Étude préalable - Partie jeu

2022

Projet pluridisciplinaire 1A, semestre 2,

groupe E32

- **Valentin CHANEL,**
- **Matéo DEMANGEON,**
- **Nicolas FRACHE,**
- **Victor VOISIN.**

Sommaire

Description fonctionnelle du projet

Présentation du projet

Concept de base et règles

La première apparition populaire de ce concept de jeu remonte probablement à l'émission télévisée américaine Lingo (1987 à 1988 puis 2007 à 2011).

Dans cette émission les candidats doivent trouver un mot mystère d'une longueur donnée en commençant avec pour seule information la première lettre du mot.

Puis à chaque tour, ils proposent un nouveau mot valide. S'en suit un processus de coloration suivant les règles suivantes (voir l'exemple de la figure 1) :

- Si une lettre du mot proposé est présente dans le mot mystère à la même place, celle-ci s'illumine en rouge.
- Si une lettre du mot proposé est présente dans le mot mystère, mais à une place différente, celle-ci s'illumine en jaune.
- Si une lettre du mot proposé n'est pas dans le mot mystère, celle-ci ne change pas de couleur.

On peut aussi ajouter que :

- Le nombre maximum d'occurrence d'une même lettre pouvant s'illuminer dans le mot proposé est égal au nombre d'occurrence de cette lettre dans le mot mystère. De plus, les lettres situées à la bonne position s'illuminent en priorité.
- Un mot est considéré valide s'il a le bon nombre de lettres et est correctement orthographié.

figure 1: Illustration d'une partie de Lingo avec le mot "Bagel", on peut noter que tous les mots intermédiaires sont valides.

Source : Wkimedia Commons, auteur : Aussie Evil

Ici, nous ne nous intéresserons pas plus aux règles propres aux jeux-télévisés (système d'équipes, conditions pour gagner la récompense etc), car cela sort du cadre de notre projet.

En France, c'est le jeu télévisé Motus qui a popularisé le concept depuis 1990.

Projet

Notre objectif ici est de recréer une version du jeu en ligne Wordle qui a connu une popularité fulgurante depuis le début de l'année 2022. Nous y ajouterons également de nouvelles fonctionnalités.

Liste des fonctionnalités du système

Notre application devra permettre de jouer au jeu Wordle classique, mais aussi donner la possibilité au joueur de personnaliser ses paramètres de jeu (nombre de tentatives, taille des mots). Ces paramètres seront sauvegardés via un système de compte, et un historique des parties sera conservé dans la base de données.

De plus, un tableau de score sera disponible pour permettre aux joueurs de comparer leurs performances.

On résume la liste des fonctionnalités principales ci-dessous.

Gestion compte

- Accès à l'application en mode anonyme sans compte
- Création d'un compte à l'aide d'un formulaire
- Connexion au compte
- Déconnexion du compte actuel
- Possibilité de rester connecté d'une session à l'autre sur le même appareil
- Possibilité de se connecter à plusieurs appareils avec le même compte
- Modification des informations d'un compte
- Suppression d'un compte, les informations sont alors supprimées de la base de données

Utilisation normale

- Choix du nombre de lettres dans le mot
- Choix de la difficulté

- Réinitialisation des paramètres de jeu pour revenir aux valeurs par défauts
- Saisie des lettres au clavier
- Saisie des lettres avec le clavier virtuel
- Validation du mot avec la touche entrée ou le bouton de l’interface
- Consultation du leaderboard (tableau des scores)
- Consultation de son historique de partie
- Continuer la partie en cours sur un autre appareil connecté au même compte
- Commencer une nouvelle partie à et abandonner automatiquement la partie en cours s’il y en a une

Cas d'utilisation et descriptions textuelles

Diagramme de cas d'utilisation général

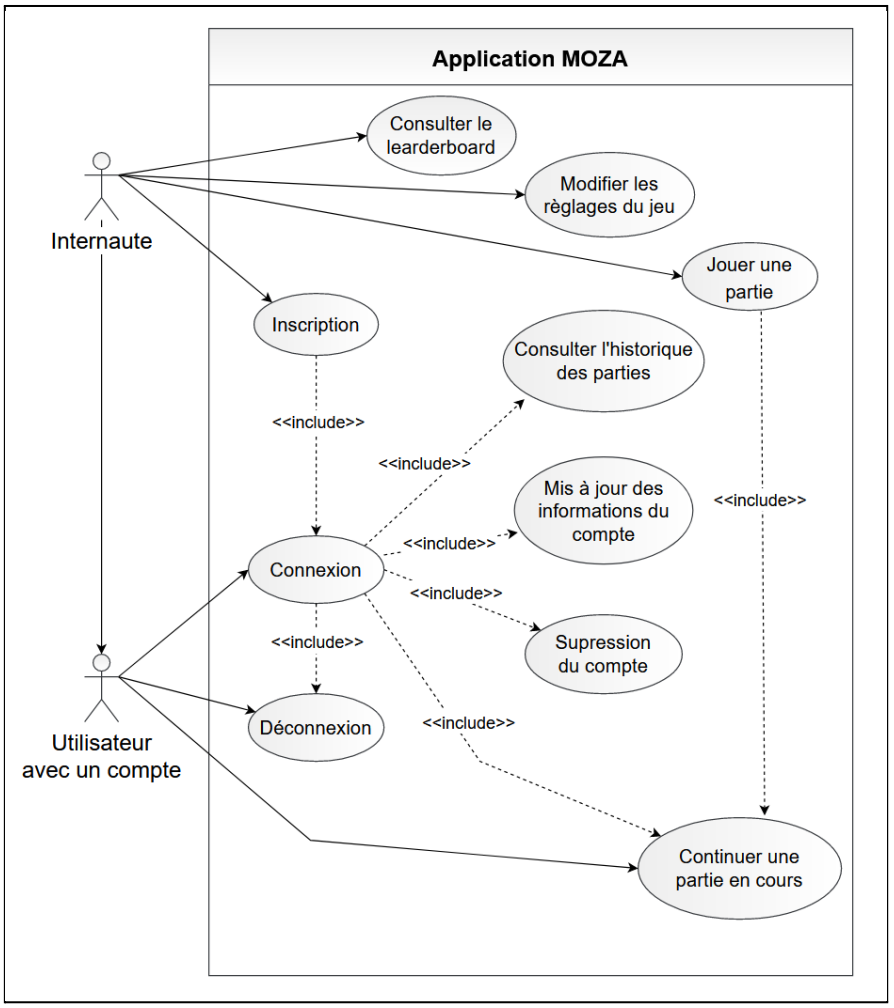


figure 2: Diagramme de cas d'utilisation général de l'application

Inscription

Précondition :

- L'internaute accède à l'application à partir du site web sur mobile ou PC.

Postcondition :

- L'utilisateur a maintenant un compte avec lequel il pourra se connecter ultérieurement.

Déroulement normal :

1. Un utilisateur, souhaite s'inscrire sur MOZA, pour cela il se rend sur l'application web via son navigateur.
2. Il se rend sur la rubrique "Inscription".
3. Le site lui demande alors d'entrer un pseudo, une adresse e-mail et un mot de passe. Il peut notamment ajouter une photo de sa bibliothèque comme avatar.
4. Suite à la validation de ses informations, celui-ci va recevoir un message de confirmation.
5. Le compte est créé

Variantes :

- (3) → Au moment de la validation le site indique à l'utilisateur un des messages d'erreur suivant :
    - Mot de passe non conforme
    - Nom d'utilisateur déjà utilisé
    - Adresse e-mail non-valide ou déjà utilisée
    - Erreur de communication avec les serveurs, merci de réessayer ultérieurement

Connexion

Précondition :

- L'utilisateur est inscrit dans l'application web

Postcondition :

- L'utilisateur accède à son espace personnel

Déroulement normal :

1. Un internaute souhaitant se connecter sur MOZA se rend sur l'application web via son navigateur
2. Il se rend sur la rubrique "Connexion"
3. Le site lui demande alors son adresse e-mail ainsi que son mot de passe afin de pouvoir s'identifier
4. L'utilisateur se trouve sur son espace personnel sinon il devra de nouveau donner ses identifiants jusqu'à validation par le système

Variantes :

- (3) → si l'utilisateur a oublié son mot de passe il peut demander à changer celui-ci à l'aide d'un lien reçu par mail lui permettant de changer son mot de passe
- (3) → si le mot de passe est faux ou le nom d'utilisateur inexistant le système demande à l'utilisateur de rentrer de nouveau ses identifiants

Reprendre une partie en cours

Préconditions :

- L'utilisateur est connecté
- Il commence une partie de jeu peu importe ses réglages

Postcondition :

- L'utilisateur peut continuer sa partie

Déroulement normal :

1. L'utilisateur accède à l'application avec n'importe quel appareil
2. Il se connecte à son compte
3. Le système détecte qu'une partie est en cours :
    a. Les paramètres de la partie sont chargés comme les paramètres actuels (nombre de lettres, de tentatives…)
    b. La partie en cours est chargée comme la partie actuelle
4. L'utilisateur reprend sa partie

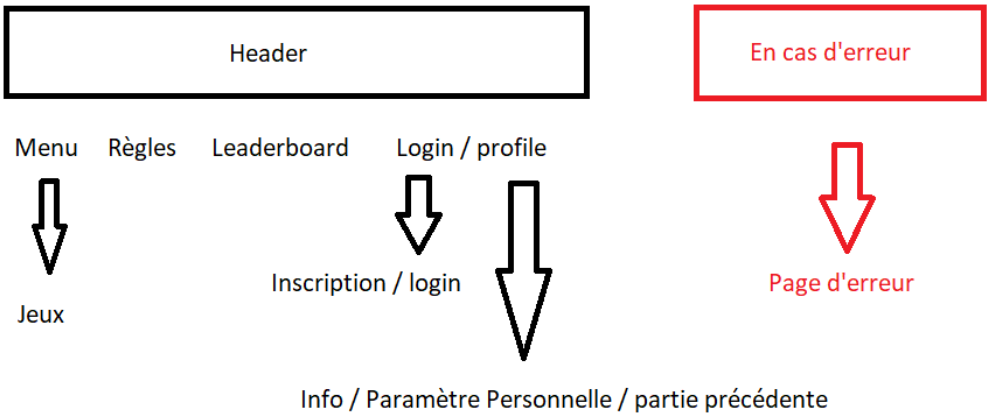
Variantes :

- (2) → L'utilisateur était préalablement connecté, il le reste donc en accédant à l'application

- (4) → L'utilisateur change les paramètres de partie, la partie en cours est donc supprimée.
- (4) → L'utilisateur préfère commencer une nouvelle partie, la partie en cours est donc écrasée par la nouvelle.

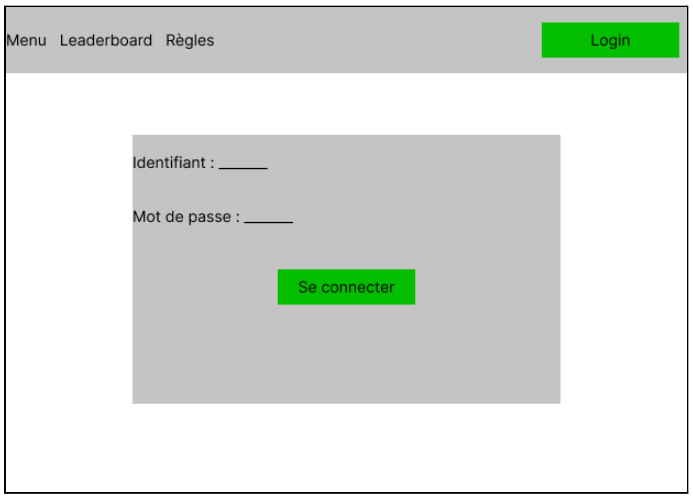
Structure du site et maquettes

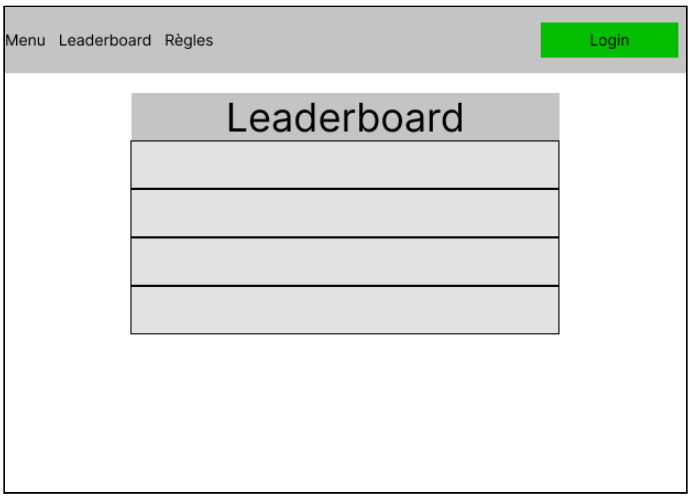
Structure du site

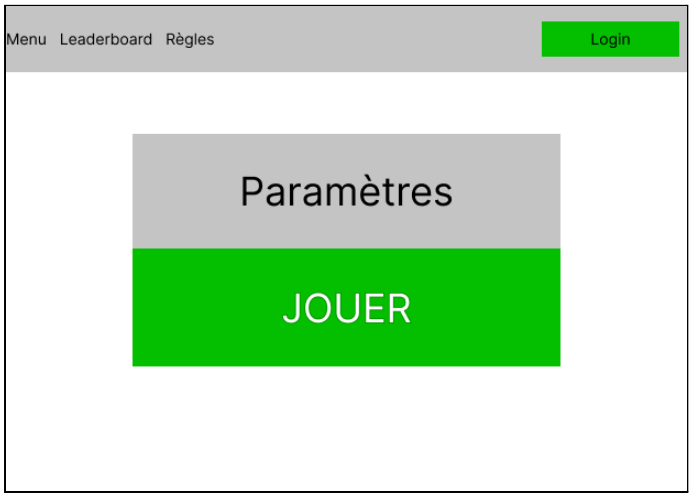


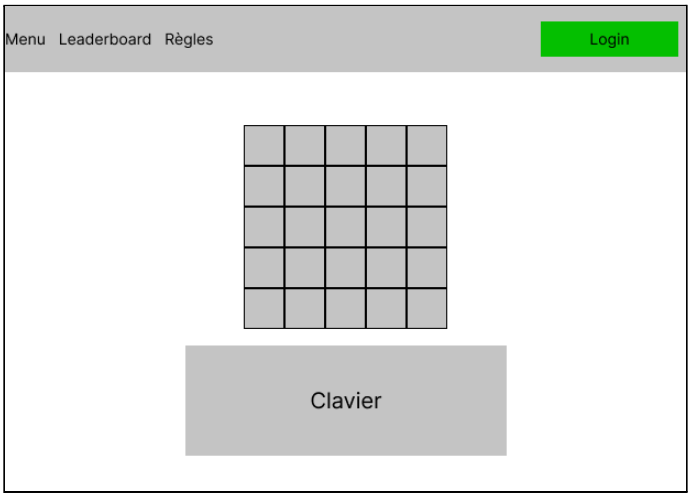
Interface utilisateur - maquettes

Voici les maquettes des différentes pages que nous comptons implémenter.









Mise en place technique

Modèle relationnel de la base de données

Une fois la liste des fonctionnalités de notre application mise en place, nous avons cherché à en déduire un modèle relationnel dont la représentation schématique est présentée ci-dessous (figure 3).

Nous avons volontairement omis le cas d'un utilisateur qui se connecte sans compte car il aura accès à un nombre de fonctionnalités restreint et rien ne sera stocké à son sujet sur la base de données.

Une partie a pour attributs non seulement le mot du dictionnaire qui représente la solution, mais également les paramètres qui lui sont appliqués et l'utilisateur qui y joue. De plus, une partie est constituée d'une liste de tentatives de la part du joueur. Ces tentatives contiennent donc un mot du dictionnaire, et sont liées à une partie.

Le système de paramètres (nombre de lettres et nombre d'essais possible) est conçu de manière à ce que chaque partie soit liée à une configuration de paramètres données. Et tant que l'utilisateur ne change pas ses paramètres, toute les nouvelles parties sont liées au même objet paramètre. Ce lien est permis par la sauvegarde des derniers paramètres utilisés par l'utilisateur avec l'attribut "parametreDernierePartie".

Passons maintenant à la reprise de la partie en cours. Pour intégrer celà, on commence par lier l'utilisateur à sa dernière partie jouée avec l'attribut "partieEnCours". Puis on vérifie si cette partie est ou non terminée avec l'attribut "estEnCours". Si la partie est terminée, le système peut ainsi en proposer une nouvelle. Et sinon, le système récupère l'ensemble des tentatives liées à cette partie pour permettre au joueur de continuer.

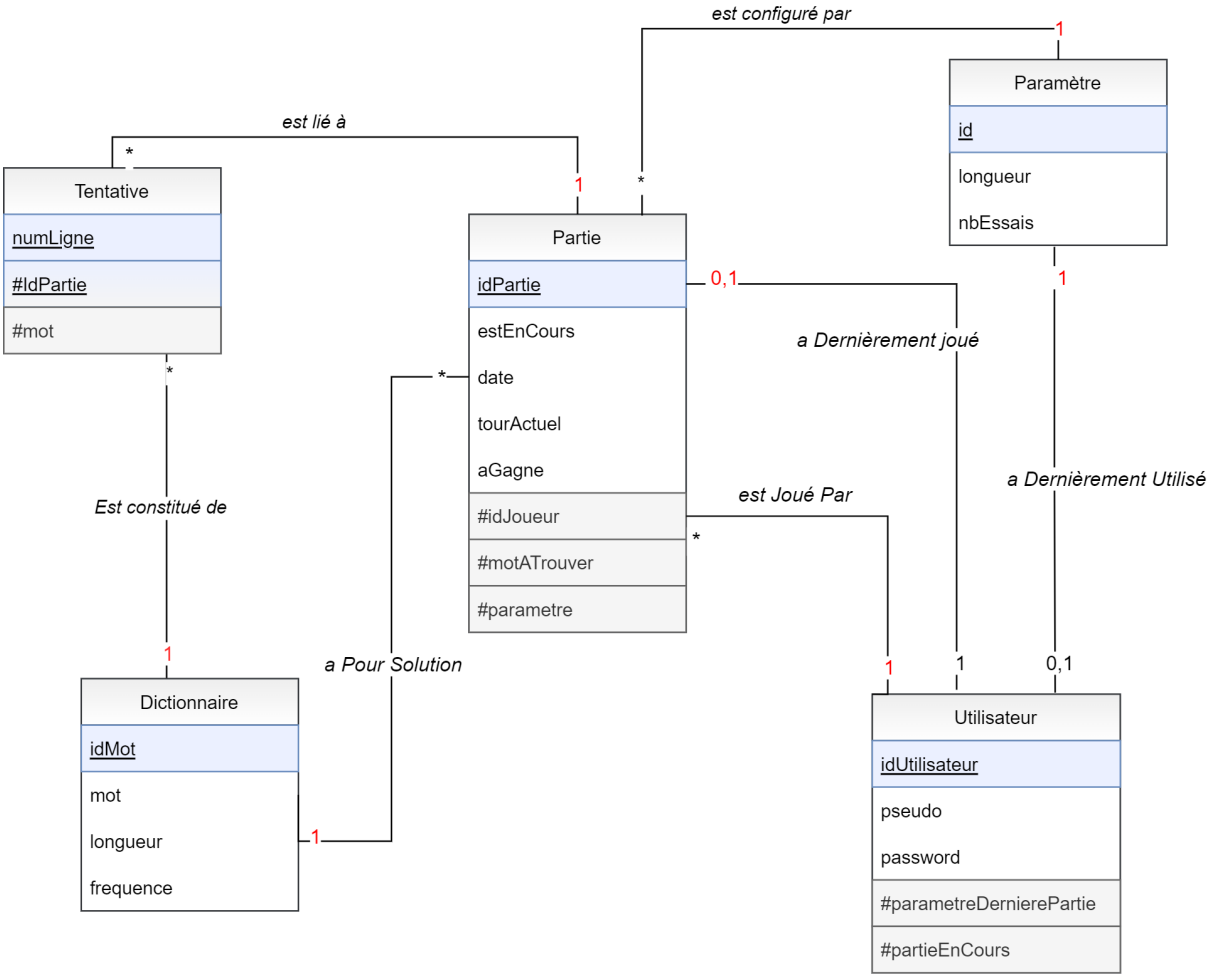


figure 3: Schéma du modèle relationnel de base de données

Interaction entre les couches de l'application

Choix et explications

Dictionnaire

La réalisation de l'état de l'art nous a permis de constater que la majorité des applications similaires existantes demandent au client (ordinateur de l'utilisateur) de télécharger un fichier contenant l'ensemble du dictionnaire, puis le front en javascript le manipule à partir de là.

Ici, nous avons fait un choix différent. Le dictionnaire n'est pas sous forme de fichier, mais intégré au sein de notre base de données. Ainsi, l'application javascript va interagir avec notre back pour avoir accès à la base de données.

Interaction serveur/client

Techniquement pour faire cela le principe est le même que pour le chargement de n'importe quelle page de notre site. Côté back, l'application Flask va recevoir une requête HTTP sur une route. Sauf qu'au lieu de renvoyer une chaîne de caractère contenant le html (généré avec le moteur de template Jinja2), nous allons ici envoyer une réponse en XML ou JSON avec les informations nécessaires.

Côté front, nous utilisons la technologie de requêtes AJAX (asynchronous JavaScript and XML) qui permet à notre application javascript d'envoyer une requête HTTP à un url du serveur. Concrètement, pour celà nous allons utiliser l'API Fetch qui offre une syntaxe plus simple et fonctionne avec un système de promesses.

Le diagramme de séquence ci-dessous (figure 4) permet de représenter chronologiquement les interactions entre les différents acteurs et couches du système pour un cas d'utilisation particulier. Ici, on s'intéresse à la situation d'un utilisateur qui joue une partie et qui saisit un mot valide (dans le dictionnaire) mais non-solution de la partie.

On peut noter que le diagramme ci-dessous a été réalisé avec l'outil open-source *PlantUML* qui permet de générer des diagrammes à partir d'un texte source. Le code correspondant à ce diagramme est donc disponible dans le répertoire *documents* de notre dépôt git (ici).

Diagramme de séquence - Saisie d'un mot et validation

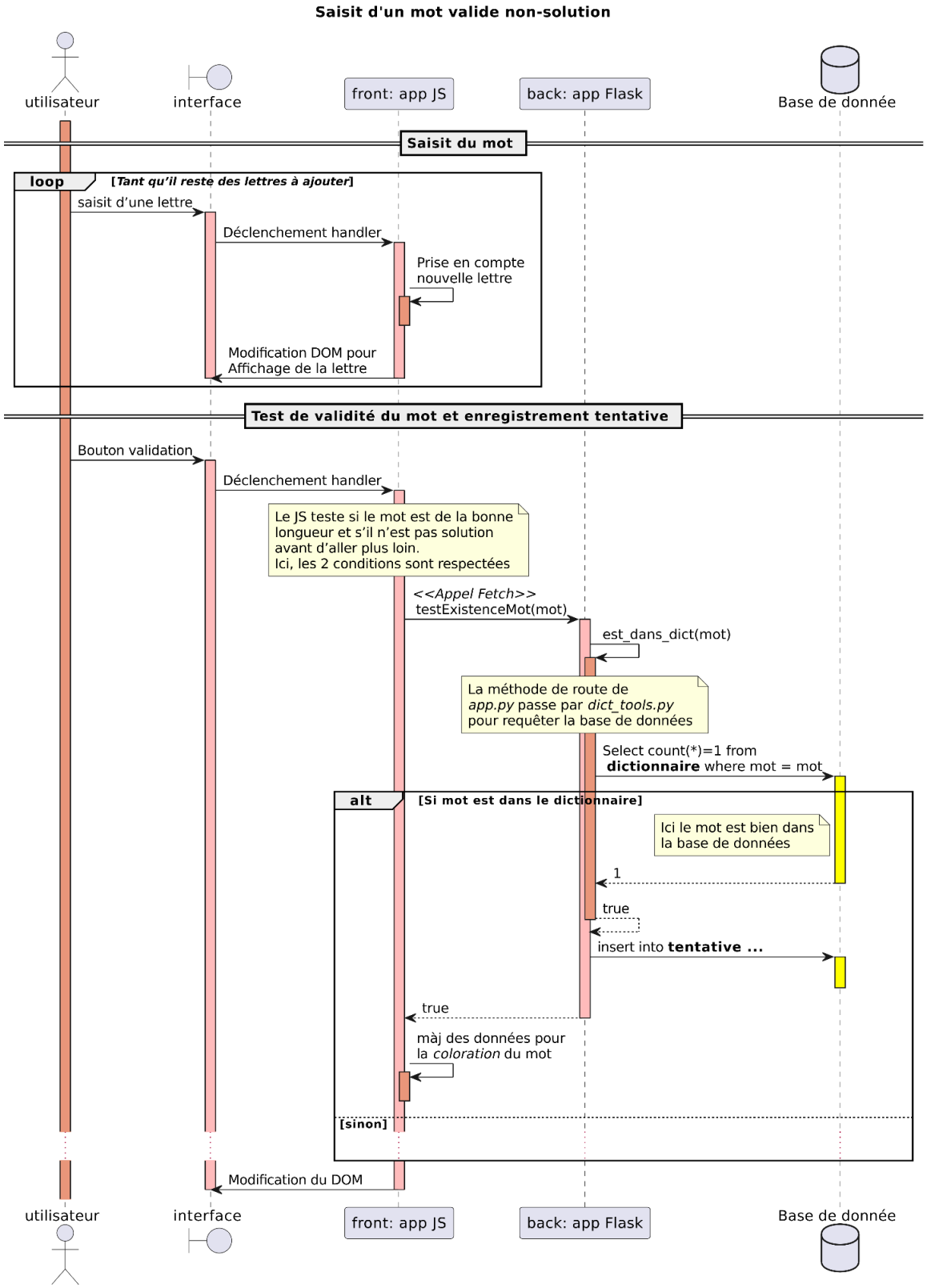


Figure 4 : Diagramme de séquence correspondant à la saisie d'un mot par le joueur puis à sa validation par le système pour déterminer s'il appartient ou non au dictionnaire. Code source.