Contenido

[Resumen ejecutivo 3](bookmark://_Toc49818380#_Toc49818380)

[Descripción del problema 3](bookmark://_Toc49818381#_Toc49818381)

[Objetivos específicos 3](bookmark://_Toc49818382#_Toc49818382)

[¿Cómo funciona la solución? 3](bookmark://_Toc49818383#_Toc49818383)

[¿Qué van a construir para resolver el problema? 4](bookmark://_Toc49818384#_Toc49818384)

[Recursos de hardware y de software 4](bookmark://_Toc49818385#_Toc49818385)

[Recursos de hardware 4](bookmark://_Toc49818386#_Toc49818386)

[Recursos de software 5](bookmark://_Toc49818387#_Toc49818387)

[Explicación paso a paso de la implementación del proyecto 5](bookmark://_Toc49818388#_Toc49818388)

[Diagramas de diseño del proyecto, diagrama de circuito, diagrama del modelo entidad-relación, diagrama de casos UML, diagrama de despliegue 6](bookmark://_Toc49818389#_Toc49818389)

[Diagrama de diseño del proyecto 6](bookmark://_Toc49818390#_Toc49818390)

[Diagrama de circuito 7](bookmark://_Toc49818391#_Toc49818391)

[Diagrama del modelo entidad-relación 7](bookmark://_Toc49818392#_Toc49818392)

[Diagrama de casos UML 8](bookmark://_Toc49818393#_Toc49818393)

[Diagrama de despliegue 8](bookmark://_Toc49818394#_Toc49818394)

[Descripción de los campos, tipos de datos creados en la base de datos, y el código SQL 9](bookmark://_Toc49818395#_Toc49818395)

[Tabla informe 9](bookmark://_Toc49818396#_Toc49818396)

[Codigo SQL 9](bookmark://_Toc49818397#_Toc49818397)

[Tabla local 11](bookmark://_Toc49818398#_Toc49818398)

[Codigo SQL 11](bookmark://_Toc49818399#_Toc49818399)

[Explicación del código fuente completo desarrollado con los comentarios correspondientes 12](bookmark://_Toc49818400#_Toc49818400)

[Código ARDUINO 12](bookmark://_Toc49818401#_Toc49818401)

[Explicación de código 12](bookmark://_Toc49818402#_Toc49818402)

[Código del IDE 12](bookmark://_Toc49818403#_Toc49818403)

[Python 15](bookmark://_Toc49818404#_Toc49818404)

[Explicación de código 15](bookmark://_Toc49818405#_Toc49818405)

[Codigo Python 15](bookmark://_Toc49818406#_Toc49818406)

[PhP 18](bookmark://_Toc49818407#_Toc49818407)

[Codigo salida\_por\_id.php 18](bookmark://_Toc49818408#_Toc49818408)

[Revisar\_informe.php 19](bookmark://_Toc49818409#_Toc49818409)

[Ingreso\_por\_id.php 20](bookmark://_Toc49818410#_Toc49818410)

[Crear\_informes\_vacios.php 22](bookmark://_Toc49818411#_Toc49818411)

[Cantidad\_python.php 23](bookmark://_Toc49818412#_Toc49818412)

[Index.php 24](bookmark://_Toc49818413#_Toc49818413)

[Get\_informes\_id.php 24](bookmark://_Toc49818414#_Toc49818414)

[Análisis de presupuesto 25](bookmark://_Toc49818415#_Toc49818415)

[Conclusiones 26](bookmark://_Toc49818416#_Toc49818416)

[Referencias bibliográficas 26](bookmark://_Toc49818417#_Toc49818417)

Manual técnico

# Resumen ejecutivo

En el presente trabajo, se resuelve un problema de índole sanitaria, por el cual se requieren implementar un plan de trabajo que garantiza la seguridad de las personas sin afectuar la rutina diaria de estas. Por el cual, utilizando, equipos electrónicos y métodos computacionales, se diseñará un sistema, no invasivo, que regule el flujo de personas en locales comerciales, para evitar generar aglomeraciones.   
Del cual obtuvimos resultados bastante favorables al momento de implementarlo, ya que nos permitía una rápida gestión de este, al mismo tiempo de llevar un correcto registro de la actividad en los locales.

# Descripción del problema

Ciertas condiciones han cambiado debido a la pandemia. La distancia para evitar contagios que se debe tener entre personas es de dos metros en los locales que dan servicio al público, para así resguardar la salud de los clientes. Aforo, es un concepto que ha ganado relevancia dentro de este contexto, haciendo referencia a la cantidad máxima de personas que pueden estar dentro de un establecimiento manteniendo estos dos metros de distancia. Por lo cual, se le solicita que gestione un sistema que permita tener un conteo de las personas que se encuentran en un establecimiento, de tal manera que cumplan el aforo. Para un mejor control el dueño del establecimiento le ha solicitado que almacene toda la información y la muestre en un aplicativo, y además envíe una alerta cuando se ha excedido el aforo.

# Objetivos específicos

1. Gestionar un sistema que permita tener un conteo de las personas que se encuentran en un establecimiento, de manera que se cumpla con las leyes del aforo.
2. Diseñar un sistema aplicativo que sea de fácil uso y acceso, para que tanto el dueño del local como el cliente pueda tener acceso a la información del número de personas que visitan locales públicos.
3. Con la ayuda de histogramas generar informes cada hora de la cantidad de personas que visitan lugares públicos para que las autoridades puedan tener un mejor control sin que se sobrepase el límite de aforo permitido, y que los clientes cumplan con la ley del distanciamiento para evitar la propagación del virus.

# ¿Cómo funciona la solución?

La solución funciona de manera que el prototipo hecho se instale en tanto en la puerta de ingreso de los locales como la puerta de salida, en la parte exterior de las puertas se instalará una cámara web-cam la misma que estará conectada a una computadora la cual gestiona la solicitud de acceso al local. La computadora envía una señal a un microcontrolador para que ejecute una acción. La computadora por medio de la web-cam se encargará del conteo de personas por medio de reconocimiento facial, así cada vez que reconozca a una persona que vaya a ingresar al local la contabilice y aumente el número de asistentes, y de forma contraria lo hará también cuando las personas se retiren del local.

El número de personas que se contabilicen serán datos que lo cuales se almacenaran en un servidor web, de esta manera se generaran informes en los que los usuarios podrán ver la cantidad de personas que hay en los locales. Los clientes que quieran ingresar a locales públicos sin que romper las leyes de distanciamiento lo podrán hacer accediendo a nuestra página web y revisando los informes que se generan en los locales, ya que estos informes son generados en tiempo real y mostraran el número de personas que se encuentra en el local. También el diseño de la página web funcionara de manera que ayude a las autoridades a que tengan un mejor control de locales que no cumplan con las leyes de aforo permitido, para que de esta manera evitar el aumento del virus y así poder regresar a una mejor estabilidad económica.

# ¿Qué van a construir para resolver el problema?

Para la resolución de la problemática el hardware que se construirá será con ayuda de elementos electrónicos como lo son:

* Placa de Arduino Controller Board
* Jumpers
* Cable USB
* Servomotores
* Leds
* Resistencias
* Cámaras web

Con estos elementos se realizará un diseño el cual sea fácil de colocar en una puerta de ingreso y salida de locales públicos. Ya que en la entrada se tendrá un reconocedor fácil para registrar la entrada al local, siempre y cuando el led 1 este encendido (Este indica si queda espacio en el local, para que las personas puedan ingresar), esta información será enviada a la base de datos en la cual se registra la visita por fechas, para que por medio de una página web se pueda acceder a este registro.

# Recursos de hardware y de software

## Recursos de hardware

-Placa de Arduino, Controles Board

Es la placa que se usara en nuestro Sistema de control ya que nos permite trabajar de manera directa con sus entradas y salidas.

-Jumpers

Los cables que se utilizaran para conectar los diferentes elementos con la placa controladora.

-Cable USB

El medio por el cual nuestra placa controladora sera energizada y ademas el medio por el cual se comunica con la computadora la cual le va a indicar las acciones que debe tomar la placa controladora.

-Servomotores

Servirán para que al momento de enviar la señal tanto de entrada como de salida de los clientes se pueda reflejar en estos, haciéndolos actuar.

-leds

Sirven para indicar estados, en nuestro caso, indicar cuando el local tenga espacio para que los clientes puedan entrar.

-Resistencias

Las usamos para evitar que nuestros componentes eléctricos se puedan llegar a quemar.

-Camaras web

Es el sistema por el cual monitorearemos la entrada y salida de los clientes

## Recursos de software

Para el almacenamiento y análisis de los datos se utilizarán lo siguiente:

-Base de datos  
Se utilizará el servicio de base de datos(webhosting) en el cual almacenaremos toda la información que tenga que ver con la administración de nuestro proyecto.

-Programación en lenguaje CSS  
Es el lenguaje que controla como se muestra la pagina web y su estructura.

-Programación en Arduino  
Lo utilizaremos para representar la entrada y salida de los clientes, por medio de servomotores simularemos cuando permite el paso, o de caso contrario, restringe el acceso al local

-Programación en Python   
Sirve para lograr la conexión entre el arduino con la base de datos, en donde están todos los métodos que se implementan con respecto a esta.   
Además de los métodos de control de las cámaras web

# Explicación paso a paso de la implementación del proyecto

1) se realiza el circuito que será controlado con un Arduino

2)se programó el Arduino para accionar los servomotores que representan las puertas de ingreso y de salida del local e indicar el estado del local(si se puede ingresar o si se encuentra en la capacidad máxima)

3)se desarrolló el codigo en Python el cual permita la comunicación entre la computadora y el Arduino escribiendo en el puerto serial

4)se creó la base de datos en un servidor remoto

5) se crearon métodos http para poder realizar peticiones a la base de datos

6) se implementó un método en Python el cual realiza las peticiones al servidor

7)se creó un script que permite reconocimiento facial por medio de cámaras

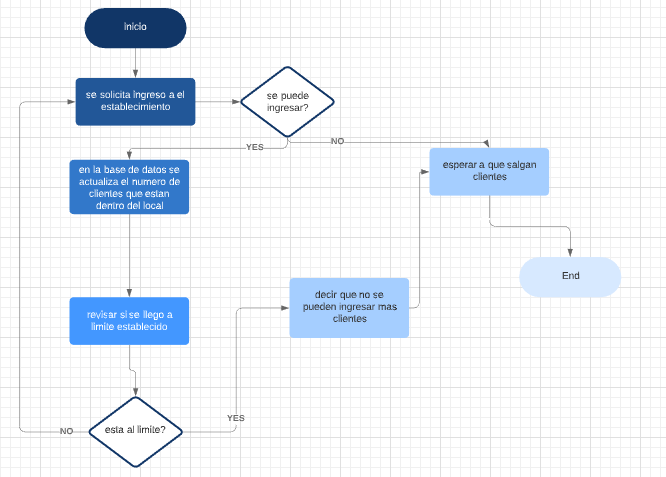
8)se modificó el script para que realice peticiones y se actualice la base de datos cuando la camara de ingreso reconoce la cara de una persona

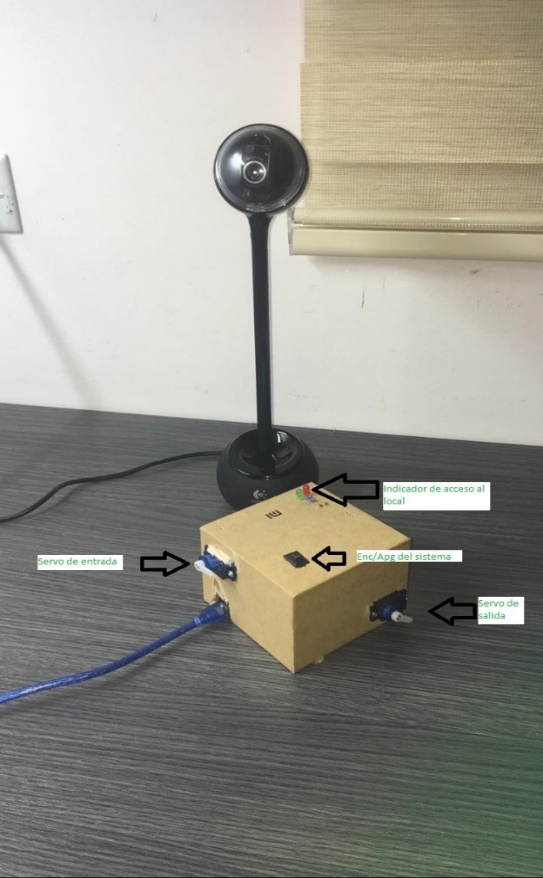
9)se modificó el script para que realice peticiones y se actualice la base de datos cuando la camara de salida reconoce la cara de una persona

10) se creó una página web que permita mostrar la información de la base de datos de manera más placentera y dinámica.

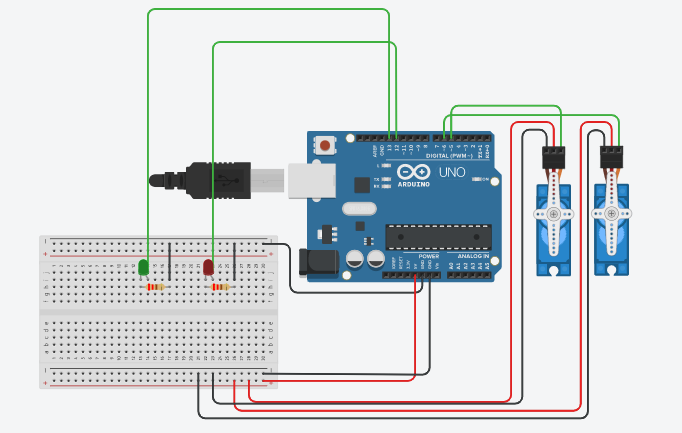
# Diagramas de diseño del proyecto, diagrama de circuito, diagrama del modelo entidad-relación, diagrama de casos UML, diagrama de despliegue

## Diagrama de diseño del proyecto

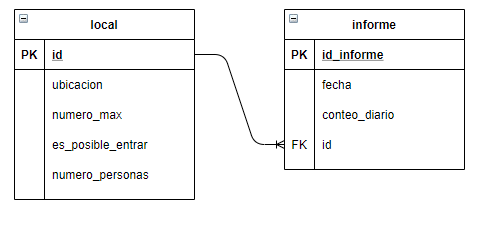


Se muestra el diseño en físico, en donde se puede apreciar los leds que indican el acceso al local, el botón que enciende o apaga el sistema.

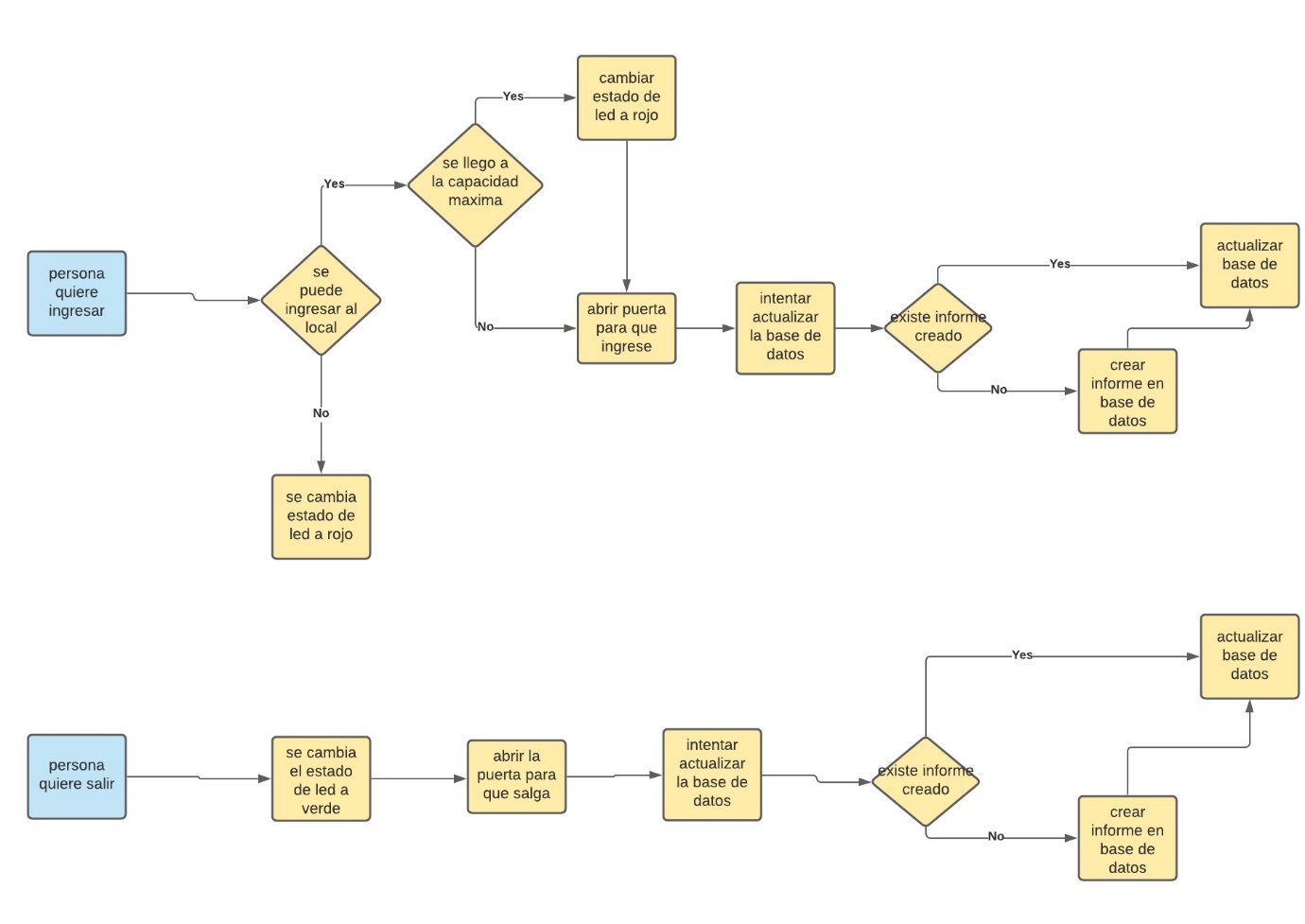
## Diagrama de circuito



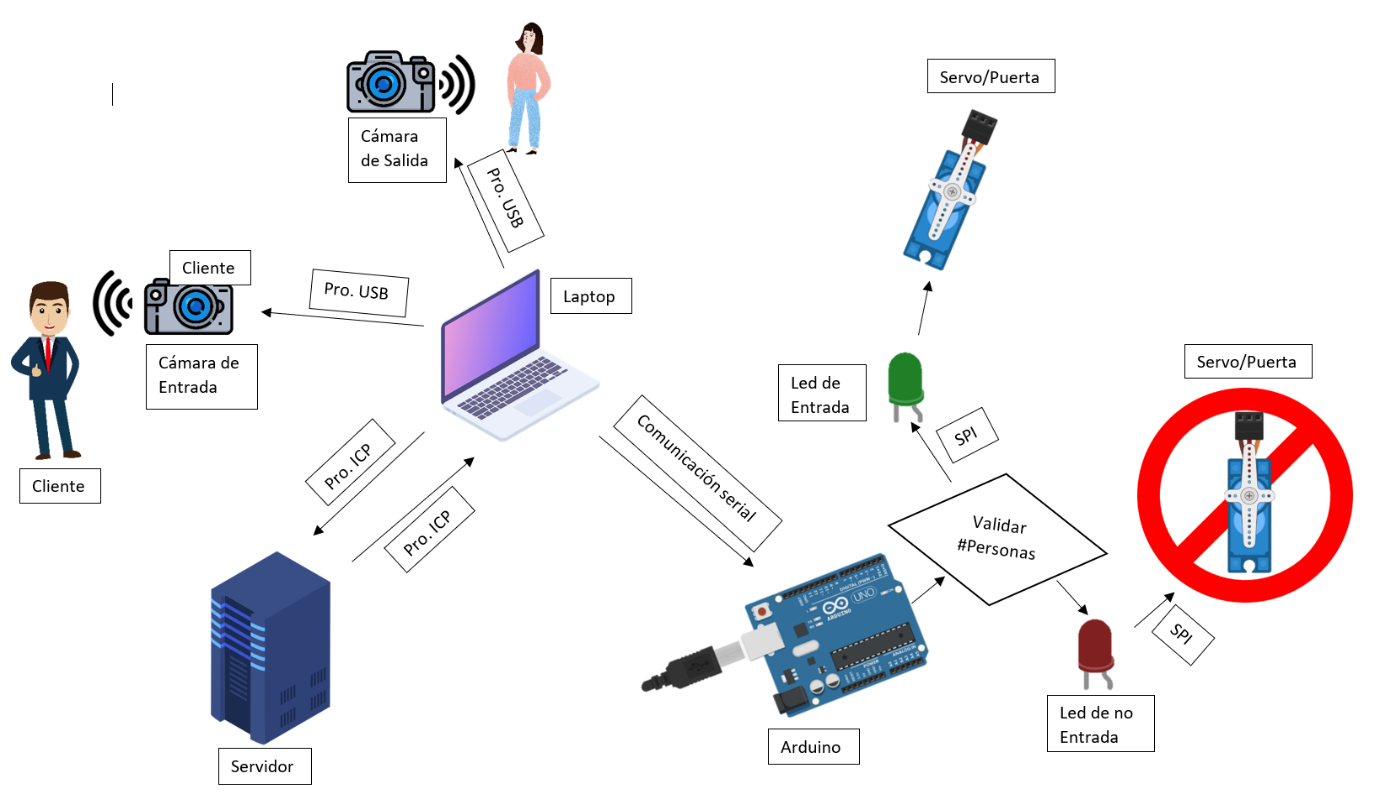
## Diagrama del modelo entidad-relación



## Diagrama de casos UML



## Diagrama de despliegue

Para el diagrama de despliegue vamos a implementar nuestro proyecto con los respectivos protocolos de comunicación en el cada uno de los componentes, después del diagrama vamos a explicar cada paso del proyecto.

# Descripción de los campos, tipos de datos creados en la base de datos, y el código SQL

## Tabla informe

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre del campo | Tipo de dato | Descripción | |
| Id\_informe | int | Es el número que se le asigna a cada informe que se introduce en la base de datos. | |
| fecha | datetime | Es la fecha y hora en la que se realiza la actualización de registro. |
| conteo\_diario | int | Es la cantidad de personas que entraron al local. | |
| id | int | Esta id pertenece al local al cual se está ingresando su registro. |

### Codigo SQL

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `informe`

--

CREATE TABLE `informe` (

`id\_informe` int(11) NOT NULL,

`fecha` datetime NOT NULL,

`conteo\_diario` int(11) NOT NULL,

`id` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

--

-- Volcado de datos para la tabla `informe`

--

INSERT INTO `informe` (`id\_informe`, `fecha`, `conteo\_diario`, `id`) VALUES

(79, '2020-08-26 07:00:00', 100, 1),

(80, '2020-08-26 08:00:00', 80, 1),

(81, '2020-08-26 09:00:00', 83, 1),

(82, '2020-08-26 10:00:00', 87, 1),

(83, '2020-08-26 11:00:00', 98, 1),

(84, '2020-08-26 12:00:00', 56, 1),

(85, '2020-08-26 13:00:00', 61, 1),

(86, '2020-08-26 14:00:00', 76, 1),

(87, '2020-08-26 15:00:00', 64, 1),

(88, '2020-08-26 16:00:00', 43, 1),

(89, '2020-08-26 17:00:00', 52, 1),

(90, '2020-08-27 07:00:00', 65, 1),

(91, '2020-08-27 08:00:00', 65, 1),

(92, '2020-08-27 09:00:00', 76, 1),

(93, '2020-08-27 10:00:00', 23, 1),

(94, '2020-08-27 11:00:00', 98, 1),

(95, '2020-08-27 12:00:00', 54, 1),

(96, '2020-08-27 13:00:00', 65, 1),

(97, '2020-08-27 14:00:00', 76, 1),

(98, '2020-08-27 15:00:00', 56, 1),

(99, '2020-08-27 16:00:00', 54, 1),

(100, '2020-08-27 17:00:00', 23, 1),

(101, '2020-08-26 07:00:00', 23, 2),

(102, '2020-08-26 08:00:00', 80, 2),

(103, '2020-08-26 09:00:00', 43, 2),

(104, '2020-08-26 10:00:00', 43, 2),

(105, '2020-08-26 11:00:00', 54, 2),

(106, '2020-08-26 12:00:00', 46, 2),

(107, '2020-08-26 13:00:00', 89, 2),

(108, '2020-08-26 14:00:00', 54, 2),

(109, '2020-08-26 15:00:00', 98, 2),

(110, '2020-08-26 16:00:00', 76, 2),

(111, '2020-08-26 17:00:00', 76, 2),

(112, '2020-08-27 07:00:00', 54, 2),

(113, '2020-08-27 08:00:00', 87, 2),

(114, '2020-08-27 09:00:00', 83, 2),

(115, '2020-08-27 10:00:00', 65, 2),

(116, '2020-08-27 11:00:00', 45, 2),

(117, '2020-08-27 12:00:00', 98, 2),

(118, '2020-08-27 13:00:00', 78, 2),

(119, '2020-08-27 14:00:00', 54, 2),

(120, '2020-08-27 15:00:00', 89, 2),

(121, '2020-08-27 16:00:00', 43, 2),

(122, '2020-08-27 17:00:00', 68, 2);

## Tabla local

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre del campo | Tipo de dato | Descripción | |
| Id | Int | Este valor es único para cada local, es por el cual se los va a identificar | |
| Ubicación | varchar | Es la locación geográfica del local |
| Numero\_max | int | Es la cantidad max de personas que puede tener el local | |
| Es\_poisible\_entrar | tinyint | Es la variable que indica si el local alcanzo su capacidad maxima |
| Numero\_personas | int | Es la cantidad de personas que se encuentran en ese momento en el local | |

### Codigo SQL

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `local`

--

CREATE TABLE `local` (

`id` int(11) NOT NULL,

`ubicacion` varchar(50) COLLATE utf8\_unicode\_ci NOT NULL,

`numero\_max` int(11) NOT NULL,

`es\_posible\_entrar` tinyint(1) NOT NULL,

`numero\_personas` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

--

-- Volcado de datos para la tabla `local`

--

INSERT INTO `local` (`id`, `ubicacion`, `numero\_max`, `es\_posible\_entrar`, `numero\_personas`) VALUES

(1, 'Ciudad Celeste', 30, 1, 8),

(2, 'la joya', 45, 0, 45);

# Explicación del código fuente completo desarrollado con los comentarios correspondientes

## Código ARDUINO

### Explicación de código

El Arduino se encarga de indicar si hay posibilidad de ingresar al local o si está lleno el local y además cuando se le envié una señal de ingreso o de salida esta abra la puerta que requiera.

El Arduino toma en total 4 posibles acciones dependiendo de la señal que lea y son:

‘I’ para indicar el estado de ingreso el cual abre una puerta de ingreso por un tiempo definido

‘S’ para indicar el estado de salida el cual abre una puerta de salida por un tiempo definido

‘F’ para indicar el estado de local lleno encendiendo un led rojo

‘D’ para indicar el estado de local disponible encendiendo un led verde

### Código del IDE

#include <Servo.h>

// Declaramos la variable para controlar el servo

Servo servoMotor\_entrada;

Servo servoMotor\_salida;

//variable para lectura puerto serial

char Dato;

//variable de estado de la puertas

bool entrando = false;

bool saliendo = false;

// variables para esperar un tiempo sin metodo delay

unsigned long previousMillis\_entrada = 0; // will store last time LED was updated

unsigned long previousMillis\_salida = 0;

// intervalo que se demoran los servos

const long interval = 1000;

//pines de los leds para indicar si se puede ingresar

int red\_light\_pin= 12;

int green\_light\_pin = 13;

void setup() {

// Iniciamos el monitor serie para mostrar el resultado

Serial.begin(9600);

// Iniciamos el servo para los pines 6 y 5

servoMotor\_entrada.attach(6);

servoMotor\_salida.attach(5);

// definir pines de salida

pinMode(red\_light\_pin, OUTPUT);

pinMode(green\_light\_pin, OUTPUT);

//encender led verde

digitalWrite(red\_light\_pin,LOW);

digitalWrite(green\_light\_pin,HIGH);

// poner en 0 grados los servos

servoMotor\_salida.write(0);

servoMotor\_entrada.write(0);

}

void loop() {

//leer tiempo actual

unsigned long currentMillis = millis();

//verificar que se escribe en el puerto serial

if (Serial.available() > 0) {

// leer el Dato del puerto serial y ejecutar la acción de abrir,cerrar,mostrar ocupado o mostrar disponible dependiendo del valor (I,S,F,D)

Dato = Serial.read();

if(Dato == 'I' and entrando == false){

entrando = true;

previousMillis\_entrada = currentMillis;

}

if(Dato == 'S' and saliendo == false){

saliendo = true;

previousMillis\_salida = currentMillis;

}

if(Dato == 'F'){

digitalWrite(red\_light\_pin,HIGH);

digitalWrite(green\_light\_pin,LOW);

}

if(Dato == 'D'){

digitalWrite(red\_light\_pin,LOW);

digitalWrite(green\_light\_pin,HIGH);

}

}

//si el estado del servo es verdadero se abre un tiempo el servo

if(entrando){

servoMotor\_entrada.write(90);

if (currentMillis - previousMillis\_entrada >= interval) {

entrando = false;

servoMotor\_entrada.write(0);

}

}

if(saliendo){

servoMotor\_salida.write(90);

if (currentMillis - previousMillis\_salida >= interval) {

saliendo = false;

servoMotor\_salida.write(0);

}

}

}

## Python

### Explicación de código

En el código de Python se instancian dos cámaras en la primera es para validar el intento de ingreso y el segundo cámara valida el intento de salida

Intento de ingreso:

se pregunta a la base de datos si se puede ingresar al local y despues se manda la señal al Arduino dependiendo de la respuesta de la base de datos para cambiar los estados del Arduino.

en caso de que se puede ingresar se valida que exista el informe a esa hora en la base de datos en caso de no existir se crea, después se suma la cantidad de personas que ingresaron.

Intento de salida:

Se llama actualiza el valor de las personas dentro de la base de datos y se cambia el estado del Arduino a disponible.

### Codigo Python

import cap as cap  
import numpy as np  
import cv2  
import requests  
import serial, time  
import datetime  
#inicializar arduino  
arduino = serial.Serial("COM3", 9600)  
  
#variable para hacer el query  
payload ={'id':1}  
time.sleep(2)  
  
#metodo que intenta acceder a la base de datos validando que se puede ingresar al local especifico: si se puede entrar se actualiza la cantidad de personas en la base de datos  
def intentar\_ingresar():  
 r = requests.get("https://trabajocastro4.000webhostapp.com/cantidad\_python.php", params=payload)  
 #print(r.text)  
 if(r.text =="verdad"):  
 arduino.write(b'I')  
 arduino.write(b'D')  
 tiempo = datetime.datetime.now()  
 tiempo = tiempo.replace(minute=0, second=0, microsecond=0)  
 #print(tiempo)  
  
 #preguntar si en la base de datos existe el informe a esa hora , en caso de no existir se crea uno  
 payload2 = {'id': 1,'fecha':tiempo}  
 r2 = requests.get("https://trabajocastro4.000webhostapp.com/revisar\_informe.php", params=payload2)  
 #print(r2.text)  
  
 if(r2.text == "no existe"):  
 payload3 = {'id': 1, 'fecha': tiempo}  
 r3 = requests.get("https://trabajocastro4.000webhostapp.com/crear\_informes\_vacios.php", params=payload3)  
 #print(r3.text)  
  
 #incrementar el numero de personas que han ingresado  
 payload4 = {'id': 1, 'fecha': tiempo}  
 r4 = requests.get("https://trabajocastro4.000webhostapp.com/ingreso\_por\_id.php", params=payload4)  
 #print(r4.text)  
  
 #en caso de que no se puede ingresar por numero de personas se cambia estado del arduino  
 else:  
 arduino.write(b'F')  
  
 #se vuelve a preguntar si se puede ingresar para actualizar el estado del arduino  
 r = requests.get("https://trabajocastro4.000webhostapp.com/cantidad\_python.php", params=payload)  
 #print(r.text)  
 if (r.text == "verdad"):  
 arduino.write(b'D')  
 else:  
 arduino.write(b'F')  
  
  
  
  
#metodo que intenta acceder a la base de datos validando que existe a esa hora un informe para un local especifico: si existe actualiza la cantidad de personas en la base de datos  
def intentar\_salir():  
 #cambiar estado del arduino para salir y que esta disponible para ingreso  
 arduino.write(b'S')  
 arduino.write(b'D')  
 tiempo = datetime.datetime.now()  
 tiempo = tiempo.replace(minute=0, second=0, microsecond=0)  
 #print(tiempo)  
 #preguntar si existe informe a esa hora para el local  
 payload2 = {'id': 1, 'fecha': tiempo}  
 r2 = requests.get("https://trabajocastro4.000webhostapp.com/revisar\_informe.php", params=payload2)  
 #print(r2.text)  
 #si no existe se crea un informe a esa hora  
 if (r2.text == "no existe"):  
 payload3 = {'id': 1, 'fecha': tiempo}  
 r3 = requests.get("https://trabajocastro4.000webhostapp.com/crear\_informes\_vacios.php", params=payload3)  
 #print(r3.text)  
 #se reduce la cantidad que esta actualmente en el local  
 payload4 = {'id': 1, 'fecha': tiempo}  
 r4 = requests.get("https://trabajocastro4.000webhostapp.com/salida\_por\_id.php", params=payload4)  
 #print(r4.text)  
  
  
  
#se hace uso de haarcascade para detectar una cara de frente con los ojos si estan los unos un tiempo definido por los contadores, cuando se cumple ese tiempo se  
# ejecutan las funciones de intento entrar y de intento salir dependiendo de la camara que esta reconociendo la cara  
face\_cascade = cv2.CascadeClassifier('haarcascade\_frontalface\_default.xml')  
eye\_cascade = cv2.CascadeClassifier('haarcascade\_eye.xml')  
cap = cv2.VideoCapture(1)  
cap2 = cv2.VideoCapture(0)  
hay\_elementos\_1 = False  
hay\_elementos\_2 = False  
contar\_cuadros\_1=0;  
contar\_cuadros\_2 = 0;  
while True:  
 ret, img = cap.read()  
 gray = cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY)  
 faces = face\_cascade.detectMultiScale(gray,1.3,5)  
 hay\_elementos\_1 = False  
 for (x,y,w,h) in faces:  
 cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)  
 roi\_gray = gray[y:y+h , x:x+w]  
 roi\_color = img[y:y+h , x:x+w]  
 eyes = eye\_cascade.detectMultiScale(roi\_gray)  
 for(ex,ey,ew,eh) in eyes:  
 cv2.rectangle(roi\_color,(ex,ey),(ex+ew,ey+eh),(0,255,0),2)  
 hay\_elementos\_1 = True  
 if(hay\_elementos\_1):  
 contar\_cuadros\_1 += 1  
 if(contar\_cuadros\_1>=30):  
 print("entrar.....")  
 intentar\_ingresar()  
 contar\_cuadros\_1 = 0  
 else:  
 contar\_cuadros\_1 = 0  
 cv2.imshow('img',img)  
 k = cv2.waitKey(30) & 0xff  
 if k ==27:  
 break  
  
  
 ret2, img2 = cap2.read()  
 gray2 = cv2.cvtColor(img2, cv2.COLOR\_BGR2GRAY)  
 faces2 = face\_cascade.detectMultiScale(gray2, 1.3, 5)  
 hay\_elementos\_2 = False  
 for (x, y, w, h) in faces2:  
 cv2.rectangle(img2, (x, y), (x + w, y + h), (255, 0, 0), 2)  
 roi\_gray2 = gray2[y:y + h, x:x + w]  
 roi\_color2 = img2[y:y + h, x:x + w]  
 eyes2 = eye\_cascade.detectMultiScale(roi\_gray2)  
 for (ex, ey, ew, eh) in eyes2:  
 cv2.rectangle(roi\_color2, (ex, ey), (ex + ew, ey + eh), (0, 255, 0), 2)  
 hay\_elementos\_2 = True  
 if (hay\_elementos\_2):  
 contar\_cuadros\_2 += 1  
 if (contar\_cuadros\_2 >= 30):  
 print("salir.....")  
 intentar\_salir()  
 contar\_cuadros\_2 = 0  
 else:  
 contar\_cuadros\_2 = 0  
 cv2.imshow('img2', img2)  
 k = cv2.waitKey(30) & 0xff  
 if k == 27:  
 break  
  
cap.release()  
cap2.release()  
cv2.destroyAllWindows()

## PhP

### Codigo salida\_por\_id.php

#### Explicacion codigo

Actualiza la base de datos restando una unidad a la cantidad de personas que se encuentra en un local especificado.

#### Código

<?php

$db\_user="\*\*\*\*\*\*\*\*\*\*\*\*\*";

$db\_password="\*\*\*\*\*\*\*\*\*\*\*\*";

$db\_name="\*\*\*\*\*\*\*\*\*";

$db\_server="localhost";

$conn = new mysqli($db\_server, $db\_user, $db\_password, $db\_name);

// Check connection

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

if($\_SERVER['REQUEST\_METHOD']=='GET'){

$id = $\_GET['id'];

$sql1 = "select \* from local where id = '$id'";

$r = mysqli\_query($conn,$sql1);

$result = mysqli\_fetch\_array($r);

if($result['numero\_personas'] > 0){

$sql = "UPDATE local SET numero\_personas=numero\_personas-1 WHERE id = '$id'";

if ($conn->query($sql) === TRUE) {

echo "Record updated successfully";

} else {

echo "Error updating record: " . $conn->error;

}

}

}

$conn->close();

?>

### Revisar\_informe.php

#### Explicación

Se pregunta si existe un informe de un local a una fecha especifico y se indica si en efecto existe el informe o no existe.

#### Código

<?php

$db\_user="\*\*\*\*\*\*\*\*\*\*\*\*\* ";

$db\_password="\*\*\*\*\*\*\*\*\*\*\*\*\*";

$db\_name="\*\*\*\*\*\*\*\*\* ";

$db\_server="localhost";

$conn = new mysqli($db\_server, $db\_user, $db\_password, $db\_name);

// Check connection

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

if($\_SERVER['REQUEST\_METHOD']=='GET'){

$id = $\_GET['id'];

$fecha = $\_GET['fecha'];

$sql = "select \* from informe where id = '$id' and fecha = '$fecha'";

$r = mysqli\_query($conn,$sql);

$result = mysqli\_fetch\_array($r);

if($result == null){

echo "no existe";

}else{

echo "existe";

}

}

$conn->close();

?>

### Ingreso\_por\_id.php

#### Explicación

Se pregunta si se puede ingresar una persona al local en caso de ser afirmativo se suma una unidad a la cantidad actual.

#### Código

<?php

$db\_user="\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ";

$db\_password="\*\*\*\*\*\*\*\*\*\*\*\*\*";

$db\_name="\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ";

$db\_server="localhost";

$conn = new mysqli($db\_server, $db\_user, $db\_password, $db\_name);

// Check connection

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

if($\_SERVER['REQUEST\_METHOD']=='GET'){

$id = $\_GET['id'];

$fecha = $\_GET['fecha'];

$sql1 = "select \* from local where id = '$id'";

$sql2 = "select \* from informe where id = '$id' and fecha = '$fecha'";

$r = mysqli\_query($conn,$sql1);

$result = mysqli\_fetch\_array($r);

$r2 = mysqli\_query($conn,$sql2);

$result2 = mysqli\_fetch\_array($r2);

if($result['numero\_personas'] + 1 <= $result['numero\_max']){

$sql = "UPDATE local SET numero\_personas=numero\_personas+1 WHERE id = '$id'";

$sql2 = "UPDATE informe SET conteo\_diario=conteo\_diario+1 WHERE id = '$id' and fecha = '$fecha'";

if ($conn->query($sql) === TRUE) {

echo "Record updated successfully";

} else {

echo "Error updating record: " . $conn->error;

}

if ($conn->query($sql2) === TRUE) {

echo "Record updated successfully";

} else {

echo "Error updating record: " . $conn->error;

}

}

}

$conn->close();

?>

### Crear\_informes\_vacios.php

#### Explicación

Crea un informe en la base de dato en un local y fechas especificados

#### Código

<?php

$db\_user="\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ";

$db\_password="\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

$db\_name="\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

$db\_server="localhost";

$conn = new mysqli($db\_server, $db\_user, $db\_password, $db\_name);

// Check connection

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

if($\_SERVER['REQUEST\_METHOD']=='GET'){

$id = $\_GET['id'];

$fecha = $\_GET['fecha'];

$sql = "INSERT INTO informe (fecha, conteo\_diario, id)

VALUES ('$fecha', 0 , '$id')";

if ($conn->query($sql) === TRUE) {

echo "New record created successfully";

} else {

echo "Error: " . $sql . "<br>" . $conn->error;

}

}

$conn->close();

?>

### Cantidad\_python.php

#### Explicación

Se hace una consulta preguntando si se puede ingresar a un local en especifico

#### Código

<?php

$db\_user="\*\*\*\*\*\*\*\*\*\*\* ";

$db\_password="\*\*\*\*\*\*\*\*\*\*\*\*";

$db\_name="\*\*\*\*\*\*\*\*\*\*\*\*\* ";

$db\_server="localhost";

$conn = new mysqli($db\_server, $db\_user, $db\_password, $db\_name);

// Check connection

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

if($\_SERVER['REQUEST\_METHOD']=='GET'){

$id = $\_GET['id'];

$sql1 = "select \* from local where id = '$id'";

$r = mysqli\_query($conn,$sql1);

$result = mysqli\_fetch\_array($r);

if($result['numero\_max']>$result['numero\_personas']){

echo "verdad";

}else{

echo "falso";

}

}

$conn->close();

?>

### Index.php

#### Explicación

Retorna lista de locales que existen en la base de datos en formato json

#### Código

<?php

$db\_user="\*\*\*\*\*\*\*\*\*\*\*\* ";

$db\_password="\*\*\*\*\*\*\*\*\*\*\*\*\*";

$db\_name="\*\*\*\*\*\*\*\*\*\*\*";

$db\_server="localhost";

header("Access-Control-Allow-Origin: \*");

try{

$DBcon = new PDO("mysql:host=$db\_server;dbname=$db\_name",$db\_user,$db\_password);

$DBcon->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

}catch(PDOException $ex){

die($ex->getMessage());

}

$query = "select \* from local";

$stmt = $DBcon->prepare($query);

$stmt->execute();

$userData = array();

while($row=$stmt->fetch(PDO::FETCH\_ASSOC)){

$userData[$row['id']][] = $row;

}

echo json\_encode($userData);

?>

### Get\_informes\_id.php

#### Explicación

Se recibe los informes de un local en especifico en formato json

#### Código

<?php

$db\_user="\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ";

$db\_password="\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

$db\_name="\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ";

$db\_server="localhost";

header("Access-Control-Allow-Origin: \*");

try{

$DBcon = new PDO("mysql:host=$db\_server;dbname=$db\_name",$db\_user,$db\_password);

$DBcon->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

}catch(PDOException $ex){

die($ex->getMessage());

}

if($\_SERVER['REQUEST\_METHOD']=='GET'){

$id = $\_GET['id'];

$query = "select \* from informe where id ='$id'";

$stmt = $DBcon->prepare($query);

$stmt->execute();

$userData = array();

while($row=$stmt->fetch(PDO::FETCH\_ASSOC)){

$userData[$row['id\_informe']][] = $row;

}

echo json\_encode($userData);

}

?>

# Análisis de presupuesto

|  |  |
| --- | --- |
| Articulo | Precio |
| Placa Arduino | $9.80 |
| Jumpers | $1.75 |
| Cable USB | $3.50 |
| Servomotores | $4.00 |
| Leds | $0.20 |
| Resistencias | $0.35 |
| Camaras Web | $30.0 |

# Conclusiones

* Con el aplicativo diseñado se pudo generar un informe donde se puede visualizar con ayuda de un diagrama de barras el número de personas que asistieron a un local público, con la fecha y hora de cada toma de datos para su consulta.
* Se diseño un prototipo que gestiona en número de personas que ingresan y salen de locales públicos, generando una alerta en caso de que un local no cumple con el aforo permitido.

# Referencias bibliográficas

A. Albiol, A. A. (2012). Who is who at different cameras:people re-identification using depth cameras. *IET Computer Vision, 6*(5), 378-387.

EFISAT+. (s.f.). *Contador de pasajeros.* Obtenido de http://www.efisat.net/Contadorpasajeros.html

FEVOX. (s.f.). Obtenido de integramos seguridad, control y automatización basados en el: https://www.fevox.co/

Gutiérrez, M. (2014). *Estudio de los efectos de perspectiva en contadores de personas basados en video con lentes de gran angular*. Obtenido de http://hdl.handle.net/10251/46298

M. Castrillón-Santan, J. L.-N.-S. (2014). Conteo de personas con un sensor RGBD comercial. *Revista Iberoamericana de, 11*(3), 348-357.

Team, O. C. (2020). *Open CV*. Obtenido de https://opencv.org/