

**DOSSIER PROJET**  
**Clément MEZERETTE**

**Titre professionnel "Développeur  
logiciel"**



## ***Remerciements***

Je tenais à remercier l'ensemble de l'équipe d'INFOTEL qui a su m'accueillir et me soutenir, tout particulièrement Hugo LAMARRE et Cyril GERBIER, Franck MAFFENINI qui m'a rejoint sur le projet, Justine et Romain de la Wild Code School, ma compagne Pauline pour son extrême patience et Stack Overflow.

Ils ont tous contribué à ma reconversion et méritent à ce titre, toute ma gratitude.



## **ABSTRACT**

This application is designed for large companies or territorial collectivities.

From the front-end point of view, we have a mobile application for residents or employees (for now, only on Android but the ios version is planned). They can use the application to send incident reports: whether it's for flooding, electrical issues, trashes... This report will be geolocated by the smartphone, localised in a small area within 200 meters. The item (aka the incident report) is made in a specific way: it has several elements such as photos, addresses (handwritten or geolocated), comments, and the nature of the incident (Chosen in a list of categories defined by the admin).

A map in the application shows the users incidents close by: They also have the possibility to back another user's incidents and thus, increase the urgency of this event.

From the back-end point of view, the reports are sent to a server and a web site allows collectivities to receive detailed reports. This way, the "admin" has access to all the data, with different functionalities : sorting incidents by nature, display them on a map, or view the present priorities in real-time...

The companies can focus on the most urgent issues, be aware of incidents faster than usual, and deploy solutions more efficiently.

## **CONTEXTE**

Mon stage s'est déroulé du 6 Février 2017 au 2 Juin 2017, au Centre de Compétences Mobiles d'INFOTEL à Balma, nouveau service créé depuis quelques mois seulement.

INFOTEL est un groupe de sociétés de services et un éditeur de logiciels international, comptant plus de 2000 collaborateurs et réalisant un chiffre d'affaires de 190 M€ (2016).

J'ai été encadré par un responsable opérationnel, Hugo LAMARRE et un responsable technique, Cyril GERBIER. Nous avons utilisé une gestion de projet agile en se basant sur le SCRUM avec des sprints de un mois ; je devais rédiger un backlog de sprint à chaque début de cycle en me basant sur les spécificités et attentes de mon Product Owner et du product backlog.

A défaut de daily scrum, nous avons fait des weekly, nous permettant de partager les avancées, retards ou éventuels problèmes rencontrés.

Les nouvelles fonctionnalités étaient présentées à la démonstration de fin de sprint chaque mois.

Bien que travaillant de manière autonome et solitaire sur le projet durant les trois premiers mois, l'ensemble des réalisations était centralisé sur la plateforme interne d'intégration continue de l'entreprise via Jenkins, sur lequel le code « poussé » était testé.

Nous avons utilisé GIT comme système de gestion de versions et SonarQube pour mesurer la qualité du code.

Ainsi, les rapports de Sonar délivrés à chaque "push" permettaient un retour sur chaque commits envoyés, et par là même, l'amélioration permanente de mon code afin de réaliser un code "industrialisé" tout en respectant les standards et normes Android : convention de nommage, sécurité, gestion des exceptions, erreurs ou autres "code smells"...

Mon dossier projet s'appuie sur l'application que j'ai développée chez INFOTEL pendant mon stage. Il s'agit d'un projet interne d'INFOTEL s'inspirant d'applications similaires avec l'ajout de features inédites.

J'ai démarré le projet du début, en faisant une étude de marché et analysant les applications existantes se rapprochant du concept de projet souhaité par INFOTEL (FixMyStreet, TellMyCity...).

Certaines fonctionnalités ne faisaient pas partie de l'offre proposée par la "concurrence", je me contenterai d'exposer ci-après, celles développées et non soumises à la décharge de confidentialité signée.

## **COMPETENCES DEVELOPPEES**

### **Maquettage :**

Après avoir défini les attentes du client et le “scope” du projet de la partie mobile, j’ai rédigé un backlog avec des users stories :

- Rédaction et définitions des histoires utilisateurs
- Priorisation des users stories
- Estimation du temps de développement des stories

Afin de mieux appréhender la méthode agile, j’ai rédigé deux documents nécessaires à la validation des user stories venant préciser des critères de validation :

- Le DoR (Document of Ready)
  - User stories claires et définies
  - Stories approuvées par le PO
  - Temps des tâches estimées
  - Documentation fonctionnelle rédigée
  - Critères d’acceptation validés
- Le DoD (Document of Done)
  - Code réalisé
  - Code commenté
  - Build sans erreurs
  - Test unitaires écrits et passés
  - Intégration testée
  - Documentation technique mise à jour
  - Approuvé par le PO



Le backlog fut ensuite soumis à mon Product Owner pour validation avant chaque début de sprint.

J'ai ensuite créé une série de wireframes pour communiquer sur des propositions d'expériences utilisateur et scénariser l'enchaînement des écrans de l'application.

A cet effet, j'ai utilisé Balsamiq pour le "mock-up" présenté au P.O.

*A défaut de cinématique à produire, vous trouverez des photographies de l'application web et mobile dans l'annexe (cf. infra – annexe p.27 à 35).*

\*\*\*

## **Conception d'une base de données :**

Côté mobile, la base de données, locale (SQLite), ne sert que deux objectifs :

- Enregistrer les signalements effectués depuis le mobile (pour avoir accès aux données des signalements effectués hors connexion)

- Définir l'arborescence de la liste des catégories

En effet, lors du signalement d'un incident, l'utilisateur doit choisir à minima, une catégorie. Il se peut dans certains cas, que la catégorie choisie soit une sous-catégorie de sous-catégorie.

Chaque catégorie est représentée par une table avec deux champs, sa clé primaire et son libellé (en string) ; la clé étrangère de chaque catégorie étant la clé primaire de sa « catégorie-mère » pour les sous-catégories.

Ainsi, à chaque lancement de l'application, une requête GET récupère la liste sous format JSON.

Une fois « parsée », une requête SQL permet de mettre à jour ou de créer notre base de données avec les informations récupérées en amont.

Côté serveur, nous avons utilisé PostgreSQL.

Nous avons donc dû :

- Réaliser un schéma entité – association de la Base de données.
- Normaliser les données.
- Définir le type de chaque donnée stockée.

La base de données, qui a connu plusieurs versions, a évolué au fil du temps, que ce soit sur son fonctionnement ou les besoins métiers (les premières versions ne prenaient pas en compte les agents par exemple).

Tous ces changements m'ont permis de me perfectionner sur la conception et la mise en œuvre d'une base de données, du CRUD aux relations entre les différentes tables, des clés primaires et étrangères.

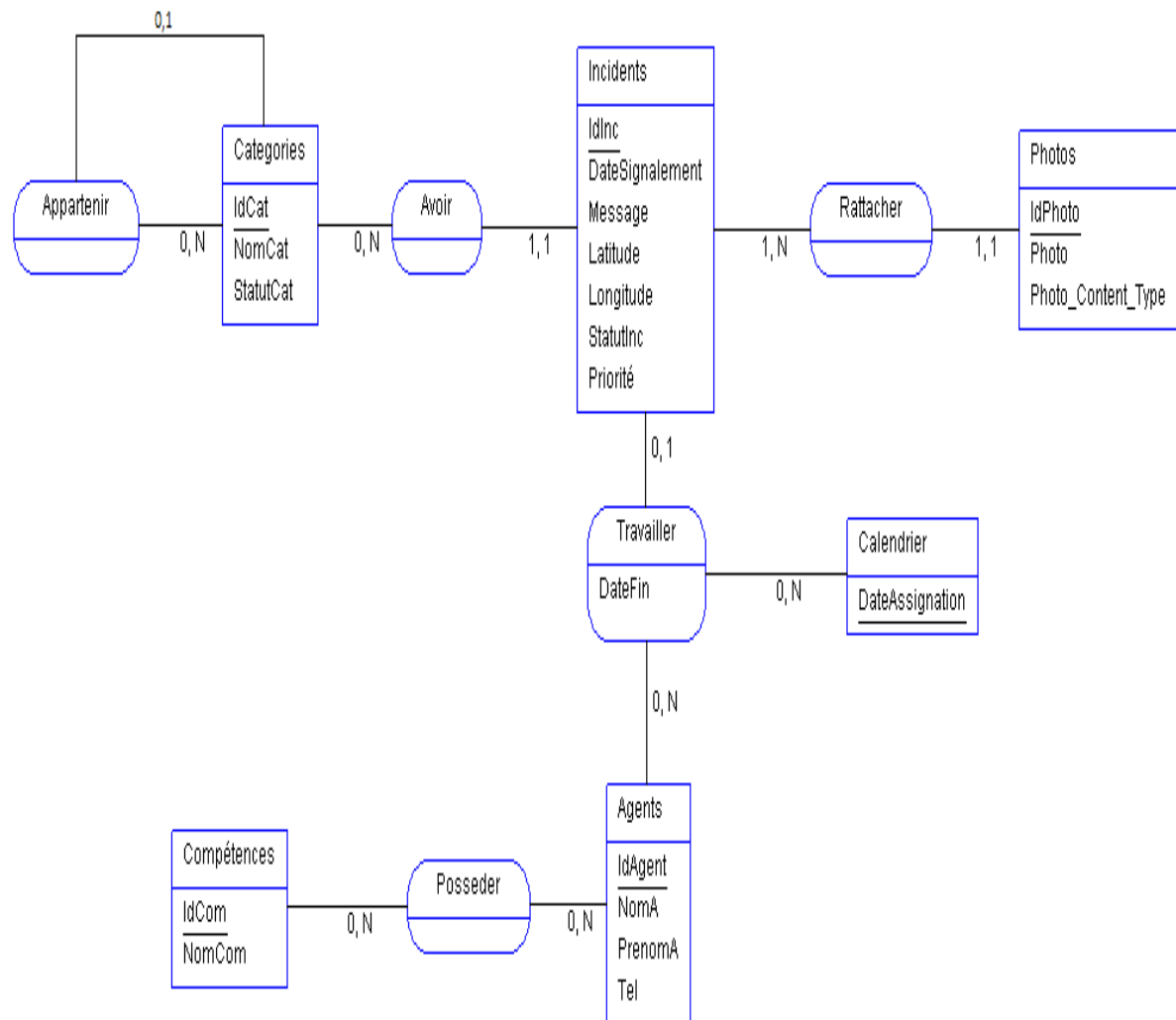
Ainsi, vous trouverez ci-après :

- 1) Un dictionnaire de données
- 2) Un schéma entité – association
- 3) Un schéma relationnel

## 1- Dictionnaire de données

Libellé	Description	Type	Domaine	Contrainte
DateSignalement	Date de l'émission d'un signalement d'un incident	Date	Incidents	jj/mm/aa
IdInc	Identifiant d'un incident	BIGINT	Incidents	NOT NULL
Latitude	Latitude GPS d'un incident	BIGDEC	Incidents	NOT NULL
Longitude	Longitude GPS d'un incident	BIGDEC	Incidents	NOT NULL
Message	Commentaire lié à un incident	Texte	Incidents	
StatutInc	Statut d'un incident.	Numerique	Incidents	
Priorité	Permet de visualiser la priorité d'un incident, plus le chiffre est élevé plus la priorité est importante	Numérique	Incidents	>0
IdCat	Identifiant d'une catégorie	BIGINT	Categories	NOT NULL
NomCat	Nom d'une catégorie	Texte	Categories	NOT NULL
StatutCat	Statut qui indique si la catégorie est active ou inactive (c'est-à-dire visible par les utilisateurs (lambda ou admin). True= Active, False= Inactive.	Boolean	Categories	Valeur possible: True, False
Photo	Photo illustrant l'incident	Blob	Photos	Encodé en base 64
Photo_Content_Type	Format de la photo	Texte	Photos	Si photo prise = NOT NULL
DateAssignment	Date d'assignation d'un incident à un agent par l'admin	Date	Calendrier	jj/mm/aa and >= DateSignalement
IdAgent	Identifiant d'un agent	BIGINT	Agents	NOT NULL
NomA	Nom de famille d'un agent	Texte	Agents	
PrenomA	Prénom d'un agent	Texte	Agents	
Tel	Numéro de téléphone d'un agent	Texte	Agents	
IdCom	Identifiant d'une compétence	BIGINT	Compétences	NOT NULL
NomCom	Nom d'une compétence	TEXTE	Compétences	

## 2- Schéma entité / association:



### **3- Schéma relationnel :**

Incidents {IdInc, DateSignalement, Message, Latitude, Longitude, StatutInc, Priorité, DateAssignment, DateFin, IdAgent\*, IdCat\*}

Categorie {IdCat, NomCa, IdCat\*}

Photos {IdPhoto, Photo, Photo\_Content\_Type, IdInc\*}

Agents {IdAgent, NomA, PrenomA, Tel}

Competences {IdCom, NomCom}

Posseder {IdCom\*, IdAgent\*}

## **Fonctionnalités développées :**

- **Géolocalisation**

Affichage d'une carte et représentation de l'utilisateur (marker) et/ou des autres signalements en récupérant les incidents sur le serveur.

Récupération des coordonnées GPS pour la géolocalisation « outdoor ».

Localisation manuelle en déplaçant le marker sur une carte.

Recherche d'adresse en cas de signalement en dehors du lieu de l'incident ou en mode hors connexion.

- **Récupération et affichage des données serveur sur le client**

Markers représentant les incidents sur une carte et leur description. Affichage du statut des incidents signalés (traités, en cours...).

Mise à jour ou peuplement de la base de données côté client avec la liste de catégorie définie côté serveur.

- **Prise et/ou envoi de photos**

Prise de photo via la galerie ou la caméra.

Visualisation des photos sous forme de vignette, possibilité d'effacer ces dernières.

Envoi de photos sous format jpeg avec qualité supérieure (faible taux de compression) pour garantir la visibilité de ces dernières.

Une photo est nécessaire au signalement, jusqu'à un maximum de trois côté mobile (pas de restriction de quantité côté serveur).

- **Envoi de l'incident via API REST**

A la fin du processus de signalisation.

Possibilité d'appuyer un signalement déjà existant via les markers représentés sur la carte.

- **Saisie de texte et informations liés à l'incident**

Possibilité de laisser un commentaire.

Choix d'une catégorie prédéfinie pour mieux cibler la nature de l'item et sa résolution par le service approprié.

Résumé de l'ensemble des informations nécessaires au signalement en fin de processus.

- **Consolidation des incidents**

Côté serveur : Regroupement des incidents par secteur, type, catégorie, date...

Côté client : un algorithme utilisant la position GPS de l'utilisateur ou de sa représentation (en cas de recherche par adresse) permet l'affichage des incidents dans un périmètre de 200 mètres maximum autour de lui.

- **Mise en place d'un site web pour consultation/tri des infos remontées**

Date et heure.

Adresses (GPS et postale).

Affichage sur carte des différents signalements.

Création et suivi de statuts afin de traiter les données remontées.

Classement et affichages des signalements suivant les filtres mis en place.

Mise en place d'outil de visualisation et métriques : graphiques, statistiques...

## **Technique :**

- **Géolocalisation de l'utilisateur**

Utilisation de Google Maps API pour la carte. Affichage de celle-ci dans un fragment (accessible dans plusieurs activités de l'application).

Une librairie utilisant le service Google Places, « Placesearchdialog:placesearch » et un outil de saisie semi-automatique de requête.

- **Reverse Géocoding**

Pour l'affichage, l'envoi ou la recherche d'adresse.

- **Traitement des images (prise, envoi, réception).**

Le système a été revu plusieurs fois suivant les exigences du Product Owner :

Côté mobile, l'affichage des photos sous forme de vignettes et la sauvegarde simultanée d'une autre version de meilleure qualité pour préserver la pertinence des données envoyées a constitué un petit challenge. En effet, la sauvegarde de photos et l'affichage affectant les performances du device, j'ai dû mettre en place un système de multi-threading puis de tâches asynchrones afin d'améliorer les performances et d'éviter de bloquer l'UI.

Le développement s'est effectué sur deux mobiles, un Samsung galaxy s7 et un galaxy s5 neo; le passage à Android Nougat (7.0) sur le s7 m'a également obligé à faire un patch de compatibilité mais a permis à l'application d'être correctement supporté sur plusieurs versions d'Android (de Kitkat à Nougat - 4.0 à 7.0 - ).

Pour l'envoi, les photos étaient convertis en base64.

- **Base de données SQLite**

Mise en place d'un Handler et implémentation des méthodes CRUD.

Les deux objectifs de la base de données étaient la sauvegarde de la liste de catégories définies depuis le serveur à chaque lancement de l'application et de



coordonnées GPS lors de la signalisation d'un incident (pour éviter les doublons, notamment lors du « soutien » d'un incident).

- **Base de données PostgreSQL** (cf.supra)
- **La plateforme d'intégration continue**

Nous utilisons comme plateforme d'intégration continue Jenkins et Git comme CVS.

A chaque « push », divers test étaient lancés, de build, de test unitaires, de qualités...

Pour les test de qualité de code SonarQube et JsLint furent utilisés. Ces derniers, m'ont énormément apporté grâce au retour quasi-immédiat sur ma production : que ce soit en termes de sécurité, de gestion des exceptions, de convention de nommages... L'IDE laissant parfois croire à un code sans erreurs mais loin de standards de code « industrialisé ».

Diverses librairies furent utilisées en back-end, que ce soit pour l'affichage des cartes (Google Maps API), ou de module graphique (NG2-charts).

- **Pour la mise en place du serveur**

Nous avons utilisé Jhipster :

Il fournit des outils pour générer un projet avec coté serveur, une pile Java (à l'aide de SpringBoot) et coté client un frontal web adaptatif(avec AngularJS et Bootstrap).

Notre version a été configuré avec Angular 4, encore en bêta, qui utilise du « typescript ».

Les langages utilisés côté web, HTML5 et CSS3 et JAVA pour la partie serveur.

L'exercice m'a permis de découvrir un nouvel IDE, IntelliJ IDEA.

Cette partie a été aussi excitante que complexe.

En effet, les forums concernant Jhipster ou son Git (pensée émue pour mes heures de recherches et mes problèmes de proxy) s'adressent à des développeurs confirmés.

Si cet outil facilite et accélère la production avec son module de génération notamment, la prise en main a représenté un défi.

- **La mise en place des API Rest**

L'API REST est le nouveau standard en matière de WebServices et le protocole utilisé pour communiquer est HTTP.

Tous les échanges entre le serveur et le mobile se font sous le format JSON, via l'utilisation de l'API REST :

*Echange lors du lancement de l'application*

Lors du lancement de l'application le mobile consommera le service (*/api/categories/trigger*) méthode GET, qui consultera la méthode java getCategoryLight. Le serveur renverra alors une liste de catégories, exemple :

```
[
  {
    "id": 1001,
    "nomCat": "ok",
    "idCat": 0
  },
  {
    "id": 1002,
    "nomCat": "test",
    "idCat": 1001
  }
]
```

Echange lors du lancement de l'application et du déplacement du curseur sur la google map

L'application consommera un deuxième service à l'adresse suivante (/api/markers) méthode GET, celui-ci utilisera la méthode java getIncidentsT.

Le service prendra deux valeurs en entrée la latitude et la longitude, ce qui donnera l'adresse suivant /api/markers?lat=42.11111&lon=1.55555, et retournera une liste d'incident.

```
[
  {
    "id": 1051,
    "message": "rrr",
    "latitude": 43.6306693562,
    "longitude": 1.489930898,
    "statutInc": 0,
    "idCat": 1005,
    "priorite": 1,
    "nomCat": "Chiot perdu"
  },
  {
    "id": 1053,
    "message": "Lampadaire cassé",
    "latitude": 43.630022,
    "longitude": 1.488729,
    "statutInc": 1,
    "idCat": 1007,
    "priorite": 2,
    "nomCat": "Lampadaire"
  }
]
```

### Echange lors de l'envoi d'un incident

L'envoi d'un incident et de ses photos sera fait en une fois. Toutes les informations sur les incidents et les photos sont donc requises. La photo sera envoyée en base64. L'adresse du webservice est (/api/mobileincidents) méthode POST.

Les informations envoyées depuis le mobile :

```
{
  "date": "2017-05-05"
  "idCat": 19,
  "latitude": 0,
  "longitude": 0,
  "message": "string",
}
{
  "photo": [
    "string"
  ]
}
```

### Echange « appuyer un incident en cours ou non traité »

Pour se faire l'application mobile consommera l'API suivante (/api/appuyer) méthode PUT, avec en paramètre l'ID de l'incident.

"id": 0,

Ce qui entrainera la mise à jour de l'incident en question est augmentera de 1 le nombre dans la colonne «priorité» (visible sur le site web de l'administrateur).

## **CONCLUSION**

Ce stage et ce projet furent particulièrement enrichissants car d'une part, j'ai découvert le fonctionnement interne d'une SSII et d'autre part, en démarrant le projet « from scratches », j'ai pu pleinement profiter de toutes les étapes de la vie d'une application jusqu'à la mise en production.

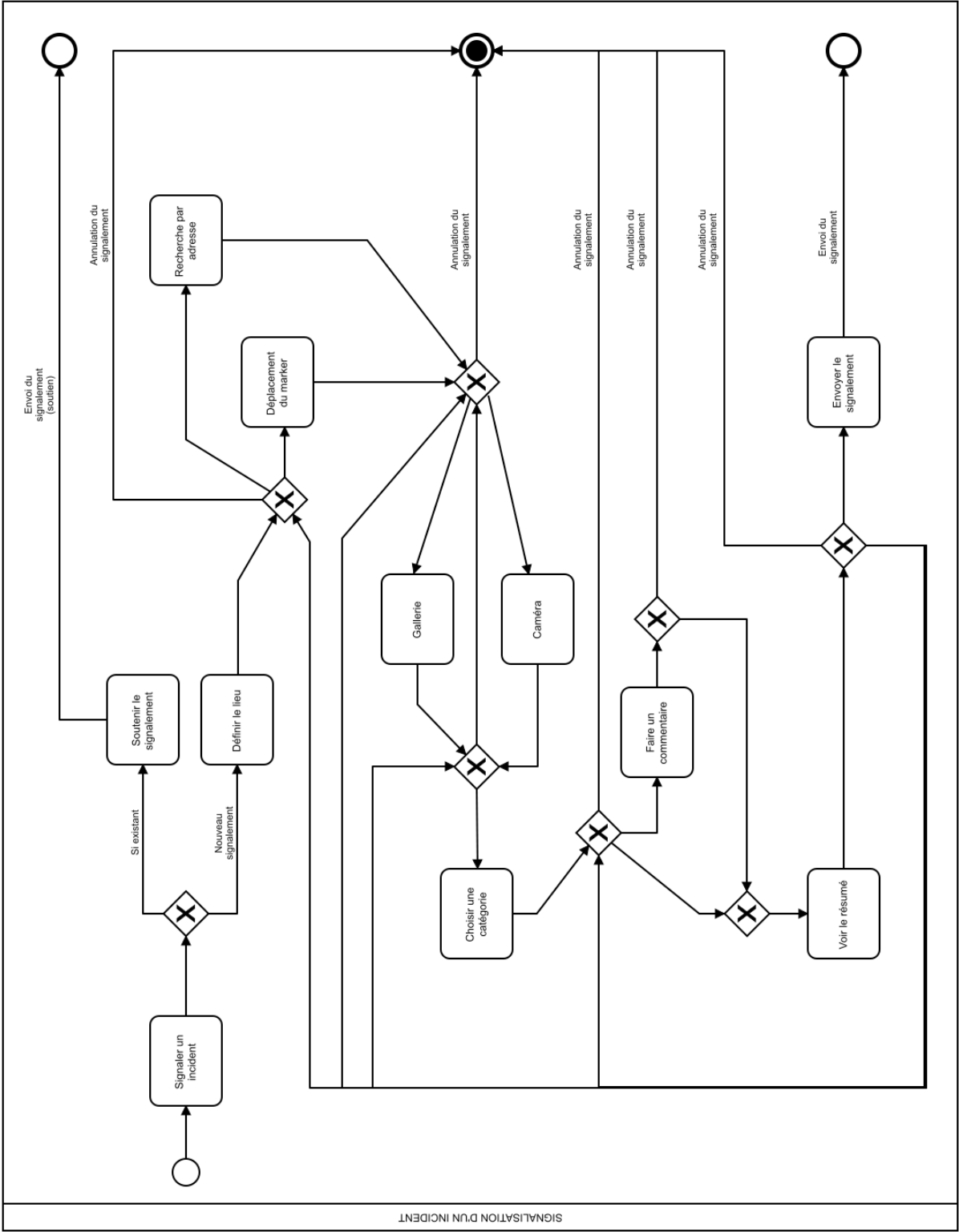
Que ce soit la partie conception, l'expression des besoins de l'utilisateur, la documentation à fournir (réalisation de schémas, rédaction de tests fonctionnels...), ou le développement des applications, j'ai vraiment pris beaucoup de plaisir sur ce projet.

A l'issue de cette expérience, j'ai eu l'opportunité de rejoindre la « famille » INFOTEL.

## **ANNEXES**

1- Diagramme d'activité d'un signalement.....	23
2- Diagramme gestion du statut des incidents.....	24
3- Pages du site web.....	25
4- Captures d'écran de l'application mobile.....	29

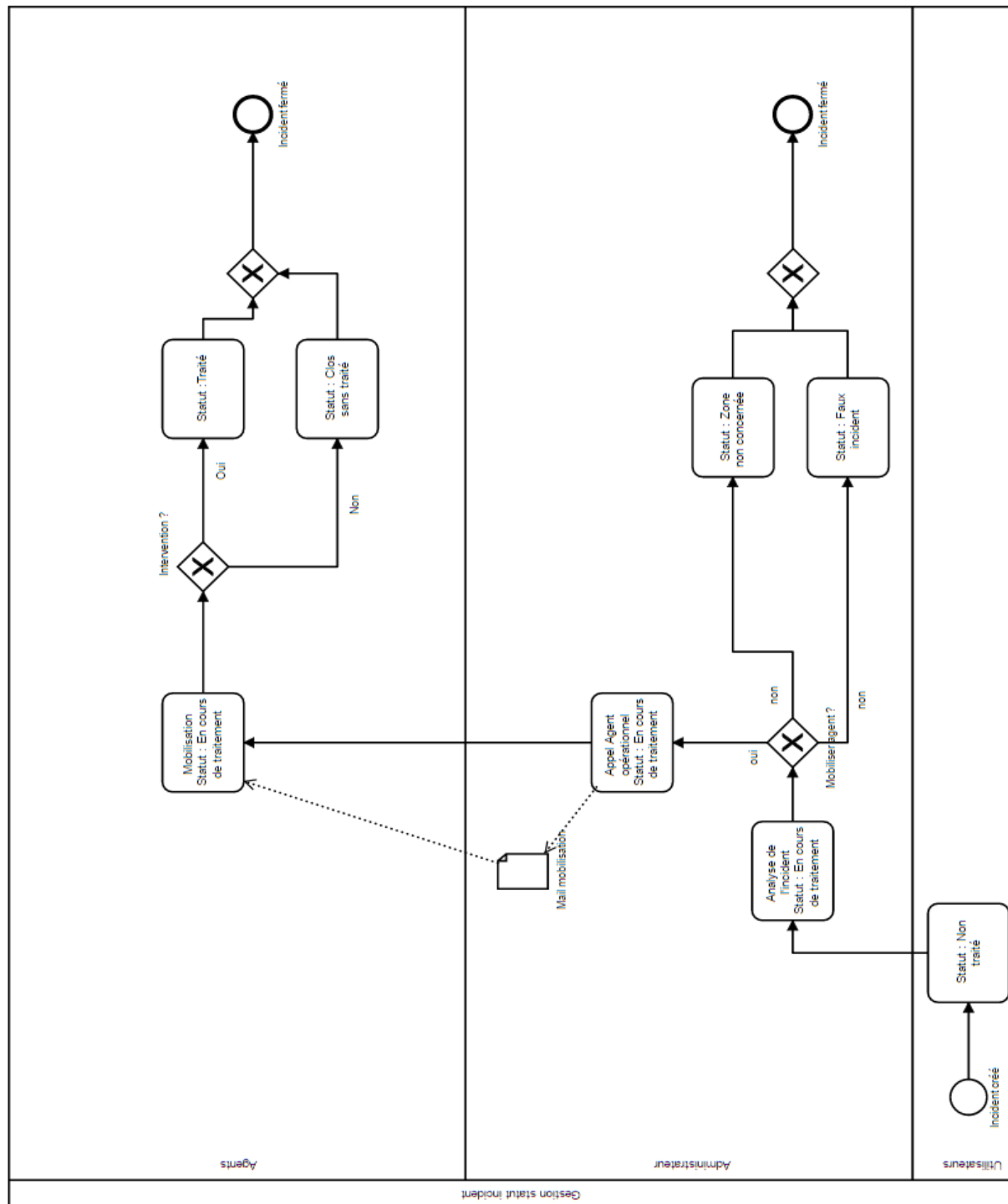
Diagramme d'activité d'un signalement



## Statut Incident

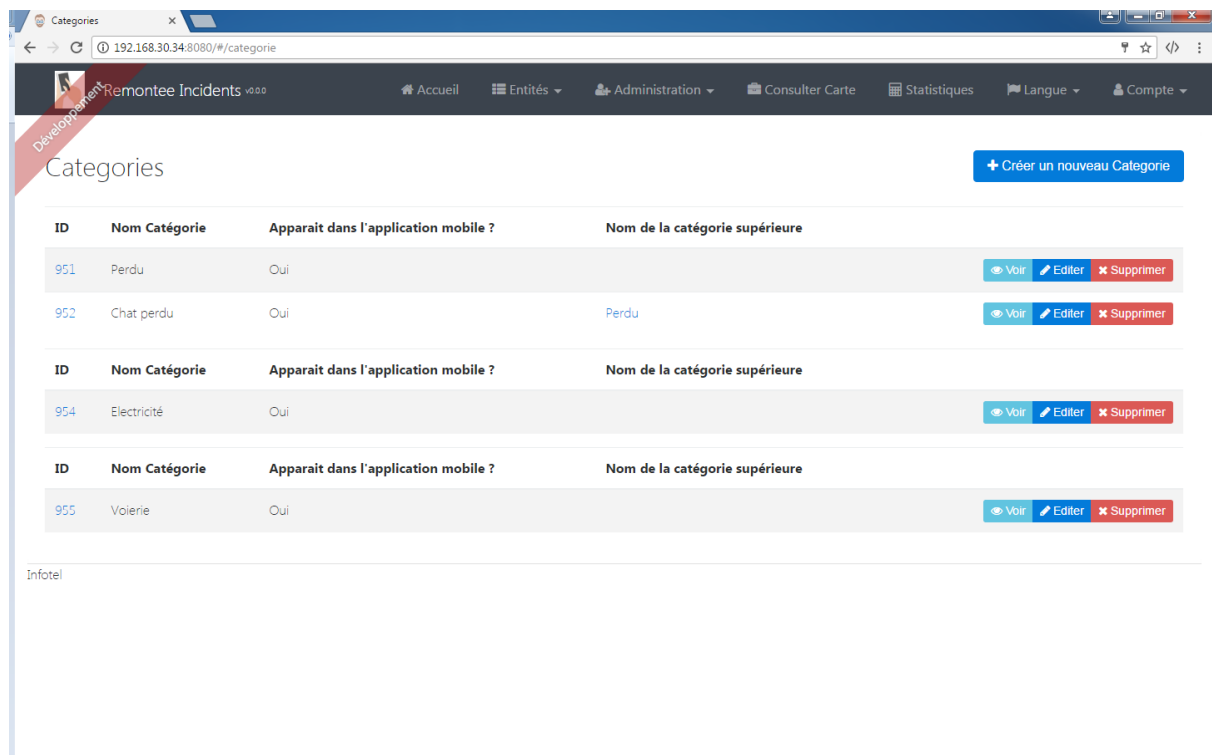
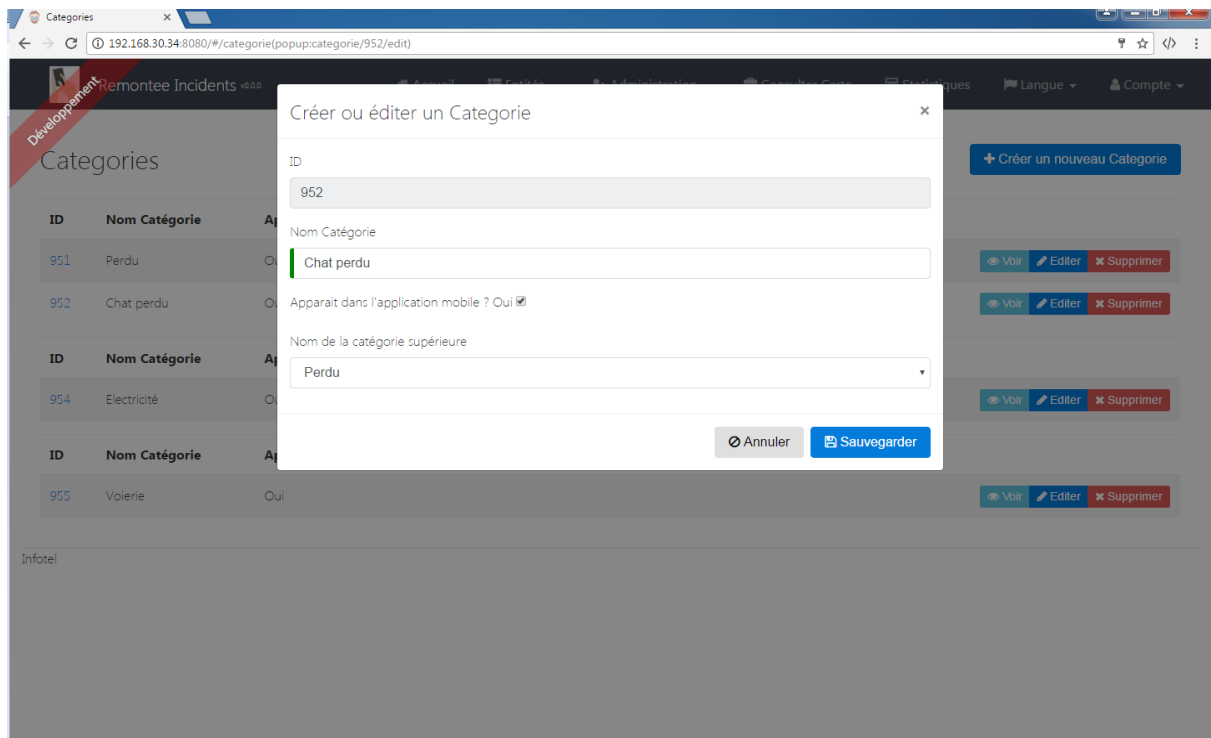
Le statut d'un incident permettrait de différencier plus facilement et rapidement quels sont les nouveaux incidents, les incidents en cours de traitement, et enfin les incidents déjà traité. Et ainsi d'éviter le risque de retraiter un incident clos.

Seul l'administrateur pourra changer le statut d'un incident et un incident créé est automatiquement sous le statut « non traité ». En découle le workflow suivant :





## Les Catégories



## Les Incidents

Incidents

Remontee Incidents v0.0.0

Accueil Entités Administration Consulter Carte Statistiques Langue Compte

### Incidents admin

+ Créer un nouveau Incidents

Paramètres avancés

ID	Date	Message	Latitude	Longitude	Statut Incident	Priorite	Catégorie	
1001	Jun 9, 2017	On a retrouvé un chat	43.630703	1.489991	Traité	1	Chat perdu	<a href="#">Voir</a> <a href="#">Editer</a> <a href="#">Supprimer</a>
1002	Jul 16, 2017	Problème de courant	43.630548	1.490466	En cours	2	Electricité	<a href="#">Voir</a> <a href="#">Editer</a> <a href="#">Supprimer</a>
1003	Jun 26, 2017	Encore un chat perdu	43.63061	1.48891	Non traité	1	Chat perdu	<a href="#">Voir</a> <a href="#">Editer</a> <a href="#">Supprimer</a>
1004	Jun 29, 2012	Le réseau a sauté	43.630295	1.488309	Incident fermé	1	Electricité	<a href="#">Voir</a> <a href="#">Editer</a> <a href="#">Supprimer</a>
1005	May 14, 2014	Le feu rouge ne marche plus	43.62967	1.487606	Faux incident	1	Voierie	<a href="#">Voir</a> <a href="#">Editer</a> <a href="#">Supprimer</a>
1006	May 6, 2017	Instable	43.629763	1.491345	En cours	4	Electricité	<a href="#">Voir</a> <a href="#">Editer</a> <a href="#">Supprimer</a>
1007	Dec 25, 2014	Coupure de courant	43.629782	1.488518	Traité	5	Electricité	<a href="#">Voir</a> <a href="#">Editer</a> <a href="#">Supprimer</a>

Infotel

Incidents

Remontee Incidents v0.0.0

Accueil Entités Administration Consulter Carte Statistiques Langue Compte

### Incidents 1007

**Date**  
Dec 25, 2014

**Message**  
Coupure de courant

**Latitude**  
43.629782



**Longitude**  
1.488518

**Statut Incident**  
Traité

**Priorite**  
5

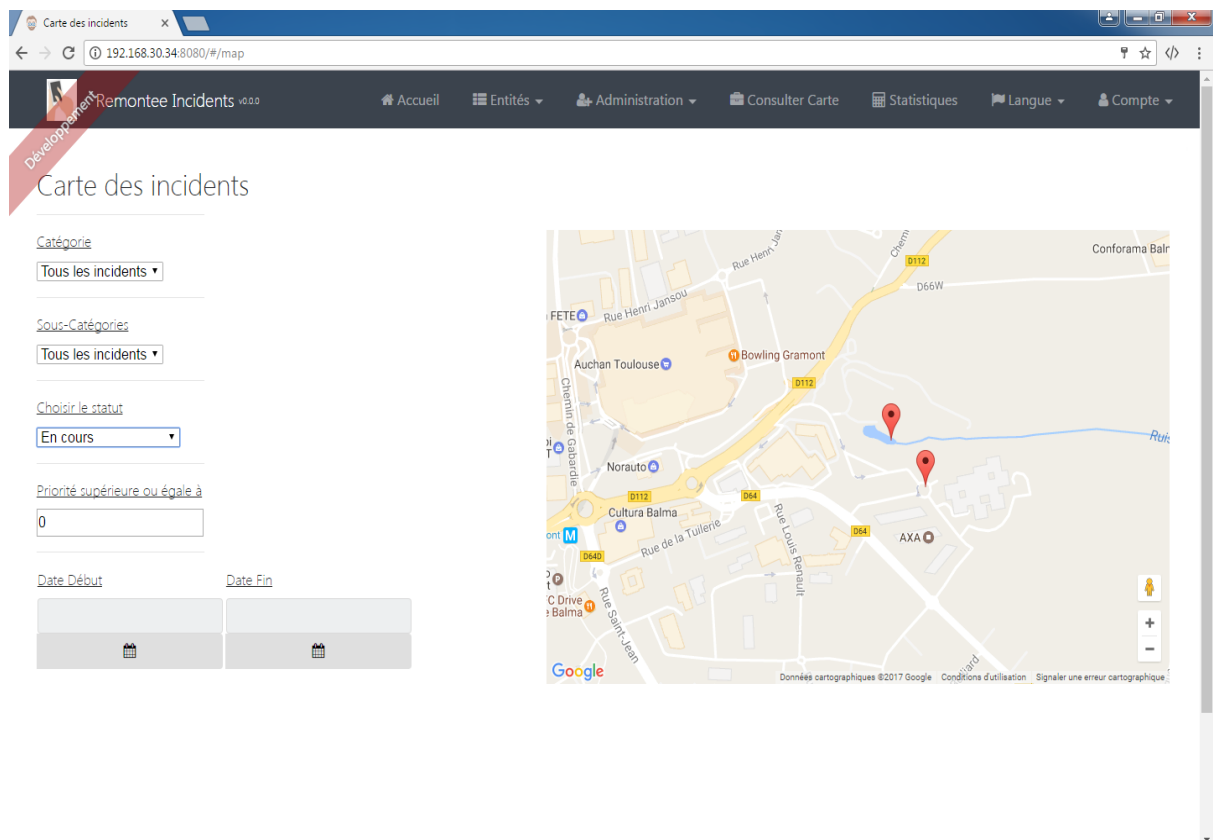
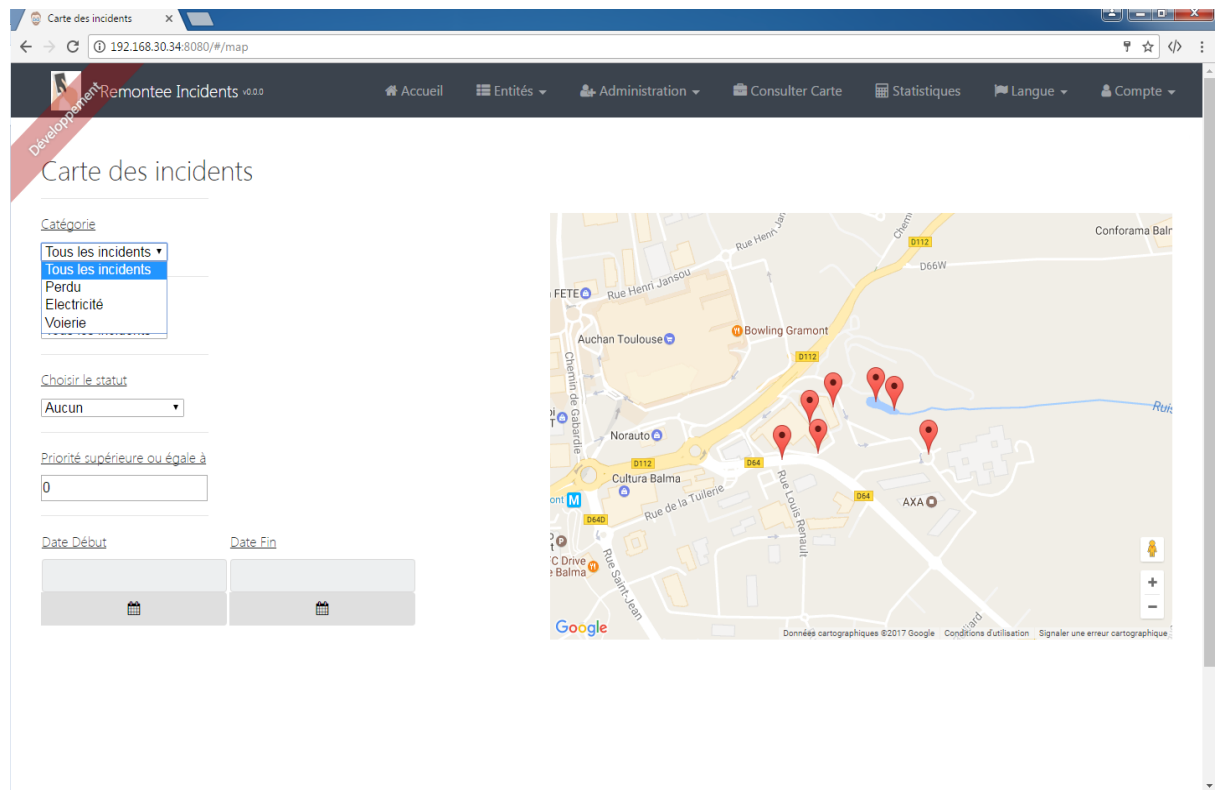
**Catégorie**  
Electricité

[Retour](#) [Editer](#) [Back](#) [Next](#)



Infotel



## Les Cartes



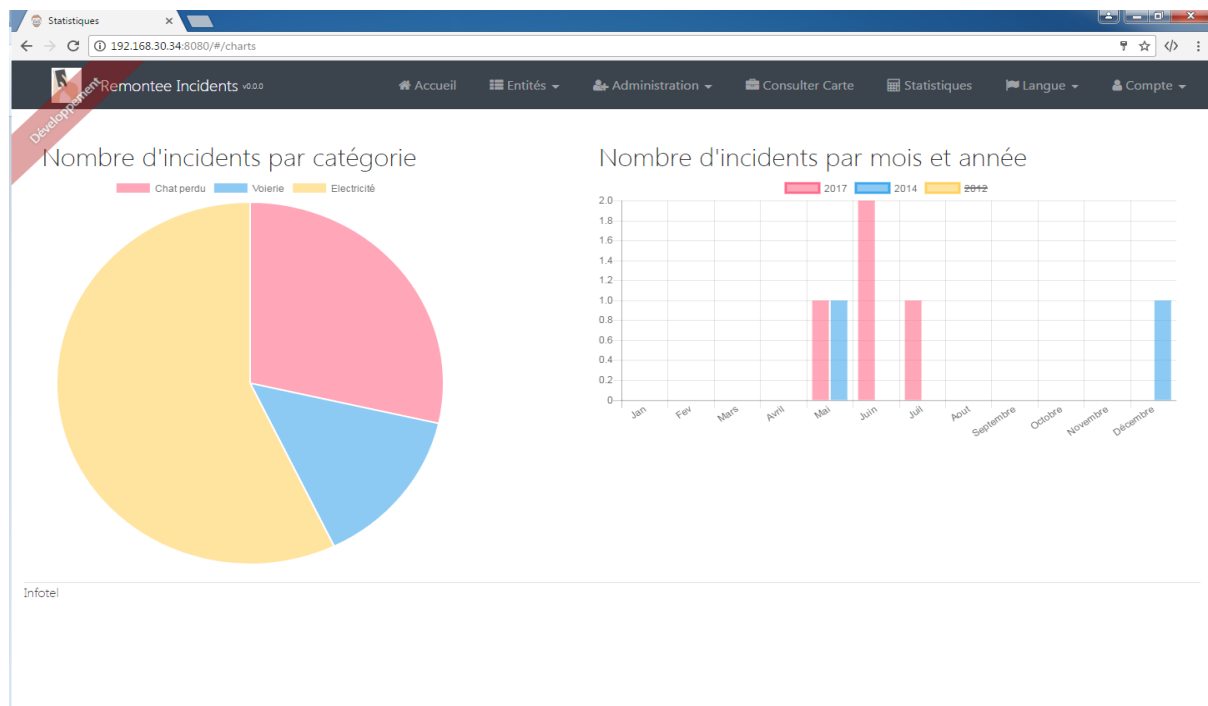
## Les Photographies et statistiques

Photos

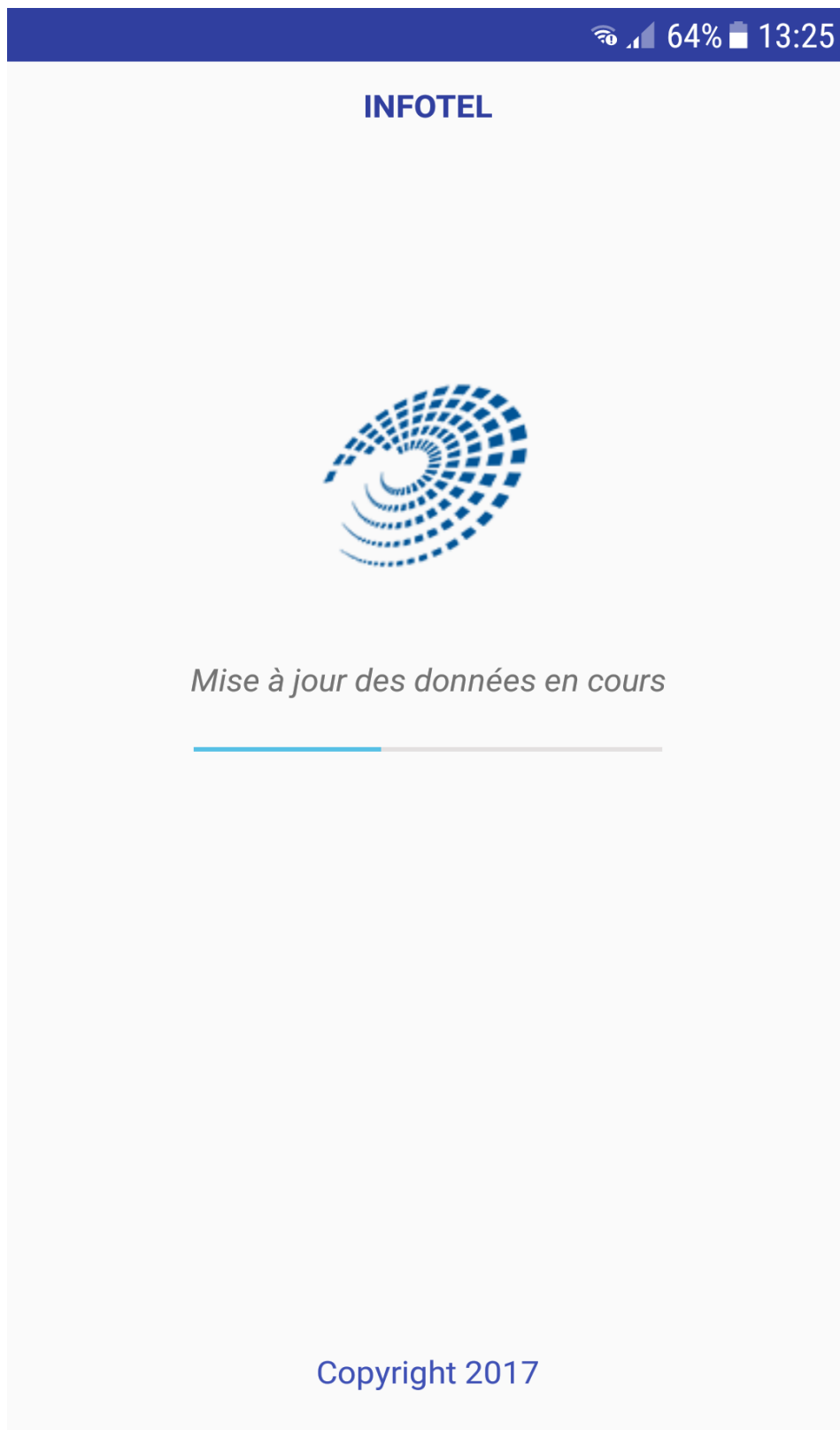
+ Créer un nouveau Photo

ID	Photo	Id Inc	
1051		1007	<a href="#">Voir</a> <a href="#">Editer</a> <a href="#">Supprimer</a>
1052		1007	<a href="#">Voir</a> <a href="#">Editer</a> <a href="#">Supprimer</a>

Infotel



## L'application mobile








4G+ 64% 13:25




ANNULER


SUIVANT

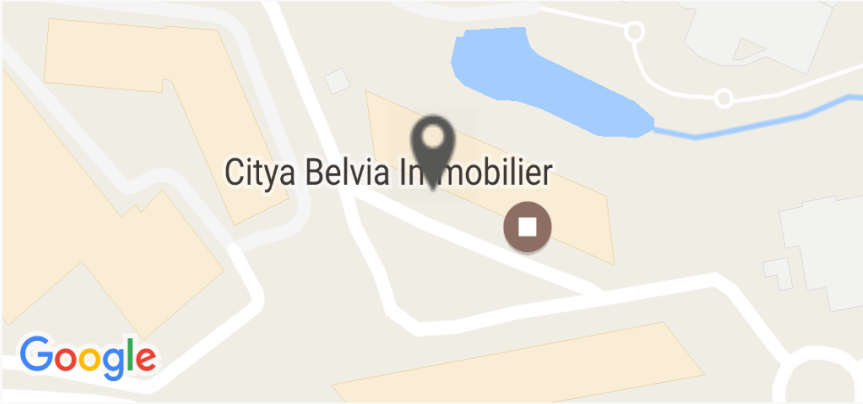



4G+ 64% 13:26

Catégorie du signalement 


Chat perdu

Adresse 



Message 

mon chat a disparu



ANNULER

ENVOI



