CAHIER DES CHARGES

# enflatme

Auteurs :
Antoine AUGUSTI (ANT)
Étienne BATISE (BAT)
Thibaud DAUCE (TIB)

Date de publication :
27 mars 2014

# Table des matières

# Chapitre 1

# Fondements du projet

## 1.1 But du projet

### 1.1.1 Problème de l'utilisateur

Notre objectif est de proposer une application pour simplifier la vie des gens vivant dans une colocation. La vie entre colocataires implique des contraintes et des problèmes ; l'objectif de notre application est de les gérer plus facilement et plus efficacement.

### 1.1.2 Objectifs du projet

Nous voulons réaliser cette application pour qu'elle s'impose comme un outil incontournable de gestion pour toutes les personnes vivant dans une colocation. Nous voulons leur rendre la vie plus facile en faisant en sorte que les petits tracas de la vie entre colocataires soient fortement réduits.

## 1.2 Personnes et organismes impliqués dans les enjeux du projet

### 1.2.1 Maître d'ouvrage

Les maîtres d'ouvrage sont :
- Antoine Augusti, étudiant à l'Institut National des Sciences Appliquées de Rouen ;
- Étienne Batise, étudiant à l'Institut National des Sciences Appliquées de Rouen ;
- Thibaud Dauce, étudiant à l'Institut National des Sciences Appliquées de Rouen.

### 1.2.2 Acheteur

L'acheteur est la société Teamflat où travaillent les maîtres d'ouvrage cités précédemment.

## 1.3 Utilisateurs du produit

### 1.3.1 Utilisateurs directs du produit

Les utilisateurs de notre application seront les personnes vivant dans une colocation. La majorité des personnes vivant dans une colocation sont des étudiants puisque ceux-ci ont souvent peu de moyens financiers et choisissent de vivre en colocation pour réaliser des économies.

Notre cible est donc un public jeune, entre 18 et 25 ans, autant hommes que femmes. Certains jeunes actifs font également le choix de la colocation. Notre public utilise très fréquemment et maîtrise les dernières nouveautés technologiques et télécharge très régulièrement de nouvelles applications.

Concernant les centres d'intérêt de notre cible ils sont autour des sorties, de la musique, des soirées, des jeux-vidéos, des sports… Notre cible n'a pas beaucoup de temps à consacrer à la réalisation de tâches ménagères et n'apprécie pas perdre du temps avec ceci.

### 1.3.2 Priorité assignée aux utilisateurs

Les utilisateurs sont notre priorité ultime car nous réalisons une application pour répondre à leurs besoins. De plus, nous comptons sur nos utilisateurs pour réaliser la promotion de notre application et ainsi faire en sorte que celle-ci ait une croissance virale.

### 1.3.3 Implication nécessaire de la part des utilisateurs dans le projet

Aucune implication particulière des utilisateurs n'est nécessaire pour mener à bien le projet. Toutefois, étant donné qu'il est primordial de concevoir un produit qui convient à notre cible que sont les colocataires, il est important de travailler de manière proche avec certains d'entre eux afin d'avoir leur avis sur l'avancement et l'orientation du projet.

### 1.3.4 Utilisateurs concernés par les opérations de maintenance du produit

Aucun utilisateur ne devra être concerné par des opérations de maintenance car les données devront être stockée sur un des serveurs de la société Teamflat.

# Chapitre 2

# Contraintes sur le projet

## 2.1 Contraintes non négociables

### 2.1.1 Contraintes sur la conception de la solution

**Téléphone mobile Android**

L'application doit être utilisable sur un smartphone Android équipé d'une version du système d'exploitation supérieure à la version 4 afin de cibler le maximum d'utilisateurs.

**Code source**

Le code source doit répondre à certaines exigences afin d'être libéré sous licence libre dans un second temps :

Le code source doit être lisible. Les développeurs ne doivent pas utiliser d'abréviations pour les noms des variables, le camelCase doit être privilégié pour le nom de variables et des fonctions. Les développeurs doivent dans la mesure du possible utiliser des opérateurs littéraux (comme 'AND' ou 'OR') à la place d'opérateurs moins lisibles (comme '&&' ou '||'). Les développeurs ne doivent en aucun cas utiliser des conditions ternaires. Pour finir, le langage de programmation choisi doit permettre une lecture facile.

Le code source doit être commenté de manière efficace. Les commentaires ne doivent pas surcharger le code source mais doivent être présents afin d'expliquer les parties complexes des fonctions. La décomposition d'un problème doit toujours être privilégiée à la surcharge d'une fonction. Les fonctions et les parties de code dites 'simples' et compréhensibles par tous ne doivent pas être commentées. Un code lisible doit toujours être privilégié par rapport à des commentaires.

Enfin, les fonctions du code source doivent être documentées afin de pouvoir générer une documentation de manière automatique (avec des outils tels que Doxygen ou JavaDoc par exemple). Chaque fonction doit présenter une description succincte, le nom des paramètres en entrée et le nom des paramètres en sortie. En cas de complexité plus importante ou d'une incompréhension possible, la documentation doit afficher une description plus complète et/ou une description de chaque attribut en entrée ou en sortie.

### 2.1.2 Environnement de fonctionnement du système actuel

Le but du projet est de créer une application à partir de zéro. Aucun système actuel n'est en place.

### 2.1.3 Applications partenaires

Nous n'envisageons aucun lien avec des applications tierces. En particulier avec les réseaux sociaux, notre application doit être complètement indépendante.

### 2.1.4 COTS : progiciels ou composants commerciaux

N/A

### 2.1.5 Lieux de fonctionnement prévus

Le lieu principal de fonctionnement prévu est la collocation. Les personnes seront dans un environnement familier, sans bruit et tranquille. Ils auront à disposition un ordinateur fixe ou portable ainsi qu'un smartphone.

Malgré la tranquillitéé du lieu, l'application doit être rapidement exécutable afin d'effectuer les mises à jour de ses activités rapidement.

### 2.1.6 De combien de temps les développeurs disposent-ils pour le projet

Le projet doit être opérationnel fin août afin de permettre aux nouveaux étudiants de s'inscrire dès le début de leur collocation. Les développeurs disposent donc de 4 mois à compter d'aujourd'hui afin de livrer l'application début août et ainsi nous permettre de mettre en place une communication et une campagne marketing pour le lancement du produit.

### 2.1.7 Quel est le budget affecté au projet ?

N/A

## 2.2 Glossaire et conventions de dénomination

**Collocation** La collocation désigne l'habitation des collocataires. Celle-ci peut être un appartement, une maison ou tout autre logement.

**Smartphone** Un smartphone est un téléphone mobile équipé d'un système d'exploitation permettant d'accèder à Internet et d'effectuer des tâches plus complexes que de téléphoner ou envoyer des SMS. Nous entendons par smartphone, tout téléphone équipé du système d'exploitation Android, iOS, Windows Phone, Sailfish OS ou Firefox OS.

**License libre** Les licenses libres permettent de mettre à disposition de la communauté le code source d'un logiciel ou d'un programme afin qu'il puisse être réutilisé, modifié et amélioré. Dans ce document nous parlons d'une license de type copyleft qui aurait pour objectif de bénéficier des améliorations qui pourraient être apportées à notre logiciel par la communauté.

**Camel case** Le camel case est une norme d'écriture facilitant la lecture. Un mot en camel case est écrit en miniscule. Dans le cas d'un groupe de mot attachés, la première lettre est une minuscule, chaque nouveau mot commence par une majsucule.
Exemple : variable trop cool⇒variableTropCool

**Conditions ternaires** Certains langages proposent une écriture simplifiée pour les conditions de type 'if / else' appelée condition ternaire et tenant sur une seule ligne.

**Application native** Une application native est une application installable par l'utilisateur sur son appareil. Elle est développée dans un langage particulier, souvent propre à la plate-forme ou possèdant des bibliothèques propres à la plate-forme. Elle présente l'avantage d'être rapide et permet l'utilisation de manière très facile des composants particuliers des appareils (géolocalisation, vibreur...).

**Application web** Une application web est une application manipulable à travers un navigateur web. Elle présente l'avantage d'être utilisable sur toutes les plate-formes (Android, iOS, Windows, Linux...) sans installation de la part de l'utilisateur. La démocratisation du HTML5 permet aux applications web d'accèder à des composants particuliers des appareils (géolocalisation, vibreur...)

## 2.3 Faits et hypothèses utiles

### 2.3.1 Facteurs influençant le produit, mais qui ne sont pas des contraintes imposées sur les exigences

Nous visons des utilisateurs de smartphone. Notre cahier des charges restreint cette catégorie aux utilisateurs d'Android permettant la création d'une application native pour ce système d'exploitation. Une application web serait compatible avec tous les smartphones, indépendamment de leur système d'exploitation, et nous permettrait donc d'élargir notre base d'utilisateurs.

### 2.3.2 Hypothèses que l'équipe fait sur le projet

L'application ne sera pas utilisée de manière continue pendant une longue période. Les utilisateurs utiliseront l'application de manière ponctuelle au cours de la journée.

# Chapitre 3

# Exigences fonctionnelles

## 3.1 Portée du travail

### 3.1.1 La situation actuelle

Aujourd'hui, il n'existe aucun type d'application correspondant à la description de notre produit. Cependant, il existe une application web publique développé par Merlin Nimier-David qui permet, de manières simpliste, à différent membres d'un groupe de gérer leurs dettes. Cette application est accessible à l'adresse `merlin.nimierdavid.fr/debts/`.

Par ailleurs, il existe aussi de nombreuse applications web / mobile permettant de créer ses listes de courses. L'une des plus connues sur le marché est l'application "PlanCourses" développée par la société Agilys qui permet notamment de choisir choisir ses articles en fonction des marques de la grande distribution comme Carrefour, Auchan, Leclerc…

### 3.1.2 Contexte du travail

### 3.1.3 Division du travail en événement métier

**Permettre le changement de colocation**

Il est possible dans la vie d'un utilisateur de déménager, il est donc nécessaire de lui permettre de changer, ou quitter, sa colocation, pour s'il le souhaite en rejoindre une autre.

**Être agréable à utiliser**

Pour que notre produit est un réel succès, il est nécessaire que l'utilisateur n'apprécie pas le produit uniquement pour le service qu'il propose. Il faut qu'il apprécie et aime l'utiliser ; il s'agit donc de s'inspirer des nombreux services populaires actuellement comme les réseaux sociaux.

**Afficher clairement la gestion des tâches**

Notre produit est destiné à rendre plus facile la gestion d'une colocation, et servir de support pour prévenir les problèmes. Ainsi, permettre aux utilisateurs d'interpréter facilement et de manière compréhensible par tous leurs propres données.

## 3.2 Portée du Produit

### 3.2.1 Modèle d'usage

### 3.2.2 Description des cas d'utilisation

## 3.3 Portée fonctionnelles

# Chapitre 4

# Exigences non fonctionnelles

## 4.1 Ergonomie et convivialité du produit

### 4.1.1 L'interface

L'application utilisera un design dit *flat*, c'est-à-dire à posséder une interface avec des formes simples et nettes (pas de bords arrondis), sans ombres et sans dégradés. L'application devra posséder une mascotte, qui reste encore à définir, qui aura vocation d'accompagner l'utilisateur dans la prise en main de l'application. Celle-ci sera présente et possédera une personnalité bien définie.

Les interactions avec l'interface doivent être rapides : l'interface doit être réactive et le nombre de gestes pour réaliser des actions doit être minimal.

### 4.1.2 Le style du produit

Le produit doit correspondre aux interfaces que les jeunes ont l'habitude d'utiliser en ce moment, on pense particulièrement aux réseaux sociaux. L'application devra être colorée et être perçue comme moderne par nos utilisateurs.

## 4.2 Facilité d'utilisation et facteurs humains

### 4.2.1 Facilité d'utilisation

La prise en main de l'application doit se faire rapidement. C'est pourquoi on utilisera dès que possible des petits didacticiels pour montrer comment l'interface doit être utilisée. Pour ceci on pourra s'aider de notre mascotte où d'écrans de présentation. L'utilisateur doit pouvoir mémoriser facilement l'interface, c'est pourquoi on essaiera d'uniformiser celle-ci au maximum entre les différentes fonctionnalités.

### 4.2.2 Personnalisation et internationalisation

Notre application sera utilisable en Français et utilisera l'Euro comme monnaie. Il n'est pas prévu de proposer d'autres langues et d'autres monnaies dans un premier temps.

### 4.2.3 Facilité d'apprentissage

L'apprentissage de l'utilisation de l'application doit être immédiat pour éviter de perdre des utilisateurs rebutés par une interface trop complexe à comprendre.

### 4.2.4 Facilité de compréhension et politesse

L'application utilisera beaucoup d'icônes et aura un ton proche de l'utilisateur qui correspond à la culture des jeunes. Les messages pourront être ironiques et taquins dans des situations non critiques.

### 4.2.5 Exigences d'accessibilité

Aucune exigence particulière concernant les handicaps. Il est nécessaire de s'assurer que les éléments de l'interface ne soient pas trop petits pour pouvoir être touchés sur un mobile sans risque d'erreur.

## 4.3 Fonctionnement du produit

### 4.3.1 Rapidité d'exécution et temps de latence

La transition entre les écrans devra être la plus rapide possible. Lorsque l'utilisateur n'est pas connecté au réseau Internet on mettra en cache les données qu'il souhaite envoyer et consulter. Celles-ci seront alors récupérées la prochaine fois qu'il sera en ligne.

### 4.3.2 Exigences critiques de sécurité

L'utilisation du produit n'entraînera pas un risque de sécurité particulier.

### 4.3.3 Précision et exactitude

Les heures seront données au maximum avec une exactitude ne dépassant pas la minute. Les unités monétaires utiliseront une précision de l'ordre du centième d'euro.

### 4.3.4 Fiabilité et disponibilité

Les serveurs hébergeant l'application du projet devront avoir une disponibilité de 99,5 % (1,83 jours d'indisponibilité par an).

### 4.3.5 Robustesse ou tolérance à un emploi erroné

L'application devra fonctionnement de manière correcte si celle-ci ne possède pas d'accès à Internet. Certaines fonctionnalités pourront être limitées pour ne pas avoir un comportement inhabituel ou des données incohérentes.

### 4.3.6 Capacité de stockage et montée en charge

Le produit doit pouvoir servir simultanément à 500 utilisateurs entre 16h et 23h (heures de pointe). La charge maximale aux autres moments sera de 200 utilisateurs.

### 4.3.7 Adaptation du produit à une augmentation de volume à traiter

Le produit doit pouvoir gérer le passage d'une base vide à une base de 100 000 utilisateurs en l'espace d'un mois.

### 4.3.8 Longévité

Le produit n'a pas de durée de vie définie. Il doit s'adapter aux évolutions pour pouvoir durer le plus longtemps possible. On estime la durée de vie d'utilisation de l'application par un utilisateur à 2-3 ans en moyenne, soit le temps moyen où celui-ci est en colocation. Le produit n'a pas de durée de vie définie. Il doit s'adapter aux évolutions pour pouvoir durer le plus longtemps possible. On estime la durée de vie d'utilisation de l'application par un utilisateur à 2-3 ans en moyenne, soit le temps moyen où celui-ci est en colocation.