

# BLITZ

## Integrantes

Jhon Alarcón

Fernando Garrido

Ana Gutiérrez

Cristobál Hernández

Felipe Lira



# Índice

1 *Introducción.*

2 *Recordando*

3 *Análisis y diseño.*

4 *Arquitectura.*

5 *Interfaz gráfica.*

6 *Implementación*

7 *Resultados.*

8 *Progreso actual.*

9 *Conclusión.*



# Introducción

Durante el transcurso del semestre, se ha planteado la tarea de realizar en equipo el proyecto "Blitz", que consiste en el diseño, construcción y programación de un robot. Utilizando el Lego Mindstorm Ev3 Core Set, se espera que este sea capaz de seguir instrucciones que le permitan desplazarse, así como recoger y soltar una pelota de ping pong.



# Objetivo general

Desarrollar un robot capaz de seguir instrucciones a través de una interfaz gráfica codificada en Python, con la finalidad de desplazarse, recoger y dejar una pelota de ping pong.





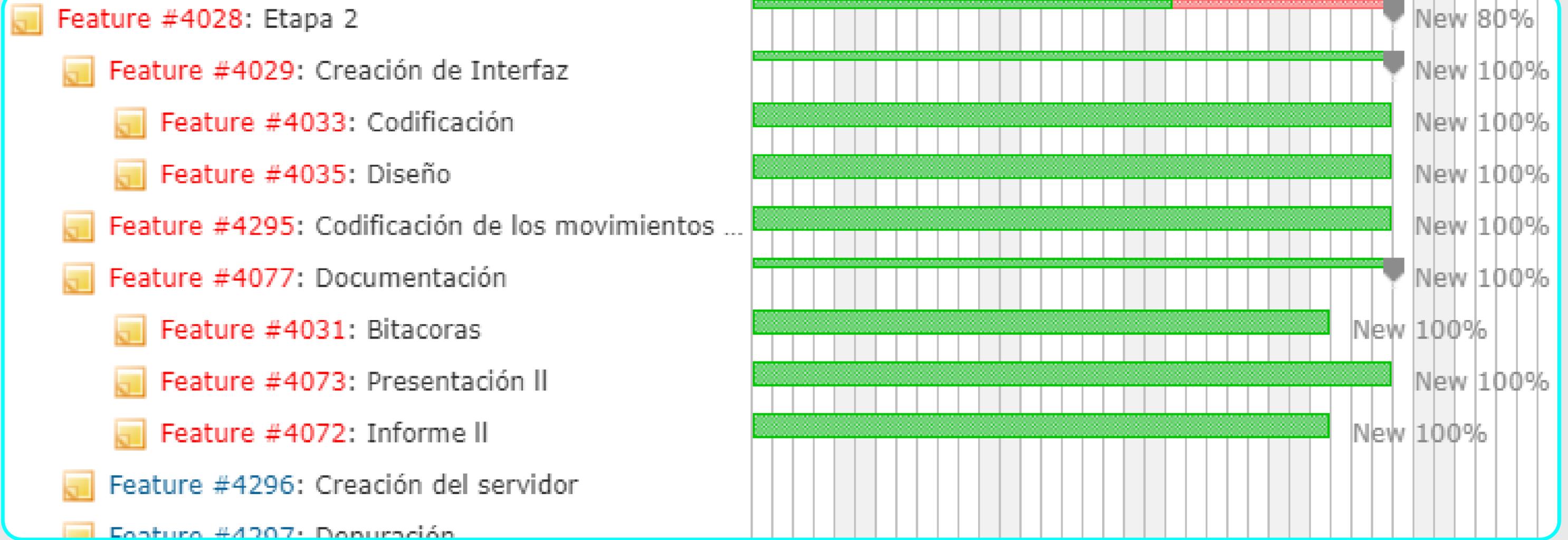
# Organización del Personal

Rol	Responsables	Involucrados
Jefe de Grupo	Cristóbal Hernández	Cristóbal Hernández
Ensamblador	Felipe Lira	Fernando Garrido
Programador	Jhon Alarcón	Cristóbal Hernández
Documentador	Ana Gutiérrez	Ana Gutiérrez
Diseñador	Ana Gutiérrez	Ana Gutiérrez

Presentación de proyecto



# Carta Gantt





# Costos

Hardware	Software	Gestión
\$2.143.970	\$0	\$9.565.000

**\$11.708.970**



# Wiki



## Índice

- ✦ *Introducción*
- ✦ *Organización y planificación*
- ✦ *Progreso del robot*
- ✦ *Análisis y diseño*
- ✦ *Código e implementación*



# Análisis y diseño

## Requerimientos funcionales

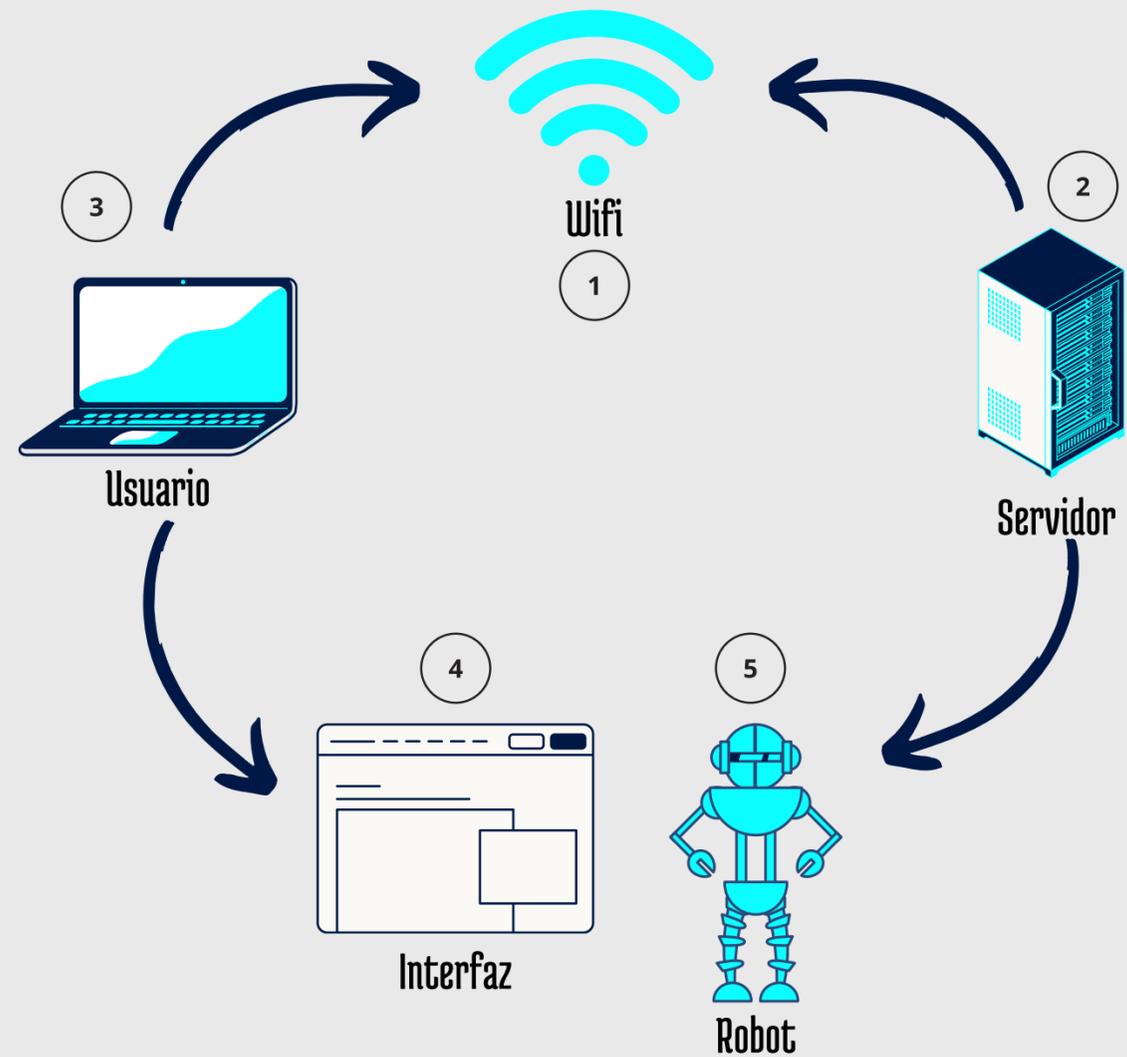
- ⚡ El robot debe poder moverse en todas las direcciones.
- ⚡ El robot debe ser capaz de recoger una pelota de ping pong y dejarla.
- ⚡ El robot debe poder ser controlado por el usuario a través de una interfaz.

## Requerimientos no funcionales

- ⚡ La interfaz gráfica debe ser fácil de usar y responder en menos de 2 segundos.
- ⚡ El manual de usuario debe ser detallado y claro.
- ⚡ El robot debe funcionar por al menos 3 horas sin necesidad de recargar.



# Arquitectura





# Interfaz gráfica



# Conectar IP

IP del Servidor:

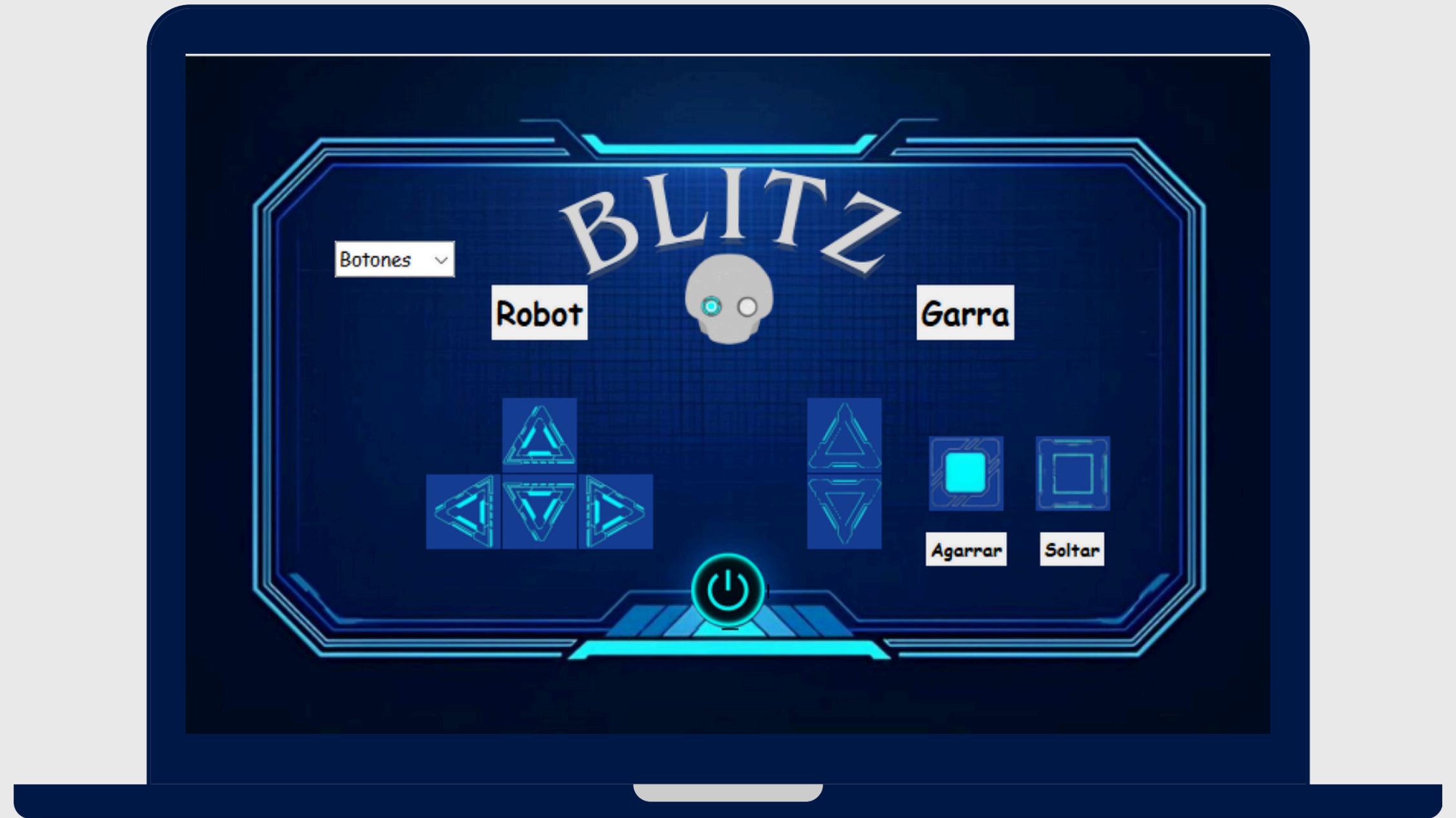
Puerto del Servidor:

Conectar

Ingresar IP guardada

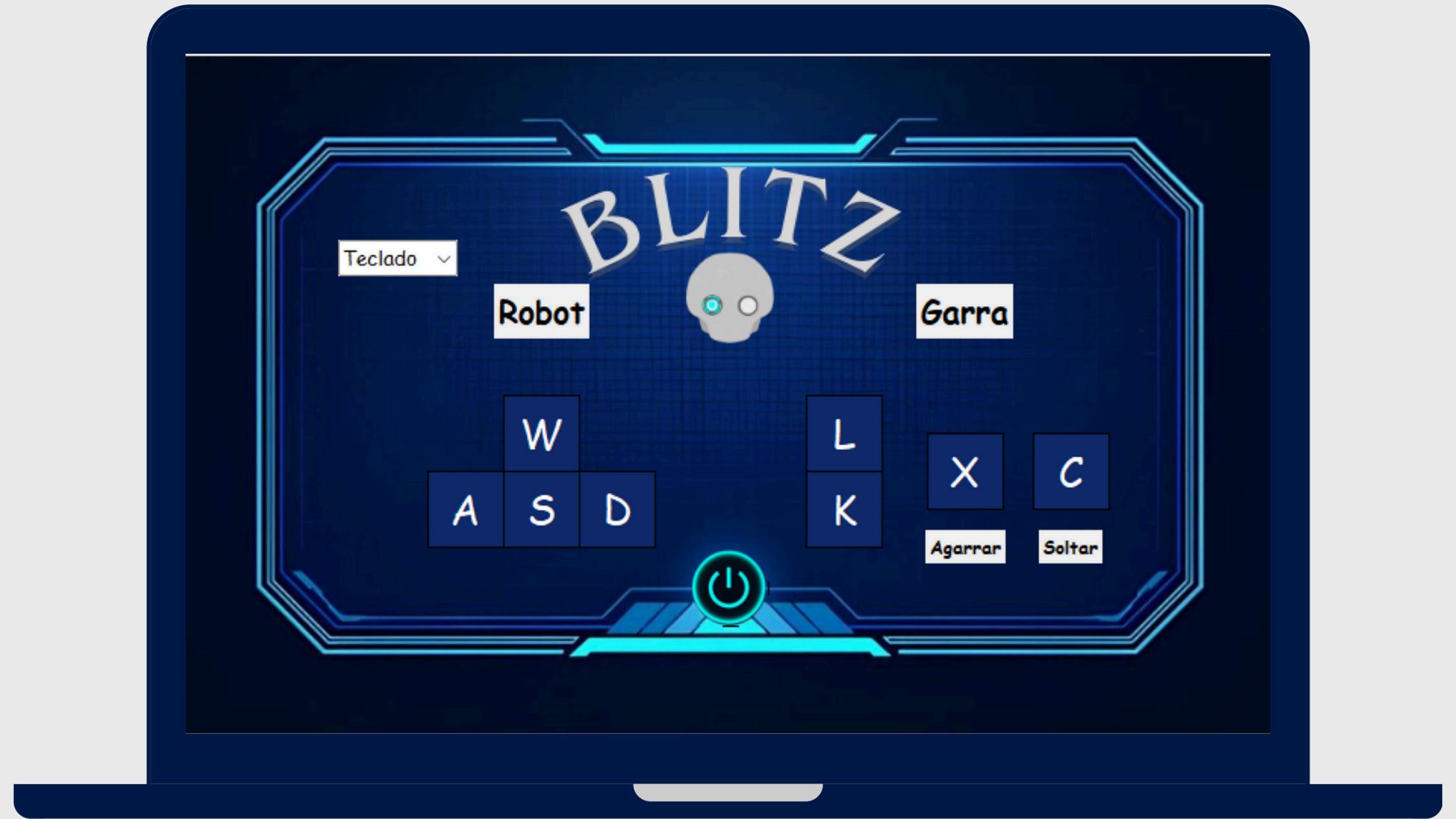


# Botones



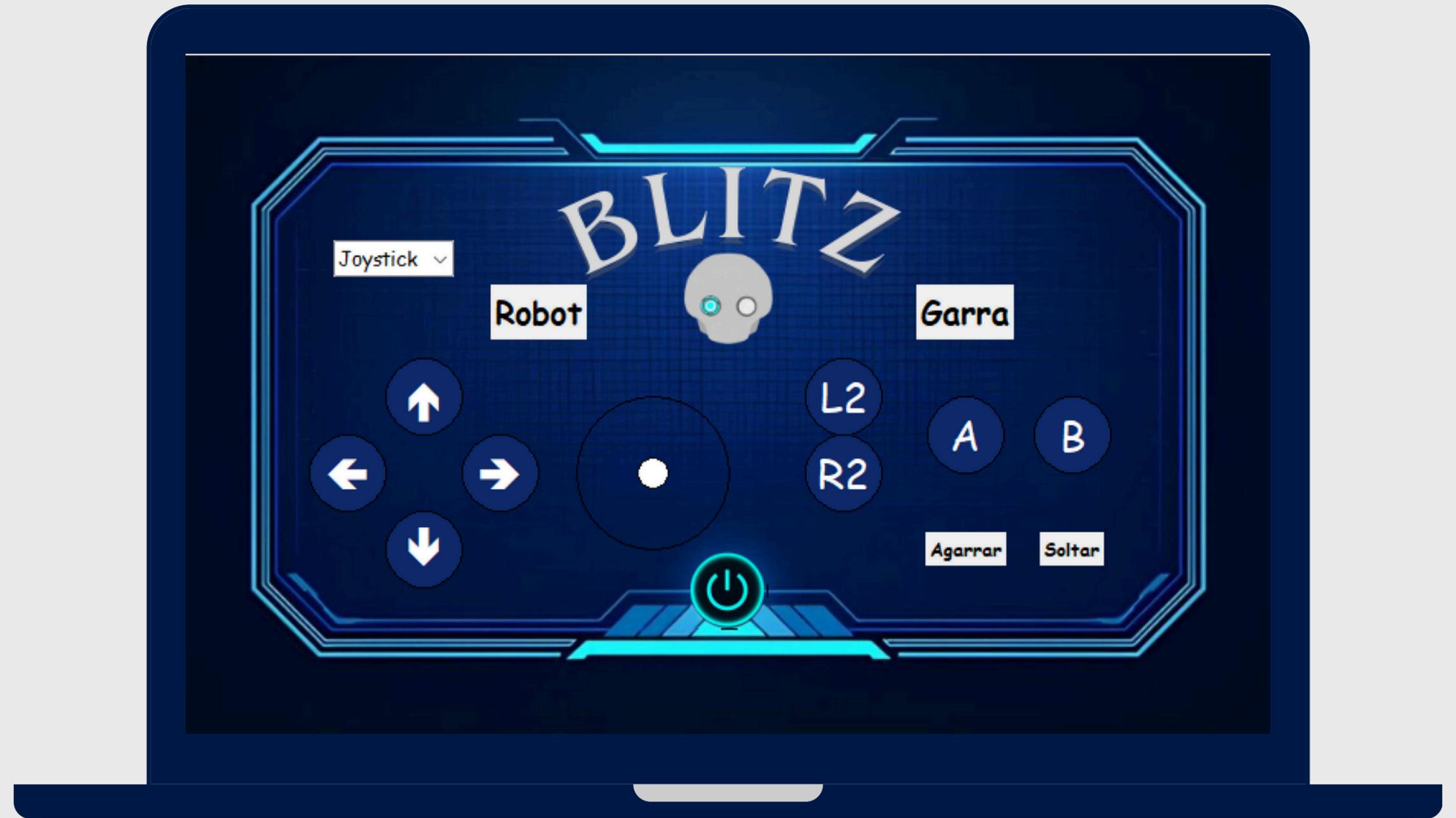


# Teclado





# Joystick





# Código



# server.py

```
robot > PY server.py
1  import socket
2  from funciones_dos import *
3
4  s = socket.socket()
5  print("Socket creado")
6  port = 8080
7  s.bind(('', port))
8
9  s.listen(5)
10 print("Servidor escuchando en el puerto 8080...")
11
12 connect, addr = s.accept()
13 print("Conectado con a {}".format(addr))
14
15 while True:
16     rawByte = connect.recv(1)
17     comando = rawByte.decode('utf-8')
18
19     if comando == 'w':
20         mover_arriba()
21
22     if comando == 's':
23         mover_atras()
24
25     if comando == 'a':
26         mover_izquierda()
27
```

```
robot > PY server.py
15  while True:
25     if comando == 'a':
26         mover_izquierda()
27
28     if comando == 'd':
29         mover_derecha()
30
31     if comando == 'x':
32         atrapar()
33
34     if comando == 'c':
35         soltar()
36
37     if comando == 'l':
38         base_subir()
39
40     if comando == 'k':
41         base_soltar()
42
43     if comando == ' ':
44         stop()
45
```



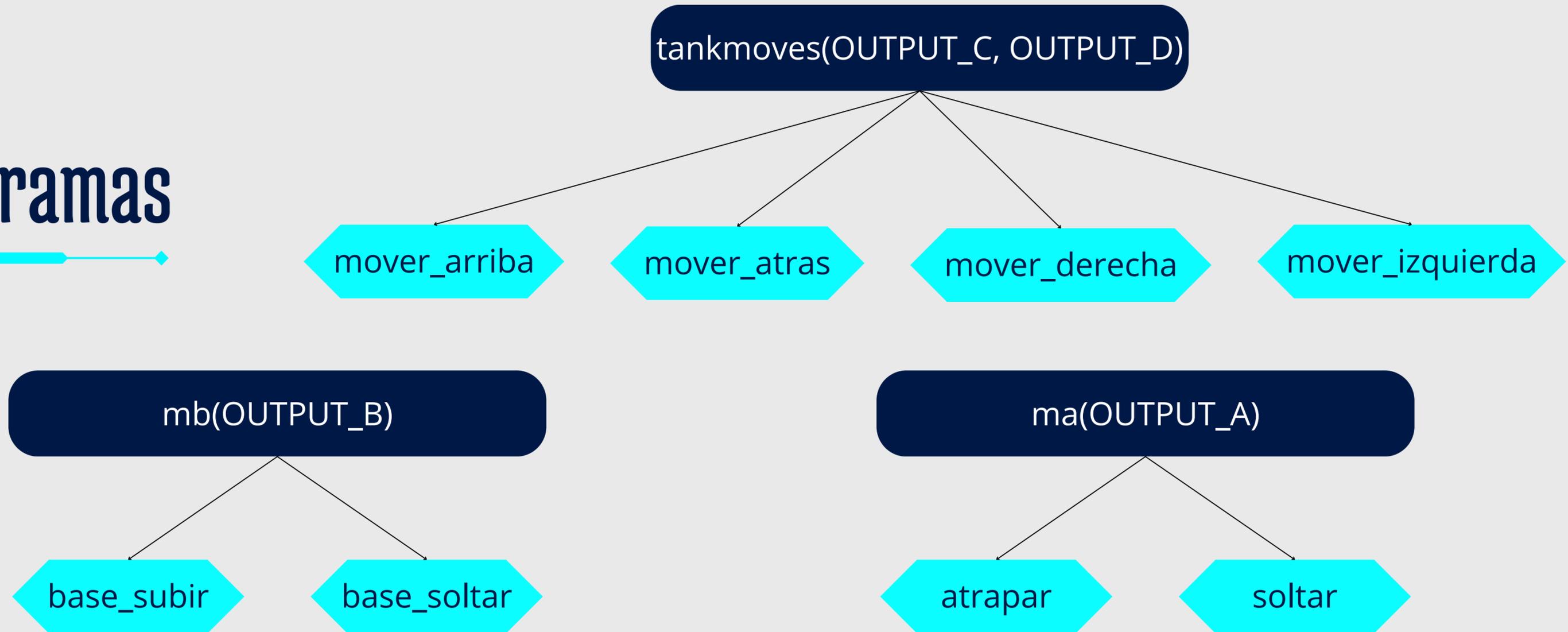


# library.py

```
robot > PY library.py
29 # Subir Garra hacia adelante
30 def base_subir():
31     mb.run_to_rel_pos(position_sp=135, speed_sp=200)
32     time.sleep(1)
33     mb.off()
34
35
36 # Bajar Garra hacia adelante
37 def base_soltar():
38     mb.run_to_rel_pos(position_sp=-135, speed_sp=200)
39     time.sleep(1)
40     mb.off()
41
42 # Movimiento Garra atrapar
43 def atrapar():
44     ma.run_to_rel_pos(position_sp=-1080, speed_sp=1000)
45     time.sleep(1)
46     md.off()
47
48 # Movimiento Garra soltar
49 def soltar():
50     ma.run_to_rel_pos(position_sp=1080, speed_sp=1000)
51     time.sleep(1)
52     md.off()
53
54 # Detener Robot
55 def stop():
56     tankmoves.off()
```



# Diagramas





# Implementación



# Torque

## 1. Torque máximo requerido

El torque depende del grado de orientación del brazo "θ":

$$T = F_{\text{total}} \cdot r \cdot \sin(\theta)$$

Donde  $F_{\text{total}} = (m_{\text{garra}} + m_{\text{carga}}) \cdot g$ :

$$T = (m_{\text{garra}} + m_{\text{carga}}) \cdot g \cdot r \cdot \sin(\theta)$$

Sustituyendo para el torque máximo en 90°:

$$T_{\text{max}} = 2.242\text{N} \cdot 0.212\text{m} \cdot 1 = 0.475\text{Nm}$$

Todo, incluyendo el peso de la garra, tendría un impacto significativo en el torque necesario: de 0.0056 Nm, que corresponde solo a la carga, a 0.475 Nm con la carga y la garra incluidas.



# Velocidad promedio

## 2. velocidad promedio

Formula de velocidad:

$$v = \frac{d}{t}$$

Reemplazando

$$v = \frac{1m}{3,5} = 0.286 \text{ m/s}$$

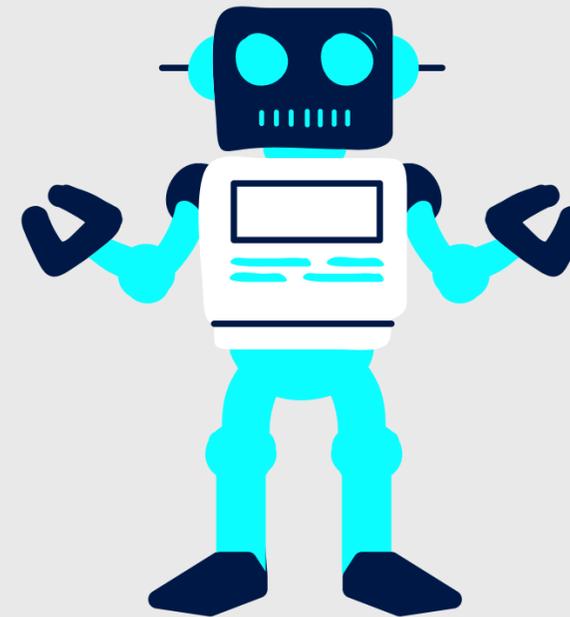


# Resultados



# Estado actual del proyecto

- ⚡ Ensamblaje del robot.
- ⚡ Funciones de movimiento del robot.
- ⚡ Interfaz.
- ⚡ Servidor.
- ⚡ Conexión remota funcional.
- ⚡ Documentación del proyecto.
- ⚡ Carta Gantt.
- ⚡ Bitácoras, informes y presentaciones.





# Problemas y soluciones

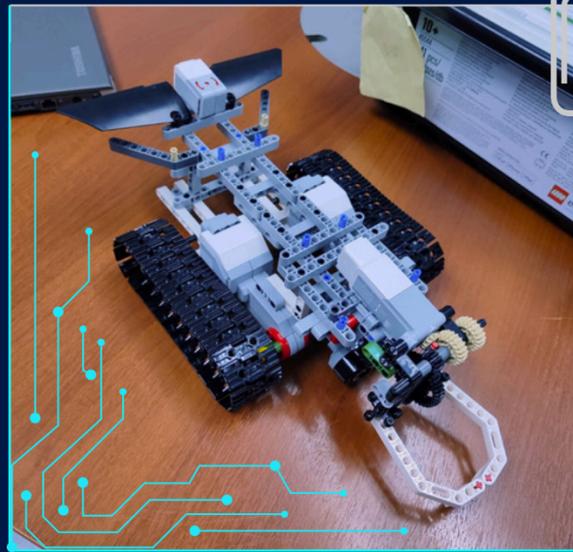
Problemas	Soluciones
Falta de materiales	Se contactó al responsable de la distribución de piezas para solicitar las piezas faltantes.
Diseño del robot inestable	Se reconstruyó el robot para cumplir con los requisitos de estabilidad, reforzando los puntos débiles.
Movimientos del robot	Se corrigieron las incongruencias entre el código y el diseño de la garra del robot.
Planificación errónea	Se reajustaron los tiempos para las actividades en la Carta Gantt.



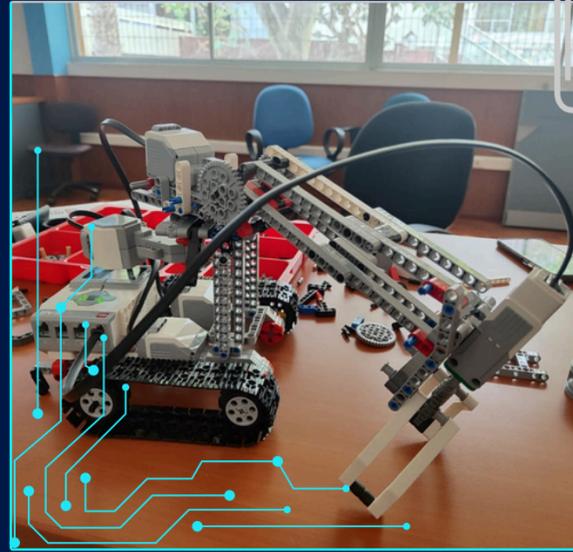
# Progreso actual



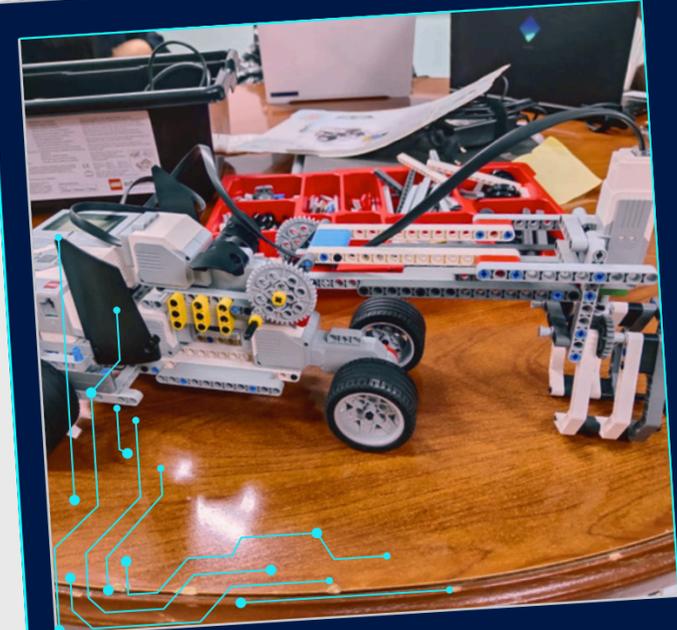
# Progreso del robot



VERSIÓN 1



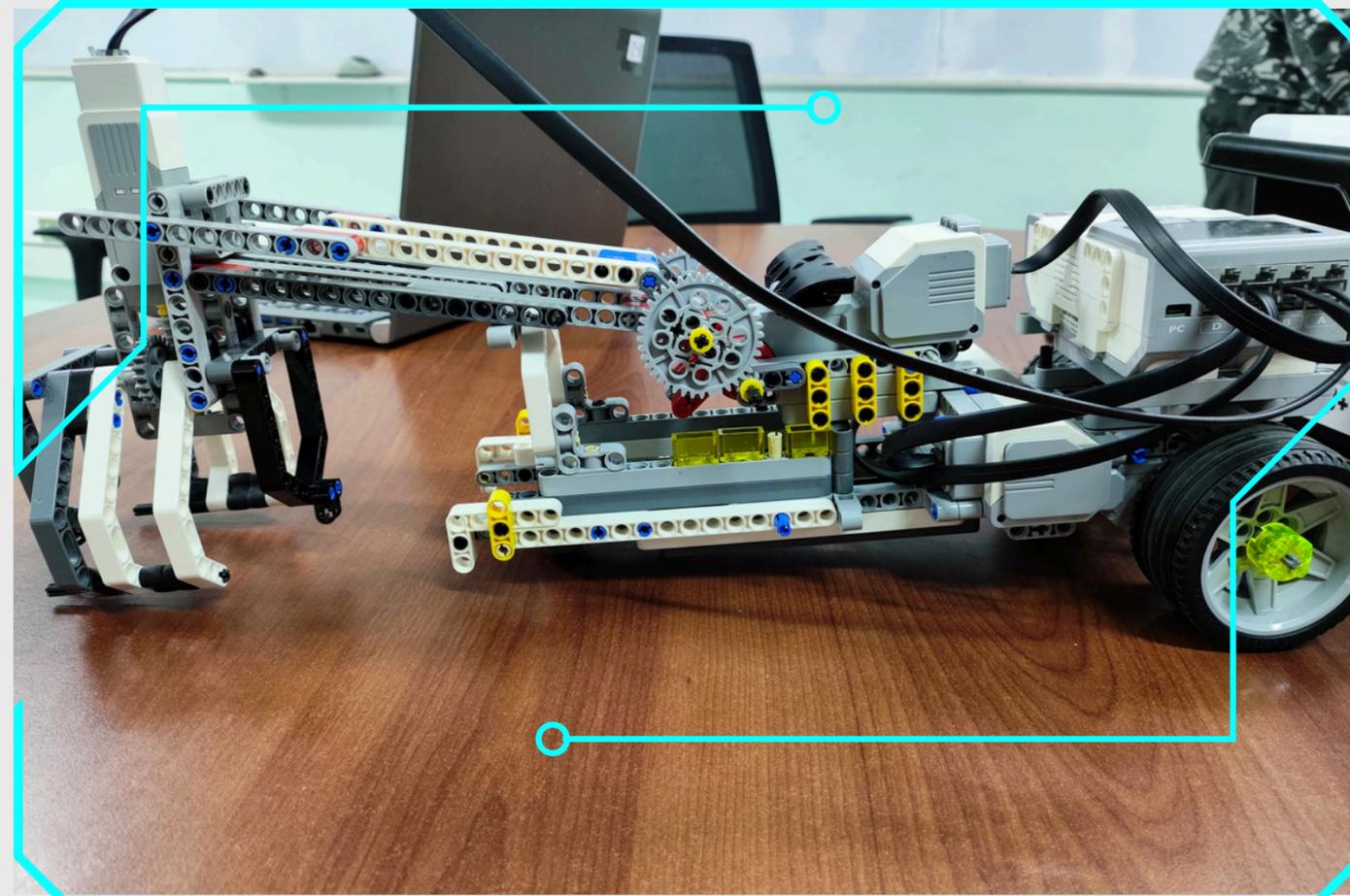
VERSIÓN 2



VERSIÓN 3



# Versión actual





# Conclusión

