



ECOLE SUPÉRIEURE PRIVÉE D'INGÉNIERIE ET DE TECHNOLOGIES

www.esprit.tn - E-mail : contact@esprit.tn

Siège Social : 18 rue de l'Usine - Charguia II - 2035 - Tél. : +216 71 941 541 - Fax. : +216 71 941 889

Annexe : Z.I. Chotrana II - B.P. 160 - 2083 - Pôle Technologique - El Ghazala - Tél. : +216 70 685 685 - Fax. : +216 70 685 454

PROJET DE FIN D'ÉTUDES

DIPLOME NATIONAL D'INGÉNIEUR

SPÉCIALITÉ : INFORMATIQUE

Conception et développement d'une plateforme de gestion d'agriculture pour le projet « SEABEX »

Réalisé par : Walid BIZID

Encadrant ESPRIT : Mme Ines CHANNOUFI

Encadrant Entreprise : M. Taher MESTIRI



REMERCIEMENTS

Le plus dur n'est pas de rédiger le rapport mais de remercier toutes les personnes qui nous ont soutenues pour réaliser ce projet. Je tiens tout d'abord à remercier ma directrice de stage **Mme Ines CHANNOUFI** d'avoir accepté de m'encadrer, le manager de I.T GRAPES **M. Taher MESTIRI**, de m'avoir acceptée pour effectuer ce stage de projet de fin d'études, pour son temps précieux et ses précieux conseils tout au long du déroulement de ce stage. Je remercie également **M. Hedi LAHOUAR et M. Mehdi JARRAYA**, ingénieurs chez I.T GRAPES, avec qui j'ai travaillé étroitement durant mon stage, pour ses conseils et discussions fructueuses. Tous les personnels de la société I.T GRAPES (Ines, Amira, Fadwa, Kadhem) pour leurs encouragements continus.

J'adresse également mes remerciements à l'ensemble des professeurs pour l'enseignement qu'ils m'ont apporté.

J'adresse mes profondes gratitude à ma famille qui a été toujours à mes côtés, pour leur soutien et leur encouragement.

DEDICACE

A Mes Très Chers Parents (Bouraoui & Lamia)

Tous les mots du monde ne sauraient exprimer l'immense amour que je vous porte, ni la profonde gratitude que je vous témoigne pour tous les efforts et les sacrifices que vous n'avez jamais cessé de consentir pour mon instruction et mon bien-être.

J'espère avoir répondu aux espoirs que vous avez fondés en moi.

Je vous rends hommage par ce modeste travail en guise de ma reconnaissance éternelle et de mon infini amour.

Que Dieu tout puissant vous garde et vous procure santé, bonheur et longue vie pour que vous demeuriez le flambeau illuminant le chemin de vos enfants.

Je dédie aussi ce travail à :

Mon Ami Mickeal pour ses encouragements incessants et son soutien et sa présence aux moments difficiles, à mes amis (el buvette) pour leurs soutiens et la bonne humeur aux moments difficiles.

À la mémoire de mes grands parents

Ce travail est dédié à mes grands-parents *Zakia* et *Hcuria*, qui m'ont toujours poussé et motivé dans mes études. Je vous aime de tout mon cœur.

"If you can't support us when we lose. You can't support us when we win "

Sir Alex Ferguson

Table des matières

Introduction générale.....	1
Cadre générale	4
I. Introduction :	4
II. Présentation de l'organisme I.T. GRAPES :	4
1. Présentation :	4
2. Services :	4
3. Exemples Projets :	5
III. Présentation du projet :	5
1. Objectifs du projet	5
2. Analyse de l'existant et problématique :	6
a. Solution proposée :	8
IV. Planning prévisionnel :	8
1. Planning du travail :	8
2. Approche du travail :	9
3. Méthodologie adoptée :	9
4. Planification du projet :	10
V. Conclusion :	11
Analyse et spécification des besoins	13
I. Spécification des besoins :	13
1. Acteurs :	13
2. Besoins fonctionnels :	13
3. Besoins non fonctionnels :	15
II. Diagramme des cas d'utilisations globale :	15
III. Cas d'utilisations raffinés :	17
1. Cas d'utilisation « gestion des sites » :	17
2. Cas d'utilisation « Gestion des stations » :	19
3. Cas d'utilisation « Gestion des plans d'irrigations » :	21
4. Cas d'utilisation « envoi d'une aide ou alerte » :	23
IV. Conclusion :	25
Etude conceptuelle.....	27
I. Introduction :	27

II.	Conception détaillée :	27
III.	Choix de la méthodologie de conception :	27
1.	Vue statique :	27
2.	Vue dynamique :	28
IV.	Développement du modèle statique :	28
1.	Diagramme ER :	28
2.	Diagramme des classes :	29
V.	Développement du modèle dynamique :	30
VI.	Conclusion :	31
Réalisation.....		32
I.	Introduction :	33
II.	Environnement du travail :	33
1.	Environnement matériel :	33
2.	Environnement logiciel :	33
3.	Architecture de l'application :	33
III.	Choix des outils de développement :	34
1.	React JS :	34
2.	Redux :	34
3.	Laravel :	35
4.	GraphQL :	36
5.	REST API :	36
6.	Material UI :	37
7.	PostgreSQL :	38
8.	Eloquent ORM :	39
9.	Autres :	39
IV.	Pourquoi Laravel :	39
1.	Le meilleur du PHP :	39
2.	La documentation :	40
3.	Autres :	40
V.	Pourquoi React JS :	41
1.	React JS est très rapide :	41
2.	Le javascript simple à écrire :	41

3. L'intelligibilité :	41
1. La communauté :	41
VI. Interfaces graphiques :	41
1. Login :	41
2. Inscription :	42
3. Gestion des sites :	44
4. Stations :	45
5. Plan d'irrigations :	47
6. Scenarios :	50
7. Logs :	51
8. Evénements :	51
9. Formules :	52
VII. Conclusion :	52
Conclusion générale	53

Tableau des figures

Figure 1: Logo iFarming	6
Figure 2: Logo Smart LOGGER	7
Figure 3: Logo Scanopy.....	7
Figure 4: Les phases et itérations de la méthode RUP [12]	10
Figure 5: Diagramme de Gant	10
Figure 6: Diagramme cas d'utilisation globale	16
Figure 7: cas d'utilisation gestion des sites.....	17
Figure 8: Diagramme des séquences ajout d'un site.....	19
Figure 9: cas d'utilisation gestion des stations.....	19
Figure 10: Diagramme de séquence création d'une station d'action	21
Figure 11: cas d'utilisation plans d'irrigations.....	21
Figure 12: Diagramme des séquences création d'un plan d'irrigation et tâches	23
Figure 13: cas d'utilisation envoi d'une aide ou alerte	24
Figure 14: Diagramme des séquences envoi d'une alerte ou aide.....	25
Figure 15: Diagramme des classes	29
Figure 16: Diagramme d'activité	31
Figure 17: Architecture de l'application	33
Figure 18: logo React JS	34
Figure 19: Logo Redux	35
Figure 20: Logo Laravel.....	36
Figure 21: Logo GraphQL	36
Figure 22: Logo REST API	37
Figure 23: logo Material UI	37
Figure 24: Logo PostgreSQL	38
Figure 25: Interface Login	42
Figure 26: Interface Inscription 1	43
Figure 27: interface inscription 2	44
Figure 28: Interface carte des sites	45
Figure 29: Interface liste des sites.....	45
Figure 30: Interfaces liste des stations	45
Figure 31: Interface position station	46
Figure 32: Interface équipements d'une station.....	46
Figure 33: Interface ajout station.....	47
Figure 34: Interface liste des plans d'irrigations	47
Figure 35: Interface détails d'un plan d'irrigation.....	48
Figure 36: Interface ajout d'un plan d'irrigation	48
Figure 37: Interface ajout des tâches	49
Figure 38: Interface ajout d'un scénario.....	50
Figure 39: Interface consultation du log.....	51
Figure 40: Interface consultation des évènements.....	51
Figure 41: Interface ajout et liste des formules	52

Liste des tableaux

Tableau 1: tableau descriptif de cas d'utilisation gestion des sites	17
Tableau 2: tableau descriptif de cas d'utilisation gestion des stations	20
Tableau 3: tableau descriptif de cas d'utilisation gestion des plans d'irrigations.....	22
Tableau 4: tableau descriptif de cas d'utilisation envoi d'une aide ou alerte	24
Tableau 5: tableau des api et bibliothèques utilisées	39

Introduction générale

Tout le monde aujourd'hui parle de l'agriculture 4.0[1] sans bien savoir à quoi ça consiste. Si nous retournons un peu en avant, la meilleure des définitions est de faire une analogie avec celle de l'industrie 4.0. L'acquisition d'une grande quantité de données et après, leur usage automatisé assisté d'une IA¹, doivent permettre à chaque exploitation d'optimiser ses performances économiques. Les changements sont en cours avec les interrogations qu'ils peuvent soulever.

L'expression "**agriculture 4.0**" a été créée par analogie au terme "industrie 4.0". On découpe généralement l'évolution du secteur industriel en quatre étapes. Dans les années 1880, il s'est transformé grâce aux machines à vapeur : c'était l'industrie 1.0. L'introduction de l'électricité sur les chaînes de montage correspond à l'industrie 2.0. La troisième phase de la transformation industrielle est caractérisée par l'avènement de l'informatique et l'automatisation des processus sur les chaînes d'assemblage. L'industrie 4.0, elle, regroupe les dernières mutations opérées grâce au digital, et à la possibilité d'interagir et de communiquer avec les différents matériels. L'agriculture a connu les mêmes changements.

Aujourd'hui en effet, toutes les machines-outils et tout l'environnement de la chaîne de production ont la possibilité de transmettre des informations en temps réel sur leur état et leurs performances. Ces informations centralisées dans l'usine permettent de commander les divers engins, en prenant en compte l'état des autres appareils. De cette façon, il est possible d'automatiser et robotiser une chaîne de production complète comprenant des robots travaillant sur le même produit, en même temps et de façon coordonnée. Le ramassage des poubelles illustre parfaitement cette problématique. Avant, un salarié passait plusieurs fois par jour pour vider des conteneurs parfois presque vides. Aujourd'hui, quand ces derniers sont pleins, ils appellent l'opérateur pour qu'il intervienne. D'où un gain de temps, donc de productivité.

¹ Intelligente Artificielle

L'**agriculture a rencontré les mêmes transformations que l'industrie** avec l'apparition de la machine à vapeur puis le développement du machinisme agricole. Aujourd'hui, l'ensemble des engins agricoles intègrent des commandes électroniques et sont entrés dans l'air du numérique, ce qui permet d'être informé de leur état et de leurs performances.

Par ailleurs, l'électronique s'est démocratisée, **les capteurs collectent des données sur la météo, les parcelles** et globalement toute la vie de la ferme. Actuellement, on peut "prendre le pouls" de n'importe quelle activité de l'exploitation agricole et la traduire en performances.

C'est la raison pour laquelle nous avons été amenés à concevoir une application informatique dont l'objectif premier est d'optimiser la gestion des ressources de la ferme.

Pour arriver à nos fins, nous avons utilisé un ensemble de notions relatives aux méthodes de conception des systèmes d'information et aux méthodes de conduite des projets. Pour réaliser ce rapport on va suivre la démarche suivante :

Chapitre 1 : Le premier chapitre est une prise de connaissance et une présentation du cadre du projet ainsi que la société. En ajoutant l'analyse du proje, les solutions existantes et la solution proposé.

Chapitre 2 : cette partie consiste à collecter, analyser et définir les besoins de haut niveau (les besoins fonctionnels et les besoin non fonctionnels) et présenter les diagrammes des cas d'utilisation et séquences.

Chapitre 3 : Conception, dans ce chapitre sera consacrée à la conception de l'application : il s'agit d'une phase de modélisation théorique de l'application.

Chapitre 4 : Réalisation, ce chapitre contient une description détaillée des outils utilisés pour développer l'application web, l'architecture du système et le matériel de déploiement de l'application. Il contient aussi quelques interfaces de l'application avec quelques explications.

Chapitre 1

Cadre générale

I. Introduction :

Dans le cadre de notre troisième année d'étude à l'Ecole Supérieure Privé d'Ingénierie et des Technologies, j'ai eu l'opportunité d'effectuer un stage de fin d'études chez IT GRAPES d'une période de six mois. La première étape est celle d'analyse et de spécification, est la première étape du processus de développement que nous avons adopté. En effet, elle met en place et détaille ce qui a été ébauché au cours de l'étude préliminaire, et permet de dégager l'étude fonctionnelle du système. Elle permet autrement d'obtenir une idée sur ce que va réaliser le système en termes de métier (comportement du système). Tout au long de ce chapitre, nous commencerons par définir les besoins fonctionnels et non fonctionnels de la solution proposer, ainsi que la planification du déroulement du travail durant la période de stage.

II. Présentation de l'organisme I.T. GRAPES :

1. Présentation :

Fondée depuis 2011, I.T. GRAPES est une agence débordante de l'innovation. Elle est formée d'une équipe de 07 personnes multidisciplinaires. Chacun est compétent dans son domaine. Petite équipe apparemment, ils ont travaillé sur plusieurs projets technologiques majeurs avec de grands comptes nationaux et gouvernementaux.

Qu'il s'agisse de systèmes d'information, temps réel, Iot, Big Data, Web et applications mobiles, l'équipe est prête, expérimentée et assoiffée d'aller de l'avant avec vos projets en tenant compte des derniers outils technologiques et en garantissant une adaptation graphique efficace.

2. Services :

- Logiciel :
 - Développement et conception des applications selon cahier de charge
 - Développement sur mesure
 - Sous-traitance des projets
- Internet :
 - Création et conception des sites et portails web
 - Maintenance des sites et portails web
 - Plan de communication et publicité

- Référencement
 - Solution E-Commerce
 - Solution pour magazine en ligne
 - Implantation de système Enquêtes (Formulaires, Statistiques)
- Conseil :

I.T. GRAPES élabore une méthodologie adaptée au projet et aux besoins de l'entreprise :

- Prise de contact
- Etude de votre projet
- Réalisation du cahier des charges
- Réalisation du projet
- Traitement des informations
- Remise du rapport final

3. Exemples Projets :

- Jawhara fm application mobile
- Ultimium caisse enregistreuse intelligente
- Ministère d'intérieure web application and mobile
- Assurances Maghreb application mobile
- SONEDE application mobile

III. Présentation du projet :

Le projet intitulé SEABEX consiste à réaliser une conception, développement et déploiement d'une solution de gestion et automatisation des ressources agricoles.

1. Objectifs du projet

- ✓ Conception du schéma de la base de données
- ✓ Conception des différents modules de l'application : backend et frontend
- ✓ Développement et codage de l'application (backend + quelques modules frontend)
- ✓ Tests et validation
- ✓ Déploiement

2. Analyse de l'existant et problématique :

Parmi les principales étapes d'étude d'un nouveau projet, l'étape de la collection des informations sur les concurrents principalement les plateformes existantes qui offrent les mêmes services que nous envisageons de réaliser, et même les tâches qui sont exécutées manuellement par les agriculteurs (ex : Ouverture/fermeture des électrovannes ...)

Les applications de nos jours, bien qu'elles soient performantes ne satisfont pas la plupart des clients. En effet certaines fonctionnalités ne se font pas de manière automatique, simple et facile (la gestion des plans d'irrigations, des scénarios, le log des données et événements etc.). Elles limitent les libertés des utilisateurs.

L'objectif visé est de satisfaire les utilisateurs en réduisant au maximum la charge de travail manuelle. Dans un souci de concevoir une application avec plus de fonctionnalités et dans le but d'avoir une interface plus conviviale et plus facile à utiliser tout en étant plus efficace, nous allons mettre sur pied une application informatique répondant à tous les besoins des différents acteurs intervenant dans le domaine d'agriculture.

Voici quelques exemples des solutions existantes :

2.1. L'échelle national :

a. iFarming :

iFarming, Start-up Tunisienne appartient au groupe Sofia Holding créée en 2017 et basée à Tunis. La société œuvre dans le domaine de l'agriculture, l'environnement et l'agro-alimentaire. iFarming a développé des applications web et mobile basées sur l'IoT (Internet des Objets) et l'intelligence artificielle.



Figure 1: Logo iFarming

b. Smart Logger :

SMART LOGGER est une startup Tunisienne nouvellement dévoilée par une équipe d'ingénieurs.

Grâce à son staff multidisciplinaire et à ses rapports étroits avec ses partenaires, elle fournit des solutions technologiques innovantes ayant pour objectif d'accélérer la transformation digitale de ses clients notamment dans les secteurs industriels, agricole et médical.



Figure 2: Logo Smart LOGGER

2.1. L'échelle internationale :

a. Scanopy :

Scanopy, entreprise implantée à Quincy, met ses compétences au service de la viticulture. Depuis 2015, ils travaillent à la construction d'outils de mesure et d'analyse agronomique pour les besoins spécifiques de la culture de la vigne. Ces services sont tournés vers le métier du viticulteur et sont développés en collaboration avec les exploitants et les organismes de conseil technique de la région Centre Val de Loire.



Figure 3: Logo Scanopy

En étudiants et analysons les solutions existantes :

- Absence de quelques fonctionnalités tel que la configuration des scenarios.
- Certaines solutions disposent de plusieurs modules décomposé chaque une dans une application.
- La maintenance demande dans la plupart de ces solutions un déplacement sur place d'où un retard.
- L'interface dans quelques solutions est difficile à utiliser pour les utilisateurs.
- La plupart des solutions sont chers

a. Solution proposée :

Notre solution consiste à développer une plateforme intelligente d'agriculture qui met à la disposition de l'agriculteur des différents outils automatique afin d'assurer la bonne gestion des ressources de sa ferme (eau, production, état du sol ...) et assure aussi la communication tous les acteurs intervenant dans ce domaine(agriculteur , expert, chercheur ...) en utilisant des mécanismes qui facilitent aux agriculteurs la gestion de ses ressources(ex : moins utilisation d'eau et plus de productivité, consultation des données en temps réel, service d'aide via sms).

Avec SEABEX, on assure l'automatisation des différentes tâches d'irrigations et la planification des différentes taches via une interface simple et facile à utiliser.

On offre la possibilité de configurer vos stations et les organisées dans des surfaces que vous dessinés directement sur la carte.

On offre aussi tout un historique détaillé des taches réalisées que ce soit automatique (plan d'irrigation, scenario) ou manuelle (ouverture/ fermeture d'un équipement comme électrovanne).

Consultation des données en temps réelles et du log de ces derniers avec une courbe qui montre l'évolution de ces données dans le temps.

Un service SMS pour tout agriculteur inscrit dans une organisation que ce soit client seabex ou pas afin de bénéficier d'un service d'aide et d'alerte.

Les principaux modules de notre solution :

- Module de configurations des areas et stations
- Module automatisation des taches
- Module Log et historique des données

IV. Planning prévisionnel :

1. Planning du travail :

La clé principale de la réussite d'un projet est un bon planning. En effet, le planning aide à bien subdiviser le travail et séparer les tâches à réaliser, il offre une meilleure estimation et gestion de temps nécessaire pour chaque tâche. De plus, il donne assez de visibilité permettant d'estimer

approximativement la date d'achèvement de chaque tâche. Dans notre projet, nous avons estimé de réaliser notre application dans une durée approximative de 120 jours (de 01/12/2018 jusqu'à 30/05/2019).

2. Approche du travail :

Le projet comprend deux phases : La première est la recherche d'une solution convenable pour réaliser l'application, et la deuxième est la conception et le développement.

- ✓ Phase de recherche : C'est l'étape qui inclut l'étude bibliographique, dans laquelle nous devons saisir les différentes notions et technologies à utiliser dans le projet, les architectures, etc. Aussi, elle renferme les tests des différentes solutions mises en hypothèse pour réaliser l'application, et nous fixons les outils nécessaires pour la réalisation du projet.
- ✓ Phase de conception et développement : C'est une étape, dans laquelle, nous spécifions les besoins fonctionnels et nous modélisons le système à réaliser pour clarifier les tâches à accomplir dans la partie développement. Cette phase se termine par une partie qui comprend la programmation et les tests de validation

3. Méthodologie adoptée :

Afin de réaliser ce projet, nous avons utilisés la méthode agile RUP² comme processus de développement logiciel. Parmi ses avantages on peut citer :

- Itérative.
- Centrée sur l'architecture.
- Pilotage par des cas d'utilisations et orientation vers la diminution des risques.
- C'est une méthode générique, itérative et incrémentale.

² Rational Unified Process

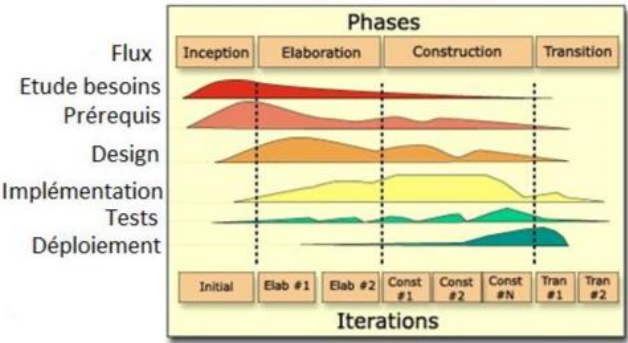


Figure 4: Les phases et itérations de la méthode RUP [12]

4. Planification du projet :

Afin de planifier notre travail et simplifier le suivi de son avancement, nous avons utilisé le diagramme de GANTT [2]. Le diagramme de Gantt est un outil utilisé en ordonnancement et en gestion de projet et permettant de visualiser dans les temps les diverses tâches composant un projet. Il s'agit d'une représentation d'un graphe connexe, évalué et orienté, qui permet de représenter graphiquement l'avancement du projet.

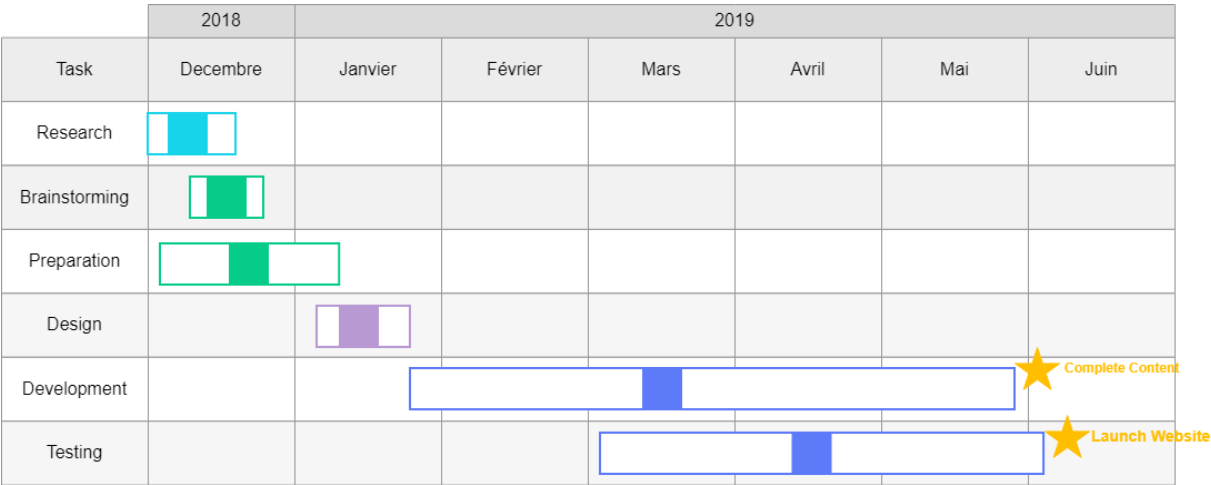


Figure 5: Diagramme de Gant

V. Conclusion :

Après avoir introduit le projet et la société qui m'a confié la réalisation de ce projet et j'ai pu l'insérer dans son contexte en présentant la solution adoptée pour résoudre les problèmes et qui répond aux besoins des intervenant dans le domaine d'agriculture.

Dans le chapitre suivant, nous allons présenter les besoins fonctionnels, non fonctionnels du projet ainsi que le diagramme des cas d'utilisations générale.

Chapitre 2 :

Analyse et spécification des besoins

I. Spécification des besoins :

Dans le cadre de cette partie, nous allons présenter les acteurs principaux de notre plateforme, la spécification de besoins qui consiste à la qualification des besoins fonctionnels et non fonctionnels attendus du système afin de mieux comprendre le projet ainsi que le diagramme des cas d'utilisations globale et raffinés.

1. Acteurs :

Dans cette section nous allons présenter les acteurs de notre plateforme et les fonctionnalités offerte à chacun d'eux. Notre analyse commencera par identifier les acteurs qui interagissent sur notre système.

Fermier : C'est un acteur principal dans notre plateforme, il a la possibilité de gérer ses sites ainsi que ses stations, plans d'irrigations et scénarios. Il peut aussi consulter toutes historique des données et événements. Au moment de l'inscription il peut s'inscrire au service de notification s'il est membre d'un organisme.

Organisme : C'est un acteur qui joue un rôle principal dans notre plateforme, il a la possibilité d'envoyer des notifications à ces membres, comme il peut consulter la liste de ses abonnées ainsi que les données en temps réel de leurs stations.

2. Besoins fonctionnels :

Les besoins fonctionnels représentent les actions que chaque acteur peut réaliser sur notre application. On distinguera ainsi trois acteurs : Les visiteurs, les membres SEABEX comme fermiers et les membres SEABEX comme organisme.

Les besoins fonctionnels des visiteurs : les visiteurs de notre application pourront réaliser les actions suivantes :

- S'inscrire : Pour bénéficier de plus d'avantages, les visiteurs pourront s'inscrire sur notre plateforme ou dans le service d'aide et alerte.
- S'authentifier : Après inscription, ces visiteurs pourront commencer à utiliser notre application et accéder à leurs espaces personnels.

Les besoins fonctionnels des membres fermiers :

- La gestion des sites : Cette fonctionnalité est primordiale dans notre plateforme pour bénéficier des autres fonctionnalités de la plateforme. Il faut commencer par ajouter les sites et les secteurs et les parcelles ensuite puissent les gérer.
- La gestion des stations et ses équipements : Cette fonctionnalité consiste à créer les différentes stations (passerelle, acquisition, action) et les attachés des équipements pour puissent les utilisées dans des autres fonctionnalités comme les plans d'irrigations.
- Consultation et manipulation des données en temps réel : A travers cette fonctionnalité l'utilisateur à l'opportunité de consulter les données (température, humidité...) des différents équipements attachés à ses stations d'acquisitions en temps réel, et voir l'état des ports des équipements (pompe, électrovanne...) reliés à ses stations d'actions.
- Configurations des plans d'irrigations : Cette fonctionnalité consiste à configurer les différents plans d'irrigations d'une façon qu'elles soient exécutées automatiquement sans interventions humaines.
- Configuration des scenarios : A travers cette fonctionnalité nous offrons à l'utilisateur l'opportunité de construire ses propres scénarios possibles afin de bien gérer ses ressources principalement en eau et matières premières.
- Consultation de logs : Ceci offre à l'utilisateur des consulté toutes les données enregistrées par les équipements attachés aux stations d'acquisitions (10HS, 5TE...) et voir son évaluation dans une durée bien déterminé.
- Consultation des évènements : Ceci offre à l'utilisateur la possibilité de consulté l'historique des tâches réalisées dans un site bien précis et à travers une passerelle (Gateway) bien précise.

Les besoins fonctionnels des membres organisme :

- Envoi des alertes et aide au abonnés : Cette fonctionnalité assure au membre de type organisme d'envoyer des notifications (alertes ou aide) aux abonnés de l'organisme.
- Consultation de log des aides et alertes : A travers cette fonctionnalité, l'utilisateur peut consulter l'historique des notifications qu'il a envoyé.

- Consultation de la liste des abonnées : A travers cette fonctionnalités l'utilisateur peut consulter la liste des abonnées à l'organisme.

3. Besoins non fonctionnels :

Les besoins non fonctionnels décrivent toutes les contraintes techniques, ergonomiques et esthétiques auxquelles est soumis le système pour sa réalisation et pour son bon fonctionnement. En ce qui concerne notre plateforme, nous avons dégagé les besoins suivants :

- UX / UI Design : L'application doit être ergonomique, et pour cela nous nous sommes inspirés du « Material UI ».
- Fiabilité : Les données fournies par l'application doivent être fiables
- Une solution ouverte et évoluée : l'application peut être améliorée par l'ajout d'autres modules pour garantir la souplesse, l'évolutivité et l'ouverture de la solution.
- Rapidité : L'application doit être rapide et légère pour fonctionner sur n'importe quel appareil qui supporte un navigateur.
- La disponibilité : l'application doit être disponible pour être utilisée par n'importe quel utilisateur.
- Sécurité : Tous les mots de passes sont cryptés en AES-256-CBC, un algorithme de cryptage robuste et utilisé partout. Il produit un résultat haché de 256 caractères. En plus d'être une fonction adaptative, c'est-à-dire que l'on peut augmenter le nombre d'itérations pour le rendre plus lent. Pour ce qui est des actions que va faire l'utilisateur sur notre application, on génère des tokens (jetons) après authentification, qui seront utilisés à la place des mots de passes pour prouver que le porteur du jeton est bien celui qu'il prétend être.

II. Diagramme des cas d'utilisations globale :

Le diagramme des cas d'utilisations permet de déterminer par rapport au vu d'acteur les fonctionnalités nécessaires dans le but de satisfaire les différentes exigences des clients. Il présente d'une façon propre et claire l'interaction entre le système et l'acteur en spécifiant les besoins de l'utilisateur.

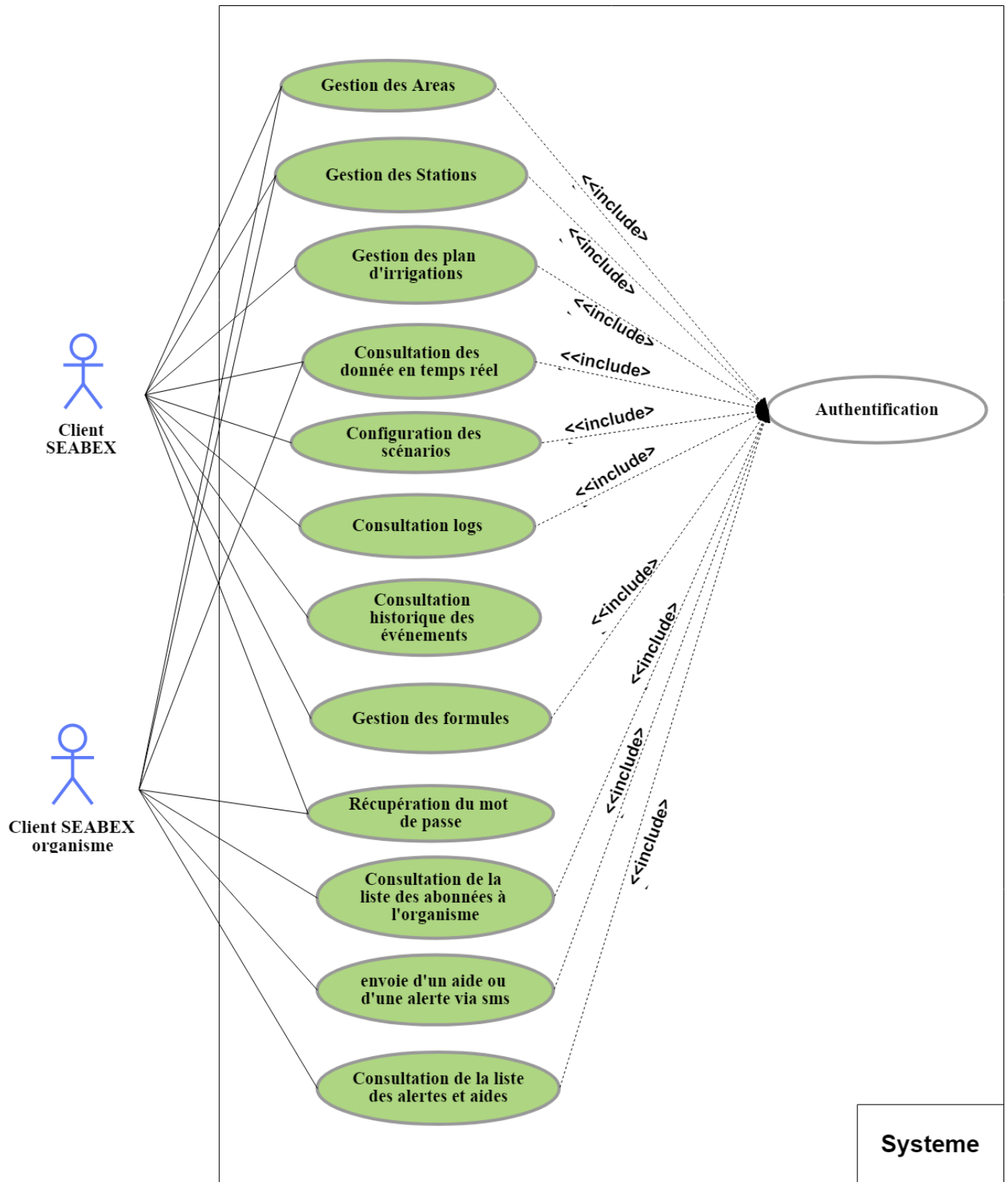


Figure 6: Diagramme cas d'utilisation globale

III. Cas d'utilisations raffinés :

Dans cette partie nous allons réaliser une étude détaillée de quelques cas d'utilisations. Nous allons diviser le cas d'utilisation globale en multiples cas d'utilisation fonctionnel à étudier pour assimiler le fonctionnement de notre plateforme, et présenter les différents scénarios modélisant l'aspect dynamique du système et ses interactions.

1. Cas d'utilisation « gestion des sites » :

1.1. Analyse de cas d'utilisation raffiné gestion des sites :

Nous présentons ci-dessous le diagramme de cas d'utilisation objet pour la gestion des sites.

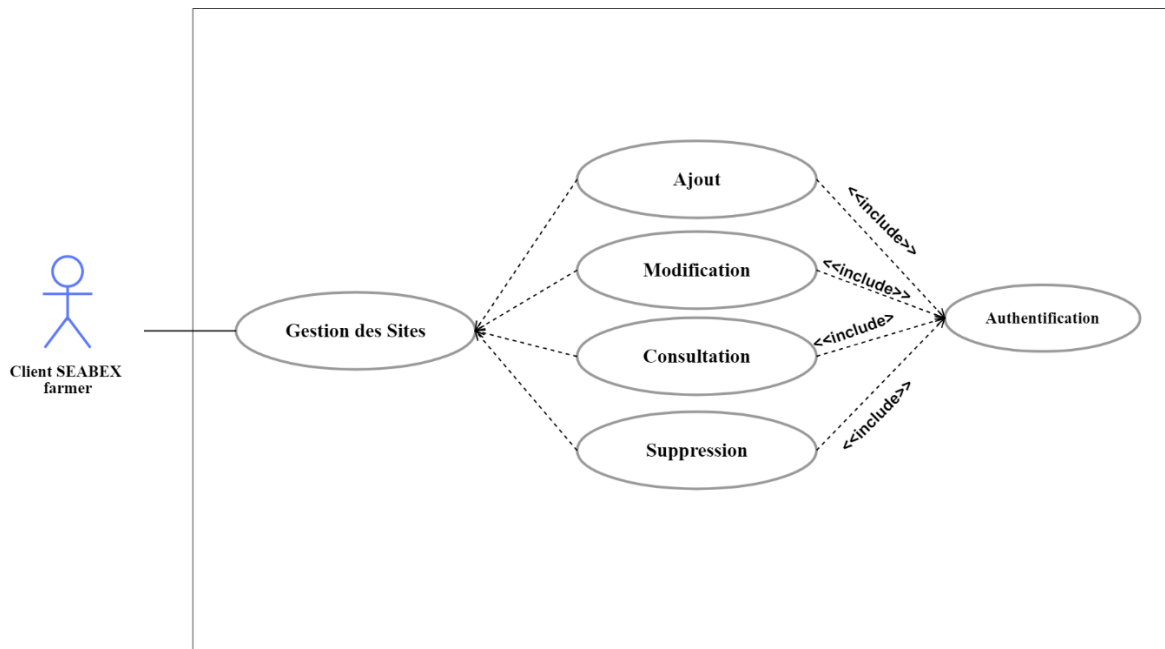


Figure 7: cas d'utilisation gestion des sites

Un tableau qui décrit le cas d'utilisation de la gestion des sites :

Tableau 1: tableau descriptif de cas d'utilisation gestion des sites

Sommaire

Fonction	Gestion des sites
Acteur	Client seabex fermier
Résumé	Le client dessine le polygone de son site, puis il doit remplir le formulaire respectif à la création d'un nouveau site et clique sur « sauvegarder » dans le menu « Sites ».

	<p>Il peut autrement créer un nouveau site en remplissant seulement le formulaire sans dessiner la surface mais en allant au menu « les zones agricoles »</p> <p>En modifiant un site, on peut modifier tout donnée relié à ce site.</p> <p>L'utilisateur peut aussi supprimer un site mais ce dernier sera supprimé seulement de l'interface de l'utilisateur et non pas de la base de données.</p>
--	--

Description des enchainements

Pré conditions	Que le client dispose d'une connexion à internet et soit authentifié.
Post conditions	Création/modification d'un nouveau site agricole sur notre plateforme. Modification ou suppression d'un site.

Scénario nominal

En dessinant la surface de notre site, puis en précisant le type d'area (site, parcelle ou secteur) et en remplissant le formulaire d'une nouvelle area, un nouveau site sera ajouté à notre carte dans la page d'accueil et la liste des sites (la parcelle doit être attaché à un site et un secteur doit être attaché à une parcelle)

Scénario alternatif

- Si l'utilisateur n'a pas précisé le site au moment de la création de la parcelle ou la parcelle au moment de la création du secteur une erreur sera produite.
- Si l'utilisateur n'a pas bien rempli les champs obligatoire une erreur se produit.

1.2. Diagramme de séquence objet de création d'un site :

Nous présentons ci-dessous le diagramme des séquences objets relative au cas d'utilisation création d'un site

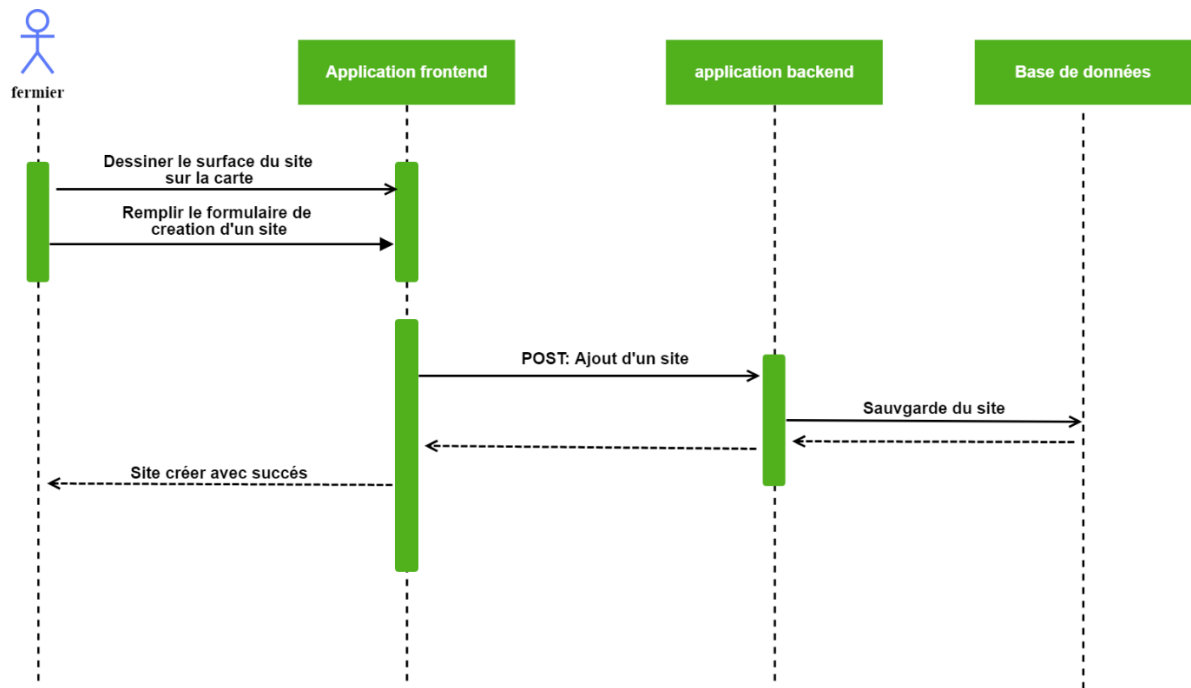


Figure 8: Diagramme des séquences ajout d'un site

2. Cas d'utilisation « Gestion des stations » :

2.1. Analyse de cas d'utilisation raffiné « gestion des stations » :

Nous présentons ci-dessous le diagramme de cas d'utilisation objet pour la gestion des stations.

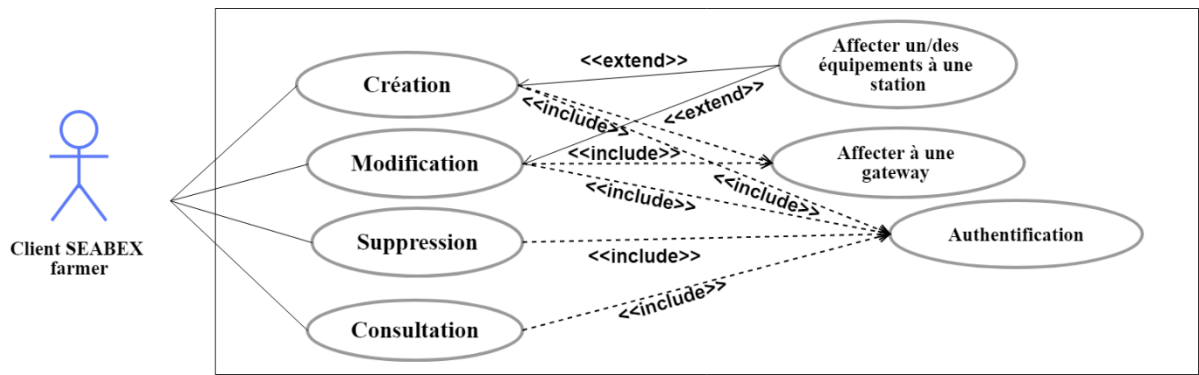


Figure 9: cas d'utilisation gestion des stations

Un tableau qui décrit le cas d'utilisation de la gestion des stations :

Tableau 2: tableau descriptif de cas d'utilisation gestion des stations

Sommaire	
Fonction	Gestion des stations
Acteur	Client seabex fermier
Résumé	<p>D'abord le client crée une station en remplissant le formulaire respectif à la station, en spécifiant le type de la station (Gateway, monitoring, action) le client peut affecter un ou plusieurs équipements à cette station, comme il peut l'associer à un site. Il peut aussi mettre la position de la station directement sur la carte.</p> <p>Chaque station peut être modifiée à travers le formulaire de modification.</p> <p>Chaque station peut être supprimée après confirmation (elle sera supprimée seulement de l'interface du client mais elle reste dans la base de données).</p> <p>Le client peut consulter la liste de ses stations avec ces équipements et sa position sur la carte.</p>
Description des enchainements	
Pré conditions	Que le client dispose d'une connexion à internet et soit authentifié.
Post conditions	Création/modification d'une nouvelle station sur notre plateforme. Modification ou suppression d'une station.
Scénario nominal	
<p>Le client remplit le formulaire de création/modification d'une (nouvelle) station en précisant l'identifiant, le site et le type. Puis selon le type de station (action, monitoring) il doit préciser les équipements associés et la gateway reliée à cette station, sinon si c'est une gateway il ne précise rien de plus.</p> <p>En cas de suppression la station sera supprimée après confirmation avec toutes ses équipements.</p>	
Scénario alternatif	
<ul style="list-style-type: none"> - Une station doit avoir un identifiant - Une station de type monitoring ou action doit être reliée à une passerelle (Gateway) 	

2.2. Diagramme des séquences objets « création d'une station » :

Nous présentons ci-dessous le diagramme des séquences objets relative au cas d'utilisation création d'une station d'action.

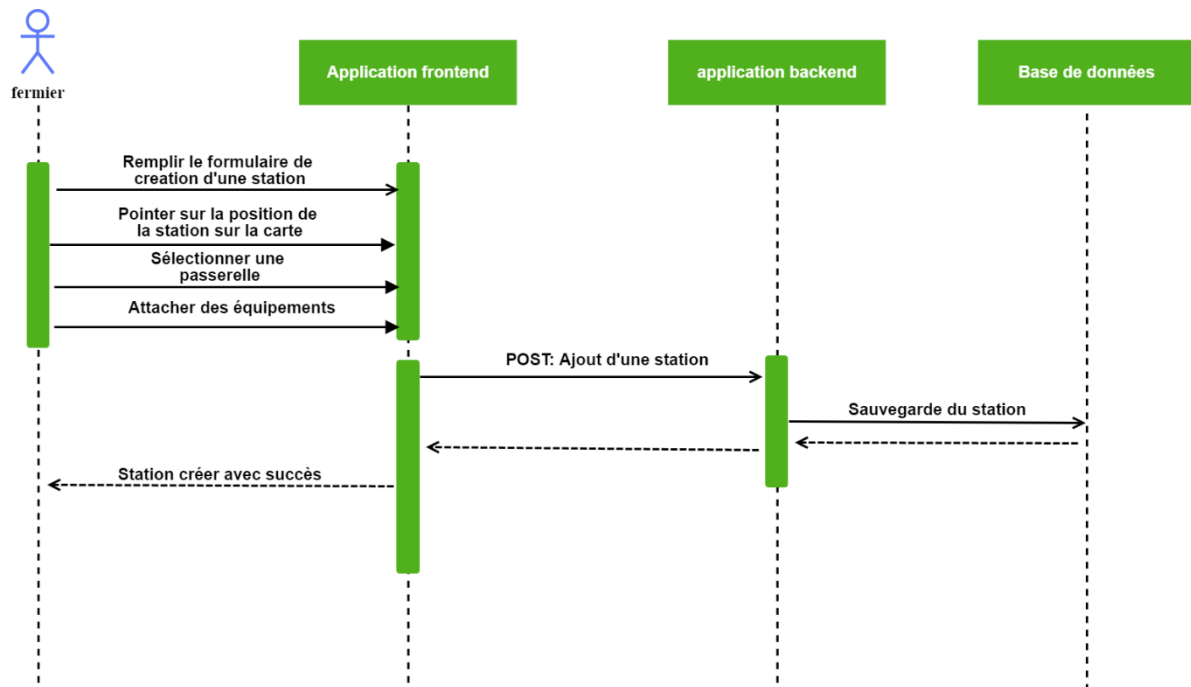


Figure 10: Diagramme de séquence création d'une station d'action

3. Cas d'utilisation « Gestion des plans d'irrigations » :

3.1. Analyse de cas d'utilisation raffiné « Gestion des plans d'irrigations » :

Nous présentons ci-dessous le diagramme de cas d'utilisation objet pour la gestion des plans d'irrigations.

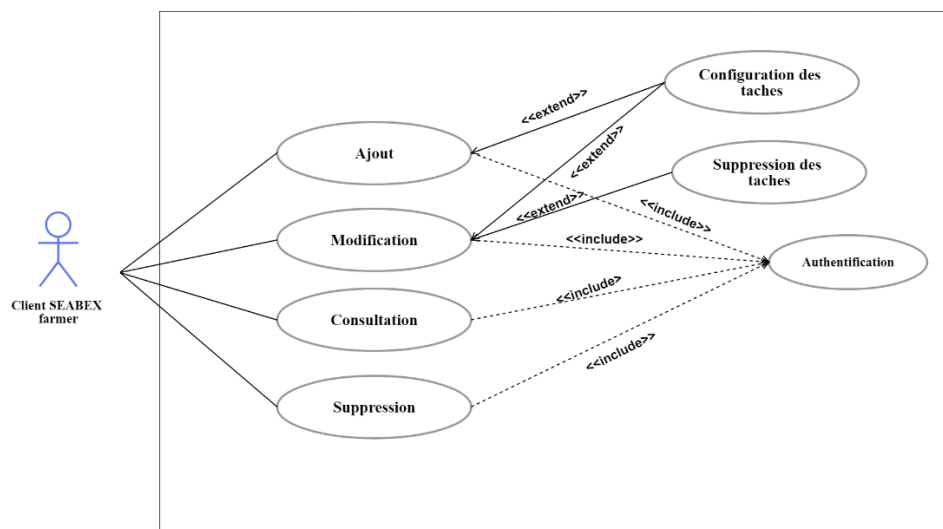


Figure 11: cas d'utilisation plans d'irrigations

Ce tableau décrit le cas d'utilisation de la gestion des plans d'irrigations :

Tableau 3: tableau descriptif de cas d'utilisation gestion des plans d'irrigations

Sommaire	
Fonction	Gestion des plans d'irrigations
Acteur	Client seabex fermier
Résumé	<p>Le client crée un plan d'irrigation à travers le formulaire, un plan d'irrigation est composé de plusieurs tâches.</p> <p>Après la configuration des différentes tâches, le plan d'irrigations sera envoyé pour être exécuté si son statut est « enabled ».</p> <p>Le client peut modifier une information concernant le plan d'irrigation via le formulaire mais s'il veut modifier une tâche il fallait recréer toutes les tâches.</p> <p>Il peut aussi supprimer un plan d'irrigation et cela va supprimer le plan d'irrigation avec ces tâches ou seulement les tâches en gardant le plan d'irrigations (tout plan d'irrigation ou tâche supprimé ne sera pas affiché pour le client mais elle n'est pas supprimée de la base de données).</p> <p>Le client peut aussi consulter la liste de ses plans d'irrigations avec ses tâches.</p>
Description des enchainements	
Pré conditions	<ul style="list-style-type: none"> -Que le client dispose d'une connexion à internet -Soit authentifié. -Dispose des stations d'actions et une passerelle
Post conditions	<p>Création/modification d'un plan d'irrigation avec des tâches bien spécifique à exécuté.</p> <p>Suppression d'un plan d'irrigation ou seulement des tâches.</p>
Scénario nominal	
<p>Le client remplit le formulaire du plan d'irrigation puis il crée les tâches reliées à ce plan en spécifiant la station, le port, le temps d'exécution de la tâche (heure début, heure fin, jour, mois, jour du mois)</p>	

Pour le cas de modification d'une des taches du plan d'irrigation, l'utilisateur es amené à reconfigurer toutes les taches.

En cas de suppression le plan d'irrigation et ses taches seront supprimé après confirmation seulement de l'interface utilisateur et non pas de la base de données.

Scénario alternatif

Les taches doivent avoir des stations d'action et des ports, une heure de début et une heure de fin et les dates, les mois, les jours du mois

3.2. Diagramme des séquences objets « Création d'un plan d'irrigation » :

Nous présentons ci-dessous le diagramme des séquences objets relative au cas d'utilisation création des plans d'irrigations.

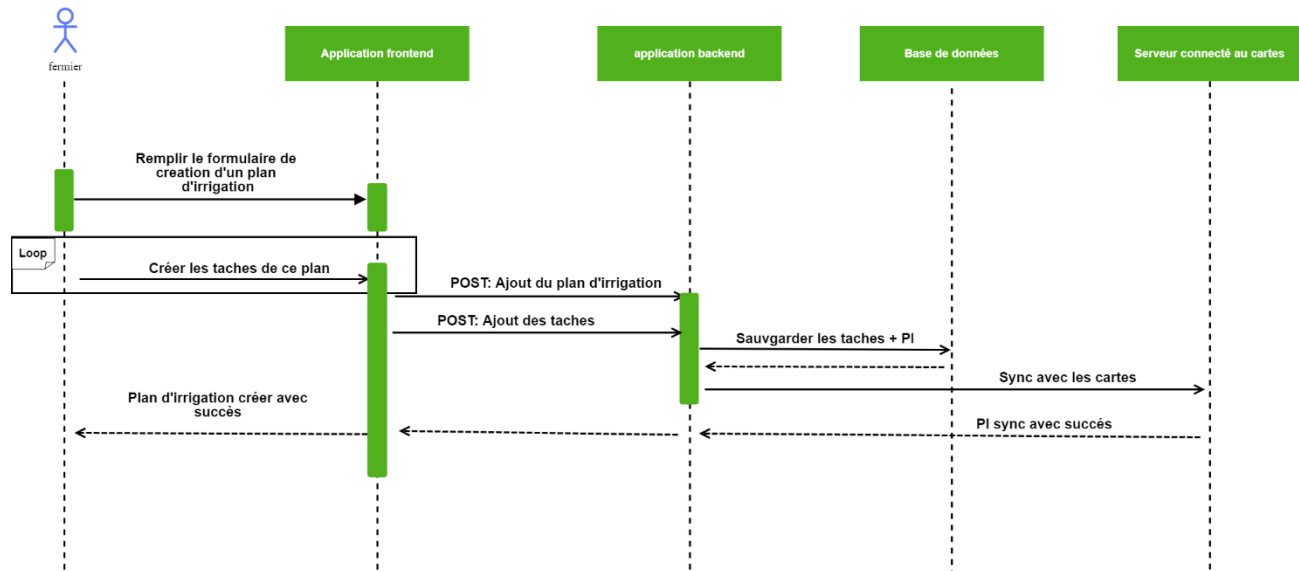


Figure 12: Diagramme des séquences création d'un plan d'irrigation et taches

4. Cas d'utilisation « envoie d'une aide ou alerte » :

4.1. Analyse de cas d'utilisation raffiné « envoie d'une aide ou alerte » :

Nous présentons ci-dessous le diagramme de cas d'utilisation objet pour l'envoi d'une aide ou alerte.

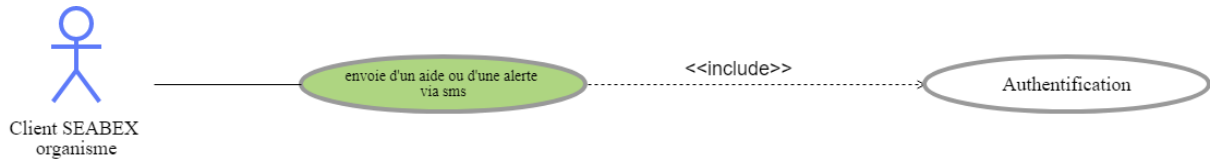


Figure 13: cas d'utilisation envoie d'une aide ou alerte

Ce tableau décrit le cas d'utilisation de l'envoi d'une aide ou alerte :

Tableau 4: tableau descriptif de cas d'utilisation envoie d'une aide ou alerte

Sommaire	
Fonction	Envoie d'une aide ou alerte au abonnés
Acteur	Client seabex organisme
Résumé	Le client seabex en tant qu'organisme peut envoyer une alerte ou aide à des abonnés spécifiques.
Description des enchainements	
Pré conditions	<ul style="list-style-type: none"> - Que le client dispose d'une connexion à internet - Soit authentifié. - Soit un client de type organisme
Post conditions	Envoie d'une alerte ou aide.
Scénario nominal	
<p>Le client commence par spécifier le type de notification qu'il va l'envoyer (alerte ou aide) puis le type de culture intéressé par cet alerte ou aide, ensuite il choisit un titre et saisie le message à envoyer, l'application lui généré le nombre d'abonnés qui vont recevoir la notification et finalement en cliquant sur envoyer la notification sera envoyé aux abonnés.</p>	

4.2. Diagramme des séquences objets « envoie d'une aide ou alerte » :

Nous présentons ci-dessous le diagramme des séquences objets relative au cas d'utilisation envoie d'une aide ou alerte

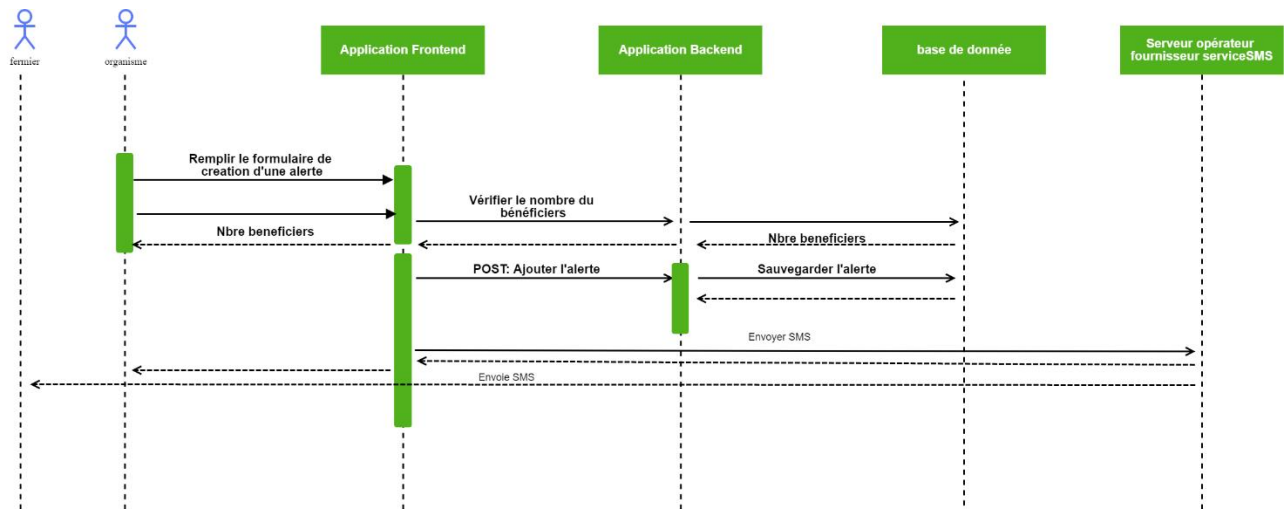


Figure 14: Diagramme des séquences envoi d'une alerte ou aide

IV. Conclusion :

Dans ce chapitre, nous avons conçu et documenté une partie des spécifications des besoins fonctionnel et non fonctionnels ainsi que les différents diagrammes qui revient au travail que nous devons produire. Le produit obtenu est un modèle graphique (ensemble de diagrammes) prêt à être codé.

Chapitre 3

Etude conceptuelle

I. Introduction :

UML4 est un langage de modélisation des systèmes standard, qui utilise des diagrammes pour représenter chaque aspect d'un système en s'appuyant sur la notion d'orienté objet qui est un véritable atout pour ce langage. Il est certes que nous adoptons UML comme un langage de modélisation pour notre projet puisque nous allons utiliser le concept de l'orienté objet et le concept de modèle MVC5. Dans ce chapitre nous aborderons la partie conception du projet, dans laquelle, nous détaillerons les différents éléments de conception, à savoir les diagrammes de cas d'utilisations, diagrammes des séquences et les diagrammes de classes.

II. Conception détaillée :

La conception est la partie la plus importante étape du cycle du développement. Elle se base sur une bonne spécification et l'analyse des besoins. Notre démarche commence par une compréhension du problème. Après, une analyse est importante pour donner une solution optimale.

III. Choix de la méthodologie de conception :

La modélisation nous aide à établir une représentation virtuelle d'une réalité d'une façon à mettre en premiers les points auxquels nous nous intéressons. Elle nous aide à déterminer un ensemble des données essentiels pour le développement de l'application. L'UML est une méthode de travail particulière puisqu'il s'intègre dans le processus de développement logiciel d'une façon transparente, pour améliorer progressivement les méthodes de travail, tout en gardant les modes de fonctionnement, ce qui nous aide de tirer deux types de vues :

1. Vue statique :

Une vue statique décrit notre système d'une manière physique :

- **Diagrammes de classes** : décrit l'ensemble d'éléments qui décrit une représentation statique (classes, paquetages...), ainsi que la structure d'un modèle.
- **Diagrammes d'objets** : c'est le diagramme qui montre les objets et des liens entre objets.
- **Diagrammes de cas d'utilisation** : identifient acteurs et leurs échanges avec le système.

- **Diagrammes de déploiement** : présente la disposition physique du matériel du système et la répartition des composants sur ce matériel.

2. Vue dynamique :

La vue dynamique explique le fonctionnement du notre solution avec :

- **Diagrammes de collaboration** : présente le contexte des interactions entre objet de systèmes (instances de classes et acteurs).
- **Diagrammes de séquence** : permettent de décrire comment les éléments collaborent selon un ordre chronologique (envois de messages).
- **Diagrammes d'activités** : (une variante des diagrammes d'états-transitions) permettent l'accent sur comportement d'une méthode ou déroulement d'un cas d'utilisation avec une représentation graphique.

IV. Développement du modèle statique :

Le développement du modèle statique est la première activité de la conception. Lors de cette étape, nous allons présenter le diagramme ER (entité-relation) et celui de déploiement.

1. Diagramme ER :

Nous décrivons dans la figure suivante le diagramme entité relation du serveur Back de notre plateforme. Ce diagramme contient toutes les tables spécifiques tels que celles de gestion des plans d'irrigations, scénarios, areas etc...

2. Diagramme des classes :

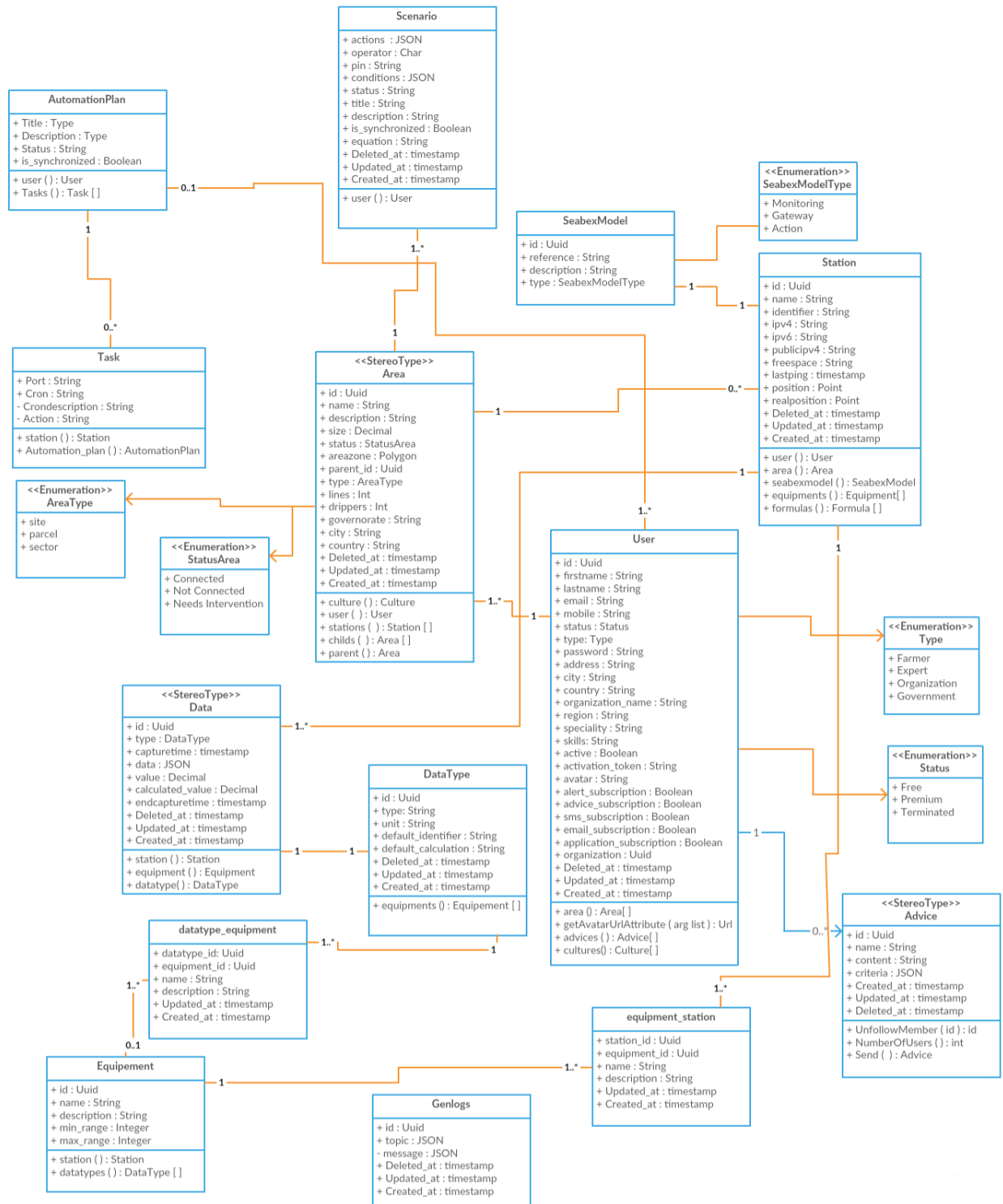


Figure 15: Diagramme des classes

Le diagramme de classe [13] représente les classes intervenant dans le système. Le diagramme de classe est une représentation statique des éléments qui composent un système et de leurs relations. Chaque application qui va mettre en œuvre le système sera une instance des différentes classes qui le compose. A ce titre il faudra bien garder à l'esprit qu'une classe est un modèle et l'objet sa réalisation.

V. Développement du modèle dynamique :

C'est l'étape suivante pour la réalisation de la conception. Cette phase est en relation direct avec l'activité de modélisation statique. C'est pourquoi, nous présentant les différentes interactions entre les objets de notre application. Effectivement, le modèle dynamique utilisé est : le diagramme de séquence décrit dans le chapitre qui précédé, puis un diagramme d'activité pour notre application. Ainsi on a réalisé le diagramme d'activité du user de type organisme :

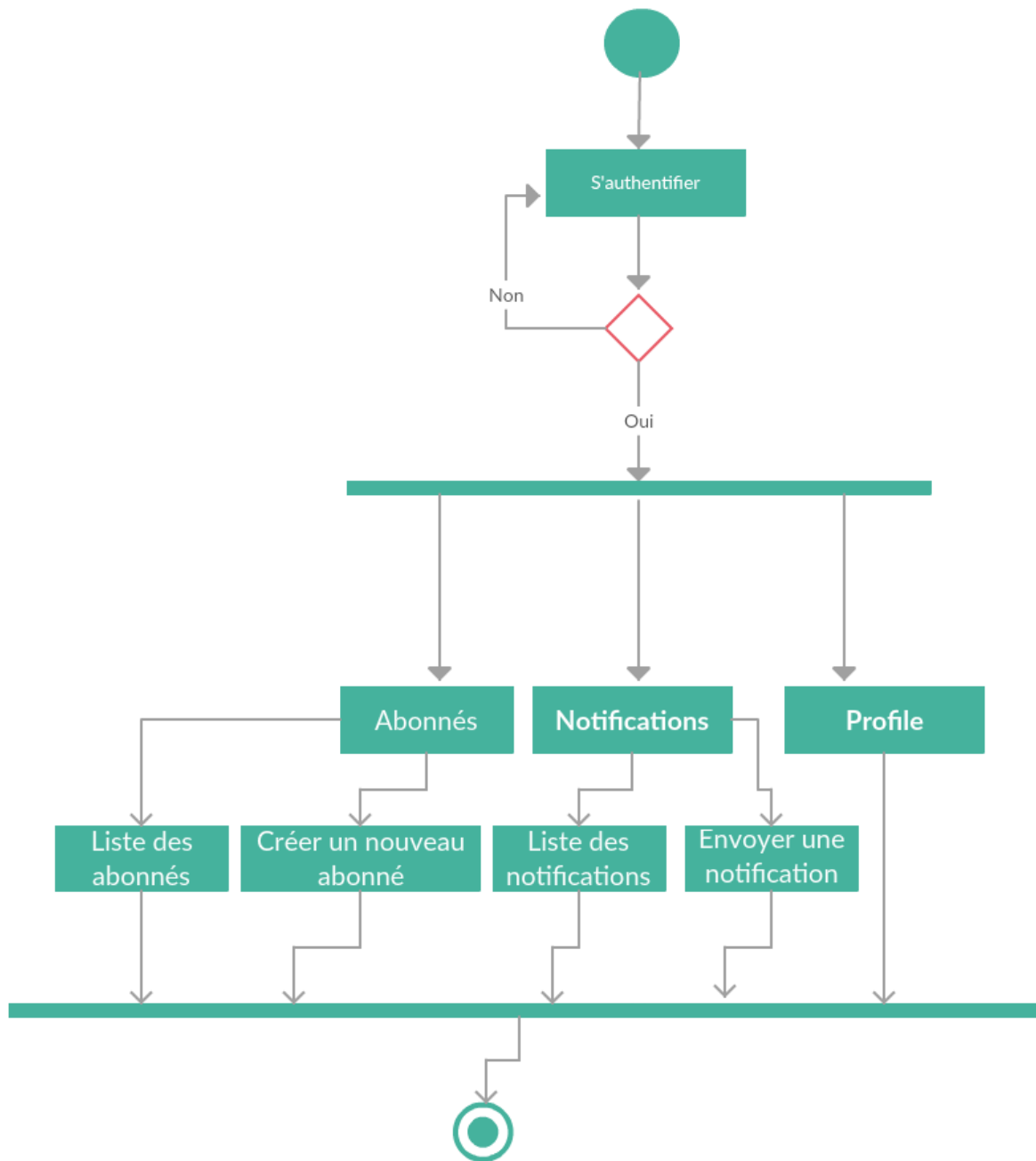


Figure 16: Diagramme d'activité

VI. Conclusion :

Dans ce chapitre, nous avons conçu et documenté une partie du code que nous devons produire.

Le produit obtenu est un modèle graphique (ensemble de diagrammes) prêt à être codé.

Chapitre 3

Réalisation

I. Introduction :

Dans ce chapitre, nous présenterons l'environnement matériel et logiciel du projet. Ensuite nous nous intéresserons à la description des outils de développement. Finalement, nous allons présenter quelques interfaces de l'application, répondant aux recommandations ergonomiques de compatibilité, de guidage, de clarté, d'homogénéité et de souplesse.

II. Environnement du travail :

1. Environnement matériel :

Machine de développement :

Pc Portable Asus X550JK : Processeur Intel Core i7-4720HQ Quad-Core 4é
Génération, up to 3.6 GHz, Mémoire 12 Go DDR3L

2. Environnement logiciel :

- Système d'exploitation : Windows 10 Pro
- Base de données : PostgreSQL
- IDE de développement : Visual studio code
- Outils de déploiements : WinSCP et PuTTY SSH
- Outil de conception : Cacao - Schémas et collaboration en temps réel
- Traitement et modification d'image : Snagit- the Windows Screen Capture Utility version 2019.1.1
- Réalisation des maquettes : Balsamiq Mockups version : 3

3. Architecture de l'application :

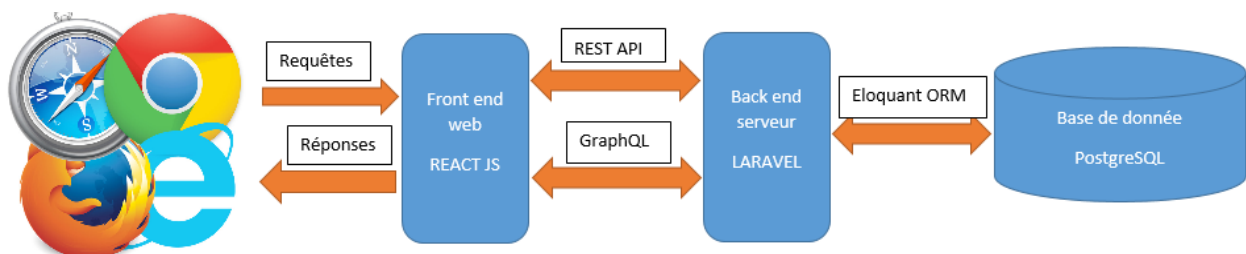


Figure 17: Architecture de l'application

III. Choix des outils de développement :

Pour la réalisation de ce projet nous avons choisi de travailler avec :

- ✓ React JS
- ✓ Redux
- ✓ Laravel
- ✓ GraphQL
- ✓ REST API
- ✓ Material UI
- ✓ PostgreSQL
- ✓ Eloquant

1. React JS :

React JS[3], est une techno Javascript, mais il ne s'agit pas d'un framework à proprement parler. En fait, il s'agit plus d'une librairie open source qui permet de construire des interfaces utilisateur dynamique. Et, de plus, cette librairie est maintenue un grand nombre de développeurs indépendants. Pour coder en React, vous devez avoir de bonnes bases en javascript, bien évidemment. C'est par ailleurs une techno très appréciée par développeurs web ces derniers temps. Elle permet de réaliser des applications web cross platform et ultra-performantes. Ce sont les équipes de Facebook qui sont à l'origine de cette plateforme. En effet, c'est en 2013 qu'est né React.

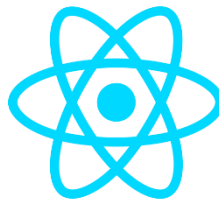


Figure 18: logo React JS

2. Redux :

Redux[4] gèrera l'état pour les applications JavaScript. Vous pouvez utiliser Redux principalement avec React ou autres framework et bibliothèque. Il possède un large écosystème d'add-ons utilisable.



Figure 19: Logo Redux

3. Laravel :

Laravel[5], créé par Taylor Otwell, initie une nouvelle façon de concevoir un framework en utilisant ce qui existe de mieux pour chaque fonctionnalité. Par exemple toute application web a besoin d'un système qui gère les requêtes HTTP. Plutôt que de réinventer quelque chose le concepteur de Laravel a tout simplement utilisé celui de **Symfony** en l'étendant pour créer un système de routage efficace. De la même manière l'envoi des emails se fait avec la bibliothèque SwiftMailer. En quelque sorte Otwell a fait son marché parmi toutes les bibliothèques disponibles. Nous verrons dans ce cours comment cela est réalisé. Mais Laravel ce n'est pas seulement le regroupement de bibliothèques existantes, c'est aussi de nombreux composants originaux et surtout une orchestration de tout ça.

Vous allez trouver dans Laravel :

- Un système de routage perfectionné (RESTFul et ressources),
- Un créateur de requêtes SQL et un ORM performants,
- Un moteur de template efficace,
- Un système d'authentification pour les connexions,
- Un système de validation,
- Un système de pagination,
- Un système de migration pour les bases de données,
- Un système d'envoi d'emails,
- Un système de cache,
- Une gestion des sessions...

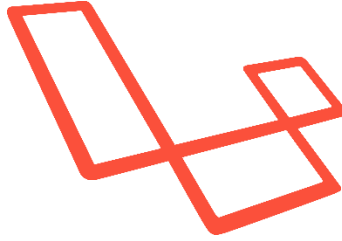


Figure 20: Logo Laravel

4. GraphQL :

GraphQL [6] est un langage de requête pour votre API et un environnement d'exécution côté serveur permettant d'exécuter des requêtes à l'aide d'un système de types que vous définissez pour vos données. GraphQL n'est lié à aucune base de données ni à aucun moteur de stockage spécifique, mais est soutenu par votre code et vos données existants.

Un service GraphQL est créé en définissant des types et des champs sur ces types, puis en fournissant des fonctions pour chaque champ de chaque type.

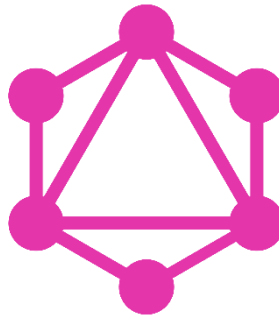


Figure 21: Logo GraphQL

5. REST API :

Une API REST [7] est le type le plus utilisé et le plus fréquent d'API sur internet. REST signifie "REpresentational State Transfer" et *wikipedia* vous dira tout sur le sujet si vous souhaitez l'approfondir.

Ce qu'il nous semble important ici de retenir est le suivant. Utiliser une API REST sur internet, c'est interroger un *serveur* tiers en utilisant les mêmes méthodes que ce que propose l'affichage de page web ou des formulaires inclus dans ces mêmes pages web. Ainsi on va interroger ce serveur à partir d'une *URL* communiquée par l'éditeur de l'API.

Cette interrogation va se faire suivant différentes méthodes : GET, POST, PUT et DELETE. Pour qui parle un peu anglais, le nom de la méthode indique parfaitement l'action qu'on va vouloir réaliser ! En général, c'est GET (obtenir) et POST (poster, donc envoyer) qui sont le plus utilisées. D'ailleurs si vous utilisez internet, vous faites du GET et du POST tous les jours sans le savoir : demander l'affichage d'une *page web* à partir de son *adresse*, c'est faire une action GET. Remplir un formulaire sur une page et le valider, c'est faire une action de type POST.

Point important maintenant, il ne s'agit pas simplement de faire des requêtes vers une API, il faut aussi savoir ce qu'elles provoquent ! En règle générale, une requête provoque en retour la réception d'une réponse. Cette réponse, comme une page web l'est pour une adresse demandée dans un navigateur, est une suite de données renvoyées à l'expéditeur avec le résultat de la requête. Deux grands formats de retour d'information sont utilisés : le *JSON* et le *XML*. Parfois on peut choisir dans une API quel format recevoir en retour.



Figure 22: Logo REST API

6. Material UI :

Material UI [9] est le React UI³ framework le plus populaire. Ils comportent des composants React pour un développement web plus rapide et plus simple. Confectionner votre propre thème graphique ou aidez-vous de Material Design. Material-UI est un projet open source sous licence MIT.



Figure 23: logo Material UI

³ User Interface

7. PostgreSQL :

PostgreSQL [10] est un système de gestion des bases de données relationnelles objet (ORDBMS) fondé sur POSTGRES, Version 4.2. Il a été développé à l'université de Californie au département des sciences informatiques de Berkeley. Postgres offre des fonctionnalités riches puisqu'il supporte la plupart du standard SQL :

- requêtes complexes
- clés étrangères
- triggers
- vues modifiables
- intégrité transactionnelle
- Contrôle des versions concurrentes (MVCC⁴).

De plus, PostgreSQL est extensibles par les utilisateurs, en ajoutant, par exemple :

- De nouveaux types de données
- De nouvelles fonctions
- De nouveaux opérateurs
- De nouvelles fonctions d'agrégat
- De nouvelles méthodes d'indexage
- De nouveaux langages de procédure.

Puisqu'il possède une licence libre, PostgreSQL™ peut être utilisé, modifié et distribué librement, quel que soit le but visé, qu'il soit privé, commercial ou académique.



Figure 24: Logo PostgreSQL

⁴ MultiVersion Concurrency Control

8. Eloquent ORM :

L'ORM Eloquent [8] fournit avec Laravel développé d'une manière simple par ActiveRecord pour manipuler votre base de données. Chaque table dans votre BD⁵ est relié à un modèle afin de pouvoir le manipuler.

9. Autres :

Tableau 5: tableau des api et bibliothèques utilisées

Nom	Bibliothèque/API	Description
Tunisie SMS	API	Tunisie SMS est une agence de communication digitale mobile via une plateforme d'envoi de campagnes SMS.
Recharts	Bibliothèque	Bibliothèque des charts construit avec React et D3
GuzzleHttp	Bibliothèque	PHP client http pour envoi et réception des web services
Gmail	API	Api d'envoi des emails développés par Google
Math-JAX	Bibliothèque	Bibliothèque javascript responsable à la manipulation des notions mathématique dans le navigateur.
React Leaflet	Bibliothèque	Bibliothèque javascript des cartes géographiques.

IV. Pourquoi Laravel :

1. Le meilleur du PHP :

Plonger dans le monde du codage en Laravel c'est accéder à un cours de développement web tant le style est clair et élégant et le code très organisé. La version courante de Laravel est la 5.8, elle nécessite minimum PHP 7.1.3 comme version. Pour traiter de façon efficace ce framework, il est préférable que vous ayez des connaissances sur ces notions :

⁵ Base de données

- **Les espaces de noms** : c'est une manière de bien organiser le code afin d'éviter les conflits de nommage. Laravel utilise beaucoup cette possibilité. Tous les composants sont mis dans des espaces de noms différents, similaires à l'application créée.
- **Les fonctions anonymes** : ce sont des fonctions sans nom (souvent appelées clôtures) qui permettent d'avoir une bonne qualité de code. Les développeurs de Javascript y sont familiers. Les développeurs de PHP un peu moins parce qu'elles y sont plus récentes. Laravel les utilise d'une manière automatique.
- **Les méthodes magiques** : ce sont des méthodes qui n'ont pas été explicitement décrites dans une classe mais qui peuvent être appelées et résolues.
- **Les interfaces** : une interface est un contrat de constitution des classes. En programmation objet c'est le sommet de la hiérarchie. Les modules de Laravel sont développés sur des interfaces.
- **Les traits** : c'est une manière d'agrandir les propriétés et méthodes à une classe sans passer par l'héritage, ce qui met des certaines limitations à l'héritage simple proposé par défaut par le langage PHP.

2. La documentation :

Quand on s'intéresse à un framework il ne suffit pas qu'il soit riche et performant, il faut aussi que la documentation soit à la hauteur. C'est le cas pour Laravel. Vous trouverez la documentation sur le site officiel.

3. Autres :

- Un framework fait gagner du temps et donne l'assurance de disposer de composants bien codés et fiables.
- Laravel est un framework novateur, complet, qui utilise les possibilités les plus récentes de PHP et qui est impeccablement codé et organisé.
- La documentation de Laravel est complète, précise et de plus en plus de tutoriels et exemples apparaissent sur la toile.
- Laravel adopte le patron MVC mais ne l'impose pas, il est totalement orienté objet.

V. Pourquoi React JS :

React est sans doute le framework coté client le plus fameux de nos jours, même s'il préfère se positionner comme une **bibliothèque** plutôt que comme un framework. En effet, React se focalise sur le cœur du problème : la gestion des interfaces utilisateur.

1. React JS est très rapide :

ReactJS crée son propre DOM virtuel où sont rattachés vos composants. Cette approche vous donne énormément de flexibilité et des performances exceptionnelles, car ReactJS calcule quel changement dans le DOM a besoin d'être fait, et modifié juste la partie qui a besoin d'être changé. De cette manière, ReactJS élimine les opérations coûteuses dans le DOM⁹.

2. Le javascript simple à écrire :

ReactJS utilise une syntaxe spéciale appelé JSX¹⁰, qui est un mixte de HTML¹¹ et JS¹². Ce n'est pas nécessaire – vous pouvez toujours développer votre application en Javascript pure – mais nous vous suggérons de coder avec cette nouvelle syntaxe car elle vous permet de coder vos composants très facilement. Être capable de mettre une touche de HTML dans vos fonctions de rendu sans avoir à concaténer vos chaînes, c'est fantastique.

3. L'intelligibilité :

ReactJS produit du code « propre » (simple à lire), sa lecture permet décomposer les fonctionnalités de votre application. Ce qui est essentiel pour la maintenance et l'expansion de votre projet dans le temps.

1. La communauté :

ReactJS possède une grande communauté des développeurs JS, bien que React Native qui est aussi offre la possibilité de publier des applications natives en écrivant du JS. Cela permet d'économiser des temps de développement considérable.

VI. Interfaces graphiques :

1. Login :



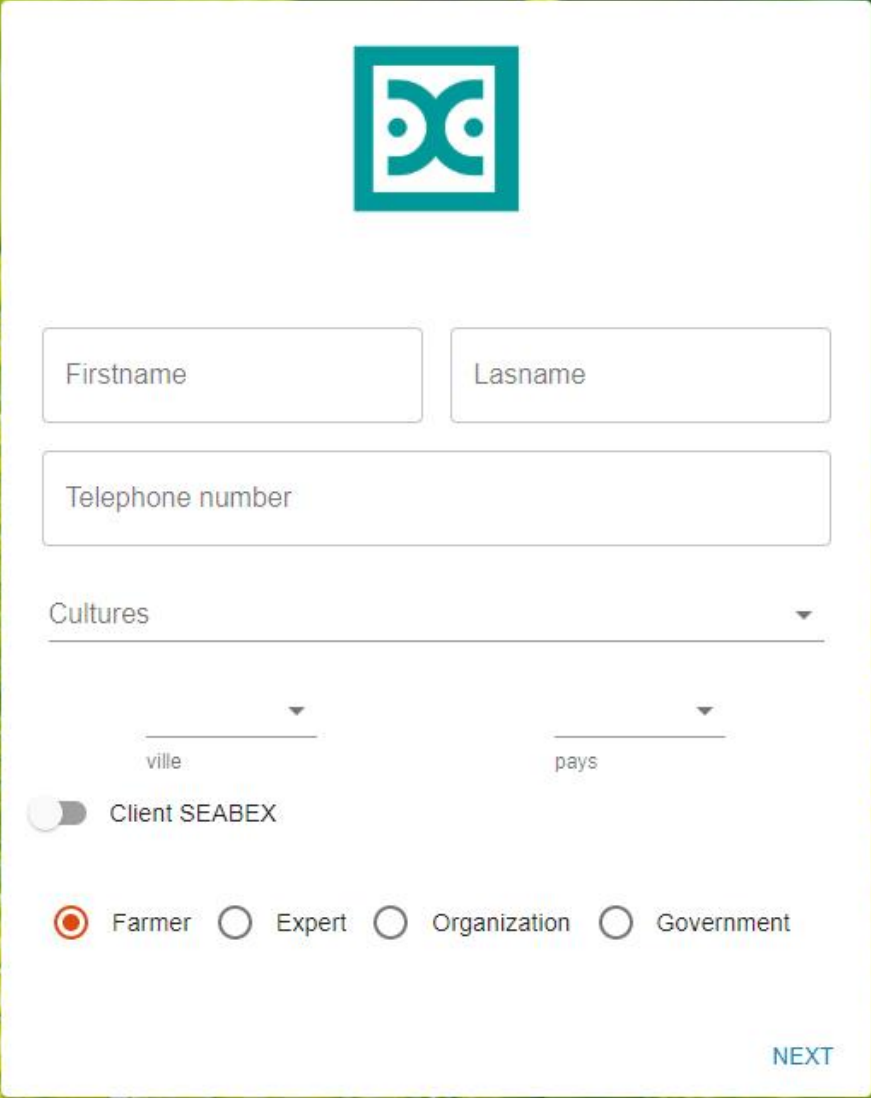
Figure 25: Interface Login

Cette interface permet à l'utilisateur de s'authentifier. L'utilisateur doit entrer son adresse e-mail ou numéro de téléphone et son mot de passe pour accéder à l'application.

2. Inscription :

Cette interface permet à l'utilisateur d'inscrire à notre site. C'est une interface commune pour tout type d'inscription (client SEABEX ou abonné service SMS⁶). Si l'utilisateur choisit d'être un client SEABEX alors il doit remplir d'autres champs.

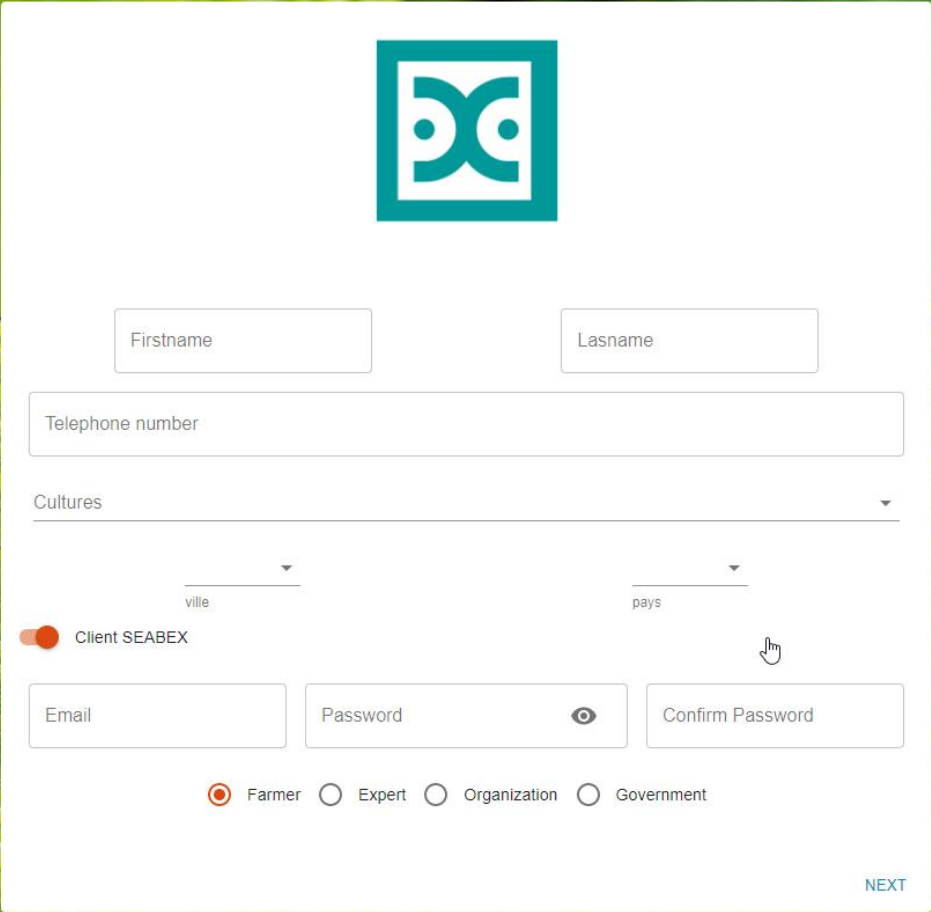
⁶ Short Message Service



The image shows a registration form interface for SEABEX, overlaid on a background of a green, textured surface with water droplets. The form is white and contains the following elements:

- A teal logo at the top center, consisting of a square with a stylized 'S' and 'B' inside.
- Two input fields for 'Firstname' and 'Lasname' (Lastname).
- A single input field for 'Telephone number'.
- A dropdown menu for 'Cultures'.
- Two dropdown menus for 'ville' (city) and 'pays' (country).
- A toggle switch for 'Client SEABEX'.
- Four radio buttons for user roles: 'Farmer' (selected), 'Expert', 'Organization', and 'Government'.
- A 'NEXT' button in the bottom right corner.

Figure 26: Interface Inscription 1



The image shows a registration form interface for SEABEX, overlaid on a background of green leaves with water droplets. The form is white and contains the following elements:

- A teal logo at the top center, consisting of a square with a stylized 'S' and 'B' inside.
- Input fields for 'Firstname' and 'Lasname' (Lastname).
- A single-line input field for 'Telephone number'.
- A dropdown menu for 'Cultures'.
- Two dropdown menus for 'ville' (city) and 'pays' (country).
- A toggle switch labeled 'Client SEABEX' which is currently turned on (red).
- Input fields for 'Email', 'Password' (with an eye icon for visibility), and 'Confirm Password'.
- Four radio buttons for user roles: 'Farmer' (selected), 'Expert', 'Organization', and 'Government'.
- A 'NEXT' button in the bottom right corner.

Figure 27: interface inscription 2

3. Gestion des sites :

Dans cette interface, l'utilisateur peut consulter tous ses sites sous formes des polygones et ses stations sous formes des pointeurs dans une carte. Comme il peut les consulter dans une liste.

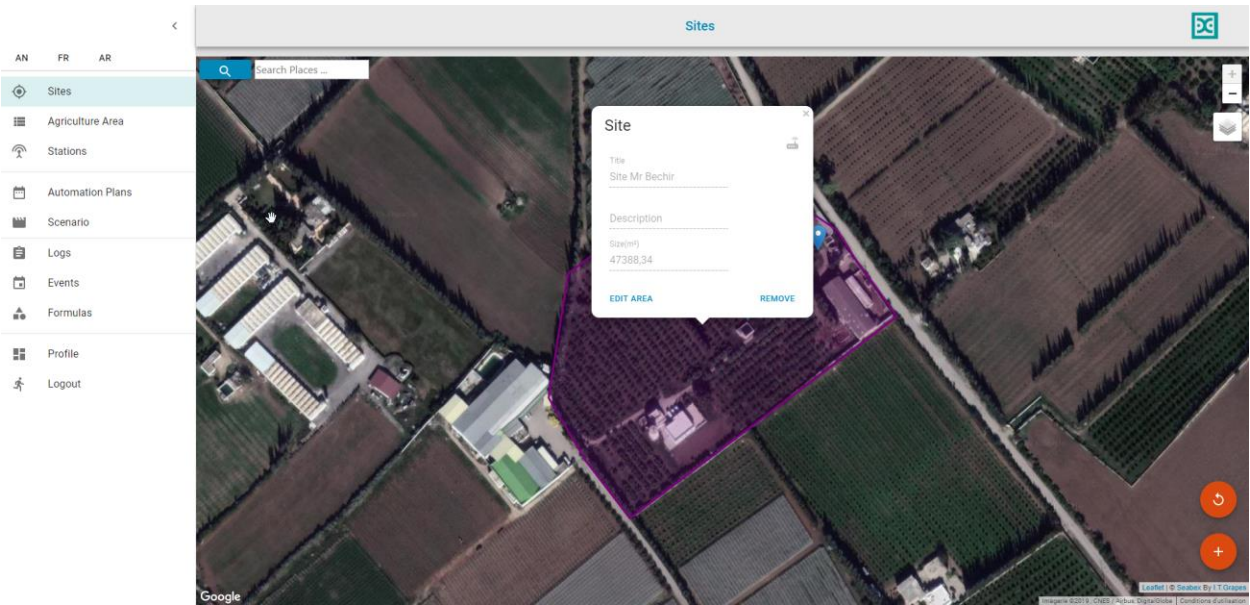


Figure 28: Interface carte des sites

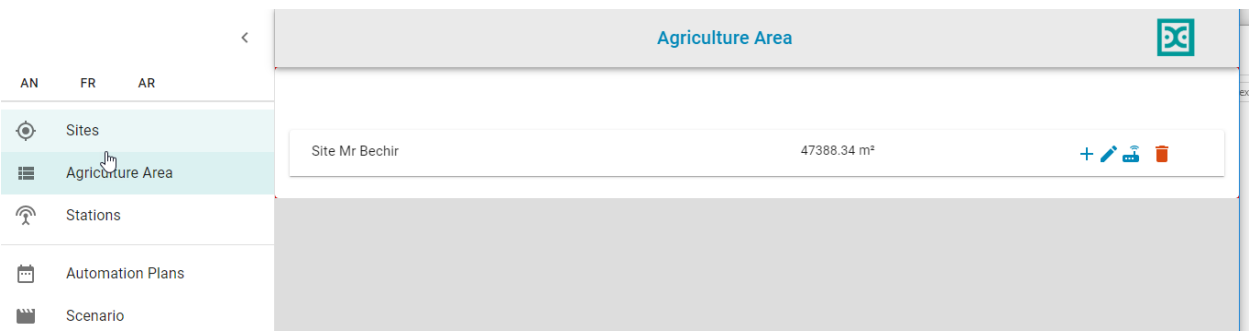


Figure 29: Interface liste des sites

4. Stations :

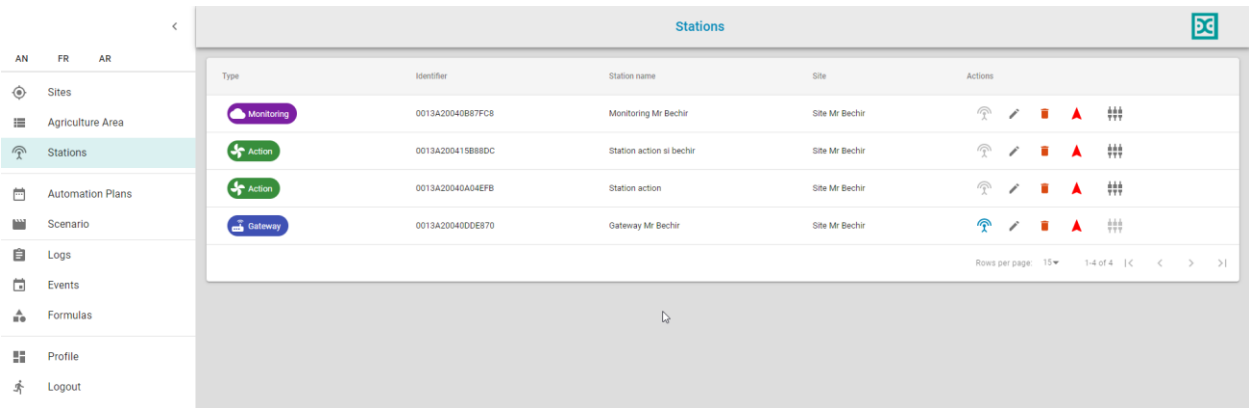


Figure 30: Interfaces liste des stations

Chapitre IV: Réalisation

Cette interface permet à l'utilisateur de consulter la liste de tous ses stations. Il peut aussi consulter la liste des équipements et la position de chaque station. Il peut aussi ajouter une nouvelle ou modifier une autre.

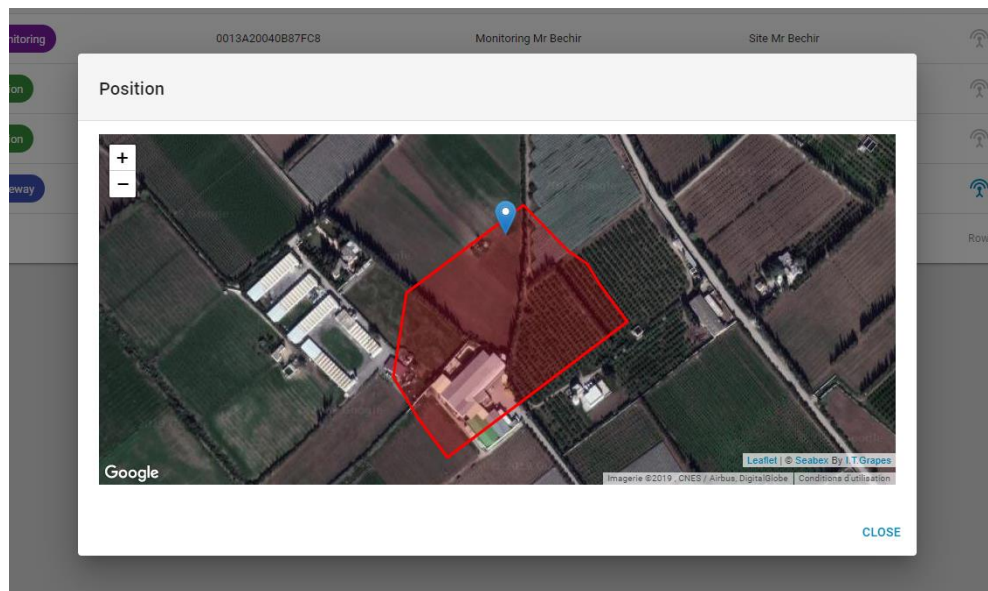


Figure 31: Interface position station

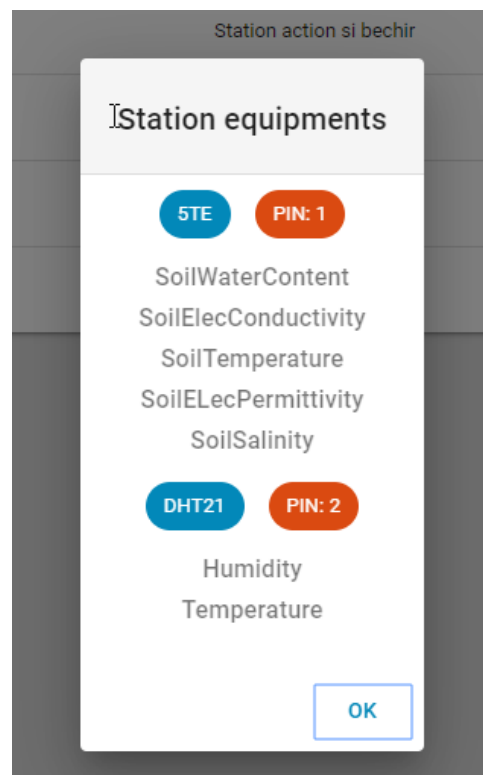


Figure 32: Interface équipements d'une station

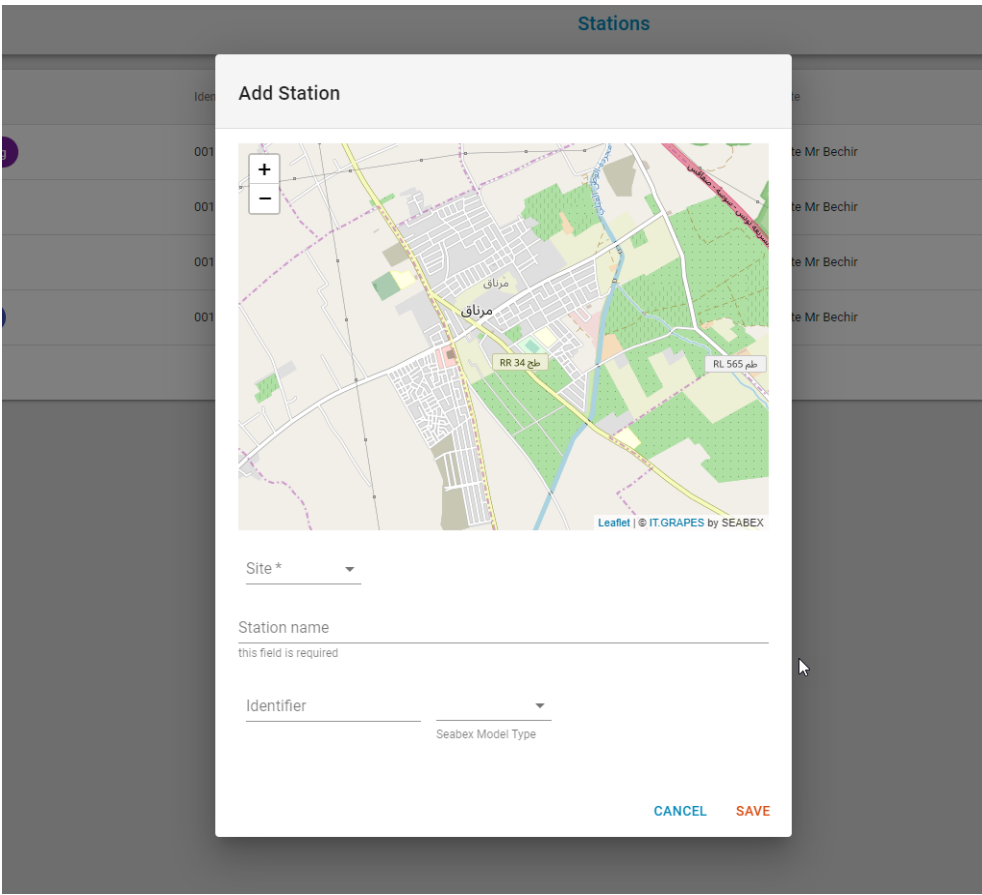


Figure 33: Interface ajout station

5. Plan d’irrigations :

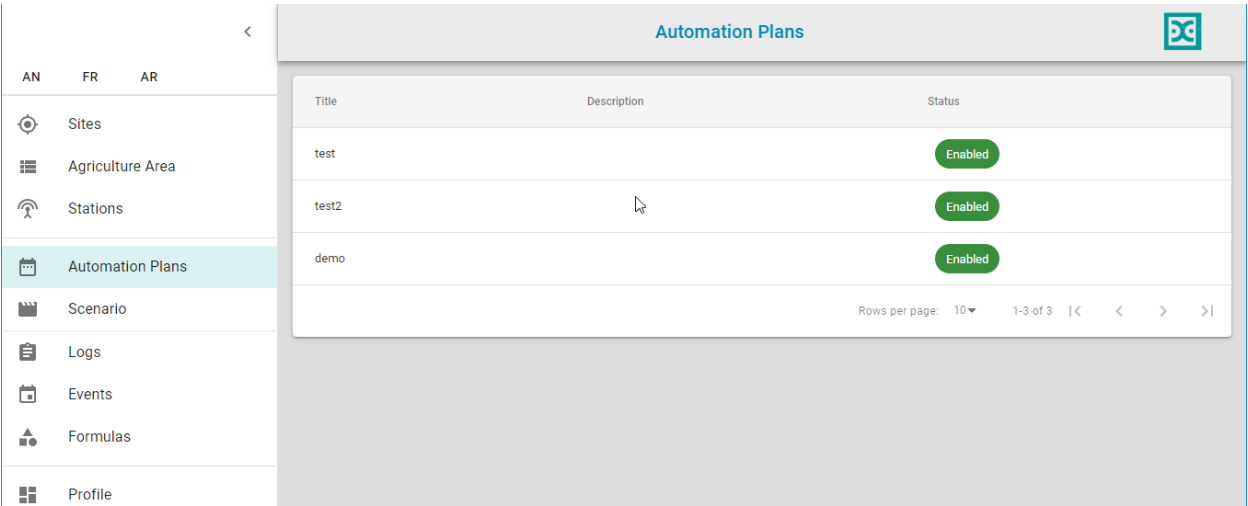


Figure 34: Interface liste des plans d'irrigations

Cette interface permet à l'utilisateur de consulter la liste de ses plans d'irrigations que ce soit il l'utilise ou pas. Chaque plan d'irrigation est composé de deux ou plusieurs tâches.

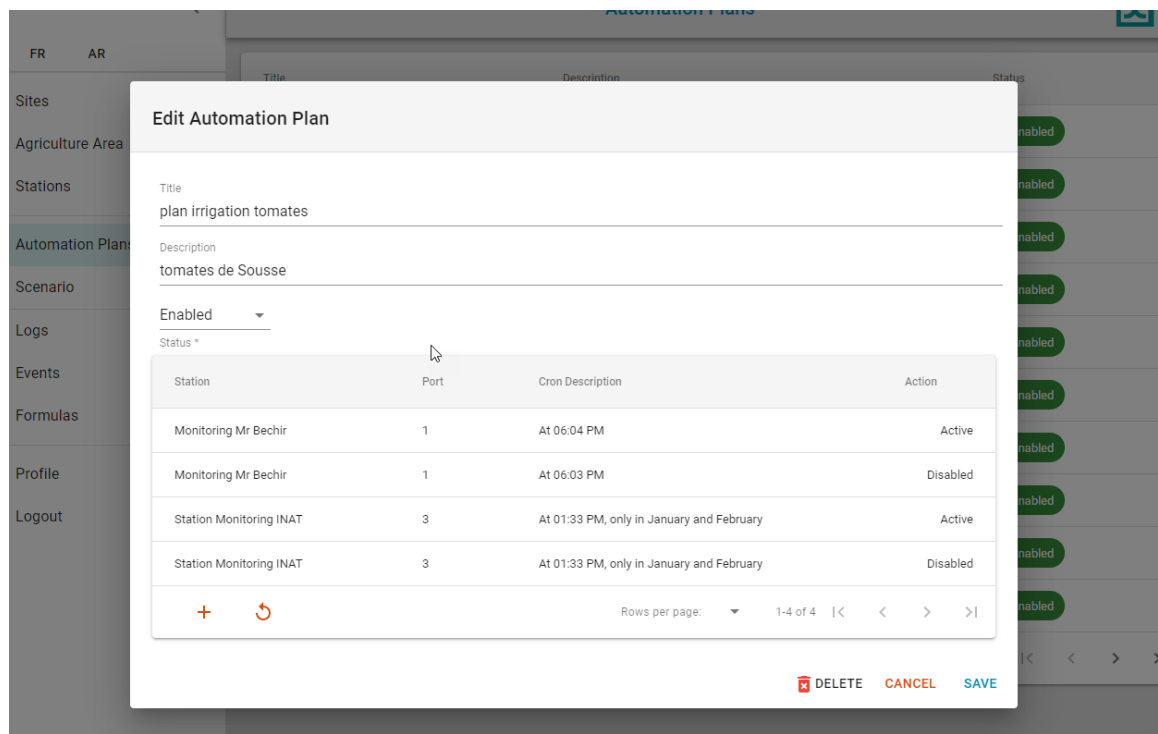


Figure 35: Interface détails d'un plan d'irrigation

Chaque utilisateur peut créer son propre plan d'irrigations avec ses tâches.



Figure 36: Interface ajout d'un plan d'irrigation

Add Irrigation Plan

Select Station *

Select Actuator *

Select Status

☐ Actuator

From

04:37 PM

To

04:37 PM

Selected Months

Every Month

Selected Month Day

Every Month Day

Selected Week Days

Every Day

CANCEL

CONFIRM

Figure 37: Interface ajout des tâches

6. Scenarios :

Scenario

AddScenario

Enabled

Status *

Title

Description

Add operator to equation

Equation

Conditions

Identifier	Stations	Sensor	Operator: <=>	Value
Rows per page: 0-0 of 0 < < > >				

Actions

Identifier	Stations	Action
Rows per page: 0-0 of 0 < < > >		

CANCEL

SAVE

DELETE

Figure 38: Interface ajout d'un scénario

Cette interface permet à l'utilisateur de créer un nouveau scénario. Un scenario est composé principalement des conditions et des actions.

7. Logs :

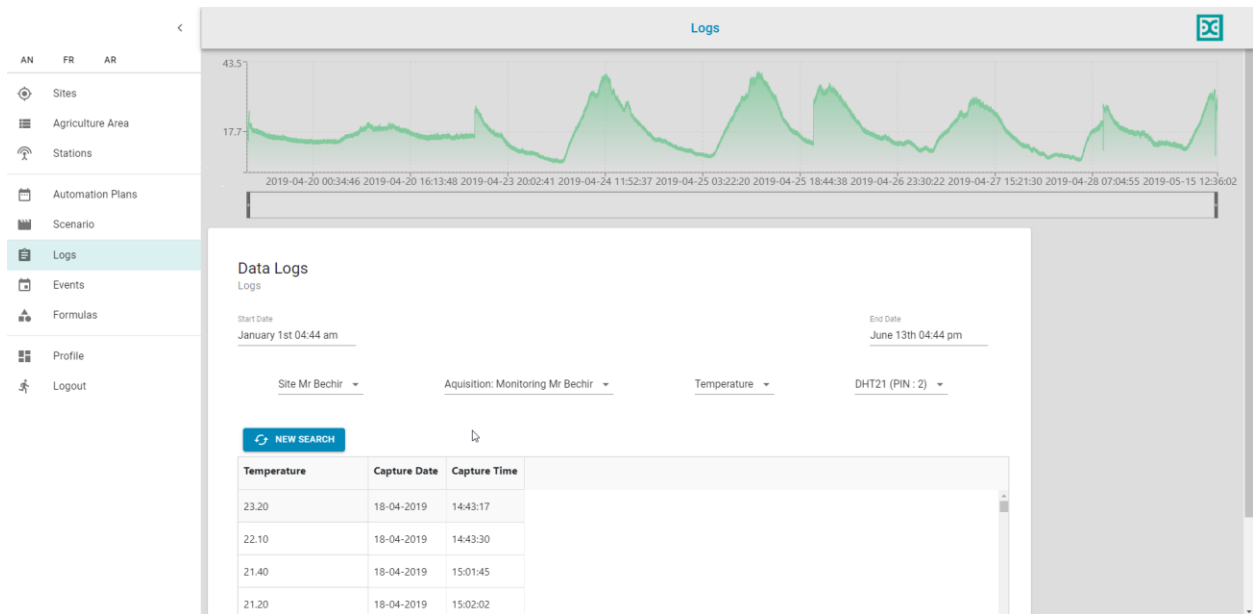


Figure 39: Interface consultation du log

Dans cette interface, l'utilisateur peut consulter l'historique des données temps réel qui ont été sauvegardé au moment de sa réception. Il peut aussi consulter un graphe qui montre l'évolution de type de données dans une période bien déterminé.

8. Evénements :

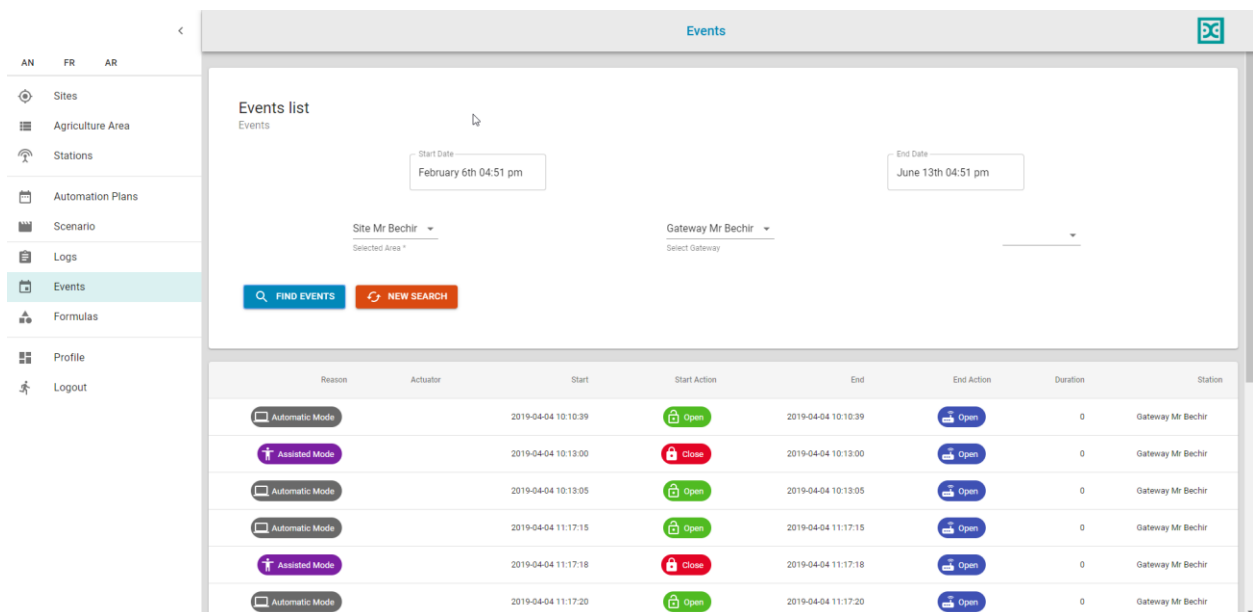


Figure 40: Interface consultation des événements

Cette interface offre aux utilisateurs la possibilité de consulter tous les événements exécutés (plans d'irrigations, scenarios, ouverture manuelle d'un port ...).

9. Formules :

The screenshot displays a web application interface titled 'Formulas'. On the left is a sidebar navigation menu with options: Sites, Agriculture Area, Stations, Automation Plans, Scenario, Logs, Events, Formulas (highlighted), Profile, and Logout. The main content area is titled 'new Formula' and contains a form with the following fields: 'Formula name' (containing 'Formula demo' with a note 'this field is required'), 'Description' (containing 'formule pour demo PFE'), 'STE' (a dropdown menu with 'Sensor' selected), and 'SoilSalinity' (a dropdown menu with 'Datatype' selected). Below these fields is a large text input containing the formula $U = 5 \times x + \sqrt{6 \times \beta} + \pi + \sin^{-1} 3 \times n$. Under the input are 'SAVE' and 'RESET' buttons. Below the buttons, it says 'This is your formula: $U = 5 \times x + \sqrt{6 \times \beta} + \pi + \sin^{-1} 3 \times n$ '. At the bottom of the form area are two buttons: 'MY FORMULAS' and 'ALL FORMULAS'. Below the form is a formula editor toolbar with various mathematical symbols and functions, including a numeric keypad, algebraic symbols, and mathematical operators.

Figure 41: Interface ajout et liste des formules

Cette interface offre à l'utilisateur la possibilité de consulter et créer ses formules pour les appliqués aux données temps réel.

VII. Conclusion :

Dans ce chapitre nous avons détaillé les technologies utilisées pour la réalisation de notre projet ainsi que l'environnement de travail. En ajoutant une figure de navigation qui montre la hiérarchie de l'application ainsi que les principales interfaces de l'application.

Conclusion générale

Notre projet a consisté en la conception, le développement d'une plateforme intelligente d'agriculture.

On a décomposé notre projet en deux phases, premièrement la phase de recherche dans laquelle nous avons saisi les différentes notions et technologies à utiliser dans le projet, les architectures..., deuxièmement, la phase de conception et développement durant laquelle on a préparé les spécifications de l'application après laquelle on a commencé à développer notre projet.

Nous sommes arrivés à développer toutes les fonctionnalités du système dans les temps. L'intégration a été réalisée avec succès, c'est-à-dire que l'application est maintenant installée sur les machines et prête à être utilisée.

Ce stage nous a permis d'approfondir nos connaissances théoriques, acquises tous le long de notre formation, par la pratique des nouvelles technologies. Cette expérience nous a permis de maîtriser le langage LARAVEL ainsi que d'autres technologies et Framework comme React JS, PostgreSQL, GraphQL....

Le stage au sein de la société a aussi été pour nous une occasion unique pour épanouir nos capacités de communication dans un environnement professionnel. C'est une expérience très enrichissante sur tous les domaines.

Enfin, l'application que nous avons développée pourrait être enrichie par des fonctionnalités avancées telle que la gestion des ressources financières.

Webographie

- [1] Agriculture 4.0 : <https://www.terre-net.fr/actualite-agricole/economie-social/article/definition-de-l-agriculture-4-0-202-138781.html>
- [2] Gantt: app.teamgantt.com (dernière visite le 5 juin 2019)
- [3] React js : <https://unscuzzy.com/react-js-cest-quoi/>
- [4] Redux: <https://redux.js.org/introduction/getting-started>
- [5] Laravel : <https://openclassrooms.com/fr/courses/1811341-decouvrez-le-framework-php-laravel-ancienne-version/1816846-presentation-generale>
- [6] GraphQL : <https://graphql.org/learn/>
- [7] REST API: <https://www.comprendre-internet.com/Qu-est-ce-qu-une-API-REST-ou-RESTful.html>
- [8] Eloquent ORM : <https://github.com/laravel-france/documentation/blob/master/4.2/eloquent.md>
- [9] MATERIAL UI: <http://material-ui.com>
- [10] PostgreSQL : <https://docs.postgresql.fr/11/preface.html#intro-what-is>
- [11] Why react js: <http://www.acseo.fr/6-raisons-daimer-reactjs/>
- [12] les Phases et itérations de la méthode RUP : era.nih.gov/docs/rup_fundamentals.htm (visité le 14 Mai 2019)
- [13] diagramme de classe : <http://www.uml-sysml.org/diagrammes-uml-et-sysml/diagramme-uml/diagramme-de-classe>

Résumé

L'objectif finale de ce travail de PFE consiste à développer une plateforme destinée aux acteurs intervenants dans le domaine d'agriculture utilisant des technologies innovantes et suivant une architecture solide afin de satisfaire les exigences de ces différents acteurs (agriculteurs, organisme d'agriculture, experts, chercheurs ...) dans le but d'avoir une bonne gestion des ressources d'eau.

Abstract

The ultimate goal of this project of the end of study is to develop a platform for actors involved in the field of agriculture using innovative technologies and following a solid architecture to meet the requirements of these different actors (farmers, agricultural organizations, experts, researchers ...) in order to have a good management of water resources.