National Institut of Applied Science and Technology

CARTHAGE UNIVERSITY

End of the Year Project

Specialty: Instrumentation and Industrial Maintenance

Design and Production of a Cows Health Monitoring Platform

Presented by:

HAJJOUJI Omar

REJAIBI Kais

OUESLATI Med Zied

HAMMEMI Houssem

Academic Supervisor: Mrs. HARBAOUI Imen

Reviewer: Mrs. BOHLI Nedra

Academic Year: 2021/2022

Contents

List of Figures

Introduction

The dairy industry has undergone massive changes in the last decades; the number of family-owned small-scale farms is decreasing leaving place for large-sized industrialized farms as a response to a constant increase in demand.

The advancements in the fields of IOT, Sensor Technology and Data Acquisition are offering new tools and innovations to increase efficiency and profitability for the agriculture industry.

The implementation of these technologies allow for real time monitoring of various health factors for each animal of the herd, allowing for an easy and accurate identification of the animals' needs.

The problem that this project tackles is the estrus detection in milk cows, which has always been a major problem for farmers as it heavily affects production. This report will present the different factors that allow to reliably predict an estrus, then present the built prototype for detection of said factors and the mobile application for real-time monitoring.

This project comes as a response for the increasing need for a low-cost estrus detection solution in the Tunisian market, and this report that is composed of 3 main chapters will present the solution.

The 1st Chapter: Introduces some general context about cow's estrous cycle and the estrus. Then the chapter presents the reasons for the interest of the dairy industry in estrus detection and present a state of the art present marketed solutions.

The 2nd Chapter: Presents the proposed solution, explains the solution's philosophy and details all used software and hardware resources.

The 3rd Chapter: Shows the implementation of the solution, details the wiring of the different components and shows portions of the code that was written to ensure proper communication between the components.

1

Project Scope

Introduction

This chapter will show the general context in which this project takes place.
It will present the different parameters we have to understand in order to tackle the dairy industry.

1 Estrous Cycle in Cows

[1] This cyclic process is called the estrous cycle and consists of a definite sequence of events, both physiological and behavioral.
The estrous cycle of the cow starts after puberty and occurs approximately every 21 days (17 to 24 days) for healthy cows.
During estrous cycle, the reproductive tract is prepared for **Estrus** (commonly referred to as Heat) which is the period of sexual receptivity and **Ovulation**. The cycle is induced and controlled by various hormones and is divided into four phases:

- **Proestrus**
- **Estrus**
- **Metestrus**
- **Diestrus**

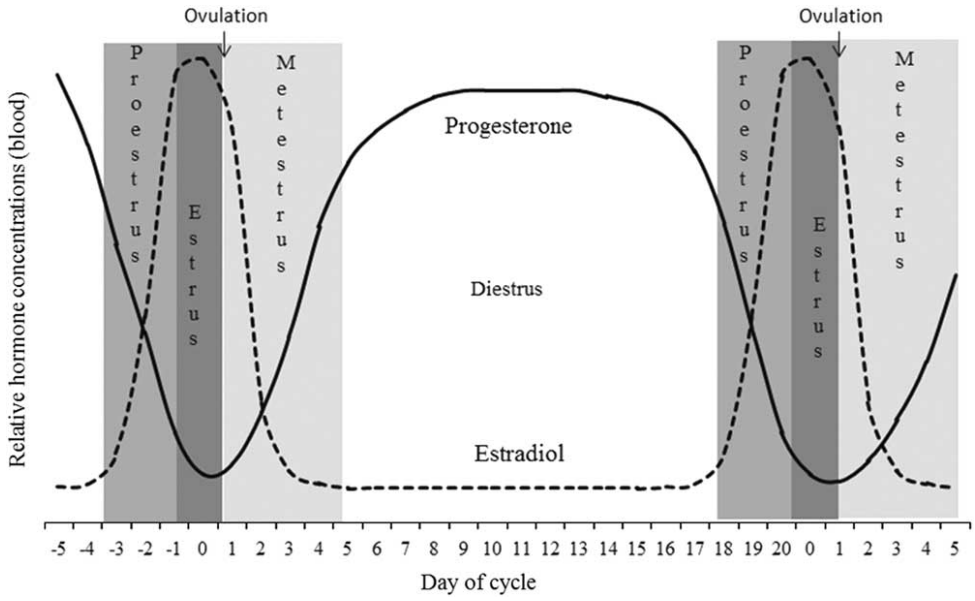

Figure 1.1: Estrous cycle

1.1 Estrus

[2] Estrus, often referred to as heat, is a period each month when a cow is receptive to reproduction through a demonstration of behavior such as receptivity to mounting. During estrus the cow is receptive to the sexual advances of the male. This phase usually

lasts about 18 hours; however, about one-third of the observed periods lasted less than 12 hours and some as short as 7 hours. This phase of the cow's sexual cycle is marked by behavioral and physical signs that facilitate heat detection which is crucial to the success of a dairy operation.

1.2 Prominent Signs of Estrus

[2] Due to drastic changes in hormones, cows show very distinct behaviour during Estrus which a trained eye could pick up.
Observable signs of heat include but are not limited to:

- Attempting to mount other cattle.
- Standing to be mounted by other cattle.
- Increase in body temperature.
- Smelling other females.
- Trailing other females.
- Loss of appetite.
- Increased restlessness and activity.
- Vulva swelling and reddening.
- Clear vaginal mucous discharge.

1.3 Insemination Timing Challenge

[3] For successful insemination, insemination must occur at the correct time of the cow's estrus cycle. There is a good correlation between when cows start heat and when they ovulate.
Statistics done in Artificial Insemination research shows that pregnancy is most likely when cows are inseminated 4-12 hours after they enter estrus.

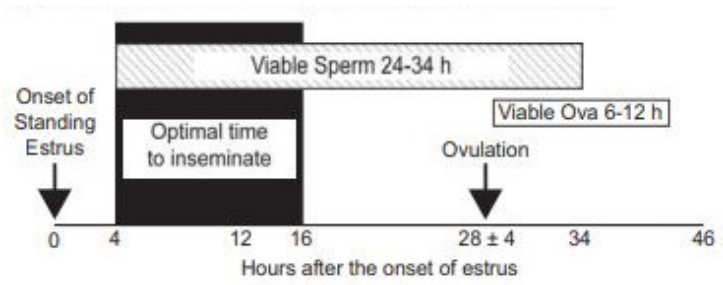

Figure 1.2: Ideal Time Period for Successful Insemination

This clearly shows that there is a small time frame in which the insemination have a higher success rate.
So, **estrus detection** is of paramount importance to ensure the viability of the dairy farms.

2 Benefits of Estrus Detection

2.1 Economic Interest

[4] The detection of the Estrus maximizes the chances for successful insemination, the farmer obviously is interested in this to rejuvenate the herd by getting calves; and mostly because cows need to get pregnant and calve in order to produce milk.
After a dairy cow calves, she produces milk in a predictable pattern called the lactation curve. As shown in the graph. Cow's milk production is highest at the beginning of lactation and then declines steadily.

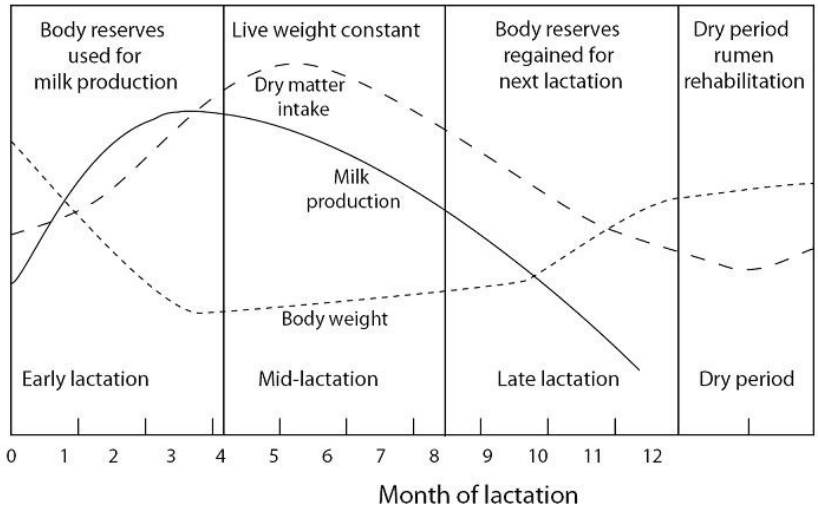

Figure 1.3: Lactation Curve

We can clearly see that each new lactation must be initiated by calving; therefore, to maintain an annual milk production cycle, a cow must maintain an annual gestation cycle. The average gestation period for dairy cows is 280 days. If you maintain an annual calving cycle, that leaves only 85 days between the time a cow calves and the time she must conceive again.

This cycle of calving and conceiving has to be strictly followed to make the herd profitable. Otherwise, if estrus is missed and cows are not inseminated in time, the cost of feeding and caring for the cow will far exceed the profit from the milk and the cow would have to be taken down.

2.2 Estrus Control and Synchronising

Estrus synchronization is the process of manipulating the estrus cycle that results in standing estrus (heat) in the majority of the herd in a short time. It is very effective to increase the number of animals bred at a particular breeding season.

The synchronization is achieved by administrating different hormones in different phases of the cycle.

This technique can't be achieved and the herd can't be synchronized without an accurate tool to predict and determine the estrus so that it can be altered to will.

3 Problems with Estrus Detection

Dairy cows are very aware of their environment and their condition is very sensitive to the slightest changes.
Traditionally Estrus detection is done visually by the farmer who is familiar with his herd and can reliably tell a cow is demonstrating Estrus-related behaviour.
Nowadays, the unnatural environment that modern farms produce (Cows are indoors most of the time, Overcrowding, artificially enhanced food...) can make cows uncomfortable and not show strong heat signs.
To add on that other can cause heats to be missed:

– Increased herd size leading to more cows per member of staff
– Failure to recognise oestrus due to inadequate staff training

4 State of the Art

Heat detection can be a difficult and time-consuming endeavor. It's especially tough in the harvest seasons because of the extra work load.
Finding a qualified person to put in the necessary time and effort to properly identify estrus can be a challenge.
Missing estrus can have disastrous consequences financially, in fact, failure to detect estrus or erroneous diagnosis of estrus results in an estimated annual loss of over $300 million to the dairy industry in the US.
As a response to these conditions multiple companies develop and commercialize solutions for the dairy industry facilitating heat detection.
[5] The most popular solutions commercialized today are:

- **Self-adhesive indicators:**
  Adhesive breeding or mounting indicators are friction-activated to identify animals in standing heat.
  EstroTect™ is the most used brand of this type of tool.
  This type of solution works like such: An indicator is fixed on the cow's back and the breeding indicator's surface ink is rubbed off by the friction of mounting, showing that mounting activity has occurred.

With each mount, the surface ink will gradually reveal an indicator color indicating a true standing heat.

One disadvantage to the self-adhesive indicators that some people have reported the adhesive may not adhere through multiple mountings.

- **Glued-on indicators:**

  Is similar to the first type of products but this line uses glue to adhere the indicator.

  The detectors are pressure sensitive and activated by standing heat behavior. Kamar Heatmount™ leads the market in this type of products and their detectors are a long-established brand.

  The built-in timing mechanism in the Kamar Detector helps distinguish between true standing heat versus false mounting activity.

  Glued onto the tail head, pressure from the brisket of a mounting animal requires approximately three seconds to turn this detector from white to red thus indicating true heat.

- **Electronic Mounting Monitoring:** Electronic monitoring is a more sophisticated but more costly method to track estrus.

  It involves using a wireless transmitter attached above the tail of a cow to digitally send a data stream with the date, time and duration of every mounting event. Data is recorded and easily accessible on software.

  The AccuBreed™ brand features an electronic monitoring solution, the components capture every mounting event as it happens. All mounting data is collected, stored and instantly reported to the farmer through their software.

[6] [7]

Solution Type	Solution Type	Performance Rate
Observation		65%
Glued-on indicators	Estrotect	70%
Self-adhesive indicators	Karmar Heatmount	84%
Elecronic Monitoring	AccuBreed	91%

Conclusion

It is obvious that the present solutions perform well and can be a great addition to the farmer in dealing with estrus detection and prediction.
But the price points and the overall design and functionalities are not quiet adequate with the Tunisian agricultural culture and market.
The following chapter will present an Electronic Monitoring solution that is low-cost and more in synch with the local market

2

Proposed Solution

Introduction

This chapter will be about explaining the solution we produced.
It will introduce the key metrics that will be used to determine Estrus probability and It will introduce in details the used hardware and software for this solution.

1 Key Metrics

Estrus is characterized by a multitude of changes in body and behaviour of the cow but 3 signs stand out as being the most reliable and quantifiable.

- **Temperature:**
  [8] In natural settings, a healthy cow's temperature in the range of (38.3°C - 39°C)
  During heat the cow's body temperature suddenly increases by 1.5 to 2 °C
- **Activity:**
  Cows tend to be very sedentary animals research show that a cow spends up to 14 hours of resting and lying down and only 2 to 3 hours of walking or standing.
  In the day or 2 leading to estrus, it has been noticed that cows are restless and tend to be more active than usual.
- **Ruminition:**
  Cows spend an average of 8 hours a day in rumination. Due to lack of interest in food during the period leading to the estrus, cows spend "only" an average of 2 hours for rumination per day.

The constant monitoring of these behavioural patterns, allows for data-based predictions of upcoming estrus.

2 Hardware Resources

2.1 DS18B20 Temperature Sensor Cable

[9] The DS18B20 Temperature Sensor is a one wire digital temperature sensor. This means that it requires only one data line to communicate data.
This sensor can measure temperatures ranging from -55°C to +125°C with an accuracy of +-5%.
Data received from the single wire is in the ranges of 9-bit to 12-bit.
1-wire protocol is an advanced level protocol and each DS18B20 is equipped with a unique serial code of 64 bit which helps in controlling multiple sensors via a single pin of the controller, in fact calling the unique address of the sensor is enough to get the data from

that sensor.
This sensor will be used to measure the body temperature of the cow.
The choice for this Sensor is based on the fact that it is water and dust proof, and made of stainless steel; allowing it to be used without fear in humid and dusty conditions.
Also, the one wire protocol that this sensor is based upon allows for an easy access to the data and more secure connection to the micro-controller.

Figure 2.1: DS18B20 Sensor Cable

2.2 ADXL345 Accelerometer

[10] The ADXL345 is a small, thin, low power, 3-axis accelerometer with high resolution measurement up to ±16 g.
The ADXL345 is well suited for mobile device applications. It measures the static acceleration as well as dynamic acceleration resulting from motion or shock.
This sensor will be used to measure the general activity of the cow but also, thanks to special calibrating libraries, as a rumination sensor.
The choice of this sensor is based on the fact that this sensor is very thin, and uses very low levels of current allowing for a bigger battery life for the device.
Also, this sensor has a built-in ADC converter, allowing for one less part on the device.

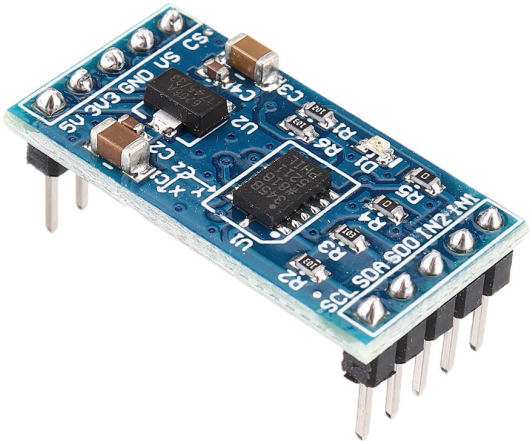

Figure 2.2: ADXL345 Accelerometer

2.3 ESP32-WROOM32

[11] The ESP32-WROOM32 is a low-cost, low-power system on chip microcontroller with integrated Wi-Fi and Bluetooth.

This card is especially adapted for mobile IOT projects because of the variety of pins it possesses and adaptability of the card.

This card will be used as a on-collar local center for receiving data from the sensors.

Thanks to the built-in WIFI module, this card will be communicating with the Raspberry Pi card to send it all the collected data.

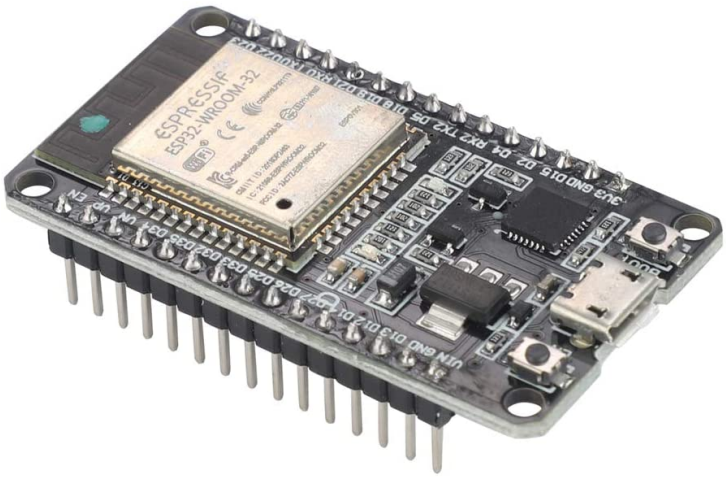

Figure 2.3: ESP32-WROOM32 Board

2.4 Raspberry Pi Zero WH

[12] Raspberry Pi Zero is a single-board mini computer. The boad presents built-in LAN network and Bluetooth capabilities.
The economical price, small-size and open-source design of this module makes it a suitable pick for a broad range of applications.

The Raspberry Pi will be the central hub for this application; the (theoretically multiple) ESP32 Card sends the collected data to the Raspberry Pi that stores the data internally (as a backup) and uploads it to an online server where it can be accessed and monitored by the monitoring mobile application.

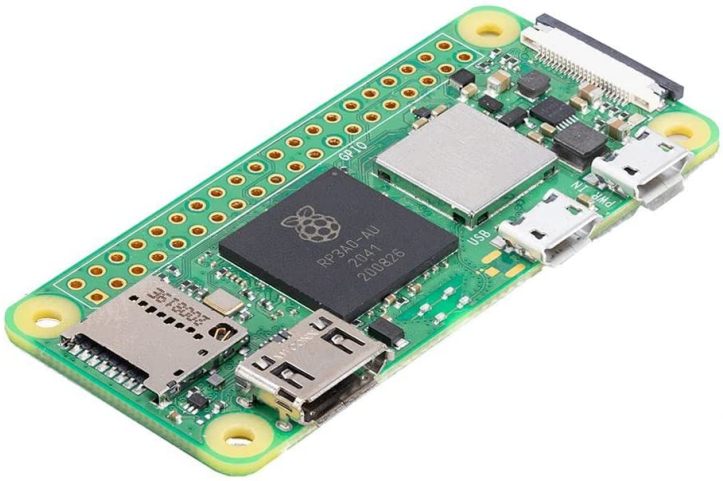

Figure 2.4: Raspberry Pi Zero WH Board

3 Software Resources

3.1 Communication Protocols

3.1.1 Hyper Text Transfer Protocol (HTTP)

The HTTP is an application layer protocol designed to transfer information between networked devices and runs on top of other layers of the network protocol stack.
A typical flow over HTTP involves a client machine making a request to a server, which then sends a response message.
This protocol is used in our case to connect the ESP32 card and the Raspberry Pi board, it allows the ESP32 to easily send data to the Raspberry Pi.

3.1.2 Hyper Text Transfer Protocol Secure (HTTPS)

HTTPS is simply HTTP with an added layer of TLS encryption.
HTTPS uses TLS (SSL) to encrypt normal HTTP requests and responses, making it safer and more secure.
This protocol is the standard when working online, it allows information to be securely transferred back and forth between servers and clients.
This protocol is used in our case to secure the communications between the Raspberry Pi card and the Firebase Server which stores all the collected data, and between the server and the mobile app.
The protocol keeps communications secure so that malicious parties can't observe what data is being sent. As a result usernames and passwords can't be stolen in transit when users enter them into the authentication form of the mobile app.

3.1.3 Secure Shell (SSH)

Secure Shell is a secure communication protocol. This connection protocol requires an exchange of encryption keys at the beginning of the connection.
Because it is required to use an extra monitor and external devices to access the Raspberry Pi card's graphic interface, we used SSH protocol to code on the Raspberry via a Laptop.

3.2 Local Server

The Raspberry Pi card collects the received data from the ESP32 in a local server built on the card.
This server is built on '**NodeJs** and is contained with **Docker** technology.

3.3 Docker

Docker is an open source software platform for creating, deploying and managing virtual application containers on an operating system. The services or functions of the application and its various libraries, configuration files, dependencies and other components are grouped within the container. Each running container shares the services of the operating system. In our project docker was used on Linux OS ,on which the Raspberry Pi runs, to contain the local server implemented on the card.

3.4 Google Firebase

Firebase is an online platform created by Google, it is mainly used to power web and mobile applications' backend. Firebase offers:

- Hosting
- Authentication
- Real-time Database
- Test Lab
- Notifications

3.5 Mobile Application

Our mobile application is built in the Ionic framework.
Ionic is a hybrid mobile application development framework. It allows for building cross-platform applications from:

- Web technology for instance: HTML5, CSS and JavaScript.
- Framework in this case Capacitor.

4 General Functioning

The monitoring device is in the form of a small box attached to the neck of the monitored cow in a tight-fitting collar allowing for precise measurements. The device is made of:

- 1 Esp32 Card.
- 2 ADX345 Accelerometers.
- 1 DS18B20 Temperature Sensor.

The necessary measurements are made by the sensors: body temperature, activity and rumination.
The data is acquired by the ESP32 card that is connected to a WIFI network generated by the Raspberry Pi card. Using that WIFI connection the ESP32 card sends the acquired data to the Raspberry Pi which stores the data internally and then uploads the data on an online Firebase server.
The data is then sorted and organized in the server.
The mobile application, gets access to the server and displays the information relative to each cow in real-time.
The app shows graphs of the evolution of each cow's stats and sends an alert to the user if it predicts the happening of an Estrus.

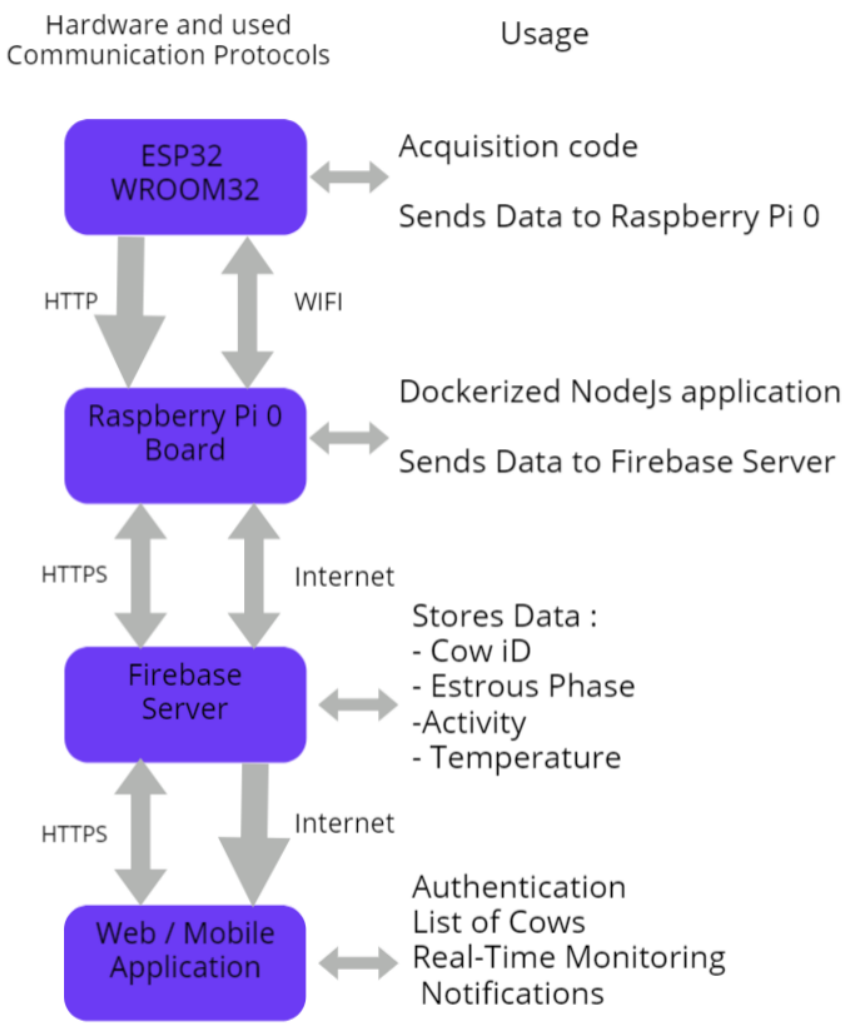

Figure 2.5: Simplified View of the Different Components, Protocols and Features

Conclusion

The proposed solution has the advantage of being built on well documented sensors with dedicated libraries and, for the most part, open-source software.
This configuration allows for easy interconnections between hardware and software and encourage future improvements on the device.

3

Implementation of the Solution

Introduction

In this chapter we will be taking a look at the wiring of the different sensors to the ESP32 card, portions of the code that allow the acquisition of the data on the ESP32 and the transmission to the Raspberry Pi Zero card.
This chapter will also present the present features on the mobile applications.

1 Hardware and Software Setup and Configuration

1.1 ESP32-WROOM32 Board

Before using the card and loading the data acquisition programs for each component, the ESP32 card has to be initialized and setup following universal protocols to allow for smooth and reliable communication between the card and the different sensors.

1.1.1 Setup

To set up the the ESP32 card, **Arduino IDE** was used as a programming interface to write and upload the necessary code and to import the needed libraries to allow for communication with the different sensors.
Arduino IDE was used because it is open-source and allows for easy and seamless integration for all needed protocols.

```
#include <Wire.h>
#include <Adafruit_Sensor.h>    // Adafruit  sensor library
#include <Adafruit_ADXL345_U.h>  // ADXL345 library
#include<cmath>
#include <WiFi.h>
#include <HTTPClient.h>
#include <OneWire.h>
#include <DallasTemperature.h>
```

Figure 3.1: Import of all Needed Libraries

```
  delay(1000);
  WiFi.disconnect();
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi..");
  }

  Serial.println("Connected to the WiFi network");

}
```

Figure 3.2: Connecting the ESP32 Card to the Internet

1.2 Temperature Sensor DS18B20

1.2.1 Wiring

As seen in the previous chapter, the DS18B20 temperature sensor works with the one wire protocol; meaning that only one GPIO (General Purpose Input Output) pin is needed for transmitting the Data.
One **pull-up resistor** (4.7kΩ) is needed on the power line of the sensor to get the recommended voltage for the sensor.

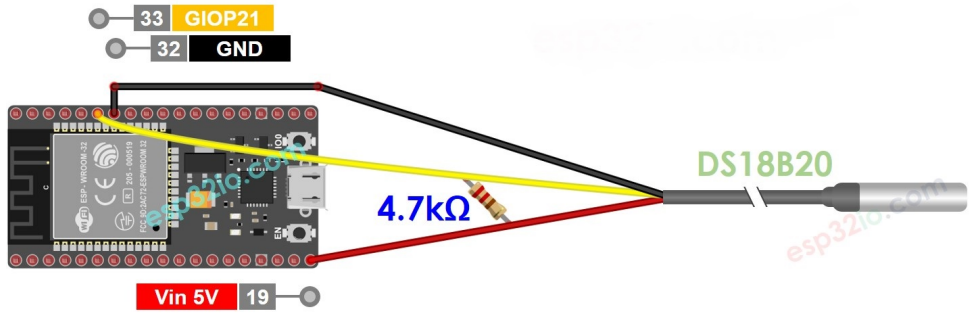

Figure 3.3: Theoretical Scheme of DS18B20 Wiring with ESP32

1.2.2 Receiving and Transmitting the DS18B20 Data

This portion of the code shows the acquision of the data by the ESP32 card and the sending of the data via an HTTP request to the Raspberry Pi Zero card.

```
74  sensors.requestTemperatures();
75  float temperatureC = sensors.getTempCByIndex(0);
76  Serial.print(temperatureC);
77  Serial.println("ºC");
```

Figure 3.4: Data Acquisition from the DS18B20 Sensor to the ESP32

```
79  //Satrt sending Temperature
80  HTTPClient http;
81  if ((WiFi.status() == WL_CONNECTED)) { //Check the current connection status
82    Serial.println("current temperature"+String(ratio));
83    http.begin("http://192.168.137.157:3000/temp?temp="+String(temperatureC)); //Specify the URL
84    int httpCode = http.GET();                                        //Make the request
85
86    if (httpCode > 0) { //Check for the returning code
87
88        String payload = http.getString();
89        //Serial.println(httpCode);
90        Serial.println(payload);
91      }
92
93    else {
94      Serial.println("Error on HTTP request");
95    }
96    http.end(); //Free the resources
97    //End Sending Temperature
98
```

Figure 3.5: Sending the Received Data from the DS18B20 Sensor to the Raspberry Pi

It is to be noted that for using this sensor, the <wire> library had to be imported in the setup of the ESP32 card because of the **one wire protocol** that the DS18B20 works with.

1.3 Accelerometer ADXL345

1.3.1 Wiring

The ADXL345 accelerometer gives acceleration data over the X, Y and Z axes.
2 GPIO pins are needed for the ESP32 to receive the data.

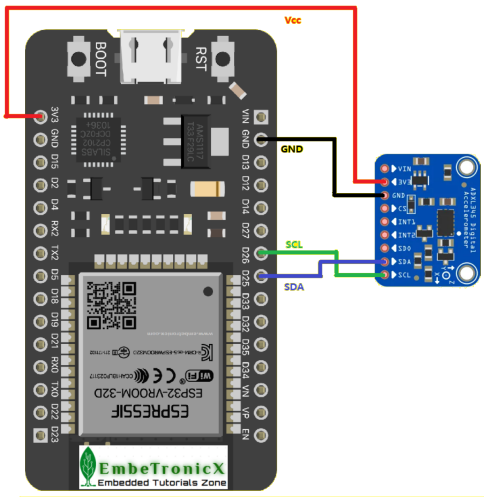

Figure 3.6: Theoretical Scheme of ADXL345 Wiring with Esp32

The second ADXL345 can be wired on the same pins as the first sensor, the unique address that characterizes each sensor allows for easy access to the data collected by each

sensor individually.

1.3.2 Receiving and Transmitting the ADXL345 Data

The ADXL345 gives the data in the form of 3 float values of acceleration following X,Y and Z axes.
We created a simple mathematical formula transforming those accelerations into a single value that translates the intensity of the movements.

```
49  float x=0;
50  float y=0;
51  float z=0;
52
53  float old_x=0;
54  float old_y=0;
55  float old_z=0;
56
57  float ratio;
58  for(int j=0;j<10;j++){
59    sensors_event_t event;
60    accel.getEvent(&event);
61    x+=  pow(event.acceleration.x-old_x,2);
62    y+=  pow(event.acceleration.y-old_y,2);
63    z+=  pow(event.acceleration.z-old_z,2);
64
65    old_x =  event.acceleration.x;
66    old_y =  event.acceleration.y;
67    old_z =  event.acceleration.z;
68    delay(500);
69  }
70  ratio=(x+y+z)/3;
71  Serial.print("Mouvement ratio: ");
72  Serial.println(ratio);
73
```

Figure 3.7: Data Acquisition from the ADXL345 Sensor to the ESP32

```
99
100 //Start Sending Mouvement
101 Serial.println("current mouvement"+String(ratio));
102 http.begin("http://192.168.137.157:3000/mouvement?mouvement="+String(ratio)); //Specify the URL
103 httpCode = http.GET();                                        //Make the request
104
105 if (httpCode > 0) { //Check for the returning code
106
107     String payload = http.getString();
108     //Serial.println(httpCode);
109     Serial.println(payload);
110   }
111
112 else {
113   Serial.println("Error on HTTP request");
114 }
115 http.end(); //Free the resources
```

Figure 3.8: Sending the Received Data from the ADXL345 Sensor to the Raspberry Pi

2 Raspberry Pi Local Server

To be able to get the data sent from the ESP32 and upload data onto the Firebase server, the Raspberry Pi card has to have a local server that answers to the HTTP requests from the ESP32 card.

This server was built on the Linux-run Raspberry Pi using **NodeJs** to code the server and **Docker** to contain the server and giving it necessary dependencies.

```
const express = require('express');
const { initializeApp, applicationDefault, cert } = require('firebase-admin/app');
const { getFirestore, Timestamp, FieldValue } = require('firebase-admin/firestore');



const app = express();
const PORT = 3000;

const serviceAccount = {
    "type": "service_account",
    "project_id": "pfa-project-e7c19",
    "private_key_id": "c75a802ef100f1f032680c54555673b1f6932cd2",
    "private_key": "-----BEGIN PRIVATE KEY-----\nMIIEvQIBADANBgkqhkiG9w0BAQEFAASCBKcwggSjAgEAAoIBAQDis94EWnTcL8FC\nclN2j44eVjafAWjyrNvyEEhPzuDL+ezR0
    "client_email": "firebase-adminsdk-8yav0@pfa-project-e7c19.iam.gserviceaccount.com",
    "client_id": "107907825513703233886",
    "auth_uri": "https://accounts.google.com/o/oauth2/auth",
    "token_uri": "https://oauth2.googleapis.com/token",
    "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
    "client_x509_cert_url": "https://www.googleapis.com/robot/v1/metadata/x509/firebase-adminsdk-8yav0%40pfa-project-e7c19.iam.gserviceaccount.com"
  }


initializeApp({
  credential: cert(serviceAccount)
});

const db = getFirestore();
```

```
app.get('/temp', (req, res)=>{
    db.collection('users').doc("omarhajjouji@gmail.com").collection("cows").doc("1").update({"temp":req.query.temp}).then((res)=>{
        res.status(200);
        res.send({"temp":req.query.temp});
    }).catch((err)=>{
        res.status(200)
        res.send({"temp":req.query.temp});
    })
});


app.get('/mouvement', (req, res)=>{
    db.collection('users').doc("omarhajjouji@gmail.com").collection("cows").doc("1").update({"mouvement.time":FieldValue.arrayUnion(Date.now()/10000),"mouvement.value":R

        res.status(200);
        res.send({"mouvement":req.query.mouvement});
    }).catch((err)=>{

        res.status(200)
        res.send({"mouvement":req.query.mouvement});
    })
});




app.listen(PORT,"0.0.0.0", (error) =>{
    if(!error)
        console.log("Server is Successfully Running, and App is listening on port "+ PORT)
    else
        console.log("Error occurred, server can't start", error);
    }
);
```

Figure 3.9: NodeJs Code for the Making of the Local Server on the Raspberry Pi

3 Firebase Server Configuration

The Google Firebase platform allows, using the present tools, to build a functioning server with no real need to write explicit code.

In our case the Server is storing and updating the data sent via HTTPS protocol from the Raspberry Pi in real-time.

This server is accessed in real time by the web/mobile app to get and display the data.

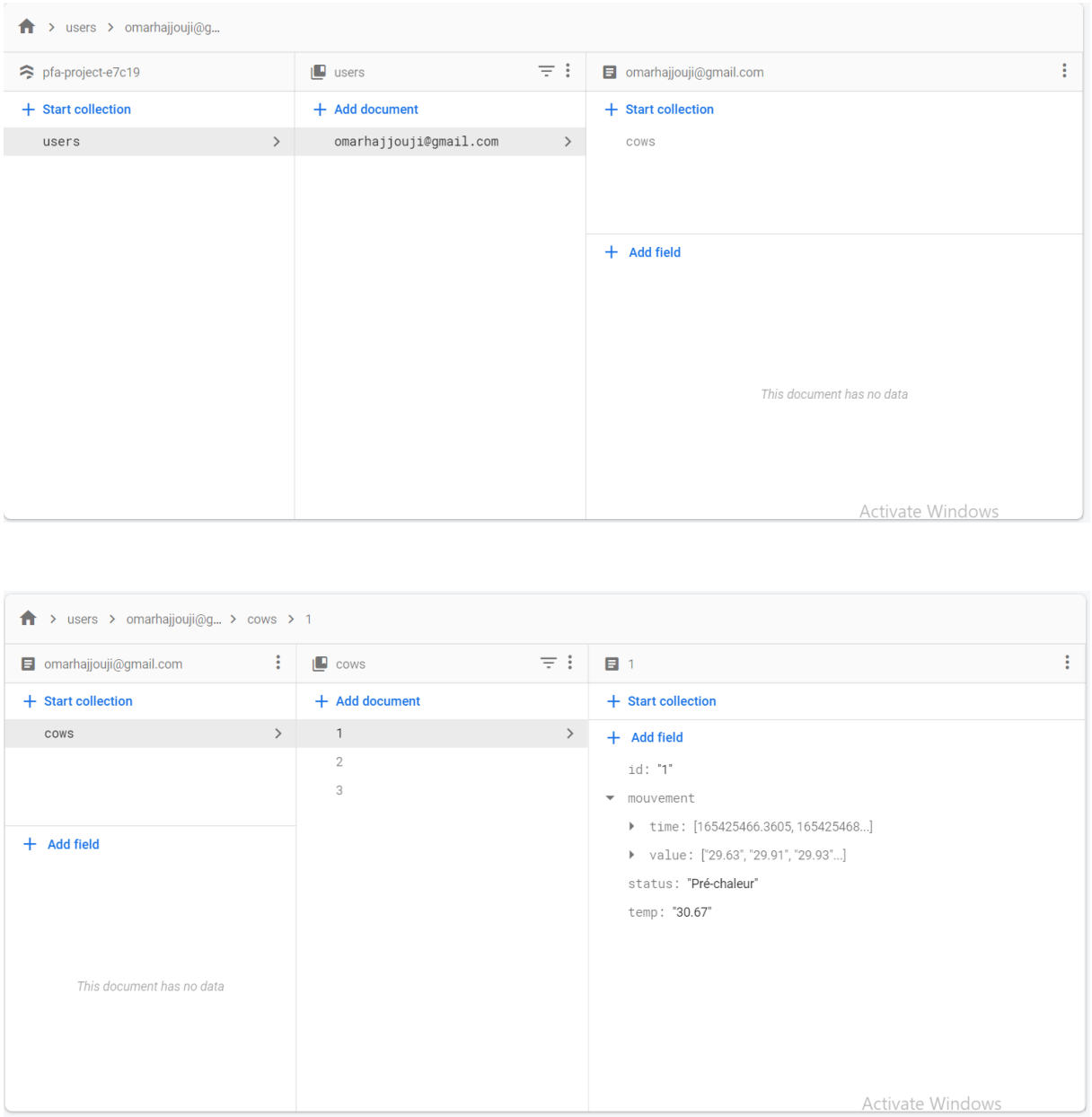

Figure 3.10: Overview of the Google Firebase Server

4 Mobile Application

4.1 Authentication

The app allows for unique access for each client, via a unique User id and password. The access is protected via the Google Firebase platform which ensures the legitimacy of the information.

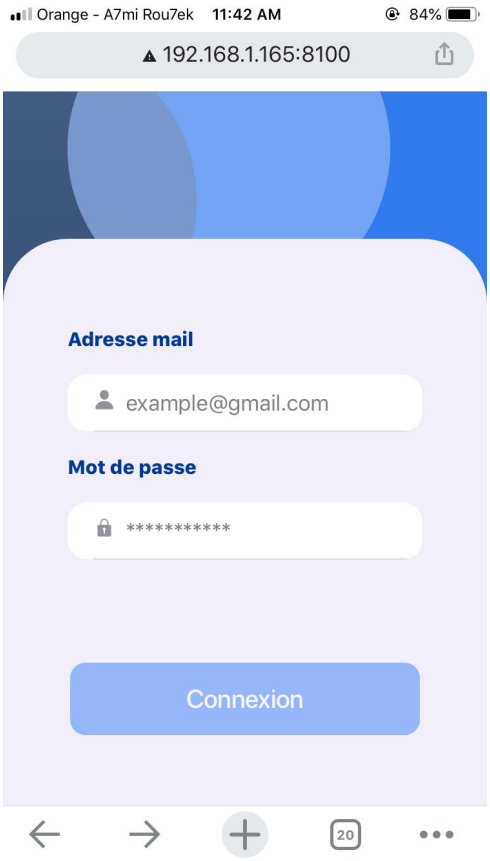

Figure 3.11: User Authentication UI

4.2 Cow iD

For each client, the app stores and shows all the monitored cow with their unique id and current state.
Each cow can be individually accessed and monitored.

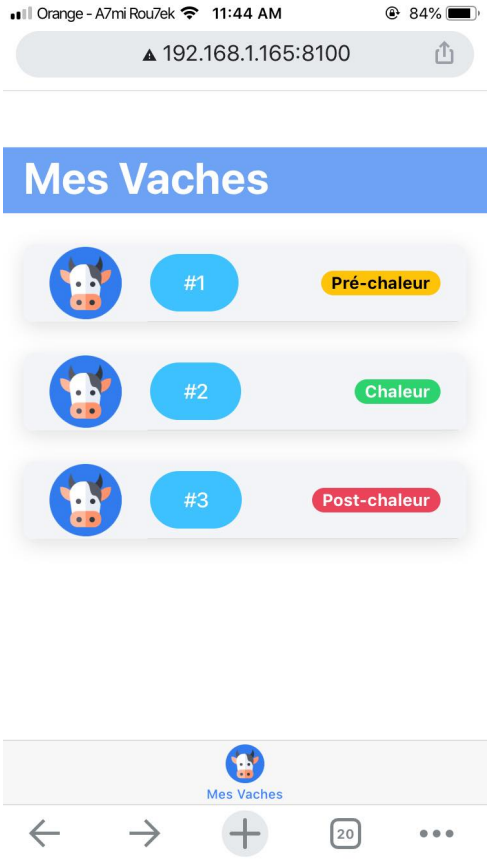

Figure 3.12: Individual Cow Access UI

4.3 Real-Time Monitoring

For each cow, the app shows in real-time the evolution of the temperature, activity and rumination.

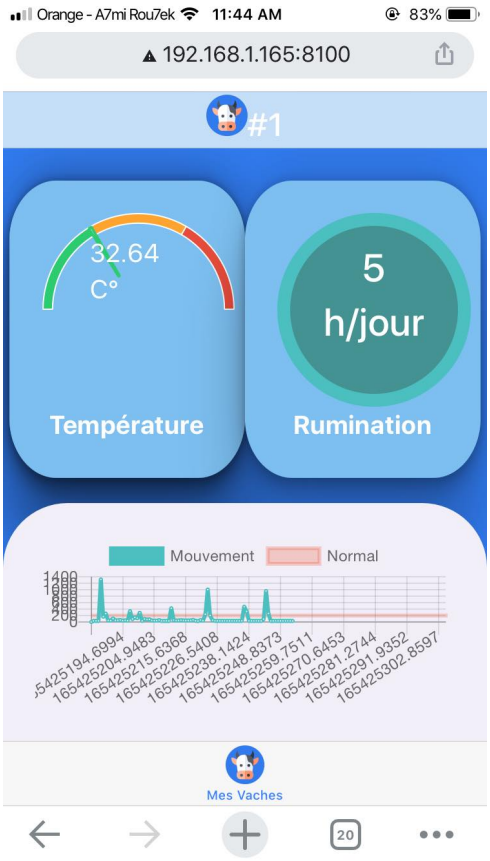

Figure 3.13: Real-Time Monitoring UI

4.4 Notifications

The application presents the feature of sending appropriate notifications detailing the unique iD of the cow that has an Estrus-like behavior or metrics.
The threshold that the data levels have to exceed in order for an alert to be sent, will be later set by placing the device on cows and collecting normalized data that would serve as reference points for abnormal behaviors

Conclusion

The presented solution has a lot of advantages that could be accommodating and adapted to the Tunisian dairy industry.
In fact, the device is low-cost (the collar device has a cost of around **180dt**) which is an attractive price for farmers. Also, the device is very low-maintenance, and that is thanks the tough-to-break type sensor that we selected and to the ESP32 capabilities of storing programs and rebooting the system even after being completely off power.

General Conclusion and Perspectives

General Conclusion

This report introduced a concerning problem that is more and more prominent in the local agriculture industry and explained all the interest and benefits of having a proper estrus detection aid.
Then proposed a viable and attractive solution especially for the Tunisian market; by bypassing the traditional observation-based predictions and turning towards a more data-driven prediction model, this can drastically enhance profitability for dairy farms.
And in the end showed one by one the different resources and steps required to make this solution.

Perspectives

The general lining of the built solution and the generic and open-source nature of most component and software make it inherently scalable.
More metrics can be used to enhance and improve the prediction model:

- An in-vein hormone detector device could be used to detect peaks of certain hormones which are a prominent indicator of different estrous cycle phases.
- A GPS module could be added, and would be particularly useful when multiple cows are equipped with the collar device, to detect the cows being close together which is an abnormal behaviour and an estrus sign.
- A pressure sensor could be added an equipped on the back of cows to accurately detect the acceptance of mounting, which is arguably the most prominent sign of estrus.

Other uses can also be considered for the collected data as a parallel process for the estrus prediction. The data could be useful, for example, for **Health Monitoring**, **Disease Detection** or even **Stress Condition**.

Bibliography

[1] S.Reith and S.Hoy. Behavioral signs of estrus and the potential of fully automated systems for detection of estrus in dairy cattle. *Animal*, 2017.

[2] Estrus detection in dairy cattle. https://www.aces.edu/blog/topics/dairy/detecting-estrus-in-dairy-cattle/.

[3] Heat detection and timing of artificial insemination. https://www.selectsires.com/article/ss-blog/2020/11/03/heat-detection-and-timing-of-artificial-insemination.

[4] Managing cow lactation cycles. https://www.thecattlesite.com/articles/4248/managing-cow-lactation-cycles/.

[5] Cattle estrous detection. https://extension.umn.edu/beef-cow-calf/cattle-estrus-detection.

[6] Estrotec proven results. https://estrotect.com/fr/proven-performance/.

[7] Karmar detection efficiency. https://kamarinc.com/kamar-detectors/.

[8] Giovanni Caola Giuseppe Piccione and Roberto Refinetti. Daily and estrous rhythmicity of body temperature in domestic cattle. *Online*, 2003.

[9] Ds18b20 manufacturer datasheet. https://www.maximintegrated.com/en/products/sensors/DS18B20.html.

[10] Adxl345 manufacturer datasheet. https://www.analog.com/en/products/adxl345.html.

[11] Esp32 wroom32 manufacturer datasheet. https://www.espressif.com/en/products/socs/esp32.

[12] Raspberry pi zero manufacturer datasheet. https://datasheets.raspberrypi.com/rpizero2/raspberry-pi-zero-2-w-product-brief.pdf.