Expertise
and insight
for the future

Trung Hoang Nguyen

Jani Valkonen, Joona Sarvi, Eemi Sammalniemi

Maisin Maatila - Big Flash

Metropolia University of Applied Sciences

Information Technology - Smart Systems,

Mechanical engineering - Manufacture and production technology

Innovation Project

Final Report

30 November 2022

Contents

List of Abbreviations

GND	Ground.
HTTP	Hypertext Transfer Protocol
I2C	Integrated Circuit
IDE	Integrated Development Environment
IoT	Internet of Things
LED	Light emitting diode
NTP	Network Time Protocol
PIR	Passive Infraed Sensor
PWM	Pulse-width modulation
SSID	Service Set IDentifier

1 Introduction

This report is the final document for the Innovation Project at Metropolia University of Applied Sciences. This project is a part of Big-Flash, which aims to help SMEs to utilize artificial intelligence and robotics as a part of their production lines. Our team is required to develop a self-pick-up system solution to optimize time and cost for Maisin' customers. For this report, the goal is to show the implementation and outcome the product demo.

Nowadays, with the development of technology, we live in a “new economy”, where customers can reach their smartphones and order almost everything on demand. One of the hottest areas for growth in the now economy is order for the pick up category, especially after Covid-19. Most people became very familiar with ordering and self-pickup or deliveries to their home, as working from home grew as a trend. A new trend has been increased and forced manufacturers to build a new self-pick-up system. This process allows for quick and convenient solutions to both manufacturing and customers,

Maisin Maatila project embraces the above idea to gratify customers and optimize logistics cost for manufacturers. Owning an automated pick-up system can save human resources, increase productivity, make it scaleable, and focus more on management. Our system will essentially permit the entry of drivers via their car plate, compared to its on our database, then send a signal to open the entrance gate. At the existing gate, an ultrasonic distance sensor will be installed in order to recognize outcome cars and send them back to database. Maisin Maatila project ensures bringing convenience, saves time and optimizes supply chain for both customers and the company

In conclusion, This project will offer a useful solution for SMEs, small manufacturers and can be upscale further in any kind of category. The following chapter provides complete implementation details of the hardware and software parts and how the whole product came together. The last chapter will reveal our problems during the developing demo.

Mechanical engineering students were responsible for designing and manufacturing the parts of the gate, and information technology students were responsible for implementing the programming.

2 Hardware

2.1 Components

In this project, we will use several different manufacturing methods for the manufacture of components. The assembly uses ready-made components selected by Arduino, as well as an H-bridge and a sensor. The programming language used in the product has several different stages.

The used of components for this project:

Type	Component	Name	Price
Controller		Arduino Portenta H7	99e
Sensor	Motion Sensor	HC-SR501	5e
	Ultrasonic Sensor	Ultrasonic Ranger (Grove)	5e
Actuators and others	H-Bridge	Motor Driver L298N	13e
	Camera	Arduino vision shield	45e
	Motor and Material cost		54e

Table 1: List of final components for this project

2.2 Arduino Portenta

Based on the requirements of project, we will need a small and powerful controlling unit to process take images and run 24/7, communicate with server and control sensors via Wifi or Ethernet so that Arduino Portenta H7 will be suitable at all. \Due to its dual- STM32H747: Arm Cortex M7 and Cortex M4, the Arduino Portenta H7 could be the appropriate board for creating high-end projects requiring a board with high computational power. In addition, the Portenta H7 can be programmed in C, C++, micro Python, and JavaScript.

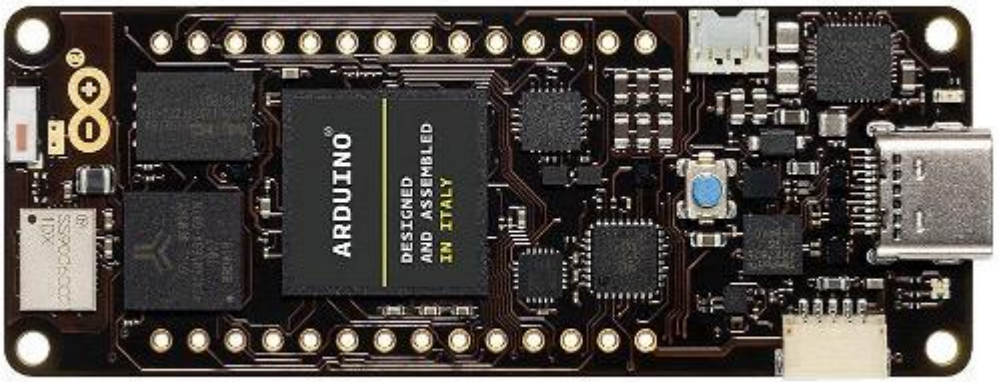


Figure 1. Arduino portenta H7

By including an ultra-low-power Himax camera module that can detect motion on its own, waking the Portenta board only when necessary to save power, and two omnidirectional MP34DT06JTR microphones set up as a beam-forming array, the Portentia Vision Shield aims to expand the Portenta H7's functionality into computer vision and voice applications. This camera will be used to identify the license plate

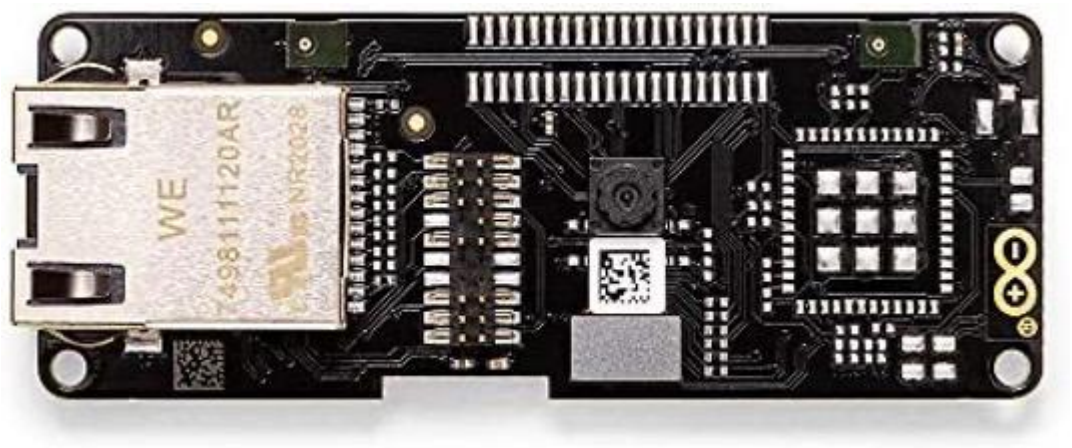


Figure 2. Arduino Portenta Vision Shield

2.2 H-bridge

An H-bridge is a voltage-switching electronic circuit. These circuits allow DC motors to rotate in either direction, making them useful in robotics and other applications. It gets its name from the way it's typically depicted on schematic diagrams: with four switching elements arranged like the forks of a "H" and the load connected like the bar in the middle. An Hbridge obtains the high and low signals from the controller and acts as a switch to control the Vin of the motor.

Figure 3. H-bridge.

2.3 PIR Sensor

The PIR sensor, which is used as a motion sensor, detects the movement in front of the sensor. The sensor is used in the program to detect movement in front of the boom and to start the camera, it is possible to adjust the detection distance and the sensor is adjusted so that it detects all things moving closer than three meters. (Figure 4)

Figure 4. PIR Sensor

2.4 Ultrasonic Sensor

An ultrasonic sensor is an electronic device that measures the distance of a target object by emitting ultrasonic sound waves and converts the reflected sound into an electrical signal. Ultrasonic waves travel faster than the speed of audible sound. It's used to detect car at leaving gate by measuring distance from sensor to car then sending a log to server

Figure 5. Ultrasonic Sensor

2.5 Motor

For opening gate, we decided to use a Kaehlig 24V DC motor. Its operating voltage range is 24V. At 24V, it can rotate up to 3100s/min. this servo can be controlled by an H-bridged, which I mentioned above to adjust motor speed by changing voltage values. More details about motor programming will be added below.

Figure 6. Motor

3 Gate components

The components used in the gate were manufactured in the Metropolia machining laboratory. There were many different kinds of processing methods used to manufacture the parts that were used in this project.

3.1 Mounting sleeve

At first, the sleeve was modeled using a CAD program. (Figure 7.) The gate was attached to the motor with a sleeve made on a HAAS lathe. (Figure 8.) Using a sleeve was necessary because the motor shaft was completely round.

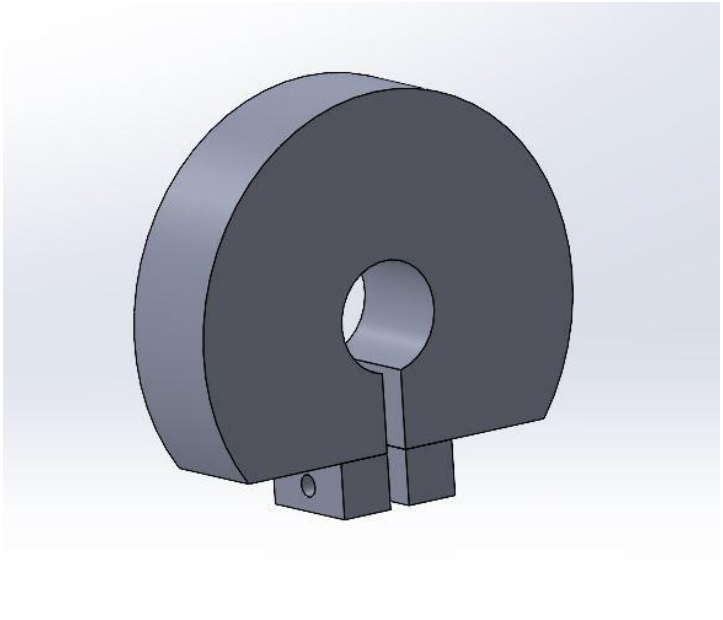


Figure 7. 3D-model of the mounting sleeve

Figure 8. Machining the center hole with a reamer

3.2 Cases

Weatherproof housings were made for all electronic components Arduinos, Sensors and H-bridge because the location of the gate is next to the entrance. CATIA V5 program was used to design the enclosures and draw the CAD models. The parts were printed with an Ultimaker 3D plastic printer.

3.2.1 Arduino portenta H7 and Portenta vision shield housing

When designing the case, the location of the camera in the vision shield was taken into account, which is why the case has a cone-shaped hollow in the middle where the camera is placed. A hole of size M20 was placed in the end, where it is possible to put a waterproof rubber grommet for the wires. (Figure 9)

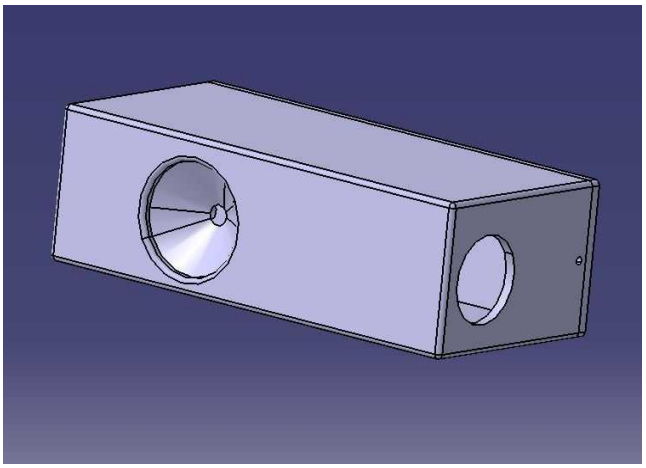


Figure 9. CAD model of the case for Arduino components.

3.2.2 Ultrasonic sensor housing

The sensor housing was dimensioned in such a way that the sensor's ultrasonic transmitters protrude slightly from the housing, so that the sensor would work correctly. There is a place on the edge for passing the wires through, which can be sealed to make it waterproof. (Figure 10)

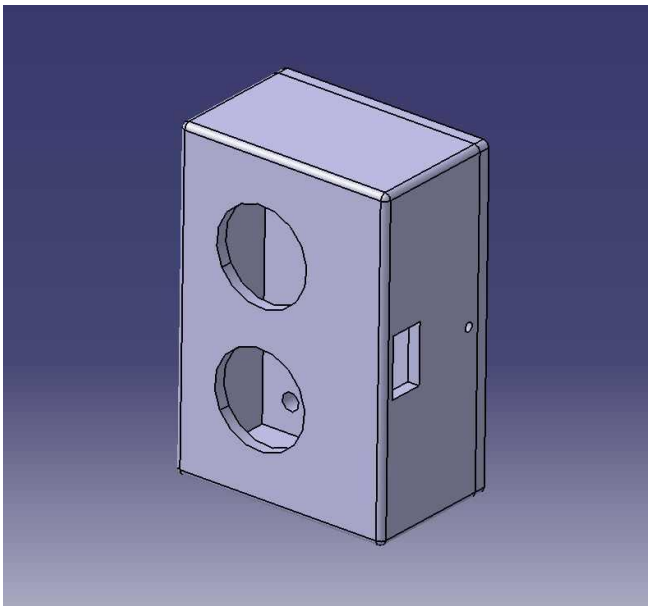


Figure 10. CAD model of the ultrasonic sensor housing.

3.2.3 H-bridge housing

The protective case for the H-bridge was designed in such a way that the H-bridge fits inside the case with all the wires. An M20 hole was also made in this case for the grommet to make the case waterproof. (Figure 11.)

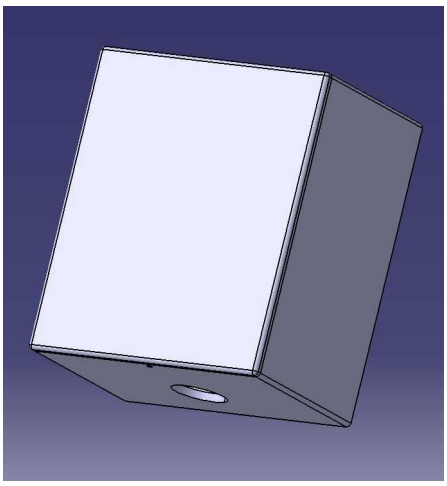


Figure 11. CAD model of the H-bridge housing.

3.3 Manufacturing of mounting plate

In order to attach the motor, a suitable bracket was cut from a 4 mm steel sheet with a laser cutter. (Figure 12.) The mounting plate is designed so that it can also be attached to the wall in the customer's premises, where the gate is designed.

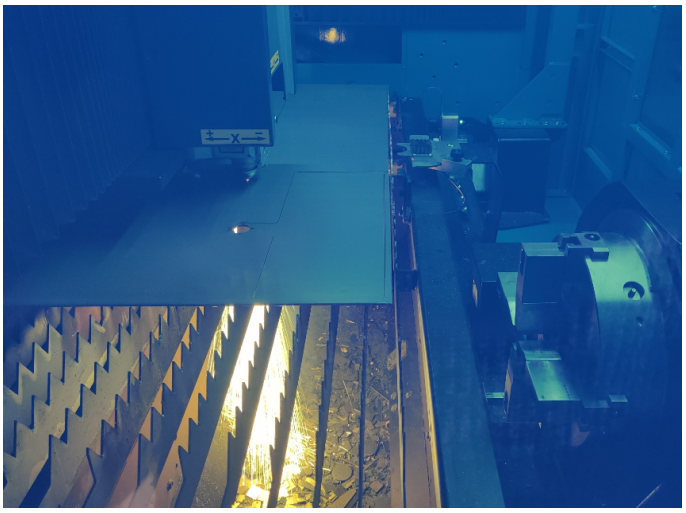


Figure 12. Manufacturing of the mounting plate with a laser cutter.

4 Assembly

The assembly was done when all the necessary components were made. The mounting sleeve was tightened on the motor shaft, after that the gate was attached to the sleeve. (Figure 13.)

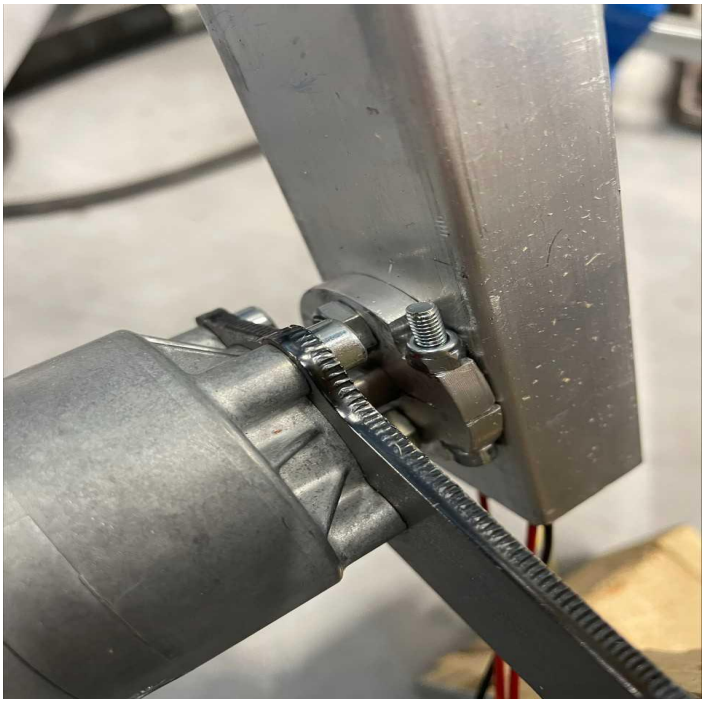


Figure 13. Motor attachment to the gate.

For the wires, a hole was made in the boom, from which it was possible to place the wires neatly inside the boom. The Arduino housing was placed at the end of the gate. (Figure 14.)

Figure 14. Mounting the arduino housing.

The H-bridge housing was attached to the engine mounting plate near the motor.

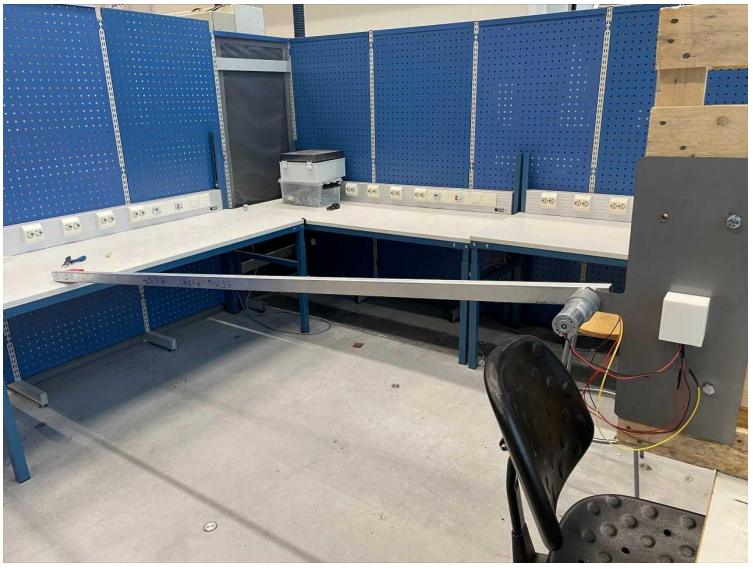
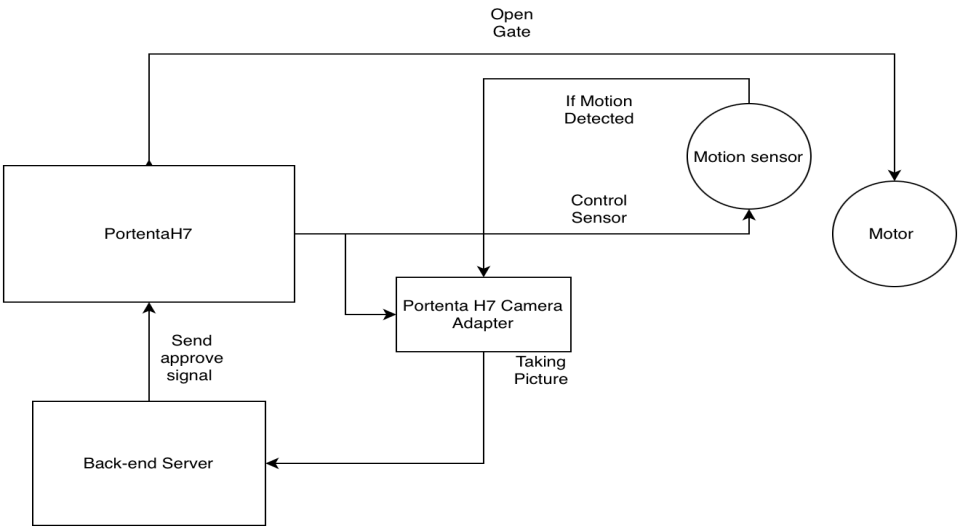


Figure 15. H-bridge housing and gate assembly.

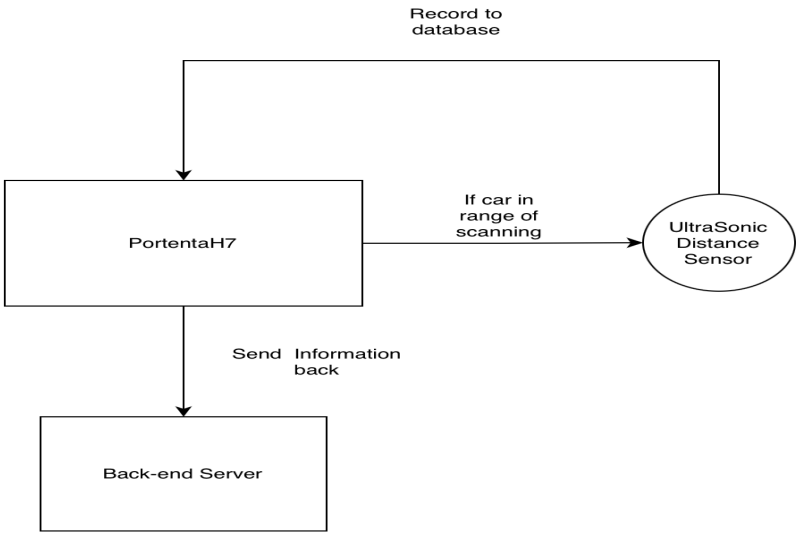
5 Functional Description and Software

5.1 Block Diagram

For entrance:



For exiting:



5.2 Software

There are several possible options of programming languages, platforms, and frameworks for programming on Arduino Portenta H7. The most common possibilities were carefully considered, and we chose Micropython, OpenMV IDE to control the microcontroller and NodeJS for back-end server. Those are the criteria to examine and choose the most suitable option.

First, MicroPython might be a good choice since it allows coding in Python 3 and provides a small subset of the Python standard library. Python is an easy and straightforward langue to work with. Compared to other programming languages, such as C++, less code in Python is needed for the same task to be done. Despite the ease, there are some limitations when it comes to components. Some sensors in this project do not have existing code in Python but have higher chances in C/C++. But till the end, we have already figured out to connect sensors to Arduino via Micropython without any errors. Additionally, MicroPython supports some libraries that can help us to connect to our back-end server. Therefore, we decided to use Micropython

The second promising option is OpenMV IDE since there are tons of tutorials online and ready-to-go code. However, it requires to familiarity with programming and configuring Arduino to get full libraries for it, and the team has very few practices in this aspect. Besides, its programming language is MicroPython, similar to Python, more widely used in reallife applications, a large user community so we can fix a lot of errors without obstacles. Consequently, Arduino IDE is not favorable.

Lastly, we will use a ready-made back-end server via NodeJS

5.3 Connectivity

Arduino Portenta H7 has a built-in wifi function and Ethernet port so depending on different usages, we can choose any kind of connection in order to optimize speed or focus on stabilization.

```
import network #Import library
sta_if = network.WLAN(network.STA_IF)


def connectWifi():
    if sta_if.isconnected() == False:
        sta_if.active(True)
        sta_if.connect('AIoT-Garage', 'asdfghjk') #Enter SSID and Wifi Password
        print('connection is established for the first time: ' + str(sta_if.isconnected())) #Print if wifi is connected
    else:
        print('Wifi already connected')
```

Figure 16. Wifi Initialization

From the code above, we can change SSID and wifi password for Arduino. The Wi-Fi connection will be automatically checked hourly to ensure that Arduino always stays connected.

5.3 Controlling Motor

We are using two motors for this project; the smaller ones will use in order to optimize the script, easy to carry and the bigger ones will be used when install to the barrier before showing them to the customer.

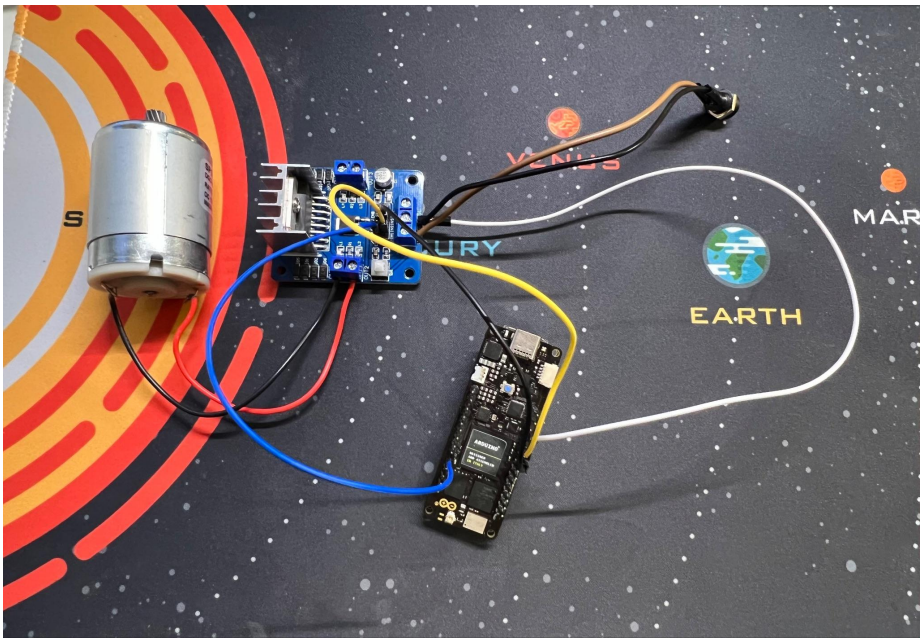


Figure 17. Connect Motor to Arduino

```
import pyb
from machine import Pin
from time import sleep

IN1 = Pin('PA9', Pin.OUT) #High voltage #Initial high voltage pin
IN2 = Pin('PA10', Pin.OUT)  #Low voltage #Initial low voltage pin
ENA = Pin('PK1', Pin.OUT) #Initial speed pin
```

Figure 18. Initialing Motor code

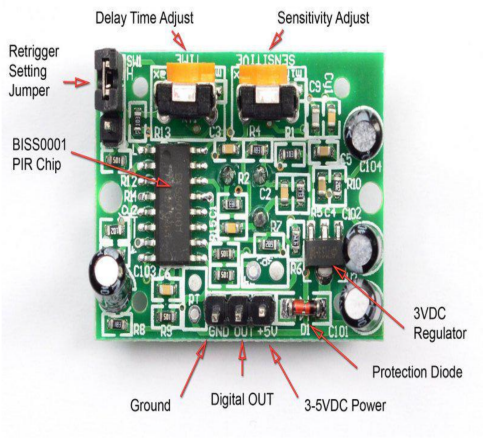
The usage of the motor to control the opening and closing gate so we will install three pins from the motor to Arduino via H-Bridge. 2 Pins, IN1 and IN2, will be connected to force motor to open or close. Otherwise, “ENA” pin will control low or high voltage in order to adjust motor speed. This works with a simple DC motor that uses 24 volts to operate. In addition, the team wants to provide the user with the ability to turn on and off the boom when needed. Therefore, an Hbridge is added to the system. An Hbridge obtains the high and low signals from the controller and acts as a switch to control the Vin of the motor. Thus, the motor can be turned on and off as commanded or set time delay by using “Time.Sleep(x)”

```
def open(): #Motor rotate Left => Right
        print("Door is opening")
        IN1.high() #Rotate motor => Opening
        IN2.low()

def close(): #Motor rotates Right => Left
        print("Door is closing")
        IN1.low()
        IN2.high() #Closing

def stop(): #Turn off Motor
        IN1.low()
        IN2.low()
```

Figure 19. Motor Functions

5.4 Motion Detection

The motion sensor is attached next to the Arduino so that it would be easy to detect motion and take picture simultaneously. The sensor detects a human being moving around during approximately 10m from the sensor. It's an average value we can change by rotating the wheel on sensor, the detection range is from 3 to 12m. PIR sensors have a 3-pin connection at the side or bottom. One pin will be ground, another will be signal and the last pin will be power. Power is usually up to 5V



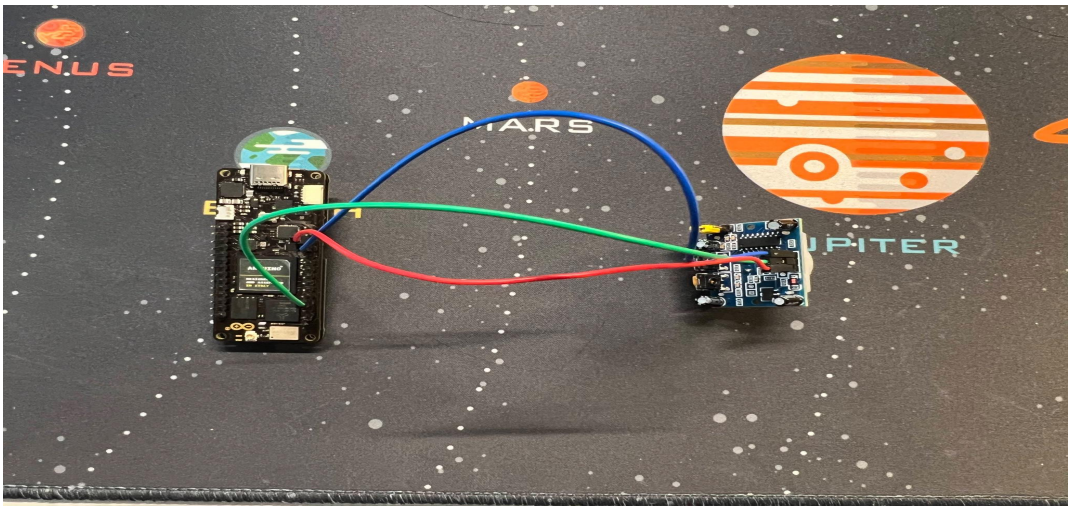


Figure 20. PIR Sensor

Figure 21. Setting up Motion Sensor to Arduino

We set the time delay to 0 so that the sensor would be as sensitive as possible and always pick up on moving objects without missing a beat on the road, where it is needed to automatically detect cars. In order to connect a sensor to an Arduino, we need only use three wires: one for ground and one for power, last one will connect to send and receive data.

```
from machine import Pin

PIR1 = Pin('PC3', Pin.IN)  #Initial Motion Sensor
```

Figure 22. Initial PIR Sensor and assign its Pin to Arduino

```
def motion_detected(): #Motion Function
    if PIR1.value() == 1:
        print("Motion Detected")
        return True
    else:
        print("No motion")
        return False
```

Figure 23. PIR Sensor script

5.5 Ultrasonic sensor

When a vehicle gets within a certain distance of this sensor, the system starts recording the time the vehicle left the parking lot. This sensor can determine both the vehicle's location and its distance from the sensor then record to database

```
#include "Ultrasonic.h"

Ultrasonic ultrasonic(7);
void setup()
{
    Serial.begin(9600);
}
void loop()
{
    long RangeInInches;
    long RangeInCentimeters;

    Serial.println("The distance to obstacles in front is: ");
    RangeInInches = ultrasonic.MeasureInInches();
    Serial.print(RangeInInches);//0~157 inches
    Serial.println(" inch");
    delay(250);

    RangeInCentimeters = ultrasonic.MeasureInCentimeters(); // two measurements should keep an
    Serial.print(RangeInCentimeters);//0~400cm
    Serial.println(" cm");
    delay(250);
}
```

Figure 24. Ultrasonic script

5.5 Arduino Camera

The Himax HM-01B0 camera on the Vision Shield has a monochrome 320x320 sensor that supports HQVGA resolution and is capturing at 320x240, in the beginning, The image is cropped to 240x160 to reduce size and optimize storage. The scaled image is sent to the image server, so the image that you get is pretty low resolution - but it works for detecting car plates and the image quality is enough for sever to recognize numbers and words.

```
import sensor, image, os, time

sensor.reset()                      # Reset and initialize the sensor.
sensor.set_pixformat(sensor.GRAYSCALE) # Set pixel format to RGB565 (or GRAYSCALE)
sensor.set_framesize(sensor.HQVGA)   # Set frame size to HQVGA (240x160)
sensor.skip_frames(time = 2000)     # Wait for settings take effect.
```

Figure 25. Setting up Camera

It saves the image as the most recent ".bmp" file after capturing it. Because the original Arduino has limited storage, every image saved overwrites the previous one. In the following stage, we set a delay time of 2.5 seconds for Arduino to process the image before sending it to the server. If the 2.5-second delay is not set, the processor may become overloaded and an error may occur.

```
def captureImage():
    i = sensor.snapshot()
    frameBuffer = sensor.get_fb()
    frameBuffer.save(os.getcwd()+'/latest.bmp') #save images as the lastest file
    time.sleep(2.5) #Wait 2.5 seconds before send confirmation
    print('Image captured and saved')
    return # Make sure the image is saved and then continue
```

Figure 26. Camera function

5.5 Database

The database that is being used is Google's Firestore, which uses the NoSQL data storage format. We decided to choose Google platform because it is light, stable, and good for beginners. The NoSQL database is a document-based system. The database is given the instruction to store the data that has been assigned to it, which in this instance takes the form of a license plate. The database is responsible for storing the timestamp of the event as well as the license plate information of the vehicle that was driving in front of the boom. The data that is already available can be used for a mobile application that is being developed or for any other application that has the capability to display data from single or multiple distinct chains of events.

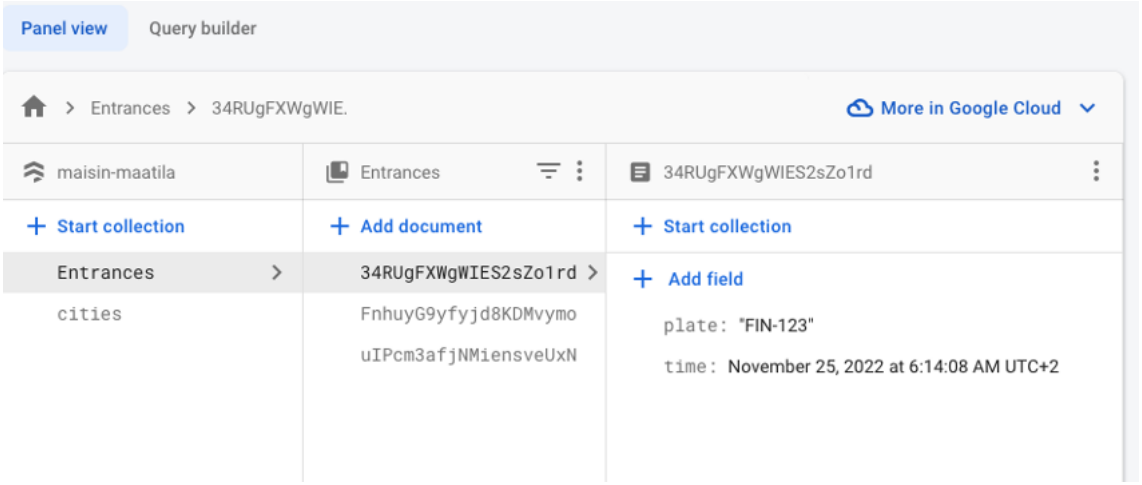


Figure 27. Overview of Google Firestone and how it work to recognize car plate

We used pre-existing scripts from one of the previous Big Flash projects to send and receive license plate data to and from a server. The primary goal of the script is to utilize Machine Vision and AI, Tensor Flows to read text from a surface or other material. Fuwad, one of our team members, contributed scripts to the effort, and these scripts have been tailored to work with our in-house database.

```python
def make_request(data, image=None):
    boundary = '---011000010111000001101001'
    #boundary fixed instead of generating new one everytime

    def encode_field(field_name): # prepares lines that include chat_id
        return (
            b'--%s' % boundary,
            b'Content-Disposition: form-data; name="%s"' % field_name,
            b'',
            b'%s'% data[field_name] #field_name conatains chat_id
        )


    def encode_file(field_name):  # prepares lines for the file
        filename = 'latest.bmp'  # dummy name is assigned to uploaded file
        return (
            b'--%s' % boundary,
            b'Content-Disposition: form-data; name="%s"; filename="%s"' % (
                field_name, filename),
            b'',
            image
        )

    lines = [] # empty array initiated
    for name in data:
        lines.extend(encode_field(name)) # adding lines (data)
    if image:
        lines.extend(encode_file('image')) # adding lines  image
    lines.extend((b'--%s--' % boundary, b'')) # ending  with boundary

    body = b'\r\n'.join(lines) # joining all lines constitues body
    body = body + b'\r\n' # extra addtion at the end of file

    headers = {
        'content-type': 'multipart/form-data; boundary=' + boundary
        }  # removed content length parameter
    return body, headers  # body contains the assembled upload package
```

```python
def send_my_photo(photo_pathstring): #  path and filename combined
    #token = 'authentication token or other data' # this my bot token
    chat_id= 999999999 # my chat_id
    #url = 'http://192.168.1.115:3000' + token
    url = 'http://192.168.0.53:3000'
    path = photo_pathstring   # this is the local path
    myphoto = open(path , 'rb') #myphoto is the photo to send
    myphoto_data = myphoto.read() # generate file in bytes
    data = { 'timestamp' : 999999999 }
    body, headers  = make_request(data, myphoto_data) # generate body to upload
    url = url + '/upload'
    headers = { 'content-type': "multipart/form-data; boundary=---011000010111000001101001" }
    response = upload_image(url, headers, body) # using function to upload to telegram
    return response
```

Figure 28,29. Demonstrate how Arduino call request to send image and sending command

```
def upload_image(url, headers, data):
    http_response = urequests.post(
        url,
        headers=headers,
        data=data
    )
    print(http_response.status_code) # response status code is the output for request made

    if (http_response.status_code == 204 or http_response.status_code == 200):
        print('Uploaded request')
    else:
        print('cant upload')
        #raise UploadError(http_response) line commneted out
    #http_response.close()
    return http_response
```

Figure 30. Demonstrate how an image is uploaded to server

5.6 Compiling

The Arduino's memory stores all of its previously used functions in individual files. We need to isolate each component so that we can make changes or correct errors without disrupting the operation of the others. The Arduino IDE and PC are not required for this automatic operation because a while-loop has been set up. Getting power to the Arduino only requires a USB-C cable.

```
#This main file will compile everything together and make Arduino run independance without IDE
from connect_wifi import connectWifi #Import Wifi function
from send_file import send_my_photo #Import Sendfile function
from capture import captureImage #import Capturing image function
from motion_check import motion_detected #Import motion detect function
import gate #Import gate function
from time import sleep
```

Figure 31. Importing functions into main file

```
while True:
    connectWifi() #Initial Wifi
    sleep(1) #wait 1 second after starting Wifi function
    if motion_detected(): #Turn on Motion sensor function, if it detected
        sleep(5) #Wait 5 seconds to make sure motion detect right
        captureImage() #Taking picture
        sleep(3) #Wait 3 seconds to process picture
        response = send_my_photo("latest.bmp") #Send picture to storage
        sleep(4)
        print(str(response.text)) #Send notification if picture saved successfully or re-take
        if response.text == 'true': #If plate same as database, return "True"
            print('open') #Print opening notificaiton
            gate.openUp() #Open gate by rotating motor
            sleep(5)
            gate.closeDown() #Closing after opening
        else:
            print('Do not open')
```

Figure 32. Main Executing File

In the loop, we'll delay the execution of the loop until all of the functions have finished processing and returned their results to the main file.

6 Problems

We continued testing the prototype in the last week before the report was returned, when the IT team completed the program. Got the boom to rise and the camera to take a picture of the license plate of the car. Problems will continue to be resolved even after the first return of the report.

Issues	Solutions
Remote working due to different schedule	Divide the works as much as possible. But in reality, the person holding the controller was the one to be able to properly develop at a time.

Only 1 GND pin on Arduino	We will need a divider for GND pin. After fixing, multiple sensors can work at the same time
Can not adjust motor speed. It runs at 100% power	Because the lack of libraries of Micropython, we can not change the speed of the motor. We found a solution by adding time delay for opening or closing
Difficulty when building a back-end server	Use a ready-made platform, open-source image processing and optimize it to run on our requirements, Ask
H-Bridge Overload	Carefully measuring voltage before changing from small to big motor

7 Conclusion

In conclusion, Maisin Maatila is a new solution for SMEs, small manufacturers in the breakthrough of technology. It meets all the requirements for optimizing the supply chain, time-saving, cost efficiency. Our solution creates a modern and more convenient way of delivering products to customers and focusing on user experience. This first phase of the project was successfully completed and the opportunities for future implementation is good. We are thankful to receive enthusiastic support and constructive comments from Big Flash's teachers, IoT assistants. The project can be carried on in the Innovation Project course.

Sources

Marcelo Rovai. Mug, or not Mug, that is the question! Arduino assembly Available from: <https://www.wikimakes.com/view-maker-hub-project/1293> (cited November 26 2022)

Arduino H7. Arduino official store. Available from: <https://store.arduino.cc/products/portenta-h7> (cited November 25 2022)

Portenta vision shield-Ethernet. Arduino official store. Available from: <https://store.arduino.cc/products/arduino-portenta-vision-shield-ethernet> (cited November 20 .2022)

Grove ultrasonic distance sensor V2.0. Digi-key electronics. Available from: <https://www.digikey.fi/fi/products/detail/seeed-technology-co.,-ltd/101020010/5482600?utm_adgroup=General&utm_source=google&utm_medium=cpc&utm_campaign=PMax%20Shopping_Product_Zombie%20SKU&utm_term=&productid=5482600&gclid=Cj0KCQiAj4ecBhD3ARIsAM4Q_jEH6dnkAmp7ZaE_ilFR7GbNWSa4SW5RIUaCcChOs3bL0WqlQc_SjHYaAuqbEALw_wcB> (cited November 19 .2022)

L298 H-Bridge Motor control board. PARTCO The Electronics Shop. Available from: <https://www.partco.fi/fi/robotit/robottielektroniikkaa/19140-4tx-l298n.html> (cited November 16 2022)