



Rapport Projet Développement Java

YNOV CAMPUS

CHAUDESSOLLE Lorenzo, JEANNOT Ferréol, LACUBE Romain,
MONTIGNEAUX Adrien
YNOV | 2 RUE DU CORBUSIER

Table des matières

TABLES DES FIGURES	2
Introduction	3
Présentation des acteurs du projet.....	3
Cahier des charges.....	3
Spécifications de l'application	3
Modélisation de la base de données.....	4
Diagrammes de classes métier	5
Diagrammes de cas d'utilisation	7
Diagrammes de séquences	8
Algorithme mis en place	13
Algorithme du mode jeu : « Ordinateur choisi, Joueur cherche »	13
Algorithme 'le joueur génère, le programme cherche'.....	15
Règles du jeu	18
Mode d'emploi et déploiement du jeu.....	18
Planification.....	21
Charte de développement	22
Bilan	22
Annexes	23
Fiche d'itération 1	23
Fiche d'itération 2	25
Fiche d'itération 3	27
Compte rendu de réunion 1	29
Compte rendu de réunion 2.....	30
Compte rendu de réunion 3.....	31
Tests nominaux.....	32

TABLES DES FIGURES

Figure 1: Modèle Conceptuel des Données	4
Figure 2: Modèle Logique des Données	4
Figure 3: Diagramme des classes	6
Figure 4: Diagramme de cas d'utilisation	7
Figure 5: Diagramme de séquence : Inscription d'un utilisateur	8
Figure 6: Diagramme de séquence: Authentification d'un utilisateur	8
Figure 7: Diagramme de séquence : Choix mode de jeu	9
Figure 8: Diagramme de séquence : Deviner la bonne combinaison	9
Figure 9: Diagramme de séquence : la machine doit trouver la bonne combinaison	10
Figure 10: Diagramme de séquence : Consulter les règles de jeu	10
Figure 11: Diagramme de séquence : Afficher le profil	11
Figure 12: Diagramme de séquence : modifier profil	11
Figure 13: Afficher les statistiques	12
Figure 14: Diagramme de séquence : Quitter application	12
Figure 15 : Menu de bienvenue	18
Figure 16: Menu d'inscription	19
Figure 17: Menu principal	19
Figure 18: Action le joueur joue	19
Figure 19: L'ordinateur joue	20
Figure 20: Menu Afficher profil	20
Figure 21: Modification profil	20
Figure 22: Page statistiques	21
Figure 23: Affichage du gantt	21
Figure 24: Fiche itération 1	23
Figure 25: Répartition des taches	24
Figure 26: Fiche itération 2	25
Figure 27: Répartition des taches	26
Figure 28: Fiche itération 3	27
Figure 29: Répartition des taches	28
Figure 30: Compte rendu de réunion du 20/12/2017	29
Figure 31: Compte rendu de réunion du 21/12/2017	30
Figure 32: Compte-rendu du 20/02/2018	31

Introduction

La société Hasbro souhaite rendre son fameux MasterMind disponible sur smartphone. En attendant la réalisation de la maquette par le prestataire graphiste, cette dernière nous a contacté pour réaliser son jeu sans interface utilisateur en utilisant le langage Java.

Le but de ce projet est donc logiquement de répondre aux attentes d'Hasbro afin de réaliser une première version du MasterMind jouable.

Présentation des acteurs du projet

La société Hasbro constitue le client de ce projet. Mr Samir AZZAG est par conséquent le contact entre le prestataire et la marque durant le réalisation et l'évaluation de ce projet.

En tant que prestataire auprès de la société Hasbro, notre équipe de développement sera composé de 4 membres à savoir Lorenzo, CHAUDESSOLLE, Ferréol JEANNOT, Romain LACUBE, Adrien MONTIGNEAUX.

Cahier des charges

Spécifications de l'application

L'application doit répondre à plusieurs critères. Les voici, lister sous forme de points.

Création du jeu MasterMind :

1. Version sans UI graphique (application console)
2. Inscription / Connexion
3. Version 1 : l'application génère, le joueur joue
4. Version 2 : le joueur génère, l'application cherche
5. Base de données pour stocker les informations des joueurs
6. BP / MP ≤ 5
7. Affichage des résultats maximum après 10 coups
8. Erreurs de saisie gérées (pour les versions 1 et 2)
9. Méthode bruteforce pour la version 2.
10. Gestion des doublons.
11. Statistiques du joueur concernant le jeu.
12. Restriction : accès au jeu, uniquement après l'authentification de ce dernier.
13. Optionnel : export des statistiques au format CSV

Modélisation de la base de données

MCD

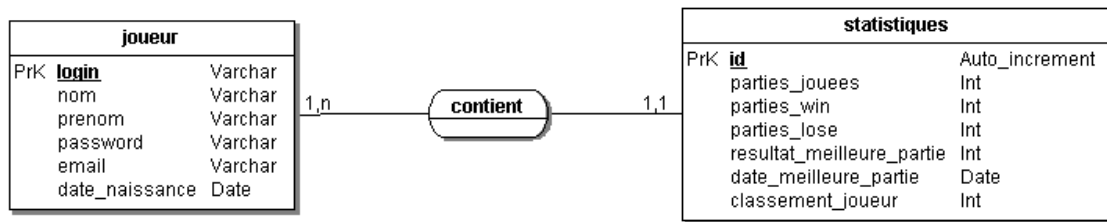


Figure 1: Modèle Conceptuel des Données

MLD

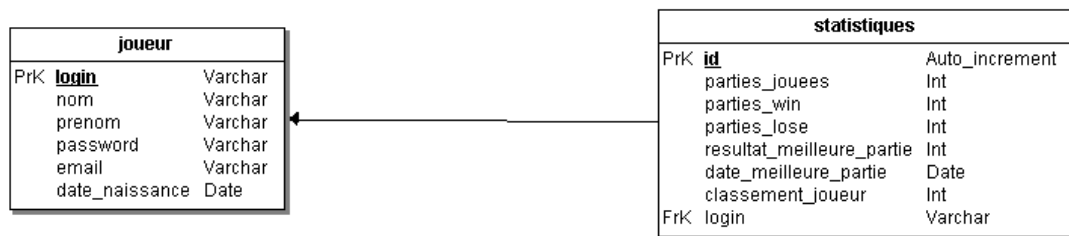


Figure 2: Modèle Logique des Données

La base de données est constituée de 2 tables distinctes :

La table **joueur** comportant les caractéristiques d'un utilisateur : un login, un nom, un prénom, un mot de passe, un email ainsi de la date de naissance de l'utilisateur.

Chaque joueur à ses propres statistiques de jeu.

La table **statistique** est donc liée à la table joueur et contient les statistiques du joueur.

Cela correspond aux nombres de parties jouées, perdues,

gagnées, au score de la meilleure partie, la date de la meilleure partie ainsi que la place du joueur dans le classement.

Diagrammes de classes métier

Nous avons choisi afin de répondre de la meilleure des façons aux demandes de la société Hasbro, de mettre en place le modèle MVC (Model, View, Controller).

Nous avons de ce fait déclaré deux classes abstraites : une classe « Controller » et une classe View.

La classe Controller a pour but de faire le lien entre les informations envoyés à l'utilisateur par le biais des différentes vues et la base de données. Ce sont aussi toutes les classes qui seront héritées de cette classe abstraites, qui effectueront les tâches de calculs employant les différents algorithmes que nous allons mettre en place. Ces classes héritées de la classe Controller sont en autres « GameUserplayer » ou encore la classe « UpdateProfil ».

La classe Abstraite Views sera la base de toutes les différentes vues qui seront des classes héritées de la classe Views. Bien évidemment, le programme est un programme en console, les vues sont donc très réduites. Néanmoins, le découpage en Modèle / vue / contrôleur permet d'avoir uniquement à modifier la vue pour changer totalement l'aspect graphique du programme.

La classe « DBConnection » compose le Model du MVC c'est-à-dire la base de données et l'interaction avec cette dernière. C'est par le moyen de cette classe que le contrôleur effectuera des actions en rapport avec la base de données, par exemple enregistrer les données de l'utilisateur, afficher ces statistiques ou encore l'inscrire et l'authentifier.

Enfin la classe « MasterMind » est la classe qui permet d'appeler le programme et qui permet de rentre le jeu exécutable par l'utilisateur.

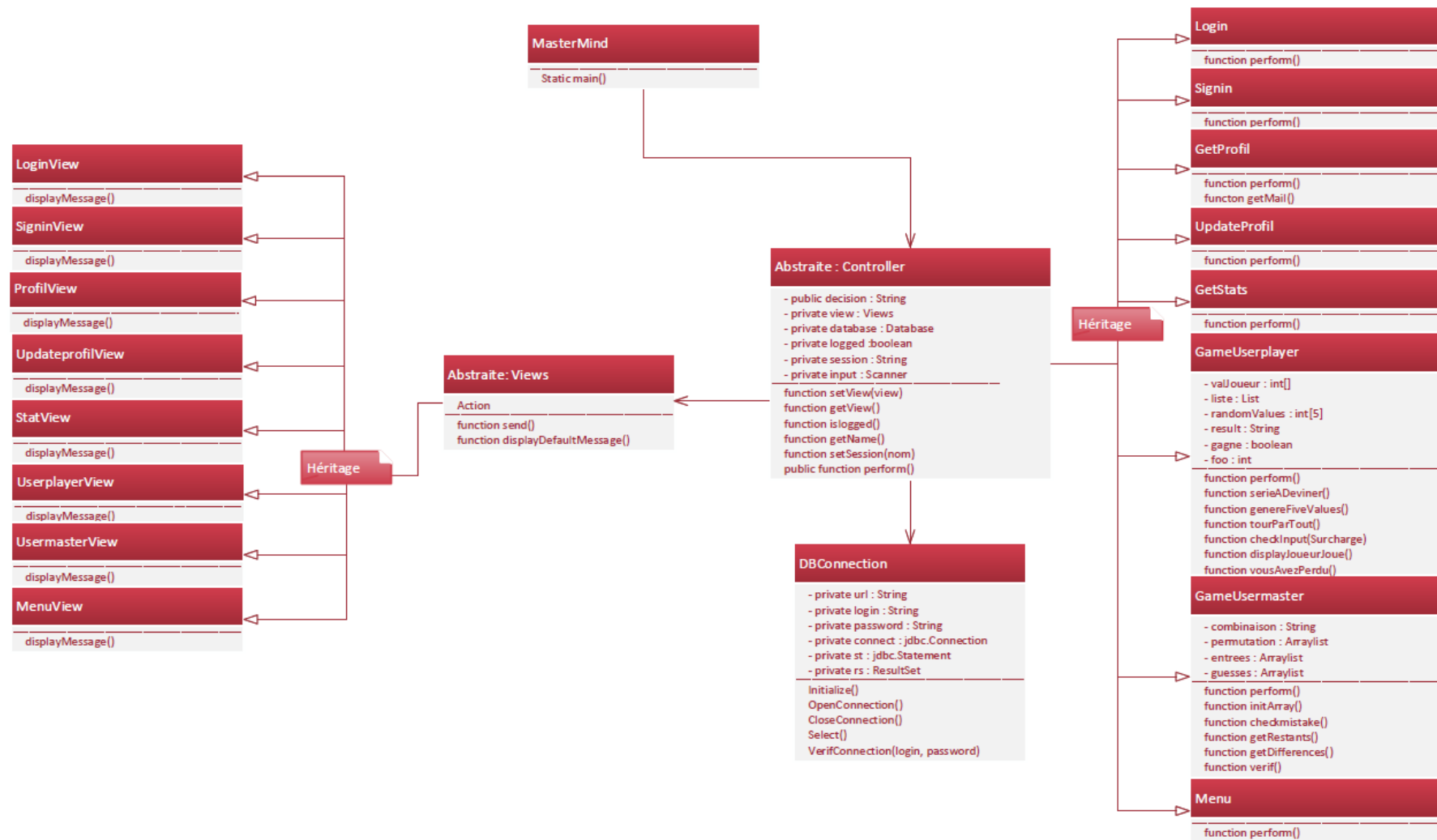


Figure 3: Diagramme des classes

Diagrammes de cas d'utilisation

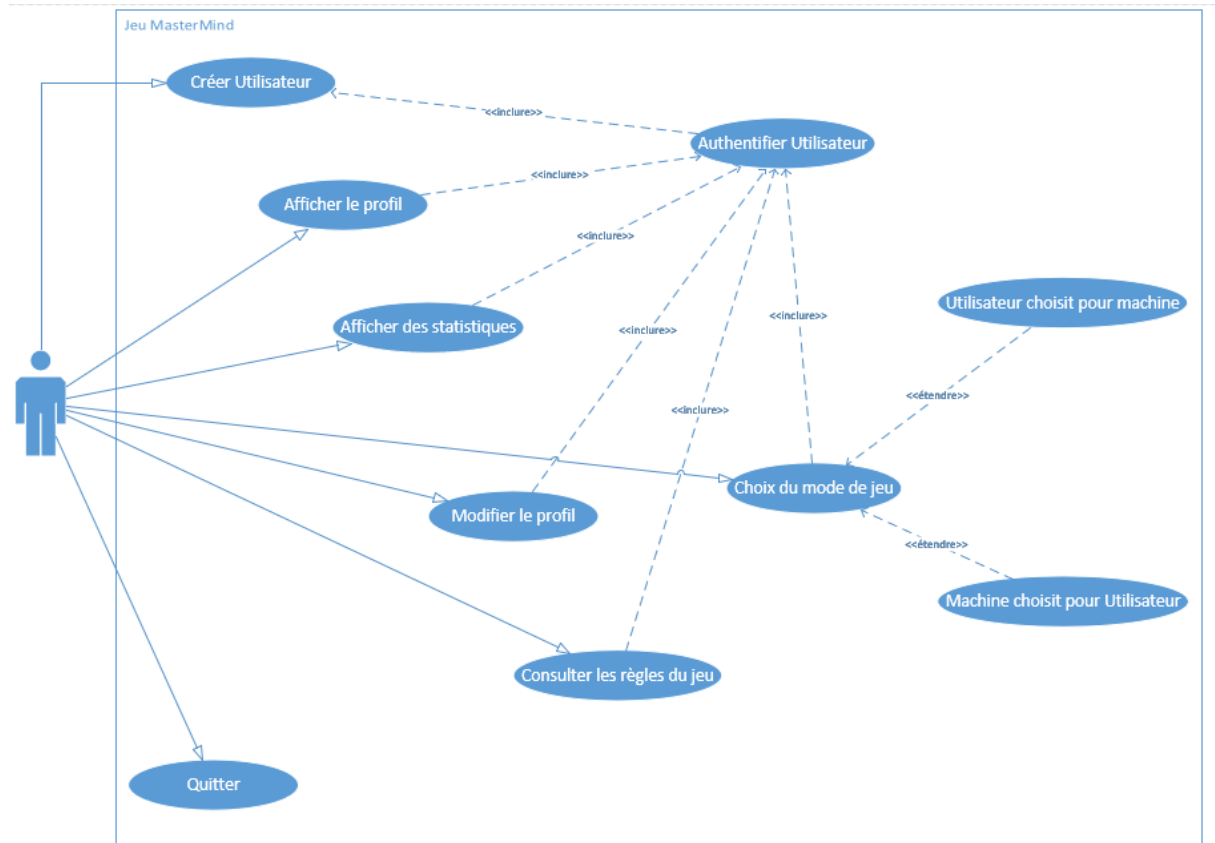


Figure 4: Diagramme de cas d'utilisation

Ce diagramme représente les différents cas d'utilisations présents au cours de l'utilisation de l'application. Nous pouvons en distinguer 10 :

- Créer un utilisateur : Permet de créer un utilisateur en demandant à ce dernier de saisir certaines informations à savoir son nom, prénom, email, date de naissance, son login et son mot de passe.
- Authentifier un utilisateur : l'utilisateur saisit son login et son mot de passe pour accéder à l'application.
- Afficher le profil d'un utilisateur : Permet de voir le profil de l'utilisateur et toutes les informations qui y sont contenues.
- Modifier le profil d'un utilisateur : L'utilisateur peut modifier les informations de son profil.
- Afficher les statistiques d'un utilisateur : Permet de consulter les résultats des parties précédentes auxquelles l'utilisateur a pris part.
- Consulter les règles du jeu : Permet de lire les règles des différents mode de jeu.
- Choix du mode de jeu : L'utilisateur choisit le mode jeu.
- Premier mode de jeu : la machine choisit et l'utilisateur devine
- Deuxième mode de jeu : l'utilisateur choisit, la machine devine
- Quitter : Permet de quitter l'application.

Diagrammes de séquences

Chaque cas d'utilisation de l'application est décrit ci-dessous par son diagramme de séquence. Ce dernier exprime l'ensemble des actions de déroulant dans un ordre chronologique lors de la réalisation du cas d'utilisation.

Inscription d'un utilisateur

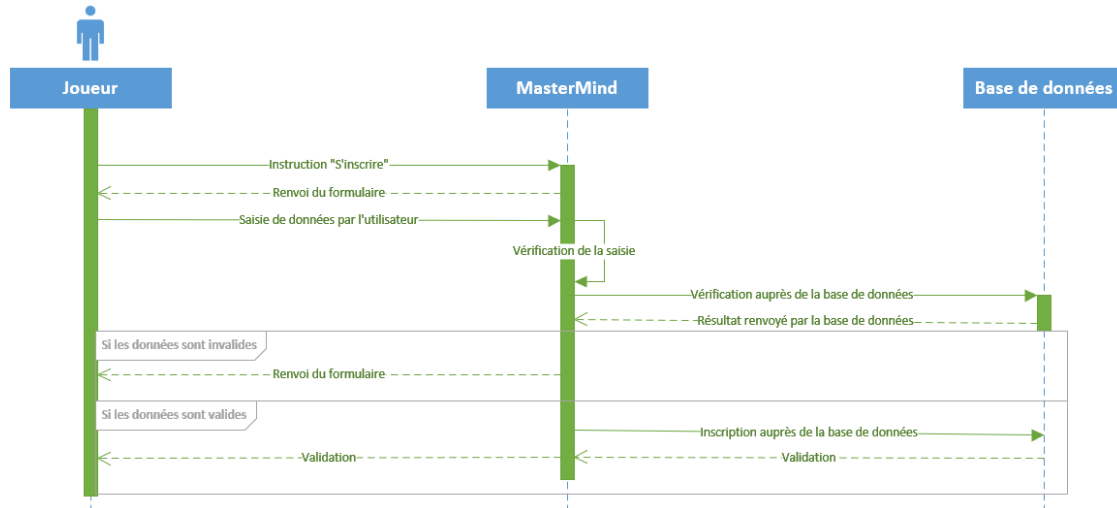


Figure 5: Diagramme de séquence : Inscription d'un utilisateur

Authentification d'un utilisateur

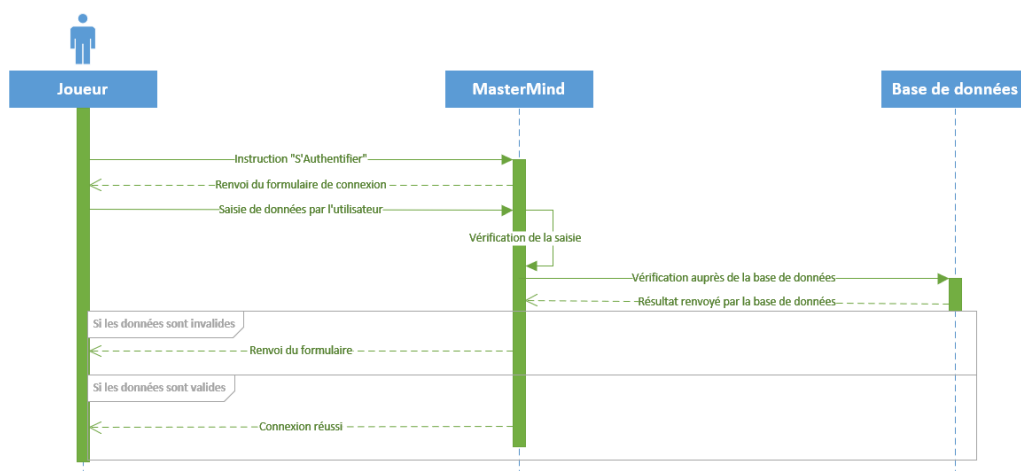


Figure 6: Diagramme de séquence : Authentification d'un utilisateur

L'utilisateur choisit son mode de jeu

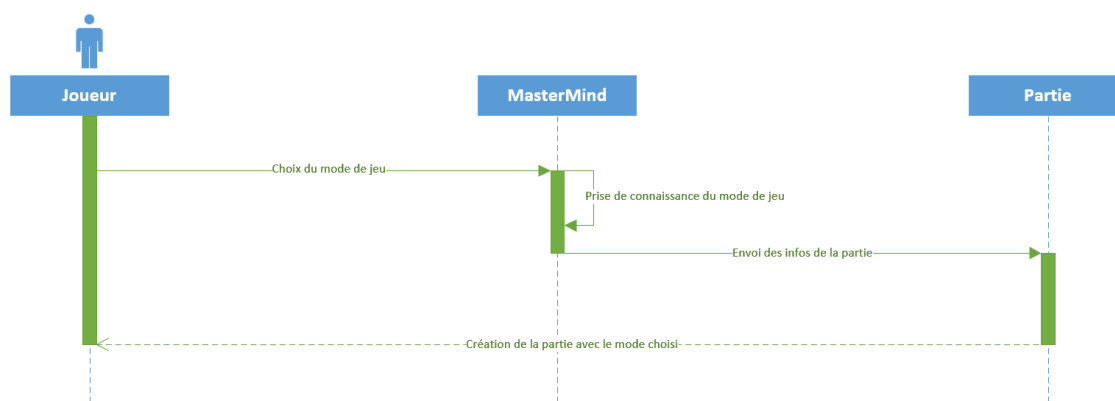


Figure 7: Diagramme de séquence : Choix mode de jeu

Diagramme de séquence dans le mode de jeu où l'utilisateur doit deviner la bonne combinaison

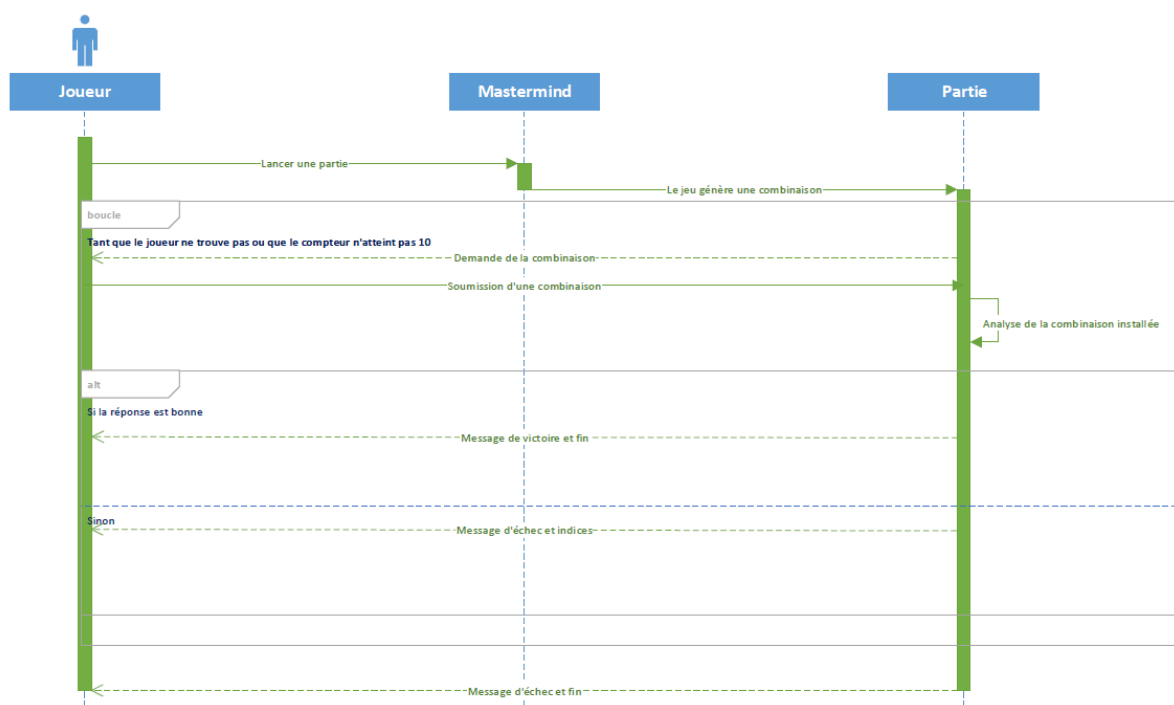


Figure 8: Diagramme de séquence : Deviner la bonne combinaison

Diagramme de séquence du mode de jeu où la machine doit trouver la combinaison

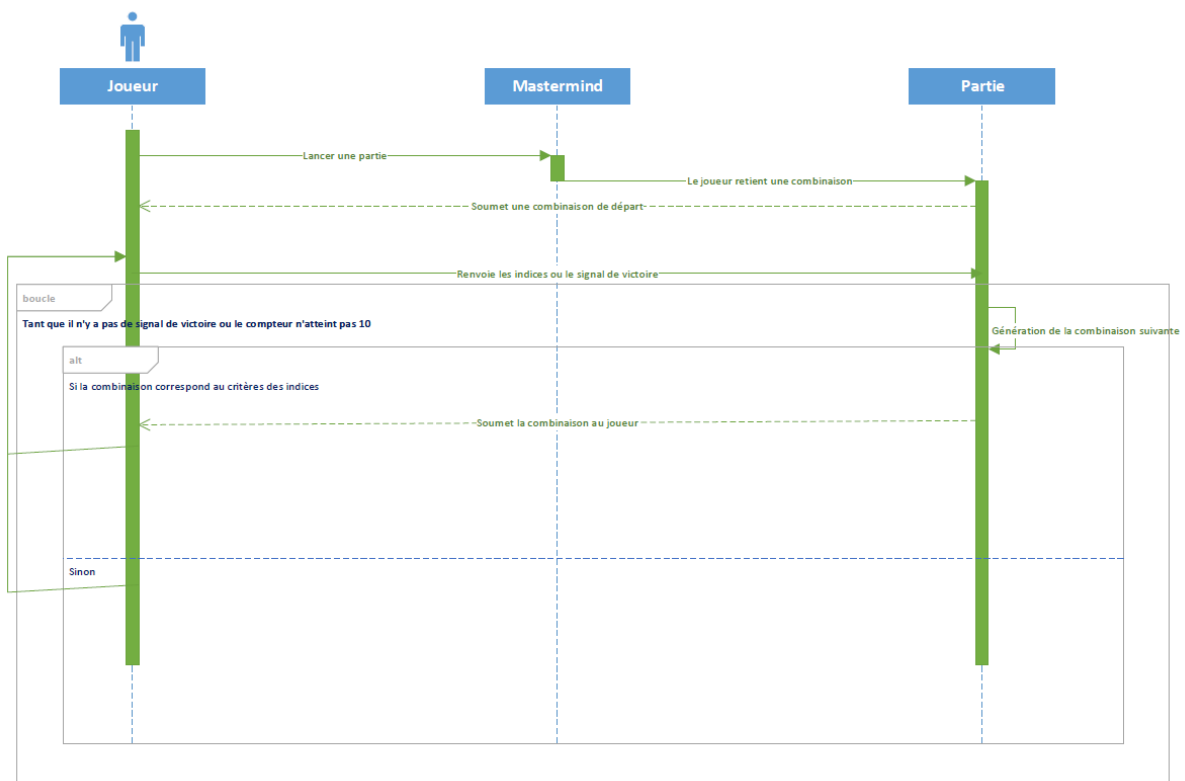


Figure 9: Diagramme de séquence : la machine doit trouver la bonne combinaison

L'utilisateur consulte les règles du jeu

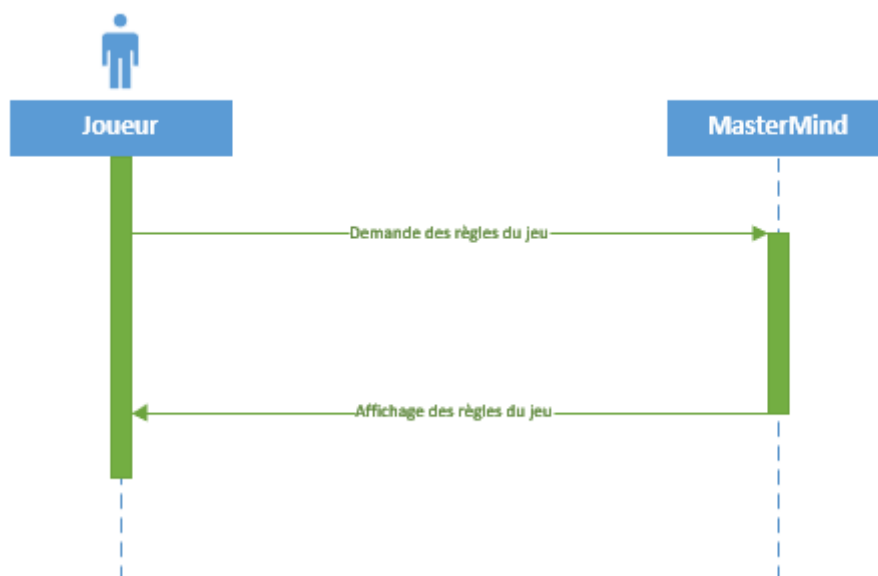


Figure 10: Diagramme de séquence : Consulter les règles de jeu

L'utilisateur consulte son profil

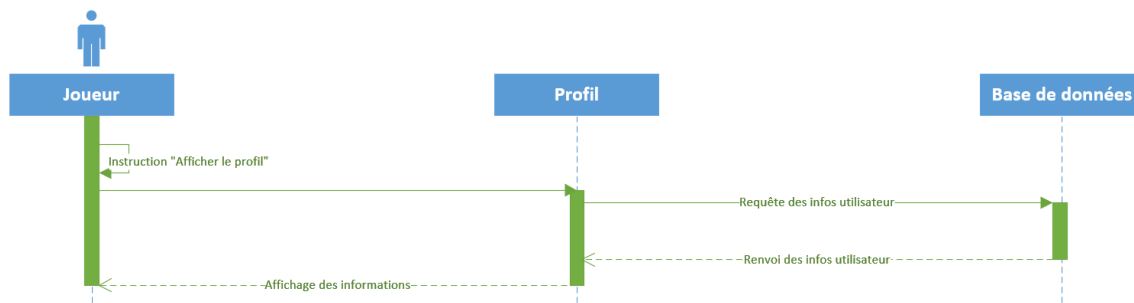


Figure 11: Diagramme de séquence : Afficher le profil

L'utilisateur modifie son profil

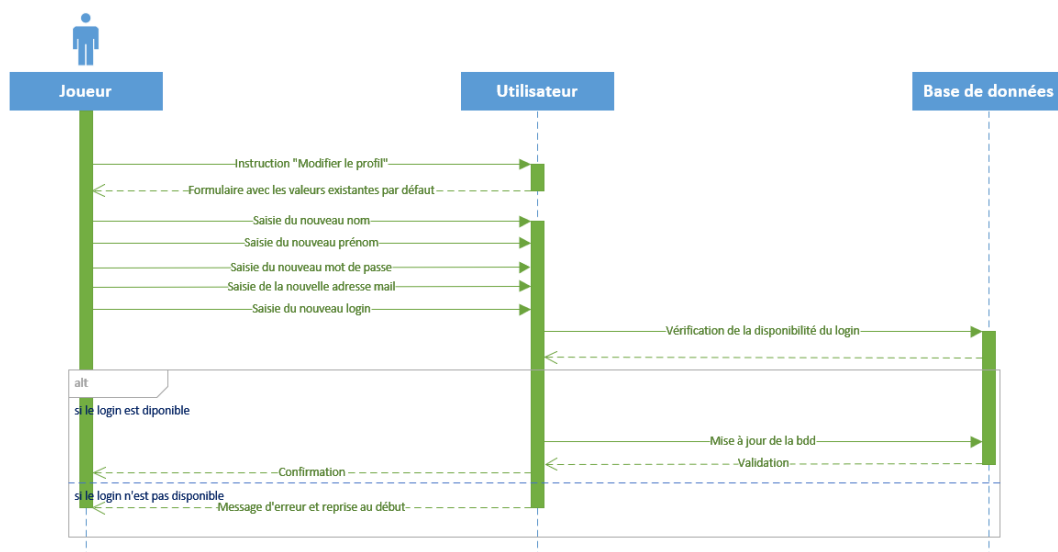


Figure 12: Diagramme de séquence : modifier profil

Diagramme de séquence lorsque l'utilisateur demande l'affichage de ses statistiques

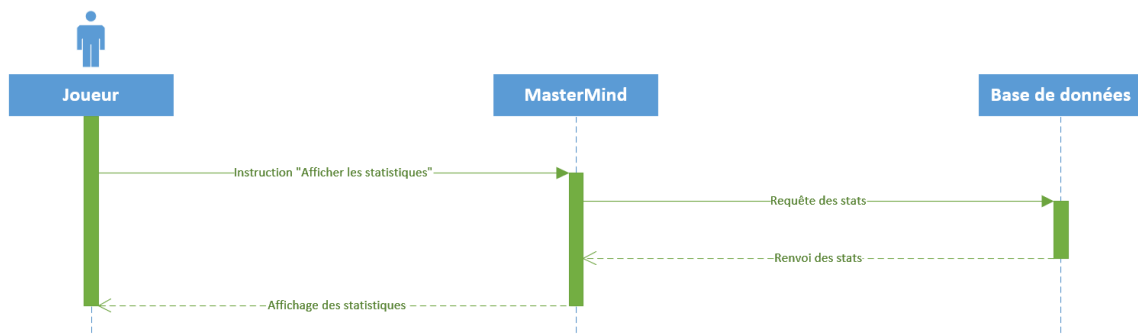


Figure 13: afficher les statistiques

Quitter l'application

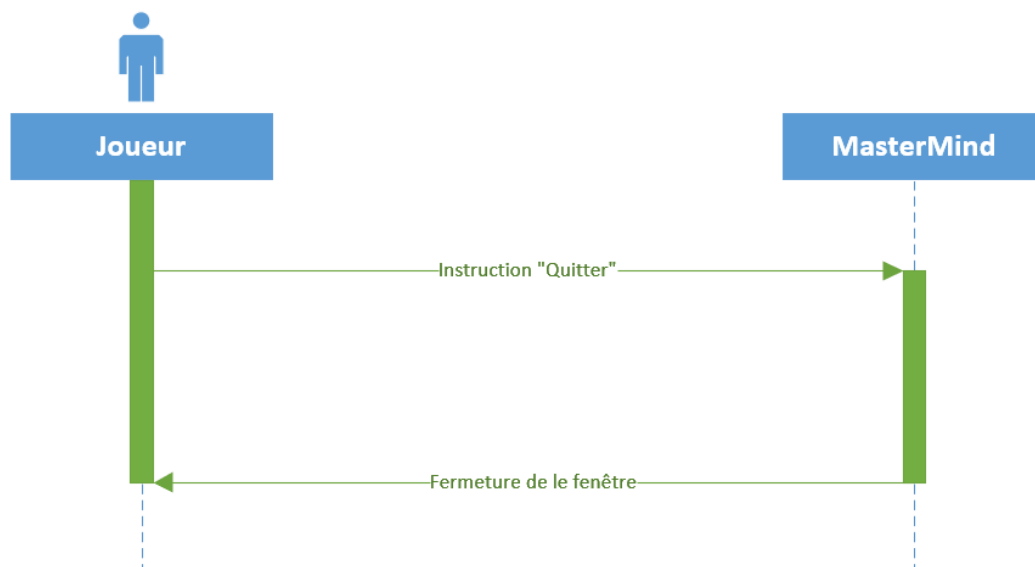


Figure 14: Diagramme de séquence : Quitter application

Algorithme mis en place

Algorithme du mode jeu : « Ordinateur choisi, Joueur cherche »

Fonction ayant pour but de gérer la partie "l'ordinateur choisi, le joueur cherche". La boucle principale, qui finit si on atteint 10 tours, appelle les fonctions qu'il faut en fonction de booléens en paramètres, mis à jour dans d'autres fonctions.

```

Fonction serieADeviner()
  Liste list = nouvelle liste liée.
  Tableau de int tab = Fonction generer5values.
  Finpartie = 0.
  Tant Que (finPartie n'est pas égale à 0)
    Fonction ToutParTour(finPartie)
    Si finPartie = 9
      fonctionVousAvezPerdu(valeurAléatoire)
      stop.
    Si gagné
      Bravo vous avez gagné
      Stop.
    FinPartie + 1
    Fonction displayJoueurJoue(liste)
  Fin serieADeviner()

```

Fonction ayant pour but de renvoyer un tableau d'entiers allant de 1 à 9.

```

Fonction statique generer5values()
  Tableau d'integer de 5 valeurs
  Random rand = new Random();
  Pour 1 à longueur de valeurs
    Valeurs de i = random values allant de 1 à 9
  Return valeurs ;
Fun generer5Values();

```

Fonction ayant pour but de gérer les inputs de l'utilisateur tour après tour.

```

Fonction toutParTour(integer tour)
  Chaîne de caractère choixUser() = vide ;
  Fait
    choixUser = input clavier.
  Tant que (fonction checkInput ne renvoie pas true)
    Foo = parsage en integer de choixUser
    Result = fonction CheckInput(foo, randomValues);
    Ajout à la liste (resultat)
    Si resultat == 5B/0N
      Vous avez gagné !
  Fin toutParTour()

```

Fonction ayant pour but de déterminer le nombre de pions blanc et/ou noir.

Les valeurs entrées par l'utilisateur et le choix du pc sont chacun stockés dans un tableau séparé. On travaille en simultané sur chacun des tableaux. Deux principales boucles vont être utilisé ici. La première a pour but de calculer les pions blancs.

Si à l'indice i le tableau choixJoueur et choixPc sont égaux, on ajoute +1 par bonne réponse. C'est que l'utilisateur a découvert le bon chiffre au bon endroit.

Maintenant, si l'utilisateur donne un bon chiffre mais pas au bon endroit, on le vérifie dans la fonction suivante. On compare, si `choixJoueur(i) == ordi(j)` et que `choixJoueur(j) == ordi(i)` on implémente les pions noirs.

```
Fonction checkInput(integer joueur, tableau d'integer ordi)  
  Chaîne de resultat result = « » ;  
  Chaîne de chiffre = parsage en string de joueur  
  Tableau de char digitis1 = parsage en character de number  
  Tableau d'integer choixJoueur = nouveau tableau d'integer de 5 cases  
  Integer blanc = 0 ;  
  Integer noir = 0 ;  
  
  Pour i = 0 ; i inférieur à longueur ordi  
    choixJoueur(i) = Obtenir numeric valeurs (digitis1[1])  
  
  Pour i = 0 ; i inférieur à longueur ordi  
    Si choixJoueur(i) == ordi(i)  
      Blanc + 1 ;  
      Continue  
    Pour j = 0 ; j inférieur à longueur ordi  
      Si choixJoueur(i) == ordi(j) et que choixJoueur(j) == ordi(i)  
        Noir + 1 ;  
  Result = blanc + « B » + noir « N »  
  Return result  
Fin checkInput();
```

Algorithme 'le joueur génère, le programme cherche'

Voici le corps principal de l'algorithme, les méthodes appelées sont détaillées plus bas.

```

Classe leProgrammeCherche {
  Attribut combinaison : Str
  Attribut permutations : Tableau de Str
  Attribut entrees : Tableau de Str
  Attribut essais : Tableau de Str

  Méthode principale() {
    permutations = initArray()
    essai : Str
    reponse : Str
    trouvé : booléen = true

    restants : Tableau de Str = Attribut permutations
    pour(i de 0 à 9) {
      afficher("tour " + i)
      essai = restants[0]
      Attribut essais.Ajouter(essai)
      afficher("Je propose : " + essai)
      reponse = lireInput()
      Tantque(verif(reponse)) {
        afficher("Vous devez envoyer votre réponse au format 'xBP/yMP'")
        reponse = lireInput()
      }
      Si(reponse == "5BP/0MP") {
        trouvé = vrai
        quitterBoucle()
      }
      memTampon : tableau de int = {NbBienPlacées,NbMalPlacées}
      Attribut entrees.Ajouter(memTampon)
      restants = getRestants(essai,NbBienPlacées,NbMalPlacées,restants)
      Si(restants.estVide()) {
        trouvé = faux
        Afficher("J'abandonne")
        quitterBoucle()
      }
    }
  }

  Si(trouvé) {
    Afficher("J'ai gagné! Vous aurez plus de chance la prochaine fois")
  } Sinon {
    Afficher("J'ai perdu! Quelle était la bonne réponse ?")
    reponse = lireInput()
    checkmistakes(reponse)
  }
}

```


Méthode initArray() :

```
Méthode initArray() {  
    chiffres : tableau de Str = {1,2,3,4,5,6,7,8,9}  
    resultat: tableau de Str  
  
    Pour(i1de 0 < 9) {  
        Pour(i2de 0 < 9) {  
            Si(chiffres[i1] != chiffres[i2]) {  
                Pour(i3de 0 < 9) {  
                    Si(chiffres[i3] != chiffres[i1] et chiffres[i3] != chiffres[i2]) {  
                        Pour(i4de 0 < 9) {  
                            Si(chiffres[i4] != chiffres[i1] et chiffres[i4] != chiffres[i2] et chiffres[i4] != chiffres[i3]) {  
                                Pour(i5de 0 < 9) {  
                                    Si(chiffres[i5] != chiffres[i1] et chiffres[i5] != chiffres[i2]  
                                    et chiffres[i5] != chiffres[i3] et chiffres[i5] != chiffres[i4]) {  
                                        resultat.ajouter(chiffres[i1]+ chiffres[i2]+ chiffres[i3]+ chiffres[i4]+ chiffres[i5])  
                                    }  
                                }  
                            }  
                        }  
                    }  
                }  
            }  
        }  
    }  
    renvoyer resultat  
}
```

Un simple contrôle peut permettre d'éviter les doublons dans cette boucle for (notez qu'un simple retrait d'une couche de conditions permettrait d'implémenter les doublons.

Méthode getRestants()

```

Méthode getRestants(guess : Str, BonsPlacements : int, MauvaisPlacements : int, restants : tableau de Str) {
    possibles : tableau de Str
    differences : tableau de int
    Pour(i de 0 à taille(restants)) {
        differences = getDifferences(guess, restants[i])
        Si(BonsPlacement == differences[0] et MauvaisPlacements == differences[1]) {
            possibles.Ajouter(restants[i])
        }
    }
    renvoyer possibles
}

```

Méthode checkMistakes()

```

Méthode getDifferences(essai : Str, reponse : Str) {
    renvoi : tableau de int = {0,0}
    tableEssai : tableau de char = essai
    tableReponse : tableau de char = reponse
    pour(i de 0 à 4) {
        si(tableEssai[i] == tableReponse[i]) {
            renvoi[0]++
        }
        pour(j de 0 à 4) {
            si(tableEssai[i] == tableReponse[j] et i != j) {
                renvoi[1]++
            }
        }
    }
}

```

Ces méthodes éliminent toutes les combinaisons qui ne sont plus possibles, déduites en utilisant les informations 'Bien placés' et 'Mal placés' renvoyées par le joueur.

La combinaison tentée par la machine est comparée à chaque combinaison du tableau des combinaisons possibles. Si le nombre de bien placés et de mal placés correspond à celui donné par le joueur, la combinaison est conservée. Sinon, elle est éliminée.

Règles du jeu

Le jeu comporte deux modes de jeu différents :

- Le premier consiste à faire deviner au joueur un nombre composé de 5 chiffres différents (dans une première version sans doublon, les doublons sont implémentés par la suite). Après chaque essai le joueur se voit indiquer le nombre de chiffres bien placés et le nombre de chiffres mal placés. La somme de ces deux nombres ne peut pas logiquement excéder 5. Le joueur dispose de 10 essais pour essayer de trouver la bonne combinaison. Des erreurs de saisies peuvent lui être indiquées s'il en commet (Pas assez de chiffre, doublon dans la première version du jeu, caractère n'étant pas un chiffre, ...). Si le joueur trouve la bonne combinaison, l'application affiche "Bravo" durant le tour. Si au bout de 10 essais, le joueur n'a toujours pas trouvé la bonne combinaison, l'application indique que le joueur a perdu.

- Le deuxième mode de jeu consiste à faire deviner une combinaison de 5 chiffres à l'application. L'application a elle aussi 10 essais pour tenter de trouver la bonne combinaison. A chaque tour de l'application, le joueur doit indiquer si les chiffres saisis par cette dernière sont mal placés ou au contraire dans la bonne position. Une vérification de la saisie des positions est faite à chaque tour, empêchant l'utilisateur de saisir des données incorrectes, mais ne l'empêchant pas de mentir sur les positions. Si l'application trouve la bonne combinaison, elle gagne la partie. Si elle ne trouve pas au bout des 10 essais, l'application demande la bonne combinaison au joueur. Une vérification est alors effectuée pour savoir si le joueur n'a pas menti.

Ces deux modes de jeu sont accessibles depuis un espace de login qui permet à l'utilisateur de consulter son profil composé de son nom, prénom, email, date de naissance ainsi que de son mot de passe.

Le menu statistiques permet à l'utilisateur de consulter les statistiques propres à ces parties.

Un autre menu permet à l'utilisateur de changer les informations de son profil.

Mode d'emploi et déploiement du jeu

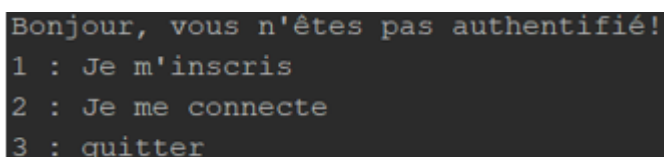
Le jeu sera un fichier d'extension « .jar », directement exécutable par le joueur. Cependant une base de données MySQL doit être présente sur l'ordinateur de l'utilisateur afin de pouvoir jouer dans de bonnes conditions. La dernière version de Java devra être présente, elle aussi.

Déploiement :

- Dézipper l'archive .rar.
- Lancer votre serveur MySQL.
- Exécuter le fichier "MasterMind.bat"
- Saisir le mot de passe de votre base de données.

Mode d'emploi du jeu :

- Pour procéder à l'inscription d'un utilisateur choisir 1 sur l'écran de login.
- Pour s'authentifier sélectionner 2



```
Bonjour, vous n'êtes pas authentifié!  
1 : Je m'inscris  
2 : Je me connecte  
3 : quitter
```

Figure 15 : Menu de bienvenue

- Lors de l'inscription les champs suivants sont à remplir. En cas d'erreur de saisie, l'application indique la marche à suivre

```
Page d'inscription
Veuillez saisir votre login :
romain
Veuillez saisir votre nom :
romain
Veuillez saisir votre prenom :
romain
Veuillez saisir votre email :
romain@gmail.com
Veuillez saisir votre date de naissance au format JJMMAAAA :
01061991
Veuillez saisir votre mot de passe :
romain
Confirmer votre mot de passe :
romain
Inscription en cours....
```

Figure 16: Menu Inscription

- Une redirection automatique est faite vers la page de login
- Une fois l'authentification passée, sélectionner 1 pour lancer une partie dans le premier mode de jeu
- Saisir 2 pour lancer une partie dans le second mode de jeu
- Sélectionner 3 pour afficher le profil
- Saisir 4 pour modifier les informations du profil
- Saisir 5 pour afficher les statistiques du joueur authentifié
- Saisir 6 pour quitter

```
login/mot de passe correct
Bonjour, romain!
1 : Je choisis la combinaison, vous jouez
2 : Vous choisissez la combinaison, je joue
3 : Afficher votre profil
4 : Modifier vos informations
5 : Afficher les statistiques
6 : quitter
```

Figure 17: Menu principal

Dans le cadre du premier mode de jeu :

```
Jeu, la machine choisi la combinaison
=====Tour N°1=====
--> Votre choix :
16253
XXXXX
1 : 16253 => 2B/0N.
```

Figure 18: Action le joueur joue

- Saisir le potentiel nombre deviné, l'ordinateur répond en indiquant les chiffres bien placés et ce qui sont présent mais mal placé.
- Le joueur dispose de 10 tentatives pour trouver la bonne combinaison.

Dans le cadre du deuxième mode de jeu :

```
2
À vous de jouer!
Choisissez une combinaison, cette combinaison doit être composée de cinq chiffres différents,
Envoyez votre réponse au format 'xBP/yMP'
Tour 1
Je propose : 12345
```

Figure 19: L'ordinateur joue

- Saisir le nombre de bonnes réponses, soit les chiffres bien placés par l'ordinateur sous forme xBP, où x représente le nombre de chiffres bien placés.
- Saisir le nombre de réponses mal placées, soit les chiffres présents mais mal placés par l'ordinateur sous forme yMP, où y représente le nombre de chiffres mal placés.
- L'ordinateur dispose de 10 tentatives pour trouver la bonne combinaison.
- Pour afficher le profil utilisateur, sélectionner l'option 3 :

```
Voici vos informations actuelles
page de profil
Login : Ferreol
Nom : JEANNOT
Prénom : Ferr0ol
Email : ferreol@jeannot.ht.st
Date de naissance : 1995-10-10
```

Figure 20: Menu Afficher profil

- Pour modifier le profil utilisateur, sélectionner l'option 4 :

```
Voici vos informations actuelles
page de profil
Login : Ferreol
Nom : JEANNOT
Prénom : Ferr0ol
Email : ferreol@jeannot.ht.st
Date de naissance : 1995-10-10
Entrez vos nouvelles informations
Nouveau mot de passe:
Rt6tTarlst
Nouvelle adresse mail [ferreol@jeannot.ht.st]:
```

Figure 21: Modification profil

Pour afficher la page de statistique, sélectionner l'option 5 :

```

Page de statistiques
Nombre de parties jouées : 4
Parties Gagnées : 2
Parties perdues : 2
Résultat meilleure partie : 9 Date : 2018-03-13
Place dans le classement : 1 / 3
  
```

Figure 22: Page statistiques

Planification

Mise en place du projet
Diagrammes de classes métier
Diagrammes de séquences
Diagramme de base de donnée
Diagrammes de cas d'utilisation
Création du squelette MVC
Algorithmes "l'application génère, le joueur joue"
Algorithmes "le joueur génère, l'application cherche"
Implémenter la base de donnée
Inscription -> BDD
BDD -> Connexion
Version 1: l'application génère, le joueur joue
Version 2: le joueur génère, l'application cherche
BDD -> stockage des statistiques
Gestion des 10 coups max avant réponse
Gestion des erreurs de saisies
Gestion des doublons
Finalisation rapport
Rendu du projet

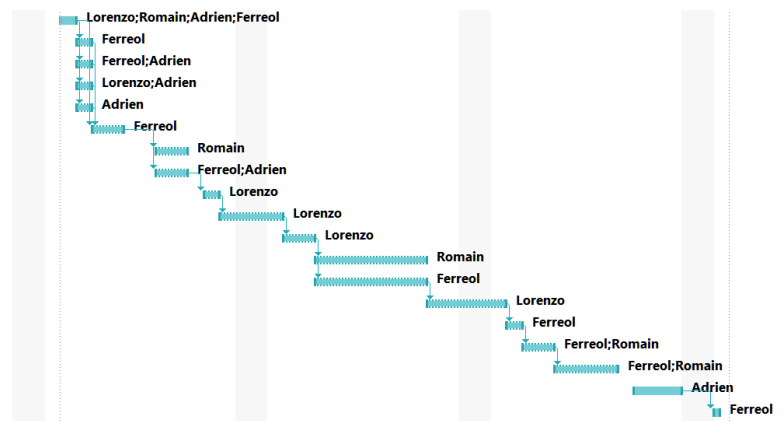


Figure 23: Affichage du gantt

Le diagramme de Gantt ci-dessus régit les différentes tâches des membres de l'équipe et leurs organisations dans le temps.

Charte de développement

Une charte de développement a été mise en place par l'équipe afin d'uniformiser l'ensemble du code. La charte comprend les nommages suivants :

- Toutes les vues comportent le mot « View »
- Tous les noms de fichiers doivent être explicites
- Les classes abstraites possèdent la même nomenclature : « Abstract.... »
- Si le nom d'une classe est composé de deux mots ou plus chaque mot comporte une majuscule
- Si le nom de la méthode ou de la variable est composée de deux mots ou plus, le premier mot ne comporte pas de majuscule, le second en comporte etc. ...
- Chaque accolade démarre sur la même ligne de l'instruction

Cette charte doit être respectée lors de la réalisation du projet.

Bilan

Au terme de ce projet nous avons pu réaliser une grande partie des objectifs qui nous avaient été fixés.

Une bonne entente et surtout un engagement des membres de l'équipe a permis de rendre le projet dans les temps.

Le principal problème rencontré correspond à la mise en commun de nos codes. Les divers push et pull ont posé de nombreux problèmes de merge. Nous avons dû reprendre le code à la main pour régler les problèmes.

Nous avons malheureusement sous-estimé la partie documentation de notre code, nous nous sommes trop concentrés sur le Gantt, et le codage du jeu. Nous nous sommes retrouvés rapidement surchargés de documentation à rendre. Nous nous en souviendrons pour la prochaine fois.

Nous estimons donc que ce projet a été très enrichissant pour notre expérience personnelle et collective.

Annexes

Fiche d'itération 1



GROUPE CHAUDESSOLLE JEANNOT LACUBE MONTIGNEAUX
NOM DU PROJET : [JAVA] PROJET MASTERMIND
Fiche d'itération 1

Auteurs :

Auteurs	Approbateurs	Validation
MONTIGNEAUX	JEANNOT LACUBE	Samir AZZAG
Rédigé le : 19/12/17	Approuvé le : 19/11/17	

Diffusion :

Diffusion	Externe
À :	samir.azzag@ynov.com
Copies à :	JEANNOT LACUBE CHAUDESSOLLE

Document de référence :

Libellé	Document
Fiche Itération 1.0	FI1.pdf

Historique :

N° Version	Auteurs	Approbateurs	Date	Historique des évolutions
1.0	MONTIGNEAUX	JEANNOT LACUBE	19/12/17	Mise en place du projet

Figure 24: Fiche itération 1

Liste des tâches	Auteurs	Approbateurs	Date début	Rendu Prévisionnel	Rendu réel	Notes
Diagrammes de classes métier	JEANNOT	MONTIGNEAUX	20/12/17	20/12/17	20/12/17	
Diagramme de séquences	JEANNOT MONTIGNEAUX	MONTIGNEAUX JEANNOT	20/12/17	20/12/17	20/12/17	
Diagramme de base de données	MONTIGNEAUX CHAUDESSOLLE	JEANNOT	20/12/17	20/12/17	20/12/17	
Diagramme de cas d'utilisations	MONTIGNEAUX	JEANNOT	20/12/17	20/12/17	20/12/17	
Création du squelette MVC	JEANNOT	MONTIGNEAUX	21/12/17	21/12/17	22/12/17	
Algorithme "l'application génère, le joueur joue"	LACUBE	JEANNOT	25/12/17	26/12/17	26/12/17	
Algorithme "le joueur génère, l'application cherche"	JEANNOT MONTIGNEAUX	JEANNOT	25/12/17	26/12/17	26/12/17	

Figure 25: Répartition des tâches

Fiche d'itération 2



GROUPE CHAUDESSOLLE JEANNOT LACUBE MONTIGNEAUX
NOM DU PROJET : [JAVA] PROJET MASTERMIND
Fiche d'itération 2

Auteurs :

Auteurs	Approbateurs	Validation
MONTIGNEAUX	JEANNOT LACUBE	Samir AZZAG
Rédigé le : 28/12/17	Approuvé le : 28/12/17	

Diffusion :

Diffusion	Externe
À :	samir.azzag@ynov.com
Copies à :	JEANNOT LACUBE CHAUDESSOLLE

Document de référence :

Libellé	Document
Fiche Itération 1.0	FI1.pdf

Historique :

N° Version	Auteurs	Approbateurs	Date	Historique des évolutions
1.0	MONTIGNEAUX	JEANNOT LACUBE	19/12/17	Mise en place du projet

Figure 26: Fiche itération 2

Liste des tâches	Auteurs	Approbateurs	Date début	Rendu Prévisionnel	Rendu réel	Notes
Implémenter la base de données	CHAUDESSOLLE	MONTIGNEAUX	28/12/17	28/12/17	28/12/17	
Inscription de l'utilisateur	CHAUDESSOLLE MONTIGNEAUX	MONTIGNEAUX JEANNOT	29/12/17	01/01/18	01/01/18	
Connexion de l'utilisateur	CHAUDESSOLLE	JEANNOT MONTIGNEAUX	02/01/18	03/01/18	03/01/18	
Version 1 du mode de jeu « Application génère, le joueur joue »	LACUBE	JEANNOT	04/01/18	10/01/18	10/01/18	
Version 2 du mode de jeu « le joueur génère, l'application joue »	JEANNOT	MONTIGNEAUX LACUBE	04/01/18	10/01/18	10/01/18	

Figure 27: Répartition des tâches

Fiche d'itération 3



GROUPE CHAUDESSOLLE JEANNOT LACUBE MONTIGNEAUX
NOM DU PROJET : [JAVA] PROJET MASTERMIND
Fiche d'itération 3

Auteurs :

Auteurs	Approbateurs	Validation
MONTIGNEAUX	JEANNOT LACUBE	Samir AZZAG
Rédigé le : 11/01/18	Approuvé le : 24/01/18	

Diffusion :

Diffusion	Externe
À :	samir.azzag@ynov.com
Copies à :	JEANNOT LACUBE CHAUDESSOLLE

Document de référence :

Libellé	Document
Fiche Itération 1.0	FI1.pdf

Historique :

N° Version	Auteurs	Approbateurs	Date	Historique des évolutions
1.0	MONTIGNEAUX	JEANNOT LACUBE	19/12/17	Mise en place du projet
2.0	MONTIGNEAUX	JEANNOT LACUBE	28/12/17	Avancement du projet

Figure 28: Fiche itération 3

Liste des tâches	Auteurs	Approbateurs	Date début	Rendu Prévisionnel	Rendu réel	Notes
Base de données : Stockage des statistiques	CHAUDESSOLLE	MONTIGNEAUX JEANNOT	11/01/18	15/01/18	15/01/18	
Gestion des 10 coups max avant réponse	JEANNOT	MONTIGNEAUX JEANNOT	16/01/18	16/01/18	16/01/18	
Gestion des erreurs de saisies	JEANNOT LACUBE	JEANNOT MONTIGNEAUX	17/01/18	18/01/18	18/01/18	
Gestion des doublons	JEANNOT LACUBE	JEANNOT MONTIGNEAUX	19/01/18	22/01/18	22/01/18	
Finalisation rapport	MONTIGNEAUX	JEANNOT LACUBE	24/01/18	29/01/18	29/01/18	

Figure 29: Répartition des tâches

Compte rendu de réunion 1



NOM DU PROJET : [JAVA] Projet MasterMind

Compte-rendu de reunion 20/12/2017

Auteurs :

Auteurs	Approbateurs	Validation
LACUBE	MONTIGNEAUX, JEANNOT, CHAUDESSOLLE	
Rédigé le : 20/12/17	Approuvé le : 21/12/17	

Diffusion :

Diffusion	Externe
À : LACUBE	
Copies à : MONTIGNEAUX, CHAUDESSOLLE, JEANNOT	

Document de référence :

Libellé	Document

Historique :

N° Version	Auteurs	Approbateurs	Date	Historique des évolutions
1.0	LACUBE	JEANNOT, CHAUDESSOLLE, MONTIGNEAUX	20/12/17	Init

Objectifs de la réunion :

Mettre en place le projet et les outils utilisés par l'équipe

Lancer le projet

Date de la réunion : 20/12/2017 à l'école.**Liste des présents :** LACUBE, MONTIGNEAUX, JEANNOT, CHAUDESSOLLE

Sujets abordés :

- Définition des outils à utiliser pour la gestion de projet
- Définition du moyen de communication utilisé par l'équipe (Slack)
- Lecture du sujet
- Distribution des rôles

Date et lieu de la prochaine réunion : 21/12/2017 à l'école.

Figure 30: Compte rendu de réunion du 20/12/2017

Compte rendu de réunion 2



NOM DU PROJET : [JAVA] Projet MasterMind

Compte-rendu de reunion 21/12/2017

Auteurs :

Auteurs	Approbateurs	Validation
LACUBE	MONTIGNEAUX, JEANNOT, CHAUDESSOLLE	
Rédigé le : 21/12/17	Approuvé le : 21/12/17	

Diffusion :

Diffusion	Externe
À : LACUBE	
Copies à : MONTIGNEAUX, CHAUDESSOLLE, JEANNOT	

Document de référence :

Libellé	Document

Historique :

N° Version	Auteurs	Approbateurs	Date	Historique des évolutions
1.0	LACUBE	JEANNOT, CHAUDESSOLLE, MONTIGNEAUX	20/12/17	Init
2.0	LACUBE	JEANNOT, CHAUDESSOLLE, MONTIGNEAUX	21/12/17	Gantt / Repartition des taches

Objectifs de la réunion :

Répartir équitablement les taches, définir l'ensemble du gantt, les classes, et les différents retour de chaque fonctions.

Date de la réunion : 21/12/2017 à l'école.

Liste des présents : LACUBE, MONTIGNEAUX, JEANNOT, CHAUDESSOLLE

Compte tenu des vacances, nous devons mettre en place tout le guideline que chacun devra suivre afin de rendre un projet fini et équitable en taches à réaliser.

Date et lieu de la prochaine réunion : 21/12/2017 à l'école.

Figure 31: Compte rendu de réunion du 21/12/2017

Compte rendu de réunion 3



NOM DU PROJET : [JAVA] Projet MasterMind

Compte-rendu de reunion 20/02/2018

Auteurs :

Auteurs	Approbateurs	Validation
LACUBE	MONTIGNEAUX, JEANNOT, CHAUDESSOLLE	
Rédigé le : 21/12/17	Approuvé le : 21/12/17	

Diffusion :

Diffusion	Externe
À : LACUBE	
Copies à : MONTIGNEAUX, CHAUDESSOLLE, JEANNOT	

Document de référence :

Libellé	Document

Historique :

N° Version	Auteurs	Approbateurs	Date	Historique des évolutions
1.0	LACUBE	JEANNOT, CHAUDESSOLLE, MONTIGNEAUX	20/12/17	Init
2.0	LACUBE	JEANNOT, CHAUDESSOLLE, MONTIGNEAUX	21/12/17	Gantt / Repartition des taches

Objectifs de la réunion :

Compte rendu final

Date de la réunion : 21/12/2017 à l'école.**Liste des présents :** LACUBE, MONTIGNEAUX, JEANNOT, CHAUDESSOLLE

6 semaines sont passés depuis la dernière itération. Nous avons mis en place tout un guideline permettant de finir chacun de son côté sa partie. Aujourd'hui nous devons mettre notre code en commun. Et tout s'est bien passé ! non je plaisante... on a eu quelques problèmes de merge avec GIT, mais sinon tout est, a l'issue de cette longue réunion, ok et fonctionnel.

En parallèle, nous générons la javadoc, et corrigeons les derniers points manquant au projet :

- Dictionnaire des variables
- Fiches d'itérations
- Modification du diagramme de classe

Date et lieu de la prochaine réunion : 21/12/2017 à l'école.

Figure 32: Compte-rendu du 20/02/2018

Tests nominaux

Ordre	Scénario : C1 - Inscription d'un utilisateur	Cas de test : valeurs	Cas de test : résultats attendus	Test : résultats obtenus	Test : diagnostic
1	C1-SN Scénario nominal	Pseudo : romain Mot de passe : romain Confirmation : romain	Message : "Inscription validée, merci". L'utilisateur a été ajouté à la base de données (TABLE users).	Figure 1	Conforme

Ordre	Scénario : C2 - Choix Menu	Cas de test : valeurs	Cas de test : résultats attendus	Test : résultats obtenus	Test : diagnostic
1	C2-SN Scénario nominal	Entrée : 2	Envoie sur la vue demandé	Figure 2	Conforme
2	C2-SA1 – Scénario <u>alternatif</u>	Entrée autre qu'entre 1 et 6	Retour menu	Figure 3	Conforme
3	C2-SA1 – Scénario <u>alternatif</u>	Entrée : alphabétique	Retour menu	Figure 4	Conforme

Ordre	Scénario : C3 - Afficher statistiques	Cas de test : valeurs	Cas de test : résultats attendus	Test : résultats obtenus	Test : diagnostic
1	C3-SN Scénario nominal				

Ordre	Scénario : C4 - Modifier profil	Cas de test : valeurs	Cas de test : résultats attendus	Test : résultats obtenus	Test : diagnostic
1	C4-SN Scénario nominal				

Ordre	Scénario : C5 - Authentification	Cas de test : valeurs	Cas de test : résultats attendus	Test : résultats obtenus	Test : diagnostic
1	C5-SN Scénario nominal	Pseudo : pseudo existant en bdd Mdp : mdp en rapport au pseudo	Affichage menu principal	Figure 5	Conforme

Ordre	Scénario : C6 - Entrer valeur	Cas de test : valeurs	Cas de test : résultats attendus	Test : résultats obtenus	Test : diagnostic
1	C6-SN Scénario nominal	Entrée : 16253	Comparaison valeur entrée/valeur à trouver	Figure 6	Conforme
2	C6-SA1 – Scénario alternatif	Entrée A9271	Rappel syntaxe, 1 ^{ère} itération continue	Figure 7	Conforme
3	C6-SA2 – Scénario alternatif	Entrée 679271	Rappel syntaxe, 1 ^{ère} itération continue	Figure 8	Conforme

```
Bonjour, vous n'êtes pas authentifié!
1 : Je m'inscris
2 : Je me connecte
3 : quitter
1
Page d'inscription
Veuillez saisir votre login :
romain
Veuillez saisir votre nom :
romain
Veuillez saisir votre prenom :
romain
Veuillez saisir votre email :
romain@gmail.com
Veuillez saisir votre date de naissance au format JJMMAAAA :
01061991
Veuillez saisir votre mot de passe :
romain
Confirmer votre mot de passe :
romain
Inscription en cours....
Bonjour, vous n'êtes pas authentifié!
1 : Je m'inscris
2 : Je me connecte
3 : quitter
```

Figure 1

```
login/mot de passe correct
Bonjour, romain!
1 : Je choisis la combinaison, vous jouez
2 : Vous choisissez la combinaison, je joue
3 : Afficher votre profil
4 : Modifier vos informations
5 : Afficher les statistiques
6 : quitter
2
À vous de jouer!
Choisissez une combinaison, cette combinaison doit être composée de cinq chiffres différents,
Envoyez votre réponse au format 'xBP/yMP'
Tour 1
Je propose : 12345
```

Figure 2

```
Bonjour, romain!
1 : Je choisis la combinaison, vous jouez
2 : Vous choisissez la combinaison, je joue
3 : Afficher votre profil
4 : Modifier vos informations
5 : Afficher les statistiques
6 : quitter
7
Bonjour, romain!
1 : Je choisis la combinaison, vous jouez
2 : Vous choisissez la combinaison, je joue
3 : Afficher votre profil
4 : Modifier vos informations
5 : Afficher les statistiques
6 : quitter
```

Figure 3

```
login/mot de passe correct
Bonjour, romain!
1 : Je choisis la combinaison, vous jouez
2 : Vous choisissez la combinaison, je joue
3 : Afficher votre profil
4 : Modifier vos informations
5 : Afficher les statistiques
6 : quitter
dzdez
Bonjour, romain!
1 : Je choisis la combinaison, vous jouez
2 : Vous choisissez la combinaison, je joue
3 : Afficher votre profil
4 : Modifier vos informations
5 : Afficher les statistiques
6 : quitter
```

Figure 4

```
Page d'inscription
Veuillez saisir votre login :
romain
romain n'est pas disponible, en prendre un autre
Veuillez saisir votre login :
```

Figure 5

```
679271
Veuillez saisir 5 chiffres
```

```
login/mot de passe correct
Bonjour, romain!
1 : Je choisis la combinaison, vous jouez
2 : Vous choisissez la combinaison, je joue
3 : Afficher votre profil
4 : Modifier vos informations
5 : Afficher les statistiques
6 : quitter
1
Jeu, la machine choisi la combinaison
=====Tour N°1=====
--> Votre choix :
16253
XXXXX
1 : 16253 => 2B/0N.
=====Tour N°2=====
--> Votre choix :
```

Figure 6

```
XXXXX
1 : 16253 => 2B/0N.
=====Tour N°2=====
--> Votre choix :
A9271
Veuillez saisir uniquement des chiffres
```

Figure 7

```
679271
Veuillez saisir 5 chiffres
```

Figure 8