

Parkingplace\_group\_9

Michiel Janssens, Viktor Nagels, Emil Dudayev, Bent Melis, Vincent Somers

Contents

[Preface 3](#_Toc89162127)

[Team 3](#_Toc89162128)

[Electrical 4](#_Toc89162129)

[Parts 5](#_Toc89162130)

[Code 5](#_Toc89162131)

[Code explained 5](#_Toc89162132)

[Full code 5](#_Toc89162133)

# Preface

The objective of this project was to automate a parking with number plate recognition and visualization it for the users and owner. We did this on a small scale using a raspberry pi and a small scale car park.

# Team

Our team exist out of 5 members, each one had their own task but off course a bit of teamwork was necessary. Emil Dudayev helped with the recognition of the numberplate and helped with the documentation. Viktor was responsible for the recognition of the numberplate and also helped making the documentation. Vincent Somers was responsible for the Correct visualization on display with data from the database. Bent Melis was responsible for the correct working remote web interface and helped with the push message. And at last Michiel Janssens was responsible for the code for the barrier with ultrasonic sensor and helped with the push message.

A picture containing person, person, outdoor, shirt

Description automatically generated

A person wearing glasses

Description automatically generatedA picture containing outdoor, person, snow, child

Description automatically generated

# Electrical

Diagram

Description automatically generated with medium confidence

Diagram, schematic

Description automatically generated

# Parts

A picture containing diagram

Description automatically generated

# Code

## Code explained

## Full code

from picamera import PiCamera

from time import sleep

import requests

from pprint import pprint #? nakijken nog nodig

import RPi.GPIO as GPIO

import time

ultrasonic1 = 20

ultrasonic2 = 21

button = 12

step1 = 6

step2 = 13

step3 = 19

step4 = 26

GPIO.setmode(GPIO.BCM)

GPIO.setup(ultrasonic2,button, GPIO.IN)

GPIO.setup((ultrasonic1,step1,step2,step3,step4), GPIO.OUT)

def stepdrive(pin1,pin2,pin3,pin4):

    pinnumbers = pin1,pin2,pin3,pin4

    GPIO.setup(pinnumbers, GPIO.OUT)

    GPIO.output(pinnumbers, 0)

    GPIO.output(pin1, 1)

    GPIO.output(pin2, 1)

    GPIO.output(pin3, 0)

    GPIO.output(pin4, 0)

    time.sleep(0.01)

    GPIO.output(pin1, 0)

    GPIO.output(pin2, 1)

    GPIO.output(pin3, 1)

    GPIO.output(pin4, 0)

    time.sleep(0.01)

    GPIO.output(pin1, 0)

    GPIO.output(pin2, 0)

    GPIO.output(pin3, 1)

    GPIO.output(pin4, 1)

    time.sleep(0.01)

    GPIO.output(pin1, 1)

    GPIO.output(pin2, 0)

    GPIO.output(pin3, 0)

    GPIO.output(pin4, 1)

    time.sleep(0.01)

def photo():

    camera = PiCamera() # TODO deze rotatie kan nog aangepast worden A.D.H.V. hoe de camera geposisioneert staat

    #camera.rotation = 0

    camera.start\_preview()

    sleep(2)

    camera.capture('/home/pi/images/photo.jpg')  # TODO make location for the pictures

    camera.stop\_preview()

def numberplate():

    regions = ['mx', 'be'] # Change to your country

    with open('/home/pi/images/photo.jpg', 'rb') as fp:

        response = requests.post(

            'https://api.platerecognizer.com/v1/plate-reader/',

            data=dict(regions=regions),  # Optional

            files=dict(upload=fp),

            headers={'Authorization': 'Token 315a9c42be797329049bf2cc52a5cb41ab960e15'})

    json\_results = (response.json())

    if (response.json()['results'] == []):

        numberplate = "FALSE"

    else:

        numberplate = (json\_results['results'][0]['plate'])

    return numberplate

def stepmotor():

    for n in range(0, 130):

            stepdrive(step1,step2,step3,step4)

    time.sleep(10)   #TODO moet nog veranderd worden naar als de auto weg is

    for n in range(0, 130):

            stepdrive(step4,step3,step2,step1)

def ultrasonic():

    GPIO.output(ultrasonic1,1)

    time.sleep(0.00001)

    GPIO.output(ultrasonic1,0)

    while(GPIO.input(ultrasonic2)==0):

        pass

    signaalhigh = time.time()

    while(GPIO.input(ultrasonic2)==1):

        pass

    signaallow = time.time()

    timepassed = signaallow - signaalhigh

    distance = timepassed \* 17000

    return distance

while True:

    distance = ultrasonic()

    print(distance)

    if distance <= 30 :    #in centimeter

        photo()

        sleep(5)

        numberplate()

        numberplate = numberplate()

        print(numberplate)

        if numberplate != "FALSE":

            print(numberplate)

            print("Car can access parking")

            stepmotor()

        else:

            print("There was no numberplate found")

    else:

        #! 111 - 114: Zijn deze verplicht?

        GPIO.output(step1, 0)

        GPIO.output(step2, 0)

        GPIO.output(step3, 0)

        GPIO.output(step4, 0)

        print('GEEN auto aan de bareel')

        time.sleep(0.5)

    #? Code for exiting the parking

    if GPIO.input(button) == GPIO.HIGH:

        print("Button was pushed!")

        photo()

        time.sleep(3)

        numberplate()

        if numberplate == "FALSE":

            print("car can exit parking")

            stepmotor()

        else:

            print("Someone is tryint to bypass the system!!!")