Faculty of Computers and Data Science
Program of Computing and Data Science
Alexandria University

Graduation Project I
Leap: A Smart Ring for Smart Homes

Submitted by:

Ayatullah Nabil El-Sayed El-Banhawy - 20221452375

Aysel Mahmoud Hassan Mohamed El-Masry - 20221465749

Habiba Mohammad Attia Mohammad - 20221379966

Abd El-Rahman Ahmed Mohamed Ibrahim - 20221460100

Abd El-Rahman Mohamed Abd El-Fattah Sayed - 20221459795

Mohamed Hassan Abdullah Hassouna – 20221458620

Mohamed Mostafa Kamel Raslan Ouf - 20221459902

Muhannad Khaled Abdel Mohsen El-Khouly - 20221459941

Supervised by:

Dr. Magda Madbouly

Acknowledgment

Thanks God, because he's graced our lives with opportunities that we know are not of our hands or of any other hand. He's shown us that it's a scientific fact that gratitude reciprocates. This final moment doesn't just belong to one of us. We wouldn't be up here if it wasn't for some very important people in our lives. To the ones who stayed with us till the moment of our achievement, families, professors, and friends, after God for sure, we want to say

"Thank you all for your invaluable love, support and guidance."

And for sure our special thanks go to Dr. Magda Madbouly for her supervision and assistance in this project.

Abstract

In the past decade, smart home technology has undergone rapid transformation, changing the way of living in homes. It has brought intelligent solutions to increase convenience, accessibility, and security at home. Traditional systems are bound by challenges such as noisy environments that impede voice commands, the need for inclusive solutions for people with disabilities, and the lack of a cohesive platform integrating security, automation, and user assistance.

In response to these challenges, our project, *Leap: Smart Ring for Smart Homes*, gives way to a new approach in controlling smart homes with the integration of gesture recognition, facial recognition, and conversational AI all into one system. A key enabler in this solution is an advanced gesture recognition-enabled smart ring worn by the user that enables one to control any connected device around a home space using intuitive hand movements. Complementing this, the facial recognition system provides enhanced security by identifying authorized users and alerting homeowners of unauthorized entries.

The project also features a conversational AI chatbot, which lies at the heart of the state-of-the-art models of NLP. The chatbot is going to be a real-time assistant, answering questions from the users' text or speech, providing them with tailor-made suggestions, and simplifying daily interactions with the smart home system. All these features are merged into a single platform accessible via a website and a mobile app.

By adopting deep learning methodologies and IoT technologies, Leap is looking to redefine how people will interact with smart homes, bridging critical gaps in usability, inclusivity, and security. Our goal is to offer users a smarter, more accessible, and secure living environment and to set a new landmark in smart home innovation.

Table of Contents

Acknowledgment ii

Abstract iii

List of Abbreviations viii

Chapter 1 Introduction 1

1.1 Problem Statement 1

1.2 Motivation and Justification 1

1.3 Goals and Objectives 2

Chapter 2: Literature Review 4

2.1 Gesture Recognition in Smart Home Systems 4

2.2 Facial Recognition for Home Security 5

2.3 Chatbots for Smart Home Assistance 6

2.3.1 Model Selection 7

2.3.2 Why Chavinlo/alpaca-native Model? 7

2.3.3 Challenges in Chatbot Development 8

2.3.4 Text-to-Speech and Speech Recognition in Conversational AI 8

2.5 Integration of Smart Home Technologies 9

2.5.1 System Integration Challenges 9

2.5.2 Prior Work in Smart Home Integration 9

2.6 Gaps in Existing Systems 10

Chapter 3: System Design and Methodology 11

3.1 Overview of the System Architecture 11

3.2 Gesture Recognition System 12

3.3 Facial Recognition System 13

3.4 Chatbot System 13

3.5 Datasets 14

3.6 Data Preprocessing 14

3.7 Choosing Learning Algorithm 15

3.8 Data Modeling 16

3.8.1 RCE NN and Dynamic Time Warping for Gesture Recognition 16

3.8.2 Siamese Network for Facial Recognition 17

3.8.3 Chatbot Model Using Hugging Face Transformers 18

3.8.4 Integration of Gesture, Facial Recognition, and Chatbot 18

3.9 Toolset Overview 18

Chapter 4: Development 19

4.1 System Requirements 19

4.1.1 Functional Requirements 19

4.1.2 Non-Functional Requirements 20

4.2 Software Model: Agile Methodology 21

4.2.1 Introduction to Agile Methodology 21

4.2.2 Features of Agile Methodology 21

4.2.3 Suitability of Agile to The Leap Project 21

4.2.4 Phases of Agile Methodology for The Leap Project 22

4.2.5 Methodology 23

4.3 Website 24

4.3.1.2 User Interface and User Experience (UI/UX) 24

4.3.1.3 UI for User Information Form Page 27

4.4 Mobile Application 30

4.4.1 Platform Selection 30

4.4.2 App Development 30

4.5 Databases 32

4.5.1 Database Design 32

Chapter 5: IoT Devices 37

5.1 Smart Rings 37

5.1.1 Hardware Components and Justification 38

5.2 Smart Band 38

5.2.1 Use Case: 38

5.2.2 Hardware Components and Justification 39

5.2.3 Software 39
5.3 Smart Switch 40
5.3.1 Use Case 40
5.3.2 Hardware Components and Justification 40
5.3.3 Software 41
5.4 Live Stream Camera 41
5.4.1 Use Case 41
5.4.2 Hardware Components 41
5.4.3 Software Setup 42
Chapter 6: Experimental Analysis 43
6.1 Risk Analysis 43
6.2 Prompt Engineering Challenges 45
Chapter 7: Conclusion and Future Work 46
7.1 Conclusion 46
7.2 Future Work 47
7.3 Broader Implications 48
7.4 Final Thoughts 48
References .... **Error! Bookmark not defined.**
8.1 Online Resources and Tools 48
8.2 Datasets 49
8.3 Academic Journals and books 50

List of Figures

Figure 2.3.1: Rule-based chatbots 6
Figure 3.1: Workflow Diagram of the Leap Ring 12
Figure 3.2: Workflow Diagram of the Leap Facial Recognition System................................ 13
Figure 3.3: Workflow Diagram of the Leap Chatbot System.. 13
Figure 3.8.1 Structure of a RCE neural network…….. 16
Figure 3.8.2: A simple 2 hidden layer of Siamese network……... 17
Figure 4.2.4: Agile Phases 23
Figure 4.3.1.1: Landing Page Webpage……………………………………………………24
Figure 4.3.1.2: Contact Us Web Page 25
Figure 4.3.1.3: About Us Web Page 25
Figure 4.3.1.4: Profile Web Page 26
Figure 4.3.1.5.1: Login Web Page 26
Figure 4.3.1.5.2: Signup Web Page 27
Figure 4.3.1.6: User Information Web Page 27
Figure 4.3.1.7: Home Information Web Page 28
Figure 4.3.1.8.1: Livingroom Web Page 28
Figure 4.3.1.8.2: Kitchen Web Page 29
Figure 4.3.1.8.3: Bedroom Web Page 29
Figure 4.4.2.1: mobile UI………………………………………………………………….31
Figure 4.5.1.1: User Entity 32
Figure 4.5.1.2: Address Entity 33
Figure 4.5.1.3: Home Entity 33
Figure 4.5.1.4: Room Entity 34
Figure 4.5.1.5: Devices Entity 34
Figure 4.5.2.2: Fully Attributed Data model 36
Figure 5.1: ring circuit……………………………………………………………………37
Figure 5.2: Smart Band Circuit………………………………………..……………..38
Figure 5.3: Smart Switch circuit……………………………………..……………40
Figure 5.4: Live Stream Camera……………………………………..……………41

List of Abbreviations

Abbreviation	**Definition**
CNN	Convolutional Neural Network
LFW	Labelled Faces in the Wild
NLP	Natural Language Processing
IoT	Internet of Things
SER	Speech Emotion Recognition
ASR	Automatic Speech Recognition
ML	Machine Learning
DL	Deep Learning
AI	Artificial Intelligence
API	Application Programming Interface
UI	User Interface
UX	User Experience
ESP-NOW	Expressive Wireless Communication Protocol
SVM	Support Vector Machines
k-NN	k-Nearest Neighbors
IMU	Inertial Measurement Unit
OCR	Optical Character Recognition
SR	Speech Recognition
RCE	Restricted Coulomb Energy
DTW	Dynamic Time Wrapping
JS	JavaScript
TTS	Text To Speech
Wi-Fi	Wireless Fidelty
IoT	Internet of Things

OpenCV	Open-Source Computer Vision
LBP	Lower Boundary Point

Chapter 1

Introduction

1.1 Problem Statement

Today, most smart home systems are almost exclusively dependent on voice commands and mobile applications as their primary interface. These interfaces, effective for most users, create a big accessibility barrier for people with disabilities, such as speech impairments, or in noisy environments where voice recognition systems usually fail. Moreover, mobile applications need a smartphone and technical competence in using complicated interfaces, which could be an especially difficult task for an older adult or somebody with poor technological literacy.

Another major problem is the lack of interoperability and integration among different smart home systems. To be sure, most of the already existing solutions are built in silos; that is, their separate devices and applications work in isolation, governing different functionalities such as security, automation, and health monitoring. This leads to a fragmentation that causes inefficiencies, increases costs, and creates a poor user experience, making it hard for the users to manage their ecosystem in the smart home efficiently. This shows the requirement for an innovative solution able to address accessibility concerns, offer seamless interaction, and integrate diverse functionalities into one coherent system.

1.2 Motivation and Justification

The project is motivated by the requirement to design an inclusive, accessible, and user-friendly approach to smart home automation. It looks into the challenges posed by the current systems and seeks to provide a solution that combines innovation with practicality. Equipped with gesture recognition, this smart ring makes it possible for one to control his or her smart home devices most naturally and intuitively—especially for people who have difficulties with voice commands or even mobile interfaces.

The integrated facial recognition system also allows for added security in personalized access and sends instant alerts in the event of unauthorized entries. Meanwhile, the conversational AI-powered chatbot responds to the users with instant help, answers to questions, and personalized suggestions, hence enhancing the experience. To make all these features easily accessible and manageable, the project entails a website and mobile application as core platforms. These platforms serve as single-control dashboards where users can set up and monitor their devices easily, either at home or remotely.

1.3 Goals and Objectives

Leap is designed to redefine the field of smart home automation with the first integrated, user-centered system in merging gesture recognition, facial recognition, and conversational AI with IoT technologies. The ultimate goal is to provide a seamless, secure, and accessible solution, delivering a better experience in the interaction between users and their living environment.

The core part of this project will be the development of a smart Ring that can recognize and interpret hand gestures. This feature overcomes some of the drawbacks of using traditional control methods, such as voice commands and mobile apps, by offering an intuitive, hands-free alternative able to respond to a wide array of user needs, including persons with disabilities or being in noisy environments. With accuracy and responsiveness assured, this gesture recognition system will allow for the effortless control of smart home devices such as lighting, thermostats, and security systems.

This would further enhance home security with a state-of-the-art facial recognition system. Using TensorFlow, OpenCV, and Keras, this module is going to be able to identify authorized persons in real time and alert the homeowner about any unauthorized entry; it would adapt to changing lighting conditions in order to guarantee consistent performance. All these really make the smart home experience quite safer and much more tailored.

The conversation will be even richer with the aid of a conversational AI-driven chatbot that will offer live help, responding to queries and giving recommendations as needed. Available via both voice and text, this chatbot will become the easy-to-use interface for all smart home devices and support requests.

The Leap project will also have its mobile application and website, developed for the management of the system in a centralized way. It would be used in the control of devices, interacting with the chatbot, setting of security, and monitoring the system's performance. A strong focus will be put on making the platforms user-friendly, accessible, and compatible on a wide variety of devices to make smart home management easy for everyone.

The architecture of IoT will be the backbone in ensuring seamless communication between the smart Ring, facial recognition system, chatbot, and connected devices. Firebase will be utilized in achieving real-time data synchronization, while ESP-NOW and Wi-Fi will be used for low-latency communication. The architecture will provide reliable, scalable operation with the possibility of adding more devices and features to the system in the future.

The Leap project is uniting all these technologies with the view to achieve a holistic and adaptive solution able to handle changing demands from today's smart home for which there must be accessibility, security, and usability for all types of users.

Chapter 2

Literature Review

The **Leap project** builds upon a rich foundation of research and technological advancements in the fields of gesture recognition, facial recognition, chatbot development, and smart home automation. This chapter provides a comprehensive review of the existing literature, highlighting the key technologies, methodologies, and challenges that have shaped the development of the Leap system. By examining prior work, this chapter positions the Leap project within the broader context of smart home technologies and identifies the gaps that the project aims to address.

2.1 Gesture Recognition in Smart Home Systems

Gesture recognition plays a pivotal role in enhancing user interaction within smart home environments. Among the various techniques available, wearable sensors have emerged as a practical solution for capturing and interpreting gestures. Devices such as rings and wristbands equipped with inertial measurement units (IMUs) are widely used in this domain. These sensors can detect motion data, including acceleration and angular velocity, enabling real-time gesture recognition. The Leap project capitalizes on this technology by incorporating IMU sensors into wearable devices, ensuring intuitive and seamless gesture control for users [1].

Camera-based gesture recognition, another widely adopted approach, relies on optical devices and machine learning algorithms to identify user movements. Although this technique offers high accuracy, it often requires significant computational resources and well-lit environments for optimal performance. These limitations make camera-based systems less ideal for smart home settings where power efficiency and adaptability are crucial [2].

A promising alternative is the use of hybrid systems that combine the strengths of multiple technologies. For example, systems that integrate IMU sensors with depth cameras can improve recognition accuracy while reducing dependency on environmental factors. However, these solutions pose challenges related to system complexity and cost, which must be carefully managed during implementation.

Despite the advancements in gesture recognition technologies, significant challenges remain. Real-time performance, variability in user gestures, and the need for robust algorithms to handle noise in sensor data are persistent issues. Addressing these challenges is essential for the Leap project to deliver a reliable and user-friendly smart home system.

2.2 Facial Recognition for Home Security

Facial recognition technology has emerged as a critical component of modern security systems, offering a non-intrusive and efficient method for identifying individuals. In the context of smart homes, facial recognition enhances security by enabling access control for authorized users while maintaining a high level of convenience. Recent advancements in facial recognition techniques have shifted from traditional methods, such as Eigenfaces and Local Binary Patterns (LBP), to more sophisticated deep learning-based approaches. These traditional methods, while foundational, were limited by their sensitivity to variations in lighting, pose, and facial expressions, making them less effective in real-world scenarios [5].

The advent of deep learning has revolutionized facial recognition systems, providing robust feature extraction and significantly improving accuracy. Models such as FaceNet [6] and DeepFace [10] utilize advanced neural network architectures to generate meaningful embeddings from facial images. These embeddings enable accurate recognition and verification even under challenging conditions, such as varying lighting or facial expressions. Such capabilities make deep learning-based methods particularly suitable for smart home applications, where scalability and adaptability are essential. For instance, the Leap project employs TensorFlow and OpenCV to implement a facial recognition system trained on a carefully curated dataset. The system uses an "Anchor," "Positive," and "Negative" classification scheme to ensure reliable performance [7].

One-shot learning techniques, such as Siamese networks [11], have also gained prominence in facial recognition tasks, particularly in scenarios with limited training data. These methods are highly effective when only a single image per individual is available for training, aligning well with the Leap project's objective of minimizing data requirements while maintaining high accuracy. By leveraging few-shot learning techniques, the project addresses challenges related to dataset sparsity and achieves robust performance [7, 11].

Despite these advancements, facial recognition systems face several challenges that must be addressed to ensure optimal performance. One significant challenge is robustness against variations in lighting conditions and facial poses, which can severely impact accuracy. The Leap project tackles this issue by preprocessing images with OpenCV and employing deep learning algorithms capable of adapting to diverse environmental conditions [8]. Privacy concerns also pose a critical challenge, as facial recognition systems involve the collection and storage of sensitive biometric data. To mitigate these concerns, the Leap project incorporates end-to-end encryption and local processing to safeguard user data, adhering to privacy regulations and the principles of Privacy by Design [9]. Additionally, the system triggers real-time alerts for unauthorized access attempts, further enhancing security.

Real-time performance is another crucial requirement for facial recognition systems, particularly in smart home applications where immediate feedback is necessary. The Leap project ensures real-time processing by leveraging hardware-accelerated computation and cloud synchronization through platforms like Firebase. This combination of local computation and cloud-based synchronization ensures both efficiency and scalability, enabling seamless integration into smart home environments [12].

2.3 Chatbots for Smart Home Assistance

Chatbots have become an integral part of modern smart home systems, providing users with real-time assistance and personalized recommendations using text or speech. Powered by natural language processing (NLP) models, chatbots can understand and respond to user queries in natural language, making them a valuable addition to the Leap project.

2.3.1 Evolution of Chatbots and the Advantages of Modern Systems

The development of chatbots has undergone a remarkable transformation, evolving from

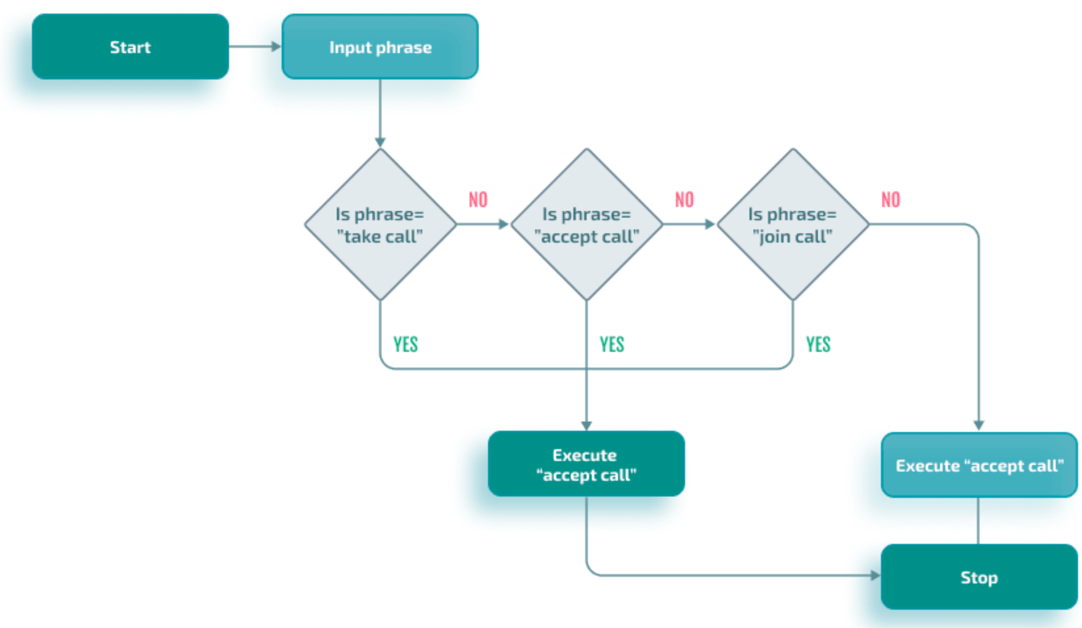

Figure 2.3.1: Rule-based chatbots

rudimentary rule-based systems to highly advanced conversational agents powered by large language models (LLMs). Early chatbots, such as ELIZA (1966), relied on predefined scripts and rule-based algorithms to generate responses. While these systems were simple to implement, they were constrained by their inability to handle complex or dynamic queries, as they lacked the flexibility to adapt to unexpected user inputs.

Advancements in artificial intelligence and machine learning led to the emergence of retrieval-based chatbots. These systems utilized machine learning techniques to identify and retrieve the most relevant responses from a predefined dataset. Although this approach improved conversational relevance and adaptability, it was still limited by its dependence on a fixed pool of responses, lacking the ability to generate novel content.

The introduction of generative chatbots marked a pivotal shift in the capabilities of conversational AI. Powered by neural networks and early transformer models, such as sequence-to-sequence (Seq2Seq) models, these systems could generate original responses based on input context. Examples include Google's Meena and OpenAI's GPT-2, which showcased significant improvements in dialogue coherence and contextual understanding. However, these systems faced challenges in maintaining factual accuracy and handling multi-turn conversations effectively.

Modern chatbots, exemplified by advanced LLMs such as ChatGPT, BERT, and Claude, represent the latest evolution in chatbot technology. These models leverage transformer-based architectures trained on extensive datasets, enabling them to process multi-turn interactions, understand context deeply, and produce human-like, contextually relevant responses. Fine-tuned on domain-specific data, these chatbots excel in a wide range of applications, from scheduling and financial analysis to educational tools and customer support.

The key advantages of modern systems lie in their flexibility, scalability, and personalization. Unlike their predecessors, these chatbots can adapt to diverse domains, support multilingual interactions, and personalize responses based on user preferences and historical data. Moreover, their seamless integration with IoT devices and edge computing allows for real-time decision-making and reduced latency, making them ideal for smart home systems and other interactive environments.

This progression from simple rule-based algorithms to sophisticated LLMs underscores the continuous advancements in chatbot technology. Modern systems offer unparalleled capabilities, combining contextual understanding, generative proficiency, and multi-domain adaptability, making them indispensable tools for both personal and professional applications.

2.3.2 Model Selection

Why Chavinlo/alpaca-native Model?

The "chavinlo/alpaca-native" model was chosen for its advanced capabilities in generating coherent and contextually relevant text, making it ideal for conversational AI tasks. Its instruction-tuning ensures a deep understanding of user input, enabling natural and dynamic interactions that closely mimic human conversation. The model's architecture is optimized for efficiency, providing fast inference and low resource consumption, which are critical for real-time applications. This balance between performance and computational efficiency ensures high responsiveness without compromising on the quality of language understanding and generation.

Moreover, the model integrates seamlessly with the Hugging Face ecosystem, a platform known for its extensive tools and community support. This compatibility facilitates easy customization and scalability, allowing the model to be fine-tuned for domain-specific tasks with minimal effort. The combination of advanced text generation, real-time performance, and adaptability makes "chavinlo/alpaca-native" a robust and versatile choice for building conversational agents in complex and interactive environments.

2.3.3 Challenges in Chatbot Development

- **Contextual Understanding**: Chatbots must maintain context across multiple turns in a conversation to provide coherent and relevant responses. This requires sophisticated models capable of capturing long-term dependencies. The Leap project addresses this by integrating advanced NLP models that maintain context throughout user interactions.
- **Personalization**: To enhance user experience, chatbots should be able to personalize responses based on user preferences and historical interactions. This requires the integration of user profiling and recommendation systems. The Leap project will incorporate personalization features to tailor the smart home experience to individual users.
- **Real-Time Performance**: Chatbots must respond to user queries in real-time, with minimal latency, to ensure a smooth and engaging interaction. The Leap system will achieve this by optimizing communication protocols and leveraging Firebase for real-time data synchronization.

2.4 Text-to-Speech and Speech Recognition in Conversational AI

Text-to-Speech (TTS) and Speech Recognition (SR) technologies are fundamental components of modern conversational AI systems, enabling seamless interaction between users and machines. Speech recognition systems, such as Python's speech_recognition library, facilitate the conversion of spoken language into text, allowing chatbots to process and understand verbal queries. These systems rely on advanced acoustic modeling and natural language processing to handle variability in speech, such as accents, noise, and diverse languages, ensuring accurate transcription for real-time applications.

Text-to-Speech systems, such as Google Text-to-Speech (gTTS), complement SR by converting textual responses generated by the chatbot into natural-sounding speech. This bidirectional capability enhances user experience by providing an interactive and human-like communication interface. The integration of TTS and SR enables applications such as voice-controlled systems, accessibility tools, and interactive customer service.

To optimize these functionalities, preprocessing tools like `pydub` are employed for audio manipulation, including format conversion, trimming, and enhancing sound quality. Real-time performance is achieved by leveraging efficient libraries and cloud-based solutions, such as Firebase, for data synchronization and latency minimization. Together, TTS and SR technologies form the backbone of voice-enabled AI systems, driving advancements in accessibility, user engagement, and interaction efficiency.

2.5 Integration of Smart Home Technologies

The integration of gesture recognition, facial recognition, and chatbot technologies into a unified smart home system presents both opportunities and challenges. While each technology has its own strengths, combining them into a cohesive system requires careful design and implementation.

2.5.1 System Integration Challenges

- **Interoperability**: Ensuring that different components of the system, such as the smart Ring, facial recognition system, and chatbot, can communicate seamlessly is critical. This requires the use of standardized communication protocols and APIs [19]. The Leap project employs a modular architecture with layered communication protocols to ensure seamless interoperability.
- **Scalability**: The system must be scalable, allowing for the addition of new devices and functionalities in the future. This requires a modular architecture that can accommodate changes without disrupting existing operations [20]. The Leap project is designed with scalability in mind, enabling future expansions such as voice assistant hardware and advanced health monitoring features.
- **User Experience**: The integration of multiple technologies must not compromise the user experience. The system should provide a consistent and intuitive interface across all components [21]. The Leap project prioritizes user-centric design, ensuring that the system is easy to use and accessible to all users.

2.5.2 Prior Work in Smart Home Integration

- **Unified Control Platforms**: Several smart home platforms, such as Google Home and Amazon Alexa, have attempted to integrate multiple control methods, including voice commands, mobile apps, and gesture recognition. However, these platforms often lack the depth of integration and personalization offered by the Leap project [22].
- **Wearable Devices for Smart Homes**: Wearable devices, such as smartwatches and fitness trackers, have been explored as control mechanisms for smart homes. However, these devices are often limited in their functionality and do not offer the same level of precision and customization as the smart Ring [23]. The Leap project addresses this limitation by introducing gesture-based rings and a smart band as the central hub for home automation.

2.6 Gaps in Existing Systems

Despite the advancements in gesture recognition, facial recognition, and chatbot technologies, several gaps remain in existing smart home systems:

1. **Limited Inclusivity**: Many smart home systems rely heavily on voice commands and mobile apps, which may not be suitable for users with disabilities or those in noisy environments. The Leap project addresses this gap by offering gesture-based control as an inclusive alternative.

2. **Fragmentation**: Existing systems often operate in silos, with separate devices and applications for different functionalities. This fragmentation leads to inefficiencies and a lack of integration. The Leap project integrates gesture recognition, facial recognition, and chatbot technologies into a unified system, providing a seamless user experience.

3. **Energy Efficiency**: Wearable devices and other hardware components must be energy-efficient to ensure continuous operation. Many existing systems fail to prioritize power management, leading to frequent recharging or battery replacements. The Leap project optimizes energy consumption through efficient hardware design and firmware optimization.

Chapter 3

System Design and Methodology

This chapter provides an in-depth overview of the system design, and the methodology employed in the Leap project. It elaborates on the integration of the gesture recognition system, facial recognition system, and the chatbot, each of which plays a critical role in the smart ring-based control system. The chapter explains the architecture, workflow, datasets, data preprocessing techniques, model selection, and tools used in developing the Leap system.

3.1 Overview of the System Architecture

The Leap system has a modular architecture that is scalable, flexible, and hence easy to integrate into the framework of smart home control. It contains three main components performing different roles in the framework:

The **first** is the smart ring that performs gesture recognition with sensors mounted on it detecting hand movements. Information relayed from the sensors is passed on to a central processing unit, the smart band, where the information captured is processed to identify certain gestures. The smart ring wirelessly communicates with the band using low-power communication protocols such as ESP-NOW.

The **second** is the facial recognition system, where pictures taken by a camera identify the user with the help of a Siamese Neural Network. It's a real-time system, giving instant feedback, hence useful for security. It ensures privacy and reduces latency since it processes data locally. Face detection is done using OpenCV, while the Siamese Neural Network is built and trained using TensorFlow and Keras, respectively, with the Labelled Faces in the Wild (LFW) dataset. It is designed to cope with lighting, pose, and facial expression variations to perform well in real-world conditions.

The **third** major element is the chatbot system, which is run using the Hugging Face Transformers library. It uses the "chavinlo/alpaca-native" model to come up with responses that sound very much like those of a human, thus giving live assistance to users. The chatbot will interact through both text and voice commands via a mobile application and website to give personalized recommendations and control smart home devices.

The app and the website are the main interfaces of the Leap system: they are how users will configure and control their smart home devices, interact with the chatbot, and set security preferences—basically, user-friendly interfaces designed to ensure accessibility on all kinds of devices.

3.2 Gesture Recognition System

The gesture recognition system relies on the smart rings, which captures hand gestures using Inertial Measurement Units (IMUs) embedded in the ring. These IMUs provide data on acceleration, and angular velocity. The captured data is transmitted to the smart band, which processes the data to identify predefined gestures. The system is designed to recognize gestures related to controlling various smart home devices, such as turning lights on/off, adjusting the thermostat, and locking/unlocking doors.

The recognition model is trained using a custom dataset of hand gestures collected, with each gesture performed multiple times to account for variability. This dataset includes a total of 1,0000 records, which undergo preprocessing techniques such as normalization, filtering, and data augmentation to improve the model's robustness and prevent overfitting. The model itself uses Dynamic Time Warping (DTW) technique and RCE (Restricted Coulomb Energy) Neural Network to capture spatial patterns in the gesture data, enabling accurate recognition of complex hand movements.

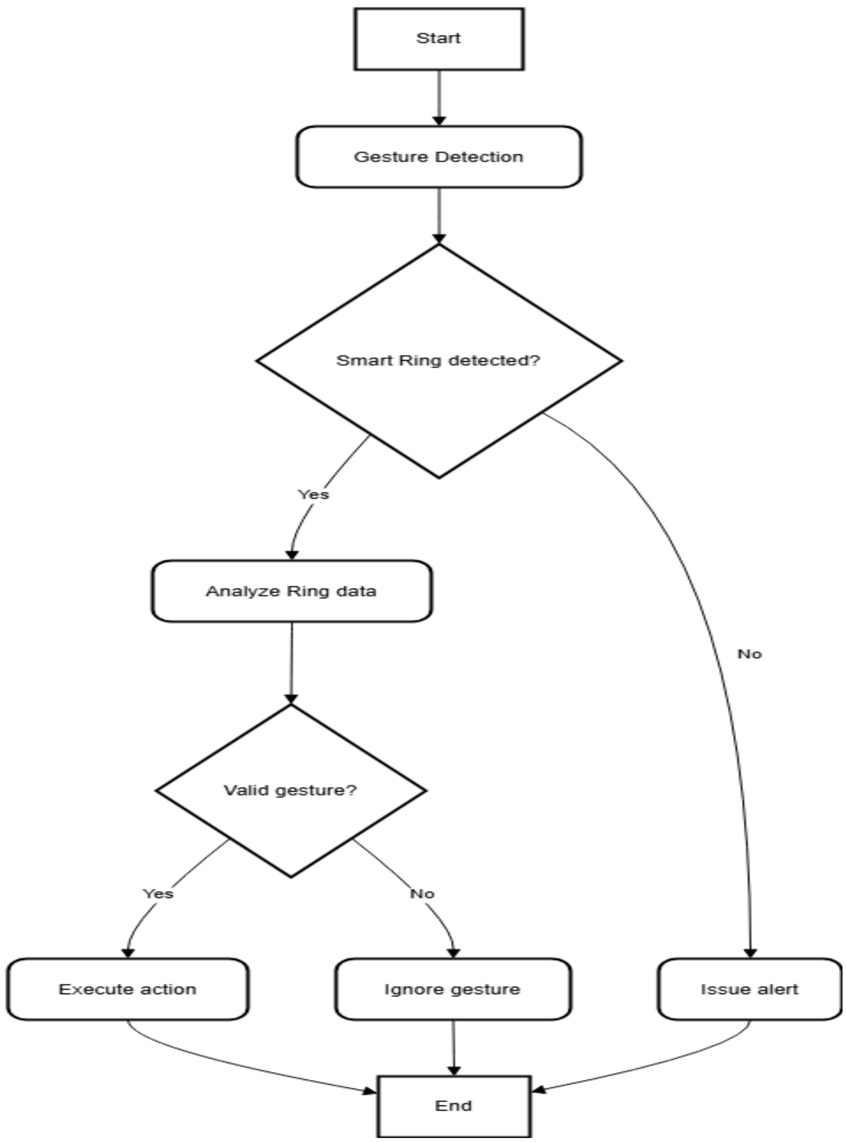

Figure 3.1: Workflow Diagram of the Leap Ring

3.3 Facial Recognition System

The facial recognition system uses a camera to capture images of users, which are then processed by a Siamese Neural Network for identity verification. The Siamese network compares the features of the captured image with a stored reference image to confirm the user's identity. This approach minimizes the need for large amounts of training data while maintaining high accuracy. The model is trained on the LFW dataset, which contains over 13,000 labeled images of faces under varying conditions such as different lighting and poses. For preprocessing, facial images are cropped to focus solely on the face, resized to a standard resolution, and normalized to improve model performance. Data augmentation techniques, such as random rotations and brightness adjustments, are applied to enhance the model's robustness in different real-world scenarios. The system is designed to run efficiently in real-time, providing immediate security feedback and reducing the risk of unauthorized access.

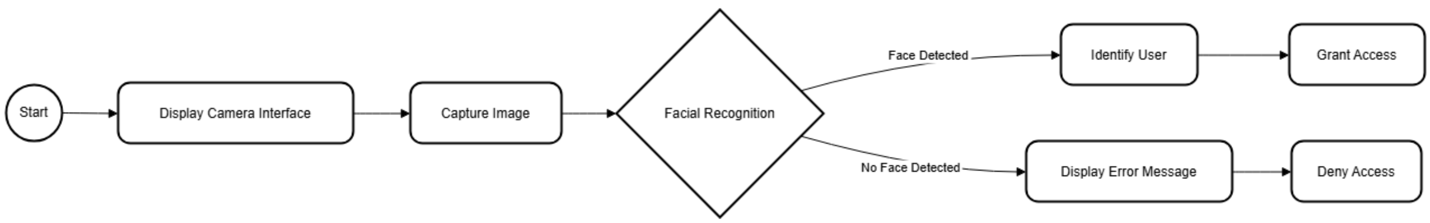

Figure 3.2: Workflow Diagram of the Leap Facial Recognition System

3.4 Chatbot System

The Leap system's chatbot is designed to provide users with interactive assistance and personalized recommendations. It is powered by the Hugging Face Transformers library and uses the"chavinlo/alpaca-native" model for natural language processing (NLP). The chatbot processes user queries related to smart home control, answering questions and executing commands based on user input. The model is trained on a dataset of conversational data, which includes queries related to smart home management and security settings. Preprocessing for the chatbot data includes tokenization, where sentences are broken into words or sub-words, followed by padding to standardize the input length. This enables efficient training and consistent performance across different input queries. The chatbot is accessible through both text and voice commands, ensuring flexibility in user interactions.

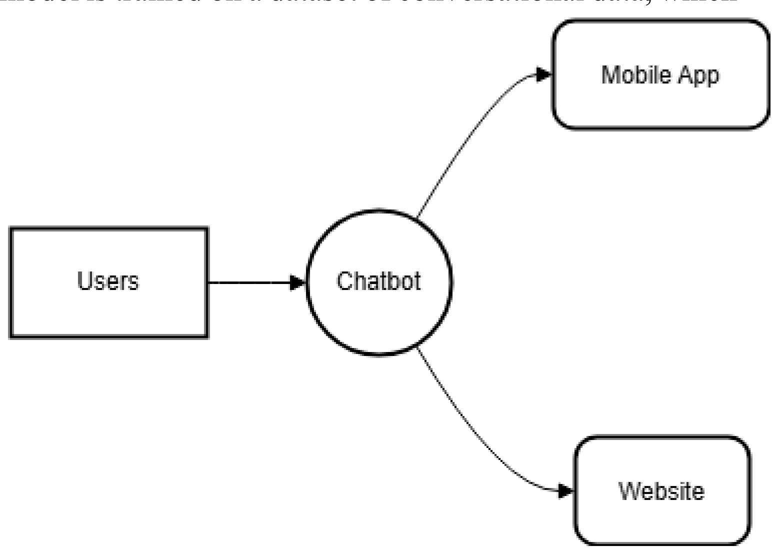

Figure 3.3: Workflow Diagram of the Leap Chatbot System

3.5 Datasets

The Leap project utilizes the following datasets for training and evaluation:

1. **Custom Gesture Dataset**:

   A custom dataset of hand gestures was created using the smart Rings. The dataset includes gestures for controlling various smart home devices, such as turning lights on/off, adjusting the thermostat, and locking/unlocking doors. The dataset contains 1,0000 records collected, with each gesture performed multiple times to ensure variability.

2. **Labelled Faces in the Wild (LFW) Dataset**:

   The Labelled Faces in the Wild (LFW) dataset is a widely used benchmark dataset in facial recognition research. It contains over 13,000 labeled facial images of more than 5,700 individuals, capturing a diverse range of variations in lighting, pose, expression, and background. The dataset is particularly valuable for training and evaluating facial recognition systems due to its realistic and unconstrained nature, which mirrors real-world conditions. By leveraging the LFW dataset, the facial recognition system can be trained to achieve robust performance across diverse scenarios, ensuring high accuracy and reliability in identifying individuals under challenging conditions.

3. **Chatbot Training Data**:

   The chatbot is trained on a dataset of conversational data, including user queries and responses related to smart home control. The dataset is preprocessed to remove noise and ensure consistency.

3.6 Data Preprocessing

Data preprocessing plays a crucial role in enhancing the accuracy and reliability of the Leap system by preparing raw data for effective model training. For the gesture data, preprocessing involves several techniques to improve the quality and consistency of the input. First, normalization is applied to ensure uniform scaling across different sensors and users. Filtering is used to remove noise and outliers through filters such as Butterworth or Kalman, while data augmentation techniques, including rotation, scaling, and time warping, help increase dataset variability and reduce the risk of overfitting. In the case of facial data, preprocessing steps focus on improving the model's ability to recognize faces under various conditions. Facial images are cropped to isolate the face region and eliminate irrelevant background, then resized to a standard resolution of 105x105 pixels for uniformity. Normalization is performed on the pixel values, adjusting them to the [0, 1] range to optimize model convergence. Additionally, data augmentation, such as random rotations, flips, and brightness adjustments, enhances the model’s robustness. For chatbot data, preprocessing primarily involves tokenization, which splits the conversational text into words or sub-words, followed by padding to ensure consistent input size, thus allowing the model to handle varying query lengths efficiently. These preprocessing techniques collectively enable the Leap system to effectively train and deploy accurate models across its gesture recognition, facial recognition, and chatbot components.

3.7 Choosing Learning Algorithm

The Leap project incorporates a combination of traditional machine learning and deep learning algorithms to optimize accuracy and real-time performance across various tasks. In the early stages of development, traditional machine learning techniques were employed, including Support Vector Machines (SVM) for gesture recognition due to their simplicity and effectiveness with smaller datasets, and k-Nearest Neighbors (k-NN) for facial recognition, which was later replaced by deep learning models for improved performance. As the project evolved, deep learning algorithms became central to the system's functionality. RCE (Restricted Columb Energy) and DTW (Dynamic Time Wrapping) technique for gesture recognition to capture spatial patterns within the gesture data, while Siamese Neural Networks were employed for facial recognition, enabling efficient comparison of facial features with minimal training data. For the chatbot component, Transformers were used to generate human-like responses based on user queries, enhancing the conversational capabilities of the system. A comparison of machine learning and deep learning highlights that while deep learning models generally outperform traditional algorithms in accuracy, particularly for complex tasks like gesture and facial recognition, they come with higher computational requirements and the need for larger datasets. Despite these demands, deep learning models, when properly optimized, can still deliver real-time performance suitable for smart home applications. The decision to focus on deep learning was driven by its superior accuracy, robustness, and the availability of pre-trained models and frameworks, such as Pytorch, TensorFlow and Hugging Face, which streamline the development process and align with the project's requirements for high performance and scalability.

3.8 Data Modeling

3.8.1 RCE NN and Dynamic Time Warping for Gesture Recognition

The Restricted Coulomb Energy Neural Network (RCE NN), when combined with Dynamic Time Warping (DTW), offers an efficient and accurate approach for gesture recognition, particularly for time-series data.

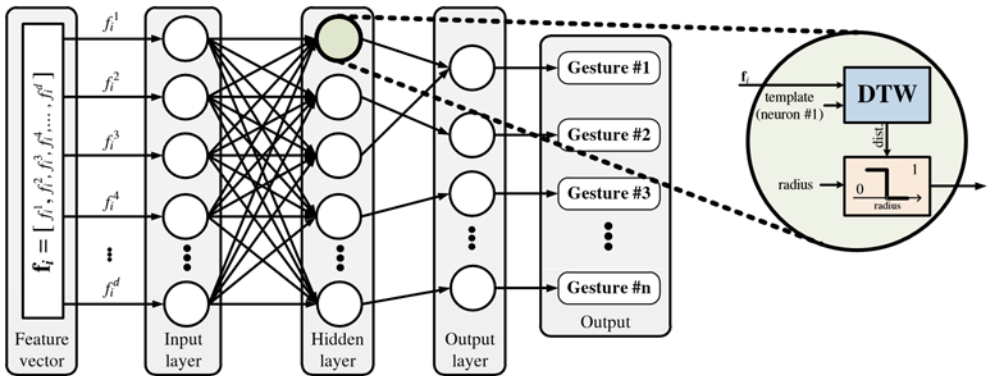

Figure 3.8.1. Structure of an RCE neural network.

3.8.2 Siamese Network for Facial Recognition

- The Siamese network is designed to compare facial features and identify users. It consists of two identical subnetworks that share weights and are trained using contrastive loss.

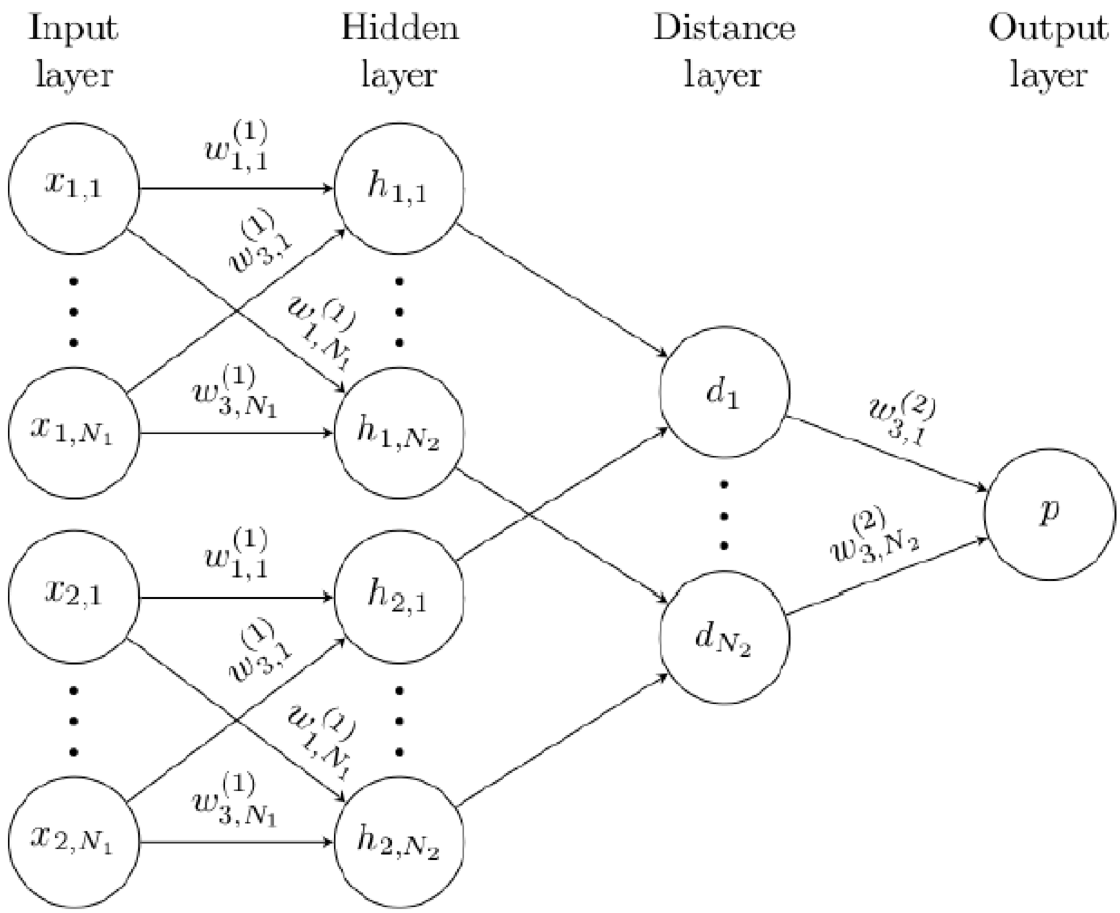

Figure 3.8.2: A simple 2 hidden layer Siamese network for binary classification with logistic prediction p. The structure of the network is replicated across the top and bottom sections to form twin networks, with shared weight matrices at each layer [11].

3.8.3 Chatbot Model Using Hugging Face Transformers

- The chatbot is powered by the **"chavinlo/alpaca-native"** model from Hugging Face. The model is fine-tuned on the chatbot training data to generate contextually relevant responses.

3.8.4 Integration of Gesture, Facial Recognition, and Chatbot

- The three components are integrated into a unified system using **Firebase** for real-time data storage and synchronization. The mobile application and website serve as the central interface for user interaction.

3.7 Toolset Overview

The Leap project utilizes a diverse set of tools and technologies to ensure the development and deployment of an efficient and scalable system. **Python** serves as the primary programming language, chosen for its versatility and robust ecosystem of libraries. Libraries such as **TensorFlow** and **Keras** are employed for building and training the gesture and facial recognition models, enabling the system to effectively learn and process data. **Hugging Face Transformers** are utilized to develop the chatbot, enabling it to generate human-like responses and enhance the interaction experience. **Firebase** plays a crucial role in the system by providing real-time data storage and synchronization, ensuring seamless communication between the smart ring, facial recognition system, and chatbot. For backend development, **Node.js** and **Express.js** are used to build and manage the website and mobile application, allowing for efficient handling of user requests and ensuring smooth integration with the other components of the system. This combination of tools facilitates the development of a comprehensive system that can deliver real-time, intelligent performance.

Chapter 4

Development

4.1 System Requirements

4.1.1 Functional Requirements

The functional requirements define the core capabilities that the Leap system must offer to ensure its success in meeting user needs.

First, the Leap system employs a multi-sensor setup designed to ensure precise data acquisition and effective gesture recognition, crucial for meeting user needs. The system features three sensors positioned on individual fingers and one sensor configured as a bracelet on the forearm. This strategic configuration serves multiple purposes:

Fine-Grained Motion Detection: Sensors on the fingers provide precise measurements of individual finger movements, allowing for the accurate identification of gestures with high fidelity.

Global Muscle Activity Tracking: The forearm sensor captures broader muscle activation patterns, enriching the dataset with complementary signals that reflect overall arm dynamics.

Enhanced Data Reliability: The integration of multiple sensors enables cross-validation of signals, mitigating noise or inaccuracies from any single sensor, thus improving the overall reliability and robustness of the data.

Comprehensive Gesture Recognition: By combining localized data from the finger sensors with global data from the forearm sensor, the system achieves a holistic understanding of hand and finger gestures, crucial for distinguishing complex or subtle gestures.

This sensor configuration ensures a diverse and robust dataset, which enhances the training of advanced machine learning algorithms. As a result, the data quality improves the accuracy and reliability of the gesture recognition system, laying a solid foundation for real-world applications, particularly in controlling smart home devices.

The core functionality of the Leap system revolves around its ability to accurately detect and interpret predefined hand gestures. These gestures are integral to controlling smart home devices, with real-time feedback provided to confirm the recognition of gestures and the execution of commands. This capability is essential for delivering an intuitive and seamless user experience, ensuring the system meets user expectations effectively.

Second, the facial recognition system must be capable of detecting and recognizing faces in real-time using a webcam or camera sensors. The system must be trained on a dataset of known individuals, allowing it to verify authorized users and distinguish them from unauthorized individuals. If unauthorized access is detected, the system should send real-time notifications to the homeowner, alerting them to potential security breaches.

The chatbot functionality is another crucial component. The chatbot must understand user queries expressed in natural language and provide relevant responses. Additionally, the chatbot should be able to offer personalized recommendations based on the user's preferences and historical data, improving the overall user experience.

The mobile application and website must provide user-friendly interfaces that allow users to control their smart home devices, interact with the chatbot, and manage security settings. These interfaces should be designed to be accessible on a variety of devices, including smartphones, tablets, and desktop computers, ensuring flexibility and convenience for the user.

Lastly, the system must seamlessly integrate the gesture recognition, facial recognition, and chatbot functionalities into a unified platform using IoT technologies. Communication between the smart ring, facial recognition system, chatbot, and smart home devices should be supported via reliable technologies such as ESP-NOW, Wi-Fi, and Firebase, enabling smooth and efficient operation.

4.1.2 Non-Functional Requirements

The nonfunctional requirements detail the overall qualities that the Leap system must embody to ensure its success in real-world applications.

Security is a primary consideration. The system must encrypt user data and ensure that all communication between devices is secure to protect user privacy. The facial recognition system must store sensitive user data securely and comply with applicable data protection regulations to ensure that user information is kept safe.

Performance is another critical requirement. The system must be responsive, with minimal latency during gesture recognition, facial recognition, and chatbot interactions. It should provide real-time responses to user inputs and be able to handle multiple users and devices simultaneously without any noticeable performance degradation. This is essential for maintaining a smooth and uninterrupted user experience, even in more complex environments.

Usability is also a key factor for the system's success. The system must be easy to use, with intuitive interfaces for the smart ring, mobile application, and website. It should provide clear feedback to users, confirming that gestures have been recognized, facial recognition has been successful, and chatbot responses have been processed. A well-designed user interface will ensure that users can interact with the system without unnecessary complexity.

Scalability is another important aspect. The system must be designed to accommodate future expansion, allowing the integration of additional devices and functionalities as required. It should be flexible enough to adapt to different user needs and environments, making it suitable for a wide range of applications, from home use to commercial settings.

Finally, **energy efficiency** is essential for the long-term usability of the system. The smart rings and other hardware components must be energy-efficient to ensure long battery life and continuous operation. The system should optimize power consumption, reducing the need for frequent recharging or battery replacement, thereby making it convenient for the user and ensuring the system's reliability over time.

4.2 Software Model: Agile Methodology

4.2.1 Introduction to Agile Methodology

Agile methodology is a dynamic, iterative, and incremental approach to software development that emphasizes flexibility, collaboration and continuous improvement. It enables teams to adapt to changing requirements efficiently while delivering functional components in short cycles called sprints.

4.2.2 Features of Agile Methodology

1. **Iterative Development:** Agile divides the project into manageable iterations, ensuring incremental delivery of functional components.
2. **Flexibility:** The methodology accommodates changes to project requirements at any stage of development.
3. **Collaboration:** Agile promotes continuous collaboration among team members, stakeholders, and users.
4. **Focus on Deliverables:** Each sprint ends with a deliverable, enabling real-time feedback and early identification of issues.
5. **User-Centric Approach:** Regular user feedback ensures the product aligns closely with user expectations and needs.

4.2.3 Suitability of Agile to The Leap Project

The Agile methodology is ideal for developing the Leap project due to the following reasons:

1. **Complexity Management:** Combining gesture recognition, facial recognition, and chatbot systems requires iterative development and integration, which Agile supports effectively.
2. **Adaptability to Changing Requirements:** Agile accommodates changes in functionality, such as adding features like OCR integration or enhancing personalization based on feedback.

3. **User-Centric Focus:** Regular testing and feedback during sprints ensure the final product meets user needs for accessibility, security, and usability.
4. **Integration Testing:** Agile's continuous integration approach ensures smooth communication between components like the smart ring, chatbot, and IoT devices.

4.2.4 Phases of Agile Methodology for The Leap Project

1. **Concept and Requirement Gathering:**
   - Collaborate with stakeholders to define high-level objectives, such as integrating gesture recognition, face recognition and real-time chatbot interaction.
   - Identify critical requirements, including modularity, scalability, and accessibility.
2. **Sprint Planning:**
   - Break down the project into manageable sprints.
   - Define tasks, such as developing the gesture recognition system or chatbot interaction flow.
3. **Design and Prototyping:**
   - Create prototypes for components like the smart ring gesture recognition or chatbot interface.
   - Review prototypes with stakeholders for early feedback.
4. **Development and Integration:**
   - Develop individual components (gesture recognition, chatbot and facial recognition) incrementally during sprints.
   - Integrate components gradually and test for seamless functionality.
5. **Testing:**
   - Conduct unit testing for individual modules and integration testing for combined functionality.
   - Perform user acceptance testing to validate the system in real-world scenarios.
6. **Delivery and Deployment:**
   - Deliver functional components at the end of each sprint.
   - Deploy a fully integrated and tested system after completing all sprints.
7. **Feedback and Iteration:**
   - Gather feedback after each sprint to refine the system.
   - Address issues, enhance features, and optimize performance in subsequent iterations.

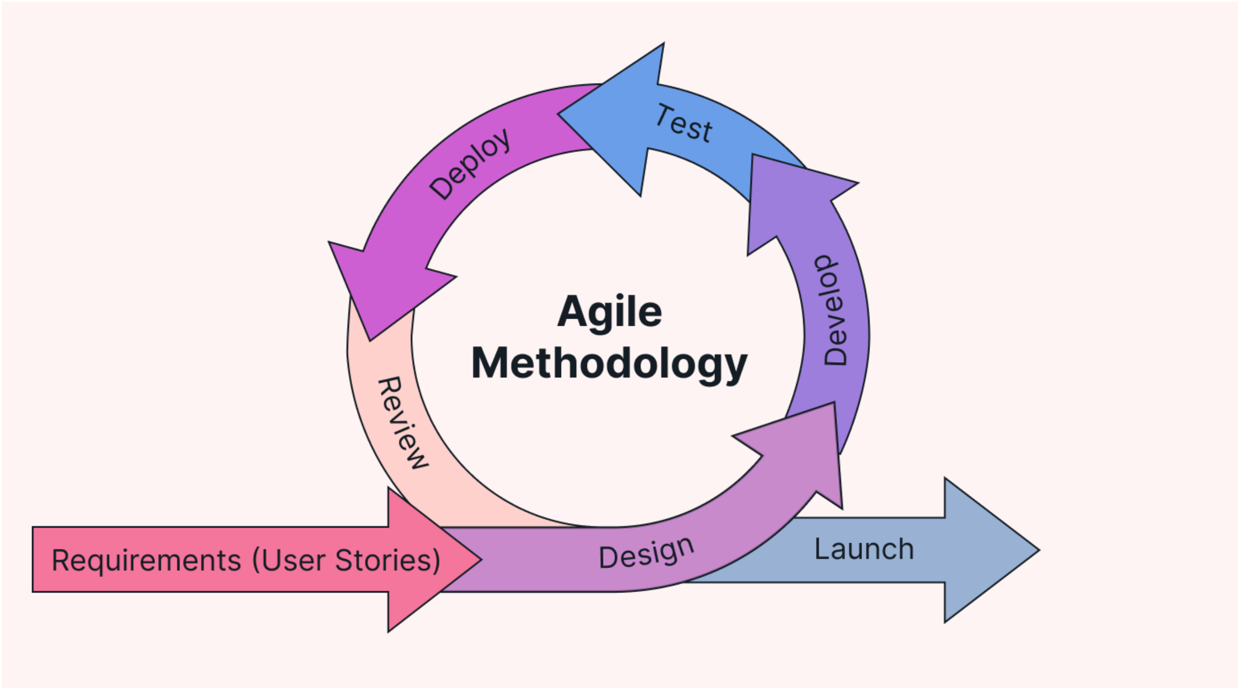

Figure 4.2.4: Agile Phases

4.2.5 Methodology

The Leap project employs a modular design, combining gesture recognition, facial recognition, and chatbot technologies into a unified smart home system. The methodology involves:

1. **Data Collection:**
   - Gathering gesture data from participants and training datasets for facial recognition and chatbot systems.
2. **Preprocessing:**
   - Applying techniques like normalization, filtering, and augmentation to enhance data quality.
3. **Model Development:**
   - Implementing RCE NN (Restricted Coulomb Energy Neural network) for gesture recognition.
   - Using Siamese Networks for facial recognition.
   - Leveraging Hugging Face Transformers for chatbot responses.
4. **Integration:**
   - Connecting all components using Firebase for real-time data storage and synchronization.
5. **Testing and Feedback:**
   - Conducting iterative testing and refining based on user feedback to ensure a user-friendly, secure, and efficient system.

4.3 Website

This section provides an overview of the web development efforts for the Leap project. The focus is on the completed sections, including the UI design, while outlining the planned future work that will further enhance the project's capabilities.

The UI/UX Design for the Leap project has been successfully completed. Figma was used to create visually appealing and user-friendly interfaces, ensuring a seamless and intuitive user experience. The following key pages were designed:

4.3.1 User Interface and User Experience (UI/UX)

4.3.1.1 User Interface Design for Landing Page

The entry point for users to learn about the system.

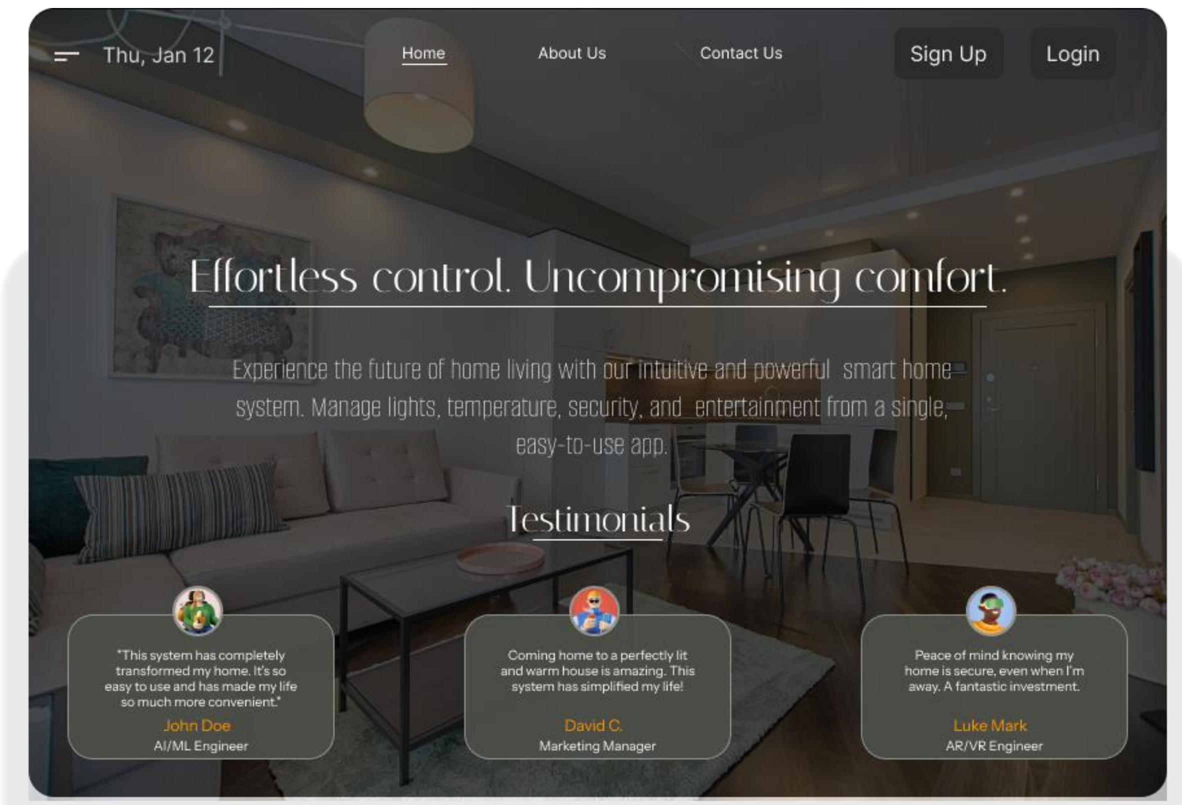

Figure 4.3.1.1: Landing Page Webpage

4.3.1.2 User Interface Design for Contact Us Page:

Enables users to reach out for support, feedback, or inquiries.

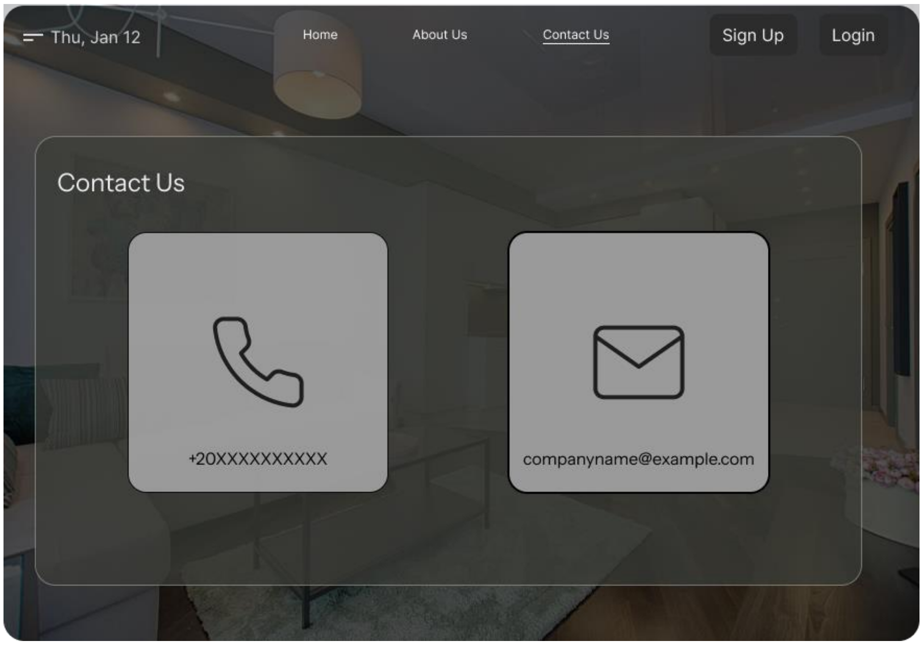

Figure 4.3.1.2: Contact Us Webpage

4.3.1.3 UI for About Us Page:

Provides information about the project, its goals, and the team behind it.

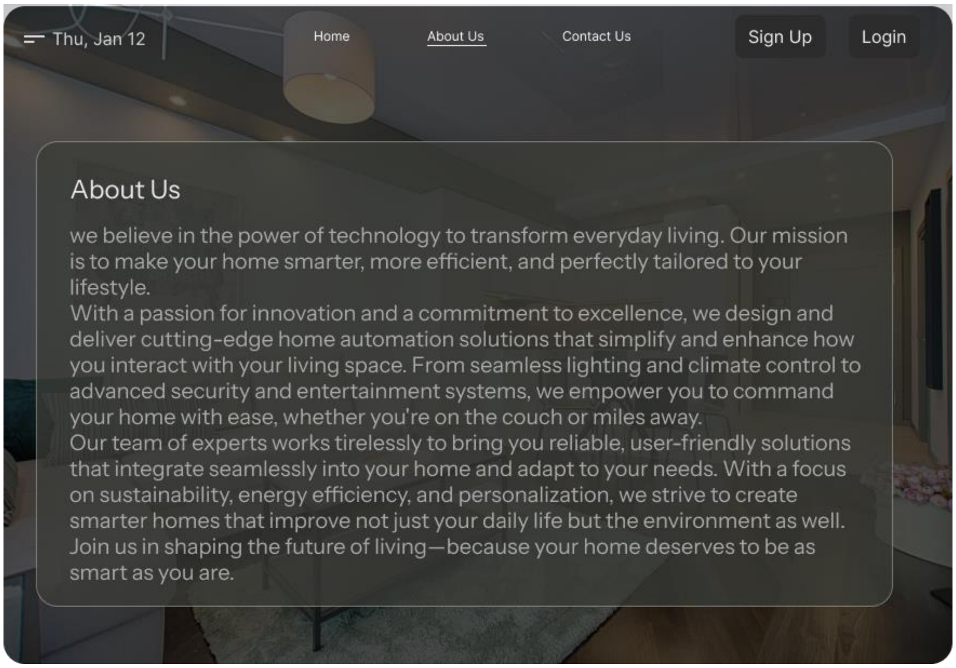

Figure 4.3.1.3: About Us Webpage

4.3.1.4 UI for Profile Page*: Allows users to manage their personal information.*

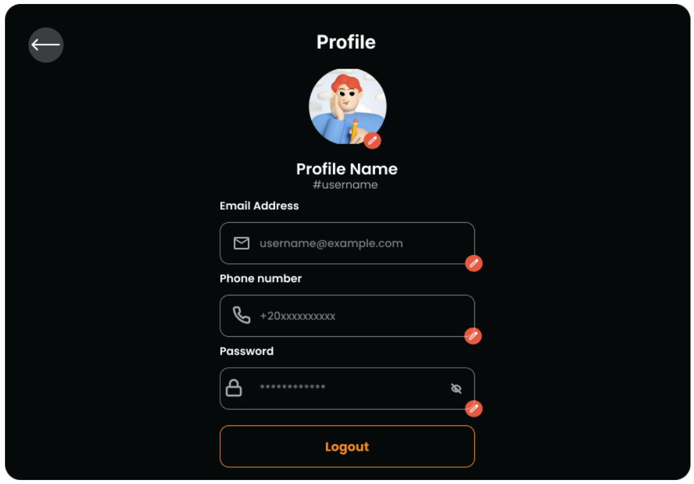

Figure 4.3.1.4**:** Profile Webpage

4.3.1.5 UI for Login/Signup Page: Enables user authentication and account creation.

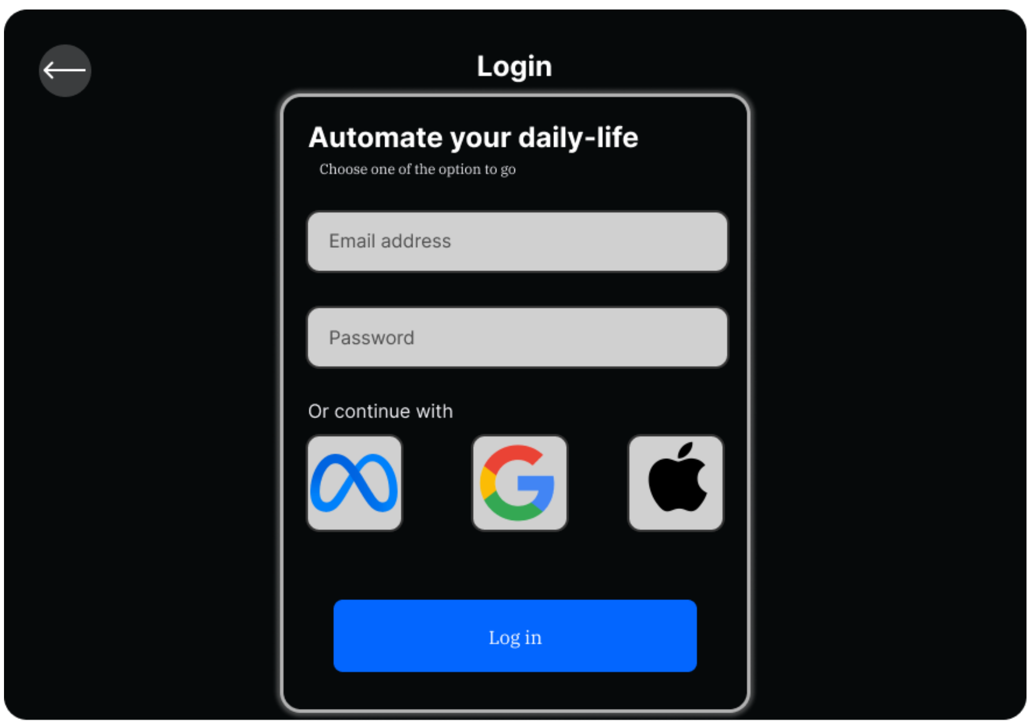

Figure 4.3.1.5.1: login Webpage

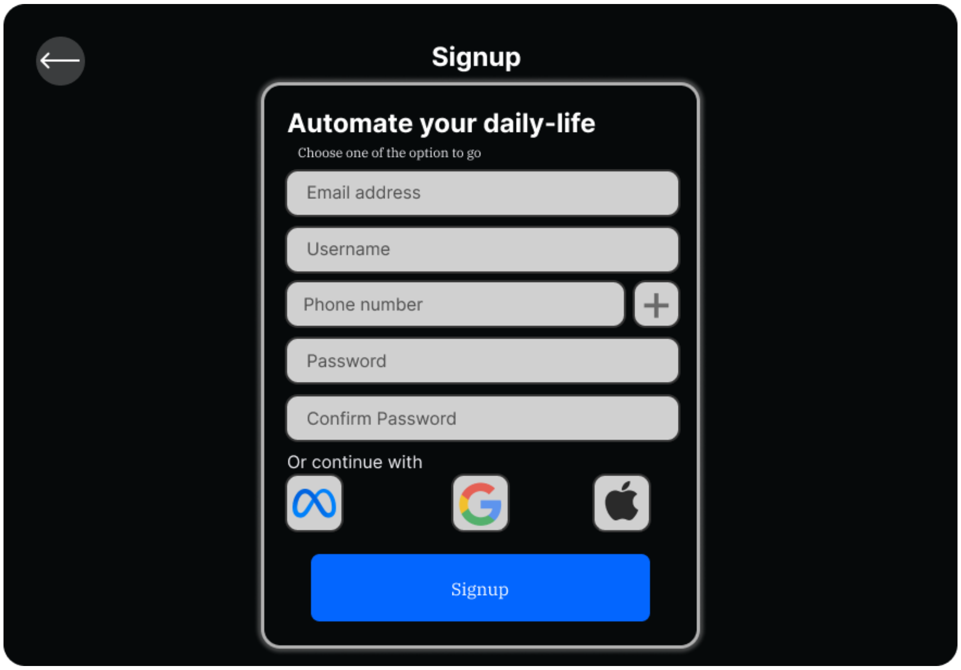

Figure 4.3.1.5.2: Signup Webpage

4.3.1.6 UI for User Information Form Page: Collects user-specific details.

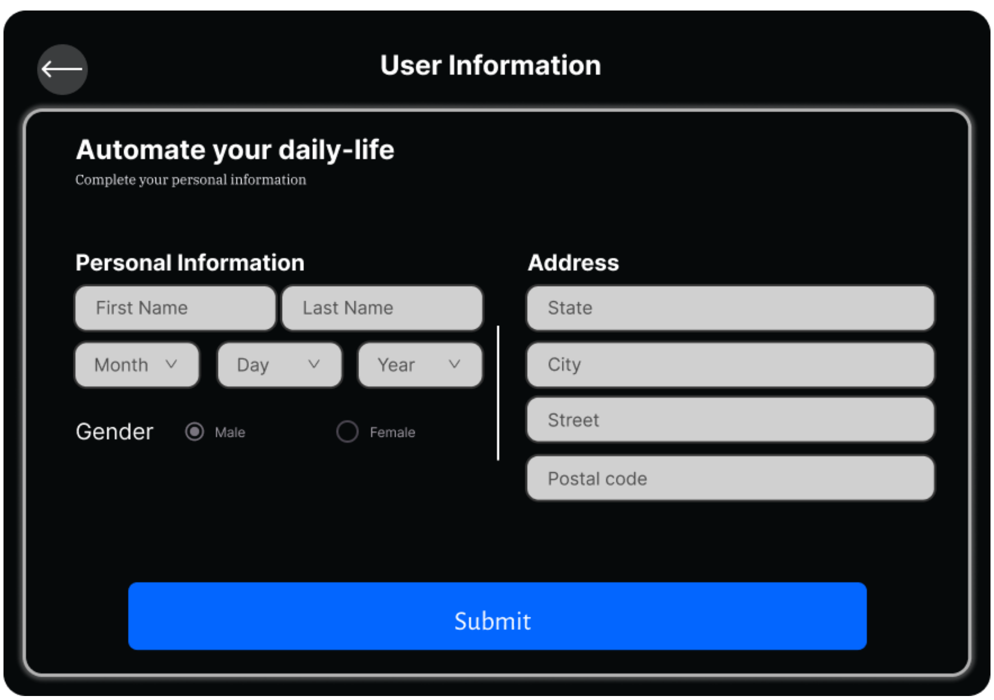

Figure 4.3.1.6: User Information Webpage

4.3.1.7 UI for Home Information Form: Gathers data about the user's home setup.

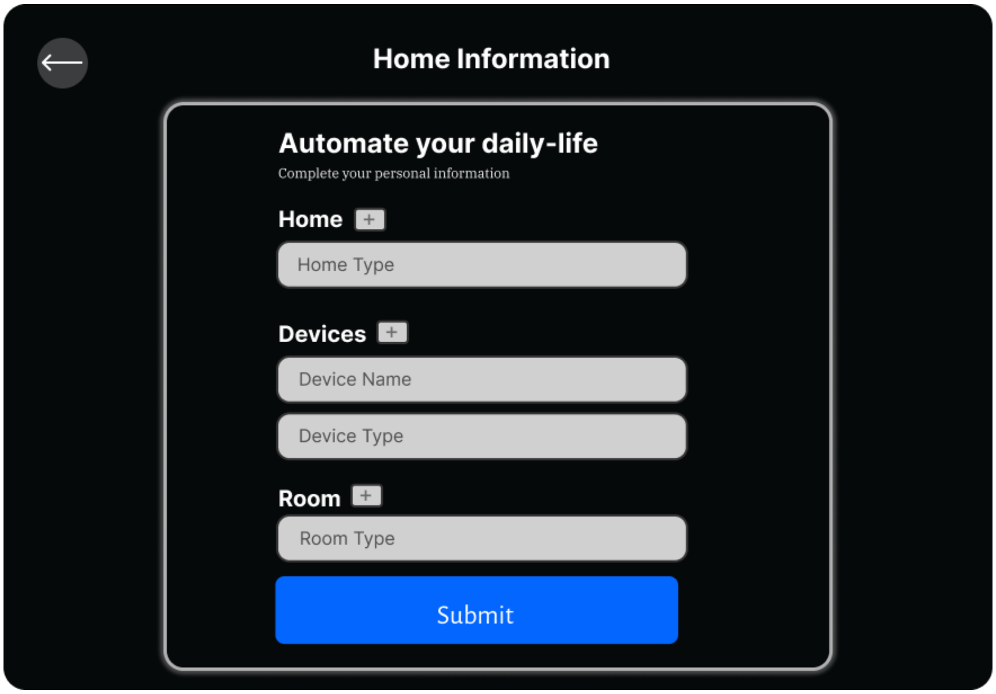

Figure 4.3.1.7: Home Information Webpage

4.3.1.8 UI for Room-Specific Pages:

Includes dedicated pages for the kitchen, bedroom, and living room, allowing users to control and monitor these spaces.

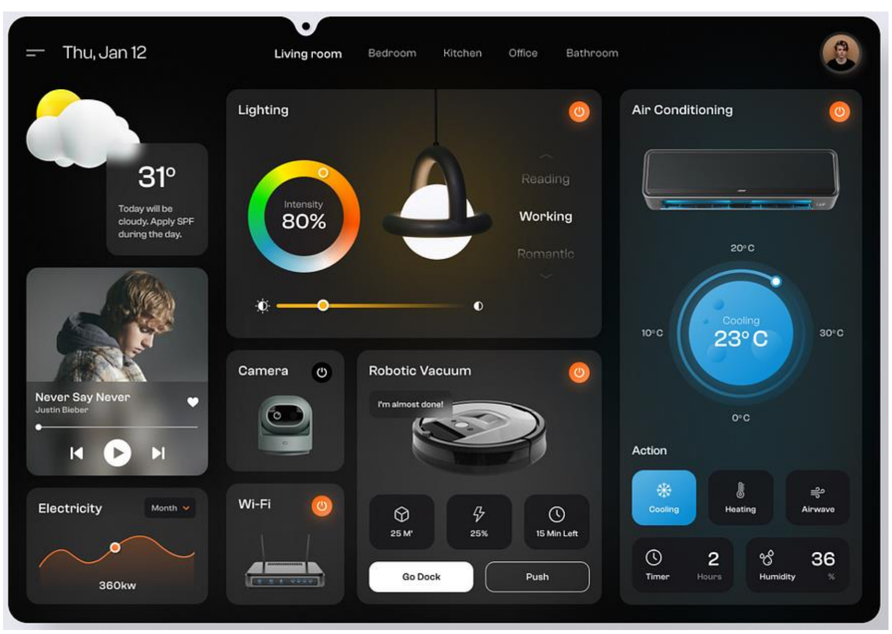

Figure 4.3.1.8.1: Living room Webpage.

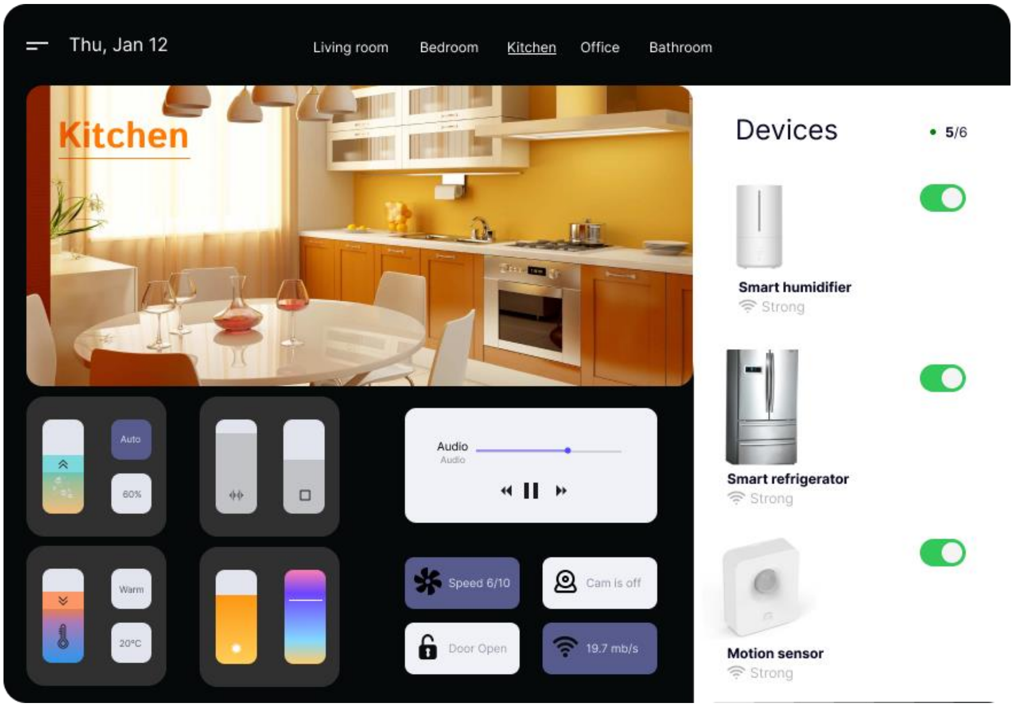

Figure 4.3.1.8.2: Kitchen Webpage

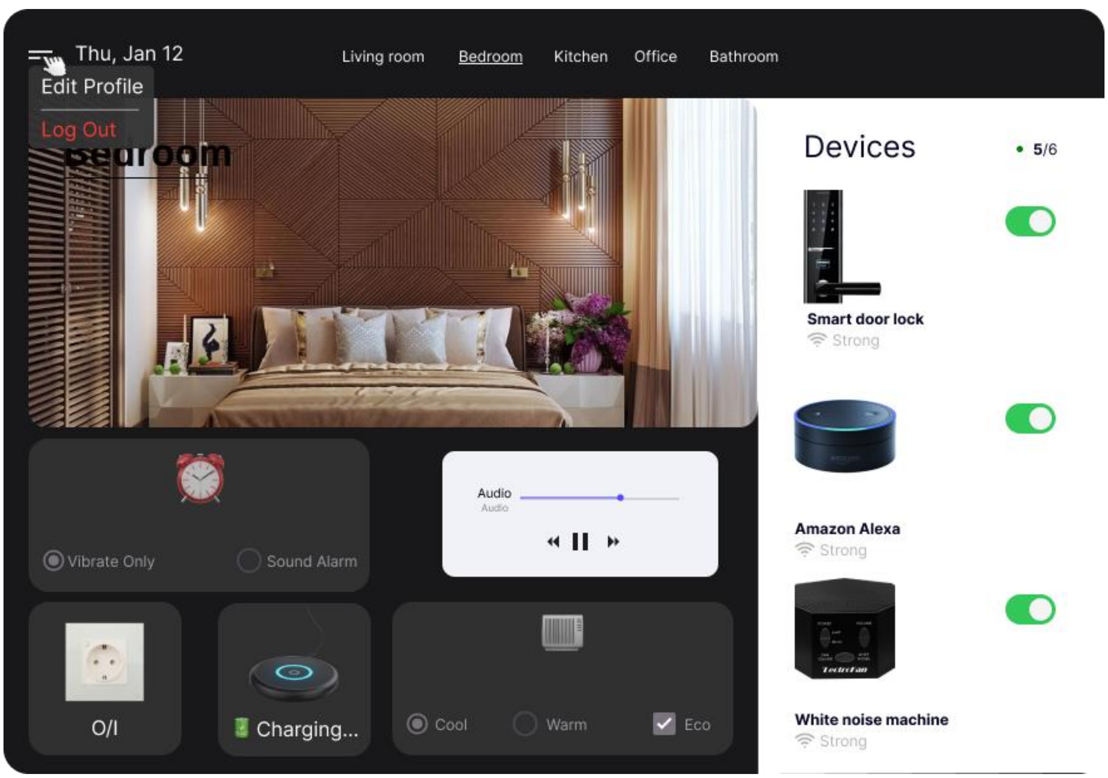

Figure 4.3.1.8.3: Bedroom Webpage

The design process emphasized consistency, accessibility, and ease of use, ensuring that users can navigate the system effortlessly.

4.4 Mobile Application

4.4.1 Platform Selection

4.4.1.1 Native vs Hybrid Apps

The mobile application for the Leap project will be developed using Flutter, a cross-platform framework by Google. Flutter was chosen over native development due to its ability to create high-performance applications for both iOS and Android platforms using a single codebase. This approach significantly reduces development time and costs while ensuring consistent functionality and design across devices. Additionally, Flutter's rich widget library and hot-reload feature streamline the development process, enabling rapid prototyping and iterative improvements.

4.4.1.2 Cross-Platform Compatibility

The decision to use a hybrid app framework (Flutter) over native development was driven by the need for cross-platform compatibility and efficient resource utilization. Native apps, while offering superior performance and access to platform-specific features, require separate codebases for iOS and Android, increasing development complexity and maintenance efforts. In contrast, Flutter provides near-native performance and access to device APIs while maintaining a single codebase, making it ideal for the Leap project's requirements.

4.4.2 App Development

The development of the mobile application focused on delivering an intuitive and user-friendly interface while integrating advanced functionalities. Flutter's widget-based architecture facilitated the creation of a responsive and visually appealing user interface (UI). The app's design adheres to modern UI/UX principles, ensuring ease of navigation and accessibility for all users.

4.4.2.1 User Interface and Experience

The app's UI was designed to prioritize simplicity and usability, with a clean layout and intuitive navigation. Key features such as real-time alerts, device controls, and health monitoring are prominently displayed, ensuring users can access critical functionalities with minimal effort. Flutter's customizable widgets enabled the creation of a cohesive and visually consistent design across all screens, enhancing the overall user experience.

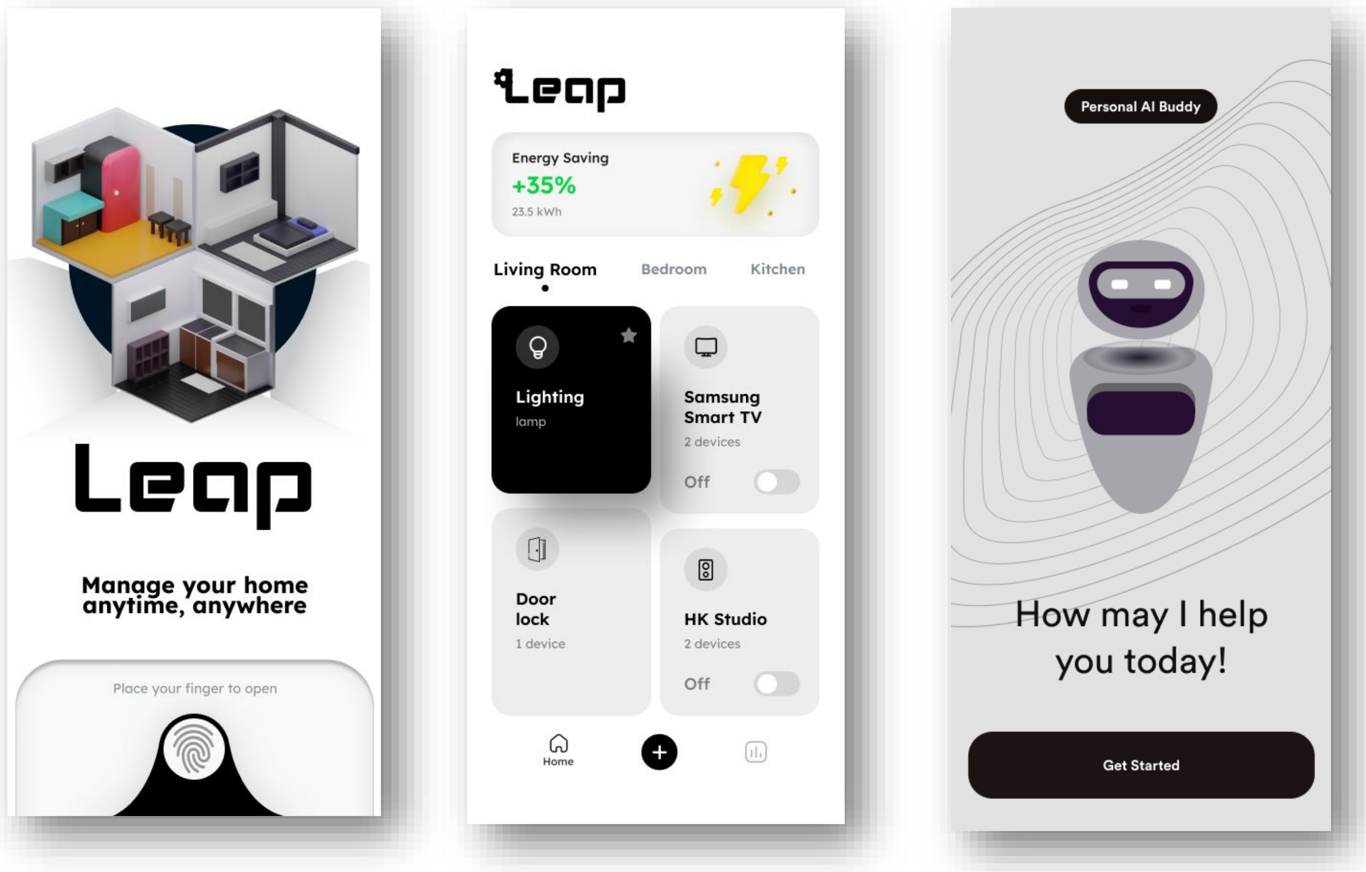

Figure 4.4.2.1: mobile UI

4.3.2.2 App Features and Functionalities

The mobile application integrates several core features, including real-time alerts from facial recognition, chatbot-based voice and text commands, home control and monitoring, and health metrics display. These functionalities are seamlessly interconnected, providing users with a unified platform for managing their smart home ecosystem. Flutter's robust ecosystem of plugins and packages facilitated the integration of third-party services such as Firebase for real-time data synchronization and cloud storage.

4.5 Databases

Here is an overview of which data may exist in our system:

Users Data: Stores personal information about users (name, email, phone, etc...).

Address Data: Store Address data such as (state, city, street, postalCode).

Home Data: Store Home data such as (Location, home type).

Room Data: Store what is the type of the rooms we have in each home.

Devices Data: Store the name of the device at each room and name of the device.

4.5.1 Database Design

In this section, we are going to analyze every data shape in our system in terms of database modeling.

Entities and Attributes:

4.5.1.1 User:

- User_id (PK): unique number to identify each User.
- FName, LName: the name of each user
- DOB: date of birth
- Phone1, Phone2: Phones of each user
- Gander: male or female
- E-mail: user registration email
- Password: user account password

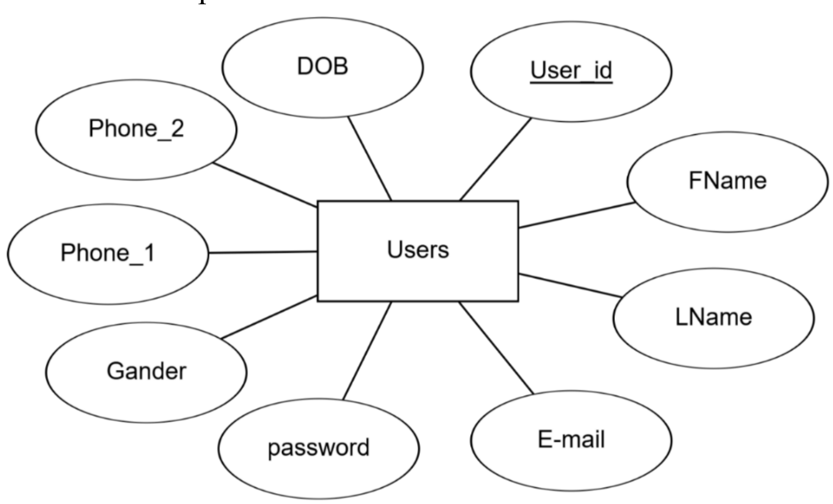

Figure 4.5.1.1: user Entity

4.5.1.2 Address:

- Address_id (PK): unique number to identify Address
- State: the user state
- City: the user city
- Street: the user street
- Postal_code: user postal code region

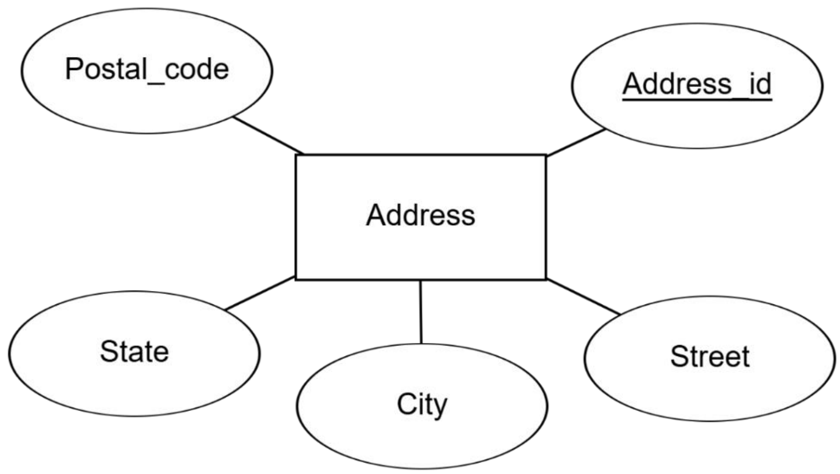

Figure 4.5.1.2: Address Entity

4.5.1.3 Home:

- Home_id (PK): unique number to identify Home in the database.
- User_id (FK): Foreign key referencing the User table.
- Address_id (FK): Foreign key referencing the Address table.
- Location: Home Location (is encoded from the address table: street, city, state)
- Type: Home type (house, department)

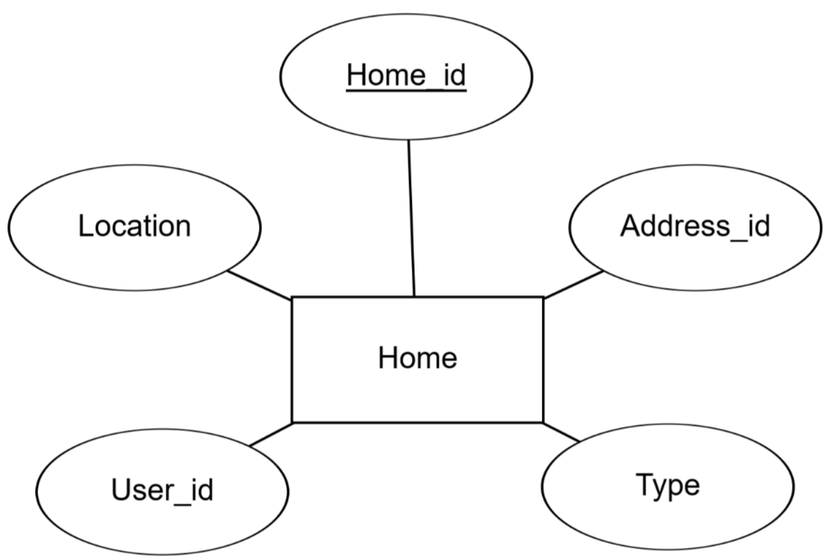

Figure 4.5.1.3: Home Entity

4.5.1.4 Room:

- Room_id (PK): unique number to identify Room in the database.
- Home_id (FK): Foreign key referencing the Home table.
- Type: Room type (living room, kitchen, bedroom)

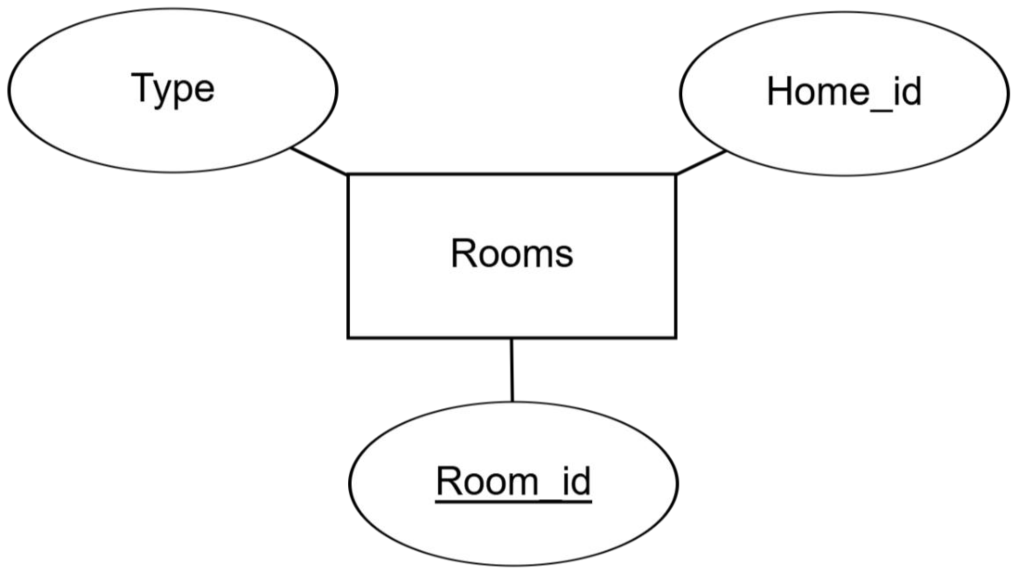

Figure 4.5.1.4: Room Entity

4.5.1.5 Devices:

- Device_id (PK): unique number to identify Device in the database.
- Name: The name of the device
- Type: Device type (camera, Wi-Fi, etc...)

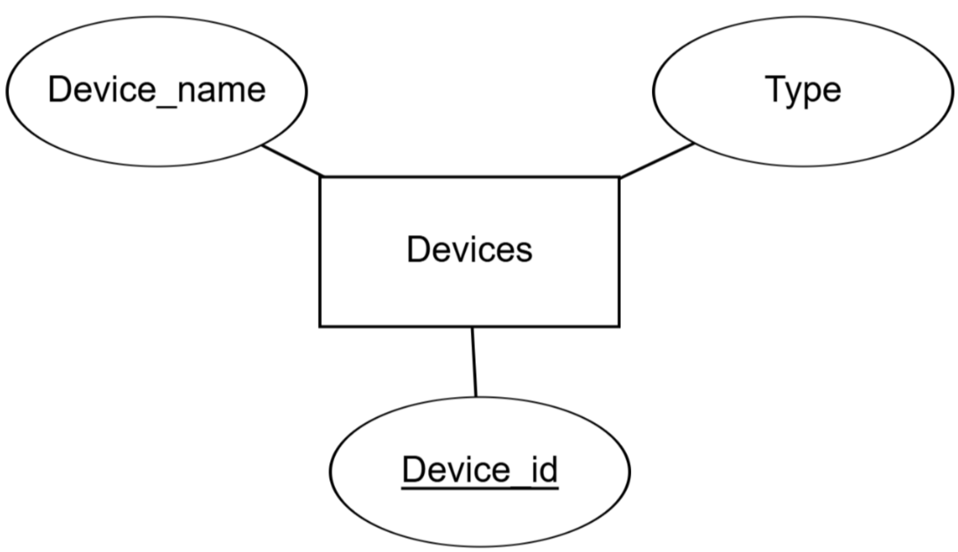

Figure 4.5.1.5: Devices Entity

4.5.2 Conceptual Data Model

4.5.2.1 Relationships Cardinalities:

Users & Address: (Many to Many)

1. For one User may have one or more than one address.
2. For one address may be owned by one or more Users.

Users & Home: (One to Many)

1. For one User may have one or more than one Home.
2. For one address must be owned by one User.

Address & Home: (One to Many)

1. For one Address may have one or more than one Home.
2. For one home must be Located in one Address.

Home & Rooms: (One to Many)

1. For one Home must have more than one Room.
2. For one Room must be in one Home (Room cannot contain The Home).

Rooms & Devices: (Many to Many)

1. For one Room may have one or more than one Device.
2. For one Device can be in one or more Room.

4.5.2.2 Fully Attributed Data Model (Full Entity-Relationship Diagram):

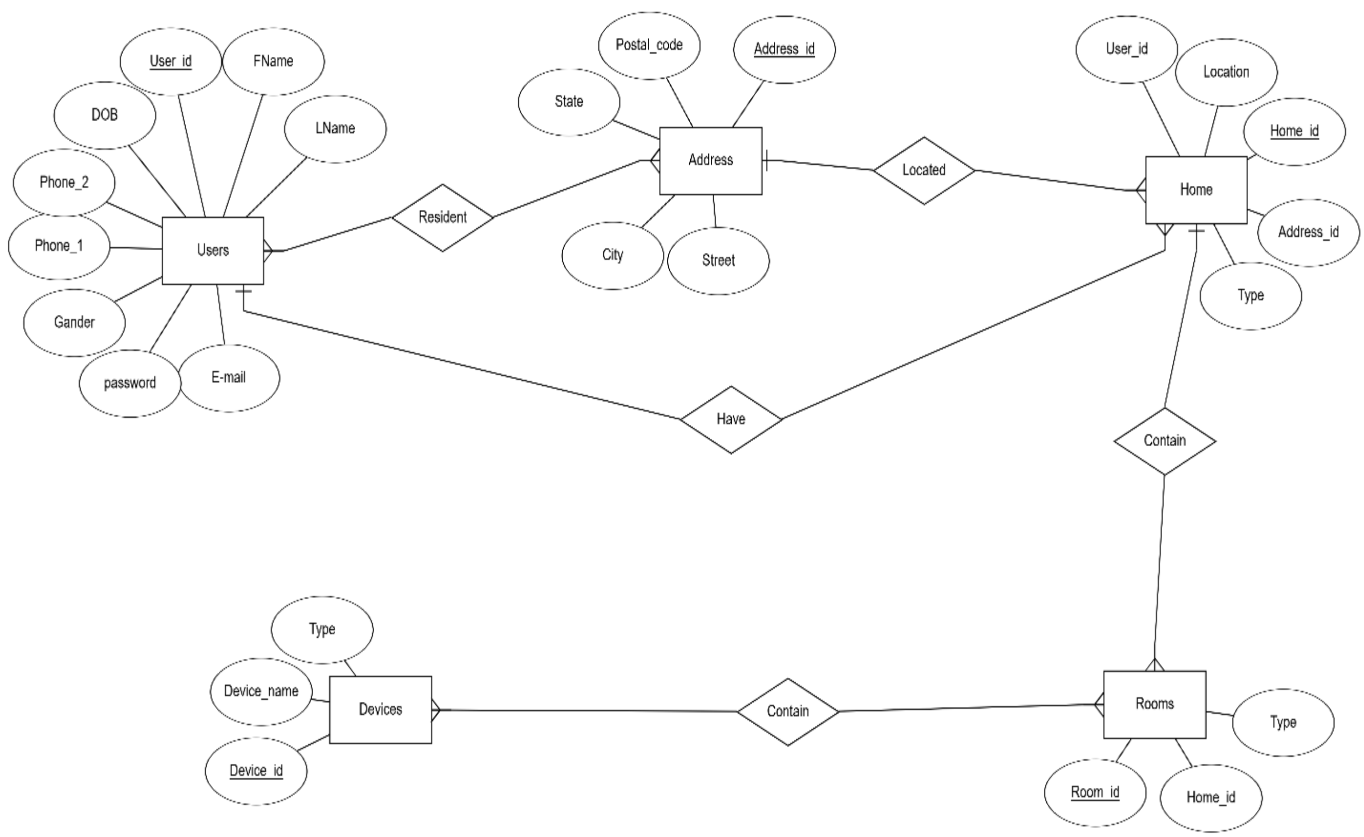

Figure 4.5.2.2 Fully Attributed data model

Chapter 5

IoT Devices

This chapter delves into the hardware and software components of the Leap IoT ecosystem, focusing on the Smart Ring and Smart Band systems. These devices form the core of the Leap project, enabling seamless interaction with smart home environments through gesture-based controls and health monitoring. The Smart Ring serves as a personal control device, allowing users to manage smart home components such as switches and door locks with intuitive gestures. Meanwhile, the Smart Band acts as a central hub, integrating gesture-based inputs with health monitoring capabilities to provide users with both convenience and valuable health insights.

5.1 Smart Rings

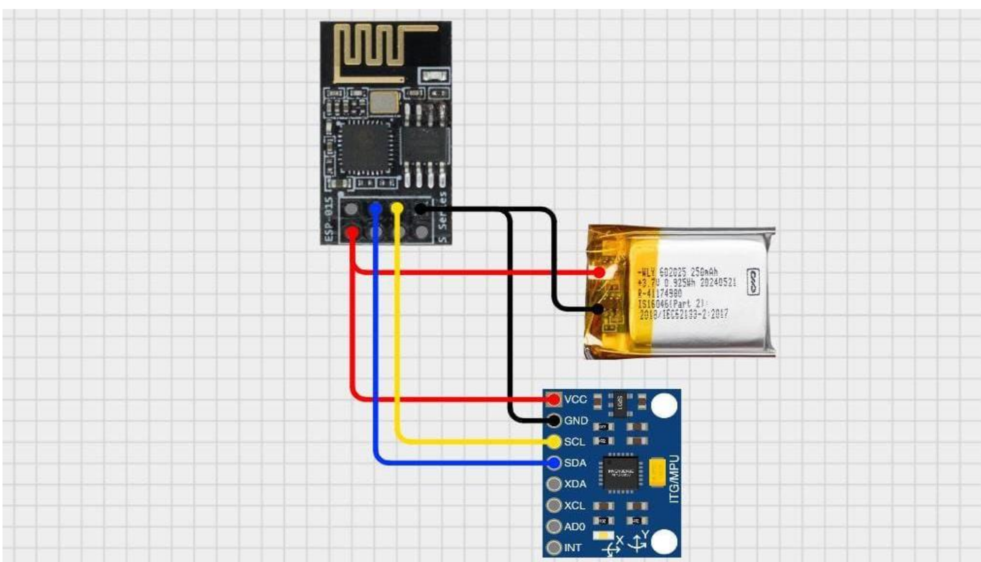

Figure 5.1: ring circuit

5.1.1 Use Case:

The Smart Ring serves as a personal control device for managing smart home components such as smart switches and smart door locks. By utilizing gesture-based inputs, the ring simplifies interactions with home automation systems, offering convenience and efficiency.

5.1.1 Hardware Components and Justification

The hardware components of the Smart Ring were carefully selected to ensure optimal performance, energy efficiency, and compactness. The ESP-01s serves as the primary microcontroller and Wi-Fi module, enabling the Smart Ring to connect to the network and communicate with the Firebase database. Its compact size, energy efficiency, and reliable Wi-Fi capabilities make it an ideal choice for small wearable devices like smart rings. For motion tracking and gesture recognition, the MPU-6050 6-axis motion sensor is employed. This sensor integrates both an accelerometer and a gyroscope in a small form factor, providing accurate and reliable motion sensing, which is critical for interpreting user inputs. Powering the Smart Ring is a Li-Po battery, chosen for its lightweight, compact, and rechargeable properties. These characteristics make it well-suited for wearable devices with limited space, ensuring portability and extended usage without compromising on size or weight.

5.2.3 Software

On the software side, the ESP-01s is programmed using the C++ programming language within the Arduino IDE for firmware development. The Firebase ESP Client Library is utilized to facilitate real-time communication with the smart home ecosystem, enabling seamless data upload and retrieval over Wi-Fi. This integration ensures that the Smart Ring can efficiently interact with the broader smart home infrastructure, providing a responsive and user-friendly experience.

5.2 Smart Band

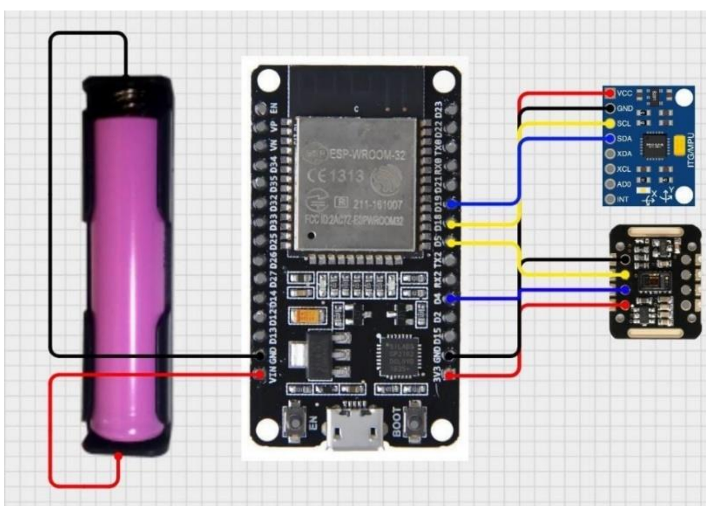

Figure 5.2: Smart Band Circuit

5.2.1 Use Case:

The Smart Band functions as a central hub for the Smart Rings while also serving as a health monitoring device. It measures vital signs such as heart rate and blood oxygen levels, providing users with both convenience and valuable health insights. This dual functionality enhances the user experience by integrating gesture-based control with health tracking capabilities.

5.2.2 Hardware Components and Justification

The Smart Band incorporates several carefully selected hardware components to ensure optimal performance and functionality. The ESP-32 microcontroller serves as the core processing unit, offering integrated Bluetooth and Wi-Fi capabilities for seamless communication with the Smart Rings and Firebase. Its high performance, support for multiple communication protocols, and low power consumption make it an ideal choice for advanced wearable devices. For motion tracking and activity monitoring, the MPU-6050 6-axis motion sensor is utilized. This sensor provides accurate motion sensing, enabling gesture-based inputs and contributing to the multi-functional nature of the Smart Band.

Powering the device is an 18650 Li-ion battery, chosen for its high energy density, long life cycle, and reusability. These characteristics ensure extended operation, making it suitable for continuous use in wearable applications. Additionally, the MAX30100 sensor module is employed for health monitoring, combining pulse oximetry and heart rate monitoring in a compact and cost-effective design. This integration allows the Smart Band to deliver reliable and efficient health tracking capabilities.

5.2.3 Software

The firmware for the Smart Band is developed using the C++ programming language within the Arduino IDE. This approach ensures efficient and modular code development. For data management and real-time monitoring, the Firebase ESP Client Library is utilized to upload health and motion data to Firebase via Wi-Fi. This integration enables centralized health monitoring and facilitates seamless interaction with the broader smart home ecosystem.

5.3 Smart Switch

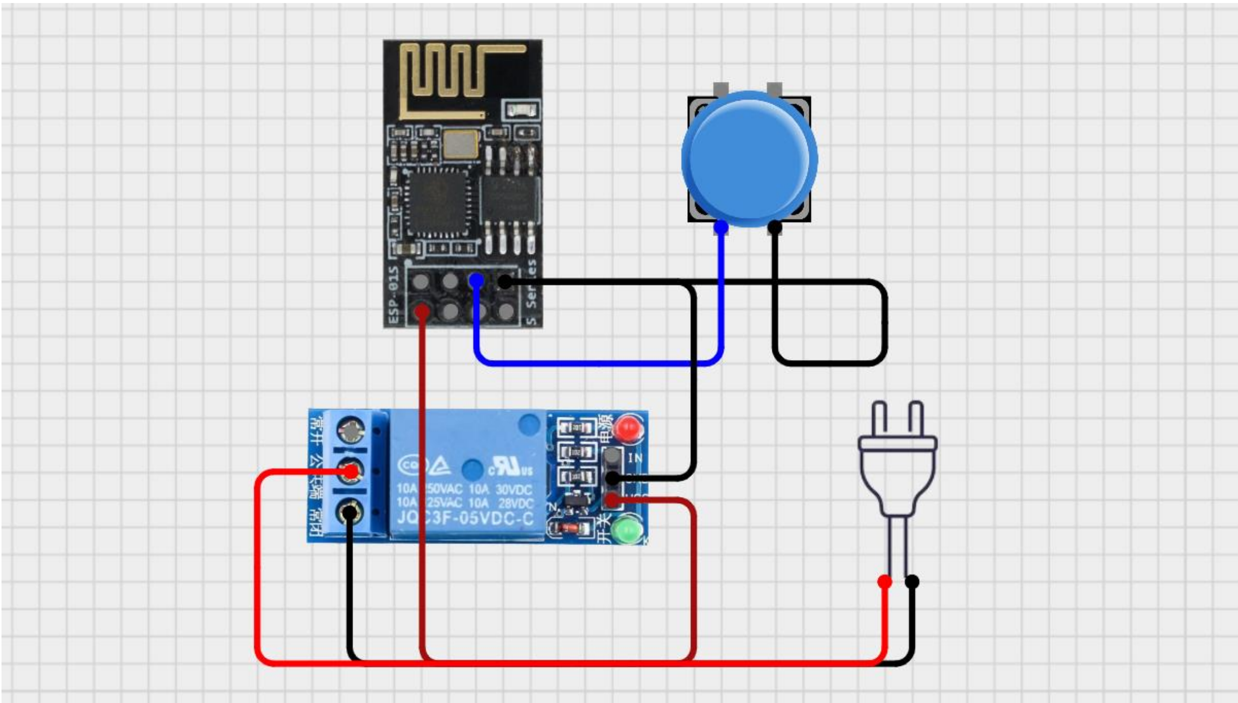

Figure 5.3: Smart Switch circuit

5.3.1 Use Case

The Smart Switch is designed to provide users with versatile control over electrical appliances through multiple interfaces, including a mobile application, website dashboard, Smart Rings, and manual controls. This multi-channel functionality ensures enhanced accessibility and usability, catering to diverse user preferences and scenarios.

5.3.2 Hardware Components and Justification

The Smart Switch incorporates several key hardware components to ensure reliable and efficient operation. The ESP-01s microcontroller serves as the core communication module, managing connectivity with Firebase and controlling the relay for appliance switching. Its compact size and integrated Wi-Fi capabilities make it ideal for embedding into the switch while maintaining efficient network connectivity. The relay acts as an electrically operated switch, controlling the flow of current to connected appliances. Relays are chosen for their reliability, ease of interfacing with microcontrollers, and ability to handle high-voltage appliances safely.

A custom-designed PCB (Printed Circuit Board) is used to house and connect all electronic components in a compact and organized layout. This ensures durability, simplifies assembly, and minimizes the overall size of the Smart Switch. Additionally, a push button is included to provide manual override functionality, allowing users to control appliances locally when smart features are unavailable. Push buttons are selected for their simplicity, reliability, and intuitive operation.

5.3.3 Software

The firmware for the Smart Switch is developed using the C++ programming language within the Arduino IDE, ensuring efficient and modular code development. For real-time data exchange and remote control, the Firebase ESP Client Library is utilized. This integration enables seamless communication with Firebase, allowing users to monitor and control connected appliances remotely through the mobile application or website dashboard.

5.4 Live Stream Camera

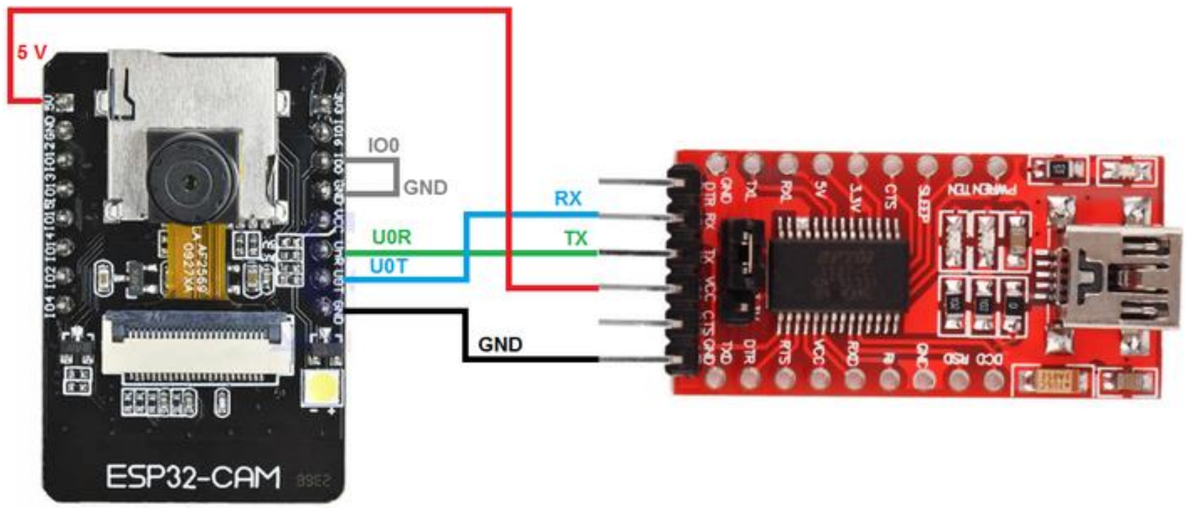

Figure 5.4: Live Stream Camera

5.4.1 Use Case

The ESP32-CAM module, programmed using an FTDI programmer, is employed for live streaming video to the facial recognition model. This low-cost, compact module integrates a built-in camera, making it highly suitable for IoT applications such as image processing, surveillance, and remote monitoring. Its ability to stream live video feeds directly to the facial recognition system enhances the project's real-time monitoring and security capabilities.

5.4.2 Hardware Components

The Live Stream Camera setup consists of two primary hardware components. The ESP32-CAM is a Wi-Fi-enabled microcontroller equipped with a built-in OV2640 camera, a microSD card slot, and multiple GPIO pins for extended functionality. This module is responsible for capturing and processing images and video, enabling seamless integration with the facial recognition system. The FTDI programmer is used to program the ESP32-CAM by establishing a connection between the module and a computer via USB. Its support for both 5V and 3.3V logic levels ensures safe and reliable interfacing with the ESP32-CAM during the programming process.

5.4.3 Software Setup

The software setup for the ESP32-CAM involves several steps to ensure proper functionality. First, the Arduino IDE is installed from the official Arduino website, and the ESP32 board package is added through the Boards Manager in the IDE preferences. Next, the required libraries, such as the ESP32 library, are installed to enable code compilation and flashing onto the ESP32-CAM. The Arduino IDE is then configured by selecting the correct board (AI-Thinker ESP32-CAM) and the appropriate COM port for the FTDI programmer. Finally, the code is uploaded to the ESP32-CAM by opening or creating a sketch in the Arduino IDE and pressing the 'Upload' button while ensuring the module is in flashing mode.

Chapter 6

Experimental Analysis

This chapter presents a detailed analysis of the experimental results obtained during the development and testing phases of the Leap project. It provides insights into the challenges encountered during system evaluation, focusing on the risks identified and the strategies employed to mitigate them. The primary goal of this chapter is to ensure that the Leap system can operate effectively across diverse environments, maintain robust security, and provide a seamless user experience.

6.1 Risk Analysis

The Leap project involves several technical, operational, and security risks that could hinder the overall performance, user experience, and adoption. A thorough risk analysis is critical to identifying potential issues and formulating strategies to address them. In this section, we examine key risks related to system performance, data privacy, and user adoption, and propose strategies to mitigate these risks in subsequent phases of development.

6.1.1 Technical Risks

System Reliability in Diverse Environments

One significant risk to the system's performance is its ability to function accurately in various environments, especially those with fluctuating lighting, background noise, or different user movement patterns. The system's accuracy may degrade in such conditions, leading to poor user experiences. To address this, adaptive machine learning models will be implemented, enabling the system to adjust dynamically to environmental changes. Real-world testing will be conducted in diverse settings to refine the system's performance, ensuring that it can handle different lighting conditions and noise levels. Additionally, the integration of environmental sensors, such as light sensors and microphones, will provide the system with real-time context to adapt its behavior accordingly.

Battery Life and Power Consumption

Another challenge arises from the limited battery life of the wearable components, particularly the smart ring, which may require frequent recharging. To mitigate this risk, the battery management system (BMS) will be further optimized to reduce power consumption during idle and active states.

Moreover, energy-efficient hardware components such as low-power inertial measurement units (IMUs) and Bluetooth Low Energy (BLE) will be incorporated to extend battery life. Power-saving modes will also be implemented to reduce energy usage during periods of inactivity, contributing to longer operational times between charges.

Data Latency

Delays in processing and transmitting data could result in significant user experience issues, particularly in applications that require real-time interaction. To reduce latency, local processing capabilities on the smart band will be enhanced, minimizing dependence on cloud-based systems and thus reducing transmission delays. Additionally, edge computing techniques will be utilized to process data closer to the source, enabling faster decision-making. Communication protocols, such as ESP-NOW and Bluetooth, will be optimized to ensure rapid data transmission between devices, further reducing latency and improving system responsiveness.

Speech Recognition Accuracy

Variations in accents, background noise, and audio quality may reduce speech-to-text accuracy, impacting the user experience. To mitigate this: Fine-tune models for diverse accents and dialects, implement noise reduction techniques to enhance audio clarity, use adaptive algorithms to learn user-specific speech patterns and conduct real-world testing in varied acoustic environments.

6.1.2 Security Risks

Data Privacy

Sensitive user data, including facial recognition embeddings and gesture patterns, is susceptible to unauthorized access, posing significant privacy risks. To safeguard this data, end-to-end encryption will be implemented for all data transmission and storage. This encryption will protect sensitive information from being intercepted or accessed by unauthorized parties. Furthermore, biometric data will be securely stored using secure enclaves or hardware-based security modules, reducing the likelihood of data exposure. Regular updates to the system's security protocols will be essential to address emerging threats and vulnerabilities, ensuring that the system remains secure over time.

Unauthorized Access

Another security concern is the potential for unauthorized users to gain access to the smart home system. To counter this risk, the facial recognition system will be enhanced with liveness detection, which can detect spoofing attempts using photos or videos. In addition, multi-factor authentication (MFA) will be implemented, combining facial recognition with other forms of authentication, such as PIN codes or gesture-based systems, to provide an additional layer of security. An intrusion detection system (IDS) will also be developed to monitor for unusual access patterns and alert users to potential security breaches, ensuring that the system remains secure from unauthorized access.

6.1.3 Usability and Adoption Risks

User Experience and Learning Curve

A major risk to the system's success is the potential for users to find the technology difficult to use, leading to low adoption rates. If the system's interface is unintuitive or complex, users may abandon it before realizing its full potential. To address this, user-centered design workshops will be conducted to gather feedback and refine the system's interface. Additionally, interactive tutorials and onboarding processes will be developed to guide users in learning how to use the system effectively. Adaptive user interfaces will also be implemented, personalizing the experience based on individual preferences and behaviors to ensure a smooth interaction with the system.

Resistance to Adoption

Some users may resist adopting the Leap system, particularly if they perceive it as unnecessary or overly complex. To overcome this resistance, the tangible benefits of the system, such as improved accessibility, convenience, and security, will be emphasized through real-life use cases and testimonials. Offering trial periods or demo versions will allow users to experience the system's capabilities firsthand, helping them understand its value. Furthermore, comprehensive customer support will be available to address any concerns and ensure a smooth transition to using the new technology.

Prompt Engineering Challenges

Prompt engineering is critical to ensuring the relevance and accuracy of AI chatbot responses, as the quality of input directly determines output performance. Challenges arise from ensuring prompts provide sufficient context for understanding, offering clear and unambiguous instructions, and defining response formats to meet user expectations. Poorly designed prompts can lead to irrelevant, incomplete, or incoherent outputs, impacting the chatbot's reliability and user satisfaction. Addressing these issues requires iterative prompt optimization, dynamic context incorporation, and the use of structured templates tailored to domain-specific needs. By mitigating these challenges, the chatbot can deliver precise, relevant, and well-structured responses, enhancing user engagement and functionality.

Chapter 7

Conclusion and Future Work

This chapter provides a detailed conclusion to the Leap project, summarizing the progress made during its initial phase, the challenges encountered, and the lessons learned. It also highlights potential areas for future enhancements to improve the system's functionality, scalability, and user experience. By structuring the chapter in this manner, it seeks to offer a comprehensive understanding of the project's current impact and its possibilities for future advancements.

7.1 Conclusion

The Leap project represents the completion of its first phase, focusing on laying the groundwork for a modular and scalable smart home automation system. This phase has aimed to address key aspects of the system, particularly gesture recognition and facial recognition, while conceptualizing the integration of a conversational AI chatbot. Although the system has not been fully implemented or integrated into web and mobile platforms yet, the progress made in this phase has established a robust foundation for subsequent development.

The development of the facial recognition component has been a central focus of this phase. The system's dataset preparation involved organizing anchor, positive, and negative samples to ensure effective training of the Siamese Neural Network. The use of the Labelled Faces in the Wild (LFW) dataset provided a robust foundation for the negative samples, while real-time image capture was utilized for anchor and positive samples. A convolutional neural network (CNN) was employed to extract facial features, and a Siamese network architecture was implemented to generate embeddings for accurate comparisons. OpenCV was used for real-time face detection and preprocessing, enabling the system to handle live video inputs effectively. These steps have demonstrated the feasibility of the facial recognition component, even though the model has yet to be fully trained and validated.

Gesture recognition development has also seen significant progress. The project has focused on collecting a custom dataset using IMU sensors to capture hand and wrist movements. Preprocessing steps, such as normalization, noise reduction, and segmentation, were applied to prepare the data for model training. A preliminary framework for implementing machine learning models, such as Radial Basis Function networks and Restricted Coulomb Energy classifiers, has been established, but these models remain to be fully tested and evaluated.

The chatbot component, while not yet developed, has been conceptually planned as an integral part of the system. Powered by conversational AI, it is intended to provide users with personalized assistance and seamless interaction with smart home devices. Future phases will focus on implementing and refining this component.

Although the integration of these components into a unified web and mobile platform has not yet occurred, this remains a key objective for the next phase. The aim is to create an intuitive and accessible interface that allows users to control and monitor their smart home devices seamlessly.

7.2 Future Work

As the Leap project advances, several key areas of improvement and expansion will be prioritized to enhance functionality, usability, and scalability. The **gesture recognition system** will be expanded to include a broader library of gestures, enabling more complex and customizable controls. Additionally, user-specific gesture learning will be implemented to improve personalization and accuracy. For the **facial recognition system**, future efforts will focus on optimizing performance in dynamic environments, such as those with varying lighting conditions or crowded spaces. Integration of environmental sensors, such as light sensors, will allow the system to adapt in real time. Multi-user support will also be introduced to handle collaborative scenarios, while **liveness detection** will be integrated to prevent spoofing attacks, leveraging advanced computer vision techniques.

The **chatbot component** will be enhanced with contextual understanding and personalization features. By fine-tuning the model on larger and more diverse datasets, the chatbot will deliver more coherent and relevant responses. Its seamless integration into web and mobile platforms will ensure a unified user experience. New functionalities, such as **Daily Task Scheduling and Reminders** and **Bill Analysis and Spending Reports**, will be developed to expand the system's usability. The latter will utilize OCR technology to extract costs from bill photos, analyze monthly expenses, and generate detailed spending reports with actionable recommendations.

On the **technical front**, the system will be designed for scalability and cross-platform compatibility. A **microservices architecture** and load balancing proxies will be employed to handle increased user demand efficiently. The development of a desktop application will further ensure cross-platform compatibility. Integration with IoT devices, such as smart armbands and rings, will be optimized using Firebase's real-time database, while lightweight machine learning models will be deployed on edge devices to enhance real-time performance and energy efficiency.

Finally, **user experience and accessibility** will be improved through the integration of voice control capabilities, haptic feedback, and voice-guided navigation. These features will ensure the system remains inclusive and practical for a diverse range of users, while future enhancements, such as additional IoT device integrations and advanced AI functionalities, will further solidify the system's position as a versatile and user-centric solution.

7.3 Broader Implications

The Leap project's technologies have potential applications beyond smart home automation. Gesture recognition could be employed in healthcare to control medical devices or assist patients with mobility impairments. Facial recognition systems could enhance security and efficiency in industrial settings by controlling access to sensitive areas. The chatbot component could serve as a virtual teaching assistant in education, providing personalized learning experiences for students. These broader implications highlight the versatility and potential impact of the technologies developed in the Leap project.

7.4 Final Thoughts

As smart home automation continues to evolve, the Leap project's first phase serves as a foundation for future advancements in human-computer interaction and IoT technologies. The insights gained and challenges addressed during this phase will guide subsequent development efforts, paving the way for a more innovative, accessible, and secure smart home solution. By addressing current limitations and leveraging emerging opportunities, the Leap project aims to redefine smart home standards and enhance the quality of life for its users.

8.1 Online Resources and Tools

Gesture Recognition Tools

- **TensorFlow**: https://www.tensorflow.org/

  A powerful open-source library for machine learning and deep learning, used for building and training gesture recognition models.

- **Keras**: https://keras.io/

  A high-level neural networks API, used for simplifying the implementation of deep learning models.

Facial Recognition Tools

- **OpenCV**
- **Tensorflow**
- **Keras**

Chatbot Development Tools

- **Google Colab**

- **Torch**: https://pytorch.org/
    - **PyTorch** is a deep learning framework that supports the training and deployment of neural networks. It is integral to running the Llama model, as it powers the computations required by these pre-trained models, enabling your project to handle large-scale, complex AI tasks efficiently.

- **Hugging Face Transformers**: https://huggingface.co/transformers/
    - A library for state-of-the-art natural language processing (NLP), used for developing the chatbot.

Speech Recognition

- **SpeechRecognition (sr):** https://pypi.org/project/SpeechRecognition/2.1.3/
    - This module is used to convert audio speech into text. It enables speech-to-text functionality in our project, which is valuable for applications like voice-controlled assistants, transcription services and enhancing the accessibility of our system for users with disabilities.

8.2 Datasets

8.2.1 Gesture Recognition Dataset:

Custom Dataset: A custom dataset of 1,0000 hand gesture samples collected, used for training, and evaluating the gesture recognition system.

8.2.2 Facial Recognition Dataset:

Labelled Faces in the Wild (LFW): http://vis-www.cs.umass.edu/lfw/

8.2.3 Chatbot Training Data

Conversational Dataset: A dataset of conversational data, including user queries and responses related to smart home control, used for training the chatbot.

References

8.3 Academic Journals and books

1. **M Kim, J Cho, S Lee, Y Jung. (2019).** *IMU Sensor-Based Hand Gesture Recognition for Human-Machine Interfaces.* [MPDI] [Google scholar]
2. **Mouser Electronics Blog (2023).** *IMU Basics.* [Mouser].
3. **Pololu Robotics and Electronics (2023).** *6-DOF IMU Data Guide.* [Pololu].
4. **P Senin (2008).** *Dynamic Time Warping Algorithm Review.* [ResearchGate] [Google Scholar].

5. **DS Cardona-Pineda, JC Ceballos-Arias, JE Torres-Marulanda, MA Mejia-Muñoz, Boada, (2022).** *A. Face Recognition-Eigenfaces*. [ResearchGate] [Google Scholar].
6. **MK Roy, P Dwibedi, A Singh, RP Chakraborty (2023).** *MTCNN and FACENET-Based Face Detection and Recognition Model for Attendance Monitoring* [Conference paper] [Google Scholar].
7. **R Kumar, S Kumar. (2023 82, 1–20).** *A Deep Learning Approach for Facial Recognition in Smart Homes. Multimedia Tools and Applications*. [Springer] [Google Scholar].
8. **Smith, J.; Lee, H. (2023).** *Challenges in Facial Recognition: Lighting and Pose Variations.* [Society of Actuaries].
9. **A Giaretta - Virtual Reality, (2024, 28, 1–15).** *Privacy-Preserving Facial Recognition for Smart Homes. Virtual Reality.* [Springer]. [Google Scholar].
10. **D Almeida, K Shmarko, E Lomas. (2021, 1, 1–12).** *Ethical and Technical Considerations in Facial Recognition Systems. AI and Ethics.* [Springer] [Google Scholar].
11. **K. Gregory, Z. Richard, S. Ruslan. (2015).** *Siamese Neural Networks for One-shot Image Recognition* [University of Toronto].
12. **J Duque Domingo, R Medina Aparicio, LM Gonzalez Rodrigo (2021, 11, 7839).** *Improvement of one-shot-learning by integrating a convolutional neural network and an image descriptor into a Siamese neural network.* [mdpi] [Google Scholar].
13. **D. Sebastián Cabezas, R. Fonseca-Delgado. (2024).** *Integrating a LLaMa-based Chatbot with Augmented Retrieval Generation.* [ResearchGate].
14. **Sumit Kumar Dam, Choong Seon Hong. (2024).** *A Complete Survey on LLM-based AI Chatbots.* [ResearchGate].
15. **Stanford**. *Agile Development Methodology* [Article].
16. **Hugging Face.** *chavinlo/alpaca-native Chatbot Model* [Article]
17. **Indira KriGan (2023).** *Natural Language to SQL using an Open Source LLM* [Medium].
18. **Muhammad Usama Khan (2024),** *How to build your Custom Chatbot with Llama 3.1 using Ollama and OpenWebUI* [Medium].