

Test Technique – Recrutement IoT

Projet: Smart Multi-Node IoT System avec API REST + Dashboard embarqué

Durée: 7 à 10 jours

Matériel requis: ESP32 (x2 ou simulation Wokwi)

Langage: Arduino C++ uniquement

1. Objectif général

Concevoir un **système IoT distribué** composé de plusieurs ESP32 capables de:

- Des nœuds capteurs collectent des données (température, humidité)
- Un nœud central reçoit ces données (logique HTTP simulée)
- Les données sont affichées dans un **dashboard web embarqué**
- L'ensemble est testable via Wokwi (ou démontré par simulation partielle)

2. Architecture minimale attendue

Nœud capteur (ESP32 1) :

- Lit un capteur de température/humidité (DHT11 ou DHT22)
- Formate les données en JSON
- Les "envoie" vers un nœud maître via :
 - **HTTP POST (si matériel réel)**
 - **Serial.print simulant une requête POST** (si Wokwi uniquement)
- Répète toutes les 10 secondes (à modifier au besoin)

Nœud central (ESP32 2 ou simulation partielle) :

- Reçoit les données simulées (Serial input ou mock code)

- Affiche :
 - les données en temps réel
 - min, max, moyenne
 - l'état du capteur (actif, inactif)
- Sert un **dashboard web embarqué** (via SPIFFS dans le code – non simulé sur Wokwi)

3. Variante 100% Wokwi (si pas de 2e ESP32 réel)

Le candidat peut:

- Simuler le nœud capteur dans Wokwi avec affichage OLED
- Simuler le traitement serveur avec une fonction dédiée qui lit les données du capteur, les affiche, les stocke en JSON en mémoire, et simule la logique HTTP
- Inclure un `Serial.println("POST /api/data...")` pour illustrer la transmission
- Fournir le code du serveur central, même non testable sur Wokwi, bien structuré

4. Livrables attendus

- **Simulation Wokwi publique (lien à fournir)**
- Code complet sur GitHub
- Schéma d'architecture (png ou dessin clair)
- Bref rapport technique expliquant :
 - Les choix techniques
 - La manière dont la logique de communication est simulée
 - Ce qui a été testé et ce qui ne l'a pas été
 - Les suggestions d'amélioration

5. Interdictions techniques

- Pas de bibliothèques “prêtes à tout faire” pour le serveur HTTP
 - Tu peux utiliser `WebServer` ou `ESPAsyncWebServer`, mais **pas de frameworks “no-code”** ou de générateurs automatiques d’API.
- Pas d’accès à Internet externe
 - L’ESP32 doit fonctionner en **réseau local uniquement**, sans dépendre d’API publiques, de NTP, ni de serveur cloud.
- Pas de `delay()` pour gérer le temps d’envoi
 - Utilise `millis()` ou une logique non-bloquante.
- Pas d’interface web externe (BootstrapCDN, etc.)
 - Le dashboard doit être **100% embarqué via SPIFFS** (ou simulé dans le code si sur Wokwi), sans dépendance à Internet.
- Pas de `String` dynamique à répétition dans les boucles
 - Préférence pour les `char[]` ou `String` statiques.

6. Structure du dépôt GitHub attendue

```
smart-multinode-iot/  
├── capteur-sim-wokwi/  
│   ├── main.ino  
│   └── README.md  
├── serveur-logic/  
│   ├── main.ino  
│   └── web/  
│       ├── index.html  
│       └── script.js  
├── docs/  
│   ├── architecture.png  
│   └── notes.md  
├── wokwi-diagram.json  
├── README.md  
└── LICENSE
```

7. Grille d'évaluation

Critère	Points
Lecture capteur + traitement correct	20
Transmission (HTTP réel ou simulée)	15
Structure du code et modularité	15
Interface web (même non testée)	15
Qualité de la simulation Wokwi (interactions, réalisme)	15
Documentation + clarté	10
Bonus (authentification, OLED, stats, config dynamique, etc.)	10
Total	/100