Perfect 👍
Below is **exactly what you asked for**:

1️⃣ **FULL working ESP8266 (NodeMCU) AP-MODE code**
2️⃣ **VERY detailed beginner-friendly instructions**
3️⃣ **No router, no IP changes, ROS-ready**

Take it slow and follow step by step — nothing is assumed.

✅ WHAT YOU ARE BUILDING (CLEAR)

Your NodeMCU will:

- Create its **own Wi-Fi** (AP mode)
- Host a **web server**
- Accept commands from:
    - Browser
    - ROS 2 (keyboard joystick)
- Control motors

📌 Fixed IP forever: **192.168.4.1**

🔧 PART 1: COMPLETE ESP8266 AP-MODE CODE (UPLOAD THIS)

📌 **Copy EVERYTHING below and upload to NodeMCU**

```
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>
#include <ESP8266HTTPClient.h>
#include <WiFiUdp.h>

/* ==================== PIN CONFIG ==================== */
int M11 = D0;
int M12 = D1;
int M21 = D2;
int M22 = D3;
int speedPin1 = D5;
int speedPin2 = D4;

/* ===================== WIFI AP CONFIG ===================== */
const char* ap_ssid = "Robot_AP";
const char* ap_password = "robot123";

/* ===================== SERVER ===================== */
ESP8266WebServer server(80);
WiFiUDP Udp;
unsigned int localUdpPort = 4210;
char incomingPacket[255];

/* ===================== FUNCTION DECLARATIONS ===================== */
void fwd();
void rev();
void left();
void right();
void stopRobot();

/* ===================== HTML PAGE ===================== */
String getHTML() {
  return R"rawliteral(
<!DOCTYPE html>
<html>
<head>
  <title>Robot Control</title>
```

```
 <style>
  body { font-family: Arial; background:#111; color:white; text-align:center; }
  button { padding:20px; margin:10px; font-size:18px; width:120px; }
 </style>
</head>
<body>
 <h1>Robot Control</h1>
 <button onclick="fetch('/fwd')">FWD</button><br>
 <button onclick="fetch('/left')">LEFT</button>
 <button onclick="fetch('/stop')">STOP</button>
 <button onclick="fetch('/right')">RIGHT</button><br>
 <button onclick="fetch('/rev')">REV</button>
</body>
</html>
)rawliteral";
}

/* ===================== SETUP ===================== */
void setup() {
 Serial.begin(115200);

 pinMode(M11, OUTPUT);
 pinMode(M12, OUTPUT);
 pinMode(M21, OUTPUT);
 pinMode(M22, OUTPUT);
 pinMode(speedPin1, OUTPUT);
 pinMode(speedPin2, OUTPUT);
```

```
  analogWrite(speedPin1, 120);
  analogWrite(speedPin2, 120);

  /* -------- AP MODE -------- */
  WiFi.mode(WIFI_AP);
  WiFi.softAP(ap_ssid, ap_password);

  Serial.println("ESP8266 AP MODE STARTED");
  Serial.print("AP IP: ");
  Serial.println(WiFi.softAPIP());

  /* -------- WEB SERVER -------- */
  server.on("/", []() {
    server.send(200, "text/html", getHTML());
  });
  server.on("/fwd", fwd);
  server.on("/rev", rev);
  server.on("/left", left);
  server.on("/right", right);
  server.on("/stop", stopRobot);

  server.begin();
  Serial.println("HTTP server started");

  /* -------- UDP -------- */
  Udp.begin(localUdpPort);
  Serial.println("UDP server started");
}
```

```
/* ==================== MOTOR FUNCTIONS ==================== */

void fwd() {

  digitalWrite(M11, HIGH);

  digitalWrite(M12, LOW);

  digitalWrite(M21, HIGH);

  digitalWrite(M22, LOW);

  server.send(200, "text/plain", "Forward");

}


void rev() {

  digitalWrite(M11, LOW);

  digitalWrite(M12, HIGH);

  digitalWrite(M21, LOW);

  digitalWrite(M22, HIGH);

  server.send(200, "text/plain", "Reverse");

}


void left() {

  digitalWrite(M11, HIGH);

  digitalWrite(M12, LOW);

  digitalWrite(M21, LOW);

  digitalWrite(M22, HIGH);

  server.send(200, "text/plain", "Left");

}


void right() {

  digitalWrite(M11, LOW);
  digitalWrite(M12, HIGH);
  digitalWrite(M21, HIGH);
  digitalWrite(M22, LOW);
  server.send(200, "text/plain", "Right");
}

void stopRobot() {
  digitalWrite(M11, LOW);
  digitalWrite(M12, LOW);
  digitalWrite(M21, LOW);
  digitalWrite(M22, LOW);
  server.send(200, "text/plain", "Stopped");
}

/* ===================== LOOP ===================== */
void loop() {
  server.handleClient();

  int packetSize = Udp.parsePacket();
  if (packetSize) {
    int len = Udp.read(incomingPacket, 255);
    if (len > 0) incomingPacket[len] = 0;

    String cmd = String(incomingPacket);
    Serial.println("UDP: " + cmd);

    if (cmd == "fwd") fwd();
    else if (cmd == "rev") rev();
    else if (cmd == "left") left();
    else if (cmd == "right") right();
    else if (cmd == "stop") stopRobot();
  }
}
```

🧭 PART 2: VERY DETAILED INSTRUCTIONS (BEGINNER MODE)

🔹 STEP 1: Upload Code to NodeMCU

1. Open **Arduino IDE**
2. Select:
   - Board → **NodeMCU 1.0 (ESP-12E)**
   - Port → correct COM port
3. Paste the code
4. Click **Upload**
5. Open **Serial Monitor**
   - Baud: **115200**

You should see:

```
ESP8266 AP MODE STARTED
AP IP: 192.168.4.1
```

✅ This confirms success

🔹 STEP 2: Connect Laptop to ESP Wi-Fi

1. Open Wi-Fi settings
2. Connect to:
3. WiFi name: Robot_AP
4. Password: robot123

⚠️ No internet — this is NORMAL

🔹 STEP 3: Test Without ROS (IMPORTANT)

Open browser and go to:

```
http://192.168.4.1
```

Click buttons:

- FWD
- LEFT
- RIGHT
- STOP

✅ Robot must move
❌ If not → fix wiring first (ROS later)

🔹 STEP 4: ROS Python Code (ONLY 1 LINE CHANGE)

In your ROS Python node:

```
ESP_IP = "192.168.4.1"
```

👉 **This NEVER changes again**

🔹 STEP 5: Run ROS Keyboard Joystick

Terminal 1

```
source /opt/ros/jazzy/setup.bash

source ~/ros2_ws/install/setup.bash

ros2 run robot_teleop_bridge cmdvel_to_esp
```

Terminal 2

```
ros2 run teleop_twist_keyboard teleop_twist_keyboard
```

Click inside terminal and press:

- w → forward
- a → left
- d → right

- s → reverse
- space → stop

🎉 ROBOT MOVES USING KEYBOARD

🧠 IMPORTANT BEGINNER NOTES

✔ ESP code uploaded **only once**
✔ No router needed
✔ No IP updates ever
✔ ROS works offline
✔ Best setup for learning & demos

🔥 NEXT SAFE UPGRADES (WHEN READY)

Say any one:

- add speed control
- real joystick (USB)
- smooth turning
- cmd_vel proper diff drive
- obstacle avoidance

You're doing this the **RIGHT WAY** 👏

Perfect 👍

Below is **THE COMPLETE PYTHON ROS 2 CODE + VERY DETAILED “WHAT TO DO” STEPS** assuming:

✅ You have **already uploaded the ESP8266 AP-mode Arduino code**
✅ You are a **complete beginner**
✅ You want **keyboard (WASD / Arrow) joystick control**

Take it slowly and follow **exactly in order**.

🧠 WHAT THIS PYTHON CODE DOES (IN SIMPLE WORDS)

- Listens to **keyboard commands** from ROS
- Converts them into **simple words** (fwd, left, right, rev, stop)
- Sends them to ESP8266 at:
- http://192.168.4.1/<command>

ESP already understands these commands.

📁 PART 1: ROS PYTHON CODE (COPY THIS EXACTLY)

📄 File name

cmdvel_to_esp.py

📄 Location

~/ros2_ws/src/robot_teleop_bridge/robot_teleop_bridge/

✅ FULL PYTHON CODE (READY TO RUN)

```
import rclpy
from rclpy.node import Node
from geometry_msgs.msg import Twist
import requests


ESP_IP = "192.168.4.1"   # FIXED in AP mode
```

```
class CmdVelToESP(Node):

  def __init__(self):
    super().__init__('cmdvel_to_esp')
    self.subscription = self.create_subscription(
      Twist,
      '/cmd_vel',
      self.cmdvel_callback,
      10
    )
    self.last_cmd = "stop"
    self.get_logger().info("Keyboard joystick → ESP bridge started")

  def cmdvel_callback(self, msg):
    linear = msg.linear.x
    angular = msg.angular.z

    if linear > 0.1:
      cmd = "fwd"
    elif linear < -0.1:
      cmd = "rev"
    elif angular > 0.1:
      cmd = "left"
    elif angular < -0.1:
      cmd = "right"
    else:
      cmd = "stop"
```

```
        if cmd != self.last_cmd:
            self.send_cmd(cmd)
            self.last_cmd = cmd

    def send_cmd(self, cmd):
        try:
            url = f"http://{ESP_IP}/{cmd}"
            requests.get(url, timeout=0.5)
            self.get_logger().info(f"Sent command: {cmd}")
        except Exception as e:
            self.get_logger().error(f"ESP not reachable: {e}")

def main():
    rclpy.init()
    node = CmdVelToESP()
    rclpy.spin(node)
    node.destroy_node()
    rclpy.shutdown()

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

🧭 PART 2: VERY DETAILED STEPS (FROM ZERO)

🔹 STEP 1: Connect Laptop to ESP Wi-Fi

1. Turn ON robot
2. On laptop Wi-Fi:
3. Network: Robot_AP

4. Password: robot123

5. No internet → this is OK

🔹 STEP 2: Verify ESP is Working (MANDATORY)

Open browser and go to:

http://192.168.4.1

Click buttons:

- FWD
- LEFT
- RIGHT
- STOP

❌ If robot doesn't move → **DO NOT GO TO ROS YET**

🔹 STEP 3: Create ROS Package (Only Once)

```
cd ~/ros2_ws/src

ros2 pkg create robot_teleop_bridge \

 --build-type ament_python \

 --dependencies rclpy geometry_msgs
```

🔹 STEP 4: Add Python File

```
cd ~/ros2_ws/src/robot_teleop_bridge/robot_teleop_bridge

nano cmdvel_to_esp.py
```

Paste the code → CTRL+O → Enter → CTRL+X

🔹 STEP 5: Make File Executable

```
chmod +x cmdvel_to_esp.py
```

🔹 STEP 6: Register Node (VERY IMPORTANT)

Open:

```
nano ~/ros2_ws/src/robot_teleop_bridge/setup.py
```

Find entry_points and change to:

```
entry_points={
    'console_scripts': [
        'cmdvel_to_esp = robot_teleop_bridge.cmdvel_to_esp:main',
    ],
},
```

Save & exit.

🔹 STEP 7: Install Python Library

```
pip3 install requests
```

If error:

```
python3 -m venv venv

source venv/bin/activate

pip install requests
```

🔹 STEP 8: Build ROS Workspace

```
cd ~/ros2_ws

colcon build

source install/setup.bash
```

⚠️ Do this after EVERY code change

🎮 PART 3: RUN JOYSTICK MODE (3 TERMINALS)

🟢 TERMINAL 1 – ROS ESP Bridge

```
source /opt/ros/jazzy/setup.bash
```

```
source ~/ros2_ws/install/setup.bash
```

```
ros2 run robot_teleop_bridge cmdvel_to_esp
```

Expected:

```
Keyboard joystick → ESP bridge started
```

🟢 TERMINAL 2 – Keyboard Control

```
ros2 run teleop_twist_keyboard teleop_twist_keyboard
```

You will see key instructions.

⚠️ CLICK INSIDE THIS TERMINAL BEFORE PRESSING KEYS

🟢 TERMINAL 3 – (OPTIONAL DEBUG)

```
ros2 topic echo /cmd_vel
```

🎉 DONE — YOUR ROBOT IS NOW A ROS JOYSTICK ROBOT

Key	Robot
w	Forward
a	Left
d	Right
s	Reverse
space	Stop

❗ COMMON BEGINNER MISTAKES (READ)

❌ Forgot to connect to Robot_AP
❌ Forgot to source install/setup.bash
❌ ESP IP wrong
❌ Robot wiring issue

🔥 NEXT SAFE UPGRADE PATH

Say ONE:

- speed control
- smooth turning
- USB joystick
- voice control
- gazebo → real robot
- obstacle avoidance

You’re learning ROS the **correct professional way** 👏