NEW YORK CITY COLLEGE OF TECHNOLOGY

Final Project

Security drop-off box

CET 4925-OL62

Group members

Sheldon Waite	SM Hossain	Tyrik Emptage
ID: 23539966	ID: 23498853	ID: 23653573

Department: Computer Engineering Technology

Semester/year: Spring 2021

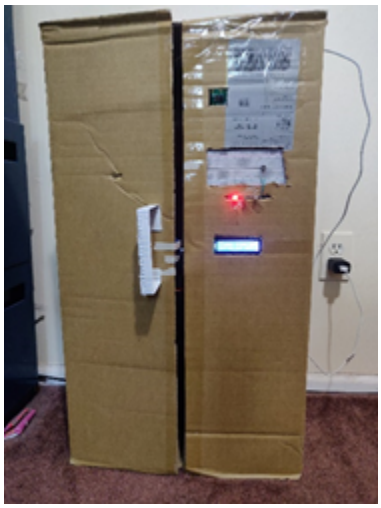
Report submission date: May 26, 2021

Table of Contents

Content ***Page***

Project Setup - Detailed Description

A secured delivery of packages has been a challenging task for couriers and the receivers alike and it was stated that in the year 2020 almost 1.7 million packages were stolen or lost every day in the U.S **[1]**.



Project Management

CET_4925 IoT
Project Management Plan
Semester: Spring 2021

Project Name: Security Drop-Off Box

Group Member Names:
A (SM Hossain)
B (Tyrik Emptage)
C (Sheldon Waite)

Task / Sub task	Person Responsible			Week														
	A	B	C	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1. Mechanical Design																		
a. Box	A																	
2. Electrical Design (Sensor & Motor Interface)	A		C															
a. LCD Display	A		C															
b. Camera Module	A		C															
c, Push button	A																	
d. Servo Motor	A																	
3. Software Design (Control Program)	A	B	C															
4. Testing and Debugging	A	B	C															
5. Project Demonstration	A																	
6. Project Report	A	B	C															

Color Code:
Work completed
Work remaining

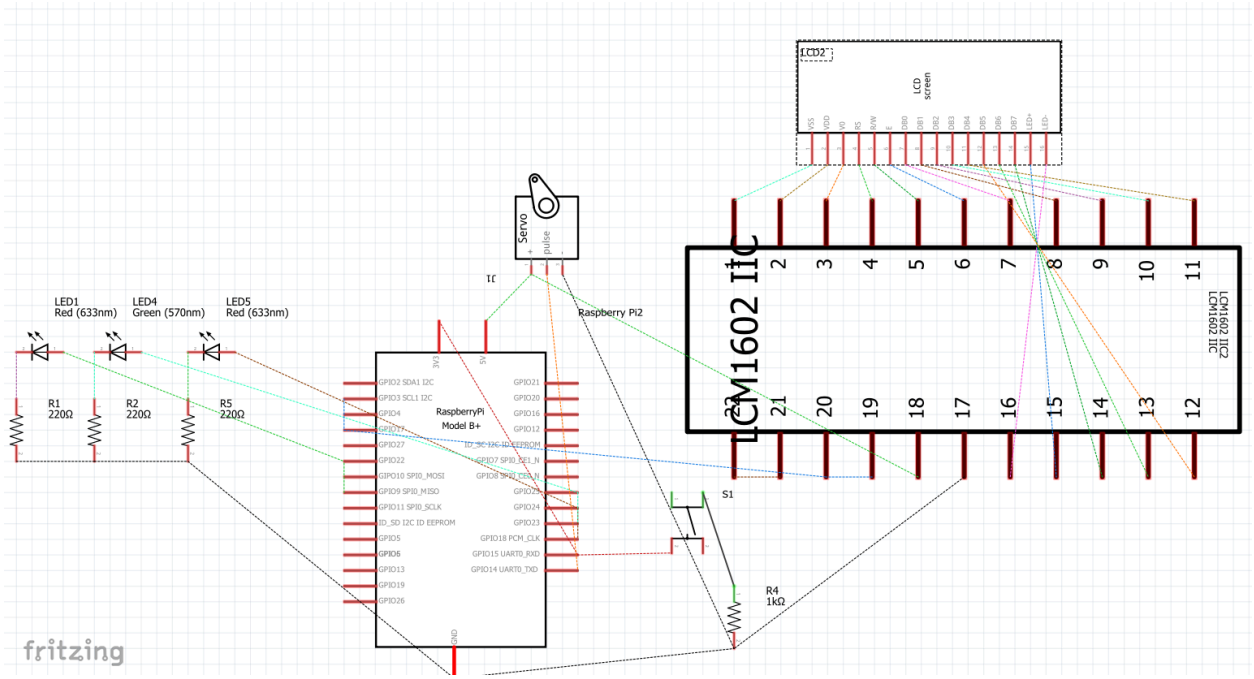
System Design

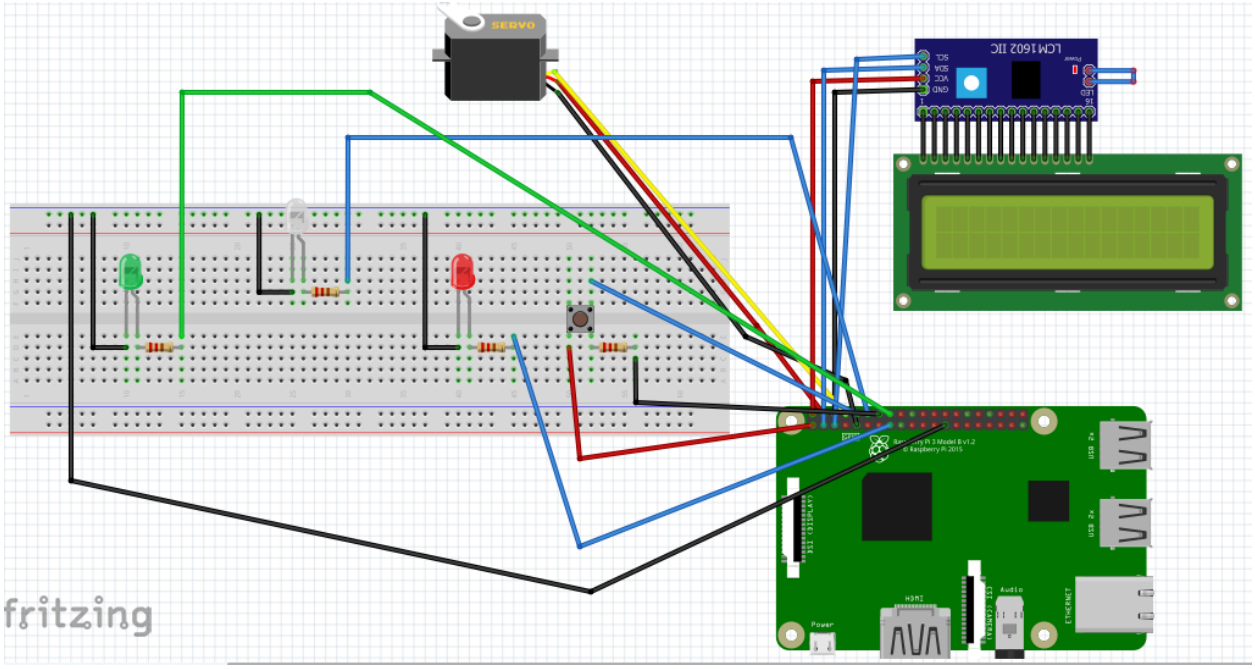
For our Final project, we decided to create a Security drop-off box, which can ensure the safety and security of these packages. The general component that we used is a raspberry pi 3 B+ with a pi cam. We implemented a push button and a couple of LEDs in the design for the delivery guy to press and visualize the processes taking place. We set it up in a way so that when a button is pressed, the users will get Multimedia message on their phone along with the picture of the courier. We created an android app using MIT app inventor to remotely unlock/lock the door. We will have the LCD screen provide interactions and step by step instructions to the courier for a guaranteed delivery

In order to build a security drop off-box, we replicated the structure by using cardboard having the dimensions 29x10.25x16 with an opening on one side as shown on the figure above. We had to cut out some parts of the cardboard to build the housing for subsystems like LCD screen, pi camera, breadboard and a couple of LEDs, these were the only components that needed to stick out of the board, however the main component i.e. the raspberry pi 3 B+ was inside the cardboard meshed firmly. A small hole was made on the cardboard to run a 2.5A micro-USB power cable through to power up the pi. One half of the open side of the cardboard was folded to create a hinge and a servo motor was attached to it.

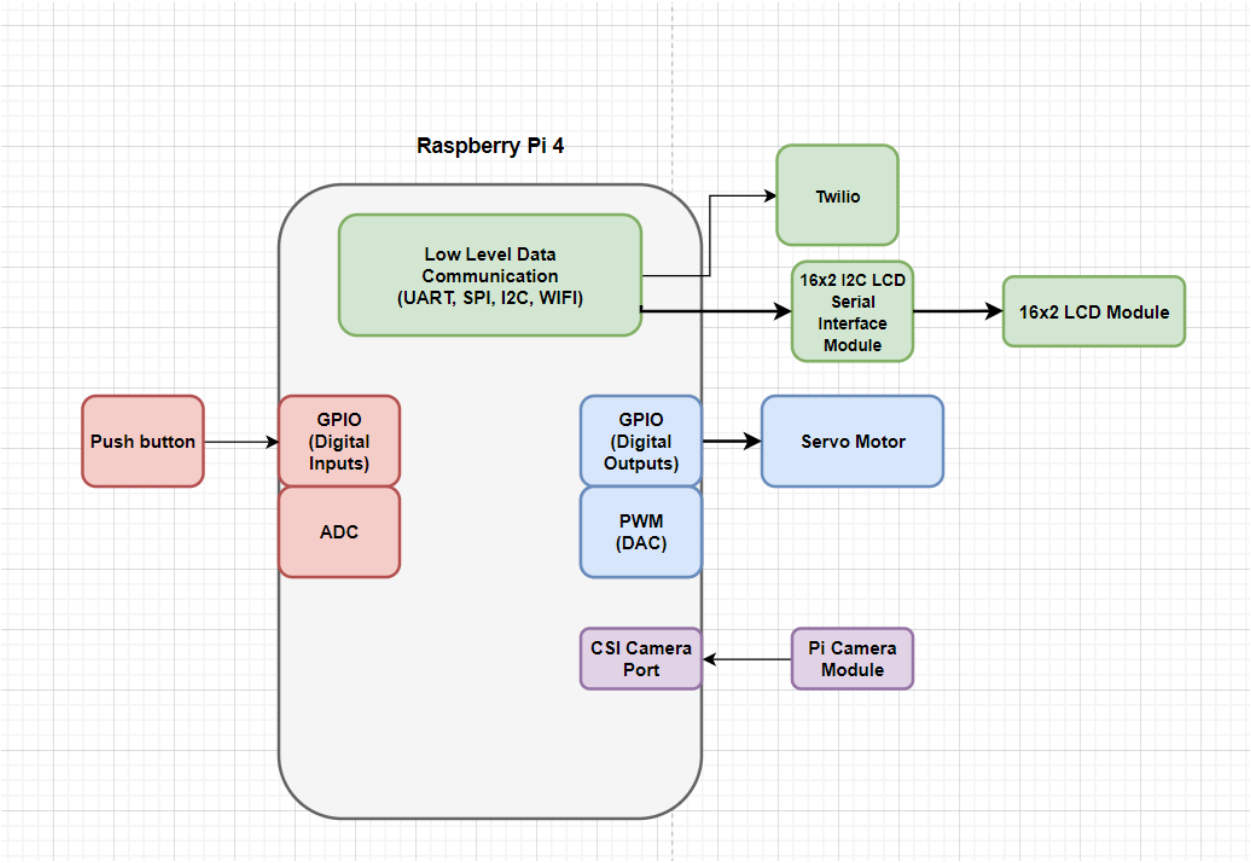
To send a picture or an image we decided to use Twilio which is an American cloud communication PaaS (platform as a service). We had to create an account and include the credentials that we obtained from our account to link the service platform with our raspberry pi so that when a script is executed communication will be established between twilio and pi and we can get notifications in the form of SMS/MMS as a result **[2]**.

Schematic Diagram



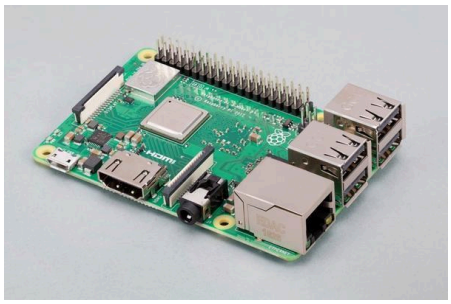


Block Diagram



Component design

- Raspberry Pi 3 B+



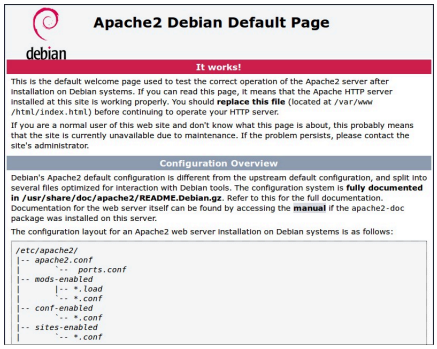
For the brain of our system, we used this powerful component called the Raspberry Pi as shown above, it is a single board computer with the specifications: 1.4GHz 64-bit quad-core processor (Arm Cortex A-53), dual-band wireless LAN, Bluetooth 4.2/BLE, faster Ethernet, and Power-over-Ethernet support **[3]**. In order to operate this device we had to install an operating system called "raspbian" by flashing the image file on a micro sd card and booting the files from this card on start-up. We used SSH to remotely connect to the terminal of our pi device after having it connected to the same wifi as the host computer. We enabled features like pi camera and VNC server so that we can remotely

utilize the GUI. Since we will be setting a up a server on this device, we had to use the following commands to set up the apache server from the terminal:

`sudo apt update` (for updating existing software and tools)

`sudo apt install apache2 -y`

One way of confirming that an apache web server has been set up is by navigating to the /var/www/html directory and finding an index.html file. Now we can use the pi's IP address to directly access the webpage or the contents of the folder (html) **[4]**. The figure shows the webpage that will display once we log the device's ip address.



Since the html file or the entire html directory is owned by the "root" user, we had to change the permission/ownership by using the following command: (assuming that the default username is "pi")

`sudo chown pi: index.html` now the regular user can edit the directory or files inside the directory however they find it convenient.

One drawback of twilio PaaS is that it cannot access any private/protected network or webserver. So, one solution to have twilio communicate with the apache server in our raspberry pi is to forward the port that is associated with the protocol that the apache server uses. Since the web server is HTTP based, we needed to forward port 80 for the pi device from the firewall of our router. This will enable Twilio to extract multimedia files from the web server and send it to the user's mobile device as an MMS. A screenshot of the port forwarding is attached below

- Raspberry Pi Camera

The camera that we used to take pictures is an 8mp camera module called the picam V2. The camera is capable of taking 3280 x 2646 pixel pictures and recording 1080p video. Ity can simply be connected to the CSI (Camera Serial Interface) of the pi device using a ribbon cable **[5]**. We tested the pi cam by running simple commands on the terminal like `Raspistill -o /var/www/html/img.jpg`**.** This will save the picture on the directory "html" due to the pathway included in the command. One downside to this camera is that it won't work in absence of visible light unlike the NoIR camera Module.

- Servo motor

A gear train is implemented inside the servo and the purpose of these gears is to control/reduce the speed of the motion. The speed is reduced by meshing a larger gear with the output shaft and increased by meshing a smaller gear with the shaft of the servo. Similarly, the torque can be increased by implementing a gear train and the speed is inversely proportional to the torque in a gear train.

The potentiometer in the servo motor is attached to the shaft of the motor and acts as a feedback device that sends signals to the controlled circuitry in the form of voltage by changing the resistance. The control circuitry then monitors the current angle of the servo motor and checks whether it's in the right position. The angle of the shaft is proportional to the speed of the motor which is determined by how much voltage/power is allowed to the circuit by the potentiometer.

Servo motors are composed of 3 wires. One goes to Vin, the second goes to ground and the third wire goes to the pin in the digital interface of photon. The servo expects a pulse every 20ms~50Hz. The position of the shaft of the servo motor corresponds to the pulse width. A pulse width of 1ms will yield a position of 0° and a pulse width of 2ms will yield a position of 180° of the output shaft. The conversion of the signal is carried in the servo library that we included in our source code. The standard voltage to run a servo motor is 4.8 V DC, however 6 V and 12 V is also used on few servos.

The needle of the servo will act as the locking/unlocking mechanism, it will unlock the door at 90° angle and lock at 0°/180°.

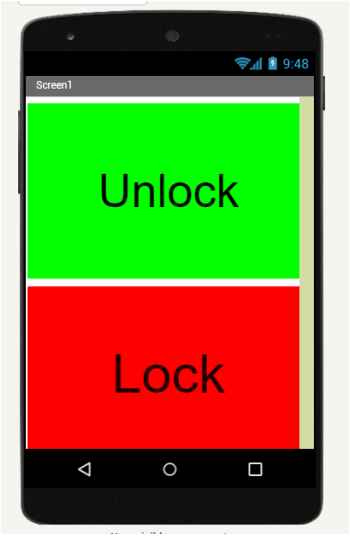
- LCD display with I2C module

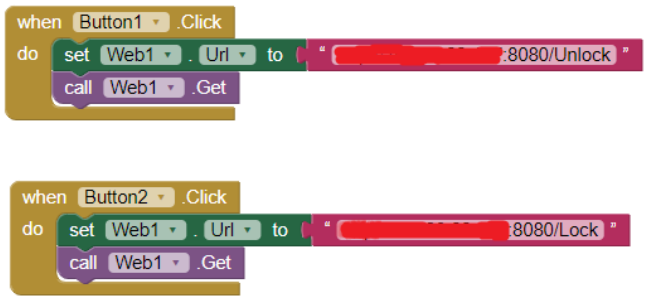
The lcd display has 16 pins. The display is powered by 5V. The Vo pin is attached to a potentiometer for controlling the contrast of the display. Next, the RS pin or register select pin is used for selecting whether we will send commands or data to the LCD. For example if the RS pin is set on low state or zero volts, then we are sending commands to the LCD. These commands are setting the cursor to a specific location, clearing the display, turning off the display, etc.

At the heart of the I2C adapter is an 8-Bit I/O Expander chip – PCF8574. This chip converts the I2C data from an Arduino into the parallel data required by the LCD display.In addition, there is a jumper on the board that supplies power to the backlight. To control the intensity of the backlight, you can remove the jumper and apply an external voltage to the header pin that is marked as ‘LED’ . To connect the I2C LCD to Arduino, connect 4 pins instead of 12. Start by connecting the VIN pin to the 5V output on the Arduino and connect GND to ground.On the Arduino boards with the R3 layout, the SDA (data line) and SCL (clock line) are on the pin headers close to the AREF pin. They are also known as A5 (SCL) and A4 (SDA).

- Phone Screen (MIT App Inventor)

For controlling the lock (servo) we created an app on MIT app inventor for an android device. The procedure is fairly straightforward, for its functionality we decided to put two buttons as shown below, by using blocks such as button logic block, call-function blocks and text blocks. Having done with the blocks we incorporated it's operability to the buttons in the GUI.





- Indicator LEDs (Quantity used: 3)

The idea behind these components is for three LEDs to serve a purpose towards activating the servo motor and pi camera. The white LED functions as a flashlight when the push button is pressed. This indicates that the pi camera has taken a photo of what was in its peripheral view. The red LED functions as an indicator that the door is locked. This can be activated through the Security_Dropbox app created on MIT App Inventor in which you press

the “Lock” feature on your phone screen. The needle of the servo indicates that the door is unlocked when turned at a 90° angle. The green LED functions as an indicator that the door is unlocked. This can be activated through the Security_Dropbox app created on MIT App Inventor in which you press the “Unlock” feature on your phone screen. The needle of the servo indicates that the door is locked when turned at a 0°/180° angle.

- Push Button

The pushbutton is a component that connects two points in a circuit when you press it. The function of this component is to activate the Raspberry pi camera photo capabilities when you press on the button. As you press the push button, the pi camera takes a photo and that image gets sent to your phone via MMS text messaging.

- Cardboard box

The setup of the cardboard box is made to represent a door that can hold packages in a secure place for delivery and the box can be unlocked/locked through the MIT App Inventor.

Testing Procedure & Analysis

- Setting up the Twilio account for MMS and using a test code to send a text message to the user’s phone for confirming the connection. For this, port forwarding was not required since the text message can be typed into the “body” section of the source code as a string. The 4 credential necessary for the communication are as follows:
  Account_sid, auth_token, Generated phone number by Twilio and user’s personal phone number.
  The most important part of the communication is the Twilio library that can be downloaded via linux terminal. The command is: `sudo pip3 install twilio`

- In a python script we needed to include the library for the camera and use the function camera = PiCamera() to enable the camera, camera.resolution to set up the resolution of the image that will be created and camera.capture to take a snap. We tested the camera using these functions and was successfully able to capture images

- In the python script we had to use the functions GPIO.PWM(8, 50) where 8 is the board PIN and 50 is the frequency at which the servo will operate. The min and max DutyCycle are 0 and 12.5 and each of these numeric values represent the different positions in our servo motor. Upon testing I found out that our servo shaft turned 90° at 5 and 180° at 10.

- Android APP: after constructing the response in HTTP server we used request handlers, the functions that handle the client request. We needed to open port 8080 to allow the communication thus we forwarded the port 8080 from our router. We used two LEDs initially to test whether both of the buttons are working and communicating.

Appendix (Source Code)

In order to run the source code(s), we had to use the command `sudo pyhton3 [1st script] & sudo python3 [2nd script]`. Similarly to stop the code from running, we can simply run the command `kill -9 [script IDs]` from the terminal of the raspberry pi. Raspberry Pi allows its user to run multiple scripts simultaneously so these processes can be run at the same time. It is also possible to combine the source codes into one script file and execute.

Script 1

```
import RPi.GPIO as GPIO # Import Raspberry Pi GPIO library
from time import sleep # Import the sleep function from the time module
import time
from picamera import PiCamera
from twilio.rest import Client

account_sid = "AC34931e498fb942fc020c6cf9e29da449"
auth_token = "b50aa98e9defac87d730279905315118"
client = Client(account_sid, auth_token)

GPIO.setwarnings(False) # Ignore warning for now
GPIO.setmode(GPIO.BOARD) # Use physical pin numbering
GPIO.setup(12, GPIO.OUT, initial=GPIO.LOW)# Set pin 8 to be an output pin and
set initial value to low (off)
GPIO.setup(10, GPIO.IN, pull_up_down=GPIO.PUD_UP) #Set pin for pushbutton
camera = PiCamera()
frame = 1

while True: # Run forever
    if GPIO.input(10) == GPIO.HIGH:
     GPIO.output(12, GPIO.HIGH)
     camera.resolution = (800, 600)
```

```
      camera.capture('/var/www/html/frame%03d.jpg' % frame)


      client.messages.create(
          to='+19174597938',
          from_='+12624445004',
          body='Delivery guy has arrived',
          media_url='http://ashrafpi.ddns.net/frame%03d.jpg' % frame
          )
      frame += 1
      print ('Button was Pushed, LED ON')# Turn on
     if GPIO.input(10) == GPIO.LOW:
       GPIO.output(12, GPIO.LOW)

GPIO.cleanup()
```

Script 2

```
import RPi.GPIO as GPIO
from http.server import BaseHTTPRequestHandler, HTTPServer
import time
import lcddriver
from time import *


GPIO.setwarnings(False)
mylcd = lcddriver.lcd()
GPIO.setmode(GPIO.BOARD)
GPIO.setup(15, GPIO.OUT, initial=GPIO.HIGH)
GPIO.setup(16, GPIO.OUT, initial=GPIO.LOW)
servoPIN = 8
GPIO.setup(servoPIN, GPIO.OUT)
p = GPIO.PWM(servoPIN, 50)
p.start(10)
mylcd.lcd_display_string("Please press the", 1)
mylcd.lcd_display_string("button and wait", 2)
Request = None

class RequestHandler_http(BaseHTTPRequestHandler):
  def do_GET(self):
    global Request
    messagetosend = bytes('To control the lock',"utf")
    self.send_response(200)
    self.send_header('Content-Type', 'text/plain')
    self.send_header('Content-Length', len(messagetosend))
    self.end_headers()
    self.wfile.write(messagetosend)
    Request = self.requestline
    Request = Request[5 : int(len(Request)-9)]
    print(Request)
    if Request == 'Unlock':
        p.ChangeDutyCycle(5)
        GPIO.output(16, GPIO.HIGH)
        GPIO.output(15, GPIO.LOW)
        mylcd.lcd_clear()
        mylcd.lcd_display_string("place package", 1)
```

```
        mylcd.lcd_display_string("and press button", 2)
        time.sleep(2)
    if Request == 'Lock':
        p.ChangeDutyCycle(10)
        GPIO.output(15, GPIO.HIGH)
        GPIO.output(16, GPIO.LOW)
        mylcd.lcd_clear()
        mylcd.lcd_display_string("Thank you!! have", 1)
        mylcd.lcd_display_string("a nice day!!", 2)
        sleep(5)
        mylcd.lcd_display_string("Please press the", 1)
        mylcd.lcd_display_string("button and wait", 2)
    return


server_address_http = ('192.168.1.189',8080)
http = HTTPServer(server_address_http, RequestHandler_http)
print('Starting Server')
http.serve_forever()
GPIO.cleanup()
```

References

1. https://www.newschannel5.com/news/report-1-7-million-packages-stolen-every-day
2. https://www.twilio.com/docs/sms/send-messages
3. https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/
4. https://www.raspberrypi.org/documentation/remote-access/web-server/apache.md
5. https://www.raspberrypi.org/products/camera-module-v2/

.