature schedules and comfort zones.
* **Adaptive Learning:** Machine learning models learn preferred temperature patterns based on time, occupancy, and past adjustments.
* **Transparency & Control:** “Explainable automation” — display reasons for each system action (e.g., *“Preheated early due to cold forecast”*).
* **Manual Override:** Always allow user override with feedback loops to improve future model predictions.
* **Multi-User Negotiation:** Use weighted preference algorithms to balance conflicting household comfort needs.

### III.2.4. System Reliability & Robustness

**Goal:** Maintain consistent performance and fail-safe operation.  
**Approach:**

* **Edge Autonomy:** System continues basic heating control even when offline (local fallback mode).
* **Redundancy:** Duplicate critical sensors (temperature, humidity) and maintain hot-swappable components.
* **Health Monitoring:** Continuous diagnostics, automatic error reporting, and predictive maintenance alerts.
* **Fail-Safe Defaults:** In case of control logic failure, default to safe, comfortable temperature thresholds.
* **Performance Analytics:** Track uptime, energy savings, and response times through a backend monitoring dashboard.

### III.2.5. Summary Table

| **Domain** | **Core Strategy** | **Implementation Example** |
| --- | --- | --- |
| Security | Multi-layer protection | TLS, RBAC, signed OTA updates |
| Privacy | Data minimization & consent | Edge computation, GDPR tools |
| User Preferences | Adaptive AI with manual control | Personalized comfort profiles |
| Reliability | Redundant & autonomous systems | Edge fallback, predictive maintenance |

# My Contributions

## IV.1. Background and skills

### IV.1.1. System Architecture & Scalability

* **Experience leverage:** You can design a robust, modular software architecture for integrating sensors, forecast APIs, optimization algorithms, and user interfaces.
* **Contribution:**
  + Define data pipelines (e.g., MQTT → time-series DB → analytics).
  + Implement scalable microservices for weather and pricing data ingestion.
  + Ensure high availability and low-latency response for real-time control.

### IV.1.2. Data Engineering & Integration

* **Experience leverage:** Your background likely includes building APIs and handling large datasets — crucial for merging sensor, weather, and market data.
* **Contribution:**
  + Develop ETL workflows for continuous data streams.
  + Create a robust schema for telemetry storage (InfluxDB, TimescaleDB).
  + Handle data synchronization, validation, and error recovery.

### IV.1.3. Optimization & AI Implementation

* **Experience leverage:** Understanding algorithms and software design patterns will help translate control theory and predictive models into production code.
* **Contribution:**
  + Implement **Model Predictive Control (MPC)** or **Reinforcement Learning (RL)** modules for energy optimization.
  + Set up pipelines for continuous learning using live sensor data.
  + Integrate simulation tools (e.g., EnergyPlus or Modelica) for validation.

### IV.1.4. User Interface & Experience

* **Experience leverage:** With your background in front-end or full-stack development, you can design intuitive dashboards.
* **Contribution:**
  + Build responsive web/mobile dashboards for real-time monitoring, history, and user control.
  + Visualize comfort vs. cost tradeoffs with interactive graphs and sliders.
  + Ensure UX transparency — users should *understand* system actions and confidence levels.

### IV.1.5. DevOps & System Reliability

* **Experience leverage:** Your familiarity with CI/CD, testing, and monitoring can ensure reliability.
* **Contribution:**
  + Set up automated deployment pipelines (e.g., Docker + GitHub Actions).
  + Implement observability (Grafana, Prometheus) to track system health.
  + Handle versioning and rollback for model or firmware updates.

### IV.1.6. Data Security & Compliance

* **Experience leverage:** Secure software practices are vital for a connected IoT system handling personal data.
* **Contribution:**
  + Implement secure communication (TLS, OAuth2, encrypted MQTT).
  + Ensure GDPR compliance and anonymized telemetry.
  + Design access controls for multi-user or multi-building deployments.

### IV.1.7. Leadership & Collaboration

* **Experience leverage:** A decade of experience often includes mentoring, code reviews, and project management.
* **Contribution:**
  + Lead cross-disciplinary collaboration between mechanical engineers, data scientists, and UX designers.
  + Establish coding standards and design guidelines.
  + Manage agile sprints to iteratively deliver usable prototypes.

### IV.1.8. Summary Table

| **Area** | **Role** | **Example Contribution** |
| --- | --- | --- |
| **Architecture** | System design | Create modular, scalable backend |
| **Data Integration** | API & sensor fusion | Merge weather, price, and IoT data |
| **Optimization Logic** | Algorithm engineering | Implement MPC / ML control models |
| **Frontend/UI** | Dashboard development | Real-time visualization and control |
| **DevOps** | Reliability engineering | CI/CD, monitoring, error recovery |
| **Security** | Privacy & compliance | Secure data pipelines and access |
| **Domain knowledge** | Project oversight | Align cross-functional teams |

## IV.2 Additional ideas or improvements

### IV.2.1. **Smarter Control Algorithms**

#### a. **Hybrid Predictive Control**

* Combine **Model Predictive Control (MPC)** with **Reinforcement Learning (RL)**:
  + MPC handles short-term accuracy using thermal models.
  + RL continuously learns user patterns and adapts to real-world deviations.
* Benefit: Better adaptation to occupant behavior and changing thermal inertia.

#### b. **Adaptive Comfort Modeling**

* Move beyond fixed temperature setpoints.
* Integrate **adaptive comfort standards** (EN 15251 / ASHRAE 55) based on season, humidity, and user activity.
* Benefit: Reduces energy use while maintaining perceived comfort.

### IV.2.2. **Integration with Renewable and Smart Grid Systems**

#### a. **PV + Storage Coordination**

* If solar panels or batteries are available, coordinate heating cycles to use excess solar power for PCM charging.
* Benefit: Increases self-consumption and grid independence.

#### b. **Demand Response Optimization**

* Link the system with **dynamic electricity tariffs** and **utility signals** (e.g., flexibility markets).
* Automatically preheat floors when grid demand is low.
* Benefit: Cost savings and potential participation in smart grid incentives.

### IV.2.3. **Advanced Analytics & Insights**

#### a. **User-Facing Energy Insights**

* Provide users with weekly/monthly reports showing:
  + Energy cost savings vs. baseline.
  + Comfort score over time.
  + Environmental impact (CO₂ reduction estimate).
* Benefit: Improves transparency and engagement.

#### b. **Predictive Maintenance**

* Use machine learning to detect anomalies in sensor data or heating response (e.g., slow heat transfer = possible PCM degradation).
* Benefit: Reduces downtime and ensures long-term reliability.

### IV.2.4. **Digital Twin Simulation**

* Build a **digital twin** of the room/building that mirrors real conditions in simulation (using tools like EnergyPlus or Modelica).
* Run optimization scenarios virtually before applying them to the real system.
* Benefit: Safe testing environment and better parameter tuning.

### IV.2.5. **Enhanced User Interface Features**

* Personalized Comfort Profiles: Allow users to set comfort preferences (e.g., “eco mode,” “comfort first”) and the system automatically adjusts optimization priorities.
* Voice and Mobile Integration:
  + Integrate with smart assistants (e.g., Google Home, Alexa).
  + Provide push notifications (e.g., “Floor preheating now due to low electricity price tomorrow morning.”)

### IV.2.6. **System Security & Edge Computing**

* Edge AI Controller:
  + Run core optimization logic on an edge device (Raspberry Pi or industrial controller).
  + Benefit: Local autonomy during connectivity loss and reduced cloud dependency.
* Secure OTA Updates: Support encrypted over-the-air (OTA) firmware/software updates with rollback capability.

### IV.2.7. **Broader Ecosystem Integration**

* Building Management System (BMS) Compatibility: Use BACnet or Modbus for interoperability with existing systems.
* Community or District-Level Optimization: Extend logic to coordinate heating across multiple buildings or apartments, sharing excess stored heat.

### IV.2.7. **Research & Innovation Opportunities**

* **PCM Characterization:** Experiment with different PCM melting points for seasonal optimization.
* **AI Explainability:** Use interpretable ML models to justify heating decisions to end-users (“The system preheated due to forecasted low tariff and temperature drop”).
* **Lifecycle Assessment:** Quantify and visualize sustainability benefits.

### **IV.2.8. Summary Table**

| **Improvement Area** | **Proposal** | **Expected Benefit** |
| --- | --- | --- |
| **Control Algorithms** | Hybrid MPC + RL | More adaptive and efficient optimization |
| **Comfort Modeling** | Adaptive comfort standards | Energy saving without comfort loss |
| **Renewable Integration** | PV + demand response | Lower costs and emissions |
| **Analytics** | Digital twin + predictive maintenance | Better insights and reliability |
| **User Experience** | Smart notifications, profiles | Higher engagement and transparency |
| **Edge & Security** | Local AI + OTA updates | Resilience and trust |
| **Scalability** | Multi-building optimization | Larger-scale energy efficiency |