Master en Architecture des Systèmes Informatiques & Master Ingénieur Industriel Informatique
Consignes pour l'examen d'**IoT** – 1ère session – Janvier 2025

1. PRESENTATION GENERALE

Vous devez concevoir un projet IoT répondant aux critères ci-dessous. Le projet sera réalisé par équipe de 2 étudiants. Attention, chaque étudiant doit être capable de répondre à des questions sur l'entièreté du projet (pas question de dire : "ce n'est pas moi qui ai travaillé sur cette partie"). Votre projet doit être innovant, actuel. L'objet IoT doit être **utile** et **utilisable**. Imaginez un scénario réaliste. Soyez ingénieux! Vous être le fondateur d'une startup et vos clients potentiels doivent avoir envie d'acheter votre objet connecté. Cela n'a pas de sens de prévoir un objet connecté qui coûte des centaines d'euros pour protéger une ressource qui coûte cinquante euros...

Cet objet permettra de solutionner un problème que vous avez identifié dans un domaine spécifique. Les domaines d'utilisation de l'IoT sont variés et nombreux : Santé, Energie, Environnement, Mobilité, Education, Loisirs, Sports, Animaux de compagnie, Commerce, Agriculture,…

Il est souhaitable de parler de votre projet à un professionnel du domaine choisi. Par exemple, si votre objet concerne l'agriculture, les soins de santé,… , il est utile d'avoir un retour sur l'utilité de votre objet.

Une fois que vous avez défini un problème que vous voulez solutionner, vous devez trouver un **titre** pour votre projet. Ce titre doit être accrocheur et on doit pouvoir tout de suite savoir à qui s'adresse l'objet connecté (utilisation de "connecté" ou "smart" ou ...).

Votre startup va réaliser un prototype d'objet connecté que vous souhaitez commercialiser. Dans ce cadre, L'architecture de votre projet IoT doit comprendre **plusieurs niveaux.**

- Niveau 1 : les objets connectés et leurs capteurs : c'est à ce niveau que l'on retrouvera les cartes ESP32 et Heltec-esp32s3-oled-LoRa. NB : Le Raspberry Pi sera au niveau 1 ou au niveau 2 ou au niveau 3.
- Niveau 2 : l'utilisateur de l'objet connecté qui va pouvoir interagir avec le ou les objets connectés pour fixer des seuils, pour recevoir des informations,… Les interactions s'effectueront par exemple via un smarphone. Le niveau 2 est connecté au niveau 1 par exemple en WiFi ou en Lora. On peut aussi imaginer que les données des capteurs partent dans le Cloud et que l'utilisateur se connecte via le Web pour interagir avec les objets connectés
- Niveau 3 : vous et votre entreprise. Vous n'avez pas qu'un seul client. Vous avez vendu votre solution à différents clients indépendants les uns des autres, mais qui ont tous été séduits par votre solution. Vous restez connecté avec les objets qui sont chez vos clients pour réaliser la télémaintenance, pour connaître l'utilisation qui est faite de vos objets, pour réaliser des mises à jour, pour pouvoir faire du reporting chaque semaine, chaque mois… C'est à ce niveau que se situe notamment une base de données. Le niveau 3 est connecté au niveau 2 à travers Internet.
- Niveau 4 : le Cloud : vous utilisez des services Cloud pour obtenir ou stocker des informations. Par exemple, vous utilisez des APIs dans le Cloud, un broker public MQTT,...

Par exemple, si votre objet connecté concerne les soins de santé, on pourrait retrouver au niveau 1 votre objet connecté qui est porté par le patient. Au niveau 2, on retrouve l'utilisateur, par exemple le médecin ou l'infirmière qui gère à son niveau différents objets et qui recoît les données pour par exemple ajuster le traitement des patients. Au niveau 3, on vous retrouve, vous et votre entreprise. Vous restez en contact pour offrir du service et du support à vos clients. Au niveau 4, on retrouve les services Cloud que vous utilisez pour par exemple stocker les données et que vous accédez via des API REST.

2. ARCHITECTURE DU PROJET

Pour présenter votre solution, vous avez rendu une première version du schéma de l'architecture de votre projet (1 page A3 ou A4 – format paysage - pas manuscrit – schéma en couleurs – attention à la taille

des caractères une fois le schéma imprimé - utilisez au mieux l'espace disponible – pas de zones vides – couleurs de fond claires différentes pour les 4 cadres représentants les différents niveaux). Vous avez utilisé par exemple https://app.diagrams.net pour réaliser ce schéma.

Ce schéma d'architecture doit comprendre tous les composants de votre projet et vous y indiquerez les interactions entre tous ces composants (lignes ou flèches) et sur ces lignes/flèches, indiquez les modes de transmission (au moins 2 logos par ligne ou flèche : par expl Lora et MQTT, WiFi et HTTPS, WiFi et VNC, WiFi et email, SMS et 4G, …). Utilisez des couleurs différentes par protocole pour ces lignes ou flèches et choisissez judicieusement l'épaisseur de ces lignes ou flèches. Ces lignes/flèches seront unidirectionnelles. En observant votre schéma, on doit pouvoir comprendre à quoi sert l'interaction. Au besoin, utilisez des petits logos supplémentaires (par expl : triangle avec point d'exclamation pour indiquer qu'il s'agit d'une alerte ou un logo battery low pour prévenir qu'il convient de recharger). Indiquez également sur les lignes ou flèches si la connexion est sécurisée en ajoutant par expl le logo d'un petit cadenas.

Quand vous envoyez un mail dans le cloud, il faut que quelqu'un reçoive l'info. Cela n'a pas de sens si personne ne vient lire ce mail… On doit pouvoir facilement visualiser le producteur (la source) d'un message et le consommateur correspondant (la destination). (idem pour SMS, Webex, MQTT, REST, …)

Quelques contraintes/remarques suite à votre avant-projet :

- Il n'est pas judicieux de mettre des photos de RPi, de capteurs,… Par contre, vous utiliserez des icones pour les différents composants (WiFi, LoRa, MQTT, Node-Red, Python, Webex, RPi, Mosquitto, base de données, Arduino IDE, Twilio, Ubuntu, Shodan, ESP32, VNC, caméra,…). L'utilisateur de l'objet connecté sera également représenté par une icone. "Une image vaut mille mots". Utilisez également des icones/images pour les particularités de votre projet : moto, plante, enfant, maison, personnel soignant, animal, … En regardant votre schéma, on doit comprendre à quoi sert votre objet connecté.
- si le logo est bien choisi, il n'est pas souhaitable d'ajouter du texte (par expl : raspberry pi, caméra,…) Par contre, si le logo n'est pas suffisamment explicite, ajouter un texte bien choisi pour signaler de quoi il s'agit.
- Il s'agit d'un seul schéma et pas de plusieurs petits schémas. identifiez clairement les 4 niveaux. Il doit être possible de comprendre clairement où se situe chaque niveau. Par exemple, vous indiquerez : niveau 1 : poignet du patient, niveau 2 : bureau des infirmières,...
- A côté des cartes ESP32 et Heltec-esp32s3-oled-LoRa et du RPi, indiquez quels capteurs/actuateurs sont connectés. Rappel : un logo est meilleur qu'un mot.
- Les VMs Ubuntu 22.04 utilisées seront également représentées avec leur rôle (MQTT Broker, DB, …) ainsi que le smartphone utilisé pour visualiser les dashboards, ...
- Les langages/outils utilisés apparaîtront à côté des composants concernés: Python, Micropython, Node-Red, Arduino, C/C++,...
- On peut évidemment avoir plusieurs ESP32 ou plusieurs RPi par client. Pour la démo, vous n'en avez qu'un seul, mais votre architecture peut en montrer plusieurs…
- Pour bien montrer qu'il existe plusieurs objets L1 contrôlés par un L2 et plusieurs clients gérés par votre société (L3), vous écrirez, à côté de niveau 2 : "gestion de x …." (par exemple, gestion de dizaines de caddies connectés). De même, à côté de niveau 3 (votre entreprise), vous écrirez : "gestion de x..." (par exemple, gestion de centaines de magasins). Remplacez "x" par un nombre réaliste !
- Note sur l'utilisation des logos : ne pas changer la couleur "officielle" d'un logo.
- L'impression de ce schéma doit être en couleurs et de qualité, sans numéro de page et entièrement en format paysage.
- Si votre schéma est difficilement lisible notamment à cause d'un grand nombre de lignes/flèches, vous devrez changer l'agencement des composants reliés par ces lignes/flèches.
- Ne pas utiliser de légende en dessous de votre schéma d'architecture : un logo WiFi sur une flèche est nettement mieux qu'une ligne pointillée qui renvoie à une légende.

3. CONTRAINTES TECHNIQUES MINIMALES

Ci-dessous, vous trouverez les contraintes techniques **MINIMALES**. Vous devez intégrer des composants additionnels. Soyez innovants !

Au minimum, vous devrez utiliser :

- Un raspberry pi avec carte GrovePi et adaptateur LA66-LoRaWan
- Un ESP32 qui communiquera en WiFi et en MQTT (utilisation de Micropython)
- 2 machines virtuelles Ubuntu 22.04 avec utilisation d'au moins un container docker
- Un broker public du type test.mosquitto.org avec utilisation de **certificats** (test.mosquitto.org/ssl) & port 8884 ainsi qu'un broker privé qui tournera sur une des VM Ubuntu 22.04
- Votre smartphone pour vous connecter à une interface Web (dashboards Node-Red) et également à une room web.webex.com.
- Une carte Heltec-esp32s3-oled-LoRa
- L'afficheur LCD connecté au GrovePi

La connexion au réseau WiFi sera sécurisée en WPA2-PSK (connection via 4G ou WiFi IOT).
L'accès à l'interface graphique du RPi s'effectuera en VNC à partir d'une VM Ubuntu 22.04.

L'ESP32 fera au minimum l'acquisition de 2 mesures à l'aide de capteurs et transmettra les résultats en MQTT via le réseau WiFi.

Votre projet montrera que vous maîtrisez Node-Red, Python (utilisation de la librairie Paho et de la librairie multithreads "threading", micropython, MongoDB (utilisation du noeud MongoDB4) et les API REST.
En ce qui concerne les API REST, vous devez au moins ajouter un service web utile à votre projet et **non vu au cours**. L'accès s'effectuera avec la librairie Python "requests". Vous intégrerez également au moins une requête à l'API shodan.
En outre, vous écrirez votre propre API REST avec authentification (vous pouvez par exemple utiliser le micro framework Flask comme montré au cours)

Votre projet utilisera au moins 5 parmi les capteurs suivants : distance, luminosité, capteur angulaire, capteur de son, de température, d'humidité, bouton, GPS.
Votre projet utilisera au moins 2 sorties parmi les suivantes : Buzzer, LED, Relais.
Votre projet proposera au moins 2 types d'accès : administrateur et utilisateur de l'objet connecté.
Des notifications signaleront des événements par mail et également par SMS.

Du logging sera mis en place et sauvegardé dans la base de données sur une VM Ubuntu 22.04 (attention : le timestamp doit être présent). Il faudra pouvoir afficher un historique des mesures (dates à encoder dans formulaire via dashboard Node-Red ou interface web). Pensez à l'ergonomie !

Votre projet utilisera impérativement le RFID (utilisé en général pour donner un accès – peut aussi se trouver sur un animal), la puce NFC, l'afficheur LCD ainsi que la caméra Picam.

Vous utiliserez un réseau privé (Peer-to-Peer) LoRa pour communiquer entre la carte Heltec-esp32s3-oled-LoRa et l'adaptateur LA66-LoRaWan connecté au RPi. La carte Heltec-esp32s3-oled-LoRa sera émetteur et l'adaptateur LA66-LoRaWan sera récepteur. Des messages pertinents seront affichés sur l'écran Oled. Acquisition de minimum une mesure avec la carte Heltec-esp32s3-oled-LoRa.

En ce qui concerne MQTT, vous devrez au moins une fois sécuriser les échanges à l'aide de certificats côté client et côté serveur. Vous devrez utiliser au moins une fois une QoS de 2 dans des échanges non-sécurisés par TLS. Utilisez Wireshark pour prouver que la QoS de 2 a été correctement implémentée. Vous utiliserez la librairie Python Paho.

En ce qui concerne Node-Red, vous devrez utiliser les dashboards et au moins une fois afficher des données sur une carte (noeud Worldmap). Utiliser un formulaire pour récupérer des données de l'utilisateur. Les dashboards, c'est pour visualiser, mais également pour interagir avec l'environnement (**fixer des seuils**,…) via un ou des formulaires Node-Red.

Sur votre RPi, il ne s'agit pas de prendre la main au démarrage du RPi, et de lancer indépendamment chaque petit programme qui gère les capteurs. Votre application est démarrée automatiquement au démarrage du RPi (utilisation d'un script systemctl). De même, les cartes ESP32 et Heltec-esp32s3-oled-LoRa démarreront automatiquement.

4. A RENDRE LE JOUR DE L'EXAMEN EN JANVIER

- Le matériel prêté (même si vous ne présentez pas l'examen en janvier, tout le matériel prêté devra être rentré le jour de l'examen).
- Un dossier de quelques pages **agrafées** reprenant (présentation et orthographe soignées svp):
  - Une page de garde reprenant au minimum : Nom+Prénom+Date+Nom de votre projet
  - Descriptif de votre projet : Description du problème - Description de votre solution – Pourquoi nous avons choisi ce projet (1 page A4)
  - Ce document-ci imprimé sur lequel vous aurez surligné en fluo toutes les contraintes que vous avez respectées. Rappel, comme ce sont les contraintes minimales, il doit y avoir bcp de fluo!
  - Le schéma en couleurs (1 page A4 ou A3) de l'architecture de votre solution reprenant les composants hardware et software utilisés. Attention, vous avez reçu un feedback concernant votre dossier. Il est essentiel de tenir compte des remarques qui ont été faites!
  - Un screenshot de vos dashboards
  - Un diagramme d'architecture de votre application (une page A4), mais cette fois reprenant la structure des fichiers, fonctions, méthodes, classes… Montrez que vous avez développé de manière structurée, modulaire, facile à maintenir.
  - Matériel nécessaire et estimation du coût de votre solution (coût du prototype – coût idéal du produit fini – coût mensuel de la solution – gestion de la consommation électrique de votre objet – vous ne pouvez pas demander à l'utilisateur de l'objet de recharger 3x/jour ou de changer les piles tous les 2 jours)
  - Estimation du marché potentiel de votre objet connecté.
  - Aspect sécurité : Relevé des actions prises pour sécuriser votre solution + utilisez https://sectigostore.com/blog/owasp-iot-top-10-iot-vulnerabilities/ et décrivez (Min 1 page et Max 2 pages) comment vous adressez le Top10 des IoT vulnerabilities. Soyez précis : il ne suffit pas par exemple de dire : "j'utilise des mots de passe complexes". Il convient d'expliquer comment vous pensez gérer les mots de passe (rappel : vous avez de nombreux clients).
  - Une conclusion : quel était votre niveau de compétence en IoT avant de démarrer ce projet ? quel est votre niveau de compétence à la fin du projet ? Quelles ont été les difficultés rencontrées lors de ce projet ?
- Une archive (.rar ou .zip ou .7z) reprenant votre contribution personnelle : 5 répertoires : un répertoire RPi avec le code que vous avez écrit sur le RPi (node-red + python + …), un répertoire ESP32 avec le code que vous avez utilisé sur votre ESP32, un répertoire Heltec-esp32s3-oled-LoRa avec le code que vous avez utilisé sur votre Heltec-esp32s3-oled-LoRa, un répertoire Ubuntu avec le code qui tourne sur vos VMs Ubuntu et un répertoire Dossier avec votre dossier au format pdf et également un fichier pdf séparé avec votre schéma d'architecture. Cette archive doit être prête avant l'examen sur votre ordinateur.

Attention : la démonstration que vous ferez le jour de l'examen doit être **fonctionnelle** et comprendre **au strict minimum toutes** les contraintes techniques minimales décrites ci-dessus.
Pensez également à un packaging (boîte en carton….) : c'est un prototype, mais il doit être présenté de manière correcte et adaptée à votre projet (il n'est pas acceptable d'avoir des capteurs reliés au RPi, à

l’ESP32 et à la carte Heltec-esp32s3-oled-LoRa sans aucun packaging). Par expl, si c’est un bracelet connecté, imaginez une manière d’attacher votre objet au poignet.
Les critères suivants seront également utilisés pour l’évaluation : originalité/ingéniosité du projet, pertinence du projet (would I buy it ?), utilité de l’objet, fiabilité de la solution, technologie(s) supplémentaire(s), niveau de difficulté + votre dossier.
L’ordre de passage vous sera envoyé par l’école virtuelle.

5. CONSIGNES POUR LA PARTIE EVALUATION ECRITE DU COURS

Le même jour que la présentation de votre projet, vous recevrez un questionnaire écrit portant sur la théorie (matière vue au cours : les principes, pas le code ni les commandes) et sur votre projet.