TECNOLOGICO NACIONAL DE MEXICO

INSTITUTO TECNOLOGICO DE MORELIA

"José María Morelos y Pavón"

Internet De Las Cosas

Profesor: Ferreira Escutia Rogelio

Proyecto Final

FACEMASK

Equipo 7

Castro Cazares Carlos Jahir

Vieyra Ernesto

Ingeniería en Sistemas Computacionales

Semestre Marzo-Julio 2021

07 de Julio de 2021

Contenido

Sobre el Proyecto

Objetivo

Brindar un control del tránsito de clientes en un local o negocio, para evitar la propagación del Covid-19.

Solución Propuesta

Hacer el uso de IA para la detección de cubrebocas en clientes que desean ingresar y el uso se sensores de temperatura y proximidad, para saber si este tiene una temperatura adecuada y si entro o salió del local. Mostrando la información a la administración.

Parte 1: Investigación

Estado del Arte

Cámara Termómetro De Reconocimiento Facial Avacom

Soluciones Inteligentes AVACOM para hacer frente a la propagación del COVID-19 en la etapa de reactivación económica de todos los sectores económicos: colegios, universidades, negocios, instituciones, hoteles, oficinas, clubes, conjuntos residenciales, edificaciones, etc.

Pantallas de control de acceso de medición de temperatura, que adoptan la plataforma de hardware de alto rendimiento de RSM micro rk3288 / rk3399 / Qualcomm msm8953, equipada con cámara binocular de grado industrial y tecnología de reconocimiento de rostros humanos, así como módulo de imagen térmica infrarroja. Admite reconocimiento de máscara facial, lector de tarjetas de identificación, instrumento de huellas digitales y otras extensiones periféricas. Se puede aplicar a la asistencia de control de acceso y canal de puerta para lograr un control de acceso seguro y eficiente del personal.

El terminal de medición de temperatura de reconocimiento facial adopta un algoritmo de inclinación profunda, logrando una alta velocidad y precisión de reconocimiento. YXD-F8 integrado con 2 millones de píxeles. Base de datos de 30,000 caras. La velocidad del pase de reconocimiento facial es inferior a 1 segundo. Admite reconocimiento facial y comparación precisos mientras usa una máscara. Por favor, consulte las soluciones de aplicación:

Cámara Termómetro De Reconocimiento Facial Avacom

https://www.avasoluciones.com/producto/camara-termometro-de-reconocimiento-facial-avacom/

Control de acceso con medición de temperatura, reconocimiento facial, de cubrebocas y casco

Con este control de acceso moderniza y automatiza el sistema de entrada en tu empresa o negocio y al mismo tiempo mantén la seguridad y protección de todos.

El sistema permite realizar configuraciones con diferentes parámetros para dar el acceso a las personas. Los parámetros disponibles son:

- Reconocimiento facial.
- Medición de temperatura
- Identificación de uso de cubrebocas.
- Identificación de uso de casco de seguridad.

Con ayuda de la interfaz del equipo, se pueden activar alarmas o mensajes que indican si alguno de los parámetros no está cumpliéndose y de esta forma negar el acceso.

El equipo está preparado para acoplarse a sistemas de apertura como chapas eléctricas o electromagnéticas, por lo que el acceso puede ser automático al cumplirse con los parámetros establecidos.

El sistema incorpora 2 cámaras de alta definición para identificar a los usuarios, tiene capacidad para almacenar hasta 10000 rostros en su base de datos y guardar hasta 8000 registros. Además, su tecnología de reconocimiento en 3D distingue entre una fotografía y una persona real, de esta manera, es inviolable y hace aún más seguro el control de acceso.

Su módulo para tomar la temperatura es de gran precisión (0,1 °C), con un rango de 30 °C a 45 °C. Al tomar la medición en la muñeca y ser infrarrojo, la temperatura se mide sin contacto, esto aumenta la higiene ya que nadie manipula el equipo.

Tiene conexión a la red por medio de Ethernet, para administrarlo desde su WEB app, ver el video en tiempo real y tener comunicación bidireccional.

Es ideal para usarlo en instalaciones como corporativos, escuelas, fábricas, instalaciones de gobierno, almacenes, etc.

cuando es configurado por usuarios o a la entrada en tiendas de autoservicio, departamentales o de servicios para verificación inmediata sanitaria.

https://www.steren.com.mx/control-de-acceso-con-medicion-de-temperatura-reconocimiento-facial-de-cubrebocas-y-casco.html

Hardware

En la siguiente tabla se encuentran los materiales que vemos que se necesitaran para poder crear el prototipo en físico, pensamos desde el raspberry, monitores, sensores, cables y otros materiales para armar la estructura que se necesitara; En esta misma se encuentran las tiendas tanto online o físicas para la compra de los mismos y los precios que se mostraron en el día de la consulta 30/06/2021, además de las cantidades de cada material, el total de cada uno, además de un pequeño colchón de dinero para imprevistos o tener que reponer materiales. Finalmente, el precio final del proyecto es de **$8,217.00 pesos.**

Material	Tienda	Presio	Cantidad	Total
Raspberry Pi4 4gb Case Ventilador Sd 32gb Hdmi Pi 4 B Kit	Mercado Libre	$2,692.00	1	$2,692.00
Monitor Hdmi De 14 Ips 1920x1080	Mercado Libre	$3,495.00	1	$3,495.00
Sensor Infrarrojo De Temperatura Gy-906 Mlx90614 I2c	Mercado Libre	$359.00	1	$359.00
Sensor De Presencia Pir Hcs501 Para Arduino Paq C/2pz	Mercado Libre	$119.00	2	$238.00
Camara Web Conferencia Usb Full Hd 1080p Pc Laptop	Mercado Libre	$298.00	1	$298.00
Dispensador Despachador De Jabón Y Gel Antibacterial 380ml	Mercado Libre	$139.00	1	$139.00
Pistola de silicon	Mercado Libre	$174.00	1	$174.00
Protoboard 830 Puntos Mb-102	DELTA	$52.00	1	$52.00
LEDS VARIOS COLORES	DELTA	$30.00	1	$30.00
RESISTENCIAS VARIAS	DELTA	$130.00	1	$130.00
CABLES PARA PROTOBOARD, 100	DELTA	$200.00	1	$200.00
CABLE UTP 4m	STEREN	$80.00	1	$80.00
Conectores RJ45	STEREN	$20.00	4	$80.00
Plataforma para prototipo de madera (Material y Mano de Obra)	CARPITERIA	$250.00	1	$250.00
Colchon para imprebistos	-	$500.00	1	$500.00
TOTAL				$8,217.00

Software

1) Plataforma
    a) Raspberry pi 4

2) Sistema Operativo
    a) Raspberry => Raspbian
    b) Servidor => Windows 10

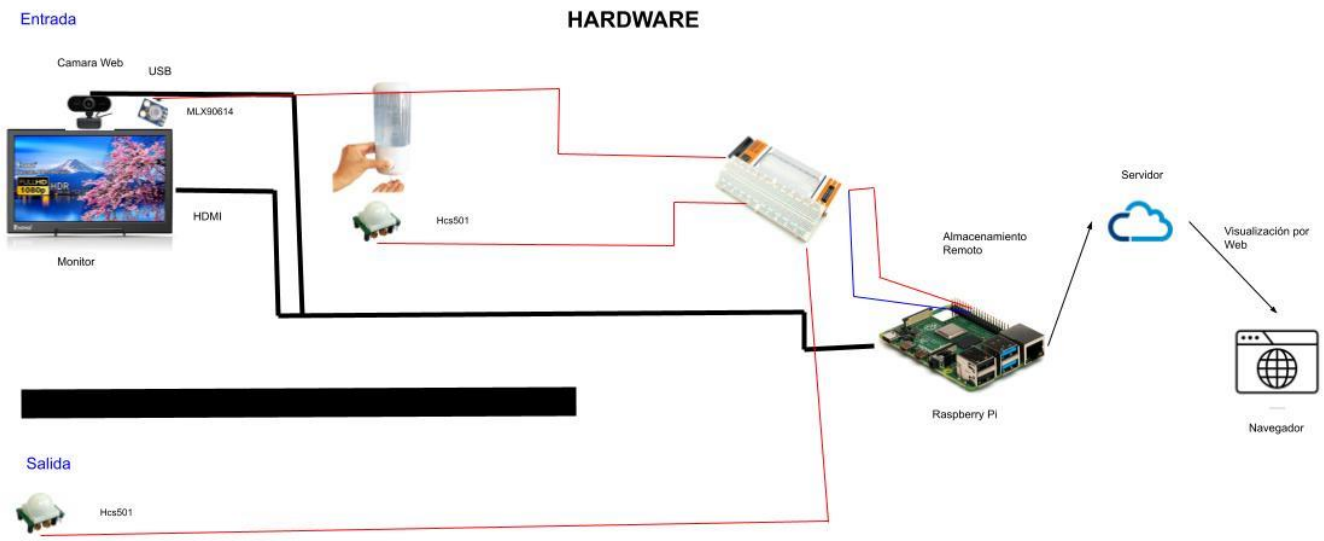
3) Lenguaje de Programación
    a) Python 3.8
    b) PHP 7.4.9
    c) HTML 5
    d) CSS 3
    e) JavaScript

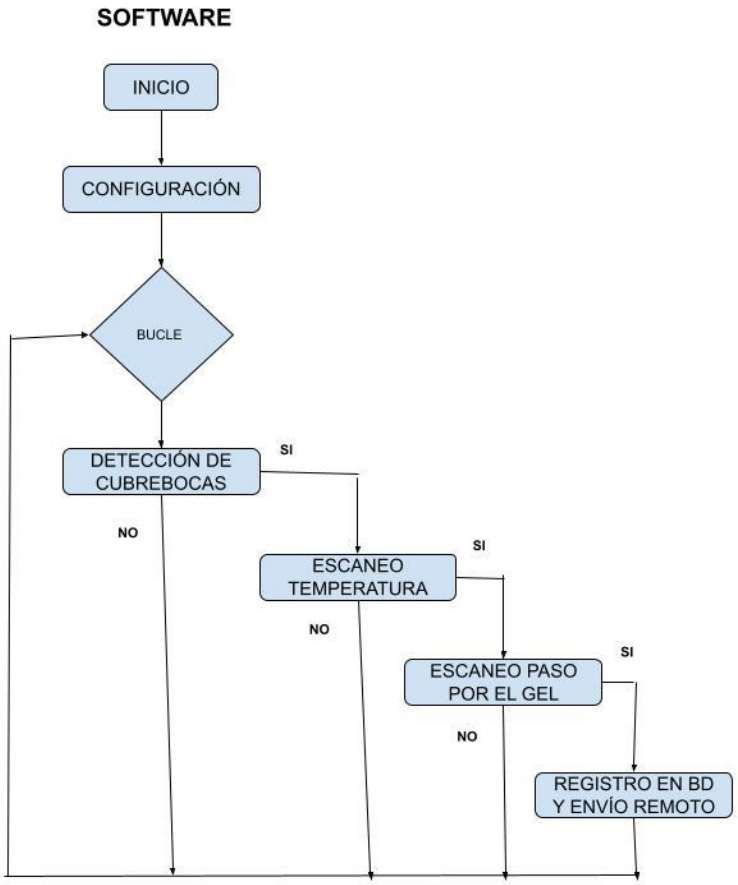
4) Librerías
    a) board 1.0
    b) busio
    c) adafruit-circuitpython-mlx90614 1.2.6
    d) pyA20
    e) Cv2
    f) Mediapipe
    g) MySQL
    h) requests
    i) Bootstrap
    j) JQuery
    k) Google chart

5) Otros
    a) MariaDB
    b) Servidor HTTP Apache
    c) Navegador Web

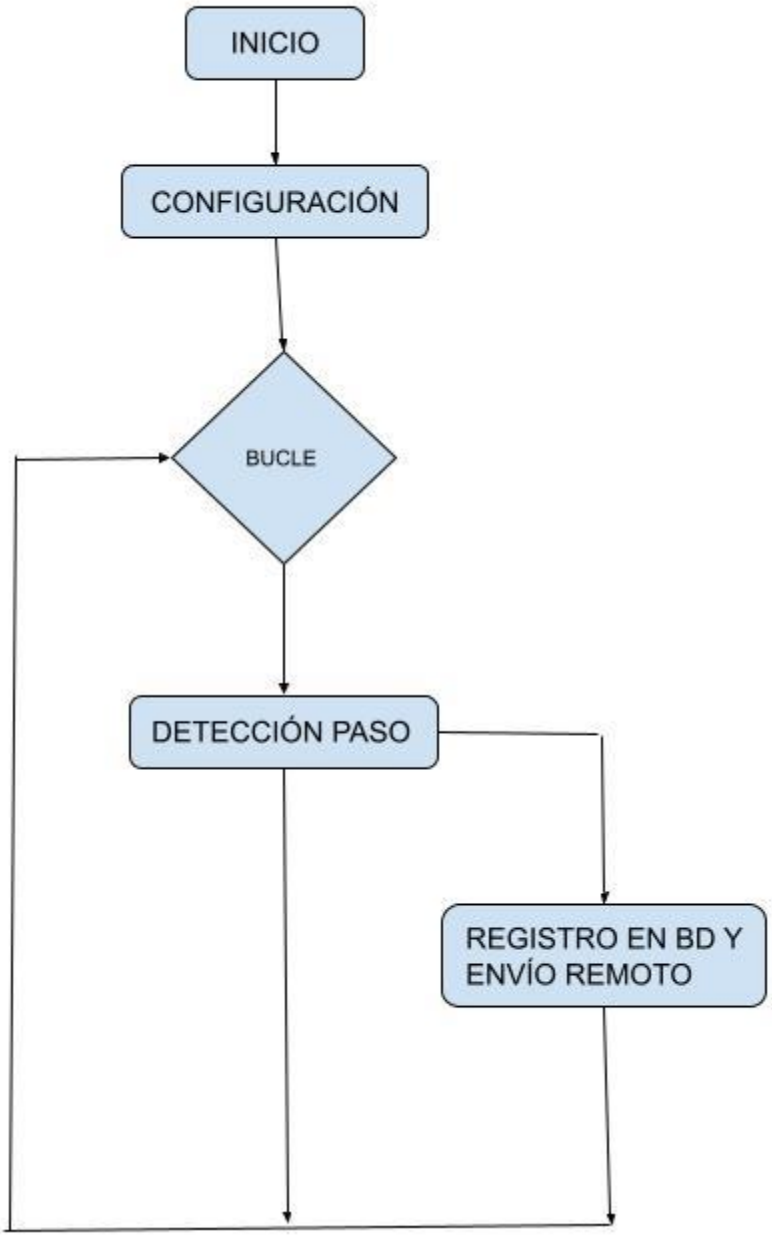
Parte 2: Instalación y Software

Diagrama a bloques del proyecto.

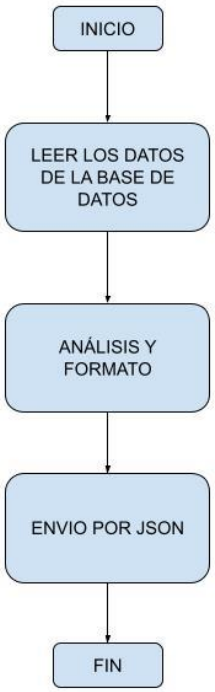


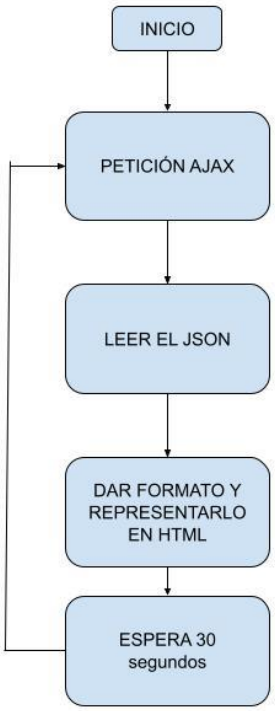


SOFTWARE


INICIO
CONFIGURACIÓN
BUCLE
DETECCIÓN PASO
REGISTRO EN BD Y
ENVÍO REMOTO

SOFTWARE


INICIO
LEER LOS DATOS DE LA BASE DE DATOS
ANÁLISIS Y FORMATO
ENVIO POR JSON
FIN


INICIO
PETICIÓN AJAX
LEER EL JSON
DAR FORMATO Y REPRESENTARLO EN HTML
ESPERA 30 segundos

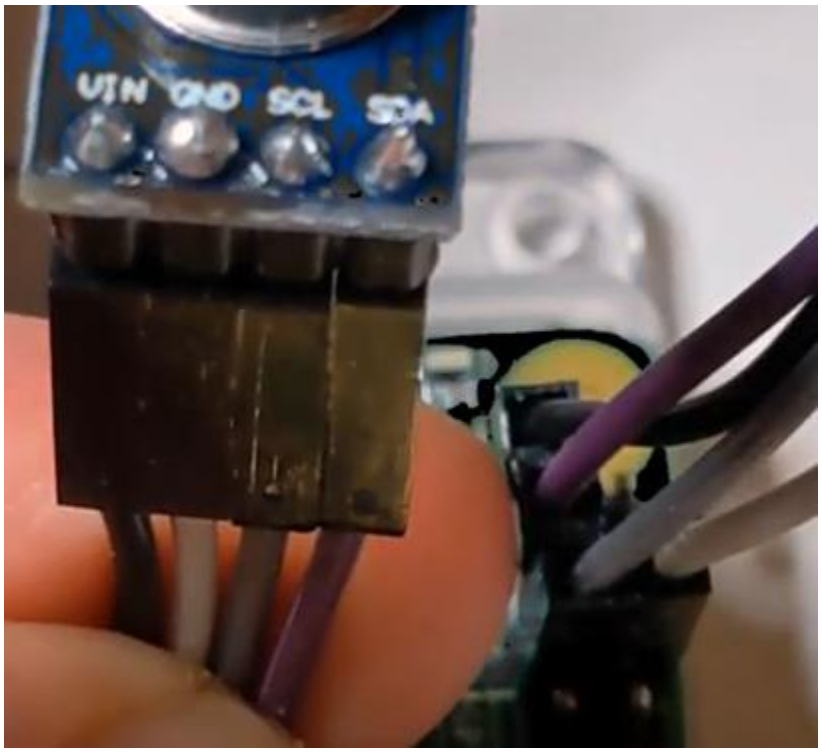
Instalación del sistema.

En seguida se mencionará como es la instalación del sistema físico, para esto previamente el monitor deberá de estar colocado en el mástil de madera, la cámara y sensor de temperatura ya pegados en la parte superior del monitor, el sensor PIR colocado y pegado debajo del dispensador de gel a la altura de la mano y el otro sensor PIR en la salida del local; Además que la computadora que hará como servidor para almacenamiento remoto ya deberá de estar conectado a la red y con una ip fija.

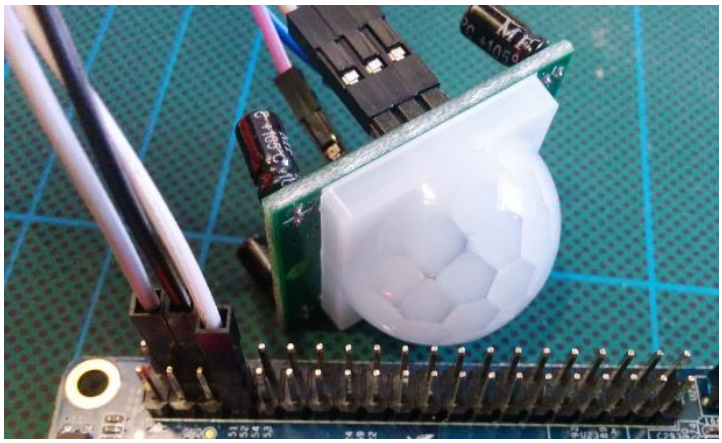
Posteriormente deberá de conectar el monitor por HDMI al micro HDMI del raspberry en la entrada 1; posteriormente el cable USB de la cámara deberá de conectarse a uno de los puertos del raspberry; Finalmente se deberá de usar el cable UTP para conectar la raspberry a la red a través de Ethernet.

Ahora para terminar la instalación física deberemos de conectar los sensores y alimentarlos, para esto deberemos de sacar una salida de tierra de la raspberry hacia una protoboard para que sirva de puente igualmente una salida de voltaje. Ahora empezaremos por la conexión del sensor de temperatura MLX90614, el cual alimentaremos con voltaje y tierra en sus salidas correspondientes, las que son salidas del

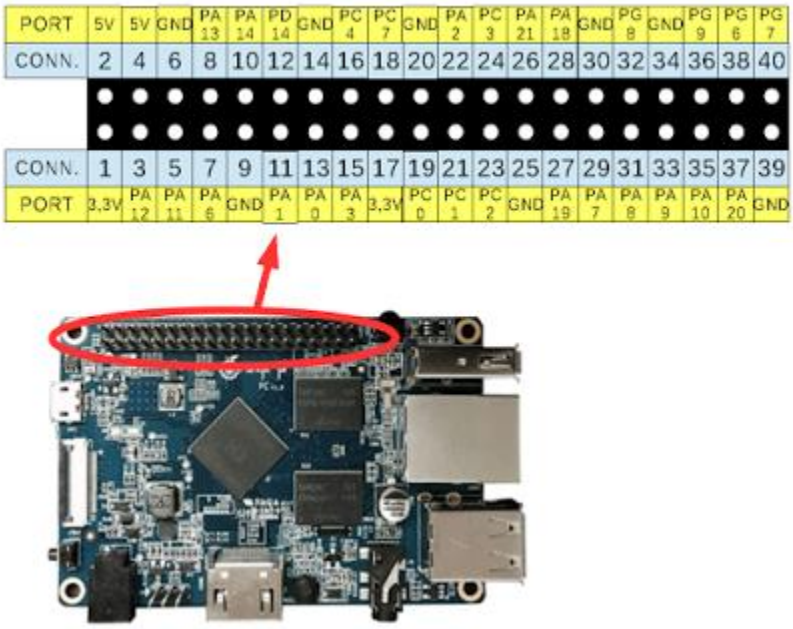
sensor lo deberemos de conectar a los pines 3 y 5 del raspberry. Se muestra un ejemplo:



Posteriormente seguiremos a la conexión de los sensores HC-SR501, para detectar seres vivos. Los cuales alimentaremos igualmente de voltaje y tierra y finalmente el pin de salida se conectará al ping 11 de la raspberry e igualmente con el otro sensor de la salida, pero en el ping 15. Se muestra un ejemplo a continuación, pero con los pines cambiados.



Finalmente, la numeración de los pines, para su conexión se lleva a cabo a través del siguiente diagrama; La conexión de los sensores se lleva a cabo por los cables para protoboard como lo son los hembra – hembra, macho – macho y hembra – macho.



Instalación del software necesario para el funcionamiento del proyecto.

Empezaremos con la instalación del SO de la plataforma de rapberry pi 4, para esto deberemos de descargar la ISO del SO de **Raspberry Pi OS** desde su página oficial (https://www.raspberrypi.org/software/operating-systems/), en su versión de escritorio, pero sin aplicaciones cargadas. Posteriormente deberemos de instalar el programa de **Ether** (https://www.balena.io/etcher/), con el cual instalaremos el ISO en la memoria SD para la Raspberry, posteriormente se deberá de poner la memoria en la Raspberry y encender para seguir los pasos de configuración y tener la Raspberry funcionando.

Posteriormente se deberá instalar los lenguajes de programación para que funcione el sistema los cuales son Python y PHP, Python ya esta instalado en el SO de Raspberry y solo quedara instalar PHP con el siguiente comando (sudo apt install php libapache2-mod-php php-mysql), con este comando se instalara php y su conector con mysql. Los lenguajes de HTML, CSS y JS son nativos de un navegador y no deberán instalarse.

Ahora deberá de instalarse los servicios de Apache y MySQL para almacenar y mostrar la información del sistema desde la raspberry, cabe destacar que estos también deberán de estar en el servidor, pero estos se pueden instalar con WAMPP o XAMPP. Los comandos para instalar mysql y apache en la raspberry son los siguientes:

1. sudo su root
2. apt-get update
3. apt-get upgrade
4. apt-get install apache2
5. apt-get install mariadb-server

Finalmente tenemos la instalación de las librerías necesarias para el funcionamiento de la cámara y sensores, las librerías para la interface de usuario son librerías web (Jquery, Bootstrap), por lo que no se deberán instalar ya que están en el mismo código; Las demás librerías son de Python y se deberán de instalar de la siguiente manera:

1. board 1.0
   pip3 install board
2. busio
   pip3 install adafruit-blinka
3. adafruit-circuitpython-mlx90614 1.2.6
   pip3 install adafruit-circuitpython-mlx90614
4. pyA20
   pip3 install pyA20
5. Cv2
   Pip3 install opencv-python
6. Mediapipe
   Pip3 install mediapipe
7. MySQL
   Pip3 install mysql
8. Requests
   Pip3 install requests

Finalmente tendremos que crear la base de datos de la aplicación tanto en la raspberry como en el servidor, para su almacenamiento remoto y local.

```
/*
SQLyog Community v13.1.7 (64 bit)
MySQL - 10.5.4-MariaDB : Database - facemask
*********************************************************************
*/

/*!40101 SET NAMES utf8 */;

/*!40101 SET SQL_MODE=''*/;

/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;
CREATE DATABASE /*!32312 IF NOT EXISTS*/`facemask` /*!40100 DEFAULT CHARACTER SET latin1 */;

USE `facemask`;

/*Table structure for table `clientes` */

DROP TABLE IF EXISTS `clientes`;

CREATE TABLE `clientes` (
  `id` bigint(20) unsigned NOT NULL AUTO_INCREMENT,
  `temperatura` float NOT NULL,
  `entrada` datetime NOT NULL,
  `salida` datetime DEFAULT NULL,
  `local` tinyint(1) NOT NULL DEFAULT 1,
  PRIMARY KEY (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

```

Código Python para el funcionamiento del proyecto.

Código principal para el monitoreo de la entrada con sensores

entrada.py

```python
# Codigo para monitoriar la entrada
import hashlib
import requests
import time
from time import sleep

# Conexion con la Base de Datos
from conexionBD import *

# Sensor MLX90614
import board
import busio as io
import adafruit_mlx90614

# Sensor de proximidad
from pyA20.gpio import gpio
from pyA20.gpio import port
from pyA20.gpio import connector

# IA deteccion cubrebocas
import cv2
import os
import mediapipe as mp

#Url del servidor
urlServidor = 'http://192.168.0.100/facemask/almacenamiento_remoto.php'
firmaDigital = 'HNos6pfF2M*$'

#Abrir camara
mp_face_detection = mp.solutions.face_detection
LABELS = ["Con_mascarilla", "Sin_mascarilla"]

```

```python
# leer el modelo
face_mask = cv2.face.LBPHFaceRecognizer_create()
face_mask.read("face_mask_model.xml")
cap = cv2.VideoCapture(1, cv2.CAP_DSHOW)

with mp_face_detection.FaceDetection(min_detection_confidence=0.5) as face_detection:

    # Configuracion de pines para el sensor de proximidad
    gpio.init()
    gpio.setcfg(port.PA1, gpio.INPUT)

    while True:
        # Monitoreo de cubrebocas
        ret, frame = cap.read()
        if ret == False: break
        frame = cv2.flip(frame, 1)

        height, width, _ = frame.shape
        frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
        results = face_detection.process(frame_rgb)

        if results.detections is not None:
            for detection in results.detections:
                xmin = int(detection.location_data.relative_bounding_box.xmin * width)
                ymin = int(detection.location_data.relative_bounding_box.ymin * height)
                w = int(detection.location_data.relative_bounding_box.width * width)
                h = int(detection.location_data.relative_bounding_box.height * height)
                if xmin < 0 and ymin < 0:
                    continue
                #cv2.rectangle(frame, (xmin,ymin), (xmin + w, ymin + h), (0,255,0), 5)
```

```python
            face_image = frame[ymin : ymin + h, xmin : xmin + w]
            face_image = cv2.cvtColor(face_image, cv2.COLOR_BGR2GRAY)
            face_image = cv2.resize(face_image, (72,72), interpolation=cv2.INTER_CUBIC)

            result = face_mask.predict(face_image)

            # Si tiene cubrebocas
            if result[1] < 150:
                color = (0,255,0) if LABELS[result[0]] == "Con_mascarilla" else (0,0,255)
                cv2.putText(frame, "{}".format(LABELS[result[0]]), (xmin, ymin - 15), 2, 1, color, 1, cv2.LINE_AA)
                cv2.rectangle(frame, (xmin,ymin), (xmin + w, ymin + h), color,2)

                # Detectar Temperatura, COnfiguras el pin para leer el de temperatura
                sensorTemperatura = adafruit_mlx90614.MLX90614(io.I2C(board.SCL, board.SDA, frequency=100000))
                temperatura = "{:.2f}".format(mlx.object_temperature)
                cv2.putText(frame, f"Temperatura: {temperatura}°C", (0,0), 2, 1, color, 1, cv2.LINE_AA)
                if temperatura >= 36.1 and temperatura <= 37.2:
                    # Detectar gel
                    if gpio.input(port.PA1) == 1:
                        cv2.putText(frame, f"Puedes pasar", (0,50), 2, 1, color, 1, cv2.LINE_AA)
                        # Insertar un cliente en la base de datos (ALmacenamiento Local)
                        fechaHora = datetime.datetime.now()
                        id = insertarCliente(temperatura, fechaHora)
                        # Generar Certificado y data
                        certificado = (hashlib.md5(firmaDigital.encode())).hexdigest()
                        dataSend = {
                            'certificado': certificado,
                            'query': f"INSERT INTO clientes(id, temperatura, entrada) VALUES({id}, {temperatura}, {
                        }
```

```
                            # Enviar al servidor
                            requests.post(urlServidor, data=dataSend)

                            # Dormir para el siguiente cliente 10s
                            time.sleep(10)
                        else:
                            cv2.putText(frame, f"Colocate Gel", (0,50), 2, 1, color, 1, cv2.LINE_AA)
                    else:
                        cv2.putText(frame, f"No tienes una temperatura ideal", (0,50), 2, 1, color, 1, cv2.LINE_AA)
            cv2.imshow("Frame", frame)
            k = cv2.waitKey(1)
            if k == 27:
                break

```

Código principal para el monitoreo de la salida con sensores

salida.py

```python
# Codigo para controlar la saida
import hashlib
import requests
import time

# Conexion con la Base de Datos
from conexionBD import *

# Sensor de proximidad
from pyA20.gpio import gpio
from pyA20.gpio import port
from pyA20.gpio import connector

#Url del servidor
urlServidor = 'http://192.168.0.100/facemask/almacenamiento_remoto.php'
firmaDigital = 'HNos6pfF2M*$'

if __name__ == '__main__':
    # Configuracion del sensor de seres vivos
    gpio.init()
    gpio.setcfg(port.PA3, gpio.INPUT)

    while True:
        # Si detecta movimiento
        if gpio.input(port.PA3) == 1:
            # Actualizar la base de datos con una salida
            fechaHora = datetime.datetime.now()
            id = updateSalida()
            # Generar Certificado y data
            certificado = (hashlib.md5(firmaDigital.encode())).hexdigest()
            dataSend = {
                'certificado': certificado,
                'query':  f"UPDATE clientes SET salida = {fechaHora}, local = 0 WHERE id = {id}"
            }
            # Enviar al servidor
            requests.post(urlServidor, data=dataSend)
```

Código para la conexión con mariaDB desde Python y hacer las querys

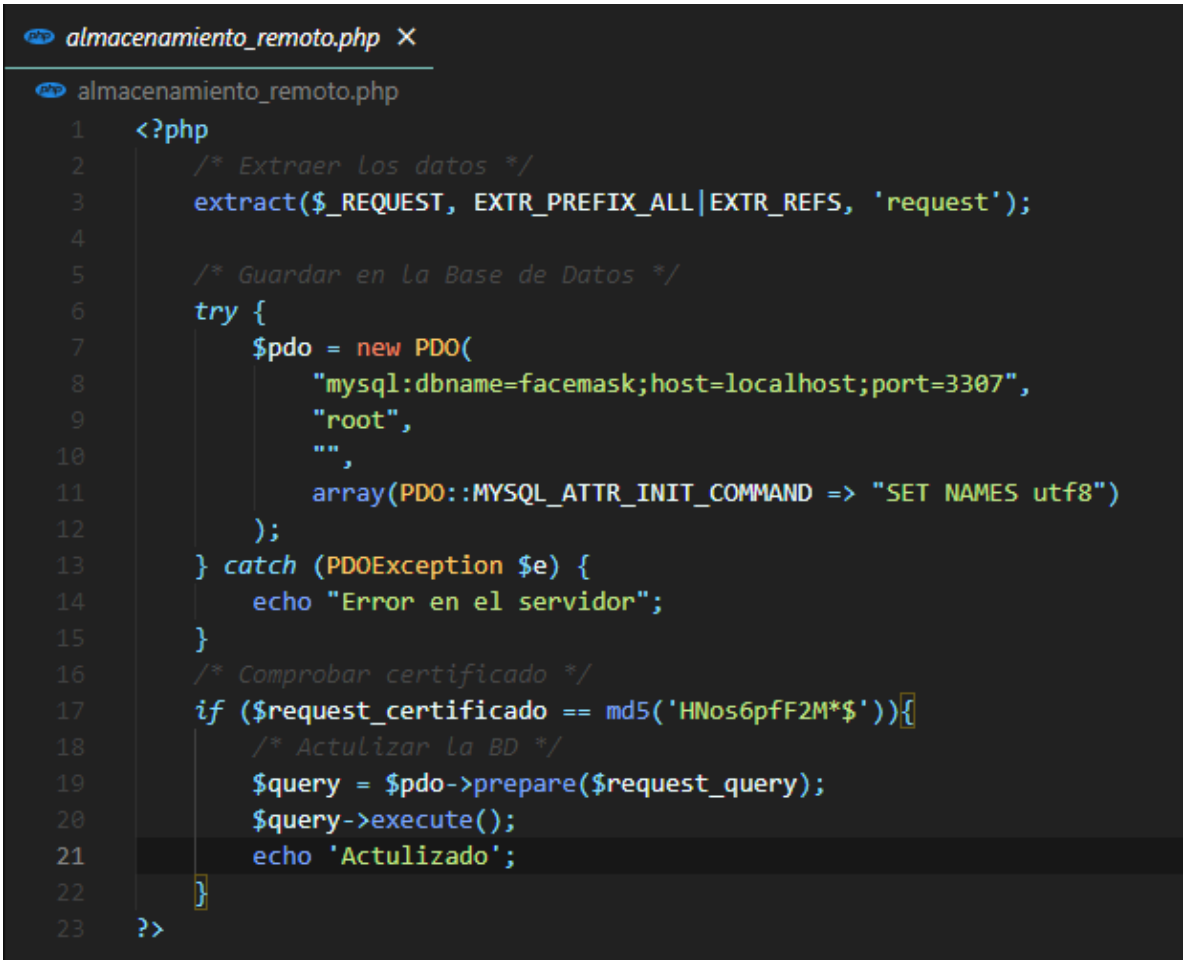
conexionBD.py

```python
# Conexion con la base de datos
import mysql.connector
import datetime

mydb = mysql.connector.connect(
        host="localhost",
        user="root",
        passwd="",
        database="facemask",
        port=3307
    )

def insertarCliente(temperatura, fechaHora):
    # Insertar
    cursor = mydb.cursor()
    query = "INSERT INTO clientes(temperatura, entrada) VALUES(%s, %s)"
    cursor.execute(query, (temperatura, fechaHora))
    mydb.commit()

    #Optener id
    query = "SELECT id FROM clientes WHERE local = 1 LIMIT 1"
    cursor.execute(query)
    datos = cursor.fetchone()
    cursor.close()

    return datos[0]

def updateSalida(fechaHora):
    # Obtener id
    cursor = mydb.cursor()
    query = "SELECT id FROM clientes WHERE local = 1 ORDER BY DESC entrada LIMIT 1"
    cursor.execute(query)
    id = cursor.fetchone()[0]

    # Actualizar
    query = f"UPDATE clientes SET salida = {fechaHora}, local = 0 WHERE id = {id}"
    cursor.execute(query)
    cursor.close()

    return id

```

* En este código se cambian los parámetros de conexión con la base de datos.

Código para el Servidor y Almacenamiento Remoto

Este script de php recibe las querys de la raspberry con lo cual llevara una copia de la información de la base de datos local asiendo así un almacenamiento remoto. Igualmente verifica las peticiones a través de la firma digital y certificado.



Código para la interface grafica

HTML

index.html

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <meta name="author" content="Carlos J. y Ernesto">
        <meta name="description" content="Proyecto final IOT de ISC ITM 2021">
        <title>Facemask</title>
        <link rel="preconnect" href="https://fonts.googleapis.com">
        <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
        <link href="https://fonts.googleapis.com/css2?family=Poppins&display=swap" rel="stylesheet">
        <link href="css/bootstrap.min.css" rel="stylesheet">
        <link rel='stylesheet' href='https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css'>
        <link rel="stylesheet" href="css/index.css">
    </head>
    <body>
        <header>
            <nav class="navbar navbar-light bg-light fixed-top">
              <div class="container-fluid">
                <a class="navbar-brand">
                    <img src="img/icono.jpg" alt="logo" height="34" class="d-inline-block align-top">
                    FaceMask
                </a>
              </div>
            </nav>
        </header>
        <div class="container">
            <div class="row">
                <div class="col-12 text-center">
                    <h2>INFORMACIÓN DEL SISTEMA</h2>
                </div>
                <div class="col-12">
                    <br>
                    <p>En esta página se muestra la información recolectada por los sensores de la entrada; para que usted pueda tomar acciones si se muestra algo que no concorde con sus estánd
                </div>
            </div>
            <br>
            <hr>
            <br>
            <div id="cargar">
                <div class="row" style="margin-bottom: 10em;">
                    <div class="col-12 text-center">
                        <div class="spinner-border text-dark" role="status">
                            <span class="sr-only">Loading...</span>
                        </div>
                    </div>
                </div>
            </div>
```

```
            <div class="row d-none" id="graficas">
                <div class="col-12 text-center">
                    <h2>ESTADO DEL LOCAL</h2>
                </div>
                <div class="col-md-6">
                    <div id="capacidadLocal" style="width: 900px; height: 500px;"></div>
                </div>
                <div class="col-md-4 offset-2">
                    <div class="card mt-5">
                        <div class="card-header bg-dark text-white">
                            <h5 class="card-title">Temperatura Promedio de los Clientes</h5>
                        </div>
                        <div class="card-body text-center">
                            <div class="temperatura text-dark" id="temperaturaPromedio">
                                36.2<sup>°C</sup>
                            </div>
                            <i class='fa fa-thermometer-half' style='font-size: 10rem;'></i>
                        </div>
                    </div>
                </div>
                <div class="col-md-12">
                    <div id="historialClientes" style="width: 100% ; height: 500px"></div>
                </div>
            </div>

        </div>
        <footer class="bg-light text-center mt-3">
            <br>
            <div class="text-center p-3" style="background-color: rgba(0, 0, 0, 0.2)">
              <p> © 2021 BAD</p>
              <p>Carlos J. Castro - Ernesto Vieyra</p>
            </div>
        </footer>

        <script src="js/bootstrap.bundle.min.js"></script>
        <script src="https://code.jquery.com/jquery-3.5.1.min.js" integrity="sha256-9/aliU8dGd2tb6OSsuzixeV4y/faTqgFtohetphbbj0=" crossorigin="anonymous"></script>
        <script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
        <script src="js/index.js"></script>
    </body>
</html>
```

CSS

index.css

```css
* {
    font-family: 'Poppins', sans-serif;
}
.container{
    margin-top: 5em;
}

.temperatura{
    font-size: 5rem;
    font-weight: bold;
    margin-top: 10px;
}
```

JS

index.js

```js
let graficas = {
    llenarCapacidad: function (clientes, faltantes) {
        google.charts.load("current", {packages:["corechart"]});
        google.charts.setOnLoadCallback(drawChart);
        function drawChart() {
            var data = google.visualization.arrayToDataTable([
                ['Task', 'Capacidad'],
                ['Personas dentro', clientes],
                ['Lugares Disponibles', faltantes]
            ]);
            var options = {
                title: 'Capacidad del Negocio',
                pieHole: 0.4,
            };

            var chart = new google.visualization.PieChart(document.getElementById('capacidadLocal'));
            chart.draw(data, options);
        }
    },
    llenarTermometro: function (temperatura) {
        $('#temperaturaPromedio').html(`
            ${temperatura}<sup>°C</sup>
        `);
    },
```

```
    historialDias: function (historial) {
        google.charts.load('current', {'packages':['corechart']});
        google.charts.setOnLoadCallback(drawChart);
            function drawChart() {

                let datos = [
                    ['Día', 'Clientes']
                ];
                historial.forEach(dia => {
                    datos.push([
                        dia.fecha,
                        parseInt(dia.numero)
                    ]);
                });
                console.log(datos);
                var data = google.visualization.arrayToDataTable(datos);

                var options = {
                    title: 'Historial de clientes por dia',
                    curveType: 'function',
                    legend: { position: 'bottom' },
                    series: {
                        0: { color: '#D00A0A' },
                    }
                };

                var chart = new google.visualization.LineChart(document.getElementById('historialClientes'));
                chart.draw(data, options);
            }
    }
};

let getDatos = function () {
    console.log('Peticion de datos');
    $('#cargar').removeClass('d-none');
    $('#graficas').addClass('d-none');
    $.ajax({
        type: "get",
        url: "getDatos.php",
        data: {},
        dataType: "json",
        success: function (response) {
            let capacidadMaxima = 200;
            graficas.llenarCapacidad(parseInt(response.numeroClientes), capacidadMaxima-response.numeroClientes);
            graficas.llenarTermometro(parseFloat(response.temperaturaPromedio).toFixed(2));
            graficas.historialDias(response.historial);
            $('#graficas').removeClass('d-none');
            $('#cargar').addClass('d-none');
        }
    });
}
$(document).ready(function () {
    getDatos();
    setInterval(() => {
        getDatos();
    }, 60000);
});
```

PHP

getDatos.php

```php
<?php
    $conexion = mysqli_connect("localhost", "root", "", "facemask", 3307);
    $query = "SELECT COUNT(*) AS numero FROM clientes WHERE LOCAL = 1 AND entrada = '2021-07-03'";
    $result = mysqli_query($conexion, $query);
    mysqli_close($conexion);
    $numeroActual = 0;
    do {
        $renglon = mysqli_fetch_array($result);
        if ($renglon != null) {
            $numeroActual = $renglon['numero'];
        }
    } while ($renglon != null);

    $conexion = mysqli_connect("localhost", "root", "", "facemask", 3307);
    $query = "SELECT AVG(temperatura) AS temperaturaPromedio FROM clientes WHERE LOCAL = 1";
    $result = mysqli_query($conexion, $query);
    mysqli_close($conexion);
    $temperaturaPromedio = 0;
    do {
        $renglon = mysqli_fetch_array($result);
        if ($renglon != null) {
            $temperaturaPromedio = $renglon['temperaturaPromedio'];
        }
    } while ($renglon != null);

    $conexion = mysqli_connect("localhost", "root", "", "facemask", 3307);
    $query = "SELECT COUNT(*) AS numero, entrada AS fecha  FROM clientes WHERE LOCAL = 1 GROUP BY entrada";
    $result = mysqli_query($conexion, $query);
    mysqli_close($conexion);
    $dias=[];
    do {
        $renglon = mysqli_fetch_array($result);
        if ($renglon != null) {
            $dias[] = [
                'fecha' => $renglon['fecha'],
                'numero' => $renglon['numero']
            ];
        }
    } while ($renglon != null);

    $data = [
        'numeroClientes' => $numeroActual,
        'temperaturaPromedio' => $temperaturaPromedio,
        'historial' => $dias
    ];

    echo json_encode($data);
?>
```

Interface grafica