Final Project – Step-by-step Guide

Here's a complete recipe for your **Final Project – Motion / Sound Detection with Arduino Uno R3** using **Ultrasonic + PIR + Red/Green LEDs in Simulink**.

I'll assume this logic:

- If **object is within a chosen distance AND motion is detected → GREEN LED ON, RED OFF**
- Otherwise → **RED LED ON, GREEN OFF**

1️⃣ Hardware Connections (step-by-step wiring)

A. Pin plan (so we stay consistent)

Device	Arduino Pin
PIR motion signal	**D2**
Ultrasonic Trigger	**D3**
Ultrasonic Echo	**D4**
Green LED	**D9**
Red LED	**D10**
Power for sensors	**5V & GND**

Use ~220 Ω resistor for each LED.

B. Set up the breadboard power rails

1. Plug the **Arduino Uno** into USB (for power & programming).
2. On the breadboard:
   - Connect **Arduino 5V → breadboard + rail**.
   - Connect **Arduino GND → breadboard – rail**.
   - Now all components can share 5V and GND from those rails.

C. Wire the PIR Motion Sensor

Typical PIR pins (left→right when facing the dome, but check your module marking):

- **VCC / +**
- **OUT / SIG**
- **GND / –**

Steps:

1. PIR **VCC → Breadboard + (5V)**.
2. PIR **GND → Breadboard – (GND)**.
3. PIR **OUT/SIG → Arduino D2** (use a male–male jumper).

D. Wire the Ultrasonic Sensor (HC-SR04)

Pins: **VCC, TRIG, ECHO, GND**

1. **VCC → Breadboard + (5V)**.
2. **GND → Breadboard – (GND)**.
3. **TRIG → Arduino D3**.
4. **ECHO → Arduino D4**.
   (HC-SR04 "5V" echo is okay for the Uno's 5V logic.)

E. Wire the LEDs with resistors

We'll use **active-HIGH**: writing logic 1 turns LED on.

Green LED (near + motion)

1. Place the **Green LED** on the breadboard (long leg = anode, short leg = cathode).
2. Connect **anode (long leg)** → one side of a **220 Ω resistor**.
3. Other side of resistor → **Arduino D9**.
4. **Cathode (short leg)** → breadboard **GND rail**.

Red LED (far / no motion)

1. Place the **Red LED** on the breadboard.
2. **Anode → 220 Ω resistor → Arduino D10**.
3. **Cathode → breadboard GND rail**.

Double-check:

- All GND pins (PIR, HC-SR04, LEDs) → same ground as Arduino.

- No component is directly from 5V to a digital pin without a resistor.

At this point the hardware is ready.

2️⃣ Simulink Diagram Layout & Pin Settings

Overall logic in blocks

1. **Ultrasonic Sensor block** → distance (m)
2. **Compare to Constant** (`< threshold`) → `isNear`
3. **Digital Input block (PIR)** → `motion`
4. **Logical Operator (AND)** : `isNear` AND `motion` → `greenCondition`
5. **Logical Operator (NOT)** of `greenCondition` → `redCondition`
6. **Digital Output block (D9)** ← `greenCondition`
7. **Digital Output block (D10)** ← `redCondition`

Block-by-block settings

All Arduino-specific blocks are under:

Simulink Library Browser ▸ Simulink Support Package for Arduino Hardware

a) Ultrasonic Sensor block

- Block: **Ultrasonic Sensor**
- Parameters:
  - **Trigger pin:** `D3`
  - **Echo pin:** `D4`
  - **Sample time:** `0.1` (for 10 Hz updates, you can adjust)

Output: distance in meters (double).

b) PIR Motion – Digital Input

- Block: **Digital Input**
- Parameters:
  - **Pin:** `D2`
  - **Sample time:** `0.1` (same as ultrasonic)
- Output data type: `boolean` (or `uint8` and treat non-zero as true).

c) Distance threshold – Compare to Constant

- Block: **Relational Operator** or **Compare To Constant**
    - Input: the **distance** from Ultrasonic block.
    - Operator: `<`
    - Constant value: e.g. `0.3` (meters) → "object is near if within 30 cm."
- Output: `isNear` (boolean).

d) Logical combination

- Block 1: **Logical Operator (AND)**
    - Inputs: `isNear` and `motion`.
    - Operator: `AND`
    - Output: `greenCondition`.
- Block 2: **Logical Operator (NOT)**
    - Input: `greenCondition`
    - Operator: `NOT`
    - Output: `redCondition`.

e) LED Outputs – Digital Output blocks

- Block: **Digital Output** (one block per LED).

Green LED block:

- Pin: `D9`
- Sample time: `0.1`
- Input: `greenCondition`.

Red LED block:

- Pin: `D10`
- Sample time: `0.1`
- Input: `redCondition`.

That's the full diagram.

3️⃣ Step-by-Step: Build the Simulink Model

Step 1 – Start a new model

1. In MATLAB, type: `simulink` and press Enter.
2. Click **Blank Model**.
3. Save it immediately as e.g. `MotionUltrasonicLED.slx`.

Step 2 – Configure the hardware (Arduino Uno)

1. In the Simulink model window, go to
   Modeling ‣ Model Settings… (or `Ctrl+E`).
2. In the left pane select **Hardware Implementation**.
3. Under **Hardware board**, choose **`Arduino Uno`**.
4. MATLAB will auto-fill other options. Make sure:
   - **Arduino IDE / toolchain** is found (from earlier setup).
5. Optional: Under **Target Hardware Resources ‣ External mode**, enable External mode if you want live tuning/monitoring.
6. Click **OK** to close the dialog.

Step 3 – Add Arduino blocks

1. Open the **Library Browser**.
2. Navigate to **Simulink Support Package for Arduino Hardware ‣ Common**.
3. Drag and drop the following into your model:
   - **Ultrasonic Sensor**
   - **Digital Input**
   - **Two Digital Output** blocks.

Step 4 – Add processing / logic blocks

From the standard Simulink library:

1. **Simulink ‣ Logic and Bit Operations**
   - **Logical Operator** (add two of these; set one to AND, one to NOT).
2. **Simulink ‣ Relational Operators**
   - **Compare To Constant** (or **Relational Operator**).
3. (Optional) **Display** or **Scope** from **Sinks** to monitor distance or logic signals while in External mode.

Step 5 – Configure each block

Ultrasonic block

- Double-click it:
  - Set **Trigger pin** to `D3`.

    - Set **Echo pin** to `D4`.
    - Set **Sample time** to `0.1`.

Digital Input (PIR)

- Double-click:
    - **Pin**: `D2`.
    - **Sample time**: `0.1`.

Compare To Constant

- Double-click:
    - **Operator**: `<`.
    - **Constant value**: e.g. `0.3` (meters).
    - Output data type: `boolean` if available.

Logical Operator (AND)

- Double-click:
    - **Operator**: `AND`.
    - **Number of input ports**: `2`.

Logical Operator (NOT)

- Double-click:
    - **Operator**: `NOT`.

Digital Output (Green LED)

- Double-click:
    - **Pin:** `D9`.
    - **Initial value:** `0`.
    - **Sample time:** `0.1`.

Digital Output (Red LED)

- Double-click:
    - **Pin:** `D10`.
    - **Initial value:** `0`.
    - **Sample time:** `0.1`.

Step 6 – Wire the blocks

Connect the lines as follows:

1. **Ultrasonic Sensor output → Compare To Constant input**.
2. **Compare To Constant output → Logical AND input 1** (`isNear`).
3. **Digital Input (PIR) output → Logical AND input 2** (`motion`).
4. **Logical AND output (greenCondition)**:
   - **→ Digital Output (D9 – Green LED)**.
   - **→ Logical NOT input**.
5. **Logical NOT output (redCondition) → Digital Output (D10 – Red LED)**.

(Optional) Also connect the Ultrasonic distance to a **Display/Scope** to see live distance in meters.

Step 7 – Choose run mode & deploy

You have two main options:

A. Run in External mode (live interaction)

1. On the Simulink toolbar, set **Simulation mode** to **External**.
2. Click the **Connect to Target** button (plug icon).
3. Then click **Run**.
4. The model compiles, downloads to the Arduino, and Simulink stays connected so you can view scopes and change constants in real time.

B. Deploy as standalone on Arduino

1. In the model window, click **Deploy To Hardware** (the blue “play-on-chip” icon).
2. Simulink generates code, flashes the Uno, and then your board runs the logic by itself (no PC needed once running).

Step 8 – Test behavior

1. Power the Arduino via USB or external 5V.
2. Put an object within your threshold distance (e.g., within 30 cm) **and** move in front of the PIR:
   - **GREEN LED** should turn ON.
   - **RED LED** should turn OFF.
3. Move away or block motion:
   - **RED LED** ON, **GREEN** OFF.

You can adjust:

- The **distance threshold** constant (e.g., 0.2 m vs 0.5 m).

- Sample times for faster/slower response.

If you’d like, next I can:

- Draw a simple ASCII or block-style diagram you can mirror in Simulink, **or**
- Add a “sound” element using a buzzer on another pin, controlled by the same logic.