AI-Enabled Cloudless Home Ecosystem A Decentralized Architecture for Enhanced Security and Intelligent Control

Andrew Nitin Joseph[1], Naindeep Singh[2], KVS Balaji[3], Jasmitha JK[4], Mathew Jacob[5], Dr. Revathi V[6]

[12345] Computer Science and Engineering,Dayananda Sagar University Bangalore, India
[6] Associate Professor, Computer Science and Engineering, Dayananda Sagar University Bangalore, India

DOI: https://doie.org/10.10399/JBSE.2025811222

Abstract— This study suggests a decentralized, cloud infrastructure-independent home automation system, with local processing and decentralized architecture, aiming for security, privacy, and efficiency. With the use of a local server for automation task execution and data processing, the system reduces third-party cloud service reliance. The system is made secure with remote access through Tailscale VPN, using WireGuard encryption, to enable direct peer-to-peer communication between devices. Modular architecture with microservices allocates duties to automation control, NAS-based file processing, AI voice assistant, and real-time monitoring to ensure performance and capture faults. The local system's AI utilizes Llama 3.1 LLM model for voice commands and adaptive scheduling from past usage behavior. The user data is utilized by the AI module to suggest power-saving procedures, and the NAS server incorporated within offers secure local storage of personal and group files without recourse to external cloud storage solutions. Initial results show success in terms of low latency, consistent data processing, and increased security via local data processing and encrypted communication protocols. The provided solution effectively addresses the risks of centralized cloud-based designs while offering an efficient and scalable solution for future-proof smart homes in general. In conclusion, this research illustrates the practical benefits of combining high-level local artificial intelligence capability with a secure, decentralized home automation system.

Keywords--- Cloudless, Decentralized, Local Server, VPN, Home Automation, NAS Server, Ollama, Llama 3.1, AI Voice Assistant and pattern recognition.

I. INTRODUCTION

The widespread adoption of cloud-based infrastructure has revolutionized modern technology, but at the cost of user privacy and data security. Traditional smart home systems employ cloud services to process and store data, thus exposing private data to external networks and possible security breaches. The AI-Enabled Cloudless Home Ecosystem contradicts this dependency by suggesting a secure, efficient, and completely cloud-free intelligent home solution. Supported on the Cloudless Home Automation System, this project integrates AI-based automation without compromising on a decentralized framework [1], with the users in full control of their data and privacy without compromising on intelligent functionality.

High-profile breaches, like the [2] Eufy security camera feed hack, underscore the dangers of relying on cloud-based technology. The AI-Enabled Cloudless Home Ecosystem [4] avoids these risks by running exclusively on a local server, keeping all data in the user's domain. Additional security [5] and remote access are handled by the use of Tailscale, a peer-to-peer VPN solution offering end-to-end encrypted, direct device links without data being sent to third-party cloud providers. This not only prevents external risk but also data sovereignty, allowing a secure, transparent, and fully user-controllable smart home experience [8].

Besides security and privacy, the system also brings AI-powered automation and a NAS server [10] within for secure personal file sharing and storage. Integration of artificial intelligence [12] offers management of intelligent devices, default scheduling, and dynamic energy management depending on real-time and historical usage patterns. With user control, efficiency, and security as top priorities, the project enhances smart home technology through integration of modern features without compromising on users' control of data and automation [11].

II. STATE OF THE ART ON RELATED SURVEYS

Design and implementation of a stand-alone storage system on the Avi RPi3B board is presented in research on low-power, energy-efficient NAS systems. The study identifies the key features such as system autonomy, static IP address,

RAID backup, Samba server integration, support for VPN, and Dynamic DNS, with secure remote access [9]. Performance testing identifies read speeds of up to 88 Mbps, comparable to commercial NAS systems, although the write speed is a little lower at 34 Mbps. Nevertheless, the proposed system reduces energy usage to as little as 6.25 W, down from commercial system usage levels of 16 W. This performance-to-power ratio renders the system a low-cost, portable system, especially for domestic use.

An IoT-based home automation [6] research investigates pattern recognition to enhance user convenience and security. The system is built o the Arduino Mega 2560 microcontroller and employs an array of sensors for sensing the environment, including temperature, light, and motion sensors, and actuators such as relays, buzzers, and glucose monitors. Wireless communication is provided by the ESP8266 module, allowing users to remotely monitor and control appliances using a web-based platform.

Prominent features of the system include automatic appliance control as per observed user patterns, remote real-time monitoring, and enhanced security through motion notification. The research indicates that the system efficiently automates home appliances, learns user habits, and offers timely security alerts [13]. Energy efficient, secure, and user-friendly, this solution is of particular advantage to the elderly and disabled. Its scalability, however, is a drawback in large, multi-system installations.

An analysis of SAGE (Smart Automation and Grounded Execution), [3] an intelligent home agent based on a large language model (LLM), discusses its application to home automation via an action-based paradigm. It supports device interactions through APIs, automates command execution, and offers context-aware automation using previously executed user interactions.

Perhaps the most significant capability of SAGE is its advanced device disambiguation feature, which allows it to interpret natural references like images and to run adaptive, condition-dependent routines defined in Python syntax. Empirical testing over 50 intricate tasks demonstrated considerable effectiveness, with the GPT-4-powered SAGE achieving a success rate of 75%, so far surpassing foundation models like One Prompt (30%), Alexa, and Google Assistant.

The study identifies SAGE's ability to synthesize heterogeneous sources of information, handle complex user interactions, and dynamically control smart home devices. Executing within a decentralized architecture, SAGE is user-centric and privacy-conscious in its automation, positioning it as a strong competitor to voice assistants based on cloud infrastructure.[4]

MQTT-based IoT automation is investigated in studies on its integration with OpenHAB for enabling smart device communication and control. MQTT, a lightweight machine-to-machine messaging protocol, is publish-subscribe and enables real-time data transfer, low-latency communication, and enhanced device interoperability. Open-source automation platforms, Home Assistant and OpenHAB, were compared in the study, implemented on Raspberry Pi and Arduino, and the performance of their remote device control and automation was compared.

The study indicates that MQTT-based automation [7] is best ideally suited to periodic and event-driven applications and thus to accessibility-oriented applications such as assistance to the disabled. The study also indicates potential risks associated with cloud-based MQTT brokers, particularly remotely accessed, and poses privacy and security risks. These findings support the necessity of local server solutions, following the model of the AI-Enabled Cloudless Home Ecosystem, seeking secure, decentralized device communication to prevent cloud dependency and external security risks while ensuring seamless automation.

III. PROPOSED DESIGN

The AI-Enabled Cloudless Home Ecosystem is rooted in the Cloudless Home Automation System (as depicted in Fig. 3), which is structured on three basic elements: the application (frontend), the local home server, and the switchboxes.

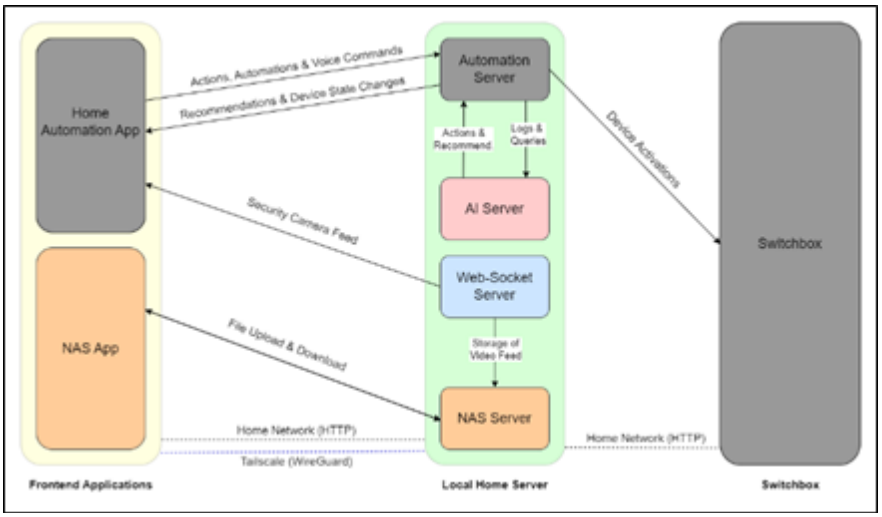

Fig. 3 Cloudless Home Ecosystem Architecture

The modular architecture allows for smooth operation and proper communication among all devices. The application is the central user interface, providing users with control over smart devices, access to files within the NAS, and setting schedules or automation presets. For secure and private communication, the application is linked to the local home server via Tailscale VPN, thus providing secure remote access even when users are away from home.

The local home server is the core of the system, managing attached devices and enabling backend automation by coordinating actions and data transfer to the application. For improving operational efficiency, the server is divided into multiple concurrent microservices based on Flask, the Automation Server, NAS Server, AI Voice Assistance Server, and WebSocket Server. The switchboxes are the hardware interface, performing actions like turning devices on or off and fan speed control. Every switchbox in the server is given a unique identifier to allow proper control and integration with the automation framework.

All the server modules run on individual ports and independently operate, thereby making communication effective, minimizing latency, and having high system performance. The decentralized approach allows every server to dedicate itself to its own responsibility without interfering with operations elsewhere in the system. User commands are governed by the automation server in sync with set-up automation protocols, while the AI server analyzes usage patterns of the devices to come up with smart suggestions for driving energy efficiency and tailored automation solutions. In all, the servers make the system learn users' habits continuously and optimize device control. Meanwhile, the NAS server provides secure storage and access to the files with both local and remote access to personal data independent of cloud resources. The WebSocket server transmits the stream from the security cameras in real-time to the application, thereby making it possible to have timely delivery of the video stream. By hosting these components under a decentralized framework, the system achieves maximum privacy, reduces external security threats, and gives users maximum control over data and automation capabilities.

IV. PROPOSED METHODOLOGY

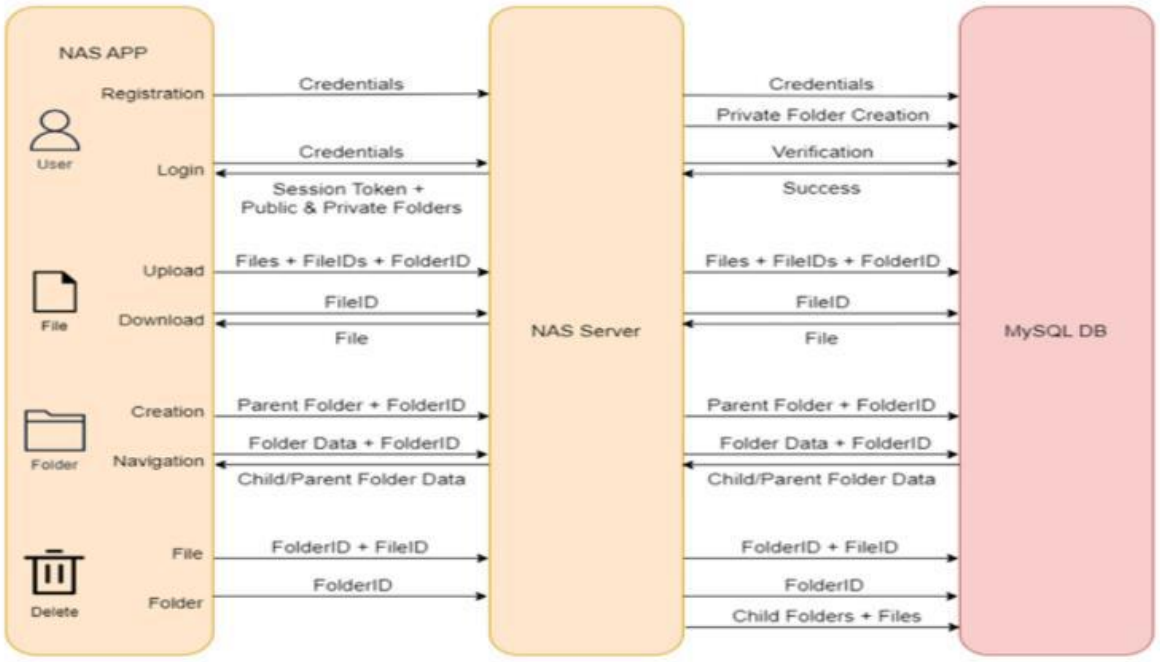

Fig. 4.1 NAS Server Architecture and Request Flowchart

The NAS server architecture, as depicted in Fig. 4.1, is the core component for personal file data storage and secure data exchange in the AI-Enabled Cloudless Home Ecosystem. It ensures seamless connectivity by communicating with the application frontend, home local server, and switchboxes and thereby improving data management efficiency. Users connect to the NAS utility through a mobile application built using the Flutter framework, enabling them to easily manage accounts, upload and download files, and classify data as personal or shared folders.

A Flask-based API manages files and user authentication securely and efficiently. Files are linearly stored on the server but hierarchically displayed to facilitate easier usability. Users are able to upload files with metadata, create a folder, preview, download, and delete content. Security is maintained through hashed credentials and session tokens that have a limited lifespan, with unauthorized access prevention. A MySQL database stores user accounts, folder hierarchies, and file metadata, and logs details for traceability and accountability. For secure connection, the NAS server utilizes Tailscale, a VPN-based solution that grants the local home server a unique local IPv4 address and a public endpoint. This setup allows remote access without the utilization of cloud infrastructure, ensuring all data are private and under the full control of the user. Through the application of end-to-end encryption, Tailscale increases network security, preventing unauthorized access while providing continuous and reliable communication within the system.

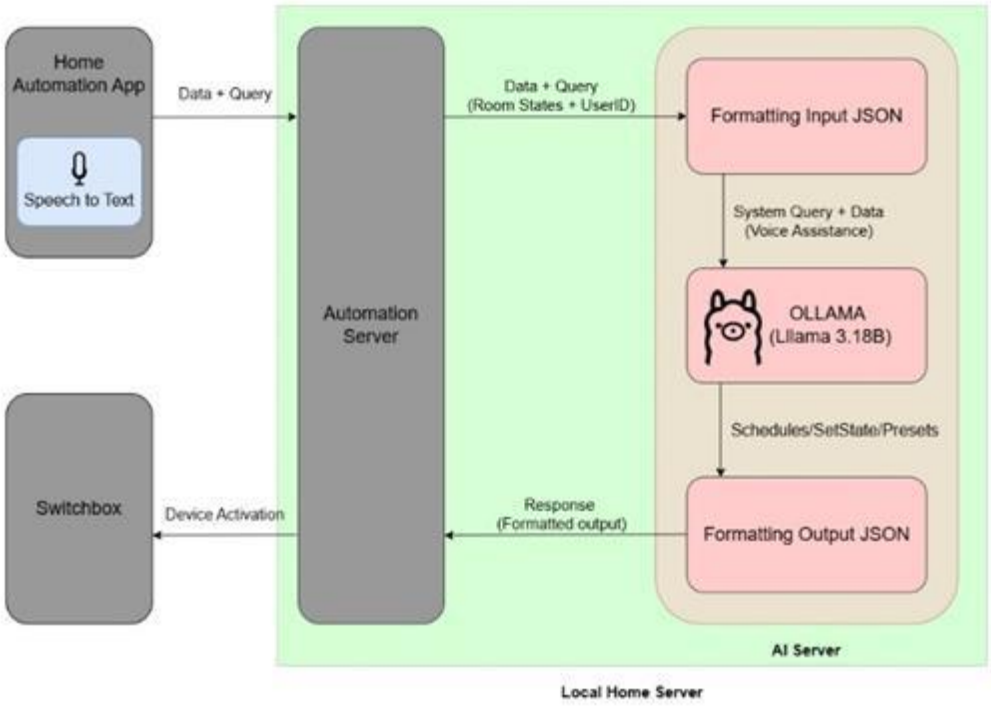

Fig. 4.2 AI Voice Assistance System Flow

The core of home automation technology is in an automation server, which runs in parallel with artificial intelligence services to drive overall system intelligence. The AI voice assistant server software stack, as shown in Fig. 4.2, is an intermediary between user commands and the automation backend to enable communication in the form of Flask-based application programming interface (API) calls. It also serves as a data pipeline between the application frontend and the Ollama AI server, thus enabling smooth request processing in natural language. Upon voice command, the application sends user information along with the query to the automation server. Before sending the request to the AI server for processing, the automation server adds context about the room and devices, such as switchbox identification numbers, current operational states, and related locations, as well as sensor data. For correct intent recognition and query processing, the system uses distributed prompt engineering, which enables domain-specific and context-aware interaction with the AI. The server also imposes a strict input-output format with JSON syntax, which enables accurate and structured communication between the AI and automation servers, ensuring that all commands are actionable, unambiguous, and are executed without failures.

The query processor processes user commands and determines actions that devices need to perform, e.g., turning devices ON/OFF, fan speed adjustments, or setting automations (presets and schedules). It identifies distinct room and switchbox names and assigns them corresponding devices. For instance, the command "*Turn on living room lights*" produces a JSON output that adjusts the living room lights alone, and the command "*Set the bedroom fan to speed 3*" adjusts the fan speed to speed 3 in the bedroom alone. The system updates the device states and sends the JSON command to the automation server to execute, thus offering accurate, effective, and reliable automation.

The system incorporates a robust error-handling mechanism that is designed to rectify unclear user commands, thus enhancing accuracy as well as user experience. For instance, when a user gives a command, "*Turn on the lights*," the system will seek clarification by asking which particular room the user is looking to light up. This step ensures that operations are carried out exactly as intended, thus minimizing errors. There is seamless communication between the artificial intelligence server and the automation server in the system, where the AI output is converted into structured JSON commands. This structured process allows the system to properly interpret and carry out user commands as well as ensure all device operations are carried out reliably in the smart home environment.

Enhancing AI-driven automation entails enhancing the JSON input format for better natural language processing. One of the enhancements is a middleware solution that simplifies device command processing without adding backend complexity. A completely context-aware smart home would not only execute commands but also understand intent as

a whole. For example, a request made by a user (John for example) "*I'm in my bedroom and it feels hot in here*" would be instantly recognized to turn on the fan to speed 2 in the bedroom named after the user (e.g., John's bedroom) and executed. By integrating advanced contextual awareness, the system evolves from a basic automation tool into an intelligent assistant, making interactions more seamless, intuitive, and natural, by leveraging best in class intent recognition by Llama model.

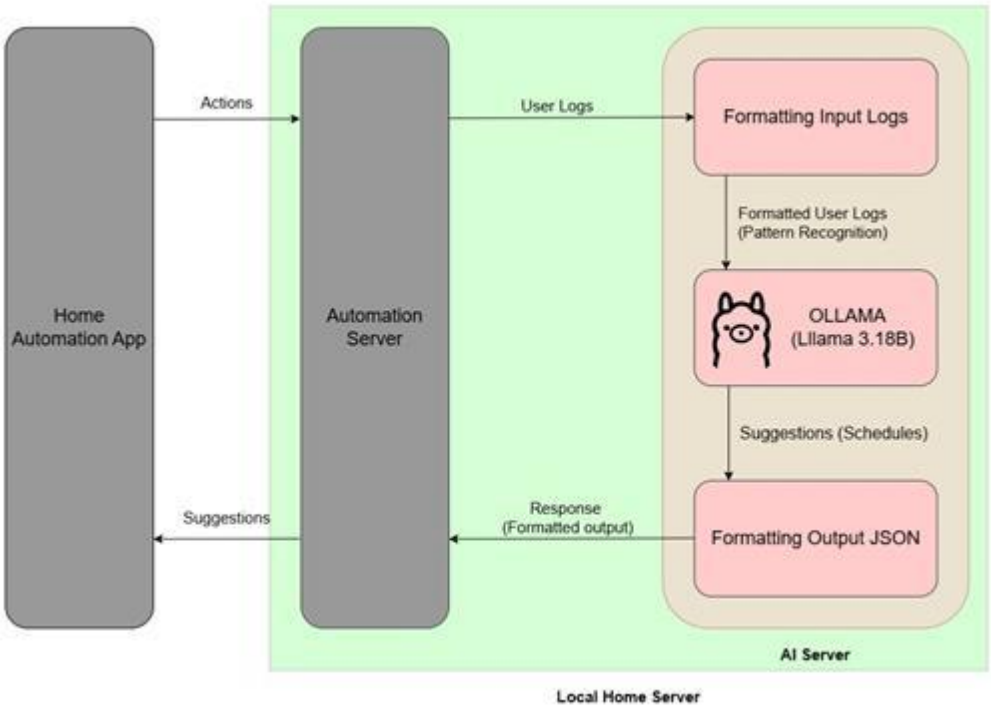

Fig. *4.3. AI Pattern Recognition System Flow*

The AI Pattern Recognition Server architecture (as depicted in Fig. 4.3) scans through logs generated by the automation server to analyze user behavior and recommend optimized scheduling of smart devices. This useful information is structured in JSON format, containing details such as the room name, switchbox ID, device status, and timestamps for each recorded action. User behavior may be initiated by the mobile application, voice commands, or direct physical interaction with switchboxes, leading to the creation of a comprehensive usage profile. Nevertheless, the raw usage logs per se do not unmistakably indicate the unique behavioral patterns adopted by individual users. For this deficiency, the AI server's system prompt is modified to facilitate to interpretate and analyze these logs intelligently. Through monitoring repeated activities, the AI detects patterns in device usage and creates a structured JSON feed to recommend optimized scheduling. For example, if the system detects that lights in one room are repeatedly switched on at 6 PM and switched off at 10 PM, it will recommend an automated scheduling routine to streamline maximizing energy efficiency and promoting automation.

The JSON output is generated in accordance with the automation server's specifications and includes essential details such as switchbox ID, room name, scheduling time, and recommended state changes. This information is efficiently integrated into the system and displayed on the frontend as a suggestion for optimization. The users can browse, edit, or ignore AI-recommended recommendations, with full control over the automation setup. By avoiding redundant actions, the system improves efficiency, minimizes human intervention, and optimizes power usage. With time, the system learns and adjusts to the user's schedule, becoming an intelligent, personal home automation network that reacts flawlessly to individual desires.

V. EXPERIMENTATION

The NAS server is a central component of the AI-powered Cloudless Home Ecosystem, providing secure storage, management, and access to files in a local network. Personal and collaborative use is supported, with the NAS server providing registration, login, and file management with a minimalistic UI. Users can access the NAS server via a private IP address in the network or a Tailscale public IP address for remote access, with the app automatically determining the server endpoint for smooth setup. Files are divided into public and private folders, with the public folder supporting collaborative sharing and the private folder providing limited access for data protection. As illustrated in Fig. 5.1, the system provides a user-friendly interface to manage files, including low-resolution preview caching for improved performance.

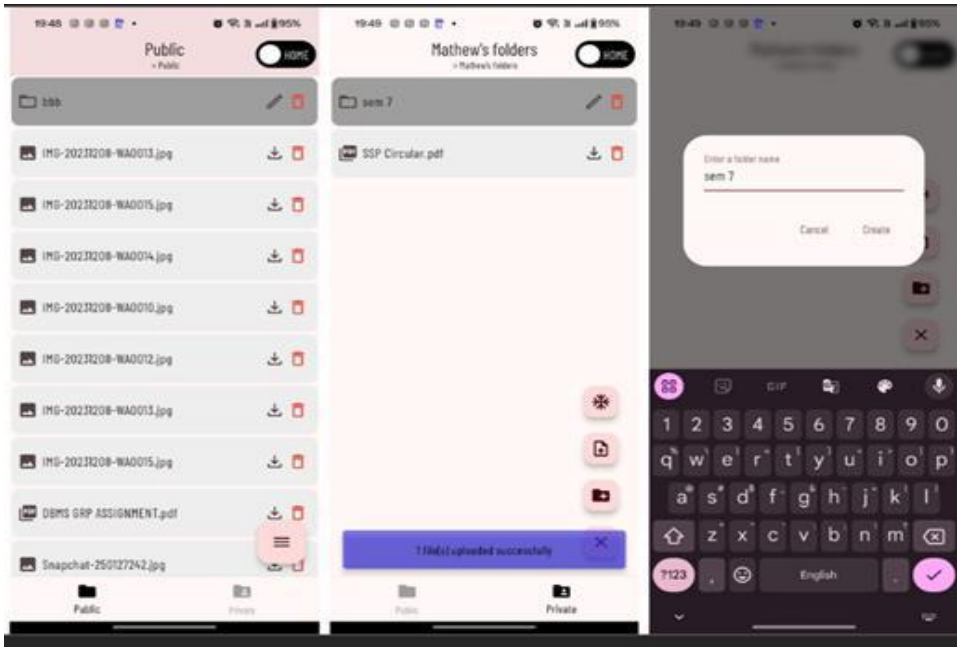

Fig. 5.1. Network Detection and Connection Establishment using Tailscale

The system has an AI Voice Assistant powered by Llama 3.1 8B for natural language processing to understand and execute home automation instructions. Using Ollama, the assistant receives raw JSON input, such as switchbox IDs, device statuses, room names, and user commands, and processes the data to generate proper automation instructions. When a user requests an action, for example, turning off lights, the AI understands the command and generates the proper response. While the model can function with raw inputs, it is complicated to process numeric IDs (e.g., "*4LLLL20231224020202*"). For accuracy, pre-processing techniques are used to convert such IDs to more descriptive labels (e.g., "*switchbox-1*"). Currently, the system is able to change device states, create automations according to user schedules.

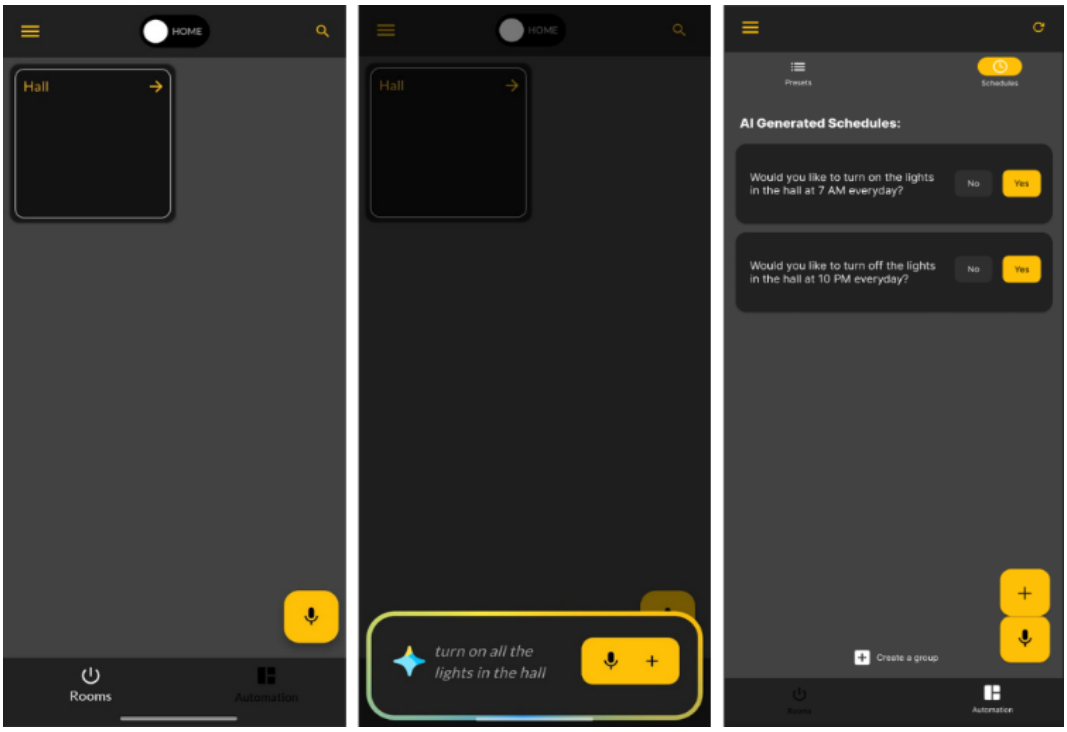

Fig. 5.2. AI Voice Assistant and AI-Generated Schedule System

As illustrated in Fig.5.2, the diagram indicates the operational working of the AI Voice Assistant within the smart home environment. Upon the release of a voice command by a user, e.g., "Turn on all the hall lights," the system captures and processes the command. The input is relayed to the Llama model, which translates the command and converts it to a structured JSON format with parameters such as device type, target room, and intended action. The resulting JSON output is sent to the automation server, which then executes the command by changing the state of the target devices. This effortless interaction provides smart and efficient automation, enabling users to manage their home environment comfortably using natural language commands.

In parallel to these functionalities, the ecosystem contains an AI-driven Scheduling System powered by Llama 3.1 8B. The system analyses past device usage patterns to create automated schedules for light and fan control. The procedure starts with processing historical JSON data containing essential attributes like room ID, switchbox ID, state, date, and time. A structured system prompt defines rules for generating distinct schedule IDs from timestamps while maintaining valid time formatting for the scheduled events. The schedules produced by the AI are presented in the form of proposals on the scheduling interface of the application (as seen in Fig. 5.2), thereby presenting users with automatic tasks whose execution needs approval by the users. For instance, when the system suggests, "Would you like to turn on the hall lights at 7 AM?" and the user approves by choosing "Yes," the action is scheduled and executed at the specified time. On the other hand, when the user refuses by clicking "No," the suggested schedule is discarded. The interactive approach thus maintains automation in a user-controllable manner while making use of AI-generated suggestions to enhance the efficiency of scheduling. Additionally, the system suggests certain devices based on past usage behavior, thus conserving energy and improving the working efficiency. For processing multi-day historical data efficiently, two meaningful optimizations have been achieved, making it possible for users to formulate schedules in an efficient and accurate manner.

VI. ANALYSIS AND RESULTS

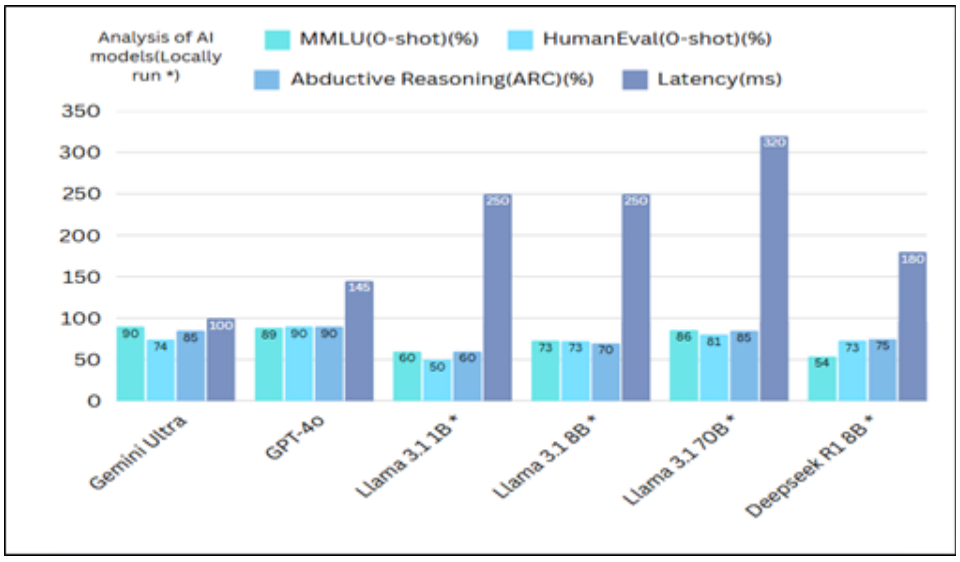

Fig. 6.1. Comparison of LLM models based on Intent Recognition, Reasoning, Latency and Efficiency Parameters

The Llama 3.1 8B test suite includes MMLU (Massive Multitask Language Understanding), HumanEval (a benchmark for code generation), ARC (the Abductive Reasoning Corpus), and latency. MMLU evaluates language understanding across a broad set of topics, which is critical for properly interpreting advanced user requests for home automation. HumanEval checks the ability of a model to generate correct code for automation tasks, thus guaranteeing safe backend logic. ARC checks abductive reasoning ability, which allows handling uncertain or atypical queries. Latency checks the input-to-output time delay, the response time being critical to guaranteeing user satisfaction.

These metrics are essential for the project because they ensure that Llama 3.1 8B can effectively understand user intent, generate accurate code for automation tasks, and respond quickly. The model achieves a 73.0% score on MMLU and 72.6% on HumanEval (0-shot pass@1), demonstrating robust general-purpose understanding and code generation capabilities. While exact ARC scores aren't available, Llama 3.1 8B's performance (as depicted in Fig. 6.1) in similar reasoning benchmarks suggests strong abductive reasoning capabilities. Its ~50 ms latency ensures near-instant responses, making it ideal for real-time home automation applications.

Targeting these particular metrics enables the project to take advantage of the balanced performance, efficiency, and privacy feature set of Llama 3.1 8B. The ability of the model to run locally with low latency and moderate resource consumption breaks cloud infrastructures' dependence, thus facilitating privacy and security of sensitive home automation processes. While smaller in size than large models like Llama 3.1 70B, Llama 3.1 8B is competitive for code generation and intent understanding and is an inexpensive and efficient solution for local deployment. This package of metrics facilitates easy choice of Llama 3.1 8B as a practical model for the home automation system.

Attribute	Tailscale	Cloud-Based VPN's	Traditional VPN's
Privacy	Peer-to-peer; no data passes through Tailscale servers	Data routed through centralized servers	Data routed through centralized servers
Encryption Protocol	WireGuard (state-of-the-art)	Varies (often TLS-based)	IPSec or OpenVPN (slower)
Latency (local)	~10-50 ms (direct peer-to-peer)	~100-300 ms (depends on server proximity)	~200-400 ms (centralized bottlenecks)
Setup Complexity	Zero-config, easy deployment	Moderate; requires cloud configuration	High; requires firewall and port setup
Security Features	Zero Trust; granular access control, end-to-end encryption	Varies; often lacks Zero Trust architecture	Basic encryption; limited access control
Scalability	Highly scalable (peer-to-peer mesh)	Limited by server capacity	Limited by central concentrator bottlenecks

Table. 6.2. Table Comparing Tailscale Against Traditional Cloud and VPN Based Systems

Tailscale offers a private, secure networking solution that links the home automation devices via peer-to-peer connections, and data never crosses centralized servers. It uses WireGuard, a cutting-edge encryption protocol that is more secure with lower latency (~10-50 ms for direct peer-to-peer connections). This peer-to-peer design outperforms cloud-based VPNs (~100-300 ms) and traditional VPNs (~200-400 ms), (as shown in Table. 6.1) which is perfect for real-time home automation applications where responsiveness is paramount.

Tailscale offers an easy and secure home automation network solution by establishing direct peer-to-peer connections under WireGuard encryption, thus keeping data private and avoiding centralized servers. This approach overcomes the complexity associated with traditional VPN setups, offers robust security through a Zero Trust model with granular access control, enables low latency for real-time response, and offers stunning scalability to accommodate more devices – all while remaining cloud-based centralized processing-free.

File Size (GB)	Google Drive		NAS (Home)		NAS (Tailscale)	
	Upload (sec)	Download (sec)	Upload (sec)	Download (sec)	Upload (sec)	Download (sec)
1	102	60	72	65	106	69
2	203	120	145	131	212	138
3	305	180	217	196	318	208
5	508	300	362	327	531	346
10	1015	600	723	654	1062	692

Table. 6.3. Table Comparing our NAS server against Cloud Based Storage Solution – Google Drive

This is a comparison between a locally installed Network Attached Storage (NAS) server, both on the local network ("NAS Home") and remotely accessed via Tailscale ("NAS Tailscale"), and Google Drive (as shown in Table. 6.2) for storing and retrieving data.

The primary advantage of a local NAS system is that it returns data ownership and control to the users, so they get to have full control of their data and hence meet stringent data privacy requirements. Financially, the initial cost of purchasing a NAS (for example, ₹10,000 for a 4TB device) is comparatively less than the recurring subscription cost of Google Drive. In addition, experiments on speed testing reveal that data retrieval is faster while accessing the NAS over a local network. However, if remote access via Tailscale is compared with Google Drive, upload and download speeds show some compromises. For a 1GB file, Google Drive shows faster upload and download times compared to "NAS Tailscale"; but for big files, differences in speed are marginal. Specifically, the upload time to Google Drive is around 102 seconds, while NAS (Tailscale) takes around 106 seconds. Note that network bandwidth and server load influence these performance numbers significantly.

VII. CONCLUSION

In summary, the project illustrates the feasibility and strengths of a home automation solution using a decentralized and cloud-free process. Utilizing local server administration, Tailscale VPN to encrypt connections, and AI support facilitated by Llama 3.1, the system appropriately protects user anonymity while providing seamless and effective automation capabilities. The architected system successfully decentralizes major functions, such as AI voice services and NAS storage, thus securing sensitive information stored under user control while offering next-generation automation functionality.

This approach avoids common susceptibilities of central cloud setups and is consistent with new directions in data sovereignty, edge computing, and human-oriented design. The modular design of the system further enables seamless incorporation of additional functions, thus ensuring future evolution of smart home solutions. In future research activities, focus will be placed on further optimizing artificial intelligence algorithms towards achieving reduced latency, system scalability, and an increase in the number of supported smart devices that can be accessed. Moreover, extended empirical evaluations in many domestic settings will provide more detailed user behavior data and the system robustness under many different operating environments. Overall, this work sets a good foundation for future home automation systems based on their security, efficiency, and responsiveness towards changing needs of modern living.

VIII. REFERENCES

[1] Vaithiyanathan, Revathi, and K. Ranjini. "Paradigm shift from machine learning to federated learning." Green Machine Learning and Big Data for Smart Grids. Elsevier, 2025. 133-146.

[2] Sean Hollister. 2023. Anker finally comes clean about its Eufy security cameras. The Verge. https://www.theverge.com/23573362/anker-eufy-security-camera-answers-encryption

[3] Rivkin, D., Hogan, F., Feriani, A., Konar, A., Sigal, A., Liu, S., & Dudek, G. (2023). SAGE: smart home agent with grounded execution. arXiv preprint arXiv:2311.00772.

[4] Rodriguez-Garcia, P., Li, Y., Lopez-Lopez, D., & Juan, A. A. (2023). Strategic decision making in smart home ecosystems: A review on the use of artificial intelligence and Internet of things. Internet of Things, 22, 100772

[5] Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., ... & Scialom,T. (2023). Llama 2: Open foundation and fine-tuned chat models. arXiv preprint arXiv:2307.09288.

[6] Zakhary S, Lodge T, McAuley D. Performance Evaluation for Privacy-preservingControl of Domestic IoT Devices. arXiv preprint arXiv:2207.08482. 2022 Jul 18.

[7] Saxena, Srijan, et al. "Implications of MQTT connectivity protocol for IoT based device automation using home assistant and OpenHAB." 2019 6th International Conference on Computing for Sustainable Global Development (INDIACom). IEEE, 2019.

[8] Iyer, R., & Sharma, A. (2019). IoT based home automation system with pattern recognition. International Journal of Recent Technology and Engineering, 8(2), 3925-3929.

[9] Lanka, A., & Gargeyas, A. (2018, April). Remotely accessible, low power network attached storage device. In 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT) (pp.1083-1088). IEEE.

[10] Shrivastava, A. R., & Gadge, J. (2017). Home server and NAS using Raspberry Pi. 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI) (pp. 2270–2275). IEEE.

[11] Jabbar, Zaid Abdulzahra, and Rameshwar Shankarrao Kawitkar. "Implementation of smart home control by using low cost arduino & android design." International Journal of Advanced Research in Computer and Communication Engineering 5.2 (2016): 248-256..

[12] Vaithiyanathan, Revathi, and Tholkappia Arasu Govindharajan. "User preference-based automatic orchestration of web services using a multi-agent." Computers & Electrical Engineering 45 (2015): 68-76.

[13] A Younis, Younis & Kifayat, Kashif & Merabti, Madjid. (2014). Cloud Computing Security & Privacy Challenges. 10.13140/2.1.1779.6809.