

# 1. Introduction du projet

**Titre :**

**Développement d'une application de jeu : Motus**

**Contexte :**

Dans le cadre de ma formation de Concepteur Développeur d'Applications (CDA), j'ai été amené à réaliser un projet technique visant à démontrer mes compétences en développement logiciel. J'ai choisi de concevoir et développer une version numérique du jeu **Motus**, un jeu de lettres bien connu qui consiste à deviner un mot mystère en un nombre limité de tentatives.

Ce projet m'a permis de mobiliser l'ensemble des connaissances acquises durant la formation, notamment en **algorithmique**, **programmation orientée objet**, **interface utilisateur** et **gestion de projet**. Il a été développé en autonomie, depuis la définition des besoins jusqu'à la finalisation de l'application.

**Objectifs :**

L'objectif principal de ce projet est de développer un jeu interactif et intuitif reprenant les règles classiques de Motus, tout en intégrant une interface utilisateur simple, ergonomique et responsive.

Plus précisément, ce projet vise à :

- Implémenter une logique de jeu complète avec gestion des essais et validation des mots,
- Concevoir une interface utilisateur permettant une expérience fluide,
- Structurer le code de manière modulaire et évolutive,
- Démontrer une capacité à gérer un projet du début à la fin, seul.

## ■ 2. Cahier des charges fonctionnel

### Règles du jeu :

- Le joueur choisit un nombre de X lettres (par exemple, 6 ou 7).
- L'application choisit aléatoirement un mot ayant le nombre de lettres sélectionnées.
- L'utilisateur doit le deviner en un nombre limité d'essais (généralement 6).
- À chaque tentative, le système indique les lettres :
  - Bien placées (par exemple, en **jaune**),
  - Mal placées mais présentes dans le mot (en **rouge**),
  - Absentes du mot (en **bleue**).

### Bienvenue dans Motus

#### Essais

Essais restants : 4

m	a	r	c	h	e
m	a	n	g	e	r

Lettres bien placées

m	a	n	_	e	_
---	---	---	---	---	---

Votre tentative :

grade

 Veuillez allonger ce texte pour qu'il comporte au moins 6 caractères. Il en compte actuellement 5.

Rejouer

### Fonctionnalités principales :

- Génération aléatoire d'un mot à partir d'une API.
- Saisie utilisateur avec vérification de la validité du mot.
- Affichage dynamique des tentatives avec code couleur.
- Comptage des essais restants.

- Message de victoire ou d'échec à la fin de la partie.



- Bouton de réinitialisation de la partie.

### Fonctionnalités secondaires (optionnelles ou prévues en amélioration) :

- Sélection du niveau de difficulté (nombre de lettres).
- Intégration d'un dictionnaire pour valider les mots.
- Historique des parties ou sauvegarde de scores.
- Interface graphique améliorée (animations, sons).
- Version web ou mobile responsive.

### 3. Analyse technique

#### Langages et technologies utilisés :

- **Langage principal** : Java
- **Bibliothèque graphique** : Bootstrap (interface utilisateur)
- **Environnement de développement** : IntelliJ
- **Contrôle de version** : Git / GitHub

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!-- object xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
4     <modelVersion>4.0.0</modelVersion>
5     <parent>
6         <groupId>org.springframework.boot</groupId>
7         <artifactId>spring-boot-starter-parent</artifactId>
8         <version>3.1.3</version>
9         <relativePath/> <!-- lookup parent from repository -->
10    </parent>
11    <groupId>Proto-Final</groupId>
12    <artifactId>web-app</artifactId>
13    <version>0.0.1-SNAPSHOT</version>
14    <name>web-app</name>
15    <description>Motus</description>
16    <url/>
17    <licenses>
18        <license/>
19    </licenses>
20    <developers>
21        <developer/>
22    </developers>
23    <scm>
24        <connection/>
25        <developerConnection/>
26        <tag/>
27        <url/>
28    </scm>
29    <properties>
30        <java.version>17</java.version>
31        <spring-cloud.version>2022.0.4</spring-cloud.version>
32    </properties>
```

## Architecture logicielle :

Le projet est structuré selon une architecture **modulaire**, avec une séparation claire des responsabilités :

- **Module de gestion des mots** : sélection aléatoire, lecture depuis une API.
- **Module logique de jeu** : comparaison des mots, gestion des essais, retour des indices.
- **Interface graphique** : gestion de l'affichage, des couleurs, des boutons.
- **Fichier principal** : coordination des modules et lancement de l'application.

## Algorithme principal :

L'algorithme de comparaison des mots repose sur une double boucle :

1. Une première boucle identifie les lettres bien placées (match exact),
2. Une seconde vérifie les lettres présentes mais mal placées (tout en évitant les doublons).

## Données utilisées :

- Liste de mots stockée dans un fichier `.txt` ou dans un tableau interne.
- Aucune base de données nécessaire pour la version simple.
- Possibilité d'évolution vers une base SQLite ou fichier JSON pour l'historique ou les scores.

## 4. Déroulement du projet

### Méthodologie de travail :

J'ai adopté une démarche de développement en **mode itératif**, proche de l'**agilité**. Le projet a été découpé en plusieurs étapes avec des objectifs clairs pour chaque phase, ce qui m'a permis d'évoluer progressivement tout en assurant la stabilité de l'application à chaque ajout fonctionnel.

### Étapes du projet :

Étape	Description	Durée approximative
Analyse et définition des besoins	Rédaction du cahier des charges, règles du jeu, choix techniques	1 jour
Conception de l'interface utilisateur (maquette)	Esquisse de l'interface, réflexion sur l'ergonomie	1 jour
Développement de la logique du jeu	Comparaison des mots, gestion des couleurs, nombre d'essais	2 jours
Intégration de l'interface graphique	Mise en place de l'affichage avec Tkinter (ou autre selon ton projet)	2 jours
Tests et débogage	Vérification du bon fonctionnement des règles, détection et correction des bugs	1 jour
Ajout de fonctionnalités secondaires	Réinitialisation, feedback utilisateur, fin de partie, nettoyage graphique	1 jour
Documentation du projet	Rédaction de ce dossier, commentaires dans le code, README GitHub	1 jour

### Outils utilisés :

- **IDE** : IntelliJ
- **Contrôle de version** : Git (GitHub)
- **Langage** : Java
- **Planification** : Trello (organisation des tâches)
- **Tests** : Manuels à chaque phase (tests unitaires simples)

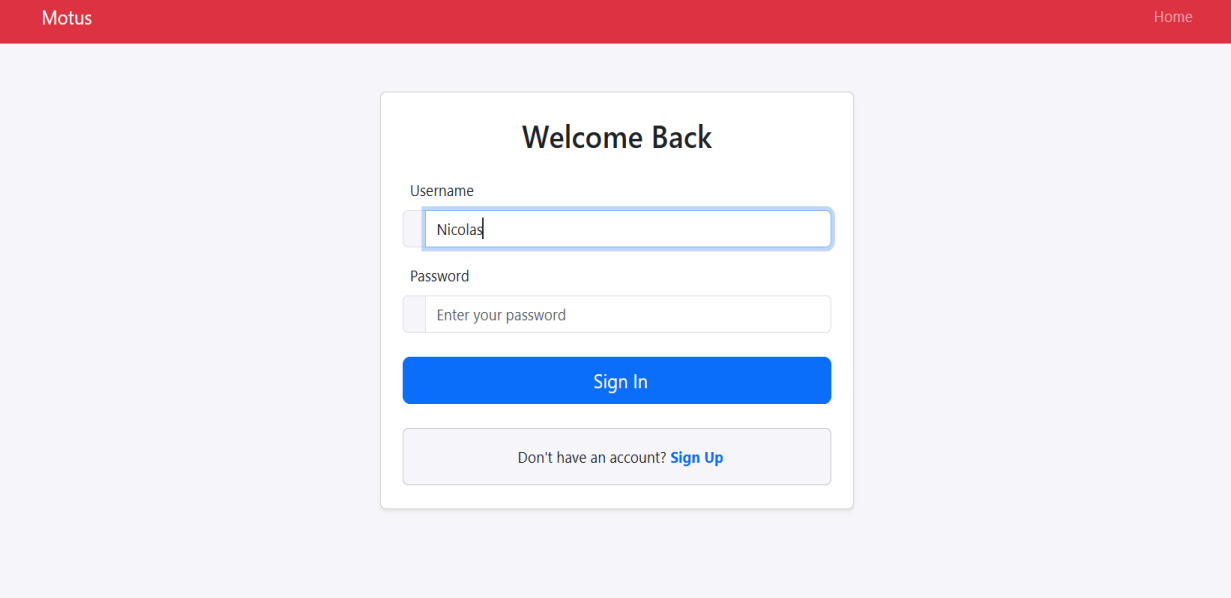
## ■ 5. Résultats obtenus

### Fonctionnalités implémentées avec succès :

- Sélection aléatoire d'un mot mystère depuis une liste
- Interface utilisateur fonctionnelle (champ de saisie, validation, affichage des essais)
- Algorithme de vérification des lettres avec code couleur :
  - Rouge pour les lettres bien placées
  - Jaune pour les lettres présentes mal placées
  - Gris pour les lettres absentes
- Limitation du nombre d'essais
- Messages de victoire ou d'échec
- Bouton de relance d'une nouvelle partie

### Captures d'écran (à intégrer dans ton dossier) :

- Page d'accueil ou écran principal



The screenshot shows a web application interface for a game called 'Motus'. At the top, there is a red header bar with the word 'Motus' on the left and a 'Home' link on the right. The main content area has a light purple background. In the center, there is a white rectangular box with a thin grey border. Inside this box, the title 'Welcome Back' is centered at the top. Below the title, there are two input fields: the first is labeled 'Username' and contains the text 'Nicolas'; the second is labeled 'Password' and contains the placeholder text 'Enter your password'. Below these fields is a prominent blue button with the text 'Sign In'. At the bottom of the box, there is a link that says 'Don't have an account? Sign Up'.

- Exemple de partie en cours

Motus

Accueil

## Bienvenue dans Motus

Essais

Essais restants : 5

m

a

r

c

h

e

Lettres bien placées

\_

\_

\_

\_

\_

e

Votre tentative :

Valider

Rejouer

- Écran de victoire ou de défaite

Motus

Accueil

## Bienvenue dans Motus

😞

Dommage c'est perdu. Le mot était : manuel

Essais

Essais restants : 0

m

a

r

c

h

e

m

a

n

g

e

r

m

a

n

d

i

e

m

a

t

u

r

e

m

a

j

e

u

r

m

a

t

u

r

e

Lettres bien placées

m

a

\_

u

\_

\_

### Difficultés rencontrées :

- Gestion des doublons dans les mots (ex : deux lettres identiques mal placées)
- Gestion propre des couleurs dans l'interface graphique
- Optimisation du code pour éviter les redondances



## Solutions apportées :

- Mise en place d'une logique de comptage pour éviter les répétitions non voulues
- Séparation claire entre logique de jeu et interface
- Refactorisation du code après chaque nouvelle fonctionnalité

## 6. Améliorations possibles

Voici quelques idées pour faire évoluer le projet dans le futur :

Amélioration	Description
Système de score	Calcul basé sur le nombre d'essais restants
Sauvegarde des parties	Enregistrement des scores dans un fichier JSON ou une base SQLite
Mode multijoueur	Deux joueurs en alternance ou contre une IA
Dictionnaire intégré	Pour vérifier la validité des mots saisis
Sélection de la longueur du mot	Choix entre mots de 5, 6, 7, ou 8 lettres
Version Web	Transposition du jeu en HTML/CSS/JavaScript avec React
Responsive design	Interface adaptable pour tablettes et smartphones

## 7. Conclusion

Ce projet m'a permis de mettre en pratique un grand nombre de compétences acquises lors de ma formation CDA. Il m'a notamment permis de :

- Concevoir un projet de bout en bout : de l'analyse des besoins jusqu'au produit final,
- Travailler en autonomie avec rigueur et organisation,
- Appliquer les concepts de programmation orientée objet et de modularité,
- Concevoir une interface graphique adaptée et fonctionnelle,
- Prendre en compte l'expérience utilisateur,
- Réaliser un projet à visée ludique tout en respectant des contraintes techniques.

Ce jeu, bien que simple en apparence, m'a confronté à des problématiques concrètes de développement logiciel, ce qui en fait une excellente mise en situation professionnelle.