

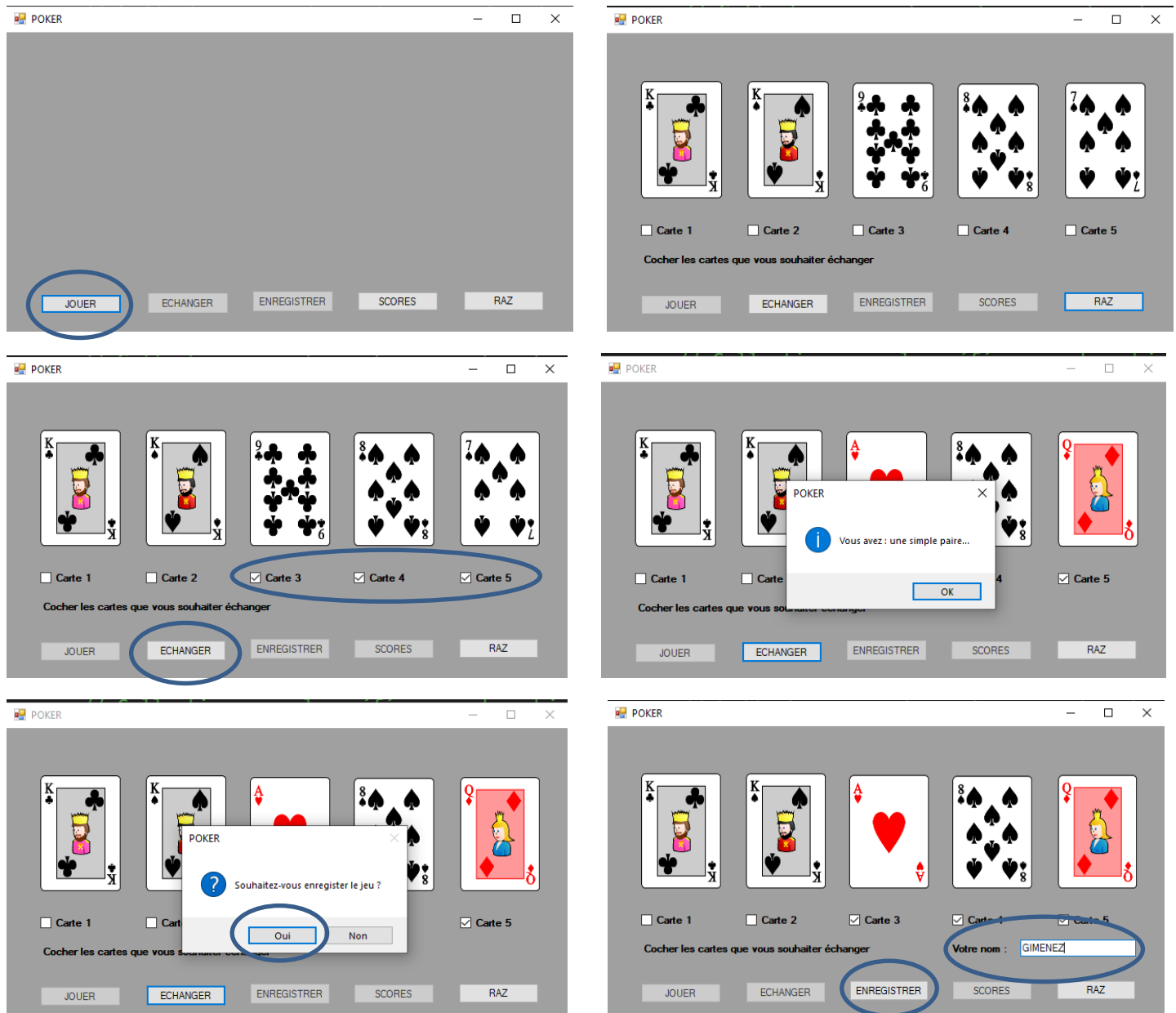
Présentation

Le succès rencontré avec la version en « mode Console » du jeu de Poker précédemment développé, a conduit le Comité d'Entreprise du groupe HsH vous a sollicité une nouvelle fois afin de faire évoluer le jeu vers une interface graphique en Windows Forms.

Votre chef de projet a réalisé une analyse de la future application. Cette analyse concerne à la fois, l'ergonomie de la nouvelle interface et des éléments techniques à prendre en compte, en particulier :

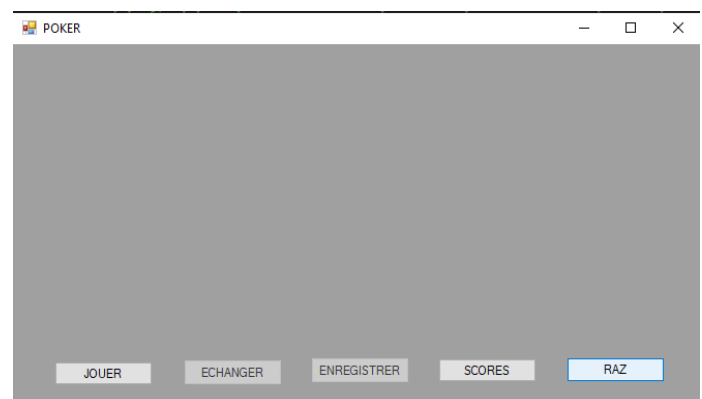
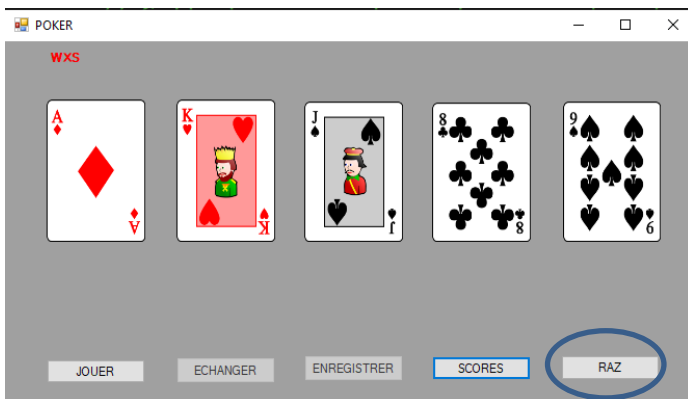
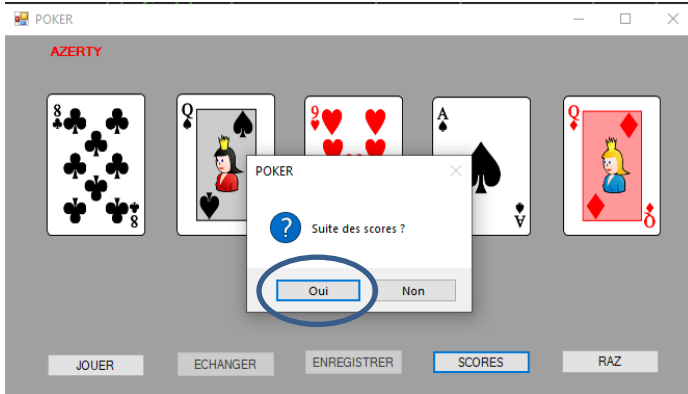
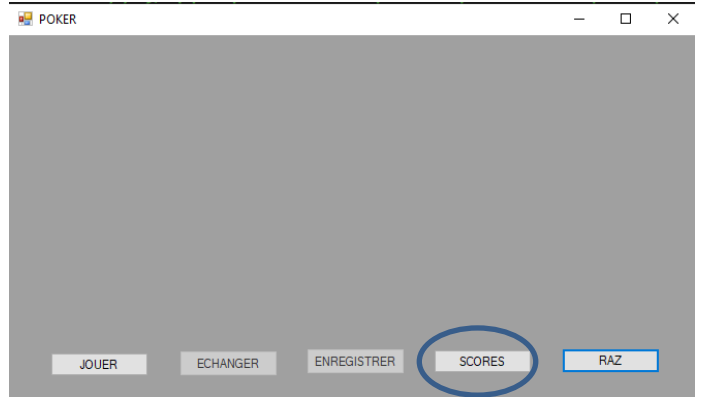
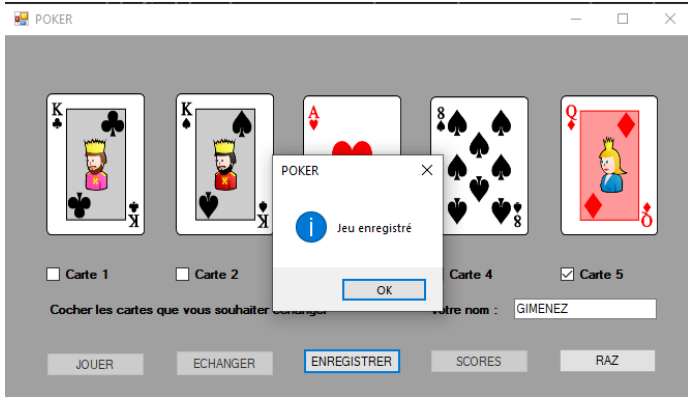
- la mise en œuvre de deux classes METIER : la classe Carte et la classe Jeu ;
- la mise en œuvre d'une classe technique de gestion du FICHIER des scores ;
- l'architecture de l'application : séparation de la classe de gestion de l'interface graphique et des classes métier et technique.

Interface graphique



The screenshots illustrate the user interface of the POKER application, showing the progression of a game session:

- Initial Screen:** Displays five cards (K♣, K♠, 9♣, 8♠, 7♠) and buttons for JOUER, ECHANGER, ENREGISTRER, SCORES, and RAZ. The JOUER button is highlighted.
- Card Selection:** The user selects cards to exchange (Carte 3, 4, 5). The ECHANGER button is highlighted.
- Exchange Confirmation:** A dialog box asks "Souhaitez-vous enregistrer le jeu ?" with "Oui" and "Non" buttons. The "Oui" button is highlighted.
- Registration:** The user enters their name (GIMENEZ) and clicks the ENREGISTRER button.
- Game Result:** A dialog box displays the result: "Vous avez : une simple paire..." with an OK button.
- Final Screen:** Shows the updated cards (K♣, K♠, A♥, 8♠, Q♦) and the ENREGISTRER button.



Classe Métier JEU

Détail :

```
#region ATTRIBUTS_ACCESSEURS_ET_CONSTRUCTEUR

// Jeu de 5 cartes
private Carte[] lesCartes;

// Liste des combinaisons possibles
20 références
private enum combinaison { RIEN, PAIRE, DOUBLE_PAIRE, BRELAN, QUINTE, FULL, COULEUR, CARRE, QUINTE_FLUSH };

// Accesseur sur le jeu de 5 cartes
1 référence
public Carte[] getlesCartes()...

// Constructeur
1 référence
public Jeu()...

#endregion
```

```
#region METHODES PUBLIQUES

// TIRAGE DU JEU
//     Retourne le jeu après le tirage des 5 cartes
1 référence
public Carte[] tirageDuJeu()...

// ECHANGE DES CARTES
//     Paramètre : un tableau avec les numéros des cartes à échanger
//     Retourne : le jeu après le retrait des cartes demandées
1 référence
public Carte[] echangeDeCartes(int[] echange)...

// AFFICHE LE RÉSULTAT DU JEU
//     Retourne : un objet string avec le message correspondant au résultat
//     (Ex : "un brelan; pas mal...")
1 référence
public string calculeResultat()...

#endregion
```

```
#region METHODES PRIVEES

// MÉTHODE DE CALCUL ALÉATOIRE D'UNE CARTE {famille ; valeur}
//     Retour : un objet CARTE
//     Utilisée par "tirageDuJeu()"
2 références
private Carte tirageUneCarte()...

// MÉTHODE PERMETTANT DE VÉRIFIER QU'UNE CARTE NE FIGURE PAS DANS LE JEU
//     Retour : un booléen avec le résultat de la recherche
//     Utilisée par "tirageDuJeu()"
2 références
private bool carteUnique(Carte uneCarte, int numero)...

// CALCULE ET RETOURNE LA COMBINAISON POUR UN JEU COMPLET DE 5 CARTES
//     Retour : un élément de l'énumération 'combinaison'
//     Utilisée par "calculeResultat()"
1 référence
private combinaison cherche_combinaison()...

#endregion
```

Classe Métier CARTE

```
public class Carte
{
    // Attributs
    private char valeur;
    private int famille;

    // Constructeur
    2 références
    public Carte()
    {
        // Valeurs des cartes : As, Roi,...
        char[] valeurs = { 'A', 'R', 'D', 'V', 'X', '9', '8', '7' };

        // Codes ASCII (3 : coeur, 4 : carreau, 5 : trèfle, 6 : pique)
        int[] familles = { 3, 4, 5, 6 };

        // Tirage aléatoire pour déterminer la valeur et la famille
        Random r = new Random(Guid.NewGuid().GetHashCode());

        // Valeur aléatoire (1 sur les 8). Attention : borne supérieure non comprise
        this.valeur = valeurs[r.Next(0, 8)];
        // Famille aléatoire (1 sur les 4)
        this.famille = familles[r.Next(0, 4)];
    }

    // Propriétés
    16 références
    public char LaValeur...

    22 références
    public int LaFamille...
}
```

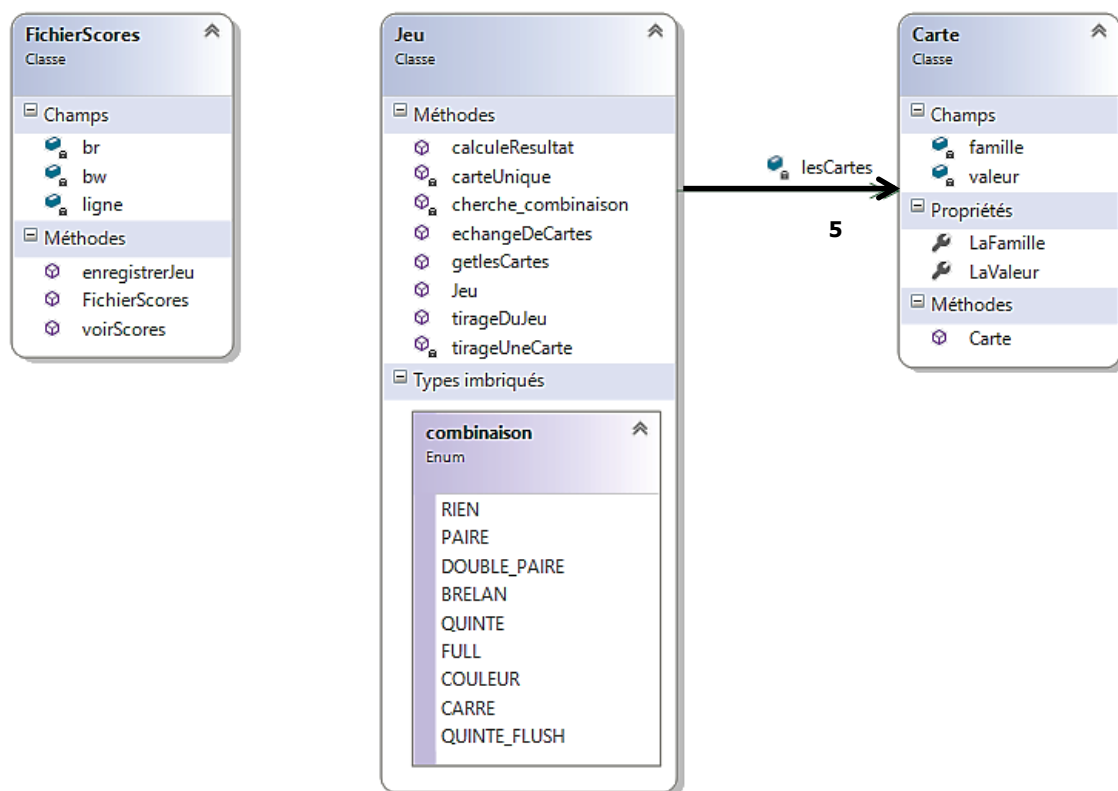
Classe TECHNIQUE de gestion du FICHIER des scores.

Suggestion :

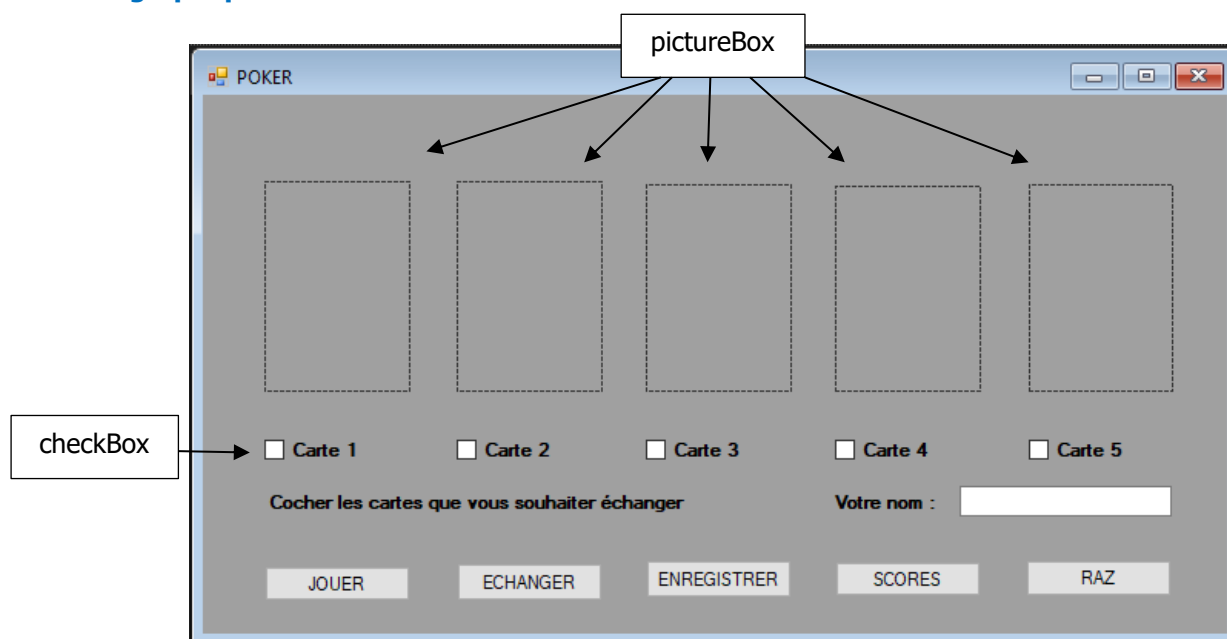
- **Attributs** : objets des classes .NET permettant la lecture et l'écriture dans le fichier.
- **Méthodes** : enregistrerJeu() et voirScores()
Les paramètres et le type du retour sont laissés à votre discrétion, en fonction des choix effectués lors du développement de la version Console.

REMARQUE : vous êtes libres de décider si les méthodes seront ou non « statiques » (= « à portée de classe ») ou non.

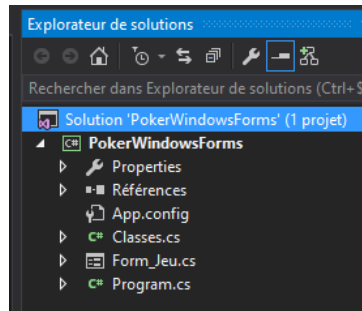
Diagramme des classes



Interface graphique



Architecture de la solution



- Les classes METIER et la classe TECHNIQUE figurent dans le fichier « Classes.cs »
- Le fichier « Form_Jeu.cs » est uniquement dédié à la gestion de l'interface graphique et à l'utilisation des classes métier et de la classe technique.

Classe correspondant au formulaire (« Form ») du jeu

```
public partial class Form_Jeu : Form
{
    // Attributs
    private Jeu leJeu;           // Classe métier Jeu
    private FichierScores f;     // Classe technique de gestion du fichier des scores
    private List<PictureBox> lesImages; // Collection avec les références des objets pictureBox
    private List<CheckBox> lesCases; // Collection avec les références des objets checkBox

    // Constructeur
    1 référence
    public Form_Jeu()...

    // RAZ
    5 références
    private void RAZ()...

    // AFFICHER un JEU de 5 cartes
    3 références
    public void afficher_Jeu(Carte[] leJeu)...

    // JOUER
    1 référence
    private void button1_Click(object sender, EventArgs e)...

    // ECHANGER
    1 référence
    private void button2_Click(object sender, EventArgs e)...

    // ENREGISTER
    1 référence
    private void button3_Click(object sender, EventArgs e)...

    // SCORES
    1 référence
    private void button4_Click(object sender, EventArgs e)...

    // RAZ de l'ensemble
    1 référence
    private void button5_Click(object sender, EventArgs e)...
}
```

Ressources

Le dossier « Cartes » contient l'ensemble des cartes du jeu au format « jpg »

