République Algérienne Démocratique et Populaire Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mohamed Sadik Benyahia –Jijel

Faculté des Sciences Exactes et d'Informatique

Département d'Informatique

Mémoire de fin d'étude

Pour obtention du diplôme Master de Recherche en Informatique

Option : Système d'information et aide à la décision

Thème

Intégration d'une Application Mobile dans un Système de Gestion Scolaire

Réalisé par :

BELMERABET Islam

MEZERREG Houssam-Eddine

Encadré par :

Mr Mokhtar TAFFAR

Promotion : 2019/2020

Remerciement

En tout premier lieu, Nous tenons à remercier en premier temps le bon Dieu, tout puissant (Allah), qui nous a donné la force et la patience pour survivre et dépasser toutes les difficultés et accomplir ce Modeste travail.

Nous exprimons nos remerciements à cœur de joie à tous les professeurs, chefs de travaux et aux assistants pour tous les enseignements bien riches et l'encadrement adéquat qu'ils nous ont fournis tout au long de notre formation.

Nos vifs remerciements vont tout droit au Chef de Travaux MR._ TAFER pour nous avoir dirigés malgré ses multiples occupations, à tous les assistants et chefs de travaux qui nous ont enseigné.

Nous tenons à remercier très sincèrement l'ensemble des membres du jury qui nous fussions le grand honneur d'accepter de juger notre travail.

Enfin, à tous ceux de près ou de loin, ont apporté leur pierre pour notre formation, dont leurs noms ne figurent pas sur cette liste nous les portons toujours au cœur.

Dédicaces

Nous dédions ce modeste travail à :

* * * Nos très chers parents * * *

* * * Nos frères et sœurs * * *

* * * Toutes nos familles * * *

* * * A tous nos amies * * *

* * * A tous nos enseignants, collègues * * *

Enfin, à tous ceux qui nous ont aidé à réaliser ce travail.

* * * * * * *

Table des matières :

Liste des tableaux :

Liste des figures :

Résumé

Le développement d'applications pour mobiles a de beaux jours devant lui. La mobilité devient un facteur clé de l'informatique ce qui permet aux utilisateurs d'accéder à leurs systèmes d'information n'importe quand et à n'importe où. Les téléphones deviennent de véritables petits ordinateurs et offrent des capacités encore sous-exploitées.

Dans ce contexte, nous proposons l'étude, la conception, et l'implémentation d'une application mobile et l'intégrer dans un système de gestion scolaire, pour offrir l'ensemble de fonctionnalités du système selon ses différents types des utilisateurs.

Mots clés : Gestion Scolaire, Développement Mobile Hybride, React-Native, RESTful API

Introduction Générale

De nos jours, de plus en plus de personnes possèdent un Smartphone ou encore une tablette. Le développement mobile est un phénomène qui a pris beaucoup d'ampleur en quelques années. Pour les entreprises, suivre cette tendance est important puisqu'elle leur permet de créer des services innovants ou encore d'améliorer leur communication.

Avec l'avènement des réseaux de communications GSM puissants et de très haut débit tels ceux de la 4G et bientôt de la 5G, le monde informatique se convertit définitivement à l'usage des plates-formes mobiles comme support primordial de conception, de développement et d'usage banalisé sur Smartphone. Le développement mobile ou la création des applications mobiles est un atout essentiel à tout produit informatique mis sur le marché par une entreprise. Tout d'abord, cela accroît les performances du produit et ses options d'utilisation ainsi que sa productivité, sa commercialisation et son usage à grande échelle sur le marché des systèmes informatique. Parler de productivité accrue nous entraine à parler de clients plus satisfaits et de chiffre d'affaires en augmentation. De même, cela signifie moins de coûts administratifs et moins de papiers à remplir.

Toute cette dynamique de développement et d'usage de plate-forme mobile permet à une entreprise de proposer des services innovants et une meilleure communication avec ses employés, ses clients et ses fournisseurs. Ce n'est que sous sa forme mobile ou du moins comme extension d'un système informatique existant que l'application prendra tout son intérêt. Elle complète, de ce fait, le système en lui assurant des options à distance à travers les réseaux de téléphonie mobile.

Objectif

Notre projet s'inscrit dans le cadre décrit précédemment, il consiste à réaliser une solution mobile pour un système de Gestion Scolaire, pour ce dernier on a choisi le Système de Gestion Scolaire (School Management System).(1)

School Management System est un système de gestion d'école polyvalent qui aide à automatiser les opérations quotidiennes de l'école.

Notre solution mobile est une application mobile qui gère ce système à partir d'un Smartphone. Pour cela nous avons développé une **API** nommée **Restful**et une interface.

Organisation du mémoire

Nous avons organisé notre mémoire en Cinq chapitres :

- ✓ Le premier chapitre est consacré à la présentation générale et la définition d'un système d'information et ses objectifs, sa place dans l'organisation et l'intégration de la notion de mobilité, avec ces outils de développements tels que le génie logiciel, UML et ses modèles, les cycles de vie, ainsi que les notions qui lui sont liées tels que ses fonctions.

- ✓ Le deuxième chapitre est présente des notions sur les appareils mobiles –Smartphones et leurs Applications avec une introduction et des définitions des Smartphones, des systèmes d'exploitation mobiles, des applications mobiles, ainsi que leurs caractéristiques et leurs différents types.

- ✓ Le troisième chapitre expose les Framework multi-plate-forme (Hybrides) avec une comparaison des différents types de Framework.

- ✓ Le quatrième chapitre est consacré aux services web et à l'***API RESTful.***

- ✓ Le cinquième chapitre détaille l'architecture de notre application mobile avec une représentation de L'API, l'interface et ses fonctionnalités en spécifiant les outils et les technologies utilisés.

- ✓ Enfin, nous terminerons notre mémoire par une conclusion générale.

Chapitre 1 : Système d'information et leur développement:

1 Introduction

Aujourd'hui, grâce aux nouvelles technologies, les Systèmes d'information représentent des outils extrêmement puissants en matière de gestion d'entreprise dans tous les métiers. Ils jouent un rôle principal dans l'amélioration de ces performances et sa réactivité en permettant de gérer les différents flux d'information présents dans toute entité.

1.1 Système d'information

Le système d'information (SI) est l'ensemble des méthodes, techniques et outils pour la mise en place et l'exploitation de la technologie informatique nécessaire aux utilisateurs et à la stratégie de l'entreprise.(2)

1.1.1Les objectifs du système d'information

Le système d'information à la particularité d'être une fonction ressource pour l'entreprise dans son ensemble. Dans cette optique systémique, nous pouvons représenter le système d'information par cinq objectifs majeurs au service du système opérationnel, de gestion et décisionnel d'une entreprise, comme l'illustre la figure 1.(2)

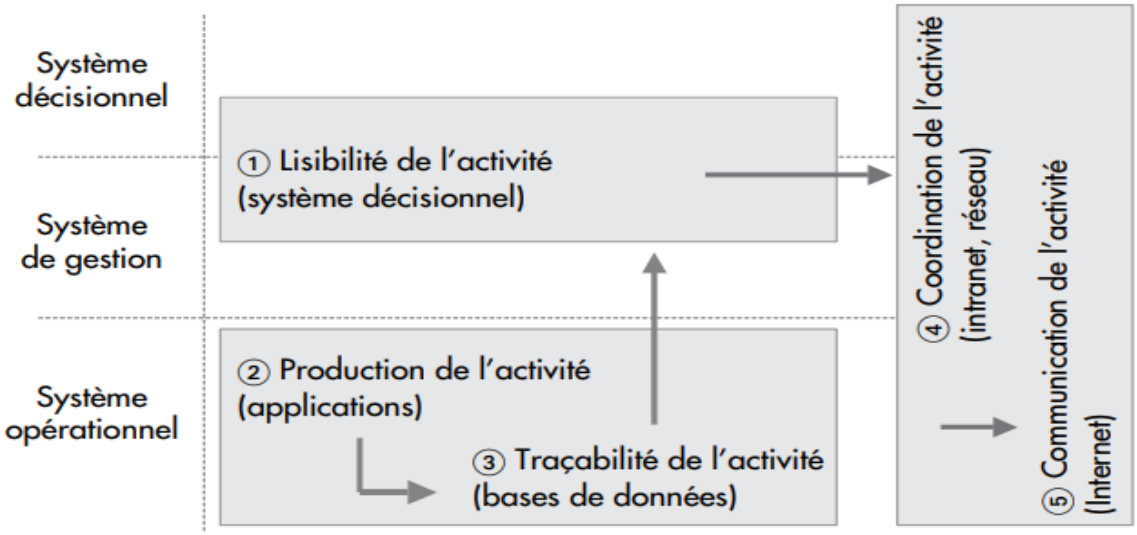


Figure 1-1 : Les cinq objectifs du SI (2)

1.1.2Les principales sources de système information

Données :sous des formes variées (chiffre, texte, images, sons …) ses ressources essentiellement matérialiser l'information détenue par l'organisation. Elles sont la matière première sur laquelle le système d'information agit, elles sont traitées à l'aide de model qui expriment des connaissances, et permet de déduire un résultat ou une action.(3)

Logiciel et procédure : il constitue la description formelle des opérations effectuée(les programmes : les système d'exploitations, traitement des textes, feuille de paie. Les procédures : saisies, correction d'erreur distribution des chèques de paie). (3)

Matériel : le système d'information repose des technologies numériques de l'information (réseaux ordinateur, unité périphériques, station de travail, papier…). (3)

Personne : il y a pas du système d'information sans personne, sans acteur : se sont soit des utilisateurs de système, employés, cadre qui pour la réalisation de leurs taches, utilisent l'information produit par le système et ces possibilités d'automatisation ou qui alimente le système.(3)

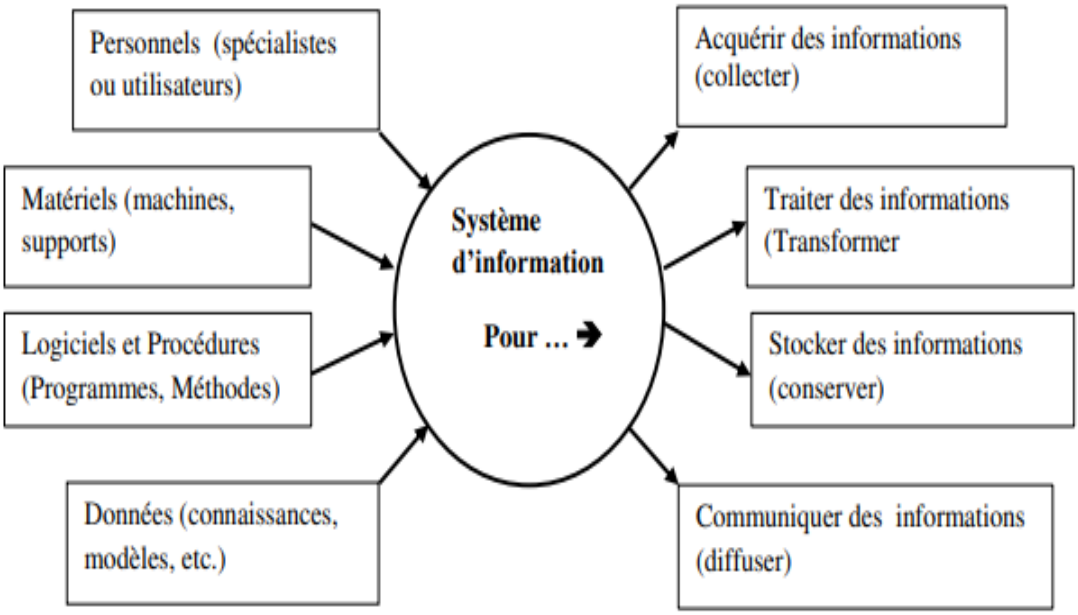


Figure 1-2 : source de système d'information(3)

1.1.3Les fonctions d'un système d'information

Dans un système d'information, il y'a trois activités principales qui aident à la production de l'information à l'organisation : l'entrée, le traitement et la sortie. (4)

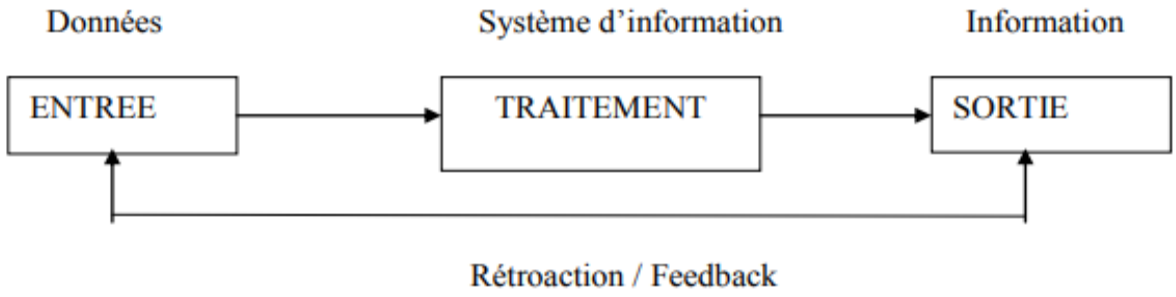


Figure 1-3 : Les fonctions d'un système d'information(4)

- L'Entrée, est le processus au cours duquel les données brutes son fournies au système. Ce processus peut prendre des formes –différenciées.(4)
- Le Traitement, est le processus qui transforme les données brutes pour leur donner un sens. (4)
- La Sortie, est le processus de diffusion de l'information traitée aux utilisateurs qui ont besoin.(4)

Remarque : Un système d'information se fonde également sur la rétroaction ou le feedback, c'est-à-dire sur le processus de transmission des informations de sortie aux utilisateurs appropriées pour les aider à évaluer l'étape antérieure et à y intervenir à nouveau si besoin (pour mise à jour, par exemple).(4)

1.1.4Le rôle de système d'information

Le système d'information est une aide pour la prise de décision : Le Système d'Information permet aux responsables d'obtenir les informations qui leurs sont nécessaires pour les prises de décision .Ils vont pouvoir étudier plus facilement les conséquences possibles de leur décision .le Système d'Information va aussi permettre d'automatiser certain décisions. (3)

Le Système d'Information est un outil de contrôle de l'évolution d'organisation : Le Système d'Information va permettre de détecter des dysfonctionnements interne ou des situations anormal pour que cet outil soit opérationnel ; le Système d'Information doit être la « mémoire collective » de l'organisation cela en gardant constamment une trace de chaque information. (3)

Le Système d'Information est un outil de coordination des déférentes activités de l'entreprise : Le Système d'Information va aussi fournir des informations sur le présent, elles seront les mêmes pour l'ensemble des services et seront mises à jour régulièrement .tout le monde est informé de mêmes manières selon son accès aux informations. (3)

1.1.5Les systèmes d'information au défi de la mobilité

La mobilité est sur le point de transformer radicalement le paysage des technologies de l'information et de la communication dans l'entreprise. Une révolution qui ouvre de réelles opportunités, mais pose aussi des questions de sécurité. D'ores et déjà, les premiers enseignements émergent pour mieux gérer les risques et les coûts associés à ce défi organisationnel.

Les directeurs des systèmes d'information (DSI) sont accoutumés à vivre avec le changement permanent. Systèmes centralisés, ordinateurs personnels et Internet ont successivement transformé les technologies de l'information, aussi bien pour les entreprises elles-mêmes que pour leurs fournisseurs de matériel et de services. La prochaine rupture, celle de la mobilité, arrive à grands pas – et les DSI l'ont désormais tous en ligne de mire.

Après des débuts modestes, la mobilité connaît un essor sans précédent, portée par des smartphones, tablettes et autres appareils toujours plus performants, grâce aux réseaux 3G et 4G ainsi qu'à l'explosion des applications innovantes. Dans les entreprises, les technologies de l'information et la communication sont à la veille d'une révolution qui promet de démultiplier la productivité en étendant le champ des fonctionnalités par-delà les limites physiques de bureaux jusqu'à présent circonscrits à leur espace physique. Toutefois, si les opportunités sont légion, il faudra également composer avec les défis en matière de coûts, de gouvernance et de sécurité. (5)

1.1.6Les avantages de la mobilité pour les entreprises

La mobilité peut servir les entreprises dans quatre domaines principaux.

- ***La communication des salariés***. Un accès amélioré aux e-mails et aux calendriers, ainsi que des applications vocales, vidéo et de messagerie vont renforcer la communication interne. A titre d'exemple, on pourra voir des vidéoconférences mobiles s'organiser spontanément.
- ***La productivité pendant les déplacements***. L'accès à distance aux contenus et aux applications permettra aux salariés de mettre pleinement leur temps à profit lorsqu'ils seront en déplacement. En fournissant un accès mobile au CRM, à un progiciel de gestion intégrée (PGI/ERP) et à des tableaux de bord, par exemple, on améliore la productivité des salariés dans le cœur de métier. Pour les collaborateurs dont le temps travaillé se fait par définition hors des bureaux (par exemple, la force de vente et les employés sur le terrain), les technologies mobiles améliorent leur productivité en amenant sur le terrain des informations et des ressources avec une facilité inédite, cela vaut également pour les tâches administratives.
- ***Les réseaux de capteurs*.** Des capteurs intelligents peuvent automatiser ou contrôler les processus et les systèmes, les rendant ainsi plus efficaces. Les capteurs peuvent aussi conférer aux produits de nouvelles capacités et susciter de nouveaux modèles d'affaires. En matière de soins de santé par exemple, des capteurs utilisés ou portés par les patients rapportent en permanence des variations des paramètres vitaux aux médecins, qui peuvent ajuster les traitements ou, le cas échéant, intervenir auprès du patient de manière proactive.
- ***Un nouveau vecteur pour atteindre les clients*.** Les TIC mobiles ne servent pas qu'à améliorer la productivité : en développant quantitativement et qualitativement les points de contact, les innovations de mobilité peuvent permettre aux entreprises d'entrer en contact avec leurs clients d'une manière tout à fait nouvelle.(5)

1.2 Le génie logiciel

Les logiciels, suivant leur taille, peuvent être développés par une seule personne, une petite équipe, ou un ensemble d'équipes coordonnées. Le développement de grands logiciels par de grandes équipes pose d'importants problèmes de conception et de coordination. Or, le développement d'un logiciel est une phase absolument cruciale qui monopolise l'essentiel du coût d'un produit et conditionne sa réussite et sa pérennité. (6)

Le génie logiciel est un domaine de recherche qui a été défini en 1968, à Garmisch-Partenkirchen, sous le parrainage de l'OTAN. Il a pour objectif de répondre à un problème qui s'énonçait en deux constatations : d'une part le logiciel n'était pas fiable, d'autre part, il était incroyablement difficile de réaliser dans des délais prévus des logiciels satisfaisant leur cahier des charges.(6)

1.2.1Objectif du génie logiciel

L'objectif du génie logiciel est d'optimiser le coût de développement du logiciel. L'importance d'une approche méthodologique s'est imposée à la suite de la crise de l'industrie du logiciel à la fin des années 1970. Cette crise de l'industrie du logiciel était principalement due à :

- l'augmentation des coûts ;
- les difficultés de maintenance et d'évolution ;
- la non-fiabilité ;
- le non-respect des spécifications ;
- le non-respect des délais.(6)

1.2.2Notion de qualité pour un logiciel

En génie logiciel, divers travaux ont mené à la définition de la qualité du logiciel en termes de facteurs, qui dépendent, entre autres, du domaine de l'application et des outils utilisés. Parmi ces derniers nous pouvons citer : (6)

Validité : aptitude d'un produit logiciel à remplir exactement ses fonctions, définies par le cahier des charges et les spécifications.

Fiabilité ou robustesse : aptitude d'un produit logiciel à fonctionner dans des conditions anormales.

Extensibilité (maintenance) : facilité avec laquelle un logiciel se prête à sa maintenance, c'est-à-dire à une modification ou à une extension des fonctions qui lui sont demandées.

Réutilisabilité : aptitude d'un logiciel à être réutilisé, en tout ou en partie, dans de nouvelles applications.

Compatibilité : facilité avec laquelle un logiciel peut être combiné avec d'autres logiciels.

Efficacité : Utilisation optimale des ressources matérielles.

Portabilité : facilité avec laquelle un logiciel peut être transféré sous différents environnements matériels et logiciels.

Vérifiabilité : facilité de préparation des procédures de test.

Intégrité : aptitude d'un logiciel à protéger son code et ses données contre des accès non autorisés.

Facilité d'emploi : facilité d'apprentissage, d'utilisation, de préparation des données, d'interprétation des erreurs et de rattrapage en cas d'erreur d'utilisation.

1.3 Cycle de vie d'un logiciel

Le cycle de vie d'un logiciel (en anglais software lifecycle), désigne toutes les étapes du développement d'un logiciel, de sa conception à sa disparition. L'objectif d'un tel découpage est de permettre de définir des jalons intermédiaires permettant la validation du développement logiciel, c'est-à-dire la conformité du logiciel avec les besoins exprimés, et la vérification du processus de développement, c'est-à-dire l'adéquation des méthodes mises en œuvre.L'origine de ce découpage provient du constat que les erreurs ont un coût d'autant plus élevé qu'elles sont détectées tardivement dans le processus de réalisation. Le cycle de vie permet de détecter les erreurs au plus tôt et ainsi de maîtriser la qualité du logiciel, les délais de sa réalisation et les coûts associés.(6)

1.3.1Etapes d'un cycle de vie

Le cycle de vie du logiciel comprend généralement au minimum les étapes suivantes : (6)

Analyse des besoins et faisabilité : c'est-à-dire l'expression, le recueil et la formalisation des besoins du demandeur (le client) et de l'ensemble des contraintes, puis l'estimation de la faisabilité de ces besoins.

Spécifications ou conception générale : il s'agit de l'élaboration des spécifications de l'architecture générale du logiciel.

Conception détaillée : cette étape consiste à définir précisément chaque sous-ensemble du logiciel.

Codage (Implémentation ou programmation) : c'est la traduction dans un langage de programmation des fonctionnalités définies lors de phases de conception.

Tests unitaires : ils permettent de vérifier individuellement que chaque sous-ensemble du logiciel est implémenté conformément aux spécifications.

Intégration : l'objectif est de s'assurer de l'interfaçage des différents éléments (modules) du logiciel. Elle fait l'objet de tests d'intégration consignés dans un document.

Qualification (ou recette) : c'est-à-dire la vérification de la conformité du logiciel aux spécifications initiales.

Documentation : elle vise à produire les informations nécessaires pour l'utilisation du logiciel et pour des développements ultérieurs.

Mise en production : c'est le déploiement sur site du logiciel.

Maintenance : elle comprend toutes les actions correctives (maintenance corrective) et évolutives (maintenance évolutive) sur le logiciel.

La séquence et la présence de chacune de ces activités dans le cycle de vie dépendent du choix d'un modèle de cycle de vie entre le client et l'équipe de développement. Le cycle de vie permet de prendre en compte, en plus des aspects techniques, l'organisation et les aspects humains. (6)

1.3.2 Modèle de cycle de vie d'un logiciel

1.3.2.1 Modèle du cycle de vie en cascade :

Le modèle de cycle de vie en cascade (cf. figure 5) a été mis au point dès 1966, puis formalisé aux alentours de 1970. Dans ce modèle le principe est très simple : chaque phase se termine à

une date précise par la production de certains documents ou logiciels. Les résultats sont définis sur la base des interactions entre étapes, ils sont soumis à une revue approfondie et on ne passe à la phase suivante que s'ils sont jugés satisfaisants.(6)

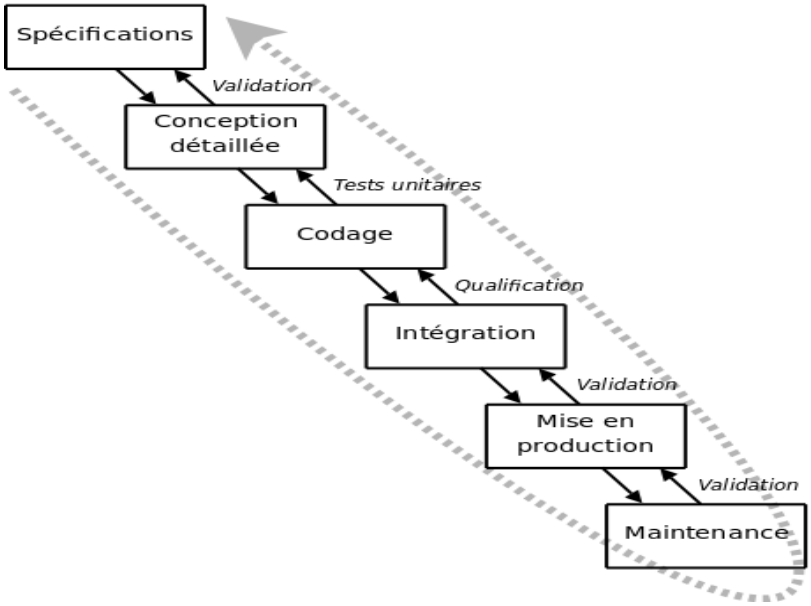


Figure 1-4 : Modèle du cycle de vie en cascade(6)

1.3.2.2 Modèle du cycle de vie en V:

Le modèle en V (cf. figure 6) demeure le cycle de vie le plus connu et certainement le plus utilisé. Il s'agit d'un modèle en cascade dans lequel le développement des tests et du logiciel sont effectués de manière synchrone. Le principe de ce modèle est qu'avec toute décomposition doit être décrite la recomposition et que toute description d'un composant est accompagnée de tests qui permettront de s'assurer qu'il correspond à sa description.(6)

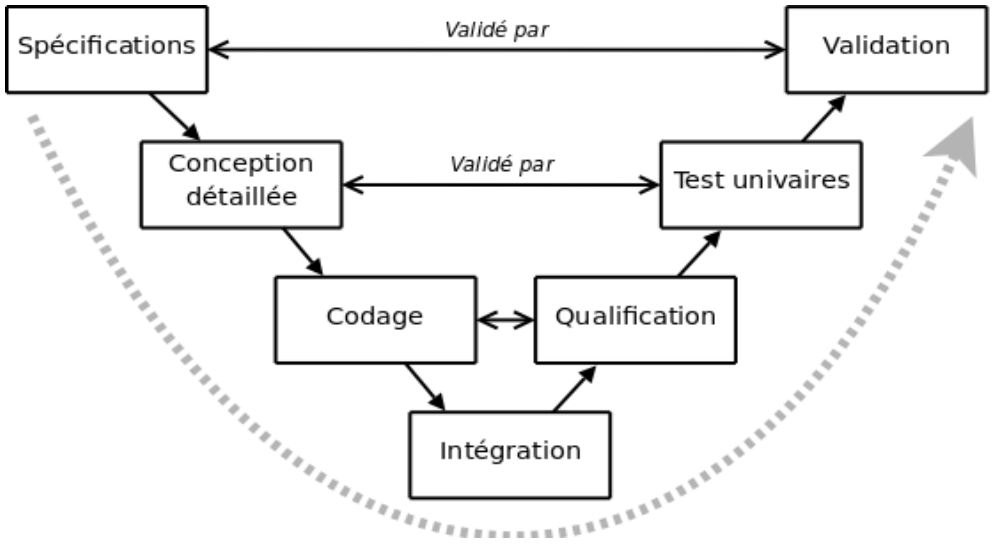


Figure 1-5 : Modèle du cycle de vie en V(6)

1.3.2.3 Modèle du cycle de vie spirale :

Proposé par B. Boehm en 1988, ce modèle est beaucoup plus général que le précédent. Il met l'accent sur l'activité d'analyse des risques : chaque cycle de la spirale se déroule en quatre phases (6) :

1. Détermination, à partir des résultats des cycles précédents, ou de l'analyse préliminaire des besoins, des objectifs du cycle, des alternatives pour les atteindre et des contraintes.
2. Analyse des risques, évaluation des alternatives et, éventuellement maquettage.
3. Développement et vérification de la solution retenue, un modèle « classique » (cascade ou en V) peut être utilisé ici.
4. Revue des résultats et vérification du cycle suivant.

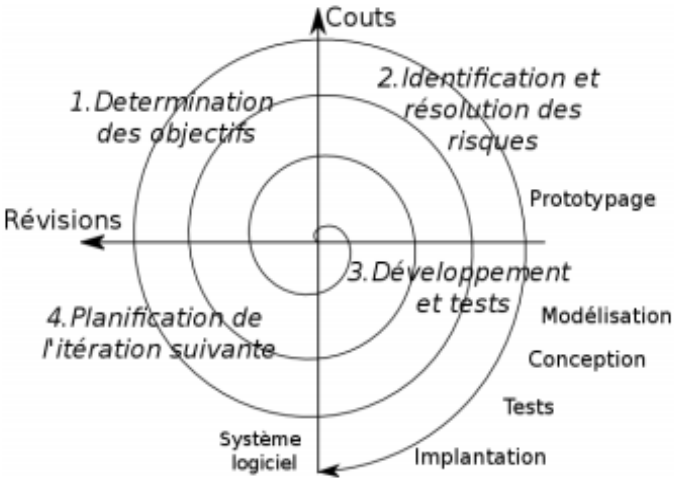


Figure 1-6 : Modèle du cycle de vie en spirale(7)

1.3.2.4 Modèle Prototype :

Le modèle de prototypage est une méthode de développement de système dans laquelle un prototype est créé, testé et ensuite reconstruit si nécessaire jusqu'à ce qu'un résultat approprié soit atteint par lequel développer le système ou produit complet. Sur la base des exigences, la conception est créé et le prototype pour une conception particulière est modélisé et livré aux utilisateurs, puis sur la base du formulaire de retour de l'utilisateur, les modifications appropriées ont été apportées.(8)

Figure 1-7 Modèle de Prototypage(9)

Phases du modèle de prototypage :

Communication: C'est la phase où le développeur et le client organisent la réunion et discutent des objectifs à atteindre pour le logiciel.(8)

Conception: La conception a été réalisée rapidement car les exigences sont éliminées des deux côtés de la fourniture et de la réception. (8)

Il est utilisé pour construire le prototype. Il comprend les aspects importants du logiciel qui sont des entrées et des sorties, mais principalement axés sur les aspects visibles que les activités planifiées.(8)

Modélisation: elle donne une meilleure idée de la nécessité de développer le logiciel et une meilleure compréhension du produit logiciel.(8)

Déploiement: avant le déploiement, le client évalue le logiciel et si le client n'est pas satisfait, il est affiné en fonction des besoins du client. Ce processus se poursuit jusqu'à ce que les exigences du client ponctuel ne soient pas satisfaites. Une fois le client satisfait du produit, leproduit se déploie enfin dans l'environnement de production. Il est soigneusement évalué et testé, et la maintenance est effectuée régulièrement.(8)

Comment fonctionne le modèle prototype?

- Détermination des objectifs.
- Développez le code.
- Communication et raffinement.
- Démontrer.
- Tester.
- Mettre en place.

1.3.2.5 Modèle RAD (Rapid Application Development) :

Le modèle RAD est un développement parallèle de fonctions et une intégration ultérieure, où chaque composant ou fonction est développé en parallèle comme s'il s'agissait de mini projets. Le développement des composants est minuté, livré, puis assemblé en prototype fonctionnel. Cette méthodologie ou ce modèle permet un changement et un développement rapides des composants ou du produit, encourage les retours clients actifs en offrant une visibilité précoce du produit à un client.(10)

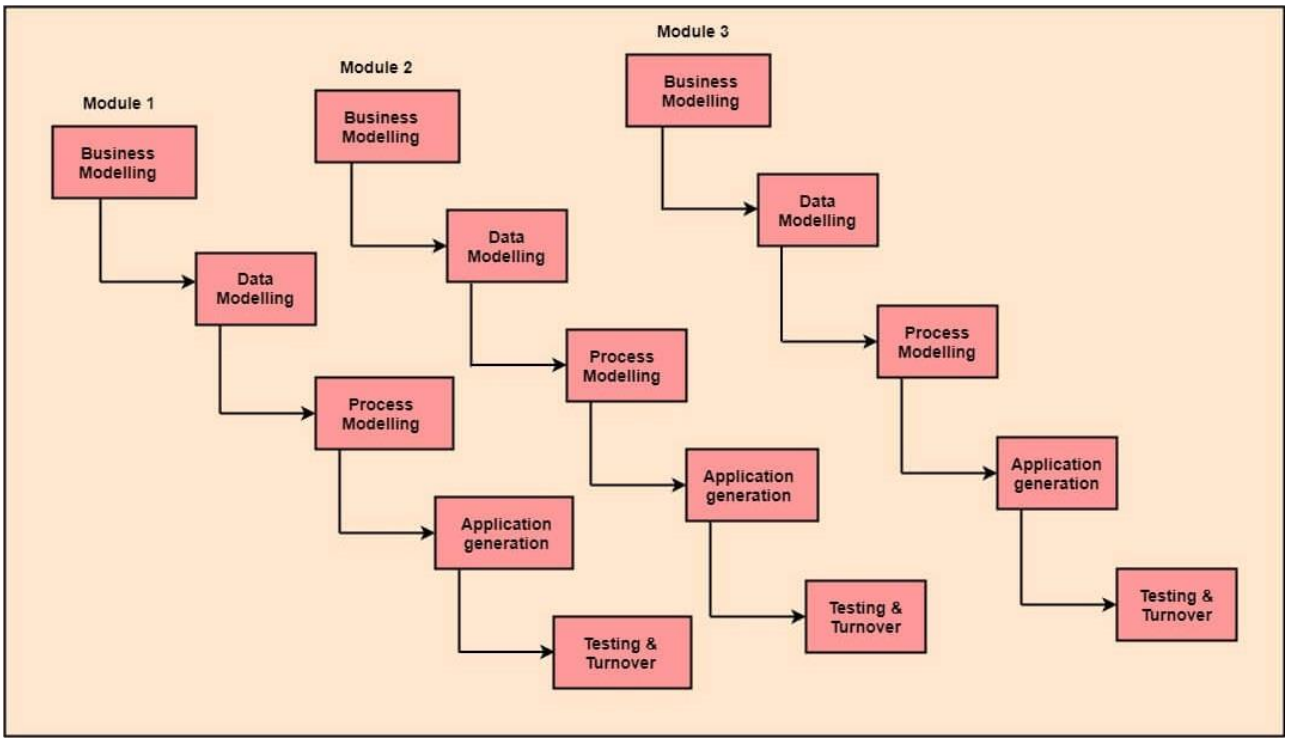


Figure 1-8 Modèle RAD(11)

Phases de modèle RAD :

Planification et analyse des besoins : Cette étape est l'une des étapes les plus cruciales. Ici, les exigences initiales sont rassemblées et analysées correctement.

N'oubliez pas qu'une bonne compréhension des exigences est absolument nécessaire pour que le produit final fabriqué réponde aux attentes.(10)

Conception de l'architecture de projet : Une fois les exigences rassemblées, le prochain objectif est le développement de l'architecture de projet. Une architecture de projet doit être suffisamment flexible pour s'adapter facilement au nouvel ajout de fichiers et de dossiers.(10)

Développement et programmation : Une fois l'architecture conçue, la prochaine tâche majeure est de développer le projet. Cette étape consiste à écrire des piles de code afin d'obtenir l'état irréalisable du produit.(10)

Test : La phase de test consiste à tester le produit développé. Une équipe est impliquée pour tester correctement le produit développé.(10)

Déploiement et maintenance : Une fois les tests terminés, le produit peut être déployé sur le serveur. Un projet déployé nécessite généralement une maintenance et peut-être l'ajout de quelques fonctionnalités supplémentaires.(10)

1.3.3Les méthodes de développement agiles

1.3.3.1 L'agilité

En gestion de projet, la méthode en Cascade c'est l'une des premières méthodes utilisées en raison de sa simplicité et de son caractère séquentiel et linéaire. Cette méthode convient plus aux projets dont les besoins et exigences sont bien définis au début du projet. Les demandes de changements ne sont pas souvent les bienvenues. Les projets de développement logiciel sont souvent caractérisés par des besoins non totalement définis et des demandes de changements qui surviennent à tout moment dans le projet. Cette complexité a conduit progressivement à l'adoption des méthodes agiles dans le domaine de l'ingénierie logicielle. Les méthodes agiles visent fondamentalement la grande satisfaction des parties prenantes du projet (les clients, les utilisateurs, les membres d'équipe, la direction, etc.). Suivant le «manifeste agile», on retient la définition suivante : « **Une méthode agile** est une approche itérative et incrémentale pour le développement de logiciel, réalisée de manière très collaborative par des équipes responsabilisées, appliquant un cérémonial minimal, qui produisent, dans un délai contraint, un logiciel de grande qualité répondant aux besoins changeants des utilisateurs ».(12)Selon le manifeste agile, une méthode agile possède quatre valeurs et 12 principes :

Valeurs

• **Les individus et les échanges** plus que processus et des outils ;
• **Le produit** plus que de la documentation excessive ;
• **La collaboration du client** plus que la négociation ;
• **La réactivité** plus que le suivi d'un plan

Principes

• Satisfaire le client en livrant tôt et régulièrement des logiciels utiles, qui offrent une véritable valeur ajoutée ;
• Accepter les changements, même tard dans le développement ;
• Livrer fréquemment une application qui fonctionne ;
• Collaborer quotidiennement entre clients et développeurs ;
• Bâtir le projet autour de personnes motivées en leur fournissant environnement et support, et en leur faisant confiance ;
• Communiquer par des conversations en face à face ;
• Mesurer la progression avec le logiciel qui fonctionne ;

• Garder un rythme de travail durable ;
• Rechercher l'excellence technique et la qualité de la conception ;
• Laisser l'équipe s'auto organiser ;
• Rechercher la simplicité ;
• À intervalles réguliers, réfléchir aux moyens de devenir plus efficace.

1.3.3.2 Les méthodes agiles en développement logiciel

Cette partie présente les méthodes agiles de développement logiciel les plus usuelles et les plus répandues.

1.3.3.2.1 eXtreme Programming (XP)

XP se définit comme un ensemble de pratiques qui vise à se consacrer à la réalisation elle-même. Ces pratiques sont essentiellement axées la programmation permettant d'améliorer continuellement la conception et code. Voici quelques pratiques d'ingénierie logicielle comme:

• **Le développement piloté par les tests** : les tests unitaires sont développés avant le développement des fonctionnalités ;
• **La programmation en binôme** : XP encourage le développement à 2 afin de faciliter la détection les anomalies et erreurs éventuelles.

1.3.3.2.2 Dynamic Systems Development Method (DSDM)

La méthode DSDM a été créée en 1994 pour répondre aux besoins de qualité et aux principes de RAD (Rapid Development Application). Elle permet donc la livraison rapide et efficace des résultats (13). La méthode AgilePF DSDM est conçue pour permettre l'intégration facile des autres approches agiles comme Scrum et XP. Elle est utilisée pour faciliter l'accessibilité des projets par les parties prenantes. Les phases de développement de la méthode AgilePF sont :

• ***Pre-Project*** **:** pour s'assurer que le projet est aligné avec les stratégies de l'entreprise et répond aux objectifs d'affaires.

• ***Feasability*****:** pour vérifier si le projet est réalisable techniquement et s'il est rentable du point de vue commercial.

• ***Foundations*****:** elle permet de s'assurer de la compréhension de la portée du projet et détermine le cycle de vie du projet ainsi que les processus du modèle à utiliser. Elle peut durer plusieurs semaines pour les gros projets.

• ***EvolutionaryDevelopment*:** utilisation d'une approche de développement incrémentale et itérative.

• ***Deployment*:** phase finale d'assemblage des différents livrables. Livraison et déploiement du produit.

• ***Post-project***: pour s'assurer que les objectifs commerciaux ont été atteints.

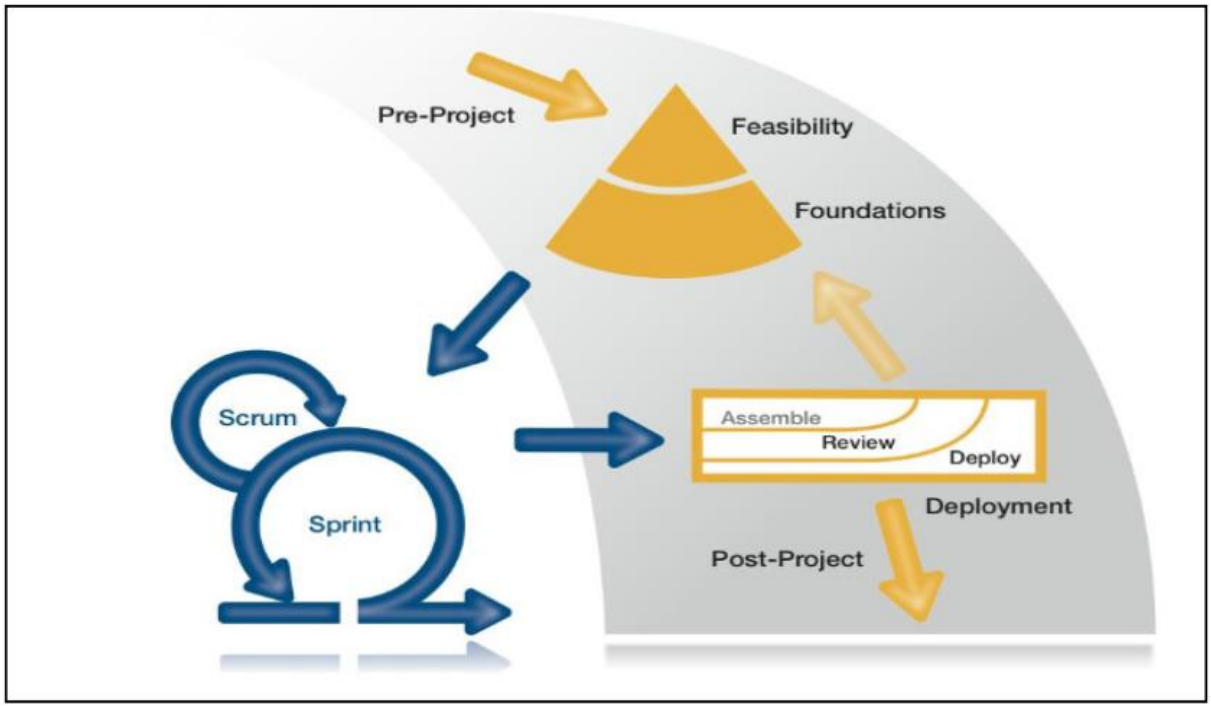


Figure 1-9 Les Phases de développement de Agile(13)

Chaque projet a un cycle de vie qui commence par l'identification d'un besoin potentiel et se termine à un moment où ce besoin est soit terminé a été rejeté (13). Le rejet peut se produire à tout moment si le projet devient non viable d'un point de vue commercial. La phase post-project permet à AgilePF de tenir compte de l'après-projet, ce que ne fait pas habituellement la méthode Scrum. AgilePF tient donc compte de 2 éléments que sont **la gestion du projet** et **la livraison du produit.**

La méthode DSDM est plus formelle avec un nombre assez important de rôles, de processus et d'artéfacts. Elle est aussi caractérisée par la méthode de classification et de priorisation des exigences suivant la méthode **MoSCoW**(13)(14) :

• **M** (Must have) : fonctionnalités obligatoires ;
• **S** (Should have) : fonctionnalités importantes à faires si possibles ;

• **C** (Could have) **:** fonctionnalités possibles, mais non indispensables ;

• **W** (Won’t have) : fonctionnalités qui peuvent attendre la prochaine fois.

1.3.3.2.3 Scrum

La méthode Scrum est la méthode agile la plus populaire. Elle est définie par son fondateur Ken Schwaber comme un cadre (framework) qui présente les éléments qui feront partie du processus appliqué pour la réalisation d'un produit. Scrum impose des itérations de courtes durées appelées ***Sprint*** pour le développement du produit. Le schéma ci-dessous montre le déroulement d'un ***Sprint.***(15)

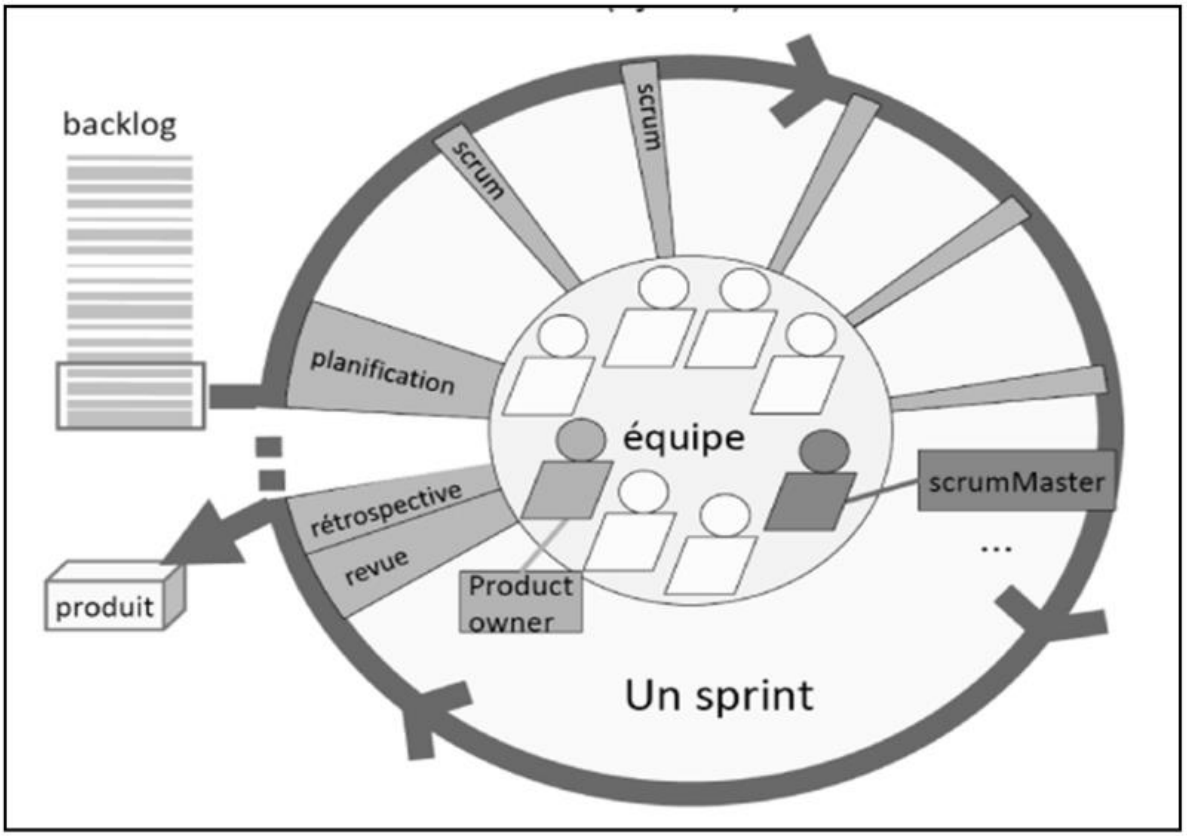


Figure 1-10 Schéma de déroulement d'un Sprint(15)

Les caractéristiques

• Les fonctionnalités à réaliser sont définies et regroupées par ordre de priorité dans le ***Backlog***de produit par le ***Product Owner***;

• À chaque itération ou ***Sprint*** de durée courte et fixe et définie à l'avance, des fonctionnalités sont développées pour créer une version du produit appelée ***Release.*** Le contenu de chaque itération est défini par le ***Product Owner***;

• Au cours des ***Sprints*** des rencontres quotidiennes appelées **mêlées** sont organisées par le ***Scrum Master*** pour l'application des principes de Scrum et suivre l'avancement par rapport aux engagements afin d'assurer le succès du ***Sprint*** ;

• À la fin de chaque ***Sprint,*** un produit partiel est livré. Son évaluation et les rétroactions permettent d'ajuster le ***Backlog***pour le prochain ***Sprint.*** À cette étape, l'équipe de projet identifie les anomalies afin d'améliorer le processus lors des prochains ***Sprints.*** Cette étape est très importante pour le succès du projet. Elle implique l'intervention du ***Product Owner***, du client ou des utilisateurs finaux.

Les composantes

Les composantes nécessaires à l'utilisation de la méthode Srcum sont présentées ci-dessous :

• **Rôles :** Définissent les acteurs clés de la méthode Scrum

• **Cérémonial** : Définissent les blocs de temps nécessaire pour créer la régularité dans les ***Sprints***

• **Artefacts :** Définissent les différents documents et outils. Le plus important est le ***Backlog***. Il faut noter que Scrum impose très peu d'artefacts.

Les phases de développement

En ingénierie logicielle, quatre phases sont habituellement utilisées pour le développement des produits :

• **S**pécification fonctionnelle (définition des exigences fonctionnelles);
• **A**rchitecture (conception);
• **C**odage (et test unitaire);
• **Te**st (d'intégration et de recette)

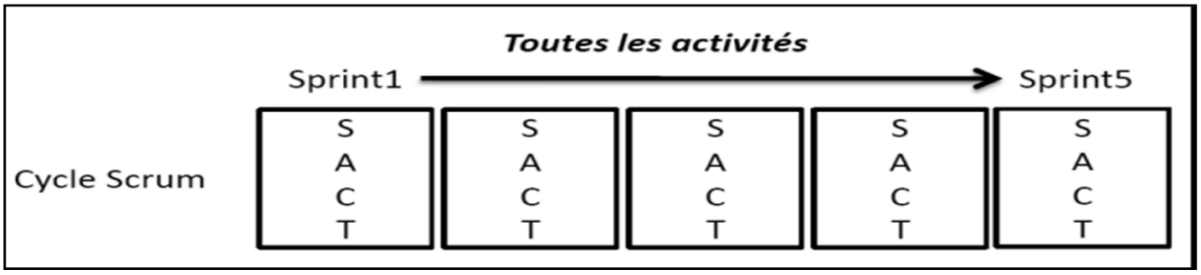


Figure 1-11 Cycle Scrum basé sur 5 Sprints – Itérations(15)

Figure 1-12 Cycle séquentiel (Classique)(15)

Les 2 figures ci-dessus montrent la différence fondamentale entre la méthode classique comme cascade et celle basée sur les principes de la méthode agile. Dans la méthode classique, les différentes activités sont séquentielles. Ce qui implique que les tests sont effectués à la fin du projet. En cas d'erreurs ou de problèmes liés à la conception ou à la définition des exigences, il est difficile de faire des changements. Les demandes de changement ne sont pas souvent acceptées. Ce qui entraîne des délais complémentaires pour la livraison du produit et l'insatisfaction du client dans certains projets. Avec les méthodes agiles comme Scrum, toutes les activités formelles ou évènements sont exécutés durant chaque itération. Ce qui facilite la détection et la correction des erreurs et la prise en compte des demandes de changements.

1.4 UML

UML n'est pas une méthode (*i.e.* une description normative des étapes de la modélisation) : ses auteurs ont en effet estimé qu'il n'était pas opportun de définir une méthode en raison de la diversité des cas particuliers. Ils ont préféré se borner à définir un langage graphique qui permet de représenter et de communiquer les divers aspects d'un système d'information. Aux graphiques sont bien sûr associés des textes qui expliquent leur contenu. UML est donc un métalangage, car il fournit les éléments permettant de construire le modèle qui, lui, sera le langage du projet. (6)

Il est impossible de donner une représentation graphique complète d'un logiciel, ou de tout autre système complexe, de même qu'il est impossible de représenter entièrement une statue (à trois dimensions) par des photographies (à deux dimensions). Mais il est possible de donner sur un tel système des *vues* partielles, analogues chacune à une photographie d'une statue, et dont la conjonction donnera une idée utilisable en pratique sans risque d'erreur grave. (6)

UML 2.0 comporte ainsi treize types de diagrammes représentant autant de *vues* distinctes pour représenter des concepts particuliers du système d'information. Ils se répartissent en deux grands groupes : (6)

1.4.1 Diagrammes structurels ou diagrammes statiques (*UML Structure*)

- diagramme de classes (*Class diagram*)
- diagramme d'objets (*Object diagram*)
- diagramme de composants (*Component diagram*)
- diagramme de déploiement (*Deploymentdiagram*)
- diagramme de paquetages (*Package diagram*)
- diagramme de structures composites (*Composite structure diagram*)

1.4.2 Diagrammes comportementaux ou diagrammes dynamiques (*UML Behavior*)

- diagramme de cas d'utilisation (*Use case diagram*)
- diagramme d'activités (*Activity diagram*)
- diagramme d'états-transitions (*State machine diagram*)
- **Diagrammes d'interaction (*Interaction diagram*)**
  - diagramme de séquence (*Sequencediagram*)
  - diagramme de communication (*Communication diagram*)
  - diagramme global d'interaction (*Interaction overviewdiagram*)
  - diagramme de temps (*Timing diagram*)

Ces diagrammes, d'une utilité variable selon les cas, ne sont pas nécessairement tous produits à l'occasion d'une modélisation. Les plus utiles pour la maîtrise d'ouvrage sont les diagrammes d'activités, de cas d'utilisation, de classes, d'objets, de séquence et d'états-transitions. Les diagrammes de composants, de déploiement et de communication sont surtout utiles pour la maîtrise d'œuvre à qui ils permettent de formaliser les contraintes de la réalisation et la solution technique. (6)

Diagramme de cas d'utilisation : Le diagramme de cas représente la structure des grandes fonctionnalités nécessaires aux utilisateurs du système. C'est le premier diagramme du modèle UML, celui où s'assure la relation entre l'utilisateur et les objets que le système met en œuvre.(6)

Diagramme de Classes : Le diagramme de classes est généralement considéré comme le plus important dans un développement orienté objet. Il représente l'architecture conceptuelle du système : il décrit les classes que le système utilise, ainsi que leurs liens, que ceux-ci représentent un emboîtage conceptuel (héritage) ou une relation organique (agrégation).(6)

Diagramme d'objets : Le diagramme d'objets permet d'éclairer un diagramme de classes en l'illustrant par des exemples. Il est, par exemple, utilisé pour vérifier l'adéquation d'un diagramme de classes à différents cas possibles.(6)

Diagramme d'états-transitions : Le diagramme d'états-transitions représente la façon dont évoluent les objets appartenant à une même classe. La modélisation du cycle de vie est essentielle pour représenter et mettre en forme la dynamique du système.(6)

Diagramme d'activités : Le diagramme d'activités n'est autre que la transcription dans UML de la représentation du processus telle qu'elle a été élaborée lors du travail qui a préparé la modélisation : il montre l'enchaînement des activités qui concourent au processus. (6)

Diagramme de séquence : Le diagramme de séquence représente la succession chronologique des opérations réalisées par un acteur. Il indique les objets que l'acteur va manipuler et les opérations qui font passer d'un objet à l'autre. On peut représenter les mêmes opérations par un diagramme de communication graphe dont les nœuds sont des objets et les arcs (numérotés selon la chronologie) les échanges entre objets. En fait, diagramme de séquence et diagramme de communication sont deux vues différentes, mais logiquement équivalentes (on peut construire l'une à partir de l'autre) d'une même chronologie. Ce sont des diagrammes d'interaction.(6)

Conclusion

Le système d'information est désormais l'une des principales valeurs dans la matière de transmission d'informations et de développement des connaissances. Dans ce chapitre nous avons parlé de système d'information et de son importance dans une entreprise et de génie logiciel, et la modélisation des systèmes d'informations avec UML.

Chapitre 2 : Environnement Mobile et Développement des Applications

2 Introduction

Ces dernières années, le marché du mobile s'est développé très rapidement. Depuis 2015, les utilisateurs naviguent plus sur leurs smartphones ou tablettes que sur les ordinateurs, car nous n'utilisons plus les Smartphones que pour passer des appels, et les géants du mobile et du logiciel informatique le savent déjà. Le plus motivant dans tout cela, c'est que la demande en application mobile ne fait qu'augmenter d'année en année et dans tous les domaines.

Tant que le marché du Smartphone est en croissance continue (et la concurrence entre les firmes ne cesse de devenir de plus en plus féroce) bien sûr le marché des applications mobile le sera également. De telles applications mobiles qu'on retrouve chez tout le monde permettent de jouer, de consulter des emails, de naviguer sur les réseaux sociaux, d'accéder à des services administratifs, d'émettre des requêtes vers une BD, etc. De manière générale, et dans le cadre de notre projet, une application mobile est très utile pour supporter certaine options mobiles (service à distance, publicité, message d'urgence, notification administrative, …) d'un système d'information d'une entreprise donnée. Ceci permettra aux différents acteurs de l'entreprise (agents, clients, fournisseurs, …) d'interagir avec le SI de l'entreprise sans y avoir un poste fixe ou à se déplacer au siège de l'entreprise.

Dans ce chapitre nous allons essayer de répondre à des questions comme : qu'est-ce qu'un Smartphone ? Quelles sont les systèmes d'exploitation mobiles ? Qu'est-ce qu'une application mobile ? Quelles sont les solutions pour développer une application mobile ? Et quelle solution choisir ?

2.1 Smartphone

Un Smartphone, synonyme de téléphone intelligent, également communément appelé mobile, est un téléphone multifonction. Il dispose d'un ensemble de composants : d'un écran tactile (rarement d'un clavier ou d'un stylet), d'un appareil photo numérique, des fonctions d'un assistant numérique personnel et de certaines fonctions d'un ordinateur portable. Sans oublier les composants de base, un Smartphone récent admet un processeur puissant et une

grande mémoire interne ainsi qu'une capacité de stockage très importante qui égale ou dépasse dans certains cas les ordinateurs portables.(16)

Selon le principe d'un ordinateur, il peut exécuter divers logiciels(applications) grâce à un système d'exploitation spécialement conçu pour mobiles, et donc en particulier fournir des fonctionnalités en plus de celles des téléphones mobiles classiques comme : l'agenda, la télévision, le calendrier, la navigation sur le Web, la consultation et l'envoi de courrier électronique, la géolocalisation, le dictaphone/magnétophone, la calculatrice, la boussole, l'accéléromètre, le gyromètre, la messagerie vocale visuelle, la cartographie numérique, etc. Les appareils mobiles les plus sophistiqués bénéficient actuellement de la reconnaissance faciale ou de la reconnaissance des empreintes digitales pour verrouiller/déverrouiller ainsi que de la reconnaissance vocale et la synthèse vocale.(16)

Il est possible de personnaliser son Smartphone en y installant des applications additionnelles telles que des jeux ou des utilitaires via un magasin (store) d'applications en ligne différent pour chaque type de système d'exploitation (exemple, PlayStore pour Android de Google et Apple Store pour IOs Phone de Apple) installé sur le mobile. Il est nécessaire d'avoir une connexion à Internet par l'intermédiaire d'un réseau de téléphonie mobile ou d'un réseau Wi-Fi pour pouvoir utiliser leur potentiel.(16)

2.2 Systèmes d'Exploitation Mobile

Un système d'exploitation est un ensemble de programmes de base qui contrôle le fonctionnement interne d'un ordinateur et dirige son exploitation par les logiciels applicatifs. Cette définition s'applique aussi aux Smartphones que nous retrouvons sur le marché, car ils possèdent les mêmes caractéristiques et structures qu'un ordinateur à savoir un système d'exploitation, un processeur, une mémoire interne RAM et une mémoire de stockage auquel s'ajoute un ensemble de ports de connexion de périphériques qu'il doit gérer.

Figure 2-1 : Logos des différents Systèmes d'exploitation de Smartphones existants

2.2.1Android

Android est le système d' exploitation mobile développé par Google. Il est utilisé par plusieurs Smartphones et tablettes . Les exemples incluent le Sony Xperia, la série Samsung Galaxy, Google Nexus One et les nouveaux du marché comme les Smartphones Xiomi et Oppo pour ne citer que ceux-là.

Le système d'exploitation Androïd (Android OS) est basé sur le noyau Linux. Contrairement à iOS d'Apple, Android est open source, ce qui signifie que les développeurs peuvent modifier et personnaliser le système d'exploitation de chaque téléphone. Par conséquent, différents téléphones basés sur Android ont souvent des interfaces graphiques différentes même s'ils utilisent le même système d'exploitation.(17)

2.2.2IOS

iOS est un système d' exploitation mobile développé par Apple. Il s'appelait à l'origine iPhone OS, mais a été renommé iOS en juin 2009. L'iOS fonctionne actuellement sur iPhone, iPod touch et iPad.(18)

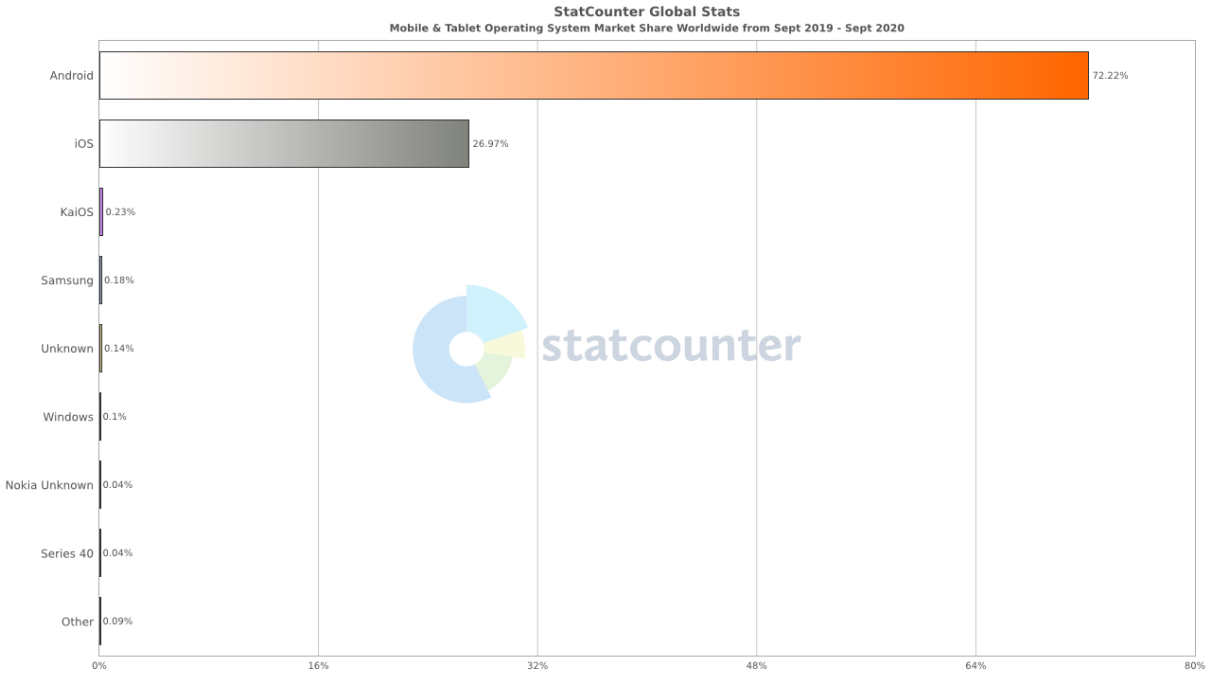


Figure 2-2 : Part de marché des systèmes d'exploitation mobiles et tablettes dans le monde septembre 2019 jusqu'à septembre 2020 (19)

2.3 Application Mobile

Une application mobile est un programme qui s'exécute sur un Smartphone. Elle est développée pour un environnement utilisant comme système un des systèmes discutés au paragraphe précèdent. Les applications mobiles sont adaptées aux différents environnements techniques des smartphones et à leurs contraintes et possibilités ergonomiques (écran tactile notamment). Elles permettent généralement un accès plus confortable et plus efficace à des sites ou services accessibles par ailleurs en versions mobile ou web. L'essentiel du temps passé sur les Smartphones est consacré à des applications mobiles.(20)Une application mobile peut être téléchargeable de façon gratuite ou payante et exécutable à partir du système d'exploitation d'un Smartphone ou d'une tablette.(20)

2.3.1Caractéristiques des applications mobiles

La conception et l'implémentation des applications mobiles sont fortement liées aux systèmes d'exploitation de l'appareil mobile sur lequel ces applications vont être installées et exécutées. Cependant, il reste toujours quelques caractéristiques communes aux applications mobiles malgré la diversité des systèmes d'exploitation mobiles et la disparité des plates-formes de systèmes d'exploitation qui les gèrent.

- ***Temps de lancement*** **:** l'application doit se lancer en moins de 5-6 seconds.(21)

- ***Haute performance*** **:**garantir constamment une haute performance et une rapide réactivité afin de répondre aux exigences de l'utilisateur qui cherche la fiabilité dans l'application utilisée.(21)

- ***Pas d'étapes intermédiaires*** **:** afin d'atteindre le but d'exécution de l'application.(21)

- ***Consommation d'énergie*** **:** l'optimisation du processus de l'application pour garantir une exploitation efficace de l'énergie.(21)

2.3.2Types d'applications mobiles

Dans le domaine du Smartphone, plusieurs types d'applications mobiles existent. Selon l'usage auquel elle est destinée, une application mobile peut être développée pour réaliser une fonction précise ou à fournir une fonctionnalité réduite (ou étendant ou dupliquant) d'un site web ou encore avoir un mélange de fonctionnalités des précédentes. La création d'une application mobile mène à beaucoup de questions concernant le développement, la

fonctionnalité et l'ergonomie. Ainsi, avant de se lancer dans le développement d'une application mobile, il est indispensable de bien choisir le type d'application à développer et de bien argumenter ce choix.(21)

2.3.2.1 Applications natives

Les applications natives sont des programmes développés pour exécuter certaines fonctionnalités et être déployables dans des plateformes ou appareils mobiles particuliers. Ces derniers permettent aux applications de prendre d'avantage de leurs caractéristiques, leurs applications préinstallées par défaut (ex., calendrier et contact), et aussi les technologies qu'elles fournissent comme la Caméra, le GPS (localisation) ou encore l'espace de stockage et les versions ou les types des périphériques.

Les applications natives peuvent être téléchargées depuis des boutiques en ligne publiques ou privées, ensuite être installées sur l'appareil mobile ou bien elles peuvent exister par défaut avec l'appareil mobile (ex. Contact, Messagerie et Calendrier, etc.).(21)

2.3.2.1.1 Avantages

- Une meilleure performance parce qu'elles utilisent les logiciels et le matériel valable dans l'appareil mobile.
- Comme ces applications sont déjà installées dans l'appareil et utilisent les données déjà existantes, elles peuvent être exécutées hors ligne sans besoin d'Internet.
- Chaque application est distinguée par son logo pour attirer l'attention de l'utilisateur.(21)

2.3.2.1.2 Inconvénients

- L'inconvénient majeur de ce type d'application, c'est qu'elles sont fortement liées avec l'appareil sur lequel elles sont installées, ce qui engendre l'impossibilité d'évoluer ou d'exploiter de nouvelles technologies.
- Rendre les applications natives déployables dans différents types d'appareils/plateformes (c.-à-d. réécrire le code dans différents langages) est une tâche consommatrice en termes de temps.
- L'exclusivité des applications natives pour un type d'appareil mobile bien particulier diminuera le nombre des utilisateurs et le gain de leurs ventes.(21)

2.3.2.2 Applications Mobiles Web

Les applications mobiles web sont une version réduite des applications web dédiées pour les appareils mobiles. L'accès à ce genre d'application est actionné par un navigateur web.Par conséquent, ces applications sont indépendantes des plateformes et caractéristiques des appareils mobiles. Nécessairement, ces applications exigent une connexion internet pour leurs utilisations.

Lors du développement des applications mobiles web, il est essentiel de:

- Définir le texte et les images statiques de l'application en utilisant par exemple HTML5.
- Définir le style et la présentation de l'application en utilisant par exemple CSS.
- Définir l'interaction et l'animation au moyen d'un outil comme JavaScript.

Une meilleure utilisation de ce genre d'application est motivée par la possibilité d'une réorganisation de leurs contenus en fonction des caractéristiques de l'appareil mobile.(21)

2.3.2.2.1 Avantages

- L'avantage le plus important des applications mobiles web est la capacité de déploiement sur les multiples plateformes indépendamment du type d'appareil mobile.
- Moins cher, facile et rapide à développer.
- Accès facile en utilisant un simple URL sans le besoin d'installation ou le téléchargement des compléments.(21)

2.3.2.2.2 Inconvénients

- Les navigateurs traditionnels sont plus performants que les navigateurs des appareils mobiles.
- Les applications mobiles web ne peuvent pas exploiter les fonctionnalités offertes par les logiciels et les hardwares des appareils mobiles.
- La performance des applications mobiles web dépend de la vitesse et l'état de la connexion via Internet ou les réseaux de téléphonie mobile GSM.(21)

2.3.2.3 Applications mobiles multi-plate-formes (hybrides)

Lors du développement des applications mobiles natives, les applications Android sont écrites en Java et celles d'iOS en Swift et Objective-C. Cette approche fournit des applications sans faille et très performantes. Mais, le chemin de développement que cette approche suit est

assez long et coûteux car le même code doit être écrit deux fois. Ainsi, la solution pour faire les choses de manière transparente (sans investir beaucoup de ressources) est de préconiser un développement d'applications mobiles multiplateformes (hybrides).

Les applications mobiles multiplateformes (hybrides) sont la combinaison entre les applications natives et les applications mobiles web. La synergie entre ces deux types d'application résulte en une réduction de temps, d'effort de développement, de prix et de maintenance. Ces applications sont téléchargeables via des boutiques en ligne, installées sur l'appareil et exécutées depuis une simple icône comme les applications natives. Les applications mobiles hybrides sont créées pour être exécutées sur plusieurs plateformes.(21)

Conclusion

Dans ce chapitre, nous avons présenté les différents types de systèmes d'exploitation pour Smartphones et appareils mobiles, les différents types d'application mobile ainsi que leurs avantages et inconvénients.

Pour répondre à notre problématique, nous avons choisi d'utiliser la solution multi-plate-formes, qui nous offrent juste ce qu'il nous faut pour développer une application à la fois pas très lourde et assez riche en fonctionnalités native et multiplateformes. Dans le chapitre suivant, nous allons aborder les meilleurs Frameworks multiplateformes (Hybrides) dans le but d'éclairer notre choix de manière convenable et le justifier.

Chapitre 3 : Framework Mobile Multiplateforme ou Hybride

3 Introduction

Parmi les trois types de solutions pour développer une application mobile présentées dans le chapitre précédent, il existe le développement hybride et le développement natif. Le long de ce chapitre, nous ferons une petite comparaison entre ces deux solutions logicielles.

Le développement natif est un développement spécifique à chaque type d'OS de mobile (iOS, Android, Windows,…). Pour pouvoir développer une application disponible sur iOS et Android, ça va être le prix de deux applications sans oublie le temps que ça prendra.

Contrairement au développement natif, le développement hybride permet un développement unique pour servir différents OS. Avant 2015, ce type de développement mobile était assez décevant du point de vue de la performance et de la marge de manœuvre autorisée au niveau du design. Depuis 2015, de nouveaux Frameworks permettent des résultats impressionnants, répondant largement aux attentes même des plus exigeants. Même si un seul code est développé, il fait appel aux fonctions natives des Devices, ce qui donne une expérience utilisateur très proche de celle obtenue avec un développement natif, mais avec un modèle économique beaucoup plus attractif.

3.1 Framework Mobile Multiplateforme ou Hybride

À la différence des implémentations natives, les Framework multiplateformes génèrent des applications compatibles avec plusieurs plateformes. La quasi-totalité de ces outils prennent en compte à la fois le développement pour Android et iOS.(22)

Les principaux outils multiplateformes en ce moment sont Ionic, Flutter, React Native et Xamarin. Sans trop tarder, nous donnons des détails sur ces plateformes hybrides.

Figure 3-1 : Logos des différents Framework disponibles et les plus utilisés dans le développement des Applications mobiles.

3.1.1Ionic :

Le Framework multiplateforme *Ionic*a été élaboré avec Apache Cordova et Angular. Avec cet outil, les applications mobiles développées sont utilisables sur plus d'une plateforme mobile. Elles fonctionnent sur Androïd et iOS. La conception de ces applications est classique. Elle se fait à l'aide de HTML, d'Angular ou de CSS.

En tant qu'outil multiplateforme, *Ionic* permet aux entreprises de développer leur application mobile avec un minimum de dépenses. L'entreprise n'a pas besoin de faire appel à plusieurs ingénieurs logiciels pour la conception d'une application mobile.

En plus de faire économiser de l'argent, le frameworkIonic permet de gagner en temps. Demandant moins des ressources et moins de jours et développeurs, l'entreprise pourra mettre son Appli plus tôt à la disposition du public et être plus réactif dans le lancement de nouvelles fonctionnalités.

Le véritable inconvénient d'Ionic est sa performance comparée à une autre native. Si l'entreprise désire concevoir une application exigeant des traitements graphiques pointus, *Ionic* n'est certainement pas le choix idéal.Le souci majeur d'*Ionic* est sa performance. En effet comme cité précédemment, *Ionic* se sert d'une WebView afin d'afficher son contenu (qui est développée grâce à des composants Web) et cela limite fortement les performances.(22)

3.1.2Flutter

Google a développé ce Framework pour offrir à ses utilisateurs une belle et immersive interface. *Flutter* est entièrement compilé en code ARM (Advanced RISC Machines) natif, par opposition aux applications mobiles qui emploient des vues Web. Une application construite avec *Flutter* offre une sensation native. Skia est un moteur graphique Open Source populaire en 2D, propre à *Flutter*. Le développement de ce Framework se fait avec le langage

de programmation Dart. Une application construite avec cet outil se caractérise par sa rapidité, environ 60*ips*(instructions par seconde). D'autres développeurs atteignent même les 120*ips*.(22)

Cependant, pour développer avec un tel outil, les développeurs devront s'approprier un nouveau langage, inconnu auparavant. Cependant, pour développer un jeu ou une application qui requiert plusieurs fonctions propres à l'appareil, *Flutter* n'est pas à priori le plus adapté.

3.1.3 React Native

Framework créé par Facebook, *React Native* utilise comme langage de programmation JavaScript (et est couplé à l'implémentation mobile du Framework ReactJS). Il se distingue des applications hybrides par le fait qu'il emploie les composants natifs du téléphone.

Le JavaScript est langage souple et rapidement assimilable. Par sa syntaxe, il permet une grande diversité dans l'écriture. L'orientation composante du Framework permet également de réutiliser différentes parties du code sans avoir à les réécrire.

React présente les mêmes avantages qu'*Ionic* et *Flutter* avec la performance en plus. L'entreprise ne développera qu'une seule application pour Android et iOS. L'équipe de développement de l'entreprise pourra mener à succès un projet alliant gain de temps, gain de budget et performance. Autre facteur qui favorise le développement, l'existence de nombreux outils préconçus et utilisables directement pour le développement, pas nécessaire de les créer. Aussi, *React Native* est le Framework de développement mobile possédant la plus grande communauté. Ce point n'est vraiment pas à négliger car une communauté bien fournie et armée permettra de résoudre plus facilement les éventuellement blocages techniques durant le développement des Applis.(22)

3.1.4Xamarin

Xamarin est une plateforme open source pour la création d'applications modernes et performantes pour iOS, Android et Windows avec .NET. Xamarin est une couche d'abstraction qui gère la communication du code partagé avec le code de plateforme sous-jacent. Xamarin s'exécute dans un environnement géré qui offre des fonctionnalités telles que l'allocation de mémoire et le garbage collection.

Xamarin permet aux développeurs de partager en moyenne 90% de leur application sur différentes plates-formes. Ce modèle permet aux développeurs d'écrire toute leur logique métier dans un seul langage (ou de réutiliser le code d'application existant) tout en obtenant des performances, une apparence et une convivialité natives sur chaque plate-forme.

Les applications Xamarin peuvent être écrites sur PC ou Mac et compilées dans des packages d'application natifs, tels qu'un fichier **.apk** sur Android ou un fichier **.ipa** sur iOS.(23)

3.2 Choix du Framework

Pour décider lequel est le meilleur pour le développement d'applications mobiles multi-plate-forme, nous définissons d'abord les attributs sur la base desquels nous pouvons comparer les différents Frameworks vus précédemment. Ces attributs se résument en :

- La pile des langues
- Performance
- Interface graphique utilisateur
- Marché et communauté
- Plateformes prises en charge
- Partage de Code
- Applications populaires
- Tarification

3.2.1Pile des Langues

- Voyons quel Framework utilise quels langages de programmation et offre quels avantages:

  - ***React Native*:** il utilise JavaScript qui est actuellement l'un des langages de programmation les plus populaires, dynamiques et de haut niveau. Il combine les avantages de JavaScript et React.JS et est parrainé par Facebook.

Le côté solide de *React Native* qui le rend meilleur parmi les trois autres Frameworks en termes de langage programmation est qu'il permet d'écrire quelques composants en Swift, Objective-C ou Java lorsque les développeurs en ont besoin. À l'aide de modules et de

bibliothèques natives dans les applications *React Native*, il est possible de gérer des opérations lourdes en termes de calcul, telles que la gestion vidéo ou l'édition d'images.(24)

- ***Xamarin* :** il utilise C# avec l'environnement *.net* pour développer des applications Android, iOS et Mac. Tout ce qui peut être accompli avec les langues natives, un développeur peut le faire en C# en utilisant *Xamarin*. Cependant, les développeurs ne peuvent pas utiliser les bibliothèques Open Source natives accessibles pour iOS et Androïd avec Xamarin. Il existe une variété de bibliothèques *.net* accessibles qui répondent au besoin convoité.(24)

- ***Ionic*** : il utilise HTML5, CSS et JS pour développer et exécuter des applications, et nécessite l'encapsuleur Cordova pour accéder aux contrôleurs de plateforme natives. En utilisant *Ionic*, il est possible également d'utiliser Type Script qui améliore la qualité du code.(24)

- ***Flutter :*** il utilise Dart pour développer des applications de haute qualité pour Android, iOS et le Web. Dart est un langage de programmation qui offre de nombreux avantages et est basé sur C/C++ et java. Bien qu'il soit nouveau, le langage devrait bientôt prendre d'assaut l'industrie. Dart est l'une des raisons pour lesquelles le développement d'applications *Flutter* est préféré par un certain nombre de développeurs d'applications ces jours-ci.(24)

La figure suivante classe en fonction des avantages offerts par les langages de programmation:

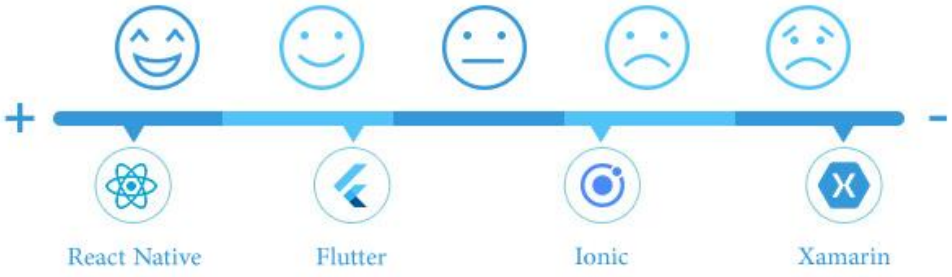


Figure 3-2 : Le classement en fonction des avantages offerts par leurs langages de programmation. (24)

3.2.2Performance

- ***React Native:*** Les performances qu'il fournit sont très similaires aux applications natives car elles rendent les éléments de code spécifiquement aux API natives. *React* permet également aux développeurs d'utiliser des modules natifs écrits dans des langages natifs pour écrire du code pour des opérations compliquées. Cependant, ils ne peuvent pas être réutilisés sur deux plates-formes; leur objectif principal est de fournir des performances plus élevées.(24)

- ***Xamarin:*** les performances de *Xamarin* sont également considérées comme proches au native. *Xamarin* a deux façons de créer des applications mobiles: Xamarin. Android/Xamarin. iOS et Xamarin. Forms.

Les applications Xamarin.Android/iOS fonctionnent comme natives car leurs capacités multiplateformes se concentrent essentiellement sur le partage de la logique métier plutôt que sur la base de code. Il aide à atteindre les performances natives qui ne sont pas possibles avec des solutions qui interprètent le code lors de l'exécution.

Cependant, l'approche Xamarin. Forms est centrée sur un large partage de code avec un comportement moins spécifique à la plate-forme. Cela diminue complètement les performances du code dans de nombreuses opérations par rapport à différentes plates-formes.(24)

- ***Ionic:*** Ses performances ne sont pas aussi similaires à celles des offres natives que *Xamarin*, *React Native* ou *Flutter*, car il utilise les technologies Web pour rendre une application. Cette approche diminue considérablement la vitesse. De plus, le développement d'applications ioniques n'utilise pas de composants natifs et essaie de créer une apparence native en utilisant les technologies Web. L'avantage d'*Ionic* est son processus de test rapide qui s'exécute instantanément dans un navigateur qui rationalise le processus de développement.(24)

- ***Flutter***: Comparer sur la base des performances de l'application, *Flutter*estle moins performants par rapport à ses concurrents. Parce qu'il a des avantages de Dart et qu'il n'y a pas de pont JavaScript pour démarrer les interactions avec les composants natifs de l'appareil, la vitesse qu'il offre est incroyable (24)

La figure 3.3 montre le classement selon la performance qu'offre chaque Framework :

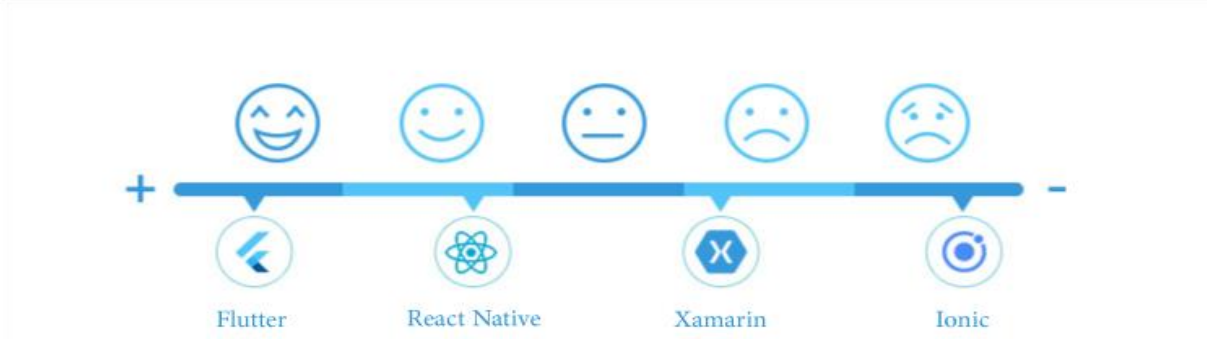


Figure 3-3 : Le classement des Frameworks selon les performances qu'ils offrent. (24)

3.2.3Interface Graphique

Les utilisateurs jugent les applications dans les premières secondes d'utilisation et c'est pourquoi l'interface graphique d'une application doit être engageante tout en étant facile.

- ***React Native:*** les modules *React Native* s'associent aux contrôleurs d'interface utilisateur natifs, ce qui offre une expérience utilisateur singulière très proche des applications natives. Il utilise en outre la bibliothèque *ReactJS* avec des éléments d'interface utilisateur étendus et qui rationalise le développement de l'interface.(24)

- ***Xamarin:*** Il permet de créer des interfaces utilisateurs de deux manières différentes: en utilisant Xamarin. Android/iOS ou Xamarin. Forms. Le premier prend beaucoup de temps, mais garantit un aspect et une convivialité natifs en termes d'UX(User Experience).

Avec Xamarin. Forms, on peut accélérer considérablement le processus de développement et économiser beaucoup de ressources, mais au détriment de l'aspect local. Cela peut être une bonne solution pour les entreprises.(24)

- ***Ionic:*** Ionic interface utilisateur n'utilise pas du tout d'éléments natifs et rend tout en HTML et CSS . Ensuite, il applique Cordova pour offrir une expérience mobile native. Les composants angulaires qui accompagnent le cadre permettent également aux applications ioniques de ressembler à celles natives.

- ***Flutter:*** il fournit les meilleures interfaces utilisateur. Ionic et Xamarin offrent des applications multiplateformes, mais leur efficacité et leurs performances ne peuvent pas battre

Flutter et React Native. Ils sont bloqués et manquent de réactivité si l'application est lourde et que des composants d'interface utilisateur plus natifs sont utilisés.(24)

La figure 3.4 montre le classement des interfaces utilisateurs selon les plateformes.

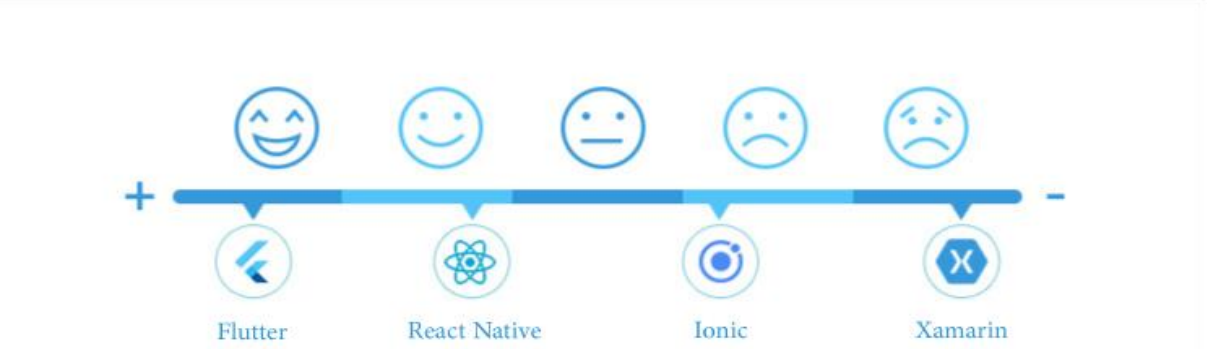


Figure 3-4 : Le classement des Frameworks selon l'interface utilisateur qu'ils proposent (24)

3.2.4Marché et Communauté

- ***React Native:*** Il utilise React une bibliothèque populaire et un langage de développement Web essentiellement en JavaScript et fournit de véritables applications natives. Ces qualités en font une plateforme solide et sont les raisons de sa renommée.(24)

- ***Ionic:*** C'est le deuxième Framework le plus populaire après *React*. Il permet aux développeurs de créer des applications mobiles natives de la manière la plus rapide possible.(24)

- ***Xamarin:*** Xamarin est également un Framework populaire, et, Microsoft continue de déployer beaucoup d'efforts pour développer la communauté *Xamarin*. Les développeurs qui travaillent à l'intérieur de l'écosystème Microsoft peuvent sans trop s'étendre commencer à travailler avec l'innovation en raison de son support actif.(24)

- ***Flutter:*** Flutter est un nouveau Framework pour la communauté en ce moment et pas très populaire. Mais, il est intensément annoncé par Google qui montre qu'ils veulent en faire une chose majeure dans le monde mobile. Bien qu'il ait encore quelques points de picotement, son utilisation est amusante et permet de passer rapidement de l'idée au prototype en passant par l'application.(24)

La figure 3.5 montre le classement des Frameworks selon leurs impacts sur le marché et leurs communautés d'utilisateurs et de développeurs, ainsi que leurs fiabilités dans l'industrie.

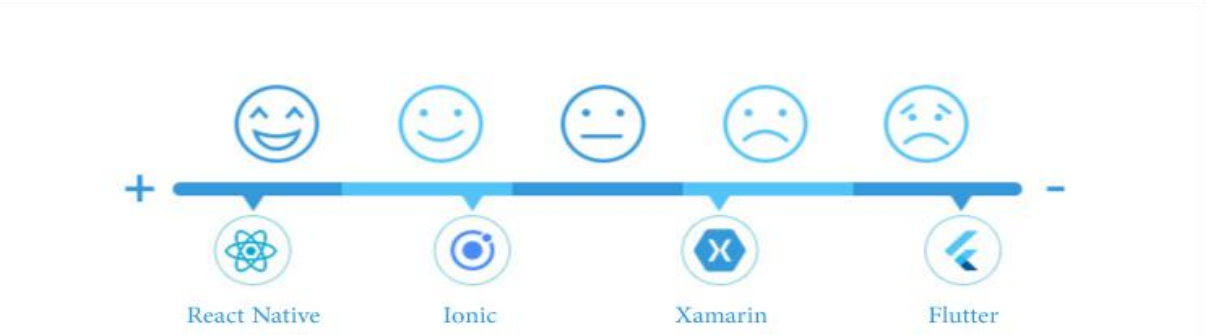


Figure 3-5 : Le classement des Frameworks basé sur leur reconnaissance et leur fiabilité dans l'industrie. (24)

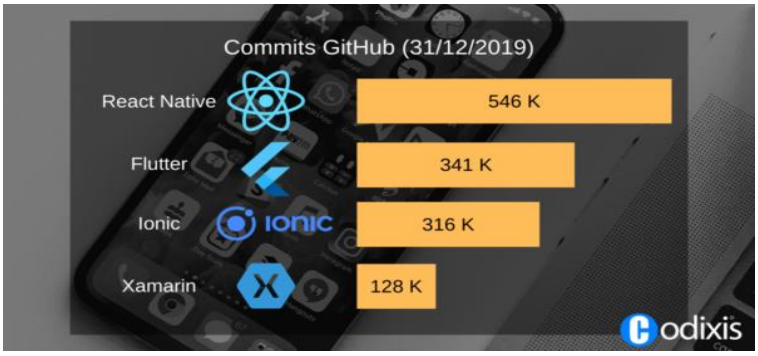


Figure 3-6 : Figure du Commit GitHub en date du 31/12/2019 (25)

3.2.5Plateformes prises en charge

- *React Native:* ***Android 4.1+, iOS 8+***
- *Xamarin:* ***Android 4.0.3+, iOS 8+, Windows 10***
- *Ionic:* ***Android 4.4+, iOS 8+, Windows 10***

3.2.6Réutilisation du code :

- ***React Native:*** le Framework utilise des composants natifs écrits en Objective-C, Swift ou Java pour améliorer les performances de l'application. Mais, ces composants natifs ne

peuvent pas être réutilisés sur d'autres plates-formes. Les développeurs doivent donc faire un peu de travail pour changer cette base de code particulière. Cependant, acceptez ces composants natifs tout ce qui reste de la base de code (environ 90%) peut être réutilisé.(24)

- ***Xamarin:*** Il ne nécessite pas de basculer entre les environnements de développement: toutes les applications Xamarin sont développées dans Visual Studio. Habituellement, jusqu'à 96% du code source peut être réutilisé avec Xamarin. Forms, ce qui accélère le processus de développement.(24)

- ***Ionic:*** Une partie essentielle des applications développées dans Ionic est leur universalité. Quel que soit le système d'exploitation fourni, ils fonctionneront également bien sur chacun d'entre eux. Mais, certains composants de l'interface utilisateur doivent être modifiés selon les règles dictées par une plate-forme particulière, ce qui nécessitera des efforts supplémentaires.(24)

- **Flutter :** Dans React Native, nous avons des fonctions prêtes à l'emploi, ce qui accélère la vitesse de développement. Cependant, dans Flutter, nous devons ajouter des fichiers dédiés pour la plate- forme Android et iOS en fonction de leurs règles.(24)

La figure 3.7 montre le classement des Frameworksbasé sur leur réutilisation de code.

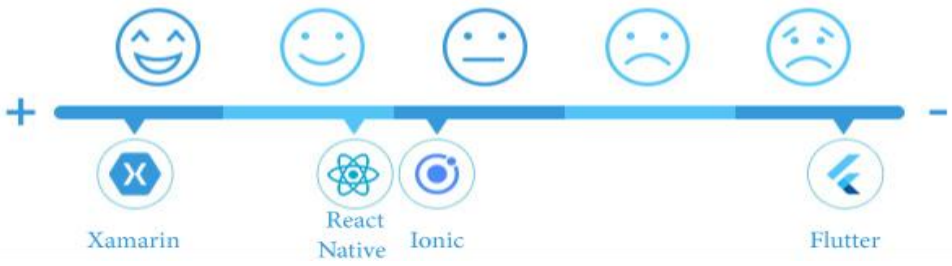


Figure 3-7 Le classement des Frameworks basé sur leur réutilisation de code. (24)

3.2.7 Applications populaires

- **React Native:** Facebook, Instagram, Airbnb, UberEats et bien d'autres.(24)

- **Xamarin:** Olo, la Banque mondiale, Storyo.(24)

- **Ionic:** JustWatch, Pacifica, Nation wide et bien d'autres.(24)

- **Flutter:** Hamilton. (24)

3.2.8Prix

- ***React Native et Flutter:*** Les deux sont des frameworks complètement open source. Les ingénieurs peuvent utiliser gratuitement ce système et leurs bibliothèques.(24)

- ***Xamarin:*** Xamarin oblige les développeurs à installer IDE, qui est fourni par abonnement. Cependant, il propose également une édition gratuite pour les étudiants et les entreprises non commerciales comptant jusqu'à 5 utilisateurs.(24)

- ***Ionic:*** C'est un framework open source gratuit pour développer des applications mobiles multiplateformes. Mais, l'organisation propose sa version Pro qui est payante. La société garantit qu'Ionic Pro accélère le processus de développement.(24)

Conclusion

Nous avons vue dans ce chapitre les Frameworks multi-plateformes les plus populaires et les plus utiliser, avec une comparaison entre eux. Ensuite, notre choix a porté sur *le* Framework *React native* pour le développement de notre application.

Pourquoi *React Native* ? Comme on l'a vu dans ce chapitre React Native est aujourd'hui le Framework Open Source le plus utilisé et que c' est un bon compromis entre performance, légèretés, en plus il est basé sur le JavaScript ce qui nous facilite la tâche de développement. React Native est, en 2020, le Framework sur lequel nous notons la plus grande activité de la part de la communauté de développeurs, et vusa vitesse de développement pour créer une application mobile avecces caractéristiques alors nous avons jugé que ce Framework est le plus adéquat comme choix parmi les autres Framework de développement multiplate forme pour réaliser notre travail.

Après le choix technique de type et de plateforme de développement de notre solution mobile, pour pouvoir intégrer cette solution dans notre système choisi, il ne faut assurer la

communication entre notre application et le système, cela se fait par la création d’ un service web qui va nous assurer cette communication.

Chapitre 4 : Les services web

4 Introduction

Tout a commencé avec l'explosion de web, que les chercheur en commencé de développée des logiciels pour faciliter la communication entre les machine et les applications connecté via internet ces logiciel est nommé « web service » en français les services web. En quelques années, les services web sont devenus le nouveau point de convergence technologique de l'industrie du logiciel.

Dans ce chapitre, nous verrons les principales caractéristiques des web services, ce qui les composent et ces aspects technologiques, ces types comme SOAP et REST avec leur avantage et inconvénient, une définition RESTFUL API et une comparaison SOAP vs REST, en termine avec une conclusion.

4.1 Définition

La Définition du W3C (World Wide Web Consortium) définie les web services comme suit :
« Un service web est un système logiciel identifié par un URI, dont les interfaces publiques et les « bindings » sont définies et décrites en XML. Sa définition peut être découverte [dynamiquement] par d'autres systèmes logiciels. Ces autres systèmes peuvent ensuite interagir avec le service web d'une façon décrite par sa définition, en utilisant des messages XML transportés par des protocoles Internet. »

Un service web est donc une application client-serveur faiblement couplée, identifiée par un URI (Uniform ressource identifier) faisant communiquer des programmes, bases de données, objets, processus d'affaire, et traitent les données en laissant accéder parfois à une partie du système. Exemple d'URI : une URL (pour un site web), un URN (l'identifiant ISBN d'un livre).

Ces applications, indépendantes de tout langage, de l'implémentation, de l'OS de la plate-forme, de l'architecture sous-jacente (.NET, JEE, …), sont destinées à être utilisées par d'autres applications et moins par des humains. Elles utilisent les

protocoles basiques d'internet, SMTP, HTTP. Bien qu'échangeant en XML, un web service peut être implémenté dans différents langages (java, c++, VB etc.).
Contrairement aux EDI, (L'Échange de Données Informatisées) fortement couplés, les web services sont faiblement couplés, donc on peut modifier L'API et l'application qui l'utilise sans que cela dégrade l'interaction.(26)

4.2 Caractéristiques des web services

Le web service doit pouvoir interagir avec le client et le serveur, et pour être réutilisable, il a donc certaines caractéristiques :

- Un contrat entre les deux parties.
- Le fournisseur du service doit utiliser ou mettre en place des standards et normes le définissant.
- Le web service doit proposer une interface d'échanges au client (interface web par exemple).
- Il doit pouvoir être décrit et être découvert.
- Le web service doit être sécurisé.
- Le web service doit être fiable au niveau de son implémentation.(26)

4.3 Avantages

Les avantages du Web services sont nombreux, comme le démontre la liste ci-dessous :

- Une intégration facilitée d'un système d'informations avec une plateforme marchande
- Ses composants sont réutilisables
- Une Interopérabilité permet de lier les différents systèmes entres eux
- Réduction de couplage entre les systèmes.
- Un périmètre fonctionnel étendu mis à la disposition des marchands : Import, Inventaire, Gestion de la commande, Pricing, Après-Vente…
- Met en relation des systèmes hétérogènes
- Interconnecte des middlewares/ou permet de les installer
- Compatible avec tous langages
- Il permet de faire communiquer serveurs et machines,

- Puissance de calcul nécessaire réduite
- Multi-utilisateur, sans perturber les sources
- Mise à jour des composants facile
- Maintenance réduite (comme tout outil de big data)
- Il n'est lié à aucun système d'exploitation ou langage de programmation

Tous ces avantages font qu'aujourd'hui les Web Services sont encore utilisés aux seins des entreprises, et des sites web.(26)

4.4 Architecture des web services

Les web services peuvent être définis sous différentes formes, en fonction de la demande.

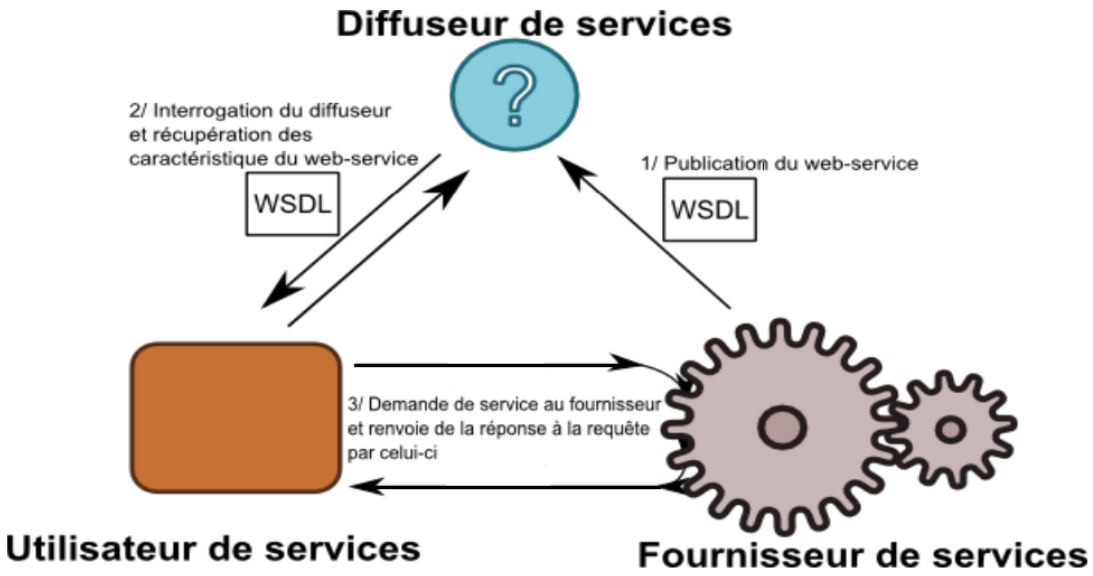


Figure 4-1 : Principe des échanges web services(27)

4.4.1Les applications format client-serveur

Un utilisateur envoie d'une machine client (ordinateur, ou autre) des requêtes à un serveur qui lui répond, ce serveur de niveau 2 (serveur d'applications) puise souvent dans un serveur de niveau 3 (serveur de bases de données). On peut résumer comme suit :

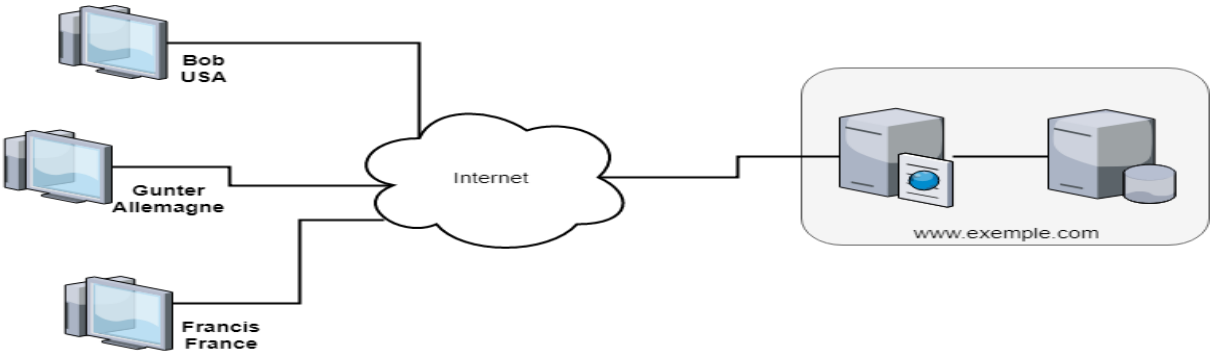


Figure 4-2 : Liaison entre clients et serveur web. (19)

4.4.2Les applications format faiblement couplées

On appelle faiblement couplées des applications peu dépendantes d'autres applications, et qui échangent peu d'information entres elles. Il existe 7 niveaux de couplage, allant du plus faible au plus fort :

1. **Sans couplage :** Aucun échange n'est effectué entre les composants
2. **Par données :** Les données échangées se font uniquement par commande avec des paramètres 'simple', comme des nombres, des tableaux ou autres.
3. **Par paquet :** même chose que par données, sauf que les paramètres sont cette fois-ci 'composé', comme des structures ou des classes
4. **Par contrôle :** les logiciels s'échangent leur contrôle en renvoyant leur 'verrou', ou drapeaux entre eux.
5. **Par liaison Externe :** Une communication externe (comme un fichier ou un lien de communication) est mise en place entre les deux composants.
6. **De type commun :** un ensemble de données communs entre les deux services est utilisé.
7. **Par contenu :** niveau le plus élevé, chaque service à accès aux données de l'autre, et vient donc puiser directement à l'intérieur.

La bonne pratique est d'avoir un niveau de couplage le plus bas possible, surtout dans les web services, car cela rend beaucoup plus difficile d'utiliser le service ailleurs, et complique le suivi des données.(26)

4.4.3Les applications format distribuées

Une application distribuée s'installe sur deux machines distantes afin de les faire communiquer, souvent par le biais d'objets. SOAP, et bien avant, CORBA (OMG), DCOM (MS), RMI (SUN) et bien d'autres utilisent tous cette technologie.(26)

4.4.4Les architecture SOA (orientée services)

Une architecture orientée services fait communiquer entre eux des objets en transcrivant la demande du serveur vers le client. Le message est donc traduit pour répondre à la demande.(26)

4.5 Les aspects technologiques

Le web service a besoin de plusieurs couches pour fonctionner :

- La couche invocation décrivant la structure des messages échangés
- La couche découverte pour trouver le web service au sein de l'annuaire de web services (nom de société, utilité de chaque service)
- et la couche description, contenant paramètres des fonctions, typages des données du web services.

Nous verrons dans ce point les différents aspects technologiques entourant les web services.(26)

4.5.1Transport : SOAP (Simple Object Acces Protocol)

SOAP est un protocole de communication d'ordinateur à ordinateur sous HTTP très simple, écrit en XML. Il permet l'échange de données, quel que soit les systèmes d'exploitation. Les messages SOAP sont des transmissions en sens unique d'un émetteur vers un récepteur.(26)

4.5.2Découverte : UDDI (Universal Description, Discovery and Integration)

UDDI est une norme d'annuaire de services Web appelée via le protocole SOAP. Pour publier un nouveau service Web, il faut générer un document appelé **Business Registry**, il sera enregistré sur un UDDI Business Registry Node (IBM, Microsoft et SAP en hébergent chacun un). Les nodes sont répliqués entre eux suivant un mécanisme analogue aux DNS. Le Business Registry comprend 3 parties :

- **Pages blanches** noms, adresses, contacts, identifiants des entreprises enregistrées.

- **Pages jaunes** informations permettant de classer les entreprises, notamment l'activité (classifications NAICS, UNSPSC...), la localisation...
- **Pages vertes** informations techniques sur les services proposés.

Le protocole utilise 3 fonctions de base :

- **publish** pour enregistrer un nouveau service,
- **find** pour interroger l'annuaire,
- **bind** pour effectuer la connexion entre l'application cliente et le service.

Comme pour la certification, il est possible de constituer des annuaires UDDI privés, dont l'usage sera limité à l'intérieur de l'entreprise.

UDDI est maintenant un standard employé.(26)

4.5.3Description : WSDL (Web Services Description Language)

WSDL, basé sur XML, permet de décrire le service Web, en précisant les méthodes disponibles, les formats des messages d'entrée et de sortie, et comment y accéder. (28)

Pour écrire le programme d'invocation du service, on utilisera les informations suivantes :

- **types :** définitions de types de donnée échangées.
- **message :** définition abstraite des données transmises
- **type de port :** ensemble d'opérations correspondant chacune à un message entrant ou sortant.
- **rattachement (binding) :** protocole de communication et format des données échangées pour un port.
- **port :** adresse (assure l'unicité du rattachement).
- **service :** regroupe un ensemble de ports. (28)

4.6 Les types de service web

Il existe différentes méthodes pour fournir des services Web, mais les plus courantes sont SOAP et REST :

4.6.1SOAP (Simple Object Access Protocol)

SOAP signifie Simple Object Access Protocol. Créé en 1998, l'objectif est de l'utiliser sur le marché des entreprises. En particulier, SOAP doit créer une logique d'application en tant que service. L'objectif est d'être un nouveau protocole de communication.(29)

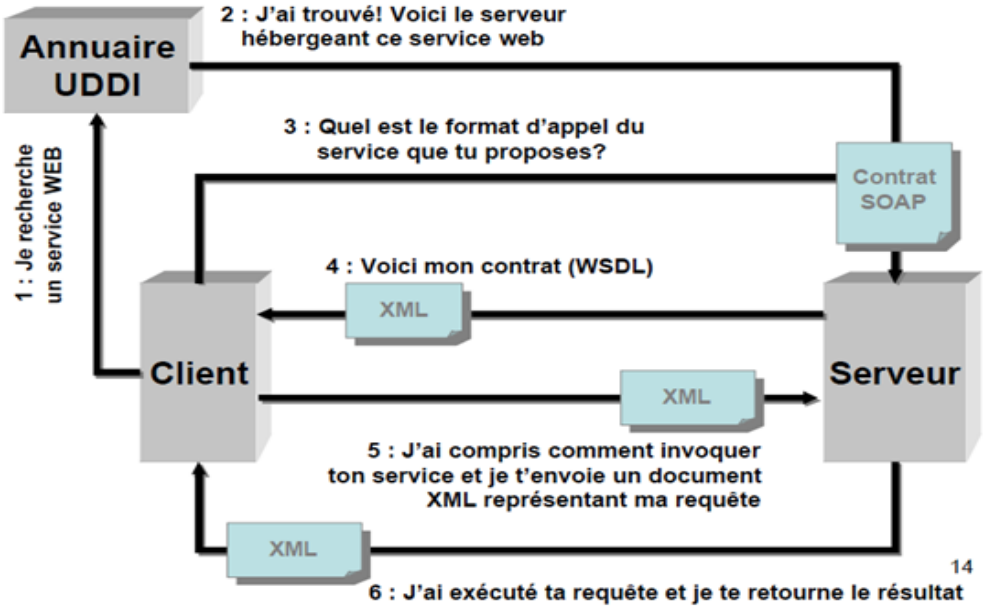


Figure 4-3 Architecture d'un Service WEB SOAP(26)

Avantages

- capable de travailler sur n'importe quel protocole.
- Expliquer le service avec WDSL (Web Service Description Language)
- Fiable Lorsqu'un problème survient, il peut être réessayé.
- La sécurité existe déjà, à la fois l'authentification, l'autorisation et le cryptage des données.(29)

Désavantage

- Difficile à développer. Rendre populaire pour le Web et le mobile.
- formats de données pris en charge, XML seul.
- parce que c'est une norme, il y a la limitation, mais en raison de ses nombreuses parties, il y a une surcharge ou le besoin de bande passante est plus élevé que le REST.(29)

Quand Utiliser SOAP

- lors de la gestion des transactions lorsque vous travaillez avec plusieurs systèmes.
- Lors de la connexion à une connexion stricte entre client / serveur
- Lorsque, par exemple, un service financier et un service de télécommunication,(29)

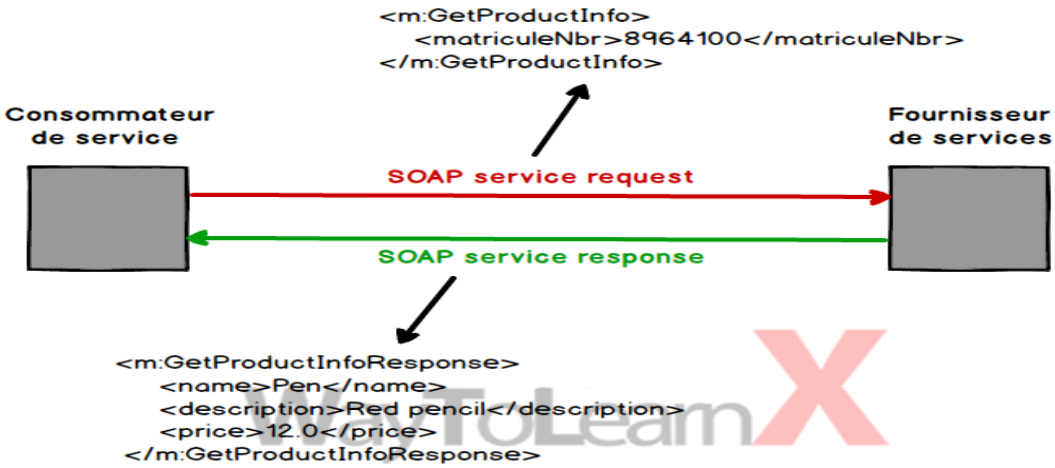


Figure 4-4 : Exemple d'une demande (request) dans le service SOAP(30)

4.6.2Method REST (Representational State Transfer)

REST signifie Représentationnel State Transfer et c'est un style architectural qui permet la communication entre les systèmes. Le terme REST a été inventé pour la première fois par Roy T. Fielding dans son doctorat. Dissertation.(31)

REST a été créé depuis 2000. Le but est d'être l'une des conceptions de technologie Web ouverte qui souhaite mettre les données sous la forme d'une ressource. Les actions suivent le verbe HTTP ou la méthode HTTP (GET, POST, PUT, DELETE).(29)

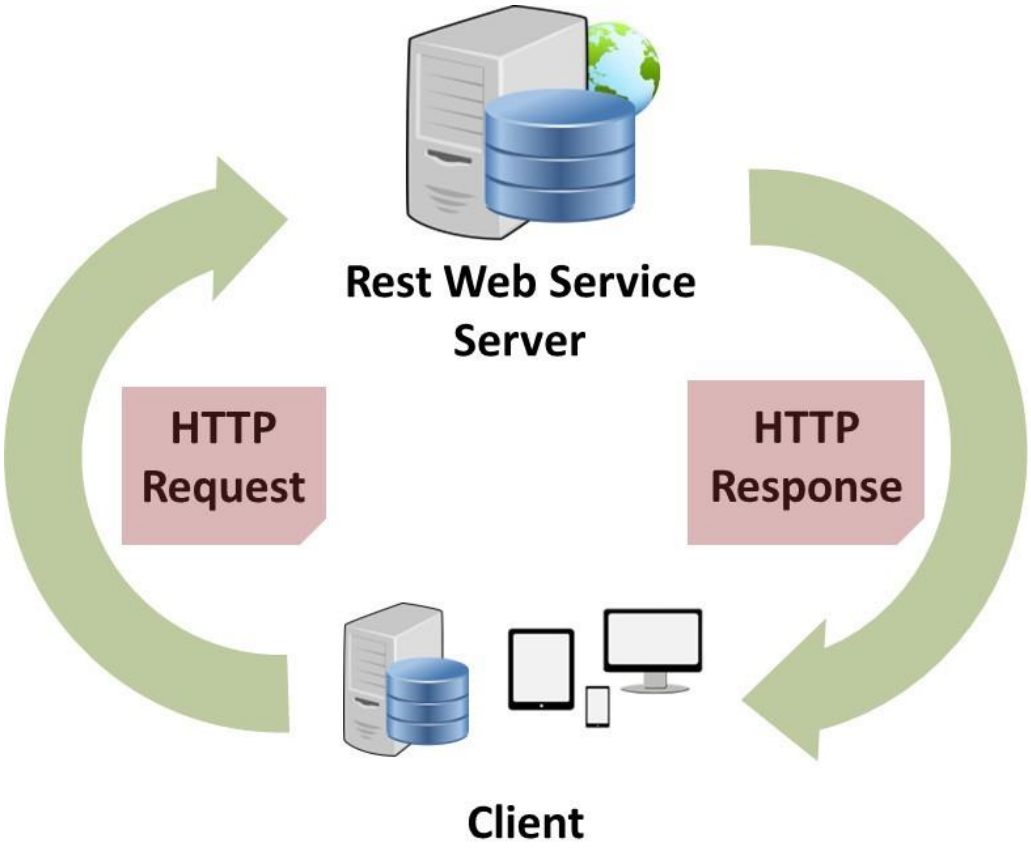


Figure 4-5 Architecture REST (32)

Avantages

- Il est basé sur HTTP et est conforme aux normes HTTP, il peut donc être facilement développé.
- Prend en charge de nombreux formats de données tels que XML, JSON, texte brut et bien d'autres.
- Prise en charge de Easy to expand du système
- Bonne performance
- Prend en charge la mise en cache des données.(29)

Désavantages

- Cela fonctionne uniquement avec le protocole HTTP.
- Il n'y a pas de sécurité et de fiabilité intégrée, alors faites-le vous-même
- formats de données envoyés entre client-serveur Il n'y a pas de restrictions.(29)

Quand utiliser REST

- lorsque vous souhaitez réduire la quantité de données et la quantité de bande passante utilisée
- lorsque cela est nécessaire lorsque vous travaillez sur des systèmes Web et mobiles, par exemple, un service de médias sociaux, un service de chat Web, il n'est donc pas étrange de savoir pourquoi les systèmes Web et API Via le Web est donc REST.(29)

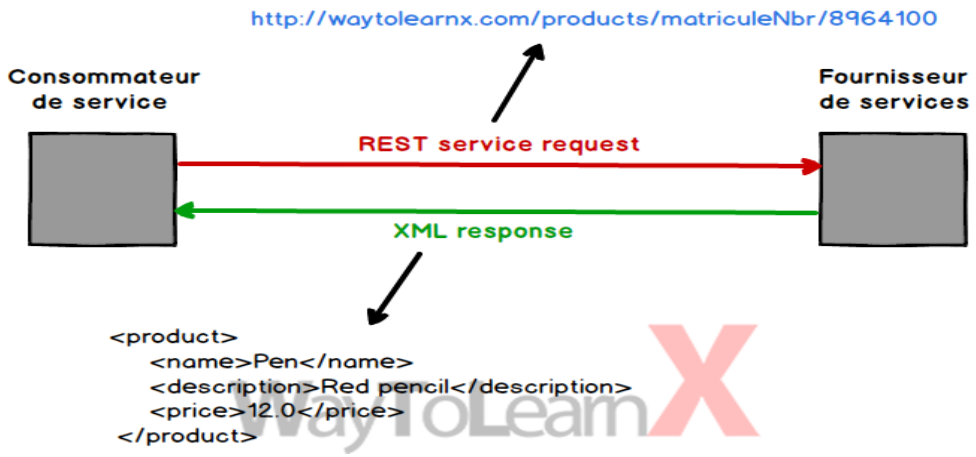


Figure 4-6 : Exemple d'une demande (request) dans le service REST(30)

4.6.3Tableau comparatif entre REST et SOAP

Après étude de différentes architectures utilisées dans les web services, on a remarqué que l'architecture REST est la plus adaptée pour notre application. Pour cela, on va montrer dans un tableau, qui représente les comparatifs entre l'architecture REST et SOAP, et qui montre les points forts de REST.

4.7 REST vs SOAP

REST et SOAP proposent différentes méthodes pour appeler un service Web. REST est un style architectural, tandis que SOAP définit une spécification de protocole de communication standard pour l'échange de messages XML. Les applications REST peuvent utiliser SOAP.(33)

Les services Web REST sont sans état. Une implémentation basée sur REST est simple par rapport à SOAP, mais les utilisateurs doivent comprendre le contexte et le contenu transmis,

car il n'y a pas d'ensemble standard de règles pour décrire l'interface des services Web REST. Les services REST sont utiles pour les appareils à profil restreint, tels que les mobiles, et sont faciles à intégrer aux sites Web existants.(33)

SOAP nécessite moins de code de plomberie c'est-à-dire un code d'infrastructure de bas niveau qui relie les principaux modules de code entre eux que la conception de services REST. Le langage de description des services Web décrit un ensemble commun de règles pour définir les messages, les liaisons, les opérations et l'emplacement du service. Les services Web SOAP sont utiles pour le traitement et l'appel asynchrones. (33)

Tableau 1 : Tableau de comparaison entre les caractéristiques des Services WEB basés sur SOAP et REST

SOAP	REST
Les opérations sont définies comme WSDL les ports	Les opérations sontdéfinies dans les messages
Adresse unique pourchaque opération	Adresse unique pourchaque instance de processus
Instances des processus multiples partagent lamême opération	Chaque objet supporte les opérations(standards) définies
Couplage serré deComposants	Couplage lâche deComposants
Le débogage est possible	La liaison tardive estpossible
Les opérations complexes peuvent être cachées derrière la façade	Les instances des processus sont créées explicitement

.	
Emballage existantAPI estsimple	Le client n'a pas besoin desinformations de routageau-delà de l'initialeURI de l'usine de processus (ProcessFactory)
Confidentialité augmentée	Le client peut en avoir une interface auditeur générique pour les notifications
Les clients doivent connaître au préalable les opérations et leurs sémantiques	Grand nombre d'objets
Le client a besoin de ports dédiés pour des différents types de notification Les instances des processus sont créées implicitement	La gestion de l'URI peut devenir lourde

4.8 RESTFULL API

Nous allons voir ce qu'est RESTful API et à quoi cela sert, mais avant cela, il est nécessaire de comprendre ce qu'est une API.

4.8.1API

Une API (Application Programming Interface),en français Interface de Programmation Applicative, est un moyen de mettre à disposition des méthodes et services à des applications tierces. Cela va permettre aux développeurs de réutiliser du code existant, plutôt que de devoir réinventer la roue à chaque fois. Un service web est un type d'API, accessible via internet, souvent via le protocole http (HyperText Transfert Protocole). Par exemple, l'utilisation du paiement par Paypal dans des applications tierces se fait via l'API mise à disposition par

Paypal. Ainsi, le système d'authentification est géré, et au lieu de passer énormément de temps à développer, il suffit de créer un compte développeur et récupérer la clé d'API, il en est de même pour l'API Facebook, Google…

4.8.2Définition

Le concept de REST est défini par certaines règles, contraintes ou principes. Le système, l'application, les services ou tout ce qui satisfait à ces principes REST sont appelés RESTful.(31)

Les services Web qui suivent les principes RESTful sont des services RESTful. L'URI est utilisé pour accéder aux services RESTful afin d'obtenir les ressources.(31)Dans le glossaire RESTful, les ressources ne sont que des données et des fonctions. Nous allons donc éventuellement appeler les services Web via URI pour accéder aux fonctions et ainsi obtenir les données de ressources.(31)

4.8.3Architecture d'API de services Web RESTful :

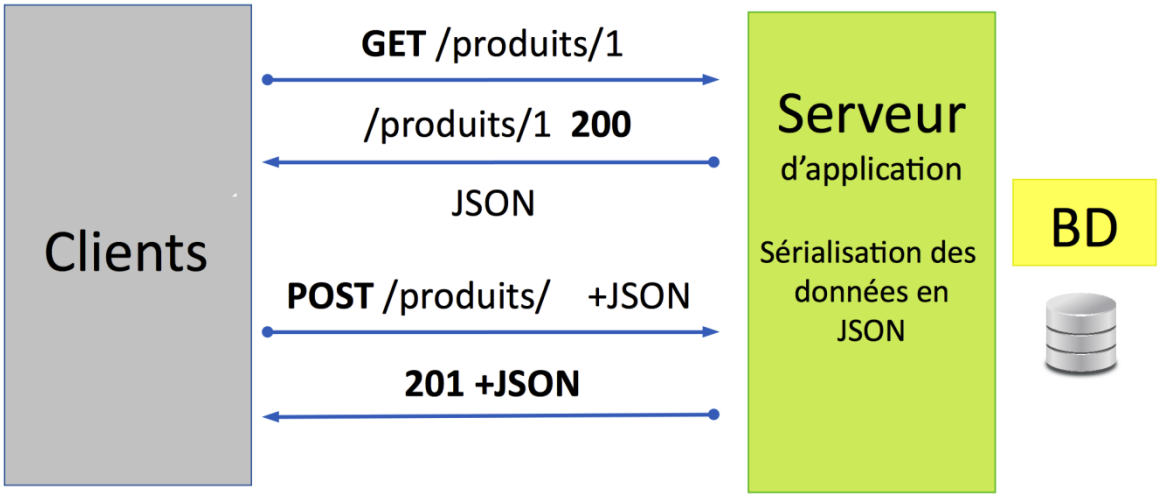


Figure 4-7 : Architecture d'une API de services web RESTful

4.8.4Fonctionnalités des services RESTFUL :

Une API RESTful décompose une transaction pour créer une série de petits modules. Chaque module aborde une partie sous-jacente particulière de la transaction. Cette modularité offre aux développeurs une grande flexibilité, mais il peut être difficile pour les développeurs de concevoir leur API REST à partir de zéro. Actuellement, plusieurs entreprises fournissent des

modèles à utiliser par les développeurs; les modèles fournis par Amazon S3 , Cloud Data Management Interface (CDMI) et Open Stack Swift sont les plus populaires.(33)

Une API RESTful utilise les méthodes HTTP existantes définies par le protocole RFC 2616. Ils utilisent GET pour récupérer une ressource; PUT pour changer l'état ou mettre à jour une ressource, qui peut être un objet , un fichier ou un bloc ; POST pour créer cette ressource; et DELETE pour le supprimer.(33)

4.8.4.1 Les Méthodes HTTP

4.8.4.1.1 GET

Utilisez les requêtes GET pour récupérer uniquement la représentation / les informations des ressources - et non pour les modifier en aucune façon. Comme les requêtes GET ne changent pas l'état de la ressource, on dit que ce sont des méthodes sûres. De plus, les API GET doivent être idempotentes**,** ce qui signifie que faire plusieurs requêtes identiques doit produire le même résultat à chaque fois jusqu'à ce qu'une autre API (POST ou PUT) ait changé l'état de la ressource sur le serveur.
Si l'URI de demande fait référence à un processus de production de données, ce sont les données produites qui doivent être renvoyées en tant qu'entité dans la réponse et non le texte source du processus, à moins que ce texte ne soit la sortie du processus.(34)

Pour toute API HTTP GET donnée, si la ressource est trouvée sur le serveur, elle doit renvoyer le code de réponse http 200 (OK) avec le corps de la réponse, qui est généralement du contenu XML ou JSON (en raison de leur nature indépendante de la plate-forme).
Si la ressource n'est PAS trouvée sur le serveur, elle doit renvoyer le code de réponse http 404 (NOT FOUND) De même, s'il est déterminé que la requête GET elle-même n'est pas correctement formée, le serveur renverra le code de réponse http 400 (BAS REQUEST).(34)

Exemple d'URI de demande :

- GET http://www.appdomain.com/users
- GET http://www.appdomain.com/users?size=20&page=5
- GET http://www.appdomain.com/users/123
- GEThttp://www.appdomain.com/users/123/address

4.8.4.1.2 POST

Utilisez les API POST pour créer de nouvelles ressources subordonnées, par exemple, un fichier est subordonné à un répertoire le contenant ou une ligne est subordonnée à une table de base de données. Lorsqu'on parle strictement en termes de REST, les méthodes POST sont utilisées pour créer une nouvelle ressource dans la collection de ressources.(34)

Idéalement, si une ressource a été créée sur le serveur d'origine, la réponse devrait être un code de réponse HTTP 201(Created) et contenir une entité qui décrit l'état de la demande et se réfère à la nouvelle ressource, et un en- tête Location .

Plusieurs fois, l'action effectuée par la méthode POST peut ne pas aboutir à une ressource pouvant être identifiée par un URI. Dans ce cas, il s'agit du code de réponse HTTP 200 (OK) ou de 204 (NO Content) l'état de réponse approprié.
Les réponses à cette méthode sont pas mis en cache, à moins que la réponse inclut appropriée Cache-Control ou Expires champs en- tête. (34)

Veuillez noter que POST n'est ni sûr ni idempotent, et l'invocation de deux requêtes POST identiques se traduira par deux ressources différentes contenant les mêmes informations (à l'exception des identifiants de ressource).(34)

Exemple d'URI de demande :

- POST http://www.appdomain.com/users
- POST http://www.appdomain.com/users/123/accounts

4.8.4.1.3 PUT

Utilisez les API PUT principalement pour mettre à jour la ressource existante (si la ressource n'existe pas, l'API peut décider de créer une nouvelle ressource ou non). Si une nouvelle ressource a été créée par l'API PUT, le serveur d'origine doit informer l'agent utilisateur via la réponse du code de 201(Created) réponse HTTP et si une ressource existante est modifiée,

les codes de réponse 200 (OK) ou 204 (NO Content) DEVRAIENT être envoyés pour indiquer la réussite de la demande. (34)
Si la demande passe par une antémémoire et que l'URI de demande identifie une ou plusieurs entités actuellement mises en cache, ces entrées devraient être traitées comme périmées. Les réponses à cette méthode ne peuvent pas être mises en cache**.**(34)

La différence entre les API POST et PUT peut être observée dans les URI de requête. Les requêtes POST sont effectuées sur des collections de ressources, tandis que les requêtes PUT sont effectuées sur une seule ressource.(34)

Exemple d'URI de demande :

- PUT http://www.appdomain.com/users/123
- PUT http://www.appdomain.com/users/123/accounts/456

4.8.4.1.4 DELLET

Comme le nom s'applique, les API DELETE sont utilisées pour supprimer des ressources (identifiées par Request-URI).(34)

Une réponse réussie aux demandes DELETE devrait être une réponse HTTP code 200 (OK)
Si la réponse comprend une entité décrivant l'état, 202 (Accepted) si l'action a été mise en file d'attente, ou 204 (NO Content) si l'action a été exécutée mais la réponse n'inclut pas d'entité.
Les opérations DELETE sont idempotentes. Si vous supprimez une ressource, elle est supprimée de la collection de ressources. L'appel répété de l'API DELETE sur cette ressource ne changera pas le résultat - cependant, appeler DELETE sur une ressource une deuxième fois renverra un 404 (NOT FOUND) car il a déjà été supprimé. Certains peuvent prétendre que cela rend la méthode DELETE non idempotente. C'est une question de discussion et d'opinion personnelle. Si la demande passe par une antémémoire et que l'URI de demande identifie une ou plusieurs entités actuellement mises en cache, ces entrées devraient être traitées comme périmées. Les réponses à cette méthode ne peuvent pas être mises en cache**.**(34)

Exemple d'URI de demande :

- DELETE http://www.appdomain.com/users/123
- DELETE http://www.appdomain.com/users/123/accounts/456

4.8.5Composants en Réseau de REST

Avec REST, les composants en réseau sont une ressource à laquelle l'utilisateur demande l'accès - une boîte noire dont les détails d'implémentation ne sont pas clairs. Tous les appels sont sans état; rien ne peut être conservé par le service RESTful entre les exécutions (33)

4.8.6Contraintes de conception et d'architecture d'API RESTful

Pour créer une véritable API RESTful, elle doit respecter les six contraintes architecturales REST suivantes:

Utilisation d'une interface uniforme (UI) : Les ressources doivent être identifiables de manière unique via une URL unique, et uniquement en utilisant les méthodes sous-jacentes du protocole réseau, telles que DELETE, PUT et GET avec HTTP, s'il est possible de manipuler une ressource. (33)

Basé sur le client-serveur : Il doit y avoir une délimitation claire entre le client et le serveur. Les problèmes liés à l'interface utilisateur et à la collecte des demandes sont du domaine du client. L'accès aux données, la gestion de la charge de travail et la sécurité sont le domaine du serveur. Ce couplage lâche du client et du serveur permet à chacun d'être développé et amélioré indépendamment de l'autre. (33)

Opérations apatrides : Toutes les opérations client-serveur doivent être sans état et toute gestion d'état requise doit avoir lieu sur le client et non sur le serveur. (33)

Mise en cache des ressources RESTful : Toutes les ressources doivent autoriser la mise en cache, sauf indication explicite que la mise en cache n'est pas possible.(33)

Système en couches : REST permet une architecture composée de plusieurs couches de serveurs. (33)

Code sur demande : La plupart du temps, un serveur renvoie des représentations statiques de Ressources sous forme de XML ou JSON . Cependant, si nécessaire, les serveurs peuvent envoyer du code exécutable au client.(33)

Conclusion

On à vue dans ce chapitre l'importance et l'avantage des services web, ces aspects technologiques, et deux type qui sont SOAP et REST avec une comparaison pour montrer quelle est la meilleur solution pour notre application, on à vue aussi c quoi une API et le RESTFUL API et en quoi le protocole REST est avantageux pour écrire des services web, notamment par rapport à SOAP.

Après cette étude, on a remarqué que l'architecture REST est la plus adaptée pour notre application on résume que REST est plus Structurant à cause du nommage des URI, Efficace et facile à développer à cause de l'utilisation de Protocol HTTP comme style de communication, ilprend en charge de nombreux formats de données tels que XML, JSON, texte brut et bien d'autres.

Le chapitre suivant décrit la phase de conception et réalisation de notre application en utilisant un système d'information qui existe déjà.

Chapitre 5 : Conception et réalisation de l'application mobile

5 Introduction

L'objectif de notre mémoire est de réaliser une application mobile à partir d'un système qui existe déjà, Le service web qu'on a choisi concerne le management d'une école, pour cela on va faire une application mobile avec le React native qui permet de connecter les enseignants, les élèves et les parents via une interface unique sur les deux systèmes mobile les plus populaire : Android et IOS.

Dans ce chapitre nous allons parlez des outils et technologies utilisés dans la conception et la réalisation de notre application mobile et ces fonctionnalité, l'interface et des exemples sur quelque scénarios d'utilisation de l'application.

5.1 Outils utilisé

5.1.1 Visual studio code

Visual Studio Code est un éditeur de code source développé par Microsoft pour Windows, Linux et macOS . Il comprend Git intégré et la prise en charge du débogage, de la mise en évidence de la syntaxe, de l'achèvement intelligent du code, des extraits de code et de la refactorisation du code. Il est hautement personnalisable, permettant aux utilisateurs de modifier le thème, les raccourcis clavier, les préférences et d'installer des extensions qui ajoutent des fonctionnalités supplémentaires. Le code source est gratuit et open-source, publié sous la licence permise MIT. Les binaires compilés sont des logiciels gratuits pour toute utilisation.(35)

Dans le StackOverflow 2019 Développer Survey, Visual Studio Code a été classé l'outil d'environnement de développeur le plus populaire, avec 50,7% des 87317 répondants déclarant l'utiliser.(35)

5.1.2 XAMPP

XAMPP est un ensemble de logiciels permettant de mettre en place un serveur Web local, un serveur FTP et un serveur de messagerie électronique. Il s'agit d'une distribution

de logiciels libres (X (cross) Apache MariaDB Perl PHP) offrant une bonne souplesse d'utilisation, réputée pour son installation simple et rapide. Ainsi, il est à la portée d'un grand nombre de personnes puisqu'il ne requiert pas de connaissances particulières et fonctionne, de plus, sur les systèmes d'exploitation les plus répandus.(36)

5.1.3 PhpMyAdmin

Est une interface web en PHP pour administrer à distance les SGBD MySQL et MariaDB.

Il permet d'administrer les éléments suivants :

-les bases de données

-les tables et leurs champs (ajout, suppression, définition du type)

-les index, les clés primaires et étrangères

-les utilisateurs de la base et leurs permissions

-importer ou exporter les données dans divers formats (CSV, XML, PDF, OpenDocument, Word, Excel et LaTeX) (37)

5.1.4 GitHub

GitHub (exploité sous le nom de *GitHub, Inc.*) est un service web d'hébergement et de gestion de développement de logiciels, utilisant le logiciel de gestion de versions Git. Ce site est développé en Ruby on Rails et Erlang par Chris Wanstrath, PJ Hyett et Tom Preston-Werner. GitHub propose des comptes professionnels payants, ainsi que des comptes gratuits pour les projets de logiciels libres. Le site assure également un contrôle d'accès et des fonctionnalités destinées à la collaboration comme le suivi des bugs, les demandes de fonctionnalités, la gestion de tâches et un wiki pour chaque projet.(38)

Heroku : est une plate-forme cloud qui permet aux entreprises de créer, de fournir, de surveiller et de mettre à l'échelle des applications.(39)

5.2 Technologies utilisés

5.2.1 React

React (aussi appelé React.js ou ReactJS) est une bibliothèque JavaScript libre développée par Facebook depuis 2013. Le but principal de cette bibliothèque est de faciliter la création

d'application web monopage, via la création de composants dépendant d'un état et générant une page (ou portion) HTML à chaque changement d'état.(40)

5.2.2 React Native

React Native est un framework JavaScript permettant d'écrire de vraies applications mobiles de rendu natif pour iOS et Android. Il est basé sur React, la bibliothèque JavaScript de Facebook pour créer des interfaces utilisateur, mais au lieu de cibler le navigateur, il cible les plates-formes mobiles. En d'autres termes: les développeurs Web peuvent désormais écrire des applications mobiles qui semblent et se sentent vraiment «natives», le tout dans le confort d'une bibliothèque JavaScript que nous connaissons et aimons déjà. De plus, comme la plupart du code que vous écrivez peut être partagé entre les plates-formes, React Native facilite le développement simultané pour Android et iOS.(41)

5.2.3 Expo

EXPO est un cadre et une plate-forme pour les applications React universelles. Il s'agit d'un ensemble d'outils et de services construits autour des plates-formes react native et natives qui vous aident à développer, créer, déployer et itérer rapidement sur des applications iOS, Android et Web à partir de la même base de code JavaScript / Type Script.(42)

5.2.4 MySQL

MySQL est un système de gestion de bases de données relationnelles (SGBDR). Il est distribué sous une double licence GPL et propriétaire. Il fait partie des logiciels de gestion de base de données les plus utilisés au monde4, autant par le grand public (applications web principalement) que par des professionnels, en concurrence avec Oracle, PostgreSQL et Microsoft SQL Server. (43)

5.2.5 Git

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. En 2016, il s’agit du logiciel de gestion de versions le plus populaire qui est utilisé par plus de douze millions de personnes. (44)

Remarque : On à utiliser Git pour gérer notre projet d'application à distance à cause de la pandémie de COVIC-19.

5.3 Application Mobile

5.3.1Présentation des Modules du Système d'information

Notre Application (School Management)comprend plusieurs modules qui s'orchestrent ensemble pour définir Ces services, voici la liste des modules:

- Groupes d'utilisateurs: enseignants, élèves et parents
- Inscription: L'administrateur doit approuver les inscriptions des enseignants, des élèves ou des parents
- Messages privés: moyen simple de communiquer avec d'autres utilisateurs par messages
- Nouveautés: publier les actualités de l'école à bord
- Événements: publier des événements programmés
- Présence: suivre la présence des étudiants
- Examens: ajoutez des examens prévus pour les élèves et les niveaux scolaires
- Devoirs: téléchargez les devoirs liés à chaque classe
- Examens en ligne: créez des examens en ligne (choix multiples) pour les étudiants avec marquage automatique
- Dortoirs: ajoutez vos dortoirs scolaires
- Classes: ajoutez vos classes scolaires
- Horaire des cours: créez un horaire de cours pour chacun
- Sujets: ajouter des sujets et leur relation avec les enseignants
- Bibliothèque de livres: créez une bibliothèque en ligne par téléchargement ou par disponibilité des livres sur la bibliothèque
- Galerie Media: voir l'ensemble d'albums multimédia et les télécharger
- Payements: voir la liste des payements effectués et à effectués

- Pages Statiques: consulter le contenu des pages statiques
- Transportations: voir la liste des transportations offerts par l'école et leurs informations
- Sondages : voir la liste des sondages et voter des sondages.

5.3.2 Rôles des utilisateurs

Notre Application (School Management) à 4 rôles d'utilisateur :

1. Administrateurs
    - Créer de nouveaux comptes et approuver les demandes d'inscription
    - Envoyer des E-mails / SMS aux utilisateurs
    - Gérer les paramètres du site, les promotions, les dortoirs, les classes, les sujets, les actualités, les événements, les pages statiques, les niveaux scolaires, la présence, les devoirs, les examens et leurs notes, les examens en ligne, les horaires des classes, les sondages, la bibliothèque, la galerie media, les payements, les transportations
    - Peut utiliser des messages privés avec d'autres utilisateurs du site
    - Télécharger des livres et des devoirs
    - Effectuer des payements
2. Enseignants
    - Consulter les informations des étudiants et leurs parents
    - Consulter le calendrier des cours
    - Consulter / gérer la présence des étudiants
    - Consulter/ gérer les notes des examens
    - Consulter/ gérer les examens en ligne et leurs notes
    - Consulter/ gérer les devoir, et les télécharger
    - Consulter/ voter les sondages
    - Consulter/ envoyer des messages privés aux autres utilisateurs
    - Consulter la bibliothèque, télécharger et vérifier l'état du livre
    - Consulterla galerie media et télécharger ses fichiers
    - Consulter les actualités, les événements, les pages statiques, niveaux scolaires, horaires des classes, les sujets, les transportations
3. Étudiants

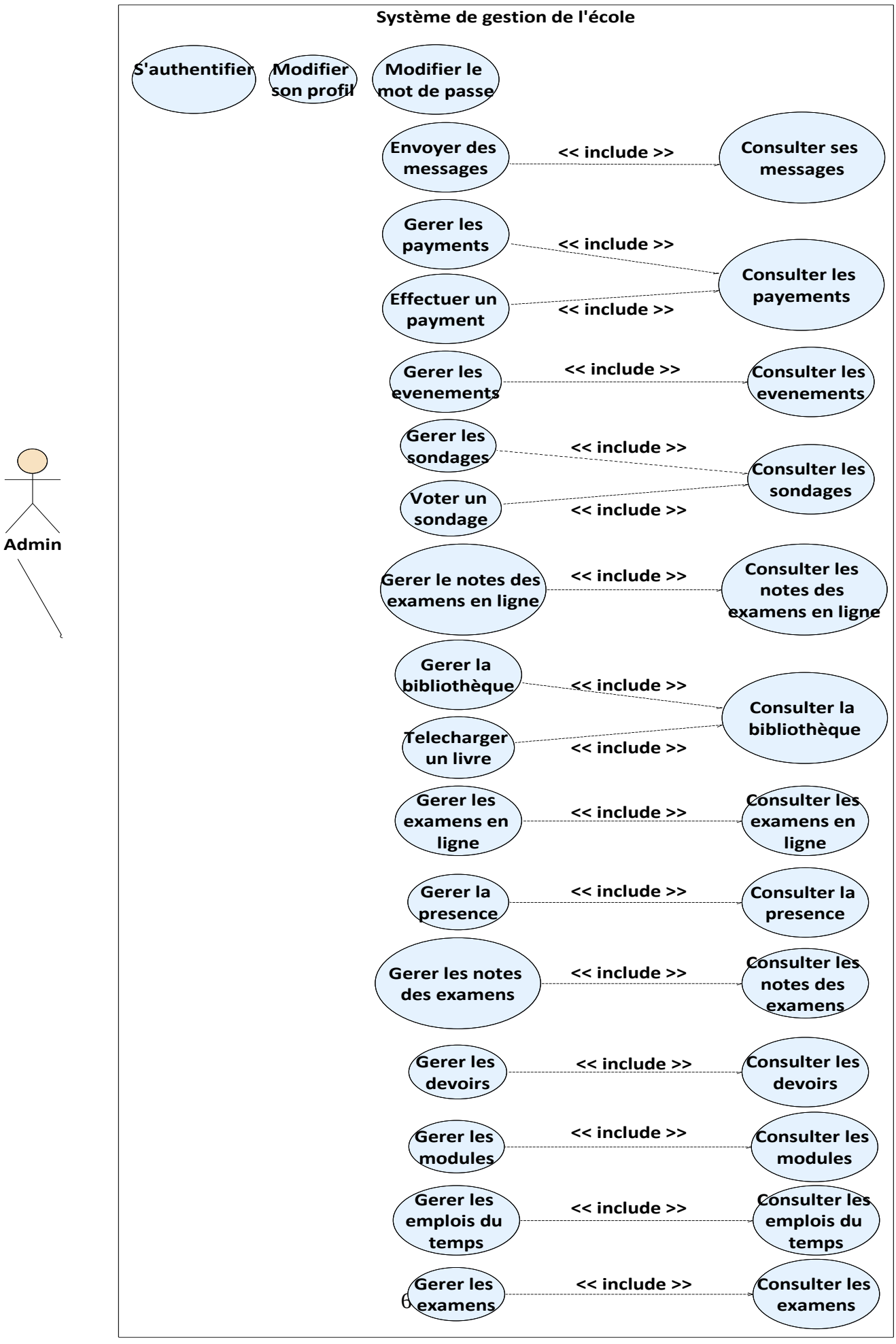
- Consulter les examens et leurs notes
- Consulter et télécharger les devoirs
- Consulter / passer les examens en ligne et voir leurs notes
- Consulter/ voter les sondages
- Consulter/ envoyer des messages privés aux autres utilisateurs
- Consulter la bibliothèque, télécharger et vérifier l'état du livre
- Consulter la galerie media et télécharger ses fichiers
- Consulter les actualités, les événements, les pages statiques, niveaux scolaires, horaires des classes, les sujets, la présence, les payements, les transportations

4. Parents
    - Consulter les examens et leurs notes
    - Consulter et télécharger les devoirs
    - Consulter les examens en ligne et voir leurs notes
    - Consulter/ voter les sondages
    - Consulter/ envoyer des messages privés aux autres utilisateurs
    - Consulter la bibliothèque, télécharger et vérifier l'état du livre
    - Consulter la galerie media et télécharger ses fichiers
    - Consulter les actualités, les événements, les pages statiques, niveaux scolaires, horaires des classes, les sujets, la présence, les payements, les transportations

5.4 Diagrammes UML

5.4.1 Diagramme du cas d'utilisation

On va utiliser le DCU(Diagramme du cas d'utilisation) pour donner une vision globale du comportement fonctionnel de notre système, dans ce diagramme on va montrer les Acteurs (Admin, Enseignent, Etudiant, Parent) et leurs cas d'utilisation.


Système de gestion de l'école
S'authentifier
Modifier son profil
Modifier le mot de passe
Envoyer des messages
<< include >>
Consulter ses messages
Gerer les payments
Effectuer un payment
<< include >>
Consulter les payements
Gerer les evenements
<< include >>
Consulter les evenements
Gerer les sondages
Voter un sondage
<< include >>
Consulter les sondages
Admin
Gerer le notes des examens en ligne
<< include >>
Consulter les notes des examens en ligne
Gerer la bibliothèque
Telecharger un livre
<< include >>
Consulter la bibliothèque
Gerer les examens en ligne
<< include >>
Consulter les examens en ligne
Gerer la presence
<< include >>
Consulter la presence
Gerer les notes des examens
<< include >>
Consulter les notes des examens
Gerer les devoirs
<< include >>
Consulter les devoirs
Gerer les modules
<< include >>
Consulter les modules
Gerer les emplois du temps
<< include >>
Consulter les emplois du temps
Gerer les examens
<< include >>
Consulter les examens

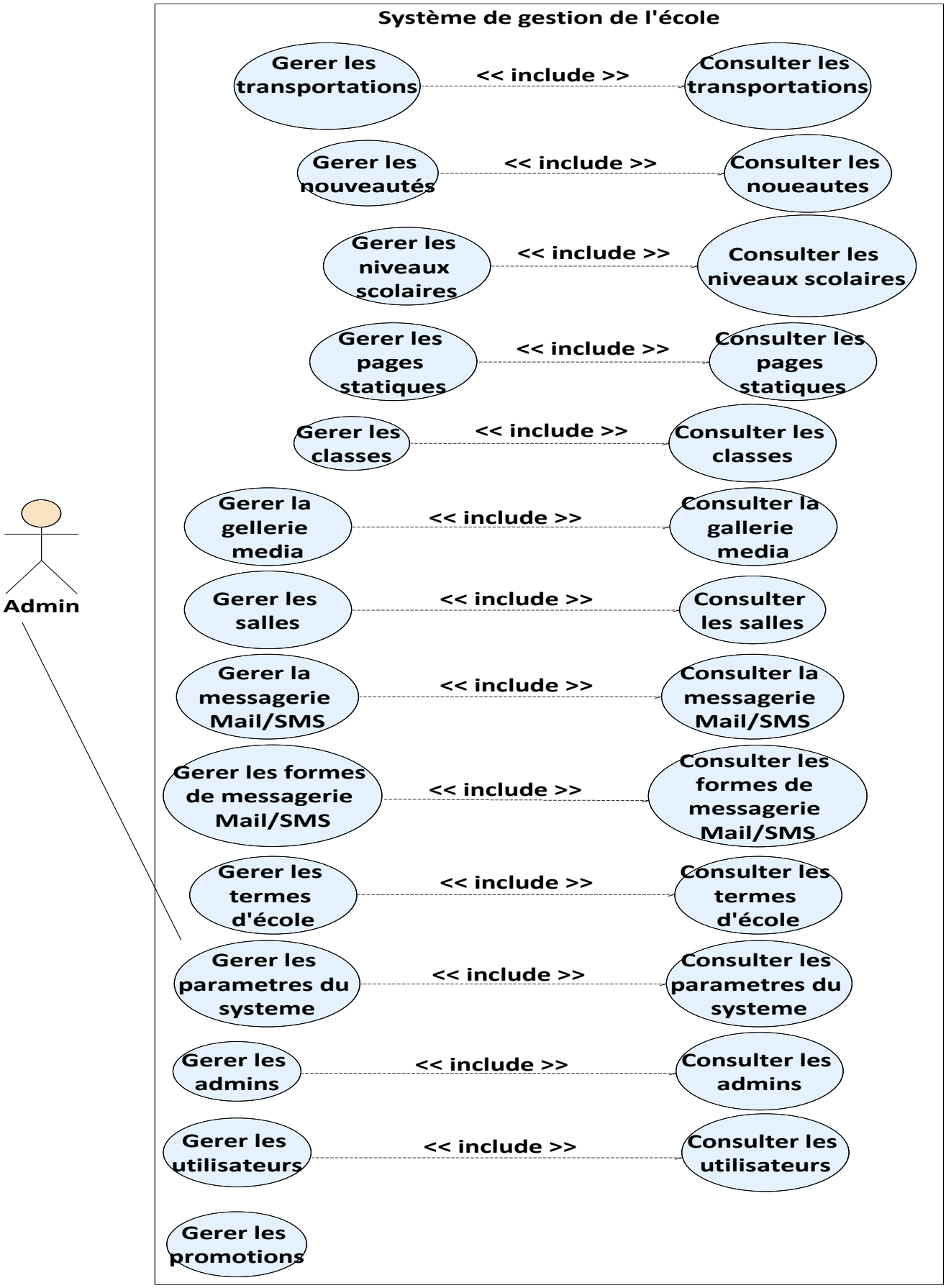


Figure 5-1: Diagramme du modèle de cas d'utilisation de l'admin

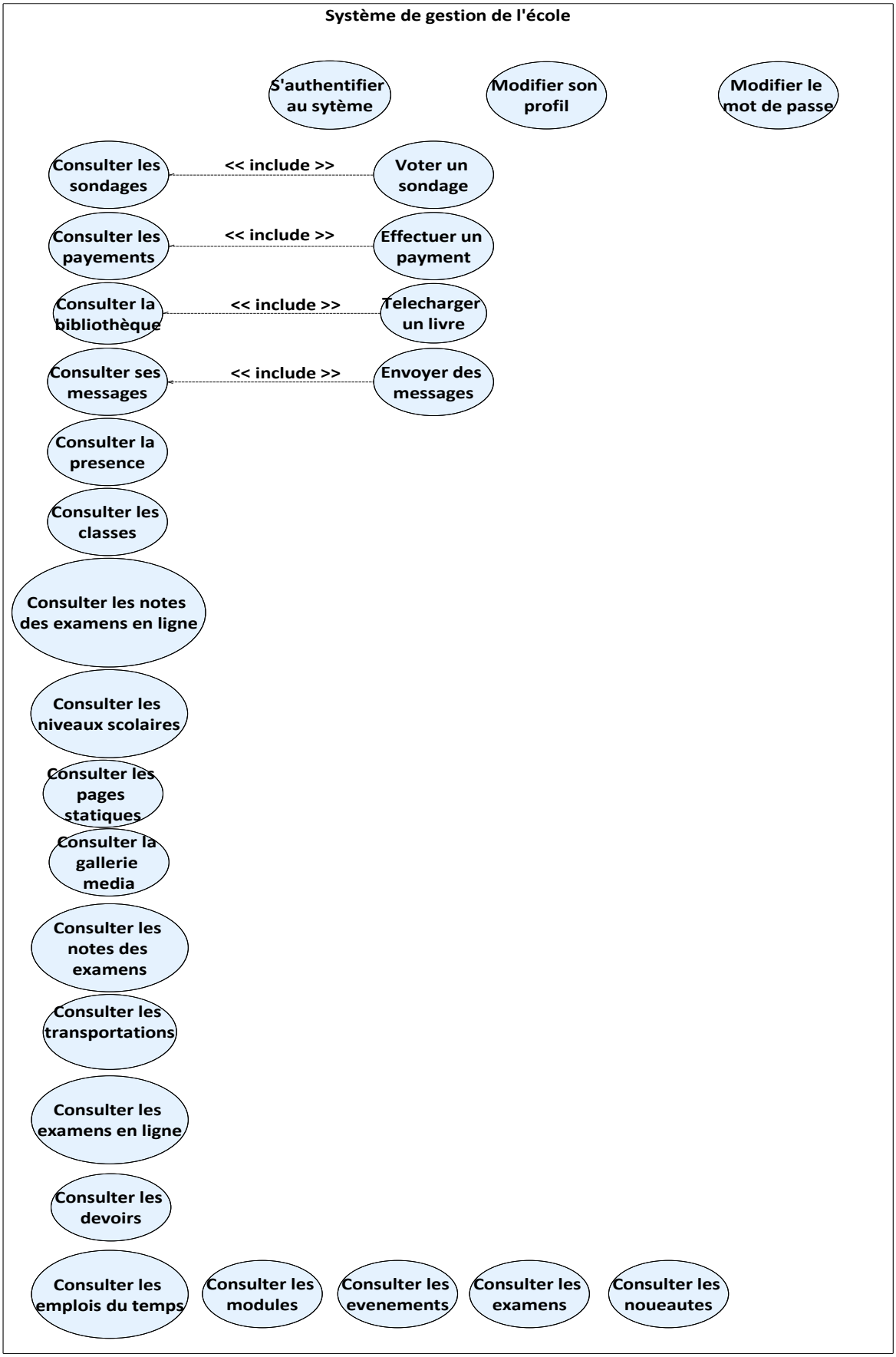


Figure 5-2 Diagramme du modèle de cas d'utilisation de parent

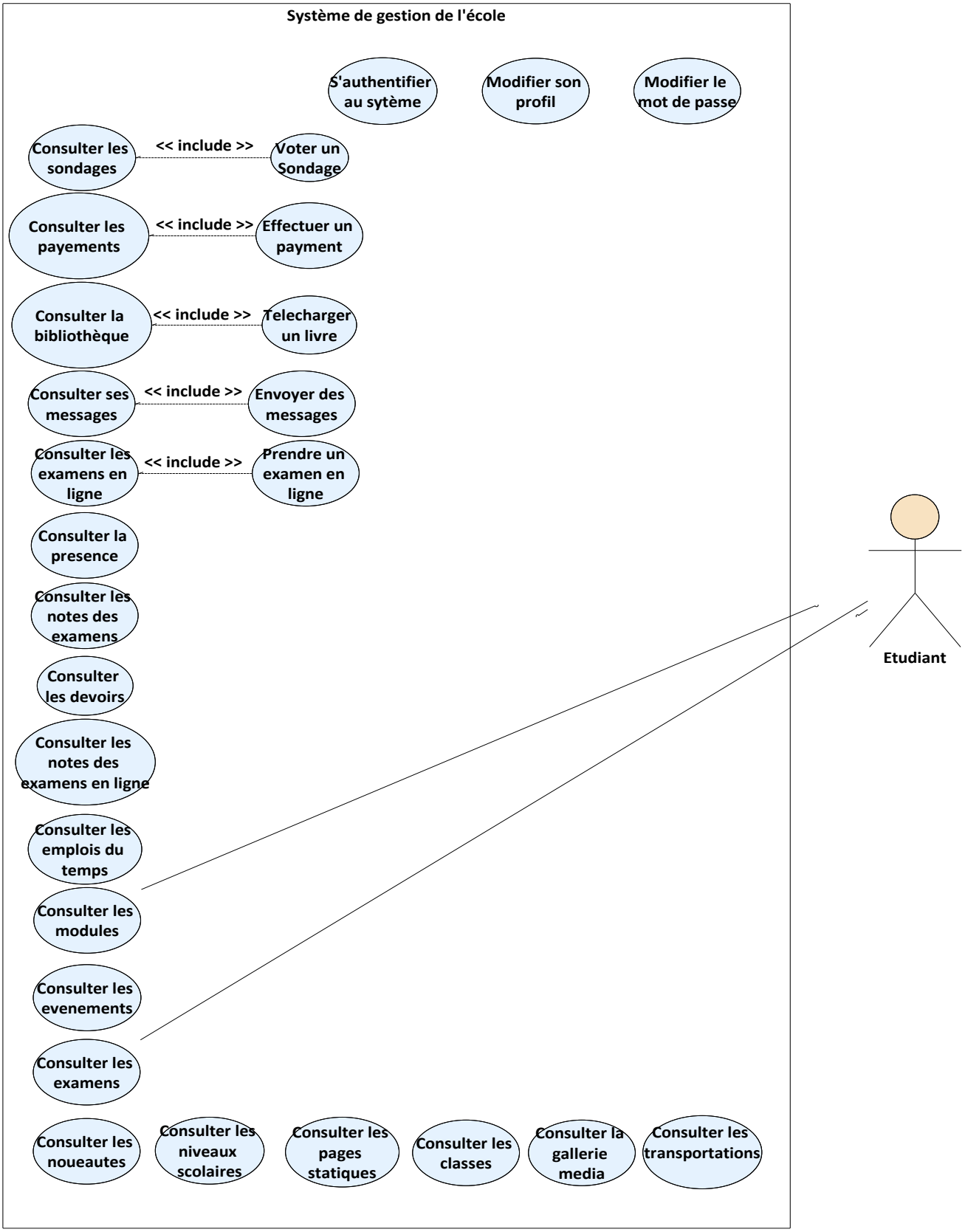


Figure 5-3 Diagramme du modèle de cas d'utilisation d'Etudiant

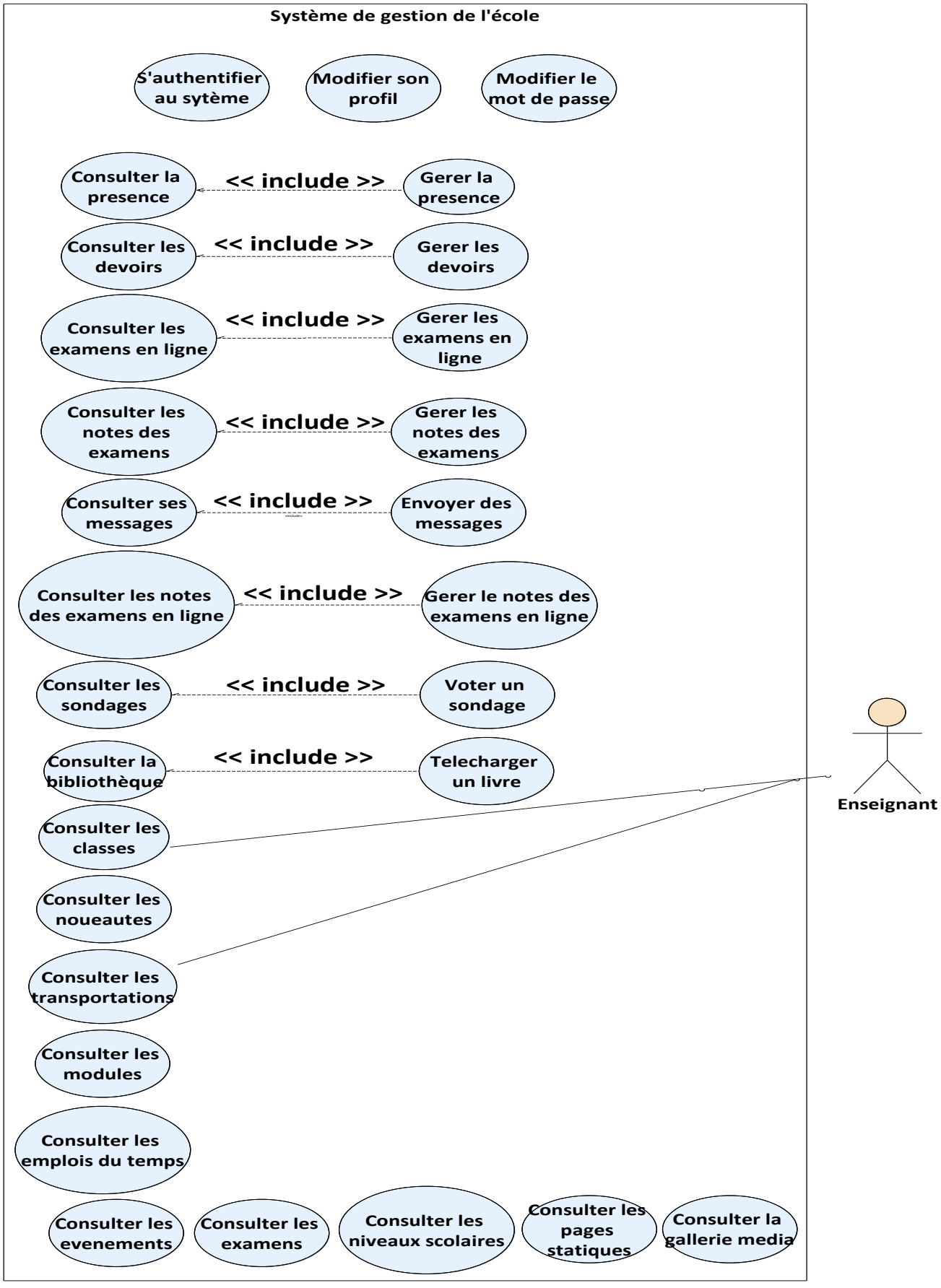


Figure 5-4 Diagramme du modèle de cas d'utilisation d'Enseignent

5.4.2Diagramme de séquence

On va utiliser le diagramme de séquence pour représentation graphique des interactions entre les acteurs et le système, Le diagramme de séquence permet de montrer les interactions d'objets dans le cadre d'un scénario d'un Diagramme des cas d'utilisation. Dans un souci de simplification.

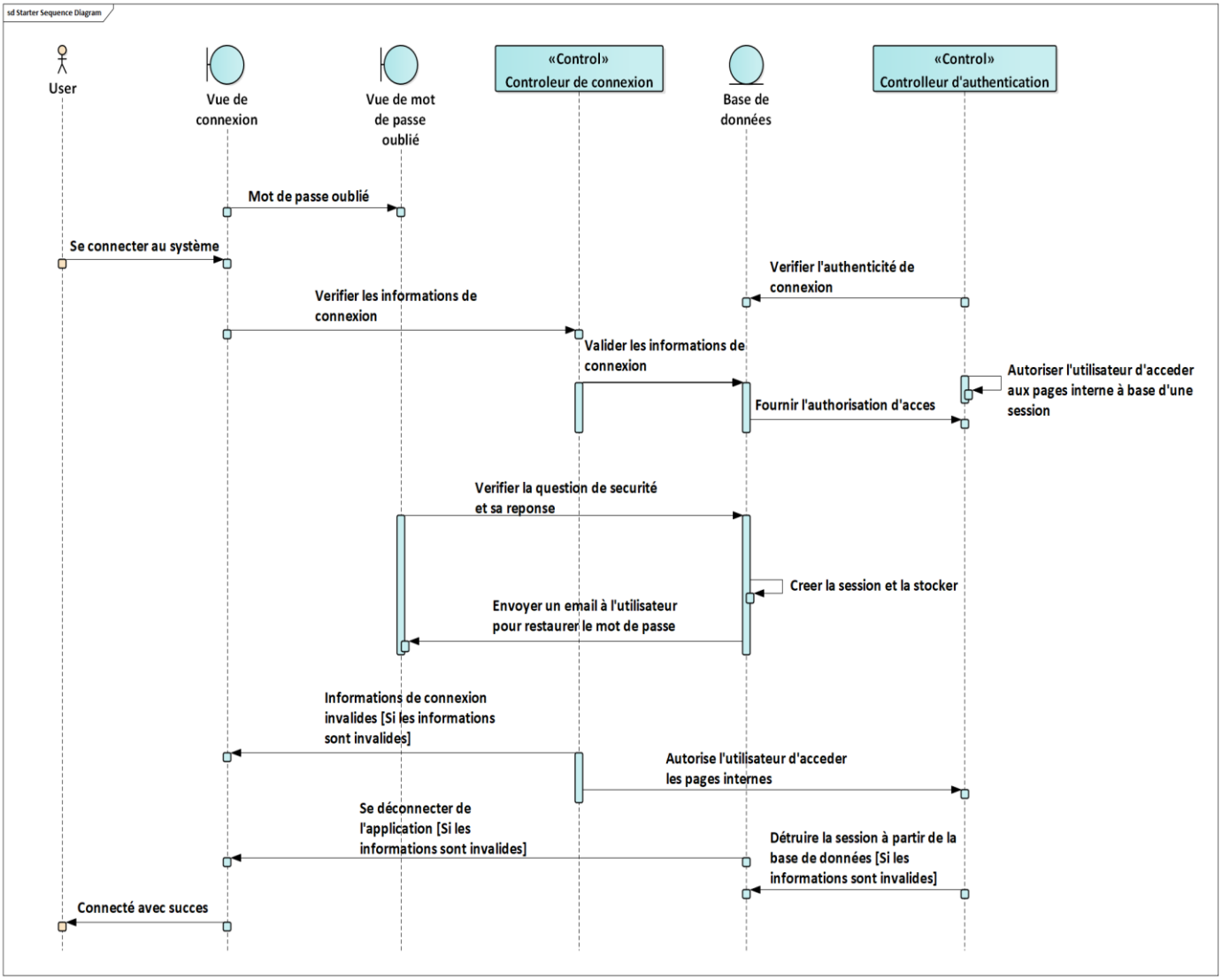


Figure 5-5 Diagramme de séquence qui montre le scénario de LOGIN

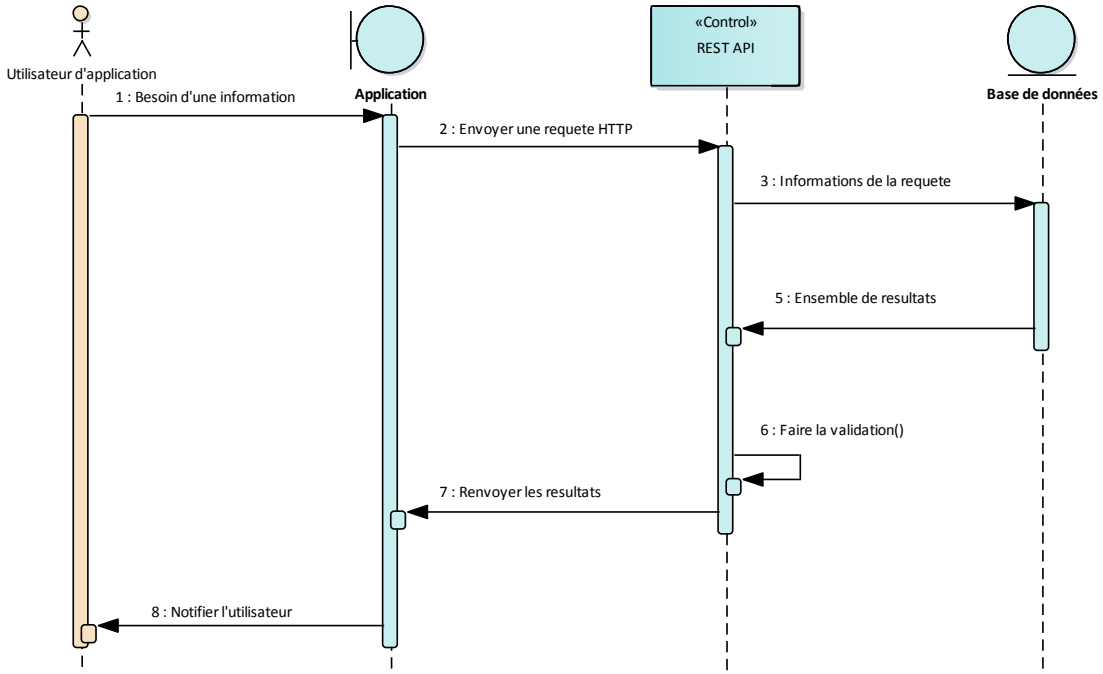


Figure 5-6 : Diagramme de séquence qui montre le scénario d'une requête http

5.5 Elaboration de l'API RESTful

5.5.1Création et Intégration dans le système

Notre API est représentée par un service WEB RESTful, ce qui est un composant logiciel qui l'on nous avons créé et intégré dans le code source du système d'information.

C'est un ensemble de classes PHP qui vont être des (End Points) est qui sont formés en suivant le modèle d'architecture REST pour recevoir des requêtes HTTP et renvoyer des réponses contenant des résultats selon le données envoyées dans le corps de la requête.

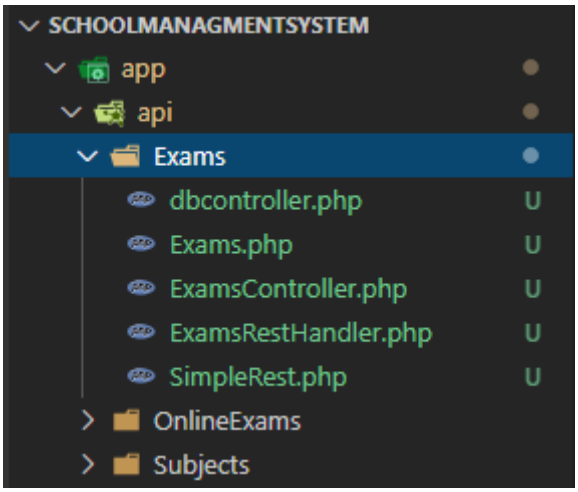


Figure 5-7 Structure de l'API

Le contrôleur de la base de données

C'est une classe PHP qui assure la connexion entre l'API et le système d'information, l'instance de cette classe contient les informations de cette connexion et des méthodes pour interroger la base de données.

```php
<?php
class DBController {
    private $conn;
    private $host;
    private $user;
    private $password;
    private $database;

    function __construct() {
        $url = parse_url(getenv("JAWSDB_URL"));
        $this->host = $url["host"];
        $this->user = $url["user"];
        $this->password = $url["pass"];
        $this->database = substr($url["path"], 1);

        $conn = $this->connectDB();
        if(!empty($conn)) {
            $this->conn = $conn;
        }
    }

    function connectDB() {
        $conn = mysqli_connect($this->host,$this->user,$this->password,$this->database);
        return $conn;
```

```
    }

    function executeSelectQuery($query) {
        $result = mysqli_query($this->conn,$query);
        while($row=mysqli_fetch_assoc($result)) {
            $resultset[] = $row;
        }
        if(!empty($resultset))
            return $resultset;
    }

    function executeInsertQuery($query) {
    $result="";
    if (mysqli_query($this->conn, $query)) {
               $result= "New record created successfully";
            } else {
               $result= "Error: " . mysqli_error($this->conn);
            }

            return $result;
        }

        function executeUpdateQuery($query) {
            $result="";
            if (mysqli_query($this->conn, $query)) {
                       $result= "Record(s) updated successfully";
                    } else {
                       $result= "Error: " . mysqli_error($this->conn);
                    }

                    return $result;
                }

        function executeDeleteQuery($query) {
            $result="";
            if (mysqli_query($this->conn, $query)) {
                       $result= "Record(s) deleted successfully";
                    } else {
                       $result= "Error: " . mysqli_error($this->conn);
                    }

                    return $result;
                }


}
?>
```

Le contrôleur des informations (Contrôleur des examens comme exemple)

C'est une classe PHP qui va être un End Point des requêtes HTTP, l'instance de cette classe reçoit les données envoyées dans le corps de la requête HTTP demandée selon ses données, ensuite elle crée une instance de Rest Handler qui effectue une recherche des résultats a renvoyer.

```php
<?php
require_once("ExamsRestHandler.php");
require_once("Exams.php");


    $json = file_get_contents('php://input');
    $obj = json_decode($json,true);
   $view = $obj['view'];

    switch($view){

        case "all":
            $examsRestHandler = new ExamsRestHandler();
            $examsRestHandler->getAllExams();
            break;
        }

        ?>
```

La classe RestHandler (RestHandler des examens comme exemple)

C'est une classe instanciée par le contrôleur des informations, elle contient un ensemble de méthodes pour qui effectuent des recherches sur les informations demandées (en créant une instance de la classe SQL avec les paramètres de recherche) et les renvoies comme résultats dans le corps du message de réponse des requêtes HTTP demandées.

```php
<?php
require_once("SimpleRest.php");
require_once("Exams.php");

class ExamsRestHandler extends SimpleRest {

    function getAllExams() {

        $exams = new Exams();
        $rawData = $exams->getAllExams();

        if(empty($rawData)) {
```

```
            $statusCode = 404;
            $rawData = array('error' => 'No exams found!');
        } else {
            $statusCode = 200;
        }

        $requestContentType = 'application/json';
        $this ->setHttpHeaders($requestContentType, $statusCode);

        $result["exams"] = $rawData;

        if(strpos($requestContentType,'application/json') !== false){
            $response = $this->encodeJson($result);
            echo $response;
        }
    }

    public function encodeJson($responseData) {
        $jsonResponse = json_encode($responseData);
        return $jsonResponse;
    }

}
?>
```

La classe SQL (classe SQL des examens comme exemple)

C'est une classe qui contient un ensemble de méthodes qui interrogent la base de données directement en effectuant des requêtes SQL selon les paramètres de recherche (en créant une instance de contrôleur de base de données) et renvoyer ses résultats a l'instance de RestHandler.

```
<?php
require_once("dbcontroller.php");

Class Exams {
    private $exams = array();

    public function getAllExams(){
        $query = "SELECT * FROM examslist order by id desc";
        $dbcontroller = new DBController();
        $this->exams = $dbcontroller->executeSelectQuery($query);
        return $this->exams;
    }

}
?>
```

La classe SimpleRest

C'est une classe qui contient des informations sur le protocole HTTP pour les utiliser, comme la version du HTTP, et l'ensemble des Status Codes et leurs messages.

```php
<?php
class SimpleRest {

    private $httpVersion = "HTTP/1.1";

    public function setHttpHeaders($contentType, $statusCode){

        $statusMessage = $this -> getHttpStatusMessage($statusCode);

        header($this-
>httpVersion. " ". $statusCode ." ". $statusMessage);
        header("Content-Type:". $contentType);
    }

    public function getHttpStatusMessage($statusCode){
        $httpStatus = array(
            100 => 'Continue',
            101 => 'Switching Protocols',
            200 => 'OK',
            201 => 'Created',
            202 => 'Accepted',
            203 => 'Non-Authoritative Information',
            204 => 'No Content',
            205 => 'Reset Content',
            206 => 'Partial Content',
            300 => 'Multiple Choices',
            301 => 'Moved Permanently',
            302 => 'Found',
            303 => 'See Other',
            304 => 'Not Modified',
            305 => 'Use Proxy',
            306 => '(Unused)',
            307 => 'Temporary Redirect',
            400 => 'Bad Request',
            401 => 'Unauthorized',
            402 => 'Payment Required',
            403 => 'Forbidden',
            404 => 'Not Found',
            405 => 'Method Not Allowed',
            406 => 'Not Acceptable',
            407 => 'Proxy Authentication Required',
            408 => 'Request Timeout',
            409 => 'Conflict',
            410 => 'Gone',
```

```
            411 => 'Length Required',
            412 => 'Precondition Failed',
            413 => 'Request Entity Too Large',
            414 => 'Request-URI Too Long',
            415 => 'Unsupported Media Type',
            416 => 'Requested Range Not Satisfiable',
            417 => 'Expectation Failed',
            500 => 'Internal Server Error',
            501 => 'Not Implemented',
            502 => 'Bad Gateway',
            503 => 'Service Unavailable',
            504 => 'Gateway Timeout',
            505 => 'HTTP Version Not Supported');
        return ($httpStatus[$statusCode]) ? $httpStatus[$statusCode] : $status
[500];
    }
}
?>
```

Représentation JSON de l'objet renvoyé par l'API (l'objet examen comme exemple)

{"exams": Array [

Object {

"examDate": "07/01/2021",

"examDescription": "Spring Session (From January To July)",

"examTitle": "Second Semester",

"id": "5",

},

Object {

"examDate": "12/15/2020",

"examDescription": "Fall Session (From September To December)",

"examTitle": "First Semester",

"id": "4",

},],}

Figure 5-8 Vue Correspondante à l'objet JSON renvoyé par l'API

5.5.2Les classes d'application communicant ave l'API

5.5.2.1 Recevoir les données

La classe fetch (fetchExams comme exemple) est une classe qui contient des méthodes qui envoient des requêtes HTTP à l'API pour demander des données et les renvoyer a la vue concernée par l'affichage de ces données.

```
import {BASE_URL} from '../../../utils/config'; // <---  Chemin de l'API

export async function fetchExams() {
  try {
    const response = await fetch(BASE_URL + '/ExamsController.php', {
      method: 'post',
      header: {
        Accept: 'application/json',
        'Content-type': 'application/json',
      },
      body: JSON.stringify({
        view: 'all',
      }),
    });
    const responseJson = await response.json();
    if (response.ok) {
      console.log(responseJson);
      return responseJson['exams'];
    }
  } catch (error) {
    alert(error);
  }
}
```

5.5.2.2 Afficher les données :

La classe d'affichage est une vue d'applications qui affiche les données renvoyés par l'API par des composants graphiques (UI Components).

```
import React, {useState} from 'react';
import {
  Text,
  ScrollView,
  View,
  FlatList,
  StyleSheet,
  Image,
```

```
  SafeAreaView,
} from 'react-native';
import {Header, Left, Right, Icon} from 'native-base';
import {fetchExams} from '../api/fetchExams'; //  <----   Classe fetch exams
import {mainTheme} from '../../../utils/config';

export default function Exams(props) {
  const [exams, setExams] = useState([]);

  const getExams = async () => {
    const res = await fetchExams();
    if (res) {
      setExams(res);
    }
  };

  const start = () => {
    getExams();
  };

  React.useEffect(() => {
    start();
  }, []);
  return (
    <SafeAreaView style={{flex: 1}}>
      <Header
        style={{
          backgroundColor: mainTheme.primaryColor,
          flexDirection: 'row',
        }}>
        <Left style={{width: '15%'}}>
          <Icon
            name="menu"
            type="MaterialIcons"
            style={{color: mainTheme.secondaryColor}}
            onPress={() => {
              props.properties.navigation.openDrawer();
            }}
          />
        </Left>
        <Text
          style={{
            width: '70%',
            textAlignVertical: 'center',
            textAlign: 'center',
            alignSelf: 'center',
            fontSize: 25,
            fontWeight: 'bold',
            color: mainTheme.secondaryColor,
```

```
          }}>
          Exams
        </Text>
        <Right style={{width: '15%'}}>
          <Icon
            style={{color: mainTheme.secondaryColor}}
            name="refresh"
            type="MaterialIcons"
            onPress={() => {
              start();
            }}
          />
        </Right>
      </Header>
      {exams.length > 0 ? (
        <ScrollView
          showsVerticalScrollIndicator={false}
          style={{flex: 1, marginVertical: 10, marginHorizontal: 5}}>
          <FlatList
            data={exams}
            renderItem={({item}) => (
              <View style={styles.container}>
                <Text style={[styles.title, {color: mainTheme.secondaryColor}]
}>

                  {item.examTitle}
                </Text>
                <Text
                  style={[styles.subTitle, {color: mainTheme.secondaryColor}]}
>

                  Description :{' '}
                </Text>
                <Text style={[styles.text, {color: mainTheme.secondaryColor}]}
>

                  {item.examDescription}
                </Text>
                <View style={{flexDirection: 'row'}}>
                  <Image
                    style={{height: 60, width: 60, marginRight: 10}}
                    source={require('../../../img/calendar.png')}
                  />
                  <View>
                    <Text
                      style={[
                        styles.subTitle,
                        {color: mainTheme.secondaryColor},
                      ]}>
                      Date :
                    </Text>
                    <Text
```

```
                    style={[styles.text, {color: mainTheme.secondaryColor}]}
>
                    {item.examDate}
                  </Text>
                </View>
              </View>
            </View>
          )}
        />
      </ScrollView>
    ) : (
      <View
        style={{
          flex: 1,
          alignItems: 'center',
          justifyContent: 'center',
        }}>
        <Image
          style={{
            height: 50,
            width: 50,
            marginVertical: 10,
            tintColor: mainTheme.primaryColor,
          }}
          source={require('../../../img/exams.png')}
        />
        <Text
          allowFontScaling={true}
          style={{
            fontSize: 20,
            fontWeight: 'bold',
            color: mainTheme.primaryColor,
          }}>
          No Exams To Show !
        </Text>
      </View>
    )}
  </SafeAreaView>
 );
}
const styles = StyleSheet.create({
 container: {
  borderColor: mainTheme.primaryColor,
  backgroundColor: mainTheme.primaryColor,
  borderWidth: 3,
  borderRadius: 30,
  margin: 10,
  padding: 10,
 },
```

```
  title: {
    alignSelf: 'center',
    fontSize: 25,
    fontWeight: 'bold',
    marginVertical: 5,
  },
  subTitle: {
    fontSize: 20,
    fontWeight: 'bold',
    marginVertical: 3,
  },
  text: {
    fontSize: 15,
    fontWeight: 'bold',
    textAlignVertical: 'center',
  },
});
```

5.5.3Interface

5.5.3.1 Connexion / Login

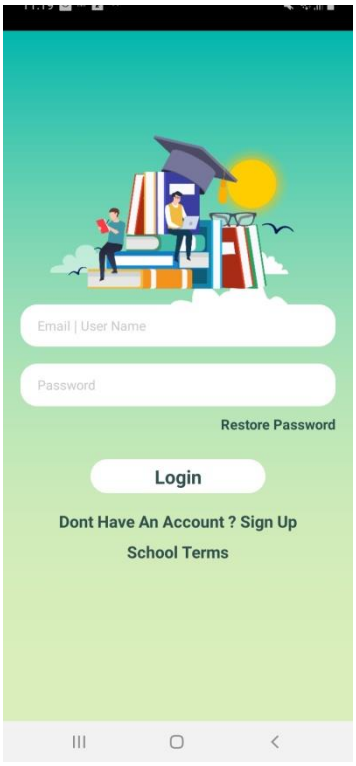


Figure 5-9 : Interface Connexion/Login

5.5.3.2 Inscription/Sign UP

L'application School management propose des systèmes d'enregistrement pour que les enseignants, les élèves et les parents s'inscrivent eux-mêmes dans le système et l'administrateur approuve leurs comptes au lieu de les ajouter manuellement.

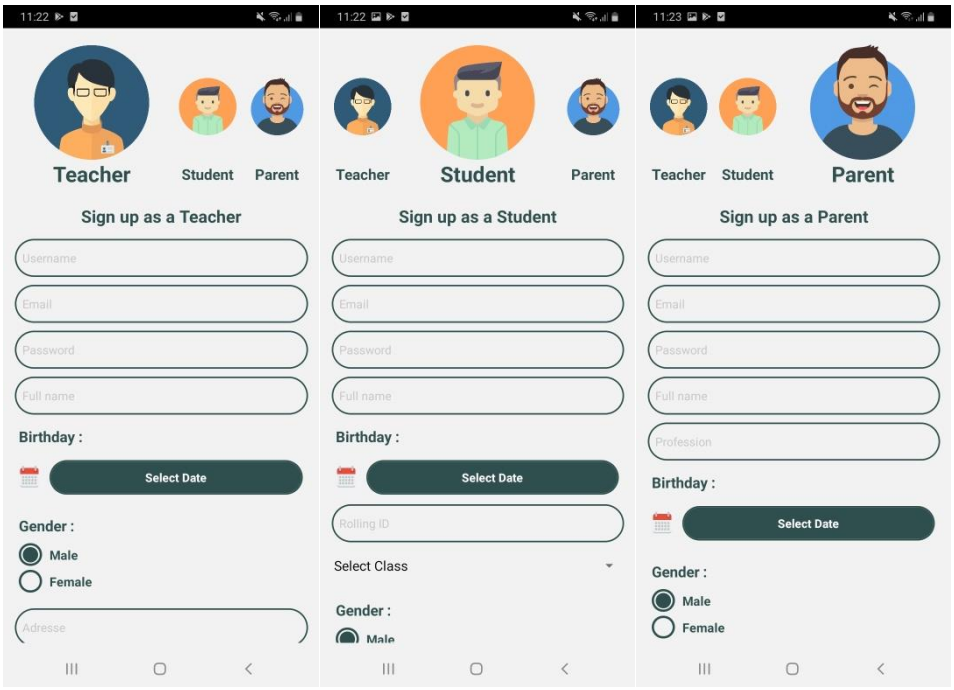


Figure 5-10 : Interface inscription/Sign UP

Remarque : Apprêt L'inscription, l'administrateur peut approuver ou supprimer les comptes nouvellement enregistrés dans (enseignants, élèves ou parents) et cliquer sur le bouton En attente d'approbation pour afficher la liste des utilisateurs en attente.

5.5.3.3 Modifie ou réinitialiser le mot de passe / Restore Password

Notre application School Management propose aux ces utilisateur de pouvoir changé lors mot de passe on tout sécurité

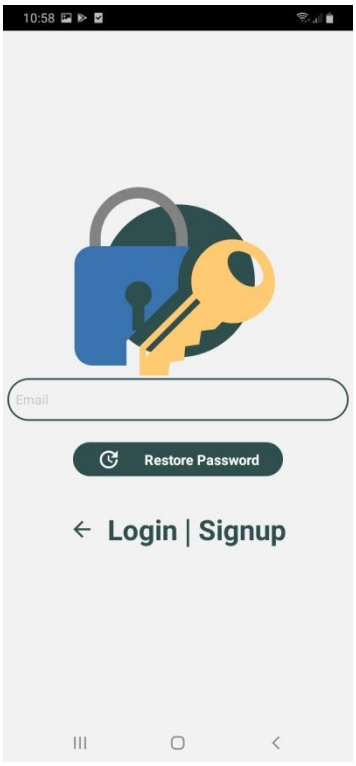


Figure 5-11 : Interface Modifie ou réinitialiser le mot de passe / Restore password

5.5.3.4 Exemple d'une connexion d'un étudiant

On va voir un exemple qui montre la connexion d'un étudiant qui à Username : Betsy_J

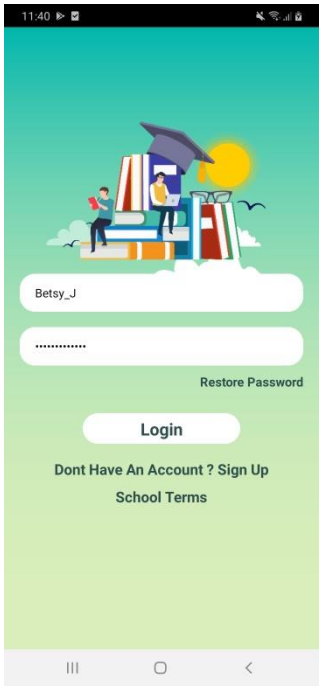


Figure 5-12 : Interface qui montre le login d'un étudiant

5.5.3.4.1 Page d'accueil/Home

Dans la page d'accueil en trouve un menu en bas (Bottom menu) qui permet d'accéder à 3 pages : Leaderboard, Polls, News and Events.

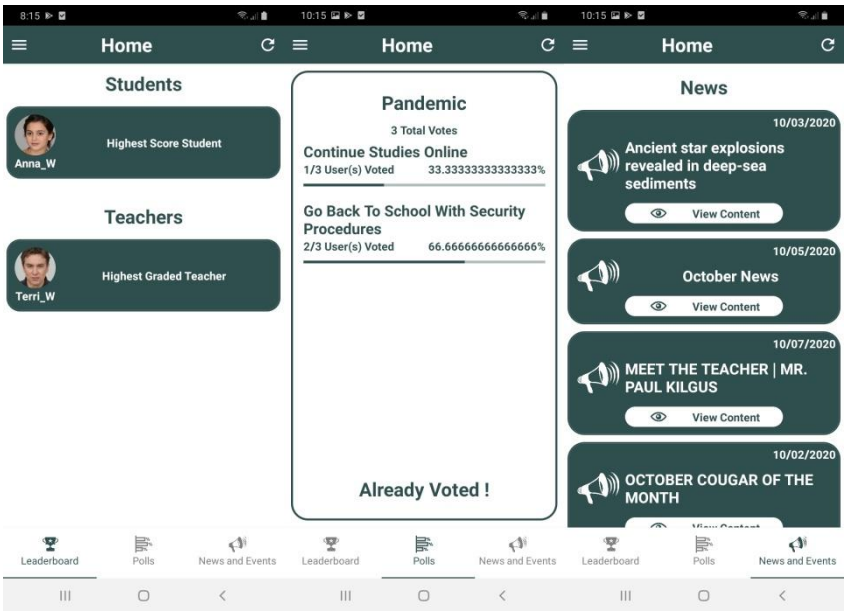


Figure 5-13 : interface qui montre la page d'accueil/Home

5.5.3.4.2 Menu d'application (Menu burger)

Pour afficher le menu de l'application, cliquez simplement sur l'icône de menu en haut à droite ou faites glisser vers la droite. Dans ce menu, il y a une photo de l'utilisateur avec son nom d'utilisateur et toutes les pages auxquelles il peut accéder et une icône d'arrêt rouge à droite pour déconnecter.

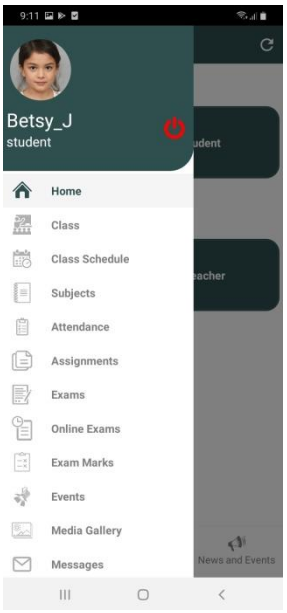


Figure 5-14 : L'interface qui montre le Menu d'application

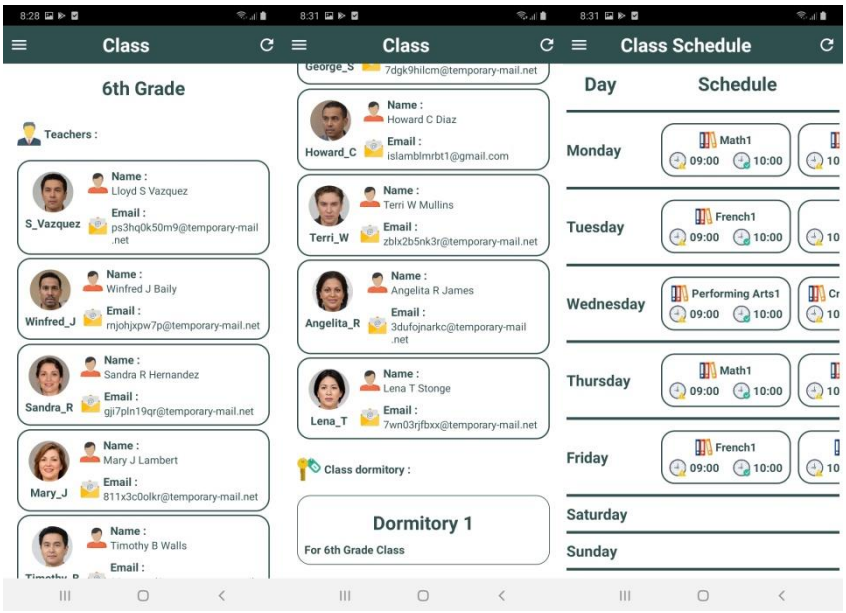


Figure 5-15 : Les interfaces qui montrent la page Classe et la page des horaires de cours

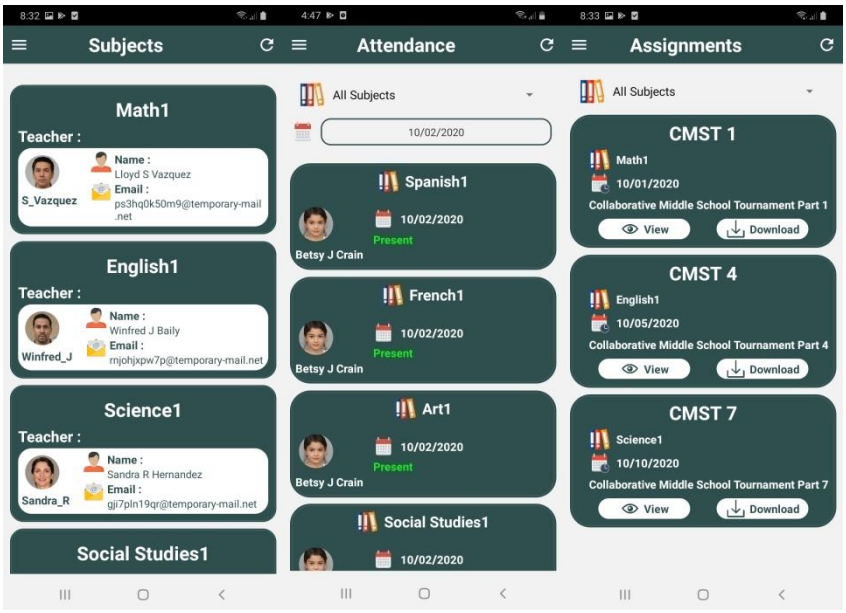


Figure 5-16 : Les interfaces qui montrent la page des Matières, Présence et Affectation

Figure 5-17 : L'interface qui montrent la page Examens déjà passé

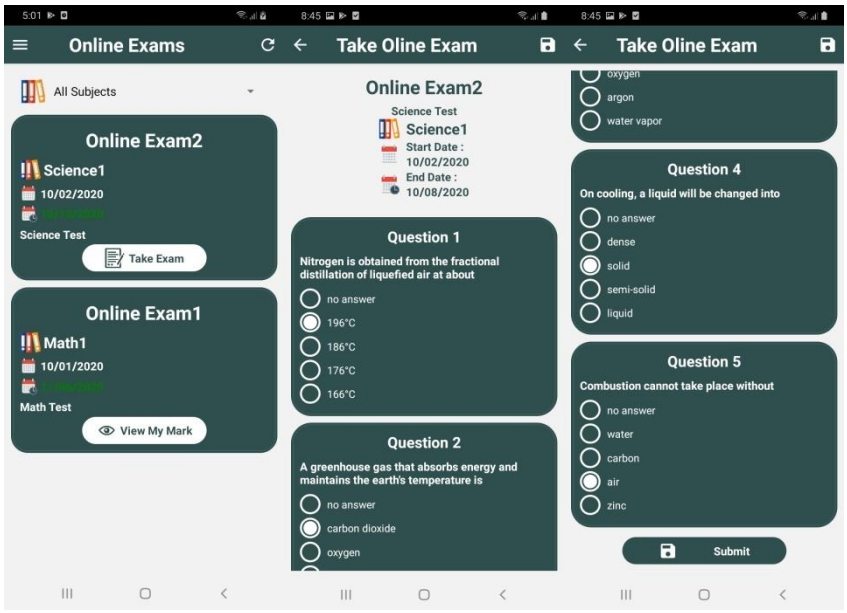


Figure 5-18 : Les interfaces qui montrent la page des examens en ligne et un exemple d'un examen

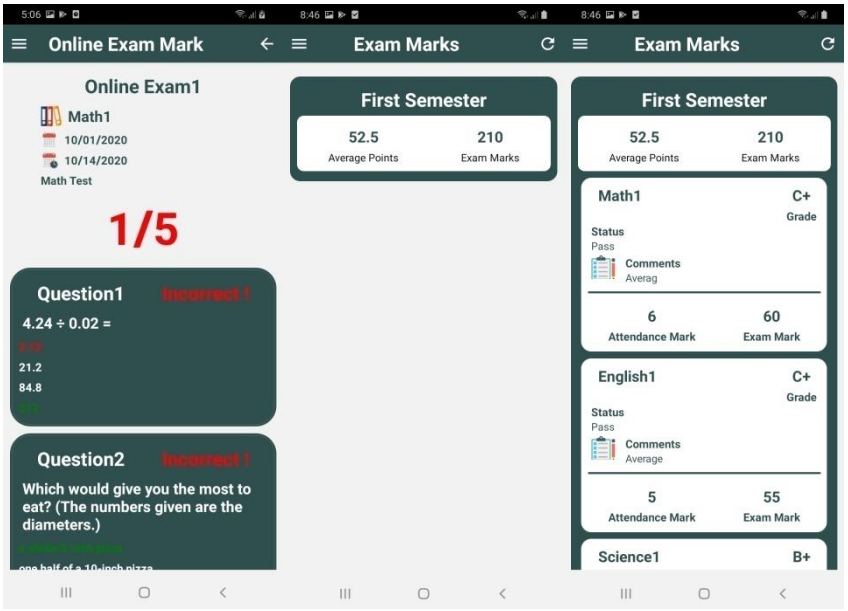


Figure 5-19 : Les interfaces qui montrons les pages des notes Examens on ligne et examens

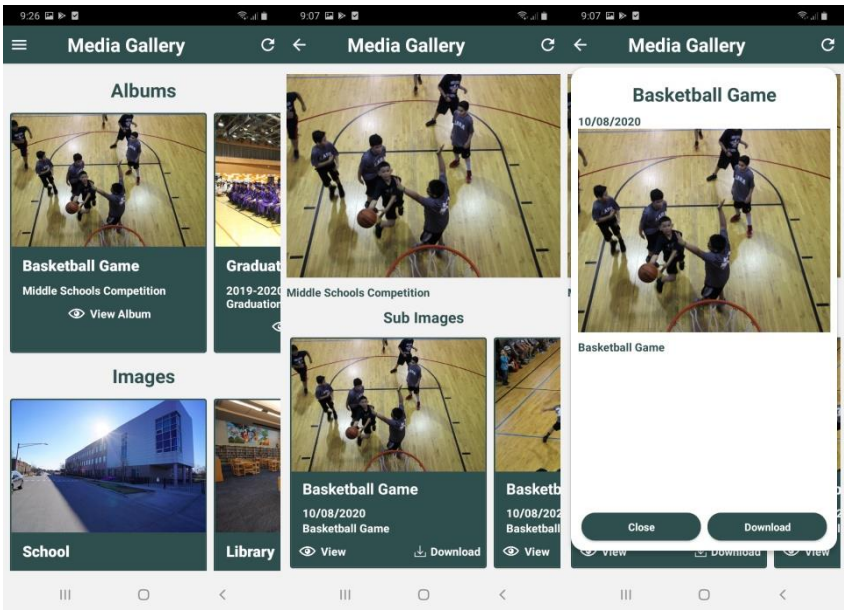
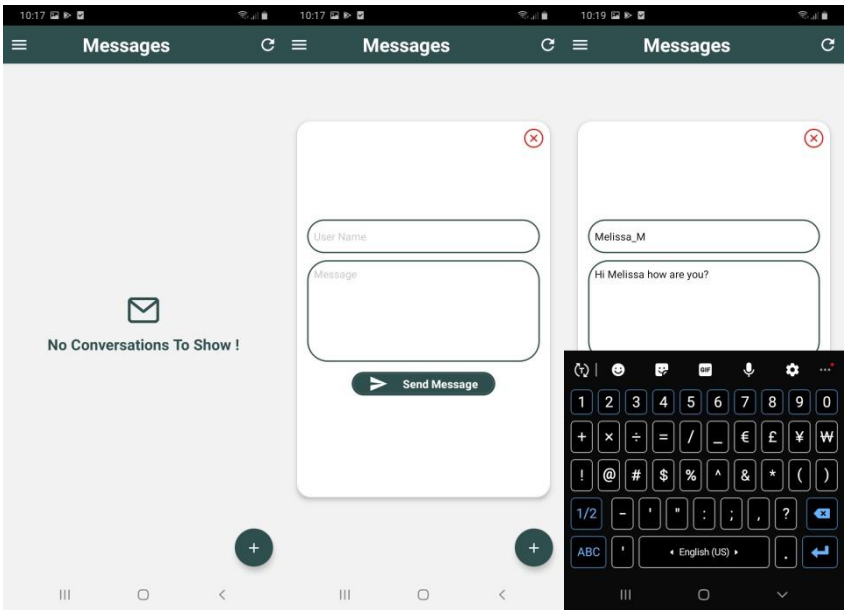


Figure 5-20 : L'interface qui montre Galerie des médias



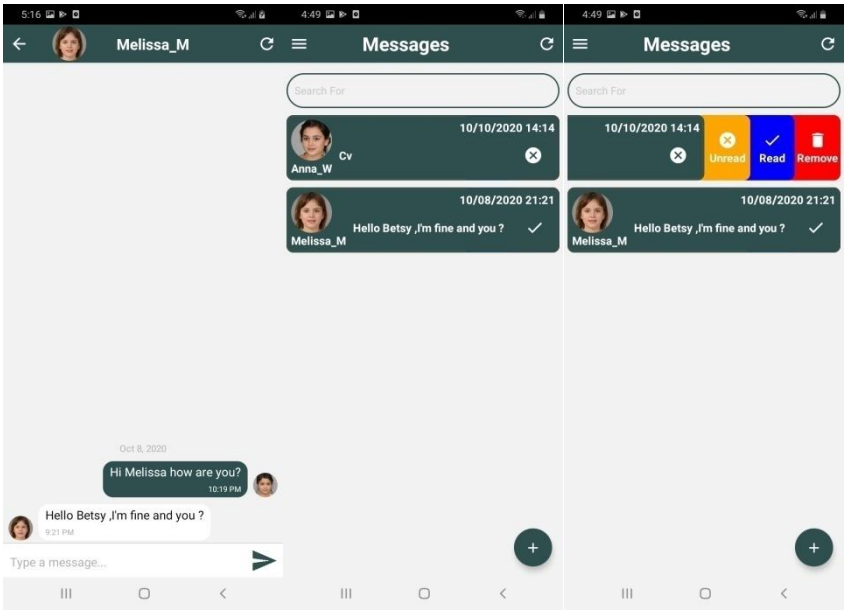


Figure 5-21 : Les interfaces montrent la page de Message avec un exemple d'une discussion

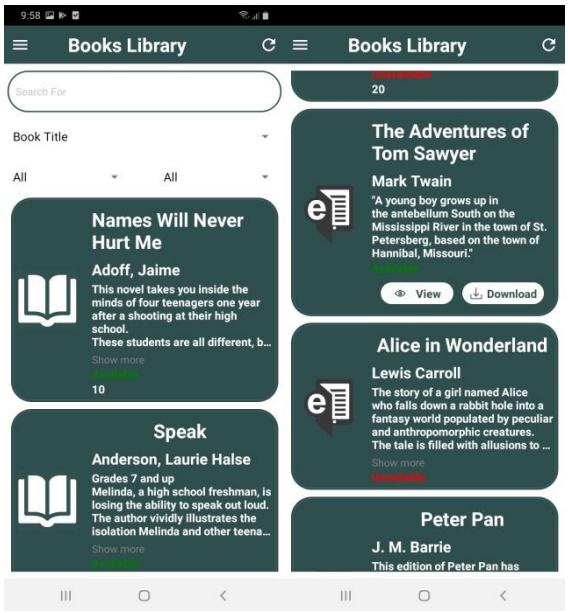


Figure 5-22 : L'interface qui montre la page de bibliothèque avec deux choix, livre traditionnel et livre électronique

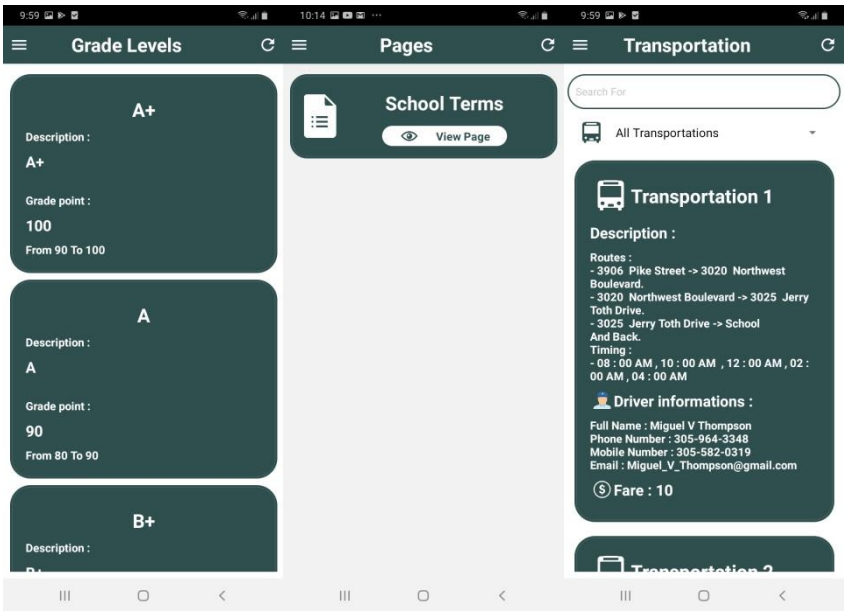


Figure 5-23 : Interfaces qui affichent la page de niveau scolaire, la page statique et la page de transport

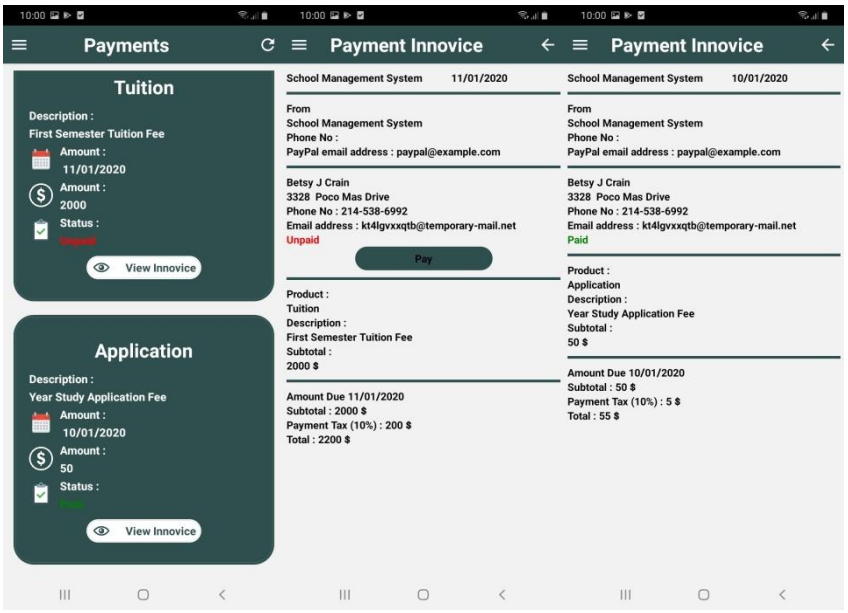


Figure 5-24 : L'interface qui montre la page de Paiement

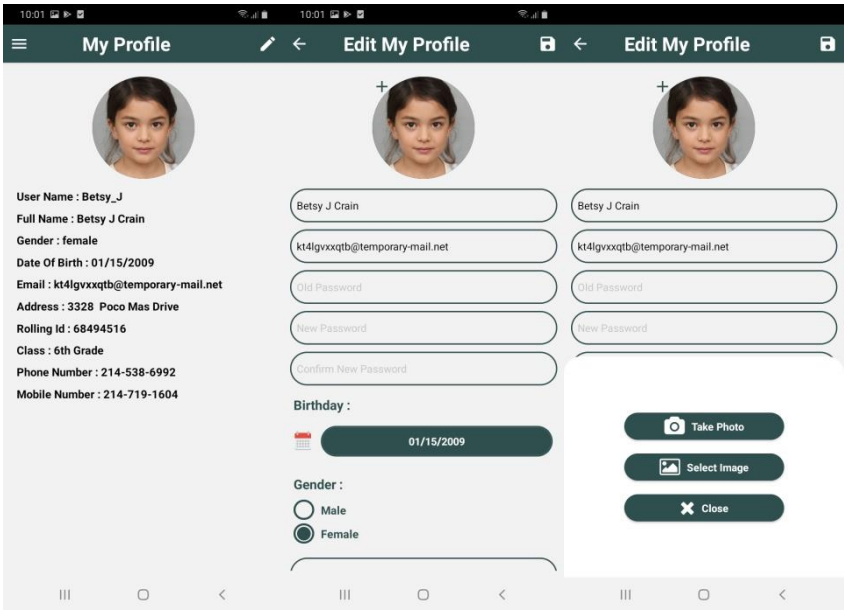


Figure 5-25 : Les interfaces qui montrons la page de profil et éditer le profil

Conclusion

Dans ce chapitre, nous avons présenté les différents outils et technologies de développement utilisés pour la réalisation de notre application mobile. Nous avons vu les diagrammes UML, le diagramme de cas d'utilisation, diagramme de classe et le diagramme de séquence de quelques scénarios pour montrer la structure de system, On aaussi présenté les modules et les rôles de chaque utilisateurs dans application,Enfin nous avonsprésenté quelques interfaces d'applications.

Conclusion général

Le but de notre travail est de réaliser une application mobile d'un système d'information qui existe déjà. Dans ce mémoire on a vu l'importance des systèmes d'information etdu génie logiciel et UML, ensuite on a parlé d'environnementmobile et développement des applications avec les types des applications mobiles. ons'est consacré pour les Framework mobile multiplateforme, ensuite on a parlé des service web et le RESTful API.Nous avons également vu les différents outils et technologies de développement utilisés pour la réalisation de notre application mobile et une démonstration de la structure de system avec des diagramme UML et quelques interfaces d'applications.

Le travail consistait à créer une application pour un système d'information.Pour cela on a choisi un système de gestion d'une école (School Management), l'objectif était d'aider les enseignaient, les étudiant, et leur parents à bien suivre leur tâche d'une façon mobile avec leur Smartphone ou Tablet.

L'application mobile quant à elle permet de faire des inscriptions en tant qu'enseignant ou élève ou parent d'un élève existant.Aprèsinscription, il est nécessaire d'attendre l'approbation de l'administrateur ; après l'approbation chaque utilisateur peut se connecter et accéder au serveur par l'application.

Concernant le coté implémentation nous avons utilisé des technologies modernes et récentes telles que : React Native, Expo et Git, etc. Et pour le service web, on a choisi l'architecture REST pour cela on a fait uneRESTful API qui se base sur le protocole http.

Bibliographie

1. School Management System PHP MYSQL Source Codes. *https://iwantsourcecodes.com/.* [En ligne] [Citation : 1 avril 2020.] https://iwantsourcecodes.com/school-management-system-php-mysql-source-codes/?fbclid=IwAR0KlPqpUbBo26kfcnyjTB1bXU4tqc7EwP5__l-i_SFM05J5rqGxOJGsxqU.

2. **Delaye, David Autissier et Valérie.***Mesurer la performance du système d'information (Vol. 3).* s.l. : Editions Eyrolles, 2008.

3. **Dahbia, CHIBOUTI Fadila & CHERFA.***Système d'Information et son rôle au sein de l ?Entreprise Cas pratique : les moulins de la Soummam.* bejaia : s.n., 2015. thése.

4. **Silva, Antonio Lopes Da.***L'information et l'entreprise: des savoirs à partager et à capitaliser méthodes , outils et application à la veille.* Marseille : s.n., 2002. thése.

5. **Quarterly, McKinsey.** Les systèmes d'information au défi de la mobilité. *http://parisinnovationreview.com/.* [En ligne] 22 octobre 2012. [Citation : 11 juin 2020.] http://parisinnovationreview.com/article/les-systemes-dinformation-au-defi-de-la-mobilite.

6. **Audibert, Laurent.***UML 2: de l'apprentissage à la pratique.* s.l. : Ellipses, 2009.

7. **Thomas, Collonvillé.** Les processus de développement ou cycles de vie du logiciel. *un-est-tout-et-tout-est-un.blogspot.com.* [En ligne] 23 Décembre 2017. [Citation : 11 juin 2020.] https://un-est-tout-et-tout-est-un.blogspot.com/2017/12/les-processus-de-developpement-ou.html?m=0.

8. *Prototype Model.*

9. *Figure Modèle Prototypage.* 30 10 2020.

10. *RAD Model.*

11. *Figure Modèle RAD.* 30 10 2020.

12. *Agilité.*

13. **Messenger, Steve.***The DSDM Agile Project Framework V2.* 2014.

14. **Hibbs, C., Jewett, S., & William, M.***L'art du Lean : Software development.* Paris : s.n., 2010.

15. **Aubry, C.***Scrum : Le guide pratique la Méthode Agile la plus populaire.* Paris : s.n., 2014.

16. Smartphone. *wikipedia.org.* [En ligne] [Citation : 20 août 2020.] https://fr.wikipedia.org/wiki/Smartphone.

17. **Per, Christensson.** Android. *techterms.com.* [En ligne] 16 mai 2016. [Citation : 24 mai 2020.] https://techterms.com/definition/android.

18. —. ios. *techterms.com.* [En ligne] 22 octobre 2011 2011. [Citation : 24 mai 2020.] https://techterms.com/definition/ios.

19. *statcounter.com.* [En ligne] [Citation : 20 août 2020.] https://gs.statcounter.com/os-market-share/mobile-tablet/worldwide/#monthly-201507-202007-bar.

20. **Bathelot, Bertrand.** Application mobile. *www.definitions-marketing.com.* [En ligne] 08 mars 2017. [Citation : 24 mai 2020.] https://www.definitions-marketing.com/definition/application-mobile/.

21. **Mane, GHERARI Mlle.***Contribution à L'évolution des Architectures Logicielles des Systèmes Intensifs.* Tebessa : s.n. Thèse de doctorat.

22. **LB, MAXIME.** Quel Framework pour développer une application mobile en 2020 ? *www.maximelb.com.* [En ligne] 23 DÉCEMBRE 2019. [Citation : 24 mai 2020.] https://www.maximelb.com/quel-framework-pour-developper-une-application-mobile-en-2020/#:~:text=Framework%20cr%C3%A9%C3%A9%20par%20Facebook%2C%20React,langage%20souple%20et%20rapidement%20assimilable..

23. **profexorgeek.** what is xamarin. *www.microsoft.com.* [En ligne] 28 mai 2020. [Citation : 15 juin 2020.] https://docs.microsoft.com/en-us/xamarin/get-started/what-is-xamarin.

24. **Sharma, Naiya.** React Native vs Xamarin vs Ionic vs Flutter: quel est le meilleur pour le développement d'applications mobiles multiplateformes? *www.apptunix.com.* [En ligne] 17 septembre 2018. [Citation : 15 juin 2020.] https://www.apptunix.com/blog/frameworks-cross-platform-mobile-app-development/.

25. **Kargerman, Sacha.** applicatio mobile quel framework choisir pour 2019. *www.codixis.com.* [En ligne] 22 décembre 2018. [Citation : 15 juin 2020.] https://www.codixis.com/application-mobile-quel-framework-choisir-pour-2019/.

26. differentes caracteristiques web services partie 1. *https://www.supinfo.com.* [En ligne] [Citation : 15 juin 2020.] https://www.supinfo.com/articles/single/7837-differentes-caracteristiques-web-services-partie-1.

27. **http://biodev.extra.cea.fr.***Architecture d'un Service WEB.*

28. **Guideinformatique.** SOAP, UDDI, WSDL : transport, découverte, description. *https://www.guideinformatique.com.* [En ligne] 23 décember 2002. [Citation : 15 juin 2020.] https://www.guideinformatique.com/dossiers-actualites-informatiques-entreprise-ressources-planning-1/soap-uddi-wsdl-transport-decouverte-description-170.html.

29. **Dedsec, Gnf.** The difference between the Web Service - API, WSDL, SOAP, REST, Simply that the farmers understand. *https://dev.to.* [En ligne] 18 avril 2020. [Citation : 20

juillet 2020.] https://dev.to/mrgreenfz/the-difference-between-the-web-service-api-wsdl-soap-rest-simple-that-the-farmers-understand-7k4.

30. Type de service Web – SOAP et REST. *https://waytolearnx.com.* [En ligne] 26 juillet 2019. [Citation : 20 juillet 2020.] https://waytolearnx.com/2019/07/type-de-service-web-soap-et-rest.html.

31. **Vincy.** PHP RESTful Web Service API – Part 1 – Introduction with Step-by-step Example. *https://phppot.com/.* [En ligne] 19 février 2019. [Citation : 20 juillet 2020.] https://phppot.com/php/php-restful-web-service/?fbclid=IwAR3qoHUGclVkvSYgsC81gIgb8-G45bL_9d8BqG0nODW5YjyN1Gj3B3GtavU#restful-web-services-api-architecture.

32. *Architecture REST.* 30 10 2020.

33. **Rouse, Margaret.** RESTful API (REST API). *https://searchapparchitecture.techtarget.com.* [En ligne] avril 2020. [Citation : 20 juillet 2020.] https://searchapparchitecture.techtarget.com/definition/RESTful-API#:~:text=A%20RESTful%20API%20is%20an,to%20communicate%20with%20each%20other..

34. http methods. *https://restfulapi.net.* [En ligne] [Citation : 20 août 2020.] https://restfulapi.net/http-methods.

35. Visual Studio Code. *https://en.wikipedia.org.* [En ligne] [Citation : 6 mai 2020.] https://en.wikipedia.org/wiki/Visual_Studio_Code.

36. XAMPP. *https://fr.wikipedia.org.* [En ligne] [Citation : 6 mai 2020.] https://fr.wikipedia.org/wiki/XAMPP.

37. phpmyadmin. *https://doc.ubuntu-fr.org/.* [En ligne] [Citation : 06 mai 2020.] https://doc.ubuntu-fr.org/phpmyadmin.

38. GitHub. *https://fr.wikipedia.org.* [En ligne] [Citation : 6 mai 2020.] https://fr.wikipedia.org/wiki/GitHub.

39. what. *https://www.heroku.com.* [En ligne] [Citation : 6 mai 2020.] https://www.heroku.com/what.

40. React(JavaScript). *https://fr.wikipedia.org.* [En ligne] [Citation : 6 mai 2020.] https://fr.wikipedia.org/wiki/React_(JavaScript).

41. learning react native. *https://www.oreilly.com.* [En ligne] [Citation : 6 mai 2020.] https://www.oreilly.com/library/view/learning-react-native/9781491929049/ch01.html .

42. Introduction to Expo. *https://expo.io/.* [En ligne] [Citation : 6 mai 2020.] https://docs.expo.io/.

43. MySQL. *https://fr.wikipedia.org.* [En ligne] [Citation : 6 mai 2020.] https://fr.wikipedia.org/wiki/MySQL.

44. Git. *https://fr.wikipedia.org.* [En ligne] [Citation : 6 mai 2020.] https://fr.wikipedia.org/wiki/Git.

45. **Morley, Chantal.***Management d'un projet Système d'Information - 6ème édition: Principes, techniques, mise en oeuvre et outils.* s.l. : Dunod, 2008.

46. Windows Phone. *wikipedia.org.* [En ligne] [Citation : 24 mai 2020.] https://fr.wikipedia.org/wiki/Windows_Phone.

47. BlackBerry OS. *wikipedia.org.* [En ligne] [Citation : 24 mai 2020.] https://fr.wikipedia.org/wiki/BlackBerry_OS.

48. Différence entre SOAP et REST. *https://waytolearnx.com.* [En ligne] 04 avril 2019. [Citation : 20 juillet 2020.] https://waytolearnx.com/2019/04/difference-entre-soap-et-rest.html.