



Université de la Manouba
École Nationale des Sciences de l'Informatique



RAPPORT DU PROJET DE CONCEPTION ET DE DÉVELOPPEMENT

Sujet : Récupération et analyse statistique des
données d'une campagne publicitaire sur les
sites des réseaux sociaux et les forums de
discussion

Auteurs :

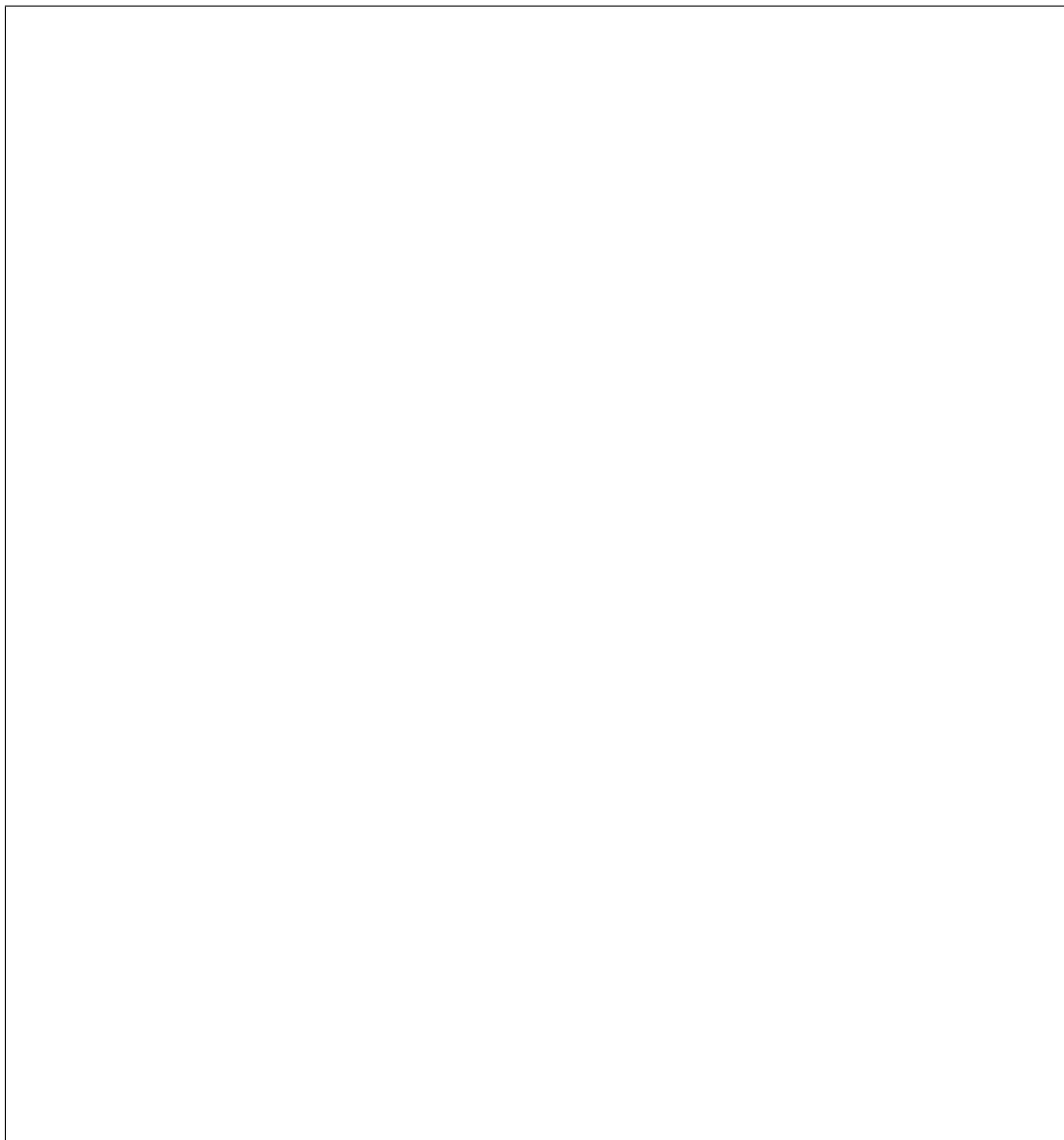
M. HBAIEB BECEM M. MESSAOUDI MEHDI

Encadrant :

Dr. BEN SAÏD WISSEM

Année Universitaire :2018 /2019

Appréciations et signature de l'encadrant

A large, empty rectangular box with a thin black border, intended for the appreciations and signature of the supervisor.

Remerciements

Nous tenons, avant de présenter notre travail, à exprimer notre gratitude envers les personnes qui nous ont, de près ou de loin, apporté leur soutien.

En premier lieu, Nous tenons à remercier Mr. Wissem BEN SAÏD, notre encadrant de ce projet de conception et de développement, pour sa patience, pour le temps qu'il nous a consacré tout au long de cette période, pour nous avoir dirigés et suivis durant la réalisation de ce travail, sa correction pour notre démarche de projet et ses précieux conseils.

Nous saisissons cette occasion pour adresser aussi notre gratitude aux membres du jury pour nous avoir honorés en acceptant d'évaluer ce travail et nous espérons qu'ils y trouvent les critères de la bonne rédaction et de motivation auxquels ils aspirent.

Ainsi que tous les enseignants de l'ENSI pour leur soutien et pour le savoir qu'ils ont partagé avec nous.

Table des matières

Introduction	1
1 Étude préalable	3
1.1 Étude de l'existant	3
1.1.1 Présentation des solutions existantes	3
1.1.2 Critique de l'existant	5
1.1.3 Méthodologie de travail	7
2 Analyse et spécification des besoins	8
2.1 Spécification informelle des besoins	8
2.1.1 Identification des acteurs	8
2.1.2 Analyse des besoins	9
2.2 Spécification semi-formelle des besoins	11
2.2.1 Diagrammes de cas d'utilisation	11
2.2.2 Description de quelques scénario	15
3 Conception	19
3.1 Conception architecturale de l'application	19
3.1.1 Architecture physique	19
3.1.2 Architecture logique	21
3.2 Conception détaillée	25
3.2.1 Conception de la base de données	25
3.2.2 Diagrammes de séquences d'objets	28
4 Réalisation	32
4.1 Environnement de travail	32
4.1.1 Environnement matériel	32
4.1.2 Environnement logiciel	33
4.1.3 Choix technologiques	33
4.2 Explication du travail réalisé	34
4.2.1 Partie Récupération	34
4.2.2 Partie Analyse	39

Table des matières	iv
---------------------------	-----------

4.2.3 Aperçu sur le travail réalisé	42
4.3 Conclusion	46
Bibliographie	48
Netograph	49

Table des figures

1.1	Interface Client de Import.io	4
1.2	Interface Client de Parsehub	5
2.1	Diagramme de cas d'utilisation pour le visiteur	12
2.2	Diagramme cas d'utilisation pour le client	13
2.3	Diagramme cas d'utilisation pour l'administrateur	14
2.4	Diagramme de séquence système d'authentification	15
2.5	Diagramme de séquence système du cas d'utilisation "Effectuer un analyse"	16
2.6	Diagramme de séquence système du scénario "Gestion des utilisateurs"	17
3.1	Diagramme de déploiement	21
3.2	Patron de conception MVT	22
3.3	Fonctionnement de l'architecture choisie	24
3.4	Modèle Entité-Association	25
3.5	Diagramme de séquences objets «S'inscrire»	29
3.6	Diagramme de séquences objets «Effectuer une analyse»	30
4.1	L'authentification pour l'API Twitter	36
4.2	L'utilisation la fonction urlopen du module urllib.request	36
4.3	Version Sans Cursor	37
4.4	Version Avec Cursor	37
4.5	Exemple d'instruction pour récupérer les données avec API Twitter	38
4.6	Exemple d'un document HTML pou analyser son contenu	38
4.7	Création de l'objet soupe	39
4.8	Exemple d'utilisation de la fonction find_all() de BeautifulSoup	39
4.9	Les données d'apprentissage et de test	41
4.10	Résultat d'apprentissage	42
4.11	Interface d'accueil	43
4.12	Interface d'inscription	43
4.13	Interface d'authentification	44
4.14	Interface pour effectuer une requête	45
4.15	Graphe montrant les réactions du consommateurs en%	45

4.16 Un échantillon des commentaires récupérés	46
--	----

Liste des tableaux

3.1	Diagramme Entité-Association	27
3.2	Table des associations	27

Liste des sigles et acronymes

API	<i>Application Program Interface</i>
IP	<i>Internet Protocol</i>
UML	<i>Unified Modeling Language</i>
URL	<i>Uniform Resource Locator</i>
XML	<i>Extensible Markup Language</i>
HTML	<i>HyperText Markup Language</i>
HTTP	<i>HyperText Transfer Protocol</i>
SGBD	<i>Système de Gestion de Base de Données</i>
SGBDR	<i>Système de Gestion de Base de Données Relationnelle</i>
SQL	<i>Structured Query Language</i>
ORM	<i>Object-Relational Mapping</i>
RAM	<i>Random Access Memory</i>
CPU	<i>Central Processing Unit</i>
DoS	<i>Denial of Service</i>
NLTK	<i>Natural Language Toolkit</i>
MVT	<i>Model-View-Template</i>
MVC	<i>Model-View-Controller</i>

Introduction générale

Le client est-il un moyen ou une finalité ? C'est la question fondamentale qui se pose dans le domaine commercial et économique. En fait, il en est les deux à la fois étant donné qu'il est le principal facteur qui va déterminer l'activité commerciale, et pour y arriver il faut qu'il soit considéré comme une fin pour agir sur lui en tant que consommateur indispensable pour la garantie des activités commerciales et économiques. C'est pourquoi de nos jours le marketing occupe une place importante dans l'économie de tous les pays du monde. Ce domaine ne cesse d'évoluer si bien que chaque année on voit naître une nouvelle conception plus efficace que les précédentes.

Le partage d'avis clients a fait une grande révolution innovatrice dans le domaine du marketing. Presque 90 % des consommateurs tiennent à consulter les avis clients sur Internet avant de faire leurs achats. Dans cette procédure de recherche, le client accorde une importance capitale à l'avis client, quel que soit le domaine d'achat. Et quoique les auteurs de ces avis leur soient inconnus, les consommateurs sont 68% à se fier à eux. Soit une confiance deux fois plus importante que celle qu'ils accordent aux publicités provenant des médias traditionnels. . .

Cette expression libre et publique des consommateurs change la conception. Un constat simple s'impose : un directeur de marketing ne peut en aucun cas se passer des avis clients. Les volumes des feedback client auxquels nous avons accès ne cessent d'augmenter. La problématique est de savoir comment les exploiter d'une manière efficace. Pour la plupart des entreprises, le plus grand challenge consiste à trouver un outil susceptible de gérer et d'analyser de nombreux types de données provenant d'une variété de sources en évolution constante. Mais avant de pouvoir analyser ou utiliser ces données, il faut d'abord les extraire.

Étant donné qu'une telle problématique a un intérêt majeur et primordial, nous avons pris l'initiative de traiter cette éventualité à travers notre projet de conception et de développement. Ce projet a pour finalité de réaliser une application web qui permette de récupérer les avis clients sur n'importe quel produit donné depuis différents réseaux sociaux et sites de vente en ligne et de convertir le langage humain en des données numériques permettant de déterminer un avis général sur le produit.

L'objectif de ce rapport est de présenter la démarche suivie pour la mise en place de notre application. Celle-ci s'articule autour de quatre chapitres. Le premier chapitre est consacré à une étude préalable où nous définissons quelques notions concernant la procédure de recrutement, nous effectuerons, par la suite, une étude de l'existant et nous montrerons les carences des solutions existantes ainsi qu'une présentation de la solution suggérée. Le second chapitre présentera une analyse et une spécification des besoins fonctionnels et non fonctionnels auxquels doit répondre notre application. Le troisième chapitre se focalisera sur l'architecture du système ainsi que sa conception détaillée. Le quatrième chapitre comporte une description des outils et des technologies employés dans le développement du projet et une illustration concise de la réalisation du système. Finalement nous clôturons ce rapport par une conclusion générale qui synthétise le travail élaboré et qui ouvre des perspectives des travaux futurs.

Chapitre 1

Étude préalable

Introduction

L'objectif de ce chapitre est de situer le projet dans son contexte général, à savoir la problématique qui a inspiré la création de notre application, la description du projet et les objectifs à atteindre. Dans ce qui suit, nous élaborerons une étude approfondie et bien détaillée sur des solutions existantes en mettant en relief leurs insuffisances. Pour finir, nous présenterons la solution adoptée pour résoudre ce problème ainsi que le travail demandé.

1.1 Étude de l'existant

Dans cette partie, nous présenterons une étude des solutions existantes, en nous concentrant sur les apports et les limites de chacune d'elles. Cette étude va nous permettre de saisir les besoins auxquels doit répondre notre application.

1.1.1 Présentation des solutions existantes

1.1.1.1 Import.io

Import.io est une plate-forme permettant une conversion d'informations semi-structurées contenues dans des pages Web en données structurées, qu'il s'agisse de la prise de décisions marketing ou de l'intégration à des applications. Elle propose la récupération d'informations en temps réel via des API de diffusion en continu, et l'intégration à différents langages de programmation et outils d'analyse de données.

Cette application est conçue pour permettre à l'utilisateur de :

- **Extraire automatiquement** : Extraire des données de pages Web dans un jeu de données bien structuré.

- **Générer des extracteurs** : Générer des extracteurs.
- **Authentifier** : Authentifier les données derrière un identifiant / mot de passe.
- **Planifier** : Planifier les extracteurs de programme à exécuter exactement lorsqu'on en a besoin.

#	Profile 3	Profile Name	Link Normal	Size Base Link	Size Mini Link 2	Text	S
1		MD007	1.0 out of 5 stars	Beware this may be a used...	Colour: SilverSize: 256 GB	I would put a zero star rating if allowed + 1 Items	54 people found this he
2		TJ	4.0 out of 5 stars	A solid phone, but an eyew...	Colour: Space GreySize: 64 ...	It's a truly great phone, at a grossly in + 1 Items	22 people found this he
3		George	5.0 out of 5 stars	It's a legit unopened I Phon...	Colour: Space GreySize: 64 ...	When I ordered this I was a bit skepti + 1 Items	17 people found this he
4		Karima Mohamed	5.0 out of 5 stars	Great brand new iPhone !	Colour: Space GreySize: 64 ...	Really good iPhone ! Arrived way earli + 1 Items	18 people found this he
5		Ritlady	5.0 out of 5 stars	Genuine iPhone x	Colour: Space GreySize: 64 ...	I received a genuine iPhone X, I was ir + 1 Items	12 people found this he
6		Rustam	5.0 out of 5 stars	Excellent cu		supplier is amazing. Received this pho + 1 Items	11 people found this he
7		Wasim	5.0 out of 5 stars	Great phone		Purchased for my daughter and she al + 2 Items	6 people found this he
8		Helicoptrema	5.0 out of 5 stars	iPhoneX, at		It's an iPhoneX, at good price - love it + 1 Items	6 people found this he
9		Matt	5.0 out of 5 stars	Five Stars		absolute bargain for the iphone 10 als + 1 Items	5 people found this he
10		kelvin	2.0 out of 5 stars	i just cost too much	Colour: SilverSize: 64 GB	review after 2 month the iphone xs ov + 4 Items	

FIGURE 1.1 – Interface Client de Import.io

1.1.1.2 ParseHub

ParseHub ParseHub est une nouvelle extension de navigateur Web utilisable pour transformer tout site Web dynamique et mal structuré en une API, sans écrire de code. ParseHub est un outil de scraping conçu pour fonctionner sur des sites Web avec JavaScript et Ajax.. L'outil ParseHub assure l'identification des relations entre les éléments, il permet aussi d'extraire toutes les données et les fournir dans un tableur ou une API facilement accessible. Les scrapers et les données sont conservés dans le cloud. Vous n'avez qu'à télécharger l'extension de navigateur ParseHub et de commencer à extraire les données dont vous avez besoin.

Cette application est conçue afin de permettre à l'utilisateur de :

- **Récupérer les données de manière instantanée** : ParseHub montre instantanément un échantillon de données.
- **Profiter d'une planification et hébergement Cloud** : Les données sont stockées et accessibles à tout moment.
- **Profiter d'une Rotation IP automatique** : Routage de toutes les demandes via une liste d'adresses IP disponibles pour garantir et préserver la confidentialité et l'anonymat du client.

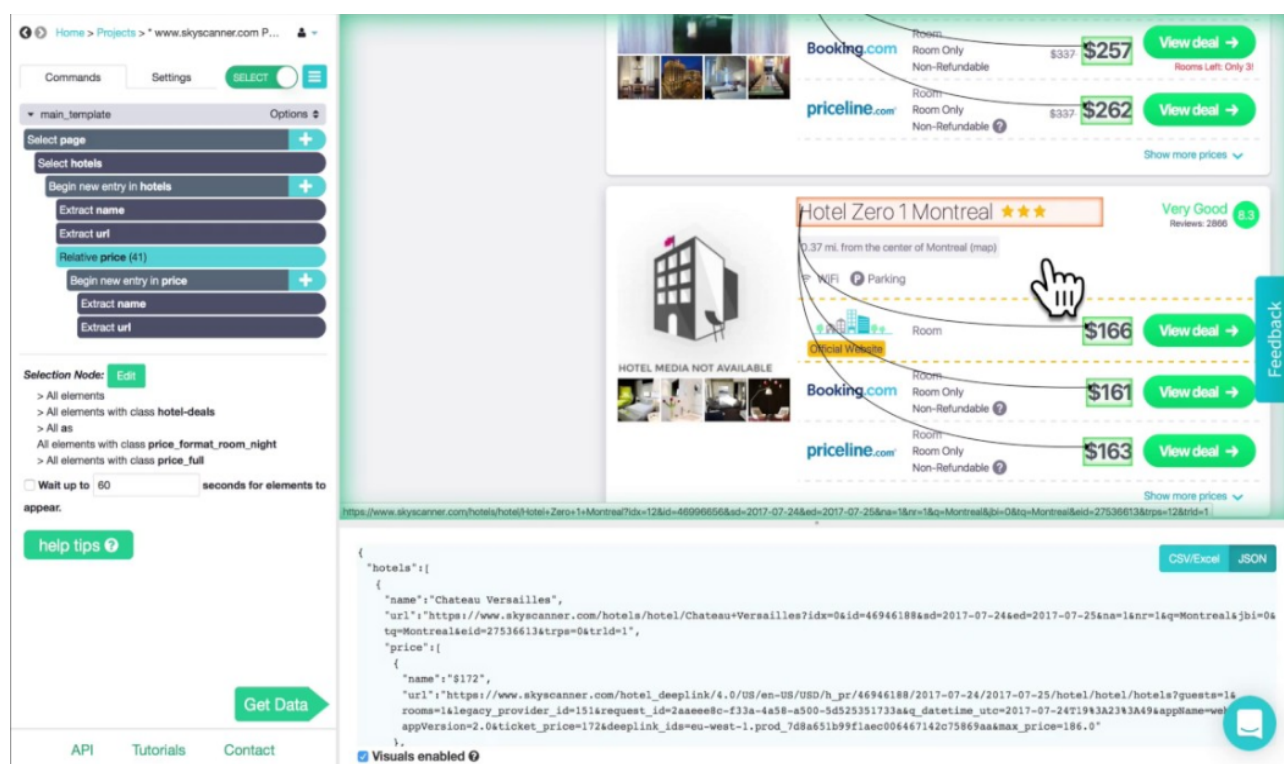


FIGURE 1.2 – Interface Client de Parsehub

1.1.2 Critique de l'existant

Afin de faire une étude minutieuse des solutions présentées, nous évaluerons dans cette partie les critères pris en considération ainsi qu'une étude comparative des outils introduits préalablement.

1.1.2.1 Critères de comparaison

- **Facilité d'utilisation et portabilité** : Est-ce que l'installation et la configuration du logiciel dans un nouvel environnement demande un effort considérable ?
- **Gratuité** : Est-ce que l'application permet de créer un compte gratuitement ? Contient-elle des services payants ?
- **Automatisation** : Est-ce que la collection des données se fait de manière automatisée ou sommes-nous contraints de regrouper les données manuellement page par page ?
- **Analyse de sentiments** : Est-ce que cette application propose un service permettant de convertir les feedback des clients, saisis en langage humain, en données numériques pouvant être nécessaires pour dégager un avis général du public ?

1.1.2.2 Problèmes dégagés

Plusieurs solutions technologiques sont présentes afin de collecter les feedback des clients. Mais les solutions étudiées précédemment présentent quelques insuffisances auxquelles nous allons chercher à remédier.

En effet, la plupart des applications proposées sont payantes et exigent une connaissance préalable des sites qu'on cherche à parser. Il sera obligatoire de préciser que la plupart des applications nécessitent bien plus qu'un simple navigateur web pour pouvoir utiliser ses fonctionnalités puisqu'elles demandent un téléchargement de l'application, son installation et sa configuration.

Toutefois, on a constaté que la récolte des informations n'est pas vraiment automatisée. Pour récupérer les avis clients, l'utilisateur est contraint d'aller manuellement sur le site en question et de sélectionner les données qu'il veut collecter, ce qui peut s'avérer long à faire, surtout si les avis clients sont éparpillés sur des centaines de pages.

Finalement, aucune des applications étudiées ne propose une solution permettant la manipulation des données collectées et le dégagement des informations sur ces données de manière automatique.

1.1.2.3 Solution proposée

L'étude de l'existant nous a permis de dégager certaines anomalies que nous avons abordées dans la partie précédente. Pour remédier à ces anomalies, nous proposons de développer une application Web, étant un apport de la solution proposée mais qui essaie

de résoudre ces problèmes en favorisant l'automatisation de la récolte des données sans que le client ait forcément une connaissance préalable des sites à parser, ainsi que l'introduction d'une nouvelle fonctionnalité qui est la classification de texte, et en particulier l'analyse de sentiments, permettant un traitement automatique du langage humain, afin d'identifier rapidement les sentiments clés issus de verbatim d'internautes.

1.1.3 Méthodologie de travail

Développer une application moderne ne peut en aucun cas être le résultat d'un effort individuel. La conception, le codage, le test des applications sont des tâches difficiles et délicates qui nécessitent une équipe inter-fonctionnelle. Afin de réduire la complexité de notre projet, et vu que notre équipe de travail est constituée de deux personnes seulement, nous nous sommes contentés d'opter pour la méthode en cascade.

Conclusion

Dans ce chapitre, nous avons présenté le contexte général du projet suivi d'une étude approfondie de l'existant et de critique des solutions présentes. Cela nous a permis de comprendre les besoins et d'envisager la solution la plus adéquate à notre application. Le chapitre suivant sera consacré à la présentation des besoins fonctionnels et non fonctionnels. Nous terminerons par une spécification de ces besoins en nous basant sur les diagrammes d'UML.

Chapitre 2

Analyse et spécification des besoins

Introduction

Afin de respecter les critères spécifiés dans le cahier des charges, de garantir le succès ainsi que l'efficacité du projet, une bonne analyse du système étudié est nécessaire. Dans ce qui suit, nous compterons parvenir à une vision plus claire des divers besoins de notre projet. Au cours de ce chapitre, nous allons dégager en détail les besoins fonctionnels et non fonctionnels de notre application, les diagrammes des cas d'utilisation des différents acteurs et les différents scénarios expliquant ces cas.

2.1 Spécification informelle des besoins

La spécification informelle est l'ensemble des exigences fonctionnelles et non fonctionnelles exprimées en langage naturel. Cette étape est très importante puisqu'elle nous permettra de réduire la complexité de la modélisation des besoins ultérieurement.

2.1.1 Identification des acteurs

L'application comprend trois acteurs principaux qui interagissent avec le système pour avoir accès aux diverses fonctionnalités du site à travers ses interfaces. Dans ce qui suit, nous allons présenter ces différents acteurs :

- Visiteur : C'est un internaute non inscrit sur notre site.
- Client : C'est un internaute déjà inscrit sur le site : Il a accès aux différentes fonctionnalités de l'application.
- L'administrateur : il peut être à la fois considéré comme un internaute et le responsable

de la gestion des comptes clients et du contrôle des activités sur le site.

2.1.2 Analyse des besoins

Les besoins sont divisés en deux catégories, les besoins fonctionnels et non fonctionnels.

2.1.2.1 Besoins fonctionnels

Un besoin fonctionnel est relatif aux fonctions, c'est-à-dire les actions et les réactions faites par le système suite à une sollicitation d'un acteur. Vu la nature de l'application, on peut distinguer les besoins par acteurs :

- **Visiteur** : Le système doit permettre au visiteur de :
 - Créer un compte :
 1. Donner un nom d'utilisateur.
 2. Donner un email valide.
 3. Choisir un mot de passe.
 4. Confirmer le mot de passe.
 - Visualiser les feedback des clients inscrits sur la performance du site.
 - Découvrir les fonctionnalités du site.
- **Client** : Le système doit permettre au client de :
 - Avoir accès aux fonctionnalités du site.
 - S'identifier :
 1. Donner son email.
 2. Donner son mot de passe.
 - Effectuer une analyse :
 1. Choisir les mots clés relatifs au produits.
 2. Choisir l'URL pour faire l'analyse.
 3. Choisir le nombres d'avis à analyser.

-
- Visualiser les résultats de l'analyse :
 1. Visualiser les avis parsés.
 2. Visualiser le graphe représentant la satisfaction des consommateurs du produit en %.
 - Gérer son profil :
 1. Changer la photo de profil.
 2. Changer l'email.
 3. Changer nom utilisateur.
 - Ajouter un commentaire pour évaluer la performance du site.
 - Visualiser son historique.
 - Demander à l'administrateur de parser un nouveau site :
 1. Choisir les mots clés.
 2. Choisir l'URL désirée.
 - **Administrateur** : Le système doit permettre à l'administrateur de :
 - S'authentifier :
 1. Donner son Id.
 2. Donner le mot de passe.
 - Gérer les comptes d'utilisateurs :
 1. Supprimer un utilisateur.
 2. Créer un utilisateur.
 - Visualiser et gérer les recherches faites par chaque utilisateur.
 - Visualiser et traiter les demandes des utilisateurs.

2.1.2.2 Besoins non fonctionnels

La solution proposée par notre application doit tenir compte des critères suivants :

- **Simplicité** : l'application doit être facile à utiliser par l'utilisateur.
- **Sécurité** : L'application doit garantir un minimum de sécurité et de confidentialité.
- **Efficacité** : L'application doit permettre à l'utilisateur de prévoir les taux de satisfaction des consommateurs sur le produit avec précision.
- **Rapidité** : L'application doit être rapide pendant le chargement et le traitement des données, afin de gagner du temps.

2.2 Spécification semi-formelle des besoins

Les techniques de spécification semi-formelle des besoins sont les plus couramment utilisées dans les méthodes modernes d'analyse et de conception. Elles favorisent la communication entre développeurs et utilisateurs en utilisant des représentations graphiques des données et des traitements. Ces représentations sont particulièrement pratiques pour fournir une vue d'ensemble d'un système.

Dans ce qui suit, nous aurons recours aux diagrammes UML et plus spécifiquement les **diagrammes de cas d'utilisation** et les **diagrammes de séquence**.

2.2.1 Diagrammes de cas d'utilisation

Pour illustrer les fonctionnalités de notre système, nous utiliserons les diagrammes de cas d'utilisation, servant à donner une vision globale du comportement de l'application.

2.2.1.1 Diagramme de cas d'utilisation relatif au visiteur

La figure ci-dessous représente le diagramme de cas d'utilisation relatif au visiteur :

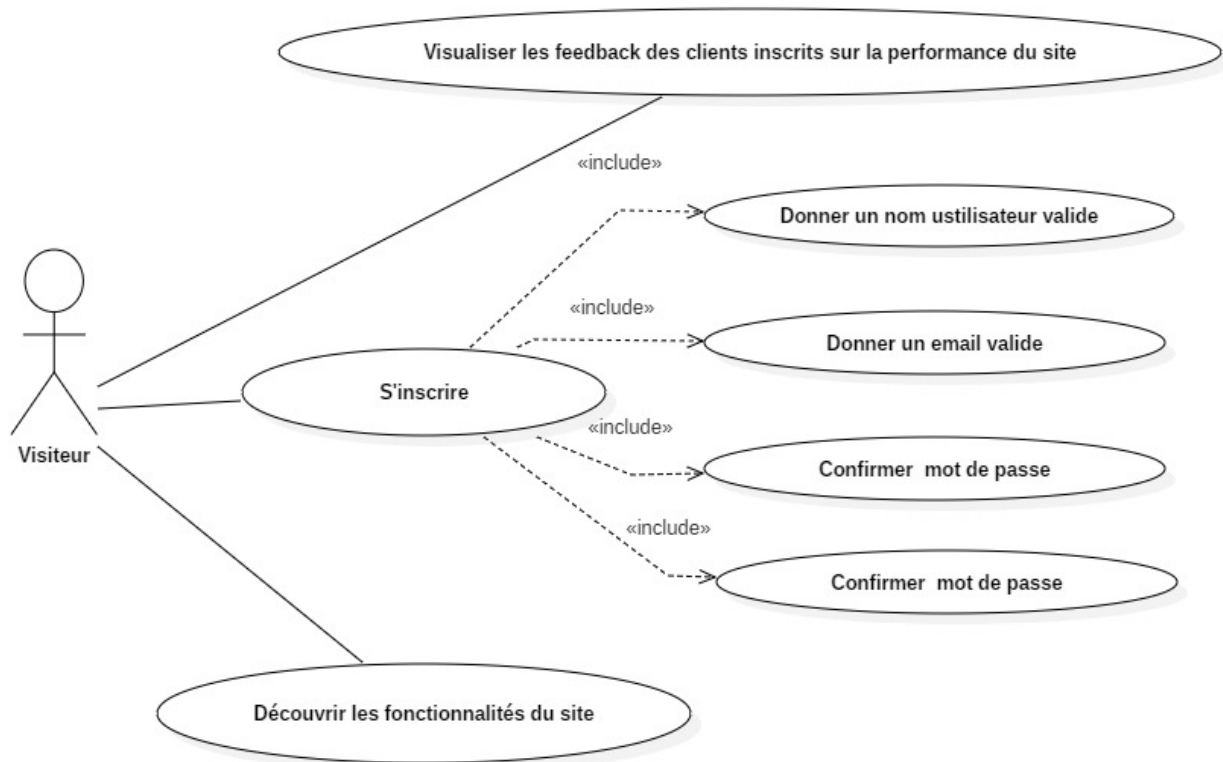


FIGURE 2.1 – Diagramme de cas d'utilisation pour le visiteur

Ce diagramme de cas d'utilisation présente les différentes fonctionnalités dont un visiteur peut profiter. Il peut s'inscrire facilement en choisissant un nom d'utilisateur et en donnant un email et un mot de passe. Il peut aussi découvrir les différents services du site et visualiser les feedbacks d'autres utilisateurs pour avoir une idée générale sur l'application.

2.2.1.2 Diagramme de cas d'utilisation relatif au client

La figure ci-dessous représente le diagramme de cas d'utilisation relatif au client :

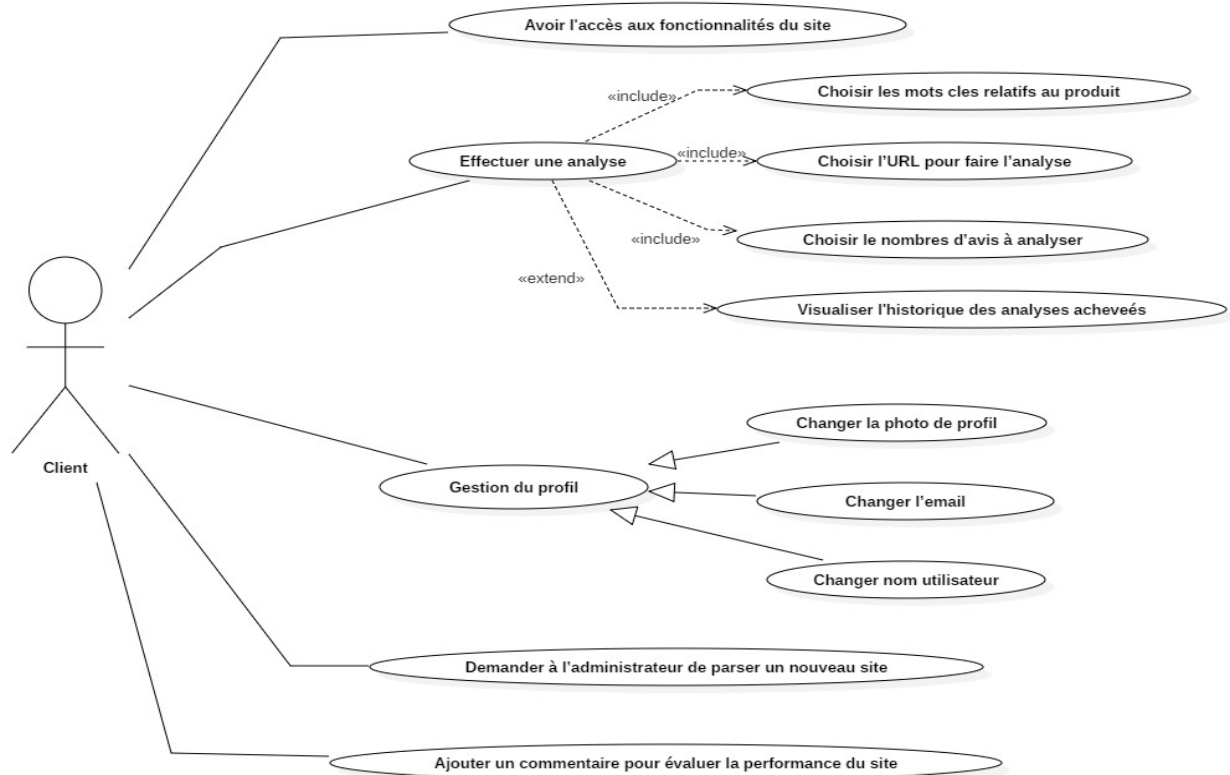


FIGURE 2.2 – Diagramme cas d'utilisation pour le client

Ce diagramme de cas d'utilisation présente les différentes tâches qu'un client peut effectuer. Le client doit d'abord s'authentifier pour pouvoir avoir accès aux services du site. Il peut consulter son historique et visualiser tous les résultats correspondants aux analyses achevées. En plus de la liste de sites proposée par l'application, le client peut soumettre une demande d'ajouter de nouveaux sites à parser en contactant l'administration. Aussi, il peut mettre à jour son profil en modifiant sa photo, son email ou son nom d'utilisateur. Il peut finalement ajouter des commentaires afin d'évaluer la performance de l'application.

2.2.1.3 Diagramme de cas d'utilisation relatif à l'administrateur

La figure ci-dessous représente le diagramme de cas d'utilisation relatif à l'administrateur de notre application :

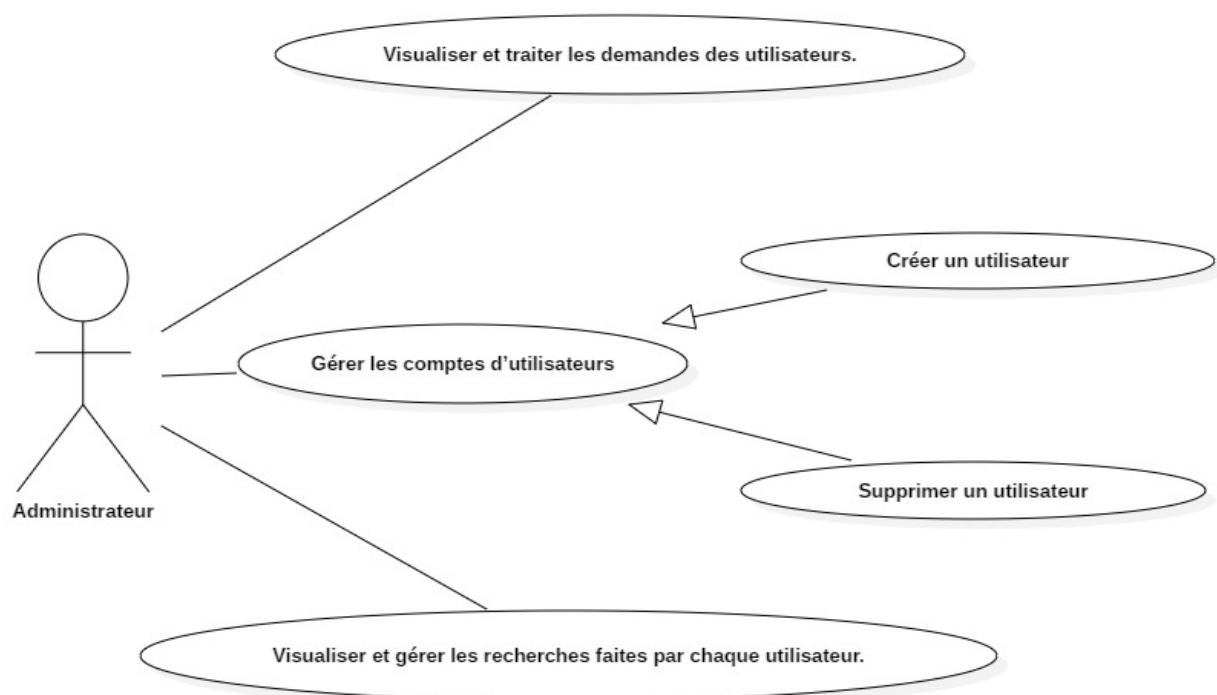


FIGURE 2.3 – Diagramme cas d'utilisation pour l'administrateur

Ce diagramme de cas d'utilisation présente les différentes fonctionnalités que l'administrateur de notre application peut faire. L'administrateur doit d'abord s'authentifier pour accéder à son espace. L'administrateur peut gérer les comptes d'utilisateurs et visualiser leurs analyses. L'administrateur peut également étudier et traiter les nouvelles demandes des utilisateurs.

2.2.2 Description de quelques scénario

2.2.2.1 Scénario du cas d'utilisation "S'identifier"

La figure ci-dessous représente le diagramme de séquence système du scénario **Authentification** pour un client :

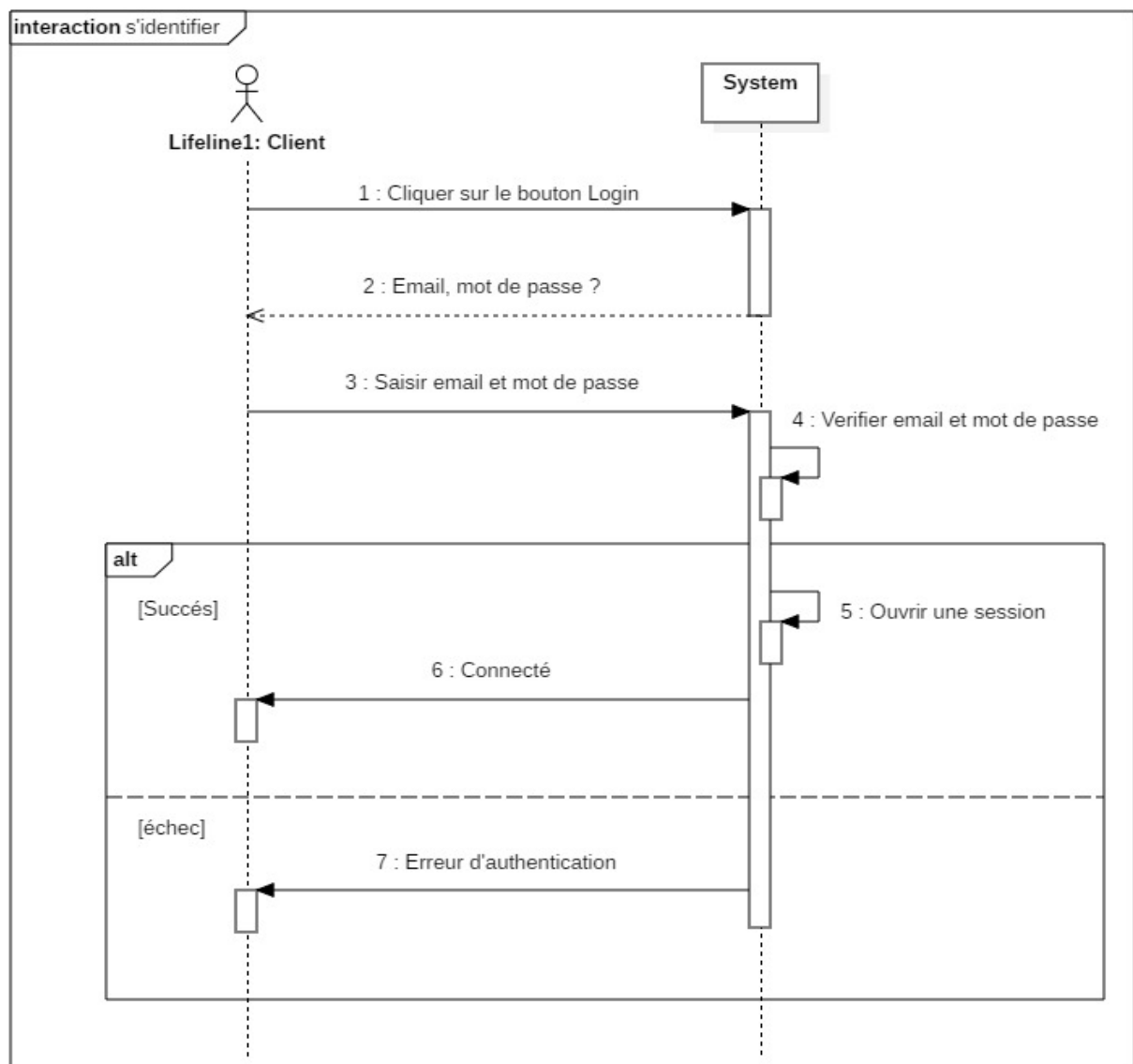


FIGURE 2.4 – Diagramme de séquence système d'authentification

Le cas d'utilisation commence lorsque l'utilisateur accède à la page "Login".

Précondition : L'utilisateur doit être enregistré dans la base de données.

Postcondition : Pas de conditions.

Enchaînement nominal :

1. L'utilisateur accède à l'URL du site.
2. L'utilisateur clique sur le bouton **Login**.
3. Le système fournit un formulaire d'authentification.
4. L'utilisateur doit saisir un email et un mot de passe valides.
5. Le système vérifie les coordonnées saisies par l'utilisateur.
6. L'utilisateur est connecté.

Enchaînement alternatif : Les données saisies par l'utilisateur sont incorrectes : L'enchaînement démarre au point 5 du scénario nominal. Le système indique que l'email ou le mot de passe est erroné via un message d'erreur. La séquence nominale reprend au point 4.

2.2.2.2 Scénario du cas d'utilisation "Effectuer une analyse"

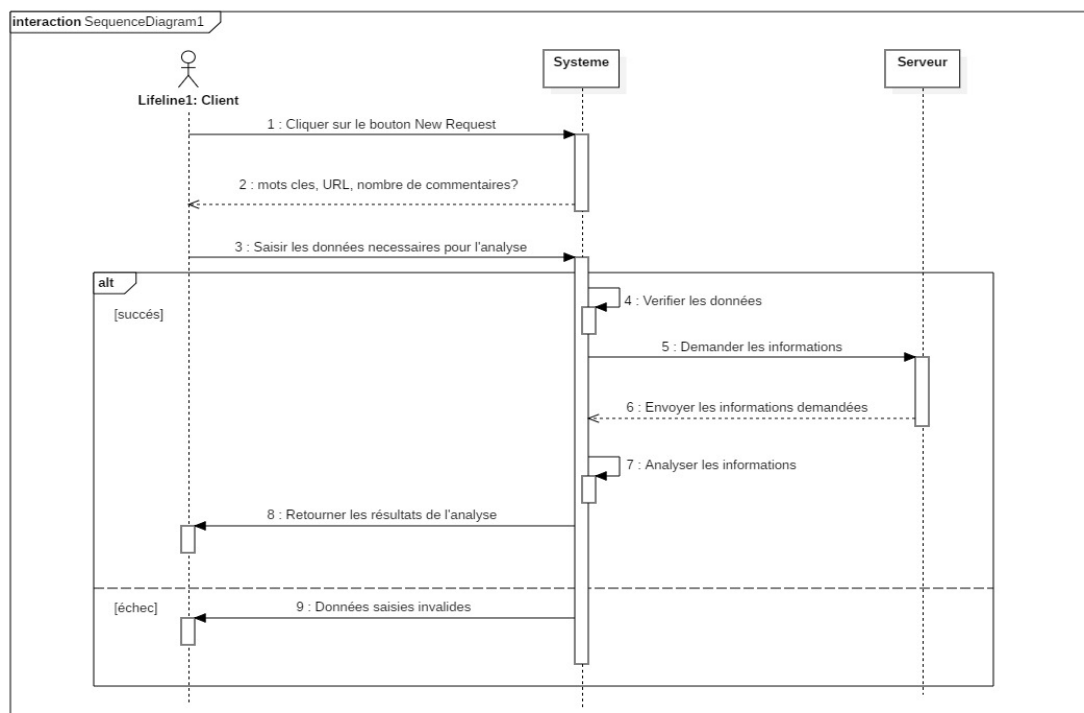


FIGURE 2.5 – Diagramme de séquence système du cas d'utilisation "Effectuer un analyse"

La figure ci-dessus représente le diagramme de séquence système du scénario **Effectuer une analyse** pour un client. Pour effectuer une analyse, le client doit tout d'abord s'identifier. Ensuite il peut effectuer une analyse en choisissant un titre significatif pour la recherche, des mots clés concernant le produit, l'URL et le nombre de commentaires des consommateurs à analyser. En cas de validation des données, l'application lance une requête afin de récupérer les informations demandées par le client à partir du serveur et ensuite les analyser et renvoyer les résultats. Sinon, le client doit vérifier la validité de ses choix.

2.2.2.3 Scénario du cas d'utilisation "Gérer les comptes des utilisateurs"

La figure ci-dessous représente le diagramme de séquence système du scénario **Gestion des comptes** pour un administrateur :

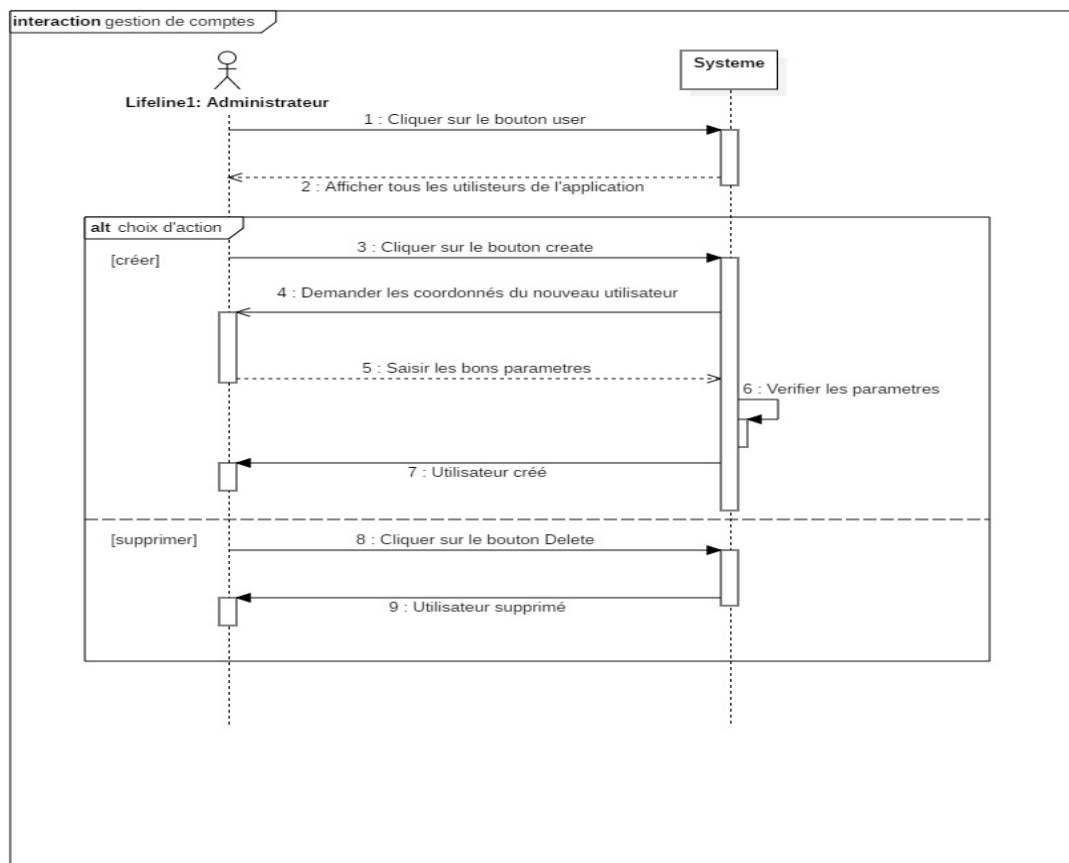


FIGURE 2.6 – Diagramme de séquence système du scénario "Gestion des utilisateurs"

Pour gérer les comptes des utilisateurs, l'administrateur doit tout d'abord s'identifier. Ensuite il peut créer un utilisateur en fournissant des coordonnées valides ou supprimer des utilisateurs de la base de données selon son choix.

Conclusion

Ce chapitre nous a aidés à présenter différents cas d'utilisation de notre application et à dégager les besoins fonctionnels du système.

En effet, cette phase engendre une conception nette et précise du travail et facilite le passage à la phase suivante, à savoir la modélisation de l'application qu'on déroulera dans le chapitre suivant.

Chapitre 3

Conception

Introduction

Dans ce chapitre, nous allons aborder la tâche la plus délicate du projet, à savoir la conception. Nous présenterons, en premier lieu, l'architecture générale de l'application et sa décomposition en différents modules. En deuxième lieu, nous détaillerons chaque module. En nous basant sur UML, nous élaborerons les différents diagrammes de séquences relatifs aux cas d'utilisation qui ont été déroulés dans le chapitre précédent.

3.1 Conception architecturale de l'application

3.1.1 Architecture physique

Avant de mettre en place l'architecture de l'application, il est recommandé d'avoir une vue globale sur les différentes architectures types existantes.

3.1.1.1 Description des architectures Types

Les architectures à présenter dans cette partie sont principalement : l'architecture **2-tiers**, l'architecture **3-tiers** et l'architecture **N-tiers**.

Architecture à deux niveaux (2-tiers) : Aussi appelée architecture client-serveur de première génération. Ce type d'architecture peut s'adapter à toute architecture comportant des composants matériels interconnectés.

Le client se contente de déléguer l'exécution des tâches au serveur, comme par exemple la page login.html, le serveur reçoit cette demande sous forme de requête http , il effectue

ensuite tout un traitement, puis renvoie en réponse la ressource sollicitée par le client.

Architecture à trois niveaux (3-tiers) : Connue aussi sous le nom de client-serveur de deuxième génération ou client-serveur distribué. Dans ce type d'architecture, nous avons trois niveaux de services distincts qui sont :

- **Le client :** Est un client léger vu qu'il n'a aucun rôle de traitement.
- **Le serveur d'application :** Responsable des traitements applicatifs globaux.
- **Le serveur de base de donnée :** S'occupe des traitements sur la base de données.

Dans une application 3-tiers :

- Le client est moins sollicité et son rôle se résume à afficher les réponses du serveur d'application.
- Les données présentes dans la base sont toujours gérées de manière centralisée.
- La logique applicative est traitée par le serveur intermédiaire.

Architecture à plusieurs niveaux (N-tiers) : Conçue principalement pour pallier aux limites des architectures trois tiers et mettre en place une application puissante et simple à maintenir. Au sein d'une architecture n-tiers typique, on a :

- **Une couche présentation :** Englobe les différents clients, qu'il soient légers, lourds ou riches.
- **Une couche applicative :** Elle s'occupe des traitements portant sur les règles métier.
- **Une couche d'objets métier :** Elle englobe les divers objets du domaine, plus concrètement l'ensemble des entités persistantes d'une application.
- **Une couche d'accès aux données :** Elle comprend les usines d'objets métier, chargés de créer et de manipuler les objets métier avec une transparence totale, indépendamment de leur mode de stockage.

3.1.1.2 Choix de l'architecture de l'application

Pour la réalisation de notre application, on a opté pour une architecture 3-tiers. Ce choix est justifié par le fait qu'une architecture 3-tiers :

- Garantit une flexibilité nettement plus importante qu'une architecture 2-tiers : Les applications au niveau serveur sont délocalisées, ce qui implique une plus grande souplesse.
- Une sécurité fiable : Au sein d'une architecture 3-tiers l'accès à la base n'est autorisé qu'au niveau du serveur applicatif.

- Réduit les coûts de déploiement : En effet, L'application elle-même n'est déployée que sur la partie serveur. Le client, lui, a juste besoin d'un navigateur web pour pouvoir y accéder.

Le diagramme de déploiement, représenté dans la figure ci-dessous, illustre la fragmentation de notre application en trois niveaux :



FIGURE 3.1 – Diagramme de déploiement

- **Le client** : Client léger. Il a juste besoin d'un navigateur web pour pouvoir profiter des fonctionnalités de l'application. Il sollicite le serveur web par des «http request» et en reçoit des «http response».
- **Le serveur web** : Est responsable d'un ensemble de traitements. Il regroupe la couche présentation et l'interface reliant l'application avec le SGBD.
- **Le serveur de base de données** : Ce tier contient la base de données de l'application.

3.1.2 Architecture logique

Nous avons choisi l'architecture MVT comme architecture logicielle pour notre application. C'est une architecture orientée autour de trois pôles : le modèle, la vue et le template. Elle s'inspire de l'architecture MVC, très répandue dans les frameworks web. Son objectif est de séparer les responsabilités de chaque pôle afin que chacun se concentre sur ses tâches.

- **Le modèle** : Le modèle interagit avec la base de données. Il a pour but de chercher dans une base de données les items correspondant à une requête et de renvoyer une réponse utilisable par le programme. Les modèles s'appuient sur un ORM (Object Relational Mapping, ou Mapping objet-relationnel).

- **La vue :** La vue fait l'interface avec l'utilisateur. Au premier lieu elle affiche les données récupérées auprès du modèle. Ensuite, elle reçoit toutes les actions de l'utilisateur (clic de souris, saisie d'une entrée, ...). Ces différents événements sont envoyés au template. Dans notre cas, les vues constituent les pages Web et n'effectuent aucun traitement.
- **Le template :** Un template est un document HTML qui reçoit des objets Python et qui est lié à une vue. Un template peut interpréter des variables et les afficher. Il sera récupéré par la vue et renvoyé à l'utilisateur. Cependant, avant d'être envoyé, il sera analysé et exécuté par le framework. Nous présentons le modèle MVT par cette figure :

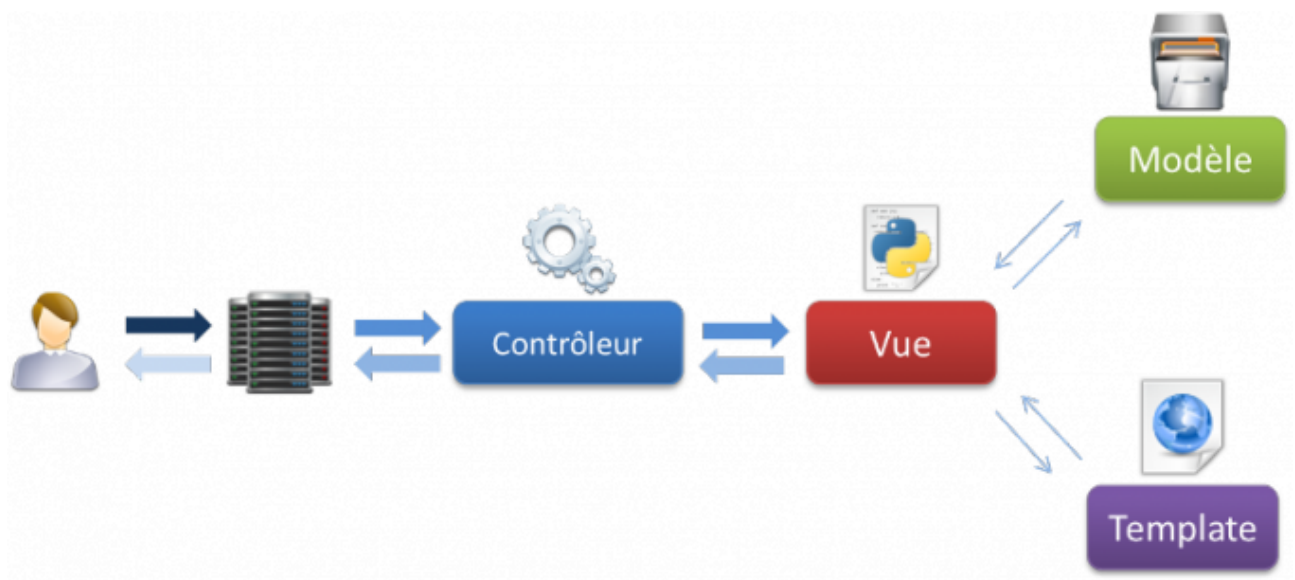


FIGURE 3.2 – Patron de conception MVT

3.1.2.1 Choix de l'architecture

Dernièrement, l'utilisation d'une architecture 3-tiers accompagnée avec une architecture MVT pour le développement des applications web est devenue fréquente, en particulier lorsqu'il s'agit d'une application basée sur des technologies python. Une telle configuration assimile d'une part le modèle de MVT à la couche accès aux données de l'architecture 3-tiers et lui alloue un serveur de base de données, d'autre part le contrôleur est considéré comme la couche métier et enfin la vue est comparée à la couche présentation. Une architecture pareille offre plusieurs avantages :

- **Une conception nette et précise** grâce à la séparation des données.
- **Couplage faible.**
- **Cohésion forte.**
- **Maintenance facile.**
- **Cas de test efficaces** car les services peuvent être testées indépendamment de l'interface.

Il est, désormais, dans l'évidence que les architectures modernes reposent sur les modèles distribués. Celles-ci favorisent une gestion meilleure et une maintenance plus simple. D'autre part, les systèmes modernes sont de plus en plus interactifs, nécessitant alors une architecture qui favorise la gestion des événements et l'interaction. Ce qui justifie, amplement, notre choix en termes d'architecture de l'application à développer.

3.1.2.2 Fonctionnement de l'architecture choisie

Choisir le modèle MVT comme patron d'architecture, signifie fondamentalement qu'on sépare les données de la logique métier et des interfaces utilisateur. Cela présente de nombreux avantages, notamment en termes de compréhension, de maintenance et d'extension de notre code. Nous représentons dans la figure ci-dessous l'architecture logique de notre l'application :

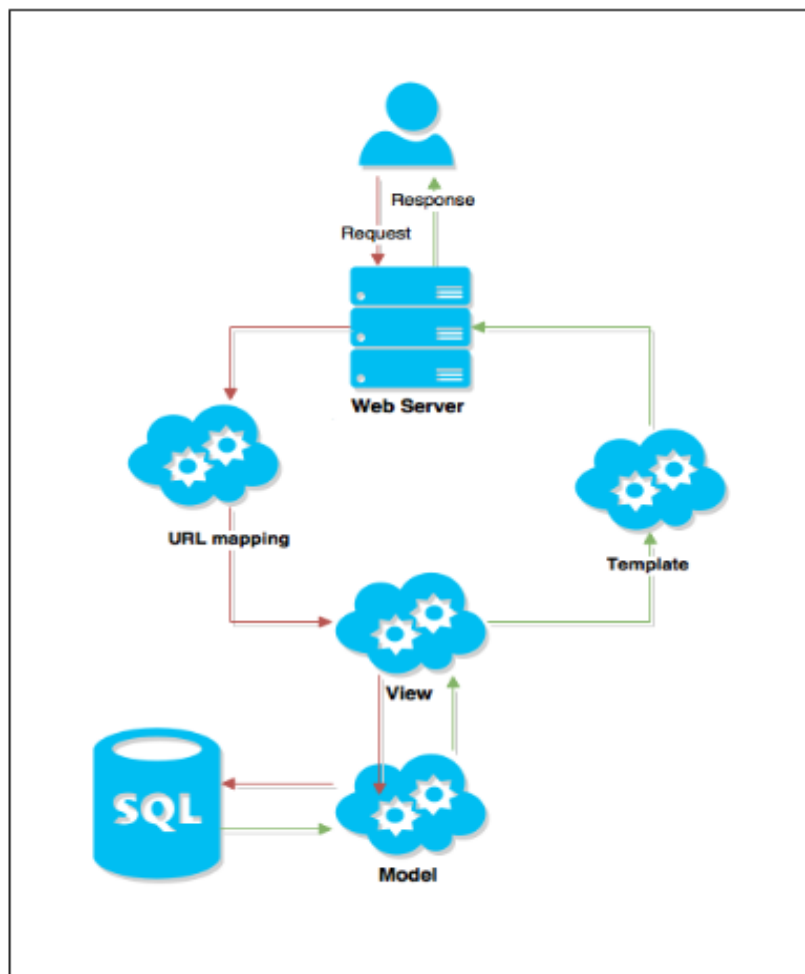


FIGURE 3.3 – Fonctionnement de l'architecture choisie

. Pour mieux éclaircir le déroulement et le fonctionnement de notre architecture, nous allons expliquer les différentes étapes de la figure 3.3 ci-dessus :

Dans une application Flask typique, lorsqu'on envoie une demande au serveur (cliquer sur un lien ou soumettre un formulaire, par exemple), l'application tentera de faire correspondre l'URL demandée à une liste d'URL définies dans le fichier routes. Chaque URL est associée à une fonction spécifique dans le fichier de vues. Ainsi, lorsque l'URL demandée est trouvée, sa fonction de vue est appelée.

La fonction de visualisation prend ensuite en charge le traitement de la demande et renvoie une réponse. Les demandes nécessitent généralement une sorte d'interaction avec la base de données, représentée par les objets définis dans le fichier de modèle. La fonction view interroge la base de données en appelant les objets du modèle et prépare une réponse

au client.

Cette réponse est généralement au format HTML que le navigateur client affichera.

3.2 Conception détaillée

Après avoir défini l'architecture globale, il faut concevoir le modèle des données approprié à l'application qui convient à identifier les divers types d'objets recensés dans cette application et de les modéliser précisément. Il s'agit donc de concevoir la base de données et représenter l'aspect dynamique du système.

3.2.1 Conception de la base de données

3.2.1.1 Modèle Entité-Association

La conception d'un modèle Entité-Association correct est essentielle pour le développement d'une application viable. Notre application nécessite une base de données relationnelle qui permet de stocker toutes les données du système étudié. La figure s illustre le diagramme Entité-Association de notre application :

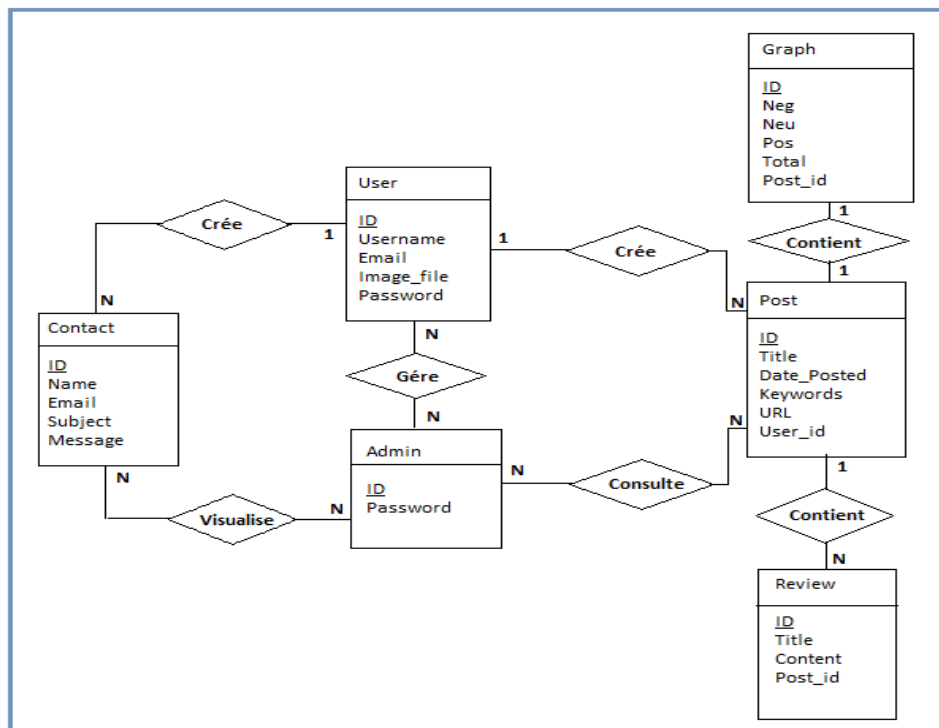


FIGURE 3.4 – Modèle Entité-Association

3.2.1.2 Identification des entités

Les entités renferment des attributs et des clés primaires qui seront représentés dans le tableau suivant :

Entité	Identifiant	Attributs	Libellé
User	ID	username	Nom d'utilisateur du client
		email	Email du client
		Image_file	Image du profil du client
		password	Le mot de passe du client
		posts	Les analyses demandées par le client
Post	ID	title	Le titre de la requête
		date_posted	La date de la requête
		keywords	Les mots clés relatifs à la requête
		url	Le lien où s'effectue la requête
		User_id	L'identifiant du client de la requête
		reviews	Les commentaires relatifs à la requête
		graphs	Le graphe relatif à la requête
Review	ID	title	Le numéro du commentaire
		content	Le contenu du commentaire
		post_id	L'identifiant de la requête
Graph	ID	neg	Le nombre des commentaires négatifs
		neu	Le nombre des commentaires neutres
		pos	Le nombre des commentaires positifs
		total	Le nombre total des commentaires
		Post_id	L'identifiant relatif da la requête

Admin	ID	password	Le mot de passe de l'administrateur
Contact	ID	name	Le nom du visiteur du site ou du client
		email	Email du visiteur de site ou du client
		subject	Le sujet de la demande
		message	Le contenu de la demande

TABLE 3.1 – Diagramme Entité-Association

3.2.1.3 Identification des associations

Plusieurs associations ont été perçues dans notre application. Nous les présentons dans le tableau suivant :

Nom de l'association	Entités impliquées	Type	Description
Créer	User-Contact	1:N	Un client peut créer plusieurs demandes.
Visualiser	Admin-Contact	N:N	Plusieurs administrateurs peuvent visualiser plusieurs demandes
Gérer	Admin-User	N:N	Plusieurs administrateurs peuvent gérer plusieurs comptes de clients
Consulter	Admin-Post	N:N	Plusieurs administrateurs peuvent consulter plusieurs requêtes
Créer	User-Post	1:N	Un client peut créer plusieurs requêtes
Contenir	Post-Graph	1:1	Une requête peut contenir un seul graph
Contenir	Post-Review	1:N	Une requête peut contenir plusieurs commentaires

TABLE 3.2 – Table des associations

3.2.1.4 Modèle logique des données

Le modèle logique des données permet de modéliser la structure selon laquelle les données seront stockées dans la future base de données. C'est un modèle relationnel créé à partir d'un modèle Entité-Association. Le modèle logique décrivant notre base de données est le suivant (**les clés primaires sont soulignées et les clés étrangères sont précédées par '#'**) :

User (#ID, username, email, image_file, password)

Admin (ID, password)

Contact (ID, name, email, subject, message)

Post (#ID, user_id, title, date_posted, keywords, url)

Review (ID, post_id, title, content)

Graph (ID, post_id, neg, neu, pos, total)

3.2.2 Diagrammes de séquences d'objets

Pour modéliser l'aspect dynamique du système, nous présentons dans ce paragraphe quelques diagrammes de séquences. En effet, un diagramme de séquences permet de décrire **COMMENT** les éléments du système interagissent entre eux et avec les acteurs en tenant compte de la chronologie d'envois de messages. Nous présentons dans ce qui suit les diagrammes de séquences « **S'inscrire** » et « **Effectuer une analyse** ».

3.2.2.1 Diagramme de séquences objets «S'inscrire»

La figure 3.5 ci-dessous résume les différentes étapes par lesquelles passe le système afin de faire une inscription dans notre site. Il faut tout d'abord appuyer sur le bouton "registre". La requête http passe du navigateur web au serveur de développement Flask. L'utilisateur sera redirigé vers la page de l'inscription où il y a un formulaire à remplir avec ses propres coordonnées convenablement. les paramètres passent par une vérification, si ces dernières sont valables, elles seront enregistrées dans la base de donnée et un message de succès serait affiché et le client sera renvoyé vers la page de Login pour s'identifier. Sinon un message d'erreur sera renvoyé au utilisateur.

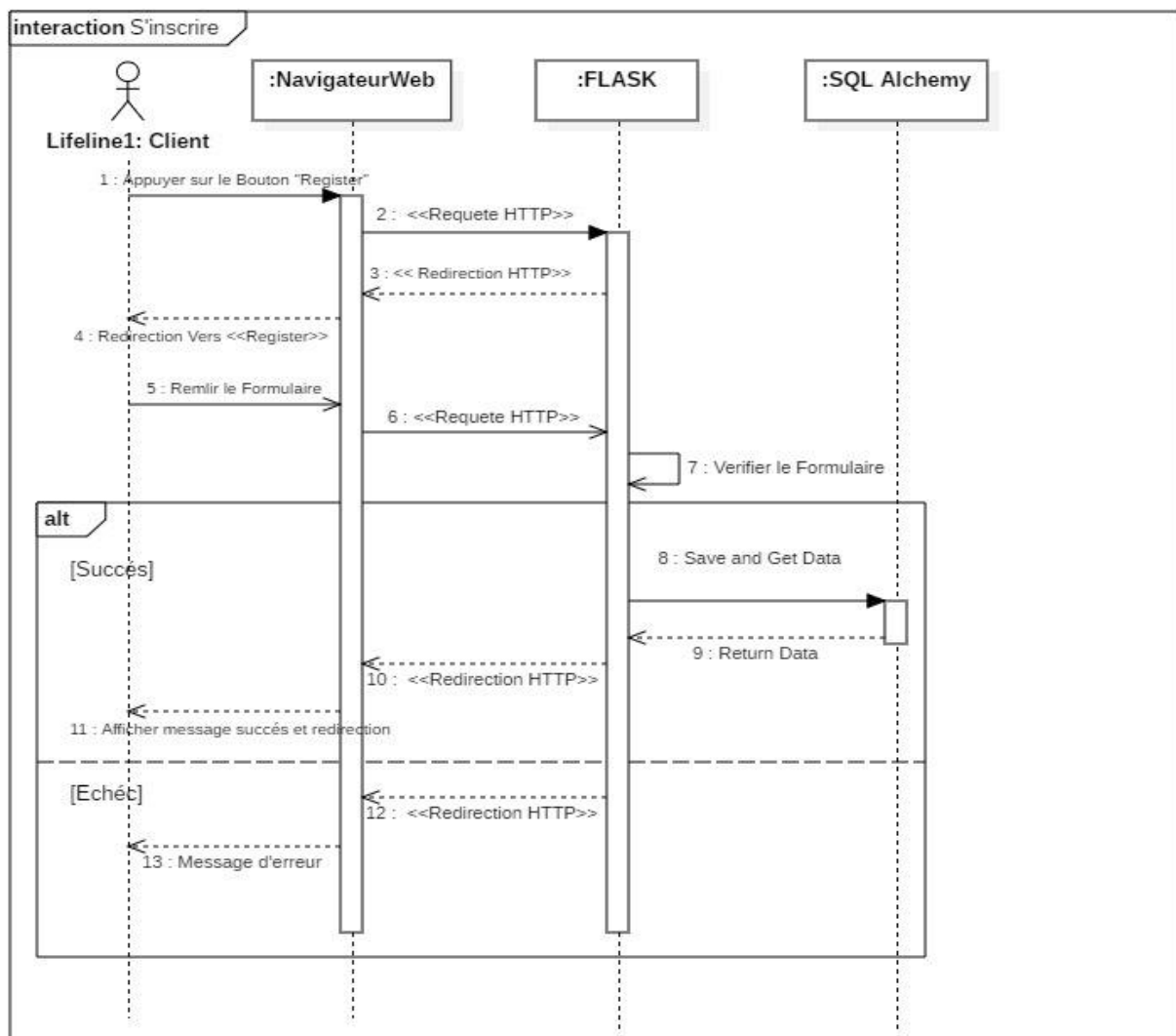


FIGURE 3.5 – Diagramme de séquences objets «S'inscrire»

3.2.2.2 Diagramme de séquences objets «Effectuer une analyse»

La figure ci-dessous résume les différentes étapes par lesquelles passe le système afin d'effectuer une analyse. Tout d'abord, le client doit s'authentifier. Lorsqu'on appuie sur le bouton "Login", la requête http passe du navigateur web au serveur de développement Flask. L'authentification passe par une vérification des données. Une fois nous avons l'accès à la page d'accueil, nous pourrions ouvrir le formulaire pour effectuer une nouvelle requête en appuyant sur le bouton "New Request". Les mêmes étapes de l'envoi de la requête HTTP et de sa redirection auront lieu. Le client pourra donc remplir le formulaire de la requête. Au final, les informations envoyées à travers le formulaire seront traitées afin de récupérer

les données demandées par le client et les analyser. Si les paramètres saisis par le client sont valides un message de succès serait affiché, sinon un message d'erreur sera renvoyé au client.

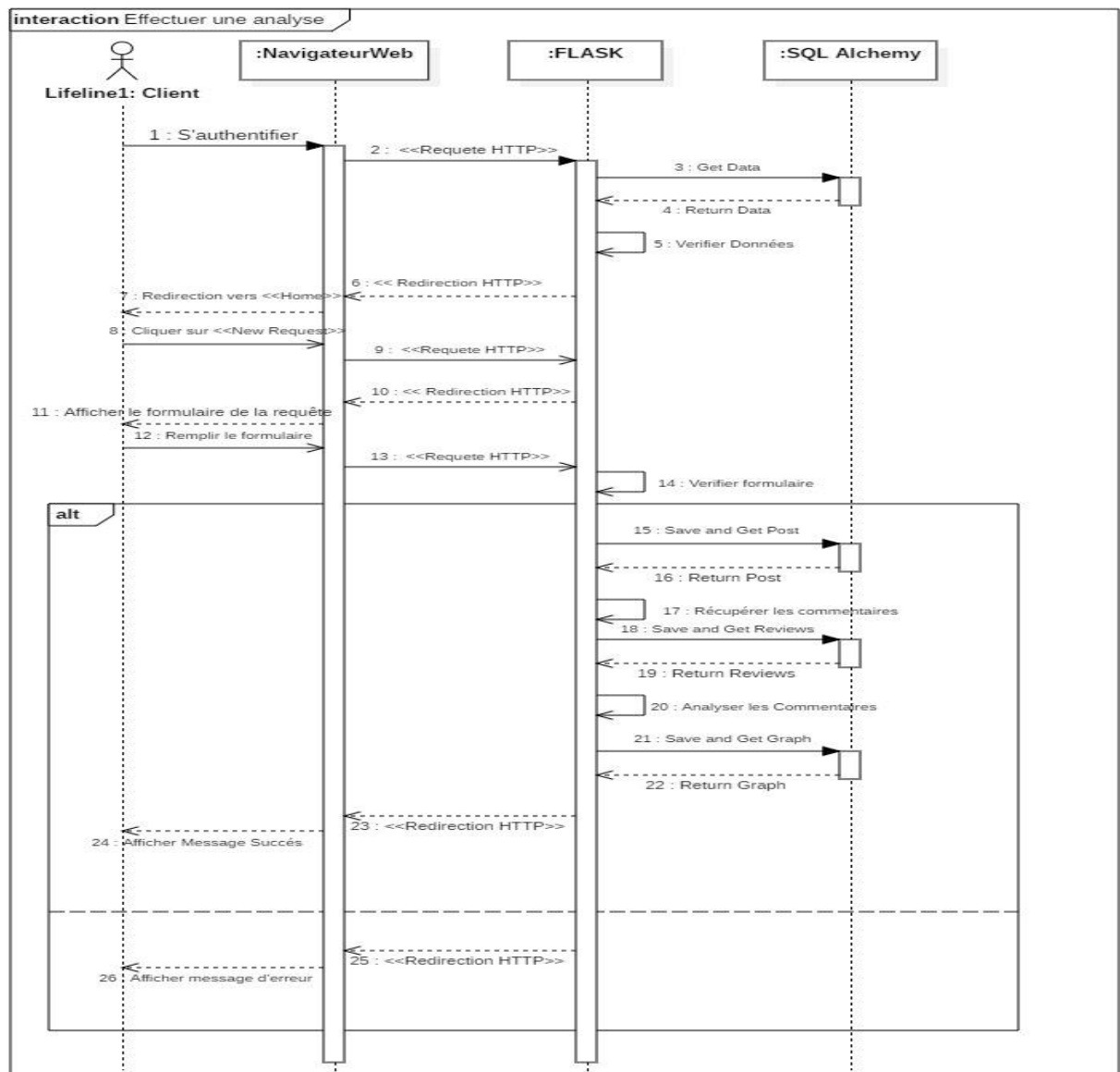


FIGURE 3.6 – Diagramme de séquences objets «Effectuer une analyse»

Conclusion

Dans ce chapitre, nous avons commencé par donner une conception architecturale logique et physique de notre application. Ensuite, nous avons présenté une conception détaillée à l'aide des diagrammes UML. Cette phase est nécessaire pour faciliter l'étape de la réalisation que nous allons la présenter dans le chapitre suivant.

Chapitre 4

Réalisation

Introduction

Après avoir achevé l'étape de la conception, en tenant compte des besoins fixés et des choix conceptuels effectués, nous consacrons ce chapitre à la description du travail réalisé. Nous commençons par décrire l'environnement matériel et logiciel sur lequel nous avons développé notre application. Ensuite, nous signalons l'apport de l'intelligence artificielle sur le déroulement de notre application, après nous identifions l'état d'avancement du projet et nous mettons en évidence le travail réalisé par la présentation de quelques captures d'écrans traduisant le déroulement du projet. Enfin, nous finissons par un chronogramme qui décrit toutes les étapes de mise en oeuvre du travail.

4.1 Environnement de travail

Dans cette partie nous nous intéressons à l'étude de l'environnement technique disponible pour la réalisation de notre projet, ensuite nous justifions les choix pris en matière d'environnement logiciel pour mener à terme la partie applicative.

4.1.1 Environnement matériel

Le développement de l'application est réalisé sur deux ordinateurs ayant les caractéristiques suivantes :

— Première station de travail :

- Marque : Laptop Lenovo Z50-70
- Processeur : Intel(R) Core(TM) i5-4210U CPU @ 1.70GHz 2.40 GHz
- Mémoire : 6,00 Go de RAM

- Disque Dur : 1 TB
- Ecran : 17
- Deuxième station de travail :
 - Marque : Laptop ASUS X550
 - Processeur : Intel(R) Core(TM) i5-4210U CPU @ 1.70GHz 2.40 GHz
 - Mémoire : 8,00 Go de RAM
 - Disque Dur : 500 Go
 - Ecran : 17

4.1.2 Environnement logiciel

Cette section présente les différents outils logiciels que nous avons exploités pour réaliser notre application. Pour la réalisation de notre projet, nous avons opté pour l'utilisation de l'environnement logiciel suivant :

- Sublime Text : Sublime Text est un éditeur de texte générique codé en C++ et Python, disponible sur Windows, Mac et Linux. Le logiciel a été conçu tout d'abord comme une extension pour Vim, riche en fonctionnalités.
- ShareLatex : Environnement de développement LATEX utilisé pour rédiger ce rapport. Il permet de créer, éditer, partager un document latex et ce en ligne, sans aucune installation préalable.
- StarUML : C'est un outil de modélisation UML, gratuit sur le plan public. Il aide à dessiner des diagrammes jusqu'à 3 fois plus vite par rapport à un logiciel de diagramme classique.
- MySql : MySQL est un Système de Gestion de Bases de Données Relationnelles (abrégé SGBDR). C'est-à-dire un logiciel qui permet de gérer des bases de données. Il utilise pour cela le langage SQL. Il s'agit d'un des SGBD les plus courants et les plus fréquents.

4.1.3 Choix technologiques

Dans ce paragraphe nous présentons les différentes technologies utilisées pour la réalisation de notre application :

- Python : C'est un langage de programmation objet, multi-paradigme et multiplateformes. Il favorise la programmation impérative structurée, fonctionnelle et orientée

objet. Il est doté d'un typage dynamique fort, d'une gestion automatique de la mémoire, d'une syntaxe simple à utiliser et d'un système de gestion d'exceptions.

- Flask : C'est un framework open-source de développement web en Python. Son but principal est d'être léger, afin de garder la souplesse de la programmation Python, associé à un système de templates. Parmi ces fonctionnalités :
 - Contient un serveur de développement et un debugger
 - Supporte les tests unitaires.
 - Utilise le moteur de template Jinja2.
 - Supporte les cookies sécurisés.
- SQLAlchemy : C'est un toolkit open source SQL et un mapping objet-relationnel (ORM) écrit en Python.
- BeautifulSoup : c'est une bibliothèque Python permettant d'extraire des données HTML, XML et d'autres langages de balisage. BeautifulSoup nous permet d'extraire un contenu particulier d'une page Web, de supprimer le balisage HTML et d'enregistrer les informations.
- Textblob : TextBlob fournit une API capable d'exécuter différentes tâches de traitement du langage naturel telles que le balisage partiel du discours, l'analyse des sentiments, la classification (Naive Bayes, Arbre de décision), la traduction et la détection de la langue et la correction orthographique.

4.2 Explication du travail réalisé

Dans cette partie ,au premier lieu nous allons montrer la démarche utilisée pour récupérer les données demandées par le client puis nous détaillerons l'aspect intelligence artificielle utilisé dans notre application web pour analyser ces informations.

4.2.1 Partie Récupération

Le Web Scraping est une technique permettant d'accéder automatiquement à de grandes quantités d'informations d'un site Web et de les extraire, ce qui permet de faire des économies de temps et d'efforts. Pour analyser le contenu d'une page web il faut générer du trafic pour le serveur qui l'héberge. Ce type de trafic "non humain" n'est souvent pas très bien accepté par les administrateurs des sites qui disposent de quelques instruments techniques pour le limiter :

- Limitation des requêtes dans un délais de temps : les administrateurs peuvent limiter le nombre de "hit" provenant de la même source car un navigateur "humain" nécessite normalement un certain temps pour passer d'une page à l'autre.
- Bloquer des adresses IP : si un site reçoit trop de requêtes en même temps, il y a le risque d'enchaîner un Denial of Service (DoS), c'est-à-dire que le serveur ne peut satisfaire toutes les requêtes qu'il reçoit et pour éviter des problèmes il préfère couper les services.
- Limitation de la bande passante : visiter un site signifie télécharger son contenu, ce qui implique la "consommation" de bande passante d'un serveur. Dans le cas d'un petit serveur, cette bande peut être limitée et une fois dépassée, le site n'est plus disponible jusqu'au mois prochain.
- Obligation de saisir des données pour accéder à un contenu : surtout dans le cadre d'ouverture de compte online, les sites demandent de saisir des champs qu'une machine ne peut pas remplir (e.g. saisir le texte tiré d'une image, CAPTCHA, etc.).

Pour ces raisons notre application s'est limitée à 3 sites par défaut : **Amazon**, **Ebay** et **Twitter**.

4.2.1.1 Les méthodes d'accès

Dans notre système nous avons utilisé 2 méthodes afin d'accéder au contenu d'un site web :

- Les APIs
- Les fonctionnalités de la bibliothèque `urllib.request`.

Les APIs :

Le terme API est un acronyme et signifie «Application Programming Interface». Une API est un ensemble d'opérations accompagnée d'une description que les développeurs peuvent utiliser. Les API permettent aux développeurs de gagner du temps en tirant parti de l'implémentation d'une plate-forme pour effectuer le travail essentiel. Cela permet de réduire la quantité de code nécessaire aux développeurs et de créer davantage de cohérence entre les applications pour la même plate-forme. Les API peuvent contrôler l'accès aux ressources matérielles et logicielles.

API Twitter : La plupart des API d'entreprise de la plate-forme Twitter demandent l'utilisation de l'authentification HTTP de base, construite à partir d'une combinaison d'adresse électronique et mot de passe valide, en tant que moyen d'interagir avec les points

de terminaison de l'API. Les informations d'identification doivent être passées comme en-tête d'autorisation pour chaque demande. Donc tout d'abord on doit s'authentifier comme suit :

```
import tweepy
from tweepy import OAuthHandler

consumer_key = "Your API KEY "
consumer_secret = "Your API SECRET "
access_token = "Your ACCESS TOKEN"
access_secret = "Your ACCESS TOKEN SECRET "

#IDENTIFICATION ET CONNEXION
auth = OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_secret)

api = tweepy.API(auth)
```

FIGURE 4.1 – L'authentification pour l'API Twitter

Tweepy : Tweepy est l'association des mots Twitter et Python :

Twitter + Python = Tweepy

Tweepy est une bibliothèque Python qui communique avec l'API de Twitter .Elle permet ainsi d'écrire des scripts Python reproduisant l'usage d'un abonné sur Twitter et surtout de profiter de plusieurs autres fonctionnalités que nous offre l'API du réseau social.

Les fonctionnalités de la bibliothèque urllib.request :

Urllib.request est un module Python permettant de récupérer des URL. Il offre une interface très simple, sous la forme de la fonction **urlopen**. Cette fonction est capable d'extraire des URL en utilisant divers protocoles. La figure 4.2 ci-dessous montre l'utilisation de la fonction urlopen :

```
import urllib.request
with urllib.request.urlopen('http://python.org/') as response:
    html = response.read()
```

FIGURE 4.2 – L'utilisation la fonction urlopen du module urllib.request

4.2.1.2 Les méthodes de collecte de données

Dans notre système nous avons utilisé deux méthodes afin d'analyser le contenu d'un site web :

- Les APIs
- Les fonctionnalités de la bibliothèque BeautifulSoup.

API Twitter :

Dans cette étape l'API Twitter offre plusieurs options ce qui rend cette tâche simple et plus efficace. En effet, l'utilisation de la pagination est importante dans le développement des API Twitter. Pour pouvoir effectuer la pagination, on doit fournir un paramètre page / curseur avec chacune de nos demandes. Le problème ici est que cela nécessite beaucoup de code pour gérer la boucle de pagination. Pour faciliter la pagination et nécessiter moins de code, Tweepy possède l'objet Cursor. Pour mieux expliquer, on peut faire une petite comparaison entre les deux versions "Sans Cursor" et "Avec Cursor". Comme le montre la figure ci-dessous, on doit gérer le paramètre «page» manuellement dans notre boucle de pagination.

```
page = 1
while True:
    statuses = api.user_timeline(page=page)
    if statuses:
        for status in statuses:
            # process status here
            process_status(status)
    else:
        # All done
        break
    page += 1 # next page
```

FIGURE 4.3 – Version Sans Cursor

Avec le Cursor on aura tout simplement cette instruction comme le montre la figure ci-dessous :

```
for status in tweepy.Cursor(api.user_timeline).items():
    # process status here
    process_status(status)
```

FIGURE 4.4 – Version Avec Cursor

Il est clair que la version "Avec Cursor" est plus facile, en effet, Cursor gère tout le travail de pagination afin que le code puisse désormais se concentrer entièrement sur le traitement des résultats. Ainsi, on peut ajouter au Cursor deux paramètres (api.search, mots_clés) pour spécifier les commentaires à récupérer. La fonction items(n) permet de contrôler le nombre de tweets.

Donc on peut résumer la méthode pour collecter les données à partir de Twitter en une seule instruction comme le montre la figure ci-dessous :

```
noOfSearchTerms = int(nb)
tweets = tweepy.Cursor(api.search, q=mots_cle).items(noOfSearchTerms)
```

FIGURE 4.5 – Exemple d’instruction pour récupérer les données avec API Twitter

Les fonctionnalités de la bibliothèque beautifulsoup :

On va présenter un petit exemple qui illustre l’utilisation des fonctions de la bibliothèque beautifulsoup.

Exemple de fonctionnement : On suppose qu’on a un document HTML et qu’on veut collecter tous les liens de référence qu’il contient. Tout d’abord, on stocke le document sous forme de chaîne comme le montre la figure ci-dessous :

```
html_doc='''<a href='www.example.com'/a>
<a href='www.codesdope.com'/a>
<a href='www.google.com'/a>
<a href='www.facebook.com'/a>
<a href='www.wikipedia.org'/a>
<a href='www.twitter.com'/a>
<a href='www.microsoft.com'/a>
<a href='www.github.com'/a>
<a href='www.nytimes.com'/a>
<a href='www.youtube.com'/a>
<a href='www.reddit.com'/a>
<a href='www.python.org'/a>
<a href='www.stackoverflow.com'/a>
<a href='www.amazon.com'/a>
<a href='www.linkedin.com'/a>'''
```

FIGURE 4.6 – Exemple d’un document HTML pour analyser son contenu

Maintenant, on passe la variable `html_doc` dans la fonction d'initialisation BeautifulSoup. De cette façon, on crée un objet soupe.

```
from bs4 import BeautifulSoup
soup = BeautifulSoup(html_doc, 'html.parser')
```

FIGURE 4.7 – Création de l'objet soupe

Maintenant on a cet objet soupe, on peut y appliquer les méthodes de la classe BeautifulSoup. Une de ces méthodes est `find_all`, qui prend les balises HTML en tant que chaînes. Donc, on peut localiser la balise `<a>` facilement, ainsi on peut accéder aux attributs d'une balise et aux valeurs des attributs avec des méthodes bien nommées.

```
for tag in soup.find_all('a'):
    print tag.get('href')
```

FIGURE 4.8 – Exemple d'utilisation de la fonction `find_all()` de BeautifulSoup

Voilà d'une façon générale ce qui concerne le déroulement de l'extraction des commentaires à l'aide de la bibliothèque BeautifulSoup à partir d'une chaîne.

4.2.2 Partie Analyse

Dans cette partie nous nous sommes basés sur le concept du "Deep Learning".

4.2.2.1 Deep Learning

À l'ère du Deep Learning, nous pourrions être tentés de passer directement à des architectures compliquées et des machines puissantes, car elles se sont révélées fournir des résultats incroyables dans divers domaines. Cependant, nous devrions toujours commencer par créer une base solide que nous devons implémenter par nous-même pour démontrer et comprendre la valeur de modèles plus avancés. Nous tentons dans cette partie d'implémenter un outil utile à appliquer aux problèmes de classification de texte.

Classification de texte

La classification de texte vise à attribuer des documents (tweets et commentaires clients dans notre projet) à une ou plusieurs catégories. Certaines applications de la classification de texte sont le filtrage du courrier indésirable, le routage des tickets de support client ou la prédiction de rubriques. L'analyse des sentiments, qui a pour objectif de déterminer le point de vue de l'auteur sur un sujet quelconque en est une autre application.

Dans cette partie, nous visons à classer les commentaires récupérés des différents sites et réseaux sociaux en commentaires positifs ou négatifs ou neutres. Comment résoudre notre problème de classification en utilisant le Deep learning ? La bibliothèque TextBlob nous donne accès à un bon classificateur : NaiveBayesClassifier.

NaiveBayes Classifier

La méthode de classification naïve bayésienne est un algorithme d'apprentissage supervisé permettant de classer un ensemble d'observations selon des règles bien déterminées par l'algorithme en question. Cet outil de classification doit, en premier lieu être entraîné sur un jeu de données, dites données d'apprentissage, qui montre la classe attendue en fonction des entrées. Durant cette phase, l'algorithme élabore ses règles de classification sur le jeu de données passé en paramètre, pour les appliquer dans un second temps à la classification d'un jeu de données de prédiction.

Le classificateur bayésien **naïf** implique que les classes du jeu de données d'apprentissage soit connu et fourni, d'où le caractère supervisé de l'outil.

A la base de la classification naïve bayésienne se trouve le théorème de Bayes avec l'hypothèse simplificatrice, dite naïve, d'indépendance entre toutes les paires de variables.

$$P_B(A) = \frac{P(A \cap B)}{P(B)} \quad (4.1)$$

Représentation des données

Dans cette partie nous préparons un modèle de classification de texte simple qui va être utilisé par TextBlob. Pour cela, nous devons d'abord préparer les données d'apprentissage et de test. Chaque entrée comporte une phrase en anglais suivie de sa classification (**neg** pour négatif, **pos** pour positif).

```
training = [  
    ('Tom Holland is a terrible spiderman.','pos'),  
    ('a terrible Javert (Russell Crowe) ruined Les Miserables for me...','pos'),  
    ('The Dark Knight Rises is the greatest superhero movie ever!','neg'),  
    ('Fantastic Four should have never been made.','pos'),  
    ('Wes Anderson is my favorite director!','neg'),  
    ('Captain America 2 is pretty awesome.','neg'),  
    ('Let's pretend "Batman and Robin" never happened..','pos'),  
]  
testing = [  
    ('Superman was never an interesting character.','pos'),  
    ('Fantastic Mr Fox is an awesome film!','neg'),  
    ('Dragonball Evolution is simply terrible!!','pos')  
]
```

FIGURE 4.9 – Les données d'apprentissage et de test

Entraînement du modèle

Pour la répartition des entrées, nous avons opté pour un ratio 80/20, appelé ratio de Pareto. C'est à dire consacrer 80% du jeu de données d'origine à l'apprentissage et les 20% restants à la validation. L'entraînement du modèle nécessite un grand nombre de données d'apprentissage et de test.

Pour éviter de saisir toutes les entrées à la main, nous utilisons le corpus `movie_reviews` de la NLTK comme données de formation étiquetées. Le corpus `movie_reviews` contient plus de 2000 critiques de films avec classification des sentiments.

Résultat d'apprentissage

Après l'apprentissage on a obtenu de bons résultats avec une précision de 95,84% comme le montre la figure ci-dessous :

```
>>> cl.show_informative_features(10)
Classifier accuracy percent: 95,84
Most Informative Features
      contains(good) = True          pos : neg    =    10.7 : 1.0
      contains(bad) = True          neg : pos    =    15.6 : 1.0
      contains(better) = True       pos : neg    =    12.4 : 1.0
      contains(amazing) = True      pos : neg    =    11.8 : 1.0
      contains(not) = True          neg : pos    =    14.3 : 1.0
      contains(love) = True         pos : neg    =    10.0 : 1.0
      contains(awesome) = True      pos : neg    =    16.0 : 1.0
      contains(hate) = True         neg : pos    =    11.9 : 1.0
      contains(very) = True        pos : neg    =    11.1 : 1.0
      contains(bad) = True         neg : pos    =    12.6 : 1.0
```

FIGURE 4.10 – Résultat d'apprentissage

Le modèle qu'on a entraîné a appris à classifier les entrées en se basant sur la présence de certains mots dans le texte, par exemple on peut voir dans la figure que la présence du mot “awesome” signifie que cette entrée a 16 fois plus de chance d'être classifiée comme entrée positive que négative.

4.2.3 Aperçu sur le travail réalisé

Dans cette partie, nous exposons l'application réalisée en montrant ses interfaces graphiques et les fonctionnalités de l'application.

4.2.3.1 Interface d'accueil

La figure ci-dessous représente la page d'accueil. En haut de cette interface on trouve le menu qui permet à l'utilisateur de s'authentifier s'il a déjà un compte en cliquant sur «**Login**» ou bien de s'inscrire en cliquant sur «**Register**». Cette interface met aussi en valeur les diverses fonctionnalités de notre application.

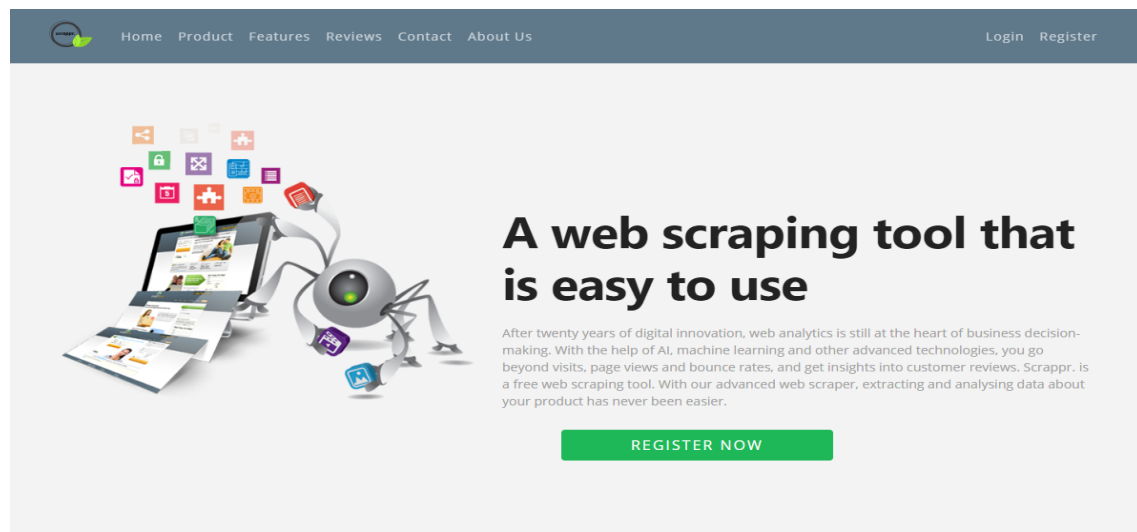


FIGURE 4.11 – Interface d'accueil

4.2.3.2 Interface d'inscription

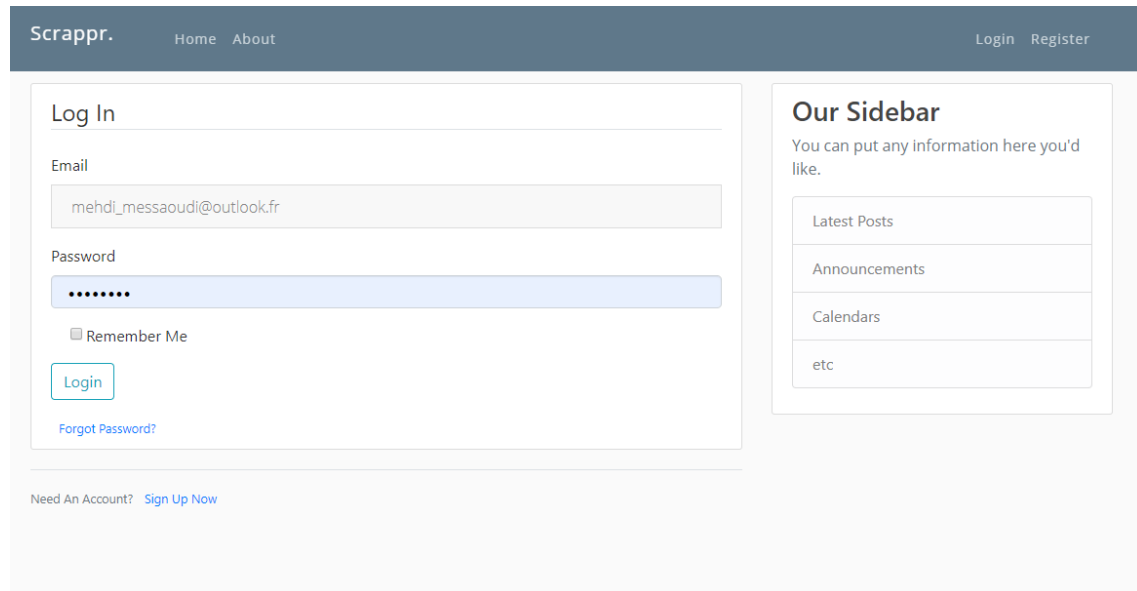
La figure ci-dessous représente l'interface d'inscription. Pour s'inscrire, l'utilisateur doit saisir son nom et son mot de passe puis confirmer son mot de passe pour éviter les fautes de frappe.

The image shows the registration page of the Scrapppr website. The top navigation bar is dark blue with the Scrapppr logo and links for Home and About. On the right side of the bar are links for Login and Register. The main content area is light gray. On the left, there is a white box with a light gray border containing the registration form. The form has a heading "Join Today" followed by four input fields: Username, Email, Password, and Confirm Password. Below these fields is a blue button with the text "Sign Up". At the bottom of the form box, there is a link that says "Already Have An Account? Sign In". To the right of the form box is a sidebar titled "Our Sidebar" with the text "You can put any information here you'd like." Below this text is a list of items: Latest Posts, Announcements, Calendars, and etc., each in its own box.

FIGURE 4.12 – Interface d'inscription

4.2.3.3 Page d'authentification

Elle constitue l'interface commune à tous les utilisateurs, qui leur permet de s'authentifier pour pouvoir profiter des fonctionnalités du site, et ce en saisissant un nom d'utilisateur et un mot de passe comme le montre la figure.

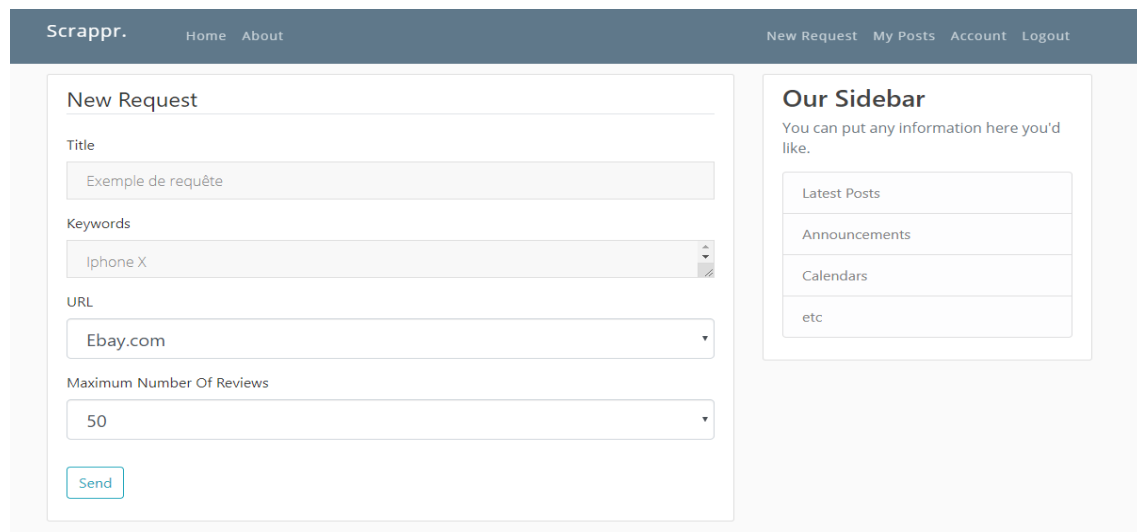


The screenshot displays the login page for 'Scrapppr'. The header is dark blue with the site name 'Scrapppr.' and links for 'Home' and 'About' on the left, and 'Login' and 'Register' on the right. The main content area is light gray. On the left, there is a 'Log In' section with an 'Email' field containing 'mehdi_messaoudi@outlook.fr', a 'Password' field with masked characters, a 'Remember Me' checkbox, a 'Login' button, and a 'Forgot Password?' link. Below this is a link for 'Need An Account? Sign Up Now'. On the right, there is a 'Our Sidebar' section with the text 'You can put any information here you'd like.' and a list of links: 'Latest Posts', 'Announcements', 'Calendars', and 'etc'.

FIGURE 4.13 – Interface d'authentification

4.2.3.4 Interface pour effectuer une requête

Après son authentification, le client peut effectuer une analyse afin de récupérer les commentaires sur un produit donné. Pour cela, il doit remplir un formulaire en fournissant un titre pour ça requête, le mot clé qu'il recherche, puis en choisissant le site depuis lequel les commentaires vont être récupérées (Amazon, Ebay, Twitter. . .) et enfin en spécifiant le nombre maximum de commentaires à récupérer. Cette interface est illustrée dans la figure suivante :



The screenshot shows the 'New Request' form on the Scrapppr website. The form includes fields for Title, Keywords, URL, and Maximum Number Of Reviews. The Title field contains 'Exemple de requête', Keywords contains 'Iphone X', URL contains 'Ebay.com', and Maximum Number Of Reviews contains '50'. A 'Send' button is at the bottom. To the right is a sidebar titled 'Our Sidebar' with a list of categories: Latest Posts, Announcements, Calendars, and etc.

Field	Value
Title	Exemple de requête
Keywords	Iphone X
URL	Ebay.com
Maximum Number Of Reviews	50

FIGURE 4.14 – Interface pour effectuer une requête

4.2.3.5 Interfaces d’affichage des résultats de l’analyse

Après avoir rempli le formulaire, le client peut visualiser l’ensemble des commentaires récupérés ainsi que visualiser le graphe qui illustre les pourcentages des commentaires positifs, négatifs et neutres parmi les données collectées et analysées. Ces Interfaces sont illustrées ci-dessous :

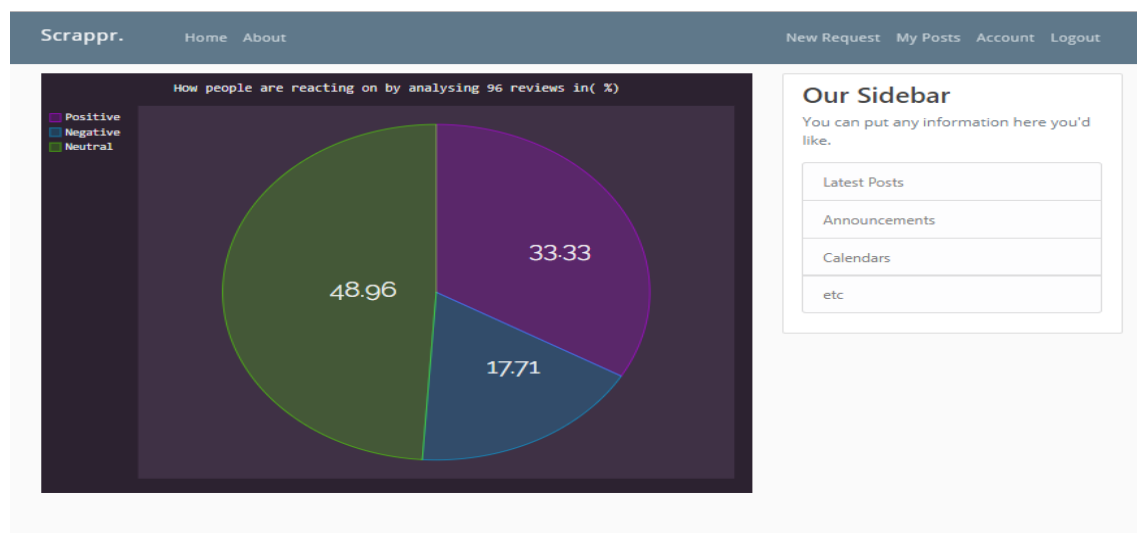


FIGURE 4.15 – Graphe montrant les réactions du consommateurs en%

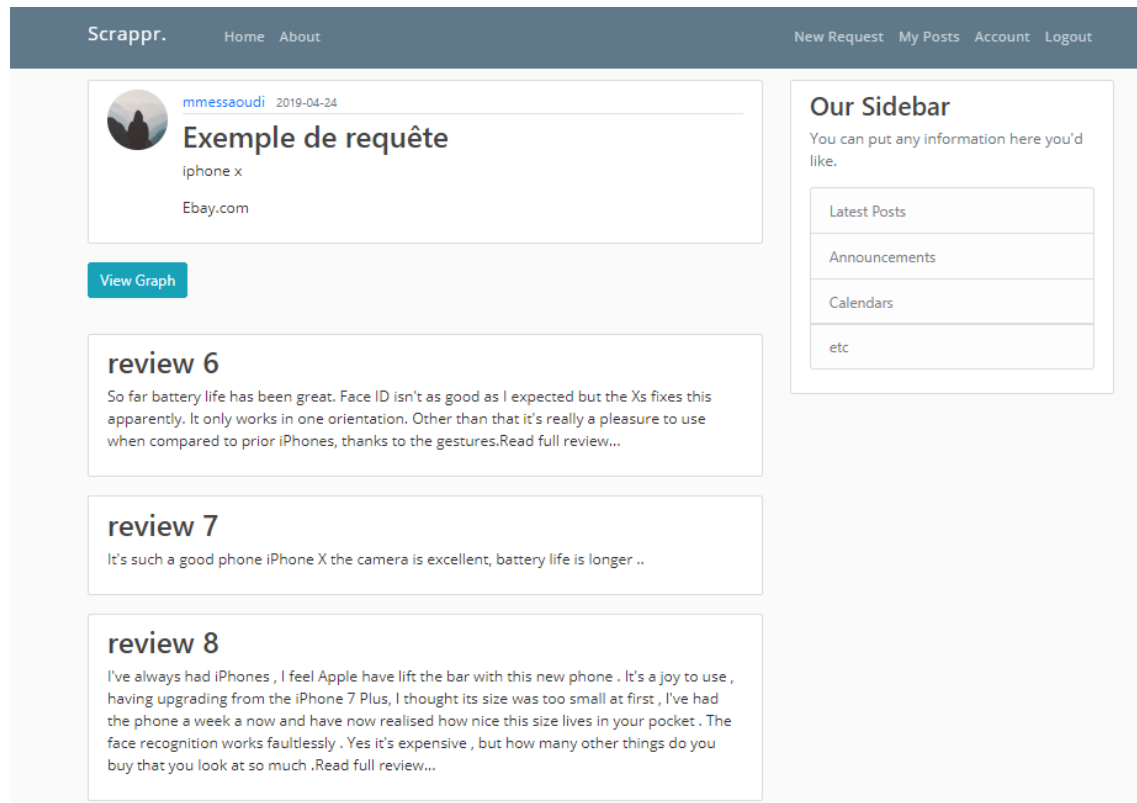


FIGURE 4.16 – Un échantillon des commentaires récupérés

4.3 Conclusion

Dans ce chapitre, nous avons exposé la réalisation de notre travail. Nous avons explicité l'environnement du travail et les différentes entités de notre application tout en présentant des captures d'écran témoignant les différentes fonctionnalités de l'application.

Conclusion générale

Comme il est dit, "le besoin rend ingénieux" et puisque la quête du savoir est sans limite et que toute évolution sera usée par le temps, les hommes se sont consacrés à travers différentes circonstances à inventer et créer de nombreux types d'accessoires et d'applications logicielles bénéfiques pour eux-mêmes. Ces inventions ont toujours permis la facilitation la simplification et le perfectionnement des activités quotidiennes.

En ce qui concerne notre cas, nous étions intéressés précisément au domaine du marketing. Tout au long de ce projet, nous avons été amenés à implémenter une application dans le cadre du projet de conception et de développement. Nous sommes arrivés à mettre en œuvre une application qui permet de récupérer les commentaires des consommateurs à partir des réseaux sociaux et sites de commerce électronique. Ensuite, nous nous sommes basés sur le "Deep Learning" pour analyser ces informations.

Dans le présent rapport, nous avons détaillé les différentes étapes que nous avons suivies pour concevoir notre solution. Nous avons tout d'abord commencé par présenter le cadre général de notre travail et faire une étude sur les solutions existantes. Puis, nous avons spécifié les différentes fonctionnalités de l'application à développer. Ensuite, nous avons abordé la phase de conception afin d'établir l'architecture de l'application. Finalement, l'étape de réalisation, au cours de laquelle nous avons précisé les logiciels utilisés ainsi que les interfaces du produit final.

L'application réalisée au cours de ce projet présente tout de même certaines insuffisances en raison des problèmes et obstacles confrontés au niveau développement. En effet, nous avons eu des problèmes concernant les méthodes pour récupérer un URL donné, ce qui nous a causé une perte de temps au début et un nombre de sites limités offerts par notre application.

Bibliographie

[B1] : Miguel Grinberg : Flask Web Development : Developing Web Applications with Python.

[B2] : Nicholas Hunt-Walker : An introduction to the Flask Python web app framework.

Netographie

[N1] : <https://flask-admin.readthedocs.io/en/latest/> (dernière visite 11/04/2019)

[N2] : https://textblob.readthedocs.io/en/dev/api_reference.html (dernière visite 14/04/2019)

[N3] : https://www.youtube.com/watch?v=eFdPGpny_hY (dernière visite 20/04/2019)

[N4] : <http://pygal.org/en/stable/> (dernière visite 22/03/2019)

[N5] : <https://www.youtube.com/watch?v=o-vsdfCBpsUt=82s> (dernière visite 23/04/2019)

[N6] : https://en.wikipedia.org/wiki/Naive_Bayes_classifier (dernière visite 10/04/2019)

[N7] : <https://www.nltk.org/> (dernière visite 16/04/2019)

[N8] : <https://stackabuse.com/accessing-the-twitter-api-with-python/> (dernière visite 09/04/2019)

Résumé

Notre travail est proposé dans le cadre du projet de conception et de développement pour les élèves ingénieurs de l'Ecole Nationale des Sciences de l'Informatique (ENSI). Notre tâche avait pour objectif d'implémenter une application qui sert à récupérer et analyser les données demandées par le client. En s'adaptant sur un modèle d'intelligence pour l'analyse de sentiment des commentaires, nous avons pu représenter les taux de satisfaction des consommateurs sur un tel produit.

Mots clés : récupérer,analyser,sentiment.

Abstract

This project is carried out as a second year design and development project at the National School of Computer Science. Our task was to implement a web application that is set to extract and analyse data requested by the client. By adapting to an intelligence model for sentiment analysis, we were able to represent consumer satisfaction rates on such a product. This report details the various theoretical studies and applied steps we have taken to design and implement this application.

Key words : extract,analyse,sentiment.

تلخيص

يُندرج هذا العمل ضمن مشروع التصميم والبرمجة للطلبة المهندسين المرسمين بالسنة الثانية في المدرسة الوطنية لعلوم الاعلامية.

و يتمثل مشروعنا في استعمال آليات التنقيب في البيانات و استخراج معلومات من المواقع الاجتماعية و منتديات النقاش اعتمادا على معايير مختلفة ثم دراستها اعتمادا على نموذج من الذكاء الاصطناعي و تمثيل معدلات رضا المستهلك على نموذج معين.

الكلمات الرئيسية :

استخراج، دراسة، التنقيب في البيانات، الذكاء الاصطناعي.
