IOT system

Components

Devices/Sensors → collect data
Connectivity → WiFi, Bluetooth, etc.
Data Processing → edge/cloud
Application → user interface

Architecture Layers

Perception → sensing (sensors, actuators)	Sensors; Actuators; RFID tags and readers; Embedded devices; Microcontrollers
Network → data transmission	Routers; Gateways; Communication modules; Internet infrastructure
Communication Technologies	Wi-Fi; Bluetooth; ZigBee
Protocols Used	MQTT; CoAP; HTTP; AMQP
Processing → storage, analytics	Cloud, edge computing, databases

Application → user services

Sensors vs Actuators

Sensors → input (temperature, light)
Actuators → output (motor, relay)

IoT protocols

reliable and efficient communication

data transfer, interoperability, low power usage, low latency and security

Protocol Stack

Link Layer: Physical communication (Wi-Fi, ZigBee)

Network Layer: Addressing and routing (IPv6, 6LoWPAN)

Transport Layer: Reliable or fast data transfer (TCP/UDP)

IoT protocols (cont)

Application Layer: IoT protocols (MQTT, CoAP, HTTP, AMQP)

Arduino

function - interface sensors + actuators; execute control logic; communicate with cloud + devices (data)

microcontroller based; uses Wifi/ bluetooth

setup() - initialize settings; executes once

loop() - continuously; repeatedly

GPIO - interact with external devices; input from sensors; output to actuators

Arduino IDE - write, compile, and upload programs (sketches); syntax checking, library support, and serial monitor

types of Arduino - 1. Arduino Uno (basic), 2. Mega 2560 (more I/O pins + memory), 3. Nano (compact), 4. MKR WiFi 1010 (WiFi) 5. Uno WiFi Rev2 (WiFi)

Arduino shields - add-on boards; extend functionalityWi-Fi, Bluetooth, motor control, and sensor interfacing (without complex wiring)

interface sensor with arduino	VCC → 5V/3.3V GND → GND Output → Analog/Digital pin (e.g., A0 or D2)

Connect sensor to Arduino → Configure pin → Read sensor data → Process and use data

Data acquisition

Sensor → Arduino reads data → Processes values → Displays results

Data acquisition (cont)

Components Required

1. Arduino Uno / Nano
2. Temperature Sensor (LM35 / DHT11) or LDR
3. 16×2 LCD Display (or Serial Monitor)
4. Resistors and connecting wires
5. Breadboard
6. Power supply

```
int sensorPin = A0;
float temperature;
void setup() {
Serial.begin(9600);
}
void loop() {
int sensorValue = analogRead(sensorPin);
temperature = sensorValue * 0.488;
Serial.print("Temperature: ");
Serial.print(temperature);
delay(1000);
}
```

Arduino Toolchain

Toolchain = set of tools to write, compile, upload, debug programs

Components

Arduino IDE → write code (sketch)
Editor → C/C++ code writing
Compiler → converts code → HEX file
Uploader → uploads to Arduino
Hardware → executes program

Program Development Flow

Write code (setup(), loop()) → Compile (error checking) → Upload to board → Execute on hardware

Debugging Methods

Serial Monitor → print values
Serial Plotter → graph output
LED indicators → check flow

Importance

Simplifies embedded programming
Detects errors automatically
Enables quick upload
Supports libraries
Helps debugging

Arduino I/O Concepts

Digital I/O

Operates in two states: HIGH (1), LOW (0)
Digital Input → binary signals from sensors
Digital Output → controls actuators (LED, relay, buzzer)
Functions: digitalRead(), digitalWrite()

Analog I/O

Handles continuous signals (0–5V)
Analog Input → uses ADC to convert signal (0–1023)
Function: analogRead()

Analog Output (PWM)

Simulates analog signal using PWM
Range: 0–255
Function: analogWrite()

Interfacing in IoT

Sensors → input via digital/analog pins
Actuators → controlled via output pins
Flow: Sensor → Arduino → Processing → Actuator

Challenges

Challenges in Arduino-based IoT Systems

Voltage mismatch between devices
Limited GPIO pins
Signal noise and inaccurate readings
Power supply constraints
Timing and synchronization issues
Limited memory and processing capability
Communication failures (WiFi/Bluetooth)
Scalability issues for large systems
Debugging difficulties
Environmental effects on sensors

Effectiveness of Arduino

Comparison

Feature	Arduino	Raspberry Pi	ESP32 / ESP8266
Ease of Use	Very Easy	Moderate	Moderate
Cost	Low	High	Very Low
Processing	Low	High	Medium
Connectivity	External	Built-in	Built-in
Power Consumption	Low	High	Medium
Real-Time	Excellent	Poor	Good
Scalability	Limited	High	Moderate
Memory Capacity	Limited	High	Moderate
Suitability	Small-scale IoT, control systems	Data-intensive IoT applications	Wireless IoT nodes

PART C

Arduino IoT Temp & Humidity System

System Components

Arduino Uno
DHT11/DHT22 sensor
ESP8266 WiFi module
Power supply
Cloud platform (ThingSpeak/AWS)

Block Flow

Sensor → Arduino → WiFi → Cloud → User

Program{{nl