

# Mémoire de Projet de fin d'études En vue d'obtention du diplôme licence

Licence en Sciences Mathématiques  
& Informatique

## La reconnaissance des modèles dans les textes en arabe (Pattern matching)

### Réalisé par :

- Hicham FADLI
- Soufiane MOUHTARAM
- Abdelhakim ZENIBI

### Devant le Jury composé de :

- Pr. A. TRAGHA      Encadrant
- Pr. A.ETTAOUIK      Examinateur
- Pr. O.ZAHOUR      Examinateur

# **Remerciements**

*On commencerait, avant tout, ce discours en faisant au Dieu vivant, qui nous a donné la force et la patience nécessaire l'accomplissement de ce travail, des remerciements solennels.*

*Nous souhaitons adresser nos remerciements les plus sincères aux personnes qui nous ont apporté leur aide et qui ont contribué à l'élaboration de ce mémoire ainsi qu'à la réussite de cette année universitaire.*

*La réalisation de ce projet est notre premier pas vers le milieu professionnel, c'est un changement du milieu théorique au milieu pratique et conceptuel.*

*Pour cela on tient en premier lieu à remercier la direction de notre Faculté des sciences Ben M'sik de nous avoir donné la possibilité de vivre cette merveilleuse expérience.*

*Nous exprimons notre gratitude aux membres du jury, qui nous ont honorés en acceptant de juger ce modeste travail et en remercie l'ensemble du corps enseignant de la Faculté des Sciences et particulièrement les enseignants de la Licence Science Mathématique Informatique.*

*Nous tenons à remercier sincèrement **Mr. Abderrahim TRAGHA**, qui, en tant qu'encadrant pédagogique, s'est toujours montré à l'écoute et très disponible tout au long de la réalisation de ce projet, ainsi pour l'inspiration, l'aide et le temps qu'il a bien voulu nous consacrer et sans qui ce projet n'aurait jamais vu le jour.*

*Nous remercions aussi nos parents pour leur contribution, leur soutien et leur patience, et aussi nos proches et nos amis qui nous ont toujours encouragé et soutenu au cours de la réalisation de notre projet.*

*En bref, nous tenons à dire MERCI à tous ceux qui ont contribué de près ou de loin à la réussite de notre projet*

## Résumé

Dans le *Traitemen*t Automatique des Langues Naturelles, il existe de nombreuses applications pour rechercher des informations dans des textes pour différentes langues du monde, mais pour l'arabe c'est rare ; en raison de la difficulté de son traitement et ses particularités.

Notre projet de fin d'étude est destiné à résoudre le problème de la recherche d'informations dans les textes arabes. Nous avons développé un site web en se basant sur un algorithme obtenu à partir de **KMP-Algorithm**. C'est une application de la concordance des modèles (Pattern Matching) aux textes écrits en arabe. Cet algorithme cherche les chaînes qui concordent avec un modèle donné. Cela dans les textes arabes avec ou sans symboles diacritiques.

L'objectif de notre site web est permettre l'utilisateur de chercher facilement des informations dans des textes arabes ainsi obtenir le nombre d'occurrence d'information et les localiser dans le texte.

Ce rapport présente les différentes étapes franchies pour aboutir à la réalisation et la mise en place de notre solution.

## تألخيص

في معالجة اللغة الطبيعية، هناك العديد من التطبيقات للبحث عن المعلومات في نصوص اللغات مختلفة في العالم، ولكن نادراً ما يحدث في اللغة العربية. نظراً لصعوبة علاجه وخصوصياته. يهدف مشروع نهاية الدراسة لدينا إلى حل مشكلة استرجاع المعلومات في النصوص العربية. لقد قمنا بتطوير موقع على شبكة الإنترنت بناءً على خوارزمية تم الحصول عليها من تلك الخاصة بـ **KMP** وعلى مطابقة الأنماط (مطابقة الأنماط). تبحث هذه الخوارزمية عن سلاسل تطابق نمطاً معيناً. هذا في النصوص العربية مع أو بدون رموز التشكيل.

الهدف من الموقع هو السماح للمستخدم بالبحث بسهولة عن المعلومات في النصوص العربية بالإضافة إلى حساب عدد تكرارات المعلومات وتحديد مكانها في النص.

يعرض هذا التقرير الخطوات المختلفة التي تم اتخاذها لتحقيق وإنجاز مشروعنا.

# Liste des figures

Figure 1: Alphabet arabe et les différentes formes d'écriture.....	14
Figure 2: Planning de projet .....	22
Figure 3 : Applications of Pattern Matching.....	25
Figure 4 : Algorithme Naïve brute force.....	27
Figure 5 :Algorithme 1Calcul-π.....	28
Figure 6: Algorithme Knuth-Morris-Pratt (KMP) .....	29
Figure 7 :Représentation graphique du KMP .....	29
Figure 8 :Programmation d'algorithme KMP .....	30
Figure 9: Algorithme Boyer Moor.....	31
Figure 10 : logo UML .....	33
Figure 11 : logo StarUML.....	34
Figure 12 : l'interface de StarUML.....	35
Figure 13 : Diagramme général de cas d'utilisation.....	36
Figure 14 : Diagramme général de classe.....	37
Figure 15 : Java SE .....	38
Figure 16 : logo Java EE .....	40
Figure 17 : logo HTML.....	41
Figure 18 :logo CSS .....	42
Figure 19 : logo JavaScript .....	42
Figure 20 : logo visual studio code .....	43
Figure 21 : Apache Tomcat.....	44
Figure 22 : logo Adobe Photoshop .....	44
Figure 23 : logo eclipse .....	45
Figure 24 : logo PhpMyAdmin .....	45
Figure 25 : logoWampServer.....	46
Figure 26 :Interface graphique .....	51
Figure 27 :Interface graphique lors de la saisie.....	52
Figure 28 :Interface après la recherche.....	53
Figure 29 : Interface graphique -fichier- .....	54
Figure 30 :Interface graphique lors de la saisie -fichier .....	55
Figure 31 :Interface graphique après la recherche -fichier .....	56
Figure 32 :Capture d'écran de la page recherche .....	57
Figure 33 :capture d'écran apres la recherche.....	58
Figure 34 :capture d'écran de la page de recherche par un fichier .....	59
Figure 35 : capture d'écran de la page de recherche par un fichier après la recherche.....	59
Figure 36 :Capture d'écran de page de CONTACT.....	60

## Liste des tableaux

Tableau 1: Variation de la lettre ئ « Ayn ».....	13
Tableau 2 : Chiffres arabes avec leur correspondance en chiffres latins.....	14
Tableau 3: Liste des préfixes verbaux les plus fréquents en arabe.....	17
Tableau 4: Liste des suffixes les plus fréquents en arabe .....	17
Tableau 5: Planning de projet .....	21
Tableau 6 : Complexity of algorithm pattern matching .....	32
Tableau 7 : Quatre cas de recherche.....	48
Tableau 8 : : Comparaison entre un mot contenant un tashkeel et un mot sans tashkeel [نجح و نَجَح].	49

## Liste des abréviations

TALN	Traitement Automatique Langage Naturel
NLP	Natural Language processing
KMP	Knuth Morris Pratt
UML	Unified Modeling Language
TALA	Traitement Automatique Langage Arabe
BF	Brute force
BM	Boyer Moore

# Table des matières

Remerciements .....	2
Résumé .....	3
الخاتمة .....	4
Liste des figures .....	5
Liste des tableaux .....	6
Liste des abréviations .....	7
Table des matières .....	8
Introduction .....	11
<b>Chapitre I : la langue arabe .....</b>	<b>13</b>
1. Caractéristiques de la langue arabe : .....	13
1.1    Les classes des mots .....	15
1.1.1    Les verbes .....	15
1.1.2    Les noms .....	16
1.1.3    Les préfixes .....	16
1.1.4    Les suffixes .....	17
2.    Les difficultés du traitement automatique de la langue arabe .....	17
2.1    L'absence de voyelles .....	17
2.2    Signes diacritiques des schèmes .....	19
3.    Problématique : .....	19
4.    Solution proposé .....	20
5.    Méthodologie de travail .....	20
5.1    Gantt Project .....	21
5.2    Planning .....	21
<b>Chapitre II : La reconnaissance de modules .....</b>	<b>23</b>
1.    Definition .....	23
2.    L'utilisation du pattern matching en intelligent artificiel .....	24
3.    Principe de pattern matching .....	24
3.1    Méthodologie de pattern matching .....	25
4.    Types de Pattern Matching Algorithms: .....	26
4.1    Naïve brute force algorithm .....	26
4.2    Knuth-Morris-Pratt (KMP) Algorithm: .....	27

4.3	Boyer-Moore (BM) Algorithm :	31
5.	Compléxité des algorithmes pattern matching:	32
<b>Chapitre III : Analyse et conception.....</b>		<b>33</b>
1.	Introduction.....	33
2.	Presentation UML.....	33
3.	Environnement de travail.....	34
4.	Identification des acteurs.....	35
5.	Diagramme de casd'utilisation .....	36
5.1	Diagramme de cas d'utilisation général.....	36
6.	Diagramme de classe.....	36
6.1	Diagramme général de classe.....	37
Conclusion .....		37
<b>Chapitre IV : réalisation.....</b>		<b>38</b>
1.	Introduction.....	38
2.	Les langages de programmation .....	38
2.1	Back End languages .....	38
2.1.1	Java SE .....	38
2.1.2	J2EE.....	39
2.2	Front end languages.....	41
2.2.1	HTML .....	41
2.2.2	CSS .....	42
2.2.3	JavaScript.....	42
3.	Environnement de développement .....	43
3.1	Visual Studio Code.....	43
3.2	Apache Tomcat.....	44
3.3	Adobe Photoshop.....	44
3.4	Plateforme de développement IDE Eclipse .....	45
3.5	PhpMyAdmin .....	45
3.6	WampServer.....	46
4.	Outils matériels .....	47
5.	Problèmes rencontrés et leurs solutions.....	47
5.1	Problèmes techniques et leurs solution .....	48
6.	Les interfaces graphiques.....	50
6.1	Test d'algorithme .....	50
6.2	Interfaces graphiques commentées.....	57
6.2.1	Page de recherche .....	57

6.2.2	Page de recherche par fichier.....	58
Conclusion et perspectifs .....	61	
Bibliographie.....	62	

# Introduction

Depuis le siècle dernier, la science du traitement automatique des langues naturelles (TALN) représente la préoccupation principale de plusieurs chercheurs. Cette science pose plusieurs difficultés au cœur du traitement à mesure de la quantité massive des documents contenant des milliers, voire des milliards, d'informations accumulées, et difficile à gérer d'une manière pertinente. La Science du traitement automatique des langues naturelles (TALN), comme des langues parlées, lues et écrites par les êtres humains, et l'ensemble des langages informatique, artificiels, mathématiques ou logiques, nous permet de produire un système linguistique facile et intelligible. D'une manière plus précise, le TALN est la conception de logiciels capables de traiter de façon automatique des données exprimées dans une langue (dite « naturelle », par opposition aux langages formels de la logique mathématique). Il se constitue de plusieurs grands domaines d'étude qui partagent un seul objectif : le traitement d'une manière automatique des langues naturelles via l'utilisation des programmes et logiciels informatiques pour traiter des données (des corpus, des documents, des textes, des phrases, des mots, etc.) à la base des règles et grammaires linguistiques. Autrement dit, pour faire une certaine application du TALN, il faut déterminer ou créer un programme, un logiciel approprié à cette application et en parallèle de déterminer les règles grammaticales associées à l'application traitée. Parmi les applications les plus connues du TALN, nous pouvons citer : la traduction automatique, la correction orthographique, la recherche d'information et la fouille de textes, le résumé automatique, la génération automatique de textes, la synthèse de la parole, la reconnaissance vocale, la reconnaissance de l'écriture manuscrite, etc...

Le traitement automatique de la langue arabe (TALA) comme une langue naturelle, fait face à un certain nombre de difficultés, les plus manifestes, sont le problème de la voyéllation, l'agglutination et l'extraction de la racine. Dans le cadre du traitement automatique de la langue et de la recherche d'information (RI) arabe, les problèmes liés à l'indexation classique sont dus aux variations linguistiques de la langue utilisée, ces variations pour la langue arabe sont divisées en trois types :

- Des variations morphologiques.
- Des variations lexicales (on utilise pour le même sens des mots différents).
- Des variations sémantiques

Dans notre sujet on va aborder la reconnaissance de modèles par rapport à la langue arabe qui traité tous les cas possibles dans la recherche d'information surtout au niveau diacritique de la langue arabe.

Dans ce cadre nous avons réalisé un site web qui permet vérifier de l'existence de l'information dans un texte arabe et compte leur nombre d'occurrences ainsi il affiche leur position, visant à faciliter le processus de recherche pour les clients et tout cela en un seul clic.

Notre travail est décomposé de quatre chapitres, Chapitre 1 qui présentera le projet, Chapitre 2 donnera une vision sur la reconnaissance de modèles et présenter différents algorithmes de pattern matching, Chapitre 3 analysera et concevra le site Web et Chapitre 4 qui traitera la phase de la réalisation, les outils et l'environnement de développement.

# Chapitre I : la langue arabe

## 1. Caractéristiques de la langue arabe :

La langue arabe se compose de vingt-huit lettres de base (elle s'écrit et se lit de droite à gauche). Dans l'écriture arabe, il n'y a pas les notions de majuscule et minuscule. Les lettres arabes changent de forme de présentation selon leur position au sein des mots soit isolée, initiale, médiane ou finale (tableau 2). Dans la langue arabe, toutes les lettres se lient entre elles sauf « و », « ؤ », « ئ », « ؔ », « ؕ », « ؜ » et « ؑ » qui ne s'attachent pas à gauche. Donc, les caractères arabes n'ayant pas de forme initiale et médiane s'attachent au caractère qui précède, mais pas à celui qui suit.

Tableau I: Variation de la lettre ء « Ayn ».

À la fin d'une lettre non joignable	À la fin	Au milieu	Au début
ء	ء	ء	ء

Un mot arabe se compose de consonnes et de voyelles. Les voyelles se positionnent au-dessus ou au-dessous des lettres. Elles sont indispensables à la compréhension et à la lecture correcte d'un texte. Le monde arabe n'utilise les voyelles que pour des textes religieux ou didactiques (en particulier pour les écoles primaires). Généralement, les journaux et les livres ne comportent pas de voyelles. Dans ce projet, nous traitons « l'arabe moderne » ou « l'arabe standard ». Il s'agit de la langue écrite dans les journaux et les livres.

Phonétique	Lettre	Fin	Milieu	Début	Phonétique	Lettre	Fin	Milieu	Début
Elif	أ	أ	أ	أ	Dad	ض	ض	ض	ض
Bè	ب	ب	ب	ب	Ta	ط	ط	ط	ط
Tè	ت	ت	ت	ت	Za	ظ	ظ	ظ	ظ
Thè	ث	ث	ث	ث	'Ayn	ع	ع	ع	ع
Djim	ج	ج	ج	ج	Ghayn	غ	غ	غ	غ
Ha	ح	ح	ح	ح	Fè	ف	ف	ف	ف
Kha	خ	خ	خ	خ	Gaf	ق	ق	ق	ق
Dèl	د	د	د	د	Kèf	ك	ك	ك	ك
Dhèl	ذ	ذ	ذ	ذ	Lèm	ل	ل	ل	ل
Ra	ر	ر	ر	ر	Mim	م	م	م	م
Zay	ز	ز	ز	ز	Noun	ن	ن	ن	ن
Sin	س	س	س	س	Hè	ه	ه	ه	ه
Shin	ش	ش	ش	ش	Waw	و	و	و	و
Sad	ص	ص	ص	ص	Yè	ي	ي	ي	ي

Figure 1: Alphabet arabe et les différentes formes d'écriture

- ❖ **Encodage:** L'encodage principal des caractères arabes est l'ISO-8859-6 (sous Windows il est appelé : Windows-1256). De plus, l'arabe est supporté aussi par l'encodage Unicode.
- ❖ **Chiffres:** Les chiffres utilisés sont les chiffres latins au Maghreb et les chiffres «Arabo-indiens » au Moyen-Orient. Les chiffres s'écrivent de gauche à droite.

Tableau 2 : Chiffres arabes avec leur correspondance en chiffres latins.

0	1	2	3	4	5	6	7	8	9
٠	١	٢	٣	٤	٥	٦	٧	٨	٩

- ❖ **Sens d'écriture :** La lecture et l'écriture d'un mot arabe se font de droite vers la gauche.
- ❖ **Marques diacritiques :** En arabe, l'écriture des marques diacritiques se fait en même temps avec l'écriture des lettres ou à la fin du mot.
- ❖ **Schèmes :** Le schème, appelé aussi pattern, est un mot composé de trois lettres radicales (f, ق), (E, ع) et (l, ل). Le schème occupe une place importante dans le processus de dérivation des mots arabes.

- ❖ **Ponctuation** : Les signes de ponctuation arabe sont identiques à ceux utilisés dans les langues européennes, mais sont renversés, exemple :
  - ✓ **La virgule renversée " ، "**
  - ✓ **Le point-virgule renversé " ؟ "**
  - ✓ **Le point d'interrogation inversé " ؟؟".**

## 1.1 Les classes des mots

Avant de commencer le traitement automatique de la langue arabe, il faut tenir compte de la classification des unités lexicales de la langue arabe (nom, verbe, pronom...) sur laquelle doit se baser le traitement.

### 1.1.1 Les verbes

La majorité de mots arabes, dérivent d'un verbe de trois consonnes qui représente lui-même une racine d'un groupe de mots. Par conséquent, le mot en arabe se déduit à partir de la racine en rajoutant des suffixes, des préfixes ou les deux en même temps.

En arabe, la conjugaison des verbes dépend de facteurs suivants :

- ❖ **Le temps** : accompli (passé) ou inaccompli (présent).
- ❖ **Le nombre du sujet** : singulier, duel, pluriel.
- ❖ **Le genre** : masculin ou féminin.
- ❖ **La personne (première, deuxième, etc.).**
- ❖ **La voix** : active ou passive.

Voici un exemple qui explique l'effet de ces différents facteurs :

Dans cet exemple, on prend la combinaison de ces trois consonnes (ك + ت + ب) « k + t + b » donne le verbe كتب « écrire ».

Les trois lettres **K, T, B**, on les trouvera dans tous les mots qui dérivent de cette racine, en rajoutant des préfixes, des suffixes ou les deux à la racine.

La conjugaison des verbes arabes dépend de plusieurs facteurs, parmi ces facteurs **le temps**. Par contre, la notion du temps est différente de celle du français, il se divise en deux temps : **l'accompli** et **l'inaccompli**. L'inaccompli recouvre le présent et le futur et l'accompli représente le passé.

- ❖ **L'accompli** : Il désigne le passé et le verbe conjugué se distingue par des suffixes. Dans notre cas, la conjugaison du verbe كتب « écrire » avec le pluriel féminin donne كتبن\_KaTaBna, « elles ont écrit » et avec le pluriel masculin donne كتبوا\_KaTaBuu, «

ils ont écrit ».

- ❖ **L'inaccompli présent :** Dans ce cas, l'action est en cours d'accomplissement. Les verbes conjugués se distinguent par les préfixes. Comme exemple, la conjugaison du verbe \_كتب\_ « écrire » au masculin singulier donne \_يكتب\_yaKTuBu, « il écrit » et avec le féminin singulier donne \_تكتب\_taKTuBu, « elle écrit ».
- ❖ **L'inaccompli futur :** Dans ce cas, l'action se déroulera au futur et elle est marquée par l'antéposition de مس sa ou سوف\_sawfa au verbe. En ajoutant l'antéposition مس sa à notre exemple, on obtient \_سيكتب\_sayaKTuBu , « il écrira » et en ajoutant l'antéposition سوف\_sawfa, on obtient \_سوفكتب\_sawfayaKTubu « il va écrire ».

## 1.1.2 Les noms

En arabe, le nom se divise en deux catégories, ceux qui sont dérivés d'une racine verbale et ceux qui ne le sont pas, tel est le cas des noms propres, des noms étrangers...

La déclinaison des noms obéit à plusieurs règles :

- ❖ **Le féminin singulier :** dans la plupart des cas, on ajoute la lettre ة t pour obtenir le nom au féminin singulier (exemple : صغير « petit » devient صغيرة « petite »).
- ❖ **Le féminin pluriel :** généralement, on rajoute les deux lettres ات (exemple : عامل «ouvrier » devient عاملات « ouvrières »).
- ❖ **Le masculin pluriel :** généralement, on rajoute les deux lettres ون ou les deux lettres بن qui dépendent de la position du mot dans la phrase (sujet ou complément d'objet).
- ❖ **Le pluriel irrégulier :** c'est le cas le plus complexe en arabe, pour transformer un mot au pluriel, on doit insérer des lettres au début, au milieu ou à la fin du mot, (exemple : قفل « serrure » devient أقفال « serrures »).

Le problème du pluriel irrégulier dans l'arabe pose un défi à la morphologie, non seulement à cause de sa nature non concaténative, mais aussi parce que son analyse dépend de la structure de chaque mot comme pour les verbes irréguliers.

## 1.1.3 Les préfixes

En arabe, les préfixes présentent les morphèmes correspondant aux lettres qui se situent en début de mot. Ils servent à indiquer la personne de la conjugaison des verbes. Le tableau suivant illustre les préfixes verbaux les plus fréquents en arabe.

Tableau 3: Liste des préfixes verbaux les plus fréquents en arabe

Lettre	Rôle
أ	Indique la première personne au singulier (je)
ن	Indique la première personne au pluriel (nous)
ت	Indique la deuxième personne féminine, masculine, singulière et duelle
ي	Indique la troisième personne masculine au singulier, duel, pluriel, masculin et féminin pluriel.

#### 1.1.4 Les suffixes

En arabe, les suffixes présentent essentiellement les terminaisons des conjugaisons verbales, ainsi que les marques du pluriel et du féminin pour les noms. On ne trouve jamais une combinaison entre deux suffixes. Le tableau suivant illustre les suffixes les plus fréquents en arabe.

Tableau 4: Liste des suffixes les plus fréquents en arabe

ات	وه	ته	هم	ية	ين	ة	ا
وا	ان	تم	هن	تك	يه	ه	هـ
ون	تي	كم	ها	نا	ية	ي	ـيـ

## 2. Les difficultés du traitement automatique de la langue arabe

### 2.1 L'absence de voyelles

L'absence de voyelles en langue arabe présente un problème majeur, et qui reproduit des différents phénomènes morphosyntaxique et sémantique. L'arabe se caractérise par quatre types de voyelles et qui présentent une difficulté au niveau du traitement par le système :

- **Les voyelles brèves ou courtes (nommées en arabe par : القصيرة الاصوات /AlAssawatte El kassira).** Ce type de voyelles se prononce brièvement et s'écrit sous la forme de quatre petits signes, et qu'on met au-dessous ou au-dessus de la lettre :
  - ❖ La Fat- ha (ـ : -a), qui est située au-dessus de la lettre. Où chaque lettre

associée par ce signe se prononce par sa tonalité, suivi par un ton « -a ».

- ❖ **La Dhamma** (ဋ္ဌ : -où), un petit 'و/Waw' situé au-dessus de la lettre. Telle que chaque lettre liée par ce signe se prononce par sa tonalité, suivi par un ton « -ou ».
  - ❖ **Le signe Kas-ra** (ဃ : -i), située au-dessous de la lettre. Où chaque lettre associée par ce signe se prononce par sa tonalité, suivi par un ton « -i »
  - ❖ **Le signe Soukonn** (ၢ). Ce type est sous la forme d'un petit cercle, et qui peut être situé au-dessus de la lettre. Il indique l'absence de voyelle ; implique que la consonne sera prononcée seule et sans voyelle.

Les voyelles brèves peuvent lier les lettres d'un mot arabe, mais restent indépendantes de ces lettres, c'est à dire que les signes ne sont pas collés par les lettres qui forment le mot arabe. Et par conséquent, cette caractéristique présente un problème lors du traitement des mots qui comportent ce type de voyelles.

Sauf, les textes religieux (**Alcoran** et **Alhadith**) et une partie de la poésie et des textes scolaires, une quantité de documents arabes très importante (dans les livres et sur les sites internet) est enregistrée sans signes. De même que parmi la quantité textuelle voyelle, on trouve des documents entièrement voyelle et autre partiellement.

Dans les détails de, nous pouvons proposer que : « l'écriture arabe courante ne note pas les voyelles brèves, la gémination des consonnes, les marques casuelles composées d'une voyelle brève suivie, pour les noms et les adjectifs indéterminés, d'une consonne."ڻ" (noumtanwîn), etc. on parle alors d'écriture ‘non voyelle’. Ces signes de voyéllation qui sont réalisés, lorsqu'ils sont notés, sous la forme des signes diacritiques placés au-dessus ou au-dessous des lettres, apparaissent dans certains textes religieux (**coran** ou **hadith**) ou littéraires (poésie classique, notamment) : on dira qu'ils sont édités en graphie voyelle. On distingue en outre deux pratiques, celle de l'écriture entièrement voyelle et celle qui l'est partiellement.

La voyéllation partielle répond, dans les éditions soignées, à la levée de certaines ambiguïtés de première lecture [...]. On notera que la voyéllation partielle ne repose pas sur une codification appuyée sur une tradition : elle ne présente donc pas un caractère systématique».

Les voyelles longues (de prolongation) : elles se prononcent de manière prolongée. Chaque type de voyelles longues correspond à un autre type de voyelles brèves au niveau de la prononciation. Elles ne sont pas comme les voyelles brèves et qui sont sous la forme des signes, mais elles sont représentées par des lettres et s'insèrent dans le mot exactement comme les consonnes.

En arabe nous pouvons distinguer trois sortes de voyelles de prolongation :

- Le « Alif » (ا : qui prolonge la voyelle brève /a/).
  - Le « Ya » (ي : qui prolonge la voyelle brève /i/).
  - Le « Waw » (و : qui prolonge la voyelle brève /ou/).

**Double voyelle.** Ce type comporte deux signes qui permettent de redoubler la consonne (le cas de la Shad-da ( ﻭ ) ou bien de créer une tonalité à la fin du mot associé (le cas du Tanwine).

**Shad-da ( ﻭ )** : un signe qui est situé au-dessus de la lettre et qui peut être associé par un certain signe des voyelles brèves.

**Tanwine.** Ce type s'écrit sous la forme suivante :

- Avec 2 Fat- ha situées au-dessus de la dernière lettre " ٍ " « Alif ».
- Avec 2 Dham-ma situées au-dessus de la dernière lettre.
- Avec 2 Kasra situées au-dessous de la dernière lettre.

## 2.2 Signes diacritiques des schèmes

Comme dit précédemment, un diacritique est un élément ajouté à une lettre d'un alphabet pour en modifier la valeur. Cet élément peut être souscrit (en indice), suscrit (en exposant) à cette lettre, à sa droite ou encore à sa gauche. L'absence de ces signes peut également affecter le sens de certains schèmes arabes.

En outre, un signe diacritique peut être placé au-dessous ou au-dessus de la consonne d'un schème arabe pour donner un sens ou une valeur spéciale aux mots associés par ce schème. Ce phénomène présente aussi, un problème au cas de la pratique par un système du traitement.

### Exemple :

Prenons dans cet exemple les mots qui ont des schèmes « فعل » et qui peuvent être : « فعل /fiaale», «/ fouaale» ou «فَعَال/faâale», où la première consonne du schème soit avec «kassra » (signe ( ﴿ ), placé au-dessous du caractère arabe): et qui peut signifier un certain sens.

Ou avec «Damma»(signe ( ۚ ), placé au-dessus du caractère arabe) : dans ce cas le schème a un autre sens différent. Ou bien avec fat-ha (signe ( ـ ) et la deuxième consonne associée par le signe (shadda ( ﻭ )) : ce qu'implique dans ce cas, que le schème peut signifier un sens différent par rapport aux cas précédents.

De manière que les voyelles en arabe jouent le rôle tel qu'elles enlèvent l'ambiguïté, elles donnent aussi l'étiquette grammaticale indépendamment de sa position dans la phrase. Cet effet permet de diminuer d'une manière relative la complexité morphologique de la langue arabe.

## 3. Problématique :

Considère la reconnaissance des modules (modèles de concordance) est parmi les applications les plus connues du TALN qui recherche des mots et des phrases dans le texte

afin qu'il supporte la plupart des langues (ex. les langues : française, anglaise, chinoise, italien etc.). En raison de la simplicité de leur ligne et de la facilité leur traitement.

Quant à la langue arabe, elle est confrontée une difficulté croissante lors du passage du niveau morphologique aux autres niveaux d'analyse (syntaxique, sémantique et pragmatique). La langue arabe est parmi les langues sémitiques vivantes qui s'écrivent généralement de droite à gauche et qui est difficile à traiter par la machine elle est également considérée comme une langue des objets et des propriétés, tels que les mots peuvent être associés par leurs caractéristiques, citons dans ce cas par une approche de gènes à partir de l'identification de types des mots. Morphologiquement, le mot arabe se caractérise par des structures très compliquées, où on peut trouver une pluralité verbale, une polysémie du mot et similitude verbale et syllabique. Pour cela l'arabe est considéré comme : « Une langue difficile à maîtriser dans le domaine du traitement automatique des langues».

Grâce à ses propriétés morphologiques et syntaxiques. A la différence des autres langues comme, le français ou l'anglais, dont les étiquettes grammaticales proviennent d'une approche distributionnelle caractérisée par une volonté "d'écartez toute considération relative au sens", les étiquettes de l'arabe viennent d'une approche où la sémantique côtoie le formel lié à la morphologie du mot, sans référence à la position de ce dernier dans la phrase. Ce phénomène est matérialisé par la notion de schèmes et de fonctions qui occupent une place importante dans la grammaire de l'arabe ».

Plusieurs niveaux d'analyses peuvent être consacrés pour la tâche du **TALN**. On spécifie dans ce cas l'analyse morphologique, syntaxique, sémantique et pragmatique

## 4. Solution proposé

L'arabe pose un certain nombre de problèmes en termes de traitement et d'analyse à travers des algorithmes de concordance de modèle de sorte que la plupart d'entre eux ne supportent pas l'arabe.

Pour résoudre le problème de la concordance des modèles dans la langue arabe, nous avons recherché des algorithmes de concordance des modèles par rapport à d'autres langues et les avons comparés entre eux pour choisir le meilleur algorithme en termes de complexité et de temps d'exécution, puis modifiés pour être capable de chercher des informations dans des textes arabe.

## 5. Méthodologie de travail

La méthode agile est une façon de gérer les projets, elle facilite le dialogue entre toutes les parties prenantes, clients, utilisateurs, développeurs et autres professionnels du projet. La planification consiste à prévoir le déroulement du projet tout au long des phases constituant le cycle de vie prévu. Pour cela nous avons utilisé **le diagramme de Gantt** (Environnement MS Project) comme outils de communication pour représenter l'avancement, dans le temps, du projet.

## 5.1 Gantt Project

**Gantt Project** est une initiative open source dont le but est de créer un programme capable de produire des graphiques représentant la distribution des tâches d'un projet à long, moyen et court terme. Ce logiciel permet de distribuer les activités par personne ou par "ressources". Il peut aussi être utilisé pour gérer des projets de groupe.

## 5.2 Planning

Tableau 5: Planning de projet

Numéro de tache	Nom de tache	Durée (Jours)	Début	Fin
1	Cadrage de projet	15	12/03/2022	27/03/2022
2	Recherche sur sujet pattern matching et NLP	10	01/04/2022	11/04/2022
3	Recherche d'algorithme de Pattern matching	9	12/04/2022	21/04/2022
4	Comparaison entre les algorithmes	6	19/04/2022	25/04/2022
5	Choisissez et modifiez le meilleur algorithme parmi les algorithmes que nous avons trouvés	4	26/04/2022	30/04/2022
6	Développement des interfaces java pour le site compagnon en vue de faciliter la saisie	15	02/04/2022	17/04/2022
7	Conception du site web	10	18/04/2022	03/05/2022
8	Créé un site web	20	05/05/2022	25/05/2022
9	Rédaction documentation	9	11/06/2022	20/06/2022

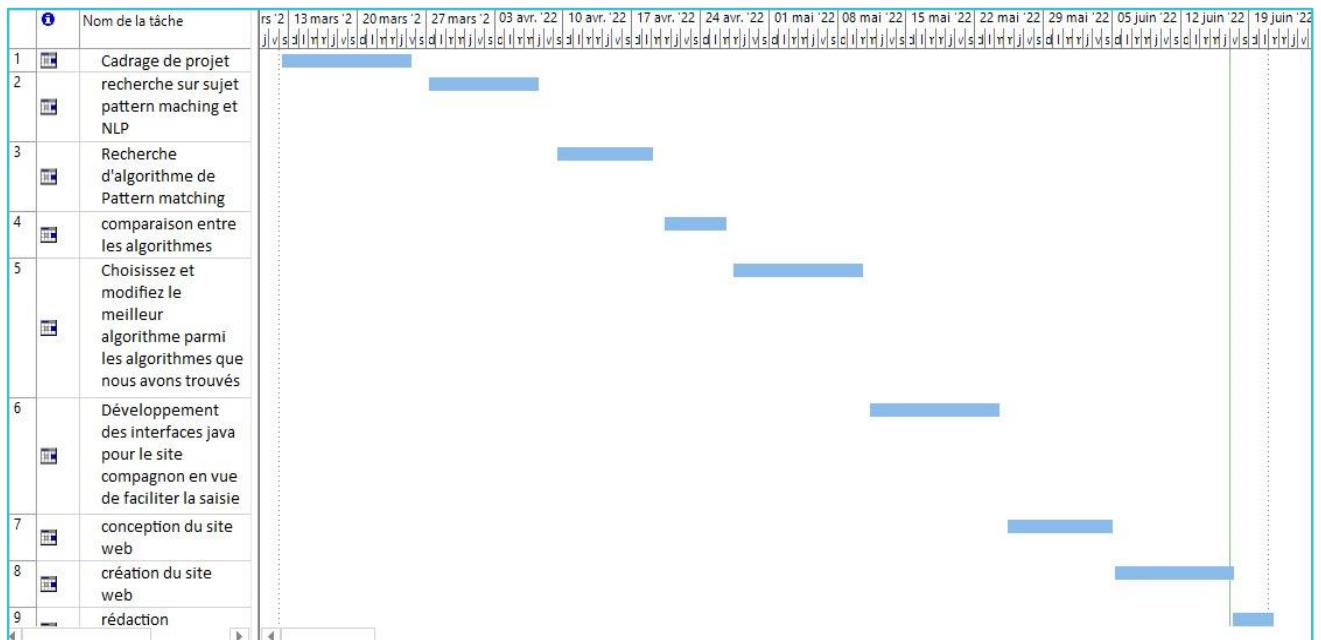


Figure 2: Planning de projet

# Chapitre II : La reconnaissance de modules

## 1. Definition

En informatique, la reconnaissance de modèles est l'action de vérifier la présence des constituants d'un motif dans une séquence donnée d'éléments verbaux. Contrairement à la reconnaissance des formes, la concordance doit généralement être exacte : "soit il y aura concordance, soit il n'y en aura pas". Les motifs ont généralement la forme de séquences ou de structures arborescentes. Les utilisations de la mise en concordance de motifs comprennent l'affichage des emplacements (le cas échéant) d'un motif dans une séquence de mots, l'affichage d'un composant du motif correspondant et la substitution du motif correspondant par une autre séquence de mots (c'est-à-dire la recherche et le remplacement).

Les motifs de séquences (par exemple, une chaîne de texte) sont souvent décrits à l'aide d'expressions régulières et mis en concordance à l'aide de techniques telles que le retour en arrière.

Les modèles d'arbres sont utilisés dans certains langages de programmation comme un outil général pour traiter les données en fonction de leur structure. Par exemple, C#, F#, Haskell, ML, Python, Ruby, Rust, Scala, Swift et le langage de mathématiques symboliques Mathematica ont une syntaxe spéciale pour exprimer les modèles d'arbres et une construction de langage pour l'exécution conditionnelle et la récupération de valeurs basée sur ces modèles.

Il est souvent possible de donner des motifs alternatifs qui sont essayés un par un, ce qui donne une construction de programmation conditionnelle puissante. La concordance de motifs inclut parfois la prise en charge des gardes.

La concordance de modèle est le processus qui consiste à vérifier si une séquence spécifique de caractères, de jetons ou de données existe parmi des données.

Les langages de programmation réguliers utilisent des expressions régulières pour la comparaison de motifs.

Le pattern matching est utilisé pour déterminer si les fichiers sources des langages de haut niveau sont syntaxiquement corrects. Il est également utilisé pour trouver et remplacer un motif correspondant dans un texte ou un code par un autre texte/code. Toute application qui prend en charge la fonctionnalité de recherche utilise le filtrage d'une manière ou d'une autre.

## 2. L'utilisation du pattern matching en intelligente artificiel

Le pattern matching, comme son nom l'indique, est une méthode utilisée par une entité d'intelligence artificielle pour comprendre une certaine chaîne de données et y trouver des séquences de motifs. Le filtrage est un concept fondamental pour toute opération de classification ou de reconnaissance. Qu'il s'agisse de la reconnaissance des visages, de la reconnaissance vocale ou même des moteurs de synthèse vocale, la concordance des formes est le principal concept utilisé. Avec l'IA, l'utilisation de langages de programmation de haut niveau et de compilateurs facilite grandement la mise en œuvre de ce concept. Le filtrage de motifs est utilisé dans les applications d'analyse des risques et de détection des fraudes. Associé à l'IA, le filtrage de modèles crée une entité puissante capable de déterminer rapidement les situations ayant des intentions malveillantes. De plus, les systèmes d'IA ont la capacité de s'adapter constamment aux nouveaux changements et aux menaces changeantes, fournissant ainsi des solutions efficaces.

## 3. Principe de pattern matching

String Pattern Matching peut être utilisé pour la recherche d'un mot dans un texte long, avec déterminer (combien de fois soit répéter ? ou bien à quelle position ?) on peut pratiquer ça dans le saint coran (القرآن الكريم). Si nous voulons savoir combien de fois va répéter mot “الله” dans coran alors ici pattern Matching va les utilisés Pour obtenir la concordance exacte tels que **KMP**, Les algorithmes de **Boyer-Moore**, **Rabin-Karp** et **Naïve brute force** sont utilisés.

L'objet de Pattern matching est de trouver l'emplacement d'un modèle de texte spécifique dans un corps de texte plus large (par exemple, une phrase, un paragraphe, un livre, etc...).

### Besoin de Pattern Matching :

La concordance de modèles est le processus de vérification d'une séquence perçue de chaîne pour la présence des constituants d'un certain modèle. Le concept de concordance de motifs est utilisé dans de nombreuses applications, la figure suivante montre les différentes applications. Tel que

Moteur de recherche web parmi d'autres applications. Aujourd'hui, presque tout le monde utilise l'application Web pour obtenir les résultats souhaités. Mais il n'est pas nécessaire que les gens recherchent uniquement du texte à chaque fois. Ils peuvent vouloir différents types de données comme l'audio, Photo et vidéo. Pour gérer ce type de données, nous avons besoin d'une meilleure méthode de recherche.

La concordance de motifs aidera à trouver un résultat correct et approprié. Il y a beaucoup de algorithmes utilisés pour trouver des motifs correspondants.

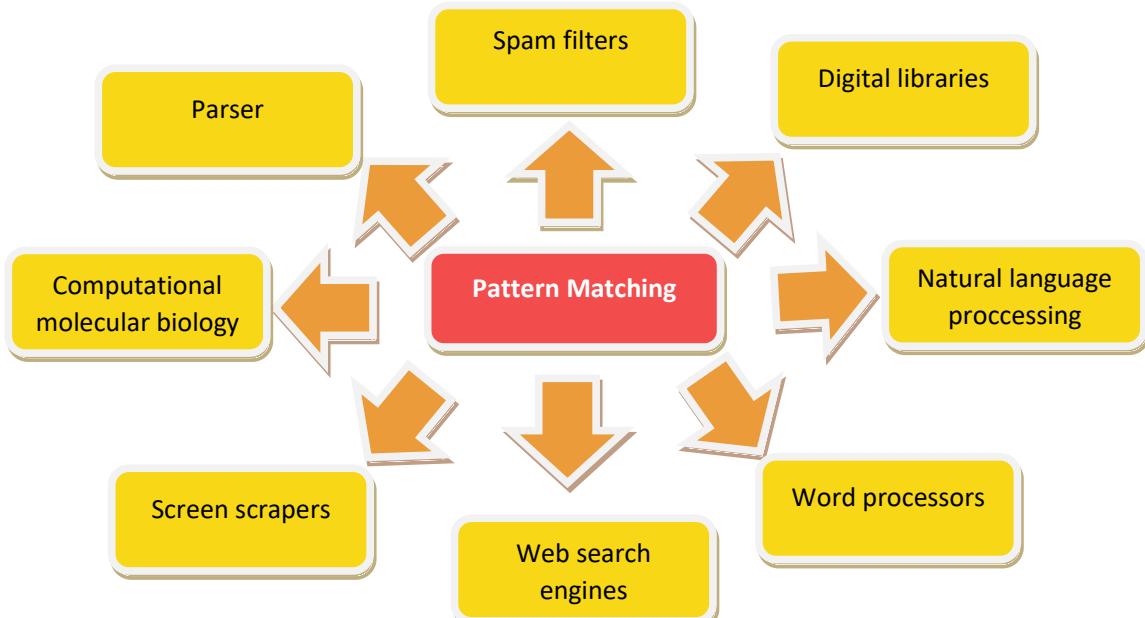


Figure 3 : Applications of Pattern Matching

### 3.1 Méthodologie de pattern matching

String pattern Matching est un sujet très important pour le texte systèmes de traitement. Comme le système contient un grand nombre de données dans le texte les algorithmes pattern Matching sont les principaux composants de tels systèmes. String Matching algorithm Se compose de 2 composants la séquence de chaîne (String) ou de texte et le modèle (pattern) à assortir. La chaîne est notée  $S = \{s_0, s_1, s_2, s_3, \dots, s_m\}$  qui contient  $m$  caractères et le modèle (pattern) est noté  $P = \{p_0, p_1, p_2, \dots, p_n\}$  contient  $n$  nombre de caractères qu'il contient. Ainsi, les caractères en pattern sont moins que les caractères présents dans la chaîne ou la séquence de texte.

Le modèle (pattern) est scanné à travers la chaîne ou la séquence de texte en utilisant la taille de la fenêtre (c'est-à-dire la taille du pattern). Si le pattern ne correspond pas, il est décalé du nombre spécifique dérivé des caractères du pattern et si le pattern obtient mis en concordance avec les caractères présents dans la séquence qu'il obtient décaler de la taille du pattern. C'est la fonctionnalité normale de string Matching algorithm.

## 4. Types de Pattern Matching Algorithms:

Dans le problème du pattern matching, il est question de trouver toutes les occurrences de tous les patterns dans un texte cible. Soit  $P = \{p_1, p_2, \dots, p_m\}$  l'ensemble des patterns et  $T = t_1, t_2, \dots, t_n$  le texte cible composé de  $N$  caractères.  $T$  et  $p_i$  sont composés de caractères d'un alphabet fixe  $\Sigma$ . Etant donnés  $P$  et  $T$ , l'algorithme doit localiser les positions de toutes les occurrences de n'importe quel pattern  $p_i$  dans le texte  $T$ . Ces algorithmes peuvent être classés en deux catégories selon le type de recherche utilisé. Il y a la recherche simple, dans ce cas un seul pattern est localisé à chaque passage. Il y a aussi la recherche multiple, qui consiste en la recherche de plusieurs patterns dans un seul balayage du texte cible. Plusieurs techniques sont utilisées dans ces algorithmes.

### 4.1 Naïve brute force algorithm

The naïve string Matching algorithm est également connu sous le nom de Brute force algorithm. C'est string Matching algorithm le plus simple algorithme. Cet algorithme commence à faire correspondre le modèle (pattern) et séquence des caractères de gauche à droite. Cet algorithme correspond à chaque caractère du pattern avec le correspondant caractère de chaîne en décalant le pattern avec 1.

L'algorithme s'arrête lorsque le pattern correspondant est trouvé avec succès dans la séquence ou leur se produit un décalage jusqu'à la fin de la séquence. Si le pattern est de longueur  $m$  et que le séquence de texte est de longueur  $n$  la complexité temporelle totale pour rechercher le pattern dans la séquence de texte est  $O(n*m)$ . Ainsi ce le processus prend du temps. La fonction de recherche de Brute la force est donnée ci-dessous.

#### Naïve string search $O(n+m)$

**ALGORITHM** BruteForceStringMatch( $T[0..n-1]$ ,  $P[0..m-1]$ )

**Input** : a pattern  $p = p_1 \dots p_m$  et a texte  $t = t_1 \dots t_n$

**Output**: l'indice de premier caractère dans le text commence Matching substring ou -1 si la la recherche a échoué

**Debut**

```
pour i<-- 0 à n-m
j<-- 0
tantque j < m et P[j]=T[i+j] faire
    j<-- j+1
    if (i==m )
```

```

return i
return -1
fin

```

Figure 4 : Algorithme Naïve brute force

Soit une chaîne de  $n$  caractères appelée texte et une chaîne de  $m$  Caractères ( $m = n$ ) appelés motif, trouver une sous-chaîne du texte qui concordent au pattern. Pour le dire plus précisément, nous voulons trouver  $i$  - l'index du caractère le plus à gauche de la première concordance Une sous-chaîne dans le texte.

Exemple:

1. Pattern ← 001011  
Text ← 100101011010011**00101111010**
2. Pattern ← السعادة  
Text ← لم يفت الأولان بعد للحصول على **السعادة**

## 4.2 Knuth-Morris-Pratt (KMP) Algorithm:

L'idée derrière l'algorithme KMP est, chaque fois que la non-concordance se produit entre le motif et les caractères de la séquence de texte nous décalons le motif du nombre de caractères correspondants précédemment lors de la traversée. Puisque nous connaissons les caractères de la séquence de texte avant la non-concordance, nous pouvons décaler le motif de manière à ce que le nouveau pattern Matching commence à partir du caractère d'index incompatible en lui incrémentant de 1 [2][3][4].

L'algorithme commence à faire correspondre le modèle avec la séquence de texte de gauche à droite. La valeur de décalage est obtenue à partir de valeur numérique des caractères du modèle. La valeur numérique est recherchée à l'aide du préfixe long correspondant aux caractères de la position actuelle des caractères. Dans le préfixe du caractère particulier et le suffixe du même caractère, certains caractères doivent être identiques. Cela augmente donc le décalage du motif et le temps de  $O(n^2)$  à  $O(n+m)$ .

L'idée derrière cela d'augmenter le décalage de plus de 1 et pour réduire la complexité. Tout en trouvant la valeur de préfixe de le modèle, si le caractère de préfixe ne correspond pas à la valeur de ce caractère est 0. Lors de la recherche de la valeur du préfixe, nous vérifions à partir des premiers caractères du motif. Pour trouver la valeur de préfixe, la valeur du 1er caractère du motif sera toujours 0. Incrémente la position du pointeur de 1. Concordance le

caractère avec le caractère précédent du motif. Si concordance trouvée, la valeur du caractère sera le nombre de caractères correspondants précédemment. Le décalage du motif va être calculé en soustrayant l'indice de caractère précédent et la valeur de préfixe précédente, où la non-concordance se produire.

### Exemple :

Le motif **abcd** a pour tableau auxiliaire  $\pi = [0 ; 0 ; 0 ; 1 ; 2 ; 0]$  car pour a, ab, abc leur bord est de longueur nulle (par convention a n'est pas suffixe et préfixe propre, donc de bord de longueur nulle). Cependant pour **abca**, a est préfixe propre et suffixe et sa longueur est 1, puis **abcab** a pour bord ab de longueur 2. Pour finir, **abcabd** n'a pas de bord de longueur non nulle.

### Algorithme 1 Calcul- $\pi$

```

Fonction Calcul- $\pi$ (P) : tableau  $\pi$ 
m = longueur(P)
 $\pi$  = tableau de taille m
 $\pi$  [1] = 0
k = 0 [indice de parcours du motif P]
Pour i de2 a m faire
  Tantque ( $k > 0$  et  $\pi$  [ $k + 1$ ]  $\neq$   $\pi[i]$ ) faire
    k =  $\pi[k]$ 
  Si ( $\pi$  [ $k + 1$ ] =  $\pi[i]$ ) alors
    k = k + 1
  Fin Si
   $\pi[i]$  = k
  Fin Pour
 $\pi$ 
Fin

```

Figure 5 : Algorithme 1[Calcul- $\pi$ ]

On veut étudier la complexité de cet algorithme. On a une boucle ‘for’ de longueur  $m - 1$  avec des opérations à coup constant à l’intérieur sauf la boucle ‘while’. On va compter le nombre d’itérations maximum de la boucle ‘while’ quand on exécute le programme. On remarque trois propriétés :

- **k > 0** car tout élément de  $\pi$  est positif (car chaque élément représente une longueur de mot)
- **$\pi(k) < k$**  car **k < i** donc quand on définit  $\pi(i)$ , on met un élément strictement inférieur à i.
- **k s’incrémentera au plus de 1** à chaque fois dans la boucle ‘for’, donc  $m - 1$  fois au plus dans le programme Ainsi on diminue strictement au plus  $m - 1$  fois la valeur de k dans le

programme, donc on passe au plus  $m - 1$  fois dans la boucle ‘while’. Ainsi la complexité est en  $O(m)$ . On peut maintenant donner l’algorithme de Knuth Morris Pratt.

## Algorithme 2 : Knuth Morris Pratt

```

Procédure KMP ((T, P))
    n = longueur(T)
    m = longueur(P)
    π = Calcul -π(P)
    k = 0 [indice de parcours du motif P]
    Pour i de 1 à n faire
        Tantque (k > 0 et π [k + 1] ≠ T[i]) faire
            k = π[k]
        Si (π [k + 1] = T[i]) alors
            k = k + 1
        FinSi
        Si (k = m + 1) alors
            Afficher "occurrence de P à l'indice i - m + 1"
            k = π(k)
        FinSi
    FinPour
Fin

```

Figure 6: Algorithme Knuth-Morris-Pratt (KMP)

- Représentation graphique du KMP String algorithme de recherche

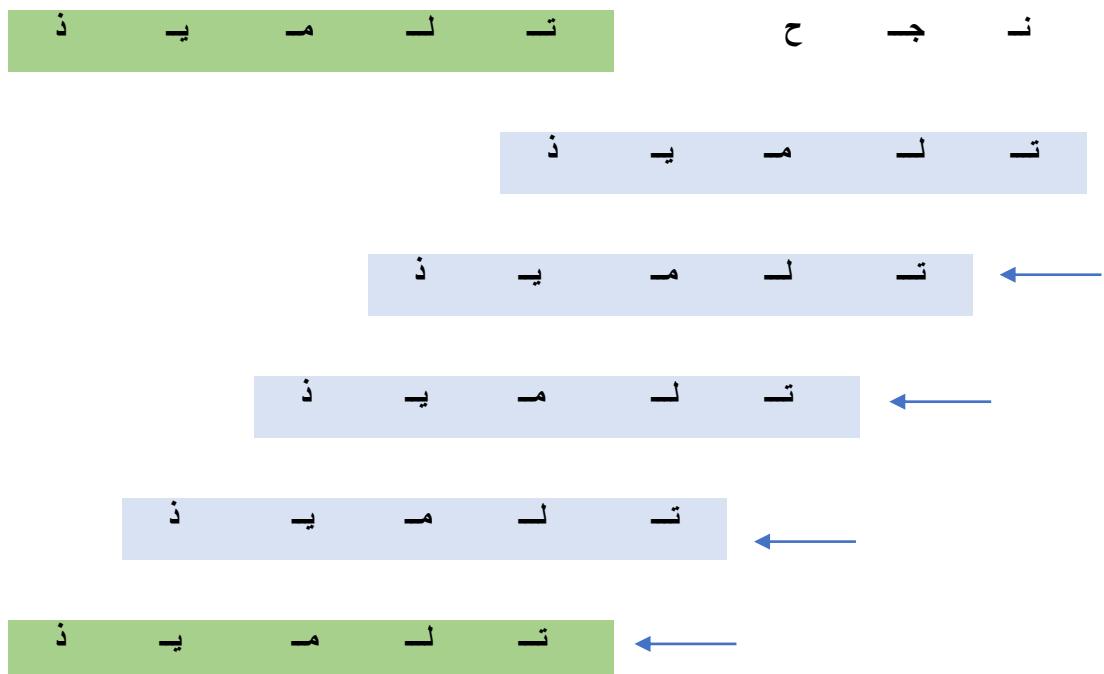


Figure 7 : Représentation graphique du KMP

## Programmation d'algorithme KMP

Pattern Prefix Value Function:

```
int[] Val = new int[pat.length];
int curr_index = 1;
int j=0, count = 0;
Val[0] = 0;
int M = pat.length;
while (curr_index< M) {
int x;
for(x=0;x<curr_index;x++) {
count=0;
int a=curr_index;
int b=x;
while(b>-1) {
if(pat[a]==pat[b]) {
count=count+1;
Val[curr_index]=count;
} else {
break;
}
a--;
b--;
}
curr_index=curr_index+1;
}
```

Search Pattern Function:

```
for (s = 0; s < N; ) {
int count = 0;
for (int j = 0; j < pat.length; j++) {
if (s + j >= N) {
break;
} else {
if (pat[j] == str[s + j]) {
count = count + 1;
} else {
s += (j + 1) - (Val[j]);
break;
}
}
}
if (count == pat.length) {
System.out.println(s+" : "+الكلمة توجد في الموضع");
s += pat.length;
}}
```

Figure 8 : Programmation d'algorithme KMP

## 4.3 Boyer-Moore (BM) Algorithm :

L'algorithme de Boyer-Moore est la concordance de chaîne la plus rapide algorithme. La complexité de cet algorithme est minimale. L'algorithme commence à balayer le motif de droite à gauche (c'est-à-dire que la concordance commence à partir du nième caractère du pattern au lieu du 1er.). Cela augmente davantage le décalage par rapport au changement d'algorithme KMP. Dans KMP, la valeur du préfixe du caractère est trouvé pour le décalage. Mais dans l'algorithme BM la méthode est différente. En BM la valeur du caractère est obtenu en soustrayant le 1 et la valeur de l'indice du caractère de la longueur totale du motif. Le spécialle symbole (\*) astérisque est ajouté à la fin du pattern. la valeur du symbole astérisque et le dernier caractère du motif est égal à la longueur du motif. Si le dernier caractère du motif et le 1er caractère du motif est identique, alors la valeur du 1er caractère sera égale au dernier caractère de la valeur du motif. Après avoir construit la table de valeurs, les caractères uniques du motif sont trouvésy compris le symbole astérisque. Les valeurs de ces personnages seront tirés du tableau des valeurs. Si un caractère est répété dans le motif, puis la dernière valeur de ce caractère est considéré à partir du LHS [2]. Dans l'algorithme de Boyer-Moore, le caractère d'incompatibilité du texte la séquence est considérée plutôt que le caractère du modèle ce qui est fait dans les algorithmes KMP et Naive. Ensuite la valeur de le caractère non concordant est trouvé dans la table de valeurs à l'aide valeur unique des caractères. Si le caractère incompatible n'est pas présent dans les caractères uniques, la valeur de l'astérisque est prise comme valeur de décalage, et le motif est décalé à l'aide desMême.

```
ALGORITHMBoyerMoore(String T, String P)
Debut
i ← m-1
j ← m-1
Faire
Si (P[i] = T[i]) alors
Si (j=0) alors
    Retourner i
Sinon
    i = i-1
    j = j-1
Finsi
Sinon
i = i + m - minimum(j, 1 + last(T[i]))
j = m-1

Finsi
Tantque (i > n-1)
Retourner -1
Fin
```

Figure 9: Algorithme Boyer Moore

## 5. Compléxité des algorithmes pattern matching:

Dans cette section, la comparaison de trois algorithmes , le tableau ci- dessous explique la meilleure et la compléxité matching time de l'algorithme.

Tableau 6 : Compléxité des algorithmes pattern matching

Algorithm	Preprocessing time	Complexity matching time
Naïve String search algorithm	$O(\text{no preprocessing})$	$O((n-m+1)m)$
Knuth-morris-Pratt Algorithm	$O(m)$	$O(n)$
Boyer-Moore String search Algorithm	$O(m)$	Average $O(n/m)$ Worst $O(m n)$

# Chapitre III : Analyse et conception

## 1. Introduction

Dans cette partie, on va analyser et modéliser les besoins du client avec le langage UML. L'activité d'analyse et de conception permet de traduire les besoins fonctionnels et les contraintes issues du cahier des charges et de la spécification des exigences dans un langage plus professionnel et compréhensible par tous les individus intervenants dans la réalisation et l'utilisation de l'application.



## 2. Presentation UML

Figure 10 : logo UML

Le langage de Modélisation Unifiée, de l'anglais Unified Modeling Language (UML), est langage de modélisation graphique à base de pictogramme conçue pour fournir une méthode normalisée pour visualiser la conception d'un système. Il est couramment utilisé en développement logiciel et en conception orienté objet. Et permet aussi de représenter l'ensemble des éléments et leurs liens respectifs. Ces éléments sont représentés par un ensemble des diagrammes classés deux vues :

- Les vues statiques qui représentent le système physiquement :
  - ❖ Diagramme de classe
  - ❖ Diagramme de cas d'utilisation
- Les vues dynamiques, montrant le fonctionnement du système :

- ❖ Diagrammes de séquence.

## 3. Environnement de travail



Figure 11 : logo StarUML

- ✓ **StarUML** est un logiciel de modélisation UML, qui a été « cédé comme open source » par son éditeur, à la fin de son exploitation commerciale (qui visiblement continue ...), sous une licence modifiée de GNU GPL.
- ✓ **StarUML** gère la plupart des diagrammes spécifiés dans la norme UML 2.0.

L'export peut se faire dans les formats JPEG, WMF, SVG et PNG

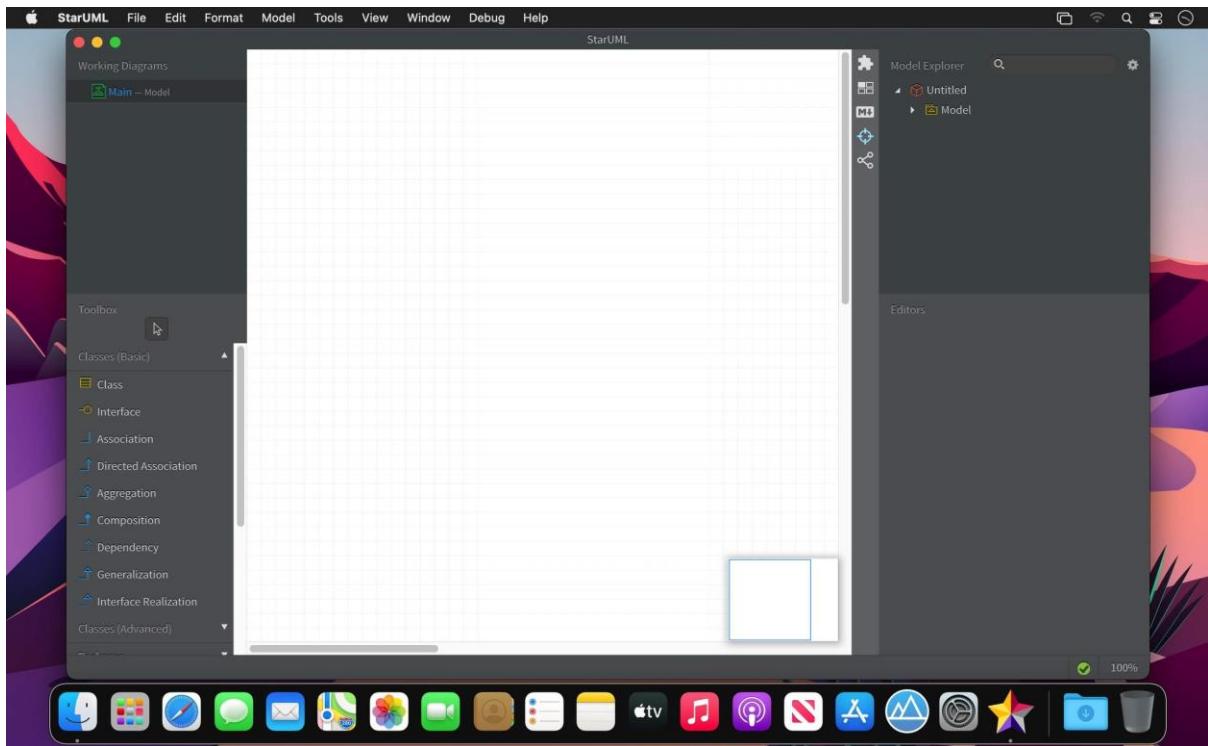


Figure 12 : l'interface de StarUML

## 4. Identification des acteurs

Un acteur est un rôle joué par un utilisateur humain ou un autre système qui interagit directement avec le système étudié. Un acteur participe à au moins un cas d'utilisation. Les acteurs qui interagissent avec notre application sont :

- **Utilisateur** : L'utilisateur est un acteur principal qui interagit avec notre site web et le plus important qui doit être :
  - ❖ Saisir le texte
  - ❖ Saisir le pattern
  - ❖ Supprimer le texte
  - ❖ Supprimer le pattern
  - ❖ Rechercher
- **Admin** : jouer un rôle majeur pour assurer le fonctionnement normal du site web.  
C'est la personne qui est au service de :
  - Gérer les problèmes de l'utilisateur à travers les commentaires.

## 5. Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation offre une vue fonctionnelle sur le comportement du système. Dans un diagramme de cas d'utilisation, les utilisateurs, appelés acteurs, interagissent avec les cas d'utilisation (les fonctions offertes par l'application).

### 5.1 Diagramme de cas d'utilisation général

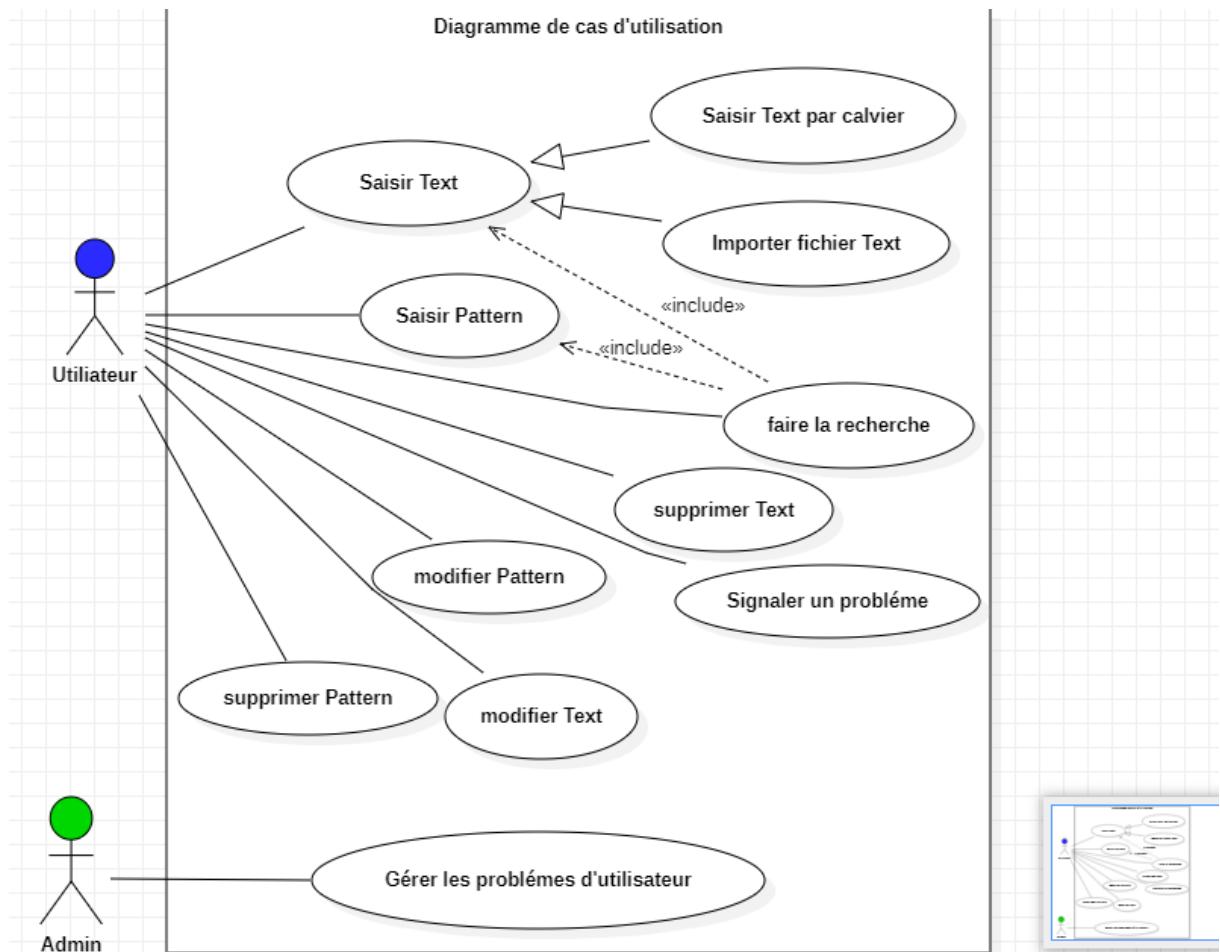


Figure 13 : Diagramme général de cas d'utilisation

## 6. Diagramme de classe

Le diagramme de classes est considéré comme le plus important de la modélisation orientée objet. Il s'agit d'une vue statique, car on ne tient pas compte du facteur temporel dans le comportement du système.

Le diagramme de classes modélise les concepts du domaine d'application ainsi que les concepts internes créés de toutes pièces dans le cadre de l'implémentation d'une application.

## 6.1 Diagramme général de classe

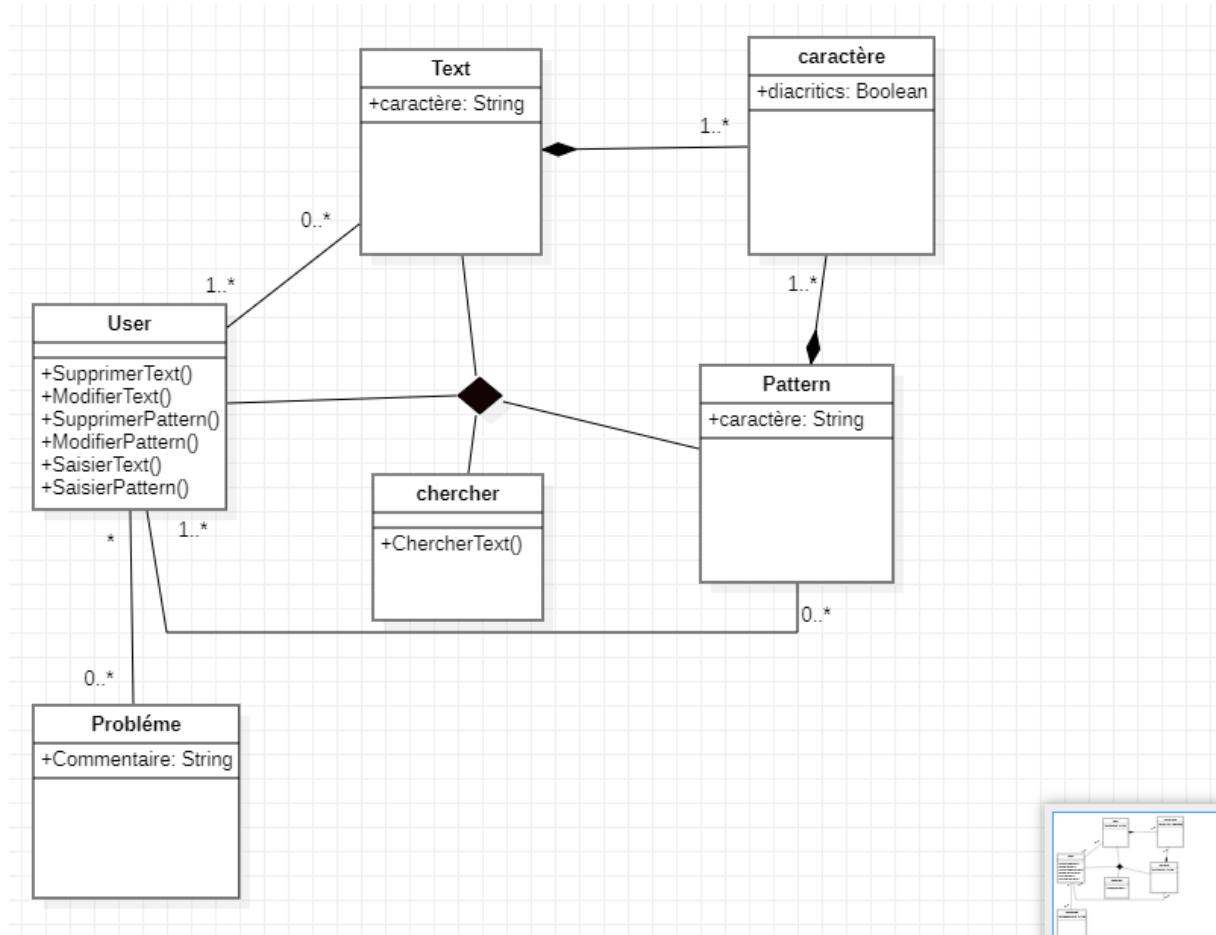


Figure 14 : Diagramme général de classe

## Conclusion

Ce chapitre a présenté la partie conceptuelle de notre application web et qui nous a permis d'élaborer le diagramme de classe. Ce diagramme va par la suite l'implémentation de notre application web.

# Chapitre IV : réalisation

## 1. Introduction

La phase de l'implémentation est la phase pendant laquelle nous travaillons sur le développement de l'application. Le choix des outils de développement a une grande influence sur le coût du temps de programmation et la flexibilité du produit à réaliser. Cette étape comprend la conversion du modèle conceptuel précédemment établi en composants logiciels qui composent notre application.

Dans ce chapitre, nous irons brièvement présenter les différents outils que nous utiliserons tout au long du processus de développement de l'application.

## 2. Les langages de programmation

### 2.1 Back End languages

#### 2.1.1 Java SE

Figure 15 : Java SE



**Java Platform, Standard Edition**, ou **Java SE** (anciennement **Java 2 Platform, Standard Edition**, ou **J2SE**), est une spécification de la plate-forme Java d'Oracle, destinée typiquement aux applications pour poste de travail.

La plate-forme est composée, outre les API de base :

- ✓ Des API spécialisées dans le poste client (JFC et donc Swing, AWT et Java2D) ;
- ✓ Des API d'usage général comme JAXP (pour le parsing XML) ;
- ✓ De JDBC (pour la gestion des bases de données).

À chaque version de Java SE correspond notamment, comme toutes les éditions Java :

- ✓ Les Java Spécification Requetés (JSR), constituant les spécifications de la version considérée ;
- ✓ Un Java Development Kit (JDK), contenant les bibliothèques logicielles ;
- ✓ Un Java Runtime Environment (JRE), contenant le seul environnement d'exécution (compris de base dans le JDK).

Les avantages :

- ❖ Portabilité excellente.
- ❖ Langage puissant
- ❖ Langage de haut niveau
- ❖ JDK très riche
- ❖ Nombreuses librairies tierces
- ❖ Très grande productivité
- ❖ Nombreuses implémentations, JVM et compilateurs, libres ou non.
- ❖ IDE de très bonne qualité et libres : Eclipse et NetBeans par exemple.
- ❖ Supporté par de nombreuses entreprises telles que Sun  
Ou encore IBM et des projets comme Apache.
- ❖ Grande stabilité du code à travers le temps.
- ❖ Le langage est soutenu par Oracle (anciennement\* Sun Microsystems).
- ❖ Stabilité du JFC.
- ❖ Sécurité excellente avec Java/CORBA.
- ❖ Solution utilisée par de nombreuses grandes entreprises et institutions financières.
- ❖ Portabilité
- ❖ Interopérabilité

Les inconvénients :

- ❖ Il est plus demandant au niveau du microprocesseur
- ❖ Ne permet pas d'accéder directement au matériel

## 2.1.2 J2EE



Figure 16 : logo Java EE

Java Enterprise Edition, ou Java EE (anciennement J2EE), est une spécification pour la technique Java d'Oracle plus particulièrement destinée aux applications d'entreprise. Ces applications sont considérées dans une approche multi-niveaux. Dans ce but, toute implémentation de cette spécification contient un ensemble d'extensions au Framework Java standard (JSE, Java Standard Edition) afin de faciliter notamment la création d'applications réparties.

Pour ce faire, Java EE définit les éléments suivants :

- ✓ Une plate-forme (Java EE Platform), pour héberger et exécuter les applications, incluant outre Java SE des bibliothèques logicielles additionnelles du Java Development Kit (JDK).
- ✓ Une suite de tests (Java EE Compatibility Test Suite) pour vérifier la compatibilité.
- ✓ Une réalisation de référence (Java EE Reference Implementation), dénommée Glass Fish.
- ✓ Un catalogue de bonnes pratiques (Java EE BluePrints).

Justification du choix :

- ✓ Son adaptation à tous les systèmes d'exploitation.
- ✓ Langage professionnel pour le développement des applications web.
- ✓ Sa syntaxe proche du langage C++, ce qui nous faciliteras le codage.

## 2.2 Front end languages

### 2.2.1 HTML



Figure 17 : logo HTML

**HTML(Hyper Text Markup Language)** est un langage de balisage conçu pour représenter les pages web. C'est un langage permettant d'écrire de l'hypertexte, d'où son nom. HTML permet également de structurer sémantiquement et logiquement et de mettre en forme le contenu des pages, d'inclure des ressources multimédias dont des images, des formulaires de saisie et des programmes informatiques. Il permet de créer des documents interopérables avec des équipements très variés de manière conforme aux exigences de l'accessibilité du web.

## 2.2.2 CSS



Figure 18 : logo CSS

**CSS** est un langage informatique qui décrit la présentation des documents HTML et XML. Les standards définissant CSS sont publiés par le World Wide Web Consortium (W3C). Introduit au milieu des années 1990, CSS devient couramment utilisé dans la conception de sites web et bien pris en charge par les navigateurs web dans les années 2000.

## 2.2.3 JavaScript



Figure 19 : logo JavaScript

**JavaScript** est un langage de programmation de scripts principalement employé dans les pages web interactives. C'est un langage orienté objet à prototype, c'est-à-dire que les bases du langage et ses principales interfaces sont fournies par des objets qui ne sont pas des

instances de classes. À chaque fois qu'une page web fait plus que simplement afficher du contenu statique (afficher du contenu mis à jour à des temps déterminés, des cartes interactives, des animations 2D/3D, des menus vidéo défilants, etc...) JavaScript a de bonnes chances d'être impliqué.

## 3. Environnement de développement

### 3.1 Visual Studio Code

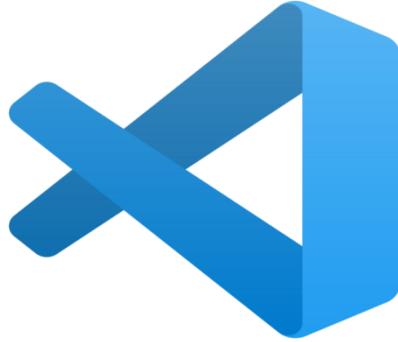


Figure 20 : logo visual studio code

**Visual Studio Code** est un éditeur de code simplifié prenant en charge les opérations de développement telles que le débogage, l'exécution de tâches et le contrôle de version. Il vise à fournir uniquement les outils dont un développeur a besoin pour un cycle de création de code-débogage rapide et laisse des flux de travail plus complexes à des IDE plus complets, tels que Visual Studio IDE.

### 3.2 Apache Tomcat



Figure 21 : Apache Tomcat

**Apache Tomcat** est un conteneur web libre de servlets et JSP Java EE. Issu du projet Jakarta, c'est un des nombreux projets de l'Apache Software Foundation. Il implémente les spécifications des servlets et des JSP du Java Community Process, est paramétrable par des fichiers XML et de propriétés, et inclut des outils pour la configuration et la gestion. Il comporte également un serveur HTTP.

Justification du choix :

- ✓ Il nous permettra de gérer les jsp et les servlets

### 3.3 Adobe Photoshop



Figure 22 : logo Adobe Photoshop

**Photoshop** est un logiciel de retouche, de traitement et de dessin assisté par ordinateur, lancé en 1990 sur Mac OS puis en 1992 sur Windows qui a révolutionné la photographie et jusqu'à la conception que l'on pouvait s'en faire. Édité par Adobe, il est principalement utilisé pour le traitement de photographies numériques, mais sert également à la création d'images. Il travaille essentiellement sur images matricielles car les images sont constituées d'une grille de points appelés pixels. L'intérêt de ces images est de reproduire des gradations subtiles de couleurs.

### 3.4 Plateforme de développement IDE Eclipse



Figure 23 : logo eclipse

**Eclipse** est un EDI (Environnement de Développement Intégré) ou en anglais un IDE (Integrated Development Environment). Il vise à développer un environnement de production de logiciels libres qui soit extensible, universel et polyvalent, en s'appuyant principalement sur Java. Son objectif est de produire et fournir des outils pour la réalisation de logiciels, englobant les activités de programmation (notamment environnement de développement intégré et Framework) recouvrant modélisation, conception, gestion de configuration, reporting...

### 3.5 PhpMyAdmin



Figure 24 : logo PhpMyAdmin

**PhpMyAdmin** (PMA) est une application web de gestion pour les système de gestion de base de données MySQL réalisée en PHP et distribuée sous licence GNU GPL.

Il s'agit de l'une des plus célèbres interfaces pour gérer une base de données MySQL sur un serveur PHP. De nombreux hébergeurs, qu'ils soient gratuits ou payants, le proposent ce qui permet à l'utilisateur de ne pas avoir à l'installer.

Cette interface pratique permet d'exécuter, très facilement et sans grandes connaissances dans le domaine des bases de données, de nombreuses requêtes comme les créations de table comme les créations de table de données, les insertions, les mises à jour, les suppressions, les modifications de structure de la base de données.

Ce système est très pratique pour sauvegarder une base de données sous forme de fichier. SQL et ainsi transférer facilement ses données. De plus celui-ci accepte la formulation de requêtes SQL directement en langage SQL, cela permet de tester ses requêtes par exemple lors de la création d'un site et ainsi de gagner un temps précieux.

#### Justification du choix :

- ✓ Permet une gestion web de MYSQL.
- ✓ Gratuit et ne nécessite pas d'installation.
- ✓ Pratique et facile dans les requêtes SQL.

## 3.6 WampServer



Figure 25 : logo WampServer

**WampServer** est une plate-forme de développement Web sous Windows pour des applications Web dynamiques à l'aide du serveur Apache2, du langage de scripts PHP et d'une base de données MySQL. Il possède également PHPMyAdmin pour gérer plus facilement vos bases de données.

## 4. Outils matériels

La programmation a été effectuée sur des ordinateurs dotés de caractéristiques suivantes :

### *Dell l'attitude E6400*

- Marque : DELL.
- Processeur : Intel(R) Core (TM) 2 Duo CPU T9600@ 2.80GHz 2.80 GHz.
- Mémoire vive 4Go.
- Disque dur 148 Go.
- Système d'exploitation : Windows 7 Professionnel 64-bits.

### *HP Elitebook*

- Marque : HP.
- Processeur : Intel(R) Core (TM) i3-3700U CPU N2840@ 2.16GHz 2.16 GH.
- Mémoire vive 4Go.
- Disque dur 512 Go.
- Système d'exploitation : Windows 10 Professionnel 64-bits

### *Hp probook 640*

- Marque : HP.
- Processeur : Intel(R) Core(TM) i5-4200M CPU @ 2.50GHz 2.50 GHz.
- Mémoire vive 6 Go.
- Disque dur 240 Go.
- Système d'exploitation : Windows 10 Professionnel 64-bits.

## 5. Problèmes rencontrés et leurs solutions

Il existe quatre cas de recherche d'un mot dans un texte spécifique :

Tableau 7 : Quatre cas de recherche

Cas	1	2	3	4
Texte	contenant tashkeel	Sans tashkeel	Sans tashkeel	contenant tashkeel
Modèle	sans tashkeel	contenant tashkeel	contenant tashkeel	Sans tashkeel

#### ❖ Texte et modèle contenant tashkeel

Dans le texte, on recherche un mot ou chaîne qui lui correspond dans le texte, comme le texte français, en ce sens qu'on utilise des algorithmes de recherche.

#### ❖ Texte et modèle sans tashkeel

Dans le texte, on recherche un mot ou chaîne qui lui correspond dans le texte, comme le texte français, en ce sens qu'on utilise des algorithmes de recherche.

#### ❖ Texte sans tashkeel et modèle contenant tashkeel

Nous supprimons tashkeel du mot et le recherchons dans le texte en faisant correspondre le mot du texte et en trouvant son emplacement en fonction du texte.

#### ❖ Texte contenant tashkeel et modèle sans tashkeel

On met un symbole derrière chaque lettre du mot sans le mettre derrière l'espace et on cherche à travers un algorithme.

## 5.1 Problèmes techniques et leurs solution

Nous avons rencontré plusieurs problèmes au niveau de la programmation que nous avons essayé de les résoudre :

- ✓ L'arabe n'est pas comme les autres langues, comme l'anglais le français, qui commencent de gauche à droite, par contre l'arabe commence de droite à gauche.  
Le langage de programmation Java a résolu ce problème parce qu'elle défend l'arabe

- ✓ Du côté diacritique, nous avons éprouvé des difficultés sa intégrer le diacritique à la lettre.  
Diacritique prend 1 case comme caractère et shada prend 2 cases.

Tableau 8 :: Comparaison entre un mot contenant un tashkeel et un mot sans tashkeel [نَجْحٌ وَنَجْحٌ]

			۲	۴	۵
۰	۱	۳	۶	۷	۸

- ❖ **Premier** : est composé de 6 caractères.
  - ❖ **Deuxième** : est composé de 3 caractères.

Pour surmonter ce problème, nous avons créé une fonction qui s'appelle **Dollar** qui augmente de \$

Après à chaque caractère d'un mot qui n'inclut pas de shakeel.

Tableau 9 : exemple d'application de la technique \$

\$	1	\$	1	•	L.
----	---	----	---	---	----

\$ : \{ \circ \circ \circ \overset{o}{\circ} \overset{\vartheta}{\circ} \overset{\vartheta}{\circ} \overset{=}{\circ} \overset{-}{\circ} \}

- ✓ Lorsque l'on compare un mot contenant shakeel dans un texte, on compare le mot avec tous les mots du texte, en cas de concordance, on l'appellera, puis on supprimera le shakeel du mot et on le cherchera à nouveau.
  - ✓ Lorsque l'on compare un mot contenant sans shakeel dans un texte, on compare le mot avec tous les mots du texte, en cas de concordance, on l'appellera, puis on va entrer la fonction dollar et appeler les mots identiques.
  - ✓ Dans le cas le mot (shakeel) et texte (sans shakeel) on a fait une fonction **Strip\_shakeel** pour supprimer shakeel.
  - ✓ Après avoir envoyé les informations de type String en arabe de JSP à Servlet ils sont reçues  
Sous la forme iso\_8859-1  
pour résoudre ce problème, on a utilisé la fonction getbytes de classe String

qui donne les octets à l'information, afin de la convertir en utf-8

## 6. Les interfaces graphiques

### 6.1 Test d'algorithme

Les étapes pour chercher les informations dans un texte arabe :

❖ **Etape1 :** il vous suffit de saisir directement votre texte arabe dans **le champ réservé à cet effet ou de faire un simple copier-coller dans le même champ**  
Ou bien Cliquer sur le bouton « **ouvrir un fichier texte** » une boite de recherche s'ouvre pour choisir votre fichier texte.

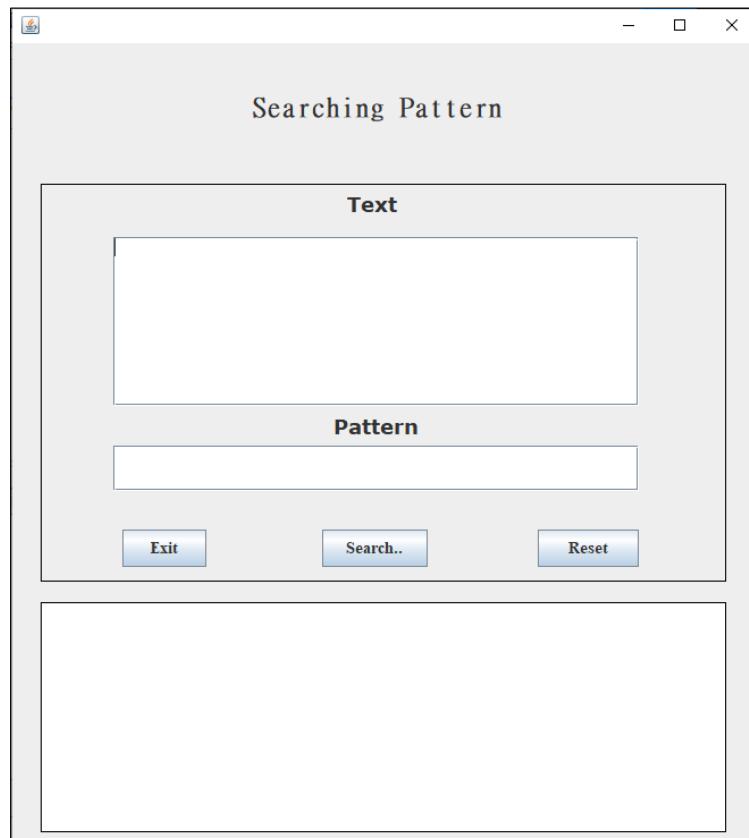


Figure 26 : Interface graphique

- ❖ **Etape 2 :** Entrez votre chaîne de caractères arabe que vous souhaitez rechercher dans le champ réservé « Text »

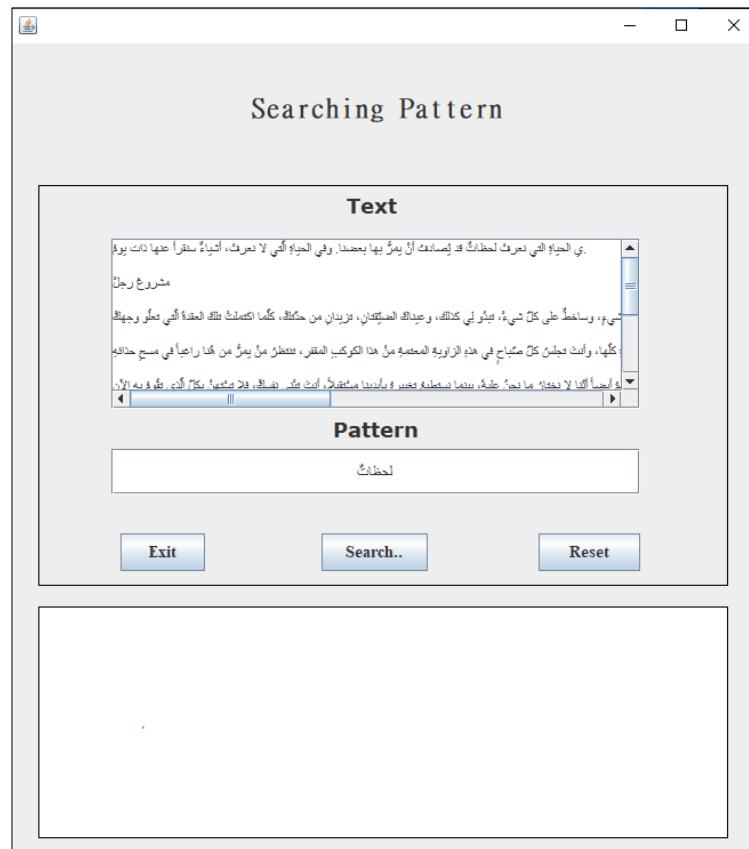


Figure 27 : Interface graphique lors de la saisie

- ❖ **Etape 3 :** Et finalement cliquer sur le bouton « search » pour affiche les résultats sous forme un message qui contient le nombre d'occurrence d'information et de les localiser dans le texte.

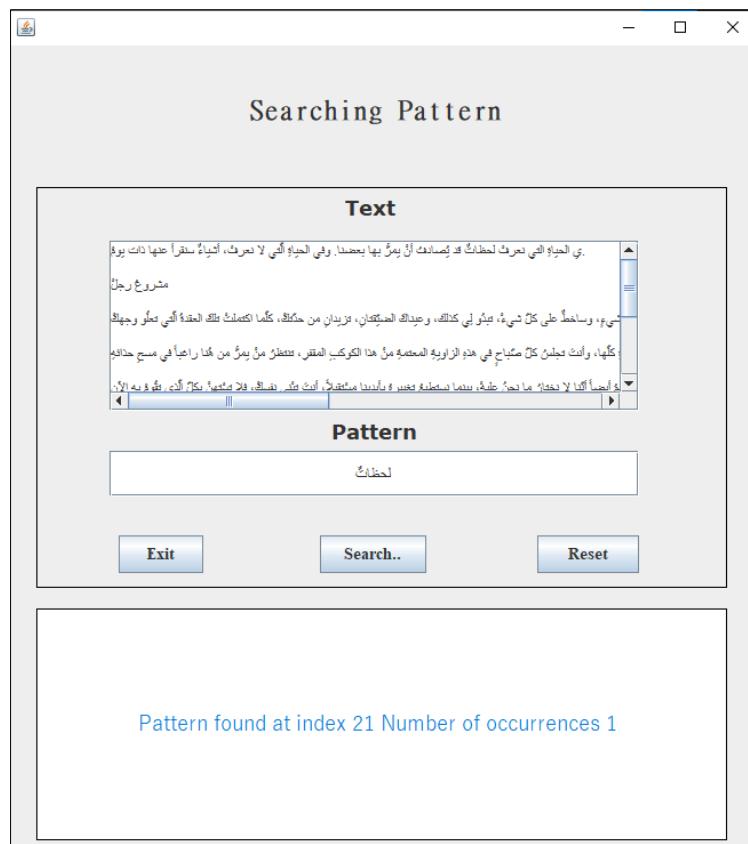


Figure 28 : Interface après la recherche

**Bouton Exit :** Une fois que vous cliquez sur le bouton "Exit", nous sortons automatiquement de l'interface

**Bouton Reset :** Une fois que vous cliquez sur le bouton "Réinitialiser", vous réinitialiserez les champs d'interface

## Les étapes pour chercher les informations dans un texte arabe par fichier :

- ❖ **Etape1 :** il vous suffit de saisir directement votre texte arabe ou Cliquer sur le bouton « **ouvrir un fichier texte** » une boite de recherche s'ouvre pour choisir votre fichier texte.

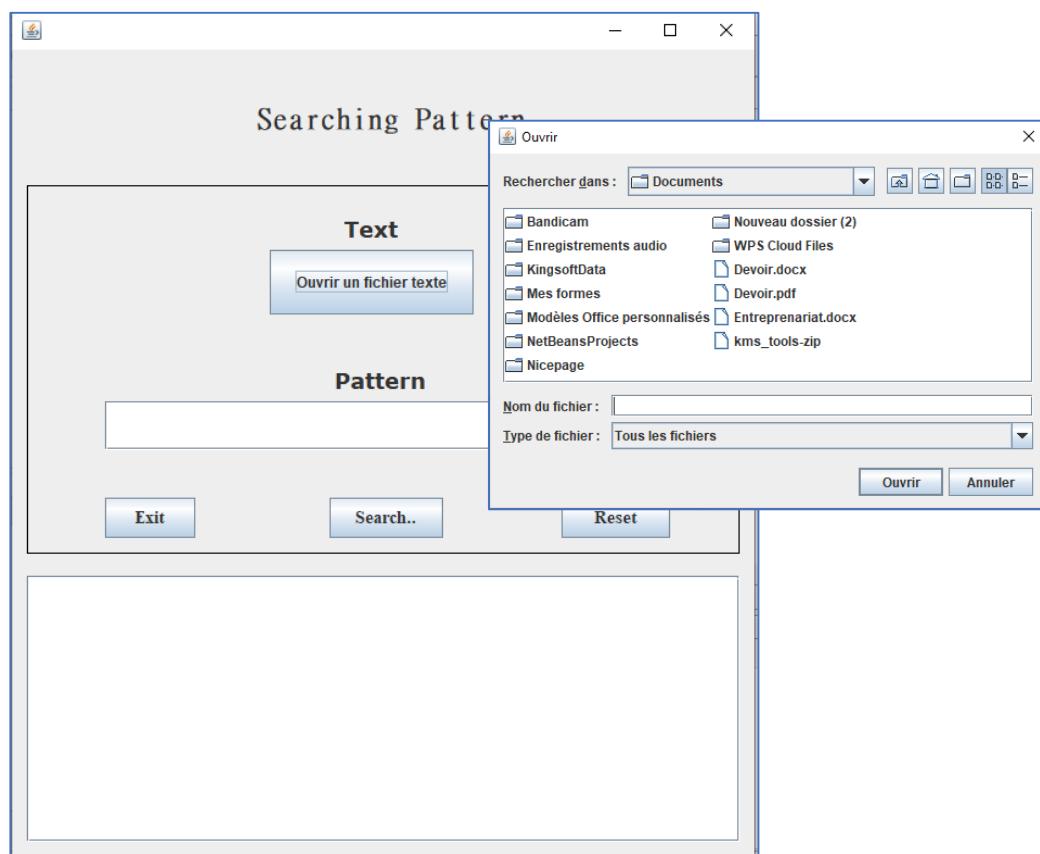


Figure 29 : Interface graphique -fichier-

- ❖ **Etape 2 :** Choisir votre fichier arabe que vous souhaitez rechercher dans le champ réservé « Text »

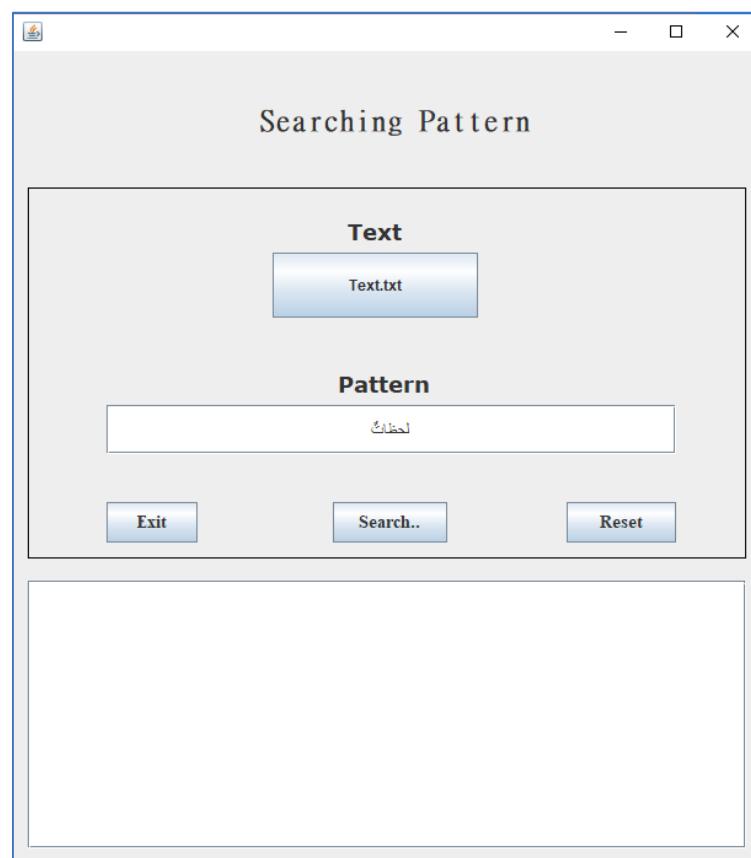


Figure 30 : Interface graphique lors de la saisie -fichier

- ❖ **Etape 3 :** Et finalement cliquer sur le bouton « search » pour affiche les résultats sous forme un message qui contient le nombre d'occurrence d'information et de les localiser dans le texte.

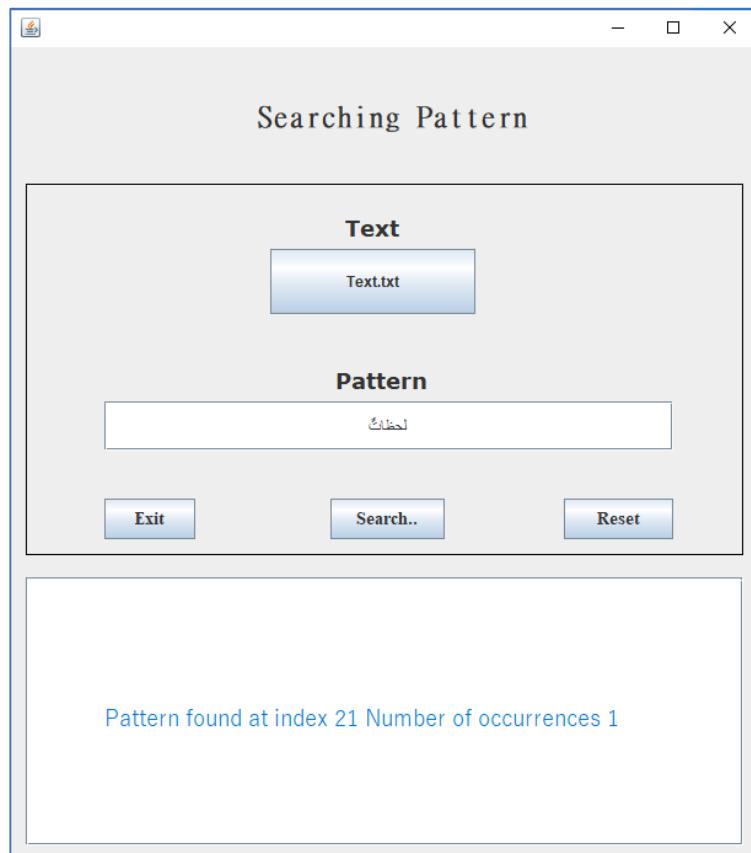


Figure 31 : Interface graphique après la recherche -fichier

**Bouton Exit :** Une fois que vous cliquez sur le bouton "Exit", nous sortons automatiquement de l'interface

**Bouton Reset :** Une fois que vous cliquez sur le bouton "Réinitialiser", vous réinitialiserez les champs d'interface

## 6.2 Interfaces graphiques commentées

Ces pages présentées ci-après représentent les fonctionnalités les plus importantes de l'application.

### 6.2.1 Page de recherche

Quand l'utilisateur accède à l'application, la première page qui s'ouvre c'est celle d'accueil :

Cette page permettre à l'utilisateur de chercher facilement des informations dans des textes arabe

Une cliquer sur « chercheur de mot et caractères » il permette **saisir directement votre texte arabe** dans le champ réservé à cet effet ou de faire un simple **copier-coller** dans le même champ et aussi Entrez votre chaine de caractères arabe que vous souhaitez rechercher dans le champ réservé.

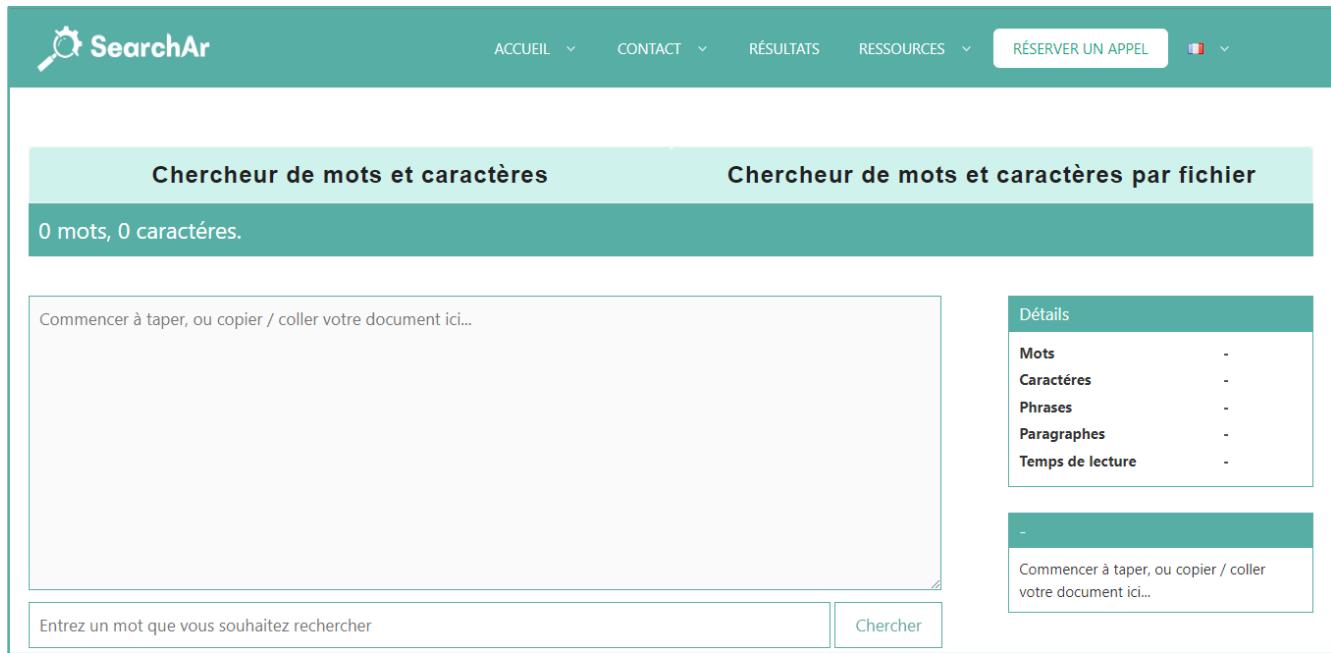


Figure 32 : Capture d'écran de la page recherche

Quand il clique sur bouton « chercher », l’interface de résultats suivante sera affichée :

The screenshot shows a search interface with two main sections: 'Chercheur de mots et caractères' (left) and 'Chercheur de mots et caractères par fichier' (right).

**Chercheur de mots et caractères**

47 mots, 240 caractères.

Result text (Arabic): قراشة ملأة تطير في النستان، غلوة مهندمة نذهبن الإنسان، أهدافها محددة، حركتها مرتيبة، تخوم يائطا، تخطي في نعومة نشر السلام، قراشة ملأة تطير بلا أنقطاع، بالنهار المشرق تملاً اليقاب، تجت القزد المزروع، تلمة في وقت الجوع، تمتص زحيف الآهار، تحيي جلبي الأشجار، من وزدة لوزدة، تطير يائطا.

**Chercheur de mots et caractères par fichier**

Détails

Mots	47
Caractères	240
Phrases	2
Paragraphes	-
Temps de lecture	~33.6s

فَرَاشَةٌ مَلَأَةٌ تَطِيرُ فِي النِّسْنَانِ، غَلُوْةٌ مَهَنْدِمَةٌ نَذَهَبُنَا إِلَى النِّسَانِ، أَهْدَافُهَا مَهَدَّدَةٌ، حَرْكَاتُهَا مَرْتِبَةٌ، تَخُومُ يَائِطَا، تَخْطُّ فِي نَعْوَمَةٍ نَشِيرُ السَّلَامَ، فَرَاشَةٌ مَلَأَةٌ تَطِيرُ بِلَا أَنْقَطَاءِ، بِالنَّهَارِ الْمُشْرِقِ تَمَلَّأُ الْيَقَابَ، تَجْتُ الْوَزَدَ الْمَزْرُوعَ، تَلْمِمَةٌ فِي وَقْتِ الْجُوعِ، تَمْنَصُ زَحِيفَ الْآهَارِ، تُحِيِّي جَلْبَيَ الْأَشْجَارِ، مِنْ وَزْدَةٍ لَوْزَدَةٍ، تَطِيرُ يَائِطَا.

Chercher

47 mots, 240 caractères.

Figure 33 : capture d’écran après la recherche

### 6.2.2 Page de recherche par fichier

Si le client cliquer sur « chercheur de mot et caractères par fichier » il permette importé votre fichier qui contient le texte arabe dans le champ réservé et aussi Entrez votre chaîne de caractères arabe que vous souhaitez rechercher dans le champ réservé



Figure 34 : capture d'écran de la page de recherche par un fichier

Quand il clique sur bouton « chercher », l'interface de résultats suivante sera affichée :



Figure 35 : capture d'écran de la page de recherche par un fichier après la recherche

Si l'utilisateur souhaite nous contacter ou envoyer des commentaires, il suffit de cliquer sur CONTACT :

The screenshot shows the 'CONTACT' page of the SearchAr website. At the top, there's a navigation bar with links for 'ACCUEIL', 'CONTACT', 'ABOUT', 'RESSOURCES', a 'RÉSERVER UN APPEL' button, and a language switcher set to 'FR'. The main title 'CONTACT' is centered at the top of the page in a large, bold, dark blue font. Below it, the sub-section title 'Joindre SearchAr' is displayed in a teal font. To the left of the contact information, there's a note: 'N'hésitez pas à nous joindre par email, ou par téléphone'. To the right, there's an illustration of a person sitting on a large telephone receiver, holding a phone to their ear. Below the illustration is a text input field and a green 'ENVOYER' button.

Figure 36 : Capture d'écran de page de CONTACT

## Conclusion et perspectifs

Notre projet est préparé une solution au problème de la recherche d'informations dans les textes arabes. Pour ce faire, nous avons commencé à étudier les affaires pour avoir une idée claire des différents aspects que notre solution doit traiter

Sur le plan des acquis fonctionnels, ce projet de fin d'études nous a permis de renforcer nos compétences relationnelles, et de mesurer les différences, notables, entre le monde Universitaire et le monde professionnel.

Sur le plan technique, ce projet nous a donné l'occasion de mettre en pratiques nos connaissances théoriques au niveau des langages de programmation, et les méthodes de modélisation, ainsi de surmonter les difficultés de programmation rencontrées.

Il est bien évident qu'une telle expérience ne pourra être qu'une bonne expérience. Dans laquelle nous avons pu renforcer nos connaissances, de développer notre confiance en soi, et de gérer les difficultés rencontrées.

Au cours de la réalisation de notre projet, nous avons étaient astreints par quelques limites notamment, la contrainte du temps qui était relativement un obstacle devant l'ajout de certaines autres fonctionnalités.

Cependant les tâches qui nous ont été assignées ont été accomplies dans l'ensemble. Notre projet sera de préparer une solution au problème de la recherche d'informations dans les textes arabes. Pour ce faire, nous avons commencé à étudier les affaires pour avoir une idée claire des différents aspects que notre solution doit traiter

Finalement, notre travail ne s'arrête pas à ce niveau, en effet plusieurs fonctionnalités peuvent être ajoutées à notre site web.

## *Bibliographie*

- [1] NYO ME TUN, THIN MYA MYA SWE, “Comparison of ThreePattern Matching Algorithms using DNA Sequences”, IJSETR, Vol.3, Issue.35, pp.6916-6920, 2014.
- [2] Robert Sedgewick, Kevin Wayne, “Algorithms”, Fourth Edition, Addison-Wesley,Pearson Edition, India, pp. 760-776, 2011.
- [3] Thomas Cormen,Charles E. Leiserson,Ronald L. Rivest,Clifford Stein, “Introduction to Algorithms”, McGraw-Hill Publication,India, pp.909-926, 2001.
- [4] Raju Bhukya, DVLN Somayajulu, “Exact Multiple Pattern Matching Algorithm using DNA Sequence and Pattern Pair”, International