

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université de la Manouba

Institut Supérieur des Arts Multimédia



MEMOIRE DE PROJET DE FIN D'ETUDES

Préparé en vue de l'obtention du diplôme d'Ingénieur en Informatique appliquée aux Multimédia

Code INLOG1

SUSTAINAPP



Réalisé par
Anas NEUMANN, Cycle Ingénieur INLOG

Encadré par
Ph.D. Jihene MALEK, ISAMM,
Ph.D. Adnène HAJJI, CIRRELT/Université Laval

Année Universitaire 2016-2017

Remerciements

Je tiens tout d'abord à remercier les membres du jury pour la lecture de ce rapport et l'attention qu'ils portent à mon travail.

Ensuite, je tiens à remercier les membres du Cirrelt/CeRCI, madame Monia Rekik, madame Jacqueline Corbett, monsieur Sehl Mellouli, monsieur Hirotoshi Takeda et surtout mon encadrant monsieur Adnene Hajji, pour m'avoir permis de vivre cette expérience et m'avoir encadré durant ces quatre mois. L'intégration dans ce nouvel environnement et ce nouveau pays aurait pu être une tâche bien difficile sans l'aide et le soutien de monsieur Hajji et Meriem Nefzi. Je tiens également à remercier monsieur Alexis Roy, du service d'administration des équipements du Cirrelt, pour le temps qu'il m'a accordé lors de l'hébergement de l'application.

Je tiens à remercier tous mes professeurs à l'ISAMM pour toutes ces années d'enseignement, cette longue aventure et particulièrement madame Jihene Malek, mon encadrante de toujours, pour toute l'aide qu'elle m'a apportée durant ce projet mais aussi dans ceux qui l'ont précédé.

Enfin, je tiens à remercier ma famille et amis, pour leur soutien sans faille et le sens qu'ils donnent à mon travail.

Dédicaces

Je dédicace ce travail à tous les membres du GDA Sidi Amor, ma famille et plus particulièrement à ma mère.

Table des matières

Introduction Générale	5
I. Présentation générale.....	6
1.1. Contexte du projet.....	6
2.2. Problématique	6
II. Organisme d'accueil.....	7
2.1. Le CIRRELT.....	7
2.2. L'Université Laval	8
2.3. Le CeRCI.....	10
III. Approche Méthodologique	10
3.1. Méthodologie de gestion de projet.....	10
3.2. Outils et environnements.....	13
IV. Structure du rapport.....	14
Etat de l'Art	15
I. Revue de la littérature	16
1.1. La Gamification	16
1.2. Les gestes écoresponsables au sein d'une ville intelligente.....	20
II. Applications similaires	22
2.1. Travaux de recherche	23
2.2. Applications et jeux sérieux	24
2.3. Gamification de la formation	26
III. Choix et orientations.....	28
3.1. Méthodologie d'évaluation de l'impact	28
3.2. Orientations en termes de gamification.....	29
3.3. Méthodologie de production d'informations	29
Sustainapp.....	30
I. Analyse des besoins	31
1.1. Application de la méthodologie de conception gamifiée	31
1.2. Spécification des besoins fonctionnels.....	32
1.3. Spécification de la qualité	38
II. Conception technique et architecturale	41
2.1. Précisions sur les fonctionnalités.....	41
2.2. Modèle de données.....	43
2.3. Architecture logicielle et technologies	45
III. Conception graphique	49

3.1. Couleurs	49
3.2. Polices.....	50
3.3. Logos et icônes	50
3.4. Organisation des pages et navigation.....	51
Réalisation du projet	54
I. Mise en place du projet	55
1.1. Planification générale de la réalisation.....	55
1.2. Planification de l'itération initiale	57
1.3. Mise en place du projet.....	57
1.4. Sustainapp Server	59
1.5. Sustainapp Mobile	66
1.6. Sustainapp Batch	70
1.7. Réalisation des premières fonctionnalités	72
II. Un réseau d'écoresponsables	77
2.1. Planification de l'itération	77
2.2. Réalisation des fonctionnalités	77
III. Les challenges.....	80
3.1. Planification de l'itération	80
3.2. Réalisation des fonctionnalités	80
IV. Les cours en écologie	82
4.1. Planification de l'itération	82
4.2. Réalisation des fonctionnalités liées aux cours.....	83
4.3. Réalisation des fonctionnalités liées aux quizz	86
V. Les actualités du réseau	88
5.1. Planification de l'itération	88
4.2. Réalisation des fonctionnalités	88
VI. L'intégration dans les villes intelligentes	93
6.1. Planification de L'itération	93
6.2. Réalisation des fonctionnalités	94
VII. Administration et finalisation du projet.....	96
7.1. Planification de l'itération	96
7.2. Réalisation des fonctionnalités	97
Tests et Déploiement.....	99
I. Tests et validation	100
1.1. Création de tests pour Sustainapp Server.....	100
1.2. Création de tests pour Sustainapp Mobile.....	103
1.3. Validation fonctionnelle et gamification	105

II.	Déploiement du projet	106
2.1.	Choix et responsabilité	106
2.2.	Installation et déploiement de l'application	107
2.3.	Outils de maintenance	110
III.	Communication.....	114
3.1.	Utilisation des plateformes sociales	114
3.2.	Communication directe.....	115
	Conclusion.....	116
I.	Bilan du projet.....	117
1.1.	Bilan scientifique.....	117
1.2.	Bilan technologique.....	117
1.3.	Bilan social.....	118
II.	Perspectives	118
2.1.	Perspectives du projet.....	118
2.2.	Perspectives personnelles.....	118
	Bibliographie.....	120
	Table des figures.....	121
	Table des tableaux	124



Introduction Générale

I. Présentation générale

1.1. Contexte du projet

Le cycle ingénieur est un cursus qui se démarque par la double aptitude qu'il confère à ses diplômés. En effet un ingénieur est d'une part un professionnel se devant de maîtriser l'aspect technique de ce qu'il réalise et dans le cas d'un ingénieur diplômé de l'Institut Supérieur des Arts Multimédia de Manouba (*ISAMM*), cet aspect concerne la réalisation d'applications multimédia telles que des plateformes web, applications mobiles, jeux interactifs ou encore de réalité virtuelle. D'autre part, l'ingénieur a également pour vocation d'être à terme un concepteur et gestionnaire de projet, formé donc aux différentes formes de gestion que cela implique (financière, humaine, et de la communication) et capable d'imaginer des solutions à de nouveaux problèmes¹.

Ce projet de fin d'études vient donc concrétiser cette formation en amenant l'étudiant à utiliser toutes les ressources qu'ils lui ont été transmises durant le cycle afin de concevoir, gérer et réaliser un projet concret. Ce projet est également une expérience d'insertion pour l'étudiant qui le réalise au sein d'une entreprise ou d'un laboratoire de recherche. Ce projet a ainsi été réalisé au sein du laboratoire *CIRRELT*² (Université Laval) à Québec durant une période de 4 mois de mars à juillet 2017 et dont l'aspect technique a porté sur la réalisation d'une plateforme web et mobile appelée « *Sustainapp* ³».

2.2. Problématique

Le sujet du projet tel que défini initialement par le *CIRRELT* consiste au développement et à l'évaluation d'une application utilisant les principes de la « *Gamification* ». Avec cette application, on peut cibler des comportements plus écoresponsables au sein d'une ville intelligente afin de promouvoir le développement durable.

Ce projet comporte deux parties : une partie de recherche et une autre purement appliquée. La première partie devra donc porter sur une recherche des principes de la « *Gamification* » ainsi que sur une méthodologie de conception applicable au domaine du développement durable et des gestes écoresponsables. La deuxième quant à elle est purement informatique et consiste donc au développement et au déploiement d'une application qui appliquera les modèles et principes étudiés dans la première partie.

On peut également noter qu'en plus de s'intéresser à la théorie socio-informatique « *Gamification* », ce projet s'intègre dans deux domaines très importants de nos jours : l'écologie et les villes intelligentes. Les gestes écoresponsables cités dans le sujet font référence à l'un des objectifs de

¹ Programme officiel du cycle ingénieur de l'*ISAMM* : <http://www.isa2m.rnu.tn/CycleIng.php>

² *CIRRELT* : Centre Interuniversitaire de Recherche sur les Réseaux d'Entreprise, la Logistique et le Transport

³ *Sustainapp* : combine « Sustainable » (Durable) et « App » (Application).

développement durable tels que décrit par les nations unis⁴ qui concerne la protection de la planète. Les Nations Unies ont par ailleurs émis une liste des gestes écoresponsables à effectuer au quotidien sur leur site web officiel⁵.

En outre, le fait de spécialiser les gestes écoresponsables aux villes intelligentes n'est pas anodin. Les villes intelligentes sont définies par l'Université Laval comme étant les villes dont la gouvernance choisit de mettre en place une infrastructure technologique et informatique permettant d'améliorer la vie du citoyen⁶. Elles ont un rôle primordial dans la réalisation des objectifs définis par les Nations Unies. La gouvernance de ces villes est constituée des décideurs qui peuvent choisir ou non de mettre en place les infrastructures permettant aux citoyens d'agir de manière écologiquement responsable. Ainsi, il est nécessaire que ce projet porté sur l'écologie intègre la notion des villes. Les premières d'entre elles avec qui le CIRRELT souhaite enclencher un partenariat autour du projet seraient Québec et Montréal.

II. Organisme d'accueil

Le projet Sustainapp a été réalisé au sein du CIRRELT, un centre interuniversitaire présent dans de nombreuses universités canadiennes et instituts tels que l'Université Laval. Plus particulièrement ce stage s'est déroulé à La Faculté des Sciences de l'Administration (FSA) de l'Université Laval, au sein d'une équipe de chercheur du CIRRELT qui se nomme le CeRCI (Centre de Recherche sur les Communautés Intelligentes).

2.1. Le CIRRELT

Le Centre Interuniversitaire de Recherche sur les Réseaux d'Entreprise, la Logistique et le Transport (CIRRELT) se décrit lui-même sur son site officiel comme « un laboratoire né en mai 2006 de la fusion du Centre de recherche sur les transports (CRT) de l'Université de Montréal (UdeM), l'École Polytechnique et HEC Montréal, du Centre de Recherche sur les Technologies de l'Organisation Réseau (CENTOR) de l'Université Laval et du groupe Polygistique de l'École Polytechnique, auxquels se sont joints les chercheurs de l'UQAM regroupés autour de la Chaire de recherche industrielle du CRSNG (Conseil de Recherches en Sciences Naturelles et en Génie) en management logistique»⁷.

⁴ Objectifs de Développement Durables : <http://www.un.org/sustainabledevelopment/sustainable-development-goals/>

⁵ Liste des gestes écoresponsables : <http://www.un.org/sustainabledevelopment/takeaction/>

⁶ Villes Intelligentes : <http://www4.fsa.ulaval.ca/la-recherche/centres-groupes-et-laboratoires/centre-de-recherche-sur-les-communautes-intelligentes-cerci/mission/>

⁷ Site web du CIRRELT : <https://www.cirrelt.ca/>



Figure 1 : Logo du CIRRELT

Ainsi le CIRRELT est aujourd'hui un laboratoire présent dans plusieurs écoles (Université Laval, McGill, Concordia, ETS, UQAM, HEC Montréal, l'Ecole Polytechnique de Montréal et l'Université de Montréal) et est principalement financé par le « *Fond de Recherche, Nature et Technologies* » ainsi que le « *Fond de Recherche Société et Culture* » du Québec.



Figure 2: Instituts membres du CIRRELT

Les cinq principaux axes de recherche du CIRRELT sont⁸ :

- **Axe 1 : Agrégation de métadonnées** : connaissance des bases de données, anticipation des besoins d'information, construction d'approches intégrées
- **Axe 2 : Innovations théoriques** : transition énergétique, inter-modalité en transport de marchandises, résilience des réseaux, internet physique
- **Axe 3 : Développements méthodologiques** : méthodes d'optimisation et de simulation, méthodes d'analyse de la résilience, méthodes de recherche comparative
- **Axe 4 : Solution de problèmes** : problèmes économiques, problèmes environnementaux, problèmes sociaux
- **Axe 5 : Indicateurs de performance** : fluidité des trafics, compétitivité des chaînes logistiques, mesures d'adaptation aux changements environnementaux

Notre sujet est donc principalement lié au quatrième axe, la mise en place de solution aux problèmes environnementaux et sociaux.

2.2. L'Université Laval

L'Université Laval (UL)⁹ est une université canadienne et plus particulièrement québécoise fondée en 1852 à l'initiative du Séminaire de Québec. L'UL est le plus ancien établissement d'enseignement supérieur francophone en Amérique et l'un des plus grands de par ses 48 000 étudiants et 422 programmes d'études.

⁸ Axes de recherches du CIRRELT : https://www.cirrelt.ca/?Page=AXES_RECHERCHE_NOUVEAUX

⁹ Site web officiel de l'Université Laval : <https://www.ulaval.ca/>



Figure 3 : Logo actuel de l'Université Laval

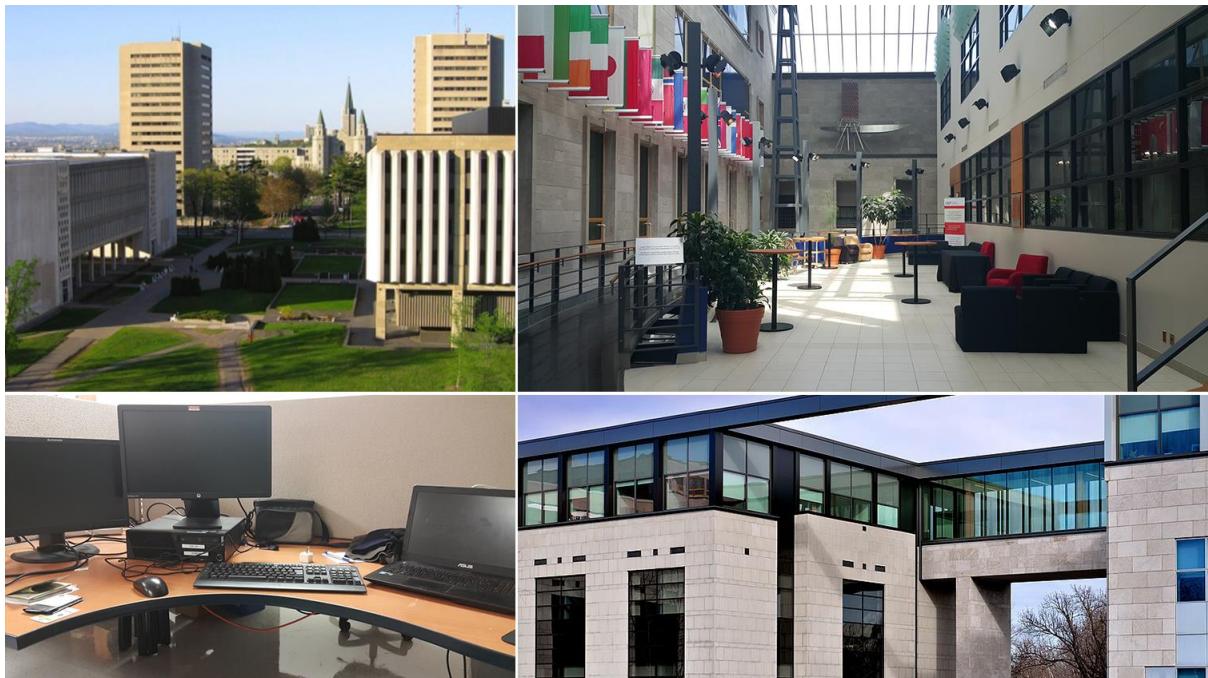


Figure 4 : l'Université Laval et locaux du CIRRELT

Le fait que ce projet a été réalisé au sein des locaux CIRRELT de l'Université Laval, dans la Faculté des Sciences de L'Administration (FSA) a une importance particulière. En effet *Sustainapp* a été pensé de manière à lier à la fois les axes de recherche du laboratoire aux champs d'action décidés par l'Université Laval en vue de créer un véritable partenariat autour de ce projet en matière notamment d'administration de la plateforme et de promotion du projet chez les étudiants.

L'Université Laval met en avant les objectifs et actions qu'elle compte entreprendre sur son site officiel. Ces objectifs sont chiffrés, décrits et planifiés dans des documents PDF mis en ligne et on retrouve sur le site l'état actuel d'avancement. Le développement durable est l'un des domaines qu'elle met le plus en avant en y consacrant dix axes d'interventions¹⁰ :

¹⁰ Axes d'interventions relatifs au développement durable : <https://www.ulaval.ca/developpement-durable.html>

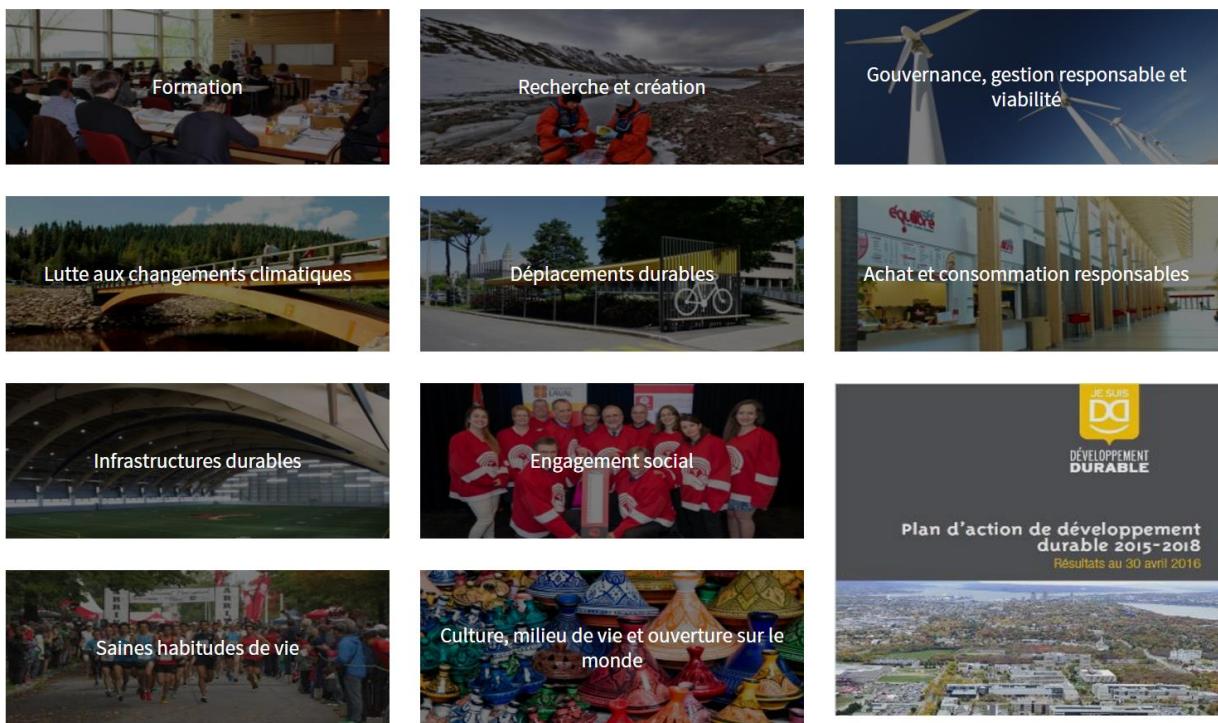


Figure 5: Axes d'interventions relatifs au développement durable

L'équipe du projet a décidé d'intégrer les fonctionnalités de *Sustainapp* principalement dans l'axe « Formation » en termes d'écologie et « Engagement Social » dans le développement durable.

2.3. Le CeRCI

Pour être encore plus précis, le projet est une initiative de l'équipe du CeRCI dont les membres, enseignants chercheurs à l'Université Laval et au CIRRELT ont été mes encadrants durant ce stage¹¹.

Le CeRCI, comme son nom le laisse entendre, oriente ses recherches autour du concept de ville intelligente et a donc souhaité finalement que le projet, en plus de respecter les axes d'intervention de l'université et les axes de recherche du CIRRELT, soit également orienté dans ce domaine.

III. Approche Méthodologique

3.1. Méthodologie de gestion de projet

De par la nature du projet (fonctionnalités non définies à l'avance, technologie nouvelle pour le développeur, forte probabilité de retour en arrière), il a paru évident qu'il fallait utiliser pour la gestion de ce projet une méthodologie de gestion « Agile » avec un cycle de vie incrémental et itératif. Cependant l'application parfaite de la méthodologie SCRUM aurait été

¹¹ Membres du CeRCI : <http://www4.fsa.ulaval.ca/la-recherche/centres-groupes-et-laboratoires/centre-de-recherche-sur-les-communautes-intelligentes-cerci/membres/>

impossible car le fait d'être un unique développeur fait perdre le sens à certains rôles (*Scrum Master*) et certaines pratiques telles que la réunion quotidienne de l'équipe (« *Daily Scrum* »).

Nous avons donc opté pour une méthodologie reprenant les avantages de *SCRUM* et supprimant uniquement les éléments incompatibles avec la nature du projet, et avons obtenu le résultat suivant :

- **La Réunion Mixte** : une réunion en début de sprint, planifiée sur l'agenda Google, à laquelle assistent le développeur ainsi que les membres dirigeants du laboratoire (encadrant et clients). Cette réunion permet à la fois de présenter la release du Sprint précédent (« *Sprint Review* ») et de planifier les fonctionnalités ou « *user stories* » à développer durant le prochain Sprint (« *Planification du Sprint* »). Chaque Sprint aura une durée de 2 semaines (contrainte CeRCI).

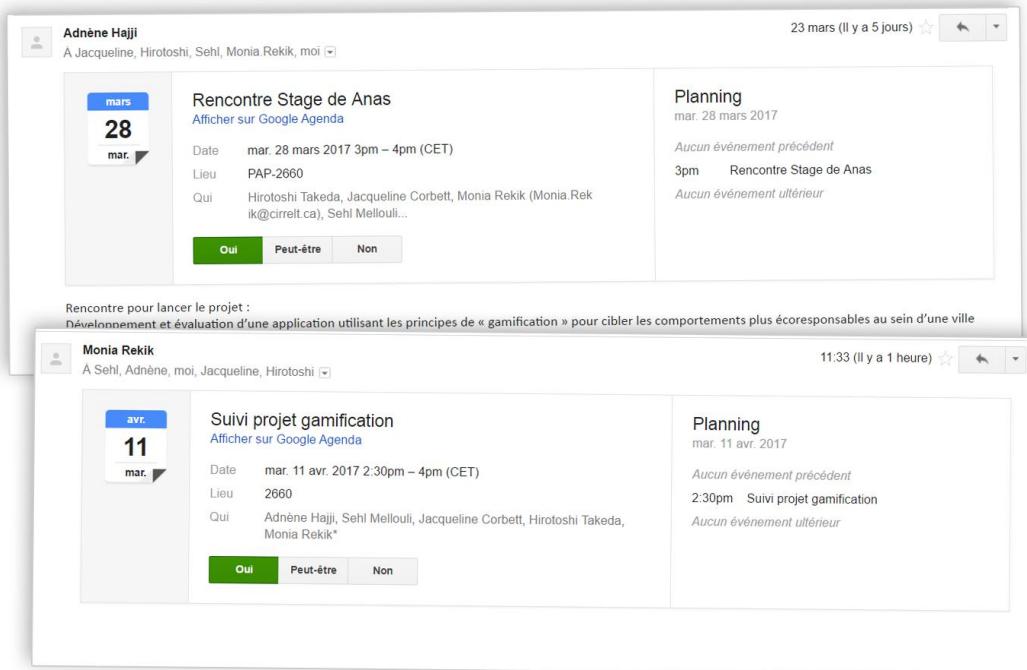


Figure 6: Réunions Mixtes

- **Le tableau de tâches** : Crée à la suite de chaque réunion mixte et maintenu à jour tout au long du Sprint, le tableau a deux principales utilités : permettre au développeur de gérer son temps et son avancement mais aussi faire connaître l'avancement en cours aux encadrants et principalement à l'encadrant ISAMM ne pouvant donc pas assister aux réunions mixtes. Pour cela nous avons choisi d'utiliser l'outil de gestion de projet de la plateforme Uprod'it¹² que j'ai développé moi-même dans le cadre d'un projet associatif. L'outil Uprod'it permet notamment d'avoir pour chaque Sprint : un tableau de tâches, un fil d'actualité et un calendrier de rappel des réunions.

¹² Plateforme web Uprod'it : www.uprodit.com

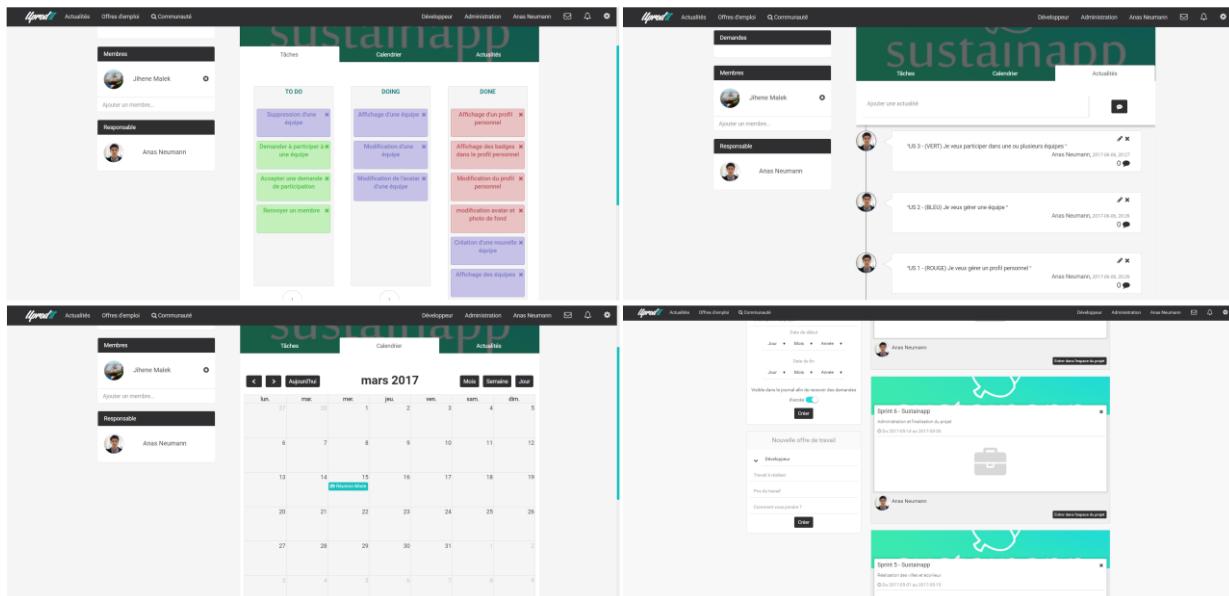


Figure 7: Module de gestion de projet de Uprod'it

- Les rôles :** Les membres du CIRRELT/CeRCI sont à la fois les clients du projet désirants obtenir l'application mais aussi des encadrants conseillant sur les méthodologies de recherche à utiliser. Tandis que j'ai joué le rôle de concepteur/développeur/testeur ainsi que de « *backlog owner* » (responsable du backlog produit).
- Les releases :** Chaque sprint a produit un prototype partiel appelé « *release* » qui peut être testé par les clients durant les réunions. Afin de garantir que le développement n'introduit pas de régressions sur la release précédente, nous avons travaillé avec le système de versions/branches *GIT*. Ce système présente par ailleurs plusieurs avantages : pouvoir revenir en arrière si la version actuelle présente des problèmes, faire de l'intégration continue des versions et travailler à plusieurs développeurs sur le même code de manière synchronisé (ce qui sera utile si à l'avenir le projet prend de l'ampleur et qu'il serait avantageux d'agrandir l'équipe).

En plus de mettre en place un processus de gestion « industriel » du projet, nous avons également souhaité garantir un certain niveau de qualité du produit (interne et externe). Pour cela, nous avons établi des règles de travail à chaque étape (conception, développement, réalisation graphique, tests, déploiement) qui permettent de satisfaire les six facteurs de qualité cités dans la norme ISO/IEC 9126. Ces règles seront détaillées dans les spécifications non fonctionnelles.

3.2. Outils et environnements

Gestion de
projet :



L'outil Uprod'it permet de concrétiser les tâches de planification et de suivi du projet



Gitlab est un outil qui s'installe sur le serveur de développement et qui fournit les technologies nécessaires à la création de version (revenir arrière, comparer, créer des branches ...) ainsi que la synchronisation du code entre différents développeurs et même le client s'il souhaite avoir le code.

Conception :



Pour la conception UML, nous avons utilisé StarUML, dont l'interface s'est révélée très agréable et complète.

Réalisation :



Pour le développement nous avons utilisé l'environnement Eclipse qui a l'avantage de comprendre les branches GIT et de gérer de nombreux langages.



Pour la réalisation graphique nous avons utilisé Adobe Photoshop, sans doute l'un des plus utilisés dans le domaine.

Déploiement
et Tests :



L'outil Jenkins s'installe sur le serveur d'exploitation et permet de déployer des plateformes web en exécutant des scripts. Jenkins permet également de lancer et suivre le résultat des tests unitaires en code.

Recherche d'informations :

Pour la recherche d'articles et travaux scientifique nous avons utilisé l'outil de l'Université Laval ainsi que le très célèbre « *Google Scholar* ».



Figure 8: Interface de l'outil de recherche d'articles

IV. Structure du rapport

Présenter les différents aspects d'un projet ayant à la fois une composante scientifique, technique, technologique et de gestion peut se révéler assez délicat. Dans l'idée de tout présenter en garantissant une organisation logique et une facilité de suivi, nous avons décidé de raconter les événements dans l'ordre chronologique.

La première partie présente l'aspect scientifique et l'approfondissement des notions théoriques et de leurs applications à travers un état de l'art ciblé. Après cela, en se basant sur les notions et expériences étudiées, la 2^e partie énonce nos choix conceptuels pour le projet. Ensuite, la réalisation technique du projet sera présentée dans l'ordre des sprints effectués en finissant par son hébergement sur une plateforme d'exploitation.

Parallèlement à cela, nous ferons référence à notre méthodologie de gestion et plan d'assurance qualité à chaque étape (conceptuelle, réalisation technique et hébergement).



Etat de l'Art

I. Revue de la littérature

Ce projet à la particularité, étant orienté recherche, de ne pas posséder de cahier des charges initial avec une liste de fonctionnalités que le client voudrait voir créer. Il contient plutôt une théorie, la « *gamification* », que le laboratoire aimerait tester sur le domaine du développement durable afin de promouvoir ce dernier. De plus, cette application devrait faire intervenir le concept de ville intelligente et s'intégrer dans les axes d'intervention de l'Université Laval.

Ainsi avant de concevoir l'application, il a été obligatoire pour nous de réaliser deux études :

- Une première étude sur les travaux scientifiques, thèses et articles publiés en relation avec la gamification, les gestes écoresponsables et les villes intelligentes.
- Une étude des produits existants (application mobile, web ou jeu) dont pourrait inspirer notre travail.

1.1. La Gamification

La première notion que nous avions à étudier est la gamification, l'intérêt de l'utiliser, ses apports possibles dans le domaine de l'écologie ainsi que les possibles méthodologies permettant de la mettre en place.

1.1.1. Définition et avantages

Bien que la définition puisse varier très légèrement d'un article à l'autre, la gamification, aussi appelée « *ludification* », correspond en fait à l'application des mécanismes issus des jeux (et des réseaux sociaux) dans un contexte dit sérieux afin d'accroître la motivation de l'utilisateur [Aparicio, 2012] [Xu, 2011] [Alaa, 2014].

Le jeu déclenche donc un plaisir à l'utilisation qui pourrait amener l'utilisateur à se fidéliser, à souhaiter accroître sa fréquence d'utilisation ou encore à éprouver de l'intérêt pour le sujet.

Cependant, il existe de nombreuses dérivées de la gamification selon le type du produit ou de l'activité créée comme le montre la figure 12 [Xu, 2011]. Par exemple le « *serious game* » (ou jeu sérieux) bien que basé sur le même esprit que la gamification se différencie par le fait que le *serious game* est un jeu à part entière possédant un *gameplay*¹³ et souhaitant faire passer un message dit sérieux alors qu'un produit dit « *gamifié* » ou « *ludifié* » n'est pas un jeu, l'objectif sérieux n'est pas sous-entendu mis au contraire c'est le mécanisme du jeu qui doit être subtil [Xu, 2011].

¹³ **gameplay** : les éléments graphiques, sensoriels, et ludiques (histoire, décors, musique) permettant de créer l'expérience d'immersion ressentie par le joueur.

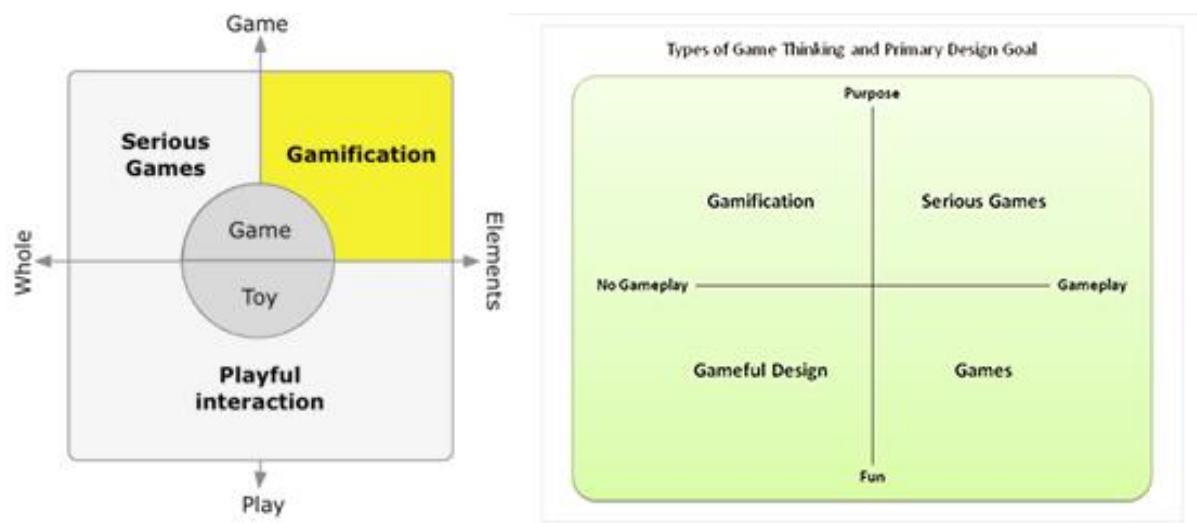


Figure 9: Dérivées de la gamification

1.1.2. Raisons de l'impact de la gamification

Il est alors important de savoir quels sont les éléments du jeu qui permettent de créer cet intérêt et surtout pourquoi le déclenchent-ils. La gamification est basée sur le fait que l'homme a différents besoins (5 types selon la théorie de Maslow) et désirs qui peuvent provoquer une volonté d'agir.

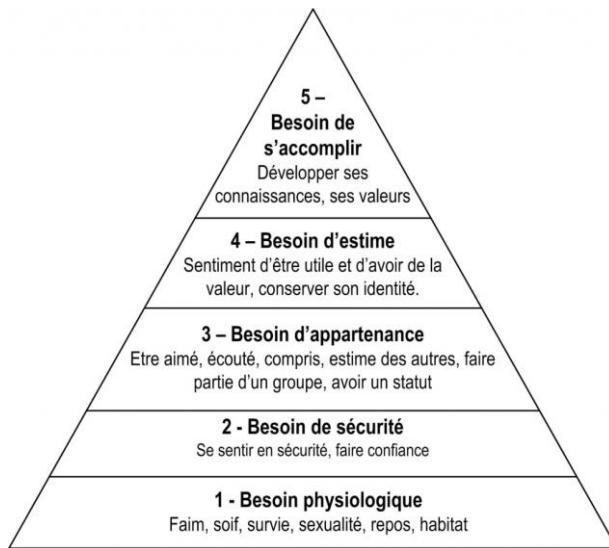


Figure 10: La pyramide de Maslow

On retrouve dans la littérature particulièrement trois besoins qui sont comblés par les mécanismes du jeu : un besoin d'autonomie, de compétence et de relation [Aparicio, 2012] [Xu, 2011]. Ces besoins étant du niveau 3 et de la pyramide de Maslow, on peut donc déduire que la gamification ne présente un intérêt que si l'utilisateur a déjà satisfait les deux premiers niveaux.

Parallèlement aux besoins et désirs, l'homme est également un être sensoriel et ses sens (ouïe, vue, odorat) ont également un impact sur sa prise de décision. Or le jeu vidéo principalement est une expérience sensorielle car des éléments graphiques et parfois sonores y sont intégrés.

Enfin, une autre classification dans laquelle deux familles de facteurs de motivation sont citées :

- facteurs personnels : découverte, rêve, immersion, challenge, objectifs, fil conducteur;
- facteurs interpersonnels : reconnaissance, collaboration, confiance, compétition ;

1.1.3. Mécanismes de la gamification

Dans la littérature on retrouve généralement les six mécanismes suivants associés à six dynamiques (besoins, désirs) comme étant les principaux moyens à intégrer dans le processus sérieux pour le « *gamifier* » [Xu, 2011]:

6 mécaniques	6 dynamiques
Les points	La gratification
Les niveaux	Le statut
Les challenges	La réalisation
Les badges	La créativité
Les classements	La compétition
Le don	L'altruisme

Figure 11: Les 6 mécanismes de la gamification

Cependant il existe des règles à respecter selon l'utilisateur ciblé. On retrouve dans différents travaux l'idée qu'une bonne conception de la gamification doit permettre d'obtenir ce que l'on appelle « *l'expérience ultime* » dans la théorie du « *flow* ». C'est-à-dire ne rendre le processus ni trop ennuyeux ni effrayant (peur de perdre, de ne pas être à la hauteur, d'être rejeté...) [Xu, 2011].

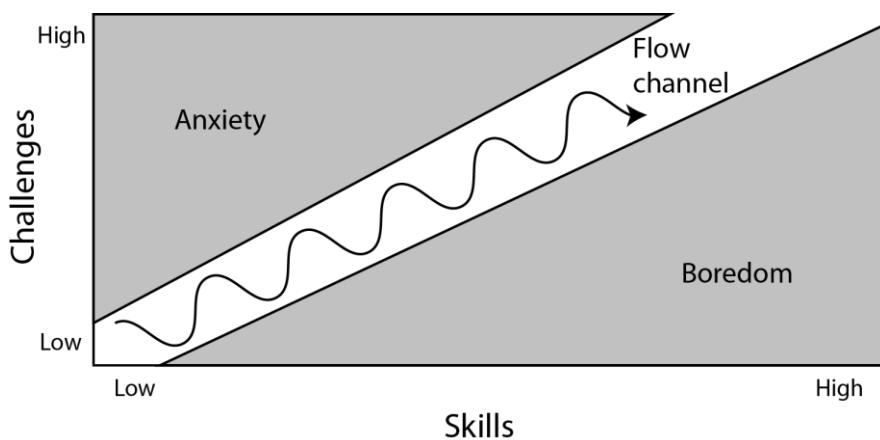


Figure 12: Théorie du Flow

Enfin, on retrouve également dans la littérature liée à la gamification des références à la théorie des « *types de joueurs de Bartle 1996* ». Théorie d'après laquelle il faut s'adapter aux différentes réactions possibles face au jeu : l'envie plutôt de dominer, de rencontrer des joueurs, de rêver ou de relever des défis [Xu, 2011] [Nassiri, 2017].



Figure 13 : Les types de joueurs de Bartle

1.1.4. Mise en place de la gamification : Méthode de gamification de fonctionnalités

La méthodologie de mise en place d'un SI¹⁴ gamifié par gamification des fonctionnalités est la même dans de nombreux travaux [Aparicio, 2012] [Xu, 2011] [Alaa, 2014] :

1. Identifier les objectifs de l'application (sans se préoccuper de la gamification) ;
2. Identifier les manières d'atteindre ces objectifs en termes de fonctionnalités (on retrouve ici le principe général du génie logiciel et des cas d'utilisation) ;
3. Pour chaque fonctionnalité : identifier le principe et le mécanisme qui pourrait motiver son utilisation sans affecter l'objectif initial de cette fonctionnalité ;
4. Analyse de la qualité de la gamification (souvent effectuée par comparaison sans/avec gamification).

Un autre framework de conception plus détaillé a été proposé par un travail de recherche [Alaa, 2014]

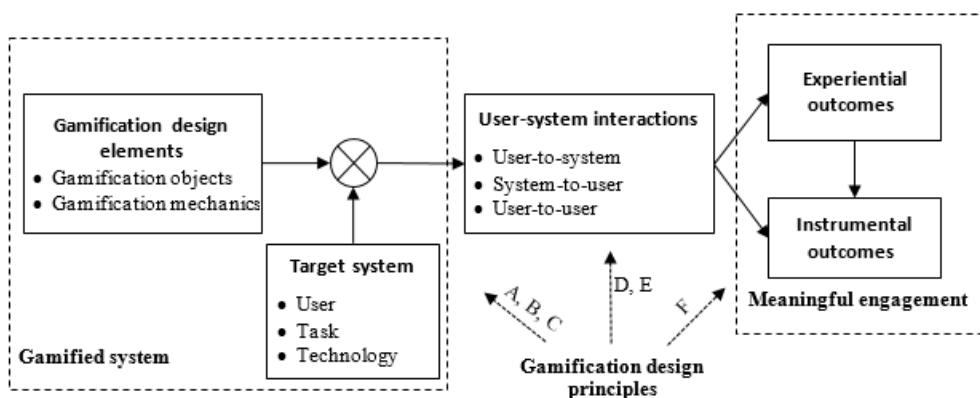


Figure 14: Framework de conception gamifiée

¹⁴ SI : Système d'Information

Ce Framework prévoit 6 principes à respecter lors de la conception et plus particulièrement lors de l'étape du choix d'un mécanisme. Donc ainsi il faut toujours choisir un mécanisme qui correspond :

- A [dans le système lui-même]: à la fonctionnalité que l'on souhaite gamifier ;
- B [dans le système lui-même]: à l'utilisateur type de la fonctionnalité ;
- C [dans le système lui-même]: à la technologie que l'on va utiliser ;
- D [dans les interactions]: aux interactions souhaitées (joueur seul, joueurs contre joueurs ou encore entraide) ;
- E [dans les interactions]: à la fréquence d'utilisation désirée ;
- F [dans la partie d'évaluation de la performance]: aux données d'analyse produits de l'itération précédente.

1.2. Les gestes écoresponsables au sein d'une ville intelligente

La gamification a donc pour but de favoriser l'intérêt d'un utilisateur à effectuer une tâche dite sérieuse. Dans notre cas on souhaiterait accroître l'intérêt à réaliser des gestes écoresponsables et plus particulièrement au sein des villes intelligentes. Ainsi, les applications qui pourraient nous inspirer devraient répondre à ce même objectif. Mais avant cela il nous a paru intéressant d'en apprendre plus sur le domaine des gestes écoresponsables et des villes intelligentes. En effet comme nous l'avons vu précédemment l'application de la gamification et le choix des mécanismes dépendent fortement des fonctionnalités, des utilisateurs et des objectifs. Ces dernières données doivent donc être bien définies.

1.2.1. Classification des gestes écoresponsables

Les choix actions qui peuvent favoriser ou au contraire défavoriser la protection de l'environnement sont indénombrables, il existe notamment des listes de conseils ou blogs sur de nombreux site web tels que vedura.fr¹⁵ ou sciencepresse.qc.ca¹⁶. Cependant la majorité des choix possibles pourrait être récapitulée dans les cinq catégories suivantes :

1. **Achat** : choisir les labels écologiques, les produits locaux et de saison, choisir des éco-matériaux et éviter les produits chimiques
2. **Consommation d'énergie** : limiter sa consommation en eau, électricité et gaz (éteindre la lumière dans les pièces vides ou encore vérifier le robinet).
3. **Transport** : choisir un mode de transport « doux » (vélo, marche à pied) ou les transports en commun plutôt qu'un mode « dur » (voiture, moto).
4. **Traitement des déchets/Réutilisation et recyclage** : choisir d'offrir une deuxième vie à un objet ne pouvant plus remplir sa fonction initiale.

¹⁵ <http://www.vedura.fr/guide/eco-geste/>

¹⁶ <http://www.sciencepresse.qc.ca/blogue/2011/04/15/leco-responsabilite-dela-green-washing>

5. **Eco-informatique** : gérer sa production et exploitation de données (nombre de données, lieux de stockage, fréquence d'utilisation).

De ces choix écologiques sont nées au fil du temps des branches d'activités économiques écologiques telles que l'éco-gastronomie, les énergies renouvelables, l'écoconstruction ou encore l'éco-gestion qui sont enseignées aujourd'hui comme des nouvelles sciences.

1.2.2. Les villes intelligentes

Le concept de ville intelligente est, d'après la définition présente sur le site du CeRCI¹⁷, « relié à trois composantes : les citoyens, les gouvernements, la technologie. La technologie permet aux citoyens d'interagir ensemble, permet aux gouvernements d'offrir de nouveaux services aux citoyens, permet aux gouvernements de s'organiser autrement à l'interne, et finalement permet de mettre en place de nouvelles façons d'interagir entre les citoyens et les gouvernements ».

Cette vision se retrouve également dans certains travaux [Nam, 2011] :

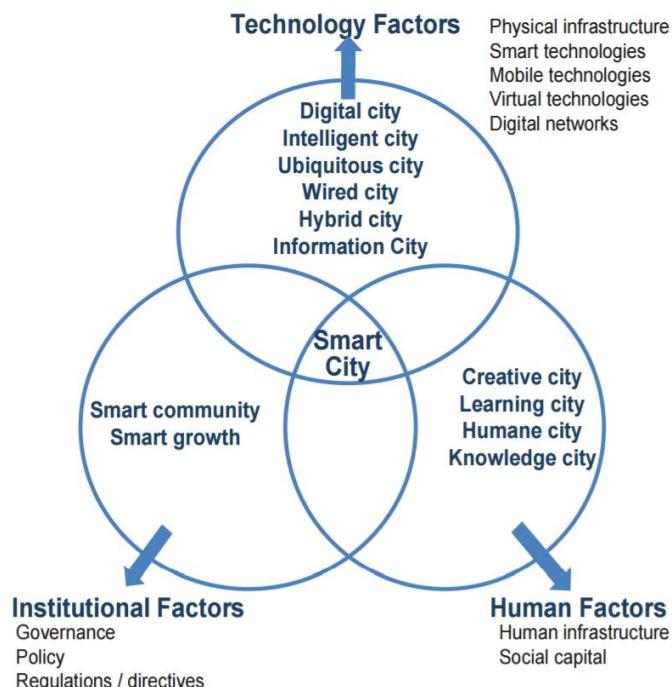


Figure 15: Les 3 composants d'une ville intelligente

Ainsi une application souhaitant être intégrée dans le mécanisme d'une ville intelligente devrait respecter au moins deux conditions : avoir pour objectif d'améliorer la qualité de vie des citoyens de cette ville et favoriser la propagation des services offerts par la gouvernance de la ville.

On peut noter également une spécificité du citoyen d'une ville intelligente : il ne se contente pas de profiter des services offerts par la ville mais produit également des informations (opinions ou

¹⁷ Site web CeRCI : <http://www4.fsa.ulaval.ca/la-recherche/centres-groupes-et-laboratoires/centre-de-recherche-sur-les-communautes-intelligentes-cerci/mission/>

connaissances) et des services. Un néologisme anglais est d'ailleurs apparu pour décrire ce comportement de consommateurs/producteurs, désormais appelés les « *procumers* ».

1.2.3. Classification des services intelligents

Les gouvernances des villes intelligentes utilisent donc les données ouvertes, les données sociales ou encore les données issues de l'internet des objets (particulièrement des réseaux de capteurs) pour effectuer ses choix de gestion et produisent elle-même de telles données pour les citoyens ou les autres villes. Ces sont ces choix d'infrastructure ou de gestion qu'elles vont mettre en place qui font d'elles des villes intelligentes. Voici une classification, très présente dans la littérature, des paramètres qui permettent d'affirmer qu'une ville est intelligente :

1. **Agriculture intelligente** : réutilisation et purification naturelle des eaux usées, gestion collaborative des productions agricoles (partenariat), compostage et recyclage des déchets.
2. **Maisons intelligente** : maisons sous contrôle distant (lumières, gaz, arrosage).
3. **Education intelligente** : nombre et qualité des instituts, création d'espaces étudiants (café culturel, bibliothèques).
4. **Sécurité intelligente** : mise en place dans la ville de caméras et de numéro d'urgence, choix d'un urbanisme responsable, sensibilisation.
5. **Energies intelligentes** : gestion de la consommation des éclairages publics ou du gaz, mise en place de poubelle de tri sélectif.
6. **Santé intelligente** : mise en place de services de soins, d'hôpitaux, maîtrise de la pollution dans la ville, campagne de sensibilisation.
7. **Transport intelligent** : création de pistes cyclables et développement de réseaux de transports en commun, publication d'informations sur le trafic dans la ville, les horaires et positions des transports, création d'infrastructures (TGV, aéroport).
8. **Magasins intelligents** : permettre l'achat à distance, publication d'informations telles que la position, le prix ou la qualité ressentie sur les lieux d'achat (magasins, restaurants, parc) présents dans la ville.

Parmi ces paramètres de nombreux sont fortement reliés à la protection de l'environnement et au développement durable.

II. Applications similaires

Maitrisant à présent les différentes notions intervenant dans le sujet (gamification, gestes écoresponsables et villes intelligentes), il nous est paru ensuite important d'avoir une visibilité sur les différents travaux qui ont été fait dans le même sens. Cela nous permettrait d'un côté de ne pas recréer un projet déjà existant mais aussi de s'inspirer des idées et fonctionnalités qui ont été pensées pour construire notre propre produit.

Pour cela nous étudions deux principaux types d'applications :

- des travaux de recherche décrits dans des articles scientifiques publiés.
- des applications web et mobiles choisies pour leur qualité en termes de note sur le store et de citations dans des sites web spécialisés.

2.1. Travaux de recherche

2.1.1. « GreenGame »

Le premier travail « *GreenGame* » est une application mobile développée dans le cadre de l'étude de cas d'un travail scientifique [Kazhamiakin, 2015]. Cette application a permis de valider l'apport de la gamification dans le processus de choix du type de transport effectué par le citoyen. Pour cela la validation s'est faite en trois étapes :

1. Semaine 1 : laisser l'utilisateur choisir son mode de transport
2. Semaine 2 : conseiller l'utilisateur simplement
3. Semaine 3 : intégrer l'utilisateur dans un processus gamifié (points selon le choix, classement des joueurs, attestation de participation, affichage d'une barre de santé de l'utilisateur selon ses choix).

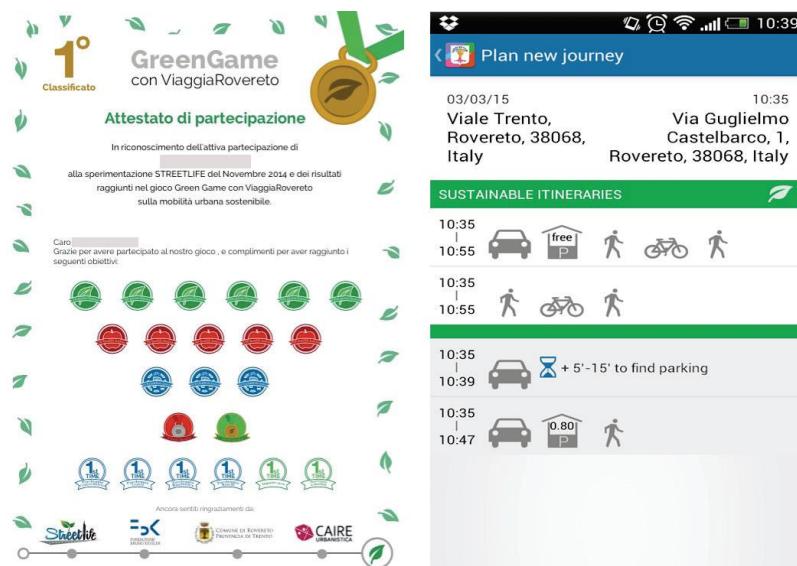


Figure 16: Interface du jeu GreenGame

Les tests ont prouvé que le choix de l'utilisateur s'améliore drastiquement lors de la phase gamifiée de l'expérience. Seulement, dans ce projet le chercheur fait confiance à l'utilisateur et ne vérifie pas réellement s'il a appliqué le choix rempli dans l'application.

2.1.2. « Mecco »

Mecco est une plateforme sociale ayant pour but de promouvoir l'écologie. Elle a été développée dans le cadre d'un projet de l'université Ramon Lull et a fait l'objet d'une étude de cas d'un travail de recherche [Danae, 2011].

Mecco cible un public de jeunes adultes (23-35 ans) possédant un Iphone et est uniquement disponible en espagnol. L'application est basée sur le concept de lieu clés sur la carte (en utilisant l'API Google Map). On y trouve des lieux clés à visiter, des challenges, des conseils ou encore publicités pour les marques écoresponsables.

La validation se fait dans ce cas par géolocalisation et avis de la communauté contrairement au projet précédent.



Figure 17: Interface de Mecco

2.2. Applications et jeux sérieux

Bien que les travaux de recherches soient assez peu nombreux dans ce domaine, il existe une grande variété d'applications gamifiées sur le « Google Play Store » ou sur internet. La plupart sont orientés conseil et lieux utiles (« Astuce Ecolo », « Pour ma planète ») ou encore challenges personnels (« 90 jours »), d'autres sont engagés comme « Ecosia » le moteur de recherche web et mobile qui reverse 80% de ses bénéfices pour la cause de l'environnement. Nous avons pu remarquer que la formation est un domaine assez absent pour ce qui est d'application gamifiée autour de l'écologie.

Nous avons également parcouru les différents « serious games » dont l'aspect sérieux est lié à l'écologie. Le tableau ci-dessous résume les différentes approches de promotion de l'écologie par la gamification que l'on peut trouver à travers les jeux sérieux :



Figure 18: Interface Tri En Folie



Figure 19: Interface Anno 2070



Figure 20 : Interface Botanica



Figure 21: Interface Oiligarchie

Formation aux gestes écologiques :

« *Tri-En-Folie*¹⁸ » est un projet développé par « *Smédar* » en partenariat avec l’Union Européenne. Le but de ce jeu sérieux est purement éducatif : apprendre à reconnaître les types de déchets, les types de poubelles (tri sélectif) et les ressources que l’on peut économiser.

Sensibilisation à l’intérêt du domaine :

Il existe aussi des jeux qui ne forment pas mais qui sensibilisent à l’intérêt de protéger l’environnement. Pour cela différents moyens : le premier consiste à amener le joueur dans l’avenir sombre qui risque d’arriver. C’est le cas du jeu « *Anno 2070*¹⁹ » dans lequel le joueur doit gérer des ressources dans un monde où elles sont épuisées.

Sensibilisation à la beauté de la nature :

Un autre moyen de sensibiliser est d’amener le joueur à aimer la nature. C’est la technique utilisée par différents projets tels que « *Botanica*²⁰ » ou « *Flower*²¹ » de « *thatGameCompany* ». Se sont de très beaux jeux de découvertes dans lesquels le joueur fait partie intégrante de la nature.

Humour et ironie :

Encore un moyen de sensibiliser : l’humour et l’ironie. C’est le choix de jeux tels que « *iPollute*²² » ou « *Oiligarchie*²³ » dans lesquels le joueur pollue et devient un gérant de chaînes pétrolières.

¹⁸ Site web *Tri en Folie* : <https://www.smedar.fr/tri-en-folie>

¹⁹ Lien de téléchargement *Anno 2070* : <http://www.jeuxvideo.com/jeux/pc/00040475-anno-2070.htm>

²⁰ Site web *Botanica* : <http://amanita-design.net/games/botanicula.html>

²¹ Site web *Flower* : <http://thatgamecompany.com/games/flower/>

²² Playstore *iPollute* : https://play.google.com/store/apps/details?id=com.bulkypix.ipollute&hl=fr_CA

²³ Lien de téléchargement *Oiligarchie* : <http://www.molleindustria.org/en/oilgarchy/>



Figure 22: Site web d' Ingress

Immersion et réalité virtuelle :

Enfin il existe un type de jeux basé sur l'intégration dans la réalité tel que « *Ingress*²⁴ », un jeu de gestion qui utilise la réalité virtuelle, la géolocalisation et la carte (avec des lieux réels stratégiques à visiter). Ainsi l'écologie naît dans le jeu et fini dans la réalité.

2.3. Gamification de la formation

Souhaitant nous orienter vers la formation des gestes écoresponsables, nous avons également souhaité avoir une idée des applications existantes en termes de formation gamifiée. Certains travaux [Flores, 2016] mettent en opposition la méthode utilisée dans l'apprentissage en ligne à travers les MOOCs²⁵ classiques qui consiste généralement à suivre une série de vidéos avant de passer un examen final, avec une méthode gamifiée. Cette deuxième méthode contient des éléments tels que :

- L'affichage d'une barre de progression qui permettrait à l'apprenant de mesurer son avancement ou le niveau de difficulté (via un code couleur) ;
- Des quiz réguliers plutôt qu'un examen final. Ces quiz seraient de préférence très interactifs et pas uniquement limité à des questions à choix multiples ;
- L'affichage d'un rang (appelé « *Leaderboard* ») permettant aux apprenants d'être en compétition et de se situer par rapport au groupe ;
- Des informations non présentes directement dans le cours, à chercher soi-même pour créer de l'implication chez l'apprenant.

Nous avons donc étudié des applications qui ont appliqué cette méthodologie et qui ont connu un succès commercial afin de nous en inspirer :

2.3.1. Duolingo

Duolingo est une application mobile disponible sur le « *Google Play Store* » qui permet d'apprendre plusieurs langues à partir de plusieurs autres langues. Cette application applique sans conteste les mécanismes et l'esthétique de la gamification :

²⁴ Site web Ingress : <https://www.ingress.com/>

²⁵ MOOC : Massive Open Online Course (Cours en ligne ouvert et massif)

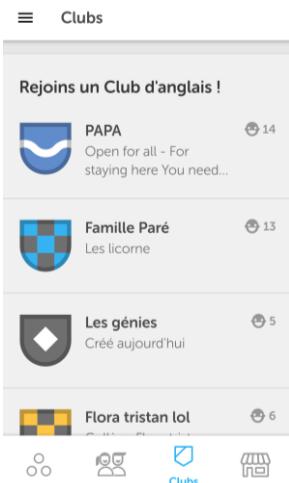


Figure 23: Les clubs dans Duolingo

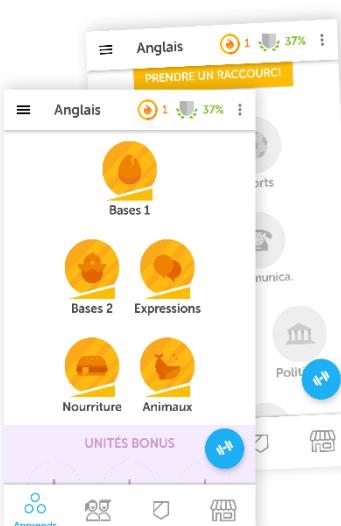


Figure 24: Les niveaux dans Duolingo

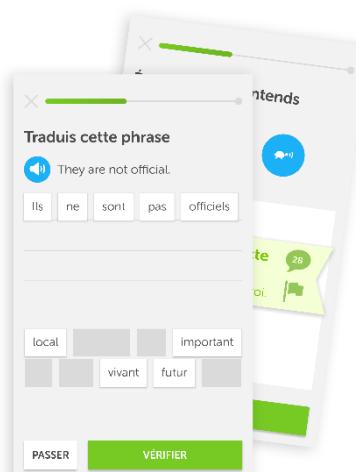


Figure 25: Les quiz dans Duolingo

Tout d'abord, Duolingo met en place les principes de coopération et compétitions via un mécanisme d'équipes d'apprentissage appelées « clubs ». Ainsi l'interaction joueur-joueur est ajoutée à un domaine où généralement l'utilisateur est seul face à son cours.

Ensuite, Duolingo ne s'est pas contenté de mettre en place une simple barre de progression verticale mais plutôt une véritable page où chaque niveau change de couleur quand il est acquis. La représentation graphique du thème présent dans chaque niveau (animaux, commerce, technologie ...) permet de susciter l'intérêt de l'apprenant.

Enfin, les quiz sont de véritables jeux interactifs et non pas de simples vrai/faux. On y retrouve des questions auxquelles l'apprenant doit répondre vocalement, d'autres en déplaçant des objets (images ou mot) et certaines où il doit écouter des phrases.

2.3.2. Projet Voltaire

Le « *Projet Voltaire* » est une plateforme web et mobile dont l'utilisation permet d'améliorer son niveau en français et plus particulièrement en orthographe. Très proche de Duolingo, le Projet Voltaire propose également une interface d'avancement attrayante, des niveaux à débloquer et des quiz interactifs dans lesquels l'apprenant doit sélectionner les mots mal orthographiés dans une phrase ou encore classer des mots selon leur catégorie.

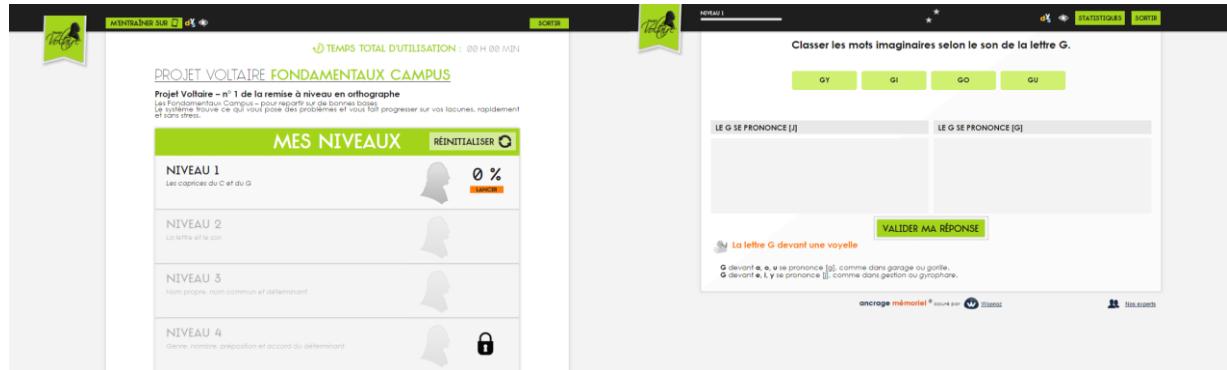


Figure 26: Interfaces du Projet Voltaire

III. Choix et orientations

Maitrisant à présent les notions de gamification, de gestes écoresponsables et de villes intelligentes et ayant une vue d'ensemble sur les projets réalisés autour de ces notions, le moment était venu pour nous de faire des choix quant à ce que nous allions concrètement développer. Ces choix ont été faits sur la base de tout ce qui a été vu précédemment, afin d'en filtrer ce que nous jugions à la fois pertinent pour notre projet et réalisable avec les moyens technologiques et le temps que nous disposions.

3.1. Méthodologie d'évaluation de l'impact

Cependant avant de décider des fonctionnalités de l'application, nous avons remarqué que le domaine des gestes écoresponsables pose parfois un problème : celui de la collecte de données pour effectuer des validations. Par exemple comment vérifier si un utilisateur effectue bien le tri sélectif ? Comment vérifier s'il consomme peu ou beaucoup d'eau ? Sans avoir accès à des technologies de capteurs et objets intelligents, la plupart des projets de recherche font confiance dans les données fournies par l'utilisateur sans toutefois avoir les moyens d'en vérifier la véracité.

En nous inspirant des projets étudiés précédemment, voici les différents moyens de collecte de données que nous avons choisi d'utiliser dans le projet :

- Les quizz gamifiés pour vérifier s'il a bien retenu une information
- L'avis des autres membres pour mesurer la qualité (d'un cours, d'un fait, d'une réalisation ou autre) : par signalement, notation ou encore commentaire
- Le recueil de données via les capteurs du téléphone : géolocalisation principalement et interaction avec les API disponible (« *Google Map* » ou encore des données ouvertes telles que www.donneesquebec.ca)

3.2. Orientations en termes de gamification

Ensuite nous avons validé une série de choix qui sont apparus lors de la revue de la littérature :

- Réaliser un projet de type gamification et pas un jeu sérieux. Nous souhaitions créer une plateforme sociale de promotion des gestes écoresponsables ;
- Rendre le projet accessible à un maximum de personnes et donc s'adapter aux différents types d'utilisateur vus précédemment (classification de *Bartle*) mais aussi aux différents types de supports (web, ios, android) ;
- Faire de la sensibilisation, de l'information et de la formation gamifiée ;
- Utiliser le principe des lieux réels vu dans le jeu *ingress* afin de faire intervenir la réalité dans le jeu ;
- Suivre le framework de conception de fonctionnalités gamifiées lors de la phase de conception du projet.

3.3. Méthodologie de production d'informations

Une dernière problématique se posait : informer et former en écologie et gestes écoresponsables nécessite d'avoir une base de données d'informations à transmettre aux utilisateurs. Nous avions alors deux choix : soit se positionner comme experts qui fournissent l'information soit choisir de copier le modèle de plateforme collaborative telles que *Wikipedia* ou *YouTube* dans lesquelles l'utilisateur est à la fois consommateur et producteur (tel que le conçoit le concept « *procumers* » au sein des villes intelligentes) et l'administration du projet ne s'occuperaît alors que de la gestion des abus.

C'est cette deuxième option que nous avons choisi et ainsi notre projet est entré dans la catégorie des projets appliquant l'« effet réseaux²⁶ », un type de projet pour lequel l'utilité est due non pas au produit en tant que tel mais au nombre de personnes qui l'utilisent.

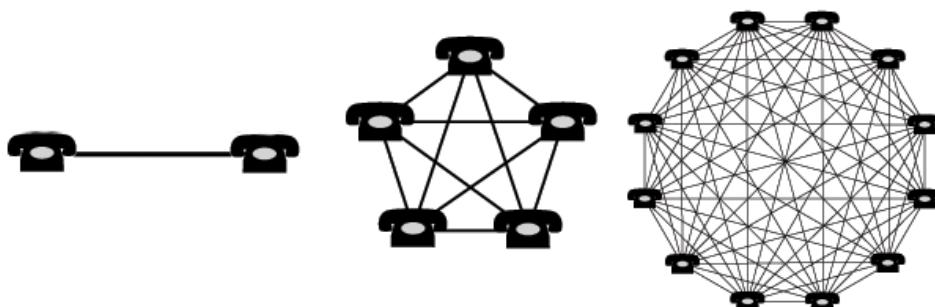


Figure 27: L'effet réseau sur l'utilisation du téléphone

Sur la base de ces trois réflexions, nous étions enfin prêts à commencer à réellement concevoir le projet.

²⁶ Article Wikipédia l'effet réseaux : https://fr.wikipedia.org/wiki/Effet_de_r%C3%A9seau



Sustainapp

I. Analyse des besoins

L'étude suivante récapitule l'ensemble des besoins de l'application, les premiers concernent les fonctionnalités à développer et la gamification de ces fonctionnalités et les seconds concernent les besoins en termes de qualité (performance, maintenabilité, sécurité).

1.1. Application de la méthodologie de conception gamifiée

Nous avons préféré avant de modéliser les cas d'utilisations de l'application et les différents acteurs qui entrent en jeu, appliquer la méthodologie de conception de fonctionnalités gamifiées. En effet la gamification d'une fonctionnalité risque fortement d'ajouter ou de modifier un cas d'utilisation et doit donc être pensée en premier.

Etape 1 : Les objectifs du projet

L'objectif principal du projet est de promouvoir le comportement et les actions écoresponsables chez le citoyen d'une ville intelligente. Cet objectif est réalisé selon les trois principaux items (sous-objectifs) suivants :

1. *Sensibiliser* : donner envie au citoyen de devenir écoresponsable.
2. *Informer/Former* : permettre au citoyen d'être capable de devenir écoresponsable.
3. *Evaluer* : vérifier si les citoyens sont bien écoresponsables.

Etape 2 : Les fonctionnalités pour chaque objectif

1. *Sensibiliser* : avoir une identité à l'aide d'un profil, faire partie d'une équipe.
2. *Informer/Former* : suivre/créer des cours, utiliser une barre de recherche, connaître les éco-lieux, être notifié des actualités.
3. *Evaluer* : participer/voter dans des challenges, signaler un fait non écologique, noter la qualité d'un cours, passer des quizz de vérification de connaissances.

Etape 3 : Gamification des fonctionnalités

Profil et équipe :

- Ajout de la notion de niveau du profil/équipe qui monte selon les actions et permet l'accès à de nouvelles fonctionnalités.
- Ajout de badges à acquérir dans le profil pour les capitaines d'équipes.

Cours :

- Ajout de badges à acquérir dans le profil pour les créateurs de cours bien notés ou ceux qui ont suivi et validé des cours.
- Affichage d'un classement des meilleurs cours et récompense des créateurs des cours présents dans le haut du classement.

- Ajout des principes de gamification de la formation (barre d'avancement, quizz interactifs)

Visite des éco-lieux :

- Ajout d'un badge dans le profil de ceux qui ont réellement visité des lieux écologique.
- Ajout également de la notion de note sur les lieux

Challenge :

- Affichage d'un classement des meilleures participations dans les challenges et récompense des profils qui ont fait partie du haut du classement.

Signalement :

- Récompense par un badge pour les profils dont les signalements auront été validés par l'administration de l'application

Le respect des différents types d'utilisateur doit également être pris en compte, ainsi les profils auront le choix soit d'être invisible et de profiter des fonctionnalités de découvertes (villes, cours...) ou au contraire d'être mis en avant et de se comparer aux autres (challenges, classement ...)

1.2. Spécification des besoins fonctionnels

Après avoir fait valider les grandes orientations fonctionnelles par les membres du CeRCI, nous pouvions approfondir et modéliser donc en détails l'ensemble des fonctionnalités. Bien évidemment, travaillant de manière Agile, les diagrammes qui vont suivre ont évolué au gré des itérations seules les versions finales seront présentées.

1.2.1. Acteurs de Sustainapp

Quatre types de rôles ou acteurs sont susceptibles d'utiliser Sustainapp :

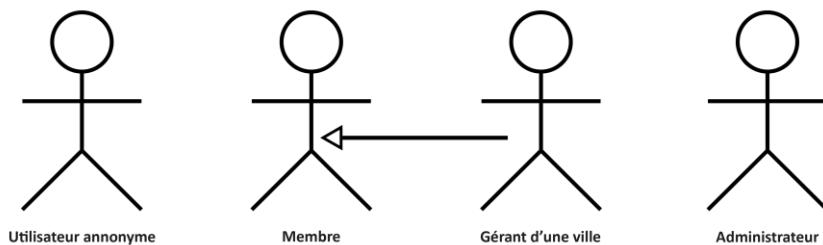


Figure 28: Les acteurs de Sustainapp

- **L'utilisateur anonyme** : est un acteur qui n'est pas encore connecté ou même inscrit et pour lequel les seules fonctions accessibles sont la connexion et la création d'un compte.
- **Le membre** : est un utilisateur connecté sur un compte Sustainapp.

- **Le gérant d'une ville** : est un membre et peut donc réaliser absolument toutes les fonctionnalités accessibles par celui-ci mais est en plus en charge de la page officielle d'une ville.
- **L'administrateur** : est un membre qui a accès à l'ensemble des fonctionnalités mais qui de plus est en charge d'une page d'administration à travers laquelle il peut accéder à certaines données ou valider des signalements ou des demandes de confirmation d'une ville.

1.2.2. Décomposition en packages

Le nombre de cas d'utilisation étant assez élevé, nous avons préféré modéliser en utilisant la notion de packages. Ainsi nous avons un package par livrable (version de l'application en fin d'une itération). Voici donc les différents packages :

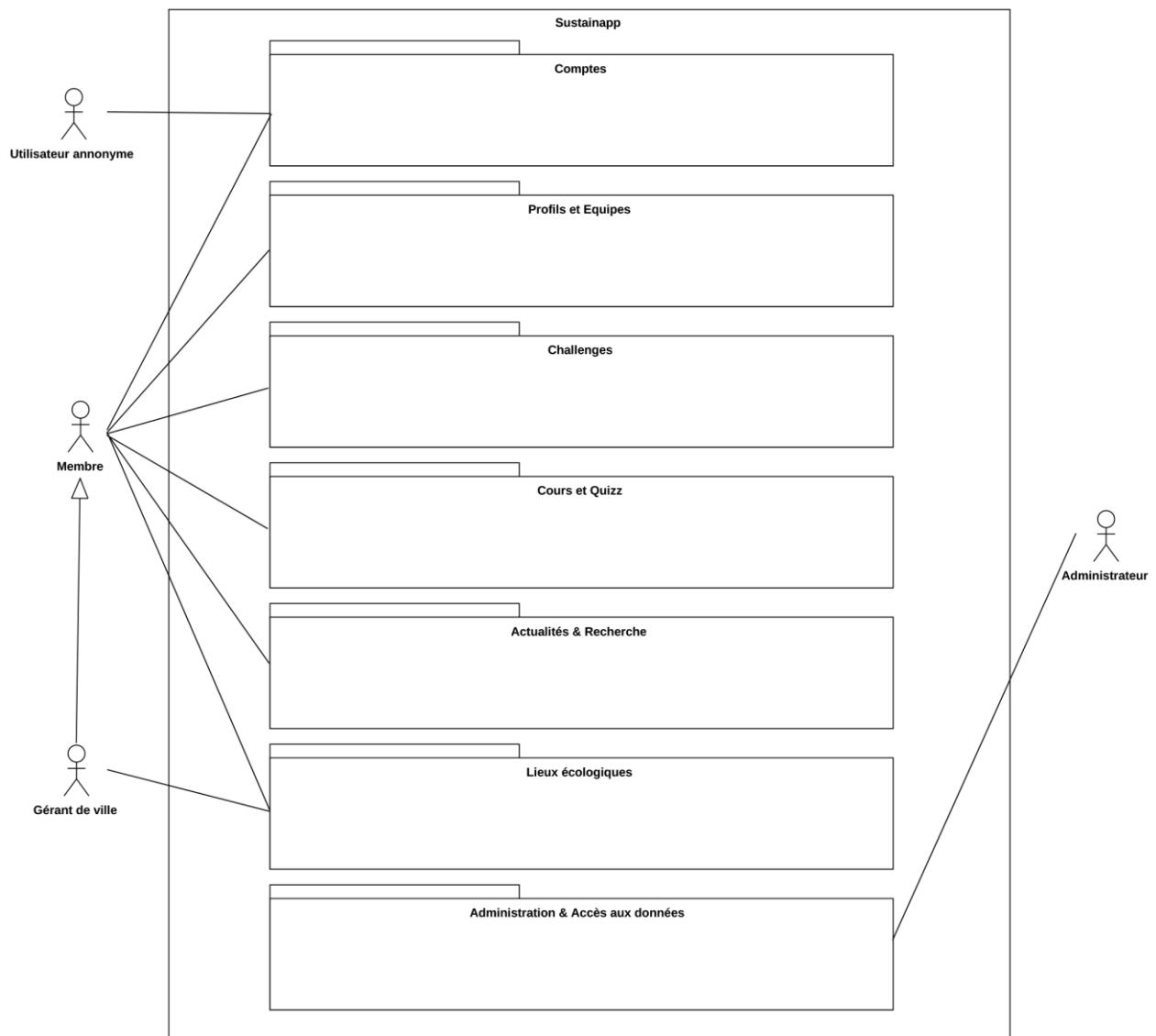


Figure 29: Diagramme de packages de Sustainapp

1.2.3. Les profils et équipes

Les deux premiers livrables concernent la mise en place des fonctionnalités de base (inscription, connexion, gestion d'un profil) ainsi que la gestion et participation dans les équipes.

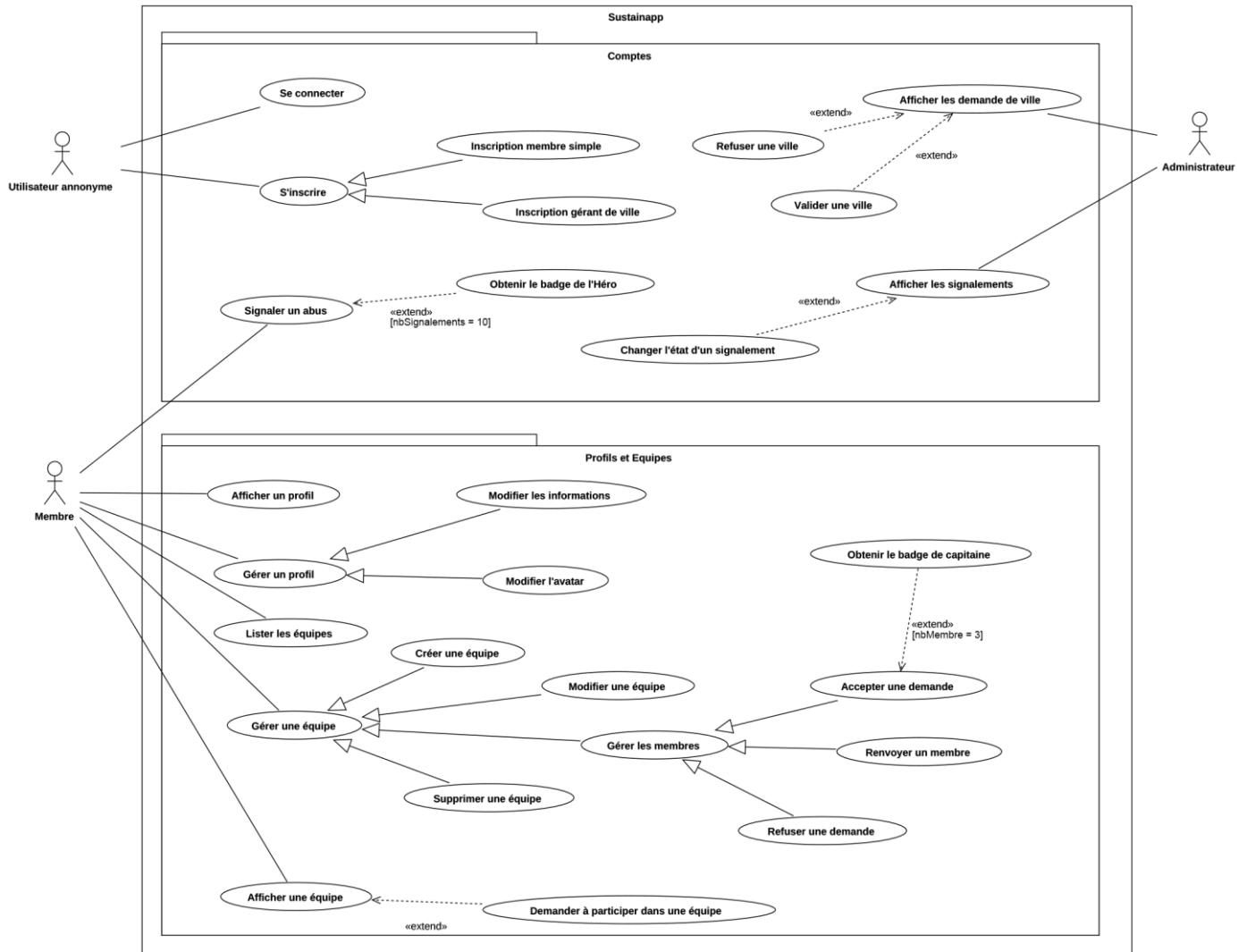


Figure 30: Cas d'utilisation 1 : profils & équipes

Comme nous pouvons le voir l'obtention des badges fait partie intégrante des cas d'utilisation. Sur ce diagramme il existe deux modes d'inscription : ville et profil simple et que l'inscription en mode ville nécessite une validation par l'administrateur. On peut également observer que la participation à une équipe nécessite la validation du « gérant/capitaine » de cette équipe.

1.2.4. Challenges et cours

Ensuite, une fois d'accord sur la structure de profil et d'équipe nous avons réfléchi aux trois grands blocs fonctionnels de l'application : les éco-lieux à visiter, les cours & quizz et les challenges écologiques. Ce deuxième diagramme présente les challenges et les cours & quizz :

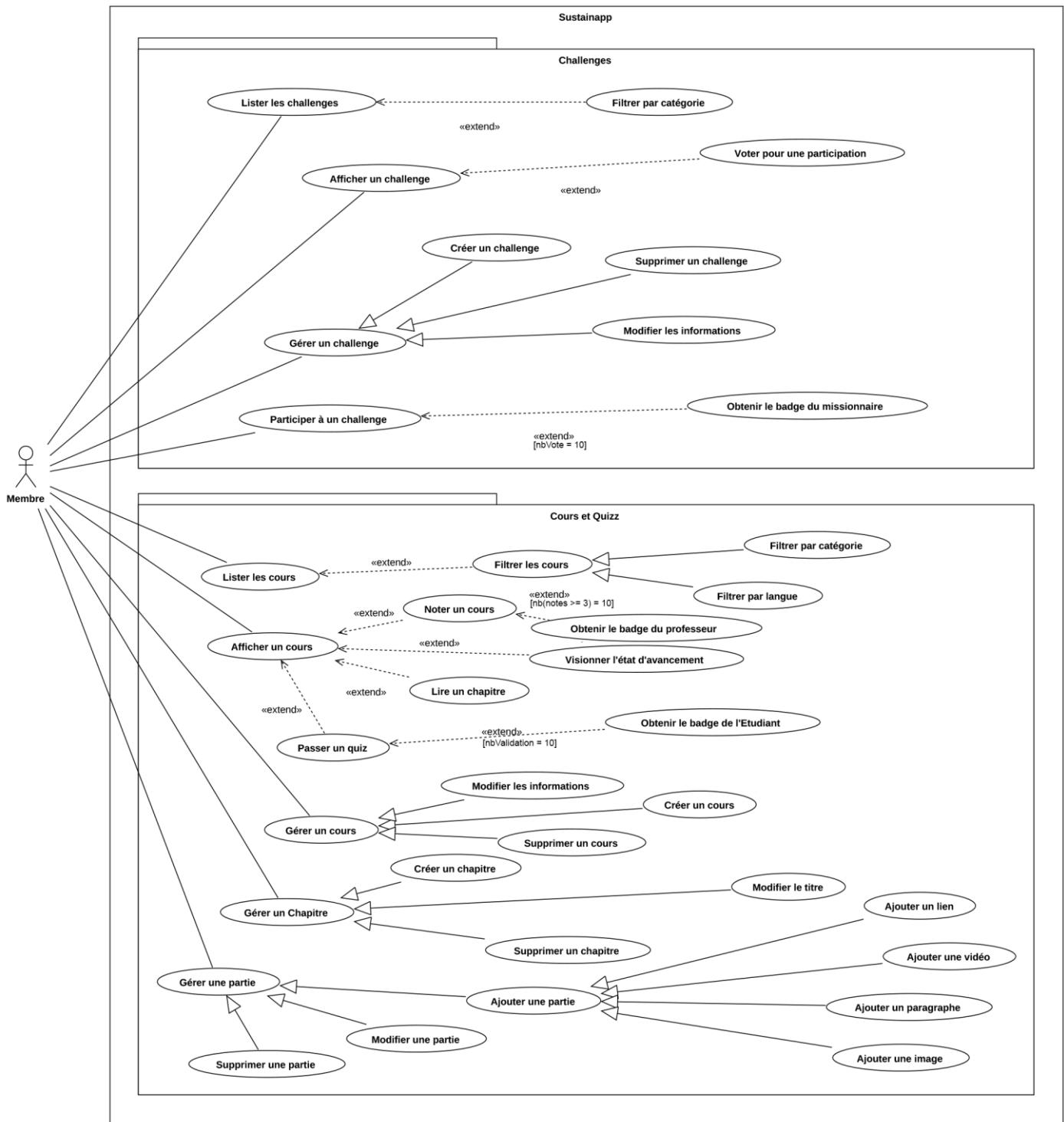


Figure 31: Cas d'utilisation 2 : challenges & cours

Comme précisé dans la description des acteurs, toutes ces fonctionnalités accessibles aux membres sont donc accessibles au gérant de ville et administrateur de par le fait qu'ils sont également des membres.

Le système de challenge est assez simple : un membre crée un challenge, certains y participent par l'envoi d'une réalisation et le reste de la communauté peut voter pour la réalisation qu'ils préfèrent.

Le système de cours est quant à lui beaucoup plus complexe et se compose de deux grands groupes de fonctions : la gestion des cours par le créateur de celui-ci (et donc des chapitres ou des questions) et la validation des cours par les membres (lecture du cours, réponse aux questions, note du cours).

1.2.5. Actualités et éco-lieux

Ce troisième diagramme présente donc le troisième grand bloc fonctionnel : les éco-lieux mais également des fonctionnalités auxiliaires qui ont surtout été pensées pour améliorer l'expérience utilisateur et la facilité d'utilisation de l'application (être notifié des nouveautés ou avoir accès à un moteur de recherche).

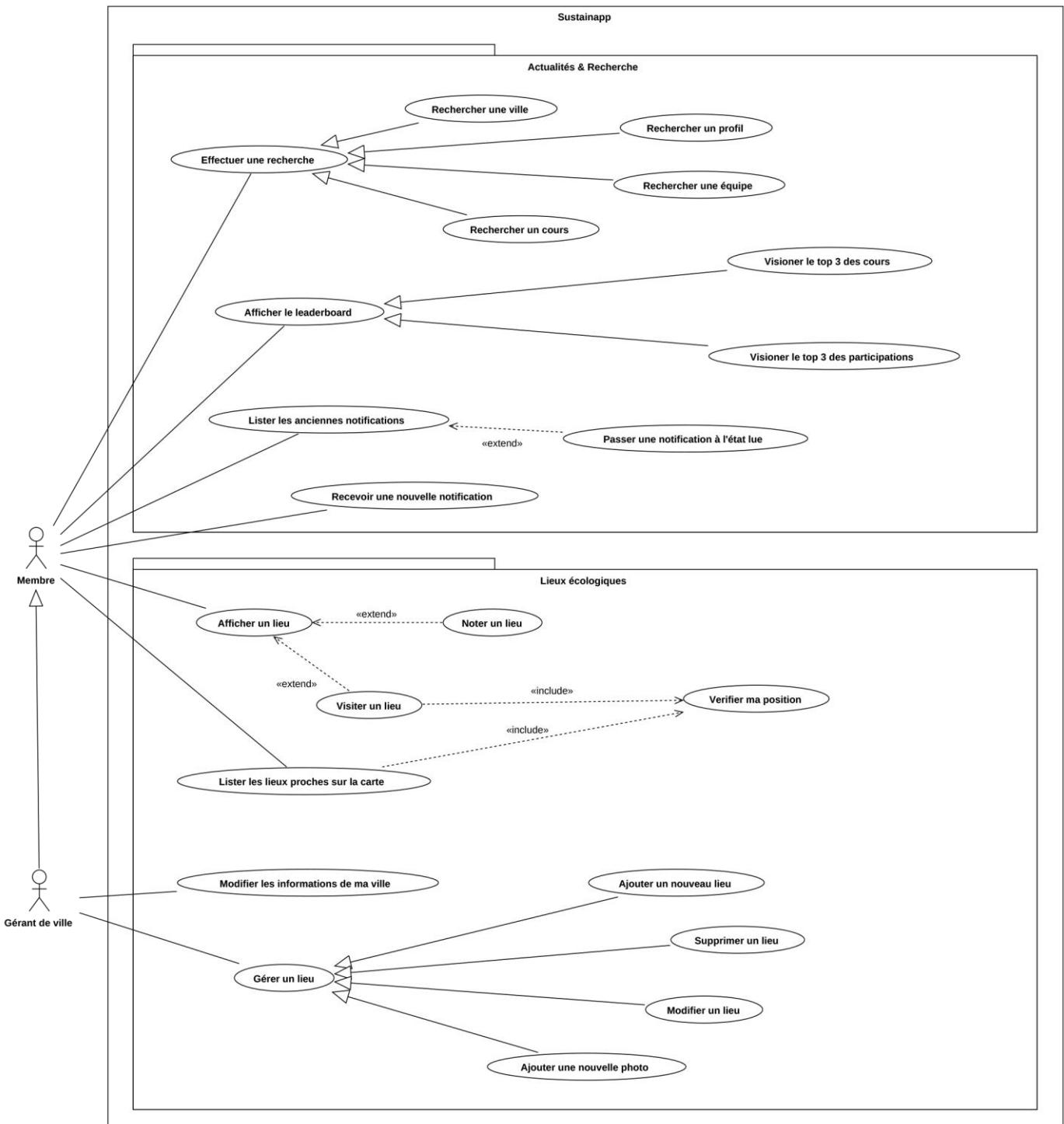


Figure 32 : Cas d'utilisation 3 : éco-lieux et actualités

Pour ce qui est des actualités, le principe est assez commun à de nombreuses applications web et mobiles. Le moteur de recherche permet de trouver un cours, un profil, une équipe ou un lieu par son nom principalement. Les notifications permettent d'être toujours à jour avec les votes que l'on a reçu, les notes sur nos cours, les badges ou encore les demandes et acceptations dans les équipes.

Comme on peut le voir dans le second package, seules les villes sont habilitées à créer et gérer des lieux, qui pourront être visités par des membres simples.

1.2.6. Administration

La page d'administration permet de visualiser des données via des graphiques et statistique sur ce qui se passe au sein de l'application afin d'évaluer le bon déroulement de la promotion des gestes écoresponsables et d'aider l'administration à prendre des décisions quant à l'avenir du projet:

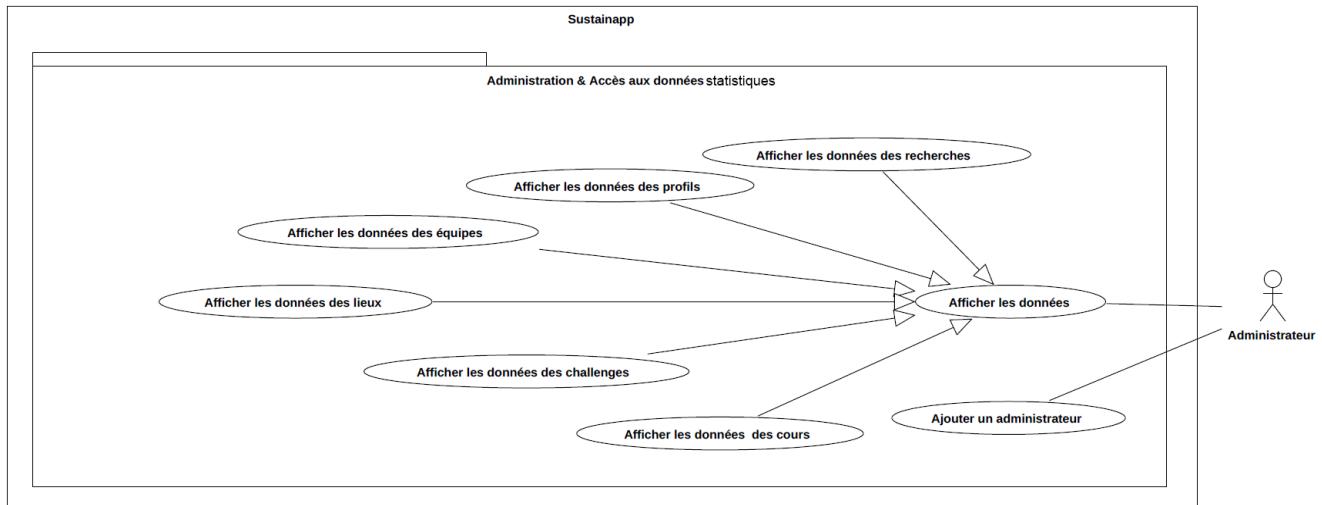


Figure 33: Cas d'utilisation 4 : administration

1.3. Spécification de la qualité

La qualité de Sustainapp est garantie par un ensemble de règles applicables aux différentes phases du projet. Pour ce qui est de la qualité du produit lui-même nous avons suivi la norme ISO/IEC 9126 (voir figure 10) tandis que pour la qualité du processus de création nous nous sommes basés sur la norme CMMI.

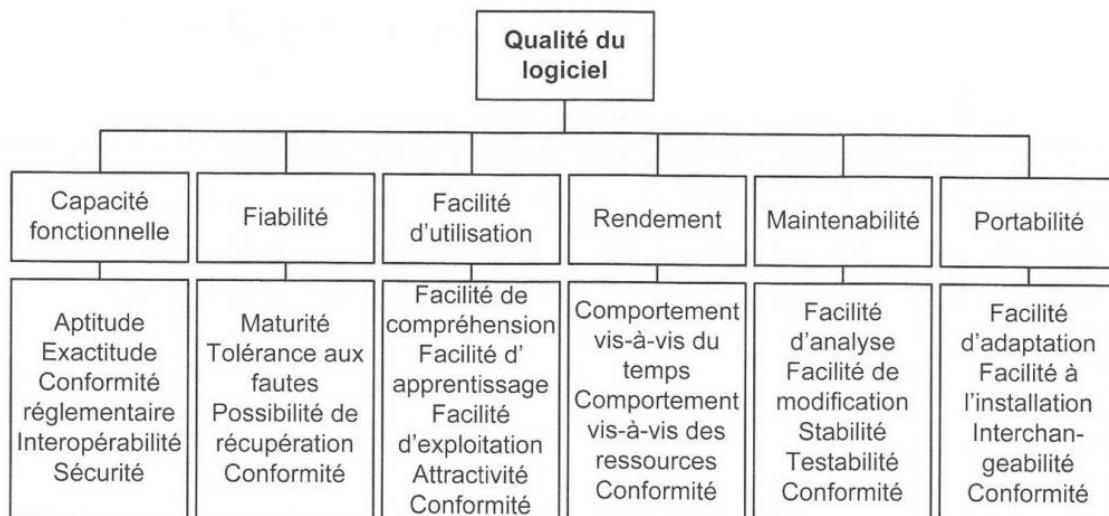


Figure 34 : Les facteurs de qualité de la norme ISO 9126

Portabilité :

- Utilisation de technologies cross-plateformes durant la phase de développement afin de pouvoir compiler nativement sur divers système d'exploitation mobile ou obtenir une version web ;
- Déploiement sur les stores les plus connus de manière à faciliter l'installation et la mise à jour.

Maintenabilité :

- Respect de règle de nommage des variables, méthodes, classes ou encore fichiers (« *Camelcase* »²⁷, nom explicite) ;
- Organisation des fichiers et du code de manière à respecter les principes de l'architecture logicielle orientée service ainsi que le MVC²⁸ (séparation des données, des traitements et de l'interface) ;
- Documentation du code en utilisant les annotations prévues à cet effet (date, version, description), de la conception technique en utilisant UML ainsi que de la conception graphique ;
- Documentation de la machine serveur, logiciels installés et méthode de redémarrage des services ;
- Respect de règles de code (limite de taille de méthodes, pas de code en doublon, pas de code non utilisé) ;
- Bonne utilisation des principes de la POO (réutilisabilité et autonomie).

Rendement :

- Bonne gestion du cycle de vie des objets (Injection de dépendances²⁹, singletons, limite des instantiations lente en temps d'exécution) ;
- Optimisation de la taille des réponses REST via un protocole;
- Compression des images et fichiers multimédia avant enregistrement en base ;
- Bonne gestion de l'*ORM* pour éviter les allers retours inutiles dus aux annotations de jointures ;
- Utilisation de *framework* modernes ;
- Utilisation de technologies temps réel (websocket) lors de la phase de développement.

Facilité d'utilisation :

- Respect des habitudes visuelles et graphiques des utilisateurs (« *material design* »³⁰, menu à gauche, barre d'entête, palette constituée de peu de couleurs) ;
- Schéma de navigation des interfaces simplifiées (avec un maximum de 3 niveaux de profondeur).

²⁷ **Camelcase** : est une manière de nommer une variable ou méthodes dans les langages objets en utilisant les majuscules pour séparer les mots.

²⁸ **MVC** : Model-View-Controller est un type d'architecture permettant une meilleure maintenabilité du code

²⁹ Article sur l'injection de dépendances : https://fr.wikipedia.org/wiki/Injection_de_d%C3%A9pendances

³⁰ Site web officiel du material design : <https://material.io/guidelines/>

Fiabilité :

- Tests unitaires sur tous les modules (serveur, web, mobile) exécuté à chaque intégration à l'aide de l'outil *Jenkins*³¹ ;
- Utilisation de *Jenkins* également pour relancer automatiquement l'application en cas de panne (lors de l'étape du déploiement) ;
- Création d'un système de signalement permettant aux utilisateurs de signaler d'éventuels problèmes à corriger.

Capacité fonctionnelle :

- Validation de chaque fonctionnalité par l'ensemble des membres dirigeants du CeRCI ;
- Gestion de la sécurité (Cryptage en base des mots de passe, utilisation d'un système de token temporel)

L'utilisation combinée de nos principes de conception/développement ainsi que la méthodologie de gestion du projet utilisée nous permet d'atteindre le niveau 2 du *CMMI*³² «discipliné», bien que cela ne puisse être prouvé puisque cela nécessiterait l'avis d'un organisme officiel.

Cependant le niveau « discipliné » implique d'avoir des rôles et responsabilités connus et de respecter les sept domaines suivants :

- **gestion des exigences du client:** réalisé durant les réunions de revue de sprint et validation des fonctionnalités
- **planification de projet :** réalisé grâce au tableau de tâches (dans lequel chaque tâche est mesurée en complexité temporelle) ainsi que du backlog produit.
- **Suivi de projet :** réalisé également durant les réunions et via l'interface du *gitlab*.
- **gestion des fournisseurs :** dans notre cas le CIRRELT possède déjà le matériel nécessaire à la réalisation et l'hébergement du projet.
- **utilisation de métriques :** nous mesurons des métriques au niveau du produit lui-même : le temps de réponse de l'application ainsi que le nombre d'aller en base, mais aussi en ce qui concerne le processus : temps de réalisation des tâches.
- **l'assurance qualité :** réalisé grâce aux règles et usages décrits dans la partie précédente.
- **gestion de la configuration :** réalisé par l'utilisation des environnements *Jenkins* et *Gitlab* côté server et est géré de manière automatique du côté mobile lors de l'installation

³¹ Site web officiel de Jenkins : <https://jenkins.io/>

³² CMMI : Capability Maturity Model Intégration

II. Conception technique et architecturale

Les fonctionnalités ayant été validées par les membres directeurs du CeRCI tout comme les règles garantissant la qualité, le moment était donc venu d'établir la manière de les réaliser. Cependant avant de passer à la modélisation des données ou de l'architecture de l'application, certaines des fonctionnalités nécessitaient des éclaircissements quant à la manière dont elles seraient réalisées. D'autres sont beaucoup plus classiques et sont donc compréhensibles directement.

2.1. Précisions sur les fonctionnalités

Ainsi nous avons modélisé quelques diagrammes de séquences système dans le but d'apporter quelques précisions sur ces fonctionnalités.

2.1.1. Système de notification en temps réel

Afin d'être pertinent, le système de notification doit être temps réel, ce qui implique d'utiliser technologiquement et conceptuellement d'autres outils que pour l'ensemble des autres fonctionnalités. En effet la grande majorité des fonctionnalités suivent le modèle suivant : *Demande du client* (« se connecter » par exemple) et puis *réponse du serveur* (« connexion faite »). Seulement pour les notifications l'utilisateur doit être averti par le serveur sans avoir envoyé de demande et donc s'être enregistré auprès de lui comme étant prêt à recevoir des notifications au besoin.

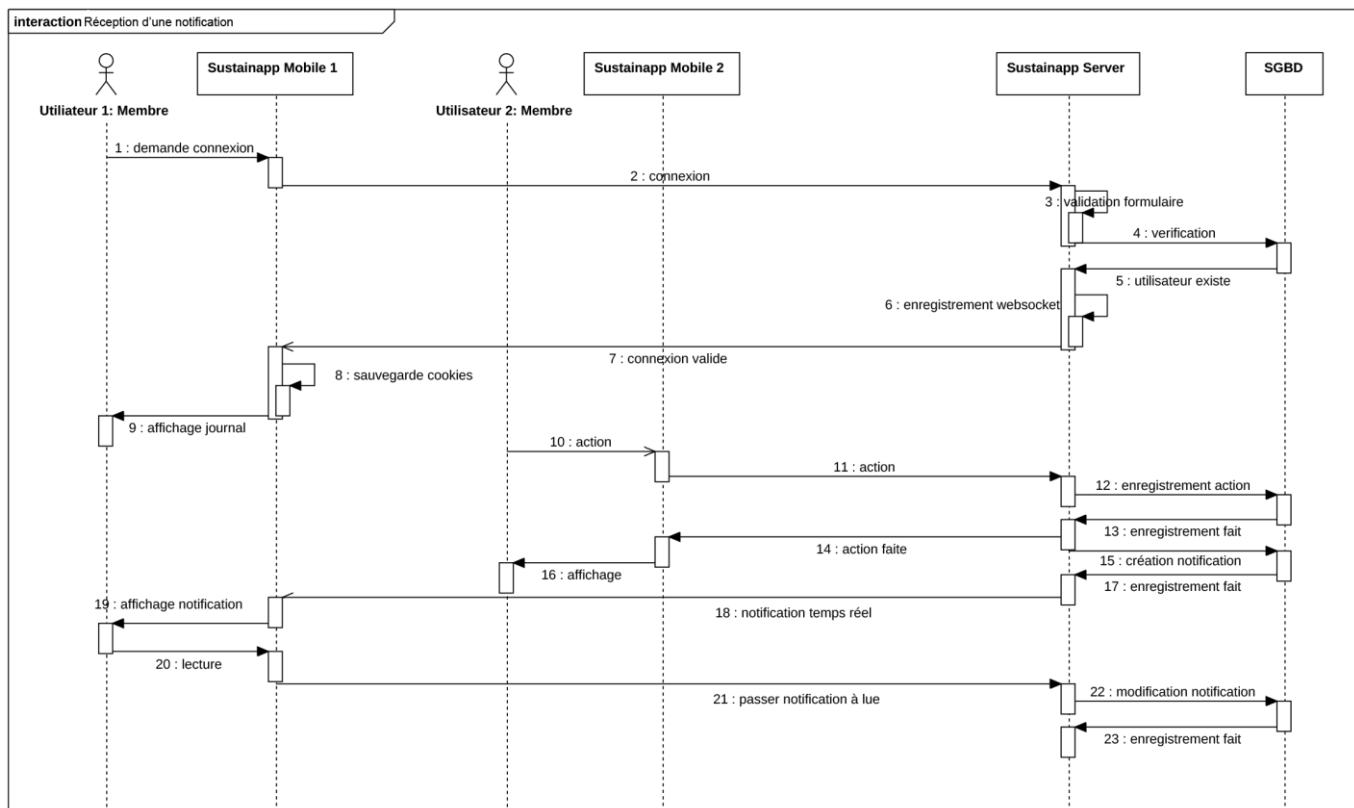


Figure 35: Séquence réception d'une notification temps réel

On peut voir que la connexion est liée à la réception de notifications car se sera durant l'étape de connexion que l'utilisateur sera enregistré comme prêt à recevoir des notifications auprès du serveur.

2.1.2. Affichages des éco-lieux proches

La seconde fonctionnalité à laquelle il nous a paru important d'apporter des précisions est l'affichage des éco-lieux proches sur la carte car cette fonctionnalité fait intervenir d'autres infrastructures que le serveur de *Sustainapp* : le serveur de l'API *Google Map* (pour l'affichage de la carte). Cette fonctionnalité fait également intervenir technologiquement un outil de plus : les capteurs du téléphone.

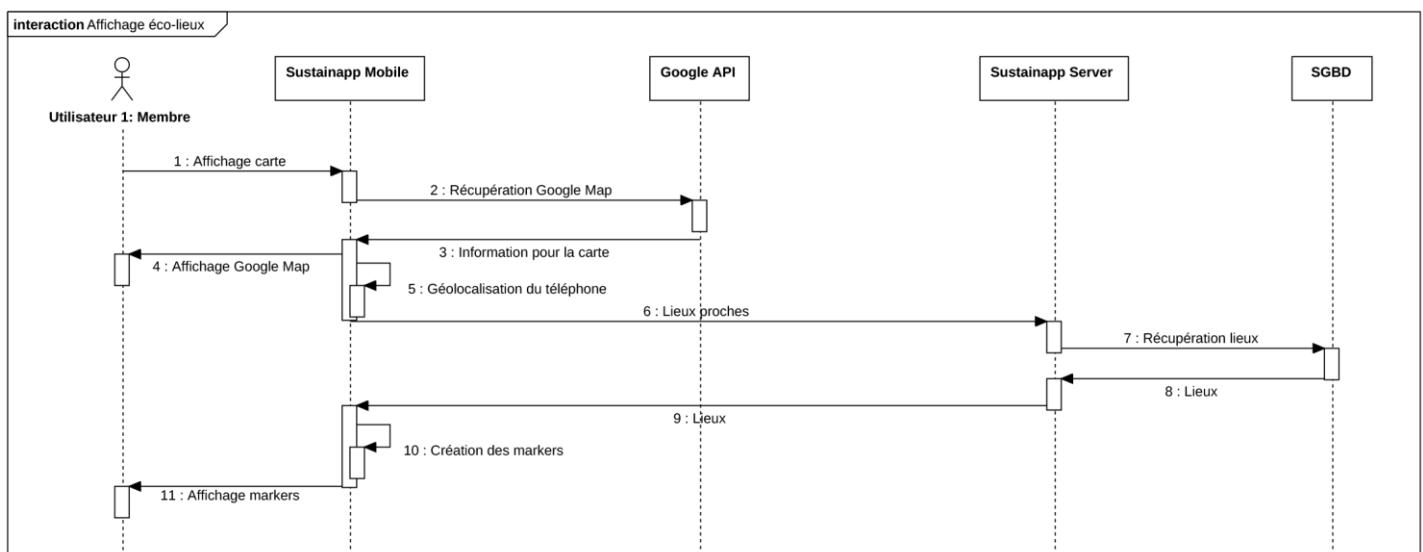


Figure 36: Séquence affichage d'un éco-lieu

2.1.3 Gestion des rôles d'une équipe

La dernière fonctionnalité qui a nécessité des éclaircissements est la gestion des rôles dans une équipe. Cette gestion se fait par le créateur de l'équipe qui peut, lorsqu'il reçoit des demandes de participation, accepter ou refuser cette demande. Cette gestion implique à la fois la réception de notification ou l'obtention d'un badge pour le gérant de l'équipe.

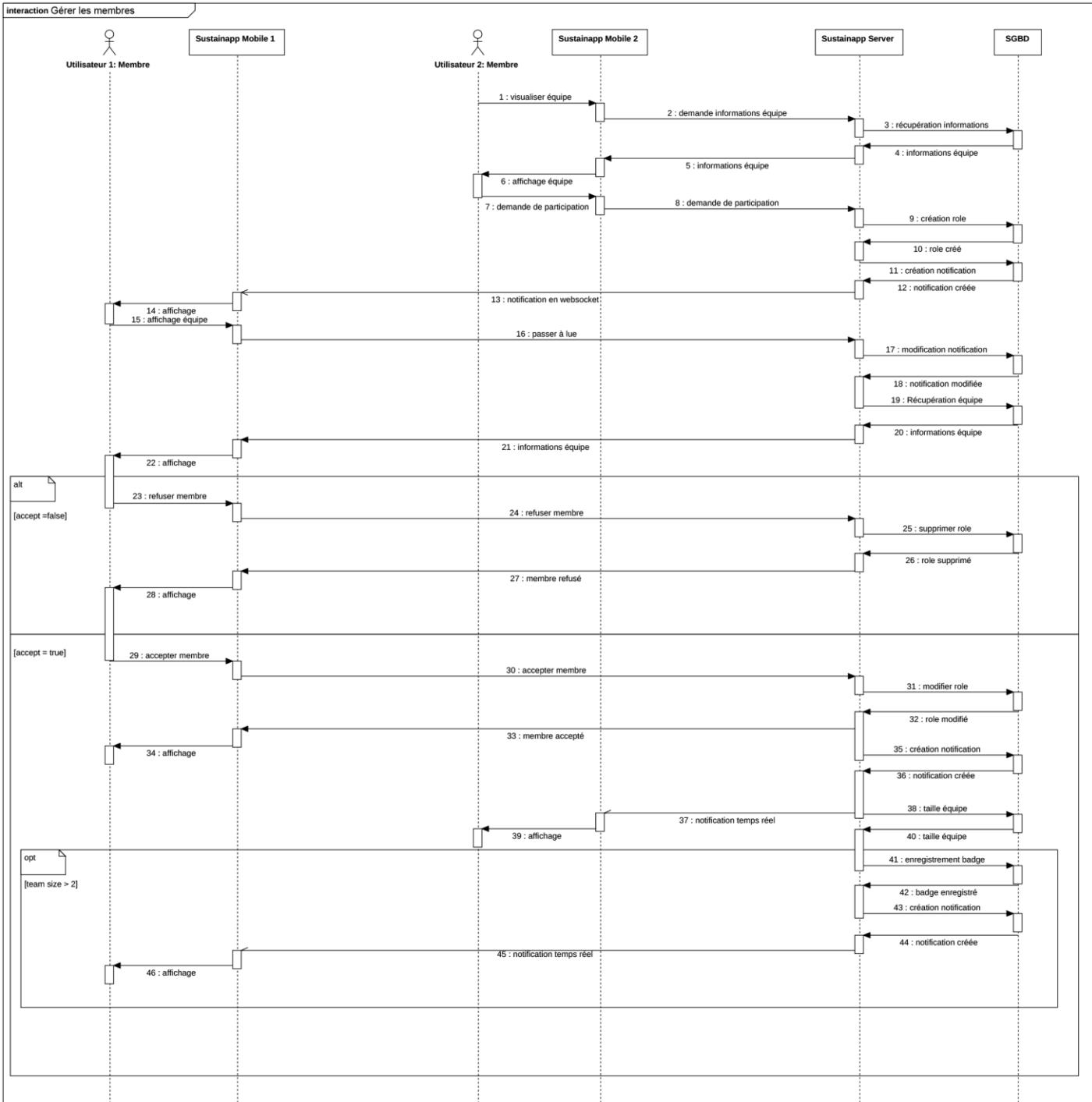


Figure 37: Séquence gestion des rôles d'une équipe

2.2. Modèle de données

Souhaitant réaliser l'application en utilisant le paradigme orienté objet, les données persistantes en base seront représentées à l'aide du diagramme de classe. Le diagramme de classe qui va suivre ne représente donc pas les classes métier mais uniquement les classes de données.

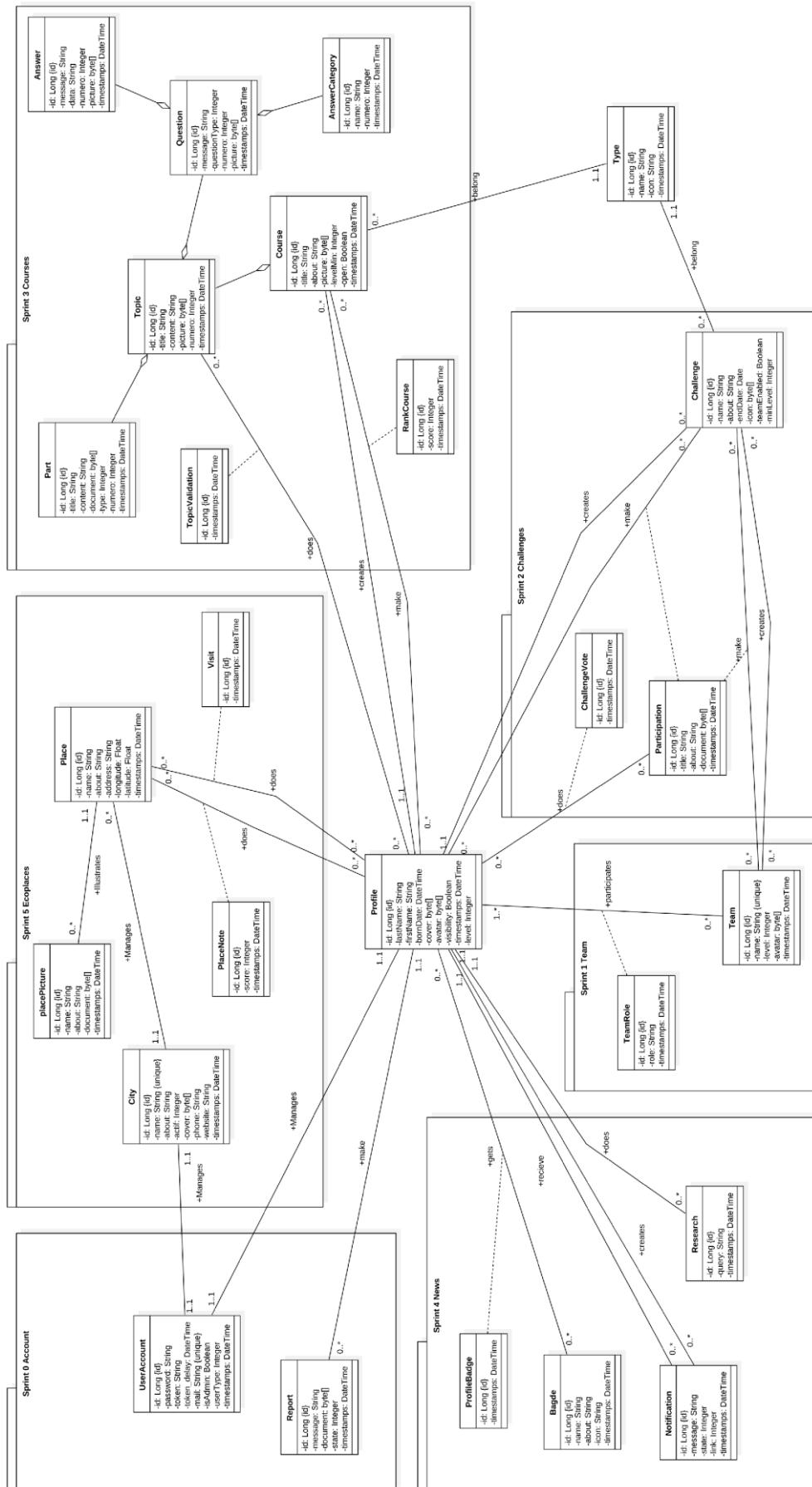


Figure 38: Diagramme de classe de Sustainapp

Tout comme pour les cas d'utilisation, nous avons essayé d'améliorer la lisibilité en regroupant les classes selon les livrables fonctionnels auxquels elles sont reliées et donc aux itérations durant lesquelles il est prévu qu'elles soient réalisées.

Le « Profil » possède dans cette conception un rôle central aussi bien en terme de données que fonctionnel ce qui est logique pour une plateforme de type réseau. Par ailleurs, les données de celui-ci sont distinctes de celles de l'utilisateur (mail, mot de passe) pour des raisons de sécurité mais également pour permettre la possible évolution d'un utilisateur gérant plusieurs profils.

Notons dans cette conception certains détails inhabituels qui impliquent par ailleurs que la base de données relationnelle qui implémentera ce diagramme ne respectera pas les trois formes normales. Mais ces choix ont été évidemment pensés à des fins particulières :

- Le premier détail de ce type est la présence d'id dans les classes d'association dont l'identifiant devrait habituellement être composé de l'identifiant des deux classes auxquelles elle est reliée. Cependant il est prouvé que la vitesse de recherche en base subit particulièrement mal les Ids (clés primaires) composés. Par ailleurs certains framework modernes (tels que « *Django* » en python) ne le permettent même pas et force la présence d'un id simple ;
- Le deuxième détail de ce type est l'utilisation multiple d'une unique classe d'association telle que « *Participation* ». Ceci ne devrait normalement pas être possible de par le fait que la table qui implémentera cette classe est censée retenir la clé étrangère des tables auxquelles elle est liée. Cependant pour éviter la surcharge de tables d'association, certains framework (« *Django* », ou « *Laravel* » en php) permettent de retenir des ids de plusieurs tables différentes (appelés « *id polymorphe* ») via un couple (targetId, targetType) que l'on peut également implémenter soi-même.

2.3. Architecture logicielle et technologies

Souhaitant que l'application soit disponible sur différents supports mobile mais aussi web, nous avons donc choisi de travailler avec une technologie cross-plateforme reconnue : *Ionic (Anular + Apache Cordova)*³³. Ce choix implique également comme nous l'avons vu d'avoir un serveur applicatif distant qui communique avec la base de données. Nous avons décidé de réaliser ce module en Utilisant le framework Jee « *Spring Boot* »³⁴.

Spring Boot et *Ionic* sont deux technologies open sources, gratuites, dont les communautés sont très actives sur internet (on peut donc trouver de nombreuses librairies utilitaires prêtées et de l'aide sur les forums). *Spring Boot* a l'avantage par rapport au *Spring* classique d'être plus léger, de ne pas avoir à effectuer de lourde étape de configuration ou encore d'utiliser un serveur container comme *Tomcat* ou *Glassfish*. Pour ce qui est de la base de données, nous avons décidé d'utiliser « *PostgreSQL* » également gratuite et pourtant plus puissante et sécurisée que *MySQL* ou *SQL Light*.

Voici, figure 39, donc l'architecture finale du projet :

³³ Site web d'*ionic* : <https://ionicframework.com/>

³⁴ Site web *Spring Boot* : <https://projects.spring.io/spring-boot/>

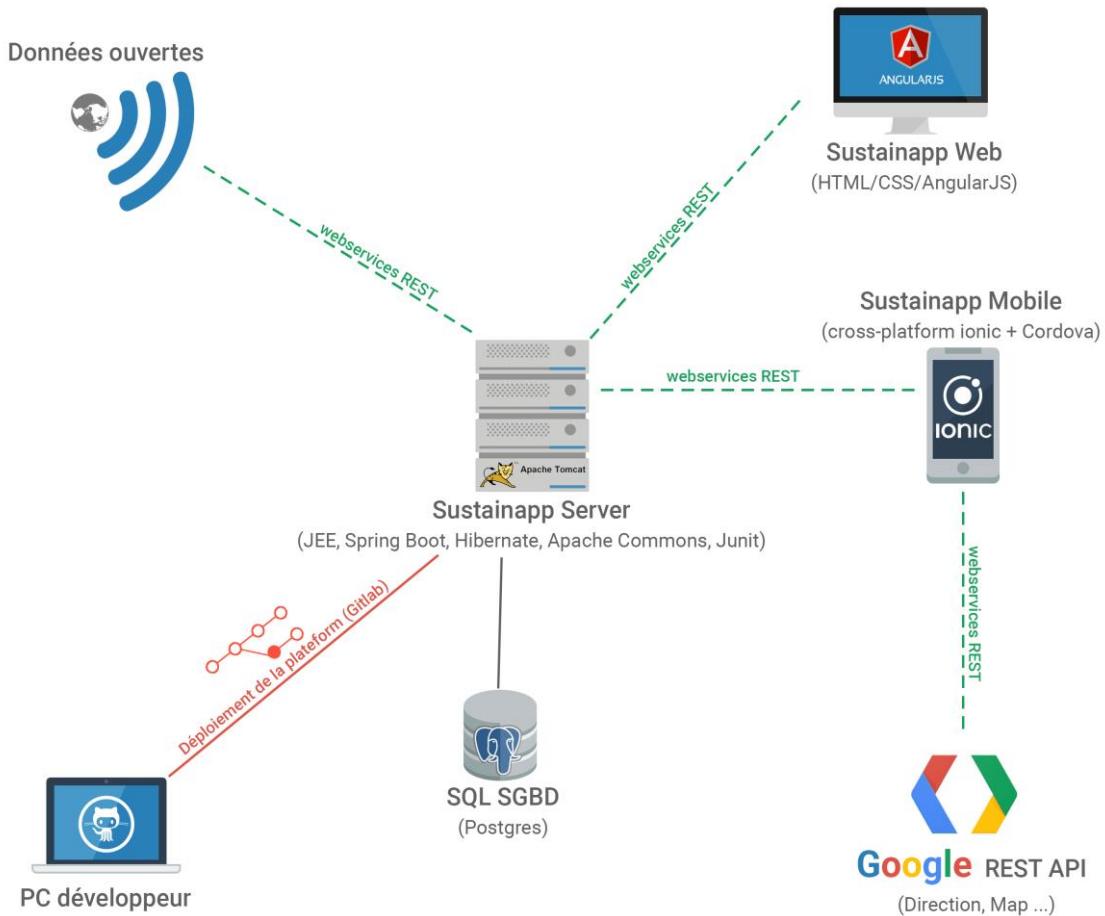


Figure 39: Architecture de Sustainapp

2.3.1. Sustainapp Server

Le premier module « *Sustainapp Server* » sera donc hébergé sur un serveur du CIRRELT. L'architecture interne de ce module sera ainsi, voir figure 40 :

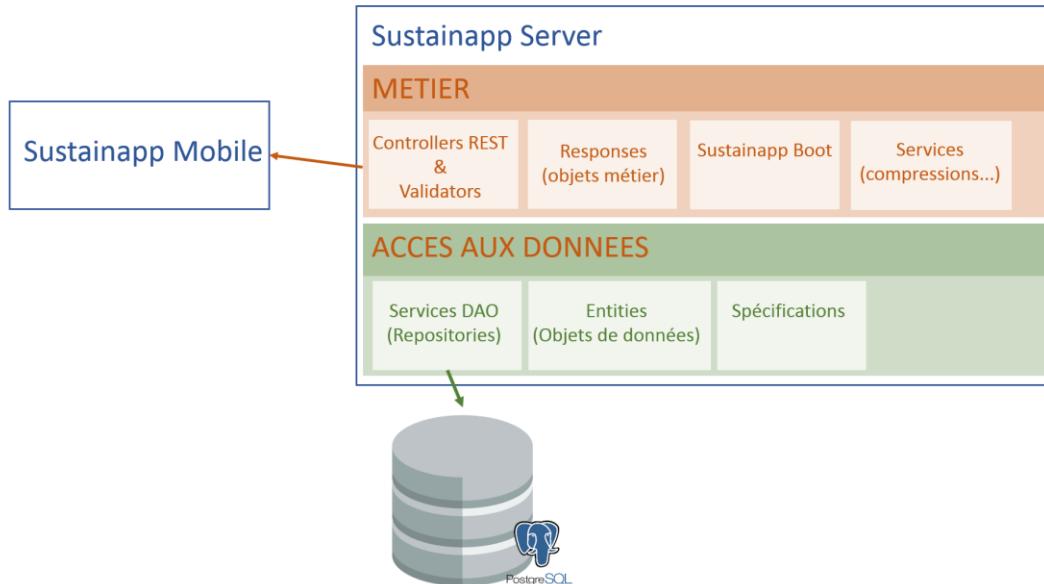


Figure 40: Architecture interne de Sustainapp Server

Comme on peut le voir, ce module ne dispose pas directement d'un tiers « Présentation » mais possède les deux autres tiers :

- **Accès aux données** : ce tiers permet la communication avec la base de données via trois types de classes : les « *entities* » qui sont l'implémentation du diagramme de classe, les « *spécifications* » classes utilisées pour le moteur de recherche, et les « *services DAO* » qui sont en réalité le code d'accès aux données (enregistrement, suppression, modification, récupération) ;
- **Métier** : ce tiers contient la communication avec l'application web/mobile distante via les classes « *controllers REST* » et les objets de réponse « *responses* », le contrôle et la sécurité via les classes de type « *validators* », des classes de traitements métier divers appelés « *services* » et enfin les classes moteurs permettant l'exécution du projet appelées « *boot* ».

2.3.2. Sustainapp Mobile/Web

Les modules « Sustainapp Mobile » et « Sustainapp Web » partagent en réalité le même code. La seule différence entre les deux est que Sustainapp Web exécute directement le code en JavaScript sur le navigateur client alors que Sustainapp Mobile est un code compilé par *Ionic* pour être natif sur sa plateforme d'exécution. Ainsi Sustainapp Web est envoyé par le serveur aux différents navigateurs clients alors que Sustainapp Mobile est présent dans le téléphone client sous forme d'application.

Voici, figure 41, l'architecture interne de ce module :

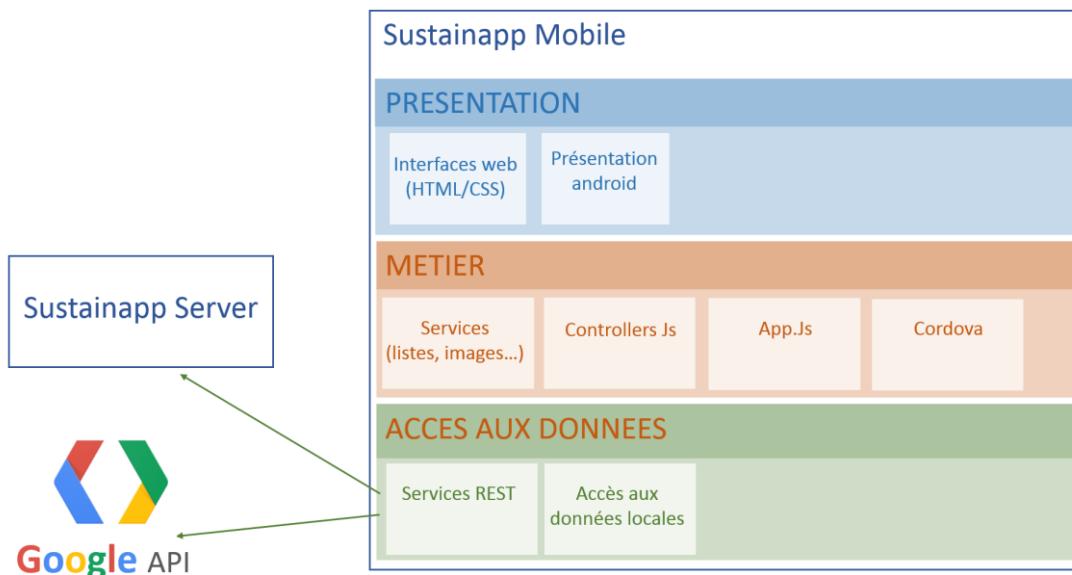


Figure 41: Architecture interne de Sustainapp Mobile/Web

Ce module contient quant à lui les trois tiers :

- **Présentation** : l'affichage des différentes pages en html/css ainsi que certains affichages natifs tels que les notifications *Android/Iphone* ;
- **Métier** : contient des fonctions utilitaires appelées « business services », des fonctions d'appel des capteurs du téléphone utilisant la technologie « *Apache Cordova* », des « controllers » pour réceptionner les actions de l'utilisateur ainsi qu'un fichier « *app.js* » permettant d'exécuter l'application ;
- **Accès aux données** : Sustainapp Mobile/Web accède à deux différents types de données : des données locales telles que les cookies et des données distantes, accessibles en REST, dont les deux principaux fournisseurs sont « *Sustainapp Server* » et les API de *Google*.

III. Conception graphique

Une dernière étape de conception était nécessaire avant de passer à la réalisation de l'application : l'établissement d'une charte graphique qui sera respectée lors du développement des interfaces.

3.1. Couleurs

Tableau 1: Les couleurs de Sustainapp

Couleur	Code hexadécimal (CSS)	Utilisations	Dénotation & raisons
	Vert clair : #2dbe60	Couleur principale utilisée dans les fonds (barre de navigation, menu à gauche), ainsi que pour les titres.	Le vert est la couleur représentative de la nature et de l'écologie.
	Vert foncé : #28a855		
	Bleu : #23d5fb	Couleur secondaire : Fond des titres secondaires, séparations de parties, icônes.	Le bleu est également une couleur utilisée pour représenter l'écologie (eau, ciel) mais c'est surtout une couleur dont le contraste avec le vert reste léger (le vert étant lui-même composé de bleu).
	Gris foncé : #CCCCCC Parfois : #000000	Textes et barre de séparations.	Utilisé pour faciliter la lisibilité et s'adapter aux habitudes graphiques modernes.
	Blanc : #FFFFFF	Fonds principaux des pages	Utilisé pour faciliter la lisibilité et s'adapter aux habitudes graphiques modernes.
	Gris clair : #E0E0E0		

3.2. Polices

Nous n'avons utilisé pour ce projet qu'une seule et unique *font* (police) que nous avons voulu un peu originale pour se distinguer des applications modernes qui utilisent en très grande majorité toutes « *Roboto* » de Google, « *Helvetica* » ou « *Calibri* ». Cependant nous n'avons pas souhaité obtenir un résultat trop extravagant et avons donc choisi une police qui bien qu'inhabituelle reste tout de même assez sobre : « *Sakkal Majalla* ».

« Voici un exemple de texte rédigé avec cette police qui est la fois légèrement oriental et évoque la nature ».

3.3. Logos et icônes

Voici le logo dans sa première version (voir figure 42), avec l'utilisation de la police choisie précédemment. Le symbole est un fruit, une noisette, mais également une ampoule. Nous souhaitions signifier par ce symbole les énergies naturelles et renouvelables ainsi que la protection de cette nature.



Figure 42 : Crédit initiale du logo

Puis nous avons créé à partir de ce logo l'icône de l'application avec de laquelle on accède à celle-ci. Voici les différentes versions de cette icône (à droite la dernière version) :

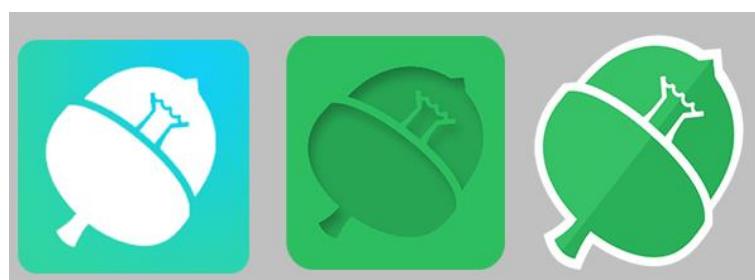


Figure 43: Icônes de Sustainapp

Enfin, nous avons adapté le logo actuel de l'Université Laval pour pouvoir l'intégrer dans certaines pages de l'application :

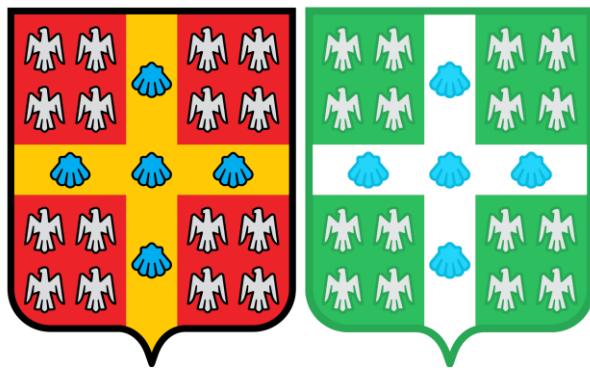


Figure 44: Adaptation du logo de L'université Laval

3.4. Organisation des pages et navigation

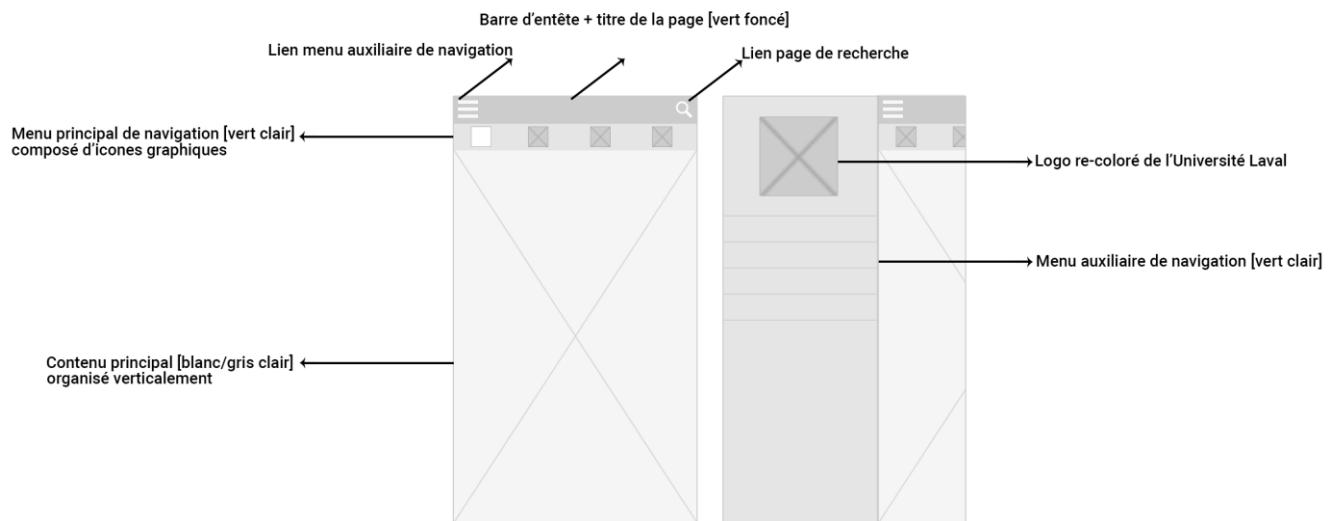


Figure 45: Organisation des pages

L'organisation des pages est assez classique pour une application mobile de type réseau : un menu en haut de page (ainsi qu'un second caché), un contenu organisé verticalement sur un fond blanc, une icône lumineuse et un titre pour indiquer où l'utilisateur se trouve et un outil de recherche accessible sur chaque page.

Nous avons enfin souhaité un schéma de navigation le plus simple possible pour une prise en main rapide de l'application. Cette tâche n'a pas été facile du fait du nombre assez élevé des fonctionnalités et des « écrans » prévus. Ainsi nous avons choisi un modèle de schéma maillé/réseau ou l'on peut passer d'un écran à la presque totalité des autres directement. Pour faire ce passage de chaque écran à l'ensemble des autres, l'utilisateur utilise les deux menus présentés précédemment (auxiliaire ou principal) selon l'importance de la page.

Voici une liste des pages prévues :

Tableau 2: Les pages de Sustainapp

Droit d'accès	numéro	Titre	Description
Utilisateur anonyme	1	Connexion	Contient un formulaire de connexion.
	2	Inscription	Contient deux formulaires d'inscription (profil personnel et gérant de ville) ainsi que les conditions d'utilisation et politique de confidentialité.
Membre connecté	3	Journal	Contient les 3 meilleurs cours et les trois meilleures participations aux challenges en termes de notes et votes.
	4	Profil	Affichage d'un profil personnel et visualisation des badges et cours publiés.
	5	Une ville	Affichage d'une ville avec sa liste de lieux écologiques
	6	Un lieu	Affichage d'un lieu avec notes et images.
	7	Notifications	Affichage de l'historique des notifications reçues
	8	Recherche	Affichage d'un champ de recherche et des résultats
	9	Signaler un abus	Signalement à l'administration d'un problème ou abus
	10	Les challenges	Affichage de tous les challenges triés par dates et filtrés par catégories
	11	Un challenge	Affichage d'un challenge avec toutes les participations
	12	Les cours	Affichage de tous les cours triés par dates et filtrés par catégories ou par langues
	13	Un cours	Affichage d'un cours avec la liste des chapitres

	14	Un chapitre	Affichage d'un chapitre d'un cours
	15	Un quizz	Affichage d'un quizz à valider
	16	Création d'un quizz	Espace de création d'un quizz
	17	La carte	Affichage d'une carte contenant les éco-lieux proches de l'utilisateur
	18	Les équipes	Affichage des toutes les équipes triées par niveaux
	19	Une équipe	Affichage d'une équipe avec la liste des membres de cette équipe et du capitaine
Administrateur	20	Menu Principal	Affichage des différents outils d'administration
	21	Données sur les recherches	Affichage à l'aide de graphiques et de statistiques des données relatives aux recherches
	22	Données sur les cours	Affichage à l'aide de graphiques et de statistiques des données relatives aux cours et quizz
	23	Données sur les profils et les équipes	Affichage à l'aide de graphiques et de statistiques des données relatives aux profils et équipes
	24	Données sur les challenges	Affichage à l'aide de graphiques et de statistiques des données relatives aux challenges et participations
	25	Données sur les villes	Affichage à l'aide de graphiques et de statistiques des données relatives aux villes et lieux
	26	Validation des villes	Espace de validation et refus des demandes d'inscription de villes
	27	Traitements des signalements	Espace de visualisation et traitement des signalements



Réalisation du projet

I. Mise en place du projet

1.1. Planification générale de la réalisation

Nous avons commencé à réaliser l'application telle que pensée dans l'étape de conception. Pour cela, nous avons commencé par évaluer la durée de chaque user story nécessaire à l'accomplissement des cas d'utilisation et les avons regroupés en releases chacune équivalent à un package de fonctionnalités et à un et unique Sprint.

La première itération fut plus centrée sur la mise en place du projet (et donc appelée « *Sprint 0* »), des dossiers et du code générique qui sera utilisé à de nombreuses reprises ainsi que sur le développement des fonctionnalités de l'application. Ainsi, les user stories qui composent cette itération ne peuvent pas toutes être exprimées sous la forme « *En tant que, Je veux* »³⁵ comme le seront celles des itérations suivantes. Pour cette raison elles sont différencieres dans le tableau 3 des autres *user stories*.

Tableau 3: User stories de Sustainapp

Numéro	Description de la user story	Complexité temporelle
01	<i>Mise en place de l'environnement de développement et création de l'architecture initiale</i>	40h
02	<i>Création d'une liste de règles de codage permettant de respecter notre « plan d'assurance qualité ».</i>	8h
03	<i>Intégration de code utilitaire</i>	32h
04	En tant que membre je veux accéder à un compte personnel	32h
05	En tant que membre je veux signaler les abus à l'administration	8h
06	En tant que membre, je veux gérer un profil personnel	36h
07	En tant que membre, je veux gérer des équipes	56h
08	En tant que membre, je veux participer dans des équipes	28h
09	En tant que membre, je veux gérer un challenge	72h
10	En tant que membre, je veux participer dans un challenge	36h
11	En tant que membre, je veux voter pour la meilleure réalisation d'un challenge	12h

³⁵ Dans la méthodologie SCRUM, les user stories sont généralement rédigée sous la forme « En tant que [rôle], je veux [fonctionnalité] afin de [raison d'être de la fonctionnalité] ». Mais cette règle ne s'applique qu'à partir du 1^{er} sprint et pas du sprint dit « zéro ».

12	En tant que membre, je veux gérer un cours	63h
13	En tant que membre, je veux suivre un cours	19h
14	En tant que membre, je veux gérer un quizz	38h
15	En tant que membre, je veux effectuer des recherches	13h
16	En tant que membre, je veux recevoir des notifications sur les actualités	27h
17	En tant que membre, je veux obtenir des badges et niveaux	53h
18	En tant que membre, je veux avoir accès au « <i>leaderboard</i> » ³⁶	27h
19	En tant que gestionnaire de ville, je veux gérer une ville	65h
20	En tant que membre, je veux visiter des éco-lieux	55h
21	En tant qu'administrateur, je veux avoir accès aux données des activités de Sustainapp	120h

Tableau 4: Plan des releases

Numéro	Libellé	User stories concernées	Début prévu	Fin prévue
00	<i>Mise en place du projet</i>	01, 02, 03, 04, 05	1 ^{er} mars 2017	15 mars 2017
01	Un réseau d'écoresponsables	06, 07, 08	16 avril 2017	31 mars 2017
02	Les challenges en écologie	09, 10, 11	1 ^{er} avril 2017	15 avril 2017
03	Formations gamifiées	12, 13, 14	16 avril 2017	30 avril 2017
04	Les actualités de Sustainapp	15, 16, 17 ,18	1 ^{er} mai 2017	15 mai 2017
05	Les villes écologiques	19, 20	16 mai 2017	31 mai 2017
06	Administration de Sustainapp	21	1 ^{er} juin 2017	15 juin 2017

³⁶ Leaderboard : Journal dans lequel on trouve un podium des meilleurs cours ou réalisations.

1.2. Planification de l'itération initiale

Pour la planification nous avons décidé de compter qu'une journée serait en moyenne de 8 heures de développement et donc une demi-journée de 4 heures. Cette première itération ainsi que toutes les suivantes auront donc une durée de deux semaines et se conclura par une réunion mixte de présentation du livrable et de planification de l'itération suivante. Ainsi l'ensemble des US³⁷ suivantes devaient être planifiées de manière à être réalisées entre le 1^{er} et le 15 mars (120h) :

Tableau 5: Découpage en tâches du sprint 0

Numéro US :	01	02	03	04	05
Priorité :	01	02	03	04	05
01	Mise en place du GIT	Documentation règles de codage	Intégration de librairies génériques	Inscription à Sustainapp	Signaler un abus
02	Mise en place Batch		Module d'upload de fichiers	Connexion à Sustainapp	
03	Mise en place Server				
04	Mise en place Mobile				

1.3. Mise en place du projet

La première étape de la réalisation concrète du projet a été de créer un « *repository GIT* » sur notre serveur *Gitlab* afin de pouvoir sauvegarder des versions du code de la machine de développement vers le serveur de production. « *Gitlab* » permet de créer le « *repository* » de versions et offre également une visibilité sur le code présent dans le repository (résultats des exécutions de tests, langages présents dans le code, fréquence de commit fait par chaque développeur ...)

³⁷ US : user stor[ies/y]

Sustainapp, Rapport de projet de fin d'études 2017

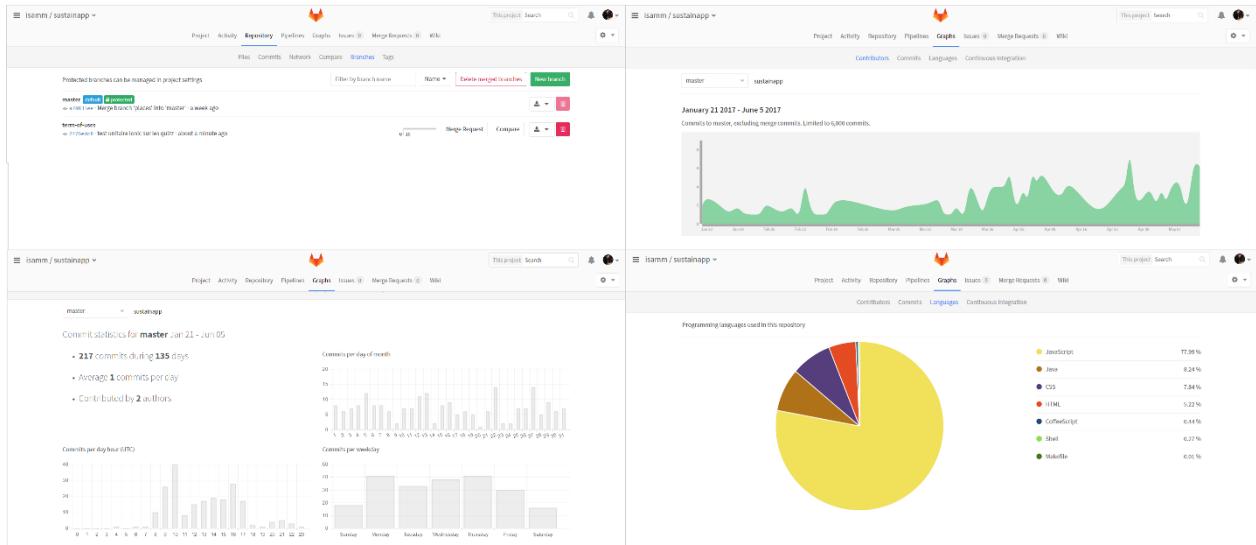


Figure 46: Interface de Gitlab

Enfin, on installe sur les postes des développeurs (un seul dans ce cas) un shell GIT permettant d'exécuter les commandes de synchronisation et de création de version. On peut également pour faciliter cette étape installer en plus une interface telle que «*Github Desktop*» qui permet d'exécuter les commandes en un simple clic.

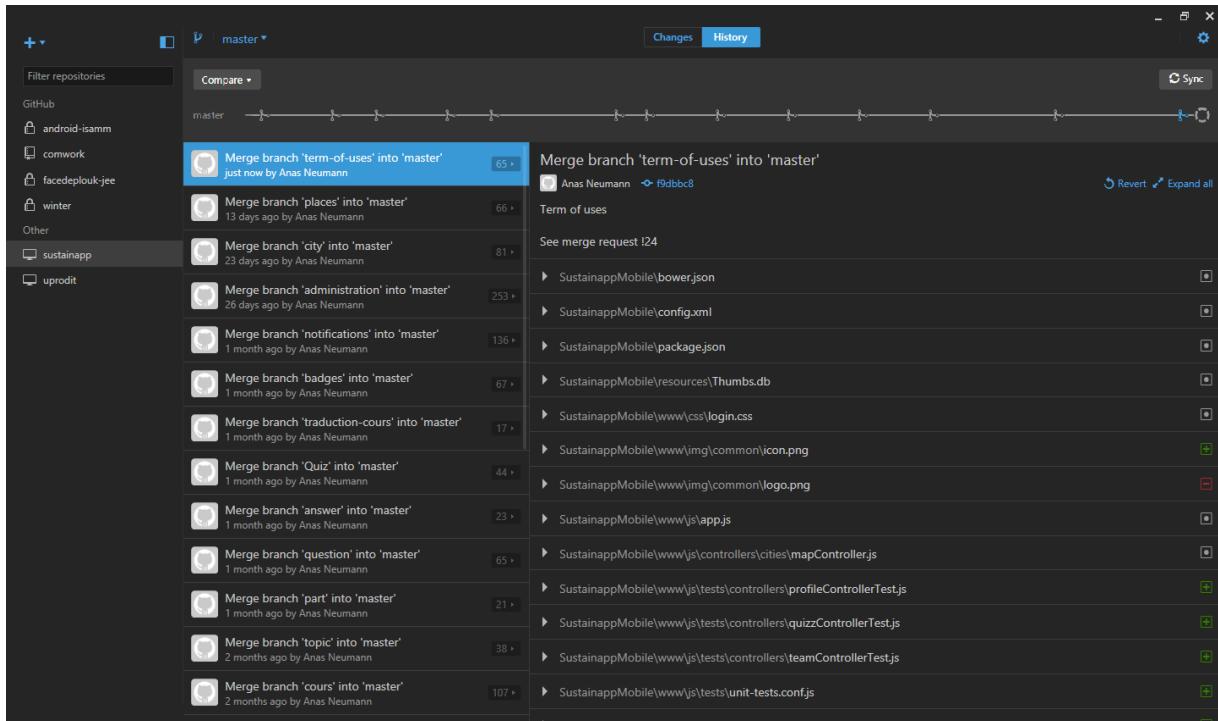


Figure 47 : Interface de Github Desktop

Dans ce *repository* nous avons créé trois dossiers et un fichier :

Name	Last Commit >	History	Last Update
📁 SustainappBatch	Merge branch 'test-server' into 'master'		2 weeks ago
📁 SustainappMobile	delete profile and user part 4		a day ago
📁 SustainappServer	delete profile part 3		a day ago
📄 .gitignore	Fix: bad files		3 weeks ago

Figure 48: Contenu du repository GIT

Le fichier *.gitignore* est un fichier permettant à l'aide d'expressions régulières de rendre local uniquement certains fichiers (particulièrement les fichiers temporaires et les fichiers générés par les IDE tels qu'Eclipse qui n'ont donc pas d'utilité pour le projet et aucun lien avec son code). Les dossiers *SustainappMobile* et *SustainappServer* sont les deux applications décrites dans la conception. Enfin, le dossier *SustainappBatch* n'est pas une application exécutable mais permet le déploiement des deux applications et la gestion de la base de données.

1.4. Sustainapp Server

Comme nous l'avons vu, Sustainapp Server ne sera pas une application destinée à être utilisée directement par l'utilisateur dans le sens où elle ne fournit pas d'interfaces graphiques. Sustainapp Server est destinée à recevoir des demandes d'action en base, généralement *CRUD*³⁸, puis effectuer des vérifications des droits d'action et finalement répondre au format *JSON*³⁹ à l'application qui a demandé l'opération.

³⁸ **CRUD** (*Create, Retrieve, Update, Delete*) : les opérations héritées du protocole http d'ajout, modification, suppression et récupération d'une donnée (avec différentes récupérations : toutes, une seule ou recherche par critères).

³⁹ **JSON** : est l'objet du langage **JavaScript** et peut être vu comme une liste de clé/valeur dans laquelle la clé est de type textuel et la valeur peut être textuelle, tableau ou encore un **JSON** imbriqué.

1.4.1. Intégration de Spring Boot

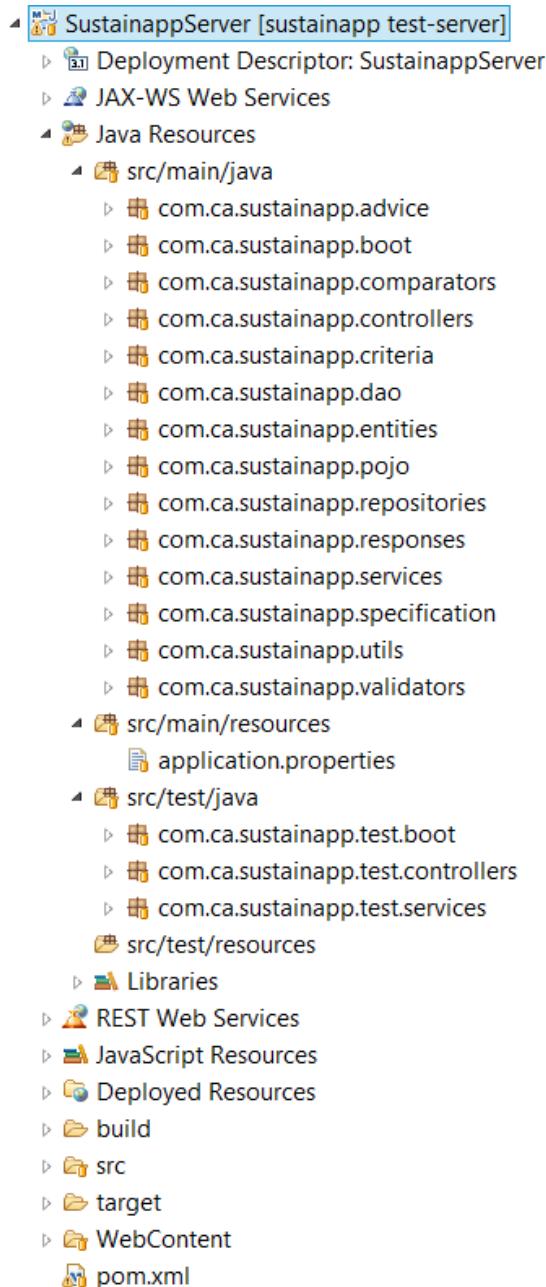


Figure 49: Packages de SustainappServer

Nous avons créé 4 dossiers sources dans ce projet : « *src/main/java* » qui contient le code fonctionnel de l'application, « *src/main/resources* » qui contient uniquement les fichiers de configuration du projet et « *src/test/java* » ainsi que « *src/test/resources* » pour les tests unitaires et d'intégration qui seront présentés plus tard.

Nous avons choisi pour l'intégration de librairies telles que « *Apache Commons* » ou de Framework et principalement « *Spring Boot* » d'utiliser la technologie « *Maven* »⁴⁰ qui permet de gérer

⁴⁰ Site web officiel de Maven : <https://maven.apache.org/>

donc toutes les dépendances nécessaires en les précisant dans le fichier « *pom.xml* »⁴¹ plutôt que de chercher à télécharger les fichiers .jar ou .war et de les intégrer manuellement ce qui est assez long et peut causer des problèmes d'interdépendance.

A cette étape nous avons un projet JEE/Maven qui pourrait « *tourner* » (être exécuté) sur un serveur conteneur tel que « *Apache Tomcat* », mais comme nous l'avons vu cela est inutile avec l'utilisation de Spring Boot qui permet d'exécuter l'application comme une application Java (à l'aide des commandes de la JVM « *java –jar [sustainappServer]* ») ce qui est plus léger (en terme d'utilisation de la mémoire) et qui facilite en plus le travail du développeur lors de l'étape du déploiement.

Pour cela il suffit de créer un fichier de configuration très simple appelé « *application.properties* » dans lequel on spécifie la base de données utilisé (dans notre cas *Postgres*), l'utilisation d'« *Hibernate* »⁴² pour le mapping de des objets de données, le port d'exploitation ou encore la configuration des fichiers acceptés à l'upload (le type de fichier que l'utilisateur peut uploader). Ensuite il faut créer une classe de démarrage qui est pratiquement identique pour tous projets Spring boot que l'on a tout simplement appelé ici « *SustainappServer.java* »

```
@SpringBootApplication
@ComponentScan(basePackages={"com.ca.sustainapp"})
@EntityScan(basePackages = {"com.ca.sustainapp.entities"})
@EnableJpaRepositories(basePackages = {"com.ca.sustainapp.repositories"})
public class SustainappServer {

    /**
     * Methode principale de démarrage de la plateforme
     * @param args
     */
    public static void main(String[] args) {
        SpringApplication.run(SustainappServer.class, args);
    }
}
```

Figure 50: *SustainappServer.java*

Cette classe spécifie les packages à scanner par Spring pour chercher les fichiers singleton à injecter⁴³, les *repositories*⁴⁴ de données Spring ou encore les classes de données *Hibernate* dites « *entities* ».

⁴¹ Repository des dépendances Maven : <https://mvnrepository.com/>

⁴² Hibernate : Librairie JEE permettant de créer une couche d'abstraction entre la base de données SQL et le code Java pour faciliter la manipulation des données par le développeur. Site web officiel : <http://hibernate.org/>

⁴³ Injection de Dépendance : un principe tiré de l'inversion de contrôle dans lequel on choisit de ne pas instancier toute classe dite sans état (identiquement utilisé en tout temps et par tout utilisateur) mais plutôt de l'injecter directement partout où elle sera utile via une annotation *@Inject* selon la norme Java 8 ou *@Autowired* selon le Framework *Spring*.

⁴⁴ Spring Repositories : classe contenant des méthodes créées par Spring permettant d'interagir avec une base de données.

1.4.2. Librairies utilitaires

Afin d'accélérer la vitesse de développement, nous avons choisi d'intégrer des classes utilitaires que j'ai développé et enrichi au cours des différents projets que j'ai eu à réaliser durant mes études ou expériences professionnelles. Ces classes ne contiennent pas de codes spécifiques à un projet mais des méthodes statiques qui sont utiles dans un grands nombre de projets : transformer un objet en String Json pour l'envoyer, des traitements divers sur des listes d'objets, changement de type d'objet de manière à ne pas générer d'exception, traitement sur les dates ...

1.4.3. Classes de données et services d'accès

Parmi les règles de codage décidées en début de projet, une concerne les classes de données : toutes les classes de données héritent d'une classe commune appelée « *GenericEntity* » cela permet d'être sûr qu'elles possèdent toutes les attributs « *id* » et « *timestamps* » (date de création de l'objet) et ainsi trier ces objets selon cette date. Cela permet aussi d'utiliser des listes d'« *entities* » différentes. Certaines « *entities* » héritent d'une seconde classe, fille de la première : « *GenericNumerotableEntity* » qui possède en plus un numéro de position pour une seconde forme de tri (par exemple pour les chapitres d'un cours).

```
@Entity
@Table(name = "USER_ACCOUNT")
@SequenceGenerator(name = "user_account_id_seq_generator", sequenceName = "user_account_id_seq")
public class UserAccountEntity extends GenericEntity implements Serializable {
    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy = GenerationType.SEQUENCE, generator = "user_account_id_seq_generator")
    @Basic(optional = false)
    @Column(name = "ID")
    private Long id;

    @Column(name = "MAIL")
    private String mail;

    @Column(name = "PASSWORD")
    private String password;

    @Column(name = "IS_ADMIN")
    private Boolean isAdmin;

    @Column(name = "TOKEN")
    private String token;

    @Column(name = "USER_TYPE")
    private Integer type;

    @Column(name = "TOKEN_DELAY")
    private Calendar tokenDelay;
```

Figure 51: Exemple d'Entity

Nous avons ainsi choisi d'utiliser les annotations d'« *Hibernate* » et des IDs générés directement par la base de données *PostgreSQL* pour éviter cette gestion. Nous avons également choisi dans les règles de codage et pour des raisons de vitesse d'exécution d'éviter les jointures générées par *Hibernate* en n'utilisant que très peu les annotations telles que « *OneToMany* », « *ManyToOne* » ou « *ManyToMany* » et donc de choisir d'appeler les classes liées uniquement quand on en a besoin.

L'interaction entre ces classes de données et la base se fait via des services « *DAO*⁴⁵ » utilisant les *repositories* de *Spring*. Ces services sont évidemment des singlenton injectés par la suite dans les *Controllers*.

```
@Service("profileService")
public class ProfileServiceDAO extends GenericServiceDAO {

    /**
     * Le repository
     */
    @Autowired
    ProfileRepository repository;

    /**
     * Accès un seul entity par son Id
     * @param id
     * @return
     */
    public ProfileEntity getById(Long id){
        if(null == id){
            return null;
        }
        return repository.findOne(id);
    }

    /**
     * Create a new entity
     * @param entity
     * @return
     */
    @Modifying
    @Transactional
    public Long createOrUpdate(ProfileEntity entity){
        return repository.saveAndFlush(entity).getId();
    }
}
```

Figure 52 : Exemple de service DAO

1.4.4. Controllers et Validators

La partie la plus conséquente de cette application serveur est les controllers : ils sont déclenchés automatiquement par *Spring* selon l'URL appelée par l'utilisateur et utilisent les services DAO pour traiter les données avant de renvoyer une réponse au format JSON.

Tout comme pour les services DAO et les entities, les controllers héritent tous d'un controller générique comportant des méthodes qu'ils utilisent tous telles que :

- vérification si l'utilisateur est connecté, si il est administrateur ou simple utilisateur
- envoi d'une réponse avec des codes spécifiques : succès, échec et ajout de données de réponse ou autre.

⁴⁵ DAO (*Data Access Object*) : Object d'accès aux données

```
/*
 * delete a participation
 * @return
 */
@ResponseBody
@RequestMapping(value="/participation/delete", method = RequestMethod.POST, produces = SustainappConstantes.MIME_JSON)
public ResponseEntity<String> delete(HttpServletRequest request) {
    Optional<Long> idParticipation = StringsUtils.parseLongQuickly(request.getParameter("participation"));
    if(!isConnected(request) || !idParticipation.isPresent()){
        return super.refuse();
    }
    ParticipationEntity participation = participationService.getById(idParticipation.get());
    if(null == participation || (!super.isAdmin(request) && !super.isOwnerParticiaption(participation, super.getUser(request).getProfile()))){
        return super.refuse();
    }
    deleteService.cascadeDelete(participation);
    return success();
}
```

Figure 53: Exemple de controller

Cela en va de même pour les classes de validation des formulaires, appelées par le *controller* avant d'effectuer des traitements en base. Ces classes héritent tous d'un *validator* générique qui décrit le format de réponse des erreurs détectées. Enfin tout comme les services, ces validators sont des singletons injectés.

```
@Component
public class SigninValidator extends GenericValidator {

    /**
     * Service d'accès à la base de données
     */
    @Autowired
    private UserAccountServiceDAO service;

    /**
     * Variables
     */
    private Pattern pattern;
    private Matcher matcher;
    private static final String EMAIL_PATTERN =
        "^[_A-Za-z0-9-\\+]+(\\.[_A-Za-z0-9-]+)*@"
        + "[A-Za-z0-9-]+(\\.[A-Za-z0-9-]+)*(\\.[A-Za-z]{2,})$";

    /**
     * {@inheritDoc}
     */
    @Override
    public Map<String, String> validate(HttpServletRequest request){
        Map<String, String> result = super.validate(request);
        if(isEmpty(request.getParameter("mail"))){
            result.put("mail", "form.mail.mandatory");
        } else if(!isValidMail(request.getParameter("mail"))){
            result.put("mail", "form.mail.format");
        } else if(existMail(request.getParameter("mail"))){
            result.put("mail", "form.mail.exist");
        }
        if(isEmpty(request.getParameter("password")) || request.getParameter("password").length() < 4){
            result.put("password", "form.password.mandatory");
        }
        if(isEmpty(request.getParameter("firstName"))){
            result.put("firstName", "form.firstName.mandatory");
        }
        if(isEmpty(request.getParameter("lastName"))){
            result.put("lastName", "form.lastName.mandatory");
        }
        Optional<Integer> type = StringsUtils.parseIntegerQuietly(request.getParameter("type"));
```

Figure 54: Exemple de validator

1.4.5. Objets réponses

Enfin, la dernière étape a été la création d'une classe générique de réponse qui pareille aux autres types de classes, sera héritée par toutes les réponses des controllers. Cette classe mère comporte une information sur sa date de création, un code de succès ou d'échec, une liste d'erreurs (remplie par les validators) et une méthode qui transforme toutes ces données en JSON « *buildJson ()* ».

```
public class HttpRESTfullResponse implements Serializable {  
  
    /**  
     * Attributes  
     */  
    protected static final long serialVersionUID = 1L;  
    protected Calendar buildDate;  
    protected Integer code;  
    protected Map<String, String> errors = new HashMap<String, String>();
```

Figure 55: Classe générique de réponse d'un controller

1.5. Sustainapp Mobile

Cette deuxième application par contre, est celle qui sera utilisée par l'utilisateur (en web ou compilée pour mobile). Elle possède donc des interfaces graphiques et c'est elle principalement qui appellera les URLs de Sustainapp Server.

1.5.1. Création d'une application Ionic/Cordova

Après avoir installé *Ionic*, on utilise directement la commande « *ionic start [nom_projet]* » pour que celui-ci génère automatiquement le projet avec l'architecture suivante :

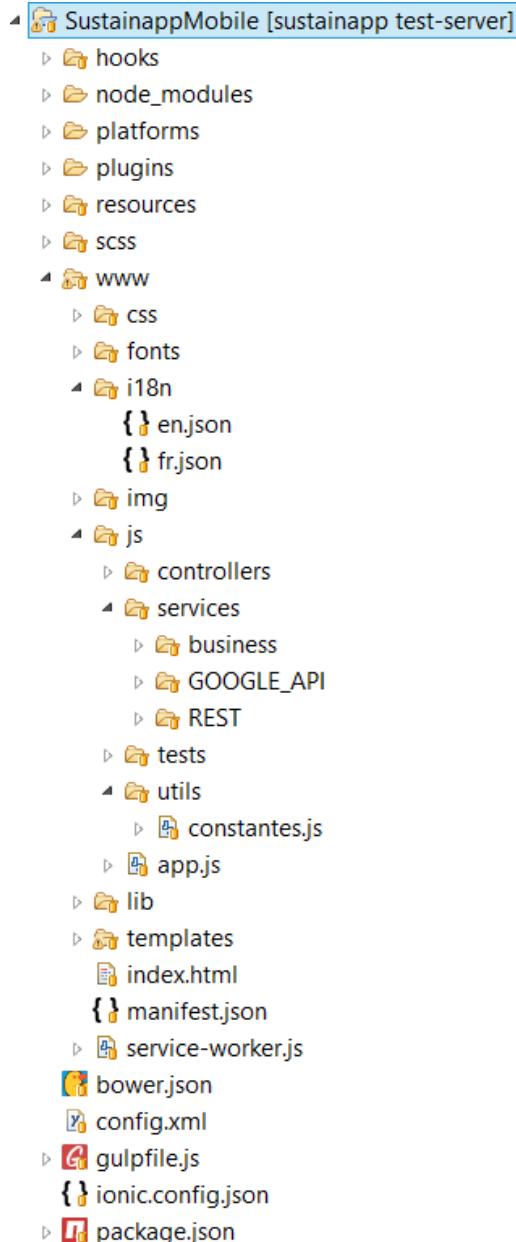


Figure 56: Architecture de Sustainapp Mobile

Tout comme JEE possède *Maven*, en *JavaScript* on gère les dépendances de librairies ou framework avec *gulp*, *bower* ou encore *npm* qui possèdent des fichiers similaires au pom.xml comme bower.json. Les librairies demandée sont téléchargées automatiquement dans le dossier « *lib* » (si ce

sont des librairies JavaScript) ou « *plugins* » (si ce sont des plugins pour *Cordova* et l'accès à un matériel/capteur particulier).

Les dossiers « *platforms* » et « *resources* » contiennent les informations nécessaires à la compilation pour une plateforme particulière (ios, Android, Web browser...). C'est également dans le dossier « *platforms* » que seront générés les exécutables finaux une fois compilés (*ionic* possède pour cela des commandes de génération telles que « *ionic build android –prod –release* »)

Tout le travail du développeur se situe donc dans le dossier www (dossier sur lequel devra pointer un server http tel que Nginx ou Varnish si l'on souhaite utiliser le projet non compilé comme site web). Ce dossier contient plusieurs sous-dossiers : « *fonts* » (les fichiers de police généralement au format TTF), « *css* » (les feuilles de styles pour l'interface qui peuvent également être en *scss*, *less* ou *sass*), « *img* » (les icônes et autres images statiques), « *i18n* » (des fichiers de traduction au format Json, un par langue souhaitée), « *lib* » (les librairies demandées via bower ou npm) « *templates* » (les interfaces au format HTML avec ajout de data-binding en utilisant angularJs), *index.html* (l'interface principale dans laquelle est importée tous les fichiers *JavaScript* et *Css* nécessaires) et enfin « *Js* » (les fichiers de développement en *JavaScript*).

Voici un exemple d'installation d'une librairie dans le projet :

```
C:\Users\Anas\git\sustainapp\SustainappMobile>bower install --save ng-stomp
bower cached          https://github.com/driftyco/ionic-bower.git#1.3.2
bower validate        1.3.2 against https://github.com/driftyco/ionic-bower.git#1.
3.2
bower not-cached
bower resolve
bower download        https://github.com/beevvelop/angular-stomp.git#*
.gz
bower extract          ng-stomp#* archive.tar.gz
bower resolved
bower not-cached      https://github.com/jmesnil/stomp-websocket.git#2.3.4
bower resolve
bower not-cached      https://github.com/sockjs/sockjs-client.git#1.1.1
bower resolve
bower download        https://github.com/jmesnil/stomp-websocket/archive/2.3.4.tar
.gz
bower download        https://github.com/sockjs/sockjs-client/archive/v1.1.1.tar.g
z
bower extract          stomp-websocket#2.3.4 archive.tar.gz
bower invalid-meta    for:C:\Users\Anas\AppData\Local\Temp\PC-ANAS-Anas\bower\f051
c257d590f17030882fbf5f2e80-9676-HMbXZh\bower.json
bower invalid-meta    The "main" field cannot contain minified files
bower resolved
bower extract          https://github.com/jmesnil/stomp-websocket.git#2.3.4
bower invalid-meta    for:C:\Users\Anas\AppData\Local\Temp\PC-ANAS-Anas\bower\3fbf
c6986c81c592d5d6da484a311b1b-9676-6AxJz1\bower.json
bower invalid-meta    The "description" is too long, the limit is 140 characters
bower resolved
bower install          https://github.com/sockjs/sockjs-client.git#1.1.1
bower install          ionic#1.3.2
bower skipped          ionic was not installed because there is already a non-bower
directory with that name in the components directory (www\lib\ionic). You can f
orce installation with --force.
bower install          ng-stomp#0.4.0
bower install          stomp-websocket#2.3.4
bower install          sockjs-client#1.1.1

ionic#1.3.2 ...\\AppData\\Local\\bower\\cache\\packages\\75785deacc09255f971c3354
2b04c50a\\1.3.2
└── angular#1.5.3
   ├── angular-animate#1.5.3
   ├── angular-sanitize#1.5.3
   └── angular-ui-router#0.2.13

ng-stomp#0.4.0 www\\lib\\ng-stomp
└── sockjs-client#1.1.1
   └── stomp-websocket#2.3.4

stomp-websocket#2.3.4 www\\lib\\stomp-websocket

sockjs-client#1.1.1 www\\lib\\sockjs-client
```

Figure 57: Installation d'une librairie via Bower

1.5.2. Règles de développement et fichiers JavaScript

Tout comme pour Sustainapp Server, nous avons respecté certaines règles de codage et de nommage. Par exemple toute classe css se trouvant dans un fichier nommé « `[nomFichier].css` » sera appelée dans les fichiers html « `s-[nomFichier]-[nomClasse]` » avec nomClasse lié à l'utilité directe de la classe (titre, menu, icone ...).

Ensuite comme pour toute application utilisant AngularJs, le cœur de l'application se trouve dans le fichier app.js : ce fichier permet :

- d'instancier les modules (controller, service constante) qui seront développés dans d'autre fichier ;
- de configurer la traduction automatique de l'application en appelant les fichiers i18n (pour cela on installe préalablement une librairie d'i18n via npm ou bower) ;

```
/**  
 * SYSTEME DE TRADUCTION  
 */  
$translateProvider.useStaticFilesLoader({prefix: 'i18n/', suffix: '.json'});  
$translateProvider.useSanitizeValueStrategy('sanitizeParameters');  
$translateProvider.registerAvailableLanguageKeys(['en','fr'], {'en_US': 'en', 'en_UK': 'en', 'fr_FR': 'fr', 'fr_BE': 'fr'})  
.determinePreferredLanguage();  
$translateProvider.use();  
});
```

Figure 58: Configuration de l'i18n

- de configurer la sécurité des liens (liens interdits ou autorisés) ;

```
/**  
 * Autoriser les urls provenant de ces sites web  
 */  
$sceDelegateProvider.resourceUrlWhitelist([  
    'self',  
    'https://www.youtube.com/embed/**',  
    'https://maps.google.com/**',  
    "https://www.cirrelt.ca/**",  
    "https://www.ulaval.ca/**",  
    "http://legisquebec.gouv.qc.ca/**",  
    "https://www.uprodit.com/**",  
    "https://play.google.com/**",  
    "http://sustainapp.cirrelt.ca/**",  
    "https://sustainapp.cirrelt.ca/**",  
    "http://localhost:8100/**",  
    "http://127.0.0.1:8100/**"  
]);  
  
/**  
 * Blacklister les urls provenants de ces sites web  
 */  
$sceDelegateProvider.resourceUrlBlacklist([  
]);
```

Figure 59: Sécurité des liens

- Ou encore de configurer le routing via des triplets url/template/controller.

```
.state('tab.administration-research', {
  url: '/administration/research',
  views: {
    'tab-news': {
      templateUrl: 'templates/administration/research.html',
      controller: 'researchDataController'
    }
  }
})
.state('tab.administration-profiles', {
  url: '/administration/profiles',
  views: {
    'tab-news': {
      templateUrl: 'templates/administration/profiles.html',
      controller: 'profilesDataController'
    }
  }
})
.state('tab.administration-admins', {
  url: '/administration/admins',
  views: {
    'tab-news': {
      templateUrl: 'templates/administration/admins.html',
      controller: 'adminsController'
    }
  }
})
```

Figure 60: Système de routing

Durant la réalisation des fonctionnalités il ne faudra plus que créer des fichiers css/templates et controllers/services JavaScript au niveau de SustainappMobile.

Il existe enfin trois types de services dans cette application : des services « REST » permettant d'appeler des controllers REST de Sustainapp Server, des services « GOOGLE API » permettant d'appeler les controllers REST de Google (pour utiliser leurs API de map, direction, geolocation...).

1.6. Sustainapp Batch

Le dernier module à mettre en place avant de commencer les fonctionnalités est Sustainapp Batch. Ce module n'a pas été évoqué durant la conception car en réalité celui-ci n'a pas eu à être conçu entièrement pour ce projet, en effet il est identique d'un projet à l'autre que j'ai eu à réaliser (mis à part quelques légères adaptation). Sustainapp Batch contient en fait toutes les informations nécessaires à l'intégration du projet sur une plateforme de déploiement (une machine serveur).

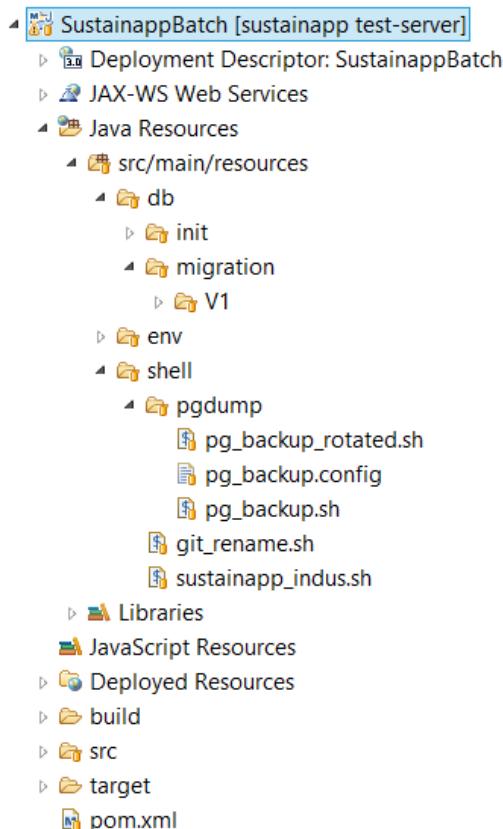


Figure 61: Architecture de Sustainapp Batch

Sustainapp Batch est également un projet Java/Maven utilisant donc le pom.xml et profite également des fonctionnalités de compilation via la commande « *mvn [command]* ».

Ce module est constitué de deux principales parties : la gestion de la base de données (installation, mise à jour) et la gestion du serveur (déploiement version, relancer le serveur, éteindre le serveur).

1.6.1. Flyway : Gestion de la base de données

Le dossier « *db* » contient des scripts SQL de migration de la base de données qui seront lancés par *Flyway*⁴⁶ (dépendance *Maven*), un outil open source de migration de base de données. Le principe est simple : *Flyway* possède une commande *Maven* pour installer/réinstaller la base de données et une pour mettre à jour en vérifiant si un nouveau fichier a été ajouté. Pour cela les scripts d'installation

⁴⁶ Site web officiel de Flyway : <https://flywaydb.org/>

(ajout d'utilisateur, de droits, création de la base...) se trouvent dans le dossier « *init* » et ceux de mise à jour se trouvent dans « *migration* ». Ainsi, on choisit de ne pas générer les tables à partir *d'Hibernate*.

```
-- ****
-- PARTIE TEAM [release 1]
-- ****

CREATE TABLE TEAM(
    ID SERIAL PRIMARY KEY,
    NAME VARCHAR(150),
    LEVEL INTEGER NOT NULL DEFAULT 0,
    TIMESTAMPS TIMESTAMP
);

CREATE TABLE TEAM_ROLE(
    ID SERIAL PRIMARY KEY,
    ROLE VARCHAR(50),
    TEAM_ID INTEGER NOT NULL,
    PROFILE_ID INTEGER NOT NULL,
    TIMESTAMPS TIMESTAMP,
    CONSTRAINT FK_TEAM_ROLE_TEAM_ID FOREIGN KEY (TEAM_ID) REFERENCES TEAM (ID),
    CONSTRAINT FK_TEAM_ROLE_PROFILE_ID FOREIGN KEY (PROFILE_ID) REFERENCES PROFILE (ID)
);
```

Figure 62: Exemple de migration SQL

Par ailleurs, nous avons choisi également pour des raisons techniques (sécurité et robustesse) de ne pas créer de système de fichier dynamique au niveau du serveur et de sauvegarder les images et documents uploadés par les utilisateurs directement dans la base de données.

Enfin afin de faciliter le travail futur nous avons choisi de ne pas réécrire les commandes *Maven* de lancement de *Flyway* ou les commandes *Spring boot* de lancement du module Sustainapp Server mais de sauvegarder dans Eclipse pour créer un bouton d'accès direct à la commande.

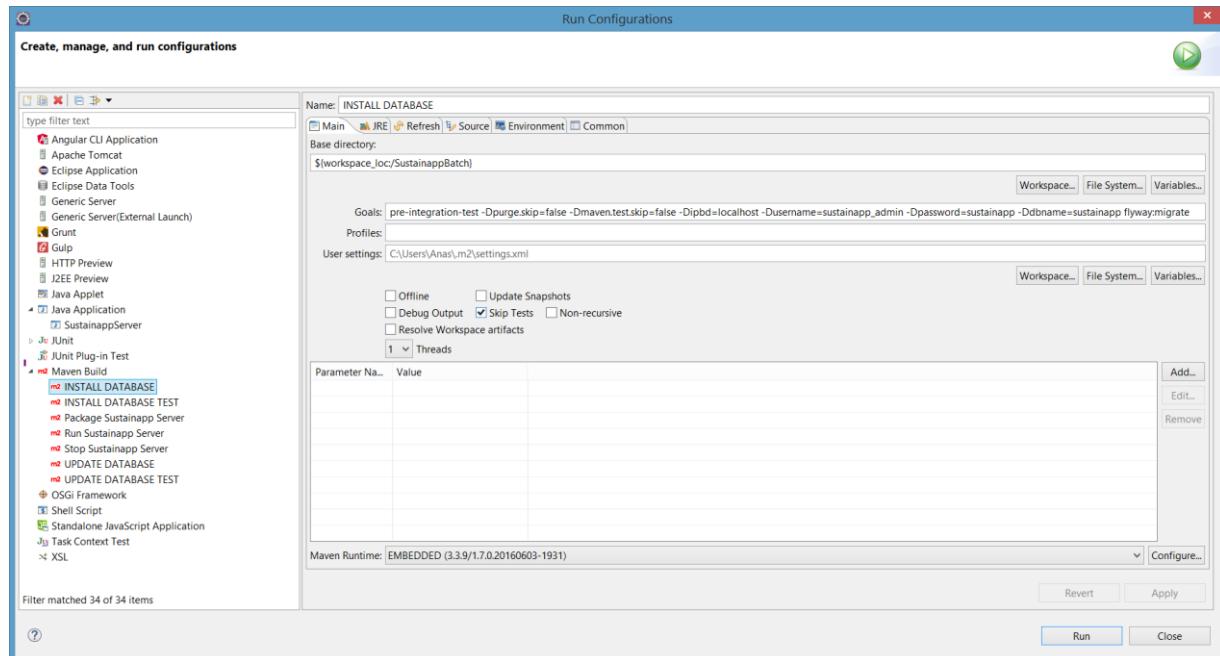


Figure 63: Crédit d'un "Job Maven" basé sur Flyway

1.6.2. Gestion du serveur de production

La seconde grande utilité du module est de permettre le déploiement mais aussi l'intégration continue de l'application sur le serveur et cette partie sera plus approfondie dans la partie consacrée au déploiement. Cependant ce module est présent depuis le début du projet. Pour cela Sustainapp Batch se base sur deux principaux outils : un fichier de configuration qui remplace celui de Sustainapp Server ainsi qu'une série de scripts écrits en Shell que l'on trouve dans le dossier « *shell* ».

Le plus important de ces script est « *sustainapp_indus.sh* » qui contient une série de fonctions shell ayant toute une fonction pour le serveur : lancer Sustainapp Serveur, lancer Flyway installation, lancer Flyway mise à jour, mettre à jour l'application (déployer la dernière version du code à partir du GIT et plus particulièrement la branche Master). On peut considérer que sur le server nous n'aurons pas accès à Eclipse et donc aux boutons que nous avons créés, ainsi nous allons utiliser plus tard un outil de déploiement nomé Jenkins qui se contentera de générer un bouton par fonction de ce fichier.

```
GIT_REPO_URL="http://91.121.80.56:8181/gitlab/isamm/sustainapp.git"
INSTALL_DIR="/BACK_SUSTAINAPP"
SAVE_DIR="${INSTALL_DIR}/old"
GIT_DIR="${INSTALL_DIR}/export_git"
BATCH_DIR="${GIT_DIR}/SustainappBatch"
TOMCAT_VERSION="8.5.15"
SUSTAINAPP_BACK_LOGFILE="${INSTALL_DIR}/sustainapp-back.log"

usage(){
    echo "Usage: ./livraison.sh [options]"
    echo "-h : afficher l'aide"
    echo "-u : mettre à jour le script de livraison"
    echo "-b : fabriquer les artifacts à partir du master"
    echo "-d : deploy de l'artifact back"
    echo "-p : purger les logs"
    echo "--repair : réparer les checksums de la bdd"
    echo "--bdd : mettre à jour la bdd"
    echo "--backup : backup de la base de données PostgreSQL (avec rotation)"
}

error(){
    echo "Erreur : paramètres invalides !" >&2
    echo "Utilisez l'option -h pour en savoir plus" >&2
    exit 1
}

git_update(){
    [[ $1 ]] && branche=$1 || branche="master"

    if [[ ! -d "$GIT_DIR" ]]; then
        git clone "$GIT_REPO_URL" "$GIT_DIR"
        cd "$GIT_DIR"
        git config credential.helper store
    fi
    cd "$GIT_DIR"
}
```

Figure 64: *sustainapp_indus.sh*

L'autre script est un script lié à la base de données PostgreSQL et permettra de la même manière d'utiliser Jenkins afin cette fois ci de lancer des commandes de backup pour la base.

1.7. Réalisation des premières fonctionnalités

Le projet est totalement prêt à être lancé : les applications sont exécutables, les environnements de développement et création de versions sont prêts, les fonctionnalités à développer sont claires et la charte graphique est également validée. Ainsi nous avons créé une première branche de version « *inscription de compte* » et avons commencé à développer les 2 fonctionnalités du livrable 0.

1.7.1. Inscription et Connexion à Sustainapp

Comme décrit dans les diagrammes de cas d'utilisation, un utilisateur anonyme ne peut rien faire d'autre que se connecter ou s'inscrire et l'accès à toutes autres fonctionnalités nécessite une connexion en tant que membre ou administrateur. Ainsi, il est logique que les fonctionnalités d'inscription et connexion soient les premières à être développées.

Ce sont des fonctionnalités comme toute assez simples à développer : vérification des champs (format du mail, existence en base) mis à part deux points :

- Le cryptage en base du mot de passe en cas d'intrusion. Pour cela nous avons développé un algorithme basé sur l'algorithme « *MD5* » et la classe « *MessageDigest* », native de java, dont voici un extrait :

```
/*
 * Methode de cryptage en md5 des mots de passes
 * @param input
 * @return
 */
public static String md5Hash(String input){
    try {
        MessageDigest md = MessageDigest.getInstance("MD5");
        md.update(input.getBytes());
        byte[] bytes = md.digest();
        StringBuilder sb = new StringBuilder();
        for(int i=0; i< bytes.length ;i++)
        {
            sb.append(Integer.toHexString((bytes[i] & 0xff) + 0x100, 16).substring(1));
        }
        return sb.toString();
    }
    catch (NoSuchAlgorithmException e)
    {
        return null;
    }
}
```

Figure 65: Cryptage en MD5 du mot de passe

- L'appel d'un Controller REST ne permet pas d'accéder aux « *HttpSession* » de la technologie JEE et donc il faut trouver un autre moyen d'identifier l'utilisateur que les sessions sans non plus envoyer ses identifiants de connexion mail/mot de passe dans les appels en cas d'interception. On utilise pour cela un *token* également crypté et surtout à durée de vie temporaire. Celui-ci génère une erreur 401 lors ce qu'il perd sa validité (principe emprunté au protocole « *Auth2.0* ») et enclenche une procédure de génération du *token* suivant.

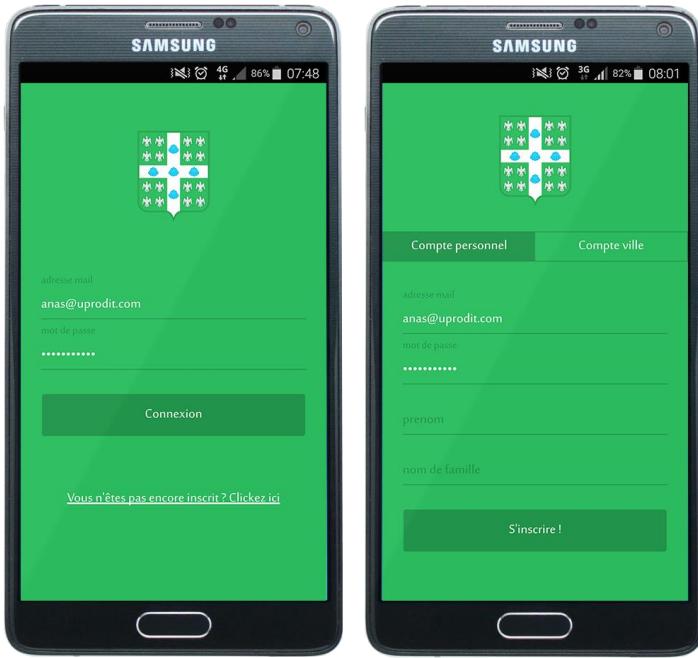


Figure 66: Interfaces de connexion et d'inscription

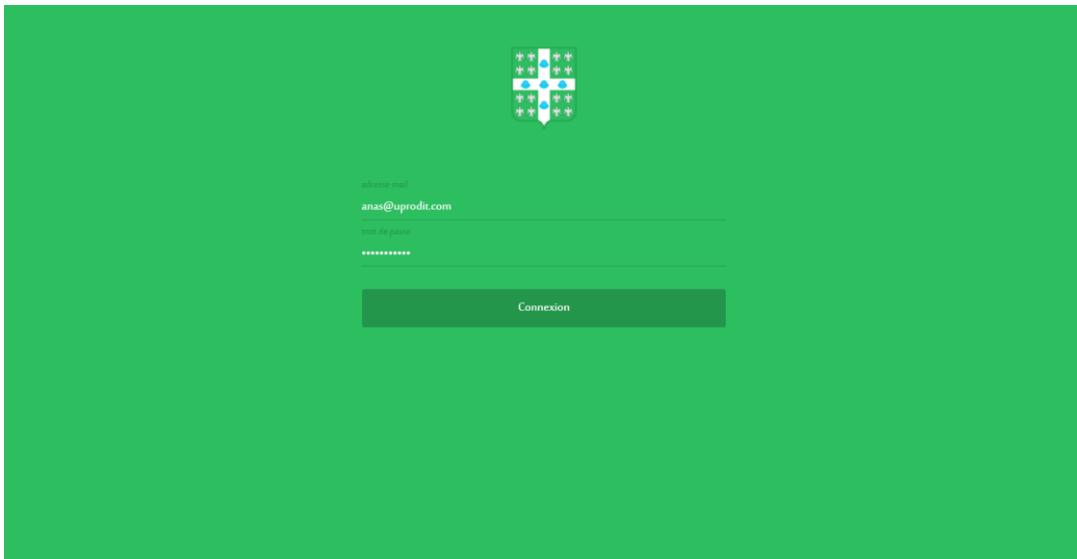


Figure 67: Connexion en version web

1.7.2. Signaler un problème à l'administration

L'autre user story concernait donc le fait de signaler un abus quelconque à l'administration⁴⁷. Ce signalement se fait par l'envoi d'une photo/image accompagnée d'un message par un membre qui sera reçue et affichée dans l'administration. Cette fonctionnalité tout comme la première est assez simple mais a permis de mettre en place la manière d'uploader les images, manière qui sera réutilisée par la suite dans de nombreuses fonctionnalités.

⁴⁷ Tutoriel pour signaler un abus : <https://www.facebook.com/cirrelt.sustainapp/videos/679041062280983/>

Le premier défi rencontré durant la réalisation de cette fonctionnalité a été de pouvoir vérifier dynamiquement si l'utilisateur utilisait l'application sur un navigateur web ou compilée pour mobile. Car en réalité l'utilisation n'est pas la même dans les deux cas :

- Dans un site web, on utilise pour sélectionner une image à uploader un simple input de type « *file* » ;
- Dans l'application mobile l'utilisateur peut choisir d'utiliser la caméra pour prendre une nouvelle photo ou de chercher dans sa galerie puis on lui propose de modifier la photo prise (recadrage principalement) via les applications qu'il possède sur son téléphone.

Ainsi on a créé pour cela un service business qui permet de déterminer la nature du « *device* » utilisé. Ensuite le deuxième défi aura été d'utiliser *ngCordova* pour accéder à l'appareil photo ou la galerie du téléphone :

```
angular.module('sustainapp.services')
.factory('fileService', function($cordovaCamera, $cordovaFileTransfer, $cordovaFile, $cordovaDevice) {
    var buildOptions = function(quality, width, height, allow){
        return {
            destinationType: Camera.DestinationType.DATA_URL,
            allowEdit: allow,
            encodingType: Camera.EncodingType.JPEG,
            quality : quality,
            targetWidth : width,
            targetHeight : height,
            saveToPhotoAlbum: false
        };
    };
    return {
        getFile : function(newFile, quality, width, height, allow){
            var options = buildOptions(quality, width, height, allow);
            if(true == newFile){
                options.sourceType = Camera.PictureSourceType.CAMERA;
            } else {
                options.sourceType = Camera.PictureSourceType.PHOTOLIBRARY;
            }
            return $cordovaCamera.getPicture(options);
        }
    };
});
```

Figure 68: Accès à la caméra du téléphone

Enfin côté serveur on optimise la taille des images en utilisant des algorithmes de compression et redimensionnement des fichiers uploadés :

```

/**
 * Methode de compression des images.
 *
 * @param input
 * @param format
 * @return byte[] compressedImage
 */
public static byte[] compressImage(byte[] input, String format) {
    InputStream inputStream = new ByteArrayInputStream(input);
    ByteArrayOutputStream outputStream = new ByteArrayOutputStream();
    BufferedImage initialImage = null;
    try {
        initialImage = read(inputStream);
        int height = initialImage.getHeight();
        int width = initialImage.getWidth();
        BufferedImage resizedImage = initialImage;

        if ((width / 2) > 2048 || (height / 2) > 2048) {
            while ((width / 2) > 2048 || (height / 2) > 2048) {
                width = width / 2;
                height = height / 2;
            }
            resizedImage = resizeImage(initialImage, width, height);
        }
        BufferedImage finalImage = resizedImage;
        if (FORMAT_PNG.equals(format)) {
            finalImage = png2jpg(resizedImage);
        }
        write(finalImage, FORMAT_JPG, outputStream);
    } catch (IOException e) {
        return null;
    }
    return outputStream.toByteArray();
}

```

Figure 69: Compression d'une image

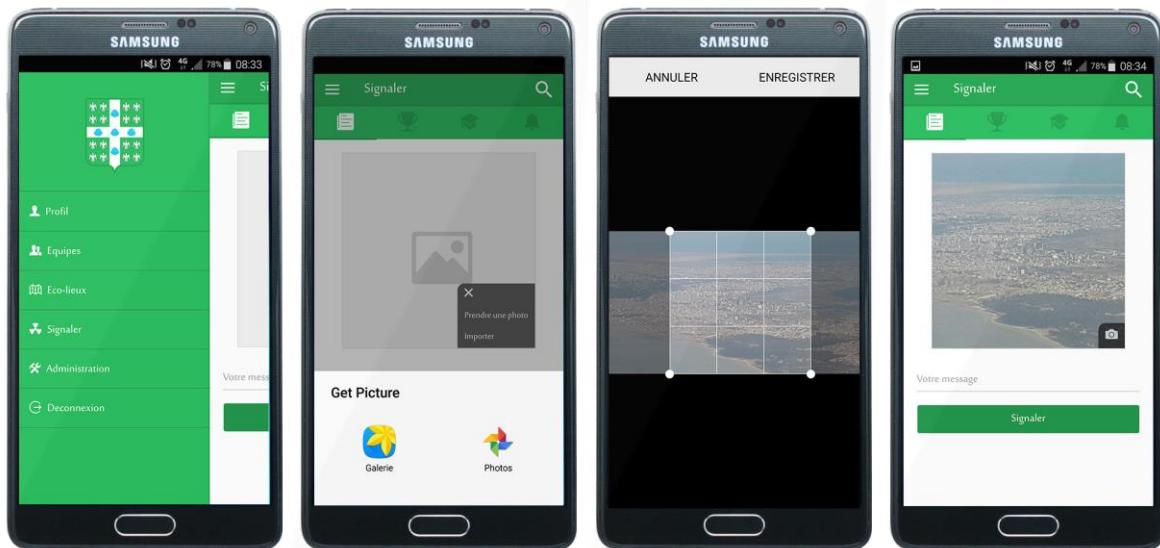


Figure 70: Signaler un abus

II. Un réseau d'écoresponsables

2.1. Planification de l'itération

Durant la réunion qui a suivi, nous avons validé les deux fonctionnalités précédentes et avons planifié la suite : l'utilisation d'un profil et la gestion des équipes. Comme la précédente, cette itération à durée 2 semaines du 15 Mars au 1^{er} Avril 2017 (120h). Voici donc les user stories à réaliser durant cette itération ainsi que les tâches réalisées :

Tableau 6: Découpage en tâches du sprint 1

Numéro US :	06	07	08
Priorité :	01	02	03
01	Affichage d'un profil	Création d'une équipe	Demander à participer dans une équipe
02	Affichage des badges	Affichage d'une équipe	Accepter les demandes de participation
03	Modification d'un profil	Modification d'une équipe	Renvoyer un membre
04	Modification des photos	Modification de l'avatar	
05		Suppression d'une équipe	

2.2. Réalisation des fonctionnalités

2.2.1. Gestion d'un profil personnel

A cette étape du projet le profil est encore assez simple : les modifications qui peuvent être effectuées sont celles de la photo de profil (avatar), de fond de page (couverture), des informations (date de naissance, nom et prénom) et du caractère secret (invisible aux autres utilisateurs) ou public du profil. Un utilisateur peut choisir de supprimer définitivement son profil et compte et pour cela bien sûr, une demande de confirmation explicative du risque a été ajoutée.

Pour cela on réutilise les services de gestion des images et de l'upload créés précédemment avec cette fois-ci l'ajout d'un critère en plus : les droits de modification. Par exemple seul les administrateurs et le propriétaire du profil ont accès aux outils de modifications du profil et les autres utilisateurs le visualisent uniquement.



Figure 71: Interfaces de modification du profil

2.2.2. Gestion et participation dans les équipes

Certaines fonctionnalités qui seront développées par l'avenir peuvent être utilisées en équipe (par exemple les challenges) et donc cette fonctionnalité a été jugée prioritaire. Une équipe regroupe plusieurs profils dont un seul appelé « *capitaine* » : le créateur de l'équipe. Celui-ci est le seul avec les administrateurs à pouvoir gérer les informations et les membres de l'équipe. Il existe donc 4 situations ou rôles entre un profil et une équipe : aucune relation, en demande de participation, membre admis dans l'équipe, capitaine d'équipe⁴⁸.

Quelques défis ont dû être relevés durant la réalisation de cette fonctionnalité : afficher les équipes de manière paginée (sinon des milliers de résultats pourraient venir en même temps et donc ralentir fortement l'application) en utilisant le concept d' « *infinity scroll*⁴⁹ ». Nous avons aussi utilisé dans la réalisation de cette fonctionnalité certains éléments graphiques intéressants tels que les listes animées d'*Ionic*.

⁴⁸ Tutoriel utilisation d'un profil et d'une équipe :
<https://www.facebook.com/cirreit.sustainapp/videos/679053078946448/>

⁴⁹ **Infinity Scroll** : technique avec laquelle du contenu asynchrone est chargé automatiquement quand l'utilisateur scroll en bas de la page.

```
/**
 * get all teams
 * @return
 */
@ResponseBody
@RequestMapping(value = "/team/all", method = RequestMethod.GET, produces = SustainappConstantes.MIME_JSON)
public ResponseEntity<String> getAll(HttpServletRequest request) {
    Optional<Long> startIndex = StringsUtils.parseLongQuickly(request.getParameter("startIndex"));
    if (!startIndex.isPresent()) {
        return super.refuse();
    }
    SearchResult<TeamEntity> listResult = teamService.searchByCriteres(null, startIndex.get(), 20L);
    List<TeamEntity> teams = new SustainappList<TeamEntity>();
    for (TeamEntity team : listResult.getResults()) {
        teams.add(team);
    }
    return super.success(new TeamsResponse().setTeams(teams));
}
```

Figure 72: Récupération paginée des équipes

La méthode du côté serveur attend un index de démarrage «*startIndex*» de la recherche et récupère un maximum de 20 résultats à la fois. Ainsi du côté client on incrémente l'index de démarrage à chaque fois que l'utilisateur atteint le bas de la page.

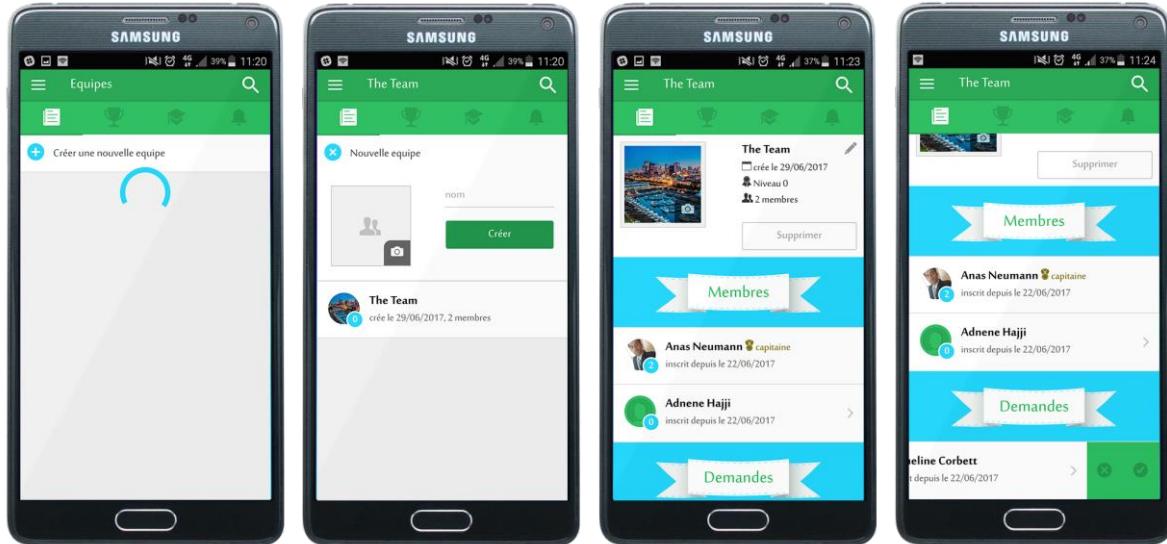


Figure 73: Interfaces liées aux équipes

III. Les challenges

Le 1^{er} avril débute une nouvelle itération, qui se démarque des deux précédentes par l'entrée en scène de la première fonctionnalité faisant intervenir la gamification, qui est au cœur de ce projet⁵⁰. Les challenges de Sustainapp sont un lieu de partage dans lequel les profils et même les équipes peuvent partager des réalisations écologiques liées à un domaine précis (éco-gastronomie par exemple) et voter pour la meilleure réalisation partagée.

3.1. Planification de l'itération

Voici donc les user stories planifiées pour cette itération entre le 1^{er} et le 15 Avril 2017 ainsi que le découpage en tâches :

Tableau 7: Découpage en tâches du sprint 2

Numéro US :	09	10	11
Priorité :	01	02	03
01	Création d'un challenge	Ajouter une participation	Voter pour une participation
02	Affichage des challenges	Modification d'une participation	
03	Filtrer par catégories	Suppression d'une équipe	
04	Supprimer d'un challenge		
05	Modification d'un challenge		
06	Modification de l'image d'un challenge		

3.2. Réalisation des fonctionnalités

3.2.1. Défis relevés

Cette fonctionnalité a, elle aussi, provoqué certains défis que nous avons dû relever. Certains d'entre eux concernaient le développement :

⁵⁰ Tutoriel de l'utilisation des challenges :
<https://www.facebook.com/cirreit.sustainapp/videos/679078368943919/>

- Pour le créateur et gestionnaire d'un challenge, une multitude de choix lui seront proposés tels que accepter le jeu en équipe ou seulement individuel, la date de fin du challenge pour qu'il se ferme automatiquement ou encore le niveau minimum des équipes ou profils qui peuvent participer ;
- Pour celui qui participe à un challenge : il pourra filtrer ses recherches de challenge par catégories pour plus de facilité. Il pourra également accéder à un menu de choix de l'équipe ou du profil avec lequel il participe.

Certains défis ont été également graphiques car cette fonctionnalité a entraîné la création de plusieurs images telles que l'image par défaut d'un challenge quand il ne possède pas sa propre image ou encore l'icône pour chaque challenge :

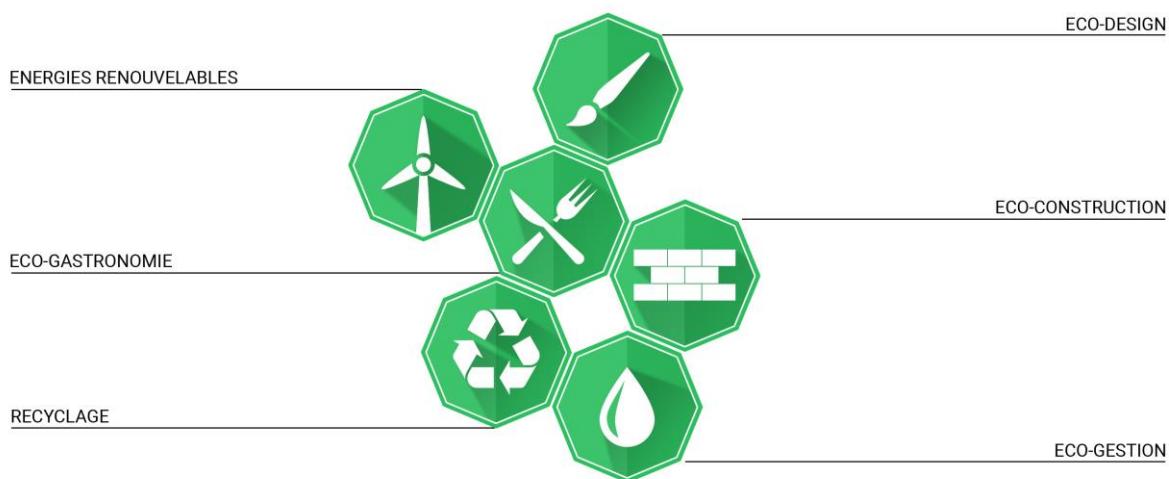


Figure 74: Les catégories pour filtrer

3.2.2. Interfaces réalisées

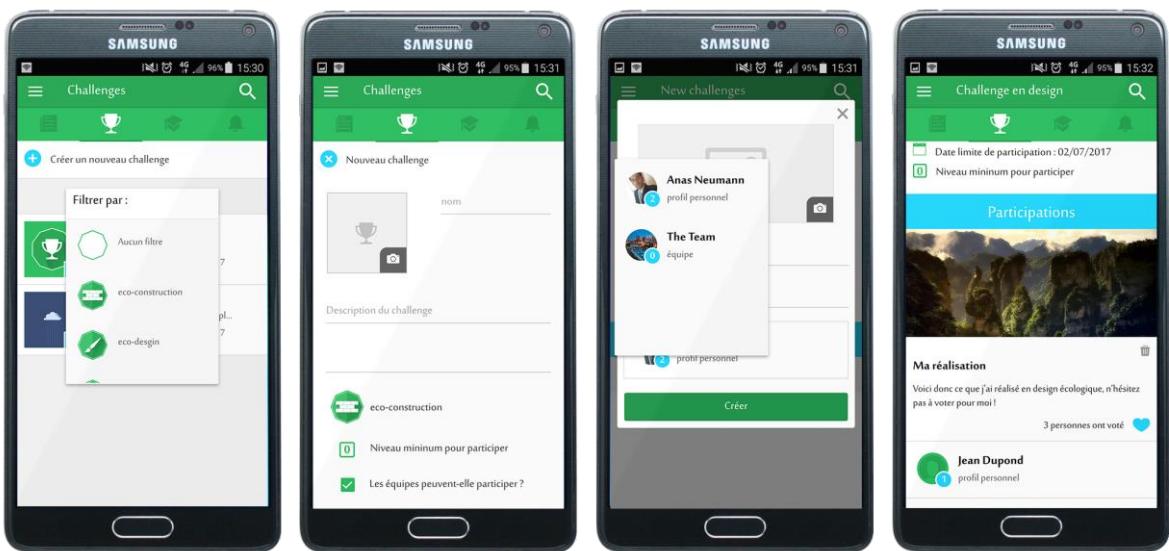


Figure 75: Interfaces liées aux challenges

IV. Les cours en écologie

Le 16 avril commence la création des fonctionnalités liées aux cours en écologie, qui représentent le cœur de l'application. Nous avons souhaité nous démarquer des cours mobiles classiques par l'ajout de la gamification et ainsi les quizz qui permettent de valider chaque chapitre ne sont pas de simples questions mais des mini-jeux interactifs⁵¹. Le contenu même des cours a été pensé pour être interactif et est donc composé d'images, de vidéos et de liens et pas seulement de textes⁵².



Figure 76: Icônes pour l'ajout d'une partie dans un chapitre

4.1. Planification de l'itération

Comme les précédents, ce livrable a été développé en deux semaines entre le 16 avril et le 1^{er} mai 2017 et voici les différentes *user stories* qui le composent ainsi que le découpage en tâches :

Tableau 8: Découpage en tâches du sprint 3

Numéro US :	12	13	14
Priorité :	01	02	03
01	Création d'un cours	Ajouter une question	Ajouter un cours
02	Suppression d'un cours	Modification d'une question	Affichage d'un quizz
03	Modification de la description d'un cours	Ajouter une réponse	Passer un quizz
04	Ajout d'un chapitre	Réorganiser les réponses	Noter un cours
05	Réorganisation des chapitres	Réorganiser les questions	

⁵¹ Vidéo tutoriel de la création et validation d'un quizz :

<https://www.facebook.com/cirreLT.sustainapp/videos/679153648936391/>

⁵² Vidéo tutoriel de la gestion d'un cours et des chapitres :

<https://www.facebook.com/cirreLT.sustainapp/videos/679153165603106/>

06	Suppression d'un chapitre	Supprimer une question	
07	Ajout d'une partie	Supprimer une réponse	
08	Suppression d'une partie		
09	Réorganiser les parties		
10	Modifier une partie		

4.2. Réalisation des fonctionnalités liées aux cours

4.2.1. Suivre un cours

Tout comme pour les challenges, les cours peuvent être triés par catégories mais, le membre peut également choisir de les trier par langue. Pour le moment il n'en existe que deux : le français et l'anglais (les deux langues du Canada) comme pour l'i18n de l'application elle-même mise en place au Sprint 0. Cependant ceci a été conçu de manière modulable et peut facilement évoluer vers l'ajout de nouvelles langues.



Figure 77: Icônes de filtrage par langue

Ensuite, un cours étant constitué de chapitres, le menu principal d'un cours présente une liste des chapitres qui le composent. Pour faciliter l'utilisation, une petite icône est présente devant chaque chapitre pour rappeler à l'utilisateur s'il a déjà validé ce chapitre, s'il ne l'a pas encore fait et une autre s'il a échoué au chapitre.

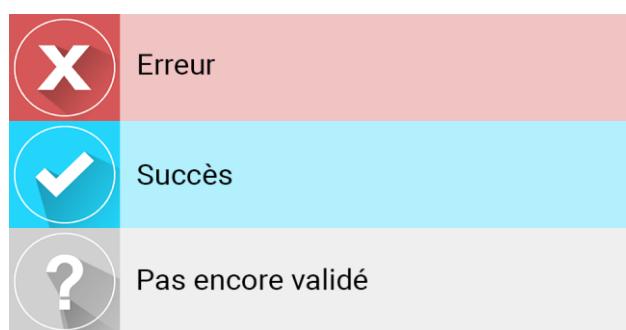


Figure 78: Les états possibles pour un chapitre

Enfin, un membre peut également noter un cours pour dire s'il le trouve bien ou au contraire s'il juge que le cours ne mérite pas sa place au sein de l'application. Pour cela nous avons décidé d'utiliser une librairie créée spécialement pour *Ionic* : « *Ionic-Rating* »⁵³ qui permet de noter avec un système d'étoiles (note allant de 0 à 5 étoiles). Nous verrons par la suite que ces notes deviendront très importantes pour le créateur du cours.

4.2.2. Création et gestion d'un cours

La création d'un cours est très similaire à la création d'un challenge (nom, image par défaut, choix du niveau et de la catégorie...) cependant là aussi de nouveaux défis sont apparus, par exemple :

- l'utilisation de la réorganisation en « *drag & drop* » (glisser et déposer avec le doigt) des chapitres si l'on trouve un nouveau plan pour le cours ;

```
/**
 * Reorder topics
 */
$scope.moveTopic = function(elt, fromIndex, toIndex) {
    var data = new FormData();
    data.append("topic", elt.topic.id);
    data.append("position", toIndex);
    data.append("sessionId", sessionService.get('id'));
    data.append("sessionToken", sessionService.get('token'));
    topicService.drop(data).success(function(result) {
        $scope.coursModel.topics.splice(fromIndex, 1);
        $scope.coursModel.topics.splice(toIndex, 0, elt);
    }).error(function(error){
        sessionService.refresh(null);
    });
};
```

Figure 79: Affichage et sauvegarde en base du déplacement d'un chapitre

- La création des parties dans un chapitre, car une partie peut être un lien, une image, du texte ou une vidéo d'un site externe tel que « *Youtube* ». Or des traitements doivent être effectués sur les liens et surtout sur les URLs de *Youtube* pour pouvoir afficher directement la vidéo.

⁵³ Ionic Rating sur Github : <https://github.com/fraserxu/ionic-rating>

```
/*
 * Méthode de parsing d'une url pour construire l'embed de youtube
 * @param url
 * @return
 */
public static String buildYoutubeEmbed(String url){
    if(null == url){
        return "";
    }
    return "https://www.youtube.com/embed/" + getVideoIDFromYoutube(url);
}

/**
 * Retrouver l'id d'une vidéo youtube à partir de l'url
 * @param url
 * @return
 */
public static String getVideoIDFromYoutube(String url){
    if(null == url){
        return null;
    }
    String pattern = "(?=<watch\\v=|/videos|/embed\\v)[^#\\&\\?]*";
    Pattern compiledPattern = Pattern.compile(pattern);
    Matcher matcher = compiledPattern.matcher(url);
    if(matcher.find()){
        return matcher.group();
    }
    return "";
}
```

Figure 80: Construction d'un lien pour l'affichage d'une vidéo Youtube

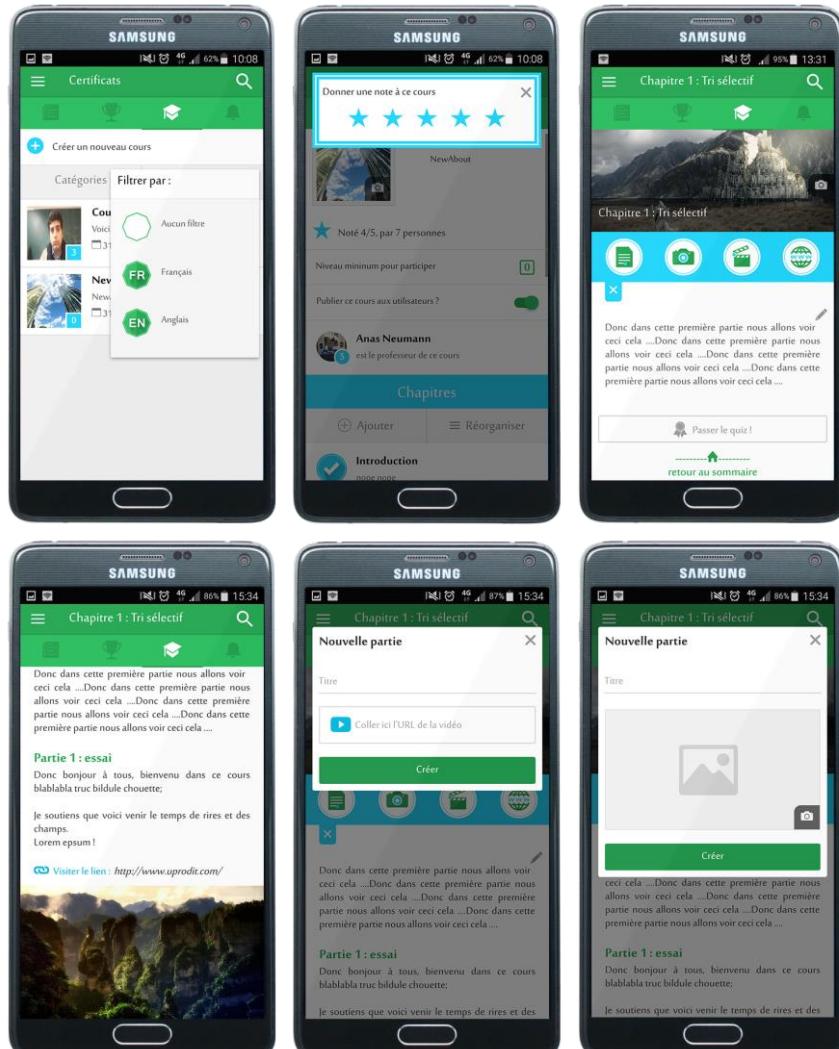


Figure 81: Interfaces liées aux cours

4.3. Réalisation des fonctionnalités liées aux quizz

Souhaitant que les quizz de validation ne soient pas de simples vrai/faux ou réponses textuelles mais bien plus interactifs que cela, le créateur à le choix entre 4 types de question : classification de termes, réordonner des images, sélection d'images ou encore QCM (questionnaires à choix multiples) textuel.

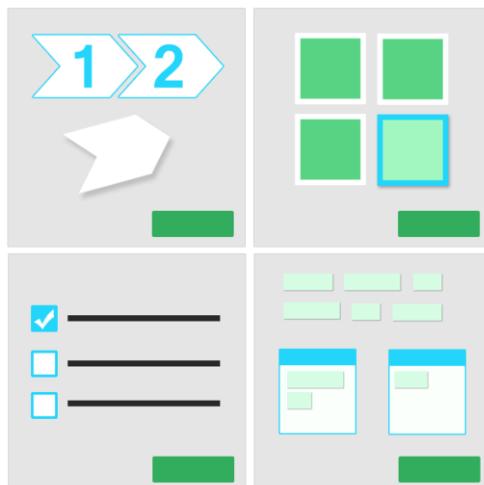


Figure 82: Icône de choix du type de question

Le créateur doit également fournir les réponses aux questions qu'il crée pour que le quizz soit fonctionnel.

Du côté de l'utilisateur qui souhaite valider le quizz, l'interface est très différente : une barre de progression s'affiche et se divise dynamiquement selon le nombre de question du quizz et le jeu se met en place au centre de l'écran. Ici encore quelques défis ont dû être relevés :

- l'utilisation et l'installation d'une librairie Angular de « *Drag & Drop* » pour les questions de tri d'images : « *ngDraggable* »⁵⁴ ;
- la validation des questions côté serveur: comment détailler les traitements de déplacement ou de sélection fait par l'utilisateur ? Nous avons pour cela choisi de créer un protocole simple qui envoie toutes les réponses dans une simple chaîne de caractères en utilisant des séparateurs « / » ou « ; » pour séparer les différentes réponses et les questions entre elles.

⁵⁴ **ngDraggable** sur Github : <https://github.com/fatlinessofcode/ngDraggable#ngdraggable>

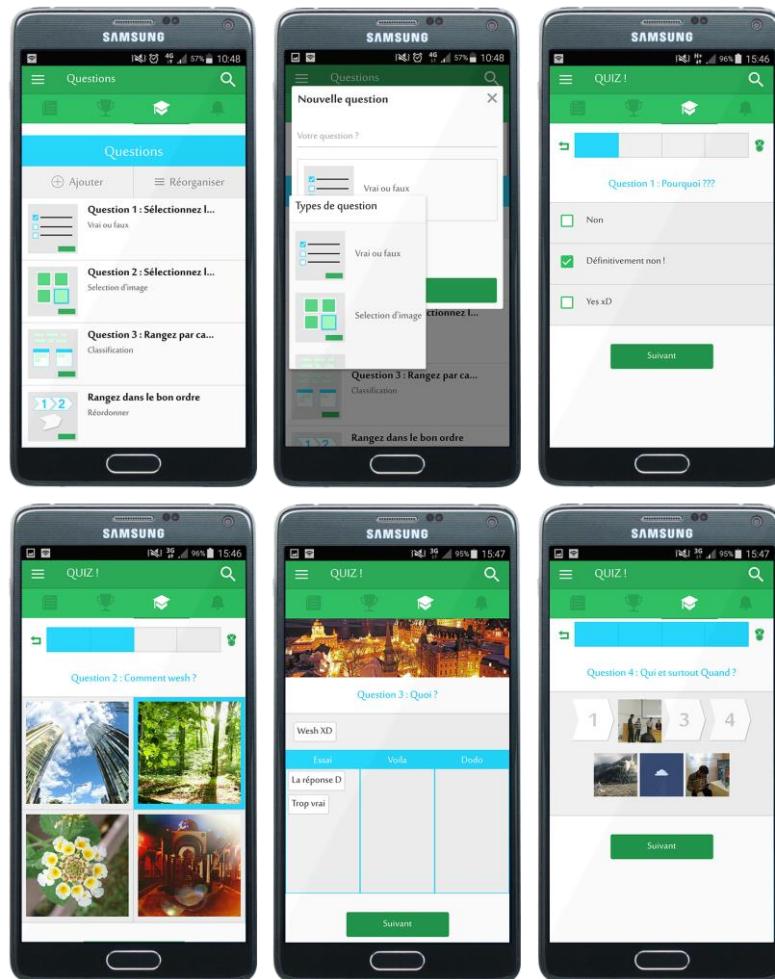


Figure 83: Interfaces liées aux quizz

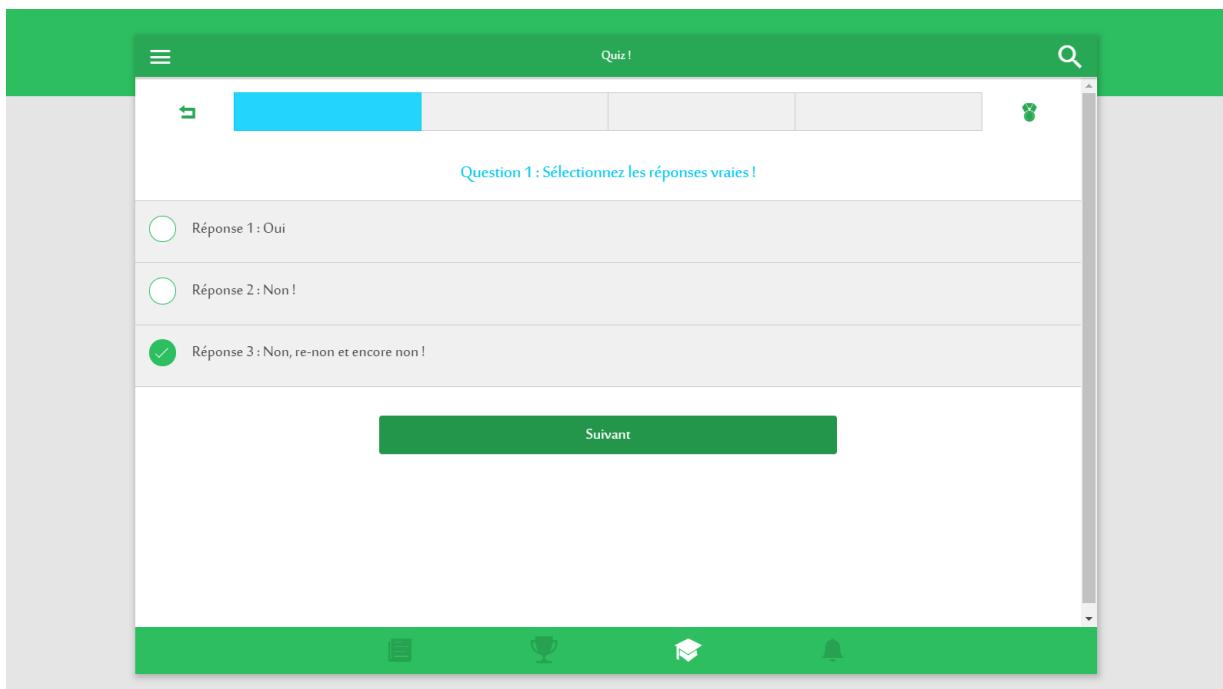


Figure 84: Interface web des Quizz

V. Les actualités du réseau

Toute application mobile et plateforme web, qui se veut être un réseau social, doit intégrer certains principes sociaux et certaines fonctionnalités devenues incontournables aujourd'hui : être prévenu des actualités du réseau. Ce 4^e livrable est donc réservé à réaliser un système de notification temps réel complet, un outil de recherche (pour profils, cours, équipes et plus tard de villes) ainsi qu'un journal présentant les meilleurs cours (en termes de notes) et les meilleures réalisations (en termes de votes) du moment. Durant cette itération nous allons aussi finaliser un des grands mécanismes de gamification que nous avions prévu : l'obtention de badges et de niveaux dans les profils.

5.1. Planification de l'itération

Voici donc comme pour chaque itération, les *user stories* que nous avons réalisées entre le 1^{er} mai et le 15 mai 2017 ainsi que le découpage en tâches :

Tableau 9: Découpage en tâches du sprint 4

Numéro US :	15	16	147	18
Priorité :	01	02	03	04
01	Affichage des meilleurs cours	Affichage des badges	Création d'une nouvelle notification	Sauvegarder les recherches en base
02	Affichages des meilleures participations	Récupération d'un badge	Affichage sur mobile	
03			Réception temps réel d'une notification	
04			Passer une notification à lue	

4.2. Réalisation des fonctionnalités

4.2.1. Recherche

Pour plus de facilité, la barre de recherche est présente dans la barre d'en-tête, elle permet de lancer une recherche sur des villes, cours, profils et équipes par leur nom en s'adaptant à chaque lettre

ajoutée ou retirée par l'utilisateur et accepte la possibilité d'une saisie non complète, dans le mauvais ordre ou encore avec des erreurs de majuscules. Pour cela nous avons créé des requêtes exploitant le « *Hql* » de *Hibernate* et naturellement les résultats sont paginés pour ne pas être trop long à charger :

```
/*
 * Select all teams by keywords
 * @param Keywords
 * @return
 */
@Query("FROM TeamEntity t where LOWER(t.name) LIKE CONCAT('%',LOWER(:keywords),'%')")
List<TeamEntity> searchByKeywords(@Param("keywords") String Keywords, Pageable pageable);
```

Figure 85: Utilisation de Hql pour lancer des recherches paginées

Les recherches de plus de 3 lettres ayant engendré un résultat sont sauvegardées à des fins statistiques pour l'administration.



Figure 86: Interfaces liées à la recherche

4.2.2. Le journal de Sustainapp

Le journal est un podium pour cours et réalisations. C'est une fonctionnalité assez simple dans sa réalisation mais très utile sur le plan social pour la gamification. Le journal n'affiche que les trois meilleurs de chaque catégorie.



Figure 87: Interface du journal de Sustainapp

4.2.3. Les badges dans le profil

Pour monter en niveau dans Sustainapp, il faut récupérer des badges. Ces badges sont très divers : il y en a pour celui qui valide beaucoup de quizz, celui qui a des cours bien notés, qui a des réalisations avec beaucoup de votes dans des challenges, qui est apparu dans le journal, qui a visité plusieurs éco-lieux ou encore qui a signalé des réels abus à l'administration...

Les badges ne s'affichent pas de la même couleur dans le profil dans le cas où le membre le possède (bleu) ou pas encore (gris) :



Figure 88: Icônes des différents badges

Techniquement nous avons créé un service business côté java avec des méthodes de vérification par exemple quand un joueur reçoit une note, si elle est bonne, combien en a-t-il reçu, possède-t-il déjà le badge du « bon professeur » ? Dans le cas où la méthode valide toutes les conditions : le badge est obtenu.

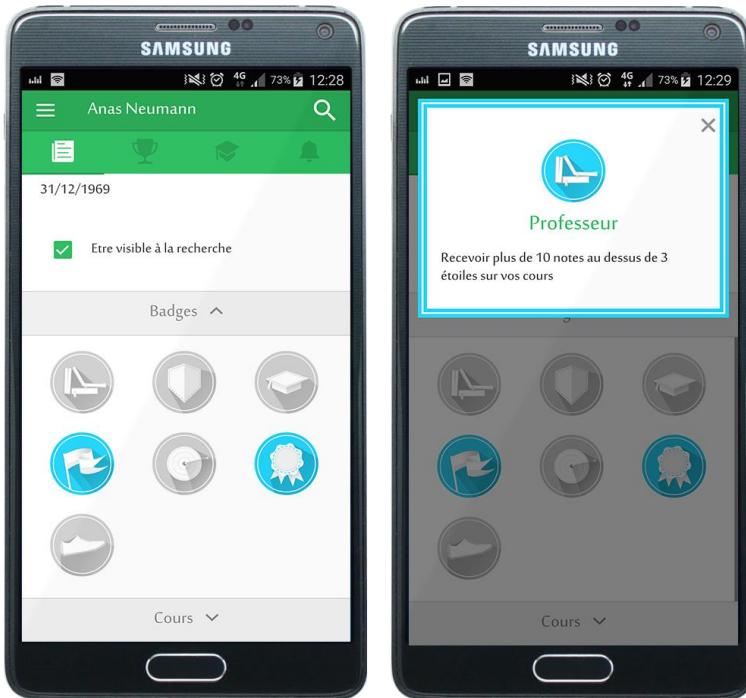


Figure 89: Interfaces liées aux badges

4.2.4. Les notifications temps réel

Nous avons souhaité que l'utilisateur soit toujours prévenu de toutes les nouveautés qui le concernent et qui risquent de l'intéresser. Ainsi l'utilisateur est notifié quand on accepte sa demande de participation à une équipe, quand il reçoit une demande en tant que capitaine, quand on vote pour une de ses réalisations, quand il reçoit une note sur l'un de ses cours, quand quelqu'un participe à l'un de ses challenges ou encore quand il obtient un nouveau badge.

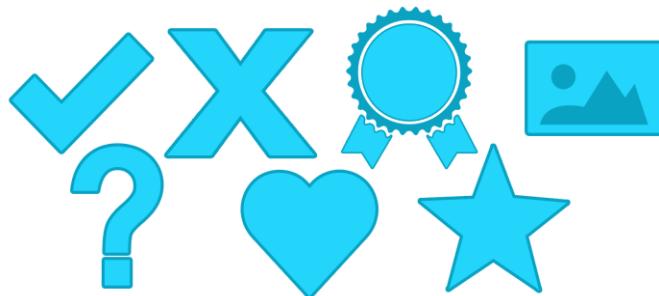


Figure 90: Icônes des différentes notifications

Cette fonctionnalité a représenté un véritable défi technique pour trois principales raisons :

4.2.4.1. Le défi du temps réel

Les notifications doivent arriver en temps réel. C'est-à-dire que le principe habituel de l'application mobile qui envoie une demande à l'application server qui ensuite lui répond (protocole HTTP/REST), n'est plus suffisant. Sinon il faudrait que l'application mobile envoie des demandes à intervalles réguliers, ce qui engendrerait beaucoup de requêtes et beaucoup d'accès en base, dont une très grande majorité inutiles. L'autre solution c'est le principe des *websockets* adapté par *Spring Boot* :

le client s'inscrit initialement sur une URL unique générée au niveau du serveur. Après quoi, le serveur envoie automatiquement les notifications via cette URL quand une nouvelle est créée. C'est du véritable temps réel et il n'y a aucun échange ou accès en base inutile. Par contre, cela nécessite de mettre en place une infrastructure supplémentaire sur les deux applications⁵⁵ :

```
@Configuration
@EnableWebSocketMessageBroker
public class SustainappWebSocketConfiguration extends AbstractWebSocketMessageBrokerConfigurer {

    /**
     * Méthode de configuration du chemin pour l'utilisation des websockets
     * @param config
     */
    @Override
    public void configureMessageBroker(MessageBrokerRegistry config) {
        config.enableSimpleBroker("/queue");
        config.setApplicationDestinationPrefixes("/app");
        config.setUserDestinationPrefix("/user");
    }

    /**
     * Méthode de configuration pour l'enregistrement via le protocol STOMP
     * @param registry
     */
    @Override
    public void registerStompEndpoints(StompEndpointRegistry registry) {
        registry.addEndpoint("/sustainapp-websocket").setAllowedOrigins("*").withSockJS();
    }
}
```

Figure 91: Mise en place des websockets sur Spring Boot

```
template.convertAndSend("/queue/notification-"+entity.getProfilId(), notification);
```

Figure 92: Envoi d'une websocket sur Spring Boot

4.2.4.2. Génération de notification native des plateformes

Le message reçu en *websocket* sera de type textuel (protocole STOMP⁵⁶) mais l'application effectue des traitements pour, notamment, afficher un petit chiffre dans la barre d'en-tête représentant le nombre de notifications non lues. Parmi ces traitements, on souhaite afficher des notifications natives (qui sont donc hors de l'application elle-même). On utilise pour cela un plugin de « *ngCordova* » (comme pour l'appareil photo) :

```
$cordovaLocalNotification.schedule({
    id: $scope.newNotifications.id,
    message: $scope.newNotifications.creator+" "+$filter('translate')($scope.newNotifications.message)+" "+$scope.newNotifications.target,
    title: "Sustainapp !",
    autoCancel: true,
    sound: null,
    icon : "notification/"+$scope.newNotifications.message.replace('.','')+".png"
}).then(function () {
});
```

Figure 93: Génération d'une notification native

4.2.4.3. Affichage complet et lien des notifications

Enfin, la dernière difficulté à résoudre concernant les notifications est leur affichage dans l'onglet d'historique des notifications. En effet on souhaite afficher un nombre assez élevé

⁵⁵ Tutoriel suivi pour la mise en place : <http://www.kswaugh.com/2016/12/spring-boot-websocket-server-push.html>

⁵⁶ Librairie STOMP pour AngularJs sur Github : <https://github.com/bevelop/ng-stomp>

d'informations et pas seulement la date et la nature de la notification, mais aussi parfois l'utilisateur qui a fait l'action, parfois sur quoi l'action a été faite, savoir si la notification a été lue, cliquée... Le lien enfin suivi par la notification est lui aussi hétérogène et peut amener parfois à un cours, un profil, une équipe, le journal ... C'est ce qui explique la nature particulière de la conception de la base de données.

Pour réaliser cela, nous avons créé un service business tout comme pour les badges qui « construit » dynamiquement les notifications avant l'affichage.

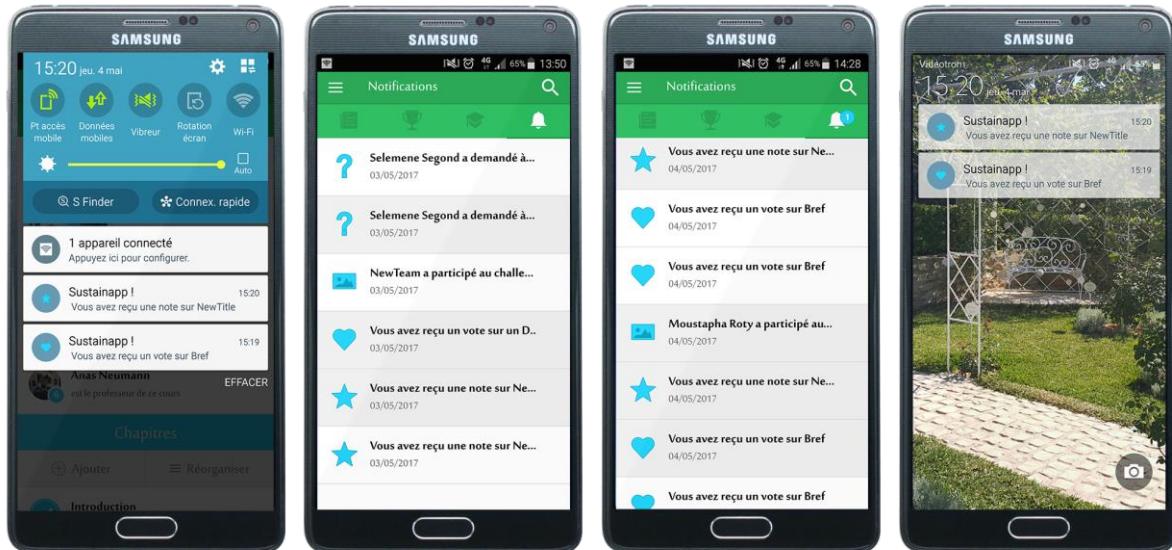


Figure 94: Interfaces liées aux notifications

VI. L'intégration dans les villes intelligentes

Québec, comme de nombreuses villes à travers le monde, s'est orienté vers une gestion écologique de son patrimoine. Son administration gère plusieurs sites liés à cette cause comme des parcs naturels, des événements, des centres de tri ou autre. Ainsi, nous avons souhaité que cette application puisse faire connaître aux habitants d'une ville les éco-lieux qui leur sont proches et qu'ils pourraient avoir envie de visiter⁵⁷.

Evidemment, le compte lié à l'administration d'une ville est soumis à bien plus de contrôles qu'un simple compte personnel. Ainsi chaque inscription en tant que compte ville doit avant tout passer par une étape de validation par l'équipe de Sustainapp qui contactera la ville préalablement.

6.1. Planification de L'itération

Cette itération a été réalisée entre le 16 mai et le 1^{er} juin 2017, et voici les *user stories* qui la composent ainsi que le découpage en tâches :

⁵⁷ Tutoriel de l'utilisation des villes et lieux :
<https://www.facebook.com/cirrelt.sustainapp/videos/679155585602864/>

Tableau 10: Découpage en tâches du sprint 5

Numéro US :	19	20
Priorité :	01	02
01	Inscription en mode ville	Afficher les éco-lieux proches sur la carte
02	Validation d'une ville	Affichage d'un lieu
03	Modification d'une ville	Vérifier ma position
04	Ajout d'un éco-lieu	Noter un lieu
05	Ajouter une image à un lieu	Rechercher une ville par son nom

6.2. Réalisation des fonctionnalités

6.2.1. Inscription sur un compte ville

Cette fonctionnalité nécessite un travail de modération par la suite et au niveau données, l'utilisateur ayant la responsabilité d'une ville doit avoir également un état : validé, en attente ou refusé.

6.2.2. La carte des éco-lieux

Une fois acceptée, une ville peut être recherchée dans le moteur et affichée mais surtout elle peut commencer à créer des éco-lieux qui seront visibles sur la carte.

Ici de nombreux défis techniques liés à la géolocalisation sont apparus :

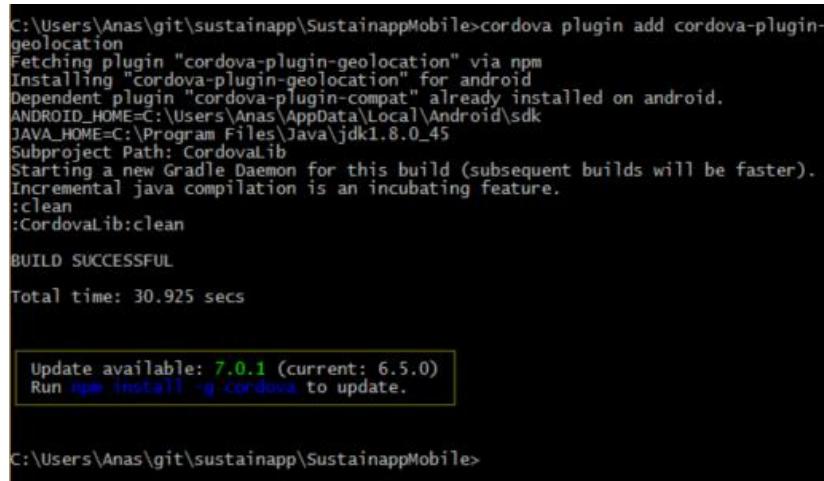
- Comment une ville fait-elle pour situer un lieu sur la carte si elle ne connaît que l'adresse mais pas directement la position en termes longitude et latitude ? Pour permettre cela, nous avons utilisé une API de google qui permet de calculer une position à partir d'une adresse. Ce qui nécessite de créer au préalable une clé auprès de la plateforme de Google API ;

```
angular.module('sustainapp.services').factory('geolocationService', function($http, config) {
  var BASE_API = "https://maps.googleapis.com/maps/api/";
  var API_KEY = "XXXXXXXXXXXXXX";
  return {
    getLocationFromAddress : function(Address) {
      return $http.get(BASE_API+"geocode/json?address="+Address+"&key="+API_KEY);
    }
  };
});
```

Figure 95: Calculer une position à partir d'une adresse

- Comment ne récupérer que les lieux proches d'un utilisateur ou vérifier qu'il a bel et bien visité l'un d'entre eux ? Cela nécessite de connaître la position exacte de l'utilisateur. Pour cela, nous

avons installé et utilisé un autre plugin de *ngCordova* qui permet d'utiliser les capteurs du téléphone pour géo-localiser l'utilisateur ;



```
C:\Users\Anas\git\sustainapp\SustainappMobile>cordova plugin add cordova-plugin-geolocation
Fetching plugin "cordova-plugin-geolocation" via npm
Installing "cordova-plugin-geolocation" for android
Dependent plugin "cordova-plugin-compat" already installed on android.
ANDROID_HOME=C:\Users\Anas\AppData\Local\Android\sdk
JAVA_HOME=C:\Program Files\Java\jdk1.8.0_45
Subproject Path: CordovaLib
Starting a new Gradle Daemon for this build (subsequent builds will be faster).
Incremental java compilation is an incubating feature.
:clean
:CordovaLib:clean
BUILD SUCCESSFUL

Total time: 30.925 secs

Update available: 7.0.1 (current: 6.5.0)
Run npm install -g cordova to update.

C:\Users\Anas\git\sustainapp\SustainappMobile>
```

Figure 96: Installation du plugin de géolocalisation

```
var posOptions = {
    timeout: 1000,
    enableHighAccuracy: false
};
$cordovaGeolocation.getCurrentPosition(posOptions)
```

Figure 97: Récuperer la position de l'utilisateur

- Enfin, comment afficher une carte du monde sur laquelle placer les éco-lieux en tant que « *marker* »⁵⁸ ? Nous avons utilisé la solution la plus connue : L'API JavaScript de « *Google Map* » qui tout comme la 1^{er} API nécessite la création préalable d'une clé.

6.2.3. Visiter des lieux

En sélectionnant le « *marker* » d'un lieu sur la carte on affiche une page qui lui est dédiée. Sur cette page on y trouve la description complète de ce lieu ainsi que des images qui l'illustre et l'utilisateur peut décider de noter le lieu de la même manière qu'il note les cours.

⁵⁸ Un **marker** sur Google map : est une icône graphique qui indique la présence d'un lieu remarquable à cet endroit. Dans notre cas, il s'agit d'éco-lieux (voir figure 98, 3^e écran).

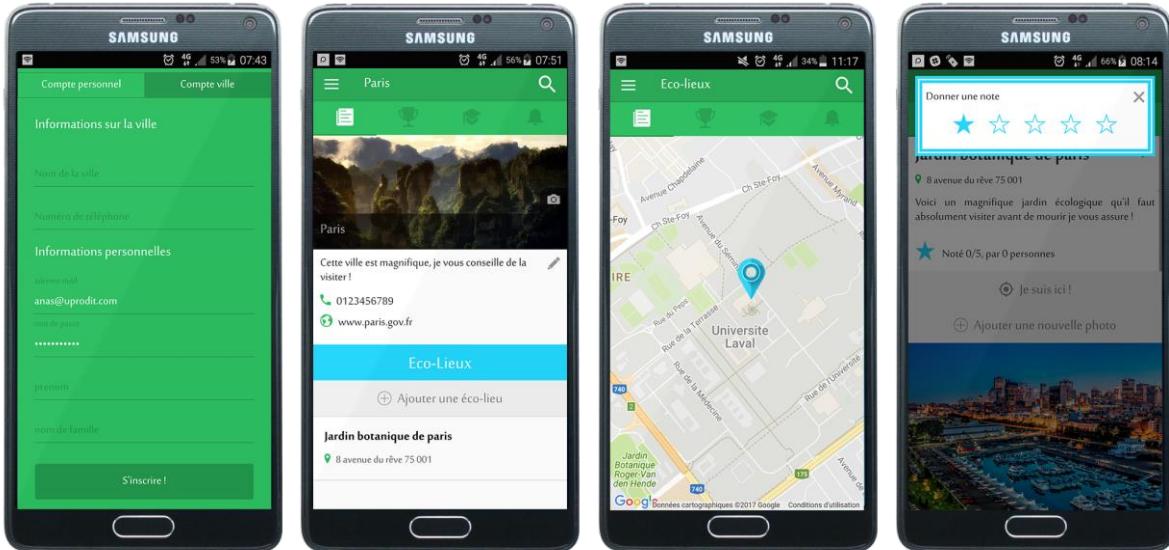


Figure 98: Interfaces liées aux éco-lieux et villes

VII. Administration et finalisation du projet

Acette étape du projet, le 1^{er} juin, toutes les fonctionnalités destinées aux utilisateurs étant terminées, il ne restait que des détails de finalisation tels que la rédaction des termes d'utilisation, politique de confidentialité ou mentions légales. Par contre, le projet étant destiné à la recherche avant tout, nous devions mettre en place le moyen de valider la théorie selon laquelle la gamification aide bien à promouvoir les gestes écoresponsables.

Afin de faciliter cette étape de validation, nous avons souhaité mettre en place pour l'administration de l'application des pages de visualisation des activités en cours sur Sustainapp. Ces pages afficheront (sous forme de graphiques ou de statistiques) certaines données calculées à partir de la base de données⁵⁹.

7.1. Planification de l'itération

Cette dernière itération a donc été réalisée du 1^{er} juin au 15 juin, ce qui permettait d'avoir encore deux semaines par la suite pour héberger et déployer tout le projet. Durant ces deux semaines voici les *user stories* réalisées ainsi que le découpage en tâches :

⁵⁹ Tutoriel de l'utilisation de l'administration : <https://www.facebook.com/cirrelt.sustainapp/videos/679156988936057/>

Tableau 11: Découpage en tâches du sprint 6

Numéro US :	21
Priorité :	01
01	Afficher les données sur les challenges
02	Afficher les données sur les cours
03	Afficher les données sur les recherches
04	Afficher les données sur les profils
05	Afficher les données sur les équipes
06	Afficher les données sur les villes et lieux

7.2. Réalisation des fonctionnalités

Voici donc les données que nous avons jugée utiles et avons donc convenu de recueillir et afficher :

- **Données sur les cours** : nombre total de cours publiés, affichage des 5 plus vus, affichage des 3 mieux notés, affichage de proportions et nombre de cours par catégories.
- **Données sur les recherches** : nombre total de recherches effectuées, affichage des 5 recherches les plus courantes, affichage du nombre de recherches par heures de la journée et par jours de la semaine.
- **Données sur les challenges** : nombre total de challenges et nombre moyen de participations par challenges, proportion et nombre de challenges par catégories, affichage des proportions et nombre de participation par heures de la journée et par jours de la semaine.
- **Données sur les profils** : nombre de profil inscrits, proportions de profils visibles et cachés, proportions et nombre de profils par niveaux, proportions et nombre de profils par âges.
- **Données sur les équipes** : nombre total d'équipes, taille moyenne en nombre de membres et proportions et nombre d'équipe par niveaux.
- **Données sur les villes et lieux** : nombre total de villes, nombre total de lieux, moyenne du nombre d'illustration par lieux, les 5 lieux les plus visités, proportions et nombre de lieux par note, proportions et nombre de visites par heures de la journée et jours de la semaine.

Pour effectuer ces affichages nous avons installé et utilisé une librairie dédiée à AngularJs appelée « *angular-chart.js* »⁶⁰. Pour obtenir les données traitées (en pourcentage, trié par heure, en moyenne ...), on a principalement utilisé les « *Map* » Java par exemple :

```
/*
 * get profiles number by level
 * @param profiles
 * @return
 */
private Map<Integer, Integer> profileByLevel(List<ProfileEntity> profiles){
    Map<Integer, Integer> result = new HashMap<Integer, Integer>();
    for(ProfileEntity profile : profiles){
        Integer level = profile.getLevel();
        if(null != result.get(level)){
            result.put(level, result.get(level)+1);
        } else {
            result.put(level, 1);
        }
    }
    return result;
}
```

Figure 99: Tri des profils par niveaux



Figure 100: Interfaces liées à l'administration

Cette itération marquant la fin du cycle de développement, le projet pouvait entrer en phase déploiement.

⁶⁰ Site officiel d'*angular-chart.js* : <http://jtblin.github.io/angular-chart.js/>



Tests et Déploiement

I. Tests et validation

Avant de commencer le déploiement à proprement parlé, nous avons décidé de mettre en place certains outils d'administration et maintenance indispensables à la survie de l'application une fois exposée aux utilisateurs au comportement parfois imprévu voire indésirable, au tentatives de piratage et aux futurs développement.

Parmi ces outils, les tests automatisés sont une pratique devenue obligatoire si l'on souhaite prétendre fournir un produit de qualité. Cette étape étant considérée aujourd'hui comme indispensable, un nouveau paradigme de développement est apparu : le TDD (*Test Driven Development*)⁶¹. En suivant ce paradigme on aurait développé le test avant même la fonctionnalité liée à celui-ci.

Ces tests sont de deux types :

- Des tests fonctionnels sans accès au code (effectués par les membres du CIRRELT) pour simplement valider les fonctions utilisateur.
- Des tests automatisés en code, unitaires (fonctionnement d'une méthode isolée du reste du code) et d'intégration (bon enchainement des méthodes entre elles). Ces tests sont indispensables car ce sont eux qui permettront lors de futurs développements de se rendre compte si il y eu une régression lors du « *build* » (compilation de la branche master du GIT) avant de déployer en exploitation.

Du fait que nous avons deux applications et qui plus est, deux applications hétérogènes en termes de technologies, nous avons dû faire deux types de tests en codes différents : certains pour Sustainapp Server et d'autres pour Sustainapp Mobile.

1.1. Création de tests pour Sustainapp Server

1.1.1. Installation de l'environnement de test

La librairie la plus utilisée pour la réalisation de tests en Java est « *Junit* »⁶² elle se couple par ailleurs avec d'autres librairies complémentaires telles que « *Mockito* »⁶³ qui permet quant à elle la création de « *Mocks* » (objets simulacres injectés et remplaçant un objet réel pour couper les appels fait pendant l'exécution de la méthode à tester. Ceci permet de limiter ainsi le test à la méthode et mieux contrôler les erreurs).

Ces deux librairies s'installent comme les autres via le fichier pom.xml de Maven :

⁶¹ TDD : https://fr.wikipedia.org/wiki/Test_driven_development

⁶² Site officiel de Junit : <http://junit.org/junit4/>

⁶³ Site officiel de Mockito : <http://site.mockito.org/>

```
<!-- Test unitaires -->
<!-- https://mvnrepository.com/artifact/junit/junit -->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.12</version>
</dependency>
<dependency>
    <groupId>org.mockito</groupId>
    <artifactId>mockito-all</artifactId>
    <version>1.9.5</version>
</dependency>
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-test</artifactId>
</dependency>
```

Figure 101: Extrait du pom.xml

La dépendance « *Spring-test* » va permettre de lier l'injection de dépendance et autre particularité de *Spring* à *Junit* et *Mockito*. Ensuite on peut commencer à utiliser *Junit* directement dans le code et pour cela on crée une architecture de package indépendante de celle des fonctionnalités de l'application



Figure 102: Architecture des packages de tests (serveur)

Nous avons effectué des tests sur les *controllers* ainsi que sur les services business, car c'est ici que l'on a le plus de code spécifiques. Les *entities*, services DAO ou classes utilitaires sont du code assez générique utilisant « *Hibernate* » ou « *Apache Commons* » et ne risquant pas d'évoluer dans le temps. Nous avons également créé une classe « *AbstractTest.java* ». Cette classe comporte en réalité tout ce qui sera utile dans toutes les classes de tests et ainsi par héritage beaucoup de répétition seront évitées.

```
/**
 * Mock de simulation d'une request
 */
@Mock
protected HttpServletRequest requestMock;

/**
 * Mock pour les services utilisé partout
 */
@Mock
protected ProfileServiceDAO profileServiceMock;
@Mock
protected CityServiceDAO cityServiceMock;
@Mock
protected UserAccountServiceDAO userAccountServiceMock;

/**
 * Mock pour les services business
 */
@Mock
protected CascadeGetService getServiceMock;
@Mock
protected CascadeDeleteService deleteServiceMock;
@Mock
protected BadgeService badgeServiceMock;

/**
 * JDD generic utile pour tout les tests
 */
protected UserAccountEntity connectedUser;
protected CityEntity city;

protected final Long GENERIC_ID = 1L;
protected final Long GENERIC_NEW_ID = 2L;
protected final Long GENERIC_UNKNOW_ID = 3L;

protected final String GENERIC_TOKEN = "x";
protected final String GENERIC_NEW_TOKEN = "y";
protected final String GENERIC_UNKNOW_TOKEN = "z";

protected final String FIRST_NAME ="anas";
```

Figure 103: Extrait de l'AbstractTest.java

1.1.2. Réalisation des tests

La réalisation d'un test selon ces technologies se fait donc en trois étapes :

1. **choix d'un jeu de données de tests (JDD)**: on souhaite choisir des données avec des données pour lesquelles le résultat des tests est connu et qui sera comparé au résultat effectivement obtenu (voir figure 103 « *AbstractTest.java* »).
2. **Initialisation des mocks et JDD** : on injecte les *mocks* et on établit des règles comportementales sous la forme « si on t'appelle avec [valeur d'entrée au choix], alors retourne [valeur à renvoyer tirée du JDD] ». Ainsi les bugs à l'intérieur du *mock* ne sont pas testés. Ces réglages se font dans des méthodes exécutées avant les tests via l'annotation « `@before` »

```
when(requestMock.getParameter("sessionId")).thenReturn(GENERIC_ID.toString());
when(requestMock.getParameter("sessionToken")).thenReturn(GENERIC_TOKEN);
when(userAccountServiceMock.getByToken(any(Long.class), any(String.class))).thenReturn(connectedUser);
```

Figure 104: Initialisation de mocks

3. **Développement d'un test** : une méthode devient un test grâce à l'annotation « `@test` ». On y appelle la méthode à tester et on observe le résultat

```
Map<String, String> errors = signinValidator.validate(requestMock);
assertNotNull(errors);
assertEquals(1, errors.size());

verify(profilServiceMock, times(0)).createOrUpdate(any(ProfileEntity.class));
verify(userAccountServiceMock, times(0)).createOrUpdate(any(UserAccountEntity.class));
verify(cityServiceMock, times(0)).createOrUpdate(any(CityEntity.class));
```

Figure 105: Vérification des résultats

1.1.3. Exécution des tests

En mode production, les tests sont exécutés à l'aide d'une commande *Maven/Spring* ce qui peut être automatisé à l'aide de *Jenkins* (tout comme le déploiement, ou l'exécution de *Flyway*). Il suffit pour cela d'ajouter la commande de lancement des tests dans le script *shell* dans la méthode par exemple de déploiement du GIT ou de créer une méthode à part et donc d'obtenir un bouton « *run tests* » dans *Jenkins*.

En mode développement, tout comme pour exécuter une application java, *Eclipse* possède déjà dans sa version JEE un bouton et une interface pour exécuter/débuguer les tests et observer graphiquement les résultats :

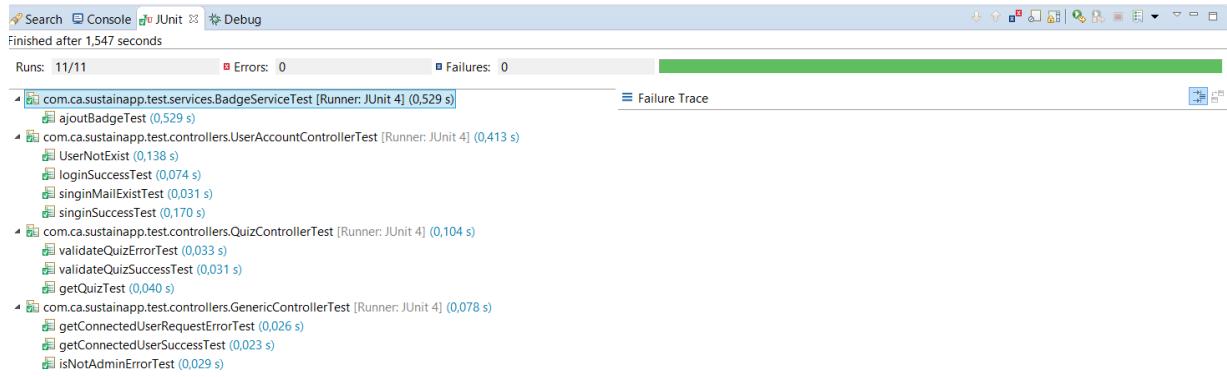


Figure 106: Junit sous Eclipse

1.2. Création de tests pour Sustainapp Mobile

Bien que les technologies changent du fait de l'utilisation *JavaScript*, la procédure est très similaire pour la réalisation d'un test.

1.2.1. Installation et configuration de l'environnement de test

La technologie utilisée pour le développement des tests de Sustainapp Mobile est « *Karma* »⁶⁴ un outil permettant d'exécuter des tests à partir d'un fichier de configuration. Cependant il faut coupler cet outil à une librairie de test du même type que *Junit* (dans notre cas c'est « *Jasmine* »⁶⁵) et à un outil de *mock* comparable à *mockito* (dans notre cas c'est « *angular-mock* »⁶⁶ qui fait partie intégrante du framework *angularJs*). Ainsi les deux technologies sont très similaires.

Comme pour toute librairie ou framework *JavaScript*, ces librairies sont installées à l'aide de *bower* ou *npm*⁶⁷. Cette installation va générer un fichier nomé « *unit-test.conf.js* » qui, comme son nom l'indique, sert à configurer l'outil karma et doit donc être modifié en fonction du projet. On y indique la librairie utilisée (*Jasmine*), le navigateur de test (dans notre cas « *Google Chrome* ») indispensable à l'interprétation du *JavaScript* ou encore les dossiers à parcourir pour obtenir toutes les dépendances dans le ordre souhaité.

⁶⁴ Site web officiel de Karma : <http://karma-runner.github.io>

⁶⁵ Site web officiel de Jasmine : <https://jasmine.github.io/>

⁶⁶ Documentation d'Angular-mock sur le site officiel d'AngularJs : <https://docs.angularjs.org/guide/unit-testing#angular-mocks>

⁶⁷ Tutoriel d'installation et configuration de Karma/Jasmine : <https://gonehybrid.com/how-to-write-automated-tests-for-your-ionic-app-part-2/>

```

module.exports = function(config) {
  config.set({
    // base path that will be used to resolve all patterns (eg. files, exclude)
    basePath: '',

    // frameworks to use
    // available frameworks: https://npmjs.org/browse/keyword/karma-adapter
    frameworks: ['jasmine'],

    // list of files / patterns to load in the browser
    files: [
      '../../lib/angular/angular.js',
      '../../lib/ionic/js/ionic.bundle.js',
      '../../lib/angular-ui-router/release/angular-ui-router.js',
      '../../lib/angular-mocks/angular-mocks.js',
      '../../lib/angular-animate/angular-animate.js',
      '../../lib/chart.js/dist/Chart.js',
      '../../lib/angular-chart.js/dist/angular-chart.js',
      '../../lib/angular-sanitize/angular-sanitize.js',
      '../../lib/angular-translate/angular-translate.js',
      '../../lib/ionic-rating/ionic-rating.js',
      '../../lib/angular-translate-loader-static-files/angular-translate-loader-static-files.js',
      '../../lib/ngCordova/dist/ng-cordova.js',
      '../../lib/ngCordova/dist/ng-cordova-mocks.js',
      '../../lib/ngDraggable/ngDraggable.js',
      '../../lib/ng-stomp/dist/ng-stomp.min.js',
      '../../lib/sockjs-client/dist/sockjs.js',
      'app.js',
      '*.js',
      '**/*.js',
      '**/**.js',
      '**/**/*.*'
    ]
  })
}

```

Figure 107: Extrait de « unit-test.conf.js »

1.2.2. Création des tests Karma/Jasmine

Ici encore, les trois étapes sont identiques : on crée des *mocks* et un JDD que l'on injecte et configure dans une méthode exécutée avant les tests

```

beforeEach(inject(function($rootScope, $controller, $httpBackend, config, $q, $stateParams, $filter, $ionicModal, $cordovaFile, $cordovaFileTransfer, $cordovaDevice, sessionService, profileService) {
  scopeMock = $rootScope.$new();
  scopeMock.profileModel = {
    "allErrors": [],
    "profileTemp": {
      "firstName": "test",
      "lastName": "test",
      "bornDate": "01/01/2017"
    },
    "profile": {
      "firstName": "befor",
      "lastName": "befor",
      "bornDate": "befor"
    }
  };
  sessionServiceMock = {
    get: jasmine.createSpy('get spy')
      .and.returnValue(1)
  };
  configMock = config;
  httpBackend = $httpBackend;
  httpBackend.whenGET('/.html').respond(function(){ return [200,""]});
  httpBackend.whenGET('/.json$').respond(function(){ return [200,""]});
  controller = $controller('profileController', {
    $scope : scopeMock,
    $stateParams : $stateParams,
    $filter : $filter,
    $ionicModal : $ionicModal,
    $cordovaFile : $cordovaFile,
    $cordovaFileTransfer : $cordovaFileTransfer,
    $cordovaDevice : $cordovaDevice,
    'sessionService' : sessionServiceMock,
    'profileService' : profileService,
    'fileService' : fileService,
    'displayService' : displayService
  });
})

```

Figure 108: Configuration des mocks en JavaScript

Après quoi, on écrit des méthodes de test qui appellent une méthode à tester et qui observent son comportement et ses résultats.

```

httpBackend.whenPOST(configMock.remoteServer+ "/profile").respond(200, result);

scopeMock.updateProfile();
httpBackend.flush();

expect(scopeMock).toBeDefined();
expect(scopeMock.profileModel.allErrors.length).toEqual(0);
expect(scopeMock.profileModel.profile.firstName).toEqual(scopeMock.profileModel.profileTemp.firstName);
expect(scopeMock.profileModel.profile.lastName).toEqual(scopeMock.profileModel.profileTemp.lastName);
expect(scopeMock.profileModel.profile.bornDate).toEqual(scopeMock.profileModel.profileTemp.bornDate);

```

Figure 109: Test unitaire en Javascript

1.2.3. Exécution des tests

Les commandes de lancement n'existent pas nativement dans Eclipse, elles seront exécutées en mode développement directement dans la console (identiquement sous Linux ou Windows) et le navigateur sera lancé automatiquement :

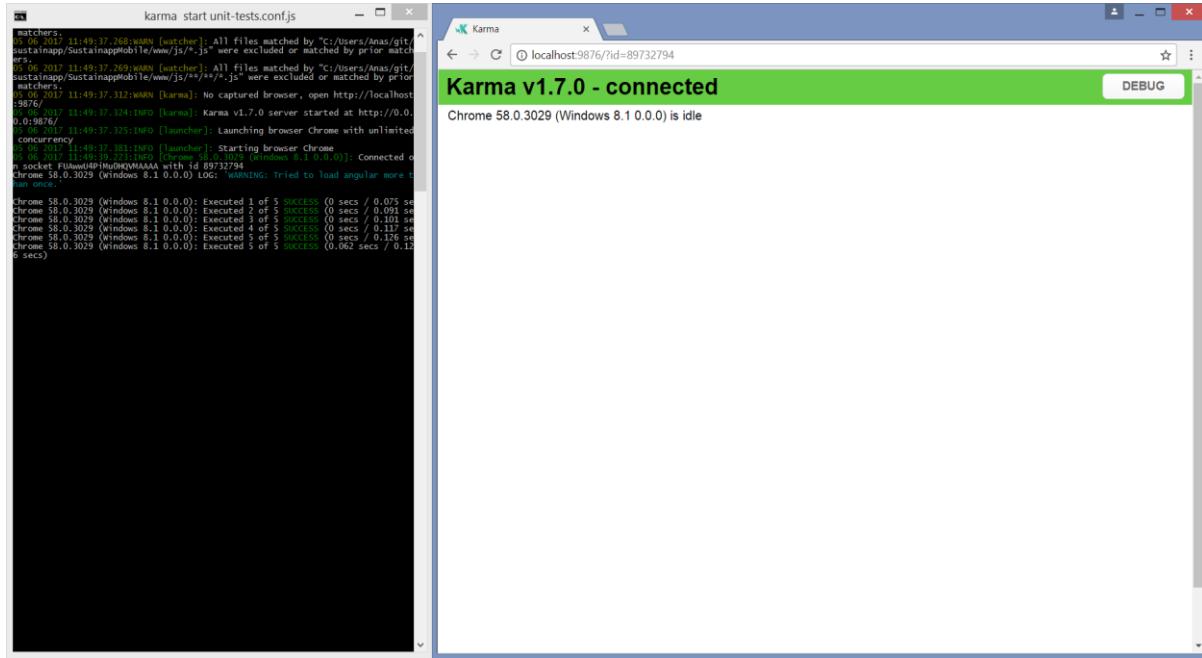


Figure 110: Exécution des tests karma/Jasmine

1.3. Validation fonctionnelle et gamification

Tous les tests précédents permettent de garantir une certaine qualité interne (au niveau du produit lui-même) mais on aimerait également valider la qualité externe du produit (celle liée à l'utilisateur et au client appelée le « *commissioning* »). Pour cela nous avons planifié deux sessions de validation.

La première s'est déroulée fin juin et a permis aux membres du CIRRELT/CeRCI (qui joue ici le rôle du client) de tester l'application pour vérifier que leurs attentes en termes de fonctionnalités et d'ergonomie étaient bien satisfaites.

La seconde aura pour but de valider la théorie selon laquelle la gamification va permettre d'intéresser et de fidéliser les utilisateurs finaux. Cette session, très importante en tant que projet de recherche, se déroulera à partir de septembre 2017 (date future à ce jour) car c'est à partir de cette date que le service juridique de l'Université Laval permettra la publication de l'application sur les fournisseurs tels que le « *Google Play Store* ». C'est également à partir de cette date, que le CIRRELT va commencer une campagne de communication autour du projet.

II. Déploiement du projet

2.1. Choix et responsabilité

La première étape du déploiement du projet a été une réflexion entre l'équipe du projet et l'équipe d'administration des équipements de l'Université Laval. En effet lors ce que l'on souhaite héberger plusieurs choix existent quant à la nature de la machine et du nom de domaine choisi. Ces choix sont à la fois stratégiques sur le plan économique, performance mais aussi juridique.

Nous avons ainsi consulté les hébergeurs en ligne *VPS*, serveurs dédiés ou des solutions *cloud*. Cependant le CIRRELT possède ses propres baies serveurs et a préféré que l'hébergement ce fasse localement plutôt que chez un hébergeur malgré que cela signifie plus de maintenance de sa part et que les performances risquent d'être moins élevées que celles d'un hébergeur. Cependant ce choix garanti pour le CIRRELT un certain contrôle sur les données et de plus cela nous a permis en outre de bénéficier des conseils et recommandations de l'équipe de l'administration des équipements lors de l'étape de l'installation.



Figure 111: Notre machine serveur au sein des baies du CIRRELT

De plus, avec cette solution, la responsabilité a été séparée en deux :

- L'équipe d'administration des équipements s'est chargée d'installer la machine serveur (Centos⁶⁸), de la configurer (redémarrage automatique en cas de panne), de l'ouverture des ports au niveau machine et réseau (principalement 22/ssh pour permettre une installation à distance, mais aussi http/https pour rendre fonctionnelle l'application), de la création des utilisateurs ou encore de la mise en place d'un VPN pour un second accès à distance en cas de problème avec le ssh.

⁶⁸ Site officiel de la distribution Centos : <https://www.centos.org/>

- Nous nous sommes occupés, sous les conseils et la surveillance de l'équipe d'administration, de l'installation de notre application et des environnements qui lui sont liés.

2.2. Installation et déploiement de l'application

Bien évidemment la 1^{ère} étape de l'installation est de se connecter au serveur en ssh via la commande « *ssh [utilisateur]@[adresse IP]* ».

2.2.1. Installation de Fail2Ban

Ensuite, le premier outil que nous avons installé est *Fail2Ban*, qui est un outil simple qui permet de bannir une IP après trois tentatives échouées (dû à un couple utilisateur/mot de passe erroné) en la retenant dans une « *blacklist* » appelée « *jail* » durant une période donnée dans le fichier « *jail.local* », rendant ainsi impossible les attaques de type « *brut force* »⁶⁹. Pour cela nous avons suivi un tutoriel complet qui décrit l'installation de *Fail2Ban* précisément sous *Centos 6*⁷⁰.

2.2.2. Protection du ssh

La troisième étape que nous avons effectué est de protéger le ssh. Initialement l'accès en ssh pouvait se faire via le compte root ce qui est une faille très dangereuse ainsi nous avons créé un nouvel utilisateur appelé « *sustainapp* » et avons retiré l'accès ssh au compte administrateur. Ainsi si l'on souhaite accéder aux privilèges root, il faudra le faire en deux étapes : accès ssh sur un compte limité puis connexion au compte administrateur.

<pre>\$ ssh root@132.203.51.130 root@132.203.51.130's password: Last failed login: Sat Jun 17 11:19:21 EDT 2017 from 181.112.118.96 on ssh:notty There were 30 failed login attempts since the last successful login. Last login: Sat Jun 17 07:19:15 2017 from 80.12.34.240 [root@cir-762628 ~]# useradd sustainapp [root@cir-762628 ~]# passwd sustainapp Changing password for user sustainapp. New password: Retype new password: passwd: all authentication tokens updated successfully.</pre>	<pre>[root@cir-762628 ~]# cat /etc/ssh/sshd_config grep -i permitroot #PermitRootLogin yes # the setting of "PermitRootLogin without-password". [root@cir-762628 ~]# vim /etc/ssh/sshd_config [root@cir-762628 ~]# cat /etc/ssh/sshd_config grep -i permitroot PermitRootLogin no # the setting of "PermitRootLogin without-password". [root@cir-762628 ~]# systemctl restart sshd</pre>
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Figure 112: Sécurité de l'accès ssh

2.2.3. Installation du SGBD

Dans notre cas, il s'agit de *PostgreSQL* et nous avons suivi pour cela un cours qui explique justement comment installer *PostgreSQL* sous *Centos 7*⁷¹. Comme pour *Fail2Ban* l'installation se fait avec le gestionnaire de dépendance « *yum* » qui par ailleurs peut être comparé à un *npm* ou *maven* à la différence qu'il ne gère pas des librairies mais plutôt des logiciels. Ensuite, la configuration de *PostgreSQL* comme celle de tout logiciel sous Linux se fait par modification (avec vi ou autre) de fichiers

⁶⁹ **Brut Force** : Essai de toutes les combinaisons possibles jusqu'à trouver la bonne

⁷⁰ Tutoriel d'installation de *Fail2Ban* sous *Centos 7* : <https://www.digitalocean.com/community/tutorials/how-to-protect-ssh-with-fail2ban-on-centos-7>

⁷¹ Tutoriel d'installation de *PostgreSQL* sous *Centos 7* : <https://www.digitalocean.com/community/tutorials/how-to-install-and-use-postgresql-on-centos-7>

de « *setting* » (de configuration). Une fois installé, on peut accéder aux commandes spécifiques de PostgreSQL (« *start* », « *stop* ») et on peut accéder aux données avec « *psql* », une interface permettant d'exécuter des commandes basées sur le « *SQL* ».

```
C:\Users\Anas\Desktop>ssh sustainapp@132.203.51.130
sustainapp@132.203.51.130's password:
Last login: Mon Jul 17 03:04:42 2017 from 154.110.170.59
[sustainapp@cir-762628 ~]$ su
Password:
[root@cir-762628 sustainapp]# su - postgres
Last login: Mon Jul 17 03:00:00 EDT 2017 on pts/1
Last failed login: Mon Jul 17 03:03:58 EDT 2017 on pts/2
There were 2 failed login attempts since the last successful login.
-bash-4.2$ psql sustainapp
psql (9.2.18)
Type "help" for help.

sustainapp=# \d user_account
                                         Table "public.user_account"
   Column    |          Type          |           Modifiers
-----+-----+-----+
 id      | integer            | not null default nextval('user_acco
unt_id_seq'::regclass)
 password | character varying(250) | not null
 token    | character varying(250) | default NULL::character varying
 is_admin | boolean             | default false
 timestamps | timestamp without time zone |
 mail     | character varying(250) | not null
 user_type | integer             | default 0
 token_delay | timestamp without time zone |
Indexes:
    "user_account_pkey" PRIMARY KEY, btree (id)
Referenced by:
    TABLE "city" CONSTRAINT "fk_city_user_account_id" FOREIGN KEY (user_account_id)
        REFERENCES user_account(id)
    TABLE "profile" CONSTRAINT "fk_profile_user_account_id" FOREIGN KEY (user_account_id)
        REFERENCES user_account(id)
sustainapp=#

```

Figure 113: Accès en psql à la base

Nous aurions également pu installer une interface web/graphique pour l'accès à la base telle que PHPPgAdmin III⁷² pour plus d'ergonomie mais nous ne l'avons pas fait à des fins d'économie de mémoire RAM.

2.2.4. Récupération du code

Pour récupérer le code sur le serveur plusieurs méthodes sont possibles, on utilise souvent le protocole FTP et des outils comme FileZilla pour cela. Mais dans notre cas, code étant déjà en ligne au sein d'un « *repository GIT* », il suffit de cloner en local le *repository*. Pour cela nous avons en premier lieu créé un dossier que l'on a appelé « *BACK_SUSTAINAPP* » (avec la commande « *mkdir* » puis « *chmod* » pour les droits).

Après cela on effectue les commandes suivantes : « *yum install -y git dos2unix maven* » pour l'installation de maven puis :

⁷² PHPPgAdmin 3 sur sourceforge : <http://phppgadmin.sourceforge.net/doku.php>

```
[root@cir-762628 ~]# cd /BACK_SUSTAINAPP/  
[root@cir-762628 BACK_SUSTAINAPP]# git clone http://91.121.80.56:8181/gitlab/isamm/sustainapp.git export_git  
Cloning into 'export_git'...  
Username for 'http://91.121.80.56:8181': sustainapp  
Password for 'http://sustainapp@91.121.80.56:8181':  
[root@cir-762628 BACK_SUSTAINAPP]# cd export_git/  
[root@cir-762628 export_git]# git config credential.helper store  
[root@cir-762628 export_git]# cp SustainappBatch/src/main/resources/shell/sustainapp_indus.sh /BACK_SUSTAINAPP/install.sh  
[root@cir-762628 export_git]# chmod +x /BACK_SUSTAINAPP/install.sh
```

Figure 114: Clonner un git

Après avoir récupéré le repository, on a copié à la racine du dossier « *BACK_SUSTAINAPP* » le fichier shell qui contient toutes les méthodes d'appel aux commandes pour déployer *SustainappServer* ou lancer *Flyway*.

2.2.5. Installation de Varnish

Pour *SustainappMobile*, on avait deux choix : utiliser *Ionic* comme serveur directement et lancer l'application avec les commandes *Ionic* ou bien utiliser un serveur comme *Varnish*⁷³, ou *Nginx* qui servirait de *reverse-proxy* et pointerait directement sur le dossier « *www* » du code de *SustainappMobile*. C'est cette 2^e option que nous avons choisi du fait des performances qu'offre Varnish. Voici les commandes d'installation et de lancement :

```
[root@cir-762628 ~]# yum install varnish  
[root@cir-762628 ~]# vim /etc/varnish/default.vcl  
[root@cir-762628 ~]# ps -ef|grep -i varnish|awk '{print $2}'|xargs kill -9  
[root@cir-762628 ~]# varnishd -f /etc/varnish/default.vcl
```

Figure 115: Installation et lancement de Varnish

Pour que Varnish pointe sur le dossier *www* de *SustainappMobile*, il faut configurer le fichier « *default.vcl* ».

2.2.6. Installation de Jenkins

Nous allons comme prévu utiliser Jenkins pour déployer le GIT et la base de données sans avoir à ré-exécuter de commandes. Pour cela il faut avant tout l'installer :

```
[root@cir-762628 ~]# yum install jenkins  
[root@cir-762628 ~]# vim /etc/sysconfig/jenkins  
[root@cir-762628 ~]# systemctl start jenkins  
[root@cir-762628 ~]# cat /var/lib/jenkins/secrets/initialAdminPassword
```

Figure 116: Installation de Jenkins

Nous avons démarré Jenkins via « *systemd/systemctl* » ce qui ne fonctionne que sur *Linux*. Cependant Jenkins étant une application JEE comme un autre (un fichier « *war* ») elle peut être démarrée ou chargée sur un serveur container. Une fois démarré, la configuration de Jenkins ce fait entièrement via interface graphique ce qui la rend facile et rapide. Il suffit de créer un « *item* » (nouveau bouton) et de le configurer pour qu'il appelle le script *shell* du projet.

⁷³ Site officiel de Varnish : <https://www.varnish-cache.org/>

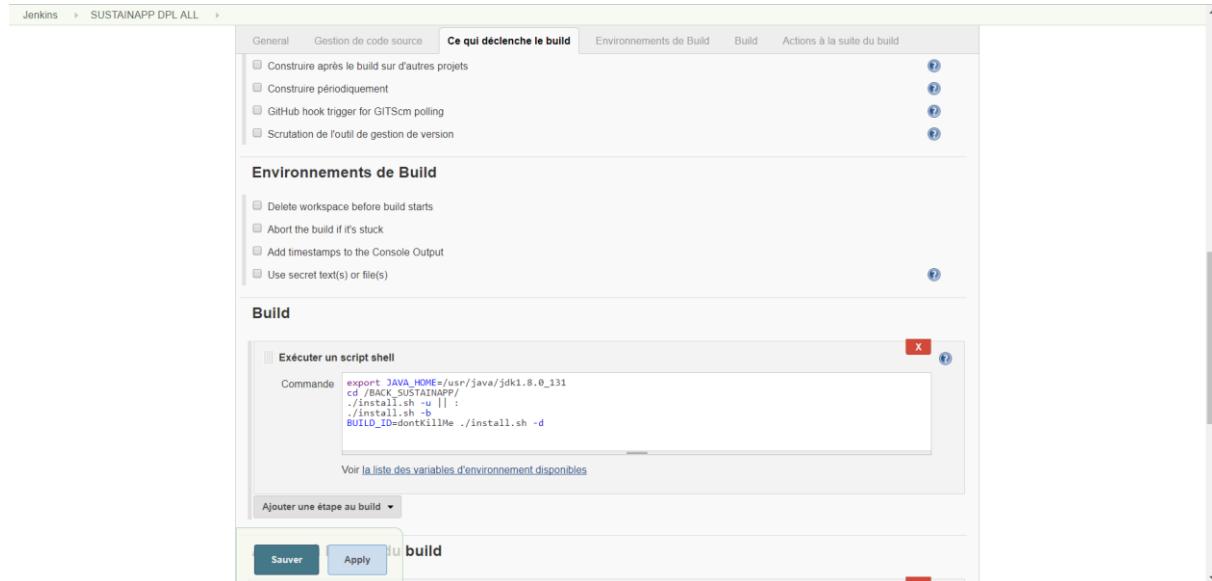


Figure 117: Configuration graphique de Jenkins

The screenshot shows the Jenkins dashboard with three projects listed:

Nom du projet	Dernier succès	Dernier échec	Dernière durée
SUSTAINAPP DPL ALL	20 h - #60	s. o.	8.7 s
SUSTAINAPP RESTART WEBSERVICES	24 j - #1	s. o.	0.92 s
SUSTAINAPP UPDATE DATABASE	24 j - #4	s. o.	4.2 s

Icons for each job status are shown to the left of the names. The 'File d'attente des constructions' and 'État du lanceur de compilations' sections are also visible.

Figure 118: Interface de Jenkins configuré avec les jobs

L'installation est maintenant terminée et les outils qui vont suivre sont des outils auxiliaires qui aident à la maintenance du projet.

2.3. Outils de maintenance

2.3.1. Le système du ping/pong

Le premier mécanisme que nous avons mis en place est très simple : il s'agit de vérifier en cas de non réponse de l'application si c'est une fonctionnalité qui provoque un problème ou si effectivement le serveur (*Varnish* ou *Spring Boot*) est « tombé » et qu'il faut donc le relancer. Pour cela on crée une URL « */ping* » qui répond tout simplement « *pong* », indépendante de tout et sans aucun code particulier, si cette URL ne répond pas alors effectivement le serveur est éteint.

```
/*
 * Test de l'existence de l'application
 * @param request
 * @return
 */
@ResponseBody
@RequestMapping(value="/ping", method = RequestMethod.GET, produces = SustainappConstantes.MIME_TEXT)
public String ping(HttpServletRequest request){
    return "pong\n";
}
```

Figure 119: Méthode ping/pong

2.3.2. Création du boot slack

Le deuxième mécanisme est beaucoup plus complexe mais est extrêmement utile à la maintenance voire indispensable. Une fois l'application en production sur le serveur, si un problème survient (qui donc a échappé aux tests automatisés), nous n'avons pas accès à la console d'Eclipse pour connaître la nature du problème (exception survenue, ligne ayant déclenché le problème, date/heure du déclenchement du problème ...) ainsi il est très difficile de « *debugger* ». Tout ce que l'on peut vérifier c'est l'existence en mémoire du serveur avec la méthode ping/pong.

Cependant Spring possède un mécanisme permettant de capturer toute erreur survenue : les « *@ControllerAdvice* » et ainsi d'avoir accès à la « *stacktrace* » et d'en faire ce que l'on veut. Dans notre cas on souhaite la recevoir sur notre téléphone en temps réel grâce à l'application « *Slack* »⁷⁴ ⁷⁵

Pour cela la procédure est simple :

1. créer un boot sur slack via l'interface graphique et obtenir ainsi un url discussion / token statique du boot
2. créer une classe de communication avec slack dans le projet SustainappServer :

⁷⁴ Site officiel de Slack : slack.com

⁷⁵ Téléchargement de Slack sur le Play Store : <https://play.google.com/store/apps/details?id=com.Slack&hl=fr>

```

@Component
public class SlackSender {
    private static final Logger LOGGER = LoggerFactory.getLogger(SlackSender.class);
    private static final String SLACK_ERROR_LOG_MSG = "Erreur à l'envoie du slack...";

    @Value("${slack.token}")
    private String slackAuthToken;

    @Value("${slack.channel}")
    private String slackChannel;

    /**
     * Envoyer un message de prod.
     * @param message
     */
    public void sendProdMessage(String message) {
        sendMessageQuietly(slackAuthToken, slackChannel, message);
    }

    /**
     * Envoyer un message sur slack
     * @param token
     * @param channel
     * @param message
     */
    public static void sendMessageQuietly(String token, String channel, String message) {
        if (isBlank(token) || isBlank(channel)) {
            LOGGER.warn("Token or channel is blank : nothing to do");
            return;
        }
        try {
            SlackSession session = SlackSessionFactory.createWebSocketSlackSession(token);
            session.connect();
            session.sendMessage(session.findChannelByName(channel), message);
            session.disconnect();
        } catch (Exception e) {
            LOGGER.error(SLACK_ERROR_LOG_MSG, e);
        }
    }
}

```

Figure 120: Envoi d'un message sur slack

3. créer un *Advice Spring* pour capturer les erreurs (sauf celles que l'on ajoute à une « *blacklist* ») et pour appeler le singleton d'envoi d'un message sur *Slack*.

```

/**
 * Catcher les erreurs pour envoyer dans slack
 * @param ex
 * @param request
 */
@ExceptionHandler(Exception.class)
@ResponseBody
public String handleAllException(Exception ex, HttpServletRequest request) {

    /**
     * Le message d'erreur
     */
    String errorMessage = ex.getMessage();
    String errorType = ex.getClass().getSimpleName();
    String stackTrace = getStackTrace(ex);
    LOGGER.error(String.format(ERR_MSG, errorMessage, errorType), ex);

    /**
     * Erreurs non supportées
     */
    if (FILTERED_ERRORS_MSG.contains(errorMessage)) {
        return new HttpRESTfullResponse().setCode(0).buildJson();
    }

    /**
     * Le user responsable
     */
    UserAccountEntity user = super.getConnectedUser(request);
    String profileName = (null == user) ? StringUtil.EMPTY : new LightProfileResponse(user.getProfile()).getDenomination();
    String mail = (null == user) ? StringUtil.EMPTY : user.getEmail();
    String strMsg = String.format(MAIL_CONTENT, mail, profileName, errorMessage, errorType, stackTrace);

    /**
     * Envoi sur slack
     */
    LOGGER.error(strMsg);
    slackSender.sendProdMessage(strMsg);
    return new HttpRESTfullResponse().setCode(0).buildJson();
}

```

Figure 121: Un Advice de Spring pour les erreurs

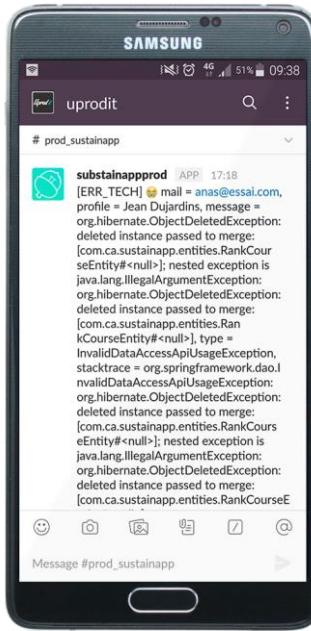


Figure 122: Réception d'une erreur sur Slack

2.3.3. Crédit d'une documentation serveur

Enfin, le dernier outil que nous ayons mis en place est la documentation complète du serveur pour les nouveaux venus dans l'équipe future qui pourrait être constituée autour du projet.

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/mapper/cl-root	50G	4.6G	46G	10%	/
devtmpfs	3.8G	0	3.8G	0%	/dev
tmpfs	3.8G	148K	3.8G	1%	/dev/shm
tmpfs	3.8G	9.1M	3.8G	1%	/run
tmpfs	3.8G	0	3.8G	0%	/sys/fs/cgroup
/dev/sda1	1014M	183M	832M	18%	/boot
/dev/mapper/cl-home	407G	78M	407G	1%	/home
tmpfs	776M	24K	776M	1%	/run/user/1000
tmpfs	776M	0	776M	0%	/run/user/0

Partition / : Il y a beaucoup de paquets inutiles pour un serveur web installé, notamment toute la partie desktop très consommatrice d'espace disque (tout les paquets GNOME, LibreOffice, X, etc), et bien d'autres.

Sécurité SSH et utilisateurs du serveur

Nous avons fait en sorte que seul les utilisateurs substainapp et cirrelt (utilisateurs sans aucun droits root) puissent se connecter en SSH.

Recommendation 1 : Il faudrait aller encore plus loin en utilisant un compte par utilisateur susceptible de se connecter sans aucun droits (c'est le cas pour substainapp) et utiliser des clefs ssh RSA ou DSA nominatives.

Nous avons également installé fail2ban afin de bannir les ips des bots qui tentent de se connecter en ssh.
=> <https://www.digitalocean.com/community/tutorials/how-to-protect-ssh-with-fail2ban-on-centos-7>

Recommendation 2 : Mettre les ips sur dans la whitelist pour ne pas être bannis en cas d'erreur

Ces mesures de sécurité permettront à la fois de laisser la possibilité de se connecter en SSH à distance aux parties prenantes de l'application tout en garantissant une sécurité optimale (il faudrait à la fois faire fuser la clef SSH d'un des utilisateurs qui n'auroit comme droit que de pouvoir se connecter, et le mot de passe root).

Consommation RAM et CPU

Il est inutile de démarrer le serveur comme un PC desktop avec une session bureau

Figure 123: Documentation du serveur

L'application mobile et la version web étaient totalement fonctionnelles et maintenables, le projet était donc terminé. La dernière tâche que nous avons effectuée ne relève donc pas directement du projet mais plutôt de sa communication.

III. Communication

3.1. Utilisation des plateformes sociales

Bien que la véritable campagne de communication de ne débutera que début septembre 2017, nous avons mis en place les outils de communication de base : la création de page dans les réseaux sociaux.

Pour cela nous avons tout d'abord crée des vidéos tutoriels pour chaque fonctionnalité en utilisant le logiciel de montage « *Camtasia* »⁷⁶ :

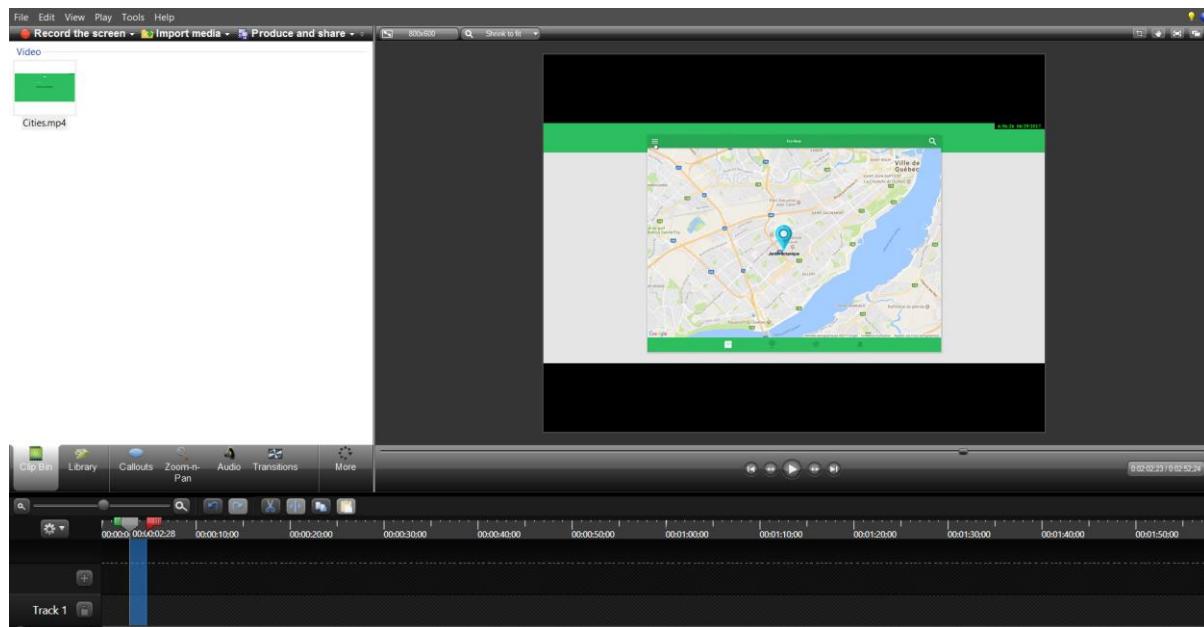


Figure 124: Edition d'un tutoriel sur Camtasia

Ensuite on a créé une page sur Facebook⁷⁷ sur laquelle l'application est décrite. On a partagé sur cette même page des images de l'application et les vidéos créées.

⁷⁶ Site officiel de Camtasia : <https://www.techsmith.com/video-editor.html>

⁷⁷ Url de la page Facebook de Sustainapp : <https://www.facebook.com/cirreit.sustainapp/>

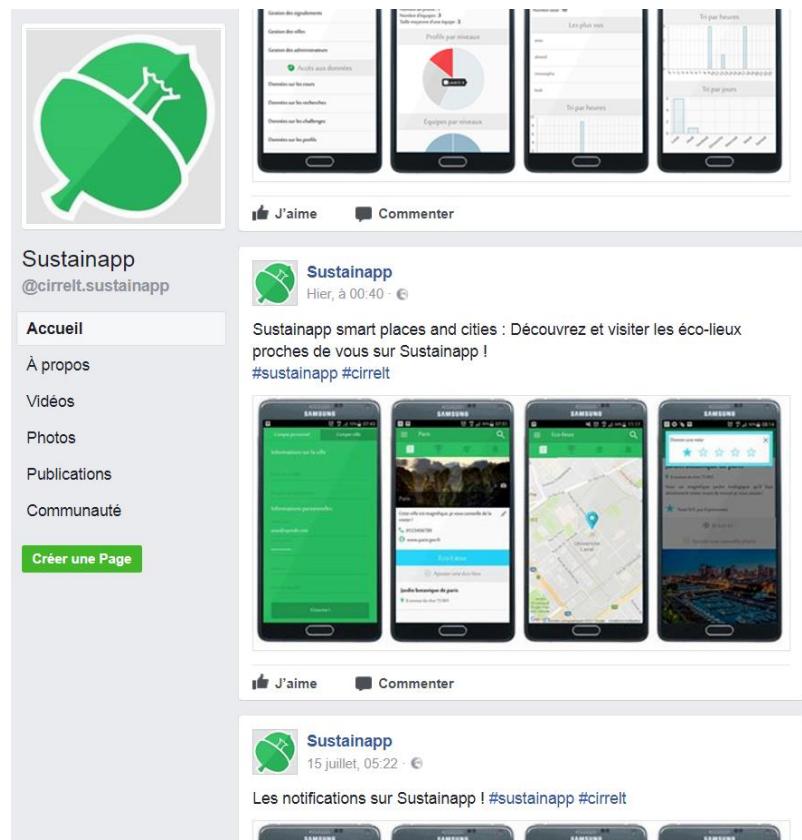


Figure 125: Page Facebook de Sustainapp

3.2. Communication directe

Les membres du CeRCI/CIRRELT qui ont encadré le projet, tous les enseignants notamment à l'Université Laval, comptent utiliser la plateforme au sein de leurs cours avec leurs étudiants à partir de septembre afin de familiariser ces derniers à l'utilisation de Sustainapp.

Nous espérons obtenir de la part des étudiants un maximum de remarques afin de pouvoir améliorer par la suite l'application avant de la communiquer en dehors de l'Université.



Conclusion

I. Bilan du projet

Ce projet avait un objectif principal : promouvoir les gestes écoresponsables des citoyens d'une ville dite intelligente. En effet, c'est de ces grandes villes qu'une grande partie de la pollution provient et ces villes ont aujourd'hui la volonté et la capacité d'agir contre cela. Mais il est également important que chaque citoyen adhère lui aussi à cette volonté et agit activement. L'environnement et le développement durable sont malheureusement des thèmes d'actualité de par la rupture des accords de Paris par certains pays et de par la crise que prévoient de nombreux scientifiques.

De nombreuses solutions sont mises en place et notamment informatiques. Le CIRRELT travail par exemple à l'analyse des opinions des citoyens partagées sur les réseaux sociaux et particulièrement Twitter afin de comprendre si la population est bien informée et sensibilisée pour ainsi décider des actions à entreprendre par la suite. Ainsi le CIRRELT a décidé de créer un produit utilisant des théories ayant fait leurs preuves pour appuyer cette démarche : Sustainapp.

Ce projet a également une dimension purement technologique : la création d'un projet informatique, maintenable et durable ainsi qu'à long terme un objectif scientifique : tester le concept de « *gamification* » sur le domaine de l'environnement et valider son apport. Cependant cette validation n'a pas pu se faire durant la période du stage du fait de l'attente d'un accord juridique avant la publication de l'application ainsi que l'attente du retour de vacances des étudiants en septembre.

1.1. Bilan scientifique

Sur le plan personnel, étant plus habitué aux projets d'entreprises, ce projet représente ma première expérience dans le monde de la recherche. J'ai eu l'occasion de profiter de l'expérience des membres du laboratoire et de découvrir les outils de recherche de travaux scientifiques, les normes de citation, l'écoute de conférences ou encore le plan type d'un article.

La recherche relative à l'état de l'art m'a également permis de découvrir et d'approfondir des notions scientifiques et sociales telles que la protection de l'environnement, le concept de ville intelligentes mais surtout la gamification et de réfléchir aux liens que l'on peut établir entre elles.

En termes d'apport scientifique, ce projet est le premier réseau social liant les trois notions et permettra une fois lancé officiellement de juger scientifiquement la pertinence de la gamification dans ce domaine.

1.2. Bilan technologique

Pareillement, ce projet fut l'occasion pour moi d'apprendre de nouvelles notions purement informatiques cette fois telles que les Framework « *Spring Boot* », « *AngularJs* » ou « *Ionic* ». Certains défis ont également eu à être relevés durant la réalisation et l'hébergement du projet, notamment la géolocalisation, l'utilisation de la technologie websocket, les échanges REST, l'utilisation de librairies *JavaScript* (création de graphique, note avec des étoiles, drag & drop), le traitement d'images ou encore le déploiement même d'une plateforme sur un serveur.

Durant les réunions de conception et d'analyse des besoins d'autres idées sont apparues et n'ont malheureusement pas été retenues par l'équipe pour des raisons de temps comme l'utilisation

de l'« *IOT* »⁷⁸ pour récolter des informations dans la ville intelligente ou encore l'utilisation des « *Big Data* » provenant des réseaux sociaux mais seront peut-être proposées de nouveau dans une future version du projet.

1.3. Bilan social

Enfin, sur le plan social, réaliser un stage à l'étranger est une expérience sociale riche de par la découverte d'une autre culture. De plus, l'expérience du laboratoire est également une expérience sociale singulière, on y apprend à défendre ses idées durant les réunions et à apprendre de personnes provenant de milieux très divers (gestion, économie, science naturelle ou informatique) et ayant donc des points de vue très différents.

II. Perspectives

2.1. Perspectives du projet

Comme nous l'avons vu, l'avenir du projet dépendra essentiellement des résultats obtenus à partir de septembre. Cependant, il est déjà prévu qu'une 2nd version soit conçue et développée à partir de mars 2018. L'équipe du CeRCI m'ont demandé de revenir à cette date afin de gérer cette évolution, qui cette fois ne sera pas un travail solitaire mais le fruit d'une équipe cette constituée de plusieurs développeurs.

Ce sont donc les orientations qui restent à déterminer mais l'intégration du projet dans le monde des réseaux sociaux et la coopération avec l'administration de villes pour la mise en place de réseaux de capteurs liés à la plateforme sont des idées qui seront proposées.

La participation des villes, non en tant que simple profils de l'application mais plutôt comme partenaires du projet, permettrait d'accroître sans doute l'envergure du projet. C'est pour cela que l'équipe du projet à Laval pense commencer par proposer à l'administration de Québec de s'investir dans le projet.

2.2. Perspectives personnelles

Personnellement, ayant apprécié le projet et l'environnement, je compte bien poursuivre et contribuer dans le projet à partir de mars 2018. D'autant plus que les axes proposés pour la suite sont des domaines techniques qui me sont jusqu'ici inconnus et lesquels j'aimeraï approfondir.

D'un autre côté, à part l'aspect technologique, j'ai remarqué que le thème des gestes d'écoresponsable est un thème d'actualité en Tunisie notamment avec la création récente d'une police de l'environnement ou encore l'arrêt des sacs en plastique dans les grandes surfaces commerciales. J'ai

⁷⁸ **Internet Of (Every) Things** : Utilisation d'objets connectés à internet et capables d'échanger des messages (signaux de capteurs principalement)

donc pensé à réaliser un projet dans ce sens par la suite, lié ou indépendant de *Sustainapp* qui aurait pour but d'impacter positivement la conscience environnementale en Tunisie.

Bibliographie

- Alaa, A., Gary, B., Vanissa, W., & Ashok, R. (2014). Towards a Sustainable Gamification Impact. *International Conference on Information Society*.
- Aparicio, A., Vela, F., Sánchez, J., & Montes, J. (2012, October). Analysis and application of gamification. *Proceedings of the 13th International Conference on Interacción Persona-Ordenador. ACM*, 17.
- Berger, V., & Schrader, U. (2016). Fostering Sustainable Nutrition Behavior through Gamification. *Sustainability*, 8(1), 67.
- Danae, V., Enric, M., Sergio, G., Alba, T., & Simón, L. (2011). MEECO: GAMIFYING ECOLOGY THROUGH A SOCIAL NETWORKING PLATFORM. *Multimedia Creation, Design and Engineering Master's Degree*.
- Flores, E., Montoya, M., & Mena, J. (2016, November). Challenge-based gamification as a teaching'open educational innovation strategy in the energy sustainability area. *Proceedings of the Fourth International Conference on Technological Ecosystems for Enhancing Multiculturality. ACM*.
- Heilbrunn, B., Herzig, P., & Schill, A. (2014, December). Tools for gamification analytics : A survey. *Utility and Cloud Computing (UCC), 2014 IEEE/ACM 7th International Conference. IEE*, (pp. 603-608).
- Kazhamiakin, R., Marconi, A., Perillo, M., Pistore, M., Valetto, G., Piras, L., & Perri, N. (2015, October). Using gamification to incentivize sustainable urban mobility. *Smart Cities Conference (ISC2), 2015 IEEE First International*, (pp. 1-6).
- Liu, D., Santhanam, R., & Webster, J. (2016). *Towards meaningful engagement : A framework for design and research of gamified information systems*. SSRN, 2521283.
- Nam, T., & Pardo, T. A. (2011, June). Conceptualizing smart city with dimensions of technology, people, and institutions. *Proceedings of the 12th annual international digital government research conference: digital government innovation in challenging times. ACM*, 282-291.
- Nassiri, A., Baudet, C., & Termins, F. (May 2017). De la compléxité de la notion de gamification à la compléxité de sa mise en oeuvre : une étude exploratoire dans une contexte d'application mobile touristique. *22eme colloque de l'AIM "Faire face à la compléxité dans un monde numérisé", Paris, France, AIM*.
- Tolmie, P., Chamberlain, A., & Benford, S. (2014). Designing for reportability : sustainable gamification, public engagement, and promoting environmental debate. *Personal and Ubiquitous Computing*, 18(7), 1763-1774.
- Vara, D., Macias, E., Gracia, S., Torrents, A., & Lee, S. (2011, July). Meeco : Gamifying ecology through a social networking platform. *Multimedia and Expo (ICME), 2011 IEEE International Conference*, (pp. 1-6).
- Xu, Y. (2011). *Literature review on web application gamification and analytics*. Honolulu: HI, 11-05.

Table des figures

Figure 1 : Logo du CIRRELT	8
Figure 2: Instituts membres du CIRRELT	8
Figure 3 : Logo actuel de l'Université Laval.....	9
Figure 4 : l'Université Laval et locaux du CIRRELT	9
Figure 5: Axes d'interventions relatifs au développement durable	10
Figure 6: Réunions Mixtes.....	11
Figure 7: Module de gestion de projet de Uprod'it.....	12
Figure 8: Interface de l'outil de recherche d'articles.....	14
Figure 9: Dérivées de la gamification	17
Figure 10: La pyramide de Maslow	17
Figure 11: Les 6 mécanismes de la gamification	18
Figure 12: Théorie du Flow	18
Figure 13 : Les types de joueurs de Bartle	19
Figure 14: Framework de conception gamifiée	19
Figure 15: Les 3 composants d'une ville intelligente.....	21
Figure 16: Interface du jeu GreenGame	23
Figure 17: Interface de Mecco.....	24
Figure 18: Interface Tri En Folie.....	25
Figure 19: Interface Anno 2070.....	25
Figure 20 : Interface Botanica.....	25
Figure 21: Interface Oligarchie	25
Figure 22: Site web d' Ingress.....	26
Figure 23: Les clubs dans Duolingo	27
Figure 24: Les niveaux dans Duolingo	27
Figure 25: Les quiz dans Duolingo	27
Figure 26: Interfaces du Projet Voltaire	28
Figure 27: L'effet réseau sur l'utilisation du téléphone	29
Figure 28: Les acteurs de Sustainapp	32
Figure 29: Diagramme de packages de Sustainapp	33
Figure 30: Cas d'utilisation 1 : profils & équipes	34
Figure 31: Cas d'utilisation 2 : challenges & cours	35
Figure 32 : Cas d'utilisation 3 : éco-lieux et actualités	37
Figure 33: Cas d'utilisation 4 : administration	38
Figure 34 : Les facteurs de qualité de la norme ISO 9126.....	38
Figure 35: Séquence réception d'une notification temps réel	41
Figure 36: Séquence affichage d'un éco-lieu.....	42
Figure 37: Séquence gestion des rôles d'une équipe	43
Figure 38: Diagramme de classe de Sustainapp	44
Figure 39: Architecture de Sustainapp.....	46
Figure 40: Architecture interne de Sustainapp Server	47
Figure 41: Architecture interne de Sustainapp Mobile/Web.....	48
Figure 42 : Création initiale du logo	50
Figure 43: Icônes de Sustainapp.....	50
Figure 44: Adaptation du logo de L'université Laval	51
Figure 45: Organisation des pages	51
Figure 46: Interface de Gitlab.....	58
Figure 47 : Interface de Github Desktop	58

Figure 48: Contenu du repository GIT	59
Figure 49: Packages de SustainappServer	60
Figure 50: SustainappServer.java	61
Figure 51: Exemple d'Entity	62
Figure 52 : Exemple de service DAO.....	63
Figure 53: Exemple de controller	64
Figure 54: Exemple de validator	64
Figure 55: Classe générique de réponse d'un controller	65
Figure 56: Architecture de Sustainapp Mobile	66
Figure 57: Installation d'une librairie via Bower	67
Figure 58: Configuration de l'i18n	68
Figure 59: Sécurité des liens	68
Figure 60: Système de routing.....	69
Figure 61: Architecture de Sustainapp Batch	70
Figure 62: Exemple de migration SQL	71
Figure 63: Création d'un "Job Maven" basé sur Flyway	71
Figure 64: sustainapp_indus.sh	72
Figure 65: Cryptage en MD5 du mot de passe	73
Figure 66: Interfaces de connexion et d'inscription	74
Figure 67: Connexion en version web	74
Figure 68: Accès à la caméra du téléphone	75
Figure 69: Compression d'une image.....	76
Figure 70: Signaler un abus.....	76
Figure 71: Interfaces de modification du profil	78
Figure 72: Récupération paginée des équipes.....	79
Figure 73: Interfaces liées aux équipes	79
Figure 74: Les catégories pour filtrer	81
Figure 75: Interfaces liées aux challenges.....	81
Figure 76: Icônes pour l'ajout d'une partie dans un chapitre	82
Figure 77: Icônes de filtrage par langue.....	83
Figure 78: Les états possibles pour un chapitre	83
Figure 79: Affichage et sauvegarde en base du déplacement d'un chapitre	84
Figure 80: Construction d'un lien pour l'affichage d'une vidéo Youtube	85
Figure 81: Interfaces liées aux cours	85
Figure 82: Icône de choix du type de question.....	86
Figure 83: Interfaces liées aux quizz.....	87
Figure 84: Interface web des Quizz	87
Figure 85: Utilisation de Hql pour lancer des recherches paginées	89
Figure 86: Interfaces liées à la recherche	89
Figure 87: Interface du journal de Sustainapp.....	90
Figure 88: Icônes des différents badges	90
Figure 89: Interfaces liées aux badges	91
Figure 90: Icônes des différentes notifications.....	91
Figure 91: Mise en place des websockets sur Spring Boot	92
Figure 92: Envoi d'une websocket sur Spring Boot	92
Figure 93: Génération d'une notification native	92
Figure 94: Interfaces liées aux notifications	93
Figure 95: Calculer une position à partir d'une adresse	94
Figure 96: Installation du plugin de géolocalisation	95
Figure 97: Récuperer la position de l'utilisateur.....	95
Figure 98: Interfaces liées aux éco-lieux et villes	96
Figure 99: Tri des profils par niveaux	98

Figure 100: Interfaces liées à l'administration.....	98
Figure 101: Extrait du pom.xml	101
Figure 102: Architecture des packages de tests (serveur).....	101
Figure 103: Extrait de l'AbstractTest.java	101
Figure 104: Initialisation de mocks.....	102
Figure 105: Vérification des résultats.....	102
Figure 106: Junit sous Eclipse	103
Figure 107: Extrait de « unit-test.conf.js ».....	104
Figure 108: Configuration des mocks en JavaScript	104
Figure 109: Test unitaire en Javascript.....	104
Figure 110: Exécution des tests karma/Jasmine.....	105
Figure 111: Notre machine serveur au sein des baies du CIRRELT.....	106
Figure 112: Sécurité de l'accès ssh.....	107
Figure 113: Accès en psql à la base	108
Figure 114: Clonner un git	109
Figure 115: Installation et lancement de Varnish.....	109
Figure 116: Installation de Jenkins	109
Figure 117: Configuration graphique de Jenkins	110
Figure 118: Interface de Jenkins configuré avec les jobs	110
Figure 119: Méthode ping/pong	111
Figure 120: Envoi d'un message sur slack.....	112
Figure 121: Un Advice de Spring pour les erreurs	112
Figure 122: Réception d'une erreur sur Slack.....	113
Figure 123: Documentation du serveur	113
Figure 124: Edition d'un tutoriel sur Camtasia	114
Figure 125: Page Facebook de Sustainapp	115

Table des tableaux

Tableau 1: Les couleurs de Sustainapp	49
Tableau 2: Les pages de Sustainapp	52
Tableau 3: User stories de Sustainapp	55
Tableau 4: Plan des releases.....	56
Tableau 5: Découpage en tâches du sprint 0.....	57
Tableau 6: Découpage en tâches du sprint 1.....	77
Tableau 7: Découpage en tâches du sprint 2.....	80
Tableau 8: Découpage en tâches du sprint 3.....	82
Tableau 9: Découpage en tâches du sprint 4	88
Tableau 10: Découpage en tâches du sprint 5	94
Tableau 11: Découpage en tâches du sprint 6	97