

# GAS BUSTERS

WRO 2023

## DESCRIPCIÓN BREVE

Nuestro objetivo es desarrollar una solución IoT para la Autoridad del Canal de Panamá (ACP) que ayude a que cumpla con el compromiso que adquirió en la Conferencia de las Naciones Unidas sobre el Cambio Climático (COP26) para convertir el Canal de Panamá en un “corredor verde” para el comercio marítimo.

## INTEGRANTES

**Stephen Pérez**

**Isabella González**

## PRESENTACIÓN DEL EQUIPO

¿Quiénes forman parte del equipo?

Los integrantes y representantes del equipo son:

**Stephen Pérez**

**Isabella González**



¿De dónde son?

Somos estudiantes del Instituto Profesional Técnico e Industrial de Aguadulce ubicado en el Distrito de Aguadulce de la provincia de Coclé.

¿Cómo se han dividido las tareas dentro del equipo?

Para los efectos de tener éxito en el desarrollo del proyecto, el mismo se a distribuido de la siguiente forma:

**Stephen Pérez:** se encargó del ensamblaje del carro, la programación del carro y de los sensores

**Isabella González:** se encargó de ensamblar el carro robot y de coordinar la elaboración del informe.

## **IDEA RESUMIDA DEL PROYECTO**

### **¿Cuál es el problema que resuelve su proyecto y por qué lo ha elegido?**

El problema consiste en que La Autoridad del Canal de Panamá (ACP) adquirió el compromiso que adquirió en la Conferencia de las Naciones Unidas sobre el Cambio Climático (COP26) para convertir el Canal de Panamá en un “corredor verde” para el comercio marítimo.

Como equipo hemos escogido este proyecto para contribuir con la salud del medio ambiente y tener una mejor calidad de vida.

### **¿Cómo resolverá la solución robótica el problema que ha establecido?**

La ACP en 2021, lanzó una plataforma donde publica mensualmente información sobre la contaminación producida en general por los barcos que transitan por el Canal de Panamá, en comparación con los que utilizan otras rutas marítimas.


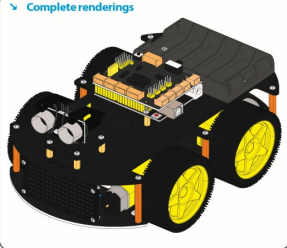




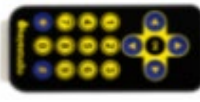
Nuestro objetivo es desarrollar una solución IoT para la Autoridad del Canal de Panamá (ACP) que ayude a que cumpla con el compromiso que adquirió en la Conferencia de las Naciones Unidas sobre el Cambio Climático (COP26) para convertir el Canal de Panamá en un “corredor verde” para el comercio marítimo.

En ese sentido nuestra solución robótica contribuye a que se pueda tener un mayor conocimiento sobre la calidad del aire y de cuanto afecta al Cambio Climático.

Nuestro proyecto ayuda a medir el nivel de partículas contaminantes emitidas en los gases expulsados por las embarcaciones que transitan por el canal de Panamá.

## ¿Cuál es el valor de su solución robótica?

El valor estipulado para este proyecto es de **B/. 385.00**

Nombre de los artículos	Modelo	Costo
<b>VEHÍCULO</b>	 	<b>B/. 150.00</b>
<b>Sensor de partículas en el aire</b>  <b>Dos unidades de Raspberry pi 4</b>	 <p style="text-align: right;"><i>Sensores</i></p>   <p style="text-align: right;"><i>Computador</i></p>	<b>B/. 200.00</b>
<b>Motor (With Welding Wire)</b>		<b>B/. 25.00</b>
<b>Remote Control</b>		<b>B/. 10.00</b>
<b>Costo Total Aproximado para implementación del proyecto</b>		<b>B/. 385.00</b>

### **¿Qué pasaría si se utilizara en la vida real?**

Se podría determinar e identificar los niveles de contaminación del aire circundante a cada embarcación que navega por la ruta transoceánica del canal de Panamá, contribuyendo de esta forma con la ACP, en aras de alcanzar los objetivos de la (COP26), mediante una base de datos en tiempo real, la cual mostraría gráficamente el comportamiento y la salud del aire en los alrededores del barco, abordado por el practico del canal y el ROBOT.

### **¿Por qué es importante su proyecto?**

Nuestro proyecto es importante porque no solo ayuda a medir el nivel de contaminación que introducen al medio ambiente los barcos que transitan por el Canal de Panamá, si no que permite tomar medidas particulares sobre cada uno de los barcos y esa información, será enviada al sistema de mando del Canal de Panamá llevando un registro del nivel de contaminación de cada embarcación.

## **PRESENTACIÓN DE SU SOLUCIÓN ROBÓTICA**

### **¿Cómo se le ocurrió esta idea? ¿Qué otras ideas has investigado?**

Se nos ocurrió esta idea debido a que gracias a los gases contaminantes emitidos por los barcos que atraviesan el canal de Panamá y afectan la calidad del aire, se han estado registrando deficiencia en la salud de la vegetación de las áreas canaleras, de tal forma que algunas especies animales han presentado cuadros infeccioso a nivel pulmonar, otras razón de porque estamos presentando esta solución robótica, es que en múltiples ocasiones el administrador de la ACP ha manifestado en diversos foros el anuncio de la grave problemática de la contaminación en el Canal de Panamá. También hemos investigado que, para algunos científicos, son la siguiente es inevitable evitar la amenaza existencial del caos climático. Para otros, ni siquiera se debería hablar de ellos.

Las tecnologías de eliminación de dióxido de carbono (CDR), que proporcionan un medio para extraer carbono de la atmósfera, son una de las áreas más candentes de la investigación climática, pero también la más controvertida.

El debate sobre si y cómo desarrollar CDR ha sido encendido por el lanzamiento el mes pasado de la sección final de la revisión integral de la ciencia del clima por parte del Panel Intergubernamental sobre el Cambio Climático (IPCC). El informe encontró que las formas de capturar y almacenar dióxido de carbono, aunque costosas, podrían desempeñar un papel en el intento de mantener las temperaturas globales dentro de límites seguros.

Pero los científicos y los políticos están divididos. Algunos dicen que la tecnología debe ser la prioridad inmediata para la investigación. Otros instan a la cautela y advierten contra confiar en tecnología no probada antes de que hayamos implementado por completo las tecnologías confiables bajas en carbono, como la

energía renovable, que ya tenemos.

### ¿Ha encontrado ideas similares disponibles?

Actualmente hemos logrado encontrar los siguientes hallazgos, La ACP en 2021, lanzó una plataforma donde publica mensualmente información sobre la contaminación producida en general por los barcos que transitan por el Canal de Panamá, en comparación con los que utilizan otras rutas marítimas.

### ¿Qué tiene de diferente su propuesta?

La solución “**Gas Busters-Monitor**” propuesta en este proyecto es mejor porque no mide en general cuál ruta marítima contamina menos con respecto a otras; si no que ofrece una medida individual del nivel de contaminación que produce cada barco que transita por el Canal de Panamá, en el tiempo. Con esta solución la ACP estaría en capacidad de levantar una señal de alerta sobre los barcos que tienen una tendencia en el tiempo de aumentar sus niveles de contaminación ambiental, para que se tomen las medidas correctivas necesarias de esta formas nos permite estimar las emisiones de gases procedentes de combustibles de los diferentes barcos que transitan en el Canal de Panamá y no solo por ruta como lo hace actualmente la ACP, además de obtener el dato nos ayuda para poder reflexionar sobre los puntos donde hay que actuar para reducir las emisiones de los gases contaminantes.

La innovación radica en que despertará la conciencia de los dueños de navieras y destacando lo contaminante que puede ser la emisión de Dióxido de Carbono y que debemos hacer algo al respecto para detener la contaminación de este gas.

## ASPECTOS TÉCNICOS

### Describir la construcción mecánica de la solución

Nuestra solución robótica está constituida por dos etapas:

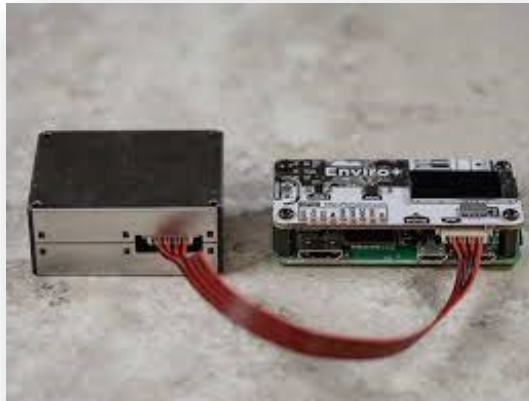
#### PRIMERA ETAPA:

Está conformada por un robot llamado Gas Buster, este robot estará en la cubierta de los barcos donde se movilizará, el corazón de este robot lo constituye una **Computadora Raspberry pi 3** y un escudo controlador de motores (Motor Drive Shield).



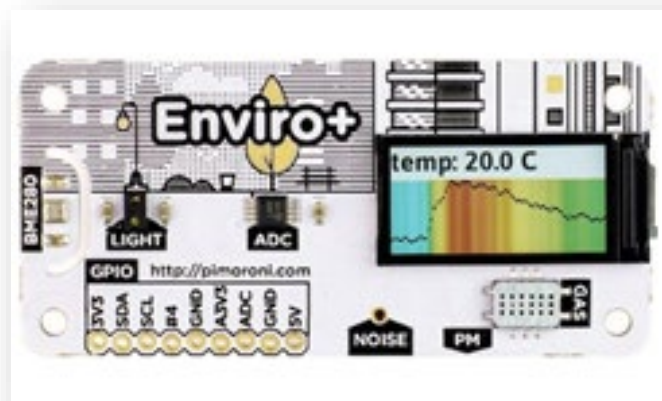
## **SEGUNDA ETAPA:**

Esta etapa está constituida por un dispositivo portátil, económico, programable, con capacidad de almacenamiento de datos y que esos datos se puedan visualizar gráficamente; al que hemos llamado “**Monitor Gas Buster**”.



Este dispositivo se encarga recolectar las partículas de aire y gases contaminantes que desechan los barcos que transitan por el canal, luego estas mediciones las compara con los valores normales en condiciones de no contaminación del aire del Canal de Panamá, estos datos son enviados y procesados en una computadora Raspberry pi 4, la cual presentamos a continuación, la cual comunicación Wi-Fi y Bluetooth.

### **Placa de monitoreo ambiental: Enviro+**



### **Esta placa incluye:**

sensores de: temperatura, presión atmosférica, humedad relativa, intensidad de luz, monóxido de carbono (reductor), dióxido de nitrógeno (oxidante), amoníaco (NH<sub>3</sub>), hidrógeno, etanol e hidrocarburos, ruido (contaminación acústica)

sensor de: partículas en el aire (pm1.0, pm2.5, pm10, polvo, polen, esporas de moho).



## Tablas de especificaciones de estándares de medición y valore de partículas en el aire.

ESTADO DE CALIDAD DEL AIRE	PM2.5	PM10
Muy Bueno	0 a 12.0	0 a 20.0
Bueno	12.1 a 35.4	21.0 a 35.0
Regular	35.5 a 55.4	36.0 a 50.0
Malo	55.5 a 150.4	51.0 a 100.0
Muy Malo	150.5 a 205.4	110.0 a 1200
Peligroso	250.5 a 500.4	1200+

**El resultado de la línea "PM2.5"**  
son las partículas producto de  
combustión, de compuestos  
orgánicos y de metales que hay en  
el aire.

**El resultado de la línea "PM10"**  
son las partículas de polvo, polen,  
y esporas de moho que hay en el  
aire.

Estado de calidad del aire	SO <sub>2</sub>	NO <sub>2</sub>	O <sub>3</sub>	PM10	PM2,5
MUY BUENO	0-100 µg/m <sup>3</sup>	0-40 µg/m <sup>3</sup>	0-80 µg/m <sup>3</sup>	0-20 µg/m <sup>3</sup>	0-10 µg/m <sup>3</sup>
BUENO	101-200 µg/m <sup>3</sup>	41-100 µg/m <sup>3</sup>	81-120 µg/m <sup>3</sup>	21-35 µg/m <sup>3</sup>	11-20 µg/m <sup>3</sup>
REGULAR	201-350 µg/m <sup>3</sup>	101-200 µg/m <sup>3</sup>	121-180 µg/m <sup>3</sup>	36-50 µg/m <sup>3</sup>	21-25 µg/m <sup>3</sup>
MALO	351-500 µg/m <sup>3</sup>	201-400 µg/m <sup>3</sup>	181-240 µg/m <sup>3</sup>	51-100 µg/m <sup>3</sup>	26-50 µg/m <sup>3</sup>
MUY MALO	501-1250 µg/m <sup>3</sup>	401-1000 µg/m <sup>3</sup>	241-600 µg/m <sup>3</sup>	110-1200 µg/m <sup>3</sup>	51-800 µg/m <sup>3</sup>



PM 2.5	AQI <small>Índice de calidad de aire</small>	Efectos sobre la salud de PM 2.5	Medidas de precaución
0 a 12.0	Bueno 0 - 50	Poco o ningún riesgo.	Ninguna
12.1 a 35.4	Moderado 51 - 100	Las personas sensibles pueden experimentar síntomas respiratorios.	Las personas sensibles deben considerar reducir el esfuerzo prolongado o intenso.
35.5 a 55.4	Nocivo para grupos sensibles 101 - 150	Mayor probabilidad de síntomas respiratorios en personas sensibles, agravamiento de enfermedades cardíacas o pulmonares y mortalidad prematura en personas con enfermedad cardiopulmonar y ancianos.	Las personas con enfermedades respiratorias o cardíacas, los ancianos y los niños deben limitar el esfuerzo prolongado.
55.5 a 150.4	Insalubre 151 - 200	Aumento del agravamiento de enfermedades cardíacas o pulmonares y mortalidad prematura en personas con enfermedad cardiopulmonar y ancianos; aumento de los efectos respiratorios en la población general.	Las personas con enfermedades respiratorias o cardíacas, los ancianos y los niños deben evitar el esfuerzo prolongado; todos los demás deben limitar el esfuerzo prolongado.
150.5 a 250.4	Muy insalubre 201 - 300	Agravamiento significativo de enfermedades cardíacas o pulmonares y mortalidad prematura en personas con enfermedad cardiopulmonar y ancianos; aumento significativo de los efectos respiratorios en la población general.	Las personas con enfermedades respiratorias o cardíacas, los ancianos y los niños deben evitar cualquier actividad al aire libre; todos los demás deben evitar el esfuerzo prolongado.
250.5 a 500.4	Peligroso 301 - 500	Agravamiento peligroso de enfermedades cardíacas o pulmonares y mortalidad prematura en personas con enfermedad cardiopulmonar y ancianos; Riesgo grave de efectos respiratorios en la población general.	Todos deben evitar cualquier esfuerzo al aire libre; las personas con enfermedades respiratorias o cardíacas, los ancianos y los niños deben permanecer en el interior.

- Configuración del hardware
  - Computador: Raspberry Pi 4



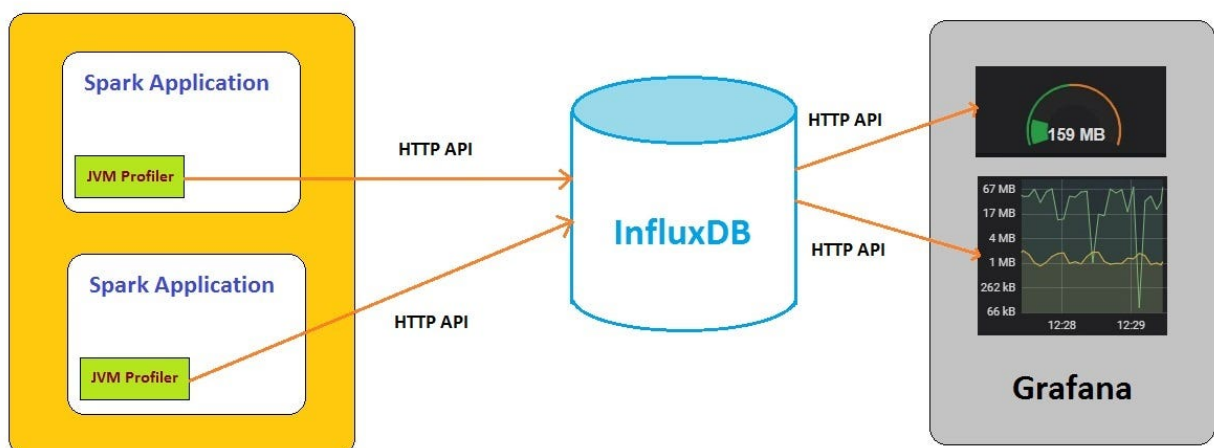
### TERCERA ETAPA:

Está constituida por un servidor local desarrollado y alojado en el **Computador Raspberry Pi 4**, en el cual corre una aplicación de página web que sirve como interface de usuario denominada Grafana, la cual muestra gráficamente el comportamiento de las mediciones de las partículas de gases. Todo el software corre bajo el sistema operativo Linux, tal como se explica y se muestra a continuación:

- Para cumplir con las características deseables del prototipo, este se desarrolló sobre un micro-computador Raspberry Pi modelo 4, usando sistema operativo Linux, con algoritmos programados en lenguaje Python.
- Se incorporó al sistema la librería “**influxdb**” para Python, para programar el almacenamiento de las lecturas obtenidas mediante los sensores incorporados al prototipo.
- También se incorporó al sistema la librería “**grafana**” para Python, para programar la visualización de los datos almacenados en formato de series temporales **influxdb**.
- El Computador utilizado incorpora en su placa electrónica los componentes necesarios para comunicación WiFi que se usarán para transmitir la descarga de los datos almacenados en el dispositivo de monitorización hacia los servidores de la ACP, al terminar el tránsito de los barcos por el Canal de Panamá.

#### • Arquitectura del Software

- Sistema Operativo (capa 1): **Linux**
- **InfluxDB** (capa 2): motor de base de datos para almacenar puntos de datos generados por los sensores en series temporales y en tiempo real

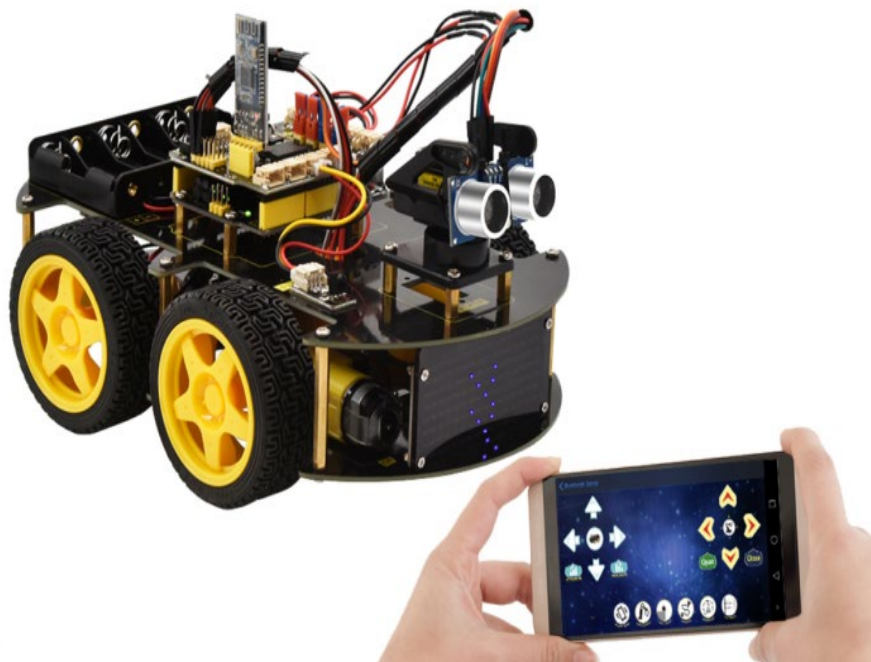


- **Grafana** (capa 3): plataforma interactiva para visualización de los puntos de datos generados por los sensores en formatos gráficos y de tablas, unificados en paneles de control (**dashboards**)



- **EnviroPlus-Python** (capa 4): librería de funciones Python para programar los sensores de la placa Enviro+.
- **Aplicaciones del Monitor Gas Buster** (capa5): algoritmos programados con lenguaje Python.

### Descripción de componentes y Pictórico del Robot







## 01 Description

This is an intelligent identification car for learning Raspberry Pi hardware control and AI. AI plays a critical role in science and technology. As electronic DIY enthusiasts, it is necessary to learn artificial intelligence.

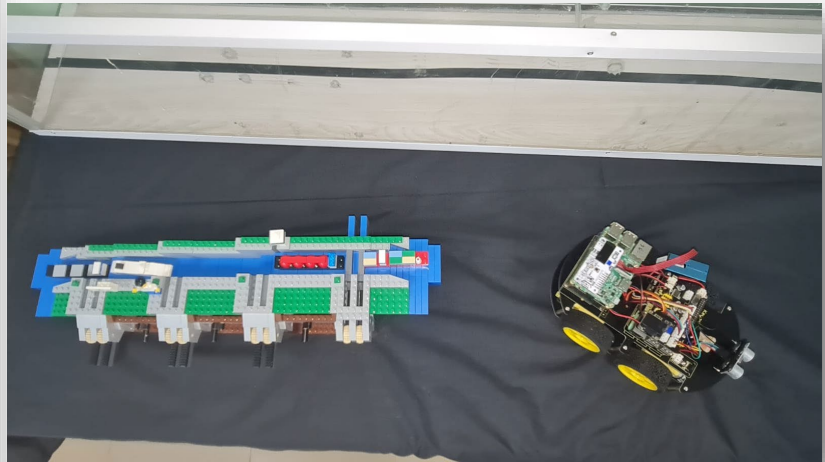
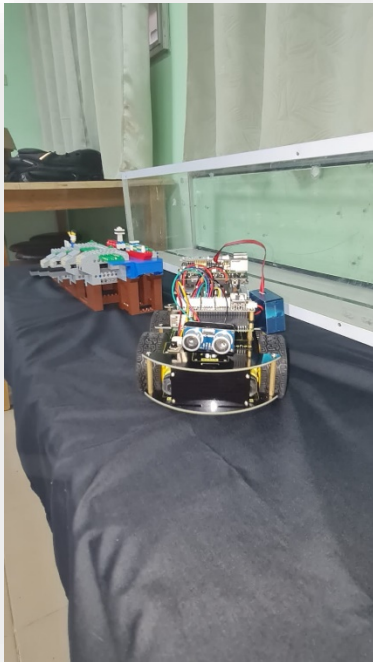
And the smart recognition car with the Raspberry Pi camera we rolled out immensely cultivates and stimulates learners' imagination and creativity. This kit includes numerous electronic modules, actuators, sensors, displays, communication modules and so on. On top of that, this smart car can perform objects identification, tracking and color, face and QR code recognition.

## 02 Kit List

Name	QTY	Picture	Name	QTY	Picture
Raspberry Pi 4WD Robot Motor Drive Shield	1		Line Tracking Sensor	1	
Acrylic Board for Ultrasonic Sensor	1		IR Receiver Sensor	1	
Acrylic Board for Cooling Fan	1		0.96 Inch OLED Display Module	1	
Acrylic Boards for Camera	1		180° Metal Servo	1	
HC-SR04 Ultrasonic Sensor	1		18650 Battery Holder	1	
Cooling Fan	1		JMFP-4 17-Key Remote Control	1	
Acrylic Board	1		8*16 LED Dot Matrix	1	
4WD Smart Car V3.0 PCB Boards (TOP+Bottom)	2		Fixed Part	4	

Wheels	4		M3*30MM Round Head Screws	8	
M3*60MM Dual-pass Copper Pillar	4		M3*8MM Round Head Screws	4	
M2.5*10MM Dual-pass Copper Pillar	8		M2*10MM Round Head Screws	7	
M3*40MM Dual-pass Copper Pillar	6		M3*8MM Flat Head Screws	3	
M2.5*6+6MM Copper Pillar	4		M1.4*10MM Round Head Screws	10	
M2 Nuts	8		M2.5*6MM Round Head Screws	13	
M1.4 Nuts	10		Five-Megapixel Raspberry Pi Camera	1	
M3 Nuts	14		4.5V 200r Motor	4	
2.0*40MM Screwdriver	1		5P XH2.54 Dupont Wire	1	
3.0*40MM Screwdriver	1		XH2.54 3P to PH2.0 3P Dupont Wire	1	
M1.2*5MM Round Head Screws	6		HX-2.54 4P Dupont Wire	1	
M3*6MM Round Head Screws	29		HX2.54mm 4P to 1P Connection Wire	1	





## Describe la codificación de la solución

### Código para movilizar el Robot

```
import RPi.GPIO as GPIO
import time

PIN = 15;
SCLK = 8
DIO = 9
# Display pattern data
matrix_smile = (0x00, 0x00, 0x38, 0x40, 0x40, 0x40, 0x3a, 0x02, 0x02, 0x3a, 0x40,
0x40, 0x40, 0x38, 0x00, 0x00)
matrix_forward = (0x00, 0x00, 0x00, 0x00, 0x12, 0x24, 0x48, 0x90, 0x90, 0x48, 0x24,
0x12, 0x00, 0x00, 0x00, 0x00)
matrix_back = (0x00, 0x00, 0x00, 0x00, 0x48, 0x24, 0x12, 0x09, 0x09, 0x12, 0x24,
0x48, 0x00, 0x00, 0x00, 0x00)
matrix_left = (0x00, 0x00, 0x00, 0x00, 0x18, 0x24, 0x42, 0x99, 0x24, 0x42, 0x81,
0x00, 0x00, 0x00, 0x00, 0x00)
matrix_right = (0x00, 0x00, 0x00, 0x00, 0x00, 0x81, 0x42, 0x24, 0x99, 0x42, 0x24,
0x18, 0x00, 0x00, 0x00, 0x00)

GPIO.setmode(GPIO.BCM)
GPIO.setup(PIN,GPIO.IN,GPIO.PUD_UP)
GPIO.setup(SCLK,GPIO.OUT)
GPIO.setup(DIO,GPIO.OUT)

print("irm test start...")
# Control M2 motor
L_IN1 = 20
L_IN2 = 21
L_PWM1 = 0
# Control M1 motor
L_IN3 = 22
L_IN4 = 23
L_PWM2 = 1
# Control M3 motor
R_IN1 = 24
R_IN2 = 25
```

```

R_PWM1 = 12
# Control M4 motor
R_IN3 = 26
R_IN4 = 27
R_PWM2 = 13

GPIO.setmode(GPIO.BCM) # use BCM numbers
#set the MOTOR Driver Pin OUTPUT mode
GPIO.setup(L_IN1,GPIO.OUT)
GPIO.setup(L_IN2,GPIO.OUT)
GPIO.setup(L_PWM1,GPIO.OUT)
GPIO.setup(L_IN3,GPIO.OUT)
GPIO.setup(L_IN4,GPIO.OUT)
GPIO.setup(L_PWM2,GPIO.OUT)
GPIO.setup(R_IN1,GPIO.OUT)
GPIO.setup(R_IN2,GPIO.OUT)
GPIO.setup(R_PWM1,GPIO.OUT)
GPIO.setup(R_IN3,GPIO.OUT)
GPIO.setup(R_IN4,GPIO.OUT)
GPIO.setup(R_PWM2,GPIO.OUT)

GPIO.output(L_IN1,GPIO.LOW)
GPIO.output(L_IN2,GPIO.LOW)
GPIO.output(L_IN3,GPIO.LOW)
GPIO.output(L_IN4,GPIO.LOW)
GPIO.output(R_IN1,GPIO.LOW)
GPIO.output(R_IN2,GPIO.LOW)
GPIO.output(R_IN3,GPIO.LOW)
GPIO.output(R_IN4,GPIO.LOW)

#set pwm frequence to 1000hz
pwm_R1 = GPIO.PWM(R_PWM1,100)
pwm_R2 = GPIO.PWM(R_PWM2,100)
pwm_L1 = GPIO.PWM(L_PWM1,100)
pwm_L2 = GPIO.PWM(L_PWM2,100)

#set inital duty cycle to 0
pwm_R1.start(0)
pwm_L1.start(0)
pwm_R2.start(0)
pwm_L2.start(0)

def nop():
    time.sleep(0.000001)

def nop2():
    time.sleep(0.01)

def start():
    GPIO.output(SCLK,1)
    nop()
    GPIO.output(DIO,1)
    nop()
    GPIO.output(DIO,0)
    nop()
    GPIO.output(SCLK,0)

def send_date(date):
    for i in range(0,8):
        GPIO.output(SCLK,0)
        nop()
        if date & 0x01:
            GPIO.output(DIO,1)
        else:
            GPIO.output(DIO,0)
        nop()
        GPIO.output(SCLK,1)
        nop()

```

```

        date >>= 1
        GPIO.output(SCLK,0)

def end():
    GPIO.output(SCLK,0)
    nop()
    GPIO.output(DIO,0)
    nop()
    GPIO.output(SCLK,1)
    nop()
    GPIO.output(DIO,1)
    nop()

def matrix_display(matrix_value):
    start()
    send_date(0xc0)

    for i in range(0,16):
        send_date(matrix_value[i])

    end()
    start()
    send_date(0x8A)
    end()

def exec_cmd(key_val):
    if(key_val==0x46):
        print("Button up")
        matrix_display(matrix_forward)
        GPIO.output(L_IN1,GPIO.LOW) #Upper Left forward
        GPIO.output(L_IN2,GPIO.HIGH)
        pwm_L1.ChangeDutyCycle(50)
        GPIO.output(L_IN3,GPIO.LOW) #Lower left forward
        GPIO.output(L_IN4,GPIO.HIGH)
        pwm_L2.ChangeDutyCycle(50)
        GPIO.output(R_IN1,GPIO.HIGH) #Upper Right forward
        GPIO.output(R_IN2,GPIO.LOW)
        pwm_R1.ChangeDutyCycle(50)
        GPIO.output(R_IN3,GPIO.HIGH) #Lower Right forward
        GPIO.output(R_IN4,GPIO.LOW)
        pwm_R2.ChangeDutyCycle(50)
    elif(key_val==0x44):
        print("Button left")
        matrix_display(matrix_left)
        GPIO.output(L_IN1,GPIO.HIGH)
        GPIO.output(L_IN2,GPIO.LOW)
        pwm_L1.ChangeDutyCycle(100)
        GPIO.output(L_IN3,GPIO.LOW)
        GPIO.output(L_IN4,GPIO.HIGH)
        pwm_L2.ChangeDutyCycle(100)
        GPIO.output(R_IN1,GPIO.HIGH) #Upper Right forward
        GPIO.output(R_IN2,GPIO.LOW)
        pwm_R1.ChangeDutyCycle(100)
        GPIO.output(R_IN3,GPIO.LOW) #Lower Right forward
        GPIO.output(R_IN4,GPIO.HIGH)
        pwm_R2.ChangeDutyCycle(100)
    elif(key_val==0x40):
        print("Button ok")
        matrix_display(matrix_smile)
        pwm_L1.ChangeDutyCycle(0)
        pwm_L2.ChangeDutyCycle(0)
        pwm_R1.ChangeDutyCycle(0)
        pwm_R2.ChangeDutyCycle(0)
    elif(key_val==0x43):
        print("Button right")
        matrix_display(matrix_right)
        GPIO.output(L_IN1,GPIO.LOW) #Upper Left forward
        GPIO.output(L_IN2,GPIO.HIGH)

```



```

        pwm_L1.ChangeDutyCycle(100)
        GPIO.output(L_IN3,GPIO.HIGH) #Lower left forward
        GPIO.output(L_IN4,GPIO.LOW)
        pwm_L2.ChangeDutyCycle(100)
        GPIO.output(R_IN1,GPIO.LOW) #Upper Right forward
        GPIO.output(R_IN2,GPIO.HIGH)
        pwm_R1.ChangeDutyCycle(100)
        GPIO.output(R_IN3,GPIO.HIGH) #Lower Right forward
        GPIO.output(R_IN4,GPIO.LOW)
        pwm_R2.ChangeDutyCycle(100)
    elif(key_val==0x15):
        print("Button down")
        matrix_display(matrix_back)
        GPIO.output(L_IN1,GPIO.HIGH)
        GPIO.output(L_IN2,GPIO.LOW)
        pwm_L1.ChangeDutyCycle(50)
        GPIO.output(L_IN3,GPIO.HIGH)
        GPIO.output(L_IN4,GPIO.LOW)
        pwm_L2.ChangeDutyCycle(50)
        GPIO.output(R_IN1,GPIO.LOW)
        GPIO.output(R_IN2,GPIO.HIGH)
        pwm_R1.ChangeDutyCycle(50)
        GPIO.output(R_IN3,GPIO.LOW)
        GPIO.output(R_IN4,GPIO.HIGH)
        pwm_R2.ChangeDutyCycle(50)
    elif(key_val==0x16):
        print("Button 1")
    elif(key_val==0x19):
        print("Button 2")
    elif(key_val==0x0d):
        print("Button 3")
    elif(key_val==0x0c):
        print("Button 4")
    elif(key_val==0x18):
        print("Button 5")
    elif(key_val==0x5e):
        print("Button 6")
    elif(key_val==0x08):
        print("Button 7")
    elif(key_val==0x1c):
        print("Button 8")
    elif(key_val==0x5a):
        print("Button 9")
    elif(key_val==0x42):
        print("Button *")
    elif(key_val==0x52):
        print("Button 0")
    elif(key_val==0x4a):
        print("Button #")

try:
    while True:
        if GPIO.input(PIN) == 0:
            count = 0
            while GPIO.input(PIN) == 0 and count < 200: # Wait for 9ms LOW level
                boot code and exit the loop if it exceeds 1.2ms
                count += 1
                time.sleep(0.00006)

            count = 0
            while GPIO.input(PIN) == 1 and count < 80: # Wait for a 4.5ms HIGH
                level boot code and exit the loop if it exceeds 0.48ms
                count += 1
                time.sleep(0.00006)

            idx = 0 # byte count variable
            cnt = 0 #Variable per byte bit

```

```

        #There are 4 bytes in total. The first byte is the address code, the
        second is the address inverse code,
        #the third is the control command data of the corresponding button, and
        the fourth is the control command inverse code
        data = [0,0,0,0]
        for i in range(0,32): # Start receiving 32BITE data
            count = 0
            while GPIO.input(PIN) == 0 and count < 15: # Wait for the LOW LOW
level of 562.5US to pass and exit the loop if it exceeds 900US
                count += 1
                time.sleep(0.00006)

            count = 0
            while GPIO.input(PIN) == 1 and count < 40: # waits for logical
HIGH level to pass and exits the loop if it exceeds 2.4ms
                count += 1
                time.sleep(0.00006)

            # if count>8, that is, the logical time is greater than
0.54+0.562=1.12ms, that is,
            #the period is greater than the logical 0 period, that is
            equivalent to receiving logical 1
            if count > 8:
                data[idx] |= 1<<cnt #When idx=0 is the first data data[idx]
= data[idx] | 1<<cnt 00000001 <<1 == 0000 0010
                if cnt == 7: #With 8 byte
                    cnt = 0 #Displacement qing 0
                    idx += 1 #Store the next data
                else:
                    cnt += 1 #The shift adds 1
            #Determine whether address code + address inverse code =0xff, control
code + control inverse code = 0xFF
            if data[0]+data[1] == 0xFF and data[2]+data[3] == 0xFF:
                print("Get the key: 0x%02x" %data[2]) #Data [2] is the control
code we need
                exec_cmd(data[2])
except KeyboardInterrupt:
    GPIO.cleanup()

```

## Código de programación para el funcionamiento del sensor de CO2sensor de

```

#!/usr/bin/env python3
#
#=====
# Programa: gasbusters-sensores.py
# Programadores: Stephen Perz
#                Isabella Gonzalez
#                Ferruccio Guicciardi (tutor)
#
# Descripcion: con este programa se controla la placa HAT Enviro+
#              instalada sobre le computador Raspberry Pi 4, para
#              leer los sensores y almacenar los datos generados por
#              los sensores en la plataforma InfluxDB.
#=====

#=====
# seccion 1 - importacion de librerias y modulos Python
#=====
import os      # funciones del sistema operativo
import time    # funciones p/acceder a fechas y tiempo
from pms5003 import PMS5003, ReadTimeoutError # funciones p/PMS5003

```

```

from influxdb import InfluxDBClient    # funciones para InfluxDB

#=====
# seccion 2 - variables globales
#=====
servidorIP      = 'localhost'
servidorPuerto = 8086
usuario         = 'admin'
passwd          = 'admin'
rutaBaseDatos   = '/home/pi'
ptoDatos         = None                # punto de datos generado x sensor
espera10Mins     = time.time() + 600   # espera 10 mins desde ahora

#=====
# seccion 3 - funciones
#=====

# hacer una pausa de 10 minutos mientras se estabilizan los sensores
def esperar10Mins():
    try:
        pms5003 = PMS5003()            # instancia sensor de particulas PMS5003
        time.sleep(1.0)
        while True:
            try:
                if time.time() > espera10Mins:
                    break
            else:
                ptoDatos = pms5003.read()
                print(str(ptoDatos))
            except ReadTimeoutError:
                pms5003 = PMS5003()
        except KeyboardInterrupt:
            pass

# grabar puntos de datos en la plataforma InfluxDB
def grabarEnInfluxDB():
    ptoDatos = [{

    }]

#=====
# seccion 4 - ejecucion del bucle principal
#=====
def principal():
    pass

#=====
# seccion 5 - ejecucion inicial del programa
#=====
if __name__ == "__main__":
    esperar10Mins()
    # configurar la plataforma InfluxDB
    clienteInfluxDB = InfluxDBClient(servidorIP,servidorPuerto,usuario,
                                     passwd, rutaBaseDatos)

    principal()

```

## Código de programación de la Aplicación Monitor GAS BUSTERS

```
#!/usr/bin/env python3
#
#=====
# Programa: gasbusters-sensores.py
# Programadores: Stephen Perz
#               Isabella Gonzalez
#
# Descripcion: con este programa se controla la placa HAT Enviro+
#             instalada sobre le computador Raspberry Pi 4, para
#             leer los sensores y almacenar los datos generados por
#             los sensores en la plataforma InfluxDB.
#=====

#=====
# seccion 1 - importacion de librerias y modulos Python
#=====

import os    # funciones del sistema operativo
import time  # funciones p/acceder a fechas y tiempo
from pms5003 import PMS5003, ReadTimeoutError # funciones p/PMS5003
                                                # sensor de particulas
from influxdb import InfluxDBClient          # funciones para InfluxDB
from bme280 import BME280                    # funciones para sensor BME280

#=====
# seccion 2 - variables globales
#=====

servidorIP    = 'localhost'
servidorPuerto = 8086
usuario       = 'admin'
passwd        = 'admin'
rutaBaseDatos = '/home/pi'
ptosDatos     = None # punto de dato - sensores
ptoPresion    = None # punto de dato - sensor de presion atmosferica
ptoHumedad    = None # punto de dato - sensor de humedad relativa
ptoPM1        = None # punto de dato - sensor de particulas < 1.0 ug/m3
ptoPM25       = None # punto de dato - sensor de particulas < 2.5 ug/m3
ptoPM10       = None # punto de dato - sensor de particulas < 10 ug/m3
espera10Mins  = time.time() + 600 # espera 10 mins desde ahora
pms5003       = None
bme280        = None

#=====
```

```

=====
# seccion 3 - funciones
#=====
=====

# hacer una pausa de 10 minutos mientras se estabilizan los sensores
def esperar10Mins():
    global pms5003
    try:
        time.sleep(1.0)
        while True:
            try:
                if time.time() > espera10Mins:
                    break
            else:
                ptoDatos = pms5003.read()
                print(str(ptoDatos))
            except ReadTimeoutError:
                pms5003 = PMS5003()
    except KeyboardInterrupt:
        pass

def leerPresion(): # leer sensor de presion atmosferica
    global bme280
    ptoPresion = bme280.get_pressure()
    print("Presion = " + str(ptoPresion))

def leerHumedad(): # leer sensor de humedad relativa
    global bme280
    ptoHumedad = bme280.get_humidity()
    print("Humedad = " + str(ptoHumedad))

def leerPM1(): # leer particulas < 1 uG/m3
    global bme280
    ptoDatos = pms5003.read()
    ptoPM1 = float(ptoDatos.pm_ug_per_m3(1.0))
    print("PM1 = " + str(ptoPM1))

def leerPM25(): # leer particulas < 2.5 uG/m3
    global bme280
    ptoDatos = pms5003.read()
    ptoPM25 = float(ptoDatos.pm_ug_per_m3(2.5))
    print("PM2.5 = " + str(ptoPM25))

def leerPM10(): # leer particulas < 10.0 ug/m3
    global bme280
    ptoDatos = pms5003.read()
    ptoPM10 = float(ptoDatos.pm_ug_per_m3(10))
    print("PM10 = " + str(ptoPM10))

```

```

# grabar puntos de datos en la plataforma InfluxDB
#def grabarEnInfluxDB():

#=====
# seccion 4 - ejecucion del bucle principal
#=====
def principal():
    try:
        while True:
            leerPresion()
            leerHumedad()
            leerPM1()
            leerPM25()
            leerPM10()
    except KeyboardInterrupt:
        pass

#=====
# seccion 5 - ejecucion inicial del programa
#=====
if __name__ == "__main__":

    pms5003 = PMS5003() # instancia sensor de particulas PMS5003
    bme280 = BME280() # instancia sensor BME280(temp/presion/humedad)

    esperar10Mins()

    # crear instancia del servidor InfluxDB
    cliente = InfluxDBClient(servidorIP,servidorPuerto,usuario,
                             passwd, rutaBaseDatos)

    # crear la base de datos para los datos de sensores
    cliente.create_database('gasbusters')
    cliente.switch_database('gasbusters')

    # ejecutar el bucle principal
    principal()

```

### **¿Se enfrentó a algún reto durante el proceso de desarrollo?**

Problemas	Soluciones
En el momento del ensamblaje, se conectaron mal los cables, llantas y servomotores.	Desensamblar el carro robot y volver a hacer el proceso de ensamblaje nuevamente
A la hora de programar el carro presentó algunas fallas en la programación que hacían que no se movilizara correctamente.	corrigiendo las fallas que tuvo en la programación para que funcionara correctamente
los sensores y el carro necesitaban programación en Python	tomamos un curso acelerado de Python

### **EL IMPACTO De SU SOLUCIÓN PARA LA SOCIEDAD**

#### **¿A quién ayudará y qué importancia tiene?**

Nuestra propuesta contribuye con un programa que la ACP ya tiene funcionando en cuanto a las metas de la reducción de los gases contaminantes y para eso nosotros hemos creado un robot que puede ser colocado en la cubierta de los barcos una vez los pasa barcos una vez los pasa barcos abordan los barcos para guiarlos en el canal.

La importancia de este robot es que tomará medidas particulares de cada una de las emisiones de los gases contaminantes que degradan el medio ambiente alrededor de la ribera del canal.

#### **Pon un ejemplo concreto de cómo o dónde podría utilizarse tu idea (piensa en quién la utilizará y cuántas personas se beneficiarían de ella).**

Se podría utilizar en el canal de Panamá específicamente en las zonas que hay mayor flujo de barcos, es decir, donde hay mayor contaminación y emisión de Dióxido de carbono, todo esto lo podríamos observar mediante una página web, un servidor que es una base de datos o también lo podremos ver desde nuestro dispositivo móvil, así también podremos recoger esta información que puede ser utilizada para multar o para incentivar la fabricación de barcos que emiten menos Dióxido de carbono, cualquiera persona podría observar lo que pasa en el ambiente del canal.



## MODELO DE NEGOCIO

### MODELO DE NEGOCIO



## Fuentes Bibliográficas Consultadas

### Referencias bibliograficas del Proyecto Gas Busters

1. Placa Enviro+ para medicion ambiental...  
<https://shop.pimoroni.com/products/enviro?variant=31155658489939>
2. Canal de Panama publica mensualmente las reducciones de CO2 de sus clientes...  
[https://www.swissinfo.ch/spa/panam%C3%A1-canal\\_canal-de-panam%C3%A1-publica-mensualmente-las-reducciones-de-co2-de-sus-clientes/46330114](https://www.swissinfo.ch/spa/panam%C3%A1-canal_canal-de-panam%C3%A1-publica-mensualmente-las-reducciones-de-co2-de-sus-clientes/46330114)
3. Calidad del aire y niveles de contaminantes en diferentes sectores de Ancon...  
[https://www.google.com/search?q=calidad+del+aire+y+niveles+de+contaminantes+en+diferentes+sectores+de+ancon&sxsrf=APwXEdfwBuIRhp-FEVfSQuyGJNQUcIB7PQ%3A1687180820474&ei=FFaQZOXCHOWNhbIP9JiQsA8&ved=0ahUKEwjlnOerts\\_AhXIRkEAHXQMBPYQ4dUDCA8&oq=calidad+del+aire+y+niveles+de+contaminantes+en+diferentes+sectores+de+ancon&gs\\_lcp=Cgxnd3Mtd2l6LXNlcnAQDEoECEEYAFaAWABgAGgAcAB4AIABAIgBAJIBAJgBAKABAQ&scient=gws-wiz-serp](https://www.google.com/search?q=calidad+del+aire+y+niveles+de+contaminantes+en+diferentes+sectores+de+ancon&sxsrf=APwXEdfwBuIRhp-FEVfSQuyGJNQUcIB7PQ%3A1687180820474&ei=FFaQZOXCHOWNhbIP9JiQsA8&ved=0ahUKEwjlnOerts_AhXIRkEAHXQMBPYQ4dUDCA8&oq=calidad+del+aire+y+niveles+de+contaminantes+en+diferentes+sectores+de+ancon&gs_lcp=Cgxnd3Mtd2l6LXNlcnAQDEoECEEYAFaAWABgAGgAcAB4AIABAIgBAJIBAJgBAKABAQ&scient=gws-wiz-serp)
4. Canal de Panama inicia proceso con miras a convertirse en carbono neutral para el ano 2030...  
<https://pancanal.com/canal-de-panama-inicia-proceso-con-miras-a-convertirse-en-carbono-neutral-para-el-ano-2030/#:~:text=26%20Abril%202021-,Canal%20de%20Panam%C3%A1%20inicia%20proceso%20con%20miras%20a%20convertirse,neutral%20para%20el%20a%C3%B1o%202030&text=El%20Canal%20de%20Panam%C3%A1%20anunci%C3%B3,neutral%20para%20el%20a%C3%B1o%202030>
5. La contaminacion marina producida por buques...  
[https://www.juntadeandalucia.es/medioambiente/web/Bloques\\_Tematicos/Publicaciones\\_Divulgacion\\_Y\\_Noticias/Publicaciones\\_Periodicas/IMA/2002/ima\\_2002\\_pdfs/MONOGRAFIA2.pdf](https://www.juntadeandalucia.es/medioambiente/web/Bloques_Tematicos/Publicaciones_Divulgacion_Y_Noticias/Publicaciones_Periodicas/IMA/2002/ima_2002_pdfs/MONOGRAFIA2.pdf)
6. Que son los gases de efecto invernadero o ‘greenhouse gases’?...  
<https://www.bbva.com/es/sostenibilidad/que-son-los-gases-de-efecto-invernadero-o-greenhouse-gases/>
7. Que es la huella de carbono y como entender este indicador ambiental...  
<https://www.bbva.com/es/sostenibilidad/que-es-la-huella-de-carbono-y-como-entender-este-indicador-ambiental>
8. Introduccion a Datos de Series Temporales Que son? Y Con que se comen?...  
<https://cduser.com/introduccion-a-datos-de-series-temporales/>
9. Que es InfluxDB (video)...  
<https://www.youtube.com/shorts/12F2ZAMoVro>
10. Visualizacion de Series Temporales...  
<https://analisis-web.es/visualizacion-de-series-temporales/>
11. Que es Grafana ?...  
<https://www.youtube.com/watch?v=zh2savGkbyU>