```
#include <Arduino.h>

// Wi-Fi settings (manual handling, could be done with AT commands)
#define SSID "Your_SSID"
#define PASSWORD "Your_PASSWORD"

// Stepper motor pins
#define BASE_STEP_PIN 19
#define BASE_DIR_PIN 18
#define SHOULDER_STEP_PIN 22
#define SHOULDER_DIR_PIN 21

// Servo control pins
#define UART_TX_PIN 17
#define UART_RX_PIN 16
HardwareSerial servoSerial(1); // UART for serial servo communication

// Load cell pins
#define LOADCELL_DT 4
#define LOADCELL_SCK 5

// I2C (For PCA9685 servo driver)
#define SDA_PIN 21
#define SCL_PIN 22
#define PCA9685_ADDR 0x40

// Inverse Kinematics Variables
float homeX = 0, homeY = 0, homeZ = 0; // Home position
float towerAX = -10, towerAY = 10, towerAZ = 5;
float towerBX = 10, towerBY = 10, towerBZ = 5;

// Helper Functions
void delayMicrosecondsCustom(int us) {
  // Bare-metal delay function
  unsigned long startTime = micros();
  while (micros() - startTime < us);
}

void smoothStep(int stepPin, int dirPin, int steps, int delayMicros) {
  for (int i = 0; i < steps; i++) {
    digitalWrite(stepPin, HIGH);
    delayMicrosecondsCustom(delayMicros);
    digitalWrite(stepPin, LOW);
    delayMicrosecondsCustom(delayMicros);
  }
}

void moveSerialServo(uint8_t servoID, uint16_t position, uint16_t speed) {
```

```
  uint8_t packet[] = {
    0xFF, 0xFF, servoID, 0x09, 0x03, 0x2A,
    position & 0xFF, (position >> 8) & 0xFF,
    0x00, 0x00, speed & 0xFF, (speed >> 8) & 0xFF, 0x00
  };
  packet[12] = ~((servoID + 0x09 + 0x03 + 0x2A +
                (position & 0xFF) + ((position >> 8) & 0xFF) +
                (speed & 0xFF) + ((speed >> 8) & 0xFF)) & 0xFF);
  for (uint8_t i = 0; i < sizeof(packet); i++) {
    servoSerial.write(packet[i]);
  }
}

void moveI2CServo(uint8_t channel, int angle) {
  int pwm = map(angle, 0, 180, 80, 600);
  Wire.beginTransmission(PCA9685_ADDR);
  Wire.write(0x06 + 4 * channel);
  Wire.write(pwm & 0xFF);
  Wire.write(pwm >> 8);
  Wire.endTransmission();
}

// Inverse Kinematics Function
void calculateIK(float x, float y, float z, int& baseSteps, int& shoulderSteps, int& elbowPos) {
  float baseAngle = atan2(y, x); // Base angle in radians
  float r = sqrt(x * x + y * y); // Radius in XY plane
  float l1 = 10.0; // Length of shoulder segment
  float l2 = 10.0; // Length of elbow segment
  float D = (r * r + z * z - l1 * l1 - l2 * l2) / (2 * l1 * l2);

  baseSteps = baseAngle * 200 / PI; // Convert radians to steps for base
  shoulderSteps = acos(D) * 200 / PI; // Convert radians to steps for shoulder
  elbowPos = 1500 + z * 10; // Simplified calculation for elbow servo
}

// Movement Routine
void executeMovement(float x, float y, float z) {
  int baseSteps, shoulderSteps, elbowPos;

  // Calculate inverse kinematics
  calculateIK(x, y, z, baseSteps, shoulderSteps, elbowPos);

  // Move base and shoulder
  digitalWrite(BASE_DIR_PIN, baseSteps > 0 ? HIGH : LOW);
  smoothStep(BASE_STEP_PIN, BASE_DIR_PIN, abs(baseSteps), 500);

  digitalWrite(SHOULDER_DIR_PIN, shoulderSteps > 0 ? HIGH : LOW);
  smoothStep(SHOULDER_STEP_PIN, SHOULDER_DIR_PIN, abs(shoulderSteps), 500);
```

```
  // Move elbow
  moveSerialServo(0x01, elbowPos, 1000);
}

void setup() {
  // Start serial communication
  Serial.begin(115200);

  // Set up GPIO for motors and load cells
  pinMode(BASE_STEP_PIN, OUTPUT);
  pinMode(BASE_DIR_PIN, OUTPUT);
  pinMode(SHOULDER_STEP_PIN, OUTPUT);
  pinMode(SHOULDER_DIR_PIN, OUTPUT);

  // Servo control setup
  servoSerial.begin(1000000, SERIAL_8N1, UART_RX_PIN, UART_TX_PIN);

  // Load cell setup
  pinMode(LOADCELL_DT, INPUT);
  pinMode(LOADCELL_SCK, OUTPUT);

  // I2C setup
  Wire.begin(SDA_PIN, SCL_PIN);

  // Move to home position
  executeMovement(homeX, homeY, homeZ);
}

void loop() {
  // Simple placeholder logic for loop (no WebSocket here, purely hardware control)
  // You can add logic here for external commands, like reading inputs or controlling
actuators.

  // For example, moving to a tower position:
  executeMovement(towerAX, towerAY, towerAZ);
  delay(2000);
  executeMovement(towerBX, towerBY, towerBZ);
  delay(2000);
}
```