

**Département Informatique**

 71 rue Peter Fink

01000 Bourg-en-Bresse

 04 74 45 50 59

*Roberjot Aloïs, Bodin Mathis, Caron Maxence, Namani Samuel*

Promotion 2023/2024

Mahjong calculator

Dossier d’améliorations

Fabrice Jaillet

Diplôme Universitaire de Technologie

Informatique

|  |
| --- |
| [Nom de la société] |
|  |

Parcours IEM :   
Informatique Et Mobilité

Table des matières

[I. Rappel : informations sur le projet 1](#_Toc157972255)

[I.1 Objectifs et contexte du projet 1](#_Toc157972256)

[I.2 État actuel du projet 1](#_Toc157972257)

[II. Détail des améliorations 1](#_Toc157972258)

[II.1 Nouvelles fonctionnalités 1](#_Toc157972259)

[II.1.1 Éditeur de règles 1](#_Toc157972260)

[II.1.2 Système de compte utilisateur 1](#_Toc157972261)

[II.2 Ergonomie / Expérience utilisateur 2](#_Toc157972262)

[II.2.1 Version responsive du site (PC) 2](#_Toc157972263)

[II.2.2 Supprimer le drag&drop 2](#_Toc157972264)

[II.2.3 Améliorer la détection des tuiles 2](#_Toc157972265)

[II.2.4 Afficher les résultats de la détection 3](#_Toc157972266)

[II.2.5 Afficher le détail des calculs 3](#_Toc157972267)

[II.3 Qualité logicielle 3](#_Toc157972268)

[II.3.1 Refactoriser le code 3](#_Toc157972269)

[II.3.2 Rédiger la documentation 4](#_Toc157972270)

[II.4 Accessibilité 4](#_Toc157972271)

[II.4.1 Vérifier le contraste pour les textes 4](#_Toc157972272)

[II.4.2 Vérifier que chaque image possède un texte alternatif 4](#_Toc157972273)

[II.4.3 Ajouter des options de navigation au clavier 4](#_Toc157972274)

[II.5 Impact environnemental 4](#_Toc157972275)

[II.5.1 Réduire la taille des images 4](#_Toc157972276)

[II.6 Sécurité 4](#_Toc157972277)

[II.6.1 Modification de l’architecture (version PC) 4](#_Toc157972278)

[II.6.2 Redirection automatique PHP (version PC) 5](#_Toc157972279)

[II.6.3 Hachage des mots de passe 5](#_Toc157972280)

[II.6.4 Blocage des connexions répétées 5](#_Toc157972281)

[II.7 Déploiement 5](#_Toc157972282)

[II.7.1 Rédiger les documents d’installation 5](#_Toc157972283)

[II.7.2 Tests unitaires 5](#_Toc157972284)

[III. Récapitulatif 6](#_Toc157972285)

# Rappel : informations sur le projet

## Objectifs et contexte du projet

L'idée de ce projet est de proposer une solution pour faciliter le calcul des points à la fin d'une partie de Mahjong : on prend une photo de sa main, et le logiciel nous renvoie le nombre de points !

## État actuel du projet

Pour ce projet, nous avons choisi de développer un site web (comme interface utilisateur), qui communique avec un serveur Java (qui effectue les calculs). Voici les fonctionnalités actuelles :

* Saisie des tuiles manuelle
* Saisie des tuiles par photo
* Saisie des bonus (vents et fleurs)
* Affichage et stockage du score (historique)

# Détail des améliorations

## Nouvelles fonctionnalités

### Éditeur de règles

Ajouter la possibilité à l’utilisateur de personnaliser ses règles. La version actuelle de site contient un nombre limité de règles, qui son personnalisables seulement en modifiant le code du serveur. L’objectif est donc de proposer une interface permettant de créer ses propres règles :

* Sélectionner une liste de tuiles et lui associer un nombre de points (grands jeux)
* Modifier les règles de base (suite, paire…)
* Combiner des règles de base pour créer des règles complexes. Par exemple, on combine la règle Suite avec la règle Bamboo, et on lui associe un nombre de points (par ex 5pts). Ainsi, toutes les suites de bambou auront une valeur de 5 points.

### Système de compte utilisateur

Ajouter une logique de compte utilisateur : un compte permettrait de sauvegarder son historique des scores, ainsi que ses règles du jeu personnalisées.

Ajouts requis :

* **Page de connexion / inscription :** ajouter de nouvelles pages sur le site web permettant de se connecter ou de s’inscrire.
* **Base de données :** Relier le serveur à la base de données contenant les informations sur les utilisateurs (historique des scores, règles du jeu…).

## Ergonomie / Expérience utilisateur

### Version responsive du site (PC)

Le site actuel est optimisé pour les petits écrans (smartphone). Bien qu’il fonctionne aussi pour les grands écrans (PC), l’expérience utilisateur est impactée négativement. L’objectif est donc d’adapter le CSS pour ajouter le support des grands écrans, en proposant une nouvelle disposition et une meilleure gestion de l’espace.

### Supprimer le drag&drop

Actuellement, il est possible d’ajouter ses tuiles par un simple drag&drop depuis le tiroir vers un emplacement libre. Cependant, la fonctionnalité est volontairement désactivée sur les écrans tactiles (smartphones). Or, le site est développé pour mobile, c’est donc une fonctionnalité inutile qui peut être supprimée.

Réactiver le drag& drop sur mobile n’est pas une bonne solution pour l’expérience utilisateur, car sur un PC, on utilise la molette de la souris pour scroller dans la page. Or, sur téléphone, c’est la même action qui est utilisée pour scroller que pour déplacer les tuiles. Ainsi, lorsque l’utilisateur glisse le doigt sur son écran, on ne peut pas savoir s’il veut déplacer une tuile, ou simplement scroller dans la page.

### Améliorer la détection des tuiles

L’algorithme actuel pour détecter les tuiles sur une image utilise une combinaison d’opérations simples (niveau de gris, flou, seuillage, érosion…) pour obtenir une maque binaire indiquant la position des tuiles. Dans de mauvaises conditions d’éclairage, il est donc fréquent que des zones de l’image qui ne sont pas des tuiles soient tout de même considérées comme tel. Pour éviter ce genre de problèmes, deux solutions sont possibles :

#### Améliorer l’algorithme existant

Il existe de nombreuses manières d’améliorer l’algorithme actuel. Voici quelques exemples :

* **Soustraction de fond :** obtenir la couleur du fond (par l’utilisateur ? par couleur moyenne ?) et la soustraire à l’image originale pour ne garder que les tuiles
* **Restriction sur les contours :** ignorer les contours qui sont trop petits ou trop grands

#### Utiliser une autre méthode

Une autre méthode consisterait à entrainer ou utiliser un modèle d’IA capable de détecter et reconnaître directement les tuiles de mahjong sur une photo.

Il est aussi possible d’utiliser une IA pour supprimer l’arrière-plan de l’image, puis conserver notre algorithme pour la suite, c’est-à-dire pour trouver le nom d’une tuile à partir de son image.

### Afficher les résultats de la détection

Lorsque l’utilisateur envoie une photo de son jeu, les tuiles sont directement ajoutées à sa main à la réception de la réponse du serveur. Cependant, le détail de la détection n’est pas affiché, et l’utilisateur doit identifier les éventuelles erreurs (en comparant avec ses propres tuiles) pour les corriger. Pour rendre ce processus plus simple, une bonne solution peut être d’afficher une fenêtre popup qui indique pour chaque tuile de l’image, quelle tuile du jeu lui a été associée.

### Afficher le détail des calculs

Sur la page du score, un bon moyen d’améliorer l’expérience utilisateur est d’afficher le détail des calculs permettant d’obtenir le score affiché. Cela permettrait à l’utilisateur d’identifier facilement quelles combinaisons rapportent le plus de points, et ainsi aider les joueurs à s’améliorer.

## Qualité logicielle

### Refactoriser le code

#### Page d’ajout des tuiles

La page d’ajout des tuiles a été la première page avec un contenu Javascript conséquent. Cette page a été créée au début du projet, lorsque l’architecture globale du javascript n’était pas totalement mise en place. Il y a donc de nombreux changement à effectuer :

* Bien séparer le code dans plusieurs fichiers
* Réécrire le code des classes Tile, Slot et Hand, en supprimant les fonctions inutiles
* Ajouter une classe Drawer
* Ajouter une classe supérieure (TileManager), qui assurera les échanges de données entre Hand et Drawer
* Inclure la gestion de évènements à l’intérieur de la classe

#### Méthode de génération HTML

Dans le projet, le Javascript a été utilisé de nombreuses fois pour générer du code HTML de manière dynamique, à l’aide de classes spécialement conçues à cet effet. Cependant, il existe une solution Javascript native qui possède plus de fonctionnalités et permet une gestion plus facile : document.createElement. Pour continuer le projet, il est donc préférable d’utiliser cette solution qui possède de nombreux avantages :

* **Pointeur direct sur le document HTML :** une modification de l’objet JS aura un effet direct sur le document HTML
* **Gestion des évènements :** la fonction addEventListener est disponible pour ce type d’objet, permettant de simplifier significativement le code
* **Plus de fonctionnalités :** l’API document.createElement permet par exemple un attribut style qui permet de modifier en temps réel l’apparence du HTML

### Rédiger la documentation

Rédiger la documentation pour les fichiers Java et Javascript afin de garder un code compréhensible et maintenable.

## Accessibilité

### Vérifier le contraste pour les textes

S’assurer que tous les textes sont lisibles et que le contraste est suffisamment important.

### Vérifier que chaque image possède un texte alternatif

S’assurer que toutes les images possèdent une description alternative.

### Ajouter des options de navigation au clavier

Dans le cas d’une navigation sur PC, rendre les boutons d’action et les liens sélectionnables au clavier.

## Impact environnemental

### Réduire la taille des images

Lors de l’envoi au serveur, réduire la taille des images permet de minimiser les échanges et des réduire la durée de traitement sur le serveur. Cela contribue donc à une réduction de la consommation d’énergie, et à une amélioration de l’expérience utilisateur. Il faut cependant réduire la taille raisonnablement afin de ne pas impacter la reconnaissance des tuiles.

Il est aussi possible de compresser les images avant l’envoi, en utilisant le format AVIF (si supporté par le navigateur), ou en utilisant une méthode d’encodage différente de Base64.

## Sécurité

### Modification de l’architecture (version PC)

Pour une meilleure sécurité, il est préférable de modifier l’architecture du site en utilisant un dossier webRoot. Ce dossier contient tous les fichiers accessibles publiquement tels que le CSS, les images, les pages statiques HTML. Les autres fichiers permettant de gérer la logique interne du site sont situés dans le dossier parent, qui est inaccessible depuis internet. Ainsi, les informations sensibles comme que l’adresse de la base de données sont protégées et inaccessibles depuis l’extérieur. Des méthodes de routages (permettant de modifier les url) doivent également être mises en place afin de ne pas dévoiler l’architecture de site.

### Redirection automatique PHP (version PC)

Lorsqu’un utilisateur n’est pas connecté, il ne doit pas accéder aux pages impliquant de connaitre les informations sur le joueur. Par exemple, la page permettant d’afficher l’historique d’un joueur a besoin de savoir qui est le joueur connecté afin d’envoyer une requête à la base de données. Il faut donc garantir que le joueur est connecté lorsqu’il arrive sur cette page. En combinant cette option avec l’architecture webRoot, il sera impossible pour un utilisateur non connecté d’accéder aux pages nécessitant une connexion.

### Hachage des mots de passe

Les requêtes vers la base de données ne peuvent pas être exécutées directement depuis le Javascript et doivent obligatoirement passer par le serveur. Il est donc important de hacher les mots de passe avant de les envoyer au serveur.

### Blocage des connexions répétées

Étant donné que le code source est public, il est actuellement très simple de faire un script qui envoie des requêtes en boucle au serveur jusqu’à le mettre hors service. Une bonne solution serait donc de détecter et de bloquer les connexions répétées d’une même adresse IP, afin d’assurer la sécurité et la stabilité du serveur.

## Déploiement

### Rédiger les documents d’installation

Compléter le fichier readme.md sur git, et ajouter une documentation, un guide d’utilisation et un manuel d’installation expliquant notamment :

* Comment installer son serveur ?
* Comment connecter le site au serveur ?
* Comment utiliser sa propre base de données ?

### Tests unitaires

Mettre en place des test unitaires automatiques pour assurer le bon fonctionnement du site tout au long du développement.

# Récapitulatif

|  |  |  |  |
| --- | --- | --- | --- |
| **Nom de la tâche** | **Catégorie** | **Difficulté** | **Durée** |
| Éditeur de règles | Fonctionnalités | 17 | Très long |
| Comptes utilisateur | Fonctionnalités | 5 | Moyen |
| Version responsive | Expérience utilisateur | 3 | Long |
| Suppression drag&drop | Expérience utilisateur, qualité du code | 7 | Moyen |
| Amélioration détection des tuiles | Expérience utilisateur | 5 | Rapide |
| Détails de la détection des tuiles | Expérience utilisateur | 7 | Moyen |
| Détails des calculs | Expérience utilisateur | 11 | Long |
| Refactorisation de la page d’ajout des tuiles | Qualité du code | 7 | Long |
| Changement des fonctions de génération HTML | Qualité du code | 3 | Moyen |
| Rédaction de la documentation | Qualité du code | 1 | Long |
| Vérifier l’accessibilité | Accessibilité | 1 | Rapide |
| Réduire la taille des images | Impact environnemental | 3 | Rapide |
| Architecture webRoot | Sécurité | 5 | Moyen |
| Redirection auto PHP | Sécurité | 5 | Moyen |
| Hachage des mots de passe | Sécurité | 3 | Rapide |
| Blocage des connexions répétées | Sécurité | 5 | Moyen |
| Rédaction du manuel utilisateur | Déploiement | 1 | Long |
| Tests unitaires | Déploiement | 3 | Moyen |