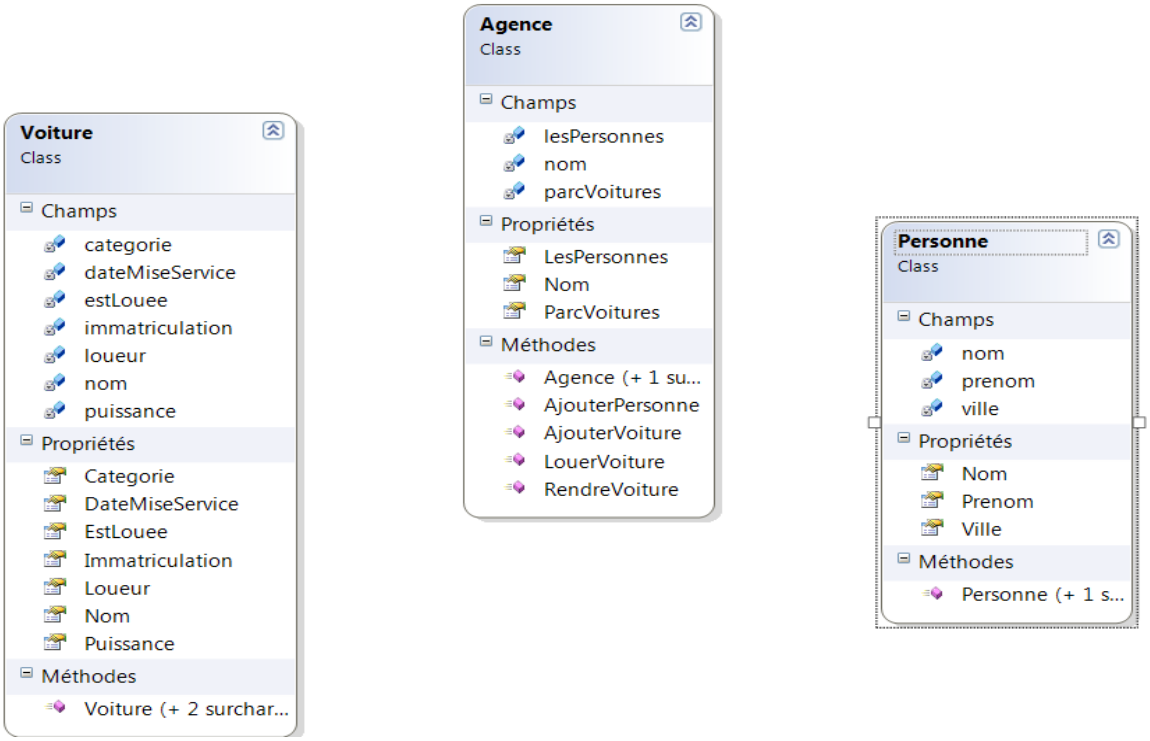
Gestion d'un parc automobile

Sujet :
Une agence gère les locations des voitures à des personnes. Elle dispose d'un par de voitures que ses clients peuvent emprunter.
Dans une première version, l'application ne gère pas :

- les dates d'emprunt
- le retour des voitures
- l'historique des locations.

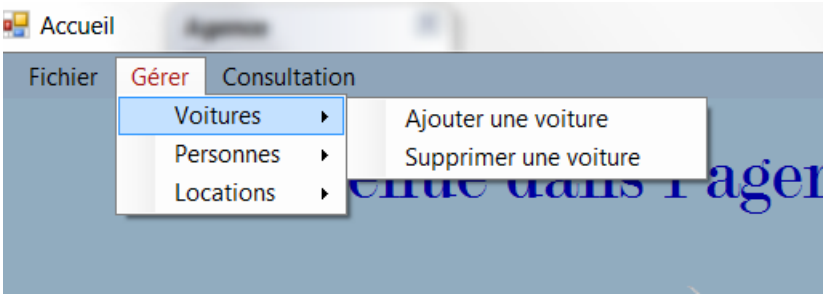
Une première analyse a mis en évidence le diagramme de classes suivant. Les liens sont matérialisés par les collections :

- lesPersonnes et parcVoitures

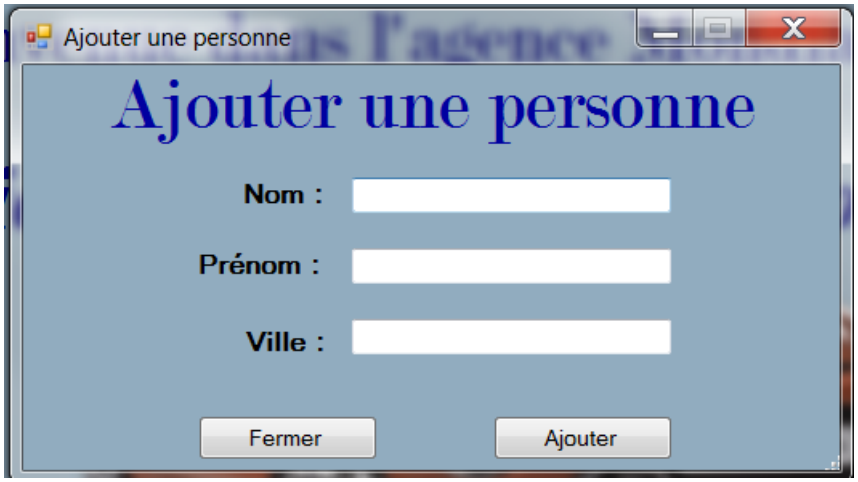


Les résultats attendus de l'application sont :

- Gestion



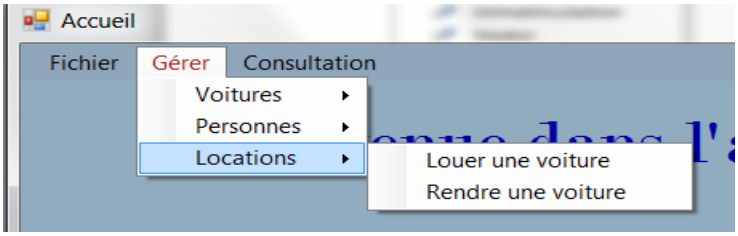
Exemple : ajout d'une personne



- Consultation

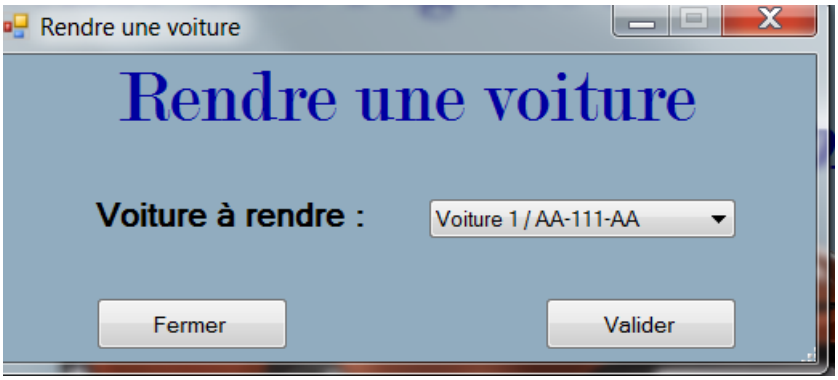
Exemple : consultation des voitures

- Location d'une voiture



Exemple : Location

Restitution



Sauvegarde – Restitution des données

La sauvegarde des données va se faire dans un fichier XML. On vous donne la procédure de sérialisation /désérialisation

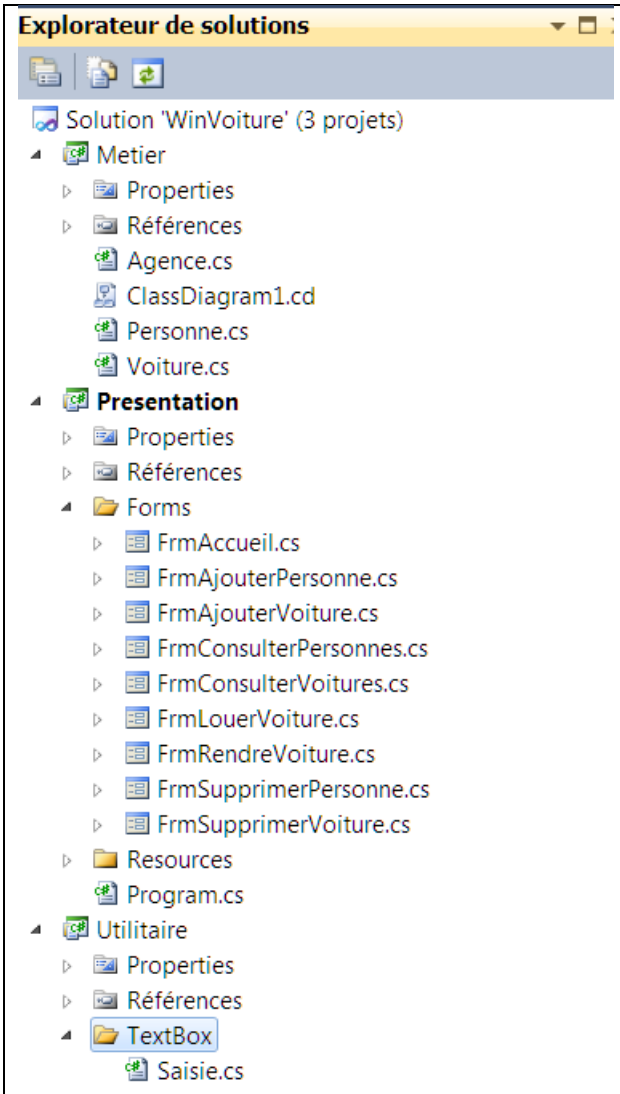
```
#region Sérialisation / Désérialisation Binaire

        /// <summary>
        /// Méthode de désérialisation binaire
        /// </summary>
        /// <param name="sender">sender</param>
        /// <param name="e">e</param>
        private void importerBinaireToolStripMenuItem_Click(object sender, EventArgs e)
        {
            //On affiche une fenêtre permettant de choisir un fichier à charger
            OpenFileDialog FileDialogBin = new OpenFileDialog();
            FileDialogBin.Title = "Choisissez un fichier";
            //On fixe l'extension à .ci
            FileDialogBin.Filter = "Fichiers CI (*.ci)|*.ci";
            DialogResult result = FileDialogBin.ShowDialog();

            //Si l'utilisateur clique sur "Charger"
            if (result == DialogResult.OK)
            {
                String cheminComplet = "";
                String nomFichier = "";
                String chemin = "";

                //On récupère le chemin complet, le nom du fichier et le chemin du dossier
contenant le fichier à charger
                cheminComplet = FileDialogBin.FileName;
                cheminComplet = cheminComplet.Replace("\\", "\\\\");
```

```
                //On récupère le nom du fichier qui se trouve en fin du chemin complet
                nomFichier = cheminComplet.Substring(cheminComplet.LastIndexOf("\\\\") + 2,
cheminComplet.Length - cheminComplet.LastIndexOf("\\\\") - 2);
                //On récupère le chemin du dossier qui contient le fichier ci-dessus
(l'autre partie du chemin complet)
                chemin = cheminComplet.Substring(0, cheminComplet.Length -
(nomFichier.Length + 2));

                FileStream unFlux = null;
                BinaryFormatter fs;

                try
                {
                    //On ouvre  un flux binaire
                    Directory.SetCurrentDirectory(chemin);
                    unFlux = new FileStream(nomFichier, FileMode.Open);
                    //On formate le flux en binaire
                    fs = new BinaryFormatter();
                    //On vide la collection avant de récupérer le contenu désérialiser
                    uneAgence = (Agence)fs.Deserialize(unFlux);
                    //On affiche un message indiquant le succès de la désérialisation
                    MessageBox.Show("La désérialisation s'est terminée avec succès !",
"Désérialisation finie", MessageBoxButtons.OK, MessageBoxIcon.Information);
                }
                catch (Exception ex)
                {
                    MessageBox.Show("\n" + ex.Message, "Erreur", MessageBoxButtons.OK,
MessageBoxIcon.Error);
                }

                finally
                {
                    unFlux.Close();
                }
            }
        }



        /// <summary>
        /// Méthode de sérialisation binaire
        /// </summary>
        /// <param name="sender">sender</param>
        /// <param name="e">e</param>
        private void exporterBinaireToolStripMenuItem_Click(object sender, EventArgs e)
        {
            //On affiche une fenêtre permettant de saisir un fichier dans lequel sauvegarder
            SaveFileDialog FileDialogBin = new SaveFileDialog();
            FileDialogBin.Title = "Saisissez un fichier";
            //On fixe l'extension à .ci
            FileDialogBin.Filter = "Fichiers CI (*.ci)|*.ci";
            DialogResult result = FileDialogBin.ShowDialog();

            //Si l'utilisateur clique sur "Enregistrer"
            if (result == DialogResult.OK)
            {
                String cheminComplet = "";
                String nomFichier = "";
                String chemin = "";

                //On récupère le chemin complet choisi dans lequel on remplace \ par \\
                cheminComplet = FileDialogBin.FileName;
                cheminComplet = cheminComplet.Replace("\\", "\\\\");

                //On récupère le nom du fichier qui se trouve en fin du chemin complet
```

```
                nomFichier = cheminComplet.Substring(cheminComplet.LastIndexOf("\\\\") + 2,
cheminComplet.Length - cheminComplet.LastIndexOf("\\\\") - 2);
                //On récupère le chemin du dossier qui contient le fichier ci-dessus
(l'autre partie du chemin complet)
                chemin = cheminComplet.Substring(0, cheminComplet.Length -
(nomFichier.Length + 2));

                //On crée un flux binaire de sortie
                FileStream unFlux = null;
                BinaryFormatter fs;
                String nom_fichier = nomFichier;

                try
                {
                    Directory.SetCurrentDirectory(chemin);
                    unFlux = new FileStream(nomFichier, FileMode.Create);
                    //On formate le flux en binaire
                    fs = new BinaryFormatter();
                    fs.Serialize(unFlux, uneAgence);
                    //S'il n'y a pas eu d'erreurs on affiche un message qui en informe
l'utilisateur
                    MessageBox.Show("La sérialisation s'est terminée avec succès !",
"Sérialisation finie", MessageBoxButtons.OK, MessageBoxIcon.Information);
                }
                catch (Exception ex)
                {
                    MessageBox.Show("/n" + ex.Message, "Erreur", MessageBoxButtons.OK,
MessageBoxIcon.Error);
                }

                finally
                {
                    unFlux.Close();
                }
            }
        }

        #endregion
        #region Sérialisation / Désérialisation XML

        /// <summary>
        /// Méthode de désérialisation XML
        /// </summary>
        /// <param name="sender">sender</param>
        /// <param name="e">e</param>
        private void importerXMLToolStripMenuItem_Click(object sender, EventArgs e)
        {
            OpenFileDialog fileDialogXML = new OpenFileDialog();
            fileDialogXML.Title = "Choisissez un fichier";
            fileDialogXML.Filter = "Fichiers XML (*.xml) | *.xml";
            DialogResult result = fileDialogXML.ShowDialog();
            if (result == DialogResult.OK)
            {
                string cheminComplet = "";
                string nomFichier = "";
                string chemin = "";

                cheminComplet = fileDialogXML.FileName;
                cheminComplet = cheminComplet.Replace("\\", "\\\\");
                nomFichier = cheminComplet.Substring(cheminComplet.LastIndexOf("\\\\") + 2,
cheminComplet.Length - cheminComplet.LastIndexOf("\\\\") - 2);
                chemin = cheminComplet.Substring(0, cheminComplet.Length -
(nomFichier.Length + 2));
                FileStream stream = null;
                XmlSerializer serializer;
```

```
                Directory.SetCurrentDirectory(chemin);
                if (File.Exists(nomFichier))
                {
                    try
                    {
                        stream = new FileStream(nomFichier, FileMode.Open, FileAccess.Read);
                        serializer = new XmlSerializer(typeof(Agence));
                        uneAgence = (Agence)serializer.Deserialize(stream);
                        stream.Close();
                        MessageBox.Show("La désérialisation s'est terminée avec succès !",
"Désérialisation finie", MessageBoxButtons.OK, MessageBoxIcon.Information);
                    }

                    catch (Exception ex)
                    {
                        MessageBox.Show("\n" + ex.Message, "Erreur", MessageBoxButtons.OK,
MessageBoxIcon.Error);
                    }

                    finally
                    {
                        stream.Close();
                    }
                }
            }
        }

        /// <summary>
        /// Méthode de désérialisation XML
        /// </summary>
        /// <param name="sender">sender</param>
        /// <param name="e">e</param>
        private void exporterXMLToolStripMenuItem_Click(object sender, EventArgs e)
        {
            SaveFileDialog fileDialogXML = new SaveFileDialog();
            fileDialogXML.Title = "Saisissez un fichier";
            fileDialogXML.Filter = "Fichiers XML (*.xml) | *.xml";
            DialogResult result = fileDialogXML.ShowDialog();
            if (result == DialogResult.OK)
            {
                string cheminComplet = "";
                string nomFichier = "";
                string chemin = "";

                cheminComplet = fileDialogXML.FileName;
                cheminComplet = cheminComplet.Replace("\\", "\\\\");
                nomFichier = cheminComplet.Substring(cheminComplet.LastIndexOf("\\\\") + 2,
cheminComplet.Length - cheminComplet.LastIndexOf("\\\\") - 2);
                chemin = cheminComplet.Substring(0, cheminComplet.Length -
(nomFichier.Length + 2));
                FileStream stream = null;
                XmlSerializer serializer;
                try
                {
                    Directory.SetCurrentDirectory(chemin);
                    stream = new FileStream(nomFichier, FileMode.Create);
                    serializer = new XmlSerializer(typeof(Agence));
                    serializer.Serialize(stream, uneAgence);
                    MessageBox.Show("La sérialisation s'est terminée avec succès !",
"Sérialisation finie", MessageBoxButtons.OK, MessageBoxIcon.Information);
                }

                catch (Exception ex)
                {
```

```
                    MessageBox.Show("\n" + ex.Message, "Erreur", MessageBoxButtons.OK,
MessageBoxIcon.Error);
                }

                finally
                {
                    stream.Close();
                }
            }
        }

        #endregion
```



L'application doit être développée en respectant l'architecture MVC et le principe des exceptions

La bibliothèque des classes Metier représente les classes métier de l'application

Presentation représente la couche de interface avec ses fenêtres et le code des classes associées.

Utilitaire est une bibliothèque de classes qui regroupe diverses fonctions qui seront utiles à l'application
Exemple Textbox
Représente des méthodes de contrôle des saisies de l'utilisateur.

Annexe : Classe Agence version XML

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Xml.Serialization;
using System.Runtime.Serialization.Json;
using System.Runtime.Serialization;

namespace Domaine
{
    /// <summary>
    /// Classe Agence
    /// </summary>
    [DataContract]
    [Serializable]
    [XmlRoot("Agence", Namespace = "", IsNullable = false)]
    public class Agence
    {
        #region Attributs

        private string nom;
        private List<Voiture> parcVoitures;
        private List<Personne> lesPersonnes;

        #endregion
        #region Getters And Setters

        [DataMember]
        [XmlElement("Nom")]
        public string Nom
        {
            get { return nom; }
            set { nom = value; }
        }

        [DataMember]
        [XmlArray("ParcVoiture")]
        [XmlArrayItem("Voiture", typeof(Voiture))]
        public List<Voiture> ParcVoitures
        {
            get { return parcVoitures; }
            set { parcVoitures = value; }
        }

        [DataMember]
        [XmlArray("LesPersonnes")]
        [XmlArrayItem("Personne", typeof(Personne))]
        public List<Personne> LesPersonnes
        {
            get { return lesPersonnes; }
            set { lesPersonnes = value; }
        }

        #endregion
        #region Constructeurs

        /// <summary>
        /// Constructeur Agence
        /// </summary>
        /// <param name="nom">Le nom de l'agence</param>
        public Agence(string nom)
        {
            this.nom = nom;
```

```
            parcVoitures = new List<Voiture>();
            lesPersonnes = new List<Personne>();
        }

        /// <summary>
        /// Constructeur par défaut
        /// </summary>
        public Agence()
            : this("Nom par default")
        {
        }

        #endregion
        #region Méthodes public

        /// <summary>
        /// Ajoute une voiture à l'agence
        /// </summary>
        /// <param name="voiture">La voiture</param>
        public void AjouterVoiture(Voiture voiture)
        {
            parcVoitures.Add(voiture);
        }

        /// <summary>
        /// Ajoute une personne à l'agence
        /// </summary>
        /// <param name="personne">La personne</param>
        public void AjouterPersonne(Personne personne)
        {
            lesPersonnes.Add(personne);
        }

        /// <summary>
        /// Loue une voiture
        /// </summary>
        /// <param name="voiture">La voiture à louée</param>
        /// <param name="personne">La personne qui loue la voiture</param>
        public void LouerVoiture(Voiture voiture, Personne personne)
        {
            voiture.Loueur = personne;
            voiture.EstLouee = true;
        }

        /// <summary>
        /// Rend une voiture
        /// </summary>
        /// <param name="voiture">La voiture rendue</param>
        public void RendreVoiture(Voiture voiture)
        {
            voiture.Loueur = null;
            voiture.EstLouee = false;
        }

        #endregion
    }
}
```

Travail à faire :

Vous devez implémenter les fonctionnalités présentées et rajouter (si vous avez le temps) la gestion des dates.
Ce travail peut être fait en binôme en utilisant le serveur TFS.