İstanbul Bilgi Üniversitesi

Smart Home with App

Mustafa Deniz Demirhas

Emre Cem Kenarcı

Lecturer - Elif Pınar Hacıbeyoğlu

October 2023 – January 2024

Contents:

Introduction And The Purpose Of The Project:

Humans are a species gifted with the ability of imagination. It is useful to visualize thoughts and facilitating a deeper understanding. Occasionally, the ability to imagine culminates in the creation of concrete works that depict these ideas, serving as a catalyst for the creation of prototypes or smaller-scaled models of envisioned products. These models can help one gain a deeper understanding of the concept meanwhile acting as a way to demonstrate ideas.

In current times, the Internet of Things (IoT) has emerged as a phenomenon that has been rapidly and constantly getting more involved in the daily life, and automation is something that has never been this widely available before. Smart homes, in that sense, are a great example to show this trend of automation, since it directly shows how much IoT is solidifying in the daily lives. Home is where humans spend the majority of their time, hence if homes start changing in a certain way, it can be used as evidence to outline current trends. So homes getting smarter shows how IoT is also expanding.

The aim of this project is to construct a miniature replica of a smart home controllable through a mobile app. This endeavor serves as a practical display of IoT functionality and home automation, fostering a deeper appreciation of their significance. This project aims to highlight how home automation can enhance the general way of doing things. It is a way to see the scope of IoT systems and their benefits.

The Importance Of The Project:

The importance of this project lies within the fact that it serves as a means to comprehend and illustrate the potential of IoT technology. Creating a small-scale replica of a smart home controllable through a mobile app allows for a practical display of how automation can enhance daily life beyond traditional methods.

As it is implemented, the project will provide valuable insights into how home automation systems function, revealing the underlying concepts and technologies that facilitate their operation. It offers an opportunity to gain experience in home automation and give a solid foundation for future works to come while at the same time acting as a solid representation of the theories that make this project possible. Furthermore, throughout the process of completing this project, the contributors can acquire a unique type of expertise regarding the events that transpire during the project's development stage. Initially, the contributors will undergo the process of conceptualizing and developing a comprehensive plan. As the project advances, the process will transition from the planning phase to constructing a functional prototype. It’s during this stage that the contributors will gain hands-on experience with IoT systems and home automation technologies. Following implementation, the team will move into the testing phase. Here, the contributors will gain an understanding of the significance of quality assurance. Finally, the project will conclude with a review and presentation phase. The team will evaluate the project’s success, identify areas for improvement, and present their work to others. This experience will enhance their communication skills and their ability to critically analyze their work. Overall, all of this will culminate in adding invaluable experience to the contributors.

Roles:

Mustafa Deniz Demirhas – Project Manager, Designer
Emre Cem Kenarcı – Scrum Master, Designer, Developer,
Tester

This project was entirely designed, developed, and tested
by Emre Cem Kenarcı and Mustafa Deniz Demirhas.

Timeline:

Plan & Design: Plan and detailed design of the project.
25/10/2023 – 19/11/2023
Implementation: Mobile app design and mechanics.
20/11/2023 – 24/12/2023
Testing: Testing the product and adjusting.
25/12/2023 - 31/12/2023
Final presentation: Final presentation and final package documenting.
01/01/2024 – 10/01/2024

Requirement List:

Identifier	Priority	Requirement
Feature 1	5	Device Control - Ability to control lights, door lock, and fish feeder.
Feature 2	5	Temperature-Gas Leak Monitoring - View and monitor temperature data and gas leak.
Feature 3	5	Locking Mechanism - Implement a locking mechanism for the door.
Feature 4	4	Notifications - Receive alerts for important events (gas leak, temperature)
UI Requirement 1	5	*Home Screen - Display overview of home status, including lights, door lock, current temperature, and active notifications.*
UI Requirement 2	2	Device Control Screen - Separate screens for controlling lights, door lock, and watering system.
UI Requirement 3	3	Temperature-Gas Leak Monitoring Screen - Display current and historical temperature readings.
UI Requirement 4	2	Notification Screen - Show all active notifications and alerts.
UI Requirement 5	1	Settings Screen - Configure app settings

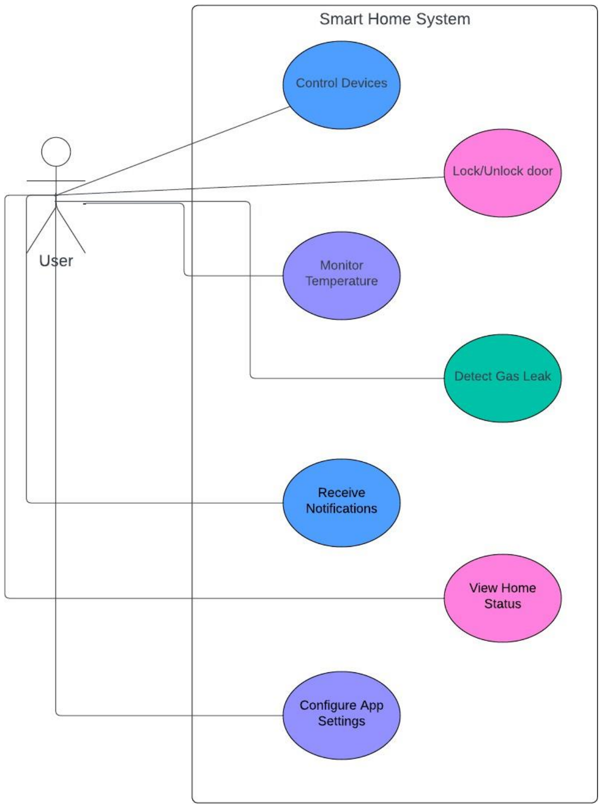
Table 1: Requirements List for the project

As it can be seen from the requirements list, the ability to control and monitor data from the miniature home is top priority and must be achieved. Notifications are also quite important as some data from the home (e.g gas sensor) can be vital in real life for saving lives.

Every data must be accessible via a home screen and historical data is a medium-importance target to have in the final product. Meanwhile seperate screens for each device and a notification screen can be implemented if deemed worthwhile. A settings screen is possible, but it is the least likely one to be implemented in the final product.

Full Design:

a) Diagrams and Test Cases Table:



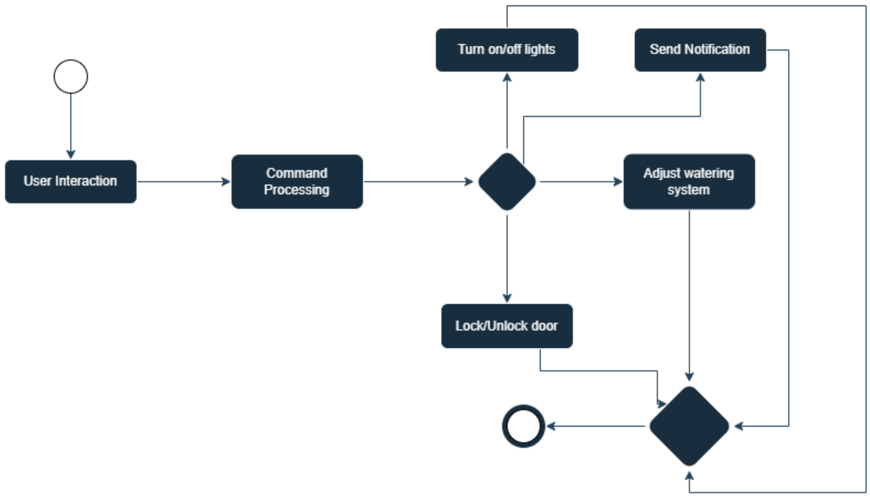


Figure 2: Activity Diagram

Figure 1: Use Case Diagram

Test Case	Steps	Expected System Response
Turn on Light	1. Open mobile app 2. Navigate to “Lights” screen 3. Tap on “Turn On” button for a light	The corresponding light in the smart home model turns on
Lock Door	1. Open mobile app 2. Navigate to “Locks” screen 3. Tap on “Lock” button for the door	The door of the smart home model locks
Temperature Monitoring	1. Open mobile app 2. Navigate to “Temperature” screen	The current temperature readings from the sensors are displayed
Receive Notification (Gas Leak)	1. Cause a gas leak in the smart home model	The mobile app sends a notification about the gas leak
Control Watering System	1. Open mobile app 2. Navigate to “Watering System” screen 3. Tap on “Activate” button	The watering system in the smart home model activates
Receive Notification (Fire Hazard)	1. Cause the temperature to rise to a certain degree	The mobile app sends a notification about the fire hazard

Figure 3: Test cases and expected system response table

These diagrams are used to showcase the capabilities of the system. Using Figure 1, what the system will let the user do can be showcased. Figure 2 shows the steps of the execution of a user interaction in the system. Meanwhile Figure 3 shows the expected result of each action taken in the system.

b) First Template UI

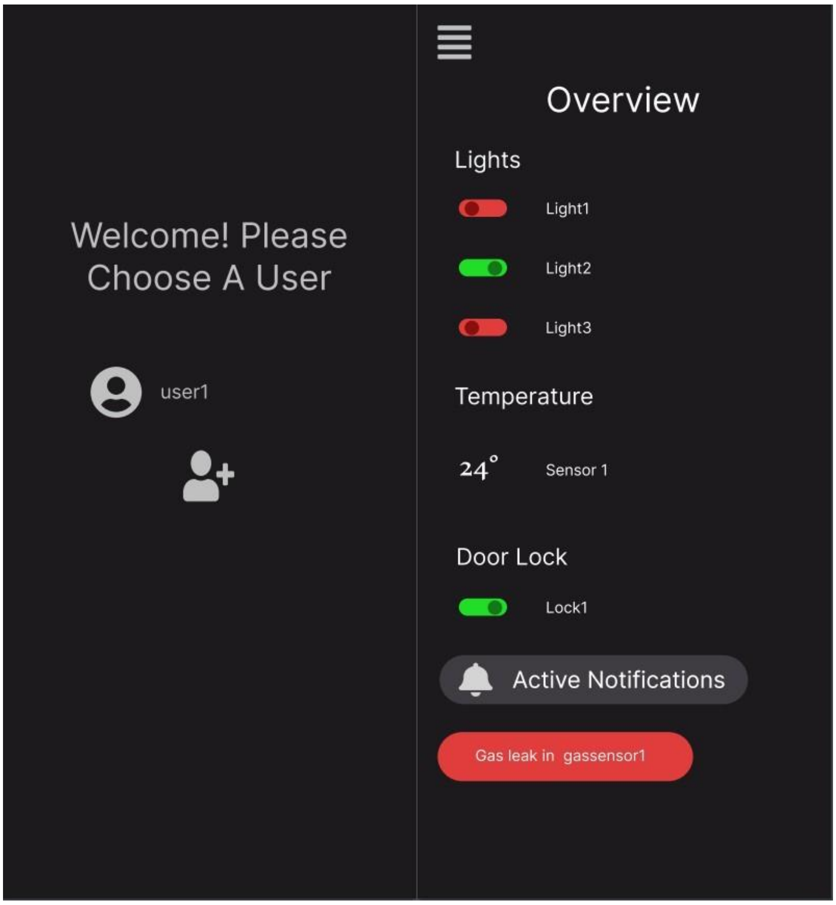


Figure 4 shows the initial UI template design of the mobile app. As it was the very first template UI that was designed before selecting the enviroment where the app would be designed, it looks mostly different from the final UI. But this temlate UI was used as a guideline to determine how the final UI would look.

Figure 4: Project Template UI

Construction Of The Project:

First step of construction was to select all the components that would be used. Next step was to make a circuit diagram involving all these component. After those steps, the blueprints of the miniature home would be made.

a) Component Selection:

-Arduino UNO: As the brain of the smart home system, Arduino UNO will be used to control and coordinate all other components. It will process sensor data, and communicate with the mobile app.

-ESP-8266: The ESP8266 is a microcontroller module equipped with integrated Wi-Fi capabilities. Supports 802.11 b/g/n Wi-Fi standards, enabling the transfer of information over the internet.

-4x LED Lights: These will be the lighting system in the smart home. They can be turned on or off via the mobile app.

-Servo motor 9g(for door lock): This motor will act as an electronic door lock. It will rotate to a specific angle to “lock” or “unlock” a miniature door in the smart home model.

-2x LM35: Lm35 is an analog sensor used for temperature mesurement. The sensor responds changes in temperature and mesure temperature values in millivolts(mV). It can temperatures between -55 and +150 degrees.

-MQ-4: The MQ-4 gas sensor is a device that detects varius gases: natural gas, Propane, Butan. Main function of this sensor is to mesure and detect the presence of these specific gases in air and provides this information as an analog or digital output signal.

-Servo motor 9g(for fish feeder): Another servo motor will be utilized for a fish feeder mechanism. It will dispense fish food automatically.

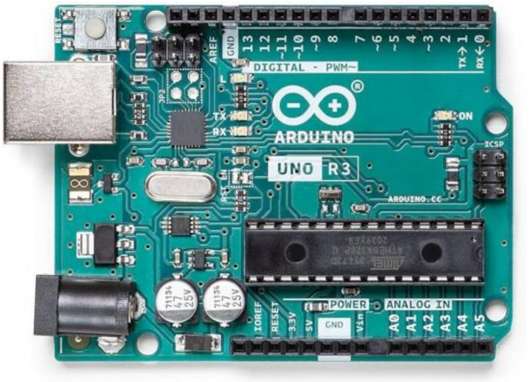


Image 1: Arduino UNO

Image 2: LED Light

Image 3: LM35

Image 4: Servo Motor 9g

Image 5: MQ-4

b) Miniature Home Blueprint:

In the Figure 1, it is the blueprint of the home miniature. This is how the home will be made and the shape it will look in. In the figure, the red rectangle represents the door lock system. It is the door that is at the main entrance of the home. The blue diamonds represent the temperature sensors. One in the common area and one in the hypothetical bedroom area. The orange circles represent the LED lights. There is one in each room. The green rectangle represents the gas leak detector. Finally the purple square represents where the automated fish feeder will be positioned.

The home miniature itself will be constructed with 6mm MDF material and assembled using strong adhesives.

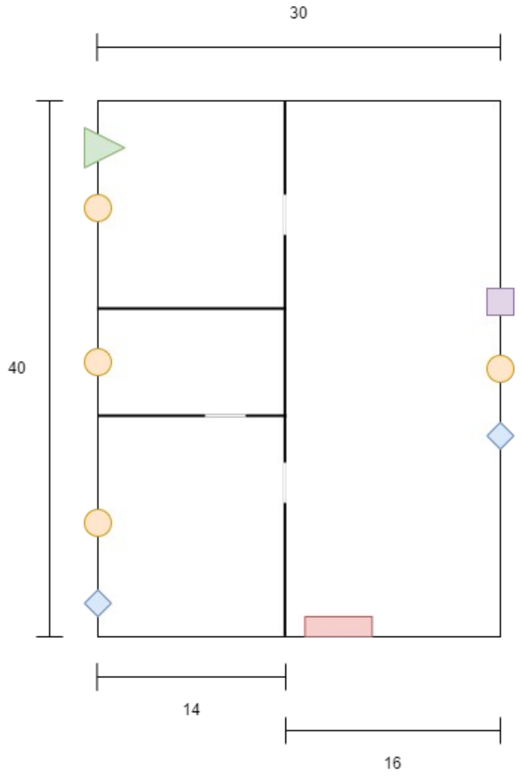


Figure 5: Home Miniature Blueprint

c) Circuit Diagram:

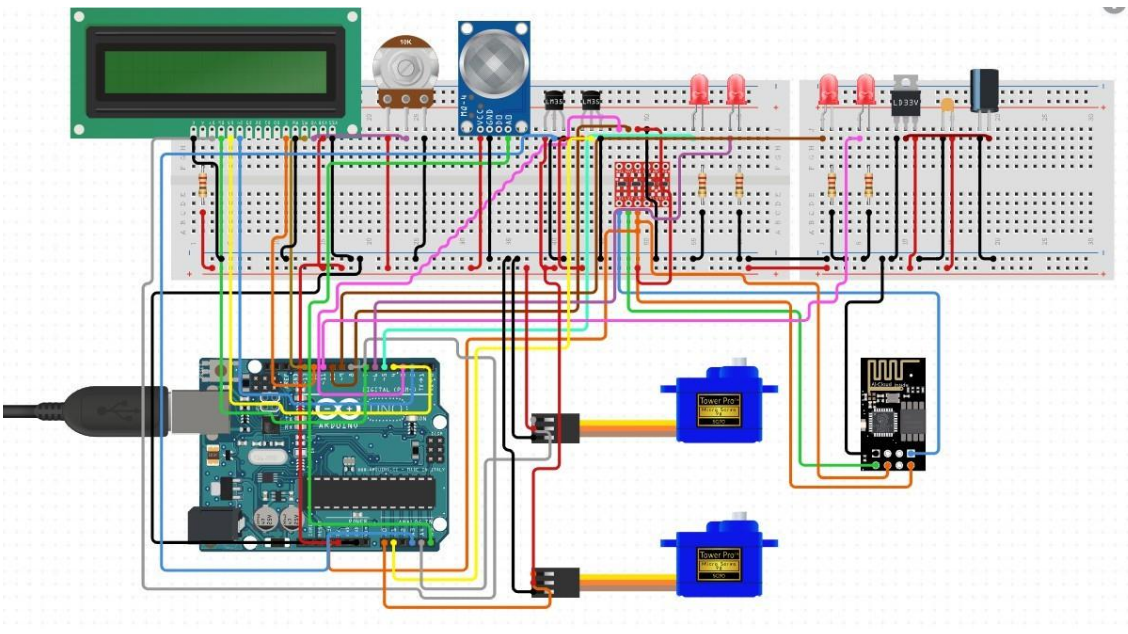


Figure 6: Circuit Diagram

Figure 6 shows the circuit diagram of every component in the system. There are 2 Servo Motors, 4 LED lights , MQ-4, LM35 on the Arduino UNO. This diagram serves as a guide for testing the system's functionality prior to installing the components into the miniature home. There is also a ESP8266 WiFi module in the diagram to maintain connection to the app using WiFi.

d) Mobile App Platform:

After these steps were done too, next a platform to develop the mobile app on was to be selected. For that, it was decided the Blynk platform will be utilized to create a mobile application that enables user interaction with the home's devices. Blynk provides an intuitive interface for designing custom app dashboards, which can be employed to monitor sensor data and control actuators such as lights and motors. The app will be connected to the Arduino UNO via WiFi, enabled by the ESP8266 module. This connection will enable communication between the app and the smart home system. Live data from the temperature sensors and gas leak detector will be viewable, and the LED lights, door lock, and fish feeder will be controllable from within the app. Upon availability of the hardware, the Blynk library will be integrated into the Arduino sketches to establish the connection between the app and the smart home devices.

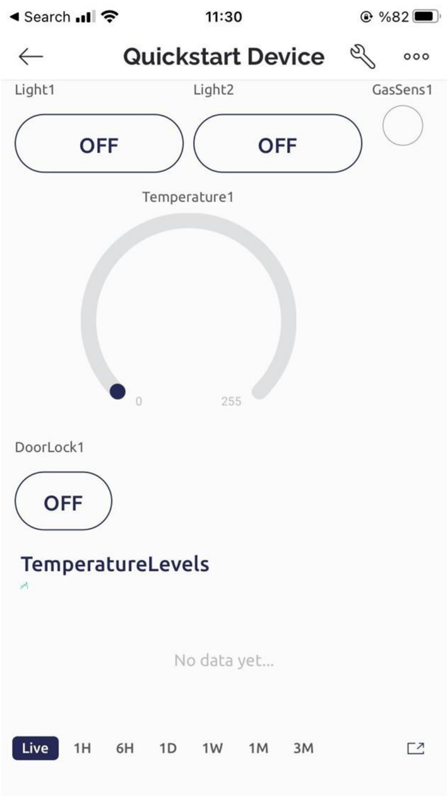


Figure 7: Template UI using Blynk

Figure 7 shows how a UI Template can be created using Blynk. This template can give a close look on how the final project may look like.

e) Creating The System:

After no more steps were left for preperations, it was tome for building the system. An iterative approach was used in making the system: each component were to be implemented on its own first, then combine it with every component that was implemented before it, reaching closer to the final product.

First components that were implemented were the LEDs, as they were seemingly the easiest of the bunch. Controlling the LEDs don’t require any codes since the Blynk platform can directly control the input pins where LEDs were connected, so there is no code for controlling LEDs.

```
// Custom function to read temperature from sensors
float readTemperature(int pin) {
  int value = analogRead(pin);
  float voltage = value * (5.0 / 1023.0);  // Convert the analog reading to voltage
  float temperature = voltage * 100;  // Convert the voltage to temperature
```

Figure 8: Code for reading temperature

Second components that were implemented were the temperature sensors. Figure 8 shows the main code that reads data from the temperature sensors.

```
// Function to send sensor data to the app
void sendSensorData() {
  int data = analogRead(smokeA3);
  Blynk.virtualWrite(V0, data);   //Sending data from gas sensor

  //The notification function for the gas sensor. 500 is the critical value for sending a notification
  if(data>500){
    Blynk.logEvent("gas_leak_1", "GAS LEAK DETECTED");
  }

  float temperature1 = readTemperature(lm35_1_Pin);
  if (temperature1 != -1.0) {             //For more information about this if statement, check the readTemperature code below
    Blynk.virtualWrite(V3, temperature1); //Sending data from temperature sensor
  }
  delay(400);

  float temperature2 = readTemperature(lm35_2_Pin);
  if (temperature2 != -1.0) {
    Blynk.virtualWrite(V4, temperature2);
  }
```

Figure 9: Code to send temperature and gas data to app

Next step was implementing the gas sensor. In Figure 9, the code sends the data from Figure 8 to the app and simultaneously reads from the gas sensor and sends that to the app too.

```
// Custom function to control servos based on value from the app
void controlServo(int virtualPin, Servo servo, int ServoState) {
  if (ServoState == 1) {
    servo.writeMicroseconds(2000);    //Here, if we send 1 (meaning on switch) from the app, servos will rotate in a direction.
    delay(167);                       //This delay function determines how long do you want servo to rotate. delay(167) corresponds to 90 degrees in real world.
    servo.writeMicroseconds(1500);
  } else {                            //If the input is 0(meaning off switch) we turn the servo back to its original position.
    servo.writeMicroseconds(1000);
    delay(167);
    servo.writeMicroseconds(1500);
  }
}
```

Figure 10: Code to control the servo motor

The final components were the 2 servo motors. Figure 10 shows the code that control the servo motors. After it was done too, only step was to transfer these to the miniature home.

While all the code developments were proceeding, the miniature home was also being built. First each MDF piece were cut into appropriate sizes, then connected together with a strong adhesive. Image 6 shows an early assembly of the miniature home using tapes to visualize the model.

Image 6: Early home assembly

When the miniature home also reached its final form, all components were transferred to the home and thus completing the project.

Challenges Encountered And Suggested Solutions:

During the development there were 3 main challenges that required solving. The first of those was related to the ESP8266's default baud rate. It was observed that the default baud rate was too high, which led to data loss. An attempt was made to solve this issue by configuring the ESP module. However, this approach was unsuccessful as the ESP module did not support the necessary commands for configuration.To overcome this obstacle, a decision was made to flash the ESP module to a different firmware. This allowed for the modification of the baud rate, effectively resolving the data loss issue.

The second significant challenge encountered during the project was related to the servo motors used in the system. The servos were continuous servos, which are controlled differently than standard servos. For instance, writing the value 90 to them does not result in a 90-degree turn as it would with a standard servo.

Initially, this caused confusion. However, upon realizing that the servos were continuous, the code was adjusted accordingly to achieve the desired functionality.

The final major challenge encountered during the project was related to the power supply. The system was initially powered by a laptop. This approach worked well in the early stages of the project due to the iterative approach taken (first only LEDs, then LEDs + LM35s, then LEDs + LM35s + MQ4). However, after connecting the servo motors, which were the last components to be added, the system required more power than the laptop could provide. This resulted in some components not behaving as expected, causing confusion as it was not immediately apparent that this was a power issue. Upon realizing the problem, a solution was implemented by connecting a 5V voltage regulator to the system and using power from an AC adapter. This effectively resolved the power supply issue, ensuring all components of the system functioned as intended.

Tests And Results:

Each component in the system was initially tested individually using simple Arduino code to ensure there were no malfunctions. For instance, the servo motors were tested to verify their rotation and response to commands. Similarly, the LM35 temperature sensors were tested to confirm the accuracy of their readings.

Once individual testing was completed, the system was tested as a whole. This involved checking the integration of all components and ensuring they worked together as expected. For example, tests were conducted to confirm that temperature readings from the LM35 sensors could trigger the servo motors correctly. Throughout the testing process, an iterative approach was taken. This allowed for testing at each step of development and helped prevent any nested problems within the system.

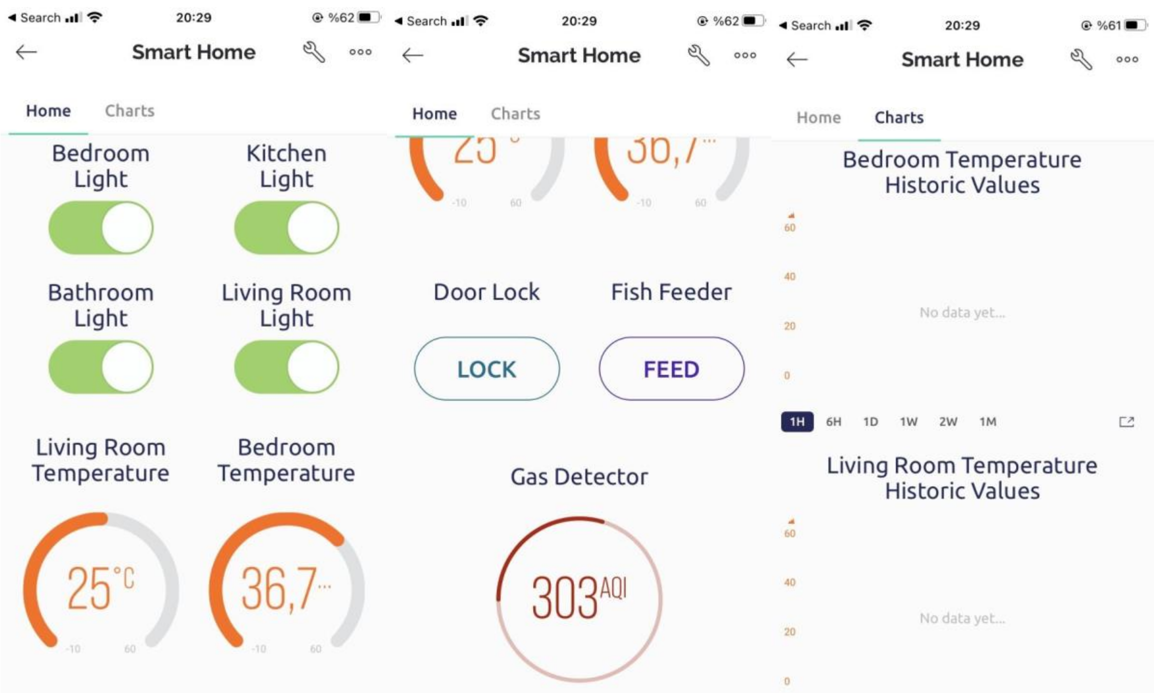


Figure 11: Final UI Design

Future Works-How Can The System Be Improved?

The system could be enhanced by connecting the temperature sensors to a fan. When the temperature exceeds a certain threshold, the fan could be activated to cool the environment. This would add an additional layer of automation to the system and improve its ability to respond to changes in temperature.

Currently, the system detects gas leaks and logs an event. In the future, an alarm could be added to provide an immediate audible or visual alert when a gas leak is detected. This would enhance the safety features of the system.

The door lock mechanism could be improved or expanded. For example, a biometric (fingerprint or facial recognition) lock could be added for increased security.

A water sensor could be installed in the bathroom to detect flooding. If the sensor detects a certain level of water, it could trigger an alert or even activate a water shut-off valve to prevent further flooding.

Implementing a feature to automatically turn off all lights after 12 PM could be a great way to save energy. This could be done using a simple timer in the system that checks the time and controls the lights accordingly.

A potential enhancement to the system could involve the use of an ultrasonic sensor placed near the door. The sensor could be programmed to detect movement. In a scenario where the door is locked, the presence of movement detected by the ultrasonic sensor could trigger an alarm. This feature could serve as an additional security measure, providing alerts for potential intruders or unexpected activity at the door.