



REPOBLIKAN'I MADAGASIKARA
FITIAVANA - TANINDRAZANA - FANDROSOANA
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR
ET DE LA RECHERCHE SCIENTIFIQUE



ECOLE SUPERIEURE POLYTECHNIQUE D'ANTSIRANANA

Mention STIC

PROJET DE FIN D'ETUDE

Parcours Électronique et Informatique Industrielles

Conception et réalisation du logiciel de gestion de la polyclinique universitaire NEXT

Par :

HAJALALAINA Fara Marie José

Encadreurs :

RAKOTOARISOA Jean Claude, Dr Ingénieur

RAMANAN'HAJA Hery Tina, Ingénieur

Dr Luigi BELLINI, Professeur

RAZAIARIMALALA Nirina Claudia, Chef de service

★ Année Universitaire 2018 - 2019 ★



REPOBLIKAN'I MADAGASIKARA
FITIAVANA - TANINDRAZANA - FANDROSOANA
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR
ET DE LA RECHERCHE SCIENTIFIQUE



ECOLE SUPERIEURE POLYTECHNIQUE D'ANTSIRANANA

Mention STIC

MÉMOIRE DE FIN D'ÉTUDES

Parcours Électronique et Informatique Industrielles

Conception et realisation du logiciel de gestion de la polyclinique universitaire NEXT

Par :

HAJALALAINA Fara Marie José

Encadreurs :

Monsieur RAKOTOARISOA Jean Claude

Monsieur RAMANAN'HAJA Hery Tina

Dr Luigi BELLINI, Professeur

Madame RAZAIARIMALALA Nirina Claudia, Chef de service

Membres de jury :

Madame ANDRIAMIHARINJAKA Hasina	Président de jury
Monsieur RAKOTOARISOA Jean Claude	Encadreur
Monsieur RAMANAN'HAJA Hery Tina	Encadreur
Madame	Examineur
Monsieur	Examineur

★ Année Universitaire 2018 - 2019 ★

Promotion :

Projet et Mémoire de fin d'études – A.U : 2018/2019

Titre : Conception et réalisation du logiciel de gestion de la Polyclinique universitaire NEXT

Contexte

Dans le cadre de son programme de partenariat public privé, la polyclinique Universitaire Next s'est mise en accord de partenariat avec l'état Malagasy pour la prise en charge des fonctionnaires d'état, depuis juin 2019. Elle souhaite mettre en place un logiciel permettant de gérer les patients ainsi que les activités au sein de l'hôpital. Parmi ces activités sont compris, la réception des patients, gestion de la facturation, gestion de l'hospitalisation et la gestion de la pharmacie.

Après avoir effectué un état de lieux du logiciel existant, en accord avec l'administration et techniciens, la conception d'un nouveau logiciel s'impose, pour gérer ces fonctionnaires d'état.

Objectif

L'objectif de ce logiciel est de numériser les données, d'automatiser les activités au sein de la Polyclinique.

Travaux demandés

Cas d'utilisation :

Pour l'administration de la polyclinique :

- Elaboration de la statistique des patients (Nombre des patients par date, par service...)
- Facturation des patients (Payement, paiement du reste à payer, annulation, modification,...)
- Gestion des prestations diverses
- Bilan régulier

Pour la réception :

- Enregistrement des patients
- Triage des patients
- Enregistrement des sorties des patients
- Facturation
- Saisie des données médicales des patients

Pour l'administration du logiciel :

- Gestion des utilisateurs
- Gestion des fonctionnalités du logiciel

Encadreur(s)

- RAKOTOARISOA Jean Claude, Dr Ingénieur
- RAMANAN'HAJA Hery Tina, Ingénieur
- Dr Luigi BELLINI, Professeur
- RAZAIARIMALALA Nirina Claudia, Chef de service Administration

0.1 Lieu de travail

- Laboratoire d'Informatique Appliquée et de Mathématique, UNA
- Polyclinique Universitaire Next

0.2 Etudiant réalisateur

HAJALALAINA Fara Marie José

Remerciements

En préambule de ce travail, j'aimerais exprimer, par ces quelques lignes de remerciements, mes gratitudes envers tous ceux en qui, par leur présence, leur soutien, leur disponibilité et leurs conseils j'ai trouvé courage pour réaliser ce travail.

En premier lieu, j'adresse mes respectueuses considérations à mon encadreur professionnel : Dr. RAKOTOARISOA Jean Claude, Monsieur RAMANAN'HAJA Hery Tina qui ont été l'esprit pensant donnant naissance à ce sujet et qui m'a dirigée par ses précieux conseils tout en me laissant la liberté d'initiative pour la conception de la plateforme. A mes encadreurs qui sont Dr Luigi BELLINI, Madame RAZAIARIMALALA Nirina Claudia pour avoir dirigés ce travail, pour leurs soutiens et aussi pour leurs aides et leurs orientations qui m'ont permis de réaliser ce travail dans les meilleures conditions.

J'adresse mes remerciements anticipés et mes honorables considérations également aux membres de Jury qui vont évaluer et porter leur jugement à ce travail, aussi en enrichir le contenu par leurs précieuses propositions.

Je dois reconnaissance également à l'ensemble du corps enseignant de la mention STIC de le ESPA (Ecole Supérieure Polytechnique d'Antsiranana).

Une mention particulière à ma famille, pour leurs soutiens et leurs attentions sans faille, dont les encouragements et l'amour.

Bref, tout ceux qui ont contribué de près ou de loin à l'accomplissement de ce travail.

HAJALALAINA Fara Marie José

Table des matières

Remerciements	i
Table des matières	ii
Liste des figures	iii
Liste des tableaux	iv
Introduction générale	1
Chapitre 1 PRESENTATION DE LA POLYCLINIQUE UNIVERSITAIRE NEXT	2
1.1 Historique	3
1.2 Localisation et information sur le complex hospitalier	3
1.3 Plan hospitalier	4
Chapitre 2 MODELISATION DE L'APPLICATION	5
2.1 Analyse de fonctionnement de l'hôpital	7
2.2 Diagramme de cas d'utilisation	9
2.3 diagramme de séquence	13
2.4 Diagramme de classe de l'application	14
Chapitre 3 Réalisation de l'application	17
3.1 Le framwork Laravel	17
Chapitre 4 Modélisation statique et dynamique du système	18
Chapitre 5 Programmation et réalisation du logiciel	19
5.1 Introduction	20
5.2 Les Systèmes de Gestion de Base de Données (SGBD)	20
5.3 Les langages de programmation	21
5.4 Conception	23
Chapitre 6 Présentation et déploiement de l'application	27
6.1 Pré-requis	28
6.2 Présentation de l'application	28
6.3 Les services backend de Firebase	33
6.4 Déploiement de l'application sur Firebase	33
Conclusion générale	35
Bibliographie	I
Webographie	II

Annexes	III
A Code TypeScript : Authentification	III
B Upload de fichier	IV
C Divers fonction de validation de formulaires	IV

Liste des figures

2.1	Diagramme de cas d'utilisation générale du système	9
2.2	Diagramme de cas d'utilisation réceptionniste	10
2.3	Diagramme de cas d'utilisation administrateur polyclinique	11
2.4	Diagramme de cas d'utilisation administrateur logiciel	12
2.5	Diagramme de séquence gestion patient	15
2.6	Diagramme de classe de l'application	16
6.1	Fenêtre d'authentification	28
6.2	Fenêtre d'inscription	29
6.3	Email de réinitialisation de mot de passe	30
6.4	Email de validation de l'adresse email	30
6.5	Tableau de bord	31
6.6	Ajout d'utilisateur	32
6.7	Liste d'utilisateur	32
6.8	Page d'erreur 404	33

Liste des tableaux

Introduction générale

Polyclinique Next dispose de différent branche qui offre de variété de service. que ce soit patient, personnel ou d'autre entité toute cette personnage engendre de grande quantité d'information. Qui nécessite d'être traité et organisé . la présence de grand quantité d'information qui s'échange constamment au sein de chaque service s'avère très difficile à manoeuvrer sans la présence d'un logiciel spécifique pour automatiser certaine fonctionnalité. Grâce au partenariat avec la polyclinique Next et l'école supérieur polytechnique, moi, les encadreur,et les gent du polyclinique ont mis d'accord pour oeuvrer à la conception et réalisation d'un logiciel de gestion qui a pur but d'automatiser la fonctionnalité précédemment évoque dans le cahier de charge suivante.

Chapitre 1

PRESENTATION DE LA POLYCLINIQUE UNIVERSITAIRE NEXT

Sommaire

1.1 Historique	3
1.2 Localisation et information sur le complex hospitalier	3
1.2.1 localisation	3
1.2.2 Information concernant le complexe hospirtalier	4
1.3 Plan hospitalier	4

1.1 Historique

La NEXT onlus a été fondée en 1998 par un chercheur scientifique, Luigi BELLINI, qui, à la suite d'un voyage à Madagascar, profondément touché par les terribles conditions de vie de la majorité (80%) des personnes vivant avec moins d'un dollar par jour, a décidé d'entreprendre une action pour aider ces gens, en consacrant toutes ses énergies et ressources à ce but.

Le travail d'organisation et d'assistance s'est progressivement développé avec une particulière attention dans le domaine de la santé dans le Nord de Madagascar, où ont été accomplies des oeuvres importantes qui interagissent avec le système actuel de la Santé Publique

L'activité de la NEXT, développée au début avec ses propres moyens financiers et sans le soutien d'organismes extérieurs, a été reconnue par l'État Italien, qui, en Octobre 2006, lui a accordé le statut juridique d'ONG (Organisation Non Gouvernementale).

Le professeur Luigi BELLINI a progressivement abandonné ses activités en Italie, ses travaux de recherche à l'Université et professionnel, même sa collaboration scientifique avec l'Agence spatiale européenne. Il a décidé de réaliser un centre de diagnostic médical à Madagascar, dénommé « Le Samaritain ». Le centre, d'une superficie de 3 000 m², comprenant des laboratoires de radiologie et d'échographie, a débuté ses activités, en 2006. Il est doté d'équipements modernes de niveau européen.

En constatant que le centre de diagnostic à lui seul ne suffit pas, l'ONG a mis sur les rails la première clinique de maternité et de chirurgie dans le Nord. Une grande réalisation, unique à Madagascar. Les travaux de construction ont débuté en 2009 : ce fut les premières fondations de la Polyclinique universitaire NEXT.

1.2 Localisation et information sur le complexe hospitalier

1.2.1 localisation

Le Centre "Le Samaritain" ainsi que la Clinique de Maternité et Chirurgie se trouvent dans la région DIANA, située au Nord de Madagascar. (Pour arriver au complexe sanitaire il faut s'engager sur la route nationale 6 (RN6) et procéder 200 m avant de tourner à gauche vers « Rue de la Fraternité ».)

1.2.2 Information concernant le complexe hospitalier

L'objectif de la NEXT était d'assurer l'assistance médicale aux gens de la Région d'Antsiranana et de toutes les Provinces du Nord de Madagascar par la construction et l'équipement d'une Clinique, située à côté du Centre de Diagnostic Médical "Le Samaritain". Cette nouvelle structure à trois étages abritera les départements de maternité, chirurgie générale et médecine et sera équipée de :

- une salle opératoire principale
- une salle opératoire d'urgence ou secondaire
- une salle d'accouchement
- une salle de stérilisation
- une salle postopératoire
- un centre d'hémodialyse
- un service d'urgences.

1.3 Plan hospitalier

L'établissement de santé est reparti comme suit :

REZ-DE-CHAUSSÉE	1ER ETAGE :	2EME ETAGE
Triage-urgence Hemodialyse Direction-administration	Bloc opératoire Chirurgie Gynéco-obstétrique Maternité Salle d'accouchement	Médecine Pharmacie Salle de classe Présidence

Chapitre 2

MODELISATION DE L'APPLICATION

Sommaire

2.1 Analyse de fonctionnement de l'hôpital	7
2.1.1 L'accueil	7
2.1.2 Les départements d'admission	7
2.1.3 L'administration	7
2.1.4 La caisse	8
2.1.5 La direction	8
2.1.6 La médecine	8
2.2 Diagramme de cas d'utilisation	9
2.2.1 Diagramme de cas d'utilisation générale du système	9
2.2.2 Diagramme de cas d'utilisation réceptionniste	10
2.2.3 Diagramme de cas d'utilisation administrateur polyclinique	11
2.2.4 Diagramme de cas d'utilisation admin logiciel	12
2.3 diagramme de séquence	13
2.3.1 Diagramme de sequence gestion patient	13
2.3.2 Diagramme de séquence gestion facturation	14
2.4 Diagramme de classe de l'application	14

Un modèle est une simplification de la réalité. Alors la modélisation permet de maîtriser la complexité du système étudié, car chaque modèle donne accès à une représentation abstraite de différents aspects du système. modéliser un système consiste à créer une représentation schématique simplifiée d'un problème. le but de :

- Nous aider à visualiser le système tel qu'il est ou tel qu'il devrait être.
- Spécifier la structure et le comportement d'un système
- Avoir un "patron" pour guider la construction du système.
- Documenter les décisions qui ont été prises

Pourquoi Modéliser? La réponse c'est pour Mieux comprendre le système en développement. Nous construisons des modèles de systèmes complexes parce nous sommes incapables d'appréhender ces systèmes dans leur entièreté.

2.1 Analyse de fonctionnement de l'hôpital

De façon générale, le fonctionnement de l'hôpital repose sur l'ordonnancement des branches ci dessous.

2.1.1 L'accueil

Un nouvel arrivant dans le centre hospitalier doit impérativement se présenter à l'accueil. De là, le responsable enregistre les renseignements de base du nouveau malade, à savoir : La date d'entrée, le nom Prénom, le sexe, l'âge, l'adresse, le téléphone, le médecin traitant, la famille ou personne externe, la profession et l'unité d'admission. Sur l'unité d'admission, ceci est à propos du département dans lequel le patient va être orienté Il est à noter que les renseignements sus enregistrés, feront l'objet d'une fiche qui sera remplie au fur et à mesure du traitement du patient concerné, une fois admis dans l'unité d'admission

Quand un malade doit être hospitalisé, c'est également à l'accueil que cet évènement doit être pris en compte. En effet, le responsable enregistre :

Le numéro du patient (entre autre, son dossier médical), la chambre le lit et la catégorie de la chambre

Comme le rôle de l'accueil se repose surtout sur l'enregistrement du mouvement de la population hospitalière, à la sortie d'un patient, un personnel de l'accueil enregistre la sortie, avec les attributs suivants : le numéro de patient le nom et prénom, l'état de sortie (mort, guérison, amélioration, statu-quo), la décision de sortie (de l'hôpital ou volontaire)

2.1.2 Les départements d'admission

A La polyclinique Universitaire NEXT, on retrouve les départements suivants :

la maternité, chirurgie,bloc, opératoire, néphrologie, pédiatrie, gynécologie, mise en observation.

Une fois que le patient est orienté vers son unité d'admission, sa fiche sera modifiée, en rapport avec son suivi médical.

2.1.3 L'administration

Toutes les prestations effectuées par le patient sont notées et enregistrées par un responsable de administration.

2.1.4 La caisse

La caisse est chargée du service facturation. Quand un patient va passer au paiement, il se présente à la caisse. Le caissier consulte les prestations saisies par l'administration. Son rôle vise également à l'évaluation et l'exploitation d'un certain nombre d'informations et de statistiques, liées à la comptabilité des prestations.

2.1.5 La direction

La direction a dans ses attributs, la gestion des utilisateurs, la diffusion d'informations/messages au personnel.

2.1.6 La médecine

La section médecine est dédiée aux médecins, plutôt dans le sens de prescription. Quand un patient se présente alors au cabinet, le médecin lui prescrit une ordonnance

2.2 Diagramme de cas d'utilisation

2.2.1 Diagramme de cas d'utilisation générale du système

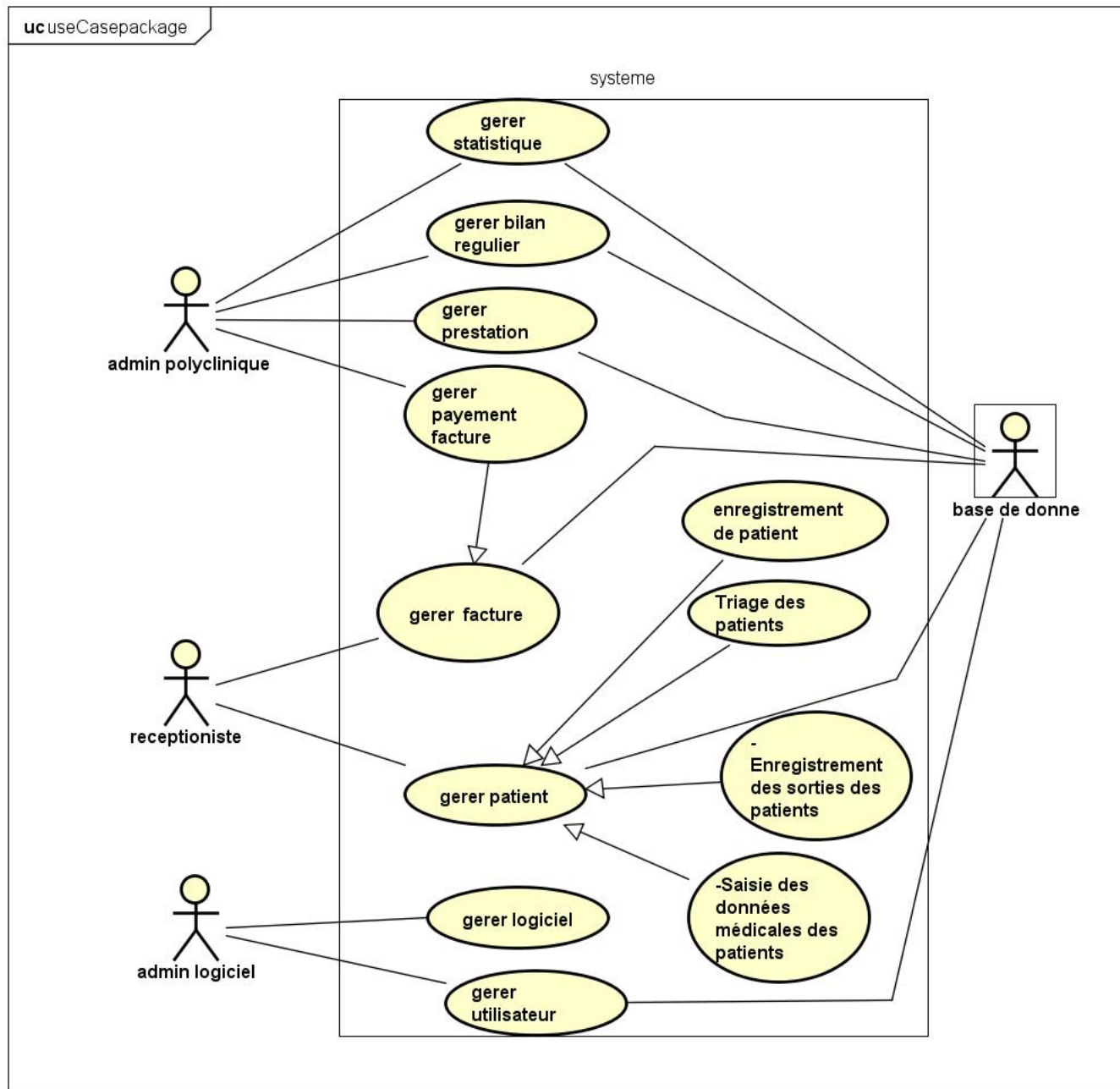


FIGURE 2.1 – Diagramme de cas d'utilisation générale du système

2.2.2 Diagramme de cas d'utilisation réceptionniste

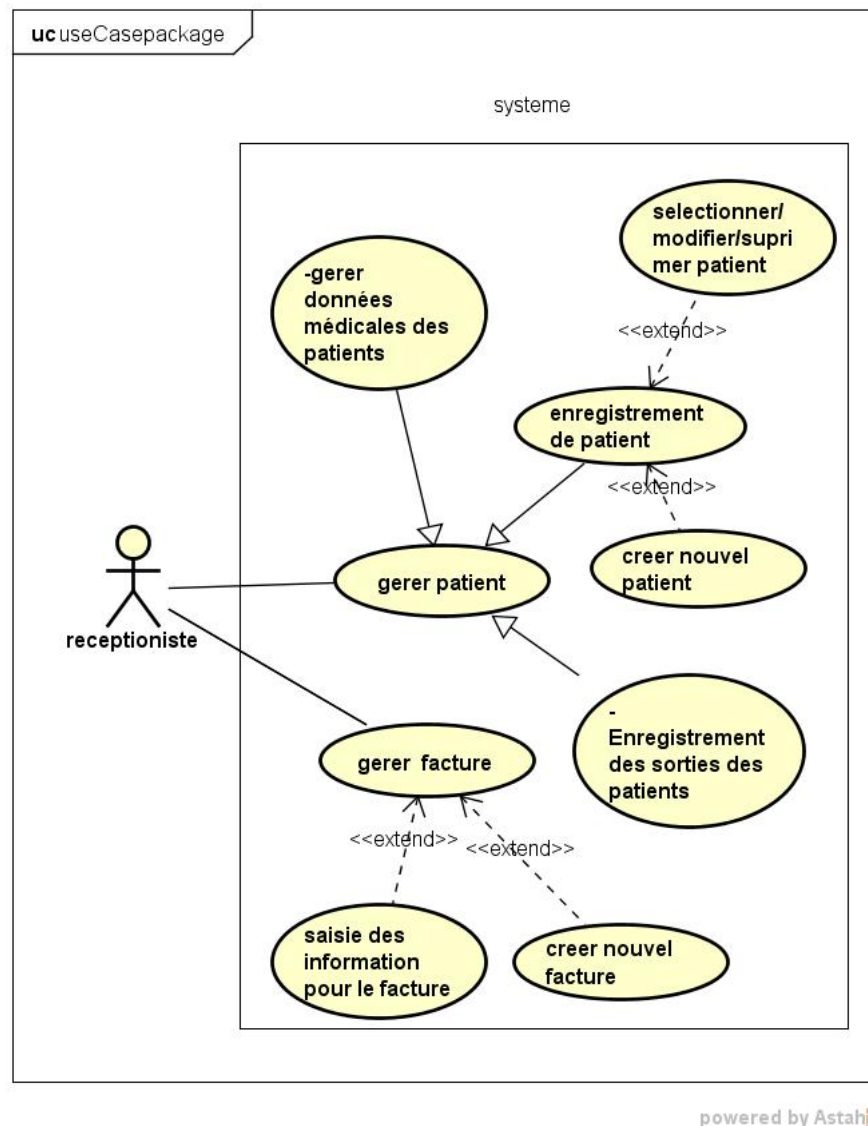
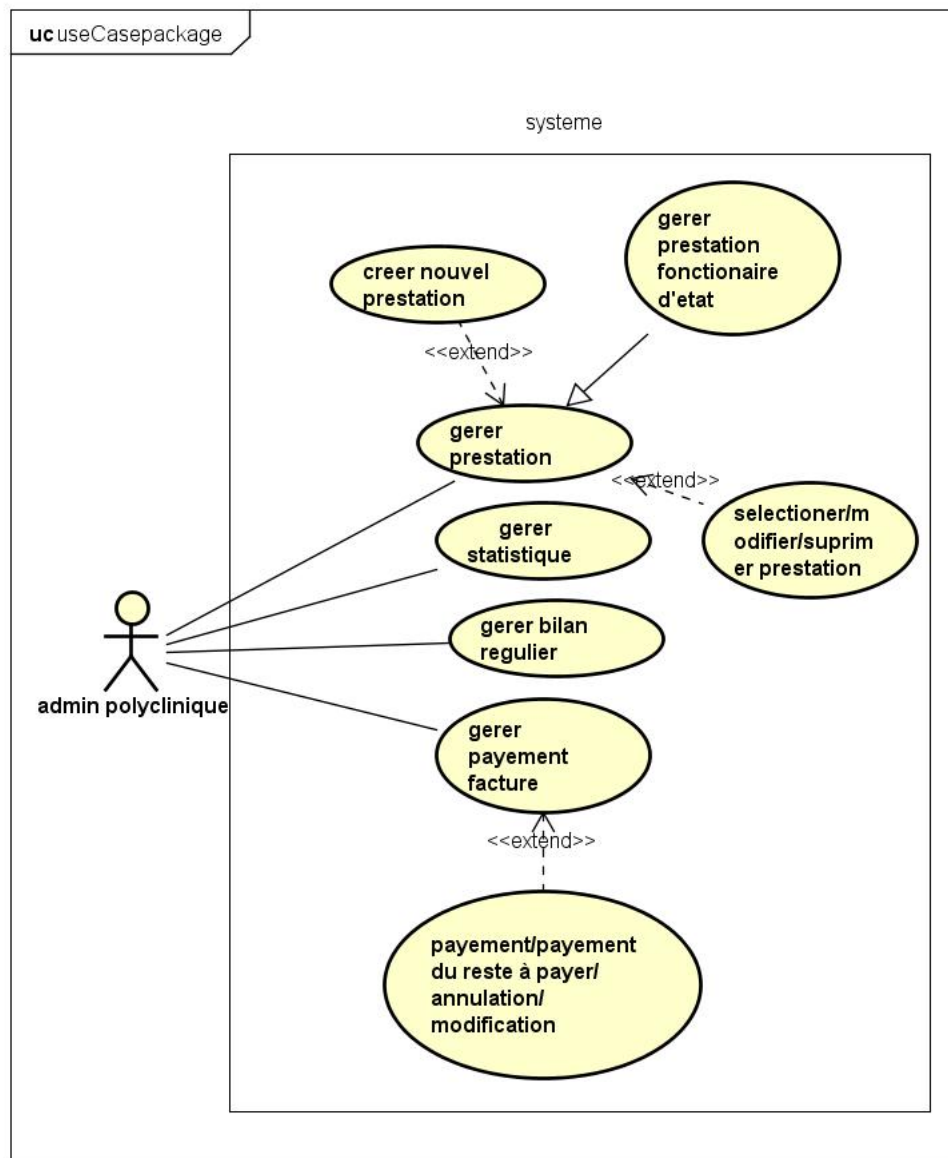


FIGURE 2.2 – Diagramme de cas d'utilisation réceptionniste

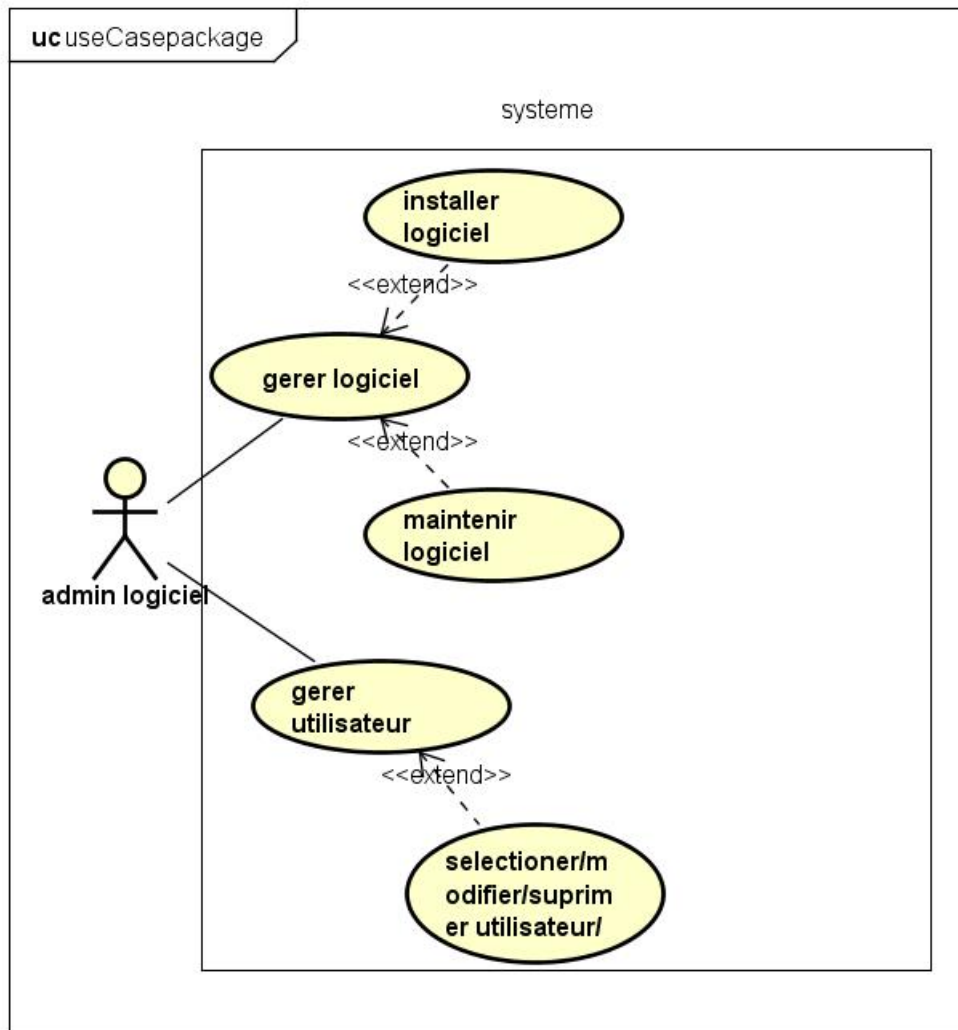
2.2.3 Diagramme de cas d'utilisation administrateur polyclinique



powered by Astah

FIGURE 2.3 – Diagramme de cas d'utilisation administrateur polyclinique

2.2.4 Diagramme de cas d'utilisation admin logiciel



powered by Astah

FIGURE 2.4 – Diagramme de cas d'utilisation administrateur logiciel

2.3 diagramme de séquence

2.3.1 Diagramme de sequence gestion patient

2.3.2 Diagramme de séquence gestion facturation

2.4 Diagramme de classe de l'application

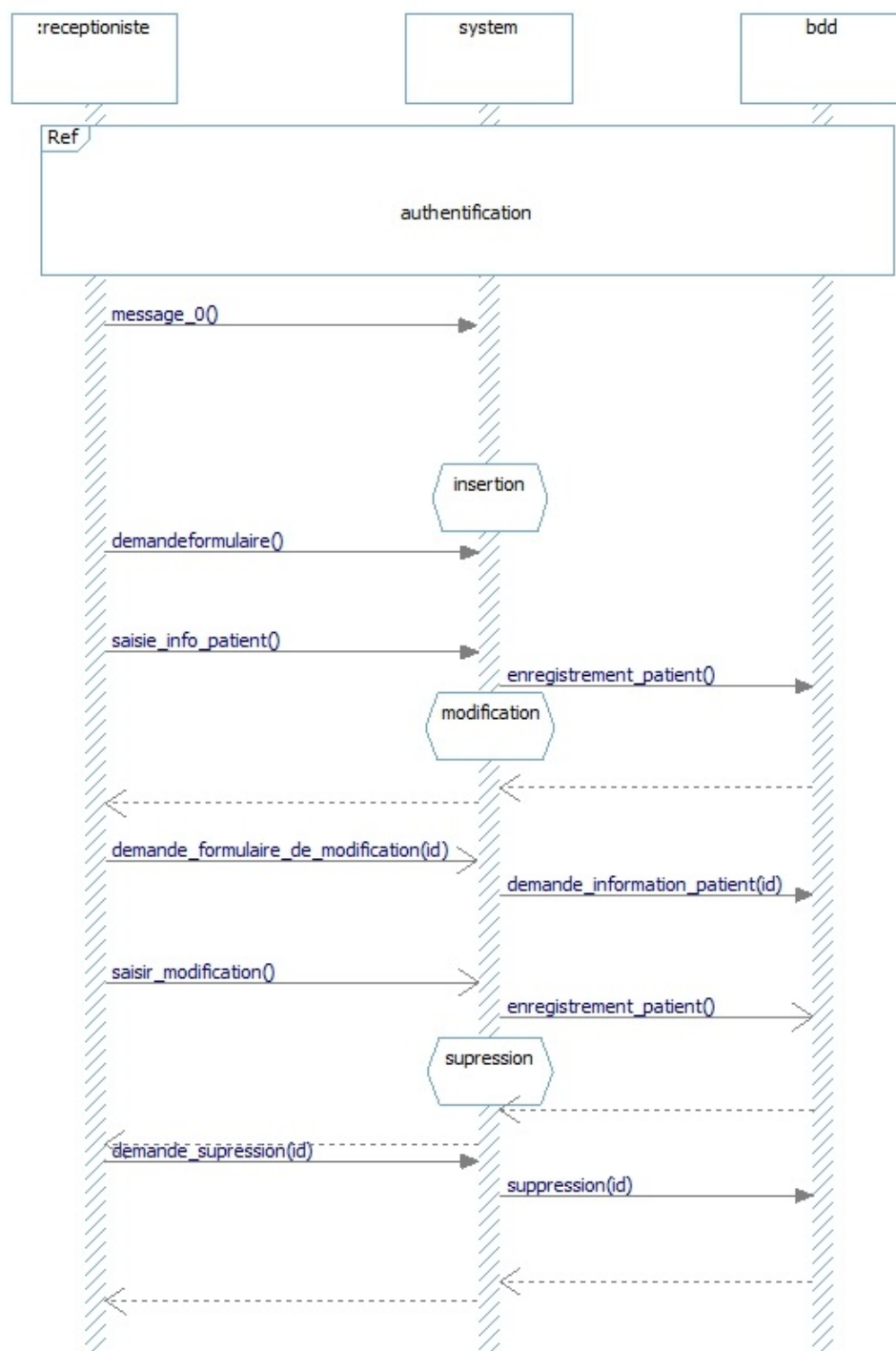


FIGURE 2.5 – Diagramme de séquence gestion patient

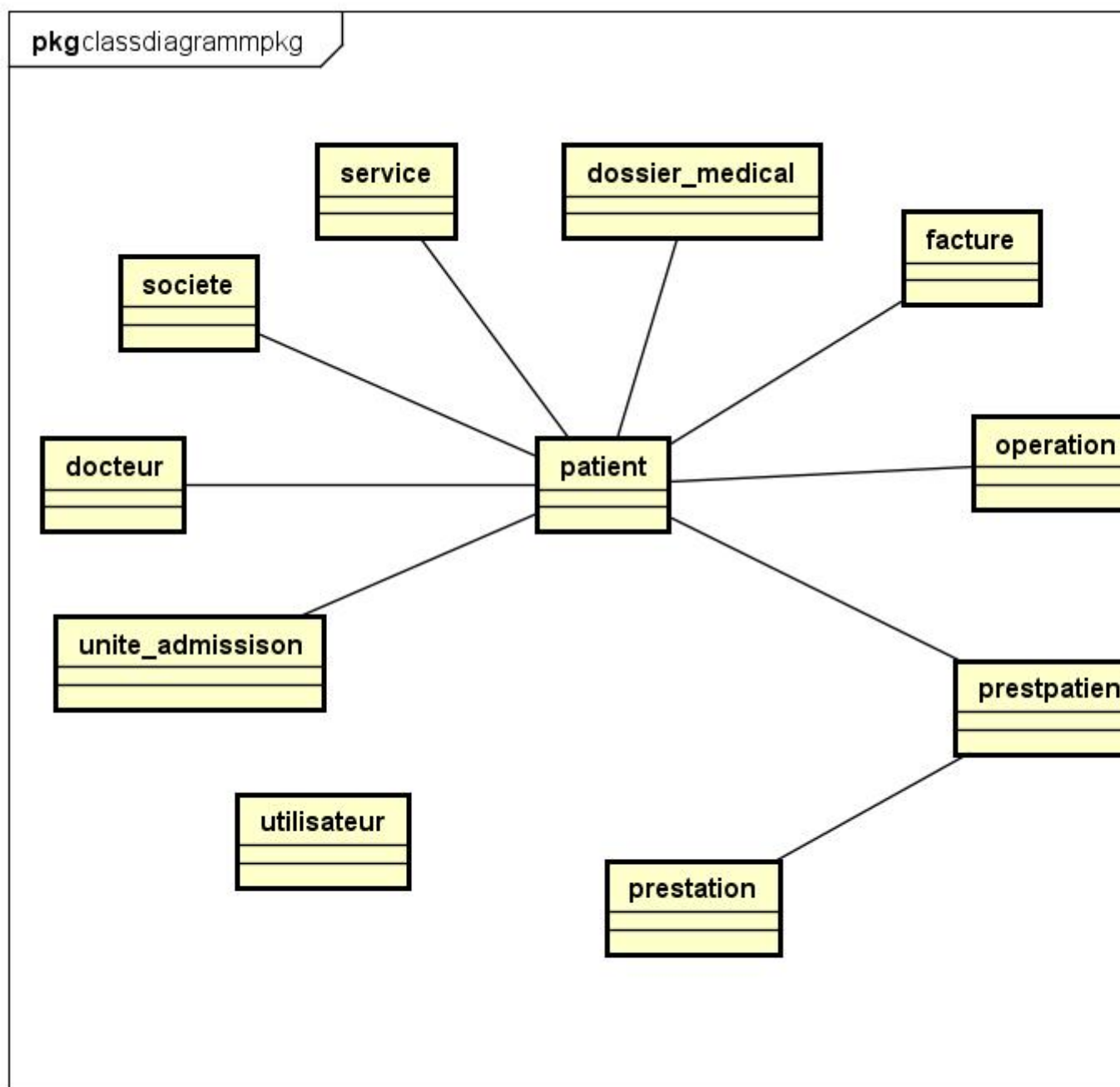


FIGURE 2.6 – Diagramme de classe de l'application

Chapitre 3

Réalisation de l'application

La réalisation et le codage de l'application est une étape cruciale et inévitable, dans le développement de l'application, l'application ne se construit tout seul après la modélisation, pour que l'application ne soit pas devenir une simple rêve il faut le produire. Nous avons utilisé laravel pour le codage.

3.1 Le framework Laravel

Les applications Laravel sont installées et gérées avec Composer , un gestionnaire de dépendance PHP. Il existe deux manières de créer une nouvelle application Laravel.

Sur le terminal : `composer create-project laravel/laravel [foldername]`

Chapitre 4

Modélisation statique et dynamique du système

Chapitre 5

Programmation et réalisation du logiciel

Sommaire

5.1 Introduction	20
5.2 Les Systèmes de Gestion de Base de Données (SGBD)	20
5.2.1 Définition	20
5.2.2 Firebase	20
5.2.3 Pourquoi le choix de Firebase	20
5.3 Les langages de programmation	21
5.3.1 De JSF à Angular	21
5.3.2 Angular	21
5.3.3 Une navigation plus fluide pour le visiteur	22
5.3.4 Une meilleure gestion de contenu dynamique	22
5.3.5 Une plateforme extensible et modulaire	22
5.3.6 Présentation en HTML5/CSS3/JS/jQuery	23
5.4 Conception	23
5.4.1 Création du projet	23
5.4.2 La structure des dossiers	23
5.4.3 Firebase Realtime Database, une base de données en temps réel	25
5.4.4 Sécurisation des données	26

5.1 Introduction

Après avoir franchi l'étape de la modélisation, on peut désormais entamer la programmation et la réalisation du système. Dans cette partie, on abordera en premier lieu la description de notre Système de Gestion de Base de Données (SGBD) ainsi que les langages de programmation utilisés et par la suite on présentera les démarches de conception pour l'implémentation du système.

5.2 Les Systèmes de Gestion de Base de Données (SGBD)

5.2.1 Définition

Les SGBD sont des logiciels système destinés à stocker et à partager des informations dans une base de données, en garantissant la qualité, la pérennité et la confidentialité des informations, tout en cachant la complexité des opérations. En bref c'est un logiciel intermédiaires entre les utilisateurs et les bases de données. Une base de données est un magasin de données composé de plusieurs fichiers manipulés exclusivement par le SGBD. Ce dernier cache la complexité de manipulation des structures de la base de données en mettant à disposition une vue synthétique du contenu.

5.2.2 Firebase

Firebase est à la fois un SGBD et un ensemble de services d'hébergement pour n'importe quel type d'application (Web service, Android, iOS, Javascript, Node.js, Java, Unity, PHP, C++ ...). Il propose d'héberger en NoSQL et en temps réel des bases de données, du contenu, de l'authentification sociale (Google, Facebook, Twitter et Github), et des notifications, ou encore des services, tel que par exemple un serveur de communication temps réel [16].

5.2.3 Pourquoi le choix de Firebase

Lorsqu'on développe une application, qu'elle soit destinée au grand public ou réservée à un usage interne à l'entreprise, certaines fonctionnalités sont systématiquement requises, telles que la gestion des utilisateurs, de la connexion et des notifications. La gestion de ces fonctionnalités est fastidieuse, répétitive si le système se compose de plusieurs applications, et critiques en termes de sécurité, dans la mesure où l'on

va stocker des mots de passe. D'où l'intervention de Firebase qui va nous permettre d'externaliser cette gestion.

5.3 Les langages de programmation

La présentation des langages de programmation sont des facteurs importants pour la réussite de la conception et la réalisation d'un logiciel. En effet, chaque langage a ses propres caractéristiques qui doivent être compatibles avec les contraintes et conditions exigées par le cahier des charges. Le choix doit être fait en fonction de leur souplesse, leur adaptation aux fonctions exigées et aux ressources matérielles disponibles.

5.3.1 De JSF à Angular

Au début nous avons utiliser le langage "Multi-end" : JSF (Java Server Faces), qui est un framework MVC (Modèle Vue Controleur) Java traditionnels à base d'actions et basé sur la notion de composants. Mais en allant plus loin dans les détails, nous pouvons en déduire des principaux inconvénients qui ont empêchés la progression du développement de l'application.

Deux inconvénients majeurs JSF :

1. Grande courbe d'apprentissage. JSF est complexe, c'est juste vrai.
2. Sa nature composante. Le framwork basé sur les composants essaye de cacher la vraie nature du Web, qui vient avec une énorme quantité de complications et de désastres (comme ne pas soutenir GET dans JSF). Chaque framework basé sur des composants ajoute de l'abstraction au développement Web, et cette abstraction se traduit par un surcoût inutile et une complexité plus élevée.

En bref, JSF essaient de cacher au programmeur la vrai requête/réponse et la nature sans état du web. C'est un désavantage majeur pour JSF, certe il peut convenir à certains types d'applications (intranet, formulaires intensifs), mais pour une application web réelle, ce n'est pas une bonne solution.

5.3.2 Angular

Après avoir été bloqué avec JSF, on a recommencer le développement avec Angular, un langage fessant partie de la nouvelle vague de frameworks JavaScript portée par Google. C'est un socle technique qui se veut extensible et qui pousse vers un développement

structuré. Il s'inscrit dans un mouvement d'innovation côté front-end, dont le but est d'éviter le chargement d'une nouvelle page à chaque action demandée [17].

Angular présente la particularité d'être totalement frontend (côté client). Pour utiliser Angular dans notre application, nous allons devoir utiliser un système backend (côté serveur), dans notre cas ce sera Firebase pour gérer la connexion à la base de données. En utilisant ce langage, on voit clairement la distinction de développement entre front et back.

5.3.3 Une navigation plus fluide pour le visiteur

L'utilisation d'Angular impose un développement selon la structure MVVM (Modèle-Vue-Vue-Modèle). Ce principe offre un avantage de taille, celui de diminuer considérablement la vitesse de chargement des pages. En effet, le nombre d'accès au serveur est fortement diminué car la communication se fait majoritairement en mode asynchrone. Autrement dit, l'interface visuelle est portée côté client. En conséquence, une importante partie des requêtes supportées en arrière-plan est ainsi supprimée, ce qui permet de concevoir des applications web plus légères. Ceci explique sa parfaite adaptation pour les applications web monopage (SPA) qui ne comportent qu'une seule et unique interface.

5.3.4 Une meilleure gestion de contenu dynamique

Le framework estampillé Google étend le langage HTML traditionnel pour enrichir davantage le contenu dynamique par le biais d'un couplage bidirectionnel (two-way data-binding). Derrière ce nom barbare se cache un concept très pratique : dès qu'une vue est modifiée, la donnée est envoyée au model associé qui rafraîchit à son tour la vue. Concrètement, si un internaute remplit un champ texte, la valeur saisie peut s'afficher à un autre endroit de la page et ce sans rechargement ni soumission au préalable de l'information. Il s'agit donc d'une synchronisation entre le modèle et la vue qui permet de créer des applications plus responsives [2].

5.3.5 Une plateforme extensible et modulaire

Pour pallier à la nature statique de la solution HTML, Angular introduit la notion de directives chargée d'associer un comportement JavaScript spécifique à chaque nouvel élément de ce langage balisé. Ces composants vont permettre de rendre le code extensible et modulable. Il devient alors facile d'ajouter, de modifier ou de supprimer des directives,

ce qui fait entre autre la popularité d'Angular. Celles-ci peuvent tout-à-fait être partagées et réutilisées de projet en projet pour éviter de réinventer la roue [3].

5.3.6 Présentation en HTML5/CSS3/JS/jQuery

Pour afficher, mettre en forme et structurer les données à l'utilisateur, nous utilisons les langages HTML (Hypertext Markup Language), le CSS (Cascading Style Sheets) et éventuellement le Javascript (JS) pour l'interactivité de l'application. Ceux-ci ont une caractéristique commune importante : ils sont tous interprétés par le navigateur, directement sur la machine client.

L'utilisation de jQuery est aussi un atout, c'est une bibliothèque JavaScript libre et multi-plateforme créée pour faciliter l'écriture de scripts côté client dans le code HTML.

Cette bibliothèque va nous servir notamment aux fonctionnalités suivantes :

- Manipulation des feuilles de style en cascade (ajout/suppression des classes, d'attributs...);
- Événements;
- Effets visuels et animations;

5.4 Conception

5.4.1 Création du projet

Nous devons avant tout installer Node et npm. Ensuite pour installer angular CLI, il suffit de taper la commande suivante dans votre bash :

```
npm install -g @angular/cli
```

Pour créer maintenant notre application, il suffit d'exécuter la commande suivante :

```
ng new g4sm
```

5.4.2 La structure des dossiers

Après l'installation et la création de votre projet avec angular-cli, nous allons voir la structure des dossiers et fichiers dans l'architecture d'angular-cli.

```
// Tout ce qui va concerner les tests end to end  
|- e2e/
```



```
|----- app.e2e-spec.ts
|----- app.po.ts
|----- tsconfig.e2e.json

// les dépendances avec npm
|- node_modules/

// l'endroit où les fichiers de build seront mis
|- dist/

// Le dossier où vous allez modifier vos fichiers de code
//Là où va se trouver vos composants, services, etc..
|- src/
|----- app/
|----- app.component.css|html|spec.ts|ts
|----- app.module.ts
|----- assets/
|----- environments/
|----- environment.prod.ts|ts
|----- favicon.ico
|----- index.html
|----- main.ts
|----- polyfills.ts
|----- styles.css
|----- test.ts
|----- tsconfig.app.json
|----- tsconfig.spec.json
|----- typings.d.ts

// la configuration globale de votre application
|- .angular-cli.json // the main configuration file
|- .editorconfig      // editorconfig which is used in some VS Code setups
|- .gitignore
```

```
|- karma.conf.js
|- package.json
|- protractor.conf.js
|- README.md
|- tsconfig.json
|- tslint.json
```

Nous allons quasiment passer tout notre temps dans le dossier *src/app*. Ce dossier contient presque tous les fichiers dont nous avons besoin pour coder notre application. Les fichiers contenus dans ce dossier sont ensuite compilés dans le dossier *dist*. Nous pouvons aussi installer des dépendances avec le gestionnaire de package de node : "npm". Ces dépendances seront installées dans le dossier "node modules".

5.4.3 Firebase Realtime Database, une base de données en temps réel

La base de données avec Firebase se présente sous la forme d'un arbre "infini", composé uniquement d'objets clé/valeur. Il s'agit d'une base de données cloud où les données sont stockées sous format JSON (JavaScript Object Notation) et synchronisées en temps réel avec chaque client connecté. Par exemple, si on souhaite stocker un ensemble d'utilisateur, cela devrait plus ou moins ressembler à cela :

```
{
  "utilisateurs": {
    "id1": {
      "nom": "Midonique",
      "email": "mido@gmail.com",
    },
    "id2": { ... },
    "id3": { ... }
  }
}
```

5.4.4 Sécurisation des données

Pour faire simple : l'accès aux données présentes dans la base est régi par un ensemble de règles. Établissons un simple fichier de règles pour nos utilisateurs :

```
{  
  "rules": {  
    ".read": "auth != null",  
    ".write": "auth != null"  
  }  
}
```

C'est uniquement un utilisateur connecté pourra lire, enregistrer et modifier ces données. Il s'agit évidemment ici d'un exemple simple : nous pouvez établir des règles de sécurité bien plus complexes afin de protéger les données des utilisateurs.

Chapitre 6

Présentation et déploiement de l'application

Sommaire

6.1	Pré-requis	28
6.2	Présentation de l'application	28
6.2.1	Fenêtre d'authentification	28
6.2.2	Fenêtre d'inscription	29
6.2.3	Les email de validation et de réinitialisation de mot de passe	30
6.2.4	Le tableau de bord	31
6.2.5	Liste et ajout d'utilisateur	31
6.2.6	Page d'erreur 404	33
6.3	Les services backend de Firebase	33
6.3.1	Authentification	33
6.4	Déploiement de l'application sur Firebase	33
6.4.1	Préparer l'application pour la production	33
6.4.2	Installer Firebase-CLI	34
6.4.3	Configurer l'application Angular	34
6.4.4	Déploiement de l'application	34

6.1 Pré-requis

Pour faire tourner l'application, l'utilisateur devrait munir d'un périphérique possédant un simple navigateur web, tels un ordinateur, une tablette ou smartphone. Puis d'être abonnés à l'application et avoir les codes d'accès. Sachant que l'application est hébergée à distance, une connexion internet est indispensable. Une connexion ADSL (Asymmetric Digital Subscriber Line) de 8Mo/s sera largement suffisante pour cela.

6.2 Présentation de l'application

Nous allons faire une brève présentation de l'interface aussi bien que les fonctionnalités de l'application. Notons au tout début que c'est une application SaaS qui a pour nomination : « Group4Success ». Cette application sera utilisé par différentes établissement en utilisant le même URL.

6.2.1 Fenêtre d'authentification

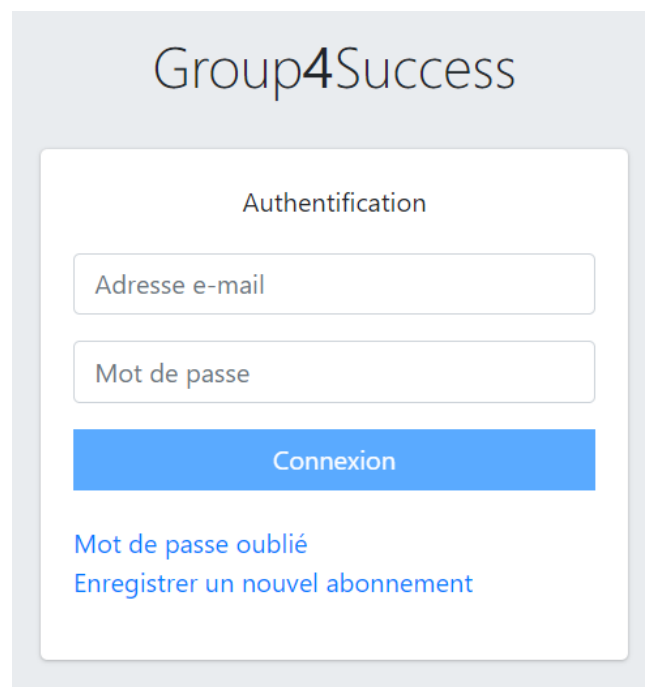
The image shows a web-based authentication window for 'Group4Success'. The window has a light gray border and a white background. At the top, the text 'Group4Success' is displayed in a large, dark gray font. Below this, the word 'Authentification' is centered in a smaller, dark gray font. There are two input fields: the first is labeled 'Adresse e-mail' and the second is labeled 'Mot de passe'. Both fields have a light gray border and a white background. Below the input fields is a blue button with the text 'Connexion' in white. At the bottom of the window, there are two links: 'Mot de passe oublié' and 'Enregistrer un nouvel abonnement', both in blue text.

FIGURE 6.1 – Fenêtre d'authentification

Avant tout, l'établissement client doit impérativement être inscrit à un abonnement sur le site web de G4SM (Group4SuccessManagement) afin de recevoir un identifiant et un mot de passe pour que l'administrateur client puisse s'authentifier dans cette application.

6.2.2 Fenêtre d'inscription

Le site web de G4SM qui devrait servir de présentation et inscription à un abonnement n'est pas encore déployer en ligne. Pour cela, on peut tester l'application par l'intermédiaire du lien "Enregistrer un nouvel abonnement" accessible depuis l'écran d'authentification. Ce dernier ouvre une fenêtre d'inscription pour pouvoir créer un compte de test.

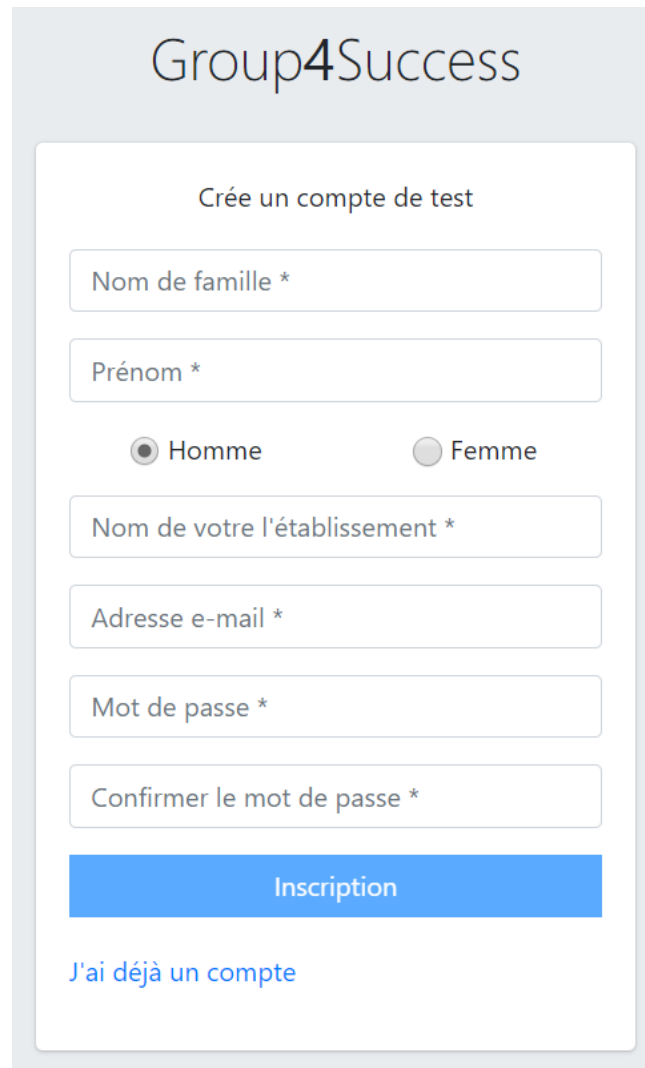
The image shows a registration form for 'Group4Success'. The form is titled 'Crée un compte de test' and contains several input fields: 'Nom de famille *', 'Prénom *', 'Nom de votre l'établissement *', 'Adresse e-mail *', 'Mot de passe *', and 'Confirmer le mot de passe *'. There are also two radio buttons for 'Homme' and 'Femme'. A blue 'Inscription' button is at the bottom, followed by a link 'J'ai déjà un compte'.

FIGURE 6.2 – Fenêtre d'inscription

La validation des données dans les champs du formulaire est gérée par la méthode de réactive fourni par Angular. Cette méthode utilise des fonctions en JavaScript pour écouter les données saisie de l'utilisateur en temps réel.

6.2.3 Les email de validation et de réinitialisation de mot de passe

A partir de la fenêtre d'authentification on peut récupérer notre mot de passe en cas d'oubli. Pour ce faire, il faut cliqué sur le lien "Mot de passe oublié" puis de saisir l'adresse email. Après cela, le système envoi un email à ce dernier pour réinitialiser le mot de passe.



FIGURE 6.3 – Email de réinitialisation de mot de passe

Le système envoi aussi un email de validation après chaque inscription pour vérifier l'authenticité de l'adresse email.



FIGURE 6.4 – Email de validation de l'adresse email

6.2.4 Le tableau de bord

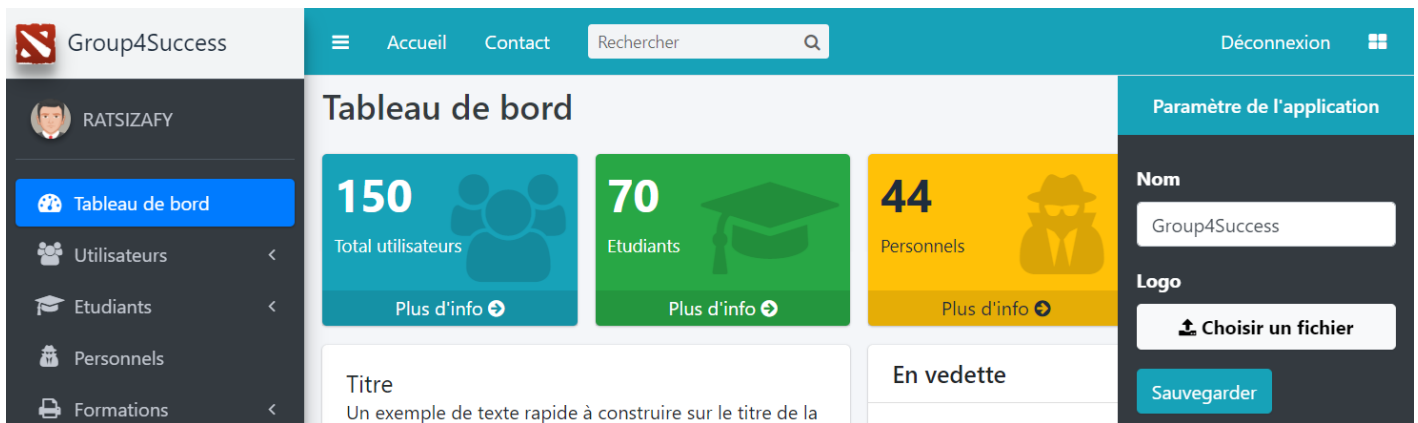


FIGURE 6.5 – Tableau de bord

Après authentification, l'utilisateur sera redirigé directement au tableau de bord orienté rôles qui affiche les chiffres clés de l'établissement et une vue d'ensemble simple et directe des anomalies. Le tableau de bord permettant de zoomer jusqu'au détail et d'utiliser toutes les informations de données de l'établissement. L'architecture du tableau de bord est adaptable et configurable, c'est-à-dire que l'on peut ajouter, supprimer ou déplacer un élément pour le façonner à notre besoin.

La paramètre de l'application, à droite n'est disponible que pour l'administrateur, elle permet de modifier les informations de l'application tels que son nom et le logo.

6.2.5 Liste et ajout d'utilisateur

Pour chaque nouveau utilisateur ajouter, un mail de vérification et réinitialisation de mot de passe sera envoyé à son adresse qu'il puisse s'authentifier à l'application. La modification ou suppression d'un compte utilisateur est accessible en passant par la liste. Cette liste est dynamique, elle offre plusieurs options d'affichage, trie par colonne ou recherche d'un élément spécifique.

Ajout d'utilisateur

Utilisateurs / Ajout

Nom *

Email *

Prénom *

Date de naissance

Rôle

Adresse

Situation Familiale

Téléphone

Sexe

☐ Homme ☐ Femme

Photo

Importer

Sauvegarder

FIGURE 6.6 – Ajout d'utilisateur

Liste des utilisateurs

Utilisateurs / Liste

Affiche 10 entrées

Chercher :

Nom	Prénom	Adresse E-mail	Rôle		
ANDRIANIVO	Dany	dany@gmail.com	Enseignant(e)		
HARIFERA	Finihavana	harifera@gmail.com	Etudiant(e)		
HERINJAKA	Ikotokely	kotokely@gmail.com	Gestionnaire Scolarité		
MANITRINIAINA	Jaosolo	manitriniaina@gmail.com	Gestionnaire RH		
RATSIZAFY	Midonique	midoniquebadgeek@gmail.com	Administrateur		

Affiche 1 à 6 de 6 entrées

Précédent 1 Suivant

FIGURE 6.7 – Liste d'utilisateur

6.2.6 Page d'erreur 404

Au cas où un utilisateur essaie d'accéder à un url non valide qui n'existe pas, il sera redirigé directement vers une page d'erreur 404.



FIGURE 6.8 – Page d'erreur 404

6.3 Les services backend de Firebase

6.3.1 Authentification

Firebase fournit des services de backend pour gérer la gestion des utilisateurs. Avec Firebase Authentication, les mots de passe des utilisateurs sont enregistrés de manière sécurisée dans le cloud, haché avec l'algorithme SCRYPT en 64bits (Voir figure 5.10). Ces mots de passe hachés ne sont même pas accessibles à l'administrateur de Firebase, ils sont conservés en sécurité.

6.4 Déploiement de l'application sur Firebase

Après avoir développé une application il faut pouvoir la déployer sur un serveur pour la rendre disponible par les utilisateurs via un internet. Pour cela nous allons utiliser Firebase hosting comme hébergeur.

6.4.1 Préparer l'application pour la production

A la racine de notre application Angular, on exécute la commande suivante :

```
ng build -prod
```

Celle-ci a pour effet de créer un répertoire *dist*. C'est là que se trouve l'application compilée, optimisée et prête à être déployée.

6.4.2 Installer Firebase-CLI

Firebase-CLI est nécessaire pour configurer notre application sur firebase. Pour cela, dans l'invite de commandes en tant qu'Administrateur, il faut entrer la commande suivante :

```
npm install -g firebase-tools
```

Une fois l'installation terminée, il faut se connecter avec notre compte Google :

```
firebase login
```

Cette commande associe notre PC à notre compte Firebase et nous permet alors d'accéder à nos projets.

6.4.3 Configurer l'application Angular

Nous devons initialiser Firebase Hosting avant de pouvoir déployer notre application Angular. Pour ce faire, on exécute la commande suivante :

```
firebase init
```

6.4.4 Déploiement de l'application

Après quelques configuration avec *firebase init*, on peut finalement déployer notre application avec la commande suivante :

```
firebase deploy
```

Le déploiement effectué, notre application est maintenant accessible via l'URL :

"g4sm-mad.firebaseio.com/"

Conclusion générale

Notre projet intitulé Développement de SaaS « Group4SuccessManagement » est destiné pour alléger les efforts de gestion de personnel, soutenir la croissance de l'établissement, stimuler la réussite des étudiants et améliorer l'efficacité administrative.

Contrairement à la majorité des applications classique, notre application de type SaaS fournit des services aux utilisateurs en échange d'abonnement mensuels, elle est installés sur des serveurs distants plutôt que sur la machine de l'utilisateur. En utilisant ce service l'établissement client n'hébergera pas l'application et ne stockera pas ses données en interne. Les utilisateurs devront simplement disposer d'un dispositif muni d'un navigateur web et des codes d'accès au service en ligne pour pouvoir travailler.

En ce qui concerne les étapes de notre travail, nous avons, en premier lieu, effectué une phase d'étude théorique pour déterminer les problématiques liés à la réalisation du projet pour ainsi fixer les solutions que nous devons adapter. En second lieu, nous avons défini les services fournis par le Group4Success en détaillant les solutions à faire pour satisfaire les quatre objectifs de notre application. En troisième lieu, nous avons modélisé l'application par rapport aux besoins fonctionnels et techniques de notre scénario de base. Après quoi, en quatrième partie, nous sommes passés à la réalisation de l'application en faisant soin de choisir l'outil idéal pour se faire. Dans la dernière étape, nous avons présentés l'application à travers des imprimés écrans et enfin le déploiement de celle-ci dans un serveur distant.

L'application ainsi développée fournit des services pour répondre aux besoins évolutifs des établissements d'enseignement supérieur. Elle permet aux établissements abonnés, possédant des clés d'accès à l'application, de gérer, d'une part, les gestions des employés et des étudiants. Ces gestions peuvent être, la création des fiches employeur, gestion des emplois, gestion de rémunération, traitement de masse, contrôles des personnelles et exportation des données, la gestion des formations, évaluation du parcours. D'autre part, l'outil gère l'établissement et l'administration, à savoir : la gestion des inscriptions,

suivis de l'institution, gestion des locaux, automatisation des procédures courantes, outil de tableau de bord.

Malgré quelques difficultés à mettre en œuvre tous les modules, ce travail nous a présenté une réelle occasion d'apprendre et de faire un premier pas vers le monde du SaaS, toujours en évolution. Ce travail fut très instructif, car l'utilisation d'Angular change la manière de navigation des utilisateurs. C'est plus fluide, une meilleure gestion de contenu dynamique et diminution considérable de la vitesse de chargement des pages. En effet, le nombre d'accès au serveur est fortement diminué car la communication se fait majoritairement en mode asynchrone.

À la fin de cet ouvrage, il nous est important de remarquer que des éventuelles améliorations doivent être apportés au présent projet. Dans la gestion des étudiants : nous pouvons envisager d'implanter un outil pour simulation pour auto-évaluer les progrès des étudiants. Pour l'exportation des fichiers, nous pouvons ajouter d'autres type de format autre que le PDF. Toutefois, nous estimons que les travaux demandés dans le cahier de charge ont été accompli à l'exception de la mise en œuvre de quelques modules qui suscitent d'autres études et analyses.

Bibliographie

- [1] Généralité sur la gestion des ressources humaines. *Robert* NIKIKO. IRIMAG - Licence en Administration et Management.
- [2] Deviens un ninja avec Angular. *Ninja Squad*. Learnangularjs, 2016.
- [3] Learn Angular 2 step by step. *Questpond*, 2016.
- [4] Cours de base de données. *R. Njakarison Menja*, 2013.
- [5] Les établissements d'enseignement supérieur : Structure et fonctionnement, 2017. Page 12-13. Brochures de PARFAIRE. Chapitre structure et fonctionnement. creativecommons.org/licenses/by-nc-nd/4.0/fr/.
- [6] Firebase, 2017. Gregory HOWARD - Alix DUCROS. *Build Extraordinary Apps*.
- [7] Angular 2 & 4 maîtriser le Framework Front-End de Google. ORSYS, La Grande Arche.
- [8] Firebase Realtime Database. *Landon COX*, 2017

Webographie

- [9] Études supérieures, 06 Mars 2016. Wikipédia. fr.wikipedia.org.
- [10] Réforme Licence-Master-Doctorat, Octobre 2017. Wikipédia. fr.wikipedia.org.
- [11] Gestion des ressources humaines, Septembre 2017. Wikipédia. fr.wikipedia.org.
- [12] Le cloud computing, 21 Mars 2016. Wikipédia. fr.wikipedia.org.
- [13] Le logiciel en tant que service, Juin 2016. Wikipédia. fr.wikipedia.org.
- [14] FCIC Média. Le mode SaaS, 2017. www.lecoindesentrepreneurs.fr/le-mode-saas/.
- [15] Secteur de l'éducation, 2017. Unit4. Erp. www.unit4.com/fr/secteurs/education.
- [16] Firebase, Novembre 2017. Wikipédia. fr.wikipedia.org/wiki/Firebase.
- [17] Les raisons clés de développer avec angular, 2016. Caroline PHILLIPS. www.pic.digital/blog/396-angularjs-pourquoi.
- [18] Initiation à la conception de bases de données avec merise, Février 2012. Idriss NEUMANN. ineumann.developpez.com/tutoriels/merise/initiationmerise/.
- [19] How to Use Local Storage and Session Storage In Angular 4, Janvier 2018. JsTutorials Team. www.js-tutorials.com/javascript-tutorial/use-localstorage-sessionstorage-using-webstorage-angular4/.
- [20] Angular Authentification Tutorial, 5 Décembre 2017. Ado KUKIC. auth0.com/blog/angular-2-authentication/.
- [21] Firebase Guides, 8 Mai 2018. firebase.google.com/docs/guides/.
- [22] Firebase Authentication, 8 Mai 2018. <https://firebase.google.com/docs/auth/>.
- [23] One framework Mobile and Desktop, Angular. angular.io.
- [24] Bootstrap Documentation, v4.1. Core Team. <https://getbootstrap.com/docs/4.1/>.
- [25] jQuery Tutorial, 2018. www.w3schools.com/Jquery/default.asp.
- [26] SweetAlert, Hand-crafted, by Tristan EDWARDS. sweetalert.js.org/guides/.

Annexes

A Code TypeScript : Authentication

```
onSubmit(formData) {
  this.errorMessage = "";
  if (formData.valid) {
    //console.log(formData.value);
    this.af.auth.signInWithEmailAndPassword(this.email, this.password)
      .then((success) => {
        var user = this.af.auth.currentUser;

        if (user.emailVerified) {
          this.isError = false;
          this.router.navigate(['/dashboard']);
          window.location.reload();
        }
        else {
          swal({
            title: "Compte désactivé!",
            text: "Veuillez consulter votre adresse e-mail pour
              activer votre compte",
            icon: "warning",
            buttons: ["Annuler", "Renvoyer l'email de confirmation"],
            dangerMode: false,
          })
            .then((willDelete) => {
              if (willDelete) {
                this.sendVerification();
              } else {
                this.logout();
              }
            });

          this.email = null;
          this.password = null;
        }
      })
      .catch((error) => {
        this.isError = true;
        console.log('Error Name', error['code']);
      });
  }
}
```



```

        console.log('Error Message', error['message']);
        this.getError(error);
    })
}
}

```

B Upload de fichier

```

pushUpload(upload: Upload, basePath) {
    const storageRef = firebase.storage().ref();
    const uploadTask = storageRef.child(`${basePath}/${new Date()
        .getTime()}_${upload.file.name}`)
        .put(upload.file);

    uploadTask.on(firebase.storage.TaskEvent.STATE_CHANGED,
        // trois observateur
        // 1.) changement d'etat observateur
        (snapshot) => {
            // upload en cours
            upload.progress =
                (snapshot.bytesTransferred / snapshot.totalBytes) * 100;
            console.log('Progress : ', upload.progress);
        },
        // 2.) erreur observateur
        (error) => {
            // upload failed
            console.log(error)
        },
        // 3.) success observateur
        (): any => {
            // upload success
            upload.url = uploadTask.snapshot.downloadURL;
            upload.name = upload.file.name;
            upload.createdOn = new Date();
            //this.saveFileData(upload, basePath);
        }
    )
};

```

C Divers fonction de validation de formulaires

```

export function RequiredField(c: AbstractControl) {
    if (c.value.trim() == '' || c.value.trim() == null) {
        return {'required': true}
    }
}

export function MinLenght(nbr: number) {

```

```
    return (c: AbstractControl) => {
      if (c.value.trim().length < nbr) {
        return {'minLenght': true}
      }
    }
  }

export function EmailValidation(c: AbstractControl) {
  let isValid = /^[_a-zA-Z0-9]+(\.[_a-zA-Z0-9]+)*@[a-zA-Z0-9-]+
(\.[a-zA-Z0-9-]+)*(\.[a-zA-Z]{2,4})$/ .test(c.value);
  if (!isValid) {
    return {'emailFormat': true};
  }
}

export function TelValidation(c: AbstractControl) {
  let isValid = /^032|033|034 \d{2} \d{3} \d{2}/.test(c.value);
  if (!isValid) {
    return {'telFormat': true};
  }
}

export function PasswordValidation(c: AbstractControl) {
  let isValid = /^(?=.*[a-z])(?=.*[A-Z])(?=.*[0-9])(?=.*{6,100})/
.test(c.value);
  if (!isValid) {
    return {'passwordValid': true};
  }
}
```

RÉSUMÉ

Le système d'information étudiants des établissements d'enseignement supérieur ne répond plus aux besoins du modèle d'activité existante. D'où l'idée de ce travail de mémoire de développer un SaaS (Software as a Service ou logiciel en tant que service) dénommé "Group4Success" qui est destiné pour : alléger les efforts de gestion de personnel, soutenir la croissance de l'établissement, stimuler la réussite des étudiants et améliorer l'efficacité administrative.

L'application est développée avec Angular, la nouvelle vague de frameworks JavaScript portée par Google. Elle est installée sur un serveur distant dans Firebase, plutôt que sur la machine de l'utilisateur, ainsi l'établissement client n'hébergera pas l'application et ne stockera pas ses données en interne.

En bref, le "Group4Success" est une application web de type SaaS qui va fournir des services pour répondre aux besoins évolutifs des établissements d'enseignement supérieur.

ABSTRACT

The student information system of higher education institutions no longer meets the needs of the existing business model. Hence the idea of this work of memory to develop a SaaS (Software as a Service) called "Group4Success" which is intended to : lighten the efforts of personnel management, support the growth of the establishment, stimulate student success and improve administrative efficiency.

The application is developed with Angular, the new wave of JavaScript frameworks promoted by Google. It is installed on a remote server in Firebase, rather than on the user's machine, so the client institution will not host the application and will not store its data internally.

In short, the "Group4Success" is a SaaS-based web application that will provide services to meet the evolving needs of higher education institutions.