HomeOrbit: A Personalized and Adaptive IoT-Based Home Automation System

RM Shahriar Hoque, Mathematical and Computational Sciences Dept.
Advised by Prof. Sofia Visa

Project Overview

- HomeOrbit is a modular smart home system using IoT technology.
- Adapts automation based on user preferences and environmental conditions.
- Uses an ESP32 microcontroller with sensors for temperature, humidity, and light.
- Employs MQTT protocol for real-time communication.
- Controlled via a custom web interface, supporting both manual and automated control.Offers context-aware responses, avoiding rigid rule-based systems
- .Designed as a scalable, user-centric solution.

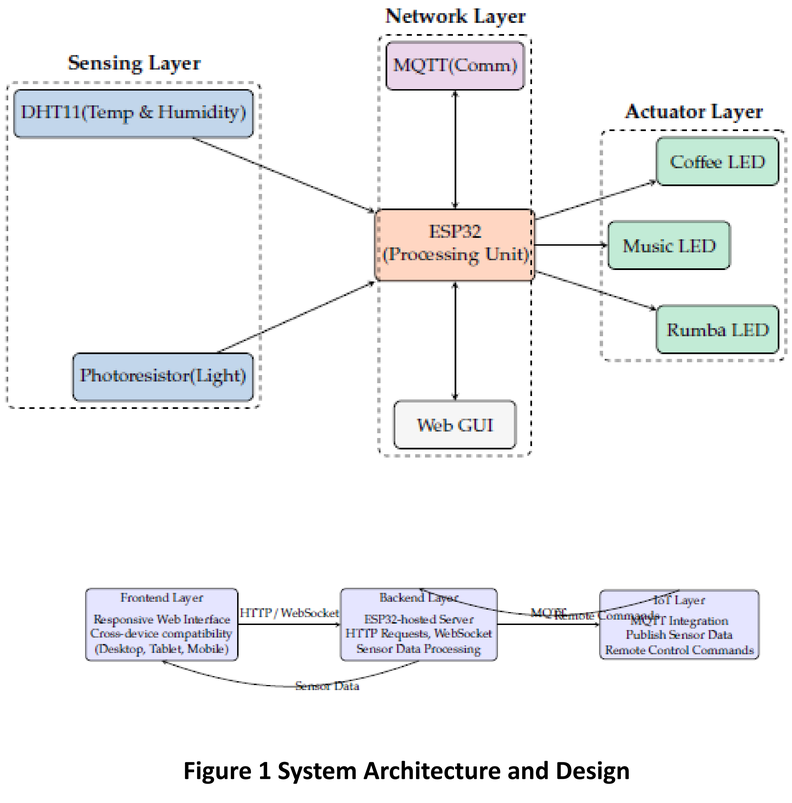

Figure 1 System Architecture and Design

- Successfully integrated hardware components and real-time software components to enable adaptive automation.
- LED indicators accurately respond to the sensor inputs and control logic.Key Outcomes:Scalable and modular design with room for expansion.
- Proof-of-concept validated through rigorous testing in controlled environments.

System Architecture

- **Hardware Components:**
  - *ESP32 Microcontroller:* Central processing unit handling sensor data and executing control logic.
  - *Sensors:* DHT11 for temperature/humidity and a photoresistor for ambient light.
  - *Actuators:* LED indicators representing different automation modes (Coffee, Music, Rumba).
- **Communication:**
  - *Protocol:* MQTT is used for efficient, real-time data exchange between the ESP32 and external systems.
- **Control Logic:**
  - Uses environmental thresholds to trigger automation modes:
    - **Coffee Mode:** Low temperature/humidity
    - **Music Mode:** Moderate ranges
    - **Rumba Mode:** High values
  - A fallback function ensures all LEDs are activated in edge cases.

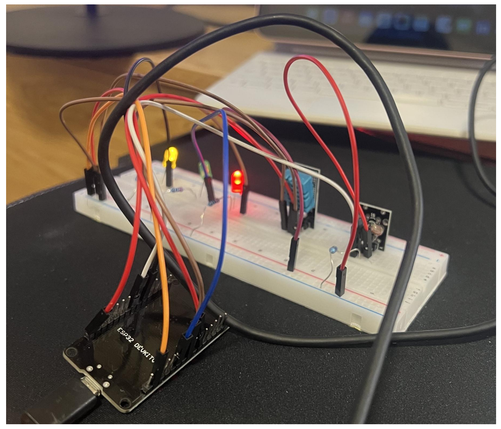

Figure 3. Circuit Configuration.

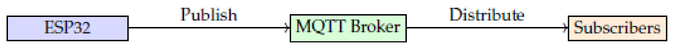

Figure 4. Establishing MQTT Communication.

Table 3.1: Environmental Thresholds for Operational Modes

Mode	Temperature (T)	Humidity (H)	Code Reference
Coffee	$T < 20°C$	$H < 50\%$	Line 11: `triggerCoffeeMode()`
Music	$20°C \leq T \leq 25°C$	$40\% \leq H \leq 60\%$	Line 14: `triggerMusicMode()`
Rumba	$T > 25°C$	$H > 60\%$	Line 16: `triggerRumbaMode()`
Fallback	¬(Above Conditions)		Line 18: `activateAllIndicators()`

Figure 5. Actuator Thresholds.

Web GUI + Real-Time Interaction

- ESP32-hosted GUI built with HTML/CSS/JavaScript.
- Real-time updates via WebSocket every 2 seconds.
- Manual override of automation modes via user-friendly dashboard.

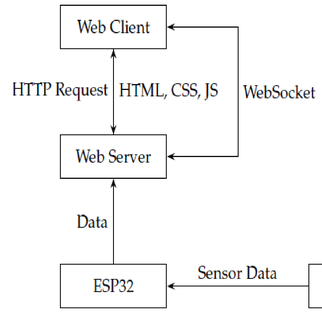

Figure 2. Screenshot of GUI and Web Communication Diagram.

Future Work and Expansion

- Integration of more sensors (e.g., motion, CO2).
- Use of ML for predictive automation.
- Compatibility with popular platforms (Google Home, Alexa).
- Enhancing user privacy and security.

References
Stojkoska & Trivodaliev – Review of IoT-based smart home architectures.Alaa et al. – Analysis of personalized home automation.Atzori et al. – Overview of IoT technologies and architectures.Sethi & Sarangi – Security and communication in IoT systems.