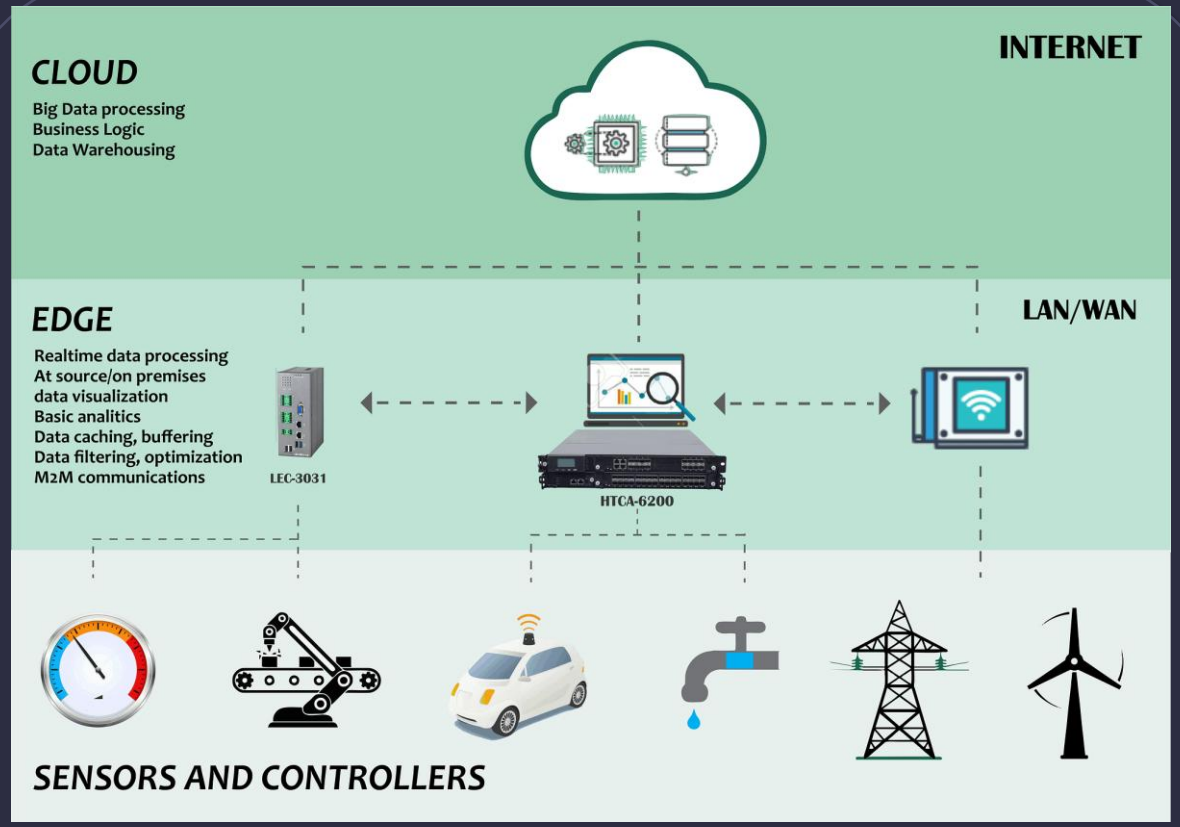
Internet of Things

Week 1: Introduction

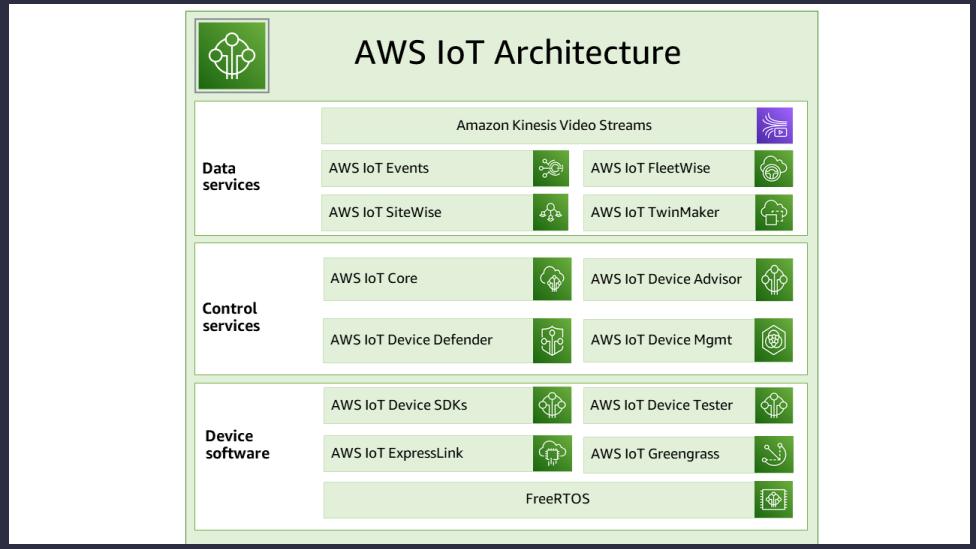
Fenerbahce University

How IoT Systems Work (End-to-End)

A practical mental model: device → network → edge/gateway → cloud → dashboard → actions



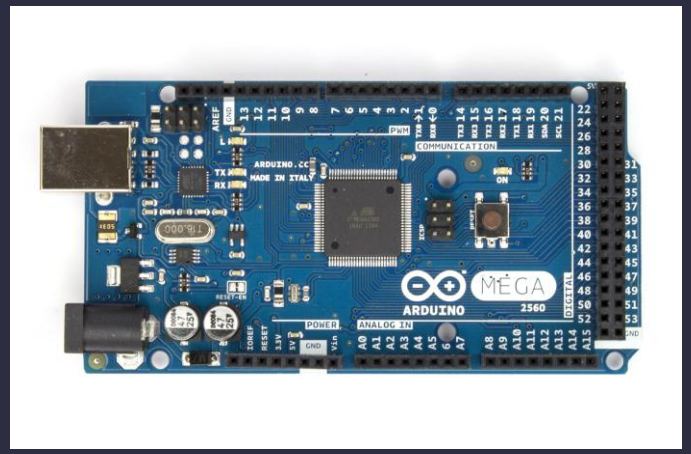
Edge–Fog–Cloud reference diagram



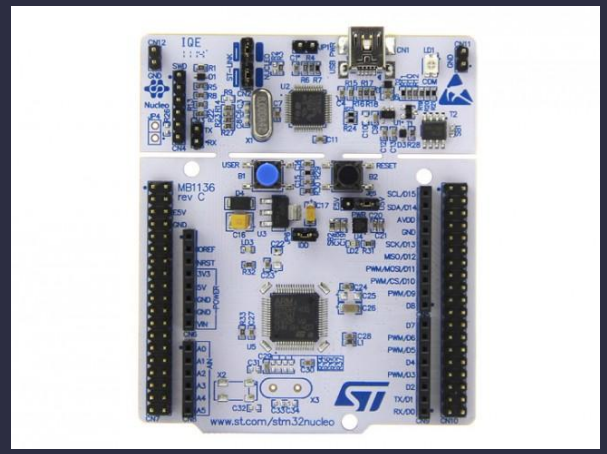
Example: AWS IoT “how it works” overview

Reference Hardware & Toolchain

We reuse embedded-systems foundations and connect them to cloud services.



Arduino Mega (MCU)



STM32 Nucleo (MCU)

Raspberry Pi (MPU)

- C/C++ toolchain + debugging
- Peripherals & drivers
- Networking (MQTT/HTTP)
- Backend API + database
- Dashboard & monitoring

Workflow: From Sensor to Dashboard

This course is project-driven. Each week adds one piece to the pipeline.

Build Loop

- Read sensor data reliably
- Timestamp + package data
- Send via wired/wireless link
- Ingest into backend (API/broker)
- Store + visualize + alert
- Close the loop (actuation)

Dashboards turn telemetry into decisions

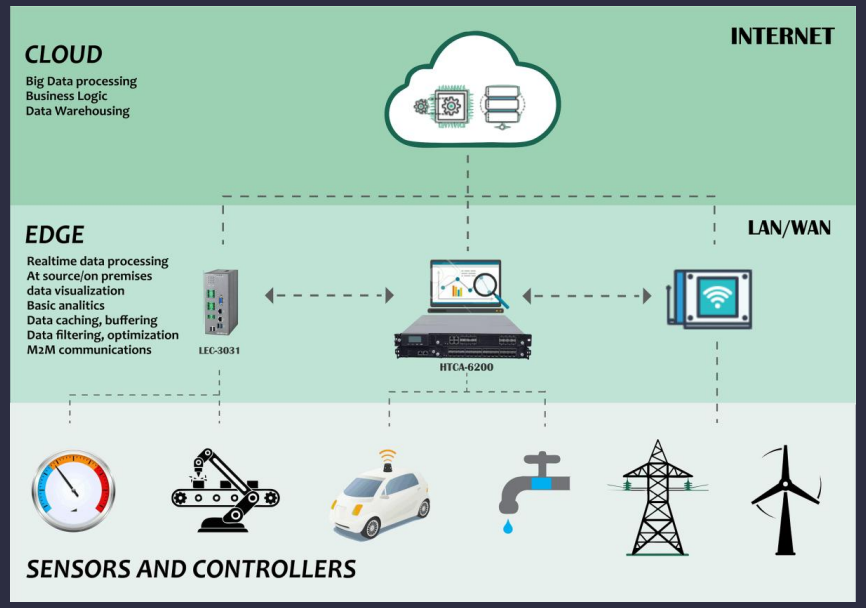
WEEK 1

Introduction to IoT Systems & Cyber-Physical Architecture

Week 1 — Key Concepts

Key concepts

- IoT = sensing + connectivity + computation + action
- Cyber-physical systems: tight loop between software and the physical world
- Key constraints: power, cost, reliability, security
- Where does processing happen? device vs edge vs cloud

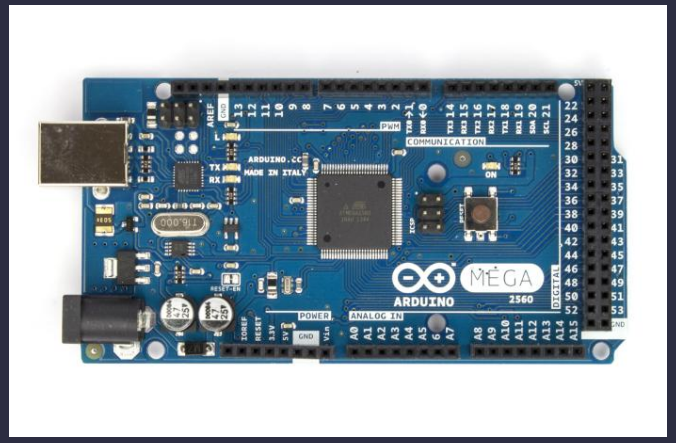


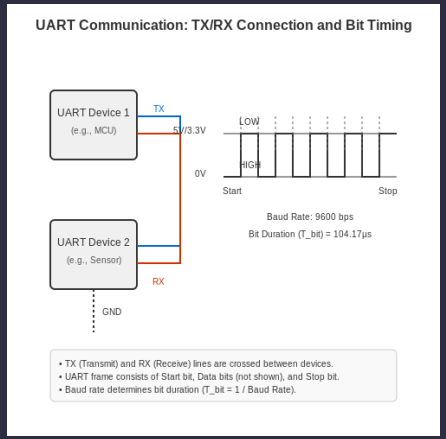
Edge/Cloud split (latency vs capability)

Week 1 — Hands-on Lab

Hands-on lab

- Set up development environment
- Blink + GPIO basics
- Read a simple sensor (digital/analog)
- Stream data over serial (CSV/JSON)



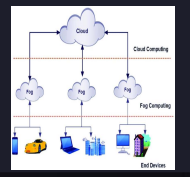


Week 1 — Case Study & Discussion

Discussion prompts

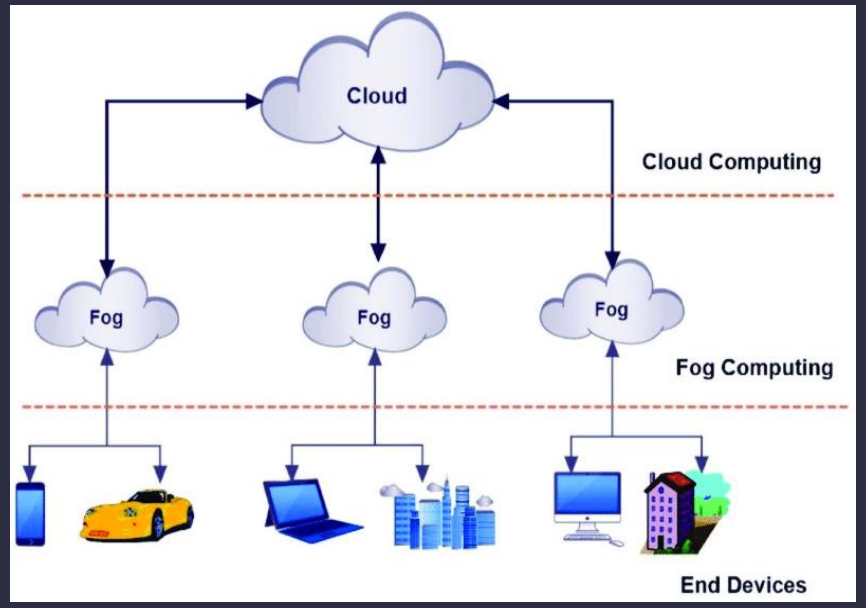
- Pick a use case: smart home / agriculture / industrial monitoring
- Define signals, sampling rate, and failure modes
- Sketch a first data pipeline (device → app)

Week 1 — Architecture: Edge → Fog → Cloud



Deep dive

- End devices: sensors + actuators + local control
- Fog/edge gateways: filter, buffer, and react locally
- Cloud: storage, analytics, dashboards, ML pipelines
- Design trade-offs: latency vs bandwidth vs power



A common Edge–Fog–Cloud IoT deployment pattern

Week 1 — Data Example: Telemetry Payload (JSON)

Deep dive

- Use stable device IDs and timestamps (UTC)
- Include units + calibration / schema version
- Prefer compact payloads on constrained networks
- Validate early to avoid “garbage in” at the cloud

JSON

```json
{
    "endereco": {
        "cep": "31270901",
        "city": "Belo Horizonte",
        "neighborhood": "Pampulha",
        "service": "correios",
        "state": "MG",
        "street": "Av. Presidente Antônio Carlos, 6627"
    }
}
```

Example JSON payload structure

Week 1 — Checklist: Lab Setup & Baseline Demo

Deliverables

- Toolchain + serial monitor installed and working
- Board flashes reliably; UART logs captured
- Baseline sensor demo with sampling rate noted
- Repo created with wiring photo + README

Example development board used for labs

WEEK 2

Microcontroller Architectures for IoT (MCU vs MPU)

Week 2 — Key Concepts

Key concepts

- MCU: deterministic, low-power, integrated peripherals
- MPU: Linux-capable, higher compute, more memory
- Choosing parts: interfaces, RAM/Flash, cost, power budget
- Boot process & memory map basics

MPU example: Raspberry Pi

Week 2 — Hands-on Lab

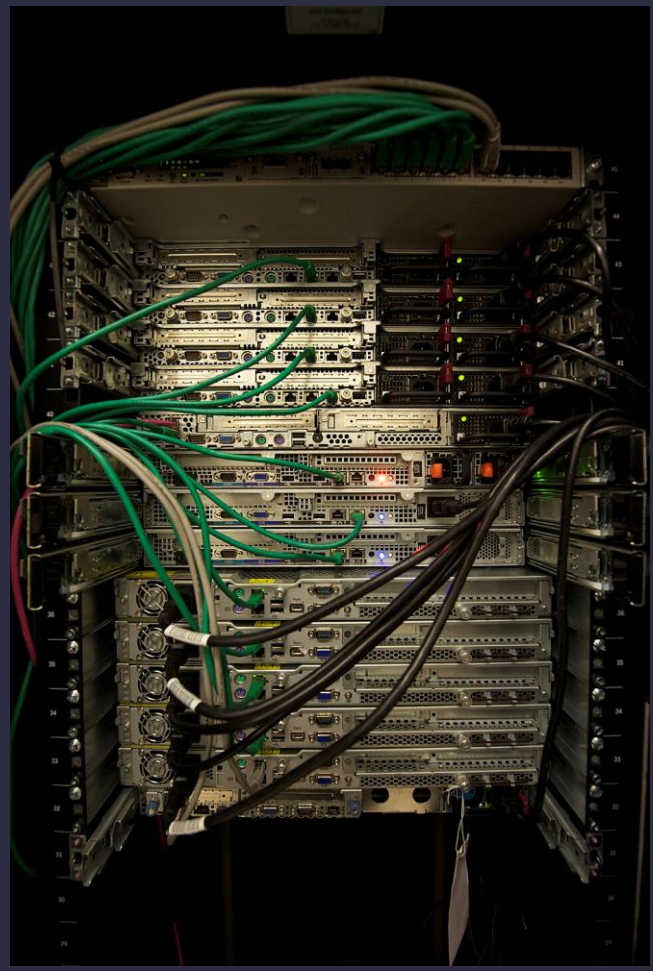
Hands-on lab

- Compile & flash firmware
- Use a debugger (breakpoints, watch, memory)
- Measure timing (SysTick / cycle counter)

Week 2 — Case Study & Discussion

Discussion prompts

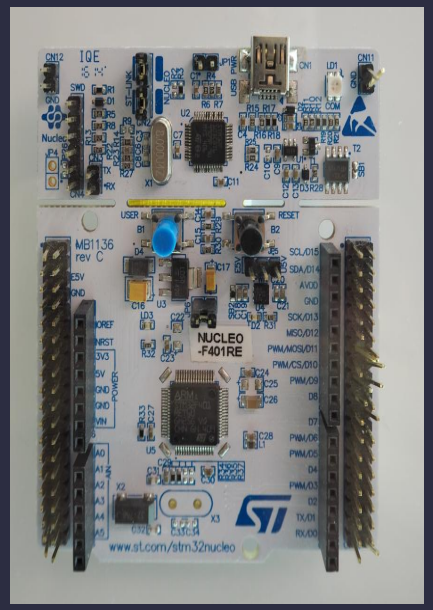
- When MCU is enough (battery nodes)
- When MPU is needed (gateway, heavy crypto, local ML)
- Hybrid design: MCU sensors + Linux gateway



Week 2 — Architecture: MCU vs MPU (SBC)

Deep dive

- MCU: deterministic timing, low power, small memory
- MPU/SBC: Linux + networking stack, higher compute
- Decision factors: power budget, latency, cost, OTA strategy
- Typical pattern: MCU nodes + SBC gateway + cloud

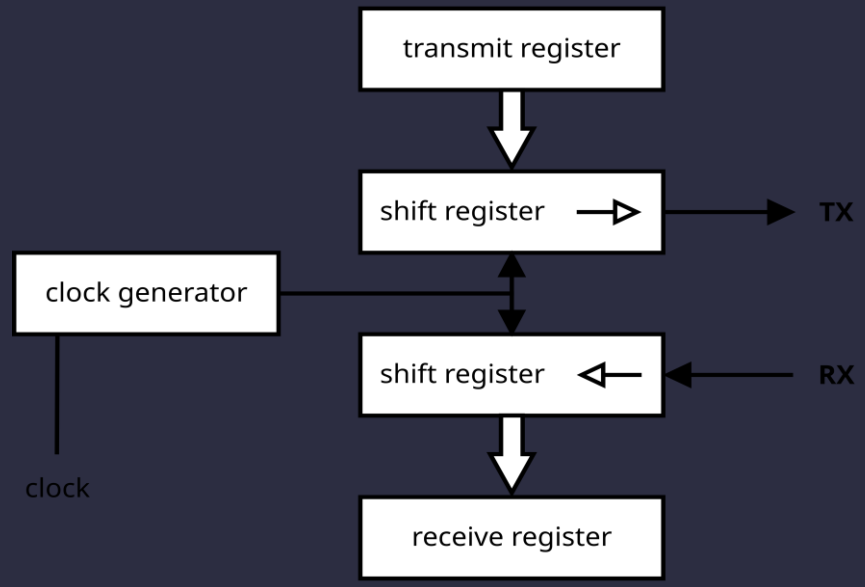


Left: MCU board • Right: Linux SBC (example)

Week 2 — Data Example: Peripheral Abstraction (UART)

Deep dive

- Peripherals expose registers + interrupts + DMA
- Clocking + prescalers define baud rate and timing
- Use circular buffers for streaming data
- Always add timeouts and error counters

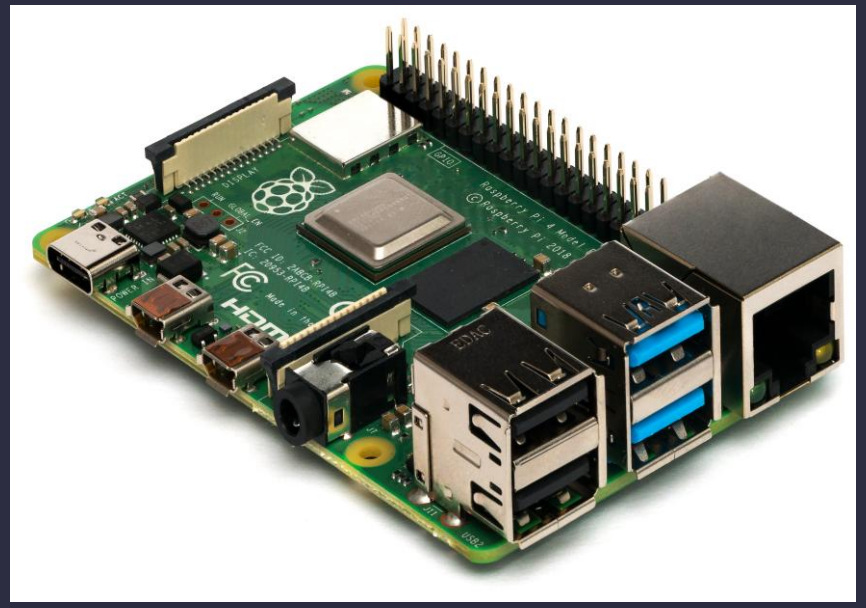


UART blocks: shift registers + clock generator

Week 2 — Checklist: Platform Selection

Deliverables

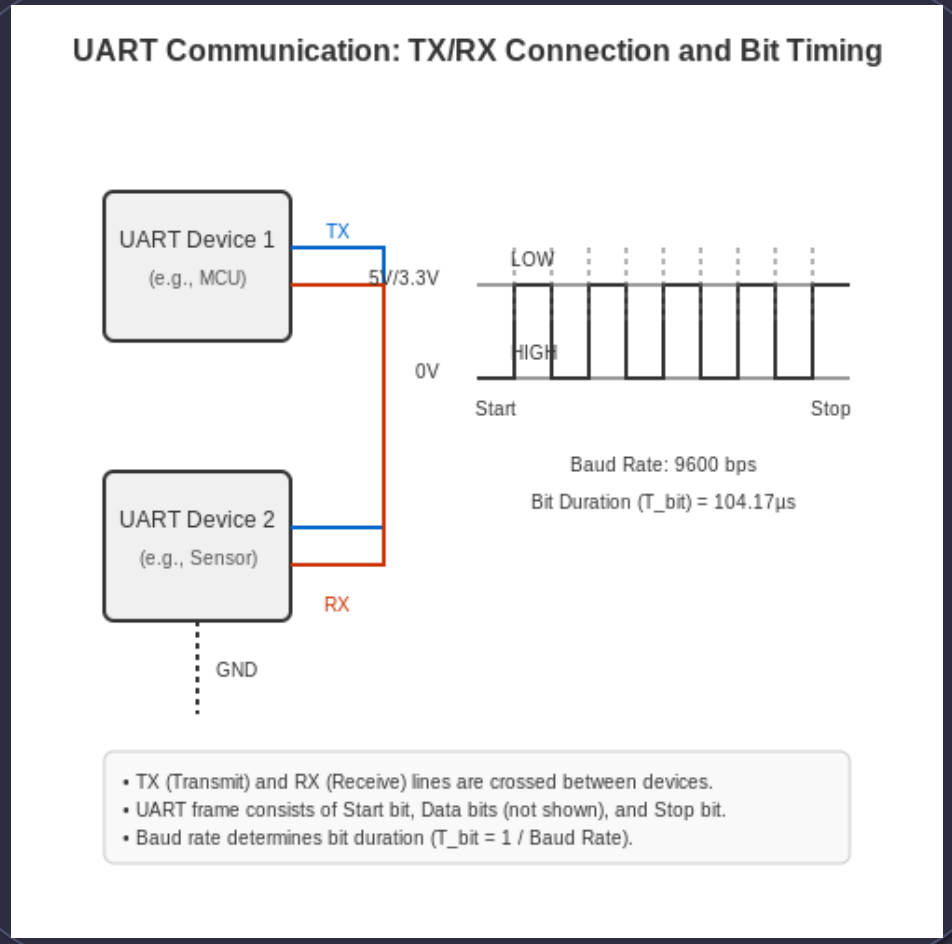
- Confirm voltage levels (3.3V vs 5V) + IO protection
- Document pinout, boot mode, and recovery method
- Decide how updates will work (bootloader/OTA)
- Define logging plan: UART + timestamps



Single-board computers can act as IoT gateways

WEEK 3

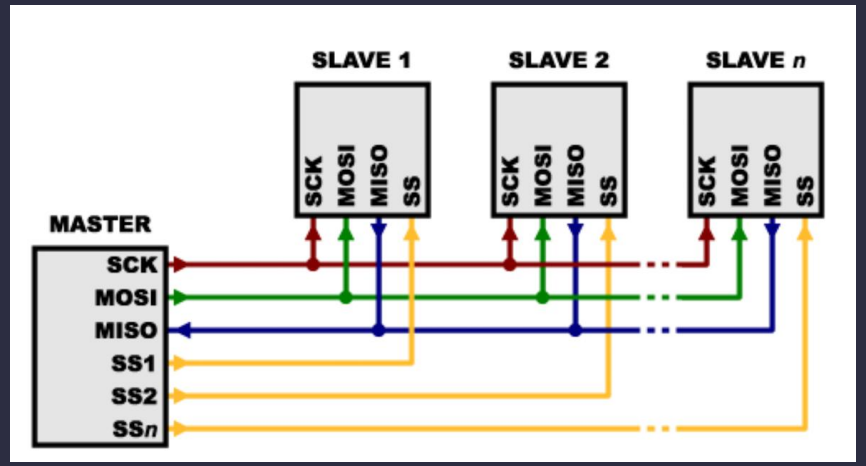
Embedded Interfaces for IoT Communication



Week 3 — Key Concepts

Key concepts

- UART: point-to-point serial, debug + modules
- I²C: two-wire shared bus (pull-ups!)
- SPI: high-speed synchronous bus for sensors/flash
- Data framing & error handling (timeouts, CRC)

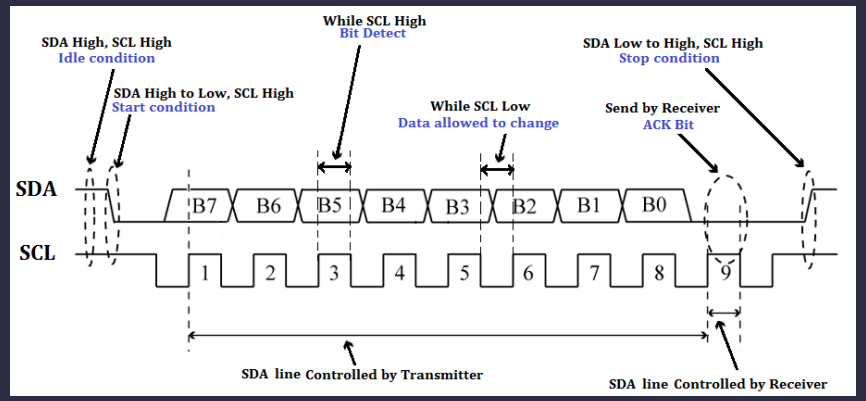


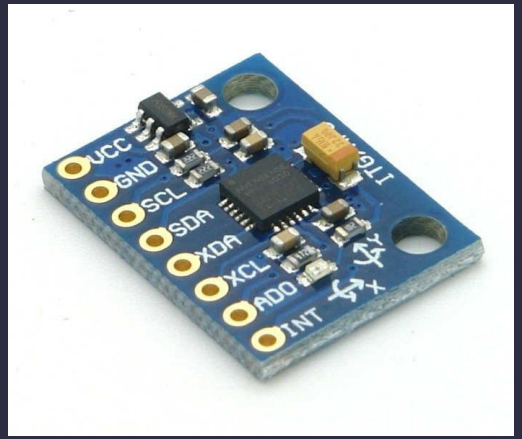
SPI bus wiring overview

Week 3 — Hands-on Lab

Hands-on lab

- Talk to a sensor over I²C
- Read registers and convert to physical units
- Design a simple driver: init → read → validate



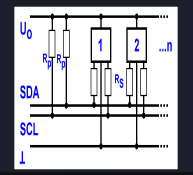


Week 3 — Case Study & Discussion

Discussion prompts

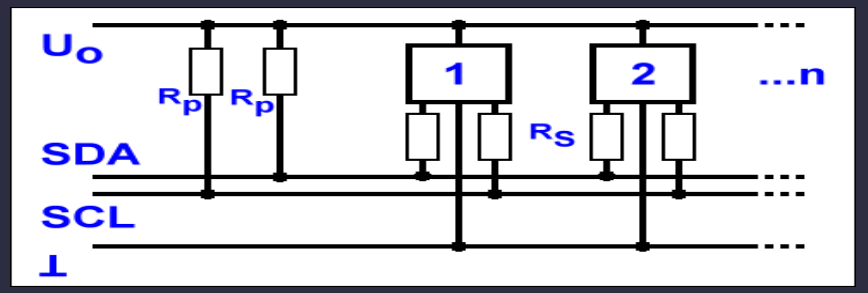
- Choosing buses in a multi-sensor node
- Address conflicts, bus length, EMI
- Using UART modules (GPS, modem)

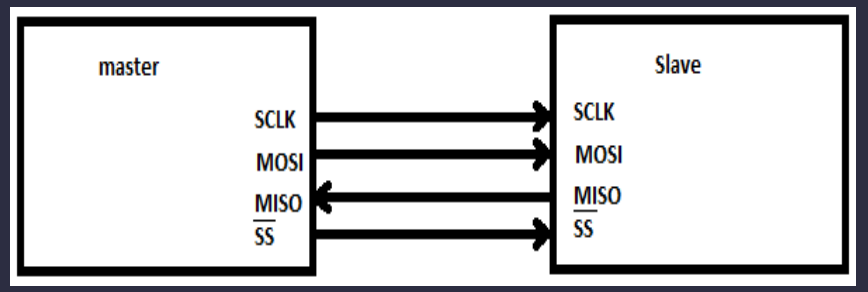
Week 3 — Architecture: Common Embedded Buses



Deep dive

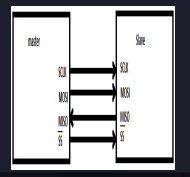
- I2C: multi-drop, address-based, requires pull-ups
- SPI: high speed, separate chip-select per device
- UART: async point-to-point, common for debugging
- Check voltage levels, wiring length, and grounding





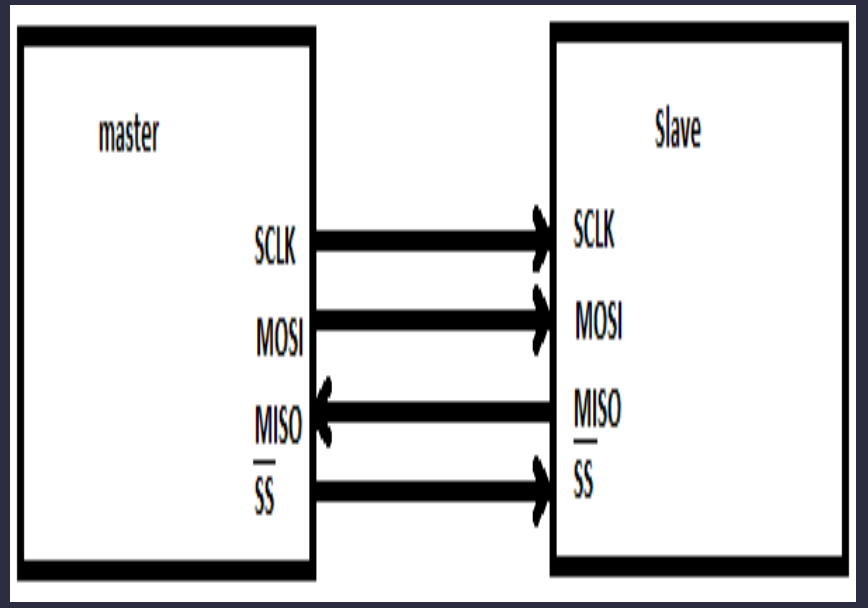
I2C (top) and SPI (bottom) bus examples

Week 3 — Data Example: Framing, CRC, and Retries



Deep dive

- Define a frame: header + payload + checksum (CRC)
- Handle partial reads and resynchronization
- Retries need idempotency or sequence numbers
- Measure throughput with realistic payload sizes

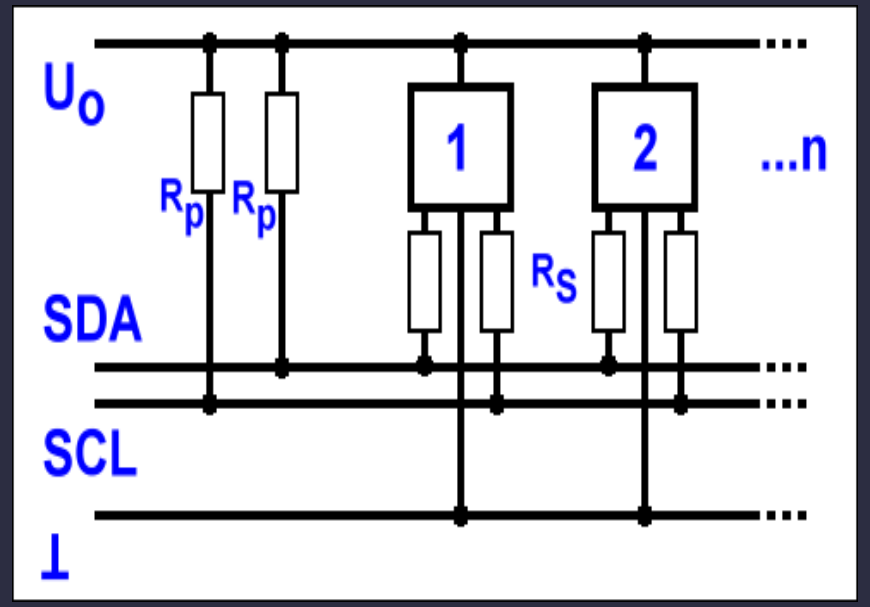


Bus framing is your first “protocol” layer

Week 3 — Checklist: Interface Bring-Up

Deliverables

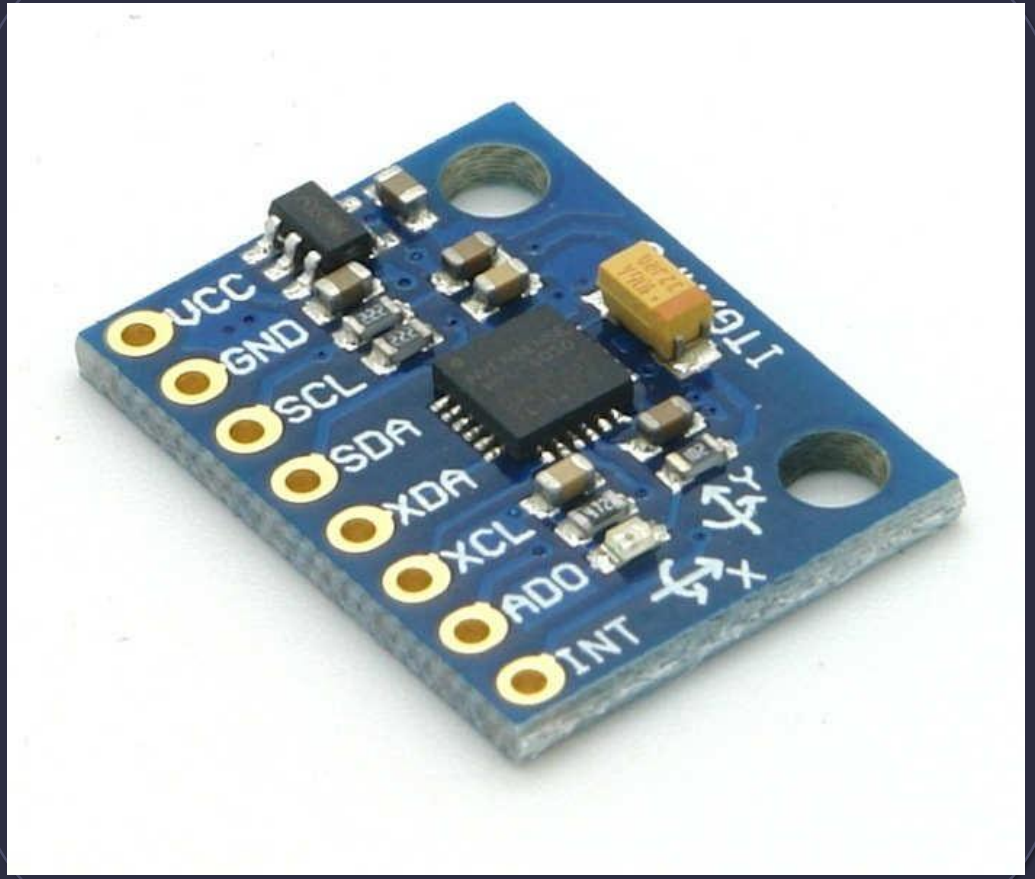
- Validate wiring + pull-ups (I2C) / chip-selects (SPI)
- Confirm bus speed and signal integrity
- Run an address scan (I2C) and sanity read
- Add timeouts, retries, and error logging



A quick I2C scan often saves hours

WEEK 4

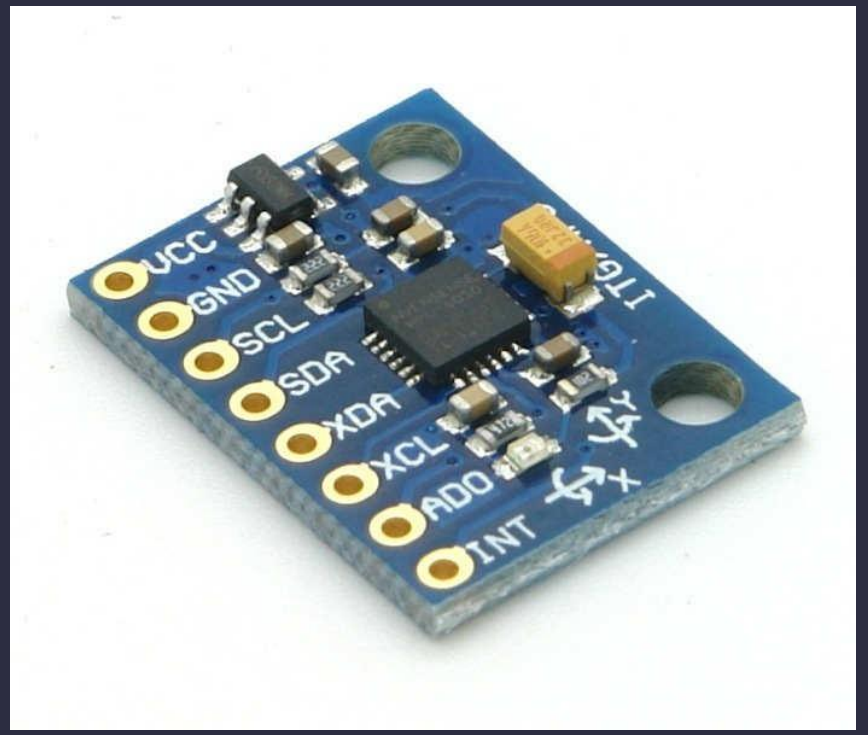
Sensor Integration & Data Acquisition (Baremetal I)



Week 4 — Key Concepts

Key concepts

- Sampling theory: rate, jitter, aliasing
- ADC basics + sensor front-end
- Calibration: offset, scale, temperature drift
- Filtering: moving average, low-pass

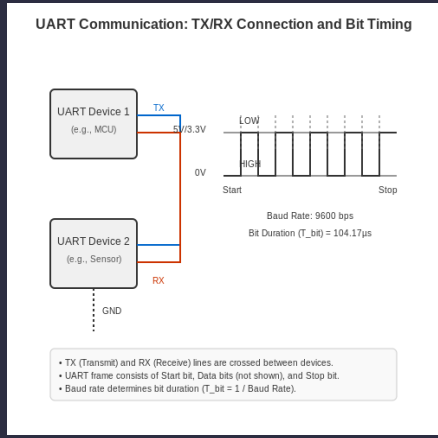


Example IMU module (MPU6050)

Week 4 — Hands-on Lab

Hands-on lab

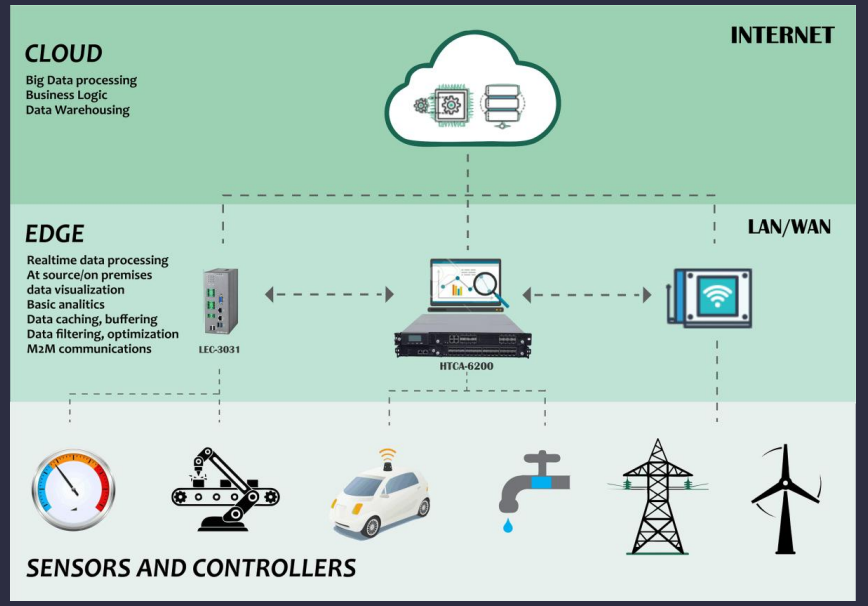
- Periodic sampling with timers
- Buffering + timestamping
- Send sensor frames over UART



Week 4 — Case Study & Discussion

Discussion prompts

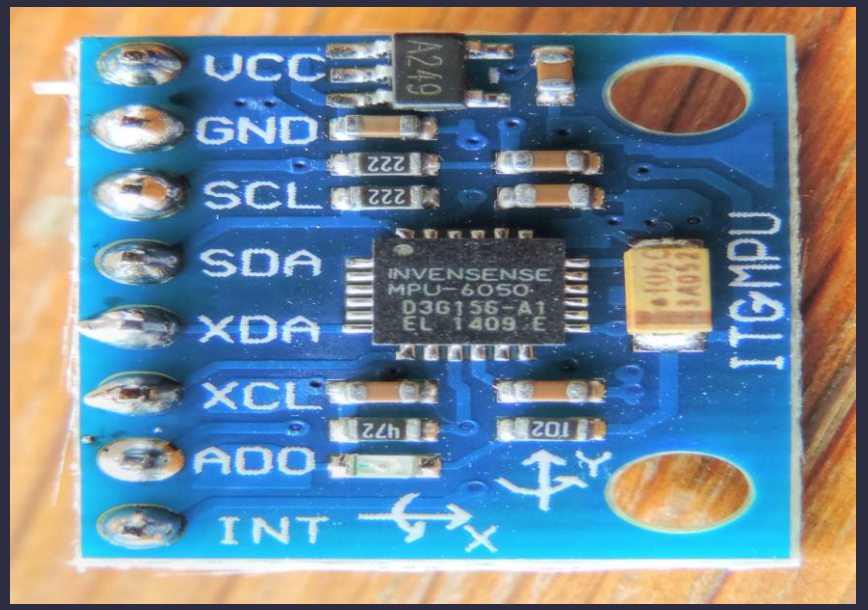
- Data quality checklist
- Outliers & missing data
- Designing a robust packet format



Week 4 — Architecture: Sensor → Data Acquisition Pipeline

Deep dive

- Signal chain: sensor → ADC/I2C/SPI → calibration
- Sampling rate is a design choice (power vs fidelity)
- Filtering: moving average / low-pass / median (as needed)
- Store metadata: units, scale, and sensor version



Example sensor breakout (MPU-6050 module)

Week 4 — Data Example: Timestamped Sensor Record

Deep dive

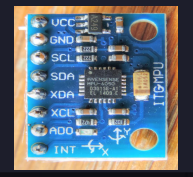
- Include: ts, deviceId, sensor values, battery, RSSI (if any)
- Keep payload stable; evolve via schema versioning
- Use integer encoding if bandwidth is tight
- Validate ranges (min/max) on-device when possible

JSON


```json
{
    "endereco": {
        "cep": "31270901",
        "city": "Belo Horizonte",
        "neighborhood": "Pampulha",
        "service": "correios",
        "state": "MG",
        "street": "Av. Presidente Antônio Carlos, 6627"
    }
}
```

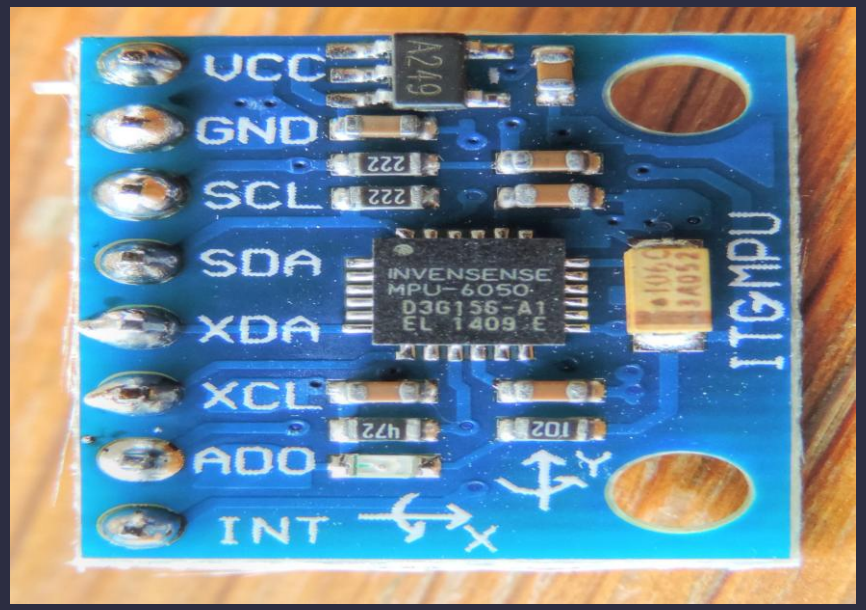

A consistent schema makes ingestion simpler

Week 4 — Checklist: Reliable Sampling



Deliverables

- Confirm sensor initialization and ID register
- Calibrate offsets (static) and verify noise level
- Add watchdog/reset strategy for sensor lockups
- Log raw and filtered values for comparison



Start by validating raw readings

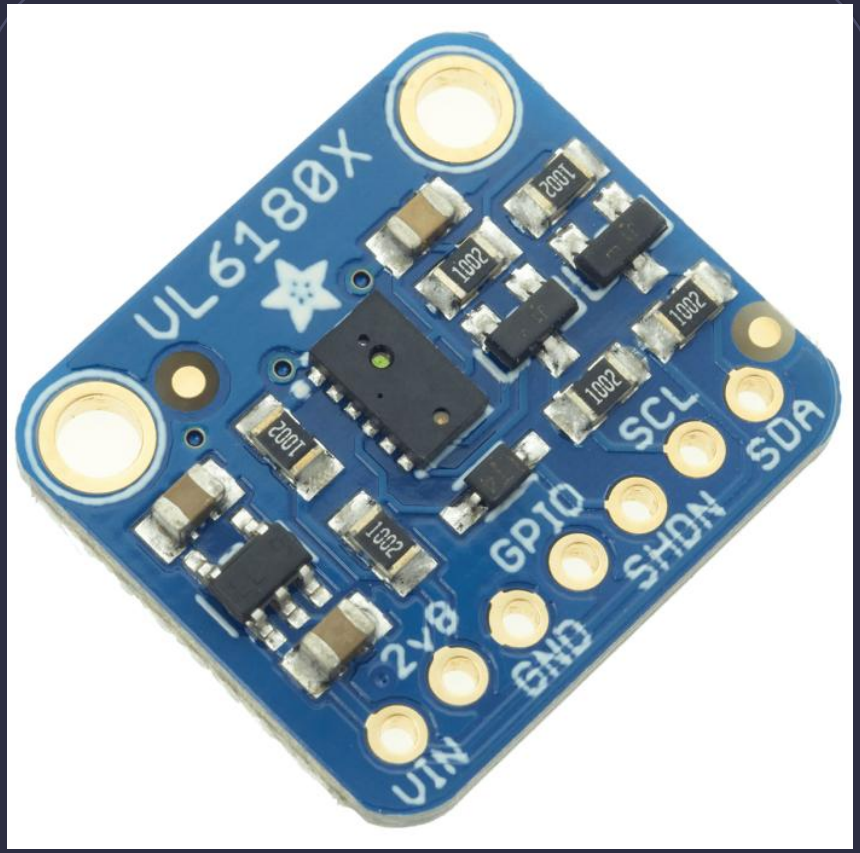
WEEK 5

IoT Node Design: Multi-Sensor System (Baremetal II)

Week 5 — Key Concepts

Key concepts

- Architecture: drivers, scheduler, communication, logging
- Data fusion: combining sensors to improve reliability
- Power management: sleep, duty cycle, batching
- Diagnostics: heartbeats, watchdogs, self-test

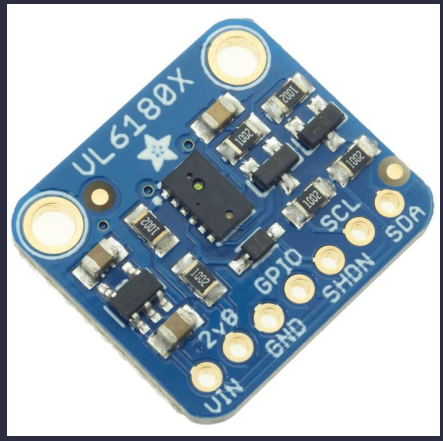


ToF sensor example (VL6180X)

Week 5 — Hands-on Lab

Hands-on lab

- Build a multi-sensor data packet
- Add a simple state machine
- Log to SD / flash (optional)
- Measure CPU load and memory

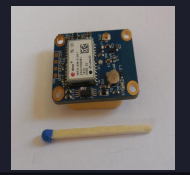


Week 5 — Case Study & Discussion

Discussion prompts

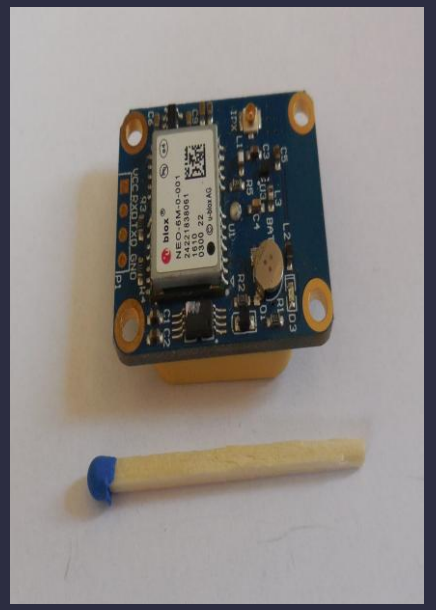
- Common failure modes: bus lockups, brownouts, memory leaks
- Design for recovery: reset strategies, safe defaults

Week 5 — Architecture: Multi-Sensor IoT Node



Deep dive

- Combine sensors: IMU + GPS + environment (example)
- Synchronize timestamps and handle different sample rates
- Power budget: duty-cycle sensors and radios
- Plan for missing/invalid readings (fault tolerance)



Example modules used in multi-sensor prototypes

Week 5 — Data Example: Sensor Fusion Packet

Deep dive

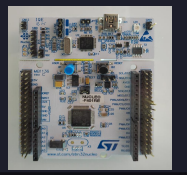
- Pack related readings under a single timestamp
- Include quality flags (e.g., GPS fix, IMU status)
- Use sequence numbers for loss detection
- Keep derived values (speed, tilt) separate from raw values

JSON


```
{
    "endereco": {
        "cep": "31270901",
        "city": "Belo Horizonte",
        "neighborhood": "Pampulha",
        "service": "correios",
        "state": "MG",
        "street": "Av. Presidente Antônio Carlos, 6627"
    }
}
```


A fusion packet can include both raw + derived values

Week 5 — Checklist: Node Design Review



Deliverables

- Document wiring harness + connector plan
- Validate current draw (active vs sleep)
- Define enclosure constraints (heat, moisture, mounting)
- Prepare a test plan (static + moving + long-run)

Prototype early; iterate the hardware

WEEK 6

Real-Time IoT Systems with FreeRTOS

Week 6 — Key Concepts

Key concepts

- Why RTOS: predictable timing with multiple activities
- Tasks, priorities, and scheduling
- Queues, semaphores, and event groups
- Avoiding race conditions + priority inversion

FreeRTOS (RTOS kernel)

Week 6 — Hands-on Lab

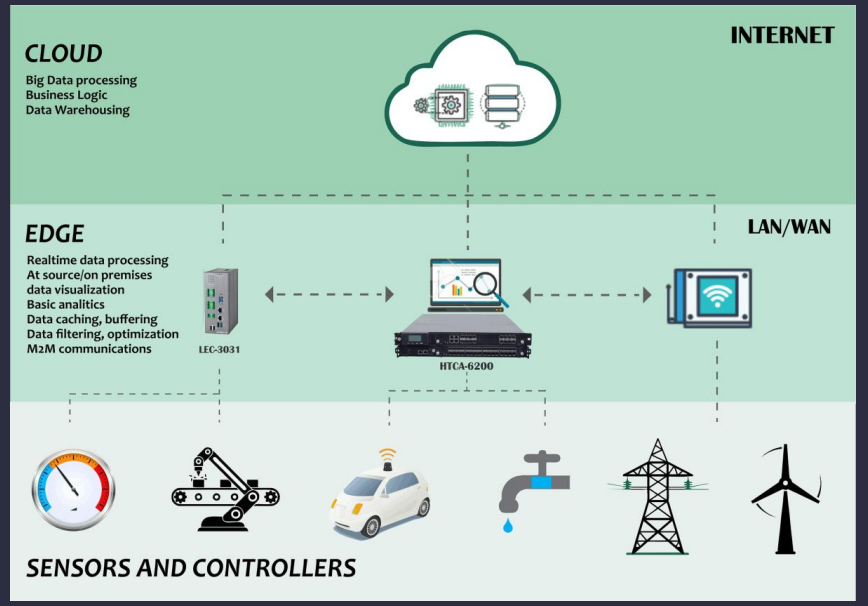
Hands-on lab

- Create tasks: Sensor / Comms / Logger
- Use queues to pass data safely
- Measure stack usage and tune priorities

Week 6 — Case Study & Discussion

Discussion prompts

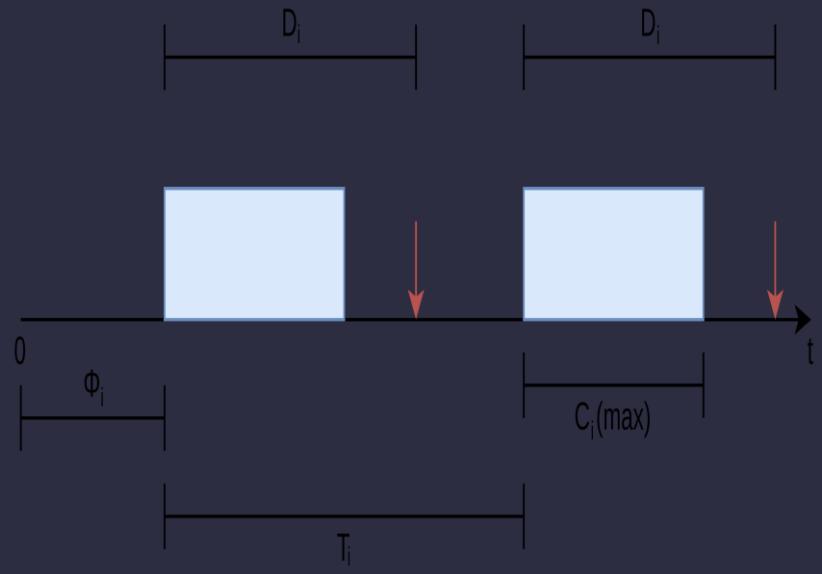
- Real-time deadlines: what happens when we miss them?
- Design patterns: producer-consumer, pipeline, watchdog



Week 6 — Architecture: FreeRTOS Task Model

Deep dive

- Tasks + priorities + preemption define responsiveness
- Queues/stream buffers for safe data handoff
- Tickless idle for low-power systems
- Use watchdogs + health checks for robustness

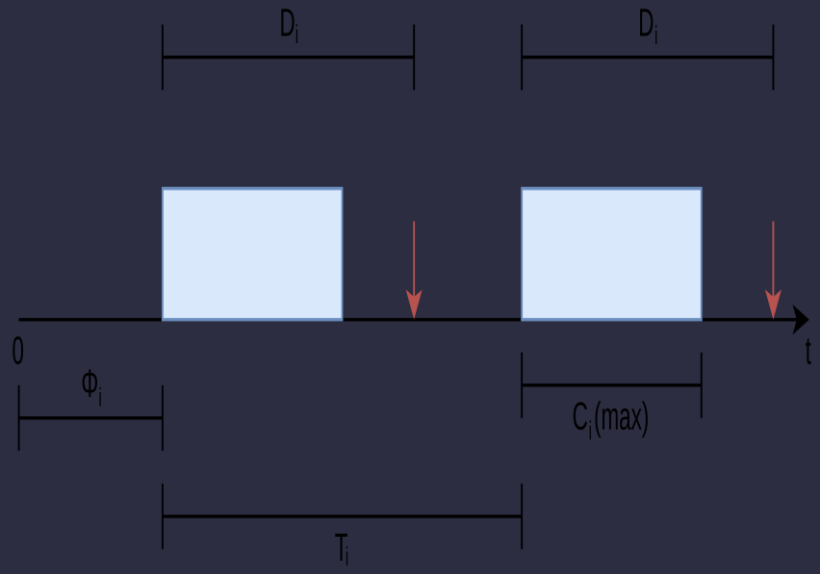


Common task model for real-time scheduling

Week 6 — Data Example: Producer → Consumer Queue

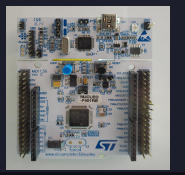
Deep dive

- Producer task samples sensors and pushes packets
- Consumer task formats + transmits over UART/radio
- Back-pressure: bounded queue avoids RAM exhaustion
- Measure jitter: timestamp before/after critical sections



Queues are a simple pattern for decoupling tasks

Week 6 — Checklist: RTOS Bring-Up



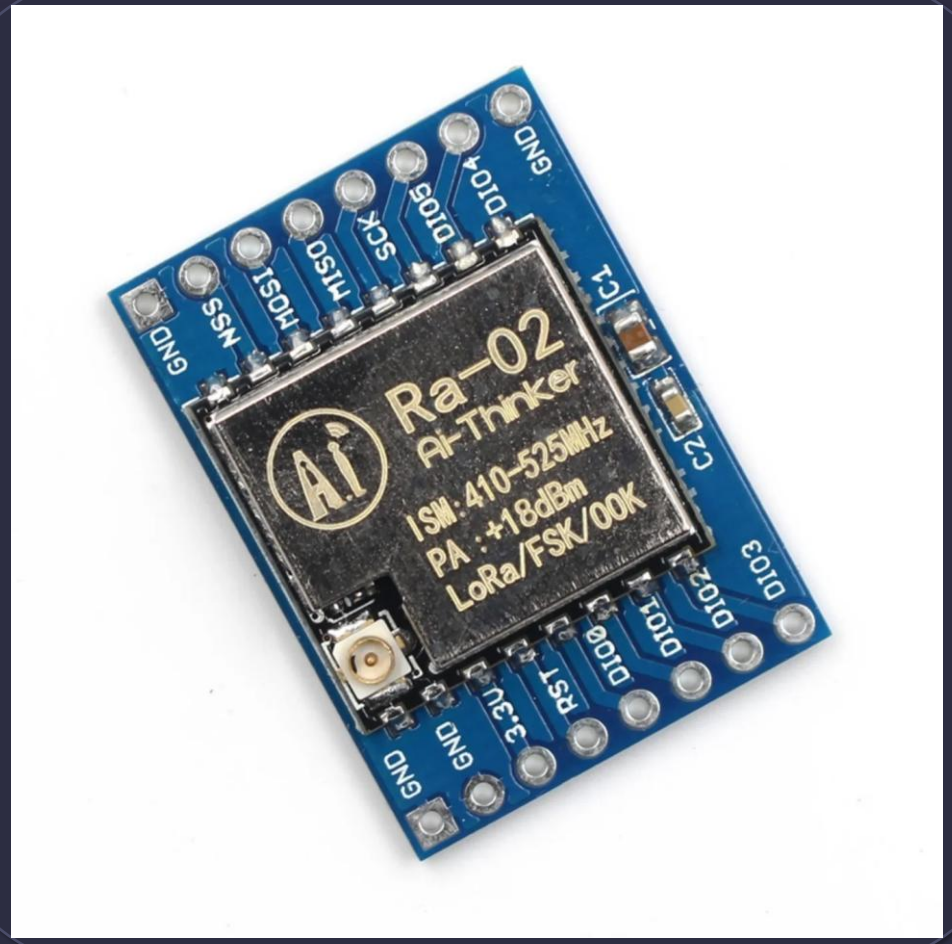
Deliverables

- Assign priorities and justify them (timing critical first)
- Size stacks using measurements (high-water marks)
- Protect shared resources (mutexes) and avoid deadlocks
- Add tracing/logging for task timing

RTOS makes timing explicit—measure it

WEEK 7

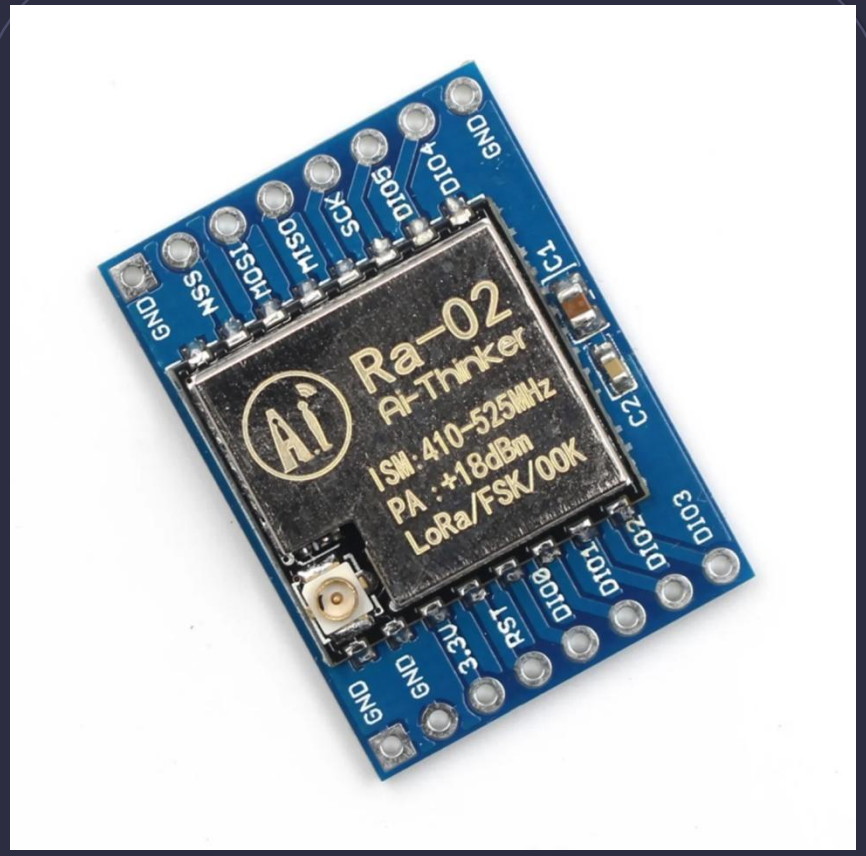
Wireless Communication for IoT Devices



Week 7 — Key Concepts

Key concepts

- Connectivity options: Wi-Fi, BLE, LoRa, Cellular
- Tradeoffs: range, power, bandwidth, cost
- Antenna basics + link budget intuition
- Gateways: bridging local links to the Internet

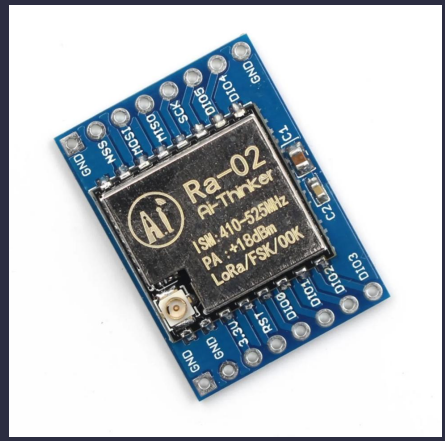


LoRa module example (SX1278)

Week 7 — Hands-on Lab

Hands-on lab

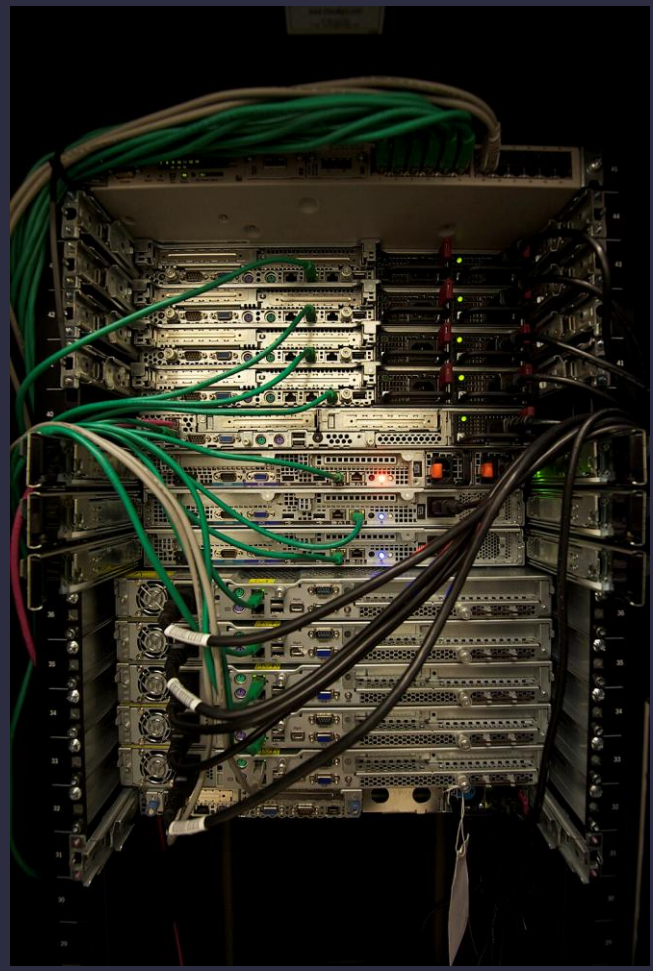
- Send payloads over a wireless link (demo setup)
- Retry logic + acknowledgements
- Measure RSSI / packet loss (if available)



Week 7 — Case Study & Discussion

Discussion prompts

- Designing a campus-scale sensor network
- Battery life estimation: duty cycle + payload size



Week 7 — Architecture: Wireless Link Budget Basics

Deep dive

- Range depends on frequency, power, antenna, and obstacles
- Data rate trades off with sensitivity and airtime
- Sleep modes dominate battery life—measure currents
- Regulatory constraints matter (band + duty cycle)

Example LoRa module (SX1278) used for long-range IoT

Week 7 — Data Example: Compact Payload Design

Deep dive

- Prefer integers + scaling (e.g., temp_c_x100)
- Include a sequence counter and battery level
- Add a simple checksum if the link is noisy
- Keep payload under network limits (LPWAN)

LPWAN payloads should be compact and robust

Week 7 — Checklist: Field Testing

Deliverables

- Test indoors and outdoors; record RSSI/SNR
- Verify antenna orientation and enclosure effects
- Measure battery life with real duty cycle
- Log failures (timeouts, retries) and refine parameters

Real environments behave differently than the lab

WEEK 8

IoT Communication Protocols & Cloud Connectivity

Week 8 — Key Concepts

Key concepts

- MQTT: publish/subscribe with topics
- QoS levels and retained messages
- HTTP/REST: request/response for device APIs
- Choosing protocol: telemetry vs commands vs firmware updates

MQTT logo (broker-centric messaging)

Week 8 — Hands-on Lab

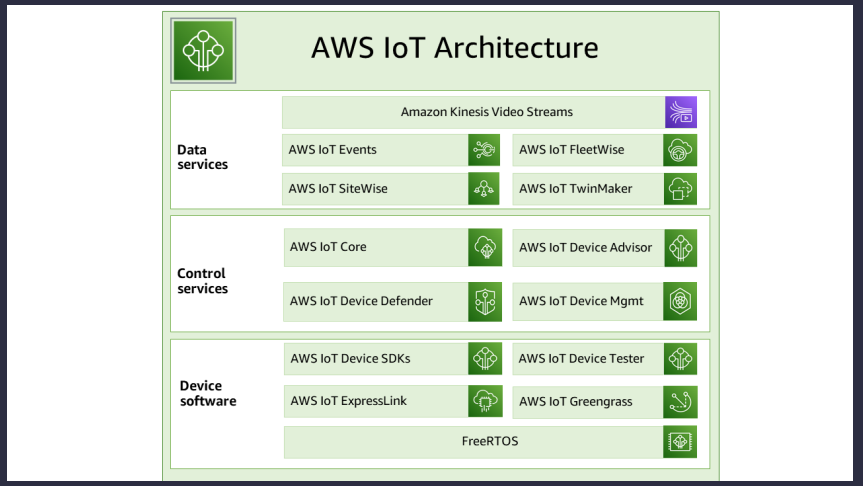
Hands-on lab

- Connect to a broker
- Publish sensor telemetry
- Subscribe to commands (LED / threshold)

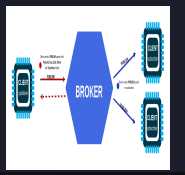
Week 8 — Case Study & Discussion

Discussion prompts

- Gateway pattern: local devices → gateway → cloud
- Offline buffering: store-and-forward

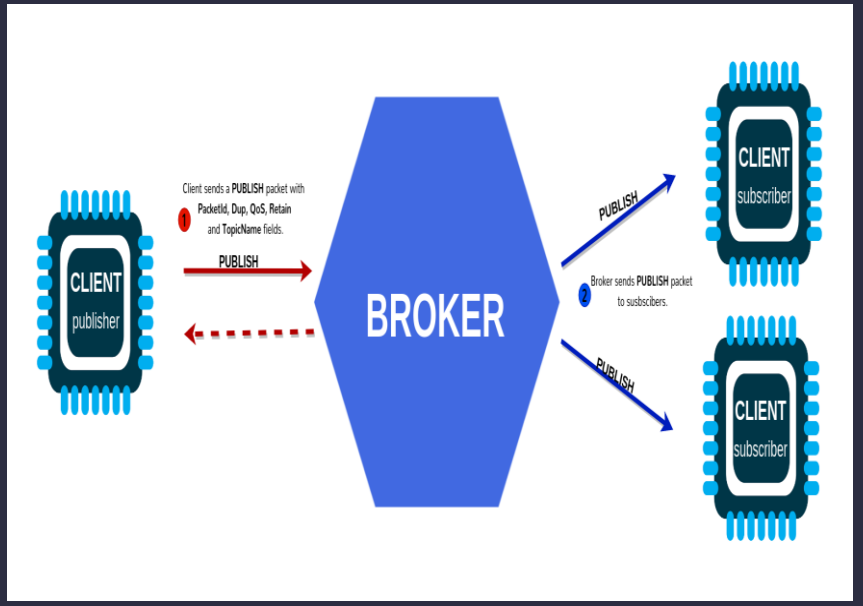


Week 8 — Architecture: MQTT Pub/Sub in IoT



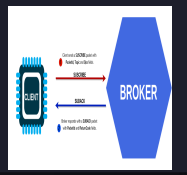
Deep dive

- Broker decouples devices from applications
- Topics create a flexible routing namespace
- QoS levels trade off reliability vs bandwidth
- Keep-alive and Last Will support robustness



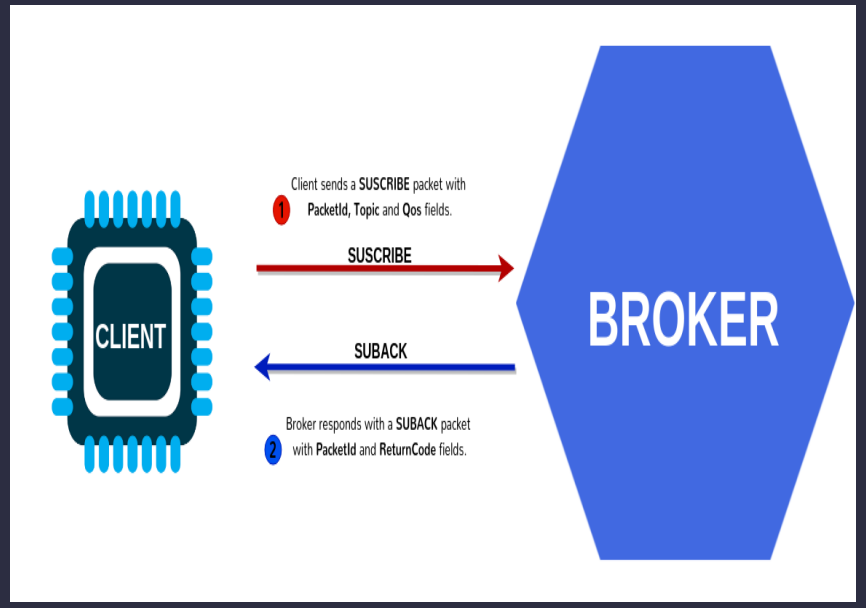
MQTT publish flow (publisher → broker → subscribers)

Week 8 — Data Example: Topic Design & Payload



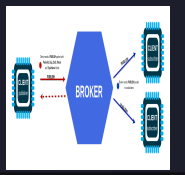
Deep dive

- Example topic: campus/lab1/device42/telemetry
- Separate telemetry, events, and commands
- Use retained messages for configuration/state
- Validate payloads at the gateway or broker bridge



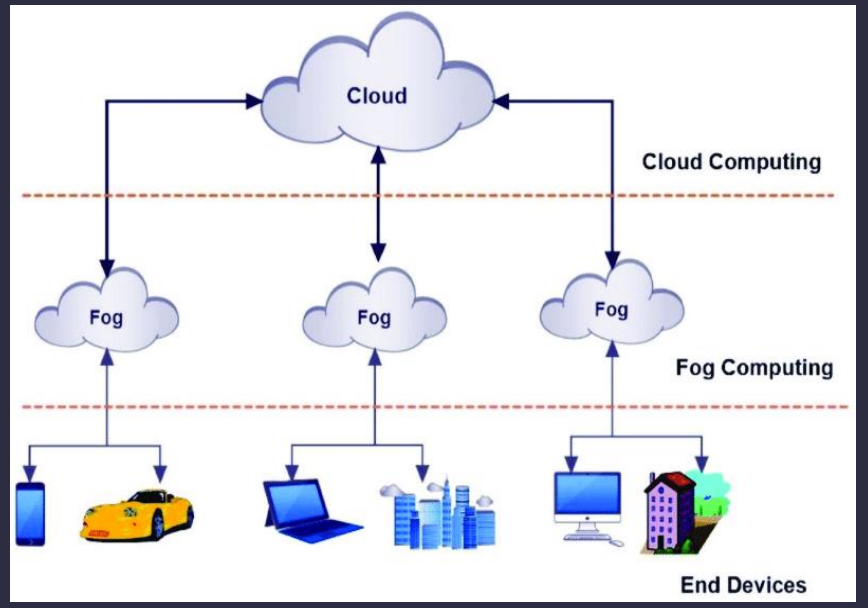
MQTT subscriptions select which topics to receive

Week 8 — Checklist: Cloud Connectivity



Deliverables

- Broker reachable (local or cloud) and credentials set
- Device reconnect logic tested (Wi-Fi drop simulation)
- QoS choice documented and justified
- Message rate measured; keep payloads under control



Connectivity choices affect latency and reliability

WEEK 9

IoT Data Storage & Time-Series Management

Week 9 — Key Concepts

Key concepts

- Time-series data: timestamp is the key
- Schema design: device_id, metric, value, ts
- Retention policies and downsampling
- Indexes and query patterns for dashboards

MySQL (example storage layer)

Week 9 — Hands-on Lab

Hands-on lab

- Create tables for telemetry
- Insert data from API/broker consumer
- Query: last value, min/max/avg over window

Week 9 — Case Study & Discussion

Discussion prompts

- From raw data to insights
- Alert conditions (thresholds, rate of change)

Week 9 — Architecture: Time-Series Data Pipeline

Deep dive

- Ingest → validate → store raw → derive aggregates
- Index on (device_id, timestamp) for fast queries
- Retention policies + downsampling reduce cost
- Separate hot (recent) vs cold (archived) storage

Databases are the backbone of IoT observability

Week 9 — Data Example: Basic SQL Query Pattern

Deep dive

- SELECT ts, value FROM telemetry WHERE device_id=? AND ts BETWEEN ? AND ?
- Aggregate: AVG(value) GROUP BY time bucket
- Use prepared statements to avoid injection
- Explain plans help with indexing decisions

A predictable query pattern makes optimization easier

Week 9 — Checklist: Storage Design

Deliverables

- Define table schema (device, ts, fields, schema_version)
- Decide retention and backup strategy
- Add indexes and validate query speed
- Design dashboards around common queries

Design storage around real query needs

WEEK 10

IoT Backend Development (REST API + Secure Ingestion)

Week 10 — Key Concepts

Key concepts

- API design: /devices, /telemetry, /commands
- Validation: schema checks + rate limiting
- Secure queries (prepared statements)
- Auth basics: API keys / JWT / sessions

Backend example: PHP

Week 10 — Hands-on Lab

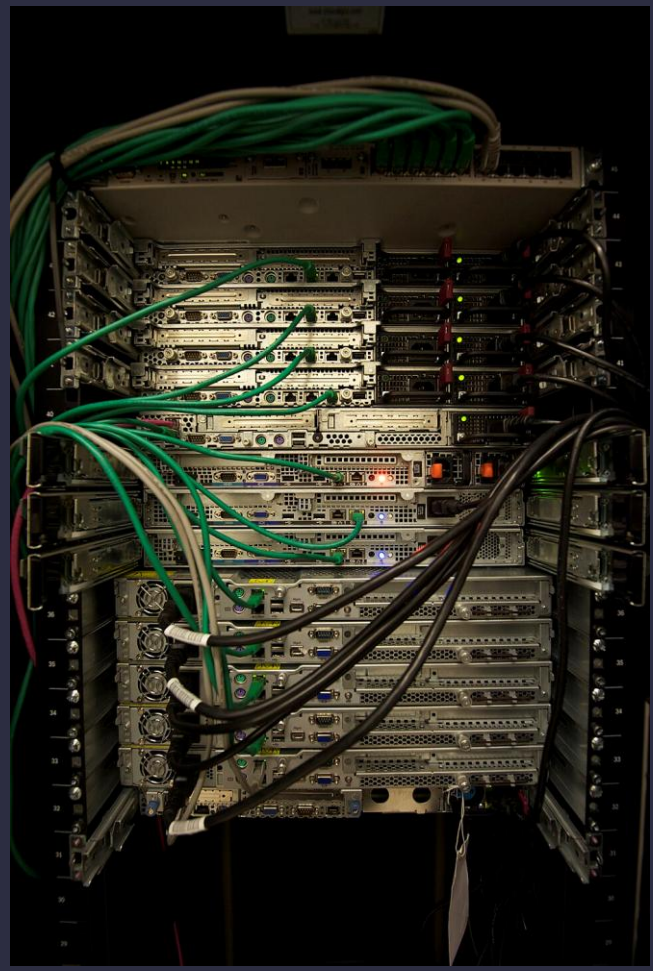
Hands-on lab

- Build a telemetry ingest endpoint
- Store payloads in DB
- Return latest values for dashboard

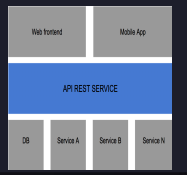
Week 10 — Case Study & Discussion

Discussion prompts

- Scaling ingestion: batching and async processing
- Deployment checklist: logs, backups, monitoring

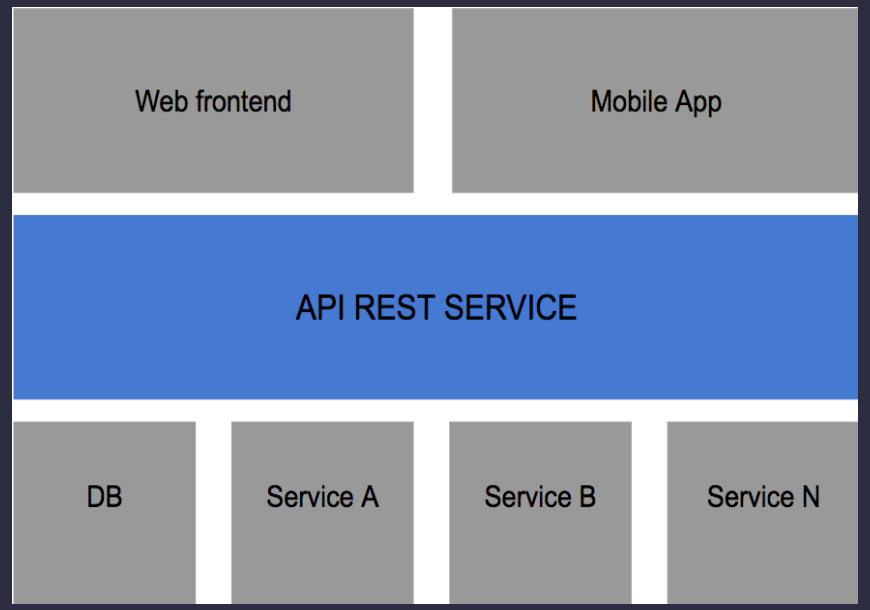


Week 10 — Architecture: Secure Ingestion API



Deep dive

- Device/gateway sends telemetry to REST endpoint
- Server validates auth + schema + rate limits
- Write to DB or enqueue for async processing
- Return acknowledgements + error codes

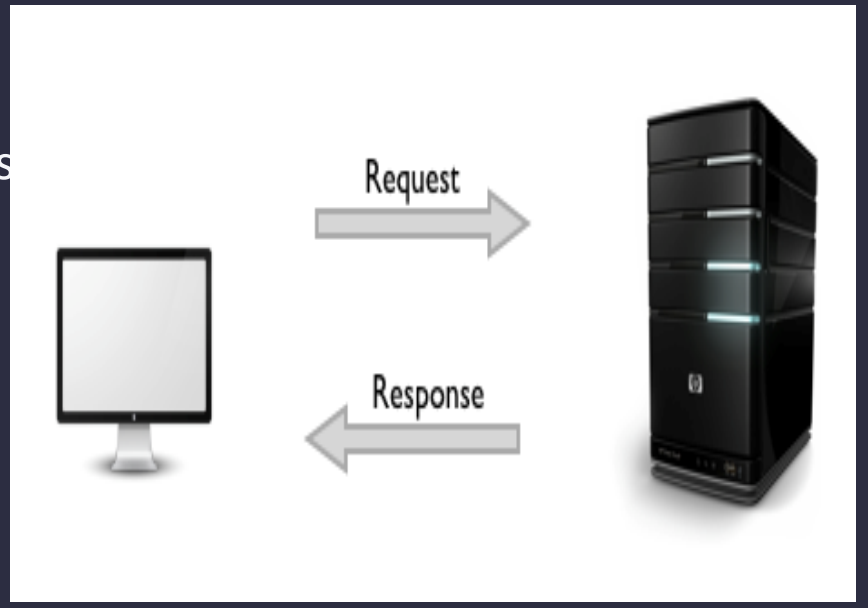


A common REST API structure in IoT platforms

Week 10 — Data Example: HTTP Request/Response

Deep dive

- POST /telemetry with JSON body and auth header
- Use idempotency keys for retries
- Return 2xx for success, 4xx for client errors, 5xx for server is
- Log correlation IDs for debugging

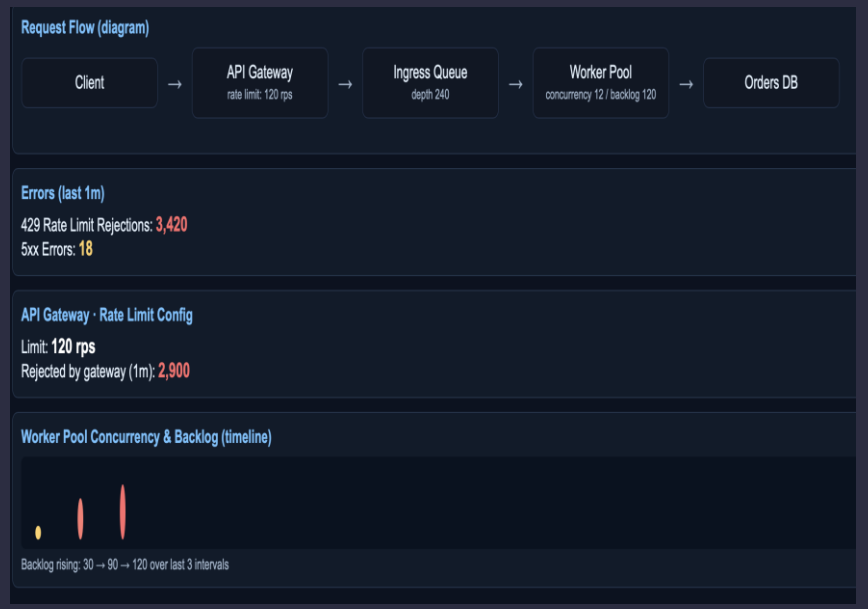


HTTP request–response model (simplified)

Week 10 — Checklist: Backend Reliability

Deliverables

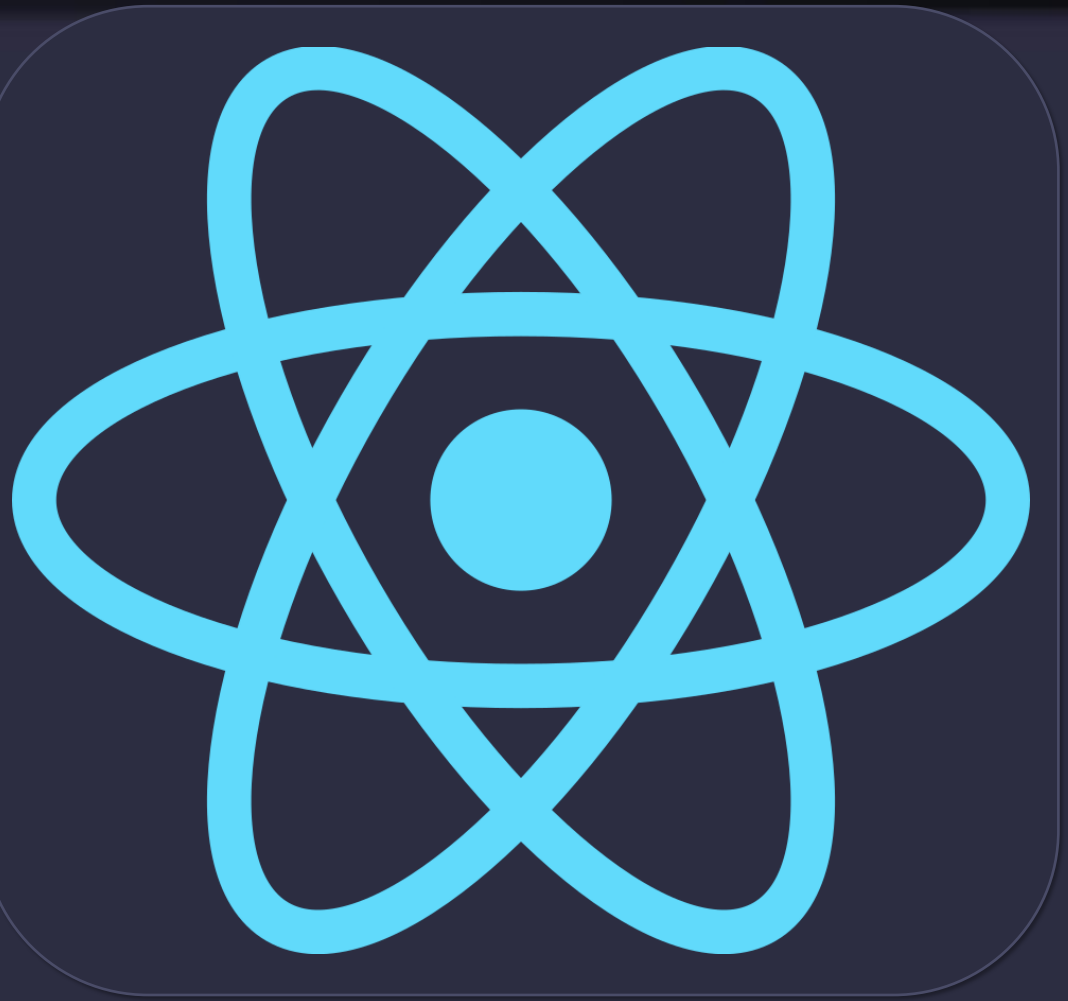
- Input validation + schema versioning implemented
- Rate limiting tested; backpressure via queue
- Secure secrets handling (env vars, vault, etc.)
- Basic monitoring: latency, error rate, throughput



Gateway + queue + worker pool is a resilient pattern

WEEK 11

IoT Dashboard Development (React + Monitoring)



Week 11 — Key Concepts

Key concepts

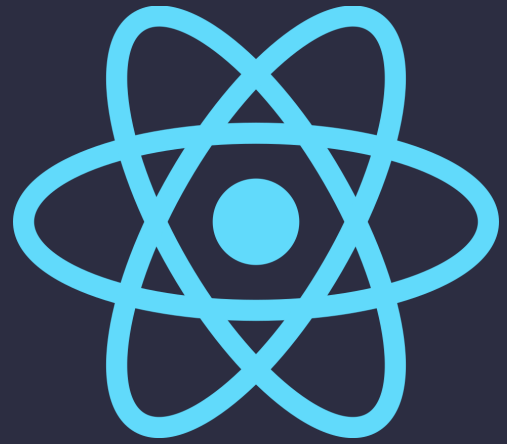
- Dashboards: realtime vs near-realtime
- Polling vs WebSockets
- Charts: last hour / day / week
- User experience: filters, device lists, alerts

Grafana dashboard example

Week 11 — Hands-on Lab

Hands-on lab

- Create a basic React dashboard
- Fetch data from REST API
- Render a simple time-series chart

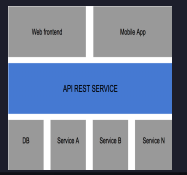


Week 11 — Case Study & Discussion

Discussion prompts

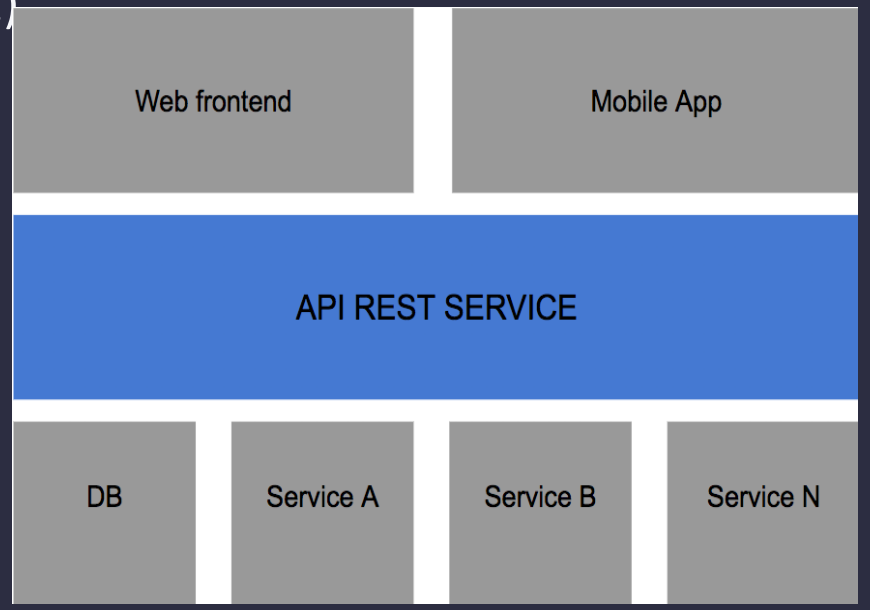
- Operational monitoring: uptime, latency, error rates
- Alert routing and escalation

Week 11 — Architecture: Dashboard Data Flow



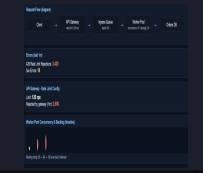
Deep dive

- Dashboard reads from API (and optionally WebSockets/SSE)
- Queries are time-range based with aggregation
- Charts: time-series, map, alerts, device status
- Cache and pagination keep the UI fast



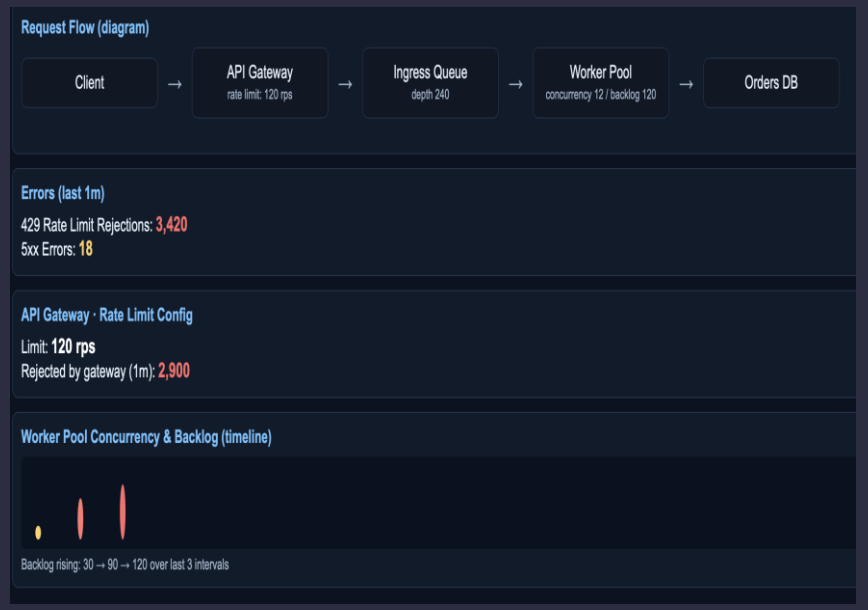
Dashboard ↔ API ↔ Database data flow

Week 11 — Data Example: Monitoring View



Deep dive

- Track p95 latency, error rate, and ingestion throughput
- Alert on sustained failures, not single spikes
- Visualize backlog to detect overload early
- Separate “system” metrics from “device” metrics



Example monitoring-style dashboard elements

Week 11 — Checklist: Useful Dashboard Features

Deliverables

- Time range picker + device filter
- At least 3 core charts (telemetry, status, errors)
- Drill-down page per device
- Export/inspect raw data for debugging

Dashboards are only as good as the data model

WEEK 12

IoT Security & Device Authentication

Week 12 — Key Concepts

Key concepts

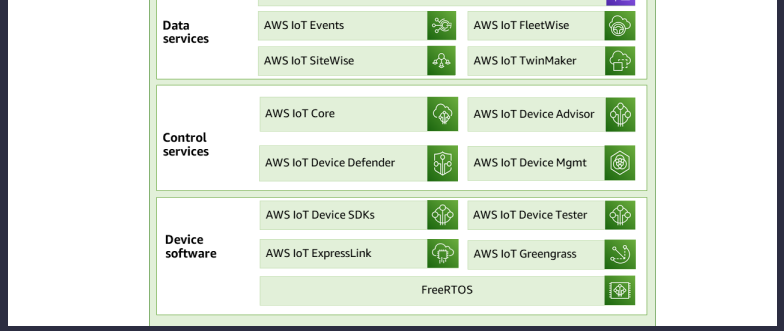
- Threats: spoofing, replay, weak credentials, insecure updates
- Transport security: TLS for MQTT/HTTPS
- Device identity: certificates vs tokens
- Secure boot & OTA update concepts

Security mindset: least privilege

Week 12 — Hands-on Lab

Hands-on lab

- Add authentication to API endpoints
- Input validation & error messages
- Store secrets safely (never hard-code in repo)



Week 12 — Case Study & Discussion

Discussion prompts

- Security checklist for a student project
- What to document: keys, roles, update process

Week 12 — Architecture: IoT Security Layers

Deep dive

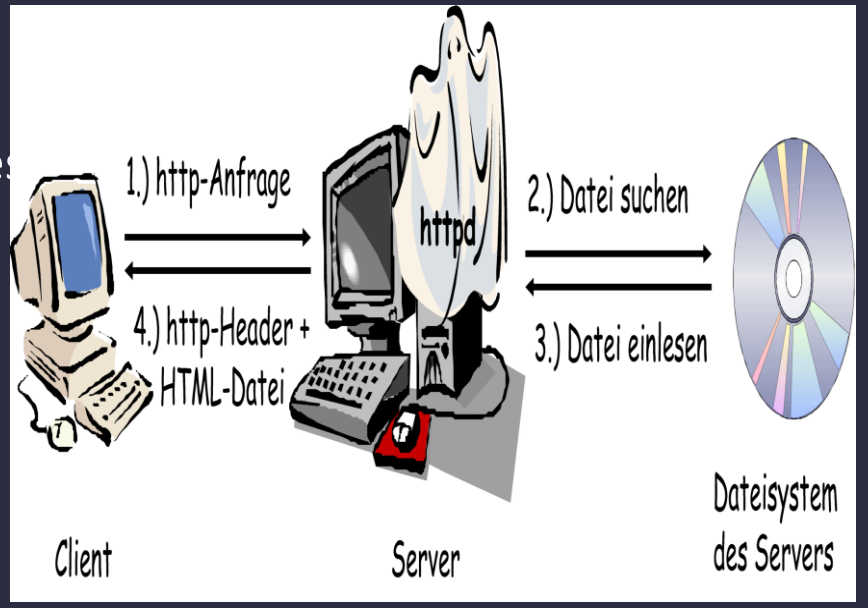
- Threat model: physical access + network attackers
- Device identity: per-device keys/certificates
- Transport security: TLS/DTLS + certificate validation
- Principle of least privilege for APIs and brokers

Security must span device, network, and cloud

Week 12 — Data Example: Auth Token + Claims

Deep dive

- JWT (or similar) can carry deviceId + expiration
- Short-lived tokens reduce risk
- Rotate secrets; support revocation for compromised devices
- Never store secrets in firmware as plain text



Secure APIs typically rely on authenticated requests

Week 12 — Checklist: Security Hygiene

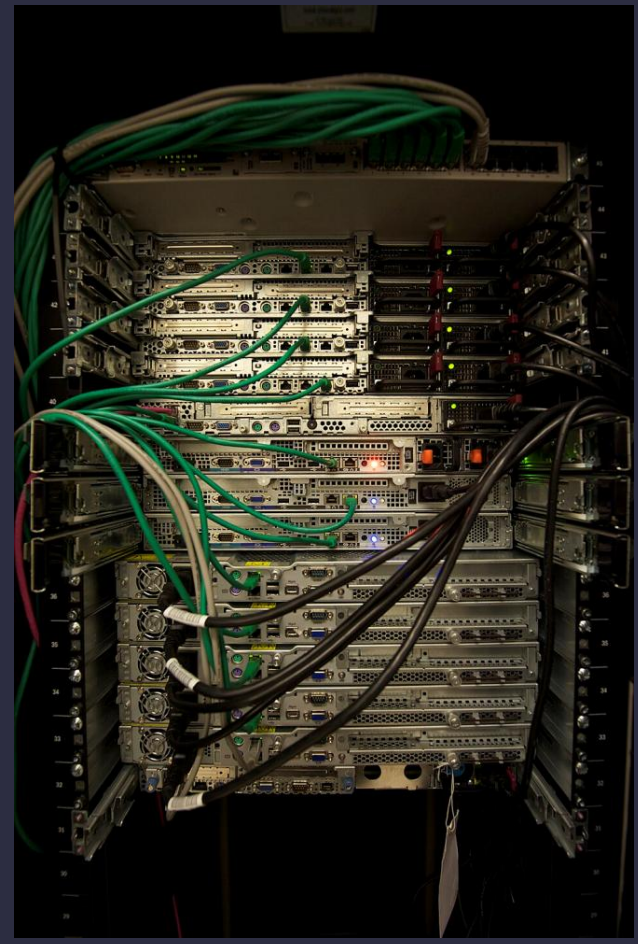
Deliverables

- Unique device credentials (no shared default passwords)
- TLS enabled and verified (no “skip cert check”)
- Audit logs: who sent what, and when
- Document incident response: rotate, revoke, patch

Security is a process, not a feature

WEEK 13

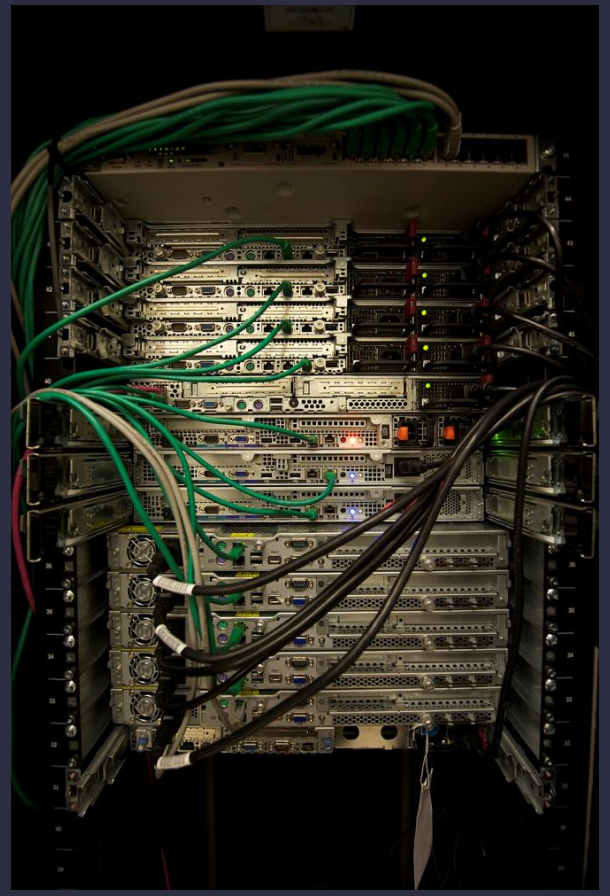
IoT Deployment & Edge-to-Cloud Integration



Week 13 — Key Concepts

Key concepts

- Deployment options: on-prem, VPS, managed cloud
- Observability: logs, metrics, traces
- Performance: caching, compression, connection reuse
- Reliability: retries, idempotency, backups

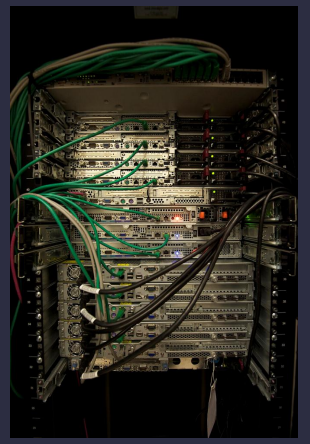


Servers & infrastructure (deployment target)

Week 13 — Hands-on Lab

Hands-on lab

- Deploy backend + DB
- Set up HTTPS
- Smoke test with a simulated device
- Monitor basic metrics



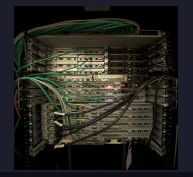
Grafana

Week 13 — Case Study & Discussion

Discussion prompts

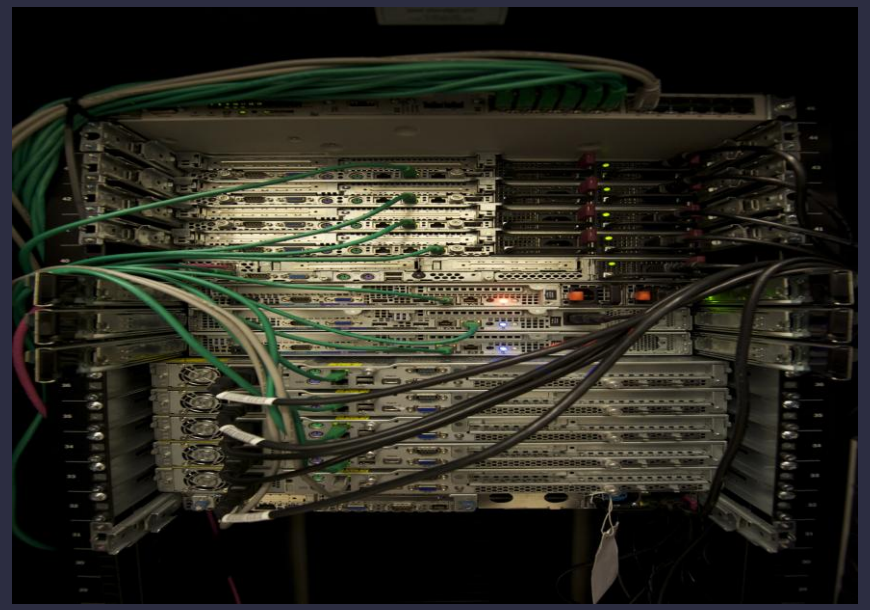
- Final project integration plan
- Demo day: what to show (device, data, dashboard, security)

Week 13 — Architecture: Deployment Topology



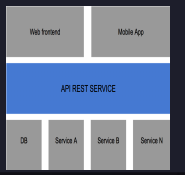
Deep dive

- Edge devices → gateway → cloud services → dashboard
- Separate environments: dev / staging / production
- Observability: logs, metrics, traces
- Rollback strategy: versioned releases + backups



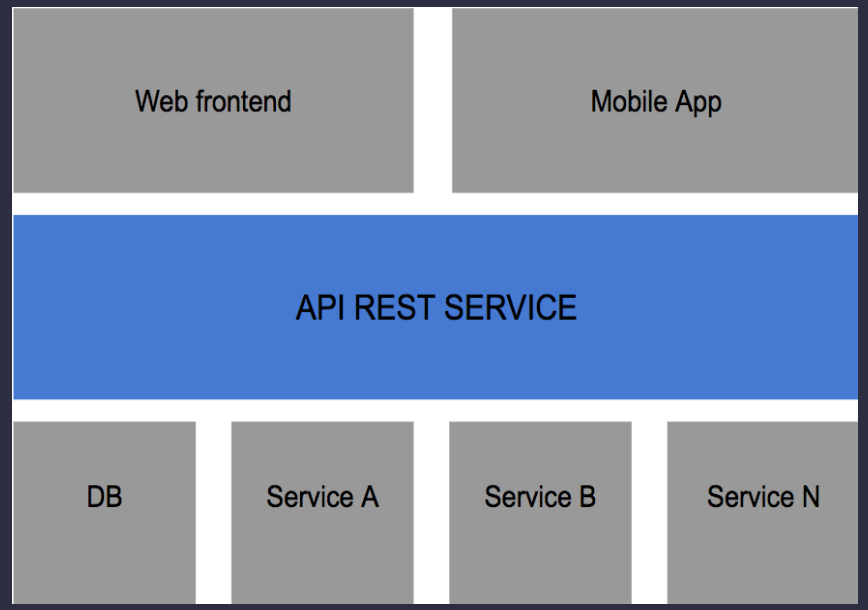
Infrastructure matters for reliability and scaling

Week 13 — Checklist: End-to-End Integration



Deliverables

- Device can send telemetry end-to-end (device → DB → UI)
- Auth works for devices and users
- Rate limits + retries behave under stress
- Final demo scenario documented with success criteria



End-to-end integration is the final goal

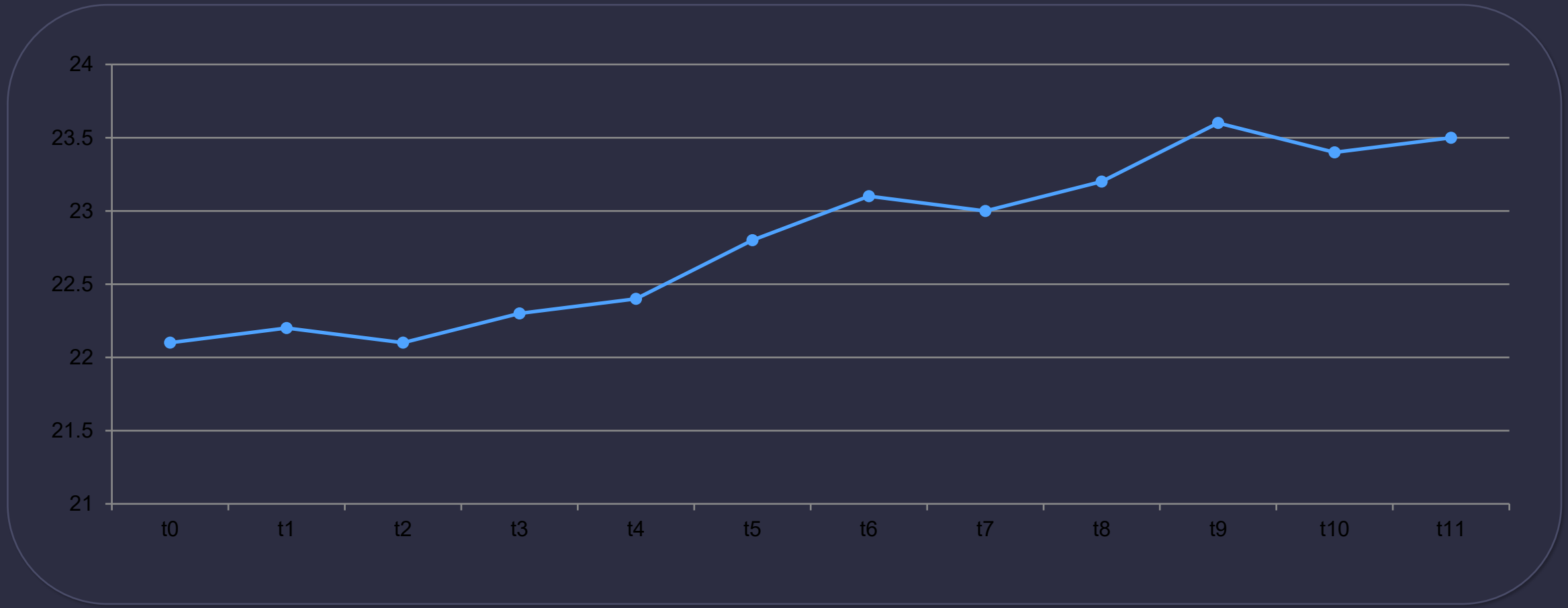
Capstone Project: A Connected Multi-Sensor Node

Goal: Build an end-to-end demo that feels like a real IoT product.

- Device reads ≥2 sensors and timestamps data
- Sends telemetry via MQTT or HTTP
- Backend stores data securely
- Dashboard shows charts + latest values
- Bonus: commands (actuation) + alerts + authentication

Example Telemetry: Temperature Over Time

Illustrative chart (synthetic data) — use it to discuss sampling, noise, and alerts.



Example alert: if Temperature > 23.5°C for 2 minutes → notify.