
Manuel technique

1. Introduction

Ce projet a été réalisé dans le cadre de mon TPI (Travail pratique individuel) de fin de scolarité au CFPT-Informatique en 2021. Il a pour but de confirmer les compétences que j'y aie acquises lors de ces trois années.

Ce document est le manuel technique de l'application. Il y contient une analyse fonctionnelle et organique ainsi qu'un plan, un rapport de tests et le planning effectif.

2. Méthodologie

J'ai choisi pour ce travail d'utiliser la méthodologie de type « WaterFall ». Celle-ci consiste à diviser le
projet en plusieurs étapes afin de pouvoir facilité la planification et le développement. Cette méthode consiste en 6 étapes qu'il faut effectuer à la suite les unes des autres, la dernière n'étant pas utilisée dans le cadre de ce projet :

- Exigences (analyse du cahier des charges)
- Conception
- Planning échéancier
- Mise en œuvre, développement
- Vérification
- (Maintenance)

3. Résumé du cahier des charges

3.1 But de l'application

Permettre à un utilisateur de jouer au Blackjack avec les règles de base (21 et As) et sans mise. Pour aider le joueur à savoir s'il doit piocher ou pas, un conseiller sera là pour l'aiguiller. Le conseiller lui affichera aussi les statistiques d'avoir un tirer gagnant.

3.2 But du jeu

"*Après avoir reçu deux cartes, le joueur tire des cartes pour s'approcher de la valeur 21 sans la dépasser. Le but du joueur est de battre le croupier en obtenant un total de points supérieur à celui-ci ou en voyant ce dernier dépasser 21. Chaque joueur joue contre le croupier, qui représente la banque, ou le casino, et non contre les autres joueurs.*"

Règles par guide blackjack

3.3 Fonctionnalités à réaliser

En plus de pouvoir joué au Blackjack contre un croupier, l'application nous permet d'être conseillé sur la pioche des cartes ainsi que de voir les probabilités que l'on a de pioché une certaine carte ou d'avoir une certaine valeur totale dans la main après la pioche. Elle nous permet aussi de choisir le nombre de paquet avec lequel on vas jouer.

3.4 Matériel et logiciels nécessaires

- Ordinateur
- Une IDE prenant en charge le C# en l'occurrence VisualStudio 2019
- GitHub Desktop

3.5 Livrables

- Planning réel
- Rapport de projet
- Résumé du rapport du TPI
- Journal de travail
- Un lien github pour accéder aux sources du TPI

4. Analyse fonctionnelle

Cette section de la documentation traite de la partie visible de l'application, vue par l'utilisateur final.
Tout d'abord, sont énumérées les fonctionnalités disponibles dans l'application. Ensuite viennent les
présentations des fenêtres de l'interface utilisateur.

4.1 Liste des fonctionnalités

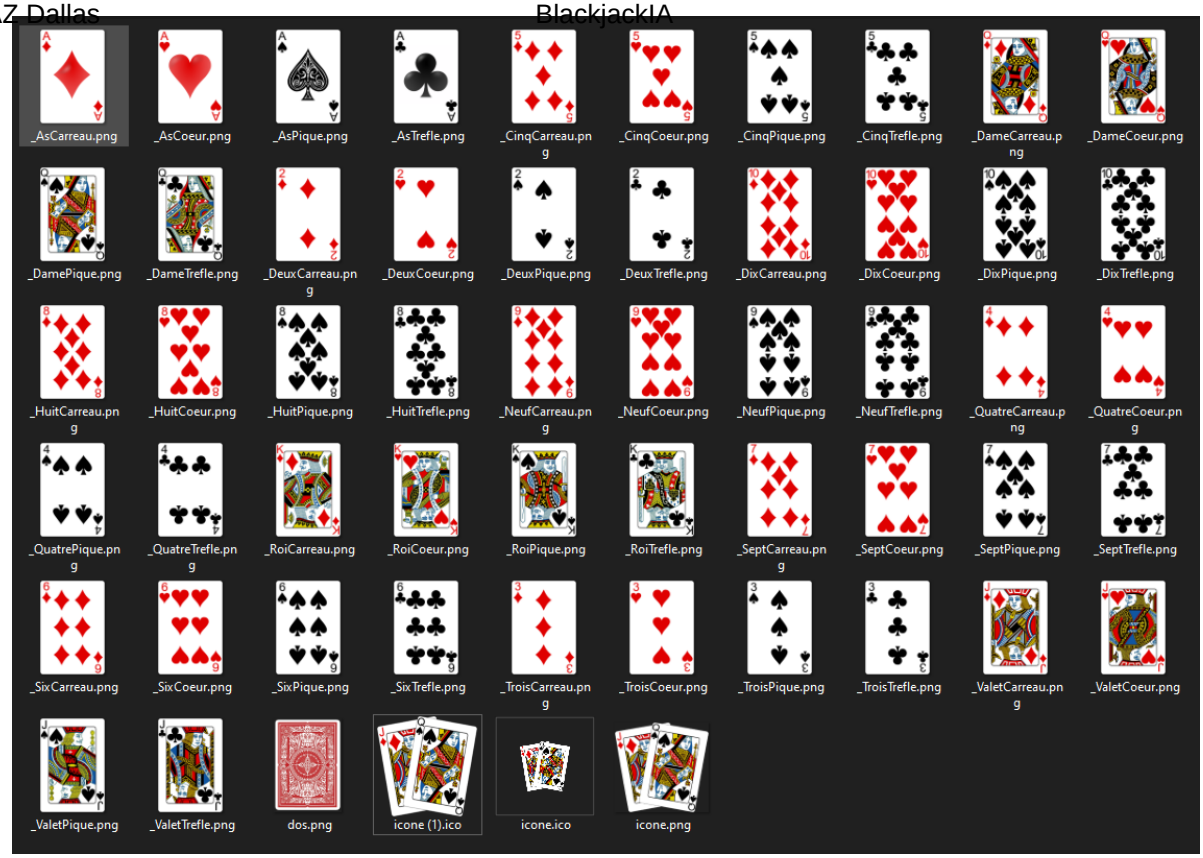
Voici la liste des actions possibles par l'utilisateur :

- Choisir son nom
- Choisir le nombre de paquet avec lesquels on vas jouer
- Piocher une carte
- Rester

4.2 Présentation de l'interface

L'interface utilisateur comprend au total 2 formulaires WindowsForms. La première sert à choisir le nom du joueur ainsi que le nombre de paquets que l'on va utiliser pour la partie. La seconde est celle ou la partie se déroulera avec l'affichage de la main du croupier et celle du joueur, des boutons pour piocher ou rester ainsi que les statistiques de pioche.

4.2.1 Les images



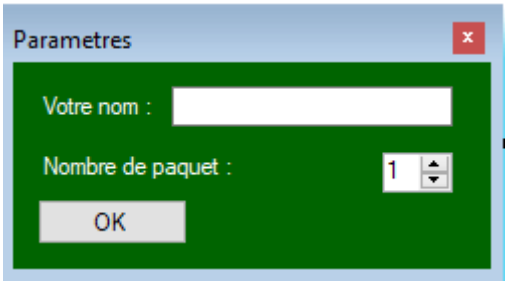
Les cartes ainsi que l'icône de l'application ont été récupéré sur commons.wikimedia.
Pour le dos des cartes l'image à été trouvée sur Google Images et peut-être retrouver à cette adresse.

À la base toutes les images de cartes étaient des SVG (Scalable Vector Graphics). Après quelques recherches car je ne savais pas comment utiliser des SVG en WindowsForm, l'implémentation me paraissait complexe et être une perte de temps. Je les ai donc toutes converties en png.

4.2.2 La vue paramètre

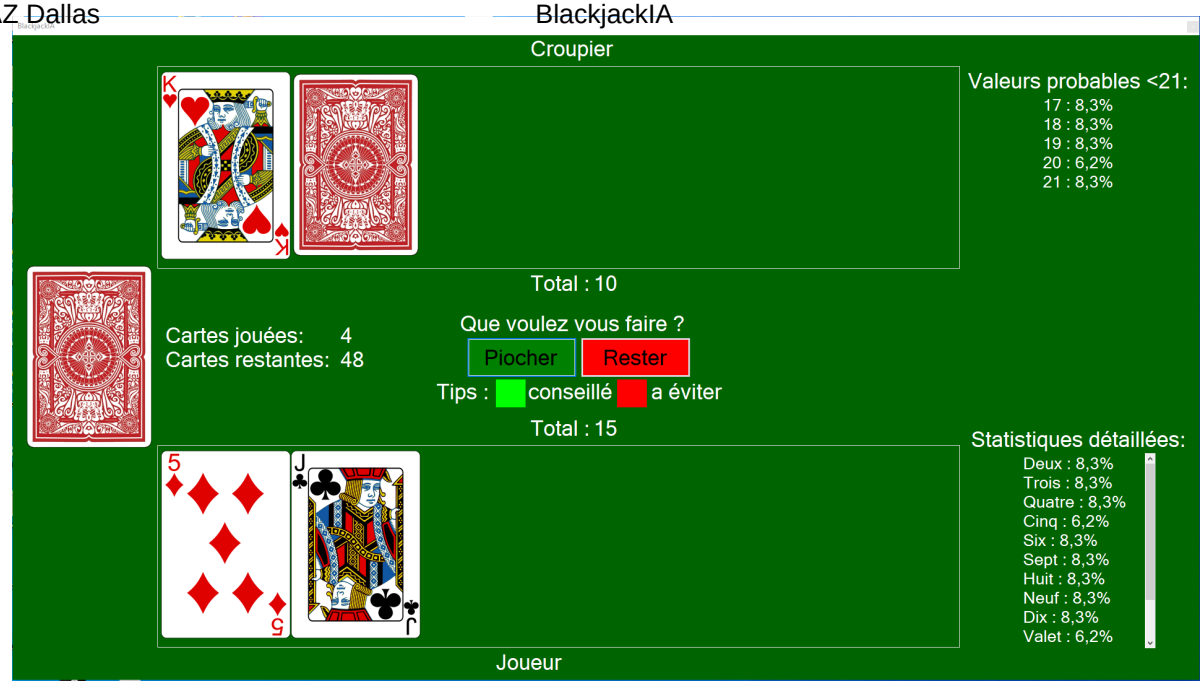
La vue paramètre est un formulaire où l'on choisit notre nom ainsi que le nombre de paquets avec lesquels on vas jouer.

Ces infos seront utilisées pour créer la vue principale.



Si le nom est vide ou uniquement composé d'espace on ne pourra pas appuyer sur OK.

4.2.3 La vue principale



C'est sur cette vue que le jeu se déroule. On peut y retrouver la main du joueur et du croupier ainsi que les boutons pour jouer qui changeront de couleur en fonction des conseils donnés. Conseils basés sur le tableau suivant que l'on peut retrouver à [cette adresse](#).

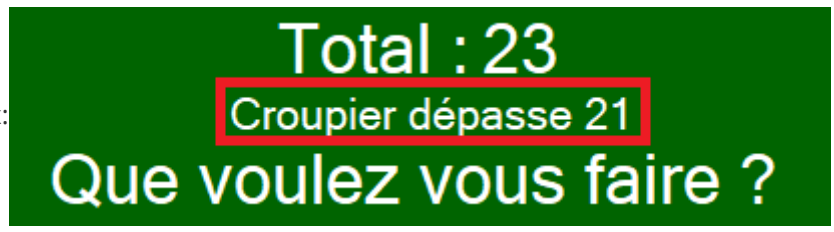
La stratégie de base en France par Monsieur G / Guide-blackjack.com

Joueur	Carte du croupier 2	3	4	5	6	7	8	9	10	A
21	R	R	R	R	R	R	R	R	R	R
20	R	R	R	R	R	R	R	R	R	R
19	R	R	R	R	R	R	R	R	R	R
18	R	R	R	R	R	R	R	R	R	R
17	R	R	R	R	R	R	R	R	R	Ab / R
16	R	R	R	R	R	T	T	Ab / T	Ab / T*	Ab / T
15	R	R	R	R	R	T	T	T	Ab / T	Ab / T
14	R	R	R	R	R	T	T	T	Ab / T	Ab / T
13	R	R	R	R	R	T	T	T	T	Ab / T
12	T	T	R	R	R	T	T	T	T	Ab / T
11	D	D	D	D	D	D	D	D	T	T
10	D	D	D	D	D	D	D	D	T	T
9	T	D	D	D	D	T	T	T	T	T
8	T	T	T	T	T	T	T	T	T	T
7	T	T	T	T	T	T	T	T	T	Ab / T
6	T	T	T	T	T	T	T	T	T	Ab / T
5	T	T	T	T	T	T	T	T	T	Ab / T

Comme l'application ne permet pas de miser des sommes les conditions ou il faut doubler sont remplacé par des simples tirages

Le joueur a juste besoin de choisir ce qu'il veut faire, le reste est géré automatiquement par l'application. Le résultat de la partie sera affiché dans un label au-dessus des boutons.

ex:



Chaque fois qu'une carte est piochée les statistiques sont mise à jour.

5. Analyse organique

Cette partie de la documentation décrit le fonctionnement interne de l'application. Il y sera listé les actions effectuées par l'application en background. Les actions du programme sont listées dans l'ordre d'exécution dans le cas d'une partie standard. Chaque classe de l'application y serra aussi détaillé.

5.1 Actions du programme

- Passer les paramètres de la vue paramètre au programme
- Créer un joueur avec le nom entré en paramètre
- Créer un paquet de carte avec le paramètre reçu
    - Créer 52 cartes * le nombre de paquet voulu
- Création du jeu avec le joueur, le paquet, la vue et une variable aléatoire
- Distribution des cartes
- Mise a jour de l'affichage
- Changement de couleurs des boutons
- Le joueur clique sur le bouton "Piocher"/"Rester"
- Le croupier pioche jusqu'à 17
- Comparaison des mains

5.2 Description des classes et méthodes

Cette section traitera du fonctionnement de toutes les fonctions utilisées dans le programme.

5.2.1 Classe Joueur

Constructeur

Le joueur est créé avec comme paramètres un nom et un groupBox qui représentera sa main.

Méthode Piocher

Le joueur peut piocher un nombre de cartes voulu d'un paquet passé en paramètre. Pour ce faire il choisit la première carte du paquet l'ajoute à sa main et la retire du paquet. Puis il ajoute la valeur de la carte à la valeur totale de sa main.

Elle fera appel à la méthode TestChangementValeurAs pour savoir s'il peut passer la valeur d'un as à 1 pour ne pas dépasser 21.
Pour finir elle fera appel à la méthode AfficherMain pour ajouter la nouvelle carte au groupe Box.

Méthode AfficherMain

Elle récupère toutes les cartes de la main, test si elles doivent être retournées si oui elle les retourne puis elle les ajoute au groupBox avec une position déterminée par leur index.

Méthode TestChangementValeurAs

Test si la valeur de notre main est >21 et qu'il y a bien un as avec comme valeur 11 dans la main. Si c'est le cas alors elle change la valeur du premier as dans notre main pour 1 puis elle recompte la valeur totale de notre main avec la méthode CalculerValeurDeLaMain.

Méthode CalculerValeurDeLaMain

Récupère toutes les valeurs des cartes dans la main et les additionnent.

5.2.2 Classe Croupier : Joueur

Constructeur

Prends un groupbox comme paramètre qui sera la main du croupier et utilise le constructeur de joueur en donnant comme nom "croupier" et comme groupbox celui actuellement en paramètres.

Méthode CacherSecondeCarte

Récupère la deuxième carte de la main du croupier et utilise la méthode Retourner dessus

5.2.3 Classe Carte : Picturebox

Constructeur

Il prend comme paramètre une valeur de l'enum Valeur (As, 2, 3 ...) ainsi qu'une couleur de l'enum Couleur (Coeur, Carreau ...). Puis il lui attribue sa valeur dans le jeu par rapport à son nom.

```
witch (valeur)
            {
                case Valeur.Deux:
                case Valeur.Trois:
                case Valeur.Quatre:
                case Valeur.Cinq:
                case Valeur.Six:
                case Valeur.Sept:
                case Valeur.Huit:
                case Valeur.Neuf:
                case Valeur.Dix:
                    ValeurDansJeu = (int)valeur + 2;
                    break;
                case Valeur.Valet:
                case Valeur.Dame:
                case Valeur.Roi:
                    ValeurDansJeu = 10;
                    break;
                case Valeur.As:
                    ValeurDansJeu = 11;
                    break;
            }
```

Comme l'objet est hérité de la classe Picturebox je lui donne aussi un nom composé du tostring de l'objet ainsi qu'une taille et une image dont le nom correspond au ToString.

Méthode Retourner

Inverse la valeur du bool "Retourne" et test sa valeur et change l'image par rapport au résultat obtenu.

Méthode ToString

Nous retourne l'objet sous la forme suivante :

```
return string.Format("_{0}{1}", CarteValeur, CarteCouleur);
```

5.2.4 Classe Paquet

Constructeur

Il prend en paramètre le nombre de paquets avec lesquels on veut jouer puis il nous répète la création du paquet ce nombre de fois.
Pour créer un paquet on parcourt les enum de couleur et de valeur et pour chaque combinaison a créé une carte correspondante puis on l'ajoute à une liste.

Méthode Mélanger

```
PaquetDuJeu = PaquetDuJeu.OrderBy(x => random.Next()).ToList();
```

On tri la liste de manière aléatoire.

5.2.5 Classe Tirage

Constructeur

Prends comme paramètre la valeur de la main du joueur et la valeur de la main du croupier.

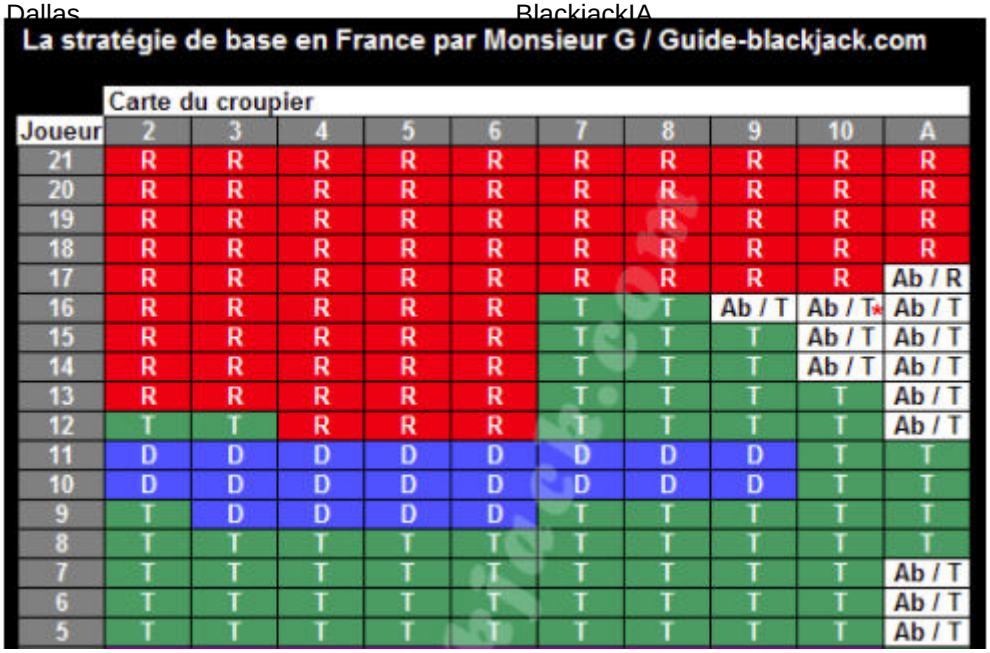
Cette classe sert à stocker des données pour pouvoir par la suite les utilisés comme key d'un dictionnaire.

5.2.6 Classe BlackjackIA

Constructeur

Prends comme paramètres un joueur, un paquet, une Form, et un Random.

Le constructeur mélangera le paquet directement le deck. Il créera le croupier et initialisera le tableau de probabilités sur lequel le conseiller se basera.



La stratégie de base en France par Monsieur G / Guide-blackjack.com

	Carte du croupier									
Joueur	2	3	4	5	6	7	8	9	10	A
21	R	R	R	R	R	R	R	R	R	R
20	R	R	R	R	R	R	R	R	R	R
19	R	R	R	R	R	R	R	R	R	R
18	R	R	R	R	R	R	R	R	R	R
17	R	R	R	R	R	R	R	R	R	Ab / R
16	R	R	R	R	R	T	T	Ab / T	Ab / T*	Ab / T
15	R	R	R	R	R	T	T	T	Ab / T	Ab / T
14	R	R	R	R	R	T	T	T	Ab / T	Ab / T
13	R	R	R	R	R	T	T	T	T	Ab / T
12	T	T	R	R	R	T	T	T	T	Ab / T
11	D	D	D	D	D	D	D	D	T	T
10	D	D	D	D	D	D	D	D	T	T
9	T	D	D	D	D	T	T	T	T	T
8	T	T	T	T	T	T	T	T	T	T
7	T	T	T	T	T	T	T	T	T	Ab / T
6	T	T	T	T	T	T	T	T	T	Ab / T
5	T	T	T	T	T	T	T	T	T	Ab / T

Pour créer ce tableau il parcourt une boucle de 2 à 11 et une autre de 5à 21.

Pour chacune des combinaisons il crée un objet tirage et si la valeur du joueur est supérieure à 17, ou que la valeur du joueur est supérieure ou égale à 13 et que celle du croupier est inférieure ou égale à 6, ou que la valeur du croupier est comprise entre 4 et 6 et que la valeur du joueur est égale à 12 alors la valeur dans notre dictionnaire pour la clé tirage serra égal a false, sinon elle sera true.

Méthode Distribution

Distribue au joueur et au croupier 2 cartes.

Puis il affiche la main du joueur et cache la seconde carte du croupier.

Méthode Comparaison

Effectue une série de tests pour savoir si le joueur ou le croupier ont une main trop grosse et si aucun des deux n'est au-dessus de 21 alors on détermine qui a la main la plus grande.

Méthode CroupierPiocheJusqua17

Parcours une boucle while qui tant que la main du joueur ne dépasse pas 17 fera pioche une carte au croupier.

Méthode CompterProbaTirageCarte

```
            probaTirageCarte.Clear();
            foreach (Carte.Valeur valeur in
(Carte.Valeur[])Enum.GetValues(typeof(Carte.Valeur)))
            {
                double nbrCarte = 0;
                foreach (var carte in Paquet.PaquetDuJeu)
                {
                    if (carte.CarteValeur == valeur)
                    {
                        nbrCarte++;
                    }
                }
```

```
                double proba = Math.Round(nbrCarte / Paquet.PaquetDuJeu.Count(),
3) * 100;
                ProbaTirageCarte.Add(valeur, proba);
            }
```

Vide la liste des probabilités d'avoir une certaine carte puis la re-rempli.

6. Fonctionnalités restantes a implémenter et améliorations possibles

Certaines fonctionnalités de l'application n'ont pas pu être implémentées dans le temps imparti mais elles ne sont pas cruciales au fonctionnement de l'application telle que :

- Les statistiques comptant l'As comme 1 et 11
- La fin du jeu avec impossibilité de re-pioché et possibilité de relancer directement une partie

Concernant les possibles ajouts de l'application plusieurs point peuvent être aborder :

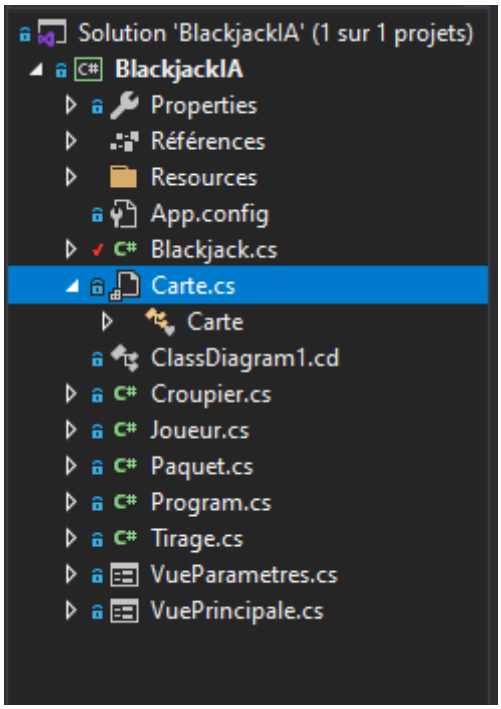
- Ajouter le système de mise
- Ajouter la règle du Blackjack (As + Face a la distribution)
- Possibilité de splitter/ partager sa main
- Possibilité de doubler sa mise
- Intégrer le système d'assurance
- Intégrer le système d'abandon

7. Réalisation

7.1 Arborescence du projet

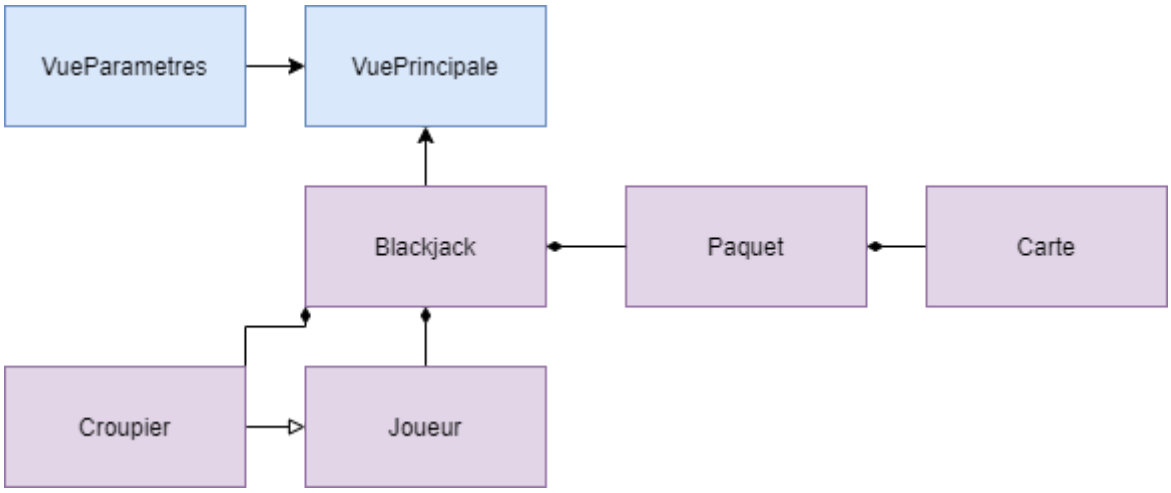
Le programme se trouve dans une solution qui continent un seul projet. Ce projet contient en tout 2 vues et 7 classes.

Voici un aperçu de l'arborescence du projet :

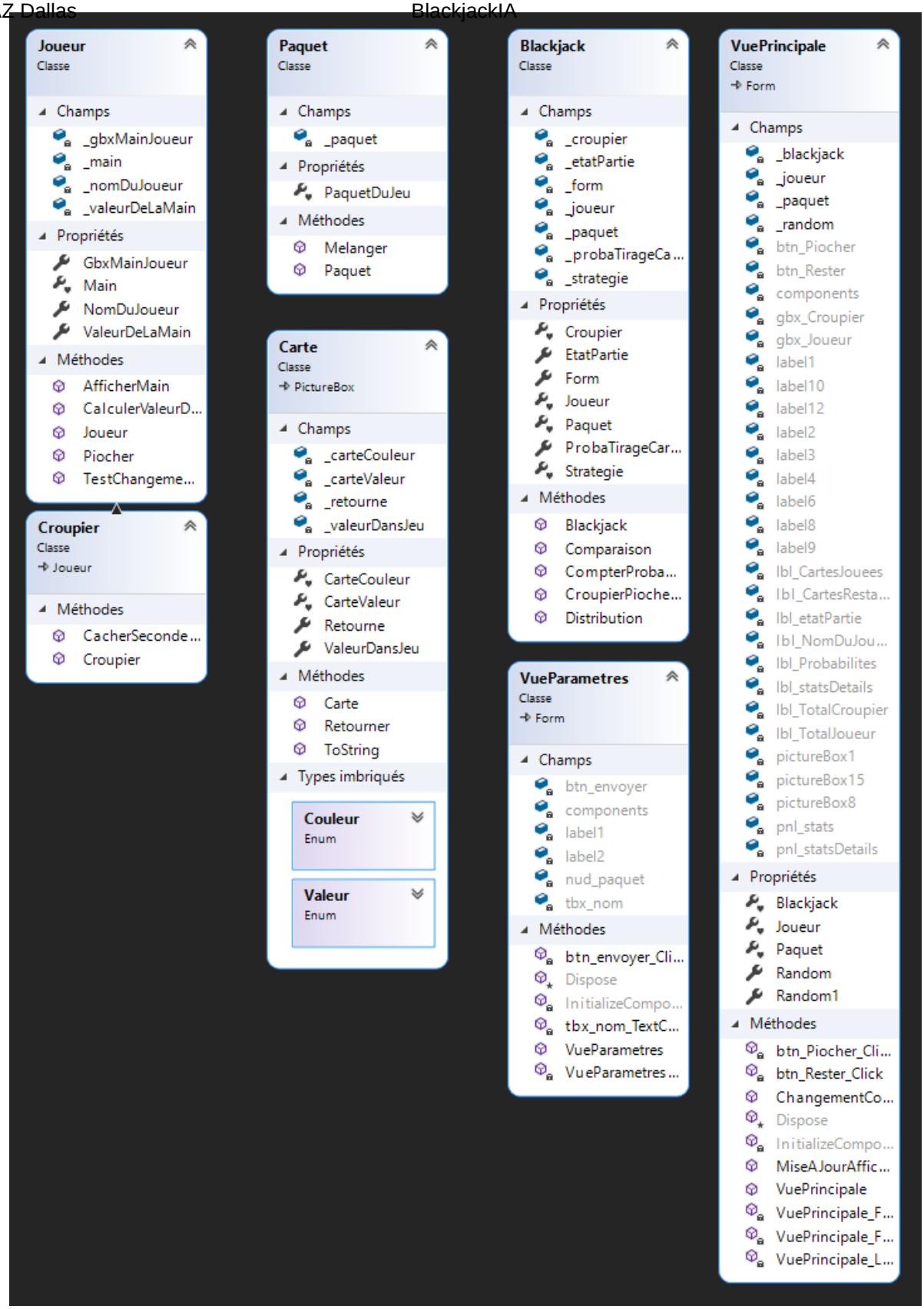


7.2 Diagramme de classe

7.2.1 Diagramme de classe global



7.2.2 Diagramme de classe de l'application



7.3 Plan de test

Ces test ont été effectué sur la dernière version de l'application le 18.05.21.

N°	Description du test	Résultat attendu	YES/NO
1	Lancer l'application	La vue paramètres est affiché	YES
2	Entrer "Joueur" dans la tbx de la vue paramètres	Le bouton ok est activé	YES
3	Appui sur le bouton OK de la vue paramètres	Les données du formulaire sont transmise a la vue principale et la vue paramètres est fermée	YES
4	La vue principale affiche le nom du joueur choisi au préalable donc "Joueur"	Le nom affiché est bien "Joueur"	YES
5	Le nombre de paquet choisi est a 2	Le nombre de carte totale correspond bien a 104	YES
6	Le joueur a une main de 19 et le croupier une main de 5 durant le tour du joueur	Le bouton Rester est en vert et le bouton piocher en rouge	YES
7	Une carte est pioché du paquet d'un paquet contenant 100 cartes restantes	Le nombre de carte jouées est augmenté de 1, le nombre de cartes restantes est réduite de 1 et les statistiques changent en fonction de la carte piochée	YES
8	Les cartes sont correctement distribué	Chaque joueur commence la partie avec 2 cartes	YES
9	Les cartes du croupier sont distribué	La deuxième carte du croupier n'est pas visible et n'est pas prise en compte pour le compte de la valeur de sa main	YES
10	La carte caché du croupier est un As	Les statistiques n'indiquent pas qu'un As manque dans le jeu.	NO
11	Le joueur appuis sur le bouton rester	Le croupier pioche des cartes jusqu'à être au moins a 17 et le résultat de la partie est affiché	YES
12	Les valeurs probables pour une main de 19	Les valeurs probables affiché sont 20 et 21.	NO
13	La main du joueur avec comme nom "Joueur" est supérieur a 21	Le croupier pioche jusqu'à avoir une main de 17 et le message "Joueur dépasse 21" est affiché	YES
14	Le croupier a une main supérieur a celle du joueur	Le message "Croupier gagne" est affiché	YES

N°	Description du test	Résultat attendu	YES/NO
15	Le joueur a un 4 de cœur et un 3 de pique dans la main provenant d'un seul paquet	La valeur totale de la main du joueur est 7, les statistiques de probabilité vont de 8 a 18 et les statistiques détaillées pour le 4 sont respectivement de 6,2% et 6.2%.	NO
16	Le joueur pioche alors que la partie est terminée	Aucune carte n'est ajouté a sa main	NO
17	La valeur de la main du joueur et du croupier sont identiques	Le message "égalité" est affiché	YES
18	La main du joueur contient un As, un 6 et est donc égal a 17 puis il pioche un roi	Le premier As de la main du joueur change sa valeur pour 1 et la valeur totale de la main du joueur devient reste 17	YES

Tests non concluants

10 : Si l'on regard attentivement les statistiques on peut savoir que la carte caché par le croupier est un As

12: la seule valeur probable affiché est 21

15: les statistiques de probabilité vont de 7 a 18

16: on peut encore pioché après la fin du jeu

8. Conclusion

Ce projet a été réalisable grâce à la somme de connaissances accumulées tout au long de ma formation d'informaticienne au CFPT-I. Même si le travail n'est pas totalement terminé il reste néanmoins fonctionnel et intéressant pour apprendre les bases du Blackjack ainsi que la stratégie a appliqué.

Le travail qui m'a été demandé était de faire un jeu de Blackjack auquel on pourrait jouer contre un croupier contrôler par un IA tout en ayant l'aide d'un conseiller qui nous indique qu'elle est la meilleur solution en fonction de la situation. Le programme devait aussi être accompagné d'un affichage qui nous indiquait la probabilité de succès en tirant la prochaine carte, le nombre de carte restante dans le paquet ainsi que le nombre de carte jouées et la valeur des cartes dans le jeu. Le jeu nous permettrais aussi de choisir le nombre de paquet avec lequel nous allions joué.

Au final j'ai pu faire en sorte que toute les fonctionnalités qui étaient demandé soit fonctionnel.

Concernant le cahier des charges il ne me reste donc plus rien a faire.

8.1 Bilan personnel

Ce TPI m'a donné l'occasion de développer un projet complet dans des conditions professionnelles et
réelles de travail. Il m'a permis de me rendre compte des enjeux et des difficultés rencontrées lors de
ce genre de développement en autonomie. J'apprécie beaucoup le jeu du Blackjack, c'est pourquoi j'ai été heureux d'apprendre qu'il s'agissait de mon sujet de TPI.

La principale difficulté rencontrée durant le développement de mon application a été le temps imparti. Bien que je n'aie pas eu énormément de difficulté à rendre fonctionnelles les fonctionnalités demandées, je sens que leur application peut être significativement améliorée. Ceci est majoritairement dû au peu de temps qui était imparti. De plus la gestion du temps concernant la documentation à aussi une des choses pénalisante.

Dans tous les cas je trouve quand même que le travail fournit correspond aux attentes et je suis fière du résultat final.
Je ne manquerais pas de mettre à profit les connaissances que ce projet m'a apportées dans ma vie professionnelle future et je pense que cela me sera d'une grande utilité.

Annexes

Planning prévisionnel

Jour	1		2		3		4		5		6		7		8		9		10		11	
Demi-Journée	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
Etude du sujet. Planification																						
Installation, découpage du jeu																						
Classes Deck52 et carte																						
Distribution, interface graphique																						
Développement de l'IA																						
Statistiques temps réel																						
Finalisation / Corrections																						
Tests																						
Documentation																						
Résumé																						
Finalisation / Impressions																						
Journal de bord																						

Planning réel