
# Ain Shams University
# Faculty of Engineering
# International Credit Hours Engineering Program

## CSE351: Computer Networks
## Project Proposal

Team 22

Boulos Wagdy Anis 22P0305

Carol Rimoun Ramsis 2201357

Farida Tawfik Abdou 2201367

George Mourad Mourice 22P0114

Maryam Saadedlin Ghalab 22P0265

Patrick Hany Adel 2200557

# Table of Contents

# 1. ASSIGNED SCENARIO

**Project 1 — IoT Telemetry Protocol (Sensor Reporting)**

# 2. PROJECT TITLE

**NanoTelemetry Protocol v1.0 (NTP-v1)** — Lightweight UDP Telemetry for IoT Sensor Reporting

# 3. MOTIVATION

In this project, we aim to design and implement a custom lightweight telemetry protocol for constrained IoT sensors that periodically send small readings (e.g., temperature, humidity, or voltage) to a central collector. Our goal is to design a compact, efficient, and robust application layer protocol operating over UDP, and to evaluate its performance under varying network conditions experimentally.

IoT sensors often run on constrained devices that must periodically send small readings such as temperature or humidity to a central collector.
Existing protocols like MQTT-SN or CoAP are have a lot of features that are relatively unsuitable for small devices and usually depend on TCP or retransmission logic, which increases overhead.
Our NanoTelemetry v1.0 provides a compact, loss-tolerant UDP-based protocol suitable for environments where minor packet loss is acceptable but simplicity and low overhead are our priority.

# 4. PROPOSED APPROACH

Our team will design and implement **NanoTelemetry Protocol v1.0 (NTP-v1)** consisting of mainly three sides:

- **Client (Sensor):** Who sends DATA and HEARTBEAT messages via UDP.
- **Server (Collector):** Who receives, validates, and logs messages to a CSV file.
- **Scripts:** Which automate the baseline local test scenario.

Our protocol will:

- Operate over UDP without retransmission.
- Include a compact binary header (≤ 12 bytes) containing version, message type, flags, device ID, sequence number, and timestamp.
- Support two message types:
    - **DATA** – contains one float sensor reading.
    - **HEARTBEAT** – sent when no data is available to indicate liveness.
- Detect duplicates and sequence gaps at the server.
- Log results as device_id,seq,timestamp,arrival_time,duplicate_flag,gap_flag.

## 5. EXPECTED OUTPUT (PHASE 1)

- A working prototype showing UDP DATA/HEARTBEAT exchange locally.
- A CSV log with all received packets.
- A shell script to automatically run the baseline scenario.
- A README with clear setup instructions.

## 6. FEASIBILITY

All functionality fits comfortably within the Phase 1 timeframe.
Implementation is less than 200 lines of Python and runs cross-platform (tested on macOS, and Windows).
Further features such as batching, multi-device testing, and performance measurement will be implemented in later phases.