Projet :

Développement d’un jeu de Motus en ligne

1. Contexte et objectifs:

Dans le cadre de ma formation de **Concepteur Développeur d’Applications**, j’ai développé une application de type *jeu de lettres Motus* accessible via une interface web.
Objectifs :

- Proposer un jeu interactif permettant aux utilisateurs de deviner un mot secret en un nombre limité de tentatives.
- Mettre en place une architecture modulaire basée sur des microservices.
- Permettre une expérience fluide grâce à une interface utilisateur ergonomique et réactive.

2. Environnement technique:

- **Langage** : Java (Spring Boot), HTML/CSS/JavaScript (Front)
- **Frameworks** : Spring Boot, Thymeleaf, Bootstrap
- **Architecture** : microservices (config-server, eureka-server, user, api, motus, webapp)
- **API externe** : Datamuse (récupération de mots)
- **Outils** : Git, Maven, IntelliJ IDEA, Docker

3. Démarche et réalisations:

- **Analyse du besoin** : rédaction d'un cahier des charges et étude des règles du jeu Motus pour définir les fonctionnalités principales (gestion des parties, validation des mots, feedback couleur, score).

- **Conception** : modélisation UML (cas d'utilisation, diagrammes de séquence) et conception de l'architecture microservices.

- **Développement** :

    - Service **API** : interroge Datamuse pour récupérer des mots.

    - Service **Motus** : contient la logique métier (génération du mot secret, comparaison des tentatives, gestion du score).

    - Service **User** : gestion des utilisateurs et sessions de jeu.

    - Service **WebApp** : interface utilisateur avec Spring Boot, Thymeleaf et Bootstrap.

    - Communication entre services via **Feign** et découverte grâce à **Eureka**.

- **Tests** : mise en place de tests unitaires sur la logique de comparaison de mots, tests d'intégration pour vérifier la communication entre services.

- **Déploiement** : conteneurisation avec Docker et configuration centralisée via config-server.

4. Compétences mises en œuvre

- Recueil et formalisation d'un besoin fonctionnel.

- Conception UML et architecture microservices.

- Développement d'une API et intégration d'un service tiers (Datamuse).

- Développement d'une interface web responsive.

- Mise en place de tests unitaires et d'intégration.

- Utilisation d'outils collaboratifs (Git, Maven, Docker).

5. Résultats et bilan:

L'application est fonctionnelle et permet de jouer à Motus en ligne avec un mot aléatoire généré à partir d'une API externe.
Points positifs :

- Application modulaire et évolutive grâce aux microservices.
- Interface intuitive et responsive.
- Intégration réussie d'une API externe.

Difficultés rencontrées :

- Mise en place de la communication entre microservices (problème de dépendances Feign/Eureka).
- Gestion de la cohérence entre mots récupérés et règles du jeu.

Solutions :

- Mise en place d'un service API centralisé.
- Filtrage et validation des mots avant leur utilisation dans la logique du jeu.

Améliorations possibles :

- Ajout d'un mode multijoueur.
- Système de score et de classement global.
- Sauvegarde de l'historique des parties.