

Card → Bot Build Book

A companion to **Loop Slap** and **Bot-Builders**. Play a loop at the table, then build its living version with the Elegoo ESP-32 Super Starter Kit — adaptations welcome, honesty required.

Play → **inspire** → **build** → **recite the If/Then**.
hibot.space/games

[Same kit as the ESP32 Project Book](#) ›

How to use this book

Close the gap between “we shouted BOT!” and “it actually beeps.”

THE LOOP

1. **Play** Loop Slap or Bot-Builders. When someone completes a Sense / Think / Act (/ Power) set, keep those cards face-up.
2. **Find** the matching build in the crosswalk (next page) — or pick the mission that matches your Bot-Builders Mission card.
3. **Adapt out loud.** If the card says Camera and you are building with PIR, say: “Our kit stand-in for Camera is motion.” That sentence is the lesson.
4. **Build** on USB power first. Recite the If/Then breath from the cards while it runs.
5. **Stretch** with “Try this next” — mash two projects to cover more of the mission.

WHAT “ADAPTATION” MEANS

The card games invent boldly (LiDAR, grippers, solar, drive wheels). The Super Starter is a real box of parts. This book keeps the **Sense** → **Think** → **Act** → **Power** story and substitutes kit-honest hardware. Desktop bots count. Rovers without wheels are still loops.

KIT

Official **Elegoo ESP-32 Super Starter Kit** — same inventory as the ESP32 Project Book / Activities planner. Pair that book for deeper wiring drills; use **this** book when the cards are still on the table.

Parent / coach tip: After each build, ask: Which card was Sense? Think? Act? Power? If they can point without looking at this page, the games worked.

Card → project crosswalk

Start here after a round. Full parts lists live on each project page. The LOOP deck is 16 cards — where it has no card for a build's Sense, this table says so rather than inventing one.

BUILD	CARDS / MISSIONS IT LIVES
1. Night Watch Light	Bot-Builders: Light sensor · If/Then board · LED array · USB power · Mission Night watch (M-06). Loop Slap: "IF/THEN" Code Script + LED Display Screen. No light-sense card in the 16 — this build's Sense is Bot-Builders only.
2. Clap Cue Bot	Bot-Builders: Microphone · If/Then · LED array. Loop Slap: "IF/THEN" Code Script + LED Display Screen. No sound-sense card in the 16 — the Microphone is Bot-Builders only.
3. Bumper Guard	Bot-Builders: Ultrasonic ranger · If/Then · Speaker · LED. Missions: Rescue rover / Delivery bot (desk stand-in). Loop Slap: ultrasonic bumper LOOP.
4. Desk Weather Station	Bot-Builders: Thermometer · Microcontroller · Display · USB · Mission Weather station (M-02). Loop Slap: thermometer + display.
5. Sweeping Sentinel	Bot-Builders: Ultrasonic ranger · Servo arm · Speaker · If/Then. Missions Night watch / Rescue rover (scanning stand-in). Loop Slap: ultrasonic + actuator.
6. Hallway Motion Watch	Bot-Builders: (motion stand-in for Camera) · If/Then · Speaker · LED · Mission Night watch. Loop Slap: HD Camera Lens (your PIR is the kit stand-in) + Audio Speaker.
7. Tip-Over Alarm	Bot-Builders: Tilt sensor · Safety lock · Speaker · LED. Loop Slap: Audio Speaker + LED Display Screen. No tilt-sense card in the 16 — the Tilt sensor is Bot-Builders only.
8. Spill Watch	Bot-Builders: (moisture stand-in) · If/Then · Speaker · Mission Garden waterer (M-03) adapted. Loop Slap: Audio Speaker. No water-sense card in the 16 — this build's Sense is Bot-Builders only.
9. Hot-Day Fan Bot	Bot-Builders: Thermometer · If/Then · Fan · USB. Missions Weather station + Lab partner. Loop Slap: thermometer + DC motor.
10. Badge Door Helper	Bot-Builders: (tag sense) · Safety lock / If/Then · LED · (relay as the Act). Mission Classroom helper adapted. Loop Slap: "IF/THEN" Code Script + LED Display Screen. No badge/identity card in the 16 — the tag read is Bot-Builders only.
11. Story Beeper	Bot-Builders: If/Then · Timer · Speaker · USB · Mission Storyteller (M-04). Loop Slap: speaker / think.
12. Rescue Beacon	Bot-Builders: Ultrasonic or PIR · If/Then · Servo arm · Speaker · LED · Mission Rescue rover (M-01). Loop Slap: full SENSE → THINK → ACT breath.

MISSIONS AT A GLANCE

- **Weather station** → Project 4 (+ 9 for an Act).
- **Night watch** → 1, then 6, mash for dark+motion.
- **Rescue rover** → 3 / 5 / 12 (desk beacon, not a chassis).
- **Garden waterer** → 8 (spill watch; pump is Phase 2).
- **Storyteller** → 11.
- **Classroom helper** → 10 (badge door).
- **Lab partner** → 4 + 9.
- **Delivery bot** → 3 as "don't hit the wall" brain only.

Honest gaps

Still great cards — not Super Starter one-box builds yet.

CARD IDEA	WHY IT WAITS	CLOSEST NOW
Camera / color sensor	Not in Super Starter	PIR or LDR as “something changed”
LiDAR turret	No LiDAR module	Project 5 sweeping ultrasonic
Drive wheels / delivery	No chassis / motor driver pair for drive	Bumper Guard + Beacon
Gripper / pump	No gripper or pump	Servo wave / Spill Watch
Solar / fuel cell / dock	USB + breadboard PSU only	Name USB as Power card aloud
Map module / neural net	Software concepts	If/Then + Serial “memory” variables

POWER CARD, EVERY TIME

Unless a project says otherwise, **Power = USB cable** (P-02). When you use the breadboard 5V supply for motors/servos, that is also your Voltage regulator / Power story — grounds joined with the ESP32.

Golden rules (same as ESP32 Project Book): 3.3V logic on GPIOs; HC-SR04 ECHO through a divider; RC522 at 3.3V only; no wall power on relays; motors via transistor or driver, never bare GPIO.

1 · Night Watch Light

When the room goes dark, the bot wakes a light — desk edition of Night watch.

IF / THEN BREATH

If the room is dark, then turn the light on.

Cards: Bot-Builders: Light sensor · If/Then board · LED array · USB power · Mission Night watch (M-06). Loop Slap: “IF/THEN” Code Script + LED Display Screen. No light-sense card in the 16 — this build’s Sense is Bot-Builders only.

Adaptation: Card “Night watch” wants motion in the dark. v1 watches light only. Add Project 6 (PIR) later for the full mission.

PARTS

- ESP32 + breadboard
- Photoresistor (LDR)
- 10K resistor
- LED + 220Ω
- Jumpers
- USB cable (Power)

WIRING

ESP32 PIN	CONNECTS TO
3V3	One leg of the photoresistor
GPIO 35	Other photoresistor leg AND one leg of the 10K (junction)
GND	Other leg of the 10K
GPIO 23	LED anode → 220Ω → GND

HOW IT WORKS

Sense: the LDR + 10K form a voltage divider into an ADC pin. Think: if the reading means “dark,” Act: turn the LED on. Power: USB from your laptop. Say the Loop Slap breath out loud: “If it is dark, then light the LED — it thinks with If/Then.”

THE CODE

```
const int LDR = 35, LED = 23;
const int DARK = 1200; // tune with Serial

void setup() {
  pinMode(LED, OUTPUT);
  Serial.begin(115200);
}

void loop() {
  int light = analogRead(LDR); // 0..4095
  Serial.println(light);
  digitalWrite(LED, light < DARK ? HIGH : LOW);
  delay(100);
}
```

TRY THIS NEXT

Tape a cardboard “cave” over the LDR and watch the bot wake. Then lower DARK until it only fires when you really cover it.

TROUBLESHOOTING

- **Always on:** your room is already dim — raise DARK, or Serial-print and pick a threshold between open-hand and covered.
- **Always off:** LDR legs swapped relative to 3V3/GND, or LED backwards.
- Pin 35 is input-only — perfect for the sensor, never for the LED.

2 · Clap Cue Bot

A clap (or shout) flips the light — Microphone → If/Then → LED.

IF / THEN BREATH

If it hears a clap, then flip the light.

Cards: Bot-Builders: Microphone · If/Then · LED array. Loop Slap: “IF/THEN” Code Script + LED Display Screen. No sound-sense card in the 16 — the Microphone is Bot-Builders only.

Adaptation: The kit sound module hears loudness, not words. “Speaker” on the Act card becomes an LED cue here; add the active buzzer for a beep reply.

PARTS

- Sound / mic module
- LED + 220Ω
- Optional: active buzzer
- Breadboard + jumpers

WIRING

ESP32 PIN	CONNECTS TO
3V3 / GND	Sound module VCC / GND
GPIO 27	Sound module DO (digital out)
GPIO 23	LED → 220Ω → GND
GPIO 19 (optional)	Active buzzer + ; buzzer – → GND

HOW IT WORKS

Sense: DO goes LOW (or HIGH — check yours) when sound crosses the onboard trim pot. Think: toggle a remembered state on each clap edge. Act: LED (and optional beep). Twist the blue pot until a clap trips DO but room hush does not.

THE CODE

```
const int MIC = 27, LED = 23, BUZZ = 19;
bool on = false;
int last = HIGH;

void setup() {
  pinMode(MIC, INPUT_PULLUP);
  pinMode(LED, OUTPUT);
  pinMode(BUZZ, OUTPUT);
}

void loop() {
  int now = digitalRead(MIC);
  // Falling edge = clap (many modules pull D0 low when triggered)
  if (last == HIGH && now == LOW) {
    on = !on;
    digitalWrite(LED, on ? HIGH : LOW);
    digitalWrite(BUZZ, HIGH); delay(40); digitalWrite(BUZZ, LOW);
    delay(250); // ignore bounce / echo claps
  }
  last = now;
}
```

TRY THIS NEXT

Count two claps to turn on and one to turn off — a tiny “filter” (Think: Signal filter card).

TROUBLESHOOTING

- **Triggers constantly:** pot too sensitive, or DO polarity inverted — try detecting LOW → HIGH instead.
- **Never triggers:** pot too dull; clap nearer the mic; confirm VCC is 3V3.

3 · Bumper Guard

Beeps faster as things get closer — Ultrasonic ranger desk bot.

IF / THEN BREATH

If something is closer than 40 cm, then warn louder.

Cards: Bot-Builders: Ultrasonic ranger · If/Then · Speaker · LED. Missions: Rescue rover / Delivery bot (desk stand-in). Loop Slap: ultrasonic bumper LOOP.

Adaptation: No drive wheels in the Super Starter. This is the “don’t crash” brain of a rover, bolted to the table. Wave your hand like a wall.

PARTS

- HC-SR04 ultrasonic
- Active buzzer
- Green + red LEDs, 2× 220Ω
- 1K + 2K resistors (ECHO divider)

WIRING

ESP32 PIN	CONNECTS TO
VIN (5V)	HC-SR04 VCC
GND	HC-SR04 GND, buzzer –, LED grounds
GPIO 5	HC-SR04 TRIG
GPIO 18	ECHO via 1K from ECHO, and 2K from GPIO 18 to GND
GPIO 19	Active buzzer +
GPIO 21 / 22	Green LED / Red LED (each → 220Ω → GND)

HOW IT WORKS

Sense: chirp + echo → distance. Think: map distance to “safe / warn.” Act: green = clear, red + faster beeps = too close. Safety: ECHO is 5V — the divider protects the ESP32.

THE CODE

```
const int TRIG = 5, ECHO = 18, BUZZ = 19;
const int GREEN = 21, RED = 22;

void setup() {
  pinMode(TRIG, OUTPUT); pinMode(ECHO, INPUT);
  pinMode(BUZZ, OUTPUT); pinMode(GREEN, OUTPUT); pinMode(RED, OUTPUT);
  Serial.begin(115200);
}

float readCM() {
  digitalWrite(TRIG, LOW); delayMicroseconds(2);
  digitalWrite(TRIG, HIGH); delayMicroseconds(10);
  digitalWrite(TRIG, LOW);
  long us = pulseIn(ECHO, HIGH, 30000);
  return us * 0.0343 / 2.0;
}

void loop() {
  float cm = readCM();
  Serial.println(cm);
  if (cm > 0 && cm < 40) {
    digitalWrite(RED, HIGH); digitalWrite(GREEN, LOW);
    digitalWrite(BUZZ, HIGH); delay(30); digitalWrite(BUZZ, LOW);
    delay(map((int)cm, 0, 40, 40, 400));
  } else {
    digitalWrite(GREEN, HIGH); digitalWrite(RED, LOW);
    delay(100);
  }
}
```

TRY THIS NEXT

Add a “safety lock”: below 8 cm, hold a solid tone and refuse to clear until distance > 20 cm.

TROUBLESHOOTING

- **Always 0:** TRIG/ECHO swapped, or VCC on 3V3 instead of VIN.
- **Wild numbers:** aim at a flat book, not a soft sweater.

4 · Desk Weather Station

Measure the day and show it — the closest 1:1 mission in the kit.

IF / THEN BREATH

If it measures the air, then show the numbers (and warn if it is too hot).

Cards: Bot-Builders: Thermometer · Microcontroller · Display · USB · Mission Weather station (M-02). Loop Slap: thermometer + display.

Adaptation: Nearly literal. “Safe to go out” is your rule in code (e.g. warn if hotter than 28°C).

PARTS

- DHT11
- LCD1602 + I2C backpack
- 10K pull-up (if needed)
- Breadboard + jumpers

WIRING

ESP32 PIN	CONNECTS TO
VIN (5V)	LCD VCC
3V3	DHT11 VCC (+), one leg of 10K pull-up
GND	LCD GND, DHT11 –
GPIO 21 / 22	LCD SDA / SCL
GPIO 4	DHT11 data (S) + other leg of 10K pull-up

HOW IT WORKS

Sense: DHT11 temperature + humidity. Think: format + optional threshold. Act: LCD text. Libraries: Adafruit DHT + LiquidCrystal I2C (Frank de Brabander).

THE CODE

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <DHT.h>

LiquidCrystal_I2C lcd(0x27, 16, 2); // try 0x3F if blank
DHT dht(4, DHT11);

void setup() {
  lcd.init(); lcd.backlight();
  dht.begin();
}

void loop() {
  float t = dht.readTemperature();
  float h = dht.readHumidity();
  lcd.clear();
  if (isnan(t) || isnan(h)) { lcd.print("Sensor error"); delay(2000); return; }
  lcd.setCursor(0,0); lcd.print("Temp "); lcd.print(t,1); lcd.print("C");
  lcd.setCursor(0,1);
  if (t >= 28) lcd.print("HOT - stay cool");
  else { lcd.print("Hum  "); lcd.print(h,0); lcd.print("% OK"); }
  delay(2000);
}
```

TRY THIS NEXT

Swap the LCD for the OLED (Project path in the ESP32 Project Book) or push readings to a phone page (WiFi weather).

TROUBLESHOOTING

- **Sensor error:** add/check 10K data → VCC pull-up; wait 2s between reads.
- **Blank LCD:** try I2C address 0x3F; twist contrast pot on the backpack.

5 · Sweeping Sentinel

A servo aims the ultrasonic “eye” — radar for Night watch / Rescue.

IF / THEN BREATH

If the sweep sees something close, then sound the alarm.

Cards: Bot-Builders: Ultrasonic ranger · Servo arm · Speaker · If/Then. Missions Night watch / Rescue rover (scanning stand-in). Loop Slap: ultrasonic + actuator.

Adaptation: Card LiDAR / camera become one HC-SR04 taped to an SG90. It sweeps and shouts when something enters the arc.

PARTS

- SG90 micro servo
- HC-SR04 + ECHO divider (Proj 3)
- Active buzzer
- Tape / rubber band

WIRING

ESP32 PIN	CONNECTS TO
VIN (5V)	Servo red, HC-SR04 VCC
GND	Servo brown, HC-SR04 GND, buzzer –
GPIO 13	Servo orange (signal)
GPIO 5 / 18	TRIG / ECHO (divider)
GPIO 19	Active buzzer +

HOW IT WORKS

Install ESP32Servo by Kevin Harrington. Sense at each angle; Think if cm < 25; Act buzz. Let the servo settle (~60–120 ms) before each ping.

THE CODE

```
#include <ESP32Servo.h>
Servo radar;
const int TRIG = 5, ECHO = 18, BUZZ = 19;

float readCM() {
  digitalWrite(TRIG, LOW); delayMicroseconds(2);
  digitalWrite(TRIG, HIGH); delayMicroseconds(10);
  digitalWrite(TRIG, LOW);
  return pulseIn(ECHO, HIGH, 30000) * 0.0343 / 2.0;
}

void setup() {
  radar.attach(13);
  pinMode(TRIG, OUTPUT); pinMode(ECHO, INPUT); pinMode(BUZZ, OUTPUT);
  Serial.begin(115200);
}

void sweep(int from, int to, int step) {
  for (int a = from; a != to; a += step) {
    radar.write(a); delay(60);
    float cm = readCM();
    Serial.printf("angle %d: %.0f cm\n", a, cm);
    digitalWrite(BUZZ, (cm > 0 && cm < 25) ? HIGH : LOW);
  }
}

void loop() {
  sweep(10, 170, 2);
  sweep(170, 10, -2);
}
```

TRY THIS NEXT

On a hit, stop the sweep, point at the angle for 2 seconds, then resume — a “priority picker” Think card.

TROUBLESHOOTING

- **Servo jitters / board resets:** power servo from breadboard 5V PSU; share GND.
- **Buzzes at ends:** sweep 10–170°, not 0–180°.

6 · Hallway Motion Watch

PIR notices a person — Night watch with a real motion sense.

IF / THEN BREATH

If something moves, then sound the night watch alarm.

Cards: Bot-Builders: (motion stand-in for Camera) · If/Then · Speaker · LED · Mission Night watch. Loop Slap: HD Camera Lens (your PIR is the kit stand-in) + Audio Speaker.

Adaptation: No camera in the kit. HC-SR501 PIR is the honest “something moved” sensor for M-06.

PARTS

- HC-SR501 PIR
- Active buzzer
- Red LED + 220Ω

WIRING

ESP32 PIN	CONNECTS TO
VIN (5V) or 3V3	PIR VCC (check module marking)
GND	PIR GND, buzzer –, LED ground
GPIO 27	PIR OUT
GPIO 19	Active buzzer +
GPIO 23	Red LED → 220Ω → GND

HOW IT WORKS

Sense: OUT goes HIGH for a hold time after motion. Think: if HIGH, Act alarm. Warm up 30–60 s after power-on before trusting it.

THE CODE

```
const int PIR = 27, BUZZ = 19, LED = 23;

void setup() {
  pinMode(PIR, INPUT);
  pinMode(BUZZ, OUTPUT);
  pinMode(LED, OUTPUT);
  delay(60000); // warm-up (shorten to 10000 while testing)
}

void loop() {
  bool move = digitalRead(PIR) == HIGH;
  digitalWrite(LED, move ? HIGH : LOW);
  digitalWrite(BUZZ, move ? HIGH : LOW);
}
```

TRY THIS NEXT

Combine with Project 1: alarm only if dark AND motion — the full Night watch mission.

TROUBLESHOOTING

- **Always HIGH:** sensitivity pot too high, or warm-up skipped.
- **Never trips:** aim the dome at the room; check VCC; move across the field, not straight at it.

7 · Tip-Over Alarm

Tilt sensor knows if the bot (or breadboard) tips — Safety lock vibes.

IF / THEN BREATH

If the bot tips, then shout.

Cards: Bot-Builders: Tilt sensor · Safety lock · Speaker · LED. Loop Slap: Audio Speaker + LED Display Screen. No tilt-sense card in the 16 — the Tilt sensor is Bot-Builders only.

Adaptation: A ball tilt switch is a blunt bumper/tilt hybrid. For finer tilt, try the MPU-6050 spirit-level build in the ESP32 planner.

PARTS

- Tilt / ball switch
- LED + 220Ω
- Active buzzer

WIRING

ESP32 PIN	CONNECTS TO
GPIO 15	One leg of the tilt switch
GND	Other tilt leg, buzzer −, LED ground
GPIO 19	Active buzzer +
GPIO 23	LED → 220Ω → GND

HOW IT WORKS

Sense: switch closes when tipped. Think: INPUT_PULLUP reads LOW when closed. Act: buzz + LED. Debounce — the ball rattles near the tipping point.

THE CODE

```
const int TILT = 15, BUZZ = 19, LED = 23;

void setup() {
  pinMode(TILT, INPUT_PULLUP);
  pinMode(BUZZ, OUTPUT);
  pinMode(LED, OUTPUT);
}

void loop() {
  bool tipped = digitalRead(TILT) == LOW;
  digitalWrite(LED, tipped ? HIGH : LOW);
  digitalWrite(BUZZ, tipped ? HIGH : LOW);
  delay(30);
}
```

TRY THIS NEXT

Require tipped for 300 ms before alarming — Think: Signal filter.

TROUBLESHOOTING

- **Alarms upright:** legs shorted, or switch orientation inverted — flip the module.
- **Flaky:** add a short sustained-LOW check before sounding.

8 · Spill Watch

Water sensor trips before the mess spreads — Garden waterer without a pump.

IF / THEN BREATH

If water rises, then warn before it is too late.

Cards: Bot-Builders: (moisture stand-in) · If/Then · Speaker · Mission Garden waterer (M-03) adapted. Loop Slap: Audio Speaker. No water-sense card in the 16 — this build's Sense is Bot-Builders only.

Adaptation: No pump or soil probe in the Super Starter. The water-level sensor is a spill / flood Watcher: Sense wet → Act alarm. A relay + house plant pump would be a Phase-2 Act.

PARTS

- Water level sensor
- LED + 220Ω
- Active buzzer

WIRING

ESP32 PIN	CONNECTS TO
GPIO 25	Water sensor VCC (power only while sampling)
GND	Water sensor GND, buzzer −, LED ground
GPIO 34	Water sensor signal (S / A0)
GPIO 19	Active buzzer +
GPIO 23	LED → 220Ω → GND

HOW IT WORKS

Sense: wetter → higher (or lower) ADC. Think: threshold. Act: beep. Power the sensor from a GPIO only while reading so the traces corrode slower.

THE CODE

```
const int PWR = 25, SENSE = 34, BUZZ = 19, LED = 23;
const int WET = 1500; // tune in Serial

void setup() {
  pinMode(PWR, OUTPUT);
  pinMode(BUZZ, OUTPUT);
  pinMode(LED, OUTPUT);
  Serial.begin(115200);
}

void loop() {
  digitalWrite(PWR, HIGH);
  delay(50);
  int v = analogRead(SENSE);
  digitalWrite(PWR, LOW);
  Serial.println(v);
  bool wet = v > WET;
  digitalWrite(LED, wet ? HIGH : LOW);
  digitalWrite(BUZZ, wet ? HIGH : LOW);
  delay(400);
}
```

TRY THIS NEXT

Invert the story: “garden is OK while wet; alarm when dry” if you later add a soil probe.

TROUBLESHOOTING

- **Corrosion:** never leave powered underwater; rinse and dry after tests.
- **Threshold wrong:** print values dry vs damp finger and pick the midpoint.

9 · Hot-Day Fan Bot

Thermometer decides; fan Act cools — Lab partner / weather with muscle.

IF / THEN BREATH

If it is hot, then blow air.

Cards: Bot-Builders: Thermometer · If/Then · Fan · USB. Missions Weather station + Lab partner. Loop Slap: thermometer + DC motor.

Adaptation: Kit DC motor + fan blade via PN2222. Not HVAC — a desk breeze that proves Sense → Think → Act.

PARTS

- DHT11 + 10K pull-up
- 3V DC motor + fan
- PN2222 NPN
- 1K base resistor
- 1N4007 diode
- Optional LED

WIRING

ESP32 PIN	CONNECTS TO
3V3 / GND	DHT11 + / -; 10K data → VCC
GPIO 4	DHT11 data
VIN or 3V3	One motor lead (start with 3V3)
Other motor lead	Transistor collector
Transistor emitter	GND
GPIO 23	1K → transistor base
Diode	Across motor (stripe toward + supply)

HOW IT WORKS

Sense temp. Think: if $t \geq 27^{\circ}\text{C}$, Act: GPIO drives transistor → motor. Flyback diode required. Never drive the motor from a GPIO directly.

THE CODE

```
#include <DHT.h>
DHT dht(4, DHT11);
const int FAN = 23;
const float HOT = 27.0;

void setup() {
  pinMode(FAN, OUTPUT);
  dht.begin();
  Serial.begin(115200);
}

void loop() {
  float t = dht.readTemperature();
  if (isnan(t)) { delay(2000); return; }
  Serial.println(t);
  digitalWrite(FAN, t >= HOT ? HIGH : LOW);
  delay(2000);
}
```

TRY THIS NEXT

Show HOT/OK on the LCD from Project 4 — Display Act card.

TROUBLESHOOTING

- **Board resets when fan starts:** use breadboard 5V PSU for the motor; share GND with ESP32.
- **Fan never spins:** transistor pinout (E-B-C), diode orientation, or HOT threshold too high.

10 · Badge Door Helper

Tap the kit fob to “unlock” — Classroom helper as access, not wheels.

IF / THEN BREATH

If the badge is known, then unlock; else buzz no.

Cards: Bot-Builders: (tag sense) · Safety lock / If/Then · LED · (relay as the Act). Mission Classroom helper adapted. Loop Slap: “IF/THEN” Code Script + LED Display Screen. No badge/identity card in the 16 — the tag read is Bot-Builders only.

Adaptation: No drive wheels to carry a note. This bot decides who may pass — a doorway Classroom helper. RC522 is 3.3V only.

PARTS

- RC522 RFID + card/fob
- 5V relay
- Green LED + 220Ω
- Active buzzer

WIRING

ESP32 PIN	CONNECTS TO
3V3	RC522 3.3V (NOT 5V)
GND	RC522 GND, relay GND, buzzer –
GPIO 5 / 18 / 23 / 19 / 22	RC522 SDA / SCK / MOSI / MISO / RST
VIN (5V)	Relay VCC
GPIO 32	Relay IN
GPIO 25	Green LED → 220Ω → GND
GPIO 26	Active buzzer +

HOW IT WORKS

Library MFRC522 (Miguel Balboa). SDA on the module means SS/CS. Learn your UID from Serial, paste into allow-list. Never put wall power on the relay in student builds.

THE CODE

```
#include <SPI.h>
#include <MFRC522.h>
#define SS_PIN 5
#define RST_PIN 22
MFRC522 rfid(SS_PIN, RST_PIN);
byte allowed[] = {0xDE, 0xAD, 0xBE, 0xEF}; // replace via Serial
const int RELAY = 32, GREEN = 25, BUZZ = 26;

bool uidMatch() {
  if (rfid.uid.size != 4) return false;
  for (byte i = 0; i < 4; i++)
    if (rfid.uid.uidByte[i] != allowed[i]) return false;
  return true;
}

void setup() {
  Serial.begin(115200);
  SPI.begin();
  rfid.PCD_Init();
  pinMode(RELAY, OUTPUT);
  pinMode(GREEN, OUTPUT);
  pinMode(BUZZ, OUTPUT);
  Serial.println("Scan to learn UID...");
}

void loop() {
  if (!rfid.PICC_IsNewCardPresent() || !rfid.PICC_ReadCardSerial()) return;
  Serial.print("UID:");
  for (byte i = 0; i < rfid.uid.size; i++) Serial.printf(" %02X", rfid.uid.uidByte[i]);
  Serial.println();
  if (uidMatch()) {
    digitalWrite(GREEN, HIGH); digitalWrite(RELAY, HIGH);
    delay(1500);
    digitalWrite(GREEN, LOW); digitalWrite(RELAY, LOW);
  } else {
    digitalWrite(BUZZ, HIGH); delay(200); digitalWrite(BUZZ, LOW);
  }
  rfid.PICC_HaltA(); rfid.PCD_StopCrypto1();
}
```

TRY THIS NEXT

Keypad Safe Lock in the ESP32 planner = PIN instead of badge (same mission energy).

TROUBLESHOOTING

- **No UID:** power must be 3V3; SDA means SS; wiring SCK/MOSI/MISO.
- **Always denied:** paste YOUR Serial UID into allowed[].

11 · Story Beeper

Buttons play a three-note “story” — Storyteller mission, kit edition.

IF / THEN BREATH

If the teller presses a beat, then play the next note of the story.

Cards: Bot-Builders: If/Then · Timer · Speaker · USB · Mission Storyteller (M-04). Loop Slap: speaker / think.

Adaptation: No speech synth in the kit. A passive buzzer + buttons tells a tiny melody story. “Stay on long enough to finish” = don’t cut power mid-tune.

PARTS

- Passive buzzer
- 2–3 push buttons
- Jumpers

WIRING

ESP32 PIN	CONNECTS TO
GPIO 18	Passive buzzer + ; – → GND
GPIO 21 / 19 / 17	Buttons (other leg → GND), INPUT_PULLUP

HOW IT WORKS

Passive buzzer needs tone(). Each button is a story beat. Think is the sequence; Power is USB staying plugged in until the tune ends.

THE CODE

```
const int BUZZ = 18;
const int B1 = 21, B2 = 19, B3 = 17;

void setup() {
  pinMode(B1, INPUT_PULLUP);
  pinMode(B2, INPUT_PULLUP);
  pinMode(B3, INPUT_PULLUP);
}

void loop() {
  if (digitalRead(B1) == LOW) { tone(BUZZ, 262, 200); delay(220); } // C
  if (digitalRead(B2) == LOW) { tone(BUZZ, 330, 200); delay(220); } // E
  if (digitalRead(B3) == LOW) { tone(BUZZ, 392, 400); delay(420); } // G
}
```

TRY THIS NEXT

Press B1 → B2 → B3 in order to unlock a longer victory melody — Memory card Think.

TROUBLESHOOTING

- **No sound:** you grabbed the active buzzer — use the passive one (often shows a circuit board).
- **Wrong notes:** fine — story > pitch. Try tone values 262 / 294 / 330 / 392.

12 · Rescue Beacon

The showcase loop: notice a person, wave, shout — Rescue rover without wheels.

IF / THEN BREATH

If someone needs help in front of me, then wave and call a grown-up.

Cards: Bot-Builders: Ultrasonic or PIR · If/Then · Servo arm · Speaker · LED · Mission Rescue rover (M-01). Loop Slap: full SENSE → THINK → ACT breath.

Adaptation: Card rover drives; this beacon stays put, finds someone in front, waves a flag (servo), and calls a grown-up (buzzer). Same loop, honest kit.

PARTS

- HC-SR04 + divider OR PIR
- SG90 servo
- Active buzzer
- LED + 220Ω
- Paper flag taped to horn

WIRING

ESP32 PIN	CONNECTS TO
VIN / GND	Sensor + servo power as in Proj 3/5/6
GPIO 5 / 18	Ultrasonic TRIG / ECHO (if using sonar)
GPIO 27	PIR OUT (if using PIR instead)
GPIO 13	Servo signal
GPIO 19	Buzzer +
GPIO 23	LED → 220Ω → GND

HOW IT WORKS

Pick one Sense (sonar OR PIR). Think: person present? Act: LED on, buzz pattern, servo wave 0 ↔ 90. Recite: “If it senses a person, then it waves and shouts — it thinks with If/Then.”

THE CODE

```
#include <ESP32Servo.h>
Servo arm;
const int TRIG = 5, ECHO = 18, BUZZ = 19, LED = 23;
// set USE_PIR 1 and wire PIR to GPIO 27 to swap senses
#define USE_PIR 0
const int PIR = 27;

float readCM() {
  digitalWrite(TRIG, LOW); delayMicroseconds(2);
  digitalWrite(TRIG, HIGH); delayMicroseconds(10);
  digitalWrite(TRIG, LOW);
  return pulseIn(ECHO, HIGH, 30000) * 0.0343 / 2.0;
}

bool person() {
  #if USE_PIR
    return digitalRead(PIR) == HIGH;
  #else
    float cm = readCM();
    return cm > 0 && cm < 50;
  #endif
}

void setup() {
  arm.attach(13);
  pinMode(TRIG, OUTPUT); pinMode(ECHO, INPUT);
  pinMode(PIR, INPUT);
  pinMode(BUZZ, OUTPUT); pinMode(LED, OUTPUT);
  arm.write(0);
}

void loop() {
  if (!person()) {
    digitalWrite(LED, LOW); digitalWrite(BUZZ, LOW);
    arm.write(0); delay(100); return;
  }
  digitalWrite(LED, HIGH);
  for (int i = 0; i < 3; i++) {
    digitalWrite(BUZZ, HIGH); arm.write(80); delay(150);
    digitalWrite(BUZZ, LOW); arm.write(20); delay(150);
  }
  delay(800);
}
```

TRY THIS NEXT

Add Night Watch Light so it only rescues in the dark — two missions, one bench.

TROUBLESHOOTING

- **False rescues:** raise distance threshold or switch to PIR.
- **Servo brownout:** breadboard 5V PSU for the servo; shared GND.

Mission mashups

Two projects, one Bot-Builders mission — still one breadboard if you plan pins.

MISSION	COMBINE	SAY ALOUD
Night watch	1 + 6	If dark AND motion, then alarm.
Weather station	4 + 9	If hot, then show HOT and run the fan.
Rescue rover	5 + 12	If the sweep finds someone, then wave and shout.
Lab partner	4 + 7	If the bench tips or overheats, then stop and warn.
Garden waterer	8 + LED from 1	If wet, then green; if dry/spill, then shout.

CHAOS, FOR BUILDERS

Loop Slap Chaos tags make great debug prompts: **WEIRD** — what unexpected reading did Serial show? **EDGE** — what happens at the threshold (39 vs 41 cm)? **TRUST** — would you let this bot wake the house overnight?

After the build: take a photo of the breadboard next to the four cards you played. That scrapbook is your clubhouse portfolio.

The cards invent. The kit proves the loop.

Every project in this book is a living If/Then. When the cards get wilder than the box, adapt on purpose — then go deeper in the **ESP32 Project Book** and the online planner.

[hibot.space/games](#) · print the decks

[hibot.space/activities](#) · more kit builds (Activities Pass)

› **Hi, Bot** · Maker Library

› **Hi, Bot**