

# DOSSIER DE CONCEPTION



**ECO-LEAVES**

**Réseau social associatif environnemental**

Date 10/11/17

Version 1.1

Nom Projet ECO-LEAVES

Auteurs BENAOU Mohammed  
CHAMBON Thomas  
EL HAJJAMI Zakariae  
GUIRO Ibrahima  
MARQUAY Christian  
PEPPER Thomas

# 1. Introduction

Cette section donne une description de portée et une vue d'ensemble de tout ce qui est inclus dans ce document.

## 1.1 Cadre du document

Ce document présente une description détaillée des exigences de la solution de la plateforme "ECO-LEAVES" (E-L). Ce document constituera une référence fonctionnelle de ce projet. Il illustrera le but et la déclaration complète pour le développement du système. Il expliquera également les contraintes du système et de l'interface. Ce document est principalement destiné à être proposé pour son approbation et à devenir une référence pour le développement de la première version du système pour l'équipe de développement.

## 1.2 Présentation de la solution

La plateforme ECO-LEAVES a pour objectif de permettre aux utilisateurs de se rencontrer en ligne afin d'organiser des événements associatifs environnementaux et d'échanger des objets qui ne leur sont plus utiles. Cet échange a pour but de combattre l'obsolescence programmée et le gaspillage des ressources.

Les utilisateurs de la plateforme sont des citoyens, membres ou non d'associations sensibilisées aux enjeux environnementaux et de développement durable.

Le Core fonctionne sous JAVA, langage indépendant de tout processeur et de tout système d'exploitation avec un fort potentiel d'industrialisation.

Les bases de données fonctionnent sous Elasticsearch, un serveur utilisant Lucene pour l'indexation et la recherche des données. Il fournit un moteur de recherche distribué et multi-entité à travers une interface REST. L'échange des données se fait au format JSON.

L'application mobile fonctionne sous Cordova, permettant de créer des applications pour différentes plateformes en HTML, CSS et JavaScript. La partie JavaScript contient l'application de messagerie (chat) pour mettre en contact les utilisateurs de la plateforme. L'application sera déployée sur les plateformes IOS et Android.

## 1.3 Cas général d'utilisation

Le cas général d'utilisation est le suivant :

- L'utilisateur (personne inscrite sur la plateforme) ou une association (membre représentant l'association) décide d'organiser un événement ou une collecte d'objets usagés dans un cadre d'actions environnementales.
- L'utilisateur peut communiquer avec d'autres utilisateurs à l'aide d'une messagerie instantanée.
- L'utilisateur peut à tout moment consulter la liste des événements créés sur la plateforme et les annonces d'échange.
- L'utilisateur peut créer une annonce d'échange de produit(s) usagé(s).
- L'utilisateur peut modifier son ou ses annonce(s) d'échange de produit(s) usagé(s).
- L'utilisateur peut supprimer son ou ses annonce(s) d'échange de produit(s) usagé(s).
- L'utilisateur peut créer un événement.
- L'utilisateur peut modifier son ou ses événement(s).
- L'utilisateur peut supprimer son ou ses événement(s).
- L'utilisateur peut créer un compte (profil utilisateur).
- L'utilisateur peut modifier son compte (profil utilisateur).
- L'utilisateur peut supprimer son compte (profil utilisateur).
- Le "Core" identifie chaque donnée envoyée et les transmet à un serveur de base de données.
- Les bases de données doivent stocker les membres inscrits de la plateforme, les événements de la plateforme et les articles à échanger.
- L'utilisateur peut se connecter à cette plateforme grâce à une application mobile multiplateforme.

## 2. Description générale

Cette section donne un aperçu de l'ensemble du système. Le système sera expliqué dans son contexte pour en introduire les fonctionnalités de base. Il décrira également le type d'intervenants qui utiliseront le système et les fonctionnalités disponibles pour chaque type. Enfin, les contraintes et hypothèses du système seront présentées.

### 2.1 Perspective du produit

Ce système se compose de trois parties : une partie web avec une API Rest qui produira les microservices consommés dans le Cloud, une partie back-end en Java, et une application mobile multi-plateformes iOS/Android. La API Rest sera utilisée pour obtenir les données stockées et les mettre dans un format JSON prêt à être utilisé tandis que l'application mobile sera utilisée pour afficher les informations auprès des usagers. La API Rest devra communiquer avec l'application mobile pour lui fournir les informations que celle-ci lui demande. Comme il s'agit d'un produit axé sur les données, il lui faudra un endroit où stocker les données. Pour cela, une base de données sera utilisée pour chaque microservice. L'application mobile communique également à la API Rest les données des événements ou des offres d'échanges de matériels à ajouter, modifier, ou supprimer dans les bases de données. Toute la communication de base passera par internet.

### 2.2 Caractéristiques de l'utilisateur

Il existe deux types d'utilisateurs qui interagissent avec le système : les utilisateurs de l'application et les administrateurs. Chacun de ces deux types d'utilisateurs a une utilisation différente du système de sorte que chacun d'eux a ses propres exigences. Les utilisateurs de l'application peuvent l'utiliser pour visualiser les informations, ainsi que pour créer, modifier, et supprimer ses propres événements. Cela signifie que l'utilisateur doit être en mesure de rechercher des informations, d'en envoyer, d'en modifier, et d'en supprimer. Pour que les utilisateurs puissent obtenir un résultat de recherche, il existe plusieurs critères que les utilisateurs peuvent spécifier et tous les résultats correspondent à ceux-ci. Les administrateurs n'interagissent avec l'application mobile qu'afin de gérer l'ensemble du système de façon à ce qu'il n'y ait pas d'information inexacte. L'administrateur peut également gérer les options pour les utilisateurs.

## 2.3 Contraintes

La connexion internet est une contrainte. Etant donnée que l'application récupère les données des bases via internet, il est essentiel qu'il y ait une connexion internet pour que l'application fonctionne. La Rest API et l'application mobile seront limitées par la capacité des bases de données.

# 3. Choix technologiques

## 3.1 Elasticsearch

Elasticsearch est un moteur de recherche sur tout type de document, ayant pour avantages :

- Architecture adaptable (montée en charge).
- Utilise le format JSON via des requêtes HTTP (moteur de recherche utilisable avec n'importe quel langage de programmation).
- Interactions avec la base par requête REST.
- Open Source

## 3.2 Java / JEE

Java est un langage de programmation orienté objet. Ce langage a été imposé pour le back-end de l'application, il a notamment pour avantages :

- Langage facilement portable (JVM).
- Résultats indépendants de la plateforme d'exécution.
- Applications facilement industrialisables (JEE).

## 3.3 Cordova

Framework de développement sur différentes plateformes notamment mobiles (IOS/Android). Ce framework repose sur les trois langages web HTML, JS et CSS et est open source. Il offre une portabilité accrue du code (iOS/Android) et une facilité de développement des langages de programmation web "simples" (HTML/CSS).

### 3.4 JQuery / ChatJS

ChatJS est un plugin jQuery pour la messagerie instantanée, qui est côté client. La communication côté serveur est implémentée via des adaptateurs. L'adaptateur est le morceau de code qui est responsable de la gestion de la communication client / serveur.

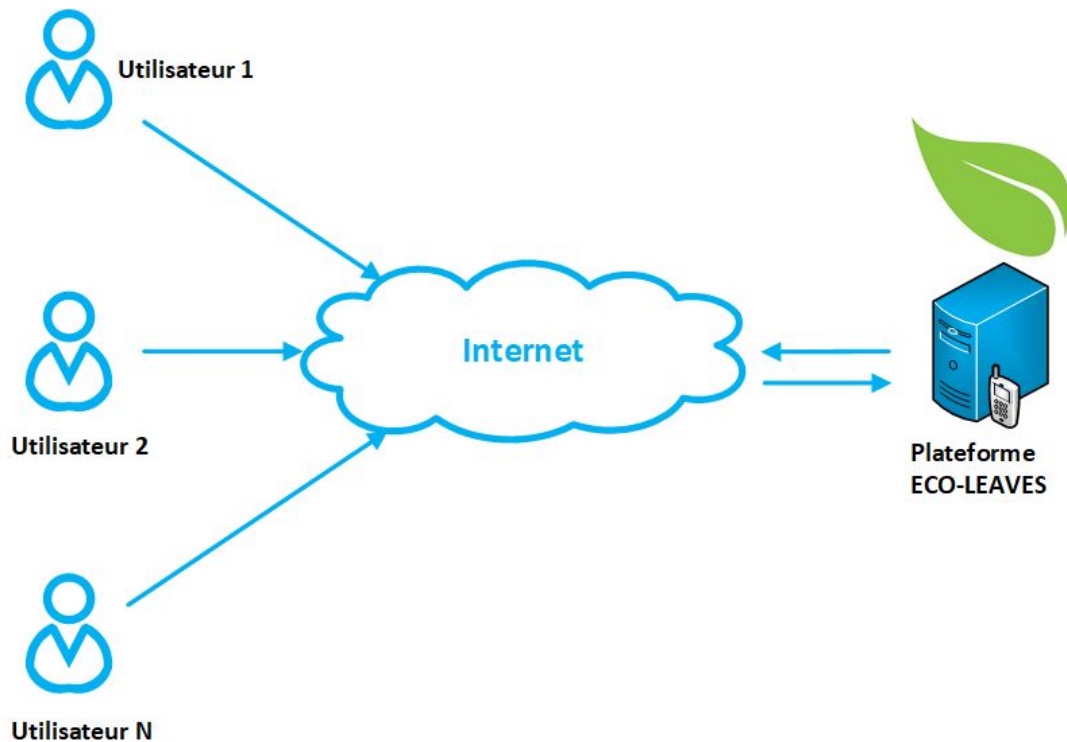
### 3.5 Spring

Spring est un framework Java pour l'utilisation d'un serveur de type MVC dans une application web, ça sert à construire et définir l'infrastructure d'une application Java.

### 3.6 Sécurité

Différents aspects de la sécurité doivent être implémentés lors de la programmation de l'application mobile : authentification, autorisations, cryptage, sessions, cookies, etc. La sécurité doit être incluse des deux côtés : client et serveur. L'authentification permet d'identifier un utilisateur, ne serait-ce que par son login et son mot de passe. Pour l'authentification nous allons utiliser JAAS (Java Authentication and authorization service), un framework de sécurité de Java. Il permet l'authentification des utilisateurs utilisées dans une application Java. Une API Key est une clé cryptée qui permet d'identifier qui appelle le programme - un utilisateur ou un développeur - pour avoir des logs d'utilisation de la plateforme. Avec Cordova, on peut ajouter une liste blanche qui contrôle l'accès à des domaines externes sur lesquels l'application n'a aucun contrôle. Concernant les bases de données, les mots de passe doivent au minimum être cryptés en sha-1.

## 4. Architecture réseau

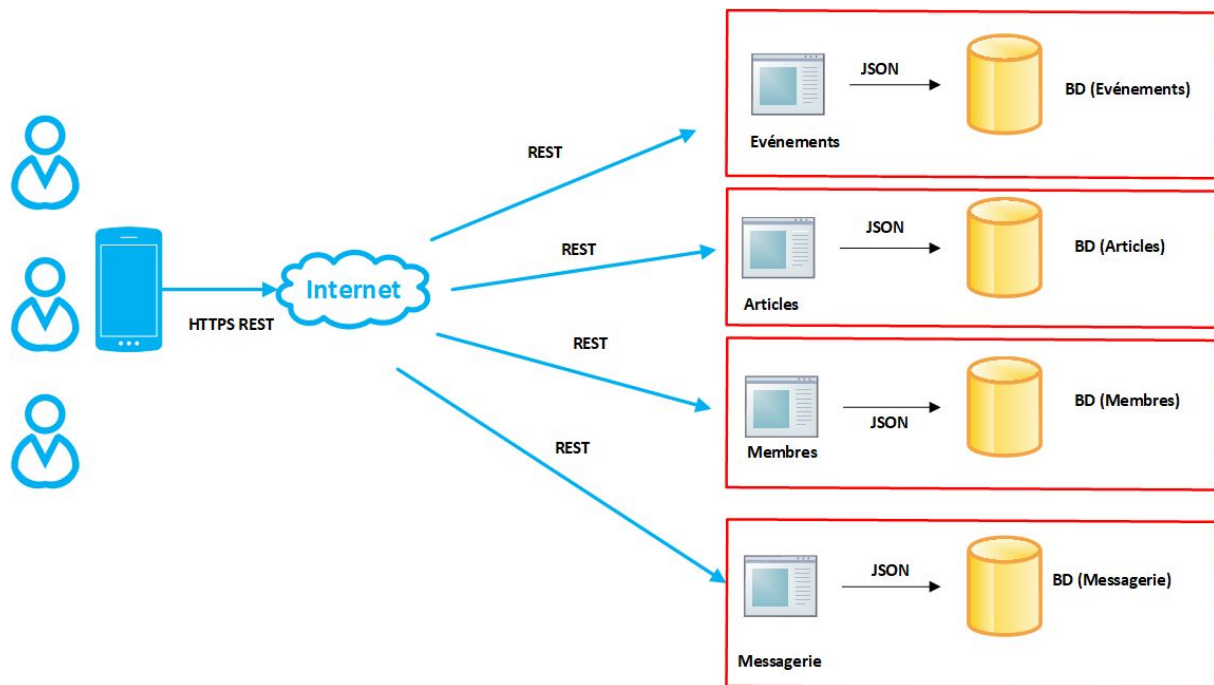


### Partitionnement de la plateforme

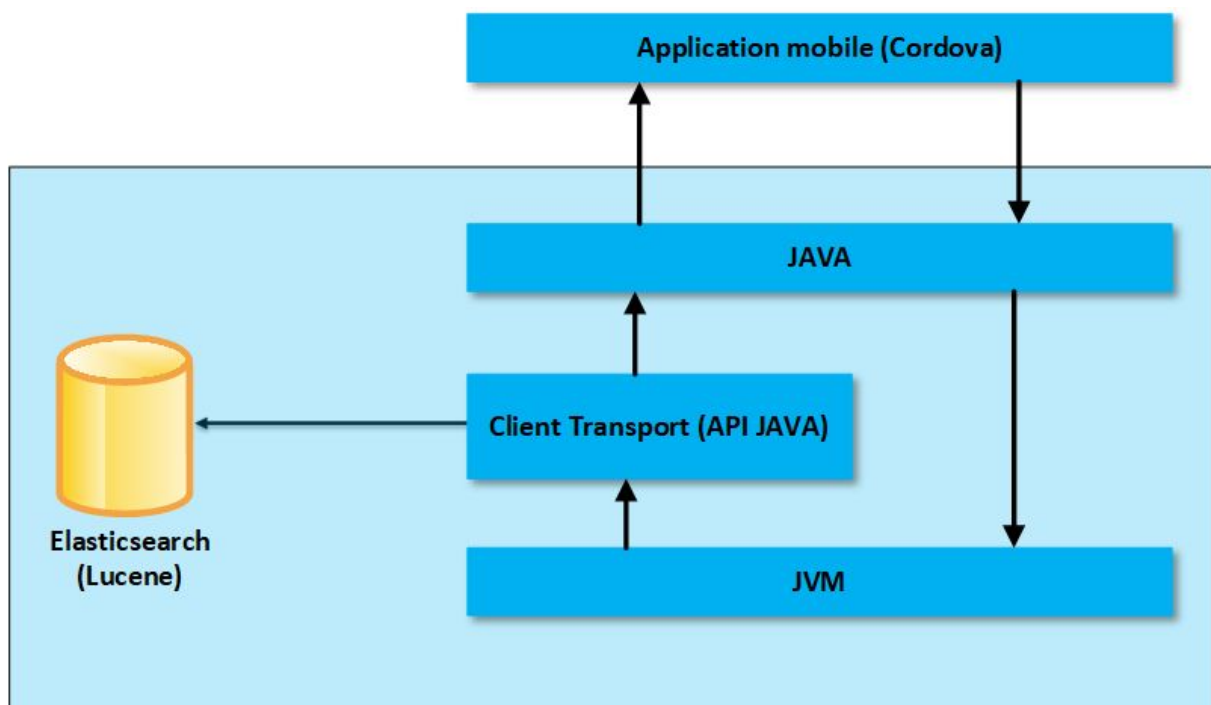
La plateforme E-L doit intégrer les interfaces suivantes :

- Les utilisateurs doivent être connectés à internet.
- L'architecture REST pour échanger des données entre les utilisateurs (côté Client) et les "Core" (côté Serveur).
- Le format JSON permet l'échange des données entre les "Core" et les bases de données spécifiques.

L'architecture réseau est définie dans la topologie suivante :



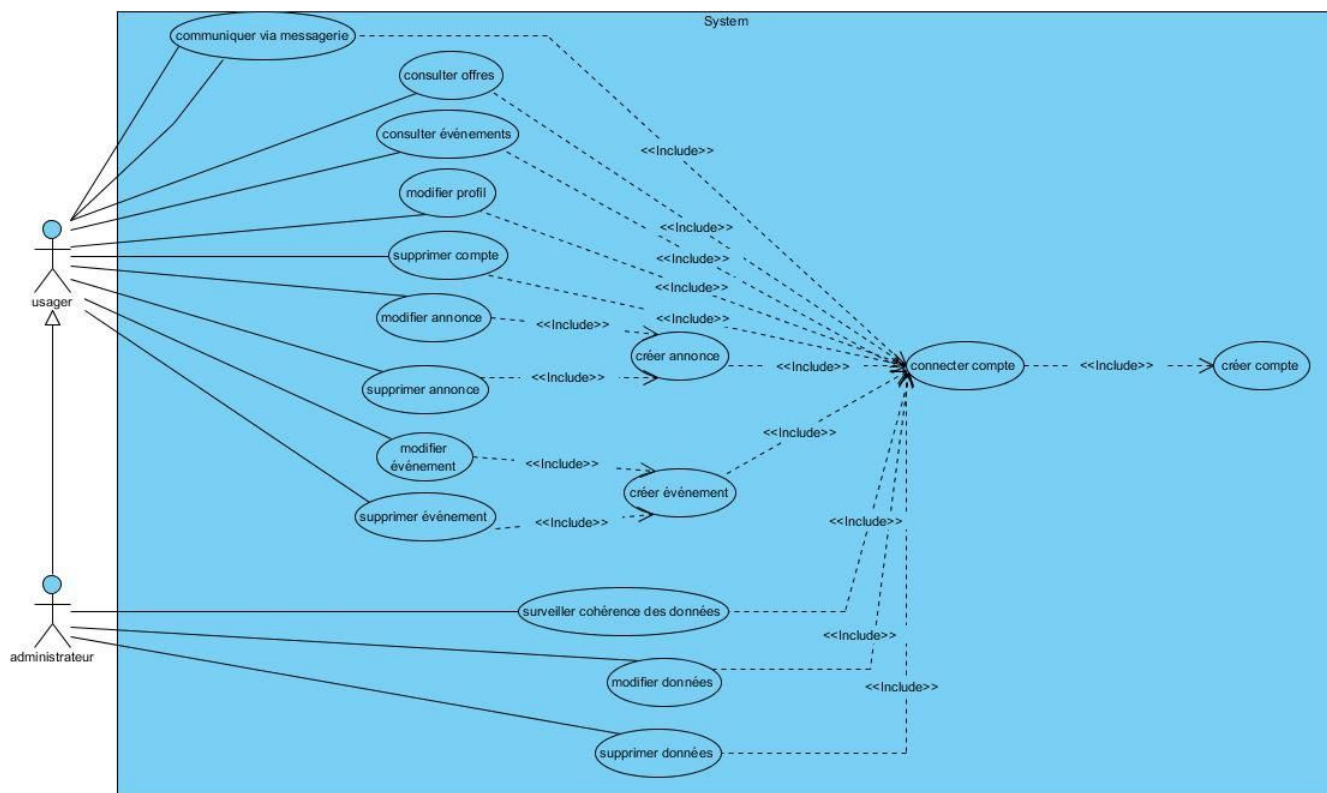
## 5. Architecture logicielle



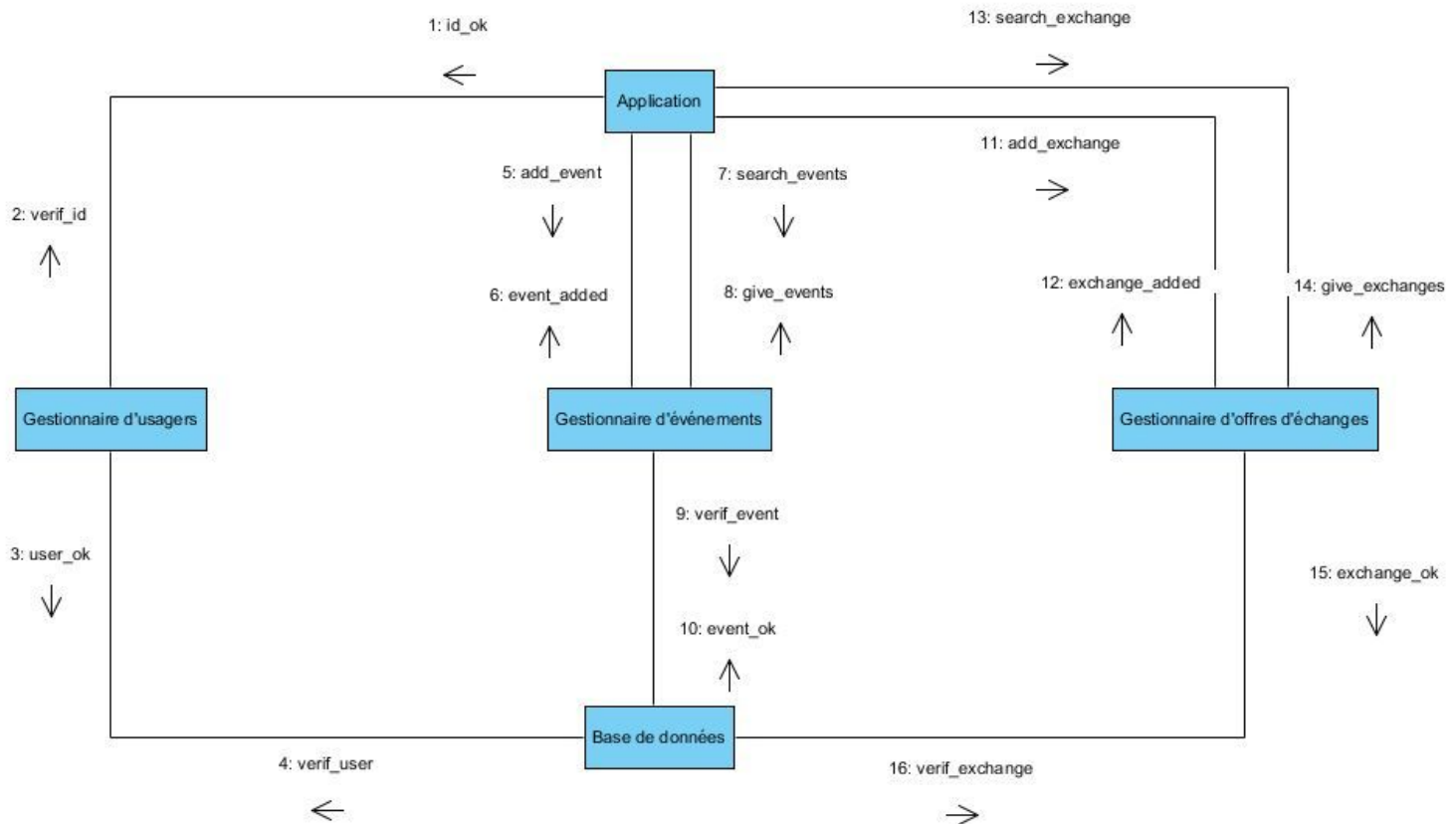
Architecture logicielle de la plateforme



## 6. Cas d'utilisation général



## 7. Interaction entre les composants



## 8. Pré Requis pour les serveurs

- REQ-1 Le Core communique avec un front end mobile service utilisé par les utilisateurs.
- REQ-2 Le Core possède un service permettant de recevoir des messages et les traiter de façon pertinente.
- REQ-3 Le Core est chargé de décoder les messages qu'il reçoit.
- REQ-4 Le Core est chargé de distribuer les données à la table appropriée.
- REQ-5 Le code doit être écrit de manière à favoriser la mise en œuvre de nouvelles fonctions.
- REQ-6 Le système doit donner le bon résultat lors d'une recherche.
- REQ-7 Le Core doit communiquer avec le protocole HTTPS.
- REQ-8 Le Core doit communiquer avec la base de données appropriée avec le format JSON.
- REQ-9 Le système doit fonctionner avec le framework Spring.

## 9. Pré Requis pour les bases de données

- REQ-1 La base de données doit stocker les données au format JSON.
- REQ-2 Les mots de passe des utilisateurs doivent être stockés en base sous forme crypté.
- REQ-3 Le profil de l'administrateur de la plateforme E-L possède les droits suivants :
- CRUD sur la configuration des paramètres de la plateforme (paramètres d'informations, paramètres d'interface).
  - CRUD sur les profils accédant à la plateforme.
  - CRUD sur les événements de la plateforme.
  - CRUD sur les échanges de la plateforme.
  - RD sur les enregistrements d'activité (logs) de la plateforme.
  - CRUD sur les enregistrements de la base de données (DB).
- REQ-4 Le profil de l'utilisateur de la plateforme E-L possède les droits suivants :
- R sur la configuration des paramètres de la plateforme (paramètres d'informations, paramètres d'interface)
  - R sur les enregistrements de la base de données (DB).
  - CRUD sur les informations concernant son profil.
  - CRUD sur les informations concernant son ou ses événements.
  - CRUD sur les informations concernant son ou ses échanges.

## 10. Pré Requis application mobile

- REQ-1 L'utilisateur doit pouvoir télécharger l'application depuis le store.
- REQ-2 L'utilisateur doit pouvoir se connecter à l'application.
- REQ-3 L'utilisateur doit être en mesure de rechercher des informations.
- REQ-4 L'utilisateur doit être en mesure de visualiser les informations.
- REQ-5 L'utilisateur doit être en mesure de modifier son mot de passe et ses informations de profil.
- REQ-6 L'utilisateur doit être en mesure de créer des annonces et des événements.
- REQ-7 L'utilisateur doit pouvoir gérer ses propres annonces et événements (modification et suppression).
- REQ-7 L'administrateur doit pouvoir se connecter à l'application.
- REQ-8 L'administrateur doit pouvoir modifier les informations le concernant, notamment le mot de passe.
- REQ-9 L'administrateur doit pouvoir gérer les annonces et événements (modification et suppression).
- REQ-10 L'administrateur doit pouvoir gérer tous les utilisateurs (modification et suppression).
- REQ-11 L'application communique avec le Core en HTTPS.

## 12. Mode de déploiement

Pour déployer notre application sur le cloud, nous allons utiliser le framework Docker pour la virtualisation d'infrastructure. Il fournit une structure spéciale appelée container, légère et isolée des machines. Cela permet d'éviter de configurer chaque serveur, et d'envoyer juste les conteneurs dont on a besoin, en étant certain qu'ils marchent partout pareil, peu importe la configuration à l'arrivée. On virtualise des services, qu'on combine, et non une machine complète. Docker peut donner une meilleure expérience au développement d'applications, surtout quand cela concerne des applications large-scale avec des environnements complexes et un temps de compilation long. Docker peut également aider à fournir un meilleur environnement de test et de build, et à allouer les ressources plus efficacement. Par la suite pour déployer l'application, nous allons mettre nos "containers" du docker sur une plateforme cloud: "Heroku". Heroku permet le déploiement très rapide d'applications web dans le cloud, avec une gestion très souple au travers d'un modèle de gestion des processus emprunté à Unix et adapté au Web.

## 13. Tests

Lorsque le développement de l'application est achevé, il est indispensable de mener une recette approfondie. C'est dans cette optique que nous allons élaborer des scénarios de test pour détecter des dysfonctionnements et des erreurs notamment sur les points suivants :


- Installation / désinstallation, lancement de l'application
- Performance : rapidité et consommation de batterie
- Connexion : impacts du réseau sur l'application
- Interruption : comportement en cas d'appel, de notification
- Interface : adaptation du design
- Mémoire : gestion de la mémoire de l'application
- Stabilité : absence de plantage
- Soumission : conformité aux règles des stores
- Unitaire : "bon fonctionnement"

## 14. Les objectifs de l'équipe

Objectif de l'équipe pour le projet ECO-LEAVES:

- Créer un réseau social associatif pour la protection d'environnement permettant à ses adhérents :
  - De faire des échanges d'objets avec d'autres adhérents
  - De créer des événements
  - D'obtenir le nombre d'événements créés
  - De participer à un événement
  - D'obtenir son nombre de participation aux événements
  - ...
- Respecter les spécifications et les contraintes de mise en oeuvre
- Mettre en place un fonctionnement basé sur nos qualités et compétences
- Etablir un plan de communication intègre
- Collaborer pour mettre en oeuvre le projet et atteindre la cible commune
- Livrer le projet dans le délai prévu.

## 15. Maquette de l'application mobile



Username

Password

[Login](#) [SignIn](#)

[Forgot password?](#)



[Déconnexion](#)

Bienvenue Utilisateur 

[Home](#) [Calendar](#) [Add Event](#) [Messages](#)

 **Utilisateur** 08/11/2017  
Créer un événement lorem Ipsum est simplement du faux texte employé dans la composition et la mise en page avant impression ...  
[Lire plus ...](#)

 **Utilisateur** 08/11/2017  
Publier un article lorem Ipsum est simplement du faux texte employé dans la composition et la mise en page avant impression ...  
[Lire plus ...](#)

 **Utilisateur** 08/11/2017  
Créer un événement lorem Ipsum est simplement du faux texte employé dans la composition et la mise en page avant impression ...  
[Lire plus ...](#)

 **Utilisateur** 08/11/2017  
Créer un événement lorem Ipsum est simplement du faux texte employé dans la composition et la mise en page avant impression ...  
[Lire plus ...](#)

## 16. Table des matières

<b>1. Introduction</b>	<b>2</b>
1.1 Cadre du document	2
1.2 Présentation de la solution	2
1.3 Cas général d'utilisation	3
<b>2. Description générale</b>	<b>4</b>
2.1 Perspective du produit	4
2.2 Caractéristiques de l'utilisateur	4
2.3 Contraintes	5
<b>3. Choix technologiques</b>	<b>5</b>
3.1 Elasticsearch	5
3.2 Java / JEE	5
3.3 Cordova	5
3.4 JQuery / ChatJS	6
3.5 Spring	6
3.6 Sécurité	6
<b>4. Architecture réseau</b>	<b>7</b>
<b>5. Architecture logicielle</b>	<b>8</b>
<b>6. Cas d'utilisation général</b>	<b>9</b>
<b>7. Interaction entre les composants</b>	<b>10</b>
<b>8. Pré Requis pour les serveurs</b>	<b>10</b>
<b>9. Pré Requis pour les bases de données</b>	<b>11</b>
<b>10. Pré Requis application mobile</b>	<b>12</b>
<b>12. Mode de déploiement</b>	<b>13</b>
<b>13. Tests</b>	<b>13</b>
<b>14. Les objectifs de l'équipe</b>	<b>14</b>
<b>15. Maquette de l'application mobile</b>	<b>15</b>
<b>16. Table des matières</b>	<b>16</b>