

Membres du groupe :

SOMMAIRE

Introduction	5
A. ANALYSE	6
I. CONTEXTE ET DEFINITION DU PROBLEME	6
II. OBJECTIF	6
III. ETUDE DE L'EXISTENCE	6
1. Analyse de la Maison des retrouvailles de Douala	7
<i>a. Fonctionnement</i>	7
<i>b. AVANTAGES ET CRITIQUES TECHNIQUES</i>	7
<i>i. AVANTAGES</i>	7
<i>ii. CRITIQUES TECHNIQUES</i>	7
2. Troov.com	7
<i>a. AVANTAGES ET CRITIQUES</i>	8
<i>i. AVANTAGES</i>	8
<i>ii. CRITIQUES FONCTIONNELLES</i>	8
IV. DESCRIPTION FONCTIONNELLE DES BESIENS	8
1. Besoin fonctionnel	8
2. Besoins non fonctionnels	9
B. CONCEPTION	10
I. ARCHITECTURE DU SYTEME	10
II. DESAGNE ET CHARTE GRAPHIQUE	10
III. ARBORESCEN DE L'APPLICATION	11
C. MODELISATION	12

I. DIAGRAMME DES CAS D'UTILISATION	12
1. ACTEURS ET ROLES	12
2. CAS D'UTILISATION GLOBAL	12
3. DESCRIPTION DES CAS D'UTILISATION	12
a. Figure : cas d'utilisation global	14
II. DIAGRAMME DE SEQUENCE	14
III. DIAGRAMME D'ETAT TRANSITION	16
IV. DIAGRAMME DE CLASSE	17
D. PRESENTATION DES INTERFACES DE L'APPLICATION	19
I. INTERFACES DE L'APPLICATION	19
1. Présentation de l'interface homme machine	19
2. Page d'authentification	19
3. Page de conseil ou de bonnes pratique	21
4. La page d'accueil	21
II. TECHNOLOGIES	23
1. LANGAGE DE MODELISATION	23
2. LANGAGE UTILISE	23
3. SGBD : MongoDB	23
4. PROCESSUS DE DEVELOPPEMENT	24

Introduction

Un objet perdu est une possession qui a été égarée par son propriétaire de manière involontaire. Au Cameroun on dénombre plusieurs milliers d'objets perdus chaque année. Bien que certains mécanismes archaïques misent sur pieds par des personnes de bonne foi possédant un objet égaré pour retrouver le propriétaire portent souvent des fruits, on dénombre cependant un grand nombre d'objets dont il n'a pas été possible de retrouver le propriétaire. C'est en considérant ce second aspect que nous avons eu l'idée de concevoir une plateforme dont le rôle principal sera d'aider des personnes qui ont égaré un objet à pouvoir le retrouver.

A. ANALYSE

I. CONTEXTE ET DEFINITION DU PROBLEME

Dans la vie courante nous perdons chaque jour des objets et ne les retrouvons dans la plupart des cas jamais ce qui peut nous mettre dans des situations inconfortables, et parfois ces objets sont retrouvés et jetés à cause du manque d'un canal pour relier celui qui a retrouvé l'objet et celui qui l'a perdu. D'où la nécessité de créer une application de recherche d'objets perdus ce qui vient résoudre en partie ce problème qui nous touche tous.

II. OBJECTIF

Nous voulons fournir une solution à ce problème de recensement d'objets perdus. Pour notre cadre expérimentale on prendra la ville de Dschang. L'objet principal de notre projet est de permettre aux gens de pouvoir retrouver les objets qu'ils perdent plus facilement grâce à des personnes qui les auront retrouvés et qui auront posté ces objets sur notre plateforme. C'est dans cette optique que nous proposons la plateforme **FreeLost**.

III. ETUDE DE L'EXISTENCE

Afin d'approfondir notre compréhension du sujet et avoir une idée plus claire sur notre projet et ses fonctions attendues, nous avons mené une étude sur des organisations de recensement des objets perdus qui s'inscrivent dans le même cadre que notre travail.

Nous analysons dans un premier temps les moyens de résolutions de ce problème pris au niveau du Cameroun comme le cas de la Maison des retrouvailles de Douala.

1. Analyse de la Maison des retrouvailles de Douala

Au niveau du Cameroun il n'existe vraiment pas de plateforme (site web, App mobile) spécialisée dans la gestion des Objets perdu, néanmoins d'après nos recherches il existe une petite structure appelée la Maison des retrouvailles de Douala qui collecte les objets trouvés et tente de retrouver leurs propriétaires. Depuis 1998, une équipe de jeunes bénévoles classe cartes d'identité, permis de conduire et trousseaux de clés. Aujourd'hui, le stock atteint les 50 000 objets collectes, mais la structure n'étant pas visible sur internet ne peut vraiment servir un grand nombre de personnes, car tout le monde n'a pas le temps d'y faire un tour pour rechercher une pièce perdue. De plus le système n'étant pas informatisé, il est difficile de rechercher un objet et on est jamais sûr a priori si on le trouvera. C'est dans l'optique de vouloir palier à ces problèmes que nous voulons mettre sur pied notre plateforme.

a. Fonctionnement

Des gens perdent des objets qui sont retrouvés par d'autres et ceux-ci se rendent à la maison des retrouvailles de Douala qui est vue comme un centre de collecte pour déposer l'objet en donnant des informations sur le lieu où l'objet a été trouvé. Les responsables du centre de collecte vont ranger l'objet selon sa catégorie et relevés les caractéristiques signifiant de l'objet.

b. AVANTAGES ET CRITIQUES TECHNIQUES

i. AVANTAGES

Comme avantages nous avons :

- ✓ Facilite la recherche d'un objet
- ✓ Réduit la zone de recherche
- ✓ Augmente les chances de retrouver un objet perdu

ii. CRITIQUES TECHNIQUES

- ✓ Le stockage des objets nécessite beaucoup de place
- ✓ L'enregistrement des informations sur l'objet est par écrit
- ✓ L'absence de visibilité web du centre de collecte freine la récupération de certains objets et réduit les possibilités de se faire connaître par la population

2. Troov.com

C'est un site web mondial de recensement des objets perdus.

a. AVANTAGES ET CRITIQUES

i. AVANTAGES

- ✓ Le site offre la possibilité de poster des objets perdus
- ✓ Le site offre la possibilité de signaler la perte d'un objet en donnant sa description
- ✓ Facilite la recherche
- ✓ Offre de l'emploi

ii. CRITIQUES FONCTIONNELLES

///

IV. DESCRIPTION FONCTIONNELLE DES BESIONS

Ce terme peu évocateur pour les non-initiés se résume simplement : à décrire notre besoin en termes de fonctionnalités. Expliquez ce que doit faire notre application. C'est l'outil de base pour la réalisation. En nous basant sur ces différents cas d'étude nous pouvons en ressortir les besoins suivants :

1. Besoin fonctionnel

Il s'agit des besoins qui mettent en évidence les fonctions de services (A quoi ça sert ?) et les fonctions techniques (Comment cela peut marcher ?). Pour notre Système nous pouvons citer :

- ✓ L'application permettra à un utilisateur de faire une déclaration (d'objet perdu ou trouvé).
- ✓ Après chaque déclaration, l'application doit pouvoir faire des correspondances dans le système, si l'objet correspond alors le propriétaire est notifié.
- ✓ L'application doit supprimer les déclarations des objets perdus ayant déjà été retrouvé et restitué aux propriétaires.
- ✓ L'application doit publier des images ou des informations sur les objets perdus
- ✓ L'application doit permettre aux utilisateurs de pouvoir rechercher eux-mêmes les objets.
- ✓ L'application doit proposer une fonctionnalité chat pour permettre aux utilisateurs de discuter entre eux.
- ✓ Avec la présence des centres de dépôt ou de collecte, l'application devra géo localiser les centres de collectes les plus proches d'un utilisateur quelconque ceci dans le but de lui faciliter la tâche.

- ✓ L'application doit pouvoir vérifier le flux d'entrée et de sortie d'un compte pour le rendre spécial(collecteur) si le flux est important.
- ✓ La gestion des conseils : l'application doit pouvoir donner des conseils aux différents utilisateurs pour éviter tout désagrément lors de la récupération de leur objet.
- ✓ La gestion des annonces : l'application dite permet aux utilisateurs de poster ou de supprimer une annonce (offre d'emploi, bourses, formations...)

2. Besoins non fonctionnels

Il s'agit des besoins qui caractérisent le système. Ce sont des besoins en matière De performance, de type de matériel ou le type de conception. Les besoins non fonctionnels sont importants car ils agissent de Façon indirecte sur le résultat et sur le rendement de l'utilisateur, ce qui Fait qu'ils ne doivent pas être négligés, pour cela il faut répondre aux Exigences suivantes :

- ✓ **Fiabilité** : L'application doit fonctionner de façon cohérente sans erreurs et doit être satisfaisante.
- ✓ **Les erreurs** : Les ambiguïtés doivent être signalées par des messages d'erreurs bien organiser pour bien guider l'utilisateur et le familiariser avec notre site web.
- ✓ **Ergonomie et bonne Interface** : L'application doit être adaptée à l'utilisateur sans qu'il ne fournisse aucun effort (utilisation claire et facile) de point de vue navigation entre Les différentes pages, couleurs et mise en textes utilisés.
- ✓ **Sécurité** : Notre solution doit respecter surtout la confidentialité des données personnelles des clients qui reste l'une des contraintes les plus importantes dans les sites web. Notamment toutes les informations sur un objet trouvé ne devraient pas être communiquées, pour prouver l'appartenance de l'objet le propriétaire pourra renseigner une question d'authentification lors la déclaration de perte, cette question concerne un détail précis sur l'objet qui prouvera sous bonne foi l'appartenance à ce dernier.
- ✓ **Aptitude à la maintenance et la réutilisation** : Le système doit être conforme à une architecture standard et claire permettant sa maintenance et sa réutilisation.
- ✓ **Compatibilité et portabilité** : Un site web quel que soit son domaine, son éditeur et son langage de programmation ne peut être fiable qu'avec une compatibilité avec tous les navigateurs web et tous les moyens que ce soit PC, IPAD ou Mobiles.

B. CONCEPTION

I. ARCHITECTURE DU SYTEME

Notre application est basée sur le **MVC (Modèle-Vue-Contrôleur)**. Le pattern MVC permet de bien organiser son code source. Il va vous aider à savoir quels fichiers créer, mais surtout à définir leur rôle. Le but de MVC est justement de séparer la logique du code en trois parties que l'on retrouve dans des fichiers distincts.

- **Modèle** : cette partie gère les données de votre site. Son rôle est d'aller récupérer les informations « brutes » dans la base de données, de les organiser et de les assembler pour qu'elles puissent ensuite être traitées par le contrôleur. On y trouve donc entre autres les requêtes SQL ;
- **Vue** : cette partie se concentre sur l'affichage. Elle ne fait presque aucun calcul et se contente de récupérer des variables pour savoir ce qu'elle doit afficher. On y trouve essentiellement du code HTML mais aussi quelques boucles et conditions PHP très simples, pour afficher par exemple une liste de messages ;
- **Contrôleur** : cette partie gère la logique du code qui prend des *décisions*. C'est en quelque sorte l'intermédiaire entre le modèle et la vue : le contrôleur va demander au modèle les données, les analyser, prendre des décisions et renvoyer le texte à afficher à la vue. Le contrôleur contient exclusivement du PHP. C'est notamment lui qui détermine si le visiteur a le droit de voir la page ou non (gestion des droits d'accès).

II. DESAGNE ET CHARTE GRAPHIQUE

Notre site étant d'origine Camerounaise, nous avons donc choisi d'utiliser les couleurs du drapeau : vert, rouge, jaune. C'est dans cette même lancée que nous avons fait le choix de couleurs du logo qui se matérialise comme suite :



III. ARBORESCEN DE L'APPLICATION

Ici nous présentons la structuration de nos pages web et mobile

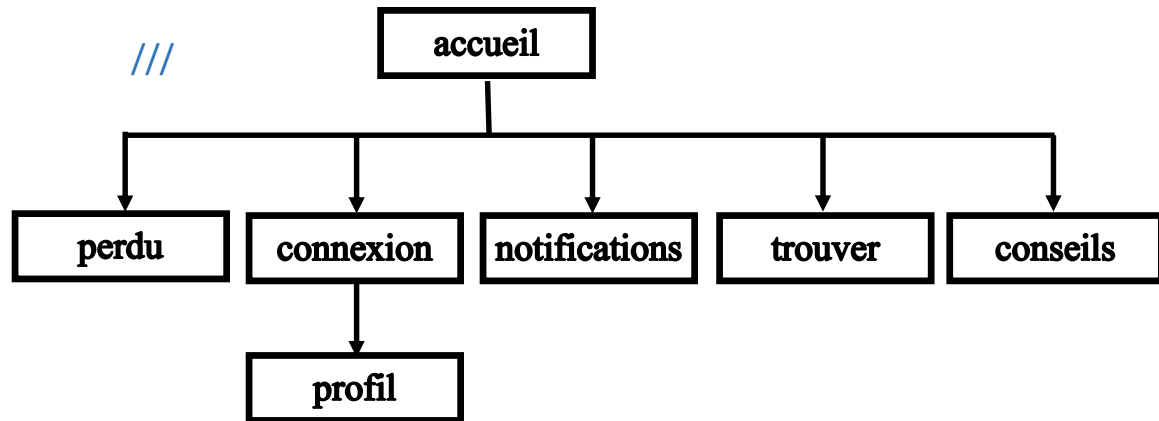


Figure : arborescence de la page

C. MODELISATION

I. *DIAGRAMME DES CAS D'UTILISATION*

1. *ACTEURS ET ROLES*

Un acteur est un agent humain ou logiciel externe (au système) qui interagit avec le système. Pour notre application en cours de développement, nous avons recensé les acteurs suivants :

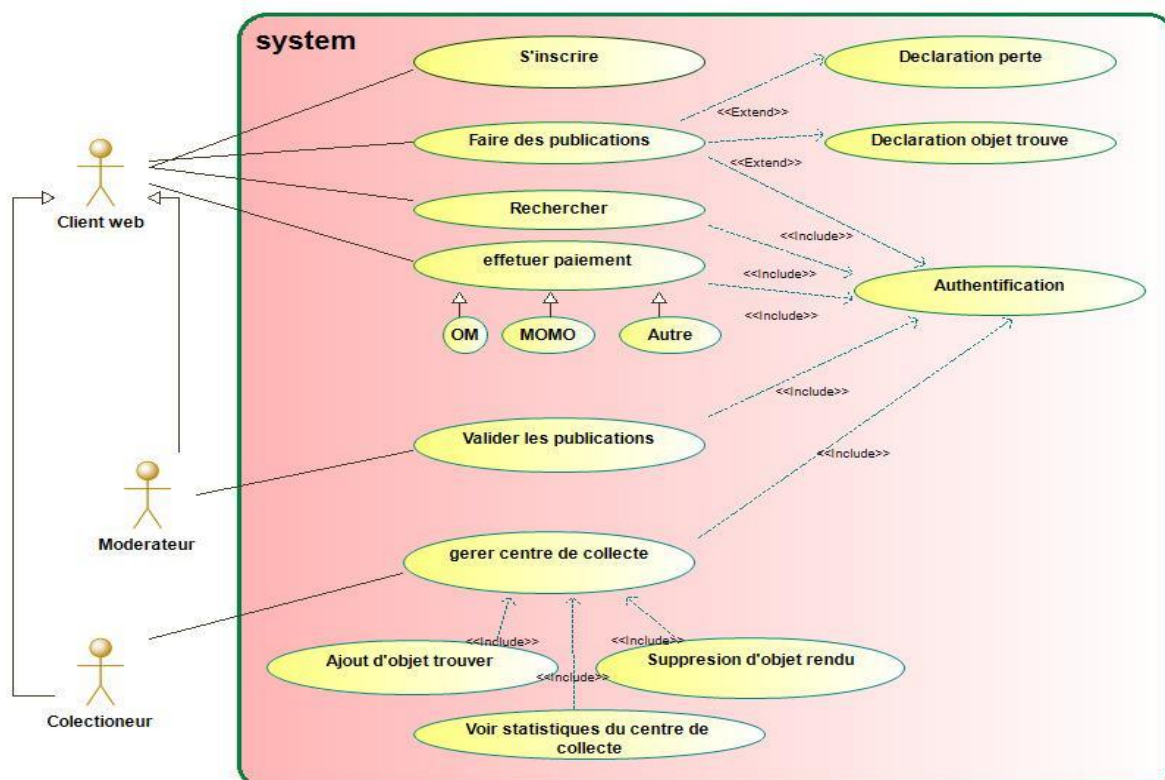
- **Les utilisateurs :**
 - ✓ **Collecteur** : c'est une personne qui gère un centre de collecte où les objets perdus seront déposés par les trouveurs.
 - ✓ **Client** : peut-être une personne ayant perdu un objet ou une personne ayant trouvé un objet perdu.
 - ✓ **Visiteur** : c'est une personne qui vient visiter la page
- **Administrateur** : personne qui dispose de tous les privilèges sur la plateforme et assure la gestion de celle-ci.
- **Modérateur** : s'occupe de la validation des différentes publications sur la page.

2. *CAS D'UTILISATION GLOBAL*

Il montre les différentes actions qu'un utilisateur peut effectuer sur le système. Ici nous montrons l'interaction qui existe entre le client, modérateur

3. *DESCRIPTION DES CAS D'UTILISATION*

Acteurs	Cas utilisation
Client	<ul style="list-style-type: none"> ✓ Création de compte ✓ Publication d'avis de recherche ✓ Publication d'objet trouver ✓ Effectue des paiements ✓ Souscrire à un abonnement
Collecteur	<ul style="list-style-type: none"> ✓ Création de compte ✓ Publication d'objets perdus ✓ Signal les objets retrouvés ✓ Effectue des paiements
Administrateur	<ul style="list-style-type: none"> ✓ Gère les utilisateurs ✓ Création d'un compte ✓ Gère les publications ✓ Effectue des publications ✓ Gère les statistiques des collectes ✓ Gère les abonnements
Banque	Gère les modes de paiement : <ul style="list-style-type: none"> ✓ MTN Mobile Money ✓ ORANGE Mobile Money ✓ Crypto monnaie
Modérateur	<ul style="list-style-type: none"> ✓ Gère la validation des publications



a. Figure : cas d'utilisation global

II. DIAGRAMME DE SEQUENCE

C'est la représentation graphique de l'interaction entre l'acteur et le système selon un ordre chronologique dans la formulation Unifié Mödling Langage (UML). Quelques-uns sont représentés ci-dessous.

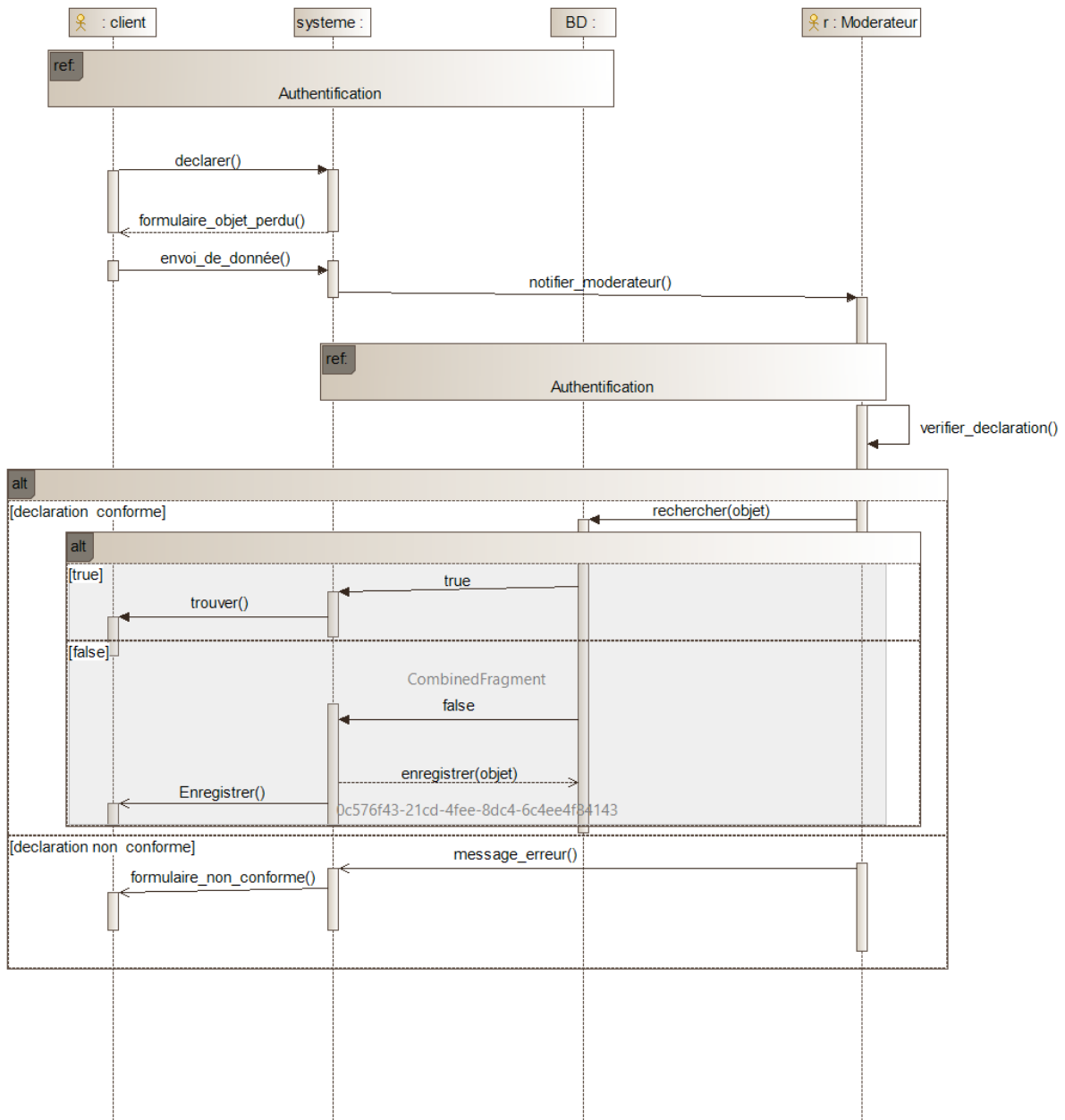


Figure a : diagramme de séquence d'objets perdus

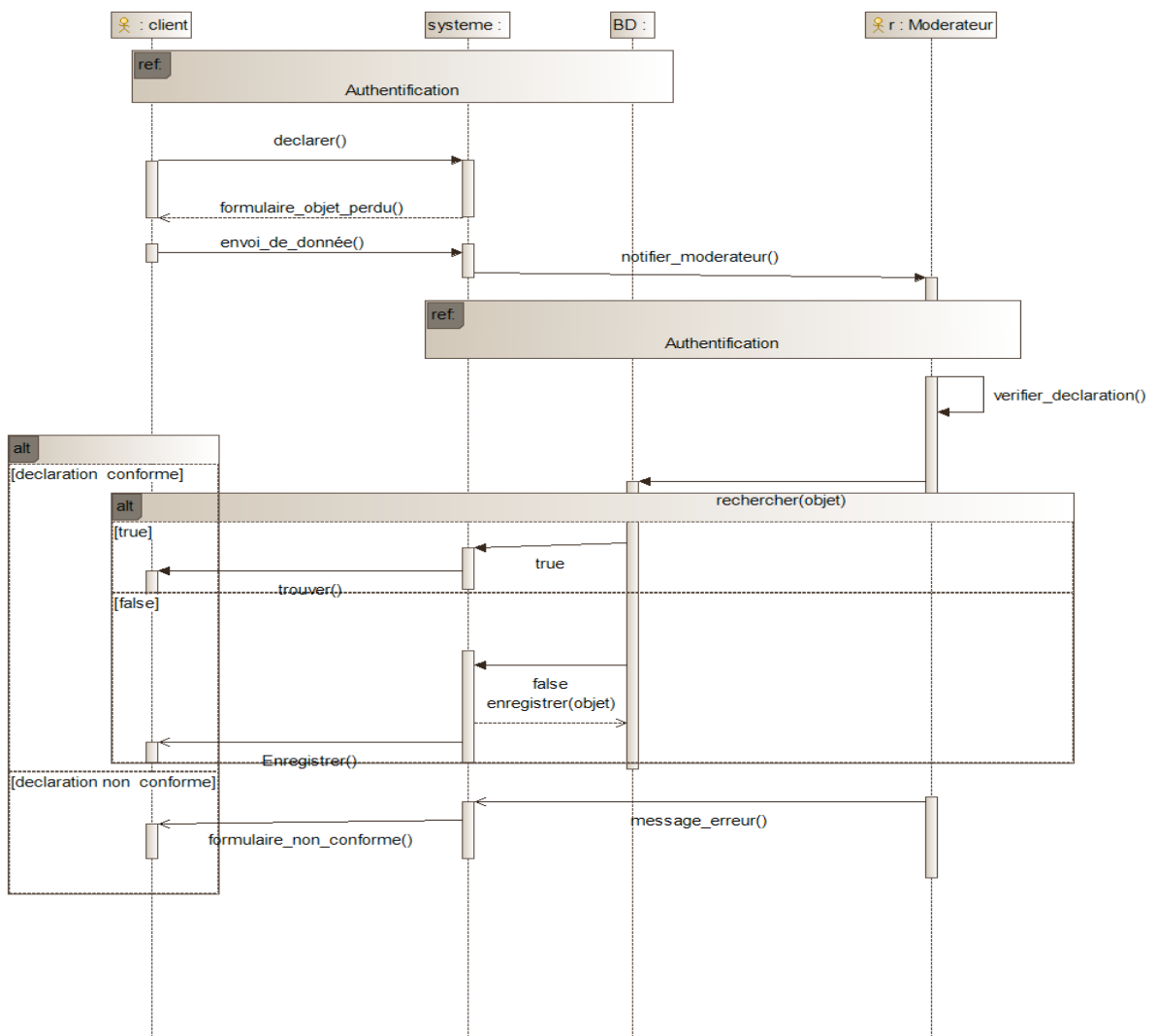


Figure b : diagramme de séquence d'objets trouvés

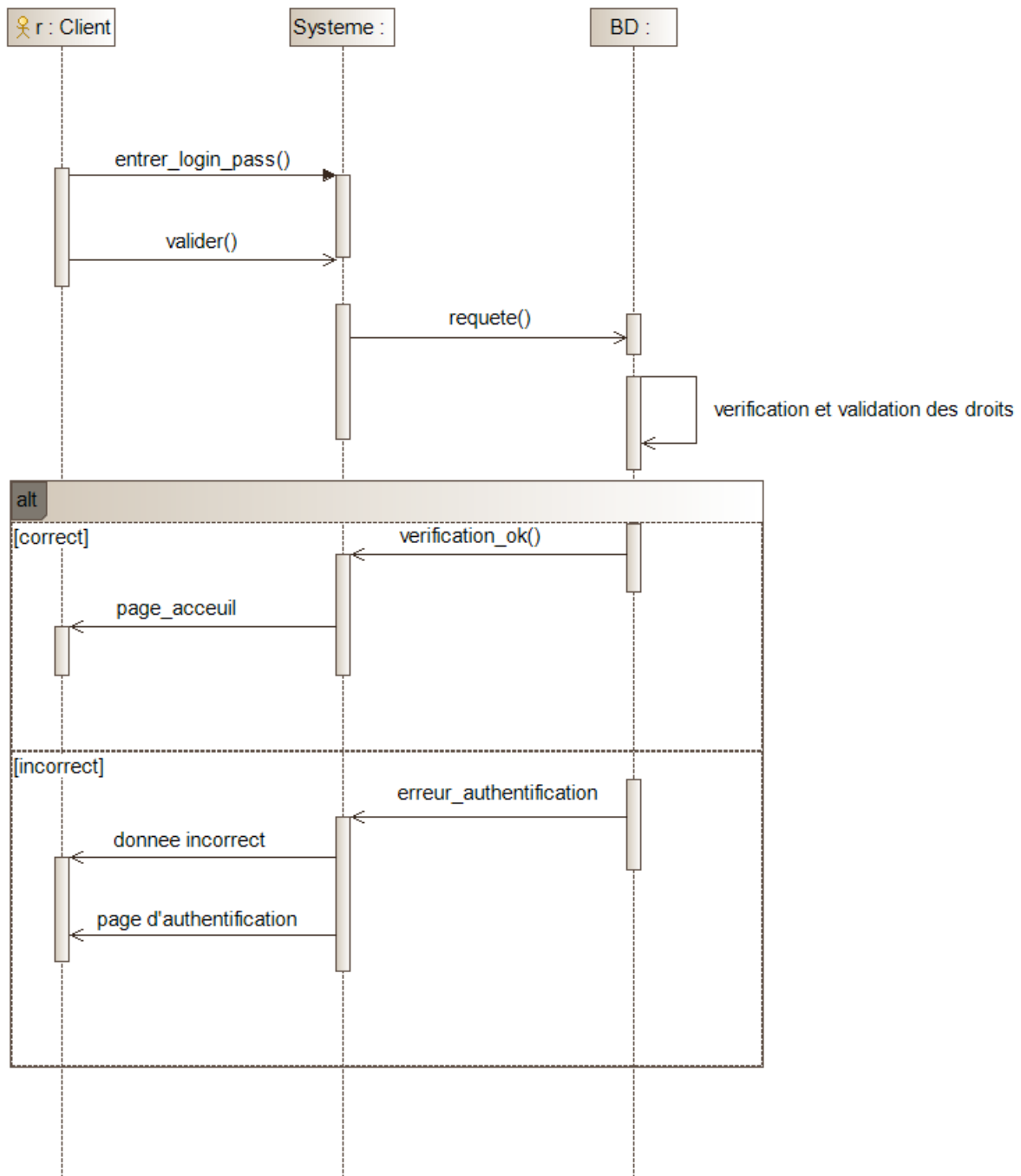


Figure c : diagramme de séquence d'une authentification

III. DIAGRAMME D'ETAT TRANSITION

Le diagramme d'état transition permet de spécifier les réactions d'une partie du système à des évènements discrets. Dans ce diagramme, lorsqu'un évènement est reçu, une transition peut être déclenchée et faire basculer l'objet dans un nouvel état.

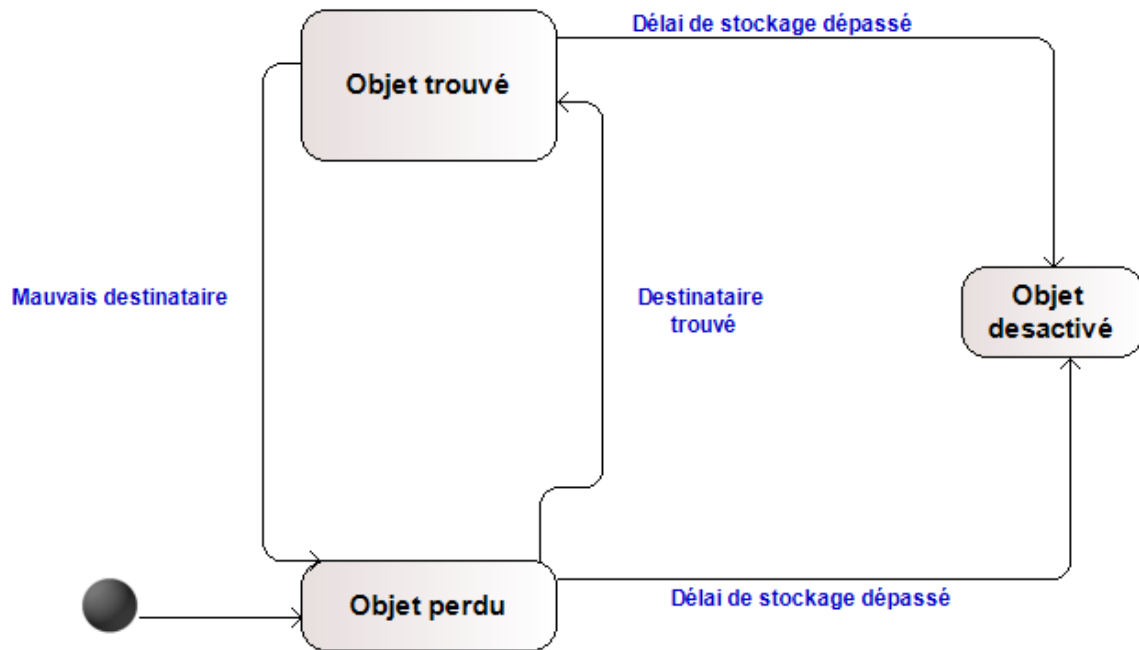


Figure d : diagramme d'état transition d'un objet

IV. DIAGRAMME DE CLASSE

Le diagramme de classe est un schéma utilisé en génie logiciel pour présenter les classes et les interfaces des systèmes ainsi que leurs relations.

Une classe est un ensemble de fonctions et de données (attributs) qui sont utilisées dans la programmation orientée objet.

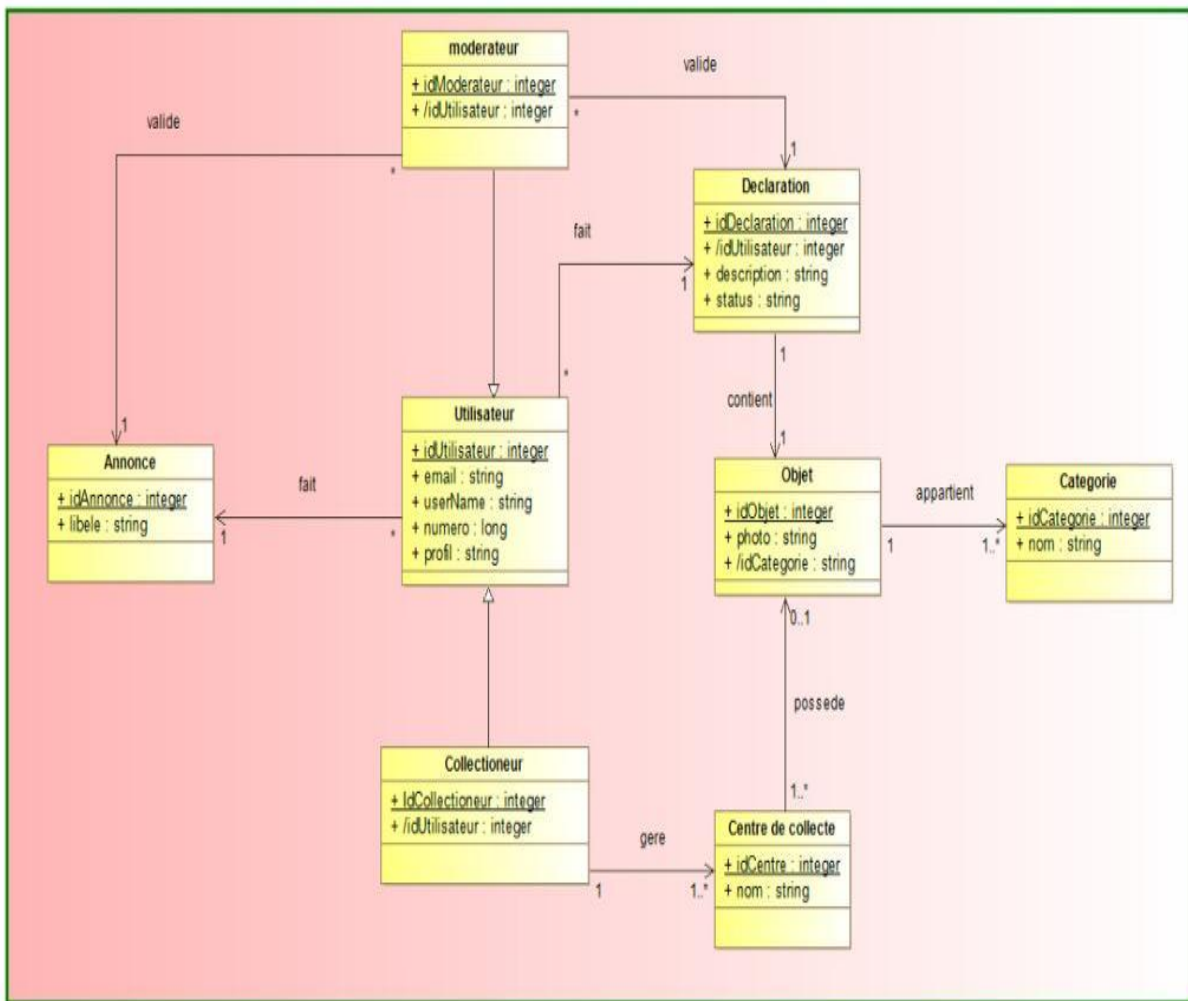


Figure 1: Diagramme de classe

D. PRESENTATION DES INTERFACES DE L'APPLICATION

I. INTERFACES DE L'APPLICATION

1. Présentation de l'interface homme machine

L'interface Homme-machine désigne l'ensemble des interfaces et fenêtres permettant de manipuler l'application. Elle représente un élément important dans l'utilisation de tout système informatique et conditionne également son succès. L'interface doit permettre une compréhension et une utilisation facile de l'application. Pour cela, elle doit être intelligente, fiable, efficace et naturelle.

2. Page d'authentification

pour pouvoir avoir accès à plus de fonctionnalités sur notre page l'utilisateur doit se créer un compte ou se connecter s'il en a déjà ceci grâce à la fenêtre suivante dans laquelle l'utilisateur entrera ses paramètres de connexion (authentification) pour son accès.

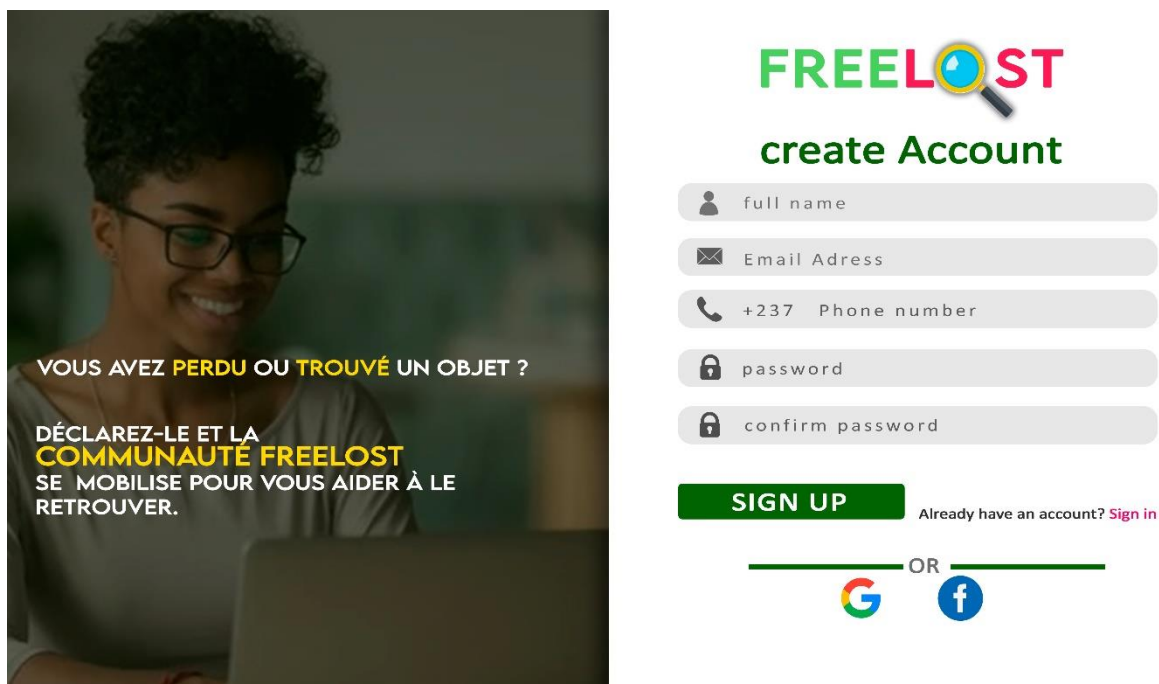


Figure a : Page d'inscription pour le web



FREELOST

LOGIN

LOGIN

create an account : [Sign up](#)

OR




 

Figure b : Page de connexion pour le web





Login Account

[Forgot password?](#)

Login

OR

Don't have account? [REGISTER](#)

Create Account
Create a new account

Sign Up

Already have an account? [Sign in!](#)

Figure c : Page de connexion et d'inscription pour le mobile

3. Page de conseil ou de bonnes pratique

Une création réussie d'un compte utilisateur conduit automatiquement à une page de conseil pour une bonne pratique dans le cas du mobile

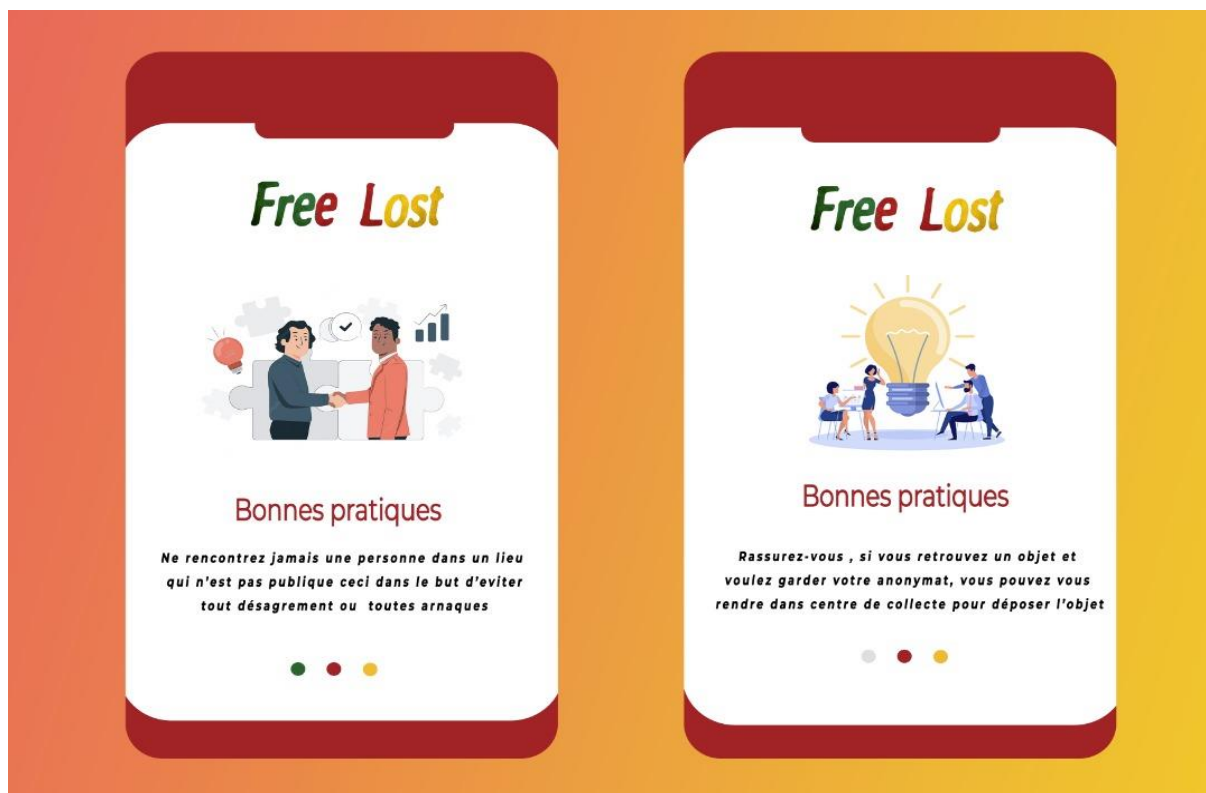


Figure 15-1 : Page de conseil pour une bonne pratique pour le mobile

Dans le cas du web elle se présente comme suite

4. La page d'accueil

Elle nous présente toutes les ouvertures chemins qu'on peut prendre en quittant de là



Figure 16-1 : Page d'accueil pour le web

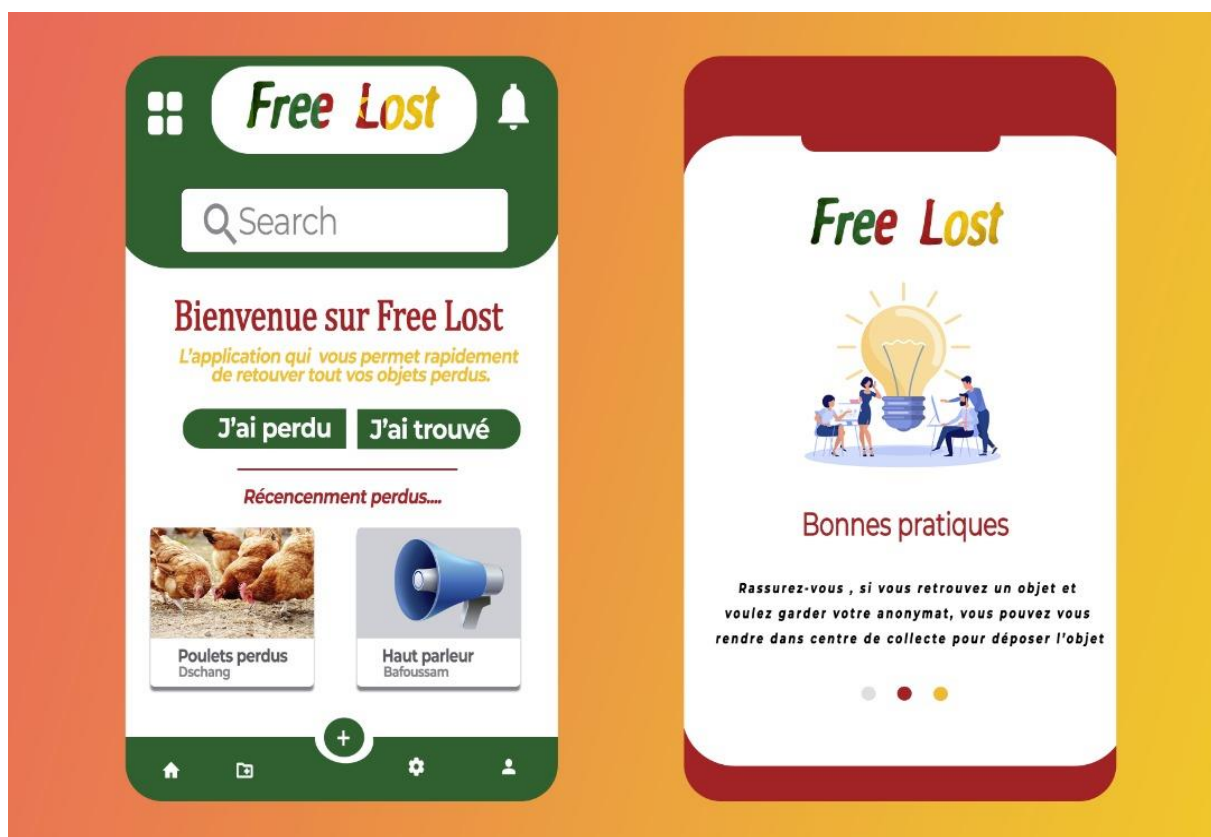


Figure 16-2 : Page d'accueil pour le mobile

II. TECHNOLOGIES

Afin de mener à bien ce projet, nous avons utilisé un ensemble de matériels dont les principales caractéristiques sont les suivantes :

1. LANGAGE DE MODELISATION

La création d'un système d'information n'est jamais évidente et c'est même inconcevable sans bonne méthode. C'est dans cette logique que s'inscrit la méthode UML qui permet de structurer et d'organiser les données de manière optimale tant au niveau de l'exploitation des données qu'au niveau des améliorations et de la maintenance de la base de données.

2. LANGAGE UTILISE

Dans ce projet nous avons choisi d'utiliser comme technologie :

- ✓ Pour le front-end de la plateforme web nous allons utiliser le Framework JavaScript **Vue JS**
- ✓ Pour le front-end de notre plateforme mobile nous allons utiliser le Framework JavaScript **React Native**
- ✓ Pour le Back-end, les deux utiliseront une API que nous avons codé avec le Framework **Node JS**

3. SGBD : MongoDB

Il permet de manipuler des objets structurés au format BSON (JSON binaire), sans schéma prédéterminé. En d'autres termes, des clés peuvent être ajoutées à tout moment « à la volée », sans reconfiguration de la base. Les données prennent la forme de documents enregistrés eux-mêmes dans des collections, une collection contenant un nombre quelconque de documents. Les collections sont comparables aux tables, et les documents aux enregistrements des bases de données relationnelles. Contrairement aux bases de données relationnelles, les champs d'un enregistrement sont libres et peuvent être différents d'un enregistrement à un autre au sein d'une même collection. Le seul champ commun et obligatoire est le champ de clé principale ("id"). Par ailleurs, MongoDB ne permet ni les requêtes très complexes standardisées, ni les JOIN, mais permet de programmer des requêtes spécifiques en JavaScript.

4. PROCESSUS DE DEVELOPPEMENT

Notre choix s'est porté sur le processus 2TUP. En effet **2TUP** (2 Track unified process, prononcez "toutiyoupi") est un processus de développement logiciel qui implémente le Processus Unifié.

Le **2TUP** propose un cycle de développement en Y, qui dissocie les aspects techniques des aspects fonctionnels. Il commence par une étude préliminaire qui consiste essentiellement à identifier les acteurs qui vont interagir avec le système à construire, les messages qu'échangent les acteurs et le système, à produire le cahier des charges et à modéliser le contexte (le système est une boîte noire, les acteurs l'entourent et sont reliés à lui, sur l'axe qui lie un acteur au système on met les messages que les deux s'échangent avec le sens). Le processus s'articule ensuite autour de 3 phases essentielles.

5. MACHINE

- Nom : Dell Latitude 3150 ;
- Architecture : x64-bit ;
- Processeur : Intel Pentium(R) CPU-N3540 @2.16 GHZ x 4;
- RAM : 4.00 GO ;
- OS : Kali GNU /Linux Rolling
- Distribution : Debian
- GPU : Intel HG Graphics (1.5Gb).