```
#include <ESP8266WiFi.h>

const char* ssid = "ROBOT_AP";
const char* password = "12345678";

WiFiServer server(8080);

// Motor pins
#define L_MOTOR_A D0
#define L_MOTOR_B D1
#define R_MOTOR_A D2
#define R_MOTOR_B D3

void stopMotors() {
 digitalWrite(L_MOTOR_A, LOW);
 digitalWrite(L_MOTOR_B, LOW);
 digitalWrite(R_MOTOR_A, LOW);
 digitalWrite(R_MOTOR_B, LOW);
}

void moveForward() {
 digitalWrite(L_MOTOR_A, HIGH);
 digitalWrite(L_MOTOR_B, LOW);
 digitalWrite(R_MOTOR_A, HIGH);
 digitalWrite(R_MOTOR_B, LOW);
}

void moveBackward() {
 digitalWrite(L_MOTOR_A, LOW);
 digitalWrite(L_MOTOR_B, HIGH);
 digitalWrite(R_MOTOR_A, LOW);
 digitalWrite(R_MOTOR_B, HIGH);
}

void turnLeft() {
 digitalWrite(L_MOTOR_A, LOW);
 digitalWrite(L_MOTOR_B, HIGH);
 digitalWrite(R_MOTOR_A, HIGH);
 digitalWrite(R_MOTOR_B, LOW);
}

void turnRight() {
 digitalWrite(L_MOTOR_A, HIGH);
 digitalWrite(L_MOTOR_B, LOW);
 digitalWrite(R_MOTOR_A, LOW);
 digitalWrite(R_MOTOR_B, HIGH);
}

void handleCommand(char cmd) {
 Serial.print("Received command: ");
 Serial.println(cmd);

 switch (cmd) {
   case 'W': moveForward(); break;
   case 'S': moveBackward(); break;
   case 'A': turnLeft(); break;
   case 'D': turnRight(); break;
   default: stopMotors(); break;
 }
}

void setup() {
 Serial.begin(115200);
 delay(500);

 pinMode(L_MOTOR_A, OUTPUT);
 pinMode(L_MOTOR_B, OUTPUT);
 pinMode(R_MOTOR_A, OUTPUT);
 pinMode(R_MOTOR_B, OUTPUT);

 stopMotors();

 Serial.println("\nStarting ESP8266 Robot Server");

 WiFi.softAP(ssid, password);

 Serial.print("AP IP address: ");
 Serial.println(WiFi.softAPIP());

 server.begin();
 Serial.println("TCP server started on port 8080");
}

void loop() {
 WiFiClient client = server.available();

 if (client) {
   Serial.println("Client connected");

   while (client.connected()) {
    if (client.available()) {
     char cmd = client.read();
     handleCommand(cmd);
    }
   }

   Serial.println("Client disconnected");
   stopMotors();       // SAFETY STOP
   client.stop();
 }
}
```

```
#!/usr/bin/env python3

import rclpy
from rclpy.node import Node
import socket
import sys
import select
import termios

import tty

import time


ESP_IP = "192.168.4.1"

ESP_PORT = 8080


KEY_MAP = {

  'w': 'W',      # forward

  's': 'S',      # reverse

  'a': 'A',      # left

  'd': 'D',      # right

  '\x1b[A': 'W',   # arrow up

  '\x1b[B': 'S',   # arrow down

  '\x1b[D': 'A',   # arrow left

  '\x1b[C': 'D',   # arrow right

}


SEND_RATE_HZ = 10   # how fast commands repeat when held


class KeyboardControl(Node):

  def __init__(self):

    super().__init__('keyboard_control')

    self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    self.sock.connect((ESP_IP, ESP_PORT))

    self.get_logger().info("Connected to ESP8266 robot")


  def send_cmd(self, cmd):
        self.sock.send(cmd.encode())


def get_key():
    tty.setraw(sys.stdin.fileno())
    r, _, _ = select.select([sys.stdin], [], [], 0.01)
    if r:
        key = sys.stdin.read(1)
        if key == '\x1b':  # arrow keys
            key += sys.stdin.read(2)
        return key
    return None


def main():
    rclpy.init()
    settings = termios.tcgetattr(sys.stdin)
    node = KeyboardControl()

    current_cmd = None
    last_send_time = 0.0
    send_interval = 1.0 / SEND_RATE_HZ

    try:
        while rclpy.ok():
            key = get_key()

            if key == 'q':
                break
            if key in KEY_MAP:
                current_cmd = KEY_MAP[key]

            now = time.time()

            # send repeatedly while key is held
            if current_cmd and (now - last_send_time) > send_interval:
                node.send_cmd(current_cmd)
                last_send_time = now

            # if no key detected, stop sending
            if key is None:
                current_cmd = None

    except KeyboardInterrupt:
        pass

    finally:
        termios.tcsetattr(sys.stdin, termios.TCSADRAIN, settings)
        node.destroy_node()
        rclpy.shutdown()

if __name__ == '__main__':
    main()
```

```
entry_points={
        'console_scripts': [
            'keyboard_control = robot_keyboard.keyboard_control:main',
        ],
```