KU LEUVEN

FACULTY OF ENGINEERING TECHNOLOGY

GROUP T LEUVEN CAMPUS

# Engineering Experience 3 - Electronics and ICT Engineering

Home 2

**Salma LOUKILI**

**Zhengchi WU**

**Enmin LIN**

**Kobe MICHIELS**

**Robbe DECAPMAKER**

Coach: Andries Martin

Technical paper submitted to obtain the degree of Bachelor of Science in Engineering Technology: Electronics and ICT Engineering

Academic Year 2023 - 2024

KU LEUVEN

Faculty of Engineering Technology Group T Leuven Campus — Technical paper submitted to obtain the degree of Bachelor of Science in Engineering Technology: Electronics and ICT Engineering
*Academic Year 2023 - 2024*

# Engineering Experience 3 - Electronics and ICT Engineering

## Home 2

Loukili Salma, Wu Zhengchi, Lin Enmin, Michiels Kobe, Decapmaker Robbe

Bachelor in Engineering Technology: Electronics and ICT Engineering, Faculty of Engineering Technology, Group T Leuven Campus, Andreas Vesaliusstraat 13, 3000 Leuven, Belgium

Coach: Andries Martin

Engineering Technology: Electronics and ICT Engineering, Faculty of Engineering Technology, Group T Leuven Campus, Andreas Vesaliusstraat 13, 3000 Leuven, Belgium, martin.andries@kuleuven.be

**ABSTRACT**

In the rapid development of modern society, pets have become important members of many families. However, pet owners often cannot continue to care for their pets at home due to work or other social commitments, which raises the question of how to ensure that pets are properly fed, entertained, and safe during the owner's absence. In view of this, this study developed an innovative smart pet feeding and entertainment system to comprehensively improve the quality of life of pets through an integrated home automation solution. The system not only automatically manages feeding, but also integrates interactive entertainment functions, which can accompany pets when their owners are not at home, effectively alleviating their loneliness. This system adopts the structure of one master node and two slave nodes. STM8 is used on the slave nodes to control the operation of each sensor function, and the ESP32 built-in Wi-Fi module is used on the master node to support users to remotely control feeding and entertainment equipment through the web interface, enhancing Improve the operability and practicality of the system. The nRF24 module is used to provide stable wireless data transmission between ESP32 and STM8, ensuring the low power consumption and communication efficiency of the system. This innovative system provides pet owners with a reliable solution that allows them to effectively manage and care for their pets even when they are away from home.

## 1 INTRODUCTION

As technology increasingly becomes a more important part of everyday life, it is only logical that it effects our homes. While there is still a certain apprehension, an increasing amount of the general public is embracing smart technology in their homes. This mounting interest is a large factor in the surge of smart home adoption [7]. Due to the technological nature of smart home devices, a lot of engineering time is needed to develop these products. This, in combination with the fact that there is a lot of public interest, leads to a situation where it is very attractive for companies and engineers to work on such problems.

This projects serves a sliver of the enormous home automation market by introducing a potential solution for some everyday worries. The problem that this paper foucses on is pet feeding during the absence of their owners. Pet feeders exist in different shapes and types, however a problem that they often have is that they only provide a solution to feeding the pet. Pets will feel lonely when there is no one around to keep them company. A basic idea to solve this problem is to provide some kind of interactive system to keep them busy while they are by themselves, this on top of taking care of the feeding task.

The aim is to tackle this problem by providing a basic mechanism by which a pet can obtain the feeling of interaction while being fed. It also provides more fine-grained control over how much and when the pet eats. The implementation of these home automation ideas is at the core of the research performed for the completion of this EE3 project.

### 1.1 Requirements analysis

The implementation of the ideas outlined in chapter 1 have to meet some basic requirements. Firstly, since they are consumer products, they need to be easy to setup and have next to no barriers to entry. Because of this we want every separate function to be easy to place around the house without adding extra clutter such as communication cables. This gives rise to a few key design features which are needed to create a good product. We need a central entity which provides a proper user interface in order to interact with all functionality of the system. Furthermore, every function needs to be able to communicate wirelessly.

Secondly, we don't want the two discussed subsystems to be to close to each other, meaning that we'll need to be able to split them into two separate modules. This way, they can be in a different physical location whilst not compromising on functionality or comfort of the pet. In order to keep this feasible and cost effective, we require modular design.

## 2 DESIGN AND MATERIALS

The system of this EE3 project contains three small subsystems. There is a master node that acts as a controller and two basic slave nodes. The basic nodes are responsible for interaction and monitoring the feeding mechanism. Figure 2.1 shows the top level block diagram of the system.

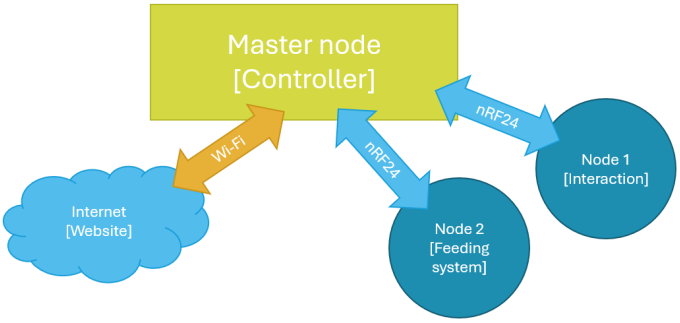


**Figure 2.1:** Top level block diagram

Figure 2.2 shows the block diagram of the master node. The main component of this node is the ESP32-S3-DevKitC-1-N8R8 [16]. This microcontroller is able to set up its own Wi-Fi network that can be used to implement a webserver for user interaction. It uses an nRF24L01 [2] to communicate wirelessly with the basic nodes.

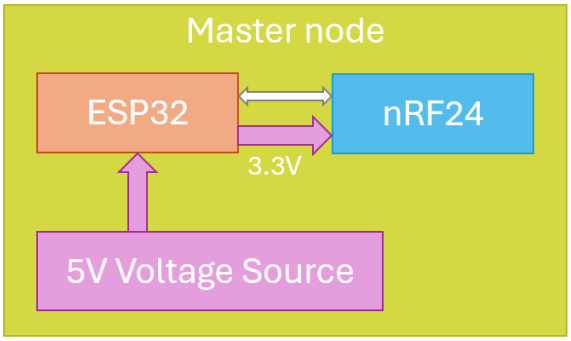


**Figure 2.2:** Block diagram of master node

Figure 2.3 shows the block diagram of the first basic node. This subsystem is controlled by an STM8S105K4 microcontroller [14] and is responsible for the interaction with the pet. It uses the PIR Motion Sensor HC-SR501 [8] to detect movement. The advantage of this sensor is that it can be used to generate an interrupt in case of motion detection. A second way of interaction is possible through the combination of the AZ-Delivery MP3 DFPlayer Mini Module and a small audio speaker [6].

Figure 2.4 shows the block diagram of the second basic node. The main controller on this node is also an STM8S105K4, which is connected to a servo motor [3] and a weight sensor [13]. This setup was chosen to per-

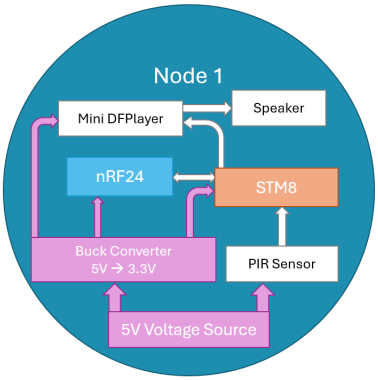


**Figure 2.3:** Block diagram of node 1

form the function of dispensing and monitoring the food portions of the pet. We use the weight sensor to determine the amount of food the pet has consumed, and can refill it's supply by rotating a valve with the servo.

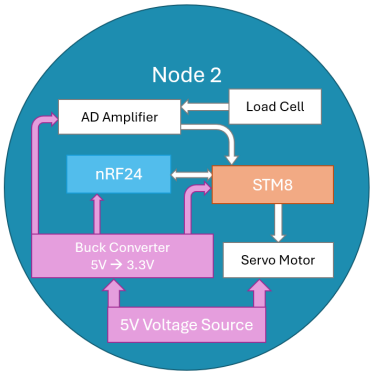


**Figure 2.4:** Block diagram of node 2

## 3 IMPLEMENTATION

### 3.1 Hardware implementation

#### 3.1.1 Master node

The master node PCB is simple from a hardware perspective. Since all sensors and actuators are connected to the interaction and feeding node, discussed in section 3.1.2. There is not much left to connect to the ESP32 on the master node. The only components on the board are thus the ESP32and an nRF24 to enable wireless communication with the slave nodes. Figure 3.1 shows the layout of the PCB itself. Power is supplied by a micro-USB connector on the ESP32 module, which instantly converts it into a 3V3 power source. This is in turn used to power the nRF24 module, with the aid of some decoupling capacitors to minimize the impact of noise and sudden current bursts. The two are connected via an SPI bus which does not need any special routing considerations due to the relatively low speed at which this bus runs. A ground plane was added to ensure that every trace has a proper and consistent reference.

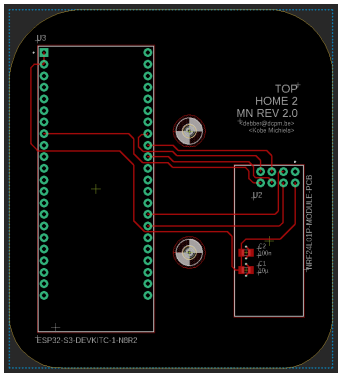


**Figure 3.1:** PCB layout for the master node

#### 3.1.2 Interaction and feeding node

Modular design

The hardware design for both the interaction and feeding node is the same, even though they have different capabilities. To achieve this, a modular PCB was designed with flexibility in mind. This design strategy decreases manufacturing cost as it enables us to have fewer designs which need to be made. Depending on which functionality is needed, components can be left off to drive down cost even further.

General overview

The PCB layout of the STM8 nodes can be seen in figure 3.2. The board features support for a high efficiency buck converter to convert the 5V USB input to 3V3 for the STM8 and several sensors and actuators. The following is a list of all supported devices:

- nRF24L01 [2]
- HX711 and load cell (weight sensor) [13]
- Servo motor MG996R [3]
- HC-SR501 PIR (motion sensor) [8]
- MP3 player + speaker [6] [5]
- Status LED
- RGB LEDs

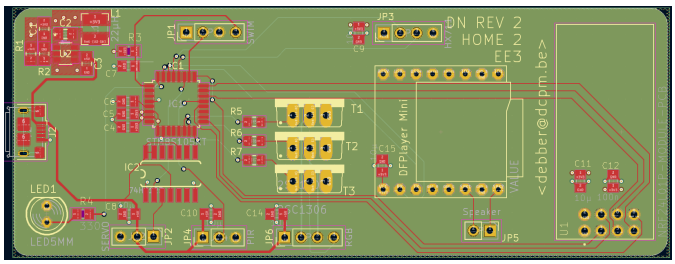


**Figure 3.2:** PCB layout for the STM8 nodes

Power delivery

The main power source for the board is a micro USB connector, which was chosen for its ease of use and relative safety. It ensures that we only get 5V under normal operation with any regular wall adapter. It also provides the built in electronic fusing provided by that same adapter [21].

The 3V3 rail is created by a LMR51610XDBVR. It is a versatile switching converter which provides great efficiency when stepping down our 5V rail. Using the reference design provided by TI, the schematic in figure 3.3 was created for implementing the IC [10].

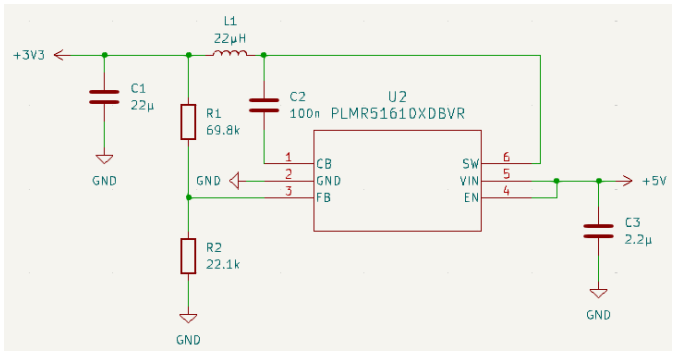


**Figure 3.3:** Shematic for LMR51610XDBVR

Power integrity

In order to ensure stability of the power rails, both 5V and 3V3, decoupling capacitors were added. The addition of these capacitors helps supply enough current to the components when there is a sudden current draw. This also provides the added benefit that the voltage rails don't show a temporary voltage drop when the current spike takes place. The second reason for adding (smaller) decoupling capacitors, is to reduce the effects of noise produced by components on the power rails.

The board also features a four layer stack-up where the two inside layers are a 3V3 and GND power plane. This means that the two are acting like a capacitor, further improving resilience against sudden current spikes.

Signal integrity

Since there are no high speed signals present on the board, there is no need to take special care of signal traces. However, there is a reference plane for each trace, and proper spacing was provided to minimize the effects of cross-talk.

## 3.2 Software implementation

Communication between the nodes goes through the nRF24. Each message has a specific structure, to which will be referred to as a datagram during the rest of this paper. This datagram consists of 5 bytes: a device byte, a command byte, 2 data bytes and a status byte.

Radio communication is set on channel 97 with a data rate of 1Mbps. Addresses are meticulously configured: 'ABCDE' for the interaction slave node, 'FGHIJ' for the master node, and 'KLMNO' for the feeding slave node, ensuring clear and effective communication channels within the system.

**Table 3.1:** Datagram commands

| Device | Command | Used data-byte | Action |
|---|---|---|---|
| 0x01 | 0x00 | | Turn LED off |
| | 0x01 | | Turn LED on |
| | 0x02 | | Enable auto-lights |
| | 0x03 | | Disable auto-lights |
| 0x02 | 0x01 | | Motor On |
| | 0x02 | | Motor Off |
| | 0x03 | | Enable auto-dispense |
| | 0x04 | | Disable auto-dispense |
| 0x03 | 0x01 | Data byte 0 | Play sound |
| | 0x02 | | Increment volume |
| | 0x03 | | Decrement volume |
| | 0x04 | Data byte 0 | Set volume |
| | 0x05 | | Enable auto-play |
| | 0x06 | | Disable auto-play |
| 0x04 | 0x01 | | Turn threshold up |
| | 0x02 | | Turn threshold down |
| | 0x03 | Data byte 0 | Send weight reading |
| 0x05 | 0x01 | | Motion detected |

### 3.2.1 Master node

User interface

The master node forms the core of the intelligent pet feeder system, serving as the main interface for user interaction and communication with the interaction and feeding

nodes. Using the powerful and versatile ESP32 module, this node acts as a robust web server while managing the data communication processes through the nRF24 module.

The ESP32 is integral to hosting the webserver [18], executed within the ESP-IDF framework [15], a well-documented development environment provided by Espressif Systems. This web server is responsible for a web application that encompasses system monitoring, control, and automation functionalities. The application handles various HTTP requests and renders web pages stored in binary format into browser-readable formats. It acts as a user interface for interacting with the pet feeder system. The front-end of the application is developed using HTML, CSS, and JavaScript, while the back-end handles the server logic, processes network requests, and manages the communication via the nRF24. The integration between the front-end and back-end is handled through AJAX calls, which facilitate the dynamic loading of content based on data received from the backend without needing to reload the web page.

The captive portal feature of the ESP32 HTTP web server is pivotal for initial device setup and network access control. Its configuration involves setting the ESP32 to operate as an Access Point (AP), creating its own Wi-Fi network [20] that users can join. Upon connection, all web traffic is redirected to the captive portal until the user completes the necessary authentication or configuration steps. This ensures that only dedicated owners of the machine can access the web application, and control and monitor the system.

Combining the webserver along with the nRF24 communication was facilitated by the use of FreeRTOS [17] of the ESP32. It ensured smooth and fast execution of both tasks simultaneously without the need to use interrupts. This real-time operating system included in the ESP32 orchestrates the operations of the webserver and the nRF24 communication. By prioritizing and scheduling these tasks, FreeRTOS enables the ESP32 to handle user interactions through the web interface while maintaining continuous data exchange with the devices of the slave nodes.

and the nRF24. The SPI protocol setup involves defining the clock polarity and phase to match the requirements of the nRF24, configuring the master node (ESP32) for full-duplex communication. Key SPI parameters include the setting of the SPI clock frequency to ensure stable communication and the configuration of the master's chip select (CS) line to correctly signal the start and end of an SPI communication session. [19]

The MiRF library [12] was used to facilitate the integration of the nRF24 into the main node. It acts as an abstraction layer on top of the nRF24's SPI communication. The library manages the configuration of radio frequency, the organization of transmission queues, and maintains the integrity of data during communication processes.

In the implementation, specific GPIO pins are assigned for effective SPI communication, following the hardware connection between the ESP32 and the nRF24 in the PCB: MISO is connected to GPIO 40, MOSI to GPIO 38, SCLK to GPIO 39, CE to GPIO 45, and CSN to GPIO 48. The system uses SPI2_HOST for managing the SPI bus, ensuring reliable data handling.

As explained in previous section 3.2, each slave node is assigned a unique address, and the master node switches its transmission address dynamically to communicate with each slave individually. This setup allows targeted commands to the appropriate device, such as dispensing food or playing audio, based on user inputs and programmed automation schedules as explained in 3.4.

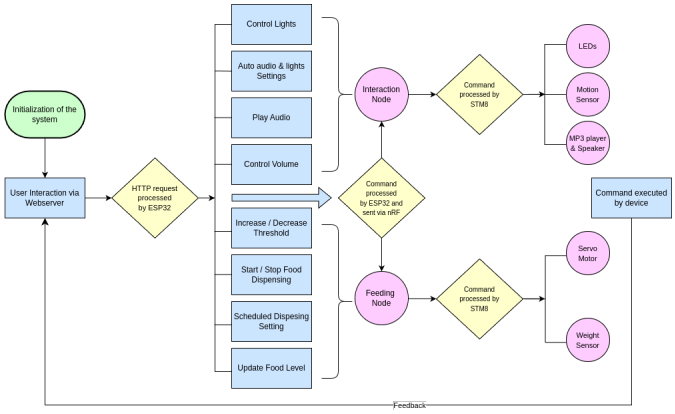


**Figure 3.4:** Master node data flow

Master-slave wireless communication

As discussed in previous chapters, wireless data transmission within the system is facilitated by the nRF24 module, a highly efficient wireless communication device that ensures reliable data transfer between the master node and the slave nodes. This implementation configures the SPI protocol to establish communication between the ESP32

The datagrams 3.1, structured data packets used within this system, consist of device identifiers, command codes, and necessary parameters. These packets are essential for executing specific actions like dispensing food, playing audio, or updating system settings based on both user inputs and automated schedules.

### 3.2.2 Feeding node

The implementation starts with a series of configuration steps including GPIO initialization, timer setup, nRF24 communication setup, clock system, and interrupt configuration. The stm8nrf24.c file is specifically dedicated to configuring the connection between the STM8 microcontroller and the nRF24 wireless communication module. This includes SPI initialization, register read/write operations, address read/write functions, work mode settings, and interrupt setups. Also, we set up the TX address of nRF24 to '0x46 0x47 0x48 0x49 0x4A', and the RX address is set to '0x41 0x42 0x43 0x44 0x45'.

The weight sensor is initialized to pin 28 and 29 of STM8, while the motor is initialized to pin 27 of STM8. Figure 3.5 shows that the interrupt for controlling the motor uses timer 3 (TIM3), the timer drives the PWM required for the motor from the PWM channel of STM8. The "PORT C interrupt" is triggered by hardware signal changes, used for capturing data datagrams transmitted through the nRF24 communication module in real-time.

The reading of the weight sensor is processed by the STM8. First, the STM8 waits for the DT line to go low, indicating that data is ready. When that's the case, it reads 24 bits of data and 1 bit for gain setting. During the reading, the STM8 first generates a rising edge, then shifts the data to the left by one bit, making space for the next bit of data. A falling edge is then generated to prepare for reading the next data bit. This loop continues until reading the entire datagram is done.

In order to enable the automatic dispensing of food by the feeder machine, it is necessary to compare the data obtained from the weight sensor with a threshold value during each interrupt. If the data read is greater than the threshold value, it indicates that the food tray is empty. Subsequently, the STM8 will output different PWM signals to drive the servo motor for rotation. The PWM signal will remain unchanged once the food on the tray reaches the threshold weight, there by preventing further rotation of the servo motor.

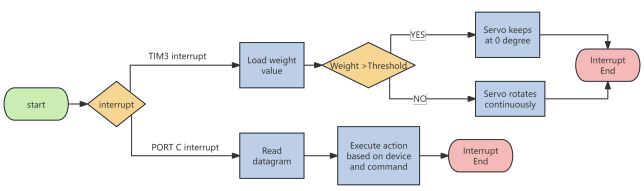


**Figure 3.5:** Feeding node interrupt flowchart

### 3.2.3 Interaction node

The code for the interaction node starts with setting up the communication bus for the nRF24. The nRF communication is handled by a self-written library based on the preliminary product specification of the nRF24L01 [2]. The RF channel frequency is set to 0x61 to obtain a frequency of 2.497GHz. The TX address is set to '0x46 0x47 0x48 0x49 0x4A'. The RX address is set to '0x65 0x66 0x67 0x68 0x69'.

After the nRF is setup, pin C4 of the STM8 is initialized as a floating input with external interrupt. The motion sensor interrupts on this port when it detects motion.

Next, the DFPlayer MP3 module is initialized. This is done by sending a command over UART to the MP3 module. Figure 3.6 shows the format of the command. The MP3 module has a collection of command types that are executed with a self-written library [6].

**$SB VB LB CMD ACK DATA1 DATA2 CHKS1 CHKS2 $EB**

| Mark | Byte | Byte description |
|---|---|---|
| $SB | *0x7E* | Start byte |
| VB | *0xFF* | Version byte |
| LB | *0xxx* | The number of bytes of the command without start and end bytes (In our case *0x06*) |
| CMD | *0xxx* | Such as *PLAY* and *PAUSE* and so on |
| ACK | *0xxx* | Acknowledge byte *0x00* = not ack, *0x01* = ack |
| DATA1 | *0xxx* | Data high byte |
| DATA2 | *0xxx* | Data low byte |
| CHKS1 | *0xxx* | Checksum high byte |
| CHKS2 | *0xxx* | Checksum low byte |
| $EB | *0xEF* | End byte |

**Figure 3.6:** DFPlayer command format

The last step of the interaction node initialization consists of setting pins B0, B1 and B2 as floating inputs with a LOW value. These pins will drive the LED.

After this initialization process, the node entirely runs via interrupts on port C. Figure 3.7 shows the working principle of this interrupt routine. If there is an interrupt on pin 4, motion is detected with the sensor and a datagram with this information is send to the master node. If autoplay is turned on, a sound will be played. If there is an interrupt on pin 3, the STM8 received a message from the ESP. This datagram is read and the corresponding action is executed, based on the format from table 3.1.

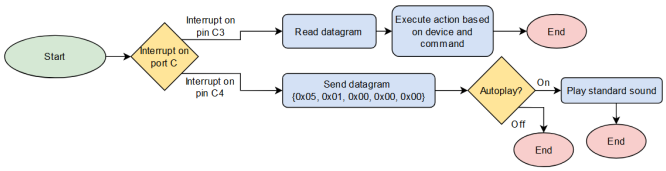


**Figure 3.7:** Interaction node interrupt flowchart

### 3.3 Prototype building

The prototype was constructed predominantly using elementary materials such as metal, wood and 3D-printed components. The dispensing mechanism, positioned at the base of the storage compartment, was engineered using the 3D modeling software Solid Edge to meet specific design criteria. The enclosure's walls and lid, crafted from 3 mm laser-cut MDF wood featuring finger joints, were designed using the graphic design software Inkscape. The structure is supported by metal legs sourced from Brico.

A critical feature of the dispensing mechanism is the blade, which is affixed to a servo motor via a threaded rod, facilitating the dispensing of food upon the motor's activation. Insertion nuts were incorporated to facilitate adjustments to the threaded rod's alignment within the blade.

Below an overview of the materials used for building the prototype (excluding electronic components):

- Four 3mm thick MDF wood was used for the walls and lid.
- Around 150 grams of 3D printing ink for the dispensing mechanism.
- Four metal legs and 16 screws. [4]
- Two M3 threaded insertion nuts. [1]
- 20 cm long M3 threaded rod. [11]
- Wood glue and white acrylic paint.

## 4 EVALUATION AND VALIDATION

### 4.1 Prototype

#### 4.1.1 Physical prototype

As shown in Figure 4.1, the feeder was designed as a fully functional prototype, capable of storing up to 5 kg of food, with a design that emphasizes simplicity and durability. The dimensions of the storage box are 25 cm by 25 cm by 25 cm. The addition of legs provides an extra 30 cm in height, bringing the total dimensions to 25 cm by 25 cm by 55 cm. The lid sits easily on the box, allowing coverage of the pet food and protecting it from insects and humidity.

The design incorporates a lid and a transparent section that allows users to monitor the food level within the storage at all times. The dispenser mechanism at the base includes a blade that facilitates rapid and precise dispensing of food. The plate sits on top of the weight sensor and can take up to 350 g of pet food.

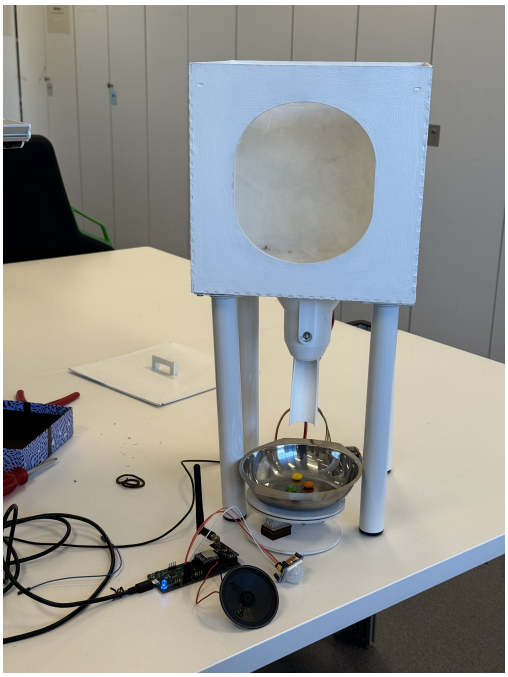

**Figure 4.1:** Final Prototype

#### 4.1.2 Web Application Interface

The web application of the Smart Pet Feeder system serves as the primary interface for user interaction, allowing for real-time monitoring and control over the feeding and interaction mechanisms. Figures 4.2, 4.3, 4.4 illustrate the various functionalities available through the web interface.

In the figures, the different features of the web application are displayed:

- **Home Page (Figure 4.2)**: Displays real-time updates of the system's status, including how much food is on the plate, and the state of the lights. The home page also provides direct access to primary actions such as dispensing food, playing audio prompts and changing settings like volume and threshold.

  The threshold represents how much the plate can be filled when empty. The threshold is by default set to 100%, meaning that the plate will be filled to its maximum capacity whenever it's empty or whenever the user commands to start dispensing.

  For playing audio, the interface allows the user to select a specific audio to play, this is done via drop-

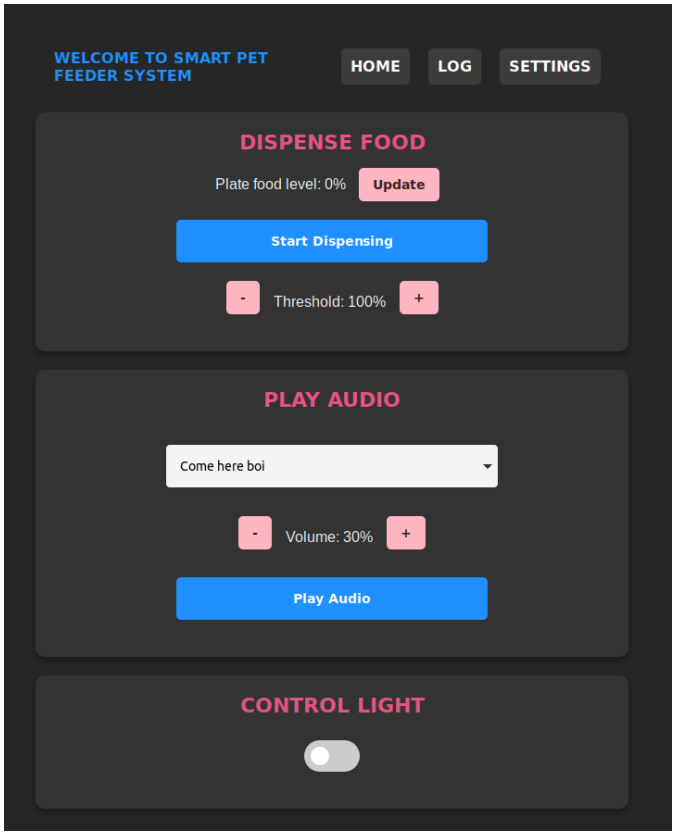


**Figure 4.2:** Home page

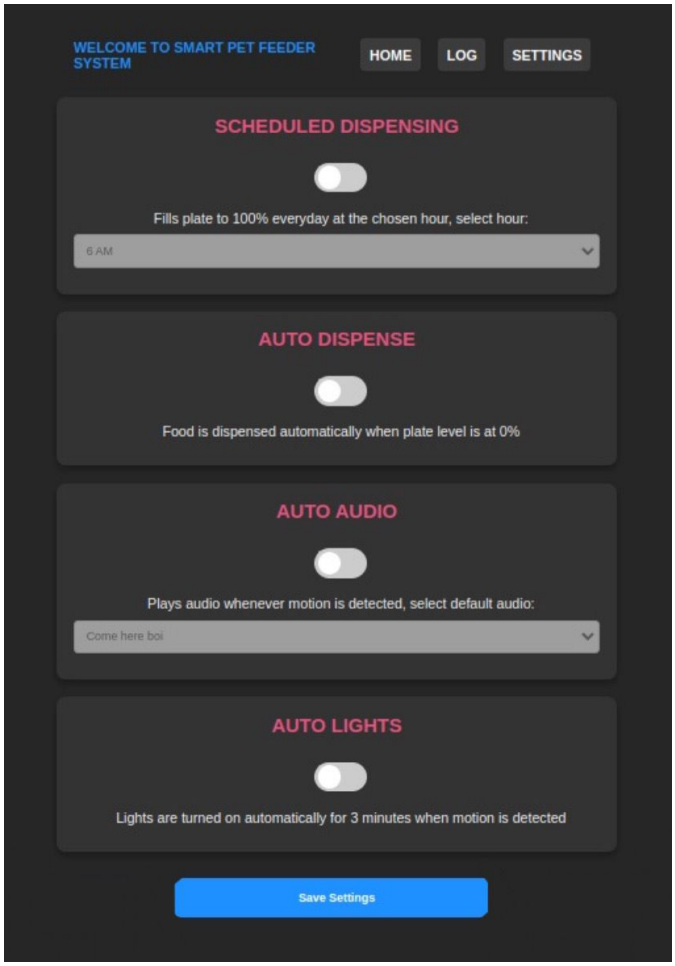


**Figure 4.3:** Settings page

down selection menu. The user can also increase or decrease the volume by levels of 10% and toggle the lights.

- **Settings Page (Figure 4.3)**: Allows users to set up and modify feeding schedules, and toggle the auto-lights, auto-audio and auto-dispense settings. When these are toggled on and saved, the system will automatically play an audio and turn on the lights whenever the pet is close to the machine. Auto-dispense means the plate will fill automatically whenever it's fully empty. The state of these settings is saved internally when the user clicks on the save button.
- **Logs Page (Figure 4.4)**: The system is configured to meticulously record and display detailed logs of all operational activities. These logs capture each instance of motion detection, audio playback, food dispensing, and updates to the plate level. Each entry in the log is marked with a precise date and timestamp to ensure accurate tracking and historical data retrieval.

## 4.2 Performance Testing

The hardware components underwent rigorous testing to assess power consumption, response times, and mechanical efficiency, verifying their functionality within the specified parameters.

### 4.2.1 Power Consumption

Power consumption was evaluated using a multi-meter, which involved direct measurement of current and voltage across each component. This approach ensured precise quantification of the energy requirements of the system. Table 4.1 displays the results per components. Assuming the device runs all the time, we can estimate the power consumption of the machine to be around 53 Watt-hours.

**Table 4.1:** Power Consumption

| Component | Power consumption (mWatts) |
|---|---|
| ESP32 | 673 |
| STM8 | 440 |
| nRF24 | 440 |
| Servo Motor | 160 |
| Load Cell HX711 | 137 |
| Motion Sensor PIR | 300 |
| DFPlayer | 60.0 |

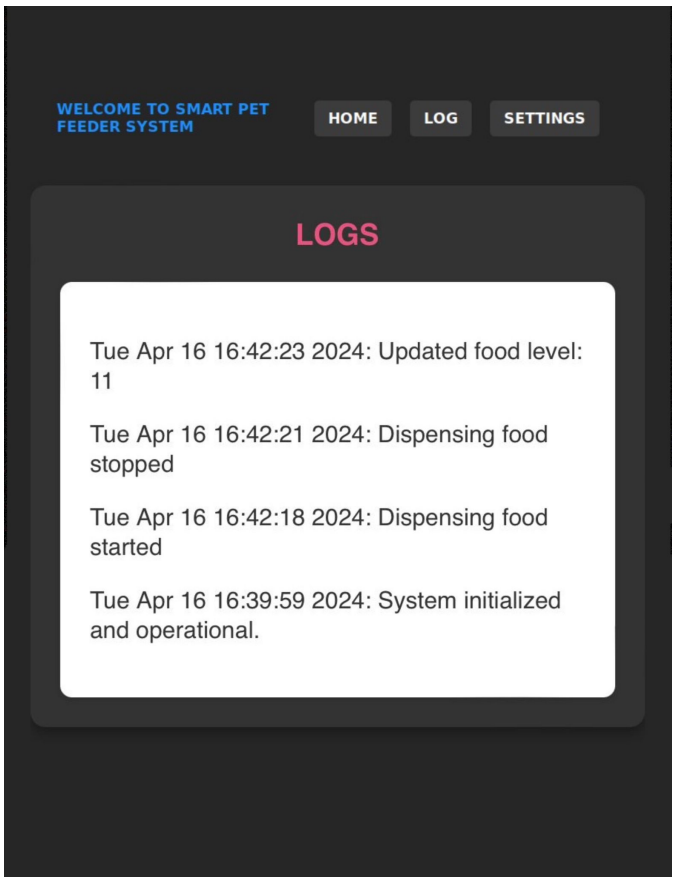


**Figure 4.4:** Logs page

### 4.2.2 Throughput and Efficiency

Using Apache JMeter, the throughput of the ESP32 was evaluated. This metric reflects the number of requests the server is capable of processing. The server was subjected to repeated testing periods involving multiple clients simultaneously. After several iterations, the average number of requests successfully handled was determined. The throughput was found to be 11.9 requests per second.

For the nRF24, a code snippet incorporating timing was utilized to continuously transmit data between the modules. Averaged over numerous trials, the nRF24 achieved a throughput of 61.5 bytes per second. This testing also assessed the efficiency of the nRF modules, with a successful transmission rate of up to 90

### 4.2.3 Input delays and reaction times

Additionally, the ESP32 webserver's response times were tested to gauge the input delay caused by HTTP request handling and rendering. Using the Requests Python library [9], the test code evaluated the response time for multiple HTTP requests to various endpoints on the webserver, averaging the results over many tests. This response time was recorded at 54 milliseconds.

For the nRF24, the round-trip time (RTT) was measured, which is the duration required for the module to send a message and receive an acknowledgment in return. Averaged over multiple tests, the RTT for the nRF24 module was measured to be 7.48 milliseconds.

Finally, the integrated system was tested as a whole. The reaction times for each functionality were timed from the user input to the physical execution of the corresponding command. Table 4.2 outlines the various reaction times, including the dispense time, which is the duration needed to fill the plate to 100%, and the connection time, which includes the time required to establish a connection with the webserver and render the homepage, factoring in the authentication time for server verification of network credentials.

**Table 4.2:** Reaction times

| Functionality | Metric | Result |
|---|---|---|
| Dispense food | Reaction time | 2.70 s |
| Toggle lights | Reaction time | 0.950 s |
| Update weight | Reaction time | 2.38 s |
| Play audio | Reaction time | 1.10 s |
| Servo motor | Dispense time | 192 s |
| Webserver | Connection time | 8.52 s |

## 4.3 Development Cost

The development costs for the smart pet feeder prototype are outlined in the Bill of Materials in Table 4.3, which itemizes each component required for the development of this system. The prototype consists of various parts categorized under PCBs, components, and prototype-specific materials. The PCBs include 10 units priced at EUR 0.8 each, totaling EUR 8. Component costs include a servo motor (EUR 12), an MP3 player (EUR 2.5), and an LED among others, with notable expenses like the STM8 microcontroller costing EUR 2 per unit. Prototype materials like the 3D printed blade and dispenser setup are priced at EUR 18. Furthermore, structural components such as walls and a lid made from lasercutting technology cost EUR 6 for 4 units. In total, the sum of all individual components and materials comes to EUR 106.24, providing a financial overview of what is needed to build one unit of the smart pet feeder prototype.

**Table 4.3:** Bill of Materials

| Part | Quantity | Cost (EUR) |
|---|---|---|
| PCBs | 10 | 8.0 |
| PCB Case | 2 | 2.6 |
| Servo | 1 | 12.0 |
| MP3 Player | 1 | 2.5 |
| Motion Sensor | 1 | 2.5 |
| Weight Sensor | 1 | 5.0 |
| Speaker | 1 | 3.0 |
| LEDs | 1 | 0.4 |
| Switching Regulator | 2 | 3.0 |
| Voltage Regulator | 1 | 1.0 |
| STM8S105K4 | 1 | 2.0 |
| ESP32S3 | 2 | 3.0 |
| nRF24L01 | 3 | 7.5 |
| Lasercutting | 4 | 6.0 |
| 3D printing | 1 | 18.0 |
| Legs | 4 | 21.2 |
| Insertion Nuts | 2 | 0.15 |
| Threaded Rod | 1 | 1.59 |

## 5 DISCUSSION

### 5.1 Physical Prototype Design

The Smart Pet Feeder prototype is engineered with a focus on simplicity and robustness. The design features a compact 25 cm cube storage box mounted on 30 cm legs, providing a functional yet straightforward structure that enhances user accessibility and ease of operation. The incorporation of a transparent section is a strategic design choice, allowing users to easily monitor the food level without needing to open the container. This feature not only facilitates user convenience but also helps in maintaining the food's protection against environmental factors such as humidity and insects. Such a characteristic is well-aligned with consumer preferences noted in competitive products, where easy monitoring capabilities are highly valued.

Despite these advantages, the feeder's minimalistic design, driven by cost and time constraints, resulted in a somewhat bulky form factor. The square edges of the box could be re-designed into rounded ones to aid in smoother dispensing. Moreover, while the feeder can store up to 5 kg of food, there are concerns about its efficiency in maintaining food freshness over long period of time. This is particularly critical in homes with elevated humidity or prevalent insects, where the effectiveness of the lid's seal against such conditions could be further evaluated and potentially enhanced in future design iterations.

### 5.2 Web Application Interface and System Operations

The web application interface acts as the central hub for controlling and monitoring the Smart Pet Feeder. Its straightforward design ensures a user-friendly experience that effectively balances functionality without overwhelming the user. For demonstration purposes, the ESP32 hosts the webserver as a captive portal, which simplifies testing and demonstration phases. Although this setup suits prototype testing, a fully deployed system would require the webserver to be hosted externally to allow remote access from any location. This adaptation would ensure that the computational demands on the ESP32, which are manageable in its current captive portal form, are appropriately scaled for broader application.

### 5.3 Power Consumption and Energy Efficiency

The recorded power usage of 53 Watt-hours is highly comparable to pet feeders on the market, While our system provides an extra interaction feature. This reflects a design that is optimized for energy efficiency. This aspect is particularly significant considering the increasing user concern over electrical costs and environmental impacts.

### 5.4 Network Performance and Response Times

The throughput rates of 11.9 requests per second for the ESP32 and 61.5 bytes per second for the nRF24, along with a 90% successful transmission rate, demonstrate robust performance suitable for typical home IoT environments. Nonetheless, the potential impact of high network traffic or adverse network conditions on these metrics should be considered. Network reliability and response times are often critical factors affecting user satisfaction in smart home devices, as noted in the relevant literature.

The system exhibits impressive input delays and reaction times, with functionalities such as toggling lights and playing audio executing within approximately one second. However, the servo motor's food dispensing time of 192 seconds, and the overall system latency in responding to dispensing and update commands, which can extend up to three seconds, may represent significant performance bottlenecks. These delays could potentially lead to user dissatisfaction, especially when rapid feeding is required, suggesting an area for substantial improvement in future development phases.

## 6 CONCLUSION

Although the developed system cam out as a fully functioning entity with good performance metrics, particularly given the constraints of time, cost, and the specific requirements of the project. The system does not provide the stability or adequacy required for long-term functionality as a commercial product. The design, while sufficient for demonstrating core capabilities and technological feasibilities, remains fundamentally a prototype, shown as Figure 4.1. This outcome aligns precisely with the research's objectives, affirming that the prototype serves as an exploratory step in understanding and refining smart feeder technology. Furthermore, the design conforms completely to the specifications and goals outlined in the initial EE3 project description, thus successfully meeting its intended academic and developmental milestones.

## 7 ACKNOWLEDGEMENTS

We would like to extend our thanks and appreciations to our coaches Mr. Andries Martin and Mr. Vargemidis Dimitri for providing excellent guidance and help throughout this project.

Thank you to Jonathan Valgaeren and Pieter nagels for making the sound effects used in the interaction node.

### Acronyms

| | |
|---|---|
| ADC | Analog to Digital Converter |
| ST | STMicroelectronics |
| PIR | Passive infrared |
| MP3 | MPEG-1 Audio Layer III |
| SPI | Serial Peripheral Interface |
| IC | Integrated circuit |
| TI | Texas Instruments |
| HTTP | Hypertext Transfer Protocol |
| GPIO | General Purpose Input/Output |
| MISO | Master In Slave Out |
| MOSI | Master Out Slave In |
| CSN | Chip Select Not |
| CE | Chip Enable |
| SCLK | Serial Clock |
| AJAX | Asynchronous JavaScript and XML |
| CSS | Cascading Style Sheets |
| PWM | Pulse Width Modulation |
| IoT | Internet of Things |

## BIBLIOGRAPHY

[1] Amazon. *M3 Threaded Inserts*. https://www.amazon.com.be/-/en/Threaded-Inserts-Knurled-Plastic-Pressure/dp/B0BX9PD6MM/ref=asc_df_B0BX9PD6MM/?tag=begogshpadde-21&linkCode=df0&hvadid=633051347177&hvpos=&hvnetw=g&hvrand=6709136618414013455&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=9040019&hvtargid=pla-2090168518778&psc=1&mcid=1d35c2a95cb736778c0e430892af5c03. [Accessed 12-05-2024].

[2] Nordic Semiconductor ASA. *PRELIMINARY PRODUCT SPECIFICATION, Single chip 2.4 GHz Transceiver, nRF24L01*. https://www.sparkfun.com/datasheets/Components/nRF24L01_prelim_prod_spec_1_2.pdf. [Accessed 07-05-2024]. 2006.

[3] AZDelivery. *MG996R Datasheet*. https://www.electronicoscaldas.com/datasheet/MG996R_Tower-Pro.pdf. [Accessed 12-05-2024].

[4] Brico. *Duraline meubelpoot rond staal 3x30cm wit*. https://www.brico.be/nl/badkamer-keuken-wonen/meubelen/meubelpoten/duraline-meubelpoot-rond-staal-3x30cm-wit/6535982. [Accessed 12-05-2024].

[5] Cuidevices. *GF0771 Datasheet*. https://www.cuidevices.com/product/resource/gf0771.pdf. [Accessed 12-05-2024].

[6] AZ-Delivery. *AZ-Delivery MP3 DFPlayer Mini Module*. https://www.az-delivery.de/en/products/mp3-player-modul. [Accessed 07-05-2024].

[7] Sara Gøthesen, M. Haddara, and Karippur Nanda Kumar. "Empowering homes with intelligence: An investigation of smart home technology adoption and usage". In: *Internet Things* 24 (2023), p. 100944. URL: https://api.semanticscholar.org/CorpusID:263321916.

[8] *HC-SR501 PIR MOTION DETECTOR, Product description*. https://www.mpja.com/download/31227sc.pdf. [Accessed 07-05-2024].

[9] Nateprewitt Ian Stapleton Cordasco Lukasa. *Requests — pypi.org*. https://pypi.org/project/requests/. [Accessed 09-05-2024]. 2023.

[10] Texas Instruments. *LMR516xx SIMPLE SWITCHER Power Converter, 4-V to 65-V, 0.6-A/1-A Buck Converter in a SOT-23 Package*. https://www.ti.com/lit/ds/symlink/

lmr51610.pdf?ts=1714548314287&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FLMR51610. [Accessed 01-05-2024]. 2023.

[11] MetaLeuven. *M3 1 meter Threaded Rod*. https://www.schroeven-express.nl/draadstang-lengte-1-meter-rvs-a2/16351-300748-draadstang-m3-lengte-1-meter-rvs-a2-3663072172791.html?utm_campaign=googleads&utm_source=shopping&utm_medium=Nederland&gad_source=1&gclid=CjwKCAjw0YGyBhByEiwAQmBEWm9wL3i2Xu-f8XOgfGZ7lOFx8fdWWlq13s_u1getRcYG6jKB3lHhShoC73oQAvD_BwE. [Accessed 12-05-2024].

[12] nopnop2002. *esp-idf-mirf*. https://github.com/nopnop2002/esp-idf-mirf. [Accessed 09-05-2024]. 2023.

[13] AVIA Semiconductor. *HX711 Datasheet*. https://www.digikey.com/htmldatasheets/production/1836471/0/0/1/hx711.html. [Accessed 12-05-2024].

[14] STMicroelectronics. *STM8S105K4 Datasheet*. https://www.st.com/en/microcontrollers-microprocessors/stm8s105k4.html. [Accessed 12-05-2024].

[15] Espressif Systems. *ESP-IDF Programming Guide*. https://docs.espressif.com/projects/esp-idf/en/stable/esp32s3/index.html. [Accessed 09-05-2024]. 2024.

[16] Espressif Systems. *ESP32-S3-DevKitC-1 v1.1*. https://docs.espressif.com/projects/esp-idf/en/stable/esp32s3/hw-reference/esp32s3/user-guide-devkitc-1.html. [Accessed 09-05-2024]. 2024.

[17] Espressif Systems. *FreeRTOS*. https://docs.espressif.com/projects/esp-idf/en/stable/esp32s3/api-reference/system/freertos_idf.html. [Accessed 09-05-2024]. 2024.

[18] Espressif Systems. *HTTP Server*. https://docs.espressif.com/projects/esp-idf/en/stable/esp32s3/api-reference/protocols/esp_http_server.html. [Accessed 09-05-2024]. 2024.

[19] Espressif Systems. *SPI Master Drive*. https://docs.espressif.com/projects/esp-idf/en/stable/esp32s3/api-reference/peripherals/spi_master.html. [Accessed 09-05-2024]. 2024.

[20] Espressif Systems. *WiFi Easy Connect*. https://docs.espressif.com/projects/esp-idf/en/stable/esp32s3/api-reference/network/esp_dpp.html. [Accessed 09-05-2024]. 2024.

[21] *USB 2.0 Specification — USB-IF — usb.org*. https://www.usb.org/document-library/usb-20-specification. [Accessed 01-05-2024].

# APPENDICES