

Avant-propos

Nom et prénom :

Hamza H'SAIN

Intitulé du travail :

Mise en œuvre d'un système de suivi du diabète à base d'IoT.

Établissement d'accueil :

Groupe SQLI - Agence OUJDA -

Coordonnées de l'établissement d'accueil :

SQLI Maroc Bd Mohammed VI, Hay El Qods, Complexe Technologique

Standard : +212 5 36 68 29 07 Web : <http://www.sqli.com>

Nom et prénom de l'encadrant du projet dans l'établissement d'accueil :

Mr Abdelkader RHOUATI

Nom et prénom de l'encadrant universitaire du projet:

Pr. Youssef BALOUKI

Date de début de stage :

15/02/2015

Date de fin de stage :

31/08/2015

Dédicace

À celle qui m'a appris à graver sur les rochers mon nom, à dépasser mes faiblesses et à surmonter mes maux ...

À celle qui m'a indiqué la bonne voie en me rappelant que la volonté fait toujours les grands hommes...

A ma mère

À celui qui a attendu avec impatience les fruits de sa bonne éducation et qui as toujours cru en ma personne

A mon père

À mon cher frère Omar

Aucune dédicace aussi parfaite ne pourra exprimer mon amour et ma gratitude envers vous...

À mes tantes et oncles

À mes cousines et cousins

À toute ma famille

À tous mes ami(e)s

Je vous remercie pour tous les instants inoubliables qu'on a passés ensemble ...

À tous ceux qui m'aiment, Je dédie ce modeste travail...

Hamza

Remerciement

On ne saurait commencer ce mémoire sans remercier ALLAH le tout-puissant, le tout miséricordieux, qui nous a donné grâce et bénédiction pour mener à terme ce projet.

Il m'est agréable de m'acquitter d'une dette de reconnaissance auprès de toutes les personnes qui ont contribué de près ou de loin à l'aboutissement de ce projet.

Je tiens à adresser mes plus vifs remerciements à Mr Abderrahmane KODADE, directeur de la société SQLI Oujda, de m'avoir offert l'opportunité d'effectuer mon projet de fin au sein de SQLI.

Je remercie également Monsieur Abdelkader RHOUATI, Manager du Skill Center Mobile & IOT et Monsieur Florian SPLENDIDO, Practice Leader IoT, qui ont supervisé mon stage au jour le jour, pour l'attention, l'aide, les conseils et la confiance qu'ils m'ont accordée tout au long de mon stage.

Je tiens à remercier tout particulièrement et à témoigner toute ma reconnaissance à tous les membres de l'équipe Mobile & IOT qui ont contribué, par leur disponibilité et leur bonne humeur, à rendre mon stage enrichissant et motivant.

L'aboutissement de ce travail me donne également l'occasion d'exprimer ma sincère reconnaissance à Monsieur Pr. Youssef BALOUKI, pour son encadrement et ses remarques constructives durant toute la période de la rédaction de ce rapport.

J'exprime mon profond respect et gratitude à tout le corps professoral et administratif de la faculté des science et techniques de SETTAT pour leurs efforts.

Résumé

Ce document synthétise le travail effectué, dans le cadre de note projet de fin d'études, au sein de SQLI Oujda, ayant comme but la conception et la mise en place d'un système de suivi du diabète à base d'IoT et Microsoft azure.

Pour mener à bien ce projet, nous avons commencé par une étude d'existant, suivie d'une analyse approfondie du projet dans le but d'identifier les besoins fonctionnels et non fonctionnels, auxquels la solution doit répondre.

Dans une seconde étape nous avons mené une étude conceptuelle qui a été traduite en diagrammes UML.

A la lumière de ces résultats nous avons mené la troisième et dernière étape qui comprend la mise en œuvre de système.

Abstract

This document is a summary of our graduation project, our training took place at SQLI Oujda, having as goal the design and implementation of a monitoring system for diabetes based IoT and Microsoft Azure.

To achieve this purpose, we started by a study of existing, then, a depth analysis of the project was conducted in order to identify the functional and non-functional requirements to be met by the solution.

The second step is designing the project witch concerns the conceptual design using object modeling language UML.

Thanks to the obtained results, we conducted the third and last part which involves the implementation of the application.

Liste des figures

Figure 1: Carte de principales agences du groupe SQLI	15
Figure 2: Les clients de SQLI.....	16
Figure 3: Évolution du chiffre d'affaires (en M€) de SQLI entre 2007 et 2012	17
Figure 4: Les cinq niveaux de CMMI	18
Figure 5: Cycle de vie d'un projet chez le groupe SQLI	21
Figure 6: Vue synthétique du processus Scrum	24
Figure 7: Planification du projet.....	27
Figure 8: Diagramme de gant.....	28
Figure 9: Architecture de solution IoT	31
Figure 10: Principe de fonctionnement du protocole CoAP.....	34
Figure 11: Principe de fonctionnement du protocole MQTT	35
Figure 12: Principe de fonctionnement du protocole AMQP.....	36
Figure 13: Architecture AWS IoT.....	39
Figure 14: Architecture Iot Hub	41
Figure 15: Diagramme de cas d'utilisation pour le patient	55
Figure 16: Diagramme de cas d'utilisation pour le système	55
Figure 17: Diagramme de classe	59
Figure 18: Diagramme de séquence relative au use cas consulter graphe.....	60
Figure 19: Diagramme de séquence relatif au use cas inscription.....	60
Figure 20: Diagramme de séquence relative au use cas inscription	61
Figure 21: Diagramme de séquence relative au use cas authentification	61
Figure 22: Architecture technique de notre projet.....	65
Figure 23: Le Patron de conception MVC	67
Figure 24: Le patron de conception MVVM.....	68
Figure 25: Les composants du patron de conception MVVM.....	68
Figure 26: Exemple des tâches affectées dans Trello.....	71
Figure 27: Exemple de gestion de notre projet avec sourceTree.....	73
Figure 28: Taux de cholestérol par temps	74
Figure 29: Taux de glucose et insuline par temps	75
Figure 30: l'évolution de masse corporelle par temps	75
Figure 31: réception du deviceId et paramètre de connexion.....	75
Figure 32: Device explorer.....	76
Figure 33: Simulation d'envoi des données de télémétrie	76
Figure 34: Réception des données.....	76
Figure 35: Réception des données.....	77
Figure 36: interface d'accueil	77
Figure 37: Historique.....	77
Figure 38: Interface d'assistant	78
Figure 39: page de chat	78
Figure 40: Calcul de Hb1AC	79
Figure 41: interface de réglages	79
Figure 42: Modifier âge.....	80
Figure 43: Exporter les données	80

Liste des abréviations

IOT	Internet of Things
e-business	Electronic Business
SAP	Systems, applications, and products for data processing
BI	Business Intelligence
R&D	Research and Development
CMMI	Capability Maturity Model Integration
CMMI-DEV	CMMI for Development
CMMI-SVC	CMMI for Services
XP	Extreme programming
IdO	Internet des objets
API	Application Program Interface
Rest	Representational State Transfer
CoAP	Constrained Application Protocol
MQTT	Message Queuing Telemetry Transport
IETF	Internet Engineering Task Force
AMQP	Advanced Message Queuing Protocol
HTTP	Hypertext Transfer Protocol
M2M	Machine To Machine
QoS	Quality of service
MQTT-SN	MQTT for Sensor Network
AWS	Amazon Web Services
IFTTT	If this then that
SDK	software development kit
TLS	Transport Layer Security
JSON	JavaScript Object Notation
UWP	Universal Windows Platform
UML	Unified Modeling Language
NoSql	Not only SQL
MVC	Modèle-Vue-Contrôleur
MVVM	Model View-View Model

MSIL	Microsoft Intermediate Language
MFC	Microsoft foundation class
IaaS	Infrastructure as a Service
PaaS	Platform as a service

Table des matières

Introduction générale.....	12
Chapitre 1.....	14
Cadre général du projet.....	14
I. Présentation de l'organisme d'accueil	15
1. Groupe SQLI	15
2. Clients SQLI	15
3. Groupe SQLI en quelques chiffres	16
4. SQLI Oujda	17
5. Qualité de service	17
5.1. Le modèle CMMI	17
5.2. Adoption du modèle CMMI par SQLI	19
5.3. Le nouveau concept SKILLS	20
II. Cadre général du projet	20
1. Présentation du projet	20
2. Comment SQLI mène-t-elle le projet ?	21
3. Méthodologie de travail	23
4. Pilotage et planification du stage	25
Conclusion	28
Chapitre 2.....	29
Internet des objets & Benchmarking	29
I. Généralité sur l'IOT	30
1. Définition	30
2. Architecture de solution IoT	30
3. les protocoles dédiés pour l'IoT	33
3.1. Architecture Request/Response	33
3.2. Architecture Publish/Subscribe	34
II. Étude et benchmarking des solutions IoT	37
1. AWS IoT	39
2. Azure IoT Hub	41
3. Résumé de l'étude comparatif pour les deux solutions	43
Conclusion	45
Chapitre 3.....	46

Analyse et conception.....	46
I. Analyse des besoins	47
1. Expression du besoin	47
1.1. BackLog	48
1.2. Référentiel des exigences:	51
II. La phase de conception :	54
1. Identification des acteurs :	54
2. Diagrammes des cas d'utilisation	55
2.1. Module AUTH : Authentification	56
2.2. Module consultation des graphes	57
2.3. Module envoi et réception des événements	58
3. Diagramme de classe	58
4. Diagramme de séquence	60
Conclusion	62
Chapitre 4.....	63
Mise en œuvre et implémentation	63
I. Architecture technique	64
II. Architecture applicative	65
1. MVC :	66
2. MVVM :	67
III. Outils et technologie de développement :	69
IV. Présentation de l'application réalisée	74
1. Partie Cloud	74
2. Partie Envoi et réception des données	75
3. Partie mobile	77
Conclusion.....	81
Conclusion Générale	82
Webographie.....	84

Introduction générale

Pour le grand public, la définition des objets connectés est simple c'est prendre des objets de tous les jours : fourchettes, toasteurs, portes, réfrigérateurs, voitures... et de les barder de capteurs et de circuits électroniques pour les rendre connectés à un réseau (internet ou autre), leurs permettant ainsi de communiquer entre eux ou avec nous.

Concrètement il s'agit de milliards d'objets connectés et potentiellement intelligents, qui communiqueraient pour transmettre à des systèmes d'information distants (mobile, tablette, Cloud ou Datacenter par exemple) un état, une consommation, une position, une température, une alerte, une vitesse, etc. Tous les appareils du quotidien sont susceptibles d'être des objets connectés à condition de leur insuffler cette capacité. Pour cela il faut donc réunir plusieurs ingrédients : l'objet, l'idée ou le besoin, les capteurs et le software.

Au-delà des prévisions de croissance du marché des IoTs, c'est une nouvelle révolution qui est engagée, qui permettra de nouveaux gains de performance et de productivité, et qui fera évoluer les business modèles des entreprises grâce à la création de nouveaux services et de nouveaux usages.

Cette révolution combine trois éléments : le réseau, des objets physiques (appareils mobiles, capteurs et autres objets connectés...) et un troisième élément, moins visible, mais essentiel : les services (Prototypage, traitement de données, conseil, déploiement Cloud, sécurité, audit...).

En effet SQLi se positionne sur ce troisième axe, en proposant une offre packagée (voire clé en main). Avec sa diversité technique et son expérience sur les métiers du E-commerce, le digital retail ou encore la santé.

C'est dans ce cadre que s'inscrit notre projet, l'idée est de mettre en place un analyseur de glycémie, avec un bracelet connecté, une balance, thermomètre ou un tensiomètre, vous pouvez réaliser vos mesures à domicile et effectuer un suivi depuis votre tablette et mobile. En suite libre à vous de communiquer les informations collectées à un médecin qui pourra ajuster les soins ou établir un diagnostic en fonction des résultats obtenus.

La mise en place de ce système fait l'objet de notre projet, et pour mieux retracer les étapes de production, le présent mémoire s'articule autour des chapitres suivants:

Tout d'abord nous présenterons dans le premier chapitre le groupe SQLI, et plus précisément l'agence SQLI Oujda, lieu de notre stage. Nous allons par la suite présenter le contexte métier du projet. Nous finirons cette partie par la méthodologie et l'outil de conduite du projet utilisé.

Quant au troisième chapitre seront focalisées sur l'internet des objets en passant tout d'abord par sa définition ses applications et son architecture. Ensuite nous illustrons dans un premier temps, à recenser les leaders des outils IoT disponibles dans le marché et de présenter une description détaillée des solutions retenues lors de notre « Benchmarking ».

Le troisième chapitre, décrit la phase de conception du projet. Elle comporte une étude statique du système ainsi qu'une vue dynamique représentant les interactions, des acteurs avec le système étudié.

Dans le quatrième chapitre, nous allons aborder la réalisation et la mise en œuvre du projet. Dans cette partie nous détaillerons l'architecture technique, l'architecture applicative, et nous finirons par les fonctionnalités mises en œuvre.

Nous clôturons ce mémoire par une conclusion générale suivie d'un ensemble de références webographiques et des annexes représentant des informations complémentaires sur les différents éléments traités dans le rapport.



Chapitre 1

Cadre général du projet

L'objectif de ce chapitre est de situer le stage de fin d'études par rapport à l'environnement global dans lequel le travail a été effectué. Nous présentons dans un premier temps l'organisme d'accueil et ces activités. Ensuite, nous introduisons de manière générale le projet. Enfin, nous expliquons la méthodologie de travail adoptée pour mener le projet à bien.

I. Présentation de l'organisme d'accueil

Nous donnons dans la première partie de ce chapitre, quelques informations sur le groupe SQLI et décrivons ses secteurs d'activités et ses différents pôles.

1. Groupe SQLI

SQLI est une société française créée en 1990, pour accompagner les entreprises dans l'utilisation des nouvelles technologies. Elle s'est spécialisée dans la réalisation des systèmes d'informations de la nouvelle génération. Elle est organisée en agences de proximité, afin de conserver le maximum de réactivité face aux besoins de ses clients. En prenant en compte, au sein même de son organisation, les préoccupations du tissu économique régional, SQLI offre une approche sur mesure aux enjeux spécifiques des entreprises.

SQLI est composée de 21 agences dont 10 en France et 11 internationales: Suisse, Luxembourg, Belgique, Pays-Bas, Espagne, Maroc (Rabat, Oujda), Canada et Singapour (figure 1).



Figure 1: Carte de principales agences du groupe SQLI.

2. Clients SQLI

Tout en développant son activité, SQLI veille à maintenir une grande diversification de sa clientèle et des secteurs d'activités auxquels elle s'adresse, de façon à contenir le risque de concentration sur un nombre restreint de clients.

Ainsi, SQLI compte plus de 1200 clients, issus de tous les secteurs d'activité à savoir:
(Voir figure 2)

- + L'industrie.
- + Les services.
- + Les banques et assurances.
- + Les administrations et services publics.
- + La distribution.
- + L'immobilier.
- + Les transports.
- + Les télécoms.

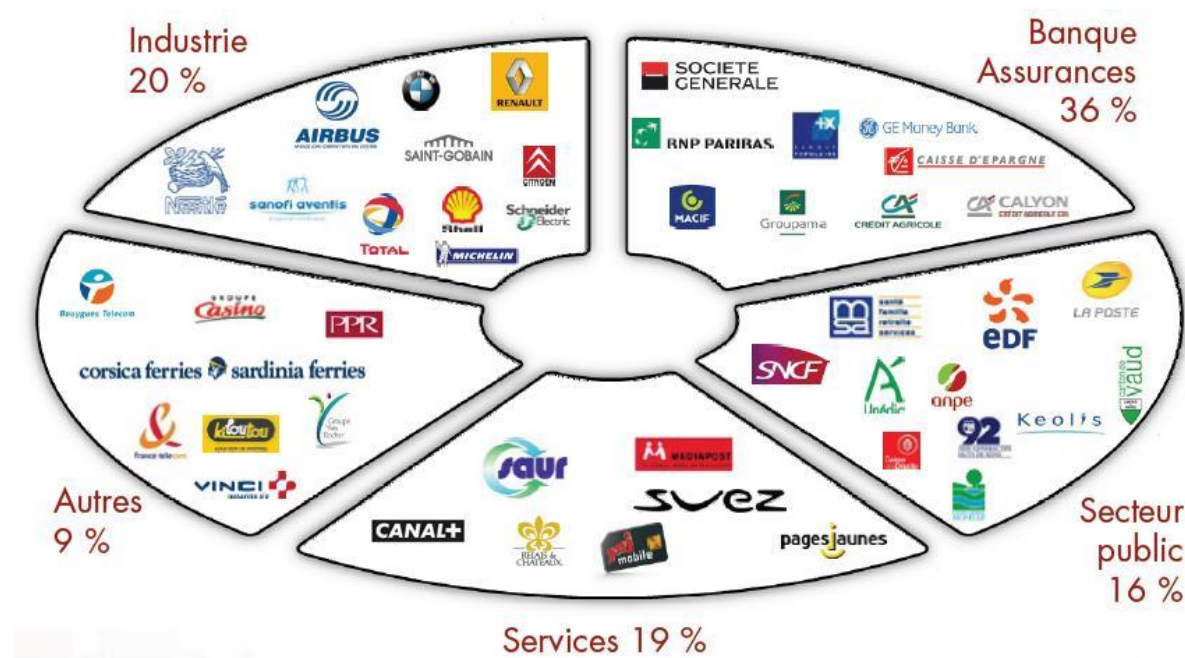


Figure 2: Les clients de SQLI.

3. Groupe SQLI en quelques chiffres

Depuis sa création en 1990, SQLI assoit son développement sur une expertise technologique de pointe et sur une politique intense de recherche et de développement ce qui lui a permis d'être le N°1 des « pure players » e-business (Usages & technologies internet, SAP, BI,...). Le Groupe SQLI a réalisé en 2014 un chiffre d'affaires de 161,4 M€.

La figure 3 présente l'évolution de quelques chiffres clés de la société SQLI :

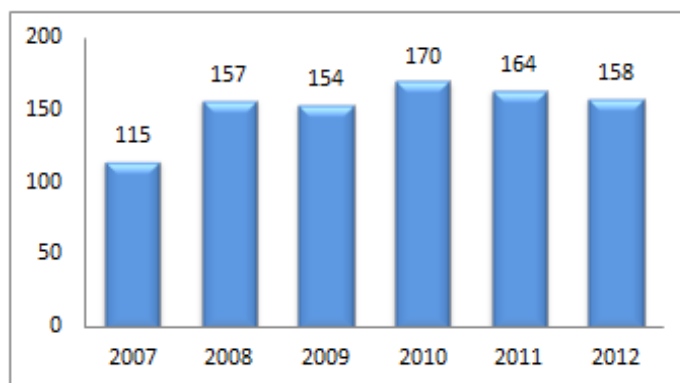


Figure 3: Évolution du chiffre d'affaires (en M€) de SQLI entre 2007 et 2012

4. SQLI Oujda

Dans le cadre de sa stratégie de développement alliant taille et densité technologique, le Groupe SQLI s'est doté de pôles spécialisés, disposant chacun d'une expertise technologique et/ou métier spécifique.

Pour accompagner le développement de son pôle dédié Open Source de Poitier, en septembre 2006, le Groupe SQLI ouvrait, en partenariat étroit avec l'Université Mohammed Premier, le tout premier centre de R&D / offshore entièrement dédié aux technologies Open Source.

Depuis, SQLI bénéficie, au sein même de l'Université, d'un fort potentiel de ressources de très haut niveau de qualité et de locaux entièrement équipés des technologies les plus avancées. En contrepartie, les experts du Groupe interviennent dans le cadre de formations spécifiques, telles que CMMI, et contribuent à l'enrichissement des cursus scolaires des modules complémentaires.

5. Qualité de service

5.1. Le modèle CMMI

CMMI, que l'on pourrait traduire par "Modèle de maturité logicielle" est le garant de la qualité des projets chez SQLI. C'est un modèle pour les entreprises désireuses d'accroître leur compétitivité en améliorant les processus liés au développement logiciel.

Les objectifs du modèle CMMI sont notamment de :

- + Définir un cadre décrivant les éléments clés d'un processus logiciel efficace.

- + Décrire les améliorations évolutives permettant le passage d'un processus improvisé vers un processus mature et discipliné.

- + Contenir des pratiques de planification, d'ingénierie, de gestion du développement et de maintenance.

- + Améliorer la capacité de l'entreprise à atteindre ses objectifs de coût, de délai, de fonctionnalités et de qualité des produits.

Le modèle CMMI, dans sa représentation étagée, est structuré en 5 niveaux de maturité (figure 4) correspondant à une amélioration du processus de développement logiciel. Pour atteindre un niveau, des objectifs, que l'on peut regrouper en trois catégories, doivent être satisfaits :

- + Les processus de gestion (planification, suivi, supervision, assurance qualité).
- + Les processus organisationnels (gestion des exigences, organisation des processus, formation).
- + Les processus d'ingénierie (test, développement, validation, configuration).

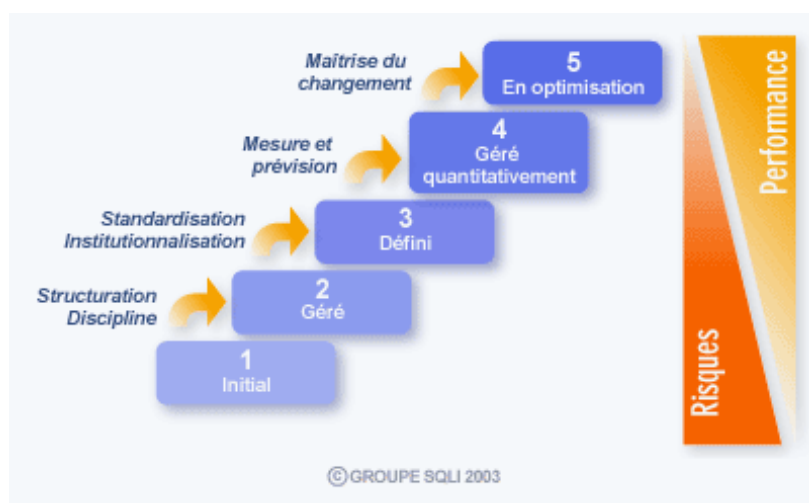


Figure 4: Les cinq niveaux de CMMI

Description des niveaux :

Niveau 1 – Initial : Au niveau 1, le processus de développement n'est pas défini. La réussite des projets dépend du savoir-faire de quelques personnes clés dans l'organisation. Il n'y a pas de formalisation du savoir-faire, des processus et pas de partage.

Niveau 2 – Reproductible : Les principaux processus de gestion du projet sont définis. Une gestion du projet élémentaire est définie pour assurer la planification puis le suivi des coûts, des délais et des exigences du projet.

Niveau 3 – Défini : Les processus définis au niveau 2 sur les projets sont généralisés à toute l'organisation. Tout processus projet est une instantiation du processus standard de l'organisation. Les processus sont affinés. Une attention particulière est portée aux processus d'ingénierie.

Niveau 4 – Contrôle : Les mesures sont mises en œuvre systématiquement sur chaque processus. Les données sont consolidées et exploitées.

Niveau 5 – Optimisé : Au niveau 5, les processus sont totalement maîtrisés. Les mesures sont exploitées. Cela permet d'anticiper les évolutions techniques des processus qui sont optimisés en permanence.

5.2. Adoption du modèle CMMI par SQLI

En génie logiciel, la qualité logicielle est une notion fondamentale essentiellement basée sur de nombreux indicateurs :

- + La complétude des fonctionnalités.
- + La précision des résultats.
- + La fiabilité.
- + La tolérance de pannes.
- + La facilité et la flexibilité de son utilisation.
- + La simplicité.
- + L'extensibilité.
- + La compatibilité et la portabilité.
- + La facilité de correction et de transformation.
- + La performance.

La qualité logicielle fait partie essentielle des priorités de SQLI, et c'est pour cette raison que cette dernière a opté pour la démarche qualité CMMI. En adoptant ses pratiques, SQLI augmente ses chances dans le marché et accroît sa compétitivité en offrant à ses clients une garantie de la qualité du produit final.

SQLI est certifiée Niveau 4 pour CMMI-DEV et Niveau 2 pour CMMI-SVC, elle est envoyée vers l'application des pratiques CMMI niveau 5. Elle s'oriente aussi vers la certification ISO 9001.

5.3. Le nouveau concept SKILLS

Afin de faciliter l'accès des collaborateurs aux informations relatives aux pratiques CMMI, SQLI a mis en place le portail Skills qui est un référentiel de processus, produits et outils pour les projets. Ces outils prennent généralement la forme soit de documents Word, utilisés comme modèles pour la rédaction par exemple de dossiers de spécifications ou de dossiers d'architecture, soit de feuilles Excel pour présenter par exemple des check-lists de revue de produits, pour organiser un suivi des actions ou plus généralement un suivi de projet, ou encore pour illustrer au travers de graphiques l'évolution d'indicateurs sur un projet.

II. Cadre général du projet

1. Présentation du projet

400 millions de personnes dans le monde vivent avec le diabète, qui est la septième cause de décès et génère des coûts de soins de santé de 245 milliards \$ par an aux États-Unis seulement. Ceux qui ont le type 1 sous forme de la maladie doivent surveiller attentivement leur taux de sucre dans le sang et ont constamment pensé à l'impact potentiel des facteurs aussi divers que ce qu'ils choisissent de manger et de faire, ou comment ils se sentent stressés. Cela ajoute généralement à moins de 180 décisions distinctes tout au long de la journée comme ils pensent à ce que les mesures qu'ils doivent prendre pour maintenir leur glycémie au bon niveau.

L'objectif de notre projet est de mettre en œuvre une application connectée conçue pour aider les personnes atteintes de diabète à mieux gérer la maladie. Ceci est la première solution mobile et IoT qui s'inscrit dans la gamme des solutions Idéo Santé réalisées par SQLI, ces dernières s'inscrivent dans une approche globale et industrialisée, résolument orientée qualité et repose sur un socle résolument ouvert pour une meilleure cohérence entre les SI de l'État, de l'Assurance Maladie, des professionnels et des établissements de Santé, notre solution rassemble les données médicales depuis des dispositifs personnels de santé et d'autres sources puis les analyse et diffuse à d'autres plateformes, l'application intègre également la technologie fournie par le géant des logiciels Microsoft Azure.

L'application présente l'information via mobile ou tablette, présentant des paramètres importants tels que la glycémie, l'utilisation de l'insuline ou nutrition et inclut une fonction d'encadrement qui donne des conseils. Il y a aussi une communauté en ligne sécurisée où les patients, les cliniciens et les soignants peuvent interagir via la messagerie privée ou partagent de messages. De cette façon, les patients peuvent obtenir une rétroaction en ligne et conseils de leur équipe de soins à l'aide de la combinaison des données et peuvent facilement partager des expériences.

L'obstacle majeur est la nécessité de gérer la sécurité et identités pour répondre à des exigences réglementaires rigoureuses sur la protection des données. Il n'est pas simplement une question de garder les données de chaque patient privées à eux et à leurs aidants. Un patient ne voudra pas partager toutes leurs données de santé.

2. Comment SQLI mène-t-elle le projet ?

SQLI dispose d'un référentiel projet disponible sur l'intranet SKILLS. Ce référentiel regroupe l'ensemble des processus d'ingénierie et informations relatives aux pratiques CMMI, de pilotage et de support mis en œuvre pour la réalisation d'un projet de développement.

La réalisation d'un projet est décomposée en 6 phases majeures :



Figure 5: Cycle de vie d'un projet chez le groupe SQLI

✕ Avant-vente

L'Avant-vente a pour objectif de répondre à un appel d'offre d'un client afin de réaliser un produit ou un service. Il doit donc permettre de comprendre les besoins du client, de déterminer les modalités de réalisation de ces besoins, d'estimer le travail à fournir ainsi que la charge et le délai. Tous ces éléments sont alors formulés dans la proposition commerciale qui est remise au client. Par la suite, des étapes de soutenance et de négociation commerciale pourront avoir lieu.

✕ Définition

La Définition a pour objectif principal de lancer le projet tant en interne qu'en externe au regard du besoin du client et de l'organisation à mettre en œuvre. D'un point de vue du besoin, l'objectif est de s'approprier et stabiliser l'ensemble des exigences (fonctionnelles, techniques...) du projet. D'autre part, cette phase permet également la mise en place de l'organisation du projet, qui couvre la planification globale, les grandes étapes de la vie du projet, la gestion des ressources impliquées, l'organisation des rituels de suivi, les risques, etc. À ce titre, c'est également l'occasion de mettre en œuvre les différents outils qui vont être utilisés par la suite.

✕ Conception

La Conception a pour objectif de définir la réponse à apporter aux besoins du client, en concevant de manière détaillée l'application à réaliser.

✕ Construction

Les objectifs de la Construction sont d'une part de développer l'application et d'autre part de vérifier sa conformité à la conception fonctionnelle et technique. La réalisation pourra être découpée en plusieurs itérations de développement de l'ensemble des modules fonctionnels et techniques, aboutissant à l'assemblage de l'application finale. Cette phase ne pourra être engagée qu'après avoir pris la décision de lancer les développements.

✕ Réception

La réception a pour objectifs de vérifier le bon fonctionnement de l'application dans l'environnement du client.

✕ Déploiement

Le Déploiement a pour objectif la mise en production de l'application, opération menée par le client avec l'assistance de SQLI.

Dans le cas d'un projet informatique, le choix d'un processus de développement, aussi contraignant peut-il paraître, est un atout permettant à tous les acteurs du projet de mener conjointement une action organisée selon des règles clairement exprimées.

3. Méthodologie de travail

Vu l'évolutivité et la taille moyenne du projet, nous avons adopté la méthodologie agile SCRUM.

✕ Méthode agile

Une méthode agile est une approche itérative et incrémentale, qui est menée dans un esprit collaboratif avec juste ce qu'il faut de formalisme.

Elle génère un produit de haute qualité tout en prenant en compte l'évolution des besoins des clients. Il existe plusieurs méthodes : Scrum (la plus populaire), XP, le Lean software développement ... etc.

Valeurs caractéristiques d'un projet agile :

- + Les individus et les interactions plutôt que les processus et les outils.
- + Produit qui fonctionne plutôt qu'une documentation exhaustive.
- + La collaboration avec le client plutôt que la contractualisation des relations.
- + L'acceptation du changement plutôt que la conformité aux plans.

Les 12 principes d'une méthode agile :

- + Satisfaire le client est la priorité.
- + Accueillir les demandes de changement « à bras ouverts ».
- + Livrer le plus souvent possible des versions opérationnelles de l'application.
- + Assurer une coopération permanente entre client et équipe projet.
- + Construire des projets autour d'individus motivés.
- + Privilégier la conversation en face à face.
- + Mesurer l'avancement du projet en termes de fonctionnalités de l'application.
- + Faire avancer le projet à un rythme soutenable et constant.
- + Porter une attention continue à l'excellence technique et à la conception.
- + Favoriser la simplicité.
- + Responsabiliser les équipes.
- + Ajuster, à intervalles réguliers, son comportement et ses processus pour être plus efficace.

✕ L'agilité avec Scrum

SCRUM est un processus léger permettant de gérer et de contrôler le développement de produits logiciels. Mettant en œuvre des pratiques itératives et incrémentales, SCRUM est, en soi, une méthode efficace. C'est l'une des méthodes agiles les plus employées, car elle permet d'améliorer notablement la productivité tout en accélérant le retour sur investissement.

La méthode s'appuie sur le découpage d'un projet en incréments, nommés "sprint", ainsi que l'auto-organisation de l'équipe de développement. Les sprints sont de durée variable entre quelques heures et un mois (avec une préférence pour deux semaines). Chaque sprint commence par une estimation suivie d'une planification opérationnelle. Le sprint se termine par une démonstration de ce qui a été achevé, et contribue à augmenter la valeur d'affaires du produit. Avant de démarrer un nouveau sprint, l'équipe réalise une rétrospective : elle analyse ce qui s'est passé durant ce sprint, afin de s'améliorer pour le prochain (figure 6).

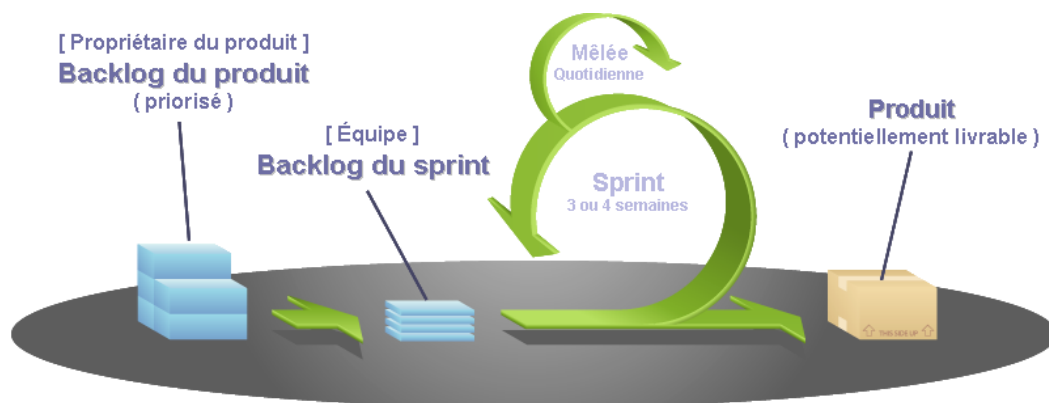


Figure 6: Vue synthétique du processus Scrum

Scrum nous permet d'inspecter rapidement et régulièrement (toutes les 1 à 4 semaines) un produit (partiel) qui fonctionne.

Le métier identifie les exigences et définit leurs priorités. L'équipe s'organise elle-même pour déterminer la meilleure façon de produire les exigences les plus prioritaires.

✕ Release

Pour améliorer la lisibilité du projet, on regroupe généralement des itérations en releases. Bien que ce concept ne fasse pas explicitement partie de Scrum, il est utilisé pour mieux

identifier les versions. En effet, comme chaque sprint doit aboutir à la livraison d'un produit partiel, un release permet de marquer la livraison d'une version aboutie, susceptible d'être mise en exploitation.

Il est intéressant de planifier à l'échelle d'un release, en répartissant les items du backlog de produit sur les sprints, en respectant leurs priorités. Bien entendu, ce qui est planifié au-delà du sprint courant peut changer à tout moment, rien n'est figé à l'avance.

✕ Sprints

À la fin du sprint, tout le monde se réunit pour effectuer la revue de sprint, qui dure au maximum 4 heures. L'objectif de la revue de sprint est de valider le logiciel qui a été produit pendant le sprint. L'équipe commence par énoncer les items du backlog du produit qu'elle a réalisés. Elle effectue ensuite une démonstration du logiciel produit. C'est sur la base de cette démonstration que le directeur du produit valide chaque fonctionnalité planifiée pour ce sprint.

Une fois le bilan du sprint réalisé, l'équipe et le directeur du produit proposent des aménagements sur le backlog du produit et sur la planification provisoire de la release. Il est probable qu'à ce moment des items soient ajoutés, modifiés ou réestimés, en conséquence de ce qui a été découvert.

4. Pilotage et planification du stage

La planification du projet est une phase importante d'avant-projet. Elle consiste à prévoir le déroulement de ce dernier tout au long des phases constituant le cycle de développement, et pour la bonne conduite de notre projet, nous avons élaboré un planning détaillé des tâches et des activités que nous serons amenés à réaliser au cours de notre projet de fin d'études.

Pour l'estimation de la charge du projet, nous avons suivi les normes standard utilisées dans la méthodologie de gestion de projets informatiques au sein de l'agence SQLI Oujda.

Nous avons commencé notre projet par une formation sur les outils de pilotage et les normes de qualités sur le site « Skills » cette formation a duré 5jours, puis une formation sur Git d'une journée et nous sommes passés à une auto formation sur les outils techniques: Microsoft Azure, Raspberry pi, et Android, cette dernière a pris plus de temps par rapport aux

précédentes, nous avons passé 20 jours afin de monter en compétences dans ces outils et langages.

Ensuite nous sommes passés à une phase du benchmark initiale l'idée c'est d'analyser les fonctions offertes par la solution azure avec un premier test de collecte de PC vers Azure et finalement nous avons travaillé sur la mise en œuvre de la solution.

Le diagramme de GANTT suivant montre les tâches effectuées et leur répartition dans le temps :

III	Task name	Start time	Duration day	Progress
▼	Estimation total	17/02/16	71	
▼	Projet Glycemapp	17/02/16	71	95%
▼	Formation	17/02/16	26	100%
	Formation Skills	17/02/16	5	100%
	Formation Git	25/02/16	1	100%
	Auto-formation Android ,Microsoft Azure,Raspberry pi	26/02/16	20	100%
▼	Itération 0	28/03/16	13	100%
	Initialisation des exigences	28/03/16	3	100%
	Analyse fonctionnelle	01/04/16	3	100%
	Analyse d'architecture	07/04/16	3	100%
	Conception graphique et ergonomique	14/04/16	1	100%
▼	Itération 1	15/04/16	18	100%
▼	Conception	15/04/16	7	100%
▼	Conception fonctionnelle	15/04/16	6	100%
	Rédaction du dossier de conception fonctionnelle détaillée	15/04/16	2	100%
	Maquettage des éditions papiers et reporting spécifiques	20/04/16	1	100%
	Modélisation des structures (modèle de classes, modèles conceptuel et physique	22/04/16	1	100%
▼	Conception technique	19/04/16	5	100%
	Définition de l'architecture technique et logicielle	19/04/16	1	100%
	Rédaction du dossier d'architecture applicative (composants logiciels, couches d	21/04/16	3	100%
▼	Construction	27/04/16	10	100%
▼	Mise en place du plateforme	27/04/16	2	100%
	Installation des services de l'environnement Azure	27/04/16	1	100%
	installation des logiciels de versionning	28/04/16	1	100%
▼	Développement	02/05/16	7	100%
	Envoi/Réception des message IOT	02/05/16	7	100%
	Stockage des données	02/05/16	3	100%
▼	Itération 2	10/05/16	12	71%
▼	Construction	10/05/16	12	71%
▼	Développement	10/05/16	12	71%
	Collecte des données par Stream Analytics	10/05/16	4	100%
	Présentation des télémétries dans Power BI	17/05/16	3	100%
	Version 1 de l'application mobile	10/05/16	11	66%
	Version 1 de l'application web	10/05/16	10	50%
	Mise en oeuvre du web service REST	23/05/16	3	100%

Figure 7:Planification du projet

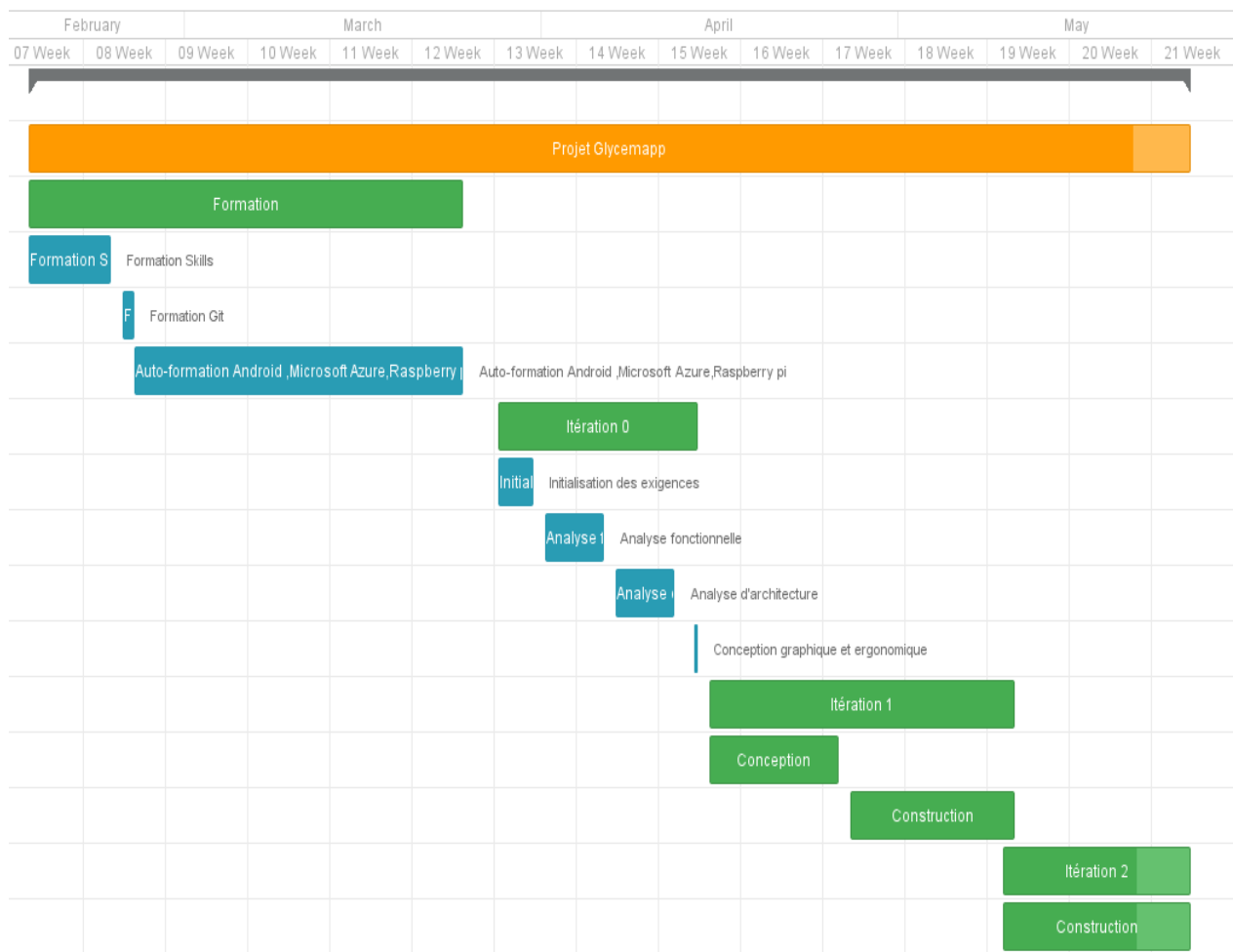


Figure 8:Diagramme de gant

Conclusion

Dans ce chapitre, nous avons dressé une vue globale sur l'organisme d'accueil : le groupe SQLI, ses métiers et sa démarche de travail. Puis nous avons présenté notre projet dans un contexte général et les différents objectifs fixés. Enfin, nous avons donné une idée globale sur la méthodologie de conduite du projet.



Chapitre 2

Internet des objets & Benchmarking

Dans ce deuxième chapitre, nous présentons quelques généralités sur le domaine d'internet des objets, puis nous allons analyser et comparer les plates-formes logicielles populaires dans le marché d'IoT.

I. Généralité sur l'IOT

1. Définition

L'internet des objets est un réseau de réseaux qui permet par des systèmes d'identifications complexes de transmettre des données entre objets physiques et virtuels. Ainsi Internet n'est plus qu'un réseau invisible mais il prend forme dans ces objets.

Les objets connectés émergent et nous les adorons, mais afin que l'évolution technologique soit optimale il faut que ces objets puissent interagir de manière autonome. Ainsi prenons un exemple de l'IdO dans la vie quotidienne, légèrement futuriste : Votre réveil connecté à votre mobile s'est programmé tout seul grâce à votre agenda électronique, ce même réveil a programmé le grille-pain pour que votre tartine saute en même temps que le café est servi. Votre voiture a programmé l'ouverture des portes de votre garage etc .. Ainsi Internet et les objets connectés ne forme plus qu'un ensemble à la fois physique et virtuel.

2. Architecture de solution IoT

Le diagramme suivant montre une architecture de solution IoT classique. Notez qu'il n'inclut le nom d'aucun service Azure, mais qu'il décrit les éléments clés dans une architecture de solution IoT générique. Dans cette architecture, les appareils IoT collectent des données qu'ils envoient à une passerelle cloud. La passerelle cloud met les données à la disposition d'autres services principaux, qui les traitent avant de les fournir à d'autres applications métier ou opérateurs humains par le biais d'un tableau de bord ou autre appareil de présentation.

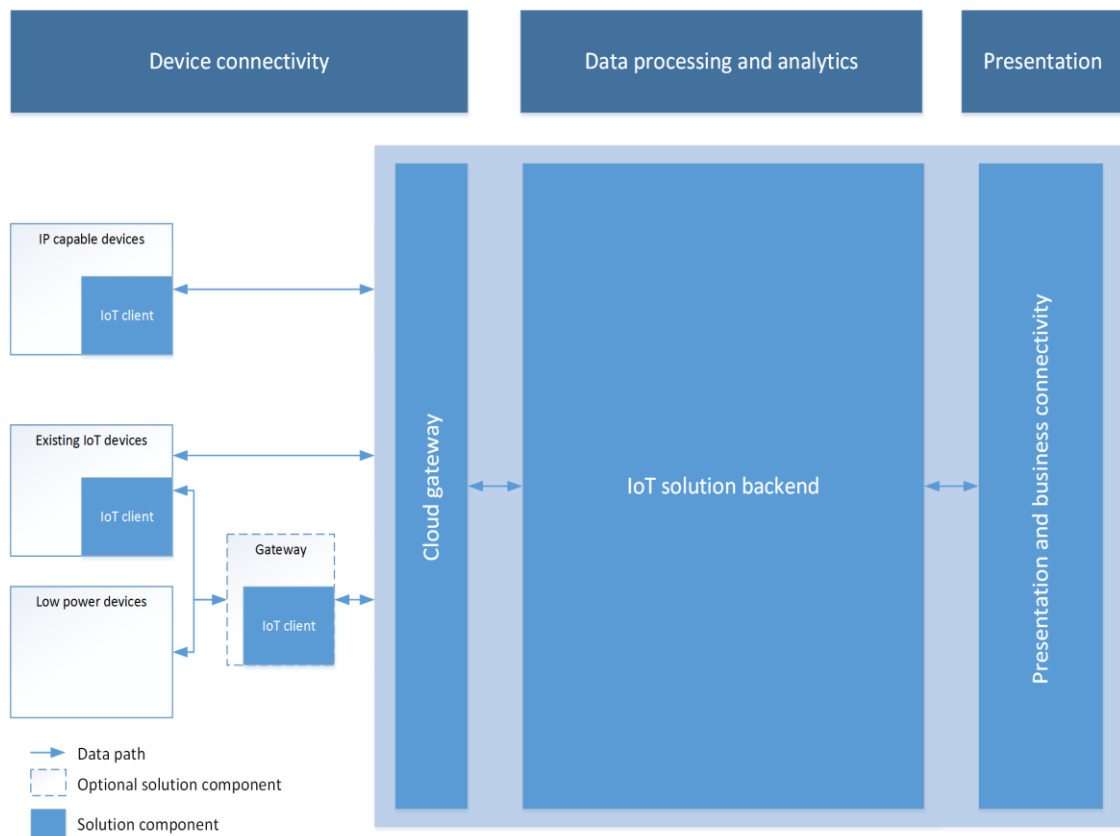


Figure 9: Architecture de solution IoT

✗ Connectivité des appareils

Dans cette architecture de solution IoT, les appareils envoient les données de télémétrie, comme par exemple des lectures de température, vers un point de terminaison du cloud pour qu'elles soient traitées et stockées. Les appareils peuvent également recevoir des commandes « cloud-à-appareil », et y répondre, en lisant les messages issus d'un point de terminaison du cloud. Par exemple, un appareil peut extraire une commande qui indique de modifier la fréquence à laquelle il échantillonne les données.

L'une des plus grandes difficultés dans les projets IoT consiste à pouvoir connecter de manière fiable et sécurisée des appareils au serveur principal de la solution. Les appareils IoT ont des caractéristiques différentes de celles d'autres clients tels que des navigateurs et des applications mobiles. Les appareils IoT:

- + sont souvent des systèmes intégrés, qui ne font appel à aucun opérateur humain.
- + peuvent être situés sur des sites distants avec un accès physique très coûteux.

- + sont accessibles uniquement via le serveur principal de la solution, Il n'existe aucun autre moyen d'interagir avec l'appareil.
- + peuvent avoir des performances et/ou des ressources de traitement limitées.
- + peuvent avoir une connectivité réseau intermittente, lente ou coûteuse.
- + peuvent nécessiter l'utilisation des protocoles d'application personnalisés, propriétaires ou spécifiques à un secteur.
- + peuvent être créés à l'aide d'un large éventail de plateformes matérielles et logicielles populaires.

Outre les exigences ci-dessus, n'importe quelle solution IoT doit également offrir des possibilités d'évolution, la sécurité et la fiabilité. Il en résulte un ensemble d'exigences de connectivité complexe, long à mettre en place avec des technologies traditionnelles telles que des conteneurs web et des courtiers de messagerie.

✗ **Traitement et l'analyse des données**

Dans le cloud, le serveur principal de solution IoT est l'endroit où se déroule la quasi-totalité du traitement des données dans la solution, notamment le filtrage et l'agrégation des données de télémétrie et leur routage vers d'autres services. Le serveur principal de solution IoT:

- + reçoit les données de télémétrie à l'échelle des appareils et détermine comment traiter et stocker ces données.
- + permettre d'envoyer des commandes à un appareil spécifique à partir du cloud.
- + fournit des fonctionnalités d'inscription permettant d'approvisionner des appareils et de contrôler ceux qui sont autorisés à se connecter à l'infrastructure.
- + permet de suivre l'état des appareils et de surveiller leurs activités.

Les solutions IoT peuvent inclure des boucles de rétroaction automatique. Par exemple, un module d'analyse dans le serveur principal peut déterminer, à partir de données de télémétrie, que la température d'un appareil spécifique est supérieure aux niveaux de fonctionnement normal et donc envoyer une commande à l'appareil pour corriger le problème.

✗ **Présentation et connectivité**

Grâce à la couche de présentation et de connectivité métier, les utilisateurs finaux peuvent interagir avec la solution IoT et les appareils. Les utilisateurs peuvent ainsi afficher et analyser les données collectées à partir de leurs appareils. Ces vues peuvent prendre la forme de tableaux de bord ou de rapports d'aide à la décision. Par exemple, un opérateur peut

vérifier l'état de camions de livraison spécifiques et consulter les alertes éventuellement déclenchées par le système. Cette couche permet également d'intégrer le serveur principal de solution IoT aux applications métier existantes pour une adaptation optimale aux flux de travail et processus métier.

3. Les protocoles dédiés pour l'IoT

La communication entre « objets connectés » ou tout simplement dans l'environnement domotique est probablement l'une des choses essentielles.

Dans les protocoles de communication que l'on retrouve le plus souvent, nous avons bien sûr http et ses fameuses API Rest ,CoAP, MQTT et AMQP ,les protocoles présentés dans cette partie reposent sur deux types distincts d'architecture : l'architecture Request/Response et l'architecture Publish/Subscribe.

La première est l'architecture traditionnelle, celle que nous utilisons lorsque nous consultons un site par exemple : le client (navigateur) émet des requêtes à destination du serveur qui héberge ce site, lequel lui répond le contenu demandé. Le protocole HTTP, largement utilisé pour l'Internet, repose sur cette architecture.

La seconde architecture, Publish/Subscribe, permet, quant à elle, à un client – un capteur de température par exemple – de publier un message sur le réseau sans se soucier des destinataires du message, le client est alors ce qu'on appelle un « publisher ». Le client « publisher » publie son message sur un topic auquel peuvent être abonnés les clients qui souhaitent recevoir le message. Les clients abonnés au topic sont les « subscribers » de ce topic. Des mécanismes permettent de réserver la souscription aux topics aux seuls clients autorisés. Le lien entre les « publishers » (ceux qui publient) et les « subscribers » (ceux qui souscrivent) est assuré par un « broker » dont la principale fonction est de distribuer les messages publiés sur un topic aux abonnés de ce topic (les subscribers).

3.1. Architecture Request/Response

CoAP (Constrained Application Protocol)

Le CoAP est un protocole conçu par l'IETF (Internet Engineering Task Force) basé sur une architecture Request/Response.

Ce protocole reprend en partie les concepts et la terminologie du protocole HTTP (GET, PUT, POST et DELETE) en allégeant considérablement les échanges, notamment en s'appuyant sur le protocole UDP plutôt que sur le protocole TCP.

Cette proximité avec le protocole HTTP rend le développement de passerelles entre ces deux protocoles relativement aisé.

Les paquets CoAP sont beaucoup plus petits que les paquets HTTP, l'en-tête d'un message traditionnellement utilisé pour les messages HTTP est remplacé par un en-tête fixe de 4 octets plus adapté aux faibles ressources des équipements de l'IoT.

Dans l'en-tête de chaque paquet, deux bits indiquent le type de message et la qualité de service souhaitée pour la transmission du message.

- + Confirmable: Message envoyé avec une demande d'acquittement.
- + Non-Confirmable: Message envoyé sans demande d'acquittement.
- + Acknowledgment: Confirmation de la réception d'un message de type « confirmable ».
- + Reset: Confirmation de la réception d'un message qui n'est pas exploitable.

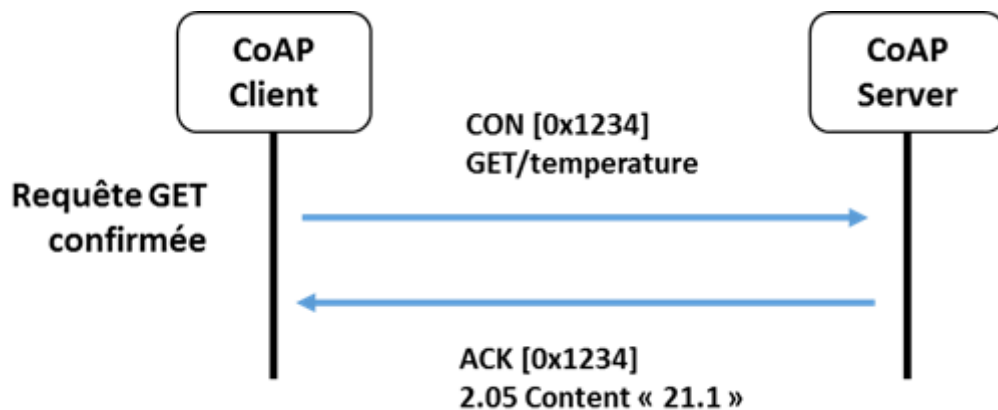


Figure 10: Principe de fonctionnement du protocole CoAP

3.2. Architecture Publish/Subscribe

✗ MQTT (Message Queuing Telemetry Transport)

Le MQTT, développé à l'origine par Arcom et IBM (pour une « smarter planet ») dans la fin des années 90, est un protocole de messagerie simple et léger, basé sur une architecture « publish/subscribe » spécifiquement conçu pour des applications M2M (Machine To Machine).

L'intérêt majeur d'utiliser ce protocole dans les applications est la notification en temps réel de la publication des nouvelles données sans devoir constamment interroger un serveur : cela apporte de la réactivité aux applications sans surcharger le serveur de requêtes inutiles.

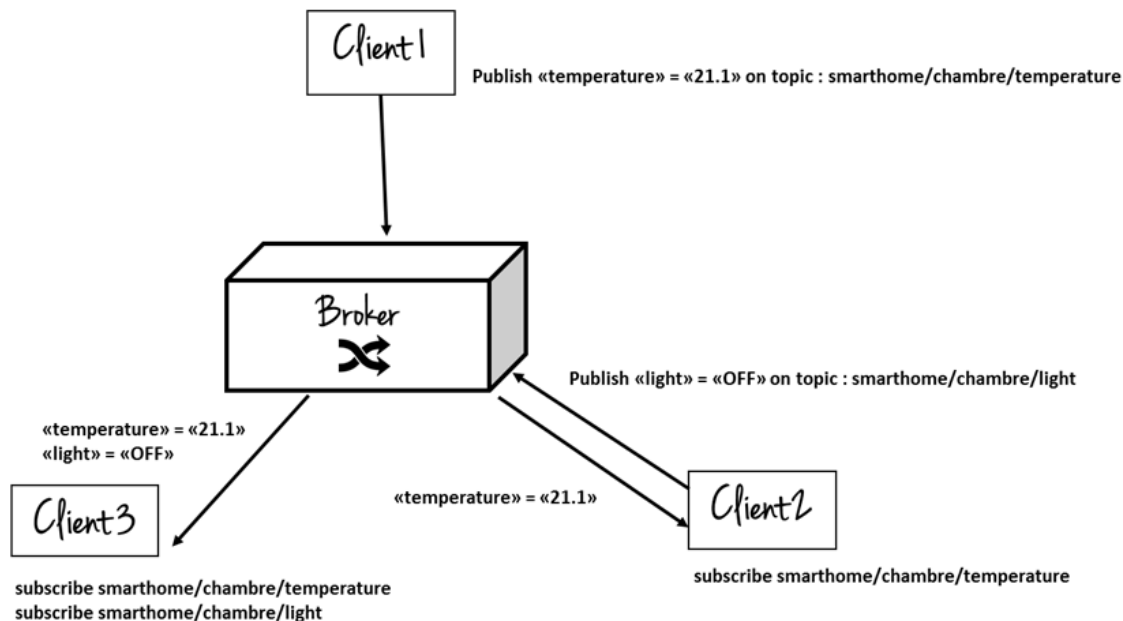


Figure 11: Principe de fonctionnement du protocole MQTT

Trois niveaux de qualité de service « QoS » sont prévus pour la transmission des messages :

- + Fire and forget: le message est envoyé une fois et aucun acquittement n'est exigé.
- + Delivered at least once : le message est envoyé au moins une fois et un acquittement est exigé.
- + Delivered exactly once : un mécanisme en quatre temps assure la délivrance du message une seule et unique fois.

Encore plus légère, une version optimisée du protocole MQTT, adaptée aux objets à très faibles ressources communicants sans fil, est spécifiée sous le nom MQTT-SN (MQTT for Sensor Network).

✕ AMQP (Advanced Message Queuing Protocol)

Enfin le protocole AMQP, originaire du monde de la finance, est dans son principe de fonctionnement assez proche du protocole MQTT.

AMQP apporte un mécanisme supplémentaire de « files d'attente ». En effet, les clients qui émettent des messages sont appelés des « producers ». Ces messages sont dirigés vers une ou plusieurs files d'attente. Une fois le message « déposé » dans la file d'attente, il est prêt à être consommé par les clients que l'on appelle les « consumers ». Sur le même principe que pour le protocole MQTT, les « consumers » reçoivent les messages des files d'attente auxquelles ils sont abonnés.

Il permet, comme les autres, d'indiquer la qualité de service souhaitée dans la transmission du message :

- + Au plus une fois : le message est transmis une fois, peu importe qu'il soit délivré ou non.
- + Au moins une fois : le message est transmis et délivré au moins une fois : il peut l'être plusieurs fois.
- + Une seule et unique fois : le message est transmis et délivré une seule et unique fois;

Les fonctionnalités et notamment la fiabilité de ce protocole sont au détriment de sa performance dans des environnements très limités en ressources.

AMQP est plus adapté aux situations exigeant fiabilité et sécurité.

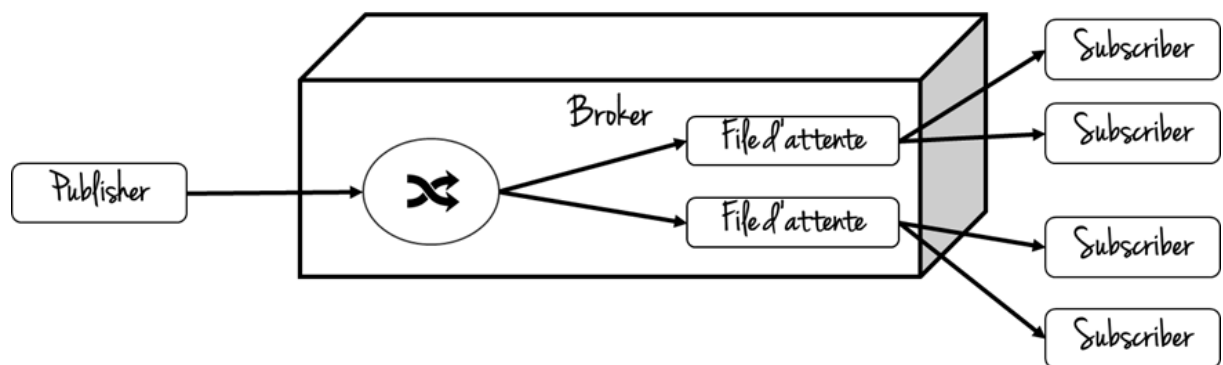


Figure 12: Principe de fonctionnement du protocole AMQP

II. Étude et benchmarking des solutions IoT

Après la brève analyse ci-dessus sur toutes les principales parties d'une solution IoT, nous pouvons dire que la partie principale dans une architecture IoT est le Cloud Gateway qui est responsable de gérer un grand nombre de dispositifs à l'échelle et un grand nombre de messages entrants par seconde dans le système.

Pour cette raison, les grandes entreprises ont commencé à développer leurs solutions pour fournir une telle passerelle et de simplifier la partie d'ingestion et de communication dans une solution IoT. Le 29 Septembre, Microsoft a annoncé son Azure IoT Hub et le 9 Octobre, Amazon a répondu avec sa plate-forme AWS IoT, sur la base de notre expérience d'IoT Hub et l'étude d'AWS IoT, nous voudrions comparer ces deux implémentations Cloud Gateway en décrivant leurs caractéristiques.

Ces offres réutilisent les briques déjà en place comme le traitement temps réel, le stockage ou l'analyse de données et rajoutent une couche pour la connectivité et quelques algorithmes pour l'intelligence. Il s'agit avant tout de rebander des services existants en ajoutant IoT.

Ce préambule fait, ces plateformes répondent néanmoins aux spécificités du monde des objets connectés puisqu'elles ont vocation à connecter un grand nombre d'appareils hétérogènes.

Il s'agit tout d'abord de proposer des protocoles de communication optimisés pour la remontée de données. Au-delà des passerelles HTTP et HTTPS, les fournisseurs s'appuient sur les protocoles MQTT (Message Queuing Telemetry Transport) et AMQP (Advanced Message Queuing Protocol) pour "travailler en environnement frugal", et ne pas trop puiser dans les ressources système et la bande passante.

Il convient ensuite de faire dialoguer des services web et des objets connectés qui ne parlent pas forcément la même langue. Ce qui passe par une automatisation des règles sur le principe IFTTT (If this then that, si ceci alors cela). Microsoft est celui qui va le plus loin dans ce sens en proposant via Azure IoT Suite des scénarios prédéfinis pour la maintenance prédictive et la gestion des actifs.

Le but de cette partie est de réaliser un benchmarking sur les plates-formes logicielles IoT, nous avons d'abord créé une liste des principales caractéristiques qui sont importantes

pour n'importe quelle plate-forme logicielle d'IoT. Ensuite, nous avons comparé l'étendue à laquelle ses caractéristiques principales ont été implémentées dans les plates-formes logicielles IoT. Enfin, nous avons comparé les solutions basées sur nos observations et tests réalisés.

Sur la base de plusieurs études, nous avons sélectionné les questions à poser comme étant cruciales pour une plate-forme logicielle IoT :

✕ **Hardware**

- + Les environnements compatibles (arduino, raspberry, mbed, odroid, etc.)
- + Ont-ils des kits de dev (SDK) ? Si oui les langages supportés.

✕ **Communication vers la plateforme**

- + Les protocoles compatibles (REST, MQTT, COAP, WebSocket, autres ?).
- + La dépendance avec le réseau. Est-ce que c'est compatible avec des offres Low Power type LoRa ou SigFox ?

✕ **La gestion de la collecte**

- + Comment gère-t-on la remontée d'information ?
- + Quels sont les éléments de configurations/personnalisations à prévoir côté solution ou côté objet ?
- + Comment configure-t-on le model de données ? Ex: je remonte des données d'un capteur de glycémie (diabète) quels sont les mécanismes pour définir que je remonte un %âge et non des chaines de caractères ?
- + Est-ce que l'on peut chiffrer les données ? Le niveau de sécurité que l'on a à disposition ?

✕ **Le device management**

- + Quelles sont les fonctions prévues ? Mise à jour de firmware ?
- + Les solutions compatibles ? Est-ce que ça gère des services sur des environnements propriétaires ?

✕ **Le traitement des données**

- + Quels outils sont mis à disposition pour traiter les données ?
- + Réalisation de Dashboard ?

+ Comment on gère l'intégration du système dans un écosystème client ? En gros l'outillage permettant d'exploiter les informations collectées.

1. AWS IoT

AWS IoT est une plate-forme qui permet de connecter des appareils aux services AWS et à d'autres appareils. Elle sécurise les données et interactions, traite les données transmises par les différents appareils et déclenche des actions en conséquence. Elle permet aux applications d'interagir avec des appareils, et ce, même lorsque ces derniers ne sont pas connectés.

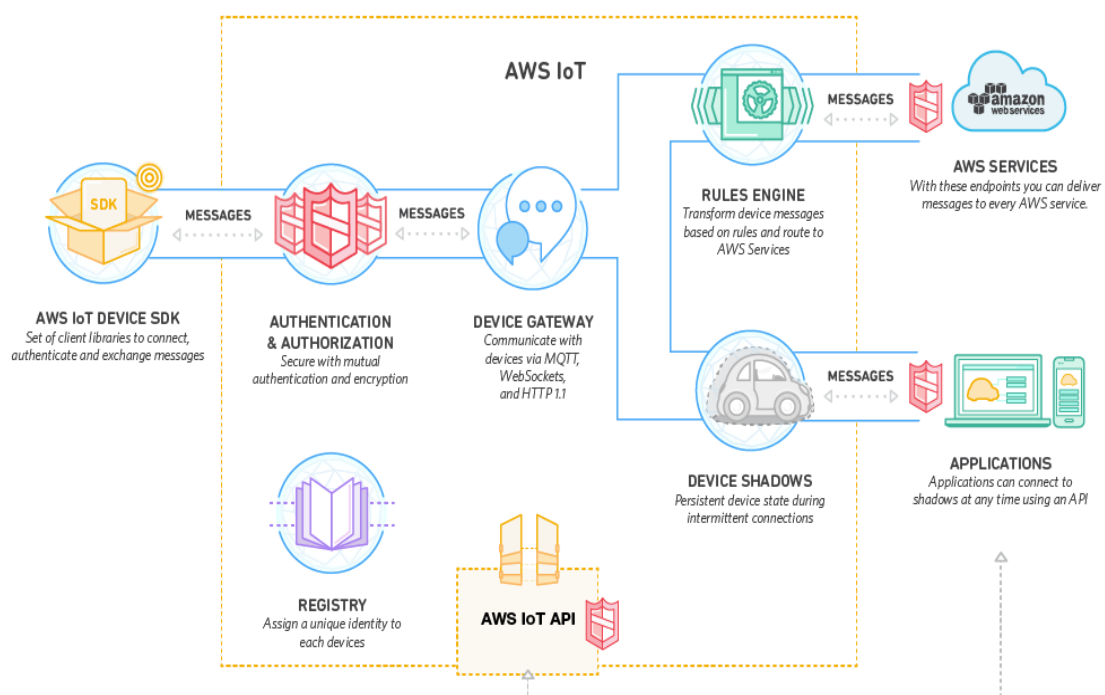


Figure 13: Architecture AWS IoT

✂ Kit SDK pour les appareils AWS IoT

AWS IoT fournit un kit SDK pour aider à connecter facilement et rapidement aux équipements matériels ou application mobile. Le kit SDK pour les appareils AWS IoT permet la connexion et l'authentification des appareils auprès d'AWS IoT, ainsi que l'échange de messages avec AWS IoT, via les protocoles MQTT, WebSockets ou HTTP. Ce kit SDK prend en charge les langages C, JavaScript et Arduino et inclut des bibliothèques de clients, un manuel destiné aux développeurs ainsi qu'un guide de portage pour les fabricants. Il est

possible également d'utiliser une solution alternative à code source libre ou concevoir notre propre kit de développement logiciel.

✗ **Registre**

Un registre identifie chaque appareil et permet d'assurer le suivi de ses métadonnées (attributs et caractéristiques notamment). Le registre affecte une identité unique à chaque appareil, selon un format cohérent, quel que soit le type d'appareil et le mode de connexion utilisés. Il intègre également des métadonnées qui décrivent les caractéristiques de l'appareil. Ainsi, dans le cas d'un capteur, elles précisent s'il s'agit d'un capteur de température et si ses mesures sont en degrés Fahrenheit ou Celsius.

✗ **Moteur de règles**

Le moteur de règles permet de concevoir des applications AWS IoT qui collecte, traitent et analysent les données transmises par les appareils connectés afin de déclencher des actions, le tout sans que nous gérons l'infrastructure. Le moteur de règles évalue les messages entrants publiés dans AWS IoT et les convertit afin de les transmettre à un autre appareil ou à un service de cloud, selon les règles métier que nous définies. Une même règle peut s'appliquer aux données issues d'un ou de plusieurs appareils et elle peut déclencher une ou plusieurs actions en parallèle.

Le moteur de règles peut aussi acheminer les messages vers les points de terminaison AWS, notamment AWS Lambda, Amazon Kinesis, Amazon S3, Amazon Machine Learning, Amazon Dynamo Db, Amazon Cloud Watch et Amazon Elastic search Service avec intégration Kibana. Pour atteindre nos éventuels points de terminaison externes, nous devons passer par AWS Lambda, Amazon Kinesis ou Amazon simple notification Service (SNS).

✗ **Sécurité**

AWS IoT assure l'authentification mutuelle et le chiffrement sur tous les points de connexion, si bien qu'aucun échange de données ne se produit entre les appareils et AWS IoT sans que les identités n'aient été vérifiées. AWS IoT prend en charge la méthode AWS d'authentification (appelée « SigV4 ») ainsi que l'authentification basée sur le certificat X.509. Les connexions par HTTP peuvent utiliser l'une de ces méthodes, tandis que celles par MQTT utilisent l'authentification par certificat. Quant aux connexions faites avec WebSockets, elles peuvent utiliser le protocole SigV4. Avec AWS IoT, il est possible d'utiliser les certificats

générés par AWS IoT, ainsi que ceux signés par notre autorité de certification (CA) favorite. nous pouvons, pour chaque certificat, associer le rôle et/ou les politiques de votre choix afin d'octroyer à des appareils et applications des autorisations d'accès spécifiques. Par la suite, il est possible de les révoquer à tout moment, sans aucune action requise au niveau des appareils.

2. Azure IoT Hub

Azure IoT Hub est un service entièrement géré qui permet des communications bidirectionnelles fiables et sécurisées entre des millions d'appareils IoT et un serveur principal de solution. Azure IoT Hub :

- + Fournit une messagerie appareil-à-cloud et cloud-à-appareil fiable à grande échelle.
- + Assure la sécurité des communications grâce aux informations d'identification de sécurité par appareil et au contrôle d'accès.
- + Fournit une surveillance complète de la connectivité des appareils et des événements de gestion de l'identité des appareils.
- + Inclut des bibliothèques d'appareils pour les langages et les plateformes les plus courants.

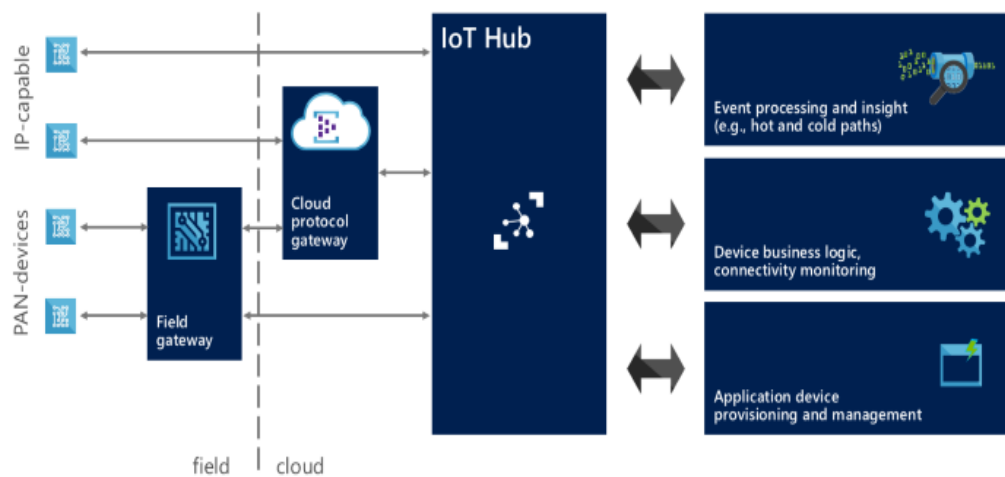


Figure 14: Architecture Iot Hub

✗ **Le device management**

IoT Hub et les bibliothèques d'appareils aident à relever les défis en matière de fiabilité et de sécurisation de la connexion d'appareil au serveur principal de la solution. Les appareils IoT :

- + sont souvent des systèmes intégrés, qui ne font appel à aucun opérateur humain.
- + peuvent être situés sur des sites distants avec un accès physique très coûteux.
- + sont accessibles uniquement via le serveur principal de la solution.
- + peuvent avoir des performances et/ou des ressources de traitement limitées.
- + peuvent avoir une connectivité réseau intermittente, lente ou coûteuse.
- + peuvent nécessiter l'utilisation des protocoles d'application personnalisés, propriétaires ou spécifiques à un secteur.
- + peuvent être créés à l'aide d'un large éventail de plateformes matérielles et logicielles populaires.

Outre les exigences ci-dessus, n'importe quelle solution IoT doit également offrir des possibilités d'évolution, la sécurité et la fiabilité. Il en résulte un ensemble d'exigences de connectivité complexe, long à mettre en place lorsque nous pouvons utiliser des technologies traditionnelles telles que des conteneurs web et des courtiers de messagerie.

✗ **Les Kits de développement logiciel (SDK)**

Les Kits de développement logiciel (SDK) d'appareil Azure IoT sont disponibles et pris en charge pour différents langages et plateformes : C pour les distributions Linux, Windows et les systèmes d'exploitation en temps réel. Les Kits de développement logiciel (SDK) d'appareil Azure IoT prennent également en charge les langages gérés tels que C#, Java et JavaScript.

✗ **Sécurité**

Azure IoT Hub répond aux défis liés à la sécurité de l'appareil par le mécanisme de l'authentification par appareil et connectivité sécurisé, nous pouvons configurer chaque appareil avec sa propre clé de sécurité pour lui permettre de se connecter à IoT Hub. Le Registre d'identité IoT Hub stocke les clés et les identités des appareils dans une solution. Un serveur principal de la solution peut intégrer des appareils individuels à une liste verte ou rouge, permettant ainsi un contrôle de l'accès à l'appareil.

Un autre mécanisme crucial est la surveillance des opérations de connectivité des appareils. Nous pouvons recevoir des journaux d'opérations détaillés sur les opérations de gestion de l'identité des appareils et sur les événements de connectivité des appareils. Ainsi, notre solution IoT peut facilement identifier les problèmes de connectivité, tels que les appareils qui tentent de se connecter avec des informations d'identification incorrectes, envoient des messages trop fréquemment ou rejettent tous les messages cloud-à-appareil.

✕ **Messagerie**

IoT Hub offre des primitives de message pour communiquer, Cloud-à-appareil : à partir d'un serveur principal d'applications (service ou cloud), et Appareil-à-cloud : à partir d'un appareil vers un serveur principal d'applications. Les principales propriétés de la fonctionnalité de messagerie IoT Hub sont la fiabilité et la durabilité des messages. Elle active la résilience de la connectivité intermittente côté appareils et des pics de chargement dans le traitement d'événements côté cloud. IoT Hub implémente au moins une fois des garanties de remise pour l'envoi de messages appareil-à-cloud et cloud-à-appareil.

IoT Hub prend en charge plusieurs protocoles d'appareil. Si notre solution ne peut pas utiliser les bibliothèques d'appareil, IoT Hub propose un protocole public qui permet aux appareils d'utiliser nativement les protocoles MQTT v3.1.1, HTTP 1.1 ou AMQP 1.0. nous pouvons également étendre IoT Hub afin de permettre la prise en charge d'un protocole personnalisé en adaptant le composant open source Azure IoT Protocol Gateway. Nous pouvons exécuter la passerelle de protocole Azure IoT (Azure IoT Protocol Gateway) localement ou dans le cloud.

3. Résumé de l'étude comparative pour les deux solutions

	MICROSOFT AZURE IOT HUB	AMAZON AWS IOT
protocoles	HTTP, AMQP, MQTT	HTTP, MQTT
Modèles de communication	Telemetry, Command	Telemetry, Command (state change)
Environnements compatibles	Intel, Raspberry Pi 2, Freescale, Texas Instruments, MinnowBoard, BeagleBoard, Seeed, resin.io	Broadcom, Marvell, Renesas, Texas Instruments, Microchip, Intel, Mediatek, Qualcomm, Seeed, BeagleBoard
SDK	.Net and UWP, Java, C, NodeJS	C, NodeJS
Sécurité	TLS (authentification serveur seulement)	TLS (authentification mutuelle)
Authentification	Par appareil avec SAS token	X.509 authentification par certification client, IAM service, Cognito service

Microsoft et Amazon ont développé leurs plates-formes avec des choix différents à partir des protocoles sous-jacents utilisés pour la communication : AMQP vs MQTT. Microsoft a déjà utilisé AMQP pour tous les services relevant du Service Bus donc il semble être un choix logique pour IoT Hub. Concernant Amazon, nous pouvons seulement supposer que le choix est lié à l'acquisition des télémetries par la société qui a développé sa plate-forme IoT en utilisant MQTT. Les deux plates-formes prennent en charge HTTP qui est un protocole bien connu, utilisé par les développeurs pour beaucoup d'applications de cloud.

Les mécanismes pour fournir le modèle « télémétrie » pourrait considérer comme tout à fait semblable, mais le mécanisme pour le modèle « command » avec un chemin de « request/reply » est tout à fait différent, C'est une fonctionnalité intégrée pour AMQP mais a besoin d'une couche supplémentaire sur le dessus de MQTT.

L'interaction avec des dispositifs est différent, il est basé sur des échanges de messages dans l'IoT Hub et plus strictement liée à la notion d'état de l'appareil dans AWS IoT grâce aux

« thing shadows ». Sur le premier côté, nous avons un message qui peut transporter des données d'application et les commandes pour les actions, de l'autre côté, nous avons la représentation de l'état dans un document JSON (bien sûr porté à l'intérieur d'un message MQTT). C'est comme d'avoir une couche d'abstraction plus données dans AWS IoT.

Les implémentations de sécurité sont assez similaires, compte tenu de l'utilisation du protocole TLS, l'identité est gérée d'une manière différente, mais avec presque les mêmes résultats.

Concernant le matériel et les kits de développement logiciel, AWS IoT supporte c et NodeJS, tandis que Microsoft ajoute .Net et UWP, Java, NodeJS.

Conclusion

Dans cette partie nous avons présenté très succinctement le concept d'internet des objets en général et ses protocoles applicatifs, ainsi qu'une étude comparative des solutions IoT dans le marché.



Chapitre 3

Analyse et conception

L'étude fonctionnelle du projet porte essentiellement sur la capture des besoins fonctionnels, d'en extraire les cas d'utilisations et de les décrire de façon détaillée. La technique des cas d'utilisation est la pierre angulaire de cette étape. Elle va nous permettre de préciser l'étude du contexte fonctionnel du projet, et d'élaborer des modèles détaillés de l'architecture du futur système.

I. Analyse des besoins

Il est à rappeler que l'objectif du projet consiste à développer un système IoT qui a pour objectif d'aider les personnes vivant avec le diabète de garder une trace et suivi de leurs médicaments et état sanitaire. Ce chapitre traite les besoins des utilisateurs en termes de fonctionnalités ainsi que de disponibilité du système. Ce qui nécessite une définition de l'architecture fonctionnelle de notre projet IoT.

Les besoins des utilisateurs peuvent être répartis selon deux catégories : des besoins en indicateurs de mesure et ceux de suivi et consigne relative au diabète.

Cette première partie sera consacrée donc à l'analyse fonctionnelle du projet et la capture des besoins du système. Pour ce faire, nous nous sommes appuyés sur l'étude des acteurs principaux et secondaires. Quant aux spécifications des besoins métiers à satisfaire par le futur système à développer.

1. Expression du besoin

Dans cette phase nous avons appliqué la méthode adoptée Scrum pour assurer un développement agile, afin d'analyser les différentes fonctionnalités du système issues des user stories, et illustrer les points estimés par l'équipe et la vélocité calculée pour chaque itération.

Scrum est plus orientée sur la gestion de projet, et fournit un ensemble de pratiques de base qui laissent volontairement le libre choix des techniques de développement.

- + Product owner : Dans notre cas, c'est Florian SPLENDIDO Practice Leader IoT chez SQLi Nante qui définit le backlog produit à mettre en place.
- + Scrum master : était nos encadrants, ils veillent sur le déroulement des sprints et la résolution des problèmes externes.
- + Sprint : on a choisi des sprints de 3 semaines, à la fin de chaque sprint, on a pu mettre en œuvre des incréments logiciels fonctionnels tenant en compte le backlog produit, la validation du savoir était effectuée en aval de chaque sprint.
- + Chaque matinée, on faisait une courte réunion (Daily scrum) où chacun présente ses aboutissements, problèmes rencontrés et objectifs.

1.1. BackLog

Le Backlog signifie la liste des fonctionnalités priorisées, et qui sont définies pour le développement d'une solution. Chaque fonction contient une courte description pour enlever les ambiguïtés que peut rencontrer le développeur.

Les fonctionnalités du BackLog ont été implémentées selon une planification courte et itérative appelée sprints.

En aval de chaque Sprint, un incrément système fonctionnel doit être validé.

+ Sprint 1 :

User Story	Détail	Points estimés
Envoie des messages télémétrie appareil vers cloud.	<ul style="list-style-type: none">• Crée une identité d'appareil dans le registre d'identité de cloud, un appareil ne peut pas se connecter, à moins de posséder une entrée dans le registre d'identité des appareils.• Créer une application qui simule un appareil envoyant des messages Appareil vers cloud.	3
Envoie des messages interactifs appareil vers cloud.	Les messages envoyés de l'appareil vers le cloud sont considérés comme interactifs lorsqu'ils agissent comme déclencheurs immédiats d'un ensemble d'actions sur le système principal de l'application, par opposition aux messages de point de données qui sont chargées dans un moteur d'analyse.	3
Envoie des messages cloud vers appareil.	Créer une application qui envoie des messages cloud vers l'appareil puis recevoir un accusé de réception.	3
Réception et stockage des données télémétrie dans le cloud.	Créer une application qui sauvegarde l'ensemble des données dans le cloud.	3

Résumé

Ce Sprint contient la réalisation de la communication bidirectionnelle fiable et sécurisée entre des appareils IoT et un serveur principal de solution. En faisant l'addition des points estimés de chaque User Story, nous allons obtenir une vélocité de 12.

+ Sprint 2 :

User Story	Détail	Points estimés
Un utilisateur veut consulter le graphe d'évolution de la température.	<ul style="list-style-type: none">▪ Collecter les données depuis le cloud.▪ Traitement et analyse des données.▪ Diffusion des résultats dans des services de visualisation.	3
Un utilisateur veut consulter les statistiques sur le glucose dans le sang.	NA	3
Un utilisateur veut consulter L'évolution du glucose et insuline dans le sang.	NA	3
Un utilisateur veut visualiser le graph taux de cholestérol dans le sang.	NA	3
Un utilisateur veut visualiser le graphe de la tension artérielle.	NA	3
Visualisation des graphes générés (Client mobile).	<ul style="list-style-type: none">▪ Création d'une API Rest pour recevoir les résultats sous le format JSON depuis le cloud.▪ Création des services de visualisation des graphes mobile.	3
Un utilisateur veut s'inscrire au système (client mobile)	<ul style="list-style-type: none">▪ Création de l'interface d'inscription.▪ Création et test du service d'inscription.	1
Un utilisateur veut s'authentifier (Client mobile)	<ul style="list-style-type: none">▪ Création de l'interface	1

	d'authentification. <ul style="list-style-type: none"> ▪ Création et test du service d'authentification. 	
Paramétrage des unités de mesures (Client mobile)	Développement d'interface et service de modification des unités et mesures.	2
Exporter les données (Client mobile)	Exporter les données vers différent format.	1
Consulter l'historique (Client mobile)	Réalisation d'interface de consultation d'historique.	2
Assistant (Client mobile)	NA	1
Renseigner les nutritions	Réalisation de formulaire pour renseigner les nutritions du jour.	1
Visualisation des graphes générés (Client Web).	<ul style="list-style-type: none"> ▪ Création d'une API Rest pour recevoir les résultats sous le format JSON depuis le cloud. ▪ Création des services de visualisation des graphes web. 	3

Résumé

Ce Sprint vise le développement des fonctionnalités liées à l'analyse, traitement et visualisation des données IoT ainsi que les fonctionnalités de la base dans les deux clients (mobile et web). En additionnant les points estimés, on va obtenir

Vélocité = 31 donc la vélocité moyenne = $30+12 / 2 = 21$ Cela veut dire que la capacité de production moyenne a augmenté dans le dernier Sprint, puisqu'elle s'est stabilisée avec la vélocité du Sprint précédent.

La prochaine étape sera consacrée au référentiel d'exigence qui sert à définir de manière structurée les attendus du métier pour le projet, en termes fonctionnels, métier, technique, sécurité, charge et temps de réponse...

1.2. Référentiel des exigences:

Cette partie retranscrit tous les besoins métiers détaillés liés au système à développer. Il permet au Product owner de contrôler que son besoin est entièrement retranscrit.

✗ Définition du statut

- + Proposé : l'exigence a été proposée et doit faire l'objet d'une étude
- + Approuvé : l'exigence correspond à un besoin mais doit faire l'objet d'une modification pour être validée
- + Validé : l'exigence donne une solution acceptée à un besoin

✗ Définition de la difficulté :

- + Basse : L'exigence nécessite des compétences bien maîtrisées par l'équipe de conception. Sa mise en œuvre sera sûre.
- + Moyenne : L'exigence nécessite des compétences maîtrisées par l'équipe de conception. Sa mise en œuvre peut rencontrer des problèmes.
- + Haute : L'exigence nécessite des compétences non maîtrisées par l'équipe de conception. Sa mise en œuvre risque d'être délicate, voire impossible.

✗ Définition de la priorité :

- + Basse : L'implémentation de cette exigence n'a pas encore besoin d'être planifiée.
- + Moyenne : L'exigence sera implémentée dans une prochaine itération.
- + Haute : L'exigence est en cours d'implémentation.
- + Réalisée : L'exigence est implémentée

Le tableau suivant décrit pour chaque référence son statut, difficulté et priorité

Id	Libellé	Catégorie	statut	priorité	Implémentation
GLY_SR_EX1	Crée une identité d'appareil	Technique	Validé	Haute	Basse
GLY_SR_EX2	Envoie des évènements vers cloud.	Technique	Validé	Haute	Haute
GLY_SR_EX3	Envoie des évènements interactifs vers cloud.	Technique	Validé	Haute	Haute
GLY_SR_EX4	Envoie des évènements vers l'appareil.	Technique	Validé	Haute	Haute
GLY_SR_EX5	Stockage fiable des données	Performance	Validé	Haute	Haute
GLY_SR_EX6	Sécurité de l'appareil	Technique	Validé	Haute	Haute
GLY_BI_EX1	Graph température	Fonctionnel	Validé	Moyenne	Moyenne
GLY_BI_EX2	Graph glucose	Fonctionnel	Validé	Moyenne	Moyenne
GLY_BI_EX3	Graph glucose et insuline	Fonctionnel	Validé	Moyenne	Moyenne
GLY_BI_EX4	Graph cholestérol	Fonctionnel	Approuvé	Moyenne	Moyenne
GLY_BI_EX5	Graph tension artérielle	Fonctionnel	Validé	Moyenne	Moyenne
GLY_MOB_EX1	Visualisation des charts coté mobilité	Fonctionnel	Proposé	Basse	Basse
GLY_MOB_EX2	Exporter les résultats vers différent format.	Fonctionnel	Proposé	Basse	Basse
GLY_MOB_EX3	S'inscrire au system IoT	Fonctionnel	Validé	Moyenne	Basse
GLY_MOB_EX4	S'authentifier au system IoT	Fonctionnel	Validé	Moyenne	Basse
GLY_MOB_EX5	Recevoir des notifications	Fonctionnel	Validé	Basse	Haute

GLY_M OB_EX6	Afficher l'historique	Fonctionnel	Proposé	Basse	Basse
GLY_M OB_EX7	Alert des médicaments	Fonctionnel	Proposé	Basse	Haute
GLY_M OB_EX8	Envoyer des consignes	Fonctionnel	Proposé	Basse	Haute
GLY_TE C_EX1	Changer le fréquence d'envoi des données	Technique	Validé	Basse	Haute
GLY_M OB_EX9	Demander une assistance	Fonctionnel	Proposé	Basse	Basse
GLY_TE C_EX2	Fiabilité des résultats.	Performance	Validé	Haute	Basse
GLY_TE C_EX3	Ergonomie des interfaces.	Ergonomique	Validé	Haute	Basse

II. La phase de conception :

L'étude fonctionnelle qui a été faite auparavant nous a permis de bien cerner le projet afin de déterminer les besoins fonctionnels, qui vont être le sujet de conception et de formalisation sous une modélisation orientée objet avec le langage de modélisation unifié UML.

UML : est un langage de modélisation graphique à base de pictogrammes conçu pour fournir une méthode normalisée pour visualiser la conception d'un système. Il est couramment utilisé en développement logiciel et en conception orientée objet.

Les diagrammes qui vont être présentés dans cette partie sont les suivants :

- + Diagramme de cas d'utilisation : Il permet de structurer les besoins des utilisateurs et les objectifs correspondants d'un système d'une part et de classer les acteurs et structurer les objectifs du système d'une autre part.
- + Diagramme de classe : Pendant que le diagramme de cas d'utilisation montre un système du point de vue des acteurs, le diagramme de classe en montre la structure interne. Il permet de fournir une représentation abstraite des objets du système qui vont interagir pour réaliser les cas d'utilisation.
- + Diagramme de séquence : il est le diagramme qui fait l'extraction des différentes interactions entre acteurs, instances et composants

1. Identification des acteurs

Un acteur représente un rôle joué par une entité externe (utilisateur humain, dispositif matériel ou autre système) qui interagit directement avec le système étudié. Un acteur peut consulter et/ou modifier directement l'état du système, en émettant et/ou en recevant des messages susceptibles d'être porteurs de données.

Les acteurs qui peuvent utiliser notre système sont les maladies du diabète, les médecins et l'administrateur qui est responsable de gestion et configuration du cloud. L'approche agile adoptée propose de réduire considérablement voire complètement l'effet tunnel en donnant davantage de visibilité, nous allons traiter dans ce rapport les cas d'utilisation des diabétiques, les autres besoins sont à définir dans les prochaines itérations.

2. Diagrammes des cas d'utilisation

Les diagrammes suivants représentent les fonctionnalités majeures de notre solution et donnent une vue sur le rôle des différents acteurs.

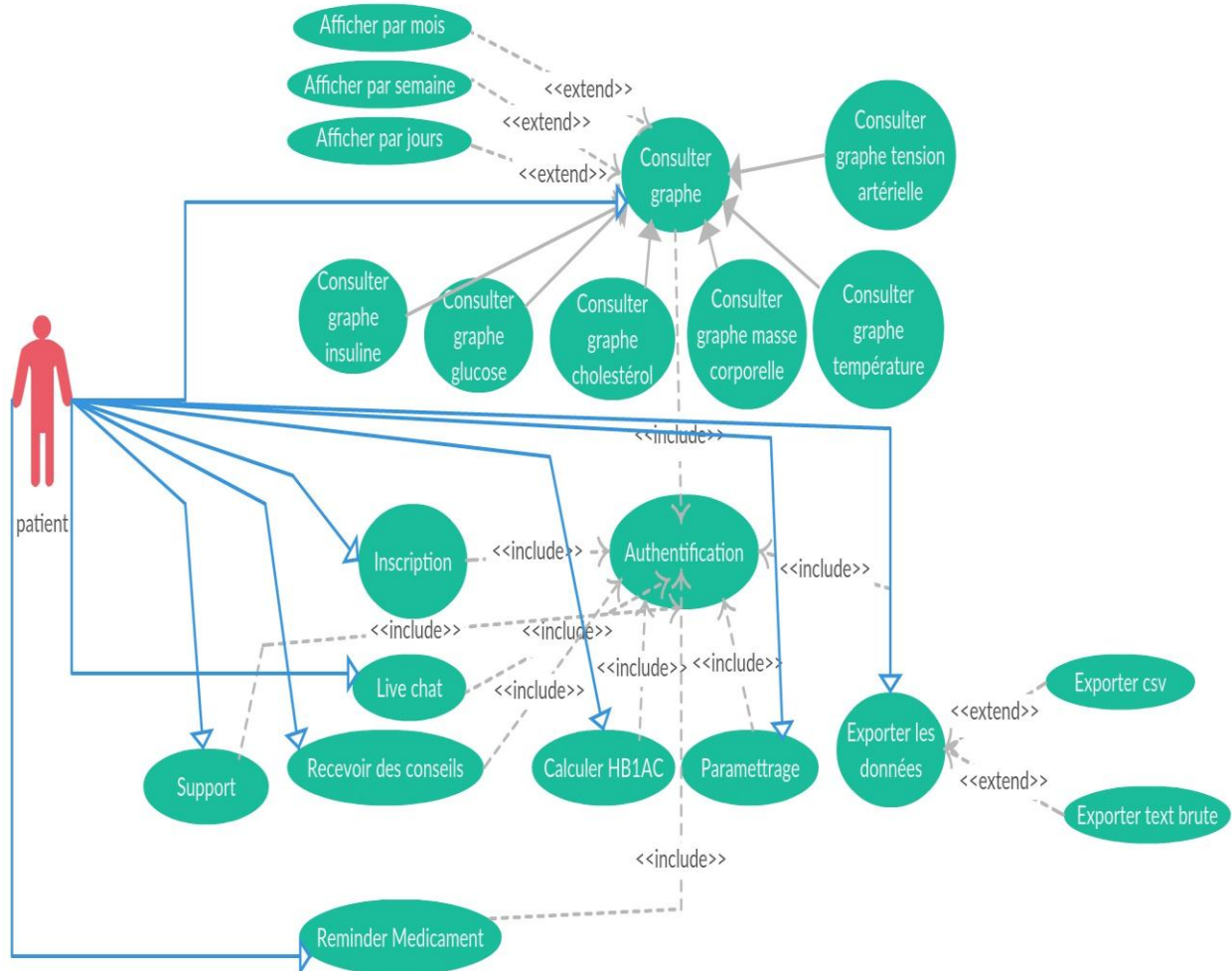


Figure 15: Diagramme de cas d'utilisation pour le patient

Ce diagramme rassemble les fonctionnalités mobiles relatives au besoin final du patient.

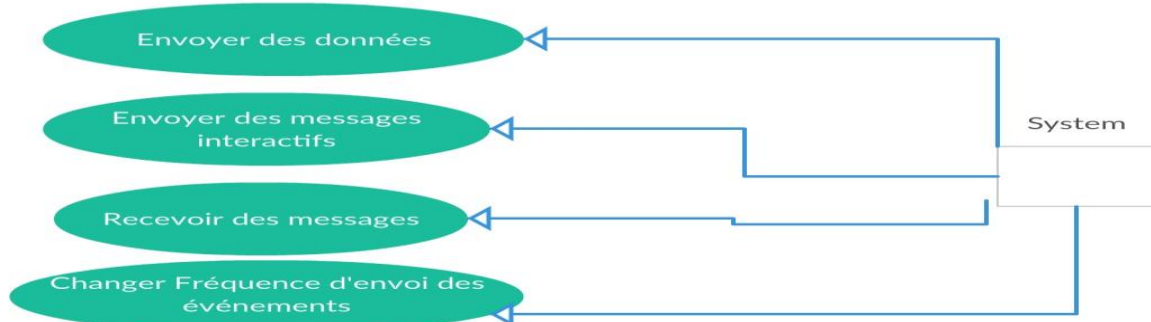


Figure 16: Diagramme de cas d'utilisation pour le système

2.1. Module AUTH : Authentification

a) Définition

Ce module permet à l'utilisateur de l'application de s'inscrire, s'authentifier, ou modifier son profil.

Use case « S'authentifier »	
Objectif	L'utilisateur se connecte à l'application et accède aux fonctionnalités de son suivi.
Acteur Principal	Patient
Acteur Secondaire	NA
Pré conditions	L'utilisateur est déjà enregistré dans le système.
Scénario nominal	<ol style="list-style-type: none">1. L'utilisateur se connecte sur la Page d'accueil de l'application.2. L'utilisateur saisit son login et son mot de passe dans la zone de connexion.3. L'utilisateur clique sur le bouton « connexion ».4. Le système vérifie le login et le mot de passe.5. L'utilisateur peut naviguer dans les fonctionnalités offertes par l'application.
Scénarios alternatifs	<p>L'utilisateur entre un mot de passe ou un login erroné.</p> <ol style="list-style-type: none">1. Le système affiche le message "Mot de passe ou Login incorrect".2. Reprendre le scénario nominal.

Use case « S'inscrire »	
Objectif	L'utilisateur se connecte à l'application et accède aux fonctionnalités de son suivi.
Acteur Principal	Patient
Acteur Secondaire	NA
Pré conditions	L'utilisateur n'est pas enregistré dans le système.
Scénario nominal	<ol style="list-style-type: none"> 1. L'utilisateur se connecte sur la Page d'accueil de l'application. 2. L'utilisateur saisit son nom, login et son mot de passe dans la zone d'inscription. 3. L'utilisateur clique sur le bouton « S'inscrire ». 4. Le système vérifie les données saisies. 5. L'utilisateur est inscrit et il peut naviguer dans les fonctionnalités offertes par l'application.
Scénarios alternatifs	<p>4a) L'utilisateur entre un login existe déjà.</p> <ol style="list-style-type: none"> 1. Le système affiche le message " Login incorrect". 1. Reprendre le scénario nominal. <p>4b) L'utilisateur entre des données erronées.</p> <ol style="list-style-type: none"> 1. Le système affiche le message " données invalides". 2. Reprendre le scénario nominal.

2.2. Module consultation des graphes

a) Définition

Ce module permet à l'utilisateur de l'application de suivre son état du diabète.

Use case « Afficher graphe »	
Objectif	L'utilisateur se connecte à l'application et accède aux fonctionnalités de son suivi.
Acteur Principal	Patient
Acteur Secondaire	NA
Pré conditions	L'utilisateur est authentifié dans le système.
Scénario nominal	<ol style="list-style-type: none"> 1. L'utilisateur se connecte dans l'application. 2. L'utilisateur clique sur l'onglet Dashboard. 3. L'utilisateur peut naviguer dans les graphes offerts par l'application.

Scénarios alternatifs	3a) Aucune donnée relative à l'utilisateur. 1. Le système affiche le message " Aucun résultat à afficher ".
------------------------------	--

2.3. Module envoi et réception des événements

b) Définition

Ce module permet l'envoi et le stockage fiable des messages envoyés de l'appareil vers le cloud dans le stockage d'objets blob Azure

Use case « Envoyer des données »	
Objectif	L'appareil envoi des données de télémétrie vers le cloud
Acteur Principal	Système (Appareil simulé)
Acteur Secondaire	NA
Pré conditions	La clé de l'appareil est enregistrée dans le cloud
Scénario nominal	<ol style="list-style-type: none"> 1. l'appareil reçoit la chaine de connexion à partir du device explorer. 2. L'appareil envoie les données vers le cloud gateway 3. le point de terminaison event hub sauvegarde les données dans des fichiers blob.
Scénarios alternatifs	1a) l'appareil n'est pas enregistré dans le cloud <ol style="list-style-type: none"> 1. Erreur d'envoi des données

3. Diagramme de classe

Le diagramme de classe considéré comme étant le plus important modèle de la modélisation orientée objet, représente d'un point de vue logique la relation qui existe entre les objets du système.

Après une étude détaillée sur les différents contraintes fonctionnelles et un ensemble des réunions avec les encadrants nous avons arrivé à élaborer le diagramme de classe suivant :

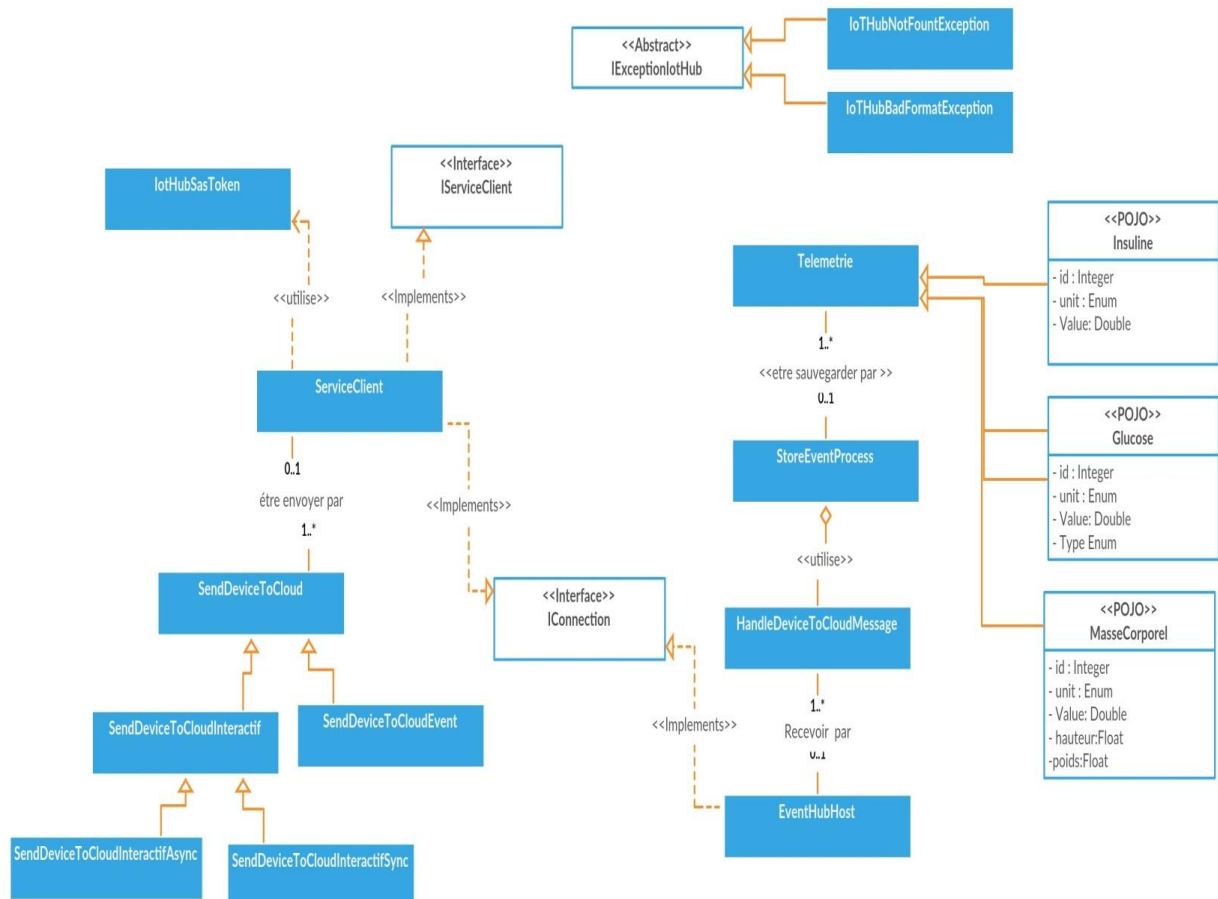


Figure 17:Diagramme de classe

Description des classes

- + Telemetrie : Cette classe encapsule les données envoyées vers le Cloud.
- + StoreEventProcess : Cette classe a pour rôle de sauvegarder les données reçues dans des fichiers non structurés mis en cache.
- + ServiceClient : Classe principale de l'appareil simulé.
- + SendDeviceToCloud : Cette classe envoie des données de télémétrie et interactives depuis l'appareil simulé vers le Cloud.
- + IoTHubSASToken : Donne l'accès de l'appareil vers un IoTHub pour une durée spécifique.
- + IConnection : Contiens les paramètres de configuration.

Il est à rappeler que nous reposons sur des cycles de développement itératifs et adaptatifs en fonction des besoins évolutifs du client pour cette raison nous avons élaboré un diagramme de classe qui prend en compte les évolutions que notre projet peut subir en cours de route.

4. Diagramme de séquence

La modélisation avec le diagramme de séquence consiste à montrer les interactions entre les instances. Donc on a choisi des scénarios prédéfinis pour les représenter sous cette forme.

A partir de l'étude préliminaire et de la description des cas d'utilisation nous avons élaboré les diagrammes de séquence suivant :

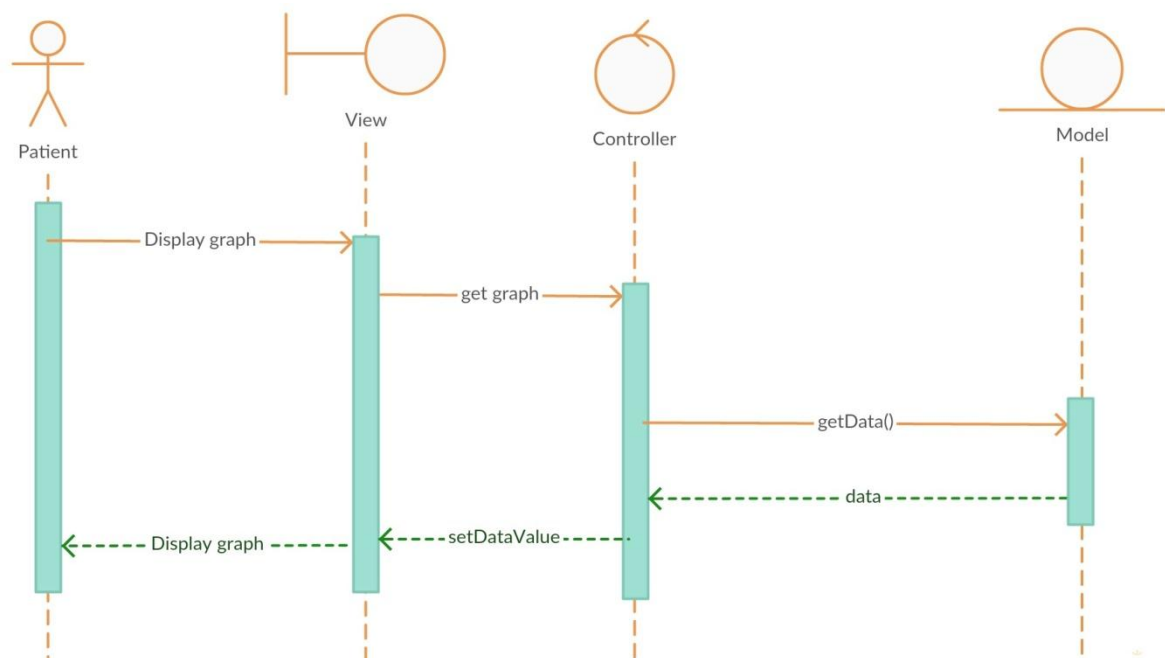


Figure 18:Diagramme de séquence relative au use cas consulter graphe

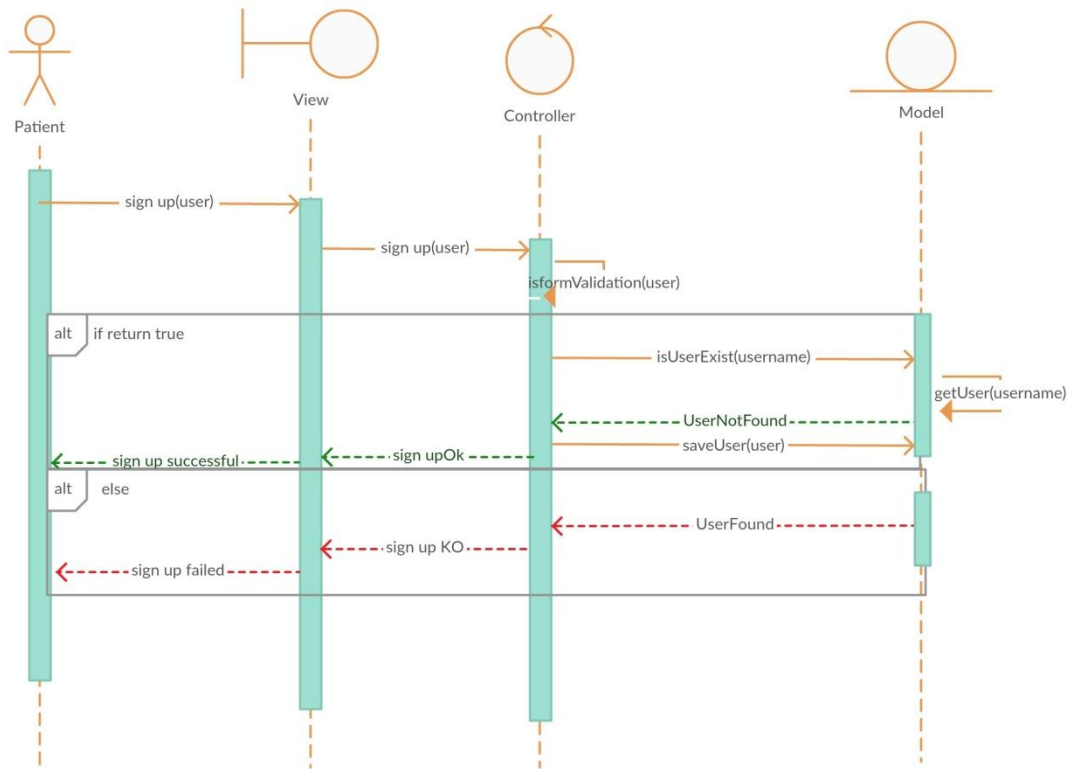


Figure 20:Diagramme de séquence relative au use cas inscription

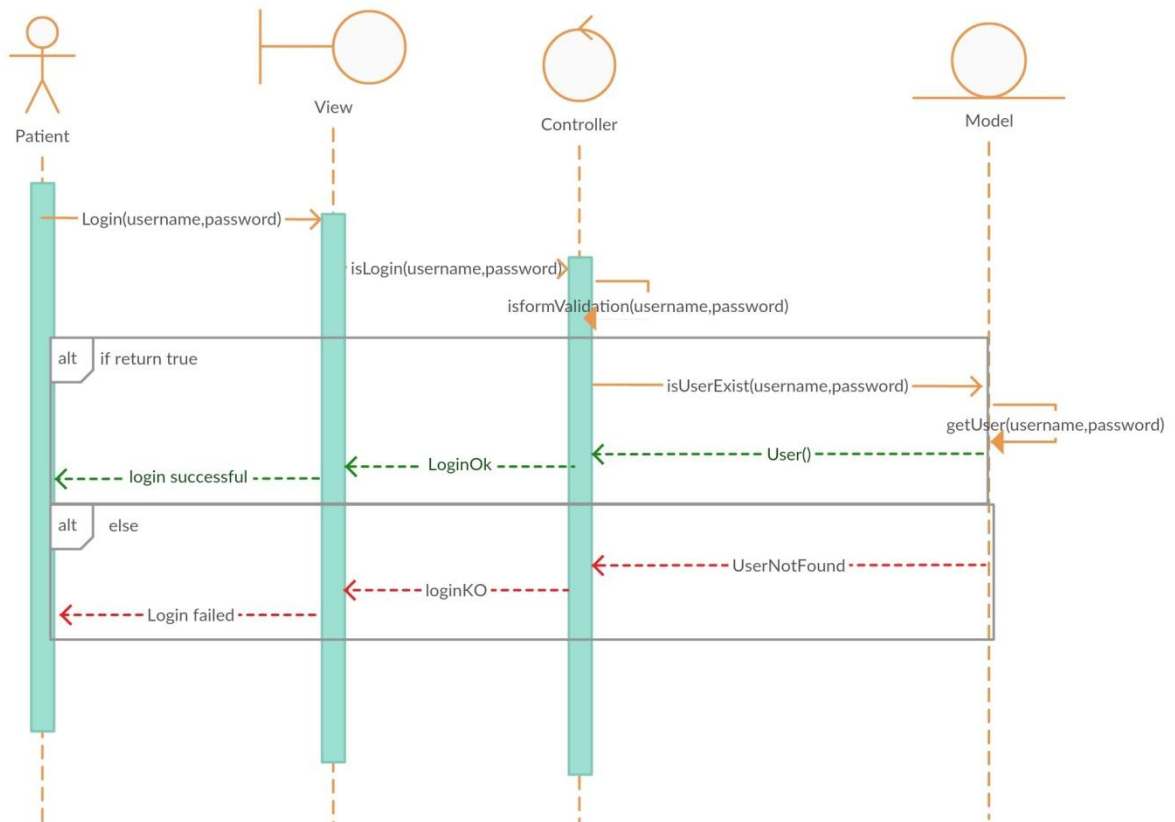


Figure 21:Diagramme de séquence relative au use cas authentification

Conclusion

Dans ce chapitre, nous avons présenté les différentes fonctionnalités de notre projet IoT. Dans un premier temps nous avons identifié les acteurs et élaborer les diagrammes des cas d'utilisation. Ensuite, la description détaillée de chaque cas d'utilisation nous a permis d'élaborer des diagrammes de séquences, ainsi le diagramme de classe.

Le pas de conception étant franchi, nous pourrons maintenant passer à l'étude technique et réalisation du système. Cette étape sera développée en détail dans le chapitre suivant.

Chapitre 4

Mise en œuvre et implémentation

La structuration d'un système permet de présenter cette dernière en des couches logiques : la logique de présentation, la logique métier et la logique de manipulation de données. Ce chapitre a pour objectif de présenter la description de l'architecture applicative cible, et l'architecture physique sur laquelle s'exécutent les différents composants techniques qui composent le système , ainsi que la façon dont ils communiquent entre eux, ensuite nous allons traiter les technologies, les outils et les langages de développement utilisés et enfin nous allons présenter les interfaces de l'application mobile et notre système IoT sur laquelle nous avons travaillé pendant la période du stage.

I. Architecture technique

Le système que nous sommes en train de réaliser, est basé sur une architecture comportant plusieurs parties à savoir :

Event Producer and Consumer : Les capteurs qui collectent les données chronologiquement (Time Series), et qui doivent être consolidés et analysés pour être utiles, nous allons utiliser des cartes Raspberry Pi et des bracelets pour mesurer la glycémie (Notez bien que jusqu'à maintenant nous utilisons des appareils de simulation seulement).

Event Queuing System : La partie responsable du traitement et réception des données de télémétrie, nous avons utilisé Event Hub qui est un service d'entrée de données hautement évolutives qui peut traiter des millions d'événements par seconde afin que nous puissions traiter et analyser des grandes quantités de données générées par nos périphériques connectés. Event Hubs fonctionne comme la « porte d'entrée » d'un pipeline d'événements, et une fois que les données sont collectées dans un concentrateur d'événements, peut être transformées et stockées à l'aide de n'importe quel fournisseur d'analyse en temps réel ou d'adaptateurs de traitement par lot ou de stockage. Les concentrateurs d'événements dissocient la production d'un flux d'événements de la consommation de ces événements, de manière à ce que les consommateurs d'événements puissent accéder aux événements selon leur propre planification.

Transformation and Analysis : S'occupe du traitement et analyse de données, le service Stream Analytics sera responsable du traitement de données en temps réel.

Storage: Représente la source et la destination des données sous format non structuré mise en cache et une base de données NoSql.

Presentation and Action : C'est la partie qui sera responsable de la visualisation et présentation de données, le service Azure en question est Power BI.

Users and Systems: L'interface client, c'est la partie responsable des interactions avec l'utilisateur.

La communication entre toutes ses parties est transparente au client de l'application, car ce dernier communique directement avec une API REST qui centralise tous les services que nous voulons exposer aux utilisateurs du système.

La figure ci-dessous illustre l'architecture globale du système et montre les flux de communication entre les différents composants.

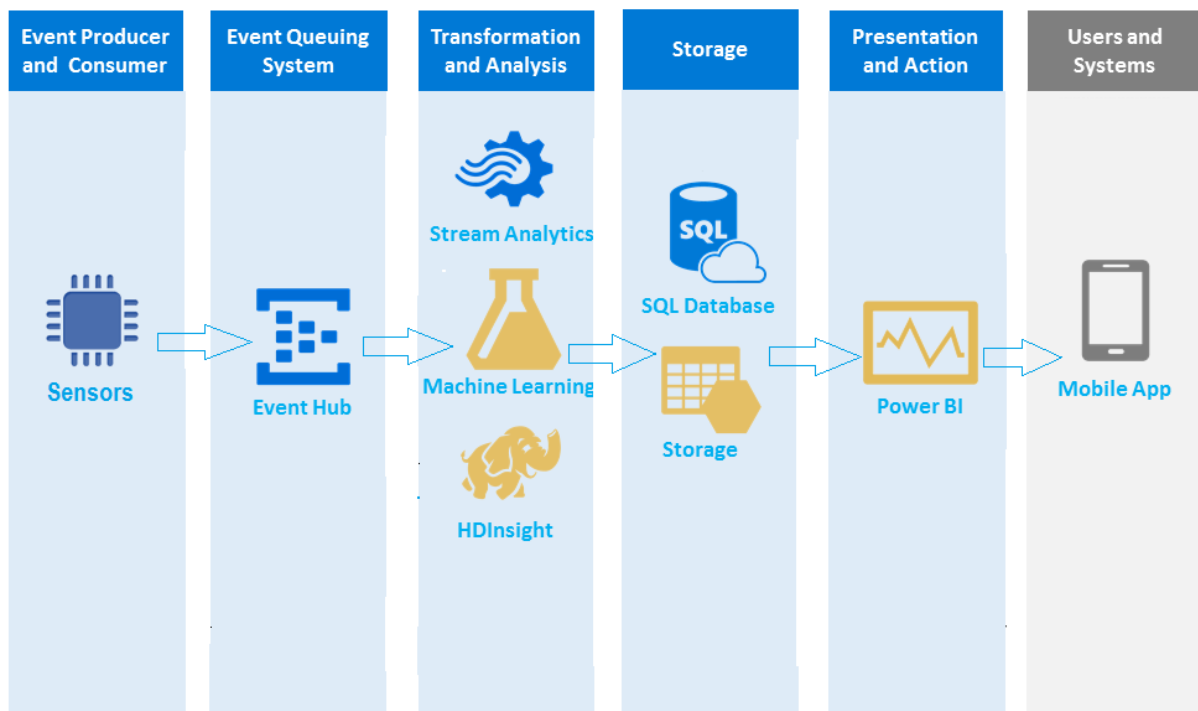


Figure 22: Architecture technique de notre projet

II. Architecture applicative

Avant de commencer le développement, nous avons déterminé l'architecture applicative du projet en vue de maîtriser les aspects du développement dans le contexte des contraintes spécifiques au projet, et afin de pouvoir concevoir un système bien structuré, évolutif et extensible. Dans cette section, nous abordons les aspects les plus importants de l'architecture applicative du projet ainsi que les approches de développement adoptées. Ceci concerne les volets suivants : Développement des services web, le développement mobile et les patrons de conception adoptés (design patterns).

✕ Services Web

Les web services représentent un mécanisme de communication entre applications distantes, à travers le réseau internet, indépendamment de tout langage de programmation et de toute plate-forme d'exécution :

- + Utilisant le protocole HTTP comme moyen de transport. Ainsi, les communications s'effectuent sur un support universel et maîtrisé.
- + Employant une syntaxe basée sur la notation XML pour décrire les appels de fonctions distantes et les données échangées.
- + Organisant les mécanismes d'appel et de réponse. Grâce aux services web, les applications peuvent être vues comme un ensemble de services métier, structurés et correctement décrits, dialoguant selon un standard international plutôt qu'un ensemble d'objets et de méthodes entremêlés.

La couche de services web représente l'intermédiaire entre le Cloud et l'application mobile. En effet, elle va exposer les informations analysées à l'application.

✕ Patrons de conception adoptés

1. MVC :

Modèle-Vue-Contrôleur (MVC) est un patron de conception qui permet de séparer l'application en trois parties distinctes, permettant d'organiser la base de code en représentations logiques des informations en fonction de leurs utilités.

La figure suivante illustre les trois composants de ce modèle et l'interaction entre eux

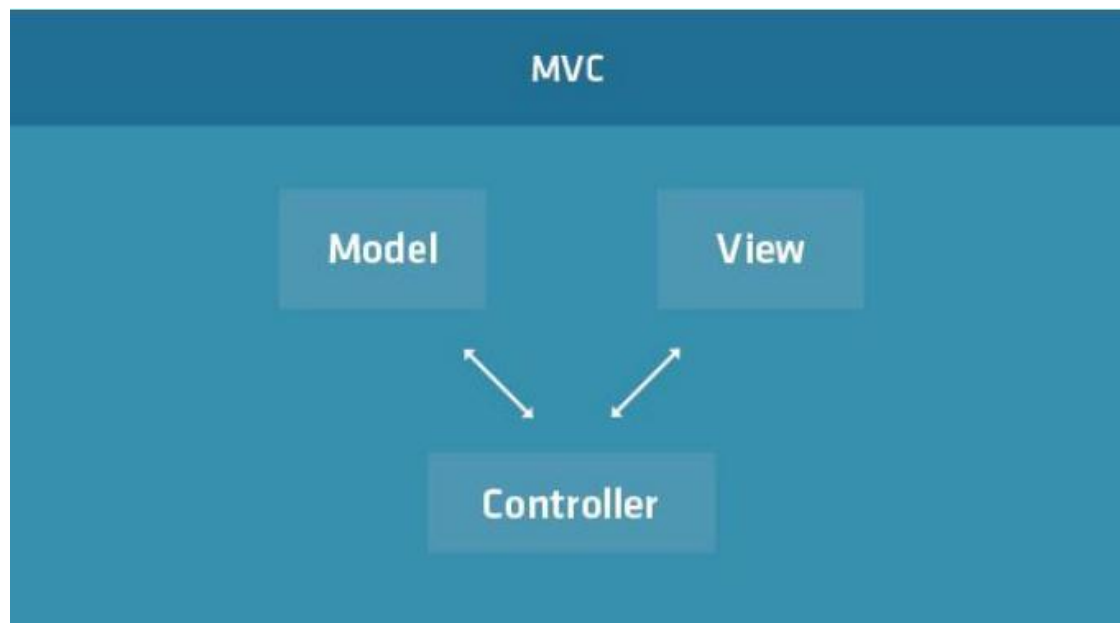


Figure 23:Le Patron de conception MVC

Les trois composants du patron MVC sont :

Le modèle : décrit un format commun pour les données utilisées dans l'application. Il peut également contenir des règles de gestion, une logique de validation, et diverses autres fonctions.

La vue : représente l'interface utilisateur.

Le contrôleur : est la pièce centrale d'une application MVC. Il écoute les événements dans l'application et les délègue des commandes entre le modèle et la vue.

L'inconvénient majeur de ce patron de conception est que le contrôleur doit assurer la tâche de synchronisation entre le modèle et la vue afin que l'application soit cohérente. Ce qui rend le développement plus lent et plus pénible.

2. MVVM :

MVVM (Model View-View Model) est une autre architecture de conception utilisée dans le génie logiciel qui se base sur l'architecture MVC. La différence entre les deux est que MVVM implémente une couche d'abstraction de la vue (ViewModel) qui permet la synchronisation automatique avec le modèle (Data-Binding). La vue reflète alors le modèle à tout moment.

La figure ci-dessous explique comment la vue et le modèle sont liés à travers la nouvelle couche ViewModel.

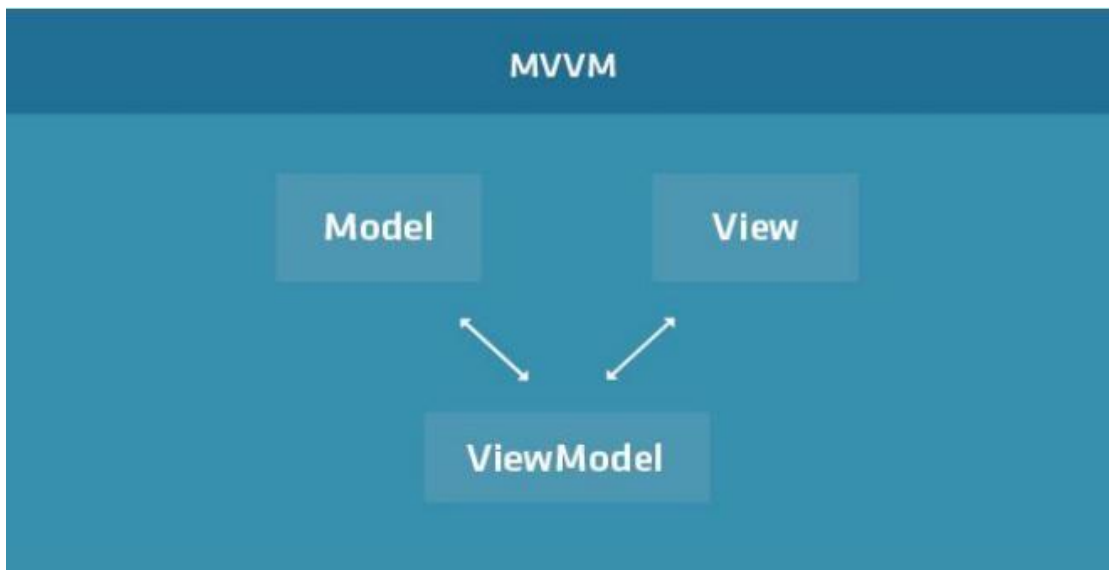


Figure 24: Le patron de conception MVVM

Les contrôleurs sont aussi existants dans l'architecture MVVM et permettent l'écoute des événements venant des vues et exécute une logique pour répondre à ces événements. Les contrôleurs sont appelé View Controller dans l'architecture MVVM et sont créés ou détruits lorsque la vue associé est créé ou détruite.

L'architecture MVVM devient ainsi :

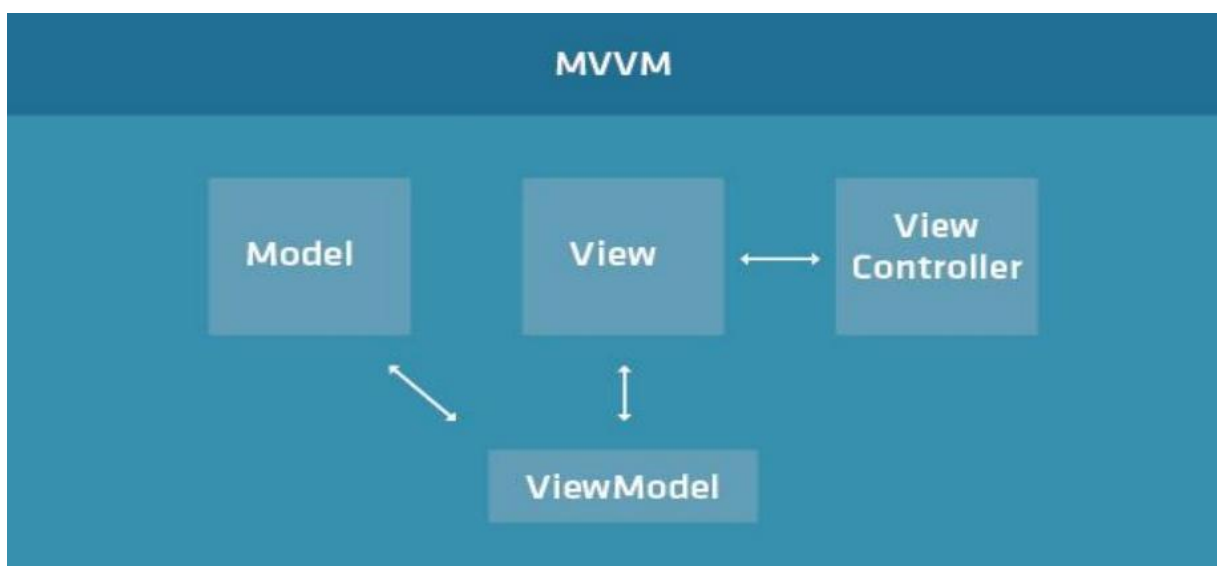


Figure 25: Les composants du patron de conception MVVM

Dans le contexte de notre projet, les deux patrons de conception sont adoptés :

MVC : est utilisé pour la mise en place de la couche des services web. Cette dernière ne contient aucune vue, elle contient juste des modèles et des contrôleurs.

L'inconvénient du MVC est l'absence de la synchronisation des données entre la vue et le contrôleur.

MVVM : est utilisé dans les autres parties clients mobile.

III. Outils et technologie de développement :

✕ JAVA



Le langage Java est un langage de programmation informatique orienté objet créé par Sun Microsystems.

La société Sun a été ensuite rachetée en 2009 par la société Oracle qui détient et maintient désormais Java.

La particularité et l'objectif central de Java est que les logiciels écrits dans ce langage doivent être très facilement portables sur plusieurs systèmes d'exploitation tels que UNIX, Windows, Mac OS ou GNU/Linux, avec peu ou pas de modifications. Pour cela, divers plateformes et frameworks associés visent à guider, sinon garantir, cette portabilité des applications développées en Java [[https://fr.wikipedia.org/wiki/Java_\(langage\)](https://fr.wikipedia.org/wiki/Java_(langage))].

✕ C#



Le langage de programmation C# a été développé par la société Microsoft. ce langage est orienté objet, avec un typage fort. Il est très proche du langage Java.

Il est précompilé en MSIL (Microsoft Intermediate Language), puis exécuté sur une machine virtuelle, ou compilé en code natif à l'exécution. Il dispose d'un ramasse-miettes (garbage collector). Il utilise l'API .NET en remplacement des MFC (Microsoft foundation class). Il semble être le nouveau langage pour développer des applications Windows, avec Visual Basic et C++ [https://fr.wikibooks.org/wiki/Programmation_C_sharp/Introduction].

✕ Android



Pour la partie mobile, on a opté pour le développement JAVA natif côté android et une base de données noSql Realmlocale.

Android a été conçu pour fonctionner d'une façon modulaire, ceci en lui-même présente un a tout indéniable, hélas la fragmentation dont souffre le système de google engendre des problèmes d'incompatibilité majeurs, notre mission à ce niveau était d'assurer la comptabilité des solutions mise en œuvre avec la majorité des versions du système afin de cibler la totalité des utilisateurs.

✕ Microsoft Azure



Microsoft Azure (Windows Azure jusqu'en 2014) est le nom de la plate-forme applicative en nuage de Microsoft. Son nom évoque le concept de « cloud computing » ou informatique en nuage (l'externalisation des ressources informatiques d'une entreprise vers des data centers distants).

Il s'agit d'une offre d'hébergement (applications et données) et de services (workflow, stockage et synchronisation des données, bus de messages, contacts...). Un ensemble d'API permet d'utiliser et d'accéder à cette plateforme et aux services associés. Un environnement d'exécution (le « Live Operating Environment ») permet une intégration étroite avec les principaux systèmes d'exploitation existant (Windows, Mac OS et Windows Phone).

La plateforme Windows Azure correspond aux offres d'informatique en nuage publics de type PaaS (maintenant) et IaaS de Microsoft [https://fr.wikipedia.org/wiki/Microsoft_Azure].

✕ Trello



Trello est une application permettant de gérer des projets et des tâches en équipe. Elle propose la notion des colonnes et des cartes (cards) pour permettre aux membres de l'équipe de différencier entre les tâches à faire, les tâches qui se font, et les tâches qui sont déjà terminées. Chaque carte contient l'intitulé de la tâche, la date d'échéance et les membres de l'équipe qui s'en chargeraient de la réalisation.

Cet outil nous a permis de bien éclaircir les tâches avec leurs durées, les personnes qui travaillent sur ces tâches, et aussi avoir une idée à n'importe quel moment sur l'évolution du projet [<http://coreight.com/content/trello-gestion-projet-taches>].

Principales fonctionnalités

- + Assigner un ou des membres à une board / card
- + Joindre des fichiers à une card
- + Commenter une card
- + Ajouter une liste de tâches
- + Mettre une date limite
- + Assigner des labels de couleur (important, urgent ...)
- + Déplacement facile d'une card d'une colonne à une autre par glisser / déposer
- + Création illimitée de boards, de colonnes et de cards
- + Système de notifications très efficace
- + Raccourcis claviers très pratiques

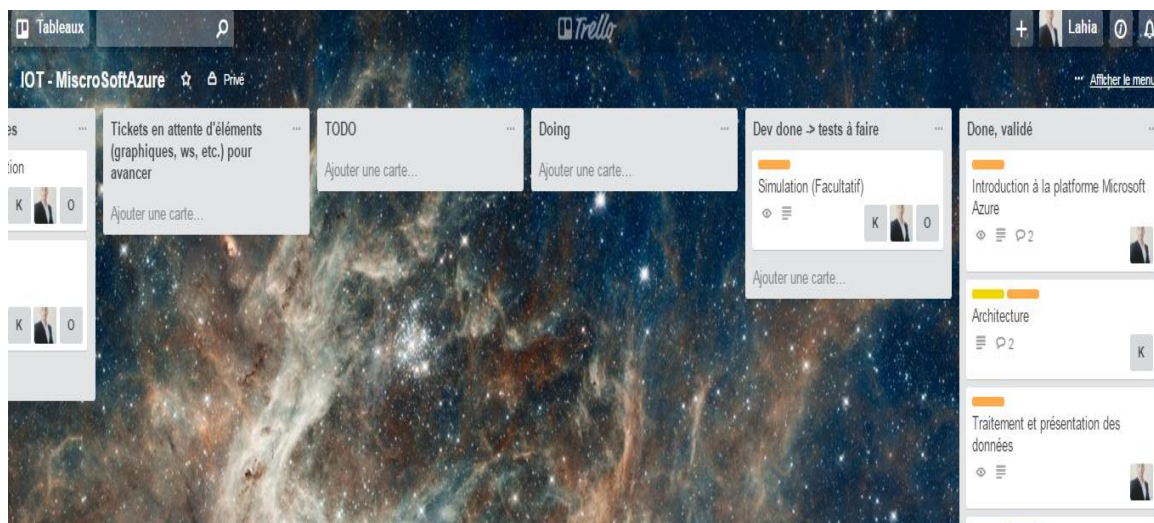


Figure 26:Exemple des taches affectées dans Trello

 Slack



Slack est une plateforme de communication et de collaboration qui rassemble les membres de l'équipe pour centraliser le suivi et la gestion d'un projet.

Cet outil était vraiment très utile pour nous, car il pousse les membres de l'équipe à bien collaborer et discuter entre eux à travers des channels (privées ou publiques) les différentes problématiques et nouveautés en relation avec le développement du projet.

On a beaucoup apprécié cet outil, parce que vraiment il propose un ensemble de fonctionnalités qui facilitent le partage des informations, de code, des couleurs, et des captures ...[http://www.atelier.net/trends/articles/slack-un-outil-de-collaboration-centralise-services-de-nombreuses-applications_428155]



L'outil GitLab, a été mis en œuvre pour gérer le versioning et garder une image claire sur les étapes de développement. L'une de ses atouts, est de pouvoir retourner en arrière vers des points de restauration de code si jamais on en a besoin.

En plus, GitLab permet la gestion des erreurs dans les projets. Chose qui facilite la tâche de validation des itérations dans le cycle de développement [<https://gitlab.com/>].



Git est un logiciel de gestion de versions décentralisé. C'est un logiciel libre créé par Linus Torvalds, auteur du noyau Linux, et distribué selon les termes de la licence publique générale GNU version 2.

Similaire en cela à BitKeeper, Git ne repose pas sur un serveur centralisé. C'est un outil de bas niveau, qui se veut simple et performant, dont la principale tâche est de gérer l'évolution du contenu d'une arborescence [<https://fr.wikipedia.org/wiki/Git>].



SourceTree permet de fonctionner avec git sans ligne de commande, il simplifie la façon dont nous interagissons avec les dépôts Git afin que nous puissions nous concentrer sur le codage [<https://www.sourcetreeapp.com/>].

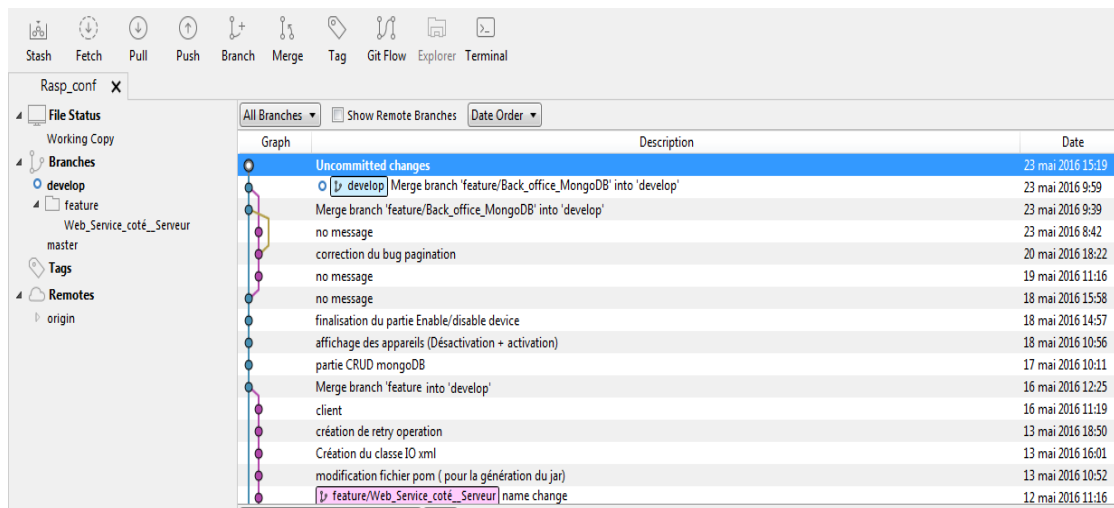


Figure 27: Exemple de gestion de notre projet avec sourceTree

✕ Realm



Realm est une base de données noSql mobile qui fonctionne directement dans les téléphones, tablettes ou wearables. Realm est plus rapide que même SQLite sur les opérations courantes, tout en maintenant un très riche ensemble de fonctionnalités [<https://realm.io/>].

✕ Stream analytics



Azure Stream Analytics (ASA) est un moteur de traitement des événements en temps réel entièrement géré et peu coûteux qui permet d'obtenir des informations détaillées à partir des données. Stream Analytics facilite la configuration de calculs d'analyse en temps réel sur les données de diffusion à partir d'appareils, de capteurs, de sites web, de médias sociaux, d'applications [<https://azure.microsoft.com/fr-fr/documentation/articles/stream-analytics-introduction/>].

- + **Simplicité d'utilisation** : Stream Analytics prend en charge un modèle de requête simple et déclaratif pour la description des transformations.
- + **Évolutivité** : Stream Analytics est capable de gérer un débit d'événements élevé allant jusqu'à 1 Go/s. L'intégration à Azure Event Hubs permet à la solution de recevoir des millions d'événements par seconde. Ces événements peuvent provenir d'appareils connectés, de flux de clics, de fichiers journaux,

- + **Connectivité** : Stream Analytics se connecte directement à Azure Event Hubs pour la réception de flux de données et au service blob Azure pour la réception de données d'historique. Les résultats peuvent être écrits à partir de Stream Analytics vers les objets blob ou tables de stockage Azure, les bases de données Azure SQL DB, les hubs d'événements Event Hubs, les rubriques ou files d'attente Service Bus Azure et Power BI, où ils peuvent être visualisés, faire l'objet d'un traitement plus poussé par les workflows, être utilisés dans les analyses par lot via Azure HDInsight ou être traités à nouveau en tant que série d'événements.

IV. Présentation de l'application réalisée

Dans ce qui suit, la présentation de quelques captures d'écrans va représenter le fruit de la conception et du développement de ce projet, ce dernier est divisé en trois parties cruciales : la partie cloud, communication entre l'appareil de simulation et le cloud et enfin l'application mobile.

1. Partie Cloud

✕ Visualisation des graphes

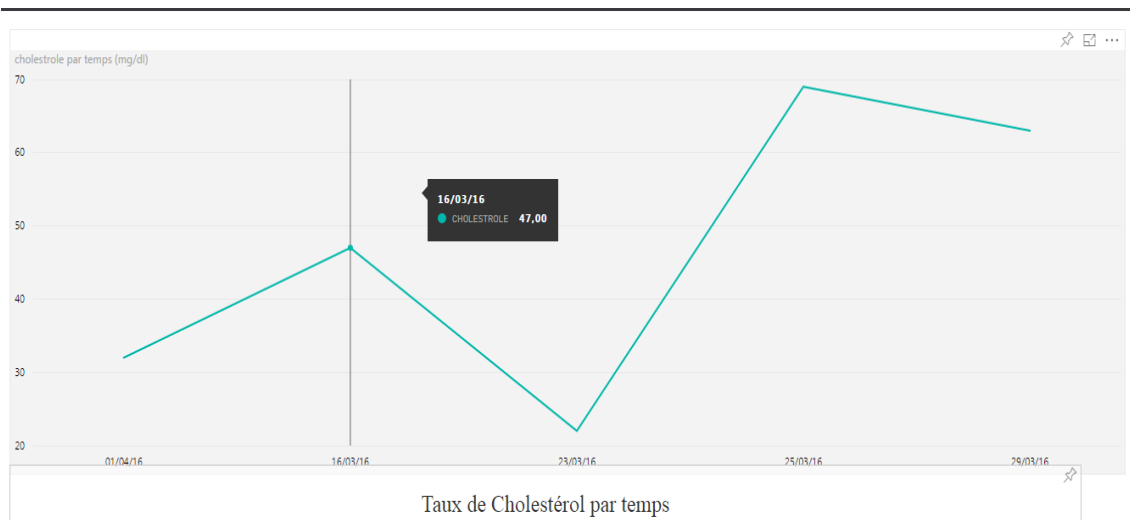
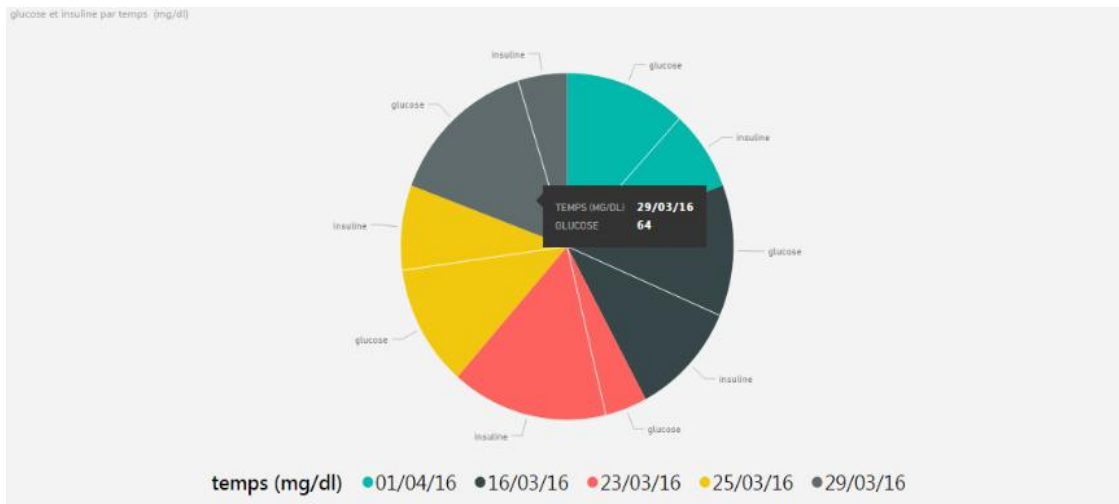
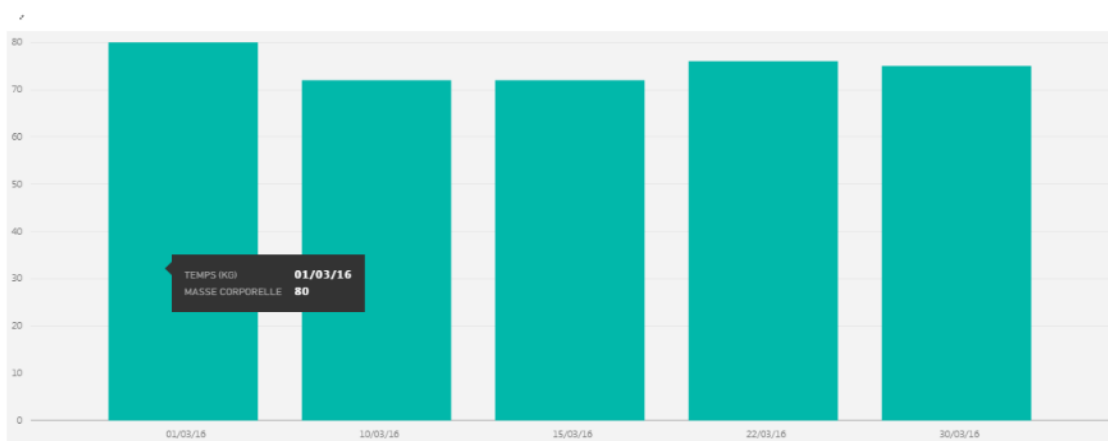


Figure 28: Taux de cholestérol par temps



Taux de glucose et insuline par Temps

Figure 29:Taux de glucose et insuline par temps



L'évolution de masse corporelle par Temps

Figure 30:l'évolution de masse corporelle par temps

2. Partie Envoi et réception des données

✗ Connexion à l'IoT hub

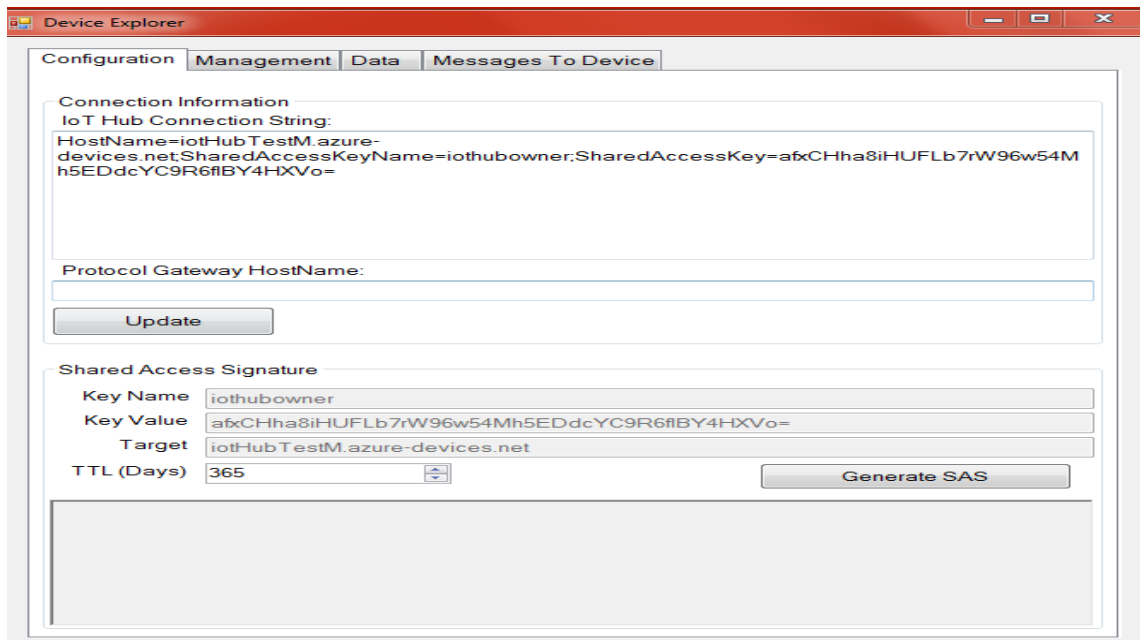


Figure 32:Device explorer

Cette interface représente le device explorer qui permet de créer et gérer les appareils.

✗ Envoi des données

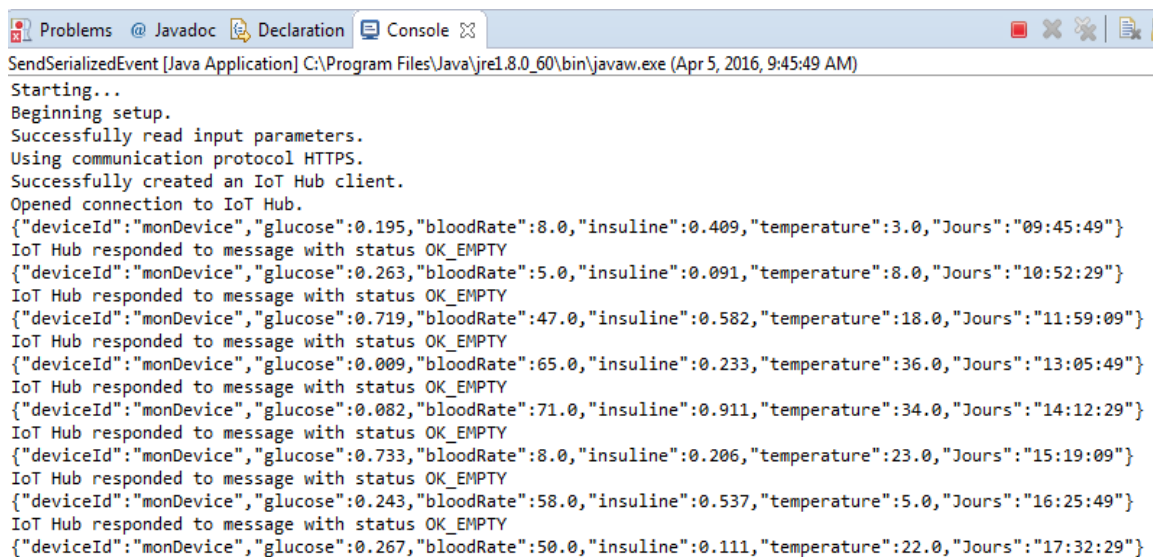


Figure 33:Simulation d'envoi des données de télémétrie

Cette interface représente la simulation d'envoi des données vers le cloud en utilisant les protocoles IoT (MQTT, AMQP, HTTP)

✗ Réception des données

```

file:///C:/Users/marouane/Documents/Visual Studio 2015/Projects/ProcessDeviceToCloudMessage...
2,"bloodRate":13.0,"insuline":0.606,"temperature":17.0,"Jours":"21:59:09"}
Message received. Partition: '0', Data: '{"deviceId":"monDevice","glucose":0.77
2,"bloodRate":28.0,"insuline":0.165,"temperature":20.0,"Jours":"23:05:49"}
Message received. Partition: '0', Data: '{"deviceId":"monDevice","glucose":0.09
5,"bloodRate":22.0,"insuline":0.827,"temperature":7.0,"Jours":"00:12:29"}
Message received. Partition: '0', Data: '{"deviceId":"monDevice","glucose":0.29
1,"bloodRate":15.0,"insuline":0.825,"temperature":28.0,"Jours":"01:19:09"}
Message received. Partition: '0', Data: '{"deviceId":"monDevice","glucose":0.05
3,"bloodRate":13.0,"insuline":0.56,"temperature":13.0,"Jours":"02:25:49"}
Message received. Partition: '0', Data: '{"deviceId":"monDevice","glucose":0.21
6,"bloodRate":48.0,"insuline":0.353,"temperature":6.0,"Jours":"03:32:29"}
Message received. Partition: '0', Data: '{"deviceId":"monDevice","glucose":0.96
9,"bloodRate":62.0,"insuline":0.816,"temperature":17.0,"Jours":"04:39:09"}
Message received. Partition: '0', Data: '{"deviceId":"monDevice","glucose":0.63
7,"bloodRate":59.0,"insuline":0.113,"temperature":23.0,"Jours":"05:45:49"}
Message received. Partition: '0', Data: '{"deviceId":"monDevice","glucose":0.70
3,"bloodRate":4.0,"insuline":0.572,"temperature":3.0,"Jours":"06:52:29"}
Message received. Partition: '0', Data: '{"deviceId":"monDevice","glucose":0.80
8,"bloodRate":45.0,"insuline":0.098,"temperature":17.0,"Jours":"07:59:09"}
Message received. Partition: '0', Data: '{"deviceId":"monDevice","glucose":0.28
2,"bloodRate":61.0,"insuline":0.651,"temperature":12.0,"Jours":"09:05:49"}

Processor Shutting Down. Partition '1', Reason: 'Shutdown'.
Processor Shutting Down. Partition '0', Reason: 'Shutdown'.

```

Figure 35: Réception des données

La réception de données envoyées vers le cloud et le stockage dans des fichiers Blob.

3. Partie mobile

✕ Page d'accueil

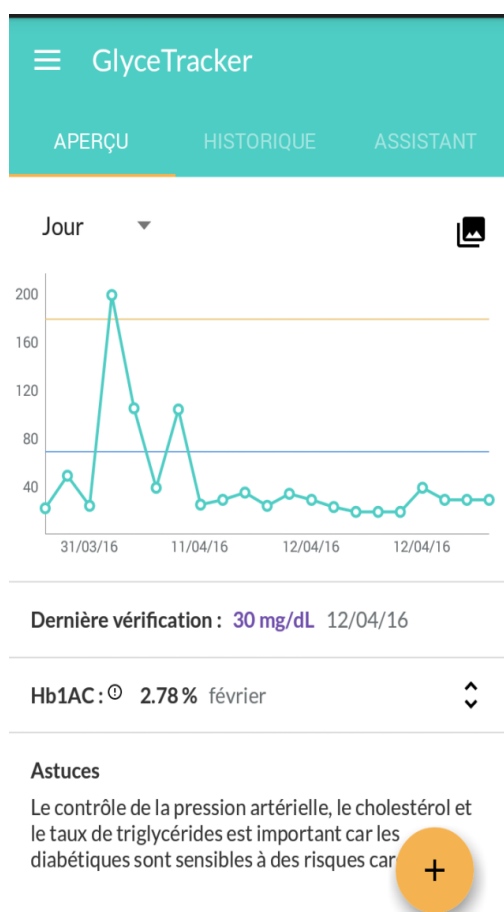


Figure 36: interface d'accueil

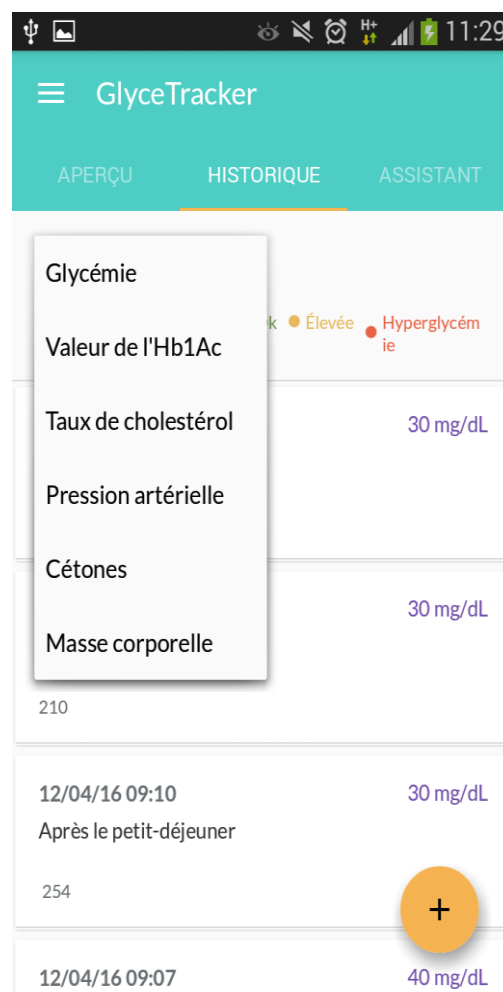


Figure 37: Historique

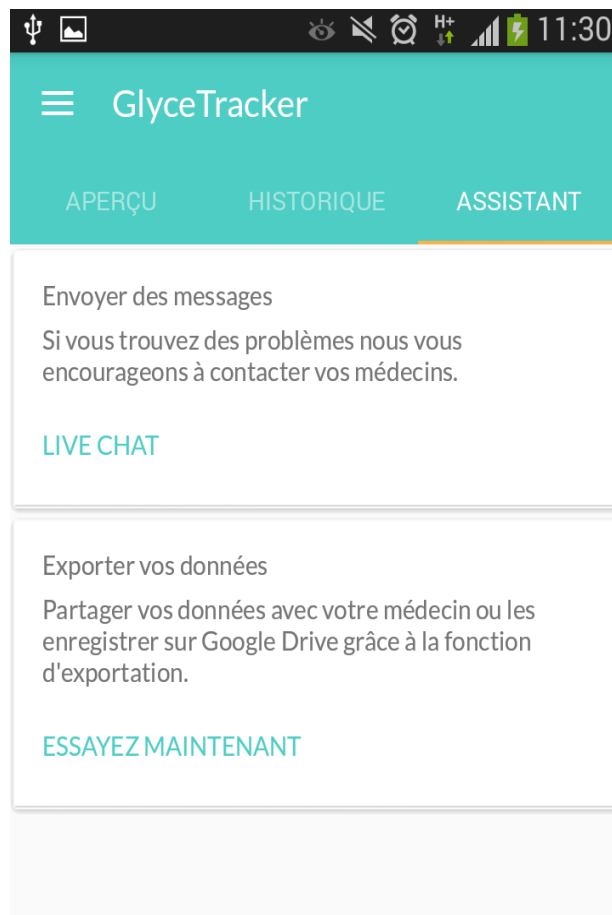


Figure 38:Interface d'assistant

L'interface d'accueil de l'application de l'enseignant lui permet d'avoir une idée sur l'état du patient ainsi que l'onglet historique permet de consulter les mesures déjà effectuées et le degré de risque.

✕ Page de chat

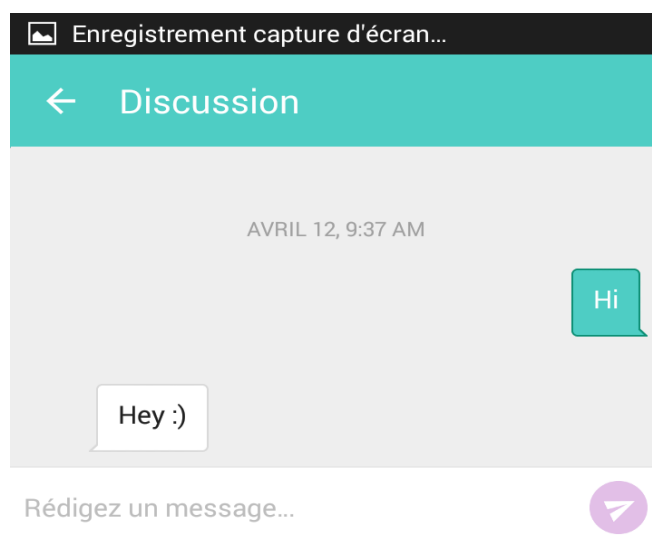
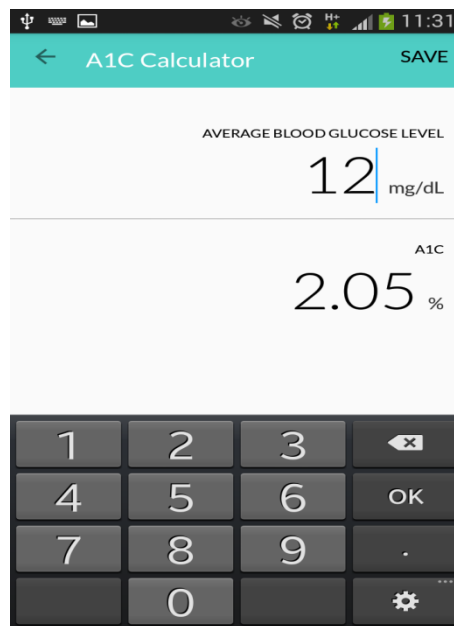


Figure 39:page de chat

Système de chat entre les utilisateurs de l'application.

✕ A1C Calculator

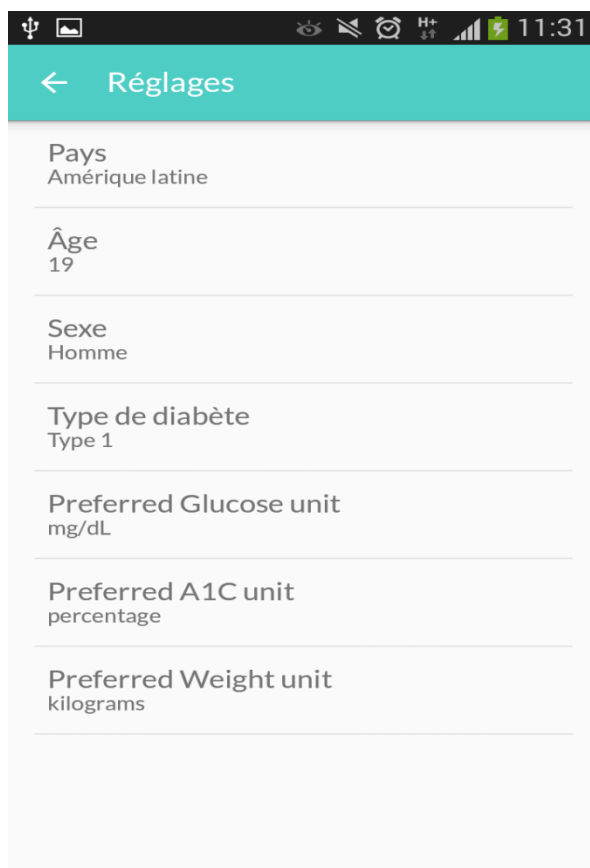


The screenshot shows the 'A1C Calculator' app interface. At the top, there is a teal header with a back arrow, the title 'A1C Calculator', and a 'SAVE' button. Below the header, the text 'AVERAGE BLOOD GLUCOSE LEVEL' is displayed above a large input field containing the number '12' followed by 'mg/dL'. Below this, the text 'A1C' is displayed above a large output field containing the number '2.05' followed by a percentage sign '%'. At the bottom, there is a numeric keypad with buttons for digits 1-9, 0, a decimal point, and an 'OK' button. There are also buttons for a back arrow and a settings gear icon.

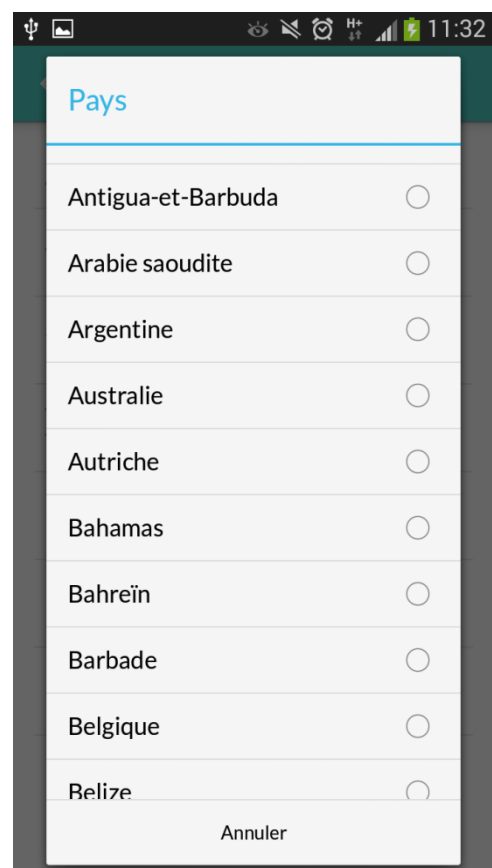
Figure 40: Calcul de Hb1AC

Permet d'évaluer l'équilibre glycémique sur une plus longue période (environ 2 à 3 mois).

✕ Page de réglages



The screenshot shows the 'Réglages' (Settings) app interface. It has a teal header with a back arrow and the title 'Réglages'. Below the header, there are several settings items, each with a label and a value: 'Pays' (Amérique latine), 'Âge' (19), 'Sexe' (Homme), 'Type de diabète' (Type 1), 'Preferred Glucose unit' (mg/dL), 'Preferred A1C unit' (percentage), and 'Preferred Weight unit' (kilograms).



The screenshot shows a dialog box for selecting a country. The title is 'Pays'. It contains a list of countries with radio buttons next to them: Antigua-et-Barbuda, Arabie saoudite, Argentine, Australie, Autriche, Bahamas, Bahreïn, Barbade, Belgique, and Belize. At the bottom, there is an 'Annuler' (Cancel) button.

Figure 41: interface de réglages

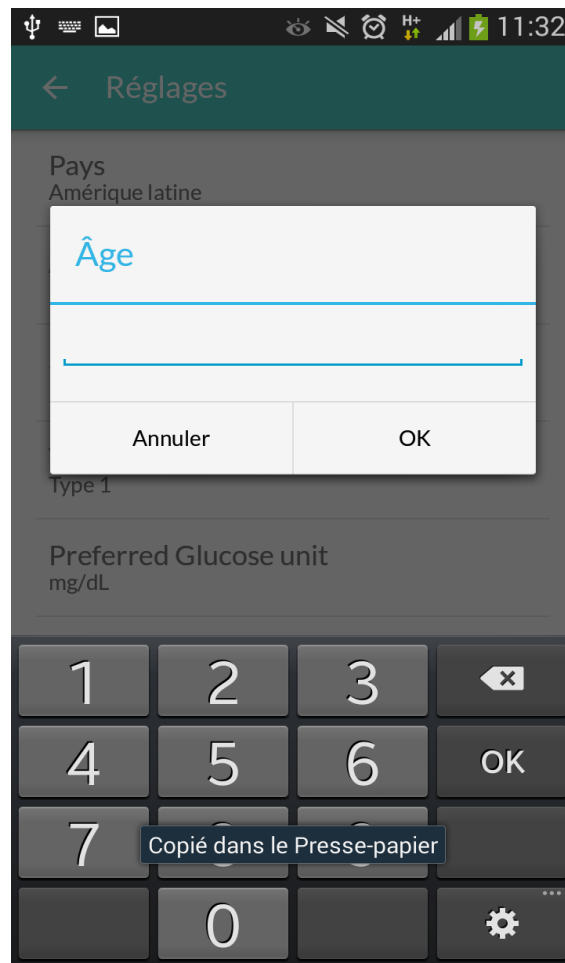


Figure 42: Modifier âge

L'interface de configuration permet de changer le pays de l'application, l'âge, le sexe ...

✕ Exporter les données

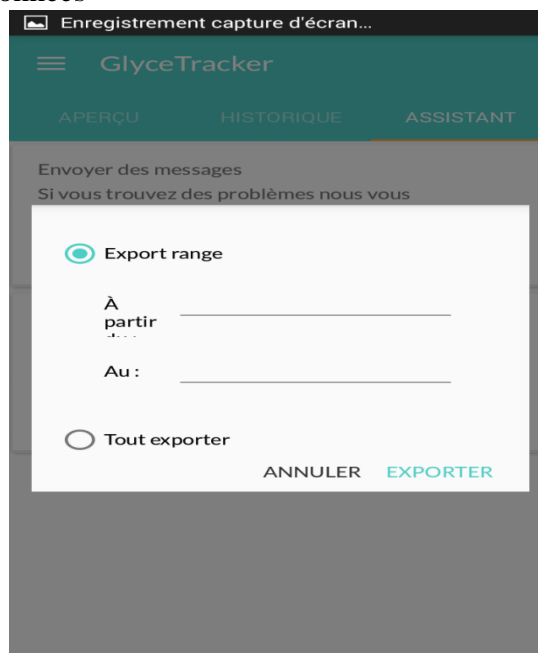


Figure 43: Exporter les données



Conclusion

Durant ce chapitre, nous avons présenté l'architecture du projet, les différents outils utilisés pour le développement de notre système, ainsi qu'un aperçu des principaux écrans de celui-ci.

Conclusion générale

Nous entendons toujours les développeurs dirent qu'un programme qui tourne correctement est la meilleure récompense au programmeur.

Par ce dire, nous allons terminer ce rapport, qui est une illustration, une transcription de notre stage de fin d'études au sein de la société SQLI, période durant laquelle nous avons pu consolider, voire combler certaines de nos connaissances en informatique et vivre une transition vers le monde professionnel.

L'objectif du projet était la mise en œuvre d'un système de surveillance de la glycémie en continu destiné aux patients atteints de diabète, ce projet est basé sur Microsoft Azure et les objets connectés.


La mission a été déclinée en trois grandes phases. Dans une première étape, une étude fonctionnelle a été menée durant laquelle le dossier de spécifications fonctionnelles détaillées a été rédigé. Cette phase a ensuite, donné lieu à l'étape d'analyse où les modèles de la solution ont été élaborés.

Une troisième phase a consisté en une étude technique et benchmarking qui comprend la capture des besoins techniques et la rédaction des spécifications techniques détaillées.

La quatrième phase traite la conception du système dans le but de fournir une image prête pour développer la solution.

Et enfin, la cinquième et dernière phase a été consacrée à la réalisation de l'application mobile ainsi que l'analyse et communication des objets (appareil de simulation) avec le cloud.

Ce stage nous a permis de développer notre sens de travail en équipe, tout en restant autonomes, ainsi que notre capacité d'adaptation, de communication et d'intégration dans le milieu professionnel. Ce stage fut très bénéfique pour nous en termes de valeur professionnelle. Sur le plan technique, nous avons pu élargir nos connaissances en travaillant sur de nouvelles technologies, notamment celle des applications mobiles et IoT. Sur le plan métier, nous avons découvert les principes de gestion de projets dans un organisme bien



structuré. Ainsi nous avons eu la chance d'améliorer nos aptitudes à communiquer et à travailler en groupe, ce qui nous aidera à bien améliorer notre travail professionnel.

En perspective pour ce projet, nous envisageons tout d'abord faire de la machine learning sur les données collecter, ensuite compléter quelques taches relatives au web service et sécurité enfin nous volons remplacer les appareils de simulation par des capteurs réels.

Pour finir, nous souhaitons que l'application réalisée apporte une grande valeur ajoutée à la bonne marche des diabétiques, et nous espérons que la solution que nous avons conçue contribuera à promouvoir le suivi du diabète.

Webographie

<https://azure.microsoft.com/fr-fr/documentation/articles/iot-hub-csharp-csharp-process-d2c/>

<https://www.google.com/design/spec/material-design/introduction.html>

<http://developer.android.com/tools/studio/index.html>

<https://www.genymotion.com/>

<https://developer.android.com/studio/index.html>

<http://www.mkylong.com/tutorials/android-tutorial/>

<https://github.com/wasabeef/awesome-android-ui>