République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université de Yahia Farés - Médéa



FACULTÉ DES SCIENCES

DÉPARTEMENT DE MATHÉMATIQUES ET INFORMATIQUE

MÉMOIRE DE FIN D'ETUDES DE MASTER

Filière : Informatique

Spécialité : Ingénierie des Systèmes et Technologie du Web

Thème

Plateforme basée sur les micro-services et les jumeaux numériques pour la supervision et le contrôle des patients au niveau d'un hôpital

Proposé et Dirigé Par :

Dr. RAHIM Messaoud

Réalisé Par :

Maria Bouaied

Chibani Khadidja

Année Universitaire 2021 / 2022

Remerciements

Ce travail est l'aboutissement d'un dur labeur et de beaucoup de sacrifices pendant ces 5 ans. Nos remerciements vont d'abords au dieu tout puissant qui nous a doté la sante surtout après cette pandémie, le courage, la volonté et de nous avoir bénie jusqu'à la réalisation de ce travail.

Nos remerciements à notre encadrant **Mr *RAHIM Messaoud*** pour son aide, ses conseils précieux et critiques pertinentes qu'il nous a apportés durant la réalisation de ce mémoire.

Nos sincère gratitude va aussi à **Mr Redha Mazzari** qui a bien voulu répondre à nos différentes questions dans le but de permettre la facilite de l'élaboration de ce travail

Mes vifs remerciements vont aux membres de jury pour avoir accepté de juger notre présent travail

Nos derniers remerciements ce ne sont pas les moindres, vont à tous ceux qui ont contribué de près ou de loin pour l'aboutissement de ce travail.



Dédicace

Je dédie ce travail :

*A **Ma mère** mon exemple éternel qui a su m'enseigner à être quelqu'un avant d'avoir quelque chose, la flamme de mon cœur qui n'a épargné aucun effort pour me rendre heureuse et pour me voir réussir que dieu lui garde dans son vaste paradis. Vous êtes toujours dans mon cœur .*

*A **Mon Père** , l'homme de ma vie. Pour être toujours près de moi pour me soutenir, m'encourager et me guider avec vos conseils et vos effort jour et nuit pour mon bien être ,je vous aime PaPa. que dieu vous procure sante et longue vie.*

A mes abordables parents, que dieu les garde pour moi, pour leurs soutien et confiance.

*Aucune dédicace me serrant exprimer mon respect, mon amour éternel et ma consideration pour les sacrifices que vous avez consentie pour mon instruction et mon bienêtre, mon paradis qui m'aide encore aujourd'hui même si je ne suis plus un enfant. Je vous adore « **OUMI** » merci d'être ma mère.*

*A mes chers frères «**Nabila**» «**Maria**» «**Mohamed**» pour votre soutien moral, votre courage et votre présence à mes côté que dieu vous garde.*

*a mes petits enfants qui je l'aime «**Haroun**» «**Aya-Hiba**» «**Ishak**» «**Anfel**» «**Iyad**»*

*A mon Binome «**Maria**» pour entente et vos travailles dur dans ce travail.*

*A tous mes amis surtout **: Ferdouse,Manel,Ibtihadj ,Liela, Sihem,Hanan** je garderai de vous les bons souvenirs.*

A toutes celles et ceux qui m'ont soutenu tout au long de ma mémoire, et à tous ceux qui sont chères de mon cœur.

Khadidja



Dédicace

Je dédie ce travail :

*A **ma mère** mon exemple éternel qui a su m'enseigner à être quelqu'un avant d'avoir quelque chose, la flamme de mon cœur qui n'a épargné aucun effort pour me rendre heureuse et pour me voir réussir que dieu lui garde dans son vaste paradis. Vous êtes toujours dans mon cœur.*

*A **mon père**, l'homme de ma vie. Pour être toujours près de moi pour me soutenir, m'encourager et me guider avec vos conseils et vos effort jour et nuit pour mon bien être, je vous aime **papa**. que dieu vous procure sante et longue vie.*

A mes abordables parents, que dieu les garde pour moi, pour leurs soutien et confiance.

*Aucune dédicace me serrant exprimer mon respect, mon amour éternel et ma consideration pour les sacrifices que vous avez consentie pour mon instruction et mon bienêtre, mon paradis qui m'aide encore aujourd'hui même si je ne suis plus un enfant. Je vous adore «**OUMI** » merci d'être ma mère.*

A mes chers frères ahmed soumia hala zeyd

A ma chère binôme khadidja

Que ma fait un grand effort et un grand volante pour terminer ce petit travail

A mes amies manel et sihem

Maria

Résumé

Notre objectif dans ce travail est de développer une plateforme basée sur les micro-services et les jumeaux numériques pour la supervision et le contrôle des patients au niveau d'un hôpital. Pour implémenter cette plateforme. Nous utilisons les outils l ' (Eclipse Ditto) pour créer des jumeaux numériques, et (Apache Kafka) comme gestionnaire d'évènement.et l'outil Node Red pour connecter notre système physique et virtuelle.

La plateforme offre ses services à ces acteurs au moyen d'une application web que nous avons conçue et développée permettant de supporter toutes les étapes de la proposée.

Mots clé : Plateforme, Jumeau Numérique, Système Supervisé, IoT, Patient, Eclipse Ditto, Node-Red, Micro-service, Lit connecte, apache Kafka, Node-Red.

Abstact

Our objective in this work is to develop a platform based on micro-services and digital twins for the supervision and control of patients at the hospital level. for implementing this platform. We use l's tools (Eclipse Ditto) to create digital twins, and (Apache Kafka) as event .handler, and Node Red tool to connect our physical and virtual system

The platform offers its services to these actors by means of a web application that we ..have designed and developed to support all stages of the proposal.

Keywords: Platform, Digital Twin, Supervised System, IoT, Patient, Eclipse Ditto, Node-Red, Micro-service, lit connects, Apache Kafka, Node-Red.

ملخص

هدفنا في هذا العمل هو تطوير منصة قائمة على الخدمات الدقيقة والتوائم الرقمية للإشراف والسيطرة على المرضى على مستوى المستشفى. لتنفيذ هذه المنصة. نستخدم أدوات (Eclipse Ditto) l لإنشاء توائم رقمية ، و (Apache Kafka) كمعالج للأحداث ، وأداة Node Red لربط نظامنا المادي والافتراضي. تقدم المنصة خدماتها لهؤلاء الممثلين عن طريق تطبيق ويب صممناه وطورناه لدعم جميع مراحل الاقتراح.

الكلمات الرئيسية: النظام الأساسي ، التوأم الرقمي ، النظام الخاضع للإشراف ، IoT ، المريض ، Eclipse Ditto، Node-Red، Micro-service، Lit connects، apache Kafka، Node-Red.

Liste Des Abréviations

AMQP: Advanced Message Queuing Protocol.

API: Application Programming Interface.

BDD : Diagramme de définition de bloc.

CEI : Commision Eclectronique Internationale.

CPU : Central Processing Unit.

CRUD : Create, Read, Update, Delete.

CSS : Cascading Style Sheets.

HTML: Hyper Text Markup Language.

HTTP: Hyper Text Transfer Protocol.

IA : Intelligence Artificiel.

IIOT : Industrial IOT.

IOT : Internet Of Thing.

ISO : Organisation Internationale de Normalisation

JN : Jumeau Numérique

MQTT: Message Queuing Telemetry Transport**.**

PHP: Hypertext Preprocessor.

REST: Representational State Transfer.

RFID : Radio Frequency Identification.

SysML: Systems Modeling Language.

TCP : Transmission Control Protocol.

URI: Ressources uniformes.

URL: Uniform Ressource Locator.

WWW: World Wide Web.

Table des matières

INTRODUCTION GÉNÉRALE 1
Chapitre I : Jumeaux Numériques 4
I. Introduction 5
II. Les objets Connectés 5
II.1. Fonctionnement des objets connectés 5
II.2. Les avantages et l'utilité des objets connectés 6
III. Internet des objets 6
III.1. Fonctionnement d'internet des objets 6
III.2. Architecture d'IOT (internet of thing) 7
IV. Jumeaux numériques 8
IV.1. Les types des jumeaux numériques 8
IV.2. Domaines d'application des jumeaux numériques 9
IV.3. Avantages des jumeaux numériques 10
IV.4. Architecture de base d'un modèle jumeau numérique 10
V. Technologies et outils utilisées pour les jumeaux 12
V.1. Plateforme Eclipse Ditto 12
V.1.1. Architecture de Eclipse Ditto 12
V.1.2. Caractéristique principale d'Eclipse Ditto 13
V.1.3. Protocole de communication utilisées par Eclipse Ditto 13
V.2. Node-Red 17
V.2.1. Principale de la programmation Node-Red 18
V.2.2. Avantage de Node-Red 18
VI. Conclusion 19
Chapitre II : Micro-services et REST API 20
I. Introduction 21
II. Micro-services 21
II.1. Architecture de micro-services 21
II.2. Caractéristiques des micro-services 22
II.3. Fonctionnement de micro-services 22
III. REST API 23

III.1. API (Application Programming Interface) 23
III.2. REST (Representational State Transfer) 23
III.3. Fonctionnement de REST API 24
III.4. Architecture de REST API 24
IV. Apache Kafka pour la Communication entre micro-services 26
IV.1.1. Architecture de Kafka 26
IV.1.2. Fonctionnement de Kafka 27
IV.1.3. Avantages de l'approche de Kafka 28
IV.2. Bus de données 28
IV.2.1. Architecture de Bus 28
IV.2.2. Fonctionnement de Bus 29
Chapitre III : Conception 31
I. Introduction 32
II. Architecture et fonctionnement global de la plateforme 32
III. Proposition d'un modèle pour les lits l'hôpital 35
III.1. Diagramme de bloc pour l'hôpital 35
III.2. Diagramme de bloc d'un lit d'hôpital intelligent 36
IV. Jumeau numérique hospitalier 37
I. Thing 37
II. Features 38
V. Model du jumeau numérique d'un lits intelligent 38
VI. Application de contrôle et de supervision 44
VI.1. Diagramme de séquence 44
VI.1.1. Diagramme de séquence pour « Authentification » 44
VI.1.2. Diagramme de séquence pour « La création de lit » 45
VI.1.3. Diagramme de séquence pour « La création de modèle d'un lit » 46
VI.1.4. Diagramme de séquence pour supervision « jumeau numérique » 47
VI.1.5. Diagramme de séquence pour contrôle « jumeau numérique » 49
VI.1.6. Diagramme de séquence pour la gestion des patients. 50
Chapitre IV :Implémentation 52
I. Introduction 53
II. Implémentation 53
II.1. Plateforme des jumeaux numériques 53
II.2. Node Red 54
II.3. Kafka 54

III. Application web 56
III.1. Interface d'authentification 56
III.2. Interface d'accueil pour l'administrateur 57
III.3. Interface d'accueil pour les utilisateurs..... **Error! Bookmark not defined.**
III.4. Interface de gestion des lits d'hôpital intelligent 57
III.5. Interface de création de modèles d'un lit intelligent 58
III.6. Interface de création de lit intelligent..... 59
III.7. Interface de mise à jour d'un lit d'hôpital intelligent.... **Error! Bookmark not defined.**
III.8. Interface de gestion des utilisateurs ..... **Error! Bookmark not defined.**
III.9. Interface de gestion des patients « création, affectation »..... 60
III.10. Interfaces de recherche ..... **Error! Bookmark not defined.**
III.11. Interface de supervision et contrôle des lits intelligents..... 62
III.12. Interface de supervision et contrôle de patient dans le lit d'hôpital intelligent..**Error! Bookmark not defined.**
III.13. 63
IV. Conclusion..... 64
Conclusion Général..... 65

Liste de Figures

Chapitre I : Jumeaux numériques

Figure I.1: Architecture d'IOT.

Figure I.2: Processus de fabrication modèle de jumeau numérique.

Figure I.3: Architecteur de ditto**.**

Figure I.4: Les protocoles utilisés par la plateforme Eclipse Ditto.

Figure I.5: Le protocole http.

Figure I.6: Le protocole MQTT.

Figure I.7: Le protocole Web Socket.

Figure I.8: Le protocole AMQP.

Figure I.9: Éditeur Node-RED.

Chapitre II : Micro-service REST API

Figure II.1: Architecture de Micro-services**.**

Figure II.2: Architecture de REST API.

Figure II.3: Architecture de Apache Kafka.

Figure II.4: Architecture de Bus.

Chapitre III : Conception

Figure III.1: Architecture à base de micro-services pour la supervision des lits d'hôpital.

Figure III.2 : Diagramme de cas d'utilisation.

Figure III.3 : Modèle de blocs d'hospitalisation.

Figure III.4 : Modèle de blocs du lit intelligent.

Figure III.5 : Méta modèle des jumeaux numériques.

Figure III.6 : Modèle jumeau numérique d'un lit intelligent.

Figure III.7 : Exemple de format de JSON du jumeau numérique.

Figure III.8 : Diagramme de séquence pour « L'Authentification ».

Figure III.9 : Diagramme de séquence pour La création des Lits.

Figure III.10 : Diagramme de séquence pour « La création des modèles (lit) ».

Figure III.11 : Diagramme de séquence pour La supervision d'un lit intelligent « jumeau numérique ».

Figure III.12 : Diagramme de séquence pour La supervision médicale.

Figure III.13 : Diagramme de séquence pour contrôler les lit « jumeau numérique ».

Figure III.14 : Diagramme de séquence pour la gestion des patients.

Chapitre IV : Implémentation

Figure IV.1 : Interface principale de l'API Eclipse Ditto.

Figure IV.2 : Connexion avec les lits d'hôpital intelligents.

Figure IV.3: Le rôle de Broker dans la gestion des événements.

Figure IV.4 : Connexion entre Node-Red Kafka.

Figure IV.5 : Interface d'authentification.

Figure IV.6 : Interface d'accueil de l'application web.

Figure IV.7 : Interface de liste modèles.

Figure IV.8: Interface de création de modèles.

Figure IV.9 : Interface de création de lit d'hôpital intelligent.

Figure IV.10 : Interface de création des patients.

Figure IV.11 : Interface d'affectation des lits intelligent pour les patients.

Figure IV.12 : Interface de consultation.

Figure IV.13 : Interface de plan médical.

INTRODUCTION GÉNÉRALE

Introduction Générale

La technologie s'est développée et a fait de grands progrès dans plusieurs domaines, et l'informatique est devenue une partie de nos vies, qu'elles soient privées ou pratiques. Cette évolution a également conduit à l'émergence de nouvelles technologies et concepts tels que la technologie de l'Internet des objets et les jumeaux numériques.

Un jumeau numérique est un clone numérique et en temps réel d'un système physique. Il réplique le fonctionnement du système physique en collectant les données issues de ces composants et cela en utilisant l'Internet des objets.

La polyvalence des utilisations et des domaines d'application des jumeaux numériques et leurs avantages, nous a pousser à penser à leur application au domaine de la santé. Parmi les composants clés d'un hôpital on trouve les lits. Ces lits permettant d'offrir un service d'hospitalisation aux patients peuvent être connectés et ainsi offrir plus de possibilités de supervision et de contrôle concernant les patients. Lier ces lits avec les différents acteurs d'hospitalisation à travers la technologie des jumeaux numériques dans l'objectif d'améliorer les soins médicaux, le contrôle et la supervision des patients au niveau d'un hôpital grâce au système physique constitue l'objet de ce travail.

Dans ce travail, nous proposons de développer une plateforme basée sur les micro-services et les jumeaux numériques pour la supervision et le contrôle des patients au niveau d'un hôpital. Cette plateforme permettant de suivre des patients à travers des lits d'hôpital connectés et de faire communiquer entre les différents acteurs d'un hôpital, vise à améliorer le processus de suivi et d'hospitalisation des patients et faciliter les tâches de contrôle et de supervision aux praticiens de santé.

Le mémoire est organisé en quatre chapitres comme suit :

Dans le premier chapitre, nous présentons les concepts concernant l'internet des objets et jumeaux numériques en décrivant des outils permettant de mettre en œuvre des solutions basées sur ces technologies.

Dans le deuxième chapitre, nous présentons les concepts et les outils concernant les micro-services et les REST API et leurs avantages pour le développement d'applications.

Dans le troisième chapitre, la solution proposée pour le contrôle et la supervision des patients au niveau d'un hôpital sera détaillée. Dans ce chapitre, nous expliquerons les étapes concernant le développement du jumeau numérique pour un lit d'hôpital connecté, ensuite nous détaillerons la conception de la plateforme développée en focalisant sur les interactions entre les différents acteurs de la plateforme, le système physique et le jumeau numérique.

Le quatrième chapitre exposera l'implémentation de la solution proposée en décrivant les différents outils utilisés et les principales interfaces qui montrent le fonctionnement de la plateforme développés.

Chapitre I : Jumeaux Numériques

I. Introduction

La technologie a fait de grands progrès et a pu combiner le monde physique et le monde virtuel grâce à la technologie des jumeaux numériques.

Le jumeau numérique permet de modéliser virtuellement un produit ou un processus. Avec l'IoT (Internet Of Thing), cette copie virtuelle peut prendre vie grâce aux données en temps réel remontées par l'IoT.

Le but de ce chapitre est de donner un aperçu de cette nouvelle technologie. En discutant le concept de « jumeaux numériques », en expliquant les principes du fonctionnement et ses composantes avec des exemples entre autres : certains protocoles de communication, des plateformes destinées à leur développement, ainsi que certains domaines d'application.

II. Les objets Connectés

Les objets connectés sont des capteurs ou des objets dotés de capteurs capables de communiquer (envoyer et recevoir) des données à travers un réseau internet. Les données sont généralement envoyées à un ordinateur, une tablette, un smartphone ou tout autre appareil électronique et parfois via Internet pour que l'information soit accessible sur tous les appareils pouvant s'y connecter, Ces objets sont connectés soit directement par Wi-Fi utilisant le protocole TCP/IP ou par intermédiaire du smartphone ou via le Bluetooth ou grâce à un protocole de communication qui leur sont propres [1].

II.1. Fonctionnement des objets connectés

Les objets connectés se fonctionnent depuis d'un réseau tel qu'un wifi ou Bluetooth ou autres. Pour cela, ces objets sont munis de plusieurs capteurs qui peuvent recueillir des informations et d'une carte électronique qui sert à prétraiter les diverses données ainsi que d'un dispositif de communications et de transmission. On pourra obtenir des utilisations très avancées de ces objets selon les besoins ainsi que le niveau de complexité. Et quand les informations sont importantes ainsi que stratégiques, les objets connectés sont reliés à Internet par une plateforme de service en ligne qui est chargée d'informer et de rassembler ainsi que de mémoriser ces informations sur des serveurs. Pour cela, l'utilisateur de ces appareils pourra les consulter avec une application mobile en permanence. En ce moment, l'Internet des objets est associé de plus en plus avec l'intelligence artificielle à l'aide de leur maillage technologique en tenant du web. On pourra aussi les présenter comme une extension d'Internet [2].

II.2. Les avantages et l'utilité des objets connectés

Les objets connectés sont à la fois utilisés comme des émetteurs ainsi que des capteurs d'informations en émettant, mais aussi en interagissant avec autres objets connectés. Ils fonctionnent de façon autonome ce qui leur donne un début de compréhension. Les objets connectés peuvent offrir des services à tous types d'applications, tout en assurant le respect des exigences de la confidentialité ainsi que la sécurité. De plus, ces objets pourront être considérés comme un concept qui a des répercussions dans les technologies ainsi que dans la société.

Les objets connectés sont une sorte d'un réseau de réseaux permettant d'identifier directement sans avoir une ambiguïté des entités numériques, mais aussi des objets physiques par des systèmes d'identification électronique unifiés et normalisés et aussi par des appareils mobiles sans fil. Ils permettent aussi de pouvoir stocker et de récupérer ainsi que de traiter et de transférer les données s'y rattachant sans discontinuité entre les mondes physiques ainsi que virtuels. En ce moment, les objets connectés sont une source de nouveaux business dans le B2B comme iot business solutions [3].

III. Internet des objets

L'internet des objets, ou IoT écrit le réseau de terminaux physiques, qui permet de disposer de services évolués en interconnectant des objets (physiques ou virtuels) grâce aux technologies de l'information et de la communication interopérables existantes ou en évolution.

Tout appareil ou gadget doté d'un interrupteur de marche/arrêt et susceptible d'être connecté à Internet peut devenir un appareil connecté. La plupart des appareils connectés sont dotés d'une sorte de capteur qui leur permet de surveiller diverses conditions. Les appareils IoT peuvent également communiquer entre eux, de la même manière que les ordinateurs et les smartphones peuvent envoyer et recevoir des informations sans intervention humaine.

La définition de l'Internet des objets comprend également le gigantesque réseau qui relie des appareils, des personnes et même des animaux à travers le cloud [4].

III.1. Fonctionnement d'internet des objets

Chaque objet pilotable à distance détient sa propre carte d'identité qui le rend unique et reconnaissable, dans la plupart des cas une adresse IP. C'est ce numéro d'identification numérique qui va permettre de trouver cet objet et de lui donner des instructions à partir d'un ordinateur ou d'un téléphone portable.

Les instructions envoyées circulent jusqu'à l'objet en question en empruntant un canal de communication : Wi-Fi, Bluetooth, puce RFID (Radio Frequency Identification) ... [6].

III.2. Architecture d'iot

L'architecture IoT varie en fait considérablement en fonction de la mise en œuvre.

Les trois couches de l'architecture IoT [7].

- **La couche perception :** c'est la couche physique de l'architecture. C'est là que les capteurs et les appareils connectés entrent en jeu car ils collectent diverses quantités de données selon les besoins du projet. Il peut s'agir d'appareils périphériques, de capteurs et d'actionneurs qui interagissent avec leur environnement.
- **La couche réseau :** les données collectées par tous ces appareils doivent être transmises et traitées. C'est le travail de la couche réseau. Il connecte ces appareils à d'autres objets intelligents, serveurs et appareils réseau. Il gère également la transmission de toutes les données.
- **La couche application :** la couche d'application est ce avec quoi l'utilisateur interagit. C'est ce qui est chargé de fournir des services spécifiques à l'application à l'utilisateur. Cela peut être une implémentation de maison intelligente, par exemple, où les utilisateurs appuient sur un bouton dans l'application pour allumer une cafetière.

La figure I.1 représente l'architecture de l'IoT en trois couches principales [34] :

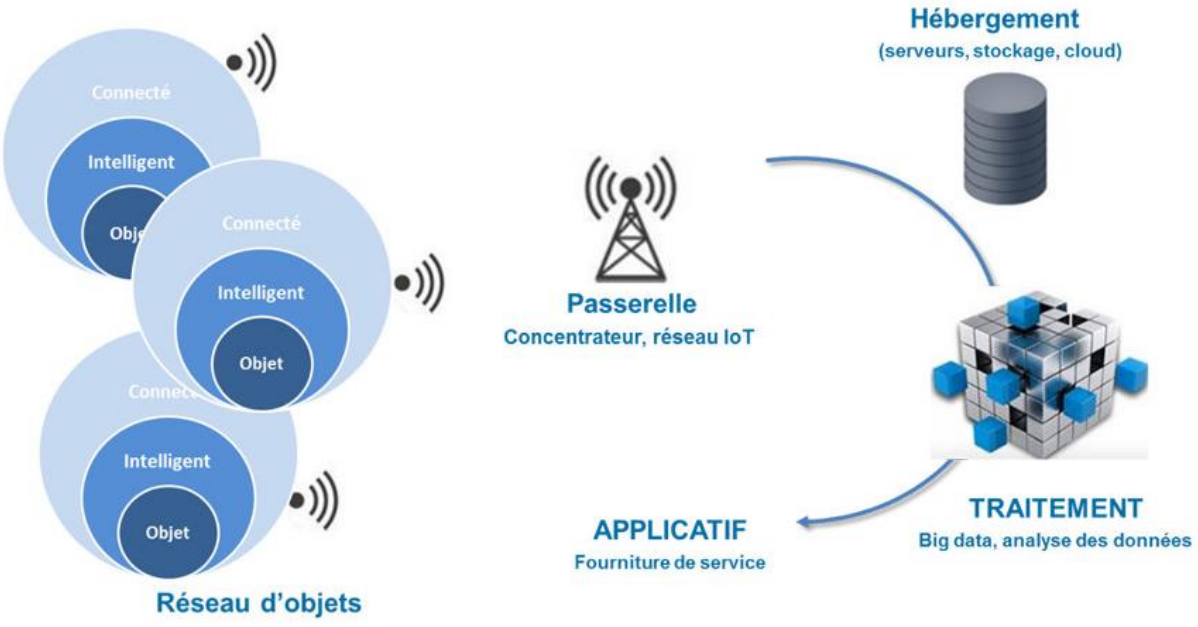


Figure I.1. Architecture d'IOT.

IV. Jumeaux numériques

Un jumeau numérique est une représentation virtuelle d'un objet ou d'un système qui couvre son cycle de vie. Il est mis à jour à partir de données en temps réel et utilise la simulation, l'apprentissage automatique et le raisonnement pour aider à la prise de décision [8].

Les jumeaux numériques sont des répliques virtuelles d'appareils physiques que les scientifiques des données et les professionnels de l'informatique peuvent utiliser pour exécuter des simulations avant que les appareils réels ne soient construits et déployés. Les jumeaux numériques peuvent également prendre des données IoT en temps réel et appliquer l'IA (Intelligence Artificiel) et l'analyse des données pour optimiser les performances.

Derrière ce terme, se cache un modèle logiciel dynamique d'un processus, produit ou service. Ce modèle peut être plus ou moins élaboré : dans certains cas, il s'agit d'une simple représentation, mais le jumeau numérique se traduit souvent en un modèle 3D plus sophistiqué. Celui-ci fonctionne via la collecte d'une combinaison de données. Des capteurs et une connexion IoT permettent de les récupérer pour représenter [9].

IV.1. Les types des jumeaux numériques

Les différents types de jumeaux numériques selon le niveau du rapport optique du produit [8].

- **Composants de jumeaux/parties de jumeaux :** Les composants de jumeaux sont l'unité de base du jumeau numérique, le plus petit exemple d'un composant fonctionnel. Les pièces de jumeaux sont à peu près la même chose, mais concernent des composants légèrement moins importants.
- **Jumeaux d'actifs :** Lorsque deux ou plusieurs composants fonctionnent ensemble, ils forment ce que l'on appelle un actif. Les jumeaux d'actifs vous permettent d'étudier l'interaction de ces composants, créant ainsi une multitude de données de performance qui peuvent être traitées puis transformées en informations exploitables.
- **Jumeaux système ou d'unité :** Le niveau suivant du rapport optique implique des jumeaux système ou d'unité, qui vous permettent de voir comment différents actifs se réunissent pour former un système fonctionnel complet. Les jumeaux système offrent une visibilité sur l'interaction des actifs et peuvent suggérer des améliorations de performances.

- **Jumeaux de traitement :** Les jumeaux de traitement, le niveau macro de rapport optique, révèlent comment les systèmes fonctionnent ensemble pour créer une installation de production entière. Ces systèmes sont-ils tous synchronisés pour fonctionner avec une efficacité maximale, ou les retards d'un système affecteront-ils les autres ? Les jumeaux de traitement peuvent aider à déterminer les schémas de synchronisation précis qui influencent finalement l'efficacité globale.

IV.2. Domaines d'application des jumeaux numériques

Les jumeaux numériques sont déjà largement utilisés dans différents secteurs pour simplifier, améliorer, automatiser et contrôler différents processus objets. Comme le suivantes applications [8].

- **Équipement de production d'électricité :** Les gros moteurs, y compris les moteurs à réaction, les moteurs de locomotive et les turbines de production d'électricité, bénéficient énormément de l'utilisation de jumeaux numériques, en particulier pour aider à établir des délais pour la maintenance régulièrement nécessaire.
- **Les structures et leurs systèmes :** Les grandes structures physiques, telles que les grands bâtiments ou les plates-formes de forage offshore, peuvent être améliorées grâce aux jumeaux numériques, en particulier lors de leur conception. Ils sont également utiles dans la conception des systèmes fonctionnant au sein de ces structures.
- **Opérations de fabrication :** Ils soient devenus omniprésents à toutes les étapes de la fabrication, guidant les produits de la conception au produit fini, et à toutes les étapes intermédiaires.
- **Services de santé :** les produits peuvent être profilés grâce à l'utilisation de jumeaux numériques, il en va de même pour les patients recevant des services de santé. Le même type de système de données générées par des capteurs peut être utilisé pour suivre une variété d'indicateurs de santé et générer des informations clés.
- **Secteur automobile :** Les voitures représentent de nombreux types de systèmes complexes qui fonctionnent ensemble, et les jumeaux numériques sont

largement utilisés dans la conception automobile, à la fois pour améliorer les performances des véhicules et augmenter l'efficacité de leur production.

- **Planification urbaine :** Les ingénieurs civils et autres personnes impliquées dans les activités de planification urbaine sont considérablement aidés par l'utilisation de jumeaux numériques, qui peuvent afficher des données spatiales 3D et 4D en temps réel et également intégrer des systèmes de réalité augmentée dans les environnements construits.

IV.3. Avantages des jumeaux numériques

Les jumeaux numériques présentent de nombreux avantages puisqu'ils améliorent le comportement des processus et des produits, ce qui se traduit généralement par une amélioration de leur efficacité [10].

- Les jumeaux numériques permettent d'anticiper les problèmes potentiels qui pourraient survenir dans le futur. Cela réduit les défauts du produit et raccourcit le temps de fabrication, entre autres.
- Amélioration et optimisation des processus de production grâce à des informations réelles.
- Réduisez les temps d'arrêt imprévus dus à des erreurs potentielles.
- Ils réduisent les accidents puisqu'ils permettent de simuler toutes sortes de situations et casuistiques.
- Réduction des coûts de maintenance en effectuant des tâches de maintenance préventive.
- Possibilités d'amélioration continue grâce à des simulations, identifiant les échecs et les inefficacités.

Le jumeau numérique est un excellent outil de l'industrie qui permet de tester virtuellement de nouvelles technologies pour simuler des comportements et réduire les risques. De plus, ils permettent de surveiller l'activité en temps réel et d'optimiser les comportements et les états qui se produisent. Nous pouvons effectuer l'évaluation des performances d'un processus ou d'un produit en temps réel avec peu de perte de productivité.

IV.4. Architecture de base d'un modèle jumeau numérique

L'architecture d'un jumeau numérique s'appuie sur la technologie de l'Internet des objets industriels (Industrial Internet of Things (IIOT)). Au moyen de capteurs, l'architecture relie les

actifs matériels (parfois appelés « objets physiques ») au monde virtuel, représenté par des solutions d'analyse et de stockage des données [11].

Monde physique

- **Capteurs** Détection d'événements et de changements chez un actif, dans son environnement immédiat, et conversion de ces occurrences physiques en impulsions électriques pouvant être interprétées à des fins utiles.

Liaison

- **Passerelle :** Déploiement de dispositifs qui ne sont pas connectés directement à Internet pour accéder à des systèmes de stockage de données.
- **Technologies et dispositifs de communication** : Activation du transfert de données entre les capteurs et les systèmes d'analyse et de stockage des données.

Monde numérique

- **Stockage de données :** Stockage et accumulation de données provenant des capteurs.
- **Capacités d'analyse des solutions de gestion des actifs de l'entreprise :** Analyse des données provenant des capteurs et acheminement de renseignements aux utilisateurs finaux afin d'appuyer leur prise de décision.

La figure I.2 représenté un modèle jumeau numérique [35] :

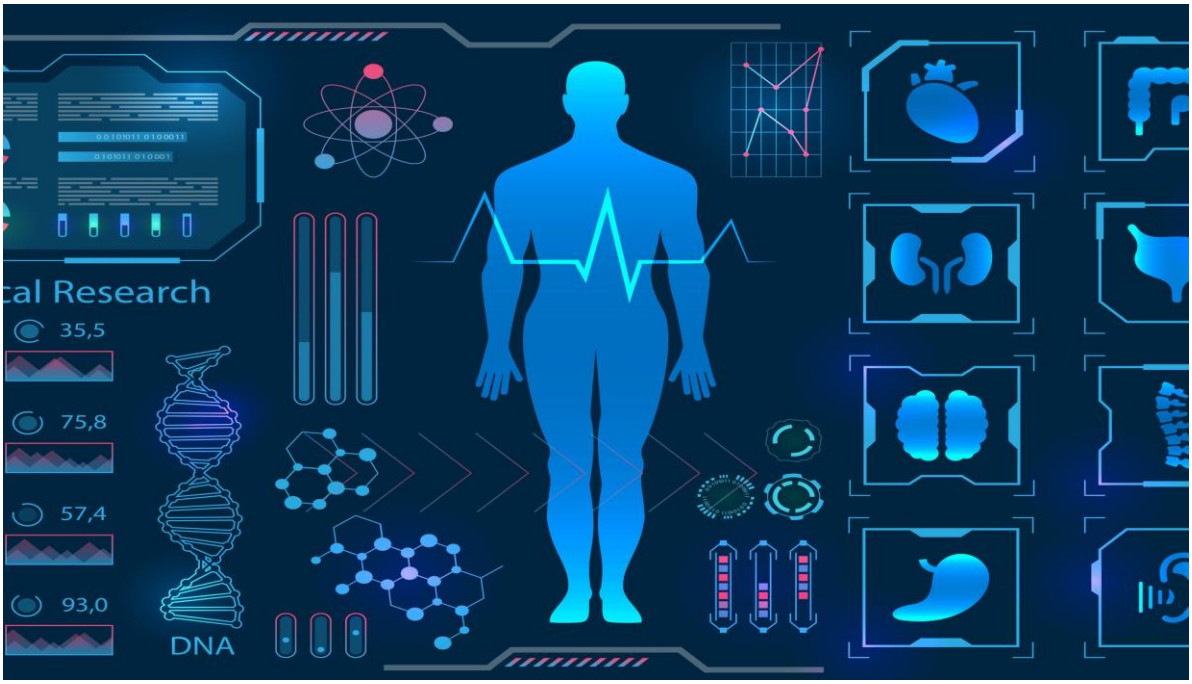


Figure I.2. Modèle de jumeau numérique.

V. Technologies et outils utilisées pour les jumeaux

V.1. Plateforme Eclipse Ditto

Eclipse DITTO est une technologie de l'IOT implémentant un modèle logiciel appelé « jumeaux numériques qui reflète potentiellement des millions et des milliards d'objet résidant dans le monde numérique jumelés avec des « choses » physiques. Cela simplifie le développement de solutions IOT pour les développeurs de logiciels car ils n'ont pas besoin de savoir comment ni où exactement les « choses » physiques sont connectées. Avec DITTO, un « thing » peut simplement être utilisée comme n'importe quel autre service Web via son jumeau numérique. [12]

V.1.1. Architecture de Eclipse Ditto

Les composants de ditto et leurs tâches :

Politiques : contrôle d'accès fin aux objets.

Objets : persistance des objets.

Recherche d'objets : suivi les modifications apportées aux objets, fonctionnalités, politiques et la mise à jour d'un index de recherche optimisé et l'exécution de requêtes sur cet index de recherche.

Concierge : orchestre et autorise les services de persistance de support.

Passerelle : fournit l'API http (Application Programming Interface, Hyper Text Transfer Protocol) Web Socket.

Connectivité : envoie des messages Ditto Protocol aux courtiers de messages externes et reçoit des messages d'eux [12].

La figure I.3 représente l'architecture de plateforme Eclipse Ditto déférent compassant et leur tâche [36] :

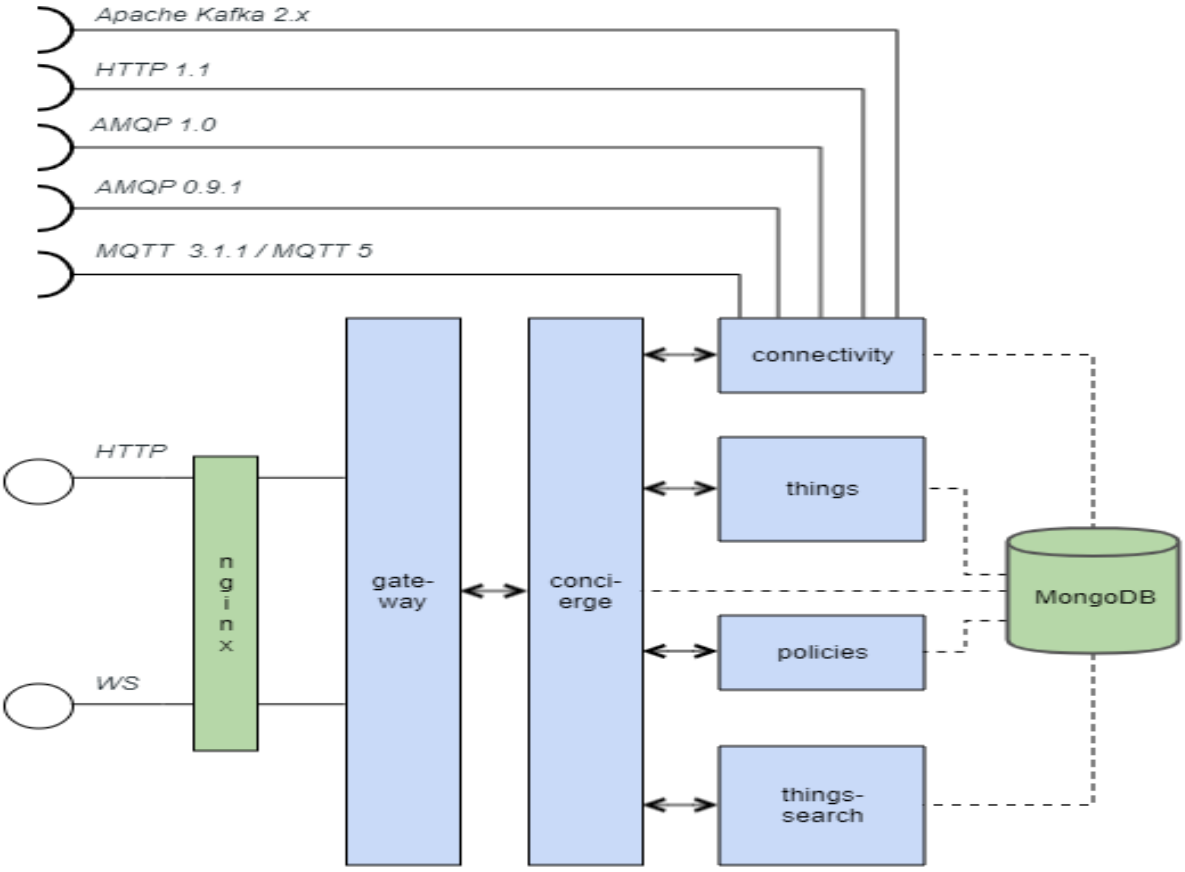


Figure.I.3. Architecteur de ditto**.**

V.1.2. Caractéristique principale d'Eclipse Ditto

Ditto se compose de plusieurs « micro-services », comme indiqué dans la vue des composants ci-dessus [12].

- A une API sous forme de signaux (commandes, réponses aux commandes, événements).
- Routage des demandes entre le matériel et applications client.
- Gestion des politiques de contrôle d'accès.
- La persistance du dernier état signalé du matériel comme Cache.
- Modifier la notification aux parties intéressées.

V.1.3. Protocole de communication utilisées par Eclipse Ditto

La plateforme Eclipse ditto utilise plusieurs protocoles, la figure suivante montre les Protocoles utilisés par la plateforme Eclipse ditto [12].

La figure I.4 représente les déférent protocole utilisé par Eclipse [37] :

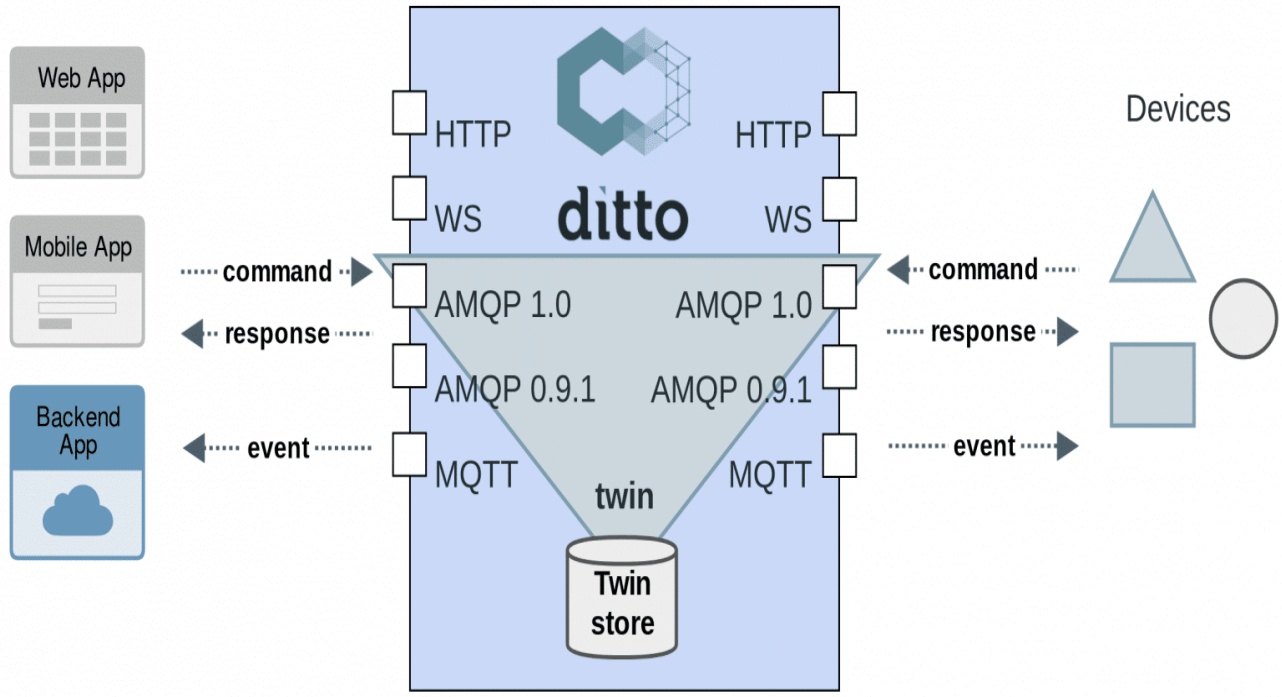


Figure.I.4. Les protocoles utilisés par la plateforme Eclipse Ditto.

Protocole http HTTP

Le protocole de transfert hypertexte est un protocole d'application pour les systèmes d'information distribués, collaboratifs et hypermédias qui permet aux utilisateurs de communiquer des données sur le World Wide Web [13].

WWW (World Wide Web) concerne la communication entre les clients Web et les serveurs La communication entre les ordinateurs clients et les serveurs Web se fait en envoyant des requêtes HTTP et en recevant des réponses http [12].

HTTP est l'ensemble de règles pour le transfert de fichiers - tels que du texte, des images, du son, de la vidéo et d'autres fichiers multimédias - sur le Web. Dès qu'un utilisateur ouvre son navigateur Web, il utilise indirectement HTTP. HTTP est un protocole d'application qui s'exécute au-dessus de la suite de protocoles TCP/IP, qui constitue la base d'Internet.

La figure I.5 montre comment le protocole http fonctionner et ce principe [38] :

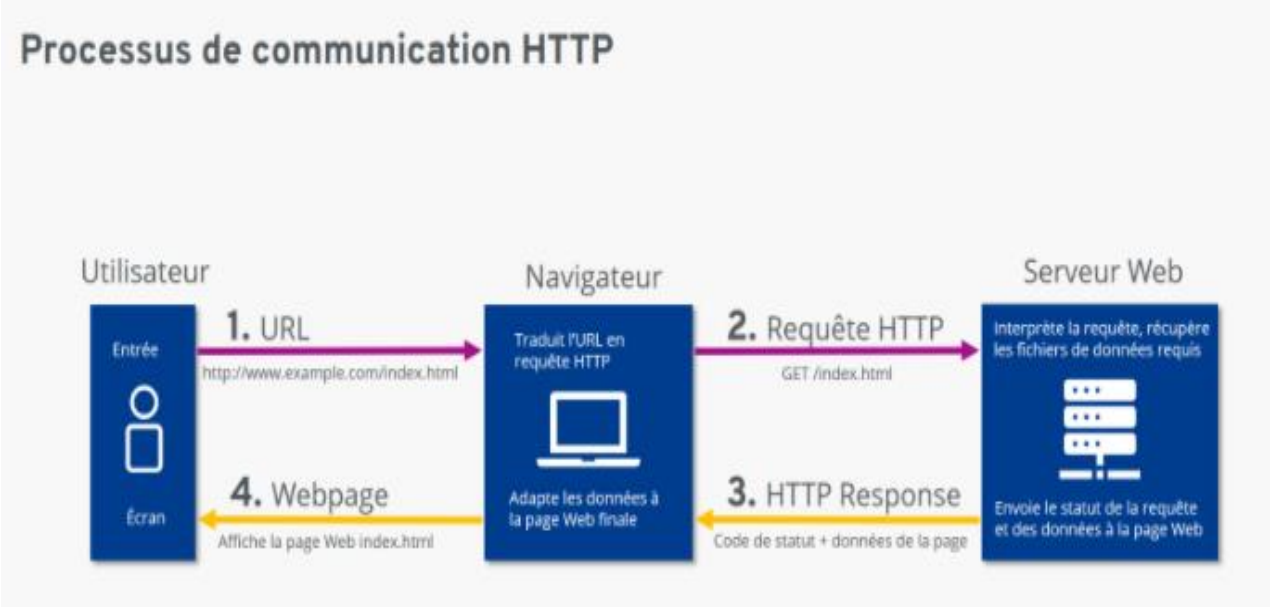


Figure I.5. Le protocole http.

MQTT (Message Queuing Telemetry Transport).

MQTT est un protocole de connectivité Internet des objets de machine à machine. Il s'agit d'un protocole de transport de messagerie extrêmement léger et de publication-abonnement. Ce protocole est utile pour la connexion avec l'emplacement distant où la bande passante est une prime. Ces caractéristiques le rendent utile dans diverses situations, y compris un environnement constant tel que pour les contextes de communication machine à machine et Internet des objets. Il s'agit d'un système de publication et d'abonnement où nous pouvons publier et recevoir les messages en tant que client. Il facilite la communication entre plusieurs appareils. Il s'agit d'un protocole de messagerie simple conçu pour les appareils contraints et avec une faible bande passante, c'est donc une solution parfaite pour les applications de l'internet des objets [14].

La figure I.6 montre comment le protocole MQTT fonctionner et ce principe :

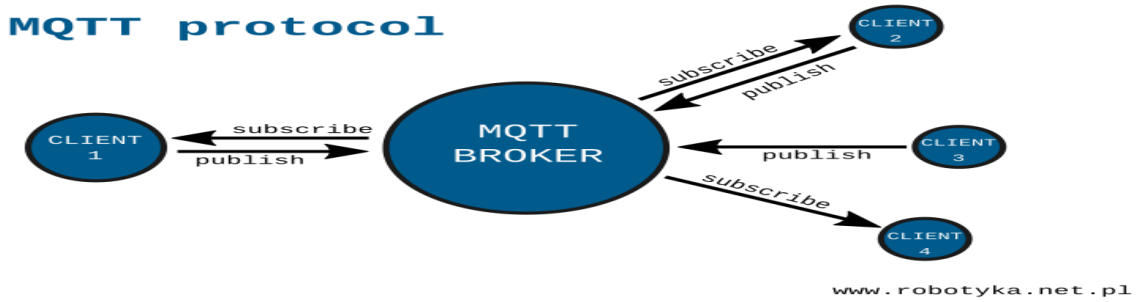


Figure I.6. Le protocole MQTT.

Le Websocket

Le protocole WebSocket est un protocole réseau basé sur TCP (Transmission Control Protocol). Il définit la manière dont les données sont échangées entre les réseaux. Parce qu'il est très fiable et efficace, il est utilisé par presque tous les clients. TCP établit la communication entre deux points de terminaison, appelés sockets. Cela permet aux données d'être transférées dans les deux sens.

Les connexions bidirectionnelles comme WebSocket (parfois écrit Web Socket) permettent d'échanger des données dans les deux sens simultanément. L'avantage est que les données peuvent être appelées plus rapidement. WebSocket permet notamment une communication directe entre une application web et un serveur WebSocket. Concrètement, cela signifie que vous pouvez appeler une page Web et la faire afficher en « temps réel » [15].

La figure I.7 montre comment le protocole WebSockets fonctionner et ce principe [40] :

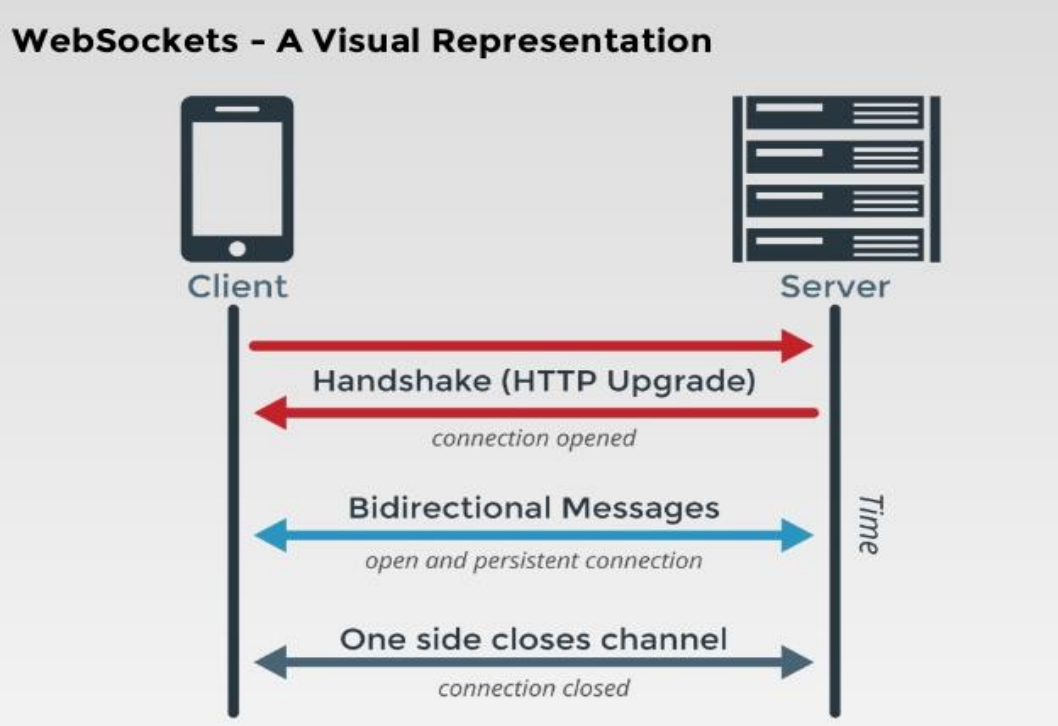


Figure I.7. Le protocol WebSocket.

AMQP (Advanced Message Queuing Protocol)

Le protocole AMQP est une norme ouverte permettant de transmettre des messages professionnels entre des applications ou des organisations. Il connecte les systèmes, alimente les processus métier avec les informations dont ils ont besoin et transmet de manière fiable les instructions qui permettent d'atteindre leurs objectifs [16].

AMQP est un protocole de couche d'application binaire standard ouvert conçu pour les intergiciels orientés message, c'est-à-dire que le protocole AMQP normalise la messagerie à l'aide de producteurs, de courtiers et de consommateurs et que la messagerie augmente le couplage lâche et l'évolutivité.

L'AMQP a été conçu pour permettre à un large éventail d'applications et de systèmes de fonctionner ensemble, indépendamment de leurs conceptions internes, normalisant la messagerie d'entreprise à l'échelle industrielle. Le protocole AMQP a été sélectionné par le groupe de normalisation de l'industrie OASIS1, avec l'intention de devenir une norme ISO/CEI (Organisation Internationale de Normalisation/Commision Eclectronique Internationale). Le protocole AMQP est presque un sur-ensemble complet, ne manquant que de la prise en charge explicite du protocole pour les messages Last-Value-Queues et Will. Cependant, sa conception délibérée pour l'extensibilité, utilisant une approche de type IANA avec une approche discursive, garantit que de telles fonctionnalités peuvent être ajoutées d'une manière compatible avec l'avenir et largement acceptée [17].

V.2. Node-Red

Node-RED est un outil puissant pour créer des applications Internet des objets en mettant l'accent sur la simplification du "câblage" des blocs de code pour effectuer des tâches. Il utilise une approche de programmation visuelle qui permet aux développeurs de connecter des blocs de code prédéfinis, appelés « nœuds », pour effectuer une tâche. Les nœuds connectés, généralement une combinaison de nœuds d'entrée, de nœuds de traitement et de nœuds de sortie, lorsqu'ils sont câblés ensemble, constituent un « flux » [18].

Node-RED est un outil visuel pour câbler l'Internet des objets et un outil de programmation pour câbler ensemble des périphériques matériels, des API et des services en ligne de manière nouvelle et intéressante.

Node-RED fournit un éditeur de flux basé sur un navigateur qui facilite la connexion de flux à l'aide de la large gamme de nœuds de la palette. Les flux peuvent être ensuite à l'exécution en un seul clic.

Les fonctions JavaScript peuvent être créées dans l'éditeur à l'aide d'un éditeur de texte enrichi.

Le runtime léger est construit sur Node.js, tirant pleinement parti de son modèle non bloquant piloté par les événements. Cela le rend idéal pour fonctionner à la périphérie du réseau sur du matériel à faible coût tel que le Raspberry Pi ainsi que dans le cloud [19].

L'éditeur se compose en quatre composants principaux :

- L'en-tête en haut, contenant le bouton de déploiement, le menu principal et, si l'authentification utilisateur est activée, le menu utilisateur.
- La palette de gauche, contenant les nœuds disponibles à utiliser.
- L'espace de travail principal au milieu, où les flux sont créés.
- La barre latérale à droite.

La figure I.9 représente l'interface principale d'éditeur Node-Red :

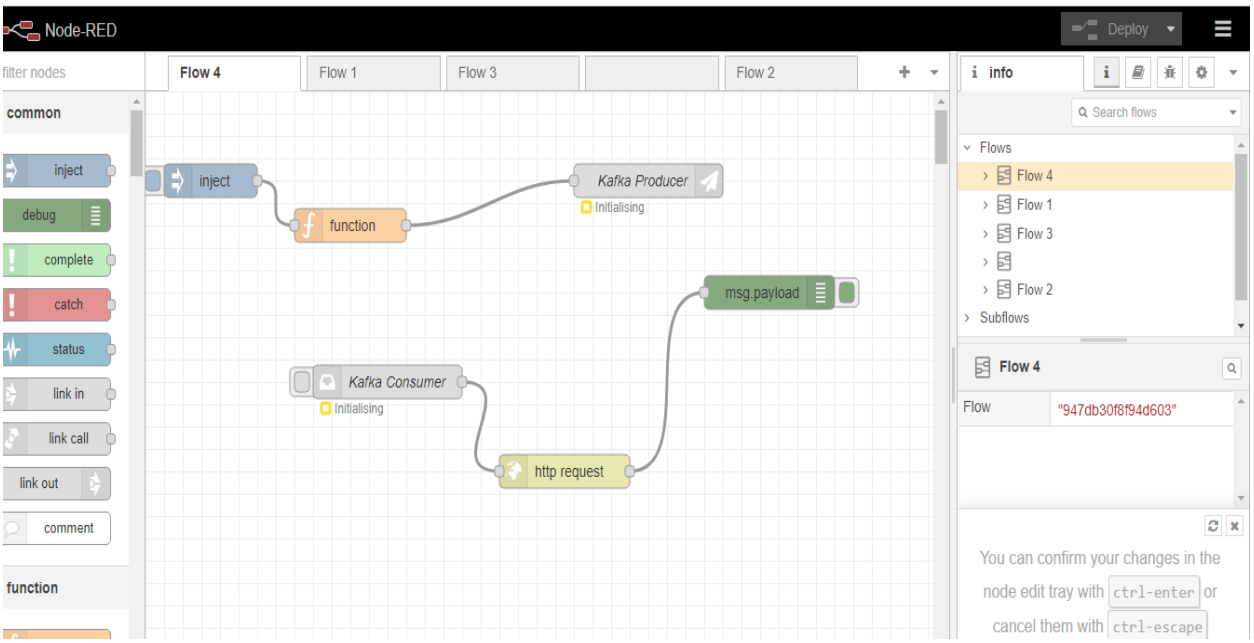


Figure I.9. Éditeur Node-RED.

V.2.1. Principale de la programmation Node-Red

Les principes de la programmation dans le Node-RED consiste à :

- Glisser-déplacer les nodes sur le flow et à les relier entre eux. Les données échangées sont des objets JSON et doivent posséder la propriété payload (charge utile)
- Le payload peut être un nombre, une chaine de caractères ou un objet JSON.
- Certains nodes peuvent avoir plusieurs sorties. Dans ce cas, le node retourne un tableau d'objets JSON dont le premier élément sort par la première sortie, le deuxième par la seconde.

V.2.2. Avantage de Node-Red

Node-RED a plusieurs avantages parmi lesquels on peut citer [20] :

- Il réduit le temps nécessaire pour créer une application fonctionnelle.
- Il est accessible à un large éventail de développeurs et de non-développeurs.
- La nature visuelle de l'interface le rend très intuitif.
- Il dispose d'une forte communauté et par conséquent il existe une panoplie de bibliothèques Node-RED.

VI. Conclusion

Dans ce chapitre, nous avons discuté les concepts de base des jumeaux numériques, son importance et son rôle dans la supervision, le contrôle et la prédiction du comportement des systèmes physiques. Notre domaine d'application s'est autour de la santé. Il s'agit de développer une plateforme basée sur les jumeaux numériques pour superviser et contrôler des patients à travers des lits d'hôpital connectés. Cette plateforme se base aussi sur les concepts micro-services et REST API qui feront l'objet du prochain chapitre.

Chapitre II : Micro-services et REST API

I. Introduction

La transformation numérique conduit souvent à briser les logiciels monolithes en micro-services exposés en tant qu'API REST (Representational State Transfer).

Les termes API RESTful et micro-services vont de pair lors de la création d'une application basée sur des micro-services. Ils se renvoient à des choses très différentes.

Le but de ce chapitre est de donner un aperçu sur les Micro-services et REST API. En discutant ces deux concepts, et en expliquant les principes de fonctionnement et l'architecture de chaque concept.

II. Micro-services

Les micro-services constituent une approche architecturale et organisationnelle du logiciel de développement, dans laquelle le logiciel compose de petits services indépendants qui communiquent via des API bien définies. Ces services sont détenus par de petites équipes auto-contenues.

Les architectures de micro-services simplifient la mise à l'échelle et accélèrent le développement des applications, pour des innovations et des délais de mise sur le marché optimisés [21].

II.1. Architecture de micro-services

Dans l'architecture monolithique, tous les composants fusionnent en un seul module. Mais, dans l'architecture des micro-services, ils sont répartis dans des modules individuels (micro-services) qui communiquent entre eux.

La communication entre les micro-services est une communication sans état où chaque paire de requête et de réponse est indépendante. Par conséquent, les micro-services peuvent

Communiquer sans effort. Dans l'architecture Micro-services, les données sont fédérées. Chaque micro-services à son magasin de données séparé [22].

La figure II.1 représente l‘architecture de Micro-services [41]:

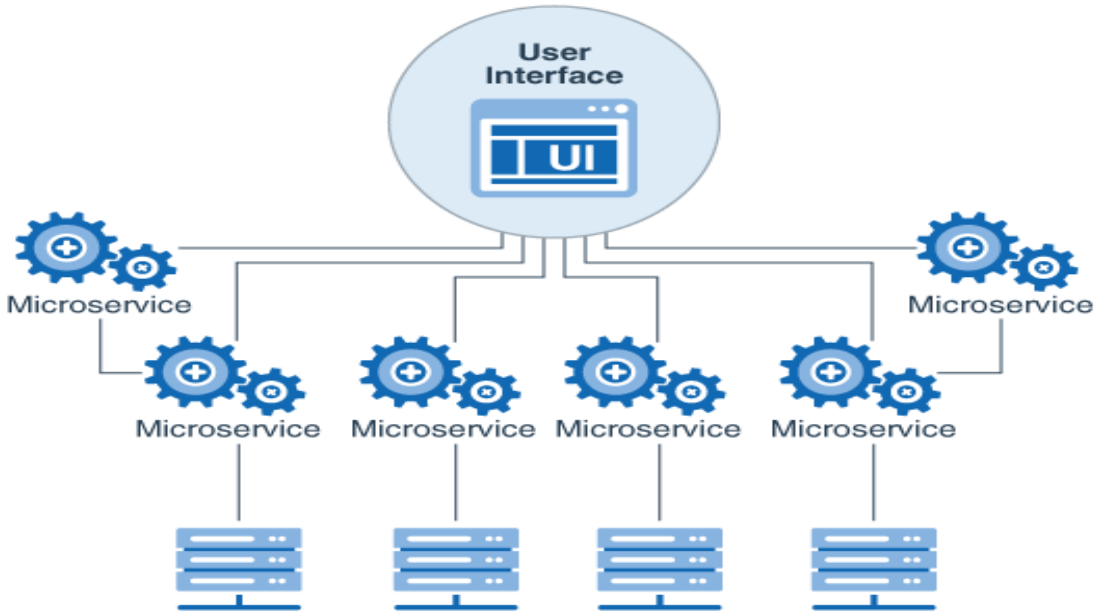


Figure II.1. Architecture de Micro-services**.**

II.2. Caractéristiques des micro-services

- **Autonomie :** tous les services des composants d'une architecture de micro-services peuvent être développés, déployés, gérés et mis à l'échelle sans affecter le fonctionnement des autres services. Les services n'ont pas besoin de partager leur code ou leur implémentation avec d'autres services. Les composants individuels communiquent par le biais d'API bien définies.
- **Spécialisation :** chaque service est conçu pour un ensemble de fonctionnalités et se concentre sur la résolution d'un problème spécifique [21].

II.3. Fonctionnement de micro-services

Bien qu'il n'existe pas de modèle ou de style universel, de nombreux micro-services partagent les caractéristiques suivantes et fonctionnent donc de la même manière :

- **Composants multiples :** chaque service individuel de cette structure peut être déployé, affiné et redéployé indépendamment sans compromettre l'ensemble de l'application.
- **Structuré autour d'unités commerciales :** sur le plan organisationnel, les micro-services s'articulent généralement autour des capacités et des priorités de l'entreprise. Chaque équipe s'approprie le fonctionnement d'un produit ou d'un service spécifique et intègre un mélange inter fonctionnel de compétences pour ce

faire. Cela contraste avec le développement monolithique, où les équipes se concentrent chacune sur une fonction pour l'ensemble de la suite d'applications.

- **Routage simple :** les micro-services utilisent les API comme un moyen léger de recevoir et de traiter les demandes, et d'émettre des réponses. Cela évite la complexité de l'utilisation des ESB, qui reposent souvent sur des systèmes avancés pour le routage de la messagerie et l'application de règles métier.
- **Décentralisation :** les micro-services intègrent une variété de technologies et de plates-formes, ce qui les rend mal adaptés à une gouvernance centralisée. Comme les développeurs de micro-services visent souvent à produire des outils utiles que d'autres peuvent utiliser pour résoudre des problèmes, ils considèrent généralement que la gouvernance décentralisée est favorable. Les micro-services favorisent également la gestion décentralisée des données, où chaque service gère sa propre base de données.[23]

III. REST API

Une API REST (également appelée API RESTful) est une interface de programmation d'application (API ou API web) qui respecte les contraintes du style d'architecture REST et permet d'interagir avec les services web RESTful.

III.1. API

Une API est un ensemble de définitions et de protocoles qui facilite la création et l'intégration de logiciels d'applications. Elle est parfois considérée comme un contrat entre un fournisseur d'informations et un utilisateur d'informations, qui permet de définir le contenu demandé au consommateur (l'appel) et le contenu demandé au producteur (la réponse).

Lorsque nous souhaitons interagir avec un ordinateur ou un système pour récupérer des informations ou exécuter une fonction, une API permet d'indiquer au système ce que vous attendez de lui, afin qu'il puisse comprendre notre demande et y répondre [24].

III.2. REST

REST est un ensemble de contraintes architecturales. Il ne s'agit ni d'un protocole, ni d'une norme. Les développeurs d'API peuvent mettre en œuvre REST de nombreuses manières.

Lorsqu'un client émet une requête par le biais d'une API RESTful, celle-ci transfère une représentation de l'état de la ressource au demandeur ou point de terminaison. Cette

information, ou représentation, est fournie via le protocole HTTP dans l'un des formats suivants : JSON (JavaScript Object Notation), HTML (Hyper Text Markup Language), XLT, Python, PHP (Hypertext Preprocessor) ou texte brut [24].

III.3. Fonctionnement de REST API

Les API REST communiquent via des requêtes HTTP pour exécuter des fonctions de base de données standard telles que la création, la lecture, la mise à jour et la suppression d'enregistrements (également appelés CRUD (Create, Read, Update, Delete) au sein d'une ressource. Par exemple, une API REST utiliserait une requête GET pour récupérer un enregistrement, une requête POST pour en créer un, une requête PUT pour mettre à jour un enregistrement et une requête DELETE pour en supprimer un. Toutes les méthodes HTTP peuvent être utilisées dans les appels d'API. Une API REST bien conçue est similaire à un site Web exécuté dans un navigateur Web avec une fonctionnalité HTTP intégrée.

L'état d'une ressource à un instant particulier, ou horodatage, est connu sous le nom de représentation de la ressource. Ces informations peuvent être fournies à un client dans pratiquement n'importe quel format, y compris JavaScript Object Notation (JSON), HTML, XLT, Python, PHP ou texte brut. JSON est populaire parce qu'il est lisible à la fois par les humains et les machines et qu'il s'agit d'une programmation indépendante du langage.

Les en-têtes et les paramètres de requête sont également importants dans les appels d'API REST, car ils incluent des informations d'identification importantes telles que les métadonnées, les autorisations, les identificateurs de ressources uniformes (URI), la mise en cache, les cookies, etc. Les en-têtes de requête et les en-têtes de réponse, ainsi que les codes d'état HTTP conventionnels, sont utilisés dans des API REST bien conçues [25].

III.4. Architecture de REST API

Il existe des règles de conception qui sont appliquées pour établir les différentes caractéristiques du style architectural REST, appelées contraintes REST [26].

Interface uniforme la conception fondamentale de tout système RESTful, qui fait la différence entre une API REST et une API non REST. Cette contrainte simplifie l'architecture en fournissant une manière uniforme d'interagir avec un serveur donné (quel que soit le type d'équipement ou d'application).

Serveur client cette contrainte indique essentiellement que les serveurs et les clients ne doivent connaître que les URL (Uniform Ressource Locator) des ressources, et c'est tout. Il

n'y a pas de dépendances. Tout peut être développé et remplacé indépendamment tant que l'interface entre eux n'est pas modifiée.

Apatridie cette contrainte, par définition, signifie que toutes les interactions client-serveur sont sans état. Chaque demande est traitée comme nouvelle, sans sessions ni historique. Si l'utilisateur final a besoin d'une application avec état, chaque demande du client doit alors contenir toutes les informations nécessaires pour traiter la demande.

Cacheabilité la mise en cache dans REST doit être appliquée aux ressources, le cas échéant. Cela peut être implémenté côté serveur ou côté client.

Système en couches REST permet d'utiliser une architecture système en couches, où les API sont peut-être déployées sur un serveur, les données sont stockées sur un autre et les demandes sont authentifiées sur un autre encore.

Code à la demande cette contrainte facultative signifie que vous pouvez, si nécessaire, fournir du code exécutable au client.

La figure II.2 représente l'architecture de REST API [42] :

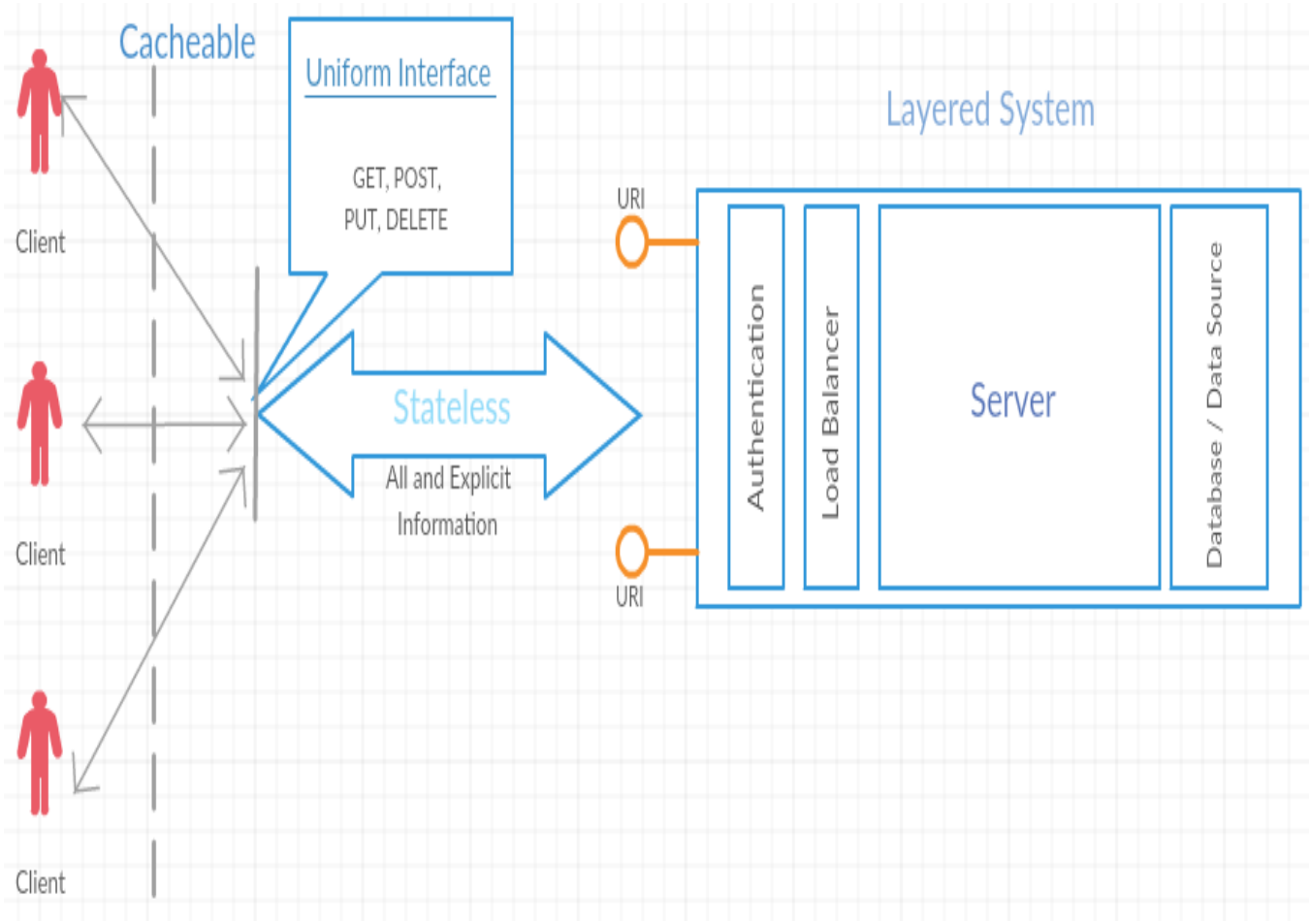


Figure II.2. Architecture de REST API.

IV. Apache Kafka pour la Communication entre micro-services

Apache Kafka est un magasin de données distribué optimisé pour l'ingestion et le traitement de données en continu en temps réel. Les données en continu sont des données générées en continu par des milliers de sources de données, qui envoient généralement les enregistrements de données simultanément. Une plate-forme de streaming doit gérer cet afflux constant de données et traiter les données de manière séquentielle et incrémentielle [27].

Apache Kafka est l'outil le plus populaire pour les micro-services, car il résout de nombreux problèmes d'orchestration des micro-services tout en activant les attributs que les micro-services visent à atteindre, tels que l'évolutivité, l'efficacité et la vitesse. Il facilite également la communication micro-services tout en préservant une latence ultra-faible et une tolérance aux pannes [28].

IV.1.1. Architecture de Kafka

Kafka remédie aux deux modèles différents en publiant des enregistrements sur différents sujets. Chaque sujet a un journal partitionné, qui est un journal de validation structuré qui garde une trace de tous les enregistrements dans l'ordre et en ajoute de nouveaux en temps réel. Ces partitions sont distribuées et répliquées sur plusieurs serveurs, ce qui permet une évolutivité, une tolérance aux pannes et un parallélisme élevé. Chaque consommateur se voit attribuer une Partition dans le sujet, ce qui permet plusieurs abonnés tout en maintenant l'ordre des données. En combinant ces modèles de messagerie, Kafka offre les avantages des deux. Kafka agit également comme un système de stockage très évolutif et tolérant aux pannes en écrivant et en répliquant toutes les données sur disque. Par défaut, Kafka conserve les données stockées sur le disque jusqu'à ce qu'il manque d'espace, mais l'utilisateur peut également définir une limite de conservation. Kafka à quatre API [27] :

- **API Producer :** utilisée pour publier un flux d'enregistrements dans un sujet Kafka.
- **API consommateur :** utilisée pour s'abonner à des sujets et traiter leurs flux d'enregistrements.
- **API Stream :** permet aux applications de se comporter comme des processeurs de flux, qui prennent un flux d'entrée à partir d'un ou plusieurs sujets et le transforment en un flux de sortie qui va dans différents sujets de sortie.

- **API de connecteur :** permet aux utilisateurs d'automatiser de manière transparente l'ajout d'une autre application ou d'un système de données à leurs sujets Kafka actuels.

La figure I.3 représente l'architecture et le principe d'Apache Kafka [43] :

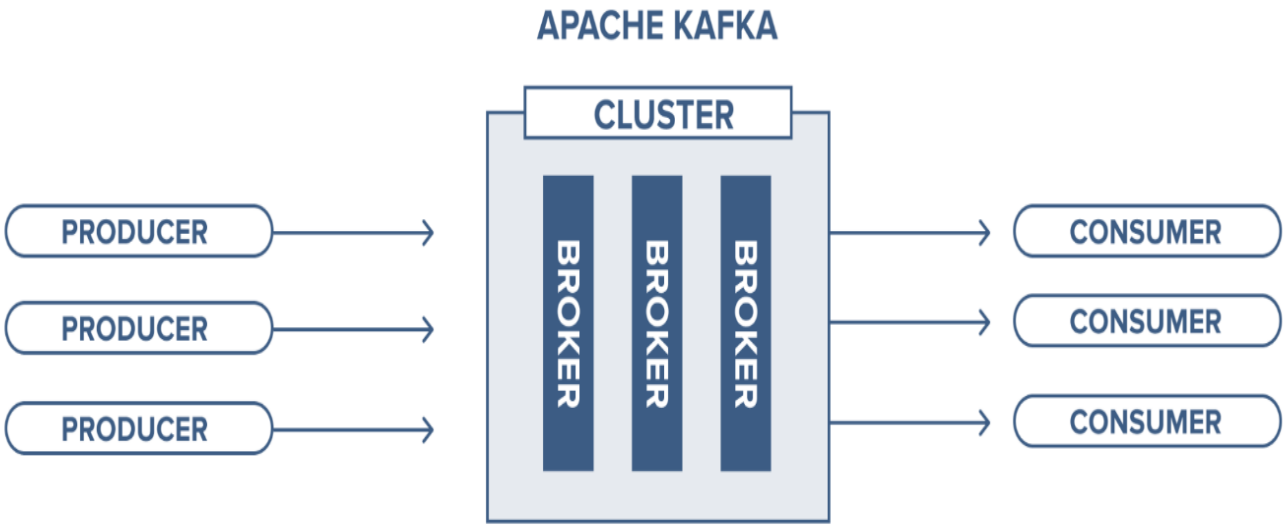


Figure II.3. Architecture de Apache Kafka**.**

IV.1.2. Fonctionnement de Kafka

Kafka combine deux modèles de messagerie, la mise en file d'attente et la publication-abonnement, pour fournir les principaux avantages de chacun aux consommateurs. La mise en file d'attente permet de répartir le traitement des données sur de nombreuses instances de consommateur, ce qui le rend hautement évolutif. Cependant, les files d'attente traditionnelles ne sont pas multi-abonnés. L'approche de publication-abonnement est multi-abonnés, mais comme chaque message est envoyé à chaque abonné, il ne peut pas être utilisé pour répartir le travail sur plusieurs processus de travail. Kafka utilise un modèle de journal partitionné pour assembler ces deux solutions. Un journal est une séquence ordonnée d'enregistrements, et ces journaux sont divisés en segments, ou partitions, qui correspondent à différents abonnés. Cela signifie qu'il peut y avoir plusieurs abonnés au même sujet et chacun se voit attribuer une partition pour permettre une plus grande évolutivité. Enfin, le modèle de Kafka offre une re jouabilité, ce qui permet à plusieurs applications indépendantes lisant des flux de données de fonctionner indépendamment à leur propre rythme [27].

IV.1.3. Avantages de l'approche de Kafka

- **Évolutif :** Le modèle de journal partitionné de Kafka permet aux données d'être distribuées sur plusieurs serveurs, ce qui les rend évolutives au-delà de ce qui pourrait tenir sur un seul serveur.
- **Rapide :** Kafka découple les flux de données afin d'obtenir une latence très faible, ce qui le rend extrêmement rapide.
- **Durable :** Les partitions sont distribuées et répliquées sur de nombreux serveurs, et les données sont toutes écrites sur le disque. Cela aide à protéger contre les pannes de serveur, ce qui rend les données très résistantes aux pannes et durables [27].

IV.2. Bus de données

Un bus est un sous-système utilisé pour connecter des composants informatiques et transférer des données entre eux. Par exemple, un bus interne relie les composants internes de l'ordinateur à la carte mère. Une « topologie de bus » ou une conception peut également être utilisée d'autres manières pour décrire les connexions numériques [29].

Le bus d'événements peut être conçu comme une interface avec l'API nécessaire pour s'abonner et se désabonner aux événements et pour publier des événements. A chaque action, le micro-services met à jour une entité métier et publie un événement qui déclenche l'action suivante [30].

IV.2.1. Architecture de Bus

Le bus système combine les fonctions des trois bus principaux, à savoir le bus de contrôle, le bus d'adresse et le bus de données. Le bus de commande transporte les signaux de commande, de synchronisation et de coordination pour gérer les diverses fonctions à travers le système.

Le bus d'adresse est utilisé pour spécifier les emplacements de mémoire pour les données transférées. Le bus de données, qui est un chemin bidirectionnel. Il transporte les données réelles entre le processeur (CPU), la mémoire et les périphériques (Entrée et Sortie).

L'architecture du bus système varie d'un système à l'autre et peut être spécifique à une conception d'ordinateur particulière. Les autres caractéristiques communes des bus système sont basées sur le rôle principal, la connexion de périphériques en interne ou en externe, etc. [31]

Bus interne : Il est également appelé bus de données interne, bus mémoire, bus système ou bus frontal. Il connecte tous les composants internes d'un ordinateur, tels que le processeur et la mémoire, à la carte mère. Les bus de données internes sont également appelés bus locaux, car ils sont destinés à se connecter à des périphériques locaux. Ce bus est rapide et indépendant du reste des opérations informatiques.

Bus externe : Il est également connu sous le nom de bus d'extension. Il est composé des voies électroniques qui relient les différents périphériques externes, tels que l'imprimante, etc.

La figure II.4 représente l'architecture de buse de donnée :

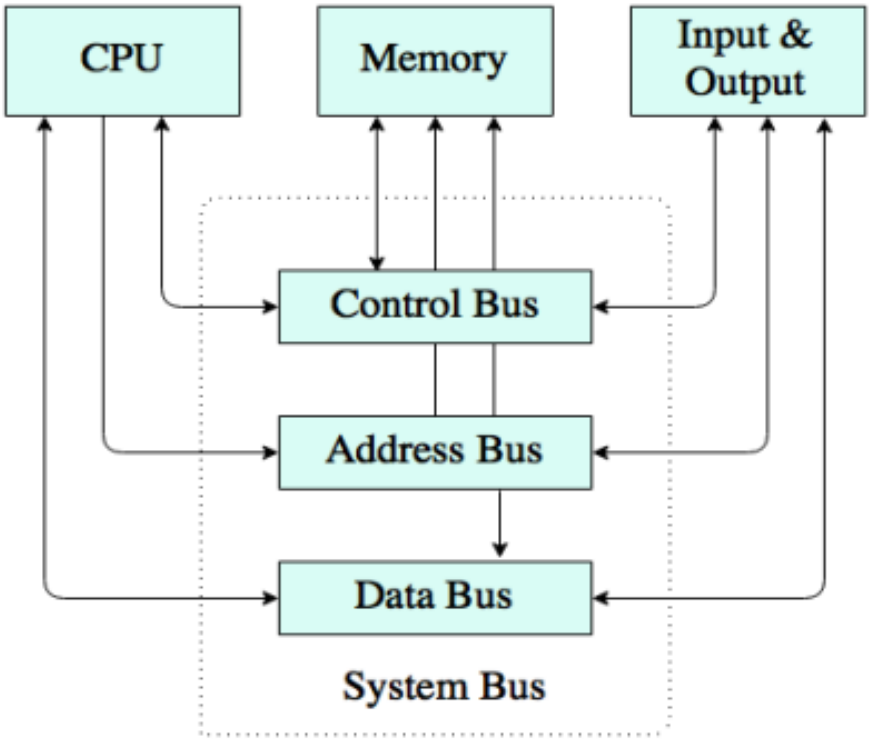


Figure II.4. Architecture de Bus**.**

IV.2.2. Fonctionnement de Bus

Un bus transfère des signaux électriques d'un endroit à un autre. Un bus réel apparaît comme une quantité infinie de circuits en cuivre gravés sur la surface de la carte mère. Le bus est connecté à la CPU (Central Processing Unit) via l'unité d'interface de bus.

Les données circulent entre le processeur et la mémoire via le bus de données. L'emplacement (adresse) de ces données est transporté le long du bus d'adresse. Un signal d'horloge qui maintient tout en synchronisation voyage le long du bus de contrôle.[32]

V. Conclusion

Dans ce chapitre nous avons discuté les concepts de base des micro-services, REST API, ces architectes et le principe de fonctionnement et ces outils de communication.

Chapitre III : Conception

I. Introduction

Les soins médicaux aux les patients hospitalisés peuvent être améliorés grâce à la technologie des jumeaux numériques en concevant un modèle de lit d'hôpital et son jumeau numérique.

Ensuite, nous développons une application basée sur des jumeaux numériques pour gérer et superviser les patients via des lits d'hôpitaux intelligents.

L'utilisation de jumeau numérique pour les lits d'hôpitaux permet d'améliorer la supervision de processus de suivi des patients.

II. Architecture et fonctionnement global de la plateforme

La solution proposée est de concevoir un modèle basé sur des jumeaux numériques pour des lits intelligents dans les hôpitaux pour le suivi et la surveillance médicale des patients.

Le lit intelligent se compose de capteurs qui permettent de capturer toutes les données possibles du patient et du lit et d'actionneurs pour contrôler les caractéristiques du lit et du patient et connecter d'autres dispositifs médicaux en fonction de la situation de patient.

La solution basée sur le jumeau numérique d'un lit d'hôpital se présente sous la forme d'un ensemble d'objets (things) avec leurs caractéristiques et leurs propriétés. Le système physique est alors connecté au jumeau numérique à travers les technologies des objets connectés (IOT). Le jumeau numérique permet de représenter toutes les caractéristiques et les propriétés du lit, de capter toutes les modifications du système physique à travers les capteurs et commander le système physique via les actionneurs.

Enfin, un utilisateur jumeau numérique accompagne une application web pour créer et contrôlé les objets (things) du jumeau numérique. La création se fait au niveau d'une plateforme open source (Eclipseditto) implémentant les jumeaux numériques [33]. La communication, l'envoi et la réception des données du système physique se fait avec d'outil Node-RED.[19]

La figure III.1 représente l'architecture proposé qui basé sur micro-services pour la supervision des lits d'hôpital :

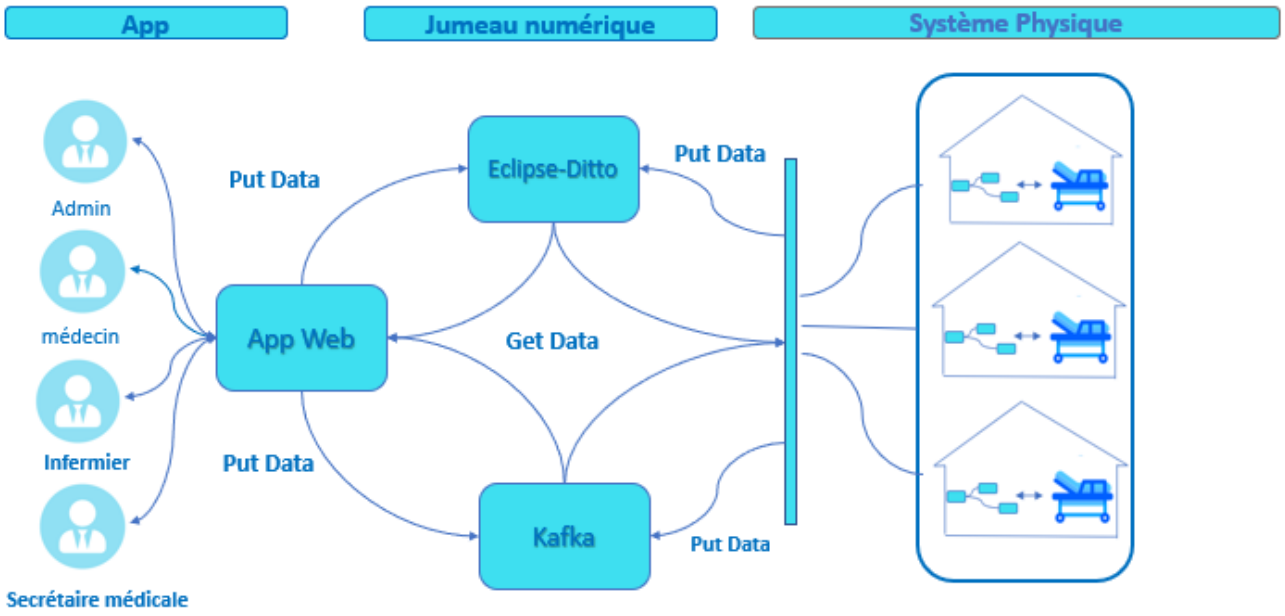


Figure III.1. Architecture à base de micro-services pour la supervision des lits d'hôpital.

Pour illustrer les fonctions du système, Nous avons généralement décrit le comportement attendu du système : ce qui se passera et quand.

Dans notre système, nous avons quatre acteurs qui sont :

- L'administrateur qui est le responsable de la gestion des lits (la création et la suppression) des jumeaux numériques et ces traitements.
- Le médecin qui fait le diagnostic, supervision, contrôle médicale.
- L'infirmière est responsable du suivi de l'évolution de la santé du patient et la supervision matérielles des lits et assurer la supervision matérielle et logicielle.
- La secrétaire médicale qui est responsable de la gestion des patients (création, affectation, libération) et le mise à jour d'état de lit.

La figure III.2 représente le diagramme de c'cas d'utilisation :

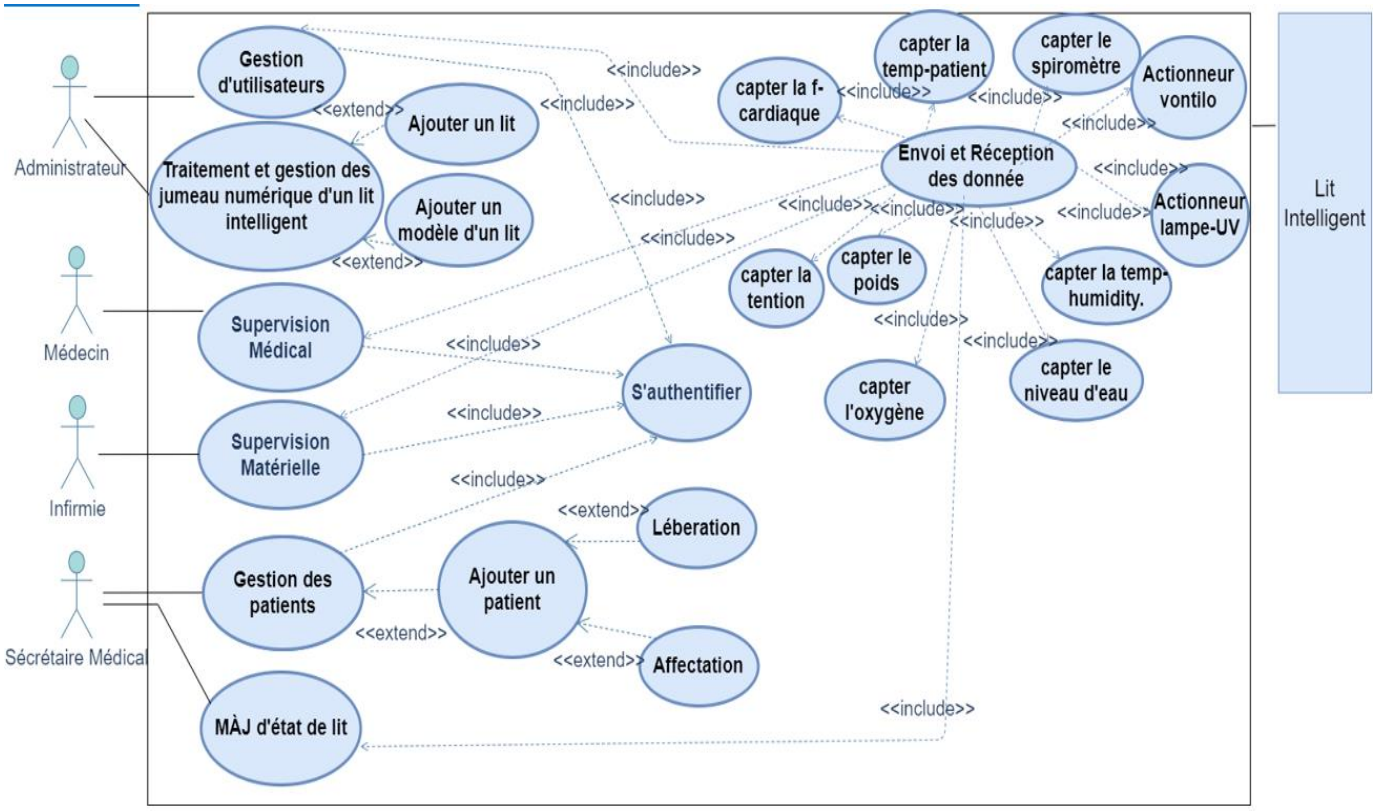


Figure III.2 Diagramme de cas d'utilisation.

Scénario :

- Les cas d'utilisation « création/suppression des lits d'hôpital intelligents » ça concerne l'administrateur. Cela nécessite de connecter à la plateforme de jumeaux numériques.
- Le cas d'utilisation « gestion des patients » et leur affectation concerne la secrétaire médicale.
- Le cas d'utilisation « supervision médical et prélèvement des données » est assuré par le médecin, le lit d'hôpital doit être fonctionnel et ses capteurs proprement positionnés sur le patient. Les données du patient sont prélevées par chacun des capteur et envoyées au jumeau numérique.
- Le cas d'utilisation « contrôle du lit d'hôpital intelligent » décrit le cas de résolution, par l'infirmier, d'un problème matériel (défaillance d'un composant physique, batterie épuisée, etc.) ou logiciel (valeurs erronées), ainsi l'état du patient occupant le lit.

III. Proposition d'un modèle pour les lits d'hôpital

Pour modéliser le system hospitalier et leurs lits intelligents, nous avons choisi le langage de modélisation SysML (Systems Modeling Language) qui contient plusieurs diagrammes, parmi ces diagrammes nous utiliserons le diagramme BDD (diagramme de définition de bloc).

Le diagramme de définition de bloc est un diagramme qui décrit la structure d'un système, d'un sous-système ou d'un composant. Un bloc est comme une classe, il peut être composé d'autres blocs et peut avoir des relations avec d'autres éléments du modèle (blocs, diagramme, activités, …). Un bloc est représenté graphiquement par un rectangle comportant plusieurs compartiments qui contient les détails sur le composant (élément) présenté par le bloc.

III.1.Diagramme de bloc pour l'hôpital

Chaque bloc du diagramme représente un élément de l'hôpital. Le bloque principal est « l'hôpital », il se compose de services, de blocs, de chambres, et de lits intelligents et de patients. La figure III.3 représente le modèle de blocs d'hospitalisation :

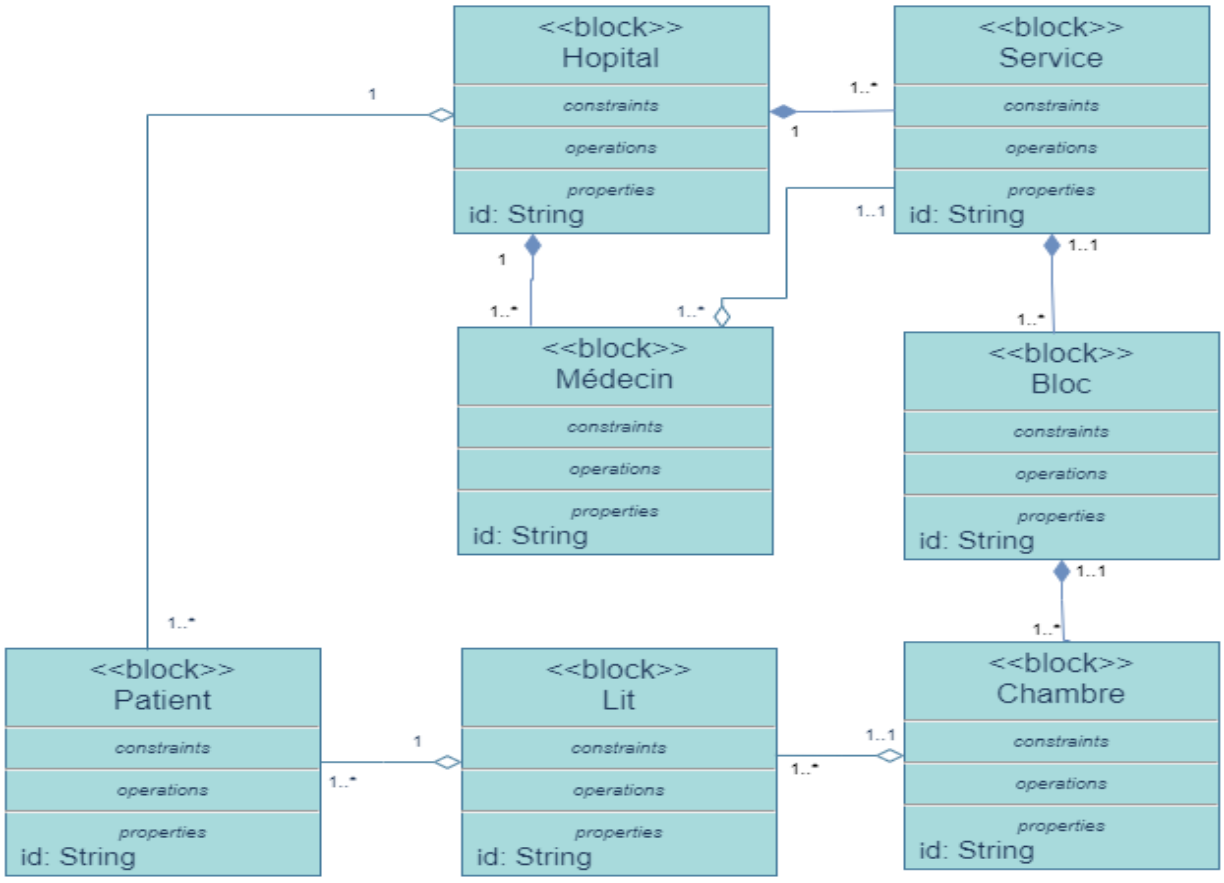


Figure III.3. Modèle de blocs d'hospitalisation.

III.2.Diagramme de bloc d'un lit d'hôpital intelligent

Chaque bloc du diagramme représente un élément du lit intelligent. Le bloque principal est « lit », il se compose de capteurs, d'actionneurs, d'un écran tactile, d'une boite alimentation et d'une carte de communication, carte Bluetooth.. . Les capteurs et les actionneurs peuvent être de différents types. Un lit peut être affecté à un patient, Le lit peut être attaché à un équipement médical. Les capteurs sont utilisés pour capturer des valeurs concernant l'état du lit ou du patient. Les actionneurs sont utilisés pour commander le lit et ses composants.

La figure III.4 représente le modèle de blocs du lit intelligent :

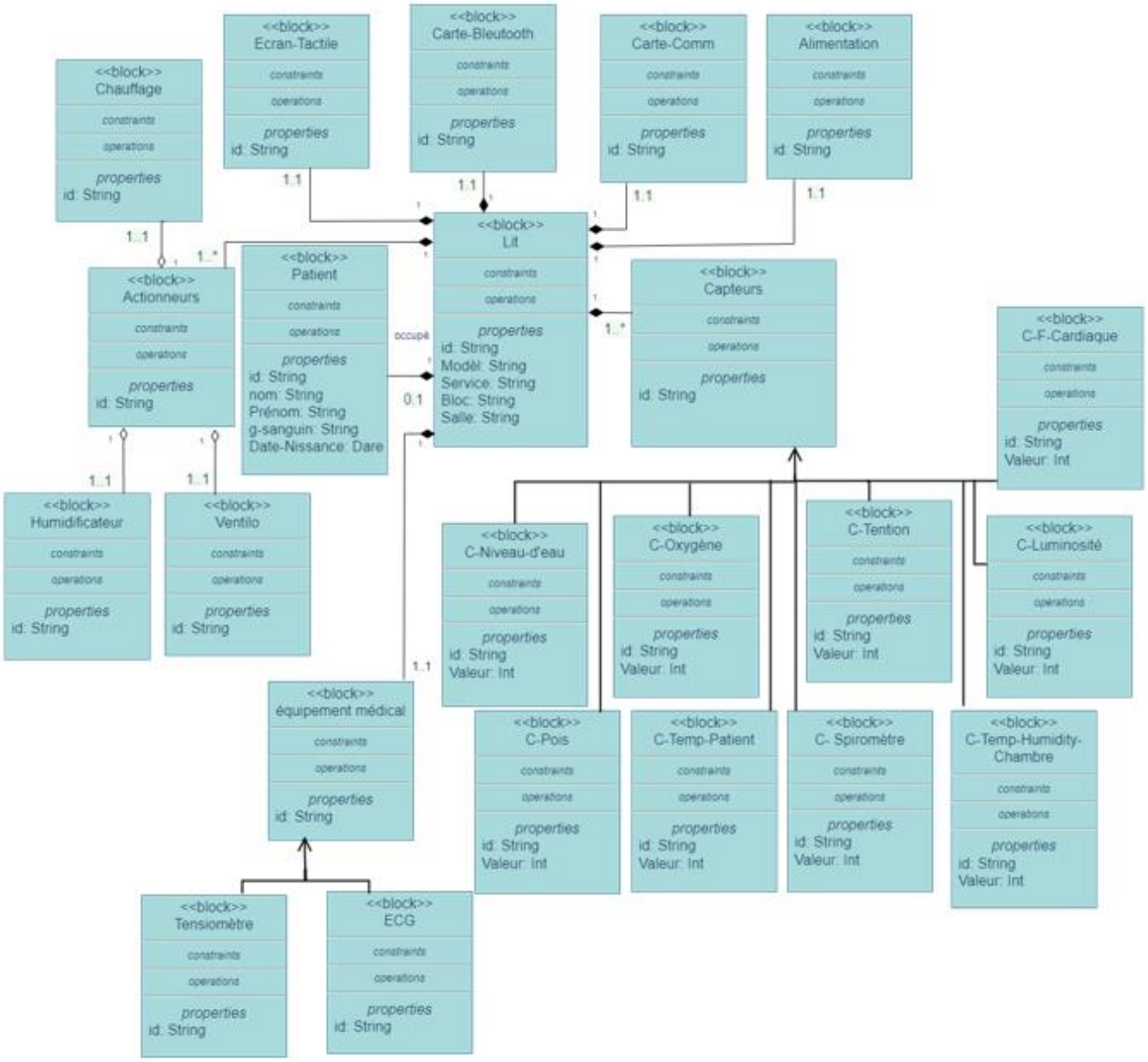


Figure III.4. Modèle de blocs du lit intelligent.

IV. Jumeau numérique hospitalier

Nous pouvons donc concevoir le jumeau numérique d'un lit d'hôpital intelligent, nous faisons la modélisation ou basé sur la création du modèle jumeau numérique et nous proposons d'utiliser le méta-modèle présenté dans la figure III.5. Ce méta-modèle a été proposé dans le cadre de la plateforme (Eclipse Ditto (API 2) [33]).

Comme montré par la figure III.5, Un modèle de jumeau numérique ditto est comparable à une classe.

Un modèle de jumeau numérique est décrit selon le méta-modèle suivant, qui est définit par deux méta-classes :

La méta classe Thing permet de représenter tous les objets. Features permet de représenter les caractéristiques des objets (Things), qui contient des propriétés de ces caractéristiques.
La Figure III.5 Cette conformité lui permet d'être supporté par la plateforme Eclipse ditto en version 2 de son API [33].

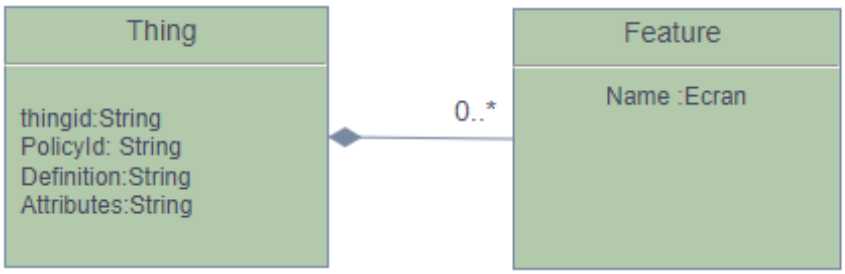


Figure III.5. Méta modèle des jumeaux numériques.

I. Chose (Thing)

Les objets sont des entités très génériques et sont principalement utilisés comme "handle" pour plusieurs fonctionnalités appartenant à cet objet.

Thingid : Identifiant unique d'un objet.

Policy ID : Liens vers l'ID d'une Policy existante qui contient les informations d'autorisation appliquées pour cet objet.

Définition : Une Chose peut contenir une définition. La définition peut également être utilisée pour trouver des (Things). La définition est utilisée pour lier une (Thing) à un modèle correspondant définissant les capacités/caractéristiques.

Les attributs : Les attributs décrivent la chose plus en détail et peuvent être de n'importe quel type. Les attributs peuvent également être utilisés pour trouver des choses. Les attributs sont généralement utilisés pour modéliser plutôt que des propriétés statiques au niveau de l'objet. Statique signifie que les valeurs ne changent pas aussi fréquemment que les valeurs de propriété des fonctionnalités.

II. Fonctionnalités (Features)

Une fonctionnalité est utilisée pour gérer toutes les données et fonctionnalités d'un objet pouvant être mises en cluster dans un contexte technique défini. Pour différents contextes ou aspects d'un objet, vous pouvez utiliser différentes fonctionnalités qui appartiennent toutes au même objet et n'existent pas sans cet objet.

ID feature : Dans un objet, chaque entité est identifiée par une chaîne unique - appelée Identité. Un ID de fonctionnalité doit souvent être défini dans le chemin d'une requête HTTP. De ce fait, il est fortement recommandé d'utiliser un jeu de caractères restreint.

Propriétés : Les données relatives aux fonctionnalités sont gérées sous la forme d'une liste des propriétés. Ces propriétés peuvent être classées, par exemple pour gérer l'état, la configuration ou toute information de panne. Les propriétés des fonctionnalités sont représentées sous la forme d'un seul objet JSON. Chaque propriété elle-même peut être une valeur simple / scalaire ou un objet complexe. Autorisé est une valeur JSON.

V. Model du jumeau numérique d'un lit intelligent

Le système de jumeau numérique d'une List d'hôpital intelligent peut être modélisés via plusieurs diagrammes de modélisation (séquence, (BDD), Diagramme interne de boc (IBD...Est). Pour le développement du jumeau numérique, le diagramme BDD est de très grande utilité. En effet, ce diagramme de bloc est un diagramme structurel qui décrit physiquement le système et/ou ses sous-systèmes et leur hiérarchie. Aussi décompose le système en blocs. Cette décomposition permet de montrer tous les composants du système d'un lit intelligent.

Le bloc « lit » devient un « thing » dans le modèle jumeau numérique. Les propriétés du bloc sont représentées par des attributs. Ceci est présenté par le tableau Suivant :

Le tableau III.1 représente les objets de notre système :

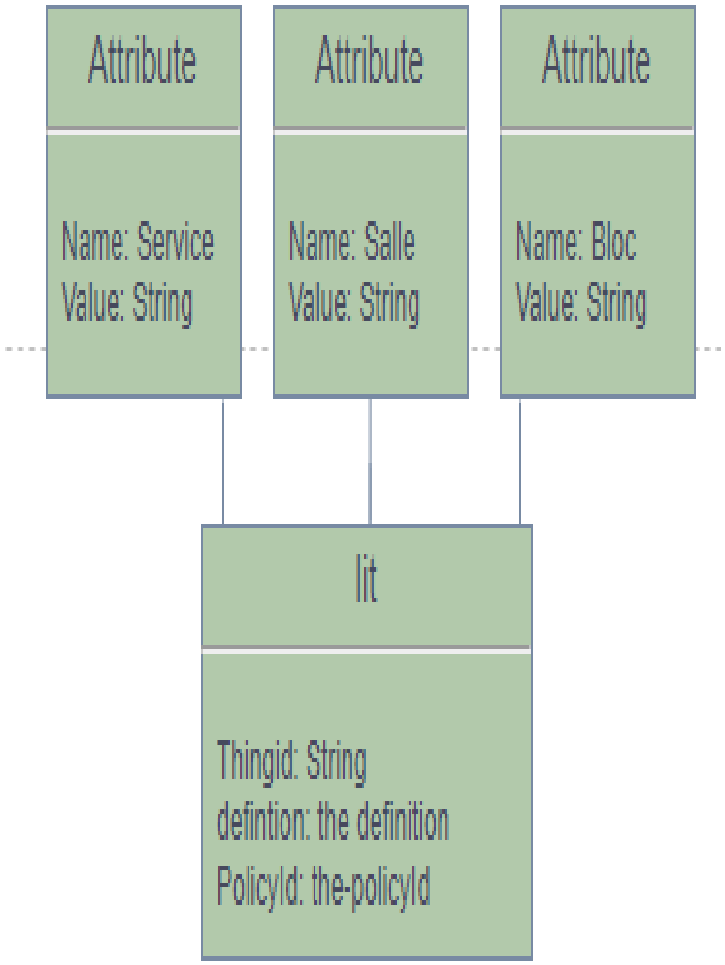
BDD	Modèle jumeau numérique
	

Tableau III.1. Représentation d'objet (things).

« Features » dans le modèle du jumeau numérique. Ceci est présenté par le tableau suivant :

Le tableau III.2 représente les objets avec les fonctionnalités (Features) :

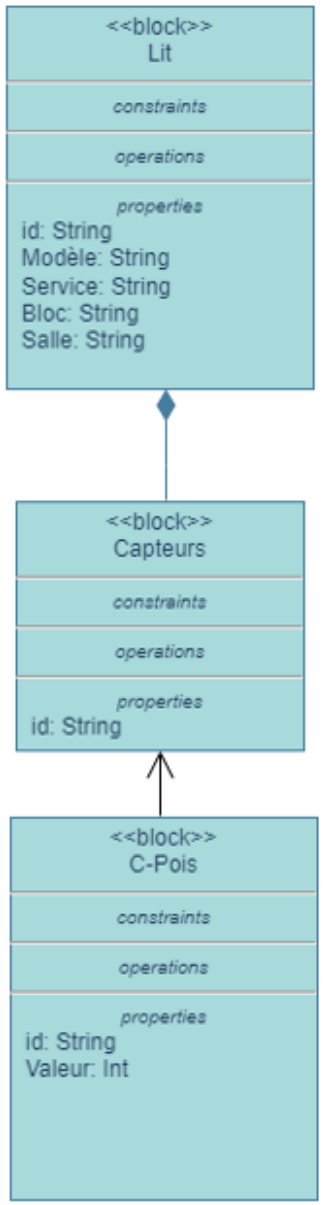
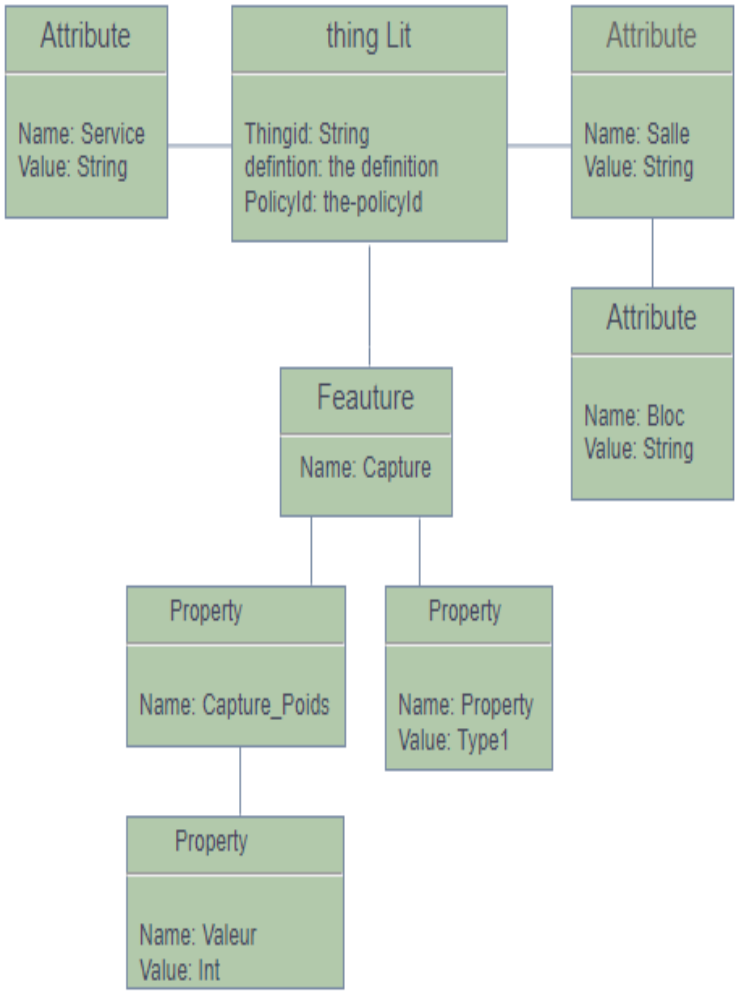
BDD	Modèle jumeau numérique
	

Tableau III.2. Représentation l'objet avec les fonctionnalités (Features).

Les propriétés des blocs composant le lit sont présentées ainsi que les blocs liés aux composants sont représentés par des « properties » de « features » complexes ou simples.

Le tableau III.3 représenté les fonctionnalités de notre modèle jumeau numérique :

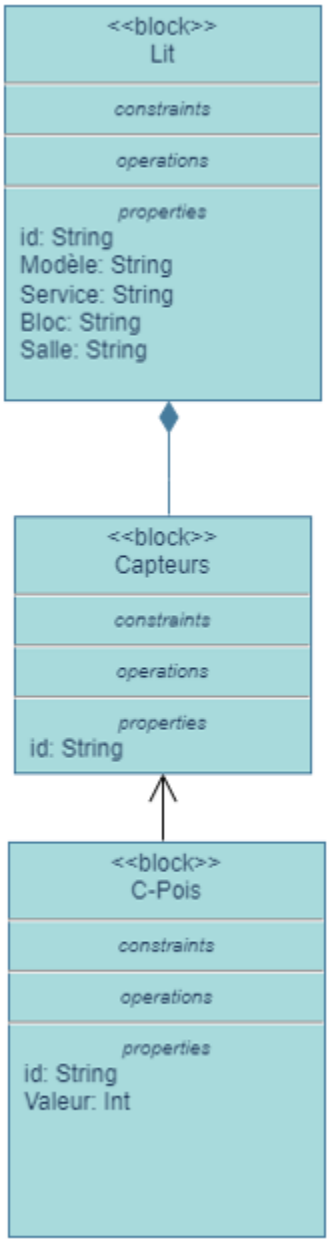
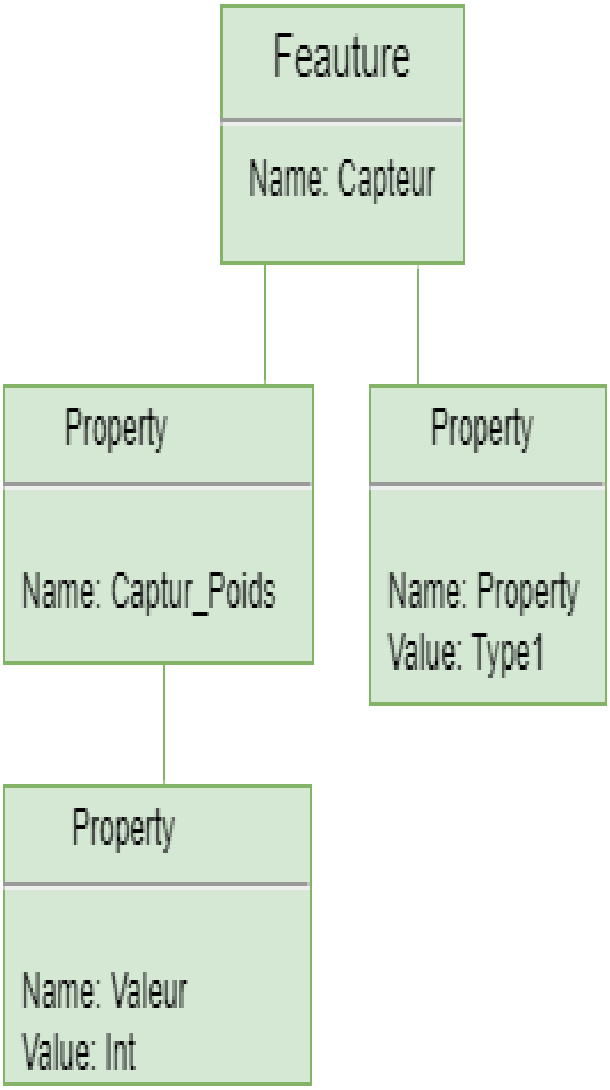
BDD	Modèle jumeau numérique
	

Tableau III.3. Représentation des fonctionnalités.

Le modèle global du jumeau numérique

La figure III.6 représente le modèle jumeau numérique d'un lit intelligent :

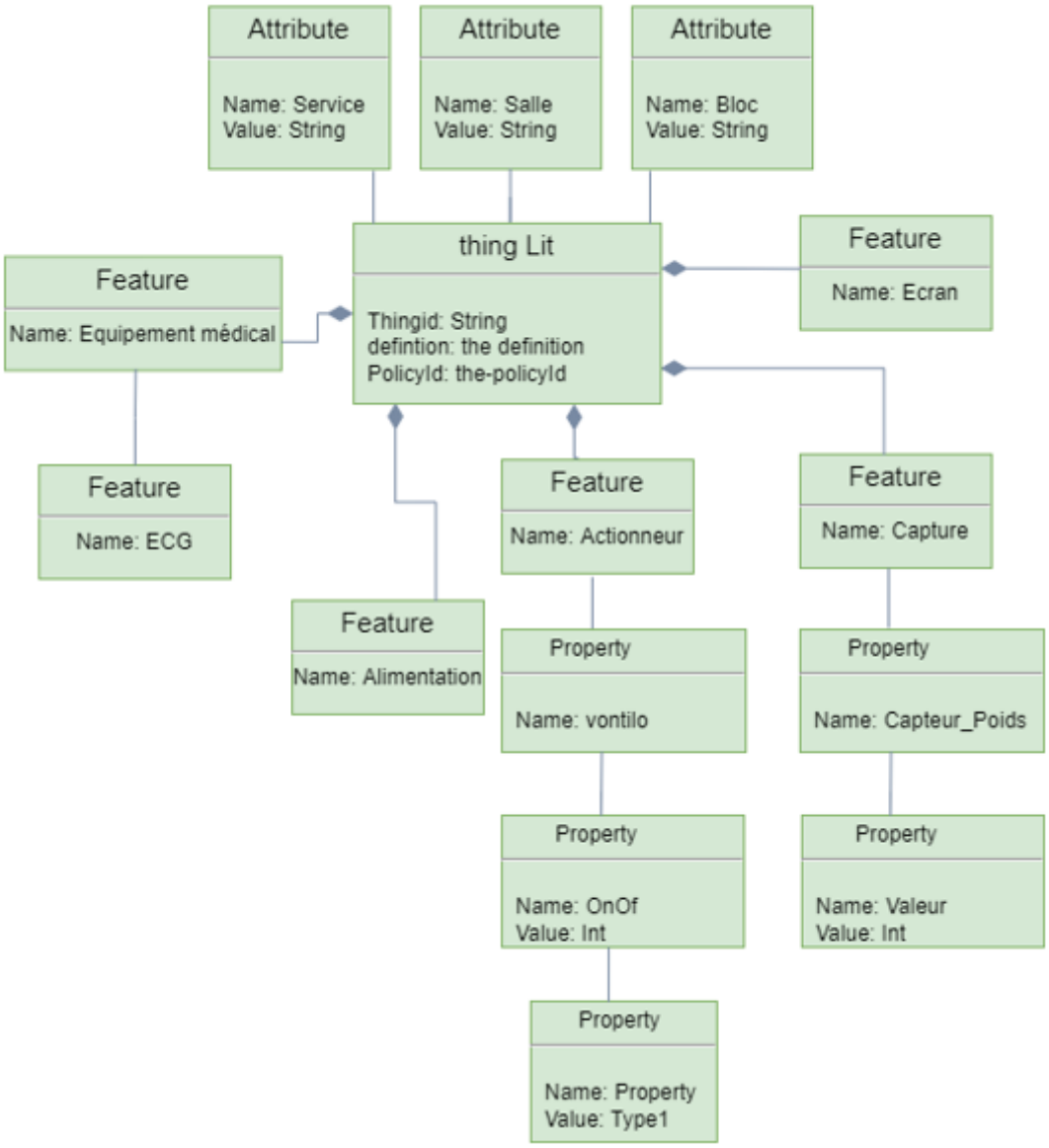


Figure III.6. Modèle jumeau numérique d'un lit intelligent.

La représentation du modèle jumeau numérique du lit d'hôpital par un fichier JSON doit respecter la hiérarchie et la décomposition des concepts de l'objet, la figure définit la forme JSON des objets du lit d'hôpital intelligent.

La figure III.7 représente le format de JSON du jumeau numérique :

```
{
    "thingId": ":1",
     "policyId": ":1",
     "attributes": {
                        "etat": "Occupied",
                        "bloc": "A ",
                        "service": "operative ",
                        "salle": "1",
                        "project": "SmartBed",
                        "modele": "standard",
                        "description": "Description"
},
    "features": {
              "capteur":{
                   "properties": {
                        "body_t_sensor": { "valeur": 0 },
                        "sensor_poit":  { "valeur": 25 } }
},
              "device": {
                   "properties": {
                        "tentionmetr": "valeur" :"13","valeur":10 }
},
              "actionneur_lampe_uv": {
                   "properties": { "action_onoff": 0 }},
                        "actionneur_ventilo": {
                             "properties": { "action_onoff": 0 }},
                        "actionneur_chauffage": {
                        "properties": { "action_onoff": 0 } },
                        "actionneur_humidificateur": {
                        "properties": { "action_onoff": 0 }},
                   "patient": {
                     "properties": {
                        "nom": "Ali",
                        "prenom": "amin",
                        "date_nais": "1994/05/05",
                        "groupage": "O+",
                        "observations": " observations"
 } },
              "ecran_tactille": {
                   "properties": {} },
               "allimentation": {
                    "properties": {} },
               "bluetooth": {
                     "properties": {} }
}
 },
```


Figure III.7. Exemple de format de JSON du jumeau numérique.

VI. Application de contrôle et de supervision

Dans notre système de gestion des patients au niveau d'hôpital nous avons créé une application basée sur le jumeau numérique qui permettant à la supervision médicale à travers les données capturées à partir des lits intelligent. Ainsi que le mécanisme pour le contrôle des lits à distance.

L'objectif de cette application c'est l'amélioration prestations de soins destinées aux patients pour les différents intervenant de la santé à travers :

- La supervision et le contrôle médicale à distance.
- La Suive médical attentif du patient.
- La gestion des lits et leur affectation aux services d'hospitalisation.
- Réduire les risques grâce à une intervention médicale rapide.

VI.1. Diagramme de séquence

Le diagramme de séquence est une solution populaire de modélisation dynamique en langage Sysml, ils se concentrent plus précisément sur les lignes de vie, les processus et les objets qui vivent simultanément, et les messages qu'ils échangent entre eux pour exercer une fonction avant la fin de la ligne de vie.

Dans ce qui suit, nous allons décrire quelques scenarios concernant le fonctionnement de notre système par des diagrammes de séquence.

VI.1.1. Diagramme de séquence pour « Authentification »

Ce diagramme de séquence de connexion du système d'authentification de connexion, où l'utilisateur pourra se connecter à son compte en utilisant ses informations d'identification. Si un champ est vide un message d'erreur sera afficher, sinon le système vérifie le mot de passe et le nome. Si les informations entrées sont incorrectes, un message d'erreur sera affiché. Si les informations entrées sont correctes l'utilisateur peut accéder au menu principal de l'application. Après la connexion, l'utilisateur peut gérer toutes les opérations sur l'autorisation du nom d'utilisateur, et l'utilisateur peut accéder à ces pages après la connexion. Le diagramme ci-dessous aide à démontrer comment la page de connexion fonctionne dans le système d'authentification de connexion. Cours de la séquence, et l'utilisateur ne pourra pas accéder à cette page sans vérifier son identité.

La Figure III.8 : description des séquences à effectuer pour entamer d'authentification.

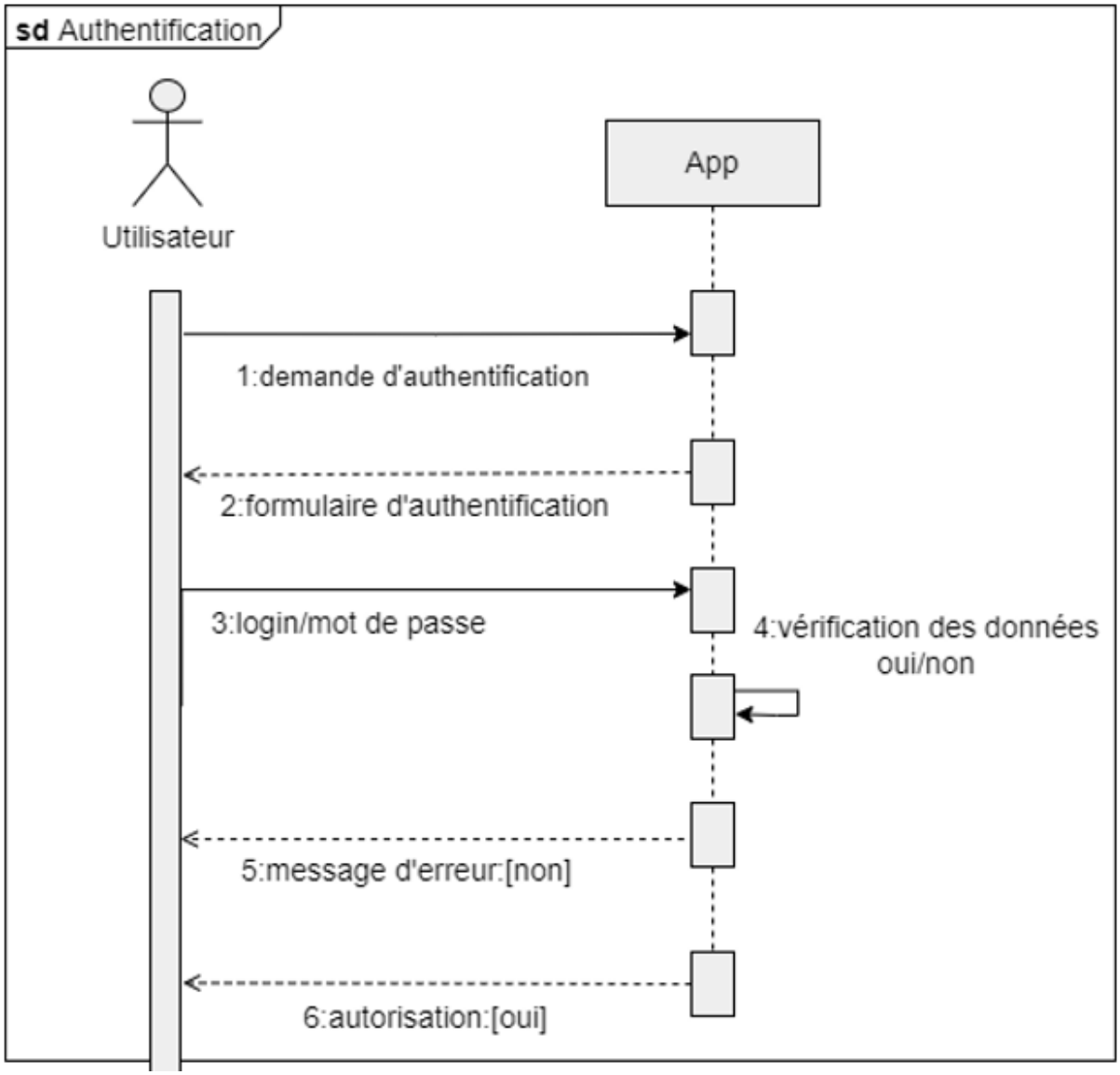


Figure III.8. Diagramme de séquence pour « L'Authentification ».

VI.1.2. Diagramme de séquence pour « La création de lit »

Ce diagramme de séquence de la création des lits dans la plateforme de jumeau numérique (Eclipse Ditto).

La Figure III.9: décrire la phase de la création des lits dans la plateforme de jumeau numérique (Eclipse Ditto).

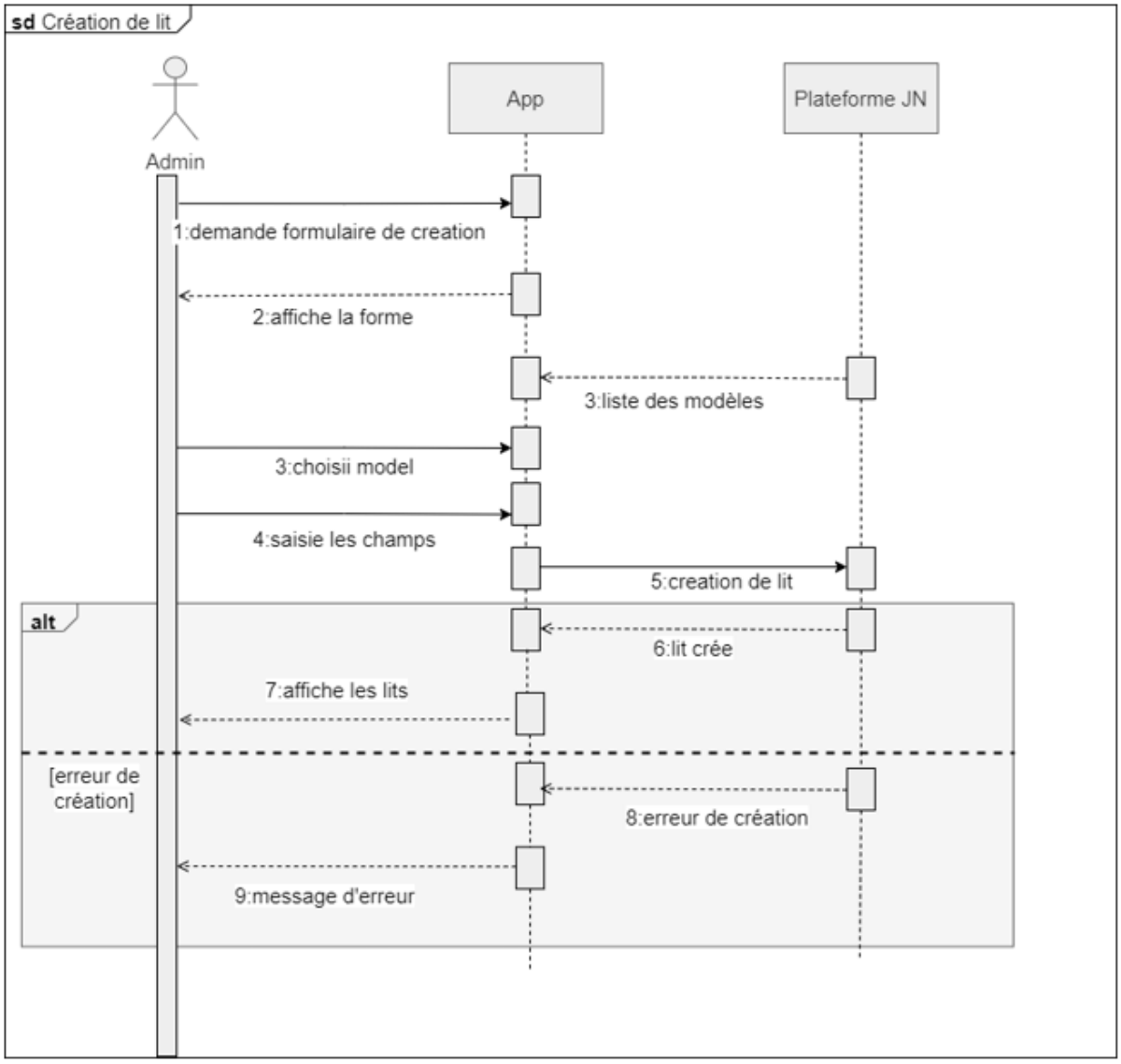


Figure III.9. Diagramme de séquence pour La création des Lits.

VI.1.3. Diagramme de séquence pour « La création de modèle d'un lit »

Ce diagramme de séquence de la création des modèles (lit) dans la plateforme de jumeau numérique (Eclipse Ditto).

La Figure III.10: décrire la phase de la création des modèles des lit intelligent dans la plateforme de jumeau numérique (Eclipse Ditto).

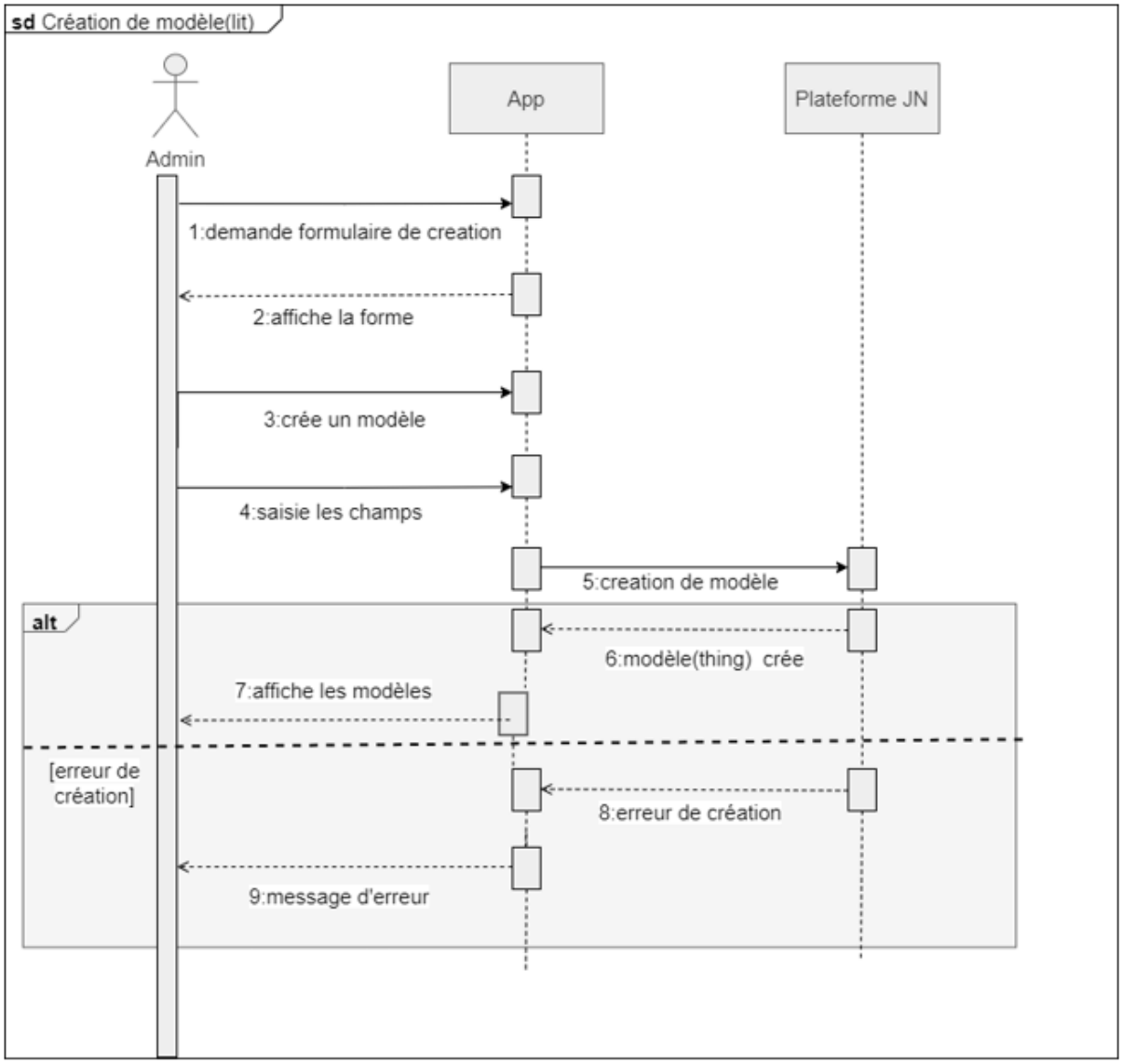


Figure III.10. Diagramme de séquence pour « La création des modèles (lit).

VI.1.4. Diagramme de séquence pour supervision « jumeau numérique »

Ce diagramme de séquence de la supervision jumeau numérique (Eclipse Ditto) doit être fait par l'administrateur.

La Figure III.11 : description des séquences à effectuer pour entamer la phase de contrôle et la supervision de jumeau numérique.

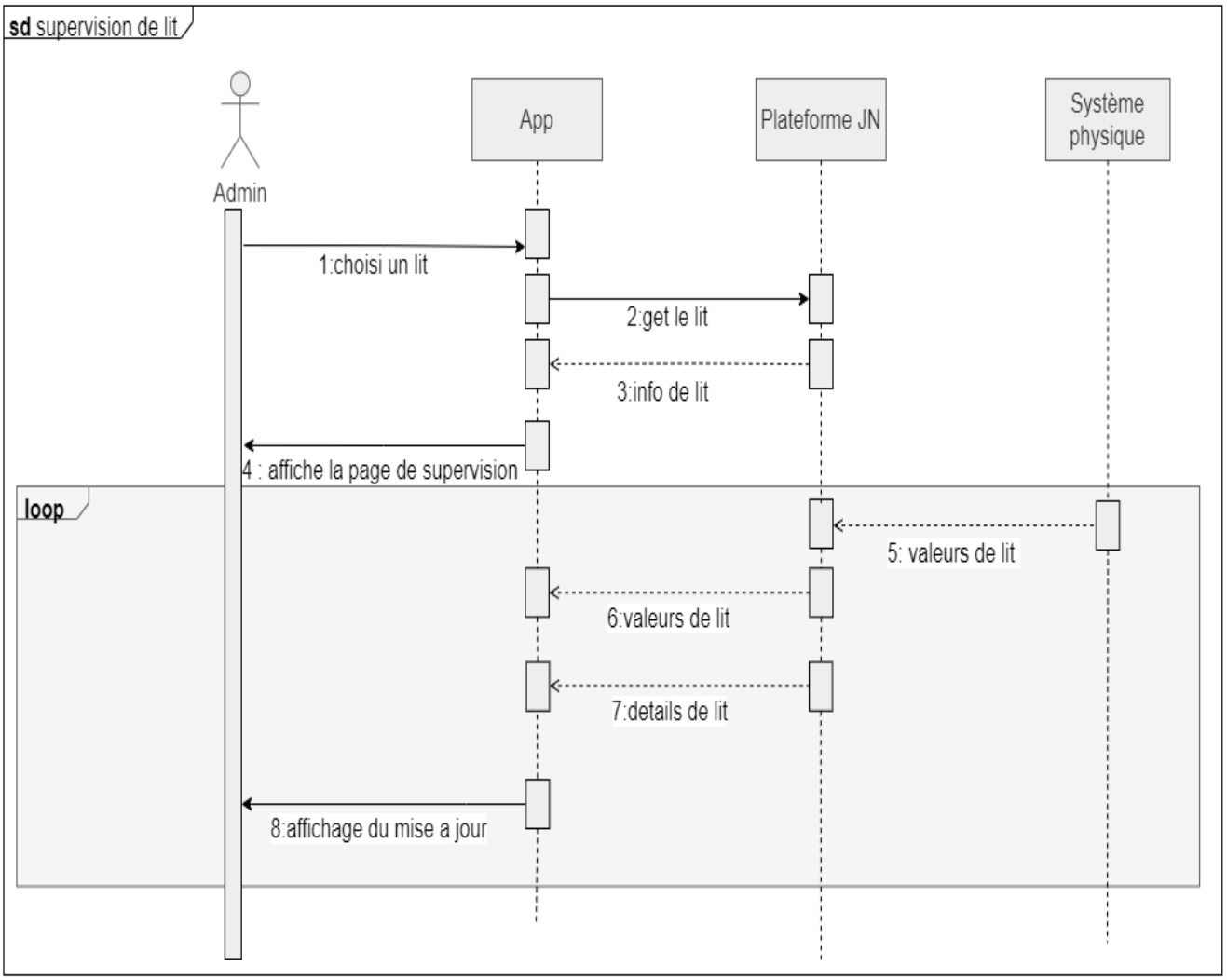


Figure III.11. Diagramme de séquence pour La supervision d'un lit intelligent « jumeau numérique ».

Ce diagramme de séquence de la supervision jumeau numérique (Eclipse Ditto) doit être fait par le médecin.

La Figure III.12 : description des séquences à effectuer pour entamer la phase de supervision des patients et leur lits intelligents.

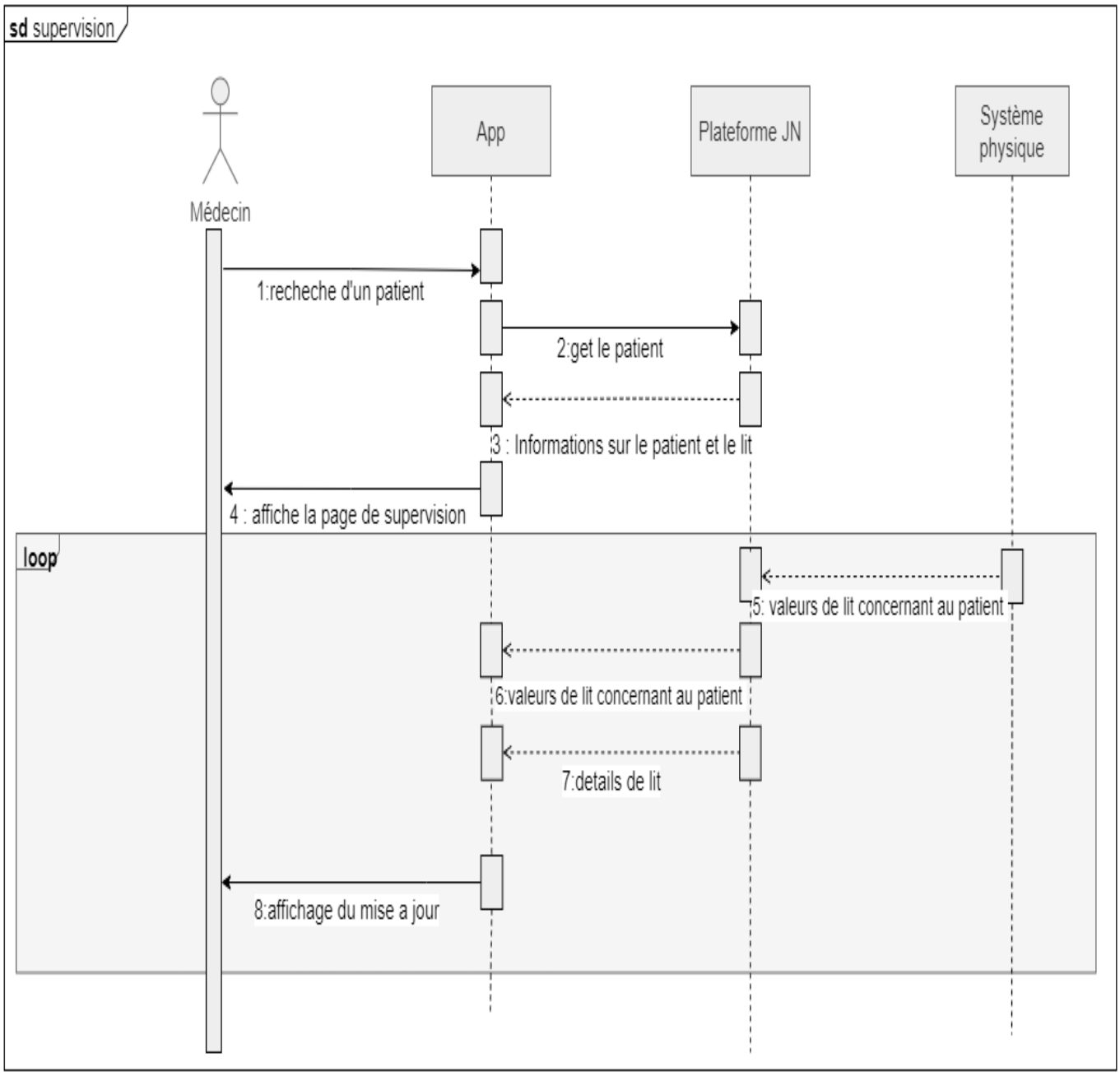


Figure III.12. Diagramme de séquence pour La supervision médicale.

VI.1.5. Diagramme de séquence pour contrôle « jumeau numérique ».

Ce diagramme de séquence du contrôle jumeau numérique (Eclipse Ditto)

La Figure III.13 : description des séquences à effectuer pour entamer la phase de contrôle les lits d'hôpital intelligent.

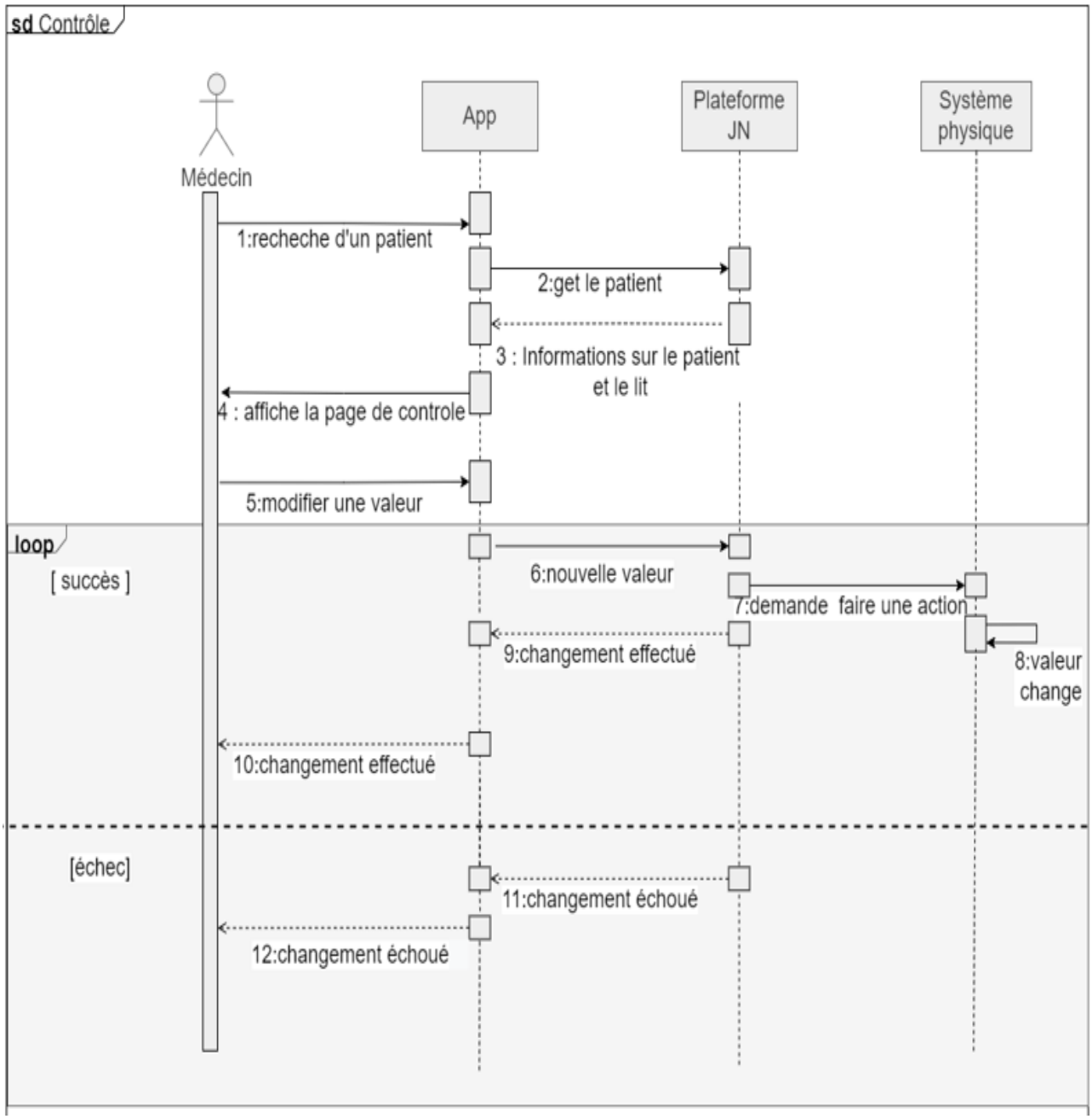


Figure III.13. Diagramme de séquence pour contrôler les lit « jumeau numérique ».

VI.1.6. Diagramme de séquence pour la gestion des patients.

Ce diagramme de séquence pour la gestion des patients

La Figure III.14 : description des séquences à effectuer pour entamer la phase de gestion des patients.

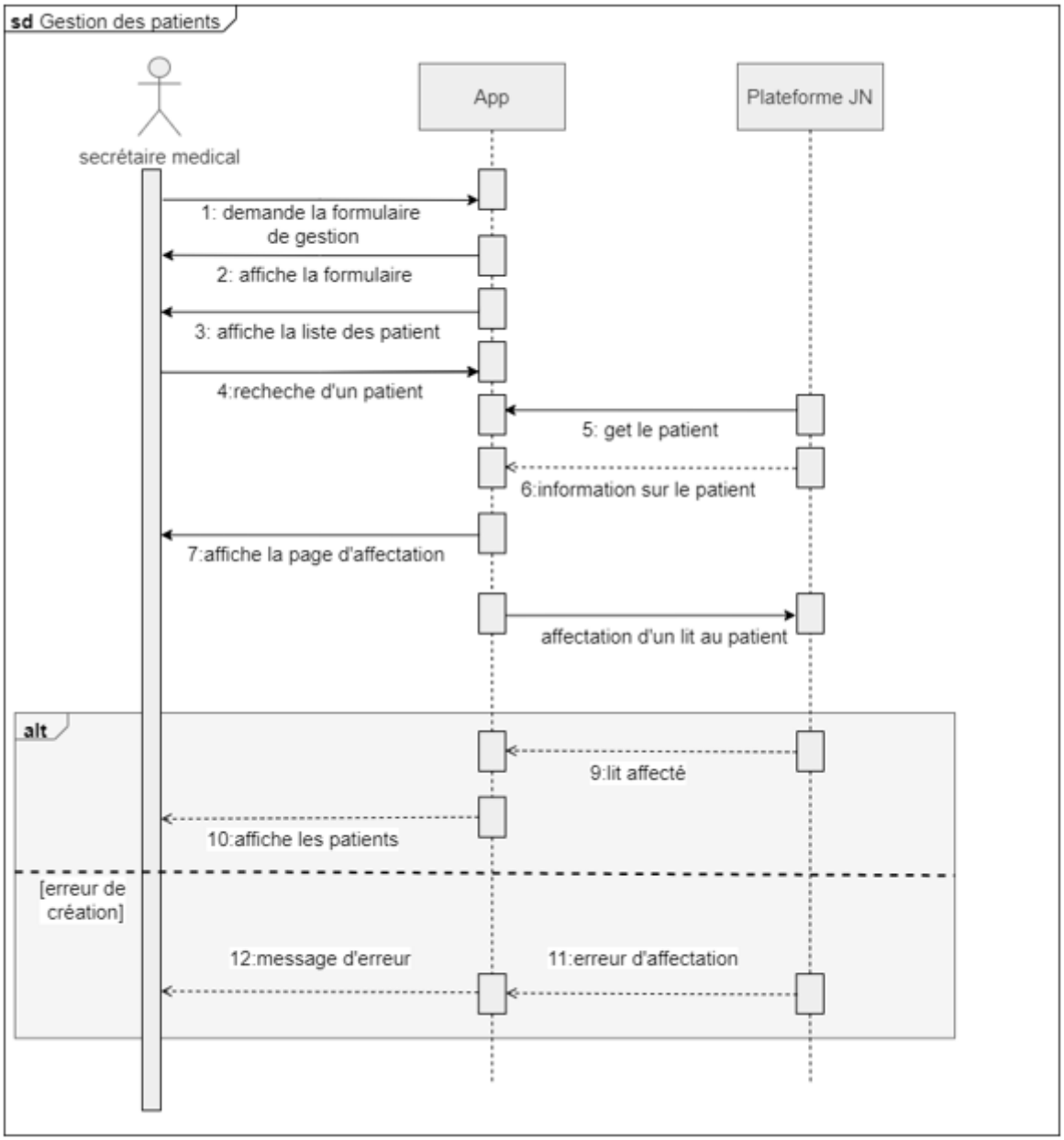


Figure III.14. Diagramme de séquence pour la gestion des patients.

VII. Conclusion

Dans ce chapitre, nous avons conçu et développé une solution basée sur les jumeaux numériques et les micro-services pour contrôler et supervisé des patients au niveau d'hôpital et leur lits intelligents. Connectés au jumeau numérique, à travers l'application conçue.

Le chapitre suivant fera l'objet du développement de l'application pour créer l'image numérique des lits d'hôpital intelligents et fournir les fonctionnalités aux différents acteurs de santé.

Chapitre IV : Implémentation

I. Introduction

Le but de ce chapitre est d'implémenter la démarche proposée dans le chapitre précédent. Pour cela, nous avons développé une application web permettant de réaliser toutes les étapes de notre conception.

Notre application permettant a supervisé et contrôlé les lits intelligent d'hôpital et gestion des patients à travers leurs jumeaux numériques. Nous décrivons aussi, dans ce chapitre l'ensemble des moyens technologiques utilisés dans le développement de notre plateforme.

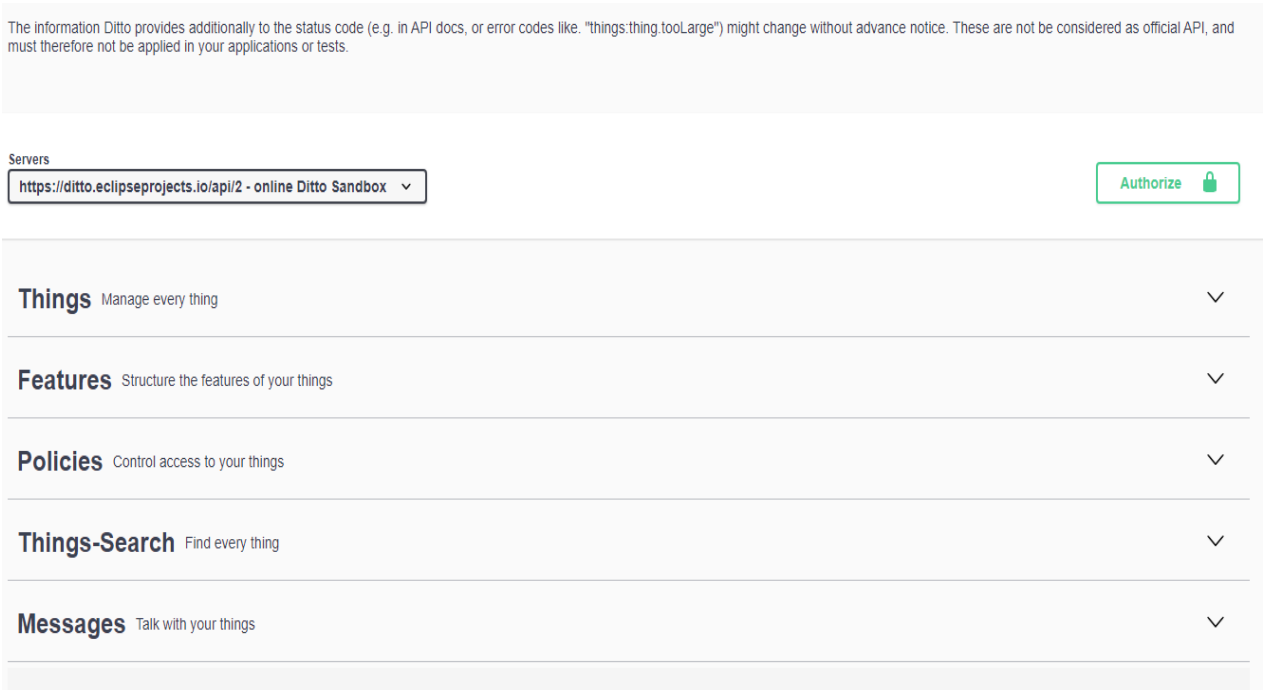
II. Implémentation

Dans la partie implémentation nous avons expliqué les différents composants de notre environnement d'implémentation et les outils utilisés, pour la créer des jumeaux numériques et communiquer avec les lits d'hôpitaux intelligents.

II.1. Plateforme des jumeaux numériques

Dans notre travail. Nous avons utilisé la plateforme Eclipse Dito, qui est une technologie de l'IOT destinée à mettre en œuvre et faciliter la création d'un model numérique. Elle est utilisée pour produire et fournir des outils pour la réalisation des logiciels.

La Figure IV.1 représenté l' interface principale de l'API Eclipse Ditto:



La Figure IV.1. Interface principale de l'API Eclipse Ditto.

II.2. Node Red

Node-RED est un outil puissant pour créer des applications (IoT) est un outil visuel pour câbler l'Internet des objets.

Pour assurer la liaison du système physique et le jumeau numérique nous avons utilisé une couche basée sur l'outil Node-RED.

L'outil Node-RED a été utilisé pour connecter les différents capteurs et actionneurs du système physique et récupérer les données capturées pour les communiquer au jumeau numérique.

La Figure IV.2 représenté la connexion avec les lits d'hôpital intelligent :

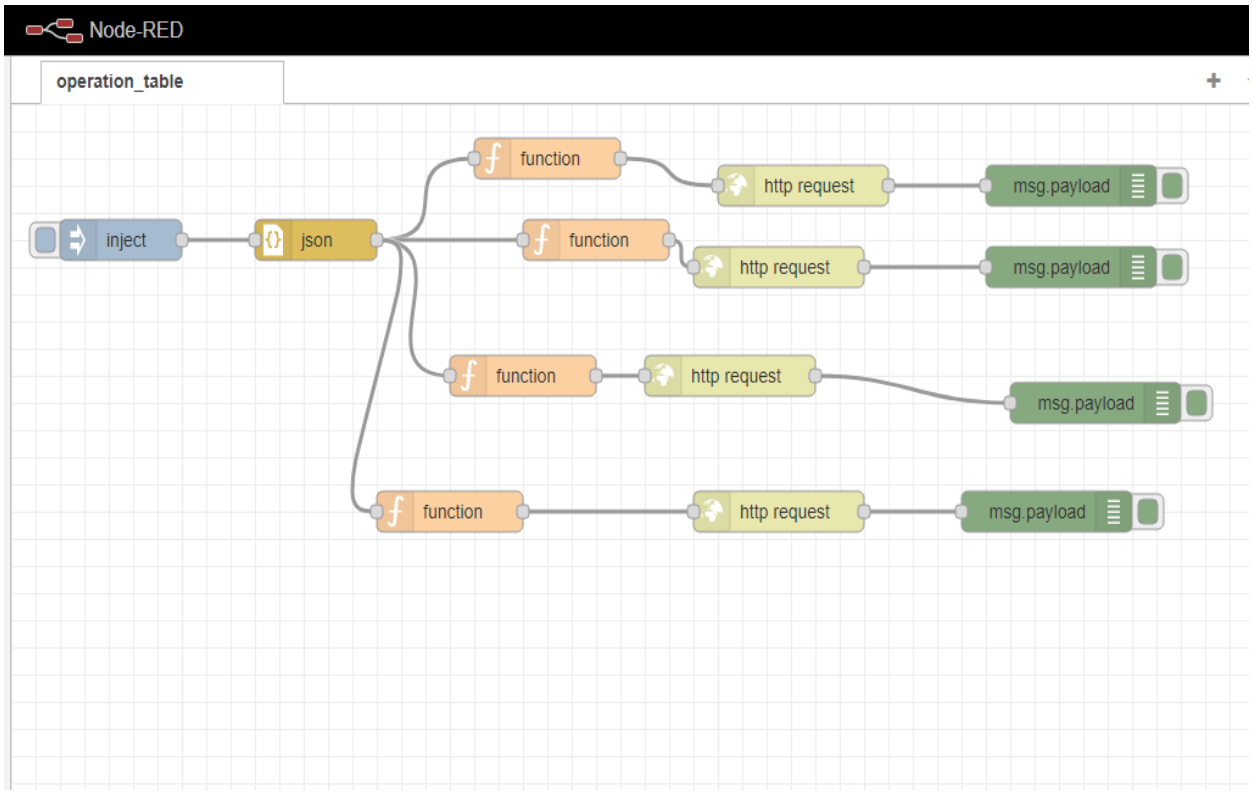


Figure IV.2. Connexion avec les lits d'hôpital intelligents.

II.3. Kafka

Apache Kafka est un outil pour les micro-services ou il permet la connexion et la communication entre eux via deux modèles de messagerie, la mise en file d'attente et la publication-abonnement (ce que nous avons utilisé).

L'outil Apache Kafka a été utilisé comme gestionnaire d'événements pour gérer tout ce qui est dynamique et communique avec différents consommateurs/Producteurs d'évènements (utilisateurs, l'App web, lits d'hôpital intelligents).

Le tableau IV.3 représenté le rôle de Broker dans la gestion des évènements.

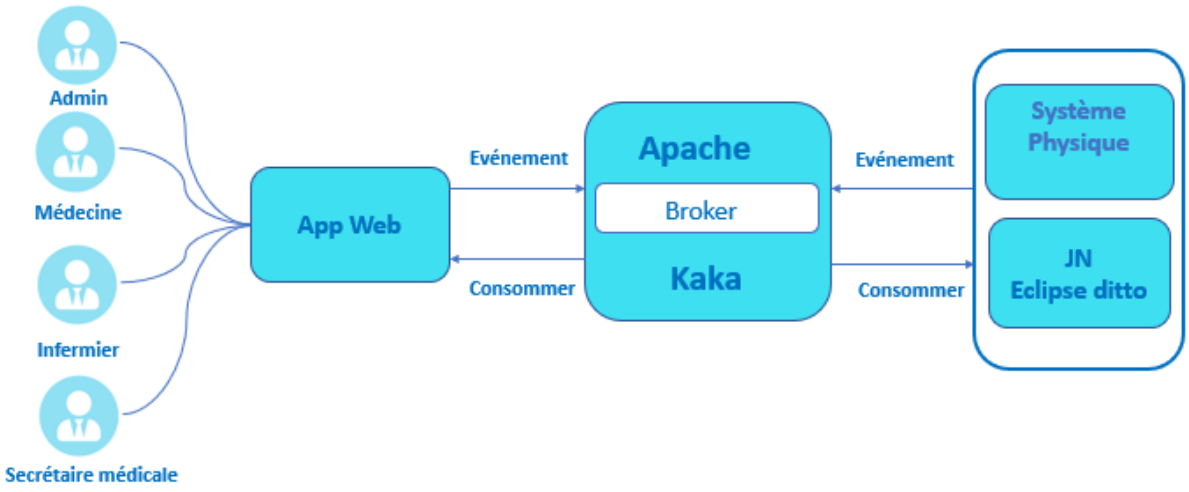


Figure IV.3 Le rôle de Broker dans la gestion des événements.

La Figure IV.4 représenté la connexion entre Node-Red Kafka :

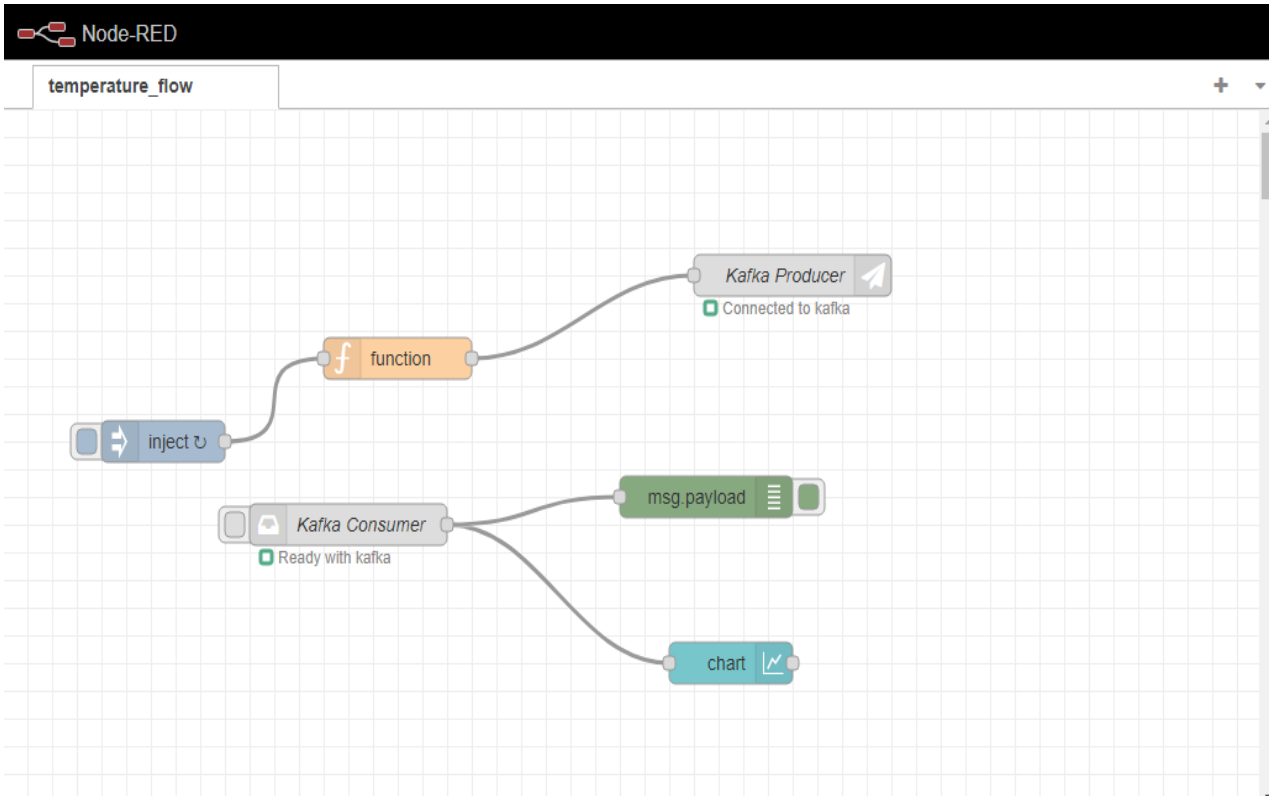


Figure IV.4. Connexion entre Node-Red Kafka.

III. Application web

Afin de réaliser notre application, Nous avons utilisé plusieurs langages et outils de développement comme le suivant :

HTML pour représenter les pages web, CSS (Cascading Style Sheets) pour le style et JavaScript et PHP pour produire des pages dynamiques et envoi/réception des requêtes HTTP sous format de JSON.

III.1. Interface d'authentification

Cette interface d'authentification permet aux utilisateurs soit (Admin, Médecin, Infirmier, Secrétaire Médical) d'accéder à l'application via l'email et le mot de passe d'utilisateur.

La figure IV.5 : représente l'interface d'authentification d'application.

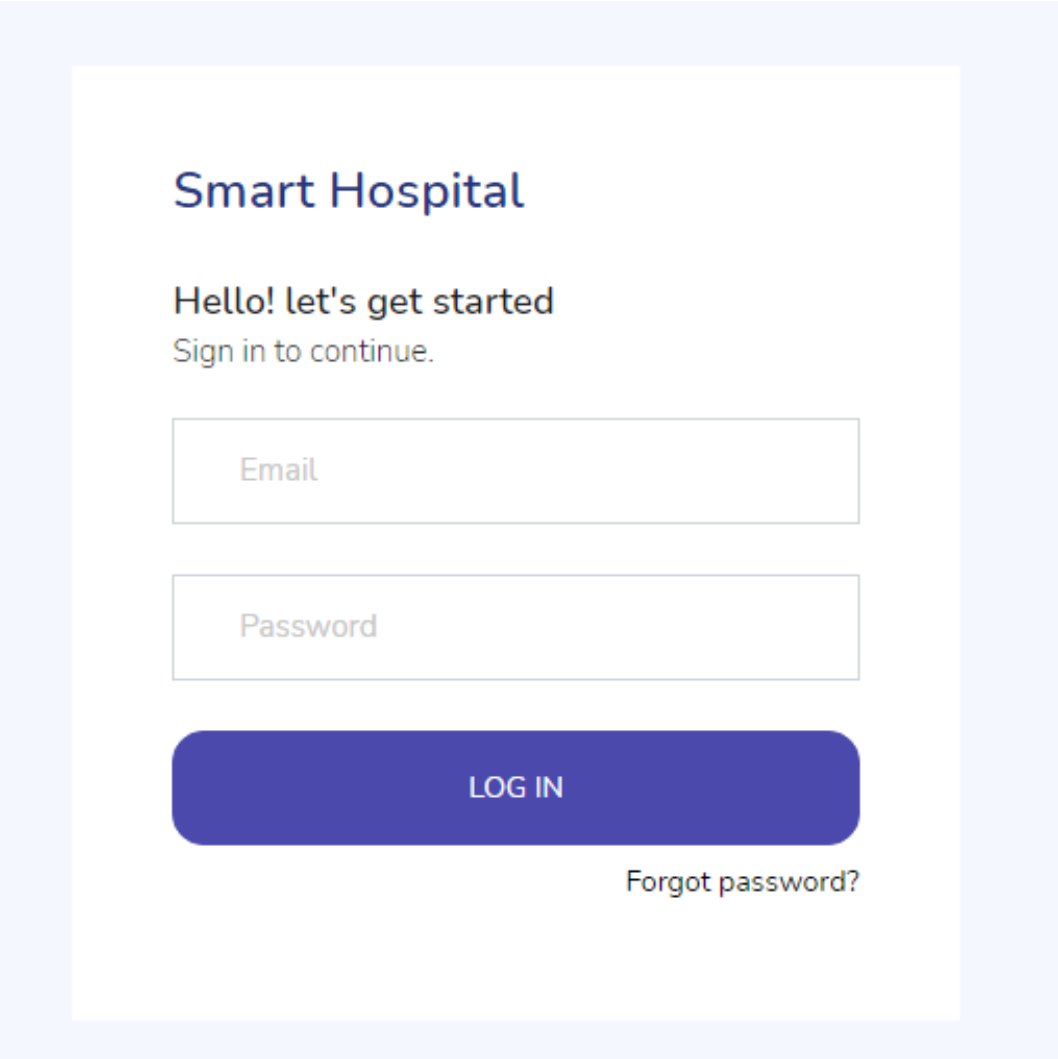


Figure IV.5. Interface d'authentification.

III.2. Interface d'accueil pour l'administrateur

L'interface d'accueil de l'application web développée est composée d'un menu permettant à l'administrateur et le médecin, infirmier, secrétaires médicale d''accès aux différentes fonctionnalités.

L'administrateur peut gérer les utilisateurs et la gestion des lits connectes (liste patient modifié, ajoute.

L'infirmier et le médecin peuvent superviser et contrôle les patients.

Le secrétaire médical peut gère la gestion des patients (ajoute, affectation.). Elle offre aussi un tableau de bord qui affiche la liste des patients (libres ou occupés) et l des statistiques sur nombre des lits (occupées et libres) dans l'hôpital ainsi que des patients sur ces lits. L'administrateur peut créer des utilisateurs de l'application avec déférents rôles (Médecin, Infirmier, Secrétaire médical).

La figure IV.6 : représente l'interface d'accueil d'application.

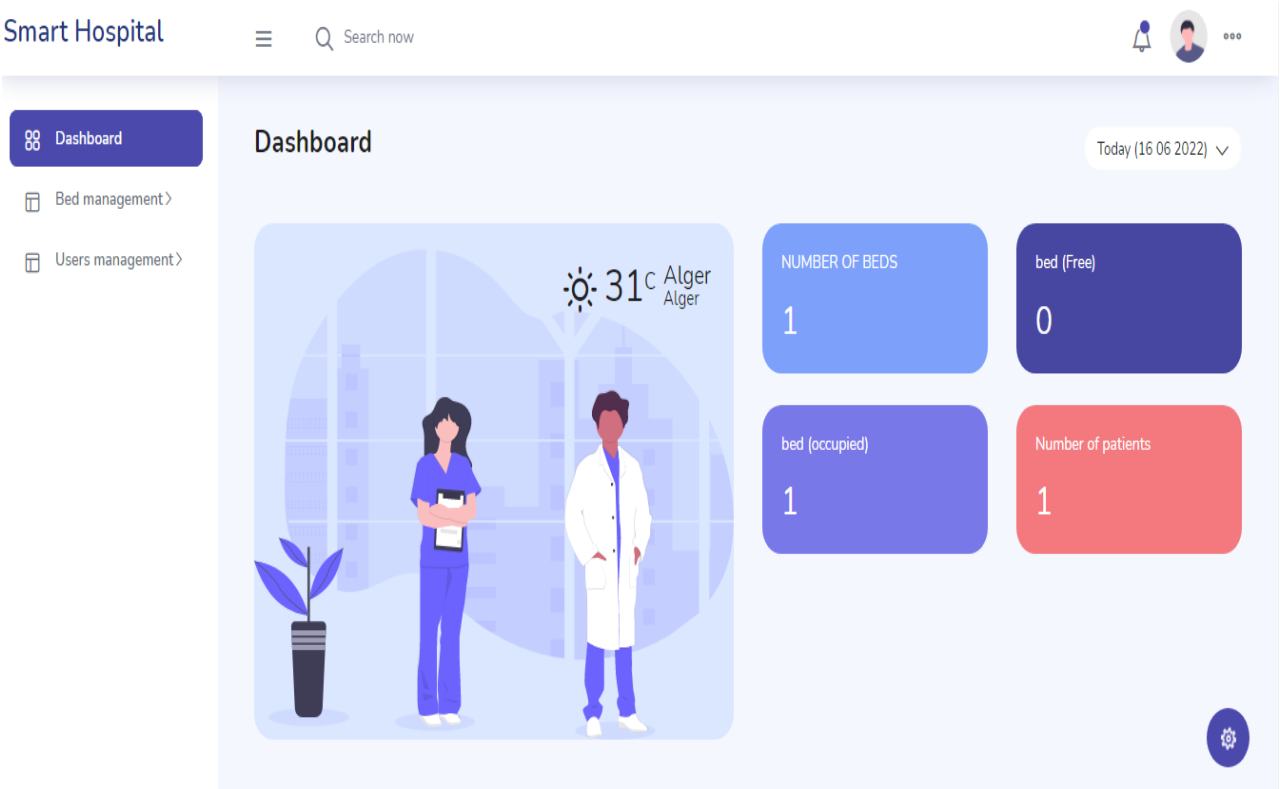


Figure IV.6. Interface d'accueil de l'application web.

III.3. Interface de gestion des lits d'hôpital intelligent

Cette interface s'affiche lorsque l'administrateur choisit de visualiser et gestion des lits d'hôpital intelligents.

L'admin peut choisir un modèle en fonction des listes modèles puis cliquer sur le bouton « create » pour continuer à créer.

La figure IV.7 : représente l'interface des listes modèles.

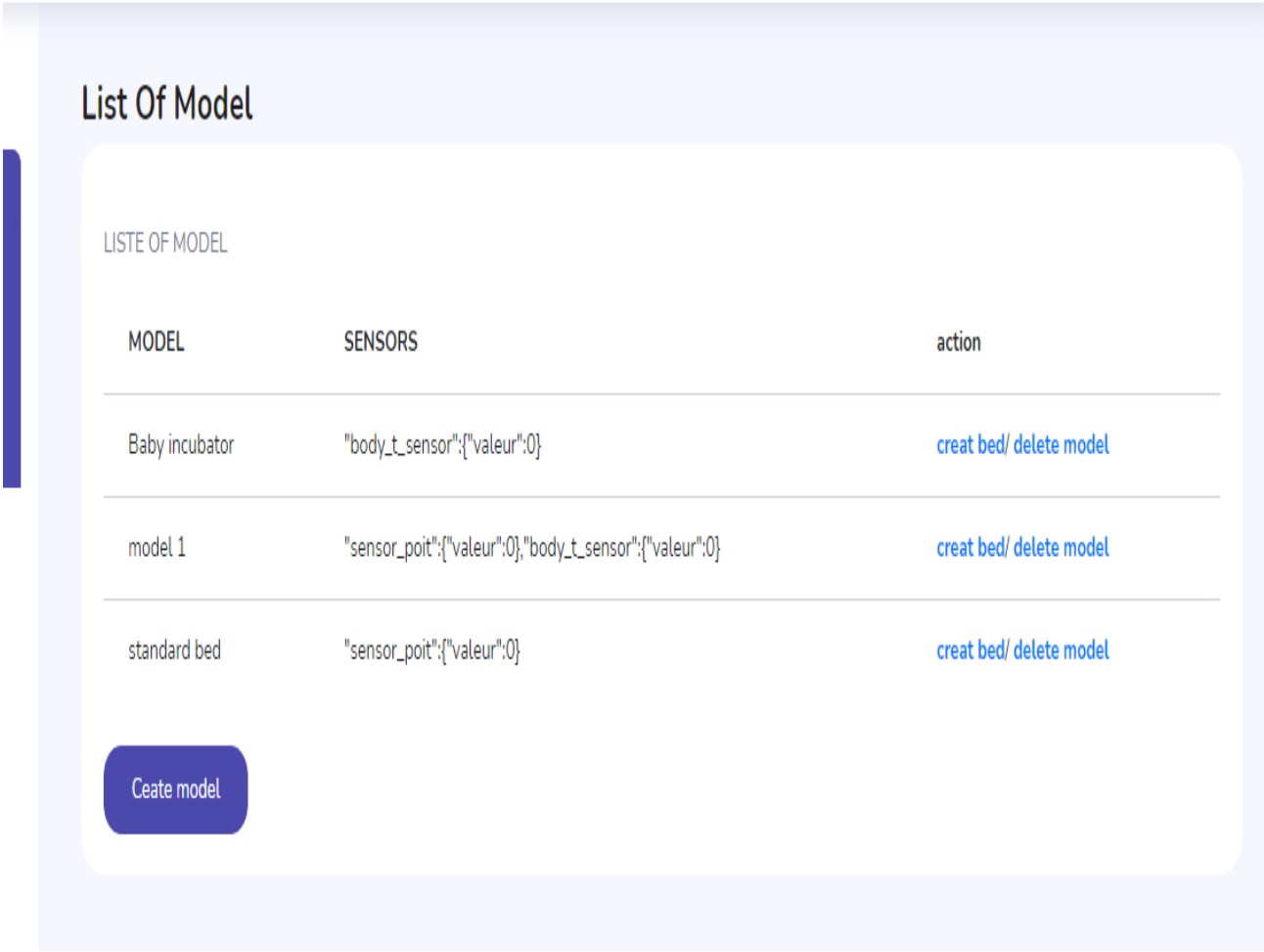


Figure IV.7. Interface de liste modèles.

III.4. Interface de création de modèles d'un lit intelligent

Cette interface s'affiche lorsque l'administrateur choisit de créer un modèle de lit d'hôpital intelligent.

La figure IV.8 représente l'interface de création de modèle d'un lit d'hôpital intelligents.

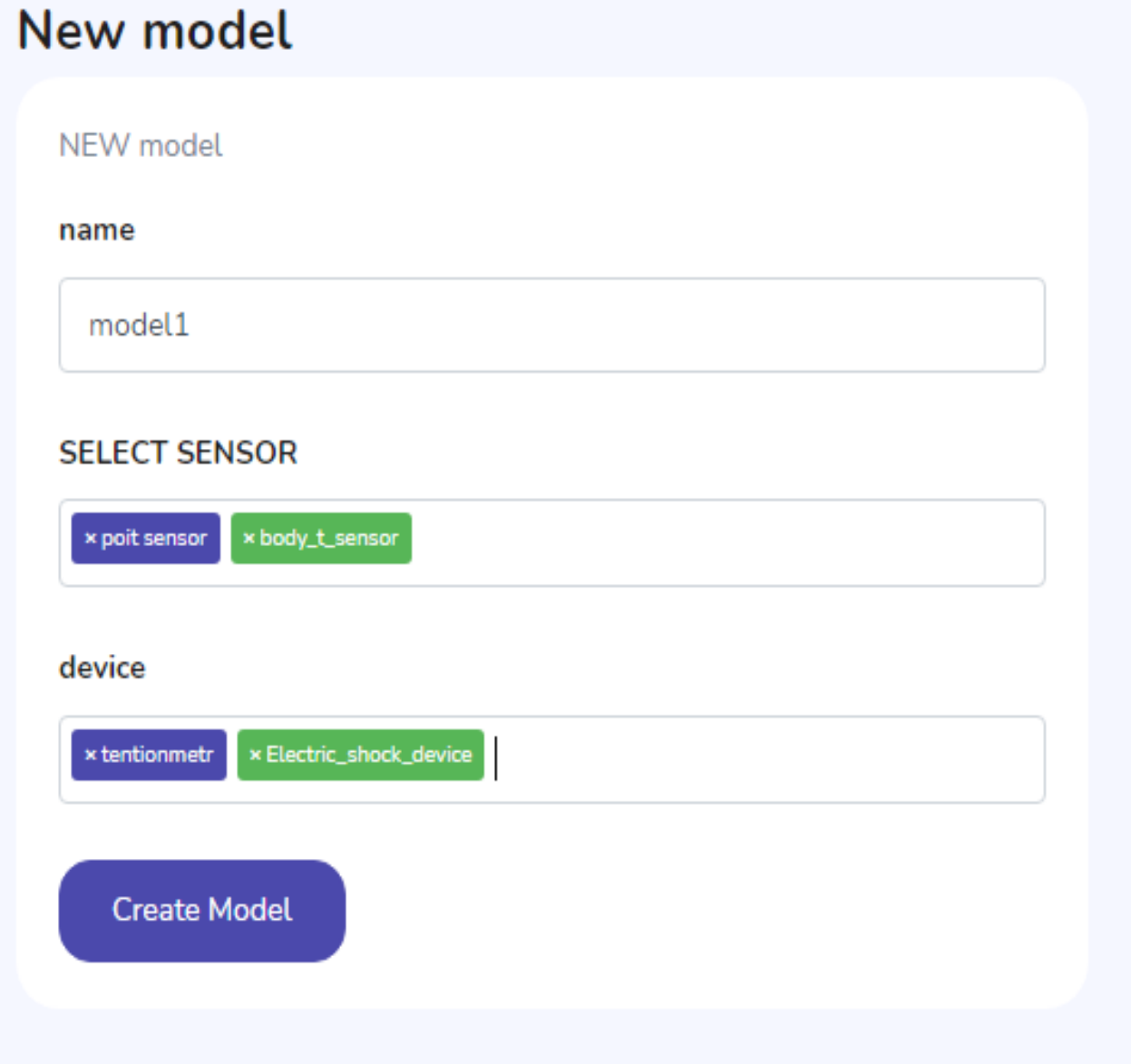
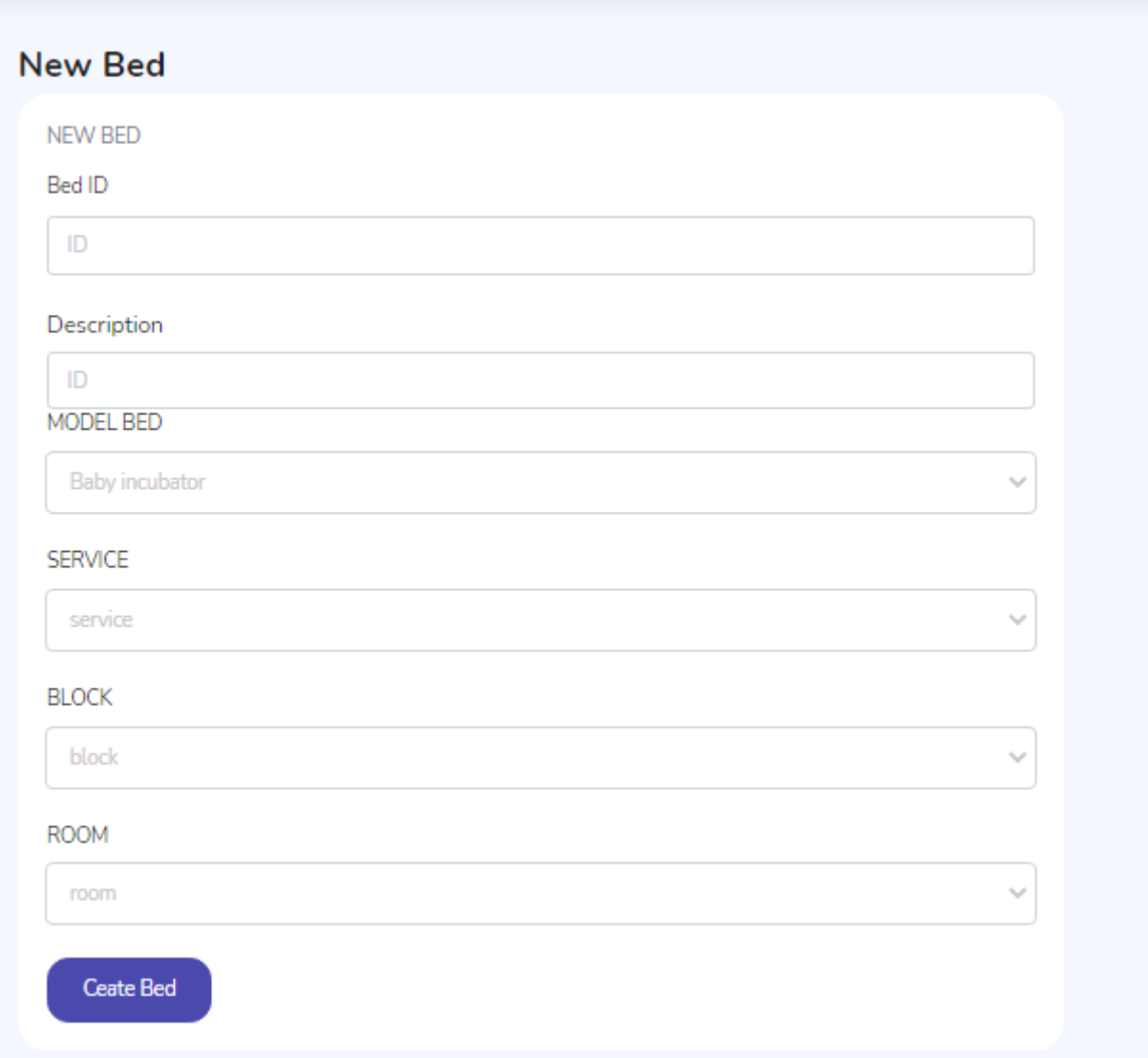


Figure IV.8. Interface de création de modèles.

III.5. Interface de création de lit intelligent

L'administrateur doit saisir les informations d lit d'hôpital intelligent et cliquer sur le Button « Create » pour crée le lit dans la plateforme du jumeau numérique ditto.

La figure IV.9 représente l'interface de création des lits d'hôpital intelligents



. **Figure IV.9.** Interface de création de lit d'hôpital intelligent.

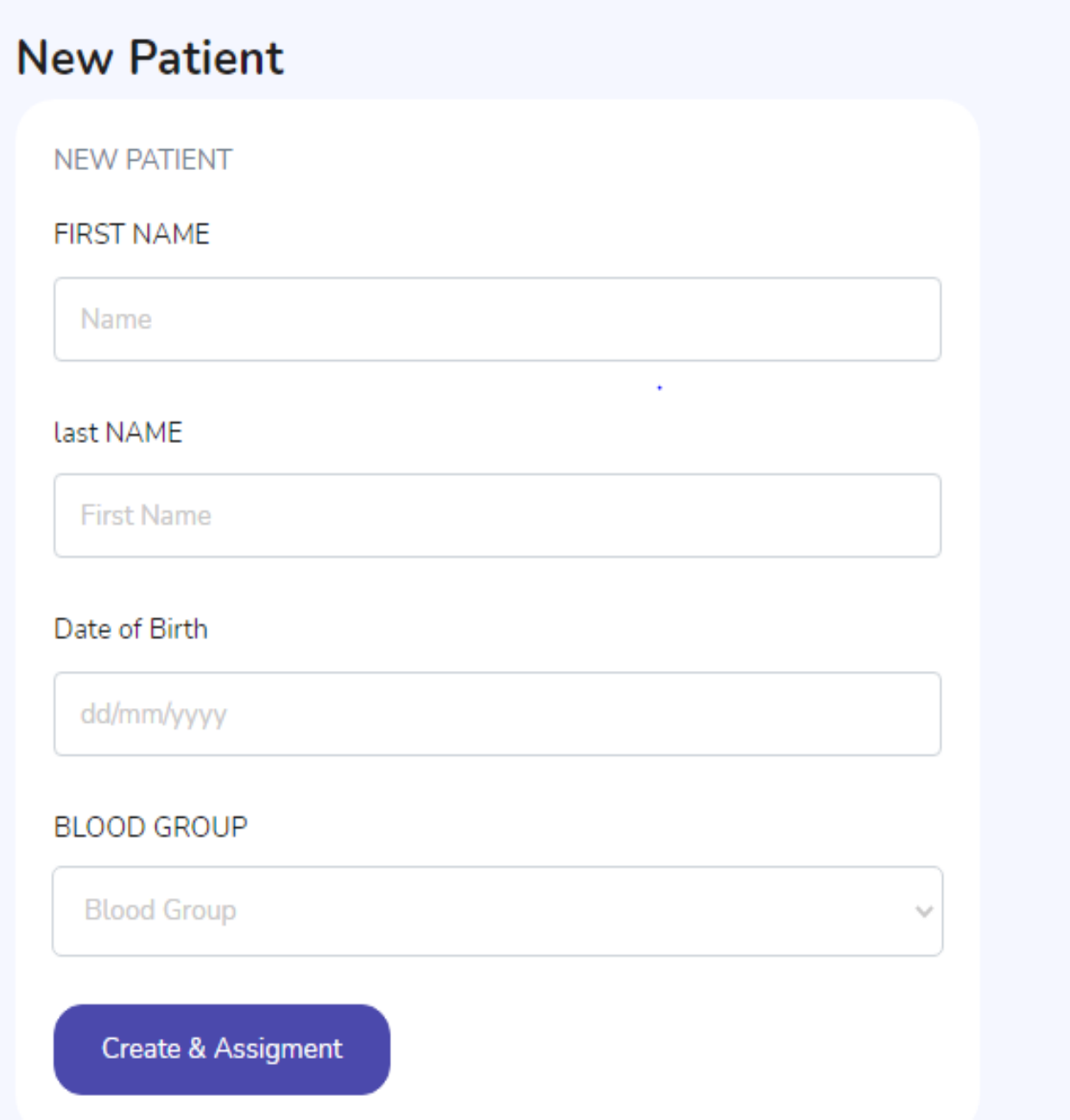
III.6. Interfaces de gestion des patients « création, affectation »

Dans ces interfaces, le secrétaire médical peut créer et affecter le patient au lit d'hôpital intelligent libre selon le cas de patient.

Le secrétaire médical peut rechercher sur un service ou une salle pour voir les détails du lit d'hôpital intelligent.

III.6.1. Interface de création des patients

La figure IV.10 représente l'interface de création des patients.



. **Figure IV.10.** Interface de création des patients.

III.6.2. Interface d'affectation des patients au lits concernant

La figure IV.11 représente l'interface d'affectation des lits.

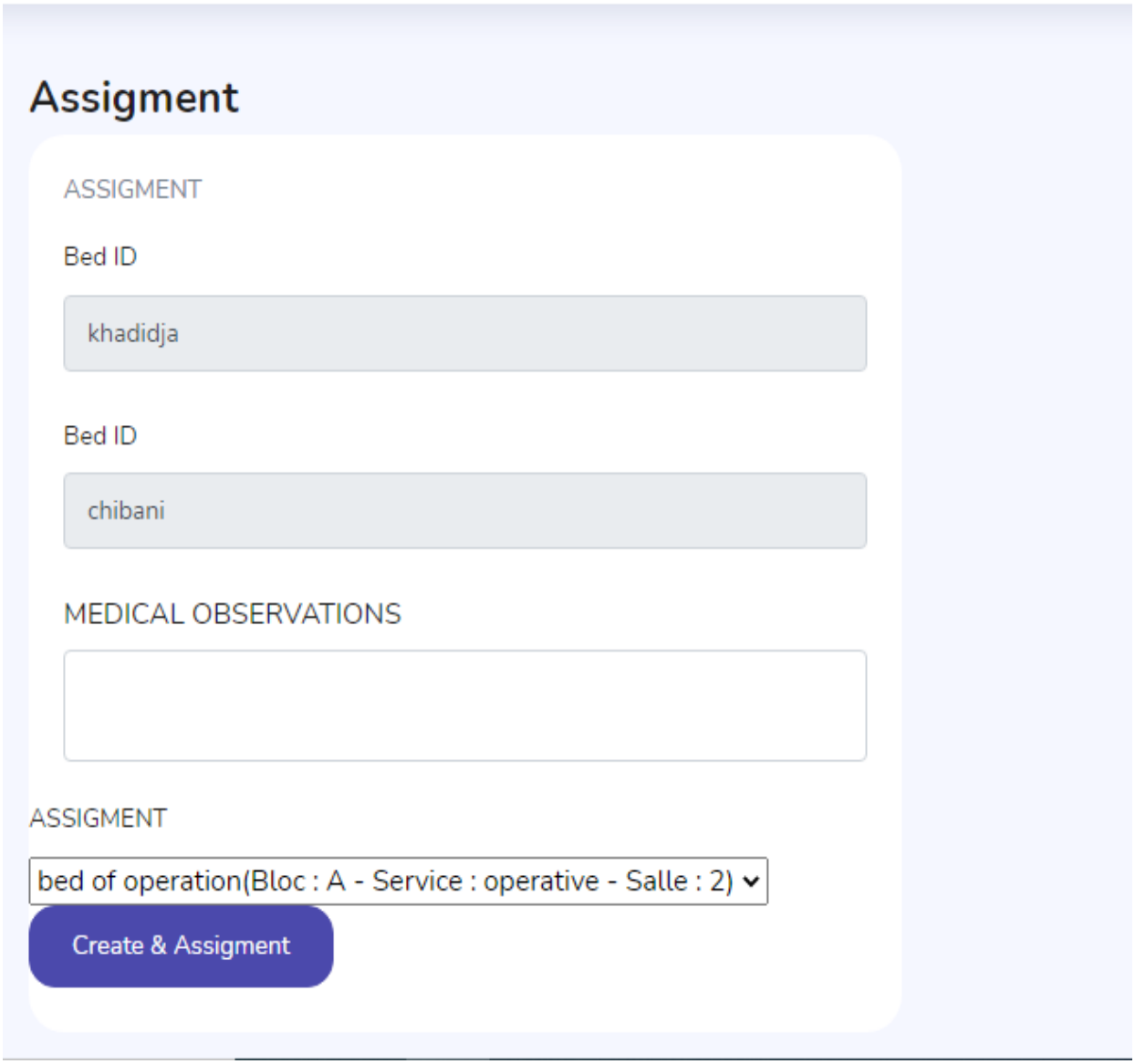


Figure IV.11. Interface d'affectation des lits intelligent pour les patients.

III.7. Interface de supervision et contrôle des patient « Consultation » « plan médical »

Cette interface s'affiche lorsque le médecine ou l'infirmier choisit ou recherche un lit d'hôpital intelligent pour superviser et consulte et contrôler le lit.

La figure IV.12 représente l'interface de consultation.

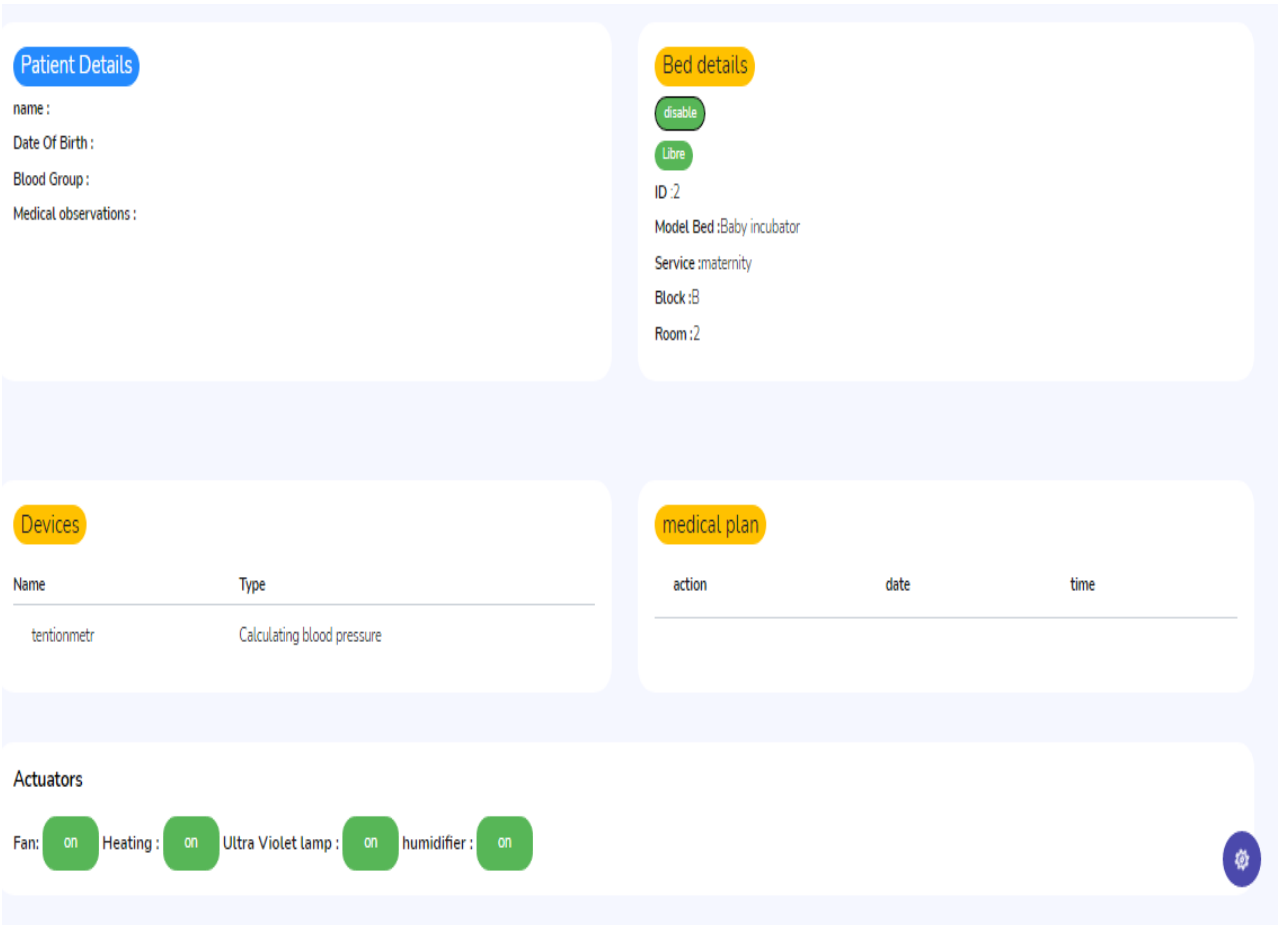


Figure IV.12. Interface de consultation.

La figure IV.13 représente l'interface de plan médical

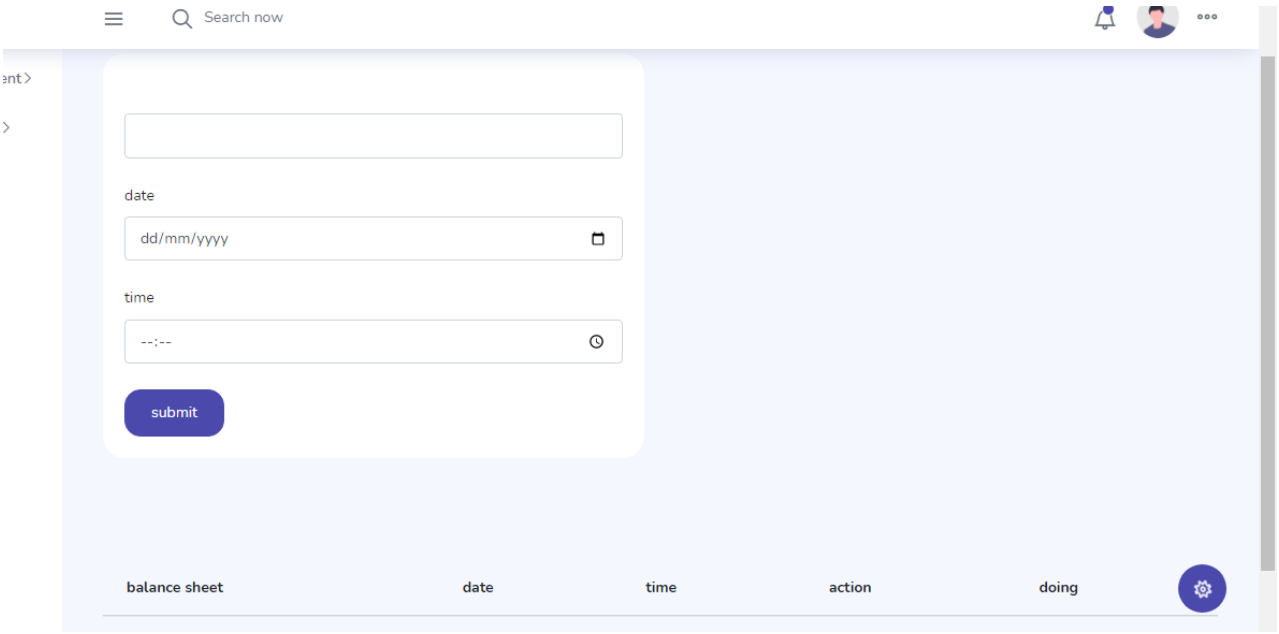


Figure IV.13. Interface de plan médical.

IV. Conclusion

Nous avons présenté en détails la réalisation de la solution en décrivant comment créer et connecter un lit d'hôpital intelligent avec son jumeau numérique, Et dans la phase réalisation nous avons décrit les différentes fonctions de l'application web mises en pratique avec quelques interfaces pour bien illustrer le projet.

Conclusion Général

Conclusion Générale

Dans ce travail, on s'est intéressé à la proposition d'une plateforme basée sur les micro-services et les jumeaux numériques pour la supervision et le contrôle des patients au niveau d'un hôpital. L'idée c'était de se baser sur les lits d'hôpital connectés pour collecter les données des patients et les présenter aux praticiens de la santé qui peuvent à leurs tours contrôler et commander l'état de santé des patients.

La plateforme a été basée sur un modèle de jumeau numérique que nous avons proposé pour les lits d'hôpital. Ce modèle permet de stocker des données collectées à travers les différents capteurs implantés sur le lit et les matériels médical aux connectés à ce dernier et les lier au patient pour capter son état de santé. L'implémentation des jumeaux numérique s'effectue en utilisant la plateforme Eclipse Ditto.

La plateforme a été construite comme un ensemble de micro-services qui sont les jumeaux numériques des lits d'hôpital et les différentes applicatifs permettant aux praticiens de santé de superviser les patients. Ces micro-services communiquent à travers le gestionnaire d'évènements apache Kafka. Le gestionnaire Kafka permet aussi de capter des évènements issus du système physique (le lit d'hôpital). La liaison des lits d'hôpital à leurs jumeaux numériques s'est implémentée par l'outil Node-Red.

La plateforme offre ses services à ces acteurs au moyen d'une application web que nous avons conçue et développée. La conception de cette application a été présentée sous forme de diagramme SysML tout en détaillant ses interactions avec les acteurs, les jumeaux numériques et les lits physiques. Le développement a été réalisé sous un environnement utilisant des conteneurs docker et s'est sanctionné par un Framework qui nous a permis de tester l'implémentation de la solution proposée. Ce test, dans le manque d'une connexion réelles aves des lits physiques, a été effectué en se basant sur une simulation que nous avons préparée au moyen de l'outil Node-Red. La connexion réelle pourra être facilement effectuée en connectant le système physique à son jumeau numérique à travers Node-Red.

A travers l'application web développée les acteurs de santé pourront superviser et contrôler les différentes mesures concernant l'état de santé du patient et pourront suivre son évolution et ses demandes et même gérer le plan médical qui doit être effectue pour sa guérison.

Enfin, cette expérience nous a été bénéfique et très utile, car elle nous a permis de savoir de nouvelles notions d'une part, et d'enrichir nos connaissances avec les outils des technologies nouvelles d'autre part.

Comme perspectives à ce qui a été réalisé, il est important d'évaluer réellement la solution développée au niveau d'un hôpital et de l'enrichir avec d'autres fonctionnalités avancées permettant une prise de décision efficace en utilisant des techniques d'intelligence artificielle.

Bibliographie

Bibliographie

[1] https://www.orma.fr/les-objets-connectes-presentationg-exemples-et-risques/#:~:text=Les%20objets%20connect%C3%A9s%20sont%20des%20capteurs%20ou%20des,et%20recevoir%29%20des%20donn%C3%A9es%20%C3%A0%20travers%20un%20r%C3%A9seau.

[2] https://www.a2-gestion.fr/c-est-quoi-les-objets-connectes/

[3] https://www.votre-it-facile.fr/usages-objets-connectes-en-medecine-iot/

[4] https://www.avast.com/fr-fr/c-what-is-the-internet-of-things#topic-1

[5] Comment fonctionne l'Internet des objets: protocoles et technologies liés à l'IdO - Blog La Triveneta Cavi

[6] https://www.futura-sciences.com/tech/definitions/internet-internet-objets-15158/

[7] https://www.zibtek.com/blog/iot-architecture/#:~:text=Even%20though%20there%27s%20no%20single%20IoT%20architecture%20that%27s,It%20proposes%20three%20layers%3A%20Perception%2C%20Network%2C%20and%20Application.

[8] https://www.ibm.com/fr-fr/topics/what-is-a-digital-twin

[9] https://www.challenge.org/insights/what-is-digital-twin/

[10] https://www.atriainnovation.com/en/digital-twins-what-are-they-advantages-and-applications/#:~:text=Digital%20twins%20have%20numerous%20advantages%20since%20they%20improve,defects%20and%20shortens%20manufacturing%20time%20among%20other%20things.

[11] Jumeaux numériques et gestion des actifs : pourquoi, comment? (createch.ca)

[12] Eclipse Ditto™ • open source framework for digital twins in the IoT

[13] HTTP Protocol - Definition & How it Works - ExtraHop

[14] What is HTTP (w3schools.com)

[15] WebSocket – communication protocol for real-time web applications - IONOS

[16] AMQP is the Internet Protocol for Business Messaging | AMQP

[17] What is AMQP Protocol ? How AMQP Protocol Works (iotboys.com)

[18] Node-RED: Lecture 1 – A brief introduction to Node-RED – Node RED Programming Guide (noderedguide.com)

[19] Node-RED (nodered.org)

[20] https://www.automation-sense.com/blog/automatisme/node-red-francais.html

-

[22] https://www.guru99.com/microservices-tutorial.html#:~:text=Microservice%20Architecture%20is%20an%20architectural%20development%20style%20that,arrange%20applications%20as%20a%20loosely%20coupled%20service%20collection.

[23] https://www.okta.com/blog/2021/02/microservices/

[24] https://www.redhat.com/fr/topics/api/what-is-a-rest-api

[25] https://www.ibm.com/cloud/learn/rest-apis

[26] https://www.astera.com/type/blog/rest-api-definition/#REST-API-Design-Principles

[27] What is Apache Kafka? | AWS (amazon.com)

[28] Microservices & Apache Kafka: Key Concepts, Benefits, and Use Cases | UK (confluent.io)

[29] What is a Bus? - Definition from Techopedia

[30] What is event bus in Microservices? (askinglot.com)

[31] CPU Organization (careerride.com)

[32] What is BUS? | Types of Computer Bus - Computer Notes (ecomputernotes.com)

[33] https://www.eclipse.org/ditto/

[34]Comment se compose un système IoT ? (connectwave.fr)

[35] https://www.hklaw.com/-/media/images/services/practices/healthcare/digital-healthcare/digitalhealthlawattorneybanner.jpg

[36] https://blog.bosch-si.com/wp-content/uploads/ditto-architecture-overview-1136x1136.png

[37] https://www.eclipse.org/ditto/images/pages/protocol/ditto-twin-channel.png

[38] https://www.ionos.it/digitalguide/fileadmin/DigitalGuide/Screenshots_2020/diagram-of-http-communication-process-it.png

[39] https://th.bing.com/th/id/OIP.nJJkjKT3bO1GUgdHp_HgQgHaEK?pid=ImgDet&rs=1

[40] https://www.reydes.com/d/sites/default/files/iblog/2017/Abr/websocket_01.jpg

[41] https://epsagon.com/wp-content/uploads/2020/05/Screen-Shot-2020-04-18-at-7.32.50-PM@2x.png

[42] https://www.oreilly.com/library/view/hands-on-restful-api/9781788992664/assets/894615f0-631a-4b3a-a38b-7b4d5750e9ea.png

[43] https://blog.kakaocdn.net/dn/nalfp/btqCU6DNBvD/o1kNIf3CXjWfkLiybkLFi0/img.png

[44]