Table des matières

Introduction générale …………………………………………………………………………8

1. Etude préalable

1.1 Introduction ..................……………………………………………………..9

1.2 Objectif du PF ………...……………………………………………………..9

1.3 Présentation de l’organisme ………………………………………………… 10

1.3.1 Identité …………………………………………………………………...10

1.3.2 Secteur d’activité .……………………………………………………….10

1.4 Problématique ……………………………………………..…………………11

1.5 Langage et méthodologies de conception ……………………………..………12

1.6 Etude de l’existant …………………………………………………….………14

1.7 Solution proposée …………………………………………………………16

1.7.1 Apport technique …………………………………….…………………17

1.7.2 Apport fonctionnel …………………………………..…………………17

1.8 Conclusion ……………………………………………..………………18

1. Le Backlog du produit
   1. Introduction ………………………………………………..………………19
   2. Capture des besoins ………………………………………….………………19
      1. Identification des acteurs ………………………………….…………19
      2. Besoins fonctionnels ………………………………….………………20
      3. Besoins non fonctionnels ……………………………………………20
   3. Planning du traitement des cas d’utilisation …………………………………21
   4. Pilotage du projet avec SCRUM ……………………………………………21
      1. Les fonctionnalités du Backlog ………………………………………21
      2. Diagramme de cas d’utilisation ………………………………………23
      3. Architecture ……………………………………………………………25
      4. Planification des sprints …………………………………..……………26
   5. Conclusion …………………………………………………………………27
2. Release 1 : Application Android
   1. Introduction …………………………………………………………………28
   2. Le premier sprint …………………………………………………….………28
      1. Analyse …………………………………………………………………28
         1. Raffinement des cas d’utilisation …………………………..………28
3. Cas d’utilisation «  Rechercher médecin » ……………………..28
4. Cas d’utilisation « Localiser médecin » …………………….…31
5. Cas d’utilisation « Prendre RDV par Mail » …………………...32
6. Cas d’utilisation « Appeler Médecin » …………………………33
   * + 1. Diagramme de classe d’analyse…………………………………….34
7. Cas « Rechercher médecin » ……………………………….…34
8. Cas d’utilisation « Localiser médecin » …………………….….34
   * 1. Conception ………………………………………………………….…35
        1. Diagramme de séquence ………………………………………...…35
9. Cas d’utilisation «  Lister médecin » …………………….….….35
10. Cas d’utilisation «  Filtrer médecin »   ………….………….…36
11. Cas d’utilisation « Localiser médecin » ………….…………….36
12. Cas d’utilisation « Prendre RDV par Mail ………….….………37
13. Cas d’utilisation « Appeler Médecin » …….…………………...38
    * + 1. Diagramme de classe de conception ………………………….….….38
14. Cas d’utilisation «  Rechercher médecin » ………………….…...38
15. Cas d’utilisation « Localiser médecin » ……………….………..39
    * 1. Codage …………………………………………………………….……39
      2. Test ………………………………………………………………………40
    1. Le second sprint ………………………………………………………………43
       1. Analyse ………………………………………………………………..…43
          1. Raffinement des cas d’utilisation …………………………………..43
16. Cas d’utilisation « Créer Compte » …………………………………..43
17. Cas d’utilisation « S’authentifier » ………………………………..…44
18. Cas D’utilisation « Gérer Médecin Favoris » ……………………..…45
    * + 1. Diagramme de classe d’analyse ……...…………………………….47
19. Cas d’utilisation « Créer Compte » …....………………………...47
20. Cas d’utilisation « S’authentifier »  …….………………………47
21. Cas d’utilisation « Gérer Médecin Favoris » ……………………48
    * 1. Conception ………………………………………………………………48
         1. Diagramme de séquence ……………………………………………48
22. Cas d’utilisation « Créer Compte » …………………………….. 49
23. Cas d’utilisation « S’authentifier » ……………………………...49
24. Cas d’utilisation « Gérer Médecin Favoris » ……………………50
    * + 1. Diagramme de classe de conception ……………………………52
25. Cas d’utilisation « Créer Compte » …………………………...52
26. Cas d’utilisation « S’authentifier » …………………………… 53
27. Cas d’utilisation « Gérer Médecin Favoris » …………….…53
    * 1. Codage …………………………………………………………..54
      2. Test ……………………………………………………………………..54
    1. Conclusion ………………………………………………………………55
28. Release 2 : Application Web
    1. Introduction ………………………….…………….…………………….…56
    2. Le premier sprint ……………..………………………………………………56
       1. Analyse …………………………………………………………………..57
          1. Raffinement des cas d’utilisation …………………………………..57
29. Cas d’utilisation «  S’authentifier » ...………………..................57
30. Cas d’utilisation «  Gérer médecins » ......……………………..58
    * + 1. Diagramme de classe d’analyse …..……………………………….…61
31. Cas d’utilisation «  S’authentifier »  .....……………………….…61
32. Cas d’utilisation «  Gérer médecins »  ….………………………..61
    * 1. Conception .....……………………………………………………………62
         1. Diagramme de séquence système …………………………………62
33. Cas d’utilisation  « S’authentifier » …………………….....62
34. Cas d’utilisation «  Gérer médecin »………………………....62
    * + 1. Diagramme de classe de conception…………………………………65

A. Cas d’utilisation « S’authentifier »………………………………..65

B. Cas d’utilisation «  Gérer médecins »……………………………..65

* + 1. Codage ………………………………………………………………..….66
    2. Test……………………………………………………………………….66
  1. Conclusion …………………………………………………………………….70

1. Phase de clôture ……………………………………………………………………...71
   1. Introduction …………………………………………………………………....71
   2. Environnement de développement et langages de programmation …………...71
      1. Application mobile……………………………………………………….71
      2. Application Web………………………………………………………....72
   3. Perspectives …………………………………………………………………...74

Conclusion Générale ………………………………………………………………………...75

Liste des figures

Figure 1 : Le processus Scrum ……………………………………………………………13

Figure 2 : Interfaces de l’application Santé Tunisie ...……………………………………….15

Figure 3 : Interfaces de l’application Dot Medical ………………………………………….15

Figure 4 : Interfaces de l’application Sehatuk………………….…………………………….16

Figure 5 : Diagramme de cas d’utilisation pour l’application Android médecin ……………24

Figure 6 : Diagramme de cas d’utilisation pour l’application Web ………………………....26

Figure 7 : Raffinement du cas d’utilisation « Rechercher médecin » ……………………….29

Figure 8 : Raffinement du cas d’utilisation « Localiser médecin » …………………………31

Figure 9 : Raffinement du cas d’utilisation « Prendre RDV par Mail» ……………………...32

Figure 10 : Raffinement du cas d’utilisation « Appeler Médecin» …………………………..33

Figure 11 : Diagramme de classe d’analyse du cas « Rechercher médecin »………………..34

Figure 12: Diagramme de classe d’analyse du cas « Localiser médecin»…………………...35

Figure 13 : Diagramme de séquence système du sous cas « lister tous les médecins »……...35

Figure 14 : Diagramme de séquence système du sous cas « Filtrer médecins »……………..36

Figure 15 : Diagramme de séquence système du cas « Localiser médecin »………………..37

Figure 16 : Diagramme de séquence du cas « Prendre RDV par Mail »………….…………37

Figure 17 : Diagramme de séquence du cas « Appeler Médecin »…………………………..38

Figure 18 : Diagramme de classe de conception du cas «Rechercher médecin»……………39

Figure 19 : Diagramme de classe de conception du cas «Localiser médecin»……………...39

Figure 20 : Interfaces recherche médecin…………………………………………………….41

Figure 21: Interfaces Localisation médecin (Terrain, Satellite, Plan)………………………..42

Figure 22 : Interfaces Appeler et Envoyer mail ……………………………………………..42

Figure 23 : Raffinement du cas d’utilisation « Créer Compte »……………………………...43

Figure 24: Raffinement du cas d’utilisation « S’authentifier » ……………………………....44

Figure 25 : Raffinement du cas d’utilisation « Gérer Favoris »………………………………45

Figure 26 : Diagramme de classe d’analyse du cas d’utilisation « Créer Compte »…….……47

Figure 27 : Diagramme de classe d’analyse du cas « S’authentifier»……………….……….48

Figure 28 : Diagramme de classe d’analyse du cas d’utilisation « Gérer Favoris»………….48

Figure 29 : Diagramme de séquence du cas d’utilisation « Créer Compte»………………….49

Figure 30 : Diagramme de séquence du sous cas d’utilisation « S’authentifier»…...………..50

Figure 31 : Diagramme de séquence du sous cas d’utilisation « Ajouter Favoris»………..…50

Figure 32 : Diagramme de séquence du sous cas d’utilisation « Consulter Favoris»……...…51

Figure 33 : Diagramme de séquence du sous cas d’utilisation « Supprimer Favoris» …..…...52

Figure 34 : Diagramme de classe de conception du cas «Créer Compte»……….………….52

Figure 35 : Diagramme de classe de conception du cas «S’authentifier»……...……………53

Figure 36 : Diagramme de classe de conception du cas «Gérer favoris»………..………….53

Figure 37 : Interfaces authentification utilisateur …………………………………..……….55

Figure 38 : Raffinement du cas d’utilisation « S’authentifier »………………………...…….57

Figure 39 : Raffinement du cas d’utilisation « Gérer médecin »………………………..……58

Figure 40: Diagramme de classe d’analyse du cas « S’authentifier»…………….……..……61

Figure 41 : Diagramme de classe d’analyse du cas « Gérer médecin»…….………………..61

Figure 42 : Diagramme de séquence du cas d’utilisation « S’authentifier»…………………62

Figure 43 : Diagramme de séquence du sous cas d’utilisation « Consulter médecin»………63

Figure 44 : Diagramme de séquence du sous cas d’utilisation « Modifier médecin»….……63

Figure 45 : Diagramme de séquence du sous cas d’utilisation « Ajouter médecin»……..….64

Figure 46 : Diagramme de séquence du sous cas d’utilisation « Supprimer médecin»…..….64

Figure 47: Diagramme de classe de conception du cas d’utilisation «S’authentifier»…..…65

Figure 48 : Diagramme de classe de conception du cas d’utilisation «Gérer Médecin»…...65

Figure 49 : interface du cas d’utilisation «S’authentifier» login et mot de passe erronés.......67

Figure 50 : interface du cas d’utilisation «Consulter médecin» …………………………..…62

Figure 51 : interface du cas d’utilisation «Modifier médecin» ………………..……………..62

Figure 52 : interface du cas d’utilisation «Ajouter médecin» …………..……………………63

Figure 53 : interface du cas d’utilisation «Supprimer médecin»…………..…………………63

Figure 54 : Android Studio ……………………………………………………………..……71

Figure 55 : Java………………..……………………………………………………………. 72

Figure 56 : Wamp server………….………………………………………………………… 72

Figure 57 : Filezilla ……………………………………………………………………….… 73

Figure 58 : CodeIgniter ……………….……………………………………………………. 73

Figure 59 : PHP ……………………….……………………………………………………. 73

Liste des Tableaux

Tableau 1 : Tableau comparatif des solutions existantes …..………………………….……17

Tableau 2 : Backlog du produit …………………………………………………….……..23

Tableau 3 : Planification des Sprints ………………………………….…………………….27

Tableau 4 : Raffinement du cas d’utilisation « Lister tous les médecins » ……..………….29

Tableau 5 : Raffinement du cas d’utilisation « Filtrer médecins »………………..…………30

Tableau 6 : Raffinement du cas d’utilisation « Localiser Médecin »……………..…………31

Tableau 7 : Raffinement du cas d’utilisation « Prendre RDV par Mail»………...…………..32

Tableau 8 : Raffinement du cas d’utilisation « Appeler médecin»………………..…………34

Tableau 9 : Table Médecin …………………………………………………….……….……40

Tableau 10 : Raffinement du sous cas d’utilisation « Créer Compte »…………………...…..43

Tableau 11 : Raffinement du sous cas d’utilisation « Rechercher Compte»…………….…..45

Tableau 12 : Raffinement du sous cas d’utilisation « Ajouter Médecin »……………………46

Tableau 13 : Raffinement du sous cas d’utilisation « Consulter Favoris »…………………..46

Tableau 14 : Raffinement du sous cas d’utilisation « Supprimer Favoris »………………….47

Tableau 15 : Table Utilisateur………………………………………………………………...54

Tableau 16 : : Table Favoris………………………………………………………………….54

Tableau17 : Raffinement du cas d’utilisation «S’authentifier»……………………..………..58

Tableau 18 : Raffinement du sous cas d’utilisation « Consulter médecin»………………….59

Tableau 19: Raffinement du sous cas d’utilisation « Modifier médecin»……………………59

Tableau 20 : Raffinement du sous cas d’utilisation « Ajouter médecin»……………………60

Tableau 21 : Raffinement du sous cas d’utilisation « Supprimer médecin»…………………60

Tableau 22 : Table Gouvernorat……………………………………………………………...66

Tableau 23 : Table Spécialité…………………………………………………………………66

Tableau 24 : Table Coordonnées Map …………………………………………………….....66

Introduction générale :

La communication et l’information sont devenues des éléments indispensables dans la vie de tous les jours. En quelques années, la recherche d’informations dans une encyclopédie ou la recherche d’un numéro de téléphone dans les pages jaunes a transité vers de simples clics sur une page internet. Actuellement, à tous moments et en tous lieux, ces renseignements sont accessibles du bout des doigts par l’intermédiaire de téléphones ou de pads.

Il est important pour tout individu de se tenir au courant de ce qui se passe autour de lui et d’avoir facilement et rapidement accès aux informations dont il a besoin quand il en a besoin. Le développement d’application embarquée devient alors une solution idéale.

En effet , les application s’avèrent plus commode dans leur utilisations que les sites mobiles car elles offrent une interface plus harmonieuse et plus spécifique , sans oublier la rapidité d’accès à l’information ainsi que la possibilité de profiter de plusieurs fonctionnalités novatrices, ludiques et intuitives qui repoussent les limites de l’expérience utilisateur comme le GPS , l’appareil photo , vidéos, réalité augmentée…

Il est important pour une société comme Get Wireless de se tenir proche de ses clients en proposant toujours des solutions à jour, adaptées et de bonne qualité d’où l’idée de développer une application sous Android permettant de trouver un médecin dans la spécialité demandée en quelques clics. Certes connaître l’entreprise s’avère indispensable afin de travailler dans de bonnes conditions et de fournir un résultat probant aux attentes de celle-ci. C’est pourquoi, la première partie du rapport comportera une présentation de Get Wireless, de ses activités, de ses objectifs et de ses attentes au sein de mon stage. N’étant pas en équipe, j’ai dû effectuer ma mission de façon autonome. De plus, une phase de recherche et d’apprentissage était donc nécessaire pour comprendre le fonctionnement d’Android. Le détail de mes objectifs et de mes recherches sera relaté dans une deuxième partie. De l’élaboration à la production, je présenterai le déroulement du développement Android dans une troisième partie. J’exposerai ensuite l’application « Médecin » en quatrième partie. J’insisterai particulièrement sur les phases de recherches, les prises de décisions, les problèmes rencontrés et résolus, et sur la part d’innovation ajoutée par rapport aux applications existantes aussi bien que les perspectives de ce projet.

Chapitre 1

Etude préalable

1. Introduction :

Dans ce chapitre nous mettrons le travail dans son contexte général. Dans un premier lieu, nous présenterons l’objectif même de ce mémoire puis nous passerons à la présentation de l’organisme accueillant ce stage. Ensuite, nous introduirons le sujet et nous aborderons la problématique ainsi que la solution proposée.

1. Objectif du PFE :

Ce projet de fin d’étude de la 3éme année informatique appliquée à la gestion a été effectué sous l’encadrement de Mr.Ahmed Badreddine professeur à l’ISG et Mr.Ahmed Belghith ingénieur à Get Wireless.

Il représente une belle occasion pour mettre en pratique toutes les connaissances et capacités analytiques, techniques et relationnelles acquises durant notre parcours académique dans un cadre professionnel.

En effet lors du stage, nous sommes amenés à prendre des décisions, respecter des deadlines et interagir avec les différents départements afin de mieux cerner les besoins et produire une application de qualité. Le projet a pour but de nous initialiser à la plateforme de développement mobile sous Android et de tirer profit des gadgets que possèdent les Smartphones de nos jours. Les principaux gadgets qui nous seraient utiles pour élaborer ce projet sont le mailing et le GPS. Etant donnée l'émergence de technologie mobile, beaucoup d'applications ont été développées dans divers domaines. Cependant nous avons pensé comment utiliser ces gadgets pour aider chaque individu à trouver un médecin en cas de besoin à partir de son Smartphone.

Ce type d’application métier s’avère très utile non seulement afin de subvenir aux besoins des patients mais il peut aussi représenter un réel avantage pour les médecins. Il s’agit de rapprocher le médecin de ses patients et de le rendre plus disponible et surtout accessible en cas de besoin.

1. Présentation de l’organisme :
2. Identité :

Get Wireless est une société à responsabilité limitée, elle est née en 2001 pour répondre à des challenges d'innovation dans un environnement technologique évolutif et révolutionnaire.  
  
Dotée d'un capital de 165.000 DT et d'un potentiel humain qualifié, GET Wireless a développé une expertise technique reconnue dans les différentes phases de conception, déploiement, optimisation et exploitation des réseaux radio en plus d'un département SAV (Services à valeurs ajoutées) très performant répondant aux exigences de communication occurrent sur le marché des TICs (technologies de l'information et de la communication).

1. Secteur d’activité :

Get Wireless agit dans le secteur des technologies de communication et de l’information, elle touche à plusieurs domaine :

* [Ingénierie](http://www.getwireless.com.tn/ingenierie/ingenierie.php) :
  + Déploiement et optimisation de réseaux cellulaires.
  + Mesure drive test.
  + Intégration des systèmes
  + Vsat.
* [Audit, Conseil et formation](http://www.getwireless.com.tn/ingenierie/audit-conseil-formation.php) :
  + Audit physique et fonctionnel des réseaux mobiles
  + Audit de processus de travail
  + Réseaux et services de télécommunication.
* Fourniture d’équipements :
* Solutions de camouflage
* Chaîne de mesure drive test 2G et 3G.
* Intégration des solutions IT :

GET Wireless a pour ambition de s'imposer sur le marché régional en tant qu'acteur majeur, dans les métiers du conseil et d'intégration des solutions IT dans le domaine des télécommunications. Par sa politique proche de ses clients, pour mieux les écouter et répondre à leurs besoins en évolution constante, la société établit des partenariats stratégiques avec les acteurs majeurs dans le domaine des Télécommunications afin d'élaborer des solutions adaptées.

1. Problématique :

Durant ce stage, il nous a été demandé de faire la conception, le développement et l’intégration d’une application Android qui permet aux utilisateurs de trouver un médecin en cas de besoin, celui-ci pouvant être recherché par nom, spécialité, gouvernorat…

L’origine de ce sujet était une simple idée pour fournir des informations concises et pertinentes sur les médecins en Tunisie facilement accessible en cas de besoin. Au fur et à mesure cette idée a évolué pour permettre aux utilisateurs d’avoir la possibilité de géolocaliser le médecin en question, d’avoir les directions pour s’y rendre et même de prendre un rendez-vous par mail…

Le sujet est certes intéressant car l’application dont nous parlons peut avoir une vraie valeur ajoutée dans la vie de chacun d’entre nous et ses perspectives d’évolution sont énormes surtout avec le progrès permanent de la technologie. D’autre part pour la réalisation d’un tel projet, nous avons été confrontés à plusieurs difficultés techniques et décisionnelles. Mis à part le développement proprement dit de l'application, la première étape consistait à nous familiariser avec l'environnement Android, puis de choisir les outils conviviaux et envisageables à l'aboutissement du projet. Par la suite, nous avons entamé la modélisation et le développement de l’application, pour cela nous avons opté pour la méthode agile Scrum afin d’avoir une meilleure adaptation aux besoins évolutifs et aussi pour garantir le respect des délais. Ensuite, nous passons à la création d’un serveur web pour stocker les données dans une base de données facilement accessible, nous avons également pensé à implémenter Les gros calculs sur ce serveur web afin d'éliminer ou de diminuer les contraintes matérielles limitant notre application sur les Smartphones ou tablettes.

Afin de garantir le succès d’une telle application, il faut surtout insister sur la pertinence des informations et la rapidité du service proposé c’est pourquoi nous avons pensé à mettre en place une application web pour gérer les CRUD de notre base de données et ainsi faciliter sa mise à jour afin de garantir la précision des informations fournies.

Nous avons également pensé à garantir l’innovation, dans notre cas la prise de rendez-vous par mail, qui éliminera l’obstacle des horaires à respecter car la prise de rendez-vous sera désormais possible à tout moment. Du coté médecin, il suffit de fournir une adresse email que la secrétaire peut consulter et qui lui permettra d’organiser les rendez-vous par la suite.

1. Langage et méthodologies de conception :

La méthodologie est une démarche organisée rationnellement pour aboutir à un résultat.

Pour bien conduire notre projet et nous assurer du bon déroulement des différentes phases, nous avons opté pour la méthode agile Scrum comme méthodologie de conception et de développement.

« *Scrum signifie mêlée au rugby. Scrum utilise les valeurs et l’esprit du rugby et les adapte aux projets de développement.*»[1] Scrum est issu des travaux de deux des signataires du Manifeste Agile, Ken Schwaber et Jeff Sutherland, au début des années 1990.Il appartient à la famille des méthodologies itératives et incrémentales et repose sur les principes et les valeurs agiles. Le plus souvent, les experts de Scrum, même ses fondateurs, le décrivent comme un cadre ou un patron de processus orienté gestion de projet et qui peut incorporer différentes méthodes ou pratiques d’ingénierie. S’il est difficile de définir la nature de Scrum, sa mise en place est beaucoup plus simple et peut être résumée par la Figure 1.

Le principe de base de Scrum est le suivant :

* Dégager dans un premier lieu le maximum des fonctionnalités à réaliser pour former le backlog du produit
* En second lieu définir les priorités des fonctionnalités et choisir lesquelles seront réalisé dans chaque itération,
* Par la suite focaliser l'équipe de façon itérative sur l’ensemble de fonctionnalités à réaliser, dans des itérations appelées Sprints,
* Un Sprint aboutit toujours sur la livraison d’un produit partiel fonctionnel appelé incrément.



Figure 1 : Le processus Scrum[1]

En effet, comme le représente ce schéma, avec SCRUM le projet est décomposé en incréments qui traitent les fonctionnalités relevées dans le Backlog une à une d’une manière itérative jusqu’à l’aboutissement du produit final. Grâce à la mêlée quotidienne nous pouvons évaluer le progrès du travail accompli et nous focaliser sur les tâches qui restent à accomplir.

Après le choix de la méthodologie, nous avons besoins d’un langage de modélisation unifié pour la modélisation de notre projet. Pour concevoir notre système, nous avons choisi UML2 comme langage de modélisation.

Notre choix s'est basé sur les points forts de ce langage notamment sa standardisation et les divers diagrammes qu’il propose. Aussi UML présente le meilleur outil pour schématiser des systèmes complexes sous un format graphique et textuel simplifié et normalisé.

Maintenant en ce qui concerne l’environnement de développement, pour l’application Android dans un premier lieu nous avons choisi de travailler avec éclipse et le plug in ADT spécialement conçu par Google afin d’offrir un puissant environnement intégré pour permettre un développent mobile souple Mais suite à des complications de compatibilités de versions de l’ADT et de l’SDK nous avons pris la décision de changer d’environnement et de passer à Android Studio. Pour la partie Web nous avons utilisé Notepad++ et le FrameWork CodeIgniter qui grâce aux fonctionnalités de Grocery CRUD permet de faciliter la gestion des CRUD.

1. Etude de l’existant :

Le potentiel du marché de l’e-santé ne cesse de croitre, en témoignent le nombre croissant d’outils et d’applications mis à disposition sur le marché. Ce potentiel se mesure notamment à deux principaux facteurs, le premier étant le taux d’acquisition croissant des smartphones et tablettes chez le Grand Public comme chez les professionnels de santé. Le second est la croissance et la multiplicité des applications métiers destinées aux professionnels de santé.

Deux grandes typologies d’applications santé sont à distinguer sur les Application Stores :

* les applications Grand Public surfant sur la vague fitness et bien-être (suivi des calories, autodiagnostic, relaxation…)
* les applications dédiées aux professionnels de santé.

Nous allons nous intéresser de plus près au sujet en examinant les applications déjà existantes qui répondent au même besoin ou presque pour en dégager les atouts clés et aussi les problèmes rencontrés par l’utilisateur.

1. Santé Tunisie :

Cette application permet de localiser et de contacter les services médicaux les plus proches ( médecin , hôpital, pharmacie . . . ) .

Figure 2 : Interfaces de l’application Santé Tunisie

Cette application est très facile à utiliser et offre une interface agréable mais la plupart des fonctionnalités ne marchent pas : l’appel n’est pas pris en charge, le filtre lors de la recherche n’est pas fonctionnel, il n y a pas de visualisation sur carte elle fournit uniquement l’adresse.

1. Dot Medical :

Elle fournit à l'utilisateur des informations sous forme d'un annuaire médical pour la Tunisie. Ces références sont classées selon les gouvernorats et les spécialités.

Figure 3 : Interfaces de l’application Dot Medical

Dot Medical offre la possibilité de chercher un médecin ou une pharmacie mais avec une difficulté de le repérer sur la carte. Il est à noter également que les informations fournies ne sont pas pertinentes.

1. Sehatuk :

Elle permet de localiser les services médicaux de proximité dans 4 pays le Maroc, l’Algérie, la Tunisie et le Sénégal et disponible en Arabe, français et anglais.

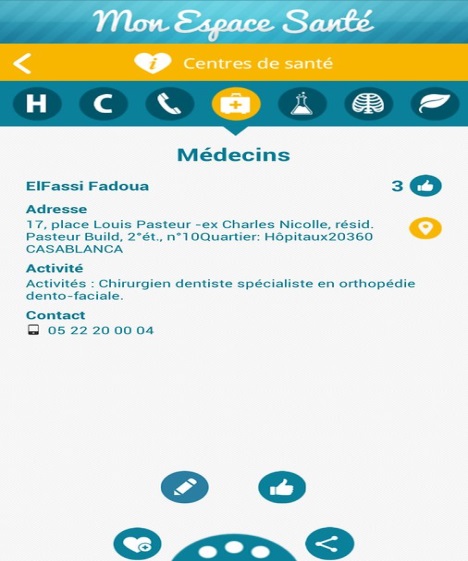
  

Figure 4 : Interfaces de l’application Sehatuk

Sehatuk présente des informations sur les services médicaux demandés mais on relève la difficulté de repérage sur la carte ainsi qu’une lenteur d’exécution et aussi le grand nombre d’autorisation d’accès que l’application demande qui s’avère très gênant pour l’utilisateur.

1. Solution proposée :

Après avoir examiné la situation actuelle, nous allons essayer de proposer une application mobile sous Android qui traite les défaillances qu’on a relevées qui propose une solution facilement accessible, rapide tout en veillant à apporter des nouveautés qui peuvent distinguer notre travail et qui sera soutenu par une application Web assurant et facilitant son administration afin d’être la plus pertinente possible.

Nous vous proposons ce tableau comparatif entre les services fournis par les applications qu’on a présenté dans l’étude de l’existant et celle que nous proposons.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Solutions** | **Santé Tunisie** | **Dot It Medical** | **Sehatuk** | **La notre** |
| Recherche Médecin | * Sans Filtre | * Filtre spécialités | * Filtre spécialités | * Filtre spécialités * Filtre Gouvernorat * Filtre CNAM * Filtre Sexe |
| Géolocalisation | Non | Repérage difficile | Repérage difficile | Google Maps |
| Données | Pas de MAJ | Pas de MAJ | Pas de MAJ | BD à jour |
| Contact médecin | * Fournit contact | * Fournit contact | * Fournit contact | * Fournit contact * Permet contact |
| Création compte | Non | Non | Non | Oui |
| Gestion Favoris | Non | Non | Non | Oui |

Tableau1 : Tableau comparatif des solutions existantes

D’après ce tableau, nous remarquons que les services que proposent les applications mobiles surtout dans le domaine de la santé en Tunisie sont assez limitées d’où le potentiel énorme de ce secteur surtout en pleine émergence.

Ce système peut avoir un grand apport aussi bien sur le plan technique que sur le plan fonctionnel.

1. Apport sur le plan technique :

Cette application permet de centraliser des données utiles pour l’utilisateur et lui permettre d’y accéder rapidement tout en veillant à la sécurité et la pertinence des informations échangées.

1. Apport sur le plan fonctionnel :

Sur le plan fonctionnel, cette application s’avère être un outil efficace et pratique qui offre principalement deux grands avantages : Le gain significatif de temps lors de la recherche d’un médecin, tout en fournissant toutes les informations sur celui-ci et d’un autre côté, il permet la prise, le changement ou l’annulation du rendez-vous avec le médecin par mail, à tout moment indépendamment des horaires de travail.

1. Conclusion

Tout au long de ce chapitre nous avons présenté le sujet de notre travail dans son contexte général pour pouvoir l’aborder de la meilleure manière possible et assurer une compréhension globale de l’objectif visé.

Chapitre 2

Le Backlog

1. Introduction :

Dans le chapitre précèdent, nous nous sommes focalisés sur l'étude approfondie de

la problématique afin de la mettre dans son cadre théorique. Dans ce chapitre nous allons dégager les principales fonctionnalités du système à mettre en place et définir son architecture globale et aussi présenter la planification de notre projet.

1. Capture des besoins :
2. Identification des acteurs :

L'analyse de l'application débute par la recherche des acteurs.

« Un acteur représente l’abstraction d’un rôle joué par des entités externes (utilisateur, dispositif matériel ou autre système) qui interagissent directement avec le système étudié. »[2]

En d’autre termes un acteur est une entité physique (personne) ou abstraite (logiciel) capable d’utiliser le système afin de répondre à un besoin bien définit. Dans notre cas, nous avons choisi de nous adresser au grand public pour offrir nos services, nous dégageons alors un seul acteur qui va interagir avec l’application Android qui est :

* L’utilisateur de l’application.

D’autre part pour assurer le back office de la partie mobile, nous avons mis en place une application Web qui à son tour aura un seul Acteur qui est :

* L’administrateur.

1. Besoins Fonctionnels :

Les besoins fonctionnels ou les cas d’utilisations en terme d’UML peuvent être définis comme suit : « *Un cas d’utilisation (use case) représente un ensemble de séquences d’actions réalisées par le système et produisant un résultat observable intéressant pour un acteur particulier.* » [2]

Un cas d’utilisation est une suite d’actions effectuées par le système afin de répondre à une demande d’un utilisateur (acteur). Notre application Android va permettre de chercher un médecin sur le territoire tunisien, selon des critères précis. Elle doit donc répondre aux besoins suivants :

* Chercher Médecin.
* Localiser Médecin.
* Prendre un rendez-vous par mail.
* Appeler Médecin.
* Gérer les médecins favoris.

Quant à la partie Web, nous relevons les fonctionnalités suivantes :

* S’authentifier.
* Gérer Médecins.
* Gérer Spécialités.
* Gérer Gouvernorat.
* Gérer coordonnées Map.

1. Besoins non Fonctionnels

* Ergonomie et souplesse : Les applications doivent offrir une interface conviviale et ergonomique exploitable par l'utilisateur en envisageant toutes les interactions possibles à l'écran du support tenu.
* Rapidité du service : Les applications doivent optimiser les traitements pour avoir un temps de consultation et de génération de schéma raisonnable.
* Sécurité : Le système doit garantir la sécurité des données de la base aussi bien que celles des utilisateurs sans les étouffer.
* **Maintenabilité et scalabilité :** Les codes des applications doivent être lisibles et compréhensibles afin d'assurer leurs états évolutifs et extensibles par rapport aux besoins du marché.

1. Planning du traitement des cas d’utilisation :

Une fois les cas d’utilisation identifiés, il est indispensable d’établir un ordre afin de développer ci, en tenant compte des priorités ainsi que de la dépendance des cas d’utilisation.

Dans une première partie nous allons mettre au point la partie Android avec ses fonctionnalités puis nous passerons à la partie web pour la gestion de la base de données.

1. Pilotage du projet avec SCRUM

4.1 Les fonctionnalités du Backlog :

Le Backlog est un artéfact très important dans SCRUM. C’est l’ensemble des caractéristiques fonctionnelles ou techniques qui constituent le produit souhaité[2].

Nous allons les décrire en détails dans le tableau qui suit :

|  |  |  |
| --- | --- | --- |
| **Fonctionnalité** | **Acteur** | **Description** |
| Chercher Médecin | Utilisateur | L’utilisateur peut chercher un médecin dans toute la liste des médecins de la base de données. |
| Filtrer Médecin | Utilisateur | L’utilisateur peut Choisir des critères de recherche selon la spécialité de médecin , le gouvernorat , le sexe et l’affiliation à la CNAM. |
| Afficher profil Médecin | Utilisateur | L’utilisateur peut afficher toutes informations relatives à un médecin donné : photo , adresse , horaire de travail… |
| Appeler Médecin | Utilisateur | L’utilisateur peut directement lancer son appel depuis l’application. |
| Prise de RDV par mail | Utilisateur | L’utilisateur peut adresser un mail pour fixer la date et l’heure du RDV. |
| Géolocaliser Médecin | Utilisateur | L’utilisation peut voir la position géographique exacte du médecin sur la carte en 3 modes différents. |
| S’authentifier | Utilisateur | L’utilisateur peut enregistrer sa session afin de retrouver les données qu’il voudra enregistrer |
| Ajouter Médecin Favoris | Utilisateur | L’utilisateur peut ajouter un médecin à la liste des favoris pour y accéder plus rapidement. |
| Supprimer Médecin Favoris | Utilisateur | L’utilisateur peut supprimer un médecin de la liste des favoris. |
| S’authentifier1 | Administrateur | L’administrateur doit s’authentifier afin d’accéder à sa session. |
| Ajouter Médecin | Administrateur | L’administrateur peut ajouter un médecin si nécessaire. |
| Supprimer Médecin | Administrateur | L’administrateur peut supprimer un médecin si nécessaire. |
| Modifier Médecin | Administrateur | L’administrateur peut modifier un médecin si nécessaire. |
| Ajouter Spécialité | Administrateur | L’administrateur peut ajouter une spécialité si nécessaire. |
| Supprimer Spécialité | Administrateur | L’administrateur peut supprimer une spécialité si nécessaire. |
| Modifier Spécialité | Administrateur | L’administrateur peut modifier une spécialité si nécessaire. |
| Ajouter Gouvernorat | Administrateur | L’administrateur peut ajouter un gouvernorat si nécessaire. |
| Modifier Gouvernorat | Administrateur | L’administrateur peut modifier un gouvernorat si nécessaire. |
| Supprimer Gouvernorat | Administrateur | L’administrateur peut supprimer un gouvernorat si nécessaire. |
| Ajouter coordonnées Map | Administrateur | L’administrateur peut ajouter des coordonnées si nécessaire. |
| Modifier coordonnées Map | Administrateur | L’administrateur peut modifier des coordonnées si nécessaire. |
| Supprimer coordonnées Map | Administrateur | L’administrateur peut supprimer des coordonnées si nécessaire. |

Tableau 2 : Backlog du produit

* 1. Diagramme de cas d’utilisation

La solution à mettre en place consiste à développer une application mobile qui permet à son utilisateur de chercher un médecin dans une spécialité et une région données ainsi que de le géolocaliser et aussi le contacter par téléphone ou prendre rendez-vous à tout moment par mail. Ainsi qu’une application Web d’administration qui permet à l’administrateur de gérer facilement toutes les tables de la base de données.

Une étude détaillée du système permet de dégager les besoins qui définissent ces fonctionnalités principales. Ces besoins sont modélisés sous forme de cas d'utilisation avec les diagrammes suivants :

* Diagramme de cas d’utilisation pour l’application mobile :

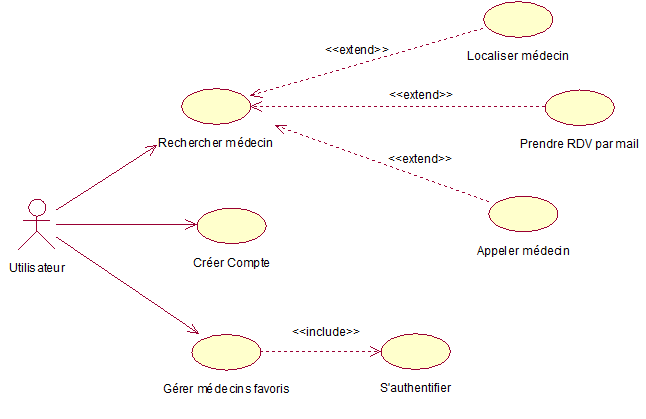


Figure 5 : Diagramme de cas d’utilisation pour l’application Android

Selon ce diagramme, l’utilisateur de notre application peut rechercher un médecin sans authentification préalable, une fois le médecin choisi, il peut le géolocaliser sur la carte, l’appeler ou lui envoyer un mail pour la prise de rendez-vous.

Afin de pouvoir enregistrer un médecin comme favoris ou l’enlever de la liste des favoris, l’authentification est alors nécessaire.

Nous avons fait ce choix afin d’alléger l’utilisation de notre application et de la rendre la plus disponible et efficace possible.

* Diagramme de cas d’utilisation pour la partie Web :

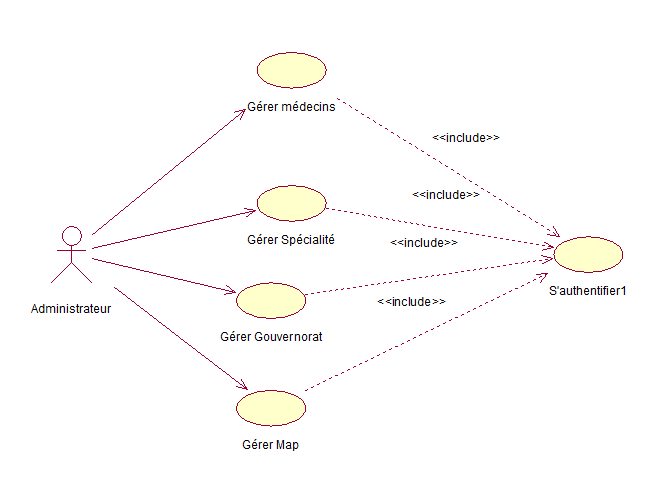
****

Figure 6 : Diagramme de cas d’utilisation pour l’application Web

Dans la partie web, afin de garantir une sécurité maximale des données, aucune action n’est permise sans authentification. Car le rôle principal de cette application et d’assurer l’efficacité et la pertinence de la partie mobile qui demeure notre application vitrine.

Dans les figures ci-dessus nous avons proposé une vue globale du système représentée par des cas d'utilisation généraux des deux applications que nous proposons. Dans les chapitres qui suivent nous allons analyser plus en détail chacun de ces cas.

* 1. Architecture

La norme IEEE 1471 définit l'architecture logicielle comme l'organisation fondamentale d'un système représenté par ses composants, la relation des composants du système les uns avec les autres et avec l'environnement, et les principes et les directives pilotant leur conception et leur évolution[4].

Il est indispensable de penser à l’architecture du système dès le départ du projet car c’est un élément maitre qui accompagne un travail de la phase de l’analyse des besoins jusqu’à sa réalisation. Son avantage principal est qu’il permet de garantir la qualité du produit dès le départ, c’est pourquoi nous allons définir l’architecture de base de nos applications comme suit :

* Partie Android :

Pour alléger l’application et surtout rendre plus rapide le temps des traitements, nous avons conçus des web services pour assurer la communication entre l’application avec la base de données. En effet la partie contrôle de notre application mobile qui est notre partie client, va faire appel à des web services que nous avons mis en place du coté serveur. Ainsi l’application mobile sera la consommatrice (Consumer) de ces services et par conséquent le fournisseur (Provider) sera le serveur mis en place.

La communication entre application, base de données et serveur se fait via ces services web. Il s’agit donc d’une une architecture orientées service.

* Partie Web :

Il s’agit d’une application d’administration, pour cela nous avons pensé à une architecture 3-tiers (client, serveur, base de données) pour bien organiser la gestion de celle-ci.

* 1. Planification des sprints

L’étape de la planification est certes très importante pour une meilleur organisation et gestion du projet. Nous avons donc décidé de travailler deux Releases :

Le premier traitant l’application Android et qui contiendra 3 Sprints, Le second pour l’application Web et qui contiendra un seul sprint.

Nous nous sommes fixés la durée maximale de 3 semaines pour un Sprint surtout comme c’est un travail en monôme.

Notre travail suivra les grandes lignes suivantes :

|  |  |  |  |
| --- | --- | --- | --- |
| Release 1 : Application Android | | | Release 2 : Application Web |
| Sprint 1  De 01/02 à 22/02 | Sprint 2  De 23/02 à 15/03 | Sprint 3  16/03 à 05/04 | Sprint1  06/04 à 26/04 |
| Chercher médecin  Afficher profil | Géolocaliser médecin  Appeler médecin  Envoyer Mail | S’authentifier  Gérer favoris | Authentification  Gestion Médecins  Gestion spécialité  Gestion Gouvernorat  Gestion Map |

Tableau 3 : Planification des Sprints

1. Conclusion :

Nous avons, tout au long de ce chapitre, présenté notre sujet afin de mieux cerner son champ d’actions. Ainsi, nous avons pu dégager les différentes fonctionnalités auxquelles le système doit répondre ainsi que les différents composants de l'architecture des applications.

Dans le chapitre suivant nous entamons le développement du premier release en suivant les lignes directrices que nous nous sommes fixées.

Chapitre 3

Release 1

1. Introduction

Le terme release désigne ici une version de notre application constituée d'une suite d'itérations (sprint) [3] qui se terminent quand les incréments de ces derniers construisent un produit présentant suffisamment de valeur à l’utilisateur.

Dans ce chapitre, nous allons nous intéresser à la partie Android du système, de l’analyse des besoins jusqu’au test. Nous traiterons en détails chacun des cas d’utilisation préalablement présentés.

1. Le premier Sprint

Selon la planification établie, le premier Sprint porte principalement sur le cas d’utilisation «  Rechercher de médecin » que nous allons décortiquer par la suite.

L’objectif de cette première itération est de fournir une interface qui va permettre à l’utilisateur d’effectuer une recherche de médecin et d’afficher le résultat de cette recherche depuis son appareil mobile.

* 1. Analyse :
     1. Raffinement des cas d’utilisations

1. Cas d’utilisation «  Rechercher médecin » :

Dans cette activité nous allons raffiner les cas d’utilisation prioritaire et les décrire en détail afin de mieux visualiser notre application.

Détailler le cas d’utilisation « Rechercher médecin » revient à détailler ses sous cas d’utilisation à savoir :

* Lister tous les médecins pour une recherche générale non conditionné, dans ce cas le système va retourner la liste de tous les médecins existants.
* Filtrer médecins pour une recherche spécifique répondant à différents critères dans notre cas on a choisi : la spécialité du médecin, le gouvernorat où se trouve son cabinet, l’affiliation de celui-ci à la CNAM et son sexe. Le système va alors récupérer les médecins qui répondent à ces critères uniquement.

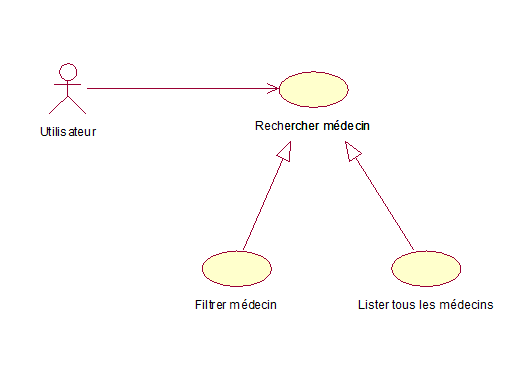


Figure 7 : Raffinement du cas d’utilisation « Rechercher médecin »

L’utilisateur peut effectuer sa recherche soit en consultant directement la liste de tous les médecins existants, soit après avoir déterminé des critères pour sa recherche pour aboutir à un résultat plus précis. D’où les descriptions suivantes :

* **Description du sous Cas d’Utilisation « Lister tous les médecins »**

|  |  |
| --- | --- |
| Sous cas d’utilisation: | Lister tous les médecins |
| Acteur : | Utilisateur |
| Pré-condition: | L’application mobile est lancée avec succès. |
| Post-condition: | Liste de tous les médecins affichée. |
| Description de scénario de base : | **1.** L’utilisateur accède à la page de recherche.  **2.** L’utilisateur clique sur «Rechercher tous »  **3.** Le système affiche la liste de tous les médecins. |
| Exception : | Echec du lancement de l’application. |

Tableau 4 : Raffinement du cas d’utilisation  « Lister tous les médecins »

* **Description du sous Cas d’Utilisation « Filtrer médecins »**

|  |  |
| --- | --- |
| Sous cas d’utilisation: | Filtrer médecins |
| Acteur : | Utilisateur |
| Pré-condition: | L’application mobile est lancée avec succès. |
| Post-condition: | Liste des médecins, correspondant aux critères de l’utilisateur, affichée. |
| Description de scénario de base : | **1.** L’utilisateur accède à la page de recherche.  **2.** L’utilisateur choisit la spécialité du médecin recherché.  **3.** L’utilisateur choisit le gouvernorat ciblé.  **4.** l’utilisateur précise s’il veut que le médecin qu’il cherche soit affilié à la CNAM.  **5.** l’utilisateur choisit le sexe du médecin recherché.  **6.** Le système affiche la liste de tous les médecins correspondant aux critères choisis. |
| Exception : | - S’il n y a aucun médecin correspondant, un massage s’affiche « Pas de médecin trouvé». |

Tableau 5 : Raffinement du cas d’utilisation  « Filtrer médecins »

1. Cas d’utilisation « Localiser médecin »

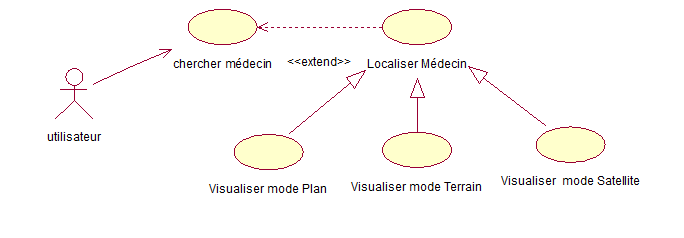


Figure 8 : Raffinement du cas d’utilisation « Localiser médecin »

La localisation d’un message n’est accessible qu’une fois celui-ci est choisi par l’utilisateur, et grâce à l’API (Application Programming Interface) de Google la visualisation se fait via Google Maps qui offre 3 mode de visualisation soit en mode plan , terrain ou satellite .

L’utilisateur peut facilement basculer d’un mode à un autre en cliquant sur le mode qui lui convient le mieux.

* **Description du Cas d’Utilisation « Localiser médecin »**

|  |  |
| --- | --- |
| Cas d’utilisation: | Localiser Médecin |
| Acteur : | Utilisateur |
| Pré-condition: | Liste de médecins recherchée affichée avec succès. |
| Post-condition: | L’application affiche la position géographique du cabinet du médecin choisi sur la carte. |
| Description de scénario de base : | **1.** L’utilisateur choisit un médecin dans la liste affiché.  **2.** L’utilisateur clique sur géolocaliser  **3.** le système affiche la position sur la carte en 3 modes disponible (terrain, plan et stellite).  **4.** L’utilisateur peut basculer d’un mode à un autre en cliquant sur le mode qu’il veut afficher.  **5.** Le système affiche la même position sous le mode choisi. |
| Exception : | Pas de médecin affiché dans la liste. |

Tableau 6 : Raffinement du cas d’utilisation  « Localiser Médecin »

1. Cas d’utilisation « Prendre RDV par Mail »

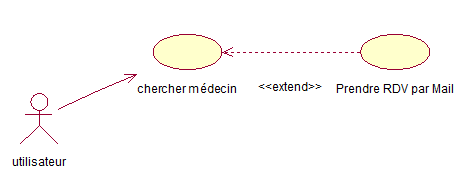


Figure 9 : Raffinement du cas d’utilisation « Prendre RDV par Mail»

La prise de rendez-vous est désormais une fonctionnalité possible sans que l’utilisateur soit obligé de s’authentifier. Dès que cette option est choisie la fonction Mail du téléphone est activé et l’adresse email de l’émetteur est automatiquement récupérer depuis son compte Google qu’il utilise dans l’application store.

* **Description du Cas d’Utilisation « Prendre RDV par Mail »**

|  |  |
| --- | --- |
| Cas d’utilisation: | Prendre RDV par Mail |
| Acteur : | Utilisateur |
| Pré-condition: | Liste de médecins recherchée affichée avec succès. |
| Post-condition: | Le Mail est envoyé avec succès au médecin choisi. |
| Description de scénario de base : | **1.** L’utilisateur choisit un médecin dans la liste affiché.  **2.** L’utilisateur clique sur Mail  **3.** Le système affiche des champs de saisies relatifs au titre et au corps du mail.  **4.** L’utilisateur saisit son texte.  **5.** L’utilisateur clique Envoyer.  6. Le mail est envoyé avec succès. |
| Exception : | Pas de médecin affichés sur la liste. |

Tableau 7 : Raffinement du cas d’utilisation  « Prendre RDV par Mail»

1. Cas d’utilisation « Appeler Médecin »

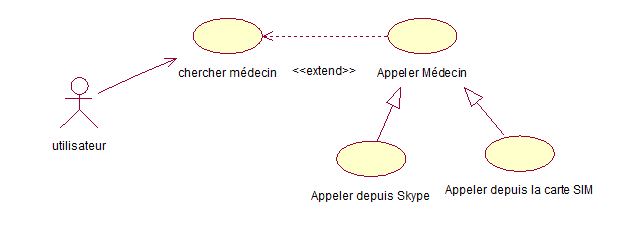


Figure 10 : Raffinement du cas d’utilisation « Appeler Médecin»

Comme le montre la figure ci-dessus, l’utilisateur de peut appeler un médecin qu’après avoir effectué une recherche. Il peut alors choisir parmi les possibilités qui s’offrent à lui ; ou bien passer l’appel sur voie téléphonique à travers sa carte SIM ou choisir de passer un appel sur Skype.

* **Description du Cas d’Utilisation « Appeler médecin »**

|  |  |
| --- | --- |
| Cas d’utilisation: | Appeler médecin |
| Acteur : | Utilisateur |
| Pré-condition: | Liste de médecins recherchée affichée avec succès. |
| Post-condition: | L’application lance un appel téléphonique au médecin choisi. |
| Description de scénario de base : | **1.** L’utilisateur choisit un médecin dans la liste affiché.  **2.** L’utilisateur clique sur Appeler.  **3.** le système propose deux options pour passer l’appel : par Skype ou depuis la carte SIM.  **4.** L’utilisateur choisit l’option qui lui correspond le mieux en cliquant sur l’option pour laquelle il a opté.  **5.** L’appel est lancé avec succès. |
| Exception : | Pas de médecins affichés sur la liste. |

Tableau 8 : Raffinement du cas d’utilisation  « Appeler médecin»

* + 1. Diagramme de classes d’analyse :

1. Cas « Rechercher médecin » :

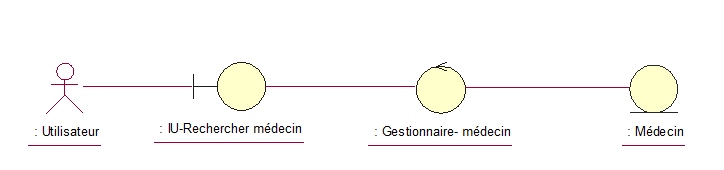
****

Figure 11 : Diagramme de classe d’analyse du cas « Rechercher médecin »

La figure 13 illustre le diagramme de classe d’analyse qui montre les différentes classes participantes à la réalisation de l’action de recherche qu’elle soit avec ou sans filtre et qui n’est autre que la table médecin.

1. Cas d’utilisation « Localiser médecin » :

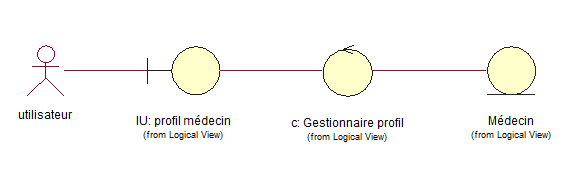


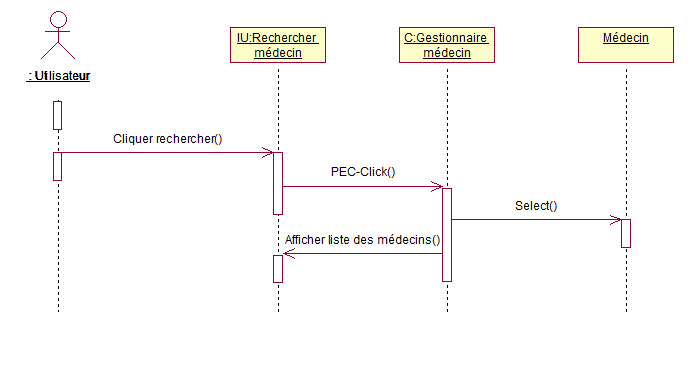
Figure 12 : Diagramme de classe d’analyse du cas « Localiser médecin»

A travers ce diagramme ci-dessus, nous mettons en relief les classes qui sont concernées par la localisation d’un médecin et sont les mêmes aussi pour Appeler un médecin ou lui envoyer un mail car il s’agit toujours de la table médecin.

* 1. Conception :

La conception est la deuxième activité dans un sprint, consiste à façonner le système et lui accorder une forme et une architecture. Elle accomplit le travail déjà commencé au niveau de l’analyse et constitue une entrée majeure pour les activités d’implémentation et de test. Elle se traduit par le diagramme de séquence système et le diagramme de classe d’UML.

* + 1. Diagramme de séquence système :
* Diagramme de séquence du sous cas d’utilisation «Lister tous les médecins »

Figure 13 : Diagramme de séquence du sous cas « lister tous les médecins »

En listant tous les médecins, une opération de lecture de fait sur la table médecin pour afficher tous les éléments qui y sont enregistrés comme l’illustre la figure 15. Le résultat de cette opération est l’affichage sur l’interface utilisateur de la liste complète des médecins qu’il pourra dérouler afin de choisir.

* Diagramme de séquence du sous cas d’utilisation «Filtrer médecin »

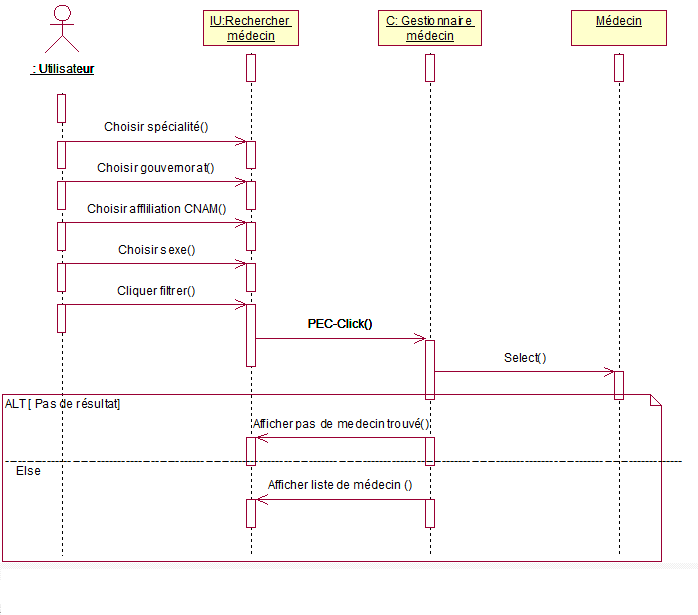


Figure 14 : Diagramme de séquence du sous cas « Filtrer médecins »

Comme l’illustre la figure 16, l’utilisateur précise dès le départ tous ses critères puis lance sa recherche, nous avons pensé à cette solution pour qu’il y ait un seul accès à la base avec toutes les conditions voulues plutôt que de filtrer le résultat un par un.

* Diagramme de séquence du cas d’utilisation «Localiser médecin »

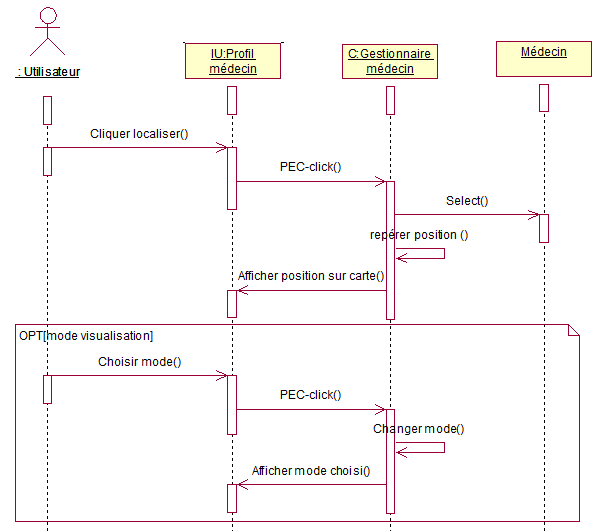


Figure 15 : Diagramme de séquence système du cas « Localiser médecin »

La localisation sur la carte est disponible en 3 modes : plan, terrain et satellite. L’utilisateur peut choisir celui qui lui convient le mieux, c’est une fonctionnalité gérée par l’API de Google adaptée pour un usage facile et rapide.

* Diagramme de séquence du cas d’utilisation «Prendre RDV par Mail »

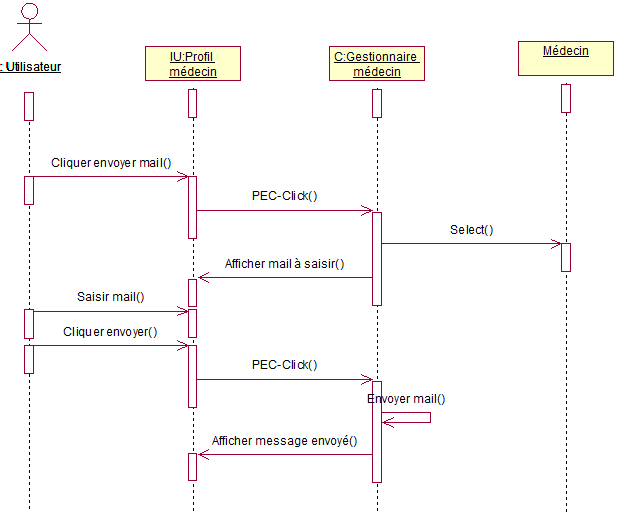


Figure 16: Diagramme de séquence du cas « Prendre RDV par Mail »

Lors de la prise de rendez-vous par mail, dés que l’utilisateur fait ce choix le contrôleur récupère l’adresse mail du médecin en question de la base de données et la prépare dans le champ adéquat, ainsi une fois la saisie est terminée l’utilisateur peut directement envoyer son mail sans se soucier d’insérer l’adresse du destinataire. Sans oublier qu’un message de confirmation d’envoi est affiché une fois l’opération effectuée.

* Diagramme de séquence du cas d’utilisation «Appeler médecin »

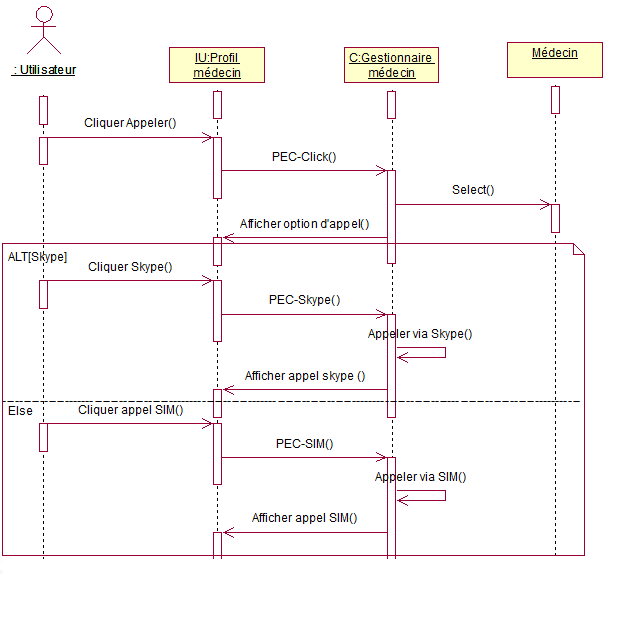


Figure 17 : Diagramme de séquence du cas « Appeler Médecin »

Dans le cas où l’utilisateur souhaiterait émettre un appel, ceci étant directement possible depuis l’application, car le numéro de téléphone est récupéré depuis la base de données avant de donner à l’utilisateur le choix entre la voie Skype ou SIM. Nous avons choisi l’application Skype comme alternative d’appel car c’est l’une des applications les plus répandues non seulement en Tunisie mais aussi dans le monde grâce aux nombreux services qu’elle offre.

* + 1. Diagramme de classe de conception

1. Cas d’utilisation « Rechercher médecin »

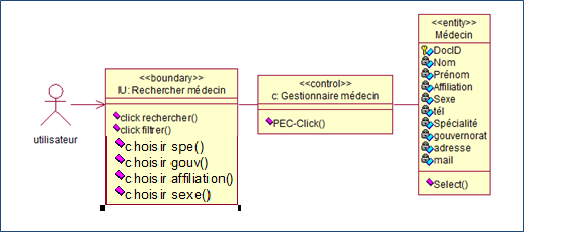


Figure 18 : Diagramme de classe de conception du cas «Rechercher médecin»

A partir de cette même interface de recherche l’utilisateur peut effectuer soit afficher tous les médecins de la base ou filtrer selon ses besoins, comme le représente la figure 20. Nous remarquons qu’il s’agit toujours d’une opération de lecture sur la même table aussi.

1. Cas d’utilisation « Localiser médecin »

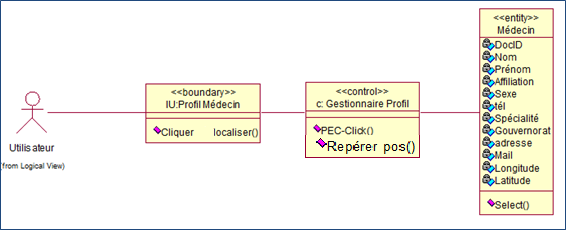


Figure 19 : Diagramme de classe de conception du cas «Localiser médecin»

Une fois le profil du médecin choisi par l’utilisateur est affiché, la localisation est désormais disponible en trois mode, l’émission d’appel et l’envoi de mail également. Il s’agit toujours de collecter les données depuis la table médecin qui contient toutes les informations nécessaires.

* 1. Codage:

Dans cette activité nous passons au développement et à l’implémentation des cas d’utilisation analysés et conçus.

Nous nous intéresserons alors de plus près au schéma de la base de données en présentant la table « médecin »tout en tenant compte du type et des contraintes de leurs champs.

|  |  |  |
| --- | --- | --- |
| Champs | Types | Contraintes |
| DocID | INT | PRIMARY KEY |
| Nom | INT | --- |
| Prénom | VARCHAR(200) | --- |
| Gouvernorat | VARCHAR(200) | --- |
| Spécialité | VARCHAR(200) |  |
| Adresse | VARCHAR(200) |  |
| Tél | INT | NOT NULL |
| Mail | VARCHAR(200) | NOT NULL |
| Affiliation | VARCHAR(20) |  |
| Sexe | VARCHAR(20) |  |
| Longitude | FLOAT | NOT NULL |
| Latitude | FLOAT | NOT NULL |

Tableau 9 : Table Médecin

* 1. Test :

Le test est un processus manuel ou automatique, qui vise à établir qu’un système vérifie les propriétés exigées par sa spécification, ou à détecter des différences entre les résultats engendrés par le système et ceux qui sont attendus par la spécification. (Définition issue de la norme IEEE-STD729, 1983).

Les activités de test constituent un axe très important dans le cycle de développement d’un logiciel. Ils permettent de détecter les erreurs afin de les corriger et d’assurer la qualité du logiciel fourni.

Contrairement aux cycles de développement séquentiel, avec la méthodologie agile, le test n'est pas une phase qui se déroule après la fin de développement. En effet, les tests seront intégrés dès le début du premier sprint jusqu’à la livraison du produit final.

Dans ce premier sprint, nous avons testé la fonctionnalité «  Rechercher médecin » qui affiche avec succès les médecins de la base avec ou sans filtre.

Ci-dessus nous ajoutons des captures écran du sprint réalisé.

Figure 20 : Interfaces recherche médecin

Nous présentons l’interface depuis laquelle l’utilisateur effectue sa recherche, il peut choisir la spécialité et le gouvernorat depuis une liste qui récupère ses valeurs depuis la base de donnée grâce aux web services qui parcourent les valeurs existantes et les affichent. Ainsi à chaque fois qu’il y a des modifications dans la base de données l’utilisateur retrouve automatiquement les nouvelles valeurs ajoutées ou modifiées. Puis il peut choisir si le médecin qu’il cherche doit être affilié à la CNAM ou pas et finalement il peut préciser le sexe du médecin recherché.

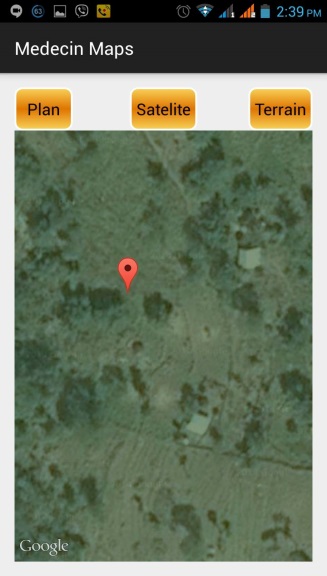
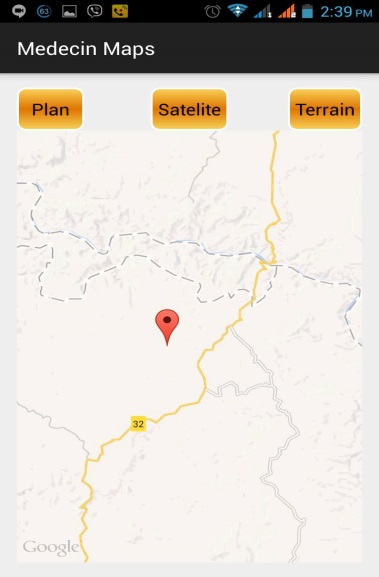
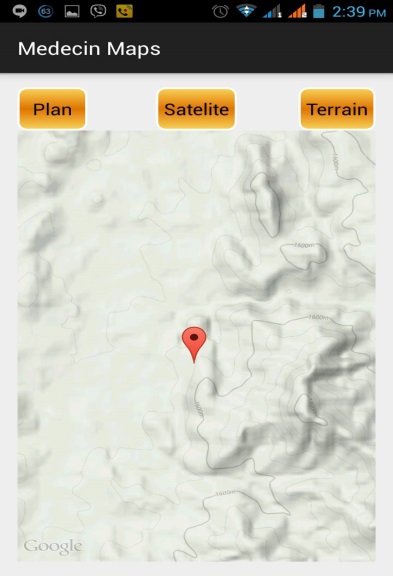
  

Figure 21 : Interfaces Localisation médecin (Terrain, Satellite, Plan)

Pour la localisation sur la carte représentée sur la figure 23, la position de médecin qui n’est autre que des coordonnées depuis Google Maps , affichées en mode terrain satellite ou plan . L’utilisateur peut choisir le mode qui lui convient le mieux.

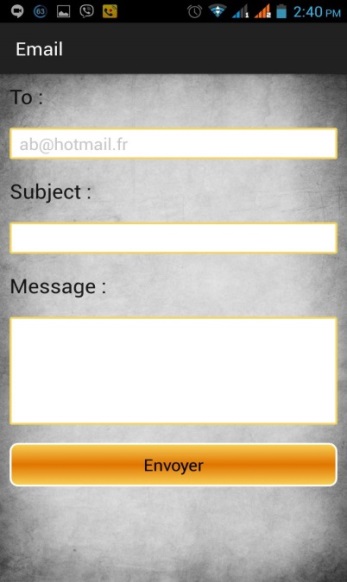
  

Figure 22 : Interfaces profil médecin, Appeler et Envoyer mail

Dans la figure ci-dessus, nous présentons les interfaces de l’utilisateur lors de l’appel ou l’envoi de mail pour un médecin qu’il a préalablement choisi. Nous avons également l’interface profil médecin depuis lequel toutes les actions possibles sont initiées.

1. Le second Sprint

En partant sur le même principe que le sprint précédent, mais cette fois ci nous nous intéressons aux cas d’utilisation suivants :

* Gérer médecin Favoris
* S’authentifier

Notre objectif est d’avoir, à l’issue de ce sprint une version de l’application qui soit stable dans les fonctionnalités offertes pour ne pas y revenir ultérieurement.

* 1. **Analyse**

Comme l’utilisateur ne peut accéder aux fonctionnalités de la gestion des médecins favoris qu’une fois authentifié, nous allons traiter le cas d’utilisation « S’authentifier » en premier lieu.

* + 1. Raffinement des cas d’utilisation

1. Cas d’utilisation « Créer Compte »

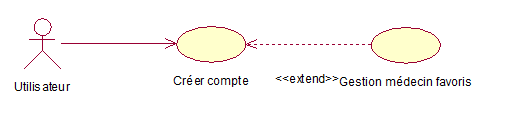


Figure 23: Raffinement du cas d’utilisation « Créer Compte»

* **Description du sous Cas d’Utilisation « Créer Compte»**

|  |  |
| --- | --- |
| Cas d’utilisation: | Créer Compte |
| Acteur : | Utilisateur |
| Pré-condition: | --- |
| Post-condition: | Utilisateur authentifié. |
| Description de scénario de base : | **1.** L’utilisateur clique sur créer compte.  **2.** Le système affiche le formulaire d’inscription.  **3.** L’utilisateur saisit les données demandées puis clique Enregistré.  **4.** Le système crée le compte et affiche un message de succès d’opération. |
| Exception : | -Le Compte est déjà existant, dance ce cas l’utilisateur est averti. |

Tableau 10 : Raffinement du sous cas d’utilisation « Créer Compte »

1. Cas d’utilisation « S’authentifier » :

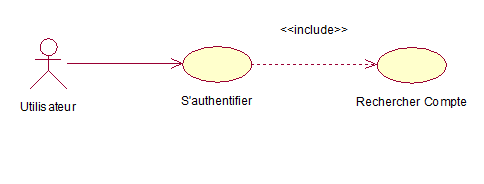


Figure 24: Raffinement du cas d’utilisation « S’authentifier »

Ce cas d’utilisation offre à l’utilisateur la possibilité de se connecter à un compte pour pouvoir stocker les informations qu’il jugera utile pour son usage futur. Un nouvel utilisateur devra créer un nouveau compte tandis qu’un utilisateur qui a déjà un compte pourra directement saisir son nom d’utilisateur et son mot de passe. Le tableaux descriptifs suivant montrent avec plus de détails le déroulement de ce cas.

* **Description du sous Cas d’Utilisation « Rechercher Compte»**

|  |  |
| --- | --- |
| Cas d’utilisation: | Rechercher Compte |
| Acteur : | Utilisateur |
| Pré-condition: | --- |
| Post-condition: | Utilisateur authentifié. |
| Description de scénario de base : | **1.** L’utilisateur Saisit son nom d’utilisateur et son mot de passe.  **2.** L’utilisateur clique sur Login.  **3.** Le système vérifie la combinaison saisie.  **4.** Compte trouvé, un massage de succès d’opération est affichée.  **5.** Le système active les nouvelles fonctionnalités possibles. |
| Exception : | -Si le nom d’utilisateur ou le mot de passe erroné, le système affiche un message d’erreur.  -Le système le redirige au début de scenario principal. |

Tableau 11 : Raffinement du sous cas d’utilisation «  Rechercher Compte»

1. « Gérer Médecin Favoris »

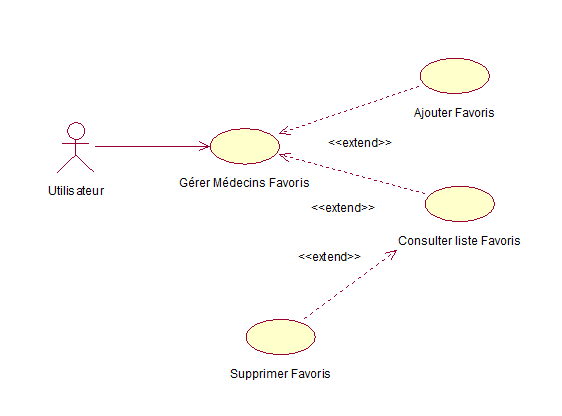


Figure 25 : Raffinement du cas d’utilisation « Gérer Favoris »

Ce cas d’utilisation offre à l’utilisateur la possibilité de gérer les médecins favoris, il permet d’enregistrer, de supprimer ou de consulter des données à partir du compte de l’utilisateur.

Nous trouvons plus de détails de chacun des cas dans les tableaux descriptifs qui suivent.

* **Description du sous Cas d’Utilisation « Ajouter Favoris»**

|  |  |
| --- | --- |
| Cas d’utilisation: | Ajouter Favoris |
| Acteur : | Utilisateur |
| Pré-condition: | Utilisateur authentifié. |
| Post-condition: | Médecin ajouté à la liste des favoris avec succès. |
| Description de scénario de base : | **1.** L’utilisateur clique sur Enregistrer favoris.  **2.** Le système enregistre le médecin sur la liste.  **3.** Le système affiche un message d’ajout avec succès. |
| Exception : | -Si le médecin choisi, figure déjà sur la liste des favoris, le système avertit l’utilisateur avec un message adéquat. |

Tableau 12 : Raffinement du sous cas d’utilisation « Ajouter Favoris »

* **Description du sous Cas d’Utilisation « Consulter Favoris»**

|  |  |
| --- | --- |
| Cas d’utilisation: | Consulter Favoris |
| Acteur : | Utilisateur |
| Pré-condition: | Utilisateur authentifié. |
| Post-condition: | Liste des médecins Favoris préalablement enregistré, affichée. |
| Description de scénario de base : | **1.** L’utilisateur clique sur Consulter favoris.  **2.** Le système Affiche la liste des médecins enregistrés comme favoris. |
| Exception : | -S’il n y a pas de médecin préalablement enregistré, un message adéquat s’affiche. |

Tableau 13 : Raffinement du sous cas d’utilisation « Consulter Favoris »

* **Description du sous Cas d’Utilisation « Supprimer Favoris»**

|  |  |
| --- | --- |
| Cas d’utilisation: | Supprimer Favoris |
| Acteur : | Utilisateur |
| Pré-condition: | Liste de médecins favoris existante. |
| Post-condition: | Médecin supprimé de la liste des favoris avec succès. |
| Description de scénario de base : | **1.** L’utilisateur clique choisit de la liste des médecins favoris, l’élément à supprimer.  **2.** L’utilisateur clique sur supprimer.  **3.** Le système affiche un message de confirmation pour que l’utilisateur valide ou pas son choix.  **4.** L’utilisateur confirme en cliquant Oui.  **5.** Le système supprime le médecin de la liste des favoris.  **6.** Le système affiche un message de succès d’opération. |
| Exception : | -Si l’utilisateur ne confirme pas le choix de suppression, l’opération entière est annulée. |

Tableau 14 : Raffinement du sous cas d’utilisation « Supprimer Favoris »

* + 1. Diagramme de classe d’analyse

1. Cas d’utilisation « Créer Compte »

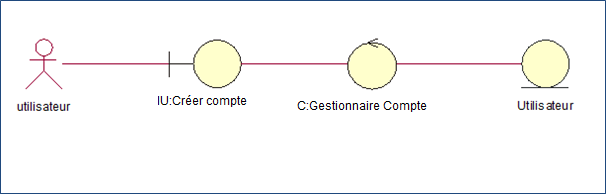
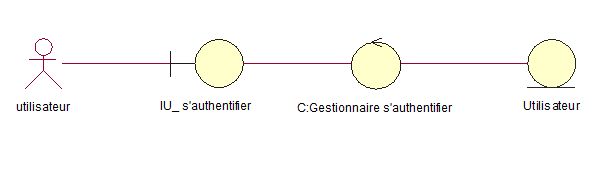


Figure 26 : Diagramme de classe d’analyse du cas « Créer Compte»

L’utilisateur peut créer un compte à tout moment afin de pouvoir se connecter par la suite et enregistrer les informations qui lui seront utiles.

1. Cas d’utilisation « S’authentifier » :

****Figure 27 : Diagramme de classe d’analyse du cas « S’authentifier»

L’authentification est désormais nécessaire si l’utilisateur veut bénéficier de plus de fonctionnalités telles que la gestion des médecins favoris et qui n’est pas permise pour un utilisateur non authentifié. La figure 27 montre les classes qui sont concernées par cette opération d’authentification, sachant que pour cela il faut déjà avoir créé un compte.

1. Cas d’utilisation « Gérer médecin favoris »

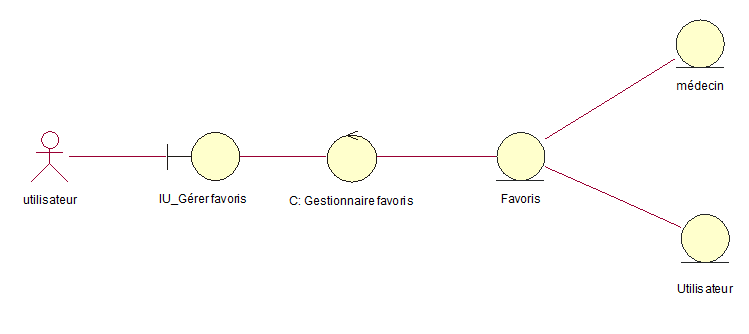


Figure 28 : Diagramme de classe d’analyse du cas d’utilisation « Gérer Favoris»

Pour la gestion des médecins favoris, comme le montre la figure 28, nous aurons recours à une nouvelle table intitulée favoris, pour enregistrer les choix des utilisateurs. Sans oublier la table médecin et aussi utilisateurs.

* 1. **Conception**
     1. Diagramme de séquence :

Afin d’assurer une bonne compréhension du comportement voulu du système nous allons traiter tous les cas en détails dans le même ordre que l’étape de l’analyse.

1. Cas d’utilisation « Créer compte »

* Diagramme de séquence du cas d’utilisation «Créer Compte »

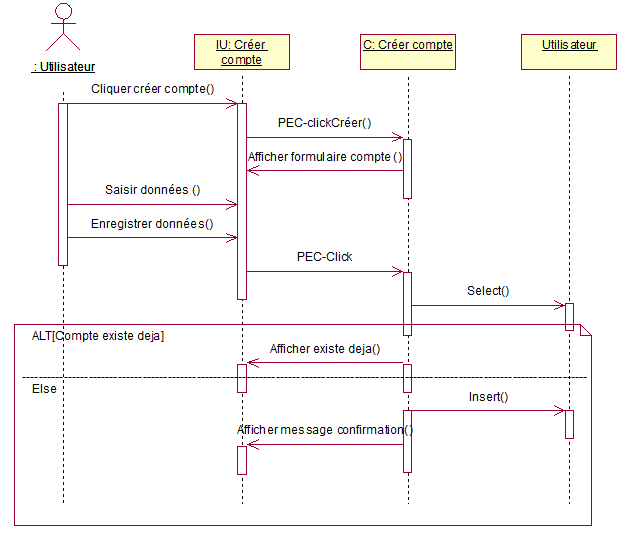


Figure 29 : Diagramme de séquence du sous cas d’utilisation « Créer Compte»

Pour créer un compte le contrôleur vérifie d’abord si les données saisies par l’utilisateur ne sont pas déjà enregistrées d’où l’opération de lecture simple de la table utilisateur afin d’éviter les redondances. Puis une fois la vérification est terminée avec succès les informations peuvent être enregistrées dans la même table.

1. Cas d’utilisation « S’authentifier »

* Diagramme de séquence du cas d’utilisation «S’authentifier »

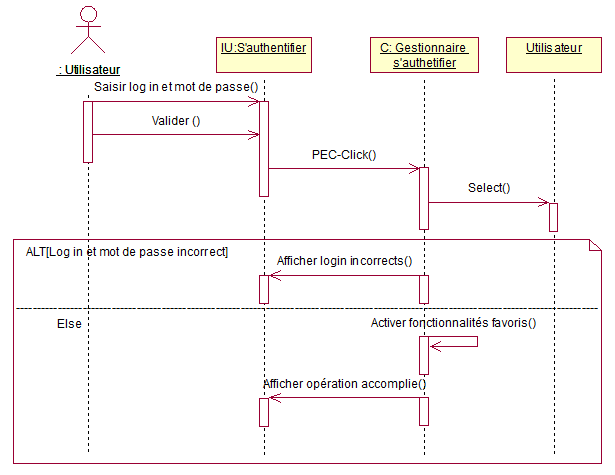


Figure 30: Diagramme de séquence du sous cas d’utilisation « S’authentifier»

En s’authentifiant au système l’utilisateur accède à son compte via le mot de passe et le nom d’utilisateur qu’il a choisi lors de la création de celui-ci. Ces données seront enregistrées dans la table intitulée utilisateur et c’est la raison pour laquelle le contrôleur effectue une opération de lecture sur cette table à la rechercher de l’utilisateur en question avant de permettre l’accès ou le refuser.

1. Cas D’utilisation « Gérer Favoris »

* Diagramme de séquence du sous cas d’utilisation « Ajouter Favoris»

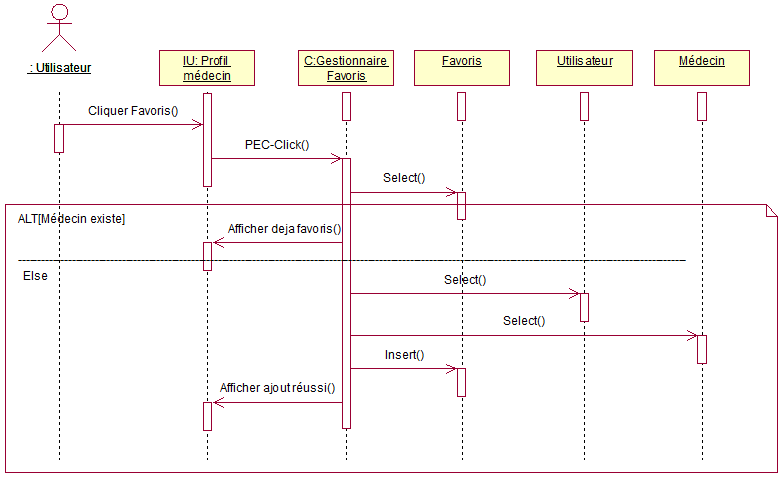


Figure 31 : Diagramme de séquence du sous cas d’utilisation « Ajouter Favoris»

L’ajout d’un médecin favoris engendre l’accès aux trois tables médecin utilisateur et favoris. Car il s’agit de récupérer des informations de l’utilisateur et puis celles du médecin afin de les stocker dans la table favoris, ce ci bien sûr après avoir vérifié que le médecin en question n’a pas été préalablement choisi comme favoris par cet utilisateur authentifié. Cette vérification permet non seulement d’éviter les redondances mais aussi donne à l’utilisateur un sentiment de confiance et le laisse se fier à son application en toute sécurité.

* Diagramme de séquence du cas d’utilisation «Consulter Favoris »

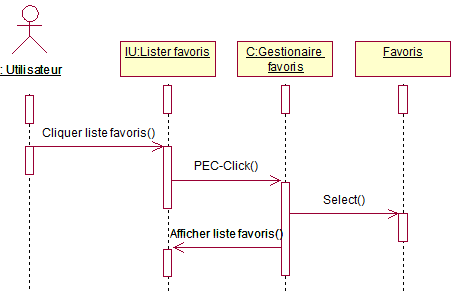


Figure 32 : Diagramme de séquence du sous cas d’utilisation « Consulter Favoris»

Un utilisateur authentifié peut à tout moment consulter la liste de ces médecins favoris pour éviter à refaire une recherche à chaque fois pour le même médecin. Et comme Lors de la création les informations nécessaires ont été enregistrées dans la table des favoris, il suffit d’y accéder en lecture pour afficher la liste complète des médecins favoris d’un utilisateur donné.

* Diagramme de séquence du cas d’utilisation «Supprimer Favoris »

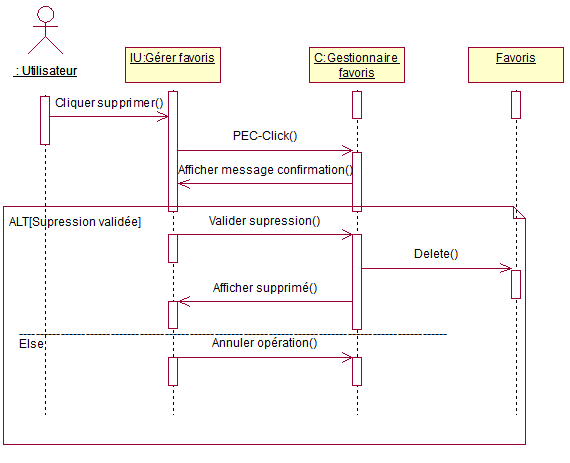


Figure 33 : Diagramme de séquence du sous cas d’utilisation « Supprimer Favoris»

L’utilisateur a la possibilité de supprimer un médecin de la liste de ses favoris, il s’agit d’une opération effectué sur la table des favoris uniquement, nous avons choisi d’afficher un message de confirmation avant d’effectuer l’opération dans un soucis de nous assurer que l’action n’a pas été initiée par erreur pour assurer plus de sécurité aux données préalablement sauvegardées par ce même utilisateur .

* + 1. Diagramme de classe de conception

1. Cas D’utilisation « Créer Compte »

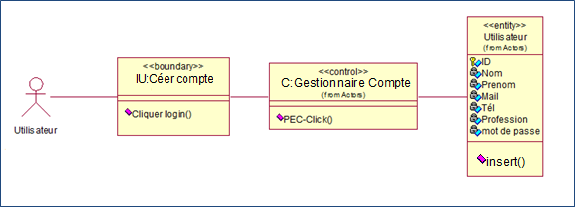


Figure 34 : Diagramme de classe de conception du cas «créer Compte»

1. Cas d’utilisation « S’authentifier »

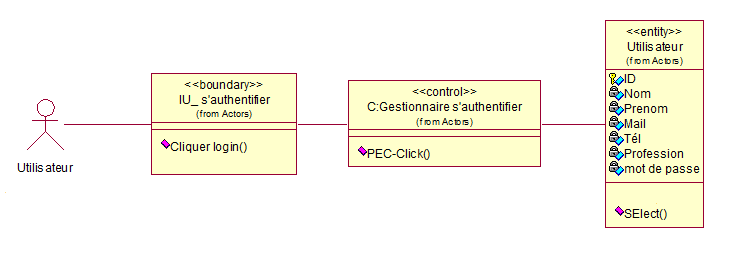


Figure 35 : Diagramme de classe de conception du cas «S’authentifier»

1. Cas d’utilisation « Gérer Favoris »

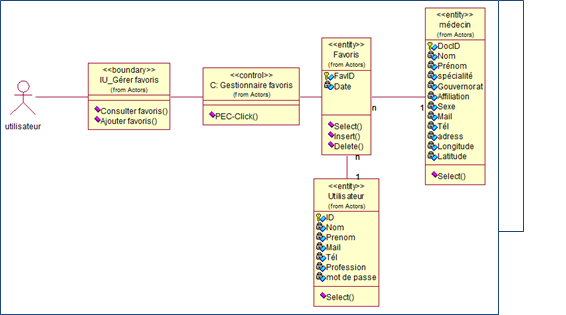


Figure 36 : Diagramme de classe de conception du cas «Gérer favoris»

* 1. **Codage**

Après avoir construit le nouveau diagramme de classe pour ce sprint et en appliquant les règles de passage vers le schéma logique de l’application, nous obtenons le schéma de la base de données suivant qui s’ajoute aux tables préalablement établies :

|  |  |  |
| --- | --- | --- |
| Champs | Types | Contraintes |
| ID | INT | PRIMARY KEY |
| Nom | VARCHAR(200) | --- |
| Prenom | VARCHAR(200) | --- |
| Telephone | VARCHAR(8) | NOT NULL |
| Email | VARCHAR(200) | UNIQUE |
| Profession | VARCHAR(200) | --- |

Tableau 15 : Table Utilisateur

|  |  |  |
| --- | --- | --- |
| Champs | Types | Contraintes |
| ID | INT | PRIMARY KEY |
| Date | DATE | --- |
| Ref-ID-User | VARCHAR(200) | NOT NULL |
| Red-ID-Doc | VARCHAR(8) | NOT NULL |

Tableau 16 : Table Favoris

* 1. **Test**

En partant toujours du même principe que le sprint précédent, et en appliquant les bonnes pratiques d’ingénierie logicielle nous commençons le test des cas d’utilisation développés.

En voici des captures écrans.

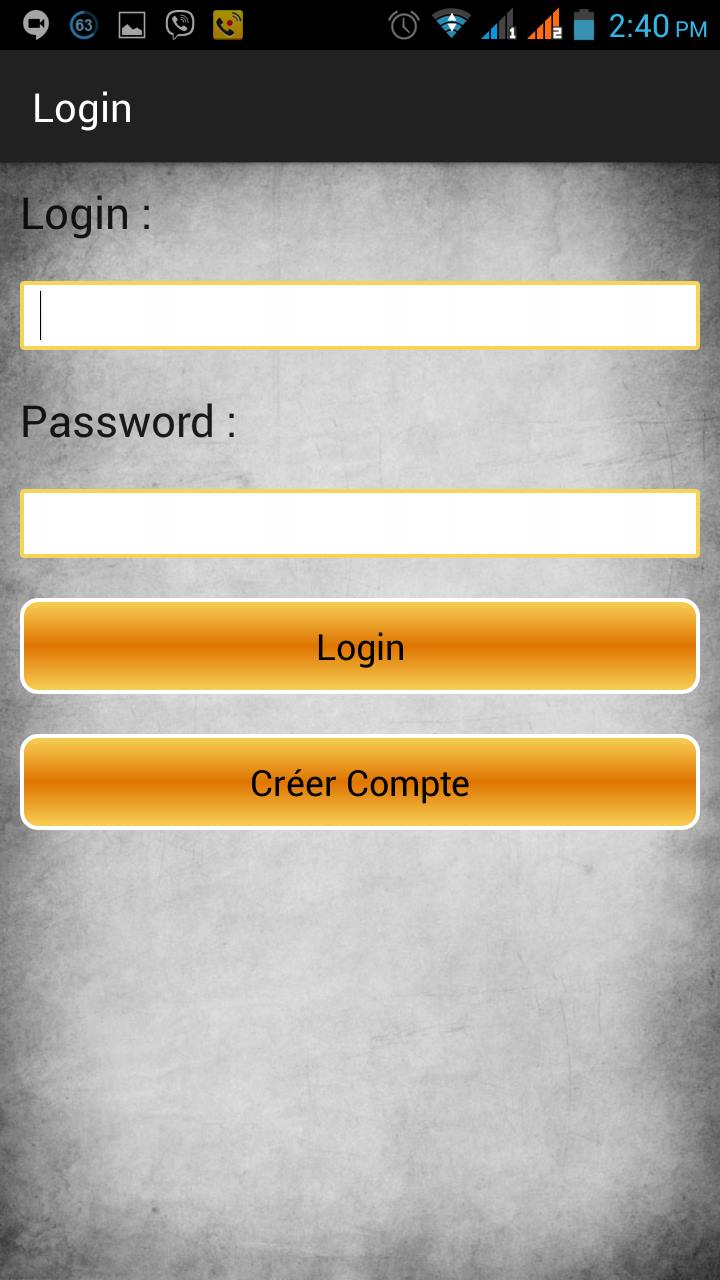
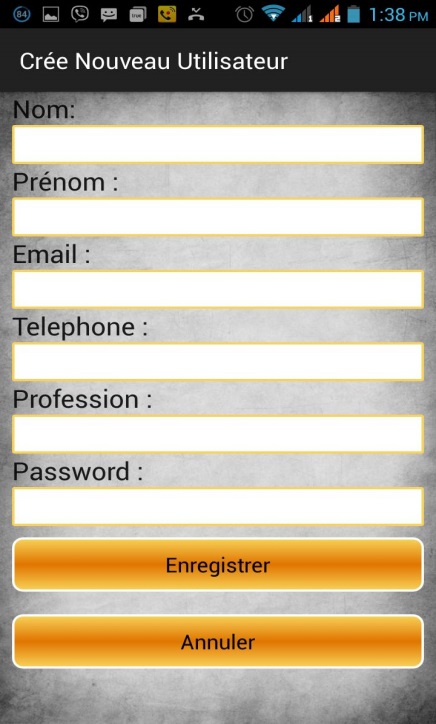
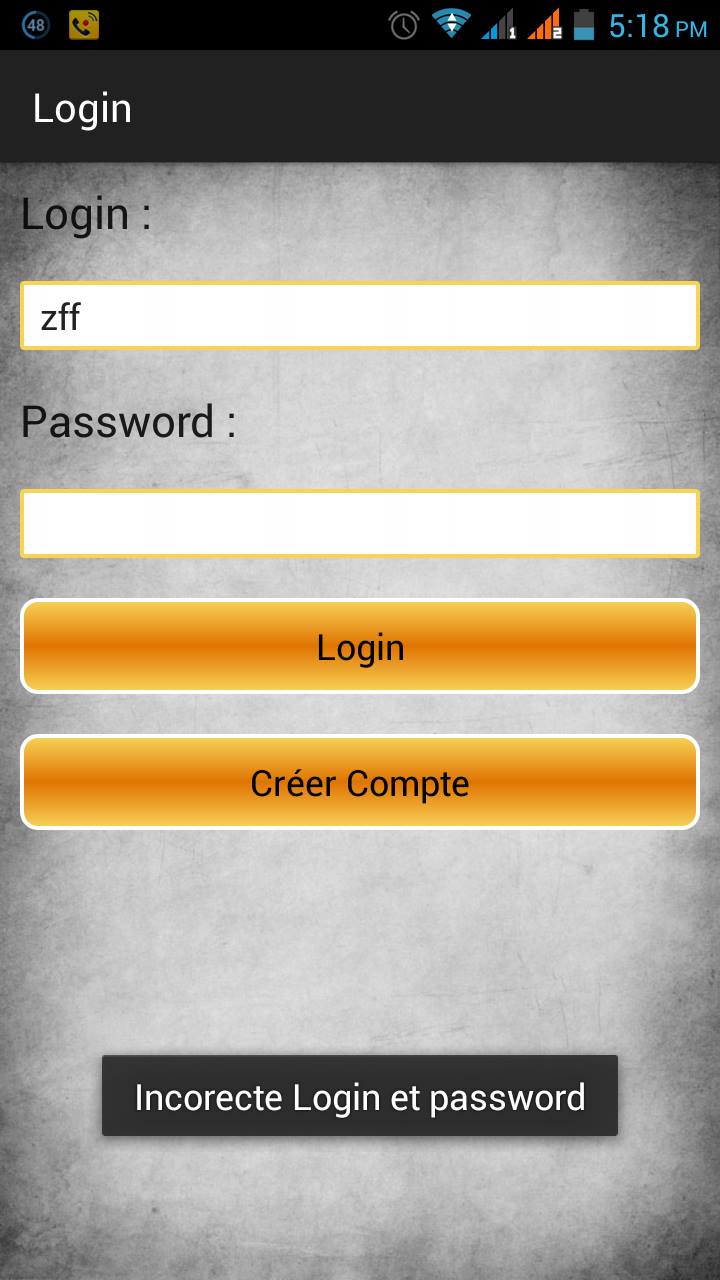
  

Figure 37 : Interfaces authentification utilisateur

Sur la figure 36, nous présentons les interfaces à partir desquelles l’utilisateur peut se connecter ainsi que le formulaire d’inscription pour la création d’un nouveau compte. Nous avons également tester une combinaison erronée lors de l’authentification afin de vérifier le retour d’information constructif à l’utilisateur.

* 1. **Conclusion**

Le résultat de ce release est un produit livrable contrairement au résultat d’un sprint qui est un produit potentiellement livrable mais qui offre de fonctionnalités limitées. A la fin de ce chapitre, nous avons réussi à produire une application mobile sous Android ayant suffisamment de valeur et qui pourra être utilisé. Dans le chapitre qui suit, notre effort sera consacré à produire une nouveau release, une nouvelle application couvrant les fonctionnalités d’administration de la partie mobile.

Chapitre 4

Release 2

1. Introduction

Après avoir entamé le premier release de notre système informatique, nous pouvons maintenant nous lancer dans les travaux nécessaires pour produire le second release. En effet les méthodologies agiles, et Scrum en particulier, sont caractérisées par un rythme régulier. Tout au long de chapitre nous allons détailler l’application web dédié à l’administration de la partie mobile, en présentant les différents cas d’utilisation proposé dans le Backlog présenté dans le deuxième chapitre. Nous un sprint ayant la même vélocité que les sprints précédents, pour avoir à la fin de ce release deux applications complémentaires.

1. Le premier Sprint

En partant du même principe que les sprints précédents, nous commençons par définir le but de notre premier sprint, les cas d’utilisation relatifs à l’application Web, présentés préalablement dans le diagramme de cas d’utilisation sont :

* S’authentifier
* Gérer médecin
* Gérer gouvernorat
* Gérer spécialité
* Gérer coordonnées Map
* Gérer utilisateur

Cette application vise à gérer les CRUD des tables de la base de données, a part le cas d’authentification, le reste des cas d’utilisation suivront exactement la même analyse, et la même conception, il s’agit de concevoir les mêmes fonctionnalités mais pour différents éléments. C’est pourquoi nous avons choisi de traiter en détail lors ce chapitre l’exemple « Gérer médecin » en plus de l’authentification.

* 1. **Analyse**
     1. Raffinement des cas d’utilisations

1. Cas d’utilisation «  S’authentifier » :

A travers ce CU, l’utilisateur va pouvoir se connecter au système en composant son login et son mot de passe, le système va s’assurer de son existence en tant qu’administrateur autorisé avant de lui donner l’accès au système.

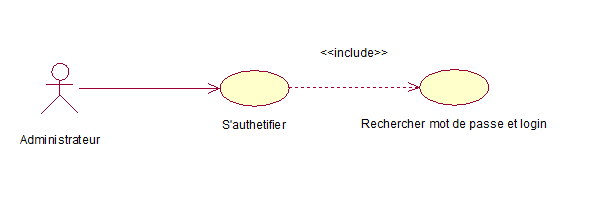


Figure 38 : Raffinement du cas d’utilisation « S’authentifier »

* **Description du Cas d’Utilisation « S’authentifier »**

|  |  |
| --- | --- |
| Sous cas d’utilisation: | S’authentifier |
| Acteur : | Administrateur |
| Pré-condition: | Avoir login et mot de passe correct. |
| Post-condition: | Administrateur authentifié. |
| Description de scénario de base : | **1.** L’utilisateur saisit son log in et son mot de passe.  **2.** L’utilisateur clique sur «Log in ».  **3.** Le système vérifie la combinaison login et mot de passe.  **4.** Login et mot de passe correcte, le système affiche l’accueil . |
| Exception : | - S’il n y a aucun médecin enregistré dans la base, un massage s’affiche « Pas de médecin enregistré ». |

Tableau 17 : Raffinement du cas d’utilisation «S’authentifier»

1. Cas d’utilisation «  Gérer médecins » :

Il s’agit de définir les CRUD de la table médecin, l’administrateur peur ajouter un nouveau médecin, modifier, supprimer ou consulter un médecin existant.

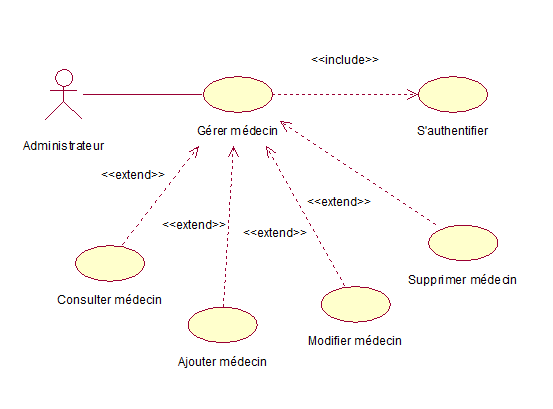


Figure 39 : Raffinement du cas d’utilisation « Gérer médecin »

Détailler le cas d’utilisation « Gérer médecins » revient à détailler ses sous cas d’utilisation présentés dans la figure ci-dessus.

* **Description du sous Cas d’Utilisation « Consulter médecin»**

|  |  |
| --- | --- |
| Cas d’utilisation: | Consulter Médecin |
| Acteur : | Administrateur |
| Pré-condition: | Administrateur authentifié |
| Post-condition: | Médecin affiché. |
| Description de scénario de base : | **1.** L’utilisateur choisit le médecin à afficher de toute la liste des médecins.  **2.** L’utilisateur clique sur View.  **3.** Le système affiche la fiche du médecin choisi. |
| Exception : | -Liste des médecins vide. |

Tableau 18 : Raffinement du sous cas d’utilisation «  Consulter médecin»

* **Description du sous Cas d’Utilisation « modifier médecin»**

|  |  |
| --- | --- |
| Cas d’utilisation: | Modifier Médecin |
| Acteur : | Administrateur |
| Pré-condition: | Administrateur authentifié |
| Post-condition: | Médecin modifié |
| Description de scénario de base : | **1.** L’utilisateur choisit le médecin à modifier de toute la liste des médecins.  **2.** L’utilisateur clique sur Edit.  **3.** Le système affiche la fiche du médecin choisi que l’administrateur peut modifier.  4. L’utilisateur modifie la fiche.  5. L’utilisateur clique sur Update pour enregistrer les modifications. |
| Exception : | -L’utilisateur peut annuler l’opération. |

Tableau 19 : Raffinement du sous cas d’utilisation « Modifier médecin»

* **Description du sous Cas d’Utilisation « Ajouter médecin»**

|  |  |
| --- | --- |
| Cas d’utilisation: | Ajouter Médecin |
| Acteur : | Administrateur |
| Pré-condition: | Administrateur authentifié |
| Post-condition: | Médecin Ajouté. |
| Description de scénario de base : | **1.** L’utilisateur clique sur Add.  **3.** Le système affiche une fiche médecin vide que l’administrateur peut remplir.  4. L’utilisateur saisit les données adéquates.  5. L’utilisateur clique sur Save pour enregistrer son entrée. |
| Exception : | -L’utilisateur peut annuler l’opération. |

Tableau 20 : Raffinement du sous cas d’utilisation «  Ajouter médecin»

* **Description du sous Cas d’Utilisation « supprimer médecin»**

|  |  |
| --- | --- |
| Sous cas d’utilisation | Supprimer Médecin |
| Acteur | Administrateur |
| Pré-condition : | Administrateur authentifié |
| Post-condition : | Médecin supprimé |
| Description de scénario de base : | **1**. L’administrateur choisit le médecin à supprimer.  **2.** Il clique sur Delete.  **3.** Le système affiche un message s’il est sûr de vouloir supprimer ce médecin.  **4.** L’administrateur clique sur le bouton de confirmation.  **5.** Le système supprime le médecin courant. |
| Exception : | -L’administrateur peut annuler l’opération. |

Tableau 21 : Raffinement du sous cas d’utilisation «  Supprimer médecin»

* + 1. Diagramme de classe d’analyse

1. Cas d’utilisation « s’authentifier »

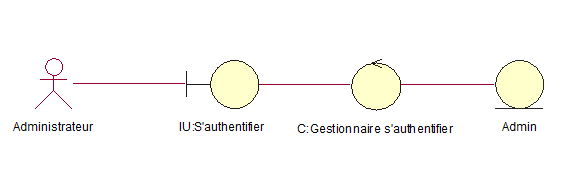


Figure 40 : Diagramme de classe d’analyse du cas « S’authentifier»

Dans cette partie l’authentification est une étape primordiale pour donner la main à l’administrateur de gérer convenablement ses tables. Aucune Opération n’est permise sans authentification et pour ce faire le système vérifie les données insérées depuis la Table Admin.

1. Cas d’utilisation «  Gérer Médecin »

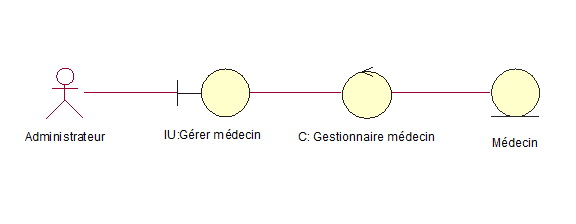


Figure 41 : Diagramme de classe d’analyse du cas « Gérer médecin»

Pour gérer les CRUD de la table Médecin, elle est la seule table qui sera concerné par les traitements à effectuer. De même pour les autres cas, quand il s’agit gérer de gérer la table Spécialité , c’est celle-ci qui figurera dans le diagramme de classe d’analyse et ainsi de suite pour les tables gouvernorats et coordonnées Map.

* 1. **Conception**
     1. Diagramme de séquence

Nous allons décrire le déroulement exact des taches que peut réaliser l’administrateur via des diagrammes séquences système, de la même manière que les sprints précédents.

1. Cas d’utilisation « S’authentifier »

* Diagramme de séquence du cas d’utilisation «S’authentifier »

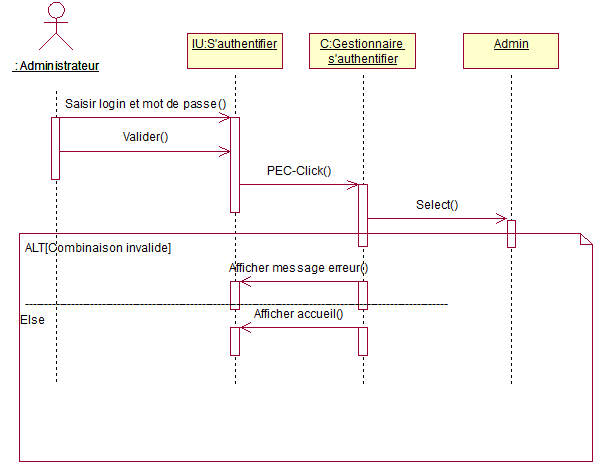


Figure 42 : Diagramme de séquence du cas d’utilisation « S’authentifier»

1. Cas d’utilisation «  gérer médecin »

Nous présentons dans cette partie les diagrammes relatifs à ses sous cas d’utilisation :

* Diagramme de séquence du sous cas d’utilisation « Consulter médecin»

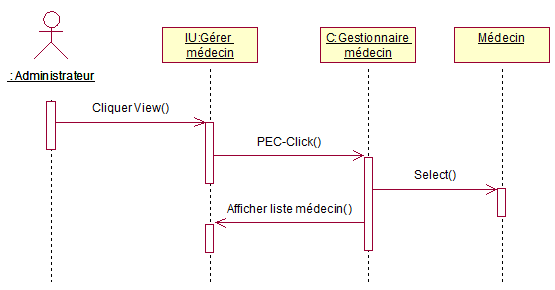


Figure 43 : Diagramme de séquence du sous cas d’utilisation « Consulter médecin»

* Diagramme de séquence du sous cas d’utilisation « Modifier médecin»

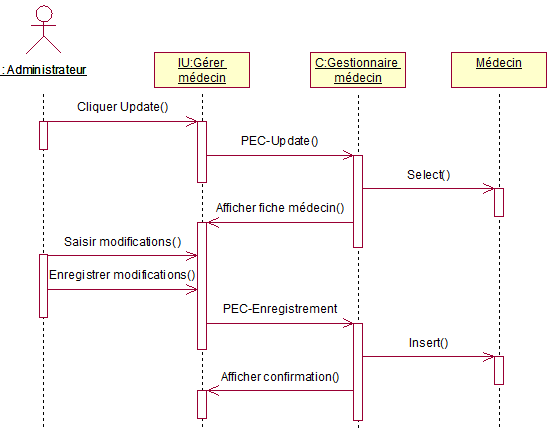


Figure 44 : Diagramme de séquence du sous cas d’utilisation « Modifier médecin»

* Diagramme de séquence du sous cas d’utilisation « Ajouter médecin»

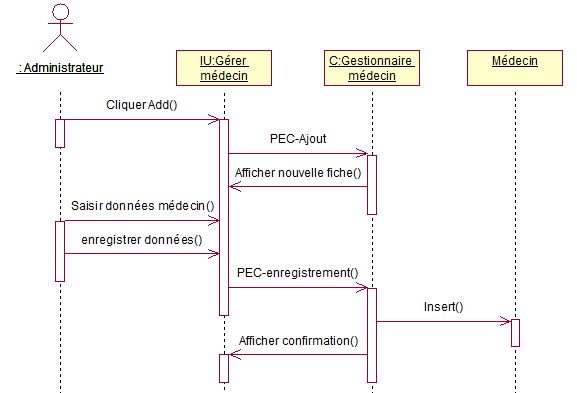


Figure 45 : Diagramme de séquence du sous cas d’utilisation « Ajouter médecin»

* Diagramme de séquence du sous cas d’utilisation « Supprimer médecin»

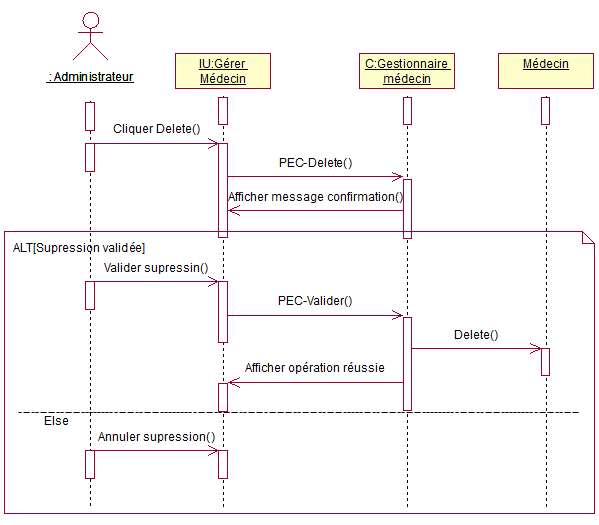


Figure 46 : Diagramme de séquence du sous cas d’utilisation « Supprimer médecin»

Nous avons traité les opérations que l’administrateur peut effectuer lors de la gestion de la table médecin, qui sont exactement les mêmes pour les autres tables de base de données. La seule différence étant la table qui sera affectée, selon les besoins de l’administrateur.

* + 1. Diagramme de classe de conception

1. Cas d’utilisation « S’authentifier »

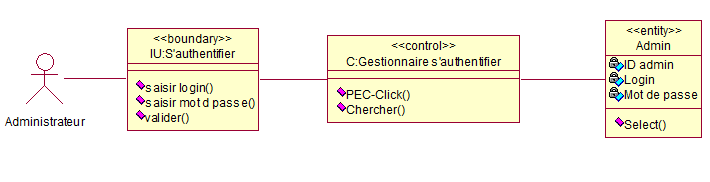


Figure 47 : Diagramme de classe de conception du cas d’utilisation «S’authentifier»

1. Cas d’utilisation « Gérer médecin »

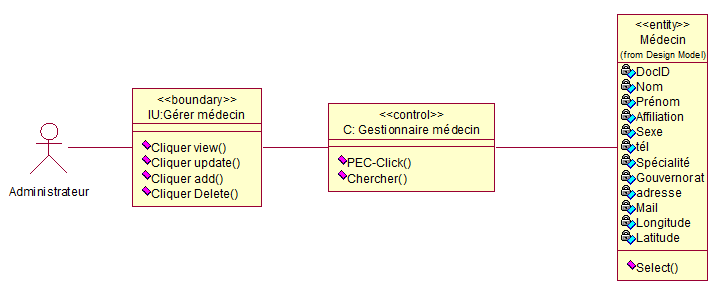


Figure 48 : Diagramme de classe de conception du cas d’utilisation «Gérer Médecin»

* 1. **Codage**

Le schéma de la base de données étant établi lors du premier release, notre application va utiliser exactement les mêmes tables Mais pour assurer une meilleur gestion et évolution du système, nous avons décidé d’éclater la table médecin en la fragmentant horizontalement, pour créer d’autres tables dont nous pouvons gérer les CRUD avec notre application Web.

Nous avons donc créé une table pour les gouvernorats, une pour la spécialité et une pour coordonnées géographique.

Voici le schéma des tables :

|  |  |  |
| --- | --- | --- |
| Champs | Types | Contraintes |
| id | INT | PRIMARY KEY |
| code | VARCHAR(200) | UNIQUE |

Tableau 22 : Table Gouvernorat

|  |  |  |
| --- | --- | --- |
| Champs | Types | Contraintes |
| id | INT | PRIMARY KEY |
| code | VARCHAR(200) | UNIQUE |

Tableau 23 : Table Spécialité

|  |  |  |
| --- | --- | --- |
| Champs | Types | Contraintes |
| id | INT | PRIMARY KEY |
| Longitude | FLOAT | NOT NULL |
| Latitude | FLOAT | NOT NULL |

Tableau 24 : Table Coordonnées Map

* 1. **Test**

Une fois l’application Web est développée, nous passons au test des CRUD sur toutes les tables de la base donnée, la manière dont nous avons procédé est la suivante :

* Pour l’authentification, nous avons essayé de nous connecté avec un login et un mot de passe erronés
* Pour la consultation d’une table, nous vérifions si tous les éléments sont affichés.
* Pour la modification, nous comparons les résultats avant et après enregistrement des modifications.
* Pour la suppression nous vérifions la cohérence de la base après suppression d’un élément.
* Et pour l’ajout nous vérifions le nombre des éléments après une opération d’ajout.

Voilà quelques exemples :

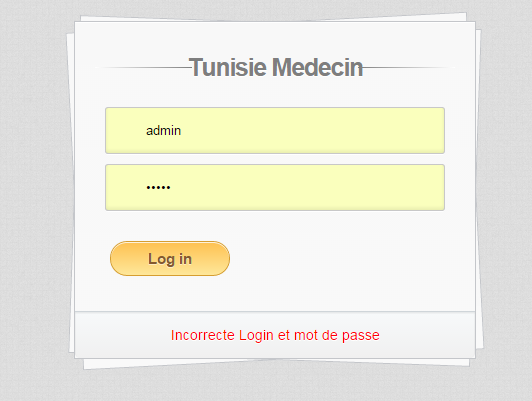


Figure 49 : interface du cas d’utilisation «S’authentifier» avec login et mot de passe erronés

La figure 47 montre le message d’erreur qui s’affiche lors d’une saisie erronée, l’utilisateur ne peut pas accéder à l’accueil tant que la bonne combinaison n’est pas saisie.

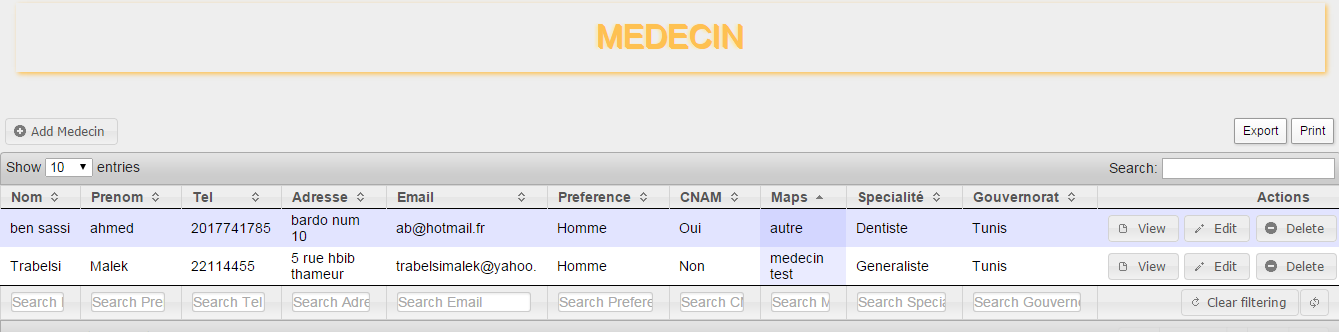
****

Figure 50 : interface du cas d’utilisation «Consulter médecin»

L’affichage de tous les médecins de la base de données est effectué sur l’interface représenté dans la figure 48. C’est à partir de cette même interface que toutes les actions d’ajout, modification ou suppression sont rendues possible.

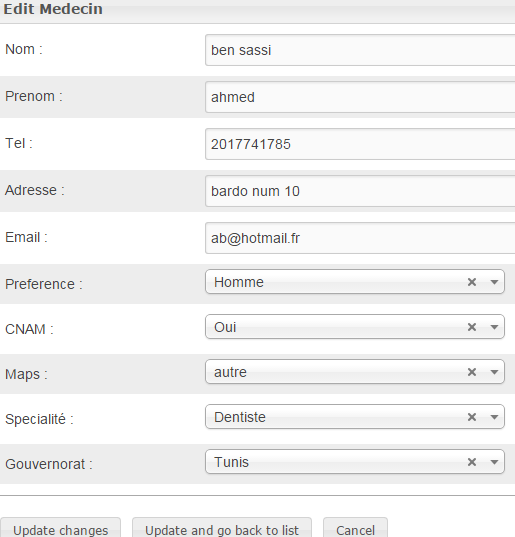
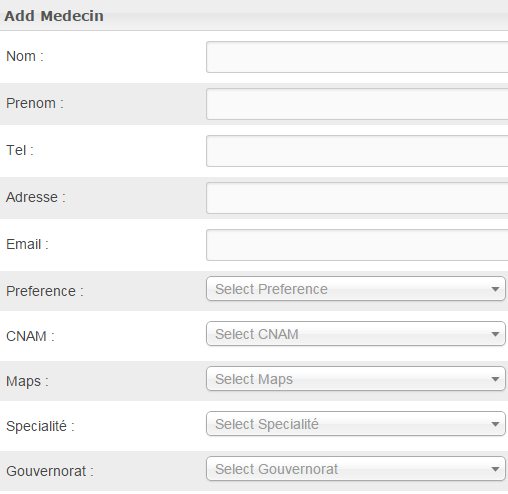
****

Figure 51 : interface du cas d’utilisation «Modifier médecin»

Lors de la modification des informations d’un médecin, la fiche de celui-ci est affichée à l’écran pour que l’utilisateur fasse les changements nécessaires. La figure 49 montre un aperçu sur cette action.

****

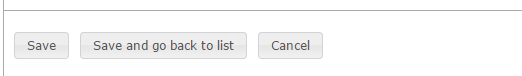
****

Figure 52 : interface du cas d’utilisation «Ajouter médecin»

La figure 50 n’est autre qu’une fiche à remplir pour ajouter un nouveau médecin.

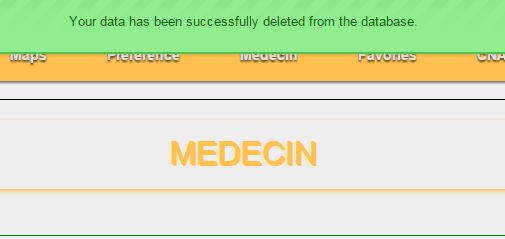
****

Figure 53 : interface du cas d’utilisation «Supprimer médecin»

Une fois que l’administrateur supprime un médecin de la base, un message de confirmation est affiché comme le montre la figure 51. Cette action est certes irréversible.

1. Conclusion

A ce stade, nous avons réussi donc à développer le dernier release de notre application pour arriver à un produit complet et fonctionnel. Notre application Web est donc prête à être exploiter en offrant à l’administrateur la possibilité de gérer convenablement toutes les tables de la base de données.

Chapitre 5

Clôture du projet

1. Introduction

La phase clôture ou de fermeture est la dernière phase dans le cycle de développement d’un logiciel avec Scrum. Cette phase est souvent appelé sprint de stabilisation. Les tâches effectuées pendant cette phase prédéfinies, et ils dépendent fortement du type de projet.

Pour notre projet, ce chapitre sera consacré pour la présentation des langages et outils de programmation utilisés pour la réalisation de nos deux applications.

1. Environnement de développement et langages de programmation :

L’environnement de développement est un terme qui désigne l’ensemble d’outils et de langage utilisé pour l’implémentation d’une solution informatique. Nous allons nous intéresser à l’environnement logiciel. Les logiciels utilisés pour l’implémentation de notre solution sont les suivants :

2.1 Application Mobile :

* Android Studio :



Figure 54 : Android Studio

Android Studio est un environnement de développement pour développer des applications Android. Il permet principalement d'éditer les fichiers JAVA et les fichiers de configuration d'une application Android[5]. Il propose entre autres des outils pour gérer le développement d'applications multilingues et permet de visualiser la mise en page des écrans sur des écrans de résolutions variées simultanément.

* JAVA :



Figure 55 : Java

C’est un langage de programmation informatique orientée Objet. La particularité et l'objectif central de Java est que les logiciels écrits dans ce langage doivent être très facilement portables sur plusieurs systèmes.

* 1. Application Web

#### Wamp Server

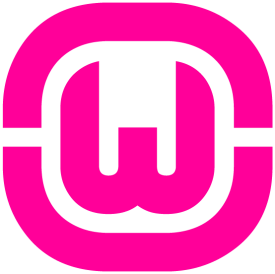


Figure 56 : Wamp server

Wamp server est une plateforme de développement des applications web dynamiques. Ce logiciel est très intéressant puisqu’il englobe tous les outils nécessaires pour le fonctionnement d’une application web notamment un serveur de base de données MySQL, un serveur web apache et une interface de gestion des bases de données facile à utiliser PHPMyadmin.

#### Filezilla



Figure 57 : Filezilla

Le but de développer des applications web c’est d’être publié sur le net par la suite. Pour ce faire, Filezilla propose un client FTP gratuit et simple à utiliser et qui permet à ses utilisateurs de se connecter à un serveur distant afin de publier leurs fichiers.

* CodeIgniter

http://upload.wikimedia.org/wikipedia/en/7/7c/CodeIgniter.png

Figure 58 : CodeIgniter

CodeIgniter est un Framework PHP Open Source et qui suit le modèle de conception MVC , c’est un Framework très puissant qui permet la création facile et rapide d’application web. La communauté de ce Framework est très active ce qui permet de trouver de l'aide très rapidement. De plus, les membres de la communauté de CodeIgniter ont développé de nombreuses bibliothèques réutilisables dont Grocery CRUD qui nous a été très utile pour notre projet.

* PHP



Figure 59 : PHP

PHP ou Hyper Text Preprocessor est un langage de script extrêmement puissant et destiné pour le développement des applications web. PHP est l’un des langages de programmation les plus populaire. Le point fort de ce langage c’est qu’il est portable et simple à utiliser.

1. Perspectives

Les perspectives d’évolution d’un tel système sont certes infinies vue le progrès technologique de plus en plus rapide et l’adaptation aussi rapide des utilisateurs à ses technologies.

En effet l’application mobile médecin sert à rapprocher le médecin de ses patients et de ce fait, nous pouvons penser à développer toute une logique médecin afin de pouvoir avoir leur propre compte et de se rendre encore plus disponibles de ses patients. Ce travail l’aidera à mieux organier ses rendez-vous et à pouvoir interagir avec ses patients d’une manière plus soutenue.

D’autre part, du côté utilisateur nous pouvons ajouter des fonctionnalités comme la gestion des rendez-vous avec la possibilité de les enregistrer sur l’agenda et de créer des rappels pour assurer leur suivi.

Cette application peut être un allié santé indispensable pour tout individu et devenir même un atout dont personne ne peut se passer.

Conclusion Générale

Dans ce rapport, nous avons abordé le domaine de l’e-santé afin de mieux comprendre l’attente d’un utilisateur d’une telle application afin de produire une application mobile de qualité. Cette étude nous a permis de constater la diversité des services pouvant être mis à disposition d’où complexité de se positionner par rapport à cet univers qu’on peut qualifier de novice en Tunisie.

En partant de cela, nous avons songé à introduire une nouvelle application mobile qui s'intitule "Médecin". Cette application peut pallier au manque d’information sur les services médicaux disponible surtout les médecins à consulter en cas de besoin ou même d’urgence.

De ce fait, notre application engloba tous les médecins de différentes spécialités exerçant en Tunisie. Cela nous a permis de mettre à la disposition des utilisateurs une riche base de données de médecins qu'ils puissent y avoir recours en cas de besoin. Ainsi plusieurs autres services ont été conçus afin d’accompagner les utilisateurs lors de leur recherche et les guider convenablement tout en assurant la possibilité de sauvegarder des informations utiles si nécessaire. Cette application n’est pas une innovation certes , mais les applications existantes ne sont pas efficace à cause du manque de leur pertinence ce qui nous a poussé une solution pour remédier à cela en offrant à l’utilisateur une solution perspicace d’où l’idée de l’application Web backoffice , qui même si elle n’est pas visible à l’utilisateur , assure qu’il obtienne toujours l’information correcte et pertinente sans le contraindre à télécharger une nouvelle version de l’application dès qu’il y a un changement d’adresse ou de numéro de téléphone. Ainsi l’utilisateur peut se fier à l’application et en faire son allié santé.

Nous avons entamé le projet avec une étude préalable pour nous éclairer sur les fonctionnalités à réaliser puis en nous basant sur la méthodologie SCRUM nous avons détaillé le système et planifié les release et sprint qu’ils contiennent pour enfin s’attaquer en boucle à la productions des incréments. De l’analyse en passant à la conception, codage puis test, nous avons enchainé le traitement des cas d’utilisation jusqu’à l’aboutissement du produit final, stable et fonctionnel. Bien que notre système soit vu comme un atout indispensable pour les utilisateurs ainsi que pour tous médecins désirant se faire plus disponible, plusieurs perspectives de travaux futurs peuvent être envisagées pour améliorer les performances de notre application.

# Bibliographie

|  |  |
| --- | --- |
|  |  |
|  | [1]C. Aubry, SCRUM le guide pratique de la méthode agile la plus populaire, Dunod, 2010.  [2] F. V. Pascal Roques, UML2 en action de l'analyse des besoins à la conception, edition Eyrolles, 2007.  [3]Institut supérieur de gestion deTunis, cours architecture logicielle , 2015 |