|  |  |
| --- | --- |
|  | |
|  | |
| **Elaborado para:** | Código IoT |
|  |  |
| **Fecha de elaboración:** | 30 de julio de 2024 |
| **Vigencia:** | 30 días naturales |
|  |  |
| **Elaborado por:**  **Revisado por:** | |  | | --- | | Sergio Rodriguez Román  Yessica Yazmin Calderón Segura  Beatriz Elizabeth Serrano Rodríguez | |  | |  | |
|  |  |
| **Documento:** | Plan de acción del Proyecto Capstone |
|  | |

Formato Kardex

Alumnos

|  |  |
| --- | --- |
| Nombre del proyecto: | Acceso-Controlado-De-Seguridad-RFID |
| Fecha de inicio del proyecto: | Junio 2024 |
| Fecha de conclusión del proyecto: | Julio 2024 |
| Descripción: | El proyecto propuesto se centra en la implementación de un sistema de Internet de las Cosas (IoT) diseñado para el acceso controlado a estacionamientos mediante dispositivos móviles y tecnología IoT. El sistema utilizará una combinación de tecnologías avanzadas, incluyendo la mensajería MQTT, Telegram, y una tarjeta inteligente RFID para gestionar el acceso y proporcionar notificaciones en tiempo real. La base de datos MySQL se empleará para la verificación y almacenamiento de datos, garantizando seguridad tanto para los administradores del estacionamiento como para los usuarios. |
| Aplicaciones: | Sector público y privado. Control de entradas y salidas de vehículos de uso oficial de la universidad, así como de particulares. Estacionamientos universitarios de diferentes unidades académicas. Empresas, organismos y asociaciones. |
| Objetivo general: | Desarrollar un sistema IoT que automatice el acceso a estacionamientos utilizando una tarjeta inteligente RFID y que permita emplear el sistema de mensajería MQTT y Telegram para notificar a los usuarios sobre el ingreso y salida del estacionamiento y alertar a la caseta de vigilancia acerca de los usuarios registrados y no registrados para mejorar la seguridad y eficiencia en la gestión en el acceso de los automóviles a los estacionamientos de la universidad. |
| Objetivos específicos: | **Sociales:**  **Objetivo 1. Proporcionar un entorno más seguro:** La implementación de vigilancia automatizada en la entrada y salida de vehículos contribuirá a un entorno más seguro para los universitarios y usuarios del estacionamiento. Esto reducirá el riesgo de incidentes relacionados con la seguridad y mejorará la protección general del área.  **Industriales:**  **Objetivo 2. Reducir el tiempo y esfuerzo del personal de vigilancia:** La automatización del registro y monitoreo de vehículos permitirá reducir significativamente el tiempo y esfuerzo requerido del personal de vigilancia. Esto optimiza los recursos humanos y permite una gestión más eficiente del estacionamiento.  **Técnicos:**  **Objetivo 3. Integrar un sistema de control de acceso eficiente:** Utilizando mensajería MQTT y Telegram, el sistema notificará a los usuarios sobre el ingreso y salida del estacionamiento y alertará a la caseta de vigilancia sobre los usuarios registrados y no registrados. La verificación de datos mediante tarjetas inteligentes RFID y el uso de una base de datos MySQL permitirán una gestión y almacenamiento eficiente de la información, asegurando la integridad y disponibilidad de los datos.  **Objetivo 4. Desarrollar un sistema de registro rápido y eficiente:** La integración de una tarjeta inteligente RFID permitirá un registro rápido y preciso de los vehículos, mientras que el sistema de confirmación de acceso y salida facilitará la experiencia del usuario, haciéndola más ágil y libre de inconvenientes. |
| Justificación: | En el contexto de los estacionamientos universitarios y corporativos, el control eficiente de acceso es crucial para garantizar la seguridad y optimizar la gestión del espacio. Los sistemas actuales suelen presentar limitaciones en seguridad, eficiencia y facilidad de uso. La implementación de un sistema IoT avanzado ofrece una solución integral que aborda estos desafíos mediante la automatización y la integración de tecnologías modernas. |
| Integrantes del equipo: | Sergio Rodríguez Román |
|  | Yessica Yazmin Calderón Segura |
|  | Beatriz Elizabeth Serrano Rodríguez |
| Validado por: | <Nombre de facilitador Código IoT> |
| Contenido Temático: | Temario.   1. Introducción 2. Principio de funcionamiento 3. Material necesario 4. Herramientas computacionales 5. Circuito recomendado 6. Lecturas de sensor y Envío de información 7. Recepción de información 8. Almacenamiento de información 9. Visualización de datos y Panel de control 10. Automatización 11. Utilización |
| Productos: | Sistema de IoT con submódulos de:  - Lectura de tarjeta inteligente RFID  - Envió de notificación al propietario por Telegram (Bot)  - Flujo del proceso en node red  - Maqueta de funcionamiento  - Base de datos de usuarios  - Código para la lectura de tarjeta RFID y movimiento de pluma de acceso vehicular. |
| Alcances: | El sistema está diseñado para ser implementado en estacionamientos universitarios de diferentes unidades académicas, así como en empresas, organismos y asociaciones que requieran un control riguroso del acceso a sus instalaciones. El enfoque en el sector público y privado asegura que el sistema pueda adaptarse a diversas necesidades y contextos. |
| Requisitos: | Contar con los conocimientos en programación en Python, Arduino, Node-red, base de datos MySQL y electrónica para configurar el circuito y el prototipo. |
| Software: | MySQL, Node-RED version: v3.0.2, Node.js version: v16.20.0, Computadora con virtualización de Linux 5.15.0-73-generic x64 LE, MQTT 5.0 y Python 3.8.8 |
| Hardware: | Raspberry Pi: Serie de ordenadores monoplaca u ordenadores de placa de bajo costo.  Sensores: RFID  Actuadores: Servo motor  Circuito: Protoboard |

**1. Introducción**

En el contexto actual, la seguridad y la eficiencia en la gestión de accesos a instalaciones se han convertido en aspectos fundamentales tanto en el ámbito público como en el privado. Con el avance de las tecnologías de la información y la comunicación, la implementación de sistemas de acceso controlado ha evolucionado significativamente, integrando tecnologías avanzadas que facilitan la administración y aumentan la seguridad. En este sentido, el proyecto propuesto se centra en la implementación de un sistema de Internet de las Cosas (IoT) diseñado para el acceso controlado a estacionamientos, utilizando dispositivos móviles y tecnología RFID (Radio Frequency Identification).

El sistema planteado busca combinar múltiples tecnologías innovadoras para ofrecer una solución robusta y eficiente. La mensajería MQTT (Message Queuing Telemetry Transport) y la plataforma de comunicación Telegram serán utilizadas para proporcionar notificaciones en tiempo real a los usuarios sobre el ingreso y salida del estacionamiento. Además, la verificación de datos se realizará mediante tarjetas inteligentes RFID, y una base de datos MySQL se empleará para almacenar y gestionar la información de manera segura. Este enfoque garantizará la integridad y disponibilidad de los datos, brindando seguridad tanto para los administradores del estacionamiento como para los usuarios.

La implementación de este sistema tiene aplicaciones en diversos sectores, incluidos los estacionamientos universitarios, empresas, organismos y asociaciones. En el ámbito universitario, por ejemplo, el control de entradas y salidas de vehículos de uso oficial y particular se beneficiará enormemente de la automatización, mejorando la seguridad y optimizando la gestión del espacio. De igual manera, en el sector corporativo, la implementación de un sistema IoT para el control de acceso puede significar una reducción considerable en el tiempo y esfuerzo del personal de vigilancia, permitiendo una gestión más eficiente y segura.

Desde una perspectiva social, uno de los objetivos principales es proporcionar un entorno más seguro para los usuarios, reduciendo el riesgo de incidentes y mejorando la protección general del área. Desde el punto de vista industrial, se busca optimizar los recursos humanos mediante la automatización del registro y monitoreo de vehículos. Técnicamente, el proyecto pretende integrar un sistema de control de acceso eficiente y desarrollar un sistema de registro rápido y preciso, facilitando una experiencia de usuario más ágil y libre de inconvenientes.



Imagen. Prototipo de Acceso-Controlado-De-Seguridad-RFID

**2. Principio de funcionamiento**

El sistema de acceso controlado de seguridad RFID se basa en la integración de tecnologías avanzadas para garantizar un control eficiente y seguro del acceso a estacionamientos. A continuación, se detalla el principio de funcionamiento del sistema, el modo de lectura del sensor RFID, y el circuito de acondicionamiento de señal.

El sistema emplea una base de datos de registro de usuarios a la que se asocian tarjetas inteligentes RFID. Cada usuario, ya sea docente, administrativo, estudiante o invitado, tiene asignada una tarjeta RFID única. El funcionamiento del sistema sigue estos pasos:

1. **Lectura de la Tarjeta RFID**: Al acercar la tarjeta RFID al lector, el sistema lee la información almacenada en la tarjeta.
2. **Verificación en la Base de Datos**: La información leída se envía a la base de datos MySQL, donde se verifica si la tarjeta está registrada y vinculada a un usuario.
3. **Procesamiento de Datos**: Utilizando salas de mensajería MQTT, la información verificada se procesa para determinar si el acceso es autorizado.
4. **Notificación al Vigilante**: Un bot de Telegram notifica al vigilante sobre el estado del acceso (autorizado o no autorizado). El vigilante recibe la notificación en su dispositivo móvil, pudiendo ver detalles como la identidad del usuario, y la fecha y hora de entrada o salida.
5. **Acciones de Seguridad**: Además de notificar al vigilante, el sistema puede realizar otras acciones de seguridad, como registrar y monitorear la información del usuario para mantener un historial de accesos.
6. **Modo de Lectura del Sensor RFID**

El sensor RFID utilizado en el sistema es un dispositivo digital que comunica la información de las tarjetas RFID a través de una interfaz de comunicación serial, generalmente UART (Universal Asynchronous Receiver-Transmitter). El proceso de lectura incluye:

* **Detección de Tarjeta**: Cuando una tarjeta RFID se acerca al lector, el sensor detecta la presencia de la tarjeta mediante señales electromagnéticas.
* **Transmisión de Datos**: La información contenida en la tarjeta se transmite al lector, que a su vez la envía al microcontrolador del sistema.
* **Decodificación y Verificación**: El microcontrolador decodifica los datos y los envía a la base de datos para su verificación.

1. **Circuito de Acondicionamiento de Señal**

El circuito de acondicionamiento de señal del sensor RFID incluye:

* **Antena del Lector**: La antena genera el campo electromagnético necesario para activar y leer la tarjeta RFID.
* **Módulo de Lectura**: El módulo de lectura recibe las señales de la tarjeta RFID, las convierte en datos digitales, y las transmite al microcontrolador.
* **Microcontrolador**: El microcontrolador procesa los datos recibidos, los verifica en la base de datos y maneja la comunicación con los sistemas MQTT y Telegram.

1. **Capacidades y Limitaciones**

* **Capacidades**:
  + **Lectura Rápida y Precisa**: El sensor RFID puede leer la información de las tarjetas rápidamente y con alta precisión.
  + **Comunicación Eficiente**: Utiliza MQTT para la transmisión rápida de datos y Telegram para notificaciones en tiempo real.
  + **Seguridad y Gestión de Datos**: La base de datos MySQL garantiza una gestión segura y eficiente de la información.
* **Limitaciones**:
  + **Rango de Lectura**: La distancia efectiva de lectura del sensor RFID está limitada a unos pocos centímetros.
  + **Interferencias Electromagnéticas**: Puede haber interferencias en entornos con alta densidad de dispositivos electrónicos.
  + **Dependencia de la Conectividad**: El sistema depende de una conexión estable a la red para la transmisión de datos y notificaciones.

**3. Material necesario**

|  |
| --- |
| Software: MySQL, Node-RED version: v3.0.2, Node.js version: v16.20.0, Computadora con virtualización de Linux 5.15.0-73-generic x64 LE, MQTT 5.0 y Python 3.8.8  Hardware: |
| Raspberry Pi: Serie de ordenadores monoplaca u ordenadores de placa de bajo costo. Sensores: RFID , Actuadores: Servo motor y Circuito: Protoboard   |  |  |  | | --- | --- | --- | | Sensor RFID MFRC522 RF IC | GPIO Extension Board | Protoboard | | Leds | Jumpers | Resistencias 220 Ohms | | Tarjeta RFID | Servomotor con Arduino |  |   **4. Herramientas computacionales**  Para este proyecto, se necesita una computadora con sistema operativo Linux 5.15.0-73-generic x64 LE. Esta computadora debe tener instalados Python y Node-RED. Además, es necesario contar con el IDE de Arduino para cargar y ejecutar el programa, lo que implica la configuración de una Raspberry Pi. También se requiere MySQL para la gestión de bases de datos, Node-RED para la creación de flujos de trabajo, MQTT para la transmisión de datos, y la IDE Visual Studio Code para el desarrollo de software. El manejo de repositorios se realizará a través de GitHub.  **5. Circuito recomendado**  Para montar los circuitos necesarios para este proyecto, se recomienda el uso del software Fritzing. Este programa permite modelar los circuitos en una protoboard virtual, facilitando la visualización y conexión de los diferentes componentes utilizados en el proyecto. A continuación, se presenta cómo se conectan los dispositivos empleados en este sistema. |

Fuente. <https://edu.codigoiot.com/mod/lesson/view.php?id=2089&pageid=2579>

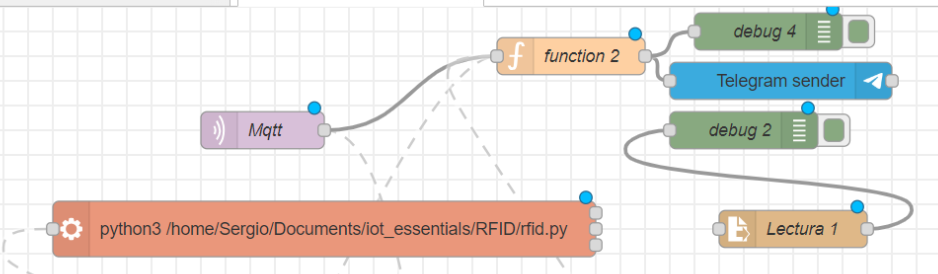
**6. Lecturas de sensor y Envío de información**

Como se ha mencionado anteriormente, el envío de la información es principalmente por MQTT. MQTT son las siglas de Message Queuing Telemetry Transport. Se trata de un protocolo de mensajería ligero para usar en casos de clientes que necesitan enviar datos, que están conectados a redes no fiables o con recursos limitados en cuanto al ancho de banda. Se utiliza principalmente para comunicaciones de máquina a máquina (M2M) o conexiones del tipo de Internet de las cosas.

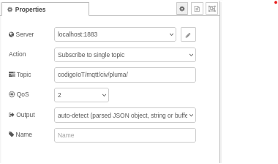
Para poder enviar información, y dar apertura de pluma utilizamos dos diferentes salas para cada evento:

1. codigoIoT/mqtt/civ/pluma

Ejemplo de como se ven los nodos en Node-Red.



La configuración para que funcione MQTT es estándar y se realiza sobre la Raspberry Pi o un servidor para que a través de los nodos de Node-red se recuperé esa información y se ejecuten diversos eventos de acuerdo con los datos procesados.

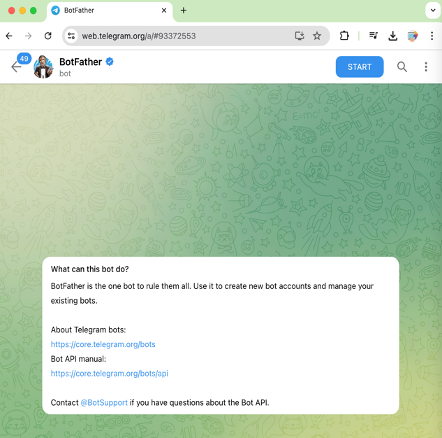


Por otro lado, tenemos la comunicación con el propietario del vehículo para ejecutar ciertas acciones. Para esto se desarrollo un Bot de Telegram, para poder recibir comandos que el sistema pueda ejecutar. A continuación se muestra el procedimiento para generar esta aplicación.

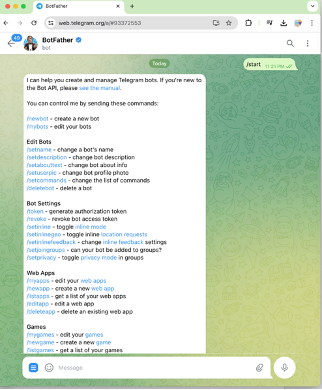
**Telegram y NodeRed. Envía y recibe mensajes en Telegrama**

Objetivo: Integrar directemente telegram dento de NodeRed

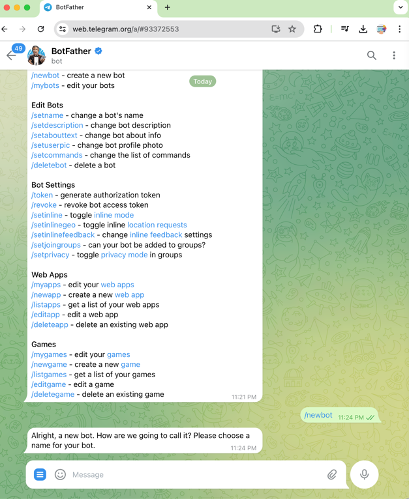
1. Crear un bot (es el encargado de recibir y enviar mensajes dentro de NodeRed).
2. Para crear un bot, se abre BotFather y se da clic en iniciar (start)



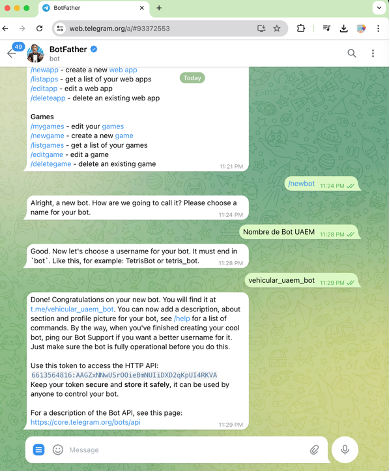
1. Se abren las siguientes opciones:



1. La opción que se tiene que crear es /newbot



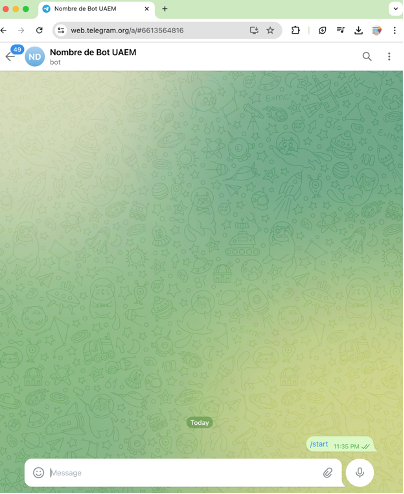
1. Y le especificamos un nombre para el bot: Nombre del bot, username y token



1. Después abre el enlace dl bot creado [t.me/vehicular\_uaem\_bot](https://t.me/vehicular_uaem_bot), y para iniciar la conversación con el bot da clic en star



1. Se da clic en iniciar (start) para abrir conversación.



**Instalar NODE\_RED**

Se descarga e instala <https://nodejs.org/en>

<https://apuntes.de/nodejs/instalacion-en-mac/#gsc.tab=0>

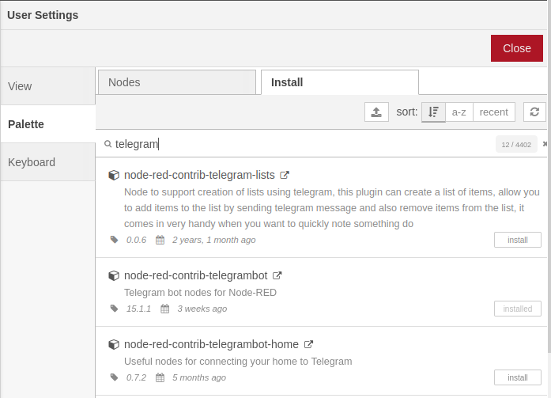
Node.js v20.15.1 to /usr/local/bin/node

npm v10.7.0 to /usr/local/bin/npm

Make sure that /usr/local/bin is in your $PATH.

**¿Cómo descargar el nodo de Telegram?**

Dentro del entorno de Node Red debemos descargar desde el manage palette el nodo “node-red-contrib-telegrambot” el cual contiene un nodo receptor y un nodo emisor que actúan como un bot de Telegram mediante un token generado por @botfather.



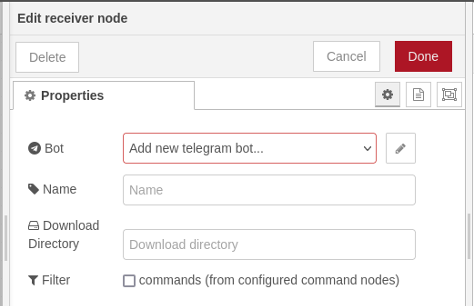
Ya que se haya instalado correctamente el paquete de nodo de Telegram nos aparecerá en el costado izquierdo donde se encuentran todos los nodos disponibles que tenemos.



**¿Como configurar un bot de telegram desde Node Red?**

Ya instalados los nodos de telegram lo próximo por hacer será configurar los nodos receptores y emisor para que puedan tener conexión con nuestro bot anteriormente creado, para esto lo que haremos será añadir en bot a nuestro Node red de la siguiente manera;

* Selecciona un nodo receiver y daremos doble clic sobre él, después nos abrirá un panel en el que vamos a agregar nuestro bot creado.



* Da clic en el incono de editar, nos desplegará un formulario en el cual agregaremos el nombre de nuestro bot, así como el token generado por el @botfather que será nuestra contraseña de acceso al bot.



**7. Recepción de información**

La recepción de información sucede en dos canales, primero la información procesada se transmite a través de MQTT. Si el reconocimiento de la placa se realiza de una forma correcta, se recupera el idChat registrado en la base de datos para que posteriormente se le envíe una notificación al usuario, donde puede autorizar la entrada al estacionamiento. Esto genera que se publique la información en la sala con la autorización del usuario en la sala (codigo/IoT/mqt/civ/pluma) de apertura de pluma. Solamente pueden recibir valores True o False (Apertura de pluma y cierre de pluma, respectivamente). Pero en el registro de los usuarios se realiza a través de la comunicación por medio del Bot de Telegram. A continuación, mostraremos como sr recibe la información para el registro de los usuarios.

***Código Autorización y apertura de pluma***

A continuación se muestra código que cambia de estado la pluma para acceder o salir del estacionamiento.

import RPi.GPIO as GPIO

import mysql.connector

import random

import time

from paho.mqtt import client as mqtt\_client

import json

from mfrc522 import SimpleMFRC522

# Conexión a la base de datos

conexion = mysql.connector.connect(host="localhost", user="SergioDB", passwd="SergioSQL", database="autoIOT")

cursor = conexion.cursor()

# Variables y constantes

broker = '127.0.0.1'

port = 1883

topic = "codigoIoT/ejemplo/ID"

client\_id = f'python-mqtt-{random.randint(0, 1000)}'

ledVerde = 7

ledRojo = 37

servoPIN1 = 11

servoPIN2 = 12

GPIO.setmode(GPIO.BOARD)

GPIO.setup(ledVerde, GPIO.OUT)

GPIO.output(ledVerde, GPIO.LOW)

GPIO.setup(ledRojo, GPIO.OUT)

GPIO.output(ledRojo, GPIO.LOW)

GPIO.setup(servoPIN1, GPIO.OUT)

GPIO.setup(servoPIN2, GPIO.OUT)

reader = SimpleMFRC522()

# Conexión al Broker MQTT

def connect\_mqtt():

def on\_connect(client, userdata, flags, rc):

if rc == 0:

print("Connected to MQTT Broker!")

else:

print("Failed to connect, return code %d\n", rc)

client = mqtt\_client.Client(client\_id)

client.on\_connect = on\_connect

client.connect(broker, port)

return client

# Publicador

def publish2(client, mensaje):

msg = mensaje

result = client.publish(topic, msg)

status = result[0]

if status == 0:

print(f"Send `{msg}` to topic `{topic}`")

else:

print(f"Failed to send message to topic {topic}")

client = connect\_mqtt()

client.loop\_start()

p1 = GPIO.PWM(servoPIN1, 50) # GPIO 11 for PWM with 50Hz

p1.start(0) # Initialization

p2 = GPIO.PWM(servoPIN2, 50) # GPIO 12 for PWM with 50Hz

p2.start(0)

try:

while True:

print("---------Leyendo Tarjeta:--------------")

id, text = reader.read()

print(f"ID leido: {id}")#

cursor.execute("SELECT \* FROM Registro\_personal WHERE id\_registro = "+str(id)+"")

Datos = cursor.fetchone()

print("Datos obtenidos", Datos)

if Datos:

print("Estas autorizado")

acceso = "Autorizado"

#GPIO.output(ledVerde, GPIO.HIGH)

#time.sleep(5)

#p.ChangeDutyCycle(5)

#time.sleep(5)

#p.ChangeDutyCycle(8)

#GPIO.output(ledVerde, GPIO.LOW)#

cursor.execute("SELECT Hora\_salida FROM Bitacora WHERE id\_registro = "+str(id)+" AND Hora\_salida IS NULL")

fila2 = cursor.fetchone()

print(f"Hora de salida encontrada: {fila2}")#

if fila2==None:

print("si")

cursor.execute("INSERT INTO Bitacora (id\_registro, Hora\_entrada, Fecha) VALUES ("+str(id)+", CURRENT\_TIME(),CURRENT\_DATE());")

print("Se inserto dato")

conexion.commit()

GPIO.output(ledVerde, GPIO.HIGH)

time.sleep(5)

p1.ChangeDutyCycle(5)

time.sleep(5)

p1.ChangeDutyCycle(8)

GPIO.output(ledVerde, GPIO.LOW)

print("Enviando mensaje")

obj = {"id\_Registro": id, "Acceso": acceso, "datos": str(Datos)}

json\_view = json.dumps(obj, indent=1)

publish2(client, json\_view)

print("mensaje Enviado!")

elif fila2[0]!=None:

print("si")

cursor.execute("INSERT INTO Bitacora (id\_registro, Hora\_entrada, Fecha) VALUES ("+str(id)+", CURRENT\_TIME(),CURRENT\_DATE());")

print("Se inserto dato")

conexion.commit()

GPIO.output(ledVerde, GPIO.HIGH)

time.sleep(5)

p1.ChangeDutyCycle(5)

time.sleep(5)

p1.ChangeDutyCycle(8)

GPIO.output(ledVerde, GPIO.LOW)

print("Enviando mensaje")

obj = {"id\_Registro": id, "Acceso": acceso, "datos": str(Datos)}

json\_view = json.dumps(obj, indent=1)

publish2(client, json\_view)

print("mensaje Enviado!")

else:

print("UPDATE Bitacora SET Hora\_salida = CURRENT\_TIME() WHERE id\_registro = "+str(id)+" AND Hora\_salida IS NULL")

cursor.execute("UPDATE Bitacora SET Hora\_salida = CURRENT\_TIME() WHERE id\_registro = "+str(id)+" AND Hora\_salida IS NULL;")

print("Se actualizo dato")

conexion.commit()

GPIO.output(ledVerde, GPIO.HIGH)

time.sleep(5)

p2.ChangeDutyCycle(5)

time.sleep(5)

p2.ChangeDutyCycle(8)

GPIO.output(ledVerde, GPIO.LOW)#

print("Enviando mensaje")

obj = {"id\_Registro": id, "Acceso": acceso, "datos": str(Datos)}

json\_view = json.dumps(obj, indent=1)

publish2(client, json\_view)

print("mensaje Enviado!")

else:

print("Acceso Denegado")

acceso = "Denegado"

GPIO.output(ledRojo, GPIO.HIGH)

time.sleep(5)

GPIO.output(ledRojo, GPIO.LOW)

with open("datos.txt", "w") as archi1:

archi1.write(str(id))

except KeyboardInterrupt:

GPIO.cleanup()

exit()

finally:

if conexion.is\_connected():

cursor.close()

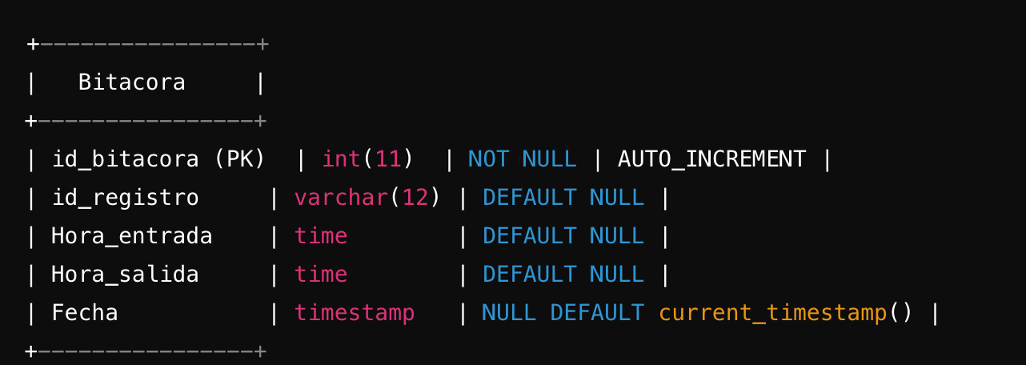
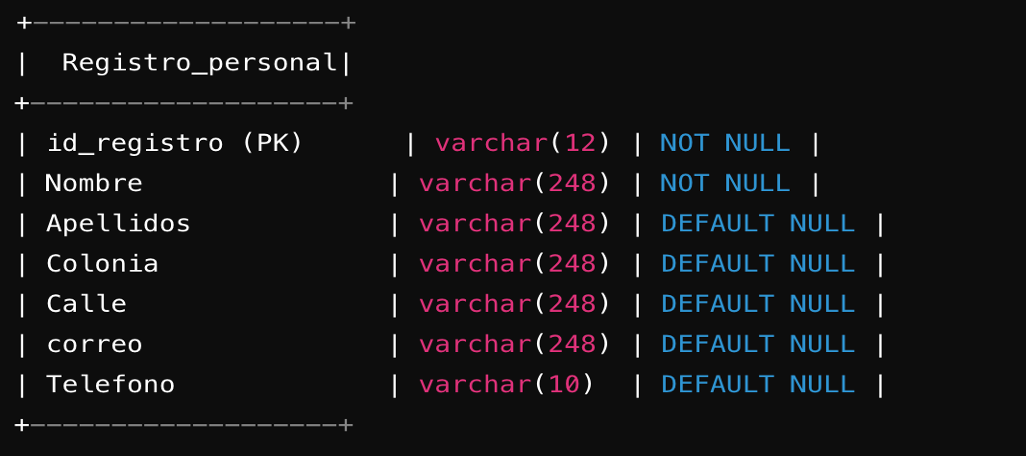
conexion.close()

**8. Almacenamiento de información**

***Almacenamiento MySQL***

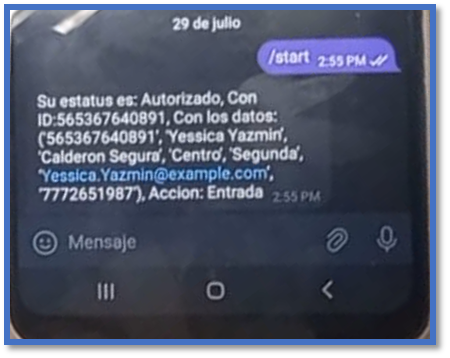
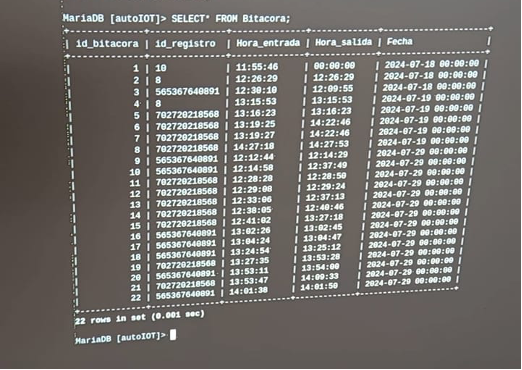
Para el sistema utilizamos un servidor MySQL en la cual estará montada toda nuestra base de datos. Nuestra BD tendrá la siguiente estructura, la cual constara de 2 tablas principales conectadas entre si con ayuda de las llaves primarias y las llaves foráneas, estas tablas serán:

* ***Tabla Registro\_personal.*** La cual contiene la información de los usuarios que estén registrados y que tendrán acceso a los estacionamientos.
* ***Tabla Bitacora*** Contiene todos los registros de las entradas y salidas al estacionamiento, con fecha y hora de entrada y salida.



**9. Visualización y Panel de Control**

Con la base de datos, es posible realizar diversas consultas, como obtener los datos del usuario y la identificación de la tarjeta inteligente RFID. Esto permite enviar una notificación al vigilante, informándole si el acceso está autorizado o no, y activar la barrera del estacionamiento para permitir la entrada o salida. Esta interactividad brinda al usuario la capacidad de ejecutar una acción. En la comunicación mediante Telegram, ocurren dos situaciones. Primero, el vigilante recibe un comando de autorización (/autoriza) que valida y se levanta la barrera de mkanera automatica. Si el comando es de no autorización (/noautoriza), el vigilante recibe un mensaje indicando que la tarjeta inteligente RFID no está registrada, lo cual permite un monitoreo en tiempo real desde un dispositivo móvil.



**10. Automatización**

En este proyecto, la automatización se enfoca principalmente en el control de acceso de usuarios que ingresan a estacionamientos. La implementación de un sistema automatizado mediante tarjetas inteligentes RFID no solo mejora la seguridad, sino que también optimiza la gestión del acceso. Por ejemplo, el acceso con una tarjeta implica un costo por su creación, y en caso de pérdida, se debe crear una nueva con la información del usuario.

Para simplificar este proceso y reducir costos, se propone el uso de una plataforma accesible a través de Telegram. Dado que la mayoría de las personas actualmente poseen un dispositivo celular, Telegram se presenta como una solución eficiente y universal. A través de esta plataforma, se pueden enviar notificaciones en tiempo real sobre el estado de acceso autorizado o no autorizado, facilitando la información al personal de vigilancia.

Además, la automatización permite proyectar un trabajo a futuro para extiender un monitoreo y control del acceso Esta integración de tecnologías no solo mejora la eficiencia operativa, sino que también proporciona una experiencia de usuario más segura y conveniente.

**11. Utilización**

La utilización del sistema propuesto permite a los usuarios de vehículos acceder automáticamente a los estacionamientos mediante un monitoreo eficiente a través de Telegram. A continuación, se describen los pasos para utilizar el sistema:

1. **Detección de la Tarjeta Inteligente RFID**: El usuario se acerca al estacionamiento y el sistema detecta la tarjeta inteligente RFID correspondiente a su vehículo.
2. **Verificación en la Base de Datos**: Una vez detectada la tarjeta, el sistema verifica si está registrada en la base de datos. Si la tarjeta está registrada, el sistema recupera la información del usuario asociada a la tarjeta. Esta información es enviada al bot de Telegram, el cual notifica al vigilante sobre el intento de acceso. El vigilante recibe dos opciones de respuesta: autorizado o no autorizado.
3. **Autorización del Acceso**:
   * Si el vigilante recibe el comando /autorizar en Telegram, se publica un valor de True en la sala correspondiente (código/IoT/mqtt/pluma/), lo que permite ejecutar la acción de abrir la barrera del estacionamiento mediante un servo motor de manera automática.
   * Si el vigilante recibe con el comando /noautorizar, el sistema envía una notificación indicando que el acceso no está autorizado. Por ejemplo, si una persona no registrada intenta entrar, se alerta al vigilante, asegurando un control riguroso y confiable del acceso.
4. Este proceso automatizado no solo optimiza el flujo de vehículos en el estacionamiento, sino que también garantiza una mayor seguridad. Al utilizar Telegram como medio de comunicación, se facilita la interacción y monitoreo en tiempo real, ofreciendo una solución accesible y eficiente tanto para los usuarios como para el personal de vigilancia. La implementación de este sistema proporciona una experiencia de acceso controlado más segura y conveniente, al tiempo que reduce el esfuerzo manual requerido para gestionar el ingreso y salida de vehículos.

